# StresStimulus v4.6

## User Guide

# TABLE OF CONTENTS

## Search the User Guide section

# 1 INTRODUCTION

StresStimulus is a load testing tool for websites, web services, web and mobile applications (collectively called websites). It is designed to be an essential component in the companies' web performance strategy, from development to maintenance and capacity planning. StresStimulus helps answer many performance related questions such as:

- Does a Web application's responsiveness correspond to the target performance objectives?
- How many users can a website handle while maintaining acceptable responsiveness?
- What is a website's operational ceiling and how does it behave once it is overloaded (e.g. presents graceful error messages or refuses connections and crashes)?
- What are the performance bottlenecks that should be addressed first?
- What is the impact of new application releases on the response times?
- How many hardware resources are necessary to meet certain performance targets?

As opposed to other load testing tools, StresStimulus's focus is on ease-of-use with websites of any size and complexity.

## 1.1 How It Works

To make load testing more realistic, in order to test a website, StresStimulus records client actions and server responses. After that, you can specify what performance conditions you need to emulate and launch a test. StresStimulus instantiates multiple virtual users (VUs) and replays the recorded actions. The web server receives simulated traffic and engages in operations as it is accessed by the real users. While interacting with the server, StresStimulus closely monitors performance metrics from the user's perspective, but on the server side. It displays the test real-time dashboard showing the progress of the main test parameters, and real-time performance graphs, giving the first impression of the website's speed and scalability. After the test is complete, StresStimulus aggregates collected information and generates various interactive reports with actionable information that helps answer the performance questions from the previous chapter.

StresStimulus:

- Records user actions from a web browser, RIA, mobile device or other HTTP client.
- Simulates traffic anticipated from the user base. A very large number (up to a million or more) of VUs can be instantiated.
- Realistically emulates the web production environment not only in terms of the amount of load, but also in terms of application traffic content.

- Creates and configures test scripts using the UI without programming. The script editor is available, but is optional.
- Automatically tests how a website works with various user entries, such as predetermined or randomly generated business data.
- Autocorrelates session tokens, cookies and hidden fields in all major web platforms.
- Automatically discovers dynamic parameters and creates parameterization rules to maintain high test fidelity.
- Monitors performance on the client and resources usage on the server side under different load conditions.
- Aggregates performance details by request, page, transaction, test case, VU, and other criteria, and generates customizable graphs and reports.
- Easily pinpoints performance bottlenecks due to high visibility into the test log stored in a queryable database. For example, a waterfall chart of a page or a transaction under different load conditions can be quickly displayed and compared side-by-side for any two VUs or test iterations. It gives precise assessment of the user experience.
- Tracks down bugs causing errors or timeouts, which can only be detected under stress. Typically, such bugs are missed in functional testing.
- Facilitates on-premise performance testing or in the cloud using, for example, Amazon WS.

To make load testing more realistic, the following application performance-impacting factors are emulated:

- Browser behavior (caching, correlation, concurrent connections handling).
- Users' behavior (think time, concurrency, independence, the composition of new and returning users, composition of returning users restarting browser on every iteration or keeping it open).
- Network behavior (upstream and downstream bandwidth).
- User based geographic distribution, when multiple load generation agents in the cloud are positioned in specified regions.

## 1.2 System Requirements

Windows Server 2016 R2 to 2003 or Windows 10 to 7

Microsoft .NET Framework 3.5 or later

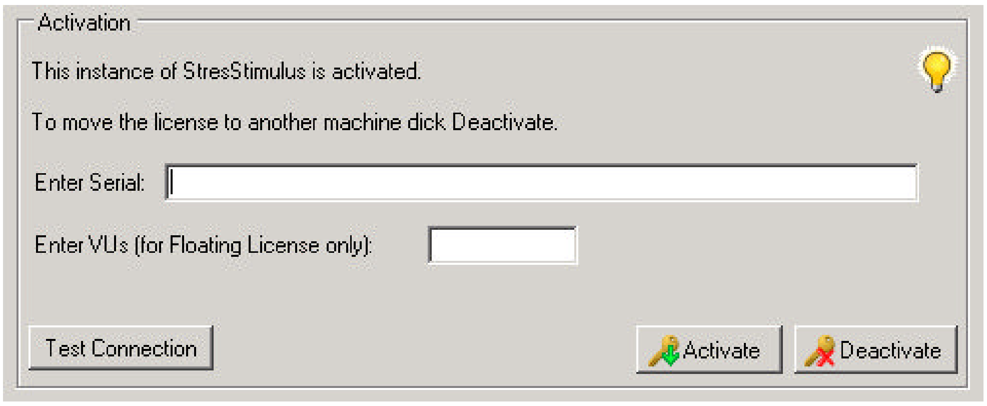
Fiddler 2.3.4.4 or later - free download (optional)

6 GB disk space / 1GHz processor

1GB RAM (2GB+ highly recommended)

## 1.3 Activating License

All StresStimulus licensing options can be found in the licensing dialog. To bring up the licensing dialog in the main menu navigate to Help > License... If you are using the Fiddler Addon version go to StresStimulus > License...



This instance of StresStimulus is activated.

To move the license to another machine click Deactivate.

Enter Serial:

Enter VUs (for Floating License only):

### 1.3.1 Activate the license

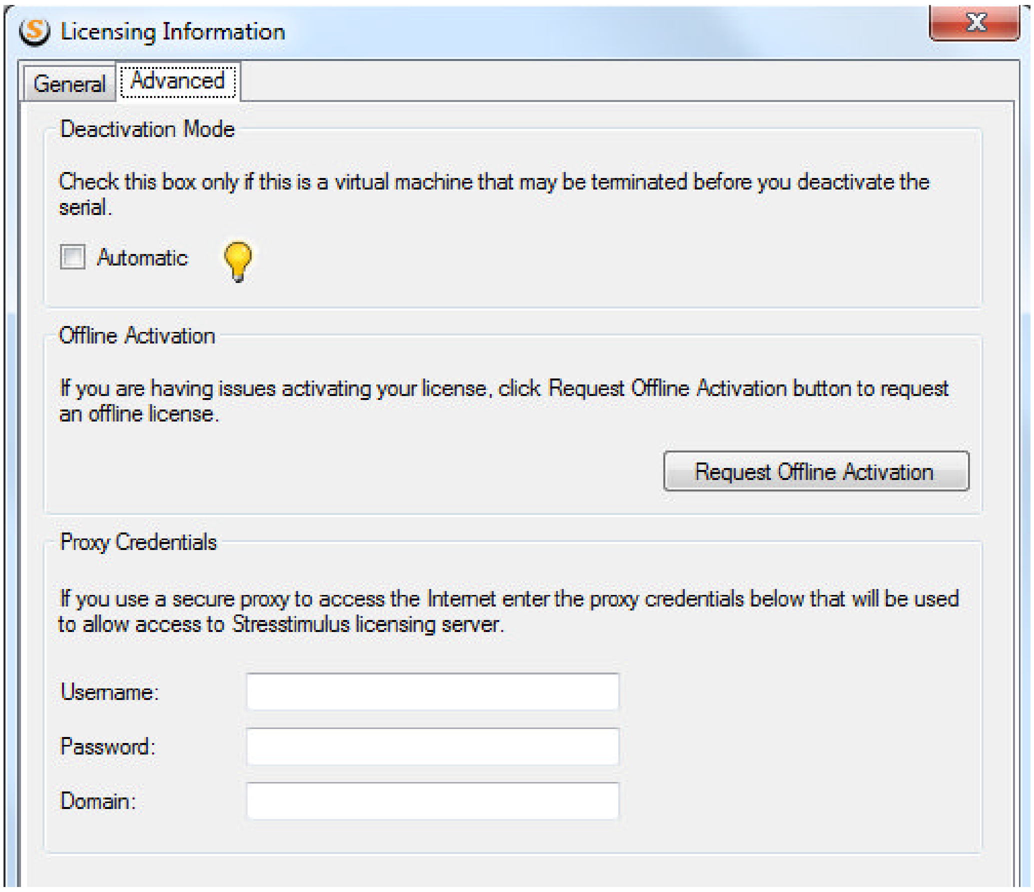
You can enter your serial number in the text box and click Activate button to activate the license. If you have a floating license, enter the number number of VUs to activate in the VUs text box before clicking the Activate button.

### 1.3.2 Moving the license

StresStimulus licenses can be moved from one machine to another. To move a license from an activated machine, first click the Deactivate button to deactivate the license. Then you can move the license by activating on any other machine.

### 1.3.3 Advanced

The following are advanced features and should only be used if activation did not work or you have special licensing requirements. To bring up Advanced options go to the Advanced tab in the licensing dialog.


Licensing Information

General | Advanced

Deactivation Mode

Check this box only if this is a virtual machine that may be terminated before you deactivate the serial.

☐ Automatic

Offline Activation

If you are having issues activating your license, click Request Offline Activation button to request an offline license.

Request Offline Activation

Proxy Credentials

If you use a secure proxy to access the Internet enter the proxy credentials below that will be used to allow access to Stresstimulus licensing server.

Username:

Password:

Domain:

### 1.3.3.1 Automatic Deactivation

If StresStimulus is activated on a virtual machine it is recommended that automatic deactivation checkbox is checked. This was StresStimulus license will automatically be deactivated when StresStimulus is closed.This way if the virtual machine is terminated, StresStimulus license will not be lost.

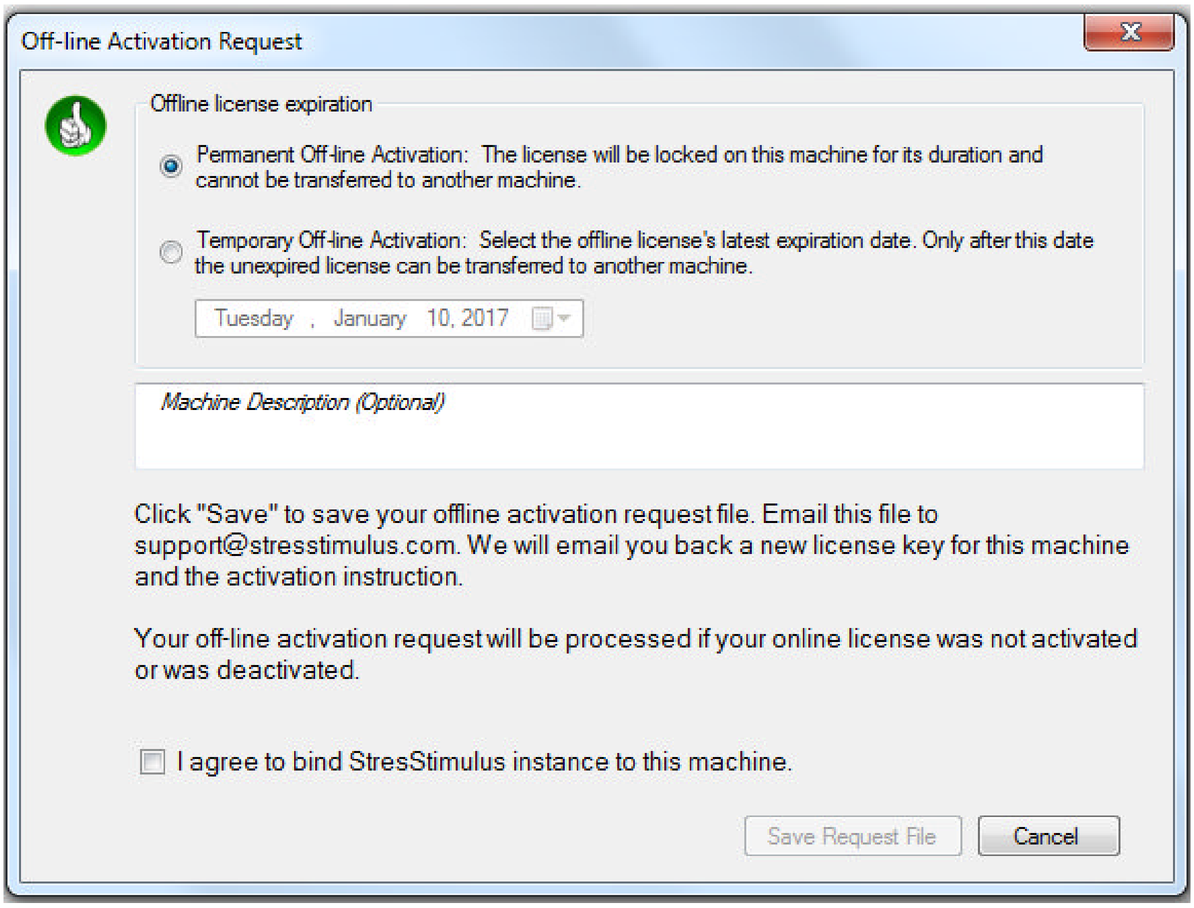
### 1.3.3.2 Proxy Credentials

An active internet connection is necessary to activate the license. If your company uses a proxy that requires authentication to connect to the internet, you can enter the credentials in the Proxy Credential section.

### 1.3.3.3 Offline Activation

It is best to activate StresStimulus using online activation, this way the license can be easily moved from one machine to another as described above. However, if you can't connect to the internet or to StresStimulus licensing server, you can request an offline activation license. When you request an offline activation, StresStimulus support will send you a license file that will lock the license to the machine. During this time, the license will not be movable to another machine. Click the Request

Offline Activation button to bring up the Offline Activation dialog. This dialog may also appear after StresStimulus determines that there is no available connection to the internet.


Off-line Activation Request

Offline license expiration

(●) Permanent Off-line Activation: The license will be locked on this machine for its duration and cannot be transferred to another machine.

( ) Temporary Off-line Activation: Select the offline license's latest expiration date. Only after this date the unexpired license can be transferred to another machine.

Tuesday , January 10, 2017

*Machine Description (Optional)*

Click "Save" to save your offline activation request file. Email this file to support@stresstimulus.com. We will email you back a new license key for this machine and the activation instruction.

Your off-line activation request will be processed if your online license was not activated or was deactivated.

☐ I agree to bind StresStimulus instance to this machine.

First, select the offline license expiration. You can either select from the following options:

- Permanent Offline Activation: The license will be locked on this machine for its duration and cannot be transferred to another machine.
- Temporary Offline Activation:  Select the offline license's latest expiration date. Only after this date the unexpired license can be transferred to another machine. This way the license can be transferred to another machine at a later time.

Then check the checkbox to agree to bind the offline license to the machine and click the Save Request File and save the license request file.

Finally, send the license request file to support@stresstimulus.com for processing.

Once your request is processed, you should receive your offline activation file. Copy the file into the installation directory (for example %Program Files%\Stimulus Technology\StresStimulus) to finish the offline activation.

# 2 GETTING FAMILIAR WITH THE USER INTERFACE

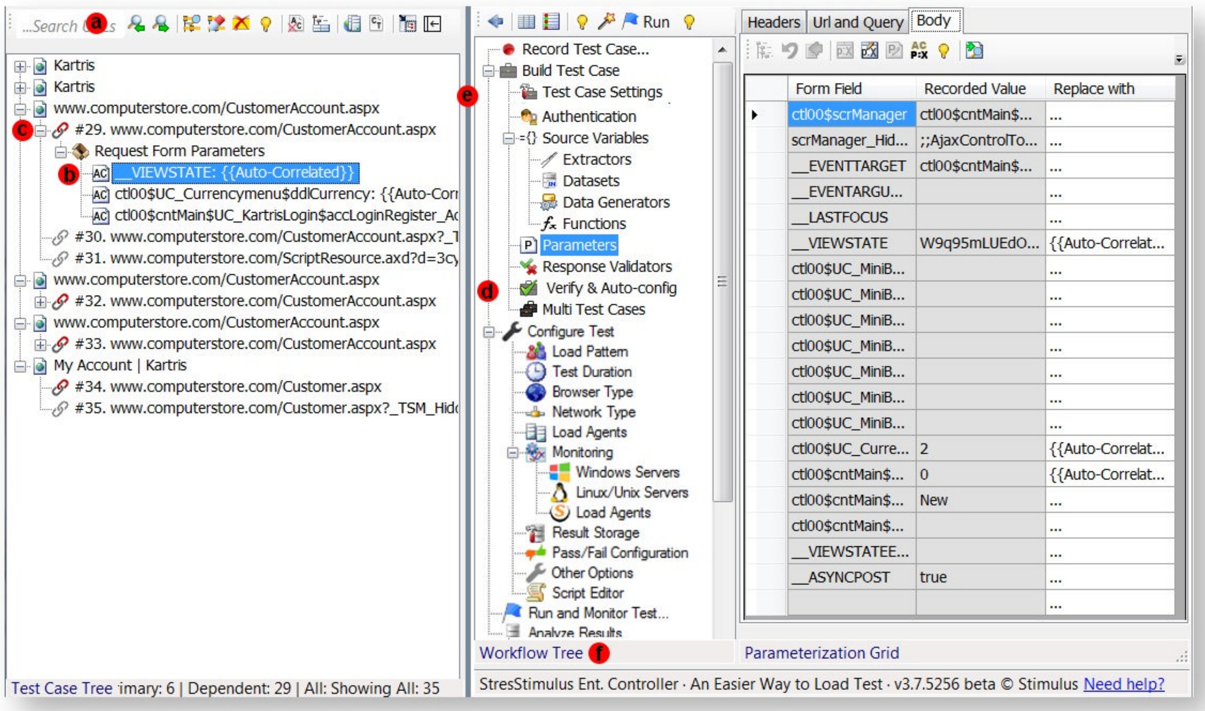
Load testing is an elaborate and time-consuming process that requires special knowledge and experience. StresStimulus is designed to simplify its adoption not only for QA professionals and test analysts but also for developers, IT personnel and other website stakeholders with limited or no experience in load testing. In order to achieve this result, several design principles, discussed in the next section, are applied.

## 2.1 Overview

The StresStimulus UI follows several design principles targeted to make it easier to use:

**1. Reflecting the testing workflow.** The Workflow Tree is the centerpiece of the StresStimulus UI. Every step in the load testing process has a corresponding node on the Workflow Tree. This makes it easier for users to navigate through a multitude of StresStimulus options in the correct order.

**2. Mirroring the Test Object Model (TOM) hierarchical structure.** StresStimulus stores internal data as an object model, where objects are joined in parent-child and other relationships. For example, a Test Case (a) is an object which can have one or several child objects Parameters (b), each of which is associated with a particular request (c). This internal structure can be quickly discovered because a **Parameter** node (d) is located under the **Build Test Case** node (e) in the **Workflow Tree** (f) and also each Parameter node is subordinate to a specific request in the **Test Case Tree**. The parent-child relationship of UI elements corresponds to the TOM hierarchical structure. Learning the UI helps to better understand TOM and vice versa.



- Kartris
- Kartris
- www.computerstore.com/CustomerAccount.aspx
  - #29. www.computerstore.com/CustomerAccount.aspx
    - Request Form Parameters
      - __VIEWSTATE: {{Auto-Correlated}}
      - ctl00$UC_Currencymenu$ddlCurrency: {{Auto-Corr
      - ctl00$cntMain$UC_KartrisLogin$accLoginRegister_Ac
  - #30. www.computerstore.com/CustomerAccount.aspx?_T
  - #31. www.computerstore.com/ScriptResource.axd?d=3cy
- www.computerstore.com/CustomerAccount.aspx
  - #32. www.computerstore.com/CustomerAccount.aspx
- www.computerstore.com/CustomerAccount.aspx
  - #33. www.computerstore.com/CustomerAccount.aspx
- My Account | Kartris
  - #34. www.computerstore.com/Customer.aspx
  - #35. www.computerstore.com/Customer.aspx?_TSM_Hid

Test Case Tree imary: 6 | Dependent: 29 | All: Showing All: 35

- Record Test Case...
- Build Test Case
  - Test Case Settings
  - Authentication
  - Source Variables
    - Extractors
    - Datasets
    - Data Generators
    - Functions
  - Parameters
  - Response Validators
  - Verify & Auto-config
  - Multi Test Cases
- Configure Test
  - Load Pattern
  - Test Duration
  - Browser Type
  - Network Type
  - Load Agents
  - Monitoring
    - Windows Servers
    - Linux/Unix Servers
    - Load Agents
  - Result Storage
  - Pass/Fail Configuration
  - Other Options
  - Script Editor
- Run and Monitor Test...
- Analyze Results

Workflow Tree

| | Form Field | Recorded Value | Replace with |
|---|---|---|---|
| ▸ | ctl00$scrManager | ctl00$cntMain$... | ... |
| | scrManager_Hid... | ;;AjaxControlTo... | ... |
| | __EVENTTARGET | ctl00$cntMain$... | ... |
| | __EVENTARGU... | | ... |
| | __LASTFOCUS | | ... |
| | __VIEWSTATE | W9q95mLUEdO... | {{Auto-Correlat... |
| | ctl00$UC_MiniB... | | ... |
| | ctl00$UC_MiniB... | | ... |
| | ctl00$UC_MiniB... | | ... |
| | ctl00$UC_MiniB... | | ... |
| | ctl00$UC_MiniB... | | ... |
| | ctl00$UC_MiniB... | | ... |
| | ctl00$UC_MiniB... | | ... |
| | ctl00$UC_Curre... | 2 | {{Auto-Correlat... |
| | ctl00$cntMain$... | 0 | {{Auto-Correlat... |
| | ctl00$cntMain$... | New | ... |
| | ctl00$cntMain$... | | ... |
| | __VIEWSTATEE... | | ... |
| | __ASYNCPOST | true | ... |
| | | | ... |

Parameterization Grid

StresStimulus Ent. Controller · An Easier Way to Load Test · v3.7.5256 beta © Stimulus Need help?

**3. Consistency and uniformity**. Common functions are implemented in the same way across all UI areas. After getting familiar with one area, a user can use them in other areas, reducing the learning time. For example, the same **Find** control is used on every grid and treeview for searching objects of different types. For more details, see Uniform UI Elements.

**4. Thinking of end-user**. A special effort was made to avoid confusion as to what to do next. For example, if you get lost and need to return back to the previously visited area, click **Back** on the Workflow Tree toolbar several times until you return to the location you want to be at. This brings the ease-of-use that is available in web browsers. Also, every control, parameter and configuration option is accompanied by a description that appears on mouse-over in help boxes, tooltips or is displayed on the bottom of the screen in the description area.

## 2.2 Standalone and Add-on Versions

The installer, by default, installs two versions of StresStimulus:

- **Standalone version:** implemented as a Windows application
- **Add-on version:** implemented as an add-on to Fiddler, a popular free web debugging proxy by Telerik

Both versions have an almost identical feature set. The same UI elements are encompassed in slightly different layouts, as described in UI Layout. Both versions use the same document format and can be interchangeably used to work with the same test. They also share the state and configuration space. For example, after opening a test in the standalone version, it will appear in the Recent list in the add-on version, and vice versa.

While very similar, each of the versions has its own benefits:

**Standalone version benefits:**

- Does not require Fiddler
- Is easier to use for non-Fiddler users due to less distraction from features unrelated to load testing

**Fiddler add-on version benefits:**

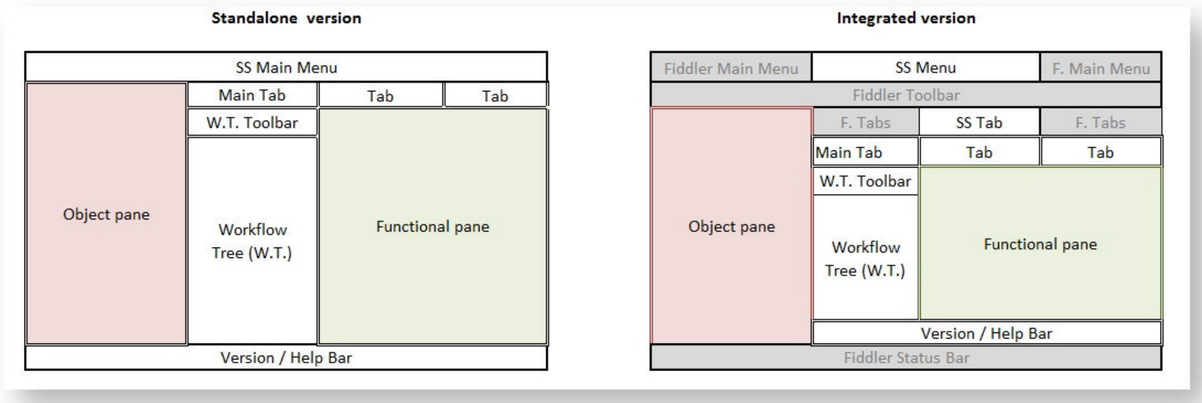
- Tight integration with Fiddler allows to web debug and functional, performance and load test in a single toolset.
- Fiddler features (such as multiple Inspectors, filtering sessions, timeline and auto-responders) can be used to extend the load testing feature set. For example, it creates the ability to simplify debugging and analysis of load tests.

**Note:** Standalone StresStimulus implements its own proxy listening on port 49386 (configurable from the Main Menu -> Options -> StresStimulus Options). It does not require Fiddler to be installed, however, it can coexist with Fiddler. You can even run Fiddler and the standalone version at the same time. Furthermore, when recording a test case in StresStimulus, Fiddler will capture the same sessions because both proxies will be automatically chained. However, while the standalone version is running, you can use Fiddler but the StresStimulus add-on will not work.

## 2.3 UI Layout

The Standalone and add-on versions use the same layout consisting of three vertical panes (see below):

- **Central pane:** Workflow Tree
- **Left pane:** Object area
- **Right pane:** Functional area



In the diagram depicting the add-on version, the gray areas designate Fiddler UI elements. The F. Tab elements show multiple tabs included in Fiddler and its various add-ons.

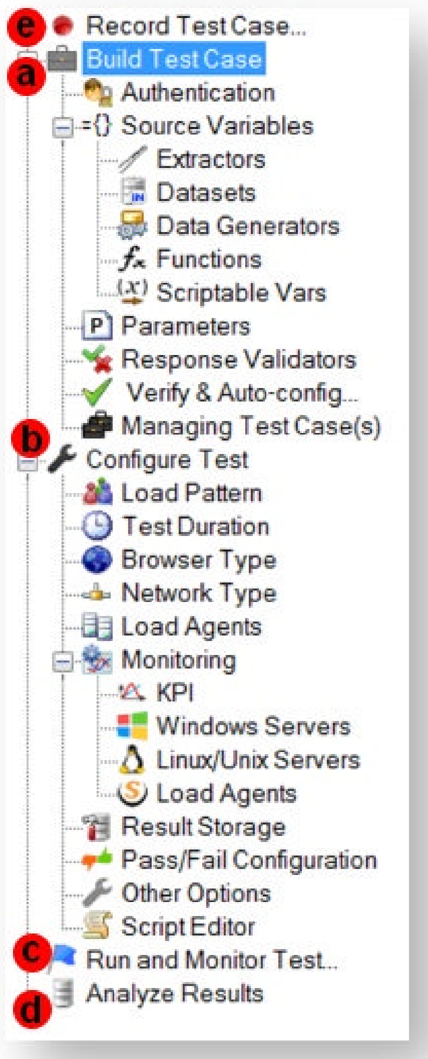
See also: User Interface Reference

### 2.3.1 Workflow Tree UI

The StresStimulus UI interface is designed to reflect the load testing workflow. The centerpiece of the UI is the **Workflow Tree**, located in the center pane. It includes hierarchically organized sections (nodes) for executing certain steps or functions in the load testing process. While moving through the Workflow Tree from top to bottom, the user is guided to perform the necessary steps in the proper order. For example, the top section of the Workflow Tree, **Build Test Case** (a), is used for recording, configuring and verifying a test case. The middle section, **Configure Test** (b), allows to setup load levels and other test run settings. Then, **Run and Monitor Test** (c) is used to execute the test and **Analyze Results** (d) works with test results. This is the exact order in which load testing is performed.

Some steps are necessary only in certain tests. For example, configuring load agents is needed only in large-scale tests. Some other steps such as recording or running the test should be used in every test. The easiest way to operate StresStimulus is to work your way from the top nodes to the bottom nodes to ensure that all necessary steps are executed in the correct order without skipping.

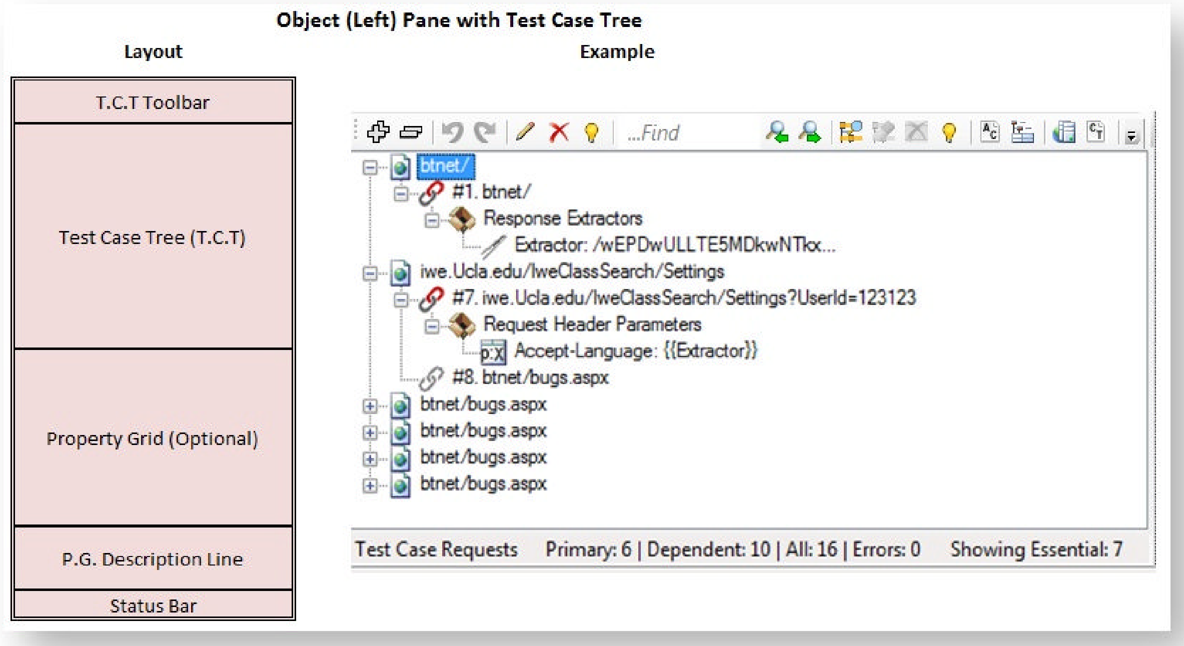
**Tip:** If you need to return back to a previously visited node after navigating to other nodes, click **Back** (e) on the Workflow Tree toolbar several times until you return to the node you need. This helps novice users avoid getting lost while operating StresStimulus.



### 2.3.2 Object (Left) Pane

StresStimulus stores load tests as a hierarchical **Test Object Model (TOM)**. Every webpage, HTTP session, extractor, parameter and validator is an object that is subordinate to its parent object and may contain child objects. **TOM** is displayed in the left object pane. Objects can be presented in one or two forms:

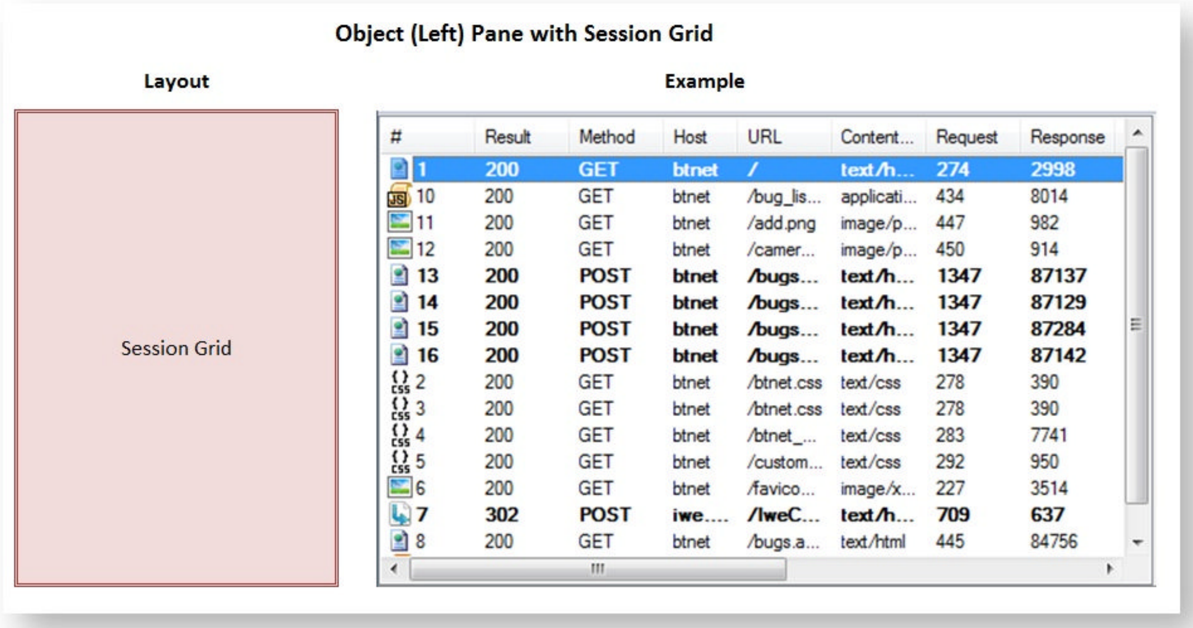
**1. Test Case Tree with nodes corresponding to the TOM's objects**. The test case tree includes a toolbar, a property grid that can be hidden and a status bar.



```
btnet/
  #1. btnet/
    Response Extractors
      Extractor: /wEPDwULLTE5MDkwNTkx...
iwe.Ucla.edu/IweClassSearch/Settings
  #7. iwe.Ucla.edu/IweClassSearch/Settings?UserId=123123
    Request Header Parameters
      Accept-Language: {{Extractor}}
  #8. btnet/bugs.aspx
btnet/bugs.aspx
btnet/bugs.aspx
btnet/bugs.aspx
btnet/bugs.aspx
```

Test Case Requests    Primary: 6 | Dependent: 10 | All: 16 | Errors: 0    Showing Essential: 7

## 2. Session Grid displays a certain subset of recorded or replayed session objects.



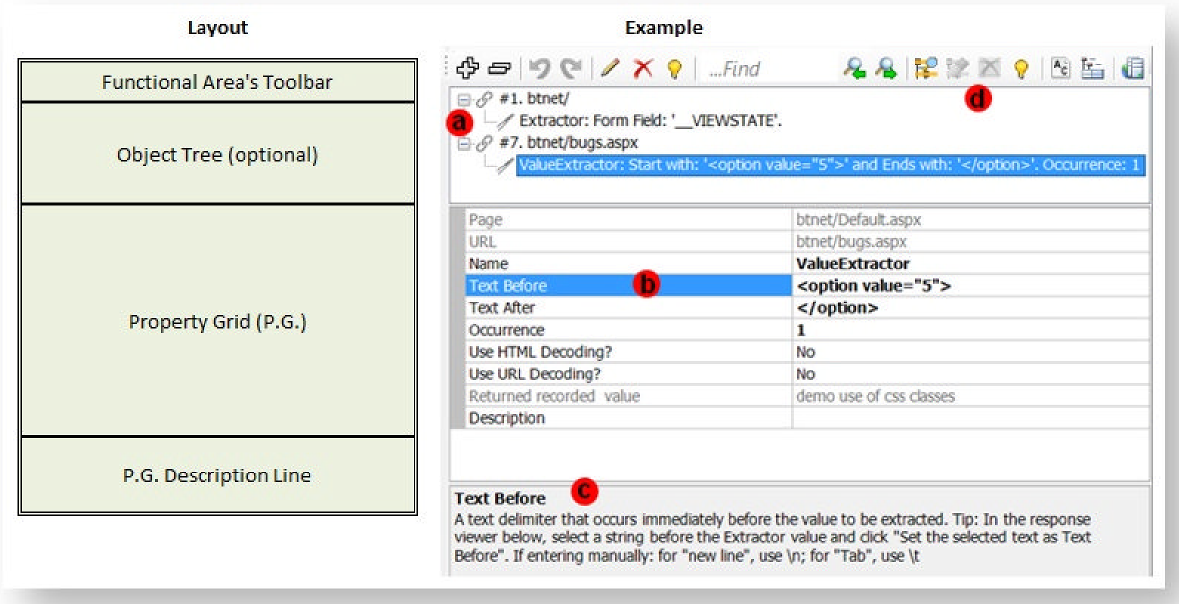
| # | Result | Method | Host | URL | Content... | Request | Response |
|---|---|---|---|---|---|---|---|
| 1 | 200 | GET | btnet | / | text/h... | 274 | 2998 |
| 10 | 200 | GET | btnet | /bug_lis... | applicati... | 434 | 8014 |
| 11 | 200 | GET | btnet | /add.png | image/p... | 447 | 982 |
| 12 | 200 | GET | btnet | /camer... | image/p... | 450 | 914 |
| 13 | 200 | POST | btnet | /bugs... | text/h... | 1347 | 87137 |
| 14 | 200 | POST | btnet | /bugs... | text/h... | 1347 | 87129 |
| 15 | 200 | POST | btnet | /bugs... | text/h... | 1347 | 87284 |
| 16 | 200 | POST | btnet | /bugs... | text/h... | 1347 | 87142 |
| 2 | 200 | GET | btnet | /btnet.css | text/css | 278 | 390 |
| 3 | 200 | GET | btnet | /btnet.css | text/css | 278 | 390 |
| 4 | 200 | GET | btnet | /btnet_... | text/css | 283 | 7741 |
| 5 | 200 | GET | btnet | /custom... | text/css | 292 | 950 |
| 6 | 200 | GET | btnet | /favico... | image/x... | 227 | 3514 |
| 7 | 302 | POST | iwe.... | /IweC... | text/h... | 709 | 637 |
| 8 | 200 | GET | btnet | /bugs.a... | text/html | 445 | 84756 |

Each pane displays one or several toolbars with buttons for performing various actions.

## 2.3.3 Functional (Right) Pane

Each node on the Workflow Tree has a corresponding window or dialog box displayed in the right functional pane when the node selected. The functional pane can be presented in two forms:

**1. Property grid with or without object tree.** Nodes (a) in the object tree and their properties (b) correspond to the TOM's objects and their respective properties. For example, the **Extractor** node and its **Text Before** property in the property grid (see User Interface Reference -> Extractors) corresponds to the **Extractor** node and the **Extractor's Text Before** attribute (see Script Quick Reference), respectively. The description of the selected property is displayed in the description line on the bottom (c). The toolbar provides access to functional commands (d).

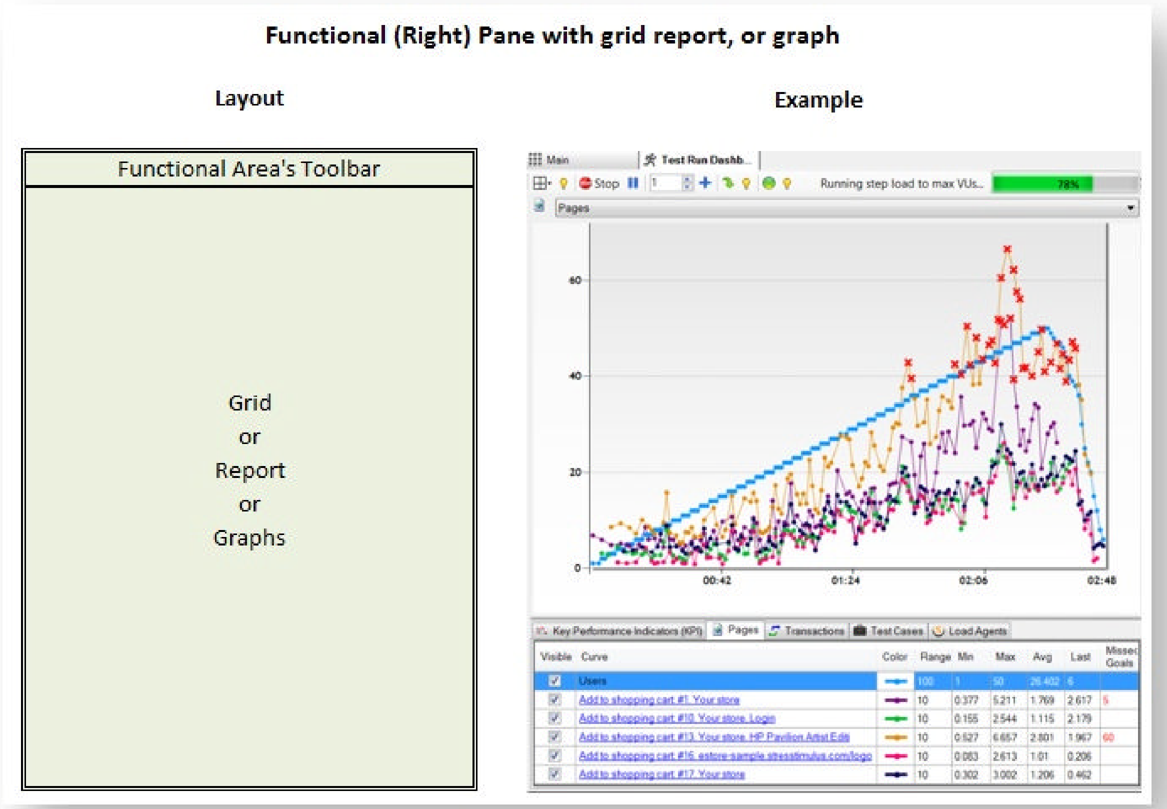


#1. btnet/
Extractor: Form Field: '__VIEWSTATE'.
#7. btnet/bugs.aspx
ValueExtractor: Start with: '<option value="5">' and Ends with: '</option>'. Occurrence: 1

| | |
|---|---|
| Page | btnet/Default.aspx |
| URL | btnet/bugs.aspx |
| Name | **ValueExtractor** |
| Text Before | **<option value="5">** |
| Text After | **</option>** |
| Occurrence | **1** |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |
| Returned recorded value | demo use of css classes |
| Description | |

**Text Before**
A text delimiter that occurs immediately before the value to be extracted. Tip: In the response viewer below, select a string before the Extractor value and click "Set the selected text as Text Before". If entering manually: for "new line", use \n; for "Tab", use \t

**2. Grid, report or graph.**


Functional (Right) Pane with grid report, or graph
Layout
Example
Functional Area's Toolbar
Grid
or
Report
or
Graphs
Main
Test Run Dashb...
Stop
Running step load to max VUs...
78%
Pages
Key Performance Indicators (KPI)
Pages
Transactions
Test Cases
Load Agents

| Visible | Curve | Color | Range | Min | Max | Avg | Last | Missed Goals |
|---|---|---|---|---|---|---|---|---|
| ✓ | Users | 🟦 | 100 | 1 | 50 | 26.402 | 6 | |
| ✓ | Add to shopping cart #1. Your store | 🟪 | 10 | 0.377 | 5.211 | 1.769 | 2.617 | 5 |
| ✓ | Add to shopping cart #10. Your store. Login | 🟩 | 10 | 0.155 | 2.544 | 1.115 | 2.179 | |
| ✓ | Add to shopping cart #13. Your store. HP Pavilion Artist Edit | 🟧 | 10 | 0.527 | 6.657 | 2.801 | 1.967 | 60 |
| ✓ | Add to shopping cart #16. estore-sample.stresstimulus.com/logo | 🟥 | 10 | 0.083 | 2.613 | 1.01 | 0.206 | |
| ✓ | Add to shopping cart #17. Your store | ⬛ | 10 | 0.302 | 3.002 | 1.206 | 0.462 | |

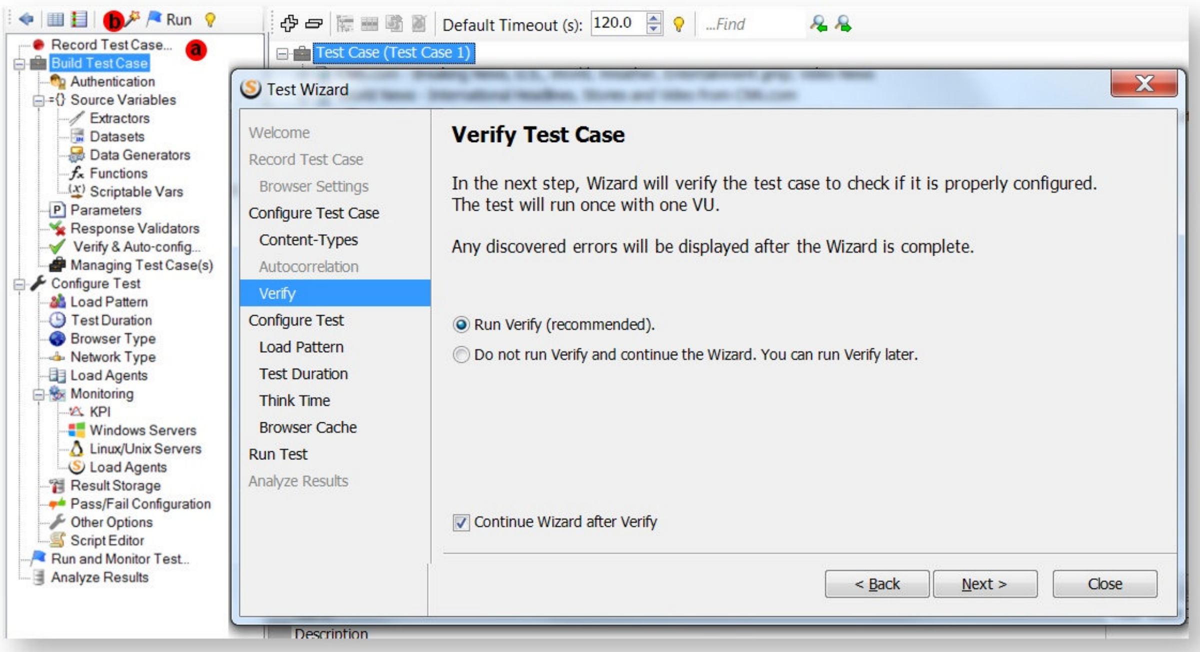
## 2.4 Test Wizard

The Test Wizard is a helpful tool for an easy jump-start with StresStimulus. It guides a user through all major steps of recording, configuring, running a test and presenting some major results. The Test Wizard allows to complete a single load test with minimal or no knowledge of performance testing or StresStimulus.

To launch the test wizard, click **Record Test Case** on the Workflow Tree (a) or click **Test Wizard** (b) on the toolbar.

Additionally, the Test Wizard has the following benefits:

**1. Expediting StresStimulus learning.** While navigating through the Test Wizard steps, the corresponding areas of the UI are highlighted to indicate how to executed the same step without the wizard. This simplifies StresStimulus adoption.



Verify Test Case

In the next step, Wizard will verify the test case to check if it is properly configured.
The test will run once with one VU.

Any discovered errors will be displayed after the Wizard is complete.

[✓ Run Verify (recommended).]
Do not run Verify and continue the Wizard. You can run Verify later.

[✓ Continue Wizard after Verify]

**2. Customizing the level of detail in the wizard.** Initially, the wizard will guide you through all relevant configuration steps. You have the option to skip certain wizard steps by selecting **Do not show...** For example, you can skip configuring the target hosts list. The wizard will remember your selection to skip the step next time.

**3. Manual selecting wizard steps.** The left wizard pane displays a step menu showing the sequence of configuration steps. Typically a user will navigate through steps sequentially by clicking **Next**. However, you can also click on the available step to change the normal execution sequence to skip certain steps or to run a specific step. Unavailable steps are disabled. For example, if the autocorrelation step was already completed automatically,

## 2.5 Uniform UI Elements

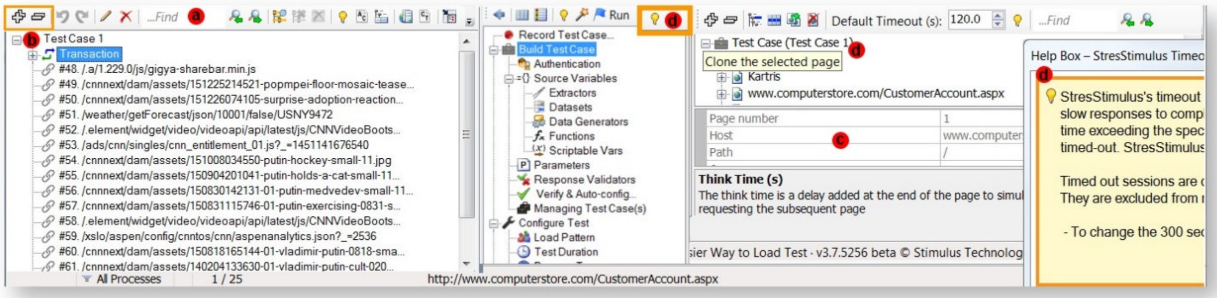
Common functions, provided below, are implemented the same way across all UI areas. This helps to reduce learning time and increase the productivity of the testing process.

**1. Uniform operations with objects.** Objects, such as pages, requests, transactions, extractors, parameters, and validators are displayed in the hierarchical treeviews which support 5 standard operations: **Create**, **Edit**, **Delete**, **Move**, and **Clone**. Operations with objects can be invoked either from the toolbar buttons or from the context menu when right-clicking on the object. Both of these options display uniformly designed icons. The icon examples are provided below. As an alternative to the **Move** button, you can drag and drop an object to a new location.

| | Operation | | | | |
|---|---|---|---|---|---|
| **Object type** | Create | Edit | Delete | Move | Clone |
| Request | | ✎ | ✗ | Drag+Drop | ⧉ |
| Page | 🔗 | ✎ | ✗ | ⬆ | ⧉ |
| Extractor | ✎ | ✎ | ✗ | Drag+Drop | |
| Validator | ✓ | ✎ | ✗ | Drag+Drop | ✓ |
| Transactions | ⇄ | ✎ | ✗ | ⬆ | ⧉ |
| Loops | ↻ | ✎ | ✗ | ⬆ | ⧉ |

**2. Uniform toolbar operations in object and functional areas.** The following commonly used controls are present in toolbars across all windows:

**a. Searching content of displaying elements.** The same **Find** control is placed in the rightmost position of every toolbar to search content of every grid and tree view displaying objects of different types. The **Next** and **Previous** buttons allow search in both directions.



```
Test Case 1
  Transaction
    #48. /.a/1.229.0/js/gigya-sharebar.min.js
    #49. /cnnnext/dam/assets/151225214521-popmpei-floor-mosaic-tease...
    #50. /cnnnext/dam/assets/151226074105-surprise-adoption-reaction...
    #51. /weather/getForecast/json/10001/false/USNY9472
    #52. /.element/widget/video/videoapi/api/latest/js/CNNVideoBoots...
    #53. /ads/cnn/singles/cnn_entitlement_01.js?_=1451141676540
    #54. /cnnnext/dam/assets/151008034550-putin-hockey-small-11.jpg
    #55. /cnnnext/dam/assets/150904201041-putin-holds-a-cat-small-11...
    #56. /cnnnext/dam/assets/150830142131-01-putin-medvedev-small-11...
    #57. /cnnnext/dam/assets/150831115746-01-putin-exercising-0831-s...
    #58. /.element/widget/video/videoapi/api/latest/js/CNNVideoBoots...
    #59. /xslo/aspen/config/cnntos/cnn/aspenanalytics.json?_=2536
    #60. /cnnnext/dam/assets/150818165144-01-vladimir-putin-0818-sma...
    #61. /cnnnext/dam/assets/140204133630-01-vladimir-putin-cult-020...
All Processes    1 / 25

Record Test Case...
Build Test Case
  Authentication
  Source Variables
    Extractors
    Datasets
    Data Generators
    Functions
    Scriptable Vars
  Parameters
  Response Validators
  Verify & Auto-config...
  Managing Test Case(s)
Configure Test
  Load Pattern
  Test Duration

Test Case (Test Case 1)
Clone the selected page
  Kartris
  www.computerstore.com/CustomerAccount.aspx
Page number    1
Host           www.computer
Path           /
Think Time (s)
The think time is a delay added at the end of the page to simul
requesting the subsequent page
sier Way to Load Test · v3.7.5256 beta © Stimulus Technolog

Help Box – StresStimulus Timeo
StresStimulus's timeout
slow responses to comp
time exceeding the spec
timed-out. StresStimulus

Timed out sessions are
They are excluded from

- To change the 300 sec

http://www.computerstore.com/CustomerAccount.aspx
```

**b. Expand / Collapse on every treeview. Expand** and **Collapse** buttons are placed in the leftmost position on every toolbar above object treeviews.

**c. Object trees coupled with property grids.** Treeviews of every object type are coupled with a **property grid**, displaying every property, along with help information explaining the property meaning and available options. The user can change values of all non-read-only properties. A property grid is a uniform data entry mechanism for configuring most of the StresStimulus test parameters.

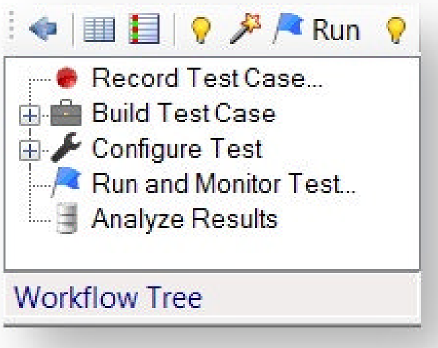
**d. Help boxes / tooltips for all controls.** Every StresStimulus control and window has **tooltips** / **help boxes** with brief instruction. Help boxes provide guidance through the load testing process.

**Tip:** If you use help boxes infrequently, you can change their behavior to pop up on-click. To do so, check the box in the Main Menu -> StresStimulus Options.

**3. Copy read-only continent to the clipboard.** Sometimes it is necessary to copy read-only values displayed in the UI into the clipboard. This information can be pasted while searching the object model or documentation for a specific value or content. It can save time and prevent typos, since no retyping is necessary. In StresStimulus, a value of every object, metrics and even message content from grids, treeviews and message boxes can be selected and copied to the clipboard.

# 3 TEST STRUCTURE

A StresStimulus Test is a self-contained project comprising of elements defining a specific function or step in the load testing process. The test can be saved and then reopened from the StresStimulus Main Menu by opening the main project file with extension .ssconfig. The test elements can be divided into 4 categories corresponding to the 4 top-level nodes on the Workflow Tree:



**1.** One or several Test Cases. Each test case includes:

- A sequence of recorded web sessions stored in a dedicated .saz file. Each web session includes an HTTP request sent by the client and a corresponding server response to it.
- A set of objects, such as extractors, parameters, transactions, loops and validators, and their properties stored in a .ssconfig file as an XML representing a hierarchical test object model (TOM). A single .ssconfig file containing TOM for all test cases is created in a test.
- One or several optional datasets, each of which is stored in a separate .csv file. A dataset can be used in more than one test case.

2. Test Configuration includes a set of objects defining test parameters such as a number of virtual users, load pattern, test duration and test completion criteria. Test Configuration properties are stored in the .ssconfig file as an XML a hierarchical object model.

3. Controls for running and monitoring Test.

4. Test Results for individual test runs. Each test run includes a test log along with collected performance metrics stored in a SQL Server CE (compact edition) .sdf file.

**Note:** SQL Server can be used instead of SQL Server CE.

The Test file set includes:

- Main configuration file: <TestName > .ssconfig.
- A subfolder with the same name <TestName>, located in the same folder, includes the following files:
    - One or several session files <TestCaseName> .saz - one per test case.
    - Optional <DatasetName>.csv files - (one per dataset).
    - __Auth.csv file with test credentials (if applicable).
    - Optional <TestRunName-Timestamp>.sdf file - one per previous test run. You can archive or delete some of the .sdf file to hide or delete unimportant run results.

The default location of the test files is %My Documents%\Fiddler2\StresStimulus. If you need test versioning, save every test version with the new name or use your external versioning (source control) tool.

# 4 BUILDING TEST CASE

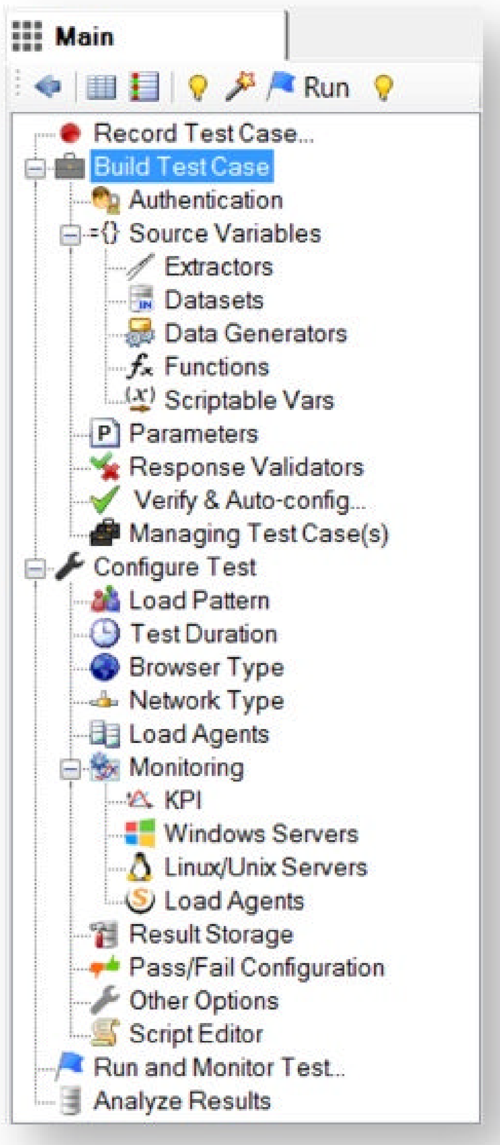
**Search Building Test Case section**

A test case is a set of HTTP/S requests generated by a user’s clicks or actions that represent a specific usage scenario. The test case represents an impact to website performance induced by a single user. In some load testing literature, a test case is called **Script.**

A test can have one or several test cases. Every test case is created and configured independently. The test case can be instantiated in StresStimulus by:

- Recording browser or other web client actions as a new test case.
- Creating from a set of HTTP sessions captured by a proxy.
- Importing a test case from a different test.
- Opening a previously stored test.

All steps of recording and configuring a test case are performed on the **Main** tab on the first two sections in the Workflow Tree: **Record Test Case** and **Build Test Case.**



## 4.1 Recording Test Case

| | |
|---|---|
| The first step in load testing with | |

StresStimulus is to record a test case. A test script is automatically created by recording a user navigation scenario.

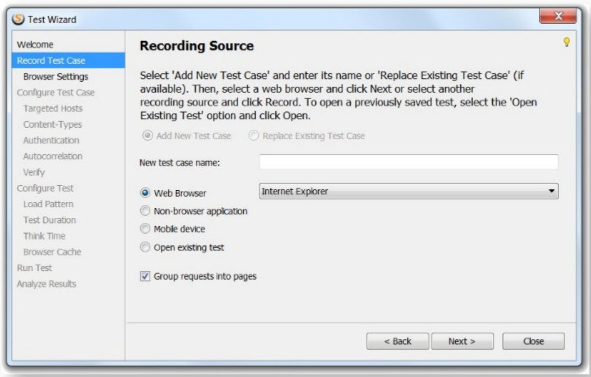
The user's requests and server responses are captured by a proxy and stored in StresStimulus as test case objects.

**Note:** The add-on version uses Fiddler's proxy. The Standalone version uses its own proxy which can work side-by-side with the Fiddler proxy.

To start recording, click **Record Test Case** in the Workflow Tree**.** The **Test Wizard** will display the **Recording Source** step.

Enter Test Case name (optional), then select a recording source.

- To load test websites, select **Web Browser**, then from the drop-down, select the browser type and click **Next.** Browsers installed on your computer will be on the list. The following browsers are supported: Internet Explorer, Firefox, Chrome, Opera, Safari.
- To load test other client applications, such as Silverlight or Flex, select **Non-browser applications,** and then click **Record**.
- To load test mobile apps accessed from external devices, select **Mobile device**, then click **Record**.
- To open a previously saved test, select the **Open existing test** and click **Open**.
- By default StresStimulus breaks down requests into pages. You can disable automatic page breakdown. To do so, un-checked the **Group requests into pages** box. This can be helpful when, instead of tracking the performance of pages, you prefer to track performance of transactions which are used to group requests manually. Disabling pages is also helpful when testing Web services.



Recording Source

Select 'Add New Test Case' and enter its name or 'Replace Existing Test Case' (if available). Then, select a web browser and click Next or select another recording source and click Record. To open a previously saved test, select the 'Open Existing Test' option and click Open.

[✓ Add New Test Case] Replace Existing Test Case

New test case name:

[✓ Web Browser] Internet Explorer

Non-browser application

Mobile device

Open existing test

[✓] Group requests into pages

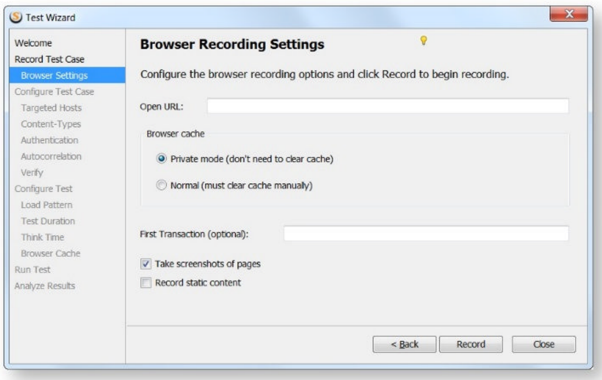
## 4.1.1 Recording with Web Browser

If you selected a browser as a recording source, the Test Wizard will display the **Browser Recording Settings** step. Enter initial URL and select a browser cache option.

- If **Private Mode** is selected (recommended), the private browsing mode will be enabled, in which browser cache is not used, so you don't need to clear it. Some web applications do not work in private browsing mode. In this case, select Normal mode. If you use any browser except Internet Explorer, make sure to clear cache (temporary Internet files and cookies) before recording. This is necessary to prevent the browser from using cached data, in which case, the proxy will not capture that traffic or empty 304 responses will be captured. Internet Explorer cache will be cleared automatically and selectively.

- Then enter the first transaction name (optional) and click **Record** to launch the web browser. If the browser is running, a message will prompt to close it first.

- Ent SP During recording, StresStimulus will take web page screenshots. When you review your test case, the screenshots are displayed as you select the session object on the test case tree in the


Test Wizard

**Browser Recording Settings**

Configure the browser recording options and click Record to begin recording.

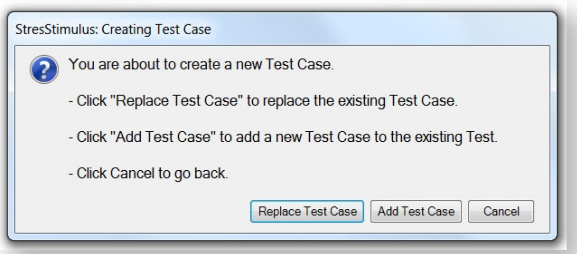
Open URL:

Browser cache

[✓ Private mode (don't need to clear cache)]
Normal (must clear cache manually)

First Transaction (optional):

[✓ Take screenshots of pages]
Record static content

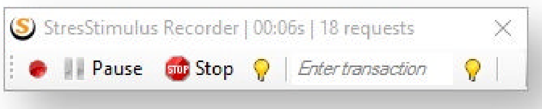

StresStimulus: Creating Test Case

You are about to create a new Test Case.

- Click "Replace Test Case" to replace the existing Test Case.

- Click "Add Test Case" to add a new Test Case to the existing Test.

- Click Cancel to go back.


StresStimulus Recorder | 00:06s | 18 requests

request properties view. These screenshots can help you remember how the page looked like before a new click occurred. If you do not wish to take screenshots, un-check the **Take screenshots of pages** box.

If you use several monitors, then in order to successfully capture webpage screenshots, display the recording browser on the same monitor as StresStimulus. Otherwise screenshots may not come out correctly.

- Typically static resources such as images and style sheets have a very little impact on website performance. Therefore they are not recorded by default. Excluding static resources from performance testing allows to emulate a greater number of virtual users on the same hardware resources. If you wish to record static resources, check the **Record static content** box.

If at least one test case already exists in the Test, the following dialog will give a choice to either replace the current test case, or add a new test case.

After that the floating **Recorder** bar will appear. StresStimulus is ready to capture client actions from the recording source. Click through the application scenario that you want to test. StresStimulus will record all user actions.

During recording, the number of captured sessions is displayed in the recorder title bar. The list of captured sessions will appear in the

StresStimulus session grid (in the standalone version) or in the Fiddler grid (in the add-on version). If the captured sessions counter in the Recorder does not increment, check Proxy Settings in Standalone Version.

To skip the recording of some pages, click **Pause** in the Recorder . To continue recording, click **Resume** .

When recording is finished, click **Stop** in the Recorder or in StresStimulus. If you record in the browser, it will be closed. A new test case will be created. StresStimulus will filter out unrelated requests from other browsers or non-browser processes and will create a test case.

**Note:** Filtering unrelated requests is based on checking initiated processes. Filtering may not work when security software installed on the test machine masks the initiated processes (e.g. Kaspersky security software marks all requests as issued by AVP process). To restore filtering, disable such security software.

**Note:** StresStimulus supports system proxy automatically. Some companies use a proxy that requires custom configuration. In this case, the recorder may not capture traffic by default. StresStimulus will display an error message recommending you modify your proxy settings if this occurs.

| | |
|---|---|
| | |

#### 4.1.1.1 Proxy Settings in Standalone Version

The standalone version is designed to turn the proxy on automatically when you start recording and remove the proxy when you close the recorder. This is necessary for the recorder to capture traffic.

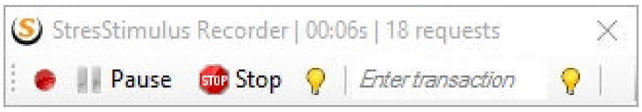
While automatic proxy configuration works in most cases, sometimes it fails. In this case, the recorder will not capture the traffic. There are several possible reasons for this. For example:

- There is a policy set that prevents the proxy from being changed
- Other software like security / networking / vpn / malware is resetting the proxy

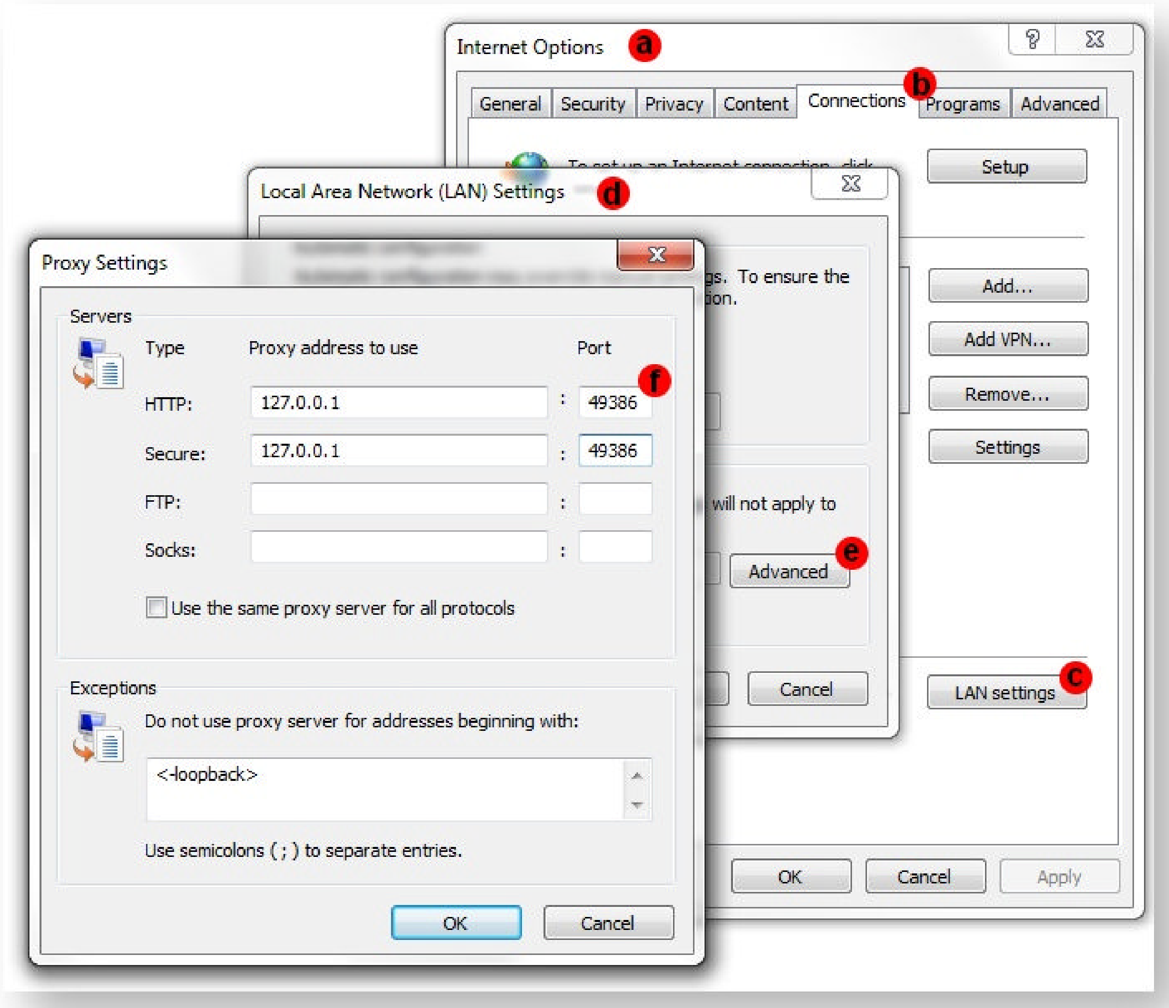
In such cases, the proxy configuration should be completed manually. To do so, follow these troubleshooting Instructions:

After you start recording in StresStimulus, it registers itself as the system proxy. You can verify that StresStimulus is correctly configured by using the following steps:

1. Make sure you are in recording mode and the recorder window is visible.



2. Open Internet Explorer if it is not already open.
3. From Internet Explorer main menu, click **Tools > Internet Options.** In the dialog (a), select the **Connections** tab (b). Click LAN **Setting** (c). In the dialog (d) click **Advanced** (e). By default, StresStimulus proxy uses port 49386 (f).
4. If the proxy is already specified correctly, then the proxy is correct. Otherwise, enter these settings manually.



Proxy Settings

Servers

| Type | Proxy address to use | | Port |
|---|---|---|---|
| HTTP: | 127.0.0.1 | : | 49386 |
| Secure: | 127.0.0.1 | : | 49386 |
| FTP: | | : | |
| Socks: | | : | |

☐ Use the same proxy server for all protocols

Exceptions

Do not use proxy server for addresses beginning with:

<-loopback>

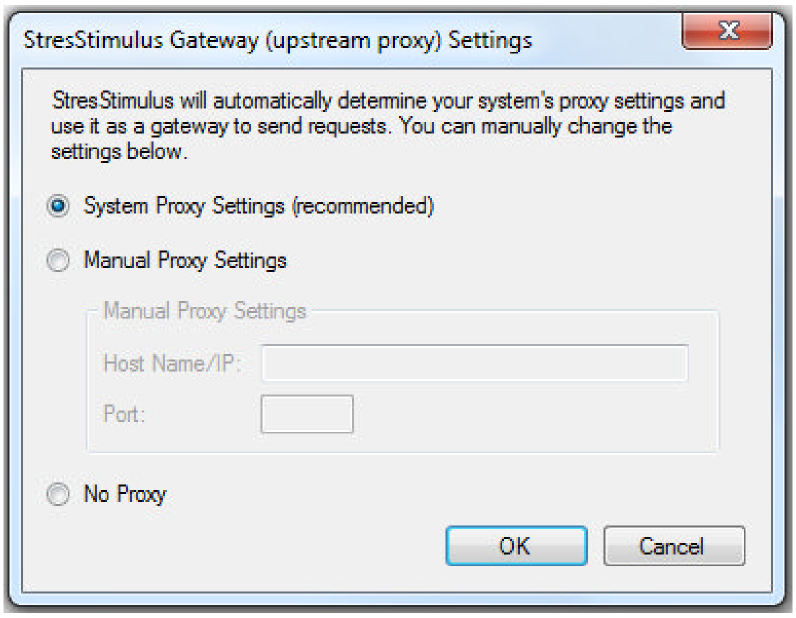
Use semicolons ( ; ) to separate entries.

**Tip:** Port 49386 is the default StresStimulus listening port. You can change this port in StresStimulus main menu **Options > Options...**

## Checking Upstream Proxy Settings

If accessing a web application under the test requires using your company proxy, then StresStimulus will try to automatically chain to it by reading system proxy settings. If the upstream proxy information cannot be accessed through the system proxy, you need to select the **Manual Proxy Settings** option and specify the company proxy host and port..

In the main menu go to **Options > Gateway Settings** to open the StresStimulus Gateway Settings dialog. Here you can select to use the system proxy settings, input manual proxy address and port, or use no proxy.


StresStimulus Gateway (upstream proxy) Settings

StresStimulus will automatically determine your system's proxy settings and use it as a gateway to send requests. You can manually change the settings below.

[✓ System Proxy Settings (recommended)]

Manual Proxy Settings

Manual Proxy Settings

Host Name/IP:

Port:

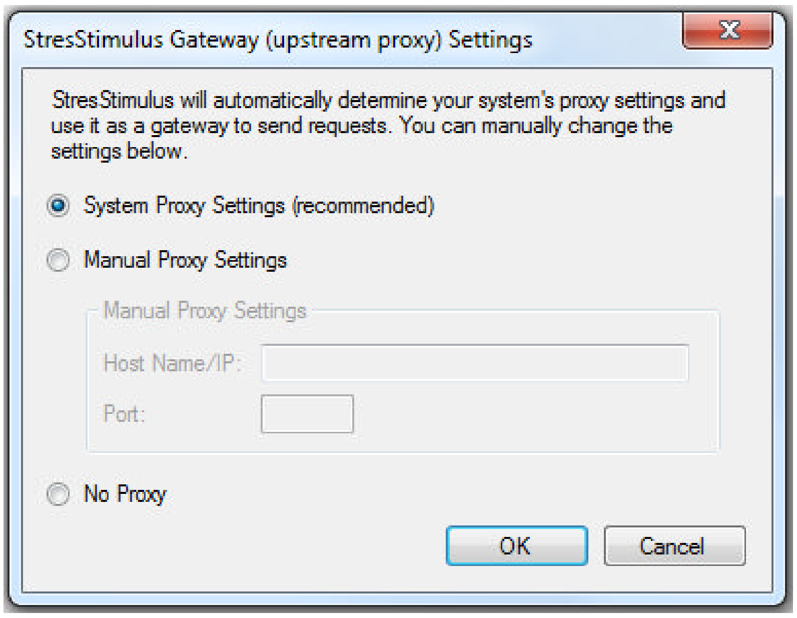
No Proxy

### 4.1.1.2 Upstream Proxy Settings in Standalone Version

If your web client is configured to use an upstream proxy, then during recording StresStimulus will chain its proxy with the upstream proxy by reading system proxy settings. While automatic configuration works in most cases, sometimes it fails because of restrictive security policies, interference with other software or if the upstream proxy information cannot be accessed through the system proxy.

In such cases, the upstream proxy configuration should be completed manually. In the main menu go to **Options > Gateway Settings** to open the StresStimulus Gateway Settings dialog.

Select **Manual Proxy Settings** and specify a host and port. You can also select use the system proxy settings or use no proxy.


StresStimulus Gateway (upstream proxy) Settings

StresStimulus will automatically determine your system's proxy settings and use it as a gateway to send requests. You can manually change the settings below.

[✓ System Proxy Settings (recommended)]

Manual Proxy Settings

Manual Proxy Settings

Host Name/IP:

Port:

No Proxy

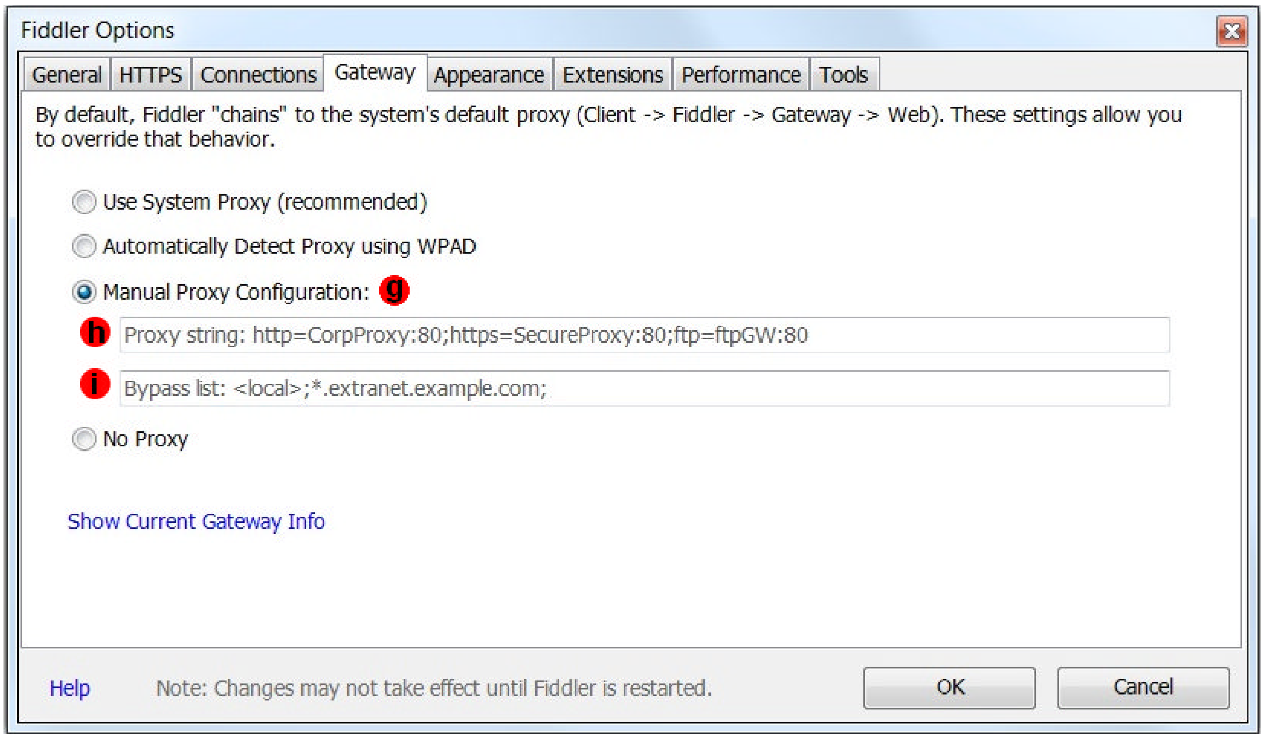
### 4.1.1.3 Checking Proxy Settings

Fiddler automatically modifies default proxy settings as described in the Fiddler documentation. However when StresStimulus is selected in Fiddler, the Fiddler proxy is disabled and default proxy settings are restored. This is necessary to make sure that Internet traffic initiated by other applications does not interfere with StresStimulus test traffic during the test run. Fiddler proxy settings are re-enabled again after recording or when the user navigates to a tab other than StresStimulus.

**Note:** If your company does not allow automatic proxy settings, enter proxy settings manually. Change the port number to the Fiddler listening port 8888.

### 4.1.1.4 Chaining to an Upstream Proxy

If the proxy settings described above are correct, but you still have issues with recording or accessing your website, then the problem may be related to a company (upstream) proxy. If such proxy is required for accessing relevant websites, then HTTP requests from the web browser that travel through StresStimulus or Fiddler, must be routed to the Company proxy. Fiddler and StresStimulus support automatic chaining to the upstream proxy. If the automatic chaining does not work, then configure the upstream proxy chain manually. Use the StresStimulus add-on version, go to Fiddler **Tools > Fiddler Options > Gateway** to specify your upstream proxy info.


Fiddler Options

By default, Fiddler "chains" to the system's default proxy (Client -> Fiddler -> Gateway -> Web). These settings allow you to override that behavior.

- Use System Proxy (recommended)
- Automatically Detect Proxy using WPAD
- [✓ Manual Proxy Configuration:] g
  - h Proxy string: http=CorpProxy:80;https=SecureProxy:80;ftp=ftpGW:80
  - i Bypass list: <local>;*.extranet.example.com;
- No Proxy

Show Current Gateway Info

Note: Changes may not take effect until Fiddler is restarted.

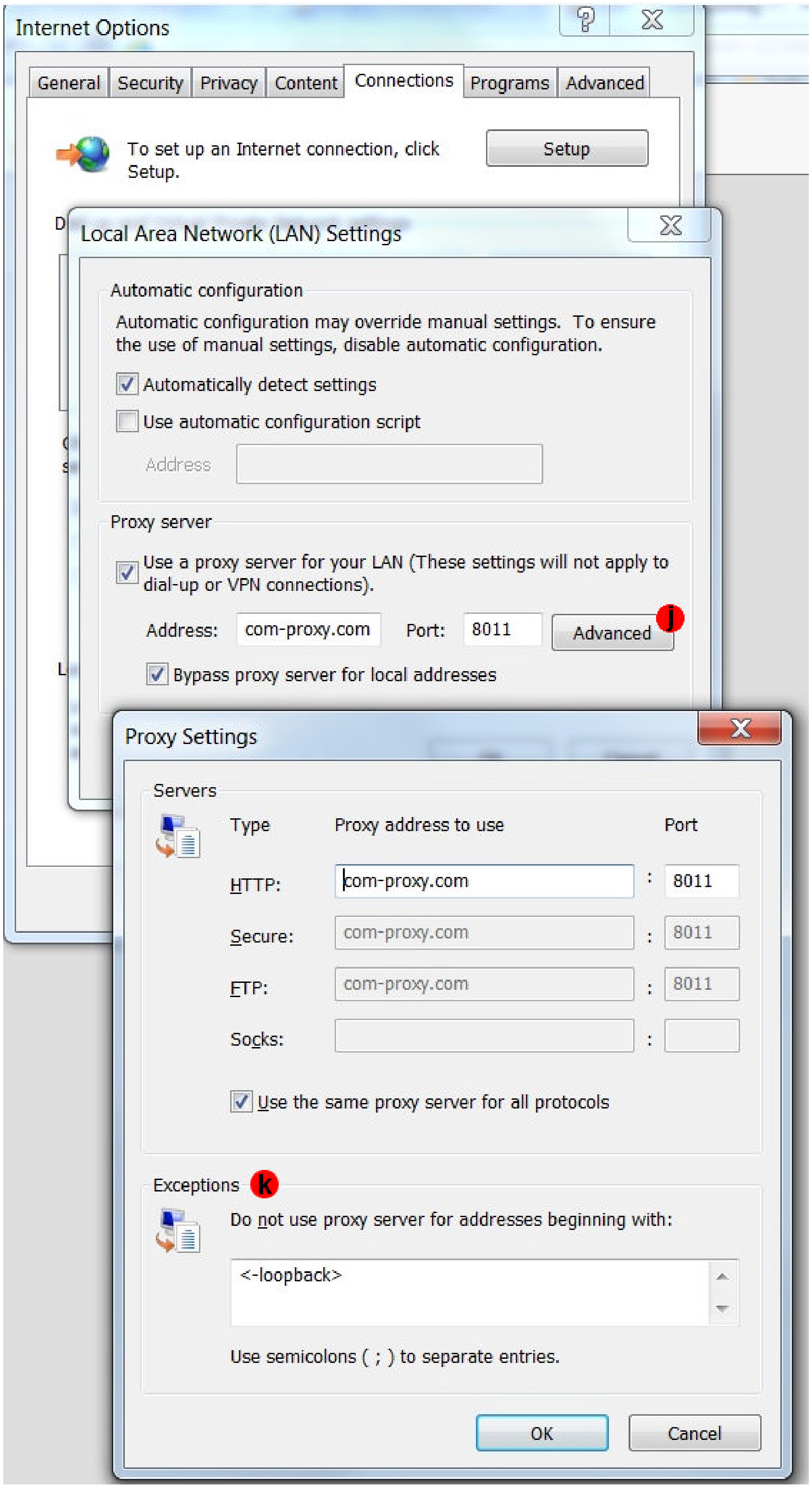
If the options **Use System Proxy** and **Automatically Detect Proxy using WPAD** do not work, use **Manual Proxy Configuration** (g).

Enter Proxy string on the first line in (h) the following format:

```
http=<CorpProxy>:<port>;https=<SecureProxy>:<secureport>;
```

Enter hosts that should be not routed through the Proxy on the second line (Bypass list) separated by ";" (i)

Your corporate proxy and port information for HTTP and HTTPS protocols can be found in the Internet Explorer -> Internet options -> Connections -> LAN settings -> Advanced (j). The hosts that should bypass Proxy should be copied from the Exception list (k). Take a note of this information when Fiddler is closed.



To set up an Internet connection, click Setup.

Local Area Network (LAN) Settings

Automatic configuration

Automatic configuration may override manual settings. To ensure the use of manual settings, disable automatic configuration.

☑ Automatically detect settings

☐ Use automatic configuration script

Address:

Proxy server

☑ Use a proxy server for your LAN (These settings will not apply to dial-up or VPN connections).

Address: com-proxy.com Port: 8011

☑ Bypass proxy server for local addresses

Proxy Settings

Servers

| Type | Proxy address to use | Port |
|---|---|---|
| HTTP: | com-proxy.com | 8011 |
| Secure: | com-proxy.com | 8011 |
| FTP: | com-proxy.com | 8011 |
| Socks: | | |

☑ Use the same proxy server for all protocols

Exceptions

Do not use proxy server for addresses beginning with:

<-loopback>

Use semicolons ( ; ) to separate entries.

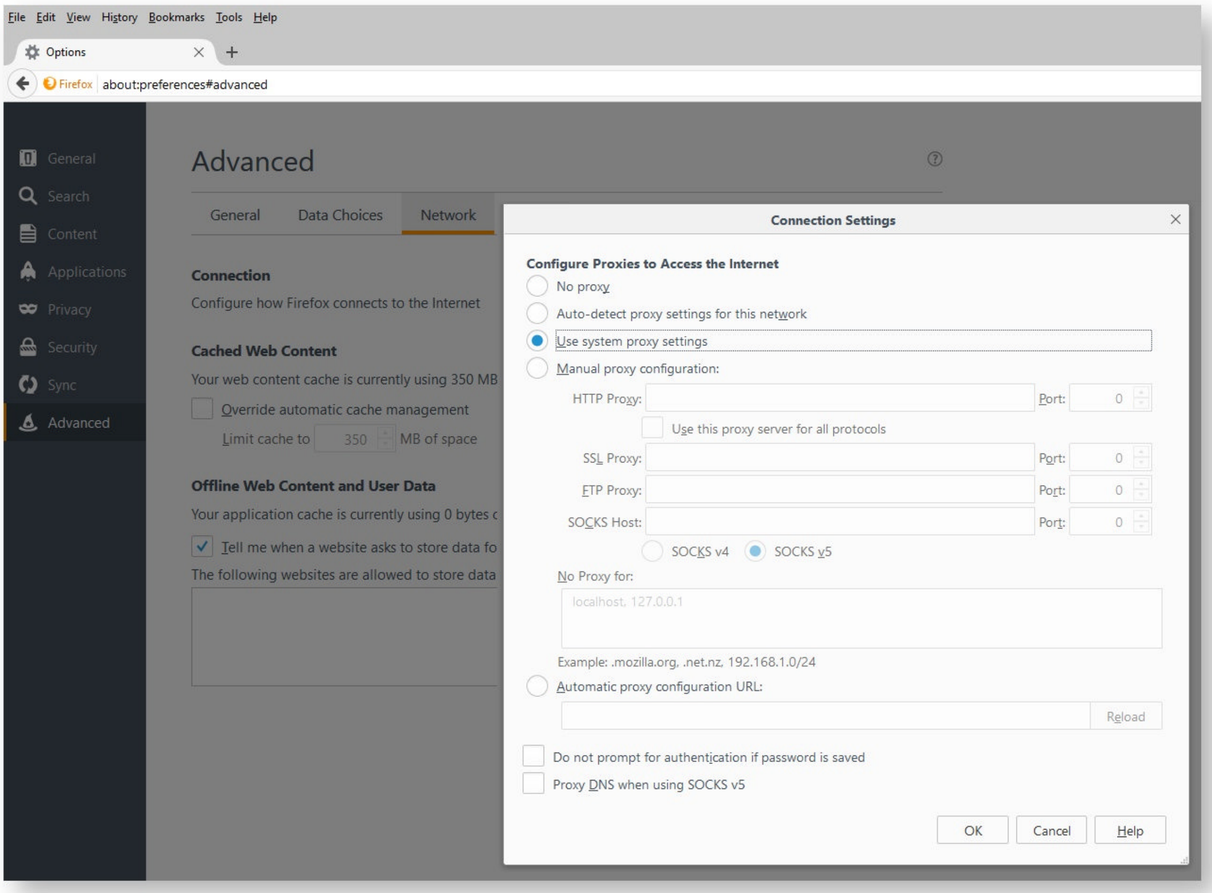
**Note:** If IE uses an auto-configuration script file for configuring corporate proxy, configure it to use the Fiddler proxy first. If the script is impossible to disable due to lack of permissions, you may need to use a different browser for recording, such as FireFox.

#### 4.1.1.5 Configuring Firefox

The StresStimulus recorder automatically routes web traffic from Firefox, however if it doesn't you can set the settings manually.

In Firefox go to **Tools > Options > Advanced**. Select the **Network** tab.

Under **Configuration**, click **Settings** and select the **Use system proxy settings** radio button.


File Edit View History Bookmarks Tools Help
Options
Firefox about:preferences#advanced
General
Search
Content
Applications
Privacy
Security
Sync
Advanced
General Data Choices Network
OK
Cancel
Help
Reload

Advanced

Connection
Configure how Firefox connects to the Internet

Cached Web Content
Your web content cache is currently using 350 MB
☐ Override automatic cache management
Limit cache to 350 MB of space

Offline Web Content and User Data
Your application cache is currently using 0 bytes c
☑ Tell me when a website asks to store data fo
The following websites are allowed to store data

Connection Settings

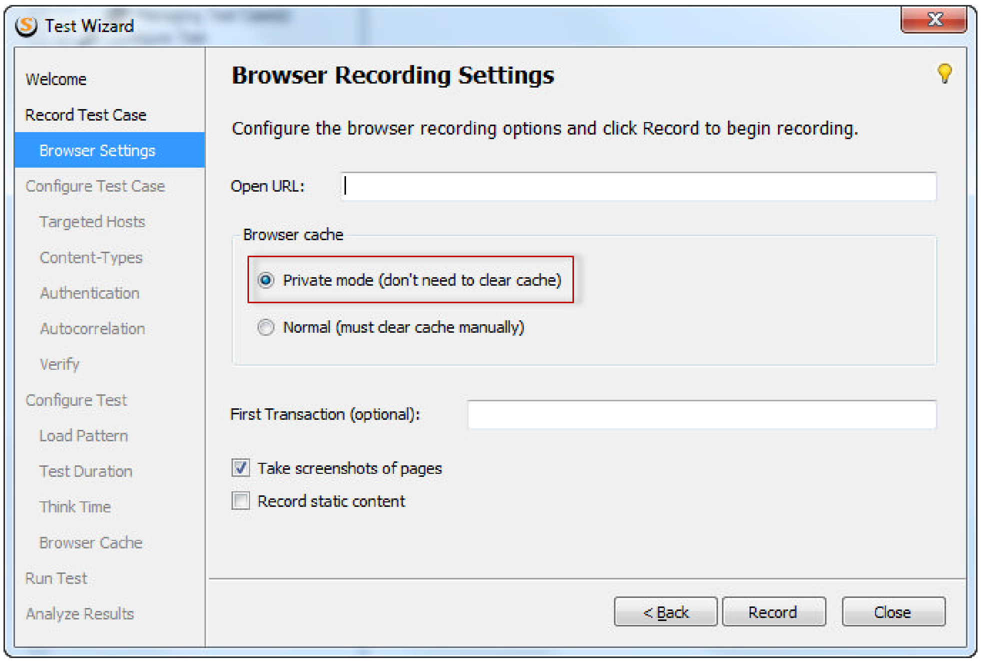
Configure Proxies to Access the Internet
○ No proxy
○ Auto-detect proxy settings for this network
● Use system proxy settings
○ Manual proxy configuration:
HTTP Proxy: Port: 0
☐ Use this proxy server for all protocols
SSL Proxy: Port: 0
FTP Proxy: Port: 0
SOCKS Host: Port: 0
○ SOCKS v4 ● SOCKS v5
No Proxy for:
localhost, 127.0.0.1
Example: .mozilla.org, .net.nz, 192.168.1.0/24
○ Automatic proxy configuration URL:
☐ Do not prompt for authentication if password is saved
☐ Proxy DNS when using SOCKS v5

In order to set up Firefox to capture HTTPS traffic click here.

#### 4.1.1.6 Automatic Clearing Cache in IE

When recording a test case from a web browser, using cache should be avoided to allow all requests to be recorded.

The easiest way to do this is to select **Private mode** that will launch IE or other browsers in Private Browsing mode. This way the browser will bypass the cache automatically.



Browser Recording Settings

Configure the browser recording options and click Record to begin recording.

Open URL:

Browser cache

[✓ Private mode (don't need to clear cache)]

Normal (must clear cache manually)

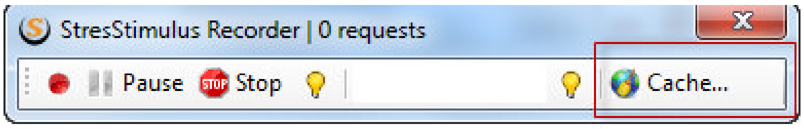
First Transaction (optional):

[✓] Take screenshots of pages

[ ] Record static content

However, some websites and web applications don't work without the browser cache. In this case, the cache must be cleared manually before recording starts.

To assist with clearing cache, the StresStimulus recorder has a browser cache settings shortcut (available for Internet Explorer only). Click the **Cache** button in the recorder to bring up the cache dialog.



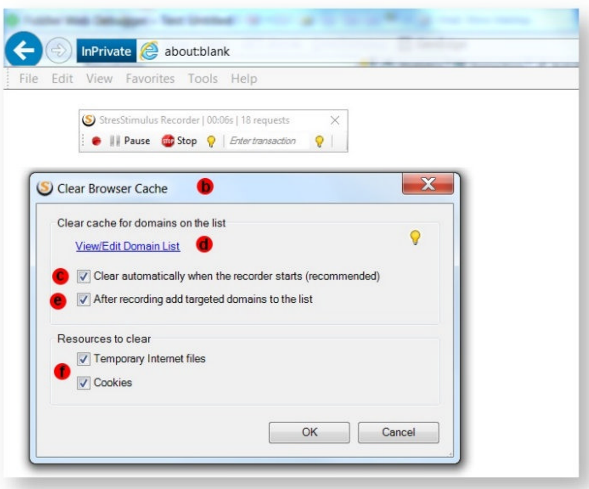
The recorder when working with Internet Explorer can automatically and selectively clear only cached resources related to the tested websites. Such targeted cache clearing is faster and allows to preserve other cached content.

StresStimulus maintains a Clear Cache Domain List. It includes names of the domains from which resources will be automatically purged from the browser cache before recording. To configuring cache clearing options, click **Cache** (a) on the recorder. A Clear Browser Cache dialog (b) will appear. Follow these steps:

- Check the Clear box (c) to automatically clear resources downloaded from the domains on the list. The resources will be purged from the browser cache before recording.
- Click the **View/Edit** link (d) to access the Clear Cache Domain list. This list can be also accessed from the Main Menu -> Hosts.

**Note:** The All hosts targeted during the test recording are automatically added to the Clear Cache Domain list, used for automatic browser cache clearing.

- Check the **After Recording** box (e) to automatically add new domains targeted during recording to the **Clear Cache for Domain** list. Next time the recorder will clear these domains from the cache as well.
- Select resources (f) you want to clear: temporary Internet files or/and cookies.


InPrivate
about:blank
File Edit View Favorites Tools Help
StresStimulus Recorder | 00:06s | 18 requests
Pause
Stop
Enter transaction
b
d
c
e
f
OK
Cancel

Clear Browser Cache

Clear cache for domains on the list

View/Edit Domain List

[✓] Clear automatically when the recorder starts (recommended)

[✓] After recording add targeted domains to the list

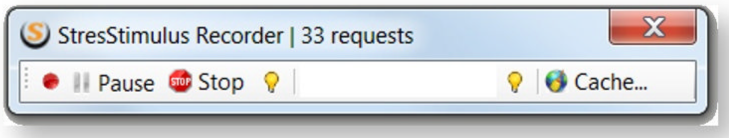
Resources to clear

[✓] Temporary Internet files

[✓] Cookies

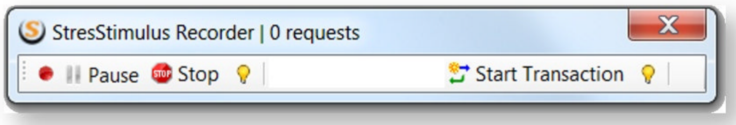
## 4.1.2 Recording from Other Sources

After the floating **Recorder** bar appeared, StresStimulus is ready to capture client actions from a non-browser recording source such as Silverlight or Adobe AIR application or for mobile apps. Navigate the tested application. When recording is finished, click **Stop** in the **Recorder** or in StresStimulus. A new test case will be created.



Unlike recording from a web browser, where StresStimulus automates several service functions, when recording from other sources, you are responsible to provide the following service functions manually.

- Make sure that the test case is recorded from the beginning of the client session. All initial steps, including session initiation information and login, must be captured. Otherwise, the test case will be incomplete that will cause errors.
- If your non-browser or mobile application uses client cache, make sure to clear it before recording. This is necessary to prevent the browser from using cached data, in which case, the test case will miss some HTTP sessions.
- StresStimulus will capture sessions from all processes on the machine without filtering, including unrelated to the test case sessions. During recording, close all browsers and other unnecessary HTTP clients. After recording is complete, delete all unrelated sessions from the test case.
- If you are recording from a mobile device and wish to add transactions during recording, you must click the **Start Transaction** button to create transaction after typing the transaction name.



### 4.1.2.1 Recording from Mobile Devices

StresStimulus can capture traffic from a mobile app. All mobile devices which allow you to set a proxy, such as an iPhone/iPad, Android, Windows phone or Blackberry are supported. The mobile device has to be connected through Wi-Fi to the same network that the StresStimulus machine is connected. In StresStimulus select **Mobile** from the record list. On the mobile device, set the network proxy settings to point to the StresStimulus machine's IP and set the port to **8888** if using Fiddler add-on or **49386** if using the standalone version (these are both the default ports and can be changed).

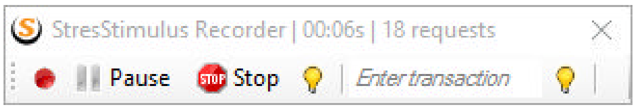
For more information on configuring Fiddler to capture traffic from another machine, check this source. Start navigating through a test scenario on the mobile device. When recording is finished, click **Stop** in the **Recorder** or in StresStimulus. A new test case will be created.

## 4.1.3 Creating Transactions

Ent SP

A transaction is a set of sequential requests representing a meaningful step in a test scenario. It is used to track performance characteristics of a specific business transaction that includes several user actions. Transactions add another level of performance tracking in addition to requests and pages.

You can define transactions while recording a test case or after the test case is created.

| | |
|---|---|
| To create the first transaction, enter its name in the Test Wizard when prompted.<br>To create a transaction during the recording, enter the transaction name in the textbox on the recorder and complete navigating through transaction steps.<br>Entering a new transaction name will designate the end of the current transaction and the beginning of the subsequent transaction.<br>To complete the current transaction without creating a new transaction, clear the transaction name from the box on the recorder.<br>Repeat these steps to create more transactions if necessary. You will be able to modify any transaction property after of the test case is created, as described in Transactions section. |  StresStimulus Recorder \| 00:06s \| 18 requests<br>Pause<br>Stop<br>Enter transaction |

## 4.1.4 Recording HTTPS

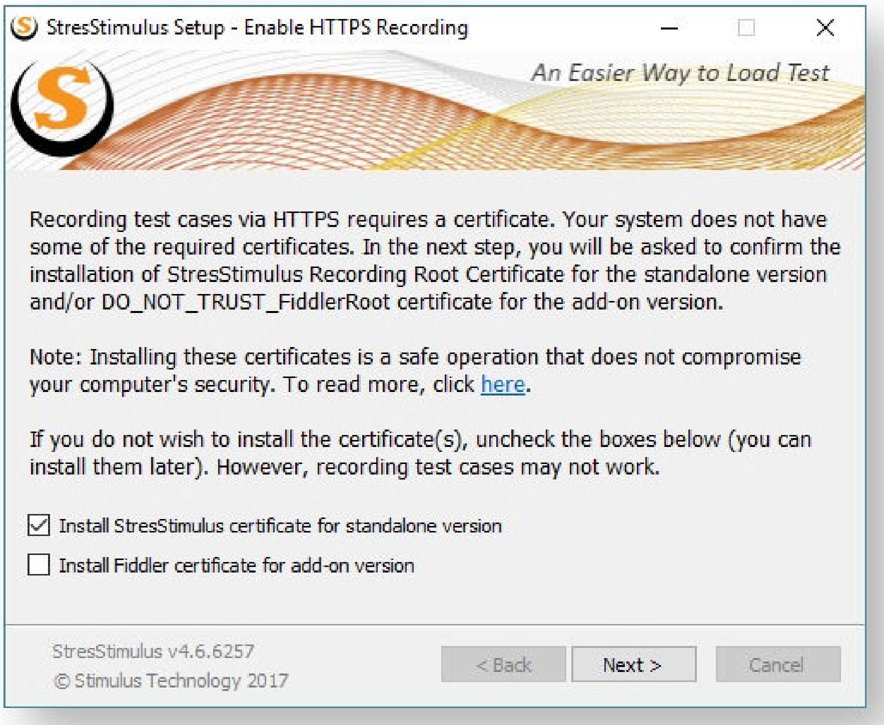
By default, support for HTTPS is not enabled in StresStimulus or in Fiddler. Before testing your first secure website, you need to enable support for HTTPS. This is a one-time process.

### 4.1.4.1 Enabling HTTPS in Standalone Version

In order to record HTTPS traffic, you need to trust the StresStimulus root certificate. This should be done only once per machine. There are several ways this can be done.

#### During Installation

During installation the installer will prompt to trust the StresStimulus root certificate. Tick the **Install StresStimulus certificate for standalone version** checkbox and click Next.


StresStimulus Setup - Enable HTTPS Recording

Recording test cases via HTTPS requires a certificate. Your system does not have some of the required certificates. In the next step, you will be asked to confirm the installation of StresStimulus Recording Root Certificate for the standalone version and/or DO_NOT_TRUST_FiddlerRoot certificate for the add-on version.

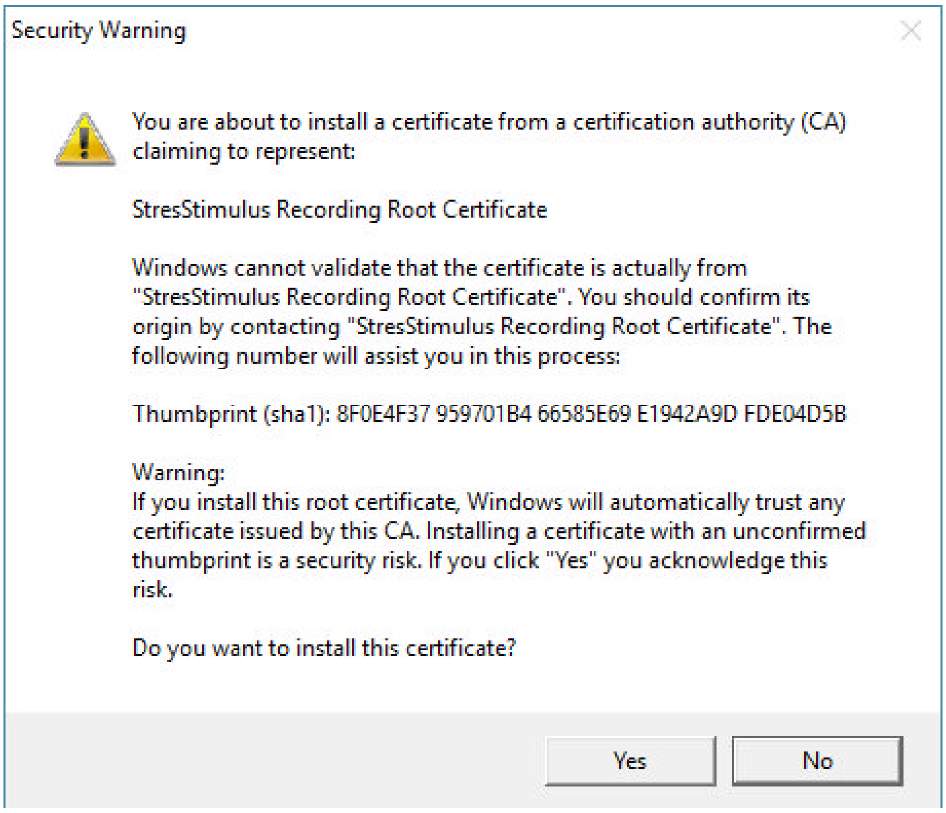
Note: Installing these certificates is a safe operation that does not compromise your computer's security. To read more, click here.

If you do not wish to install the certificate(s), uncheck the boxes below (you can install them later). However, recording test cases may not work.

- [✓] Install StresStimulus certificate for standalone version
- [ ] Install Fiddler certificate for add-on version

StresStimulus v4.6.6257
© Stimulus Technology 2017

A Windows confirmation dialog will appear next, where you can confirm that you trust the root certificate upon which it will be added to trusted Certificate Authorities.


Security Warning

You are about to install a certificate from a certification authority (CA) claiming to represent:

StresStimulus Recording Root Certificate

Windows cannot validate that the certificate is actually from "StresStimulus Recording Root Certificate". You should confirm its origin by contacting "StresStimulus Recording Root Certificate". The following number will assist you in this process:

Thumbprint (sha1): 8F0E4F37 959701B4 66585E69 E1942A9D FDE04D5B

Warning:
If you install this root certificate, Windows will automatically trust any certificate issued by this CA. Installing a certificate with an unconfirmed thumbprint is a security risk. If you click "Yes" you acknowledge this risk.

Do you want to install this certificate?

If you are also installing the StresStimulus Fiddler addon, you can also install the Fiddler Certificate root the same way by ticking the **Install Fiddler certificate for add-on version** checkbox. You will get another Windows confirmation dialog to confirm trusting the certificate.

### In the Application

If you chose not to trust StresStimulus root certificate, you can trust it inside StresStimulus application.

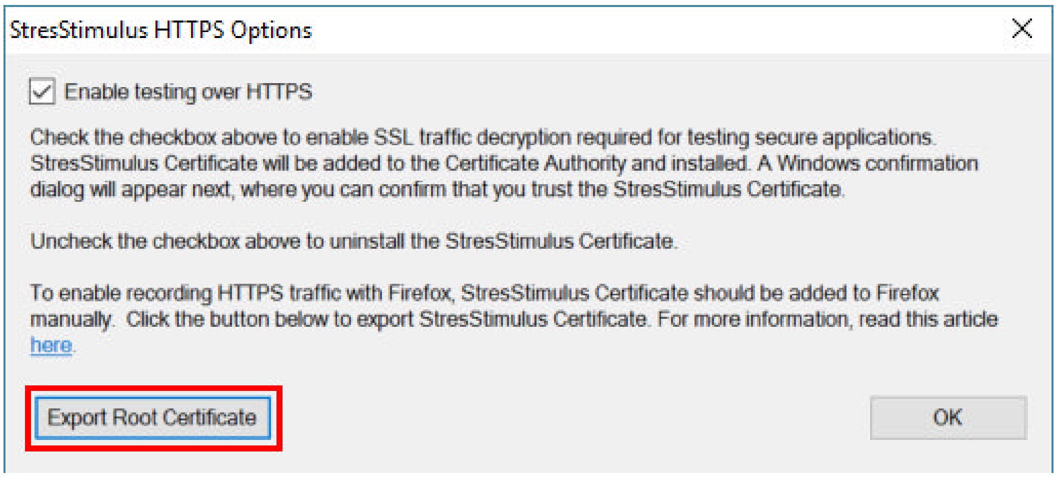
In StresStimulus menu go to **Options > HTTPS...** In the dialog, tick the **Enable testing over HTTPS** checkbox. The same Windows confirmation dialog will appear as above. Click Yes to trust StresStimulus certificate.

## Manually Enabling HTTPS

Sometimes the above two methods may not work for various reasons, including security settings. If this is the case, then your HTTPS traffic will not be captured. To fix this issue, the certificate trust should be enabled manually.

First, you need to export StresStimulus root certificate. In the StresStimulus main menu go to **Options > HTTPS...** to open the HTTPS options dialog.

Click the **Export Root Certificate** button and in the save file dialog, provide the certificate save location.


StresStimulus HTTPS Options

☑ Enable testing over HTTPS

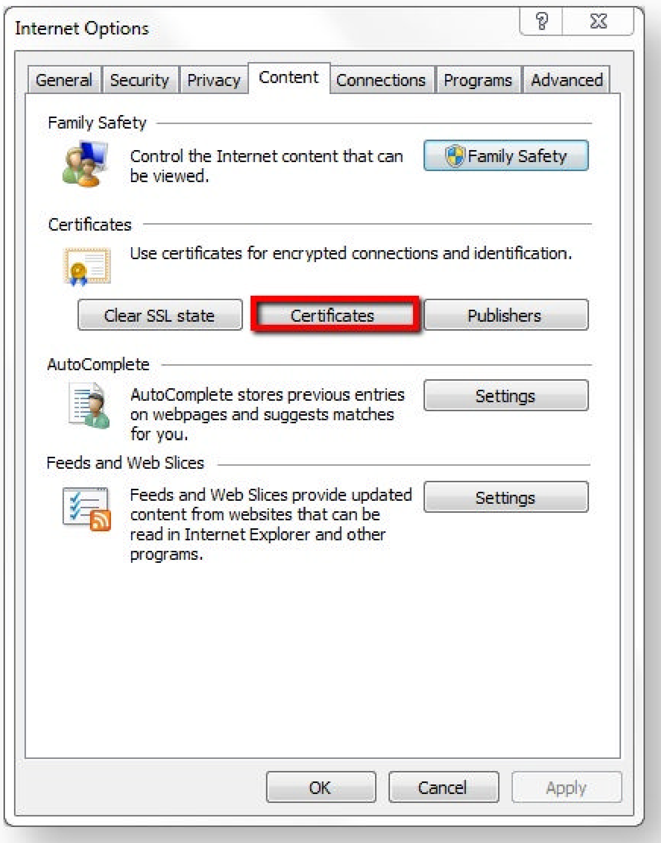
Check the checkbox above to enable SSL traffic decryption required for testing secure applications. StresStimulus Certificate will be added to the Certificate Authority and installed. A Windows confirmation dialog will appear next, where you can confirm that you trust the StresStimulus Certificate.

Uncheck the checkbox above to uninstall the StresStimulus Certificate.

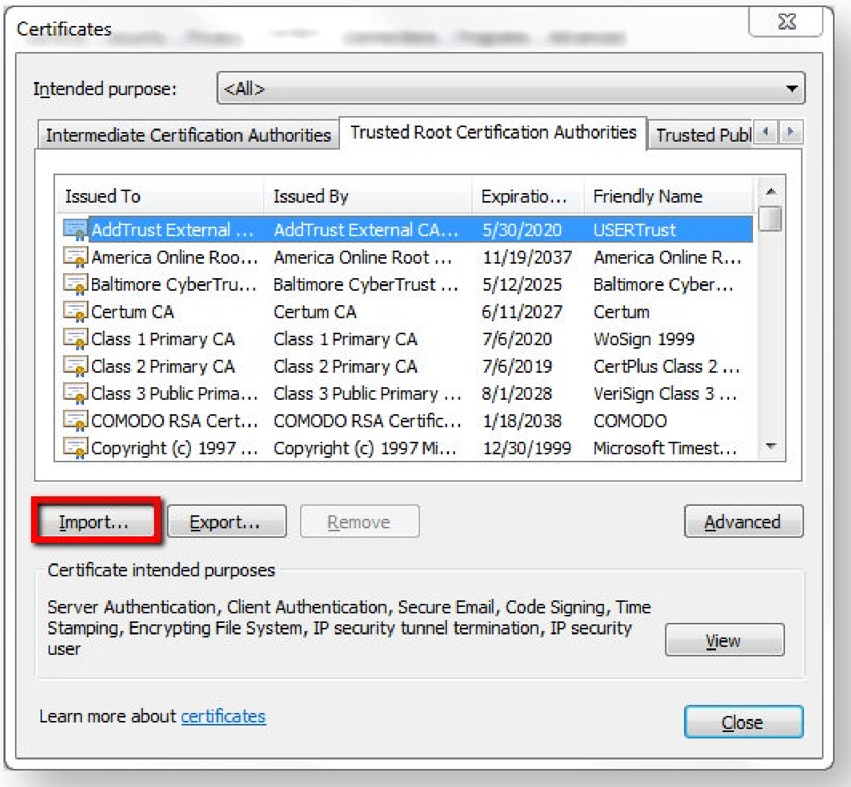
To enable recording HTTPS traffic with Firefox, StresStimulus Certificate should be added to Firefox manually. Click the button below to export StresStimulus Certificate. For more information, read this article here.

Open **Internet Explorer** and in the main menu go to **Tools > Internet Options**.

In the Internet Options dialog select the **Content Tab** and click the **Certificates** button.


Internet Options
General
Security
Privacy
Content
Connections
Programs
Advanced
Family Safety
Control the Internet content that can be viewed.
Family Safety
Certificates
Use certificates for encrypted connections and identification.
Clear SSL state
Certificates
Publishers
AutoComplete
AutoComplete stores previous entries on webpages and suggests matches for you.
Settings
Feeds and Web Slices
Feeds and Web Slices provide updated content from websites that can be read in Internet Explorer and other programs.
Settings
OK
Cancel
Apply

In the certificates dialog, select the **Trusted Root Certification Authorities** tab and click the Import... button.


Certificates
Intended purpose: <All>
Intermediate Certification Authorities
Trusted Root Certification Authorities
Trusted Publ

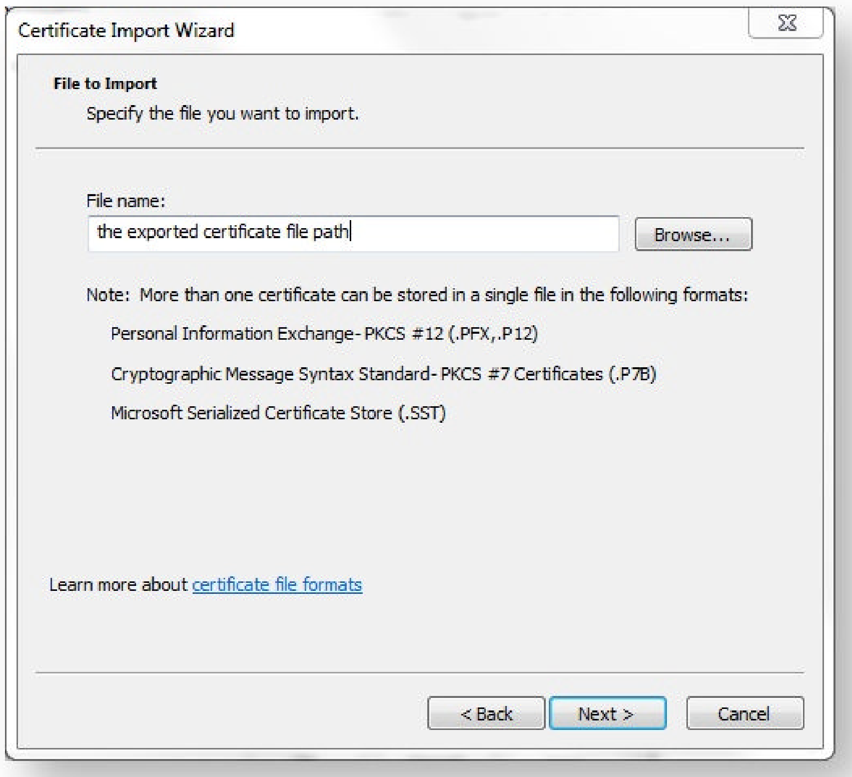
| Issued To | Issued By | Expiratio... | Friendly Name |
|---|---|---|---|
| AddTrust External ... | AddTrust External CA... | 5/30/2020 | USERTrust |
| America Online Roo... | America Online Root ... | 11/19/2037 | America Online R... |
| Baltimore CyberTru... | Baltimore CyberTrust ... | 5/12/2025 | Baltimore Cyber... |
| Certum CA | Certum CA | 6/11/2027 | Certum |
| Class 1 Primary CA | Class 1 Primary CA | 7/6/2020 | WoSign 1999 |
| Class 2 Primary CA | Class 2 Primary CA | 7/6/2019 | CertPlus Class 2 ... |
| Class 3 Public Prima... | Class 3 Public Primary ... | 8/1/2028 | VeriSign Class 3 ... |
| COMODO RSA Cert... | COMODO RSA Certific... | 1/18/2038 | COMODO |
| Copyright (c) 1997 ... | Copyright (c) 1997 Mi... | 12/30/1999 | Microsoft Timest... |

Certificate intended purposes

Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping, Encrypting File System, IP security tunnel termination, IP security user

Learn more about certificates

Follow the certificate wizard. In the second step, specify the file path of the previously exported certificate. Click next until the wizard completes the import.


Certificate Import Wizard

File to Import

Specify the file you want to import.

File name:

the exported certificate file path

Note: More than one certificate can be stored in a single file in the following formats:

Personal Information Exchange- PKCS #12 (.PFX,.P12)

Cryptographic Message Syntax Standard- PKCS #7 Certificates (.P7B)

Microsoft Serialized Certificate Store (.SST)

Learn more about certificate file formats

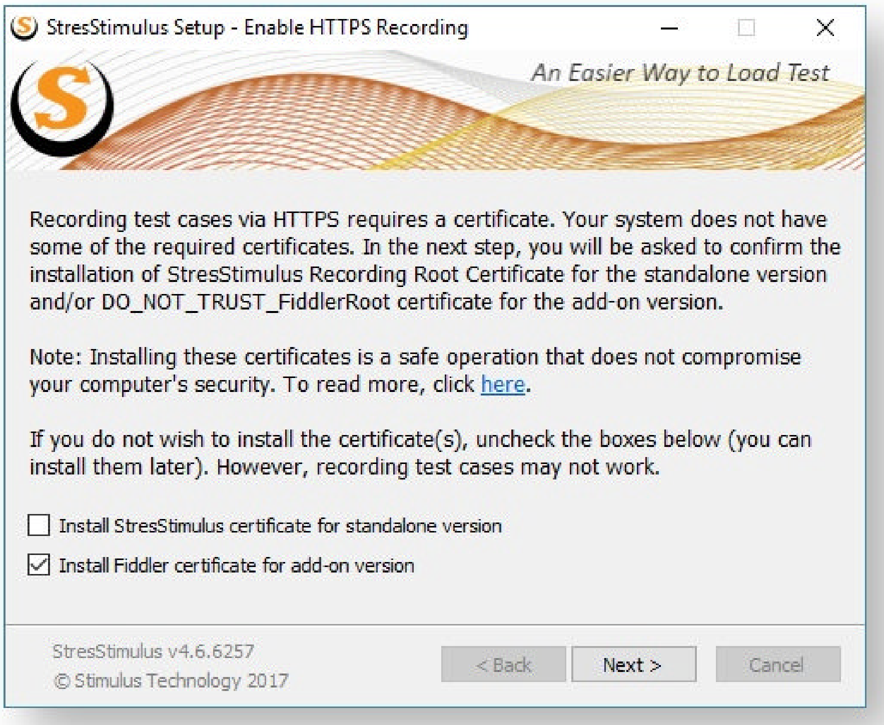
If you are using Firefox browser for recording then click here to see how to enable HTTPS in Firefox.

### 4.1.4.2 Enabling HTTPS in Fiddler Version

In order to record HTTPS traffic, you need to trust the Fiddler root certificate. This should be done only once per machine. There are several ways this can be done.

#### During Installation

During installation the installer will prompt to trust the Fiddler root certificate. Tick the **Install Fiddler certificate for add-on version** checkbox and click Next.


StresStimulus Setup - Enable HTTPS Recording

Recording test cases via HTTPS requires a certificate. Your system does not have some of the required certificates. In the next step, you will be asked to confirm the installation of StresStimulus Recording Root Certificate for the standalone version and/or DO_NOT_TRUST_FiddlerRoot certificate for the add-on version.

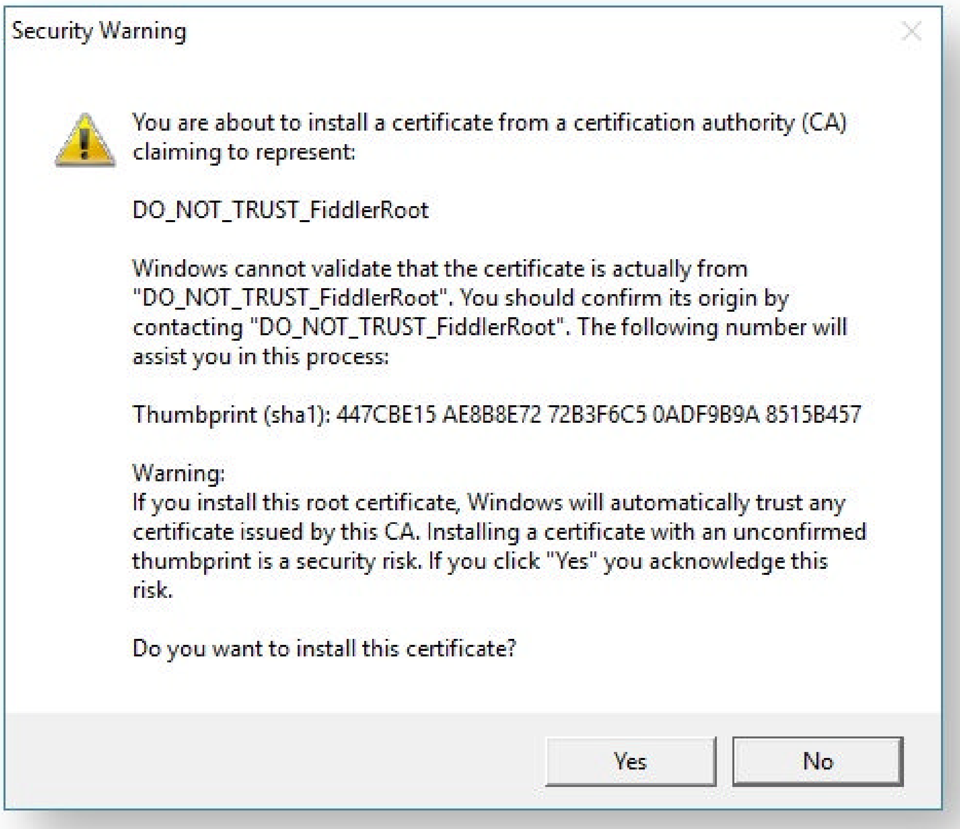
Note: Installing these certificates is a safe operation that does not compromise your computer's security. To read more, click here.

If you do not wish to install the certificate(s), uncheck the boxes below (you can install them later). However, recording test cases may not work.

- [ ] Install StresStimulus certificate for standalone version
- [x] Install Fiddler certificate for add-on version

StresStimulus v4.6.6257
© Stimulus Technology 2017

A Windows confirmation dialog will appear next, where you can confirm that you trust the root certificate upon which it will be added to trusted Certificate Authorities.



Security Warning

You are about to install a certificate from a certification authority (CA) claiming to represent:

DO_NOT_TRUST_FiddlerRoot

Windows cannot validate that the certificate is actually from "DO_NOT_TRUST_FiddlerRoot". You should confirm its origin by contacting "DO_NOT_TRUST_FiddlerRoot". The following number will assist you in this process:

Thumbprint (sha1): 447CBE15 AE8B8E72 72B3F6C5 0ADF9B9A 8515B457

Warning:
If you install this root certificate, Windows will automatically trust any certificate issued by this CA. Installing a certificate with an unconfirmed thumbprint is a security risk. If you click "Yes" you acknowledge this risk.

Do you want to install this certificate?

You can also install the StresStimulus Certificate root the same way by ticking the **Install StresStimulus certificate for standalone version** checkbox. You will get another Windows confirmation dialog to confirm trusting the certificate.

## In Fiddler

When using the add-on version, Fiddler should be configured to decrypt HTTPS traffic. Follow these steps:

1. In Fiddler's main menu select **Tools** -> **Fiddler Options** and click the **HTTPS** tab.

2. Check three boxes:

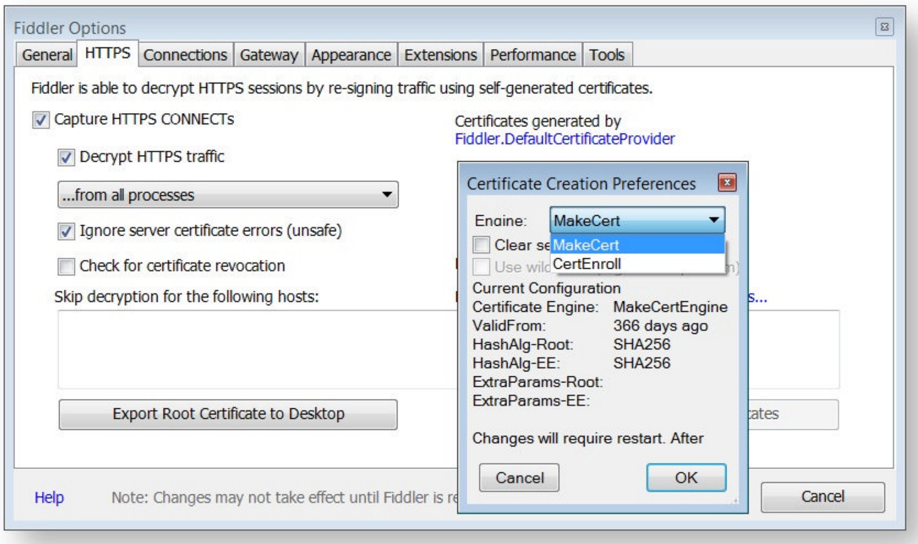
- 
  - **Capture HTTPS CONNECTs**

- **Decrypt HTTPs traffic**
- **Ignore server certificate errors** and click **OK**

3. Restart Fiddler
4. If the recorder was open, click **Close Recorder** and then re-open it from StresStimulus.

**Note:** Fiddler uses a "man-in-the-middle" technique in order to decrypt HTTPS traffic. For more details check this source.

Sometimes after enabling HTTPS and with Fiddler running, secure pages do not appear in the browser and an error message is displayed instead. In some cases, this happens because the Fiddler certificate was created using incorrect settings. Make sure the MakeCert engine is used to generate the certificates. For more information click here.



Fiddler is able to decrypt HTTPS sessions by re-signing traffic using self-generated certificates.

☑ Capture HTTPS CONNECTs

☑ Decrypt HTTPS traffic

...from all processes

☑ Ignore server certificate errors (unsafe)

☐ Check for certificate revocation

Skip decryption for the following hosts:

Certificates generated by Fiddler.DefaultCertificateProvider

Note: Changes may not take effect until Fiddler is re

Engine: MakeCert

☐ Clear se

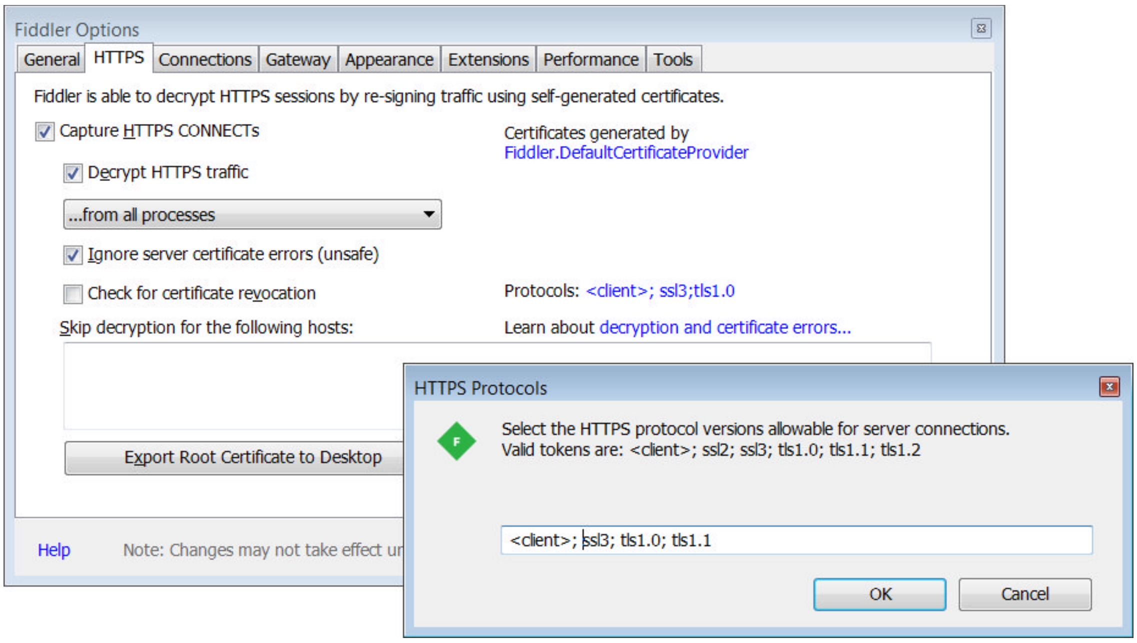
☐ Use wild

Current Configuration

| | |
|---|---|
| Certificate Engine: | MakeCertEngine |
| ValidFrom: | 366 days ago |
| HashAlg-Root: | SHA256 |
| HashAlg-EE: | SHA256 |
| ExtraParams-Root: | |
| ExtraParams-EE: | |

Changes will require restart. After

In some cases, the error pages displayed in the browser because your Web server uses HTTPS protocols that are not enabled in Fiddler by default. after installation, Fiddler is configured to support SSL3 and TLS1.0. For example, if your server uses TLS1.1, then to add this protocol, click the Protocols link and in the appeared dialog append the missing protocol.



Fiddler is able to decrypt HTTPS sessions by re-signing traffic using self-generated certificates.

☑ Capture HTTPS CONNECTs

☑ Decrypt HTTPS traffic

...from all processes

☑ Ignore server certificate errors (unsafe)

☐ Check for certificate revocation

Skip decryption for the following hosts:

Certificates generated by Fiddler.DefaultCertificateProvider

Protocols: <client>; ssl3;tls1.0

Learn about decryption and certificate errors...

Note: Changes may not take effect u

HTTPS Protocols

Select the HTTPS protocol versions allowable for server connections.
Valid tokens are: <client>; ssl2; ssl3; tls1.0; tls1.1; tls1.2

<client>; ssl3; tls1.0; tls1.1

### 4.1.4.3 HTTPS on mobile devices

When recording HTTPS traffic from mobile devices the first step is to download the StresStimulus root certificate. You need to install the root certificate only once on a device.

#### Standalone version

1. On the mobile device, open a browser and navigate to the **http://{StresStimulus_IP}:{StresStimulus_Port}/root.cer**
   a. StresStimulus_IP is the IP address of the machine that has StresStimulus running
   b. StresStimulus_port is the listening port of StresStimulus, 49386 by default.
2. Accept all the prompts to add this certificate to the trusted repository list.
3. Navigate to the website or app you want to test.

#### Fiddler a dd-on version

- For iOS devices: section **Decrypt HTTPS Traffic from iOS Devices** in this article.
- For Android devices: section **Decrypt HTTPS** in this article.

### 4.1.4.4 Troubleshooting HTTPS issues

Sometimes during recording, HTTPS pages do not appear in the browser and an error message is displayed instead.

In some cases, this happens because a required security protocol is not enabled. For example, the web server uses a security protocol such as TLS 1.2 which by default is not enabled in Fiddler (or StresStimulus). This situation is described here.

To fix the situation, follow these steps:

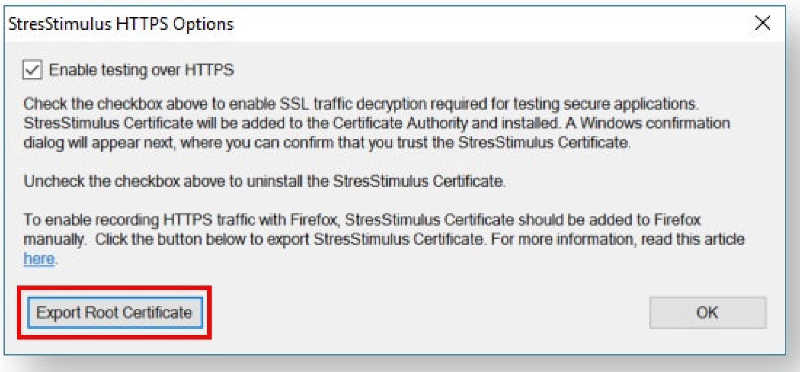
1. Make sure that you have installed the latest version of Fiddler 4. Fiddler 4 is available here
2. Start Fiddler
3. Hit Ctrl+R to bring up the custom rules editor.
4. In the appeared script, find the "OnBeforeRequest" method and add the following line inside:

```
      oSession["x-OverrideSslProtocols"] = " ssl3;tls1.0;tls1.1;tls1.2";
5. Save the script
6. Use the StresStimulus add-on version. Try recording the test case with HTTPS
pages again
```

### 4.1.4.5 Capture HTTPS traffic from Firefox

First, export the StresStimulus certificate. In the StresStimulus main menu go to **Options > HTTPS...** to open the HTTPS options dialog.

Click the **Export Root Certificate** button and in the save file dialog, provide the location where you want to save the certificate.


StresStimulus HTTPS Options

☑ Enable testing over HTTPS

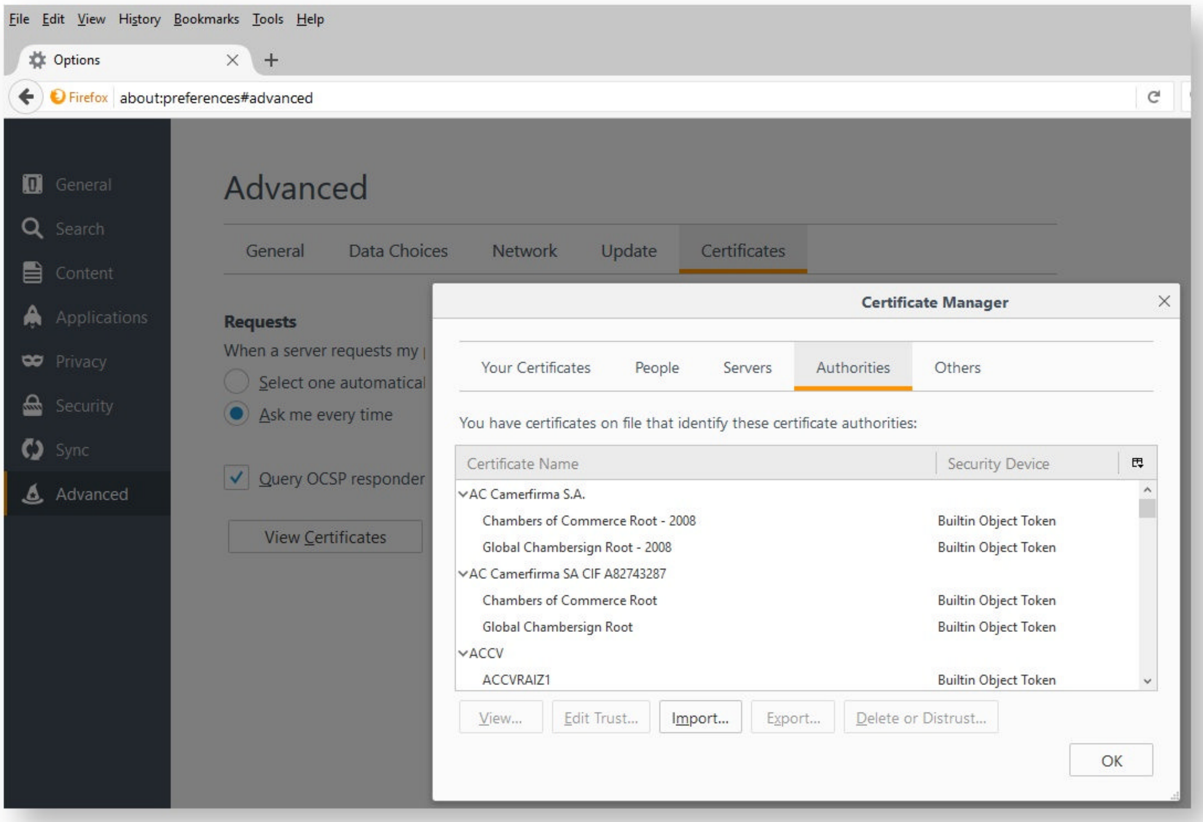
Check the checkbox above to enable SSL traffic decryption required for testing secure applications. StresStimulus Certificate will be added to the Certificate Authority and installed. A Windows confirmation dialog will appear next, where you can confirm that you trust the StresStimulus Certificate.

Uncheck the checkbox above to uninstall the StresStimulus Certificate.

To enable recording HTTPS traffic with Firefox, StresStimulus Certificate should be added to Firefox manually. Click the button below to export StresStimulus Certificate. For more information, read this article here.

In Firefox go to **Tools > Options > Advanced**. Select the **Certificates** tab. Click the **View Certificates** button. Go to the **Authorities** tab.

Click the **Import** button and provide the saved certificate's location.


File Edit View History Bookmarks Tools Help
Options
Firefox about:preferences#advanced
General
Search
Content
Applications
Privacy
Security
Sync
Advanced
General Data Choices Network Update Certificates
View Certificates
Your Certificates People Servers Authorities Others
View... Edit Trust... Import... Export... Delete or Distrust...
OK

Advanced

Requests

When a server requests my

Select one automatical

[✓ Ask me every time]

[✓] Query OCSP responder

Certificate Manager

You have certificates on file that identify these certificate authorities:

| Certificate Name | Security Device |
|---|---|
| AC Camerfirma S.A. | |
| Chambers of Commerce Root - 2008 | Builtin Object Token |
| Global Chambersign Root - 2008 | Builtin Object Token |
| AC Camerfirma SA CIF A82743287 | |
| Chambers of Commerce Root | Builtin Object Token |
| Global Chambersign Root | Builtin Object Token |
| ACCV | |
| ACCVRAIZ1 | Builtin Object Token |

In order to enable HTTPS in Firefox for Fiddler, refer to this link http://docs.telerik.com/fiddler/configure-fiddler/tasks/firefoxhttps.

### 4.1.5 Troubleshooting Recording

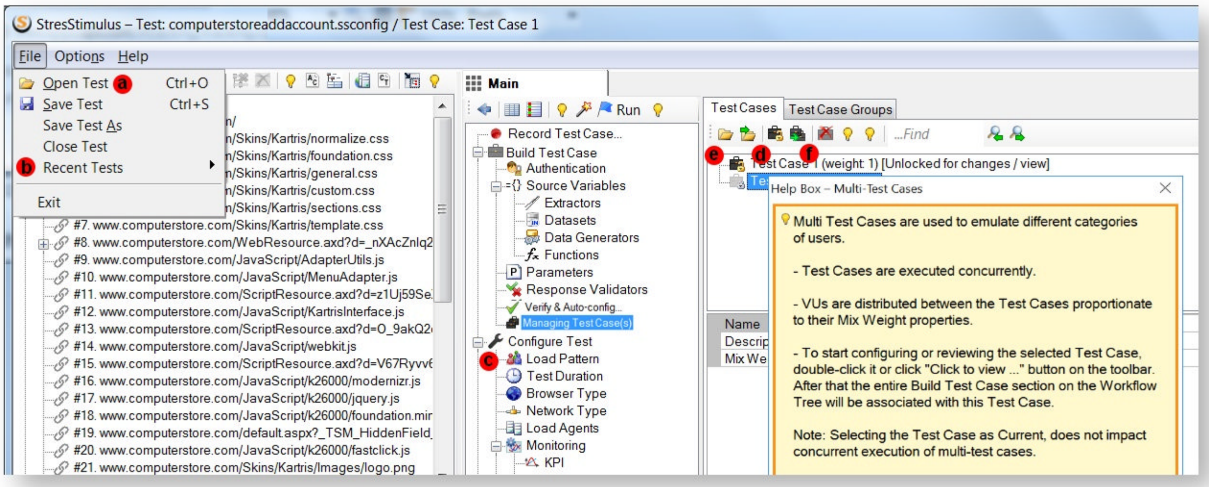
The following table will assist you with troubleshooting your recording problem.

| **Standalone Version** | | **Add-on Version** | |
|---|---|---|---|
| **IE or Other Clients** | | **IE or Other Clients** | |
| Troubleshooting Proxy Settings | Proxy Settings in Standalone Version | Troubleshooting Proxy Settings | Proxy Settings in Fiddler Version |
| Troubleshooting HTTPS Settings | Enabling HTTPS | Troubleshooting HTTPS Settings | Enabling HTTPS in Fiddler Version |
| **Firefox** | | **Firefox** | |
| Troubleshooting Proxy Settings | Configuring Firefox | Troubleshooting Proxy Settings | Configuring Firefox for Fiddler |

## 4.2 Other Methods of Creating Test Case

### 4.2.1 Open test

To open a previously saved Test, select **Open Test** (a) or **Recent Tests** (b) from the StresStimulus Main Menu. From the **Select Recording Source** step in the **Test Wizard,** select **Open Test.**



Multi Test Cases are used to emulate different categories of users.

- Test Cases are executed concurrently.

- VUs are distributed between the Test Cases proportionate to their Mix Weight properties.

- To start configuring or reviewing the selected Test Case, double-click it or click "Click to view ..." button on the toolbar. After that the entire Build Test Case section on the Workflow Tree will be associated with this Test Case.

Note: Selecting the Test Case as Current, does not impact concurrent execution of multi-test cases.

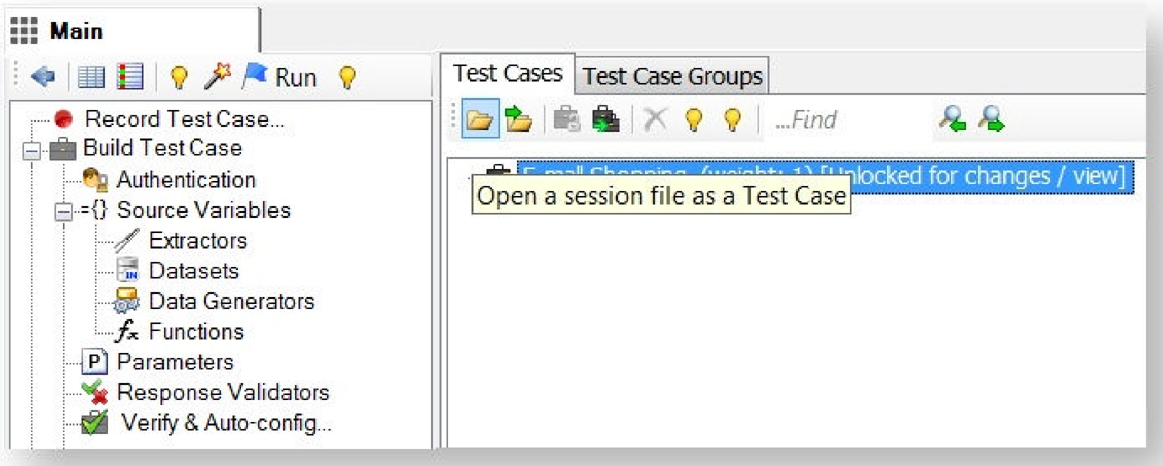
### 4.2.2 Import from another test

To import Test Cases from another Test, in the **Managing Test Case(s)** section (c), click **Import Test Cases From Another Test** (d) on the toolbar and then select a .ssconfig file and click **Open**. All test cases from the selected test will be imported into the current test. All test objects will be copied into the .ssconfig file of the current test, and all .saz files with sessions will be copied into the current folder.

**Tip:** If you need to import some, but not all test cases, after import, delete the test cases you do not need.

## 4.2.3 Import session file

To import a previously saved session file, in the **Managing Test Case(s)** section, click **Open a session file as a Test Case** and select a Fiddler file (.saz) or HTTP archive file (.har). The selected file will be imported as a new Test Case.



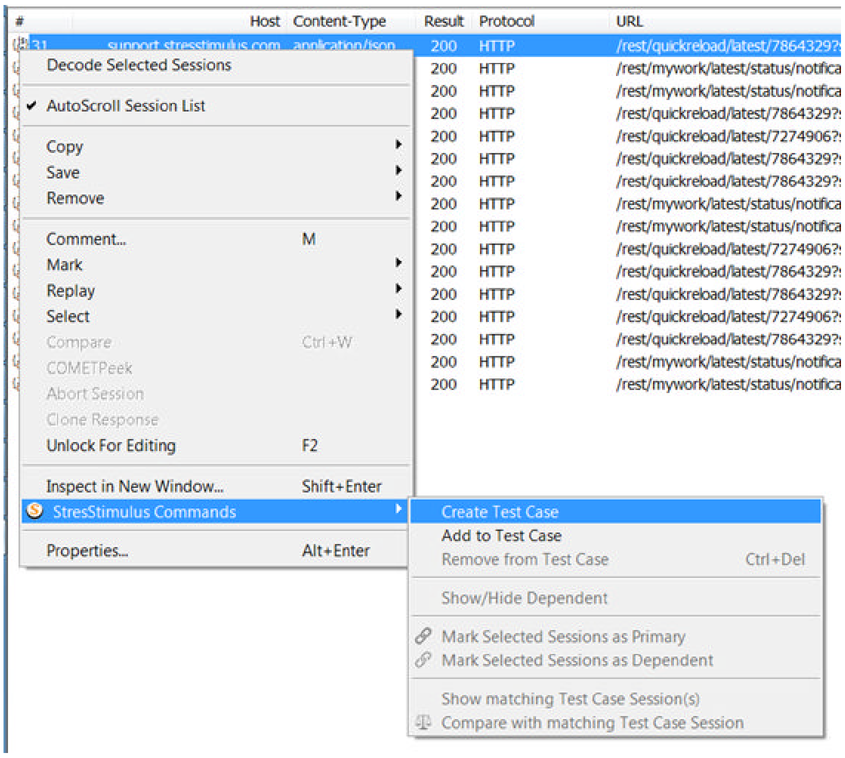
## 4.2.4 Clone test case

To clone the selected Test Case, in the **Managing Test Case(s)** section, click **Clone Test Case** (f) on the toolbar.

## 4.2.5 Add additional sessions to the existing or new test case

To create a test case using sessions captured in the session grid or add to existing test case,

1. Capture sessions in Fiddler or Load Archive or Import Sessions into Fiddler. Or in standalone version use the composer to create a session.
2. In the Sessions grid, select the sessions.
3. Right-click and in the context menu under **StresStimulus Commands**, click **Create Test Case** to create a new test case or **Add to Test Case** to add selected sessions to the test case.



| # | Host | Content-Type | Result | Protocol | URL |
|---|---|---|---|---|---|
| 31 | support.stresstimulus.com | application/json | 200 | HTTP | /rest/quickreload/latest/7864329? |
| | | | 200 | HTTP | /rest/mywork/latest/status/notifica |
| | | | 200 | HTTP | /rest/mywork/latest/status/notifica |
| | | | 200 | HTTP | /rest/quickreload/latest/7864329? |
| | | | 200 | HTTP | /rest/quickreload/latest/7274906? |
| | | | 200 | HTTP | /rest/quickreload/latest/7864329? |
| | | | 200 | HTTP | /rest/quickreload/latest/7864329? |
| | | | 200 | HTTP | /rest/mywork/latest/status/notifica |
| | | | 200 | HTTP | /rest/mywork/latest/status/notifica |
| | | | 200 | HTTP | /rest/quickreload/latest/7274906? |
| | | | 200 | HTTP | /rest/quickreload/latest/7864329? |
| | | | 200 | HTTP | /rest/quickreload/latest/7864329? |
| | | | 200 | HTTP | /rest/quickreload/latest/7274906? |
| | | | 200 | HTTP | /rest/quickreload/latest/7864329? |
| | | | 200 | HTTP | /rest/mywork/latest/status/notifica |
| | | | 200 | HTTP | /rest/mywork/latest/status/notifica |

## 4.3 Post recording steps

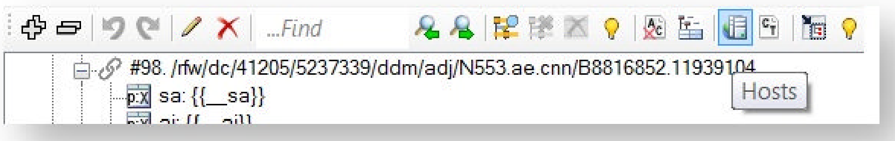
### 4.3.1 Purging requests to unwanted hosts

Some websites use third party web services (such as Google Analytics) that should be excluded from the load test. Also, during recording, a web browser add-on may generate unrelated traffic that also should be excluded.

When Test Wizard reached the **Targeted Hosts** step, a list of hosts accessed during the recording will be displayed.

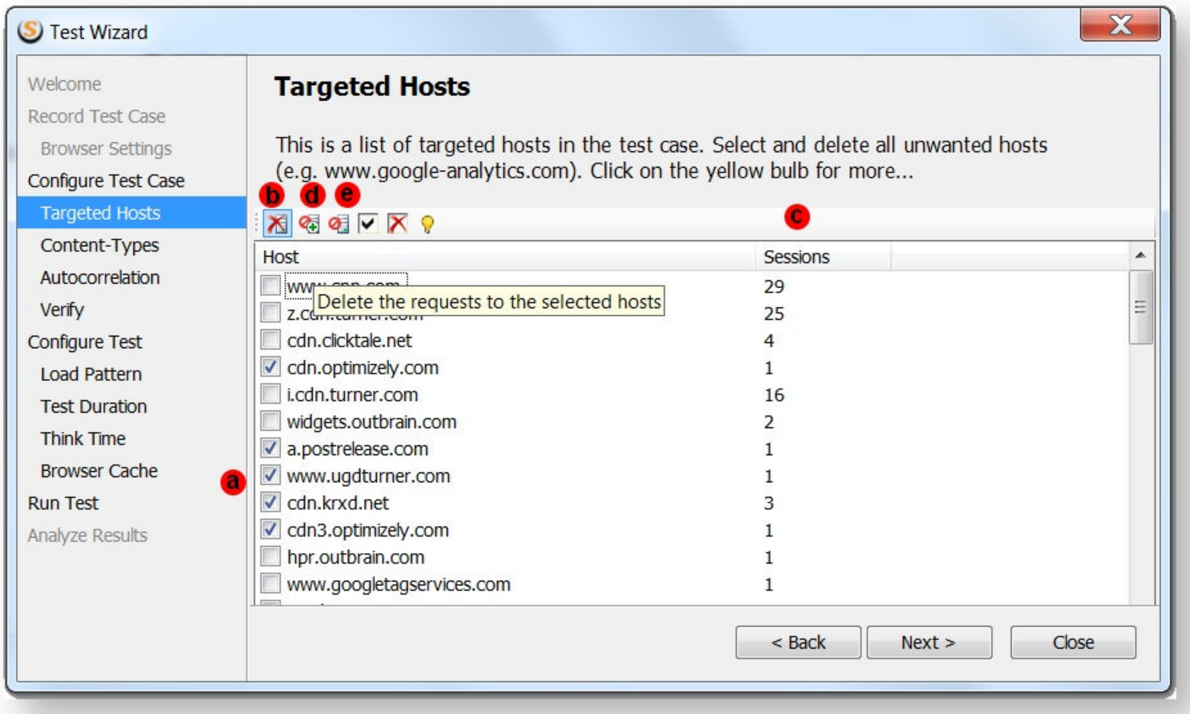
**Note:** Requests to the hosts added to the **Excluded Hosts List** before the recording, will be automatically deleted.

**Tip:** You can also display the **Test Case Hosts** dialog by clicking **Hosts** on the Test Case Tree toolbar.


...Find
#98. /rfw/dc/41205/5237339/ddm/adj/N553.ae.cnn/B8816852.11939104
sa: {{__sa}}
Hosts

There are two ways to remove requests to the unwanted hosts.

- **Manual purging.** To delete sessions sent to hosts which do not require testing, from the test case, in the **Test Case Hosts** list, check the boxes (a) on the left of the host names and click **Delete** (b) on the toolbar.



**Targeted Hosts**

This is a list of targeted hosts in the test case. Select and delete all unwanted hosts (e.g. www.google-analytics.com). Click on the yellow bulb for more...

| | Host | Sessions |
|---|---|---|
| ☐ | www.cnn.com | 29 |
| ☐ | z.cdn.turner.com | 25 |
| ☐ | cdn.clicktale.net | 4 |
| ✓ | cdn.optimizely.com | 1 |
| ☐ | i.cdn.turner.com | 16 |
| ☐ | widgets.outbrain.com | 2 |
| ✓ | a.postrelease.com | 1 |
| ✓ | www.ugdturner.com | 1 |
| ✓ | cdn.krxd.net | 3 |
| ✓ | cdn3.optimizely.com | 1 |
| ☐ | hpr.outbrain.com | 1 |
| ☐ | www.googletagservices.com | 1 |

**Tip:** Hosts that do not need to be tested often have a smaller number of sessions. To locate such hosts, sort the Test Case Hosts list by the **Session** column (c) which displays the number of captured sessions per host.

- **Automatic purging.** After every re-recording, sessions which target unwanted hosts should be deleted again. To automate this process, StresStimulus maintains the **Excluded Hosts List**. Requests to these hosts will be automatically deleted in subsequent recordings. To add the selected hosts to the **Excluded Hosts List**, click the button (d) on the toolbar. To edit the

**Excluded Hosts List**, click the button (e). You can access the **Excluded Hosts List** from Stimulus Main Menu -> Host/Content-types -> Excluded Hosts. This list can have the full host name or host name with wildcards.

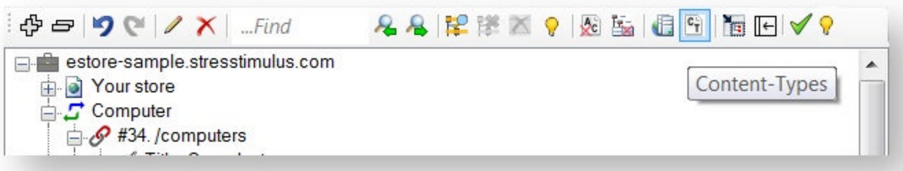
### 4.3.2 Purging sessions with unwanted content types

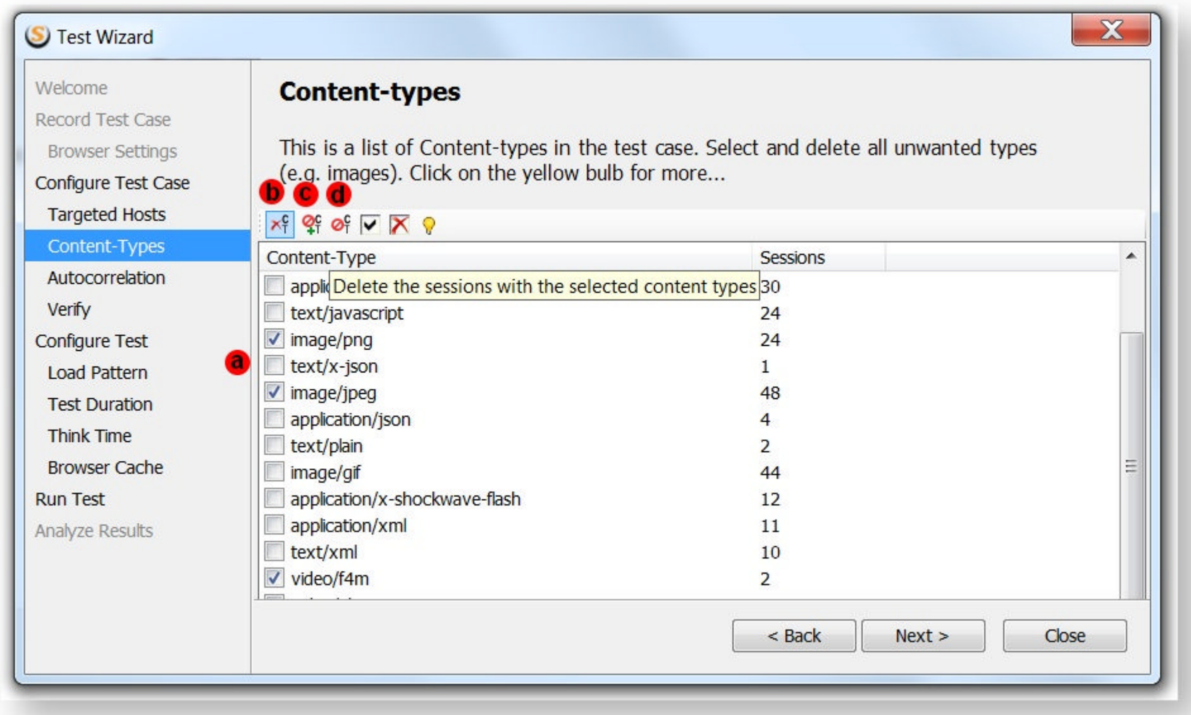
Certain web resources create little impact on website performance and can be excluded from the load test. You can filter out HTTP sessions with certain content types that you aren't interested in testing.

When Test Wizard reaches the **Content-Types** step, a list of hosts used in during the recording will be displayed.

**Note:** Sessions with the content types added to the **Excluded Content Types List** before the recording will be automatically deleted.

**Tip:** You can also display the **Content Types** dialog by clicking **Content Types** on the Test Case Tree toolbar.





**Content-types**

This is a list of Content-types in the test case. Select and delete all unwanted types (e.g. images). Click on the yellow bulb for more...

Delete the sessions with the selected content types

| | Content-Type | Sessions |
|---|---|---|
| ☐ | appli | 30 |
| ☐ | text/javascript | 24 |
| ☑ | image/png | 24 |
| ☐ | text/x-json | 1 |
| ☑ | image/jpeg | 48 |
| ☐ | application/json | 4 |
| ☐ | text/plain | 2 |
| ☐ | image/gif | 44 |
| ☐ | application/x-shockwave-flash | 12 |
| ☐ | application/xml | 11 |
| ☐ | text/xml | 10 |
| ☑ | video/f4m | 2 |

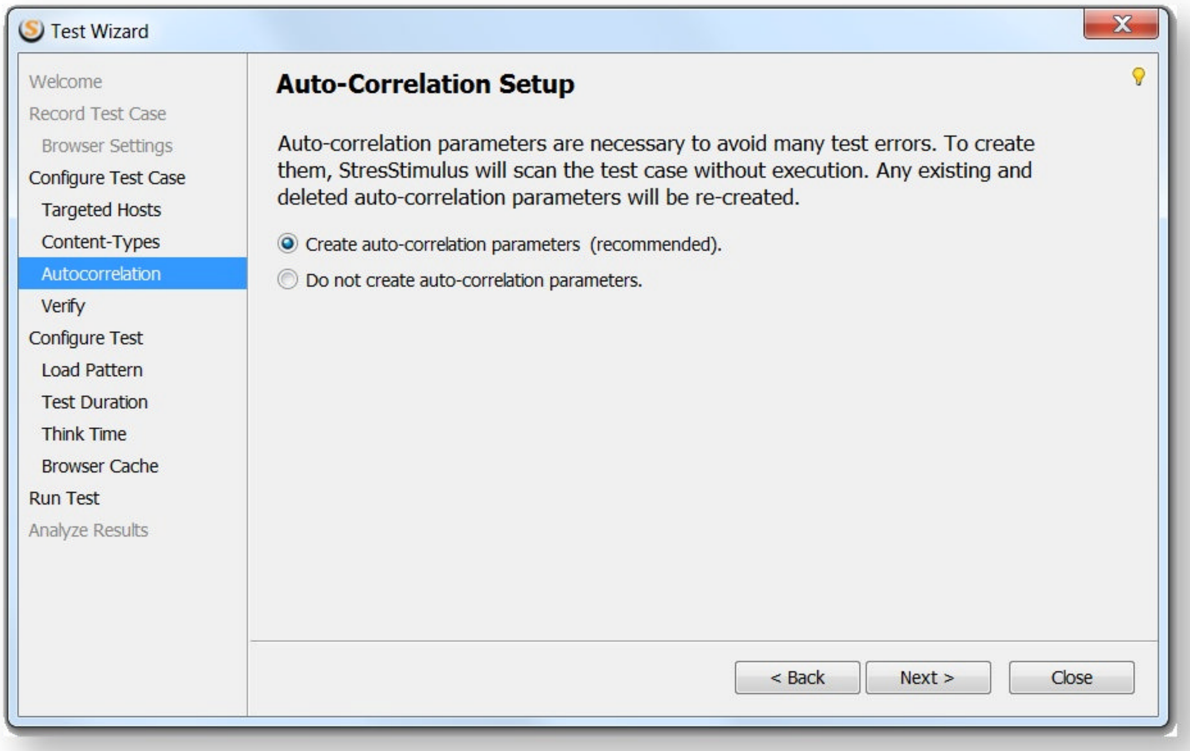
There are two ways to remove sessions with unwanted content types.

- **Manual purging.** To delete sessions from the test case with content types that do not require testing, in the **Content-type filter** list, check the boxes (a) on the left of the content types names and click **Delete** (b) on the toolbar.

- **Automatic purging .** After every re-recording, sessions with unwanted content types should be deleted again. To automate this process, StresStimulus maintains the **Excluded Content Type List**. Sessions with content types from this list will be automatically deleted in subsequent recordings. To add the selected content types to the list click the button (c) on the toolbar. To edit the **Excluded Content Type List**, click the button (d). You can access and edit the Excluded Content Type List after the recording from Stimulus Main Menu -> Host/Content-types -> Excluded Content-types. This list can have the content type or content type with wildcards.

### 4.3.3 Running Autocorrelation

After the test case is recorded, the test wizard will offer to create autocorrelation parameters. They are necessary to avoid many test errors.



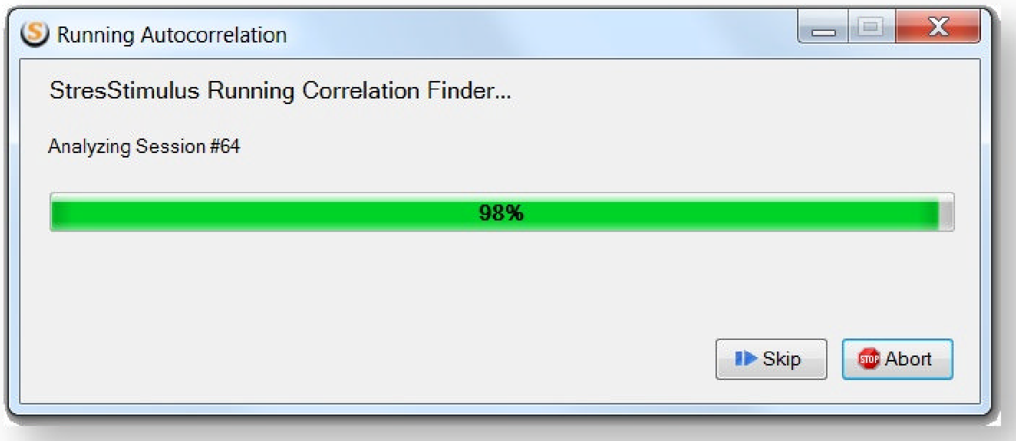
Auto-Correlation Setup

Auto-correlation parameters are necessary to avoid many test errors. To create them, StresStimulus will scan the test case without execution. Any existing and deleted auto-correlation parameters will be re-created.

[✓ Create auto-correlation parameters (recommended).]

Do not create auto-correlation parameters.

To create autocorrelation parameters, StresStimulus will scan the test case without execution. Any existing and deleted autocorrelation parameters will be re-created.



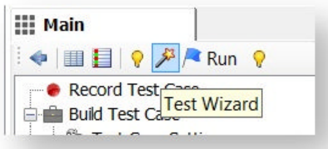
StresStimulus Running Correlation Finder...

Analyzing Session #64

**Note:** Sometimes scanning certain large request can be time-consuming. Click **Skip** to interrupt scanning of the current request. The interruption might not be immediate.

Or click **Abort** to stop the autocorrelation operation. Stopping the scan might not be immediate as well.

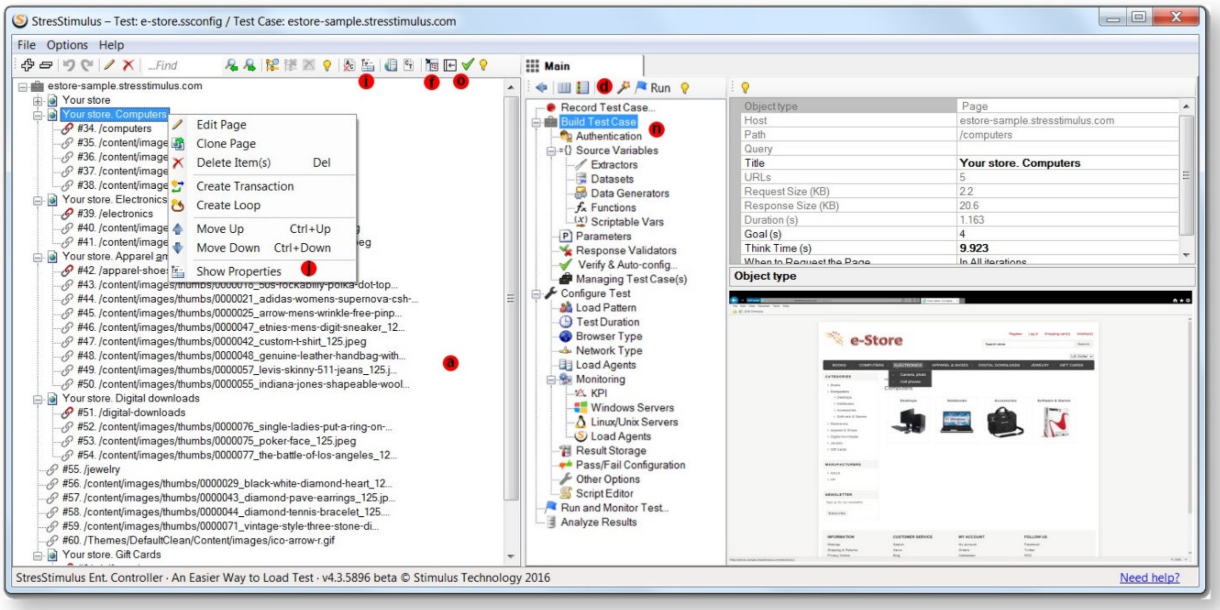
To rerun autocorrelation at anytime, click **Test Wizard** on the Workflow Tree toolbar.



## 4.3.4 Getting Familiar with the Test Case

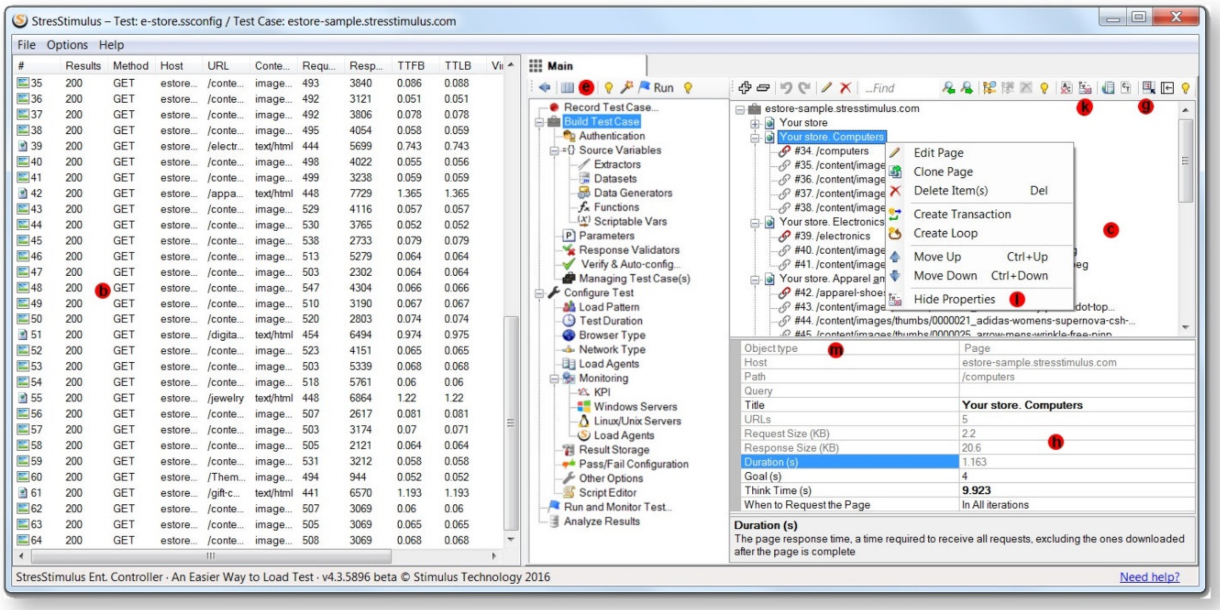
### 4.3.4.1 Test Case Presentation

A test case can be presented  as a **Test Case Tree** (a) or a **Session Grid** (b), as shown below. Test Case Tree can be displayed on the left pane (a) or on the right pane (c)


StresStimulus – Test: e-store.ssconfig / Test Case: estore-sample.stresstimulus.com
File Options Help
...Find
estore-sample.stresstimulus.com
Your store
Your store. Computers
#34. /computers
Your store. Electronics
#39. /electronics
Your store. Apparel and
#42. /apparel-shoes
#43. /content/images/thumbs/0000018_50s-rockabilly-polka-dot-top...
#44. /content/images/thumbs/0000021_adidas-womens-supernova-csh-...
#45. /content/images/thumbs/0000025_arrow-mens-wrinkle-free-pinp...
#46. /content/images/thumbs/0000047_etnies-mens-digit-sneaker_12...
#47. /content/images/thumbs/0000042_custom-t-shirt_125.jpeg
#48. /content/images/thumbs/0000048_genuine-leather-handbag-with...
#49. /content/images/thumbs/0000057_levis-skinny-511-jeans_125.j...
#50. /content/images/thumbs/0000055_indiana-jones-shapeable-wool...
Your store. Digital downloads
#51. /digital-downloads
#52. /content/images/thumbs/0000076_single-ladies-put-a-ring-on-...
#53. /content/images/thumbs/0000075_poker-face_125.jpeg
#54. /content/images/thumbs/0000077_the-battle-of-los-angeles_12...
#55. /jewelry
#56. /content/images/thumbs/0000029_black-white-diamond-heart_12...
#57. /content/images/thumbs/0000043_diamond-pave-earrings_125.jp...
#58. /content/images/thumbs/0000044_diamond-tennis-bracelet_125...
#59. /content/images/thumbs/0000071_vintage-style-three-stone-di...
#60. /Themes/DefaultClean/Content/images/ico-arrow-r.gif
Your store. Gift Cards
Edit Page
Clone Page
Delete Item(s) Del
Create Transaction
Create Loop
Move Up Ctrl+Up
Move Down Ctrl+Down
Show Properties
a
b
c
d
e
f
Main
Run
Record Test Case
Build Test Case
Authentication
Source Variables
Extractors
Datasets
Data Generators
Functions
Scriptable Vars
Parameters
Response Validators
Verify & Auto-config
Managing Test Case(s)
Configure Test
Load Pattern
Test Duration
Browser Type
Network Type
Load Agents
Monitoring
KPI
Windows Servers
Linux/Unix Servers
Load Agents
Result Storage
Pass/Fail Configuration
Other Options
Script Editor
Run and Monitor Test
Analyze Results
e-Store
StresStimulus Ent. Controller · An Easier Way to Load Test · v4.3.5896 beta © Stimulus Technology 2016
Need help?

| | |
|---|---|
| Object type | Page |
| Host | estore-sample.stresstimulus.com |
| Path | /computers |
| Query | |
| Title | **Your store. Computers** |
| URLs | 5 |
| Request Size (KB) | 2.2 |
| Response Size (KB) | 20.6 |
| Duration (s) | 1.163 |
| Goal (s) | 4 |
| Think Time (s) | **9.923** |
| When to Request the Page | In All iterations |

**Object type**



| # | Results | Method | Host | URL | Conte... | Requ... | Resp... | TTFB | TTLB | Vi |
|---|---|---|---|---|---|---|---|---|---|---|
| 35 | 200 | GET | estore... | /conte... | image... | 493 | 3840 | 0.086 | 0.088 | |
| 36 | 200 | GET | estore... | /conte... | image... | 492 | 3121 | 0.051 | 0.051 | |
| 37 | 200 | GET | estore... | /conte... | image... | 492 | 3806 | 0.078 | 0.078 | |
| 38 | 200 | GET | estore... | /conte... | image... | 495 | 4054 | 0.058 | 0.059 | |
| 39 | 200 | GET | estore... | /electr... | text/html | 444 | 5699 | 0.743 | 0.743 | |
| 40 | 200 | GET | estore... | /conte... | image... | 498 | 4022 | 0.055 | 0.056 | |
| 41 | 200 | GET | estore... | /conte... | image... | 499 | 3238 | 0.059 | 0.059 | |
| 42 | 200 | GET | estore... | /appa... | text/html | 448 | 7729 | 1.365 | 1.365 | |
| 43 | 200 | GET | estore... | /conte... | image... | 529 | 4116 | 0.057 | 0.057 | |
| 44 | 200 | GET | estore... | /conte... | image... | 530 | 3765 | 0.052 | 0.052 | |
| 45 | 200 | GET | estore... | /conte... | image... | 538 | 2733 | 0.079 | 0.079 | |
| 46 | 200 | GET | estore... | /conte... | image... | 513 | 5279 | 0.064 | 0.064 | |
| 47 | 200 | GET | estore... | /conte... | image... | 503 | 2302 | 0.064 | 0.064 | |
| 48 | 200 | GET | estore... | /conte... | image... | 547 | 4304 | 0.066 | 0.066 | |
| 49 | 200 | GET | estore... | /conte... | image... | 510 | 3190 | 0.067 | 0.067 | |
| 50 | 200 | GET | estore... | /conte... | image... | 520 | 2803 | 0.074 | 0.074 | |
| 51 | 200 | GET | estore... | /digita... | text/html | 454 | 6494 | 0.974 | 0.975 | |
| 52 | 200 | GET | estore... | /conte... | image... | 523 | 4151 | 0.065 | 0.065 | |
| 53 | 200 | GET | estore... | /conte... | image... | 503 | 5339 | 0.068 | 0.068 | |
| 54 | 200 | GET | estore... | /conte... | image... | 518 | 5761 | 0.06 | 0.06 | |
| 55 | 200 | GET | estore... | /jewelry | text/html | 448 | 6864 | 1.22 | 1.22 | |
| 56 | 200 | GET | estore... | /conte... | image... | 507 | 2617 | 0.081 | 0.081 | |
| 57 | 200 | GET | estore... | /conte... | image... | 503 | 3174 | 0.07 | 0.071 | |
| 58 | 200 | GET | estore... | /conte... | image... | 505 | 2121 | 0.064 | 0.064 | |
| 59 | 200 | GET | estore... | /conte... | image... | 531 | 3212 | 0.058 | 0.058 | |
| 60 | 200 | GET | estore... | /Them... | image... | 494 | 944 | 0.052 | 0.052 | |
| 61 | 200 | GET | estore... | /gift-c... | text/html | 441 | 6570 | 1.193 | 1.193 | |
| 62 | 200 | GET | estore... | /conte... | image... | 507 | 3069 | 0.06 | 0.06 | |
| 63 | 200 | GET | estore... | /conte... | image... | 505 | 3069 | 0.065 | 0.065 | |
| 64 | 200 | GET | estore... | /conte... | image... | 508 | 3069 | 0.068 | 0.068 | |

| | |
|---|---|
| Object type | Page |
| Host | estore-sample.stresstimulus.com |
| Path | /computers |
| Query | |
| Title | **Your store. Computers** |
| URLs | 5 |
| Request Size (KB) | 2.2 |
| Response Size (KB) | 20.6 |
| Duration (s) | 1.163 |
| Goal (s) | 4 |
| Think Time (s) | **9.923** |
| When to Request the Page | In All iterations |

**Duration (s)**
The page response time, a time required to receive all requests, excluding the ones downloaded after the page is complete

These two presentation forms are compared in the table below. The **Test Case Tree** displays a hierarchical view of TOM with multiple object types. However, it only works with one selected object at a time. The **Session Grid** only displays sessions, but allows to sort and select multiple sessions to perform operations quicker.

| Test Case Presentation Form | Test case Tree | Session Grid |
|---|---|---|
| View | Hierarchical | Flat |
| Displayed objects | All objects or Sessions | Sessions only |
| Sorting | No | Yes |
| Invoke operations | Toolbar or Context menu | Context menu |

After the test case is recorded, the Test Case Tree appears in the object pane (a) on the left. The following commands allow to change its presentation and location:

- To switch the Test Case view from the Tree to the Grid, click **Show Recorded Test Case Sessions In The Session Grid** (d) on the **Workflow Tree Toolbar**. This will also dock the Test Case Tree in the functional pane (c) on the right.
- To switch Test Case view back to the Tree, click the **Tree View button** (e).
- To move the tree to the right pane, click **Dock to the right** (f) on the **Toolbar** of the **Test Case Tree.**
- To move the tree to the left pane, click **Dock to the left** (g) on the Toolbar of the Test Case Tree.

- To display a property grid (h) below the tree, click **Show property grid on the bottom** (i) or right-click and select **Show Properties** in the context menu (j)
- To hide the property grid below the tree, click **Hide property grid on the bottom** (k) or right-click and select **Hide Properties** in the context menu (l).
- The property grid (h) displays properties of the selected object. The first property for all objects is **Object Type** (m)
- To display the tree and the property grid side-by-side, dock the tree on the left, then select **Build Test Case** node (n) in the workflow tree.
- To leave more screen real estate to the right panel, click **Collapse to the left** (o) on the toolbar to maximally shrink the left panel.

**Tip:** When selecting the request in the treeview on the right, the corresponding request in the session grid on the left gets highlighted.

**See Also:**

UI Reference -> Session Grid

#### 4.3.4.2 Test Case Object Hierarchy

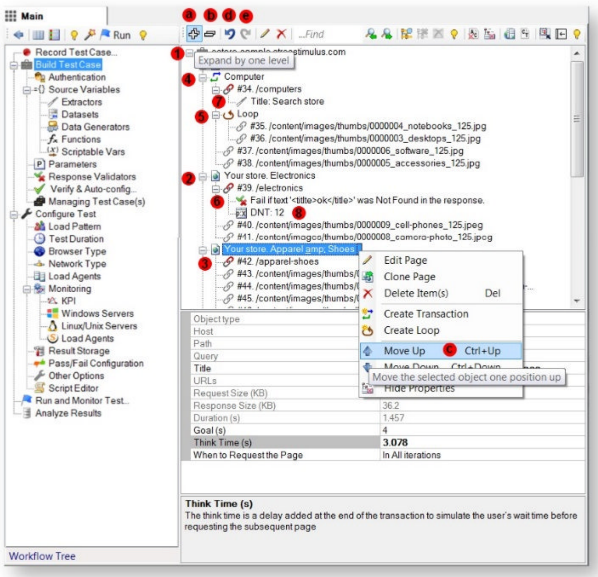
Test Case Tree displays Test Case Objects Hierarchy. The number of supported hierarchical levels is practically unrestricted. The following object types and their relationships are currently supported:

1. Test case can have Pages, Requests, Transactions and Loops.
2. Pages can have Requests Transactions and Loops.
3. Transactions can have Pages, Requests, Transactions, and Loops.
4. Loops can have Pages, Requests, Transactions, and Loops.
5. Requests. Can have Validators, Extractors, and Parameters.
6. Validator. Leaf objects.
7. Extractor. Leaf objects.
8. Parameter. Leaf objects.

Rich object hierarchy allows more precise emulation of the most complex test scenarios and also more granular performance metering. For example, you can create a transaction with any number of pages, loops and other transactions or any parts of a loop, page or transaction. After that, StresStimulus will monitor this transaction and create its performance subreport.

The following commands with objects are supported:

- **You can display as many or as little details on the tree as you need.** Every additional click on the Expand (a) button displays one more hierarchical level. To display the entire tree, click the Expand button several times. The Collapse (b) button works the same way, but in reverse.
- You can re-position objects on the tree. To move loops, transactions and pages on the test case tree, select the object, right-click and select **Move Up** (c) or **Move Down** in the context context menu.
- To move requests, simply select, then drag

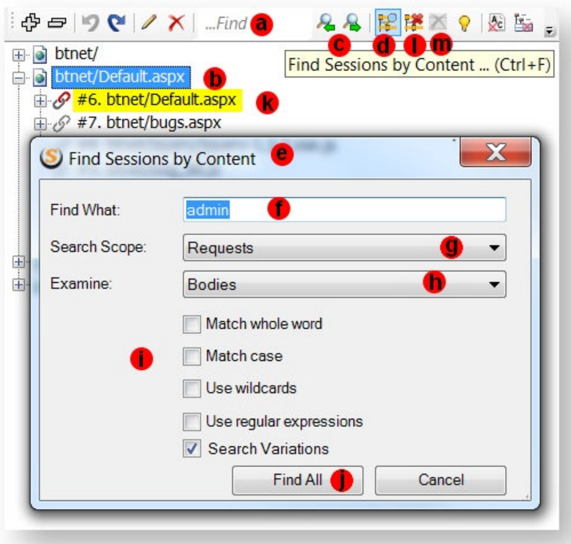


| | |
|---|---|
| Object type | |
| Host | |
| Path | |
| Query | |
| Title | |
| URLs | |
| Request Size (KB) | |
| Response Size (KB) | 36.2 |
| Duration (s) | 1.457 |
| Goal (s) | 4 |
| Think Time (s) | 3.078 |
| When to Request the Page | In All iterations |

**Think Time (s)**
The think time is a delay added at the end of the transaction to simulate the user's wait time before requesting the subsequent page

| | |
|---|---|
| and drop to a new position. From the context menu you can also create or delete new objects.<br>• If you made a mistake in creating, deleting or re-positioning an object, and wish to cancel the change click **Undo** (d) on the toolbar or hit Ctr+Z.<br>• To cancel Undo, click **Redo** (e) or hit Ctr+Y. Multiple sequential Undo and Redo are supported.<br><br>You can change the properties of any selected object in its property grid, displayed below the object tree.<br><br>**Info:** The list of properties of these objects is available in User Interface Reference ->Test Case Tree. | |

### 4.3.4.3 Searching Test Case Tree

| | |
|---|---|
| There are two type of searches available in the test case tree: quick search and deep search. Quick search can find information displayed in the tree itself, such as object name or request URL. Deep search searches the content of the objects.<br><br>**1. Search URL**. In order to do a quick object search, use the **Search URLs** textbox (a). As you type, the first found tree node with the given text will be highlighted (b). To search next or previous, click the **Next** or **Previous** buttons, or F3 or hit Shift+F3 respectively (c)<br><br>**2. Search content.** In order to do an extensive content search of every session's request and response, follow these steps:<br>• Click Find Session by Content or hit Ctrl+F (d). |  |

btnet/
btnet/Default.aspx
#6. btnet/Default.aspx
#7. btnet/bugs.aspx

Find Sessions by Content ... (Ctrl+F)

Find Sessions by Content

Find What: admin
Search Scope: Requests
Examine: Bodies

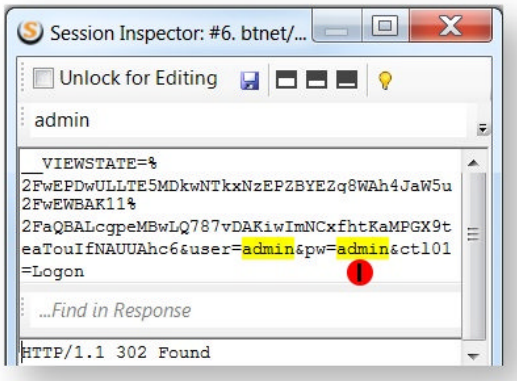
- [ ] Match whole word
- [ ] Match case
- [ ] Use wildcards
- [ ] Use regular expressions
- [x] Search Variations

Find All Cancel

- The search dialog will appear (e). Fill in the following properties:
    - **Find What** (f): The text to search for.
    - **Search Scope** (g): The session parts to search. The options are:
        - **Request** (only)
        - **Response** (only)
        - **Request and Response**
    - **Examine** (h): Within the search scope, what part to examine. The options are:
        - **Headers** (only)
        - **Bodies** (only)
        - **Headers and Bodies**

- You can optionally use the supplemental check boxes (i):
    - **Match whole word:** The search text will be matched as a whole word.
    - **Match case:** The search will be case insensitive.
    - **Use wildcards:** The search text contains wildcards (such as ? to match any one character and * to match any characters)
    - **Use regular expressions:** The search text is a regular expression.
    - **Search Variations.** Search for encoded/decoded variations of the string (selected by default).

- Click **Find All** (j) to begin the search. All the sessions that meet the search criteria will be highlighted in yellow (k).
- Double click on the highlighted session to



admin

```
__VIEWSTATE=%
2FwEPDwULLTE5MDkwNTkxNzEPZBYEZq8WAh4JaW5u
2FwEWBAK11%
2FaQBALcgpeMBwLQ787vDAKiwImNCxfhtKaMPGX9t
eaTouIfNAUUAhc6&user=admin&pw=admin&ctl01
=Logon
```

```
HTTP/1.1 302 Found
```

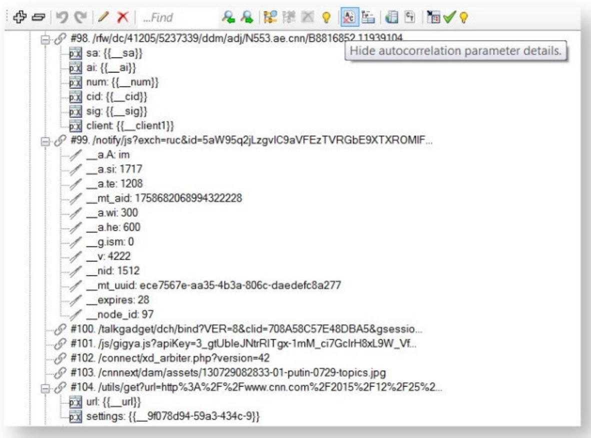
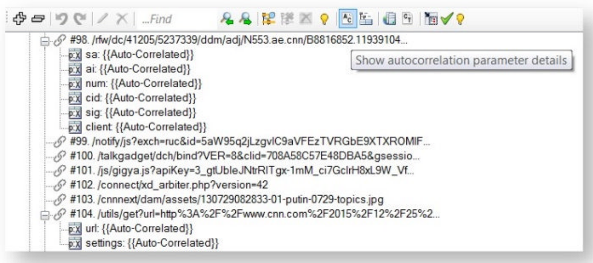
| | |
|---|---|
| see the search criteria highlighted in the Session Inspector (l).<br>• Click the Clear search button (m) to clear the search and un-highlight the sessions. | |
| | |

### 4.3.4.4 Displaying AutoCorrelation Details

Every autocorrelation parameter consists of an autocorrelation extractor (f) and a parameter (g). By default, the extractor is hidden and the parameter is marked as {{Autocorrelated}}

To display the autocorrelation details, click **Show autocorrelation parameter details** on the test case tree toolbar. The extractors will un-hide and their names will be displayed next to the corresponding parameters. To switch back to the default view, click **Hide autocorrelation parameter details.**

> **Note:** Autocorrelated extractors and parameters have names that begin with two underscores.

| | |
|---|---|
|  |  |

```
#98. /rfw/dc/41205/5237339/ddm/adj/N553.ae.cnn/B8816852.11939104
    sa: {{__sa}}
    ai: {{__ai}}
    num: {{__num}}
    cid: {{__cid}}
    sig: {{__sig}}
    client: {{__client1}}
#99. /notify/js?exch=ruc&id=5aW95q2jLzgvIC9aVFEzTVRGbE9XTXROMIF...
    __a.A: im
    __a.si: 1717
    __a.te: 1208
    __mt_aid: 1758682068994322228
    __a.wi: 300
    __a.he: 600
    __g.ism: 0
    __v: 4222
    __nid: 1512
    __mt_uuid: ece7567e-aa35-4b3a-806c-daedefc8a277
    __expires: 28
    __node_id: 97
#100. /talkgadget/dch/bind?VER=8&clid=708A58C57E48DBA5&gsessio...
#101. /js/gigya.js?apiKey=3_gtUbleJNtrRITgx-1mM_ci7GclrH8xL9W_Vf...
#102. /connect/xd_arbiter.php?version=42
#103. /cnnnext/dam/assets/130729082833-01-putin-0729-topics.jpg
#104. /utils/get?url=http%3A%2F%2Fwww.cnn.com%2F2015%2F12%2F25%2...
    url: {{__url}}
    settings: {{__9f078d94-59a3-434c-9}}
```

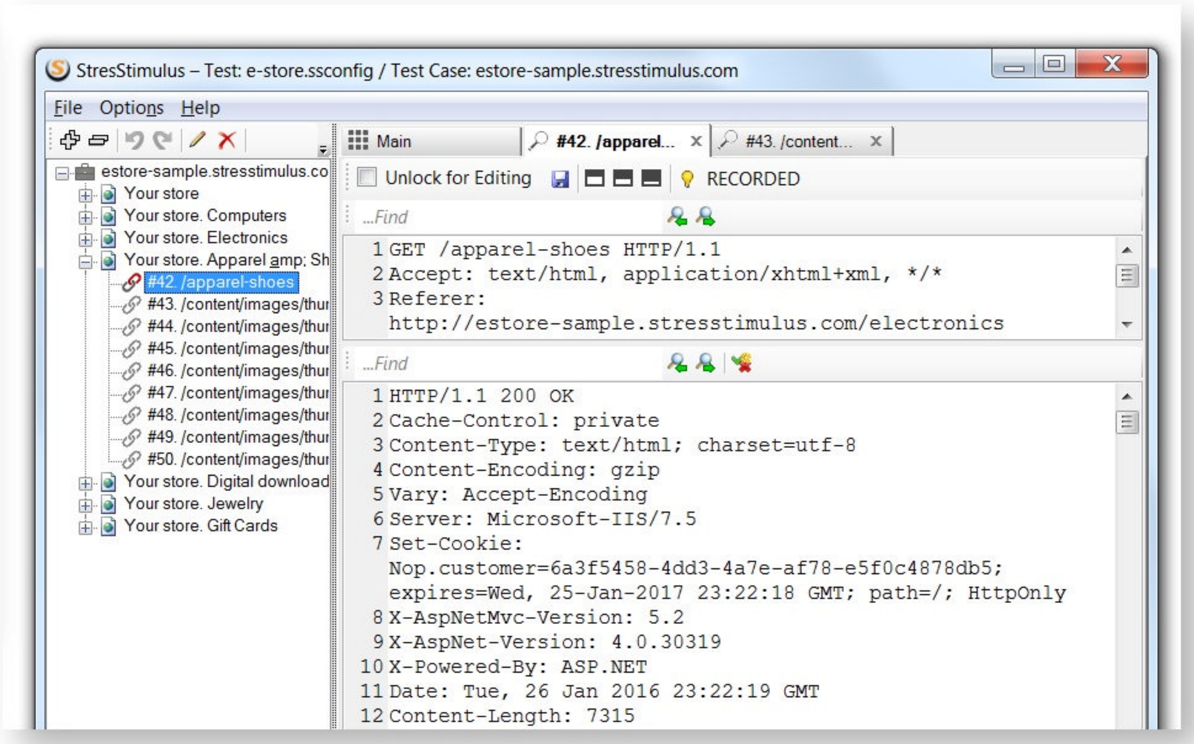
```
#98. /rfw/dc/41205/5237339/ddm/adj/N553.ae.cnn/B8816852.11939104...
    sa: {{Auto-Correlated}}
    ai: {{Auto-Correlated}}
    num: {{Auto-Correlated}}
    cid: {{Auto-Correlated}}
    sig: {{Auto-Correlated}}
    client: {{Auto-Correlated}}
#99. /notify/js?exch=ruc&id=5aW95q2jLzgvIC9aVFEzTVRGbE9XTXROMIF...
#100. /talkgadget/dch/bind?VER=8&clid=708A58C57E48DBA5&gsessio...
#101. /js/gigya.js?apiKey=3_gtUbleJNtrRITgx-1mM_ci7GclrH8xL9W_Vf...
#102. /connect/xd_arbiter.php?version=42
#103. /cnnnext/dam/assets/130729082833-01-putin-0729-topics.jpg
#104. /utils/get?url=http%3A%2F%2Fwww.cnn.com%2F2015%2F12%2F25%2...
    url: {{Auto-Correlated}}
    settings: {{Auto-Correlated}}
```

### 4.3.4.5 Pages and HTTP Sessions

After recording the test case, the Test Case Tree displays the following objects:

- **Pages.** StresStimulus automatically generates page names based on their HTML title Tag. You can change page names during recording in the IE recorder. You can also rename the page by right-clicking the page in the test case tree.

- **HTTP sessions.** Each session consists of a pair of request and response messages.
  - To view the recorded requests and responses or to make small modifications without re-recording the entire test case, use the session inspector. Double-click on the session to display the session inspector. For more details, see User Interface Reference -> Session Inspector.



```
GET /apparel-shoes HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Referer:
 http://estore-sample.stresstimulus.com/electronics
```

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
Content-Encoding: gzip
Vary: Accept-Encoding
Server: Microsoft-IIS/7.5
Set-Cookie:
 Nop.customer=6a3f5458-4dd3-4a7e-af78-e5f0c4878db5;
 expires=Wed, 25-Jan-2017 23:22:18 GMT; path=/; HttpOnly
X-AspNetMvc-Version: 5.2
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Tue, 26 Jan 2016 23:22:19 GMT
Content-Length: 7315
```

You may also need to add, delete or reposition sessions. For more details, see Adding, Deleting and Changing Sessions

- **Autocorrelation rules.** Some of the sessions may have automatically created Autocorrelation rules which should not be changed.

More objects are created in the subsequent configuration steps.

**See also:**

For Page and Session Context Menu Commands, see User Interface Reference -> Toolbar

For Test Case Tree's toolbar commands, see and User Interface Reference -> Toolbar

### 4.3.4.6 Adding, Deleting and Changing Sessions

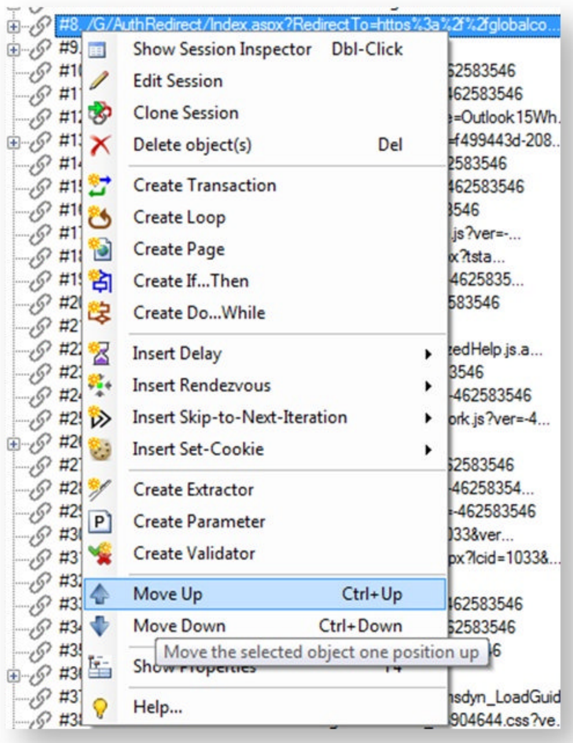
Sometimes after a test case is recorded, you need to change some sessions. This may be necessary because of changes in the web application or a requirement to modify the set of recorded sessions. The following types of session changes are available:**Re-positioning**

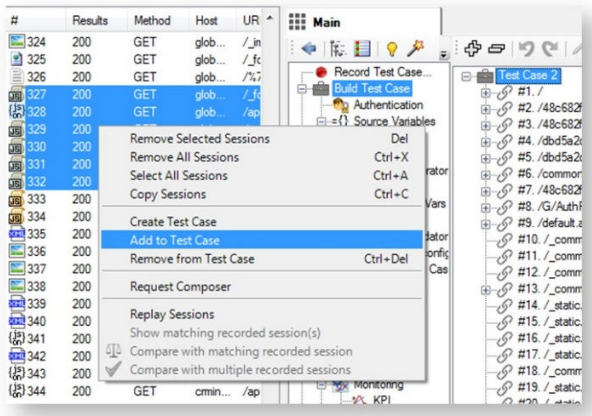
To re-position the selected in the Test Case Tree request or page, drag and drop it into a new position. You can also move them up or down by pressing Ctrl + Up/Down arrow keys.

**Adding**

To add new sessions from the session grid to the Test Case Tree.

1. Select **Build Test Case** node to display the Test Case Tree on the right.
2. Display new sessions in the session grid on the left in one of the following ways:
   - Issue requests from a web browser or other web client and capture them in Fiddler.
   - Open a previously saved .saz file with HTTP sessions.
3. Select the sessions you want to add to the test case then right click > StresStimulus Commands > Add to Test Case in order to add the sessions to the end of the test case.
4. You can re-position the added sessions to the desired positions as described above.

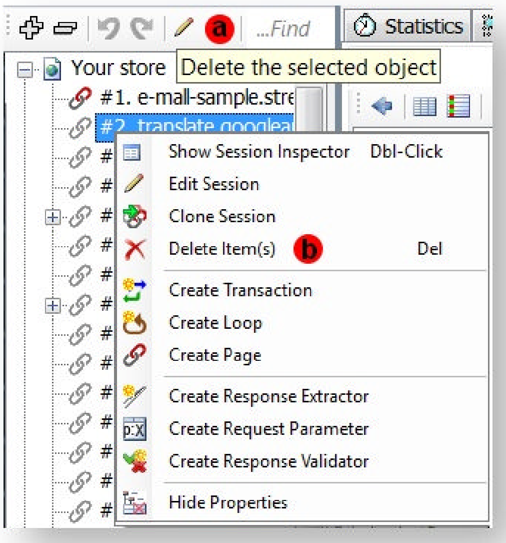



Remove Selected Sessions Del
Remove All Sessions Ctrl+X
Select All Sessions Ctrl+A
Copy Sessions Ctrl+C
Create Test Case
Add to Test Case
Remove from Test Case Ctrl+Del
Request Composer
Replay Sessions
Show matching recorded session(s)
Compare with matching recorded session
Compare with multiple recorded sessions

### Deleting

**From the Test Case Tree:** select a session, you need to delete or select multiple sessions by clicking while holding Ctrl or Shift key. After that click Delete in the toolbar (a), right-click and select click Delete or hit (Del)

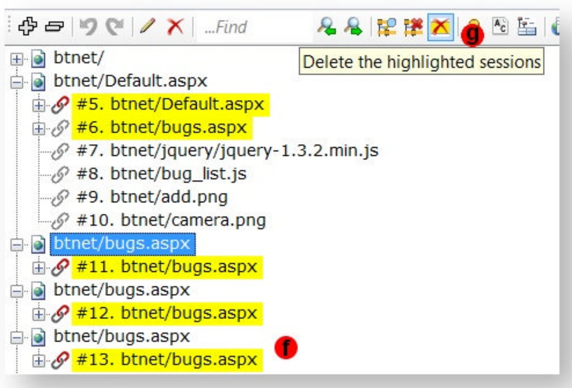
**From the Session Grid:** click **Show recorded test case sessions in the session grid** (c), select sessions to be deleted, right-click and in **StresStimulus Commands** , select **Remove from Test Case** or hit (Ctrl+Del).

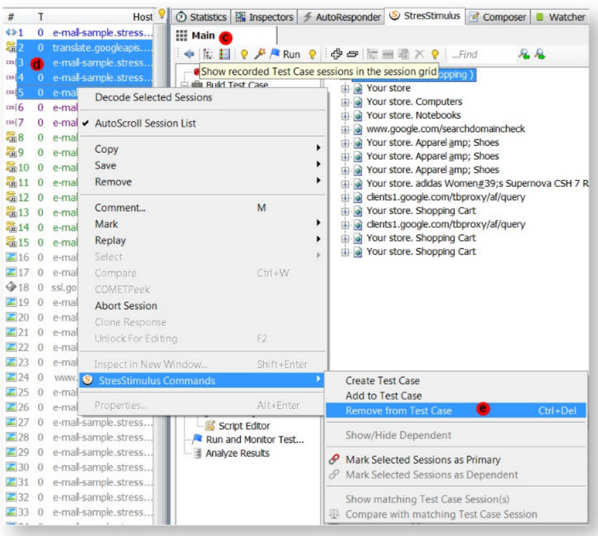


### Searching and Deleting Multiple Sessions

You can delete multiple sessions matching certain criteria. This is a two-step process:

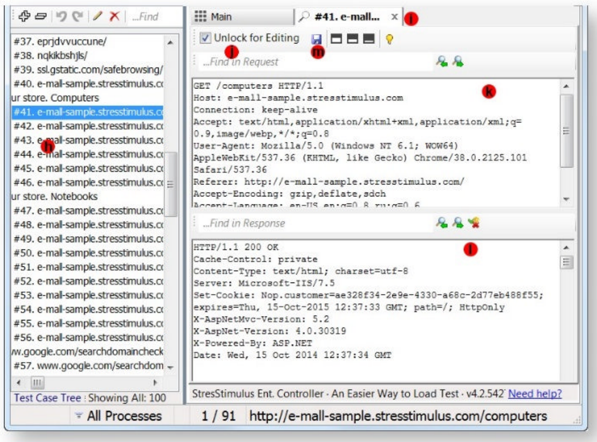
1. Search for the session matching the criteria, as described in Searching Test Case Tree. All session matching the criteria will be highlighted (f)
2. Click **Delete the found highlighted sessions** (g) on the toolbar.





**Editing**

To edit a session in the Test Case Tree, double-click the session (h), and in the appeared in a new Tab (i) session inspector, check **Unlock for editing** (j). After that you can make changes in the session request (k) and response (l). When finished, click **Save** (m).


...Find
#37. eprjdvvuccune/
#38. nqkikbshjjs/
#39. ssl.gstatic.com/safebrowsing/
#40. e-mall-sample.stresstimulus.c
ur store. Computers
#41. e-mall-sample.stresstimulus.c
h
ur store. Notebooks
www.google.com/searchdomaincheck
Test Case Tree : Showing All: 100
All Processes
Main
#41. e-mall...
i
Unlock for Editing
j
m
...Find in Request
k
...Find in Response
l
StresStimulus Ent. Controller · An Easier Way to Load Test · v4.2.542 Need help?
1 / 91
http://e-mall-sample.stresstimulus.com/computers

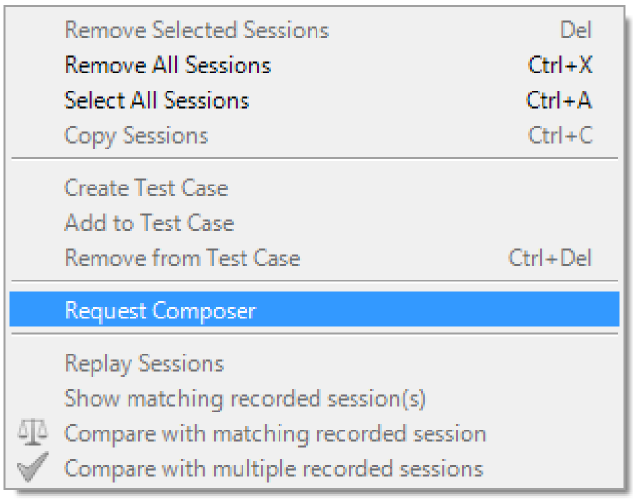
```
GET /computers HTTP/1.1
Host: e-mall-sample.stresstimulus.com
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=
0.9,image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/38.0.2125.101
Safari/537.36
Referer: http://e-mall-sample.stresstimulus.com/
Accept-Encoding: gzip,deflate,sdch
```

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
Server: Microsoft-IIS/7.5
Set-Cookie: Nop.customer=ae328f34-2e9e-4330-a68c-2d77eb488f55;
expires=Thu, 15-Oct-2015 12:37:33 GMT; path=/; HttpOnly
X-AspNetMvc-Version: 5.2
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Wed, 15 Oct 2014 12:37:34 GMT
```

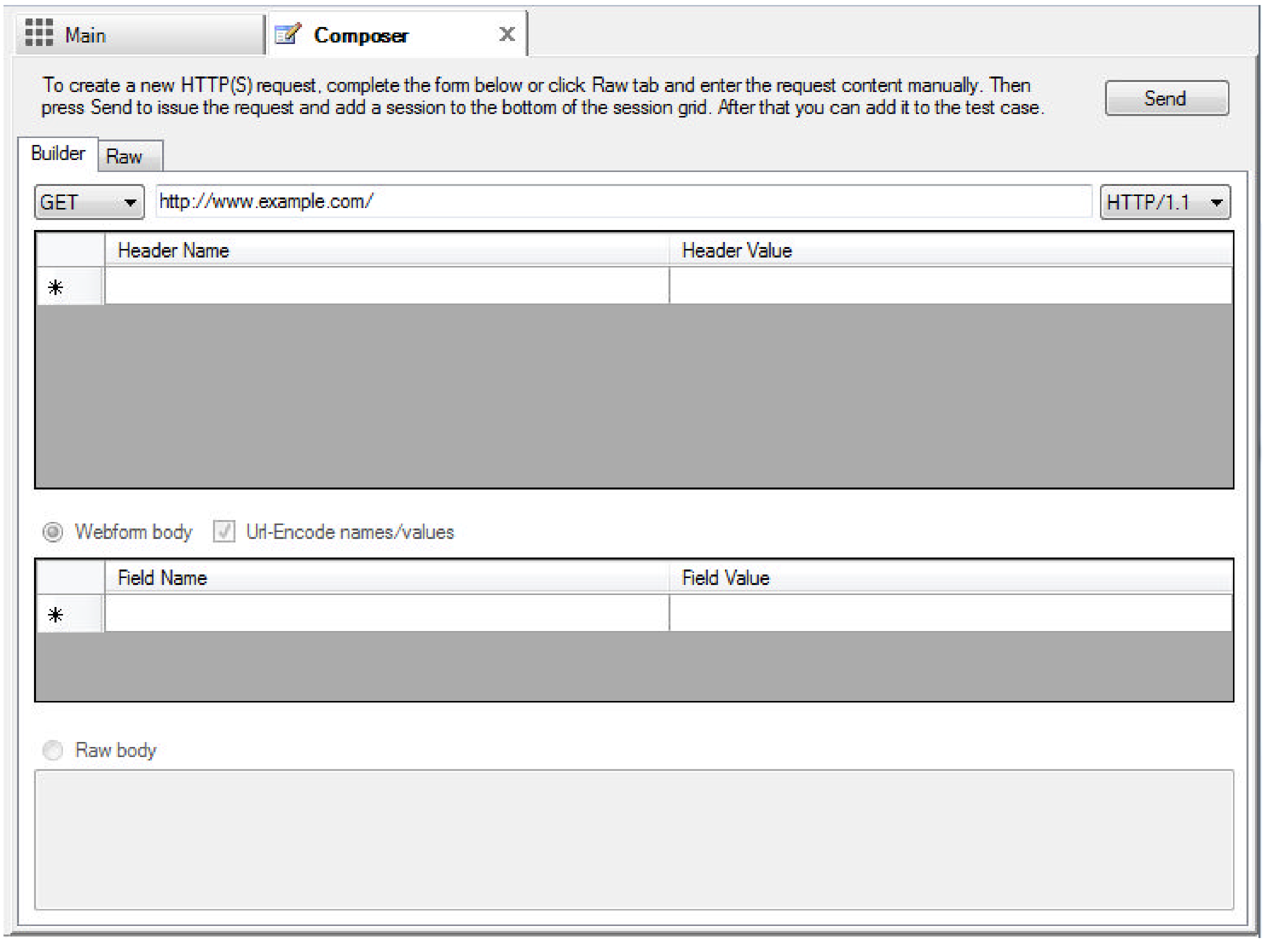
### 4.3.4.7 Creating Sessions Using Request Composer

Most test cases are created as a result of recording a test case. However, in some instances, one, several requests, or entire test cases can't be recorded and must be created manually. This is especially helpful for performance testing of a restful API. For these cases you can use the Request Composer.

To open the Request Composer, right click on the Sessions Grid and select the Request Composer option.


Remove Selected Sessions Del
Remove All Sessions Ctrl+X
Select All Sessions Ctrl+A
Copy Sessions Ctrl+C
Create Test Case
Add to Test Case
Remove from Test Case Ctrl+Del
Request Composer
Replay Sessions
Show matching recorded session(s)
Compare with matching recorded session
Compare with multiple recorded sessions

The Request Composer should open in a new tab.



To create a new HTTP(S) request, complete the form below or click Raw tab and enter the request content manually. Then press Send to issue the request and add a session to the bottom of the session grid. After that you can add it to the test case.

http://www.example.com/

To create a new HTTP(S) request, you can either complete the builder form or click the Raw tab and enter the request content manually.

Then press Send to issue the request and add a session to the bottom of the session grid. After that you can add it to the test case as described in **Adding** section here.

### Builder Options

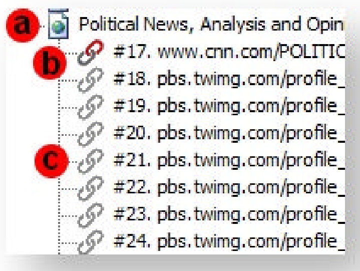
1. Select the request method.
2. Type the absolute request url including the query string.
3. Select the HTTP Version number (1.1 is the default).
4. If you want to add request additional request headers like User-Agent or Accept. Do not add the Content-Length header as it will be added automatically.
5. If you selected the POST or PUT method then you can add a request body.
    a. If you have a webform body then you can add name/value names.
    b. Check the Url-Encode names/values checkbox to url-encode the names and values and adds the Content-Type: application/x-www-form-urlencoded header.
    c. If you have any other body then select the Raw body radio button and type the request body.

**Note**

Request Builder is only available in the Standalone version of StresStimulus. If you are using StresStimulus from Fiddler-Addon, use Fiddler's Composer feature to create requests.

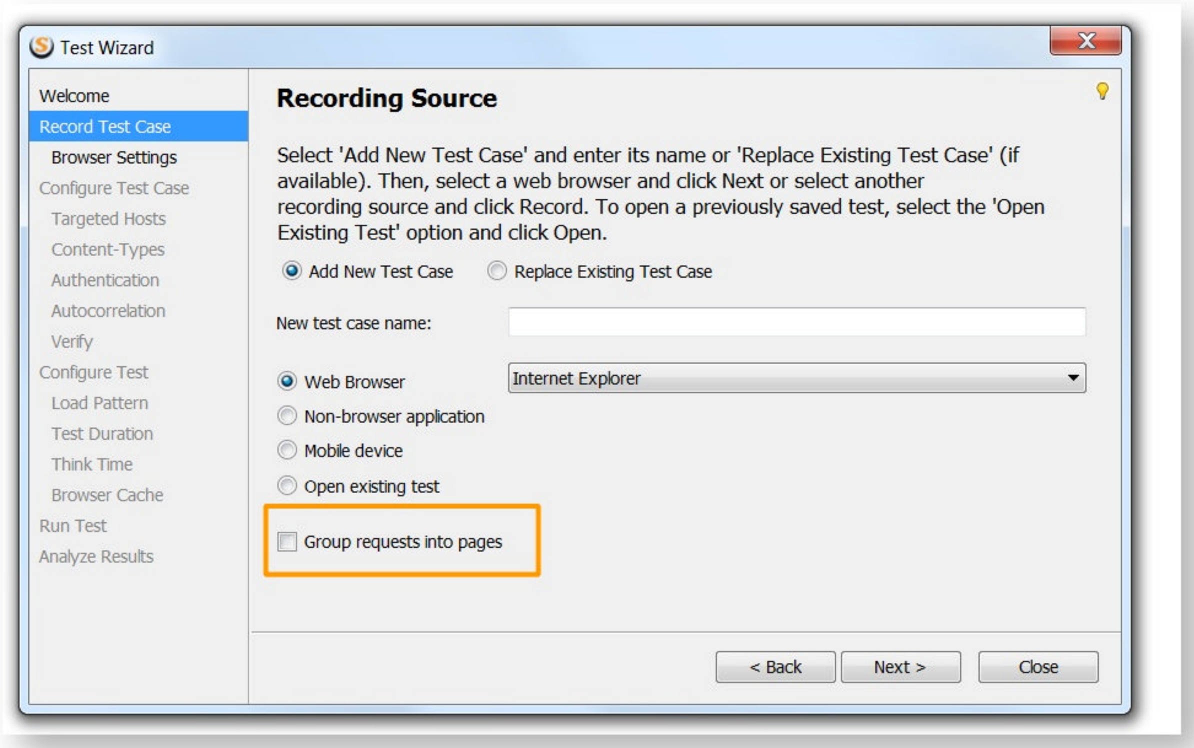
## 4.3.5 Page Structure

When you start recording a test case from a web browser or mobile device, StresStimulus automatically groups the recorded requests into pages (a) using a heuristic algorithm. Pages are essential elements of the test, and StresStimulus collects and reports page performance metrics in addition to request performance metrics. In some cases, requests may not be associated with any page. These requests will be left ungrouped.


a
Political News, Analysis and Opin
b
#17. www.cnn.com/POLITIC
#18. pbs.twimg.com/profile_
#19. pbs.twimg.com/profile_
#20. pbs.twimg.com/profile_
c
#21. pbs.twimg.com/profile_
#22. pbs.twimg.com/profile_
#23. pbs.twimg.com/profile_
#24. pbs.twimg.com/profile_

**Note:** Requests issued after the web page is already displayed do not affect the page download time. Such requests are sent with significant delay after the response and are not assigned to any page. Examples of unassigned requests are AJAX requests triggered by a mouse-over action or a request trigger by a client's JavaScript (i.e.JQuery timer event). StresStimulus uses a heuristic page breakdown algorithm to determine which requests are part of the page and which are unassigned requests.

You can disable automatic page breakdown. To do so, in the **Test Wizard** -> **Record Test Case** step, un-checked the **Group requests into pages** box. This can be helpful when, instead of tracking the performance of pages, you prefer to track performance of transactions which can be used to group requests manually. Disabling pages is also helpful when testing Web services.



Recording Source

Select 'Add New Test Case' and enter its name or 'Replace Existing Test Case' (if available). Then, select a web browser and click Next or select another recording source and click Record. To open a previously saved test, select the 'Open Existing Test' option and click Open.

[✓ Add New Test Case] Replace Existing Test Case

New test case name:

[✓ Web Browser] Internet Explorer
Non-browser application
Mobile device
Open existing test

☐ Group requests into pages

### 4.3.5.1 Request Concurrency

Modern browsers issue requests in parallel to download webpages faster. StresStimulus simulates concurrent requests by mimicking the browser's behavior based on the following test factors:

- Page breakdown information.
- Types of simulated browsers in the mix.
- Types of simulated network bandwidth in the mix.
- Server responses.

It does not use the information about parallel connections collected during the recording because this information is not fully relevant to the emulated test conditions. Emulated browser types can be different from the browser type used for recording. Also, the server load during test run is typically higher than the one during the recording.

The following behavior is emulated for every page and for every VU independently:

1. Issue primary request and wait for the response. During this time all dependent requests are blocked.
2. Receive the primary response and handle dependent requests, which can be issued in parallel or sequentially.

1.

a. **Parallel dependent requests.** Dependent requests, representing webpage resources, such as images or style sheets, are typically sent in parallel, using multiple connections. The number of connections depends on the maximum number of available concurrent connections supported by the emulated browser. Each browser type has two limits: Max connections per hosts and Max connections per proxy. Both limits are enforced, so the number of connections per any single hosts and the number of total connections are independently limited. All other dependent requests, if any, are blocked (queued). As responses are being received, some of the queued dependent requests are being issued without violating the browser's connection limit constrain. Additional blocking rules are imposed to prevent issuing subsequent dependent requests until certain critical responses are received. These rules follow standard browsers specifications.

1.

a. **Sequential dependent requests.** Some dependent requests on a page must be issued sequentially, according to the page logic. For example, a video player on the page request subsequent video fragment after receiving and processing previous fragments. Some MIME types (e.g. Text/HTML) are always requested sequentially. Depending on the tested application, you can add additional MIME types whose requests must be issued only after receiving all previous responses. This will prevent dependent requests of these types from being requested in parallel with other dependent requests on a page. To do so, in Configure Test --> Other Options, in the property **MIME Types requested sequentially,** click the drop-down and enter additional MIME types. Separate multiple entries by ",". For example, enter "image,video", as shown below, to request all images and videos sequentially; or enter "video/mp4" to Request MP4 video sequentially. After modifying this property, launch test wizard and rerun autocorellation.

**Info:** Sent and received traffic cannot exceed emulated network type bandwidth limits. To achieve this the traffic can be appropriately slowed down.

3. After the last dependent response was received, the current page is finished processing.
4. Optional think time delay is injected.
5. The VU is ready for processing the next page.

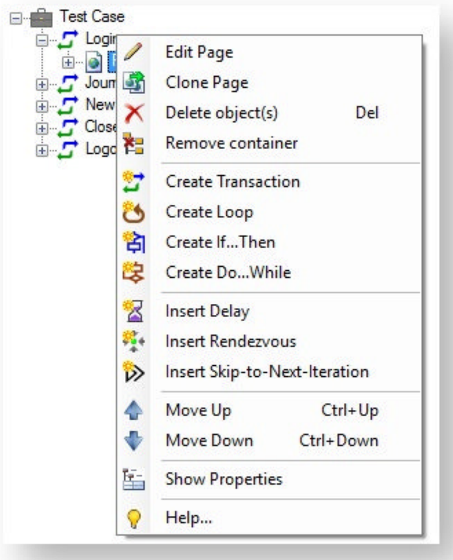
You can change the default request concurrency by changing the page breakdown, as described in the next section.

### 4.3.5.2 Managing Page Breakdown

The initial page breakdown can be changed manually. This can be necessary for a more accurate grouping of requests related to a webpage.

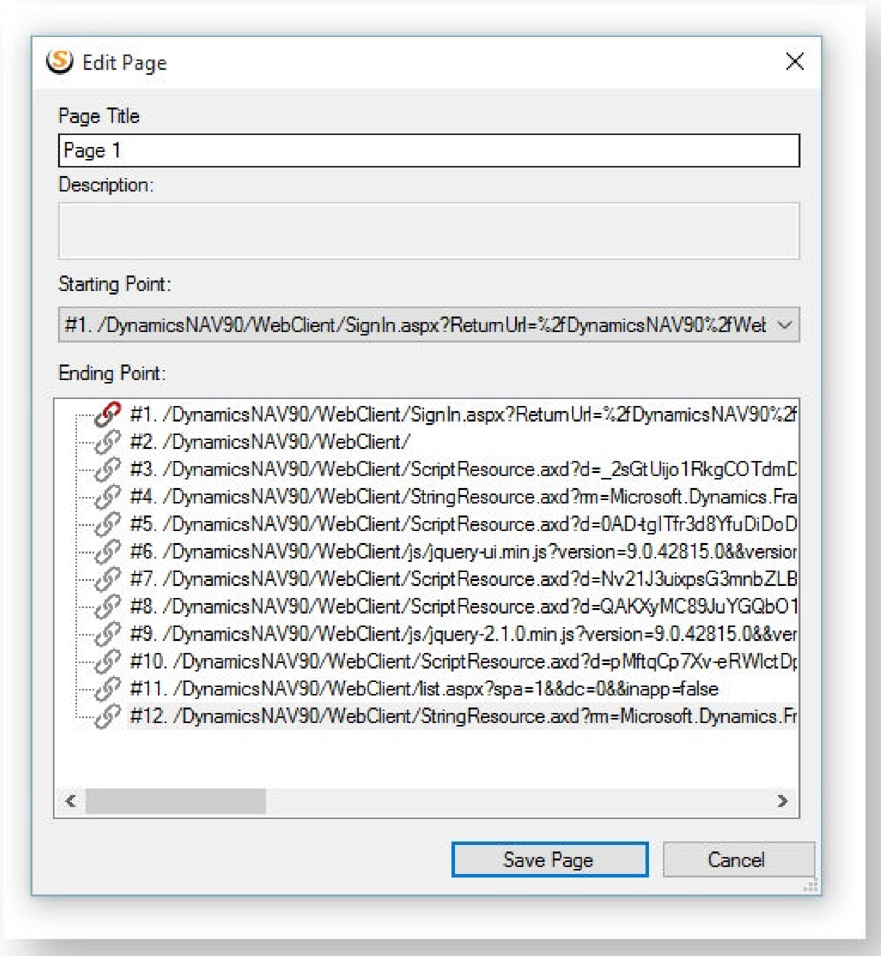
#### Editing an Existing Page

To edit a page after recording, open its context menu by right mouse clicking on it. Then select **Edit Page.**



A pop-up will appear which will allow you to edit the page title, the request with which the page begins loading, and the request with which the page stops loading.

After you finish saving all changes, click **Save.**



Edit Page

Page Title

Page 1

Description:

Starting Point:

#1. /DynamicsNAV90/WebClient/SignIn.aspx?ReturnUrl=%2fDynamicsNAV90%2fWet

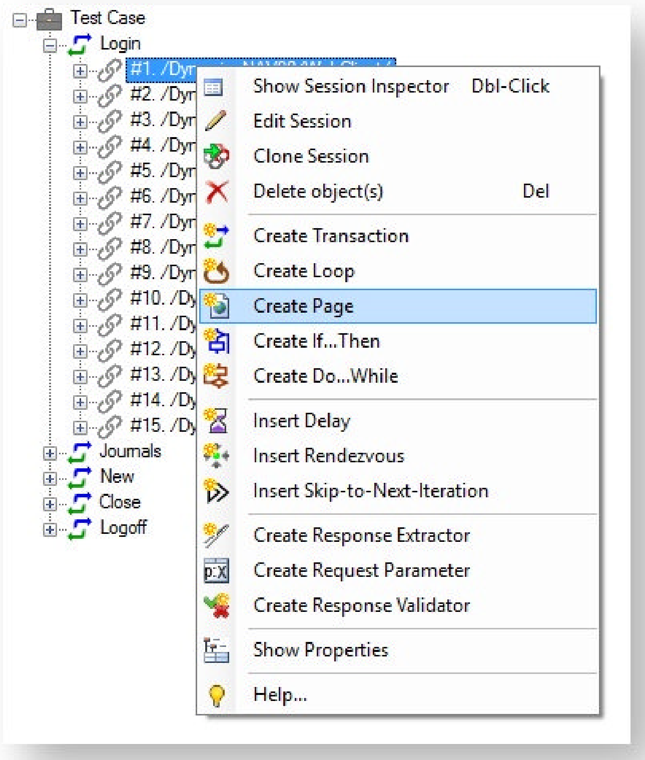
Ending Point:

- #1. /DynamicsNAV90/WebClient/SignIn.aspx?ReturnUrl=%2fDynamicsNAV90%2f
- #2. /DynamicsNAV90/WebClient/
- #3. /DynamicsNAV90/WebClient/ScriptResource.axd?d=_2sGtUijo1RkgCOTdmD
- #4. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dynamics.Fra
- #5. /DynamicsNAV90/WebClient/ScriptResource.axd?d=0AD-tgITfr3d8YfuDiDoD
- #6. /DynamicsNAV90/WebClient/js/jquery-ui.min.js?version=9.0.42815.0&&versior
- #7. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3mnbZLB
- #8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQbO1
- #9. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0.42815.0&&ver
- #10. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eRWlctD
- #11. /DynamicsNAV90/WebClient/list.aspx?spa=1&&dc=0&&inapp=false
- #12. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dynamics.Fr

## Creating a New Page

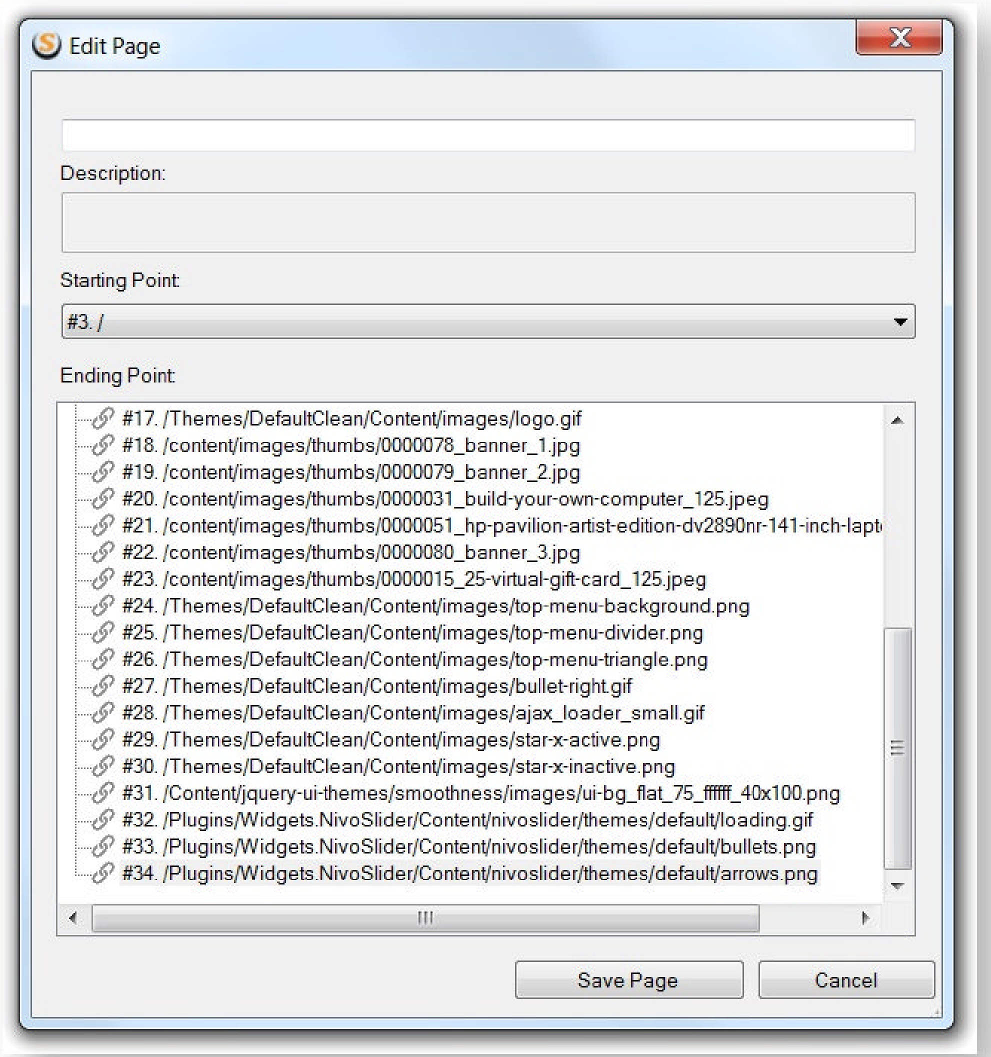
If your test case has requests unassigned to any pages, you can group them in a new page.

To create a new page, right mouse click on a request to open its context menu.



Then specify a page title, the request with which the page begins loading, and the request with which the page stops loading.

After you finish saving all changes, click **Save.**



Description:

Starting Point:
#3. /

Ending Point:
#17. /Themes/DefaultClean/Content/images/logo.gif
#18. /content/images/thumbs/0000078_banner_1.jpg
#19. /content/images/thumbs/0000079_banner_2.jpg
#20. /content/images/thumbs/0000031_build-your-own-computer_125.jpeg
#21. /content/images/thumbs/0000051_hp-pavilion-artist-edition-dv2890nr-141-inch-lapt
#22. /content/images/thumbs/0000080_banner_3.jpg
#23. /content/images/thumbs/0000015_25-virtual-gift-card_125.jpeg
#24. /Themes/DefaultClean/Content/images/top-menu-background.png
#25. /Themes/DefaultClean/Content/images/top-menu-divider.png
#26. /Themes/DefaultClean/Content/images/top-menu-triangle.png
#27. /Themes/DefaultClean/Content/images/bullet-right.gif
#28. /Themes/DefaultClean/Content/images/ajax_loader_small.gif
#29. /Themes/DefaultClean/Content/images/star-x-active.png
#30. /Themes/DefaultClean/Content/images/star-x-inactive.png
#31. /Content/jquery-ui-themes/smoothness/images/ui-bg_flat_75_ffffff_40x100.png
#32. /Plugins/Widgets.NivoSlider/Content/nivoslider/themes/default/loading.gif
#33. /Plugins/Widgets.NivoSlider/Content/nivoslider/themes/default/bullets.png
#34. /Plugins/Widgets.NivoSlider/Content/nivoslider/themes/default/arrows.png

## 4.4 Parameterizing dynamic tests

When recorded requests are replayed during a test run, some of them need to be modified before issuing. The process of replacing fixed recorded values in the requests with dynamic values is called parameterization. Most modern websites, except those that merely service static pages and images, are dynamic. Parameterization is an important part of performance testing such websites. The dynamic values can be derived from the following sources:

1. **Calculated on-the-fly based on intrinsic rules built-in a load testing too**l.

Example: cookie correlation and autocorrelation (see Cookie Correlation and Autocorrelation). This type of parameterization, called correlation, works automatically. Its effectiveness depends on the quality of the load testing tool. Some tools can miss necessary correlation rules, and as a result a test will generate errors or more work is required to manually create missing parameters as noted in the next item. StresStimulus supports autocorrelation in all major web platforms.

2. **Calculated on-the-fly based on rules created by a test analyst**.

Example: extracting server-based data from responses (see the paragraph Parameterization with Server Data). Discovering such rules can be an elaborate process that depends on the quality of the load testing tool. StresStimulus has various features to simplify manual parameterization, such as Parameter Finder and Auto-Configurator.

3. **Fetched from external datasets or generated in real-time**.

Example: parameterization with data sets and data generators (see the paragraph Parameterization with User Data). StresStimulus has data generators for building all major data types.

**Evidence that some parameters are missing**

- Test Case Verify command returns errors or warnings.
- After the test run is complete, the error detail report is not empty.
- It is expected that recorded user’s actions should modify backed data (e.g. database information, log file entries, file updates, communication messages) but after the test run such changes did not take place.

### 4.4.1 Types of Dynamic Requests

#### 4.4.1.1 Cookie Correlation

**Example**. A public website uses a session cookie to maintain state integrity for every visitor. The server issues a unique cookie to every user on the first visit. A cookie is stored in the user's browser and is included in requests on subsequent visits. If cookie handling is disabled in the browser, the website might not work properly.

**Load testing requirements**. To avoid request failures, a recorded cookie must be replaced by a unique value, generated by the server for every VU. Cookie persistence should be provided for the duration of the test for every VU. StresStimulus handles cookie correlation automatically, regardless of the web server platform to ensure that every VU adheres to its client session. For example, it automatically handles the JSESSIONID cookie in Java applications and ASP. NET_SessionId cookie in ASP.net applications.

#### 4.4.1.2 Autocorrelation

**Example**. A stateless web server sends the client a dynamic value representing the application state. This value is included in the response body as a hidden web form field. On the subsequent request, the server expects to receive the same value to maintain application integrity from this client. The application state can change from request to request.

**Load testing requirements.** During load test execution, reissuing requests with recorded application states can result in server errors. To avoid these errors, every request from a virtual user must include the dynamic application state that the server provided. To achieve that, the dynamic value must be identified, extracted from the responses and then used to replace the recorded value in the subsequent requests. This process is called correlation. StresStimulus handles correlation automatically for all web application platforms. For example, it automatically correlates __VIEWSTATE, and __REQUESTDIGEST hidden web form fields for ASP.NET and SharePoint applications, respectively. Autocorrelation supports various types of web sessions such as HTML and AJAX pages. During test execution, autocorrelation rules automatically modify requests to make VUs compatible with dynamic web applications.

**Note:** Internally, each autocorrelation rule is implemented as a hidden unnamed response extractor and a parameter that uses the extractor to substitute a recorded value in the subsequent request. The extractor is hidden, but the parameter appears on the Test Case Tree. Though it is not recommended, you can delete autocorrelation rules.

#### 4.4.1.3 Parameterization with Server Data

**Example 1**. A secure Web application requires users to login. The web server creates an authenticated session for every user, generates a unique token, and sends it back to the browser in the response body. On every subsequent request, the browser sends the token to identify the user and its session.

**Load testing requirements.** Reissuing requests with the recorded session ID during the load test execution will result in server errors. To avoid these errors, every request from a virtual user must include a dynamic token that this virtual user received from the server. To achieve that, the dynamic tokens must be identified, extracted from the responses and used to replace the recorded values in the subsequent requests.

**Example 2.** A web application maintains application-specific dynamic values that are generated on the server and are included in some of the requests. Parameterizing such requests is more difficult than in the previous example.

Several factors listed below complicate finding the dynamic values and the dynamic requests:

- A field name in the response and request are different.
- HTTP messages are not in the name/ value format and the dynamic value is "buried" inside JSON, XML, and proprietary or even binary data stream.

- A value is received in one part of the response (e.g. the header) and submitted in another part of the request (e.g. the query string).
- A value is received from the server, stored in the browser, and after a substantial delay during which many unrelated HTTP roundtrips are performed, is submitted back to the server.

**Load testing requirements.** The process of finding and matching dynamic fields in the recorded requests to responses and creating parameterization rules by injecting proper dynamic values in the proper request, is called parameterization. StresStimulus supports manual and automatic parameterization:

- **Manual parameterization:** the Extractors and Parameters are created manually.

- **Automatic parameterization.** Three tools are used to automatically create Extractors and Parameters: Parameter Finder, Parameter Creator and Auto-Configurator.

#### 4.4.1.4 Parameterization with User Data

**Example**. A test scenario includes a search page which submits a user's search text. When running a load test, the server receives the same recorded search text for all VUs. Due to server caching, the server's response will be substantially accelerated because a search result will be retrieved from memory without accessing the database. As a result, performance testing will be inaccurate.

**Load testing requirements.** For realistic performance testing, multiple sets of user data should be used during the replay to properly emulate multiuser data entry. Such dynamic sets of data can be either predefined or dynamically created on-the-fly.

- To maintain predefined business data StresStimulus uses Datasets. Any external tabular data stored in .csv format can be added to a test. A dataset field can be bound to the appropriate request parameter.
- To generate data in real-time during test execution, StresStimulus uses Data Generators and Functions. All major data types can be generated.

StresStimulus supports all listed types of dynamic requests and provides increased performance testing accuracy as well as saves time on test configuration.

## 4.4.2 Manual Parameterization

This section describes manual test parameterization and the following section describes automatic test parameterization. It is recommended to start test configuration with automatic parameterization, as it likely will configure all or most of the required parameters while saving time. Use the manual parameterization only if the automatic parameterization missed some necessary parameters or errors are displayed while verifying the test. The manual parameterization is described first in this document for clarity.

Manual parameterization includes two main steps:

1. Creating one or several variables that are sources of dynamic data.

2. Create a parameter that uses the variables to substitute recorded value with dynamic data.

### 4.4.2.1 Variables

Variables are used to reference various types of dynamic data for the purpose of parameterizing simulated requests. Each variable is evaluated during the test run before issuing a request where the variable is used for parameterization.

StresStimulus supports four types of variables provided below:

| Variable Type | Data Source | Use in Parameters |
|---|---|---|
| Extractor | Server-Based | {{extractor_name}} |
| Dataset | Predefined | {{dataset.field$databinding_method}} |
| Data Generator | Generated on-the-fly | {{data_generator_name}} |
| Function | Generated on-the-fly | {{function_name}} |
| External | Provided in your custom code | {{variable_name}} |

Each variable is stored in the test script as a software object of a specific type that has appropriate set of properties. When you create a variable in StresStimulus UI, an appropriate property grid will be presented. After the variable is created, almost all of its property can be changed through the same property grid. You can also edit variables' properties using the script editor.

When creating a variable, you do not need to worry about specifying its data type. When a variable is populated with external data, StresStimulus automatically converts external data into the text readable format if necessary, using the appropriate decoding method. For example, if a variable is populated with data extracted from a WCF binary server response, then binary data will be automatically converted to XML before populating the variable.

#### Extractors

StresStimulus uses extractors to take dynamic values from test run server responses, and re-insert them into subsequent requests using parameters. An Extractor is a variable that derives the dynamic values from a server response. It is also a rule of retrieving a value by parsing a response.

Extractors support various response encoding, including:

- Text based formats, such as HTML forms, XML, JSON,
- Binary formats, such as WCF or AMF

The process of creating an extractor includes the following steps:

1. Determine a dynamic value that should be extracted from a particular response. This value will be used to form a parameter that will replace the recorded value in one of the subsequent requests.

2. Determine a rule that extracts the value of the response.

3. Depending on the rule, select a type of the extractor to use. The five types of supported extractors are provided in the table below. Some of them are available only for specific types of responses.

| **Extractor Type** | **Description** | **Restrictions** |
|---|---|---|
| Text Delimited | Extracts a value that occurs in the response between two specified strings: Starting string (Text before) and ending strings (Text after). | |
| Regular Expression | Extracts a value that is found in the response using a regular expression search. | |
| Header | Extracts a value of a specific response header. | |
| Form Field | Extracts a value of a specific web form field. | Available only in web forms |
| XPath Query | Extracts a value that is found in an XML response body by an XPath query. This includes WCF binary responses. | Available only in XML responses |
| JPath Query | Extracts a value that is found in an JSON response body by an JPath query. | Available only in JSON responses |

1. Test that the Extractor returns the expected value from the recorded response.

2. Make sure that this rule is generic enough and works on other instances of the same response generated by a different virtual user on a different iteration.

3. Give the Extractor a meaningful name and save it.

## How to Create an Extractor

### Simple Extractor

A simple extractor selects a string or a value from a response. To create it:

1. Select the request or an existing extractor in the test case tree,

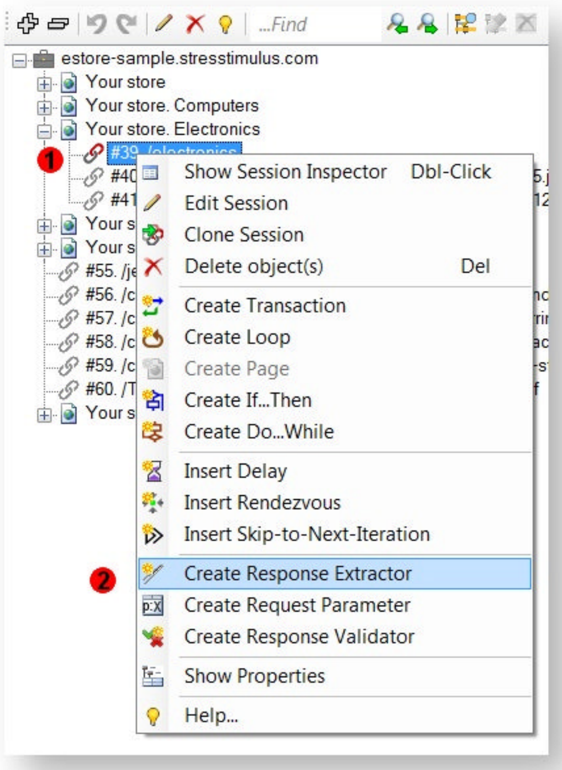
2. Right click and choose Create Response Extractor.

3. The extractor control will appear. It displays four areas:

1.

a. Extractor type selector
b. Property grid
c. Toolbar
d. Response window, displaying response content

**Note:** Most often, response content is text based, and it is displayed in the response window as a clear text. Compressed and/or encrypted text based responses are automatically decompressed /decrypted and displayed as clear text as well. WCF binary responses are decoded and displayed as XML.



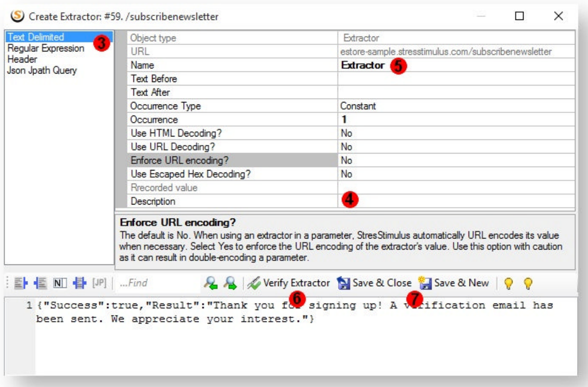
3. Select the extractor type.

5. Complete an optional description and fill out other required properties defining a text search rule that depends on the extractor type.

4. Give it a unique and meaningful name. It will appear in the Variable Picker when you create parameters.

6. Verify the Extractor.

7. Click **Save & Close**. To create more extractors, click **Save & New**.

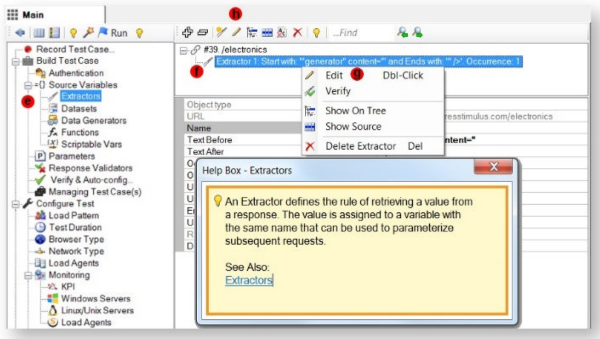


| | |
|---|---|
| Object type | Extractor |
| URL | estore-sample.stresstimulus.com/subscribenewsletter |
| Name | **Extractor** |
| Text Before | |
| Text After | |
| Occurrence Type | Constant |
| Occurrence | 1 |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |
| Enforce URL encoding? | No |
| Use Escaped Hex Decoding? | No |
| Rrecorded value | |
| Description | |

**Enforce URL encoding?**
The default is No. When using an extractor in a parameter, StresStimulus automatically URL encodes its value when necessary. Select Yes to enforce the URL encoding of the extractor's value. Use this option with caution as it can result in double-encoding a parameter.

```
{"Success":true,"Result":"Thank you for signing up! A verification email has
been sent. We appreciate your interest."}
```

Once the extractor is created, you can edit it. To do so, select the extractor node on the Workflow Tree (e), right-click the extractor you wish to edit (f) and then in the context menu select **Edit** (g) or double-click on the extractor. You can also click **edit** (h) on the toolbar instead. The same extractor control will appear. Change the extractor's properties, verify it and save changes.



Extractor 1: Start with: '"generator" content="' and Ends with: '" />'. Occurrence: 1

An Extractor defines the rule of retrieving a value from a response. The value is assigned to a variable with the same name that can be used to parameterize subsequent requests.

## Extractor from Extractor

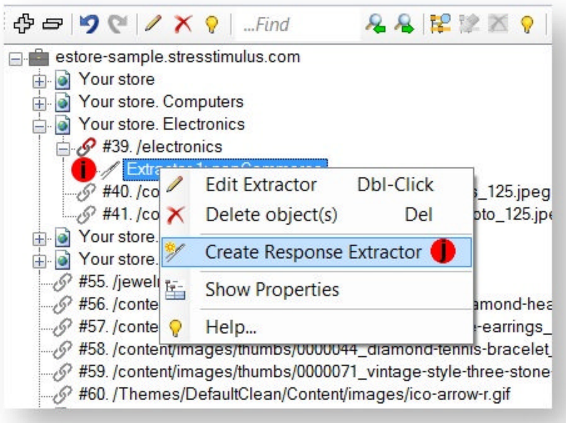
Sometimes it is necessary to create an extractor that selects a string or a value from another extractor. An example when this would be useful is when response body contains XML or JSON content intermingled with other content. The easiest way to extract a value from XML or JSON is by creating an XPath or JPath extractor, respectively. If you create a simple extractor from the body it will fail while parsing parts of the body which are not XML or JSON.

You can resolve this issue in two steps:

1. Create a simple text delimiter extractor which selects the entire XML or JSON block from the body, as described above.

2. Create an XML or JSON extractor from the simple extractor. To do so, right-click on the existing extractor (i) and then in the context menu, select **Create Response Extractor** (j). The same extractor control will appear. The response window (d) will display the content of the first simple extractor instead of the response content. Continue configuring the extractor as you would the simple extractor.

**Note:** Extractors can be created from other extractors in unlimited levels. Multiple related extractors are evaluated sequentially at run-time starting from the top simple extractor down the chain to the last extractor.



## Text Delimited Extractor

This is the most common extractor type. Its value is defined by extracting a string that occurs between two delimiters **Text Before** and **Text After** in the response.**Creating Text Delimited Extractor**

1. In the response window, find the dynamic value you want to extract, such as a token. Use the find box on the toolbar, if necessary.

2. Select preceding delimiter, and click **Set the selected text as Text Before** on the toolbar.

3. Select the succeeding delimiter and click **Set the selected text as Text After**.

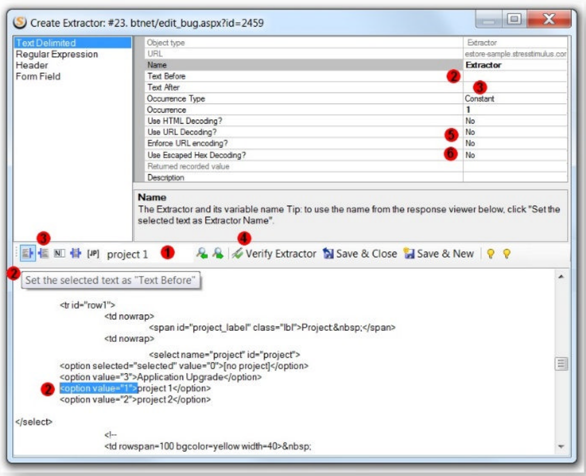
**Tip: When select delimiters consider these points:**

- Longer delimiters impact performance.
- Shorter delimiters increase chances that more than one pair of delimiters will be found (see the next step).

4. Click Verify Extractor on the toolbar and the calculated value will appear in the Extractor Check window. Compared to the expected value, it can be different in the following cases:

1. Incorrect delimiters. In this case, correct the delimiters and try again.
2. More than one pair of delimiters was found and therefore the Occurrence property must be properly set. Occurrence property is described in the next section.

5. Turn on HTML Decoding and URL Decoding properties if either encoding was used on the server to generate the response. Example of HTML Decoding is converting


Create Extractor: #23. btnet/edit_bug.aspx?id=2459

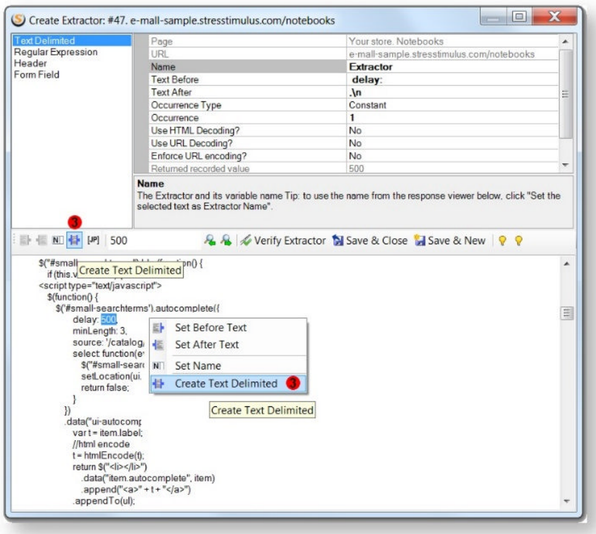
| | |
|---|---|
| Object type | Extractor |
| URL | estore-sample.stresstimulus.cor |
| Name | Extractor |
| Text Before 2 | |
| Text After | 3 |
| Occurrence Type | Constant |
| Occurrence | 1 |
| Use HTML Decoding? | No |
| Use URL Decoding? 5 | No |
| Enforce URL encoding? | No |
| Use Escaped Hex Decoding? 6 | No |
| Returned recorded value | |
| Description | |

**Name**
The Extractor and its variable name Tip: to use the name from the response viewer below, click "Set the selected text as Extractor Name".

```
        <tr id="row1">
                <td nowrap>
                        <span id="project_label" class="lbl">Project&nbsp;</span>
                <td nowrap>

                        <select name="project" id="project">
        <option selected="selected" value="0">[no project]</option>
        <option value="3">Application Upgrade</option>
        <option value="1">project 1</option>
        <option value="2">project 2</option>

</select>
                <!--
                <td rowspan=100 bgcolor=yellow width=40>&nbsp;
```

| | |
|---|---|
| "&gt;" to ">". Example of URL Decoding is converting "%3F" to "?".<br><br>6. Turn on Escaped Hex Decoding to apply escaped hex decoding to the Extractor value. An example of decoding: '\x23' to '#'. This is used in instances when extracting values from JSON responses. | |
| **Shortcut for creating Text Delimited Extractor**<br><br>1.<br>a. In the response window, find and highlight the dynamic value you want to extract.<br>b. Right click and select **Create Text Delimited** or click **Create Text Delimited** on the toolbar.<br>**c. Text Before** and **Text After** properties will be populated automatically.<br>d. Click **Verify Extractor** to confirm that the extractor is created correctly. |  |

Create Extractor: #47. e-mall-sample.stresstimulus.com/notebooks

Text Delimited
Regular Expression
Header
Form Field

| | |
|---|---|
| Page | Your store. Notebooks |
| URL | e-mall-sample.stresstimulus.com/notebooks |
| Name | **Extractor** |
| Text Before | **delay:** |
| Text After | **.\n** |
| Occurrence Type | Constant |
| Occurrence | 1 |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |
| Enforce URL encoding? | No |
| Returned recorded value | 500 |

**Name**
The Extractor and its variable name Tip: to use the name from the response viewer below, click "Set the selected text as Extractor Name".

```
$("#small          pn() {
  if (this.v
<script type="text/javascript">
  $(function() {
    $('#small-searchterms').autocomplete({
        delay: 500,
        minLength: 3,
        source: '/catalog,
        select: function(e'
          $("#small-searc
          setLocation(ui.
          return false;
        }
      })
      .data("ui-autocomp
        var t = item.label;
        //html encode
        t = htmlEncode(t);
        return $("<li></li>")
          .data("item.autocomplete", item)
          .append("<a>" + t + "</a>")
        .appendTo(ul);
```

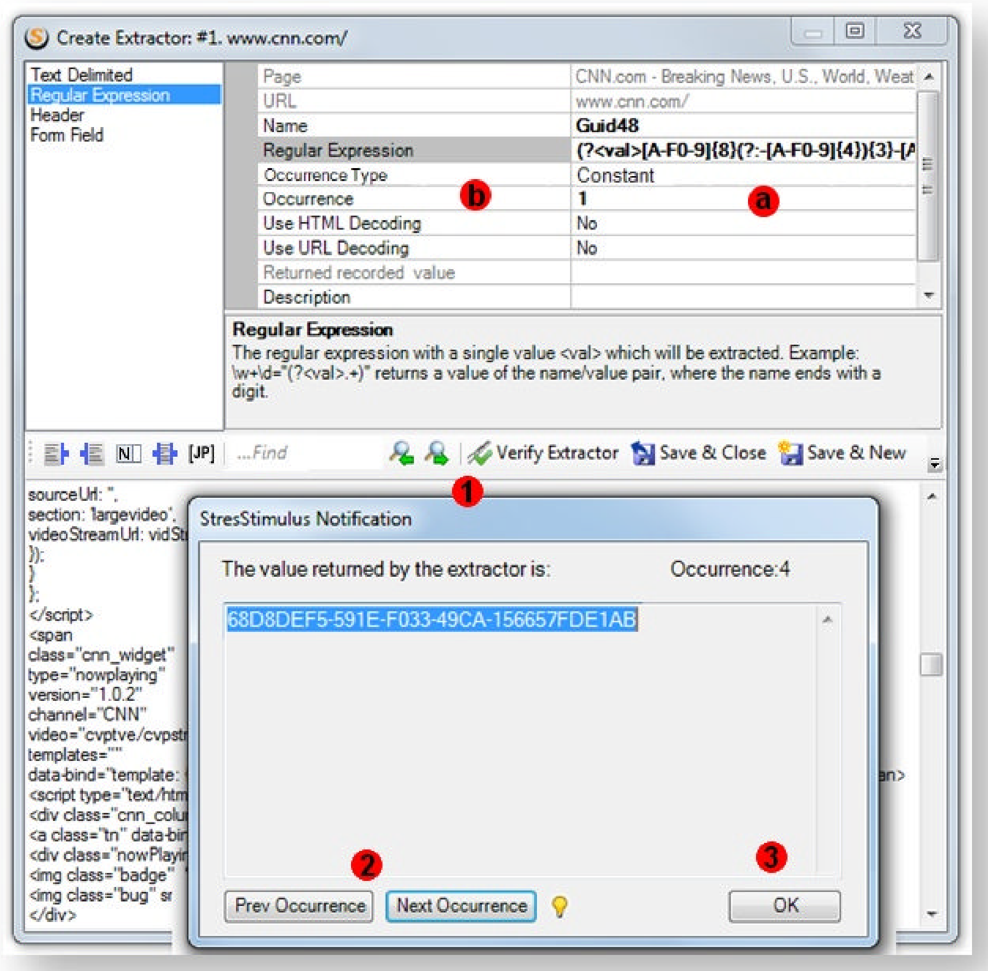
## Regular Expression Extractor

Regular Expression extractors should be used when you can't define an extractor using text delimiters. For example, you need to extract a GUID, but you did not know where it will appear in the response and its value.

The difference between the regular expression extractors and text delimited extractors is that instead of delimiters, you need to define the Regular Expression property (a). For example, if you need to extract a value in a name/value pair combination and the name is some text followed by a number, then you can define a regular expression as follows:

```
\w+\d+="(?<value>.+)"
```

The regular expression for GUID is: `(?<val>[A-F0-9]{8}(?:-[A-F0-9]{4}){3}-[A-F0-9]{12})`



| | |
|---|---|
| Page | CNN.com - Breaking News, U.S., World, Weat |
| URL | www.cnn.com/ |
| Name | Guid48 |
| Regular Expression | (?<val>[A-F0-9]{8}(?:-[A-F0-9]{4}){3}-[A |
| Occurrence Type | Constant |
| Occurrence | 1 |
| Use HTML Decoding | No |
| Use URL Decoding | No |
| Returned recorded value | |
| Description | |

**Regular Expression**
The regular expression with a single value <val> which will be extracted. Example: \w+\d="(?<val>.+)" returns a value of the name/value pair, where the name ends with a digit.

The value returned by the extractor is: Occurrence:4

68D8DEF5-591E-F033-49CA-156657FDE1AB

If the regular expressions find more than one value, the Occurrence property (b) must be properly set. The search starts from the beginning of the response. By default, the first occurrence of the value will be used for the extractor. In order to use the second matching value, the Occurrence property should be changed from 1 to 2. To do so:

1. Click Verify Extractor on the toolbar and compare the value in the Extractor Check window with the expected value.
2. If an incorrect value is displayed, find the correct occurrence of the matching value by clicking Next/Previous Occurrence until you find the correct value.
3. Click Set The Occurrence to properly adjust the Occurrence property.

For instruction on how to set other properties of the regular expression extractors, see the previous section Text Delimited Extractors.

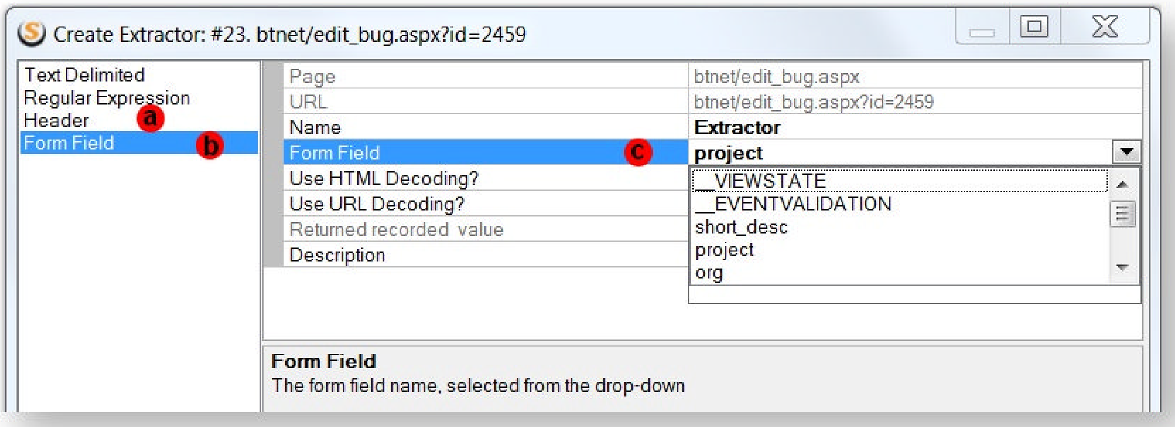
For the regular expressions reference, see this source.

### Header, Form Field and XPath/JPath Extractor

**Header extractors** (a) return a response header's value. To define the Header Extractor, select the header name from the Header (a) drop-down.

For instructions on how to set other properties of the Header extractor, see the section Text Delimited Extractors.

If a response is a web form, then you can use **Form Field (b) extractors.** They return a web form field value, for example a value of any html textbox or hidden field. To define a Form Field Extractor, select the field name from the Form Field (c) drop-down.


Create Extractor: #23. btnet/edit_bug.aspx?id=2459

Text Delimited
Regular Expression
Header
Form Field

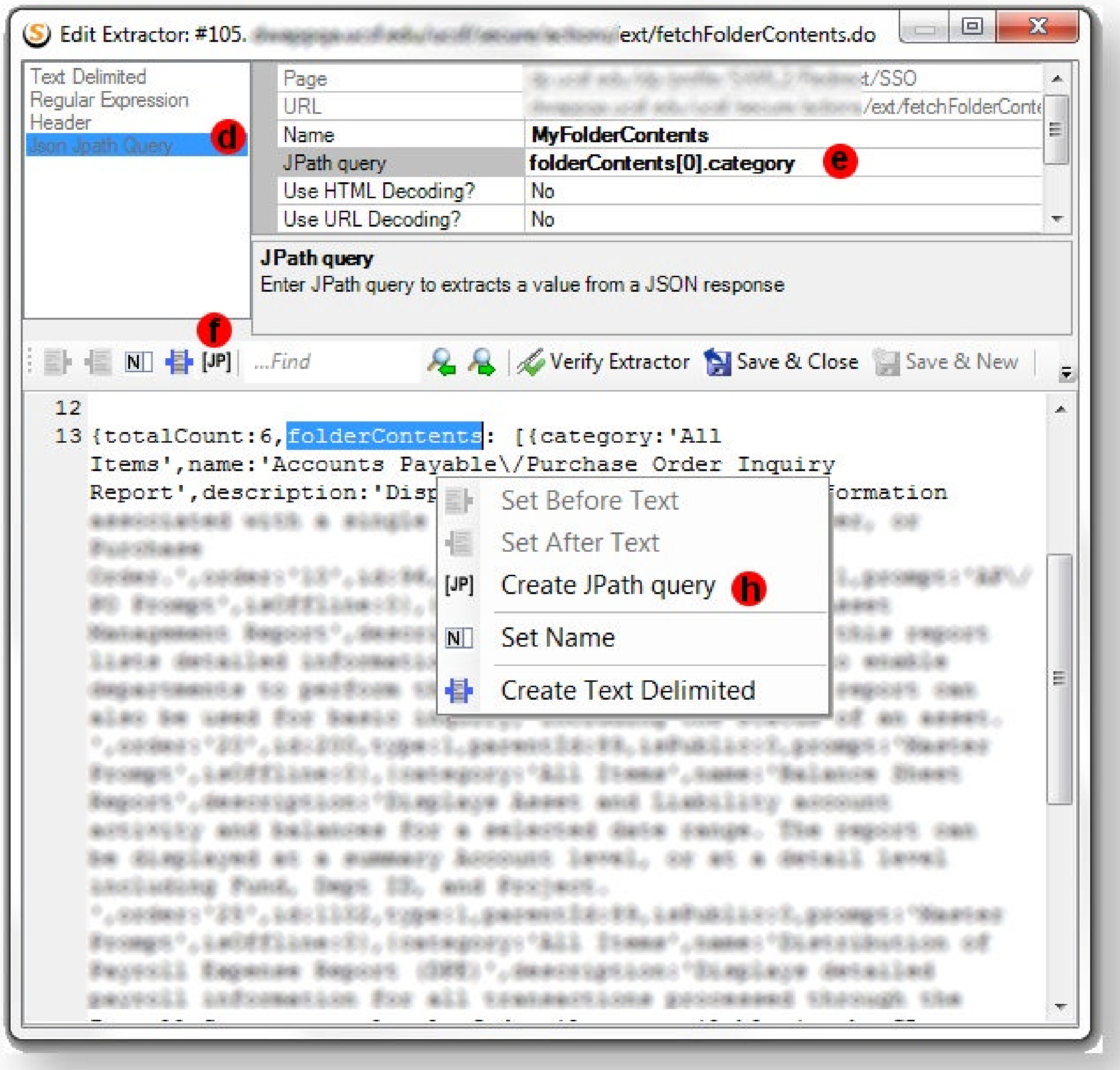
| | |
|---|---|
| Page | btnet/edit_bug.aspx |
| URL | btnet/edit_bug.aspx?id=2459 |
| Name | **Extractor** |
| Form Field | **project** |
| Use HTML Decoding? | __VIEWSTATE |
| Use URL Decoding? | __EVENTVALIDATION |
| Returned recorded value | short_desc |
| Description | project |
| | org |

**Form Field**
The form field name, selected from the drop-down

If a response is XML, then you can use an **XPath Extractor**. The difference between XPath extractors and text delimited extractors is that instead of delimiters, you need to define an XPath query.

The XPath tutorial can be found at this location.

If a response is JSON, then you can use a **JPath Extractor** (d). The difference between JPath extractors and text delimited extractors is that instead of delimiters, you need to define an **JPath** query (e). You can also use the JPath creator feature by selecting the value you want to extract and clicking the Create JPath button (f) or by right clicking and selecting **Create JPath Query** (g) in the context menu. StresStimulus will automatically create a JPath extractor that will extract the selected value.



| | |
|---|---|
| Page | t/SSO |
| URL | /ext/fetchFolderConte |
| Name | MyFolderContents |
| JPath query | folderContents[0].category |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |

**JPath query**
Enter JPath query to extracts a value from a JSON response

```
12
13 {totalCount:6,folderContents: [{category:'All
   Items',name:'Accounts Payable\/Purchase Order Inquiry
   Report',description:'Disp                          formation
```

The JPath description can be found at this location.

## Emulating Clicks on Random Links

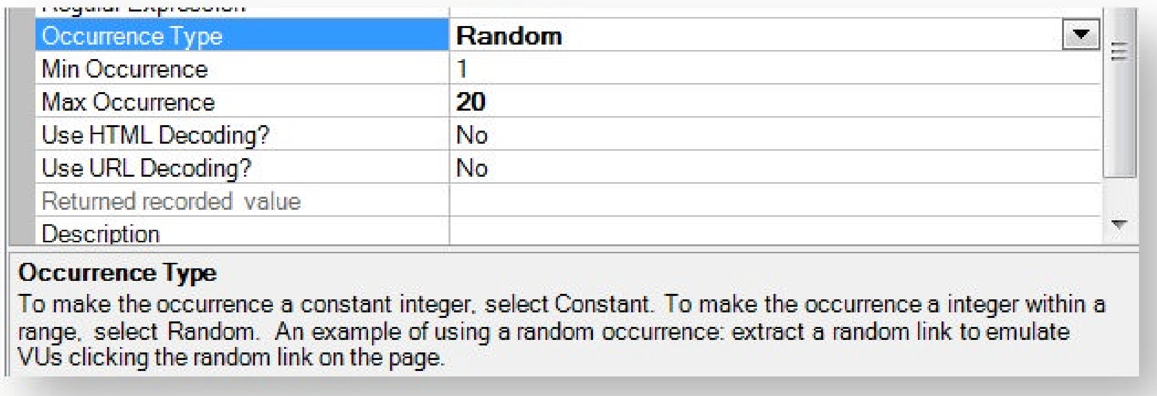
Sometimes a test scenario cannot be defined precisely because different users choose different website navigation routes. For example, a content management system (CMS) runs a news website. Pages with news articles are created daily, and the corresponding links are updated at the same time. The goal of a load test is to emulate users visiting different pages depending on their personal interests.

In order to emulate a click on a random link, create an extractor with a generic search rule for finding a hyperlink.

**Examples**: This is an example of a regular expression finding a hyperlink:
`<a\s+href="(?<url>.+?)".*>.*?</a>`

This is another example of a regular expression finding a relative hyperlink :
`<a\s+href="(?<url>[^:]+?)".*>.*?</a>`

Then randomize the extractor's Occurrence property so different VUs will extract different URLs. To do so, set the Occurrence Type property to Random, and define the range of links by setting the Min Occurrence and Max Occurrence properties. And finally, use this extractor to parameterize the subsequent request header's {PATH} property.



| | |
|---|---|
| Occurrence Type | Random |
| Min Occurrence | 1 |
| Max Occurrence | 20 |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |
| Returned recorded value | |
| Description | |

**Occurrence Type**
To make the occurrence a constant integer, select Constant. To make the occurrence a integer within a range, select Random. An example of using a random occurrence: extract a random link to emulate VUs clicking the random link on the page.

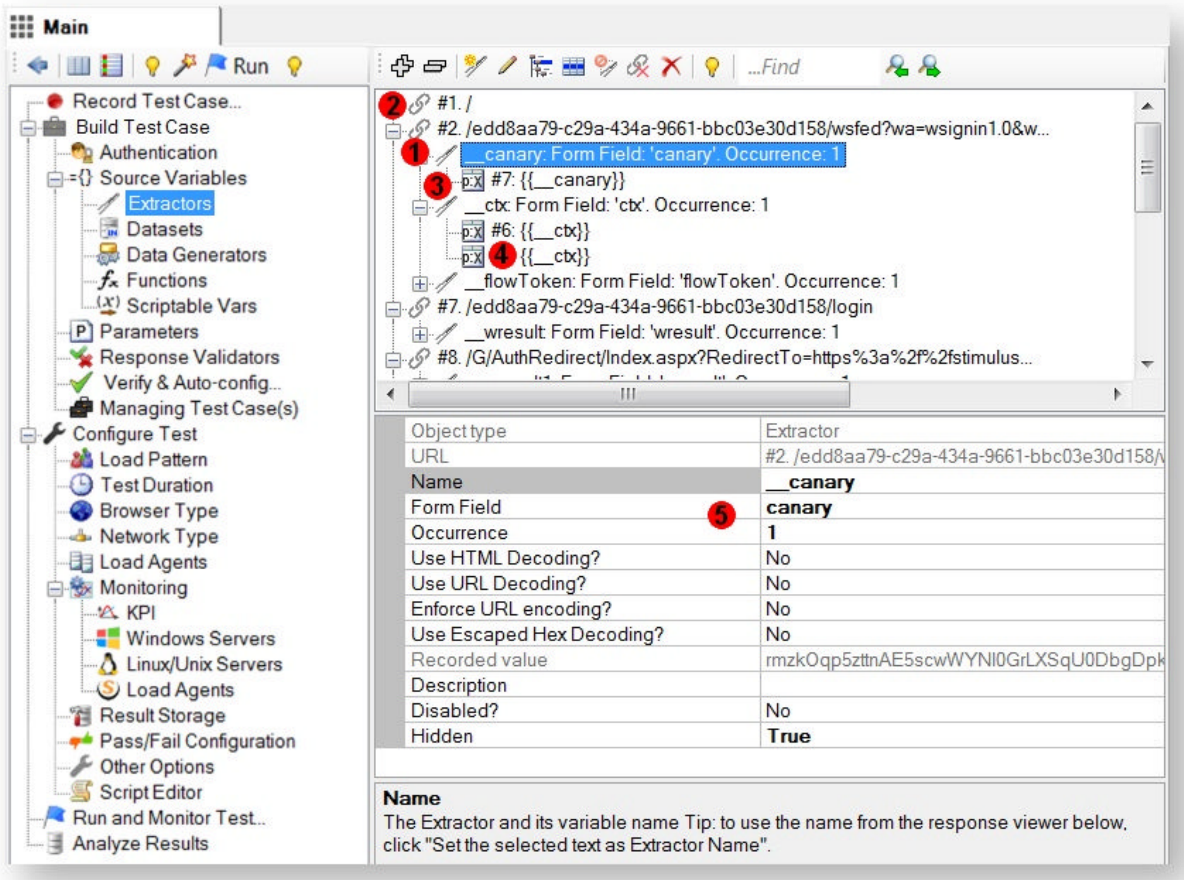
Since all links are determined dynamically, the same test can be used without re-recording after the website was updated and links were changed. This approach allows to create a large number of random and unpredictable traversing routes that are necessary to test. Combining this approach with random think times allows to realistically emulate the impact of a user-base on server performance and to test a very large number of real-life situations.

## Extractor Tree

Extractors (1) are created from responses (2). They are displayed in sequential order.

Each extractor is used to create one or several parameters in subsequent requests. Parameters are displayed as subordinate elements (3) of this extractor. The request number (4) where the parameter is created precedes the parameter.

The property grid (5) displays the properties of the selected extractor.



```
#1. /
#2. /edd8aa79-c29a-434a-9661-bbc03e30d158/wsfed?wa=wsignin1.0&w...
    __canary: Form Field: 'canary'. Occurrence: 1
        #7: {{__canary}}
    __ctx: Form Field: 'ctx'. Occurrence: 1
        #6: {{__ctx}}
        {{__ctx}}
    __flowToken: Form Field: 'flowToken'. Occurrence: 1
#7. /edd8aa79-c29a-434a-9661-bbc03e30d158/login
    __wresult: Form Field: 'wresult'. Occurrence: 1
#8. /G/AuthRedirect/Index.aspx?RedirectTo=https%3a%2f%2fstimulus...
```

| | |
|---|---|
| Object type | Extractor |
| URL | #2. /edd8aa79-c29a-434a-9661-bbc03e30d158/\ |
| Name | __canary |
| Form Field | canary |
| Occurrence | 1 |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |
| Enforce URL encoding? | No |
| Use Escaped Hex Decoding? | No |
| Recorded value | rmzkOqp5zttnAE5scwWYNl0GrLXSqU0DbgDpk |
| Description | |
| Disabled? | No |
| Hidden | True |

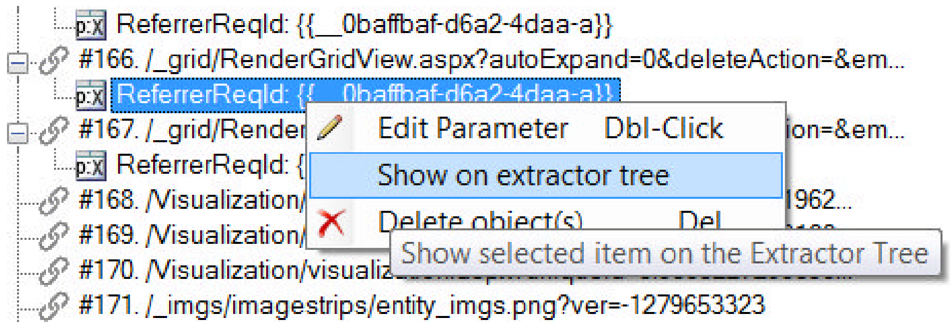
**Name**
The Extractor and its variable name Tip: to use the name from the response viewer below, click "Set the selected text as Extractor Name".

The list of extractor commands and properties is provided here.

## Matching Extractors to Parameters

The extractor tree automatically matches parameters to extractors. Sometimes it is necessary to solve the opposite task, matching an extractor to a parameter.

To do so, on the Test Case Tree, select a parameter, right click, and chose **Show on extractor tree**. StresStimulus will highlight the matching extractor on the Extractor Tree.

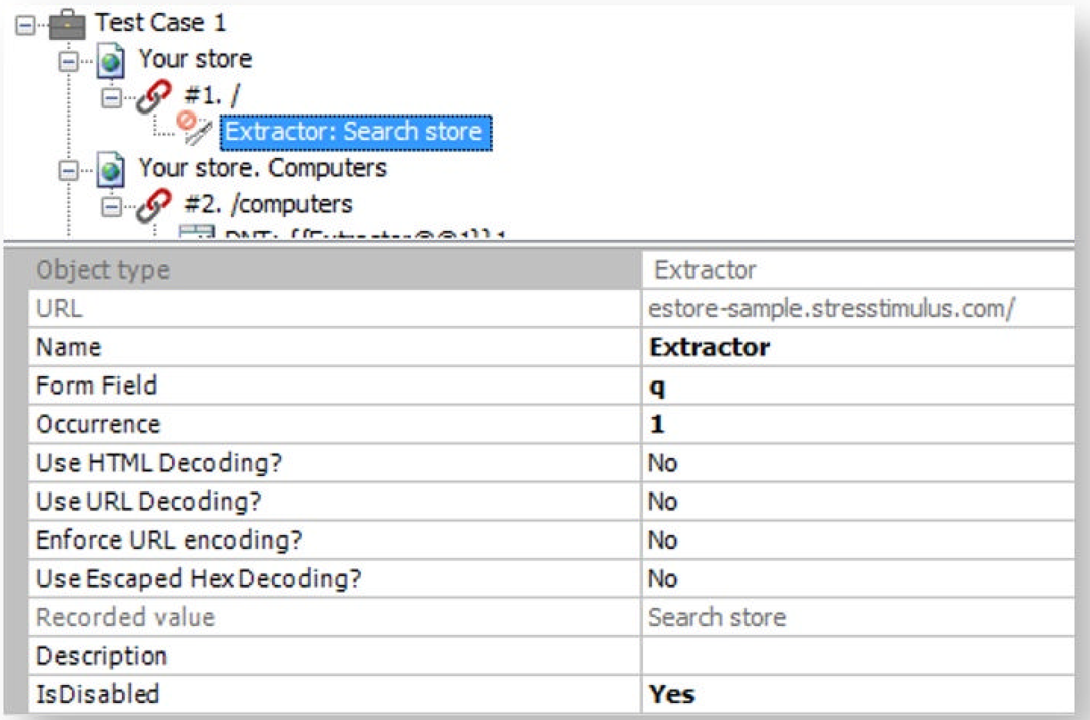

ReferrerReqId: {{__0baffbaf-d6a2-4daa-a}}
#166. /_grid/RenderGridView.aspx?autoExpand=0&deleteAction=&em...
ReferrerReqId: {{__0baffbaf-d6a2-4daa-a}}
#167. /_grid/Render...ion=&em...
ReferrerReqId: {
#168. /Visualization/...1962...
#169. /Visualization/
#170. /Visualization/visualiz...
#171. /_imgs/imagestrips/entity_imgs.png?ver=-1279653323

## Disabling and Hiding Extractors

Sometimes it may be necessary to check if extractor created manually or by autocorrelation is required. To do so, you can temporary disable an extractor by changing its property **IsDisabled** to "Yes".

By default all extractors are enabled, so the default value for this property is "No".

Changing the IsDisabled parameter to "Yes" will automatically disable all the dependent parameters. When you re-enable the extractor, all related parameters are automatically re-enabled.



| | |
|---|---|
| Object type | Extractor |
| URL | estore-sample.stresstimulus.com/ |
| Name | **Extractor** |
| Form Field | **q** |
| Occurrence | **1** |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |
| Enforce URL encoding? | No |
| Use Escaped Hex Decoding? | No |
| Recorded value | Search store |
| Description | |
| IsDisabled | **Yes** |

## Datasets

Datasets store predefined sets of data that are used to feed request during the test execution. The data is stored in tabular form with multiple rows and columns. A test can have multiple datasets.

The dataset is also a variable that dynamically selects a single value from a specific column and replaces a recorded value in the request parameter.

The row from which the value is retrieved is determined dynamically based on a specified databinding rule. For example, the value can be selected from a random or a subsequent row.

## Creating Datasets

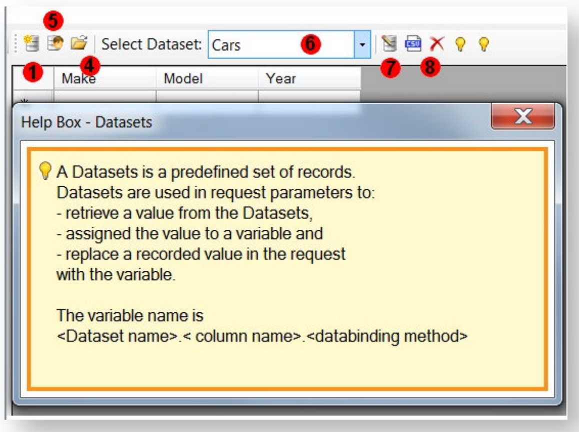
After a dataset is created it is stored in the test folds as a csv file. The following methods of creating data sets are supported:

- Creating a dataset structure and entering data in StresStimulus.
- Creating a structure of an empty dataset in StresStimulus, copy tabular data from any external data sources, such as Excel, and paste it to the dataset grid in StresStimulus.
- Creating a structure of an empty dataset in StresStimulus and managing the .csv file outside StresStimulus UI. For example, the entire .csv file can be re-generated using external data sources before running a new test.
- Importing an external data source as a new dataset.

Once a data set is created, it can be maintained in StresStimulus. Data can be deleted or modified manually, and additional data can be entered.

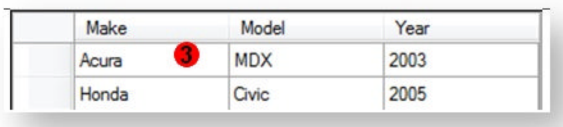
## How to create and edit a dataset

1. Click the create a new Dataset button and the Add/Edit Dataset dialog will popup,

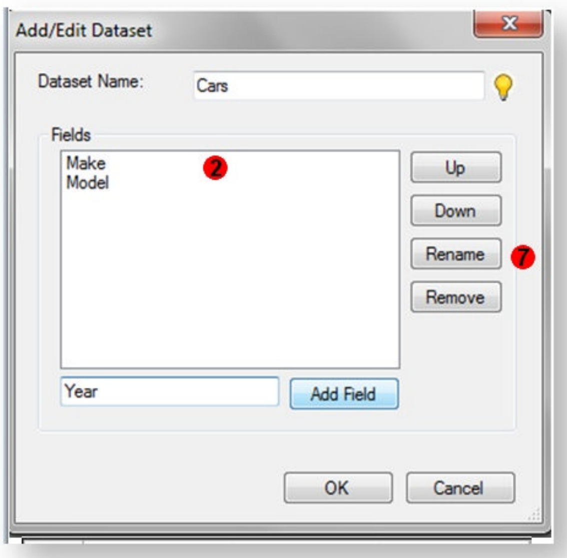


A Datasets is a predefined set of records.
Datasets are used in request parameters to:
- retrieve a value from the Datasets,
- assigned the value to a variable and
- replace a recorded value in the request
with the variable.

The variable name is
<Dataset name>.< column name>.<databinding method>



|  | Make | Model | Year |
|---|---|---|---|
|  | Acura | MDX | 2003 |
|  | Honda | Civic | 2005 |



Add/Edit Dataset

Dataset Name: Cars

Fields

Make
Model

Year

2. Set the dataset name and add the necessary fields. The dataset name will appear in the Variable Picker when you create parameters. Click OK when done. This will create an empty Dataset.

Note: For better readability, using a space and many special characters as a part of the data set and field names is permitted. Exceptions are "." and "$"

3. After that, you can type or paste data into the grid.

4. To create a dataset with a user's credentials for form authentication, click Authentication Dataset.

**Tip:** For Basic, Windows Integrated (e.g. NTLM) or other Kerberos authentication use Authentication section on the Test Workflow Tree.

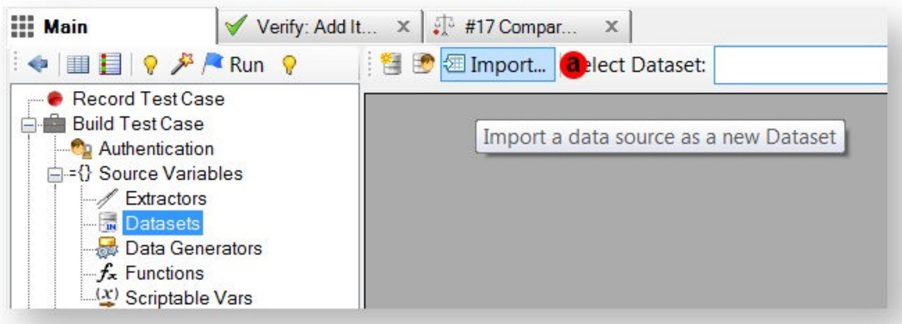
5. To edit data, select a dataset from the drop-down and edit data in the grid

6. To rename a dataset or edit its structure, click Edit. To add a field, enter the field name and click Add Field. Double-click the field to rename it. To reposition, rename or delete a selected field, use the Up, Down, Rename or Remove buttons.

7. Other available operations are Export Data and Delete.

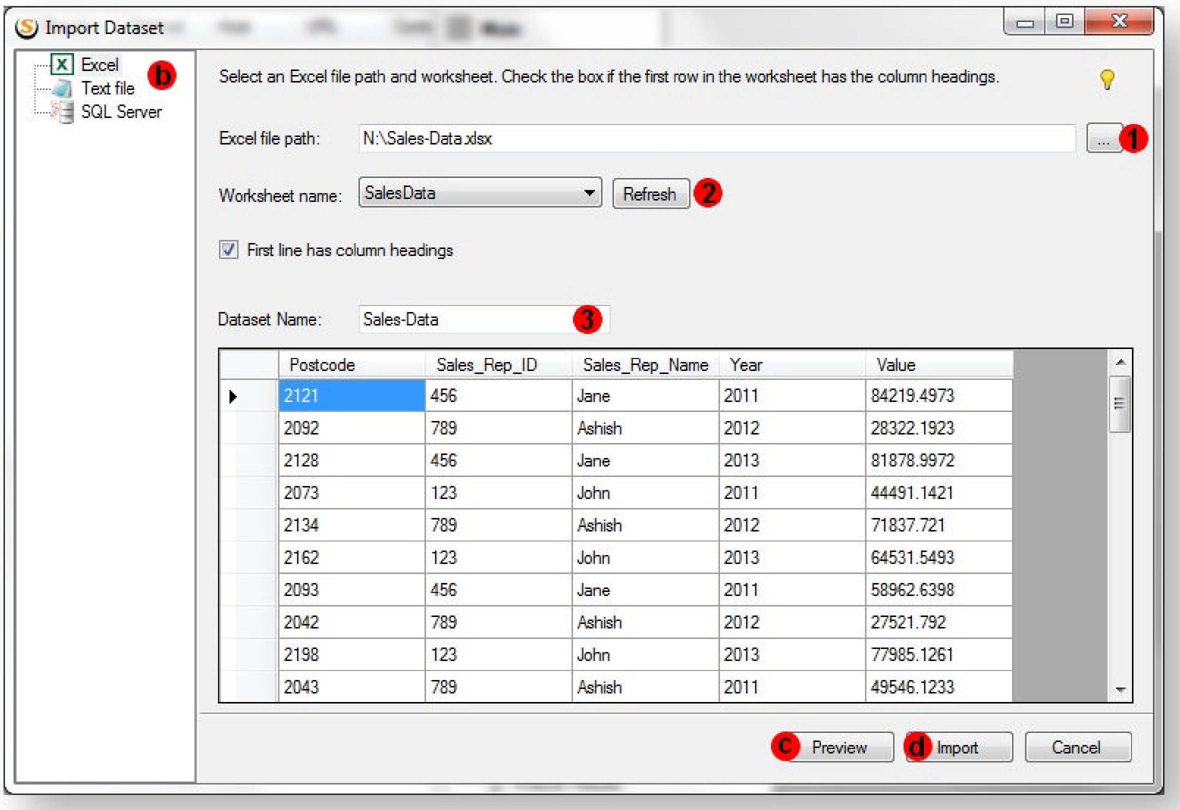
## Importing Datasets

To create a data set with external data, click the **Import** button (a). In the Import Dataset window, select the data source on the left (b). Either Excel, Text file, or SQL Server query. Fill out the corresponding form to import a dataset. Click the **Preview** button (c) to see a preview of the dataset. When you are ready to import the dataset, give it a unique name and click the **Import** button (d).



Excel:

1. Browse for or provide a path to the Excel file.
2. Click Refresh button to refresh the worksheets in the Excel workbook.
3. Check the First line has column headings, if the first row contains column headings.



Select an Excel file path and worksheet. Check the box if the first row in the worksheet has the column headings.

Excel file path: N:\Sales-Data.xlsx

Worksheet name: SalesData

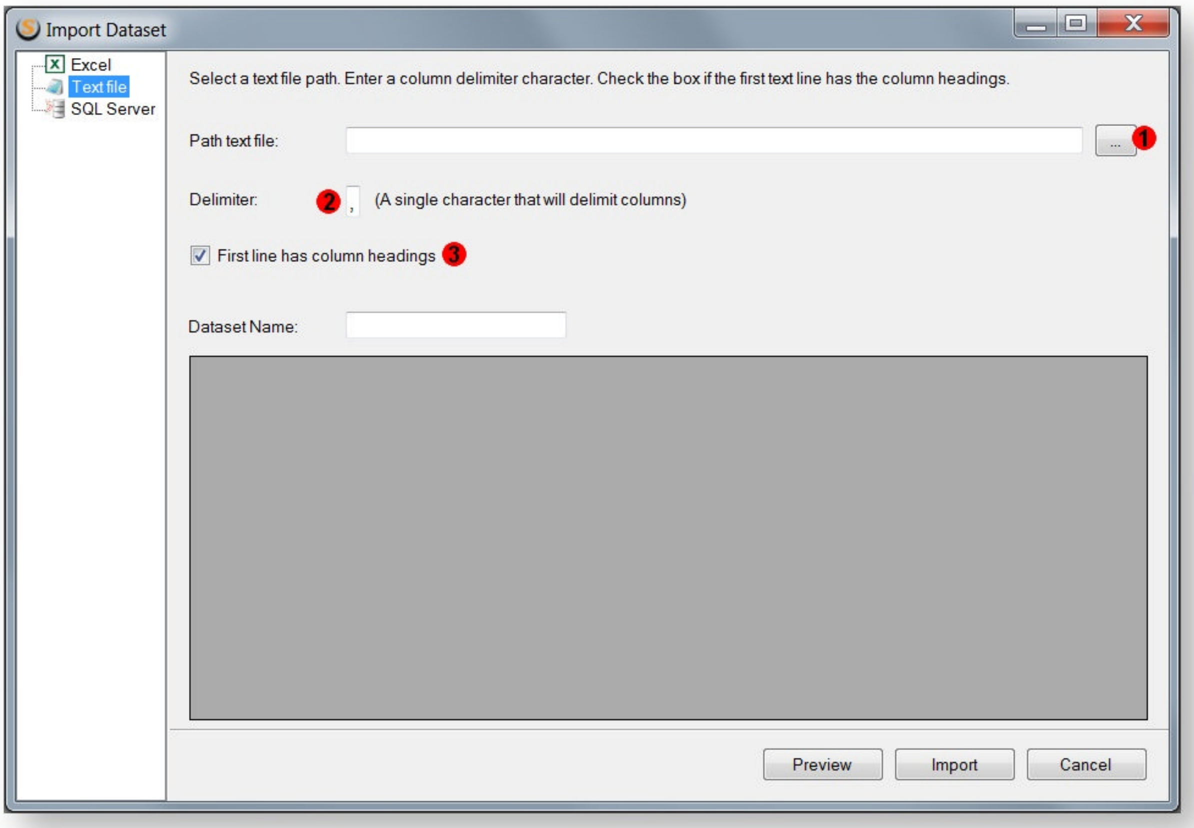
☑ First line has column headings

Dataset Name: Sales-Data

| Postcode | Sales_Rep_ID | Sales_Rep_Name | Year | Value |
|---|---|---|---|---|
| 2121 | 456 | Jane | 2011 | 84219.4973 |
| 2092 | 789 | Ashish | 2012 | 28322.1923 |
| 2128 | 456 | Jane | 2013 | 81878.9972 |
| 2073 | 123 | John | 2011 | 44491.1421 |
| 2134 | 789 | Ashish | 2012 | 71837.721 |
| 2162 | 123 | John | 2013 | 64531.5493 |
| 2093 | 456 | Jane | 2011 | 58962.6398 |
| 2042 | 789 | Ashish | 2012 | 27521.792 |
| 2198 | 123 | John | 2013 | 77985.1261 |
| 2043 | 789 | Ashish | 2011 | 49546.1233 |

Text file

1. Browse for or provide a path to the text file.
2. Type the delimiter character that will separate the columns. For instance, if the text file is a .csv then the ',' should be the delimiter.
3. Check the First line has column headings, if the first row contains column headings.



Import Dataset

Select a text file path. Enter a column delimiter character. Check the box if the first text line has the column headings.

Path text file: 1

Delimiter: 2 , (A single character that will delimit columns)

☑ First line has column headings 3

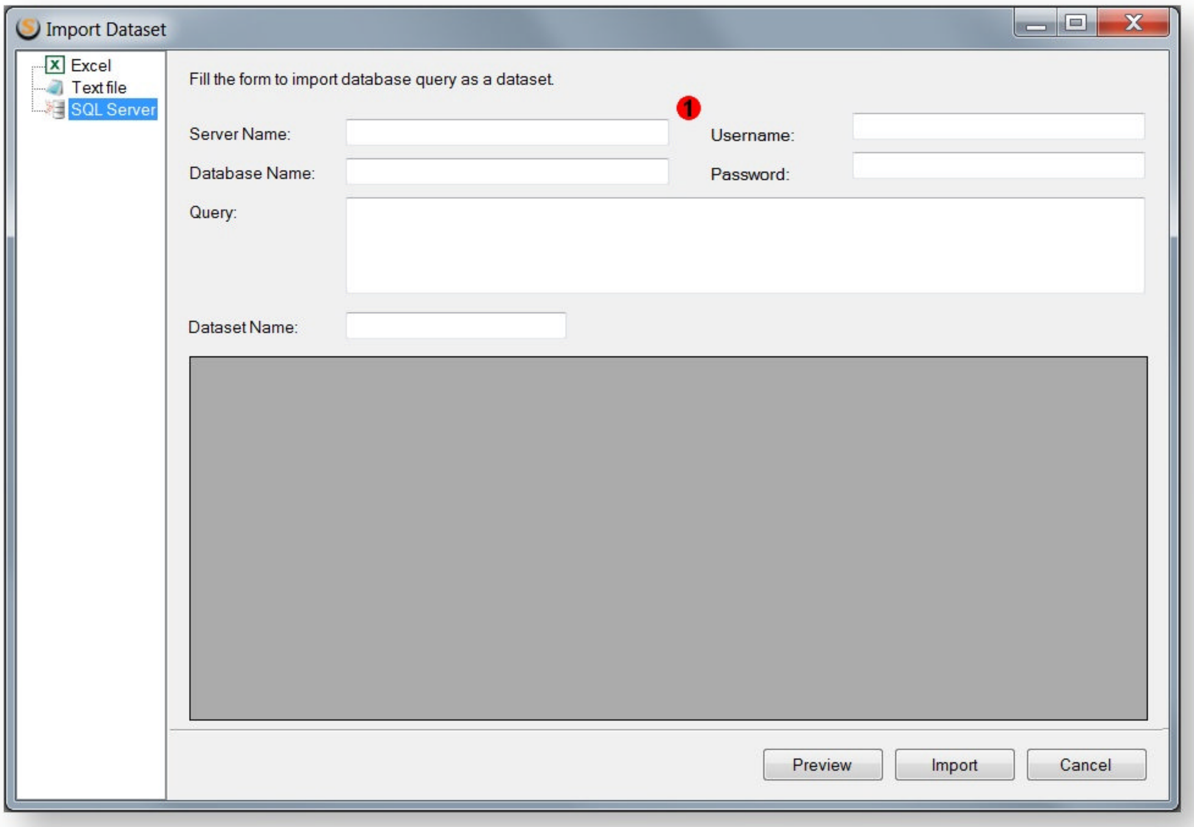
Dataset Name:

**Special Characters**

- Special characters can be included into the dataset text file using URL encoding. To insert a character with hexadecimal code XX, add the %XX sequence to the file. In rare cases, when %XX should be interpreted as a 3-character string, URL-encode the percent character, so the string will look like %25XX
- The coma character is used as a field separator in the text file. In order to use a coma within values, surround it with quotes (ex. ",")

SQL Server

1. Fill out the SQL Server form. Provide the Server Name, Database Name, Credentials, and a Select Query.



Fill the form to import database query as a dataset.

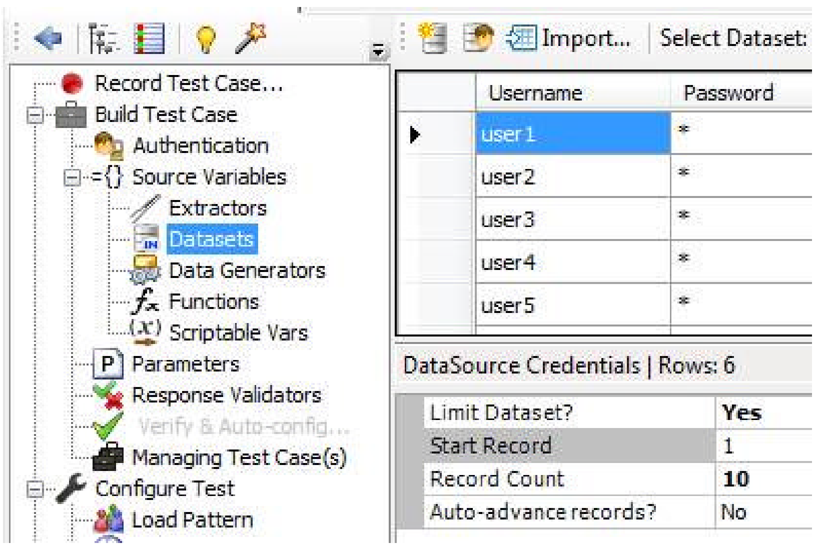
Server Name:
Username:
Database Name:
Password:
Query:
Dataset Name:

## Dataset Ranges

It is sometimes necessary to use unique records from a dataset for every test run, for example in situations where each dataset record contains data to be inserted into the database and the same data can't be inserted twice. One way to accomplish this is before running a subsequent test, delete all the records from the dataset and replace them with new records. Another way is to limit the dataset ranges and only use a subset of records for the test.

To enable dataset ranges do the following:

1. In the dataset set the **Limit Dataset** property to **Yes**.
2. Set the starting record to use in the current test.
3. Set the record count to set the number of records to use from the starting record.
4. Set the auto-advance records to Yes to automatically advance the starting record (by record count amount) after each test run. If the starting record value gets larger than the actual number of records, the starting record count will reset to 1.



| Username | Password |
|---|---|
| user1 | * |
| user2 | * |
| user3 | * |
| user4 | * |
| user5 | * |

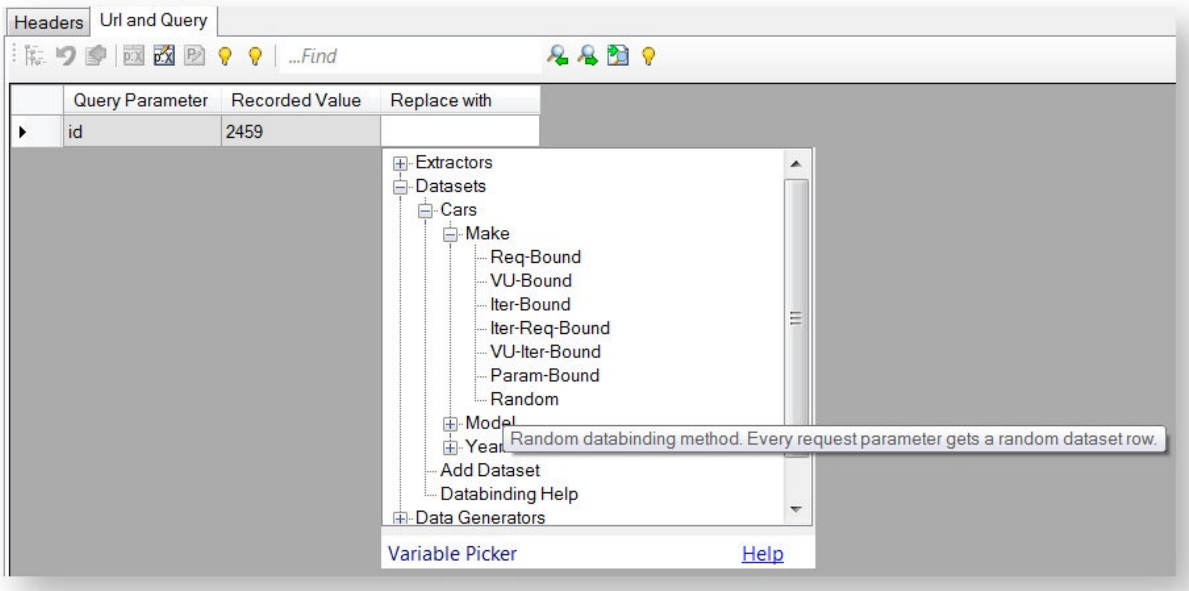
DataSource Credentials | Rows: 6

| | |
|---|---|
| Limit Dataset? | Yes |
| Start Record | 1 |
| Record Count | 10 |
| Auto-advance records? | No |

## Using Datasets

Datasets are used to feed request during the test execution. A dataset variable is associated with a specific datasets column. The row from which the value is retrieved is determined based on a specified databinding rule.

The dataset variable is referred by as `[dataset name].[field name]$[data bind method]` surrounded by the brackets. For example: `{{Users.username$VU_Bound}}`. When the Variable Picker is used to create a parameter, the properly formatted dataset name reference is inserted automatically.



| Query Parameter | Recorded Value | Replace with |
|---|---|---|
| id | 2459 | |

Random databinding method. Every request parameter gets a random dataset row.

StresStimulus supports 7 databinding methods:

1. Request-Bound (Req-Bound)
2. VU-Bound (VU-Bound)
3. Iteration-Bound (Iter-Bound)
4. Iteration-Request-Bound (Iter-Req-Bound)
5. VU-Iteration (VU-Iter-Bound)
6. Parameter-Bound (Param-Bound)
7. Random (Random)

The dataset value is fetched just before issuing the requests. It replaces a recorded value in the request parameter.

For more information about databinding methods, see Databinding.

**Info:**

Sometimes it is necessary to parameterize dynamic file upload. The recorded request typically will be a multipart post request where one of the fields has a value containing the file content. For example, the request will look like:

```
-----------------------------3e04d523481038
Content-Disposition: form-data; name="submittedfile"; filename="SELECTED.pdf"
Content-Type: application/pdf
%PDF-1.7
%□□□□□
7052 0 obj
```

The field "submittedfile" here represents the recoded content of the file. In the course of load testing you want to replace this content with dynamically selected files with the content of files locally available to the load testing machine. To do so, create a dataset with a field for storing the full paths of such files. Then parameterize the multipart request by replacing the value of the field with the recoded file content (in this case "submittedfile" field) with the dataset.

StresStimulus will replace recorded file content with the dynamic content during replay.

**Note**: if you are using this feature on a distributed test with one or more agents, then all files must exist on all the agents in the same location.

**Tip:** One dataset can be used in several test cases.

### Data Generators

Ent SP

A data generator (DG) returns a generated on-the-fly random value assigned to a variable with the same name, which is used to parameterize requests. Depending on the DG type, the value can be random or systematic. DG can be used for creating streams of emulated business data of different types which can be included into the HTTP load stream. It allows to more realistically mimic physical users interacting with web application by entering real-world business data. It helps to avoid creating large test datasets manually.

A data generator type determines what random value will be returned: Integer, Double, Date/Time, GUID or Text.

### The following DG types are supported

- Integer - Returns an integer between the Min Value and Max Value.

- Set Type to AutoIncrement to generate sequential integers starting from the Min Value. After the Max Value is reached, the next integer is the Min Value. If uniqueness is required, make sure that Max Value will not be reached.
- Set Type to Random to generate random integers between the Min Value and Max Value. The uniqueness of the random numbers is not guaranteed.

- Double - Returns a double precision point number between the Min Value and Max Value.
- Datetime - Returns a date time object between the Min Value and Max Value.
- Guid - Returns a new Guid.
- Text - Returns a set of characters with a length between Min Length and Max Length.

### How to create a Data Generator

1. Select the data generators node on the Test Workflow Tree.
2. Click Create New Data Generator to bring up the generator dialog.
3. Select DG type that will return the appropriate datatype.
4. Set **When To Evaluate** property to **On Iteration** or **On Request**

**Tip:**

Set the **When To Evaluate** property to **On Iteration** to generate a new value on every iteration (default). In this case, the random values persist throughout the iteration for every VU. This allows to use the same random value more than once.

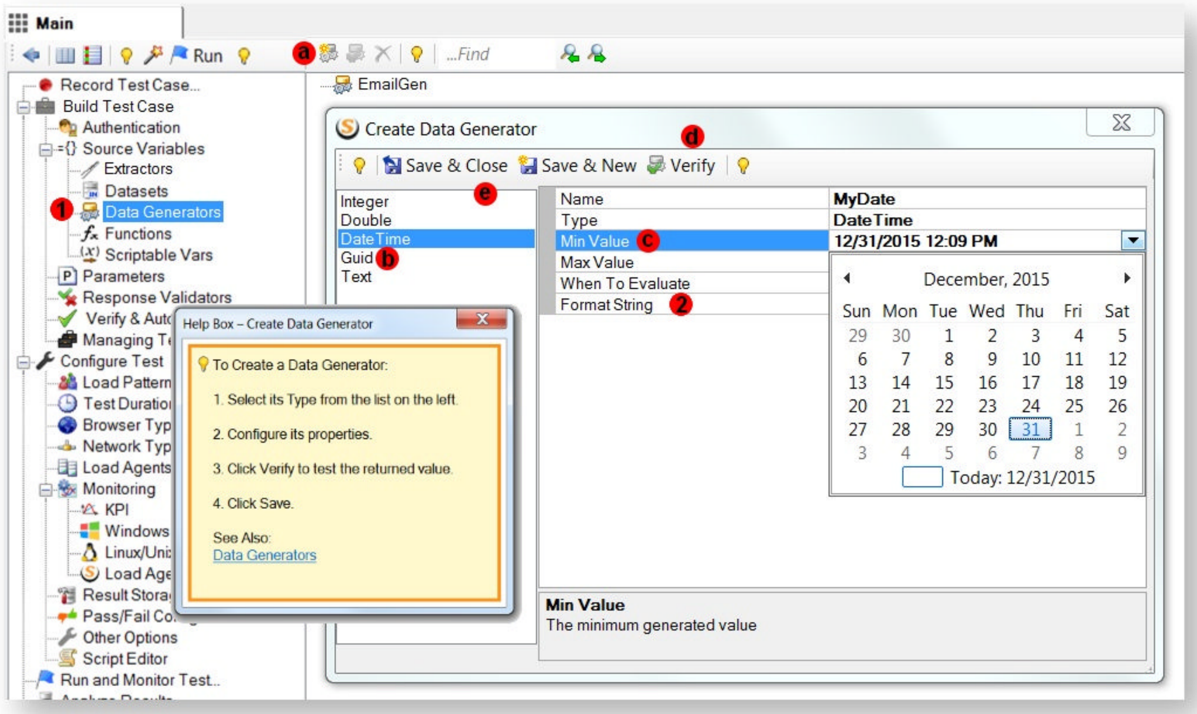
Example: A web form requires entering an email address created by a data generator, twice for verification purposes. In this case, set **When To Evaluate** property to **On Iteration** to avoid failing due to entry mismatch

Alternatively, set it to **On Request** to generate a new value on every on every request

5. Configure remaining properties and give it a unique name. It will appear in the Variable Picker when you create parameters. Use the Format String property to format the generator output.

6. Click Verify to test the return value.

7. Click **Save & Close**.



| | |
|---|---|
| Name | MyDate |
| Type | DateTime |
| Min Value | 12/31/2015 12:09 PM |
| Max Value | |
| When To Evaluate | |
| Format String | |

To Create a Data Generator:

1. Select its Type from the list on the left.
2. Configure its properties.
3. Click Verify to test the returned value.
4. Click Save.

See Also:
Data Generators

**Min Value**
The minimum generated value

C:\NING\2013\2013-02\pic_31.png

When the DG is used to parameterize a request, it is referred to by its name surrounded by the brackets, for example: `{{MyRndInteger}}`.

When the Variable Picker is used to create a parameter, a properly formatted DG name reference is inserted automatically.

## Functions

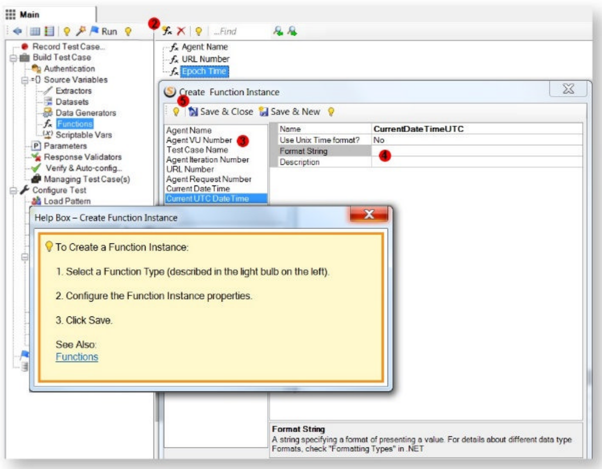
A function instance returns a determined on-the-fly value of a specific type. The value is assigned to a variable with the same name, which is used to parameterize requests.

The returned value depends on the function type and the context where the function is called from. Some of the function types provide access to internal load engine variables.

The function instance is evaluated before issuing a request.

The following function types are supported:

| Function | Return Value |
|---|---|
| `Agent Name` | The name of the current agent |
| `Agent VU Number` | The current VU number within an Agent's pool of VUs |
| `Test Case Name` | The current test case name |
| `Agent Iteration Number` | The current iteration number executed by an Agent |
| `URL Number` | The current request number within a test case |
| `Agent Request Number` | The current request number issued by an Agent from the beginning of the test |
| `Current Date Time` | The current date time stamp |
| `Current UTC Date Time` | The current date UTC time stamp |



| | |
|---|---|
| Name | CurrentDateTimeUTC |
| Use Unix Time format? | No |
| Format String | |
| Description | |

Help Box – Create Function Instance

To Create a Function Instance:

1. Select a Function Type (described in the light bulb on the left).
2. Configure the Function Instance properties.
3. Click Save.

See Also:
Functions

**Format String**
A string specifying a format of presenting a value. For details about different data type Formats, check "Formatting Types" in .NET

How to Create a Function Instance

1. Select the **Function** node on the **Test Workflow Tree**.
2. Click **New** to bring up the **Function** dialog.
3. Select a **Function Type.**
4. Configure its properties and give it a unique name. It will appear in the **Variable Picker** when you create parameters. Use the **Format String** property to format the generator output.
5. Click **Save & Close**

When the Function instance is used to parameterize a request, it is referred by its name surrounded by brackets, for example: {{MyFunction}}

When the Variable Picker is used to create a parameter, the properly formatted function instance reference is inserted automatically.

The function instance is evaluated just before issuing the requests. For example, if you use the **Current Date Time** function in 3 requests, it will be called 3 times and will return 3 different instant time values.

Date Time Functions

StresStimulus supports two date and time functions:

- `Current Date Time` - Returns the local date and time at the time of evaluation.
- `Current UTC Date Time` - Returns the universal (GMT) date and time at the time of evaluation.

These functions use the following additional attributes:

- Time offset (s) - The number of seconds to add (or subtract if the value is negative) from the resulting date time. Example: use this property to create an expiration time in the future.

- Use Unix Time format - This returns the date time as Unix time (Epoch time) format which is the number of seconds that have elapsed since January 1, 1970.

### Formatting Data Generator and Function Output

Standard .NET formatting strings are used to format the output of the Data Generators and Function. This provides a robust method of representing any numeric, GUID and date/time data types as strings.

To receive a desire output string, enter the necessary formatting string into the FormattingString property of a Data Generator. For example, to represent long format date and time as

"25 February 2013 10:25:30", use FormatString "U"

A quick reference guide to format string in StresStimulus is provided in the post Format Specifiers available here.

The complete reference of .NET Formatting Types is provided here.

**Note:** The **Current Date Time** and **Current UTC Date Time** Functions have an additional formatting property: **Use Unix Time format?** Select **Yes** to return the number of milliseconds that have elapsed since Jan-1, 1970. Select **No** (default) for all other formatting options. This format is used in several web frameworks, for example Oracle ADF.

### 4.4.2.2 Parameterization

Parameterization is a rule of replacing the recorded value in requests with a different dynamic or static value that will be used on test replay. Dynamic values are derived from Variables.

All variable types, including extractors, datasets, data generators and functions are used in parameterization in a very similar way. Any part of the request, including the header, URL, query string and body, can be parameterized in a very similar way as well. The steps for parameterizing a header using an extractor for example, are the same as the steps for parameterizing a body using a function.

Dynamic values replacing recorded values are not limited to just a variable. You can use expressions that combine several variables, text strings, or a combination of thereof. Variables or text strings should appear one after another in the expression without delimiters. During the test run, StresStimulus will resolve variables into strings and concatenate all dynamic and static strings to create an expression that will be inserted in the request.

**Test Integrity Safeguard**

A parameter can operate correctly only if it uses a correct existing variable. StresStimulus has built in verification that preserves the test case integrity.

- If a variable is renamed, all parameters using it will be automatically adjusted to use the new name.
- If a variable is deleted, then all depedent parameters will be deleted as well.

Such validation helps to avoid sending incorrect requests during the test and thus helps to prevent test errors.

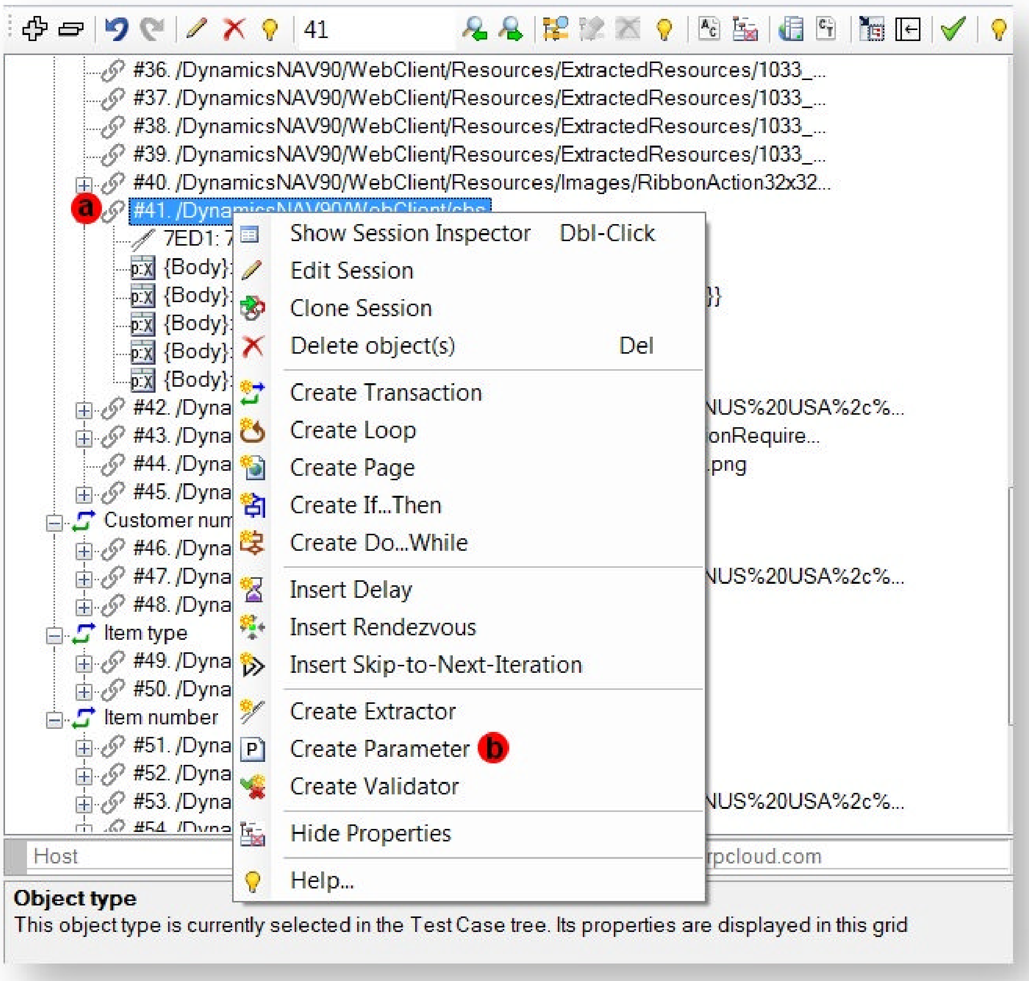
### Creating a Parameter

There are two methods of creating parameters:

- Using the pop-up parameterization window
- Using the parameterization window in the right frame

### Creating parameters using the pop-up parameterization window

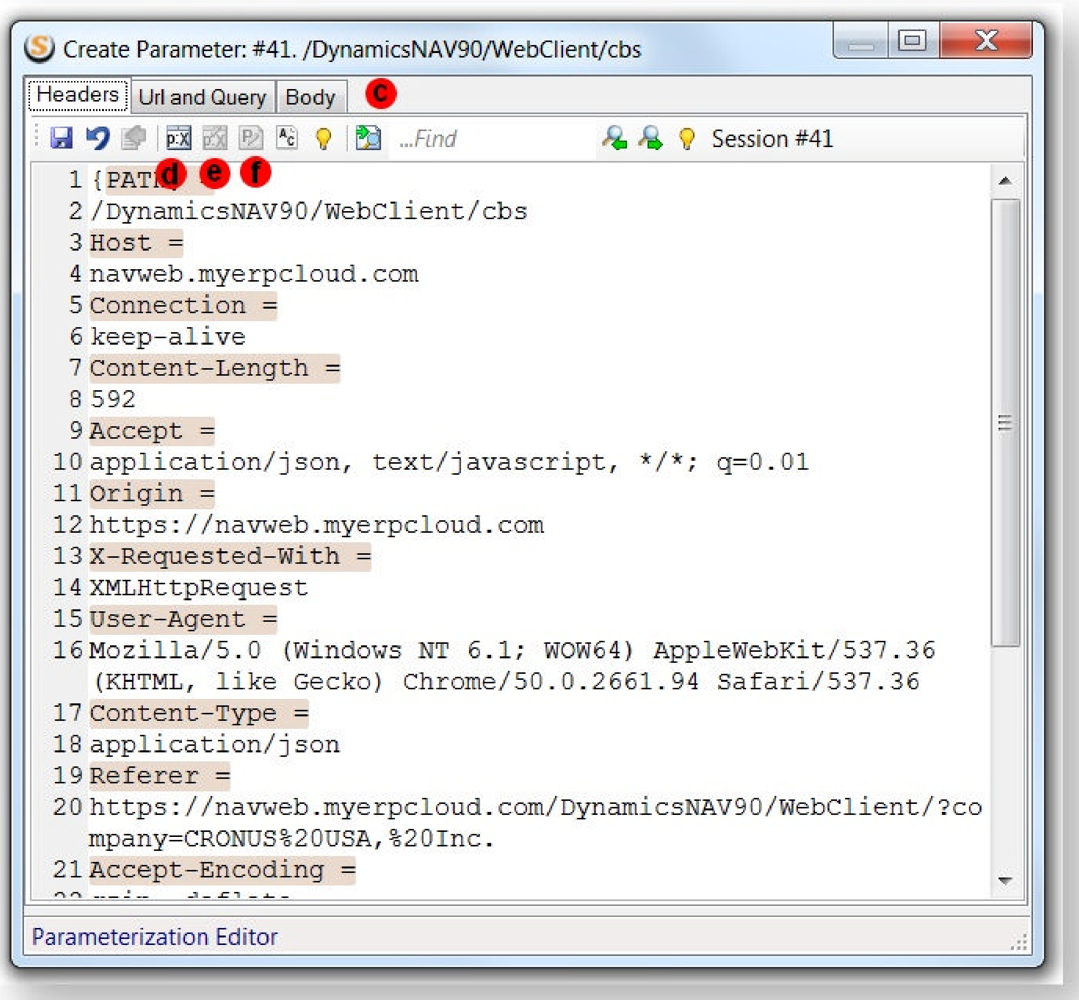
1. Right-click on a request (a) in the Test Case Tree and select **Create Parameter** (b) in the Context menu.



Object type
This object type is currently selected in the Test Case tree. Its properties are displayed in this grid

2. In the appeared pop-up parameterization window, select a tab (c) corresponding to one of three request parts:

- 
    - Request Header
    - Request URL and Query
    - Request Body



```
{PATH} =
/DynamicsNAV90/WebClient/cbs
Host =
navweb.myerpcloud.com
Connection =
keep-alive
Content-Length =
592
Accept =
application/json, text/javascript, */*; q=0.01
Origin =
https://navweb.myerpcloud.com
X-Requested-With =
XMLHttpRequest
User-Agent =
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/50.0.2661.94 Safari/537.36
Content-Type =
application/json
Referer =
https://navweb.myerpcloud.com/DynamicsNAV90/WebClient/?co
mpany=CRONUS%20USA,%20Inc.
Accept-Encoding =
```

**Note:**

- GET requests do not have a Body tab.
- While virtually any header can be parameterized, it is recommended to change only custom headers or values added by your application. Otherwise, modifying standard headers can cause the server to fail the request.

4. Depending on the selected tab and content of the request, StresStimulus will automatically select one of three parameterization controls:

- 
  - Parameterization Grid (d)

  - Parameterization Editor (e)
  - Free Format Request Editor (f)

For example:

- A request part that can be represented as a name/value pair form is shown in Parameterization Grid by default.
- A request part that cannot be represented as a name/value pair form is shown in the **Free Format Request Editor**.

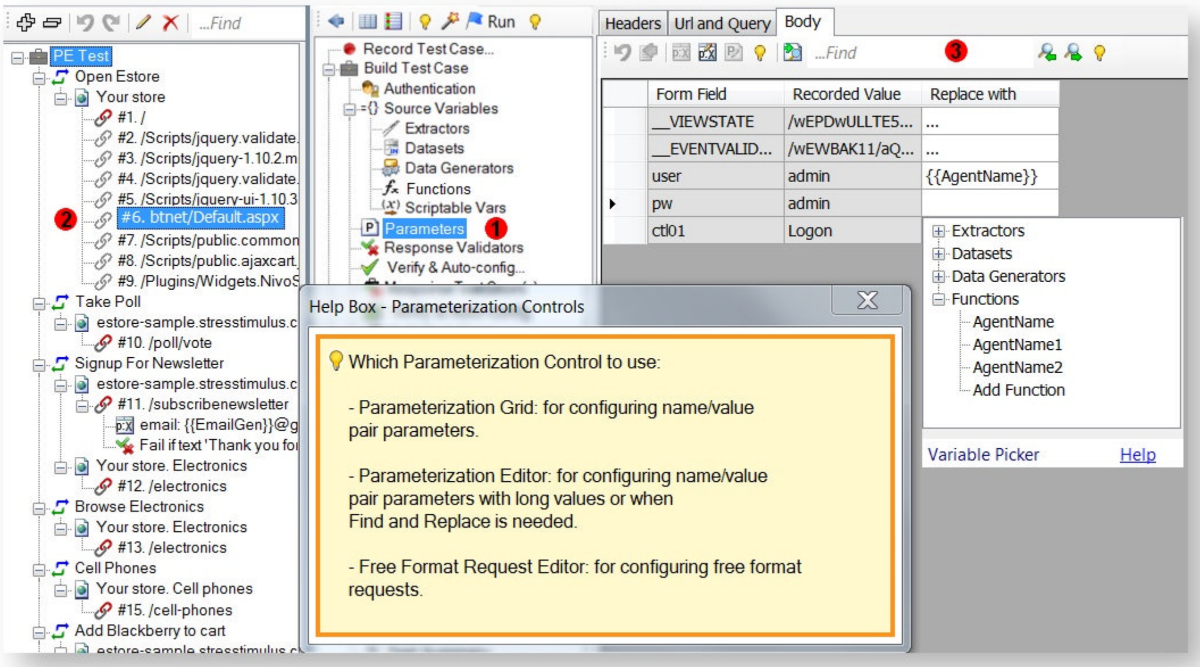
**Note:** Parameterization controls that should not be used are automatically disabled to prevent user errors.

If necessary, select another available control that is more suitable for your situation. For example, you can switch from the Parameterization Grid to the Parameterization Editor if you need to enter long value strings or use the Find and Replace by value feature. To select a different Parameterization Control, click the corresponding button on the toolbar.

The final steps of creating a parameter depend on which parameterization control you will use. The next section describes parameterization controls.

### Creating parameters using the parameterization window in the right frame

1. Select the Parameters node in the Workflow Tree.



| Form Field | Recorded Value | Replace with |
|---|---|---|
| __VIEWSTATE | /wEPDwULLTE5... | ... |
| __EVENTVALID... | /wEWBAK11/aQ... | ... |
| user | admin | {{AgentName}} |
| pw | admin | |
| ctl01 | Logon | |

Help Box - Parameterization Controls

Which Parameterization Control to use:

- Parameterization Grid: for configuring name/value pair parameters.

- Parameterization Editor: for configuring name/value pair parameters with long values or when Find and Replace is needed.

- Free Format Request Editor: for configuring free format requests.

2. Select a request on the Test Case Tree.

3. The parameterization window is displayed on the right frame. Use it similarly to the pop-up parameterization window described above.

## Parameterization Controls

Three parameterization controls are compared in the table below.

| | | **Parameterization Controls** | | |
|---|---|---|---|---|
| **Category** | **Feature** | **Parameterization Grid** | **Parameterization Editor** | **Free Format Request Editor** |
| Toolbar button | | p:x | p:x | 🖉 |
| Functionality | Supports name/value pair format | Yes | Yes | Yes |
| | Preserves name/value pair structure | automatically | automatically | manually |

| | | Parameterization Controls | | |
|---|---|---|---|---|
| | Supports free format (non-name/value pair; e.g. JSON, XML) | No | No | Yes |
| | Supports WCF binary | No | No | Yes |
| | Scope of editing | one parameter at a time | one parameter at a time | entire request part |
| | Edit short string values | Yes | Yes | Yes |
| | Edit long string values | No | Yes | Yes |
| | Find parameters by name and Replace their values | Yes | No | No |
| | Find and Replace value | No | Yes | Yes |
| | Supports Variable Picker | left-click | right-click | right-click |
| In which request part can be used | Request Header | Yes | Yes | No |
| | Request URL and Query | Yes | Yes | Yes |
| | Request Body | Yes | Yes | Yes |

C:\NING\2013\2013-12\11.xlsx

**Tip: Which Parameterization Control to use:**

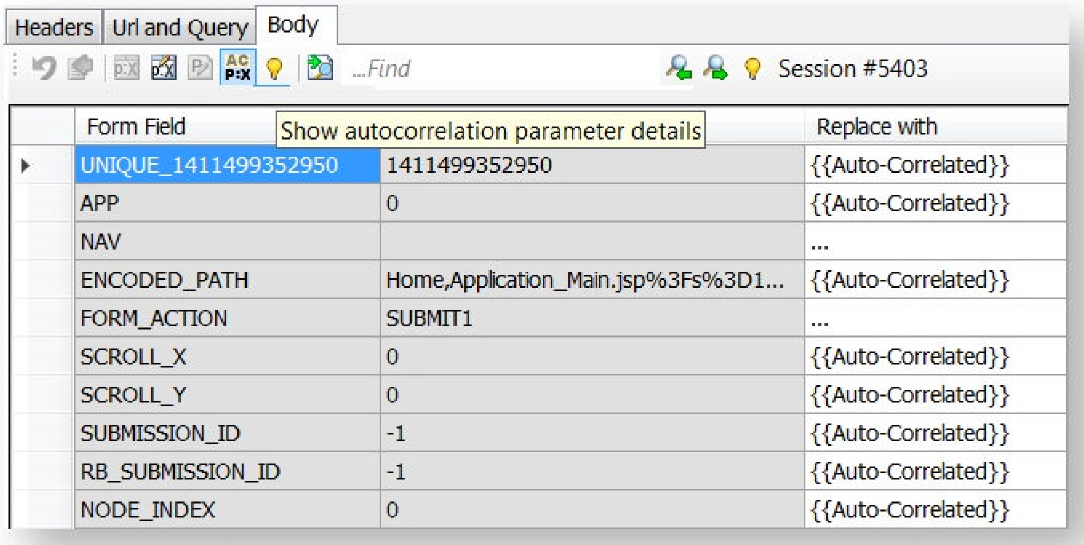
**Parameterization Grid:** for configuring name/value pair parameters.

**Parameterization Editor:** for configuring name/value pair parameters with long values or when Find and Replace is needed.
**Free Format Request Editor:** for configuring free format requests.

## Parameterization Grid

Depending on the selected request part, the first read-only column displays:

- 
    - Header - header names
    - Query Parameter - query string parameter names
    - Form Field - web form parameter names



| Form Field | | Replace with |
|---|---|---|
| UNIQUE_1411499352950 | 1411499352950 | {{Auto-Correlated}} |
| APP | 0 | {{Auto-Correlated}} |
| NAV | | ... |
| ENCODED_PATH | Home,Application_Main.jsp%3Fs%3D1... | {{Auto-Correlated}} |
| FORM_ACTION | SUBMIT1 | ... |
| SCROLL_X | 0 | {{Auto-Correlated}} |
| SCROLL_Y | 0 | {{Auto-Correlated}} |
| SUBMISSION_ID | -1 | {{Auto-Correlated}} |
| RB_SUBMISSION_ID | -1 | {{Auto-Correlated}} |
| NODE_INDEX | 0 | {{Auto-Correlated}} |

Recorded Value: a read-only column that, depending on the selected request part, displays:

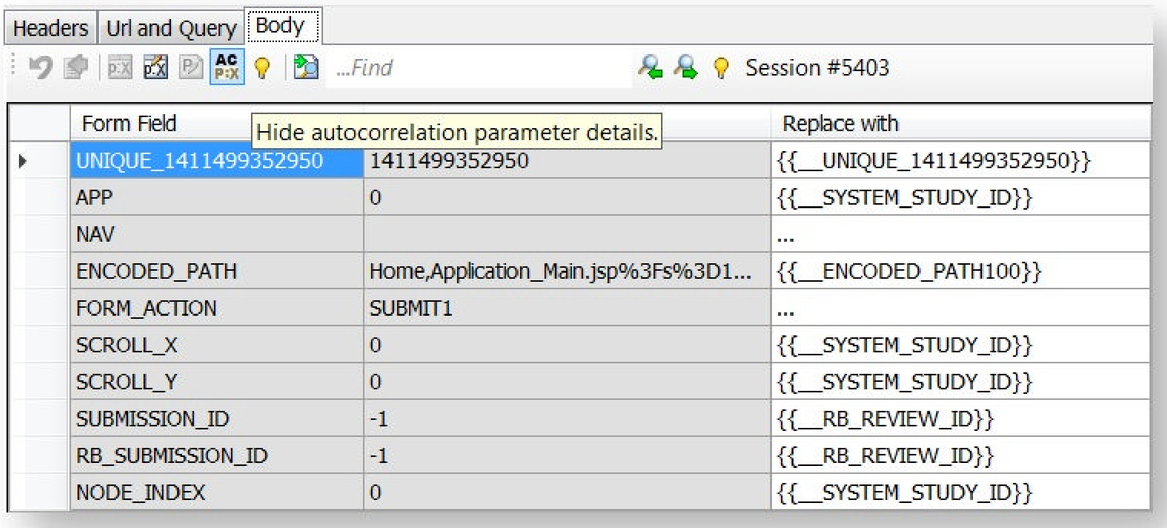
- 
    - recorded header values
    - recorded query string parameter values
    - recorded web form parameter values

Replace with: a column for storing new dynamic or static values replacing the recorded values. After recording a test case, this column contains:

-

- "…" - A placeholder for entering enter a new value. If no value is entered, then the recorded value will be used.
- {{Auto-Correlated}} - A system variable indicating that this field will be autocorrelated.

**Tip:** To display the autocorrelation details, click **Show autocorrelation parameter details** on the toolbar. The extractor names will will be displayed. To switch back to the default view, click **Hide autocorrelation parameter details.**



| Form Field | | Replace with |
|---|---|---|
| UNIQUE_1411499352950 | 1411499352950 | {{__UNIQUE_1411499352950}} |
| APP | 0 | {{__SYSTEM_STUDY_ID}} |
| NAV | | ... |
| ENCODED_PATH | Home,Application_Main.jsp%3Fs%3D1... | {{__ENCODED_PATH100}} |
| FORM_ACTION | SUBMIT1 | ... |
| SCROLL_X | 0 | {{__SYSTEM_STUDY_ID}} |
| SCROLL_Y | 0 | {{__SYSTEM_STUDY_ID}} |
| SUBMISSION_ID | -1 | {{__RB_REVIEW_ID}} |
| RB_SUBMISSION_ID | -1 | {{__RB_REVIEW_ID}} |
| NODE_INDEX | 0 | {{__SYSTEM_STUDY_ID}} |

It is recommended to use the Variable Picker for entering a variable into the **Replace with** column for a parameter. You can also enter an expression containing static text, variables or combination of thereof.

**Creating an expression containing one or several variables and arbitrary text:**

- use variable picker to enter one variable at a time;
- manually cut and paste variables into a text editor ( i.e.Notepad) one after another;
- insert text strings before, between or after the variables, if necessary;
- copy and paste resulting expression back to the **Replace with** column.

**Parameterizing URL.** URL (GET or POST, with or without query string) can be parameterized using a pseudo header called {PATH} located under the Header tab. To do so, right-click in the Value column of the URL row and in the Parameterization Control select desired extractor or data source field.

## Variable Picker

The variable picker control helps to select and enter a variable name in the correct format when creating a parameter. The variable picker is used in all 3 parameterization controls: Parameterization Grid, Parameterization Editor and Free Format Request Editor.

Using Variable Picker

To enter a variable in the Replace With column in the In Parameterization Grid, follow these steps:

1. Invoke Variable Picker, using one of the following methods:

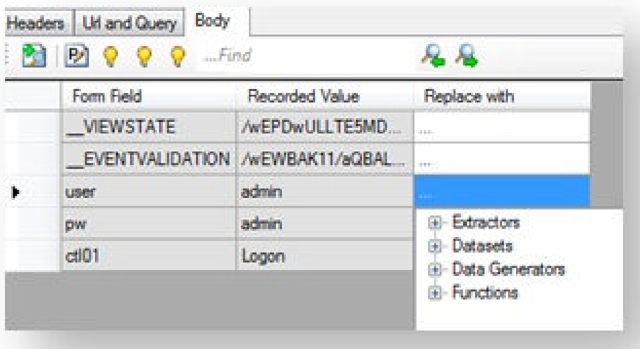
•

  - Click the **...** column.
  - Click over {{Auto-Correlated}} to override autocorrelation with a parameter.
  - If the **Replace with** column already contains a variable or other static text, click where you want to inject a new variable, or select the text that you would like to replace. Then right-click.

2. The variable picker will appear with a list of the existing variables in four categories.

•

  - Extractors
  - Datasets
  - Data Generators
  - Functions



| Form Field | Recorded Value | Replace with |
|---|---|---|
| __VIEWSTATE | /wEPDwULLTE5MD... | ... |
| __EVENTVALIDATION | /wEWBAK11/aQBAL... | ... |
| user | admin | ... |
| pw | admin | |
| ctl01 | Logon | |

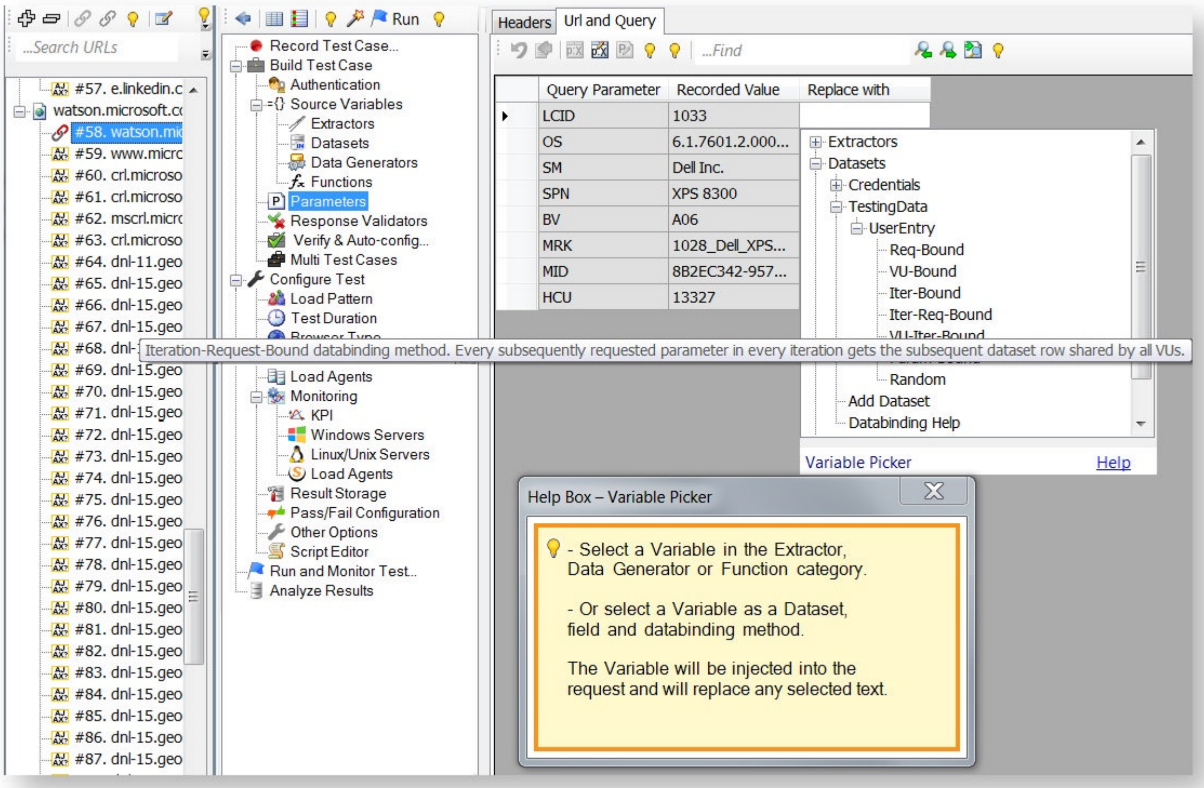
3. Select a variable in the Extractor, Data Generator or Function category, or select a variable as a Dataset, field and databinding method.

4. The Variable Picker will disappear and the variable will be injected into the request parameter.

## Databinding

Databinding is a rule for fetching dataset records used for parameterization requests. The dataset stores multiple sequential records. Every time a dynamic request with a parameter bound to the dataset is generated, a single dataset record is consumed. In some tests it is necessary to use different records every time by repeatedly looping through the records in the dataset. However, in other tests, it is required that a record is repeated for either the same VU, or requests or iterations or combinations of thereof. For example, in login requests a VU has to use the same credentials stored in the dataset on all requests and all iterations. At the same time, to realistically load test a data entry scenario, a VU needs to use different dataset records.

StresStimulus supports seven databinding methods shown below. They appear in the variable picker control under the selected dataset field. To display the description of a data binding method, mouse over it.



| Query Parameter | Recorded Value | Replace with |
|---|---|---|
| LCID | 1033 | |
| OS | 6.1.7601.2.000... | |
| SM | Dell Inc. | |
| SPN | XPS 8300 | |
| BV | A06 | |
| MRK | 1028_Dell_XPS... | |
| MID | 8B2EC342-957... | |
| HCU | 13327 | |

Iteration-Request-Bound databinding method. Every subsequently requested parameter in every iteration gets the subsequent dataset row shared by all VUs.

Help Box – Variable Picker

- Select a Variable in the Extractor, Data Generator or Function category.

- Or select a Variable as a Dataset, field and databinding method.

The Variable will be injected into the request and will replace any selected text.

c:\NING\2013\2013-12\12.pdn

A description of the databinding methods is provided in the table below. The following examples show which dataset record is used in each method: The sample dataset has 20 records. The test case includes 5 parameterized requests bound to this dataset. Two VUs run the test through two iterations. Both VUs traverse through their respective iterations asynchronously as they emulate independent physical users, so there is no synchronization between requests issued by different users.

| Description | Example | | | | |
|---|---|---|---|---|---|
| 1. Request-Bound data-binding method (Req-Bound).<br>Every parameter requested by any VU in any iteration gets a subsequent dataset row. | | VU1 | VU2 | VU1 | VU2 |
| | Req. | Iter. 1 | Iter. 1 | Iter. 2 | Iter. 2 |
| | 1 | 1 | 2 | 10 | 13 |
| | 2 | 3 | 4 | 12 | 14 |
| | 3 | 5 | 7 | 17 | 15 |
| | 4 | 6 | 9 | 18 | 16 |
| | 5 | 8 | 11 | 19 | 20 |
| 2. VU-Bound data-binding method (VU-Bound).<br>Every VU gets a subsequent dataset row used for all its parameters requested in all iterations. | | VU1 | VU2 | VU1 | VU2 |
| | Req | Iter. 1 | Iter. 1 | Iter. 2 | Iter. 2 |
| | 1 | 1 | 2 | 1 | 2 |
| | 2 | 1 | 2 | 1 | 2 |
| | 3 | 1 | 2 | 1 | 2 |
| | 4 | 1 | 2 | 1 | 2 |
| | 5 | 1 | 2 | 1 | 2 |
| 3. Iteration-Bound data-binding method (Iter-Bound).<br>Every iteration gets a subsequent dataset row used by all VUs in all requested parameters. | | VU1 | VU2 | VU1 | VU2 |
| | Req | Iter. 1 | Iter. 1 | Iter. 2 | Iter. 2 |
| | 1 | 1 | 1 | 2 | 2 |
| | 2 | 1 | 1 | 2 | 2 |

| Description | Example | | | | |
|---|---|---|---|---|---|
| | 3 | 1 | 1 | 2 | 2 |
| | 4 | 1 | 1 | 2 | 2 |
| | 5 | 1 | 1 | 2 | 2 |
| 4. Iteration-Request-Bound data-binding method (Iter-Req-Bound).<br>Every subsequently requested parameter in every iteration gets the subsequent dataset row shared by all VUs. | | VU1 | VU2 | VU1 | VU2 |
| | Req | Iter. 1 | Iter. 1 | Iter. 2 | Iter. 2 |
| | 1 | 1 | 1 | 6 | 6 |
| | 2 | 2 | 2 | 7 | 7 |
| | 3 | 3 | 3 | 8 | 8 |
| | 4 | 4 | 4 | 9 | 9 |
| | 5 | 5 | 5 | 10 | 10 |
| 5. VU-Iteration data-binding method (VU-Iter-Bound).<br>Every VU on every iteration gets a unique value. The value is the same for all requests within the iteration. | | VU1 | VU2 | VU1 | VU2 |
| | Req | Iter. 1 | Iter. 1 | Iter. 2 | Iter. 2 |
| | 1 | 1 | 2 | 3 | 4 |
| | 2 | 1 | 2 | 3 | 4 |
| | 3 | 1 | 2 | 3 | 4 |
| | 4 | 1 | 2 | 3 | 4 |
| | 5 | 1 | 2 | 3 | 4 |
| 6. Parameter- Bound data-binding method (Param-Bound).<br>Every requested parameter gets a subsequent dataset row shared by all VUs in all iterations. | | VU1 | VU2 | VU1 | VU2 |
| | Req | Iter. 1 | Iter. 1 | Iter. 2 | Iter. 2 |
| | 1 | 1 | 1 | 1 | 1 |
| | 2 | 2 | 2 | 2 | 2 |
| | 3 | 3 | 3 | 3 | 3 |

| Description | Example | | | | |
|---|---|---|---|---|---|
| | 4 | 4 | 4 | 4 | 4 |
| | 5 | 5 | 5 | 5 | 5 |
| 7. Random data-binding method (Random).<br>Every request parameter gets an arbitrary dataset row. | | | | | |

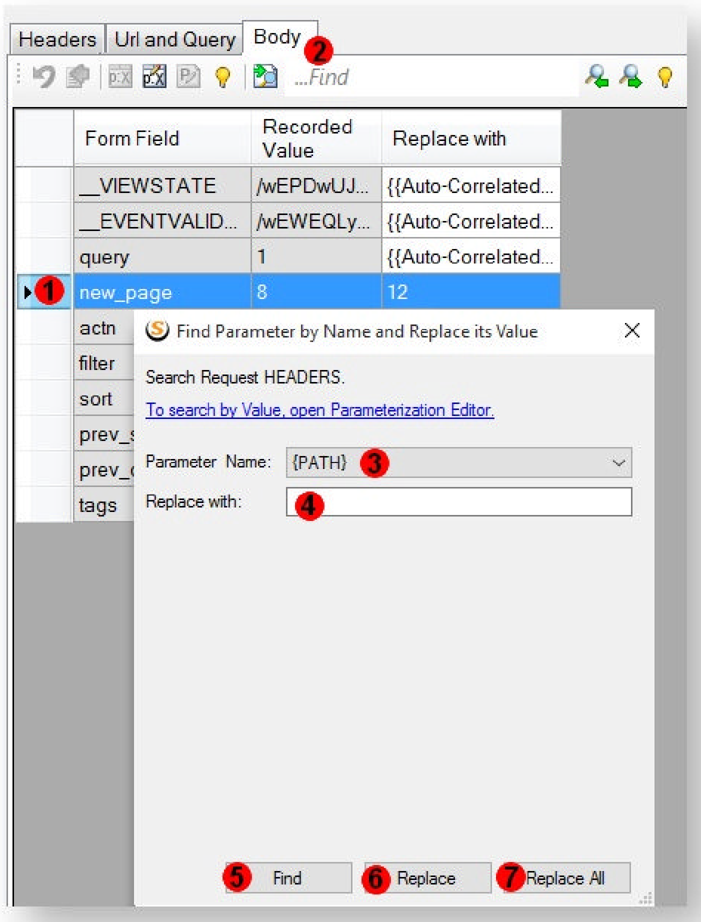
**Note:** If the number of records in a dataset is insufficient, then the records will be reused according to the round-robin algorithm.

### Creating Multiple Parameters from Grid

Sometimes the same variable should be used across multiple GET or POST requests. For example, in the application where the session ID is marshaled as a query parameter, this parameter will be included in every request.

StresStimulus allows to quickly parameterize multiple similar parameters using the Find and Replace feature. To parameterize multiple requests:

1. Select a parameter.



| Form Field | Recorded Value | Replace with |
|---|---|---|
| __VIEWSTATE | /wEPDwUJ... | {{Auto-Correlated... |
| __EVENTVALID... | /wEWEQLy... | {{Auto-Correlated... |
| query | 1 | {{Auto-Correlated... |
| new_page | 8 | 12 |
| actn | | |
| filter | | |
| sort | | |
| prev_s | | |
| prev_c | | |
| tags | | |

Search Request HEADERS.

To search by Value, open Parameterization Editor.

Parameter Name: {PATH}

Replace with:

2. Click **Find and Replace** on the toolbar or hit Ctrl+F to show the Find and Replace dialog.

3. The selected parameter name will be displayed in the Parameter Name box. You can select a different parameter from the drop-down.

•

  - If the Header tab is selected, the drop-down will display all headers.
  - If the URL and Query tab is selected, the drop-down will display all query string parameters in the name/value format.
  - If the Body tab is selected, the drop-down will display all form fields.

4. A current value from the grid's Replace With cell will be displayed in the Replace With text box. Right-click and select a variable from the Variable Picker.

5. Click Find to find the next parameter by name. The search is conducted across all requests in the test case. Only the selected part of the request is searched. For example, if the Body tab is selected, then only form fields will be searched.

6. Click Replace to replace the value of the current parameter with the new value and then find the next parameter with the same name.

7. Click Replace All to bulk replace values in all parameters with the search name.

**Tip:** To selectively parameterize multiple requests, click the Replace or Find buttons depending whether you need to create or skip the parameter.

To parameterize all requests with the searched parameter in one sweep, click Replace All.

## Parameterization Editor

The Parameterization Editor is used for configuring name/value pair parameters. Only the value portion can be changed. It has advantages over the Parameterization Grid in the following cases:

- The values are long strings that are difficult to handle in the grid.
- Multiple similar parameters across many requests are necessary and the Find and Replace commands are more advanced in the editor compared to the grid.

## Using Parameterization Editor

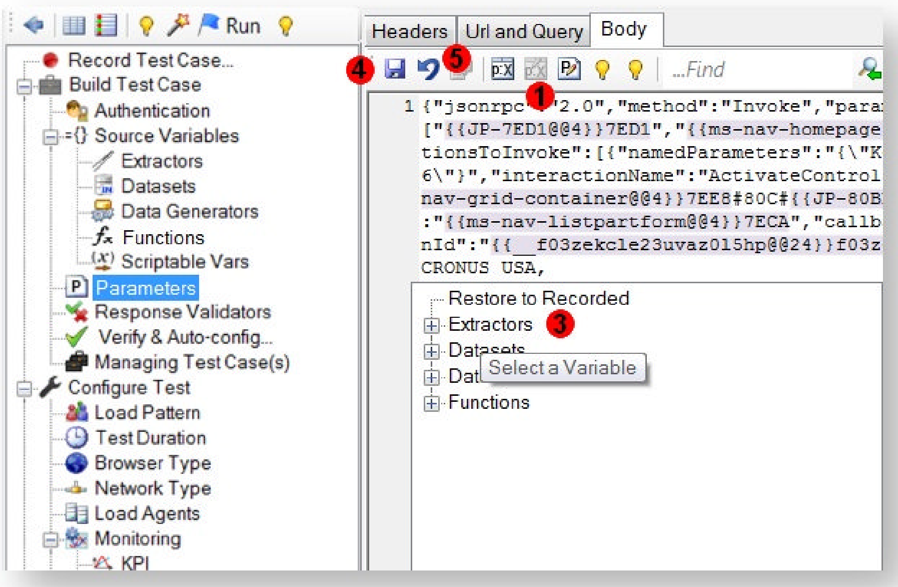
1. To switch to the Parameterization Editor from the Parameterization Grid click the switch button

2. The parameterization editor will appear.

3. To edit data, select text in the **Value Line** that should be replaced and right-click in the Value Line. Then select a **Source Variable** in the Variable Picker.

4. When you finish editing, click **Save**.

5. To discard the changes, click **Undo To The Last Save**.



```
{"jsonrpc":"2.0","method":"Invoke","para
["{{JP-7ED1@@4}}7ED1","{{ms-nav-homepage
tionsToInvoke":[{"namedParameters":"{\"K
6\"}","interactionName":"ActivateControl
nav-grid-container@@4}}7EE8#80C#{{JP-80B
:"{{ms-nav-listpartform@@4}}7ECA","callb
nId":"{{__f03zekcle23uvaz015hp@@24}}f03z
CRONUS USA,
```

## Free Format Request Editor

The Free Format Request Editor is used for parameterizing requests without preserving their name/value structure, if it exists. This is the only parameterization control available for requests that do not have a name/value structure. Here are a few examples of such requests:

- GET request query string does not conform to the format ?<name>=<value>.
- GET requests with Restful URLs.
- POST request body that includes XML or JSON.
- POST request body that includes binary, such as WCF binary or AMF encoding.

Use the Free Format Request Editor similarly to the Parameterization Editor. To edit data, select a value (a) that should be replaced and r ight-click . Then select a source variable in the Variable Picker (b). The color-coded read-only variable control (c) will be inserted. it has the following format (c):

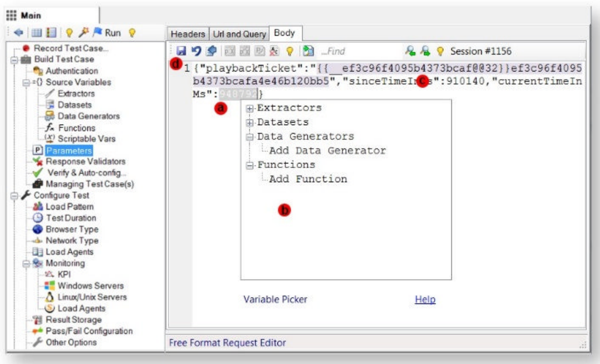
`{{<variable>}}<value>`

where:

`<variable>` is the name of the variable selected in the variable picker;

`<value>` is the recorded value.

When you finish editing, click **Save** (d).



```
1 {"playbackTicket":"{{__ef3c96f4095b4373bcaf@@32}}ef3c96f4095
b4373bcafa4e46b120bb5","sinceTimeIns":910140,"currentTimeIn
Ms":948792}
```

**Concatenating variables**

To concatenate two variables, after inserting the first variable, place the cursor at the end of the first variable control, right-click and select the second variable in the Variable Picker. The second variable control will be inserted.

**Parameterizing WCF binary requests:**

- You can parameterize textual tokens embedded into binary requests, such as AMF requests used in Flash and AIR applications. A token can be surrounded by a binary unreadable message. Just select the recorded token, right-click and in the variable picker, select a variable to replace the token.
- WCF binary requests can be parameterized as easily as regular text encoded requests because StresStimulus decodes them and displays as text. The message "Format: WCF binary" will appear on the toolbar.

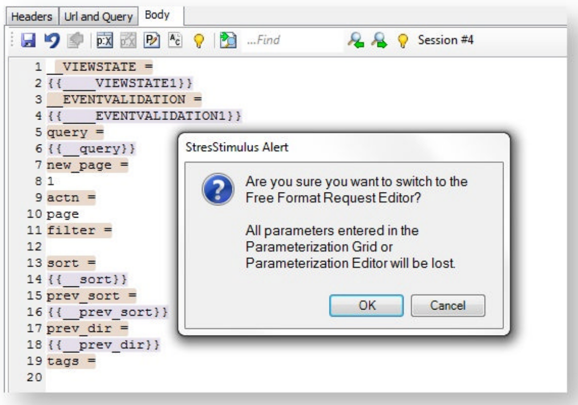
**Parameterizing RESTful requests:**

To parameterize a RESTful GET request (without a query string), replace any sub-string of the URL with a variable. For example, If you recorded the request `http://example.com/resources/item01` , then you can replace the "`item01`" with a dataset value or extractor.

**Changing requests with name/value structure**

- Requests that have a name/value structure can be modified using the Free Format Request Editor. However, all changes made in the Parameterization Grid or Parameterization Editor will be lost. You will be prompted about that when switching to Free Format Request Editor.

- When switching from the Free Format Request Editor back to the Parameterization Editor or Parameterization Grid, all changes will be reverted to its recorded state.



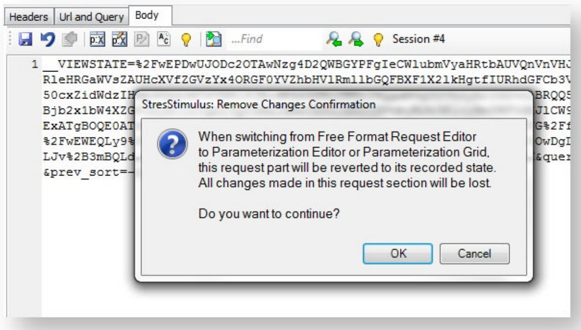
```
__VIEWSTATE =
{{___VIEWSTATE1}}
__EVENTVALIDATION =
{{___EVENTVALIDATION1}}
query =
{{__query}}
new_page =
1
actn =
page
filter =

sort =
{{__sort}}
prev_sort =
{{__prev_sort}}
prev_dir =
{{__prev_dir}}
tags =
```

StresStimulus Alert

Are you sure you want to switch to the Free Format Request Editor?

All parameters entered in the Parameterization Grid or Parameterization Editor will be lost.



```
__VIEWSTATE=%2FwEPDwUJODc2OTAwNzg4D2QWBGYPFgIeCWlubmVyaHRtbAUVQnVnVHJ
RleHRGaWVsZAUHcXVfZGVzYx4ORGF0YVZhbHVlRmllbGQFBXF1X2lkHgtfIURhdGFCb3V
```

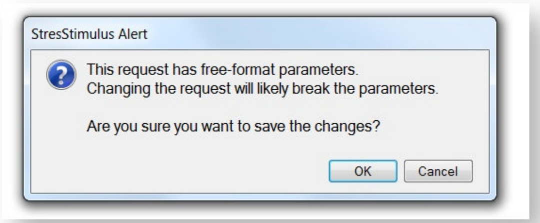
StresStimulus: Remove Changes Confirmation

When switching from Free Format Request Editor to Parameterization Editor or Parameterization Grid, this request part will be reverted to its recorded state. All changes made in this request section will be lost.

Do you want to continue?

**Note**: After creating at least one free-format parameter in the response, you can no longer safely edit it. If you change the response in the session inspector and click **Save**, you will receive a warning.

If changes you made in the response are positioned after the free format parameter, then such changes will not break the parameter.



StresStimulus Alert

This request has free-format parameters. Changing the request will likely break the parameters.

Are you sure you want to save the changes?

## Creating Multiple Parameters at Once

Sometimes the same variable should be used across multiple GET or POST requests. For example, in the application where the session ID is marshaled as a query parameter, this parameter will be included in every request.

In some applications, a parameter's value submitted multiple times can have a different name in different requests.

If such parameter has to be parameterized, searching by the parameter name will not find all instances of the parameter. In this case, instead of the Parameterization Grid, use the Parameterization Editor.

StresStimulus allows to quickly parameterize multiple similar parameters and search by name and value using the Find and Replace feature in Parameterization Editor and Free Format Request Editor.

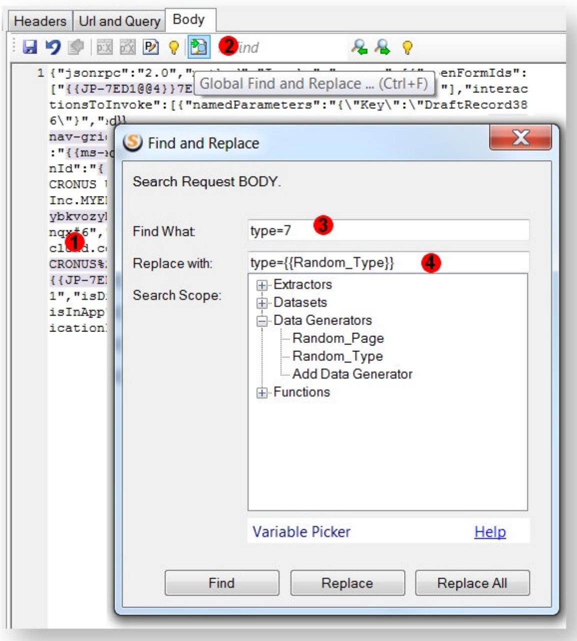
To parameterize multiple requests:

1. Select a value to parameterize

2. Click Global Find and Replace

3. Searched value will appear in the Find What box

4. Right-click in the Replace with box and in the appeared Variable Picker, select a variable. Alternatively, you can type your own value, wildcard or regular expression.

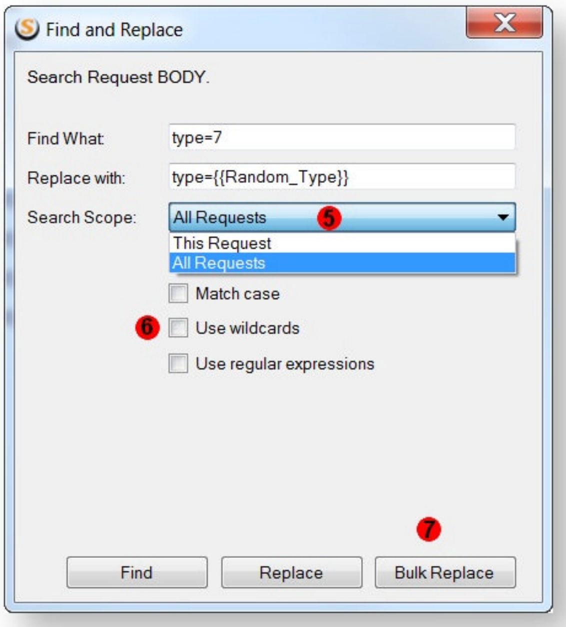


Search Request BODY.

Find What: type=7

Replace with: type={{Random_Type}}

Search Scope:
- Extractors
- Datasets
- Data Generators
  - Random_Page
  - Random_Type
  - Add Data Generator
- Functions

| | |
|---|---|
| 5. Select Search Scope as This Request or All Requests<br>6. Check other applicable options<br>7. Click Bulk Replace |  |

Find and Replace

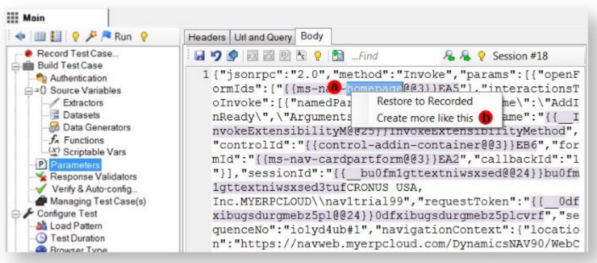
Search Request BODY.

Find What: type=7

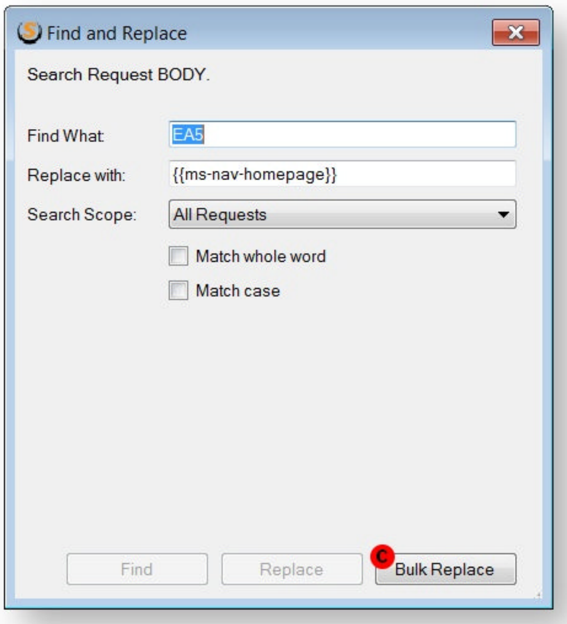
Replace with: type={{Random_Type}}

Search Scope: All Requests

You can also create multiple parameters from an existing parameter. Right click on an existing parameter (a) and select **Create More Like This** (b). The find and replace dialog will appear with the find and replace criteria pre-filled. Click the **Bulk Replace** button (c) to create all possible similar parameters.



```
{"jsonrpc":"2.0","method":"Invoke","params":[{"openF
ormIds":["{{ms-nav-homepage@@3}}EA5"],"interactionsT
oInvoke":[{"namedPar                    me\":\"AddI
nReady\",\"Arguments                    ame":"{{__I
nvokeExtensibilityM@@25}}InvokeExtensibilityMethod",
"controlId":"{{control-addin-container@@3}}EB6","for
mId":"{{ms-nav-cardpartform@@3}}EA2","callbackId":"1
"}],"sessionId":"{{__bu0fm1gttextniwsxsed@@24}}bu0fm
1gttextniwsxsed3tufCRONUS USA,
Inc.MYERPCLOUD\\nav1trial99","requestToken":"{{__0df
xibugsdurgmebz5pl@@24}}0dfxibugsdurgmebz5plcvrf","se
quenceNo":"iolyd4ub#1","navigationContext":{"locatio
n":"https://navweb.myerpcloud.com/DynamicsNAV90/WebC
```



Find and Replace

Search Request BODY.

Find What: EA5

Replace with: {{ms-nav-homepage}}

Search Scope: All Requests

### Regular Expression and Wildcards

When the regular expression or wildcard option is selected, the entire value is replaced unless a match group is specified, in which case only the first group is replaced. The examples below show the result of replacement with variable: {{MyVariable}}.

### Regular Expression Examples

| Expression | Match | Result |
|---|---|---|
| ([0-9A-F]{8}\-{0,1}[0-9A-F]{4}\-{0,1}[0-9A-F]{4}\-{0,1}[0-9A-F]{4}\-{0,1}[0-9A-F]{12}) | b558b213-8501-894b-b0b3-39c7a4f2e840 | {{MyVariable}} |
| <id>(\d+)</id> | <id>123456789</id> | <id>{{MyVariable}}</id> |

### Wildcard Examples

| Expression | Match | Result |
|---|---|---|
| {id : "(*)"} | {id : "jd123"} | {id : "{{MyVariable}}"} |
| 5_char_secret=(?????) | 5_char_secret=5d3hp | 5_char_secret={{MyVariable}} |

## 4.4.3 Automatic Parameterization

One of the main challenges in creating a load testing script is to identify dynamic variables. Your knowledge about the tested website design may be insufficient to quickly locate and manually configure all dynamic parameters. Your test case can be configured much faster using the automatic parameterization method consisting of three steps:

**1. Verify.** This step generates a list of errors and warnings displayed in the Session Verification Tree.

**2. Run Parameter Finder.** This step finds requests that likely need to be parameterized and gives recommendations about missing extractors and parameters.

**3. Create extractors and parameters.** This step can be performed in one or two ways

**a. Semi-automatically:** Run the **Parameter Tool** to auto-configure an extractor with related parameters.

**b. Automatically:** Run the AutoConfigurator to create all extractors with related parameters.

#### 4.4.3.1 AutoCorrelation

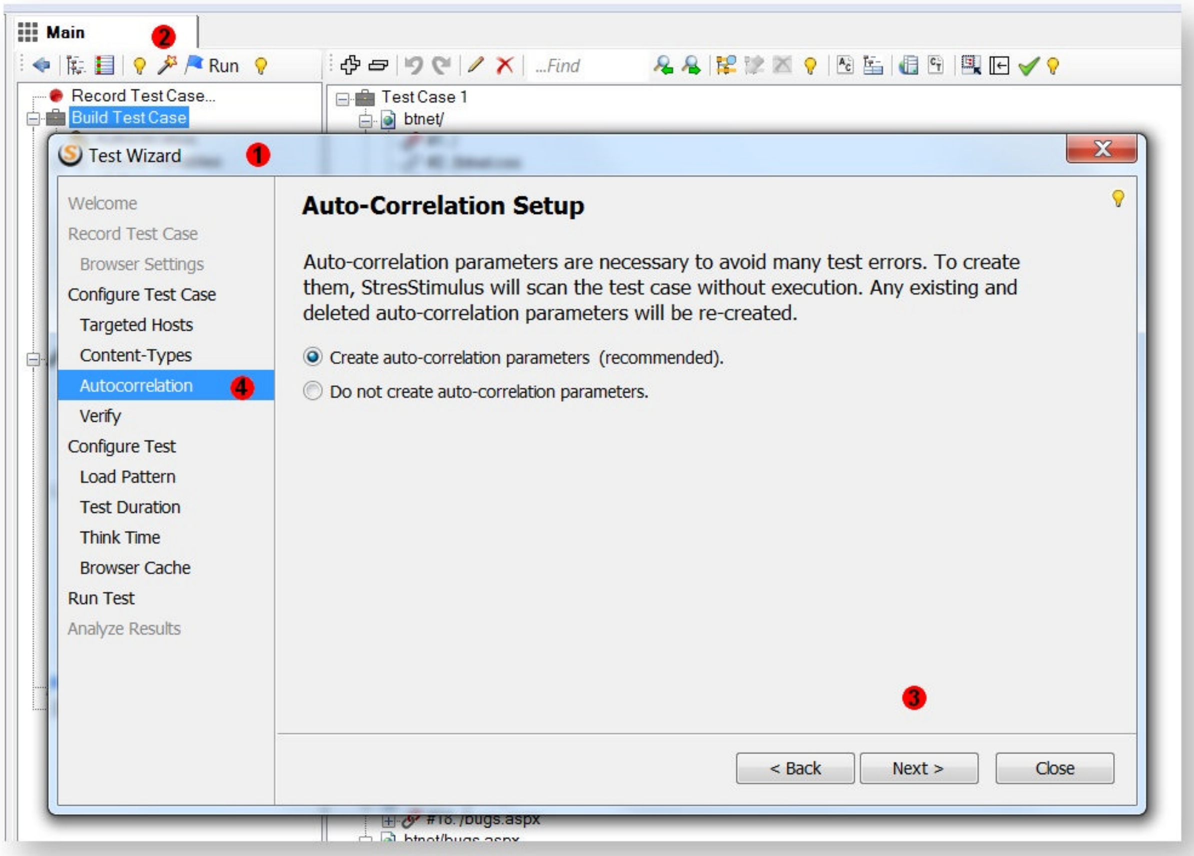
In StresStimulus, the majority of dynamic parameters are found and correlated automatically. Autocorrelation finds hidden dynamic parameters in the query string, body, header or restless URL in all web application platforms, including binary or XML messages. In simple and medium tests cases, typically the entire parameterization is configured automatically.

Autocorrelation runs automatically in the test wizard (1) after the test case recording. The autocorrelation progress bar displays the number of create parameters.

It may be necessary to rerun autocorrelation manually after the following changes in the test case:

- sessions are deleted, repositioned or more sessions are added
- the property **MIME Types requested sequentially** in Configure Test --> Other Options was changed (for more see Request Concurrency

To re-run autocorrelation manually, click **Test Wizard** (2) on the **Workflow Tree** toolbar,then click Next (3) one or several times until you execute the **Autocorrelation** step (4).



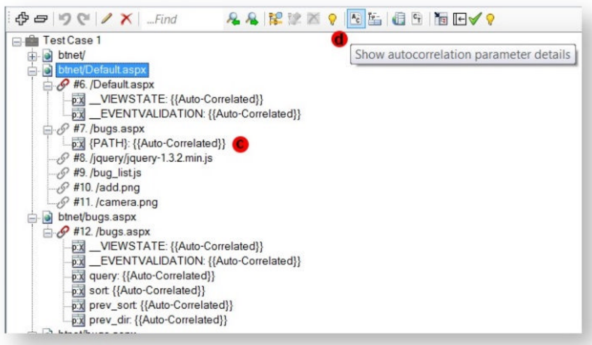
**Auto-Correlation Setup**

Auto-correlation parameters are necessary to avoid many test errors. To create them, StresStimulus will scan the test case without execution. Any existing and deleted auto-correlation parameters will be re-created.

[✓ Create auto-correlation parameters (recommended).]
Do not create auto-correlation parameters.

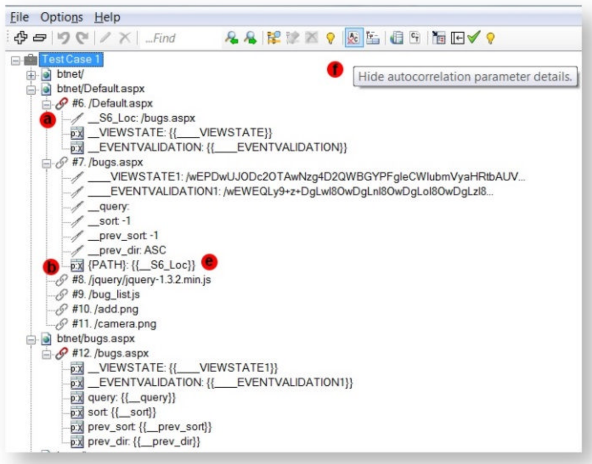
**Autocorrelation transparency.** Autocorrelation script is available for review. An autocorrelation parameter consists of an autocorrelation extractor (a) and a parameter (b). By default, the extractor is hidden and the parameter is marked as {{Autocorrelated}} (c)

To display the hidden details, click **Show autocorrelation parameters details** (d) on the test case tree toolbar. The extractors will un-hide and their names will be displayed next to the corresponding parameters (e). To switch back to the default view, click **Hide autocorrelation parameters details** (f).



```
Test Case 1
  btnet/
  btnet/Default.aspx
    #6. /Default.aspx
      __VIEWSTATE: {{Auto-Correlated}}
      __EVENTVALIDATION: {{Auto-Correlated}}
    #7. /bugs.aspx
      {PATH}: {{Auto-Correlated}}
    #8. /jquery/jquery-1.3.2.min.js
    #9. /bug_list.js
    #10. /add.png
    #11. /camera.png
  btnet/bugs.aspx
    #12. /bugs.aspx
      __VIEWSTATE: {{Auto-Correlated}}
      __EVENTVALIDATION: {{Auto-Correlated}}
      query: {{Auto-Correlated}}
      sort: {{Auto-Correlated}}
      prev_sort: {{Auto-Correlated}}
      prev_dir: {{Auto-Correlated}}
```

Show autocorrelation parameter details



```
Test Case 1
  btnet/
  btnet/Default.aspx
    #6. /Default.aspx
      __S6_Loc: /bugs.aspx
      __VIEWSTATE: {{____VIEWSTATE}}
      __EVENTVALIDATION: {{____EVENTVALIDATION}}
    #7. /bugs.aspx
      ____VIEWSTATE1: /wEPDwUJODc2OTAwNzg4D2QWBGYPFgleCWlubmVyaHRtbAUV...
      ____EVENTVALIDATION1: /wEWEQLy9+z+DgLwl8OwDgLnl8OwDgLol8OwDgLzl8...
      __query:
      __sort: -1
      __prev_sort: -1
      __prev_dir: ASC
      {PATH}: {{__S6_Loc}}
    #8. /jquery/jquery-1.3.2.min.js
    #9. /bug_list.js
    #10. /add.png
    #11. /camera.png
  btnet/bugs.aspx
    #12. /bugs.aspx
      __VIEWSTATE: {{____VIEWSTATE1}}
      __EVENTVALIDATION: {{____EVENTVALIDATION1}}
      query: {{__query}}
      sort: {{__sort}}
      prev_sort: {{__prev_sort}}
      prev_dir: {{__prev_dir}}
```

Hide autocorrelation parameter details.

**Note:** Autocorrelated extractors and parameters have names that begins with two underscores.

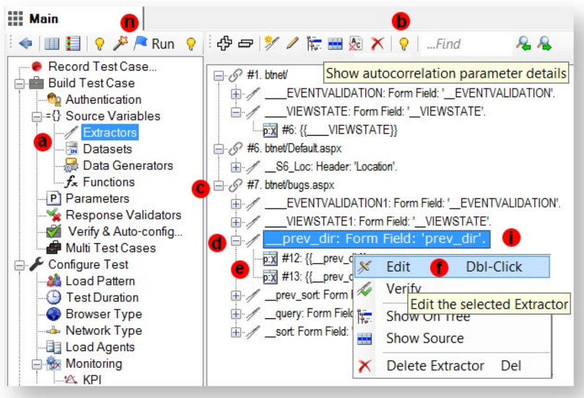
## Deleting & Restoring AutoCorrelation Rules

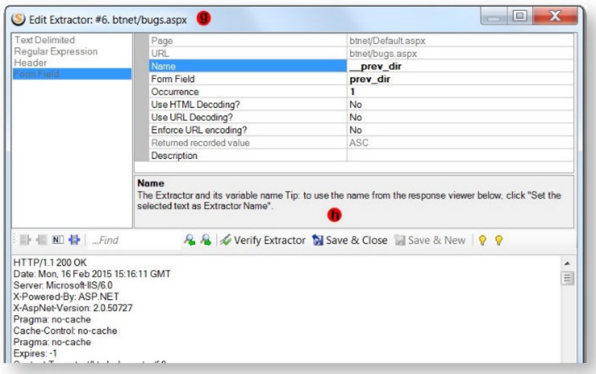
AutoCorrelation creates a set of rules, each of which includes an extractor and one or several parameters necessary to keep your test from application errors. You can review, change or delete these rules as follows:

**Warning:** Modifying and Deleting AutoCorrelation rules is an advanced script development technique. If not performed properly, it can cause test errors.

1. Navigate to the Extractors node (a) in the Navigation Tree.
2. By default, only user created objects a are displayed and autocorrelation objects are hidden. To display then, click **Show autocorrelation parameter details** (b) on the toolbar. The object tree will display the following hierarchy of objects: Requests (c) -> Extractors (d) -> Parameters (e). You can expand / collapse the tree to the detail level you need.
3. To review the selected extractor, right-click and select **Edit** (f), or click **Edit** on the toolbar. The Extractor editor (g) will appear. When complete reviewing the extractor, close the editor.
4. To edit the extractor, make necessary changes in the extractor editor and click **Save** (h). For more about the extractors, check the Extractors section.
5. To review a parameter that uses the extractor:
    a. Double-click a parameter (i) subordinate to the extractor. Test case tree on the left will highlight the parameter (j).
    b. Navigate to the **Parameter** node (k) in the Navigation Tree.
    c. Navigate through tabs (l) Header, URL and Query, or Body to find in which part of the request this parameter is located.
    d. The parameter definition will be displayed in on of three parameterization controls:
        i. Parameterization Grid (m)
        ii. Parameterization Editor
        iii. Free Format Request Editor

To edit or delete a parameter, follow these steps depending on the parameterization



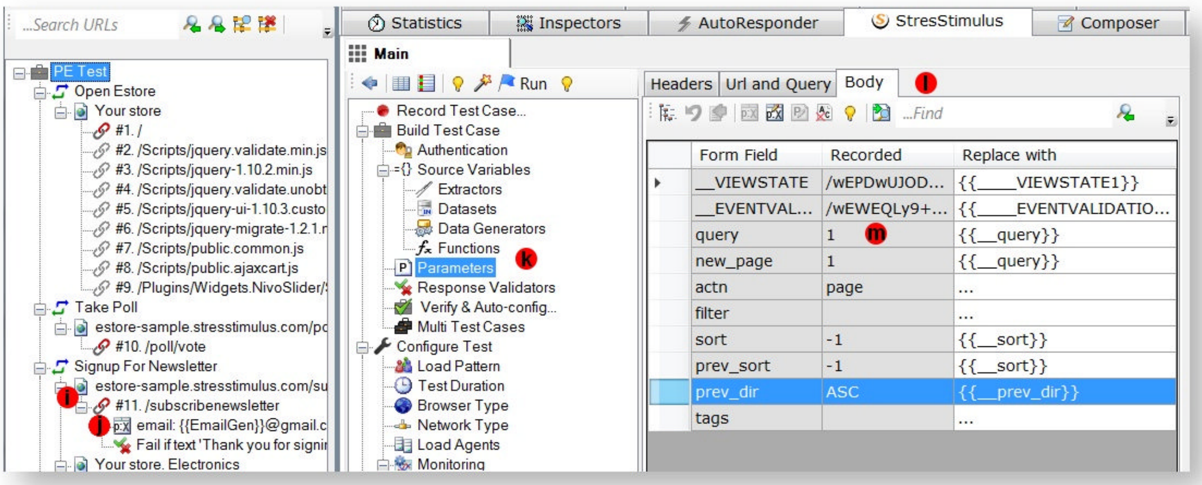

Edit Extractor: #6. btnet/bugs.aspx

| | |
|---|---|
| Page | btnet/Default.aspx |
| URL | btnet/bugs.aspx |
| Name | __prev_dir |
| Form Field | prev_dir |
| Occurrence | 1 |
| Use HTML Decoding? | No |
| Use URL Decoding? | No |
| Enforce URL encoding? | No |
| Returned recorded value | ASC |
| Description | |

**Name**
The Extractor and its variable name Tip: to use the name from the response viewer below, click "Set the selected text as Extractor Name".

```
HTTP/1.1 200 OK
Date: Mon, 16 Feb 2015 15:16:11 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
Pragma: no-cache
Cache-Control: no-cache
Pragma: no-cache
Expires: -1
```

| | |
|---|---|
| control used.<br>6. To restore autocorrelation, click the **Test Wizard** (n) button on the Navigation Tree toolbar and follow the step off re-creating autocorrelation rules. | |



| Form Field | Recorded | Replace with |
|---|---|---|
| __VIEWSTATE | /wEPDwUJOD... | {{____VIEWSTATE1}} |
| __EVENTVAL... | /wEWEQLy9+... | {{____EVENTVALIDATIO... |
| query | 1 | {{__query}} |
| new_page | 1 | {{__query}} |
| actn | page | ... |
| filter | | ... |
| sort | -1 | {{__sort}} |
| prev_sort | -1 | {{__sort}} |
| prev_dir | ASC | {{__prev_dir}} |
| tags | | ... |

### 4.4.3.2 Verifying Test Case

After recording a test case, it's highly recommended to verify it. If you run a test without verification, you may encounter issues related to incorrect or incomplete test configuration. The purpose of verification is to discover such issues.

This information should be used to add some configuration settings that will fix verification errors or warnings. After all verification errors and warnings are resolved, all the test errors are likely related to your website's inability to properly handle multiple users.

During verification, StresStimulus replays the recorder test case once with one virtual user in debug mode. Reissued sessions are automatically compared with the corresponding recorded sessions, and the recorded and replayed server responses are analyzed. After the verification is completed, errors, warnings, configuration recommendations and other diagnostic are displayed.

**Note:** Some Configure Test settings, such as browser type and network type are not simulated during verify.

To start the verify process:

1. Click **Verify & Auto-Config** (a) in the workflow tree.

2. A pop-up dialog **Verify the Test Case** (b) will appear.

3. Sometimes it is desirable to verify the test case with a different VU. For example, if a test case uses a dataset to parameterize an authentication form, then you might want to verify a few times under different credentials each time. To simulate a different VU, set the numeric box (c) to the desired VU.

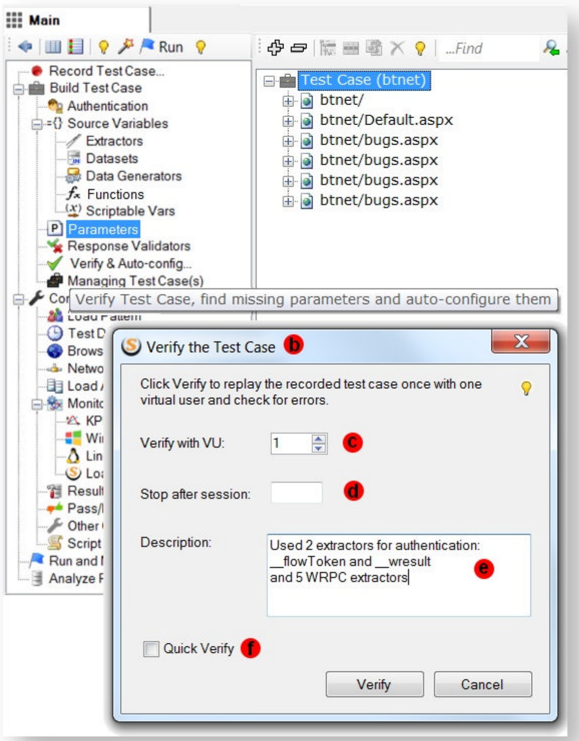
4. Optionally, you can specify a session number, after which Verify will stop, to save time. For example, if the error / warnings occurred in the sessions 20, 22, 40, 42, 60 … and you just addressed session 20, then you can run Verify to session 40 or 42. To do so, enter the session number in the **Stop after session** box (d).

5. You can add a description for each verify run (e) so that you can keep track of the changes you make to your test case between verifys.

6.During verify, the test case is replayed at a slower pace to give user a chance to preview succession of web pages appeared in the web view. To run verify at a faster pace without the page preview, check **Quick Verify** box (f).

7. Click **Verify**. The session grid will display sessions as they get issued in real time.

**Tip:** If during the test case verification some errors or warnings are



Verify Test Case, find missing parameters and auto-configure them

Verify the Test Case

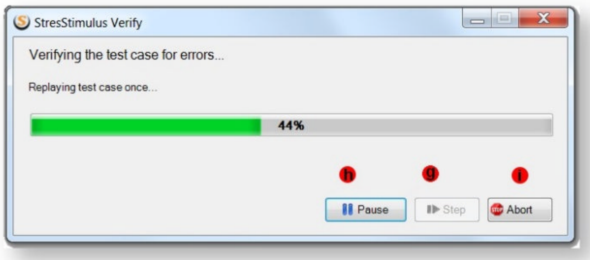
Click Verify to replay the recorded test case once with one virtual user and check for errors.

Verify with VU: 1

Stop after session:

Description: Used 2 extractors for authentication: __flowToken and __wresult and 5 WRPC extractors

☐ Quick Verify



Verifying the test case for errors...

Replaying test case once...

discovered, they should be reviewed one by one in the order of appearance from top to bottom. Depending on the situation, use the appropriate technique to address them. For example, create a missing parameter, delete a request that should be excluded from the test or ignore warnings that are irrelevant to the test's execution. After that, re-run Verify and move to the next error warning resolution. Since you should only focus on the next 1-2 errors/ warnings, there is no need to Verify the entire test.

8. During the test replay, the browser object displays the sequence of the web pages as they get accessed. This helps to quickly determine whether the test case is replayed correctly. For example, if the pages in the browser object repeatedly display the login screen or error messages, it is an indication of authentication or session integrity issues.

- In the standalone version, webpage view Is located in the right pane.
- In the add-on version, Fiddler automatically switches to the Inspectors -> web view tab

9. The following options are available to control the verification process:

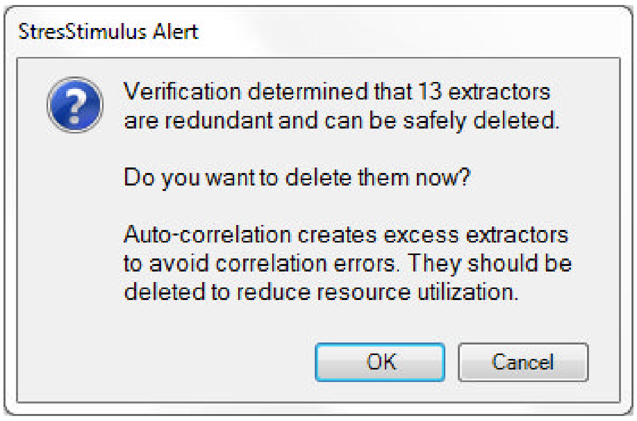
- Click **Pause** (h) to suspend issuing requests. Click **Resume** to continue verification
- Click **Step** (g) after pausing to send a single next-in-the-queue request for every VU. This option is used to further debug

the test case.

- Click **Abort** (i) to stop verification

**Info**: During verify, timeout mechanism is disabled. StresStimulus will wait for all responses to come back, and the wait time is not limited by the timeout of the request.

9. If verification determines that there are too many redundant extractors which can be safely deleted, it will prompt you to confirm their deletion. Autocorrelation sometimes creates excess extractors to avoid correlation errors. They should be deleted to reduce resource utilization.


StresStimulus Alert

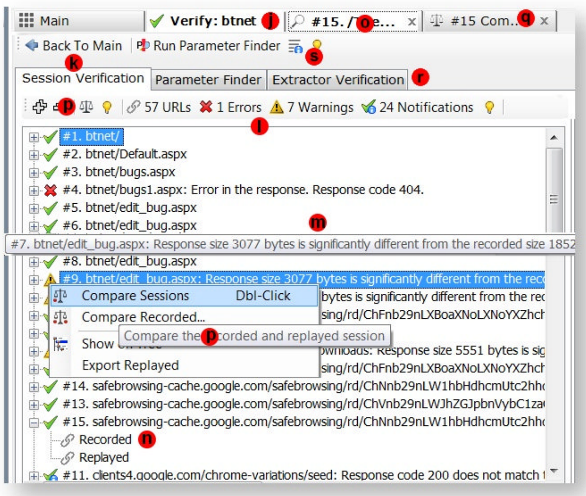
Verification determined that 13 extractors are redundant and can be safely deleted.

Do you want to delete them now?

Auto-correlation creates excess extractors to avoid correlation errors. They should be deleted to reduce resource utilization.

10. After Verify is completed, a new tab (j) will appear. It displays **Session Verification** tab (k) with tree showing the outcome of verification for each sessions and the test case overall. Next to every request, there is a status image describing the outcome. One of 4 statuses can be assigned to a session:

| | Status | Description |
|---|---|---|
| ✓ | Pass | Session completed normally |
| ✗ | Error | Sessions with errors related to the test configuration. Configuration-related errors indicate that some test configuration should be adjusted. Most often such errors can be fixed by adding missing parameters. Try to find and create such parameters using Parameter Finder. |



#1. btnet/
#2. btnet/Default.aspx
#3. btnet/bugs.aspx
#4. btnet/bugs1.aspx: Error in the response. Response code 404.
#5. btnet/edit_bug.aspx
#6. btnet/edit_bug.aspx
#7. btnet/edit_bug.aspx: Response size 3077 bytes is significantly different from the recorded size 1852
#8. btnet/edit_bug.aspx
#9. btnet/edit_bug.aspx: Response size 3077 bytes is significantly different from the rec
bytes is significantly different from the rec
sing/rd/ChFnb29nLXBoaXNoLXNoYXZhch
wnloads: Response size 5551 bytes is sig
sing/rd/ChFnb29nLXBoaXNoLXNoYXZhch
#14. safebrowsing-cache.google.com/safebrowsing/rd/ChNnb29nLW1hbHdhcmUtc2hh
#13. safebrowsing-cache.google.com/safebrowsing/rd/ChVnb29nLWJhZGJpbnVybC1za
#15. safebrowsing-cache.google.com/safebrowsing/rd/ChNnb29nLW1hbHdhcmUtc2hh
Recorded
Replayed
#11. clients4.google.com/chrome-variations/seed: Response code 200 does not match t

| | | |
|---|---|---|
| ⚠ | Warning | Sessions with issues that may or may not be related to the test configuration. Try using Parameter Finder to eliminate or reduce number of warnings. After that, inspect remaining warnings to make sure that they are not caused by missing settings in the test configuration. |
| ✓ | Notification | Sessions with issues unrelated to the test configuration. Notifications can be rectified by refining your application. For example, broken links of missing images will causes 404 HTTP errors. Such errors can be fixed by making adjustments in the tested application. |

11. To display a subset of sessions with the same status, click one of the filtering buttons (l) on the toolbar session verification: Errors, Warnings or Notifications. To display all sessions, click URLs.

12. To display a tooltip with a specific error or warning, mouse-over a session (m).

13. To view session content, double-click the **Recorded** or **Replayed** node (n) and **Session Inspector** (o) will appear.

14. Right-click a session to display the context menu. To compare recorded and replayed sessions, click **Compare** (p) in the context menu or on the toolbar, and the **Compare Sessions Inspector** (q) will appear. Session Inspectors are described in the next section.

15. The **Verify** command also generates an **Extractor Verification Tree** (r). In a separate tab, it displays extractor values,

| | |
|---|---|
| errors (if the extractors cannot be validated) or warnings (if the extractors are not used).<br>16. Click **Verify Description** (s) to see the test case you verified, the time or the description. You can also modify the description here. | |

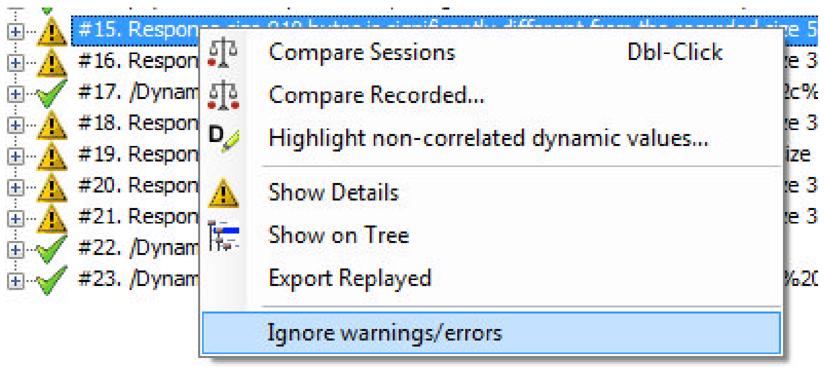
**Tip:** You can export a session replayed during verify and save it as an .saz file. To do so, right-click anywhere in the verification tree, then select **Export Replayed.**

### Verification Errors and Warnings

StresStimulus uses a heuristic algorithm to mark verification errors and warnings, and sometimes false positives are identified.

### Suppressing errors and warnings

In order to ignore these errors and warnings from future verification, right-click and select **Ignore warnings/errors**.



To undo this option, right-click and select **Undo Ignore warnings/errors**.

**Note:** Sessions that have errors suppressed, will not show up on the test error report.

### Errors and Warnings Details

A list of diagnostic messages is provided below.

| Status Type | Diagnostic Message |
|---|---|
| Error | Error in the response. Response code is 4xx or 5xx. |
| Error | The response was not received |
| Warning | Replayed request was not authenticated. Check Test authentication settings. If subsequent requests are authenticated, then this may be normal. |
| Warning | Response code does not match the recorded code. |
| Warning | Redirect to a different URL. |
| Warning | First request in the test case contains ASP.NET SessionId cookie. The beginning of the test case may be missing. |
| Warning | Response size is significantly different from the recorded size. |
| Notification | Error in the response code during recording. This error is unrelated to load testing. |
| Notification | The resource was cached during the Test Case recording (response code 304) and not cached during the replay (response code 200). |
| Notification | The resource was cached during the Test Case recording (response code 304) and not cached during the replay (response code 200). |
| Notification | The resource was not cached during the Test Case recording (response code 200) and cached during the replay (response code 304). |

#### 4.4.3.3 Resolving Errors and Warnings

If all requests pass verification, and the Verification Tree displays all green checkboxes, then your test case is configured and you can move on to the next step of configuring the load and running the test.

However, sometimes the session verification tree displays diagnostic messages. This means that some issues were discovered when a virtual user replayed the test case.

Different issues can impact test accuracy with various degrees, from an insignificant impact to total test failure. For example, if a virtual user sends a request to a static image without caching headers, the test's accuracy will be minimally affected. However, if the authentication process fails, then all remaining requests will not be authenticated and the test will be totally incorrect.

In order to address verification issues, try to find what is causing them, fix the problem and then verify the test case again. A solution to the problem depends on the type of message that you receive. In some instances, such as authentication failures, fixing an earlier issue will solve later ones. Therefore it is recommended that issues be addressed from the beginning of the test case to the end.

The table below lists the most frequent issues and gives some recommendations on how to resolve them.

| Session Status Type | Diagnostic Message Example | Possible Resolution |
|---|---|---|
| Error | Response Code 500<br>Server error. | Find and create missing parameter(s) as described below. |
| Warning | Response Code 401.<br>The user is not authenticated. | The replayed request was not authenticated.<br>Check Test authentication settings.<br>If subsequent requests are authenticated, this may be normal. |
| Warning | Verified response code does not match the recorded response code<br>Example: The verified response code is 302 and the recorded response is code 200. | Find and create missing parameter(s). |
| Warning | Verified redirect URL is different than the recorded redirect URL. | 1. If you are redirected to the same page but a different query string, this may be normal.<br>2. If you are redirected to a different page, you most likely need to create a parameter. |
| Warning | Verified response size is significantly different from the recorded response size. | Compare the recorded and replayed sessions. If the verified response is expected then ignore this warning. Otherwise, create a parameter. |
| Notification | Response Code 404.<br>The resource is not found on the server. | 1. Add the missing resource to the server.<br>2. If the resource is a static file (such as an image) that is not significant from a performance standpoint, delete this request from the test case. |
| Notification | The first request in the test case contains an ASP.NET SessionId cookie. The beginning of the test case<br>may be missing | This may be an indication that the beginning of the test case is not recorded. Re-record if this is true, otherwise ignore it. |
| Notification | The resource was cached during Test Case recording (response code 304) and not cached during replay | This notification can be ignored. If you with to avoid it, consider: |

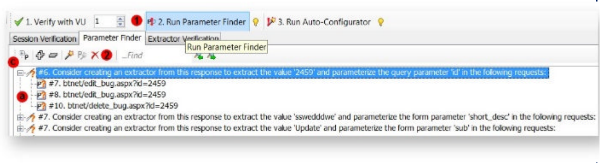
| Session Status Type | Diagnostic Message Example | Possible Resolution |
|---|---|---|
| | (response code 200). | - when recording a test case select browser Private Mode. See Recording with Web Browser<br>- or clear the browser cache and re-recording the Test.<br>- enabling cache control emulation in StresStimulus. See Test Case: Caching Rules |
| Notification | The resource was not cached during Test Case<br>recording (response code 200) and cached during<br>replay (response code 304). | Disable cache control. See Test Case: Caching Rules |

One of the most frequent reason of load test errors is missing parameter(s). There are several methods of identifying and and creating missing parameters which are described in the next several subsections listed below.

### Parameter Finder

Ent SP

Parameter Finder finds possible missing parameters and directs which extractors should be created to feed into these parameters. You need to run Parameter Finder only if test case verification created any diagnostic errors or warnings in the Session Verification tree.

| | |
|---|---|
| 1. Click Run Parameter Finder.<br>2. After Parameter Finder completes execution, it generates the Parameter Finder Tree, which displays suggested missing objects. The Parameter Finder Tree has two views.<br>a. In the Group by Extractors view:<br>•<br>○<br>▪ |   1. Verify with VU<br>1<br>1<br>2. Run Parameter Finder<br>3. Run Auto-Configurator<br>Session Verification<br>Parameter Finder<br>Extractor Verification<br>Run Parameter Finder<br>2<br>Find<br>c<br>a #6. Consider creating an extractor from this response to extract the value '2459' and parameterize the query parameter 'id' in the following requests:<br>#7. btnet/edit_bug.aspx?id=2459<br>#8. btnet/edit_bug.aspx?id=2459<br>#10. btnet/delete_bug.aspx?id=2459<br>#7. Consider creating an extractor from this response to extract the value 'sswedddwe' and parameterize the form parameter 'short_desc' in the following requests:<br>#7. Consider creating an extractor from this response to extract the value 'Update' and parameterize the form parameter 'sub' in the following requests: |

- Each parent node represents a recommended extractor with indication to which prior request it should be created from.
- Each child node represents a request with the missing parameter's name along with the recorded value that should be parameterized.

b. In the Group by Requests view:

•

○

▪

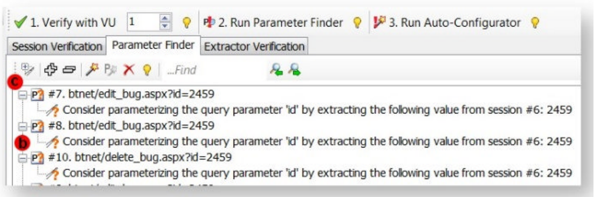
- Each parent node represents a request that likely requires parameterization.
- Each child node represents a pair of:

1. The missing parameter's name along with the recorded value that should be parameterized.
2. The recommended extractor with indication to which prior response it should be created from

To switch between two views, use the Click To Group By button (c).

**Tips:**

- To copy the selected object content, hit



#7. btnet/edit_bug.aspx?id=2459
Consider parameterizing the query parameter 'id' by extracting the following value from session #6: 2459
#8. btnet/edit_bug.aspx?id=2459
Consider parameterizing the query parameter 'id' by extracting the following value from session #6: 2459
#10. btnet/delete_bug.aspx?id=2459
Consider parameterizing the query parameter 'id' by extracting the following value from session #6: 2459

(Ctrl+C).

- To find and highlight an object selected on the Parameter Finder Tree on the Test Case Tree, double-click the object or right-click and select Highlight.

**Note:** Parameter Finder inspects a test case and the result of the Verify process, so you must run verify before running parameter Finder for the first time. After adding or removing extractors and parameters, in order to determine if all missing objects are created, rerun Parameter Finder. In order to determine if added parameterization fixed all test errors, rerun Verify.

### Parameter Creator

Ent SP

Creating Extractors and related parameters recommended by the Parameter Finder can reduce test configuration errors and help make the test more realistic. **Parameter Creator** automates the creation of such objects.

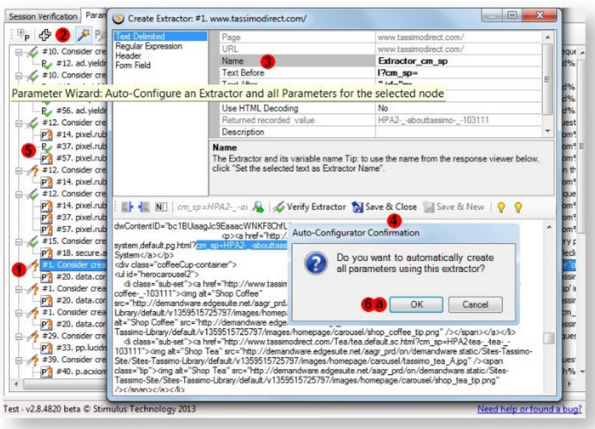
To use **Parameter Creator**, follow these steps:

1. Select an extractor.

2. Click **Parameter Creator:** Auto-configure an Extractor and all Parameters for the selected node on the toolbar or right-click and select it on the context menu.

3. In the popup extractor window, the Text Before and Text After properties are already pre-filled.

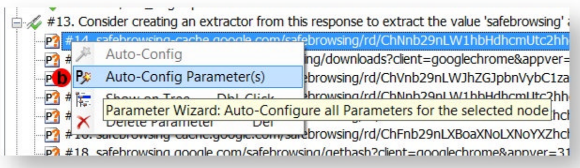
4. You can examine and adjust the extractor configuration if necessary, or simply click Save to create the


Session Verification
Create Extractor: #1. www.tassimodirect.com/
Text Delimited
Regular Expression
Header
Form Field
Parameter Wizard: Auto-Configure an Extractor and all Parameters for the selected node
Verify Extractor
Save & Close
Save & New
Auto-Configurator Confirmation
OK
Cancel
1
2
3
4
5
6 a
Test - v2.8.4820 beta © Stimulus Technology 2013
Need help or found a bug?

| | |
|---|---|
| Page | www.tassimodirect.com/ |
| URL | www.tassimodirect.com/ |
| Name | Extractor_cm_sp |
| Text Before | I?cm_sp= |
| Use HTML Decoding | No |
| Returned recorded value | HPA2-_-abouttassimo-_-103111 |
| Description | |

**Name**
The Extractor and its variable name Tip: to use the name from the response viewer below, click "Set the selected text as Extractor Name".

Do you want to automatically create all parameters using this extractor?

| | |
|---|---|
| extractor.<br><br>5. Initially, the Parameter Finder Tree designates all missing object with the **?** image. As missing objects get created their image is modified to a green checkbox.<br><br>6. After the extractor is created, Auto-Configurator displays a **Do you want to automatically create all parameters using this extractor?** MessageBox.<br><br>a. To create all the parameters using the extractor automatically at once, click OK.<br><br>b. To create the parameters automatically one-by-one, click Cancel and then select a parameter one at a time, right-click and select Auto-Config Parameter or click the same button on the toolbar.<br><br>7. Run Verify Test Case again to check if newly created objects reduce the number of errors or warnings. Delete unnecessary objects. |  #13. Consider creating an extractor from this response to extract the value 'safebrowsing'  Auto-Config<br>b<br>Auto-Config Parameter(s)<br>Parameter Wizard: Auto-Configure all Parameters for the selected node |

## Auto-Configurator

Ent SP

Auto-Configurator creates all extractors and parameters discovered by the Parameter Finder, in one sweep. It automatically executes the **Parameter Creator** multiple times to create every missing object. It is the fastest way to configure your test. However, it does not allow to review new objects before creating them, and as a result, some unnecessary objects can be created.

**Note:** Unnecessary objects do not address any errors or warnings discovered during test verification. They can even increase the number of errors and can create performance overheads on the test machine.

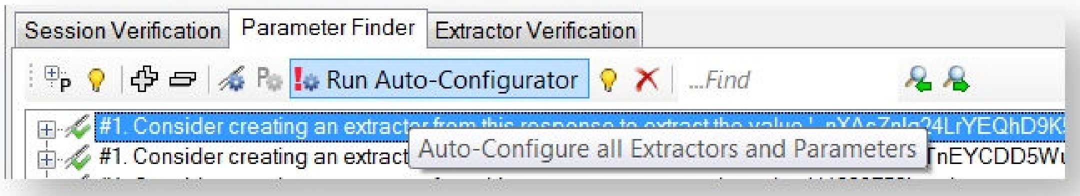
**Tip: When to use Auto-Configurator**

- If you are not familiar with the tested website and manually examining every extractor and parameter is difficult.

- If you need to quickly complete test configuration for preliminary load testing.

Otherwise, create parameters one-by-one using the **Parameter Creator.**

Run the **Auto Configurator** after running the **Parameter Finder**. On the Verify & Auto-Config toolbar, click **Run Auto-Configurator.**



#1. Consider creating an extractor from this response to extract the value '_nYAcZnlq24LrYEQhD9K
#1. Consider creating an extract TnEYCDD5W

After the Auto-Configurator completes execution, all missing objects will be designated with a green checkbox on the Parameter Finder Tree.

Run **Verify Test Case** again to check if the newly created objects reduced the number of errors or warnings. Delete unnecessary objects.

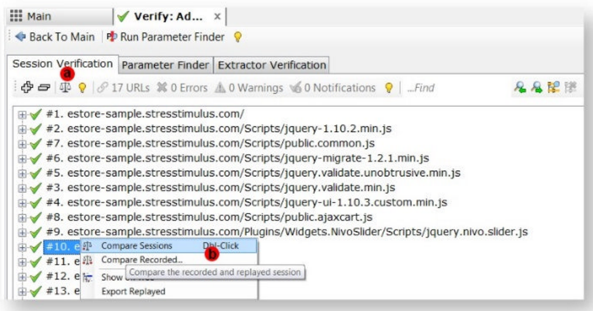
### Resolving Errors Manually

If Parameter Finder, Parameter Creator and Auto-Configurator failed to find all necessary parameters and did not resolve test errors, then use the following method:

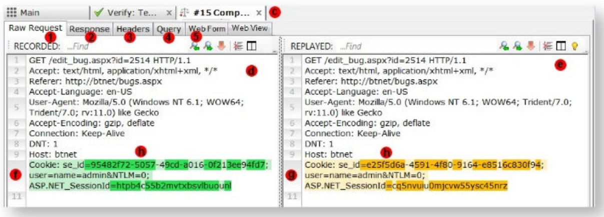
1. Use one of two techniques of locating dynamic request values:
   a. comparing recorded and replayed session.
   b. comparing two record sessions.
2. Create necessary variables to parameterize dynamic values:
   a. for every dynamic value originated on the server, create an extractor.
   b. for every dynamic value originated on the client, or entered by a user, create a data set, a generator or a function.
3. Create missing parameters, as described in the Parameterization section.

## Comparing Recorded and Replayed Sessions

To display Compare Sessions Inspector, select a session, click Compare (a) on the toolbar or in the sessions' context menu (b) or double-click the session. The Compare Sessions Inspector will open in the new tab (c). You can select from the several of views by clicking an appropriate tab. Every view shows a split screen displaying:



```
#1. estore-sample.stresstimulus.com/
#2. estore-sample.stresstimulus.com/Scripts/jquery-1.10.2.min.js
#7. estore-sample.stresstimulus.com/Scripts/public.common.js
#6. estore-sample.stresstimulus.com/Scripts/jquery-migrate-1.2.1.min.js
#5. estore-sample.stresstimulus.com/Scripts/jquery.validate.unobtrusive.min.js
#3. estore-sample.stresstimulus.com/Scripts/jquery.validate.min.js
#4. estore-sample.stresstimulus.com/Scripts/jquery-ui-1.10.3.custom.min.js
#8. estore-sample.stresstimulus.com/Scripts/public.ajaxcart.js
#9. estore-sample.stresstimulus.com/Plugins/Widgets.NivoSlider/Scripts/jquery.nivo.slider.js
#10. e
#11. e
#12. e
#13. e
```



RECORDED:

```
GET /edit_bug.aspx?id=2514 HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Referer: http://btnet/bugs.aspx
Accept-Language: en-US
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64;
Trident/7.0; rv:11.0) like Gecko
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
DNT: 1
Host: btnet
Cookie: se_id=95482f72-5057-49cd-a016-0f213ee94fd7;
user=name=admin&NTLM=0;
ASP.NET_SessionId=htpb4c55b2mvtxbsvlbuounl
```

REPLAYED:

```
GET /edit_bug.aspx?id=2514 HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Referer: http://btnet/bugs.aspx
Accept-Language: en-US
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0;
rv:11.0) like Gecko
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
DNT: 1
Host: btnet
Cookie: se_id=e25f5d6a-4591-4f80-9164-e8516c830f94;
user=name=admin&NTLM=0;
ASP.NET_SessionId=cq5nvuiu0mjcvw55ysc45nrz
```

- recorded session content on the pane left (d)
- replayed session content on the pane right (e)

The content is automatically compared. All differences are highlighted as follows:

- in the recorded pane (on the left) in green (f)
- in the replayed pane (on the right) in orange (g)

The following views are available:

**1. Raw request.** Displays an unformulated request content.

- 
    - Recorded and replayed line are numbered and auto aligned.
    - Lines with discrepancies are colored.
    - Characters with discrepancies are displayed in bold (h).

**2. Response.** Displays an unformulated response content.

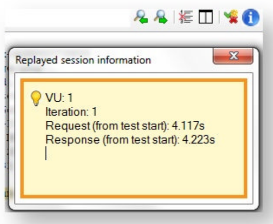
- 
    - Recorded and replayed line are numbered and auto aligned.
    - Left and right panes can be auto-scrolled vertically.
    - Line numbering on very large

sessions is disabled by default for performance, It can be manually re-enabled by checking the box.

- Lines with discrepancies are colored.
- Characters with discrepancies are displayed in bold.
- To show or hide line number, click ☰ or ☰.
- To split the screen 1:1, click ◫.
- To display replayed session information click ⓘ

The following information will be displayed:

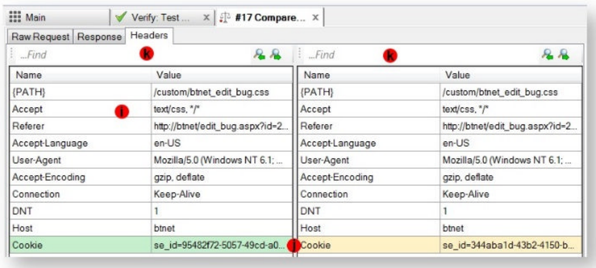
- VU number
- Iteration number
- Request issuing time from the test start
- Response receiving time from the test start


Replayed session information

VU: 1
Iteration: 1
Request (from test start): 4.117s
Response (from test start): 4.223s

**3. Header.** Displays request headers in the name / value pair format

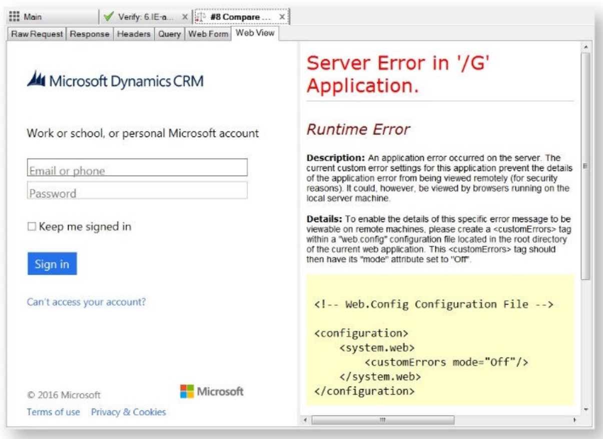
- Recorded and replayed headers are displayed in two aligned grids (i).
- Lines with discrepancies are colored (j).



| Name | Value |
|---|---|
| {PATH} | /custom/btnet_edit_bug.css |
| Accept | text/css, */* |
| Referer | http://btnet/edit_bug.aspx?id=2... |
| Accept-Language | en-US |
| User-Agent | Mozilla/5.0 (Windows NT 6.1; ... |
| Accept-Encoding | gzip, deflate |
| Connection | Keep-Alive |
| DNT | 1 |
| Host | btnet |
| Cookie | se_id=95482f72-5057-49cd-a0... |

| Name | Value |
|---|---|
| {PATH} | /custom/btnet_edit_bug.css |
| Accept | text/css, */* |
| Referer | http://btnet/edit_bug.aspx?id=2... |
| Accept-Language | en-US |
| User-Agent | Mozilla/5.0 (Windows NT 6.1; ... |
| Accept-Encoding | gzip, deflate |
| Connection | Keep-Alive |
| DNT | 1 |
| Host | btnet |
| Cookie | se_id=344aba1d-43b2-4150-b... |

**4. Query.** Displays only if the request has a

| | |
|---|---|
| query string in the name / value pair format<br>•<br>○<br>▪ Recorded and replayed query string are displayed in two aligned grids<br>▪ Lines with discrepancies are colored.<br><br>**5. Web Form.** Displays only if the request is a web form.<br>•<br>○<br>▪ Recorded and replayed web forms are displayed in two aligned grids.<br>▪ Lines with discrepancies are colored.<br><br>In all of the views, both panes are independently searchable using the Find boxes on the toolbar (k). | |
| **6. Web View.** Compare screenshots captured during records and verify. This helps to quickly pinpoint the differences or view errors. |  |

Work or school, or personal Microsoft account

☐ Keep me signed in

Can't access your account?

© 2016 Microsoft

Terms of use   Privacy & Cookies

Server Error in '/G' Application.

*Runtime Error*

**Description:** An application error occurred on the server. The current custom error settings for this application prevent the details of the application error from being viewed remotely (for security reasons). It could, however, be viewed by browsers running on the local server machine.

**Details:** To enable the details of this specific error message to be viewable on remote machines, please create a <customErrors> tag within a "web.config" configuration file located in the root directory of the current web application. This <customErrors> tag should then have its "mode" attribute set to "Off".

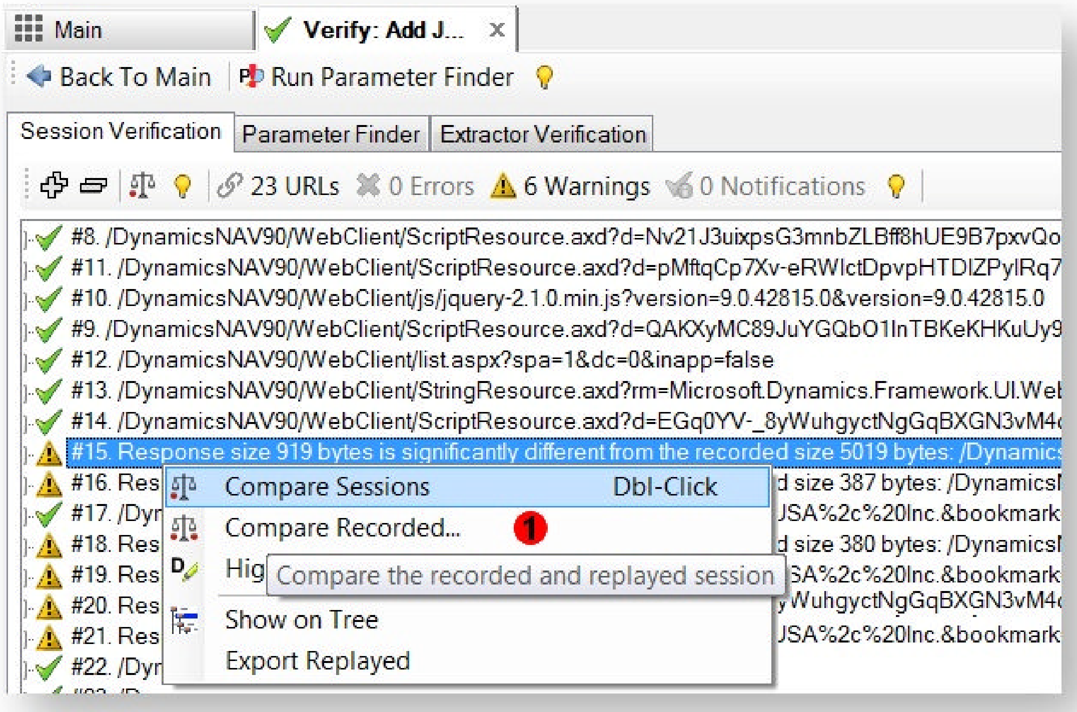
```
<!-- Web.Config Configuration File -->

<configuration>
    <system.web>
        <customErrors mode="Off"/>
    </system.web>
</configuration>
```

### Comparing Two Recorded Sessions

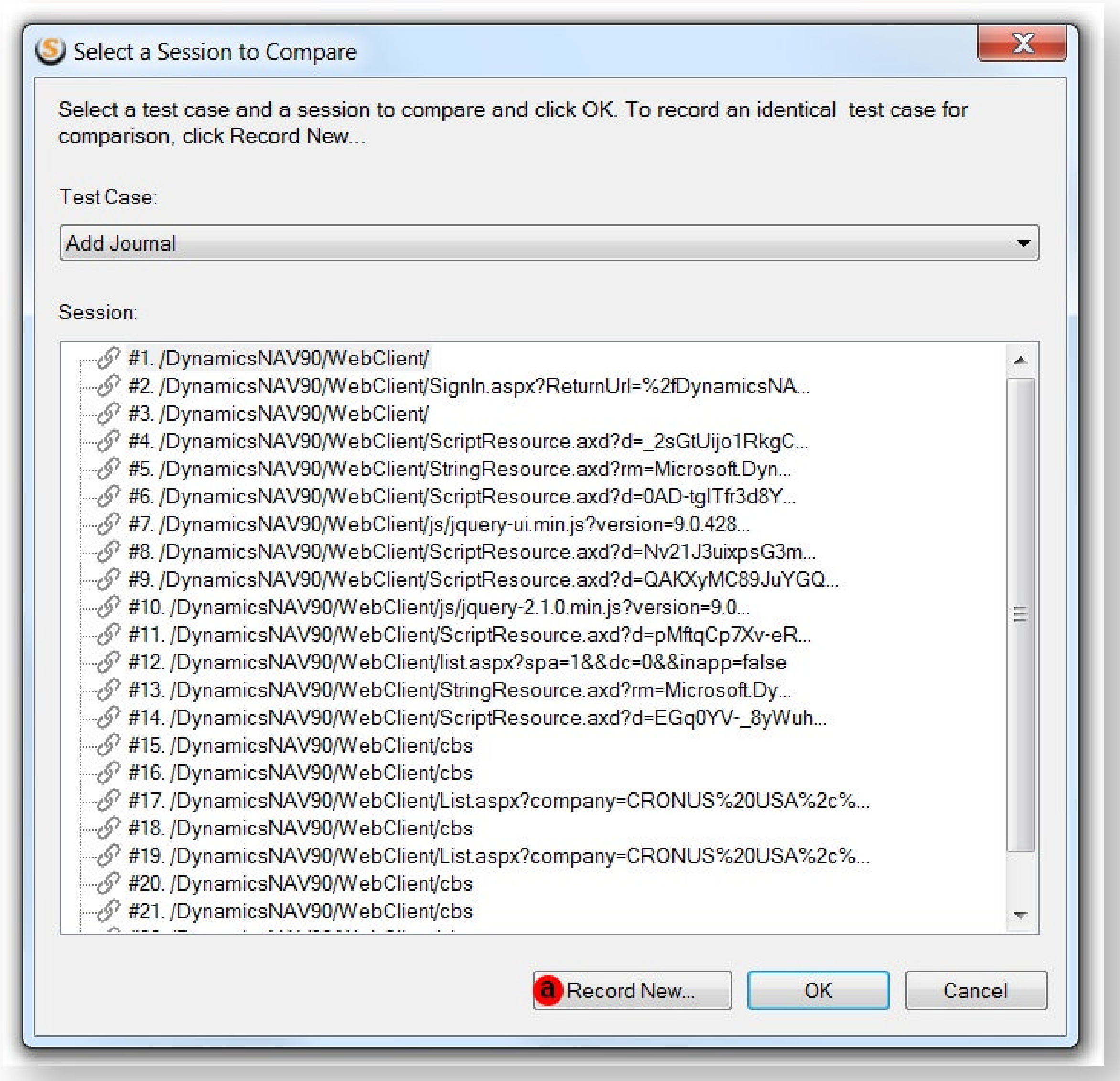
Another way to find missing parameters is to re-record the same scenario again and compare the two recordings to identify dynamic values. It consists of the following steps:

1. After running verify, in the Verification tree, right-click on the error/warning session in the original test case and select **Compare Recorded** option.



```
#8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3mnbZLBff8hUE9B7pxvQo
#11. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eRWIctDpvpHTDIZPyIRq7
#10. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0.42815.0&version=9.0.42815.0
#9. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQbO1InTBKeKHKuUy9
#12. /DynamicsNAV90/WebClient/list.aspx?spa=1&dc=0&inapp=false
#13. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dynamics.Framework.UI.Wel
#14. /DynamicsNAV90/WebClient/ScriptResource.axd?d=EGq0YV-_8yWuhgyctNgGqBXGN3vM4
#15. Response size 919 bytes is significantly different from the recorded size 5019 bytes: /Dynamics
#16. Res                                                  d size 387 bytes: /Dynamics
#17. /Dyr                                                 JSA%2c%20Inc.&bookmark
#18. Res                                                  d size 380 bytes: /Dynamics
#19. Res                                                  SA%2c%20Inc.&bookmark
#20. Res                                                  yWuhgyctNgGqBXGN3vM4
#21. Res                                                  JSA%2c%20Inc.&bookmark
#22. /Dyr
```

2. In the **Select a Session to Compare** window, click **Record New...** (a) to re-record the identical test scenario for comparison. The recording wizard will appear to take you through the steps of re-recording. This will create a test case that can be deleted after the correlation process is complete. Therefore you don't need to annotate transactions. The recording wizard will automatically give a suggested test case name `<Current Test Case> - Compare`. The default mix weight for this test case will be set to 0, so it will not be executed during the test run. Once recording is complete, the verify tree should still be open, go back to Step 1 of the process above and then proceed to Step 3.


Select a Session to Compare

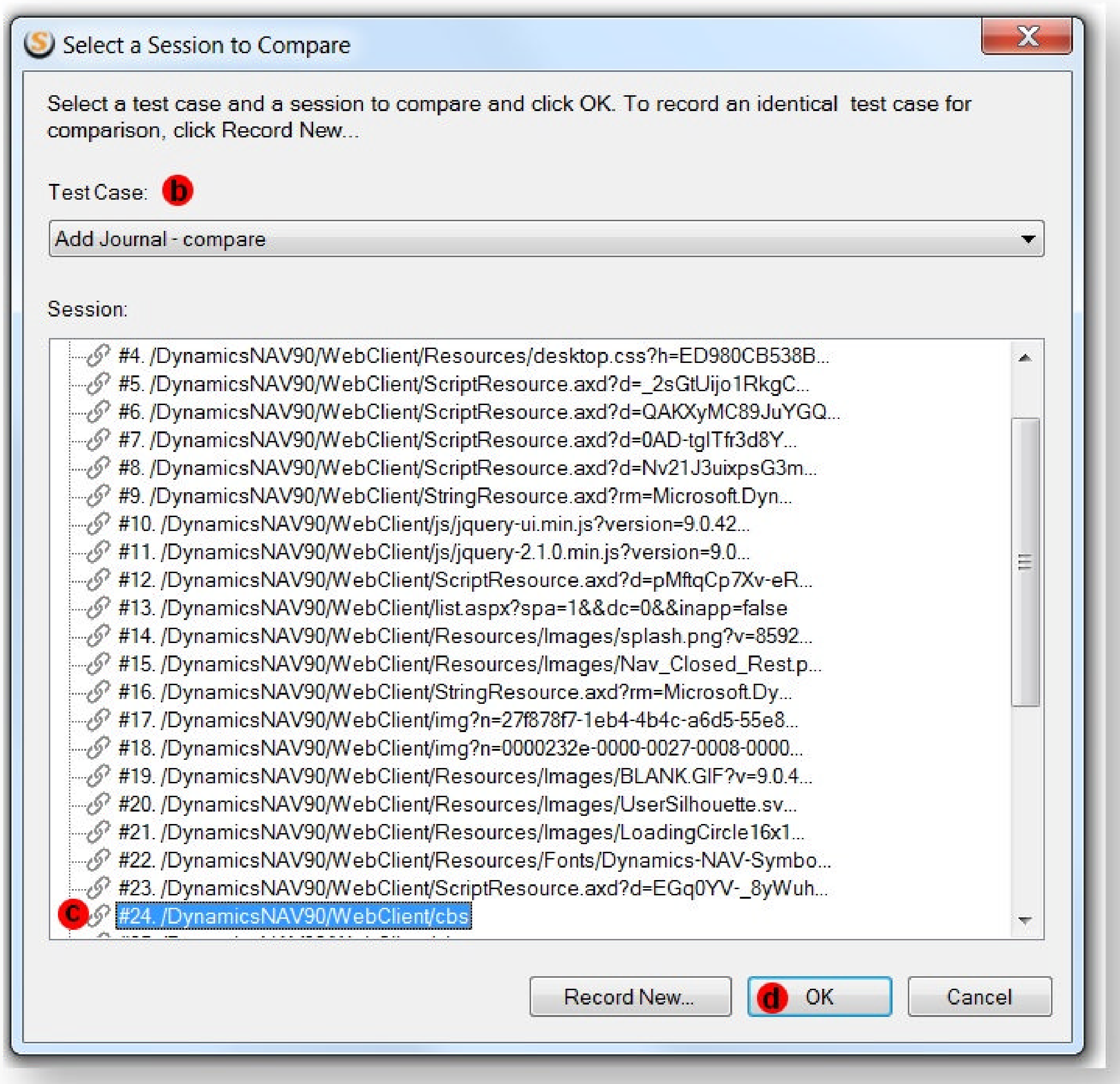
Select a test case and a session to compare and click OK. To record an identical test case for comparison, click Record New...

Test Case: Add Journal

Session:

```
#1. /DynamicsNAV90/WebClient/
#2. /DynamicsNAV90/WebClient/SignIn.aspx?ReturnUrl=%2fDynamicsNA...
#3. /DynamicsNAV90/WebClient/
#4. /DynamicsNAV90/WebClient/ScriptResource.axd?d=_2sGtUijo1RkgC...
#5. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dyn...
#6. /DynamicsNAV90/WebClient/ScriptResource.axd?d=0AD-tglTfr3d8Y...
#7. /DynamicsNAV90/WebClient/js/jquery-ui.min.js?version=9.0.428...
#8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3m...
#9. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQ...
#10. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0...
#11. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eR...
#12. /DynamicsNAV90/WebClient/list.aspx?spa=1&&dc=0&&inapp=false
#13. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dy...
#14. /DynamicsNAV90/WebClient/ScriptResource.axd?d=EGq0YV-_8yWuh...
#15. /DynamicsNAV90/WebClient/cbs
#16. /DynamicsNAV90/WebClient/cbs
#17. /DynamicsNAV90/WebClient/List.aspx?company=CRONUS%20USA%2c%...
#18. /DynamicsNAV90/WebClient/cbs
#19. /DynamicsNAV90/WebClient/List.aspx?company=CRONUS%20USA%2c%...
#20. /DynamicsNAV90/WebClient/cbs
#21. /DynamicsNAV90/WebClient/cbs
```

3. If the test case was already recorded twice, then StresStimulus will automatically find a matching test case and matching session. The matching test case will be selected in the Test Case drop-down as `<Current Test Case> - Compare` (b). The corresponding session (c) will also be selected. While test cases recorded twice are similar, they are not identical. Therefore the session number will not necessarily match, but the session content will. If the session is not selected, select it manually. Then click **OK** (d).



Select a test case and a session to compare and click OK. To record an identical test case for comparison, click Record New...

Test Case: Add Journal - compare

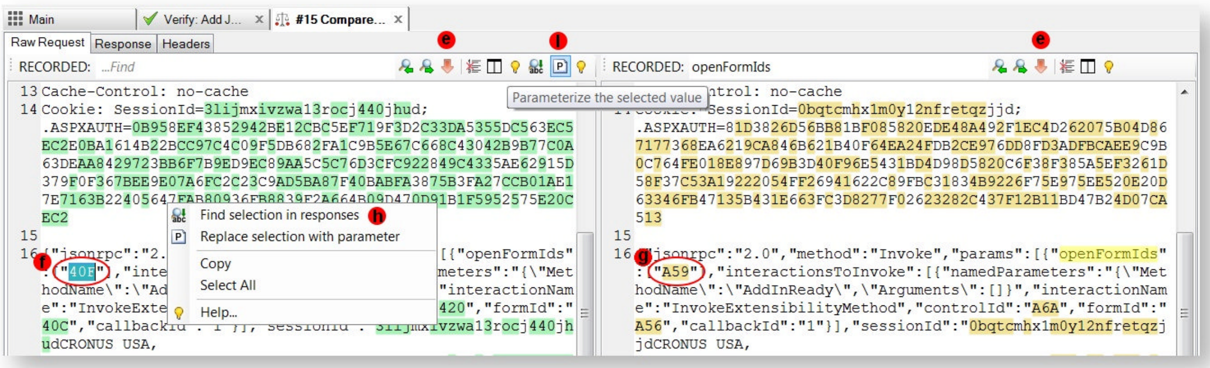
Session:

```
#4. /DynamicsNAV90/WebClient/Resources/desktop.css?h=ED980CB538B...
#5. /DynamicsNAV90/WebClient/ScriptResource.axd?d=_2sGtUijo1RkgC...
#6. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQ...
#7. /DynamicsNAV90/WebClient/ScriptResource.axd?d=0AD-tgITfr3d8Y...
#8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3m...
#9. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dyn...
#10. /DynamicsNAV90/WebClient/js/jquery-ui.min.js?version=9.0.42...
#11. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0...
#12. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eR...
#13. /DynamicsNAV90/WebClient/list.aspx?spa=1&&dc=0&&inapp=false
#14. /DynamicsNAV90/WebClient/Resources/Images/splash.png?v=8592...
#15. /DynamicsNAV90/WebClient/Resources/Images/Nav_Closed_Rest.p...
#16. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dy...
#17. /DynamicsNAV90/WebClient/img?n=27f878f7-1eb4-4b4c-a6d5-55e8...
#18. /DynamicsNAV90/WebClient/img?n=0000232e-0000-0027-0008-0000...
#19. /DynamicsNAV90/WebClient/Resources/Images/BLANK.GIF?v=9.0.4...
#20. /DynamicsNAV90/WebClient/Resources/Images/UserSilhouette.sv...
#21. /DynamicsNAV90/WebClient/Resources/Images/LoadingCircle16x1...
#22. /DynamicsNAV90/WebClient/Resources/Fonts/Dynamics-NAV-Symbo...
#23. /DynamicsNAV90/WebClient/ScriptResource.axd?d=EGq0YV-_8yWuh...
#24. /DynamicsNAV90/WebClient/cbs
```

4. The Compare Session inspector will open in a separate tab. The original session will be on the left, secondary session will be on the right. Compare the two recorded requests, and pay attention to the differences. Usually, these differences point out the dynamic parameters that need to be created. To quickly traverse through the differences, repeatedly click **Highlight the Next Difference** (e) on the toolbar. In the example shown in the screenshot above, the first value of the parameter **openFormIds** was recorded as 40F (f). The second time the same value was recorded as A59 (g). This is an indication that this parameter is dynamic. There are three possibilities:

- the dynamic value should be correlated
- the dynamic value is already auto-correlated

- the dynamic value should be parameterized, unless you've already parameterized it.



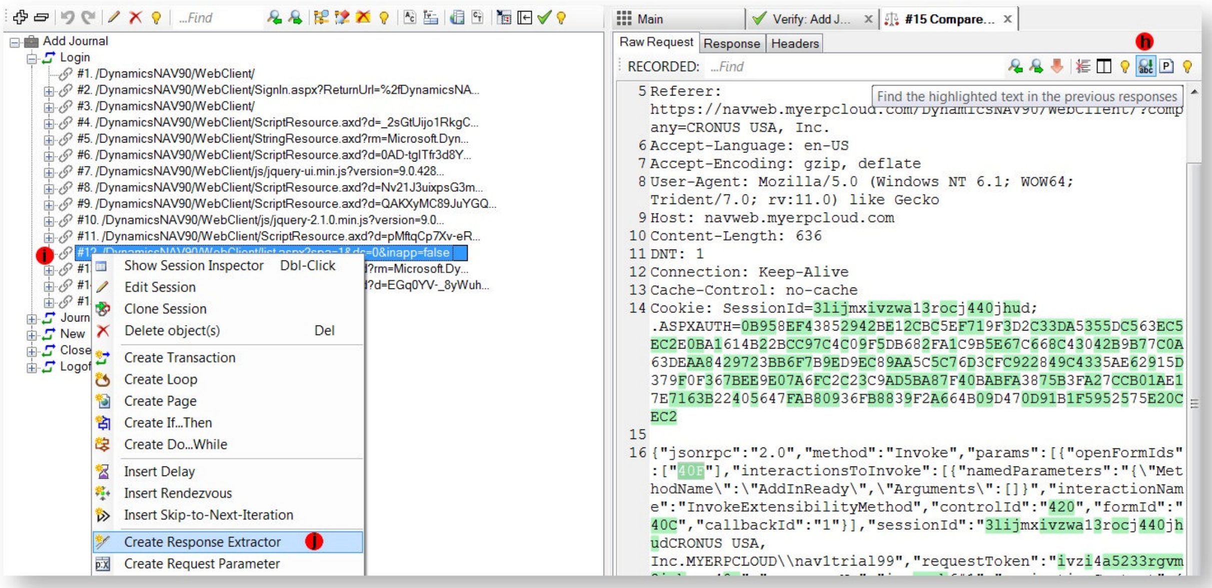
```
Cache-Control: no-cache
Cookie: SessionId=3lijmxivzwa13rocj440jhud;
.ASPXAUTH=0B958EF43852942BE12CBC5EF719F3D2C33DA5355DC563EC5
EC2E0BA1614B22BCC97C4C09F5DB682FA1C9B5E67C668C43042B9B77C0A
63DEAA8429723BB6F7B9ED9EC89AA5C5C76D3CFC922849C4335AE62915D
379F0F367BEE9E07A6FC2C23C9AD5BA87F40BABFA3875B3FA27CCB01AE1
7E7163B22405647FAB80936FB8839F2A664B09D470D91B1F5952575E20C
EC2

{"jsonrpc":"2.                                   [{"openFormIds"
:["40E"],"inte                                   meters":"{\"Met
hodName\":\"Ad                                   "interactionNam
e":"InvokeExte                                   420","formId":"
40C","callbackId":"1"}],"sessionId":"3lijmxivzwa13rocj440jh
udCRONUS USA,
```

```
ntrol: no-cache
SessionId=0bqtcmhx1m0y12nfretqzjjd;
.ASPXAUTH=81D3826D56BB81BF085820EDE48A492F1EC4D262075B04D86
7177368EA6219CA846B621B40F64EA24FDB2CE976DD8FD3ADFBCAEE9C9B
0C764FE018E897D69B3D40F96E5431BD4D98D5820C6F38F385A5EF3261D
58F37C53A19222054FF26941622C89FBC31834B9226F75E975EE520E20D
63346FB47135B431E663FC3D8277F02623282C437F12B11BD47B24D07CA
513

{"jsonrpc":"2.0","method":"Invoke","params":[{"openFormIds"
:["A59"],"interactionsToInvoke":[{"namedParameters":"{\"Met
hodName\":\"AddInReady\",\"Arguments\":[]}","interactionNam
e":"InvokeExtensibilityMethod","controlId":"A6A","formId":"
A56","callbackId":"1"}],"sessionId":"0bqtcmhx1m0y12nfretqzj
jdCRONUS USA,
```

5. Select the green text on the left and click the **Find the highlighted text in the previous responses** button (h). Alternatively, right click and select the same item on the context menu. The responses that contain the selected value should be highlighted in the test case tree. If no responses are highlighted, then the dynamic value was originated on the client and should be parameterized. Otherwise, create the appropriate extractor for the highlighted value (i) (if one doesn't already exist). Right-click and select Create Response Extractor (j).

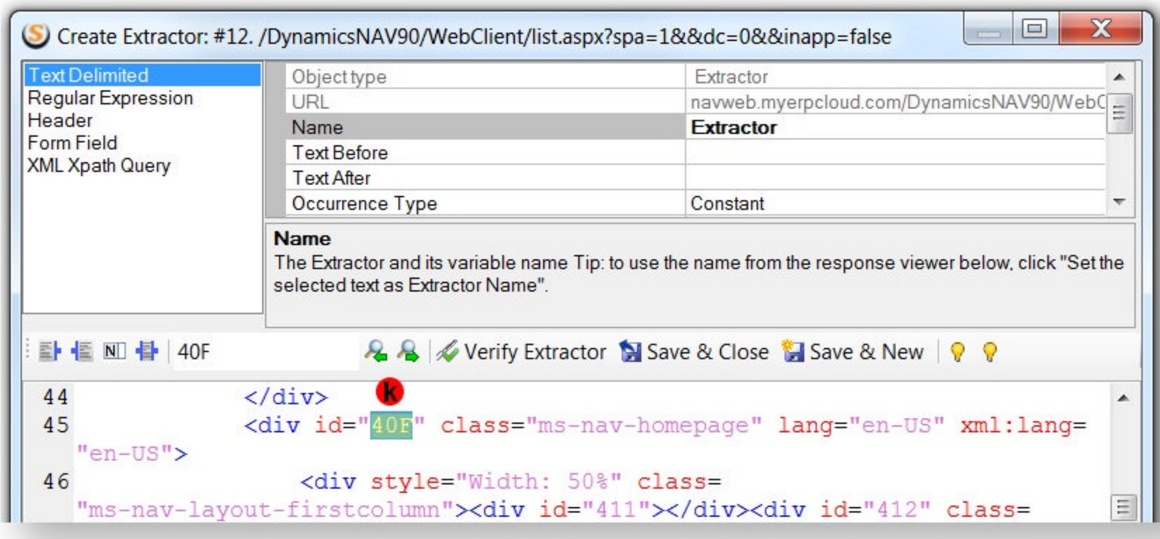


```
Add Journal
  Login
    #1. /DynamicsNAV90/WebClient/
    #2. /DynamicsNAV90/WebClient/SignIn.aspx?ReturnUrl=%2fDynamicsNA...
    #3. /DynamicsNAV90/WebClient/
    #4. /DynamicsNAV90/WebClient/ScriptResource.axd?d=_2sGtUijo1RkgC...
    #5. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dyn...
    #6. /DynamicsNAV90/WebClient/ScriptResource.axd?d=0AD-tglTfr3d8Y...
    #7. /DynamicsNAV90/WebClient/js/jquery-ui.min.js?version=9.0.428...
    #8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3m...
    #9. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQ...
    #10. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0...
    #11. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eR...
    #12. /DynamicsNAV90/WebClient/list.aspx?spa=1&dc=0&inapp=false
    #1                                 ?rm=Microsoft.Dy...
    #1                                 ?d=EGq0YV-_8yWuh...
    #1
  Journ
  New
  Close
  Logof
```

```
Referer:
https://navweb.myerpcloud.com/DynamicsNAV90/WebClient/?comp
any=CRONUS USA, Inc.
Accept-Language: en-US
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64;
Trident/7.0; rv:11.0) like Gecko
Host: navweb.myerpcloud.com
Content-Length: 636
DNT: 1
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: SessionId=3lijmxivzwa13rocj440jhud;
.ASPXAUTH=0B958EF43852942BE12CBC5EF719F3D2C33DA5355DC563EC5
EC2E0BA1614B22BCC97C4C09F5DB682FA1C9B5E67C668C43042B9B77C0A
63DEAA8429723BB6F7B9ED9EC89AA5C5C76D3CFC922849C4335AE62915D
379F0F367BEE9E07A6FC2C23C9AD5BA87F40BABFA3875B3FA27CCB01AE1
7E7163B22405647FAB80936FB8839F2A664B09D470D91B1F5952575E20C
EC2

{"jsonrpc":"2.0","method":"Invoke","params":[{"openFormIds"
:["40E"],"interactionsToInvoke":[{"namedParameters":"{\"Met
hodName\":\"AddInReady\",\"Arguments\":[]}","interactionNam
e":"InvokeExtensibilityMethod","controlId":"420","formId":"
40C","callbackId":"1"}],"sessionId":"3lijmxivzwa13rocj440jh
udCRONUS USA,
Inc.MYERPCLOUD\\nav1trial99","requestToken":"ivzi4a5233rgvm
```

6. Create Extractor dialog will appear with the dynamic value highlighted (k). Continue creating the extractor as described in the Extractors section.

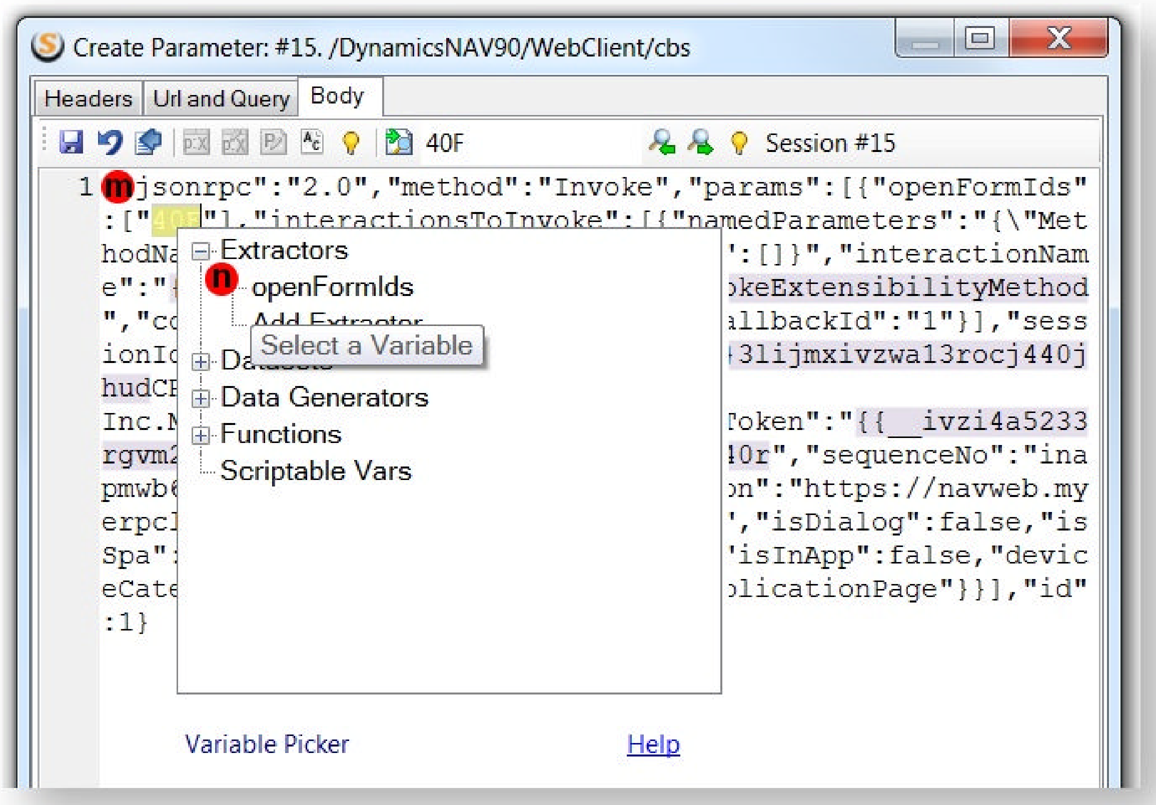

Create Extractor: #12. /DynamicsNAV90/WebClient/list.aspx?spa=1&&dc=0&&inapp=false

| | |
|---|---|
| Object type | Extractor |
| URL | navweb.myerpcloud.com/DynamicsNAV90/WebC |
| Name | **Extractor** |
| Text Before | |
| Text After | |
| Occurrence Type | Constant |

**Name**
The Extractor and its variable name Tip: to use the name from the response viewer below, click "Set the selected text as Extractor Name".

```
                </div>
                <div id="40F" class="ms-nav-homepage" lang="en-US" xml:lang=
"en-US">
                    <div style="Width: 50%" class=
"ms-nav-layout-firstcolumn"><div id="411"></div><div id="412" class=
```

7. To create the parameter, go back to the the Compare Session inspector, select the dynamic value (f) and click the **Parameterize the selected value** button (l) on the toolbar. A Create Parameter dialog with the pre-selected value (m) will open. Right-click, and from the Variable Picker, select the extractor (n) created in step 6.

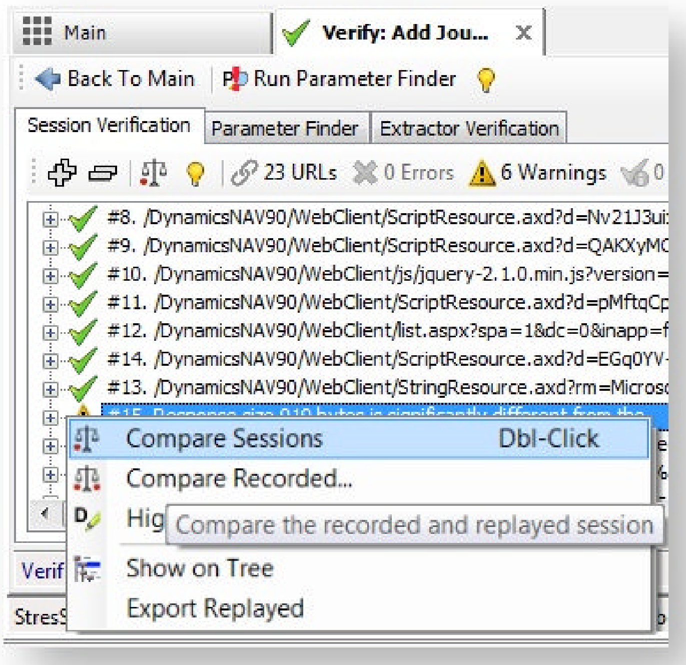


```
1 mjsonrpc":"2.0","method":"Invoke","params":[{"openFormIds"
:["40F"],"interactionsToInvoke":[{"namedParameters":"{\"Met
hodNa                                  ':[]}","interactionNam
e":"                                   okeExtensibilityMethod
","c                                   allbackId":"1"}],"sess
ionIc                                  3lijmxivzwa13rocj440j
hudCF
Inc.N                                  Token":"{{__ivzi4a5233
rgvm2                                  40r","sequenceNo":"ina
pmwb6                                  on":"https://navweb.my
erpcl                                  ',"isDialog":false,"is
Spa":                                  'isInApp":false,"devic
eCate                                  plicationPage"}}],"id"
:1}
```

## Highlighting Non-Correlated Dynamic Values

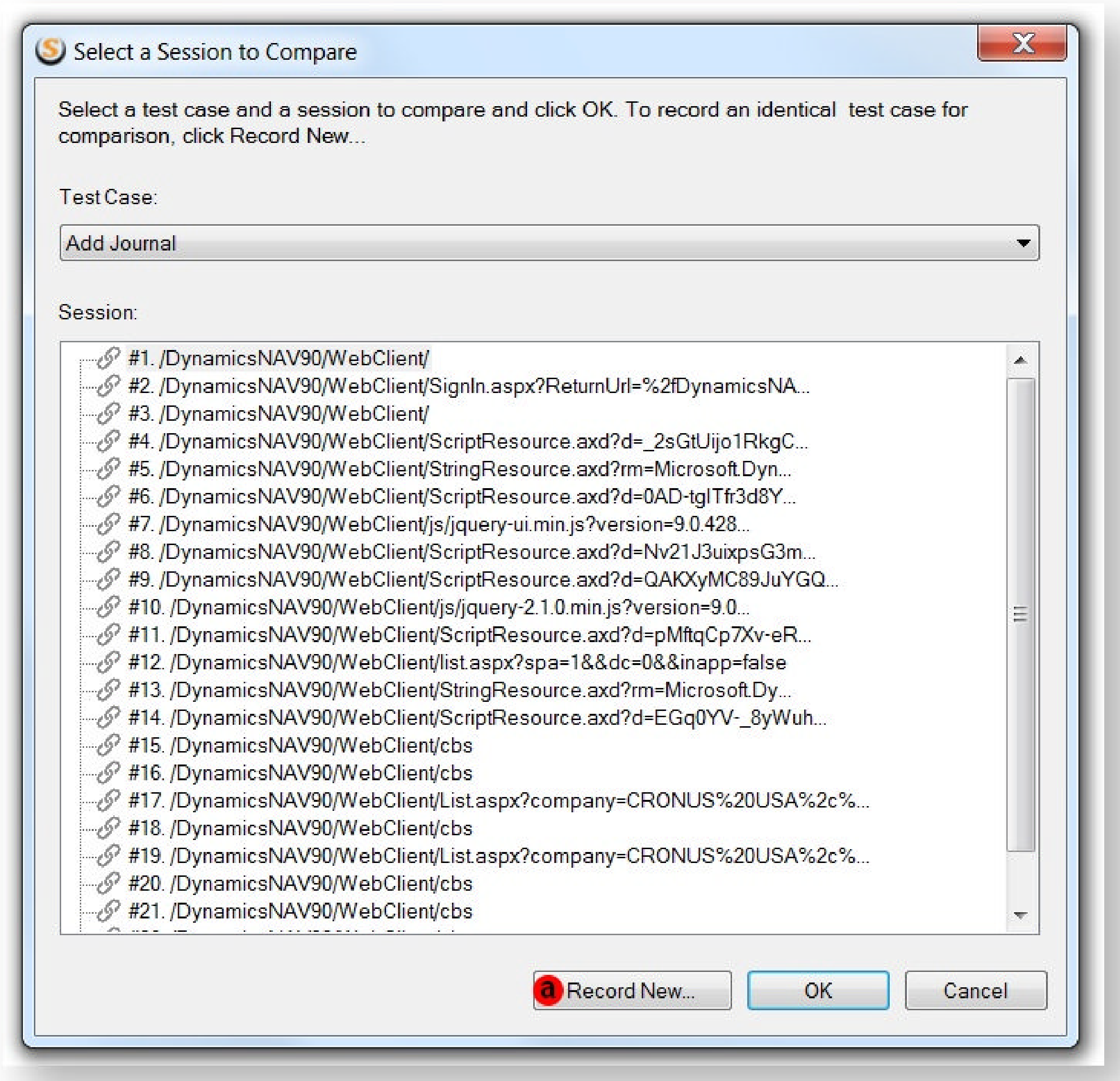
After re-recording the same scenario, highlighting non-correlated dynamic values is another useful feature. This feature highlights the possible values that might need correlation by comparing the re-recorded scenario with the verified test case.

1. After running verify, in the Verification tree, right-click on the error/warning session in the original test case and select **Highlight non-correlated dynamic values...**



#8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3ui
#9. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC
#10. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=
#11. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp
#12. /DynamicsNAV90/WebClient/list.aspx?spa=1&dc=0&inapp=f
#14. /DynamicsNAV90/WebClient/ScriptResource.axd?d=EGq0YV
#13. /DynamicsNAV90/WebClient/StringResource.axd?rm=Micros

2. In the **Select a Session to Compare** window, click **Record New...** (a) to re-record the identical test scenario for comparison. The recording wizard will appear to take you through the steps of re-recording. This will create a temporary test case that can be deleted after the correlation process is complete. Therefore you don't need to annotate transactions. The recording wizard will automatically give a suggested test case name `<Current Test Case> - Compare` . Once recording is complete, the verify tree should still be open. Go back to Step 1 of the process above and then proceed to Step 3.


Select a Session to Compare

Select a test case and a session to compare and click OK. To record an identical test case for comparison, click Record New...

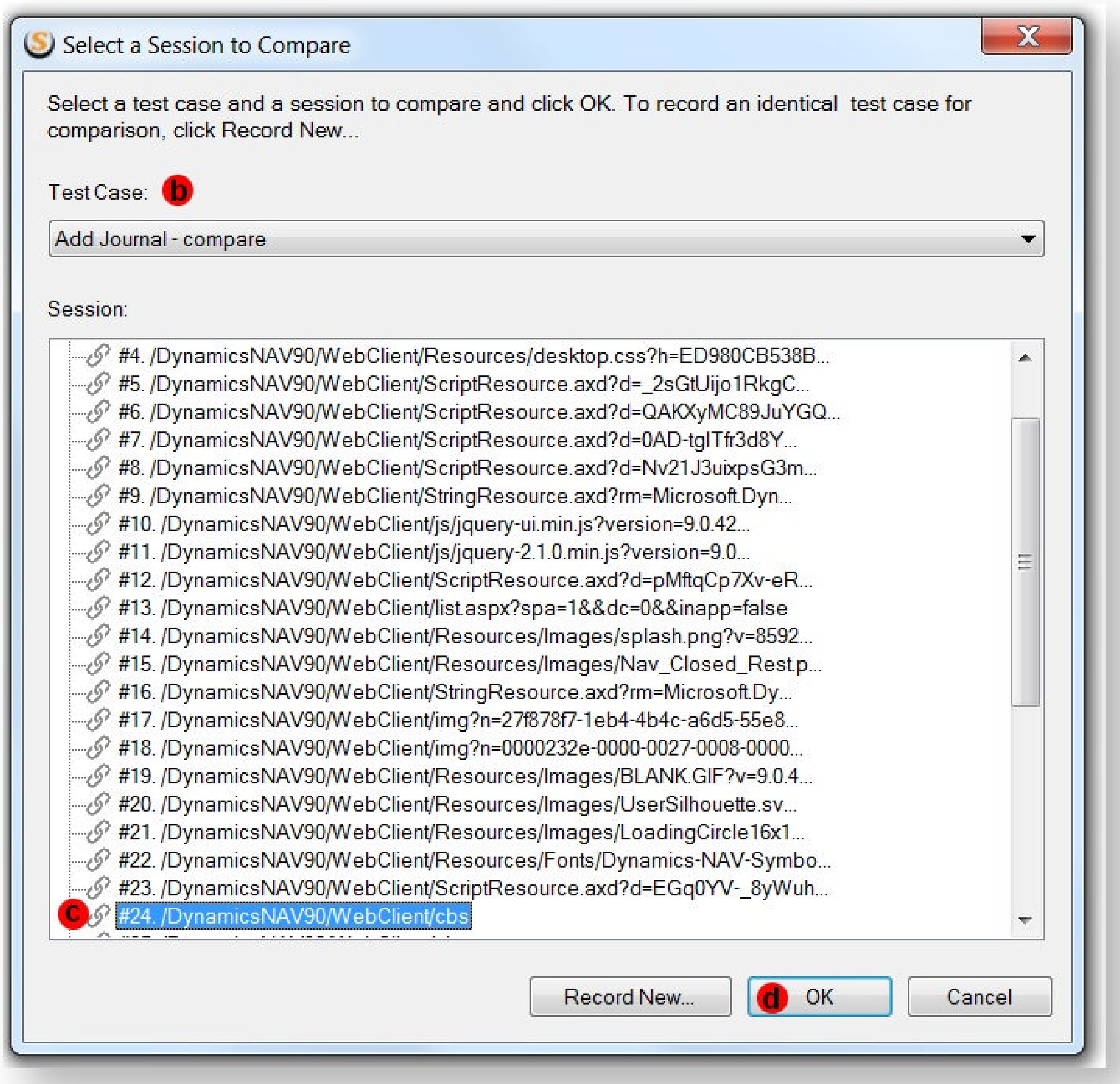
Test Case:

Add Journal

Session:

```
#1. /DynamicsNAV90/WebClient/
#2. /DynamicsNAV90/WebClient/SignIn.aspx?ReturnUrl=%2fDynamicsNA...
#3. /DynamicsNAV90/WebClient/
#4. /DynamicsNAV90/WebClient/ScriptResource.axd?d=_2sGtUijo1RkgC...
#5. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dyn...
#6. /DynamicsNAV90/WebClient/ScriptResource.axd?d=0AD-tglTfr3d8Y...
#7. /DynamicsNAV90/WebClient/js/jquery-ui.min.js?version=9.0.428...
#8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3m...
#9. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQ...
#10. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0...
#11. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eR...
#12. /DynamicsNAV90/WebClient/list.aspx?spa=1&&dc=0&&inapp=false
#13. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dy...
#14. /DynamicsNAV90/WebClient/ScriptResource.axd?d=EGq0YV-_8yWuh...
#15. /DynamicsNAV90/WebClient/cbs
#16. /DynamicsNAV90/WebClient/cbs
#17. /DynamicsNAV90/WebClient/List.aspx?company=CRONUS%20USA%2c%...
#18. /DynamicsNAV90/WebClient/cbs
#19. /DynamicsNAV90/WebClient/List.aspx?company=CRONUS%20USA%2c%...
#20. /DynamicsNAV90/WebClient/cbs
#21. /DynamicsNAV90/WebClient/cbs
```

3. If the identical test scenario for comparison is already recorded, then it will be selected in the Test Case drop-down as `<Current Test Case> - Compare` (b). StresStimulus will try to automatically select the corresponding session (c). The session number will not necessarily match, but the session content will. If the session is not selected, select it manually. Then click **OK** (d).


Select a Session to Compare

Select a test case and a session to compare and click OK. To record an identical test case for comparison, click Record New...

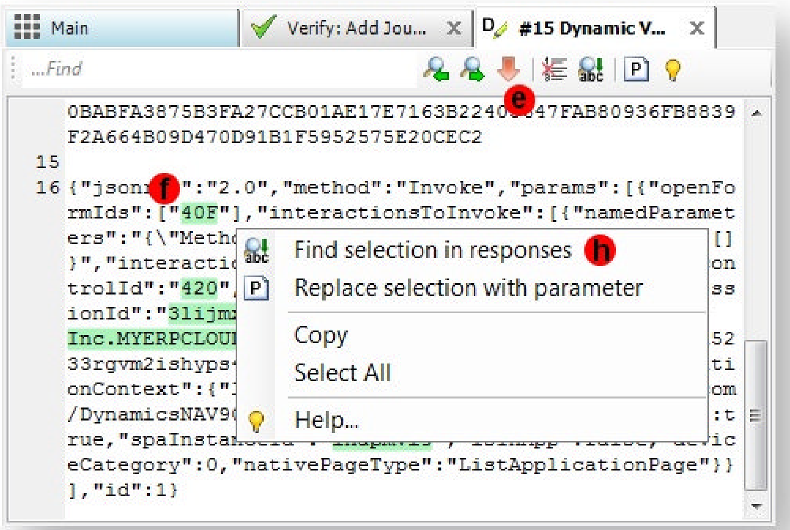
Test Case: b

Add Journal - compare

Session:

#4. /DynamicsNAV90/WebClient/Resources/desktop.css?h=ED980CB538B...
#5. /DynamicsNAV90/WebClient/ScriptResource.axd?d=_2sGtUijo1RkgC...
#6. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQ...
#7. /DynamicsNAV90/WebClient/ScriptResource.axd?d=0AD-tglTfr3d8Y...
#8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3m...
#9. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dyn...
#10. /DynamicsNAV90/WebClient/js/jquery-ui.min.js?version=9.0.42...
#11. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0...
#12. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eR...
#13. /DynamicsNAV90/WebClient/list.aspx?spa=1&&dc=0&&inapp=false
#14. /DynamicsNAV90/WebClient/Resources/Images/splash.png?v=8592...
#15. /DynamicsNAV90/WebClient/Resources/Images/Nav_Closed_Rest.p...
#16. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dy...
#17. /DynamicsNAV90/WebClient/img?n=27f878f7-1eb4-4b4c-a6d5-55e8...
#18. /DynamicsNAV90/WebClient/img?n=0000232e-0000-0027-0008-0000...
#19. /DynamicsNAV90/WebClient/Resources/Images/BLANK.GIF?v=9.0.4...
#20. /DynamicsNAV90/WebClient/Resources/Images/UserSilhouette.sv...
#21. /DynamicsNAV90/WebClient/Resources/Images/LoadingCircle16x1...
#22. /DynamicsNAV90/WebClient/Resources/Fonts/Dynamics-NAV-Symbo...
#23. /DynamicsNAV90/WebClient/ScriptResource.axd?d=EGq0YV-_8yWuh...
c #24. /DynamicsNAV90/WebClient/cbs

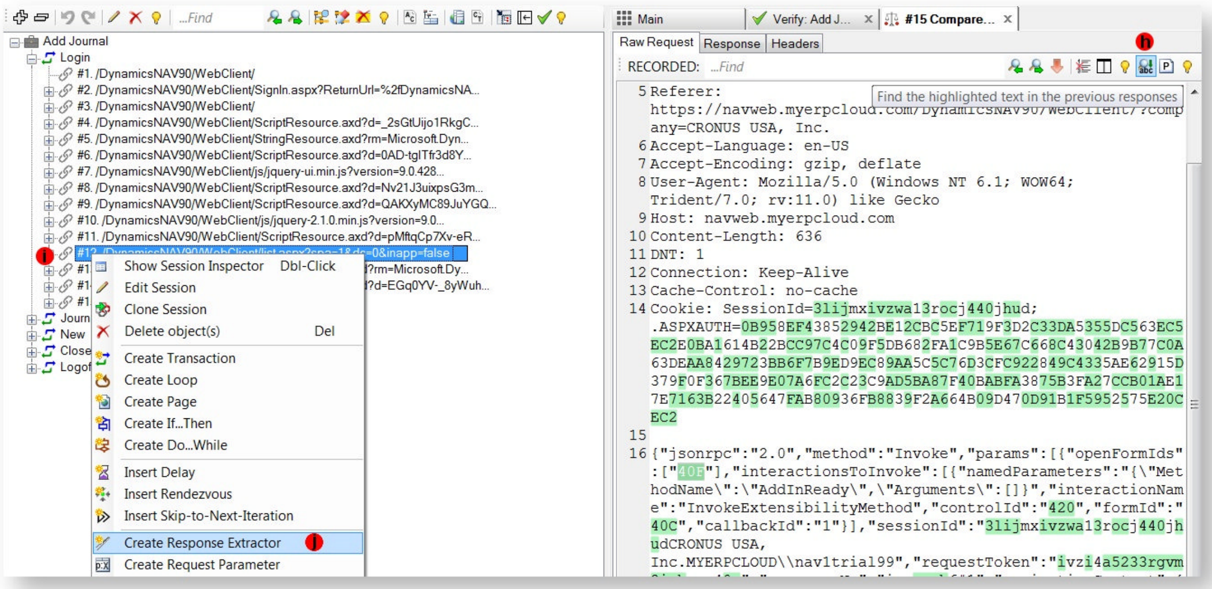
4. The Highlighting non-correlated dynamic values inspector will open in a separate tab. The highlighted text points out the dynamic parameters that need to be created. To quickly traverse through the differences, repeatedly click **Highlight the Next Difference** (e) on the toolbar. In the example shown in the screenshot above, the first value of the parameter **openFormIds**, recorded as 40F (f), should be parameterized.



```
0BABFA3875B3FA27CCB01AE17E7163B2240 47FAB80936FB8839
F2A664B09D470D91B1F5952575E20CEC2

{"json    ":"2.0","method":"Invoke","params":[{"openFo
rmIds":["40F"],"interactionsToInvoke":[{"namedParamet
ers":"{\"Meth                                        []
}","interacti                                        on
trolId":"420",                                       ss
ionId":"3lijm
Inc.MYERPCLOU                                        52
33rgvm2ishyps                                        ti
onContext":{"                                        om
/DynamicsNAV9                                        :t
rue,"spaInstan                                       ic
eCategory":0,"nativePageType":"ListApplicationPage"}}
],"id":1}
```

5. Select the green text on the left and click the **Find the highlighted text in the previous responses** button (h). Alternatively, right click and select the same item in the context menu. The responses that contain the selected value will be highlighted in the test case tree (i). If no responses are highlighted, then the dynamic value was originated on the client and should be parameterized. Otherwise, create the appropriate extractor for the highlighted value (i) (if one doesn't already exist). Right-click and select Create Response Extractor (j).

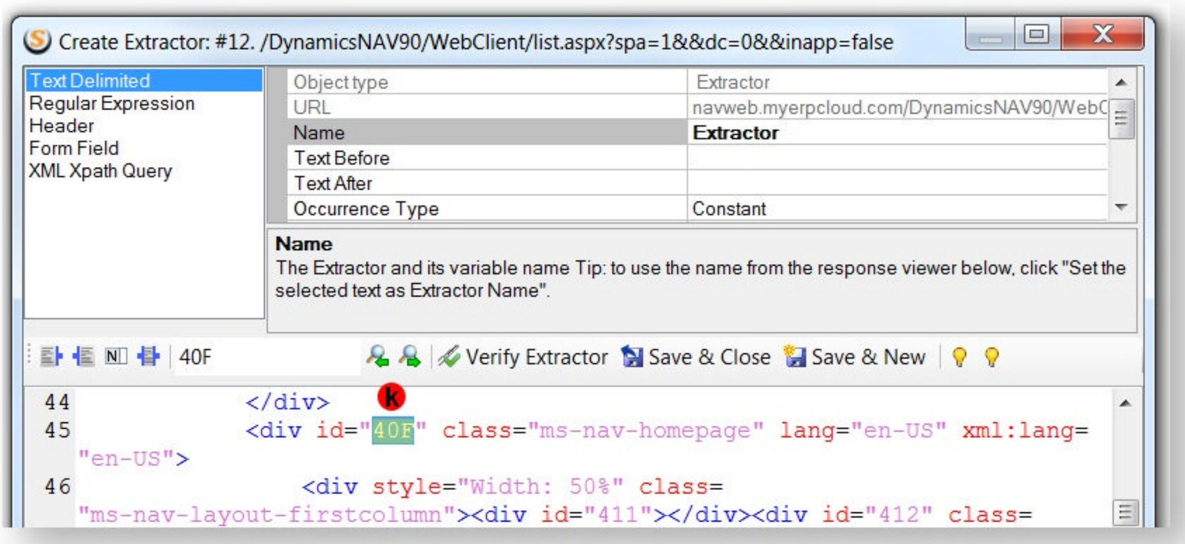


```
Add Journal
  Login
    #1. /DynamicsNAV90/WebClient/
    #2. /DynamicsNAV90/WebClient/SignIn.aspx?ReturnUrl=%2fDynamicsNA...
    #3. /DynamicsNAV90/WebClient/
    #4. /DynamicsNAV90/WebClient/ScriptResource.axd?d=_2sGtUijo1RkgC...
    #5. /DynamicsNAV90/WebClient/StringResource.axd?rm=Microsoft.Dyn...
    #6. /DynamicsNAV90/WebClient/ScriptResource.axd?d=0AD-tglTfr3d8Y...
    #7. /DynamicsNAV90/WebClient/js/jquery-ui.min.js?version=9.0.428...
    #8. /DynamicsNAV90/WebClient/ScriptResource.axd?d=Nv21J3uixpsG3m...
    #9. /DynamicsNAV90/WebClient/ScriptResource.axd?d=QAKXyMC89JuYGQ...
    #10. /DynamicsNAV90/WebClient/js/jquery-2.1.0.min.js?version=9.0...
    #11. /DynamicsNAV90/WebClient/ScriptResource.axd?d=pMftqCp7Xv-eR...
    #12. /DynamicsNAV90/WebClient/list.aspx?spa=1&dc=0&inapp=false
    #1                       ?rm=Microsoft.Dy...
    #1                       ?d=EGq0YV-_8yWuh...
    #1
  Journ
  New
  Close
  Logof
```

```
Referer:
https://navweb.myerpcloud.com/DynamicsNAV90/WebClient/?comp
any=CRONUS USA, Inc.
Accept-Language: en-US
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64;
Trident/7.0; rv:11.0) like Gecko
Host: navweb.myerpcloud.com
Content-Length: 636
DNT: 1
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: SessionId=3lijmxivzwa13rocj440jhud;
.ASPXAUTH=0B958EF43852942BE12CBC5EF719F3D2C33DA5355DC563EC5
EC2E0BA1614B22BCC97C4C09F5DB682FA1C9B5E67C668C43042B9B77C0A
63DEAA8429723BB6F7B9ED9EC89AA5C5C76D3CFC922849C4335AE62915D
379F0F367BEE9E07A6FC2C23C9AD5BA87F40BABFA3875B3FA27CCB01AE1
7E7163B22405647FAB80936FB8839F2A664B09D470D91B1F5952575E20C
EC2

{"jsonrpc":"2.0","method":"Invoke","params":[{"openFormIds"
:["40F"],"interactionsToInvoke":[{"namedParameters":"{\"Met
hodName\":\"AddInReady\",\"Arguments\":[]}","interactionNam
e":"InvokeExtensibilityMethod","controlId":"420","formId":"
40C","callbackId":"1"}],"sessionId":"3lijmxivzwa13rocj440jh
udCRONUS USA,
Inc.MYERPCLOUD\\nav1trial99","requestToken":"ivzi4a5233rgvm
```

6. Create Extractor dialog will appear with the dynamic value highlighted (k). Continue creating the extractor as described in the Extractors section.



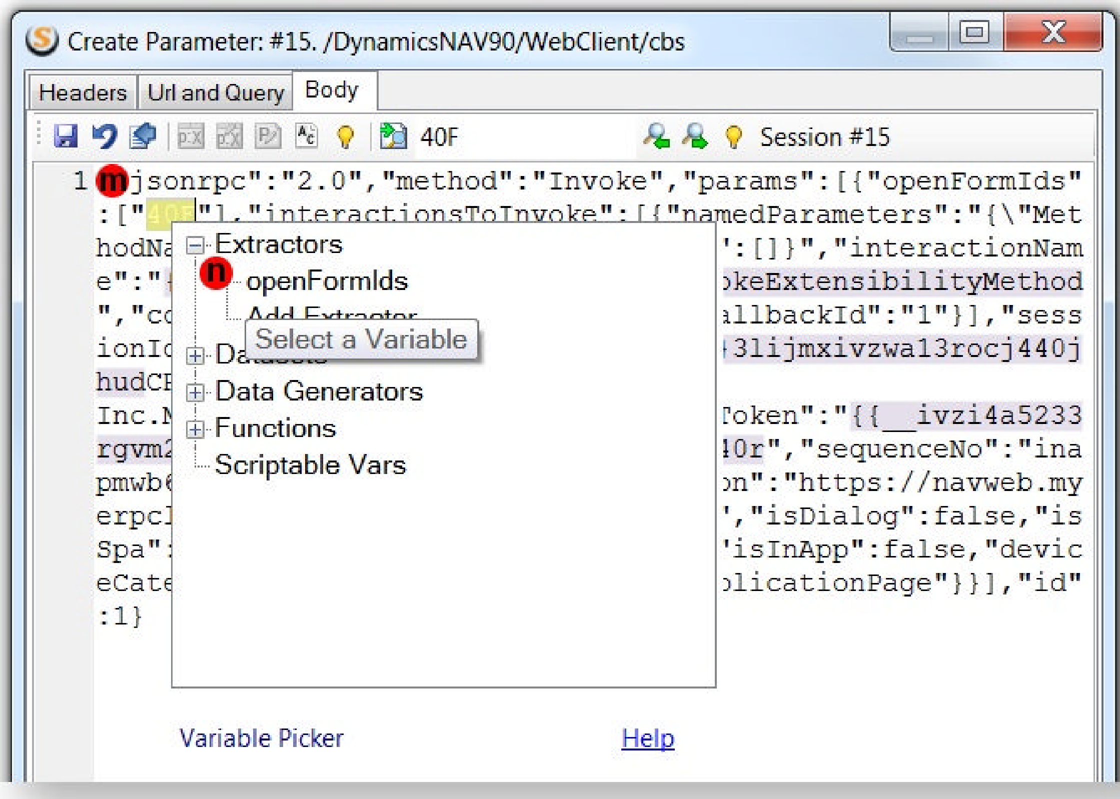
Create Extractor: #12. /DynamicsNAV90/WebClient/list.aspx?spa=1&&dc=0&&inapp=false

| | |
|---|---|
| Object type | Extractor |
| URL | navweb.myerpcloud.com/DynamicsNAV90/WebC |
| Name | **Extractor** |
| Text Before | |
| Text After | |
| Occurrence Type | Constant |

**Name**
The Extractor and its variable name Tip: to use the name from the response viewer below, click "Set the selected text as Extractor Name".

```
            </div>
            <div id="40F" class="ms-nav-homepage" lang="en-US" xml:lang=
"en-US">
                <div style="Width: 50%" class=
"ms-nav-layout-firstcolumn"><div id="411"></div><div id="412" class=
```

7. To create the parameter, go back to the the Compare Session inspector, select the dynamic value (f) and click the **Parameterize the selected value** button (l) on the toolbar. A Create Parameter dialog with the pre-selected value (m) will open. Right-click, and from the Variable Picker, select the extractor (n) created in step 6.



```
{"jsonrpc":"2.0","method":"Invoke","params":[{"openFormIds"
:["40F"],"interactionsToInvoke":[{"namedParameters":"{\"Met
hodNa                                               ":[]}","interactionNam
e":"                                                InvokeExtensibilityMethod
","co                                               allbackId":"1"}],"sess
ionIc                                               3lijmxivzwa13rocj440j
hudCF
Inc.N                                               Token":"{{__ivzi4a5233
rgvm2                                               40r","sequenceNo":"ina
pmwb6                                               on":"https://navweb.my
erpcl                                               ',"isDialog":false,"is
Spa":                                               'isInApp":false,"devic
eCate                                               plicationPage"}}],"id"
:1}
```

### 4.4.3.4 Inspecting Sessions

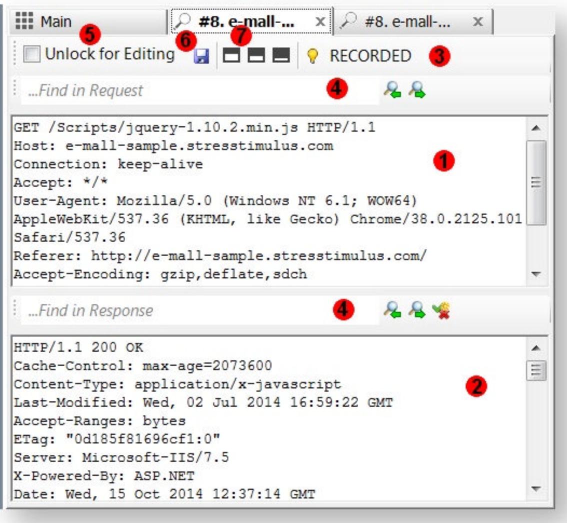
Sometimes it is necessary to inspect or modify a recorded or replayed session by itself. Session Iinspector is a tool for browsing and editing session content.

There are several ways to open Session Inspector,

- double-click **Recorded** or **Replayed** session in the verification tree.
- double-click **Recorded** session in the Test Case tree.
- double-click a session bar in the Waterfall.

A new tab with the following information will open:

1. The request is displayed on the top.



```
GET /Scripts/jquery-1.10.2.min.js HTTP/1.1
Host: e-mall-sample.stresstimulus.com
Connection: keep-alive
Accept: */*
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/38.0.2125.101
Safari/537.36
Referer: http://e-mall-sample.stresstimulus.com/
Accept-Encoding: gzip,deflate,sdch
```

```
HTTP/1.1 200 OK
Cache-Control: max-age=2073600
Content-Type: application/x-javascript
Last-Modified: Wed, 02 Jul 2014 16:59:22 GMT
Accept-Ranges: bytes
ETag: "0d185f81696cf1:0"
Server: Microsoft-IIS/7.5
X-Powered-By: ASP.NET
Date: Wed, 15 Oct 2014 12:37:14 GMT
```

2. The response is displayed on the bottom.

3. The recorded and replayed session inspectors are marked accordingly. Replayed session display VU number, iteration and a test case associated was the replayed session.

4. You can search the request and the response independently using the Find boxes.

5. To edit the session, click **Unlock for Editing.**

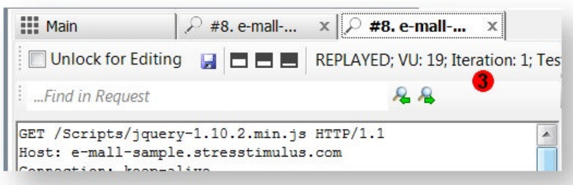
6. To save changes, click **Save.**

7. To quickly auto size the request and response panes, use resize buttons:

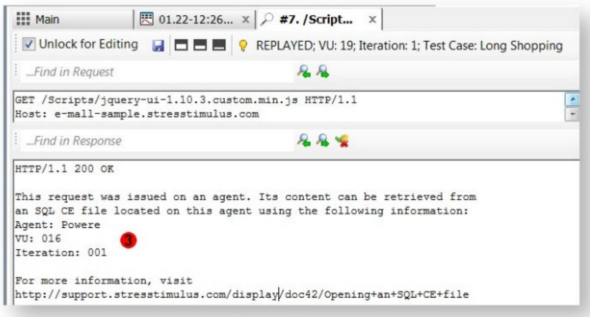
- 
    - 
        - Split the window at 1/4
        - Split the window at 1/2
        - Split the window at 3/4

**Note:** In order to open Session Inspector, session content must be saved in storage accessible from the controller. This information is available in tests that ran from the controller without agents . In distributed tests , when SQL Server is used as storage, this information is also available on the controller as well. However, in distributed tests with SQL Server CE-based storage, the sessions initiated on the agents are stored on the agents. You still can open such sessions in the Session Inspector, if **Save sessions from agents** property in **Configure Test** -> **Test Result Storage** section is set to **Yes** (default), because this setting will force replicating minimum necessary data from the agents to the controller.

8. During test run, a significant amount of



```
GET /Scripts/jquery-1.10.2.min.js HTTP/1.1
Host: e-mall-sample.stresstimulus.com
```

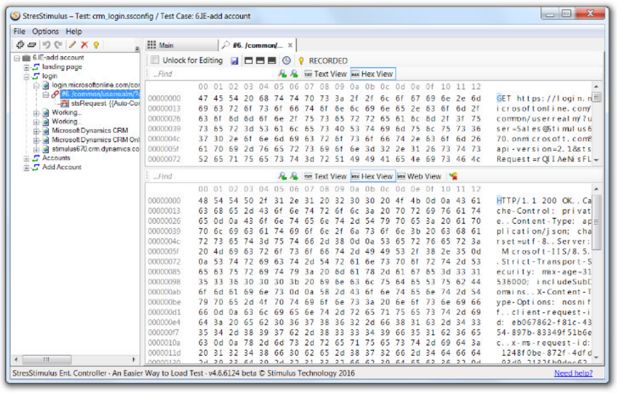


```
GET /Scripts/jquery-ui-1.10.3.custom.min.js HTTP/1.1
Host: e-mall-sample.stresstimulus.com
```

```
HTTP/1.1 200 OK

This request was issued on an agent. Its content can be retrieved from
an SQL CE file located on this agent using the following information:
Agent: Powere
VU: 016
Iteration: 001

For more information, visit
http://support.stresstimulus.com/display/doc42/Opening+an+SQL+CE+file
```


StresStimulus – Test: crm_login.ssconfig / Test Case: 6.IE-add account
File Options Help
Main
#6. /common/...
Unlock for Editing
RECORDED
...Find
Text View
Hex View
Web View
6.IE-add account
landing page
login
Working..
Working
Microsoft Dynamics CRM
Microsoft Dynamics CRM Onl
stimulus670.crm.dynamics.co
Accounts
Add Account
StresStimulus Ent. Controller - An Easier Way to Load Test - v4.6.6124 beta © Stimulus Technology 2016
Need help?

```
          00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12
00000000  47 45 54 20 68 74 74 70 73 3a 2f 2f 6c 6f 67 69 6e 2e 6d   GET https://login.m
00000013  69 63 72 6f 73 6f 66 74 6f 6e 6c 69 6e 65 2e 63 6f 6d 2f   icrosoftonline.com/
00000026  63 6f 6d 6d 6f 6e 2f 75 73 65 72 72 65 61 6c 6d 2f 3f 75   common/userrealm/?u
00000039  73 65 72 3d 53 61 6c 65 73 40 53 74 69 6d 75 6c 75 73 36   ser=Sales@Stimulus6
0000004c  37 30 2e 6f 6e 6d 69 63 72 6f 73 6f 66 74 2e 63 6f 6d 26   70.onmicrosoft.com&
0000005f  61 70 69 2d 76 65 72 73 69 6f 6e 3d 32 2e 31 26 73 74 73   api-version=2.1&sts
00000072  52 65 71 75 65 73 74 3d 72 51 49 49 41 65 4e 69 73 46 4c   Request=rQIIAeNisFL
```

```
          00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12
00000000  48 54 54 50 2f 31 2e 31 20 32 30 30 20 4f 4b 0d 0a 43 61   HTTP/1.1 200 OK..Ca
00000013  63 68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 70 72 69 76 61 74   che-Control: privat
00000026  65 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 61 70   e..Content-Type: ap
00000039  70 6c 69 63 61 74 69 6f 6e 2f 6a 73 6f 6e 3b 20 63 68 61   plication/json; cha
0000004c  72 73 65 74 3d 75 74 66 2d 38 0d 0a 53 65 72 76 65 72 3a   rset=utf-8..Server:
0000005f  20 4d 69 63 72 6f 73 6f 66 74 2d 49 49 53 2f 38 2e 35 0d    Microsoft-IIS/8.5.
00000072  0a 53 74 72 69 63 74 2d 54 72 61 6e 73 70 6f 72 74 2d 53   .Strict-Transport-S
00000085  65 63 75 72 69 74 79 3a 20 6d 61 78 2d 61 67 65 3d 33 31   ecurity: max-age=31
00000098  35 33 36 30 30 30 3b 20 69 6e 63 6c 75 64 65 53 75 62 44   536000; includeSubD
000000ab  6f 6d 61 69 6e 73 0d 0a 58 2d 43 6f 6e 74 65 6e 74 2d 54   omains..X-Content-T
000000be  79 70 65 2d 4f 70 74 69 6f 6e 73 3a 20 6e 6f 73 6e 69 66   ype-Options: nosnif
000000d1  66 0d 0a 63 6c 69 65 6e 74 2d 72 65 71 75 65 73 74 2d 69   f..client-request-i
000000e4  64 3a 20 65 62 30 36 37 38 36 32 2d 66 38 31 63 2d 34 33   d: eb067862-f81c-43
000000f7  35 34 2d 38 39 37 62 2d 38 33 33 34 39 66 35 31 62 36 65   54-897b-83349f51b6e
0000010a  63 0d 0a 78 2d 6d 73 2d 72 65 71 75 65 73 74 2d 69 64 3a   c..x-ms-request-id:
0000011d  20 31 32 34 38 66 30 62 65 2d 38 37 32 66 2d 34 64 66 64    1248f0be-872f-4dfd
```

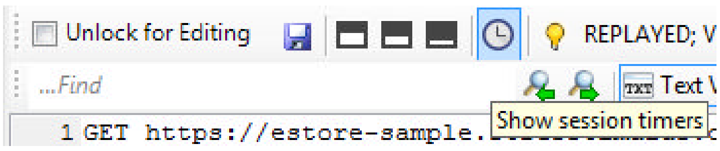
performance information collected on agents. Only part of this data, which is most important for consolidated reports, is mirrored to the controller. Some of the data is exclusively stored on the agent. For example, if the session was initiated on the agent and is opened on the controller, its response will display some session metadata, such as host, URL, and response code. However, detailed response body information will not be presented. Instead directions will be provided to access complete request and response content on the agent by Opening an SQL CE file on the agent.

The request and response inspectors have the following views:

- Text View: Shows the request/response headers and body as text.
- Hex View: Shows the request/response headers and body in hexadecimal format. This is especially helpful when the content of the session is binary.
- Web View: Shows the how the response will look in the browser (only in the response inspector)

#### 4.4.3.5 Session Timers

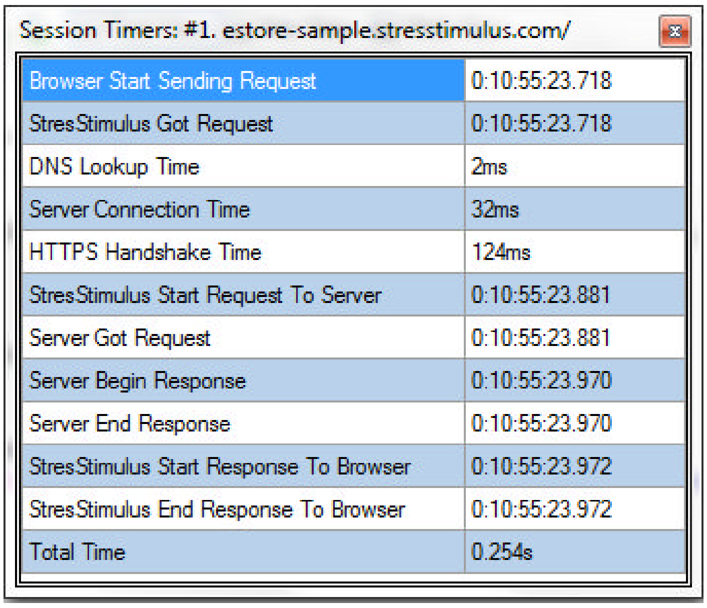
Each session has a list of timers and time-stamps of multiple events. To access them, click the **Show session timers** button in the inspector toolbar.



```
1 GET https://estore-sample.
```

The timers dialog will pop-up and show the following info:

| | |
|---|---|
| Browser Start Sending Request | Time when the browser initiate the request. (This can be ignored for replayed sessions.) |
| StresStimulus Got Request | Time when the StresStimulus received the request. |
| DNS Lookup Time | Number of ms to for DNS lookup-up. |
| Server Connection Time | Number of ms to connect to the server. |
| HTTPS Handshake Time | Number of ms to perform an HTTPS handshake. |
| StresStimulus Start Request To Server | Time when StresStimulus initates request to the server. |
| Server Got Request | Time when server received the request. |
| Server Begin Response | Time when server initates response to StresStimulus. |
| Server End Response | Time when StresStimulus received the response. |
| StresStimulus Start Response To Browser | Time when StresStimulus initates response to the browser. (This can be ignored for replayed sessions.) |
| StresStimulus End Response To Browser | Time when browser received the response. (This can be ignored for replayed sessions.) |
| Total Time | Number of seconds the entire operation took. |


Session Timers: #1. estore-sample.stresstimulus.com/

| | |
|---|---|
| Browser Start Sending Request | 0:10:55:23.718 |
| StresStimulus Got Request | 0:10:55:23.718 |
| DNS Lookup Time | 2ms |
| Server Connection Time | 32ms |
| HTTPS Handshake Time | 124ms |
| StresStimulus Start Request To Server | 0:10:55:23.881 |
| Server Got Request | 0:10:55:23.881 |
| Server Begin Response | 0:10:55:23.970 |
| Server End Response | 0:10:55:23.970 |
| StresStimulus Start Response To Browser | 0:10:55:23.972 |
| StresStimulus End Response To Browser | 0:10:55:23.972 |
| Total Time | 0.254s |

### 4.4.3.6 Extractor Verification and Script Optimization

After Verify is completed, the tree in the Extractor Verification tab will show extractors' values. Next to every extractor, there is a status image showing whether an exception in handling the extractor was encountered. One of 4 statuses can be assigned to an extractor:

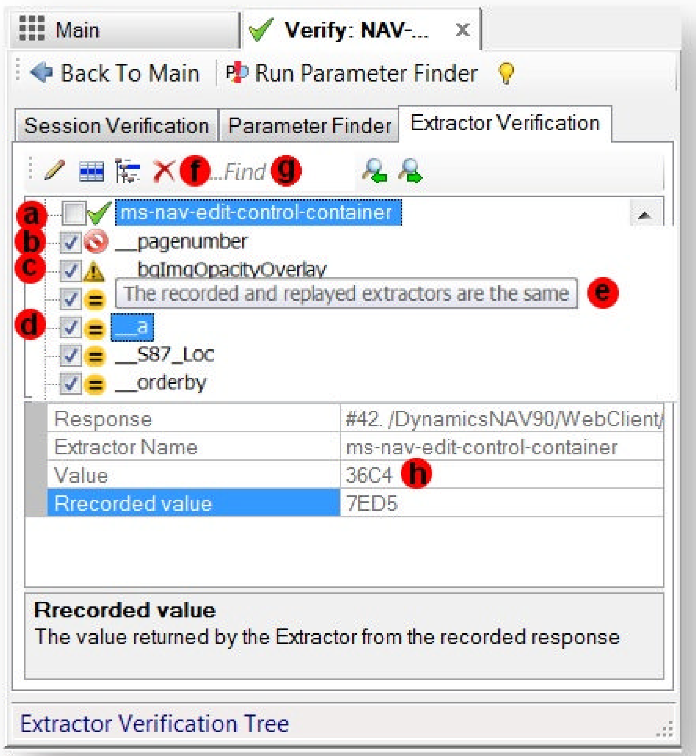
a. The extractor is OK.

b. The extractor is not found in the response.

c. The extractor's value is not used in the Test Case.

d. The recorded and replayed extractors are the same.

The status (e) is displayed when mouse over the extractor



| | |
|---|---|
| Response | #42. /DynamicsNAV90/WebClient/ |
| Extractor Name | ms-nav-edit-control-container |
| Value | 36C4 |
| Rrecorded value | 7ED5 |

**Rrecorded value**
The value returned by the Extractor from the recorded response

The autocorrelation process creates extractors and parameters to avoid test execution errors. However, some of these parameters may be unnecessary. If the test machine is overloaded, you can reduce performance overhead by deleting unnecessary parameters. The extractors with statuses b, c and d are automatically selected for easy removal. To delete them, click Delete (f) on the toolbar.

To search the Extractor Verification tree use search control (g).

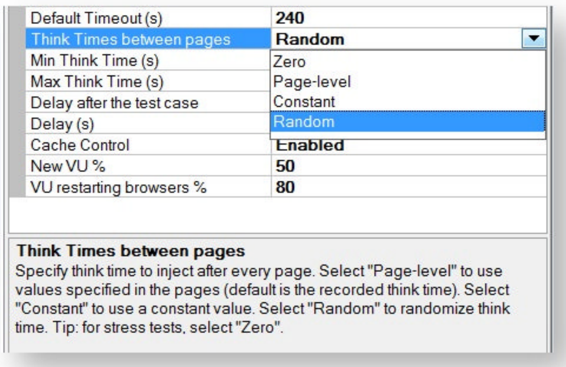
The value returned by a selected extractor is displayed in the property grid (h)

## 4.5 Test Case object properties

### 4.5.1 Test Case: Think Time

A physical user typically spends some time on each page to analyze the displayed information or enter data. The delay between opening a page and navigating to a subsequent page is called think time. Think time is an important factor in realistic load tests. You can customize page think times to emulate different users' iteration for a more realistic load test. The following think time mode options are supported.

| Think Time Mode | Description |
|---|---|
| Zero | No think time is used. This is generally **not** recommended for load tests because it results in an abnormally high load on the tested website. Use this option when a maximum load is required in stress tests. |
| Page-level | The think time specified in each page will be used. The default value is the recorded think time. When recording a test case, StresStimulus registers the time you spent on web pages and saves it in the page **Think Time** property. You can modify this property before running a test. To do so, in the **Pages Settings** tree node select a page and make a change in its property grid. |
| Constant | A constant think time will be used. Another property will appear called **Think Time (s)** where you can specify a fixed value. |
| Random | When any of the previous options is selected, all VUs will exercise the same think time for pages. |



| | |
|---|---|
| Default Timeout (s) | 240 |
| Think Times between pages | Random |
| Min Think Time (s) | |
| Max Think Time (s) | |
| Delay after the test case | |
| Delay (s) | |
| Cache Control | Enabled |
| New VU % | 50 |
| VU restarting browsers % | 80 |

**Think Times between pages**
Specify think time to inject after every page. Select "Page-level" to use values specified in the pages (default is the recorded think time). Select "Constant" to use a constant value. Select "Random" to randomize think time. Tip: for stress tests, select "Zero".

| | |
|---|---|
| | This never happens in real life. To make the test more realistic, use randomized think time for pages. When this option is selected, a random value in a given range will be used. Two properties will appear called **Min Think Time (s)** and **Max Think Time (s)** in which you can specify the think time range. |

The think time is a delay inserted before requesting a subsequent page. The think time will not be applied after the last page in the test case because there is no subsequent page. If you need to inject a delay after the last page before starting a new iteration, use Delay after the Test Case instead.

**Note:** The default think time setting is constant at 2 seconds.

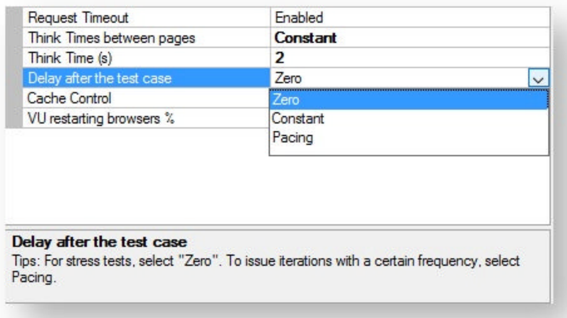
Think time is not included in the page response time. However, higher think time slows down the client request rate, which results in a lower server load. Increasing think time will typically decrease the request rate and will allow the server to handle more virtual users.

## 4.5.2 Delay after the Test Case

Typically during test run, the test case is replayed multiple times. A single replay of the test case by a single VU is called **test iteration.**

You can set a delay between iterations in using **Delay after the Test Case** property. The following options are available:

| Options | Description |
|---|---|
| Zero | There is no delay. The next iteration or test case will start as soon as the previous one is complete. |
| Constant | There is a constant delay. The next iteration or test case will start after a constant time period elapsed.<br><br>Another property will appear called **Constant delay (s)** where you can specify the delay value. |
| Pacing | This selection ensures that the execution time of a test case is equal or exceeds a given minimum duration. If necessary, a delay is automatically added to meet that requirement. Then the next iteration or next test case will start after a minimum duration has elapsed. Another property will appear called **Minimum Iteration duration(s)** where you can specify the delay value. |



| | |
|---|---|
| Request Timeout | Enabled |
| Think Times between pages | **Constant** |
| Think Time (s) | **2** |
| Delay after the test case | Zero |
| Cache Control | Zero |
| VU restarting browsers % | Constant |
| | Pacing |

**Delay after the test case**
Tips: For stress tests, select "Zero". To issue iterations with a certain frequency, select Pacing.

When the test cases are executed sequentially, they are combined in the test case groups. **Delay after the Test Case** property is used to set the wait time between the test case executions in the group.

## 4.5.3 Test Case: Caching Rules

HTTP caching is one of the most important mechanisms of increasing website performance. It allows to substantially reduce bandwidth and offload the server from sending resources cached on the client. Users with empty cache and users with primed cache impact the Web servers load differently. While HTTP caching is an essential factor impacting server load, it is frequently overlooked in load testing.

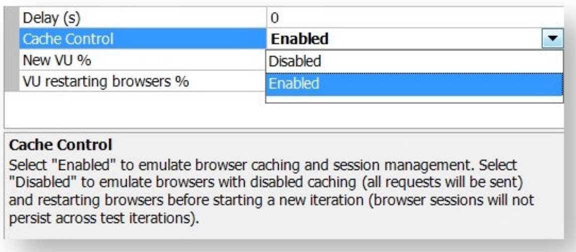
StresStimulus provides a comprehensive mechanism of emulating the browser caching factor. The **Cache Control** property allows to configure a mix of virtual users with different browsing history (cache) management. Every VU can either have cache enabled or disabled.

- A VU with **disabled browser cache** will send all recorded requests on all iterations without any HTTP caching headers. For every request, the server is expected to send a full response.

- A VU with **enabled browser cache** will send requests based on the caching rules. These rules are determined for every request after recording a test case and can be changed as described below.

There are 2 flavors of a VU with enabled cache: new user and returning user.

- **New users** emulate a VU with browser cache enabled, but empty (like visiting a site for the first time). Such users behave differently on the first and subsequent iterations:
- In the first iteration caching rules are ignored and all requests are unconditionally sent without HTTP caching headers (like a VU with disabled cache),
- In all subsequent iterations, the HTTP caching rules are used.
- **Returning users** emulate a VU with primed browser cache and will follow the caching rules on all iterations.



| | |
|---|---|
| Delay (s) | 0 |
| Cache Control | Enabled |
| New VU % | |
| VU restarting browsers % | |

**Cache Control**
Select "Enabled" to emulate browser caching and session management. Select "Disabled" to emulate browsers with disabled caching (all requests will be sent) and restarting browsers before starting a new iteration (browser sessions will not persist across test iterations).

**Note:** The caching rules are determined by the server's recorded response headers. For information about browser caching rules, see RFC 2616, Caching in HTTP.

Disabling the cache control will render all VUs with disabled browser caches. Enabling the browser cache will render all VUs with enabled browser caches. In this case the **New VU %** property will appear. It is used to emulate the proportion of new vs. returning users. Cache control properties are summarized in the table below:

| Property | Value | Description |
|---|---|---|
| Cache Control | Disabled | Every VU will not emulate browser cache capabilities. Every request will be sent, and every request will have recorded caching headers removed. |
| Cache Control | Enabled | Every VU will emulate browser cache capabilities. Every VU can emulate storing resources in the browser cache for further subsequent iterations.<br>See below for further options. |

| Property | Value | Description |
|---|---|---|
| New VU% | A number between 0 and 100% | The percent of VUs that will be new with cache enabled. The rest will be returning with cache enabled. This setting is only available with Cache Control enabled. |

**Note:** Cache properties described in this section are not available on the Test Case level if Test Case Groups are used. In this case, these properties are available in the Test Case Group section (see Sequential Test Case Groups).

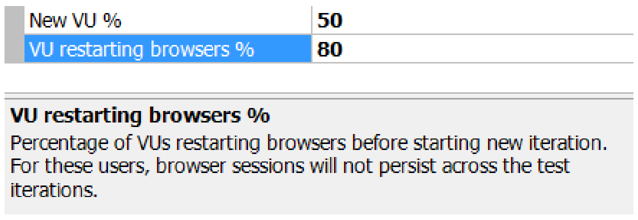
## 4.5.4 Test Case: Session Persistence

A physical browser that accesses a website maintains application session persistence until it is closed. After the browser is restarted and navigated to the same website, a new application session will be established. A website session is controlled by one (or more) HTTP session cookies, so a new session means a new value of the cookie. In StresStimulus, a VU can either restart a browser on every iteration (the session cookie will change on every iteration), or leave the browser open across all iterations (the session cookie will stay the same on every iteration). Maintaining session persistence across all iterations can be useful when a test case has a login process that needs to be done only on the first iteration and not on subsequent iterations.

Here are two examples:

- Internal users are signing in to the application in the morning and maintain the same session while navigating through the scenario many times throughout the day. The server renders a session cookie on the first iteration, and StresStimulus will reuse it in subsequent iterations. A login is not required in subsequent iterations.
- On the contrary, external users who use the application occasionally are signing in every time when they need to navigate through a scenario. A new session cookie is assigned in every iteration. A login is most likely required in every iteration as well.

StresStimulus can emulate a user base that includes both of these user types. Use the **VUs restarting the browser %** to set the percentage of VUs that will be restarting the browser on every iteration and change the session. The rest will keep their browsers open and have their session persist across all iterations. For example, a value of 0 for **VUs restarting the browser %** will render session persistence for all VUs across all iterations; a value of 80 will disable session persistence across iterations for 80% of VUs and enable it for 20% of VUs.



| | |
|---|---|
| New VU % | 50 |
| VU restarting browsers % | 80 |

**VU restarting browsers %**
Percentage of VUs restarting browsers before starting new iteration. For these users, browser sessions will not persist across the test iterations.

**Note:** Session persistence within an iteration is always maintained.

## 4.5.5 Page Properties

Some of the page properties are described in the next several sections.

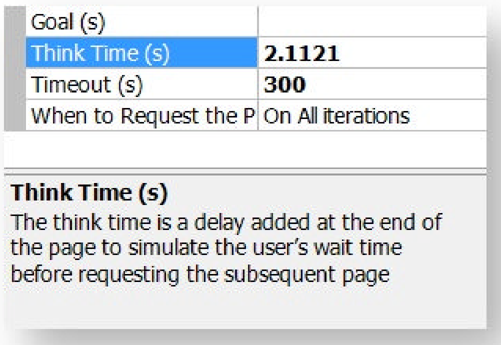
**Info:** All page properties are displayed in the property grid. You can modify any property which is not grayed out.

The full list of page properties, toolbar commands and context menu options is provided in the User Interface Reference -> Test Case Tree -> Page.

### 4.5.5.1 Think Time

Configuration of the think time in the Test Case is described in Test Case: Think Time. The page's **Think Time** displays an actual time interval in seconds. This is the wait time that will be used after the page is complete if you set **Think Time Mode** to **Page-level** in the **Test Case**.

**Note:** After recording a Test Case it displays the recorded think time, which is the delay between the completion of the current page and the requesting of the subsequent page during recording. After recording you can change this property.



| | |
|---|---|
| Goal (s) | |
| Think Time (s) | 2.1121 |
| Timeout (s) | 300 |
| When to Request the P | On All iterations |

**Think Time (s)**
The think time is a delay added at the end of the page to simulate the user's wait time before requesting the subsequent page

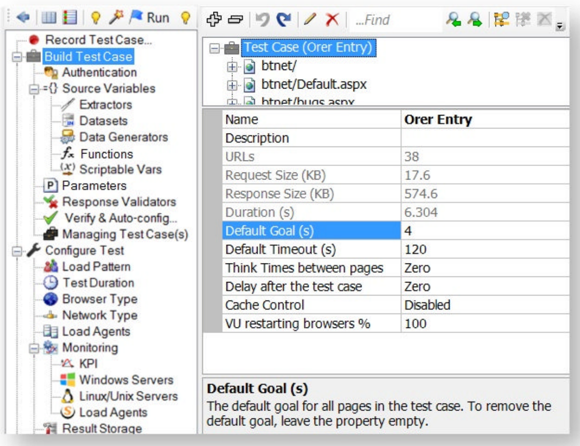
### 4.5.5.2 Page Goal

A page goal is the expected maximum time for all the responses in a page to come back. If the page's actual response time in one of the test iterations exceeds the goal, then the page's **Missed the Goal** counter is incremented. After the test is complete, the percentage of the page requests that missed the goal is reported in the **Missed Goal %** column in the **Page Details** section. Also, **Goal** and/or **Missed Goal** curves will be plotted on the graphs in the Page Result tab.

To set or change the page goal use the following information:

**1. Default goal**

- The initial **Default Goal** value is set to 4 seconds for all pages.
- If you change the test case **Default Goal (s)** property then the goal of every page for which the **Goal (s)** property was not modified, will be equal to the new default goal.
- To remove the goal in all pages with default goal, leave the **Default Goal (s)** value empty.

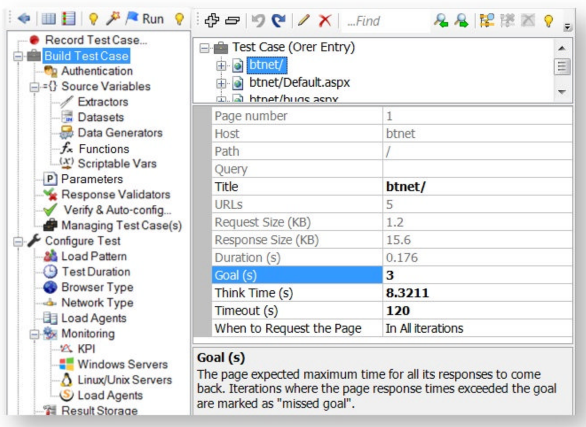


| | |
|---|---|
| Name | Orer Entry |
| Description | |
| URLs | 38 |
| Request Size (KB) | 17.6 |
| Response Size (KB) | 574.6 |
| Duration (s) | 6.304 |
| Default Goal (s) | 4 |
| Default Timeout (s) | 120 |
| Think Times between pages | Zero |
| Delay after the test case | Zero |
| Cache Control | Disabled |
| VU restarting browsers % | 100 |

**Default Goal (s)**
The default goal for all pages in the test case. To remove the default goal, leave the property empty.

**2. Individual page goal.** To override the page default goal value, change the **Goal(s)** property in the property grid.

It is recommended to select the goal relevant to your application quality of service requirements.



| | |
|---|---|
| Page number | 1 |
| Host | btnet |
| Path | / |
| Query | |
| Title | btnet/ |
| URLs | 5 |
| Request Size (KB) | 1.2 |
| Response Size (KB) | 15.6 |
| Duration (s) | 0.176 |
| Goal (s) | 3 |
| Think Time (s) | 8.3211 |
| Timeout (s) | 120 |
| When to Request the Page | In All iterations |

**Goal (s)**
The page expected maximum time for all its responses to come back. Iterations where the page response times exceeded the goal are marked as "missed goal".

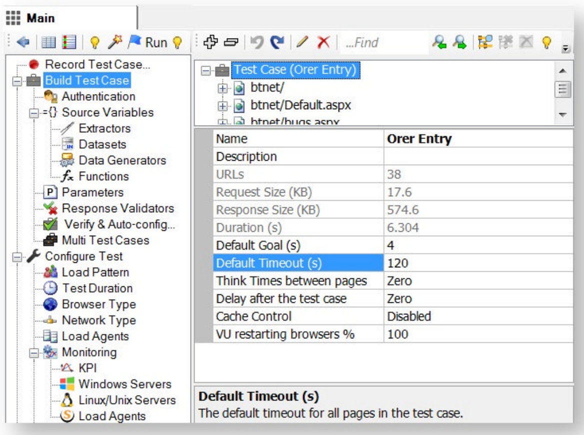
### 4.5.5.3 Page Timeout

Timeout is the maximum amount of time for receiving any of the page responses. Like in physical browsers, a virtual user will stop sending new requests after the number of its pending requests reaches a certain browser-specific limit. The VU will resume sending requests only after some responses are received, so the number of open connections does not exceed the limit. If the server will stop responding, the load test will halt. To avoid this situation, StresStimulus has a timeout mechanism allowing the Tests with very slow responses to complete faster. Any request which does not receive a response after the timeout, will be aborted and marked as a Timeout, so the load test can continue without waiting such very slow responses. All timeout occurrences are reported in the test error log. The timeout sessions are excluded from response time calculations.

To set or change page timeout use the following information:

**1. Default timeout**

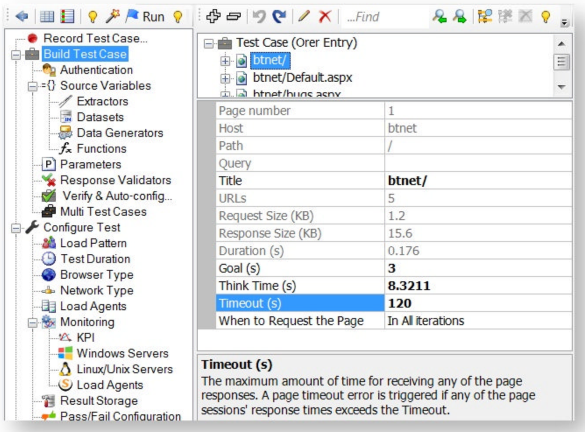
- The initial default timeout value is 60 seconds for all pages.
- If you change the test case **Default Timeout(s)** property (a) then the timeout of every page for which the **timeout (s)** property was not modified, will be equal to the new default timeout.



| Name | Orer Entry |
|---|---|
| Description | |
| URLs | 38 |
| Request Size (KB) | 17.6 |
| Response Size (KB) | 574.6 |
| Duration (s) | 6.304 |
| Default Goal (s) | 4 |
| Default Timeout (s) | 120 |
| Think Times between pages | Zero |
| Delay after the test case | Zero |
| Cache Control | Disabled |
| VU restarting browsers % | 100 |

**Default Timeout (s)**
The default timeout for all pages in the test case.

**2. Individual page timeout.** To override the page default timeout value, change its **Timeout(s)** property in the property grid. The new value will set the timeout of every request in this page. You can further change the individual requests timeout as described in the **Request** properties.



| Page number | 1 |
|---|---|
| Host | btnet |
| Path | / |
| Query | |
| Title | btnet/ |
| URLs | 5 |
| Request Size (KB) | 1.2 |
| Response Size (KB) | 15.6 |
| Duration (s) | 0.176 |
| Goal (s) | 3 |
| Think Time (s) | 8.3211 |
| Timeout (s) | 120 |
| When to Request the Page | In All iterations |

**Timeout (s)**
The maximum amount of time for receiving any of the page responses. A page timeout error is triggered if any of the page sessions' response times exceeds the Timeout.

**Note:** You can disable Timeouts for certain test cases. To do so, in the test case property grid, change the **Request Timeout** property to **Disabled**. Use Disabled for determining response times of slow requests. However, this may cause long or indefinite wait times during the test run.

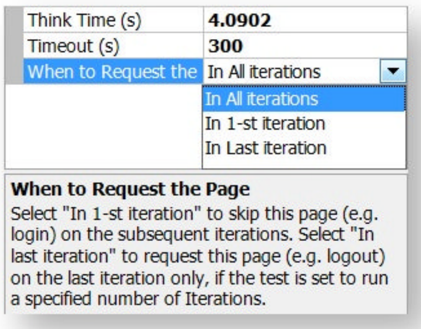
### 4.5.5.4 When to Request the Page

Sometimes in tests executing multiple iterations, it is desirable to request a page only once, in the beginning or at the end, to perform initial or final actions. For instance, in a test case, a user logs in, performs some actions, and logs out. During the test run, you want every VU to login only on the first iteration and logout only on the last iteration, while other pages should be repeated multiple times.

Use the **When to Request the Page** property to configure such tests. Set this property to **On 1-st Iteration** on the login page and to **On Last iteration** on the logout page.

The table below shows how to use this property.

| When to Request the Page | Properties |
|---|---|
| In All iterations | (Default) The page and all its requests will be sent in all VU's iterations. |
| In 1-st iteration | The page and all its requests will be sent only in the first iteration. This is used for a log-in type page. |
| In Last iteration | The page and all its requests will be sent only in the last iteration. This is used for a log-out type page. |



| | |
|---|---|
| Think Time (s) | 4.0902 |
| Timeout (s) | 300 |
| When to Request the | In All iterations |

**When to Request the Page**
Select "In 1-st iteration" to skip this page (e.g. login) on the subsequent iterations. Select "In last iteration" to request this page (e.g. logout) on the last iteration only, if the test is set to run a specified number of Iterations.

**Note:**

- Actions executed on the 1st iteration, create certain state that must be carried over throughout the entire test. For example, the log-in process on the 1st iteration generates a session ID that should persist throughout the test. To satisfy this requirement, enable Cache Control and set the VU restarting browsers % to 0. For more information see Test Case: Caching Rules and Test Case: Session Persistence.

- If you are running the log-out process on the last iteration only (as in the example above) your test must set **Test Completion Criteria** to **Number of iterations.**

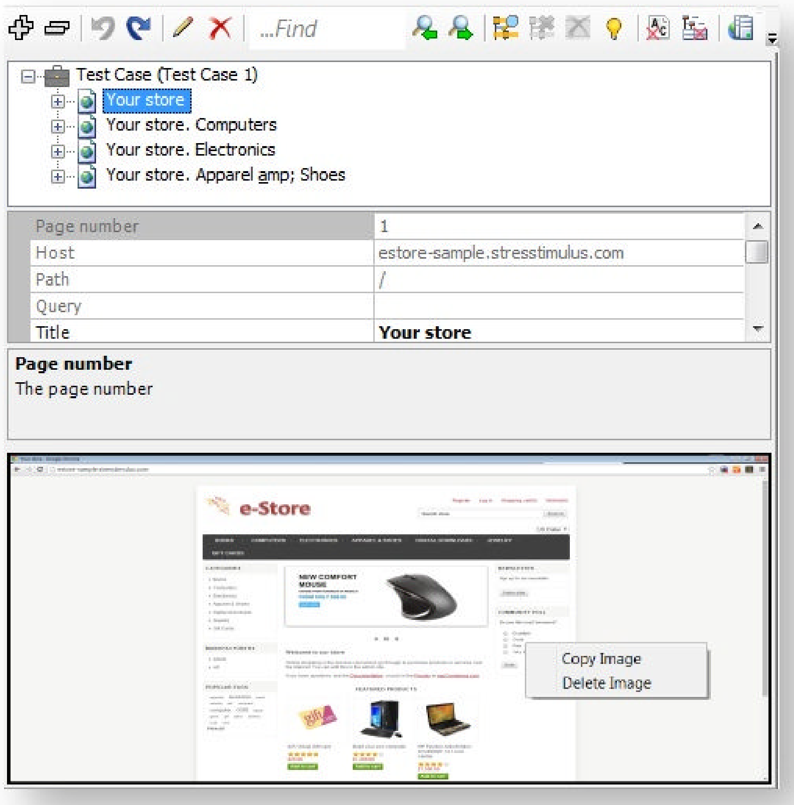
## 4.5.6 Request Properties

Ent SP

During recording, StresStimulus automatically takes screenshots of the webpage on every click. Each screenshot is associated with a primary request, page or transaction issued immediately before the click. It is stored within a corresponding page object.

The screenshots are displayed when you select the request on the Test Case tree. They help users remember what was displayed in the browser when the web page was recorded. Screenshots are only available for sessions representing an HTML page. The screenshots are displayed below the page property grid. You can disable taking screenshot by un-checking the box **Take screenshot of**

**pages** as described in Recording Test Case. You can copy the screenshot to clipboard or delete it by right-clicking the image.



| | |
|---|---|
| Page number | 1 |
| Host | estore-sample.stresstimulus.com |
| Path | / |
| Query | |
| Title | **Your store** |

**Page number**
The page number

**Note:** StresStimulus tries to capture how the browser window looked just before the click. Sometimes when the page refreshes too quickly or the system is a bit slow, StresStimulus can miss the right moment for a screenshot, so they may not come out well or will miss altogether.

Some of the request properties are described in this section.

**Info:** The full list of request properties, toolbar commands and context menu options is provided in the User Interface Reference -> Test Case Tree.

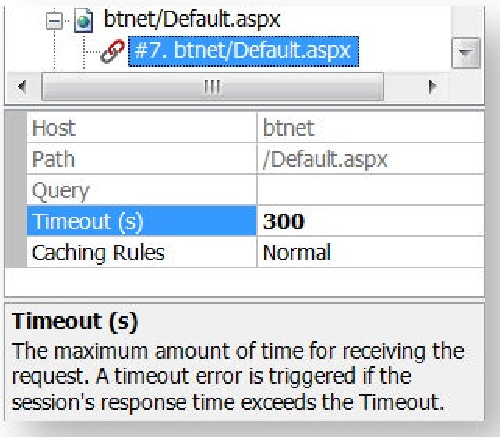
**Timeout**

Request Timeouts - the maximum amount of time for receiving the request. Initially, the timeout of every page's request is equal to the timeout property set on the pages (see Page Timeout). You can override the request timeout on its property grid.

**Info:** Because responses for the timed out sessions are not received in time, they cannot be stored in the test log. Session inspector will display the message "StresStimulus timeout" in place of the response content when you open a session with a timeout.

```
HTTP/1.1 502 StresStimulus Timeout


```



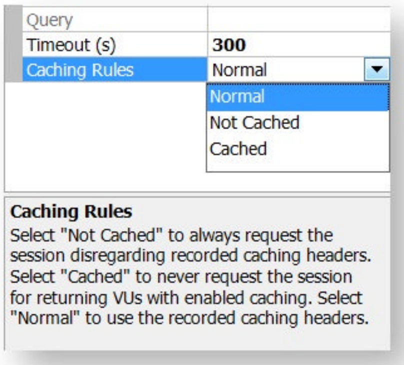
btnet/Default.aspx
#7. btnet/Default.aspx

| | |
|---|---|
| Host | btnet |
| Path | /Default.aspx |
| Query | |
| Timeout (s) | 300 |
| Caching Rules | Normal |

**Timeout (s)**
The maximum amount of time for receiving the request. A timeout error is triggered if the session's response time exceeds the Timeout.

Caching Rules

Caching emulation in StresStimulus is described in Test Case: Caching Rules. Every request has recorded caching behavior that is a result of the server's response headers. This information is stored in the request's **Caching Rules** property. You can view or change this property on the request property grid. This property only pertains to test cases with **Cache Control** enabled. The **Caching Rules** property is summarized in the table below.



| | |
|---|---|
| Query | |
| Timeout (s) | 300 |
| Caching Rules | Normal |

**Caching Rules**
Select "Not Cached" to always request the session disregarding recorded caching headers. Select "Cached" to never request the session for returning VUs with enabled caching. Select "Normal" to use the recorded caching headers.

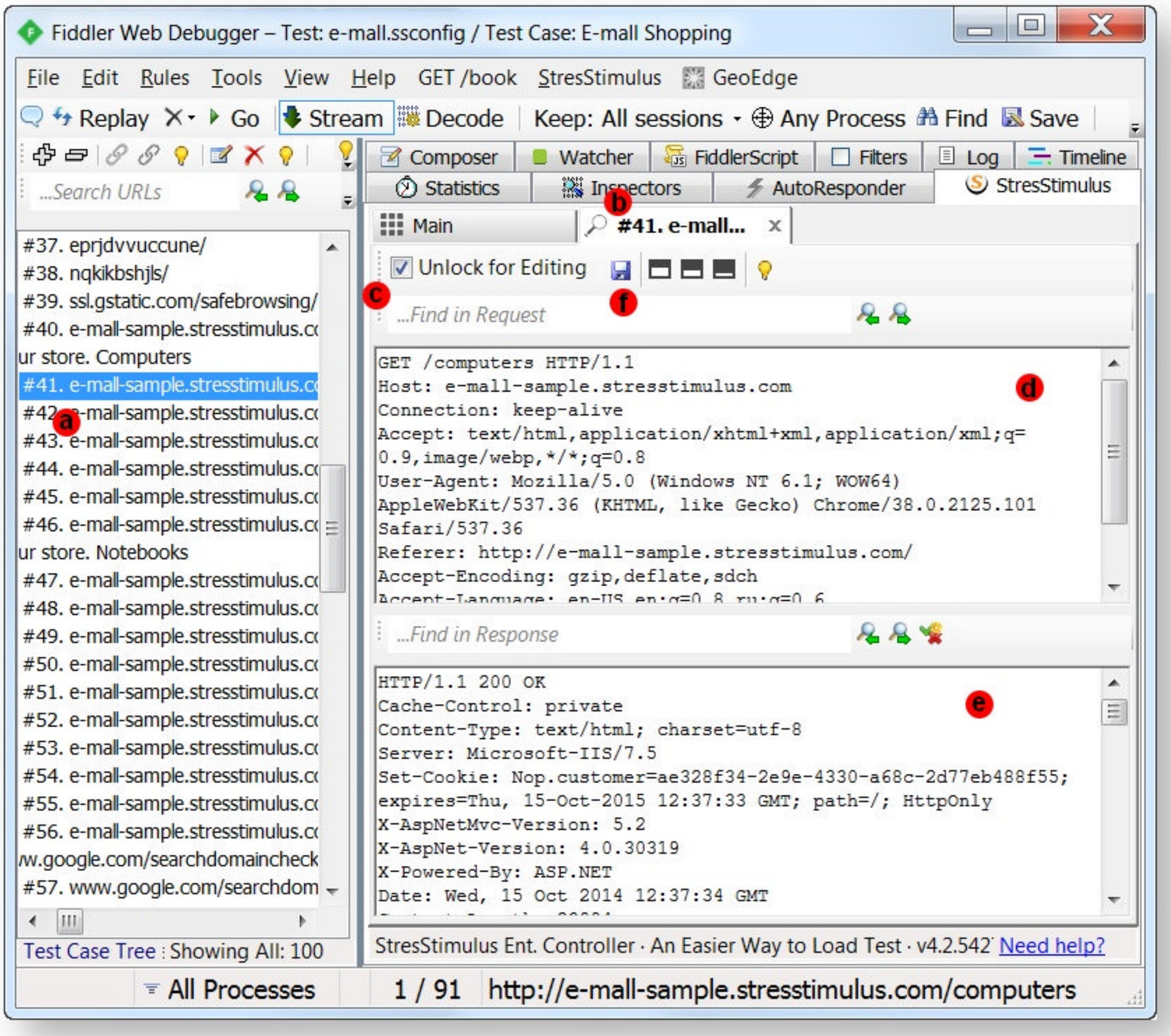
| Caching Rules | Properties |
|---|---|
| Normal | All recorded caching headers will be sent. A possible 304 Response can come if the resource has not changed. |
| Not Cached | All recorded caching headers are stripped off and the request is sent. Expect a 200 response. |

| | |
|---|---|
| Cached | This request will not be sent because it is cached. |

**Note:** If the URL of a request is parameterized, then caching for such requests is automatically disabled regardless of user settings. This helps to avoid the mistake of caching requests with dynamic URLs.

Editing Sessions

You can edit recorded sessions in the test case. To do so, in the test case tree, double-click the session (a), and in the appeared in a new Tab (b) session inspector, check “unlock for editing” (c). After that you can make changes in the session request (d)and response (e). When finished, click **Save** (f)


Fiddler Web Debugger – Test: e-mall.ssconfig / Test Case: E-mall Shopping
File Edit Rules Tools View Help GET /book StresStimulus GeoEdge
Replay Go Stream Decode Keep: All sessions Any Process Find Save
Composer Watcher FiddlerScript Filters Log Timeline
Statistics Inspectors AutoResponder StresStimulus
...Search URLs
#37. eprjdvvuccune/
#38. nqkikbshjls/
#39. ssl.gstatic.com/safebrowsing/
#40. e-mall-sample.stresstimulus.co
ur store. Computers
#41. e-mall-sample.stresstimulus.co
#42. e-mall-sample.stresstimulus.co
#43. e-mall-sample.stresstimulus.co
#44. e-mall-sample.stresstimulus.co
#45. e-mall-sample.stresstimulus.co
#46. e-mall-sample.stresstimulus.co
ur store. Notebooks
#47. e-mall-sample.stresstimulus.co
#48. e-mall-sample.stresstimulus.co
#49. e-mall-sample.stresstimulus.co
#50. e-mall-sample.stresstimulus.co
#51. e-mall-sample.stresstimulus.co
#52. e-mall-sample.stresstimulus.co
#53. e-mall-sample.stresstimulus.co
#54. e-mall-sample.stresstimulus.co
#55. e-mall-sample.stresstimulus.co
#56. e-mall-sample.stresstimulus.co
w.google.com/searchdomaincheck
#57. www.google.com/searchdom
Test Case Tree : Showing All: 100
Main
#41. e-mall...
Unlock for Editing
...Find in Request
...Find in Response
a
b
c
d
e
f
StresStimulus Ent. Controller · An Easier Way to Load Test · v4.2.542 Need help?
All Processes
1 / 91
http://e-mall-sample.stresstimulus.com/computers

```
GET /computers HTTP/1.1
Host: e-mall-sample.stresstimulus.com
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=
0.9,image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/38.0.2125.101
Safari/537.36
Referer: http://e-mall-sample.stresstimulus.com/
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8,ru;q=0.6
```

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
Server: Microsoft-IIS/7.5
Set-Cookie: Nop.customer=ae328f34-2e9e-4330-a68c-2d77eb488f55;
expires=Thu, 15-Oct-2015 12:37:33 GMT; path=/; HttpOnly
X-AspNetMvc-Version: 5.2
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Wed, 15 Oct 2014 12:37:34 GMT
```

## 4.6 Other Test Case Elements

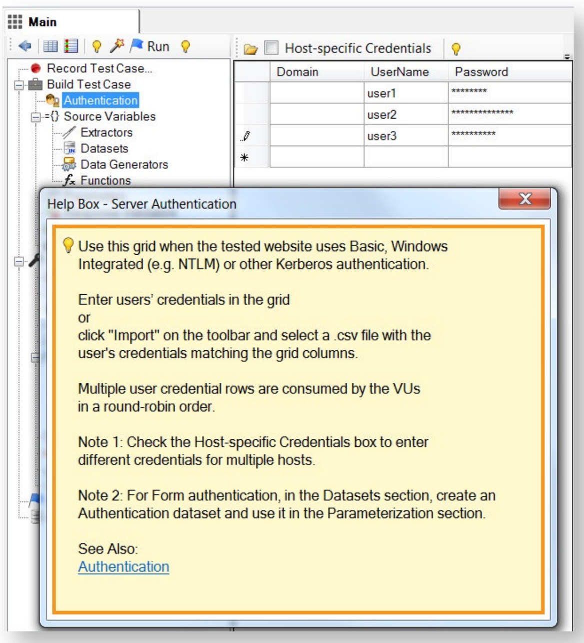
### 4.6.1 Authentication

StresStimulus supports all major authentication methods. They are divided into two types: server authentication and application authentication.

## Server Authentication

Server authentication refers to any machine, operating system or domain level authentication. It includes Basic, Windows Integrated (e.g. NTLM) or other Kerberos authentication.

To configure server authentication, provide a set of the credentials that the tested website recognizes, and use the **Authentication** node in the **Workflow Tree** to enter the provided credentials. You can also paste data (from Excel) into the authentication grid. The **Domain** field might be optional depending on your server. You can also import credentials stored in a .csv file by clicking **Import** on the toolbar. The .csv file must have 3 grid columns and no header.



| Domain | UserName | Password |
|---|---|---|
| | user1 | ******** |
| | user2 | ************** |
| | user3 | ********** |
| | | |

Use this grid when the tested website uses Basic, Windows Integrated (e.g. NTLM) or other Kerberos authentication.

Enter users' credentials in the grid
or
click "Import" on the toolbar and select a .csv file with the user's credentials matching the grid columns.

Multiple user credential rows are consumed by the VUs in a round-robin order.

Note 1: Check the Host-specific Credentials box to enter different credentials for multiple hosts.

Note 2: For Form authentication, in the Datasets section, create an Authentication dataset and use it in the Parameterization section.

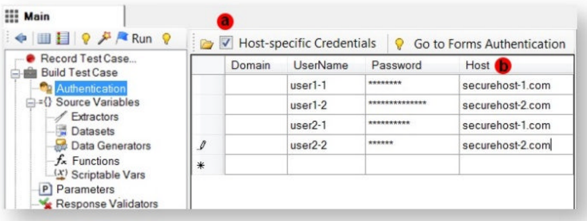
See Also:
Authentication

**Note 1:** If you are located in the Authentication section and need to configure form authentication or any other type of application authentication, described below, click **Go to Forms Authentication** on the toolbar.

**Note 2:** Every new VU will use a subsequent set of credentials. If the number of VUs exceeds the number of credentials, then they are assigned using a round robin algorithm. For example, if you have 10 rows in the Authentication grid and emulate 20 VUs, then VU1 and VU11 will use row 1, VU2 and VU12 will use row 2, and so on.

Authentication is configured per Test. Credentials created in one test case are used in all test cases

Some web applications use several hosts which require different credentials for authentication. In order to support such authentication schema, you need to enable host specific credentials. To do so, check the Host-specific Credentials box (a). The Host column (b) appears in the authentication grid. For every set of credentials enter a host to which the credentials will be submitted.

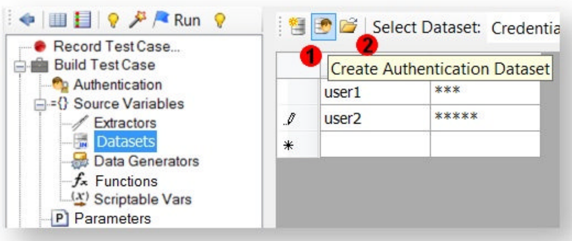


| | Domain | UserName | Password | Host |
|---|---|---|---|---|
| | | user1-1 | ******** | securehost-1.com |
| | | user1-2 | ************** | securehost-2.com |
| | | user2-1 | ********** | securehost-1.com |
| | | user2-2 | ****** | securehost-2.com |
| * | | | | |

## Application Authentication

Application level authentication refers to the authentication method that takes place inside the web application (e.g. Form authentication). The test case will store the set of credentials entered during recording. By default, these credentials will be used for all VUs. To test VUs with different credentials, you need to:

- Provide a set of the credential that the tested website recognizes.
- Create a dataset, populate it with the provided credentials, and parameterize the appropriate request as described below:



| | | |
|---|---|---|
| | user1 | *** |
| | user2 | ***** |
| * | | |

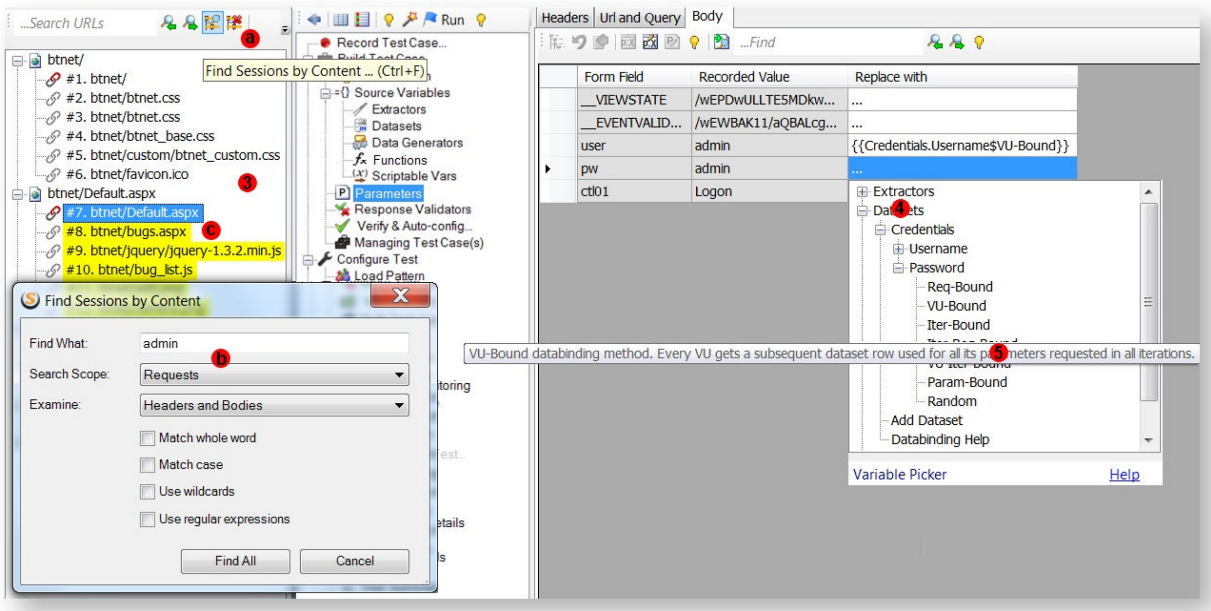
1. In the **Datasets** section, click Create Authentication Dataset on the toolbar. The **Credentials** dataset will be created. If your authentication process includes additional properties, such as security questions, you can edit the **Credentials** dataset structure by adding additional fields (see Datasets ).

2. Populate the authentication dataset by entering data, pasting data (from Excel) or importing a .csv file.

3. Find the login request in the test case (it is usually one of the first POST requests). To do so:

a. Click **Find Session by Content** or hit <Crl+F>

| | |
|---|---|
| b. Enter one of the credentials you used in recording (e.g. a username, email address or password).<br><br>c. The first highlighted session is a login request. Select it. | |



btnet/
#1. btnet/
#2. btnet/btnet.css
#3. btnet/btnet.css
#4. btnet/btnet_base.css
#5. btnet/custom/btnet_custom.css
#6. btnet/favicon.ico
btnet/Default.aspx
#7. btnet/Default.aspx
#8. btnet/bugs.aspx
#9. btnet/jquery/jquery-1.3.2.min.js
#10. btnet/bug_list.js

Find Sessions by Content

Find What: admin
Search Scope: Requests
Examine: Headers and Bodies
☐ Match whole word
☐ Match case
☐ Use wildcards
☐ Use regular expressions

Find All | Cancel

| Form Field | Recorded Value | Replace with |
|---|---|---|
| __VIEWSTATE | /wEPDwULLTE5MDkw... | ... |
| __EVENTVALID... | /wEWBAK11/aQBALcg... | ... |
| user | admin | {{Credentials.Username$VU-Bound}} |
| pw | admin | ... |
| ctl01 | Logon | |

VU-Bound databinding method. Every VU gets a subsequent dataset row used for all its parameters requested in all iterations.

**Tip:** If you cannot find a request recorded with credentials, it is likely that your application uses server authentication (see above).

4. Parameterize the **Credential** parameters using the **Credentials** dataset. Use the **VU-Bound** databinding method.

**Tip:** Typically credentials are submitted in web form. In this case they will appear in the parameterization grid in the **Body** tab.

**Note:** The Credentials are distributed between VUs using a round robin algorithm.

### 4.6.2 Transactions

Ent SP

A transaction is a set of sequential requests representing a meaningful step in a test scenario. It is used to track performance characteristics of time-critical business transactions consisting of several user actions. Transactions add another level of performance tracking in addition to requests and pages.

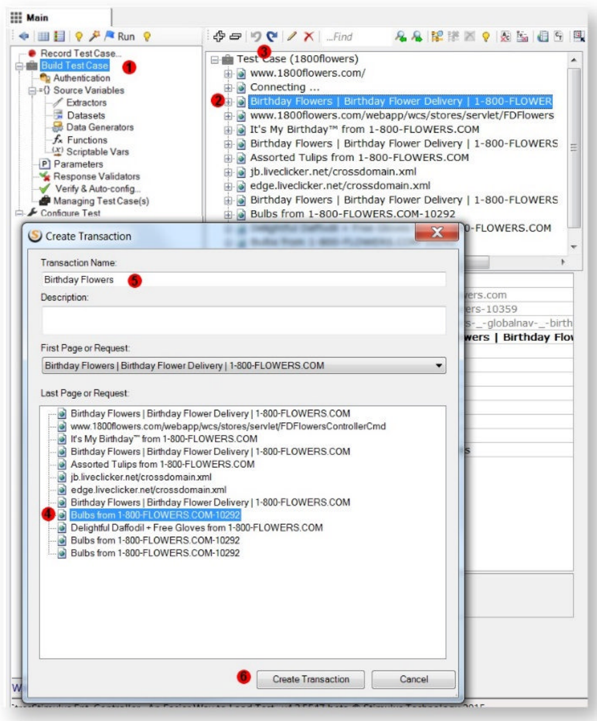
You can define transactions while recording a test case, as described in the Creating Transactions section, or after the test case is created.

To define a transaction after the test case is created:

1. Go to the **Build Test Case** node.
2. Select the starting page or top level request.
3. Click **Create a Transaction**. A new transaction dialog will appear.
4. Select the last request in the transaction.
5. Give it a meaningful name and description (optional).
6. Click **Create Transaction**.

**Transaction Goal** (a) property is a transaction completion time limit. Transactions which missed the goal are reported in the **Test Results -> Transaction Details** section. Transaction goal is not set by default.

**Transaction Think Time** (b) is a delay added at the end of the transaction to simulate the user's wait time before requesting the subsequent page. Transaction think time is not set by default.



Test Case (1800flowers)
www.1800flowers.com/
Connecting ...
Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWER
www.1800flowers.com/webapp/wcs/stores/servlet/FDFlowers
It's My Birthday™ from 1-800-FLOWERS.COM
Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWERS
Assorted Tulips from 1-800-FLOWERS.COM
jb.liveclicker.net/crossdomain.xml
edge.liveclicker.net/crossdomain.xml
Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWERS
Bulbs from 1-800-FLOWERS.COM-10292

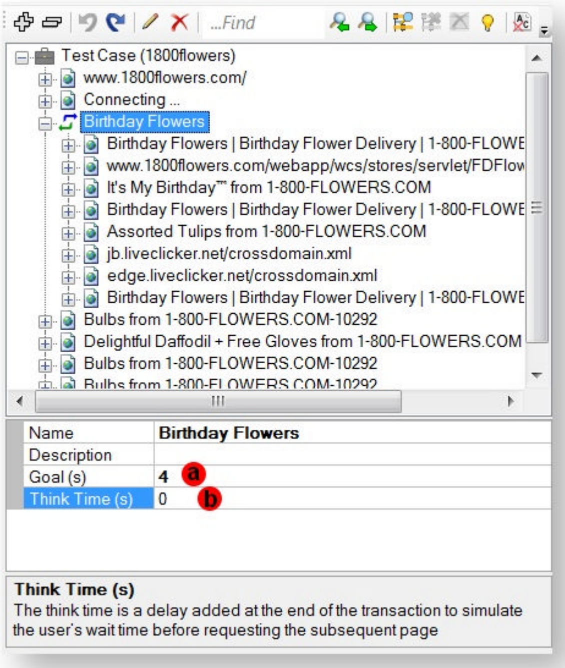
Create Transaction

Transaction Name: Birthday Flowers

Description:

First Page or Request: Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWERS.COM

Last Page or Request:
Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWERS.COM
www.1800flowers.com/webapp/wcs/stores/servlet/FDFlowersControllerCmd
It's My Birthday™ from 1-800-FLOWERS.COM
Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWERS.COM
Assorted Tulips from 1-800-FLOWERS.COM
jb.liveclicker.net/crossdomain.xml
edge.liveclicker.net/crossdomain.xml
Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWERS.COM
Bulbs from 1-800-FLOWERS.COM-10292
Delightful Daffodil + Free Gloves from 1-800-FLOWERS.COM
Bulbs from 1-800-FLOWERS.COM-10292
Bulbs from 1-800-FLOWERS.COM-10292



Test Case (1800flowers)
- www.1800flowers.com/
- Connecting ...
- Birthday Flowers
  - Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWE
  - www.1800flowers.com/webapp/wcs/stores/servlet/FDFlow
  - It's My Birthday™ from 1-800-FLOWERS.COM
  - Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWE
  - Assorted Tulips from 1-800-FLOWERS.COM
  - jb.liveclicker.net/crossdomain.xml
  - edge.liveclicker.net/crossdomain.xml
  - Birthday Flowers | Birthday Flower Delivery | 1-800-FLOWE
- Bulbs from 1-800-FLOWERS.COM-10292
- Delightful Daffodil + Free Gloves from 1-800-FLOWERS.COM
- Bulbs from 1-800-FLOWERS.COM-10292
- Bulbs from 1-800-FLOWERS.COM-10292

| | |
|---|---|
| Name | Birthday Flowers |
| Description | |
| Goal (s) | 4 a |
| Think Time (s) | 0 b |

**Think Time (s)**
The think time is a delay added at the end of the transaction to simulate the user's wait time before requesting the subsequent page

**Note:** After the test is complete, transaction performance metrics are located in **Transaction Details** section under the test results.

**Note:** Think time of a transaction and think times of all children transactions are excluded from the transaction response time.

**Info:** The full list of Transaction properties, toolbar commands and context menu options is provided in User Interface Reference -> Transactions & Loops.

### 4.6.3 Loops

Ent SP

A Loop will execute its child objects multiple times within a test case iteration. Loops allow to simplify recording test scenarios consisting of repeated actions.

To create a loop, perform the following steps:

1. Go to the **Build Test Case** node.

2. Select the starting page, top level request, or transaction

3. Open the Context Menu by right mouse clicking.

4. Click **Create Loop**.

5. A new loop dialog will appear.

6. Select the last request or transaction in the loop.

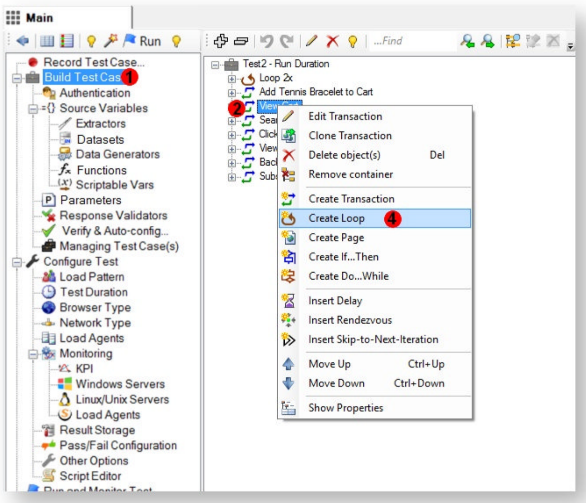
7. Give it a meaningful description (optional).

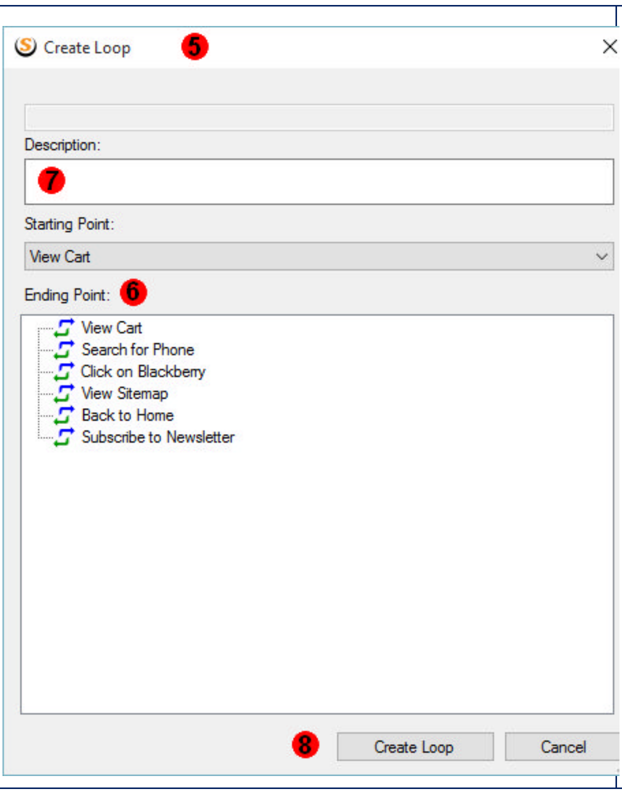
8. Click **Create Loop**. A basic loop will be created.To change the loop definition, click **Edit** on the toolbar.

9. Re-open the Context Menu by right mouse clicking and click Show Properties.

10. set the **Number of Repeats** property to the required number of loop iterations

11. Optionally, specify in the **Delay before next Loop** property, the number of seconds that will be injected before starting the next loop cycle.





Create Loop 5

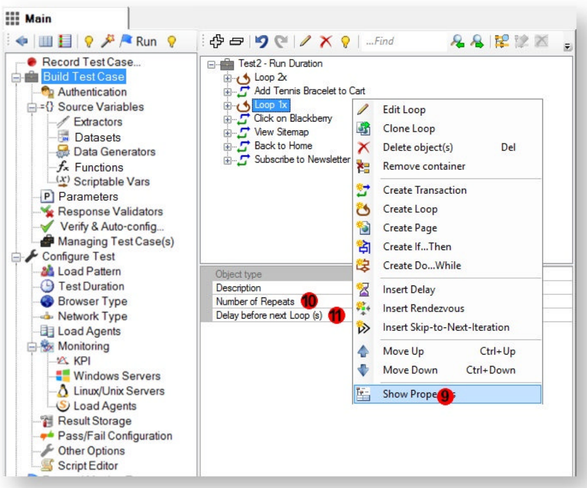
Description: 7

Starting Point: View Cart

Ending Point: 6

- View Cart
- Search for Phone
- Click on Blackberry
- View Sitemap
- Back to Home
- Subscribe to Newsletter

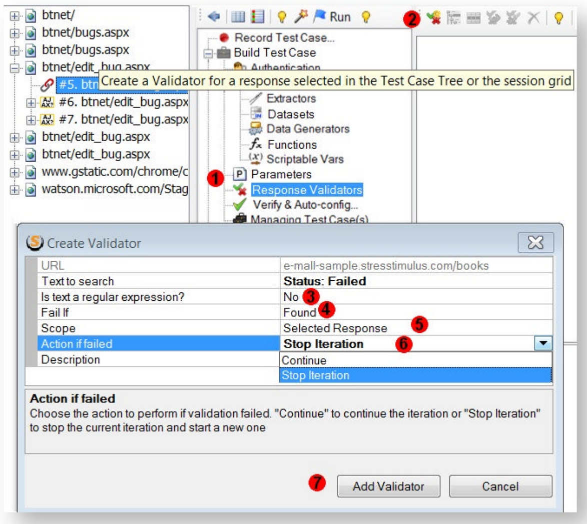
8 Create Loop | Cancel



| | |
|---|---|
| Object type | |
| Description | |
| Number of Repeats 10 | |
| Delay before next Loop (s) 11 | |

## 4.6.4 Response Validators

StresStimulus automatically detects HTTP errors by checking response status codes of the emulated traffic. All responses with the status code of 400 and above are listed as errors in the **Error Details** report. However, this level of verification is insufficient to track down application-specific errors. For this purpose, StresStimulus supports Validators that allow to compare responses against some expected values. Also, validators take precedence over the HTTP errors to make error reporting as granular as necessary. For example, one custom validator will recognize a database error message in response with status code 500, and another one will recognize a web server error message in response with status code 500. As a result, 500 responses with database error, 500 responses with Web server error and 500 responses with other types of errors will be



Create a Validator for a response selected in the Test Case Tree or the session grid

Create Validator

| | |
|---|---|
| URL | e-mall-sample.stresstimulus.com/books |
| Text to search | **Status: Failed** |
| Is text a regular expression? | No 3 |
| Fail If | Found 4 |
| Scope | Selected Response 5 |
| Action if failed | **Stop Iteration** 6 |
| Description | |

Continue
Stop Iteration

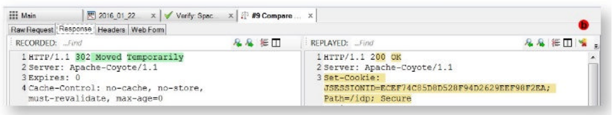
**Action if failed**
Choose the action to perform if validation failed. "Continue" to continue the iteration or "Stop Iteration" to stop the current iteration and start a new one

7 Add Validator | Cancel

| | |
|---|---|
| reported as 3 different error types.<br><br>A Validator is a rule of comparing a server's response with a text pattern. In the case of a mismatch, a custom error is raised. You can create validators for a single response (local), or all responses (global). To create a new validator:<br><br>1. Go to the **Validators** node on the **Workflow Tree**<br><br>2. Click **Create a new Validator**. A new validator dialog will appear.<br><br>3. Type the text or regular expression to search for in the response. If it's a regular expression then set the **Is text a regular expression?** to **Yes**.<br><br>4. Change the **Fail If** setting if you want an error to occur if the text is found. The value **Found** is not available for global validators. The default is **Not Found.**<br><br>5. The **Scope** setting allows to change the validator from local to global.<br><br>6. Change the **Action if Failed** setting to **Abort Iteration** to stop the current iteration, mark it as failed, and start a new one in the event the validation failed.<br><br>7. Click **Add Validator** when finished. | |

Alternatively you can create a validator for a recorded session by opening its session inspector and clicking **Create Validator** (a) on the toolbar.

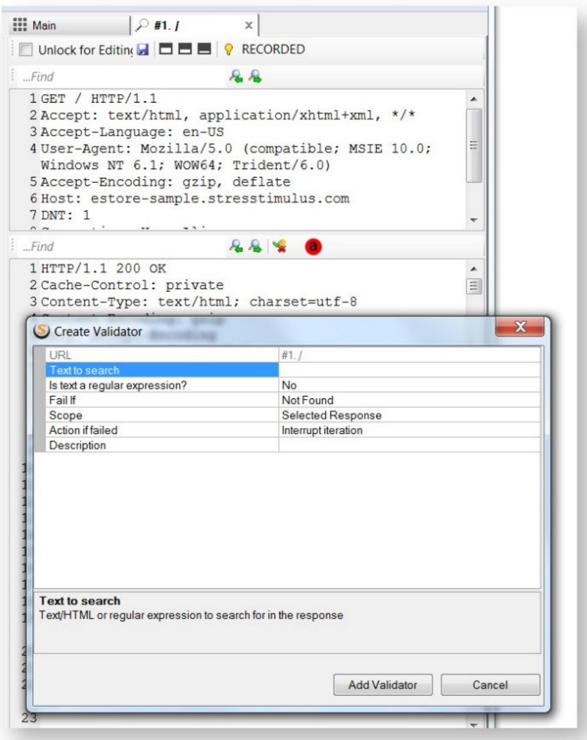
Another convenient way of creating validators is when comparing two sessions. In the Compare Sessions Inspector, select the **Response** tab and on the toolbar of the replayed window on the right, click **Create Validator** (b).



```
HTTP/1.1 302 Moved Temporarily
Server: Apache-Coyote/1.1
Expires: 0
Cache-Control: no-cache, no-store,
must-revalidate, max-age=0
```

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie:
JSESSIONID=ECEF74C85D8D528F94D2629EEF98F2EA;
Path=/idp; Secure
```

**Note:** You can highlight text in the replayed response before clicking the Create Validator button and this text will automatically populate the Text to search field in the Create Validator dialog.



```
GET / HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Accept-Language: en-US
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0;
Windows NT 6.1; WOW64; Trident/6.0)
Accept-Encoding: gzip, deflate
Host: estore-sample.stresstimulus.com
DNT: 1
```

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
```

| | |
|---|---|
| URL | #1. / |
| Text to search | |
| Is text a regular expression? | No |
| Fail If | Not Found |
| Scope | Selected Response |
| Action if failed | Interrupt iteration |
| Description | |

**Text to search**
Text/HTML or regular expression to search for in the response

### 4.6.4.1 Validators and Failed Iterations, Pages and Transactions

#### Iteration, Page and Transaction Counters

StresStimulus keeps track of all started, (successfully) completed and failed iterations, pages and transactions. These three counters are monitored and displayed in the real-time dashboard during the test run and are reported in the test results.

After the test is complete, the following equation should be true for all iterations, pages and transactions:

```
started = completed + failed
```

During the test run, this equation has the following form:

```
started = completed + failed + pending
```

### How to Validate an Iteration

The iteration fails only when it contains a validator which fails and has the property **Action if Failed** set to **Stop Iteration.**

If the iteration during execution encountered an HTTP error in one or several responses, this will not cause the iteration to fail and it will continue its execution.

If you want a particular HTTP error to cause an iteration to fail, you need to define a validator detecting this error, for example by checking the response status-code.

If you want all HTTP error to cause any iteration to fail, you can define a validator with **Scope** property sets to **Global. A single** global validator is capable to fail any iteration.

If an iteration is marked as failed, it will be counted toward started and failed iterations. It will not be counted toward passed or incomplete iterations. Performance metrics, such as average response time for such iteration is excluded from the test metrics.

### How to Validate a Page and Transaction

All Pages and Transactions that fully complete before a validator failed, are considered as started and completed.

A Page or Transaction that contains the failed request with a validator is considered as started and failed.

All Pages and Transactions that did not start as a result of the failed validator which stopped the iteration are **not** considered as started.

If a page or transaction is marked as failed, its performance metrics, such as average response time, is excluded from the test metrics.

## 4.6.5 If...Then

Ent SP

An If...Then condition will execute its child objects if a given condition is true (or false). The condition will compare an extractor to a string to determine if the loop should run.

To create an If...Then condition, follow the following steps:

1. Go to the **Build Test Case** node.

2. Select the starting object

3. Right mouse click to open the Context Menu and click **Create If...Then**.

4. A new object dialog will appear.

5. Select the last object in the condition.

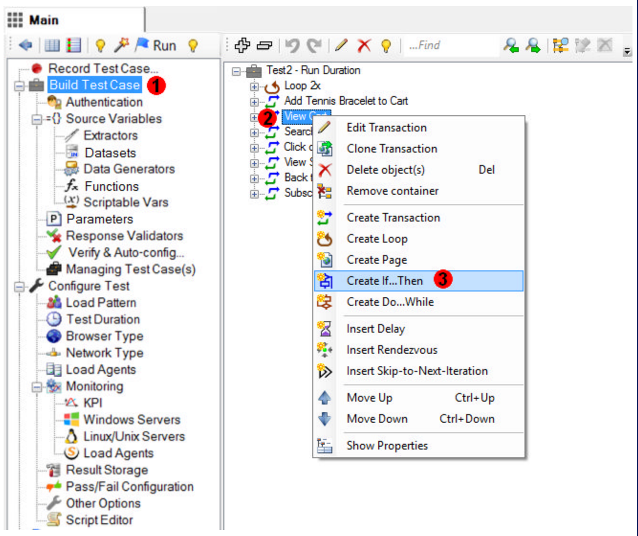
6. Click **Create If...Then**.

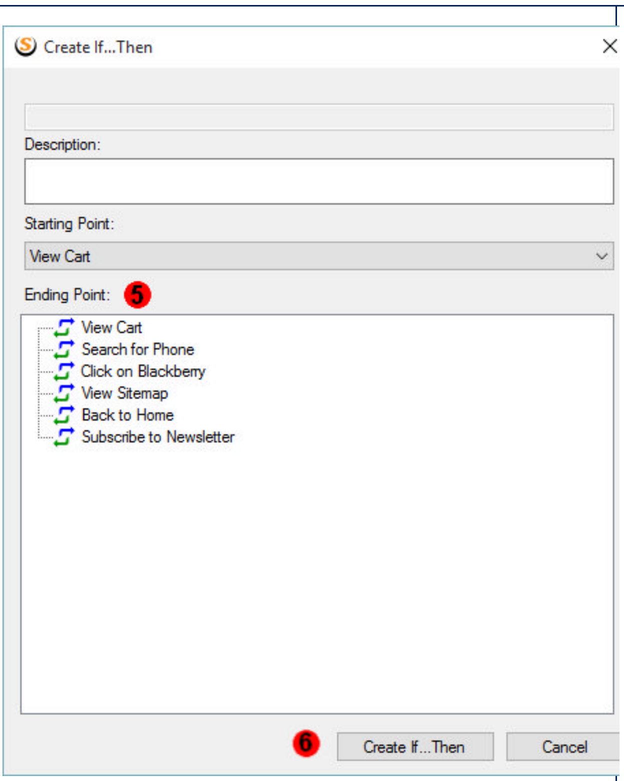
7. In the property **Extractor to match,** select an Extractor from the drop-down to compare with the string.

8. In the property **Text to match** enter a string to compare with the Extractor.

9. In the property **Run the child objects if Match?** select **Yes** to run the children when the Extractor matches the String, and skip the children when the Extractor does not match the String. Select **No** to skip the children when the Extractor matches the String, and run the children when the Extractor does not match the String.

After that, before executing the child objects, the extractor's value will be compared to a string to determine if the they should run.





Create If...Then

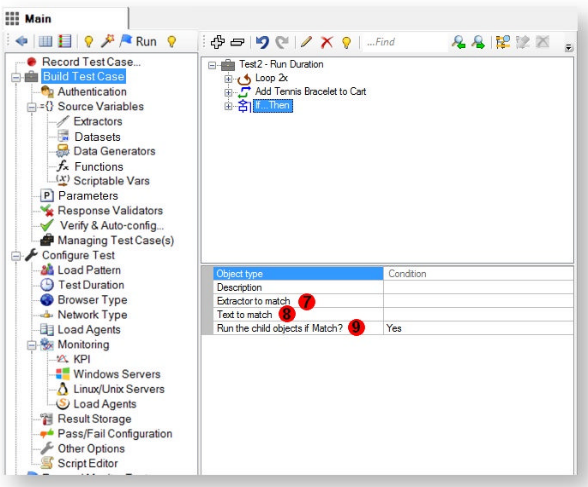
Description:

Starting Point: View Cart

Ending Point: 5

- View Cart
- Search for Phone
- Click on Blackberry
- View Sitemap
- Back to Home
- Subscribe to Newsletter

6 Create If...Then | Cancel



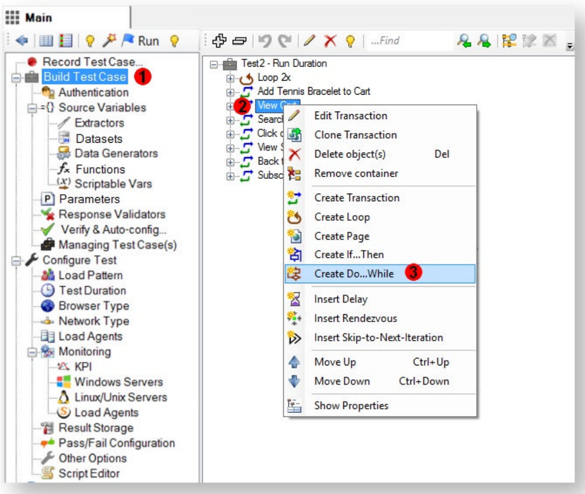
| | |
|---|---|
| Object type | Condition |
| Description | |
| Extractor to match 7 | |
| Text to match 8 | |
| Run the child objects if Match? 9 | Yes |

## 4.6.6 Do...While

Ent SP

Sometimes it is not possible to determine in advance how many times a loop should be repeated. Here is an example:

A user queries a Check Status page multiple times until the server completes a slow asynchronous transaction and redirects to a Result page. When recording such test case, it is unknown how many times during the load test each virtual user should query the Check Status page. To configure such scenario, it is necessary to use a **Do...While** (Conditional Loop)**.** Conditional loops have a condition which is checked at the end of the loop to determine if the loop should continue or exit. You can configure a loop to exit when the condition returns true or false.



Two types of conditions are supported.

•

- A **Text Based condition** depends on finding a specified text or regular expressions in a response.
- An **Extractor Based condition** depends on matching an extractor value with a specified text.

To make the provided example work, a **Text Based** exit condition should be defined. This condition has to verify if the response contains "Location: http://www.website.com/result.aspx". In this case, exit the loop.

To create a conditional loop, follow these steps:

1. Go to the Build Test Case node.

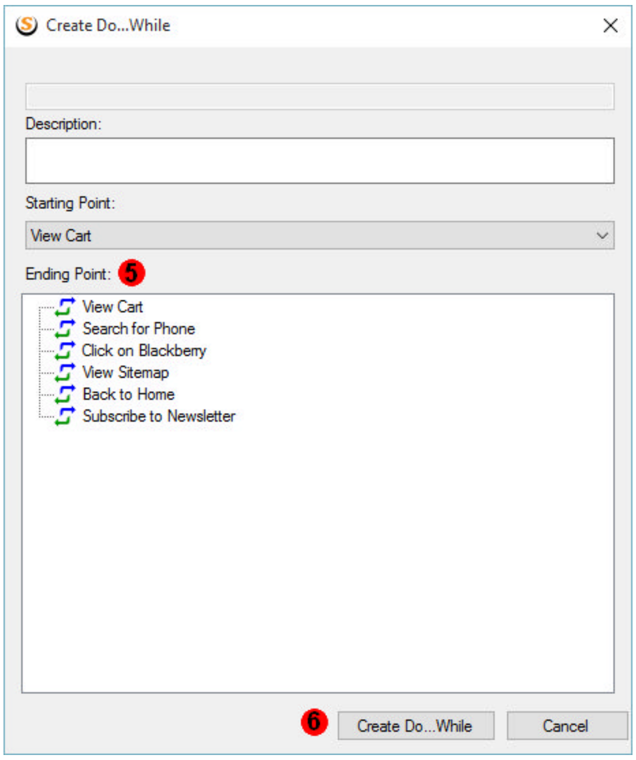
2. Select the starting object.

3. Right mouse click to open the context menu and Click Create **Do...While**.

4. A new object dialog will appear.

5. Select the last object in the loop

6. Click Create **Do...While**.



Create Do...While

Description:

Starting Point: View Cart

Ending Point: 5

- View Cart
- Search for Phone
- Click on Blackberry
- View Sitemap
- Back to Home
- Subscribe to Newsletter

7. Set the **Number of Repeats (Max)** property to the maximum allowed number of loop iterations to avoid endless loops.

8. Optionally specify in the **Delay before next Loop**.

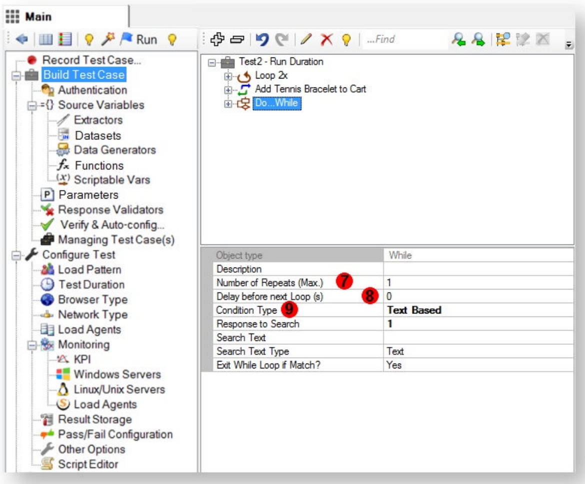
9. Select condition type as **Text Based or Extractor Based**

10. The Text Based condition is based on finding a specified text or regular expressions in an HTTP response . For this condition, type specify the following:

a. From the drop-down select **Response to Search**, where the specified text or regular expressions will be searched.

b. In the **Search Text** property, specify a character string that will be searched in the HTTP response.

c. In the **Search Text Type** property select **Text** if the search string is a text, or select **Regular Expression** if the search string is regular expressions.



| Object type | While |
|---|---|
| Description | |
| Number of Repeats (Max.) | 1 |
| Delay before next Loop (s) | 0 |
| Condition Type | **Text Based** |
| Response to Search | **1** |
| Search Text | |
| Search Text Type | Text |
| Exit While Loop if Match? | Yes |

d. In the **Exit While Loop if Match?** property select **Yes** to repeat the loop, when the search text is not found, and exit the loop, when the search text is found. Or select **No** to repeat the loop, when the search text is found and exit the loop when the search text is not found.

11.The Extractor Based condition is based on evaluating an extractor. For this condition type specify the following:

a. From the drop-down, select the **Extractor Name** that will be used

b. In the **Text to compare** property, specify a text to compare with the Extractor

c. In the **Exit While Loop if Match ?** property, select **Yes** to repeat the loop,when the extractor value is not found, and exit the loop when the extractor value is found. Or select **No** to repeat the loop when the extractor value is found, and exit the loop when the extractor value is not found.

## 4.6.7 Delay

Ent SP

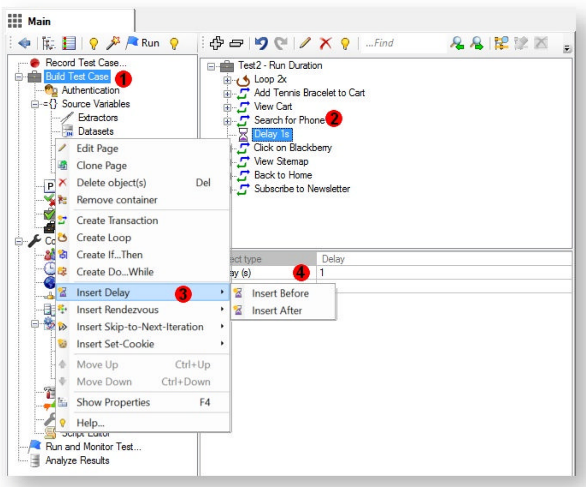
The delay object adds a pause in iteration execution.

1. Go to the **Build Test Case** node.

2. Select the object where you want to insert the delay.

3. Right click and select **Insert Delay** then select **Insert Before** or **Insert After**.

4. In the property grid set the **Delay (s)** property to the number of seconds to delay.



### 4.6.8 Rendezvous Points

Ent SP

Rendezvous are points in a test case that are used to synchronize Virtual Users to carry out tasks at the same moment. This is used to order to create a simultaneous load on the server at specific points in the test case.

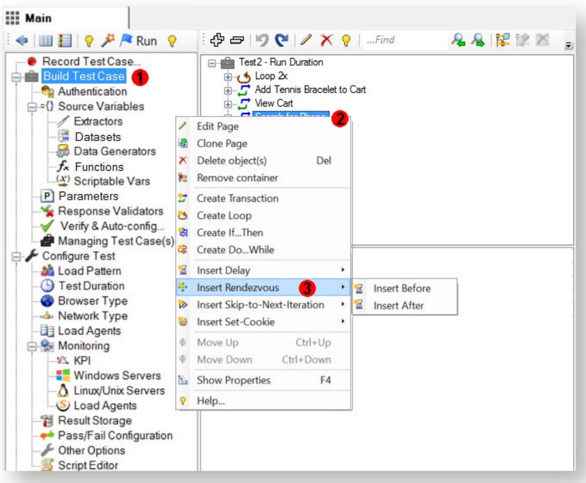
For example, imagine a test case that logs in, performs a critical action and logs out. It is required to test the critical action under a 100 VU load. However, r unning a steady load with 100 VUs will not guarantee 100 VU load on the critical action because each VU must first login. Since it is unpredictable how long the login will take, it can't be guaranteed that all 100VUs will execute the critical action at the same time. In these situations you can add a rendezvous before the critical action to have each VU wait after its login for all other VUs to login.

To add a rendezvous, do the following:

1. Go to the **Build Test Case** node.

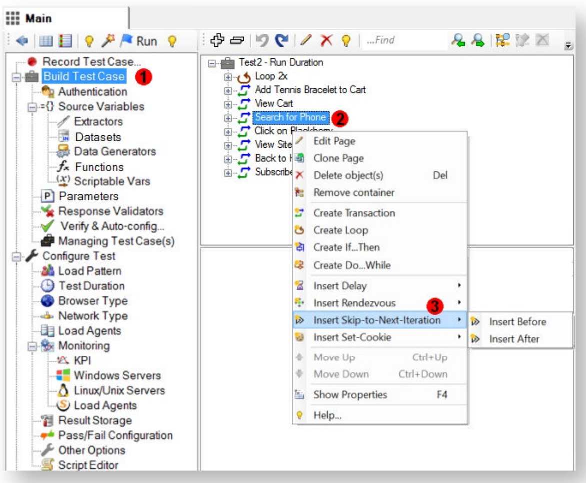
2. Select the object where you want to insert the rendezvous.

3. Right click and select **Insert Rendezvous** then select **Insert Before** or **Insert After**.



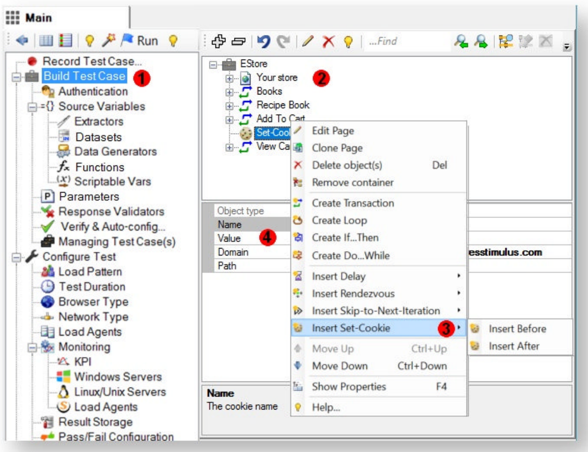
### 4.6.9 Skip-to-Next-Iteration

Ent SP

| | |
|---|---|
| The skip to next iteration object stops the current iteration execution. This might be useful if the test case has a fail condition and should retry by starting the test case iteration over again. Unlike a failed iteration, an iteration that was skipped will count toward the total test case iterations.<br>1. Go to the **Build Test Case** node.<br>2. Select the object after which you want to stop the current iteration execution.<br>3. Right click and select **Insert Skip-to-Next-Iteration** then select **Insert Before** or **Insert After**. |  |

## 4.6.10 Set-Cookie

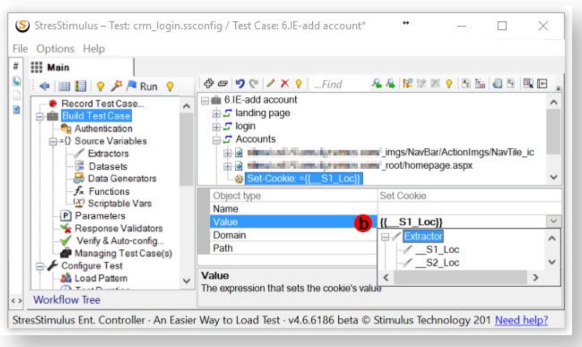
StresStimulus handles cookie correlation automatically. Every time a server response contains a Set-Cookie header, StresStimulus acts accordingly and sets the request cookie value in all subsequent requests. However, in some rare instances the request cookie value is set by some client side script and can't be automatically correlated. For example, an application can have a client script that after every click, sets a timeout cookie to a few minutes into the future to monitor user inactivity. While load testing such test case, the recorded value of this cookie can't be used because the server will think the VU is inactive. In these cases, you can use the Set-Cookie object to manually parameterize the cookie value.

| | |
|---|---|
| To use the set-cookie object:<br>1. Go to the **BuildTest Case** node.<br>2. Select the object where you want to insert the set-cookie after. The value of the cookie will be set at that time during the test case execution.<br>3. Right click and select **Insert Set-Cookie** then select **Insert Before** or **Insert After**.<br>4. In the property grid set the following properties:<br>a. Set the Name of the cookie.<br>b. Set the expression that will be evaluated to determine the value of the cookie. The expressions should be be selected from the variable |  |

picker in the property grid.

c. Set the Domain (host) that the cookie belongs to.

d. Set the Path that the cookie belongs to. The default is "/" which means all pages in the domain.



| | |
|---|---|
| Object type | Set Cookie |
| Name | |
| Value | {{__S1_Loc}} |
| Domain | |
| Path | |

**Value**
The expression that sets the cookie's value

## 4.7 Managing Test Case(s)

Ent SP

Sometimes it is necessary to have VUs emulate different navigating scenarios during a test in order to more accurately mimic real life situations. For example, in an e-commerce website, some VUs are browsing the catalog while other VUs are placing orders. To accommodate such testing requirements, StresStimulus should execute several test cases concurrently during the test. Multiple test cases are used mainly to emulate different categories of users, concurrently accessing a website.

### 4.7.1 Creating Multiple Test Cases

The following methods are used to add more test cases to an existing test.

- Recording a test case
- Setting a test case from existing sessions
- Cloning a test case
- Importing a test case from another test

- Setting a Test Case from a .saz or .har session file or JMeter result file.

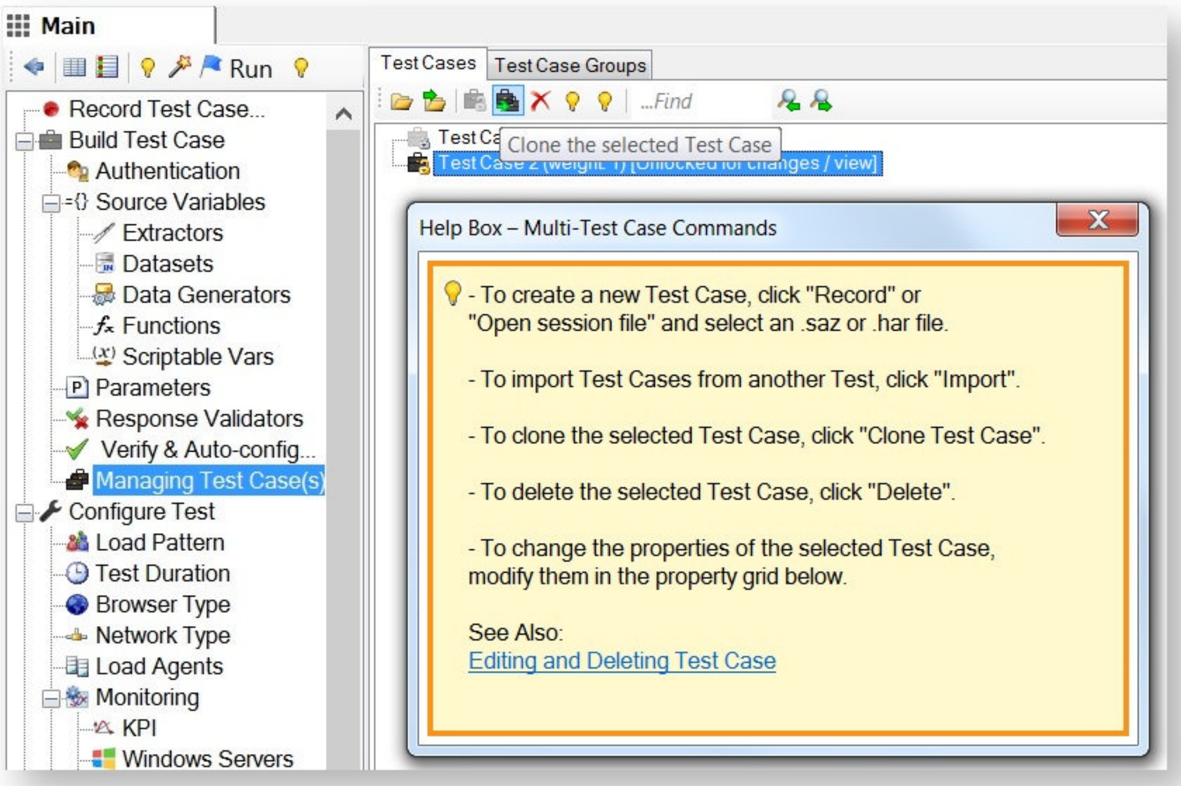
### 4.7.1.1 Recording a Test Case

Adding additional test cases to a test is done the same way as creating the first one. First you can record a test case or set existing sessions as a test case as described in Recording Test Case.

Select the **Add Test Case** option to create another test case.

### 4.7.1.2 Cloning a Test Case

To clone a selected test case, in the **Managing Test Case(s)** section, click **Clone Test Case** on the toolbar. All sessions and test case objects will be duplicated.



Help Box – Multi-Test Case Commands

- To create a new Test Case, click "Record" or "Open session file" and select an .saz or .har file.

- To import Test Cases from another Test, click "Import".

- To clone the selected Test Case, click "Clone Test Case".

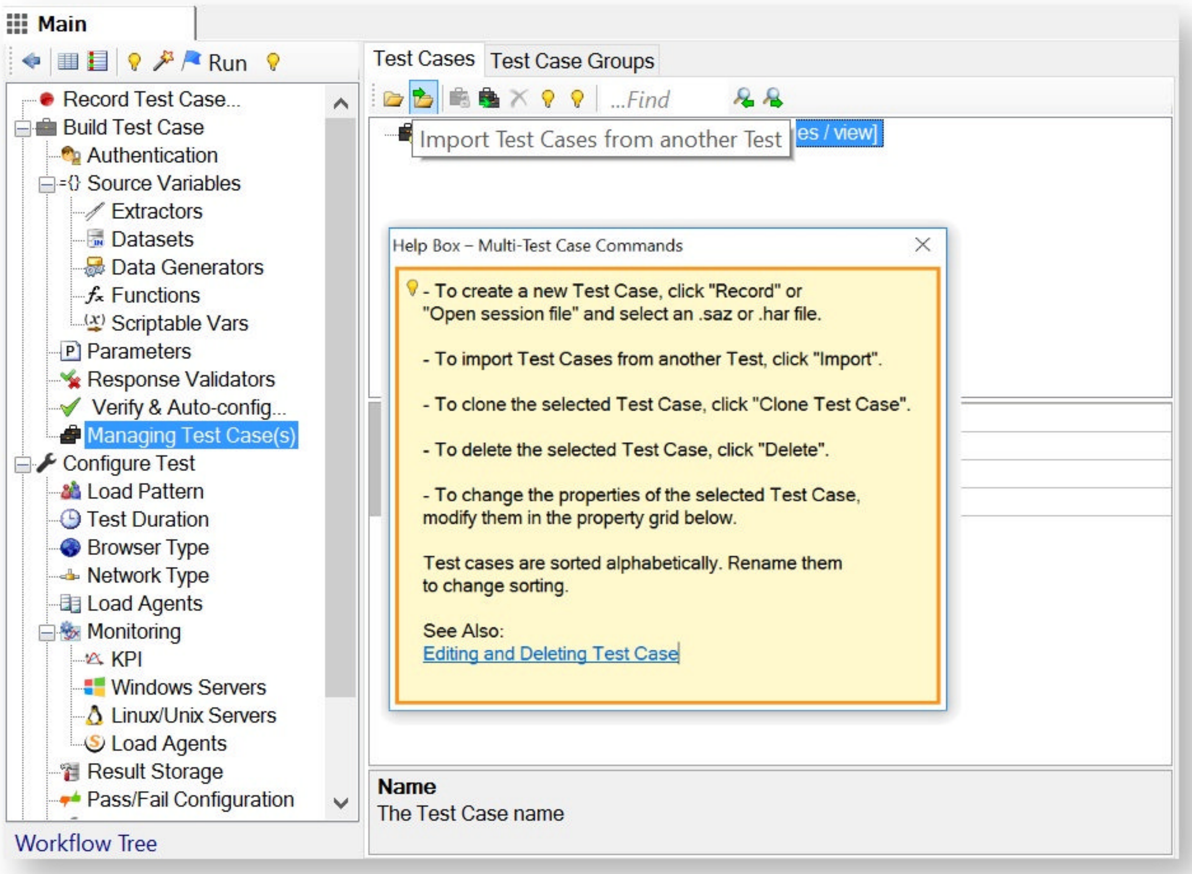
- To delete the selected Test Case, click "Delete".

- To change the properties of the selected Test Case, modify them in the property grid below.

See Also:
Editing and Deleting Test Case

### 4.7.1.3 Importing a Test Case from another test

Test cases can be imported from one test to another. In the **Managing Test Case(s)** section, click **Import Test Cases from another Test** on the toolbar and then in the Windows Explorer pop-up window, select an .ssconfig file and click **Open**. All test cases from the selected test will be imported into the current test. All test objects will be copied into the .ssconfig file of the current test, and all .saz files with sessions will be copied into the current folder.



Help Box – Multi-Test Case Commands

- To create a new Test Case, click "Record" or "Open session file" and select an .saz or .har file.

- To import Test Cases from another Test, click "Import".

- To clone the selected Test Case, click "Clone Test Case".

- To delete the selected Test Case, click "Delete".

- To change the properties of the selected Test Case, modify them in the property grid below.

Test cases are sorted alphabetically. Rename them to change sorting.

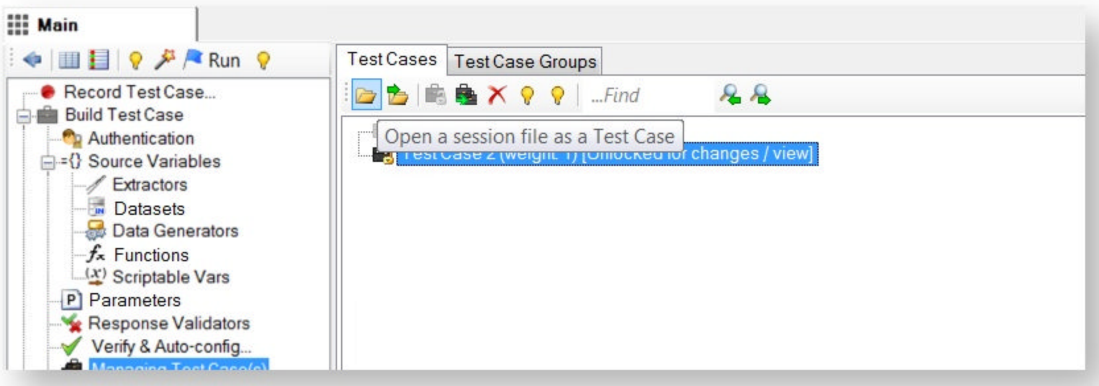
See Also:
Editing and Deleting Test Case

**Name**
The Test Case name

**Tip:** If you need to import some but not all test cases, then after the import, delete the test cases you do not need.

### 4.7.1.4 Setting a Test Case from a Session File

Click **Open a session file as a Test Case** and select a Fiddler file (.saz, HTTP archive file (.har) or JMeter result file. The selected file will be imported as a new Test Case.



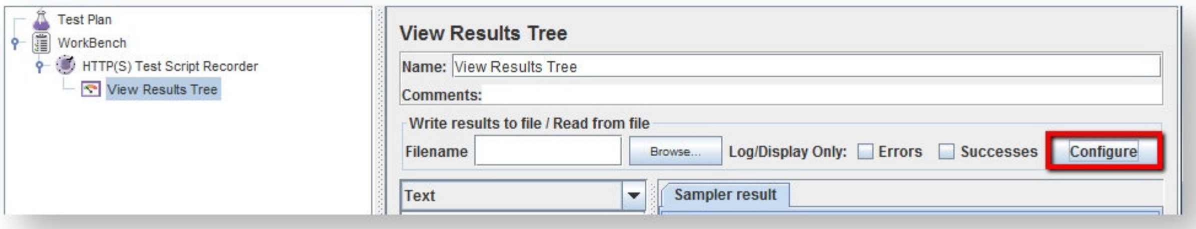
**Tip**: Test cases on the list are sorted alphabetically. Rename the test cases to displays them in the order you need.

#### 4.7.1.5 Import JMeter result file

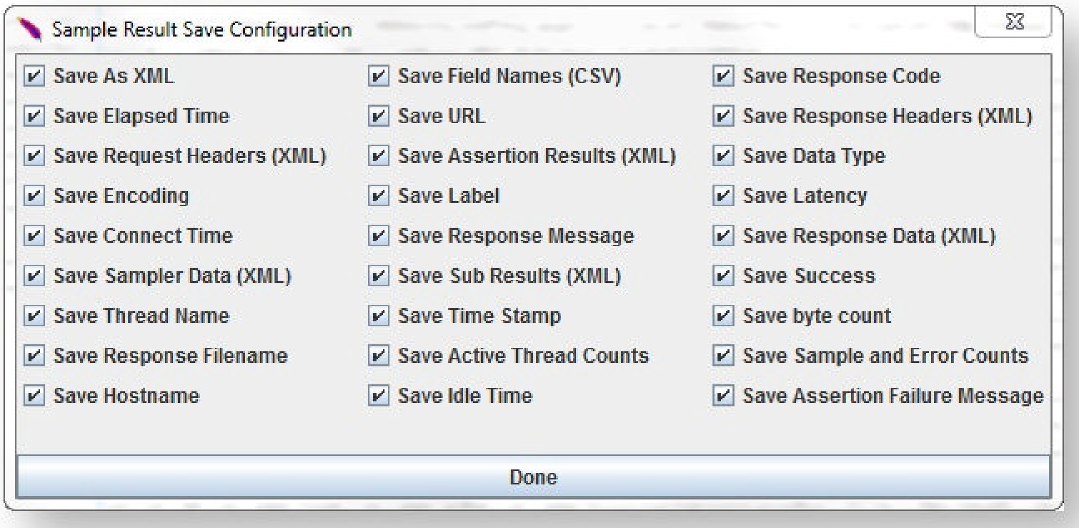
StresStimulus can import JMeter test cases saved as a sample results file. The results file must be in XML format and must include all requests and responses.

To create this file, in JMeter add a Table or Tree Listener to a Test Script Recorder. It is highly recommended to use sample results from the recording to make sure all the request and response data is accurate. The results listener should now be attached to the recorder.

Click the configure button to bring up the sample save results configuration dialog.



In the appeared dialog, check all the boxes to make sure the results file will have all the necessary data .

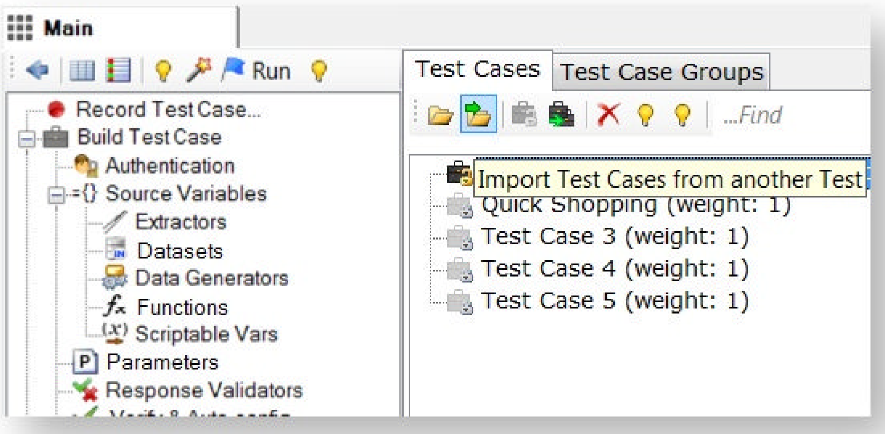


Sample Result Save Configuration

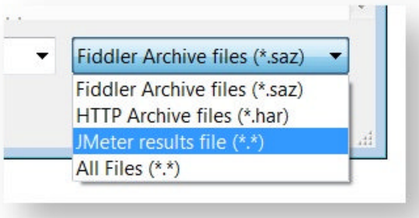
|  |  |  |
|---|---|---|
| ✓ Save As XML | ✓ Save Field Names (CSV) | ✓ Save Response Code |
| ✓ Save Elapsed Time | ✓ Save URL | ✓ Save Response Headers (XML) |
| ✓ Save Request Headers (XML) | ✓ Save Assertion Results (XML) | ✓ Save Data Type |
| ✓ Save Encoding | ✓ Save Label | ✓ Save Latency |
| ✓ Save Connect Time | ✓ Save Response Message | ✓ Save Response Data (XML) |
| ✓ Save Sampler Data (XML) | ✓ Save Sub Results (XML) | ✓ Save Success |
| ✓ Save Thread Name | ✓ Save Time Stamp | ✓ Save byte count |
| ✓ Save Response Filename | ✓ Save Active Thread Counts | ✓ Save Sample and Error Counts |
| ✓ Save Hostname | ✓ Save Idle Time | ✓ Save Assertion Failure Message |

Done

To import the results file, in StresStimulus go to the **Managing Test Case(s)** section node on the Workflow tree and click Open a session file as a Test Case.



In the appeared open file dialog, select the JMeter results file in the filter and select the path of the saved Jmeter file.


Fiddler Archive files (*.saz)
Fiddler Archive files (*.saz)
HTTP Archive files (*.har)
JMeter results file (*.*)
All Files (*.*)

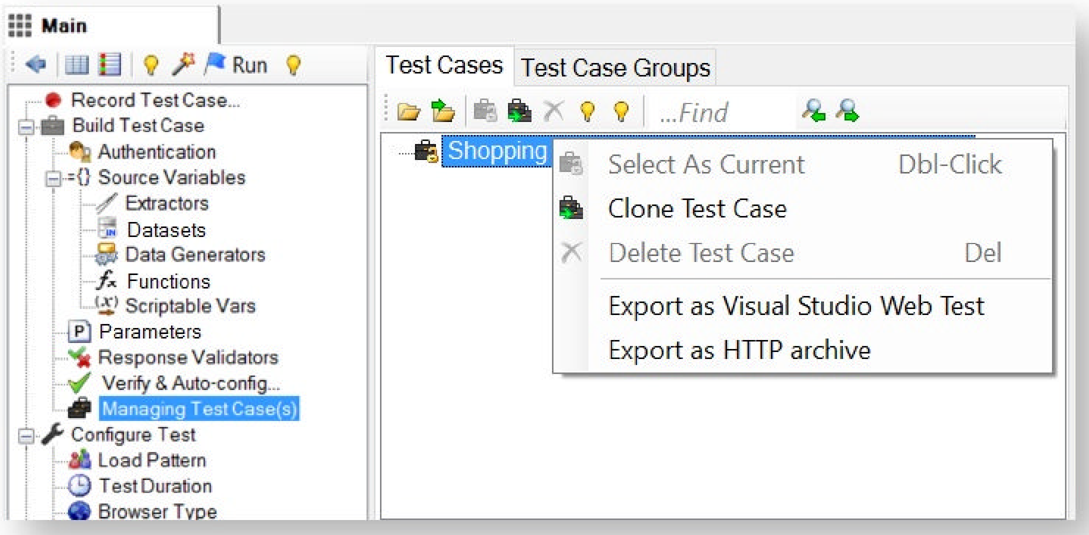
### 4.7.2 Exporting a Test Case

Test case sessions are stored internally in Fiddler .saz format. A session .saz file is a part of the test. It can be used with comparable programs.

Additionally, test case sessions can be exported as an HTTP archive (.har).

**Info:** HAR format is owned by **W3C** (see **HTTP Archive (HAR) format**). It is adopted by all major web browsers and many web metering and performance tools , listed here.

To save test case sessions as an HTTP archive, in **Managing Test Case(s)** section, right-click a test case and select **Export Test Case.**



### 4.7.2.1 Web Test Script Generator for Visual Studio

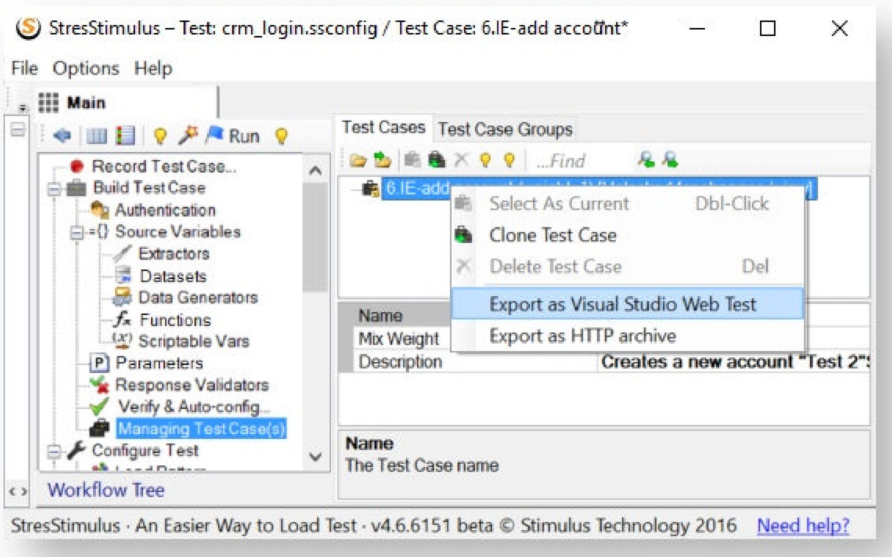
A test case can be exported to Visual Studio ".webtest" format. This can be useful when test cases recorded using Visual Studio recorder do not work because of missing correlation parameters or a portion of the traffic (i.e. requests issued by Active-X controls) . In this case, the test case can be recorded in StresStimulus and then exported to .webtest format, so that it can be opened and properly executed in Visual Studio.

To export a test case, in the Test Case Tree, navigate to **Managing Test Case(s)**, select a test case, right-click and select **Export as a Visual Studio Web Test.**

When exporting, StresStimulus will export only the following test case elements:

- All Requests, including headers and body.
- Extractors
- Parameters
- Pages and Transactions

All other test case objects must be recreated manually when the .webtest is imported into Visual Studio.



## 4.7.3 Editing and Deleting a Test Case

**1. Editing a Test Case**

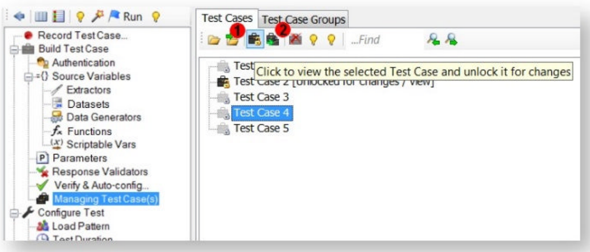
Only one test case can be edited at a time. In order to unlock a different test case, which is currently locked for editing, select it and click **Click to view the selected Test Case and unlock it for changes**. After that, the test case name is displayed in the StresStimulus application title bar or in the Fiddler title bar when the StresStimulus tab is selected.

Unlocking a Test Case for changes does not impact its or any other test case's execution. This options impacts only which test case is available for editing.

**2. Deleting a Test Case**

Test cases can be deleted by selecting them and clicking **Delete**.

A Test should have at least one Test Case. The last Test Case cannot be deleted. You can replace it either by re-recording or setting a new Test Case.



## 4.7.4 Running Multiple Test Cases

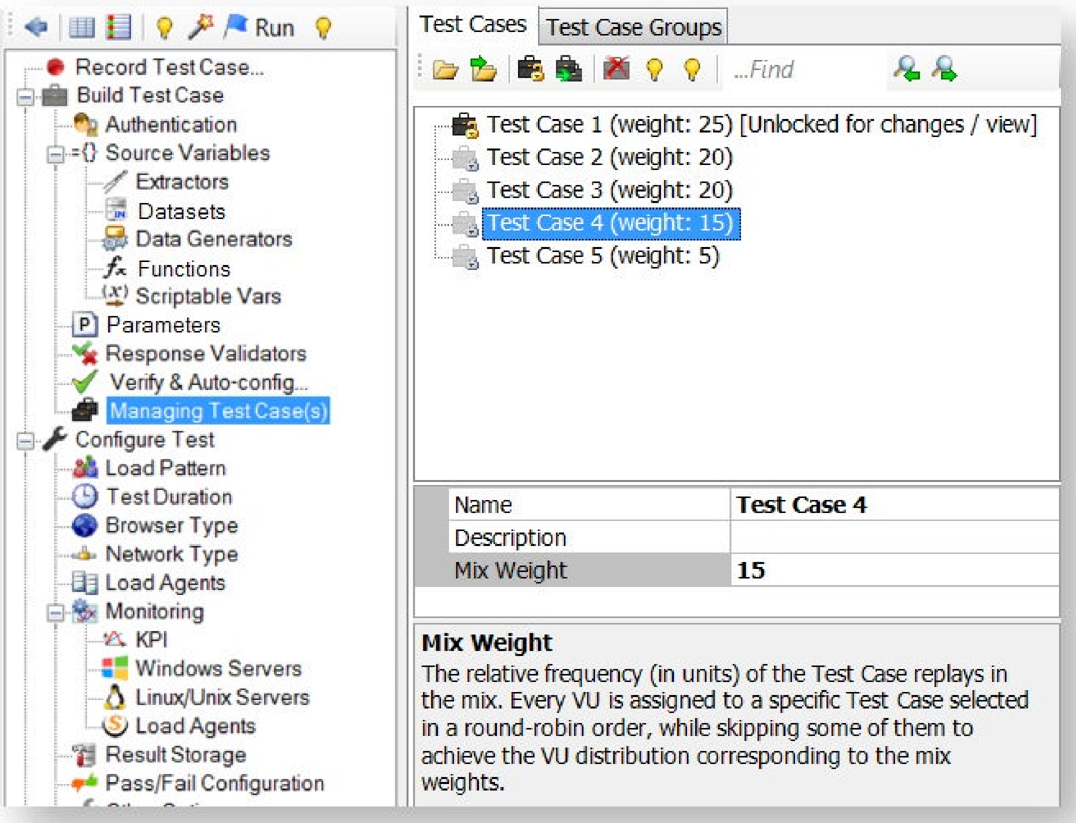
There are two test case mixing models: concurrent and sequential-concurrent.

### 4.7.4.1 Concurrent Test Case

By default multiple test cases are executed concurrently (in-parallel). Every test case has a **Mix Weight** configuration property. A mix weight is a relative frequency (in units) of the Test Case replay in the mix. It is also a relative probability that a virtual user will be assigned to this test case. VUs are distributed between the Test Cases proportionately to their Mix Weights.

**Note:** Every VU, after its instantiation, is assigned to a specific Test Case for the entire duration of the test. For every subsequent VU, test cases are selected in round-robin order, while skipping some of them to achieve the VU distribution corresponding to the mix weights.

To set a test case mix weight, change the Mix Weight in the property grid.



Test Case 1 (weight: 25) [Unlocked for changes / view]
Test Case 2 (weight: 20)
Test Case 3 (weight: 20)
Test Case 4 (weight: 15)
Test Case 5 (weight: 5)

| | |
|---|---|
| Name | Test Case 4 |
| Description | |
| Mix Weight | 15 |

**Mix Weight**
The relative frequency (in units) of the Test Case replays in the mix. Every VU is assigned to a specific Test Case selected in a round-robin order, while skipping some of them to achieve the VU distribution corresponding to the mix weights.

### 4.7.4.2 Sequential Test Case Groups

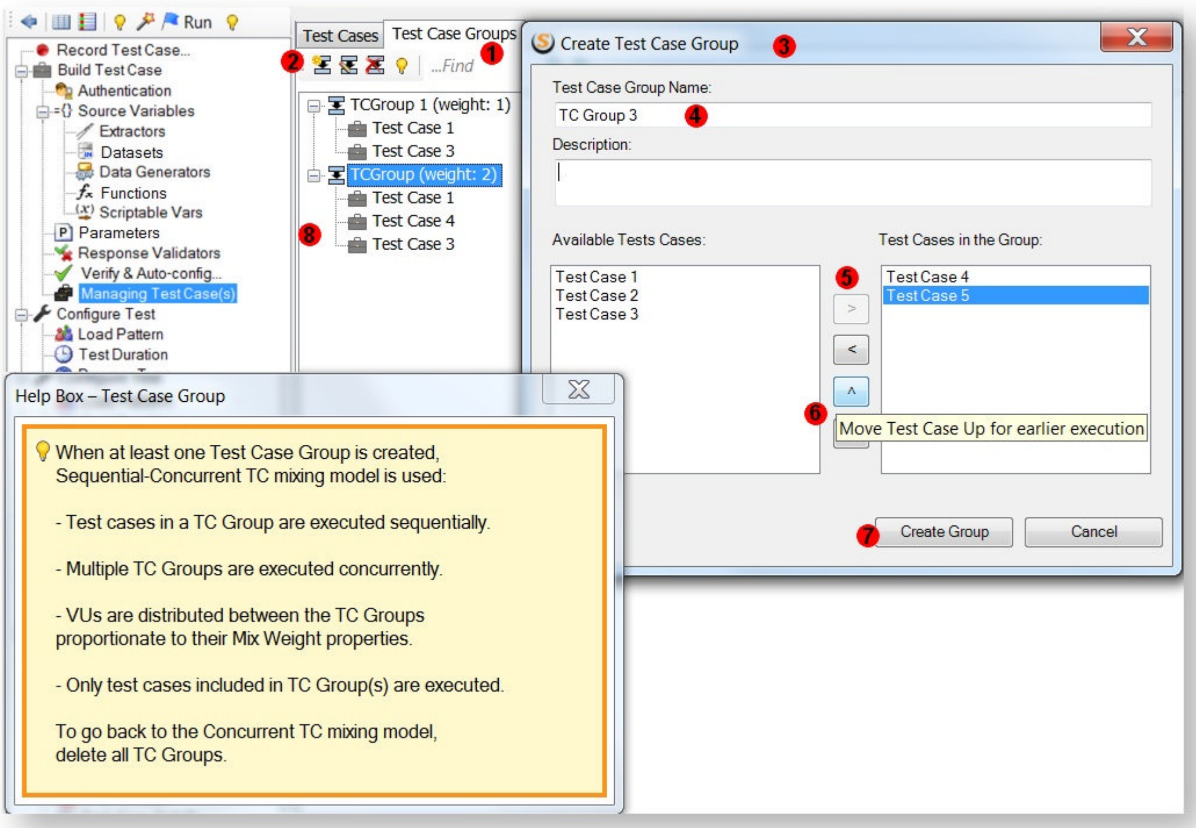
Test cases can be combined into groups, called Test Case (TC) Groups. Test cases in a TC Group are executed sequentially in a predetermined order by a group of VUs. Using TC Groups allows recording smaller and less complex test cases, and then combining them into needed sequences. Several TC Groups are executed in parallel by different groups of VUs. A test case can be added to several different TC Groups.

#### Creating TC Groups

To create a test case group,

1. Select the **Test Case Group** tab,
2. Click **Create** on the toolbar
3. **Create Test Case Group** dialog will appear
4. Enter TC Group name
5. Add the selected test cases to the group.
6. Arrange the order of test case execution by moving them up or down on the list.

7. Click **Create Group**
8. Test cases will appear on the tree in the order of their execution.



When at least one Test Case Group is created, Sequential-Concurrent TC mixing model is used:

- Test cases in a TC Group are executed sequentially.

- Multiple TC Groups are executed concurrently.

- VUs are distributed between the TC Groups proportionate to their Mix Weight properties.

- Only test cases included in TC Group(s) are executed.

To go back to the Concurrent TC mixing model, delete all TC Groups.

**Note:** Once at least one TC Group is created, then each VU can only execute a TC Group, not a test case outside a TC Group. If you want to execute a TC not included in any TC Group, then create a TC Group with a single test case. Before creating the first TC Group the following warning will appear.

**Note:**

- Every TC should be configured independently either before or after adding it to the TC Group.
- When at least one TC Group is created, the Mix Weight and cache control-related properties

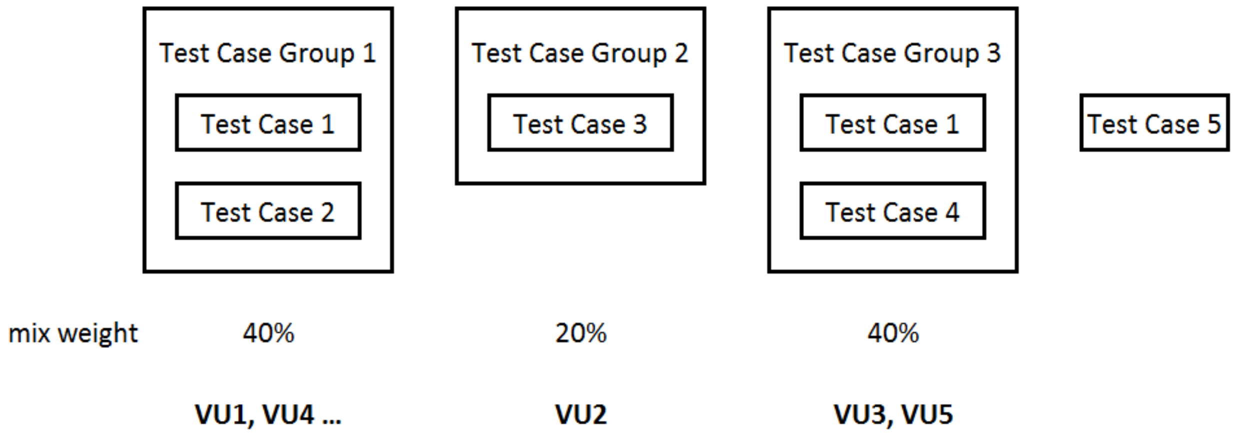
are moved from Test Cases to TC Groups because all test cases in a TC Group are executed with the same frequency and must have the same cache control settings.

TC Groups execution is similar to that of test cases. If a Test has more than one TC Group, then they are executed concurrently. The Mix Weight property of each TC Group determines the relative frequency (in units) of its replay in the mix.

Every VU is assigned to a specific TC Group for the entire duration of the test. A VU executes all test cases in a TC Group before starting the next iteration of the TC Group.

In the example shown below, a test has 5 test cases. 4 out of them are part of 3 TC Groups with mix weights 40%, 20% and 40% respectively. Test emulated 5 VUs.

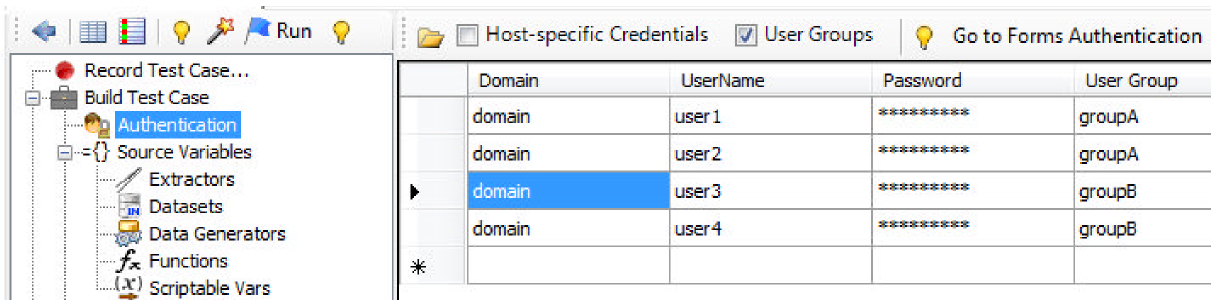
VU1 and VU4 will execute TC1 and TC2 sequentially, VU2 will execute TC3, VU3 and VU5 will execute TC1 and TC4 sequentially, TC5 will not be executed as it is not part of ant TC Group.



### 4.7.4.3 User Groups

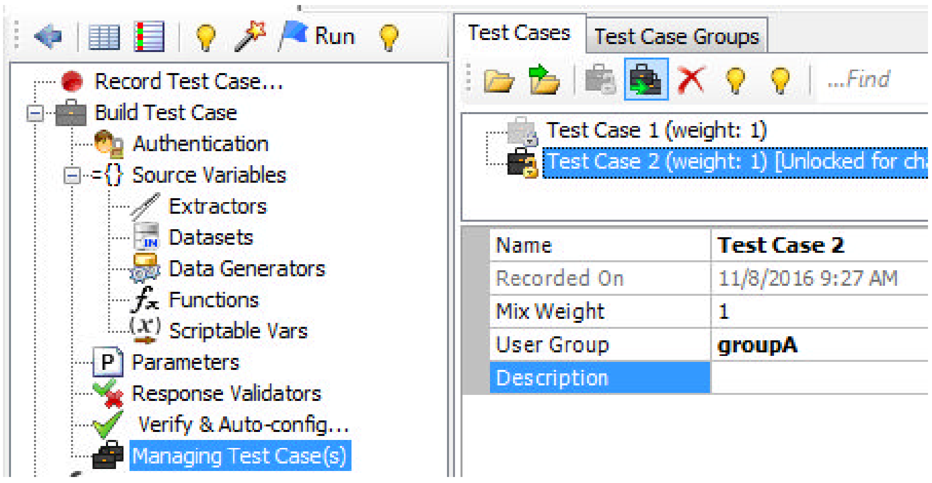
When server authentication is used, some test cases may require users that have certain permissions. Therefore there is a need to map certain VUs' credentials to certain test cases. In StresStimulus, this is implemented using **User Groups**. Every VU credential is assigned to a user group, and every test case runs with a selected user group. Here is how to use these user group:

1. In the Authentication section, input all the credentials.
2. Click the **User Groups** checkbox to show the User Group column.
3. In each credential record, write in the group name that the record will belong to.



| Domain | UserName | Password | User Group |
|---|---|---|---|
| domain | user1 | ********** | groupA |
| domain | user2 | ********** | groupA |
| domain | user3 | ********** | groupB |
| domain | user4 | ********** | groupB |

4. In the Managing Test Case section, select a test case (or test case group) and set the **User Group** property to the group name that this Test Case will use.



| | |
|---|---|
| Name | Test Case 2 |
| Recorded On | 11/8/2016 9:27 AM |
| Mix Weight | 1 |
| User Group | groupA |
| Description | |

# 5 CONFIGURING TEST

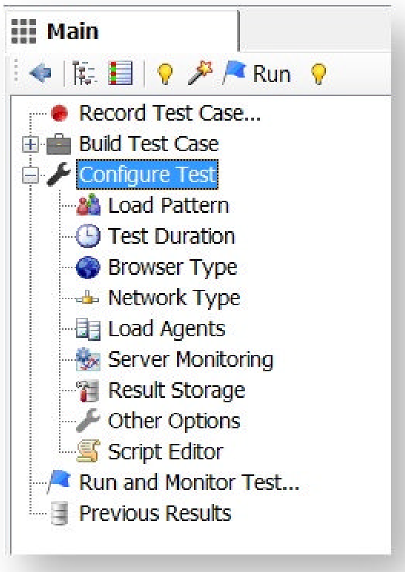
**Search the Configuring Test section**

StresStimulus generates test workload by executing several procedures:

- **Completing test iteration s.** StresStimulus replays a test case once per VU. A test iteration is a basic unit of load test.
- **Implementing a load pattern .** StresStimulus instantiates a certain number of VUs at certain moments. Two load patterns are supported: **Steady load** and **Step load** (see User Interface Reference -> Load Pattern).
- **Emulating a VU load**. After the VU is instantiated, test iteration completes multiple times until test ends.
- **Ending a test .** StresStimulus registers the moment a test completion condition occurs and finalizes the test according to selected test completion method (see User Interface Reference -> Test Duration).

These procedures are executed independently and concurrently. As a result, multiple VUs may be instantiated at different times, and each of them emulates the load by looping through their respective iterations independently until the test tends. If the test consists of several test cases, then different VUs will loop through different test cases.

All steps of configuring a test are performed on the **Main** tab on the third sections in the Workflow Tree, **Configure Test.**

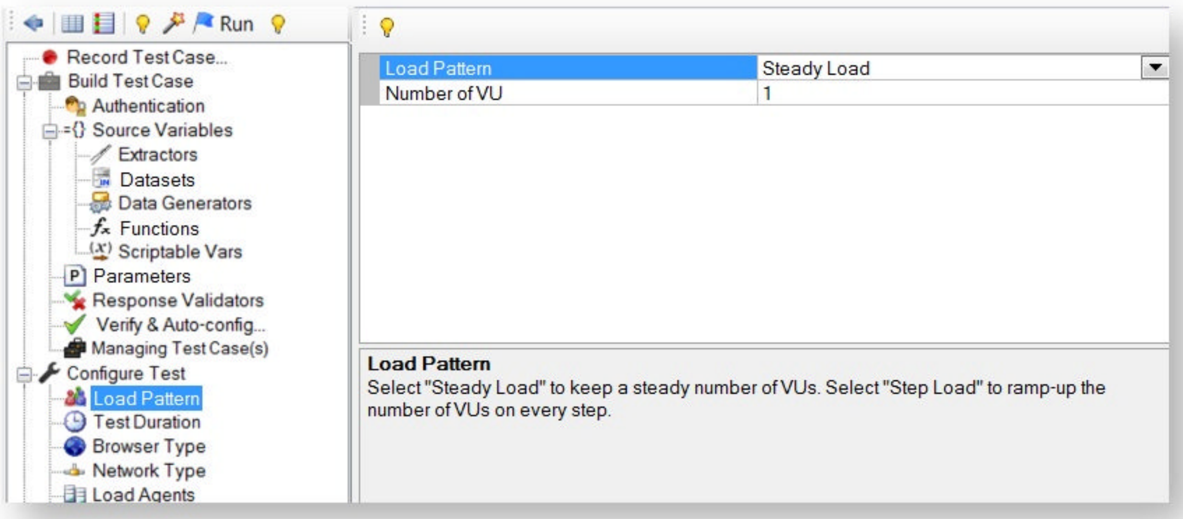


## 5.1 Load Pattern

Load Pattern defines how virtual users are instantiated. To configure VU load, select the **Load Patten** node in the **Workflow Tree**. There are two ways to create a load: Steady Load and Step Load.

### 5.1.1 Steady Load

A Steady load is a constant number of users during the entire test. When choosing Steady Load, set the Number of VU property to the number of users you want to create during the test.



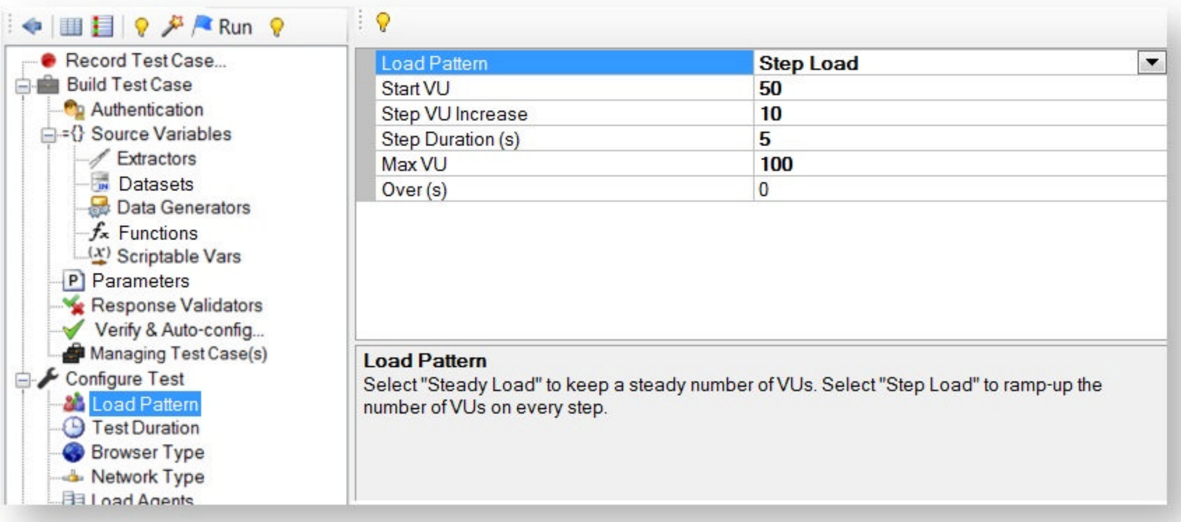
| | |
|---|---|
| Load Pattern | Steady Load |
| Number of VU | 1 |

**Load Pattern**
Select "Steady Load" to keep a steady number of VUs. Select "Step Load" to ramp-up the number of VUs on every step.

**Note**: The initial number of .NET threads allocated to the StresStimulus process before the test starts is equal to the number of VUs. During the test run, operating system can increase the number of threads if necessary because every VU can issue more than one concurrent request. You can increase the number of threats manually as described in the Load Generator Performance section.

### 5.1.2 Step Load

In a step load, the number of VUs increases up to a certain point. The test starts with a **Start VU** number. Then on every **Step Duration,** the number of VUs increases by **Step VU Increase**. Once the **Max VU** is reached, new VUs will no longer be instantiated.



| | |
|---|---|
| Load Pattern | Step Load |
| Start VU | 50 |
| Step VU Increase | 10 |
| Step Duration (s) | 5 |
| Max VU | 100 |
| Over (s) | 0 |

**Load Pattern**
Select "Steady Load" to keep a steady number of VUs. Select "Step Load" to ramp-up the number of VUs on every step.

You can optionally set the **Over(s)** property to gradually increase the step VUs over a period of time. If **Over** is zero, then the increase is instant.

**Note**: The initial number of .NET threads allocated to the StresStimulus process before the the test starts is equal to the maximum number of VUs.

## 5.2 Test Duration

StresStimulus determines when to complete a test based on the test configuration information. Ending a test is a two-step decision making process:

- Step 1: Wait until the Primary test completion conditions are met.
- Step 2: Finish the remaining part according to the selected Secondary test completion condition.

Primary test completion conditions:

- **Number of Iterations**: The test will complete after a certain number of iterations is performed by VUs.
- **Run Duration:** The test will complete after a certain amount of time elapsed.
- **Reaching Max Users**: The test will complete after a certain number of VUs are created (only used in step load).

Secondary test completion conditions:

- **Stop the Test**: End test immediately after the Primary conditions are met.is stored.
- **Wait for Responses:** Stop sending new requests and wait only for already issued requests to come back. This condition does not guarantee completion of any specific test case, transaction or even page.
- **Wait for Iterations to Complete**: Do not start any new iterations, but complete already started iteration and wait for responses in iteration. This secondary condition will allow to run test longer.

If the primary test completion condition is **Number of Iterations**, then the secondary condition is implicitly set to **Wait for Iteration to Complete**.

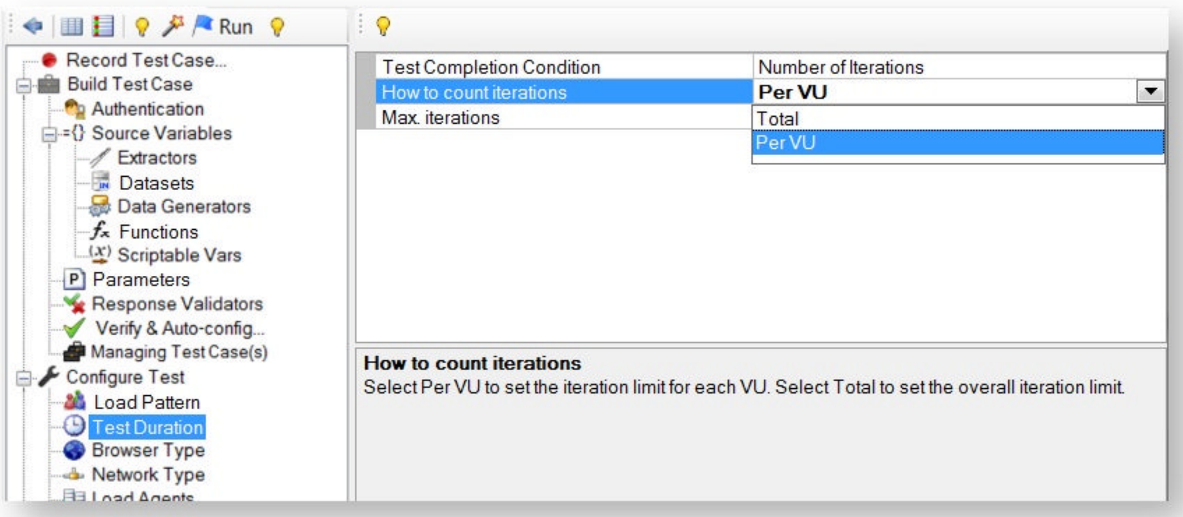
### 5.2.1 Number of Iterations

The test will complete after a certain number of iterations are performed by VUs.

1. Set the **Max iterations** property to specify the number of iterations to perform.
2. Set **How to count Iterations.**

- 
    - **Total -** The **Max** iterations will be shared between all VUs. This value must be at least equal to the **Number of VUs** in **Steady Load** or **Max VU** in **Step Load**. The new VU will start a new iteration only if there are non-started iterations remaining. As a consequence, not all VUs will complete the same number of iterations. Some VUs might not run because by the time they are instantiated, there are no iterations left to execute.
    - **Per VU -** Each VU will complete the same number of Max iterations.

**Note:** When the **Test Completion Condition** is set to **Number of Iterations**, then all iterations will be always complete.



| | |
|---|---|
| Test Completion Condition | Number of Iterations |
| How to count iterations | Per VU |
| Max. iterations | |

Total
Per VU

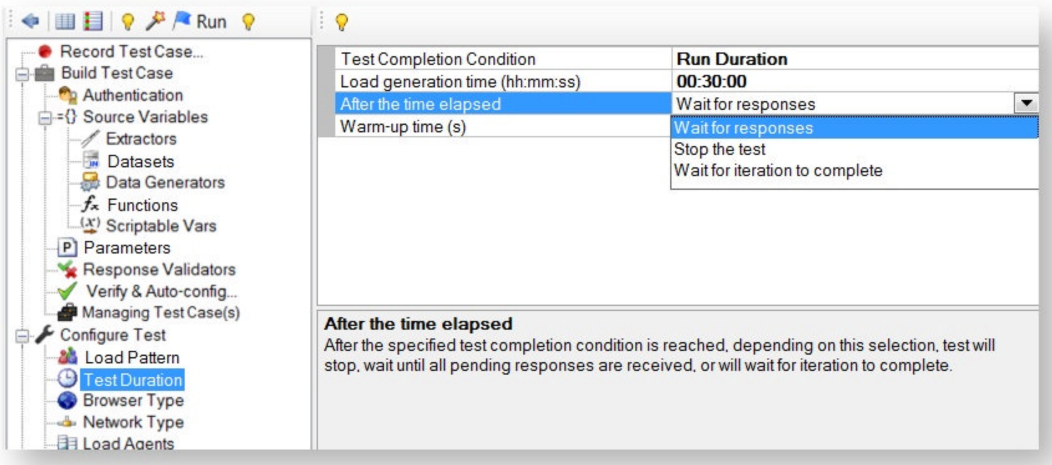
**How to count iterations**
Select Per VU to set the iteration limit for each VU. Select Total to set the overall iteration limit.

## 5.2.2 Run Duration

The test will complete after a specified period elapsed and the secondary condition is met.

1. Set the **Load generation time (hh:mm:ss)** property.
2. Set the secondary condition by configuring the **After the time elapsed** property by selecting one of the following

- 
    - **Wait for responses**: No new requests will be sent and VUs will wait for pending requests' responses.
    - **Stop the test**: No new requests will be sent and all pending requests will be aborted and excluded from calculations. This option is helpful when precise test duration is required; for example, when comparing two tests that should have the same duration.
    - **Wait for iterations to complete**: StresStimulus will send all unsent requests in the current iteration and wait for their responses. As a result, because all iterations complete, StresStimulus will accurately calculate iteration specific parameters such as **Avg. Iteration Time(s).**



| | |
|---|---|
| Test Completion Condition | Run Duration |
| Load generation time (hh:mm:ss) | 00:30:00 |
| After the time elapsed | Wait for responses |
| Warm-up time (s) | |

**After the time elapsed**
After the specified test completion condition is reached, depending on this selection, test will stop, wait until all pending responses are received, or will wait for iteration to complete.

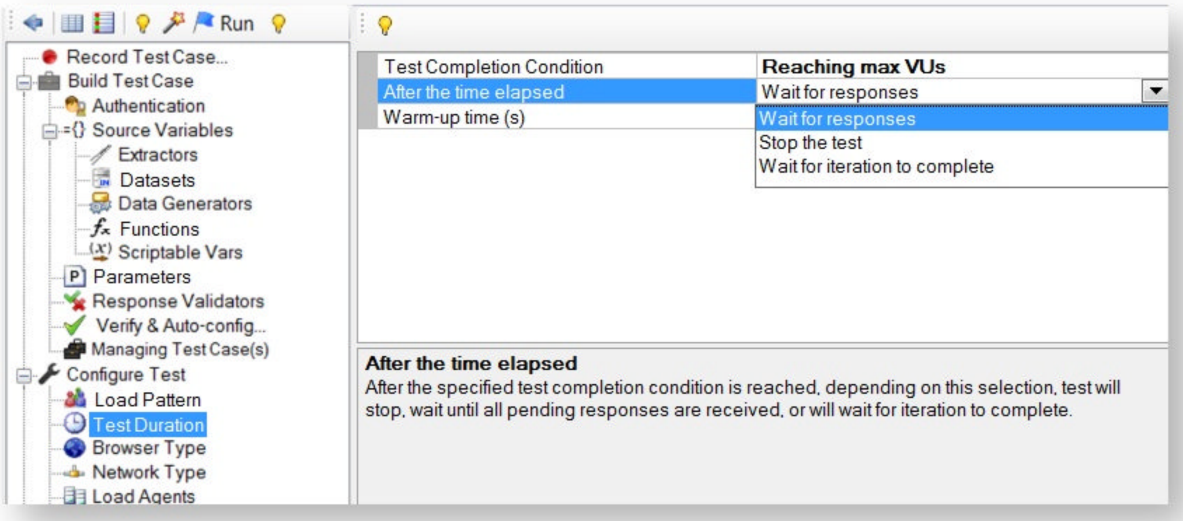
Here is an example: A test case consists of 10 requests and a test runs with one VU for 60 seconds. Let's say, after 60 seconds this VU completed 5 full iterations (which is 50 completed requests) and started iteration #6 by sending 7 requests and receiving 4 of them back.

- If you selected **Wait for Responses**, then VU1 will stop issuing requests and will wait only for the 3 responses which were not received back. After that, VU1 finishes the test with the following result: 6 iterations started / 5 iterations completed / 7 requests in the last iteration / No errors.
- If you selected **Wait for Iterations to Complete**, then VU1 will continue issuing the remaining 3 requests (to complete the iteration) and will wait to receive 6 responses back (the 3 responses which were not received plus 3 new responses) . After that, VU1 finishes the test with the following result 6 iterations started / 6 iterations completed.

If the test has more VUs, then StresStimulus will wait until the last VU finishes the test.

### 5.2.3 Reaching Max VUs

The test will end once the **Max VU** is reached in **Step Load**. For that reason, this can only be used with **Step Load.** As with **Run Duration**, you can set a mode for what to do when the **Max VUs** are reached.



| | |
|---|---|
| Test Completion Condition | **Reaching max VUs** |
| After the time elapsed | Wait for responses |
| Warm-up time (s) | |

Wait for responses
Stop the test
Wait for iteration to complete

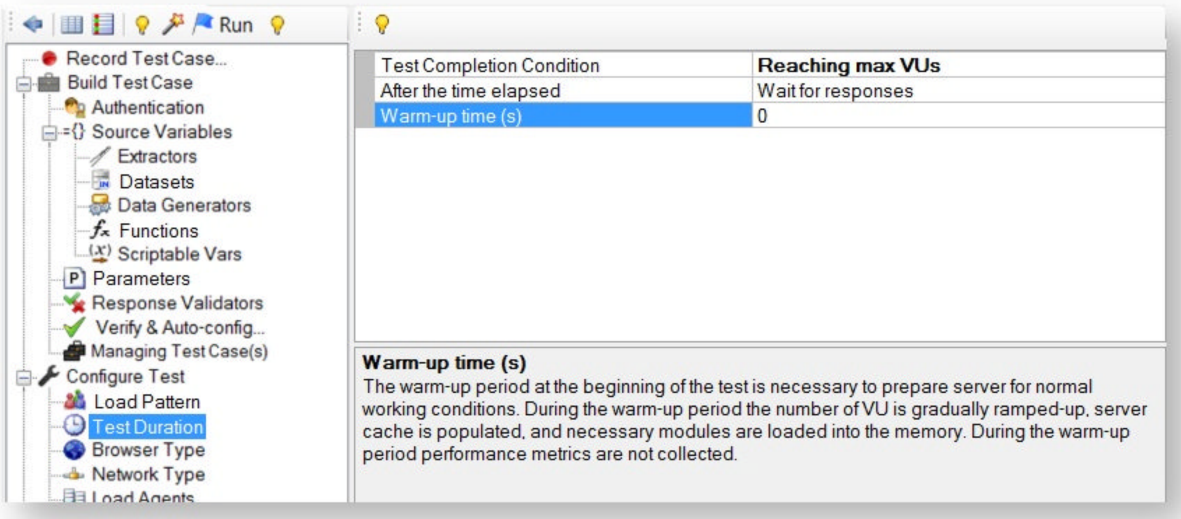
**After the time elapsed**
After the specified test completion condition is reached, depending on this selection, test will stop, wait until all pending responses are received, or will wait for iteration to complete.

## 5.2.4 Warm-up

A warm-up period is necessary to make sure that state of server resources, such as memory or cache, are prepared for normal operating mode. For example, after the server restarts, the web application will operate slower until all necessary processes are loaded into memory, and the database / application cache is primed.

During the warm-up period StresStimulus runs a test scenario, but ignores captured HTTP sessions and does not include them into the load test report. This allows to avoid performance measurement errors related to a cold server. After completion of the warm-up period, StresStimulus executes the load test normally.

Warm-up capability is available when **Test Duration** is set to **Run Duration** or **Reaching Max VUs**. During the warm-up period, the number of virtual users will gradually ramp-up to the starting user setting (**Number of VUs** in **Steady Load** or **Start VU** in **Step Load**). To set up the warm-up duration, set the **Warm-up(s)** property in the grid to the desired number of seconds. After the warm-up period is over, the same set of users will be used for the actual test. All requests, pages, transactions, iterations initiated during the warm-up time are excluded from the test metrics.



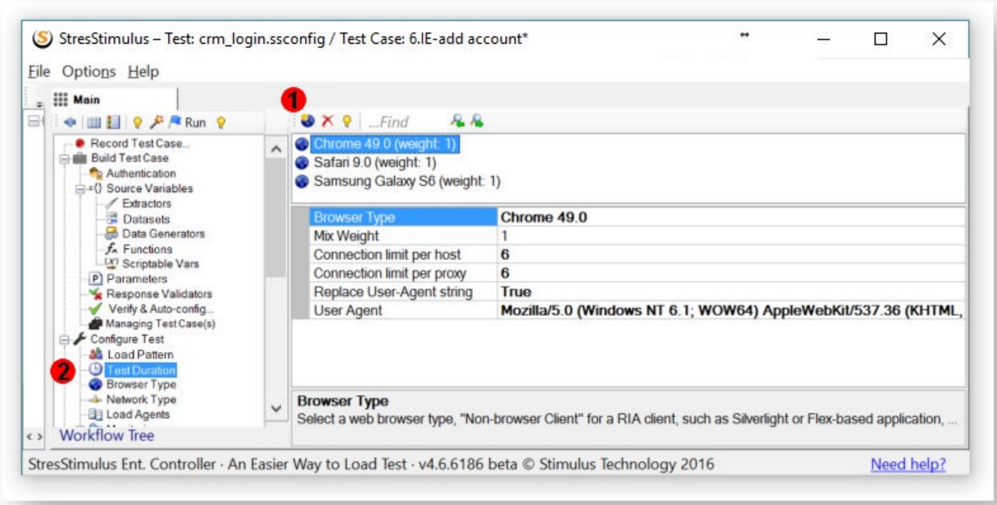
| | |
|---|---|
| Test Completion Condition | **Reaching max VUs** |
| After the time elapsed | Wait for responses |
| Warm-up time (s) | 0 |

**Warm-up time (s)**
The warm-up period at the beginning of the test is necessary to prepare server for normal working conditions. During the warm-up period the number of VU is gradually ramped-up, server cache is populated, and necessary modules are loaded into the memory. During the warm-up period performance metrics are not collected.

## 5.3 Browser Settings

One of the factors impacting users' experience is the browser type. For more realistic performance testing, StresStimulus emulates users working on different browsers. It allows to detect any issues that the tested website can exhibit in servicing requests from some browsers.

To add a new browser to the mix, click **Add** (1) in the **Browser Type** node (2) to display the **Browser Picker** dialog (3).



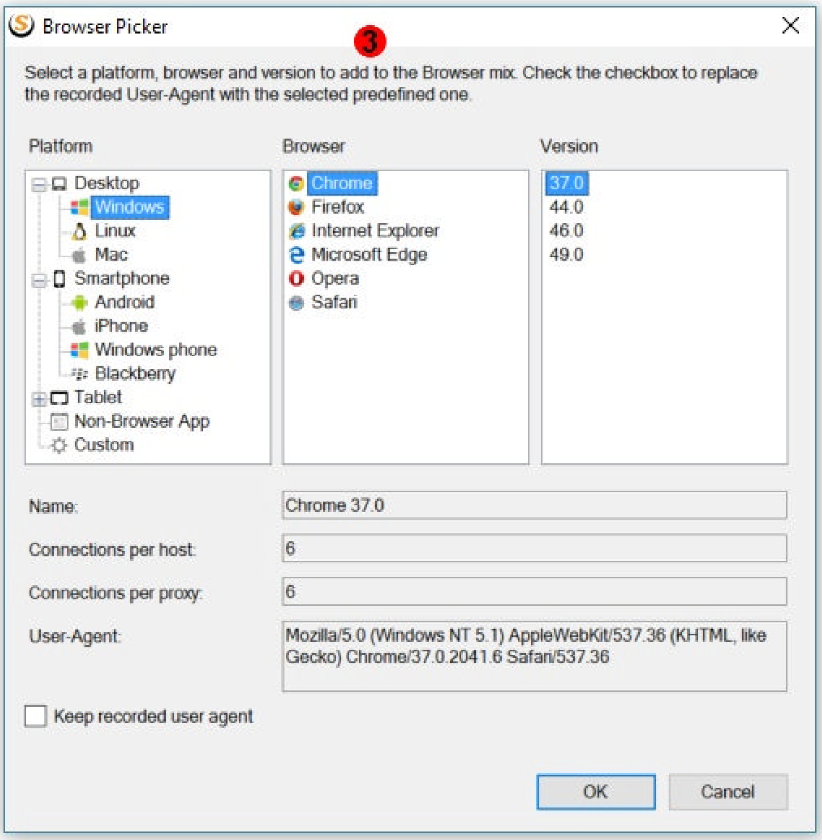
Chrome 49.0 (weight: 1)
Safari 9.0 (weight: 1)
Samsung Galaxy S6 (weight: 1)

| | |
|---|---|
| Browser Type | **Chrome 49.0** |
| Mix Weight | 1 |
| Connection limit per host | **6** |
| Connection limit per proxy | **6** |
| Replace User-Agent string | **True** |
| User Agent | **Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML,** |

**Browser Type**
Select a web browser type, "Non-browser Client" for a RIA client, such as Silverlight or Flex-based application, ...

StresStimulus Ent. Controller · An Easier Way to Load Test · v4.6.6186 beta © Stimulus Technology 2016 Need help?

In the appeared browser picker, select the platform, browser and version of the browser you want to add. If you would like to keep the recorded user agent headers, instead of replacing it was the selected browser user agent, check the appropriate box.



Select a platform, browser and version to add to the Browser mix. Check the checkbox to replace the recorded User-Agent with the selected predefined one.

| Platform | Browser | Version |
|---|---|---|
| Desktop | Chrome | 37.0 |
| Windows | Firefox | 44.0 |
| Linux | Internet Explorer | 46.0 |
| Mac | Microsoft Edge | 49.0 |
| Smartphone | Opera | |
| Android | Safari | |
| iPhone | | |
| Windows phone | | |
| Blackberry | | |
| Tablet | | |
| Non-Browser App | | |
| Custom | | |

Name: Chrome 37.0

Connections per host: 6

Connections per proxy: 6

User-Agent: Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/37.0.2041.6 Safari/537.36

☐ Keep recorded user agent

After adding a new browser you can change the following properties In the property grid:

- **Browser Type** is the name of the browser.
- **Mix Weight** is the relative frequency (in units or percent) each VU will use that browser type. It represents the relative size of the group of users working with this browser.
- **Connection limit per host** is the number of maximum simultaneous connections (requests) a single VU can create to a single host.
- **Connection limit per proxy** is the total number of maximum simultaneous connections (requests) a single VU can create.
- **Replace User-Agent string** indicates if the recorded User-Agent request header should be replaced with the browser's value.
- **User Agent** is the given User-Agent value for the browser.

**Note:** How StresStimulus emulates web browsers:

Click For Details...

(a) Maintains request concurrency based on specific connection limits for the selected browser type. For non-browser application, the limit is 1.

(b) Injects the appropriate user-agent string into the requests;

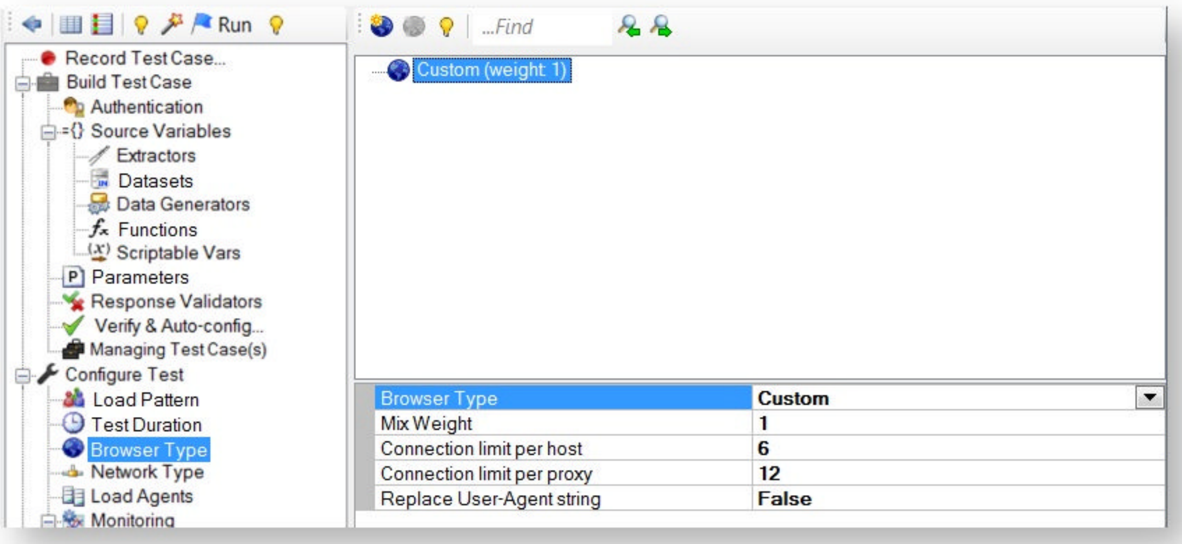
(c) Maintains the browser mix distribution, if more than one browser is selected.

### 5.3.1 Connections Per Host and Proxy

HTTP connection limits per host (server) and per-proxy are enforced based on the selected web browser.

A web browser typically establishes several concurrent connections to the server so that the page resources are loaded faster.

StresStimulus emulates such behavior and opens several concurrent connections to load dependent requests. Depending on the emulated browser type, StresStimulus sets limits on **Requests per Host** (maximum simultaneous requests for each domain) and **Requests per Proxy** (maximum simultaneous requests to all domains). If you wish to set your own limits, set the **Browser Type** to **Custom** and set the desired connections.

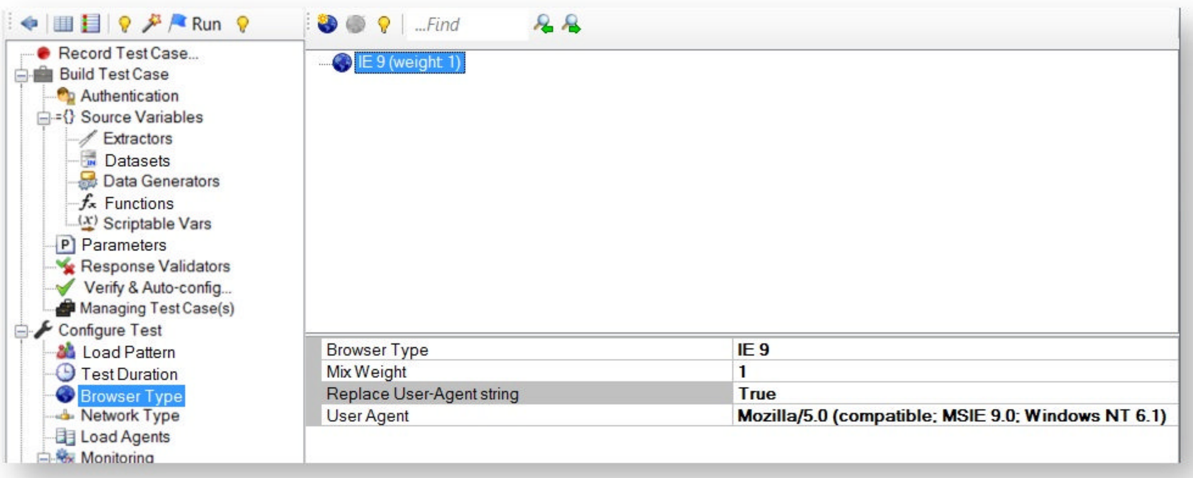


| | |
|---|---|
| Browser Type | Custom |
| Mix Weight | 1 |
| Connection limit per host | 6 |
| Connection limit per proxy | 12 |
| Replace User-Agent string | False |

## 5.3.2 User-Agent

The browser sends the User-Agent request header, which includes information about the browser type and the features it supports (e.g. compression), as well as the client operating system and installed frameworks and software. This information can impact request processing and performance.

By default, StresStimulus executes the test case without modifying the recorded User-Agent request header. To change the user agent, set the **Replace User Agent String** property to **True**. A generic string for the selected browser type will appear in the **User Agent** property. You can further change the string to specify the desired client configuration (e.g. a .NET framework version) to be used for simulating VUs. These User-Agent settings affect only VUs assigned to emulate this browser type. To change user agents for other uses, make the appropriate changes in the User-Agent for other browsers.



| | |
|---|---|
| Browser Type | IE 9 |
| Mix Weight | 1 |
| Replace User-Agent string | True |
| User Agent | Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1) |

**Note:** StresStimulus emulates only the most important browser factors impacting website scalability, such as connection limit, caching and user-agent. Other browser factors, such as creating web page DOM and executing JavaScript, are not emulated.

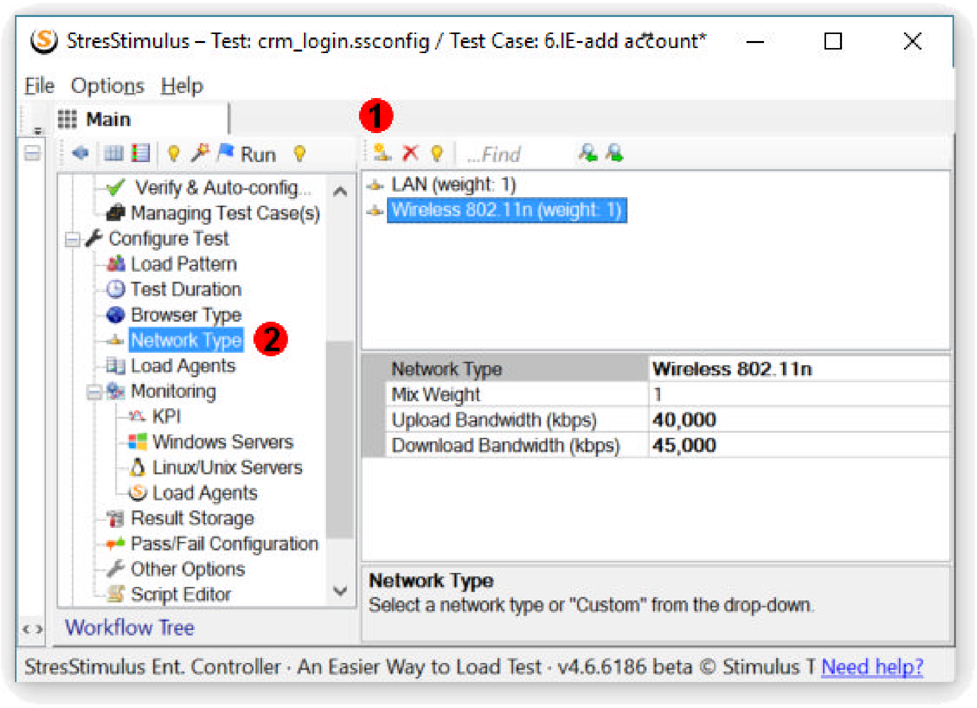
**Tip:** If a webpage was substantially modified after recording the test case by adding new external resources, then during the load test the recorded resources will be not be requested. In order to test a website's performance with new resources, rerecorded the test case.

**See Also:** You can also use the test case configuration technique described in Emulating Clicks on Random Links.

## 5.4 Network Settings

In the real world, different users will likely have different Internet connection speeds provided by their ISPs. StresStimulus can simulate such uneven network bandwidth distribution by creating a mix of several network types, providing various bandwidths. To add a new Network to the mix:

Click **Add** (1) in the **Network Type** (2) tree node to display the **Network Picker** dialog (3).

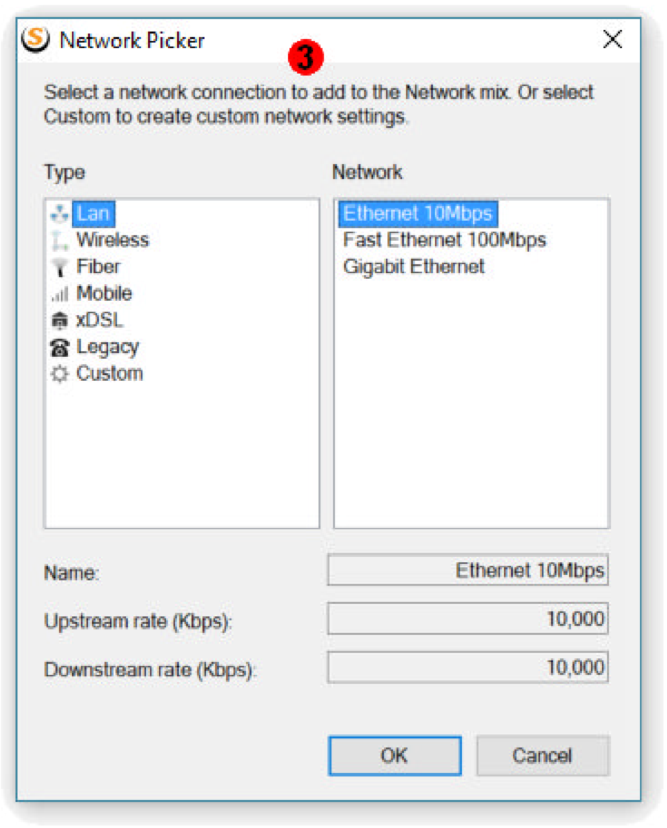


| Network Type | Wireless 802.11n |
|---|---|
| Mix Weight | 1 |
| Upload Bandwidth (kbps) | 40,000 |
| Download Bandwidth (kbps) | 45,000 |

**Network Type**
Select a network type or "Custom" from the drop-down.

StresStimulus Ent. Controller · An Easier Way to Load Test · v4.6.6186 beta © Stimulus T Need help?

Select the type of network you want to add. The text boxes will display the selected network's upstream and downstream rates. Or select Custom to create your own network and provide the upstream and downstream rate.


Network Picker

3

Select a network connection to add to the Network mix. Or select Custom to create custom network settings.

Type: Lan, Wireless, Fiber, Mobile, xDSL, Legacy, Custom

Network: Ethernet 10Mbps, Fast Ethernet 100Mbps, Gigabit Ethernet

Name: Ethernet 10Mbps

Upstream rate (Kbps): 10,000

Downstream rate (Kbps): 10,000

**Note:** A Network type is emulated by injecting a certain wait time into every request and response, weighted to its size and the network type bandwidth.

Once the new network appears, configure its Mix Weight which is the relative frequency (in units or percent) each VU will use that network type. It represents the relative size of the group of users working with this network.

**Tip:** Mix weight values can be either percentage or weight.

The network emulation can only slow-down the traffic and cannot speed it up. It is recommended to use different network type other than LAN only when the test machine is connected to the Web server through a fast network (LAN). For example, if your physical connection is DSL, then the emulation of the dial-up will be not accurate.

## 5.5 Server and Agent Monitoring

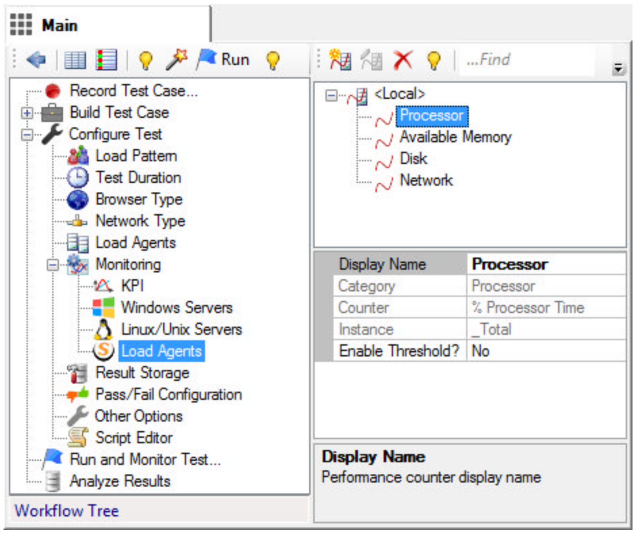
Ent SP

In addition to monitoring Key Performance Metrics (KPI) as well as page and transaction performance, StresStimulus has the instrumentation to profile the performance of the remote machines involved in the test through the local network or Internet. Performance counters are mainly used for monitoring resource utilization on multiple tested servers including Web, application and database servers, to help in load test diagnostics. They allow to estimate the impact of the users' activity on hardware, operating system an application and help to track down the source of a website's unproductiveness. Performance counters are also used to monitor the health of load agents on client testing machines to make sure that the test rig hardware resources are provisioned correctly.

Three groups of counters can be monitored:

1. Performance counters on Windows servers
2. Performance counters on Linux/Unix servers via SNMP protocol
3. Performance counters on load agents

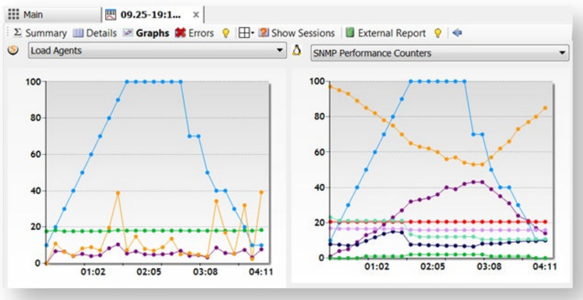
To configure each group of counters, select an appropriate node under the **Configure Test** -> **Monitoring** section of the workflow tree.



| Display Name | Processor |
|---|---|
| Category | Processor |
| Counter | % Processor Time |
| Instance | _Total |
| Enable Threshold? | No |

**Display Name**
Performance counter display name

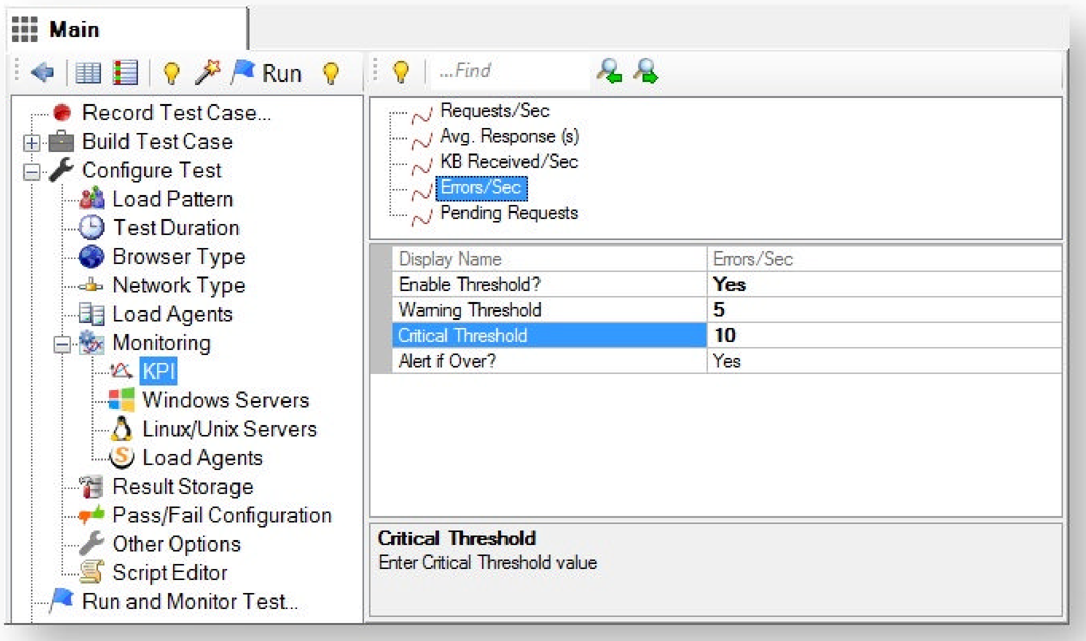
Each group of counters is displayed in a separate graph panel, presenting a separate curve for each parameter. They are displayed on the Runtime Dashboard during the test run and on the test results to help in load test diagnostics.



### 5.5.1 KPI Thresholds

Ent SP To simplify monitoring key performance indicators (KPIs), you can define threshold rules. A threshold is a borderline value associated with a KPI. When the actual value of the KPI crosses the borderline, an event is triggered. Such events are registered and presented as part of performance analytics.

Handling the KPI thresholds is the same as handling the server or agent performance counters thresholds, described in the Threshold Rules section.



| | |
|---|---|
| Display Name | Errors/Sec |
| Enable Threshold? | **Yes** |
| Warning Threshold | **5** |
| Critical Threshold | **10** |
| Alert if Over? | Yes |

**Critical Threshold**
Enter Critical Threshold value

### 5.5.2 Windows Servers Monitoring

StresStimulus uses the same API as Windows performance monitor (**Perfmon).** Therefore, any Windows performance counters can be monitored. Since the Windows networking and instrumentation infrastructure is utilized, there is no need to install any additional software on remote servers.

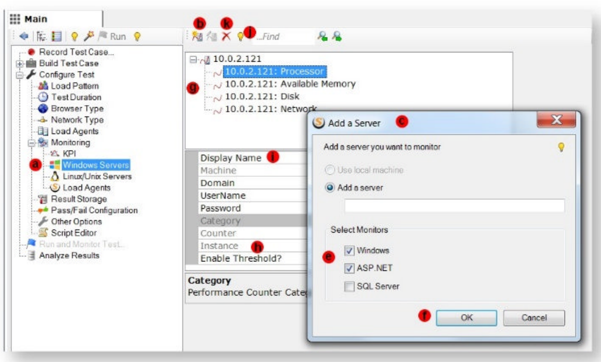
#### 5.5.2.1 Adding Monitors

A **Monitor** is a predefined set of server performance counters. To add monitors:

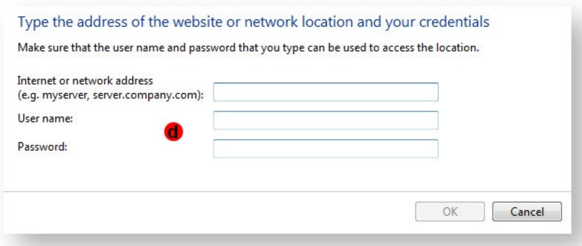
1. Navigate to the **Windows Servers** (a) section located under the **Monitoring** node.

2. On the toolbar, click **Add a machine** (b).

3. In the appeared dialog (c), enter server IP address or computer name without "//". Example: 10.2.2.169 or WEB_SRV5



> **Note:** You can add a web server targeted in the test. You can also add application or database servers that are not directly targeted by HTTP requests issued by the load generator(s).


Type the address of the website or network location and your credentials

Make sure that the user name and password that you type can be used to access the location.

Internet or network address
(e.g. myserver, server.company.com):

User name:

Password:

4. You may be asked to provide server credentials. Provide the credentials with sufficient permissions. For example, use an account that is part of the Performance Monitor User group.

If you failed to connect to the server, follow these steps:

- Navigate to Control Panel -> User Accounts -> Manage Your Credentials

- In the Credential Manager, click "Add a Windows credentials link" and add the server account credentials. If the account already exists, update it with the correct credentials (d) or remove and re-create it.

5.Depending on server functions you wish to monitor, you can select one or several **Monitors** by checking boxes (e). The list of supported monitors and counters (along with corresponding Windows performance counters definition) is provided below.

**Windows:**

-

- Processor - {machine name} > Processor > % Processor Time > _Total
- Available memory - {machine name} > Memory > Available Mbytes
- Disk - {machine name} > PhysicalDisk > % Disk Time > _Total
- Network - {adapter name} > Network Interface > Packets/sec

**Note:** StresStimulus attempts to find the network adapter automatically. If it can't be found then no adapter is automatically added. It can still be added.

**ASP.NET:**

- 
    - Requests Queued
    - Request Wait Time
    - Request Execution Time

**SQL Server:**

- 
    - SQLServer Memory
    - SQLServer Cache
    - SQLServer User Connections

6. Click OK (f). A new server with the selected performance counters (g) will be added.

You can add several servers one after another.

7. Performance counters' properties are displayed in the property grid (h). To rename a counter, change its **Display Name** property (i)

8. To delete a selected counter or server, click **Delete** (j) on the toolbar.

9. To add performance counters to the selected server, click **Edit** (k) on the toolbar and follow the instructions on the next page.

### 5.5.2.2 Adding More Performance Counters

The basic set of performance counters described in the previous section is a convenient way to quickly add a server for monitoring.

You can extend the basic set by aiding more counters in the areas you wish to monitor.

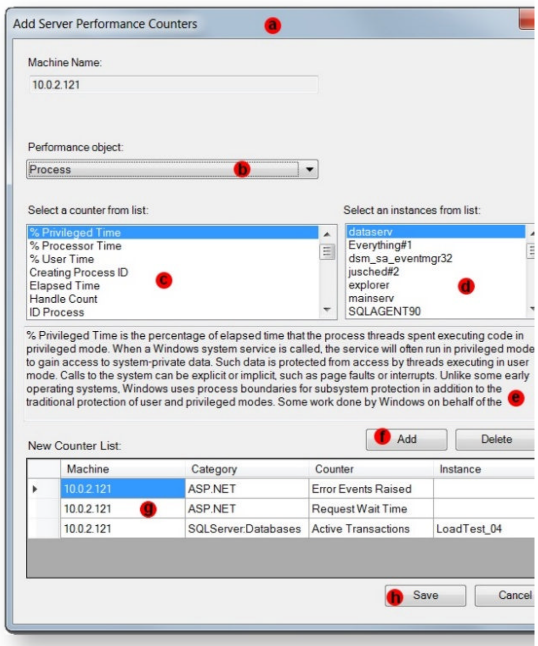
To add more performance counters to the selected server

1. Click edit on the toolbar. The **Add Performance Counters** dialog (a) will appear
2. From the drop-down (b), select the performance objects to monitor.
3. Select the specific counter (c).
4. You may need to select an instance of the counter (d).
5. Highlight the counter to see its description (e).
6. Click **Add** (f) to add the counter to the New Counter List (g).
7. Click **Delete** to remove the highlighted counters from the New Counter List.
8. When finished adding new counters, click **Save** (h) to add the New Counter List to the Test.

The selected set of performance counters is stored with the test configuration. After reopening a test, its list of performance counters will be retrieved

**Info:** There are many sources explaining Windows performance counters, for example:

- System http://technet.microsoft.com/en-us/library/cc768048.aspx
- ASP.NET http://msdn.microsoft.com/en-us/library/fxk122b4.aspx
- SQL Server http://www.extremeexperts.com/sql/articles/sqlc



Machine Name: 10.0.2.121

Performance object: Process

Select a counter from list:
% Privileged Time
% Processor Time
% User Time
Creating Process ID
Elapsed Time
Handle Count
ID Process

Select an instances from list:
dataserv
Everything#1
dsm_sa_eventmgr32
jusched#2
explorer
mainserv
SQLAGENT90

% Privileged Time is the percentage of elapsed time that the process threads spent executing code in privileged mode. When a Windows system service is called, the service will often run in privileged mode to gain access to system-private data. Such data is protected from access by threads executing in user mode. Calls to the system can be explicit or implicit, such as page faults or interrupts. Unlike some early operating systems, Windows uses process boundaries for subsystem protection in addition to the traditional protection of user and privileged modes. Some work done by Windows on behalf of the

New Counter List:

| Machine | Category | Counter | Instance |
|---|---|---|---|
| 10.0.2.121 | ASP.NET | Error Events Raised | |
| 10.0.2.121 | ASP.NET | Request Wait Time | |
| 10.0.2.121 | SQLServer:Databases | Active Transactions | LoadTest_04 |

ounters.aspx

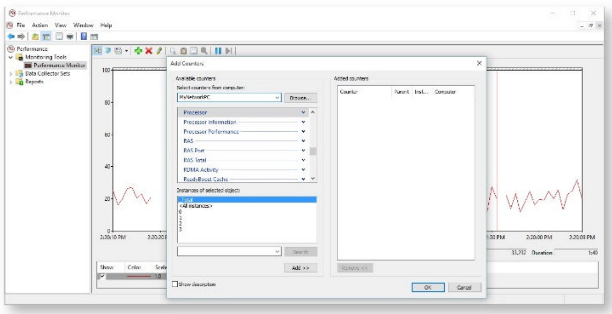
- Microsoft Dynamics CRM http://www.microsoft.com/en-us/download/details.aspx?id=27119

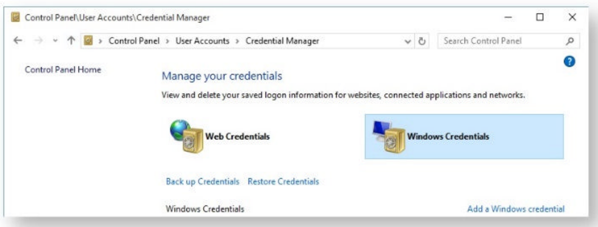
**Note:** If you added the performance counters to the test a while ago, they must be valid in order to work. For example, performance counters on a remote server will work only if the same server name or IP address still exists and the server is accessible with the saved credentials.

### 5.5.2.3 Troubleshooting Monitors

The Windows monitoring module in StresStimulus is built on the top of Windows Perfmon API. If you have issues adding monitors, then they are most likely related to your Windows domain/security infrastructure.

Troubleshooting such issues should be done outside StresStimulus. First make sure that you can monitor your remote server with Perfmon. Launch Perfmon on the StresStimulus machine and try adding performance counters from the server you want to monitor. When troubleshooting such issues, use all available support resources related to monitoring performance of a remote Windows machine. Typically such resources are part of Windows server documentation or related articles and blogs.





Manage your credentials

View and delete your saved logon information for websites, connected applications and networks.

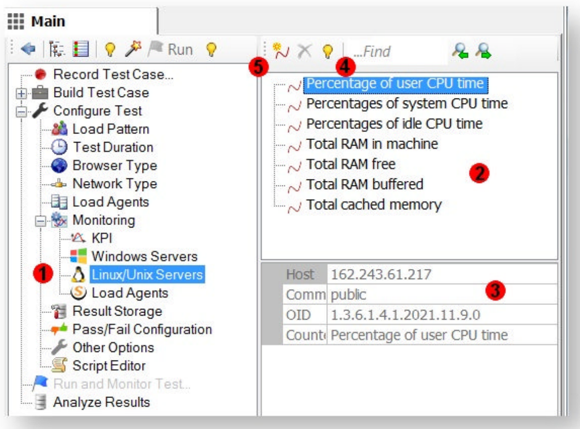
Windows Credentials

| | |
|---|---|
| After you made Perform work to monitor the remote server, use the same server name or IP address and user credentials in StresStimulus. The resolution might require checking various computer and domain credentials, account permissions, domain and local policies, intermediate firewalls, etc. It all depends on your infrastructure settings.<br><br>Here is just one example that you can check: If the Perfmon connection does not work, in Control Panel on the StresStimulus machine go to Credential Manager. Click Add a Windows credentials and add the remote server account information. After that, try the Perfmon connection again.<br><br>Once Perfmon on the StresStimulus machine can connect to your server, StresStimulus would be able to connect as well. | |

## 5.5.3 Linux/Unix Servers Monitoring

| | |
|---|---|
| StresStimulus can monitor various metrics of Linux/Unix-based servers during the load test execution using SNMP protocol. The next two sections provide examples of configuring SNMP services on different versions of Linux.<br><br>To configure monitoring performance counters, navigate to the **Linux/Unix Servers** section (1) located under the **Monitoring** node. Any added previously counters and their properties are displayed in the tree (2), and the property grid (3).<br><br>To delete a selected counter or all counters, | |

click the appropriate delete button (4).



| | |
|---|---|
| Host | 162.243.61.217 |
| Comm | public |
| OID | 1.3.6.1.4.1.2021.11.9.0 |
| Count | Percentage of user CPU time |

To add one or several counters, click Add (5) to start **Add SNMP Counters** dialog (6). After that:

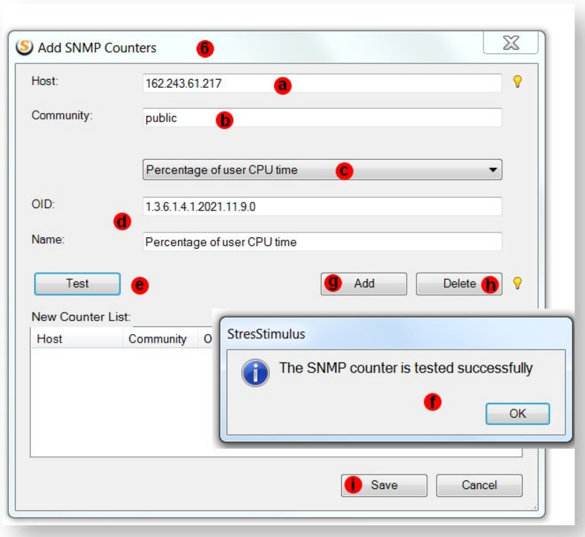
1. Enter a host IP address or domain name without "//". Example: 10.2.2.169 or WEB_SRV5.
2. Change Community, if necessary.
3. Select a counter from the drop-down. The boxes OID and Name will be populated
4. To add a counter which is not on the list, enter its OID and Name.

Tip: To find other system and application performance counters available via SNMP protocol, search for MIB object definitions, available from multiple sources, for example http://www.mibdepot.com/ .

1. Click Test.
2. Make sure that the object is available and the SNMP connection works.

Info: For configuring and troubleshooting SNMP protocol on the server, see the next sections

g. Click **Add** to add the counter to the New Counter List.



Host: 162.243.61.217

Community: public

Percentage of user CPU time

OID: 1.3.6.1.4.1.2021.11.9.0

Name: Percentage of user CPU time

The SNMP counter is tested successfully

| | |
|---|---|
| h. Click **Delete** to remove the highlighted counters from the New Counter List.<br><br>i. When finished adding new counters, click **Save** to add the New Counter List to the Test.<br><br>The selected set of performance counters is stored with the test configuration. After reopening a test, its list of performance counters will be retrieved. | |

#### 5.5.3.1 CentOS/Red Hat Configuration

1. FIREWALL

If you use an external firewall make sure the following ports are open: 22 (SSH, tcp), 161(SNMP, tcp/udp), 53 (DNS, udp), 80 and 443 (HTTP HTTPS, tcp), ICMP (ping)

2. INSTALLATION.

To install SNMP Agent Daemon and SNMP clients, use Yum package manager available on all RedHat Operating systems. Login into the server as a **root** user and execute the following:

```
~]# yum -y install net-snmp net-snmp-libs net-snmp-utils
```

Tip: If you are using Sudo then do “sudo su - root” or “su root” to become a root user.

3. CONFIGURATION

- Move snmd.conf to snmpd.conf_bak:(in order to backup defaults)

  ```
  ~]# cd /etc/snmp/
   ~]# mv snmpd.conf snmpd.conf_bak
  ```

- **Add** the following configuration to new snmpd.conf file (replace “My Location” and “My Name” with your data):

  ```
  ~]# echo 'syslocation "My Location"' >> snmpd.conf
  ~]# echo 'syscontact "My Name"' >> snmpd.conf
  ~]# echo '' >> snmpd.conf
  ~]# echo 'rocommunity public' >> snmpd.conf
  ~]# echo '' >> snmpd.conf
  ~]# echo 'disk /' >> snmpd.conf (this is for disk counters)
  ```

4. STARTING THE SERVICE

- To run the **snmpd** service in the current session, enter the following at a shell prompt as a **root**:

```
~]# service snmpd start
```

- To configure the service to be automatically started at boot time, use the following command:

```
~]# chkconfig snmpd on
```

5. TESTING, TROUBLESHOOTING

- To make sure SNMP listens on valid port and to all IPs execute this command

```
~]# netstat -nepl
```

- If SNMP listener works only on 127.0.0.1 IP addresses, try to restart snmpd daemon (~]# service snmpd restart) .
- If the previous step does not help, check “SNMPDOPTS=” string at **/etc/default/snmpd** and make sure it looks like this

```
SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -g snmp -I -smux -p /var/run/snmpd.pid -c
/etc/snmp/snmpd.conf'
```

- Make sure SNMP is installed properly. For example, the following **snmpwalk** command shows the **system** tree with a default agent configuration.

```
~]# snmpwalk -v2c -c public localhost system
```

Output should look like this:

```
public static void main(String[] args) { System.out.println("
SNMPv2-MIB::sysDescr.0 = STRING: Linux ip-10-239-15-166 2.6.32-358.14.1.el6.x86_64 #1
SMP Mon Jun 17 15:54:20 EDT 2013 x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (1013444) 2:48:54.44
SNMPv2-MIB::sysContact.0 = STRING: "My Name"
SNMPv2-MIB::sysName.0 = STRING: ip-10-239-15-166
SNMPv2-MIB::sysLocation.0 = STRING: "My Location"
"); }
```

- Select a counter by completing the **Add SNMP Counters** dialog, as described in the previous section, and click **Test**. You should receive "The SNMP counter is tested successfully" message.

- If instead you receive "Can't connect to the host" message, make sure that port 161 is open for UDP and TCP traffic on the Linux server. To open these ports, execute the following commands:

```
public static void main(String[] args) { System.out.println("
iptables -I INPUT -p udp --dport 161 -j ACCEPT
iptables -I INPUT -p tcp --dport 161 -j ACCEPT
iptables -I FORWARD -p udp --dport 161 -j ACCEPT
iptables -I FORWARD -p tcp --dport 161 -j ACCEPT
 iptables-save
"); }
```

### 5.5.3.2 Ubuntu / Debian Configuration

1. FIREWALL

If you use external firewall make sure the following ports are open: 22 (SSH, tcp), 161(SNMP, tcp/udp), 53 (DNS, udp), 80 and 443 (HTTP HTTPS, tcp), ICMP (ping)

2. INSTALLATION.

- To install SNMP daemon, utilities and libs run this command

```
~# apt-get install snmpd snmp  smistrip
```

**Tip:** If you are using sudo then do “sudo su - root” or “su root” to become root user

3. CONFIGURATION

- Move snmd.conf to snmpd.conf_bak:(in order to backup defaults)

```
~]# cd /etc/snmp/
 ~]# mv snmpd.conf snmpd.conf_bak
```

- **Add** the following configuration to new snmpd.conf file (replace “My Location” and “My Name” with your data):

```
~]# echo 'syslocation "My Location"' >> snmpd.conf
~]# echo 'syscontact "My Name"' >> snmpd.conf
~]# echo '' >> snmpd.conf
~]# echo 'rocommunity public' >> snmpd.conf
~]# echo '' >> snmpd.conf
~]# echo 'disk /' >> snmpd.conf
```

- Open **/etc/default/snmpd**

```
~]# nano /etc/default/snmpd
```

- Find “SNMPDOPTS=” parameter. On different systems the configuration strings can be different, but generally it should look like

```
SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -g snmp -I -smux -p /var/run/snmpd.pid'
or
SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -I -smux -p /var/run/snmpd.pid 127.0.0.1'
```

- Change the configuration strings to

```
SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -g snmp -I -smux -p /var/run/snmpd.pid -c
/etc/snmp/snmpd.conf'
```

## 4. STARTING THE SERVICE

To run the **snmpd** service in the current session, enter the following at a shell prompt as **root**:

```
~]# service snmpd start
```

## 5. TESTING, TROUBLESHOOTING

- To make sure SNMP listens on valid port and to all IPs execute this command

```
~]# netstat –nepl
```

- If SNMP listener works only on 127.0.0.1 IP addresses, try to restart snmpd daemon (**~]# service snmpd restart**) .
- If the previous step does not help, check “SNMPDOPTS=” string at /etc/default/snmpd and make sure it looks like this

  SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -g snmp -I -smux -p /var/run/snmpd.pid -c /etc/snmp/snmpd.conf'

- Make sure SNMP is installed properly. For example, the following **snmpwalk** command shows the **system** tree with a default agent configuration.

```
~]# snmpwalk -v 2c -c public localhost .1.3.6.1.2.1.1
```

Output should look like this

```
SNMPv2-MIB::sysDescr.0 = STRING: Linux snmptest.com 3.8.0-29-generic #42~precise1-
Ubuntu SMP Wed Aug 14 16:19:23 UTC 2013 x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (52495) 0:08:44.95
SNMPv2-MIB::sysContact.0 = STRING: "My Name"
SNMPv2-MIB::sysName.0 = STRING: snmptest.com
SNMPv2-MIB::sysLocation.0 = STRING: "My Location"
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORID.1 = OID: SNMP-FRAMEWORK-
MIB::snmpFrameworkMIBCompliance
SNMPv2-MIB::sysORID.2 = OID: SNMP-MPD-MIB::snmpMPDCompliance
SNMPv2-MIB::sysORID.3 = OID: SNMP-USER-BASED-SM-MIB::usmMIBCompliance
```

- Select a counter by completing the **Add SNMP Counters** dialog, as described in the previous section, and click **Test**. You should receive a "The SNMP counter is tested successfully" message.
- If instead you receive a "Can't connect to the host" message, make sure that port 161 is open for UDP and TCP traffic on the Linux server. To open these ports, execute the following commands:

```
iptables -I INPUT -p udp --dport 161 -j ACCEPT
iptables -I INPUT -p tcp --dport 161 -j ACCEPT
iptables -I FORWARD -p udp --dport 161 -j ACCEPT
iptables -I FORWARD -p tcp --dport 161 -j ACCEPT
 iptables-save
```

### 5.5.4 Load Agents Monitoring

Load Agents are executed on Windows machines or cloud instances. Their monitoring is configure similarly to Windows Server(s) Monitoring

### 5.5.5 Threshold Rules

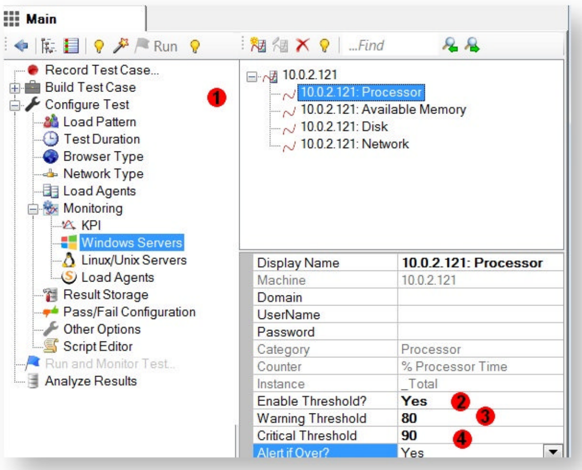
To simplify monitoring servers and load agents, you can define threshold rules. A threshold is a borderline value associated with a performance counter. When the actual value of the counter crosses the borderline, an event is triggered. Such events are registered and presented as part of performance analytics.

Thresholds can be created for any of the server or agent performance counters. To create a threshold rule, follow these steps:

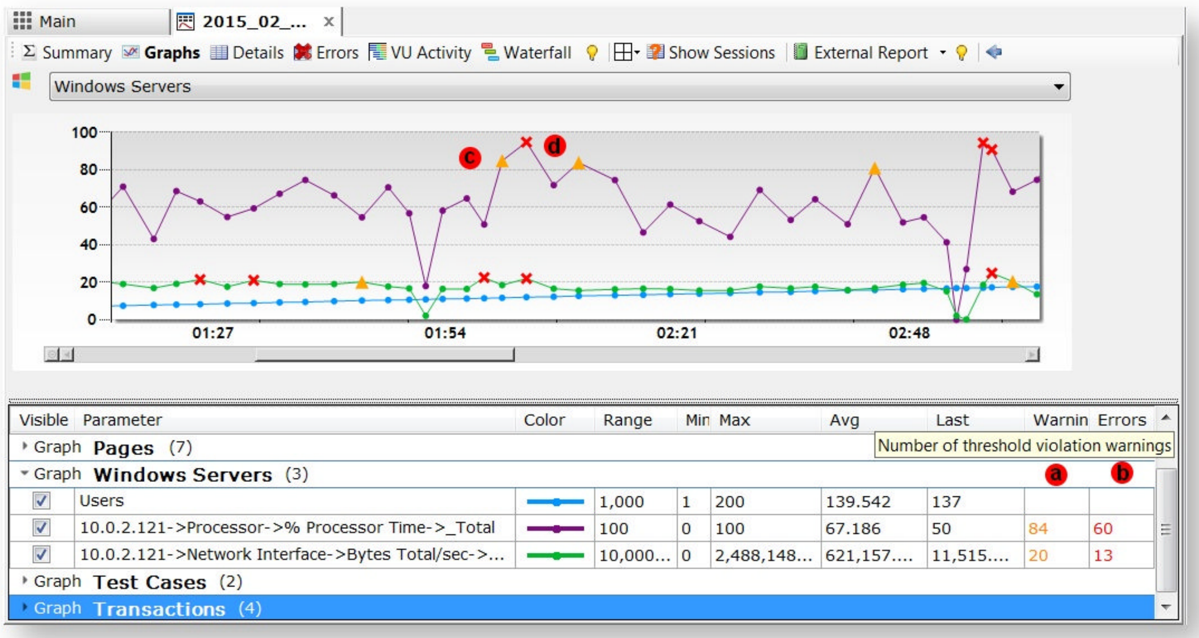
1. Select a server or agent performance counter
2. In the property grid, Set **Enable Threshold** property to **Yes.**
3. Enter two threshold values: **Warning Threshold** to monitor warnings and **Critical Threshold** to monitor errors.
4. If thresholds reflect high-values, set **Alert if Over** to **Yes** to indicate that exceeding a threshold is a problem. If thresholds reflect low-values, set **Alert if Over** to **No** to indicate that falling below a threshold is a problem.

During a load test execution, you can analyze violations that occur for the configured threshold rules. Results of threshold monitoring are presented in the Runtime Dashboard and test reports on the curve grid below the performance counter graph. Warnings are displayed in orange (a), errors are displayed in red (b).

Curves on which the threshold is defined, will display warning violation as orange thriangles(c) and critical violations as a red crosses (d).



| | |
|---|---|
| Display Name | **10.0.2.121: Processor** |
| Machine | 10.0.2.121 |
| Domain | |
| UserName | |
| Password | |
| Category | Processor |
| Counter | % Processor Time |
| Instance | _Total |
| Enable Threshold? | **Yes** |
| Warning Threshold | **80** |
| Critical Threshold | **90** |
| Alert if Over? | Yes |



| Visible | Parameter | Color | Range | Min | Max | Avg | Last | Warnin | Errors |
|---|---|---|---|---|---|---|---|---|---|
| Graph | **Pages** (7) | | | | | | | | |
| Graph | **Windows Servers** (3) | | | | | | | | |
| ✓ | Users | 🟦 | 1,000 | 1 | 200 | 139.542 | 137 | | |
| ✓ | 10.0.2.121->Processor->% Processor Time->_Total | 🟪 | 100 | 0 | 100 | 67.186 | 50 | 84 | 60 |
| ✓ | 10.0.2.121->Network Interface->Bytes Total/sec->... | 🟩 | 10,000... | 0 | 2,488,148... | 621,157.... | 11,515.... | 20 | 13 |
| Graph | **Test Cases** (2) | | | | | | | | |
| Graph | **Transactions** (4) | | | | | | | | |

## 5.6 Test Result Storage

Ent SP

Test results are stored in a database. There are four types of settings that can be configured:

- **How much data to store:** Specify what type of information should be saved (reports, test log, etc.).
- **Session content:** Specify how many details to store in the test log.
- **Data storage medium:** Specify what database type to use.
- **Agent to controller replication:** Specify if you want to copy information to the controller about sessions issued by the agents with SQL Server CE-based storage. This information is necessary for waterfall charts.

1. Go to **Result Storage** node in the **Workflow Tree** .

2. Configure what data to store by selecting one of the **How Much Data to Store** property values:

-
    - **All:** All the resulting data, including a summary report, all result grids, graphs, and all request/response session info (this is the largest part).Use this mode if you would like to view individual request/response info of replayed sessions.

- **Partial:** All the resulting data, including a summary report, all result grids, graphs, but no request/response session info.
- **None:** No info result info is stored. You can view the resulting summary report, all result grids and graphs, but they will be deleted once you run another test or close the test. Use this mode for quick practice tests. No other settings will apply in this mode.

3. Select what database type to use by selecting **Data Storage** .

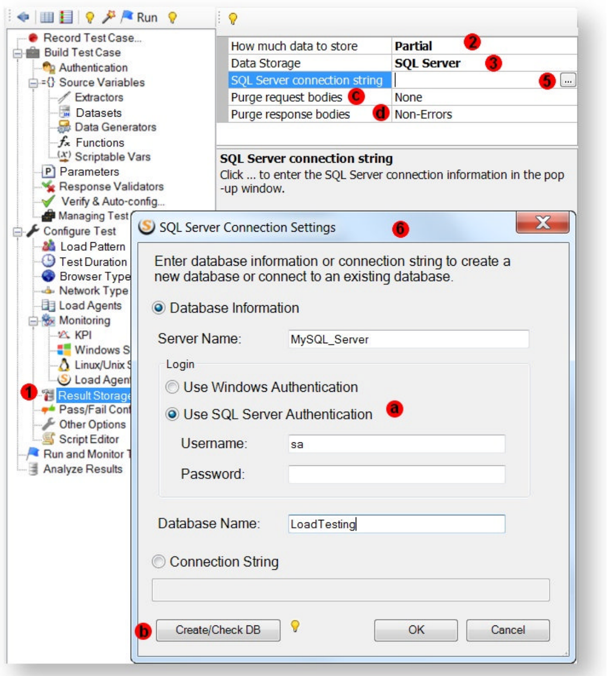
•

- **Embedded SQL Server CE:** Every test run will have its own database file with up to 4 GB of data. It can store a test log with up to 4 million sessions with the average session size (excluding the response body) up to 1 KB.
- **SQL Server:** Provide your SQL Server to store virtually unlimited test logs.

4. If **Embedded SQL Server CE:** is selected, the property **Save sessions from agents** will be available. In distributed tests with SQL Server CE-based storage, the content of the sessions generated on the agents is stored on the agents.

•

- Select **Yes**, to copy this content to the controller. This will allow generating waterfall charts for VUs emulated on the agent.
- Select **No**, to reduce the traffic between agents and controller when the network bandwidth is limited.



| | |
|---|---|
| How much data to store | **Partial** |
| Data Storage | **SQL Server** |
| SQL Server connection string | |
| Purge request bodies | None |
| Purge response bodies | Non-Errors |

**SQL Server connection string**
Click ... to enter the SQL Server connection information in the pop-up window.

SQL Server Connection Settings

Enter database information or connection string to create a new database or connect to an existing database.

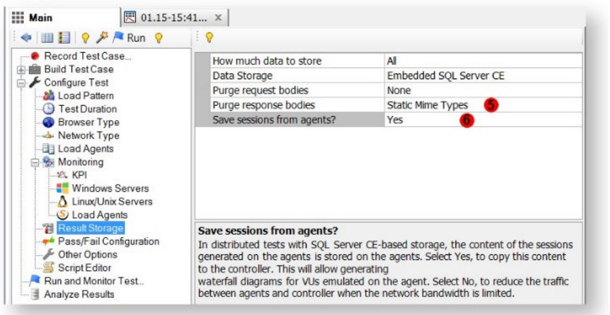
(●) Database Information

Server Name: MySQL_Server

Login
( ) Use Windows Authentication
(●) Use SQL Server Authentication
Username: sa
Password:

Database Name: LoadTesting

( ) Connection String



| | |
|---|---|
| How much data to store | All |
| Data Storage | Embedded SQL Server CE |
| Purge request bodies | None |
| Purge response bodies | Static Mime Types |
| Save sessions from agents? | Yes |

**Save sessions from agents?**
In distributed tests with SQL Server CE-based storage, the content of the sessions generated on the agents is stored on the agents. Select Yes, to copy this content to the controller. This will allow generating
waterfall diagrams for VUs emulated on the agent. Select No, to reduce the traffic between agents and controller when the network bandwidth is limited.

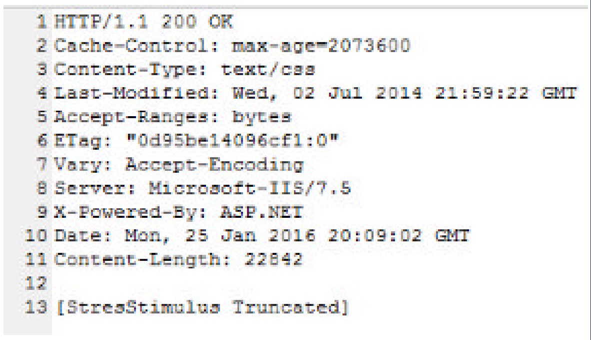
5. If **SQL Server** is selected, enter an **SQL Server Connection String** property or
6. Click the square in the property grid to bring up the connection string dialog
1.
   - Enter SQL Server name, authentication type, credentials and database name.
   - Click **Create/Check DB**. If you use a new database, all tables and other database objects will be created automatically. If use an existing database, the connection will be verified.
   - If you selected **All** during step #2, then most of your data storage will be occupied by the request/response bodies that are stored in the test log. You can reduce the amount of stored data by selecting one of the purge options:

**SQL Server Notes:**

- The user specified in the Login section should be a database owner (dbo) to have sufficient permissions to create tables and other objects as well as insert and update data. If you specified a new database, the user should have permission to create databases.
- The network connection between the StresStimulus controller and SQL Server should be fast enough to transmit real-time insert transactions. VPN or WiFi networks maybe not fast enough. If StresStimulus determines that the network connection slows down the test execution, it will stop the test and display a slow connection error message.

- **Purge request bodies** property (c)
  - **Non-Errors:** Non-Error request bodies are purged and error bodies are not.
  - **All:** All request bodies are purged, no request bodies are stored in the database.
  - **None:** (Default) No request bodies are purged, all request bodies are stored in the database.

- **Purge response bodies** property (d)
  - **Static Mime Types:** (Default) Response bodies of images, video and other static resources are purged. Dynamic responses meaningful for performance analysis will be stored.
  - **None:** No response bodies are purged, all response bodies are stored in the database.
  - **Non-Errors:** Non-Error response bodies are purged and error bodies are not.
  - **All:** All response bodies are purged, no response bodies are stored in the database.

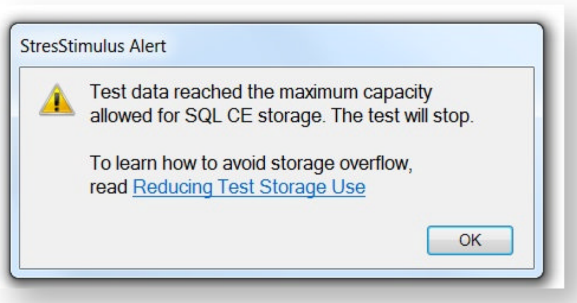
| | |
|---|---|
| **Info:** When you enable purging response bodies, some responses will not be stored in the test log. Session inspector will display the message "StresStimulus truncated" in the place of the response content when you open a session with the purged response. |  |

```
HTTP/1.1 200 OK
Cache-Control: max-age=2073600
Content-Type: text/css
Last-Modified: Wed, 02 Jul 2014 21:59:22 GMT
Accept-Ranges: bytes
ETag: "0d95be14096cf1:0"
Vary: Accept-Encoding
Server: Microsoft-IIS/7.5
X-Powered-By: ASP.NET
Date: Mon, 25 Jan 2016 20:09:02 GMT
Content-Length: 22842

[StresStimulus Truncated]
```

## 5.6.1 Reducing Test Storage Use

Sometimes during long tests with SQL CE based repository, you might get a notification that test data reached the maximum capacity of 4 GB allowed for SQL CE storage. After that, the test will stop, but the test results will be preserved and a final set of reports will be generated.



StresStimulus Alert

Test data reached the maximum capacity allowed for SQL CE storage. The test will stop.

To learn how to avoid storage overflow, read Reducing Test Storage Use

In StresStimulus, you can configure how much, or how little data, you want to store as well as what data storage medium to use. A number of configuration options provided below can help to avoid such situation if you plan to run similar or larger tests.

1. If **Purge response bodies** property is set to **None**, then change to **Static Mime Types**. Response bodies of images, video and other static resources will be purged, but dynamic responses meaningful for performance analysis will be stored.
2. Change **Purge response bodies** property from **Static Mime Types** to **Non-Errors**. Non-Error request bodies will be purged, but error bodies will be saved. It will allow using storage space more efficiently, while keeping the diagnostic information to troubleshoot errors. This will allow you to run much bigger tests with SQL CE based repository.

3. If your test generates a very large number of errors and you would like to run it for long time, consider changing **Purge response bodies** property to **None**.
4. In some rare occasions, application under test uses very large requests. For example, application uploads large files. Consider changing **Purge request bodies** property from **None** to **Non-Errors** or **All**.
5. If you can use SQL Server, as test repository, then change **Data Storage** from **Embedded SQL Server CE** to **SQL Server.** This will allow to use test repository of virtually unlimited size.

## 5.7 Configuring Test Pass/Fail Qualification

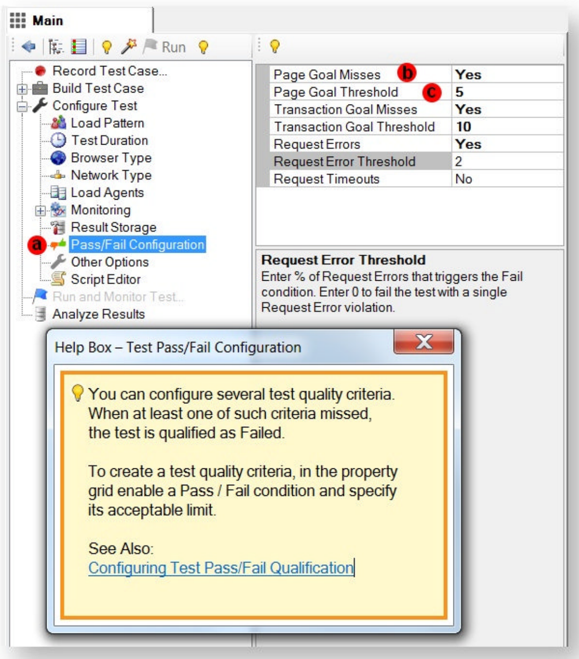
You can define a custom test quality criteria which will determine the outcome of the test. The criteria includes a list of condition thresholds that if exceeded, will fail the test.

There are 4 condition to choose from:

- Page goal misses
- Transaction goal misses
- Request errors
- Request timeouts

In order to enable the test quality criteria:

1. Navigate to the Pass/Fail configuration settings (a).
2. Select and enable the desired conditions to monitor (b).
3. For each criteria, set the percent threshold for violations (c).



| | |
|---|---|
| Page Goal Misses | Yes |
| Page Goal Threshold | 5 |
| Transaction Goal Misses | Yes |
| Transaction Goal Threshold | 10 |
| Request Errors | Yes |
| Request Error Threshold | 2 |
| Request Timeouts | No |

**Request Error Threshold**
Enter % of Request Errors that triggers the Fail condition. Enter 0 to fail the test with a single Request Error violation.

Help Box – Test Pass/Fail Configuration

You can configure several test quality criteria. When at least one of such criteria missed, the test is qualified as Failed.

To create a test quality criteria, in the property grid enable a Pass / Fail condition and specify its acceptable limit.

See Also:
Configuring Test Pass/Fail Qualification

| | |
|---|---|
| Setting any threshold to 0 will trigger a threshold violation if there is at least one instance of the condition. If at the end of the test there are any threshold violations then the test did not pass the test quality criteria and will fail.<br><br>For example, choosing page goal misses and setting the threshold to 10 will trigger a threshold violation if 11% of pages missed their goals.<br><br>A complete list of Pass/Fail Qualification properties is provided in Test Pass/Fail Qualification<br><br>**Note:** If the test was started from a command line or a batch and failed, a status code 1 will be returned. For more about command line interface, check Automation. | |

## 5.8 Other Test Options

### 5.8.1 Host and Port Remapping

By default, a test case recorded against a website will test the same website. The Host remapping feature allows to re-target the host, which was used to record a test case, to a host which should be load tested. If you need to re-use a test case for another website, the host name or IP address and its port can be changed. For example, a test case originally recorded against a production server has to run against a test server.

Port remapping allows to record a test case against the host listening on the port other than 80 (or 443 for HTTPS) and then target a different port during load tests. If the port is not specified, then HTTP will be used port 80 and for HTTPS will be used port 443

To configure host/port remapping, follow these steps

1. In **StresStimulus Main Menu -> Hosts**, select **Remap Hosts**

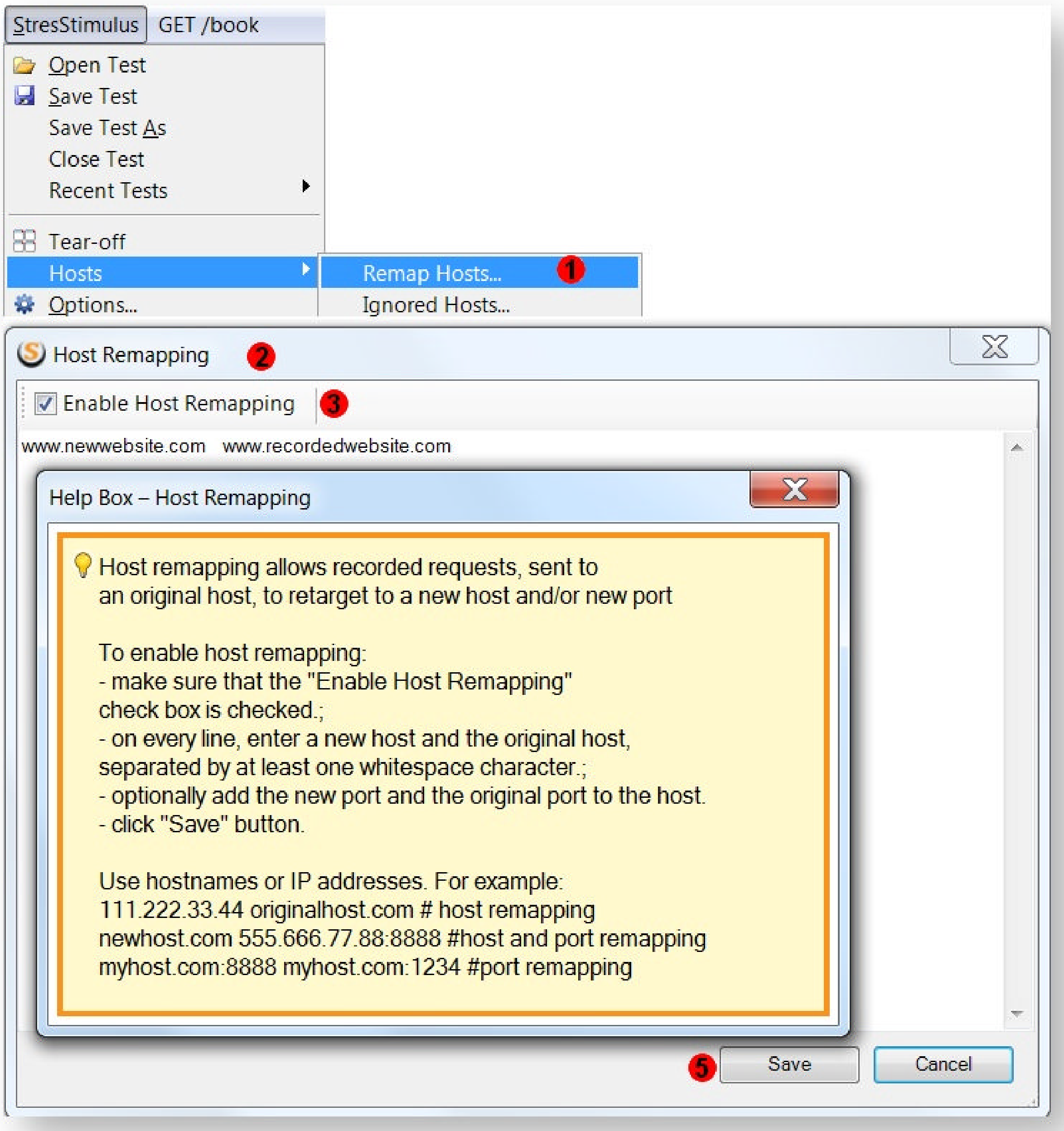
2. The **Host Remapping** dialog will appear

3. Make sure that the **Enable Host Remapping** is checked;

4. On every line, enter a new host and the original host, separated by at least one whitespace character ; optionally add the new port and the original port to the host: {New Host}:{New Port} {Old Host}:{Old Port}

Use host names or IP addresses. For example: 111.222.33.44:8888 testedwebsite.com

5. Click **Save**.



www.newwebsite.com www.recordedwebsite.com

Host remapping allows recorded requests, sent to an original host, to retarget to a new host and/or new port

To enable host remapping:
- make sure that the "Enable Host Remapping" check box is checked.;
- on every line, enter a new host and the original host, separated by at least one whitespace character.;
- optionally add the new port and the original port to the host.
- click "Save" button.

Use hostnames or IP addresses. For example:

```
111.222.33.44 originalhost.com # host remapping
newhost.com 555.666.77.88:8888 #host and port remapping
myhost.com:8888 myhost.com:1234 #port remapping
```

## 5.8.2 Change URI scheme (HTTP/HTTPS)

Sometimes a secure website is hosted in the development environment as a non-secure website. The **Change URI scheme** feature allows to record a test case against a website hosted with the HTTP URI scheme, and then replay it against the same website hosted with the HTTPS URI scheme.

This feature allows to specify hosts for which the recorded scheme will be switched to the alternative scheme: HTTP to HTTPS and vice versa.

To configure **Change URI scheme**, follow these steps.

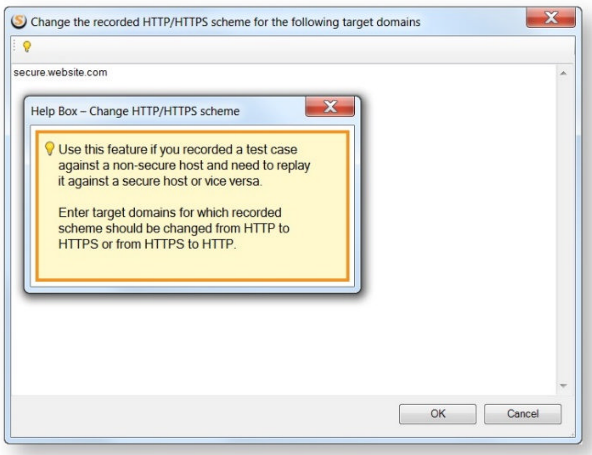
1. In **StresStimulus Main Menu -> Hosts**, select **Change URI scheme.**

2. The **Change URI scheme** dialog will appear.

3. On every line, enter a host for which you want to change the URI scheme.

4. Delete lines with the hosts for which you no longer need to change the URI scheme,

5. Click **Save**


Change the recorded HTTP/HTTPS scheme for the following target domains
Help Box – Change HTTP/HTTPS scheme
OK
Cancel

secure.website.com

Use this feature if you recorded a test case against a non-secure host and need to replay it against a secure host or vice versa.

Enter target domains for which recorded scheme should be changed from HTTP to HTTPS or from HTTPS to HTTP.

**Note:** You can disable the schema change settings for certain test cases. To do so, in the test case property grid, change the **Ignore Schema Change?** property to Yes.

## 5.8.3 Dynatrace Integration

Dynatrace is a Compuware Application Performance Management (APM) suite. It allows to monitor web application performance, pinpoint bottlenecks, isolate application errors and optimize performance on all system tiers.

StresStimulus integration with Dynatrace allows to optimize application performance under load, when performance issues are maximally exposed. StresStimulus allows to include additional data in HTTP requests to correlate client requests with the server-side code analysis. This gives performance engineers more information to optimize application responsiveness.

To enable Dynatrace Integration, in the Configure test -> Other Options section, set the property **Enable Dynatrace integration?** to **Yes.**

StresStimulus integration with Dynatrace is implemented according to the Compuware specification: **Integration with Web Load Testing and Monitoring Tools**

When Dynatrace Integration is enabled, the following pieces of information are included in every request under the “x-Dynatrace” header:

- PC - A Page or transaction name
- ID - A session number in the test case
- NA - The path and query of the request
- AN – An Agent name
- SN - A test case name
- TE - A test name
- VU - A virtual user number
- SI - “StresStimulus”, the issuer of the request

### 5.8.4 Connections Pools

In order to send an HTTP request, a connection must be established to the server. During load test execution, every VU establishes multiple connections to the web server(s) being load tested.

There are several steps required to create a new connection:

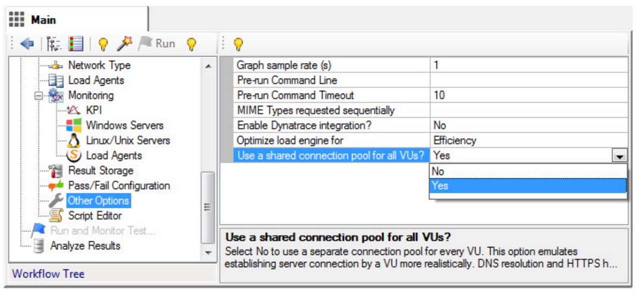
- Establish connection
- DNS resolution
- HTTPS handshaking (if applicable)

Creating a connection consumes some system resources. In some cases the performance impact of creating connections can be significant. If too many connections are created then the performance of the test machine degrades, which may impact accuracy of the test result. In order to save resources on maintaining connections, connection pools are used to reuse connections for subsequent requests.

There are two ways how connection pooling can be handled in StresStimulus:

- **Per VU**: Each VU has its own connection pool with its own DNS cache. On a first request to every host, DNS resolution takes place and then it is cached in the connection pool. This behavior of a VU accurately emulates behavior of a physical user when every user conducts its own DNS resolution. Use this option when You need to accurately estimate DNS resolution time in your performance tests and when system resources are sufficient. While providing accurate DNS resolution emulation, this connection pooling model requires more hardware resources, as the number of connection pools will equal the number of VUs. This pooling model should be used when the number of VUs is not too high or when the load generator machines have sufficient resources. To enable Per VU Pooling, in the workflow tree go to **Other Options** and change the **Use a shared connection pool for all VUs?** to No.

- **Shared**: There is a single connection pool that all VUs share. This model consumes the least amount of system resources and works best in large-scale tests. The only disadvantage of this model is that DNS resolution is performed once per host for all VUs. For practical purposes, DNS resolution time in most cases is insignificant comparison to application server response time and network latency. Therefore this pooling model should be used in most cases as it provides best load test scalability. It is enabled by default.



| | |
|---|---|
| Graph sample rate (s) | 1 |
| Pre-run Command Line | |
| Pre-run Command Timeout | 10 |
| MIME Types requested sequentially | |
| Enable Dynatrace integration? | No |
| Optimize load engine for | Efficiency |
| Use a shared connection pool for all VUs? | Yes |

No
Yes

**Use a shared connection pool for all VUs?**
Select No to use a separate connection pool for every VU. This option emulates establishing server connection by a VU more realistically. DNS resolution and HTTPS h...

# 6 RUNNING AND MONITORING TEST

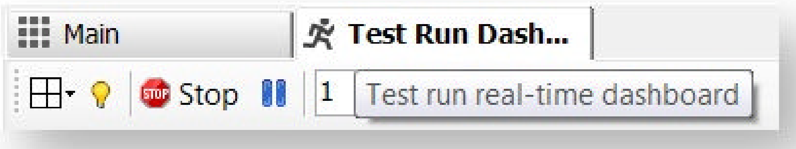
**Search the Running and Monitoring Test section**

StresStimulus executes the currently loaded test. It simulates traffic contained in all test cases (or TC Groups) which have a **Mix Weight** property greater than zero. All configuration settings for the test, test cases, TC groups, and their respective objects are taken into account.

StresStimulus emulates physical users by instantiating VUs. A VU is a software object that is assigned to a specific browser type, network type and a test case (or TCGroup). In distributed tests, a VU is also assigned to a controller or one of the agents. During test execution, a VU continuously iterates through the test case, issues corresponding requests, waits for a corresponding server response, validates them and collects performance metrics.

StresStimulus emulates client requests that look like realistic requests issued by physical browsers, from the server's perspective. In reality, however, StresStimulus does not instantiate browser objects, create web page document object model (DOM), nor run webpage JavaScript.

Once the test is started, a new tab displaying Runtime Dashboard tab will open. It allows to monitor all aspects of test execution and performance metrics.



## 6.1 Starting Test

In order to start the test

- click **Run** on the Workflow Tree toolbar (a).

or

click **Run and Monitor Test** (b) on the Workflow Tree.

Optionally enter a **Test Run Name** (c) and a **Test Run Description** (d) for the current test run. You can also enter or modify the Test Run Name and Test Run Description at any time time in the property grid of the **Configure Test** section (e).

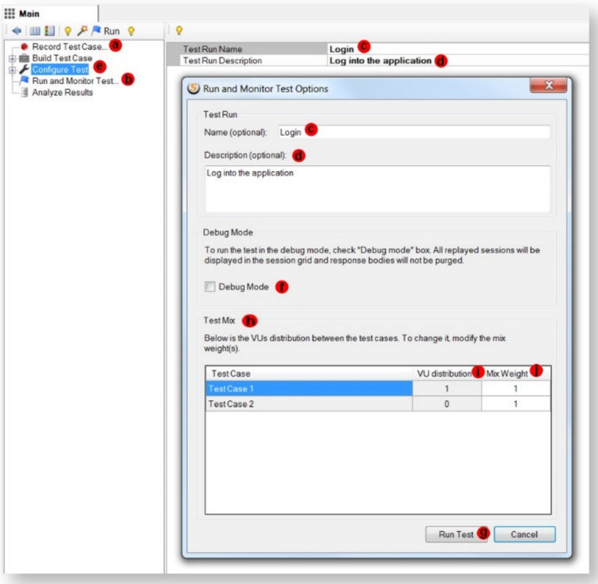
**Info:** Test Run Name and a Test Run Description appear on the test reports and help to identify past test runs.

To run the test in debug mode, check the **Debug mode** box (f) and then click **Run Test** (g).

While running a test in debug mode, all replayed sessions will be displayed in the session grid and response bodies will not be purged for easier diagnostic.

**Note:** Debug mode requires more hardware resources on the test machine and should be used only for test troubleshooting. It is not recommended to run debug tests with many VUs.

The Test Mix group (h) will show a table with all test cases (or test case groups) and the corresponding mix weights (i) and VU



|  |  |
|---|---|
| Test Run Name | Login c |
| Test Run Description | Log into the application d |

Name (optional): Login c

Description (optional): d Log into the application

To run the test in the debug mode, check "Debug mode" box. All replayed sessions will be displayed in the session grid and response bodies will not be purged.

☐ Debug Mode f

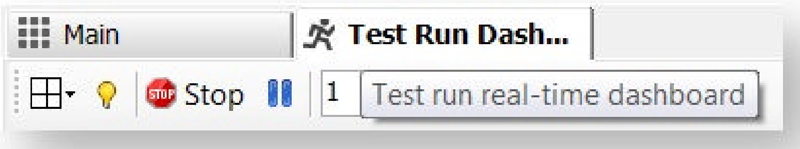
Below is the VUs distribution between the test cases. To change it, modify the mix weight(s).

| Test Case | VU distribution i | Mix Weight j |
|---|---|---|
| Test Case 1 | 1 | 1 |
| Test Case 2 | 0 | 1 |

| | |
|---|---|
| distribution (j). The Mix Weight column may be modified to change the number of VUs that each test case will execute. | |

## 6.2 Runtime Dashboard

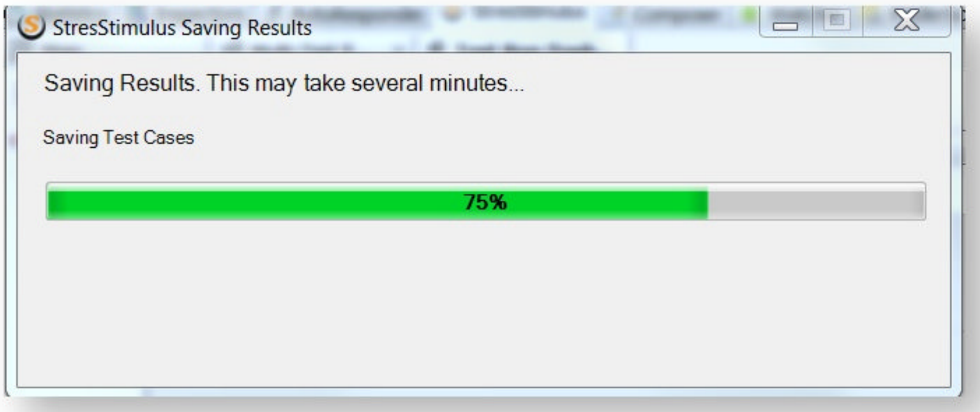
Once the test is started, a new tab displaying the Runtime Dashboard will open. It provides all test execution instrumentation and controls so a user would not need to navigate to any other parts of StresStimulus during the test run.



All aspects of test execution and performance metrics are monitored from the Runtime Dashboard.

**Info:** The full list of Runtime Dashboard elements for monitoring and controlling the test execution is provided in User Interface Reference -> Runtime Dashboard.

After the test is complete, the Saving Results progress bar will appear to indicate that StresStimulus processes collected metrics and generates reports.

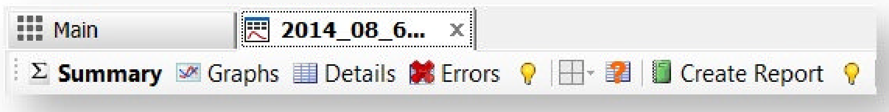

StresStimulus Saving Results

Saving Results. This may take several minutes...

Saving Test Cases

75%

After that, the Runtime Dashboard tab will be converted into the test result tab that will display the final test execution snapshot along with all other performance testing information.



**Note:** By default, the test result name includes a test run time-stamp followed by a test name: `YYYY_MM_DD_HH:MM:SS<test-name>`

## 6.2.1 Layout

Runtime Dashboard includes the following elements:

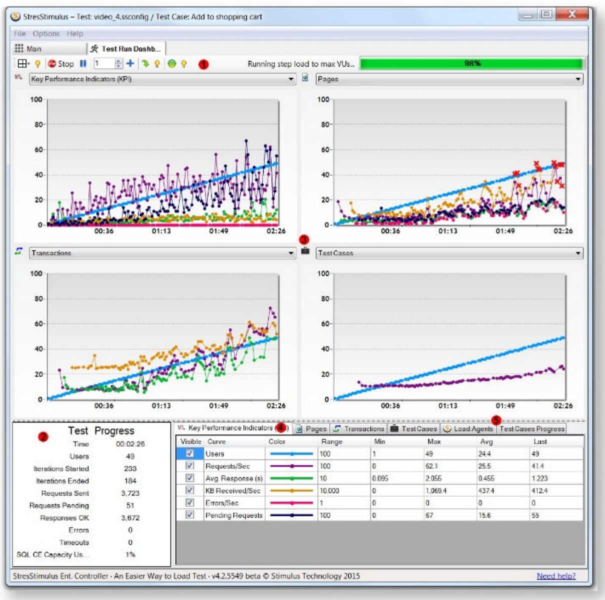
1. **Toolbar** has elements for controlling the test Monitoring Test Progress and Health

2. **Test Progress Panel** displays general parameters

3. **Graph section** can display between one and four configurable graph panels.

4. Each graph has a corresponding tab with a grid that displays additional information about every graph curve.

5. There are 2 optional **Agents Progress** and **Test Cases Progress** Grids


StresStimulus – Test: video_4.ssconfig / Test Case: Add to shopping cart
File Options Help
Main
Test Run Dashb...
Stop
Running step load to max VUs...
98%
Key Performance Indicators (KPI)
Pages
Transactions
Test Cases
Key Performance Indicators
Pages
Transactions
Test Cases
Load Agents
Test Cases Progress
1
2
4
5
StresStimulus Ent. Controller · An Easier Way to Load Test · v4.2.5549 beta © Stimulus Technology 2015
Need help?

Test Progress

| | |
|---|---|
| Time | 00:02:26 |
| Users | 49 |
| Iterations Started | 233 |
| Iterations Ended | 184 |
| Requests Sent | 3,723 |
| Requests Pending | 51 |
| Responses OK | 3,672 |
| Errors | 0 |
| Timeouts | 0 |
| SQL CE Capacity Us... | 1% |

| Visible | Curve | Color | Range | Min | Max | Avg | Last |
|---|---|---|---|---|---|---|---|
| ✓ | Users | 🟦 | 100 | 1 | 49 | 24.4 | 49 |
| ✓ | Requests/Sec | 🟪 | 100 | 0 | 62.1 | 25.5 | 41.4 |
| ✓ | Avg. Response (s) | 🟩 | 10 | 0.095 | 2.055 | 0.455 | 1.223 |
| ✓ | KB Received/Sec | 🟧 | 10,000 | 0 | 1,069.4 | 437.4 | 412.4 |
| ✓ | Errors/Sec | 🟥 | 1 | 0 | 0 | 0 | 0 |
| ✓ | Pending Requests | 🟦 | 100 | 0 | 67 | 15.6 | 55 |

### 6.2.2 Controlling the Test

**1. Controlling the Test.** You can pause and resume the test by clicking on the corresponding buttons.

**2. Skip pending requests.** The Skip button allows the load engine to abandon all pending requests instead of waiting for the arrival of the corresponding server responses. Abandoned requests are counted as timeouts. Skipping pending requests allows the load engine to abandon slow requests and proceed with issuing remaining ones. This feature is helpful when the user wants to complete the test that seems to be "frozen" because some of the (pending) requests are causing slow server response.

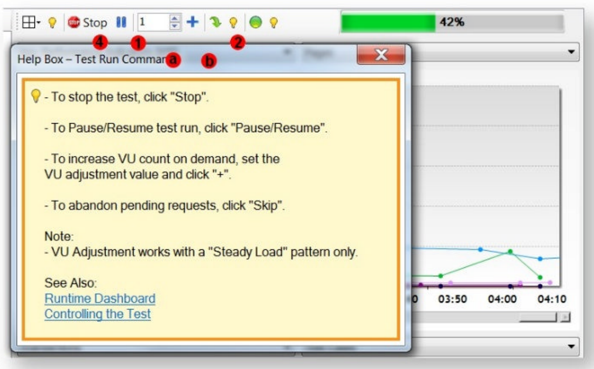
**Note**: Skip does not work on Agents.

**3. Changing VUs during the test run.** You can adjust the VU count on demand. This allows to more granularly control the load. To add VUs during the test:

a. Set the VU adjustment value.

b. Click **+** to increase the VU count.

**Note:** To use VU Adjustment, select **Steady Load** in the **Load Pattern.**



Help Box – Test Run Commar

- To stop the test, click "Stop".

- To Pause/Resume test run, click "Pause/Resume".

- To increase VU count on demand, set the VU adjustment value and click "+".

- To abandon pending requests, click "Skip".

Note:
- VU Adjustment works with a "Steady Load" pattern only.

See Also:
Runtime Dashboard
Controlling the Test

**4. Test Completion.** After the test completion criteria are met, the test will stop automatically. A test can be stopped manually before its completion. It can be necessary in the following situations:

- While test is running, sufficient information about the tested application performance was collected and further test execution is not necessary.
- The test configuration does not reflect your testing goal.

- Too many errors are received and some adjustments are required.

To terminate a running test, click **Stop.**

## 6.2.3 Monitoring Test Progress and Health

Test progress elements are located on the toolbar and in the Test Progress Panel

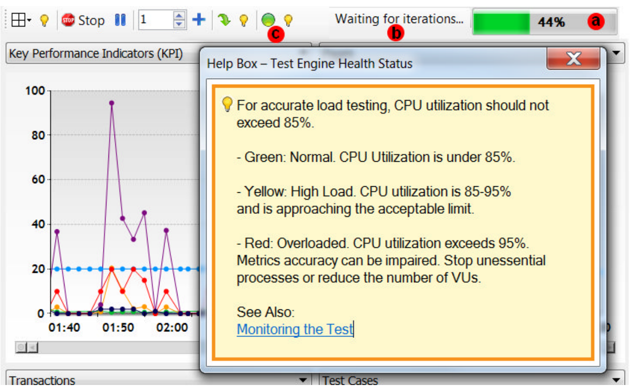
**Toolbar Elements:**

1. The **Progress bar** displays the test's general progress status (a). A load test typically consists of 2 main phases:

- 
  - 
    - Running the test by specified time, number of iterations or reaching maximum users.
    - Waiting for iterations to complete or issued requests to come back.

The progress of the each of the phases is displayed in two separate progress cycles from 0 to 100%. The name of the phase (b) is displayed on the left of the progress bar as follows.

- 
  - On the first phase:
  - Running steady load by time
  - Running steady load by iterations
  - Running step load by time
  - Running step load by iterations
  - Running step load to max VUs
  - On the second phase:
    - Waiting for responses
    - Waiting for Iterations



Help Box – Test Engine Health Status

For accurate load testing, CPU utilization should not exceed 85%.

- Green: Normal. CPU Utilization is under 85%.

- Yellow: High Load. CPU utilization is 85-95% and is approaching the acceptable limit.

- Red: Overloaded. CPU utilization exceeds 95%. Metrics accuracy can be impaired. Stop unessential processes or reduce the number of VUs.

See Also:
Monitoring the Test

**Info:** The progress is also displayed in the program's taskbar icon as a colored background wave.

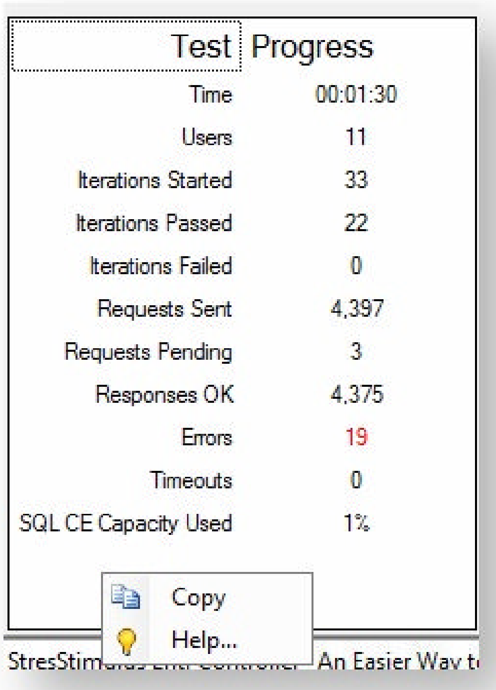
2. The **Health Monitor** (c) monitors the StresStimulus test machine and alerts when it is approaching its capacity limit. This helps to avoid test machine overload in order to sustain high test accuracy. The following color coding is used:

- Green: Normal. CPU Utilization is under 85%.
- Yellow: High Load. CPU utilization is 85-95%. Metric accuracy can be impaired.
- Red: Overloaded. CPU utilization exceeds 95%. Metric accuracy will be likely impaired. Stop unessential processes or reduce the number of VUs.

**Test Progress Panel:**

3. **Test progress panel** display the following parameters:

| Parameter | Description |
|---|---|
| Time | The time elapsed from the beginning of the test |
| Users | The number of instantiated and active VUs, iterating trough their respective test cases. Some VUs can complete the test earlier than others and become inactive. VUs which completed all their iterations before the test ends are excluded from the active user count. If the test is configured to complete only after all VUs complete their iterations, then the VU count may go down before the end of the test. |
| Iterations Started | The number of started test iterations |
| Iterations Passed | The number of Iterations in which all responses were received |
| Iterations Failed | The number of failed (aborted) iterations |
| Requests Sent | The number of issued requests |
| Requests Pending | The number of issued requests, for which responses are not received yet. |
| Responses OK | The number of received responses excluded errors and timeouts |
| Errors | The number of errors |
| Timeouts | The number of timeouts |



Test Progress

| | |
|---|---|
| Time | 00:01:30 |
| Users | 11 |
| Iterations Started | 33 |
| Iterations Passed | 22 |
| Iterations Failed | 0 |
| Requests Sent | 4,397 |
| Requests Pending | 3 |
| Responses OK | 4,375 |
| Errors | 19 |
| Timeouts | 0 |
| SQL CE Capacity Used | 1% |

| | |
|---|---|
| SQL CE Capacity used | (Displayed only if Data Storage is Embedded SQL Server CE)The percentage of the 4 GB storage limit used to store test data accumulated up to this point.<br><br>If using SQL CE and your test is reaching capacity limit, learn how to Reduce Test Storage Use. |

**Tip:** Requests Sent = Requests Pending + Responses OK + Errors + Timeouts

**Note:** The test log is updated in batches with a 60-90 second delay.

**Info**: The full list of parameters in the test metric grids is provided in the User Interface Reference-> Runtime Dashboard.

## 6.2.4 Monitoring Performance

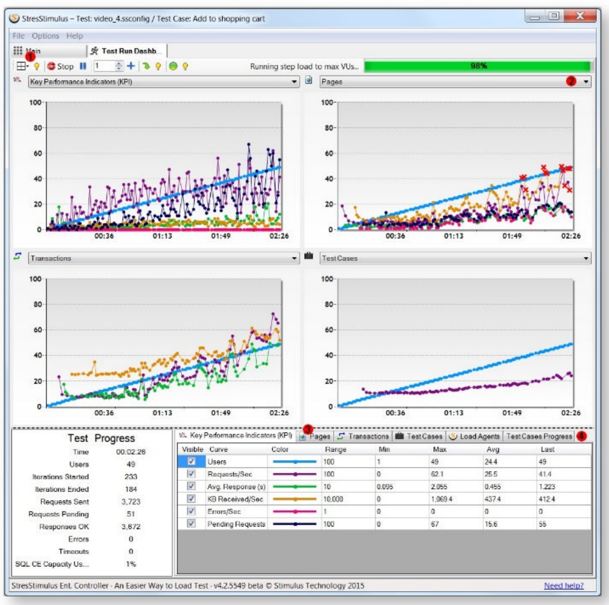
StresStimulus monitors various performance parameters of the tested website in the following categories (for more details, see Graphing Performance Metrics):

- Key performance Indicators
- Pages
- Transactions Ent SP
- Test Cases Ent SP
- Windows, Linux or Unix Servers performance counters Ent SP
- Load Agents performance counters Ent SP

Each category is presented as a graph with real-time performance curves showing dynamics of monitored parameters. Multiple curves displayed on the same graph simplify discovering correlation between parameters. For example, a sharp increase in response time coinciding with reaching a certain number of VUs can help to discover the website's operational ceiling. Another example is if you notice that your server resource utilization is not stressed enough, you can manually increase the number of virtual users to reach the needed threshold for resources.

One to four panel layouts can be selected by clicking the **Layout** drop-down (1). To select which graph to display in a panel, click a drop-down (2) above it. A graph icon of the selected graph type, will be displayed on the left of the drop-down. Here's an example of a graph layouts settings:

- the upper left panel displays KPI,
- the upper right panel displays server performance counters
- the bottom left panel displays pages
- the bottom right panel displays transactions


StresStimulus – Test: video_4.ssconfig / Test Case: Add to shopping cart
File
Options
Help
Main
Test Run Dashb...
1
Stop
Running step load to max VUs...
98%
Key Performance Indicators (KPI)
Pages
2
Transactions
Test Cases
3
4
Key Performance Indicators (KPI)
Pages
Transactions
Test Cases
Load Agents
Test Cases Progress
Need help?

Test Progress

| | |
|---|---|
| Time | 00.02.26 |
| Users | 49 |
| Iterations Started | 233 |
| Iterations Ended | 184 |
| Requests Sent | 3,723 |
| Requests Pending | 51 |
| Responses OK | 3,672 |
| Errors | 0 |
| Timeouts | 0 |
| SQL CE Capacity Us... | 1% |

| Visible | Curve | Color | Range | Min | Max | Avg | Last |
|---|---|---|---|---|---|---|---|
| ✓ | Users | 🟦 | 100 | 1 | 49 | 24.4 | 49 |
| ✓ | Requests/Sec | 🟪 | 100 | 0 | 62.1 | 25.5 | 41.4 |
| ✓ | Avg. Response (s) | 🟩 | 10 | 0.095 | 2.055 | 0.455 | 1.223 |
| ✓ | KB Received/Sec | 🟧 | 10.000 | 0 | 1,069.4 | 437.4 | 412.4 |
| ✓ | Errors/Sec | 🟥 | 1 | 0 | 0 | 0 | 0 |
| ✓ | Pending Requests | ⬛ | 100 | 0 | 67 | 15.6 | 55 |

StresStimulus Ent. Controller · An Easier Way to Load Test · v4.2.5549 beta © Stimulus Technology 2015

| | |
|---|---|

The tab control under the graph panels allows to display several grids, including Curve grids (3), designated with an icon, and optional Agent and Test Case Progress grids (4).

The curve grids display the following additional information about each curve:

| Column | Description | Tabs |
|---|---|---|
| Visible | Whether the curve is currently visible. Click the checkbox to hide/unhide the curve. | All |
| Curve | The name of the curve. | All |
| Color | The curve color and point shape. | All |
| Range | The current value range. The y-value of each point corresponds to a percent of the range. | All |
| Min | Minimum value of the curve - datapoint aggregation (Sortable) | All |
| Max | Maximum value of the curve - datapoint aggregation (Sortable) | All |
| Avg | Average value of the curve - datapoint aggregation (Sortable) | All |
| Last | The last value of the curve - datapoint aggregation (Sortable) | All |
| Warnings | Number of threshold violation warnings (Sortable) Ent SP | KPI, Agents |
| Errors | Number of threshold violation errors (Sortable) Ent SP | KPI, Agents |
| Missed Goals | Number of missed goals (Sortable) Ent SP | Page,Transaction |
| Iterations | The number of times this Transaction or Page was completed successfully Ent SP | Page,Transaction |

Note: Not every grid has all columns.

You can sort the grid by clicking on the desired column header. Click once to sort ascending and again to sort descending.

To see a graph's corresponding grid, select the appropriate tab.

#### 6.2.4.1 Graphing Performance Metrics

The following real-time performance metrics is displayed in each graph

1. **Key performance Indicators (KPI)** monitors a fixed set of performance parameters from the user perspective. They include:

- 
    - **Users -** The number of instantiated and active VUs, iterating trough their respective test cases. Some VUs can complete the test earlier than others and become inactive. VUs which completed all their iterations before the test end are excluded from the active user count. If the test is configured to complete only after all VUs complete their iterations, then the User graph will show gradual declining the number of VUs at the end of the test.
    - **Req/Sec** - The number of requests being sent per second.
    - **Avg. Response(s)** - The average response time of the sent requests. Every datapoint on the response time graph reflects the average value during the checkpoint intervals. If **no responses were received** between two datapoints, the response time cannot be determined, so such datapoints are skipped to reflect an accurate response time curve.
    - **KB Received/Sec** - The number of bytes currently received per second.
    - **Errors/Sec** - The number of errors currently received per second.
    - **Pages/Sec** - The number of pages currently sent per second.
    - **Pending Requests** - The number of currently pending requests. Pending requests are those that are issued but the response is not yet received by StresStimulus. Generally, the greater number of pending requests indicates slower server response. This parameter can be used to gauge performance change in several tests runs.

2. **Pages** monitor an average response time of every page in the test

3. **Transactions** monitor an average response time of every transaction in the test case

4. **Test Cases** monitor an average response time of every transaction in the test case

5. **Windows Servers counters** monitor a set of performance parameters configured in the **Configure Test -> Monitoring -> Windows Servers Monitoring** section. The counters are collected from remote Windows servers involved in the test, including web, application and database servers. Hundreds of different parameters, such as CPU load, memory and disk usage can be monitored. Monitoring server metrics can help to identify potential bottlenecks limiting the application's performance.

6. **Linux/Unix Servers counters** monitor a set of performance parameters configured in the **Configure Test -> Monitoring -> Linux Unix Servers Monitoring** section. The counters are collected from the remote Linux or Unix servers involved in the test using SNMP protocol.

5. **Load Agents counters** monitor a set of performance parameters configured in the **Configure Test -> Monitoring -> Load Agents Monitoring** section. The counters are collected from client testing machines. This use used to make sure that the testing machines are not overloaded.

The Graphs display instant performance characteristics and the performance counter's data, plotted with the frequency defined by the **Graph Sample Rate** property located in the **Configure Test**-> **Other Options** section.

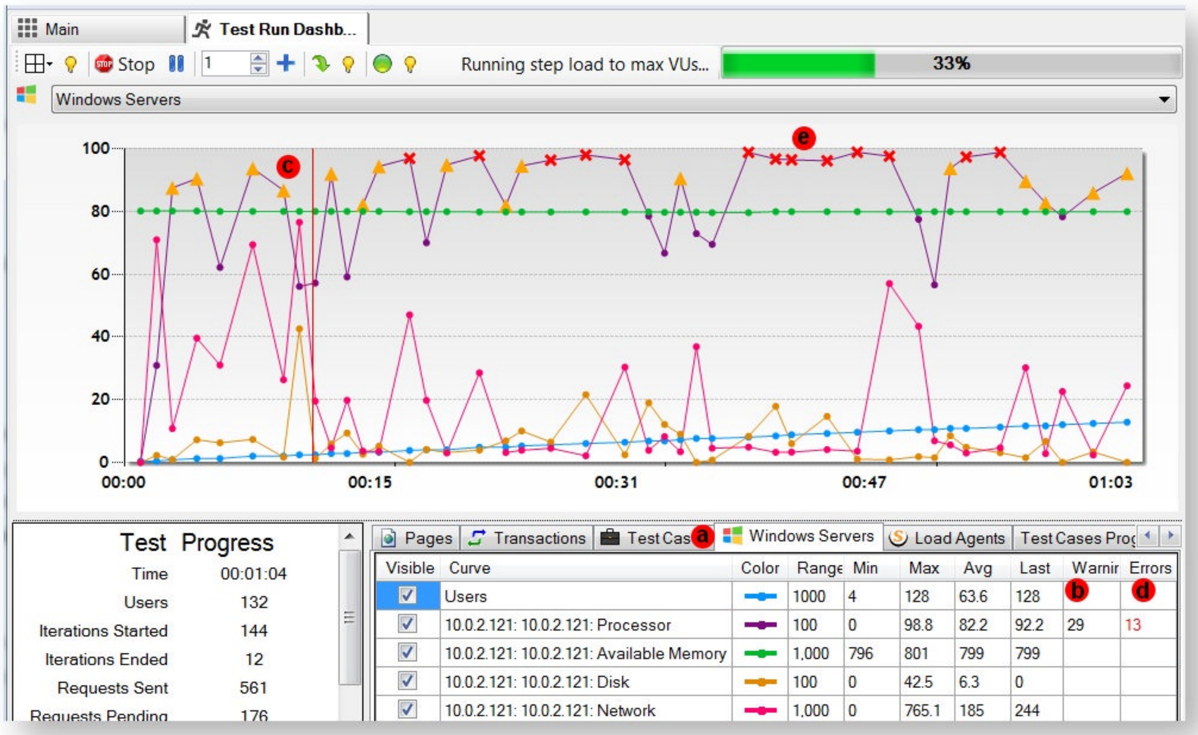
### 6.2.4.2 Graphing Threshold Violations

During a load test execution, you can analyze violations that occur for the configured threshold rules. Results of threshold monitoring are presented in the Runtime Dashboard and test reports on the curve grid below the performance counter graph.

The following thresholds can be monitored:

**Server and Agent performance counters (a).**

- For each performance counter you can configure warning and critical threshold, as described in the Threshold Rules section
- Exceeding the warning threshold will be displayed in the Warning column (b) in orange and the datapoint has a yellow triangle shape (c).
- Exceeding the critical threshold will be counted in the Error column (d) in red and the datapoint has a red cross shape (e).



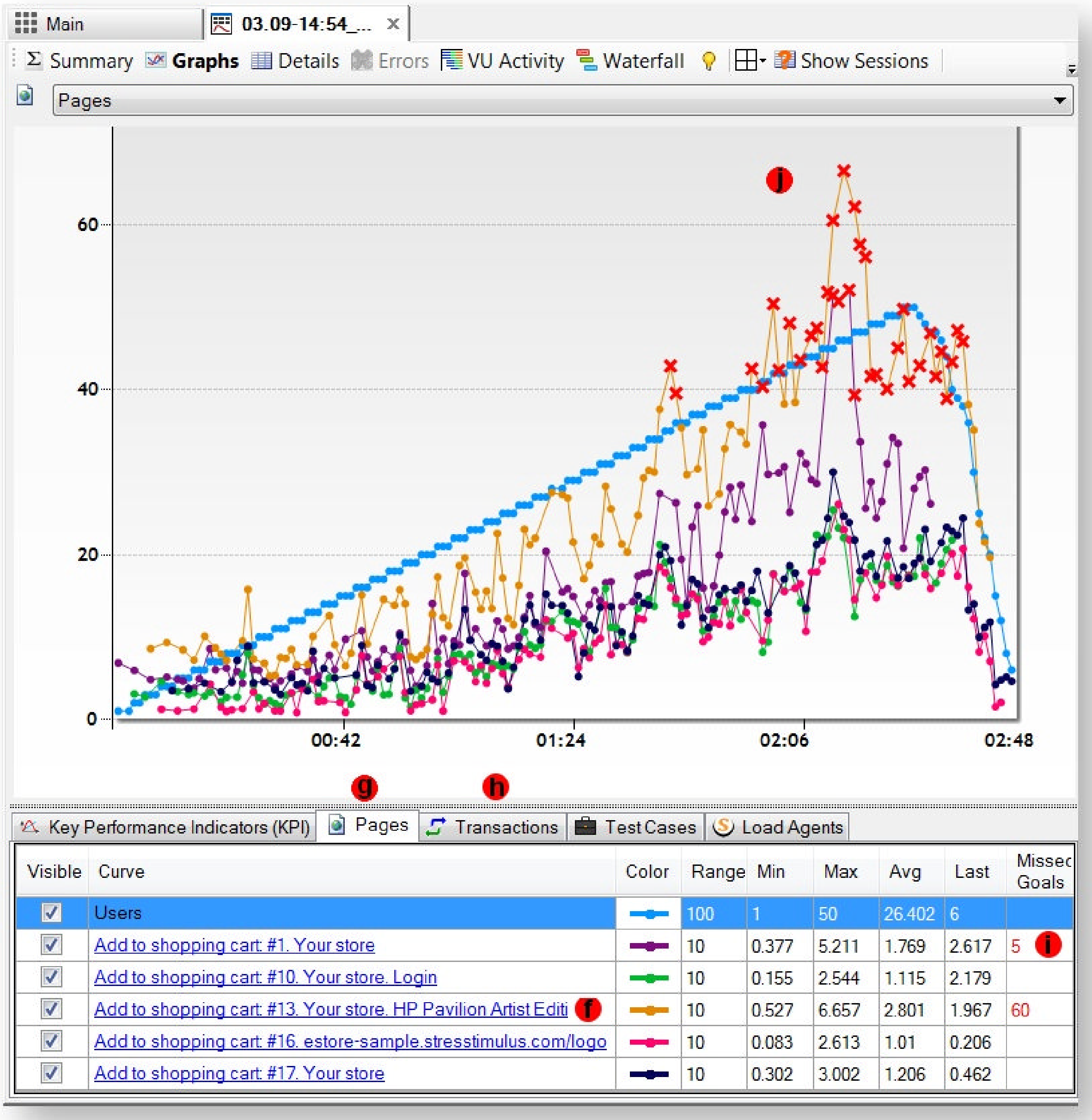
Test Progress

| | |
|---|---|
| Time | 00:01:04 |
| Users | 132 |
| Iterations Started | 144 |
| Iterations Ended | 12 |
| Requests Sent | 561 |
| Requests Pending | 176 |

| Visible | Curve | Color | Range | Min | Max | Avg | Last | Warnir | Errors |
|---|---|---|---|---|---|---|---|---|---|
| ✓ | Users | 🟦 | 1000 | 4 | 128 | 63.6 | 128 | | |
| ✓ | 10.0.2.121: 10.0.2.121: Processor | 🟪 | 100 | 0 | 98.8 | 82.2 | 92.2 | 29 | 13 |
| ✓ | 10.0.2.121: 10.0.2.121: Available Memory | 🟩 | 1,000 | 796 | 801 | 799 | 799 | | |
| ✓ | 10.0.2.121: 10.0.2.121: Disk | 🟧 | 100 | 0 | 42.5 | 6.3 | 0 | | |
| ✓ | 10.0.2.121: 10.0.2.121: Network | 🟥 | 1,000 | 0 | 765.1 | 185 | 244 | | |

**Page and transaction goal misses.**

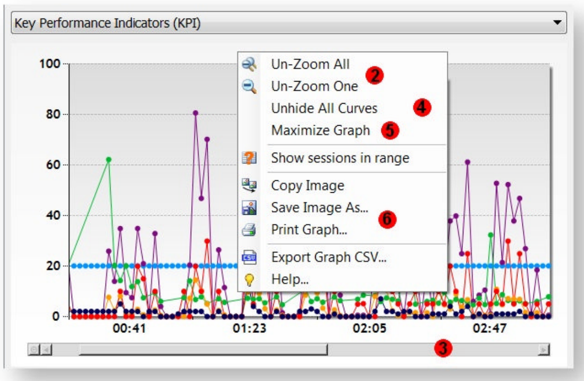
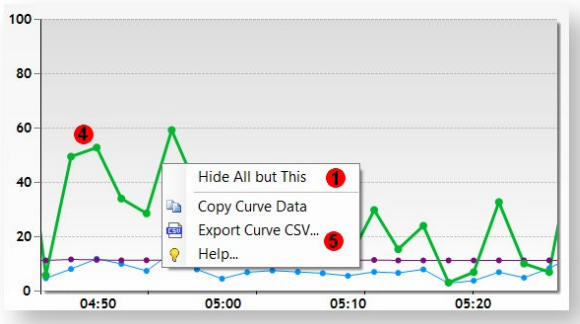
- If you defined a page or transaction goal, it will be monitored as a threshold. Violations are displayed in the appropriate page or transaction curve (f)
- The counter of violations will be displayed in the **Missed Goals** column on the appropriate row of the Page (g) or Transaction (h) grid.
- The datapoints with missed goal has a red cross shape (i)



| Visible | Curve | Color | Range | Min | Max | Avg | Last | Missec Goals |
|---|---|---|---|---|---|---|---|---|
| ✓ | Users | 🟦 | 100 | 1 | 50 | 26.402 | 6 | |
| ✓ | Add to shopping cart #1. Your store | 🟪 | 10 | 0.377 | 5.211 | 1.769 | 2.617 | 5 |
| ✓ | Add to shopping cart #10. Your store. Login | 🟩 | 10 | 0.155 | 2.544 | 1.115 | 2.179 | |
| ✓ | Add to shopping cart #13. Your store. HP Pavilion Artist Editi | 🟧 | 10 | 0.527 | 6.657 | 2.801 | 1.967 | 60 |
| ✓ | Add to shopping cart #16. estore-sample.stresstimulus.com/logo | 🟥 | 10 | 0.083 | 2.613 | 1.01 | 0.206 | |
| ✓ | Add to shopping cart #17. Your store | ⬛ | 10 | 0.302 | 3.002 | 1.206 | 0.462 | |

### 6.2.4.3 Graph Commands

Multiple commands are available to customize graphs and access graph information. Some of them are located in the context menus exposed when right-click on a graph or a curve.

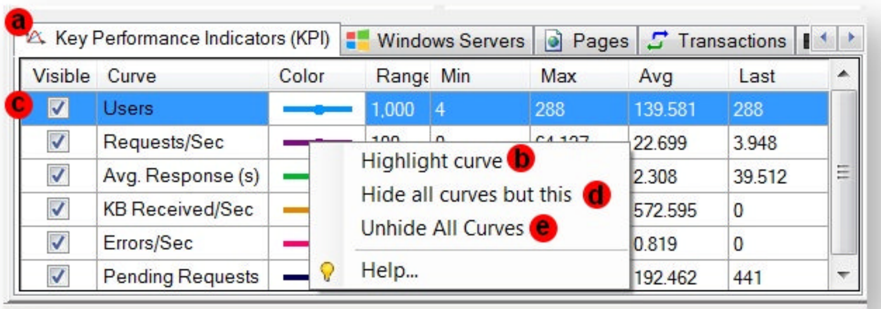
| | |
|---|---|
| Graph commands<br>1. To zoom-in on a graph, select a time range that you wish to zoom.<br>2. To zoom-out one step, click **Un-Zoom One**. To remove zoom, click **Un-Zoom All**.<br>3. To stop time auto-scrolling, scroll to the left.<br>To resume time auto-scrolling, scroll to the right.<br>4. To show hidden curves, click **Unhide**.<br>5. To maximize the graph click **Maximize Graph**<br>6. Other commands:<br>•<br>○<br>▪ Copy, Save, Print Graph Image<br>▪ Export Graph datapoints<br><br>**Note:** Clicking on the graph will select the tab with the corresponding curve grid. |  Key Performance Indicators (KPI)<br>Un-Zoom All<br>Un-Zoom One<br>Unhide All Curves<br>Maximize Graph<br>Show sessions in range<br>Copy Image<br>Save Image As...<br>Print Graph...<br>Export Graph CSV...<br>Help...<br>2<br>4<br>5<br>6<br>3<br>100<br>80<br>60<br>40<br>20<br>0<br>00:41<br>01:23<br>02:05<br>02:47 |
| Curve Commands<br>To access curve commands, mouse over a curve and then right-click<br>1. To hide all but the selected curve, click "Hide".<br>2. To hide / unhide specific curves, check /uncheck corresponding Visible boxes in the curve grid.<br>3. To unhide all curves, in the graph context |  Hide All but This<br>Copy Curve Data<br>Export Curve CSV...<br>Help...<br>1<br>4<br>5<br>100<br>80<br>60<br>40<br>20<br>0<br>04:50<br>05:00<br>05:10<br>05:20 |

| | |
|---|---|
| menu, click **Unhide All**.<br>4. To highlight a curve, click its name in the curve grid or mouse over it in the graph.<br>5. To copy or export core data, click Copy or Export. | |

### 6.2.4.4 Grid Commands

Grid commands:

1. To select a curve grid corresponding to a graph, click the graph or click a matching tab (a).
2. To highlight the curve on the graph, click the curve row or select **Highlight Curve** (b).
3. To show/hide the curve on the graph, check or / uncheck a box on the corresponding row (c).
4. To show just this curve on the graph, select **Hide all Curves but This** (d).
5. To show all curves on the graph, select **Unhide all Curves** (e).



| Visible | Curve | Color | Range | Min | Max | Avg | Last |
|---|---|---|---|---|---|---|---|
| ✓ | Users | | 1,000 | 4 | 288 | 139.581 | 288 |
| ✓ | Requests/Sec | | 100 | 0 | 64.127 | 22.699 | 3.948 |
| ✓ | Avg. Response (s) | | | | | 2.308 | 39.512 |
| ✓ | KB Received/Sec | | | | | 572.595 | 0 |
| ✓ | Errors/Sec | | | | | 0.819 | 0 |
| ✓ | Pending Requests | | | | | 192.462 | 441 |

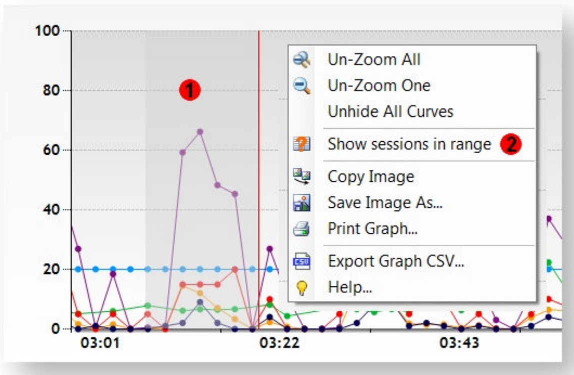
### 6.2.4.5 Retrieving Sessions from Graphs

StresStimulus graphs allow to track down certain performance trends on a test timeline. For example, some sharp spikes in the response time in certain phases of the test become clearly visible. To fully analyze such trends, it is important to understand the underlying web traffic. Querying the test log from the graphs allows to quickly retrieve web sessions captured during such moments for better performance diagnostic. You can also examine test errors as they are received and based on this information decide to continue or to stop the test.

To retrieving sessions from graphs

1. On any graph, select a time range that you wish to analyze. The graph will Zoom-in on the selected time range

2. Right-click on the graph and then click **Show Sessions in Range**

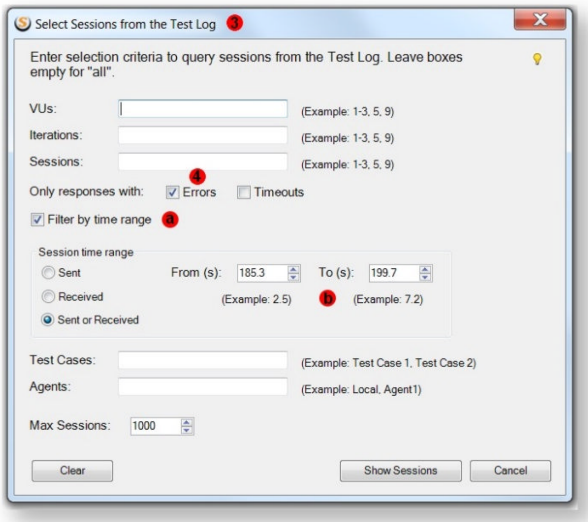


3. A query builder window will appear with the **Filter by time range** box checked (a) and From / To boxes to select the time range (b).

4. Configure any additional search criteria you need. For example, to display only responses with errors, click **Errors**.

The sessions will be retrieved from the test log and displayed in the session grid.

**Note**: The test log is updated with a one minute delay.



Select Sessions from the Test Log 3

Enter selection criteria to query sessions from the Test Log. Leave boxes empty for "all".

VUs: (Example: 1-3, 5, 9)

Iterations: (Example: 1-3, 5, 9)

Sessions: (Example: 1-3, 5, 9)

Only responses with: 4 [✓ Errors] Timeouts

[✓ Filter by time range] a

Session time range

Sent

Received

[✓ Sent or Received]

From (s): 185.3 (Example: 2.5)

b To (s): 199.7 (Example: 7.2)

Test Cases: (Example: Test Case 1, Test Case 2)

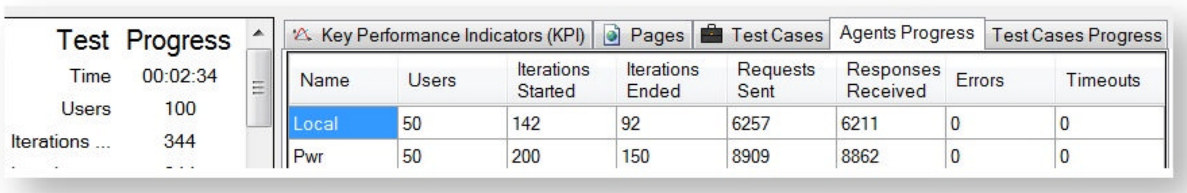
Agents: (Example: Local, Agent1)

Max Sessions: 1000

Clear Show Sessions Cancel

### 6.2.4.6 Agents and Test Cases Progress Grid

The **Agent Progress** tab shows a breakdown of some of these parameters between agents. This tab is only visible when load agents are used in the test.

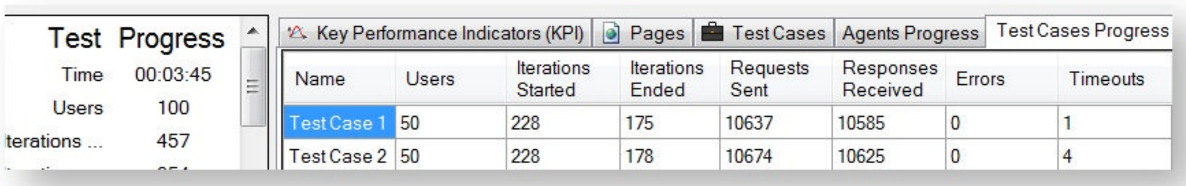


Test Progress

| | |
|---|---|
| Time | 00:02:34 |
| Users | 100 |
| Iterations ... | 344 |

| Name | Users | Iterations Started | Iterations Ended | Requests Sent | Responses Received | Errors | Timeouts |
|---|---|---|---|---|---|---|---|
| Local | 50 | 142 | 92 | 6257 | 6211 | 0 | 0 |
| Pwr | 50 | 200 | 150 | 8909 | 8862 | 0 | 0 |

The **Test Case Progress** (or **TC Groups Progress** ) tab breaks down these parameters between test cases. This tab is only visible when multiple test cases (or TC groups) are used in the test.



Test Progress

| | |
|---|---|
| Time | 00:03:45 |
| Users | 100 |
| terations ... | 457 |

| Name | Users | Iterations Started | Iterations Ended | Requests Sent | Responses Received | Errors | Timeouts |
|---|---|---|---|---|---|---|---|
| Test Case 1 | 50 | 228 | 175 | 10637 | 10585 | 0 | 1 |
| Test Case 2 | 50 | 228 | 178 | 10674 | 10625 | 0 | 4 |

Both grids display the following columns:

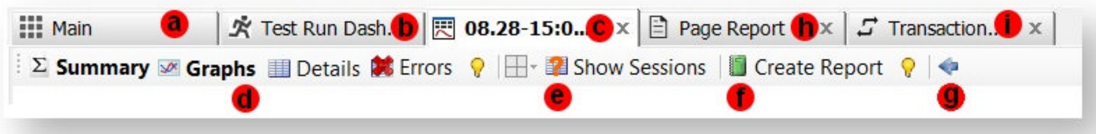
| Column | Description |
|---|---|
| Name | The Test Case or Agent Name |
| Users | The number of active VUs |
| The number of active VUs | Iterations Started |
| The number of started test iterations | Iterations Ended |
| Iterations Ended | The number of completed test iterations |
| Requests Sent | The number of issued requests |
| The number of issued requests | Responses Received |
| The number of received responses | Errors |
| Errors | The number of errors |
| Timeouts | The number of timeouts |

# 7 ANALYZING RESULTS

**Search the Analyzing Results section**

A test is launched from the main tab (a). After test completion, the following steps are performed automatically:

- A new test result is generated and saved in the test repository.
- The Runtime Dashboard tab (b) is closed.
- A new Test Result tab (c) is opened



You can open multiple Test Result tabs from the **Previous Results** section located on the bottom of the Workflow Tree in the Main tab.

**Note**: It is not recommended to heavily use the multiple Test Result tabs during test execution because some of the test machine hardware resources will be taken away from the load generation.

In each **Test Result tab** , the following views (d) are available (i.e. Summary, Details, Graphs, Errors). Each of them displays various aspects of website performance information in rich tabular, graphical and text format. In any of these views, the following additional toolbar commands are available:

- **Show Sessions** (e) to select HTTP sessions from the test log using multiple search criteria, and display them in the session grid for further analysis. For more information, see **Querying Test Log**
- **Create Report** (f) to generate a **Load Test Report** , a document in a portable format (HTML, Excel, etc.) that can be distributed outside StresStimulus.

- **Previous results** (g) to navigate to the **Previous Results** section where other test results can be opened.

Additionally, from a Test Result tab you can open a separate Page Result (h) and a Transaction Result (i) tab for each webpage and transaction presenting specific performance information related to individual pages and transactions.

## 7.1 Opening Previous Results

Each test run creates a separate result set stored in the storage repository. After test completion, its result opens automatically in the Test Result tab. Each previous result can be also opened in a separate Test Result tab.

Because StresStimulus stores all test metrics in the test repository, the result of the test that ended previously will have the same information as test which was just completed.

To load the previous test run results:

1. Select **Analyze Results** in the **Workflow Tree.** The list displaying the result sets of the previous test runs will appear. Recent test runs will be on the top.

2. Click **Refresh Previous Results** on the toolbar to make sure that the displayed set of available previous results is up-to-date

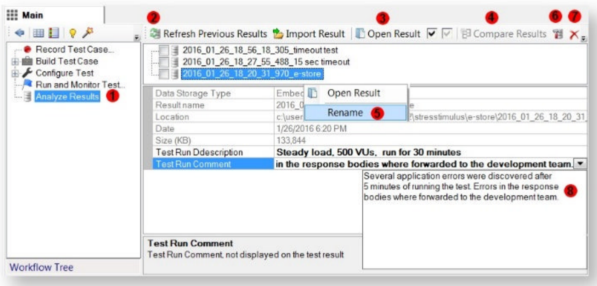
3. To load the selected Result in a new Test Result tab, click **Open Result** or double-click the Result.

**Tip:** To close a Test Result tab, click the x icon on the tab or hit (Ctrl+F4)

4. You can compare multiple previous results side-by-side in one report. Select the results to compare by checking corresponding boxes and clicking **Compare Tests** on the toolbar. The comparison report will appear in the browser

Additional functionality:

5. To rename the selected result, right-click,



2016_01_26_18_56_18_305_timeout test
2016_01_26_18_27_55_488_15 sec timeout
2016_01_26_18_20_31_970_e-store

| | |
|---|---|
| Data Storage Type | Embed |
| Result name | 2016_0 |
| Location | c:\user \stresstimulus\e-store\2016_01_26_18_20_31 |
| Date | 1/26/2016 6:20 PM |
| Size (KB) | 133,844 |
| Test Run Ddescription | Steady load, 500 VUs, run for 30 minutes |
| Test Run Comment | in the response bodies where forwarded to the development team. |

Several application errors were discovered after 5 minutes of running the test. Errors in the response bodies where forwarded to the development team.

**Test Run Comment**
Test Run Comment, not displayed on the test result

select **Rename** and type a new name.

6. To change the data storage settings or the amount of the saved data for subsequent tests, click the **Configure Result Storage settings** on the toolbar.

7. To delete previous results, select them by checking the checkbox on the left and click **Delete** on the toolbar.

8. To quickly add comments to multiple tests after their execution, select a test and enter comments in the **Test Run Comment** property. The comments are not included in the test results. You can also edit **Test Run Description** property which is included in the test Summary.

**Info:** The full list of page properties, toolbar commands and context menu options is provided in the User interface reference -> Previous Results.

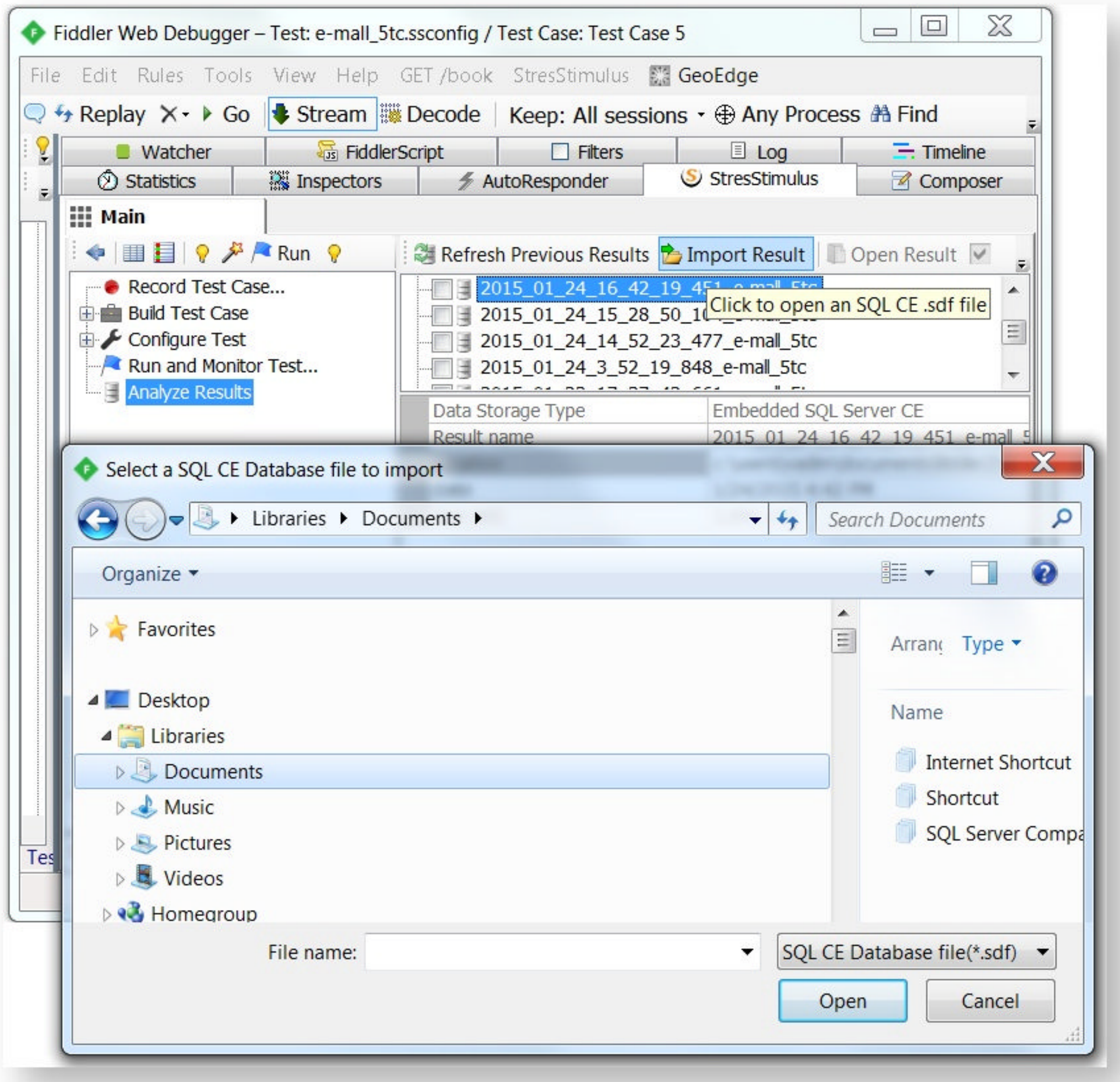
**Tip for advanced users:** By default test result is stored in a SQL Server CE .sdf file. You can access it directly using Microsoft® SQL Server® 2008 Management Studio Express.

## 7.2 Opening an SQL CE file

In some situations it is necessary to open test result without opening the entire test. For example:

- You have a test folder containing multiple test runs, but you need to share one test run result with a team member.
- You would like to open local test result file generated on an agent.

To open a standalone test result, from the **Analyze Results** section, click **Import Result** on the toolbar


Fiddler Web Debugger – Test: e-mall_5tc.ssconfig / Test Case: Test Case 5
File Edit Rules Tools View Help GET /book StresStimulus GeoEdge
Replay Go Stream Decode Keep: All sessions Any Process Find
Watcher FiddlerScript Filters Log Timeline
Statistics Inspectors AutoResponder StresStimulus Composer
Main
Run
Record Test Case...
Build Test Case
Configure Test
Run and Monitor Test...
Analyze Results
Refresh Previous Results
Import Result
Open Result
2015_01_24_16_42_19_451_e-mall_5tc
Click to open an SQL CE .sdf file
2015_01_24_15_28_50_10
2015_01_24_14_52_23_477_e-mall_5tc
2015_01_24_3_52_19_848_e-mall_5tc
Data Storage Type
Embedded SQL Server CE
Result name
2015_01_24_16_42_19_451_e-mall_5
Select a SQL CE Database file to import
Libraries
Documents
Search Documents
Organize
Favorites
Desktop
Libraries
Documents
Music
Pictures
Videos
Homegroup
Arrang Type
Name
Internet Shortcut
Shortcut
SQL Server Compa
File name:
SQL CE Database file(*.sdf)
Open
Cancel

Open file dialog will appear. Navigate to a necessary folder and open an .sdf file.

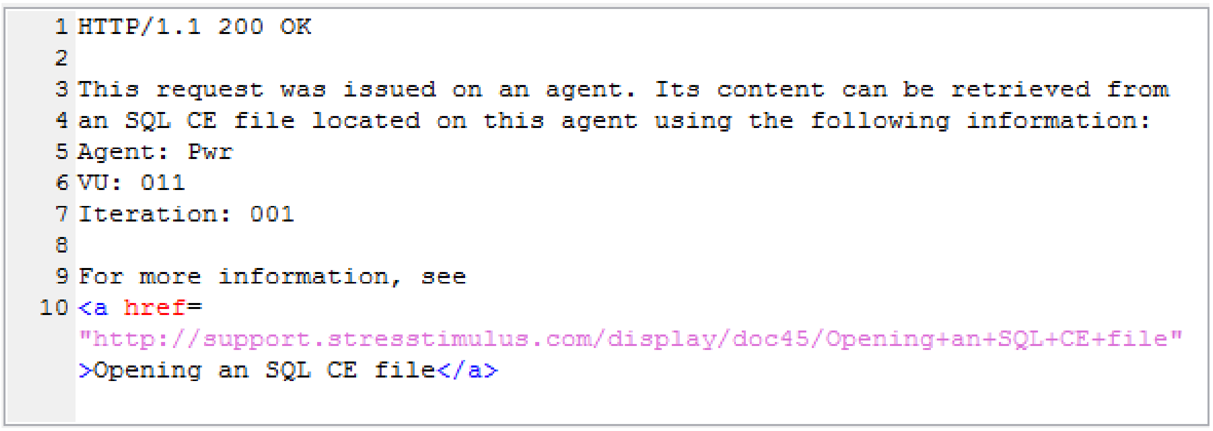
After that, a standalone test result will appear in the analyze results section. Double-click it to open the test report.

### 7.2.1 Opening Agent SQL CE file

If running a distributed test with one or more agents, while using SQL CE as data storage, then each agent and the controller will have its own SQL CE file.

- The agent's SQL CE file will contain the replayed sessions that the agent initiated.
- The controller's SQL CE file will contain all the final results and calculations, along with replayed sessions that the controller initiated.

It is sometimes necessary to query the replayed sessions from an agent. If trying to do this from the controller, you will get the following response in the inspector window.



```
HTTP/1.1 200 OK

This request was issued on an agent. Its content can be retrieved from
an SQL CE file located on this agent using the following information:
Agent: Pwr
VU: 011
Iteration: 001

For more information, see
<a href=
"http://support.stresstimulus.com/display/doc45/Opening+an+SQL+CE+file"
>Opening an SQL CE file</a>
```

To see the request/response info from a replayed session on the agent, go to the agent machine, open StresStimulus, go to **Analyze Results** as described here. The .sdf file you are looking for will be in the folder **%My Documents%\Fiddler2\StresStimulus\AgentTemp\{Test Id}\{Test File}**. The {Test Id} can be found in the {Test File}.ssconfig file in the TestModel/TestConfiguration TestId node. There will be one .sdf file for every test run with the following format **{Test Date}_{Test Run Name}.sdf**, so pick the one that corresponds to your test.

**Note:** If Test Run Name was empty then {Test File} is used.

## 7.3 Test Result Tab

A new Test Result tab is automatically created after the test run is complete. It can be also opened from the **Previous Results** section. Several results can be opened at the same time.

Test Result tab provide reach functionality to analyze various aspects of application performance exposed by the test and drill down layers of the stack impacting performance. The following views of the test result that can be selected on the toolbar: Summary View, Graph View, Detail View, Error View, VU Activity View and Waterfall View.

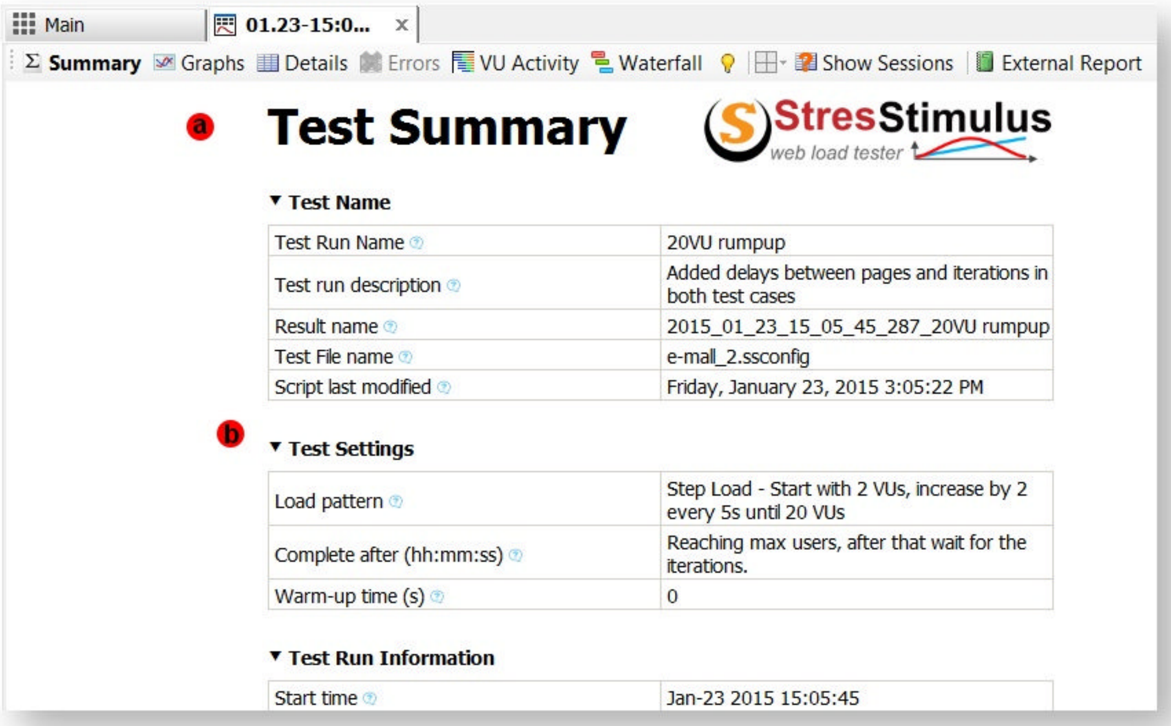
**Tip:** To switch between two last used tabs, hit **Ctrl+Tab**. This keyboard shortcut allows to alternate between the two most recent tabs without using the mouse. Use it when you need to compare results of two test runs visually.

### 7.3.1 Summary View

The Summary View displays key performance characteristics aggregated for the test duration. It gives a bird's eye view on your load test.

To select Summary View, click Summary (a) on the toolbar.

The test summary includes several subsections that can be expanded / collapsed by clicking the triangle icon (b).



Test Summary

▼ Test Name

| | |
|---|---|
| Test Run Name | 20VU rumpup |
| Test run description | Added delays between pages and iterations in both test cases |
| Result name | 2015_01_23_15_05_45_287_20VU rumpup |
| Test File name | e-mall_2.ssconfig |
| Script last modified | Friday, January 23, 2015 3:05:22 PM |

▼ Test Settings

| | |
|---|---|
| Load pattern | Step Load - Start with 2 VUs, increase by 2 every 5s until 20 VUs |
| Complete after (hh:mm:ss) | Reaching max users, after that wait for the iterations. |
| Warm-up time (s) | 0 |

▼ Test Run Information

| | |
|---|---|
| Start time | Jan-23 2015 15:05:45 |

A list of the Summary View subsections is provided below:

#### 7.3.1.1 Test Name

| | |
|---|---|
| Test Run Name | Optional one-word Test Run Name specified in the Configure Test section. It is used as a suffix following a time-stamp of the test run displayed in the "Analyze Results" section. |
| Test run description | Test description specified in the Configure Test section prior to the test run |
| Result name | The name automatically created for every test run. In SQL CE, a name of a .sdf file. |
| Test File name | The name of the test script file |
| Script last modified | The timestamp of the last script modification |

#### 7.3.1.2 Test Settings

| | |
|---|---|
| Load pattern | Steady load pattern with number of VUs or step load pattern |
| Complete after (hh:mm:ss) | The test primary and secondary completion criteria |
| Warm-up time (s) | The warm-up time actually used in the test |

#### 7.3.1.3 Test Run Information

| | |
|---|---|
| Start time | Test start time |
| End time | Test end time |
| Test run duration | The test duration in seconds. [End Time] - [Start Time] |

#### 7.3.1.4 Overall Result

| | |
|---|---|
| Completion Status | Shows whether the test was Completed or Aborted |
| Pass/Fail Status | Shows whether the test has Passed or Failed |
| Fail Conditions | Shows the Failed Conditions that were violated. The following quality parameters can be missed:<br>• Page Goal Misses exceeded the fail limit |

| | |
|---|---|
| | • Transaction Goal Misses exceeded the fail limit<br>• Request Errors exceeded the fail limit<br>• Request Timeouts exceeded the fail limit |
| Max User Load | The maximum number of VUs reached during the test. When Constant Load Pattern is selected, it will be equal to the set number of VUs. When Step Load Pattern is selected, it will be equal to the Max number of VUs or a smaller number that the test duration limit permits to reach. |
| Total sent (KB) | The total amounts of test upload traffic that is equal to the sum of all requests' sizes |
| Total received (KB) | The total amount of test download traffic that is equal to the sum of all responses' sizes |
| KB sent/sec | KB sent per second. Calculated as [Total KB sent] / [Test run duration] |
| KB received/sec | KB received per second. Calculated as [Total KB received] / [Test run duration] |

#### 7.3.1.5 Test Iterations

| | |
|---|---|
| Avg. Iteration time (s) | The average time it took to complete an iteration. Only displayed if all iterations were completed. |
| Iterations started | The combined number of test iterations started by all VUs |
| Iterations passed | The combined number of test iterations completed by all VUs |
| Iterations failed | The combined number of failed (aborted) test iterations |
| Incomplete Iterations | The difference between Iterations started and Iterations passed or failed |

#### 7.3.1.6 Requests

| | |
|---|---|
| Avg. response time (s) | Calculated as [Aggregate Session time] / [Total Requests]. |
| Requests/sec | The number of HTTP transactions (sessions) issued by the load test engine. It is calculated as [Total Requests] / [Test run duration]. |
| Number of URLs | The total number of recorded URLs |

| | |
|---|---|
| Total requests issued | The combined number of requests issued during the test |
| Request errors | The total number of errors |
| Request timeouts | The total number of timeouts |

#### 7.3.1.7 Transactions

| | |
|---|---|
| Avg. response time (s) | Average transaction response time. |
| Transactions/sec | Average transactions requested per second |
| Number of transactions | The number of transactions in the test |
| Requested Transactions | The total number of Transactions requested in all iterations |
| Transactions with error(s) | The number of transactions with error(s) |
| Requested Transactions with error(s) | The total number of Transactions requested in all iterations with error(s) |
| Transactions with timeout(s) | The number of transactions with with at least one timeout request. See Page Timeout for more details. |
| Requested Transactions with timeout(s) | The total number of Transactions requested in all iterations with timeout(s) |
| Transactions with missed goal(s) | The number of transactions with missed goal |
| Requested Transactions with missed goal(s) | The total number of Transactions requested in all iterations with missed goal(s) |
| Slowest Transaction(s) | Up to five slowest transactions. Click a transaction link to open a transaction report |

#### 7.3.1.8 Pages

| | |
|---|---|
| Avg. response time (s) | Average page response time. |
| Pages/sec | Average pages requested per second |
| Number of pages | The number of pages in the test |
| Requested Pages | The total number of pages requested in all iterations |

| | |
|---|---|
| Pages with error(s) | The number of pages with error(s) |
| Requested Pages with error(s) | The total number of pages requested in all iterations with error(s) |
| Pages with timeout(s) | The number of pages with with at least one timeout request. See Page Timeout for more details. |
| Requested Pages with timeout(s) | The total number of pages requested in all iterations with timeout(s) |
| Pages with missed goal(s) | The number of pages with missed goal |
| Requested Pages with missed goal(s) | The total number of pages requested in all iterations with missed goal(s) |
| Slowest Page(s) | Up to five slowest pages. Click a page link to open a page report |

Errors

| | |
|---|---|
| Top Errors | Up to five most occurring errors. Click an error link to retrieve and display all occurrences of this error in the session grid on the left. |

### 7.3.2 Graph View

To select Graph View, click **Graphs** on the toolbar.

Graph view includes in-depth graphs with load testing metrics demonstrating the performance of a tested website. It is similar to Runtime Dashboard. It however misses several elements such as **Test Progress Panel**, Agents and Test Cases Grid and UI elements for controlling the test on the toolbar

- The layout of the Graph view is described in the Layout section.
- Performance graphs are described in the Graphing Performance Metrics section.
- Graph commands are described in the Graph Commands section.

**Note:** Customization changes made in the Graph View will be reflected in the Load Test report.

- Curve grid data shown below the graphs is described in the Monitoring Performance section. The final snapshot of the curves' data taken at the test completion is displayed.

**Note:** There will be some discrepancies between the values on the curve grids and the corresponding values in the page and transaction detailed view. The Min, Max and Avg values on the curve grids are aggregations of the data points collected during the test run. The data points are the average values during the checkpoint intervals. Such metrics are primarily collected for monitoring test progress and presenting performance in real time without waiting for the final results. In contrast, the Min, Max and Avg values on the detail views are an aggregation of the actual measurements of every occurrence of the parameter. The detail data is calculated after the test is complete and cannot be accessed during the test run, but presents a more complete performance picture.

You can fetch load test sessions from the test log as described in Retrieving Sessions from Graphs section.

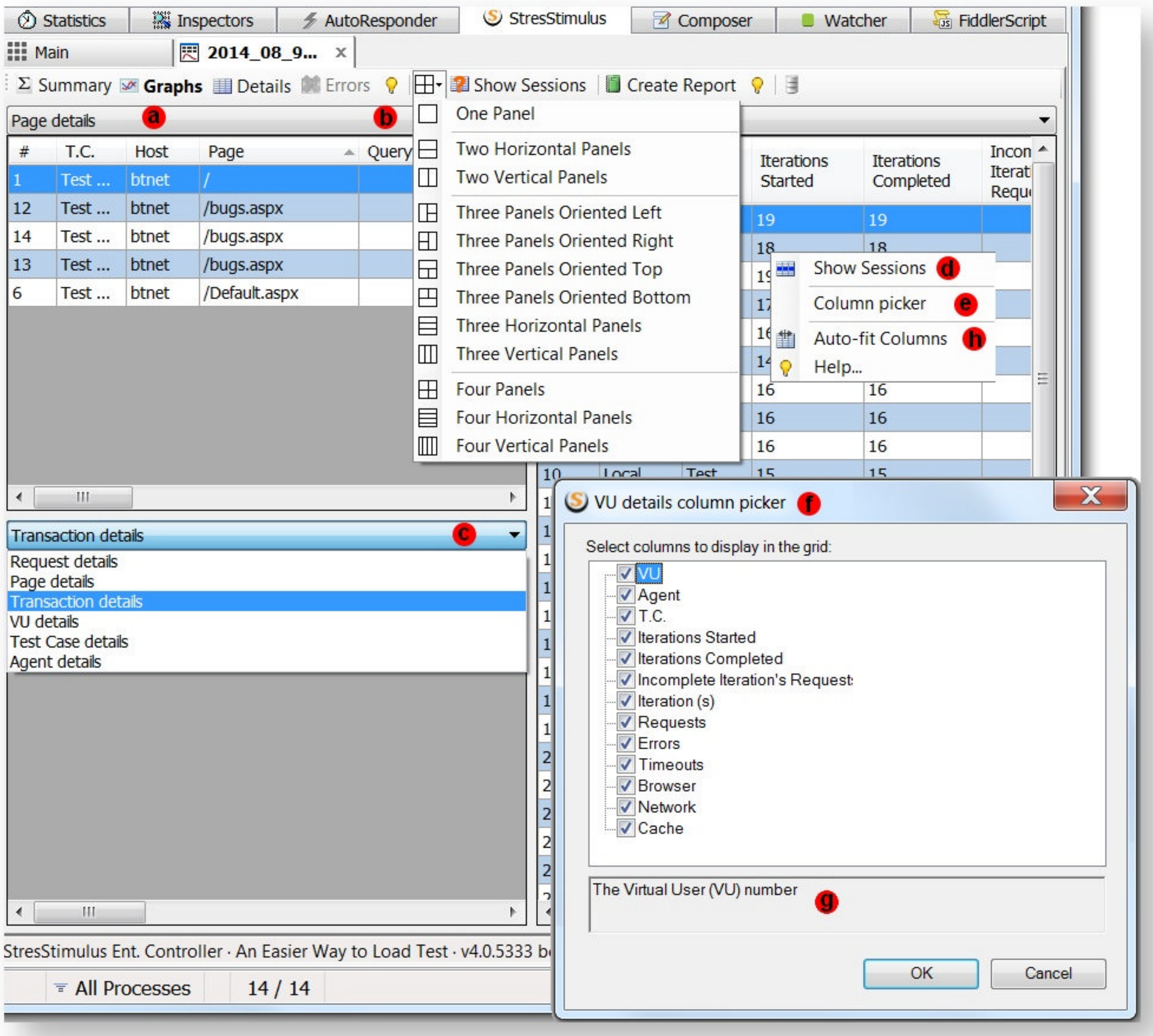
## 7.3.3 Detail View

### 7.3.3.1 Layout and functionality

Detail View displays the following types of tabular performance data:

- Page details
- Transaction details
- Request details
- VU details
- Test Case details
- Agent details

To select Detail View, click **Details** on the toolbar (a). One to four panel layouts can be selected by clicking the **Layout** drop-down (b). Each panel displays a grid with a specific type of performance data. To select which grid to display in a panel, click the drop-down (c) above it.



Tip. When you select the order in the test result detailed view, the system will remember it for next time. Report configuration persistence helps to preserve users' favorite settings.

To sort a grid column double-click it.

You can copy selected grid rows and paste the content into Excel. To select multiple rows, use (CTRL + click) / (SHIFT + click) / (CTRL + A)

Several commands are available in the context menus. To access them, right-click on a grid:

- To show sessions associated with selected grid rows, Click **Show Sessions** (d). The sessions will be retrieved from the test log and displayed in the session grid. The number of displayed sessions is limited by a number entered in the **Maximum Sessions** box on the dialog displayed when **Show Sessions** on the toolbar is clicked. Depending on the grid you selected, sessions associated with a page, transaction, request, test case or agent can be retrieved.
- To customize grid columns; click **Column Picker** (e). Column Picker dialog (f) will appear. Uncheck the columns you wish to hide. Each column description is provided in the area (g) below. Click OK.

**Note:** A customization changes made in the Detail View will be reflected in the Load Test report.

- To Auto-fit column with click Auto-fit columns.

#### 7.3.3.2 Page Details

The **Page Details** panel displays performance characteristics of individual pages from the end-user's perspective, calculated by aggregating its requests.

**Note**: Page response time includes time for loading all requests. Requests loaded after the page is displayed (e.g. AJAX requests), as determined by StresStimulus, are excluded from the page.

**Page Details Grid Columns**

| **Column** | **Description** |
|---|---|
| # | The page's request number |
| T.C. | The Test Case name |
| Host | The host |
| Path | The page path |
| Query | The query string |
| Title | The page title |
| Requests | The number of requests |
| Successful | The number of times this page was completed successfully |

| Iterations | |
|---|---|
| Req. Issued | The number of requests issued during all iterations |
| Avg. (s) | The average response time |
| Min. (s) | The minimum response time |
| Median (s) | The median response time |
| 90% (s) | The maximum response time after excluding the slowest 10% of page iterations |
| 95% (s) | The maximum response time after excluding the slowest 5% of page iterations |
| 99% (s) | The maximum response time after excluding the slowest 1% of page iterations |
| Max. (s) | The maximum response time |
| STD | The response time standard deviation |
| Goal (s) | The response time goal |
| Missed Goals | The number of iterations where the response time exceeded the goal |
| Errors | The number of iterations with errors |
| Timeouts | The number of iterations with timeouts |
| Missed Goal % | % of page iterations where the response time exceeded the goal |
| Errors % | % of page iterations with at least one error |
| Timeouts % | % of page iterations where a timeout was registered for at least one request |
| HTTP Errors | The total number of errors registered during all iterations |
| Timeouts | The total number of timeouts registered during all iterations |

#### 7.3.3.3 Transaction Details

A transaction is a custom set of sequential requests or pages representing a complete business transaction. It is used to determine performance characteristics of a specific business transaction that includes several user actions.

The **Transaction Details** panel displays performance characteristics of individual transactions (below) from the end-user perspective, calculated by aggregating its requests.

### Transaction Details Grid Columns

| Column | Description |
|---|---|
| Transaction | The Transaction name |
| Description | The Transaction description |
| # | The transaction's first request number |
| T.C. | The Test Case name |
| Requests | The number of requests in the Transaction |
| Successful Iterations | The number of times this Transaction was completed successfully |
| Req. Issued | The number of requests issued during all successful and failed iterations |
| Avg. (s) | The average response time in successful iterations. Transaction response time is an interval between its first session's request and the latest response received. |
| Min. (s) | The minimum response time |
| Median (s) | The median response time |
| 90% (s) | The maximum response time after excluding the slowest 10% of transaction iterations |
| 95% (s) | The maximum response time after excluding the slowest 5% of transaction iterations |
| 99% (s) | The maximum response time after excluding the slowest 1% of transaction iterations |
| Max. (s) | The maximum response time |
| STD | The response time standard deviation |
| Goal (s) | The response time goal |
| Missed Goals | The number of iterations where the response time exceeded the goal |
| Errors | The number of iterations with errors |
| Timeouts | The number of iterations with timeouts |
| Missed Goal % | % of page iterations where the response time exceeded the goal |

| | |
|---|---|
| Errors % | % of transaction iterations with at least one error |
| Timeouts % | % of transaction iterations where a timeout was registered for at least one request |
| HTTP Errors | The total number of errors registered during all iterations |
| Timeouts | The total number of timeouts registered during all iterations |

#### 7.3.3.4 Request Details

The **Request Details** section displays aggregated performance characteristics of individual requests grouped by URL. Time characteristics are averaged. Request counts are summed.

**Note** : If a request timed-out and subsequently failed, it's counted as a timeout.

#### Request Details Grid Columns

| **Column** | **Description** |
|---|---|
| # | The request number |
| T.C. | The Test Case name |
| Host | The host |
| Path | The page path |
| Query | The query string |
| Type | The request's Content-Type. |
| Response (s) | The average time to receive the response (TTLB) |
| TTFB (s) | The average time to receive the first byte of the response |
| Network (s) | The average time between receiving the first and the last byte of the response necessary for transferring it over the network |
| Request (KB) | The average Request Body size |
| Response (KB) | The average Response Body size |

| | |
|---|---|
| Requests | The number of times the request was issued during the test run |
| Errors | The number of the request's failures during the test run |
| Timeouts | The number of the request's timeouts registered during the test run, |
| Timeout (s) | The timeout settings for this request |
| STD | Response Time Standard Deviation |

### 7.3.3.5 VU Details

The **VU Details** section displays statistics of the test Iterations executed by every VU.

#### VU Details Grid Columns

| **Column** | **Description** |
|---|---|
| VU | The Virtual User (VU) number |
| Agent | The Agent name |
| T.C. | The Test Case name |
| Iterations Started | The number of started Test Iterations |
| Iterations Passed | The number of Iterations in which all responses were received |
| Iterations Failed | The number of failed (aborted) iterations |
| Incomplete Iteration's Requests | The number of requests sent in the last incomplete Iteration |
| Iteration Time (s) | Average time (s) of a passed iteration |
| Requests | The total number of requests sent in all Iterations |
| Errors | The number of response errors |
| Timeouts | The number of response timeouts |
| Browser | The browser type used by this VU |
| Network | The network type used by this VU |
| Cache | Whether this VU is a new or a returned user |

#### 7.3.3.6 Test Case and TC Group Details

The **Test Case Details** and **TC Group Details** panel displays performance characteristics of test cases / **TC Group** calculated by aggregating their iterations.

**Test Case Grid Columns**

| **Column** | **Description** |
|---|---|
| T.C. | Test Case Name |
| VUs | Number of users running the Test Case |
| Iterations Started | The number of test iterations started |
| Iterations Passed | The number of Iterations in which all responses were received |
| Iterations Failed | The number of failed (aborted) iterations |
| Incomplete Iteration's Requests | The number of requests sent in the last incomplete Iteration |
| Requests | The total number of requests sent in all Iterations |
| Avg. (s) | The average iteration time |
| Errors | The number of response errors |
| Timeouts | The number of response timeouts |
| Think time | Think time settings |
| Test case delay | Delay after the test case run |

**Test Case Group Grid Columns**

| **Property** | **Description** |
|---|---|
| TC Group | Test Case Group Name |
| VUs | Number of users running the Test Case |
| Iterations Started | The number of test iterations started |
| Iterations Passed | The number of Iterations in which all responses were received |
| Iterations Failed | The number of failed (aborted) iterations |
| Incomplete Iteration's Requests | The number of requests sent in the last incomplete Iteration |

| | |
|---|---|
| Requests | The total number of requests sent in all Iterations |
| Avg. (s) | The average iteration time |
| Errors | The number of response errors |
| Timeouts | The number of response timeouts |
| Think time | Think time settings |
| Test case delay | Delay after the test case settings |

#### 7.3.3.7 Agent Details

The **Agent Details** panel displays statistics and performance characteristics aggregated by the agent.

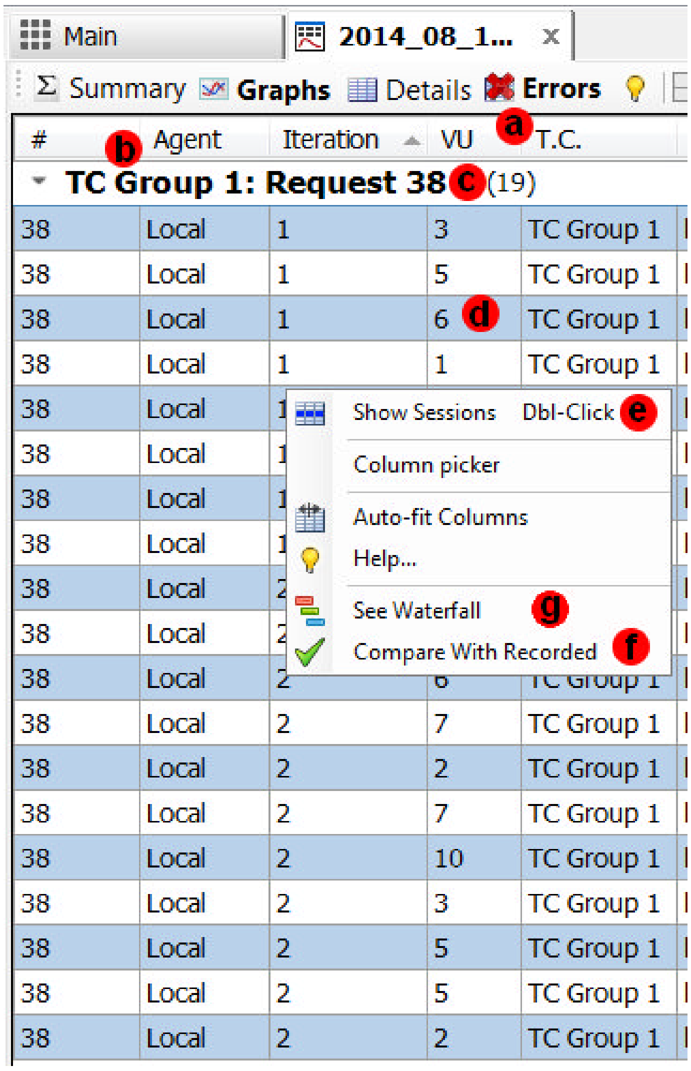
**Agent Details Grid Columns**

| Column | Description |
|---|---|
| Agent | Load Agent name |
| Start | Test start time |
| End | Test end time |
| Aggregate (s) | Aggregate Session Time, the sum of all session response times. The session duration is the time between when a request is sent and the corresponding response is received.<br><br>Because sessions are occurring in parallel, the Aggregate Session Time substantially exceeds the clock time. |
| Max VUs | The maximum number of VUs, reached during the test run |
| Sent (KB) | The sum of all requests sizes |
| Received (KB) | The sum of all responses sizes |
| Sent (KB/s) | Average upload bandwidth [Total KB sent] / [Test run duration] |
| Received (KB/s) | Average download bandwidth [Total KB sent] / [Test run duration] |
| Errors | The number of responses with status codes in the 400s or 500s range, or where a custom error was registered |

| | |
|---|---|
| Iterations Started | The number of test iterations started |
| Iterations Passed | The number of Iterations in which all responses were received |
| Iterations Failed | The number of failed (aborted) iterations |
| Iteration Time (s) | Average Iterations duration |
| Requests | The number of requests issued during the test run |
| Request /sec | Request rate [Total Requests] / [Test run duration] |
| Response (s) | Average Response time [Aggregate Session time] / [Total Requests] |

### 7.3.4 Error View

To select Error View, click **Error** on the toolbar (a).



**TC Group 1: Request 38** (19)

| # | Agent | Iteration | VU | T.C. |
|---|---|---|---|---|
| 38 | Local | 1 | 3 | TC Group 1 |
| 38 | Local | 1 | 5 | TC Group 1 |
| 38 | Local | 1 | 6 | TC Group 1 |
| 38 | Local | 1 | 1 | TC Group 1 |
| 38 | Local | 1 | [illegible] | [illegible] |
| 38 | Local | 1 | [illegible] | [illegible] |
| 38 | Local | 1 | [illegible] | [illegible] |
| 38 | Local | 1 | [illegible] | [illegible] |
| 38 | Local | 2 | [illegible] | [illegible] |
| 38 | Local | 2 | [illegible] | [illegible] |
| 38 | Local | 2 | 6 | TC Group 1 |
| 38 | Local | 2 | 7 | TC Group 1 |
| 38 | Local | 2 | 2 | TC Group 1 |
| 38 | Local | 2 | 7 | TC Group 1 |
| 38 | Local | 2 | 10 | TC Group 1 |
| 38 | Local | 2 | 3 | TC Group 1 |
| 38 | Local | 2 | 5 | TC Group 1 |
| 38 | Local | 2 | 5 | TC Group 1 |
| 38 | Local | 2 | 2 | TC Group 1 |

Every request issued during the load test receives one or three flags: Passed, Failed with Error and Failed with Timeout which is determined as follows:

1.

   a. **Failed with Error**: The response came back before a given timeout, but it violated a custom validator's criteria **or** has a failing code of 400 and above.

   b. **Failed with Timeout:** The corresponding response did not come back before a given timeout.

   c. **Passed**: None of the above. This request is considered successful.

Error View displays responses with errors and timeouts. A panel shows information about failed request instances grouped by a Test Case (b) and a request number (c). Below is the example displaying 19 errors of the request # 38 in the TC Group 1. This request failed 8 times during iteration 1 and 11 times during iteration 2. Failing users are displayed in the VU column (d). To view the selected error, double-click it or right-click on a grid and click **Show Sessions** (e). The sessions will be retrieved from the test log and displayed in the session grid.

For quick error analysis, select a row, right-click and select **Compare with Recorded** (f). The system will determine which user in which iteration encountered this error and then will display a tree which compares all replayed sessions for this user's iteration with the recorded sessions. This can help to discover preceding and subsequent errors and find out the root cause of the error faster. Another troubleshooting option is in the error view is right-click on the error and select **See Waterfall** (g) for this VU iteration.

#### 7.3.4.1 Error Details Grid Columns

| Column | Description |
|---|---|
| # | The request number |
| Agent | The Agent name |
| Iteration | The Iteration number |
| VU | The Virtual User (VU) number |
| T.C. | The Test Case name |
| URL | The URL |
| Description | The error description |

## 7.3.5 VU Activity View

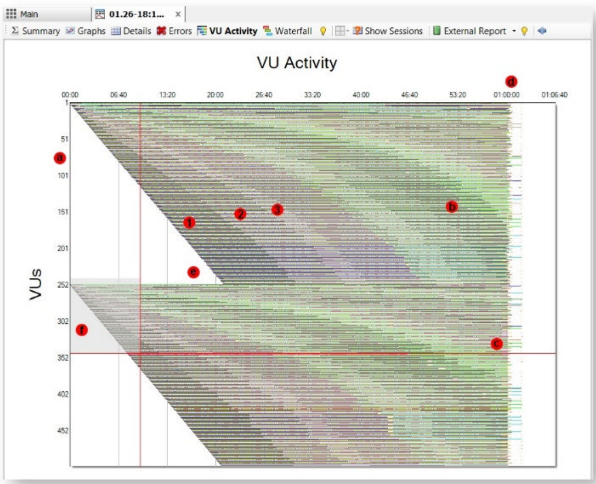
Ent SP

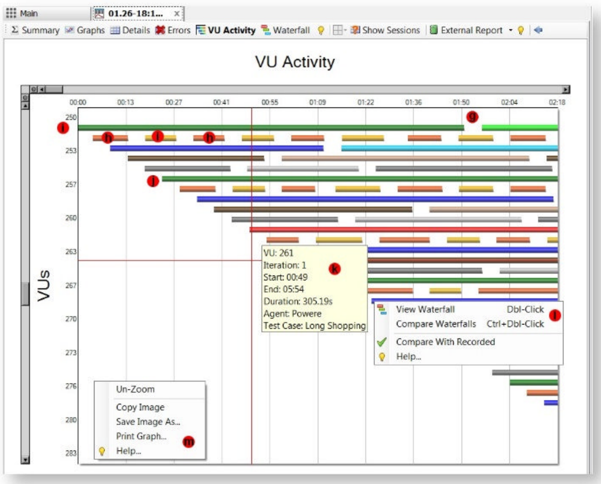
The VU Activity Chart shows the activity of every VU associated with the load test. It allows to visualize the VU activity and how it relates to other VUs actions. Specifically, it shows which test case and iteration each VU was executing, on which load generator it was emulated and what other VUs were executing at the same time. Because this information is depicted on the timeline, it helps to isolate performance issues by seeing how load patterns and test case concurrences correlate with slower test iterations. You can further drill down every test iteration to display a waterfall chart corresponding to this iteration which displays activity detailed on the level of individual pages and requests.

**Info:** The VU Activity Chart is available only after the load after has finished running.

The example of VU Activity chart is provided on the right:

- Vertical axle shows VUs (a).
- VUs 1-250 were emulated on a controller (b); VUs 251-500 were emulated on the agent (c).
- Horizontal axle (d) is the timeline. Test duration was a little longer than one hour.
- During the first 22 minutes, both, the controller and the agent, where ramping up VUs to the full capacity of 250 VUs each (e).
- To zoom-in to a specific VUs/Iterations range, select an appropriate rectangular area (f).
- Each VU is depicted by a set of horizontal bars, each of which represents single test





iteration.

- The length of a bar indicates how long it took to complete the iteration. Breaks between the bars (g) represent delays after the test cases.
- The bars are color-coded. Each test case is assigned two unique colors. One of them is used for even iterations (h) and another one is used for odd iterations (i) (alternate coloring). For example, VU 251 and 256 (j) executed the same test case. Also, areas 1, 2 and 3 display how the first three iterations where executed by VUs at different times.
- To display details of each test iteration, mouse over its bar. The tooltip (k) will display the following information:
    - VU number
    - Iteration number
    - Iteration start and end time
    - Iteration duration
    - Agent name
    - Test case name

**Generating Test Iteration Waterfall**

- To display a waterfall of an iteration, Dbl-Click its horizontal bar, or right-click (l) and select **View Waterfall**.
- You can compare waterfalls of two iterations. To do so, after selecting the first waterfall, navigate back to **VU Activity** view and Ctrl+Dbl-Click the second iteration bar, or right-click it (l) and select **Compare Waterfalls.**

**Other Commands:**

- To zoom-out, right-click (m) and select Un-Zoom.
- For additional context menu commands, right-click and select **Copy Image, Save Image** or **Print Graph Image.**

## 7.3.6 Waterfall View

Ent SP

One of the most effective tools for evaluating the performance-related user experience of accessing a webpage is a Waterfall chart. It depicts a timeline diagram of requesting and loading page resources . Waterfall charts are broadly used by performance engineers to granularly analyze factors impacting page responsiveness. It gives full visibility into every single piece of content that is downloaded from the server. Not only does it present all the resources, but it shows them in sequence on the timeline with indication when every request was issued and when every response was received. As a result, the duration of each load is clearly visualized in the context of other activities on the page. This allows to easily pinpoint bottlenecks on slow webpages.

In StresStimulus, waterfall charts are extended beyond analyzing a single page and single user experience. You can analyze two waterfall charts of a test iteration, page or transaction side-by-side in conjunction with KPI of a website with given load. Each of charts depicts a timeline of loading the iteration / page / transaction by any VU during any iteration of a load test.

This provides the following benefits:

1. Helps to visualize the impact of different levels of the server load on individual requests and overall user experience.

**Example**: in a load test with user ramp-up, compare waterfall chart for VU1 on the first iteration (light load) and on the last iteration (heavy load). Determine which requests are affected by a higher load. Recognize if load-related delays are in the server processing or on the network side.

2. Helps to better understand and or audit result of a load test.

**Example**: on a load test report, a page response time seems to be too high. To clarify this issue, analyze waterfall charts to determine on which processes most of the response time was spent. Such analysis can rectify incorrect expectation and provide the insight to the web system behavior.

3. Helps to better evaluate application scalability and isolate bottlenecks.

**Example**: a load test report indicated that a transaction becomes slow when a certain load level is reached. To further narrow down the problem area, analyze the transaction waterfall to determine a processes or request causing the most delays.

The waterfall view of the test result displays all requests in a test iteration. Page and transaction sub reports described in the Page and Transaction Result Tab section also have waterfall views which display fewer requests, limited to a single page or transaction, but otherwise are very similar to the test result waterfalls. A description of waterfall charts provided in the subsequent pages of this section applies for test, page and transaction waterfalls.

### 7.3.6.1 Single Waterfall Chart

To select Waterfall View, click **Waterfall** (a) on the toolbar.

By default, Waterfall includes one panel (b) to displays a chart for a selected VU (c) and iteration (d).

On the chart, the vertical axle (e) displays requests, and horizontal axle (f) displays the timeline.

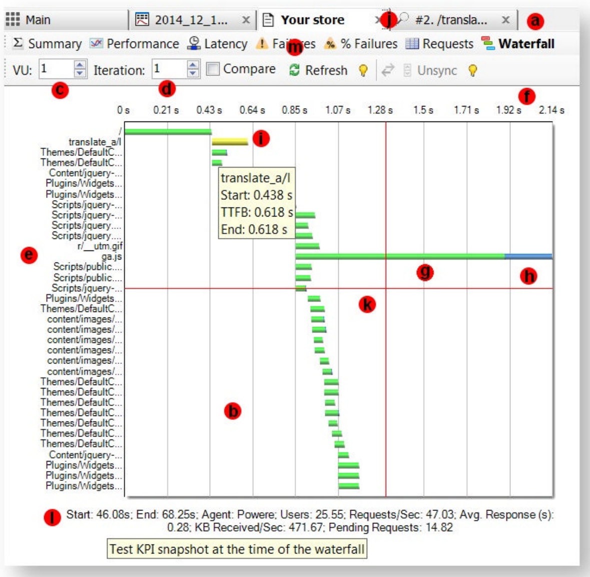
Request bars depict with green horizontal area (g) representing server time and a blue area (h) representing network time.

To display request details, mouse over its bar. It turns yellow (i) and a tool-tip displays the request name and the following timing information:

- Start - the time of issued the request
- TTFB - the time of receiving the first byte
- TTLB - the time of receiving the last byte
- End - the time when the resource was downloaded

All times are measured from the moment of issuing the first request in the page / transaction.

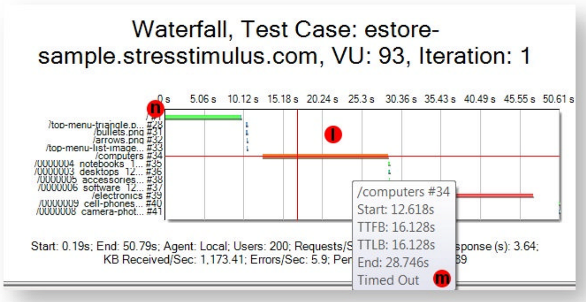
To access the content of this request, double-click it and a session inspector will open in a



translate_a/l
Start: 0.438 s
TTFB: 0.618 s
End: 0.618 s

Start: 46.08s; End: 68.25s; Agent: Powere; Users: 25.55; Requests/Sec: 47.03; Avg. Response (s): 0.28; KB Received/Sec: 471.67; Pending Requests: 14.82

Test KPI snapshot at the time of the waterfall

| | |
|---|---|
| new tab (j) (see Inspecting Sessions).<br><br>To evaluate positioning of request/response events of page/transaction resources on the timeline, click the appropriate point in the graph panel and a vertical and horizontal red line (k) crossing this point will appear. | |
| Session errors and timeouts are also color-coded (l). Sessions with an error or timeout are displayed as orange bar. The tool-tip displayed on mouse over displays its error or time out status (m).<br><br>To identify a specific request, look for its request URL or a session number (n). |  |

Waterfall, Test Case: estore-sample.stresstimulus.com, VU: 93, Iteration: 1
0 s
5.06 s
10.12 s
15.18 s
20.24 s
25.3 s
30.36 s
35.43 s
40.49 s
45.55 s
50.61 s
/ #1
/top-menu-triangle.p... #28
/bullets.png #31
/arrows.png #32
/top-menu-list-image... #33
/computers #34
/0000004 notebooks 1... #35
/0000003 desktops 12... #36
/0000005 accessories... #38
/0000006 software 12... #37
/electronics #39
/0000009 cell-phones... #40
/0000008 camera-phot... #41
n
l
m

Start: 0.19s; End: 50.79s; Agent: Local; Users: 200; Requests/S sponse (s): 3.64;
KB Received/Sec: 1,173.41; Errors/Sec: 5.9; Per 39

/computers #34
Start: 12.618s
TTFB: 16.128s
TTLB: 16.128s
End: 28.746s
Timed Out

To put the waterfall in the context of the load test, a key performance indicator snapshot (l) taken during the time of the waterfall is displayed on the bottom. it includes the following parametersStart, End - waterfall beginning and ending timestamps on the test timeline.

- Start, End - waterfall beginning and ending timestamps on the test timeline.
- Agent - The name of the Agent where the waterfall was captured.
- **Users** - The number of instantiated and active VUs, iterating trough their respective test cases. Some VUs can complete the test earlier than others and become inactive. VUs which completed all their iterations before the test end are excluded from the active user count. If the test is configured to complete only after all VUs complete their iterations, then the User graph will show gradual declining the number of VUs at the end of the test.
- **Req/Sec** - The average number of requests being sent per second.
- **Avg. Response(s)** - The average response time of the sent requests. Every datapoint on the response time graph reflects the average value during the checkpoint intervals. If **no responses were received** between two datapoints, the response time cannot be determined, so such datapoints are skipped to reflect an accurate response time curve.
- **KB Received/Sec** - The average number of bytes received per second.
- **Pending Requests** - The average number of pending requests. Pending requests are those that are issued but the response is not yet received by StresStimulus. Generally, the greater number of pending requests indicates slower server response. This parameter can be used to gauge performance change in several tests runs.

To select a different VU / iteration, adjust information in the numeric text boxes (c) and (d). Sometimes, you may also need to click the Refresh button (m).

**Note:** In order to generate a waterfall chart, session content must be saved in storage accessible from the controller. This information is available in tests that ran from the controller without agents . In distributed tests , when SQL Server is used as storage, this information is also available on the controller. However, in distributed tests with SQL Server CE-based storage, the sessions initiated on the agents are stored on the agents. You still can create waterfall chart on the controller, if **Save sessions from agents** property in **Configure Test** -> **Test Result Storage** section is set to **Yes** (default), because this setting will force replicating necessary data from the agents to the controller.

### 7.3.6.2 Waterfall Chart Commands

Several commands are available to customize waterfall charts:**Zoom / Un-Zoom**

- To zoom-in on a graph, select a rectangular area (a) where you wish to zoom. Vertical and horizontal scroll-bars (b) will appear.
- For additional zooming, select a rectangular area again.
- To undo to last zoom, click the minus icon on the scroll bars (b).
- To completely remove zoom, right-click and select **Un-Zoom** (c).

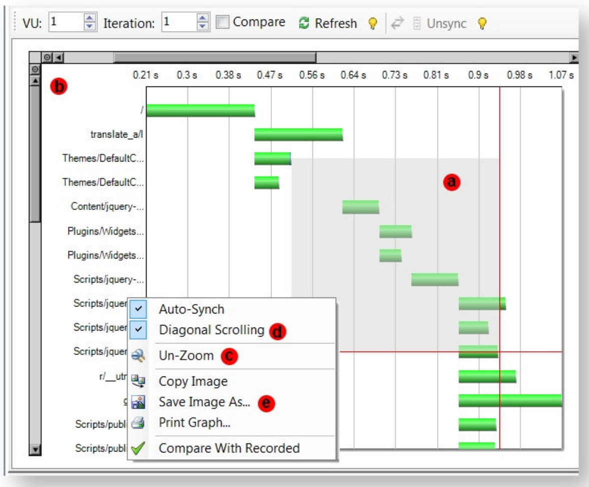
**Diagonal Scrolling**

Because a waterfall has a "diagonal" shape, vertical scrolling can move all request bars outside the visible area of the chart. To bring them back, additional horizontal scrolling is necessary which complicates this operation.

To solve this issue, the ratio of horizontal scrolling to vertical scrolling is set, so the request bars are always visible. This method is called Diagonal Scrolling.

To turn Diagonal Scrolling off, right-click and un-check the Diagonal Scrolling box (d).**Other commands (e):**

- Copy Image
- Save Image as



| | |
|---|---|
| • Print Graph | |
| | |

### 7.3.6.3 Dual Waterfall Chart

Two charts can be displayed side-by-side to compare waterfalls of any two VUs during selected iterations.

To turn-on the dual Waterfall view, check the **Compare** box (a) on the toolbar. The 2nd pair of numeric text boxes (b) will appear. Select the 2nd VU / Iteration. The second panel (c ) with the graph will appear on the right.

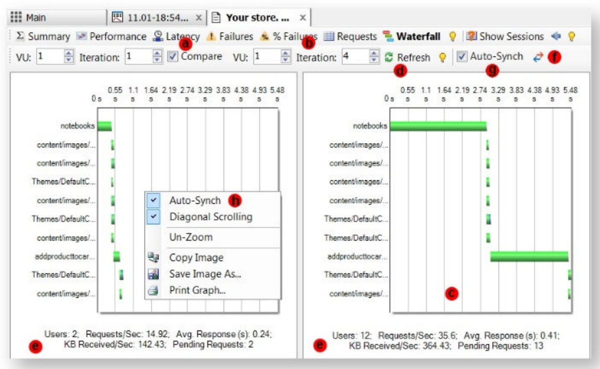
> **Note:** Sometimes you need to click **Refresh** (d) on the toolbar after you change a VU or iteration.

The initial timeline of the two charts has the same scale for easier comparison. **Example on the figure**: transactions on the left and right charts completed in 0.7 and 5.48 seconds respectively. Both charts will display a 5.48 seconds timeline.

A snapshot of KPI of the time of the page / transaction request will be displayed below each waterfalls (e). For example in the figure:

- the left waterfall took place when an application under the test was accessed by 2 users with hit rate 14.92 req/sec
- the right waterfall took place when an application under the test was accessed by 12 users with hit rate 35.6 req/sec

To swap the charts, click the **Swap** button (f) on the toolbar.



Users: 2; Requests/Sec: 14.92; Avg. Response (s): 0.24; KB Received/Sec: 142.43; Pending Requests: 2

Users: 12; Requests/Sec: 35.6; Avg. Response (s): 0.41; KB Received/Sec: 364.43; Pending Requests: 13

### Auto-Synch mode

By default, the scrolling and zooming in the charts is synchronized. When you zoom or vertically scroll either of the charts, the opposite chart will automatically keep the same vertical zoom / scroll position, so the corresponding requests in both charts are always aligned.

To turn Auto-Synch mode off, un-check the Auto-Synch on the toolbar (g) or right-click and un-check the Auto-Synch box (h). After that, you can zoom and scroll charts independently.

## 7.3.7 Finding Hidden Errors

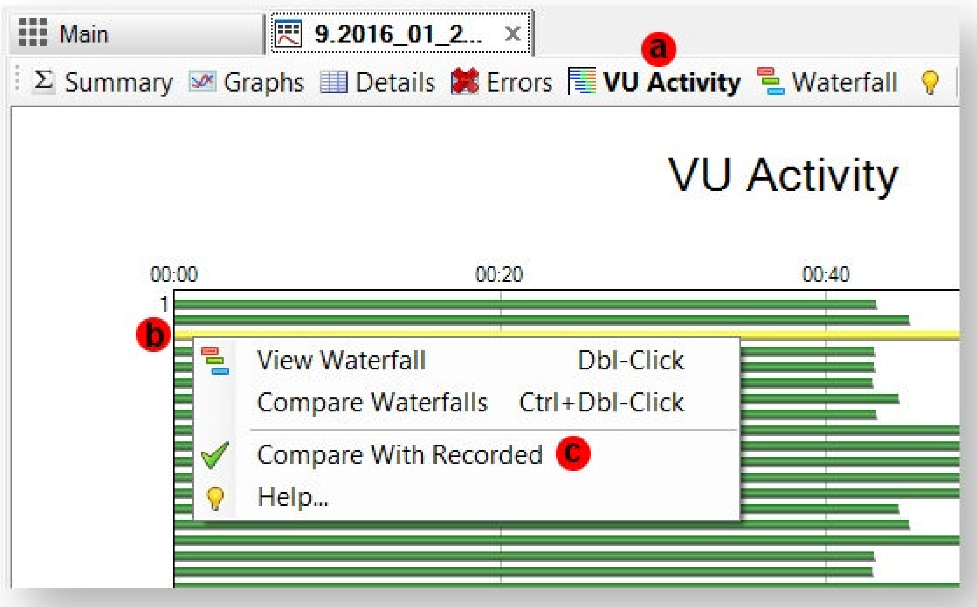
Ent SP

Depending on the application, in some cases, the application starts returning HTTP errors in the response when the server load gets too high. For example, with too many concurrent requests, the application's database begins throwing connection timeout errors.

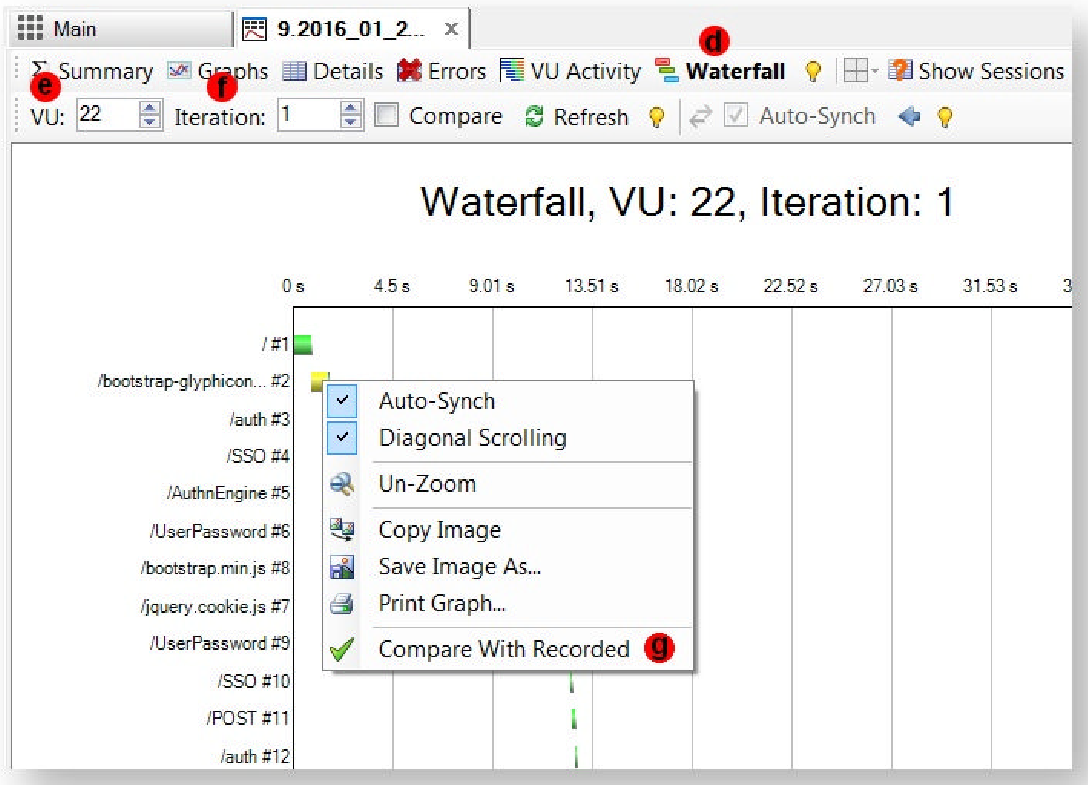
Usually, HTTP errors have an error response code, such as 500. In this case StresStimulus will report this response as an error. However, in some instances, the response code is 200 (means an successful response) and the response body has some error text. In this case, StresStimulus will not be able to determine this response as an error and therefore it is considered as a hidden error.

In order to expose these hidden errors, you can compare a replayed iteration with the recorded test case. There are two ways of doing that:

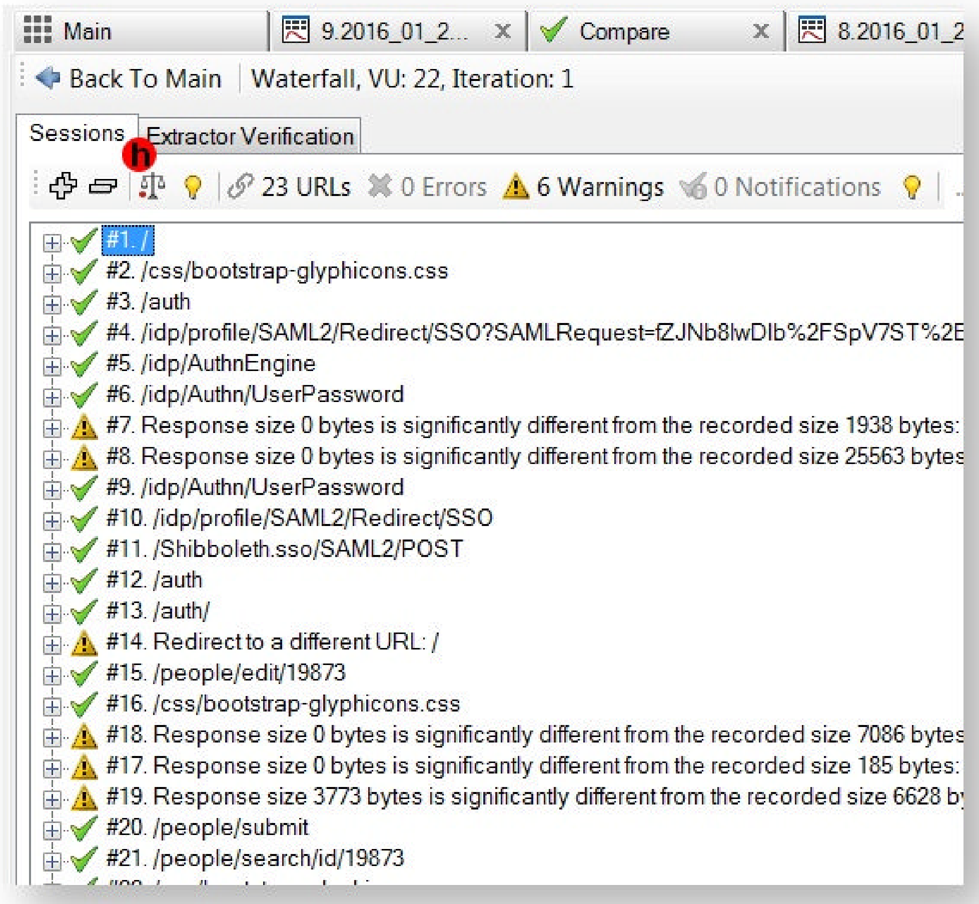
1. Open the VU Activity View (a) and right click on the desired VU and Iteration (b) to investigate. Right click and select Compare With Recorded (c).



2. Open the Waterfall chart (d) and select the desired VU (e) and Iteration (f) to investigate. Right click and select Compare With Recorded (g).



In either case. a verification tree (h) that shows a comparison report between the recorded and replayed iteration will open. It is similar to verification.

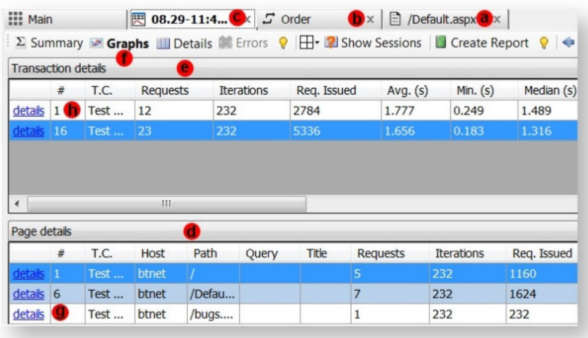


```
#1. /
#2. /css/bootstrap-glyphicons.css
#3. /auth
#4. /idp/profile/SAML2/Redirect/SSO?SAMLRequest=fZJNb8lwDIb%2FSpV7ST%2E
#5. /idp/AuthnEngine
#6. /idp/Authn/UserPassword
#7. Response size 0 bytes is significantly different from the recorded size 1938 bytes:
#8. Response size 0 bytes is significantly different from the recorded size 25563 bytes
#9. /idp/Authn/UserPassword
#10. /idp/profile/SAML2/Redirect/SSO
#11. /Shibboleth.sso/SAML2/POST
#12. /auth
#13. /auth/
#14. Redirect to a different URL: /
#15. /people/edit/19873
#16. /css/bootstrap-glyphicons.css
#18. Response size 0 bytes is significantly different from the recorded size 7086 bytes
#17. Response size 0 bytes is significantly different from the recorded size 185 bytes:
#19. Response size 3773 bytes is significantly different from the recorded size 6628 by
#20. /people/submit
#21. /people/search/id/19873
```

## 7.4 Page and Transaction Result Tab

Page Result (a) and Transaction Result (b) tabs display performance characteristics of a single page or a transaction registered during a particular test run. You can open these tabs from the Test Result tab (c) in one of 3 ways:

1. Page Details grid (d) and Transaction Details grid (e), located in the Detail view (f) on the Test Result tab, display hyperlinks to all pages (g) and transactions (h). Click a hyperlink to open the corresponding Page Result or Transaction Result tab.



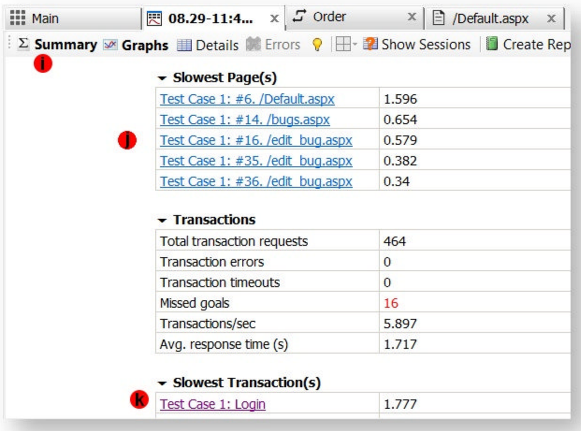
Transaction details

| | # | T.C. | Requests | Iterations | Req. Issued | Avg. (s) | Min. (s) | Median (s) |
|---|---|---|---|---|---|---|---|---|
| details | 1 | Test ... | 12 | 232 | 2784 | 1.777 | 0.249 | 1.489 |
| details | 16 | Test ... | 23 | 232 | 5336 | 1.656 | 0.183 | 1.316 |

Page details

| | # | T.C. | Host | Path | Query | Title | Requests | Iterations | Req. Issued |
|---|---|---|---|---|---|---|---|---|---|
| details | 1 | Test ... | btnet | / | | | 5 | 232 | 1160 |
| details | 6 | Test ... | btnet | /Defau... | | | 7 | 232 | 1624 |
| details | | Test ... | btnet | /bugs.... | | | 1 | 232 | 232 |

2. Summary view (i) on the Test Result tab displays hyperlinks to up to 5 slowest pages (j) and up to 5 slowest transactions (k). Click a hyperlink to open a Page Result or Transaction Result tab.



Slowest Page(s)

| | |
|---|---|
| Test Case 1: #6. /Default.aspx | 1.596 |
| Test Case 1: #14. /bugs.aspx | 0.654 |
| Test Case 1: #16. /edit_bug.aspx | 0.579 |
| Test Case 1: #35. /edit_bug.aspx | 0.382 |
| Test Case 1: #36. /edit_bug.aspx | 0.34 |

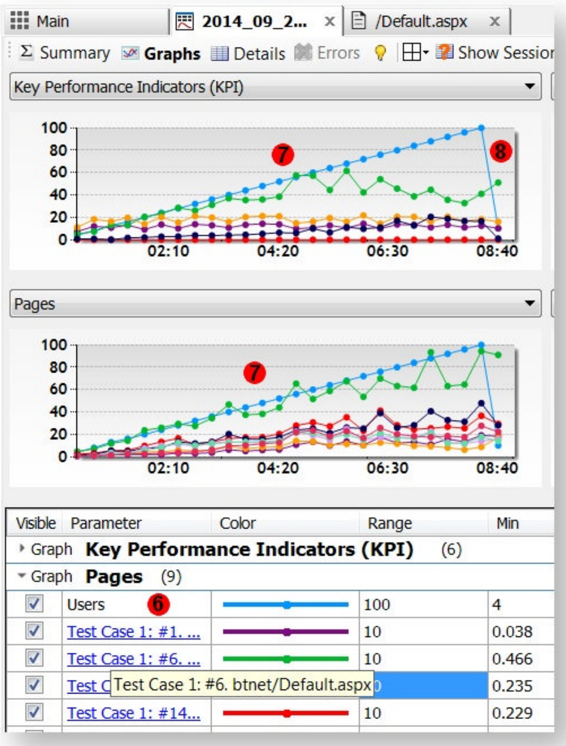
Transactions

| | |
|---|---|
| Total transaction requests | 464 |
| Transaction errors | 0 |
| Transaction timeouts | 0 |
| Missed goals | 16 |
| Transactions/sec | 5.897 |
| Avg. response time (s) | 1.717 |

Slowest Transaction(s)

| | |
|---|---|
| Test Case 1: Login | 1.777 |

3. Page section (l) and Transaction section of the Curve Grid located in the Graph View (m) displays links (n) to all pages and transactions. Click a hyperlink to open the corresponding Page Result or Transaction Result tab.

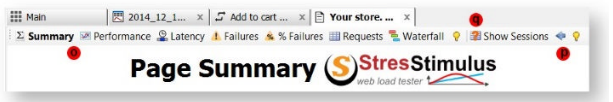


| Visible | Parameter | Color | Range | Min |
|---|---|---|---|---|
| Graph | Key Performance Indicators (KPI) (6) | | | |
| Graph | Pages (9) | | | |
| ✓ | Users | | 100 | 4 |
| ✓ | Test Case 1: #1. ... | | 10 | 0.038 |
| ✓ | Test Case 1: #6. ... | | 10 | 0.466 |
| ✓ | Test C Test Case 1: #6. btnet/Default.aspx | | | 0.235 |
| ✓ | Test Case 1: #14... | | 10 | 0.229 |

The following views (o) can be selected on the toolbar:

- Summary View
- Performance View
- Latency View
- Failure View
- Failure % View
- Requests View
- Waterfall View

To go back to the Test Result, click Back (p).



To retrieve page or transaction sessions from the test log, click **Show Session** (q) on the toolbar. The **Query Log** dialogue with pre-populated session range and the test case, will open. Click **Show Session.** Session grid will display the result,

### 7.4.1 Summary View

Summary view lists page or transaction basic performance metrics and failures. To select Summary View, click **Summary** on the toolbar.

The Summary includes several subsections that can be expanded / collapsed by clicking the triangle icon.

#### 7.4.1.1 General

| | |
|---|---|
| Result name | Optional one-word Test Run Name specified in the Configure Test section. It is used as a suffix following a time-stamp of the test run displayed in the "Analyze Results" section. |
| Test run description | Test description specified in the Configure Test section prior to the test run. |
| Test File name | The name of the test script file |
| Start time | Test start time |
| Name | The page or transaction name |
| Description | The page or transaction description |
| Test Case | The test case name |
| Number of URLs | The number of recorded URLs in the page or transaction |
| Title | The page title |
| Host | The host |
| Path | The page path |

| | |
|---|---|
| Query | The query string |

### 7.4.1.2 Performance Metrics

| | |
|---|---|
| Iterations | The number of page or transaction iterations started |
| Requests | The number of requests in the page or transaction |
| Avg. response time (s) | The average response time (s) |
| Min. Response (s) | The minimum response time |
| Median Response (s) | The median response time |
| Max. Response (s) | The maximum response time |
| Std. Dev. | The response time standard deviation |
| Fastest 90% (s) | The maximum response time after excluding the slowest 10% of page iterations |
| Fastest 95% (s) | The maximum response time after excluding the slowest 5% of page iterations |
| Fastest 99% (s) | The maximum response time after excluding the slowest 1% of page iterations |

### 7.4.1.3 Failures

| | |
|---|---|
| Goal (s) | The page or transaction response time goal |
| Missed Goals | The number of iterations where the response time exceeded the goal |
| Errors | The number of iterations with errors |
| Timeouts | The number of iterations with timeouts |
| Missed Goals % | % of iterations where the response time exceeded the goal |
| Errors % | % of iterations with errors |
| Timeouts % | % of iterations with timeouts |
| Error requests | The number of error requests registered during all iterations |
| Timeout requests | The number of timeout requests registered during all iterations |

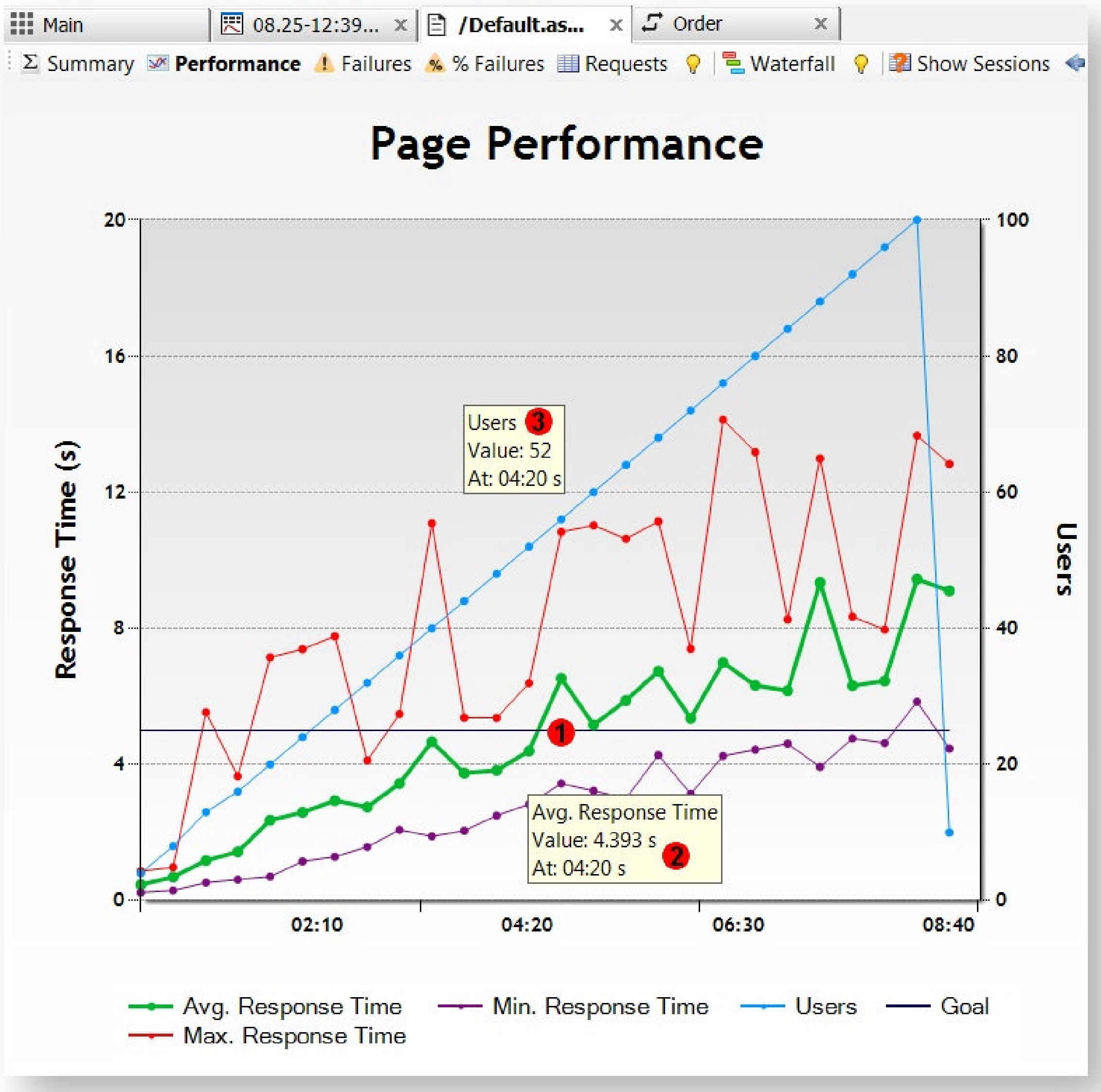
#### 7.4.1.4 Screenshot

Displays the page screenshot if it is exists.

### 7.4.2 Performance View

Ent SP

Performance view presents a page or a transaction response timeline and its changes, depending on the number of emulated VUs. It features 5 curves: minimum, average and maximum response time, goal and number of VUs.



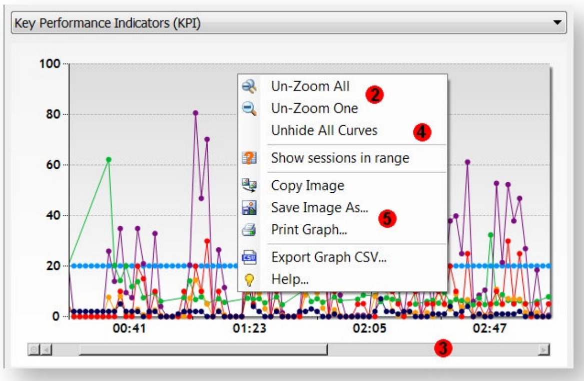
Performance view graph helps to analyze scalability and to determine how many virtual users the page or transaction can handle. For example, to determine how many users can be handled to meet the goal of 5 seconds in at least 50% cases, follow these steps:

1. Find an approximate point of intersection of Avg. Response Time curve (green) and Goal curve (navy blue).
2. Mouse over it to determine its timestamp (04:20)
3. Find a point with the same timestamp on the Users curve (blue) and note the number of users reached this point (52)

This page can handle 52 concurrent users to meet the page response goal of 5 seconds in at least 50% cases.

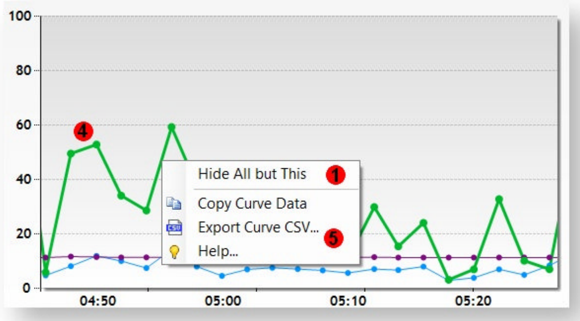
### 7.4.2.1 Graph Context Menu

Multiple commands are available to customize graphs and access graph information. Some of them are located in the context menus exposed when right-click on a graph or a curve.

| | |
|---|---|
| Graph commands<br>1. To zoom-in on a graph, select a time range that you wish to zoom.<br>2. To zoom-out one step, click **Un-Zoom One**. To remove zoom, click **Un-Zoom All**.<br>3. To stop time auto-scrolling, scroll to the left.<br>To resume time auto-scrolling, scroll to the right.<br>4. To show hidden curves, click **Unhide**.<br>5. Other commands:<br>•<br>○<br>▪ Copy, Save, Print Graph Image<br>▪ Export Graph datapoints |  Key Performance Indicators (KPI)<br>Un-Zoom All<br>Un-Zoom One<br>Unhide All Curves<br>Show sessions in range<br>Copy Image<br>Save Image As...<br>Print Graph...<br>Export Graph CSV...<br>Help...<br>2<br>4<br>5<br>3<br>100<br>80<br>60<br>40<br>20<br>0<br>00:41<br>01:23<br>02:05<br>02:47 |

### 7.4.2.2 Graph Curve Context Menu

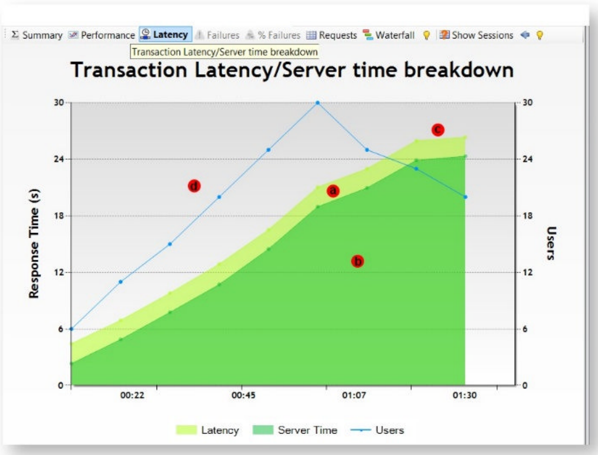
Multiple commands are available to customize graphs and access graph information. Some of them are located in the context menus exposed when right-click on a graph or a curve.

| | |
|---|---|
| Curve Commands<br>To access curve commands, mouse over a curve and then right-click<br>1. To hide all but the selected curve, click "Hide".<br>2. To hide / unhide specific curves, check /uncheck corresponding Visible boxes in the curve grid.<br>3. To unhide all curves, in the graph context menu, click **Unhide All**. |  Hide All but This<br>Copy Curve Data<br>Export Curve CSV...<br>Help...<br>1<br>4<br>5<br>100<br>80<br>60<br>40<br>20<br>0<br>04:50<br>05:00<br>05:10<br>05:20 |

| | |
|---|---|
| 4. To highlight a curve, click its name in the curve grid or mouse over it in the graph.<br>5. To copy or export core data, click Copy or Export. | |

## 7.4.3 Latency View

Ent SP

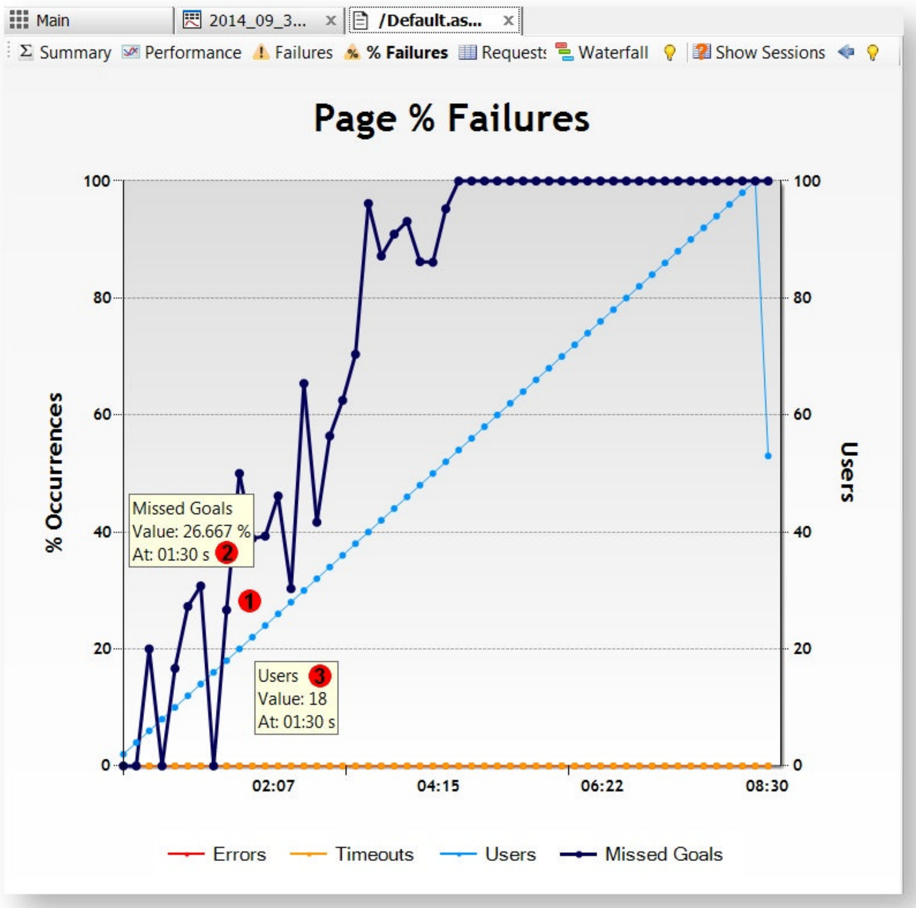
| | |
|---|---|
| Latency view presents a page or transaction response time breakdown between Latency and Server Time. The latency (or network time) is a portion of the response time attributed to the network delays, necessary for server responses to reach the client.<br><br>It depends on several factors, such as the response size, network bandwidth, server and client global and network positioning on the Internet. Latency view graph displays the following elements:<br><br>a) Latency area<br><br>b) Server Time area<br><br>c) Total response time = Server Time +Latency<br><br>d) User curve<br><br>In slow pages and transactions, the total response time breakdown helps to determine whether the slowness is caused by the server processing or by the network latency. Resolving each of these two bottlenecks entails very different performance engineering steps. To address high latency consider the following measures:<br><br>• Decrease server response size by optimizing the application and / or using compression.<br><br>• Reduce the number of network round-trips.<br><br>• Increase network bandwidth.<br><br>• Move the server geographically closer to the clients use CDN. |  Summary<br>Performance<br>Latency<br>Failures<br>% Failures<br>Requests<br>Waterfall<br>Show Sessions<br>Transaction Latency/Server time breakdown<br>Transaction Latency/Server time breakdown<br>Response Time (s)<br>Users<br>0<br>6<br>12<br>18<br>24<br>30<br>00:22<br>00:45<br>01:07<br>01:30<br>a<br>b<br>c<br>d<br>Latency<br>Server Time<br>Users |

|  |  |
|---|---|
| • Use client caching. |  |

## 7.4.4 Failure View

Ent SP

Failure and Failure % view presents a page or a transaction failure rate on a timeline and its changes depending on the number of emulated VUs. Four curves are displayed: errors, timeouts, users and missed goals. This view can be used to answer many performance related questions regarding quality of service as a function of load.



Failure view's graph helps to analyze scalability and to determine how many virtual users the page or transaction can handle.

For example, to determine how many users can be handled to meet the response time goal in at least 75% cases, follow these steps:

1. Find an approximate point of intersection of Missed Goal curve (navy blue) and 25% failure rate.
2. Mouse over it to determine its timestamp (01:30)
3. Find a point with the same timestamp on the Users curve (blue) and note the number of users reached this point (18)

This page can handle 18 concurrent users to meet page response call it, at least 75% cases.

**Note:** Failure view is available only if a page or transaction has at least one error, timeout, or missed goal occurrence.

## 7.4.5 Request View

The Request View displays aggregated performance characteristics of each request related to a page or transaction in tabular form. Time characteristics are averaged. Request counts are summed.

**Note** : If a request timed-out and subsequently failed, it's counted as a timeout.

### 7.4.5.1 Request Grid Columns

| **Column** | **Description** |
|---|---|
| # | The request number |
| T.C. | The Test Case name |
| Host | The host |
| Path | The page path |
| Query | The query string |
| Type | The request's Content-Type. |
| Response (s) | The average time to receive the response (TTLB) |
| STD | Response Time Standard Deviation |

| | |
|---|---|
| TTFB (s) | The average time to receive the first byte of the response |
| Network (s) | The average time between receiving the first and the last byte of the response necessary for transferring it over the network |
| Request (KB) | The average Request Body size |
| Response (KB) | The average Response Body size |
| Requests | The number of times the request was issued during the test run |
| Errors | The number of the request's failures during the test run |
| Timeouts | The number of the request's timeouts registered during the test run, |
| Timeout (s) | The timeout settings for this request |

## 7.4.6 VU Activity View

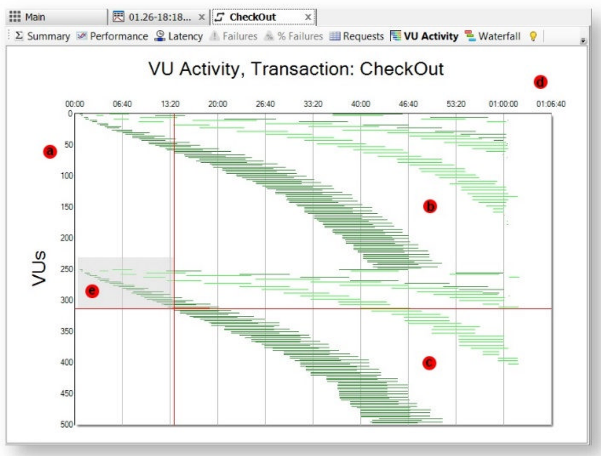
Ent SP

The VU Activity Chart for a page or a transaction shows on the test timeline which VUs requested the page or executed the transaction on different test iteration. It also shows on which agent it was emulated.

The VU Activity Chart helps to isolate performance issues by showing how load patterns and VUs concurrences correlate with slow pages / transactions. You can further drill down every page / transaction to display its waterfall, which breaks it down to the level of individual requests.

An example of a transaction VU Activity chart is provided on the right:

- Vertical axle shows VUs (a).
- VUs 1-250 were emulated on a controller (b); VUs 251-500 were emulated on the agent (c).
- Horizontal axle (d) is the timeline. Test



duration was a little longer than one hour.

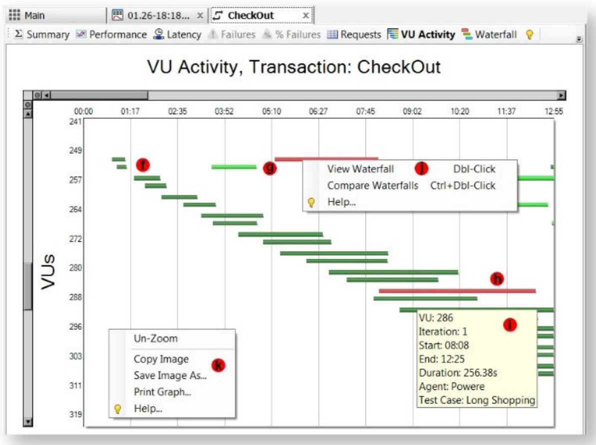
- To zoom-in to a specific VUs/Iterations range, select an appropriate rectangular area (e).
- Each page or transaction associated with a VU is depicted by a sequence of horizontal bars colored to highlight the alternate iterations: iterations, 1,3,5... are dark green (f); 2.4.6... are light green (g).
- The length indicates how long it took to complete the page or transaction on this iteration.
- To display details of each page /transaction, mouse over its bar (h). The tooltip (i) will display the following information:
    - VU number
    - Iteration number
    - Iteration start and end time
    - Page/transaction duration
    - Agent name
    - Test case name

**Generating Test Iteration Waterfall**

- To display a waterfall of a page/transaction, Dbl-Click its horizontal bar, or right-click (j) and select **View Waterfall**.
- You can compare waterfalls of two iterations. To do so, after selecting the first waterfall, navigate back to **VU Activity** view and Ctrl+Dbl-Click the second iteration bar, or right-click it and select **Compare Waterfalls.**

**Other Commands:**

- To zoom-out, right-click (k) and select Un-



VU: 286
Iteration: 1
Start: 08:08
End: 12:25
Duration: 256.38s
Agent: Powere
Test Case: Long Shopping

| | |
|---|---|
| Zoom.<br>• For additional context menu commands, right-click and select **Copy Image, Save Image** or **Print Graph Image.** | |

### 7.4.7 Waterfall View

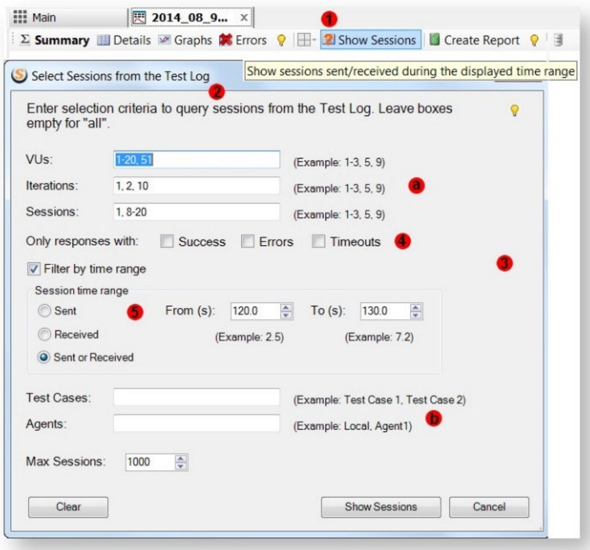
Ent SP

Waterfall View in page or transaction sub report, as compared to Waterfall View in a test result, displays fewer requests, limited to a single page or transaction, but otherwise they are very similar. Waterfall View in a test result is described in the Waterfall View subsection of the Test Result Tab section.

This section applies for test, page and transaction waterfalls.

## 7.5 Querying Test Log

During test execution, StresStimulus records HTTP session information into the test log which is stored in the test repository along with metadata. The Metadata includes information, such as VU number, iteration number and test case name, which help to find session information quickly. To display necessary replayed sessions in the session grid:



Show sessions sent/received during the displayed time range

Select Sessions from the Test Log

Enter selection criteria to query sessions from the Test Log. Leave boxes empty for "all".

| | | |
|---|---|---|
| VUs: | 1-20, 51 | (Example: 1-3, 5, 9) |
| Iterations: | 1, 2, 10 | (Example: 1-3, 5, 9) |
| Sessions: | 1, 8-20 | (Example: 1-3, 5, 9) |

Only responses with: ☐ Success ☐ Errors ☐ Timeouts

☑ Filter by time range

Session time range

Sent Received [✓ Sent or Received]

From (s): 120.0 (Example: 2.5) To (s): 130.0 (Example: 7.2)

| | | |
|---|---|---|
| Test Cases: | | (Example: Test Case 1, Test Case 2) |
| Agents: | | (Example: Local, Agent1) |

Max Sessions: 1000

1. Click **Show Sessions** located in the **Test Result** tab toolbar.

2. A query builder window pops up.

3. Each search criteria, such as VUs, Iterations, Sessions, Test Case and Agent names, has its own designated text box.

a. Multiple numeric values or ranges can be combined in each text box. For examples: for VUs, Iterations and Sessions use format 1-3, 5, 9;

b. Multiple names should be separated by a comma. For example, for Test Cases and Agents, use format Name1, Name2

**Tip:** Leave textboxes empty to broaden the search.

4. For responses with Successes, Errors and/or Timeouts: check 1 or 2 or 3 boxes. Checking all or un-checking all will render all sessions.

5. You can select sessions sent within a specific time range, received within a specific time range or combination of thereof.

- 
    - 
        - Check the **Filter by Time Range** box
        - Select **Send**, **Received** or both

| | |
|---|---|
| ▪ Enter the time range into the **From / To** boxes in seconds from the beginning of the test<br><br>**Tip:** The easiest way to enter retrieve sessions sent or received during certain time intervals it to query the test log from the graphs ( see Retrieving Sessions from Graphs).<br><br>6. Click **Show Sessions** button<br><br>7. Requested sessions will be displayed in the session grid. | |

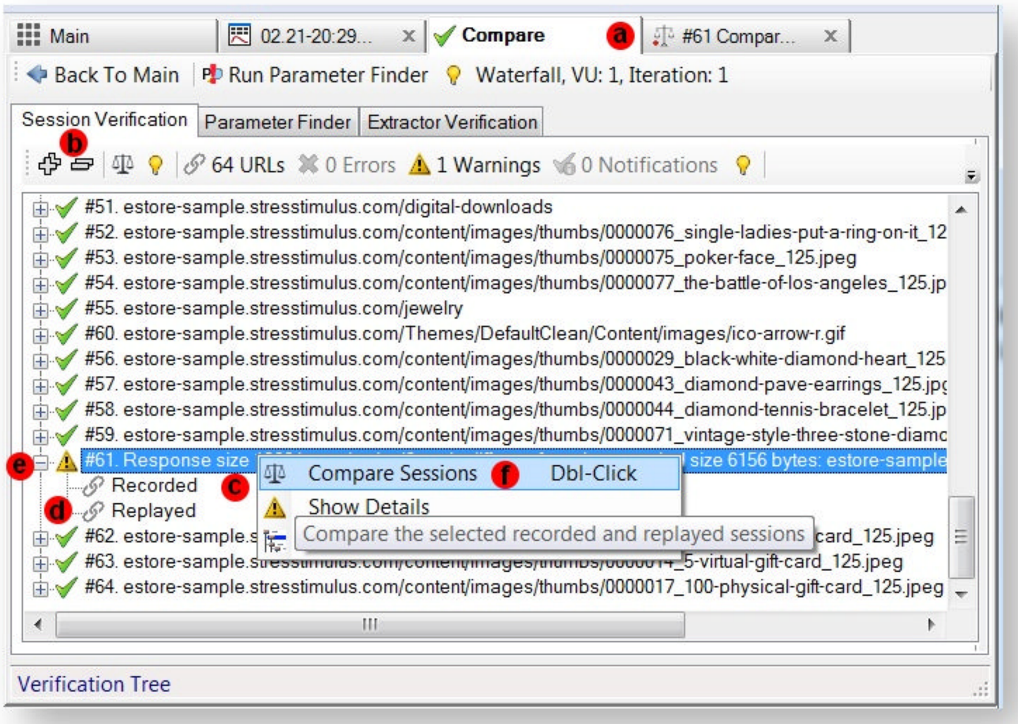
**Note:**

- Retrieving more than 1,000 records, as entered in the **Max Sessions** box can impact performance.
- Querying Test Log requires that session content be saved in the storage accessible from the controller. This information is available in tests that ran from the controller without agents. In distributed tests, however, this information is available only when SQL Server is used as storage. In distributed tests with SQL Server CE-based storage, some of the session content is stored locally on the agents and is not accessible from the controller. Therefore, only VUs emulated on the controller can be queried.

## 7.6 Finding Performance Errors

Comparing Recorded and Replayed Sessions is an effective method of detecting performance errors. The result of such comparison is displayed in a new tab (a) displaying **Session Verification** tree (b) showing the outcome of the comparison of sessions recorded (c) with the same sessions (d) replayed by particular users on a specific test run iteration. Next to every request, there is a status image (e) characterizing the outcome of comparing. The same comparison statuses are used as during the test case verification.

To compare session content, right-click on a selected session and select **Compare Sessions** (f) to display compare session inspector.



```
#51. estore-sample.stresstimulus.com/digital-downloads
#52. estore-sample.stresstimulus.com/content/images/thumbs/0000076_single-ladies-put-a-ring-on-it_12
#53. estore-sample.stresstimulus.com/content/images/thumbs/0000075_poker-face_125.jpeg
#54. estore-sample.stresstimulus.com/content/images/thumbs/0000077_the-battle-of-los-angeles_125.jp
#55. estore-sample.stresstimulus.com/jewelry
#60. estore-sample.stresstimulus.com/Themes/DefaultClean/Content/images/ico-arrow-r.gif
#56. estore-sample.stresstimulus.com/content/images/thumbs/0000029_black-white-diamond-heart_125
#57. estore-sample.stresstimulus.com/content/images/thumbs/0000043_diamond-pave-earrings_125.jpg
#58. estore-sample.stresstimulus.com/content/images/thumbs/0000044_diamond-tennis-bracelet_125.jp
#59. estore-sample.stresstimulus.com/content/images/thumbs/0000071_vintage-style-three-stone-diamc
#61. Response size ... size 6156 bytes: estore-sample
#62. estore-sample.s ... card_125.jpeg
#63. estore-sample.s ... 5-virtual-gift-card_125.jpeg
#64. estore-sample.stresstimulus.com/content/images/thumbs/0000017_100-physical-gift-card_125.jpeg
```

Comparing recorded and replayed responses can expose application errors thanks to rich text comparison capabilities of the Compare Sessions Inspector. The following response elements will be highlighted in the replayed session:

- application error messages or warnings,
- data of a significantly different size or format

Carefully examine such change to confirm or rule out the errors.

The session verification tree can be recalled from reports and from the session grid.

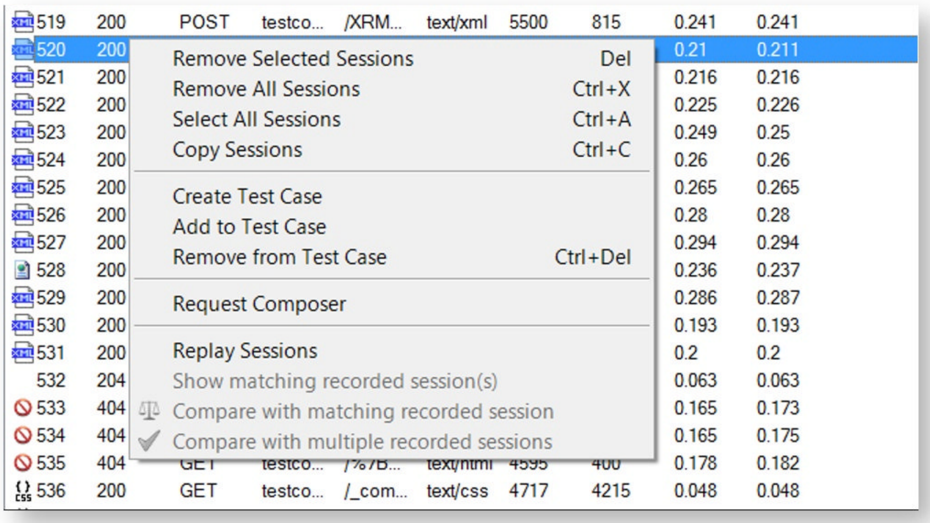
## 7.6.1 From Report

In the Error View select a row, right-click and select Compare with Recorded. The system will determine which user in which iteration encountered this error and then will display a tree which compares all replayed sessions for this user's iteration with the recorded sessions. This can help to discover preceding and subsequent errors and find out the root cause of the error faster. Another troubleshooting option is in the error view. Right-click on the error and select See Waterfall for this VU iteration  From there you can select Compare with Recorded

### 7.6.2 From Session Grid

Once the replayed sessions are retrieved from the test repository, it is often necessary to compare them to the corresponding recorded sessions for troubleshooting.

Select one or several replayed sessions and right click to bring up **StresStimulus Commands** menu. Chose from the following options:



| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 519 | 200 | POST | testco... | /XRM... | text/xml | 5500 | 815 | 0.241 | 0.241 |
| 520 | 200 | | | | | | | 0.21 | 0.211 |
| 521 | 200 | | | | | | | 0.216 | 0.216 |
| 522 | 200 | | | | | | | 0.225 | 0.226 |
| 523 | 200 | | | | | | | 0.249 | 0.25 |
| 524 | 200 | | | | | | | 0.26 | 0.26 |
| 525 | 200 | | | | | | | 0.265 | 0.265 |
| 526 | 200 | | | | | | | 0.28 | 0.28 |
| 527 | 200 | | | | | | | 0.294 | 0.294 |
| 528 | 200 | | | | | | | 0.236 | 0.237 |
| 529 | 200 | | | | | | | 0.286 | 0.287 |
| 530 | 200 | | | | | | | 0.193 | 0.193 |
| 531 | 200 | | | | | | | 0.2 | 0.2 |
| 532 | 204 | | | | | | | 0.063 | 0.063 |
| 533 | 404 | | | | | | | 0.165 | 0.173 |
| 534 | 404 | | | | | | | 0.165 | 0.175 |
| 535 | 404 | GET | testco... | /%7B... | text/html | 4595 | 400 | 0.178 | 0.182 |
| 536 | 200 | GET | testco... | /_com... | text/css | 4717 | 4215 | 0.048 | 0.048 |

Remove Selected Sessions Del
Remove All Sessions Ctrl+X
Select All Sessions Ctrl+A
Copy Sessions Ctrl+C
Create Test Case
Add to Test Case
Remove from Test Case Ctrl+Del
Request Composer
Replay Sessions
Show matching recorded session(s)
Compare with matching recorded session
Compare with multiple recorded sessions

#### Replay Sessions

This will reissue the selected sessions' requests to the server. Then you can view the server's response to ensure the requests were correctly constructed and if the server returns the expected response.

#### Show matching recorded session(s)

This will add the corresponding recorded sessions to the session grid. Use this option to compare multiple recorded and replayed sessions' properties displayed in the grid columns such as response time, payload size or response status.

#### Compare with a recorded session

This option works when one session is selected in the session grid. It opens the Compare Sessions Inspector where you can compare the content of the recorded and replayed session.

**Compare with multiple recorded sessions**

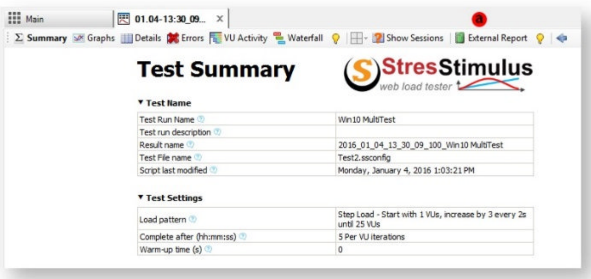
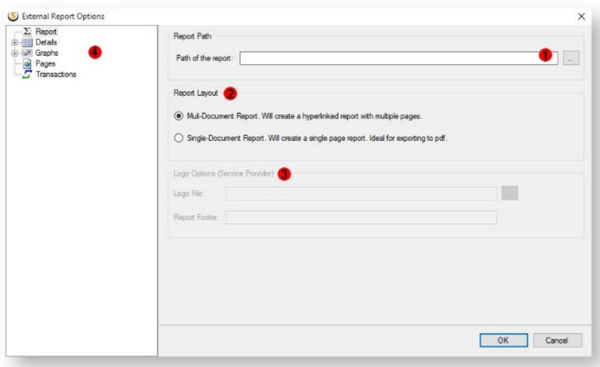
Use this option to simplify comparing multiple sessions. It opens a new tab with the Session Verification tree having a node for every session selected in the session grid.

**Note:** Unlike executing Verify command, that will replay the test case, executing **Compare with multiple recorded sessions** will not replay the test case. It will only display the **Session Verification Tree** for quick comparison of multiple sessions.

Double-click a Session Verification tree node to display the **Compare Sessions Inspector** for a session.

## 7.7 External Reports

Ent SP

| | |
|---|---|
| An External Report presents test results in a portable (HTML) format that can be distributed outside StresStimulus. It includes data from the Test Result tab and subordinate page / transaction result tabs.<br><br>To create an external report, on the test results toolbar, click on **External Report**. A pop-up will appear with options to customize your report. |  |
| There are 2 types of reports:<br><br>• **Multi-Document** - creates multiple hyper-linked hierarchical HTML pages with a report and sub reports. This option provides convenient navigation between various parts of the report. External distribution requires sending multiple HTML and image files. To distribute the multi-document report via email you can archive it (i.e. a zip file) preserving its folder structure. |  |

Main | 01.04-13:30_09... ✕ | a

Σ Summary | Graphs | Details | Errors | VU Activity | Waterfall | Show Sessions | External Report

**Test Summary**

StresStimulus web load tester

▾ **Test Name**

| | |
|---|---|
| Test Run Name | Win10 MultiTest |
| Test run description | |
| Result name | 2016_01_04_13_30_09_100_Win10 MultiTest |
| Test File name | Test2.ssconfig |
| Script last modified | Monday, January 4, 2016 1:03:21 PM |

▾ **Test Settings**

| | |
|---|---|
| Load pattern | Step Load - Start with 1 VUs, increase by 3 every 2s until 25 VUs |
| Complete after (hh:mm:ss) | 5 Per VU iterations |
| Warm-up time (s) | 0 |

External Report Options

Σ Report
Details
Graphs
Pages
Transactions
4

Report Path 
Path of the report: 1 ...

Report Layout 2

[✓ Muli-Document Report. Will create a hyperlinked report with multiple pages.]
Single-Document Report. Will create a single page report. Ideal for exporting to pdf.

Logo Options (Service Provider) 3
Logo File:
Report Footer:

OK Cancel

- **Single-Document** - creates a single page that includes a report and all sub reports. This option is convenient when you need to create a single report printout or import it into a MS Word document. The single-document external report combines the content of the multi-document external report in one page.

Specify the path in which you want to save your report (1) and the type of report which you wish to generate (2).

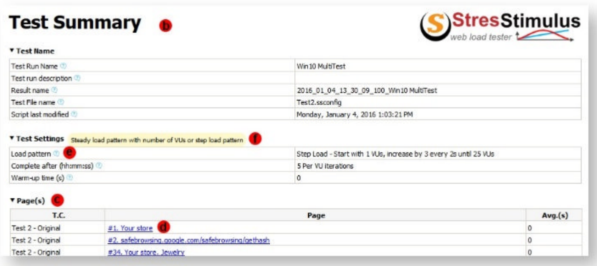
By default, the StresStimulus logo appears on the first page of every test report section. Also, every test report page footer displays the Stimulus Technology copyright statement and website reference. If you have a service-provider license, you have the option to put your custom logo on the report and change the report footer (3).

Then, navigate through the tabs on the left of the pop-up (4). The options available in the tabs are described in Additional Options.

When you finish customizing your report, click OK and your report will be generated and saved to the path which you selected. After it finishes generating, it will open in your default browser automatically.

The multi-document external report contains multiple HTML pages. The default page, **index.htm** displays the Test Summary (b) containing the information from the Test Result Tab. It has a transaction and a page sections (c) containing hyperlinks (d) to the individual page and transaction reports. Individual page and transaction reports displayed in a separate html pages contain the information from the Page and Transaction Result Tab.

All HTML report are self-documented to allow you to easily interpret the test result. Every parameter displays an "?" icon (e) that on mouse-over displays a value description.



**Test Summary**

▾ **Test Name**

| | |
|---|---|
| Test Run Name | Win10 MultiTest |
| Test run description | |
| Result name | 2016_01_04_13_30_09_100_Win10 MultiTest |
| Test File name | Test2.ssconfig |
| Script last modified | Monday, January 4, 2016 1:03:21 PM |

▾ **Test Settings** Steady load pattern with number of VUs or step load pattern

| | |
|---|---|
| Load pattern | Step Load - Start with 1 VUs, increase by 3 every 2s until 25 VUs |
| Complete after (hh:mm:ss) | 5 Per VU iterations |
| Warm-up time (s) | 0 |

▾ **Page(s)**

| T.C. | Page | Avg.(s) |
|---|---|---|
| Test 2 - Original | #1. Your store | 0 |
| Test 2 - Original | #2. safebrowsing.google.com/safebrowsing/gethash | 0 |
| Test 2 - Original | #34. Your store. Jewelry | 0 |

Every grid column on mouse-over displays a tooltip (f) with the column description.

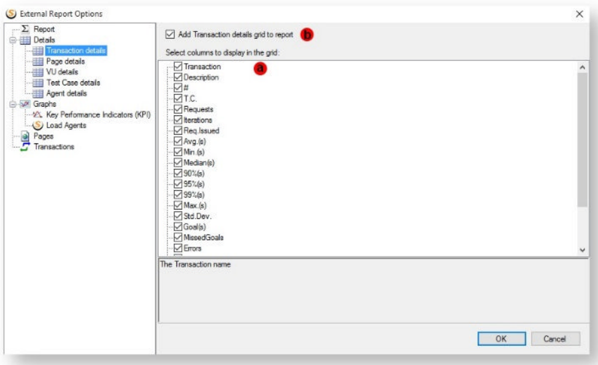
## 7.7.1 Additional Options

Ent SP

The Details tab lets you select which columns you want to be excluded in the grid on your report (a). You can customize the **Transaction details**, **Page details**, **VU details**, **Test Case details**, and **Agent details** grids. To exclude a details grid in the report, uncheck “Add details grid to the report” (b) in all of the sub-page you wish to not include.

The Graphs tab gives you the option to exclude the **Key Performance Indicators (KPI)** and **Load Agent** graphs entirely or select which curves you wish to omit.

All settings in the Report, Details and Graphs tabs are stored in the application and will persist after StresStimulus restart.



☑ Add Transaction details grid to report

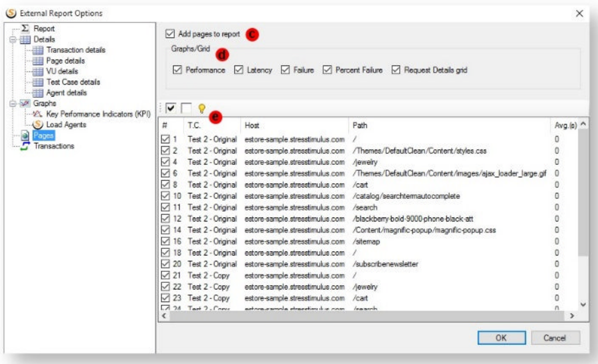
Select columns to display in the grid:

- ☑ Transaction
- ☑ Description
- ☑ #
- ☑ T.C.
- ☑ Requests
- ☑ Iterations
- ☑ Req.Issued
- ☑ Avg.(s)
- ☑ Min.(s)
- ☑ Median(s)
- ☑ 90%(s)
- ☑ 95%(s)
- ☑ 99%(s)
- ☑ Max.(s)
- ☑ Std.Dev.
- ☑ Goal(s)
- ☑ MissedGoals
- ☑ Errors

The Transaction name

In the Pages tab, you can exclude the page summaries form the report entirely (c) or chose which details you want to omit: **Performance**, **Latency**, **Failure**, **Percent Failure**, and **Request Details Grid** (e). Then, you can select which pages you want to include in the report (e). You can also exclude recorded **Screenshot** if the page has them.

Lastly, in the Transactions tab, you can exclude transactions from your report entirely or select which details to omit from the same criteria available in the Pages tab. Then, select which transactions you want to include in the report.



☑ Add pages to report

Graphs/Grid

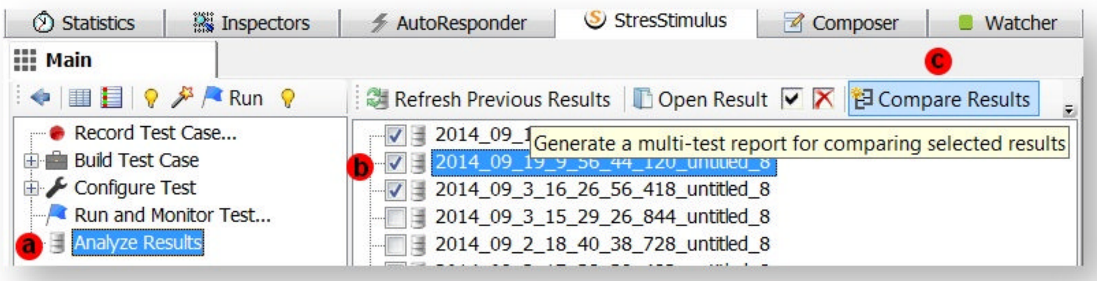
☑ Performance ☑ Latency ☑ Failure ☑ Percent Failure ☑ Request Details grid

| | # | T.C. | Host | Path | Avg.(s) |
|---|---|---|---|---|---|
| ☑ | 1 | Test 2 - Original | estore-sample.stresstimulus.com | / | 0 |
| ☑ | 2 | Test 2 - Original | estore-sample.stresstimulus.com | /Themes/DefaultClean/Content/styles.css | 0 |
| ☑ | 4 | Test 2 - Original | estore-sample.stresstimulus.com | /jewelry | 0 |
| ☑ | 6 | Test 2 - Original | estore-sample.stresstimulus.com | /Themes/DefaultClean/Content/images/ajax_loader_large.gif | 0 |
| ☑ | 8 | Test 2 - Original | estore-sample.stresstimulus.com | /cart | 0 |
| ☑ | 10 | Test 2 - Original | estore-sample.stresstimulus.com | /catalog/searchtermautocomplete | 0 |
| ☑ | 11 | Test 2 - Original | estore-sample.stresstimulus.com | /search | 0 |
| ☑ | 12 | Test 2 - Original | estore-sample.stresstimulus.com | /blackberry-bold-9000-phone-black-att | 0 |
| ☑ | 14 | Test 2 - Original | estore-sample.stresstimulus.com | /Content/magnific-popup/magnific-popup.css | 0 |
| ☑ | 16 | Test 2 - Original | estore-sample.stresstimulus.com | /sitemap | 0 |
| ☑ | 18 | Test 2 - Original | estore-sample.stresstimulus.com | / | 0 |
| ☑ | 20 | Test 2 - Original | estore-sample.stresstimulus.com | /subscribenewsletter | 0 |
| ☑ | 21 | Test 2 - Copy | estore-sample.stresstimulus.com | / | 0 |
| ☑ | 22 | Test 2 - Copy | estore-sample.stresstimulus.com | /jewelry | 0 |
| ☑ | 23 | Test 2 - Copy | estore-sample.stresstimulus.com | /cart | 0 |
| ☑ | 24 | Test 2 - Copy | estore-sample.stresstimulus.com | /search | 0 |

## 7.8 Comparing Tests

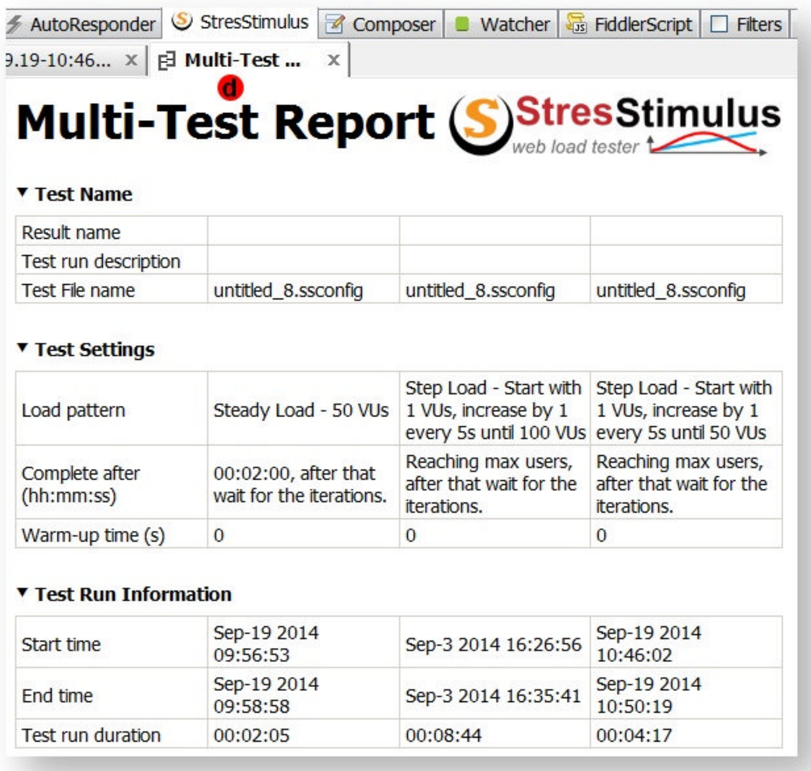
Several test results can be compared side-by-side on a Multi-Test report tab. To create the Multi-Test report, follow these steps

1. From the **Analyze Results** section (a), select the results to compare by checking corresponding boxes (b).



2. Clicking **Compare Tests**(c) on the toolbar.

3. A new **Multi-Test Report** tab (d) will display the report



Multi-Test Report

▼ Test Name

| | | | |
|---|---|---|---|
| Result name | | | |
| Test run description | | | |
| Test File name | untitled_8.ssconfig | untitled_8.ssconfig | untitled_8.ssconfig |

▼ Test Settings

| | | | |
|---|---|---|---|
| Load pattern | Steady Load - 50 VUs | Step Load - Start with 1 VUs, increase by 1 every 5s until 100 VUs | Step Load - Start with 1 VUs, increase by 1 every 5s until 50 VUs |
| Complete after (hh:mm:ss) | 00:02:00, after that wait for the iterations. | Reaching max users, after that wait for the iterations. | Reaching max users, after that wait for the iterations. |
| Warm-up time (s) | 0 | 0 | 0 |

▼ Test Run Information

| | | | |
|---|---|---|---|
| Start time | Sep-19 2014 09:56:53 | Sep-3 2014 16:26:56 | Sep-19 2014 10:46:02 |
| End time | Sep-19 2014 09:58:58 | Sep-3 2014 16:35:41 | Sep-19 2014 10:50:19 |
| Test run duration | 00:02:05 | 00:08:44 | 00:04:17 |

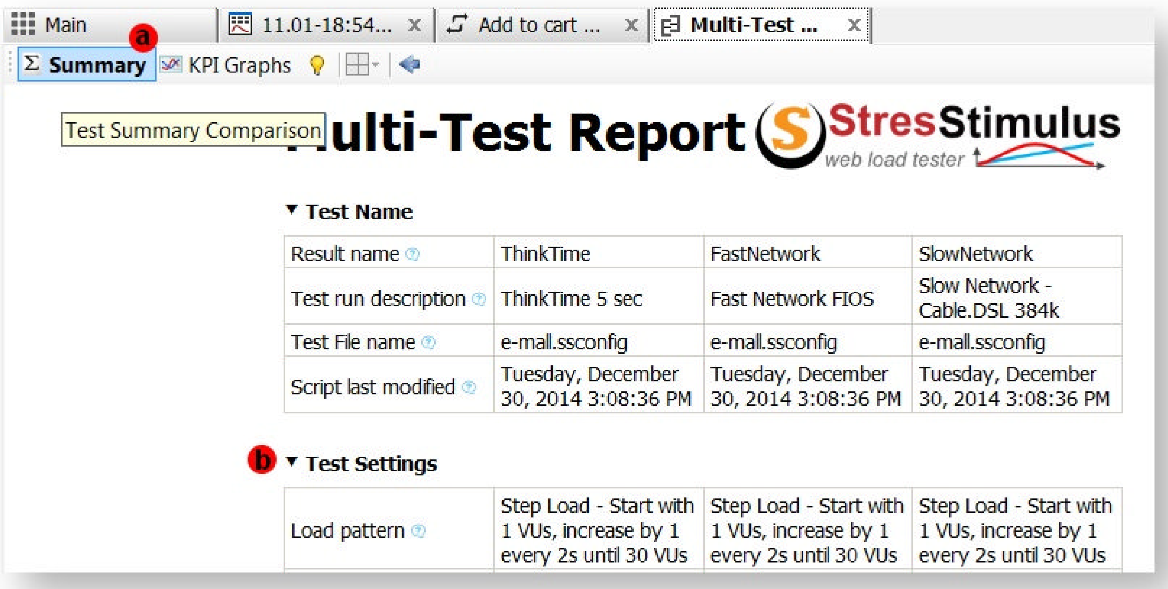
Multi-Test report displays several views described in the following sections.

### 7.8.1 Test Comparison Summary View

This view displays key performance characteristics of the  multiple selected test results side-by-side. It allows to quickly compare main performance metrics of several tests.

To select Summary View, click Summary (a) on the toolbar.

The test summary includes several subsections that can be expanded / collapsed by clicking the triangle icon (b).



▼ Test Name

| | | | |
|---|---|---|---|
| Result name | ThinkTime | FastNetwork | SlowNetwork |
| Test run description | ThinkTime 5 sec | Fast Network FIOS | Slow Network - Cable.DSL 384k |
| Test File name | e-mall.ssconfig | e-mall.ssconfig | e-mall.ssconfig |
| Script last modified | Tuesday, December 30, 2014 3:08:36 PM | Tuesday, December 30, 2014 3:08:36 PM | Tuesday, December 30, 2014 3:08:36 PM |

▼ Test Settings

| | | | |
|---|---|---|---|
| Load pattern | Step Load - Start with 1 VUs, increase by 1 every 2s until 30 VUs | Step Load - Start with 1 VUs, increase by 1 every 2s until 30 VUs | Step Load - Start with 1 VUs, increase by 1 every 2s until 30 VUs |

A list of the Summary View subsections is provided below:

#### 7.8.1.1 Test Name

| | |
|---|---|
| Test Run Name | Optional one-word Test Run Name specified in the Configure Test section. It is used as a suffix following a time-stamp of the test run displayed in the "Analyze Results" section. |
| Test run description | Test description specified in the Configure Test section prior to the test run |
| Result name | The name automatically created for every test run. In SQL CE, a name of a .sdf file. |
| Test File name | The name of the test script file |
| Script last modified | The timestamp of the last script modification |

#### 7.8.1.2 Test Settings

| | |
|---|---|
| Load pattern | Steady load pattern with number of VUs or step load pattern |
| Complete after (hh:mm:ss) | The test primary and secondary completion criteria |
| Warm-up time (s) | The warm-up time actually used in the test |

#### 7.8.1.3 Test Run Information

| | |
|---|---|
| Start time | Test start time |
| End time | Test end time |
| Test run duration | The test duration in seconds. [End Time] - [Start Time] |

#### 7.8.1.4 Overall Result

| | |
|---|---|
| Completion Status | Shows whether the test was Completed or Aborted |
| Max User Load | The maximum number of VUs reached during the test. When Constant Load Pattern is selected, it will be equal to the set number of VUs. When Step Load Pattern is selected, it will be equal to the Max number of VUs or a smaller number that the test duration limit permits to reach. |
| Total sent (KB) | The total amounts of test upload traffic that is equal to the sum of all requests' sizes |
| Total received (KB) | The total amount of test download traffic that is equal to the sum of all responses' sizes |
| KB sent/sec | KB sent per second. Calculated as [Total KB sent] / [Test run duration] |
| KB received/sec | KB received per second. Calculated as [Total KB received] / [Test run duration] |

#### 7.8.1.5 Test Iterations

| | |
|---|---|
| Avg. Iteration time (s) | The average time it took to complete an iteration. Only displayed if all iterations were completed. |
| Iterations started | The combined number of test iterations started by all VUs |
| Iterations passed | The combined number of test iterations completed by all VUs |
| Iterations failed | The combined number of failed (aborted) test iterations |
| Incomplete Iterations | The difference between Iterations started and Iterations passed or failed |

#### 7.8.1.6 Requests

| | |
|---|---|
| Avg. response | Calculated as [Aggregate Session time] / [Total Requests]. |

| | |
|---|---|
| time (s) | |
| Requests/sec | The number of HTTP transactions (sessions) issued by the load test engine. It is calculated as [Total Requests] / [Test run duration]. |
| Number of URLs | The total number of recorded URLs |
| Total requests issued | The combined number of requests issued during the test |
| Request errors | The total number of errors |
| Request timeouts | The total number of timeouts |

#### 7.8.1.7 Transactions

| | |
|---|---|
| Avg. response time (s) | Average transaction response time. |
| Transactions/sec | Average transactions requested per second |
| Number of transactions | The number of transactions in the test |
| Requested Transactions | The total number of Transactions requested in all iterations |
| Transactions with error(s) | The number of transactions with error(s) |
| Requested Transactions with error(s) | The total number of Transactions requested in all iterations with error(s) |
| Transactions with timeout(s) | The number of transactions with with at least one timeout request. See Page Timeout for more details. |
| Requested Transactions with timeout(s) | The total number of Transactions requested in all iterations with timeout(s) |
| Transactions with missed goal(s) | The number of transactions with missed goal |
| Requested Transactions with missed goal(s) | The total number of Transactions requested in all iterations with missed goal(s) |
| Slowest Transaction(s) | Up to five slowest transactions. Click a transaction link to open a transaction report |

#### 7.8.1.8 Pages

| | |
|---|---|
| Avg. response time (s) | Average page response time. |

| | |
|---|---|
| Pages/sec | Average pages requested per second |
| Number of pages | The number of pages in the test |
| Requested Pages | The total number of pages requested in all iterations |
| Pages with error(s) | The number of pages with error(s) |
| Requested Pages with error(s) | The total number of pages requested in all iterations with error(s) |
| Pages with timeout(s) | The number of pages with with at least one timeout request. See Page Timeout for more details. |
| Requested Pages with timeout(s) | The total number of pages requested in all iterations with timeout(s) |
| Pages with missed goal(s) | The number of pages with missed goal |
| Requested Pages with missed goal(s) | The total number of pages requested in all iterations with missed goal(s) |
| Slowest Page(s) | Up to five slowest pages. Click a page link to open a page report |

#### 7.8.1.9 Errors

| | |
|---|---|
| Top Errors | Up to five most occurring errors. Click an error link to retrieve and display all occurrences of this error in the session grid on the left. |

#### 7.8.1.10 Transaction Response Times

This subsection displays side-by-side response time for all transaction across all compared tests. Every transaction is listed on a separate row.

#### 7.8.1.11 Page Response Times

This subsection displays response times for all pages across all compared tests side-by-side. Every pages is listed on a separate row.

### 7.8.2 KPI Graph Comparison View

Ent SP

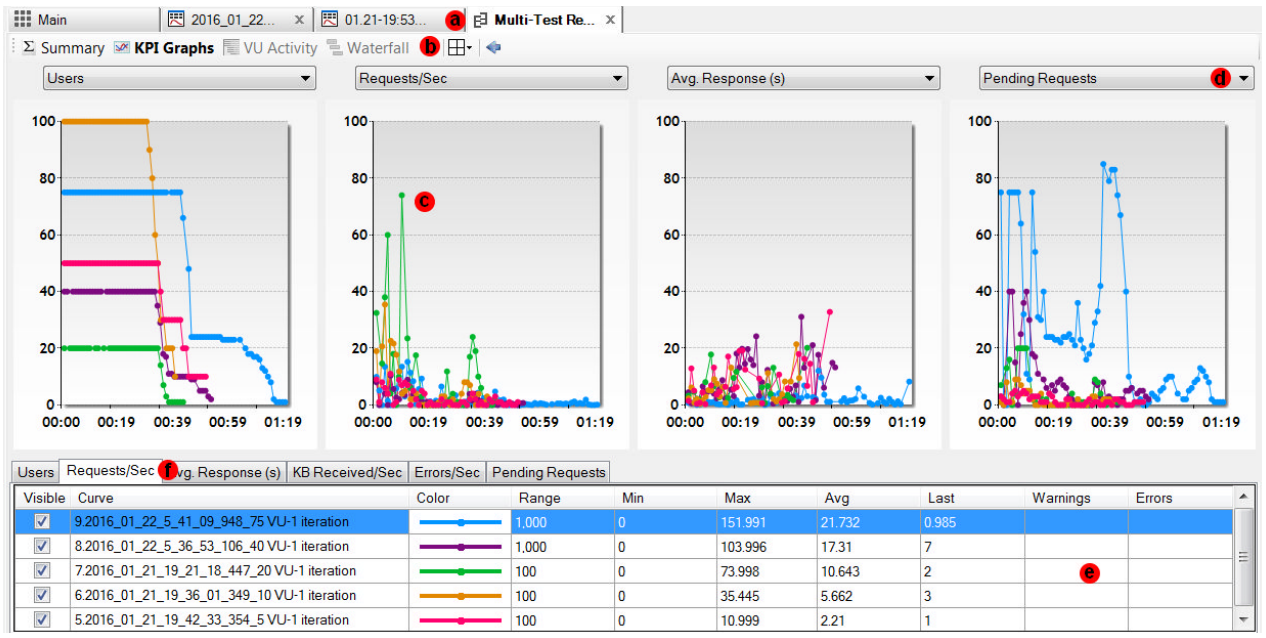
KPI Graph comparison view includes comparison graphs of major performance indicators of the selected test results. It allows to quickly compare performance outcome of every test run.

To select KPI Graph Comparison View, click **KPI Graphs** (a) on the toolbar. One to four panel layouts can be selected by clicking the **Layout** drop-down (b).

Each graph (c) will display one of six KPI counters for every selected result.

- **Users** - The number of instantiated and active VUs, iterating trough their respective test cases. Some VUs can complete the test earlier than others and become inactive. VUs which completed all their iterations before the test end are excluded from the active user count. If the test is configured to complete only after all VUs complete their iterations, then the User graph will show gradual declining the number of VUs at the end of the test.
- **Req/Sec** - The number of requests being sent per second.
- **Avg. Response(s)** - The average response time of the sent requests. Every datapoint on the response time graph reflects the average value during the checkpoint intervals. If **no responses were received** between two datapoints, the response time cannot be determined, so such datapoints are skipped to reflect an accurate response time curve.
- **KB Received/Sec** - The number of kilobytes currently received per second.
- **Errors/Sec** - The number of errors currently received per second.
- **Pending Requests** - The number of currently pending requests. Pending requests are those that are issued but the response is not yet received by StresStimulus. Generally, the greater number of pending requests indicates slower server response. This parameter can be used to gauge performance change in several tests runs.

To select which graph to display in a panel, click the drop-down (d) above it.



| Visible | Curve | Color | Range | Min | Max | Avg | Last | Warnings | Errors |
|---|---|---|---|---|---|---|---|---|---|
| ✓ | 9.2016_01_22_5_41_09_948_75 VU-1 iteration | 🟦 | 1,000 | 0 | 151.991 | 21.732 | 0.985 | | |
| ✓ | 8.2016_01_22_5_36_53_106_40 VU-1 iteration | 🟪 | 1,000 | 0 | 103.996 | 17.31 | 7 | | |
| ✓ | 7.2016_01_21_19_21_18_447_20 VU-1 iteration | 🟩 | 100 | 0 | 73.998 | 10.643 | 2 | | |
| ✓ | 6.2016_01_21_19_36_01_349_10 VU-1 iteration | 🟧 | 100 | 0 | 35.445 | 5.662 | 3 | | |
| ✓ | 5.2016_01_21_19_42_33_354_5 VU-1 iteration | 🟥 | 100 | 0 | 10.999 | 2.21 | 1 | | |

Each graph depicts one curve for every compared test run. For example, if you select 3 tests results, each graph will display three curves, as shown on the figure.

**Info**: Each graph, supports the full range of graph and curve commands described in the Graph Commands section.

Curve grid (e) located under the graph panels shows the following additional information about each curve

| | |
|---|---|
| Visible | Whether the curve is currently visible. Click the checkbox to hide/unhide the curve. |
| Indicator | The name of the curve. |
| Color | The curve color and point shape. |
| Range | The current value range. The y-value of each point corresponds to a percent of the range. |
| Min | Minimum value of the curve - datapoint aggregation (Sortable) |
| Max | Maximum value of the curve - datapoint aggregation (Sortable) |
| Avg | Average value of the curve - datapoint aggregation (Sortable) |
| Last | The last value of the curve - datapoint aggregation (Sortable) |
| Warnings | Number of threshold violation warnings (Sortable) Ent SP |

| | |
|---|---|
| Errors | Number of threshold violation errors (Sortable) Ent SP |
| Missed Goals | Number of missed goals (Sortable) Ent SP |

To view other other metrics, navigate through the tabs (f).

# 8 ADVANCED TOPICS

**Search the Advanced Topics section**

## 8.1 Distributed Testing

Ent SP

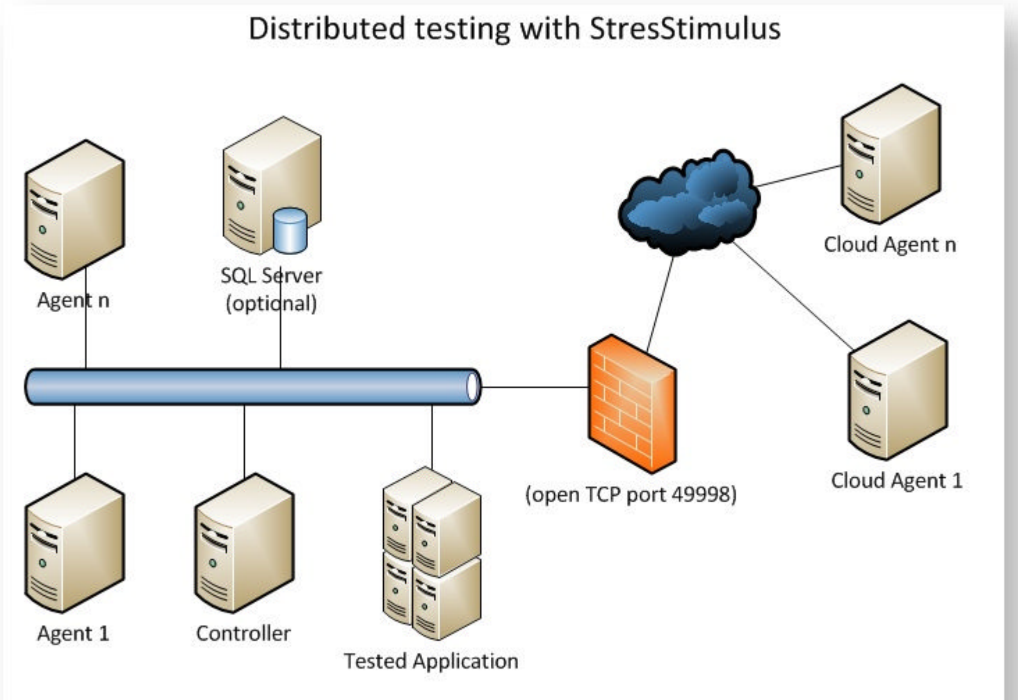
Distributed testing allows to spread load generators across multiple concurrent computers, called **Load Agents** (agents), operated under the management of a Controller. Distributed testing has the following benefits:

- The ability to emulate a very large number of VUs (up to a million and above), unrestricted by hardware capacity of a single machine.
- The ability to emulate users disseminated across a network and geographically.

### 8.1.1 Controller and Agents

A controller orchestrates a test concurrently running from multiple agents and consolidates performance metrics. An agent is an unattended computer with a StresStimulus instance which has agent mode turned on. The agent emulates virtual users in distributed tests, orchestrated by the controller.

A single StresStimulus software product performs three roles: Test Designer, Controller, and Load Agent. During the test design stage, the Designer is used for designing test scenarios (test script) and configuring test parameters. The Controller executes the test. It distributes the amount of load between Load Agents, coordinates their operation, collects and aggregates real-time performance metrics and generates final reports. The Load Agents operate concurrently under the orchestration of the Controller. Load Agent can be installed on a physical or virtual machine as well as on an Amazon EC2 or Microsoft Azure instance. A diagram below shows an example of distributed testing schema.

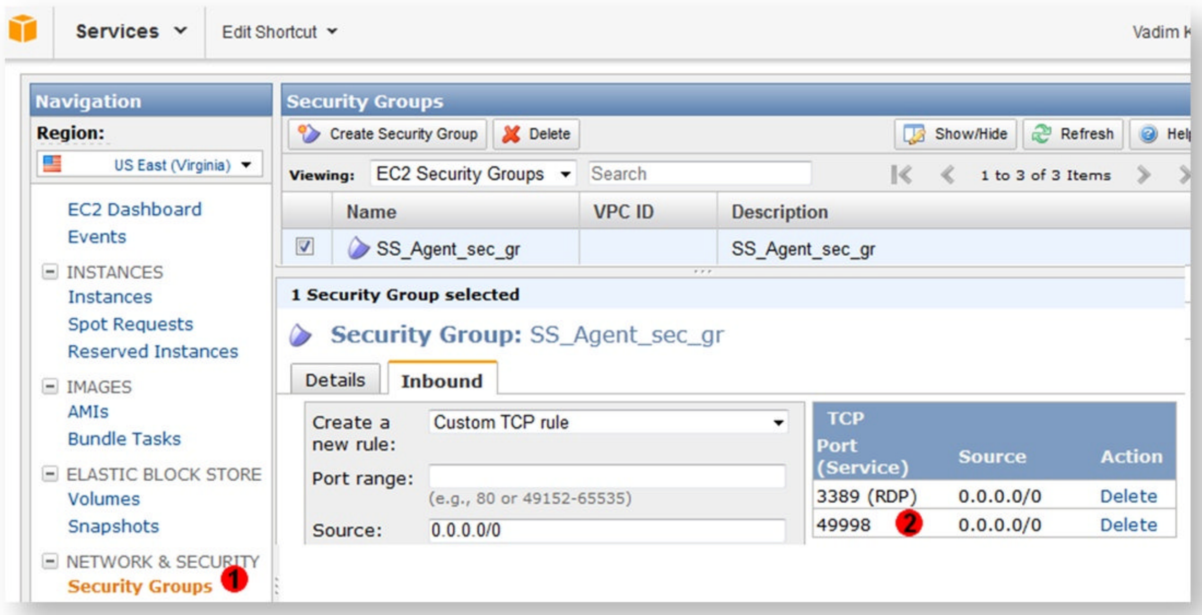


1. All hosts can be installed locally, on the cloud or both.

2. By Default, local SQL Servers CE are used on the Controller and Agents as a data storage. SQL Server is optional.

### 8.1.1.1 Installing Agents on Amazon EC2

Install StresStimulus on Amazon EC2 Windows instances. Make sure that port 49998 is open on Amazon virtual firewall. Fig below shows the example how it can be configured.



| | Name | VPC ID | Description |
|---|---|---|---|
| ☑ | SS_Agent_sec_gr | | SS_Agent_sec_gr |

**Security Group:** SS_Agent_sec_gr

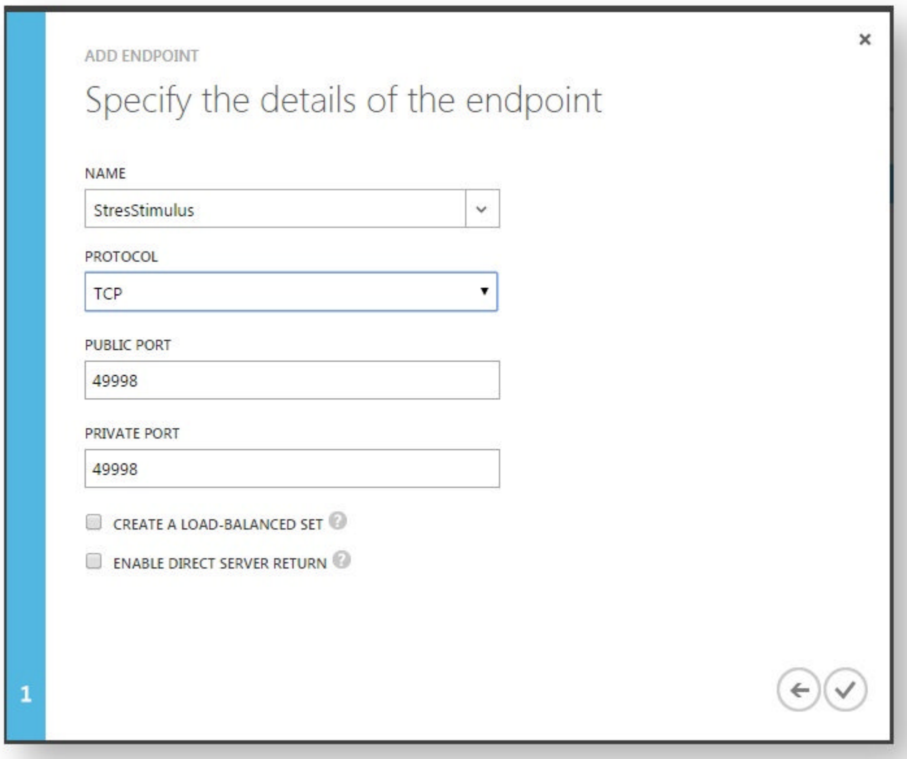
| | |
|---|---|
| Create a new rule: | Custom TCP rule |
| Port range: | (e.g., 80 or 49152-65535) |
| Source: | 0.0.0.0/0 |

| TCP Port (Service) | Source | Action |
|---|---|---|
| 3389 (RDP) | 0.0.0.0/0 | Delete |
| 49998 | 0.0.0.0/0 | Delete |

1. From the Amazon web console, create a new security group, SS_Agent_sec_gr.
2. Create a custom TCP rule to keep inbound port 49998 (or another port if you changed the default port while configuring the agent) open.
3. Use this security group when creating an EC2 instance.

#### 8.1.1.2 Installing Agents on Microsoft Azure

Install StresStimulus on Microsoft Azure Virtual Machines. Add an Endpoint the virtual machine to port 49998. Fig below shows the example how it can be configured.



ADD ENDPOINT

Specify the details of the endpoint

NAME: StresStimulus

PROTOCOL: TCP

PUBLIC PORT: 49998

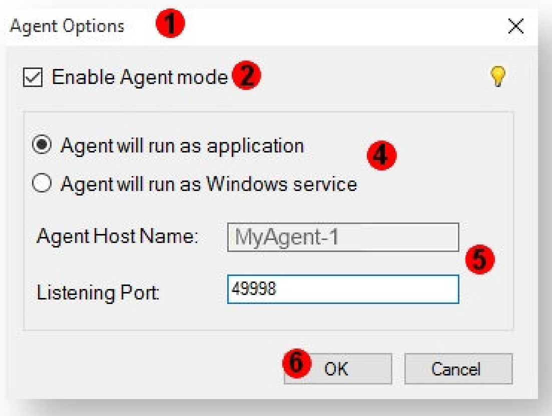
PRIVATE PORT: 49998

☐ CREATE A LOAD-BALANCED SET

☐ ENABLE DIRECT SERVER RETURN

1. From the Microsoft Azure dashboard, select your virtual machine and select the Endpoints option.
2. Create a custom Endpoint for TCP port 49998 or another port if you changed the default port while configuring the agent.

### 8.1.1.3 Configuring Agent

| | |
|---|---|
| After installation, StresStimulus operates as a Designer and Controller. In order to instantiate a Load Agent or a computer, enable **Agent Mode** after installing StresStimulus.<br><br>**Note:** To avoid controller-agent compatibility issues, the same StresStimulus version must be installed on all machines participated in a test.<br><br>To enable Agent mode: |  Agent Options<br>☑ Enable Agent mode<br>◉ Agent will run as application<br>○ Agent will run as Windows service<br>Agent Host Name: MyAgent-1<br>Listening Port: 49998 |

| | |
|---|---|
| 1. Bring up Agent Option dialog:<br>•<br>○<br>▪ **Standalone version**: in Main Menu -> **Options** -> **Agent Options**<br>▪ **add-on version:** in Main Menu -> **StresStimulus** -> **Agent Options**<br>2. Check **Enable Agent Mode**.<br>3. A popup will appear asking if you are sure you wish to create an Agent. Click OK to confirm.<br>4. Select one of two available options:<br>•<br>○<br>▪ **Agent will run as application.** The agent will run a StresStimulus application or Fiddler add-on. Before starting the test, StresStimulus must be manually launched. During the test run, graphs and run-time monitor will be visible on the Agent, so while all interactive UI controls will be disabled, you can visually monitor how the agent participates in the distributed test.<br>▪ **Agent will run as a Windows service.** The agent will run inside a Windows Service. By default, StresStimulus Agent service will start automatically. In this mode, the agent has no UI. This agent mode uses less hardware resources and has the capacity to emulate about 20% more VUs. | |

5. Then give the agent host a name and specify a listening port. The default port is 49998. The port can't be changed for agents that run as a service and will always be the default 49998.

6. Click **OK** when finished. The StresStimulus status bar will display **StresStimulus Load Agent**.

**Note**: Unlike the controller that requires a serial number, Agents do not require activation. With the Enterprise / SP Edition license, you can install as many load agents as you need.

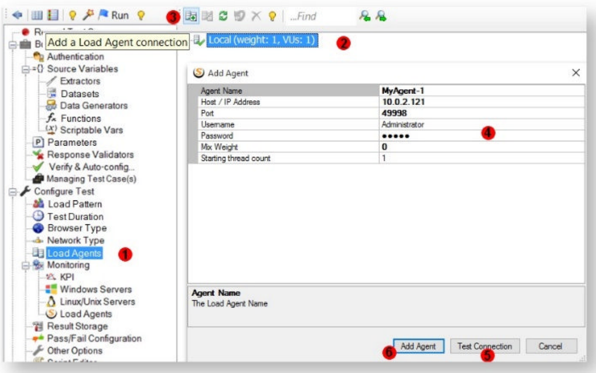
Once the agent is created, go to the controller and attached the agent.

## 8.1.2 Attaching Agents to Controller

1. On the controller, in the **Configure Test -> Load Agents** section of the **Workflow Tree**, the grid displays agents attached the controller.
2. The first agent called Local is the controller by itself.
3. To attach a new agent, on the grid's toolbar, click **Add**.
4. Enter required connection properties: Agent Name, Host name or IP address without "//", Port, Username, and Password
5. Click **Test Connection**. If a pop-up message indicates any connectivity issues, correct them and try again.
6. If the message indicates that the connection is successfully verified, click **Add Agent**.
7. A new agent will appear in the list.

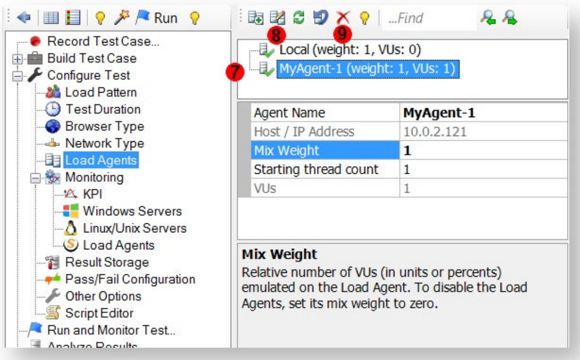
To edit the existing connection, selected it on the grid and click **Edit Selected Load Agent Connection** on the toolbar. Alternatively, you can right-click and select **Edit**, or double-click on the agent's name.

To delete the existing connection, selected it



| | |
|---|---|
| Agent Name | MyAgent-1 |
| Host / IP Address | 10.0.2.121 |
| Port | 49998 |
| Username | Administrator |
| Password | ●●●●● |
| Mix Weight | 0 |
| Starting thread count | 1 |

**Agent Name**
The Load Agent Name



| | |
|---|---|
| Agent Name | MyAgent-1 |
| Host / IP Address | 10.0.2.121 |
| Mix Weight | 1 |
| Starting thread count | 1 |
| VUs | 1 |

**Mix Weight**
Relative number of VUs (in units or percents) emulated on the Load Agent. To disable the Load Agents, set its mix weight to zero.

in the grid and click **Delete Selected Load Agent Connection** on the toolbar. Alternatively, you can right-click and select **Delete**.

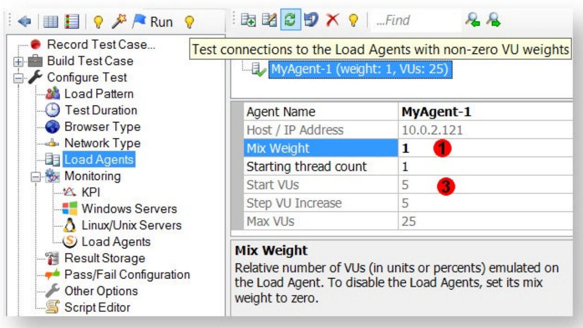
**Note:** To communicate with the controller, the Agent has a listener on TCP port 49998 (or another port if you changed the default port while configuring the agent). If the Controller displays a communication error, make sure that StresStimulus is running on the remote computer in agent mode, and that TCP port 49998 is opened on firewalls between the controller and agents.

## 8.1.3 Configuring Load Distribution

The number of VUs emulated by a Load Agent is proportionate to its VU weight.

1. Enter the desired **VU weight** for each agent. To disable a Load Agent, set its **VU weight** to zero.
2. To verify how the controller is going to distribute VUs, click **Test connections to the Load Agents with non-zero VU weights**. This operation will re-test the connections as well.
3. The expected number of VUs will be displayed in the property grid.

**Note:** If you set the Local agent's mix weight to 0 then it will not create any VUs, and will act solely as a Controller.



Test connections to the Load Agents with non-zero VU weights

MyAgent-1 (weight: 1, VUs: 25)

| | |
|---|---|
| Agent Name | MyAgent-1 |
| Host / IP Address | 10.0.2.121 |
| Mix Weight | 1 |
| Starting thread count | 1 |
| Start VUs | 5 |
| Step VU Increase | 5 |
| Max VUs | 25 |

**Mix Weight**
Relative number of VUs (in units or percents) emulated on the Load Agent. To disable the Load Agents, set its mix weight to zero.

## 8.1.4 Distributed Test Results

**KPI, Page, Transaction and Test Case Graphs**
During the test run, every agent collects data points and send them to the controller. If the agent runs in the application mode, it displays graphs with performance metrics collected on the agent. Since the agents are most likely operated on unattended computers, agent graphs and progress monitors can be viewed via Remote Desktop sessions.

The controller communicates with the agents, collects, and displays graphs with aggregated real-time performance metrics.

**Note: The** number of data points on the agent graphs can be slightly higher than the number of data points on the controller graphs. This is a normal behavior caused by limitation of the network bandwidth.

**Load Agent Performance Counters**

Every agent in application mode displays graphs of its own local performance counters such as CPU and memory utilization . By default, the Controller does not collect such Agent Performance Counters. If you wish to display agent computer performance counters on the controller, make sure to add them to the controller when configuring the test,

**Test Run Report**

After test completion, the controller generates a consolidated summary and detailed reports that include information aggregated from all agents. The test run report is stored on the controller. It can be reopened from the **Test Results** section.

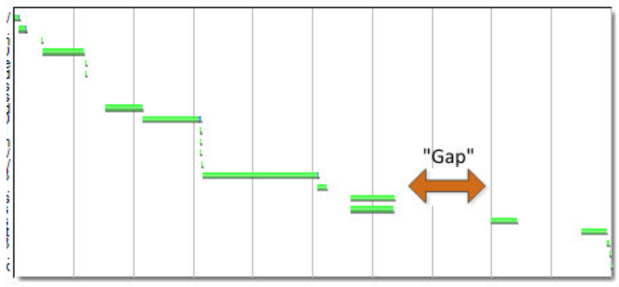
**Test Log**

If the SQL Server CE is used as a repository, every agent will create its own database file locally for every test run. HTTP session information is storedin the test log on a machine where such sessions were initiated. You can query the test log as described in the Querying Test Log section. However, the controller can not query a remote agent's database. Therefore it cannot query a portion of the test log generated by the VUs,instantiated on agents. The **VUDetails** report showson which test machine every VU was instantiated.

If the SQL Server is used as a repository, a single database, shared between all agents/controller will be used. In this case you can query the entire test log.

## 8.1.5 Load Generator Performance

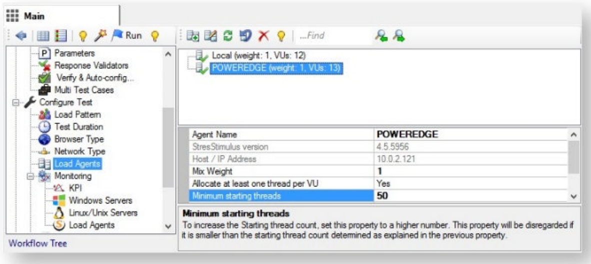
An essential element of the controller or agent is the load generator that handles request /response processing. It needs a sufficient number of .NET threads to timely send and process HTTP messages.

| | |
|---|---|
| The system creates the necessary number of threads automatically, depending on the test complexity, the number of VUs and the test machine's capacity. The actual number of threads allocated to StresStimulus process is controlled by the operating system, and depends on the load generator's needs to issue requests.<br><br>By default, before the test begins, StresStimulus allocates the number of .NET threads threads equal to the maximum number of VUs expected in this test.<br><br>During test execution, one VU may consume more than one thread to handle parallel requests. If StresStimulus needs more resources, the OS will gradually increase threads to the sufficient level. On some systems, however, OS fails to ramp-up threads fast enough. This may result in delays in issuing requests after receiving a response at some moments during the test execution. You can check if such delays exist in your system by analyzing a page or transaction waterfall chart.<br><br>If you see a gap between the moment of receiving a response and issuing a subsequent request, your system may need more threads. To address the situation, you can manually adjust the initial number of threads, which is a property of a load agent. |  |

**Note**: The gap between a response and a subsequent request can be caused by different reasons. For example, StresStimulus intentionally delays subsequent request to comply with the emulated browser characteristics. Each browser type has a limited number of connections-per-host and connections-per-proxy. When the number of connection exceeds this limit, StresStimulus will pause requests to realistically mimic a physical browser behavior.

If you need to ramp up threads faster, which may be necessary when running large-scale tests on a high performance hardware, set the **Minimum starting threads** property to the level 120%-150% of the number of virtual users. This should boost performance of your load generator. After that, rerun your test to check if such increase was sufficient. You might need to try several different numbers until your load generator will receives sufficient resources. This property is configured for each load agent individually. Note that setting a too high number of threads may leave insufficient resources for the OS and can slow down your computer.

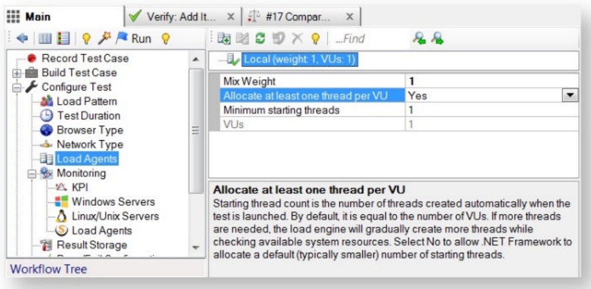
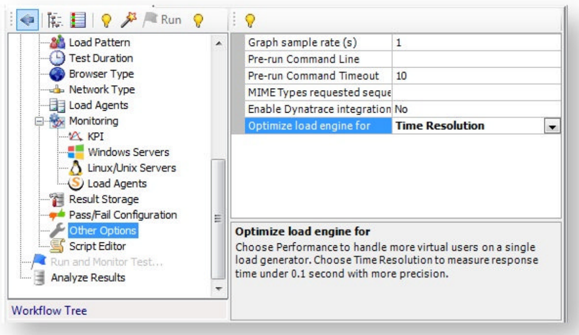
**Info:** One VU is not equal to one thread. Threads available to the process (thread pool) can be used by different VUs at different times. Each VU can issue multiple concurrent requests (connections), depending on your test case and the connection limit of the selected browser. For example, IE9 can open up to 6 connections.



Local (weight: 1, VUs: 12)
POWEREDGE (weight: 1, VUs: 13)

| | |
|---|---|
| Agent Name | POWEREDGE |
| StresStimulus version | 4.5.5956 |
| Host / IP Address | 10.0.2.121 |
| Mix Weight | 1 |
| Allocate at least one thread per VU | Yes |
| Minimum starting threads | 50 |

**Minimum starting threads**
To increase the Starting thread count, set this property to a higher number. This property will be disregarded if it is smaller than the starting thread count determined as explained in the previous property.

| | |
|---|---|
| **Note:** In the previous versions, thread allocation algorithm relied on .NET Framework's default algorithm of allocating threads. As a result, the starting thread count was typically smaller than the number of virtual users. For backward compatibility the option to use the legacy thread allocation algorithm exist. To use it, set property **Allocate at least one thread per VU** to **No**. |   Main<br>Verify: Add It...<br>#17 Compar...<br>Run<br>...Find<br>Record Test Case<br>Build Test Case<br>Configure Test<br>Load Pattern<br>Test Duration<br>Browser Type<br>Network Type<br>Load Agents<br>Monitoring<br>KPI<br>Windows Servers<br>Linux/Unix Servers<br>Load Agents<br>Result Storage<br>Workflow Tree Local (weight 1, VUs: 1)<br>Mix Weight: 1<br>Allocate at least one thread per VU: Yes<br>Minimum starting threads: 1<br>VUs: 1<br>**Allocate at least one thread per VU**<br>Starting thread count is the number of threads created automatically when the test is launched. By default, it is equal to the number of VUs. If more threads are needed, the load engine will gradually create more threads while checking available system resources. Select No to allow .NET Framework to allocate a default (typically smaller) number of starting threads. |
| Typically the load engine is optimized for performance to handle the more virtual users on a single load generator. However, if your test has many resource requests that have a small expected response time want to change the load engine optimization for "Time Resolution". This will enable measuring response times under 0.1 second with more precision, however will have a performance impact with more virtual users. |   Run<br>Load Pattern<br>Test Duration<br>Browser Type<br>Network Type<br>Load Agents<br>Monitoring<br>KPI<br>Windows Servers<br>Linux/Unix Servers<br>Load Agents<br>Result Storage<br>Pass/Fail Configuration<br>Other Options<br>Script Editor<br>Run and Monitor Test...<br>Analyze Results<br>Workflow Tree Graph sample rate (s): 1<br>Pre-run Command Line:<br>Pre-run Command Timeout: 10<br>MIME Types requested seque<br>Enable Dynatrace integration: No<br>Optimize load engine for: **Time Resolution**<br>**Optimize load engine for**<br>Choose Performance to handle more virtual users on a single load generator. Choose Time Resolution to measure response time under 0.1 second with more precision. |

## 8.2 Automation

Ent SP

StresStimulus supports a command line interface that can be used for various automation purposes such as scheduling test runs using Windows Task Scheduler, or interfacing with other systems such as continuous integration systems, etc.

The Standalone and Fiddler add-on versions have different command line formats.

**Note:** When StresStimulus is started from a command line, some of the interactive features, like test wizard, are disabled to allow StresStimulus to function in unattended mode,

### 8.2.1 Command line interface for Standalone version

The standalone version uses the following command line:

```
StresStimulus.Application.exe /SS <File.ssconfig> [/ssclose]
```

where:

`<File.ssconfig>` is a StresStimulus test configuration file. Specify a full path or the default directory `%My Documents%/Fiddler/StresStimulus` will be used.

`/ssclose` (optional) is a switch to close the application after the test is complete. The command will not return until the application is closed. Use this flag to run multiple contiguous tests.

**Exit code:** The process will return the exit code 1 if the test failed the test quality criteria.

### 8.2.2 Command line interface for Add-on version

Use the following command to launch Fiddler, start the test, close Fiddler and return the control to the batch processor:

```
LaunchFiddler.exe /ss <file.ssconfig> [/wait | /ssclose]
```

where:

`<file.ssconfig>` is a StresStimulus test configuration file. Specify a full path or the default directory `%My Documents%/Fiddler/StresStimulus` will be used.

`/wait` (optional) is a switch to wait until the test is complete, close Fiddler and after that return control to the batch processor. Use this flag to run multiple sequential tests. if the wait flack is not used, the control to the batch processor will be returned immediately after launching Fiddler.

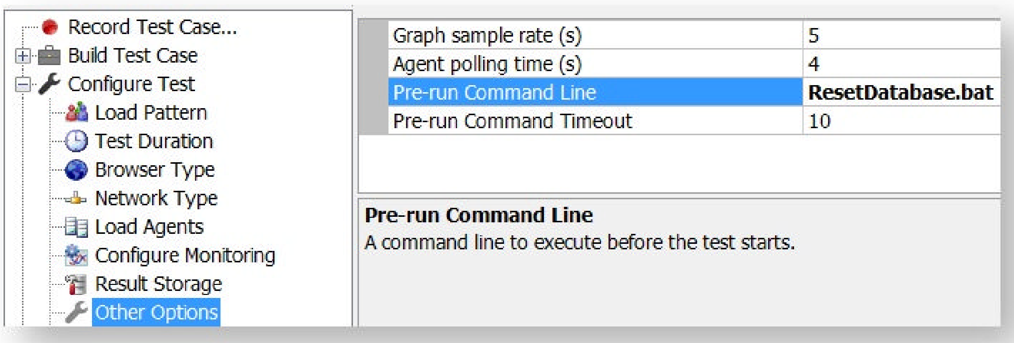
`/ssclose` is obsolete, has the same functionality as /wait and is used for backward compatibility only.

**Exit code:** The process will return the exit code 1 if the test failed the test quality criteria.

### 8.2.3 Pre-run command line

You can execute a custom procedure before running a test case. One example of when this feature can be helpful is when specific test data can be inserted into the database only once. Before running a user can run a script that deletes the database records inserted by the previous test run.

To do so, in the Other Options section enter the command to launch the script in the **Pre-run Command Line** property. You can also specify a script timeout in the **Pre-run Command Timeout**



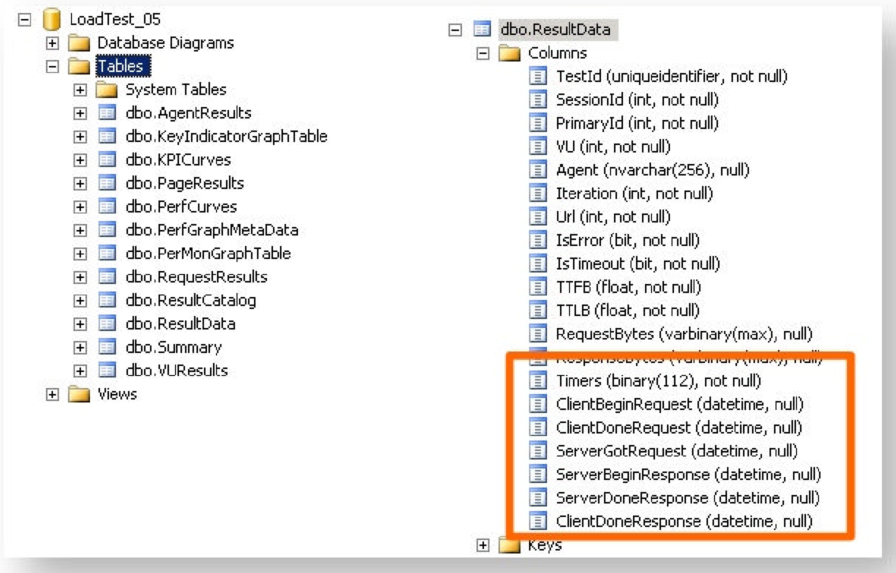
|  |  |
|---|---|
| Graph sample rate (s) | 5 |
| Agent polling time (s) | 4 |
| Pre-run Command Line | ResetDatabase.bat |
| Pre-run Command Timeout | 10 |

**Pre-run Command Line**
A command line to execute before the test starts.

## 8.3 Querying Test Repository

Ent SP

When SQL Server is set as test repository you can query test log directly and generate custom reports using external analytical tools. The test repository table structure is shown below.



```
LoadTest_05
  Database Diagrams
  Tables
    System Tables
    dbo.AgentResults
    dbo.KeyIndicatorGraphTable
    dbo.KPICurves
    dbo.PageResults
    dbo.PerfCurves
    dbo.PerfGraphMetaData
    dbo.PerMonGraphTable
    dbo.RequestResults
    dbo.ResultCatalog
    dbo.ResultData
    dbo.Summary
    dbo.VUResults
  Views

dbo.ResultData
  Columns
    TestId (uniqueidentifier, not null)
    SessionId (int, not null)
    PrimaryId (int, not null)
    VU (int, not null)
    Agent (nvarchar(256), null)
    Iteration (int, not null)
    Url (int, not null)
    IsError (bit, not null)
    IsTimeout (bit, not null)
    TTFB (float, not null)
    TTLB (float, not null)
    RequestBytes (varbinary(max), null)
    ResponseBytes (varbinary(max), null)
    Timers (binary(112), not null)
    ClientBeginRequest (datetime, null)
    ClientDoneRequest (datetime, null)
    ServerGotRequest (datetime, null)
    ServerBeginResponse (datetime, null)
    ServerDoneResponse (datetime, null)
    ClientDoneResponse (datetime, null)
  Keys
```

`ResultData` is the main table that stores all session information. Session's time-based performance metrics is stored in Fiddler session timer format described here. The following six timers used:

- `ClientBeginRequest` - Time at which this HTTP request began. May be much later than ClientConnected due to client connection reuse.

- `ClientDoneRequest` - Exact time that the client browser finished sending the HTTP request to StresStimulus or Fiddler.

- `ServerGotRequest` - Exact time that StresStimulus or Fiddler finished (re)sending the HTTP request to the server.

- `ServerBeginResponse` - Exact time that StresStimulus or Fiddler got the first bytes of the server's HTTP response.

- `ServerDoneResponse` - Exact time that StresStimulus or Fiddler got the last bytes of the server's HTTP response.

- `ClientDoneResponse`- Exact time that StresStimulus or Fiddler finished transmitting the HTTP response to the client browser.

**API for querying session content**

The Query Log allows to display session content in the session grid. However, the capacity of the session grid to display a very large number of records is limited. For such situations an option exists to query session content directly from the test repository.

Session data is stored in the `ResultData` table in the SQL Server database. However requests and response information is stored UTF8 binary encoded form.

to decode session content to clear text use build-in function `Utf8ToNVarChar()` as shown below:

```
SELECT dbo.Utf8ToNVarChar(RequestBytes) AS RequestChars,
dbo.Utf8ToNVarChar(ResponseBytes) AS ResponseChars FROM ResultData
```

## 8.4 Extensibility

Ent SP

Extensibility allows to programmatically extend StresStimulus functionality and behavior. You can write your own custom logic that can access internal test object model (TOM), generate test parameters, modify HTTP requests and responses and create other customizations.

Extensibility allows to:

- create scriptable variables that can be used to create custom request parameter of any complexity;

- create external components that can be used to modify StresStimulus event handling and change requests and responses based on the test context information.

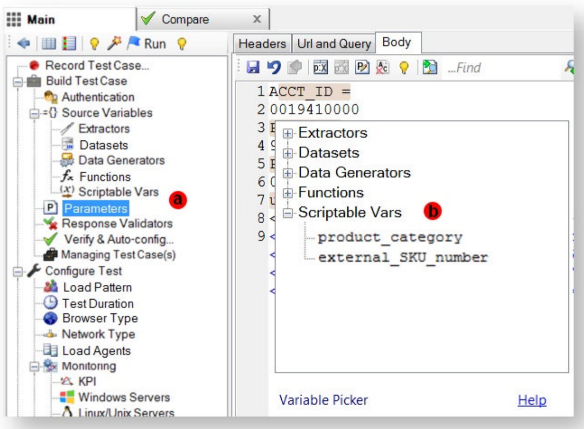
### 8.4.1 Scriptable Variables

Ent SP

Scriptable variables can be created by writing .NET code. Once created, they are are displayed in the **Source Variable** section of the workflow tree (a) and appear in the Variable Picker (b) alongside with other variables.

Scriptable variables are used similarly to other source variables such as extractors, datasets, functions and data generators. Once the scriptable variable is implemented, it can be used to parameterize requests the same way as other variables.

StresStimulus evaluates the scriptable variable before sending the request that has the parameter where the variable is used ("related request"). At this moment, your custom code is executed, the return value is assigned to the variable and is used to parameterize the request.



There are 2 modes of evaluation the scriptable variable: on request and on iteration:

- **On request**: the evaluation event takes place before issuing every related request, in every iteration for every VU. Use this evaluation mode only when the variable value is expected to change on every occurrence of your code execution.

- **On iteration**: the evaluation event takes place once per VU-iteration before issuing the first related request. After that, the scriptable variable value is stored in StresStimulus and is re-used within the same iteration in any subsequent related requests, if they exist. Use this mode if the value of the scriptable variable must stay the same for the duration of the VU-iteration.

There are two type of scriptable variables that are differs by the method they are created:

- **Internal scriptable variables** are created in StresStimulus. Their benefit is simplicity as they do not require any additional development tools.

- **External scriptable variables** are created by developing external .NET DLLs using Microsoft Visual Studio. This allows to

| | |
|---|---|
| develop more sophisticated functionality and behavior using external libraries and powerful development and debugging Visual Studio environment. | |

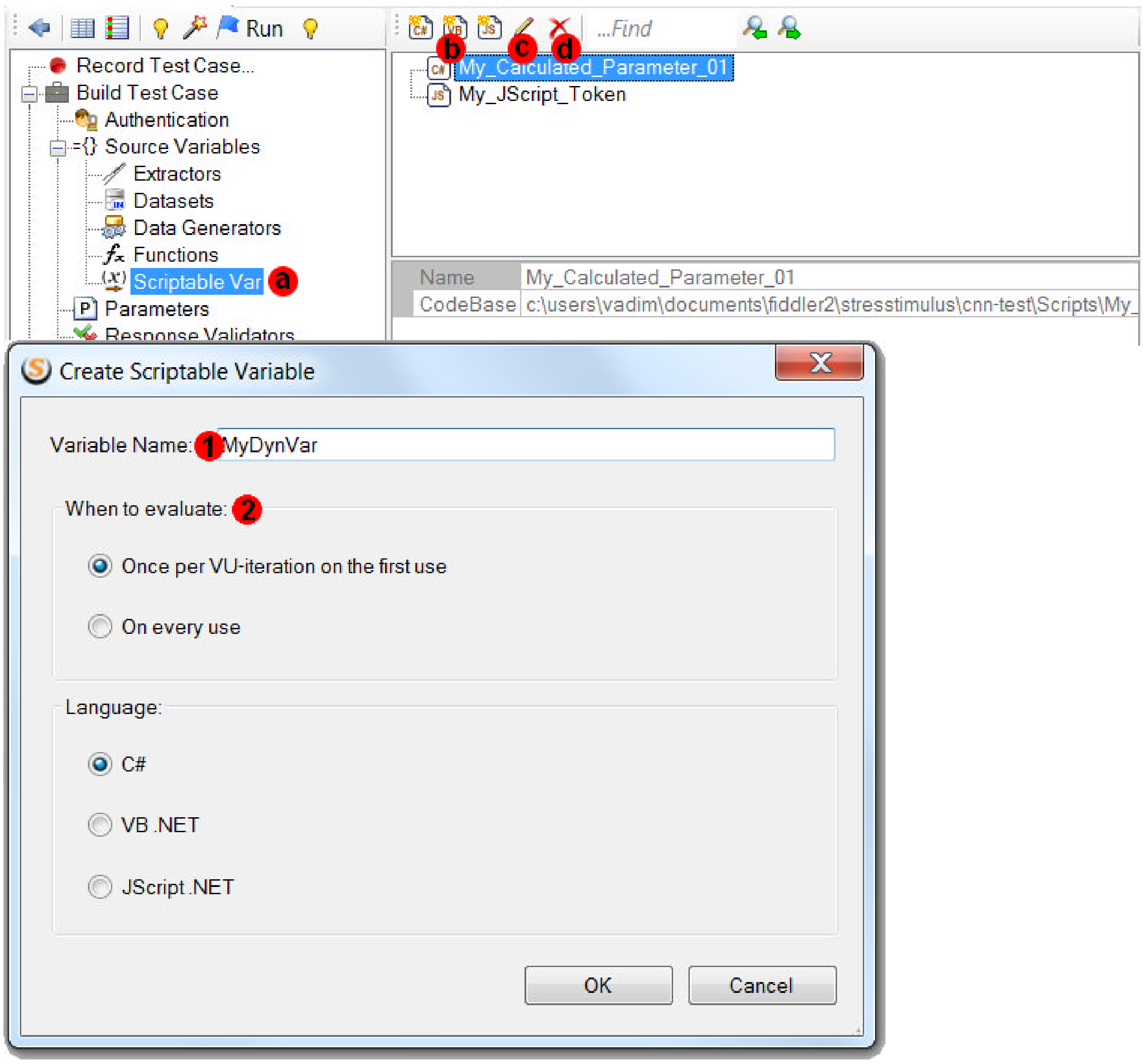
### 8.4.2 Internal Scriptable Variables

Ent SP

To create a scriptable variable inside StresStimulus, navigate to the Scriptable Var tab (a). From there, you can create your own variable. Select one of three languages you wish to use: C#, VB.NET or JScript.NET (b).

Give your variable a name (1) and chose when you want to evaluate it (2). StresStimulus will either evaluate the variable once per VU-iteration on the first use or on every use. Then the script editor will open and you can write your code, compile it and save it.

You can edit or delete your variable later on by going back to the Scriptable Var tab, selecting the variable you wish to modify and then clicking either **Edit** (c) or **Delete** (d).

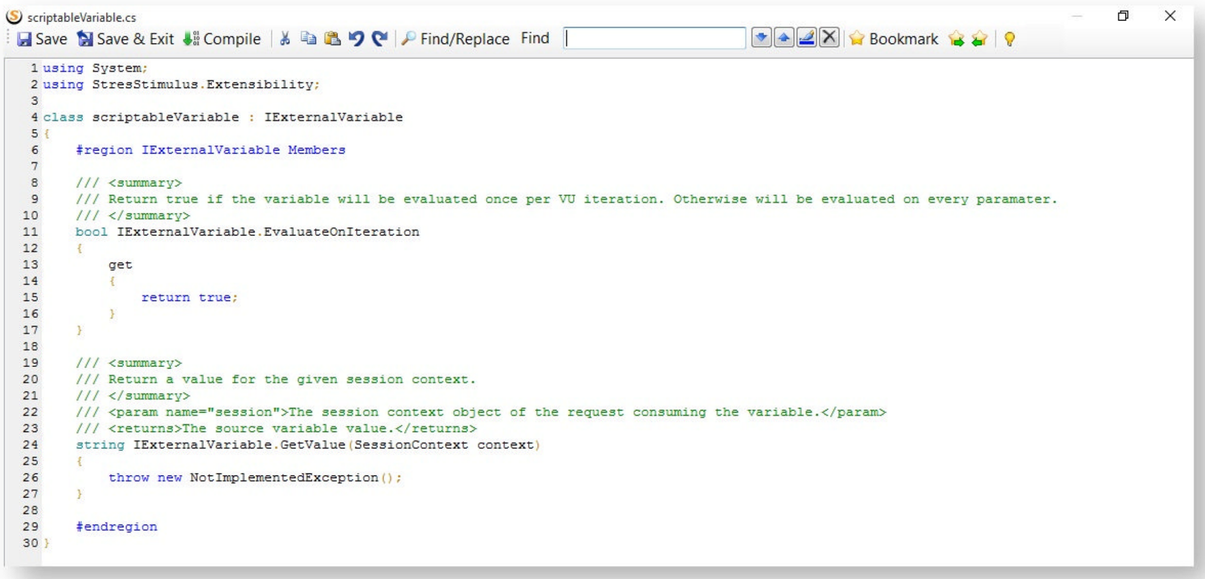


| | |
|---|---|
| Name | My_Calculated_Parameter_01 |
| CodeBase | c:\users\vadim\documents\fiddler2\stresstimulus\cnn-test\Scripts\My_ |

Variable Name: MyDynVar

When to evaluate:
[✓ Once per VU-iteration on the first use]
On every use

Language:
[✓ C#]
VB .NET
JScript .NET



```
using System;
using StresStimulus.Extensibility;

class scriptableVariable : IExternalVariable
{
    #region IExternalVariable Members

    /// <summary>
    /// Return true if the variable will be evaluated once per VU iteration. Otherwise will be evaluated on every paramater.
    /// </summary>
    bool IExternalVariable.EvaluateOnIteration
    {
        get
        {
            return true;
        }
    }

    /// <summary>
    /// Return a value for the given session context.
    /// </summary>
    /// <param name="session">The session context object of the request consuming the variable.</param>
    /// <returns>The source variable value.</returns>
    string IExternalVariable.GetValue(SessionContext context)
    {
        throw new NotImplementedException();
    }

    #endregion
}
```

## 8.4.3 Programming Scriptable Variables

Ent SP

In order to create a scriptable variable, implement the `StresStimulus.Extensibility.IExternalVariable` interface. It has the following interface definition:

**Tabelle 1 IExternalVariable Members**

```
    /// <summary>
     /// An external variable.
     /// </summary>
     public interface IExternalVariable
     {
         /// <summary>
         /// Return true if the variable will be evaluated once per VU
iteration. Otherwise will be evaluated on every parameter.
         /// </summary>
         bool EvaluateOnIteration { get; }

         /// <summary>
         /// Return a value for the given session context.
         /// </summary>
         /// <param name="session">The session context object of the
request consuming the variable.</param>
         /// <returns>The source variable value</returns>
         string GetValue(SessionContext context);
```

```
        }
```

This interface has one property and one method:

- The **GetValue()** method should have all the logic of the scriptable variable and must return the parameter value. If no value (or null value) is returned, then StresStimulus will use the recorded value when sending the request. This method receives a **SessionContext object** object that describes the iteration and VU context of which this scriptable variable is called from.
- The **EvaluateOnIteration** property should return **true** if this scriptable variable have the same value throughout each VU's iteration. This property should only return **false** if the scriptable variable is used in several requests in the test case and the value is expected to change from request to request.

#### 8.4.3.1 Practical considerations

One instance of the class is created for every parameter in a test case. Here is an example: in a test case a scriptable variable `MyScriptVar` is used to create 3 parameters in 3 different requests. In this case, 3 instances of the **MyScriptVar** class are created. They are instantiated at the moments when the scriptable variable is bound to the parameters. This happens when you open the test, or on-demand when you create a parameter in the test designer. There are several consequences of this design:

1. Every VU and every iteration during the test will use the same instance of the class associated with the corresponding parameter (3 instances per `MyScriptVar` variable in this case).
2. Class level variables persist throughout StresStimulus application session. If you restart the test without closing StresStimulus, the class level variables will be not initialized, which makes them less predictable. In most cases it makes sense to use local method variables.
3. During test runtime, `GetValue()` is called when evaluating the corresponding parameters. For example, if you put your test case in a loop which is executed twice, then every VU on every iteration will trigger six `GetValue()` calls.
4. `GetValue()` is called asynchronously for every VU on every iteration. So if you have 100 VUs each running 5 iterations the total number of calls will be 100 * 5 * 6 = 3000. This method is not thread-safe, so use locks when using shared resources.
5. The `GetValue()` call in Stimulus is located within a try/catch block. If your `GetValue()` implementation throws an exception, it will be ignored and the recorded value will be used in StresStimulus to parameterize the request.
6. `GetValue()` execution time can have performance impact on the StresStimulus load generator hit rate. Design `GetValue()` for performance.

#### 8.4.3.2 Example

The following is a simple example creates a parameter that returns a unique time-based id. For more complex examples see the subsequent sections.

**Tabelle 2 Simple Example**

```
using System;
using StresStimulus.Extensibility;
class Ticks : IExternalVariable
{
    public Ticks()
    {
          //Do not edit this section.
    }

    /// <summary>
    /// Return true if the variable will be evaluated once per VU
iteration. Otherwise will be evaluated on every paramater.
    /// </summary>
    bool IExternalVariable.EvaluateOnIteration
    {
          get
          {
                return true;
          }
    }
    /// <summary>
    /// Return a value for the given session context.
    /// </summary>
    /// <param name="session">The session context object of the
request consuming the variable.</param>
    /// <returns>The source variable value.</returns>
    string IExternalVariable.GetValue(SessionContext context)
    {
          return DateTime.Now.Ticks.ToString();
    }
}
```

#### 8.4.3.3 SessionContext object

The SessionContext object describes the iteration and VU context of which a scriptable variable is called from. This object can be accesses as an argument **GetValue(SessionContext)** method when implementing **IExternalVariable** interface. It also has access to helper object Extractor manipulation to further expand functionality.

**Tabelle 3 SessionContext Members**

```
    /// <summary>
     /// The runtime session iteration and VU context.
     /// </summary>
     public class SessionContext
     {
         /// <summary>
         /// The session id.
         /// </summary>
         public int RecordedSession { get; }
         /// <summary>
         /// The test case name.
         /// </summary>
         public string RecordedTestCase { get; }
         /// <summary>
         /// The current iteration number.
         /// </summary>
         public int IterationNumber { get; }
         /// <summary>
         /// The current request number in iteration.
         /// </summary>
         public int RequestNumber { get; }
         /// <summary>
         /// The VU number.
         /// </summary>
         public int VUNumber { get; }
         /// <summary>
         /// The extractor runtime object for the VU.
         /// </summary>
         public ExtractorRuntime ExtractorRuntime { get; }
         /// <summary>
         /// Returns the data source with the given name.
         /// </summary>
         /// <param name="name">The name of the data set.</param>
         /// <returns>The DataTable object that represents the data
 set. Returns null if data source does not exist.</returns>
         public DataTable GetDatasource(string name);
     }
```

## Example

The following example implements the Big IP persistence cookie encoding for F5 load-balancer. In this environment the target server is specified in the request cookie header, so the example parameterizes the cookie for every VU and makes sure the distribution is even between the servers. There are a few steps performed here:

1. The IP Addresses and port of the servers are stored in an array called EndPointList.
2. Use the SessionContext.VUNumber to determine what VU is currently asking for the cookie. In order assign every VU to the next endpoint perform a vu number modulo number of enpoints operation (see below).
3. Encode the IP Address to the BIGipServer specification.

For more details on BigIPServer cookie this click here.

**Tabelle 4 Big IP Server Example**

```
class BIGipServer : IExternalVariable
{
    public BIGipServer()
    {
        //Do not edit this section.
    }

    /// <summary>
    /// Return true if the variable will be evaluated once per VU
iteration. Otherwise will be evaluated on every parameter.
    /// </summary>
    bool IExternalVariable.EvaluateOnIteration
    {
        get
        {
            return true;
        }
    }

   /// <summary>
   /// List of server endpoints behind the load balancer. Add as
many as necessary.
   /// </summary>
   static string[] EndPointLists = new string[] {
         "10.238.108.162:443",
         "10.1.1.100:8080",
   };
    /// <summary>
    /// Return a value for the given session context.
    /// </summary>
    /// <param name="session">The session context object of the
request consuming the variable.</param>
    /// <returns>The source variable value.</returns>
    string IExternalVariable.GetValue(SessionContext context)
    {
        int index = context.VUNumber % EndPointLists.Length;
//Perform the modulo operation.

         return EncodeEndpoint(EndPointLists[index]); //Return the
encoded value.
    }

   /// <summary>
   /// Returns encoded representation of an endpoint.
   /// </summary>
   /// <param name="endpoint">The endpoint to encode
{ip_address}:{port}.</param>
   /// <returns>The encoded endpoint string.</returns>
   static string EncodeEndpoint(string endpoint)
   {
         string[] parts = endpoint.Split(':');
         string sIp = parts[0], sPort = parts[1];
         string[] octets = sIp.Split('.');
         long eAddress =
               (byte.Parse(octets[0]) |
               byte.Parse(octets[1]) << 8 |
               byte.Parse(octets[2]) << 16 |
               byte.Parse(octets[3]) << 24) & 0xFFFFFFFFL;
         short port = short.Parse(sPort);
         int ePort =
               ((port & 0xFF) << 8) |
               ((port & 0xFF00) >> 8);

         return string.Format("{0}.{1}.0000", eAddress, ePort);
   }
}
```

### 8.4.3.4 Extractor manipulation

A scriptable variable can access the runtime extractor values using the **ExtractorRuntime** object available from the **SessionContext** object. This is useful in situations where it necessary to emulate javascript actions that the browser would otherwise perform. See example below.

**Tabelle 5 ExtractorRuntime**

```
/// <summary>
/// A virtual user's extractor runtime.
/// </summary>
public class ExtractorRuntime
{
    /// <summary>
    /// Returns the extractor's current value for the VU.
    /// </summary>
    /// <param name="extractor">The name of the extractor.</param>
    /// <returns>The value of the extractor. Returns an empty string
if the extractor is not found.</returns>
    public string GetExtractorValue(string extractor);

    /// <summary>
    /// Set the extractor's value for the current VU iteration.
    /// </summary>
    /// <param name="extractor">The name of the extractor.</param>
    /// <param name="value">The value of the extractor to
set/</param>
    public void SetExtractorValue(string extractor, string value);
}
```

## Changing extractor values

In some cases it may be necessary to change the extractor's value for the current iteration. You can use the **SetExtractorValue()** method to do so.

## Example

The following example returns a value of a login key that is a combination of a server defined prefix and the current epoch timestamp. In the browser environment, the server sends a key prefix and then some client javascript function would take the prefix and add the current epoch timestamp to it to be used in subsequent requests. This behavior can be replicated in StresStimulus using the following code. For this example assume that we created an extractor named *KeyPrefix*.

**Tabelle 6 Extractor example**

```
class ExtractorKeyVariable : IExternalVariable
{
    public ExtractorKeyVariable()
    {
        //Do not edit this section.
    }

    /// <summary>
    /// Return true if the variable will be evaluated once per VU
iteration. Otherwise will be evaluated on every parameter.
    /// </summary>
    bool IExternalVariable.EvaluateOnIteration
    {
        get
        {
            return true;
        }
    }

    /// <summary>
    /// Return a combination of server defined prefix and current
timestamp.
    /// </summary>
    /// <param name="session">The session context object of the
request consuming the variable.</param>
    /// <returns>The source variable value.</returns>
    string IExternalVariable.GetValue(SessionContext context)
    {
        string prefix =
context.ExtractorRuntime.GetExtractorValue("KeyPrefix");
        if (!string.IsNullOrEmpty(prefix)) //check this to make sure
the server did in fact send the prefix.
        {
            return string.Format("{0}_{1}", prefix,
ToUnixTime(DateTime.Now));
        }
        return null;
    }

    /// <summary>
    /// Returns given DateTime to epoch time format.
    /// </summary>
    /// <param name="date"></param>
    /// <returns></returns>
    static long ToUnixTime(DateTime date)
    {
        var epoch = new DateTime(1970, 1, 1, 0, 0, 0,
DateTimeKind.Utc);
        return Convert.ToInt64((date - epoch).TotalMilliseconds);
    }
}
```

**Note**

In the above example, the extractor *KeyPrefix* must be extracted from a response that comes before the request where *ExtractorKeyVariable* scriptable variable is used.

### 8.4.3.5 Dataset manipulation

A scriptable variable can access the datasets that are part test in order to perform data manipulation. Datasets data can be accessed from the **SessionContext.GetDatasource(string)** method that returns a System.Data.DataTable object. This is useful in situations where it necessary to emulate javascript actions that the browser would otherwise perform. See example below.

#### Example

The following example returns base64 encoded username and password that come from a dataset. In this example we'll implement VU binding, so each VU use its own credentials from the dataset. There are a few steps performed here:

1. Get the dataset called *Credentials* which has columns *Username* and *Password*. Note: This dataset must exist in the test.
2. Retrieve the row number that corresponds to the VU.
3. Return the base64 encoded value of a combination of username and password column for the determined row.

**Tabelle 7 Base64 encode example**

```
public class Base64EncodeCredentials : IExternalVariable
{
    public Base64EncodeCredentials()
    {
        //Do not edit this section.
    }

    /// <summary>
    /// Return true if the variable will be evaluated once per VU
iteration. Otherwise will be evaluated on every parameter.
    /// </summary>
    bool IExternalVariable.EvaluateOnIteration
    {
        get
        {
            return true;
        }
    }

    /// <summary>
    /// Return a combination of server defined prefix and current
timestamp.
    /// </summary>
    /// <param name="session">The session context object of the
request consuming the variable.</param>
    /// <returns>The source variable value.</returns>
    string IExternalVariable.GetValue(SessionContext context)
    {
        DataTable dt = context.GetDatasource("Credentials");
        if (dt != null)
        {
            int row = context.VUNumber; //Use this option when number
of VUs <= number of rows in UserData table
            //int row = context.VUNumber % dt.Rows.Count; //Use this
option when number of VUs > number of rows in UserData table
            if (row < dt.Rows.Count)
                return
base64Encode(dt.Rows[row]["Username"].ToString(),
dt.Rows[row]["Password"].ToString());
        }
        return null;
    }

    /// <summary>
    /// Base64 encode username and password.
    /// </summary>
    /// <param name="username">Username string</param>
    /// <param name="password">Password string</param>
    /// <returns>Base64 string of username : password.</returns>
    string base64Encode(string username, string password)
    {
        string userpass = username + ":" + password;
        return
Convert.ToBase64String(Encoding.ASCII.GetBytes(userpass));
    }
}
```

## 8.4.4 External Scriptable Variables

Ent SP

To create an external scriptable variable, in Visual studio create a new class library project. Then add a reference to StresStimulus extensibility API located in the `StresStimulus.Extensibility` namespace using the following assembly:

- **In standalone version**, use namespace is located in the `StresStimulus.Core` assembly
- **In Fiddler add-on version**, use namespace is located in `StresStimulus.Core.Fiddler` assembly

If you wish to use your custom External Scriptable Variables with both StresStimulus versions, create/compile two versions of the extensions using the same codebase, but referencing two different StresStimulus assemblies.

Your assembly should be stored into the `Bin` subfolder in the test folder. For example, for the test `MyTest.ssconfig`, copy the assembly dll to `MyTest\bin folder`.

Programming of the External Scriptable Variables is similar to programming the Internal Scriptable Variables.

### 8.4.4.1 .NET versions

StresStimulus is built on .NET 3.5, however it  runs seamlessly on machines that have .NET 3.5 and/or .NET 4 Framework. However, this becomes important when building external StresStimulus extensions. Particularly, extensions built in .NET 4 will not load on .NET 3.5. If you built an extension in .NET 4 and have .NET 3.5 and .NET 4 installed on your machine then do the following:

### 8.4.4.2 Standalone Version:

In StresStimulus install directory, find file StresStimulus.Application.exe.config.  Open and remove the line <supportedRuntime version="v2.0.50727"/> and save it. After that, StresStimulus will launch under .NET 4.0.  You would need to make this change again after every StresStimulus reinstall / update.

### 8.4.4.3 Fiddler addon:

Use Fiddler4 (not Fiddler2). StresStimulus will run inside Fiddler process under .NET 4.0. This workaround does not required you to make any changes after StresStimulus reinstall / update. StresStimulus add-on version has all features of the standalone version plus Fiddler integration.

### 8.4.4.4 Distributed Testing:

If using external variables in a distributed testing environment, it's important to consider what version of StresStimulus is running on the the controller and agents being used.

- If the controller is running **standalone version**, the agents must be running **standalone version** or as a **Windows service**.

- If the controller is running **Fiddler add-on version**, the agents must be running **Fiddler add-on version**.

**External Components**

The information provided in this section also pertains to external components.

## 8.4.5 External Components

Ent SP

You can further extend StresStimulus run-time by creating an external component. An external component traps StresStimulus run-time events to allow extending its functionality.

To create an external component implement the `StresStimulus.Extensibility.IExternalComponent` interface. It has the following interface definition:

**Tabelle 8 IExternalComponent Members**

```
/// <summary>
/// Interface to handle test events. Must have a no argument
contructor.
/// </summary>
public interface IExternalComponent
{
    /// <summary>
    /// Fired when test started.
    /// </summary>
    void OnTestStart();
    /// <summary>
    /// Fired when test ends.
    /// </summary>
    void OnTestEnd();
    /// <summary>
    /// Fired before a request is sent.
    /// </summary>
    /// <param name="session">The session object</param>
    void OnBeforeRequest(RuntimeSession session);
    /// <summary>
    /// Fired after a response is received.
    /// </summary>
    /// <param name="session">The session object</param>
    void OnAfterResponse(RuntimeSession session);
}
```

- **OnTestStart()**: Use this method initialize any test specific objects that will be used during the test, or to run any pre-test script (such as cleaning previously created database records).
- **OnTestEnd()**: Use this method to clean up any objects used during the test or run any post-test script.
- **OnBeforeRequest()**: This is called before a request is sent. It receives an instance of **RuntimeSession** object that can be used to customize the request headers and body. This method is called after all parameters have been applied to the request.
- **OnAfterResponse()**: This is called after a response is received. It receives an instance of **RuntimeSession** object to read the response headers and body for further automation.

The **RuntimeSession** class has has access to the run-time session object that has access to request and response data. It contains the following properties and methods:

**Tabelle 9 RuntimeSession Members**

```
 /// <summary>
/// A replayed runtime session object.
/// </summary>
public class RuntimeSession
{
    /// <summary>
    /// The recorded session id.
    /// </summary>
    public int RecordedSession { get; }
    /// <summary>
    /// The recorded test case name.
    /// </summary>
    public string RecordedTestCase { get; }
    /// <summary>
    /// The current iteration number.
    /// </summary>
    public int IterationNumber { get; }
    /// <summary>
    /// The current request number in iteration.
    /// </summary>
    public int RequestNumber { get; }
    /// <summary>
    /// The VU number.
    /// </summary>
    public int VUNumber { get; }
    /// <summary>
    /// The extractor runtime object for the VU.
    /// </summary>
    public ExtractorRuntime ExtractorRuntime { get; }
   /// <summary>
    /// Returns the data source with the given name.
    /// </summary>
    /// <param name="name">The name of the data set.</param>
    /// <returns>The DataTable object that represents the data set.
Returns null if data source does not exist.</returns>
    public DataTable GetDatasource(string name);
    /// <summary>
    /// The url.
    /// </summary>
    public Uri Url { get; }
    /// <summary>
    /// Get request headers.
    /// </summary>
    /// <returns>An enumerable list of request headers.</returns>
    public IEnumerable<NameValuePair> GetRequestHeaders();
    /// <summary>
    /// Gets a string representation of the request body.
    /// </summary>
    /// <returns>The string representation of the request
body.</returns>
    public string GetRequestBody();
    /// <summary>
    /// Gets the raw request body.
    /// </summary>
    /// <returns>The byte[] representation of the request
body.</returns>
    public byte[] GetRawRequestBody();
    /// <summary>
    /// Change or add a request header. Set the value to null to
remove the header.
    /// </summary>
    /// <param name="name">The name of the request header.</param>
    /// <param name="value">The value of the request header.</param>
    public void SetRequestHeader(string name, string value);
    /// <summary>
    /// Change the request body.
    /// </summary>
    /// <param name="body">The byte[] of the new response
body.</param>
    public void SetRequestBody(byte[] body);
    /// <summary>
    /// Returns the response code.
    /// </summary>
    public int ResponseCode;
    /// <summary>
    /// Get response headers.
    /// </summary>
    /// <returns>An enumerable list of response headers.</returns>
    public IEnumerable<NameValuePair> GetResponseHeaders();
    /// <summary>
    /// Gets a string representation of the response body.
    /// </summary>
    /// <returns>The string representation of the response
body.</returns>
    public string GetResponseBody();
    /// <summary>
    /// Gets the raw request body.
    /// </summary>
    /// <returns>Returns the byte[] representation of the response
body.</returns>
    public byte[] GetRawResponseBody();
}
/// <summary>
/// Name/value pair representation.
/// </summary>
public class NameValuePair
{
    /// <summary>
    /// The name.
    /// </summary>
    public string Name { get; }
    /// <summary>
    /// The value.
    /// </summary>
    public string Value { get; }
}
```

See the subsequent section for examples of external components.

### 8.4.5.1 Save Response Example

The following examples demonstrates how to create an external component that saves requested excel files from the server. Here are the steps:

1. Implement the **IExternalComponent.OnAfterResponse(RuntimeSession)** method.
2. Determine the URL of the response that contains the excel file to save.

3. Create a unique name of the location or file name of the file to be saved.
4. Save the response body to disk.

**Tabelle 10 Save Response Example**

```
public class SaveExcelResponse : IExternalComponent
{
    void IExternalComponent.OnTestStart()
    {
        //--- Not necessary.
    }
    void IExternalComponent.OnTestEnd()
    {
        //--- Not necessary.
    }
    void IExternalComponent.OnBeforeRequest(RuntimeSession session)
    {
        //--- Not necessary.
    }
    void IExternalComponent.OnAfterResponse(RuntimeSession session)
    {
        if (session.Url.LocalPath == "/file.xlsx" &&
            session.ResponseCode == 200)//find the proper response
and make sure it's a success response.
        {
            byte[] response = session.GetRawResponseBody();
            if (response.Length > 0)
            {
                string fileName = string.Format("file_{0}_{1}.xlsx",
session.VUNumber, session.IterationNumber); //give unique name
                string path =
System.IO.Path.Combine("C:\\DownloadedFiles", fileName);
                using (System.IO.FileStream fs =
System.IO.File.Open(path, System.IO.FileMode.CreateNew))
                {
                    fs.Write(response, 0, response.Length); //write
file
                }
            }
        }
    }
}
```