



Бесплатная электронная книга

учусь

# Apache JMeter

Free unaffiliated eBook created from  
**Stack Overflow contributors.**

#jmeter

	1
<b>1: Apache JMeter</b>	<b>2</b>
	2
	2
Examples	3
	3
Apache JMeter	4
<b>2: Apache JMeter Correlations</b>	<b>7</b>
	7
Examples	7
Apache JMeter	7
XPath Extractor JMeter	12
CSS / JQuery JMeter	16
JSON	19
«SmartJMX» BlazeMeter	22
<b>3: Apache JMeter:</b>	<b>26</b>
	26
Examples	26
JMeter	26
- JMeter	28
	31
HTTPS	33
Chrome BlazeMeter	35
BadBoy	38
<b>4: Apache JMeter</b>	<b>40</b>
	40
Examples	40
	40
	47
Parameterized Controller	57
	63

# Около

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [apache-jmeter](#)

It is an unofficial and free Apache JMeter ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Apache JMeter.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to [info@zzzprojects.com](mailto:info@zzzprojects.com)

# глава 1: Начало работы с Apache JMeter

## замечания

JMeter - это инструмент тестирования нагрузки, используемый для [тестирования производительности](#). Тестер производительности может записывать действия в веб-браузере или вручную создавать *сценарий*, который затем может запускаться сотнями или тысячами пользователей.

JMeter может использоваться для создания невероятно динамичных пользователей и сценариев с использованием различных элементов. Например, [CSV Data Set Config](#) может использоваться для указания набора пользователей для входа в веб-приложение. [Regular Expression Extractor](#) или [CSS / JQuery Extractor](#) можно использовать для сохранения идентификаторов сеанса, которые будут использоваться в будущих запросах. [JSR223 PreProcessor](#) подключенный к [Groovy](#), может использоваться для создания динамических уникальных данных для каждого пользователя, который будет отправлен как часть тела POST .

## Версии

Версия	Версия Java	Дата выхода
<a href="#">3.2</a>	Java 8+	2017-04-14
<a href="#">3.1</a>	Java 7+	2016-11-20
<a href="#">3.0</a>	Java 7+	2016-05-17
<a href="#">2.13</a>	Java 6+	2015-03-13
<a href="#">2.12</a>	Java 6+	2014-11-10
<a href="#">2.11</a>	Java 6+	2014-01-05
<a href="#">2.10</a>	Java 6+	2013-10-21
<a href="#">2.9</a>	Java 6+	2013-01-28
<a href="#">2.8</a>	Java 5+	2012-10-06
<a href="#">2.7</a>	Java 5+	2012-05-27
<a href="#">2.6</a>	Java 5+	2012-02-01
<a href="#">2.5.1</a>	Java 5+	2011-10-03

Версия	Версия Java	Дата выхода
2.5	Java 5+	2011-08-17
2.4	Java 5+	2010-07-12
2.3.4	Java 1.4+	2009-06-21

## Examples

### Установка или настройка

1. Загрузите распределенный архив из раздела Binaries в JMeter с страницы [загрузки Apache JMeter](#).
2. В зависимости от загруженной версии проверьте [минимальные требования к Java-версии](#) и установите Java, если это необходимо. Убедитесь, что переменная среды JAVA\_HOME установлена и указывает на правильную версию.
3. Извлеките архив распространения в каталог по вашему выбору.
4. Открыть JMeter UI:
  - **В Windows** : перейдите в <jmeter\_location>\bin и запустите jmeterw.bat **ИЛИ** jmeter.bat
  - **В Linux / Mac** : перейдите в <jmeter\_location>/bin и запустите jmeter **ИЛИ** 'jmeter.sh`.

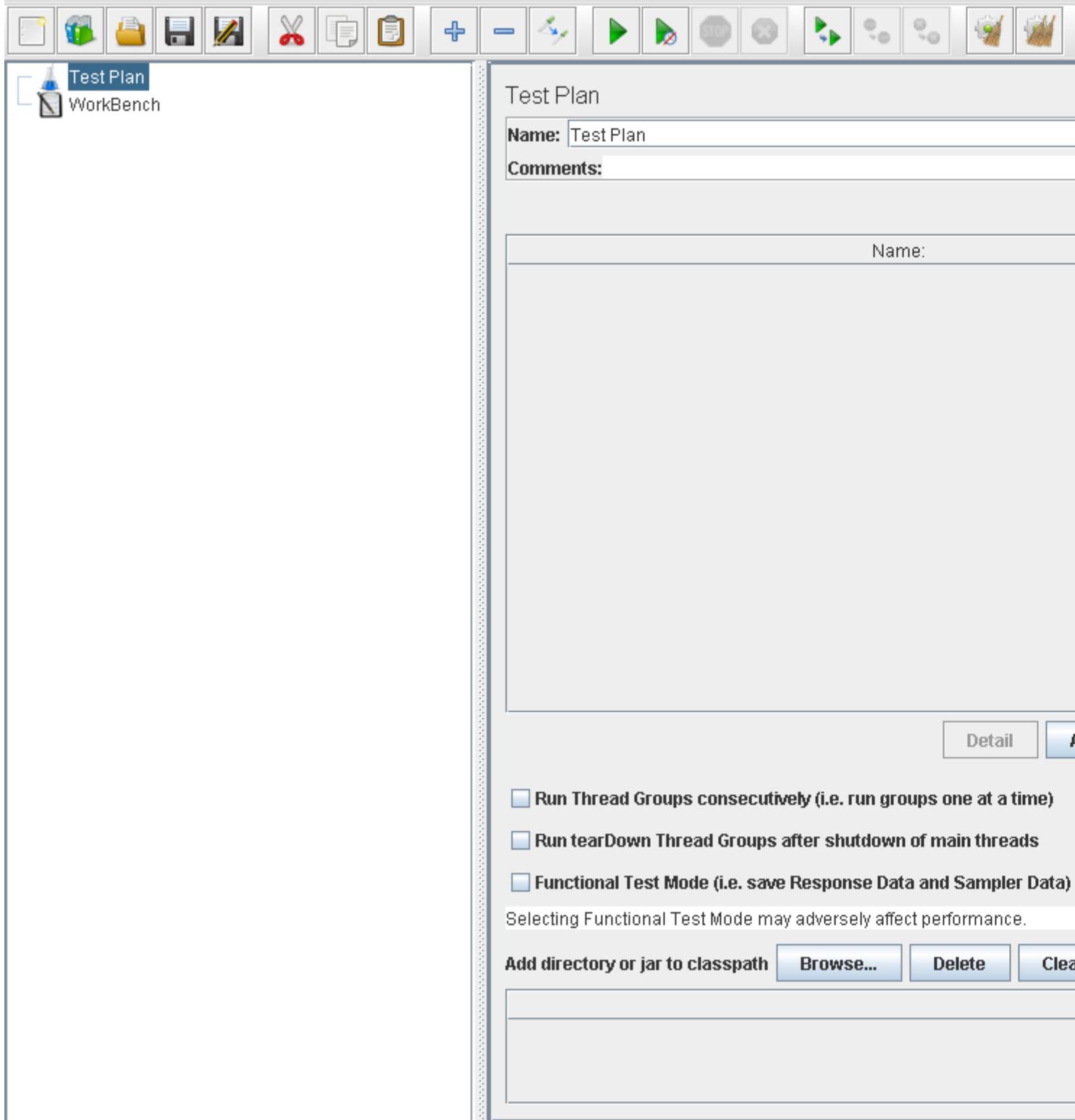
Например:

```
cd /Users/me/apache-jmeter/bin
./jmeter
```

**Примечание** : если вышеприведенная команда завершилась с ошибкой Permission denied , установите разрешение на jmeter файле jmeter :

```
cd /Users/me/apache-jmeter/bin
chmod u+x ./jmeter
```

Если вы можете видеть JMeter UI, базовая настройка была успешной.



## Обзор компонентов Apache JMeter на высоком уровне

Apache JMeter разделил все компоненты на следующие группы в зависимости от их функциональности:

1. **Test Plan**: начальная точка для сценариев. JMeter сохраняет план тестирования в формате .jmx. Вы добавляете компоненты в план тестирования, щелкнув правой

- кнопкой мыши на тестовой панели и перейдя к компоненту, который хотите добавить.
- 2. **Workbench** : Временное место для запуска скриптов. Наряду со всеми компонентами, доступными в плане тестирования, вы получаете `HTTP(s) Test Script Recorder` для `record` действий браузера. Скрипты могут быть сохранены в Workbench при условии, что вы установите флајок «Save Workbench», иначе они не будут.
  - 3. **Threads (Users)** : вы можете определить количество (виртуальных) пользователей для запуска, времени нарастания и количества циклов. вы также можете определить в `Test Plan`, должны ли группы потоков запускаться последовательно или параллельно в случае нескольких групп Thread. некоторые примеры представляют собой `Thread Group`, `setUp Thread Group`, and `tearDown Thread Group`
  - 4. **Logic Controller** : Позволяет определить поток выполнения и группировку сэмплеров. одним из полезных примеров является Transaction Controller, где вы объединяете все пробоотборники страницы входа (все ресурсы, включая изображения, файлы .css и .js), чтобы можно было получить общее время отклика.
  - 5. **Sampler** : Sampler является ядром JMeter. Он предоставляет компоненты для имитации запросов различных протоколов, таких как HTTP, JDBC, FTP, SMTP и т. д., Например, HTTP-пробоотборник позволяет моделировать HTTP-пакет (GET, POST или любые поддерживаемые методы). Поддерживаются протоколы основного потока, для других вы можете использовать бесплатные или коммерческие плагины.
  - 6. **Config Element** : Элементы конфигурации могут использоваться для настройки значений по умолчанию и переменных для последующего использования сэмплерами. Обратите внимание, что эти элементы обычно обрабатываются в начале области, в которой они найдены, т.е. перед любыми пробоотборниками в той же области. `CSV Dataset Config` позволяет вам предоставлять тестовые данные, такие как имена пользователей, пароли сценария входа в систему `from a file`. `User Defined variables` элемент конфигурации `User Defined variables` позволяет определять переменные, которые могут использоваться в плане тестирования, но где каждый поток имеет свою собственную копию.
  - 7. **Timer** : по умолчанию поток JMeter выполняет последовательные последовательности без паузы. Представленные здесь компоненты обеспечивают функциональность, позволяющую ввести `User Think Time` в различные формы среди пробоотборников. некоторые примеры - Constant Timer, Constant Throughput Timer.
  - 8. **Pre Processors** : позволяют выполнять операции / действия перед выполнением сэмплера. JSR223 Pre Processor с `Apache Groovy` (аналогично стилю java-кодирования) позволяет вам внести изменения в пробоотборник перед его отправкой.
  - 9. **Post Processors** : позволяет выполнять операции / действия после выполнения пробоотбора. некоторые полезные примеры извлекают динамическое значение, такое как идентификатор сеанса, с использованием `Regular Expression Extractor` для любого типа текста, `CSS/JQuery Extractor` для HTML, `JSON Extractor` для JSON, `XPath Extractor` для XML.
  - 10. **Assertions** . Как следует из названия, вы можете утверждать ответ пробников по-разному, например, искать какой-либо текст, размер ответа и продолжительность для получения ответа и т. д. Например, вы можете использовать `Response Assertion`

для поиска некоторого текста в ответе. Если Assertion терпит неудачу, JMeter отмечает пробоотборник, к которому применяется Assertion, как Failure.

11. Слушатели: Слушатели позволяют сохранять результаты теста, видеть выполнение теста и т. д., Например, используя « View Results Tree , вы можете увидеть запрос / ответчик сэмплеров и обозначены ли они как PASS (зеленый цвет) / FAIL (красный цвет) JMeter. используя сводный отчет, вы можете сохранить результаты теста в формате CSV. Важно отметить, что вы используете слушателей либо до запуска теста (для отладки тестового сценария), либо после тестового запуска (для просмотра результатов в графиках или сводке), но не во время прогона. мы должны удалить слушателей во время теста, поскольку он потребляет много системных ресурсов. Итак, мы запускаем тест в режиме без GUI и сохраняем результаты с использованием опции -l в .csv/.jtl . Отправьте тест, вы можете загрузить эти сохраненные файлы в любой из прослушивателей JMeter, чтобы просмотреть графики / сводки.

Ниже приведен общий синтаксис ( you add any component on need basis ):

```
Test Plan
  Thread Group
    Config Element
    Logic Controller
      Pre Processor
      Sampler
      Timer
      Post Processor
      Assertion
    Listener
```

Рекомендации:

1. [План тестирования и компоненты](#)
2. [Порядок исполнения](#)
3. [Правила определения области действия](#)

Прочтайте Начало работы с Apache JMeter онлайн:

<https://riptutorial.com/ru/jmeter/topic/1941/начало-работы-с-apache-jmeter>

# глава 2: Apache JMeter Correlations

## Вступление

При тестировании производительности JMeter Correlations означает возможность получать динамические данные из ответа сервера и отправлять его на последующие запросы. Эта функция имеет решающее значение для многих аспектов тестирования, таких как защищенные приложения на основе токенов.

## Examples

### Корреляция Использование экстента регулярного выражения в Apache JMeter

Если вам нужно извлечь информацию из текстового ответа, самым простым способом будет использование регулярных выражений. Соответствующий шаблон очень похож на шаблон, используемый в Perl. Предположим, мы хотим протестировать рабочий процесс покупки авиабилета. Первый шаг - отправить операцию покупки. Следующий шаг - убедиться, что мы можем проверить все детали, используя идентификатор покупки, который должен быть возвращен для первого запроса. Представим, что первый запрос возвращает html-страницу с этим типом идентификатора, который нам нужно извлечь:

```
<div class="container">
  <div class="container hero-unit">
    <h1>Thank you for your purchase today!</h1>
    <table class="table">
      <tr>
        <td>Id</td>
        <td>Your purchase id is 1484697832391</td>
      </tr>
      <tr>
        <td>Status</td>
        <td>Pending</td>
      </tr>
      <tr>
        <td>Amount</td>
        <td>120 USD</td>
      </tr>
    </table>
  </div>
</div>
```

Такая ситуация является лучшим кандидатом на использование экстрактора JMeter Regular Expression. Регулярное выражение представляет собой специальную текстовую строку для описания шаблона поиска. Существует множество онлайн-ресурсов, которые помогают писать и тестировать регулярные выражения. Один из них - <https://regex101.com/>.

## REGULAR EXPRESSION

// Your purchase id is (\d\*).

## TEST STRING

```
....|  
<tbody>  
  <tr>  
    <td>Id</td>  
    <td>Your purchase id is 1484697832391. T  
  </tr>  
  <tr>  
    <td>Status</td>  
    <td>PendingCapture</td>  
  </tr>  
....
```

Чтобы использовать этот компонент, откройте меню JMeter и: *Add -> Post Processors -> Extractor Expression Extractor*

The screenshot shows the JMeter Test Plan editor interface. On the left, there's a tree view of the test plan structure:

- Test Plan
- Purchase workflow
- Perform flight purchase request
- Regular Expression Extractor (selected)

On the right, the configuration for the selected 'Regular Expression Extractor' is displayed:

**Regular Expression Extractor**

Name: Regular Expression Extractor

Comments:

Apply to:  Main sample and sub-samples  M

Field to check:  Body  Body (unesaped)  Bod

Reference Name: purchaseld

Regular Expression: Your purchase i

Template: \$1\$

Match No. (0 for Random): 1

Default Value: NOT\_FOUND

Экземпляр регулярного выражения содержит следующие поля:

- Reference Name - имя переменной, которая может быть использована после извлечения
- Регулярное выражение - последовательность символов и символов, выражающих строку (шаблон), которая будет искать в тексте
- Шаблон - содержит ссылки на группы. Поскольку регулярное выражение может иметь более одной группы, оно позволяет указать, какое значение группы нужно извлечь, указав номер группы как \$ 1 \$ или \$ 2 \$ или \$ 1 \$\$ 2 \$ (извлечь обе группы)
- Номер совпадения - указывает, какое совпадение будет использоваться (значение 0 соответствует случайнм значениям / любое положительное число N означает выбор N-го совпадения / отрицательного значения, которое должно использоваться с контроллером ForEach)
- Значение по умолчанию - значение по умолчанию, которое будет сохранено в переменной в случае, если совпадения не найдены, сохраняется в переменной.

Флажок «Применить к» относится к образцам, которые делают запросы на встроенные ресурсы. Этот параметр определяет, будет ли регулярное выражение применяться к основным результатам выборки или ко всем запросам, включая внедренные ресурсы. Для этого параметра есть несколько вариантов:

- Основная выборка и подвыборки
- Только основной образец
- Только суб-выборки
- JMeter Variable - утверждение применяется к содержимому именованной переменной, которое может быть заполнено другим запросом

Флажок «Поле для проверки» позволяет выбрать, в какое поле должно быть применено регулярное выражение. Почти все параметры самоописательны:

- Тело - тело ответа, например содержимое веб-страницы (исключая заголовки)
- Body (unesaped) - тело ответа, с заменой всех кодов языка HTML. Обратите внимание, что экранирование HTML обрабатывается без учета контекста, поэтому могут быть сделаны некоторые неправильные замены (\* этот параметр сильно влияет на производительность)
- Тело - Тело как документ - текст извлечения из различных типов документов через Apache Tika (\* также может повлиять на производительность)
- Заголовок тела - запрос - может отсутствовать для образцов, отличных от HTTP
- Заголовок Body - Response - может отсутствовать для образцов, отличных от HTTP
- Тело - URL
- Код ответа - например, 200
- Тело - ответное сообщение - например, OK

После выделения выражения его можно использовать в последующих запросах с помощью переменной \${purchaseld}.

The screenshot shows the JMeter Test Plan editor. On the left, the test plan tree is visible, showing a 'Test Plan' node expanded to reveal a 'Purchase workflow' node, which further expands to show a 'Perform flight purchase request' node, a 'Regular Expression Extractor' node, and an 'Open purchased flight details' node. The 'Open purchased flight details' node is currently selected, indicated by a blue selection bar. On the right, the configuration panel for this selected step is displayed. The title of the panel is 'HTTP Request'. The 'Name:' field contains the value 'Open purchased flight details'. The 'Comments:' field is empty. Below these, there are two sections: 'Web Server' and 'HTTP Request'. Under 'Web Server', the 'Server Name or IP:' field is set to 'http://demoapp.com'. Under 'HTTP Request', the 'Implementation:' dropdown is set to 'HTTP(S) Test Script Recorder'. The 'Path:' field contains '/viewPurchase.php'. There are two checkboxes at the bottom: 'Redirect Automatically' (unchecked) and 'Follow Redirects' (checked). A preview window shows a simple form with a single input field labeled 'Name:' containing the value 'purchaseId'.

Эта таблица содержит все сокращения, которые поддерживаются регулярными выражениями JMeter:

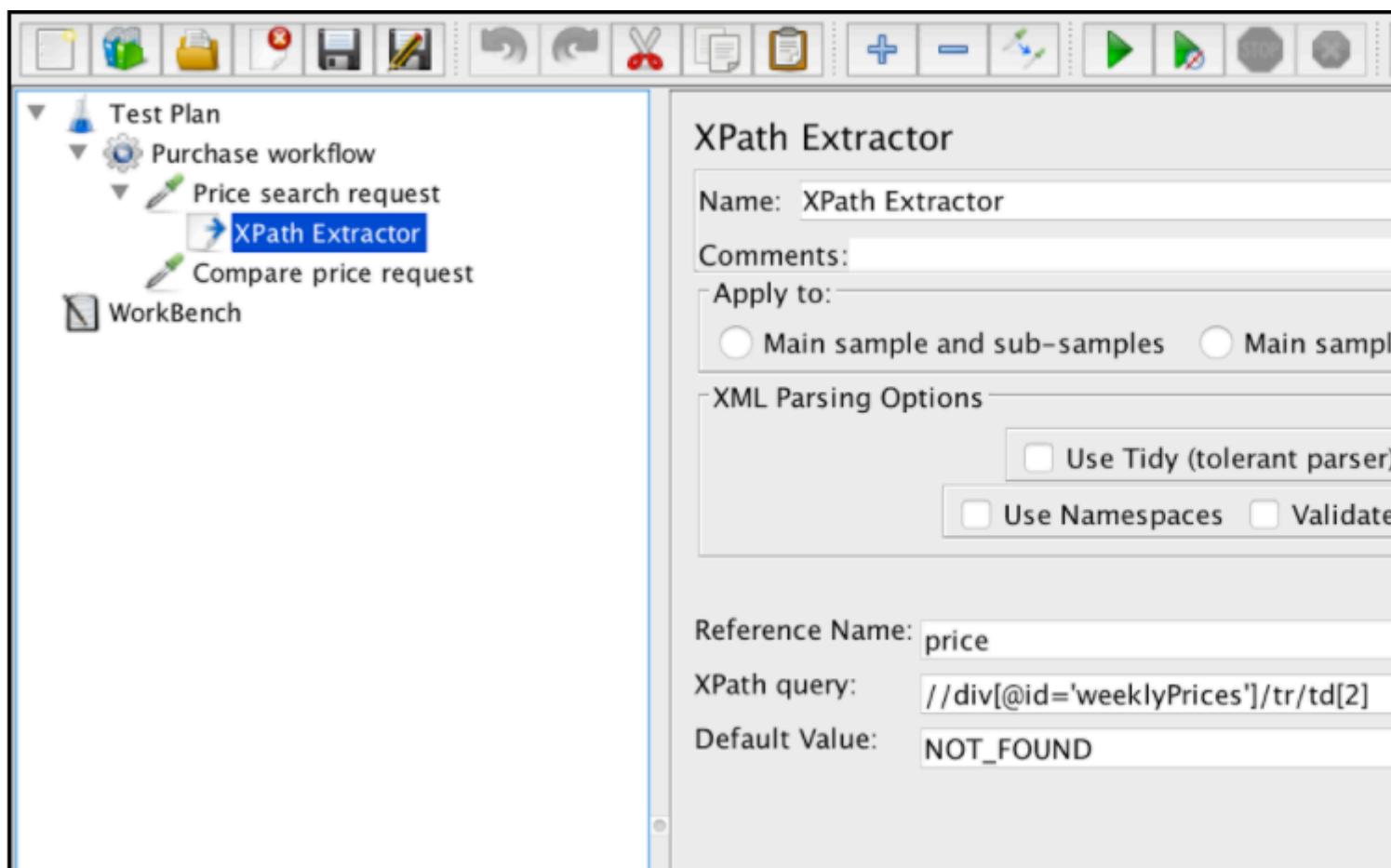
<b>chars</b>	<b>action</b>
abc...	Match that character (metacharacter)
\W.\*...	Match that metacharacter following
""	De-meta any chars inside quotes
\t,\n,\r,\f	tab, newline, return, form feed
.	Match any character
[]	Character class
[^]	Inverse Character class
[-]	Character ranges
\w	Match a "word" character (alphanumeric)
\W	Match a non-word character
\s	Match a whitespace character
\S	Match a non-whitespace character
\d	Match a digit character
\D	Match a non-digit character
<b>anchors</b>	<b>action</b>
^	Match the beginning of the line
\$	Match the end of the line
\b	Match a word boundary
\B	Match a non-(word boundary)

Это может быть полезно, когда данные из ответа не могут быть извлечены с помощью экстента регулярного выражения. Например, в случае сценария, в котором вам необходимо извлечь данные из похожих тегов с одинаковыми атрибутами, но с разными значениями. XPath Extractor похож на CSS / JQuery Extractor, но XPath Extractor следует использовать для содержимого XML, в то время как CSS / JQuery Extractor следует использовать для содержимого HTML. Предположим, что в ответе у нас есть таблица с разными значениями, где нам нужно извлечь значение из второй строки таблицы.

```
<div id="weeklyPrices">
<tr>
<td>$56.00</td>
<td>$56.00</td>
<td>$56.00</td>
<td>$56.00</td>
<td>$60.00</td>
<td>$70.00</td>
<td>$70.00</td>
</tr>
</div>
```

Забегая вперед, правильный XPath для этого случая будет: `//div[@id = 'weeklyPrices']/tr/td[1]`

Чтобы использовать этот компонент, откройте меню JMeter и: `Add -> Post Processors -> XPath Extractor`



XPath Extractor содержит несколько общих элементов конфигурации, которые упоминаются в «Корреляции с использованием экстента регулярных выражений». Это включает в себя Name, Apply to, Reference Name, Match No. (с JMeter 3.2) и значение по умолчанию.

Существует множество веб-ресурсов с онлайн-чит-листами и редакторами для создания и тестирования созданного xpath (как [этот](#)). Но на основе приведенных ниже примеров мы можем найти способ создания наиболее распространенных локаторов xpath.

//hr[@class='edge' and position()=1]	every first hr
//table[count(tr)=1 and count(tr/td)=2]	all tables
//div/form/parent::*	all divs that contain forms
.//div/b	a relative locator
/html/body/div/*[preceding-sibling::h4]	give me all preceding siblings of h4
//tr/td[font[@class="head" and text()="TRACK"]]	all td that have text "TRACK"
.//table/tr[last()]	the last row of the table
//rdf:Seq/rdf:li/em:id	using namespaces
//a/@href	hrefs of all anchor tags
//*[count(*)=3]	all nodes with three children
//var //acronym	all vars and acronyms

Если вы хотите проанализировать HTML-код на XHTML, нам нужно проверить опцию «Использовать Tidy». После определения статуса «Использовать Tidy» есть дополнительные опции:

Если «Использовать Tidy» отмечен:

- Quiet - устанавливает флаг Tidy Quiet
- Ошибки отчета - если возникает ошибка Tidy, установите Assertion соответственно
- Show Warnings - устанавливает опцию предупреждения Tidy show

Если флагок Использовать Tidy не установлен:

- Использовать пространства имен - если флагок, парсер XML будет использовать разрешение пространства имен
- Проверка XML - проверка документа по указанной схеме
- 忽略 пробелы - игнорировать пробелы элемента
- Fetch External DTD - если выбрано, выбираются внешние DTD

«Возвращать весь фрагмент XPath вместо текстового содержимого» является самоописательным и должен использоваться, если вы хотите вернуть не только значение xpath, но также и значение в его локаторе xpath. Это может быть полезно для нужд отладки.

Также стоит упомянуть список очень удобных плагинов браузера для тестирования локаторов XPath. Для Firefox вы можете использовать плагин « [Firebug](#) », а для Chrome « [XPath Helper](#) » - самый удобный инструмент.

The screenshot shows the Stack Overflow Documentation page for the search term 'jmeter'. At the top, there's a navigation bar with links like 'Documentation', 'Help Center', 'FAQ', and 'About'. Below the navigation is a search bar with the text 'Type to find a tag: jmeter'. A search icon is to the right of the search bar. The main content area displays a card for the 'jmeter' tag. The card has a blue header with the tag name, '2 topics' below it, and a 'Dashboard' section. Below the dashboard are three small icons: a blue square with a white plus sign, a blue square with a white minus sign, and a blue square with a white document icon. At the bottom of the card is a 'Report Abuse' link. The bottom of the page features a toolbar with various icons (e.g., magnifying glass, file, refresh) and tabs for 'Console', 'HTML', 'CSS', 'Script', 'DOM', 'Net', and 'Code'. A dropdown menu 'Top Window' is open, showing 'Highlight' as the selected option. To the right of the dropdown is an 'XPath' field containing the expression //div[@class='card-top']/a[contains(text(),'jmeter')]. Below the XPath field is a tree view of the DOM structure, with nodes highlighted in red. The first node is a div with class 'doctag-card', which contains another div with class 'card-top'. This second div contains an anchor tag with the href '/documentation/jmeter'. The anchor tag is also highlighted in red.

## Корреляция Использование экстрактора CSS / JQuery в JMeter

Экстрактор CSS / JQuery позволяет извлекать значения из ответа сервера с помощью синтаксиса селектора CSS / JQuery, который в противном случае было бы трудно записать с использованием регулярного выражения. В качестве постпроцессора этот элемент должен быть выполнен для извлечения запрошенных узлов, текста или значений атрибутов из пробоотбора запроса и сохранения результата в заданную переменную. Этот компонент очень похож на XPath Extractor. Выбор между CSS, JQuery или XPath обычно зависит от предпочтений пользователей, но стоит упомянуть, что XPath или JQuery могут перемещаться вниз, а также перемещаться по DOM, в то время как CSS не может подойти к DOM. Предположим, что мы хотим извлечь все темы из документации переполнения стека, которые связаны с Java. Вы можете использовать плагин Firebug для тестирования

селекторов CSS / JQuery в Firefox или тестера CSS Selector в Chrome.

The screenshot shows the Stack Overflow Documentation page. At the top, there's a search bar with the placeholder "Type to find a tag:" containing the word "java". Below the search bar, there are four search results cards:

- Java Language** (153 topics): Includes a "Dashboard" button, a "16" icon, a "0" icon, and a "4" icon.
- JavaScript** (101 topics): Includes a "Dashboard" button, a "0" icon, a "0" icon, and a "1" icon.
- java-ee** (6 topics): Includes a "Dashboard" button, a "0" icon, a "0" icon, and a "1" icon.
- java.util.scanner** (1 topic): Includes a "Dashboard" button, a "0" icon, a "0" icon, and a "1" icon.

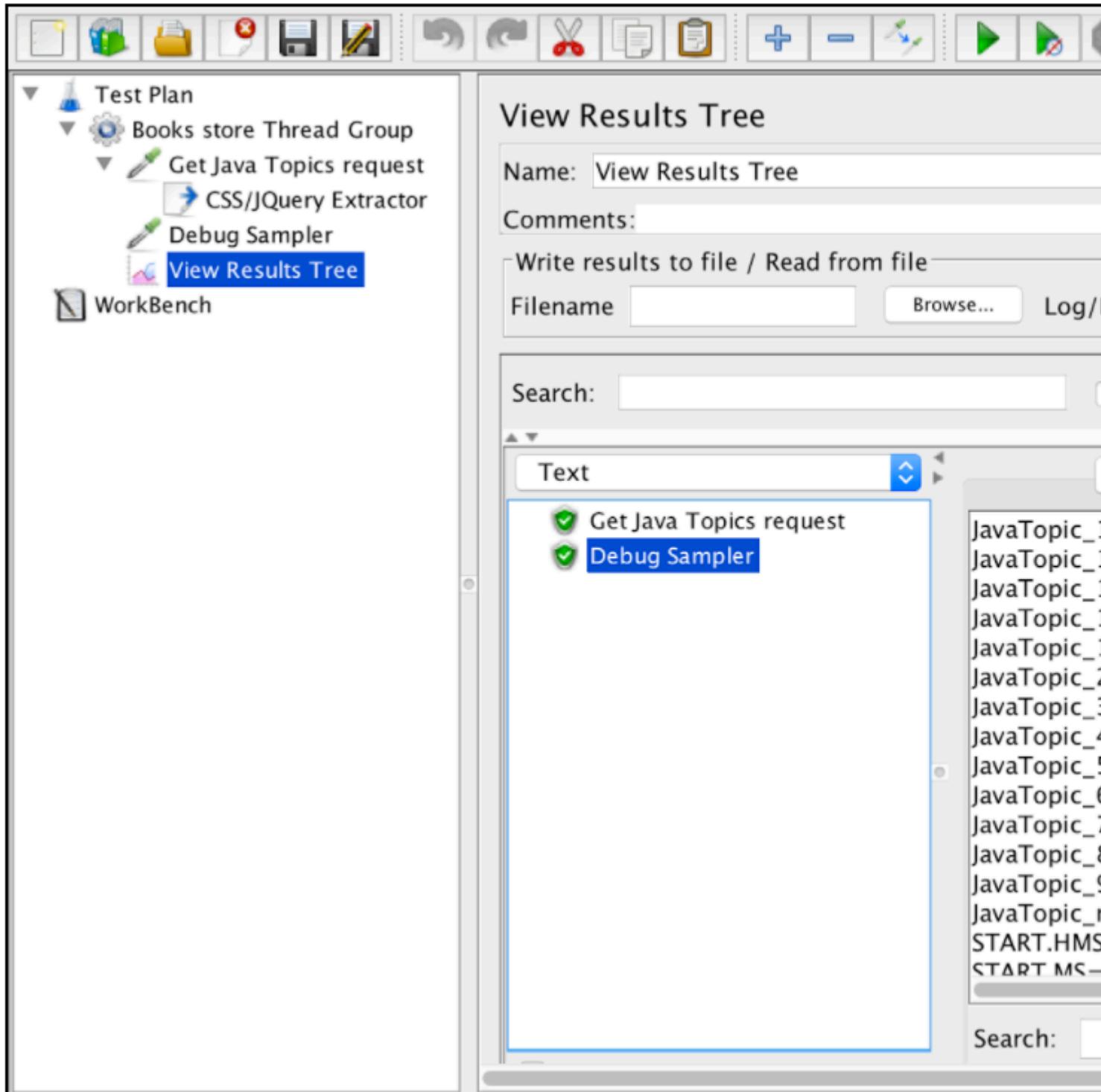
At the bottom of the page, there's a navigation bar with icons for Bee, Filter, Back, Forward, and More, followed by tabs for Console, HTML, CSS, Script, DOM, Net, and C. A "Top Window" dropdown is set to "Highlight". The "Highlight" tab is selected, showing a CSS selector tree:

```
CSS: .card-top>a[href*="documentation"]  
      ↓  
      <div class="filter-and-propose-controls"  
      ↓<div class="doctag-cards-container">  
      ↓<table class="doctag-cards">  
      ↓<tbody>  
      ↓<tr>  
      ↓<td>  
      ↓<div class="doctag-card">  
      ↓<div class="card-top">  
      ↓<a href="/documentation/j...  
      ↓<span class="item-multipl...  
      ↓<a class="help-bubble js-...  
      track="helpbubble.shown({}  
      type="4" data-bubble-posi...  
      class="dive-in help-bubb..."
```

Чтобы использовать этот компонент, откройте меню JMeter и: Add -> Post Processors -> CSS / JQuery Extractor

The screenshot shows the JMeter interface. On the left, the Test Plan tree view displays a 'Books store Thread Group' containing a 'Get Java Topics request' sampler, which has a 'CSS/JQuery Extractor' configured under it. Other items like 'Debug Sampler' and 'View Results Tree' are also visible. On the right, a detailed configuration dialog for the 'CSS/JQuery Extractor' is open. The 'Name' field is set to 'CSS/JQuery Extractor'. Under 'Apply to:', the 'Main sample and sub-samples' radio button is selected. In the 'Implementation' section, 'JSOUP' is chosen over 'JODD Lagarto'. The 'Reference Name' is set to 'JavaTopic', and the 'CSS/JQuery expression' is '.card-top>a[href\*="docu']'. The 'Attribute' field is empty. The 'Match No.' field is set to '-1', and the 'Default Value' is 'NOT\_FOUND'.

Почти все поля этого экстрактора аналогичны полям экстрактора регулярного выражения, поэтому вы можете получить их описание из этого примера. Однако одно отличие - это «реализация реализации CSS / JQuery Extractor». Начиная с JMeter 2.9 вы можете использовать экстрактор CSS / JQuery на основе двух разных реализаций: реализации [jsoup](#) (подробное описание его синтаксиса [здесь](#)) или [JODD Lagarto](#) (подробный синтаксис можно найти [здесь](#)). Обе реализации почти одинаковы и имеют лишь небольшие различия в синтаксисе. Выбор между ними основан на предпочтениях пользователя.



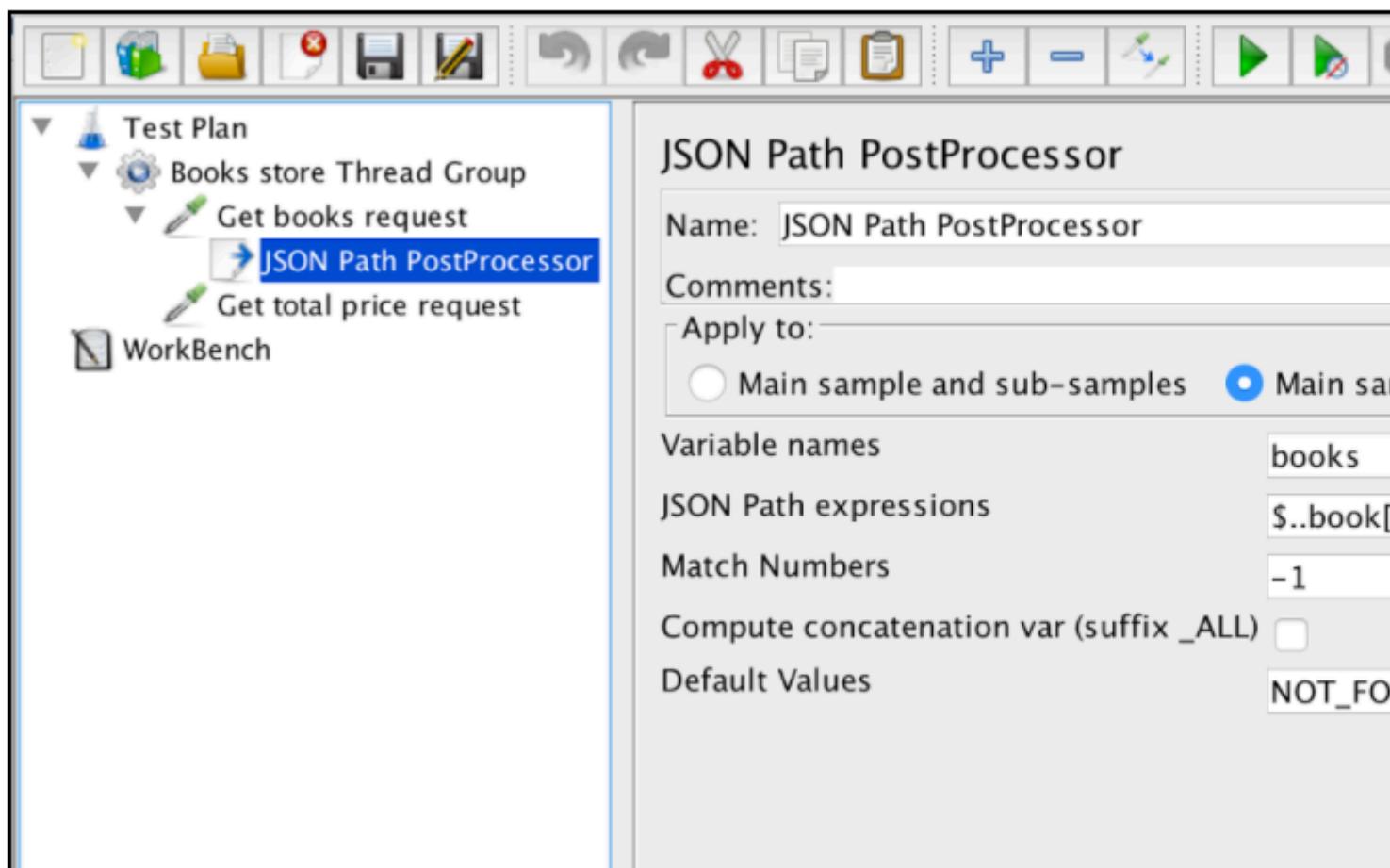
На основе вышеупомянутой конфигурации мы можем извлечь все темы из запрошенной страницы и проверить извлеченные результаты с помощью «Debug Sampler» и «View Results Tree».

## Корреляция Использование экстрактора JSON

JSON - широко используемый формат данных, который используется в веб-приложениях. JMeter JSON Extractor предоставляет способ использования выражений JSON Path для извлечения значений из JSON-ответов в JMeter. Этот почтовый процессор должен быть помещен как дочерний элемент пробоотбора HTTP или для любого другого сэмплера,

который имеет ответы.

Чтобы использовать этот компонент, откройте меню JMeter и: *Add -> Post Processors -> JSON Extractor*.



JSON Extractor очень похож на экспрессион регулярного выражения. В этом примере упоминаются почти все основные поля. Существует только один конкретный параметр JSON Extractor: «Вычислить конкатенацию var». В случае обнаружения многих результатов этот экстрактор будет конкатенировать их, используя разделитель «,» и сохраняя его в var с именем \_ALL.

Предположим, что этот ответ сервера с JSON:

```
{  
    "store": {  
        "book": [  
            { "category": "reference",  
              "author": "Nigel Rees",  
              "title": "Sayings of the Century",  
              "price": 8.95  
            },  
            { "category": "fiction",  
              "author": "Evelyn Waugh",  
              "title": "Sword of Honour",  
              "price": 12.99  
            },  
            { "category": "fantasy",  
              "author": "Terry Pratchett",  
              "title": "Discworld",  
              "price": 22.99  
            }  
        ]  
    }  
}
```

```
{ "category": "fiction",
  "author": "Herman Melville",
  "title": "Moby Dick",
  "isbn": "0-553-21311-3",
  "price": 8.99
},
{ "category": "fiction",
  "author": "J. R. R. Tolkien",
  "title": "The Lord of the Rings",
  "isbn": "0-395-19395-8",
  "price": 22.99
}
],
"bicycle": {
  "color": "red",
  "price": 19.95
}
}
```

В приведенной ниже таблице представлен отличный пример различных способов извлечения данных из указанного JSON:

\$.store.book[*].author	the authors of all books
\$.author	all authors
\$.store.*	all things in the store, including red bicycle
\$.store..price	the prices of all books
\$.book[2]	the third book
\$.book[(@.length-1)] \$.book[-1:]	the last book
\$.book[0,1] \$.book[:2]	the first two books
\$.book[?(@.isbn)]	filter all books by ISBN
\$.book[?(@.price<10)]	filter all books with price less than 10
\$..*	all Elements in the JSON structure

С помощью этой [ссылки](#) вы можете найти более подробное описание формата JSON Path с соответствующими примерами.

## Автоматическая корреляция с использованием «SmartJMX» от BlazeMeter

Когда вы вручную записываете свои скрипты производительности, вам нужно иметь дело с корреляцией самостоятельно. Но есть еще один вариант для создания сценариев - записи сценариев автоматизации. С одной стороны, ручной подход помогает писать

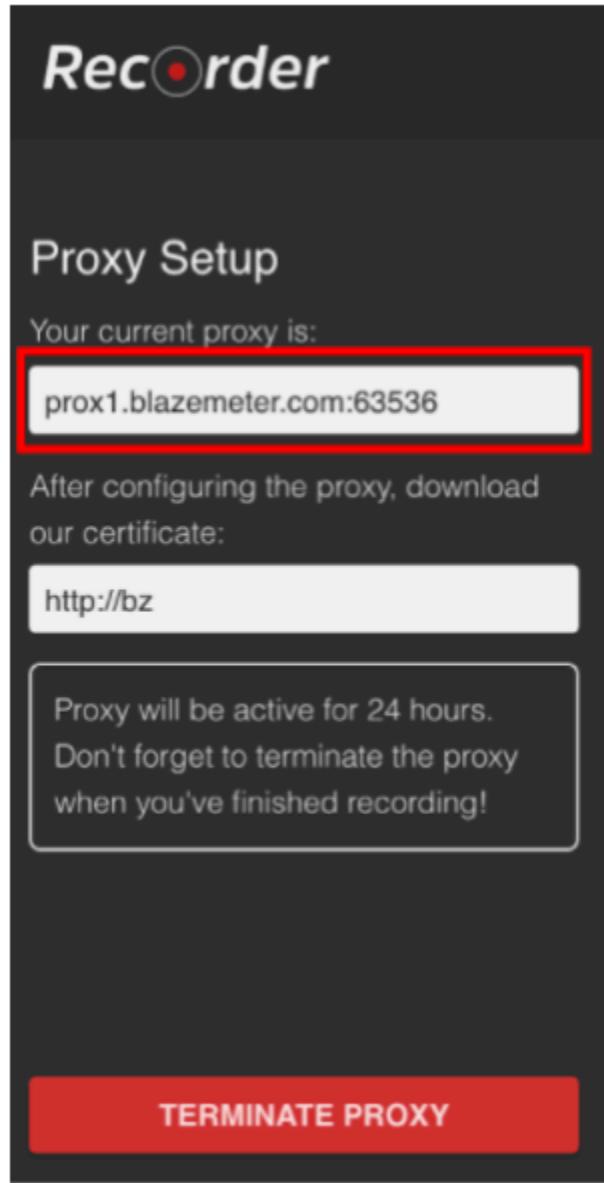
структурированные скрипты, и вы можете одновременно добавить все необходимые экстракторы. С другой стороны, этот подход требует много времени.

Запись сценариев автоматизации очень проста и позволяет выполнять одну и ту же работу, только намного быстрее. Но если вы используете общие способы записи, сценарии будут очень неструктурированными и обычно требуют добавления дополнительной параметризации. Функция «Smart JMX» на рекордере BlazeMeter сочетает в себе преимущества обоих способов. Его можно найти по этой ссылке: [<https://a.blazemeter.com/app/recorder/index.html>][1]

После регистрации перейдите в раздел «Рекордер».

The screenshot shows the BlazeMeter dashboard for user Yuri Bushnev. At the top, there's a navigation bar with links for Projects, Tests, Reports, and Private Locations. Below that is a user profile section with the name 'Yuri Bushnev'. A navigation menu at the top includes Dashboard, Projects (which is selected), Resources, and Settings. The main content area features a greeting 'Hi, Yuri Bushnev' and several buttons for creating new tests: 'Add JMeter Test' (purple), 'Add Taurus Test' (dark grey, marked as 'New'), 'Add URL/API Test' (blue), and 'Add Webdriver Test' (green). Below these buttons is a 'Recent Activity' section. It lists three items: 1) '2 hours ago' - 'Yuri Bushnev updated test TEST SCRIPT (50 virtual users)'. 2) '4 hours ago' - 'Yuri Bushnev terminated session 30 Users - Search without charts'. 3) '4 hours ago' - 'Yuri Bushnev started test RC - search (1 virtual user)'. To the right of the activity list, there's a small image of a smartphone displaying a mobile application interface. The overall layout is clean and modern, typical of a web-based testing tool.

Чтобы начать запись сценария, сначала вам нужно настроить прокси-сервер вашего браузера ( [здесь](#) и [здесь](#) ), но на этот раз вы должны получить прокси-хост и порт, предоставленный рекордером BlazeMeter.



The screenshot shows the 'Recently Captured Requests' screen of the Recorder app. At the top right is a search bar with a magnifying glass icon and the placeholder 'Search...'. Below it is a section titled 'Recently Captured Requests'. To the left of this section, a red arrow points from the text 'proxy host]:[port]' to the red-bordered proxy address in the 'Proxy Setup' screen. On the far right edge of the screen, there is a red circular icon with a white dot in the center.

Когда браузер настроен, вы можете продолжить запись сценария, нажав красную кнопку внизу. Теперь вы можете перейти к тестируемому приложению и выполнить пользовательские рабочие процессы для записи.

# Recorder

## Proxy Setup

Your current proxy is:

prox1.blazemeter.com:63536

After configuring the proxy, download our certificate:

http://bz

Proxy will be active for 24 hours.  
Don't forget to terminate the proxy when you've finished recording!

**TERMINATE PROXY**

Search...

## Recently Captured Requests

[ 20:45:39 ]	POST	http://blazemeter.com/confirmation.php
[ 20:45:36 ]	POST	http://blazemeter.com/purchase.php
[ 20:45:31 ]	GET	http://blazemeter.com/favicon.ico
[ 20:45:29 ]	GET	http://blazemeter.com/assets/bootstrap.min.j
[ 20:45:29 ]	GET	http://blazemeter.com/assets/bootstrap-table.
[ 20:45:29 ]	GET	http://ajax.googleapis.com/ajax/libs/jquery/2
[ 20:45:29 ]	GET	http://blazemeter.com/assets/bootstrap-table.
[ 20:45:29 ]	GET	http://blazemeter.com/assets/bootstrap.min.
[ 20:45:29 ]	GET	http://blazemeter.com/reserve.php

После записи сценария вы можете экспортить результаты в файл SMM JMX. Экспортированный файл jmx содержит список параметров, которые позволяют настраивать скрипт и параметризовать без дополнительных усилий. Одним из таких усовершенствований является то, что «SMART» JMX автоматически находит кандидатов на корреляцию, заменяет их соответствующим экстрактором и предоставляет простой способ дальнейшей параметризации.

Прочтите Apache JMeter Correlations онлайн:

<https://riptutorial.com/ru/jmeter/topic/8978/apache-jmeter-correlations>

# глава 3: Apache JMeter: запись сценария сценария

## Вступление

Запись тестовых сценариев - один из наиболее удобных способов создания тестовых сценариев. Это связано с тем, что тестовые записи позволяют воспроизводить реалистичные рабочие процессы пользователей, вместо того, чтобы вручную создавать тестовый скрипт. Записи захватывают все запросы браузера к веб-приложению, а затем автоматически создают файл jmx, который можно запускать в тестах производительности. Используя функции записи / воспроизведения JMeter или сторонние инструменты, такие как BlazeMeter и BadBoy, тестеры могут сделать свою работу в 3 раза быстрее.

## Examples

### Запись скриптов с помощью функции шаблона JMeter

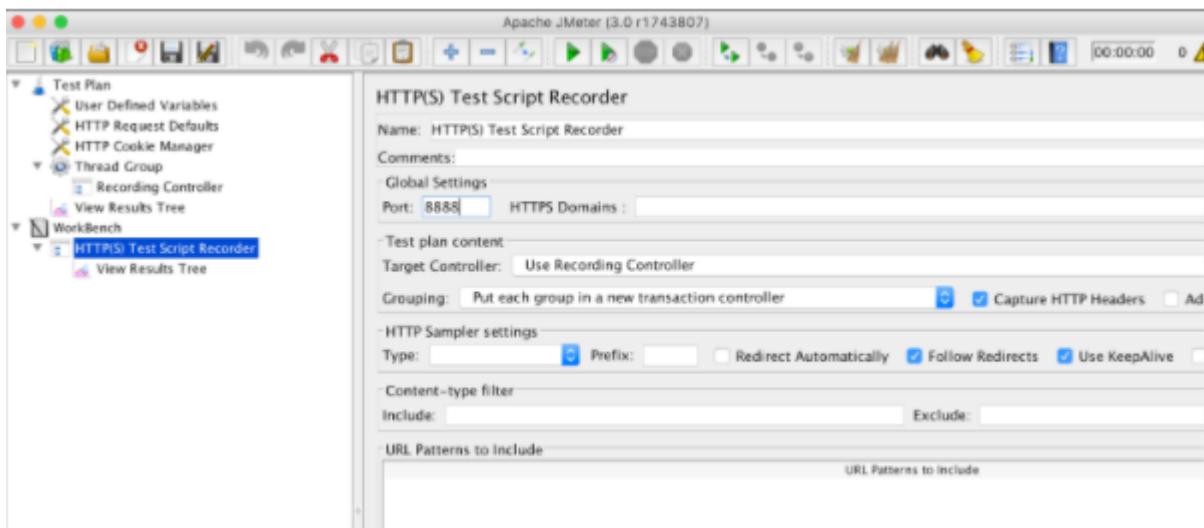
В версии 2.10 [JMeter](#) представил механизм, который экономит ваше время при создании скриптов - шаблонов JMeter. Эти шаблоны являются скелетами, которые можно использовать повторно в качестве базы для новых сценариев.

JMeter уже имеет несколько доступных шаблонов с подробными описаниями, и вы также можете добавить свой собственный. Шаблоны имеют всю необходимую конфигурацию и элементы для записи сценариев производительности с нуля.

Вот как использовать шаблон шаблона JMeter:

#### Настроить JMeter

1. Открыть JMeter
2. Выберите шаблон для записи сценария:  
Файл -> Шаблоны ... -> Выбрать шаблон -> Запись -> Создать JMeter добавит соответствующие элементы в тестовое дерево.



## Настройте свой прокси-сервер

Чтобы использовать JMeter Recorder, вам необходимо настроить браузер для отправки всех запросов через прокси. Любой браузер может быть использован для этих нужд, хотя могут быть различия между местоположениями конфигураций браузеров, которые зависят от браузера и могут варьироваться в зависимости от ОС.

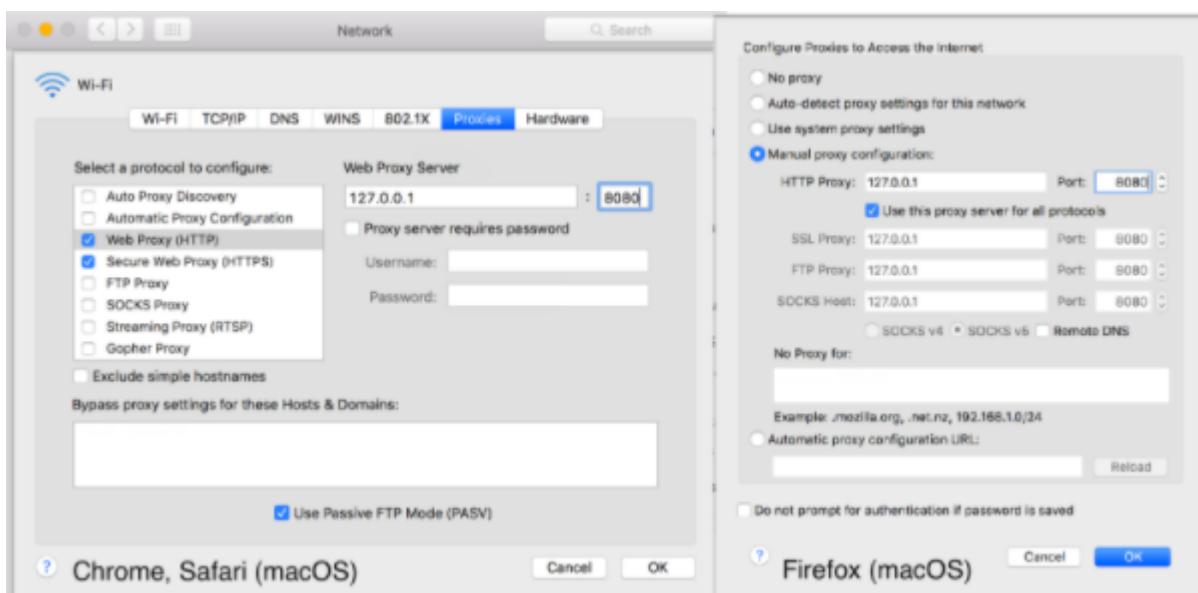
### 3. Чтобы настроить браузер:

**Chrome** : кнопка меню -> Настройки -> Показать дополнительные настройки ... -> Сеть -> Изменить настройки прокси-сервера

**Safari** : Настройки -> Дополнительно -> Прокси -> Изменить настройки ...

**Firefox** : Кнопка меню -> Настройки -> Дополнительно -> Сеть -> Соединение -> Настройки ..

### 4. Например, вы можете использовать localhost 127.0.0.1. Измените порт на порт в HTTP (S) Script Recorder.



Если у вас возникли проблемы при завершении этого этапа, убедитесь, что у вас нет сторонних плагинов, которые могут управлять настройками прокси-сервера вашего браузера. Если вы это сделаете, например, Hola VPN, настройки прокси-сервера будут недоступны в меню вашего браузера.

5. Нажмите кнопку «Пуск», которая находится внизу страницы «Тестер сценариев тестирования HTTP (S)», и просмотрите рабочий процесс веб-приложения, который вы хотите протестировать. Когда вы вернетесь в JMeter, вы увидите все захваченные запросы из своего браузера.

## Запись сценария с помощью прокси-рекордера JMeter

JMeter также позволяет вам вручную настраивать рабочее пространство. Это сложнее, но вы можете сделать скрипты подходящими для ваших конкретных потребностей.

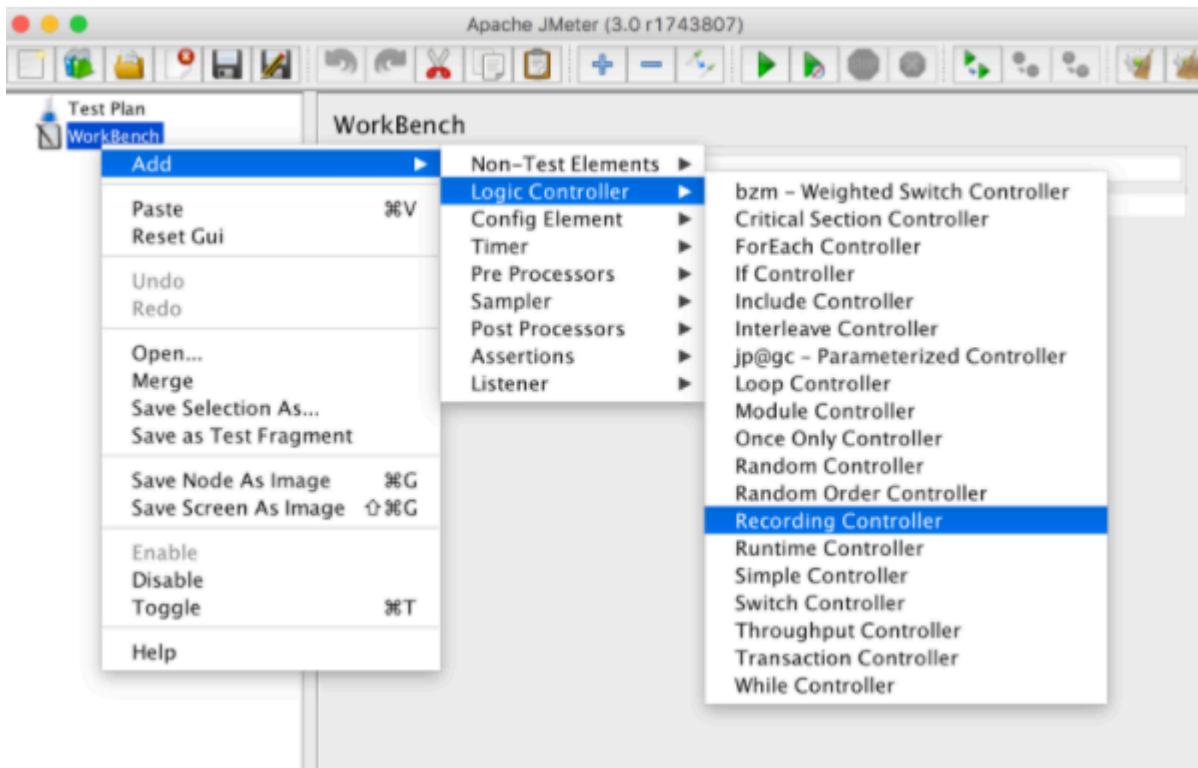
### Настройте свой прокси-сервер

1. Настройте свой браузер, как описано в главе 1.

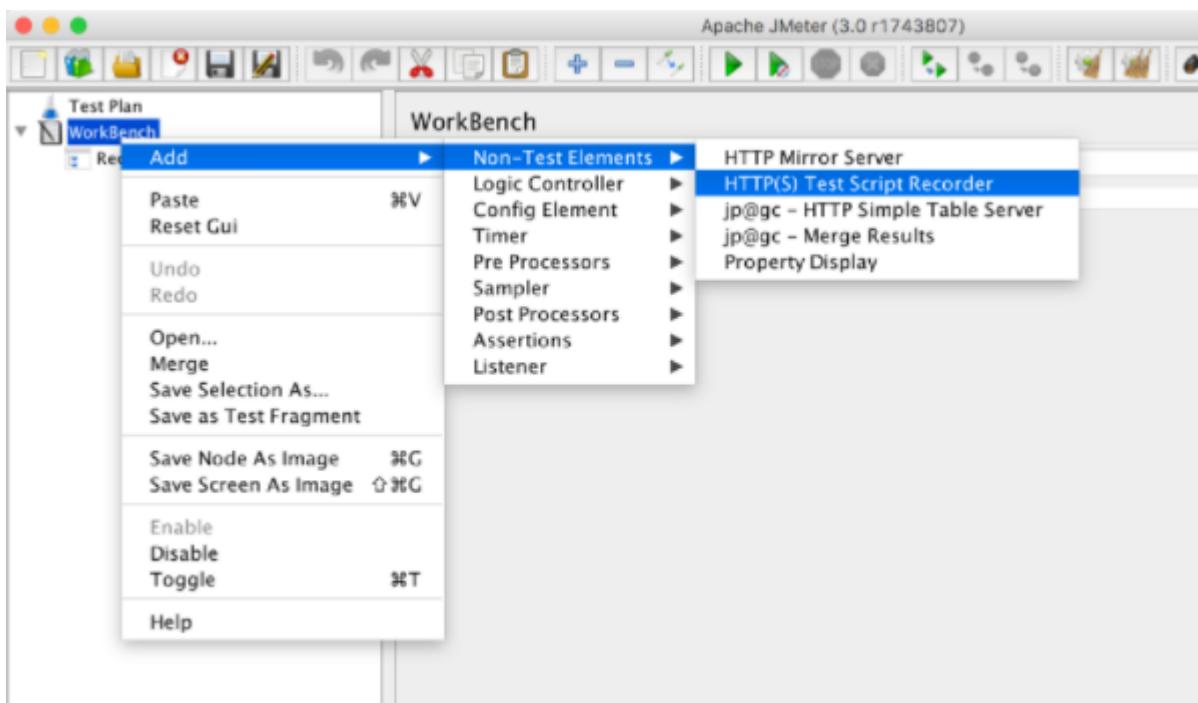
### Настроить JMeter

Развертка «WorkBench» может использоваться как временное рабочее пространство для создания сценариев. Имейте в виду, что записи, добавленные в этот раздел, не будут сохранены в плане тестирования. Поэтому, если вы хотите повторно использовать одну и ту же конфигурацию записи в будущем, вам нужно будет скопировать и вставить ее в раздел «План тестирования».

2. Добавьте «Recording Controller» в «WorkBench»: щелкните правой кнопкой мыши на «WorkBench» -> «Добавить» -> «Logic Controller» -> «Recording Controller» -



- Добавьте «тестовый скрипт сценария HTTP (S)» в тот же «WorkBench»: щелкните правой кнопкой мыши на «WorkBench» -> «Добавить» -> «Не тестируемые элементы» -> «HTTP (S) Test Script Recorder»

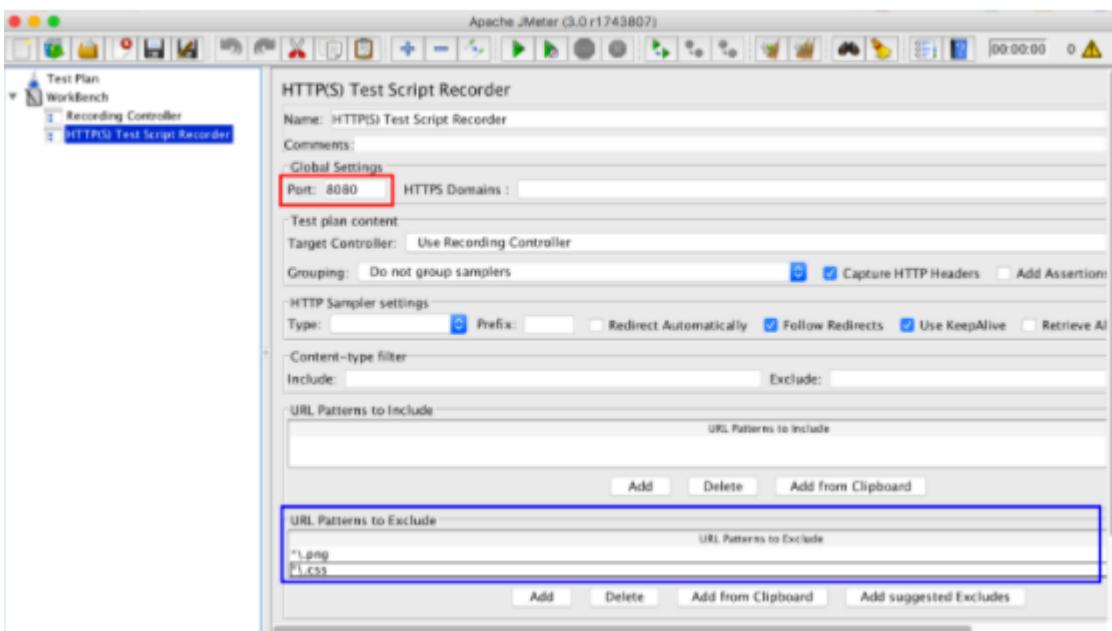


- На странице конфигурации «HTTP (S) Test Script Recorder» в «Global Settings: Port» вам нужно поместить тот же порт, который указан в настройке прокси-сервера вашего браузера, например 8080.
- Если вы хотите исключить запросы к определенным ресурсам, вы можете использовать раздел «Шаблоны URL для исключения». Это может быть полезно, если

вы хотите включить только те типы контента, которые хотите запросить (например, \*.html, \*.php и т. д.) Или исключить типы контента, который вы не хотите запрашивать (например, \*.jpg, \*.png, \*.js и т. д.).

Когда мы будем использовать это? Например, при записи сценария, вызывающего стороннее приложение или при тестировании сценария на стороне сервера, вы можете не захотеть загружать активы, поскольку они могут загромождать ваши тесты и потреблять полосу пропускания, или вы можете записать определенные запросы от определенный путь.

Наиболее распространенные шаблоны исключений: «.. .png „.. .jpg „.. .gif „.. .css „.. .js». Вы также можете комбинировать разные модели вместе. Этот комбинированный шаблон должен избавиться от всех избыточных запросов, которые могут отвлечь вас от важных: «.. (BMP | CSS | JS | GIF | ICO | JPE г |? PNG | SWF | Уофф)»



6. С другой стороны, вы можете подражать полному поведению браузера и включать загрузку всех ресурсов. В этом случае нет необходимости исключать шаблоны URL. Имейте в виду, что браузер загружает все внедренные ресурсы с запрашиваемой страницы и имеет механизм кэширования, который может повлиять на производительность результатов.

В этом случае рекомендуется загрузить все встроенные ресурсы в скрипт: Щелкните правой кнопкой мыши «План тестирования» -> «Добавить» -> «Элемент конфигурации» -> «Настройки HTTP-запроса» -> «Дополнительно» -> выберите Установите флажок «Получить все встроенные ресурсы».

7. Чтобы сделать JMeter более похожим на реальный браузер, рекомендуется добавить «HTTP Cache Manager», который позволяет имитировать функциональность кэширования браузера в ваших тестах производительности. Щелкните правой кнопкой мыши «План тестирования» -> «Добавить» -> «Элемент конфигурации» -> «HTTP

Cache Manager».

8. Теперь нажмите кнопку «Пуск», которая находится в нижней части страницы «Протокол сценариев тестирования HTTP (S)», и просмотрите рабочий процесс веб-приложения, который вы хотите протестировать. Когда вы вернетесь в JMeter, вы должны увидеть все захваченные запросы из своего браузера под «Recording Controller».

Apache JMeter (3.0 r1743807)

Test Plan

WorkBench

Recording Controller

- 19 /
- 24 /login
- 20 /reserve.php
- 21 /purchase.php**
- 22 /confirmation.php
- 23 /home

HTTP(S) Test Script Recorder

**HTTP Request**

Name: 21 /purchase.php

Comments:

Web Server

Server Name or IP: blazedemo.com

Port Number:

HTTP Request

Implementation:

Protocol [http]: http

Path: /purchase.php

Parameters

Name:	
flight	43
price	472.5
airline	Virgin
fromPort	Paris
toPort	Buenos

Send Parameters

Detail Add Add from

Proxy Server

Server Name or IP:

Port Number:

## Запись сценариев производительности для мобильных устройств

JMeter также может использоваться для регистрации тестирования производительности мобильных устройств. Запись мобильных сценариев очень похожа на запись сценариев веб-приложений.

## Настроить JMeter

1. Настройте «Шаблоны JMeter», как указано в главе 1.

## Настройте свой мобильный телефон

После того, как будет подготовлена конфигурация JMeter, в том числе элемент записи JMeter «HTTP (S) Test Script Recording» на указанном порту, вы можете настроить свой мобильный телефон на отправку запроса на веб-приложение, которое вы тестируете через прокси-сервер JMeter.

### 2. iOS :

- Настройка -> Wi-Fi
- Нажмите на подключенную сеть
- Перейдите в раздел конфигурации HTTP PROXY
- Перейдите на вкладку «Руководство»
- Задайте IP-адрес компьютера. Приложение JMeter запущено в разделе «Сервер».
- Установите порт, указанный в «Запись тестового сценария HTTP (S)» в разделе «Порт»

### Android:

- Настройка -> Wi-Fi
- Длинный клик по подключенной сети и нажмите кнопку «Изменить сеть»
- Установите флагок «Дополнительные параметры»
- Установите для параметра «Прокси» значение «Вручную»
- Установите «Имя хоста прокси» в качестве IP-адреса вашего компьютера и «Прокси-порт», как указано в конфигурации «Проверка тестового сценария HTTP (S)» в разделе «Порт»,
- Нажмите «Сохранить».

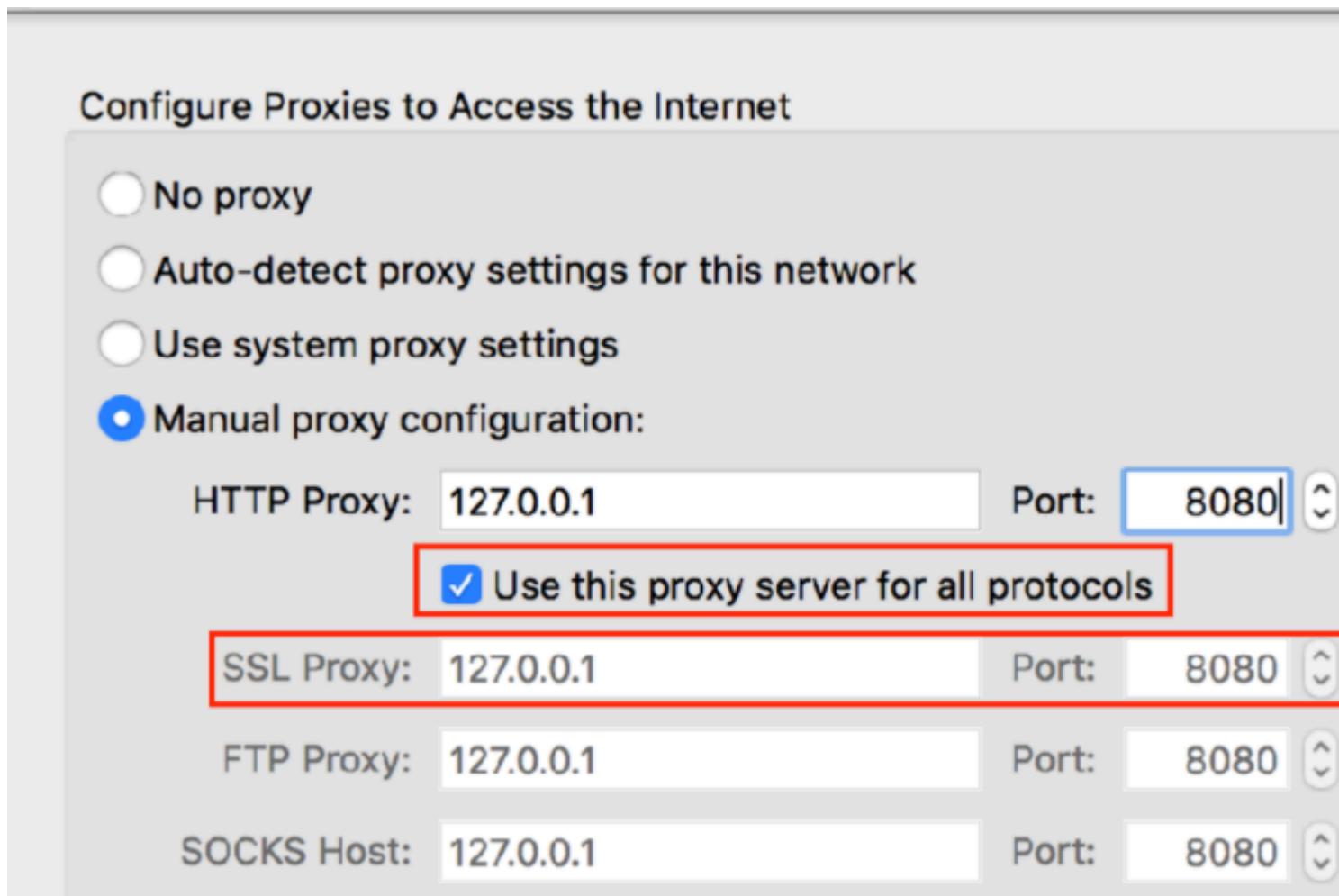
3. Теперь вы можете запустить приложение на своем мобильном устройстве. Запросы будут записаны на JMeter.

## Запись трафика HTTPS

Если ваше веб-приложение использует SSL-шифрование, вам нужно захватить HTTPS-трафик вместо HTTP. Для записи трафика HTTPS с помощью JMeter вам необходимо настроить SSL-сертификаты.

### Настройте свой SSL-прокси

- Убедитесь, что прокси-сервер SSL настроен так же, как настроен HTTP-прокси:



### Настройте JMeter

- Запустите запись сценария, используя функцию «JMeter Recording Template», как описано в примере «Запись сценария с помощью функции шаблона JMeter».
- После открытия веб-приложения вы увидите сообщение о незащищенном соединении. Чтобы продолжить, вам просто нужно принять сертификат Jummer dummy:
  - Нажмите «Дополнительно»
  - Нажмите «Добавить исключение ...».
  - Снимите флагок «Постоянное сохранение этого исключения»
  - Нажмите «Подтвердить исключение безопасности».



## Your connection is not secure

The owner of **example.com** has configured their website improperly. To protect your information from being stolen, Firefox has not connected to this website.

[Learn more...](#)

[Go Back](#)

[Advanced](#)



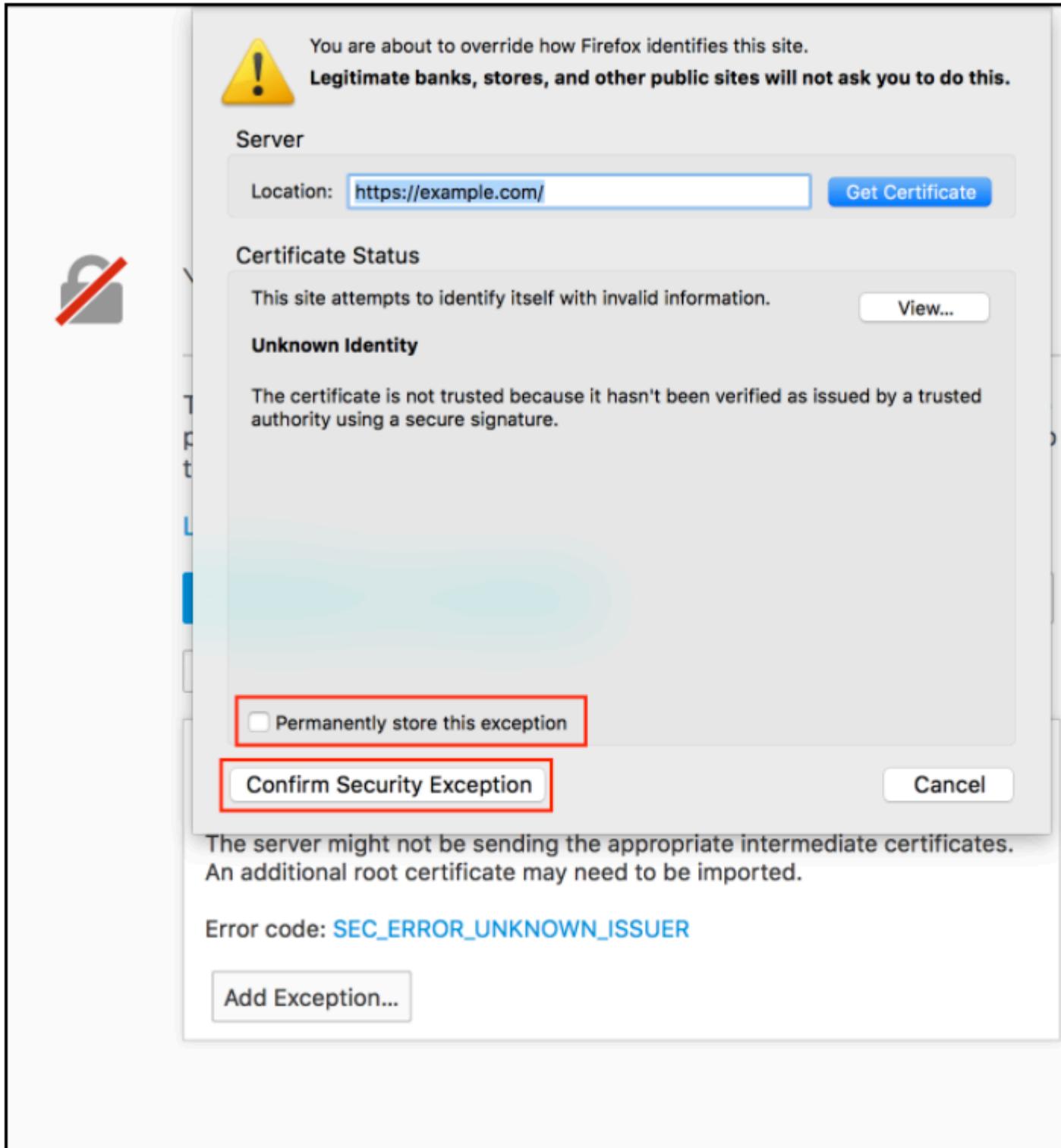
Report errors like this to help Mozilla identify misconfigured sites

example.com uses an invalid security certificate.

The certificate is not trusted because the issuer certificate is unknown.  
The server might not be sending the appropriate intermediate certificates.  
An additional root certificate may need to be imported.

Error code: [SEC\\_ERROR\\_UNKNOWN\\_ISSUER](#)

[Add Exception...](#)



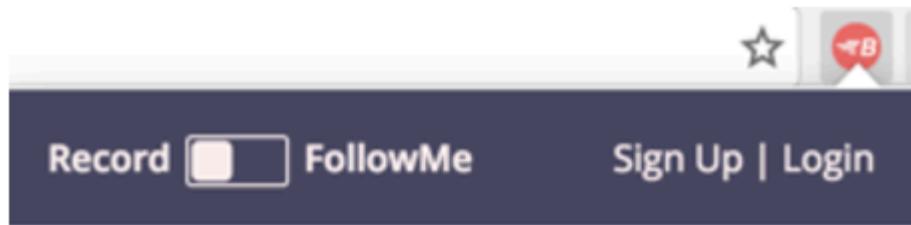
4. Если вы видите «Этот сайт содержит действительную, подтвержденную идентификацию. Не нужно добавлять исключение.», Вы должны очистить историю браузера для своего приложения, включая файлы cookie, кеш, данные оффлайнового веб-сайта. Затем повторите те же шаги.

Этот подход также работает для записи мобильных сценариев, поскольку сертификат JMeter необходимо установить только на хост, который используется для запуска JMeter.

## Запись скриптов с расширением Chrome BlazeMeter

До сих пор мы рассмотрели основные способы записи тестовых сценариев. Но одним из самых быстрых и простых способов записи сценариев производительности, который также является бесплатным, является использование расширения [BlazeMeter Recorder Chrome](#). Эти записи можно запустить в JMeter или в BlazeMeter.

Причина, по которой расширение является настолько полезным, заключается в том, что он позволяет записывать сценарии производительности из вашего браузера без необходимости настройки прокси-сервера.



ENTER THE NAME OF THE TEST

Concurrency

50

Load Origin

User Agent

Include Filter Pattern \*

Advanced Options [Help](#)

Чтобы создать новый скрипт производительности:

1. Откройте рекордер из Chrome
2. Введите тестовое имя в верхнем поле
3. Начните запись, нажав на кнопку записи, в форме круга, и выполните действия, которые вы хотите записать. Все ваши запросы будут сняты. Расширение Chrome Blazemeter также поддерживает запись трафика HTTPS.

- После завершения записи нажмите кнопку остановки в форме квадрата. Вы также можете приостановить запись, а затем возобновить, а также отредактировать ее, в формате .jmx или JSON или в облаке.
- Экспортируйте свою запись - для запуска теста в JMeter, экспортируйте в формат .jmx, нажав кнопку .jmx. Чтобы запустить тест в BlazeMeter, нажмите «play».

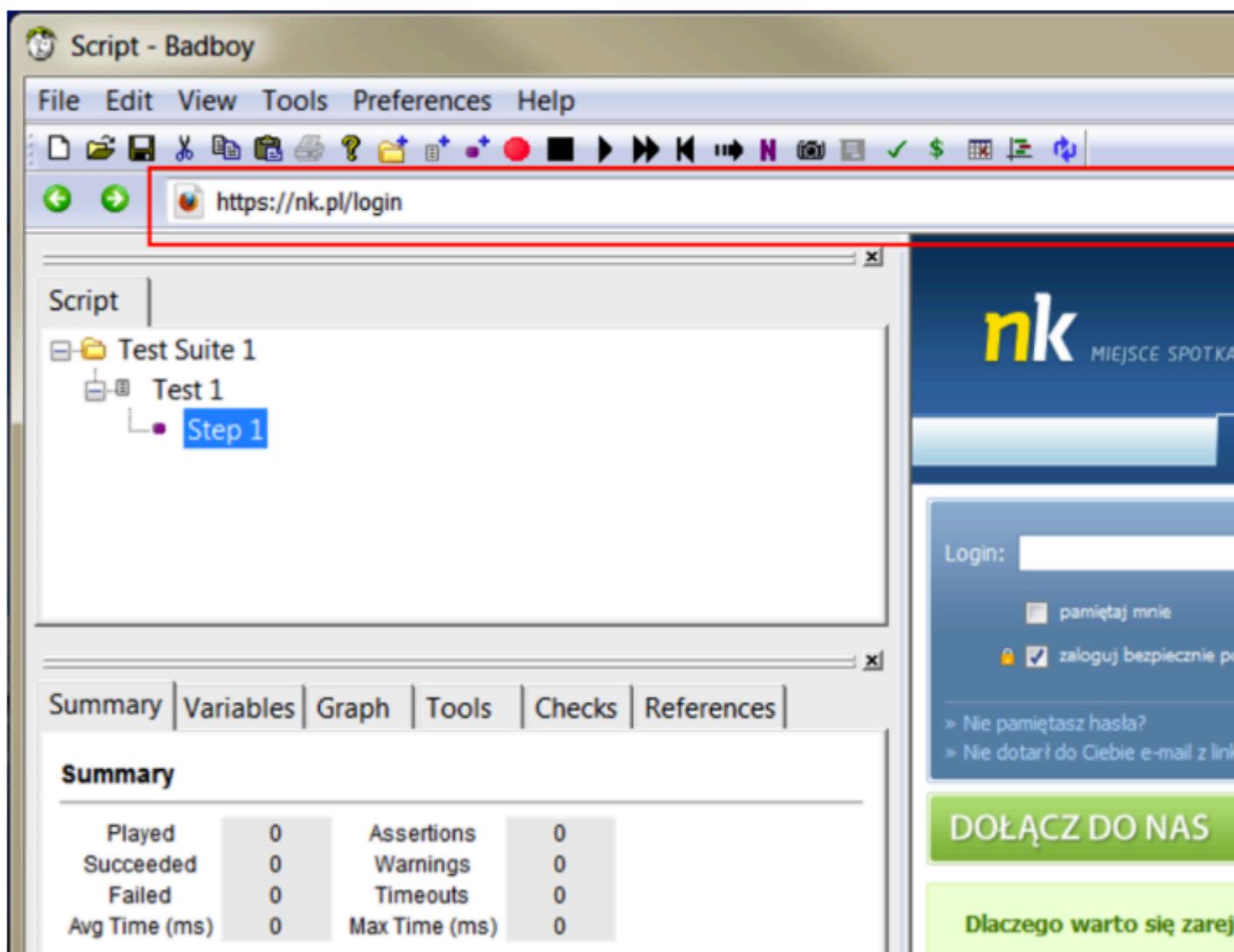
Для получения дополнительной информации см. [Здесь](#).

## Запись скриптов с помощью BadBoy

Еще один полезный инструмент для записи третьей стороны - BadBoy. Однако он работает только для ОС Windows.

Чтобы создать новый скрипт производительности:

- Установите BadBoy [здесь](#)
- Ведите тестовый URL в адресной строке



- Нажмите кнопку записи, сформированную как красный круг, и выполните действия, которые вы хотите захватить.

#### 4. Экспорт сценария в JMeter - Файл -> Экспорт в JMeter

Для получения дополнительной информации см. [Здесь](#).

Использование рекордера сценариев производительности - отличный способ избежать рутинных задач и по-прежнему получать лучшие тестовые сценарии. После записи настройте тест на количество виртуальных пользователей, которые вы хотите протестировать, а также дополнительные тестовые конфигурации, запустите свой тест и проанализируйте результаты, чтобы определить ошибки и узкие места и охарактеризовать тенденции, которые показывают вам здоровье вашей системы.

Прочтайте Apache JMeter: запись сценария сценария онлайн:

<https://riptutorial.com/ru/jmeter/topic/8798/apache-jmeter--запись-сценария-сценария>

# глава 4: Параметризация Apache JMeter

## Вступление

Параметризация - это создание разных наборов данных для разных пользователей в одном тестовом сценарии. Например, запуск нескольких пользователей с разными учетными данными в одном скрипте. Это делает его одним из основных аспектов создания тестов производительности.

## Examples

### Параметрирование с использованием внешних файлов

Один из распространенных способов параметризации ваших сценариев производительности - использовать CSV-файл. Лучшим примером использования входных файлов CSV является процесс входа в систему. Если вы хотите протестировать приложение у разных пользователей, вам необходимо предоставить список учетных данных пользователя.

Предположим, что у нас есть запрос на вход, который работает для одного конкретного пользователя:

The screenshot shows the Apache JMeter interface version 3.0 r1743. The left sidebar displays the 'Test Plan' tree, which includes a 'Login using CSV file' item containing a 'Login request' element, an 'HTTP Header Manager', and other options like 'View Results Tree' and 'WorkBench'. The main panel is titled 'HTTP Request' and contains the configuration for this specific request. The 'Web Server' section has 'Server Name or IP:' set to 'localhost' and 'Port Number:' left blank. The 'HTTP Request' section shows 'Implementation:' as 'Apache HTTP Client' (selected via a dropdown), 'Protocol [http]:' set to 'http', and 'Path:' set to '/login'. There are three checked checkboxes: 'Redirect Automatically', 'Follow Redirects', and 'Use KeepAlive'. Below these, a 'Parameters' section contains a code block with the value: 

```
1 {"email":"testuser1@test.com", "password":"pass1234567890"}
```

. The 'Proxy Server' section is partially visible at the bottom.

Мы можем легко параметризовать этот запрос с помощью внешнего файла CSV и запуска скрипта для разных пользователей. Чтобы добавить конфигурацию параметризации CSV:

*Щелкните правой кнопкой мыши запрос на вход -> Добавить -> Элемент конфигурации -> Конфигурация набора данных CSV*

The screenshot shows the Apache JMeter 3.0 interface. On the left, the Test Plan tree is visible, containing a 'Login using CSV file' sampler, an 'HTTP Header Manager' and a 'CSV Data Set Config' element under it, and 'View Results Tree' and 'WorkBench' options. The 'CSV Data Set Config' element is selected and highlighted with a blue border. On the right, the 'CSV Data Set Config' configuration dialog is open, displaying various parameters:

- Name: CSV Data Set Config
- Comments:
- Configure the CSV Data Source
  - Filename: [empty input field]
  - File encoding: [empty input field]
- Variable Names (comma-delimited): [empty input field]
- Delimiter (use '\t' for tab): ,
- Allow quoted data?: False
- Recycle on EOF ?: True
- Stop thread on EOF ?: False
- Sharing mode: All threads

Краткое описание параметров конфигурации CSV Data Set Config:

- Имя - имя элемента, которое будет использоваться в дереве JMeter
- Имя файла - имя входного файла. Относительные имена файлов разрешаются на основе пути активного плана тестирования. Абсолютные имена файлов также поддерживаются
- Кодирование файлов - кодирование входного файла, если это не платформа по умолчанию
- Переменные имена - список разделенных имен переменных, которые будут использоваться в качестве контейнера для анализируемых значений. Если пустым, первая строка файла будет интерпретирована как список имен переменных
- Разделитель - разделитель, который будет использоваться для разделения проанализированных значений из входного файла
- Разрешить цитируемые данные? - true, если вы хотите игнорировать двойные кавычки и позволить таким элементам содержать разделитель.
- Перерабатывать на EOF? - true в случае, если план тестирования файла должен

перебирать файл более одного раза. Он будет инструктировать JMeter перемещать курсор в начале файла

- Остановить поток на EOF? - false в случае итерации цикла над CDC-файлом и true, если вы хотите остановить поток после прочтения всего файла
- Режим обмена:
  - Все потоки - файл разделяется между всеми виртуальными пользователями (по умолчанию)
  - Текущая группа потоков - файл будет открыт один раз для каждой группы потоков
  - Текущий поток - каждый файл будет открыт отдельно для каждого из потоков
  - Идентификатор - все потоки, имеющие один и тот же идентификатор, также имеют один и тот же файл

Давайте создадим файл csv, содержащий разных пользователей с именами и паролями:

```
[→ tempTestFiles cat /tmp/tempTestFiles/TestU  
testuser1@test.com,password1  
testuser2@gmail.com,password2  
testuser3@gmail.com,password3  
testuser4@gmail.com,password4  
testuser5@gmail.com,password5  
testuser6@gmail.com,password6  
testuser7@gmail.com,password7  
testuser8@gmail.com,password8  
testuser9@gmail.com,password9  
testuser10@gmail.com,password10
```

Теперь мы можем использовать этот файл с конфигурацией набора данных CSV. В нашем случае достаточно добавить значения конфигурации «Имя файла» и «Переменные имена»:

The screenshot shows the Apache JMeter interface version 3.0 r174. The left pane displays the Test Plan tree, which includes a 'Login using CSV file' sampler, a 'Login request' sampler, an 'HTTP Header Manager', and a 'CSV Data Set Config' configuration element. The 'CSV Data Set Config' element is selected and highlighted in blue. The right pane is the 'CSV Data Set Config' editor, containing the following configuration settings:

- Name: CSV Data Set Config
- Comments:
- Configure the CSV Data Source
  - Filename: /tmp/tempfile.csv
  - File encoding:
- Variable Names (comma-delimited): email,password
- Delimiter (use '\t' for tab): ,
- Allow quoted data?: False
- Recycle on EOF ?: True
- Stop thread on EOF ?: False
- Sharing mode: All threads

Последним шагом, который мы должны предпринять, является параметризация запроса на вход с переменными CSV. Это можно сделать, заменив исходные значения соответствующими переменными из поля конфигурации «Переменные имена» в CSV Data Set Config, например:

The screenshot shows the Apache JMeter interface version 3.0 r174. On the left, the Test Plan tree view displays a 'Login using CSV file' plan containing a 'Login request' sampler, an 'HTTP Header Manager', and a 'CSV Data Set Config'. Below the tree are 'View Results Tree' and 'WorkBench' buttons. The main panel on the right is titled 'HTTP Request' and contains the configuration for the selected 'Login request' sampler. The 'Web Server' section has 'Server Name or IP: localhost'. The 'HTTP Request' section shows 'Implementation:' dropdown, 'Path: /login', and checkboxes for 'Redirect Automatically' (unchecked), 'Follow Redirects' (checked), and 'Advanced' (checked). A preview window at the bottom shows the JSON payload: '1 {"email": "\${email}", "password": "\${password}"}'. The 'email' and 'password' fields are highlighted with red boxes.

Если мы запустим наш тестовый скрипт, JMeter заменит эти переменные на значения из файла TestUsers.csv. Каждый виртуальный пользователь JMeter получит учетные данные из следующей строки файла csv.

Запрос на вход первого пользователя:

Apache JMeter (3.0 r174)

The screenshot shows the Apache JMeter interface. The top menu bar displays "Apache JMeter (3.0 r174)". Below the menu is a toolbar with various icons for file operations like Open, Save, and Print, as well as other tools like a calculator and a clipboard. The left sidebar contains a tree view of the "Test Plan". Under "Test Plan", there is a "Login using CSV file" node, which further contains "Login request", "HTTP Header Manager", and "CSV Data Set Config" sub-nodes. The "View Results Tree" node is also listed under "Test Plan" and is highlighted with a blue selection bar. To the right of the tree view is the "View Results Tree" configuration panel. It includes fields for "Name" (set to "View Results Tree"), "Comments", and options for "Write results to file / Read from file" with a "Filename" input field and a "Browse..." button. Below this is a "Search:" input field. The main workspace shows a "Text" listener result tree. The tree has a single root node labeled "Login request" with a green checkmark icon. This node has ten child nodes, all of which are also labeled "Login request" with green checkmarks. At the bottom of the result tree panel is a checkbox labeled "Scroll automatically?". To the right of the main workspace, there is a partial view of the request details, showing a POST method with JSON body parameters and headers.

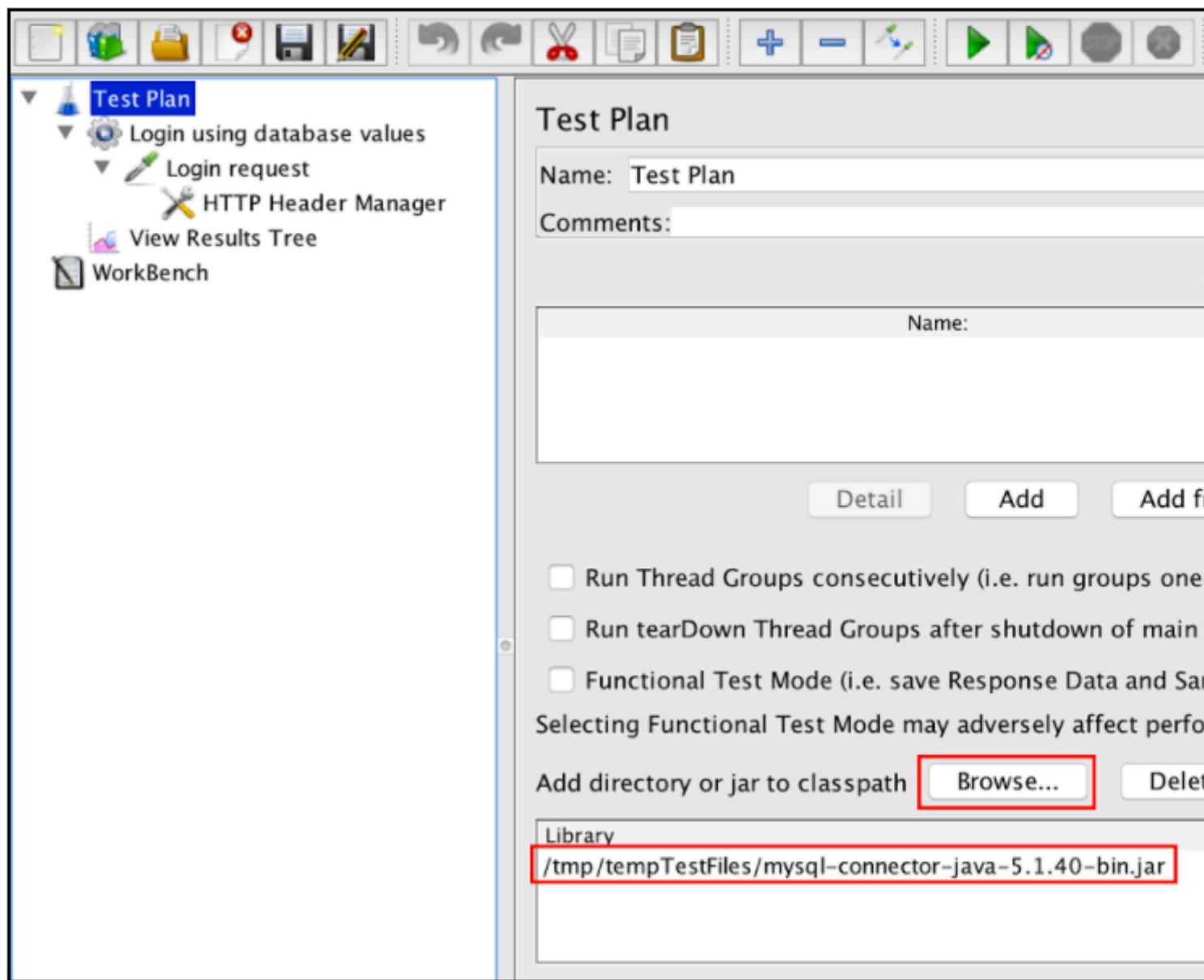
Запрос на вход второго пользователя:

The screenshot shows the JMeter interface with a test plan containing a 'View Results Tree' listener. The 'View Results Tree' dialog is open, displaying a list of 'Login request' entries, each with a green checkmark icon. The 'Text' tab is selected. A checkbox at the bottom left of the dialog is labeled 'Scroll automatically?'. The right side of the interface shows the detailed configuration for the 'View Results Tree' listener, including fields for 'Name', 'Comments', 'Write results to file / Read from file', 'Filename', and a 'Browse...' button. The 'Search:' field is empty.

## Параметрирование с использованием баз данных

Другим способом параметризации ваших сценариев производительности является использование данных базы данных через JDBC. JDBC - это интерфейс прикладного программирования, который определяет, как клиент может получить доступ к базе данных.

Прежде всего, загрузите драйвер JDBC в свою базу данных (обратитесь к поставщику базы данных). Например, драйвер mysql можно найти здесь. Затем вы можете добавить его, добавив файл .jar в план тестирования, используя следующую форму:



Но лучше добавить Jar-файл в папку lib и перезапустить JMeter.

После этого настройте соединение с базой данных, используя элемент «Конфигурация соединения JDBC». Пример: Щелкните правой кнопкой мыши на группе потоков -> Добавить -> Элемент конфигурации -> Конфигурация соединения JDBC

## JDBC Connection Configuration

Name: JDBC Connection Configuration

Comments:

Variable Name Bound to Pool

Variable Name:

Connection Pool Configuration

Max Number of Connections: 10

Max Wait (ms): 10000

Time Between Eviction Runs (ms): 60000

Auto Commit: True

Transaction Isolation: DEFAULT

Connection Validation by Pool

Test While Idle: True

Soft Min Evictable Idle Time(ms): 5000

Validation Query: Select 1

Database Connection Configuration

Database URL:

JDBC Driver class:

Username:

Password:

Параметры конфигурации подключения JDBC:

- Имя - имя конфигурации подключения, которое будет отображаться в дереве групп потоков
- Variable Name - имя, которое будет использоваться как уникальный идентификатор для соединения db (можно использовать несколько соединений, и каждый из них будет привязан к другому имени)

- Максимальное количество подключений - максимальное количество подключений, разрешенных в пуле подключений. В случае 0 каждый поток будет иметь свой собственный пул с одним соединением в нем
- Max Wait (ms) - пул выдаёт ошибку, если указанный тайм-аут превышен при подключении db
- Время между прогонами (ms) - количество миллисекунд для паузы между прогонами потока, который выдает неиспользуемые соединения из пула db
- Auto Commit - да, чтобы включить автоматическую фиксацию для связанных подключений db
- Test While Idle - проверять соединения на холостом ходу перед обнаружением эффективного запроса. Дополнительные сведения: BasicDataSource.html # getTestWhileIdle
- Soft Min Evictable Idle Time (ms) - период времени, в течение которого указанное соединение может простоять в пуле db, прежде чем оно может быть выведено. Подробнее: BasicDataSource.html # getSoftMinEvictableIdleTimeMillis
- Запрос проверки - запрос на проверку работоспособности, который будет использоваться для проверки того, отвечает ли база данных
- URL-адрес базы данных - строка подключения JDBC для базы данных. См. Здесь примеры
- Класс JDBC Driver - соответствующее имя класса драйвера (для каждого db). Например, «com.mysql.jdbc.Driver» для MySQL db
- Имя пользователя - имя пользователя базы данных
- Пароль - пароль базы данных (будет сохранен незашифрованным в плане тестирования)

В нашем случае нам нужно только установить обязательные поля:

- Имя переменной Bound to Pool.
- URL базы данных
- Класс JDBC Driver
- имя пользователя
- пароль

Остальные поля на экране можно оставить по умолчанию:

The screenshot shows the JMeter interface with a test plan named "JDBC Connection Configuration". The plan includes a "Login using database values" step which contains a "JDBC Connection Configuration" sub-step. Other steps in the plan are "Login request" and "View Results Tree". A "WorkBench" icon is also present.

**JDBC Connection Configuration**

Name: JDBC Connection Configuration

Comments:

Variable Name Bound to Pool

Variable Name: UsersDbConnection

Connection Pool Configuration

Max Number of Connections: 10

Max Wait (ms): 10000

Time Between Eviction Runs (ms): 60000

Auto Commit: True

Transaction Isolation: DEFAULT

Connection Validation by Pool

Test While Idle: True

Soft Min Evictable Idle Time(ms): 5000

Validation Query: Select 1

Database Connection Configuration

Database URL: jdbc:mysql://localhost:3306/statsell

JDBC Driver class: com.mysql.jdbc.Driver

Username: root

Password:

Предположим, что мы храним тестовые учетные данные пользователя в базе данных:

The screenshot shows a Java IDE interface with a code editor at the top containing a SQL query:

```
1 select email,password from
```

Below the code editor is a table with two columns: "email" and "password". The table contains 10 rows of data:

email	password
testuser1@test.com	password1
testuser2@gmail.com	password2
testuser3@gmail.com	password3
testuser4@gmail.com	password4
testuser5@gmail.com	password5
testuser6@gmail.com	password6
testuser7@gmail.com	password7
testuser8@gmail.com	password8
testuser9@gmail.com	password9
testuser10@gmail.com	password10

Теперь, когда настроено соединение с базой данных, мы можем добавить сам запрос JDBC и использовать его запрос для получения всех учетных данных из базы данных: *Щелкните правой кнопкой мыши на группе потоков -> Добавить -> Пример -> Запрос JDBC*

Используя запрос 'Select Statement' и 'Variable Names', мы можем проанализировать ответ на пользовательские переменные.

The screenshot shows the JMeter interface with a test plan containing a JDBC Connection Configuration and a Get credentials request. The Get credentials request is selected. On the right, the JDBC Request configuration dialog is open, showing the following details:

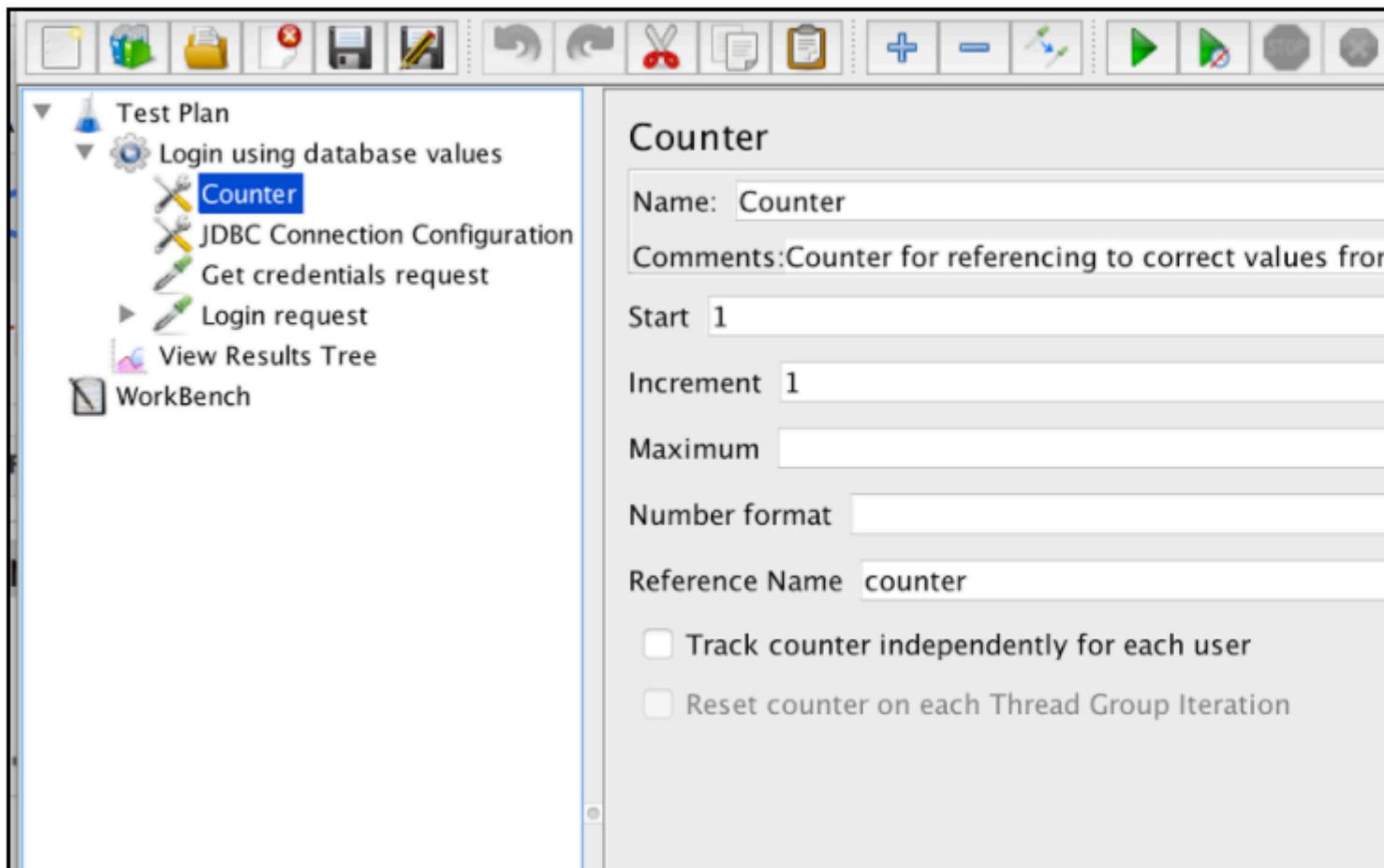
- Name: Get credentials request
- Comments:
- Variable Name Bound to Pool: UsersDbConnection
- SQL Query:

```
1 select email,password from users;
```
- Query Type: Select Statement
- Parameter values:
- Parameter types:
- Variable names: email,password (highlighted with a red box)
- Result variable name:
- Query timeout (s):
- Handle ResultSet: Store as String

Теперь мы будем иметь переменные JMeter, которые могут быть использованы далее в последующих запросах. Указанные переменные будут созданы с добавочным суффиксом (email\_1, email\_2, email\_3 ...).

Чтобы использовать эти переменные в «Запросе на вход», нам нужно добавить счетчик, который будет использоваться для доступа к правильным значениям из ответа на запрос.

JDBC. Чтобы добавить элемент «Счетчик» в JMeter: щелкните правой кнопкой мыши на группе потоков -> Добавить -> Элемент конфигурации -> Счетчик



После этого мы можем обновить «запрос на вход» с помощью функции \_\_V. Это возвращает результат оценки выражения имени переменной и может использоваться для оценки ссылок вложенных переменных:

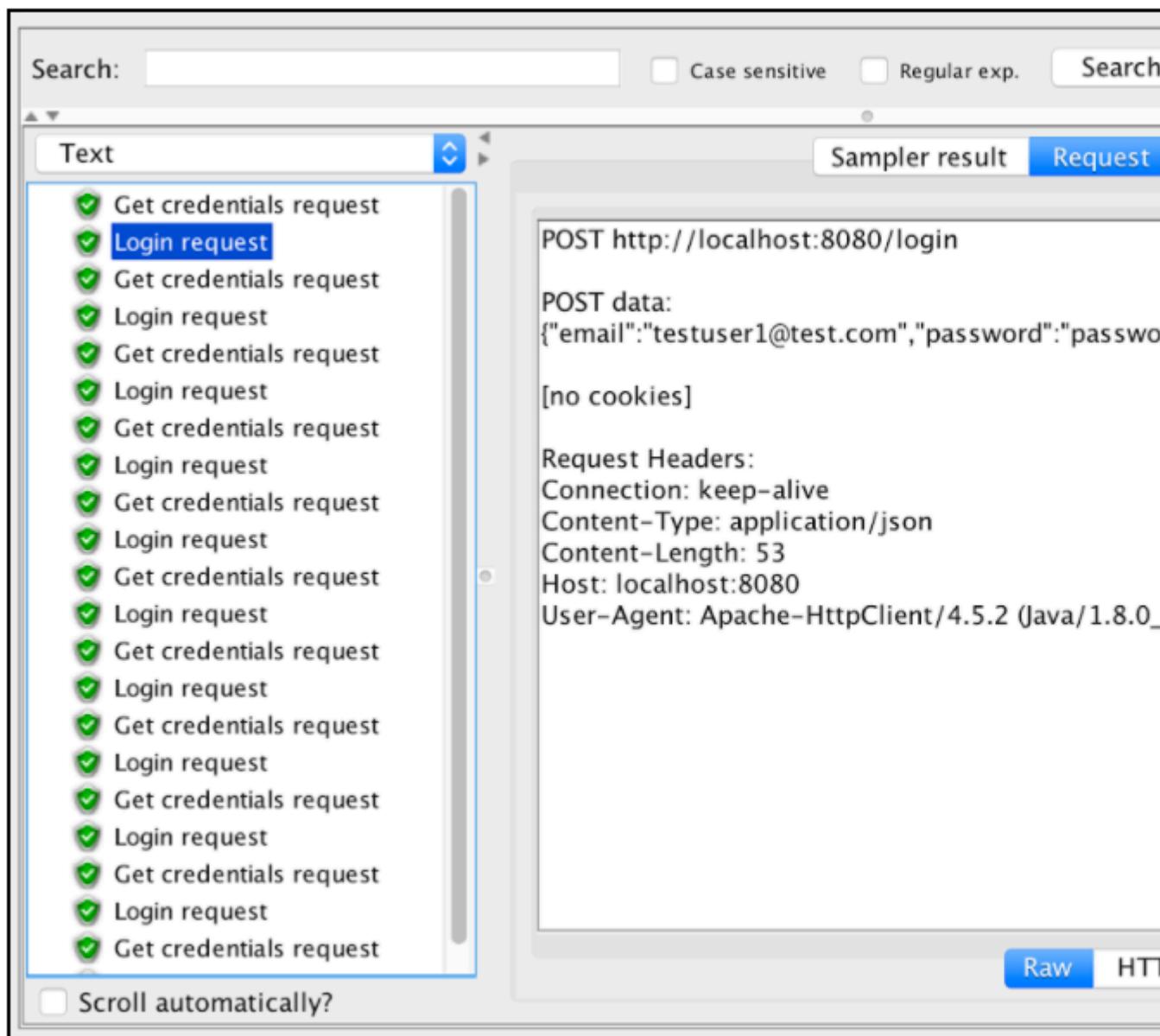
The screenshot shows the JMeter interface with a test plan named "Login using database values". The "Login request" step is selected, highlighted in blue. The right panel displays the "HTTP Request" configuration for this step. The "Name" field is set to "Login request". The "Web Server" section shows "Server Name or IP: localhost". The "HTTP Request" section shows "Implementation: [dropdown]" and "Path: /login". Under "Advanced" settings, "Follow Redirects" is checked. The "Script" tab contains the following JSON payload:  
1 {"email":"\${\_\_V(email\_\${counter})}"}, "password": "1234567890"}

Указанной конфигурации достаточно, чтобы использовать значения базы данных для запуска скрипта для разных пользователей:

The screenshot shows the JMeter interface with the following details:

- Test Plan Tree:** A tree view on the left showing the test plan structure:
  - Test Plan
  - Login using database values
    - Counter
    - JDBC Connection Configuration
    - Get credentials request
    - Login request
  - View Results Tree
- View Results Tree Panel:** A right-hand panel titled "View Results Tree" with the following fields:
  - Name: View Results Tree
  - Comments:
  - Write results to file / Read from file
  - Filename: [empty]
  - Search: [empty]  Case sensitive
- Result Data View:** A large central area displaying a list of results. The list is titled "Text" and contains 20 entries, each with a green checkmark and the text "Get credentials request".
  - Get credentials request
  - Get credentials request
- Right Panel:** A vertical panel on the right showing a list of email addresses:

email	[redacted]
testuser1@test.c	[redacted]
testuser2@gmail.c	[redacted]
testuser3@gmail.c	[redacted]
testuser4@gmail.c	[redacted]
testuser5@gmail.c	[redacted]
testuser6@gmail.c	[redacted]
testuser7@gmail.c	[redacted]
testuser8@gmail.c	[redacted]
testuser9@gmail.c	[redacted]
testuser10@gmail.c	[redacted]
- Bottom Panel:** A footer panel with a search field and a checkbox labeled "Scroll automatically?".

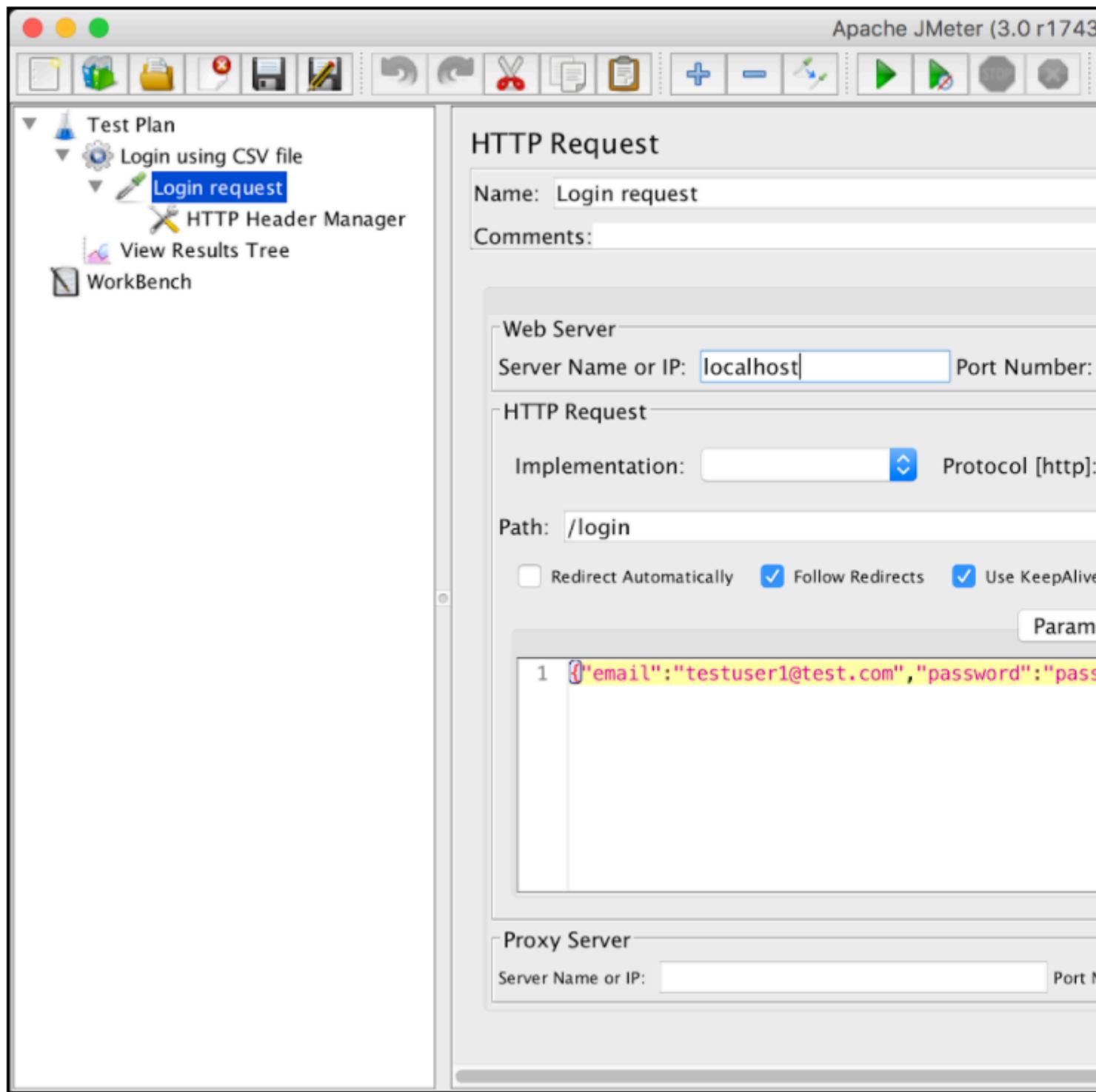


## Параметрирование с использованием плагина Parameterized Controller

Если вам нужно выполнить повторяющуюся последовательность одного и того же действия с разными параметрами, используйте плагин стороннего разработчика «Parameterized Controller» из проекта [JMeter-Plugins](#).

Сначала вам необходимо установить этот плагин, выполнив процедуру установки.

Предположим, что мы хотим параметризовать рабочий процесс входа в систему:



Прежде всего, вам нужно установить плагин «Parameterized Controller», поскольку он не включен в ядро JMeter. Этапы установки этого процесса можно найти здесь.

Давайте переместим «Запрос на вход» в отдельный контроллер и отключим его (щелкните его правой кнопкой мыши и выберите «Отключить»). Это наиболее предпочтительный способ иметь контейнер модулей внутри вашего плана тестирования и избегать использования Workbench как такового. По завершении установки вы можете добавить два контроллера «Parameterized Controller» с разными учетными данными пользователя:  
Щелкните правой кнопкой мыши на группе потоков -> Добавить -> Логический контроллер -> Параметризованный контроллер

The screenshot shows the JMeter interface with a test plan structure on the left and configuration details on the right.

**Test Plan Structure:**

- Test Plan
  - Login using 'Parameterized controller' plugin
    - Parameterized Controller – User 1
    - Parameterized Controller – User 2
  - Reusable Controller
    - Login request
- View Results Tree
- WorkBench

**Configuration Details (jp@gc – Parameterized Controller):**

- Name: Parameterized Controller
- Comments:
- [Help on this plugin](#)

**User Defined Variables:**

- Name: User Defined Variable
- Comments:

**Add New Variable:**

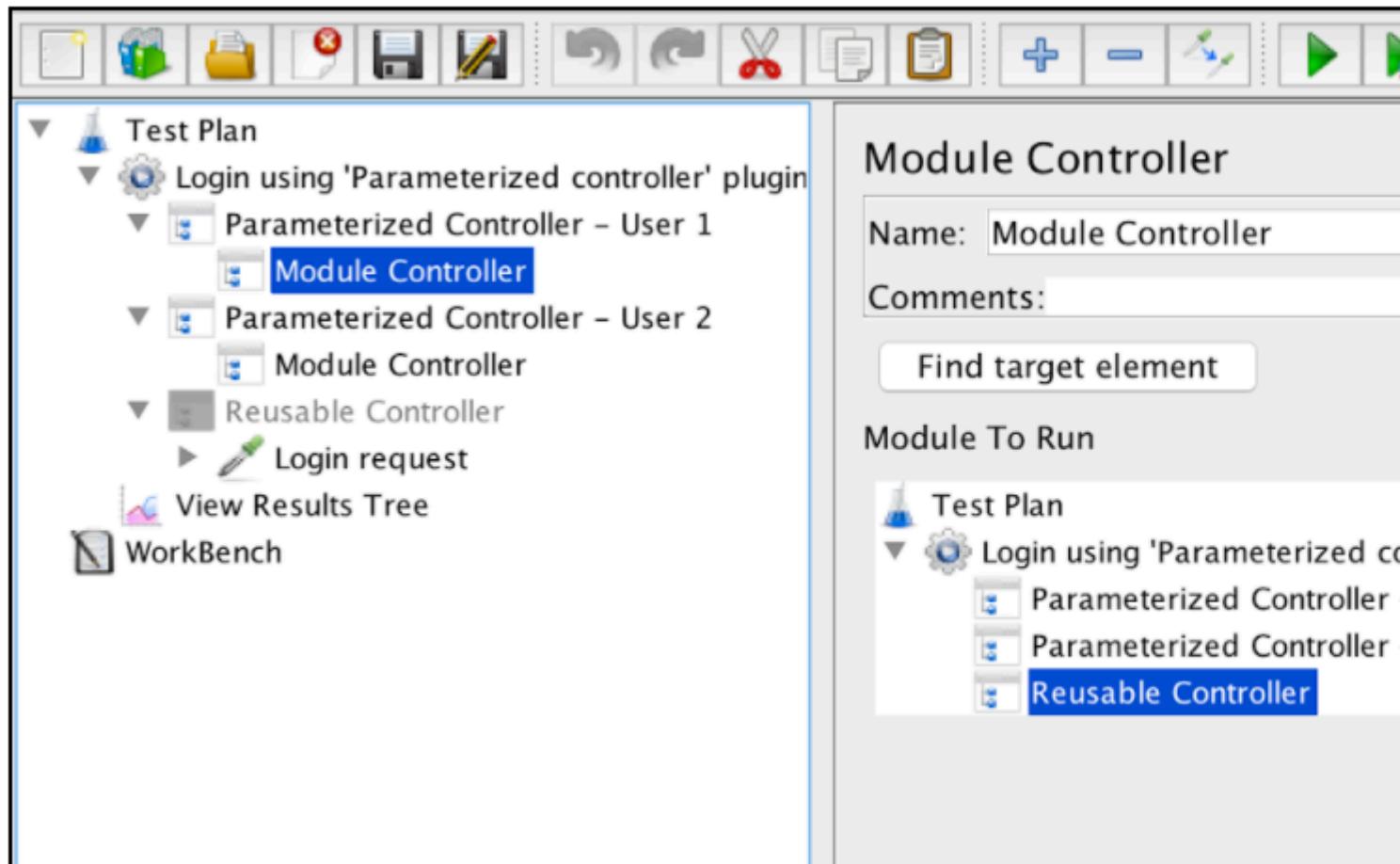
- Name: [Empty input field]
- Detail
- Add

Параметризованные контроллеры содержат раздел «Определенные пользователем переменные», где вы можете указать свои параметры. Поместите учетные данные первого пользователя в первом параметризованном контроллере и вторых учетных данных пользователя во втором параметризованном контроллере.

The screenshot shows the JMeter interface. On the left, the Test Plan tree view displays a structure under 'Test Plan': 'Login using 'Parameterized controller' plugin' which contains 'Parameterized Controller – User 1' and 'Parameterized Controller – User 2'; 'Reusable Controller' which contains 'Login request'; and 'View Results Tree'. Below this is a 'WorkBench' icon. On the right, a dialog box titled 'jp@gc – Parameterized Controller' is open. It shows fields for 'Name: Parameterized Controller' and 'Comments:'. A link 'Help on this plugin' is also present. Below this is another dialog titled 'User Defined Variables' with fields for 'Name: User Defined Variables' and 'Comments:'. Under 'User Defined Variables', there is a table with one row containing 'Name: email' and 'password'. At the bottom of this dialog are 'Detail' and 'Add' buttons, with 'Add' being highlighted with a red box. The overall interface is in light gray with blue highlights for selected items.

Внутри обоих параметризованных контроллеров добавьте ссылки на «Многоразовый контроллер», чтобы вызвать «запрос на вход» с различными параметрами. Это можно сделать так:

*Щелкните правой кнопкой мыши «Параметрированный контроллер» -> «Добавить» -> «Логический контроллер» -> «Контроллер модуля»*



## Module Controller

Name: Module Controller

Comments:

Find target element

## Module To Run

Test Plan

Login using 'Parameterized controller' plugin

- Parameterized Controller – User 1
- Parameterized Controller – User 2
- Reusable Controller

При запуске скрипта вы увидите, что «запрос на вход» запускает каждый из параметризованных контроллеров отдельно. Это может быть очень полезно, если вам нужно запустить скрипт в разных комбинациях входных параметров.

The screenshot shows the Apache JMeter interface. On the left, the 'Test Plan' tree view is expanded to show a 'Parameterized controller' plugin configuration. It includes three controllers: 'Parameterized Controller - User 1' with a 'Module Controller', 'Parameterized Controller - User 2' with a 'Module Controller', and a 'Reusable Controller' containing a 'Login request' sampler. A 'View Results Tree' button is highlighted with a blue border. On the right, the 'View Results Tree' panel is open, showing settings for saving results ('Name: View Results Tree', 'Comments:', 'Write results to file / Read from file', 'Filename:'), a search bar ('Search:'), and a results table titled 'Text' with two entries: 'Login request' and 'Login request'.

Прочтайте Параметризация Apache JMeter онлайн:

<https://riptutorial.com/ru/jmeter/topic/9602/параметризация-apache-jmeter>

# кредиты

S. No	Главы	Contributors
1	Начало работы с Apache JMeter	Aliaksandr Belik, Chulbul Pandey, Community, Kiril S., M Navneet Krishna, Milamber, Naveen, NaveenKumar Namachivayam, RowlandB, UBIK LOAD PACK, Venkatesh Achanta
2	Apache JMeter Correlations	UBIK LOAD PACK, Yuri Bushnev
3	Apache JMeter: запись сценария сценария	UBIK LOAD PACK, Yuri Bushnev
4	Параметризация Apache JMeter	UBIK LOAD PACK, Yuri Bushnev