Практическая работа.

Использование Soap UI в тестировании для тестирования веб-сервисов

Пререквизиты

- 1. Выполнить установку **SOAP UI**. <u>Официальный сайт</u>.
- 2. Имена создаваемых объектов должны быть в формате #Name-Surname#-TypeOfObject.
- 3. Требуемые в заданиях описания оформить в таблицу следующего вида:

Пункт	Описание			
2.2	Вкладка Assertions стала красной, потому что Response не содержит указанного в			
	Assert Content-a			

Задание 1

 Создайте SOAP Project используя следующий WSDL: http://www.webservicex.net/CurrencyConvertor.asmx?WSDL

🌖 So	apUI 4.6.	4		
<u>F</u> ile	<u>T</u> ools	<u>D</u> esktop	<u>H</u> elp	
N	lew SOAI	P Project	Ctrl-N	
N	lew REST	Project	Ctrl+Alt-N	
N	lew Gene	ric Project	Ctrl+Alt+Shift-N	
Import Project			Ctrl-I	
Import Packed Project				
Ir	npo <u>r</u> t Re	mote Projec	ct	
	AU 0	· .		

New SOAP Project New SOAP Project Creates a WSDL/SOAP based Project in this workspace Project Name: NameSurnameSoapProject Initial WSDL: http://www.webservicex.net/CurrencyConvertor.asmx?WSDL Browse... Create Requests: Create sample requests for all operations? Create TestSuite: Creates a TestSuite for the imported WSDL Relative Paths: Stores all file paths in project relatively to project file (requires save) OK Cancel

2. Добавьте в него TestSuite



3. В TestSuite выполните создание TestCase

Projects Projects P 0 NameSunameSoapProject P 1 CurrencyConvertorSoap D						
🗄 NameSurna	meTestSuite					
	Show TestSuite Editor Disable TestSuite	Enter				
	🛃 New TestCase	Ctrl-N				
	F9					
	Launch TestRunner					
	Rename	F2				
	Remove	Delete				
	Move TestSuite Up	Ctrl-Up				
	Move TestSuite Down	Ctrl-Down				
	Export					
I	1 T+ C					

Задание 2

В TestCase своего проекта создайте следующие TestSteps.

TestSteps добавляются с помощью следующих кнопок:



1. Создайте **Request** для конвертации **USD** в **UAH**, для этого подставьте соответствующие значения вместо знаков вопроса

<web:FromCurrency>?</web:FromCurrency> <web:ToCurrency>?</web:ToCurrency>

2. В созданном Request добавьте следующий Assert(проверку)



Property Content > Contains в который занесите **Content**: 0.1 и запустите **Request**. Выполните запись полученного результата в таблицу (Формат таблицы описан в пункте 3 раздела **Пререквизиты**).

- 3. Измените ранее созданный **Assert**, добавив в **Content** получаемое в **Response** значение и запустите его, выполните запись полученного результата в таблицу.
- 4. Добавьте Properties TestStep со следующими Property:
 - From = USD

To = UAH

Result

- 5. Создайте **Property Transfer TestStep** для передачи значений вычисляемых валют из переменных созданных в предыдущем шаге и разместите его перед **Request.**
- 6. В созданный **Property Transfer TestStep** добавьте **Property Transfer** для подстановки параметра **FromCurrency**, для этого заполните **Property Transfer** следующими значениями:

🕨 🕒 n	S				
Source:	NameSurnameProperties	•	Property:	From	-
~~					
Target:	👪 NameSurnameRequest	•	Property:	Request	•
//*[local-name()='FromCurrency']					

7. Запустите созданный **Property Transfer** и выполните запись полученного результата в таблицу.

- 8. Добавьте еще один **Property Transfer** для подстановки параметра **ToCurrency**, по аналогии с предыдущим, запустите его и выполните запись полученного результата, в таблицу
- Создайте Property Transfer TestStep для передачи полученного значения вычисляемых валют в переменную созданную в предыдущем шаге и разместите его после Request, запустите его и выполните запись полученного результата в таблицу.
- 10. Запустите ваш **TestCase**.
- 11. Проверьте содержимое файла, который вы указали в полу **Save to** в 4-м шаге, выполните запись полученного результата в таблицу.
- 12. Поменяйте местами значения переменных From и To в Properties TestStep и запустите ваш TestCase.
- 13. Проверьте содержимое файла, который вы указали в полу **Save to** в 4-м шаге, Выполните запись полученного результата в таблицу.

Задание 3

1. Создайте 2 LoadTest для вашего TestCase



2. Сконфигурируйте их следующим образом:

🚯 NameSurnameLT1			8 6 2 🗵
🕨 🗙 🖻 🚥 诸 💥 🔞	Limit:	5 🖨 Seconds 💌	100%
Threads: 100 🖨 Strategy Simple 💌	Test Delay	1000 Random 0.5	
🙆 NameSurnameLT2			8 - 5 2
🕨 🗶 🖂 🚥 诸 💥 🔞	Limit:	5 🔹 Seconds 💌	100%
Threads: 5 - Strategy Simple -	Test Delay	1000 Random 0.5	

3. Запустите их и проанализируйте результаты, выполните запись полученных результатов, в таблицу.

Задание 4

Теперь давайте попробуем написать некое подобие сервиса которым мы пользовались, для этого воспользуемся возможностями **Soap UI**, а именно **MockService** для этого:

1. Сгенерируйте MockService на основе CurrencyConvertorSoap в контекстном меню CurrencyConvertorSoap.



- 2. Далее откройте **Response 1** который сгенерировался автоматически как ответ на **Request 1** из **CurrencyConvertorSoap.**
- 3. Напишите вместо знака вопроса любое число.
- 4. Запустите созданный MockService.



5. Создайте Request который будет ссылаться на ваш MockService .



- 6. В созданном **Request** вместо знаков вопросов поставьте нужные параметры и запустите его.
- 7. Выполните запись полученного результата в таблицу.

Задание 5

В таком варианте наш **MockService** при любых вводных параметрах будет выдавать одинаковое значение, давайте это изменим:

1. Добавьте в свой Response следующий скрипт на Groovy:



context.setProperty("rate", Math.random() *10)

- 2. Вместо произвольного числа добавленного в предыдущем задании укажите ссылку на переменную **rate** из скрипта (\${rate})
- 3. Запустите несколько раз **Request** созданный в предыдущем задании.
- 4. Выполните запись полученных результатов в таблицу

Задание 6

 Сделайте чтобы созданный во втором задании TestCase работал с вашим MockService, для этого в адресной строке Request вашего TestCase вставьте адрес вашего MockService (узнать вы его можете из адресной строки Request созданного в 5-м шаге 4-го задания)
 NameSurnameRequest

 *
 IP

 IP
 IP

 IP
 IP

 IP
 IP

 Измените скрипт Response таким образом чтобы Response возвращал реальное значение в зависимости от входных данных Request.
 Получить эти значения как переменные можно следующим образом:

def holder = new com.eviware.soapui.support.XmlHolder(mockRequest.requestContent)

def from = holder.getNodeValue("//web:FromCurrency/text()")

def to = holder.getNodeValue("//web:ToCurrency/text()")

Сохраните проект