

使用Python实现 Selenium自动化测试

SELENIUM WITH PYTHON DAY02

内容

Part01	视频01	Selenium IDE高级1
Part02	视频02	Selenium IDE高级2



Selenium IDE高级(1)

Selenium IDE高
级(1)

Value

Value分类

Value为空

常量

变量

变量案例

其他存储器类型命令简介

Value



Value分类

- Value的三种类型：
 - 1、为空：value中值为空
 - 2、常量：输入的值是固定的
 - 3、变量：输入的值是可变的



Value为空

- 命令中不需要值的情况，常见的有：
 - Open打开网页

Table Source		
Command	Target	Value
open	file:///E:/demo/day01/example01.html	
store	张三	name
store	zhangsan@126.com	mail
store	你好, selenium !	liuyan
type	name=name	\${name}
type	name=e-mail	\${mail}
type	name=comments	\${liuyan}

Command: open

Target: file:///E:/demo/day01/example01.html

Value:

Select Find

- click点击

type	name=comments	\${liuyan}
clickAndWait	css=input[type="submit"]	
echo	我是: \${name}, 邮箱是: \${mail}, 留言内容...	

Command: clickAndWait

Target: css=input[type="submit"]

Value:

Select Find

- 常量：在程序中始终不会变化的值

Table Source		
Command	Target	Value
open	file:///E:/demo/day01/example01.html	
type	name=name	张三
type	name=e-mail	zhangsan@tedu.cn
type	name=comments	hello,Selenium
clickAndWait	css=input[type="submit"]	

Command	type	
Target	name=name	Select Find
Value	张三	



- 变量：在程序中可以变化的值
- 如何定义变量
 - Command : store
 - Target : 变量值
 - value : 变量名
- 如何使用变量
 - 在需要使用变量的那一行value中，把录制的常量删除掉，使用变量替代，变量使用格式为：
 \${变量名}



变量案例

- 案例1：把example01.html中的名字、E-Mail、留言全部使用变量
- 步骤1：名字处理，插入一行命令
 - Command：store
 - Target：张三
 - Value：name
 - 相当于：name=张三

Table

Source

Command	Target	Value
open	file:///E:/demo/day01/example01.html	
store	张三	name
type	name=name	张三
type	name=e-mail	zhangsan@tedu.cn
type	name=comments	hello,Selenium
clickAndWait	css=input[type="submit"]	

Command

store

Target

张三

Select

Find

Value

name



变量案例（续1）

- 步骤2：E-Mail处理，插入一行命令
 - Command：store
 - Target：zhangsan@126.com
 - Value：mail
 - 相当于：mail=zhangsan@126.com

知识讲解

Table Source		
Command	Target	Value
open	file:///E:/demo/day01/example01.html	
store	张三	name
store	zhangsan@126.com	mail
type	name=name	张三
type	name=e-mail	zhangsan@tedu.cn
type	name=comments	hello,Selenium
clickAndWait	css=input[type="submit"]	

Command	store	
Target	<input type="text" value="zhangsan@126.com"/>	<input type="button" value="Select"/> <input type="button" value="Find"/>
Value	<input type="text" value="mail"/>	



变量案例（续2）

- 步骤3：留言处理，插入一行命令
 - Command：store
 - Target：你好，Selenium！
 - Value：liuyan
 - 相当于：liuyan=你好，selenium！

Table

Source

Command	Target	Value
open	file:///E:/demo/day01/example01.html	
store	张三	name
store	zhangsan@126.com	e-mail
store	你好, selenium !	liuyan
type	name=name	张三
type	name=e-mail	zhangsan@tedu.cn
type	name=comments	hello,Selenium

Command

store

Target

你好, selenium !

Select

Find

Value

liuyan



变量案例（续3）

- 步骤4：把名字、E-Mail、留言中的常量使用变量替代
 - 名字：\${name}

Table

Source

Command	Target	Value
open	file:///E:/demo/day01/example01.html	
store	张三	name
store	zhangsan@126.com	e-mail
store	你好，selenium！	comments
type	name=name	\${name}
type	name=e-mail	zhangsan@tedu.cn
type	name=comments	hello,Selenium

Command

type

Target

name=name

Select

Find

Value

\${name}



变量案例（续4）

– E-Mail : \${e-mail}

Table	Source	
Command	Target	Value
open	file:///E:/demo/day01/example01.html	
store	张三	name
store	zhangsan@126.com	e-mail
store	你好，selenium！	comments
type	name=name	\${name}
type	name=e-mail	\${e-mail}
type	name=comments	hello,Selenium

Command

type

Target

Select

Find

Value



变量案例（续5）

– 留言：\${liuyan}

Table Source		
Command	Target	Value
store	你好，selenium！	comments
type	name=name	\${name}
type	name=e-mail	\${e-mail}
type	name=comments	\${liuyan}
clickAndWait	css=input[type="submit"]	

Command	type	
Target	name=comments	Select Find
Value	\${liuyan}	



变量案例（续6）

– 使用echo输出信息：

Table Source

Command	Target	Value
store	你好，selenium！	liuyan
type	name=name	\${name}
type	name=e-mail	\${mail}
type	name=comments	\${liuyan}
clickAndWait	css=input[type="submit"]	
echo	我是：\${name},邮箱是：\${mail},留言内容...	

Command

echo

Target

我是：\${name},邮箱是：\${mail},留言内容：\${liuyan}

Select

Find

Value

Log Reference UI-Element Rollup Info+ Clear

[info] Executing: |type | name=name | \${name} |

[info] Executing: |type | name=e-mail | \${mail} |

[info] Executing: |type | name=comments | \${liuyan} |

[info] Executing: |clickAndWait | css=input[type="submit"] | |

[info] Executing: |echo | 我是：\${name},邮箱是：\${mail},留言内容：\${liuyan} | |

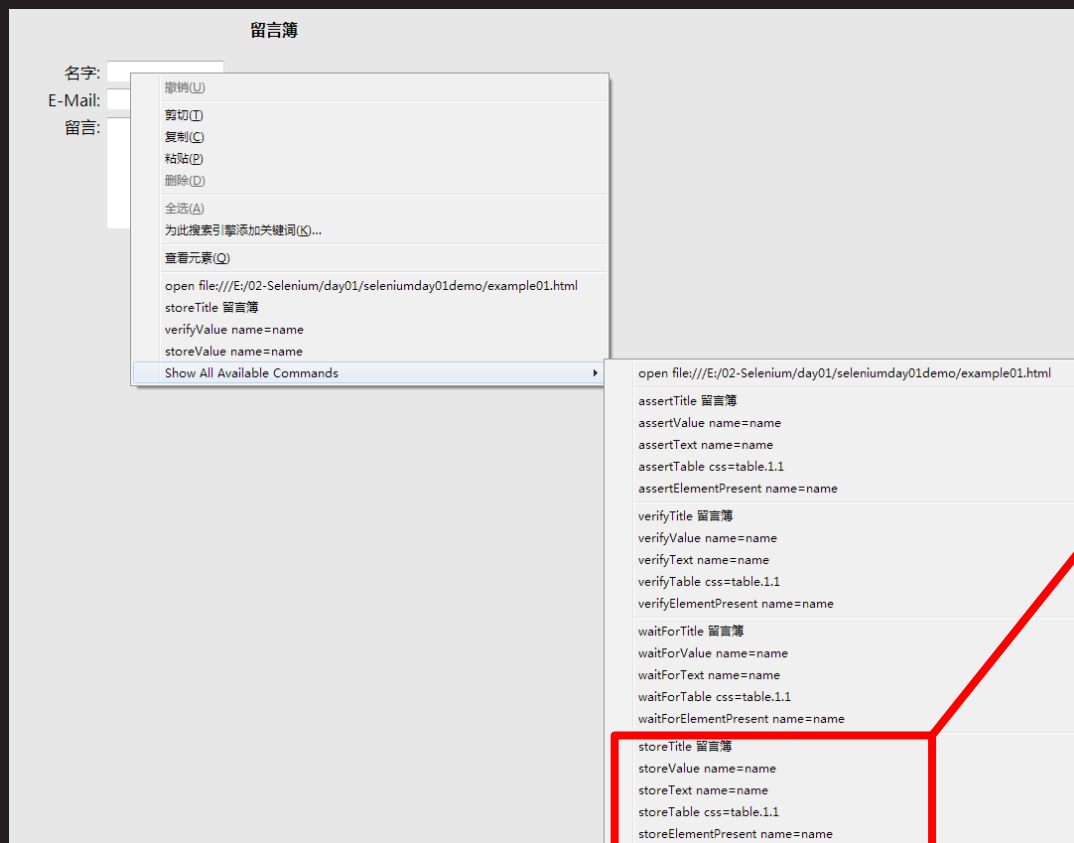
[info] echo: 我是：张三,邮箱是：zhangsan@126.com,留言内容：你好，selenium！

[info] Test case passed



其他存储器类型命令简介

- 以store开头的命令为存储器类型命令，可以在运行过程中自动获取一些页面元素信息，可以使用变量来存储这些有用的信息

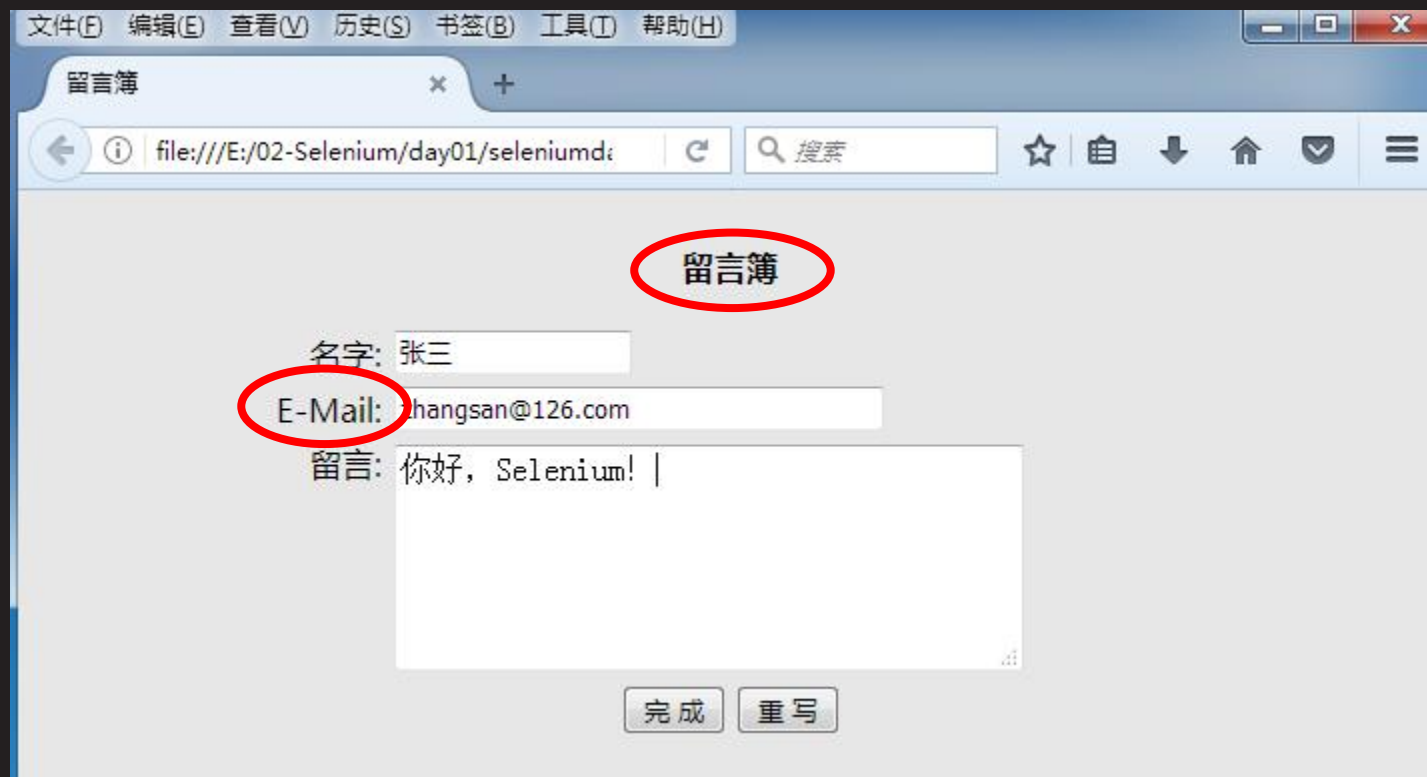


```
storeTitle 留言簿
storeValue name=name
storeText name=name
storeTable css=table.1.1
storeElementPresent name=name
```



其他存储器类型命令简介（续1）

- 案例2：
 - 获取example01.html页面的标题（留言簿），并输出
 - 获取页面的题目“留言簿”和邮箱的提示文本“E-Mail：”，并输出



其他存储器类型命令简介（续2）

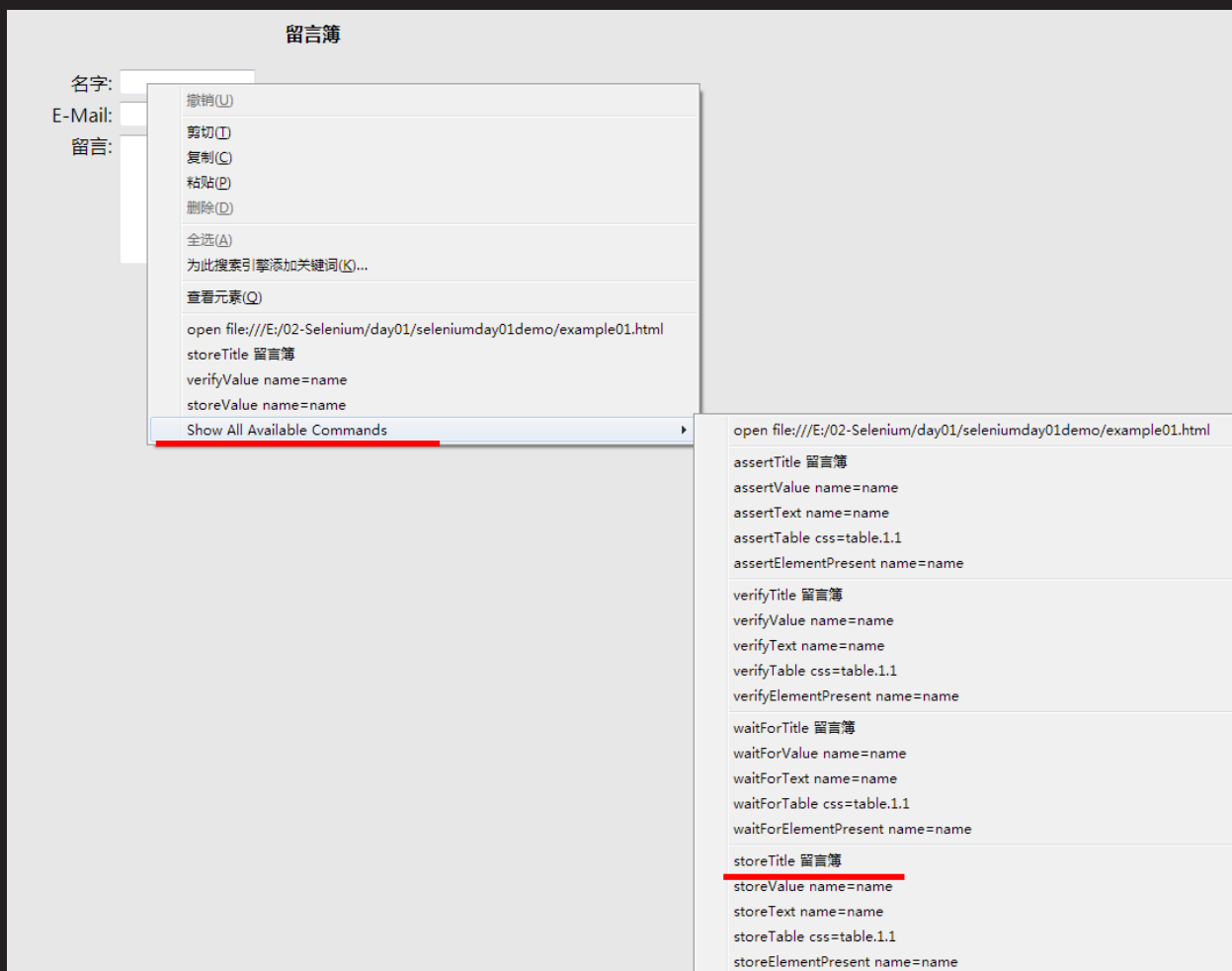
- 完成任务1：获取example01.html页面的标题（留言簿），并输出
 - 步骤1：选中Table中的点击“完成”按钮（即准备在此步骤前插入命令）

Table Source		
Command	Target	Value
store	zhangsan@126.com	mail
store	你好,Selenium!	liuyan
type	name=name	\${name}
type	name=e-mail	\${mail}
type	name=comments	\${liuyan}
selectWindow	null	
clickAndWait	css=input[type="submit"]	
echo	我是：\${name},邮箱是：\${mail},留言内容：\$...	



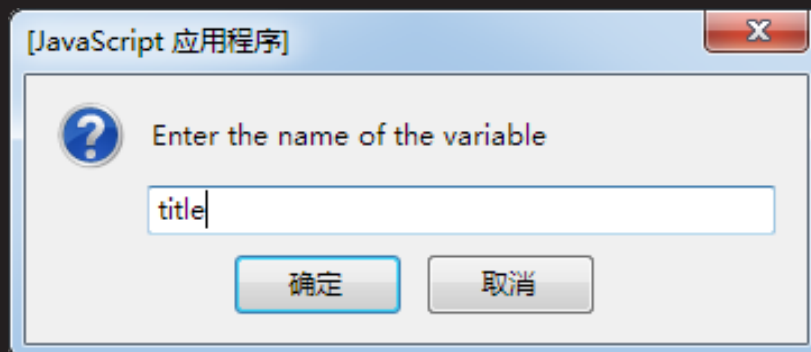
其他存储器类型命令简介（续3）

- 步骤2：鼠标放在example01.html页面上，点击右键，菜单：Show All Available Commands->storeTitle 留言簿

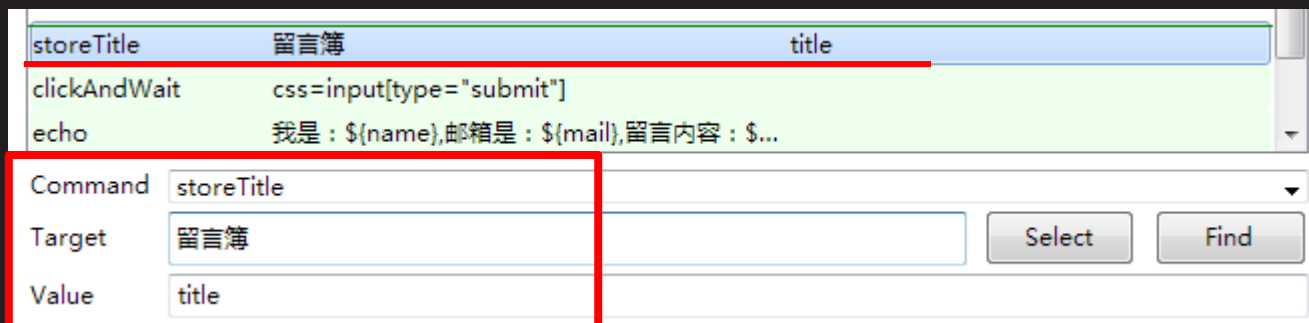


其他存储器类型命令简介（续4）

- 步骤3：输入变量名称，即使用什么变量来存储获取的网页标题



- 步骤4：在table中添加了命令



其他存储器类型命令简介（续5）

- 步骤5：按照帮助文档修改命令

Log Reference UI-Element Rollup

storeTitle(variableName)
Generated from **getTitle()**

Returns:
the title of the current page
Gets the title of the current page.

- 修改完成

storeTitle	title
clickAndWait	css=input[type="submit"]
echo	我是：\${name},邮箱是：\${mail},留言内容：\$...

Command

storeTitle

Target

title

Value

Select

Find



其他存储器类型命令简介（续6）

— 步骤6：使用echo命令输出标题

storeTitle	title
echo	网页标题：\${title}
clickAndWait	css=input[type="submit"]

Command	echo	
Target	<input type="text" value="网页标题：\${title}"/>	<input type="button" value="Select"/> <input type="button" value="Find"/>
Value	<input type="text"/>	



其他存储器类型命令简介（续7）

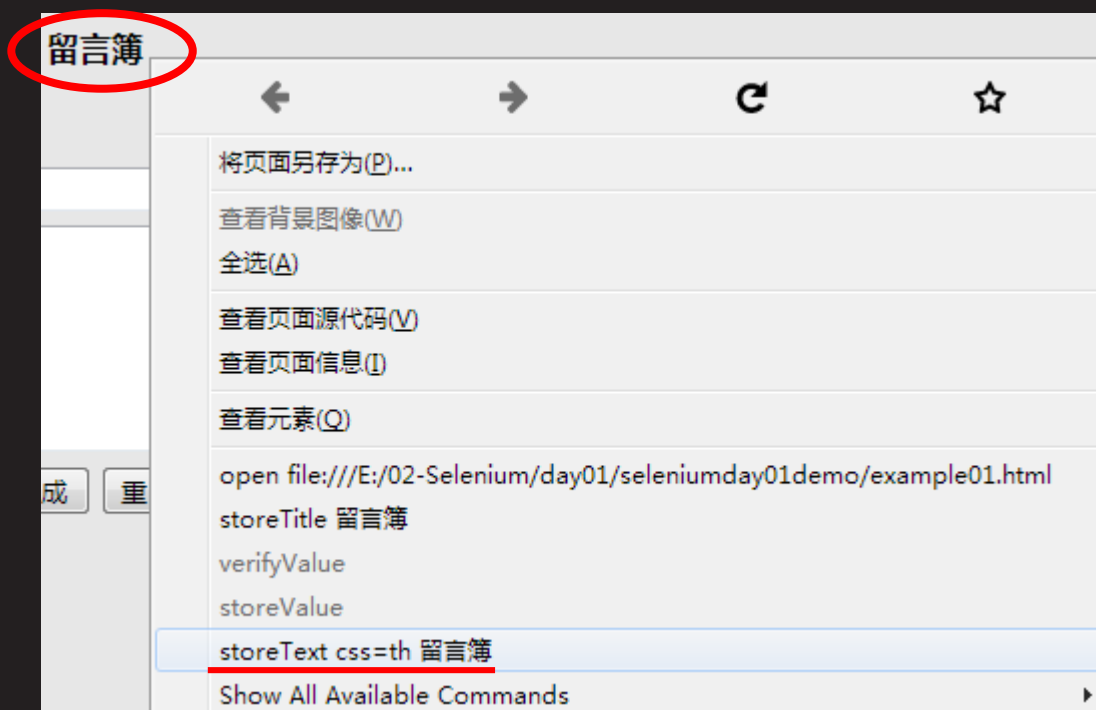
- 完成任务2：获取页面的题目“留言簿”和邮箱的提示文本“E-Mail：”，并输出
 - 步骤1：选中Table中的点击“完成”按钮（即准备在此步骤前插入命令）

Table	Source	
Command	Target	Value
type	name=name	\${name}
type	name=e-mail	\${mail}
type	name=comments	\${liuyan}
storeTitle	title	
echo	网页标题：\${title}	
clickAndWait	css=input[type="submit"]	
echo	我是：\${name},邮箱是：\${mail},留言内容：\$...	



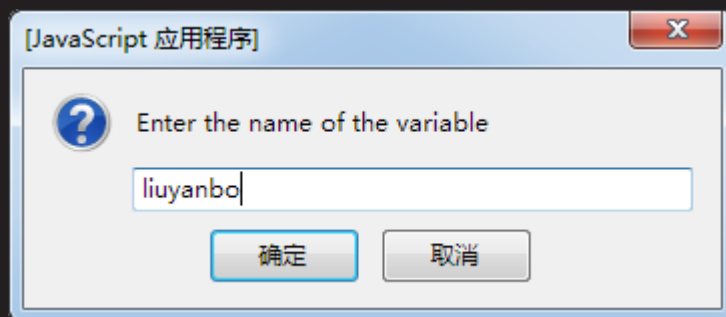
其他存储器类型命令简介（续8）

- 步骤2：鼠标放在“留言簿”文本上，点击右键，菜单：
storeText css=th 留言簿

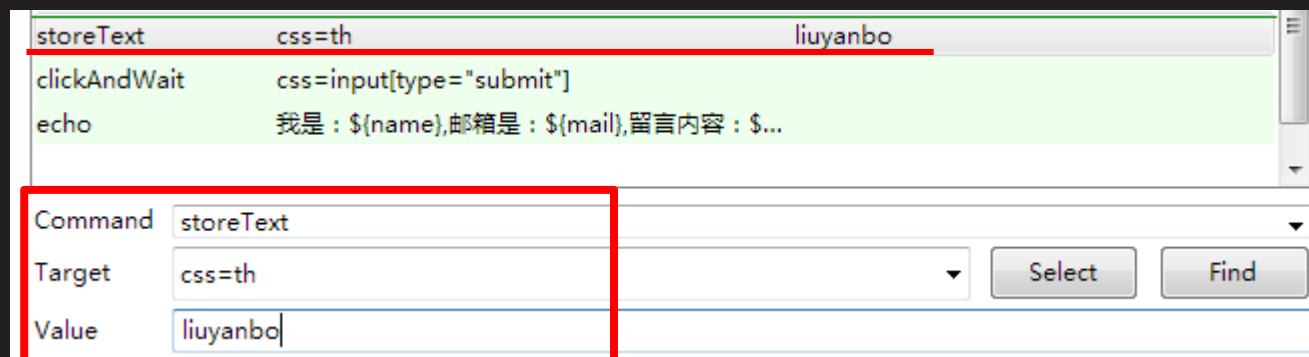


其他存储器类型命令简介（续9）

- 步骤3：输入变量名称，即使用什么变量来存储获取的文本



- 步骤4：在table中添加了命令



其他存储器类型命令简介（续10）

— 步骤5：参考帮助文档命令格式

The screenshot shows the Selenium IDE Reference window with the 'storeText' command selected. The window has tabs for Log, Reference, UI-Element, and Rollup. The Reference tab is active, displaying the following information:

storeText(locator, variableName)
Generated from **getText(locator)**

Arguments:

- locator - an element locator

Returns:

the text of the element

Gets the text of an element. This works for any element that contains text in all browsers or the innerText (IE-like browsers) of the element, which is the text of the element.

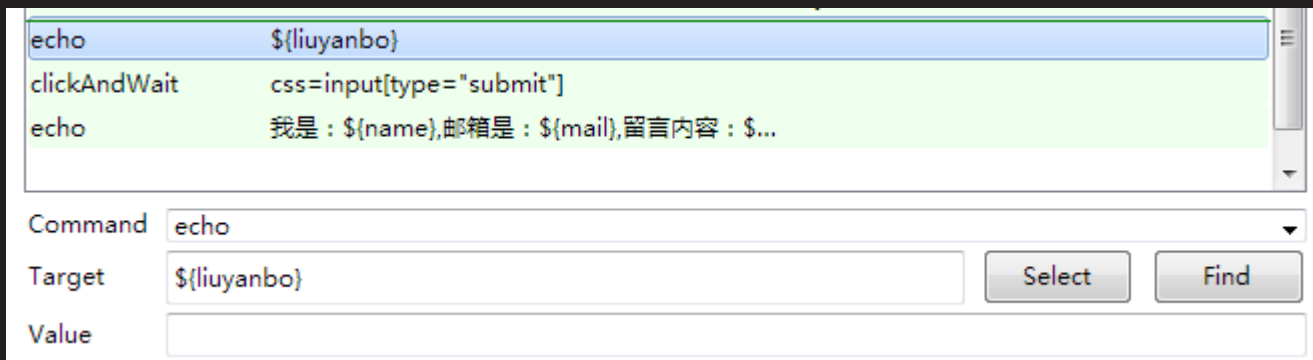
— 检查命令是否正确

The screenshot shows the Selenium IDE interface. The command list at the top contains three commands: 'storeText' with target 'css=th' and value 'liuyanbo', 'clickAndWait' with target 'css=input[type="submit"]', and 'echo' with target '我是：\${name},邮箱是：\${mail},留言内容：\$...'. Below the command list is a configuration panel with three fields: 'Command' set to 'storeText', 'Target' set to 'css=th', and 'Value' set to 'liuyanbo'. The 'Command' and 'Target' fields are highlighted with a red box.



其他存储器类型命令简介（续11）

— 步骤6：使用echo命令输出标题



- 练习：完成“E-Mail：”的获取和输出



Selenium IDE高级(2)

Selenium IDE高级(2)

结果判断

结果判断概述

断言概述

验证概述

验证案例

等待概述

等待案例

导出Python代码

概述

导出方式1

导出方式2

结果判断



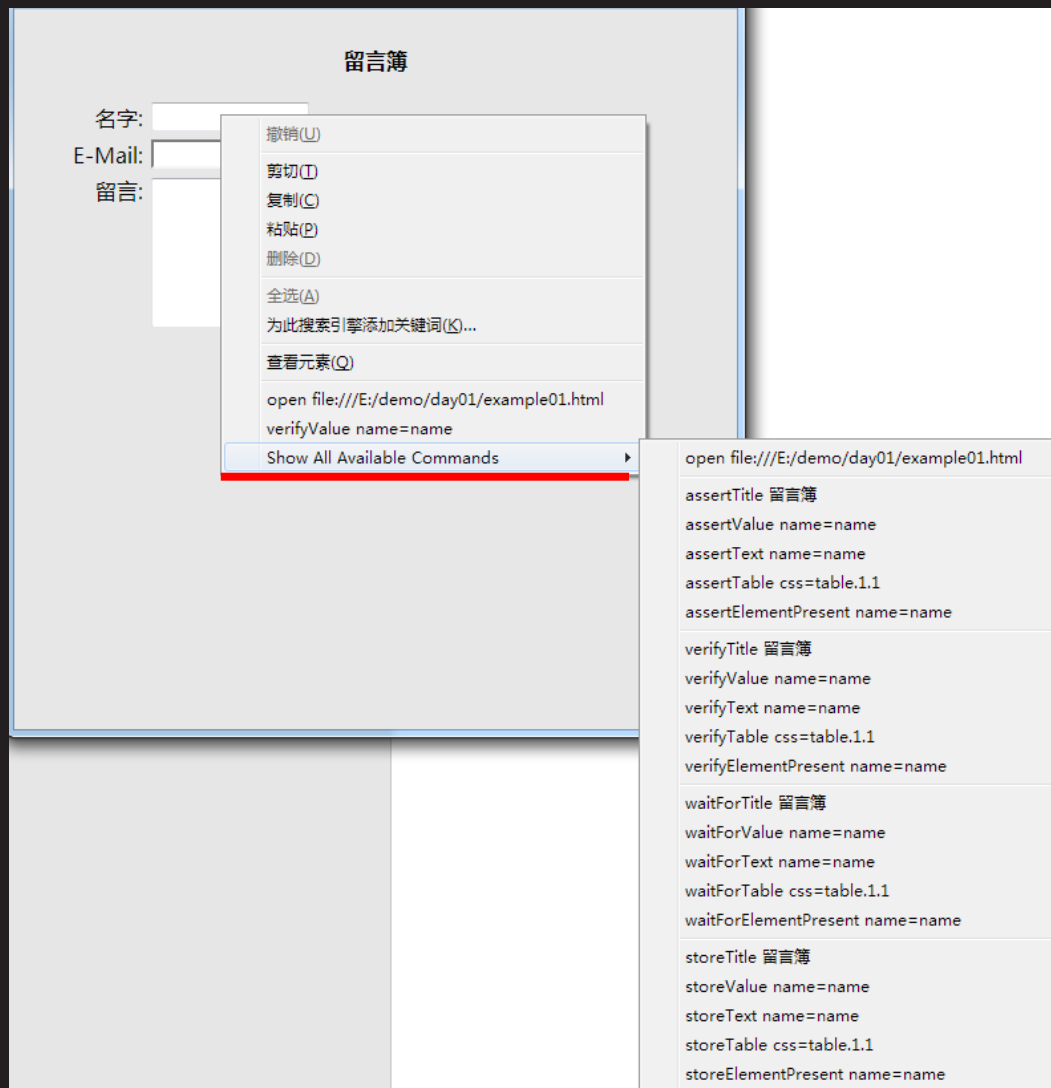
结果判断概述

- 测试用例不仅包含测试步骤和使用的数据，还要包括预期结果，执行的时候，需要用实际结果与预期结果进行对比，才能判断用例执行是成功还是失败，否则用例执行是对是错无法判断，测试的执行将失去意义
- 在Selenium IDE中提供断言与验证来表示预期结果，并在脚本执行过程中，自动把程序实际结果与设定的预期结果进行比较，并给出反馈



结果判断概述（续1）

- 添加预期结果的方法：



结果判断概述（续2）

- 断言类型命令可以细分为：

- assert（断言）

当断言命令执行失败，脚本会立刻停止执行，后续脚本内容不会执行

- verify（验证）

当验证命令执行失败，会在执行日志中输出失败信息，然后继续执行后续脚本内容

- waitFor（等待）

在继续下一条命令之前，waitFor命令会等待某个条件真实发生

等待期间，条件定义的情况发生了，脚本会继续执行

等待期间，条件定义的情况没有发生，会在执行日志中输出失败信息，然后继续执行后续脚本内容

默认情况下，等待的时间为30秒



结果判断概述（续3）

- 每一种断言类型，常见的5种检查方式：
 - Title：页面标题
 - value：元素的值
 - Text：元素的文本信息
 - Table：元素的表格
 - ElementPresent：页面元素是否出现

```
open file:///E:/demo/day01/example01.html

assertTitle 留言板
assertValue name=comments
assertText name=comments
assertTable css=table.3.1
assertElementPresent name=comments

verifyTitle 留言板
verifyValue name=comments
verifyText name=comments
verifyTable css=table.3.1
verifyElementPresent name=comments

waitForTitle 留言板
waitForValue name=comments
waitForText name=comments
waitForTable css=table.3.1
waitForElementPresent name=comments

storeTitle 留言板
storeValue name=comments
storeText name=comments
storeTable css=table.3.1
storeElementPresent name=comments
```



- 断言（assert）：
 - 如果使用断言，测试用例将会在断言失败后停止运行
 - 优点：明确测试是否通过
 - 缺点：若检查失败，后续的检查不会被执行，无法收集检查的结果状态



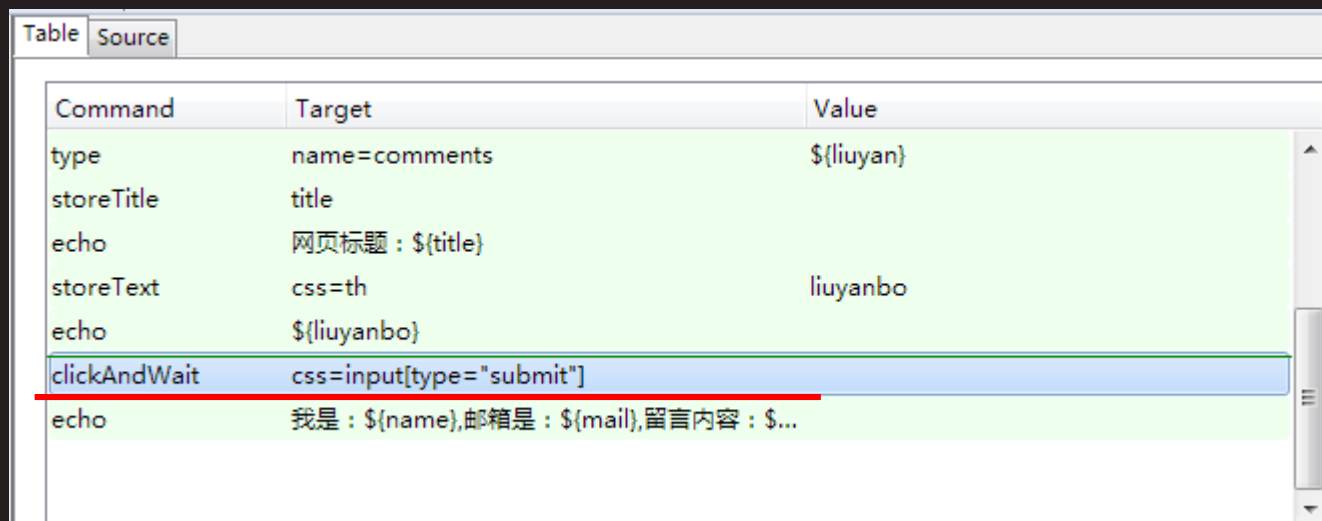
断言案例

- 案例3：使用断言检查example01.html程序：
 - 该网页标题是否为留言簿
 - 名字中输入的是否为张三
 - 第一个按钮是否为“完成”



断言案例（续1）

- 完成任务1：该网页标题是否为留言簿
 - 步骤1：选中Table中的点击“完成”按钮（即准备在此步骤前插入命令）

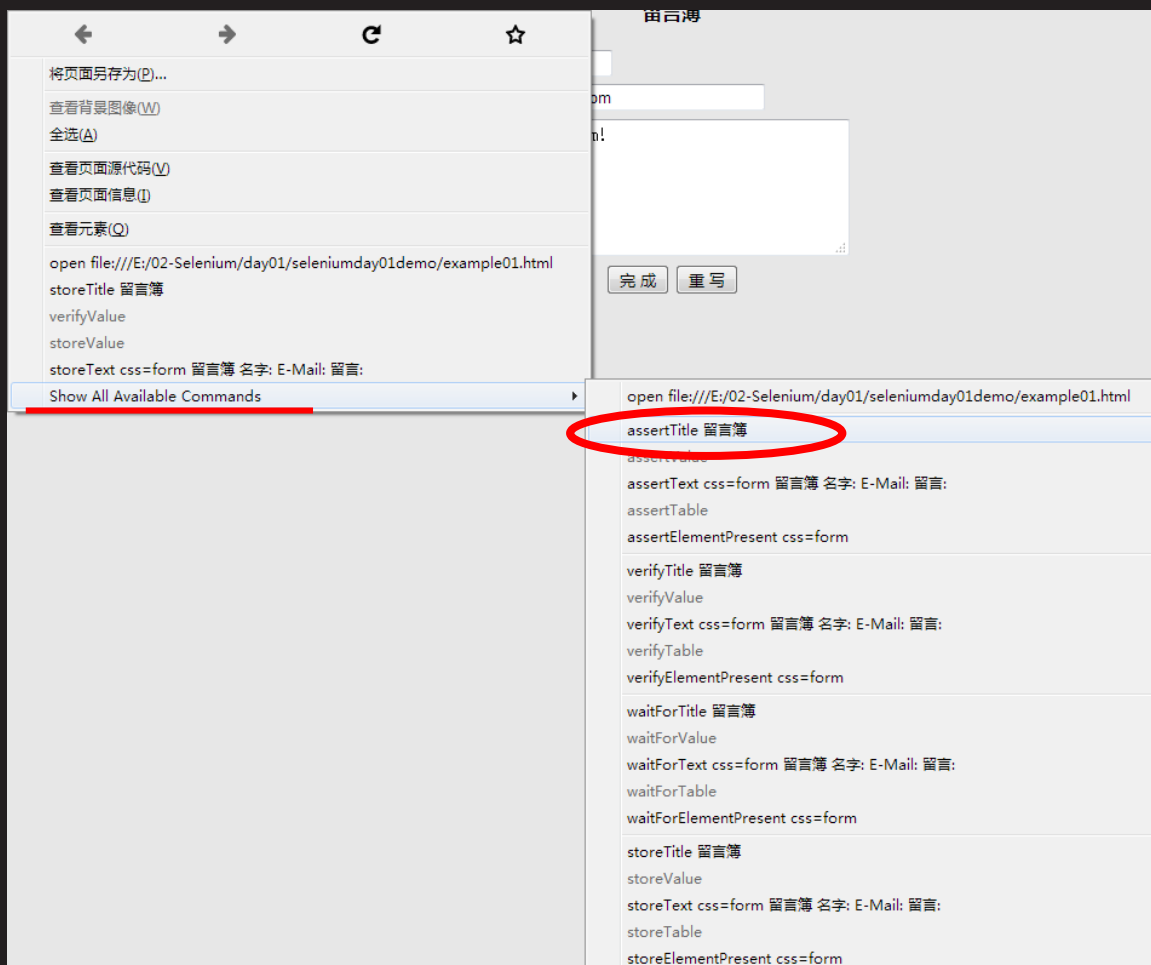


Command	Target	Value
type	name=comments	\${liuyan}
storeTitle	title	
echo	网页标题：\${title}	
storeText	css=th	liuyanbo
echo	\${liuyanbo}	
clickAndWait	css=input[type="submit"]	
echo	我是：\${(name)},邮箱是：\${(mail)},留言内容：\$...	



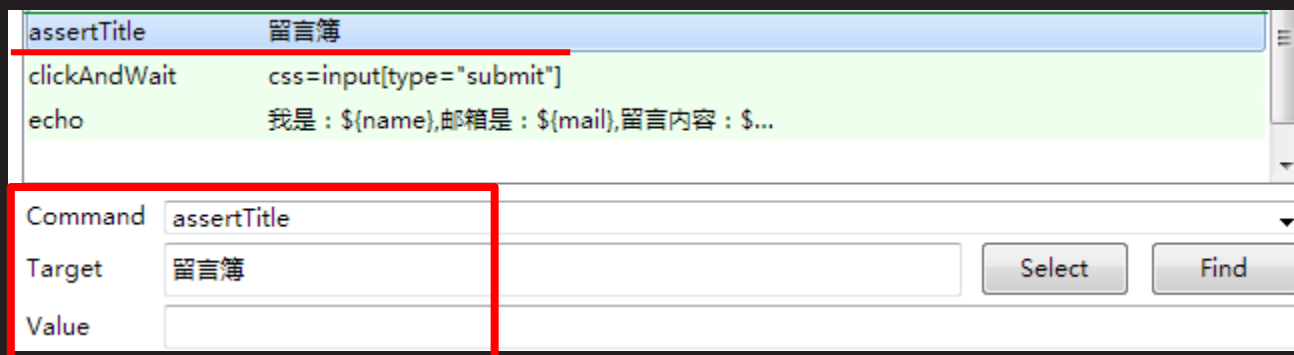
断言案例（续2）

- 步骤2：鼠标放在example01.html页面上，点击右键，菜单：Show All Available Commands->assertTitle 留言簿



断言案例（续3）

- 步骤3：在table中添加了命令



断言案例（续4）

— 步骤4：执行脚本并查看Log信息

Log	Reference	UI-Element	Rollup	Info	Clear
[info]	Executing:	storeTitle title			
[info]	Executing:	echo 网页标题 : \${title}			
[info]	echo:	网页标题 : 留言簿			
[info]	Executing:	storeText css=th liuyanbo			
[info]	Executing:	echo \${liuyanbo}			
[info]	echo:	留言簿			
[info]	Executing:	assertTitle 留言簿			
[info]	Executing:	clickAndWait css=input[type="submit"]			
[info]	Executing:	echo 我是 : \${name},邮箱是 : \${mail},留言内容 : \${liuyan}			
[info]	echo:	我是 : 张三,邮箱是 : zhangsan@126.com,留言内容 : 你好,Selenium!			
[info]	Test case	passed			



断言案例（续5）

- 完成任务2：名字中输入的是否为张三
 - 步骤1：选中Table中的点击“完成”按钮（即准备在此步骤前插入命令）

Table	Source
-------	--------

Command	Target	Value
type	name=comments	\${liuyan}
storeTitle	title	
echo	网页标题：\${title}	
storeText	css=th	liuyanbo
echo	\${liuyanbo}	
assertTitle	留言簿	
clickAndWait	css=input[type="submit"]	
echo	我是：\${(name)},邮箱是：\${(mail)},留言内容：\$...	

Command

clickAndWait

Target

css=input[type="submit"]

Select

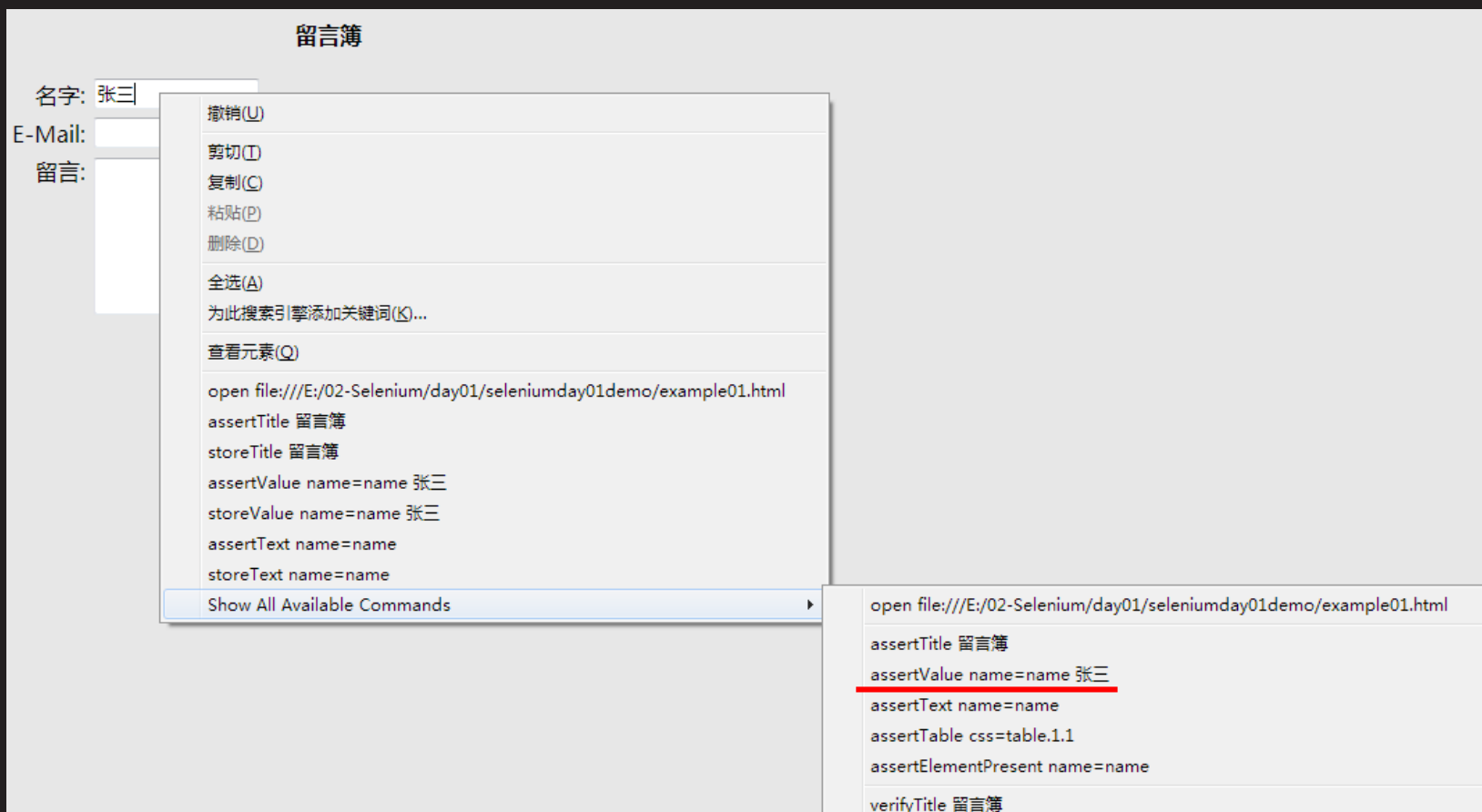
Find

Value



断言案例（续6）

- 步骤2：在名字文本框中输入预期结果“张三”，鼠标放在名字文本框上，点击右键，菜单：Show All Available Commands->assertValue name=name 张三



断言案例（续7）

– 步骤3：在table中添加了命令

Table		
Source		
Command	Target	Value
type	name=comments	\${liuyan}
storeTitle	title	
echo	网页标题：\${title}	
storeText	css=th	liuyanbo
echo	\${liuyanbo}	
assertTitle	留言簿	
assertValue	name=name	张三
clickAndWait	css=input[type="submit"]	
echo	我是：\${name},邮箱是：\${mail},留言内容：\$...	

Command	assertValue
Target	name=name
Value	张三



断言案例（续8）

— 步骤4：执行脚本并查看Log信息

Log	Reference	UI-Element	Rollup	Info	Clear
[info] Executing: echo 网页标题: \${title}					
[info] echo: 网页标题: 留言板					
[info] Executing: storeText css=th liuyanbo					
[info] Executing: echo \${liuyanbo}					
[info] echo: 留言板					
[info] Executing: assertTitle 留言板					
[info] Executing: assertValue name=name 张三					
[info] Executing: clickAndWait css=input[type="submit"]					
[info] Executing: echo 我是: \${name},邮箱是: \${mail},留言内容: \${liuyan}					
[info] echo: 我是: 张三,邮箱是: zhangsan@126.com,留言内容: 你好,Selenium!					
[info] Test case passed					

- 练习：完成任务3，第一个按钮是否为“完成”



- 验证（verify）：
 - 当执行验证命令失败后不会终止测试
 - 优点：使用验证，在没有意外异常发生的情况下，则测试会被执行完毕，而不管是否发现bug
 - 缺点：检查测试结果花费的时间比较多



验证案例

- 案例4：使用验证检查example05.html
 - 该网页标题是否为“达内测试学院请您留言”
 - 单选按钮“男”选中
 - 职业选择“行政管理”
 - 个人爱好选中“电脑网络”和“棋牌游戏”

文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)

达内测试学院请您留言

file:///E:/02-Selenium/day01/seleniumday01demo/example05.html

请完成以下表格

姓名 test

密码 ●●●●

请在此处填写姓名，
字符最长为四个汉字，或八个英文字母。

性别 ☒ 男 ☐ 女

电子邮件地址 test@tedu.cn

职业 行政管理

个人爱好 ☒ 电脑网络 ☐ 影视娱乐 ☒ 棋牌游戏
☐ 读书读报 ☐ 美酒佳肴 ☐ 绘画书法

在此选择兴趣爱好，可以选择一个以上的选项。

留言内容 你好达内!

填写完成后，选择下面的提交按钮提交表单。

提交 重置

验证案例（续1）

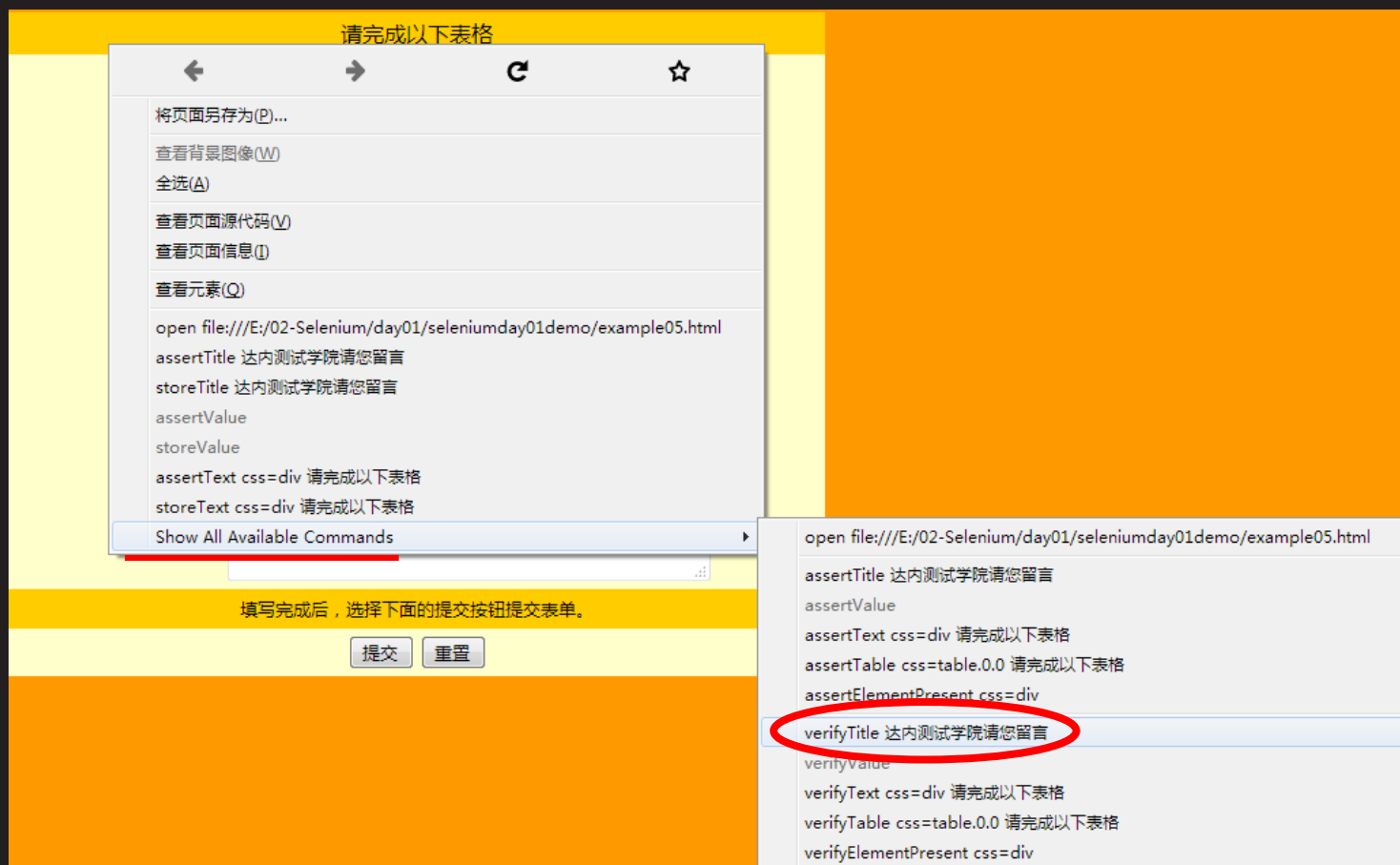
- 完成任务1：该网页标题是否为留言簿
 - 步骤1：选中Table中的点击“提交”按钮（即准备在此步骤前插入命令）

Table Source		
Command	Target	Value
click	name=sex	
type	name=email	test@tedu.cn
select	name=profession	label=行政管理
click	name=computer	
click	name=chess	
type	name=textfield3	你好达内！
clickAndWait	name=Submit	



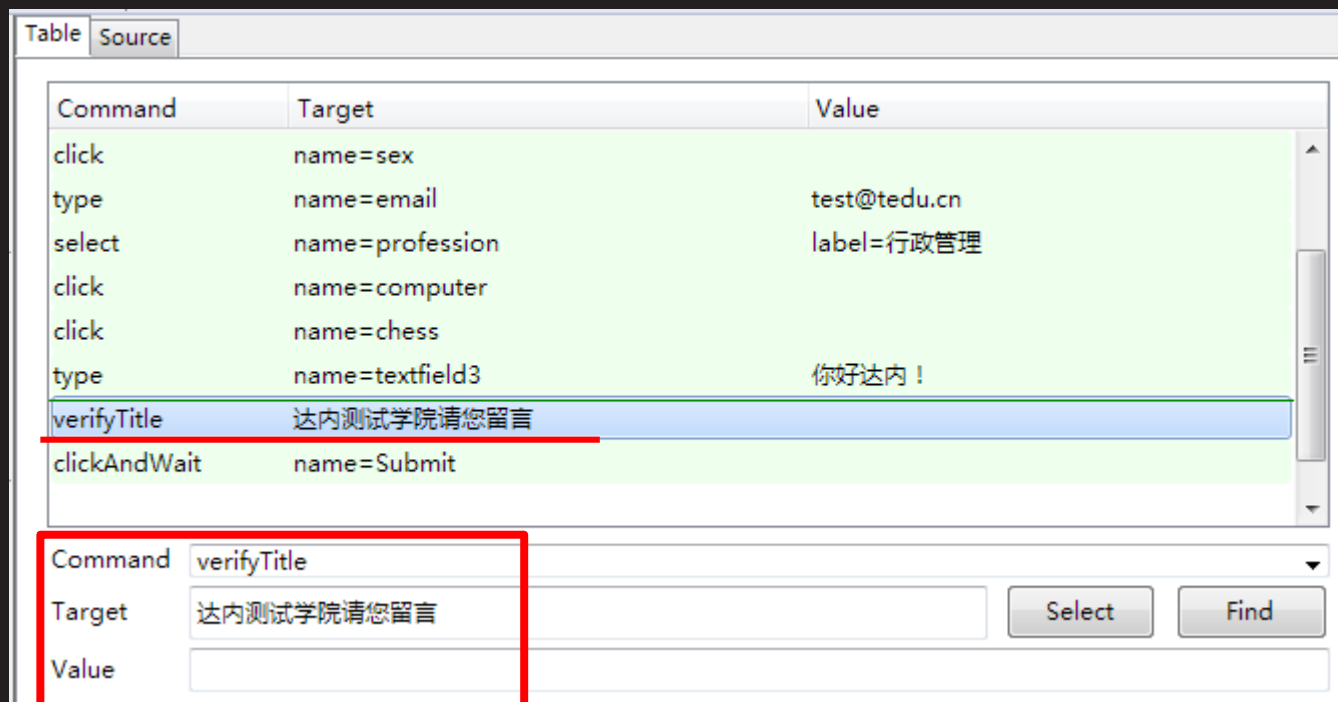
验证案例（续2）

- 步骤2：鼠标放在example05.html页面上，点击右键，菜单：Show All Available Commands->verifyTitle 达内测试学院请您留言



验证案例（续3）

– 步骤3：在table中添加了命令



验证案例（续4）

— 步骤4：执行脚本并查看Log信息

Log	Reference	UI-Element	Rollup
[info]	Executing:	type name=username test	
[info]	Executing:	type name=passwords 1234	
[info]	Executing:	click name=sex	
[info]	Executing:	type name=email test@tedu.cn	
[info]	Executing:	select name=profession label=行政管理	
[info]	Executing:	click name=computer	
[info]	Executing:	click name=chess	
[info]	Executing:	type name=textfield3 你好达内！	
[info]	Executing:	verifyTitle 达内测试学院请您留言	
[info]	Executing:	clickAndWait name=Submit	
[info]	Test case passed		



验证案例（续5）

- 完成任务2：单选按钮“男”选中
 - 步骤1：选中Table中的点击“提交”按钮（即准备在此步骤前插入命令）

Table Source		
Command	Target	Value
click	name=sex	
type	name=email	test@tedu.cn
select	name=profession	label=行政管理
click	name=computer	
click	name=chess	
type	name=textfield3	你好达内！
verifyTitle	达内测试学院请您留言	
clickAndWait	name=Submit	

Command

clickAndWait

Target

name=Submit

Select

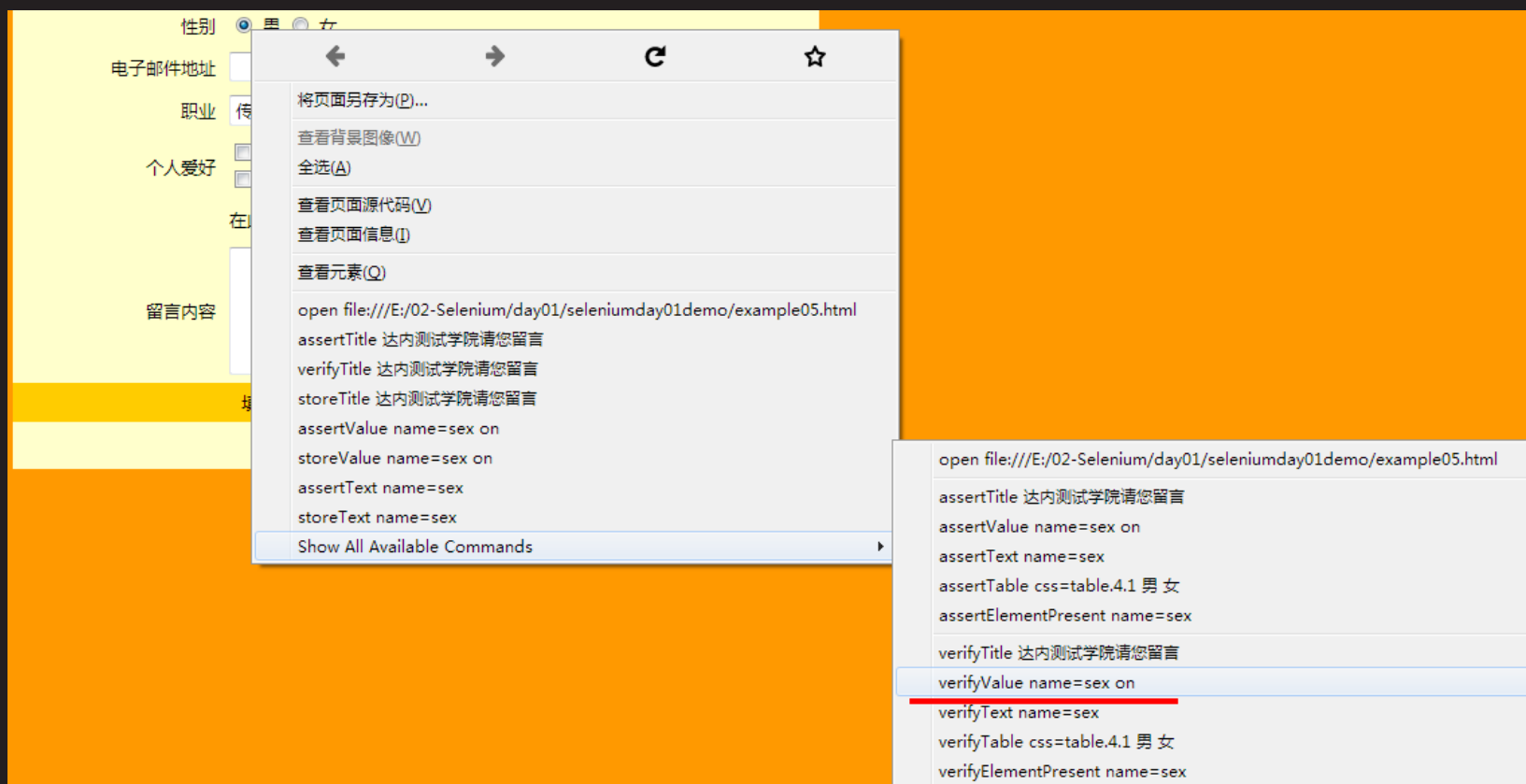
Find

Value



验证案例（续6）

- 步骤2：选中单选按钮“男”，鼠标放在其上，点击右键，菜单：Show All Available Commands->verifyValue name=sex on



验证案例（续7）

- 步骤3：在table中添加了命令

verifyTitle	达内测试学院请您留言	
verifyValue	name=sex	on
clickAndWait	name=Submit	

Command	verifyValue	
Target	name=sex	Select
Value	on	

Find



验证案例（续8）

— 步骤4：执行脚本并查看Log信息

Log	Reference	UI-Element	Rollup
[info]	Executing:	type name=passwords 1234	
[info]	Executing:	click name=sex	
[info]	Executing:	type name=email test@tedu.cn	
[info]	Executing:	select name=profession label=行政管理	
[info]	Executing:	click name=computer	
[info]	Executing:	click name=chess	
[info]	Executing:	type name=textfield3 你好达内！	
[info]	Executing:	verifyTitle 达内测试学院请您留言	
[info]	Executing:	verifyValue name=sex on	
[info]	Executing:	clickAndWait name=Submit	
[info]	Test case passed		



验证案例（续9）

- 练习：完成任务3：
 - 职业选择 “行政管理”

Table	Source	CH	S	?
Command	Target	Value		
select	name=profession	label=行政管理		
click	name=computer			
click	name=chess			
type	name=textfield3	你好达内！		
verifyTitle	达内测试学院请您留言			
verifyValue	name=sex	on		
verifyValue	name=profession	行政管理		
clickAndWait	name=Submit			

Command	verifyValue		
Target	name=profession	Select	Find
Value	行政管理		



验证案例（续10）

- 练习：完成任务4：
 - 个人爱好选中“电脑网络”和“棋牌游戏”

Table Source		
Command	Target	Value
click	name=computer	
click	name=chess	
type	name=textfield3	你好达内！
verifyTitle	达内测试学院请您留言	
verifyValue	name=sex	on
verifyValue	name=profession	行政管理
verifyValue	name=computer	on
verifyValue	name=chess	on
clickAndWait	name=Submit	

Command	verifyValue
Target	name=computer
Value	on

SelectFind

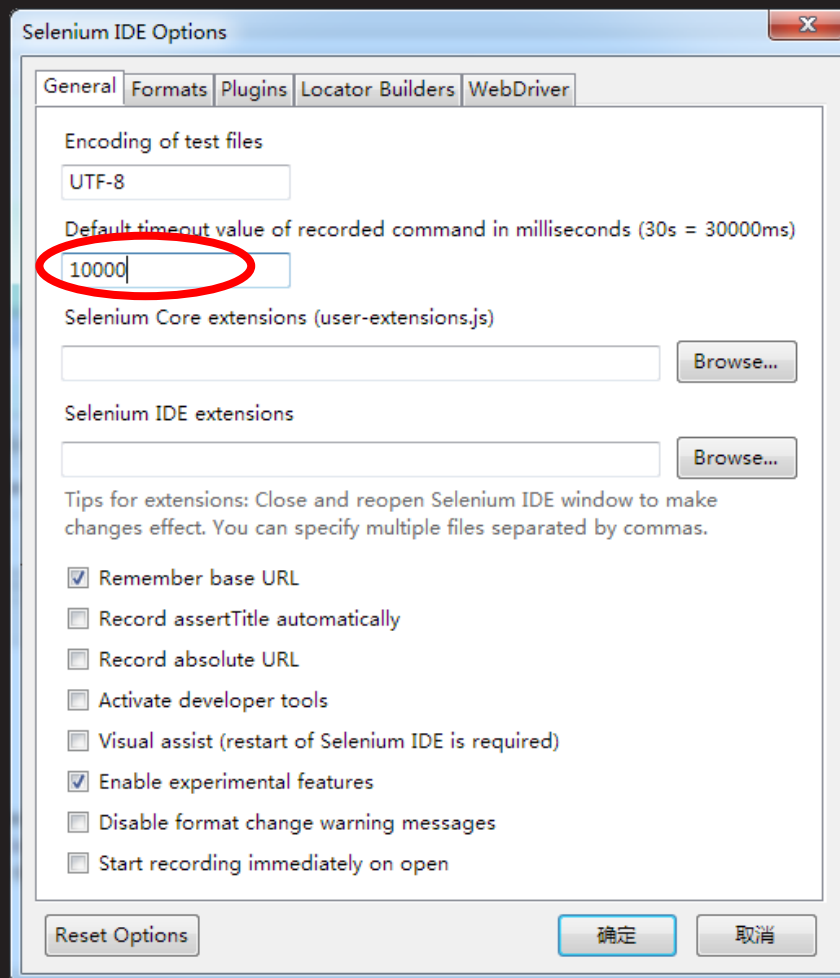


- waitFor (等待)
 - 在继续下一条命令之前，waitFor命令会等待某个条件真实发生
 - 等待期间，条件定义的情况发生了，脚本会继续执行
 - 等待期间，条件定义的情况没有发生，会在执行日志中输出失败信息，然后继续执行后续脚本内容
 - 默认情况下，等待的时间为30秒



等待概述（续1）

- 设置等待时间
 - 菜单：Options->Options



等待案例

- 案例5：使用等待检查example05.html
 - 密码是否为“1234”
 - 单选按钮“女”选中
 - 职业选择“艺术/设计”
 - 个人爱好选中“影视娱乐”和“绘画书法”

请完成以下表格

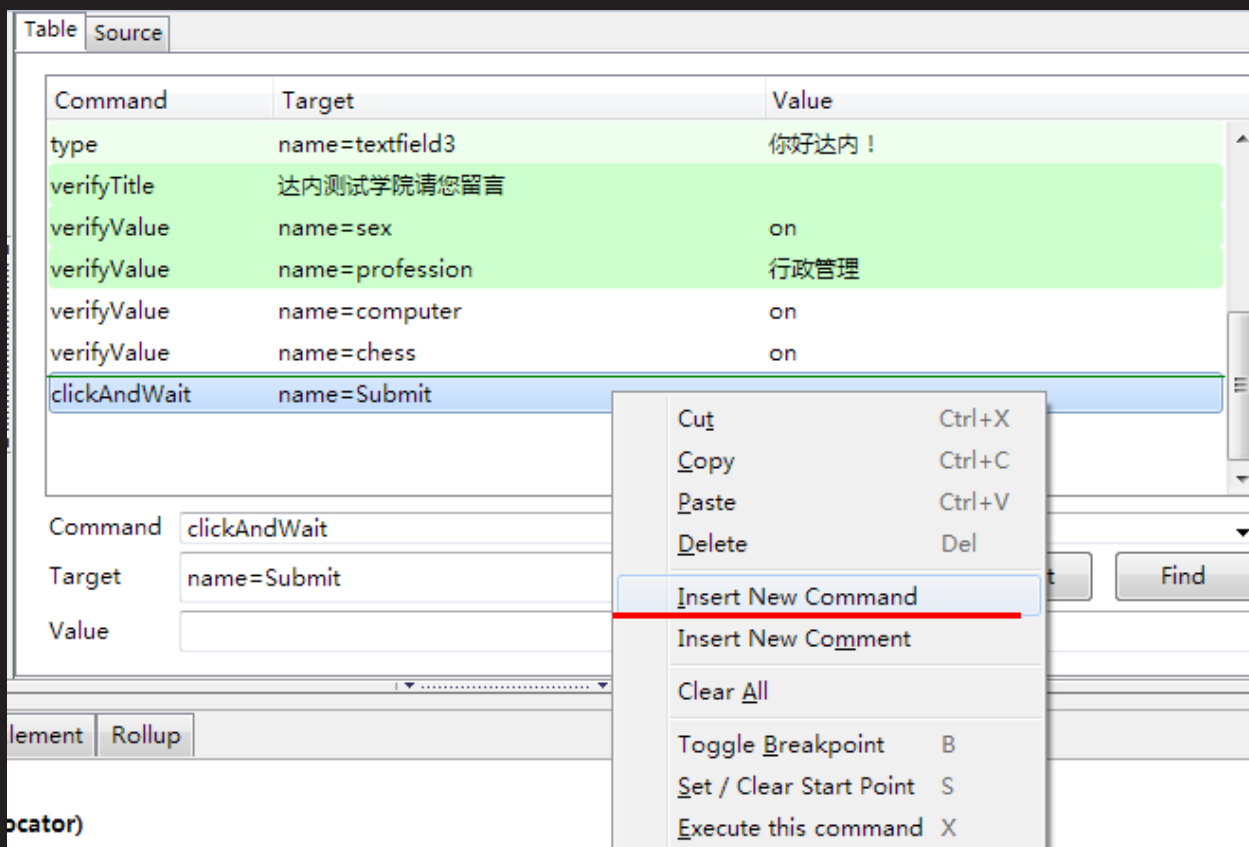
姓名	<input type="text" value="abcd"/>
密码	<input type="password" value="...."/>
请在此处填写姓名， 字符最长为四个汉字，或八个英文字母。	
性别	<input type="radio"/> 男 <input checked="" type="radio"/> 女
电子邮件地址	<input type="text" value="abcd@126.com"/>
职业	<input type="text" value="艺术/设计"/>
个人爱好	<input type="checkbox"/> 电脑网络 <input checked="" type="checkbox"/> 影视娱乐 <input type="checkbox"/> 棋牌游戏 <input type="checkbox"/> 读书读报 <input type="checkbox"/> 美酒佳肴 <input checked="" type="checkbox"/> 绘画书法
在此选择兴趣爱好，可以选择一个以上的选项。	
留言内容	<input type="text" value="达内教育"/>

填写完成后，选择下面的提交按钮提交表单。



等待案例（续1）

- 完成任务1：密码是否为“1234”
 - 步骤1：选中Table中的点击“提交”按钮，点击鼠标右键，选择“Insert New Command”



等待案例（续2）

- 步骤2：输入如下命令：
Command : waitForValue
Target : name=passwords
Value : 1234

type	name=textfield3	达内教育
waitForValue	name=passwords	1234
clickAndWait	name=Submit	

Command	waitForValue
Target	name=passwords
Value	1234



等待案例（续3）

– 步骤3：执行脚本并查看Log信息

Log	Reference	UI-Element	Rollup
[info]	Executing:	select name=profession label=艺术/设计	
[info]	Executing:	click name=computer	
[info]	Executing:	click name=computer	
[info]	Executing:	click name=film	
[info]	Executing:	click name=painting	
[info]	Executing:	type name=textfield3 达内教育	
[info]	Executing:	<u> waitForValue name=passwords 1234 </u>	
[info]	Executing:	clickAndWait name=Submit	
[info]	Test case passed		



等待案例（续4）

- 练习：完成任务2：单选按钮“女”选中

waitForValue	name=passwords	1234
waitForValue	xpath= (//input[@name='sex'])[2]	on
clickAndWait	name=Submit	

Command	waitForValue	
Target	xpath= (//input[@name='sex'])[2]	Select Find
Value	on	

Log	Reference	UI-Element	Rollup
[info]	Executing:	click name=computer	
[info]	Executing:	click name=computer	
[info]	Executing:	click name=film	
[info]	Executing:	click name=painting	
[info]	Executing:	type name=textfield3 达内教育	
[info]	Executing:	waitForValue name=passwords 1234	
[info]	Executing:	waitForValue xpath= (//input[@name='sex'])[2] on	
[info]	Executing:	clickAndWait name=Submit	
[info]	Test case passed		



等待案例（续5）

- 练习：完成任务3：职业选择“艺术/设计”

Table Source

Command	Target	Value
click	name=painting	
type	name=textfield3	达内教育
waitForValue	name=passwords	1234
waitForValue	xpath=//*[@name='sex']	on
waitForValue	name=profession	艺术/设计
clickAndWait	name=Submit	

Command

waitForValue

Target

name=profession

Select

Find

Value

艺术/设计

Log	Reference	UI-Element	Rollup
[info]	Executing:	click name=computer	
[info]	Executing:	click name=film	
[info]	Executing:	click name=painting	
[info]	Executing:	type name=textfield3 达内教育	
[info]	Executing:	waitForValue name=passwords 1234	
[info]	Executing:	waitForValue xpath=//*[@name='sex'] on	
[info]	Executing:	waitForValue name=profession 艺术/设计	
[info]	Executing:	clickAndWait name=Submit	
[info]	Test case passed		

等待案例（续6）

- 练习：完成任务4：个人爱好选中“影视娱乐”和“绘画书法”

waitForValue	name=profession	艺术/设计
waitForValue	name=film	on
waitForValue	name=painting	on
clickAndWait	name=Submit	

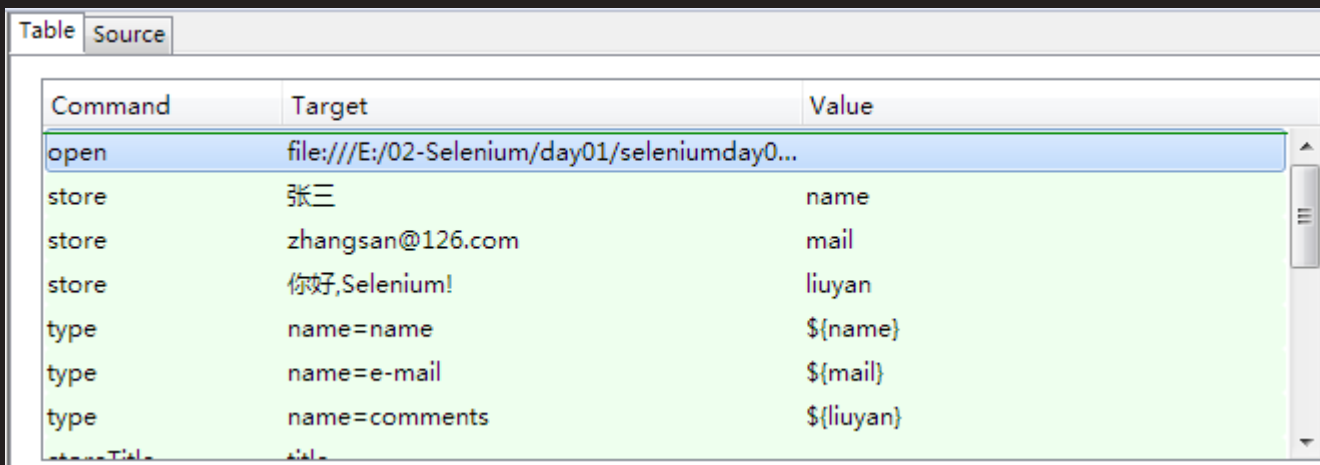
Command	waitForValue		
Target	name=film	Select	Find
Value	on		

Log	Reference	UI-Element	Rollup
[info]	Executing:	click name=painting	
[info]	Executing:	type name=textfield3 达内教育	
[info]	Executing:	waitForValue name=passwords 1234	
[info]	Executing:	waitForValue xpath=(//input[@name='sex'])[2] on	
[info]	Executing:	waitForValue name=profession 艺术/设计	
[info]	Executing:	waitForValue name=film on	
[info]	Executing:	waitForValue name=painting on	
[info]	Executing:	clickAndWait name=Submit	
[info]	Test case passed		



导出Python代码

- 在Selenium IDE中的Table视图中，是以表格形式存储每一条Selenese命令



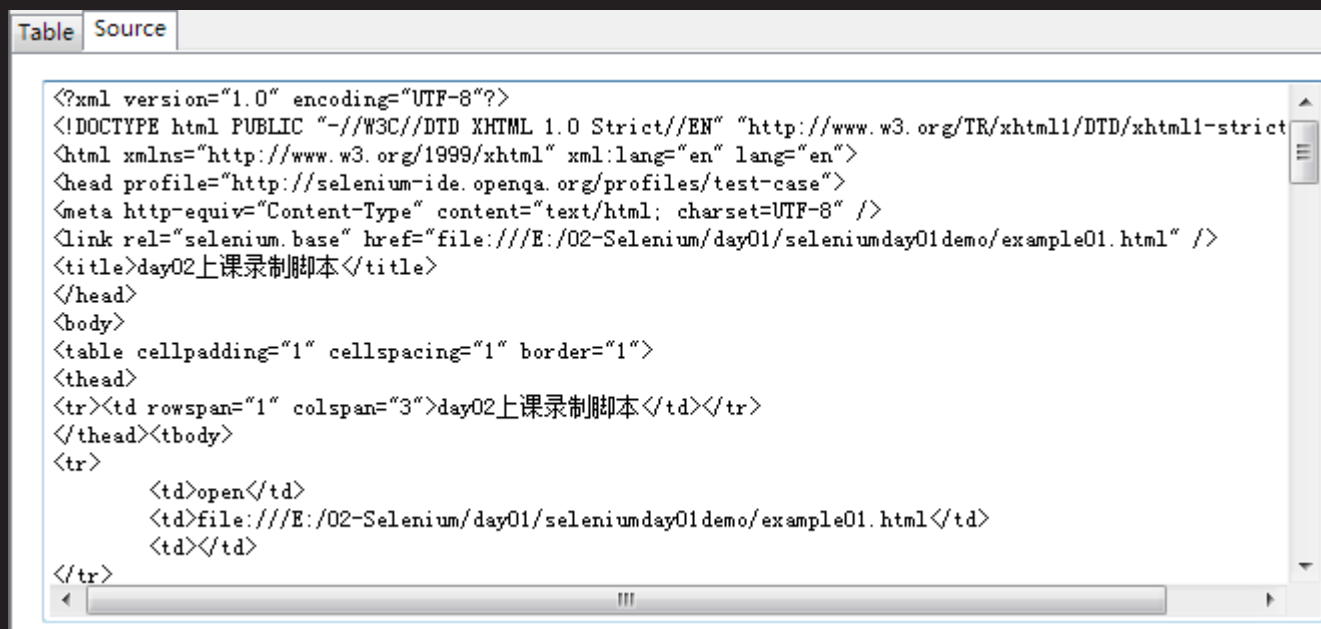
The screenshot shows the Selenium IDE interface with the 'Table' tab selected. It displays a table of commands and their targets/values. The first row is highlighted in blue.

Command	Target	Value
open	file:///E:/02-Selenium/day01/seleniumday0...	
store	张三	name
store	zhangsan@126.com	mail
store	你好,Selenium!	liuyan
type	name=name	\${name}
type	name=e-mail	\${mail}
type	name=comments	\${liuyan}



概述（续1）

- 在Source视图中，是与Table对应的测试脚本代码，默认情况下，是HTML代码



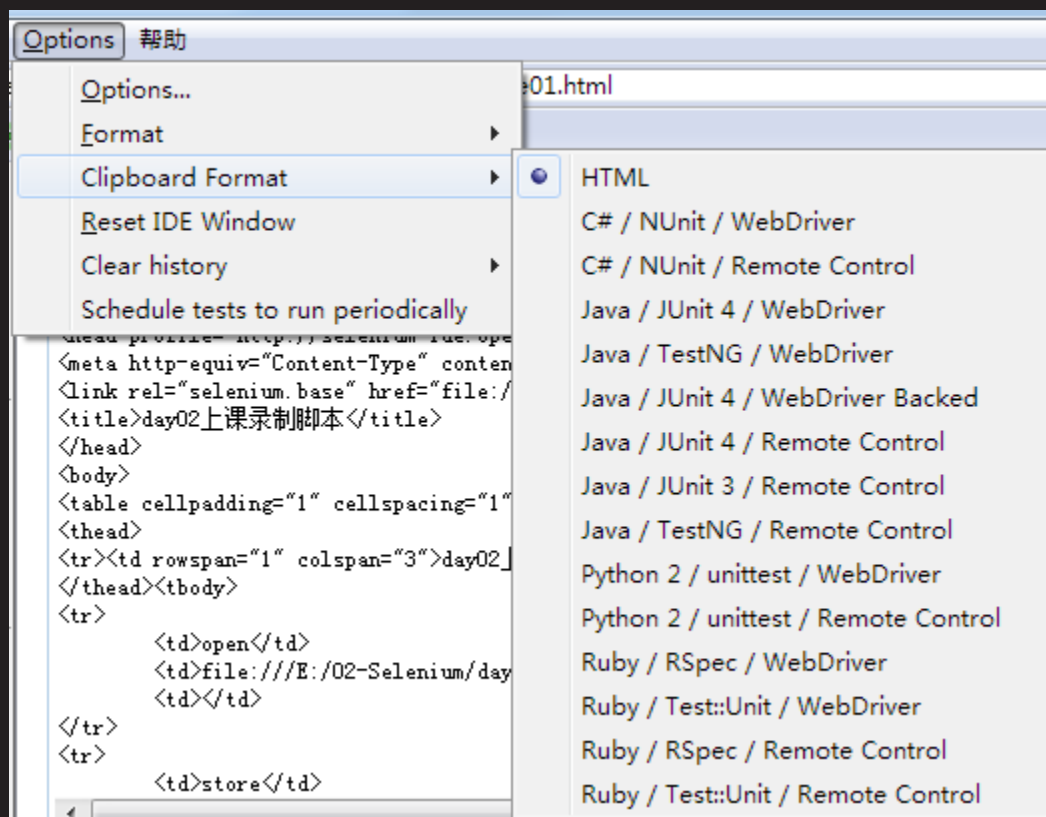
The screenshot shows the 'Source' tab of the Selenium IDE interface. The code is an XML document that defines an HTML table. The table has one row with three columns. The first column contains the text 'open', the second column contains a file path, and the third column is empty. The title of the document is 'day02上课录制脚本'.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict" [
  <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
    <head profile="http://selenium-ide.openqa.org/profiles/test-case">
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
      <link rel="selenium.base" href="file:///E:/02-Selenium/day01/seleniumday01demo/example01.html" />
      <title>day02上课录制脚本</title>
    </head>
    <body>
      <table cellpadding="1" cellspacing="1" border="1">
        <thead>
          <tr><td rowspan="1" colspan="3">day02上课录制脚本</td></tr>
        </thead><tbody>
          <tr>
            <td>open</td>
            <td>file:///E:/02-Selenium/day01/seleniumday01demo/example01.html</td>
            <td></td>
          </tr>
        </tbody>
      </table>
    </body>
  </html>
]
```



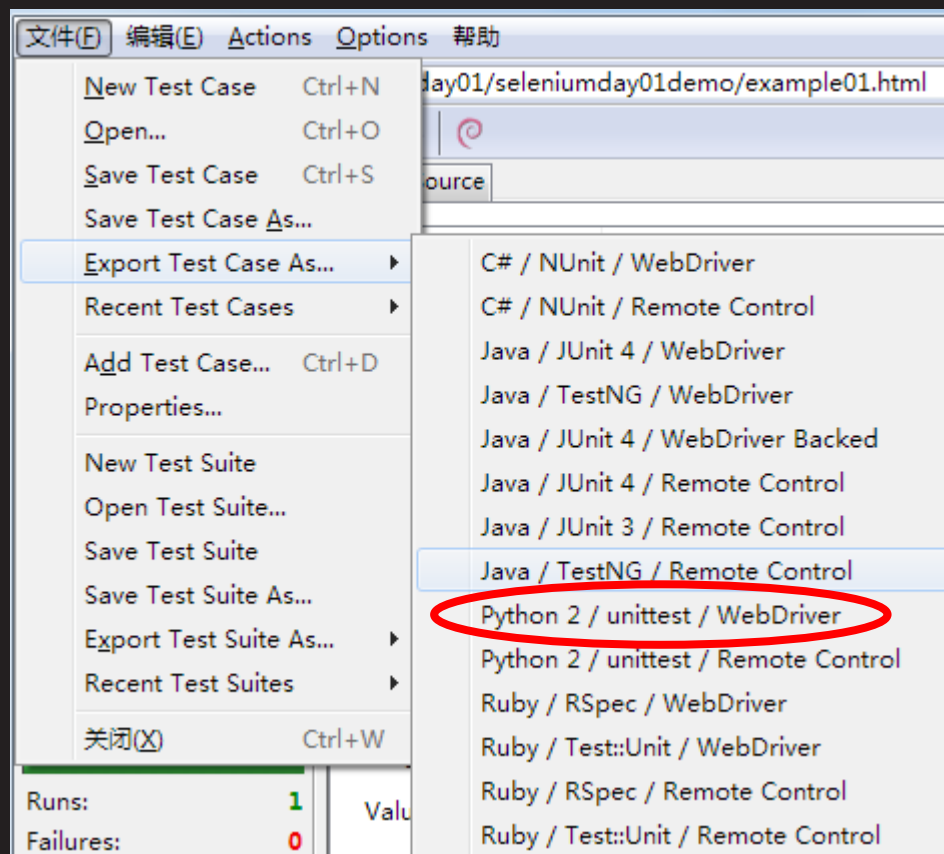
概述（续2）

- Selenium IDE支持多种语言的代码格式，如：HTML、C#、Java、Python、Ruby



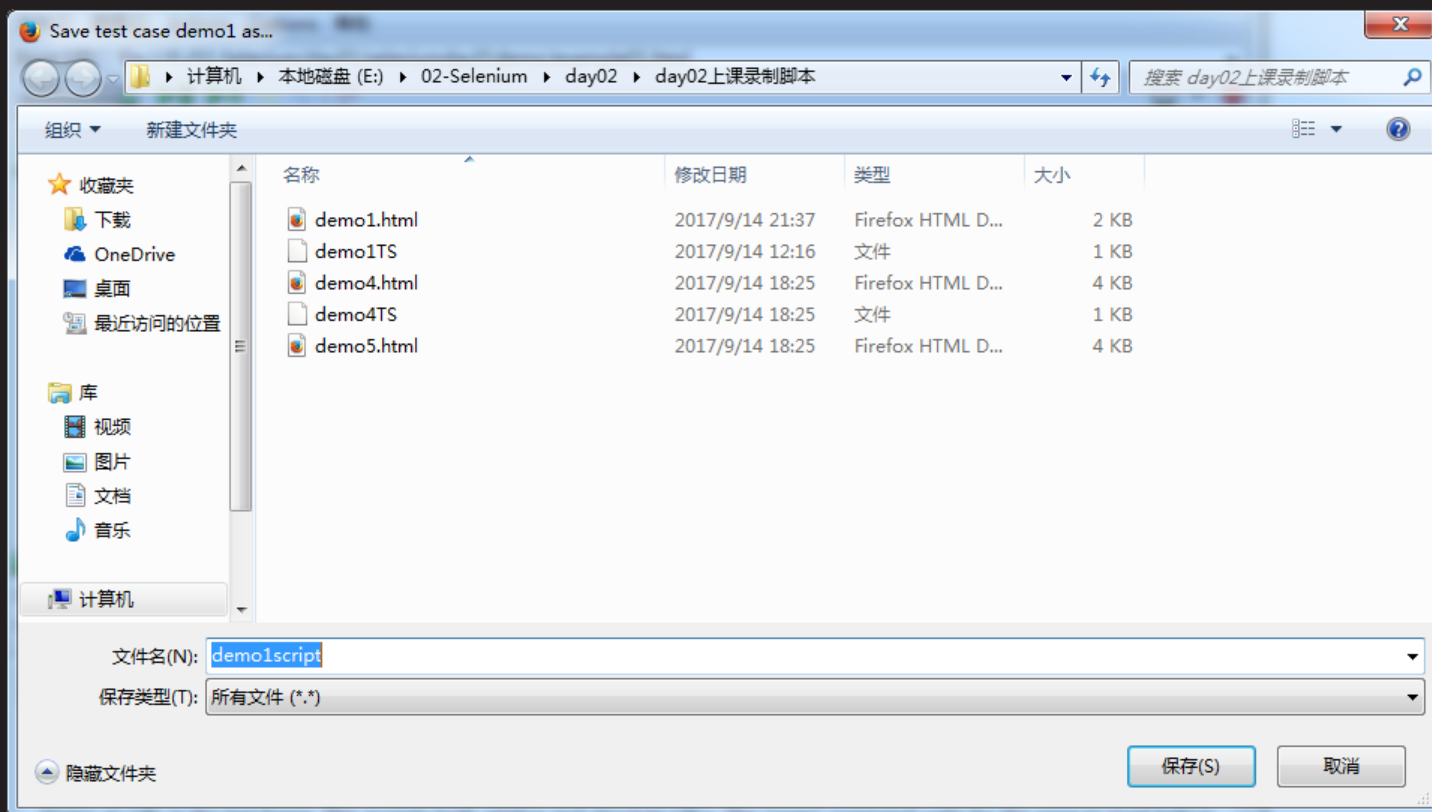
导出方式1

- 导出方式1：
 - 文件->Export Test Case as->Python 2/unittest/WebDriver



导出方式1(续1)

– 保存Python源文件名称



导出方式1(续2)

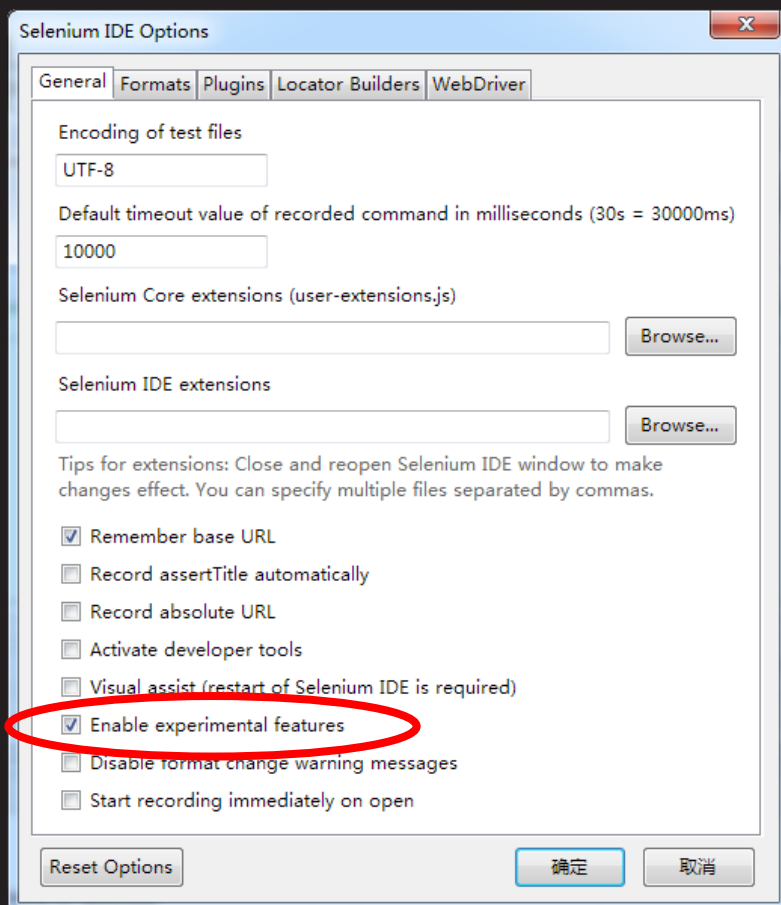
- 使用Pycharm打开导出Python源文件：

```
demolscript.py x
1  # -*- coding: utf-8 -*-
2  from selenium import webdriver
3  from selenium.webdriver.common.by import By
4  from selenium.webdriver.common.keys import Keys
5  from selenium.webdriver.support.ui import Select
6  from selenium.common.exceptions import NoSuchElementException
7  from selenium.common.exceptions import NoAlertPresentException
8  import unittest, time, re
9
10 class Demolscript(unittest.TestCase):
11     def setUp(self):
12         self.driver = webdriver.Firefox()
13         self.driver.implicitly_wait(30)
14         self.base_url = "file:///E:/02-Selenium/day01/seleniumday01demo/example01.html"
15         self verificationErrors = []
16         self.accept_next_alert = True
17
18     def test_demolscript(self):
19         driver = self.driver
20         driver.get(self.base_url + "file:///E:/02-Selenium/day01/seleniumday01demo/example01.html")
21         name = u"张三"
22         mail = "zhangsan@126.com"
23         liuyan = u"你好, Selenium!"
24         driver.find_element_by_name("name").clear()
25         driver.find_element_by_name("name").send_keys(name)
26         driver.find_element_by_name("e-mail").clear()
27         driver.find_element_by_name("e-mail").send_keys(mail)
28         driver.find_element_by_name("comments").clear()
29         driver.find_element_by_name("comments").send_keys(liuyan)
```



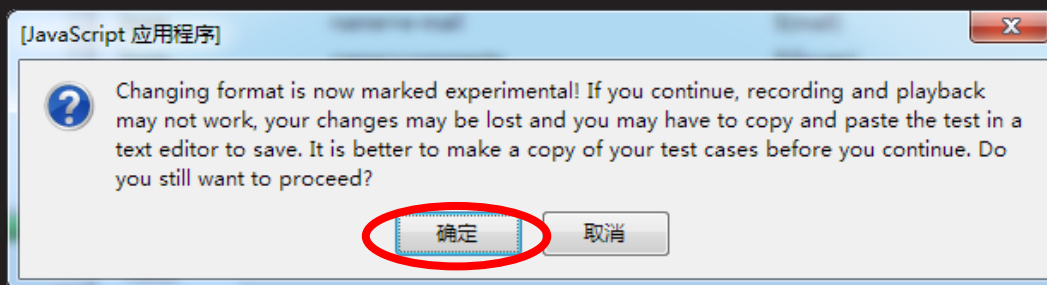
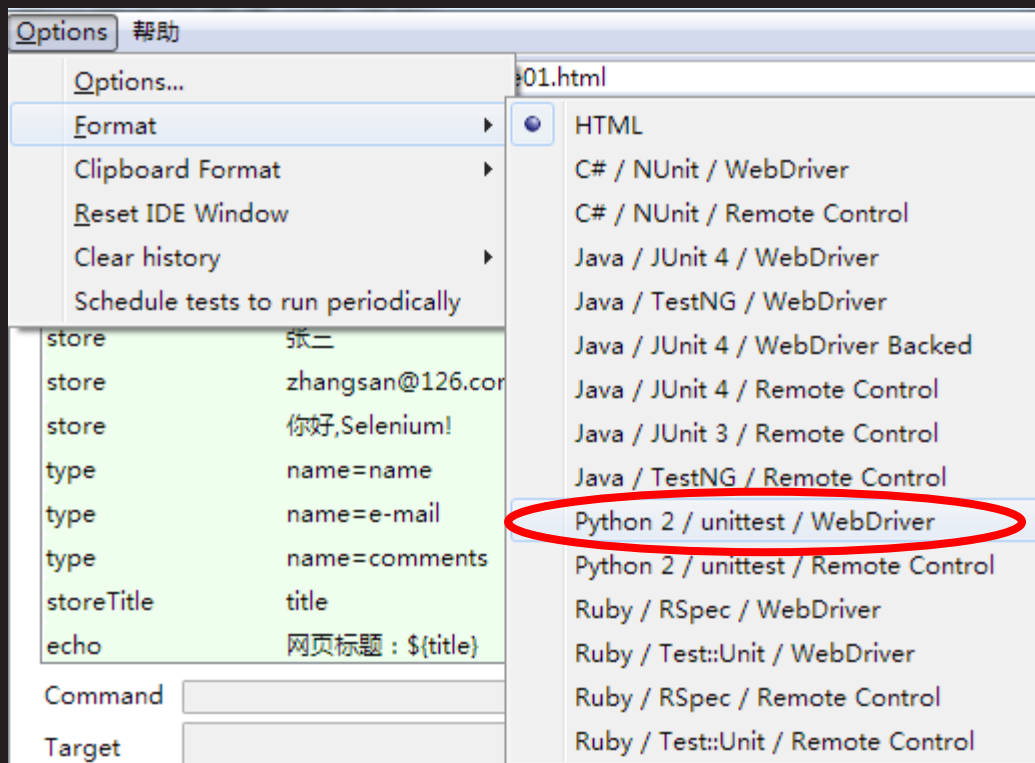
导出方式2

- 导出方式2：
 - 设置Selenium IDE Options
 - 勾选 “Enable experimental features” （可以使用试验的一些特性）



导出方式2(续1)

- 菜单：Options->Format->Python 2/unittest/WebDriver



导出方式2(续2)

- 在Source视图得到Python代码：

```
Table Source
# -*- coding: utf-8 -*-
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import Select
from selenium.common.exceptions import NoSuchElementException
from selenium.common.exceptions import NoAlertPresentException
import unittest, time, re

class Demol(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Firefox()
        self.driver.implicitly_wait(30)
        self.base_url = "file:///E:/02-Selenium/day01/seleniumday01demo/example01.html"
        self.verificationErrors = []
        self.accept_next_alert = True

    def test_demol(self):
        driver = self.driver
        driver.get(self.base_url + "file:///E:/02-Selenium/day01/seleniumday01demo/example01.html")
        name = u"张三"
        mail = "zhangsan@126.com"
        liuyan = u"你好, Selenium!"
        driver.find_element_by_name("name").clear()
        driver.find_element_by_name("name").send_keys(name)
        driver.find_element_by_name("e-mail").clear()
        driver.find_element_by_name("e-mail").send_keys(mail)
        driver.find_element_by_name("comments").clear()
        driver.find_element_by_name("comments").send_keys(liuyan)
        title = driver.title
        print(u"网页标题: " + title)
```



总结和答疑
