

# QTP

## 入门到高级

---

## 目录

第一章	QTP 简介 .....	2
1.1	自动化测试的好处 .....	2
1.2	QuickTest 工作流程 .....	2
1.3	QTP 程序界面 .....	3
1.4	Mercury Tours 示范网站 .....	5
第二章	录制/执行测试脚本 .....	5
2.1	录制前的准备 .....	6
2.2	录制测试脚本 .....	6
2.2.1	录制测试脚本 .....	6
2.2.2	分析录制的测试脚本 .....	8
2.3	执行测试脚本 .....	10
2.3.2	执行脚本出现错误 .....	11
2.4	分析测试结果 .....	11
第三章	建立检查点 .....	12
3.1	QuickTest 检查点种类 .....	13
3.2	创建检查点 .....	13
3.2.1	对象检查 .....	14
3.2.2	网页检查 .....	16
3.2.3	文字检查 .....	17
3.2.4	表格检查 .....	18
3.3	执行并分析使用检查点的测试脚本 .....	20
第四章	参数化 .....	24
4.1	参数化步骤和检查点中的值 .....	24
4.1.1	参数化对象和检查点的属性值 .....	24
4.1.2	参数化操作的值 .....	25
4.2	参数种类 .....	26
4.2.1	使用数据表参数 .....	27
4.2.2	使用环境变量参数 .....	28
4.2.3	使用随机数字参数 .....	28
4.3	参数化测试脚本 .....	29
4.3.1	定义参数 .....	29
4.3.2	修正受到参数化影响的步骤 .....	31
4.3.3	执行并分析使用参数的测试脚本 .....	32
第五章	输出值 .....	33
5.1	创建输出值 .....	34
5.1.1	输出值类型 .....	34
5.1.2	存储输出值 .....	35
5.2	输出属性值 .....	36
5.2.1	定义标准输出值 .....	36
5.2.2	指定输出类型和和设置 .....	37
5.3	在脚本中建立输出值 .....	39
5.3.1	建立输出值 .....	39
5.3.2	执行并分析使用输出值的测试脚本 .....	41

---

# 第一章 QTP 简介

## 1.1 自动化测试的好处

如果你执行过人工测试，你一定了解人工测试的缺点，人工测试非常浪费时间而且需要投入大量的人力。使用人工测试的结果，往往是在应用程序交付前，无法对应用程序的所有功能都作完整的测试。

使用 QuickTest 可以加速整个测试的过程，并且建置完新版本的应用程序或网站后，可以重复使用测试脚本进行测试。

以 QuickTest 执行测试，就与人工测试一样。QuickTest 会仿真鼠标的动作与键盘的输入，不过 QuickTest 比人工测试快了很多。

自动化测试的好处	
快速	QuickTest 执行测试比人工测试速度快多了。
可靠	QuickTest 每一次的测试都可以正确的执行相同的动作，可以避免人工测试的错误。
可重复	QuickTest 可以重复执行相同的测试。
可程序化	QuickTest 可以以程序的方式，撰写复杂的测试脚本，以带出隐藏在应用程序中的信息。
广泛性	QuickTest 可以建立广泛的测试脚本，涵盖应用程序的所有功能。
可再使用	QuickTest 可以重复使用测试脚本，即使应用程序的使用接口已经改变。

## 1.2 QuickTest 工作流程

### 1. 录制测试脚本前的准备

在测试前需要确认你的应用程序及 QuickTest 是否符合测试需求？

确认你已经知道如何对应用程序进行测试，如要测试哪些功能、操作步骤、预期结果等。

同时也要检查一下 QuickTest 的设定，如 Test Settings 以及 Options 对话框，以确保 QuickTest 会正确的录制并储存信息。确认 QuickTest 以何种模式储存信息。

### 2. 录制测试脚本

操作应用程序或浏览网站时，QuickTest 会在 Keyword View 中以表格的方式显示录制的操作步骤。每一个操作步骤都是使用者在录制时的操作，如在网站上点击了链接，或则在文本框中输入的信息。

### 3. 加强测试脚本

在测试脚本中加入检查点，可以检查网页的链接、对象属性、或者字符串，以验证应用程序的功能是否正确。

将录制的固定值以参数取代，使用多组的数据测试程序。使用逻辑或者条件判断式，可以进行更复杂的测试。

### 4. 对测试脚本进行调试

修改过测试脚本后，需要对测试脚本作调试，以确保测试脚本能正常并且流畅的执行。

#### 5. 在新版应用程序或者网站上执行测试脚本

通过执行测试脚本，QuickTest 会在新本的网站或者应用程序上执行测试，检查应用程序的功能是否正确。

#### 6. 分析测试结果

分析测试结果，找出问题所在。

#### 7. 测试报告

如果你安装了 TestDirector (Quality Center)，则你可以将发现的问题汇报到 TestDirector (Quality Center) 数据库中。TestDirector (Quality Center) 是 Mercury 测试管理工具。

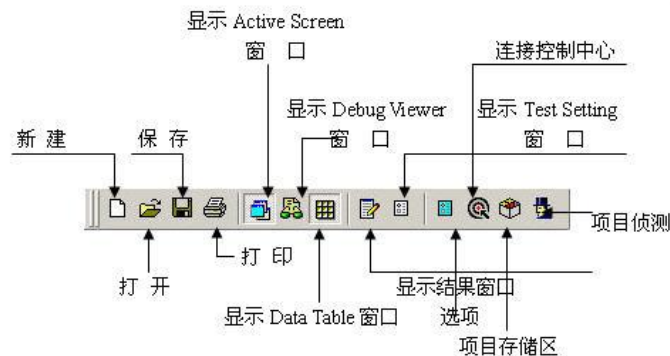
## 1.3 QTP 程序界面

在学习创建测试之前，先了解一下 QuickTest 的主界面。下图是录制了一个操作后 QuickTest 的界面。

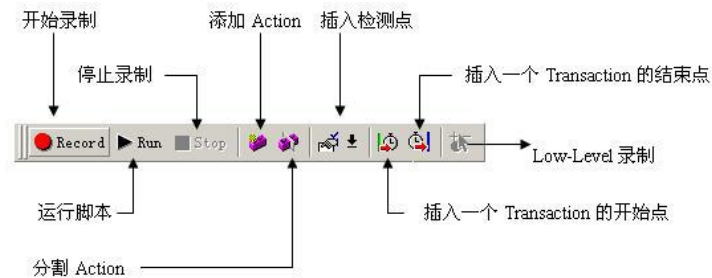


在 QTP 界面包含标题栏、菜单栏、文件工具条等几个界面元素，下面简单解释各界面元素的功能：

- n 标题栏，显示了当前打开的测试脚本的名称。
- n 菜单栏，包含了 QuickTest 的所有菜单命令项。
- n 文件工具条，在工具条上包含了以下几个按钮：



- n 测试工具条，包含了在创建、管理测试脚本是要使用的按钮，如下图：



- n 调试工具条，包含在调试测试脚本时要使用的工具条，如下图：



- n 测试脚本管理窗口，提供了两个可切换的窗口，分别通过图形化方式和 VBScript 脚本方式来管理测试脚本。
- n Data Table 窗口，用于参数化你的测试。
- n 状态栏，显示测试过程中的状态。

在上面上面简要介绍了 QuickTest 的主窗口，你可能对一些窗口元素到底是干什么的感到很困惑，在我们下面介绍 QuickTest 具体的功能时，会真正了解它们的作用。但在现在，应该尽可能的去熟悉这些界面元素，记住它们大概的功能，最好是花一些时间通过实际的操

作来探索一下它们的功能，这对你能够顺利学习下面的内容是有帮助的。

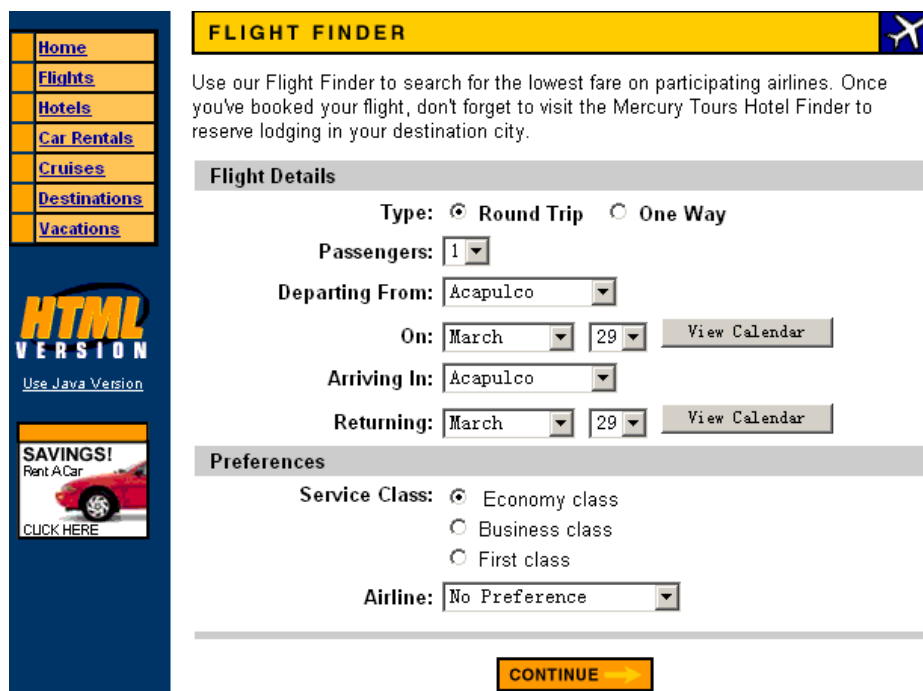
## 1.4 Mercury Tours 示范网站

Mercury Tours 示范网站是一个提供机票预订服务的网站，在本课程中，我们使用 MI 公司提供的 Mercury Tours 示范网站作为演示 QuickTest 各个功能的例子程序。

1. 在开始使用 Mercury Tours 示范网站（<http://newtours.demoaut.com/index.php>）之前，首先要在 Mercury Tours 网站上注册一个使用者账号。

2. Mercury Tours 示范网站使用

要登录并使用 Mercury Tours 示范网站必须使用注册账号。



The screenshot shows the 'FLIGHT FINDER' section of the Mercury Tours website. On the left is a vertical navigation menu with links: Home, Flights, Hotels, Car Rentals, Cruises, Destinations, and Vacations. Below the menu is a banner for 'HTML VERSION' and 'Use Java Version', and a 'SAVINGS! Rent A Car' advertisement with a 'CLICK HERE' button. The main form area has a yellow header 'FLIGHT FINDER' with an airplane icon. Below the header is a text block: 'Use our Flight Finder to search for the lowest fare on participating airlines. Once you've booked your flight, don't forget to visit the Mercury Tours Hotel Finder to reserve lodging in your destination city.' The form is divided into 'Flight Details' and 'Preferences' sections. 'Flight Details' includes: 'Type' (radio buttons for Round Trip and One Way), 'Passengers' (a dropdown menu showing 1), 'Departing From' (a dropdown menu showing Acapulco), 'On' (month and day dropdowns showing March and 29, with a 'View Calendar' button), 'Arriving In' (a dropdown menu showing Acapulco), and 'Returning' (month and day dropdowns showing March and 29, with a 'View Calendar' button). 'Preferences' includes: 'Service Class' (radio buttons for Economy class, Business class, and First class) and 'Airline' (a dropdown menu showing No Preference). At the bottom of the form is a yellow 'CONTINUE' button.

在使用网站时，从 [Flight Finder] 网页开始，按照画面上的指示预订机票。在 Book a Flight 网页，无需填写真实的旅客信息，信用卡卡号等标示为红色的字段中添加虚拟数据就可以了。

3. 结束订票动作

完成订票动作后，在 [Flight Confirmation] 网页上点选[LOG OUT] 按钮或是选择 [SIGN-OFF]按钮。

4. 关闭浏览器

现在知道如何使用 Mercury Tours 示范网站，就可以开始使用 QuickTest 录制测试脚本了。

## 第二章 录制/执行测试脚本

当浏览网站或使用应用程序时，QuickTest 会纪录你的操作步骤，并产生测试脚本。当

---

停止录制后，会看到 QuickTest 在 Keyword View 中以表格的方式显示测试脚本的操作步骤。

## 2.1 录制前的准备

在录制脚本前，首先要确认以下几项：

- n 已经在 Mercury Tours 示范网站上注册了一个新的使用者账号。
- n 在正式开始录制一个测试之前，关闭所有已经打开的 IE 窗口。这是为了能够正常的进行录制，这一点要特别注意。
- n 关闭所有与测试不相关的程序窗口。

## 2.2 录制测试脚本

### 2.2.1 录制测试脚本

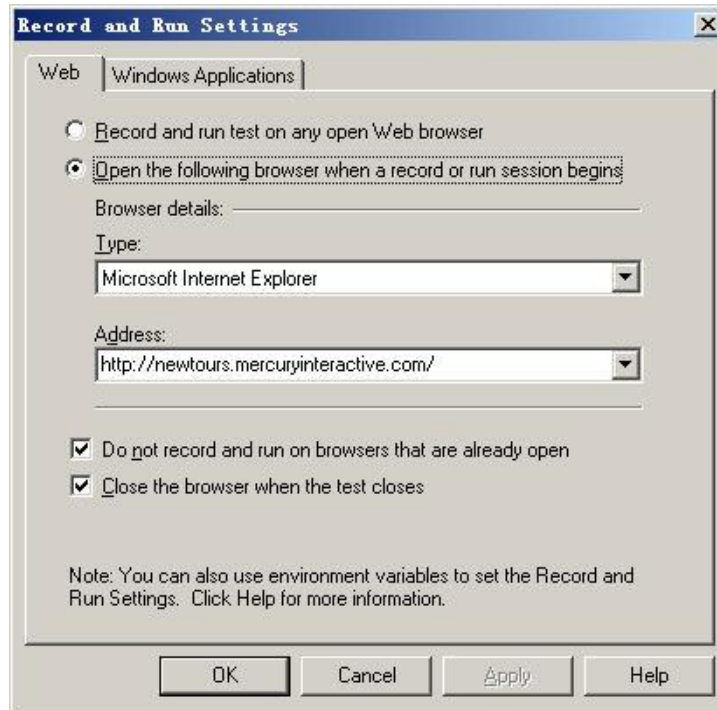
在这一节中我们使用 QuickTest 录制一个测试脚本，在 Mercury Tours 范例网站上预定一张从纽约（New York）到旧金山（San Francisco）的机票。

#### 1. 执行 QuickTest 并开启一个全新的测试脚本

- n 开启 QuickTest，在“Add-in Manager”窗口中选择“Web”选项，点击“OK”关闭“Add-in Manager”窗口，进入 QuickTest Professional 主窗口。
- n 如果 QuickTest Professional 已经启动，检查“Help>About QuickTest Professional”查看目前加载了那些 add-ins。如果没有加载“Web”，那么必须关闭并重新启动 QuickTest Professional，然后在“Add-in Manager”窗口中选择“Web”。
- n 如果在执行 QuickTest Professional 时没有开启“Add-in Manager”则点击“Tool>Options”，在“General”标签页勾选“Display Add-in Manager on Startup”，在下次执行 QuickTest Professional 时就会看到“Add-in Manager”窗口了

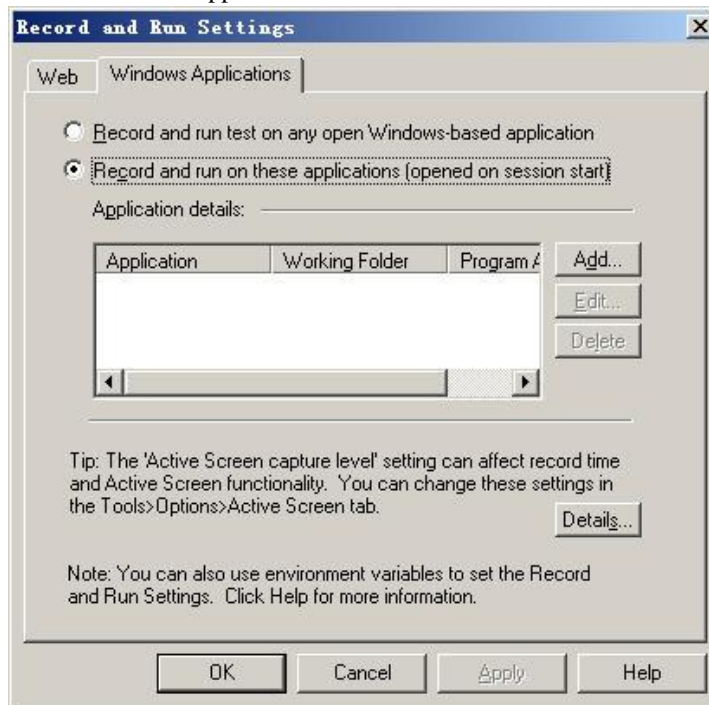
#### 2. 开始录制测试脚本

选中“Test>Record”或者点选工具栏上的“Record”按钮。打开“Record and Run Settings”对话框：



在“Web”标签页选择“Open the following browser when a record or run session begins”  
在“Type”下拉列表中选择“Microsoft Internet Explorer”为浏览器的类型；在“Address”中添加“<http://newtours.mercuryinteractive.com/>（网站地址）”这样，在录制的时候，QuickTest会自动打开 IE 浏览器并连接到 Mercury Tours 范例网站上。

现在我们在切换到“Windows Application”标签页，如下图所示：





---

如果选择“Record and run test on any open Windows-based application”单选按钮，则在录制过程中，QuickTest 会记录你对所有的 Windows 程序所做的操作。如果选择“Record and run on these application(opened when a session begins)”单选按钮，则在录制过程中，QuickTest 只会记录对那些添加到下面“Application details”列表框中的应用程序的操作（你可以通过“Add”、“Edit”、“Delete”按钮来编辑这个列表）。

我们选择第二个单选按钮。因为我们只是对 Mercury Tours 范例网站进行操作，不涉及到 Windows 程序，所以保持列表为空。

点击“确定”按钮，开始录制了，将自动打开 IE 浏览器并连接到 Mercury Tours 范例网站上。

### 3. 登录 Mercury Tours 网站

在用户名和密码输入注册时使用的账号和密码，点击“Sign-in”，进入“Flight Finder”网页。

### 4. 输入订票数据

输入以下订票数据：

Departing From: New York

On: May 14

Arriving In: San Francisco

Returning: May 28

Service Class: Business class

其他字段保留默认值，点击“CONTINUE”按钮打开“Select Flight”页面。

### 5. 选择飞机航班

可以保存默认值，点击“CONTINUE”按钮打开“Book a Flight”页面。

### 6. 输入必填字段（红色字段）

输入用户名和信用卡号码（信用卡可以输入虚构的号码，如 8888-8888）。

点击网页下方的“SECURE PURCHASE”按钮，打开“Flight Confirmation”网页。

### 7. 完成定制流程

查看订票数据，并选择“BACK TO HOME”回到 Mercury Tours 网站首页。

### 8. 停止录制

在 QuickTest 工具列上点击“Stop”按钮，停止录制。

到这里已经完成了预定从“纽约-旧金山”机票的动作，并且 QuickTest 已经录制了从按下“Record”按钮后到“Stop”按钮之间的所有操作。

### 9. 保存脚本

选择“File>Save”或者电机工具栏上的“Save”按钮，开启“Save”对话框。选择的路径，填写文件名，我们取名为 Flight。点击“保存”按钮进行保存。

通过以上九个步骤，我们录制了一个完整的测试脚本—预定从纽约到旧金山的机票。

## 2.2.2 分析录制的测试脚本

在录制过程中，QuickTest 会在测试脚本管理窗口（也叫 Tree View 窗口）中产生对每一个操作的相应记录。并在 Keyword View 中以类似 Excel 工作表的方式显示所录制的测试脚本。当录制结束后，QuickTest 也就记录下了测试过程中的所有操作。测试脚本管理窗口显示的内容如下图所示：

Item	Operation	Value	Assignment	Comment	Documentation
Action1					
Welcome: Mercury Tours					
userName	Set	"jojo"		Enter "jojo" in the "userName" edit box	
password	SetSecure	"446845bf84444adc2E56C863C0185254"		Enter the encrypted string "446845bf84444adc2E56C863C0185254" in the "password" edit box	
Sign-In	Click	41,4		Click the "Sign-In" image	
Find a Flight: Mercury					
fromPort	Select	"New York"		Select the "New York" item in the "fromPort" list	
toPort	Select	"San Francisco"		Select the "San Francisco" item in the "toPort" list	
toCity	Select	"29"		Select the "29" item in the "toCity" list	
airClass	Select	"Business"		Select radio button "Business" in the "airClass" radio button group	
findFlight	Click	70,12		Click the "findFlight" image	
Select a Flight: Mercury	Click	70,12		Click the "selectFlight" image	
Book a Flight: Mercury					
confirmFirst	Set	""		Enter "" in the "confirmFirst" edit box	
confirmLast	Set	"jojo"		Enter "jojo" in the "confirmLast" edit box	
creditNumber	Set	"3000 0000"		Enter "3000 0000" in the "creditNumber" edit box	
bookFlight	Click	82,13		Click the "bookFlight" image	
Flight Confirmation: Mercury					
Home	Click			Click the "Home" image	
Welcome: Mercury Tours 2	Stop			Stop for the video case or send message before continuing the run.	

在 Keyword View 中的每一个字段都有其意义：

- n **Item:** 以层次式的图标表示这个操作步骤所作用的组件（测试对象、工具对象、函数呼叫或脚本）。
- n **Operation:** 要在这个作用到的组件上执行的动作，如点击、选择等。
- n **Value:** 执行动作的参数，例如当鼠标点击一张图片时是用左键还是右键。
- n **Assignment:** 使用到的变量。
- n **Comment:** 你在测试脚本中加入的批注。
- n **Documentation:** 自动产生用来描述此操作步骤的英文说明。

脚本中的每一个步骤在 **Keyword View** 中都会以一系列来显示，其中用来表示此组件类别的图标以及步骤的详细数据。

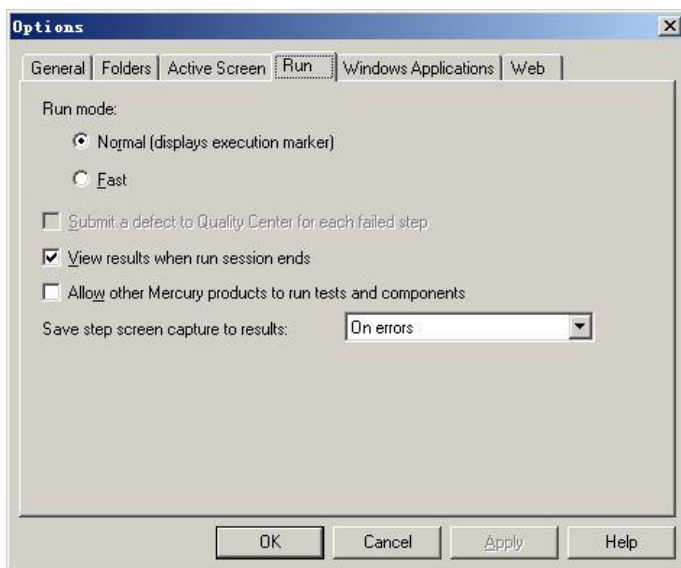
下面我们针对一些常见的操作步骤作详细说明：

步骤	说明
 Action1	Action1 是一个动作的名称
 Welcome: Mercury Tours	Welcome: Mercury 是被浏览器开启的网站名称
 Welcome: Mercury Tours	Welcome: Mercury Tours 是网页的名称
 userName    Set    "jojo"	userName 是 edit box 的名称 Set 是在这个 edit box 上执行的动作 jojo 是被输入得值
 password    SetSecure    "446845bf84444adc2E56C863C0185254"	password 是 edit box 的名称 SetSecure 是在这个 edit box 上执行的动作，此动作有加密的功能 446845bf84444adc...是被加密过的密码
 Sign-In    Click    41,4	Sign-In 是图像对象的名称 Click 是在这个图像上执行的动作 41, 4 则是这个图像被点击的 X, Y 坐标

## 2.3 执行测试脚本

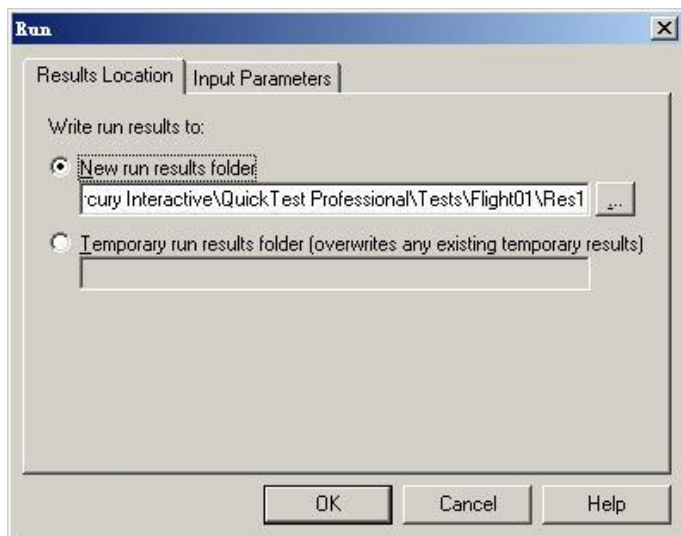
当运行录制好的测试脚本时，QuickTest 会打开被测试程序，执行你在测试中录制的每一个操作。测试运行结束后，QuickTest 显示本次运行的结果。接下来，我们执行在上一节中录制的 Flight 测试脚本。

1. 打开录制的 Flight 测试脚本。
2. 设置运行选项。点击“Tool>Options”打开设置选项对话框，选择“Run”标签页，如下图：



如果要将所有画面储存在测试结果中，在“Save step screen capture to results”选项中选择“Always”选项。一般情况下我们选择“On error”或“On error and warning”表示在回放测试过程中出现问题时，才保存图象信息。在这里我们为了更多的展示 QuickTest 的功能，所以选择使用“Always”选项。

3. 在工具条上点击“Run”按钮，打开“Run”对话框：



询问要将本次的测试运行结果保存到何处。选择“New Run results folder”单选按钮，设定好存放路径（在这使用预设的测试结果名称）。

4. 点击“OK”按钮开始执行测试。

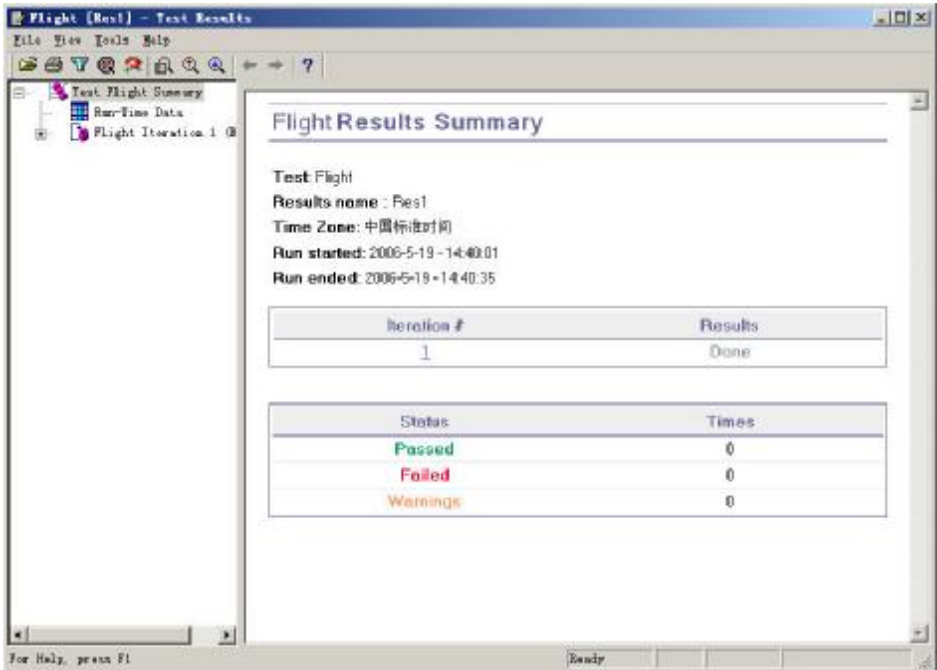
可以看到 QuickTest 按照你在脚本中录制的操作，一步一步的运行测试，操作过程与你手工操作时完全一样。同时可以在 QuickTest 的 Keyword View 中会出现一个黄色的箭头，指示目前正在执行的测试步骤。

### 2.3.2 执行脚本出现错误

如果在执行测试的时候出现错误，会显示一个错误信息对话框？

## 2.4 分析测试结果

在测试执行完成后，QuickTest 会自动显示测试结果窗口，如下图所示：



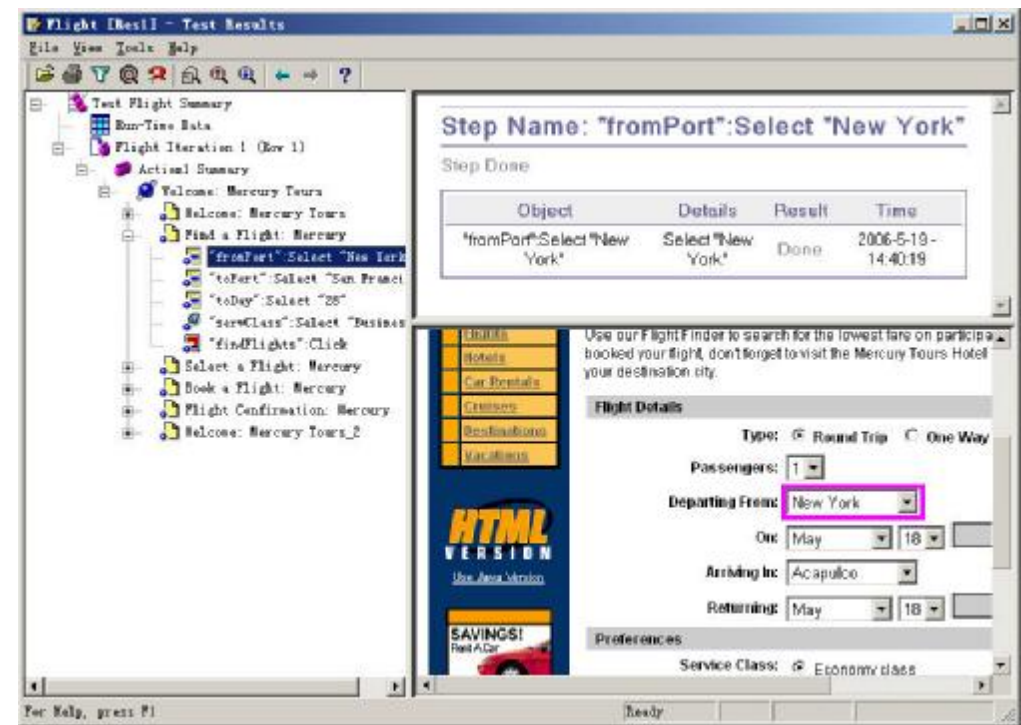
在这个测试结果窗口中分二个部分显示测试执行的结果

- n 左边显示 Test results tree，以阶层图标的方式显示测试脚本所执行的步骤。可以选择“+”检查每一个步骤，所有的执行步骤都会以图示的方式显示。可以设定 QuickTest 以不同的资料执行每个测试或某个动作，每执行一次反复称为一个迭代，每一次迭代都会被编号（在上面的例子中只执行了一次迭代）。
- n 右边则是显示测试结果的详细信息。在第一个表格中显示哪些迭代是已经通过的，哪些是失败的。第二个表格是显示测试脚本的检查点，哪些是通过的，哪些是失败的，以及有几个警告信息。

在上面的测试中，所有的测试都是通过的，在脚本中也没有添加检查点（有关检查点的

内容我们将在以后的课程中学习)。接下来我们查看 QuickTest 执行测试脚本的详细结果，以及选择某个测试步骤时出现的详细信息。

在树视图中展开 “Flight Iteration 1(Row 1)>Action1 Summary>Welcome Mercury Tours>Find a Flight: Mercury>”，选择 “ ” fromPost ” : Select ” New York ” ”。



在这个测试结果窗口中显示三个部分，分别是：

- n 左边是 Test results tree: 展开树视图后，显示了测试执行过程中的每一个操作步骤。选择某一个测试步骤，会在右边区域显示相应的信息。
- n 右上方是 Test results detail: 对应当前选中的测试步骤，显示被选取测试步骤执行时的详细信息。
- n 右下方是 Active Screen: 对应当前选中的测试步骤，显示该操作执行时应用程序的屏幕截图。

当选中 test results tree 上的网页图示，会在 “Active Screen” 中看到执行时的画面。当选中 test results tree 上的测试步骤(在某个对象上执行某个动作)，除了显示当前时的画面外，对象还会被粉色的框框框住。在上面的例子中，在 “Active Screen” 中点击被框住的 “Departing From” 下拉菜单，会显示其他的选项。

### 第三章 建立检查点

通过上一章的学习，我们已经掌握了如何录制、执行测试脚本以及查看测试结果。但是我们只是实现了测试执行的自动化，没有实现测试验证的自动化，所以这并不是真正的自动

化测试。在这一章我们学习如何在测试脚本中设置检查点，以验证执行结果的真确性。

“检查点”是将指定属性的当前值与该属性的期望值进行比较的验证点。这能够确定网站或应用程序是否正常运行。当添加检查点时，QuickTest 会将检查点添加到关键字视图中的当前行并在专家视图中添加一条“检查检查点”语句。运行测试或组件时，QuickTest 会将检查点的期望结果与当前结果进行比较。如果结果不匹配，检查点就会失败。可以在“测试结果”窗口中查看检查点的结果。

## 3.1 QuickTest 检查点种类

首先我们了解一下 QuickTest 支持的检查点种类，如下表所示，QuickTest 支持以下检查点：

检查点类型	说明	范例
标准检查点	检查对象的属性	检查某个按钮是否被选取
图片检查点	检查图片的属性	检查图片的来源文件是否是正确的
表格检查点	检查表格的内容	检查表格内的内容是否是正确对的
网页检查点	检查网页的属性	检查网页加载的时间或是网页是否含有不正确的链接
文字/文字区域检查点	检查网页上或是窗口上出现的文字是否正确	检查登陆系统后时候出行登陆成功的文字
图像检查点	提取网页和窗口的画面检查画面是否正确	检查网页或者网页的一部分是否如期显示
数据库检查点	检查数据库的内容时候正确	检查数据库查询的值是否正确
XML 检查点	检查 XML 文件的内容	XML 检测点有两种—XML 文件检测点和 XML 应用检测点。XML 文件检测点用于检查一个 XML 文件；XML 应用检测点用于检查一个 Web 页面的 XML 文档。

你可以在录制测试的过程中，或录制结束后，向测试脚本中添加检测点。下面我们学习如何在测试脚本上建立检查点。

## 3.2 创建检查点

打开 Flight 测试脚本，将脚本另存为“Checkpoint”测试脚本。我们在 Checkpoint 测试脚本中创建 4 个检查点，分别是：对象检查、网页检查、文字检查以及表格检查。

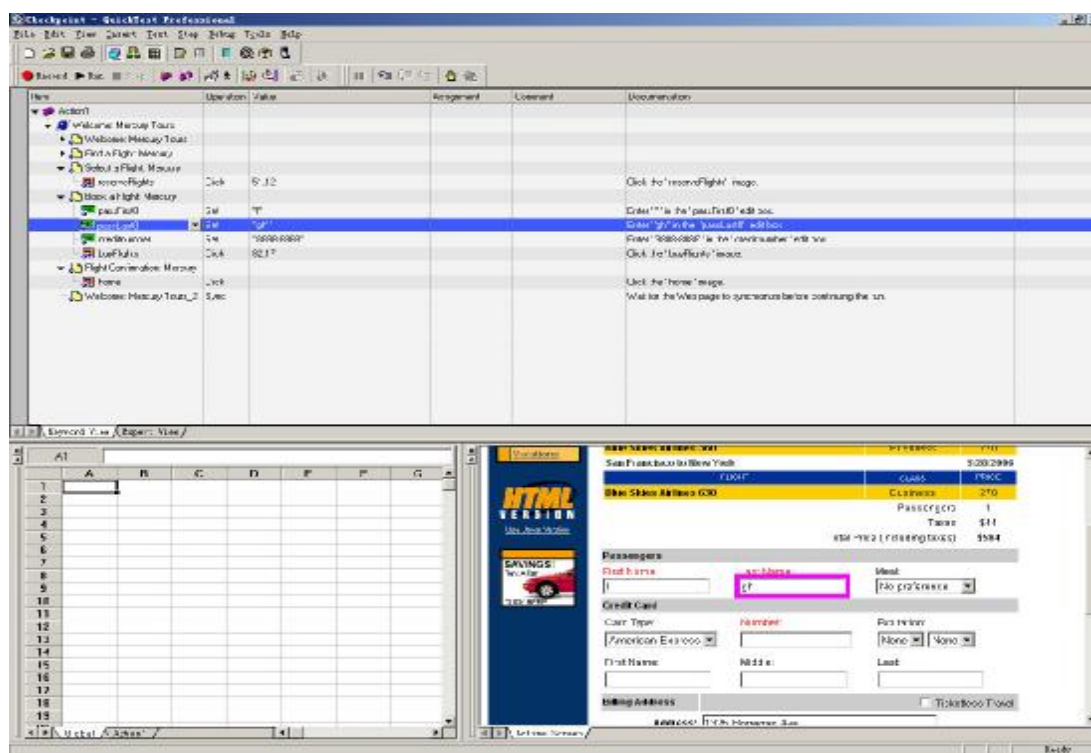
### 3.2.1 对象检查

通过向测试或组件中添加标准检查点,可以对不同版本的应用程序或网站中的对象属性值进行比较。可以使用标准检查点来检查网站或应用程序中的对象属性值。标准检查点将对录制期间捕获的对象属性的预期值,与运行会话期间对象的当前值进行比较。

首先在 Checkpoint 测试脚本上添加一个标准检查点,这个检查点用以检查旅客的姓氏。创建标准检查点:

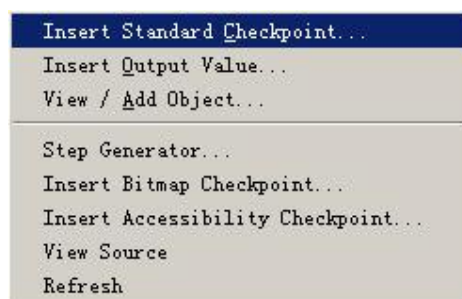
1. 打开 Checkpoint 测试脚本。
2. 选择要建立检查点的网页

在 QuickTest 的视图树中展开 “Action1>Welcome: Mercury Tours>Book a Flight: Mercury”,由于输入使用者姓氏的测试步骤是 “passFirst0” 这个步骤,所以要选择这个步骤的下一个测试步骤,以便建立检查点。如下图所示:



#### 3. 建立标准检查点

对 “Active Screen” 中的 First Name 编辑框点击鼠标右键,显示插入选择点的类型。

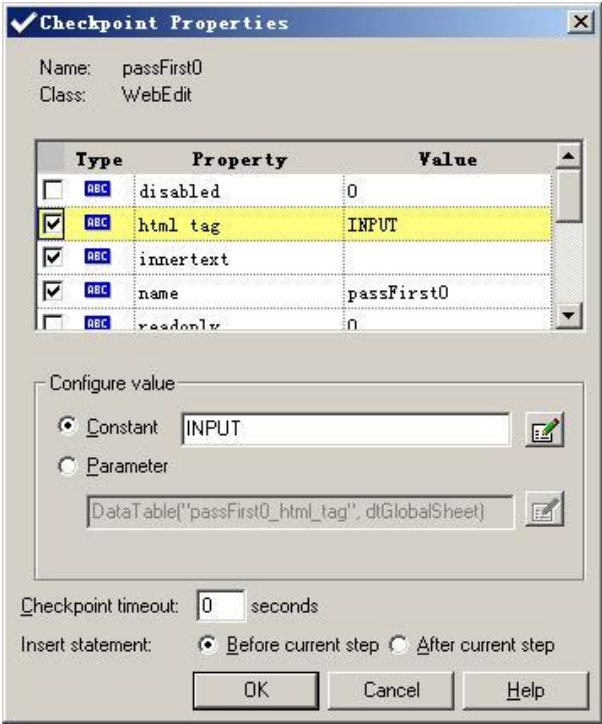




选择 “Insert Standard Checkpoint” 选型，显示 “Object Selection-Checkpoint Properties” 对话框：



确保当前的焦点定位在 “WebEdit: passFirst0” 上，点击 “OK” 按钮，弹出如下的窗口：



在检查点属性窗口会显示将查点的属性：

- n Name: 将查点的名称
- n Class: 检查点的类别，WebEdit 表示这个检查点是个输入框
- n “Type” 字段中的 “ABC” 图标：表示这个属性的值是一个常数


对于每一个检查点，QuickTest 会使用预设的属性最为检查点的属性，下表说明这些预设的属性。

属性	值	说明
html tag	INPUT	HTML 原始码中的 INPUT 标签
innertext		在这个范例中，innertext 只是空的，检查点



		会检查当执行时这个属性是不是空的
name	passFirst0	passFirst0 是这个编辑框的名称
type	text	text 是 HTML 原始码中 INPUT 对象的类型
value	姓氏（录制脚本是输入的姓氏）	在编辑框中输入的文字

我们接受预设的设定值，点击“OK”。QuickTest 会在选取的步骤之前建立一个标准检查点。



Book a Flight: Mercury				
passFirst0	Set	"T"		Enter "T" in the "passFirst0" edit box.
passFirst0	Check	CheckPoint("passFirst0")		Check whether the "passFirst0" edit box has the proper values for the selected object.
passLast0	Set	"gh"		Enter "gh" in the "passLast0" edit box.
creditnumber	Set	"0000-0000"		Enter "0000-0000" in the "creditnumber" edit box.

4. 在工具栏上点击“Save”保存脚本。

通过 1-4 的步骤，添加一个标准检查点的操作就此结束。

### 3.2.2 网页检查

我们在 Checkpoint 测试脚本中再添加一个网页检查点，网页检查点会检查网页的链接以及图像的数量时候与当前录制时的数量一致。网页检查点只能应用于 Web 页面中。

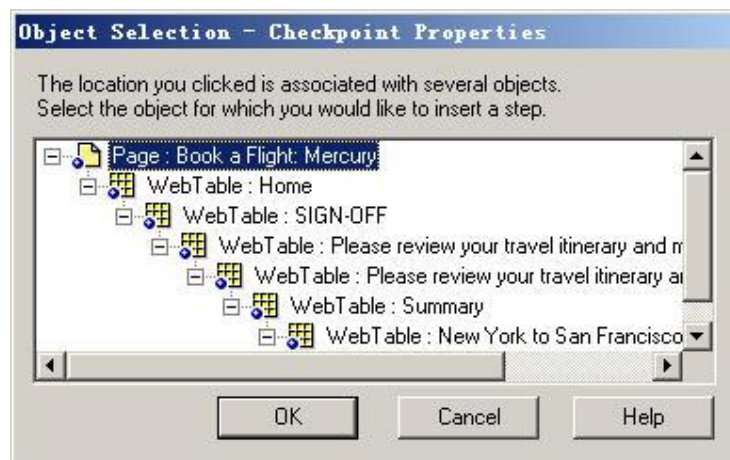
创建网页检查：

1. 选择要建立检查点的网页

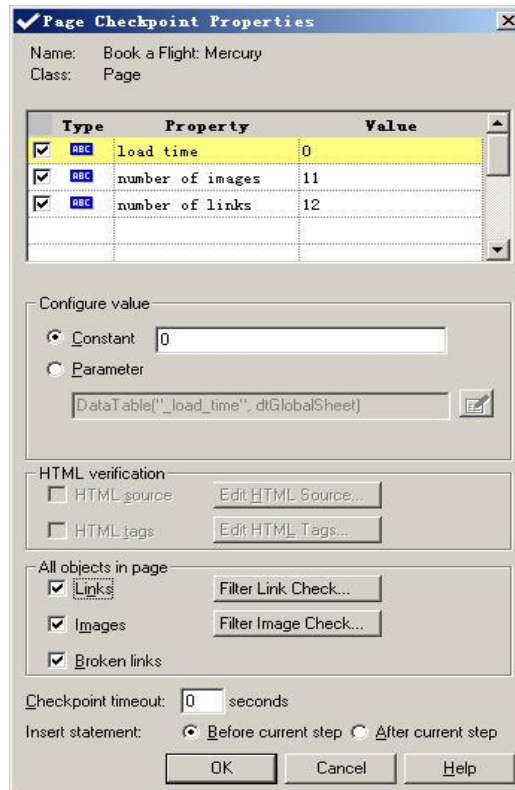
展开“Action1>Welcome: Mercury Tours”选择“Book a Flight: Mercury”页面，在“Active Screen”会显示相应的页面。

2. 建立网页检查点

在“Active Screen”上的任意地方点击鼠标右键，选取“Insert Standard Checkpoint”，开启“Object Selection-Checkpoint Properties”对话框（由于选择的位置不同，对话框显示被选取的对象可能不一样）。



选择最上面的“Page: Book a Flight: Mercury”，并点击“OK”按钮确认，将打开“Page Checkpoint Properties”对话框。



当执行测试时，QuickTest 会检查网页的链接与图片的数量，以及加载的时间，如同对话框上方所显示的那样。

QuickTest 页检查每一个链接的 URL 以及每一个图片的原始文件是否存在。

接受默认设定，点击“OK”。QuickTest 会在 Book a Flight: Mercury 网页上加一个网页检查。

4. 在工具栏上点击“Save”保存脚本。

### 3.2.3 文字检查

在这一节中我们学习建立一个文字检查点，检查在“Flight Confirmation”网页中是否出现“New York”？

建立文字检查点：

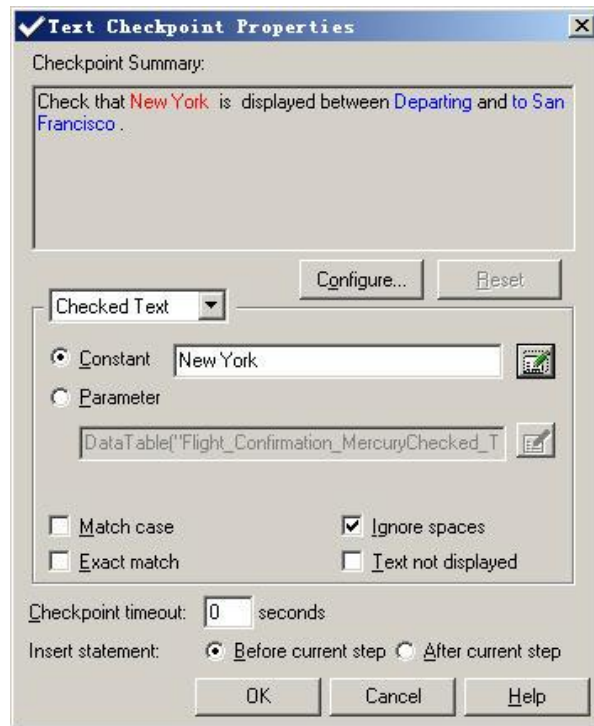
1. 确定要建立检查点的网页

展开“Action1>Welcome: Mercury Tours”选择“Flight Confirmation: Mercury”页面，在“Active Screen”会显示相应的页面。

2. 建立文字检查点

在“Active Screen”中选择在“Departing”下方的“New York”。

对选取的文字按下鼠标右键，并选取“Insert Text Checkpoint”打开“Text Checkpoint Properties”对话框。



当“Checked Text”出现在下拉式清单中时，在“Constant”字段显示的就是选取的文字。这也就是 QuickTest 在执行测试脚本时所检查的文字。

3. 点击“OK”关闭窗口。

QuickTest 会在测试脚本上加上一个文字检查点，这个文字检查点会出现在“Flight Confirmation: Mercury”网页下方。

4. 在工具栏上点击“Save”保存脚本。

### 3.2.4 表格检查

通过添加表检查点，可以检查应用程序中显示的表的内容。通过向测试或组件中添加表检查点，可以检查表的单元格中是否显示了指定的值。对于 ActiveX 表，还可以检查表对象的属性。要添加表检查点，可使用“检查点属性”对话框。

在上面我们已经添加了标准、网页、文字将查点，接下来我们在 Checkpoint 测试脚本中再添加一个表格检查点，检查“Book a Flight: Mercury”网页上航班的价格。

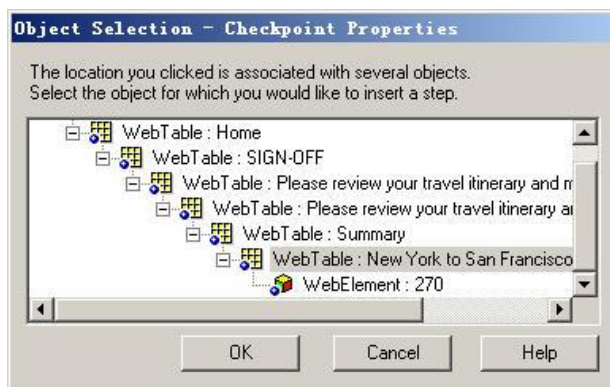
创建表格检查点：

1. 选取要建立检查点的网页

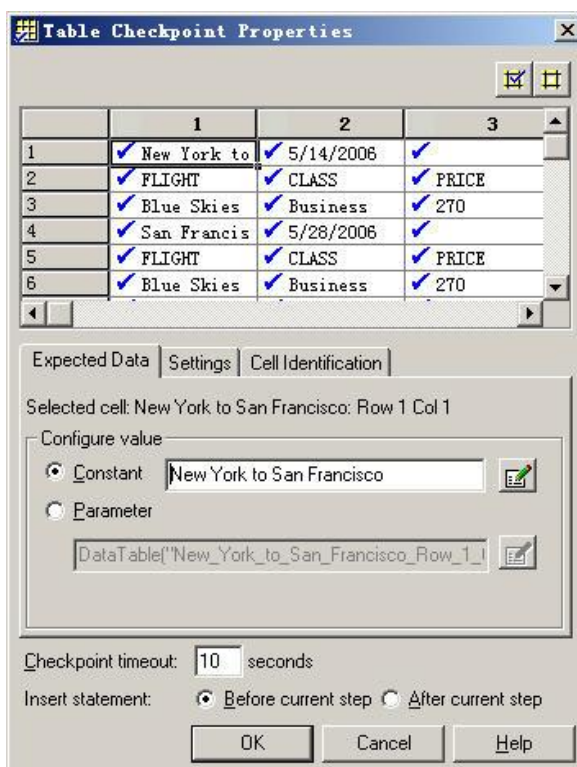
展开“Action1>Welcome: Mercury Tours”选择“Book a Flight: Mercury”页面，在“Active Screen”会显示相应的页面。

2. 建立表格检查点

在“Active Screen”中，在第一个航班的价钱上“270”上点击鼠标右键，选择“Insert Standard Checkpoint”打开“Object Selection-Checkpoint Properties”对话框。



刚打开时选取的是“WebElement:270”，这时要选择上一层的 WebTable 对象，在这个例子中选择“WebTable: New York to San Francisco”。点击“OK”打开“Table Checkpoint Properties”对话框，显示整个表格的内容。



预设每一个字段都会被选择，表示所有字段都会检查，可以对某个字段双击，取消检查字段，或者选择整个栏和列，执行选取或取消的动作。

在每个字段的列标题上双击，取消勾选的图标，然后再 270 字段处双击，这样执行时 QuickTest 只会对这个字段值作检查。

	1	2	3
1	New York to S	5/14/2006	
2	FLIGHT	CLASS	PRICE
3	Blue Skies Ai	Business	✓ 270
4	San Francisco	5/28/2006	
5	FLIGHT	CLASS	PRICE
6	Blue Skies Ai	Business	270

3. 点击“OK”关闭对话框。

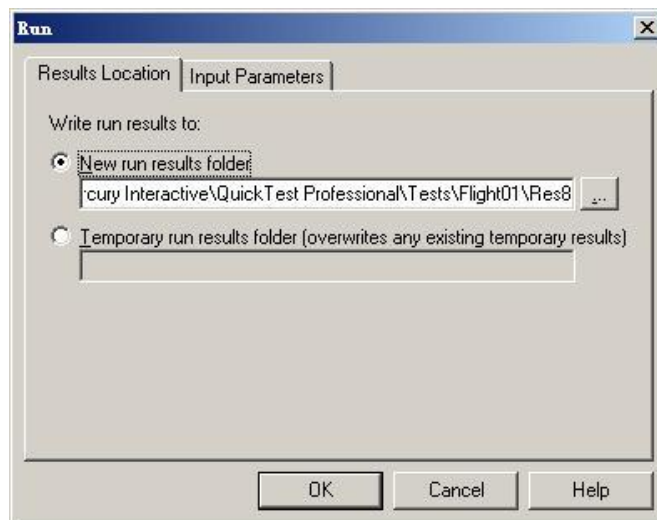
QuickTest 会在测试脚本中，“Book a Flight: Mercury”页面下加上一个表格检查点。

4. 在工具栏上点击“Save”保存脚本。

### 3.3 执行并分析使用检查点的测试脚本

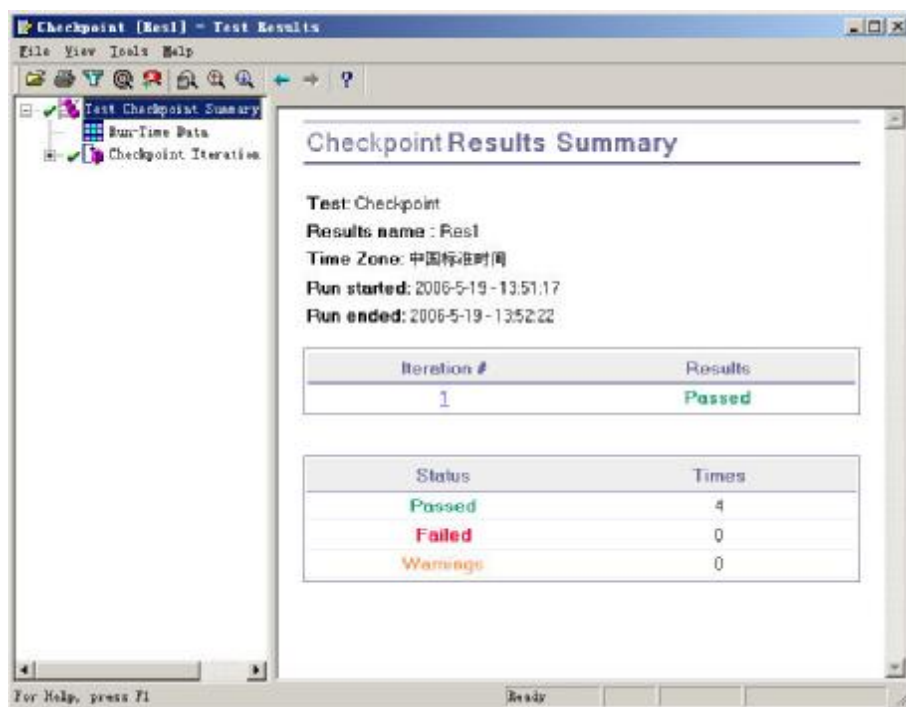
在上一节中，我们在脚本中添加了 4 个检查点，现在，运行 Checkpoint 测试脚本，分析插入检查点后，脚本的运行情况。

1. 在工具栏上点击“Run”按钮，弹出如下窗口：



这个页面是询问将本次测试结果保存在哪个目录，选择“New run results folder”单选按钮，接受默认设置，点击“OK”按钮确认。这时 QuickTest 会按照脚本中的操作，一步一步进行测试，操作过程和手工操作是完全一样。

2. 当 QuickTest 执行完测试脚本后，测试执行结果窗口会自动开启。如果所有的检查点都通过了验证，运行结果为 Passed。如果有一个或多个检查点没有通过验证，这运行结果显示为 Failed，如下图所示：

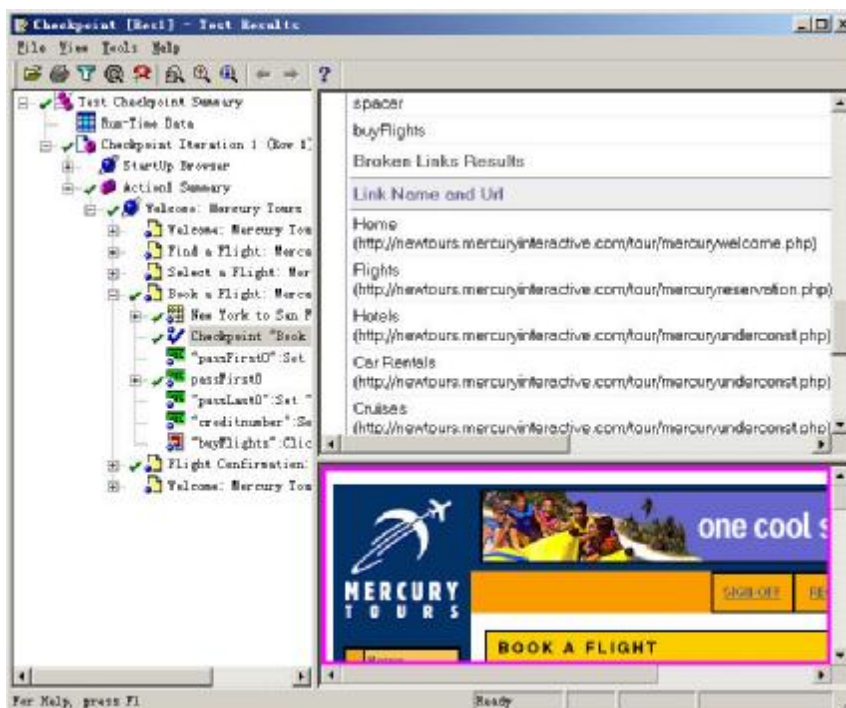


在上图中可以看到，设置的 4 个检查点都通过了验证，下面我们看一下各个检查点的验证结果。

#### n 验证网页检查点

在 test results tree 中展开“Checkpoint Iteration 1 (Row 1) > Action1 Summary > Welcome: Mercury Tours > Book a Flight: Mercury”，并选择“Checkpoint "Book a Flight: Mercury"”。

在右边的“Details”窗口中，可以看到网页检查点的详细信息，例如网页检查点检查了哪些项目。



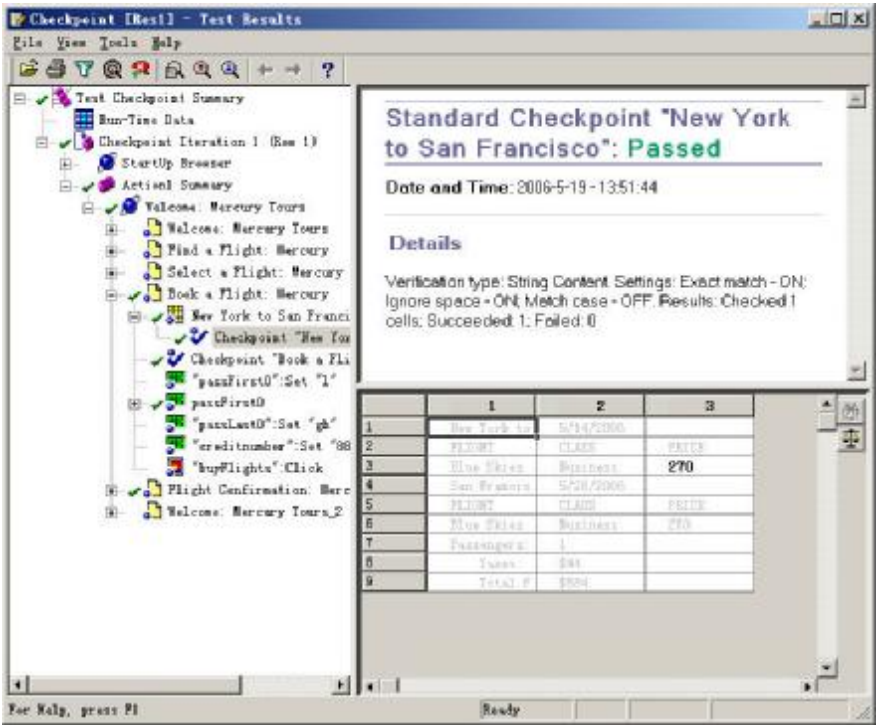


由于所有网页检查的项目，其实际值与预期值相符，所以这个网页检查点的结果为 Passed。

n 验证表格检查点

在 test results tree 中展开 “Book a Flight: Mercury>New York to San Francisco ”，并选择 “Checkpoint " New York to San Francisco " ”。

在 “Details” 窗口可以看到表格的详细结果。也可以在下方看到整个表格的内容，被检查的字段以黑色的粗体文字显示，没有检查的字段以灰色文字显示。如下图所示：

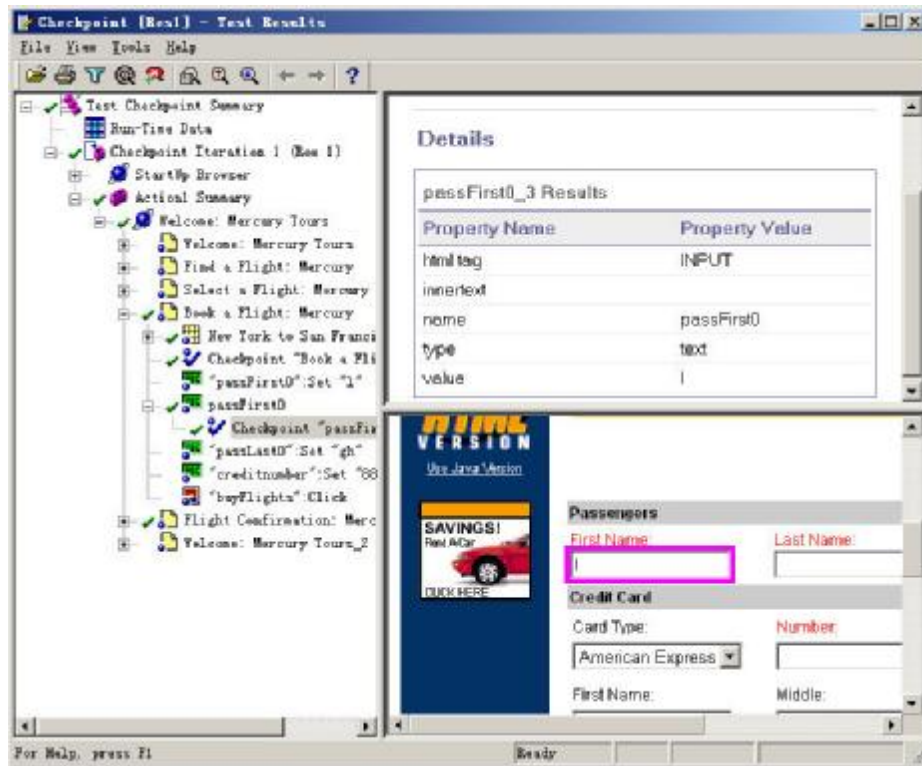


这个表格检查点检查的字段值，其实际值与预期值相符，所以检查点的结果为 Passed。

n 验证标准检查点

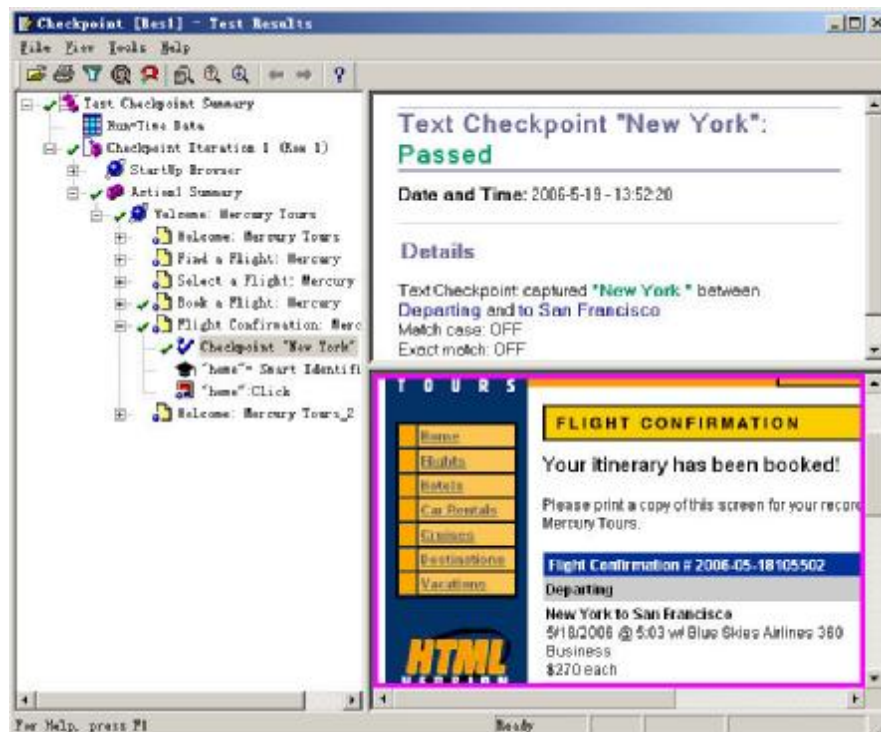
在 test results tree 中展开 “Book a Flight: Mercury >passFirst0”，并选择 “Checkpoint " passFirst0 " ”。

在 “Details” 窗口可以看到标准检查点的详细结果，如检查了哪些属性，以及属性的值。



#### n 验证文字检查点

在 test results tree 中展开“Checkpoint Iteration 1 (Row 1) > Action1 Summary > Welcome: Mercury Tours > Flight Confirmation: Mercury”，并选择“Checkpoint " New York " ”。显示如图界面，因为文字检查点的实际值与预期值相同，所以检查点的结果为 Passed。





---

## 第四章 参数化

在测试应用程序时，可能想检查对应用程序使用不同输入数据进行同一操作时，程序是否能正常的工作。在这种情况下，你可以将这个操作重复录制多次，每次填入不同的数据，这种方法虽然能够解决问题，但实现起来太笨拙了。QuickTest 提供了一个更好的方法来解决这个问题——参数化测试脚本。参数化测试脚本包括数据输入的参数化和检测点的参数化。

使用 QuickTest 可以通过将固定值替换为参数，扩展基本测试或组件的范围。该过程（称为参数化）大大提高了测试或组件的功能和灵活性。

可在 QuickTest 中使用参数功能，通过参数化测试或组件所使用的值来增强测试或组件。参数是一种从外部数据源或生成器赋值的变量。

QuickTest 可以参数化测试或组件中的步骤和检查点中的值。还可以参数化操作参数的值。如果希望参数化测试或组件中多个步骤中的同一个值，可能需要考虑使用数据驱动器，而不是手动添加参数。

### 4.1 参数化步骤和检查点中的值

录制或编辑测试或组件时，可以参数化步骤和检查点中的值。可以参数化选定步骤的对象属性的值。还可以参数化为该步骤定义的操作（方法或函数参数）的值。

例如，应用程序或网站可能包含一个带有编辑字段的表单，用户可以在该编辑字段中键入用户名。你可能希望测试应用程序或网站是否读取该信息并将其正确显示在对话框中。可以插入一个对已登录的用户名使用内置环境变量的文本检查点，以检查显示的信息是否正确。

通过参数化检查点属性的值，可以检查应用程序或网站如何基于不同的数据执行相同的操作。

例如，如果要测试 Mercury Tours 示例网站，可以创建一个检查点，以便检查预订机票后该机票是否被正确预订。假设您需要检查针对各种不同目的地所预订的航班是否正确。可以为目的地信息添加一个数据表参数，而不是为每个目的地分别创建带有单独检查点的不同测试或组件。对于测试或组件的每次循环，QuickTest 都会针对不同目的地检查航班信息。

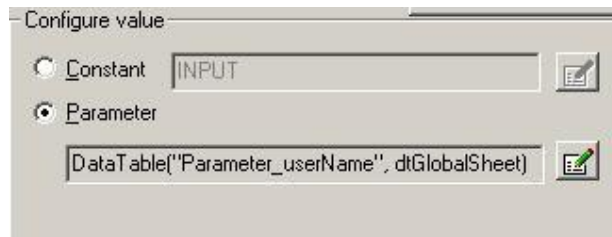
#### 4.1.1 参数化对象和检查点的属性值

可以在“对象属性”或“对象库”对话框中参数化对象的一个或多个属性的值。可以在“检查点属性”对话框中参数化检查点的一个或多个属性的值。

采用下列方式可以打开“对象属性”对话框或“检查点属性”对话框：

- n 选择“步骤” > “对象属性”，或者右键单击某个步骤并选择“对象属性”。将打开“对象属性”对话框。
- n 选择“工具” > “对象库”，单击“对象库”工具栏按钮，或者右键单击包含该对象的操作或组件，然后选择“对象库”。将打开“对象库”对话框。
- n 选择“步骤” > “检查点属性”，或者右键单击该检查点并选择“检查点属性”。

然后在对话框的“配置值”区域中选择参数，




如果该值已经参数化，则“参数”框将显示该值的当前参数定义。如果该值尚未参数化，则“参数”框将显示该值的默认参数定义。


#### 4.1.2 参数化操作的值

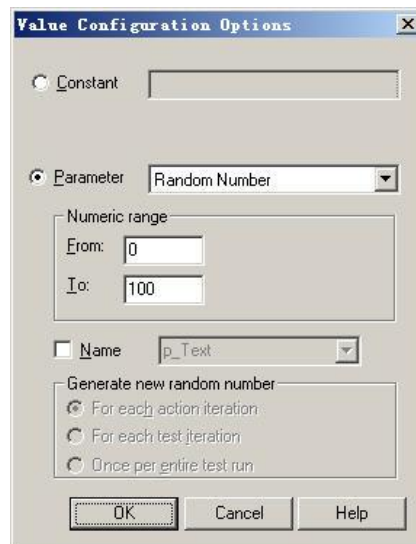
如果步骤中使用的方法或函数具有参数，则可以根据需要参数化该参数值。例如，如果操作使用 **Click**方法，则可以参数化x参数、y参数或这两者的值。

在关键字视图中选择已参数化的值时，将显示该参数类型的图标。例如，在以下片段中，已将**Set** 方法的值定义为随机数字参数。每次运行测试或组件时，**QuickTest** 都会在 **creditnumber**编辑框中输入一个随机数字值。



可以使用视图中的“值”列中的参数化图标来参数化操作值。

单击参数化图标，打开“值配置选项”对话框，将显示当前定义的值。



选择“参数”。如果该值已经参数化，则“参数”部分将显示该值的当前参数定义。如果该值尚未参数化，则“参数”部分将显示该值的默认参数定义。单击“确定”接受显示的参数语句并关闭该对话框。

选择一个尚未参数化的值时，**QuickTest** 会为该值生成默认参数定义。下表描述了如何确定默认参数设置：

执行参数化时	条件	默认参数类型	默认参数名
--------	----	--------	-------

操作中的步骤或检查点的值	至少在当前操作中定义了一个输入操作参数	操作参数	在“操作属性”对话框的“参数”选项卡中显示第一个输入参数
嵌套操作的输入操作参数值	至少为调用该嵌套操作的操作定义了一个输入操作参数	操作参数	在调用操作的“操作属性”对话框的“参数”选项卡中显示第一个输入参数
顶层操作调用的输入操作参数值	至少为测试定义了一个输入参数	测试参数	在“测试设置”对话框的“参数”选项卡中显示第一个输入参数
组件中的步骤或检查点的值	至少为该组件定义了一个输入参数	组件参数	在“业务组件设置”对话框的“参数”选项卡中显示第一个输入参数

如果上述相关条件不为真，则默认参数类型为“数据表”。如果接受了默认参数详细信息， QuickTest 将用基于选定值的名称新建一个数据表参数。

## 4.2 参数种类

QuickTest 有四种类型的参数：

- n 测试、操作或组件参数，通过它可以使用从测试或组件中传递的值，或者来自测试中的其他操作的值。为了在特定操作内使用某个值，必须将该值通过测试的操作层次结构向下传递到所需的操作。然后，可以使用该参数值来参数化测试或组件中的步骤。例如，假设要使用从运行（调用）测试的外部应用程序传递到测试中的某个值来参数化 Action3 中的一个步骤。可将该值从测试级别传递到 Action1 （顶层操作）至 Action3 （Action1 的子操作），然后使用该“操作”输入参数值（从外部应用程序传递的值）来参数化所需的步骤。
- n 数据表参数，通过它可以创建使用您所提供的数据多次运行的数据驱动的测试（或操作）。在每次重复（或循环）中， QuickTest 均使用数据表中不同的值。例如，假设您的应用程序或网站包含一项功能，用户可以通过该功能从成员数据库中搜索联系信息。当用户输入某个成员的姓名时，将显示该成员的联系信息，以及一个标记为“查看 <MemName> 的照片”的按钮，其中<MemName>是该成员的姓名。可以参数化按钮的名称属性，以便在运行会话的每次循环期间， QuickTest 可标识不同的照片按钮。
- n 环境变量参数，通过它可以在运行会话期间使用来自其他来源的变量值。这些变量值可能是您所提供的值，或者是 QuickTest 基于您选择的条件和选项而生成的值。例如，可以让 QuickTest 从某个外部文件读取用于填写 Web 表单的所有值，或者可以使用 QuickTest 的内置环境变量之一来插入有关运行测试或组件的计算机的当前信息。

- n 随机数字参数，通过它可以插入随机数字作为测试或组件的值。例如，要检查应用程序处理大小机票订单的方式，可以让 QuickTest 生成一个随机数字，然后将其插入到“票数”编辑字段中。

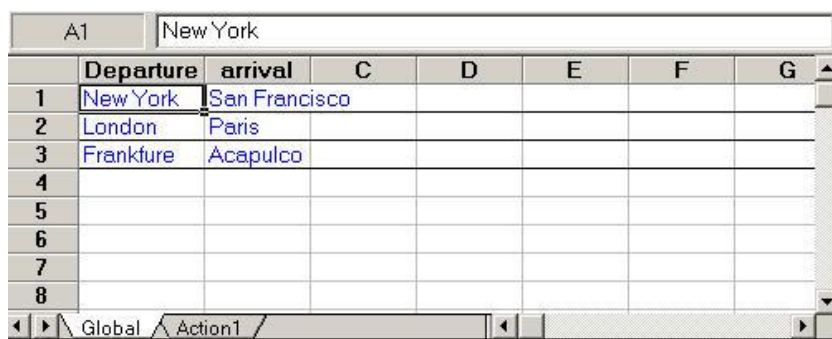
#### 4.2.1 使用数据表参数

可以通过创建数据表参数来为参数提供可能的值列表。通过数据表参数可以创建使用所提供的数据多次运行的数据驱动测试、组件或操作。在每次重复中，QuickTest 均使用数据表中不同的值。

例如，考虑 Mercury Tours 示例网站，通过该网站可预订航班请求。要预订航班，需要提供航班路线，然后单击“继续”按钮。该网站将针对请求的路线返回可用的航班。

可通过访问网站并录制大量查询的提交来执行该测试。这是一个既费时又费力的低效解决方案。通过使用数据表参数，可以连续对多个查询运行测试或组件。

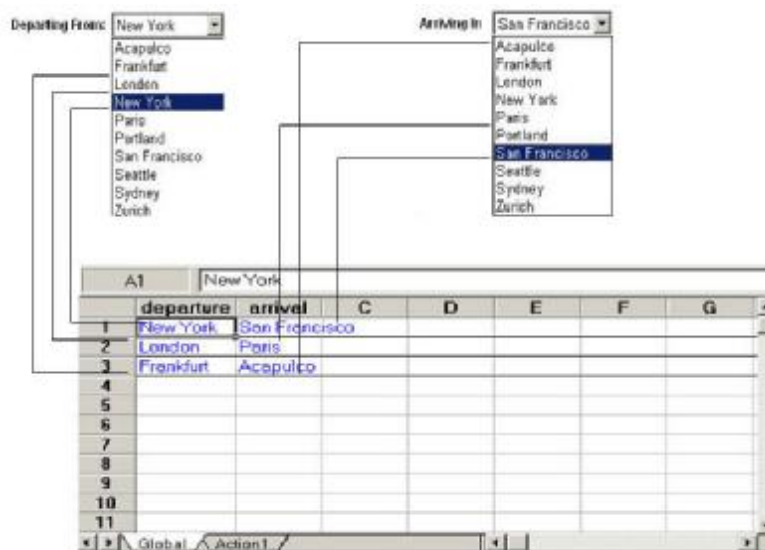
参数化测试或组件时，需要首先录制访问网站并针对所请求的一条路线来检查可用航班的步骤。然后将录制的路线替换为某个数据表参数，并在数据表的全局表中添加自己的数据集，每条路线一个。



	A1	Departure	arrival	C	D	E	F	G
1		New York	San Francisco					
2		London	Paris					
3		Frankfure	Acapulco					
4								
5								
6								
7								
8								

新建数据表参数时，将在数据表中添加新的一列，并将参数化的当前值放在第一行中。如果要对值进行参数化并选择现有的数据表参数，则将保留所选参数的列中的值，并且这些值不会被参数的当前值覆盖。

表中的每个列都表示单个数据表参数的值列表。列标题是参数名。表中的每一行都表示 QuickTest 在测试或组件的单个循环期间为所有参数提交的一组值。运行测试或组件时，QuickTest 将针对表中的每一行数据运行一次测试或组件循环。例如，如果测试在数据表的全局表中有十行，则运行十次循环。



在上面的例子中，当运行测试时，QuickTest降为每一个路线分别提交一个查询。

#### 4.2.2 使用环境变量参数

QuickTest 可以插入环境变量列表中的值，该列表是可通过测试访问的变量和相应值的列表。在测试运行的整个过程中，无论循环次数是多少，环境变量的值始终保持不变，除非在脚本中以编程方式更改变量的值。

QuickTest有以下三种环境变量：用户定义的内部环境变量、用户定义的外部环境变量以及内置环境变量。

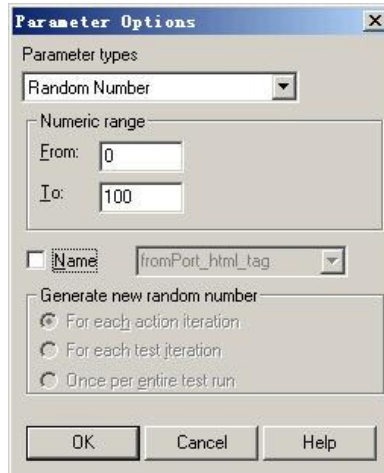
用户定义的内部环境变量--在测试内定义的变量。这些变量与测试一起保存，并且只能在定义这些变量的测试内访问。在“测试设置”对话框或“参数选项”对话框的“环境”选项卡中，可以创建或修改测试中用户定义的内部环境变量。

用户定义的外部环境变量--在活动外部环境变量文件中预定义的变量。可根据需要创建任意多的文件，并为每个测试选择一个适当的文件，或者更改用于每个测试运行的文件。

内置环境变量--表示有关测试和运行测试的计算机的信息的变量，例如测试路径和操作系统。从所有测试和组件中都可以访问这些变量，并且它们都被指定为只读变量

#### 4.2.3 使用随机数字参数

当选择“随机数字”作为参数类型时，可以通过“参数选项”对话框将参数配置为使用随机数字。“值配置选项”对话框的“参数”部分与“参数选项”对话框非常相似。



数字范围--指定用于生成随机数字的范围。默认情况下，随机数字范围介于 0 和100 之间。可通过在“从”和“到”框中输入不同的值来修改此范围。该范围必须介于 0 和 2147483647（包含）之间。

名称--指定参数的名称。通过为随机参数指定名称可以在测试中多次使用同一个参数。可以选择现有的命名参数，或者通过输入新的描述性名称来新建命名参数。

生成新随机数字--定义命名随机参数的生成计时。选中“名称”复选框时会启用该框。可以选择下列选项之一：


- n 为每次操作循环：在每次操作循环结束时生成一个新数字。
- n 为每次测试循环：在每次全局循环结束时生成一个新数字。
- n 为整个测试运行生成一次：第一次使用参数时生成一个新数字。在整个测试运行中，对参数使用同一个数字。

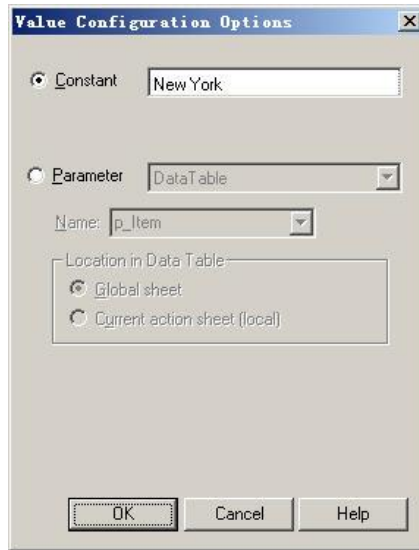
## 4.3 参数化测试脚本

### 4.3.1 定义参数

在上面的课程中我们学习了参数的种类以及参数化步骤和检查点中的值，现在我们使用 Checkpoint 脚本，在测试脚本中，纽约是个常数值，也就是说，每次执行测试脚本预定机票时，出发地点都纽约，现在，我们将测试脚本中的出发地点参数化，这样，执行测试脚本时就会以不同的出发地点去预定机票了。

1. 首先，我们打开 Checkpoint 测试脚本，将脚本另存为“Parameter”，然后选择要参数化的文字：在视图树中展开“Action1>Welcome: Mercury Tours>Find a Flight: Mercury”。

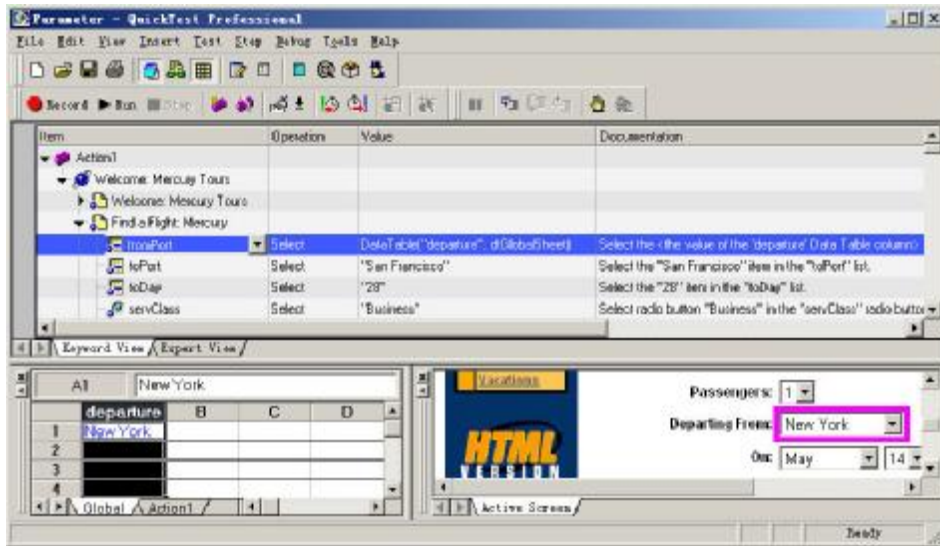
2. 在视图树中选择“fromPort”右边的“Value”字段，然后再点击参数化图标，开启“Value Configuration Options”对话框：




3. 设置要参数化的属性，选择“Parameter”选择项，这样就可以用参数值来取代“New York”这个常数了，在参数中选择“Data Table”选项，这样这个参数就可以从 QuickTest 的 Data Table 中取得，将参数的名字改为“departure”。



4. 点击“OK”确认，QuickTest 会在 Data Table 中新增 departure 参数字段，并且插入了一行 New York 的值，New York 会成为测试脚本执行使用的第一个值。



参数化以后可以看到树视图中的变化，在参数之前，这个测试步骤显示“fomPost ...Select... New York”，现在，这个步骤变成了“fomPost ...Select... Data Table ( " departure " , dtGlobalSheet)”。而且当点击 Value 字段时，Value 字段会显示如图所示：

，表示此测试步骤已经被参数化，而且其值从 Data Table 中的 departure 字段中获得。

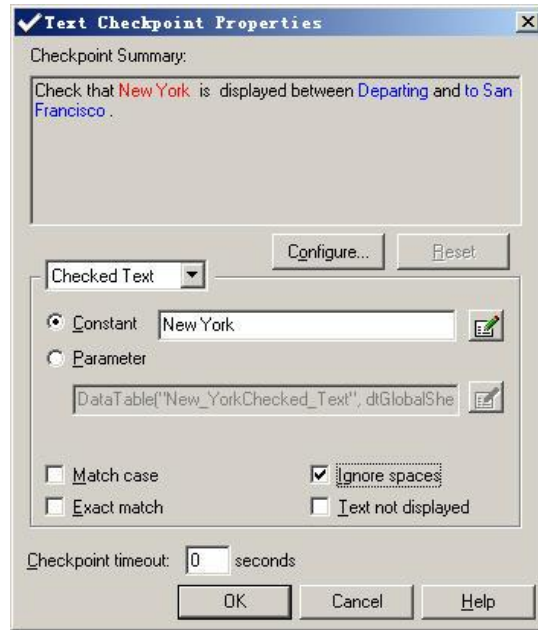
5. 在 departure 字段中加入出发点资料，使 QuickTest 可以使用这些资料执行脚本。  
在 departure 字段的第二行，第三行分别输入：Portland、Seattle。
6. 保存测试脚本。


#### 4.3.2 修正受到参数化影响的步骤

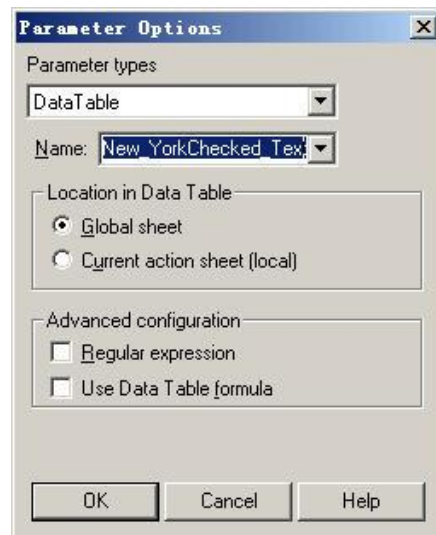
当测试步骤被参数化以后，有可能会影响到其它的测试步骤也要参数化，例如我们为了验证在“Flight Confirmation”网页中是否出现“New York”（第三章创建文字检查点），在网页上添加了一个文字检查点。那么，就要对出发地的文字检查点作参数化，以符合对出发地点参数化的预期结果。

修正文字检查点，首先在树视图中，展开“Action1>Welcome: Mercury Tours>Flight Confirmation: Mercury”页面，然后点击鼠标右键，选择“Checkpoint Properties”，打开“Text Checkpoint Properties”对话框：





在“Checked Text”的 Constant 字段中显示为“New York”，表示测试脚本在每次执行时，这个文字检查点的预期值都为“New York”。我们选择 Parameter，点击旁边的“Parameter Options”按钮，打开“Parameter Options”对话框：



在参数类型选择框选择“Data Table”选项，在名字选择框选择“departure”选项，指明这个文字检查点使用 departure 字段中的值当成检查点的预期值。

点击“OK”关闭窗口，这样文字检查点也被参数化了。

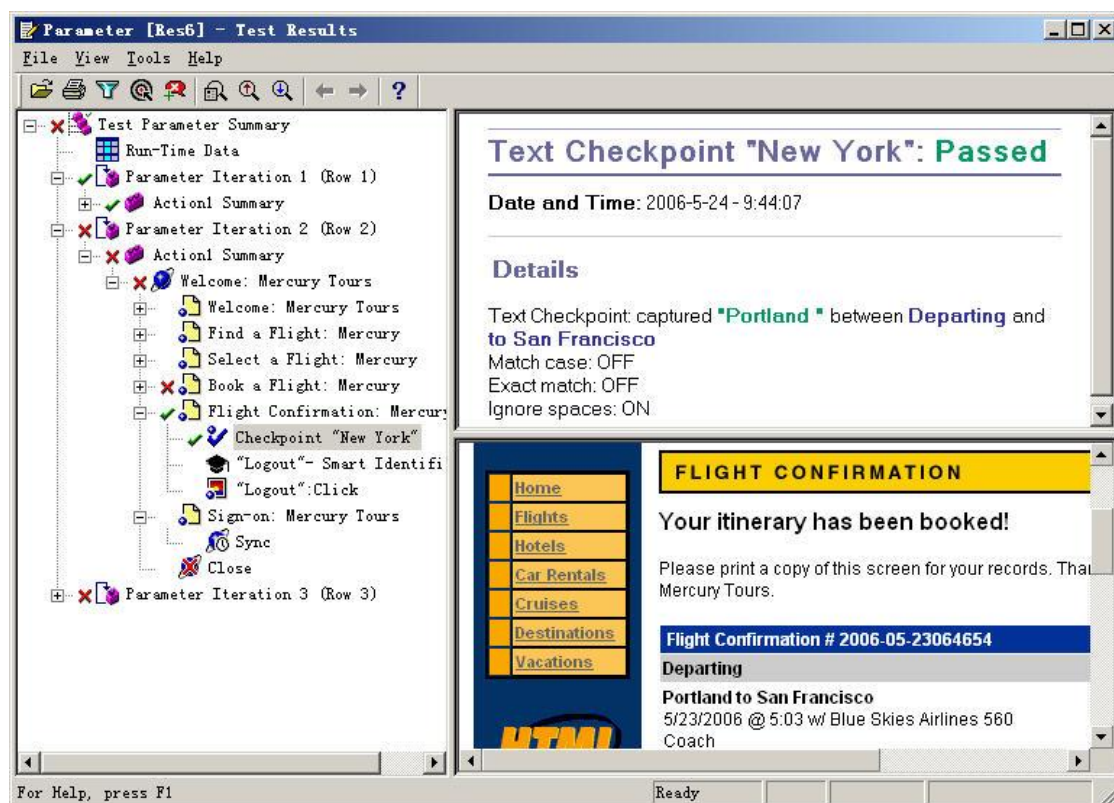
### 4.3.3 执行并分析使用参数的测试脚本

参数化测试脚本后，我们运行 Parameter 测试脚本。QuickTest 会使用 Data Table 中

departure 字段值，执行三次测试脚本。

执行测试脚本：点击工具栏上的“Run”按钮，开启 Run 对话框，选取“New run results folder”，其余为默认值，点击“OK”开始执行脚本。当脚本运行结束后，会开启测试结果窗口。

在树视图中，展开“Parameter Iteration2 >Action1 Summary >Welcome Mercury Tours>Flight Confirmation: Mercury”，选择“Checkpoint "New York "”，显示如下图：



在检查点“Details”窗口中，显示 Portland 为预期记过同时也是实际的值，所以文字检查点为通过。同时也可以看到在下方的“Application”窗口中，显示机票的出发地点也是 Portland。

在图中可以看出，虽然每次执行时，文字检查点的结果是通过的，但是第二次与第三次的执行结果仍然为失败。这是因为出发地点的改变，造成在表格检查点中的机票价钱改变，导致表格检查点失败。在以后的课程中，我们将学习修正表格检查点，让 QuickTest 自动更新表格检查点的预期结果，就可以检查正确的票价了。

## 第五章 输出值

通过 QuickTest 可以检索测试或组件中的值，并将这些值作为输出值存储。此后，就可以检索这些值，并在运行会话的不同阶段使用该值作为输入。

输出值是一个步骤，在该步骤中，捕获测试或组件中某个特定点的一个或多个值，并在

---

运行会话持续时间存储这些值。随后，在运行会话中的不同点，可以将这些值作为输入使用。

可以输出任何对象的属性值。还可以从文本字符串、表单元格、数据库和 XML 文档输出值。

创建输出值步骤时，可以确定运行会话持续时间内的值存储在哪里，以及如何使用这些值。运行会话期间，QuickTest 检索指定点的每个值并将其存储在指定位置。以后当运行会话中需要值时，QuickTest 将从该位置检索值并根据需要来使用。

## 5.1 创建输出值

### 5.1.1 输出值类型

将输出值步骤添加到测试或组件时，首先选择要输出的值的类别，例如，属性值、文本值或 XML 元素值。然后，就可以确定要输出的值以及每个值的存储位置。在 QuickTest 中可以创建以下几个类别的输出值：

- n 标准输出值
- n 文本和文本区输出值
- n 数据库输出值
- n XML 输出值

#### 1. 标准输出值

可以使用标准输出值来输出大多数对象的属性值。例如，在基于 Web 的应用程序中，一个网页中的链接数可能基于用户在上一页的表单中所做选择的不同而变化。可以在测试中创建一个输出值，来存储页面中的链接数。还可以使用标准输出值来输出表单元格的内容。

#### 2. 文本和文本区输出值

可以使用文本输出值来输出屏幕或网页中显示的文本字符串。创建文本输出值时，可以输出对象文本的一部分。还可以指定要在输出文本之前和之后输出的文本。

可以使用文本区域输出值来输出 Windows Applications 中屏幕已定义区域内显示的文本字符串。例如，假设在测试的应用程序中，想要存储显示在特定步骤之后的任何错误消息的文本。在 If 语句中，查看带有已知标题栏值（例如 Error）的窗口是否存在。如果该窗口存在，则输出该窗口中的文本（假设窗口大小与所有可能的错误消息的大小相同）。

在使用基于 Windows 的应用程序文本输出值时应注意以下事项：

- n 在基于 Windows 的应用程序中创建文本或文本区输出值时使用文本识别机制，有时会检索到不想要的文本信息（例如隐藏文本和带阴影的文本，这些文本会作为同一字符串的多个副本显示）。
- n 此外，在不同的运行会话中，文本（和文本区）输出值的表现方式可能不同，具体取决于使用的操作系统版本、已经安装的 Service Pack、安装的其他工具包、的应用程序中使用的 API 等等。

#### 3. 数据库输出值

可以使用数据库输出值，基于在数据库上定义的查询的结果（结果集）来输出数据库单元格的值。可以从结果集的全部内容中创建输出值，也可以从其中某一部分创建输出值。在运行会话过程中，QuickTest 从数据库中检索当前数据，并根据指定的设置来输出值。

#### 4. XML 输出值

可以使用 XML 输出值输出 XML 文档中的 XML 元素和属性的值。运行会话完成后，可以在“测试结果”窗口中查看 XML 输出值的概要结果。还可以通过打开“XML 输出值结果”窗口来查看详细结果。例如，假设网页中的某个 XML 文档包含新车的价目表。可以通过选择要输出的相应的 XML 元素值来输出特定汽车的价格。

以下给出每种环境支持的输出值类型：

输出值类别	Web	标准 Windows	VB	ActiveX	其它环境
标准	S	S	S	S	NA
页（标准）	S	NA	NA	NA	NA
表（标准）	S	NA	NA	S	NA
文本	S	S	S	S	NA
文本区	NS	S	S	S	NA
数据库	NS	NA	NA	NA	S（DbTable）
XML	S	NA	NA	NA	XML文件

\*S--支持 NS--不支持 NA--不适用

#### 5.1.2 存储输出值

定义输出值时，可以指定运行会话期间在哪里以及如何存储每个值。可以将值输出到：

- n 测试、操作或组件参数

- n 运行时数据表

- n 环境变量

##### 1. 将值存储在测试、操作或组件参数中

可以将值输出到操作或组件参数，以便可以在运行会话后面的部分中使用来自运行会话某一部分的值，或者传递回运行（调用）测试或组件的应用程序。

例如，假设要测试一个购物应用程序，该程序计算采购费用，并自动从账户中扣除采购金额。想要测试在每次运行带有不同的采购单的操作或组件时，该应用程序是否能够正确地从账户中扣除采购金额，可以将花费的总金额输出到某个操作或组件的参数值，然后在稍后的扣除该金额操作中的运行会话部分使用该值。

##### 2. 将值存储在运行时数据表中

对于要运行多次的由数据驱动的测试（或操作）来说，将值输出到运行时数据表的选项特别有用。在每次重复或循环中，QuickTest 检索当前值并将其存储在运行时数据表的相应的行中。

例如，要测试一个航班预定应用程序，因此设计了一个测试来创建新预定，随后查看预定详细信息。每次运行测试时，应用程序为新预定生成一个唯一的订单号。要查看预定，应用程序要求用户输入相同的订单号。运行该测试之前，还不知道订单号。

要想解决这个问题，就要将在创建新预定时生成的唯一订单号的值输入数据表中。然后，在“View Reservation”屏幕中，使用包含存储值的列将输出值插入订单号输入字段中。运行测试时，QuickTest 检索站点为新预定生成的唯一订单号，并在运行时数据表中输入此输出值。测试到达查看预定所需的订单号输入字段时，QuickTest 将存储在运行时数据表中的唯一订单号插入订单号字段中。

##### 3. 将值存储在环境变量中

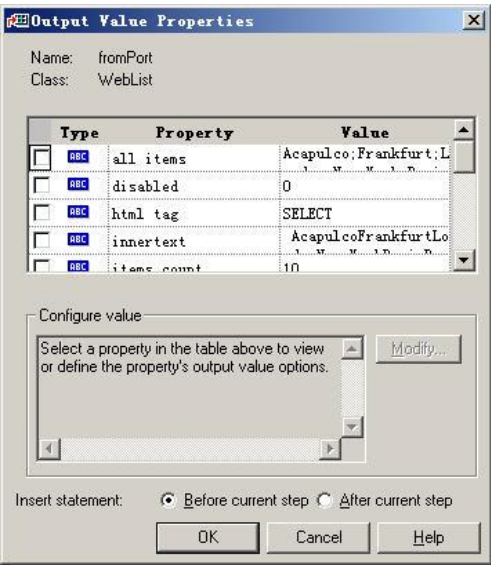
将值输出到内部用户定义的环境变量时，可以在运行会话后面的阶段使用该环境变量输入参数。

例如，假设在测试一个应用程序，该程序会提示用户在“欢迎使用”页输入账号，然后显示用户姓名。就可以使用文本输出值来捕获显示的名称值，并将其存储在环境变量中。然后，可以检索环境变量中的值以便在应用程序的其他位置中输入用户的姓名。

## 5.2 输出属性值

### 5.2.1 定义标准输出值

通过“输出值属性”对话框可以选择要输出的属性值，并定义您选择的每个值的设置。



关闭此对话框之前，可以为相同对象选择许多属性并为每个属性值定义输出设置。运行会话过程中到达输出值步骤时，QuickTest 将检索所有指定的属性值。



#### 1. 标识对象



对话框的上部显示有关要创建输出值的测试对象的信息：

项目	描述
名称	测试对象的名称
类	对象的类别

#### 2. 选择要输出的属性值

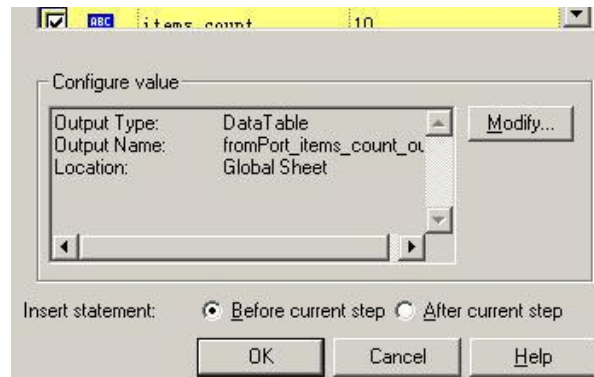
对话框的上半部分包含一个窗格，其中列出选定对象的属性，以及它们的值和类型。该窗格包含以下项：

窗口元素	描述
复选框	要指定将输出的属性，选择相应的复选框，可以为对象选择多个属性，并为选择的每个属性值指定输出选项。
类别	 图标表示属性的值当前为常量  图标表示属性的值当前存储在测试、操作或组件参数中

	 图标表示属性的值当前存储在运行时数据表中  图标表示属性的值当前存储在环境变量中
属性	属性的名称
值	属性的当前值

### 3. 指定属性值的输出设置

选择属性的复选框时，将突出显示属性详细信息，并且在“配置值”区域中显示选定属性值的当前输出定义。



第一次选择要输出的属性值时，“配置值”区域中会显示值的默认输出定义。选择要输出的属性值时，可以：

- n 通过选择其他属性值或单击“OK”接受显示的输出定义。
- n 通过单击“修改”按钮更改选定值的输出类型和/或设置。将打开“输出选项”对话框并显示该值当前的输出类型和设置。

## 5.2.2 指定输出类型和和设置

为每个值定义的输出类型和设置决定该值在运行会话中的存储位置以及使用方式。到达输出值步骤时，QuickTest 检索为输出选定的每个值并将其存储在指定位置，以供以后在运行会话中使用。新建输出值步骤时，QuickTest 为选定要输出的每个值指定一个默认定义。

可以通过选择不同的输出类型并/或更改输出设置来更改选定值的当前输出定义：

- n 将值输出到操作或组件参数
- n 将值输出到数据表
- n 将值输出到环境变量

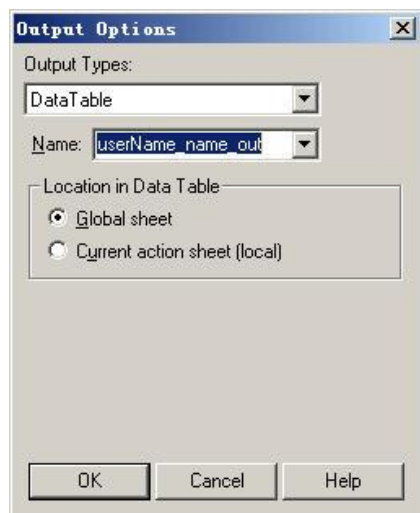
#### 1. 将值输出到操作或组件参数

可以将值输出到操作或组件参数，以便这些值可以在运行会话的后面部分中使用，或者传递回运行（调用）测试或组件的外部应用程序。如果参数已经定义为用于调用操作或组件的输出参数，只能将值输出到操作或组件参数。此外，仅当输出值类型和参数值类型匹配时，将值输出到操作或组件的选项才可用。选择“测试参数”、“操作参数”或“组件参数”作为输出类型时，通过“输出选项”对话框可以选择在其中存储运行会话持续时间的选定值的参数。

#### 2. 将值输出到数据表

选择“数据表”作为输出类型时，通过“输出选项”对话框可以指定在运行时数据表中

存储选定值的位置。



在将值输出到数据表时，有以下选项可以修改：

- n** 名称--指定数据表中要存储值的列的名称。QuickTest 建议使用输出的默认名称。可以从列表中选择现有的输出名称，也可以通过使用默认输出名称或输入有效的描述性名称来新建输出名称。
- n** 数据表中的位置--输出测试的值时，指定将数据表列名称添加到数据表的全局工作表还是当前操作工作表中。

### 3. 将值输出到环境变量

如果选择“环境”作为输出类型时，通过“输出选项”对话框，可以指定要在其中存储运行会话持续时间的选定值的环境参数，该参数由内部用户定义。





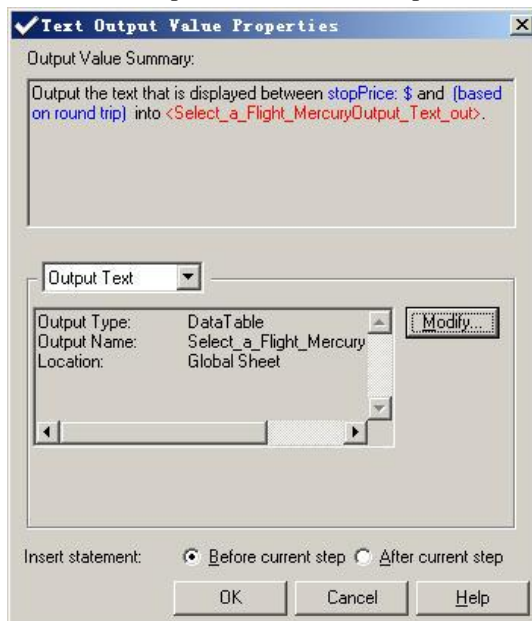
## 5.3 在脚本中建立输出值

### 5.3.1 建立输出值

在上一章中，因为在表格检查点中机票价钱的预期结果，并没有随着出发地点的改变而变动，导致第二、第三次的执行结果是失败的。

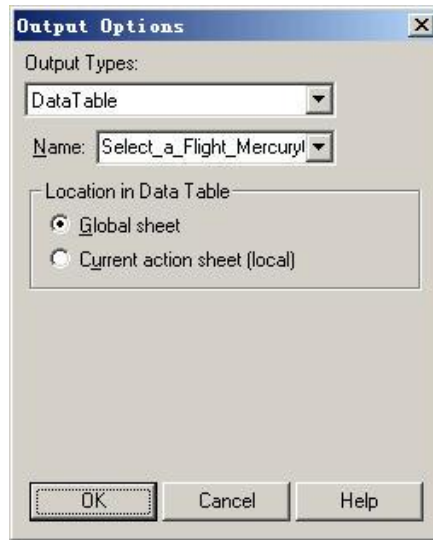
现在，我们从“Select a Flight: Mercury”网页上取得机票价钱，并且已取得的机票价钱更新表格检查点的预期结果，这样一来，测试脚本就可以利用在“Select a Flight: Mercury”网页上取得的机票价钱去验证“Book a Flight: Mercury”上显示的机票价钱。

1. 首先，我们打开 Parameter 测试脚本，将脚本另存为“Output”测试脚本。
2. 在树视图中，展开“Welcome: Mercury Tours”并且点击“Select a Flight: Mercury”网页，在 Active Screen 窗口会显示相应的页面。在 Active Screen 窗口中选取框住 270，然后点击鼠标右键，选择“Insert Text Output”，打开“Text Output Value Properties”对话框：



3. 在“Text Output Value Properties”对话框中点击“Modify”按钮，打开“Output Options”对话框：





在名字字段显示 `Select_a_Flight_MercuryOutput_Text_out`, 将其改成 `depart_flight_price` , 接受其它默认值, 点击“OK”确认, QuickTest 会在 Data Table 中加入 `depart_flight_price` 字段。

在 Data Table 上的 `depart_flight_price` 字段的第一行会显示从应用程序上取得的输出值 (270)。在执行时, 第一次 QuickTest 会取得一样的值 270, 接下来的第二、第三次会从应用程序上取得实际值, 并存放在 Data Table 中。

#### 4. 修正表格检查点的预期值

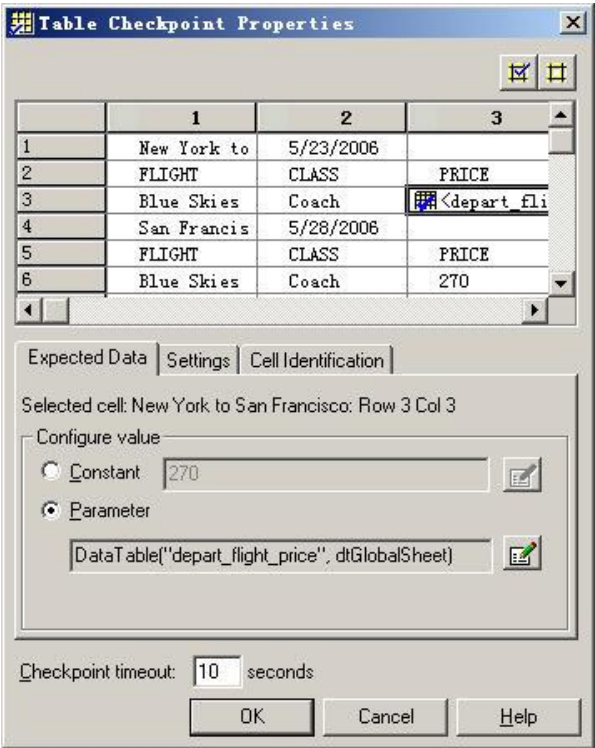
在树视图中, 展开“Welcome: Mercury Tours >Book a Flight: Mercury”, 在“Checkpoint "New York to San Francisco " ”上点击鼠标右键, 选择“Checkpoint Properties”, 打开“Table Checkpoint Properties”对话框。

选中第三行, 第三列 (被勾选的字段), 在“Configure value”中选择“Parameter”然后点击“Parameter Options”按钮, 打开“Parameter Options”对话框:



在窗口的名字下拉列表中选择 `depart_flight_price`。

5. 点击“OK”回到“Table Checkpoint Properties”对话框，可以看到这个检查点的预期结果已经被参数化了。



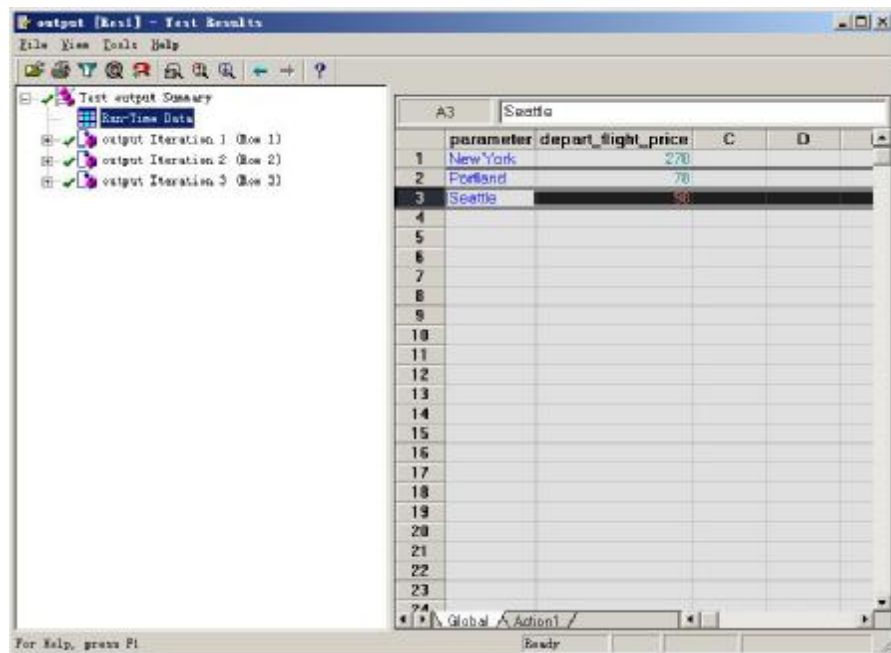
点击“OK”关闭“Table Checkpoint Properties”窗口，保存测试脚本

### 5.3.2 执行并分析使用输出值的测试脚本

在上面我们在建本中建立侧输出值，并且将表格检查点参数化，现在，执行 Output 测试脚本。

执行测试脚本：点击工具栏上的“Run”按钮，开启 Run 对话框，选取“New run results folder”，其余为默认值，点击“OK”开始执行脚本。当脚本运行结束后，会开启测试结果窗口。

在执行结果窗口中，点击树视图中的“Run-Time-Data”，可以在表格中看到执行测试时使用的输出值，在 depart\_flight\_price 字段中显示了不同的机票价钱。



在结果窗口中点击“Test output Summary”可以看到，12 个检测点都通过了验证，运行结果均为 Passed。