TestPlant Documentation

English 简体中文 日本語

查找图像

查找可能有几种状态的图像

编写脚本时，不能总指望SUT上的图像在捕获后，就始终保持不变。例如，想要点击窗口上的关闭按钮，窗口可能处于活动状态，也可能未处于活动状态。在Windows系统上，关闭按钮可能是红色或灰色的，取决于窗口是否处于活动状态。

面对上述类似情景，可使用图像集合。图像集合使eggPlant能够在一次图像搜索时，搜索多张图像。然后运行图像集合中任意图像对应的一行代码。这使eggPlant能够在多个浏览器、平台和/或设备之间，运行同一个脚本。

有不同的方法，可创建图像集合。请参考创建图像集合。

规避定时问题

有时候，一个已经多次成功运行的脚本突然报告，不能在查看器窗口查找到图像。通常，原因是定时问题，而不是图像识别问题。

快速测试是否是定时问题的方法：选中失败的代码行，点击脚本编辑器中的"运行所选脚本"按钮。如果运行成功，则很可能是eggPlant Functional在图像显示在屏幕上之前，就在搜索图像。这就是定时问题。（如果仍未搜索到图像，在查看器窗口检查图像是否有改变或者改变图像搜索类型。）

以下方法，可帮助你防止定时问题：

WaitFor命令

WaitFor命令可阻止脚本的下一行代码运行，直到搜索到指定图像后才运行。在前一步是打开新应用程序或网页的情况下，这个命令特别有用。（打开已开启的应用程序的其它部分时，通常不需要上述命令。）

示例：使用WaitFor命令

```
Click "SaveAs"

WaitFor 5, "SaveDialog"                                    //为保存对话框等待5秒钟。

TypeText "FileName"
```

本例中，脚本在保存对话框输入内容前，最多等待5秒钟等弹出保存对话框。（如果保存对话框在5秒内未弹出，脚本失效。）

可大胆设置WaitFor的值，不需要猜测究竟需要等多久。由于脚本一搜索到图像就继续执行，不会在成功搜索到图像时浪费时间。一个较好的经验方法是，将MaxWait的值设置为用户合理等待的时间。如果时间不够，应用程序可能出现问题。

WaitFor 对比 Wait

Wait命令按照时间参数定义的值暂停脚本。（默认设置值的单位是秒，也可输入*minutes*（分钟）或*milliseconds*（毫秒）变更设置。）

Wait命令运行过程中，eggPlant Functional不执行任何操作。如果只想在某时刻暂停脚本运行，该命令很有用，但是如果是在SUT上运行某任务前等待一段时间，则使用WaitFor命令更可靠（可能也更快）。

使用WaitFor命令时，指定的时间是 *最大*时间，而不是绝对时间。如果在最大时间内弹出图像，脚本立即继续执行。

示例：比较Wait和WaitFor命令

```
Wait 8                                     //将脚本运行暂停8秒钟。

WaitFor 8.0,"OK Button"                    //暂停脚本运行，搜索到图像后才继续运行，最多等待8秒钟。
```

远程工作间隔时间

如果SUT经常随机出现定时问题，需要稍微降低eggPlant Functional的速度。提高远程工作间隔时间，可降低eggPlant Functional的速度。远程工作间隔时间是eggPlant Functional将命令发送给SUT的间隔时间。两种方法提高远程工作间隔时间：在运行选项设置中，修改远程工作间隔时间设置，或者设置RemoteWorkInterval全局属性。

远程工作间隔时间偏好设置

如果大多数SUT频繁出现定时问题，则在运行选项设置中修改远程工作间隔时间的默认设置。

1. 选择偏好设置>运行选项>系统。
2. 只需稍微提高远程工作间隔时间的值，1/10或2/10秒可能就能满足要求。

RemoteWorkInterval全局属性

如果只需要在特定（或部分）脚本内降低SUT速度，可根据情况设置RemoteWorkInterval全局属性。

示例：修改RemoteWorkInterval设置

```
add .1 to the remoteWorkInterval
```

```
subtract .1 from the remoteWorkInterval

setOption remoteWorkInterval, .1
```

如果编写处理程序按需求更改全局属性，然后将恢复为初始值，请参考优化脚本性能中FastImageFound这个例子。

图像搜索时间

图像搜索时间是指eggPlant Functional搜索图像所用的最少时间。可在运行选项设置中，将其设置为“永久”值，或者利用ImageSearchTime全局属性，根据情况在脚本内设置。

注：图像搜索时间详细信息

更改图像搜索时间设置时，同时修改搜索次数和搜索延迟的值，反之亦然。图像搜索时间总是小于搜索次数与搜索延迟值之积。

图像搜索时间偏好设置

如果图像搜索时间的值一*直*太低，可在运行选项设置中，更改默认值。（提高图像搜索时间的不良影响是条件块用时更长。）

1. 选择偏好设置>运行选项>屏幕。
2. 以较小的增量，提高图像搜索时间（不超过0.5秒）。

ImageSearchTime全局属性

如果只需临时提高图像搜索时间（通常这是最佳选择），可按需要设置 ImageSearchTime全局属性。

示例：临时修改ImageSearchTime()

```
put getOption (ImageSearchTime) into IST                    //保存ImageSearchTime初始值。

set the ImageSearchTime to 2                                //修改ImageSearchTime值。

(*在此继续运行脚本，然后...*)

setOption ImageSearchTime, IST                              //恢复ImageSearchTime初始值。
```

使用图像函数简化流程

本文说明使用图像函数，节省搜索时间和收集更多脚本运行信息的方法。

FoundImageLocation()

在同一行两次调用同一图像时，FoundImageLocation()函数可节省大量时间。FoundImageLocation()返回最近一次查找到的图像的屏幕坐标，下次调用该图像时，可使用屏幕坐标调用，节省搜索时间。

示例：使用FoundImageLocation函数

```
WaitFor 8, "Save Button", "OK Button","Retry Button"        //查找到按钮图像前最多等待8秒钟。

Click FoundImageLocation()                                  //点击查找到的图像的热点。
```

在上述例子中，eggPlant Functional在WaitFor命令运行过程中查找到一个按钮图像。接下来，既然已经知道按钮的位置，就立即点击。如果使用图像名称而不是FoundImageLocation()函数，eggPlant Functional必须重新搜索后才能运行Click命令。

ImageFound()

有时需要知道图像是否显示在屏幕上，然后才根据结果执行不同的动作。ImageFound()函数返回一个真值或假值，表明是否搜索到一个图像（或多个图像中的某个图像）。

示例：在条件块中调用ImageFound()

```
If ImageFound (10,"Save Button") then                       //如果找到“SaveButton”，则...

Click FoundImageLocation()                                  //点击查找到的图像的位置。

else                                                        //否则

LogError "Save Button not found"                            //记录错误。

end if
```

ImageLocation()

如果想知道图像的位置，可调用ImageLocation()函数返回其热点相对于屏幕左上角的坐标。下面的例子使用ImageLocation()函数比较两张图像的X坐标：

示例：使用ImageLocation比较图像在屏幕上的位置

```
If item 1 of ImageLocation ("ImageOne") < item 1 of ImageLocation ("ImageTwo")
then                                                        //如果ImageOne的X坐标小于ImageTwo的X坐标，则...

Log "Image One is to the left of ImageTwo"                  //记录消息。

End If
```

注：如果未查找到指定图像，`ImageLocation()`函数将产生异常。

使用`ImageLocation()`调整窗口大小

如果必须捕获想要停止拖到位置的图像，在窗口的边角执行`DragandDrop`会非常复杂。（如果只有空白桌面作为参考点，怎么办？）幸运的是，可向ImageLocation()函数中添加坐标完成上述任务。

示例：使用`ImageLocation()`调整窗口大小

```
Drag imagelocation("Corner")                                    //点击并按住图像的“边角”。

Drop imageLocation("Corner") + (10, -20)                        //向右移动10个像素，向上移动20个像素，然后释放。
```

搜索屏幕的一部分

相同图像出现在屏幕上多个位置时，通常需要一个方法指定所需的图像。一个方法是设置`SearchRectangle`全局属性。

`SearchRectangle`全局属性取两对屏幕坐标作为参数，这两个点定义了搜索区域的两个对角。（屏幕左上角的坐标是0,0。）

下面的范例脚本旨在在活动窗口中点击“定制”按钮（交叉的锤和扳手）。由于活动窗口中的定制按钮同其它窗口中的定制按钮是一样的，不能保证查找的“定制”按钮图像就是所需的图像，但是，活动窗口左上角的红色关闭按钮是唯一的。（在后台窗口中是无色的。）如果将左上角的红色关闭按钮用作搜索矩形框，eggPlant Functional（从左到右，从上到下搜索）肯定首先在同一个窗口内查找到定制按钮。

示例：调整搜索矩形框

```
put ImageLocation("CloseButton") into UpperLeft

put RemoteScreenSize() into LowerRight

Set the SearchRectangle to(UpperLeft, LowerRight)

Click "Customize"

Set the SearchRectangle to ()                                   //将搜索矩形框恢复成SUT全屏。
```

缩放图像

默认情况下，查找到的所有图像均是原始分辨率（比例为1.0）。可设置defaultScale全局属性更改默认设置。

单个图像可在脚本里设定图像搜索时进行缩放。在脚本里设定图像搜索时，缩放图像的三种方法：

- 按缩放比例缩放
- 缩放到指定大小
- 动态缩放

按缩放比例缩放

该方法利用`Scale`参数按照原始大小的缩放比例缩放图像。按照图像搜索参数设置`Scale`参数。例如同Click命令一起使用，如下例所示。

示例：按缩放比例缩放

```
Click (image:"OK_button", scale:0.5)                    //以50%比例搜索图像。

Click (image:"OK_button", scale:(0.5, 1.0, 1.5))        //以这三种大小搜索图像。

Click (image:"OK_button", scale:0.5 to 1.5 by .25)      //在这个0.5-1.5范围内搜索图像，增量为.25（即.5, .75, 1.0, 1.25, 1.5）。
```

注：按指定值缩放图像时必须使用`Image`参数。
如果`defaultScale`全局属性已设置，将会被替代。
如果要在`defaultScale`中采用特定图像搜索的特定比例，可利用下例中的类似代码。

示例：在defaultScale中使用特定比例

```
Click (image:"OK_button", scale:(0.5, 1.0, “Default”?))         //按一半大小、正常大小和当前“默认”大小搜索图像。
```

确定缩放比例

某些情况下，从一个设备迁移到另一个设备或变更SUT上图像元素的大小时，可能很难确定缩放比例（比如测试Mac设备上的Dock）。

应对上述情景，可首先采用缩放比例范围（如上述按缩放比例缩放中的例子），获取额外信息确定使用foundImageInfo()函数时应采用的正确缩放比例。搜索图像采用的缩放比例记录在`foundImageInfo()`函数返回的结果中。

示例：确定搜索图像时采用的缩放比例

```
Click "OK_Button"

Log foundImageInfo().Scale                              //记录搜索图像时采用的缩放比例。
```

缩放到指定大小

该方法可利用`scaleToSize`参数为新图像指定特定的大小。要执行此操作，设置任何图像搜索时，都需设置`scaleToSize`参数。例如，同MoveTo命令一起使用，作为

`imageLocation()`函数的一部分。

示例：缩放到指定大小

```
MoveTo imageLocation(Image:"anImage", scaleToSize:(300,100))
```

注：如果指定的新尺寸与原始图像捕获不相称的话，会造成图像大小出现偏差，从而造成显示的图像变形。

动态缩放

从eggPlant Functionalv12.20开始，eggPlant Functional在捕获图像时记录SUT（被测系统）的屏幕尺寸。然后，比较捕获时所记录的SUT屏幕尺寸与图像搜索时的SUT屏幕尺寸的差异，自动缩放图像。

为此，需要设置`Proportional`或`Stretch`参数。设置这两个参数均不会按捕获时的尺寸搜索图像。

- *Proportional*：图像在宽和高两个方向上的调整是一致的，保持比例不变。根据SUT屏幕在高或宽上的最小变化按比例缩放，因此如果SUT屏幕尺寸在某个方向上的调整比另一个方向上的调整要大，则按照较小的调整缩放图像。
- *Stretch*：图像将在宽和高两个方向上缩放，但是不需要保持捕获图像的比例。

示例：动态缩放

```
Click (image:"OK_button", scale:(“Proportional”?, 1.0,
1.5))                                   //以这三种大小搜索图像。
                                        如果想要eggPlant Functional按捕获图像的原始大小搜索图像，必须包含比例
                                        1.0。
```

注:eggPlant Functionalv12.20之前版本在捕获图像时不会记录SUT屏幕尺寸。可在套件的图像页签中设置历史图像的捕获尺寸（必要的话可批量设置）。也可设置`DefaultCaptureScreenSize`，在没有记录的SUT屏幕尺寸时使用。

保存缩放后的图像

有时需要保存缩放后的图像。例如，可能需要保存缩放后的图像进行分析或创建永久记录。使用`SaveImage`命令保存缩放后的图像。

最佳实践：在脚本运行过程中缩放图像而无需保存的话是非常快的操作，通常比从磁盘读取额外的图像等图像集合操作要快。因此，推荐在脚本运行过程中采用动态图像缩放，（使用上述缩放方法或者设置DefaultScale），而不是保存缩放后的图像用于图像搜索。

旋转图像

如果想要图像显示的角度与捕获时的角度不同，可设置图像搜索的旋转参数。

下面的例子说明如何在Click命令中使用`Rotate`参数。

```
Click (name:"appIcon", rotate:180)
//搜索appIcon，并且角度与原始角度相差180度。

Click (name:"appIcon", scale:45 to 90 by 5)
//搜索appIcon，角度倾斜45-90度，每次增量为5度。
```

优化脚本性能

默认情况下，eggPlant Functional搜索一张图像的次数最多为6次（每次搜索前都有延迟），最后在执行全屏刷新和最后一次搜索后才上报未能查找到图像。

如果知道在脚本运行的特定时刻图像已经存在或不会再存在，则不需要等待长时间搜索。

以下是一个范例函数可*快速*从指定参数列表中搜索图像：

示例：FastImageFound

```
function fastImageFound                                 //指定函数。

set originalSearchCount to the imageSearchCount         //获取ImageSearchCount全局属性的初始值。

set the imageSearchCount to 1                           //将ImageSearchCount设置为1。

set returnValue to ImageFound (parameterList)           //将returnValue设置为ImageFound函数的值。

set the imageSearchCount to originalSearchCount         //恢复到ImageSearchCount原始值。

return returnValue                                      //返回ImageFound的值。

end function
```

改写该函数

除了`FastImageFound`函数之外，可插入命令或函数满足特殊需求。例如，使用“FastClick”。

- 创建图像集合
- 图像更新工具

‹ 编写文本图像生成器（TIG） up 创建图像集合 ›

Printer-friendly version