欢迎来到 Yahoo!Widget Engine 的精彩世界！

亲爱的用户你好！通过学习本课程你可以用 XML 与 JavaScript 编写简单的 Widget。首先，我们将介绍编写 Widget 需要用到的一些常用工具，以及Windows Widget的封装方式；然后介绍如何编写最佳的 XML 文档；接下来深入探讨 Yahoo!Widget Engine 中的 JavaScript 脚本语言；最后介绍向我们的官方网站时提交 Widget 时需要遵循的步骤。

下面是你在阅读本课程之前需要了解的：

绿色代表这是一个指示区域,请依照其中的指示来进行操作。

蓝色代表这是程序代码区域,其中包含的是可以使用的 XML 或 JavaScript 的代码。

红色代表这是一个包含了重要信息的区域,请依照其中的指示进行操作以避免可能出现的问题。

Yahoo!Widget Engine 简介

编写 Widget 需要的工具

Yahoo!Widget Engine 3.x

可以从 http://cn.widget.yahoo.com/ 下载。

文本编辑器

NotePad 或者 或者其他任何支持 Unicode 的文字编辑器。

图形编辑器

任何一款具有图形编辑功能的软件。

一些有用的参考资料

「Yahoo!Widget Engine 参考手册」(可以在 http://cn.widgets.yahoo.com/ 找到最新的版本)。

在这里 http://cn.widget.yahoo.com/也可以找到很多有用的内容。

Widget是以「包」的形式出现的，可以将“包”想象成一个文件夹,里面包含该 Widget 运行需要的一切内容。

可以使用 Widget Converter 来查看包中的内容；也可以将扩展名 .widget 改为 .zip，然后解压缩文件。Windows Widget 包基本上都是更改了后缀名的 zip 文件。

所有 Widget 都有一个名为 "Contents" 的文件夹，可以在其中找到一个 .kon 文件，通常还会包含一个存放图片的文件夹。

Widget 中文件类型的简略分类

.kon

包含 Widget工具的主要代码。当用户双击 Widget 时 Yahoo!Widget Engine 会首先找到这个文件，读取其中的内容。 .kon 包含了初始图片的位置，程序代码 (XML 或 eXtensible Markup Language)，以及偏好设置等项目。有时候，尤其是在比较复杂的 Widget工具中，JavaScript 会保存在 .js 文件中。

.js

包含大部分 (如果不是全部) Widget 执行所需要的 JavaScript 代码。这个文件中不能包含 XML 标签。

可能还会遇到其它一些文件类型，它们通常是各类图片。

XML 基础

请在你的纯文本编辑器中打开 Weather.widget/Contents/The Weather.kon。

首先,你最好先在第一行进行 XML 声明来开始你的 Widget代码。XML 声明会告诉 Yahoo!Widget Engine 正在读取的内容是有效的 XML ，以及 XML 的版本信息和编码类型。

```
<?xml version="1.0" encoding="UTF-8"?>
```

指定的文件编码类型必须与实际文件编码类型相符。否则 Widget中可能会出现一些奇怪的字符，甚至无法载入。最佳编码方案是 UTF-8 或 UTF-16。如果复制另一个 Widget 的文字，请记住这点，确保编码类型是你所预期的。

在 XML 声明下方，肯定会有一个 <widget> 标签来告诉 Yahoo!Widget Engine 代码从此处开始，该 Widget 执行所需的 Yahoo!Widget Engine 的最低版本和 Widget 自己的版本信息。你还会看到 .kon 文件的结尾有一个 </widget> 标签，它告诉 Yahoo!Widget Engine 代码到此结束。

```
<widget version="1.6.7" minimumVersion="1.7">
  [...]
</widget>
```

XML 标签一定会有一个用来结束它的标签 (通知解析器停止读取该元素的代码)。例如，如果你有一个 <image> 标签，你就必须使用相对应的 </image> 标签来结束它。可以把标签写成树形结构，例如：

```
<widget>
  <image>
    <src>Images/My Great Image.png</src>
    <hOffset>74</hOffset>
    <opacity>85%</opacity>
  </image>
</widget>
```

也可以像这样写，以 /> 来结束标签：

```
<widget>
  <image
    src="Images/My Great Image.png"
    hOffset="74"
    opacity="85%"
/>
</widget>
```

混合上述两种格式也是可以的 (但这种情况下请勿使用 "/>")：

```
<widget>
  <image src="Images/My Great Image.png">
    <hOffset>74</hOffset>
    <opacity>85%</opacity>
  </image>
</widget>
```

需要强调的是，请确保所有开头的标签都有一个结束标签与之对应（而且都必须正确拼写），这样你执行 Widget时才不会遇到一些奇怪的错误。

Widget 中一些标签可以位于其它标签的树形结构之下，为该标签提供所需的属性。例如，<image> 标签中通常会有一个 <src> 标签，以指出该图片的来源。

Widget 常用标签使用范例如下所示：

<?xml version="1.0" encoding="(Your .kon file encoding)"?>：XML 声明，它会通知解析器正在读取哪一种 XML 及编码。

<widget>：表示 Widget 编码开始的强制标签。

<debug>：开启与关闭调试功能。

<version>：定义 Widget 版本信息。

<minimumVersion>：定义 Widget 执行所需要的 Yahoo!Widget Engine 的版本。

<window>：定义 Widget 的显示区域。窗口范围之外的任何内容都会被裁减掉。

<name>：定义窗口名称。

<title>：定义窗口的标题。

<height>：定义窗口的高度，单位为像素。

<width>：定义窗口的宽度，单位为像素。

<visible>:这是一个*布尔值* (1 或 0,可用 true 或 false 来取代),它定义 Widget 是否可见。

<image>：允许在 Widget 中使用图片。

<name>：定义图片名称。

<hOffset>：定义从窗口的左上角水平位移多少个像素。

<vOffset>：定义从窗口的左上角垂直位移多少个像素。

<hRegistrationPoint>：定义在位移与旋转等操作中以其为基础的 X 像素坐标。默认为 0 (图片的左侧)。

<vRegistrationPoint>： 定义在位移与旋转等操作中以其为基础的 Y 像素坐标。默认为0 (图片的上边缘)。

<rotation>：定义图片旋转的度数。

<opacity>：0 到 255 之间的数值，它定义了图片的透明度。也可以使用百分比数值。

<onMouseDown>：定义在图片对象上按下鼠标按钮时所要进行的操作。

<onMouseUp>：定义在图片对象上放开鼠标按钮时所要进行的操作。

<text>：允许在 Widget 中使用文本。

<name>：定义文本的名称。可用于稍后在 JavaScript 中的操作。

<data>：定义文字的内容。

<hOffset>：定义从窗口的左上角水平位移文字多少个像素。它会受到 <alignment> 标

签的影响。

<vOffset>：定义从窗口的左上角垂直位移文字多少个像素。它使用文字的基线来位移，而不使用文字的顶点。

<alignment>：可以在此处使用 "left"、"center" 及 "right" 数值，以定义从哪里开始显示文字。

<color>：以 16 进位数值定义文字的颜色（例如：#000000 是黑色，#8000FF 是紫色，#FFFFFF 是白色）。

<font>：定义文本的字体。
<size>：字号，以点为单位。
<opacity>：0 到 255 之间的数值，定义文字的透明度。也可以使用百分比数值。
<onMouseDown>：定义在文字对象上按下鼠标按钮时发生的事情。
<onMouseUp>：定义在文字对象上放开鼠标按钮时发生的事情。

<textarea>：允许在 Widget 中输入文字。
<name>：定义 textarea 的名称。可用于稍后在 JavaScript 中的操作。
<hOffset>：定义从窗口的左上角水平位移多少个像素。它会受到 <alignment> 标签的影响。
<vOffset>：定义从窗口的左上角垂直位移多少个像素。它使用文字的基线来位移，而不使用文字的顶点。
<lines>：定义一次显示多少行。
<columns>：当有更多文字输入而必须卷动之前，定义文字输入的宽度。
<color>：以 16 进位数值定义文字的颜色（例如：#000000 是黑色，#8000FF 是紫色，#FFFFFF 是白色）。
<font>：定义在输入中使用的字体。
<bgColor>：以 16 进位数值定义输入背景的颜色（例如：#000000 是黑色，#8000ff 是紫色，#ffffff 是白色）。
<bgopacity>：0 到 255 之间的数值，定义输入背景的透明度。也可以使用百分比数值。
<onKeyPress>：定义每次按下按键时进行的操作。
<action trigger="(Some event here)">：可让某个事件在 Widget 第一次加载时、计算机从睡眠模式恢复之后、偏好设置变更时，以及其它更多情况下发生。可以针对不同事件多次使用！
<about-box>：与图片对象中 <src> 标签的工作方式一样，但它是用于“关于”窗口的。

还有许多其它标签，就不一一列出了。请查阅「Yahoo!Widget Engine 参考手册」来了解其它对象、属性及动作触发器。

JavaScript 基础

在以上的 XML 标签中，有一个共同的特征：包括 <name> 标签，此标签允许在 JavaScript 中对此对象进行操作。

JavaScript 又称为指令码语言，它无须编译即可执行。我们之前曾经提到过，使 JavaScript 与你的 XML 对象能够搭配使用的主要元素是名称标签。还有许多方法也可以执行此项操作，我们先从一种最简单的方法开始：

```
name.attribute = 255;
```

• 句点前面的部分是放置对象名称的地方。只能以字母（且必须是小写字母)开头，而且只能包含字母数字以及下划线线（"_" 字符)。常用的命名方式如下所示：

- myGreatName79
- my_great_name_79
- mygreatname79

•，句点后面的部分是对象的属性。为句点前的对象设定其属性。
• "=" 运算符号会将它后面的数值指定给对象的属性。
• "=" 右边是指定给对象属性的数值。有三种不同的数值。

- *布尔值* - 它可以是 0 或 1，或者如果你想更容易识别的话，可以将其设定为 false 或 true。
- *文字值* - 它可以是任何数字。
- *字符串* - 此种类型的数值可以包含文字与数字。当设定此数值时，要用引号将其括起来。

下面的一些范例，可以帮助你更详细的了解。

```
sun.opacity = 145;
main_window.visible = true;
main_window.shadow = 0;
myText.data = "Hello, World!";
weatherinfo.tooltip = "Today, I think, is a good day for some sun.";
colorChangedText.color = "#6C189C";
```

上面的范例中有一个很酷的地方，如果已经使用 XML <name> 标签来正确定义及识别元素，当计算机读取以上的程序代码时，将会看见以纯英文记录的内容。

```
  Change the opacity of the object named "sun" to 145/255, or about 57%. This uses a literal
value.
  Make the Widget window "main_window" visible. This uses a boolean value.
  Change the Window "main_window" so it doesn't render a shadow (Mac only). This uses a
boolean value.
  Change the text object "myText" so that it reads "Hello, World!" This uses a string.
  Change the tooltip of the "weatherinfo" object so that it reads "Today, I think, is a
good day for some
  sun." This uses a string value.
  Change the color of the text "colorChangedText" to a royal violet. This uses a string
value (in
  hexadecimal format).
```

可以定义变量，变量与数学方程式中的变量类似。除了标准文字值，例如 47 或 3.5 以外，也可以将字符串与布尔数值赋给它们。如对象名称一样，变量名也是对象。

```
var literalValue = 83.4523;
var booleanValue = true;
var stringValue = "I can see my house from here!";
```

Yahoo!Widget Engine 能够获取的系统属性大多数是动态的。Widget 中的系统属性与对象属性之间，唯一区别就是对于大多数的系统属性只能读取并使用它们，而不能设定。但是也有一些例外，例如 system.volume，可以将其设定至 0 与 16 之间的值。下面让我们来看一下这些属性的作用。

```
image.opacity = system.cpu.idle * 2.55;
                                                    // 由于透明度属性是超出
255 的文字值
                                                    // 且 system.cpu.idle
会返回从 0 到 100 的数字，因此
                                                    // 我们将该数值乘以
2.55 让它可使用全部的透明度
                                                    //这些都是单行批注...
对于
                                                    // 提供解释或注意事项很
有帮助
system.mute = true;
                                                    // 这将会关闭计算机的声
音效果
                                                    // 将其设定为 false 则
开启声音效果
```

编写第一个Widget（“Hello, World!”）；

现在让我们将新学到的知识付诸实践。

在纯文本编辑器中建立新文件，并将其命名为 "My First Widget.kon"，将以下代码粘贴进去。

```
<?xml version="1.0" encoding="UTF-8"?>

<widget>
  <debug>on</debug>

  <window>
    <name>main_window</name>
    <title>My First Widget</title>
    <height>30</height>
    <width>300</width>
    <visible>true</visible>
  </window>

  <text>
    <name>myText</name>
    <color>#FF0000</color>
    <size>18</size>
    <alignment>left</alignment>
    <vOffset>25</vOffset>
    <hOffset>2</hOffset>
  </text>

  <timer>
    <name>timer</name>
    <interval>1</interval>
    <ticking>true</ticking>
    <onTimerFired>
       var cpuLoad = system.cpu.activity;
       myText.data = cpuLoad + "% CPU load";
       myText.opacity = cpuLoad * 2.55;
    </onTimerFired>
  </timer>

</widget>
```

请注意代码的缩进方式。为了使你的程序代码清楚易读，应该始终对其进行缩进排列。使用 tab 键是执行此操作最简单与最好的方法，也可以使用空格键。

此程序代码每秒钟执行一次，如定时器的 "interval" 标签所定义的一样，可以将它设为更长时间或更短时间。

保存 "My First Widget.kon"，然后可以双击运行它。

注意：如果 CPU 负载不高，可能无法看见它，如果发生此类情况，请播放视频文件或

开启某些程序，这样将会加大cpu的负载，使得该 Widget 可见。

让我们看一下 "if...else" 陈述式的简介。

```
if (cpuLoad > 80)
{
    myText.data = "You're working your computer rather hard!";
}
else
{
    myText.data = "The computer isn't doing much right now... Get back to work!";
}
```

某些人会使用如上所示的大括号来写程序。有些人则会在与 if...else 条件陈述式同一行上使用大括号，以使其更为精简。

```
if (cpuLoad > 80) {
    myText.data = "You're working your computer rather hard!";
} else {
    myText.data = "The computer isn't doing much right now... Get back to work!";
}
```

Yahoo!Widget Engine 不限制你编写程序的风格，也不限制你缩排程序代码的方式。

对 "My First Widget.kon" 进行以下改动并保存它，然后在调试窗口中按下一下「重新加载Widget工具」。

```
<?xml version="1.0" encoding="UTF-8"?>

<widget>
  <debug>on</debug>

  <window>
    <name>main_window</name>
    <title>My First Widget</title>
    <height>30</height>
    <width>300</width>
    <visible>true</visible>
  </window>

  <text>
    <name>myText</name>
    <color>#FF0000</color>
    <size>18</size>
    <alignment>left</alignment>
    <vOffset>25</vOffset>
    <hOffset>2</hOffset>
  </text>

  <timer>
    <name>timer</name>
    <interval>1</interval>
    <ticking>true</ticking>
    <onTimerFired>
    <!--
       var cpuLoad = system.cpu.activity;
       myText.data = cpuLoad + "% CPU load";
       if (cpuLoad < 40)
       {
          myText.opacity = 102;
       }
       else
       {
          myText.opacity = cpuLoad * 2.55;
       }
      //-->
    </onTimerFired>
  </timer>

</widget>
```

这会使 Widget 随时保持一些可见度。请注意，我们已经将一些 XML 批注标签 (<!-- 与 //-->)

加到 JavaScript 中，XML 解析器不会在它读到 if 陈述式中的 "<" 符号时误以为插入了另一个 XML 标签。

到目前为止，你的 Widget 还只是纯文本而已。如何为你的用户提供一些方法来满足他们的偏好设置呢？ 只需将偏好设置标签加到程序代码中便可新增偏好设置。如果你想要知道如何新增不同类型的偏好设置，可以参考「Yahoo!Widget Engine 手册 PDF」。

下面是popup菜单偏好设置的例子。包括偏好设置的标题、类型、popup的所有选项、第一次打开偏好设置时的默认值，以及偏好设置的简要描述等等。

```
<preference>
  <name>continentSelector</name>
  <title>Your Continent:</title>
  <type>popup</type>
  <option>Africa</option>
  <option>Antarctica</option>
  <option>Asia</option>
  <option>Australia</option>
  <option>Europe</option>
  <option>North America</option>
  <option>South America</option>
  <defaultValue>North America</defaultValue>
  <description>Select your continent from the menu above.</description>
</preference>
```

颜色偏好设置可以让用户选择适合文字颜色;更新间隔偏好设置让用户决定他们想要 Widget 更新 CPU 信息的时间间隔。。。

下面介绍执行这些操作的程序代码。

在 Widget 标签中的某处新增此程序代码，但不要将其以树形结构置于另一个标签中。

```
<preference>
  <name>textColorPref</name>
  <title>Text Color:</title>
  <type>color</type>
  <defaultValue>#FF0000</defaultValue>
  <description>Select a color for your text.</description>
</preference>

<preference>
  <name>textFontPref</name>
  <title>Text Font:</title>
  <type>font</type>
  <defaultValue>Futura Medium</defaultValue>
  <description>Select a font to use for your text.</description>
</preference>

<preference name="timerIntervalPref">
  <title>Update Interval:</title>
  <type>slider</type>
  <minLength>1</minLength>
  <maxLength>15</maxLength>
  <ticks>15</ticks>
  <tickLabel>1</tickLabel>
  <tickLabel>Seconds</tickLabel>
  <tickLabel>15</tickLabel>
  <defaultValue>1</defaultValue>
  <description>Select how often you want the CPU information to update.</description>
</preference>
```

现在默认的颜色值为红色、用户可以选择他已安装的任何字体，设定每秒刻度标记的定时器间隔。接下来还必须将这些偏好设置连结至 Widget 代码 ，以便在它们发生变动时执行某些特定操作。

用户自己编写函数的方法...

```
function muteVolume()
{
  system.mute = true;
}
```

使用时如此调用即可。

```
muteVolume();
```

可以在调用的时候将若干个变量传送至被调函数中，只需将逗号置于各变量之间即可。

```
function adjustVolume(volumeToUse, muteIt)
{
  if (muteIt) // 这将会检查是否有数据被传送至 muteIt 变量中。
  {
     system.mute = muteIt;
  }
  system.volume = volumeToUse;
}
```

调用函数时，请将变量用逗号分隔开来， 以便 Yahoo!Widget Engine 可以识别它们：

```
adjustVolume(14, false); // 将音量设定为 14，并关闭系统声音
```

这样也可能侥幸成功，因为我们所编写的函数不处理此处的 muteIt 数值。请注意，这里并没有逗号，因为只传送一个数值：

```
var addedValue = 8
adjustVolume(6 + addedValue); // 将音量设定为 14 而非 16
```

Yahoo!Widget Engine 提供许多内建函数，跟 JavaScript 本身一样。大多数函数要求你提供一个或多个数值，但某些函数不需要。关于 Yahoo!Widget Engine 内建函数的完整信息，请查看「Yahoo!Widget Engine 参考手册」。

```
alert("Your house is on fire.", "Call 911", "That's nice");
print(unescape("This%20is%20some%20web%2Dready%20text%21"));
beep();
var textToSpeak = "Your computer is talking to you. What say you?";
speak(textToSpeak);
closeWidget();
```

将以下程序代码写到文件结尾，就在 </widget> 标签的上方。

```
<action trigger="onLoad">
   function updateBehavior()
  {
     myText.font = preferences.textFontPref.value;
     myText.color = preferences.textColorPref.value;
     timer.interval = preferences.timerIntervalPref.value;
  }
   updateBehavior();
</action>
<action trigger="onPreferencesChanged">
   updateBehavior();
</action>
```

保存 .kon 文件，重新加载你的 Widget工具。

这段代码将会使 Widget 在每此启动或其偏好设置发生变动时更新其外观与行为。

偏好设置将会在 Widget 关闭时自动保存，因此不需要手动保存偏好设置。

培养良好的 Widget 编写习惯

现在你已经拥有自己的 Widget 了，应该如何将它打包，以便其它人使用并分享呢？

以下是你与他人共享 Widget 之前应该进行的一些操作：
要做的第一件事就是设定 <debug>off</debug> 来关闭调试窗口。
将所有的图片，.kon 文件以及其他所有东西放进 “Contents” 文件夹，然后将 Contents 文件夹放进另一个以你的 Widget 名称命名的文件夹中，例如 “My Great Widget”。然后，为文件名称加上 .widget 后缀名。

分享成果

现在你的 Widget 已经最优化，就可以将其打包以便跨平台使用了！要将单一文件从文件夹中取出，在Windows 中会变得难以处理，这时需要使用 Widget Converter。

Widget Converter 可以让你以 Windows Widget 格式打包Widget 。你需要做的就是取得含 .widget 后缀的文件夹，并将其拖到 Widget Converter 的窗口上。Widget Converter 的屏幕上将会显示此 Widget 的类型。

接下来只需点击「转换」按钮，Widget Converter 将会完成剩下的工作。

如果你遵循了以上所有步骤，那么就可以将你的 Widget 提交到我们的官方网站上了，以便更多

的用户可以分享你的成果。

提交前请注意以下一些事项：

- Widget 必须执行在 **Widget 说明**中提到的函数。
- 提交的文件（**.dmg、.zip、跨平台 Widget 文件**或 **.sit**）必须准备好供用户下载（我们不会因任何理由重新封装提交内容）。
- 调试窗口不能在 Widget 执行时显示出来（由于错误或仍然开启着 **<debug>** 选项）。
- Widget 不能包含你不是其唯一拥有者，或你尚未获得使用权的数据。
- Widget 不能包含随 Yahoo!Widget Engine 提供的 Widget 中的图片、声音或 Info.plist 文件（人们使用时经常遇到的问题是在其 **Info.plist** 中出现 **Part of Konfabulator** 字符串，以及在其「关于窗口」上使用未经 **Getty Images** 授权的图片）。
- Widget 不能包含任何具有冒犯性意识的图片、文字或声音。
- Widget 不是本教学课程的直接结果 – 亦即，如果 Widget 没有本教学课程所提供内容中的主要新增部分，我们将拒绝接受它。

我们希望你的 Widget 具有以下一些特性：

- 具有漂亮的外观。
- 提供有用的功能。
- 具有一些独特功能（可将其与官方网站中『更多Widget』项目中的其它 Widget 区别开来）。

我们保留拒绝任何未达到这些标准的 Widget 的权利。

你的 Widget 符合这些标准吗？ 如果符合，并且你打算将 Widget 提交给我们，那么你需要做的就是先到我们的网站上建立一个你的账号，只需进入 http://cn.widget.yahoo.com/ ,建立您的账号,填写你的信息，登录;点击「发表Widget工具」，填写必要信息，提交Widget工具，一切便大功告成了！