本页由killman.liu翻译自原英文教程 object页

Object（对象）教程

总结

由于Orx是数据驱动的，我们只需要两行代码创建一个Viewport（视口）和一个Object（它们的所有属性都定义在配置文件（01_Object.ini）中。

Viewport关联到一个按照配置文件中的信息隐含创建的Camera(摄像头)。在配置文件里，你还可以设置它们的大小，坐标，对象的颜色，缩放，旋转，动画，物理属性等等。你甚至无需增加一行代码就可以让任何的配置获得随机值。

在后面的一个示例中我们将看到如何使用一行代码生成复杂的Object体系甚至整个 Scene（场景）（所有的背景对象和普通对象）。

现在，你可以尝试取消01_Object.ini中某些行的注释，自己尝试一下，然后再继续学习这个教程。完整的选项列表请查看CreationTemplate.ini。

详细说明

创建一个object是相当简单的。不过，我们首先需确保已经加载了定义了所有object(对象)的属性的配置文件。我们还要通过viewport/camera组合显示创建好的object(对象)。

不要慌张！所有这些都很容易。

在这篇教程中，我们将加载一个位于父目录中的配置文件。正如你可能想到的，在所有的可执行程序都根据其构建类别（mingw, msvs2005, msvs2008, 等）位于各自的子目录的情况下，我们不打算在每个地方重复同样的配置文件。[1)]

在我们的例子中，加载配置文件使用类似下面这行代码的方式实现：

```
orxConfig_Load("../01_Object.ini");
```

然后我们创建viewport(视口)。注意 camera的创建是按照为这个viewport预置的配置信息自动完成的。

```
orxViewport_CreateFromConfig("Viewport");
```

我们差不多完成了。现在我们只需要创建 object。

```
orxObject_CreateFromConfig("Object");
```

就这样了。object(对象)已经创建，并且由于在camera的视觉平截体（frustum）内，将会被显示出来。 现在，因为我们使用Orx默认的启动器，我们需要申明我们的插件入口点（这里是我们的Init函数）。这可以使用一个宏很容易地实现。

```
orxSTATUS Init(){...}
```

```
orxPLUGIN_DECLARE_ENTRY_POINT(Init);
```

因为orx是数据驱动的，我们不需要手动加载任何数据，例如一个sprite（精灵）。一切都由数据管理器为我们搞定，它会确保sprites不在内存中重复并在其不再使用时自动释放的 如果你查看配置文件，在[Object]这一节，你将看到你可以设定所有的对象属性，例如 graphic (sprite)，锚点，颜色，透明度，物理属性，坐标，旋转，缩放，tiling(平铺)（重复），动画，视觉特效，等等。 不要担心，这一切都将在后面的教程中讲到。

现在我们拥有了一个object（对象），我们需要学习如何与之交互。这将我们带入第二个教程：clock.

资源

源代码：01_Object.c

配置文件：01_Object.ini

[1]
不过，如果你的配置文件名字与可执行文件匹配并且在同一个文件夹下，它将被自动加载。