MORGAN KAUFMANN

Maya Python

游戏与影视编程指南

A Complete Reference for
Maya Python and the Maya Python API

[美] Adam Mechtley [美] Ryan Trowbridge 著
宋松 译

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

Maya Python

游戏与影视编程指南

A Complete Reference for
Maya Python and the Maya Python API

[美] Adam Mechtley [美] Ryan Trowbridge 著
宋松 译

人民邮电出版社
北京

图书在版编目（CIP）数据

Maya Python游戏与影视编程指南 / （美）梅切特里(Mechtley,A.)，（美）特罗布里奇（Trowbridge,R.）著；宋松译. -- 北京 : 人民邮电出版社，2016.2
ISBN 978-7-115-40669-9

Ⅰ. ①M… Ⅱ. ①梅… ②特… ③宋… Ⅲ. ①三维动画软件－指南②软件工具－程序设计－指南 Ⅳ. ①TP391.41-62②TP311.56-62

中国版本图书馆CIP数据核字(2015)第300721号

内 容 提 要

本书涵盖了与 Maya 相关的 Python 的主要内容，解答了有关 Maya 的多个 Python 实施问题，包括最强大的 PyMEL 和漂亮的 PyQt 用户界面。本书内容包括基础知识和更高级的主题，分为 3 部分，共 12 章。第 1 部分为 Python 和 Maya 的基础知识，第 2 部分是使用 Python 设计 Maya 工具，第 3 部分介绍了 Maya Python API 基础知识。其中，面向对象编程和过程式编程、环境设置和 PyQt GUI 等一般性主题也有所涉及，但着眼点是在 Maya 中的集成。掌握了本书的内容，读者可为以后的学习奠定坚实的基础。

本书适用于运用 Maya Python 或 Maya API 编程的专业人员，以及影视特效脚本编写的专业人员。

◆ 著　[美]Adam Mechtley　[美]Ryan Trowbridge
译　宋　松
责任编辑　赵　迟
责任印制　陈　犇

◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路 11 号
邮编　100164　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京昌平百善印刷厂印刷

◆ 开本：787 × 1092　1/16
印张：19.25
字数：462 千字　2016 年 2 月第 1 版
印数：1 – 2 200 册　2016 年 2 月北京第 1 次印刷
著作权合同登记号　图字：01-2014-0768 号

定价：79.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315

致谢

本书的完成离不开许多人的帮助，非常感谢他们。首先感谢我们两个人的妻子Laurel Klein和Brenda Trowbridge，没有她们的不懈支持，我们将一筹莫展。还要感谢本项目的各位贡献者付出的努力，Riot Games的Seth Gibson编写了第3章和第5章，Autodesk的Kristine Middlemiss编写了第8章，Autodesk的Dean Edmonds进行了技术编辑，他拥有丰富的知识，能够涵盖本书的内容。

还要感谢Focal Press团队在此项目上为我们提供的帮助，包括Sara Scott、Laura Lewin和Anaïs Wheeler。另外，感谢Steve Swink让我们有机会与Focal Press的精英们取得联系。

感谢Maya Python和API社区中的所有人，他们不仅推动了我们的成长，还为我们所有人带来了丰富的知识。我们不可避免地会遗漏许多重要人物，但还是要特别感谢Chad Dombrova、Chad Vernon、Paul Molodowitch、Ofer Koren，以及帮助维护tech-artists.org和PyMEL社区的每个人。

最后，感谢读者对本书的支持。如果对本书有任何建议，请联系我们！

引言：欢迎使用 Maya Python

假设你正在Maya中为一个角色创建动画。在为此角色创建动画的过程中，你会发现自己在重复以下几个步骤。

- 引用一个角色。
- 导入一个动作捕捉动画。
- 设置帧范围。
- 引用背景场景。
- 配置摄像机。

如果参与一个包含10到50名动画师的大型制作项目，这些简单的步骤可能会带来一些问题。如果分解此过程，可以发现许多适合引入工具的地方。

- 首先，动画师需要查找角色的正确路径。可以在任意给定时间更改此路径，所以让动画师任意选择此文件可能导致挑选了错误的文件。包含角色列表的简单工具可帮助快速、可靠地完成此任务。
- 接下来，你需要一个工具来对将运动捕捉数据正确地导入角色控件的过程进行组织和管理。在此过程中，此工具也可设置动画的帧范围并将摄像机移动到正确的位置。
- 最后一步是引用正确的背景场景，动画师每次手动搜索正确的文件时，可能都要花一分钟。可创建另一个简单工具，显示可供选择的所有背景的列表。

你会看到这些工具能够节省时间，让文件选择更准确，而且让动画师能专心完成其创造性工作。创建这些工具的最佳方式是使用Maya的一种内置脚本语言——具体来讲就是Python。

Python是一种最初在Maya外部开发的脚本语言，所以它拥有强大的功能集合和庞大的用户群。在Maya 8.5中，Autodesk添加了对Python脚本语言的正式支持。此语言已合并到Maya现有的编程界面中（MEL脚本语言和C++ API）。Maya嵌入式语言（MEL）已存在多年，所以你可能想知道为什么Python至关重要。通过更宽广的视角可很快发现许多重要优势。

- 社区：与Python相比，MEL拥有非常小的用户群，因为只有Maya开发人员使用MEL。而所有软件开发人员和许多类型的应用程序都可以使用Python。
- 强大功能：Python是一种更高级的脚本语言，它可完成在MEL中无法完成的工作。Python是完全面向对象的，而且它能够轻松地与Maya命令引擎和C++ API通信，支持使用同一种语言编写脚本和插件。即使使用C++编写插件，Python也支持在Maya脚本编辑器中交互式地测试API代码！
- 跨平台：Python可在任何操作系统上执行，这消除了针对不同操作系统、处理器架构或软件版本而重新编译工具的需求。事实上完全不需要编译！
- 行业标准：由于Python的优势，它正被迅速集成到对娱乐行业专业人员至关重要的许多其他应用程序中。可以在Maya和你的渠道（如MotionBuilder）中的其他应用程序之间轻松共享库。

0.1 Python 与 MEL 对比

使用Python而不使用MEL有许多理由，但这并不意味着必须完全放弃MEL！如果你的工作室已有多个MEL工具，而且它们已投入使用，则没有理由改变它们。可使用你目前的工具，无缝地将Python脚本集成到开发流程中。Python可调用MEL脚本，MEL也可调用Python脚本。Python是一种像C++一样的深奥语言，但它的语法简单且很容易上手。

不过Python对复杂数据的处理比MEL更得体。MEL程序员有时会尝试模仿复杂的数据结构，但这么做常常需要麻烦的代码，而且很难扩展。因为Python是面向对象的，而且它允许嵌套变量，所以它能够更轻松地处理这些情形。此外，Python访问文件和系统数据的速度比MEL快得多，这使各种工具能够在生产中更迅速地满足美工人员的需求。程序员在Python中拥有的选择也比MEL中多得多。由于Python存在的时间较长，所以你可能发现另一个用户已创建了一个模块来帮助Python执行你需要执行的任务。

如果还不理解其中一些语言，不要担心。本书可帮助你理解所有这些概念，为高效工作做好准备！

0.2 配套网站

本书应与配套网站内容一起阅读，该网站的地址为：http://maya-python.com。

配套网站包含本书中引用的示例项目的下载文件，以及大量有用链接和补充资料。

0.3 有关代码示例和语法的说明

本书包含许多代码示例，仅通过各种电子书格式提供给大家，因此我们无法保证何处将出现换行符。此问题对许多编程语言（包括MEL）都不是问题，但对换行符和空白来说Python是特殊的。因此，值得简要说明一下Python如何处理这些问题，以及我们在本书中选择如何解决它们。现在看看第一个示例！

空白

许多编程语言都对前导空白毫不重视，而使用花括号（{和}）等机制来表示代码块，就像下面这个虚构的MEL示例一样，该示例打印数字1到5。

```
for (int $i=0; $i<5; $i++)
{
    int $j = $i+1;
    print($j+"\n");
}
```

在本例中，代码块内的缩进是可选的。该示例也可重写为以下代码，而且将得到同样的结果。

```
for (int $i=0; $i<5; $i++)
```

```
{
int $j = $i+1;
print($j+"\n");
}
```

而Python使用前导空白来搭建代码块。这个示例在Python中看起来类似以下代码。

```
for i in range(5):
    j = i+1
    print(j)
```

尽管Python不关心代码块中的准确缩进方式，但它们必须缩进，在语法上才有效！Python社区中的标准是为每个缩进级别使用一个制表符或4个空格。

由于本书的格式要求，可能无法从电子书中复制和粘贴示例。我们在配套网站上提供了尽可能多的代码示例供下载，以尝试减轻此问题，这些示例可安全地复制和粘贴，只有一些短示例需要誊写。

换行符

Python还有一些约束换行符的特殊规则，这对我们特别重要，因为我们无法知道本书中的文本在不同的电子书硬件上的准确的显示方式。

大多数编程语言都允许以你想要的方式插入换行符，因为它们需要分号来表示代码中的换行。例如，下面这个虚构的MEL示例将构造句子“I am the very model of a modern major general.”，然后打印它。

```
string $foo = "I am the very" +
"model of a modern" +
"major general");
print($foo);
```

如果尝试在Python中采用相同方法，会得到语法错误。下面的方法将无法运行。

```
foo = 'I am the very' +
'model of a modern' +
'major general'
print(foo)
```

在Python中，在大多数情况下，必须在行末添加一个特殊转义字符（\），才能让它延续到下一行。你需要以下面这种方式重写前面的示例。

```
foo = 'I am the very' + \
'model of a modern' + \
'major general'
print(foo)
```

在一些特殊场景中，无需转义序列即可跨越多行，如代码在圆括号内或方括号内时。可在Python Language Reference（Python语言参考指南）的第2章中进一步了解Python如何处理换行符。

第1章将指出可在何处找到此文档。

我们的方法

由于Python的词法要求，很难编写出能保证为所有读者都能正确显示的代码示例。因此，我们的一般方法是使用Python允许的可选分号字符（;）来在打印的代码中指明实际的行尾。相信你会在应该存在回车的地方将它们视为行尾，以及在必要时调整缩进。

例如，尽管充分确信前面的示例很短，可为了给所有读者正确地显示，但在本书剩余内容中会以下面这种方式重写它们。

```
for i in range(5):
j = i+1;
print(j);
foo = 'I am the very model of a modern major general';
print(foo);
```

因为Python允许但不需要分号，所以我们避免在配套网站上的实际代码示例中使用它，但对于计算机上没有代码示例的读者来说，我们会以文本形式显示代码。

引号

还需要提及的是，本书中的一些代码示例使用了双引号（"），一些使用了单引号（'），还有一些使用了一种类型的三元序列（"""或'''）。尽管在阅读印刷版时可清楚地分辨我们使用了哪些标记，但双引号标记可能无法正确地复制和粘贴。总之，请确保反复检查了你尝试从本书复制和粘贴的任何代码。

注释

最后值得提及的是，像其他编程语言一样，Python允许在代码中插入注释。注释是仅供开发人员参考的语句，它们不会在程序中计算。创建注释的一种方式是，将它写为文字字符串。后面将更详细地介绍字符串，基本理念是将一个独立语句单独放在双引号中，它就会变成一条注释。在下面这段代码中，第一行是一条注释，描述了第二行的用途。

```
'''Use the print() function to print the sum of 5 and 10'''
print(5+10);
```

插入注释的另一种方式是，在注释语句上添加#符号作为前缀。可使用此技术注释一整行，或者将注释添加到行尾。下面两行代码完成与前面的代码段相同的任务。

```
# Use the print() function to print the sum of 5 and 10
print(5+10); # This statement should output 15
```

我们的许多示例，尤其是来自配套网站上示例文件的代码副本，都没有注释。我们选择限制注释以节省打印空间，但希望作为开发人员的你不会这么轻率！配套网站上的大部分示例都包含非常

全面的注释，而且我们可能会分解大段代码，以便一次讨论其中的一部分。

0.4 Python 动手实验

在Python介绍中，我们讲到了它在Maya中很有用。现在看看一个简短的示例项目，体会一下即将介绍的内容。如果还不能理解示例中的一些内容，请不要担忧，因为此项目仅用于演示。

在本例中，你将执行同一段脚本的两个版本，一个是在MEL中仅使用Maya命令编写的，另一个是使用Maya Python API编写的。这些脚本着重演示了使用Python代替MEL时，可实现的完全不同结果的示例。

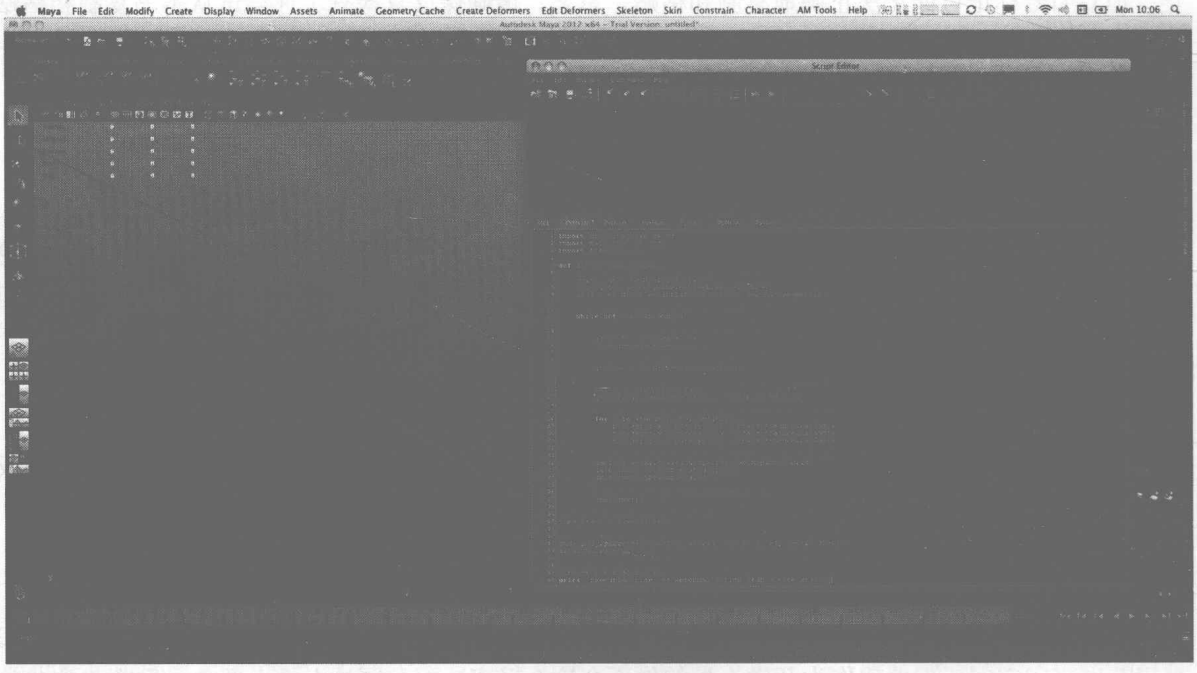
每段脚本创建一个细分为200 × 200个多边形（或39 802个顶点）的基本多边形球体，并向网格应用噪声变形。基本来讲，它们迭代球体的所有顶点，将每个顶点偏移一个随机值。我们提供给脚本的一个量对此值进行拉伸，以调整噪声量。

另外请注意，下面这个示例中的Python脚本使用了在Maya 2009中添加到API中的函数。因此，如果使用更低版本的Maya，你肯定可以检查源代码，但Python脚本将无法运行。

1. 在计算机上打开Maya应用程序。

2. 在Maya的主菜单中，选择Window（窗口）→General Editors（常规编辑器）→Script Editor（脚本编辑器），打开脚本编辑器。

3. 现在应该会看到出现了Script Editor（脚本编辑器）窗口，它分为两部分，在下半部分的上方，应该会看到两个选项卡，如图0.1所示。单击Python选项卡，将Python设置为当前激活的语言。



■ 图 0.1 Maya 脚本编辑器中的 polyNoise.py 脚本

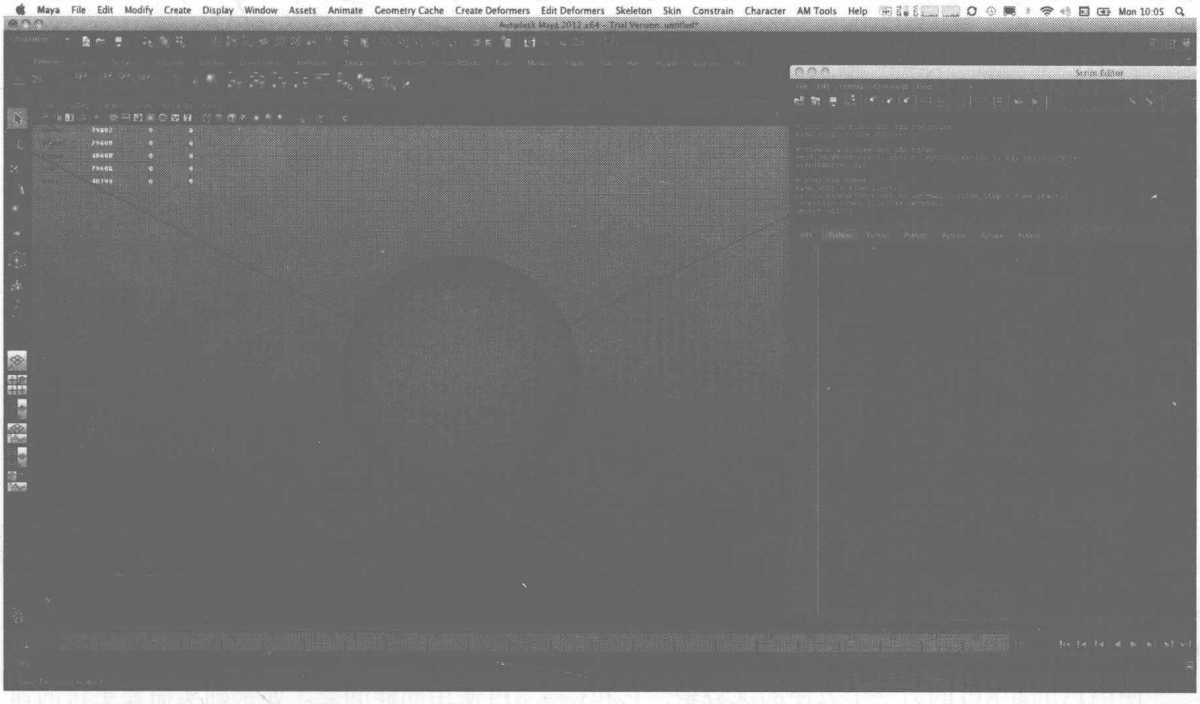
4. 从配套网站下载polyNoise.py脚本并记下它的位置。

5. 在Script Editor（脚本编辑器）窗口（而不是主应用程序窗口）中选项菜单选项File（文件）→ Load Script（加载脚本），打开刚下载的脚本。找到该脚本并加载它之后，将会看到该脚本的内容出现在Script Editor（脚本编辑器）窗口的下半部分中，它们可能已经被突出显示了。

6. 单击脚本编辑器下半部分中的任何地方，以将光标放在其中，然后按快捷键Ctrl+Enter执行该脚本。

执行该脚本后，会在视口中看到一个扭曲的球体，如图0.2所示。但是，在Script Editor（脚本编辑器）窗口上半部分中也会看到一些有趣的信息。除了刚执行的脚本，还应会看到一行与以下信息类似的信息：

```
Execution time: 0.53 seconds.
```



■ 图 0.2　一个应用了噪声脚本的多边形球体

最后这行显示了计算机创建这个球体，细分它，并向网格应用噪声所花的时间。在这个特定情况下，计算机用了0.53s，基于你计算机的速度，结果可能稍微不同。

现在，执行此脚本的MEL版本，看看它执行同一个操作要花多长时间。

1. 在Maya的脚本编辑器中，单击出现在窗口下半部分上方的MEL选项卡。

2. 从配套网站下载polyNoise.mel脚本并记下它的位置。

3. 在Script Editor（脚本编辑器）窗口中选择菜单选项File（文件）→Load Script（加载脚本），打开刚下载的脚本。加载该脚本之后，应会看到它的内容出现在Script Editor（脚本编辑器）窗口的下半部分中，它们可能已经被突出显示了。

4. 单击脚本编辑器下半部分中的任何地方，以将光标放在其中，然后按快捷键Ctrl+Enter执行该脚本。

你应该很快会注意到一个问题——或者我们应该说很慢。Maya将在执行MEL脚本后停止响应。Mac用户将看到臭名昭著的“水皮球”加载光标。但不用担心！ Maya并没有崩溃。它（最终）将创建这个球体。你可能希望喝一两杯咖啡，然后再回来检查一下Maya。经过很长时间后，Maya最终创建了一个具有噪声变形的球体，并在Script Editor（脚本编辑器）窗口中上半部分打印类似下面这样的一行信息。

```
Execution time: 203.87 seconds.
```

因为Python脚本和MEL脚本都执行完全相同的任务，所以理论上讲可使用任意一种语言来完成此工作。但是很显然，你会希望使用Python版本。如果我们创建了一个执行此任务的MEL脚本，它在生产中将没什么用。没有美工人员愿意使用需要花几分钟来执行的脚本。它使美工人员不愿意尝试该工具，甚至很可能抛弃它。

只有一点不足：Python脚本是使用Maya Python API创建的。如果并列对比两段脚本，将会看到Python脚本比MEL脚本稍微复杂一点。使用API比使用Maya命令更复杂，但MEL无法直接使用API。MEL的这个不足是本例中性能差异如此之大的主要原因。

关于这段Python示例脚本，需要指出的另一个问题是，它不完整。因为它使用API调用来修改对象，所以如果不喜欢结果，无法像MEL那样使用撤销功能。尽管Python肯定可像MEL那样使用Maya命令（并自动获得同样的撤销支持），但我们在本例中使用了API，因为这对获得显著的速度提升是必不可少的。如果使用Python来调用Maya命令，Python脚本可能会像MEL脚本一样慢。

我们需要将此脚本转换为Python API插件，以保留撤销功能并仍使用API来修改对象，但在本例中出于简单考虑，我们未选择这么做。事实上，在许多时候你希望采用与此完全相同的做法来模拟一个能运行的版本，然后再添加最终的脚本来创建插件。不要担心——这并不难，它只是你需要采取的另一个步骤。幸运的是，本书的一个主要主题就是解释如何通过Python来使用API，所以你将能够轻松地创建快如闪电的工具。对Maya中的Python的工作原理有了更深的了解后，可以编写其他处理网格的脚本。可以看到，在包含大量顶点的对象中MEL运行得非常慢，所以这为在生产中引入新工具提供了机会。

使用Python API的另一个方便的优势是，它和C++ API使用同样的类。如果确实需要更快的速度，可轻松地将Python脚本转换为C++插件。如果不懂C++，可将该脚本作为模板传递给你工作室的程序员，因为API类在两种语言中完全相同。C++程序员也会发现Python API是很有用的，因为他们可在Maya中交互式地访问Python API来测试代码。另一方面，如果使用C++，肯定必须编译一个插件来测试每行代码。甚至可将Maya API与使用Maya命令的Python脚本混合使用。

对所有这些功能心中有数之后，你肯定已非常激动地想要开始学习如何在Maya中使用Python了。学完本书后，你应能够创建比本章的示例更复杂的脚本。所以不要再迟疑，立即开始学习吧！

目 录