[ModLoader1\_v1.0.0]源码帮助文档

注：如果是来看如何制作模组的话请确保有一定的编程基础。

由于我不是专门搞这些的所以会做成一坨，如果这样也能接受的话就看下去吧。

前言

如果是来做桌宠模组的话主要看[如何制作一个模组](#目录_如何制作一个模组)和[桌宠前置模组](#项目基本介绍_桌宠前置模组)的详细文档。

**目录**

[文件清单 2](#_Toc30034)

[关于[ModLoader1\_v1.0.0] 2](#_Toc3239)

[各个项目的基本介绍 3](#_Toc6925)

[模组加载的大致流程 4](#_Toc32155)

[如何制作一个模组 4](#_Toc19431)

[1． 模组的文件结构 4](#_Toc24977)

[2． 如何为模组添加可执行内容 5](#_Toc14231)

[3． 如何继承公用类中的模组接口 6](#_Toc26214)

[项目生成建议 6](#_Toc9604)

文件清单

该文档同一目录下一共有：5份帮助文档、4个源码文件夹。

|  |  |
| --- | --- |
| 文件夹 | 介绍 |
| ModLoader1 | 一共1个项目、1个解决方案。包含主程序项目和解决方案。  该解决方案包含了全部项目(一共8个项目)。 |
| ModLoaderLibrary1 | 一共1个项目。包含ModLoader1中主程序使用的库。 |
| Mods1 | 一共5个项目。包含所有模组项目的文件夹(一共5个模组)。 |
| PublicLibrary1 | 一共1个项目。包含主程序和模组共同使用的公用库。 |

关于[ModLoader1\_v1.0.0]

|  |  |
| --- | --- |
| 名称 | ModLoader1 |
| 版本 | v1.0.0 |
| 语言 | 后端：C#  前端：WPF |
| 框架 | .NET Framework 4.7.2 |
| 工具 | Visual Studio Professional 2019 |
| 简单  介绍 | 该项目是单纯的一个模组加载器。主程序和模组之间由一个公用库(PublicLibrary1)作为主程序和模组之间的交互，不过主程序和模组之间并没有实际的交互内容，模组在被加载后该干什么并没有要求。在主程序中只需处理加载模组和弹窗功能的实现。 |

各个项目的基本介绍

解决方案文件位置在”ModLoader1\ModLoader1.sln”，一共包含8个项目。

|  |  |
| --- | --- |
| 解决方案内项目介绍 | |
| ModLoader1 | |
| 介绍 | 主程序，主程序的东西很少没有特别需要讲的东西。主要放了控件的样式资源和实现了公用库的接口，其中实际加载模组的代码也在主程序使用的库中(ModLoaderLibrary1) |
| ModLoaderLibrary1 | |
| 介绍 | 主程序使用的库。主要内容：加载模组的实现、模组管理器的内容、消息框的实现 |
| PublicLibrary1 | |
| 介绍 | 公用库。主要内容：程序和模组使用的接口、模组对象和模组配置类、其它工具类。一共3个接口：程序接口、模组接口、消息框接口。消息接口主要用于弹出消息窗口。详细见”项目\_PublicLibrary1”文档 |
| desktopPet1\_Front | |
| 介绍 | 桌宠前置模组，详细见”项目\_desktopPet1\_Front”文档 |
| desktopPet1 | |
| 介绍 | 桌宠管理器模组，要做桌宠的话看桌宠前置的文档就行了 |
| desktopPet1\_expansion\_ldnn | |
| 介绍 | 绫地宁宁桌宠模组，当时觉得随便写一下就好了，结果里面乱成一团不想再看了 |
| ldnn\_expansion | |
| 介绍 | 绫地宁宁桌宠右键扩展模组，没啥好写的。为了实现这个特意给绫地宁宁桌宠模组直接用危险的方式开了个口 |
| desktopPet1\_minecraft\_Inventory | |
| 介绍 | 桌面物品栏模组。不想写文档了！ |

模组加载的大致流程

[公用库](#项目基本介绍_共用库)内有一个主程序的接口和一个模组的接口。主程序继承接口后由主程序实现接口中的加载模组方法，大致实现方式为：根据模组文件创建模组对象(共用库中的*ModObject*)，然后根据对象内的数据将模组的程序集加载到程序，并创建模组入口类的对象，若创建的对象继承了模组的接口，那么就调用模组接口中的初始化方法，以此实现加载模组。

如何制作一个模组

可以先看文档”附加\_模组制作过程简易版”，能够大致了解一遍。

全部项目的输出目录在”ModLoader1\ModLoader1\bin\Debug”，可以当成下面说的程序运行目录。存放模组的文件夹在程序运行目录下方的”File1\mods”，该目录下每一个文件夹表示一个模组。在新建一个模组时为了方便可以直接复制已有的模组然后进行修改。

1. 模组的文件结构

若文件未介绍说明该文件属于模组自身的文件。在一个模组的文件夹下会包含一个配置目录(config)和一个资源目录(Resources)。配置目录下的文件介绍如下：

|  |  |
| --- | --- |
| 文件 | 介绍 |
| config.json | 该文件是模组必须的。这是模组的配置文件，配置内容详细见文档”项目\_PublicLibrary1”。 |
| modInfor.json | 该文件非必须。该文件是在模组管理器内显示的内容。 |

资源目录下的文件”modIco.png”是在模组管理器内显示的图标，该文件非必须。

以上就是一个模组会用到的全部文件，若需要新建一个模组可以复制已有的模组然后保留以上介绍的内容，其它内容可以删除。

1. 如何为模组添加可执行内容

如果已经根据上面讲的新建了一个什么内容都没有模组，那么接下来将为你的模组添加可执行内容。

根据前面的[[关于[ModLoader1\_v1.0.0]](#目录_关于ModLoader1_v1_0_0)]的环境创建一个类库。为了方便调试，也可以打开解决方案，移除解决方案中的其它模组项目(如果你要使用它们那就保留)，然后在解决方案下新建类库。

在模组配置文件中有记录模组dll(上文新建的类库)文件位置的属性和记录dll入口类的属性，如果配置中只写了dll文件位置，那在加载模组时只会加载dll的程序集而不会进行其它操作，[[桌宠前置模组](#项目基本介绍_桌宠前置模组)]就是这种情况。

如果在配置中写了dll入口类的位置，那入口类将被创建，若该类继承了共用库中的模组接口(继承模组接口的方法下面会讲)，那接口中的[ModInitialize]方法会在每次重新加载模组完成后调用，并将主程序的接口作为参数传入，而接口中的[Dispose]方法会在模组重新加载前调用。

类库编译后的dll文件可以放在你创建的模组文件夹下的任意地方，也可以直接将文件的输出路径直接改到模组文件夹下面，然后确保配置文件里写的dll文件位置和实际文件位置相同即可。

然后就没了，你要在模组里写啥都可以。

1. 如何继承公用类中的模组接口

直接添加[共用库](#项目基本介绍_共用库)的项目引用，或者引用已经编译好的共用库的dll文件，在添加引用后把引用的属性[复制本地]改为[False]。然后让类继承[PublicLibrary1.Mod.Apis.IMod1]即可。

项目生成建议

主程序、主程序库、公用库，之间的引用是*项目引用*，而其它5个模组项目是使用已经生成后的dll文件的引用。

如果公用库出现更改则先编译公用库。如果要编译的模组有前置模组并且前置模组有更改，那建议先编译前置模组。不过如果你的项目中添加的引用都是*项目引用*那应该不用管这个，编译器应该就会按顺序编译了吧。