**手游项目数据  
版本号：Ver2.0**

# 版本说明：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本号** | **日期** | **內容** | **作者** | **审核人** |
| **Ver1.0** | 2016-03-04 | 初稿 | 邓世伟 |  |
| **Ver2.0** | 2016-03-21 | 文档补全 | 邓世伟 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 目录：

[版本说明： 2](#_Toc21204)

[目录： 3](#_Toc31622)

[1. 数据缓存 4](#_Toc17883)

[1.1 解析后端返回数据 4](#_Toc28174)

[1.2 数据缓存 4](#_Toc8848)

[1.3 消息通知 4](#_Toc30115)

[1.4 数据获取 4](#_Toc722)

[2. 生成数据结构类工具 4](#_Toc32641)

[2.1 解析\*.md 4](#_Toc13940)

[2.2 解析\*.php 5](#_Toc32138)

[2.3 分析组装 5](#_Toc19367)

[2.4 生成类 5](#_Toc416)

[3. 数据缓存代码设计 5](#_Toc8052)

[3.1 数据解析类 (NetworkDataParser.cs) 5](#_Toc25773)

[3.2 数据缓存类 (TableDataManager.cs) 5](#_Toc24343)

[3.3 结构体类 (AbsTableDataStruct.cs) 6](#_Toc24331)

[4. 类生成代码设计 6](#_Toc26952)

[4.1 生成类管理 (GenerateManager.cs) 6](#_Toc25910)

[4.2 生成数据结构 (GenerateStruct.cs) 6](#_Toc14000)

[4.3 读取md文件 (ReaderMD.cs) 6](#_Toc29410)

[4.4 读取php文件 (ReaderPhp.cs) 7](#_Toc27420)

[4.5 类结构生成 (ClassStructWindow.cs) 7](#_Toc32750)

[5. 类生成工具使用 7](#_Toc32043)

[5.1 显示界面 7](#_Toc30157)

[5.2 功能介绍 7](#_Toc4233)

[5.3 功能使用 8](#_Toc29021)

# 数据缓存

## 解析后端返回数据

1. 采用JsonDotNet插件，反序列化解析数据
2. 需要不同的结构类进行测试
3. 目前JsonDotNet插件不支持internal的反序列化

## 数据缓存

1. 采用dictionary的缓存池的方式来缓存
2. 用表名来当key值，进行缓存
3. 根据后端返回的msg内容，采用replace的方式
4. 根据后端返回update内容，采用add或update的方式
5. 根据后端返回delete内容，采用delelte的方式 (需要和后端确定)

## 消息通知

1. 采用delegate的方式来触发通知
2. 提供注册及移除的方式
3. 根据表内的内容有了修改或者添加或者删除进行派发消息通知

## 数据获取

1. 根据表名获取数据，返回T
2. 根据表名及Predicatet条件的形式获取数据，返回T
3. 根据表名获取数据，返回List<T>
4. 根据表名及Predicatet条件的形式获取数据，返回List<T>
5. 根据表名获取数据，返回Dictionary<string,T>
6. 根据表名及Predicatet条件的形式获取数据，返回Dictionary<string,T>

# 生成数据结构类工具

## 解析\*.md

1. 获取文件夹中的所有.md格式的所有文件
2. 解析md格式的文件，对其进行分解
3. 缓存分解的所有表结构数据

## 解析\*.php

1. 获取orm.config.php的文件
2. 解析orm.config.php的文件
3. 缓存orm.config里面的数据用于查找对应的表名

## 分析组装

1. 对md格式的文件进行逐行分析及分类
2. 缓存分析的表及结构数据
3. 对数据进行json拼装缓存

## 生成类

1. 通过缓存的表结构数据进行查找对应的表名
2. 通过缓存的表结构数据进行解析
3. 根据orm.config对应的表结构生成结构体类
4. 采用JsonCSharp插件来生成，目前不支持dictionary

# 数据缓存代码设计

## 数据解析类 (NetworkDataParser.cs)

根据后端返回的数据进行分类转换

1. 进行json的转换
2. 分类处理返回的结果
3. 管理服务器时间
4. 根据表格进行结构反系列化
5. 对数据进行add、remove、update的操作
6. 值类型转换

## 数据缓存类 (TableDataManager.cs)

管理所有表格结构体数据

1. 缓存表格对应的类结构体
2. 获取对应的表结构类
3. 管理当前Post的数据请求返回结构
4. 消息通知变更数据委托
5. 管理所有表格数据存取
6. Add、remove、update表格数据的处理
7. 条件查询

## 结构体类 (AbsTableDataStruct.cs)

管理表格结构体

1. 管理对应的数据表格
2. 管理表格变动的消息通知

# 类生成代码设计

## 生成类管理 (GenerateManager.cs)

管理读取及消息机制和查找表名处理

1. Md文件类结构生成
2. Md文件读取及解析分类
3. Php文件读取及数据缓存
4. 查找相应的表名及json系列化

## 生成数据结构 (GenerateStruct.cs)

管理生成数据结构

1. 生成数据结构
2. Json转换成类结构
3. 存在的文件备份

## 读取md文件 (ReaderMD.cs)

对md文件进行读取及解析分类

1. 读取并解析md文件
2. 逐行解析md文件
3. 表格字段解析读取缓存
4. 表格字段赋值
5. 内容json化

## 读取php文件 (ReaderPhp.cs)

对php文件进行读取及解析分类

1. 读取并解析php文件
2. 逐行解析md文件
3. 缓存表名数据

## 类结构生成 (ClassStructWindow.cs)

对md文件中查找到的表格数据进行数据结构类的生成

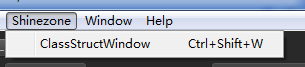
1. 界面绘制
2. 读取文件夹中的所有md文件及缓存数据
3. 生成所有md文件类结构
4. 选择单个md文件进行类结构生成
5. Log文件生成

# 类生成工具使用

## 显示界面

启动类生成界面，可通过以下2种方式

1. 通过菜单栏的Shinezone->ClassStructWindow，如图：

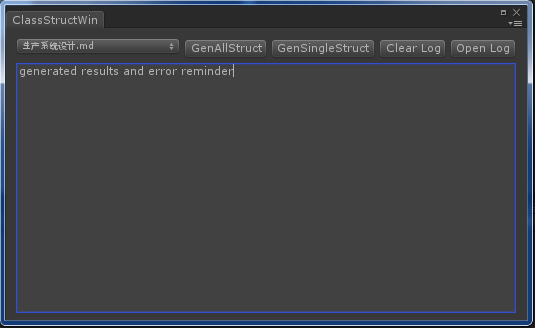


1. 通过快捷键 Ctrl+Shift+W 启动

## 功能介绍

具体功能介绍

1. 界面图：

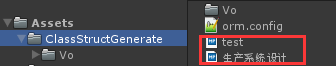


1. 罗列文件夹中的所有的md文件（选择列表）
2. 一次性生成所有md文件中的表结构类（GenAllStruct按钮）
3. 选择单个md进行表结构类生成（GenSingleStruct）
4. 清除产生的log（Clear Log）
5. 打开log文件夹（Open Log）
6. 记录对md文件进行生成数据结构类的过程日志（Text文本）

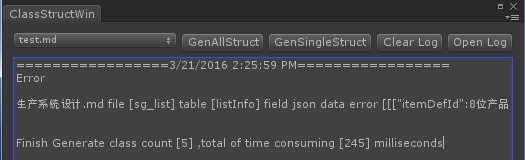
## 功能使用

具体的操作流程

1. 请在Assets目录下面新建一个叫ClassSturctGenerate的文件夹，该文件夹存放所有的md文件及生成的数据结构类文件



1. Ctrl+Shift+W 启动工具
2. 根据需求点击不同的按钮，生成的数据类结构在Vo目录下面。Vo目录下面会有一个叫old的文件夹，该文件夹中存放的历史生成的类结构文件
3. 生成信息提示会输出在Text文本当中及本地的genLog.txt文件中



1. 信息中带有Error字样的表示有部分md文件生成类结构时产生了错误，具体问题可以查看提示对相应的md文件进行修正后在重新生成