<My PC Logo>

软件架构文档

版本 <1.0>

**13组**

[要定制 Microsoft Word 中的自动字段（选中时显示灰色背景），请选择 File>Properties，然后将 Title、Subject 和 Company 等字段替换为此文档的相应信息。关闭该对话框后，通过选择 Edit>Select All（或 Ctrl-A）并按 F9，或只是在字段上单击并按 F9，可以在整个文档中更新自动字段。对于页眉和页脚，这一操作必须单独进行。按 Alt-F9，将在显示字段名称和字段内容之间切换。有关字段处理的详细信息，请参见 Word 帮助。]

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2020/10/30 | <1.0> | <初代架构文档>：用例视图、物理视图、逻辑视图 | 窦嘉伟、敖宇晨 |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 参考资料 4

2. 用例视图 4

2.1 概述 4

2.2 架构上的重要用例 4

2.2.1 Use-case 1 用户模块 4

2.2.2 Use-case 2 单机模式 4

2.2.3 Use-case 3 匹配模块 5

2.2.4 Use-case4 双人协作 5

2.2.5 Use-case5 双人对战 6

3. 逻辑视图 6

3.1 概述 6

3.2 逻辑视图的结构层次 6

3.3 在架构方面具有重要意义的设计包 7

3.3.1 应用层 7

3.3.2 业务服务层 8

3.3.3 中间层 9

4. 进程视图 9

5. 部署视图 9

6. 实现视图 9

7. 数据视图（可选） 9

8. 核心算法设计（可选） 9

软件架构文档

# 简介

## 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面作出的重要决策。

## 参考资料

1. My\_PCLogo项目迭代计划1.0 2020.09 本项目组

2. 项目软件需求规约 2020.09 本项目组

3. 项目数据库设计规范 2020.10 本项目组

4. 项目词汇表 2020.09 本项目组

5. 谷歌编程规范（试用于本项目） 2020.09 本项目组 （摘自《谷歌编程规范》）

6. 项目要求 2020.xx <软件工程原理与实践>课程要求

7. 课程教材《软件工程原理》 2013.xx 沈备军 陈昊鹏 陈雨亭编著

8. 用例模型 2020.10 本项目组

9. Vision文档2.0 2020.10 本项目组

# 用例视图

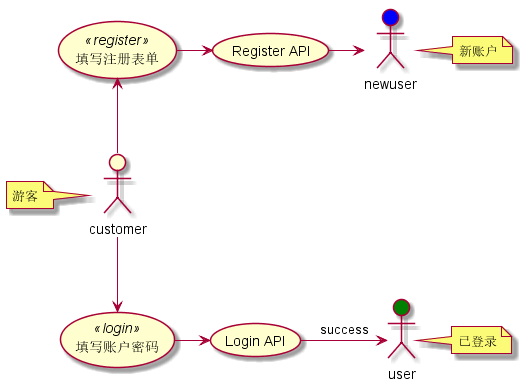
## 概述

本部分描述了架构设计中的用例视图，它描述了一组场景或用例，这些场景或用例强调或说明了架构设计中的一些特点。

## 架构上的重要用例

### Use-case 1 用户模块

游客用户可以通过填写注册信息，发送到后台注册接口，注册新账户。注册表单包含用户名，邮箱/电话号码，密码等信息。游客用户若已有账户，通过填写账号及密码，发送到后台登录接口，校验成功即可登录。



图表 1 use-case1 用户模块

### Use-case 2 单机模式

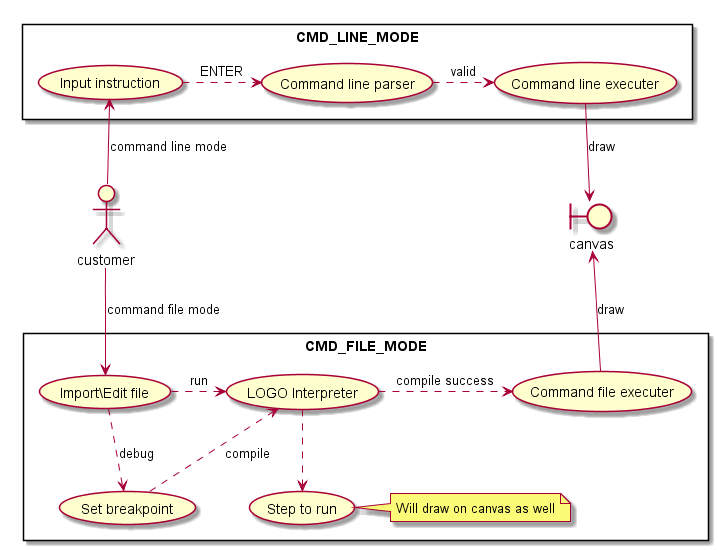
在单机模式下，用户可以有两种使用方法：

#### *命令行模式*

每一条指令输入后，经过解析器解析，确认合乎语法规范后，交付给执行器将结果绘制在画布上输入指令可以是直接键入命令行，或者使用自然语言通过语言方式（进阶需求）。

#### *命令文件模式*

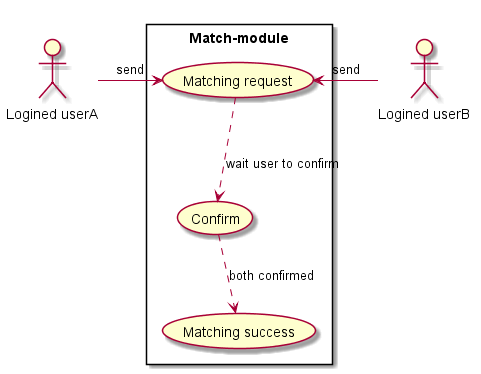
文件可以自行在编辑器上编写或者直接导入通过丰富的指令组合，解释器确认在合乎语法规范后，绘制出丰富多彩的图案也可设置断点进行调试，分步运行。



图表 2 use-case2 单机模式

### Use-case 3 匹配模块

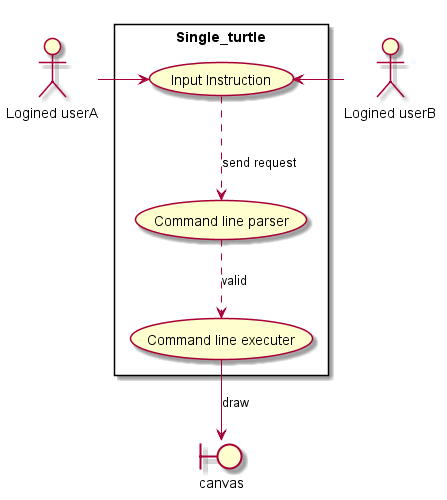
已登录用户发送匹配请求，由后台自动匹配。匹配到小伙伴后，双方需要在有限时间内（如30秒）确认。双方在时间限制内均确认后建立连接，即可进行联网游戏。



图表 3 use-case3 匹配模块

### Use-case4 双人协作

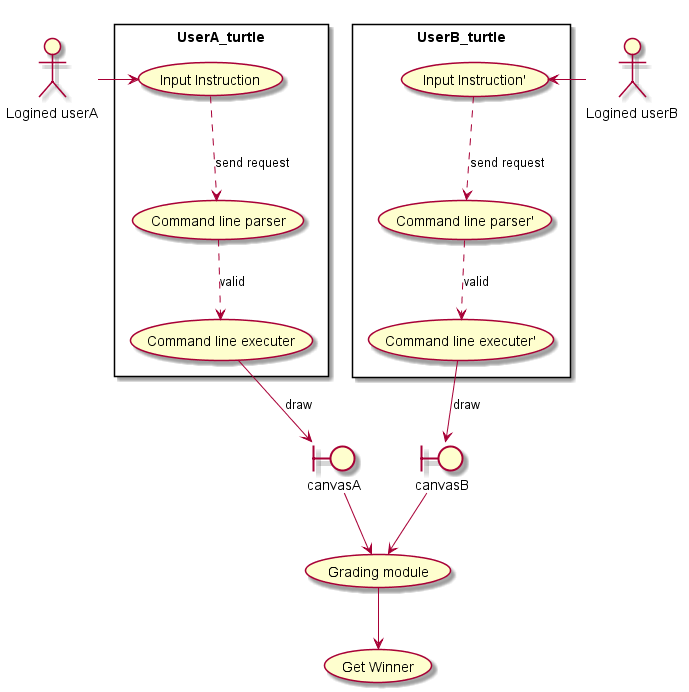
已经匹配的两名用户，通过轮流发送指令控制单个海龟绘图，每人需要在固定的操作时间限制内（如30秒）发送合乎语法的指令（在本地校验合法后才能发出），否则海龟的控制转移到另一用户手中。



图表 4 use-case4 双人协作模式

### Use-case5 双人对战

已经匹配的两名用户，各自发送指令控制自己的海龟绘图，在规定时间结束后，经由评分模块通过一定的评分机制决出胜者。



图表 5 use-case5 双人对战模式

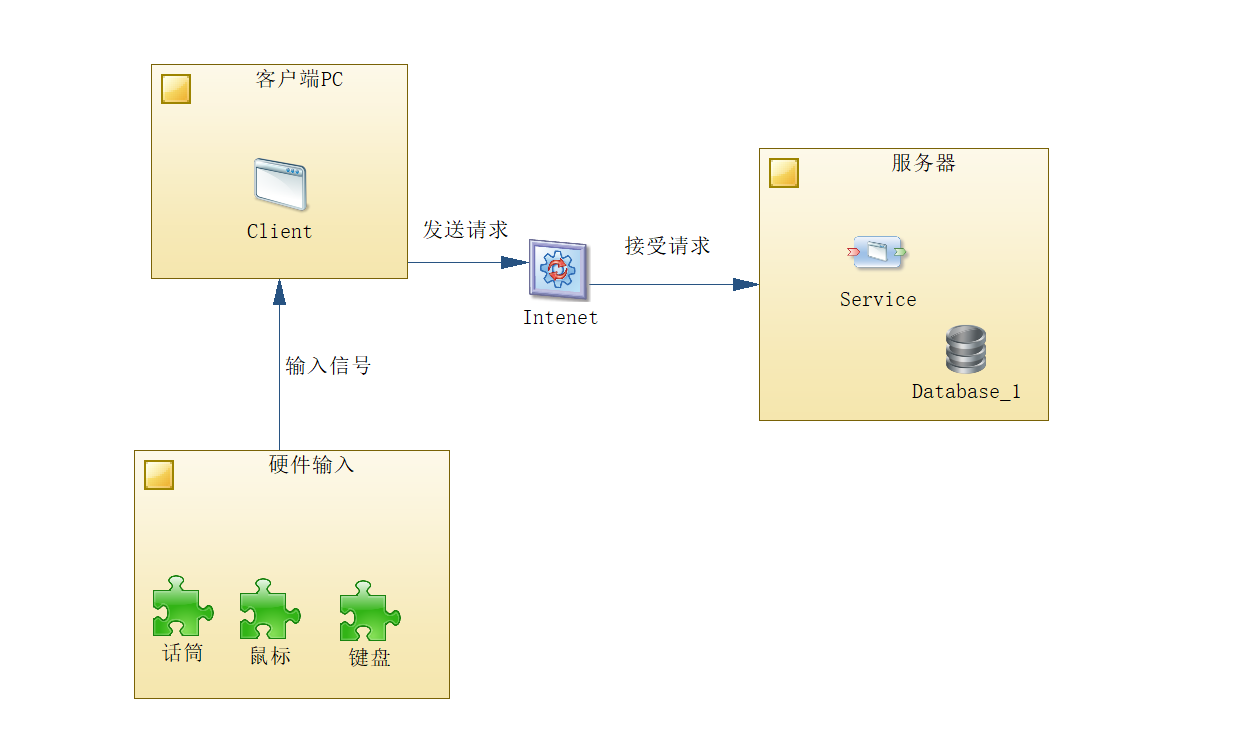
# 逻辑视图

## 概述

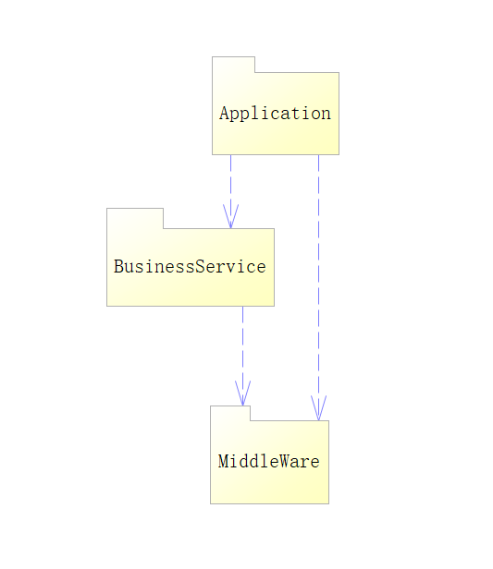
本部分描述了架构的逻辑视图，它描述了服务和功能在项目架构中的层次结构，此外，还描述了层次与包之间的关系。

## 逻辑视图的结构层次

本项目的逻辑视图由三个主要层次组成：应用层、业务服务层、中间层。



图表 6物理视图



图表 7逻辑视图的三个层次及其关系

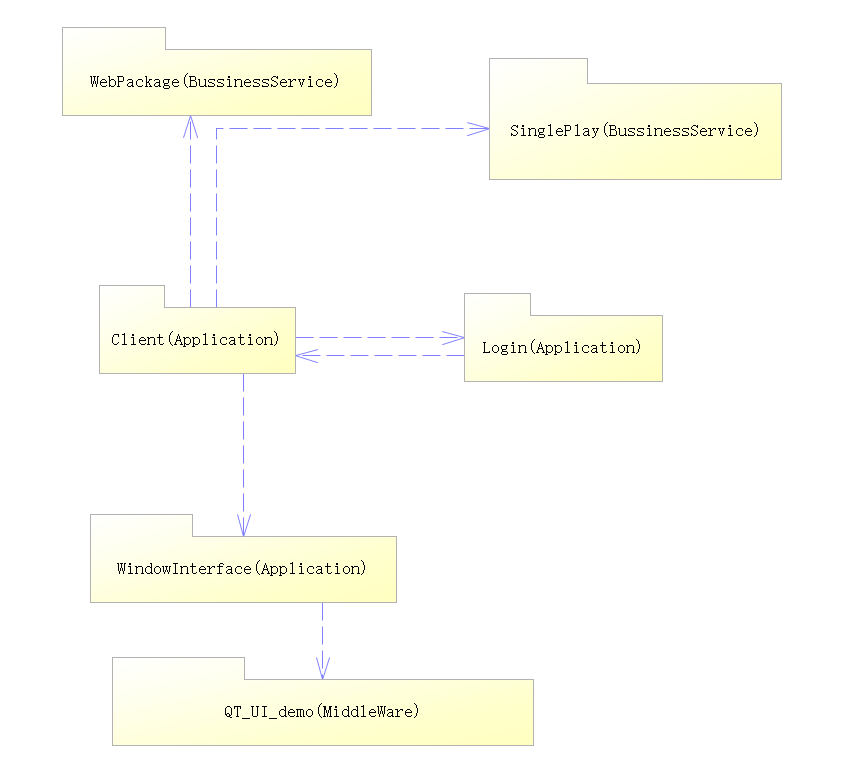
## 在架构方面具有重要意义的设计包

### 应用层

应用层包括与用户交互的边界类（Client和界面类）以及用户信息相关。这一层依赖于业务服务层和中间层。其中：

client包： 负责处理用户操作产生的各种请求。

windowinterface包： 包含用户连接的边界类，负责展示客户端界面。



图表 8应用层

### 业务服务层

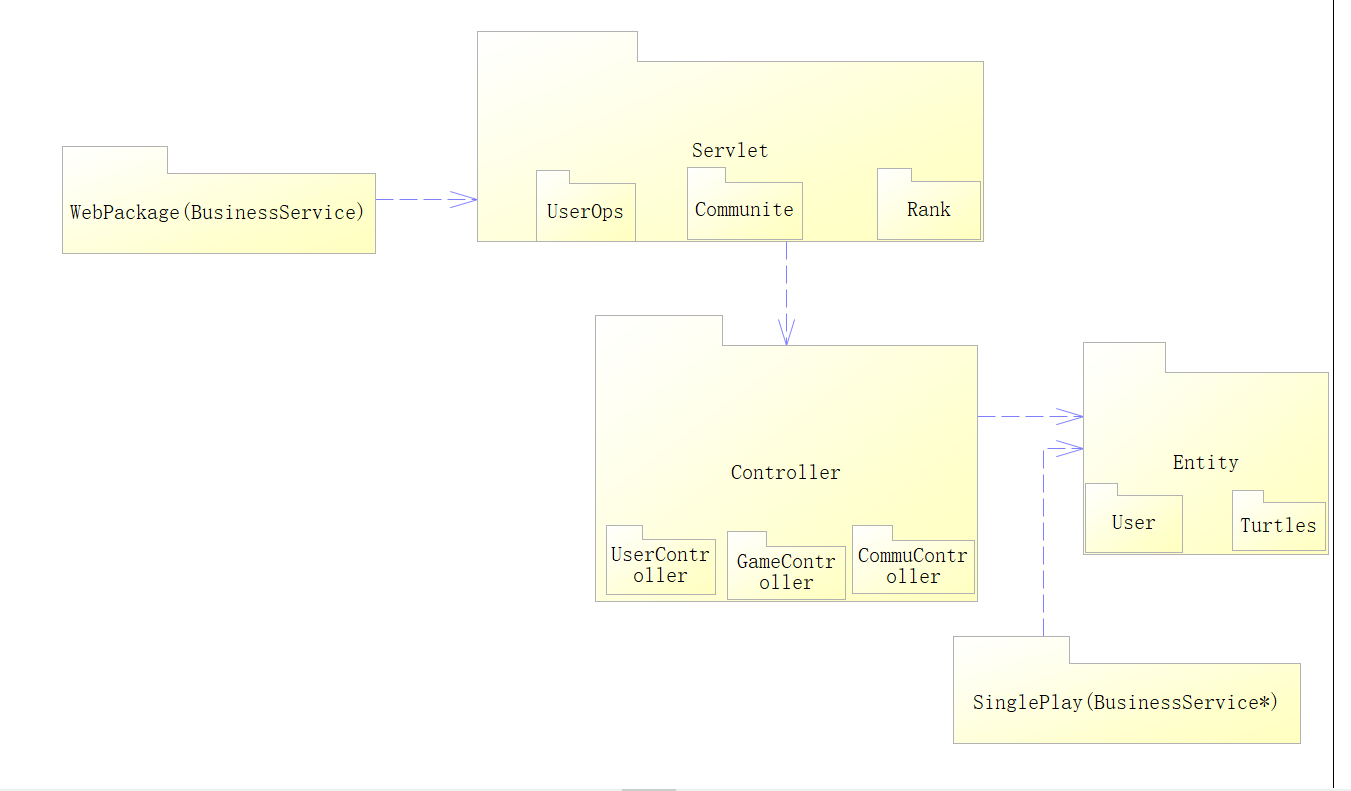
业务服务层包括控制类和web包以及后端spring与hibernate等包。还包扩一些entity类。这一层依赖于中间层。其中：

webpakage包： 处理client产生的请求。

servelet包： 处理请求并分类为用户操作、匹配、与交流三类。

controller包： 包含用户操作、用户互动以及游戏的控制类。

entity包： 存放用户数据的实体类。



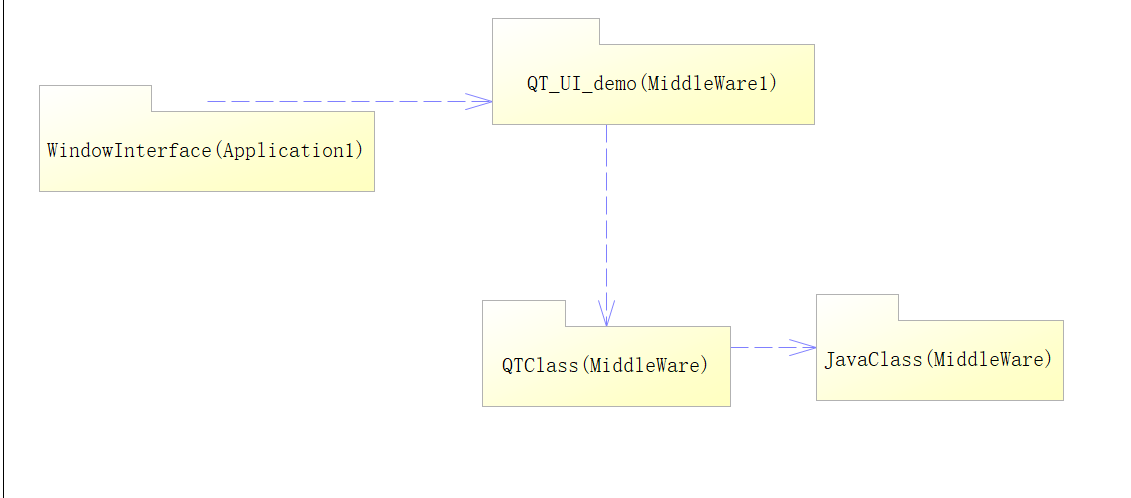
图表 9业务服务层

### 中间层

中间层包括部分底层代码块。其中：

QTclass包： 包含QTUI部分的原始类。

Java包： 包含Java部分原始类。



图表 10中间层

# 进程视图

[本节说明将系统分解为轻量级进程（单个控制线程）和重量级进程（成组的轻量级进程）的情况。本节的内容按照各个通信或交互的进程组来进行组织。说明进程之间的主要通信模式，例如消息传递、中断和会合。]

# 部署视图

[本节说明用来部署和运行该软件的一种或多种物理网络（硬件）配置。对于每种配置，它至少应该指出执行该软件的物理节点（计算机、CPU）及其互连情况（总线连接、LAN 连接、点到点连接等）。另外还要包括**进程视图**中的各进程到物理节点的映射。]

# 实现视图

[本节说明实现模型的整体结构、软件分解为实现模型中的层和子系统的情况，以及所有在构架方面具有重要意义的构件。]

# 数据视图（可选）

[从永久性数据存储方面来对系统进行说明。如果几乎或根本没有永久性数据，或者设计模型与数据模型之间的转换并不重要，那么本节就为可选。]

# 核心算法设计（可选）

[对系统中的核心算法进行设计。如果没有什么重要的算法，那么本节就为可选。]