

Java联网猜拳小游戏[内容预览]

2.1.4. 模块类定义-V	1. 框架类作用简要说明
2.1.5. 模块类定义-C	1.1. 客户端
2.1.6. 打开/关闭模块	1.1.1. com.frm.base (常用base类)
2.1.7. 关闭/重新打开界面 (保留ctrl+model)	1.1.2. com.frm.cfg (配置包)
2.1.8. 隐藏/显示界面	1.1.3. com.frm.global_observer (全局观察...)
2.1.9. 界面扩展参数	1.1.4. com.frm.mvc (mvc框架包)
2.1.10. 本地通过PID进行消息数据传递	1.1.5. com.frm.netc (客户端网络实现包)
2.1.11. Model例重数据逻辑	1.1.6. com.frm.proto (协议包)
2.2. 获取模块数据	1.1.7. 协议衍生包 (双端都有这两个包)
2.3. 客户端发协议	1.1.8. com.frm.thread (线程包)
2.3.1. sendPackMsg	1.1.9. com.frm.util_res (资源使用工具包)
2.4. 服务器端响应协议	1.1.10. com.frm.utils (工具包)
2.4.1. 原包响应指定 - response	1.1.11. com.frm.z_common_entity (复用实...)
2.4.2. 响应指定用户 - responseWithSpecifie...	1.1.12. com.frm.zserver_connect (连接服务...)
2.4.3. 响应指定用户数组 - responseWithSpe...	1.1.13. com.z.main (程序入口类)
2.4.4. 广播给所有用户	1.1.14. com.z.test (练习测试包)
3. 开发一个模块的流程	1.2. 服务端
4. 常用介绍	1.2.1. com.frm.cfg (配置包)
4.1. JDK常用	1.2.2. com.frm.db (数据库包)
4.1.1. GridBagLayout	1.2.3. com.frm.nets (服务端网络包)
4.1.2. 添加按钮事件	1.2.4. com.frm.utils (工具包)
4.1.3. 窗体各个时机监听接口	1.2.5. com.frm.z_game_hall (游戏大厅)
4.1.4. 界面始终在最顶层	1.2.6. com.frm.z_game_match (游戏匹配)
4.1.5. 设置界面关闭的触发类型	1.2.7. com.frm.z_game_room_mgr (游戏房...)
4.1.6. 设置标题	1.2.8. com.frm.z_game_room_mgr.sub (游...)
4.2. 框架生命周期	1.2.9. com.frm.zserver_start (启动服务器模...)
4.2.1. View生命周期函数	2. 创建一个模块的流程
4.2.2. Model的生命周期函数	2.1. 双端通用流程
4.2.3. Controller的生命周期函数	2.1.1. 模块包定义
4.3. 框架自定义布局	2.1.2. 实体包定义
4.3.1. 默认是居中随机偏移	2.1.3. 模块类定义-M
4.3.2. 让界面靠在四个角	2.1.4. 模块类定义-V
4.3.3. 进行自定义操作	

02_API及应用框架说明.docx

文档

Java联网猜拳小游戏

过往案例展示

demo实现前置了解

MVC架构

设计模式

UI编辑环境

demo实现

系统分析与设计

框架引入及基本介绍

前后端实现-现场版

(过程中伴随AI的使用)



脆皮鸡写代码

获取最新分享 + 源码等源资料

微信扫一扫二维码, 关注我的公众号

获取源码|最新分享:

-2051享...

+Java联...

@GD-低...

综合栏目

源码获取

视频分享

回复JDK获取同款环境

通过网盘分享的文件: Eclipse+JDK链接: <https://pan.baidu.com/s/1FQGEmsW13IYxTNULhlamQ?pwd=p6kk> 提取码: p6kk --来自百度网盘超级会员的分享

Java联网猜拳小游戏

关键流程块解析+demo实现

分享目标：我们学习了知识点后，总是希望能基于知识点产出一点看得见的小成果，而不仅仅是止步于学知识点。

因此本次分享将提供一个纯Java的可复用可扩展的框架，并基于框架实现一个有界面能联网进行猜拳的小游戏demo。

希望本次分享大家都能有一定的收获。

适合人群

- 有一点Java基础还没做出，但想要做出一点实际成果的同学
- 想要完成大学Java课程作业但没思路的同学
- 想要基于Java开发一些小工具/软件/游戏的同学

分享大纲

- 过往案例展示

- Demo实现前提:

 - MVC架构

 - 用到的设计模式

 - UI编辑环境

- Demo实现(现有部分演示):

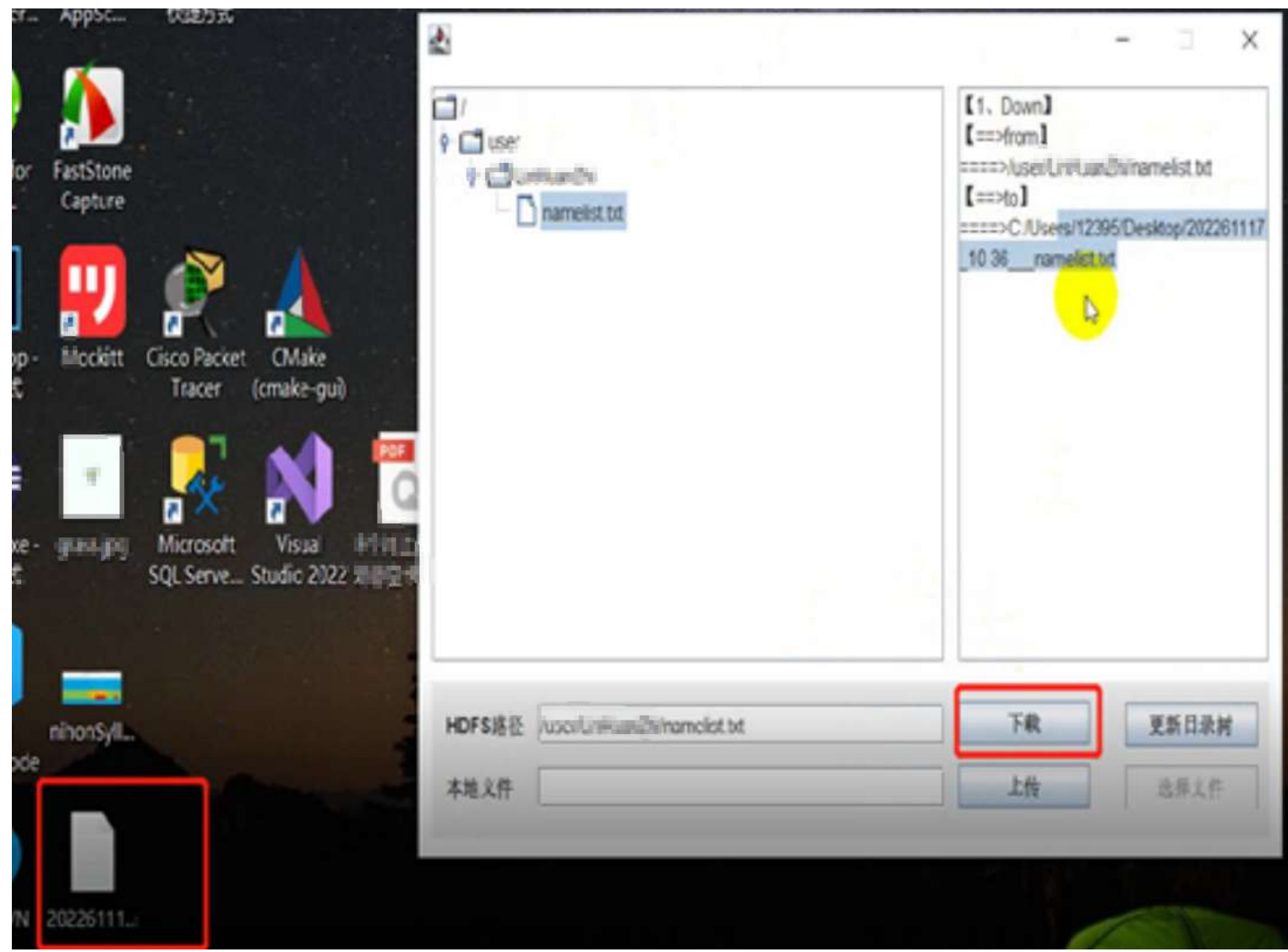
 - 系统分析与设计

 - 框架引入及基本介绍

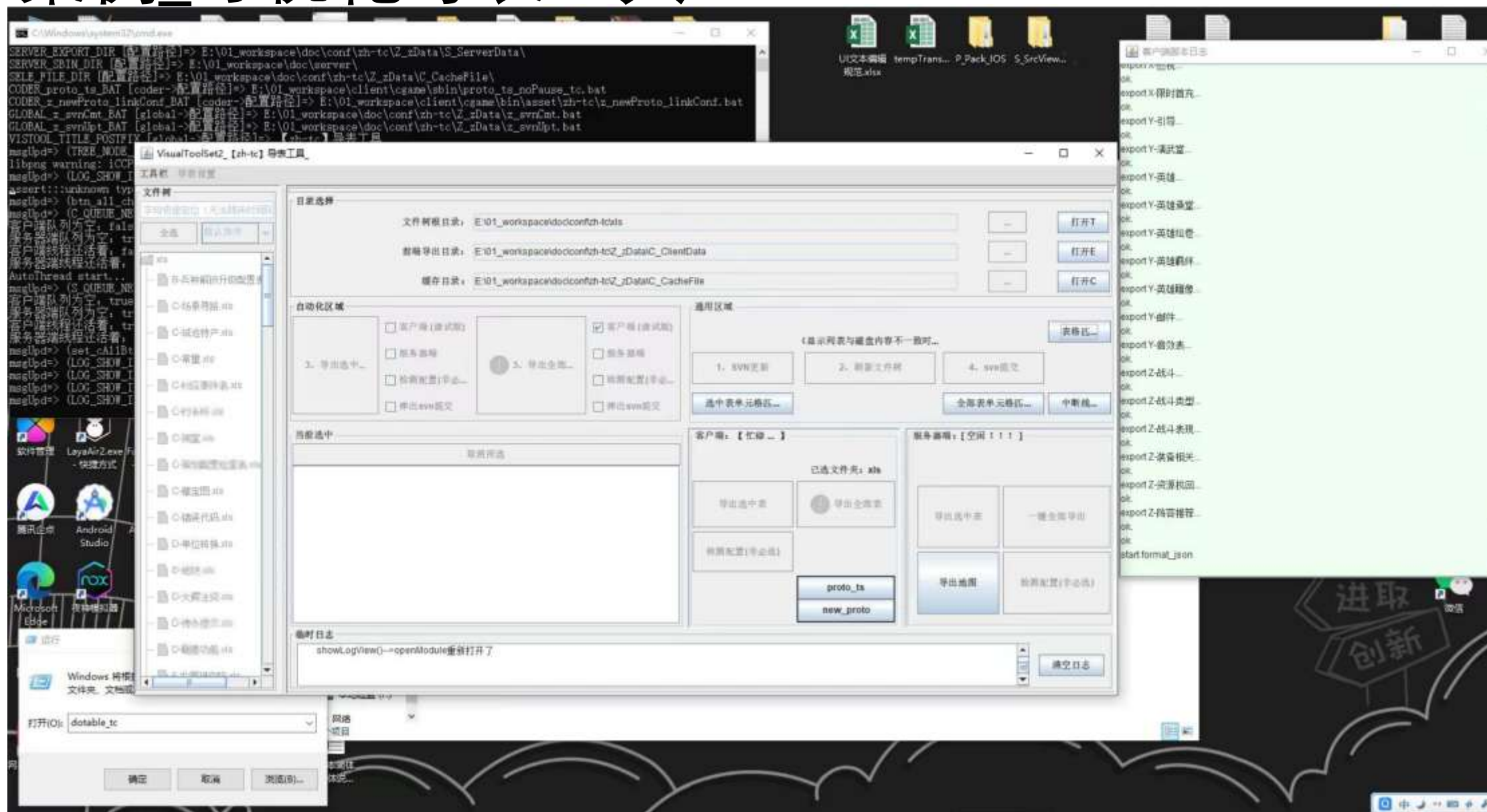
 - 前后端实现-现场版

 - (过程中伴随AI的使用)

过往案例_Hadoop文件浏览系统



过往案例_可视化导表工具



过往案例_多人聊天系统



登录演示



过往案例_答题系统



分享大纲

- 过往案例展示

- Demo实现前提：

 - MVC架构

 - 用到的设计模式

 - UI编辑环境

- Demo实现(现有部分演示)：

 - 系统分析与设计

 - 框架引入及基本介绍

 - 前后端实现

 - (过程中伴随AI的使用)

Demo_联网猜拳小游戏_原型图

猜拳小游戏

你的用户名:

服务器地址:

连接服务器

等待房主开始游戏

玩家A

玩家B

开始游戏

Demo_联网猜拳小游戏_原型图

第1轮:

0/0

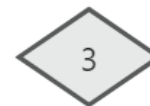
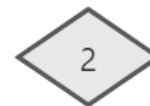
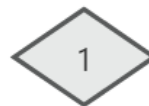
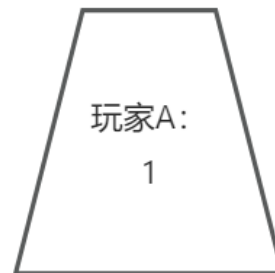
3...2...1...



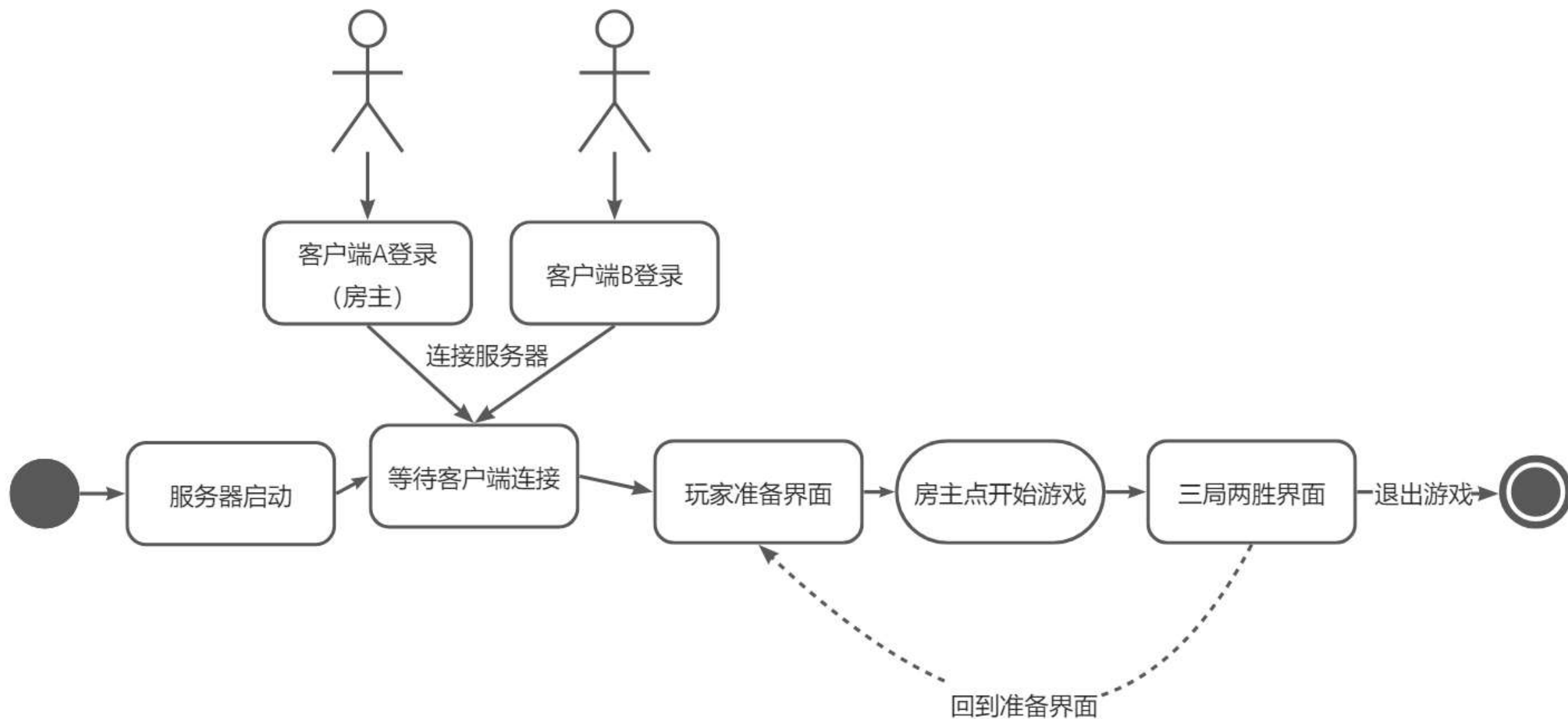
第1轮:

0/1

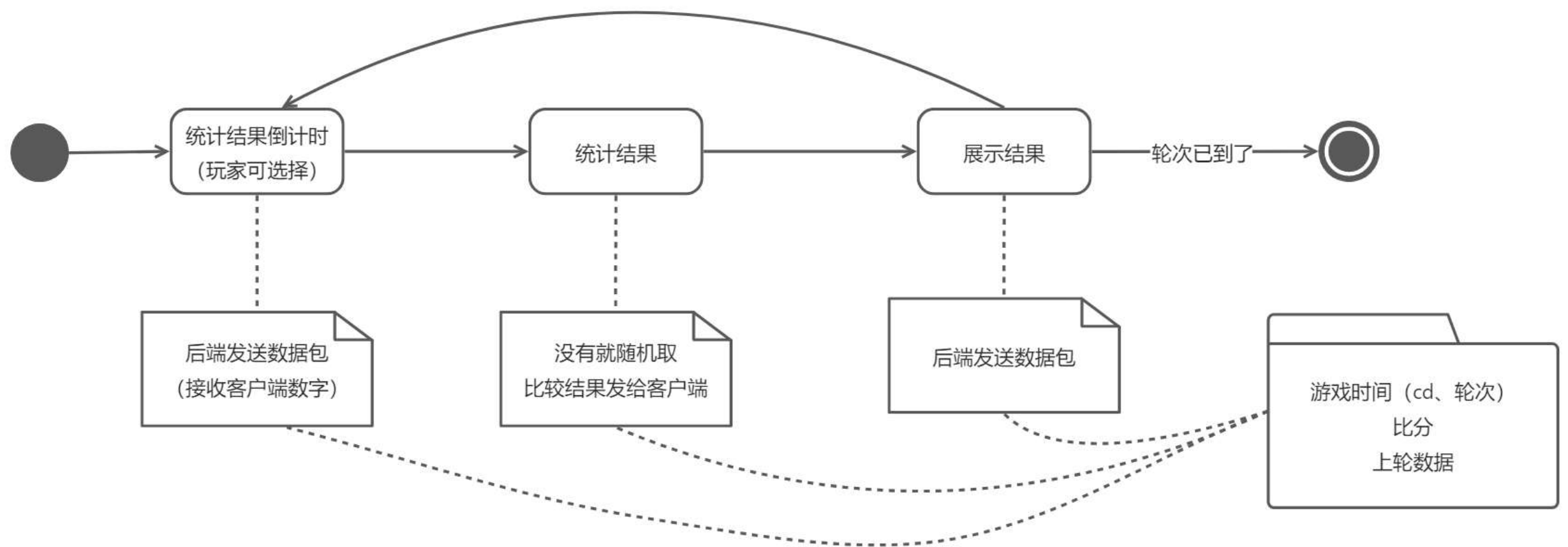
3...2...1...



Demo_联网猜拳小游戏_流程图



Demo_联网猜拳小游戏_流程图



Handwritten notes and a timeline illustrating game mechanics:

Timeline: 0 1 2 3 4 5 6 7 8 9 12 15 16 17 18

Annotations:

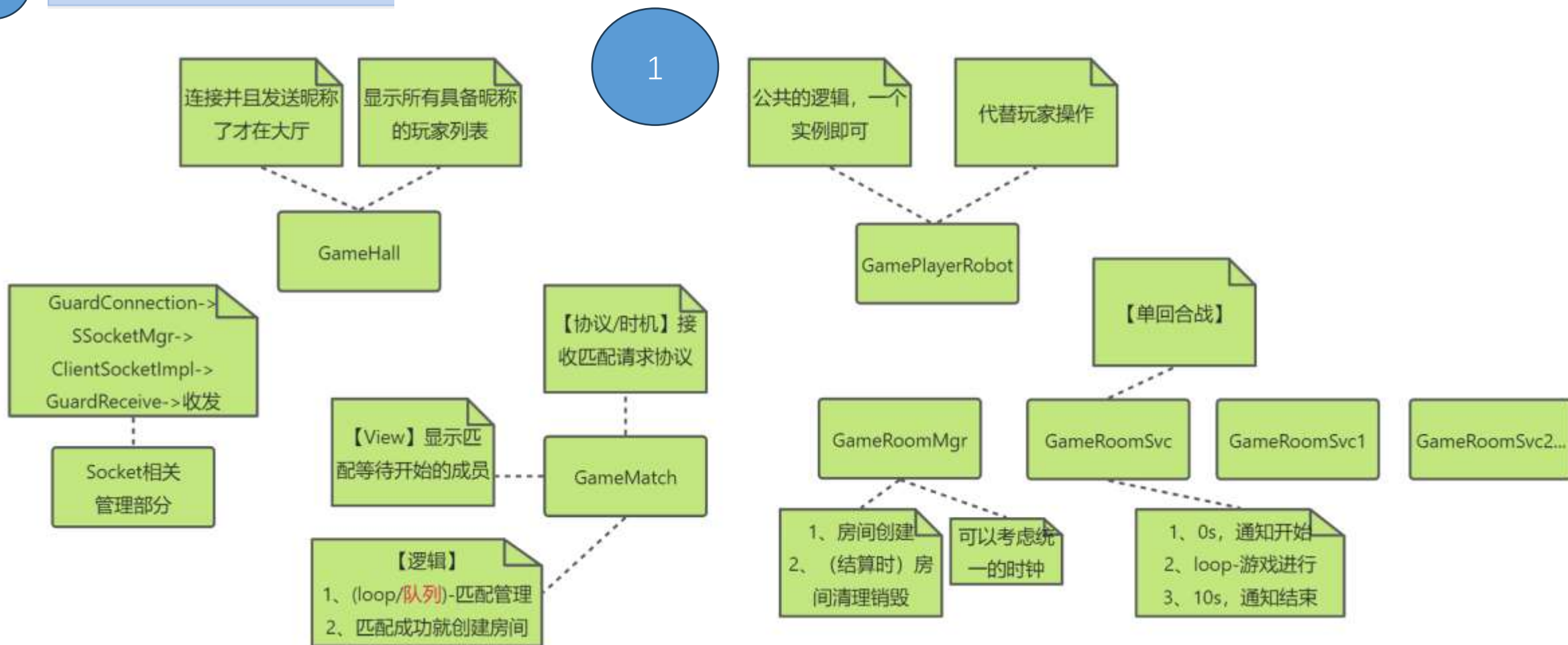
- Green: $(\%3) \& (\%640)$
- Red: $\%6 = 0$
- Red arrows point to positions 6, 12, and 18 on the timeline.
- Bottom right calculations:
 - $16 \% 3 = 1 \rightarrow 2$
 - $17 \% 3 = 2 \rightarrow 1$
 - $18 \% 3 = 0 \rightarrow 0$
- Red text: 展示倒计时

计划有时是会变的(为了完善后端框架及文档)

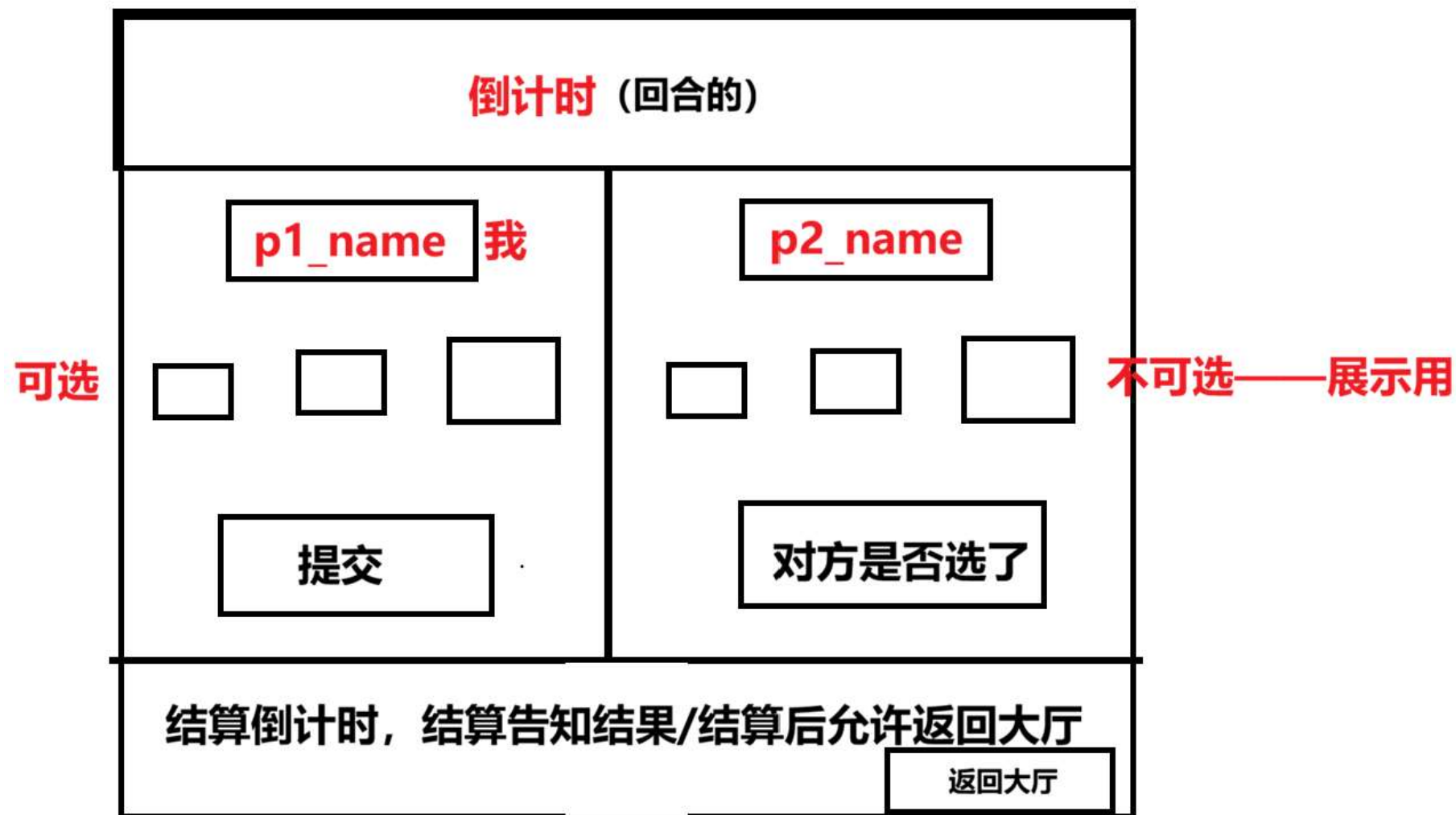
2

3. 开发一个模块的流程

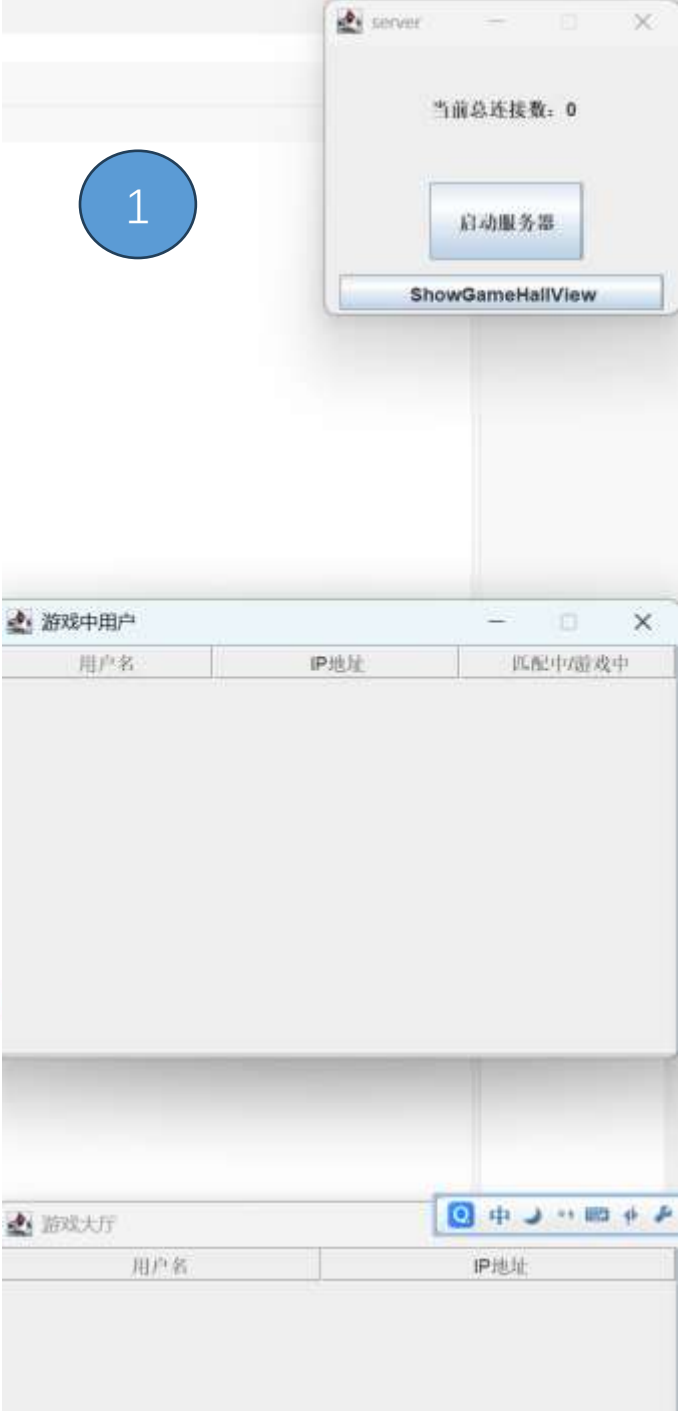
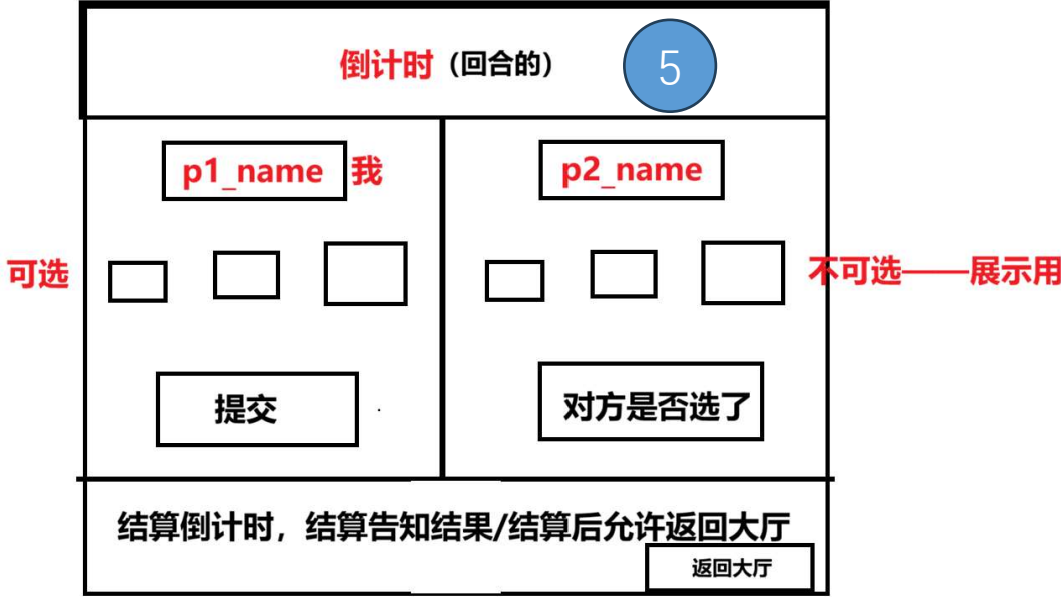
1



需求有时是会变的（简化为单回合）



最终决定



分享大纲

- 过往案例展示

- Demo实现前提：

- MVC架构 ②

- 用到的设计模式

- UI编辑环境

①

- Demo实现(现有部分演示)：

- 系统分析与设计

- 框架引入及基本介绍

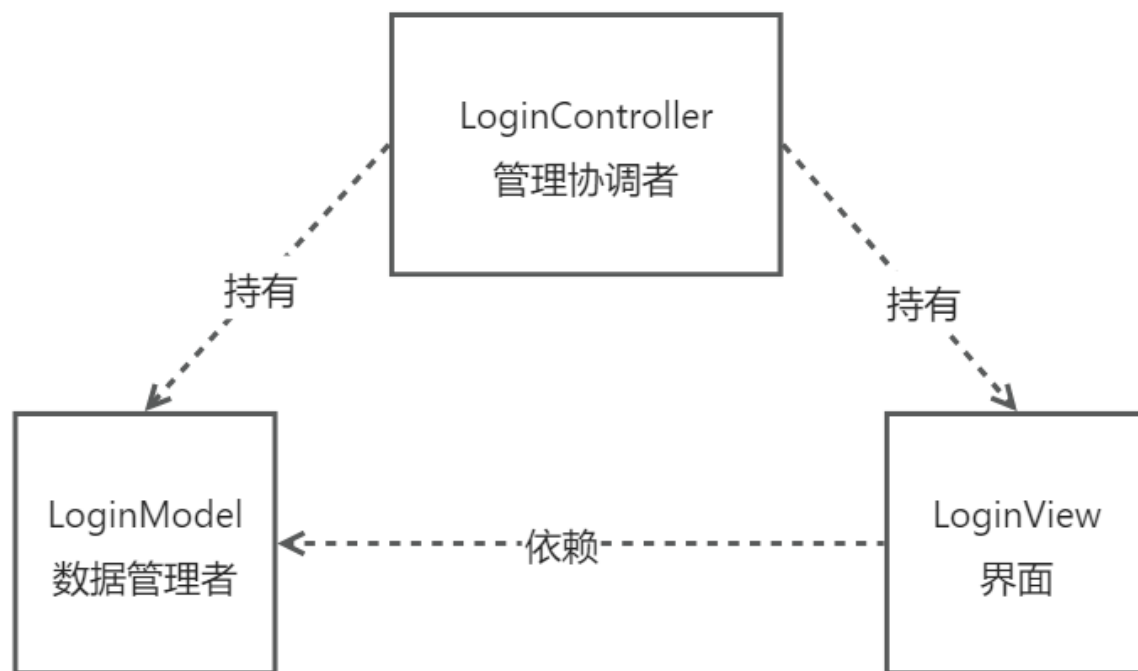
- 前后端实现-现场版

- (过程中伴随AI的使用)

MVC架构

Model-View-Controller (MVC) 是一种软件架构模式，是软件设计模式的体现 [8-9]，用于组织代码并分离关注点，广泛应用于用户界面的开发中。该模式将相关程序逻辑划分为三个相互关联的组成部分：模型、视图和控制器 [10]。MVC模式最初用于桌面图形用户界面 (Graph User Interface, GUI)，但后来因设计Web应用程序而变得流行。许多流行的编程语言都有MVC框架，这些框架促进了MVC模式的实现。

注意：此处只是举个例子，仅仅是本框架设计如此，MVC本来就只是一个参考。



分享大纲

- 过往案例展示

- Demo实现前提：

- MVC架构

- 用到的设计模式

- UI编辑环境

- Demo实现：

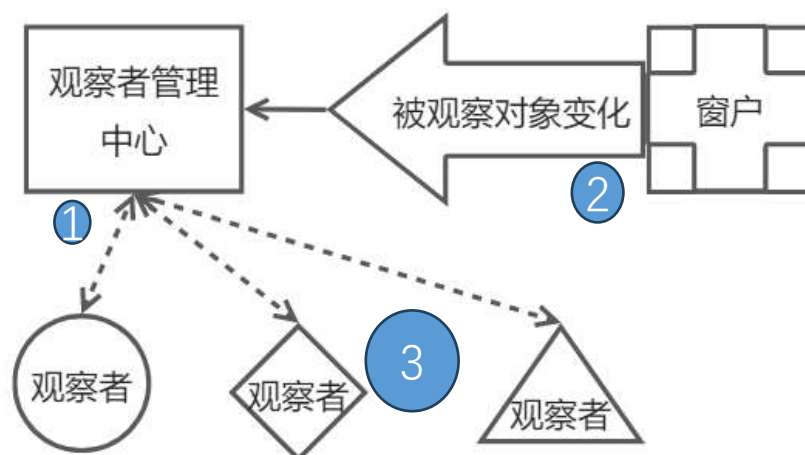
- 系统分析与设计

- 框架引入及基本介绍

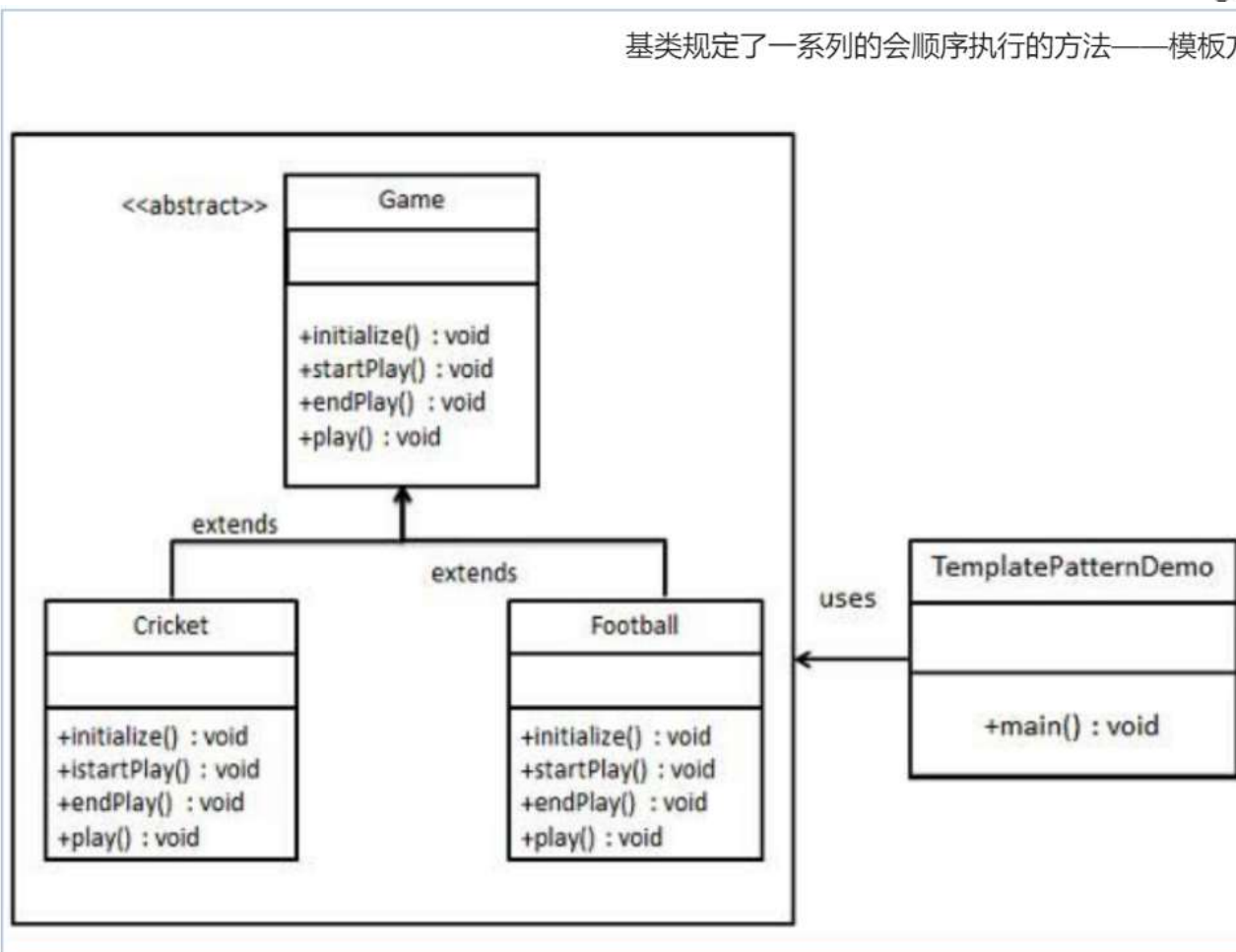
- 前后端实现-现场版

- (过程中伴随AI的使用)

用到的设计模式



基类规定了一系列的会顺序执行的方法——模板方法



```
public abstract class Game {
    abstract void initialize();
    abstract void startPlay();
    abstract void endPlay();

    //模板
    public final void play(){

        //初始化游戏
        initialize();

        //开始游戏
        startPlay();

        //结束游戏
        endPlay();
    }
}
```

设计/约定一个类只实例化一次，即为单例——整个应用只会产生单个实例。

分享大纲

- 过往案例展示

- Demo实现前提：

- MVC架构

- 用到的设计模式

- UI编辑环境

- Demo实现：

- 系统分析与设计

- 框架引入及基本介绍

- 前后端实现-现场版

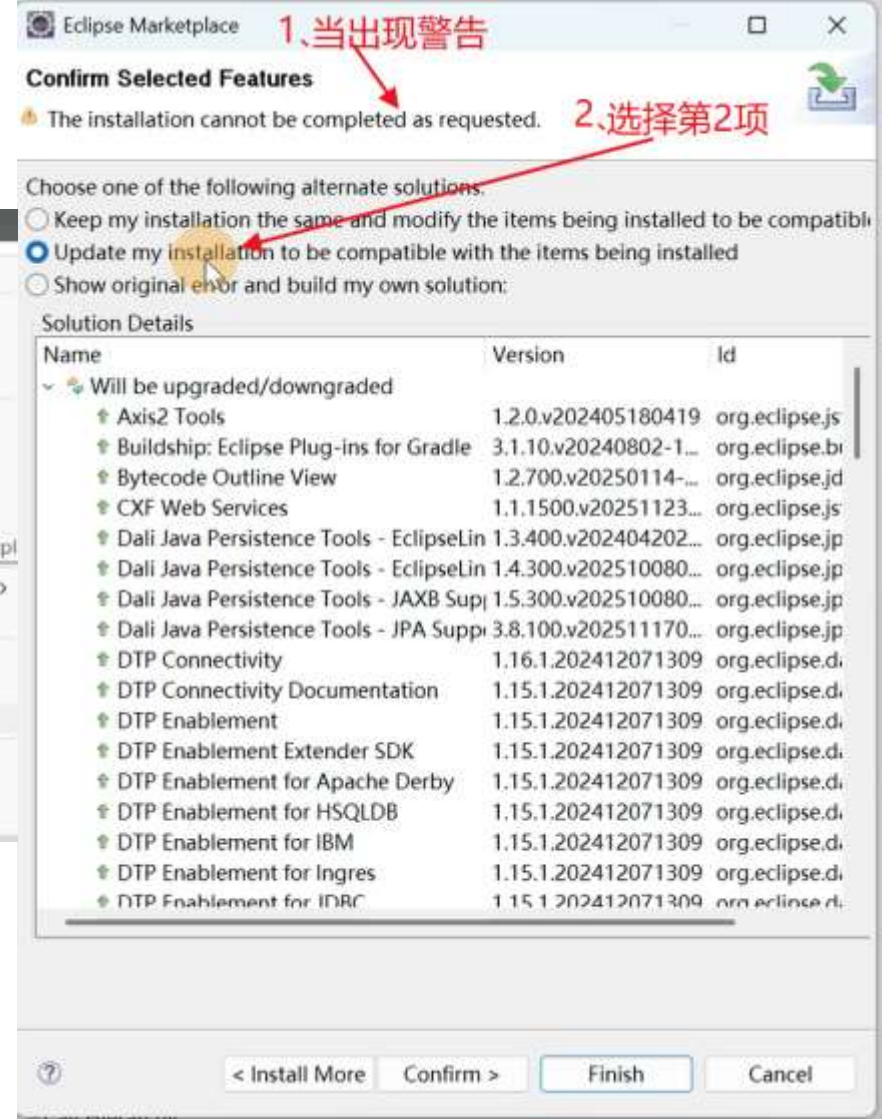
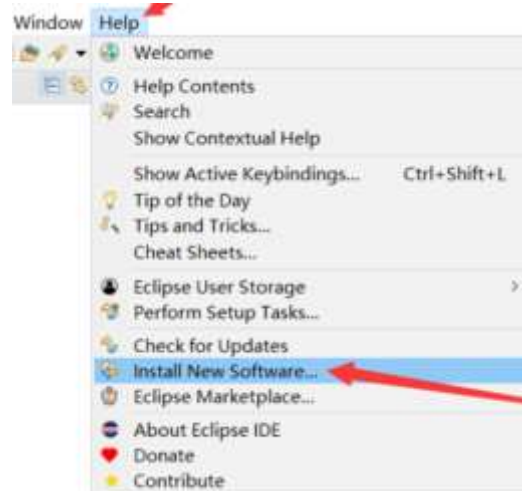
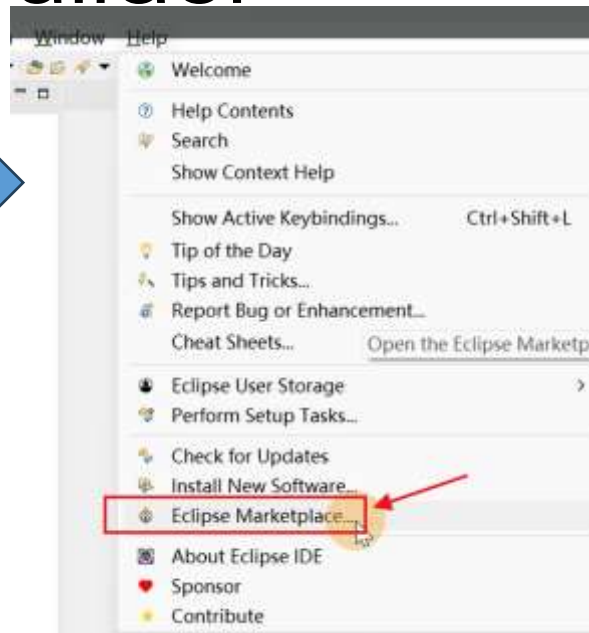
- (过程中伴随AI的使用)

UI编辑环境_WindowBuilder

1、在线-商店（力荐）

2、在线-链接

3、离线安装包



分享大纲

- 过往案例展示

- Demo实现前提：

- MVC架构

- 用到的设计模式

- UI编辑环境

- Demo实现：

- 系统分析与设计

- 框架引入及基本介绍

- 前后端实现-现场版

- (过程中伴随AI的使用)

框架引入及基本介绍

模拟下载源码后，用eclipse运行起来

M:

创建+继承+实现

V:

创建 (windowbuilder)

继承

实现

C:

创建+继承+实现

打开界面:

```
MVC.openModule(ClsCtrl.class);
```

关闭界面:

```
MVC.closeModule(ClsCtrl.class);
```

添加按钮事件:

拖动+双击+添加action

本地数据传递:

增加PID.java

```
TESTVIEW_TO_MAINVIEW
```

增加结构体: comp.ntss.proto.pdtos

```
PDTO_TestView2MainView
```

构造+赋值+发送消息:

```
PDTO_x t = new PDTO_x();
```

```
t.testStr = s;
```

```
sendMsg(PID.TESTVIEW_TO_MAINVIEW, t);
```

(tarCtrl)注册消息:

```
public PID[] registerPIDList() {  
    PID[] list = {  
        PID.TESTVIEW_TO_MAINVIEW,  
    };  
    return list;  
}
```

(tarView)接收数据:

```
public void updateView(PDTO pdto) {  
    PID pId = pdto.getPID();  
    switch (pId) {  
        case TESTVIEW_TO_MAINVIEW:  
            this.textArea.setText(((PDTO_TestView2MainView)pdto).testStr);  
            break;  
    }  
}
```

界面初始化后调用:

```
/**  
 * @param params  
 * @description 界面打开调用方法，用于子类重写  
 */  
protected void onOpen(ArrayList<Object> params) {  
    //this.updateCurPath();  
}
```

关闭界面不退出程序，且能够再次打开:

```
@Override  
public void windowClosing(WindowEvent e) {  
    MVC.closeModule(XxxCtrl.class);  
}
```

分享大纲

- 过往案例展示

- Demo实现前提：

- MVC架构

- 用到的设计模式

- UI编辑环境

- Demo实现：

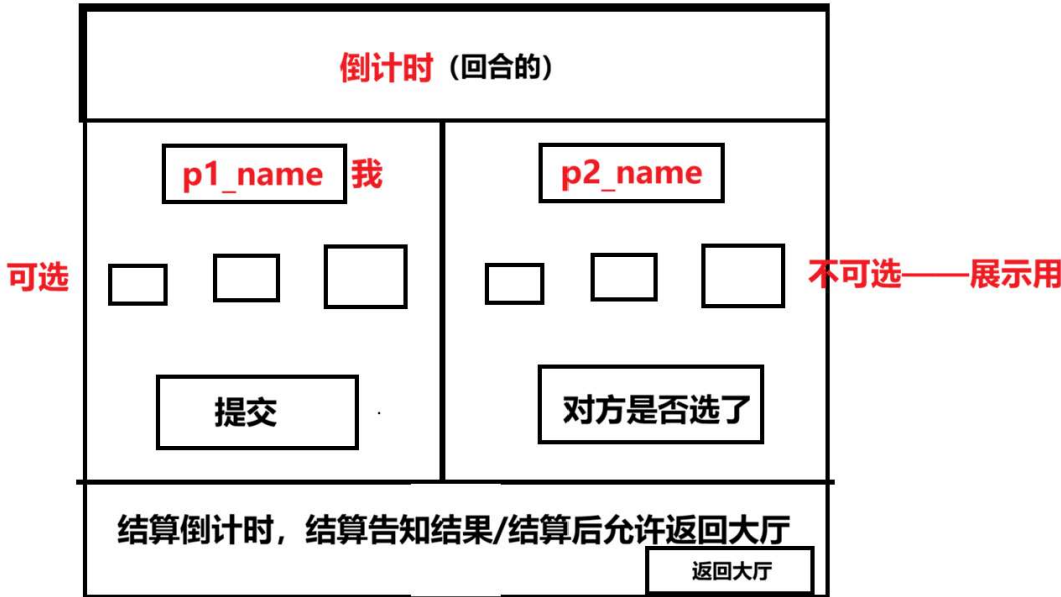
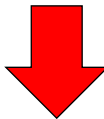
- 系统分析与设计

- 框架引入及基本介绍

- 前后端实现-现场版

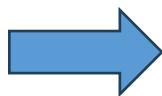
- (过程中伴随AI的使用)

基于现有讲解+实现：



全面回顾，查漏补缺

API 及应用框架说明



- ▲ 1. 框架类作用简要说明
 - ▷ 1.1. 客户端
 - ▷ 1.2. 服务端
- ▲ 2. 创建一个模块的流程
 - ▷ 2.1. 双端通用流程
 - 2.2. 获取模块数据
 - ▷ 2.3. 客户端发协议
 - ▷ 2.4. 服务器端响应协议
- 3. 开发一个模块的流程
- ▲ 4. 常用介绍
 - ▷ 4.1. JDK常用
 - ▷ 4.2. 框架生命周期
 - ▷ 4.3. 框架自定义布局
- ▲ 5. 注意事项
 - 5.1. IDE的编码——UTF-8
- ▲ 6. 考虑扩展
 - 6.1. BaseAloneView

感谢聆听，欢迎反馈

如本次分享有所帮助，请点个赞，有任何问题请评论/留言

本次一些内容只是概括性地讲解下，如果有任何想要更详细分享的部分/想要听的内容，也都欢迎评论/留言

如有需要可以讲特定内容：

- 框架源码的解读<熟悉了可以自己改框架>
- 框架中设计模式的具体体现
- 如何结合数据库进行数据持久化开发
- 如何和舍友使用svn协同开发小软件<框架提供了模块化开发的基础>

获取源码|最新分享：



源码+文档地址github、gitee、（下方为网盘）：

通过网盘分享的文件：02_Java联网猜拳小游戏链接：
<https://pan.baidu.com/s/1n571GwCwKCGnnqOI6UW9kg?pwd=38pk> 提取码: 38pk --来自百度网盘超级会员的分享

