阶段一：学习使用框架

阶段二：使用框架实现游戏业务

阶段三：写框架

阶段四：商业化部署

学习使用框架的方法：

- 读文档
- 装环境
- 写用例

zinx描述

zinx框架是一个处理多路IO的框架。在这个框架中提供了若干抽象类，分别在IO处理的多个阶段生效。开发者可以重写抽象类中的虚函数完成自己需求的处理功能。

zinx框架的使用步骤

1. ZinxKernel::ZinxKernelInit() 初始化框架
2. 写类继承AZinxHandler，重写虚函数，在函数中对参数进行处理（比如将参数内容打印到标准输出）
3. 写类继承Ichannel，重写虚函数完成数据收发，重写GetInputNextStage 函数，返回第二步创建类的对象
4. 添加步骤3类创建的对象到框架中
5. 运行框架

标准输入回显标准输出的编写思路

1. 创建三个类：标准输入类，回显类，标准输出类
2. 重写标准输入类的读取函数
3. 重写回显类处理函数
4. 重写标准输出类的写出函数
5. 创建以上三个类的全局对象（堆对象），添加通道对象到框架（kernel）
6. 运行框架

添加命令处理类

1. 创建命令处理类继承AzinxHandler，重写处理函数和获取下一个处理环节的函数
2. 处理函数内，根据输入内容不同，要么添加输出通道，要么摘除输出通道
3. 获取下一个处理环节函数中，指定下一个环节是退出或回显
4. 设定输入通道的下一个环节是该类对象

添加日期前缀

1. 创建添加日期类，继承AzinxHandler。重写处理函数和获取下一环节函数
2. 处理函数：将日期和输入字符串拼接后，new一个对象返回
3. 获取下一环节函数：返回回显对象
4. 在命令处理类的处理函数中：根据输入命令设置当前是否要添加前缀的状态位
5. 在命令处理类的获取下一环节函数中，判断当前状态，需要添加前缀--》返回添加日期前缀的对象；不需要添加前缀--》返回回显对象

需要调用的框架静态函数

- 初始化，去初始化 `ZinxKernel::ZinxKernelInit()` 和 `ZinxKernel::ZinxKernelFini()`
- 运行框架 `ZinxKernel::Zinx_Run()`
- 通道添加和摘除 `ZinxKernel::Zinx_Add_Channel()` 和 `ZinxKernel::Zinx_Del_Channel()`
- 退出框架 `ZinxKernel::Zinx_Exit()`

多个AzinxHandler对象之间的信息传递

- 数据封装成IzinxMsg类在多个AzinxHandler对象之间传递
- 使用时，要现将IZinxMsg类型引用动态转换成所需类型引用

zinx框架处理数据的本质

- 将数据在多个AzinxHandler对象之间传递，挨个处理
- 传递的规则通过重写GetNextHandler函数定义