tank_01

项目的目的：通过项目建立面向对象的设计思维，并复习OO，容器、网络、线程等基础知识。为以后的课程做好准备。

资料：

1：阿里Java手册

2：GUI入门（不重要，入门即可）

学编程的不二法门：敲 敲 敲

代码量代表一切

代码量代表一切

代码量代表一切

1. IJ建立新项目 Tank2019V2 .
2. 创建一个窗口
3. 显示这个窗口
4. 画出一个方块
    a. 谁初始化了Graphics g这个参数
5. 移动这个方块
    a. 自动化，repaint - paint
    b. 手动控制：响应键盘事件

i. Frame.addKeyListener() -> Observer

6. 为什么使用内部类?
    a. 不需要让别的类访问键盘监听类
    b. 高内聚 低耦合
7. 为什么不使用方法的内部类（局部内部类）
    a. 可以用，看起来不方便
    b. 内部类可以非常方便访问包装类的局部变量
8. 为什么用KeyAdapter而不是直接实现KeyListener
9. 最基础的面向对象的设计思想
    a. 抽象出名词：类， 属性
    b. 抽象出动词：方法
10. 为什么Enum 比 int类型好?
    a. 编译期间就能知道赋值是不是有问题
11. STOP作为单独的方向不合适
    a. 怎么做？ （作业）

tank_02

1. 怎么 样处理坦克静止状态
    a. moving = false;
2. 将坦克换成图片
    a. 关于classloader的基础知识
    b. 显示图片，使用ImageIO
3. 用双缓冲解决闪烁问题（不重要）
    a. repaint - update
    b. 截获update
    c. 首先把该画出来的东西（坦克， 子弹）先画在内存的图片中，图片大小和游戏画面一致

d. 把内存中图片一次性画到屏幕上（内存的内容复制到显存）

4. 加入敌军坦克

a. 分拨儿Group

5. 打出一颗子弹

a. 按下Ctrl键，主战坦克打出一颗子弹

b. 用面向对象的思想考虑

6. 打出一串子弹

a. 将子弹装在容器中

tank_03

7. 做边界检查，当子弹飞出游戏区，应该从List中删掉
8. 一个排的敌人坦克，全部干掉
9. 将子弹从坦克中心位置打出

a. 根据坦克图片的大小，和左上角的位置计算子弹左上角的位置

10. 子弹与敌军坦克的碰撞检测

a. 击毁一辆坦克

11. 加入多辆敌军坦克
12. 加入爆炸
13. 加入声音

bug的级别：

1：编译问题

2：运行有异常

非常好定位 - 沿着逻辑线检查 - 输出中间结果

3：运行没异常，结果不对

4：运行没异常，结果时而对，时而不对

记录详细的日志 通过日志排查

tank_04

1. 用配置文件让程序更灵活
    a. java.util.Properties
2. 用策略模式让子弹发射更灵活
    a. Player Tank
    b. 方便切换主战坦克的发射模式
        i. default 发一颗子弹
        ii. 高级的子弹 4个方向的子弹
3. 考虑加入一堵墙 一枚地雷 一座机枪塔
    a. 添加新的游戏物体，让TankFrame表现的比较优雅 add(xx) add(xx)...
    b. 碰撞如何考虑?
        i. 新加进来的游戏物体 -> 无缝结合的碰撞检测
        ii. 不需要修改原来的代码

tank_05

1. 碰撞如何考虑?
        iii. 新加进来的游戏物体 -> 无缝结合的碰撞检测
        iv. 不需要修改原来的代码

作业:

    c. 加入新的碰撞逻辑
        i. Tank-Tank back()

tank_06

1. Model和View分离 （MVC）
    a. 张三 88 李四 92 王五 72 ...
    b. 图表 饼图 线框 ...

2. 坦克 Model View
    a. GameModel
    b. Facade
        i. 门面
    c. Mediator
        i. 调停者
3. 存盘功能（序列化）
    a. Serializable接口
4. 线程
    a. 复习与补充

tank_07

1. 网络模型：
    a. 通讯方式：TCP UDP
        i. TCP -> 可靠连接 使命必达 速度慢
        ii. UDP -> 不可靠，速度快
2. TCP的编程模型
    a. BIO OIO - >
        i. Blocking IO : Old IO
    b. NIO
        i. New IO : Non-Blocking IO
        ii. Selector
        iii. ByteBuffer (single pointer)
    c. AIO
        i. Asynchronous IO
3. Netty
    a. 封装了NIO ByteBuf (read pointer, write pointer)

tank_08

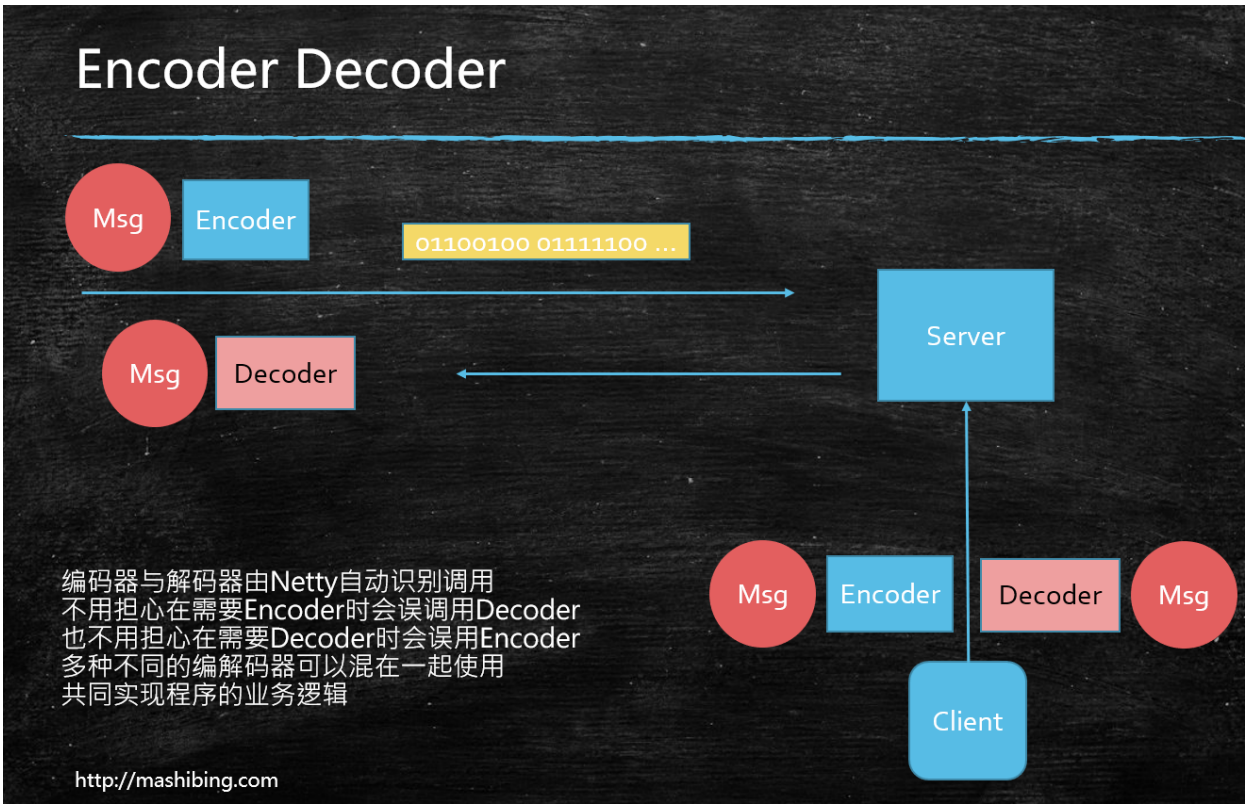
1. Callable and Future
2. hello netty

tank_09

1. 写一个基于Netty的聊天室
2. 基于Netty的基本回路实验

tank_10

1. Netty Coder and Decoder
2. Embedded Channel 进行单元测试
3. 作业：
    a. 完成模型：



tank_12

1. 抽象出Msg
2. MsgType
3. 搭建消息处理的框架
4. TankStartMoving
5. TankStop
6. 作业：
    a. TankDirChanged