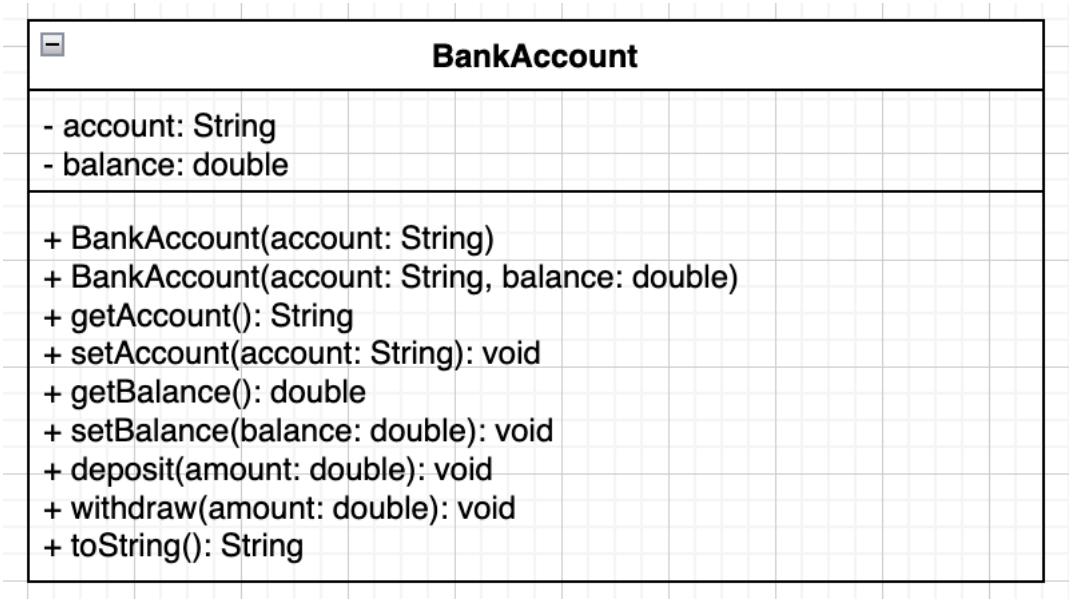
1. 银行账户类

涉及知识点：

– UML类图
– 类和对象
– 构造函数
– public/private
– setter/getter
– 重载/重写
– 正则表达式
– 异常处理

根据类图，实现BankAccount类。



属性

– account：银行账号（字符串），私有属性，长度必须为8位的纯数字，且不以0开头。
– balance：账户余额（双精度浮点数），私有属性，余额不能为负数。

方法

public BankAccount(String account)
根据账号，创建一个银行账户对象。如账号不合法，设为为空字符串；余额默认为0。

public BankAccount(String account, double balance)
根据账号和给定余额，创建一个银行账户对象。如账号不合法，设为空字符串；如余额不合法，设为0。

public String getAccount()
返回账号。

public void setAccount(String account)

设置账号，如账号不合法，不作修改。

public double getBalance()
返回余额。

public void setBalance(double balance)
设置余额，如余额不合法，不作修改。

void deposit(double amount)
存入amount金额到账户中，如amount不合法，不作任何操作。

void withdraw(double amount)
从账户取出amount金额，如amount不合法或余额不足，不作任何操作。

@Overload
public String toString()
返回一个用于描述账户信息的字符串，格式如下：
账号：xxxxxxxx，余额：xx.xx元

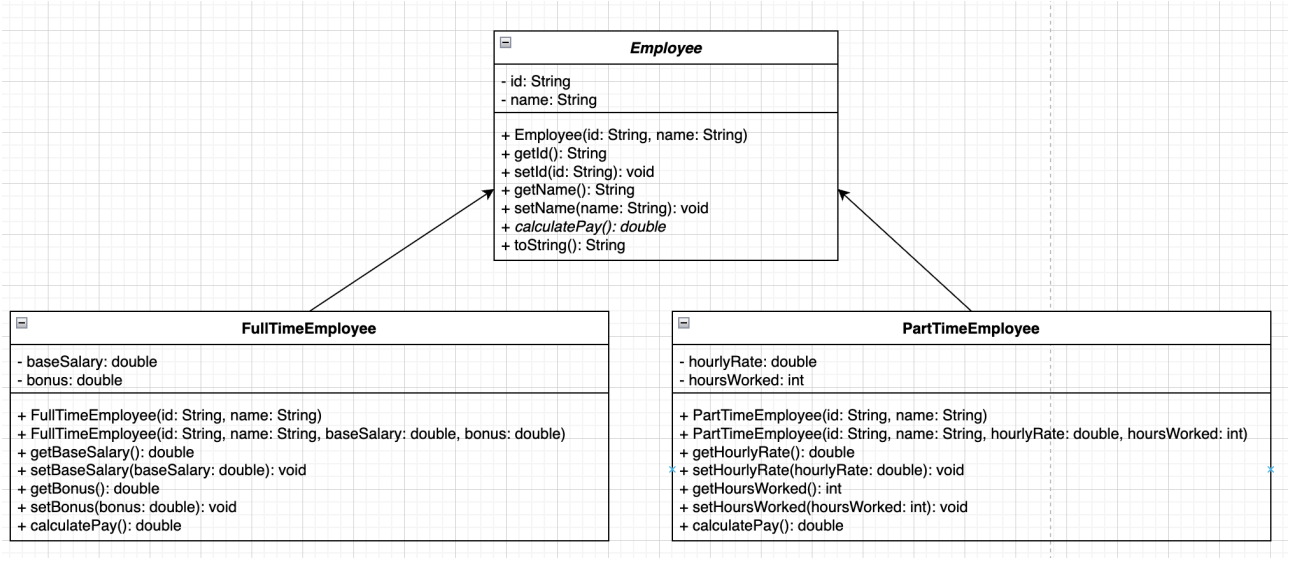
改进：使用抛出异常，来处理不合法的情况。

2. 员工工资计算

设计知识点：

- 继承
- 异常处理
- 抽象类/抽象方法
- 重载/重写
- 多态
- 接口

需要为一个公司的人事部门设计一个用于计算员工工资的系统。公司员工分为全职员工和兼职员工。



员工类Employee

该类为抽象类，包含一个抽象方法calculatePay()，用于计算员工的工资。

全职员工类FullTimeEmployee

该类继承于Employee。全职员工的工资由基本工资和奖金构成，工资=基本工资+奖金。

兼职员工类PartTimeEmployee

该类继承于Employee。兼职员工的工资由时薪和工作小时构成，工资=时薪*工作小时。

改进1：使用多态，将多个员工保存在一个数组中。

改进2: 设计Payable接口，让Employee类实现Payable接口。