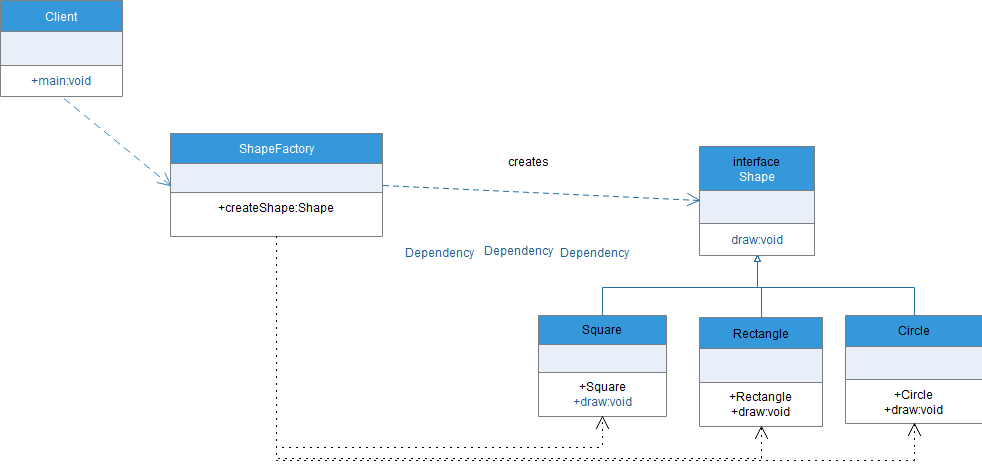
1. 简单工厂UML



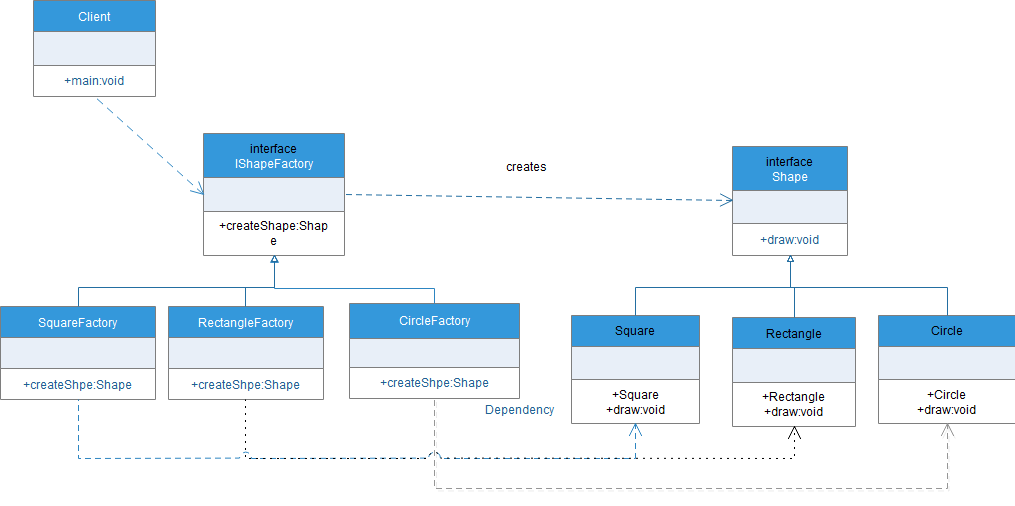
工厂角色：ShapeFactory

抽象产品角色: Shape

具体产品角色:Square,Rectangle,Circle

客户端:Client

1. 工厂方法UML



抽象工厂角色:IShapeFactory

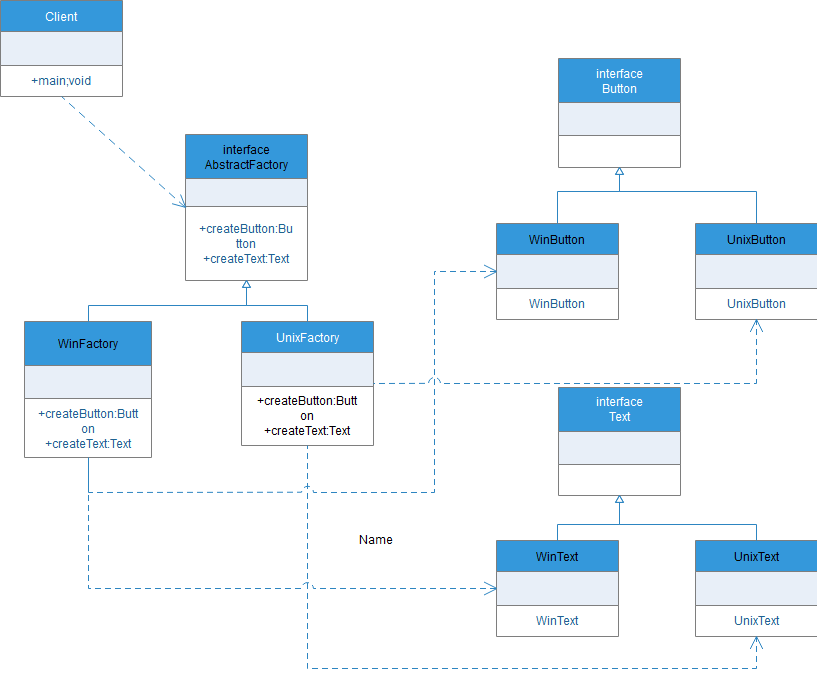
工厂角色：SquareFactory,RectangleFactory,CircleFactory

抽象产品角色: Shape

具体产品角色:Square,Rectangle,Circle

客户端:Client

1. 抽象工厂UML



抽象工厂角色:AbstractFactory

工厂角色：WinFactory,UnixFactory

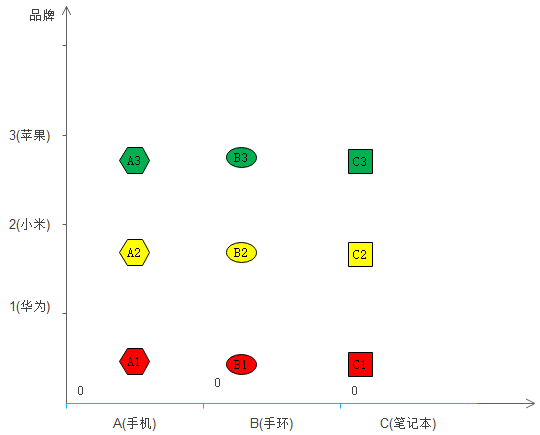
抽象产品角色: Button,Text

具体产品角色:WinButton,WinText, UnixButton,UnixText

客户端:Client

4.深刻理解产品族和产品等级结构之间的关系

产品族可以理解为一种品牌（如华为，小米，苹果等），产品等级结构可以理解为某种品牌下面的不同产品（如华为旗下的手机，笔记本，智能手环）,通过品牌和产品等级结构可以确定具体的产品角色，如下图所示



如上图A1代表华为手机，B1代表华为手环，C1代表华为笔记

5.自主完成三种工厂模式的代码实现，理解简单工厂、工厂方法和抽象工厂的区别

工厂方法定义了一个抽象工厂管理，每一种具体的产品由实现了抽象管理工厂的具体产品工厂负责创建，使得客户端不依赖产品的具体实现。符合开闭原则，增加新的产品时只需增加具体工厂实现和具体产品实现即可，无需修改原有代码

简单工厂是工厂方法模式的简化和特殊实现，简单工厂省略了抽象工厂角色，直接以具体工厂管理多个产品的创建。不完全符合开闭原则，增加新产品时，需要对工厂角色代码进行修改和增加具体产品实现

抽象工厂是工厂方法模式的进一步优化和演变，抽象工厂可以减少了具体工厂的数量，以品牌工厂管理某种品牌的具体产品的创建。不完全符合开闭原则，增加产品族时符合开闭，只需增加新的品牌工厂和增加相应具体极品实现；增加产品等级结构时不符合开闭原则，需要修改原品牌工厂和所有原具体产品实现。