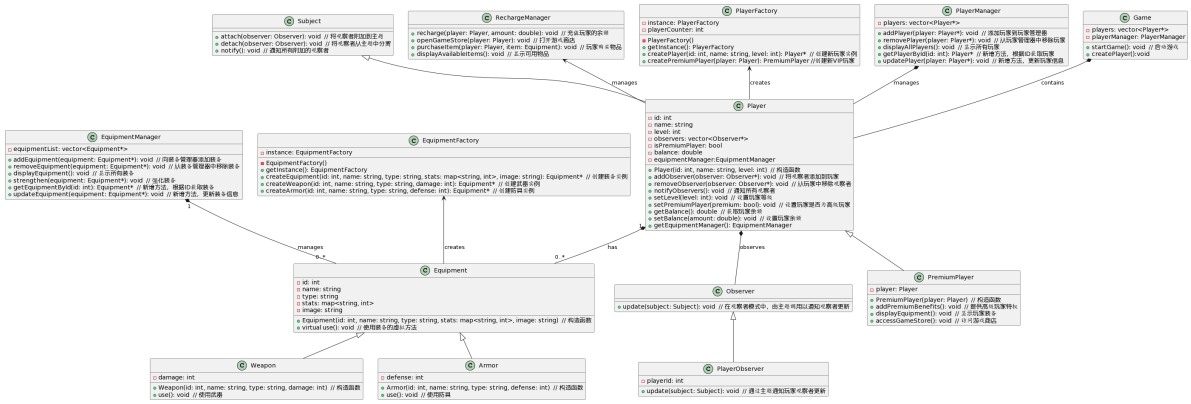
圣遗物管理系统




```
@startuml
class Observer {
  + update(subject: Subject): void  // 在观察者模式中，由主题调用以通知观察者更新
}

class Subject {
  + attach(observer: Observer): void  // 将观察者附加到主题
  + detach(observer: Observer): void  // 将观察者从主题中分离
  + notify(): void  // 通知所有附加的观察者
}

class PlayerObserver {
  - playerId: int
  + update(subject: Subject): void  // 通过主题通知玩家观察者更新
}

class RechargeManager {
  + recharge(player: Player, amount: double): void  // 充值玩家的余额
  + openGameStore(player: Player): void  // 打开游戏商店
  + purchaseItem(player: Player, item: Equipment): void  // 玩家购买物品
  + displayAvailableItems(): void  // 显示可用物品
}

class PlayerFactory {
  - instance: PlayerFactory
  - playerCounter: int
  - PlayerFactory()
  + getInstance(): PlayerFactory
  + createPlayer(id: int, name: string, level: int): Player*  // 创建新玩家实例
  + createPremiumPlayer(player: Player): PremiumPlayer //创建新VIP玩家
}

class Player {
  - id: int
  - name: string
  - level: int
  - observers: vector<Observer*>
  - isPremiumPlayer: bool
```

```
  - balance: double
  - equipmentManager:EquipmentManager
  + Player(id: int, name: string, level: int)  // 构造函数
  + addObserver(observer: Observer*): void  // 将观察者添加到玩家
  + removeObserver(observer: Observer*): void  // 从玩家中移除观察者
  + notifyObservers(): void  // 通知所有观察者
  + setLevel(level: int): void  // 设置玩家等级
  + setPremiumPlayer(premium: bool): void  // 设置玩家是否为高级玩家
  + getBalance(): double  // 获取玩家余额
  + setBalance(amount: double): void  // 设置玩家余额
  + getEquipmentManager(): EquipmentManager
}

class PlayerManager {
  - players: vector<Player*>
  + addPlayer(player: Player*): void  // 添加玩家到玩家管理器
  + removePlayer(player: Player*): void  // 从玩家管理器中移除玩家
  + displayAllPlayers(): void  // 显示所有玩家
  + getPlayerById(id: int): Player*  // 新增方法，根据ID获取玩家
  + updatePlayer(player: Player*): void  // 新增方法，更新玩家信息
}

class PremiumPlayer {
  - player: Player
  + PremiumPlayer(player: Player)  // 构造函数
  + addPremiumBenefits(): void  // 提供高级玩家特权
  + displayEquipment(): void  // 显示玩家装备
  + accessGameStore(): void  // 访问游戏商店
}

class Game {
  - players: vector<Player*>
  - playerManager: PlayerManager
  + startGame(): void  // 启动游戏
  + createPlayer():void
}

class Equipment {
  - id: int
  - name: string
  - type: string
  - stats: map<string, int>
  - image: string
  + Equipment(id: int, name: string, type: string, stats: map<string, int>,
image: string)  // 构造函数
  + virtual use(): void  // 使用装备的虚拟方法
}

class EquipmentManager {
  - equipmentList: vector<Equipment*>
  + addEquipment(equipment: Equipment*): void  // 向装备管理器添加装备
  + removeEquipment(equipment: Equipment*): void  // 从装备管理器中移除装备
  + displayEquipment(): void  // 显示所有装备
  + strengthen(equipment: Equipment*): void  // 强化装备
  + getEquipmentById(id: int): Equipment*  // 新增方法，根据ID获取装备
```

```
  + updateEquipment(equipment: Equipment*): void  // 新增方法，更新装备信息
}

class Weapon {
  - damage: int
  + Weapon(id: int, name: string, type: string, damage: int)  // 构造函数
  + use(): void  // 使用武器
}

class Armor {
  - defense: int
  + Armor(id: int, name: string, type: string, defense: int)  // 构造函数
  + use(): void  // 使用防具
}

class EquipmentFactory {
  - instance: EquipmentFactory
  - EquipmentFactory()
  + getInstance(): EquipmentFactory
  + createEquipment(id: int, name: string, type: string, stats: map<string, int>,
image: string): Equipment*  // 创建装备实例
  + createWeapon(id: int, name: string, type: string, damage: int): Equipment*
// 创建武器实例
  + createArmor(id: int, name: string, type: string, defense: int): Equipment*
// 创建防具实例
}

Player "1" *-- "0..*" Equipment : has
EquipmentManager "1" *-- "0..*" Equipment : manages
Equipment <|-- Weapon
Equipment <|-- Armor
Subject <|-- Player
Observer <|-- PlayerObserver
Player *-- Observer : observes
PlayerFactory <-- Player : creates
EquipmentFactory <-- Equipment : creates
Player <|-- PremiumPlayer
Game *-- Player : contains
RechargeManager <-- Player : manages
PlayerManager *-- Player : manages
@enduml
```