策略模式

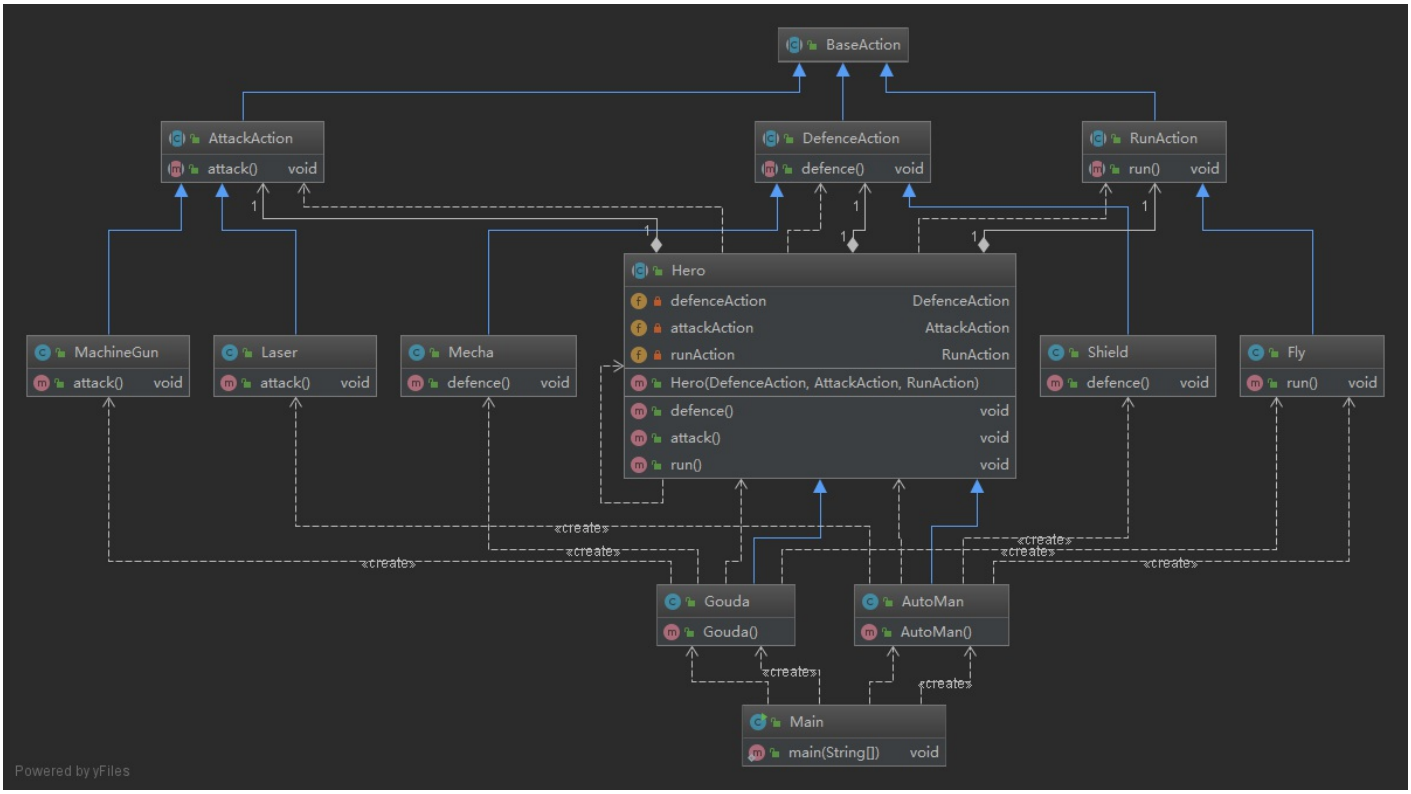
模式定义

定义了算法族, 分别封装起来, 让他妈之间可以互相替换, 此模式让算法的变化独立 于使用算法的客户

问题需求描述

模拟一个奥特曼和高达, 他们有Attack, Defence, Run三种动作

UML



Code

Hero.java

```
package Strategy;

public abstract class Hero {
    private DefenceAction defenceAction;
    private AttackAction attackAction;
    private RunAction runAction;

    public Hero(DefenceAction defenceAction, AttackAction attackAction, RunAction runAction) {
        this.defenceAction = defenceAction;
        this.attackAction = attackAction;
        this.runAction = runAction;
    }

    public void defence() {
        defenceAction.defence();
    }

    public void attack() {
        attackAction.attack();
    }
```

```
    public void run() {
        runAction.run();
    }
}
```

Gouda.java

```
package Strategy;

public class Gouda extends Hero{
    public Gouda() {
        super(new Mecha(), new MachineGun(), new Fly());
    }
}
```

AutoMan.java

```
package Strategy;

public class AutoMan extends Hero {
    public AutoMan() {
        super(new Shield(),new Laser(),new Fly());
    }
}
```

BaseAction.java

```
package Strategy;

public abstract class BaseAction {
}
```

DefenceAction.java

```
package Strategy;

public abstract class DefenceAction extends BaseAction{
    public abstract void defence();
}
```

AttackAction.java

```
package Strategy;

public abstract class AttackAction extends BaseAction {
    public abstract void attack();
}
```

RunAction.java

```
package Strategy;

public abstract class RunAction extends BaseAction {
    public abstract void run();
}
```

Fly.java

```
package Strategy;

public class Fly extends RunAction {

    @Override
    public void run() {
        System.out.println("good bye!!");
    }
}
```

MachineGun.java

```
package Strategy;
```

```
public class Laser extends AttackAction {
    @Override
    public void attack() {
        System.out.println("biubiubiu!!!");
    }
}
```

Mecha.java

```
package Strategy;

public class Mecha extends DefenceAction {

    @Override
    public void defence() {
        System.out.println("Mecha defence!");
    }
}
```

Shield.java

```
package Strategy;

public class Shield extends DefenceAction {
    @Override
    public void defence() {
        System.out.println("ha!!");
    }
}
```

运行结果

Main.java

```
package Strategy;

public class Main {
    public static void main(String[] args) {
        AutoMan autoMan = new AutoMan();
        Gouda gouda = new Gouda();
        System.out.println("now is automan!");
        autoMan.attack();
        autoMan.defence();
        autoMan.run();
        System.out.println("now is gouda!!");
        gouda.attack();
        gouda.defence();
        gouda.run();
        System.out.println("finish!");
    }
}
```

out

```
now is automan!
biubiubiu!!!
ha!!
good bye!!
now is gouda!!
boom!boom!boom!!!
Mecha defence!
good bye!!
finish!
```