Applying Object Orientation

Roland Guijt
INDEPENDENT SOFTWARE DEVELOPER AND TRAINER

@rolandguijt   www.rmgsolutions.nl

Module Overview

Polymorphism

Inheritance

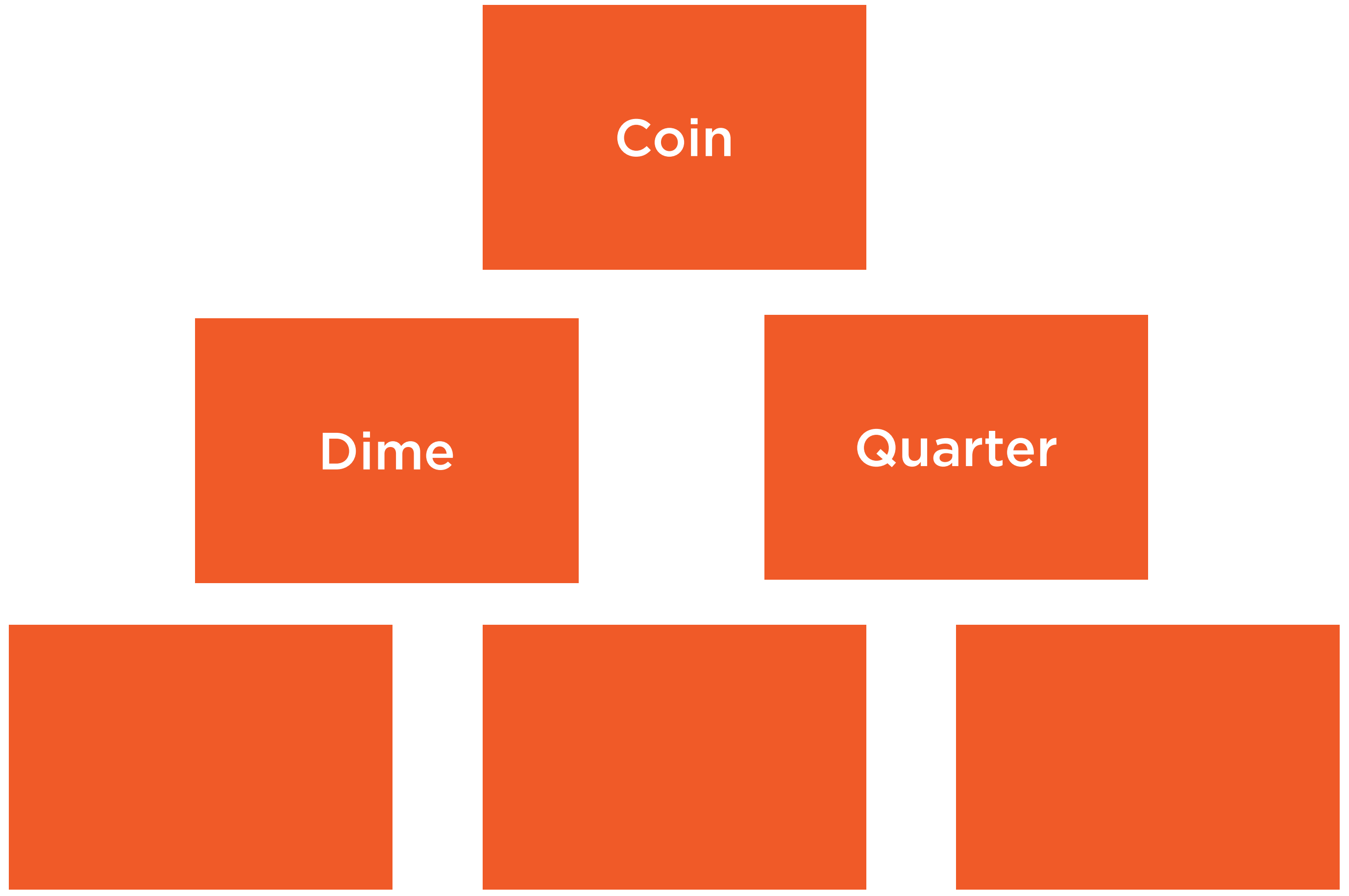
Abstract classes

Interfaces

Inheritance

Base class

Derived classes



Super

```
class Base {
    baseMethod() {

    }
}

class Derived extends Base {
    baseMethod() {
        //before
        super.baseMethod();
        //after
    }
}
```

Protected

```
class Base {
        protected num = 12;
}

class Derived extends Base {
        someMethod() {
                return num; //ok
        }
}

let d = new Derived();
d.num; //doesn't work
```

Protected

For encapsulation

Hide complexity

Maintainable code

Hide as much as possible

Low coupling

Interfaces

Separate Types

Can’t contain implementation

Act as a contract for a class

A class can implement multiple

Can be used with unrelated classes

Summary

Inheritance

Polymorphism

Abstract classes

Super

Interfaces