

OO Programming

The details of Java

OO Concepts

1. Abstraction

2. Encapsulation

3. Inheritance

4. Polymorphism and **Dynamic Dispatching**

5. Runtime vs Compile Time

Abstraction

Class vs Instance

Creating
abstractions of
concepts that we can
re-use

BankAccount
accountNo balance

Robert's Account
A8624 \$500

Julia's Account
A6363 \$800

```
BankAccount robertsAccount;  
BankAccount juliasAccount;
```

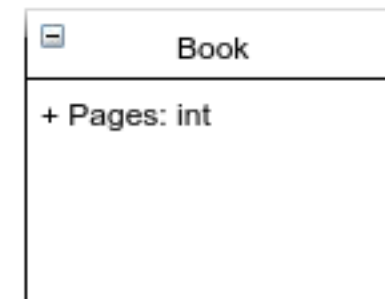
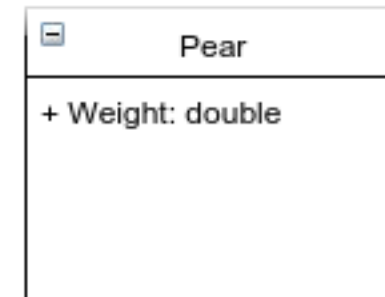
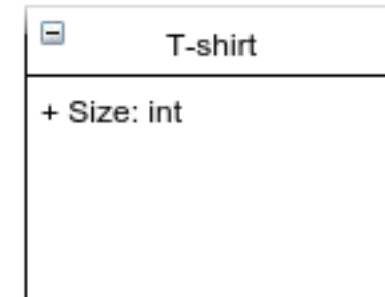
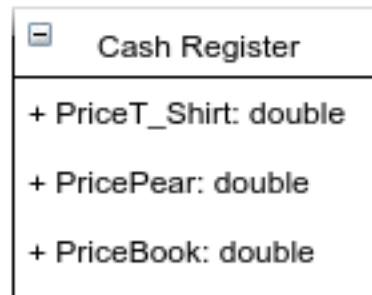
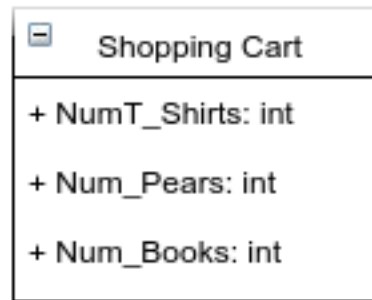
Encapsulation (information Hiding)

Hiding parts of a program in order to create a **Standard Interface**

```
public class Employee {  
    private BigDecimal salary = new BigDecimal(50000.00);  
  
    public BigDecimal getSalary() {  
        return salary;  
    }  
  
    public static void main() {  
        Employee e = new Employee();  
        BigDecimal sal = e.getSalary();  
    }  
}
```

Encapsulation (information Hiding)

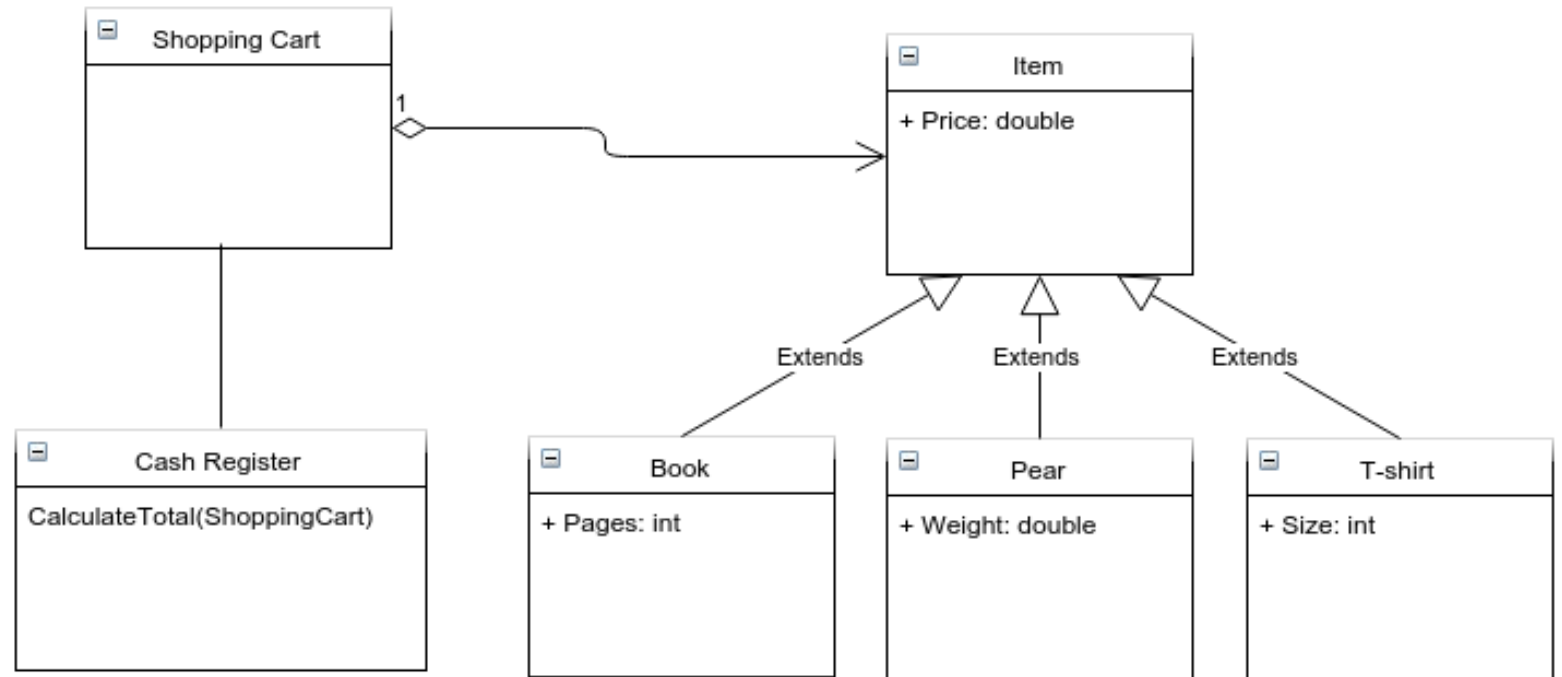
Who should have the **price** attributes?



Encapsulation (information Hiding)

Who should have the **price** attributes?

The information Expert



Inheritance

When to use:
Reusing **attributes**
and **Methods**
Creating
abstractions of
concepts



```
class Convertible {  
  // Key (private)  
  // Speed : 155 (miles / hour)  
  // Weight 1600 kg  
  // Engine : 3.2 L S54 inline-6  
}
```



```
class Roadster extends Convertible {  
  // Speed : 165 (miles / hour)  
  // Weight 1399 kg  
}
```

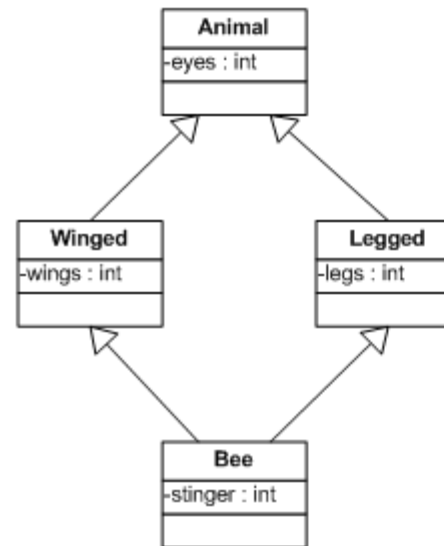
Inheritance Diamond Problem

How can we create a class that can be **charged** and **paid**?

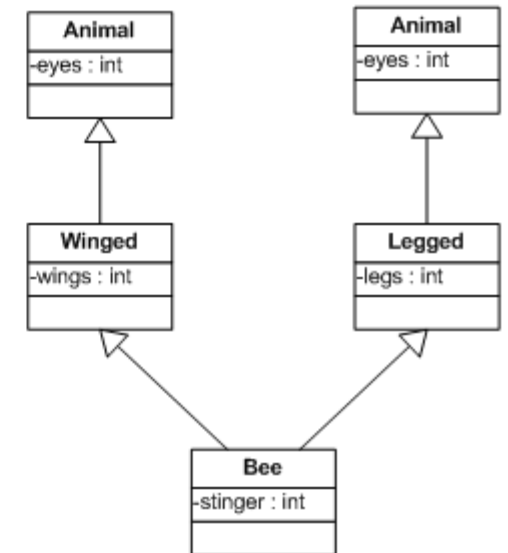
Java will not allow this!

The Diamond Problem

Virtual Inheritance

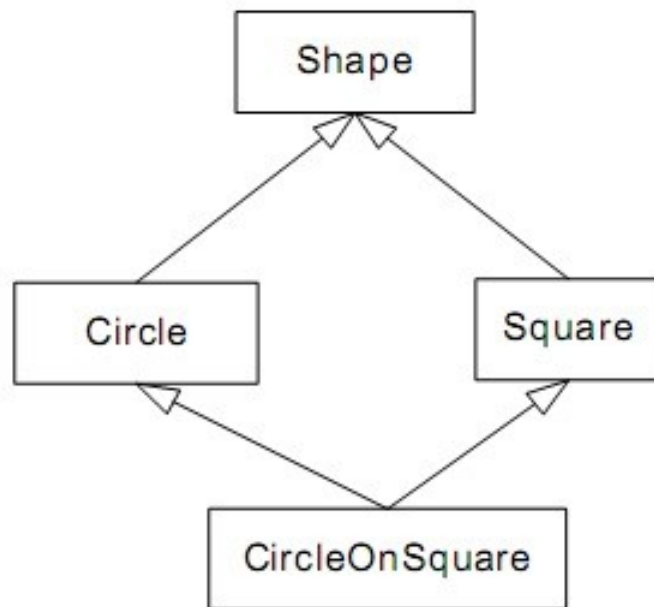


Non-Virtual Inheritance

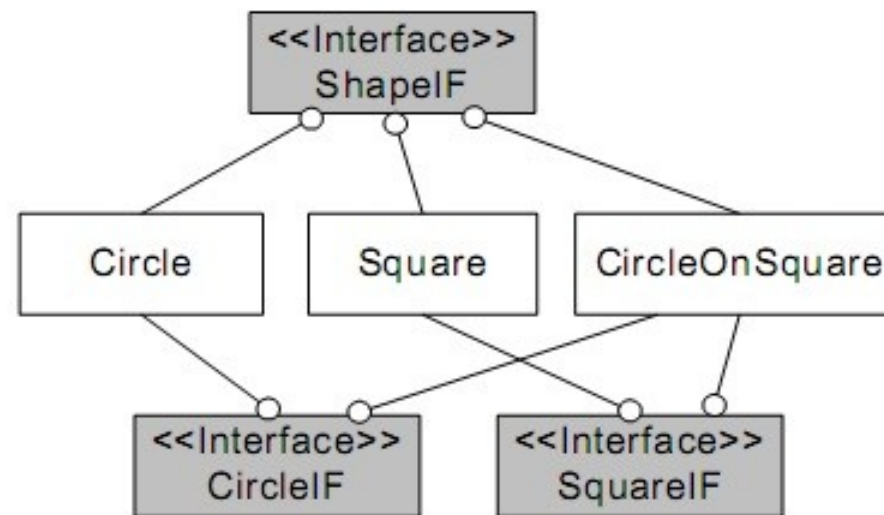


Interfaces

Standardizing a method across multiple classes



No multiple inheritance in JAVA

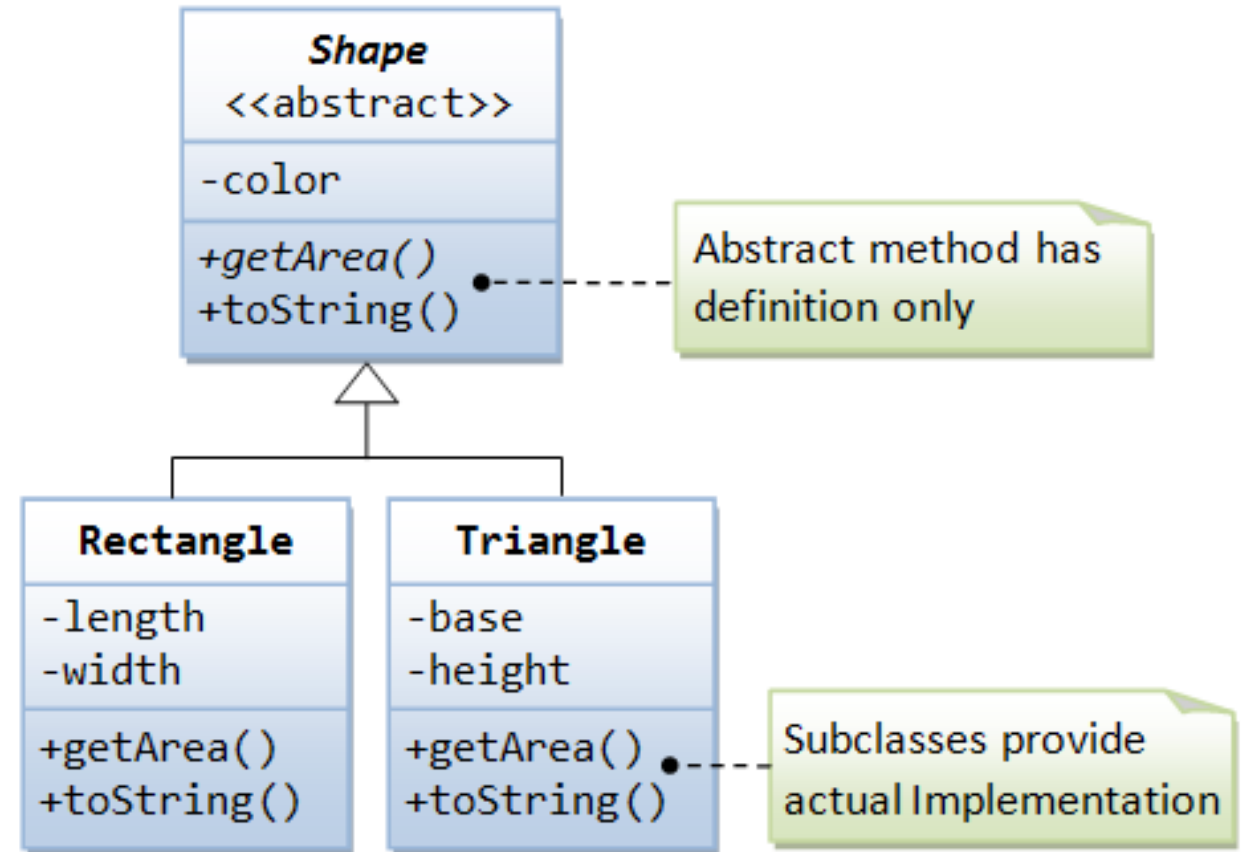


Multiple interface inheritance in JAVA

Abstract class

Problem: We want to reuse properties across many classes. The result does not exist in real life. Can we represent that abstraction in code?

Solution: Abstract class



Dynamic Method dispatching

Problem: Want to create multiple enemies with the same interface but have different implementations

Solution: Dynamic method Dispatching

See code example in the repo if you did not attend tutorial.

Runtime vs Compile time

When is **Compile** time? When is **Runtime**? What does it mean?
Mistakes happen when you are:

1. writing code
2. Compiling
3. Executing code

Understanding where in the **cycle** the error happens helps **diagnose** the issue.

. Is there an example?

. **Yes**, see the code on the repo if you did not attend the tutorial

Concept Questions

Will an IDE catch compile time errors?

Will an IDE catch runtime errors?

Is there a difference between an abstract class and an interface?

Deliverables

Milestone 1 is due.

References

1.Runtime vs compile time explanation:

<https://www.youtube.com/watch?v=QmvmZqpthbc>

Dynamic method dispatching: <https://www.youtube.com/watch?v=PK2mZ39AAzE>

OO blog: <https://theleanocoder.wordpress.com/2015/10/11/the-four-pillars-of-object-oriented-design/>