enum
interface
class

enum vs Enumeration

compile
unity
In C:

```c
enum
{
    enum season = { SPRING,
                    SUMMER,
                    WINTER,
                    FALL },

    enum season current Season;
};
```
if ( \text{WINTER} == \text{current season} ) \{ \\
    \text{...} \\
\}

if ( z == c ) \{ \\
    \text{...} \\
\}
In C

```c
CurrentSeason = 17;
CurrentSeason = WINTER + 12;
// neither line is a compile error
```
// iterate seasons in order

currentSeason += j // ?

Enums in Current typesafe!

Enums are self-documenting
printf("Season", currentSeason);

[In Java objects have toString]

Enums are hard to work with.

.
In Java, dozens of different techniques for implementing Enums.
In Java 1.5 enum
official part of Java

Shape.java

```
class Shape {

enum Planet {

}
```
public enum Planets {
    MERCURY,
    VENUS,
    EARTH,
    NEPTUNE
}

Planets myPlanet;
myPlanet = MARS;

myPlanet = 14;  // Error: myPlanet is a constant
myPlanet = MARS + 14;  // Error: myPlanet is a constant
```java
public enum Planets {
    private double mass;
    private double radius;

    public Planets(double m, double r) {
        // constructor
    }

    public void setMass() {
        // return mass
    }

    public String toString() {
        // return string
    }
}
```
public enum Planets {
    MERCURY (10, 15),
    VENUS (8, 5),
    MARS (5, 3),
    EARTH (3, 4),
    ;
The third type of entity in Java is the interface.

(Enumeration is an interface)
Polymorphism —

use one type of variable
in place of another.

Shape

\ /
\ /
\ /
Triangle Circle ... Polygon
For polymorphism all we (usually) care about is whether a method exists.
class Circle extends Shape {
    public void draw();
}

class Triangle extends Shape {
    public void draw();
}

class Shape {
    public void draw();
}
class Rhombus extends Shape {

  // either inherits or
  // provides a draw method
What code should I put in Shape.draw()?

```java
class Shape {
    // Some methods
}
```
class Rhombus extends Rhombus

    int height;
    int base;
    int tilt;

    Vector<shape> list;

    list.next().draw(); // object is a rhombus
abstract class Shape

public abstract double getArea();
public abstract void draw();

public String toString();

return "hello";

Cannot create object of class Shape.

Shape s = new Shape();
Shape s; // ok

Shape t = new Shape(); // won't compile

s = new Circle();

t = new Shape();

.
public abstract class Shape {
    private Color c;
    public Color getColor() {
        return c;
    }
    public abstract double area();
}
public abstract class MessyShape extends Shape;

Shape =>

MessyShape =>

Messy Shape
Single Inheritance

In Java, every class has exactly one parent.
Employees' full-time vs part-time

Classes?
Solution: interfaces