Class 16: Casting Wrappers Containers
Casting
Casting (In General)

- Treating one type of data as another.

- How specified:
  - Implicit
  - Explicit

- What happens:
  - Change representation.
  - Treat representation differently.
Implicit Casting in Java

- Nothing specified
- Examples:
  - Copying int into double.
  - “Mixed-mode” arithmetic.
- Only for primitive types in the language.
- In Java, need an explicit cast for casting to a narrower type (e.g., from double to int)
Explicit Casting

- Specify you want casting.
- Example:
  ```
  float x, y; int i;
  x = (float) i; // explicit cast not necessary
  i = (int) y;  // could lose information
  ```
Object-Oriented Casting

- Upcast
- Downcast
Upcast

z Treat object as if it were an instance of a superclass.
  y E.g. put reference into a variable.

z Happens implicitly when needed

z Remember that the object keeps its type.

z Skipping levels in the hierarchy is OK.
Upcast Example

class Instrument {
    public static void play (Instrument i) {...}}

public class Wind extends Instrument { ... }

in main ...

Wind flute = new Wind();
Instrument.play(flute); // upcasting
Downcast

Inform compiler that the object referred to is an instance of a subclass.

Must be explicit. Why?

Remember that the object keeps its type.

Skipping levels in the hierarchy is OK.
Downcast Example

Object[] shapeList = {new Circle(4), new Square(6)};
for (int i = 0; i < shapeList.length; i++)
    ((Shape)shapeList[i].draw(); // must cast
Changing An Object’s Type

1 In Java, you cannot.
2 In Java, cannot cast outside the same branch of the hierarchy.
3 If you need to “convert” an object of type X into an object of type Y, write a toY() method in class X which instantiates a new object of type Y with appropriate values.
Wrappers
Wrapper Classes
For Primitive Types

z Sometimes you need an object which just holds a primitive.

z Why?
  y A method requires a parameter of type Object.
    x E.g. many container classes.
  y Need a place to define utility functions.
Wrapper Examples

- Subclasses of standard class Number
  - Integer, Long, Short, Byte, Float, Double
- Others
  - Character
  - Boolean
Things in the Wrapper Classes

- Defined constants
  - MAX_VALUE
  - MIN_VALUE

- Methods
  - Conversions (e.g. toString())
  - Comparisons
  - Parsing
**Parsing Methods in Wrapper Classes**

- `public static int parseInt(String s) throws NumberFormatException.`
  - Converts String to integer.
  - Throws NumberFormatException if the String is not a valid integer.
  - Can also specify radix.
- Analogous methods for other wrapper classes.
Review of Arrays
Review of Arrays

- Arrays are objects.
- Arrays hold primitives or references to other objects.
- Fixed size.
Use of Arrays

Create like this:

```java
Weeble[] a = new Weeble[2];
```

This creates a new array (object) which can hold 2 Weebles (objects), and stores the reference to this array in the variable `a`.

Access like this:

```java
a[0] = new Weeble();
System.out.println(a[0]);
```

Z
Usefulness of Arrays

- Efficient for access and update.
- Remembers position.
- OK to have empty elements.
- But: Size is fixed.
Containers
Containers

z Objects which hold objects
  y What they hold are of class Object.
  y Not any specific subclass.

z Categories
  y Collection
    x Set
    x List
  y Map
Containers

In Java 1.3, these are Interfaces.

- Need to actually use some of the classes which implement these interfaces.
- Or create your own.

Containers were completely revamped in Java 2 (as implemented in JDK 1.2).

- Old ways, e.g. Vector, are de-emphasized.
Collections vs. Maps

- Collections hold individual elements.
- Maps hold pairs.
  - Key and value.
Collections

- Hold individual elements

Types (subinterfaces):
  - A **Set** cannot have duplicate elements
  - A **List** holds elements in a particular sequence.

- Prints in square brackets, separated by commas.
Maps

A **Map** holds a group of **pairs** of objects.

Each pair is a key and a value.

a.k.a. associative array.

Acts like a simple database.

No duplicate keys.

Prints in curly braces, with pairs shown as key=value.
Next Time

- More on Containers
- Enumerations & Iterators