

---

# Classes in Java

CSE 142, Summer 2003  
Computer Programming 1

<http://www.cs.washington.edu/education/courses/142/03su/>

---

## Readings and References

- Reading
  - » Chapter 3, *Intro to Programming and Object-Oriented Design Using Java*, Niño and Hosch

---

## Outline for Today

- Review of objects and classes
- Acrobat class design
- Class definitions in Java
- Specifications and implementations
- Specifying methods in Java

---

## Classes Reviewed

- The basic unit of programming in Java is a *class definition*
  - » The *class* specifies properties and responsibilities
  - » Individual *objects* are created as needed
  - » All objects of the same class have the same list of properties and responsibilities
  - » Properties can contain simple values or be references to other objects
- Every object is an *instance* of some class
- Each class defines a new *type*

## Objects Reviewed

- Objects have the properties and responsibilities of their class
- Properties
  - » Sets of *values* that have a specific *type* (simple or reference to an object type)
  - » The current collection of property values is the object's *state*
- Responsibilities
  - » *Queries* and *commands*, the behavior of the object

## Recall the specification of an Acrobat

### You are an Acrobat

When you are asked to **Clap**, you will be given a number.  
Clap your hands that many times.

When you are asked to **Twirl**, you will be given a number.  
Turn completely around that many times.

When you are asked to **Count**, announce how many actions you have performed. This is the sum of the numbers you have been given to date.

## What does it mean to be an Acrobat?

- What are the properties?
- What are the responsibilities?
  - » Commands
  - » Queries

## A Java class definition

A class definition identifies the name of the class and provides its implementation

```
/**
 * This class can be used to represent a member of the CSE 142 Acrobat
 * community. In this simple implementation, such people have a name and
 * a cumulative action count and they know how to twirl, clap, and count.
 * @author Doug Johnson, for CSE 142 Su03
 */
public class Acrobat {

    // the properties and responsibilities will go here

}
```

## The implementation of a class

- Between the braces { ... } we give details of the implementation
  - » the properties of an object from this class
    - properties are stored in *instance variables*
  - » the responsibilities of an object from this class
    - implemented by the *methods* (and *constructors*) of the class which are sequences of Java code that carry out the object's responsibilities

## Identifiers – Names of Things

- Names in Java are called *identifiers*
  - » Combination of letters, digits, underscores "\_" starting with a letter. (don't use \$)
  - » Case sensitive (abc, Abc, ABC are all different)
- There are many uses for identifiers in Java

```
public class Acrobat { ... }
private int actionCount;
public void twirl(int k) { ... }
```
- Identifiers may not be a *keyword* or reserved word that has a special meaning in Java  
`class, public, if, for, int, double, boolean, ...`

## Choosing Names

- For names of classes and properties, use noun phrase that describes instances of the class or the property  
`Acrobat, actionCount, givenName`
  - » Avoid cryptic, cute, or vague names  
`cc, ccc, junk, moreJunk, value`
- For methods, use verb phrase that describes action  
`twirl, setBalance, changeDate`
- Capitalization – Java convention
  - » Class names are capitalized: `Acrobat, UWPerson`
  - » Instance variables and methods begin with lower case letter: `familyName, clap`

## Comments

- Java compiler: will follow whatever instructions you have written, right or wrong
- Another programmer: may or may not be able to understand what you have written
- Good comments are very useful to the next human to read your code
  - » succinct statements of basic information
  - » javadoc is a tool that generates documentation based on the comments you write in the code  
`c:\ex5> javadoc -d doc *.java`

## Comments

- Kinds of comments

```
// the entire rest of the line is a comment
/* everything is a comment until reaching this */
/** special comment form for documentation */
```

- Good commenting is an art

- » Include essential information, but don't overdo it
- » The program code itself should make sense with well selected names and logical design

## Specification vs Implementation

- Specification – view of the class as seen by *client* code that uses instances of the class
  - » the public methods and properties of the class
- Implementation – internal details
  - » Client should not know anything about this
- Some specifications in real life
  - » Automobile “user interface” – steering wheel, pedals, etc.
  - » Electric power outlet

## Specifying an Acrobat

- Class: Acrobat
- Queries
  - » getCount
  - » getGivenName
  - » getFamilyName
- Commands
  - » twirl
  - » clap
- A special “command” is the constructor – initialize new Acrobat instance when it is created

## Acrobat specification in Java

- In Java, the specification and implementation are given in a single file
- To create a class we start by writing the specification parts of methods
  - » i.e., the operations visible to client code
  - » comment using javadoc style comments, then use javadoc to create a snazzy set of web pages
- After specifying, we fill in the implementation details

## Specifying a Query method

```
/**
 * Tell the caller how many things we've done so far.
 * @return the number of claps and twirls to date
 */
public int getActionCount() {
    ...
}
```

- **public**
  - » defines this as part of the public specification
- **int**
  - » defines the type of the value returned by this query
- **getActionCount**
  - » the name of the method

## Specifying a Command Method

```
/**
 * Clap as instructed.
 * @param k the number of times to clap
 */
public void clap(int k) {
    ...
}
```

- **public**
  - » defines this as part of the public specification
- **void**
  - » this particular command does not return a value (some do)
- **clap**
  - » the name of the method
- **int k**
  - » the type and name of the parameter that is supplied to clap

## Constructor initializes a new object

```
/**
 * Create a new Acrobat using the name information provided.
 * @param given the specific name of this person
 * @param family the surname or family name of this person
 */
public Acrobat(String given,String family) {
    ...
}
```

- **public**
  - » defines this as part of the public specification
- **Acrobat**
  - » name of the constructor is the same as name of the class
- **String given, String family**
  - » the type and name of the parameters supplied when the new Acrobat object is created

## Summary

- Class definitions are the unit of programming in Java
  - » Individual objects are instances of these classes
- Specification vs Implementation
  - » What is publicly available to client code vs what is private information hidden inside the class
- Specifications for class methods
  - » Queries and Commands
  - » Constructors – a specialized kind of command