Method

A method is a named, parameterized group of statements

```
modifiers return-type method-name ( parameter-list ) {
    statement-list
}
```

*return-type void* means nothing is returned from the method

There must be a *return* statement, unless return-type is *void*

Calling a static method ...

... that is in a different class:

```
classname.methodname(...)
```

Examples:

```
Math.random()
Math.pow(2.5,2)
```

... that is in the same class:

```
methodname(...)
```

Our class *Prelim2Q1a* has a *static* method *randInt*, so an example method call within the class can be

```
randInt(3,8)
```
import javax.swing.*;

public class MakeFrame {
    public static void main(String[] args) {
        JFrame f = new JFrame();
        f.show();
        f.setSize(600,200);
        int w = f.getWidth();
        System.out.println("Width is " + w);
    }
}

Pre-defined class JFrame

- Deals with windows (frames) on the monitor
- All the predefined classes are collectively called the Java API
- Classes are grouped into packages. E.g., java.io, java.net, javax.swing
- Use the import statement:
  ```java
  import javax.swing.*;
  ```
- To find out what the classes do, read the API specifications:
  ```java
  http://java.sun.com/j2se/1.5.0/docs/api
  ```

Object & Class—an analogy

- **Object**: a folder that stores information (data and instructions)
- **Class**: a drawer in a filing cabinet that holds folders of the same type

What is in an object?
(What is in a folder?)

- **Fields**: to store data
- **Instance Methods**: for dealing with the object

Creating an object

- The expression `new JFrame()`
- Creates a JFrame object (folder) and gives it a reference name
- Calls method JFrame() to set initial values for the object
- Yields the reference of the object

Reference variable

- Use a reference variable to hold on to an object:
  ```java
  JFrame f = new JFrame();
  ```
- Use the class name as a type

October 20, 2005 Lecture 16