Methods, Inheritance, overriding and GUI

Object-oriented programming

Method

- A **method** is a named sequence of statements that performed some task.
  - takes in a *(perhaps empty)* collection of information
  - optionally sending back a piece of information

```java
void methodName(list of arguments){
    list of statements
}
```

```java
returnTypeNamE methodName(list of arguments){
    list of statements
}
```
Call a method

**objectName**.MethodName**(argument list)**

```java
import java.applet.*;
import java.awt.*;
public class NoPaint extends Applet
{
    Button btn = new Button("Click me");
    public void init()
    {
        // this method will be called when the applet is first loaded
        add(btn);
    }
}
```

API

- [http://java.sun.com/j2se/1.4/docs/api/](http://java.sun.com/j2se/1.4/docs/api/)
- `java.awt` Class `Component`
  - `java.lang.Object`
  - `java.awt.Component`
  - All Implemented Interfaces:
    - `ImageObserver`, `MenuContainer`, `Serializable`
  - Direct Known Subclasses:
    - `Button`, `Canvas`, `Checkbox`, `Choice`, `Container`, `Label`, `List`, `Scrollbar`, `TextComponent`
- public abstract class `Component` extends `Object`
- A `component` is an object having a graphical representation that can be displayed on the screen and that can interact with the user. Examples of components are the buttons, checkboxes, and scrollbars of a typical graphical user interface.
Paint in component

- public void paint(Graphics g)
  - Paints this component.
  - This method is called when the contents of the component should be painted in response to the component first being shown or damage needing repair. The clip rectangle in the Graphics parameter will be set to the area which needs to be painted. Subclasses of Component that override this method need not call super.paint(g).
  - For performance reasons, Components with zero width or height aren’t considered to need painting when they are first shown, and also aren’t considered to need repair.

- Parameters:
  - g - the graphics context to use for painting

- Since:
  - JDK1.0

- See Also:
  - update(java.awt.Graphics)

Graphical Programming

- Coordinate System

(0,0) x

(24,3)

(13,15)

(13,21)
Graphics Class

- **void clearRect(int x, int y, int width, int height)**
  Clears the specified rectangle by filling it with the background color of the current drawing surface.

- **void draw3DRect(int x, int y, int width, int height, boolean raised)**
  Draws a 3-D highlighted outline of the specified rectangle.

- **void drawLine(int x1, int y1, int x2, int y2)**
  Draws a line, using the current color, between the points (x1, y1) and (x2, y2) in this graphics context's coordinate system.

- **void drawArc(int x, int y, int width, int height, int startAngle, int arcAngle)**
  Draws the outline of a circular or elliptical arc covering the specified rectangle.

- **void drawOval(int x, int y, int width, int height)**
  Draws the outline of an oval.

- **void drawRect(int x, int y, int width, int height)**
  Draws the outline of the specified rectangle.

- **void drawString(String str, int x, int y)**
  Draws the text given by the specified string, using this graphics context's current font and color.

Graphics Classes (Cont.)

- **void fill3DRect(int x, int y, int width, int height, boolean raised)**
  Paints a 3-D highlighted rectangle filled with the current color.

- **void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle)**
  Fills a circular or elliptical arc covering the specified rectangle.

- **void fillOval(int x, int y, int width, int height)**
  Fills an oval bounded by the specified rectangle with the current color.

- **void fillRect(int x, int y, int width, int height)**
  Fills the specified rectangle.

- **void fillRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)**
  Fills the specified rounded corner rectangle with the current color.
Color class

`java.lang.Object`

`|--java.awt.Color`

static color: black, red, blue, cyan, gray, darkGray, green, lightGray, magenta, orange, pink, white, yellow

- `Color.green`
- `Color(int r, int g, int b)`
  Creates an opaque sRGB color with the specified red, green, and blue values in the range (0 – 255).

Font Class

`java.lang.Object`

`|--java.awt.Font`

- **Style**: BOLD, ITALIC, PLAIN
- **Name**: SansSerif, Serif, Monospaced, Dialog, DialogInput

```java
g.setFont(new("SansSerif", Font.BOLD, 12));
g.drawString("This is Helvetica, bold-12", 10, 20)
```
```java
import java.applet.*;
import java.awt.*;

public class Wormhole extends Applet{

    int X0 = 50;
    Y0 = 50;
    X0 = 200;
    Y0 = 200;
    X_INC = 50;
    Y_INC = 10;
    REL_INC = 10;
    Blue_INC = 20;

    public void paint(Graphics g){

        g.setColor(Color.black);
        g.fillRect(0, 0, 800, 400);
        for(int i=0; i<10; i++){
            g.setColor(new Color(1*X_INC, 1*Y_INC, (2-1)*BLUE_INC));
            g.fillOval(X0+i*X_INC, Y0+i*Y_INC, 90-1*i*REL_INC, 90-1*i*REL_INC);
        }

        g.setColor(Color.blue);
        g.setFont(new Font("TimesRoman", Font.BOLD, 36));
        g.drawString("On to Cydonia", 20, 250);

    }
}
```

---

**HTML for Wormhole**

```html
<HTML>
<HEAD>
<TITLE>
Shelley's #2 test applet
</TITLE>
</HEAD>

<BODY>
Here Shelley's applet #2 coming ...

<HR>
<APPLET code = "Wormhole.class"
width = 800
height = 400>
</APPLET>

<HR>
</BODY>
</HTML>
```

Dr Xiaojin Zhang, CIS, UMassD  
9/2/2002
On to Cydonia