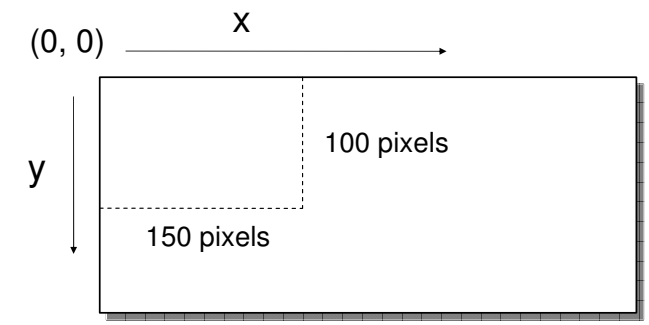
CS133: Developing Programming Principles

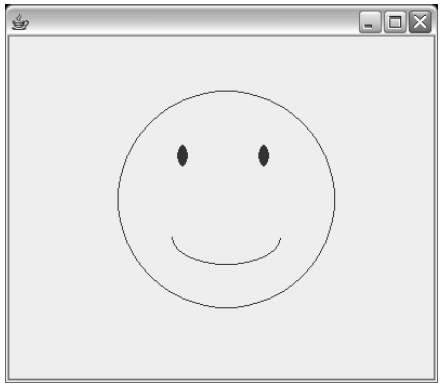
Lecture 19

Graphics: `JPanel`s, `JFrame`s, drawing primitives, the `Board` Class (1)

Screen Coordinates



Example



Example

```
import javax.swing.*;
import java.awt.*;

public class HappyPanel extends JPanel
{
  public void paintComponent(Graphics canvas)
  {
    canvas.drawOval(100,50,200,200);
    canvas.fillOval(155,100,10,20);
    canvas.fillOval(230,100,10,20);
    canvas.drawArc(150,160,100,50,180,180);
  }
}
```

Swing Library

Swing is a special class library for creating graphical user interface (GUI) components, including windows, buttons, text, and mouse support.

note the 'x'

```
import javax.swing.*;
import java.awt.*;
```


Class **`JPanel`**

A **`JPanel`** is a simple GUI component that provides a surface for drawing, as well as a container to group together other GUI components, such as buttons, labels or even other **`JPanel`**s

```
public class HappyPanel extends JPanel
```

Class **`Graphics`**

The **`Graphics`** class provides a simple interface that allows the application to draw onto **`JPanel`**s and other components.

```
public void paintComponent(Graphics canvas)
```


`Graphics` Methods

```
drawOval(int x, int y,
         int width, int height)

fillOval(int x, int y,
         int width, int height)

drawArc(int x, int y,
        int width, int height,
        int angle, int degrees)
```

Getting Panels on the Screen: `JFrame`

```
import javax.swing.*;
import java.awt.*;
public class HappyFrame extends JFrame
{
  public HappyFrame()
  {
    HappyPanel happyPanel = new HappyPanel();
    getContentPane().add(happyPanel);
    setSize(400, 350);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setVisible(true);
  }
}
```


Class `JFrame`

To appear onscreen, every component must be contained in a **`JFrame`** or other top-level ***container class***.

```
public class HappyFrame extends JFrame
```

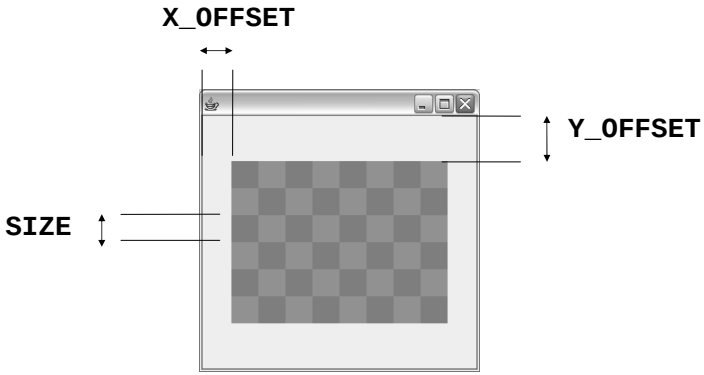
Example

```
public class TestHappy
{
  public static void main(String[] args)
  {
    HappyFrame happy = new HappyFrame();
  }
}
```


Simple Version of `Board`

```
public class BoardPanel extends JPanel
{

  private int rows, cols;
  private boolean[][] grid;

  // etc.

}
```

Simple Version of **Board**



Constructor

```
public BoardPanel(int rows, int cols)
{
    int xSize = 2*X_OFFSET + cols*SIZE;
    int ySize = 2*Y_OFFSET + rows*SIZE;
    this.rows = rows;
    this.cols = cols;
    emptyGrid();
    setPreferredSize(
            new Dimension(xSize, ySize));
}
```

Initializing the grid

```
private void emptyGrid()
{
    grid = new boolean[rows][cols];

    for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
        grid[i][j] = false;
      }
    }
}
```

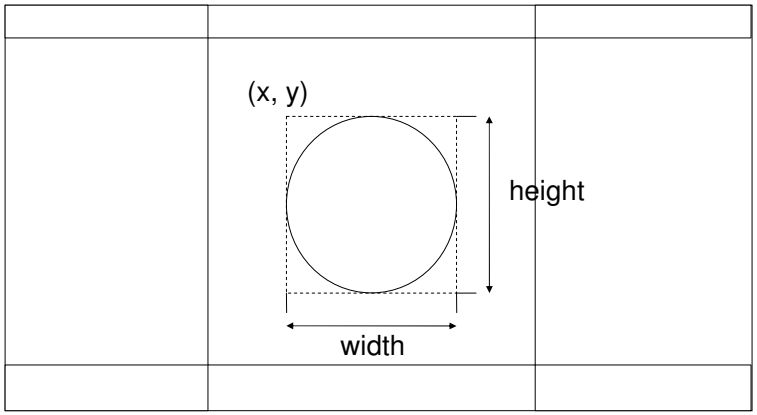

Drawing the Board

```
public void paintComponent(Graphic canvas)
{
    paintGrid(canvas);
    paintPegs(canvas);
}
```

Drawing the Grid

```
private void paintGrid(Graphics canvas)
{
  for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
      if ((i+j)%2 == 1) {
        canvas.setColor(LIGHT);
      } else {
        canvas.setColor(DARK);
      }
      canvas.fillRect(X_OFFSET + SIZE*j,
                      Y_OFFSET + SIZE*i, SIZE, SIZE);
    }
  }
}
```


Drawing a Peg



Drawing the Pegs

```
private void paintPegs(Graphics canvas)
{
  for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
      if (grid[i][j]) {
        int x = X_OFFSET + j*SIZE + SIZE/4;
        int y = Y_OFFSET + i*SIZE + SIZE/4;
        int size = SIZE/2;
        canvas.setColor(Color.YELLOW);
        canvas.fillOval(x, y, size, size);
      }
    }
  }
}
```


Adding Pegs

```
public void setPeg(int row, int col, boolean on)
{
  if (row >= 0 && row < rows &&
      cols >= 0 && col < cols) {
    grid[row][col] = on;
  }

}
```

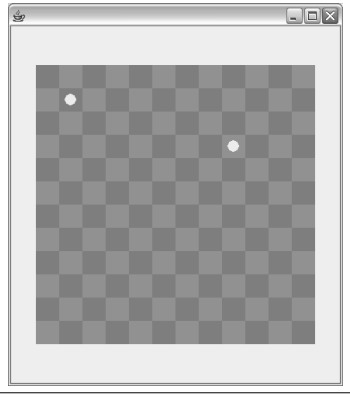
Board extends JFrame

```
public class Board extends JFrame
{
  private BoardPanel board;

  public Board (int rows, int cols) {
    Container content = getContentPane();
    board = new BoardPanel(rows, cols);
    content.add(board);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    pack();
    setVisible(true);
  }
  public void setPeg(int row, int col, boolean on)
  {
    board.setPeg(row, col, on);
  }
}
```


Example

```
public class TestBoard
{
  public static void main(String[] args)
  {
    Board board = new Board(12, 12);

    board.setPeg(1, 1, true);
    board.setPeg(3, 8, true);
  }
}
```

The Result



ERROR: undefined
OFFENDING COMMAND:

STACK: