Writing User Interfaces
Your First Sip of Java
Sept 11, 2019

Announcements
- Congrats on surviving lab 1!
- You should now have:
  - 2 course packets
  - Completed lab 1 (As long as you pressed the commit/push button in BlueJ, your lab has been submitted)
- Lab due dates:
  - Mon aft - Wed @ 11pm
  - Mon evening - Thur @ 5pm
  - Tue aft - Thur @ 11pm
- Homework 1 online: Due in class on Monday
- PLEASE USE YOUR ANONYMOUS ID (not your name)
- Office hours (for me, Tom, and TAs) have have started
- Questions about Lab 1?

Java in a Nutshell
Make things that do stuff!
(with other things)
(with themselves)

A World of Things
- Things: Perform tasks & store information
- Object (Thing): Instance of a type or class
- Class: Describes a family of objects
- Objects modify their internal state
- Objects interact with other objects
Java Program Structure

Classes, Objects, Methods

- Java lets programmer define **classes** of **objects**
- **Classes** describe the **data** and **functionality** of **objects**
  - **Data**: Numbers, strings, programmer-defined objects
  - **Functionality**: **Methods** that access or modify **data** in an **object**
- A Java program creates **objects** and manipulates them by invoking their **methods**

This Is Not A Java Program

(Classes, Objects, Methods, Data)

- **Create** a bank account **T** containing $0
- Credit $400 to bank account **T**
- Create a bank account **B** containing $500
- Credit $300 to bank account **T**
- Transfer $700 from **T** to **B**
- Debit $1200 from bank account **B**
- Close bank account **B**
- Send alert “Account Empty!” to bank account **T**

Neither is This...

…but it’s close!

(Classes, Objects, Methods, Data)

- T = new BankAccount();
- T.credit(400);
- B = new BankAccount(500);
- T.credit(300);
- B.transferFrom(T, 700);
- B.debit(1200);
- B.close();
- T.alert(“Account Empty!”);

A Simple Example : Hello

```java
public class Hello {
    public Hello() {
        System.out.println("Hello!");
    }
}
```

Note: Even this simple class uses other classes!
A Simple GUI: EmptyWindow

- GUI = graphical user interface
- Define class EmptyWindow
- Describe how to make an EmptyWindow
- Give the window dimensions
- GUls are complex: We need help!

EmptyWindow in Java

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

public class EmptyWindow extends GUIManager {
    private final int WINDOW_WIDTH = 170;
    private final int WINDOW_HEIGHT = 300;

    public EmptyWindow() {
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
    }
}
```

EmptyWindow Explained

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

public class EmptyWindow extends GUIManager {
    private final int WINDOW_WIDTH = 170;
    private final int WINDOW_HEIGHT = 300;

    public EmptyWindow() {
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
    }
}
```

Your First Java Vocabulary

- Java language words and symbols
  - class, extends, final, import, int, private, public
  - ( ) { } .
- Class names
  - EmptyWindow
  - GUIManager: part of squint library, created by Tom
- Method names
  - EmptyWindow: a "special" name (same as class) - a constructor
  - createWindow: from the class GUIManager
- Variable names
  - Primitive types
    - WINDOW_WIDTH, WINDOW_HEIGHT are of type ‘int’
  - Objects
    - this: object of the class (EmptyWindow) being defined
- Constants
  - 170, 300
- Libraries of pre-defined classes
  - squint, javax.swing
Another Example: TouchyButton

Define class TouchyButton

- how to make a TouchyButton
  - create an empty window
  - display text “Click on the button below” in window
  - add a button labeled “Click here” to window
- how to respond to a button click in window
  - display text “That tickles” in window

Writing TouchyButton

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

public class TouchyButton extends GUIManager {
    private final int WINDOW_WIDTH = 170, WINDOW_HEIGHT = 300;

    public TouchyButton() {
        this.createWindow(WINDOW_WIDTH, WINDOW_HEIGHT);
        contentPane.add(new JLabel("Click on the button below"));
        contentPane.add(new JButton("Click Here"));
    }

    public void buttonClicked() {
        contentPane.add(new JLabel("That tickles!"));
    }
}
```

TouchyButton in Java

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

public class TouchyButton extends GUIManager {
    private final int WINDOW_WIDTH = 170, WINDOW_HEIGHT = 300;

    public TouchyButton() {
        this.createWindow(WINDOW_WIDTH, WINDOW_HEIGHT);
        contentPane.add(new JLabel("Click on the button below"));
        contentPane.add(new JButton("Click Here"));
    }

    public void buttonClicked() {
        contentPane.add(new JLabel("That tickles!"));
    }
}
```

Java language words and symbols

- class, extends, final, import, int, new, private, public, void
- () {} .

Class names

- TouchyButton
- GUIManager: created by Tom
- JLabel, JButton: from the Java “Swing” classes

Method names

- TouchyButton: a special name - a constructor
- buttonClicked

Objects

- contentPane: an object that is part of a GUIManager
- this: the object currently being defined

Primitive types

- WINDOW_WIDTH, WINDOW_HEIGHT

Constants

- 170, 300, "Click on the button below", "Click Here", "That tickles!"

Libraries of pre-defined classes

- squint, javax.swing
Summary

- Java programs are composed of classes (one or more)
  - Classes describe types of objects
  - Objects have state and functionality
  - Objects are accessed/changed through method invocation
- Key words (English)
  - class, object, method, instance variable, construction, constructor, constant
- Key words (Java)
  - class, extends, final, import, int, new, private, public, void

Types of Entities in Java

- Java language words and symbols
- Class names
- Method names
  - Always includes a constructor method with same name as class
- Variable names
  - Objects
  - Primitive types
- Constants
- Libraries of pre-defined classes

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

public class Class-name extends GUIManager {
  . . . instance variable declarations . . .
  public Class-name () {
    . . . constructor's body . . .
  }
  public void some-method-name( ) {
    . . . some-method's body . . .
  }
  public void some-other-method-name( ) {
    . . . some-other-method's body . . .
  }
  . . .
}
```
import squint.*;
import javax.swing.*;

public class Class-name extends GUIManager {

    ... instance variable declarations ...
    public Class-name () {
        ... constructor's body ...
    }

    public void some-method-name() {
        ... some-method's body ...
    }

    public void some-other-method-name() {
        ... some-other-method's body ...
    }
}

public void some-method-name( ) {
    ... some-method's body ...
}

public void some-other-method-name( ) {
    ... some-other-method's body ...
}

More Examples

- ClickCounter
Creating objects with Constructions

```java
new JLabel( "Click on the button below" )
new JButton( "Click Here" )
new JTextField( 8 )
```

Constructions

```java
new JLabel( "Click on the button below" )
new JButton( "Click Here" )
new JTextField( 8 )
new type-of-thing-to-create( ... details ... )
```
contentPane.add( new JLabel( "Click on the button below" ) );

this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );

actor . thing-to-do ( ... details/parameters ... );
Mutator Method Invocation

```java
contentPane.add( new JLabel( "Click on the button below" ) );
this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
```

```
actor . thing-to-do ( ... details/parameters ... );
```

```
noun?   verb?   object/subordinate phrase?
```

Accessor Method Invocation

```java
contentPane.add( new JLabel( "Be kind to your " + firstNoun.getText() );
```

```
actor . thing-to-get ( ... details/parameters ... )
```

```
noun?   verb?   object/subordinate phrase?
```

Working with Variables

```java
private JTextField firstNoun;
private JTextField pluralNoun;
....
```

```java
firstNoun = new JTextField( 10 );
pluralNoun = new JTextField( 10 );
....
```

Instance Variable Declarations

```java
private JTextField noun1;
private int count;
private JLabel counterLabel;
```

```java
private type-of-thing-name-will-identify name;
```
Assignment Statements

```java
firstNoun = new JTextField( 10 );
pluralNoun = new JTextField( 10 );

variable-name = expression-describing-value;
```

Local Variables

```java
public LoginWithPanels() {
    this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );

    JPanel userNamePanel;
    JPanel passwordPanel;

    userNamePanel = new JPanel();
    userNamePanel.add( new JLabel( "Username:" ) );
    userNamePanel.add( new JTextField( 8 ) );
    contentPane.add( userNamePanel );

    ... 
}
```

Further Summary

- A Java class consists of
  - State: instance variables and constants hold object data
  - Functionality: methods that manipulate the object
  - Constructor(s): method(s) describing object creation
- Graphic User Interfaces (GUIs) in Java
  - AWT: classes developed by Java, superseded by
  - Swing: classes developed by Java, nicely wrapped by
  - Squint: classes developed by Tom

Swing and Squint

- Swing provides a family of GUI components
  - Buttons, menus, areas for entering/editing text, mouse interaction, …
  - Containers for sets of GUI components and layout managers for positioning container contents in windows
- Squint provides a package of classes to
  - Simplify use of Swing classes
  - Create and access network connections easily
  - Load and manipulate images, access web-cam, …
Coming Next

Network Communication in Java

More Examples

- ClickCounter
- ToDoList