GUI Programming

Graphical User Interface

- A **GUI** (graphical user interface) is a system of interactive visual components for computer software.
- Structure
  - Create the icons and widgets that are displayed to a user and organize them inside a screen window.
  - Define functions that will process user and application events.
  - Associate specific user events with specific functions.
  - Start an infinite event-loop that processes user events. When a user event happens, the event-loop calls the function associated with that event.

How does a GUI work?

- A GUI uses windows, icons, and menus to carry out commands, such as opening, deleting, and moving files. Although a GUI operating system is primarily navigated using a mouse, a keyboard can also be used via keyboard shortcuts or the arrow keys.

What are the benefits of GUI?

- Unlike a command-line operating system or CUI, like Unix or MS-DOS, GUI operating systems are much easier to learn and use because commands do not need to be memorized. Additionally, users do not need to know any programming languages. Because of their ease of use and more modern appearance, GUI operating systems have come to dominate today's market.

The Basic GUI Application

- There are two basic types of GUI program in Java: stand-alone applications and applets.
- A stand-alone application is a program that runs on its own, without depending on a Web browser.
- An applet is a program that runs in a rectangular area on a Web page.

JFrame and JPanel

- In a Java GUI program, each GUI component in the interface is represented by an object in the program.
- One of the most fundamental types of component is the window.
- Windows have many behaviors
- They can contain other GUI components such as buttons and menus.

Contd.

- A *JFrame* is an independent window that can, for example, act as the main window of an application. One of the most important things to understand is that a *JFrame* object comes with many of the behaviors of windows already programmed in.

Components and Layout

- Another way of using a *JPanel* is as a container to hold other components
- Java has many classes that define GUI components.
- Componenets must have some containers
  - content.add(displayPanel, BorderLayout.CENTER);
  - content.add(okButton, BorderLayout.SOUTH);
- content refers to an object of type *Jpanel*
- this panel becomes the content pane of the window

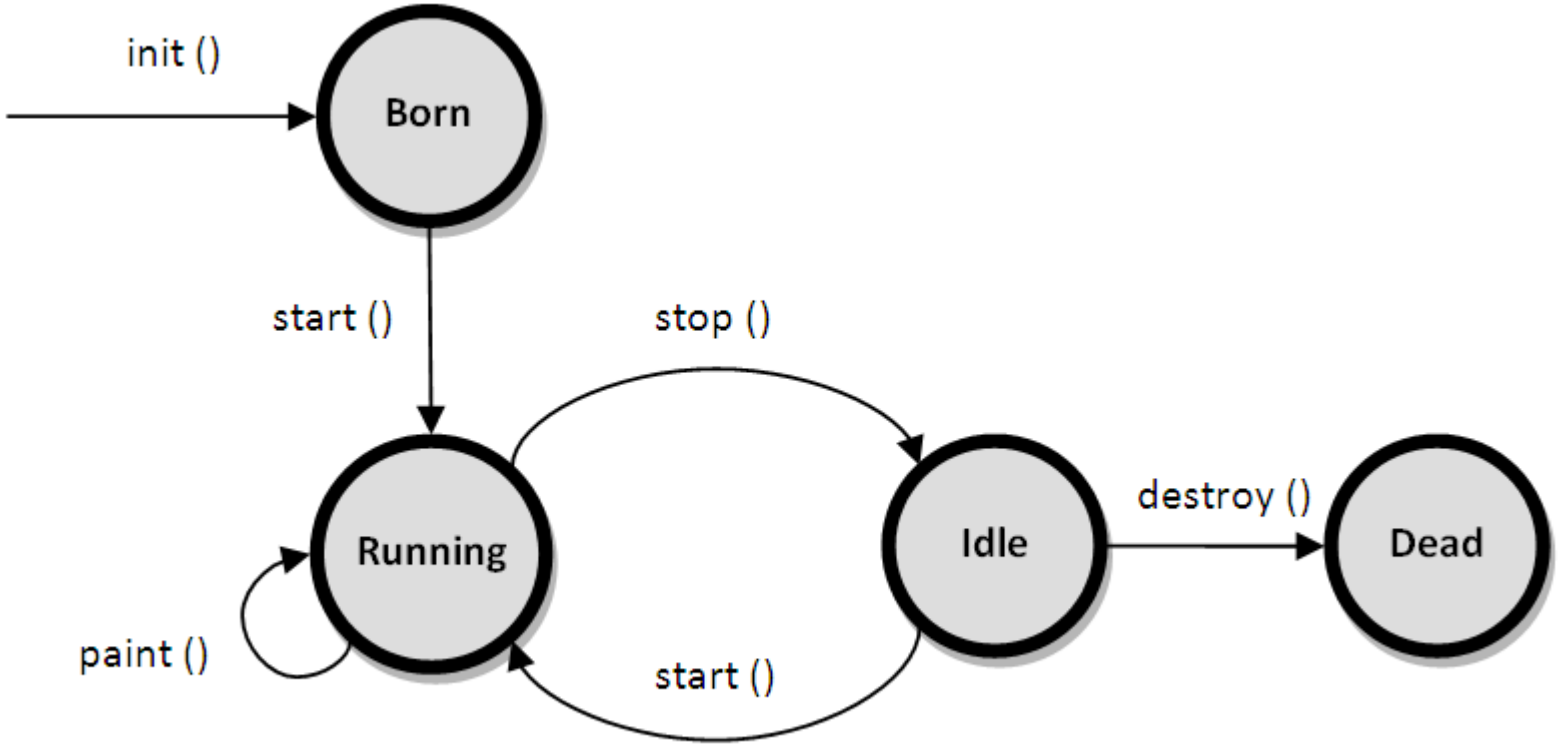
Events and Listeners

- The structure of containers and components sets up the physical appearance of a GUI, but it doesn't say anything about how the GUI behaves.
- GUIs are largely event-driven; When an event occurs, the program responds by executing an event-handling method.
- The most common technique for handling events in Java is to use event listeners. A listener is an object that includes one or more event-handling methods.

Applets and HTML

- Japplet - The *JApplet* class (in package javax.swing) can be used as a basis for writing applets in the same way that *JFrame* is used for writing stand-alone applications.
- To create an applet, you will write a subclass of *JApplet*. The *JApplet* class defines several instance methods that are unique to applets.

Contd.



Applets on Web Pages

- The <applet> tag can be used to add a Java applet to a Web page.
  - E.g. <applet code="HelloWorldApplet.class" height=100 width=250> </applet>
- Applets can use applet parameters to customize their behavior.
- The value what will be returned is always a string.

Graphics and Painting

- The Java API includes a range of classes and methods that are devoted to drawing.
- The physical structure of a GUI is built of components. The term component refers to a visual element in a GUI, including buttons, menus, text-input boxes, scroll bars, check boxes, and so on.
- Java, GUI components are represented by objects belonging to subclasses of the class java.awt.Component
- Most components comes under the Swing GUI.

Contd.

- A JPanel, like any JComponent, draws its content in the method
  - public void paintComponent(Graphics g)
  - the paintComponent() method has a parameter of type *Graphics*.
  - The *Graphics* object will be provided by the system when it calls the written method
  - To do any drawing at all in Java, we need a graphics context.
  - A graphics context is an object belonging to the class java.awt.Graphics.
  - *Graphics* object can draw to only one location.
  - Co-ordinates
  - Color
  - Fonts
  - Shapes
  - Graphics 2D

Mouse Events

- In Java, events are represented as objects.
- When an event occurs, the system collects all the information relevant to the event and constructs an object to contain that information.
- For example, when the user presses one of the buttons on a mouse, an object belonging to a class called *MouseEvent* is constructed.
- When the user presses a key on the keyboard, a *KeyEvent* is created.

Event Handling

- For an event to have any effect, a program must detect the event and react to it. In order to detect an event, the program must "listen" for it. Listening for events is something that is done by an object called an event listener. An event listener object must contain instance methods for handling the events for which it listens. For example, if an object is to serve as a listener for events of type *MouseEvent.*
- The body of the method defines how the object responds when it is notified that a mouse button has been pressed. The parameter, evt, contains information about the event.
- The methods that are required in a mouse event listener are specified in an interface named *MouseListener.*
- Mouse Coordinates.
  - *MouseEvent*
- MouseMotionListeners and Dragging
  - public void mouseDragged(MouseEvent evt);
    public void mouseMoved(MouseEvent evt);

Timers, KeyEvents, and State Machines

- A *Timer* generates events at regular intervals.
  - class javax.swing.Timer.
- Timers and Animation
  - the *ActionListener* interface.
  - timer = new Timer(millisDelay, listener);
- Keyboard Events
  - KeyEvent
- Focus Events
  - In Java, objects are notified about changes of input focus by events of type *FocusEvent*. An object that wants to be notified of changes in focus can implement the *FocusListener* interface.
- State Machines
  - In computer science, there is the idea of a state machine, which is just something that has a state and can change state in response to events or inputs. The response of a state machine to an event or input depends on what state it's in. An object is a kind of state machine.

Basic Components

- API are defined by subclasses of the class *JComponent*, which is itself a subclass of *Component*.
- Jbutton
  - Constructors
  - Events
  - Listeners
  - Registration of Listeners
  - Event methods
  - Component methods
- Jlabel
- JCheckBox
- JTextField and JTextArea
- JComboBox
- Jslider

Basic Layout

- Basic Layout Managers
  - *FlowLayout*, *BorderLayout*, and *GridLayout*.
- A *FlowLayout* simply lines up components in a row across the container.
- The default layout for a *JPanel* is a *FlowLayout*;

Menus and Dialogs

- Menus and Menu bars
- Dialogs
- Fine Points of Frames
- Creating Jar Files