Understanding User Interfaces

User Interfaces:

- User interfaces allow users to interact with applications.
- They can take input from users and provide visual or audible feedback.
- Real-world interfaces include things like ATMs and bank tellers.
- Virtual interfaces include software menus, buttons, and text fields on apps and websites.

Model

Model:

- Represents the business logic of an application.
- Consists of classes that handle the underlying system processes.
- Developed separately from the user interface.
- Should not assume any knowledge of the user interface (e.g., no print statements for debugging).

User Interface

User Interface:

- The part of the application that interacts with the user.
- Uses model classes to process and display information.
- Changes in the model often reflect visually on the user interface for immediate feedback.

GUI (Graphical User Interface)

GUI:

- A GUI uses windows, buttons, text fields, and other components to interact with the user.

- It is preferred over text-based interfaces because it is more intuitive and user-friendly.
- GUIs are built from components known as controls or widgets.

Application

Application:

- A computer program with a graphical user interface.
- Interacts with users to perform tasks, obtain and visualize information, and potentially interact with the real world through sensors and hardware.

Window Component

Window Component:

- A visual element placed on a window, such as a button or text field.
- Allows the user to interact with the application.

Container

Container:

- An object that holds components and/or other containers.
- Manages the layout and positioning of its child components.
- Examples in JavaFX include Pane, VBox, and HBox.

History of GUI with Java

History:

- **AWT (Abstract Window Toolkit):** Java's original GUI library.
- **Swing:** Added in Java SE 1.2, it became the primary GUI technology for Java.
- **JavaFX:** Announced by Sun Microsystems in 2007 as a modern GUI framework, replacing Swing.

What is JavaFX?

JavaFX:

- An open-source, Java-based framework for developing rich client applications.
- Comparable to Adobe AIR and Microsoft Blazor.
- Provides a comprehensive API for GUI, graphics, and multimedia.

JavaFX vs Swing

JavaFX vs Swing:

- **Swing:** Only for GUIs; requires additional APIs for graphics and multimedia.
- **JavaFX:** One API for GUI, graphics, and multimedia; easier to use.
- **SwingNode:** Allows embedding Swing components in JavaFX applications.
- **JFXPanel:** Allows embedding JavaFX components in Swing applications.

JavaFX Scene Builder

Scene Builder:

- A visual layout tool for designing JavaFX GUIs.
- Allows dragging and dropping GUI components.
- Generates FXML code, an XML vocabulary for defining GUIs.

Basic Structure of JavaFX

JavaFX Structure:

- **Application:** The main class that extends `Application` and overrides the `start(Stage)` method.
- **Stage:** The main window of the application.
- **Scene:** Contains the visual elements (nodes) of the application.
- **Nodes:** Basic building blocks like buttons, text fields, and shapes.

Example: Basic Structure of JavaFX

Example Code:

java

```
Copy code
import import import
import public class JavaFXBasic extends Application public void
start Button btOK new Button "OK" Scene scene new Scene 200
250 "JavaFX Basic Structure"
public static void main
```

- **Explanation:**
  - Creates a button.
  - Adds the button to a scene.
  - Sets the scene on the stage.
  - Displays the stage.

Example: Display a Simple Shape

Example Code:

```
java
Copy code
import import import import
import import public class
ShowCircle extends Application public void start Circle circle
new Circle 100 100 50 Pane pane new
Pane Scene scene new Scene 200 200
"ShowCircle" public static
void main
```

- **Explanation:**
  - Creates a circle with specified properties.
  - Adds the circle to a pane.
  - Creates a scene with the pane.
  - Sets the scene on the stage and displays it.

Components of JavaFX

Components:

- **Stage:** The main window.
- **Scene:** Contains the GUI elements.
- **Window Component:** Visual elements like buttons and text fields.
- **Node:** Each visual element in the scene.

- **Container:** Holds and manages the layout of components.
- **Controls:** GUI components like labels, text fields, and buttons.
- **Event Handler:** Methods that respond to user interactions.

Welcome Application Example

Welcome Application:

1. Create a new JavaFX project in Eclipse.
2. Use Scene Builder to design the GUI.
3. Modify FXML file to define the layout.
4. Write event handlers in a controller class.
5. Run the application.

Example FXML:

xml
Copy code

```
        alignment  "CENTER" prefWidth  "450" prefHeight  "300"      text  "Welcome to JavaFX!"
style  "-fx-font-size: 36; -fx-font-weight: bold;"          fx:id  "imageView"
```

- **Explanation:**
  - Defines a VBox layout with a label and an image view.
  - Sets properties for alignment, size, and style.

Running the Application

Run the Application:

- Right-click `Main.java` and select "Run As" → "Java Application".

By following these steps and understanding these concepts, you can effectively develop JavaFX applications with a clear separation between the model and the user interface.