Adrianna L.Lambert   **October 14, 2024**

Module 6 Assignment

Module 6 Assignment: Java AWT and Concepts

1. What does it mean that Java AWT is platform-independent? Why do we desire such a trait?

Java AWT (Abstract Window Toolkit) is platform-independent because it uses Java's **Write Once, Run Anywhere** principle. AWT relies on the native GUI components of the operating system but abstracts this through the Java Virtual Machine (JVM).

Why it is desirable:

- Code written once works seamlessly across multiple platforms (Windows, macOS, Linux).

- Reduces development time and effort while ensuring broader compatibility.

---

2. What does it mean that Java runs on a virtual machine? How is that beneficial?

Java runs on a **Java Virtual Machine (JVM)**, which translates platform-independent bytecode into platform-specific machine code.

Benefits:

- **Platform independence**: Java programs can run on any system with a JVM.

- **Security**: JVM provides an isolated environment, reducing risks to the host system.

- **Portability**: Applications behave consistently across different platforms.

---

3. In every hierarchy map of Java, Object is always at the top. Why is this so?

In Java, Object is the **root class** of all classes. This means every Java class directly or indirectly inherits from the Object class.

Reason:

- Provides common methods (e.g., toString(), equals(), hashCode()) that all objects can use or override.

- Enables polymorphism and a unified structure for handling objects.

---

4. What is a Java package?
A Java package is a namespace that organizes related classes and interfaces into a single unit.

Purpose:

- Prevents naming conflicts.

- Promotes modularity and code reuse.

- Example: java.awt contains classes for GUI development.

---

5. What is an event in Java?
An event in Java represents an action or occurrence in a program (e.g., clicking a button, typing, or mouse movement).

Key components:

- **Event Source**: The component that generates the event (e.g., a button).

- **Event Listener**: The object that listens for and processes the event.