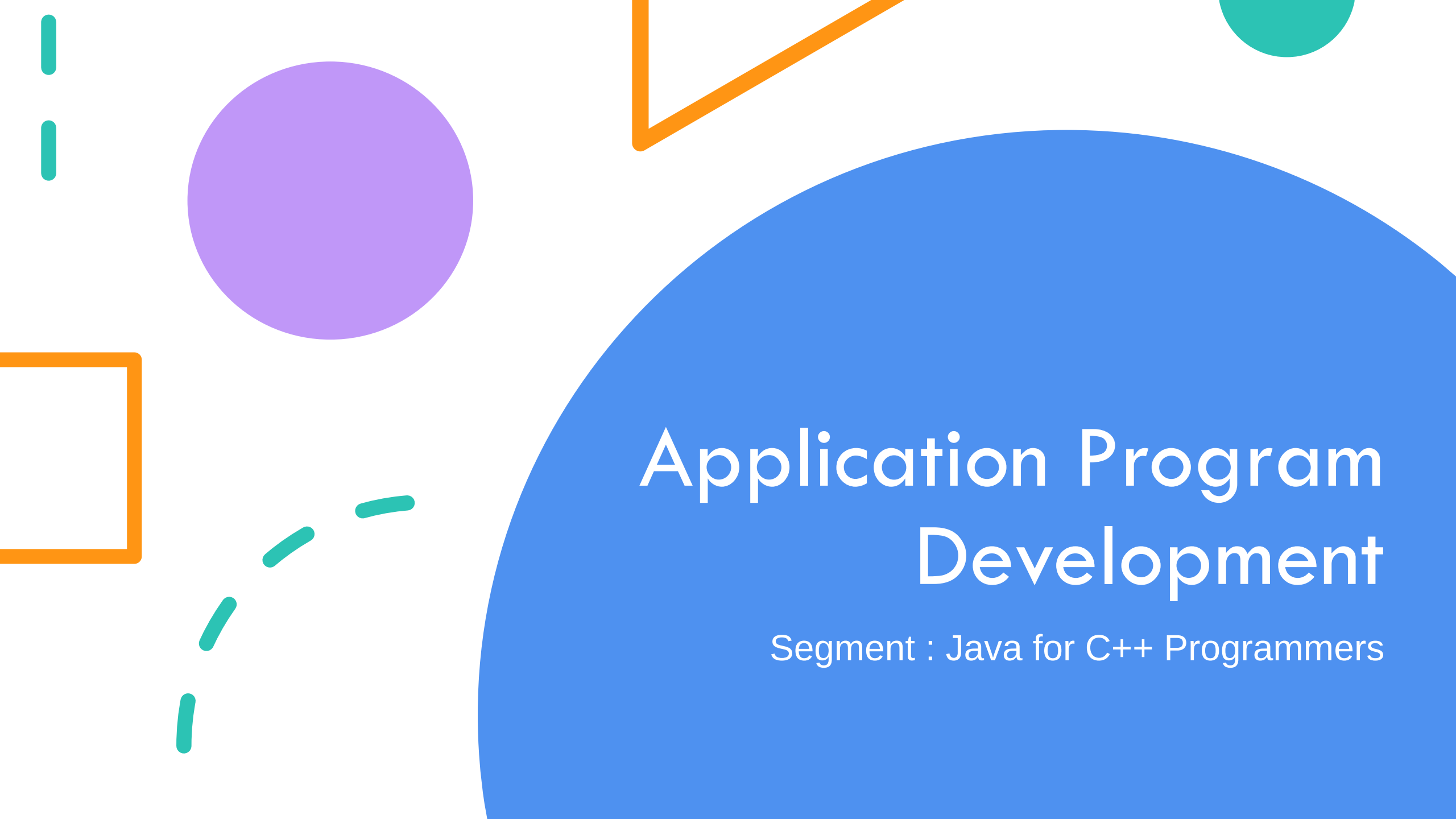
Application Program Development
Segment : Java for C++ Programmers

Agenda for Week 1

- Lecture
  - Java for C++ Programmers - Introduction
  - Similarities of Java and C++
  - Differences between Java and C++
  - MVC Pattern
  - Designing a Model – View – Controller to work together
- Lab
  - Installation process of Java, JavaFX and Scene Builder.

Outcomes

- Understanding of Java as Language.
- Understanding the Java Virtual Machine.
- Understanding the Java Garbage Collector.

What is Java?

- Java can be broadly defined as
  - A General Purpose → Its not constraint to one particular domain
  - Object Oriented → Helps model real world scenarios in more natural way
  - Platform Independent → Write Once Run Anywhere (WORA)
  - Concurrent → Multithreaded

What is Java Programming Language used for?

- Because Java is a free-to-use and a versatile language, it builds localized and distributed software. Some common uses of Java include:
- **Game Development**
  - Many popular mobile, computer, and video games are built in Java. Even modern games that integrate advanced technology like machine learning or virtual reality are built with Java technology.
- **Cloud computing**
  - WORA – Write Once and Run Anywhere, making it perfect for decentralized cloud-based applications. Cloud providers choose Java language to run programs on a wide range of underlying platforms.
- **Big Data**
  - Java is used for data processing engines that can work with complex data sets and massive amounts of real-time data.
- **Artificial Intelligence**
  - Java is a powerhouse of machine learning libraries. Its stability and speed make it perfect for artificial intelligence application development like natural language processing and deep learning.
- **Internet of Things**
  - Java has been used to program sensors and hardware in edge devices that can connect independently to the internet.

Why is Java such a Popular Language?

- Java is popular because it has been designed for ease of use. Some reasons developers continue to choose Java over other programming languages include:
- **High quality learning resources**
  - Java has been around for a long time, so many learning resources are available for new programmers. Detailed documentation, comprehensive books, and courses support developers through the learning curve. In addition, beginners can start writing code in Core Java before moving to Advanced Java.
- **Inbuilt functions and libraries**
  - When using Java, developers don’t need to write every new function from scratch. Instead, Java provides a rich ecosystem of in-built functions and libraries to develop a range of applications.
- **Active community support**
  - Java has many active users and a community that can support developers when they face coding challenges. The Java platform software is also maintained and updated regularly.

Why is Java such a Popular Language?

- **High-quality development tools**
  - Java offers various tools to support automated editing, debugging, testing, deployment, and change management. These tools make Java programming time and cost-efficient.
- **Platform Independent**
  - Java code can run on any underlying platform like Windows, Linux, iOS, or Android without rewriting. This makes it especially powerful in today's environment, where we want to run applications on multiple devices.
- **Security**
  - Users can download untrusted Java code over a network and run it in a secure environment in which it cannot do any harm. Untrusted code cannot infect the host system with a virus nor can it read or write files from the hard drive. The security levels and restrictions in Java are also highly configurable.

How does Java Works? The Platform

- All programming languages are a means to communicate with machines. Machine hardware only responds to electronic communication (010101).
- High-level programming languages like Java act as a bridge between human language and hardware language.
- To use Java, a developer needs to understand two things:
- **Java language and APIs**
  - This is the front-end communication between the developer and the Java platform.

- **Java Virtual Machine**
  - This is the back-end communication between the Java platform and the underlying hardware. Let’s look at each of these in detail below.

Few words about Java API.

- Java defines the syntax and semantics of the Java programming language. This includes the basic vocabulary and rules used to write algorithms such as primitive data types, if/else blocks, loops, etc.

- APIs are important software components bundled with the Java Platform. These are pre-written Java programs that can plug and play existing functionality into your own code. For example, you could use Java APIs to get the date and time, perform mathematical operations, or manipulate text.

- Any Java application code written by a developer will typically combine new and pre-existing code from Java APIs and Java libraries.

Few words about JVM (Java Virtual Machine).

- The Java Virtual Machine acts as an additional abstraction layer between the Java platform and the underlying machine hardware.

- Java source code can run only on those machines that have JVM installed on them.

- Why?

Compilers: The complete program is written in natural English-like syntax with compilers, and the language then compiles (or translates) the entire code into machine code. The compiled code is then run on the hardware.

Interpreters: With interpreters, every high-level code statement is interpreted into machine code on the fly. Written statements are run immediately by the hardware before looking at the next statement.

Java Runtime Environment

- The Java program was the first language to combine both methods (compiler & Interpreted) using a Java Virtual Machine (JVM).
- The Java code compiler is called the Java Virtual Machine.
- Any Java file is first compiled into bytecode.
- Java bytecode can only run in the JVM.
- The JVM then interprets the bytecode to run it on the underlying hardware platform.
- So, if the application is running on a Windows machine, the JVM will interpret it for Windows. But if it is running on an open-source platform like Linux, the JVM will interpret it for Linux.

Question Time...

Is C++ an Interpreted Language?

Yes

No

How to program in Java?

- The are 3 main java editions to chose from
- Java Standard Edition (JSE)
  - To develop client-side applications. (For Desktop Application Development)
- Java Enterprise Edition (JEE)
  - To develop server-side applications, like Java Servlets, Java Sever Pages, Java Server Faces.
- Java Micro Edition (JME)
  - To develop mobile based applications.

Note: We are going to use JSE for the course.

Steps to Use Java SE

- **Coding:** produced by the programmer.
- **Compiling:** build the program into bytecode
  - Result is a ".class" file.
  - Compiler command: javac e.g. javac HelloWorld.java results in HelloWorld.class
- **Running:** class file (bytecode) is loaded by the JVM.
  - JVM command: java e.g. java HelloWord results in Output of the program
- **Bytecode Verification:** The JVM verifies the class file digital signature. The JVM is simply an interpreter.
- **Internal Integrity Check:** It checks if loaded Java program is well formed. Data types are verified along with other syntax structure.
- **Execution:** Program execution begins from the main entry point.