welcome

GANESH

TOPICS :

- ➢JAVA INTRODUCTION
- ➢JAVA FEATURES
- ➢JAVA PROGRAMMING STRUCTURE

Java introduction:

- History
- Java is a High Level Programming language
- It is a pure Object Oriented Programming Language
- It is Platform independent programming language
- The main purpose of java application is develop applications software
- Extensive Library support
- Versatility
- Community and Ecosystem

WHY:

1. **History:**

- Java was created in the mid-1990s by a team at Sun Microsystems, led by James Gosling. It was designed to be platform-independent and has since become a foundational language for a wide range of applications, from web and mobile development to enterprise software

1. **Object-Oriented**:

- Java supports OOPS concepts
- emphasizing classes, objects, and the principles of encapsulation, inheritance, and polymorphism.

2.**High-level programming** :

- A high-level programming language, like Java, is a user-friendly way to write code that abstracts low-level details, making software development more accessible and efficient.

3.**Independent Language**:

- Java, is a programming language that allows software to run on different computer systems without modification. This is achieved through the use of a virtual machine, which interprets the code and adapts it to the specific platform, making it highly portable.

4. **Versatility:**

- It's used in web development (Java EE), mobile app development (Android), desktop applications, server-side programming, and even embedded systems.

JAVA FEATURES:

- Java features are the unique traits and tools within the Java programming language that make it useful and versatile for software development.
- These features include portability, object-oriented structure, built-in security, and a rich library, making Java well-suited for a wide range of applications.
- **Multi-Threading:**
- Java supports multithreading, enabling the concurrent execution of multiple threads within a single program. This is crucial for building responsive and efficient applications.
- **Architecture-Neutral:**
- Java is designed to be architecture-neutral, meaning that it can be used in various computing environments without modification, which is important for networked and distributed systems.
- **Backward Compatibility:**
- Java has a strong commitment to backward compatibility, meaning that applications written for older versions of Java can often run on newer versions with minimal or no modification.
- **Exception Handling:** Java has a robust exception-handling mechanism that enables developers to handle errors gracefully, improving the reliability of applications.
- **Documentation and Community Support:** Java has extensive documentation, tutorials, and a vast community of developers who are ready to help, making it easier for developers to learn and solve problems.
- **Robust and Secure:** Java's strong type-checking at compile-time and runtime, along with features like automatic memory management (garbage collection), contribute to the language's robustness. It also includes security features, such as a SecurityManager and sandboxing, to create secure execution environments.

Structure of java programming:

Syntax:

```
Import java.io.*;

Public Class Test{

Public static void main(string args []){

System.out.println( )

      }

}er
```

java	➔Super package
Io	➔sub package
*	➔collection of classes
Class	➔keyword
Test	➔identifier (user modifier)
Public	➔Accessing modifier
Static	➔keyword
Void	➔keyword(no return values)
Main	➔method
String	➔Class
Args	➔ variables(arrays)
System	➔ClassName
Out	➔object
println( )	➔function

Keypoints of structure:

- One keyword is one Statement
- Public keyword who will access the data or who are not access the data
- Class is nothing but a one container
- Every java coden can write with in the Class{ }
- we can use println function and also used to another way

- System.out.   println();
- System.out.   print();
- System.out.   printf();
- System.out.   printerror();

- Keyword: Keyword are reserved words given by peogramming language it self
  purpose and definition not change by programming and also 53 keywords

- Identifiers: An identifier is a name given to entities like class function variable interfaces
  .   packages it helps to different one entity from one onother

- Class: Class is a blueprint for creating objects. It defines the structure and behavior of those objects, including t .
  . their attributes (fields) and methods (functions).

- Object: Object is a real world entity or it is a instance of class
  ex: rose flower

- Package: A Java package is like a folder that helps organize and group related Java classes and interfaces in your
  . code, making it easier to manage and maintain your projects.

- Variables: Variables in Java are like containers for storing data. They have a name, a type, and a value, and they a
- . allow you to work with different types of information in your programs.

THANK YOU