Hello every one,

In this video we will learn about the internal structure of JAVA. What are the components available in java.

Cursor

This picture represents the internal Structure of Java.

On High level there is JDK and JDK contains JRE with devtools. And JRE contains JVM and Libraries.

Each container in this image has its own feature. Now we will see one by one.

Cursor

JVM:

JVM is known as Java Virtual machine.

In previous video we have seen than Java code is compiled into Byte code and using interpreter byte code is converted into machine code.

SO JVM is responsible for converting bytecode to machine understandable code.

Java Virtual Machine, or JVM, loads, verifies and executes Java bytecode. Java acts as an interpreter.

Cursor

Libraries:

Libraries are just a collection of classes which are predefined in JAVA. And these classes we get when we install the Java.

Examples for java libraries are

**java.** **lang, java.** **util, and java.** **math**

Cursor

JRE-

The Java Runtime Environment (JRE) is a software package which bundles the libraries (jars) and the Java Virtual Machine. With the help of libraries JVM will execute or run the java application.

Cursor

So to execute any java program, we need JRE to be installed in the machine in which code is executed. With out JRE we can not run the program.

Cursor

JDK-

In the image we saw that JDK contains JRE and development tools. In previous slide we came to know the usage of JRE.

Now what is the development tools.

Cursor

Development tools are used for developing, debugging Java applications. Till now we have see how java code is executed, but we didn’t discuss on how java code is written. So to write the java code, and generate the executables, we need this developing and debugging tools.

For example Compiler , jar creation are some of the tools available in development tools.

Compiler is used to compile the code and once code is written and compiles we will generate the jar so that it can be transferred for deployment.

SO For developing the java program we need JDK. And for running the java program we need only JRE.

Close video

New Video

JVM Architecture.

Hello every one

in This video we will see the JVM architecture and what happens in JVM.

This image represents the internal structure of JVM. As JVM is used to load verifies and execute byte code and all these things will happen step by step.

Here you can see that JVM has different modules.

First one is class loader, and this center one is Memory management and execution engine.

We will discuss each module.

CURSOR

The first module in JVM is class loader

As JVM is used to execute the java code. First we need to loas the byte code into JVM for this loading we will use class loader

The class loader is used for loading the class files.

CURSOR

There are three phases in class loader.

CURSOR

The first one is loading.

What happens in Loading Phase -

When java program is executed the java class file will be loaded and all the dependencies present in that class files will be loaded. So JVM will try to load all the class files and there dependencies.

The dependencies which were defined in a class file will not present at one place. They might be from different modules. Some dependencies might be the one which user was created and some might be the default java dependencies. So Based on the dependencies, loading has again three phases.

For example IF JVM tries to load some abc.class dependency.

CURSOR

The first bootstrap class loader will try to find the class in rt.jar which was present in JRE lib folder.

CURSOR

IF class is not found then next extension class loader will try to find the class in jre\lib\ext folder. Even if extension class loader didn’t find the class.

CURSOR

Then next application class laoder will search all the jar files and classfiles in the folders provided in classpah environment variable of system.

CURSOR

After running all the loaders still the class is not found then classnotfoundexception will be thrown.

This is how a class file is loaded into JVM

CURSOR

And the second phase is linking

So After the class is loaded by the loader next linking is performed. In this phase the verifier will verify whether the generated bytecode is proper or not. IF verification fails then error will be generated.

And in this linking stage memory will be allotted to static variables and for all the methods present in class files.

Next phase is initialization

CURSOR

Initialization

In this phase all the static variables will be assigned with the values defined and all the static blocks will be executed. Basically initialization of static files and blocks will happen in this phase.

CURSOR

We have discussed about class loader.

Next module is JVM memory. We will discuss JVM memory in separate video. Because it need lot to discussion.

And Next module is execution engine

As we have discussed that the class loader will load the classes, now the loaded byte code will be executed in execution engine. This execution engine will communicate with all the memory modules and execute the code.

Execution engine uses interpreter and Jit compiler to execute the code.

CURSOR

CURSOR

Interpreter will read the byte code and convert to machine code and executes in a line by line manner. Here as it is executed line by line, the same and common code will be executed multiple times and performance of the system will be reduced.

Jit Compiler- jit compiler will consider the block of byte code and convert it into machine code and execute the code. It will not consider line by line as interpreter does and because of this performance of JIT compiler is better than interpreter.

For example we have a for loop in byte code, to execute this loop JIT compiler will first check the byte code and will compile that into machine code at once in which system can understand. As first time it I converted into machine understandable language for next time or for every loop it will not again check the byte code and convert . from second loop on wards it will directly execute the code.

But incase of interpreter it is not like this. As it is executed line by line, for every loop it will consider one line in bytecode loop and converts into machine code and executes and same thing happens for every line in loop and for all the loops. Because of this performance will be reduced. When compare to JIT compiler.

In java Jit compiler is enabled by default. We can also disable the JIT compiler. IF JIT compiler is disabled then jVM will use the interpreter to execute the code. But disabling jit compiler is not recommended.

CURSOR

And one more process of this execution engine is garbage collector. This garbage collector is used to clean up the unwanted objects. We will discuss about this in separate video.

CURSOR

Next the other modules in JVM are native method Interface and native method libraires.

Native methods are basically the code which is not in JAVA. So If you want to run the native methods in JVM then we can use JNI and can run those in JVM.

JNI is an interface that allows Java to interact with code written in another language.