|  |  |
| --- | --- |
| **appletviewer:** | This tool is used to run and debug Java applets without a web browser. |
| **apt:** | It is an annotation-processing tool. |
| **extcheck:** | it is a utility that detects JAR file conflicts. |
| **idlj:** | An IDL-to-Java compiler. This utility generates Java bindings from a given Java IDL file. |
| **jabswitch:** | It is a Java Access Bridge. Exposes assistive technologies on Microsoft Windows systems. |
| **java:** | The loader for Java applications. This tool is an interpreter and can interpret the class files generated by the javac compiler. Now a single launcher is used for both development and deployment. The old deployment launcher, jre, no longer comes with Sun JDK, and instead it has been replaced by this new java loader. |
| **javac:** | It specifies the Java compiler, which converts source code into Java bytecode. |
| **javadoc:** | The documentation generator, which automatically generates documentation from source code comments |
| **jar:** | The specifies the archiver, which packages related class libraries into a single JAR file. This tool also helps manage JAR files. |
| **javafxpackager:** | It is a tool to package and sign JavaFX applications. |
| **jarsigner:** | the jar signing and verification tool. |
| **javah:** | the C header and stub generator, used to write native methods. |
| **javap:** | the class file disassembler. |
| **javaws:** | the Java Web Start launcher for JNLP applications. |
| **JConsole:** | Java Monitoring and Management Console. |
| **jdb:** | the debugger. |
| **jhat:** | Java Heap Analysis Tool (experimental). |
| **jinfo:** | This utility gets configuration information from a running Java process or crash dump. |
| **jmap:** | Oracle jmap - Memory Map- This utility outputs the memory map for Java and can print shared object memory maps or heap memory details of a given process or core dump. |
| **jmc:** | Java Mission Control |
| **jps:** | Java Virtual Machine Process Status Tool lists the instrumented HotSpot Java Virtual Machines (JVMs) on the target system. |
| **jrunscript:** | Java command-line script shell. |
| **jstack:** | It is a utility that prints Java stack traces of Java threads (experimental). |
| **jstat:** | Java Virtual Machine statistics monitoring tool (experimental). |
| **jstatd:** | jstat daemon (experimental). |
| **keytool:** | It is a tool for manipulating the keystore. |
| **pack200:** | JAR compression tool. |
| **Policytool:** | It specifies the policy creation and management tool, which can determine policy for a Java runtime, specifying which permissions are available for code from various sources. |
| **VisualVM:** | It is a visual tool integrating several command-line JDK tools and lightweight [clarification needed] performance and memory profiling capabilities |
| **wsimport:** | It generates portable JAX-WS artifacts for invoking a web service. |
| **xjc:** | It is the part of the Java API for XML Binding (JAXB) API. It accepts an XML schema and generates Java classes. |

JDK: Java Development Kit

JDK is an acronym for Java Development Kit. The Java Development Kit (JDK) is a software development environment which is used to develop java applications and applets. It physically exists. It contains JRE + development tools.

JDK is an implementation of any one of the below given Java Platforms released by Oracle corporation:

|  |  |  |
| --- | --- | --- |
| Top of Form   |  |  | | --- | --- | | * Standard Edition Java Platform * Enterprise Edition Java Platform * Micro Edition Java Platform   The JDK contains a private Java Virtual Machine (JVM) and a few other resources such as an interpreter/loader (Java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc) etc. to complete the development of a Java Application. **Components of JDK** Following is a list of primary components of JDK: |  |   Bottom of Form |

### Usages of JDK:

1. **Development of Java Applications**: JDK provides tools like the compiler and runtime environment necessary for developing, debugging, and running Java applications.
2. **Debugging**: With tools like jdb, developers can find and fix issues in their Java programs.
3. **Packaging**: The jar tool allows developers to bundle multiple Java files into a single archive file, making distribution and deployment easier.
4. **Documentation**: The javadoc tool helps generate API documentation from Java source code, which is useful for developers to understand and use APIs effectively.
5. **Execution of Java Programs**: JDK includes the JRE, which is essential for running Java programs on a development machine.