

Java Programming		Index	
Overview	Getting Started	Fundamentals	Classes and Objects
Collections	Exceptions	Concurrent Programming	Reflection
Applets	JavaBeans	Libraries	

Execution

Navigate **Getting Started** topic:

- Installing Java on Your Computer
- Compiling programs
- Running Java programs
- Developing the first Java program
- Understanding a Java program

There are various ways Java code can be executed. A complex Java application are usually using third party APIs or services. In this section we list the most popular way a piece of Java code may be packed together and/or executed.

JSE code execution

Java language first edition came out in the client-server era. Thick clients were developed with rich GUI interfaces. Java first edition, JSE(Java Standard Edition) had/has the following in its belt:

- GUI capabilities (AWT, Swing)
- Network computing capabilities (RMI)
- Multi-tasking capabilities (Threads)

With JSE the following Java code executions are possible:

Stand alone Java application

(Figure 1) Stand alone application refers to a Java program where both the user interface and business modules are running on the same computer. The application may or may not use a database to persist data. The user interface could be either AWT or Swing.

The application would start with a `main()` method of a Class. The application stops when the `main()` method exits, or if an exception is thrown from the application to the JVM. Classes are loaded to memory and compiled as needed, either from the file system or from a *.jar file, by the JVM.

Invocation of Java programs distributed in this manner requires usage of the command line. Once the user has all the class files, he needs to launch the application by the following command line (where `Main` is the name of the class containing the `main()` method.)

```
java Main
```

Java 'jar' class libraries

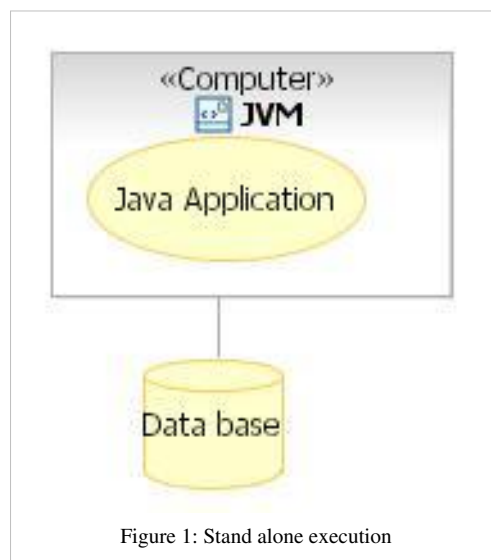


Figure 1: Stand alone execution

Utility classes, framework classes, and/or third party classes are usually packaged and distributed in Java '*.jar' files. These 'jar' files need to be put in the CLASSPATH of the java program from which these classes are going to be used.

If a jar file is executable, it can be run from the command line:

```
java -jar Application.jar
```

Java Applet code

(Figure 2) Java Applets are Java code referenced from HTML pages, by the <APPLET> tag. The Java code is downloaded from a server and running in the client browser JVM. Java has built in support to render applets in the browser window.

Sophisticated GUI clients were found hard to develop mostly because of the download time, incompatibilities between browser's JVM, and its communication requirements back to the server. Applets are rarely used today, and are most commonly used as small separate graphic-like animation applets. The popularity of Java declined when Microsoft withdrew its Java support from Internet Explorer default configuration, however, the plugin is still available as a free download from java.com^[23].

More information can be found about applets at the Applet Chapter, in this book. Also, Wikipedia has an article about Java Applets.

Client Server applications

The client server applications consist of a front-end, and a back-end part, both running on a separate computer. The idea is that the business logic would be on the back-end part of the program, which would be reused by all the clients. Here the challenge is to achieve a separation between front-end user interface code, and the back-end business logic code.

The communication between the front-end and the back-end can be achieved by two ways.

- One way is to define a data communication protocol between the two tiers. The back-end part would listen for an incoming request. Based on the protocol it interprets the request and sends back the result in data form.
- The other way is to use Java Remote Invocation(RMI). With the use of RMI, a remote object can be created and used by the client. In this case Java objects are transmitted across the network.

More information can be found about client-server programming, with sample code, at the Client Server Chapter in this book.

Web Applications

For applications needed by lots of client installations, the client-server model did not work. Maintaining and upgrading the hundreds or thousands of clients caused a problem. It was not practical. The solution to this problem was to create a unified, standard client, for all applications, and that is the *Browser*.

Having a standard client, it makes sense to create a unified, standard back-end service as well, and that is the *Application Server*.

Web Application is an application that is running in the Application Server, and it can be accessed and used by the Browser client.

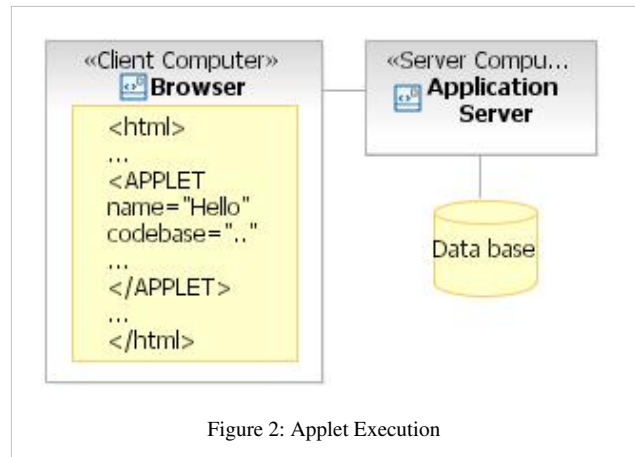


Figure 2: Applet Execution

There are three main area of interest in Web Applications, those are:

- The Web Browser. This is the container of rendering HTML text, and running client scripts
- The HTTP protocol. Text data are sent back and forth between Browser and the Server
- The Web server to serve static content, Application server to serve dynamic content and host EJBs.

Wikipedia also has an article about Web application.

J2EE code execution

As the focus was shifting from reaching GUI clients to thin client applications, with Java version 2, Sun introduced J2EE(Java 2 Extended Edition). J2EE added :

- Components Base Architecture, (Servlet,JSP,EJB Containers)

With J2EE the following Java component executions are possible:

Java Servlet code

(Figure 3) Java got its popularity with server side programming, more specifically with J2EE servlets. Servlets are running in a simple J2EE framework to handle client HTTP requests. They are meant to replace CGI programming for web pages rendering dynamic content.

The servlet is running in a so called servlet-container/web container. The servlet's responsibility is to:

- Handle the request by doing the business logic computation,
- Connecting to a database if needed,
- Create HTML to present to the user through the browser

The HTML output represents both the presentation logic and the results of the business computations. This represents a huge problem, and there is no real application relying only on servlets to handle the presentation part of the responsibility. There are two main solutions to this:

- Use a template tool (Store the presentation part in an HTML file, marking the areas that need to be replaced after business logic computations).
- Use JSP (See next section)

Wikipedia also has an article about Servlets.

Java Server Pages (JSP) code

(Figure 4) JSP is an HTML file with embedded Java code inside. The first time the JSP is accessed, the JSP is converted to a Java Servlet. This servlet outputs HTML which has inside the result of the business logic computation. There are special JSP tags that helps to add data dynamically to the HTML. Also JSP technology allows to create custom tags.

Using the JSP technology correctly, business logic computations should not be in the embedded Java part of the JSP. JSP should be used to render the presentation of the static and dynamic data. Depending on the complexity of the data, 100% separation is not easy to achieve. Using custom tags, however may help to get

