Creating a JAR (Java ARchive) file from a simple "Hello, World!" Java program involves compiling the Java code and then packaging the compiled class files into a JAR file. Here are the steps to achieve this:

**Write the Java Program**

Create a file named HelloWorld.java with the following content:

public class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello, World!");

}

}

**Compile the Java Program**

Open a terminal or command prompt and navigate to the directory where HelloWorld.java is located. Compile the Java file using the javac command:

javac HelloWorld.java

**Create a Manifest File**

To specify the entry point of the application (the main class), you need to create a manifest file. Create a file named manifest.txt with the following content:

Main-Class: HelloWorld

**Create the JAR File**

Use the jar command to create the JAR file. Run the following command in the terminal:

jar cfm HelloWorld.jar manifest.txt HelloWorld.class

This command creates a JAR file named HelloWorld.jar that includes the HelloWorld.class file and the manifest file.

**Run the JAR File**

You can run the JAR file using the java -jar command:

java -jar HelloWorld.jar

**Notes**

* Ensure that the JDK (Java Development Kit) is installed and the javac and jar commands are available in your system's PATH.
* The manifest.txt file should have a newline at the end to be properly recognized by the jar command.
* If you have additional class files or resources to include in the JAR, list them after manifest.txt in the jar command.

JRE vs JDK

Java Development Kit (JDK) and Java Runtime Environment (JRE) are both parts of the Java platform, but they serve different purposes.

### Java Runtime Environment (JRE)

* **Purpose**: The JRE provides the libraries, Java Virtual Machine (JVM), and other components needed to run applications written in Java. It does not contain tools for Java development (like compilers and debuggers).
* **Components**:
  + **Java Virtual Machine (JVM)**: The engine that runs Java applications.
  + **Core Libraries**: Pre-written libraries that Java applications use for common tasks.
  + **Java Plug-in**: A component to run applets in browsers.
  + **Java Web Start**: Allows you to deploy standalone applications over a network.
* **Use Case**: If you only need to run Java applications and do not need to write or compile Java code, you only need the JRE.

### Java Development Kit (JDK)

* **Purpose**: The JDK is a full-featured software development kit for Java, which includes the JRE along with tools for developing, debugging, and monitoring Java applications.
* **Components**:
  + **Java Runtime Environment (JRE)**: Everything needed to run Java applications.
  + **Development Tools**:
    - **javac**: The Java compiler that translates Java source code into bytecode.
    - **jar**: The tool for packaging Java classes into JAR files.
    - **javadoc**: Generates API documentation from Java source code comments.
    - **jdb**: The Java debugger.
    - **Other tools**: Includes various utilities for monitoring and profiling Java applications.
* **Use Case**: If you need to develop Java applications, you need the JDK. It includes the compiler, libraries, and other tools necessary for Java development.

### Comparison

| **Feature** | **JDK** | **JRE** |
| --- | --- | --- |
| Contains JVM | Yes | Yes |
| Contains Libraries | Yes | Yes |
| Contains Compiler | Yes (javac) | No |
| Development Tools | Yes (debugger, javadoc, jar, etc.) | No |
| Primary Use | Developing and running Java applications | Running Java applications only |

### Installation

### apt install openjdk-17-jre-headless

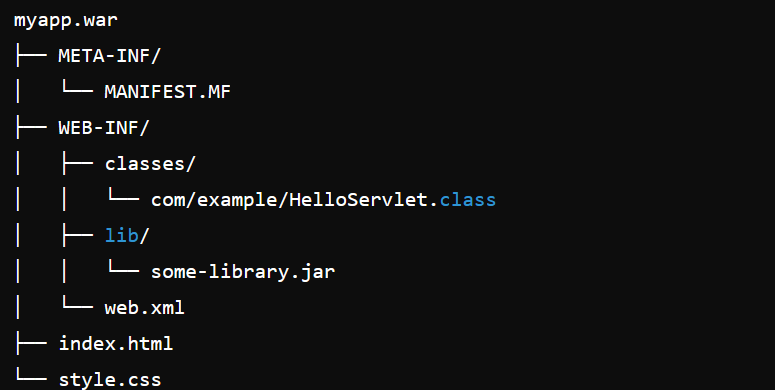
* apt install openjdk-21-jdk-headless

WAR file and EAR file

WAR (Web Application Archive) and EAR (Enterprise Application Archive) files are both archive files used to package Java applications, but they serve different purposes and are used in different contexts.

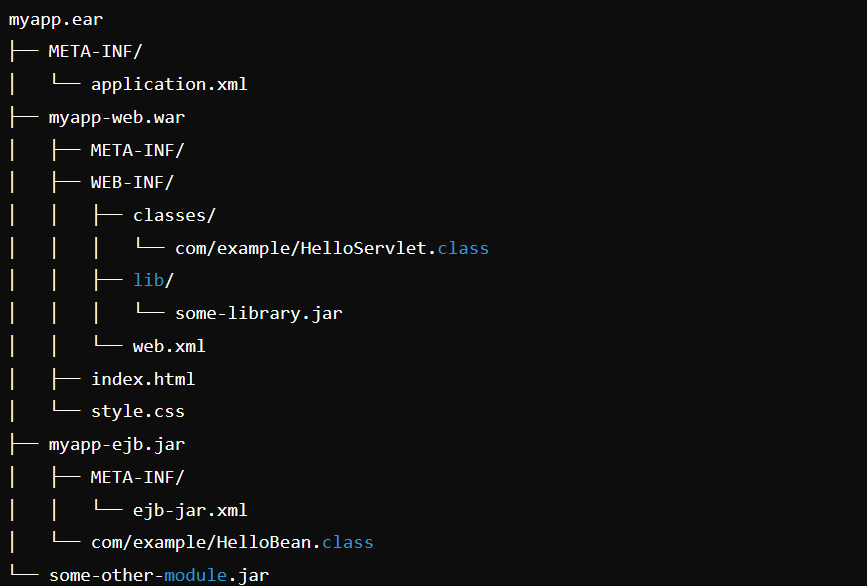
**WAR File**

* **Full Form**: Web Application Archive
* **Purpose**: WAR files are used to package web applications. They contain the web application’s resources, including servlets, JavaServer Pages (JSP), HTML, JavaScript, and other web components.
* **Structure**:
  + WEB-INF/: Contains web.xml (deployment descriptor), classes (compiled servlets and classes), and lib (JAR files).
  + META-INF/: Contains metadata about the WAR file.
  + Other folders and files that make up the web application (HTML, JSP, images, etc.).
* **Deployment**: Deployed to a web server or servlet container such as Apache Tomcat, Jetty, or JBoss.
* **Usage**: Suitable for web-based applications that run on a web server.

****

**EAR File**

* **Full Form**: Enterprise Application Archive
* **Purpose**: EAR files are used to package enterprise applications that consist of multiple modules. These modules can include web modules (WAR files), EJB modules (JAR files), and other resource adapters.
* **Structure**:
  + META-INF/: Contains application.xml (deployment descriptor) and other metadata.
  + Various JAR files for EJB modules.
  + WAR files for web modules.
  + Other resource adapter modules.
* **Deployment**: Deployed to an application server such as IBM WebSphere, Oracle WebLogic, or JBoss/WildFly.
* **Usage**: Suitable for complex, large-scale enterprise applications that require multiple components and modules to work together.



**Comparison**

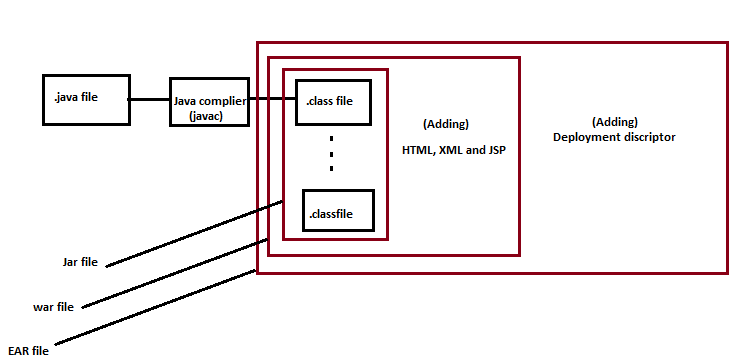
| **Feature** | **WAR File** | **EAR File** |
| --- | --- | --- |
| Acronym | Web Application Archive | Enterprise Application Archive |
| Purpose | Package web applications | Package enterprise applications |
| Contains | Servlets, JSP, HTML, JavaScript | WAR files, EJB modules, resource adapters |
| Deployment | Web servers (Tomcat, Jetty) | Application servers (WebLogic, JBoss) |
| Typical Use Case | Web-based applications | Complex enterprise applications |

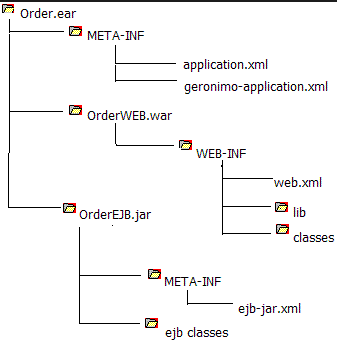
**Build Management**

Every project has 2 logic [Business logic, and presentation logic [referred as non-business logic]]

Build Management is the process of assembling all the components of a software application into an installable software product. This process usually includes the following steps: The Preparing the Build Environment. The Gathering of the Source Code. The Labeling of the Source Code.

In the simple term we can say that build management is a process of creating EAR file that we can deploy/execute from Application server.





|  |  |
| --- | --- |
| Topic | Description |
| Jar file | Collection of .class file is called jar file. |
| War file | Jar file along with JSP, HTML and xml file called war file. |
| EAR file | war file along with deployment descriptor is called EAR file. |