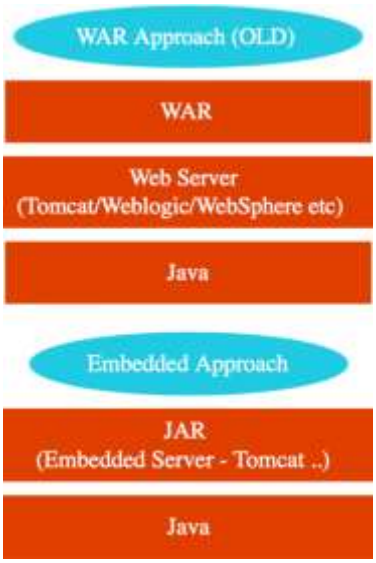
EMBEDDED SERVERS

- **How do you deploy your application?**
  - Step 1: Install Java
  - Step 2: Install Web/Application Server
    - Tomcat/WebSphere/WebLogic etc
  - Step 3: Deploy the application WAR (Web ARchive)
    - This is the OLD WAR Approach
    - Complex to setup!
- **Embedded Server - Simpler alternative**
  - Step 1: Install Java
  - Step 2: Run JAR file.
  - Make JAR not WAR (Credit: Josh Long!)
  - Embedded Server Examples:
    - spring-boot-starter-tomcat
    - spring-boot-starter-jetty
    - spring-boot-starter-undertow



Spring Boot is a popular framework for building enterprise-level Java applications. One of the key features of Spring Boot is the ability to create and run an **embedded web server** within your application.

With the embedded server approach, the **web server is packaged with your application**, and you can run your application as a standalone process, without the need for an external web server like Tomcat or Jetty. This **simplifies the deployment process** and makes it easier to develop, test, and deploy your application.

Spring Boot supports several embedded servers, including Tomcat, Jetty, and Undertow. You can choose the embedded server that best suits your needs and configure it through the Spring Boot auto-configuration feature, which provides sensible default settings based on your application's dependencies and other configuration parameters.

In addition to simplifying deployment, the embedded server approach also provides several performance benefits, as it reduces the overhead of managing a separate web server process. It also allows you to easily package and deploy your application as a **single executable JAR file**, which can be run using a simple command-line interface.

Overall, the Spring Boot embedded server approach provides a streamlined and efficient way to build, package, and deploy web applications, and it is a popular choice among developers for its ease of use and flexibility.