Maven means accumulator of knowledge. This process was introduced to build the application

1. Making the build process easy
2. Providing a uniform build process
3. Providing quality project information
4. Encouraging better development practice
5. Maven is reusable ANT scriptlet.

Maven CLI is available

* To create project
* To Clean
* To compile
* To package
* To do Unite Test
* To run application

Spring Tool Suites IDE or Plugged-in be used for eclipse to bootstrap the process of project creation with pre-defined dependencies

OR Click here to create a project <https://start.spring.io>

Graphical user interface, text, application, email

Description automatically generated

A zip file with the name Artifact Name will be downloaded.

Unzip it

and place it to appropriate location and open this project from eclipse

in Eclipse – File - > Open Project from File System

create a Employee class and under com.example.model package

annotate it @Component

Go to main class

ApplicationContext ctx=SpringApplication.*run*(Jor2HelloWordApplication.**class**, args);

Employee e=(Employee) ctx.getBean(Employee.**class**);

e.setName("Nitin");

System.***out***.println(e.getName());

And run the java application

**Spring Boot Web**

* It has an embedded web server. It supports the following servers:

1. Tomcat [ default ]
2. Jetty
3. Undertow

You no longer need any server to be installed and therefore no need to “run on server” means deployment.

You can simply run the application using **run as Java application** from eclipse. Spring Boot Web Application has main method that you need to run it using the java command of your JRE

java -jar EmonicsHrm.jar

* Java runtime will start the embedded tomcat server at port 8080 [ default ] and the application’s jar will be deployed automatically.

**Spring Rest**

* De-facto standard for building web services on the web because it is easy to build and consume
* REST embraces the precepts of the web including architecture, benefits and everything else.
* REST supports interoperable communication between client and server over the HTTP/s protocol.
* The web & its core protocol HTTP.
  + GET ------- Read
  + POST------- Create
  + PUT -------- Update
  + DELETE ------- Delete
  + OPTION
  + HEAD
  + TRACE
* Caching
* Redirection & Forwarding
* Security [ Encryption & Authentication ]
* Backward compatibility
* Evolving APIs
* Scalable Services
* Stateless Service

Difference between Controller & RestController

@Controller + @ResponseBody = @RestController