

**Framework  
Spring**



**Formadora: Calletana López Baleta**

# Spring



Spring es un proyecto de código abierto que proporciona un enfoque simplificado y modular para crear aplicaciones con Java. La familia de proyectos Spring comenzó en 2003 como respuesta a las complejidades del desarrollo inicial de Java y proporciona compatibilidad para desarrollar aplicaciones de Java. El nombre, Spring, por sí solo suele hacer referencia al marco de trabajo de la aplicación en sí o a todo el grupo de proyectos o módulos. Spring Boot es un módulo específico que se compila como una extensión del marco de Spring.

## Página oficial

<https://spring.io/>

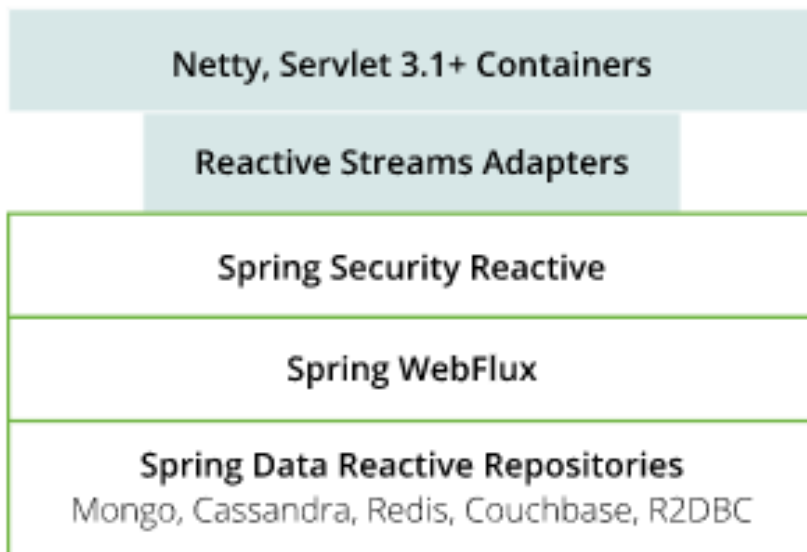
Fuent: <https://azure.microsoft.com/>

Fuente: Cazorso, Carlo. Estructuras de datos y <https://guru99.es/java-data-abstraction/>



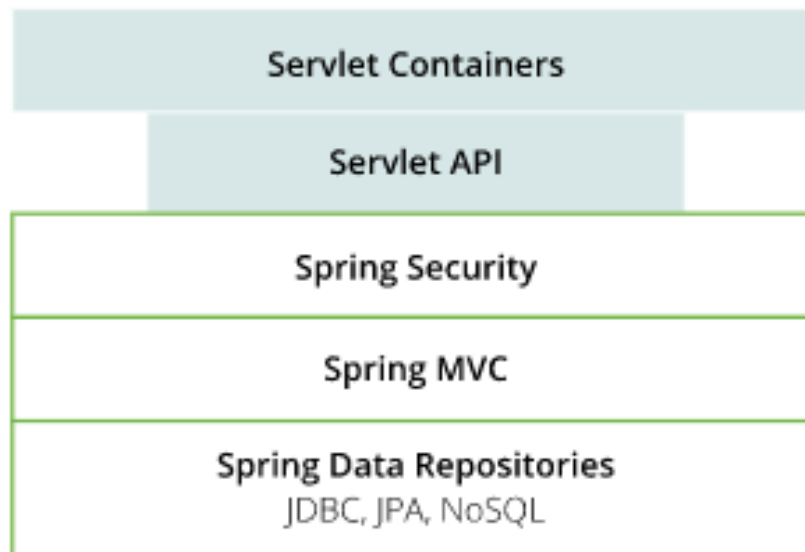
## Reactive Stack

Spring WebFlux is a non-blocking web framework built from the ground up to take advantage of multi-core, next-generation processors and handle massive numbers of concurrent connections.



## Servlet Stack

Spring MVC is built on the Servlet API and uses a synchronous blocking I/O architecture with a one-request-per-thread model.

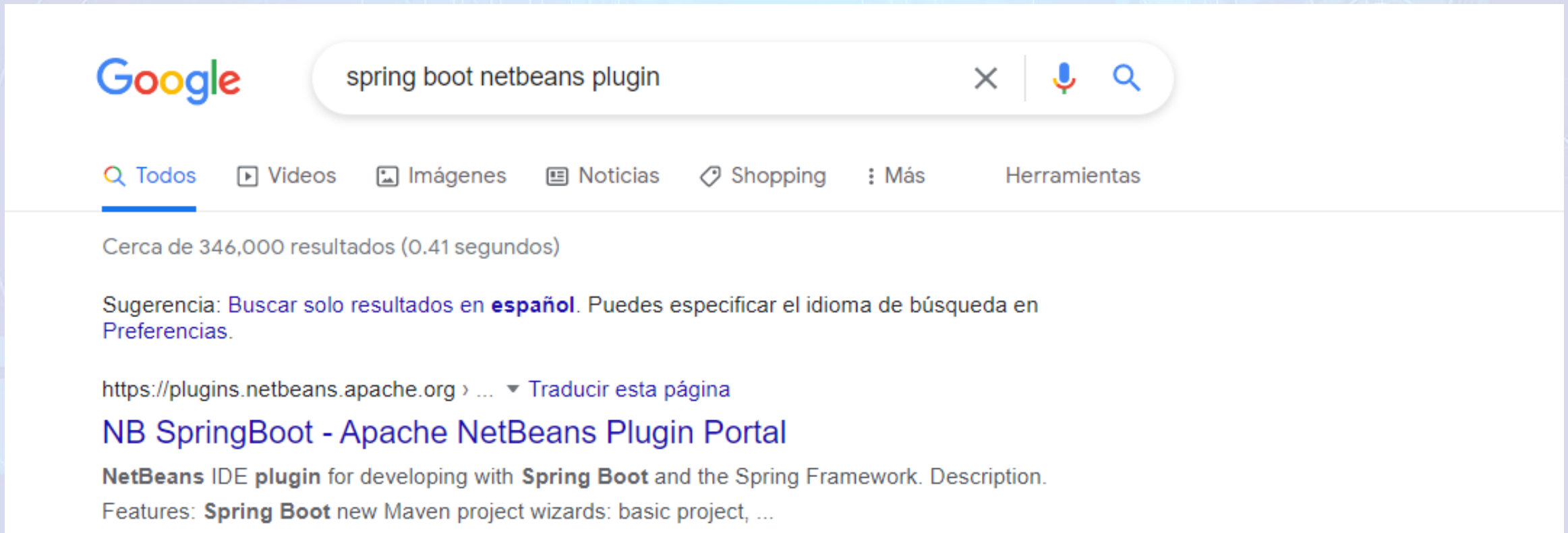


# Pasos para crear un proyecto con Spring

1. Ingresar a: <https://start.spring.io/>
2. Y crear el proyecto...
3. Pero así no lo vamos a hacer, lo vamos a crear directamente de Netbeans.....

# Pasos para crear un proyecto con Spring

## 1. Descargar el plugging para Netbeans

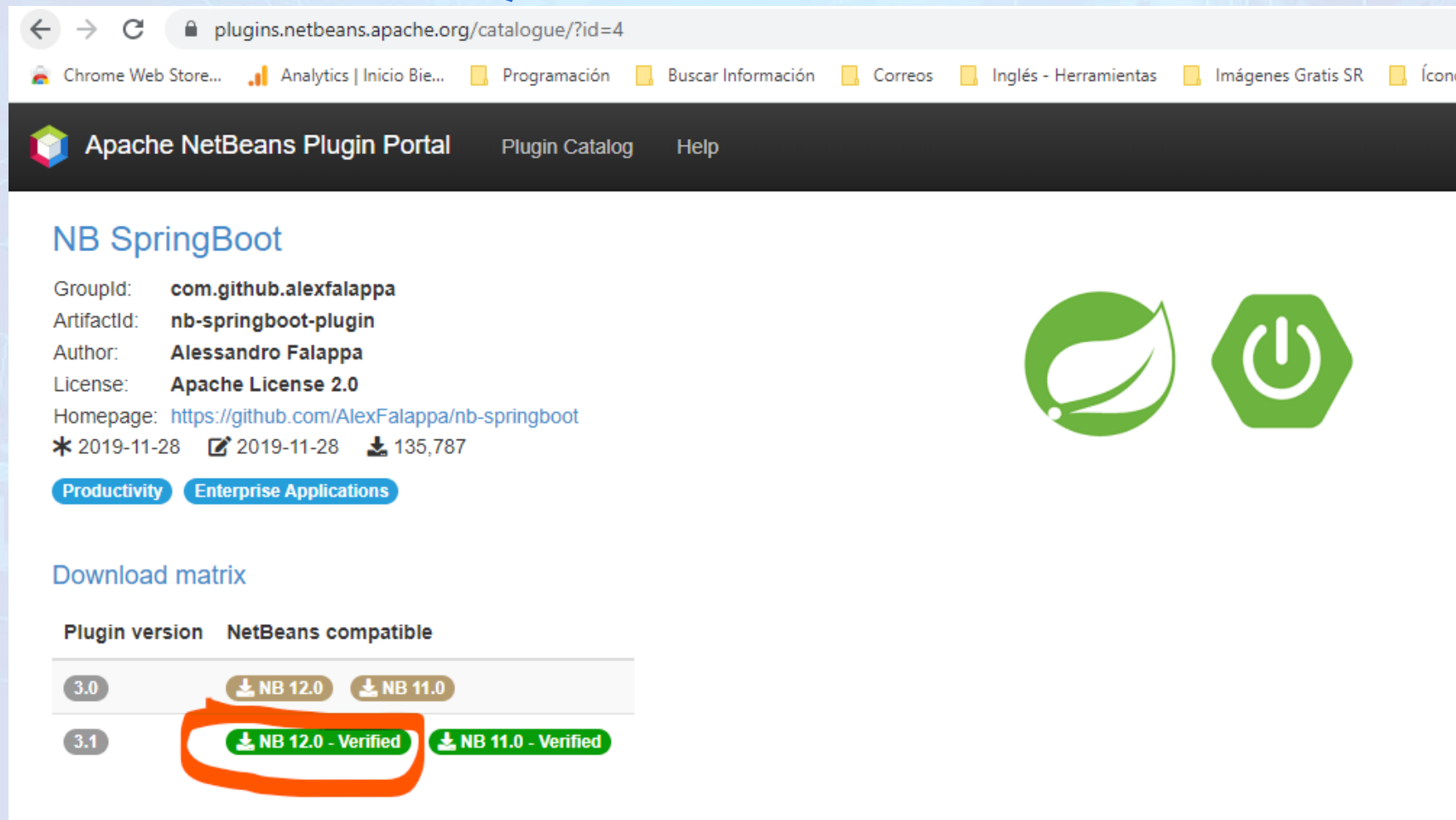




# Pasos para crear un proyecto con Spring

2.

Descargar el plugging de acuerdo a la versión de Netbeans instalada, se recomienda la última



The screenshot shows the Apache NetBeans Plugin Portal for the 'NB SpringBoot' plugin. The page includes the following information:

- GroupId:** com.github.alexfalappa
- ArtifactId:** nb-springboot-plugin
- Author:** Alessandro Falappa
- License:** Apache License 2.0
- Homepage:** <https://github.com/AlexFalappa/nb-springboot>
- Release Date:** 2019-11-28
- Download Count:** 135,787

Two green icons are displayed: a leaf and a power button.

Tags: **Productivity** **Enterprise Applications**

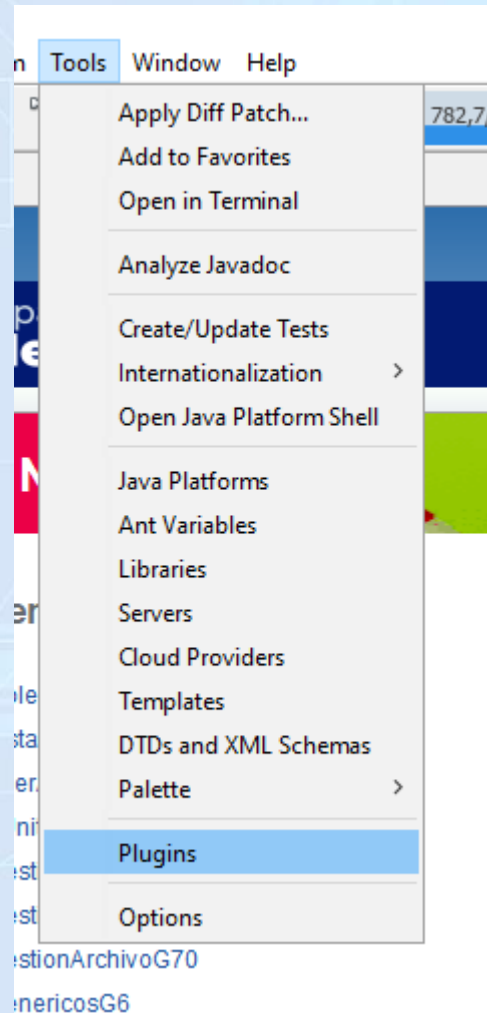
**Download matrix**

Plugin version	NetBeans compatible
3.0	<a href="#">NB 12.0</a> <a href="#">NB 11.0</a>
3.1	<a href="#">NB 12.0 - Verified</a> <a href="#">NB 11.0 - Verified</a>

The 'NB 12.0 - Verified' link for version 3.1 is highlighted with an orange circle.

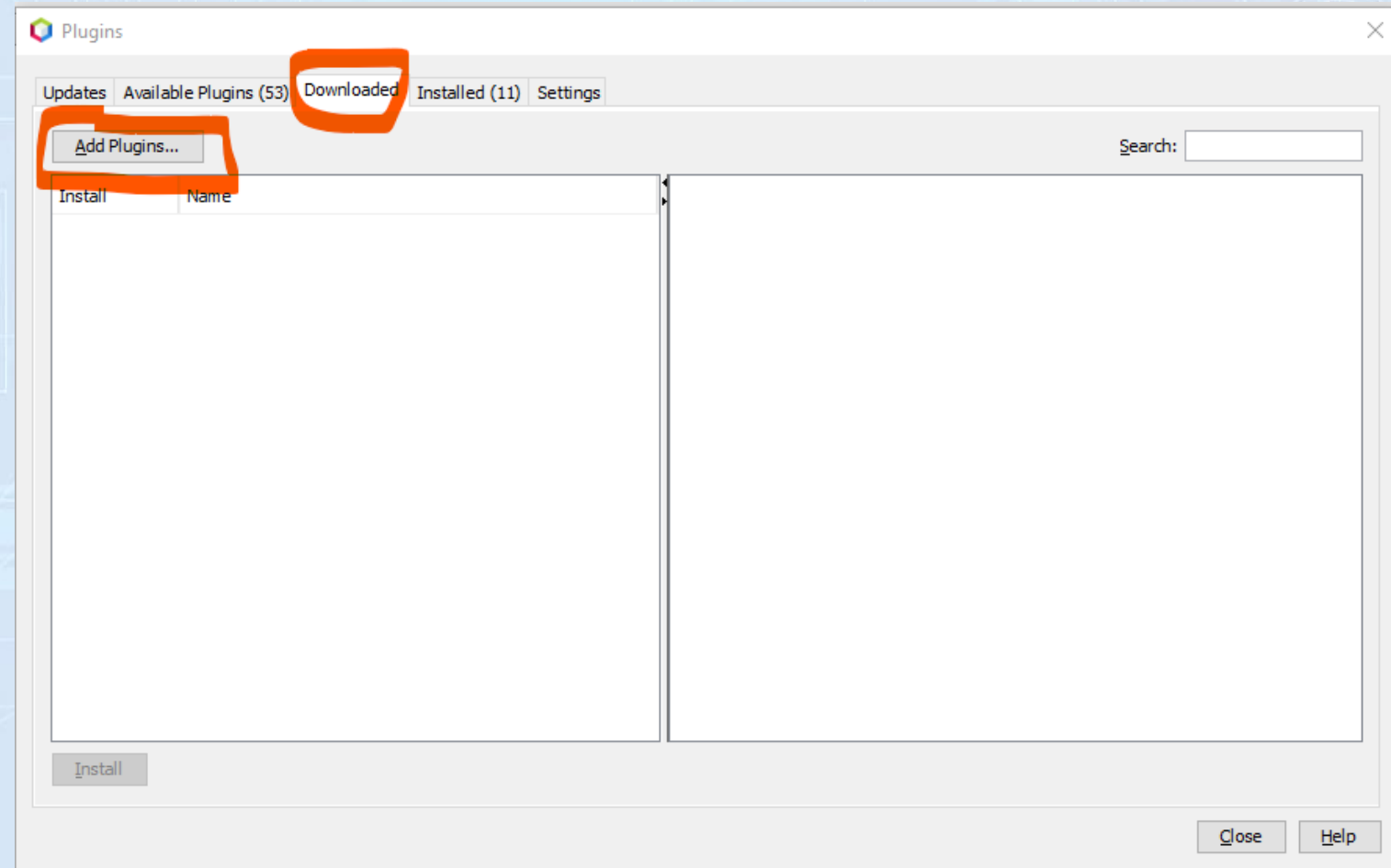
# Pasos para crear un proyecto con Spring

## 3. Abrimos Netbeans y nos vamos a: Tool/Plugging



# Pasos para crear un proyecto con Spring

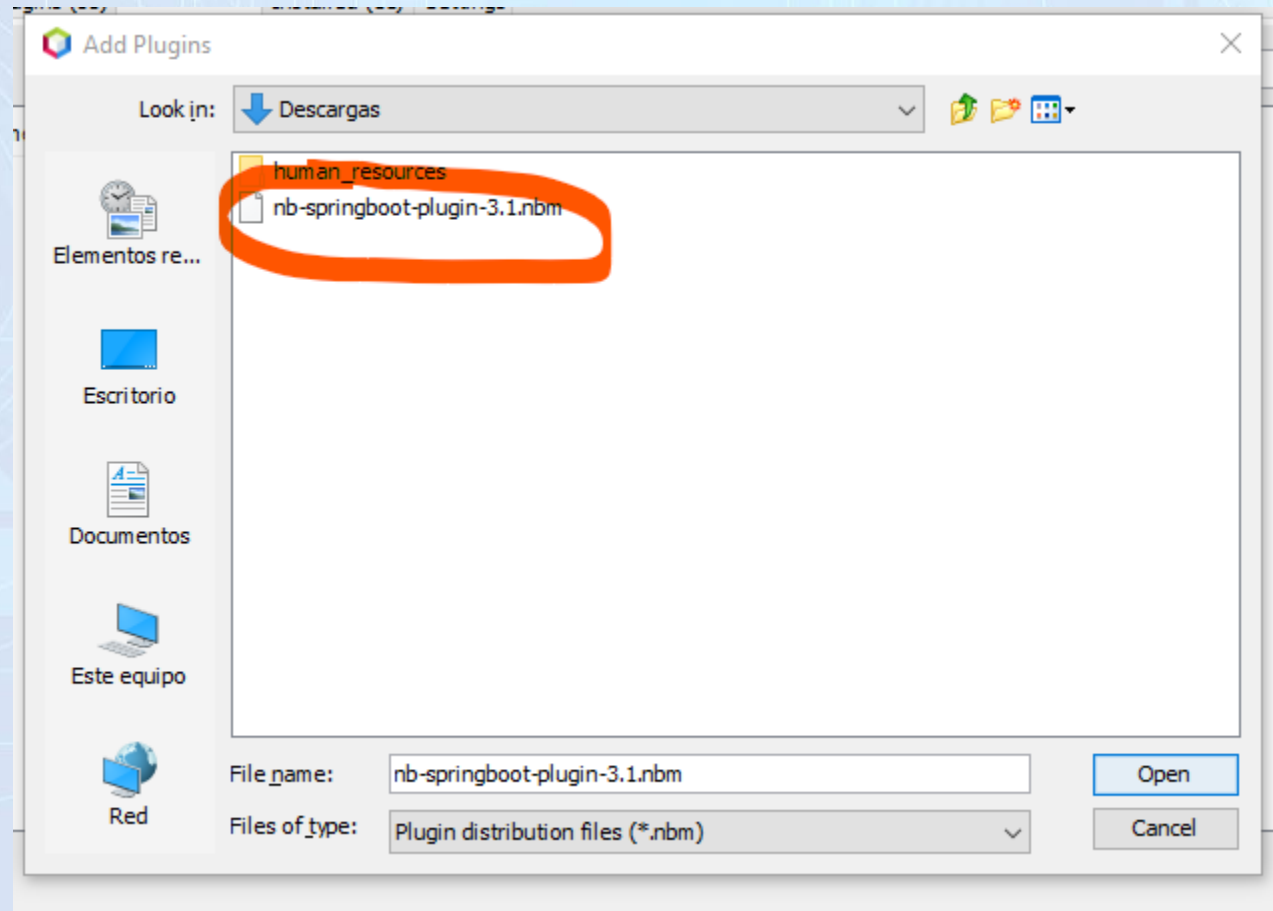
## 4. Abrimos Netbeans y nos vamos a: Tool/Plugging/Downloaded





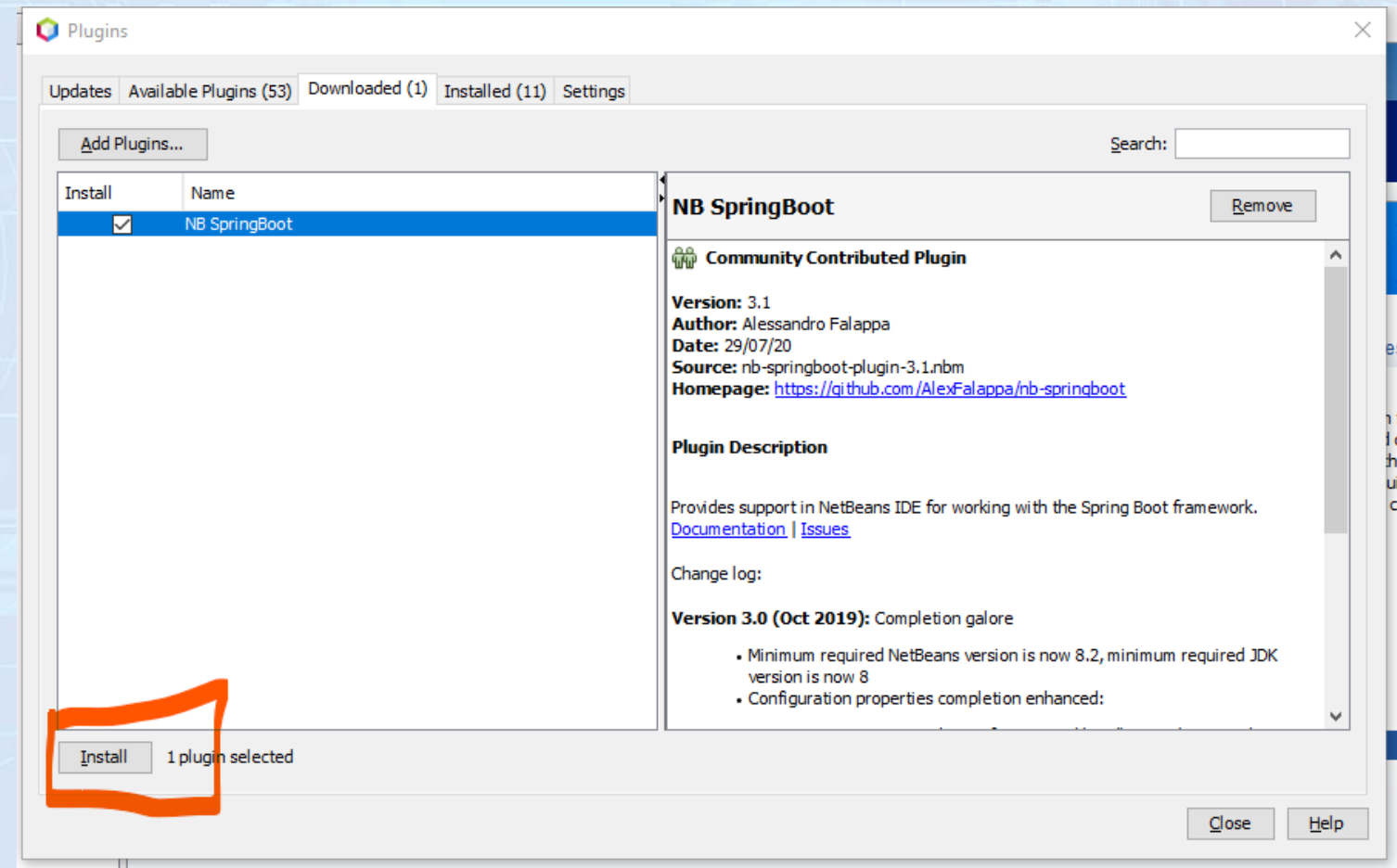
# Pasos para crear un proyecto con Spring

## 5. Seleccionamos el plugging Tool/Plugging/Downloaded



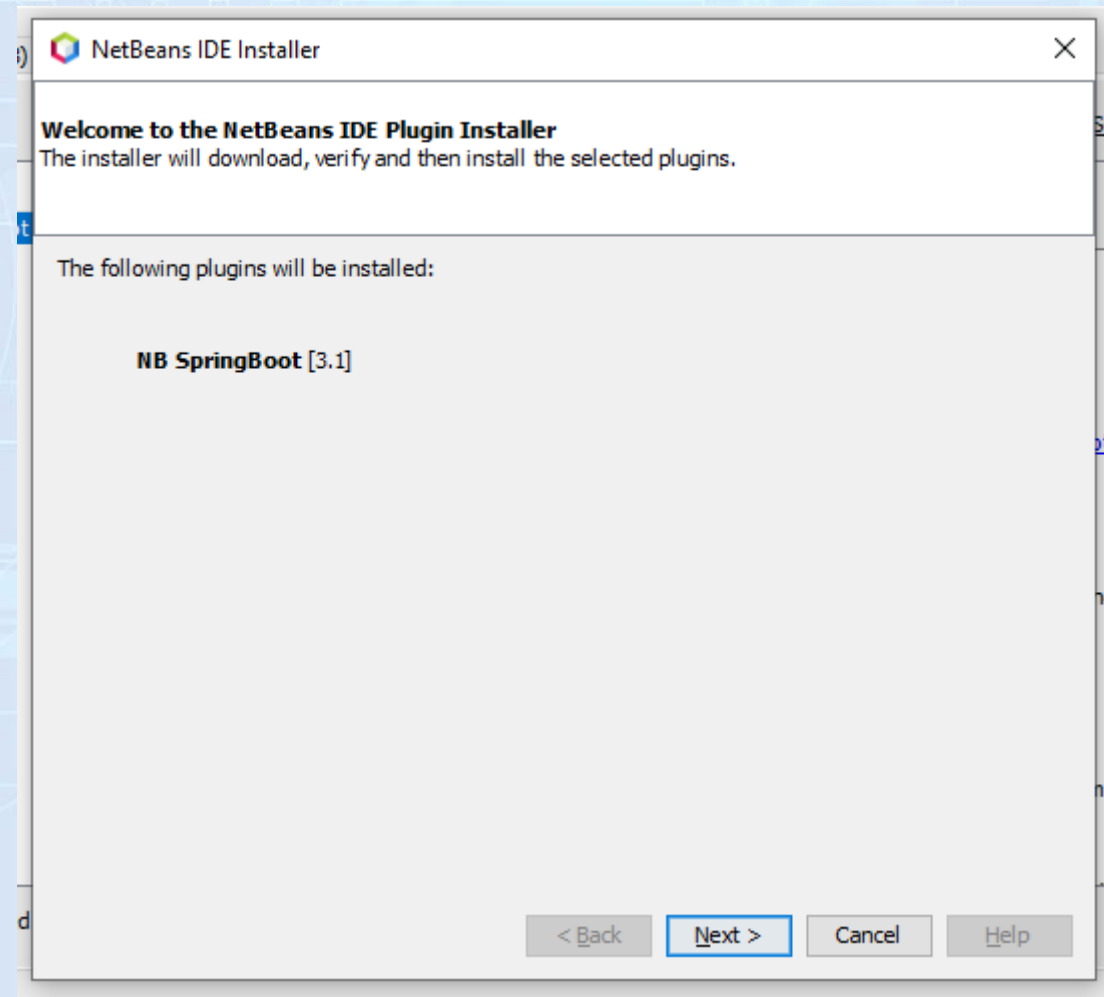
# Pasos para crear un proyecto con Spring

## 6. Instalamos el plugging Tool/Plugging/Downloaded



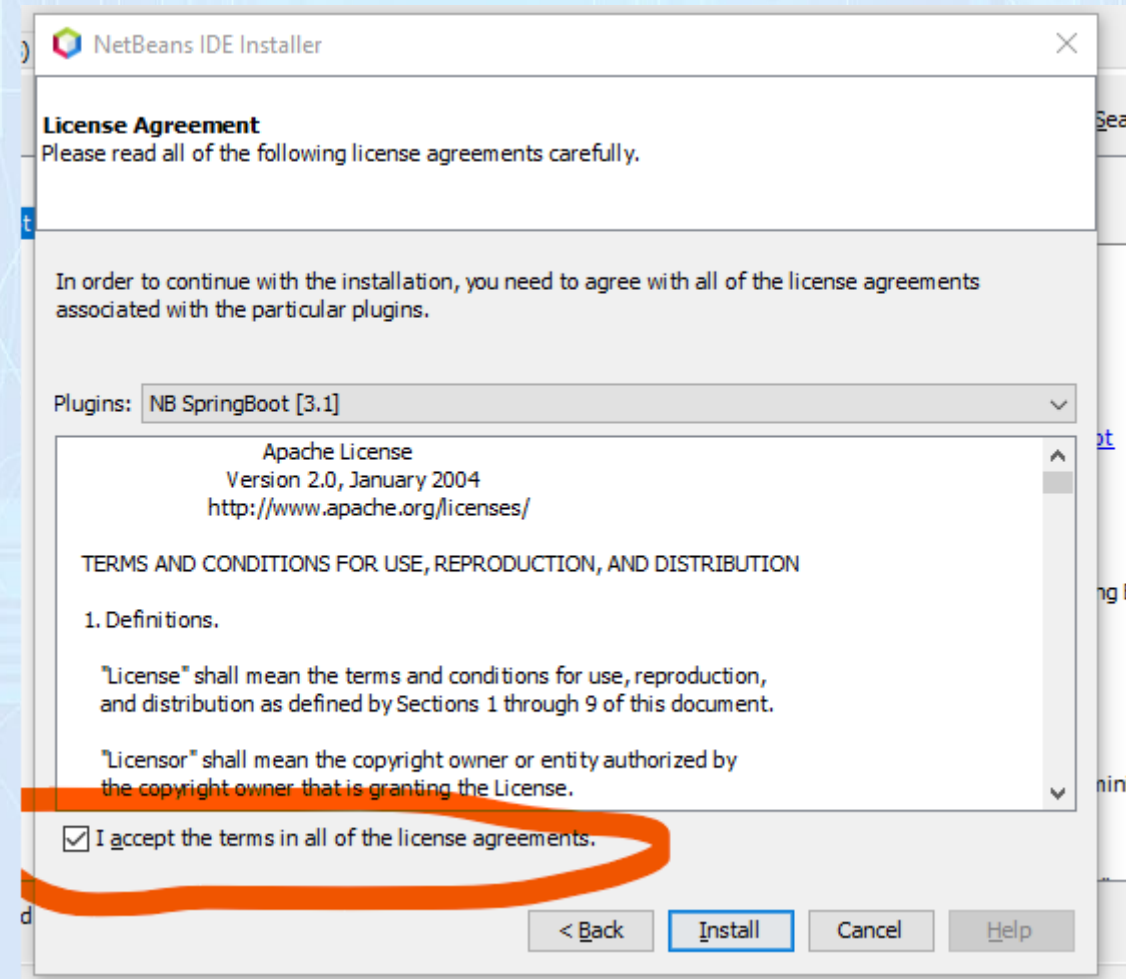
# Pasos para crear un proyecto con Spring

## 7. Instalamos el plugging Tool/Plugging/Downloaded



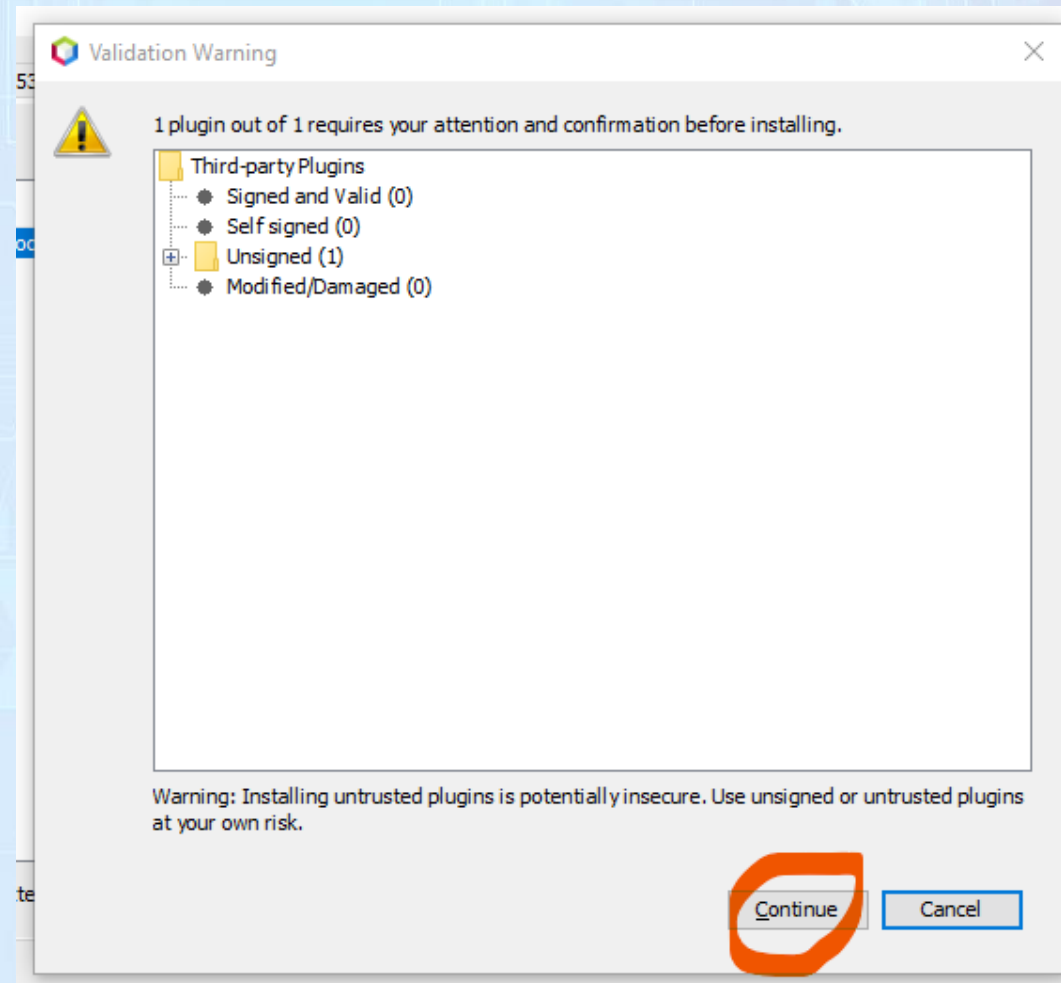
# Pasos para crear un proyecto con Spring

## 8. Instalamos el plugging Tool/Plugging/Downloaded



# Pasos para crear un proyecto con Spring

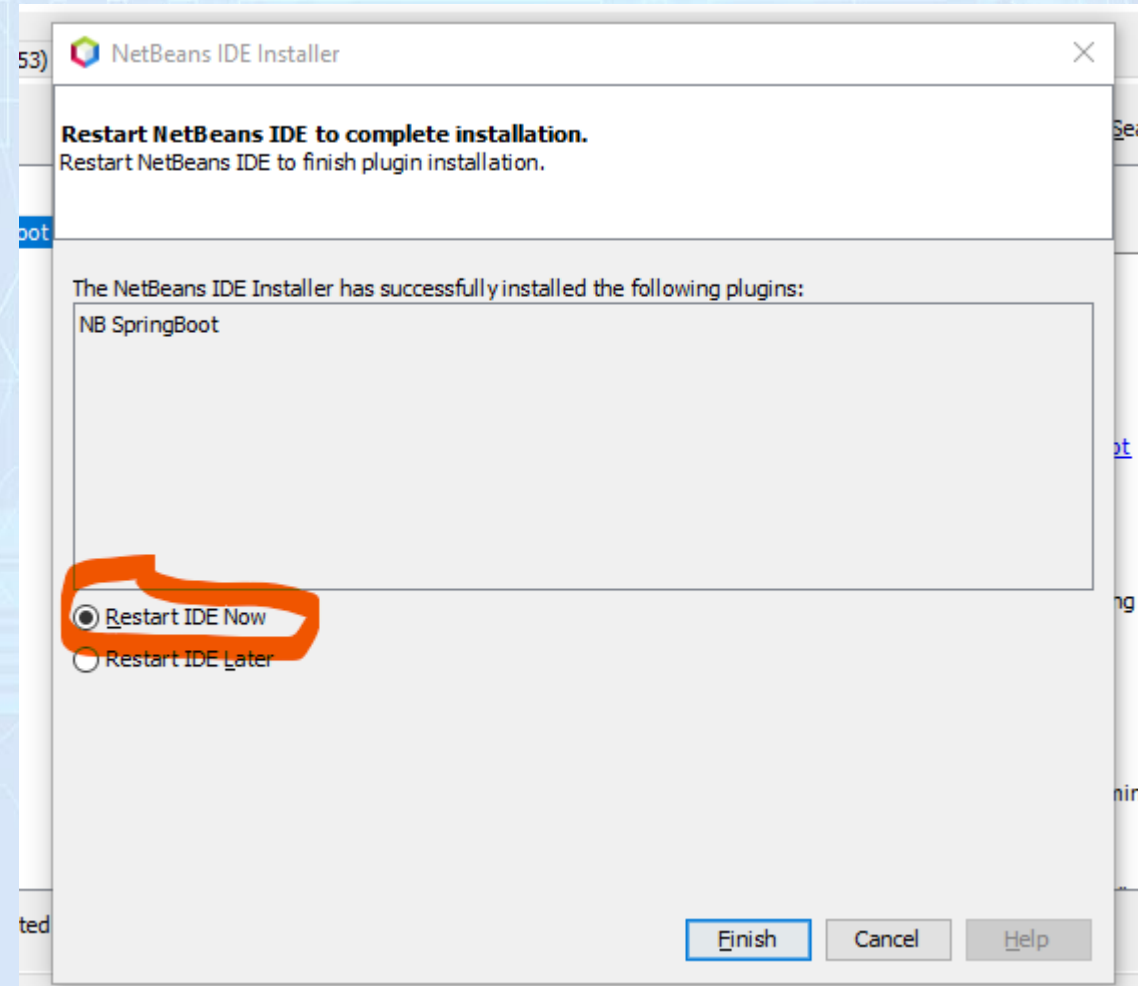
## 9. Instalamos el plugging Tool/Plugging/Downloaded





# Pasos para crear un proyecto con Spring

10. Finish y esperamos que reinicie y listo....  
Tool/Plugging/Downloaded

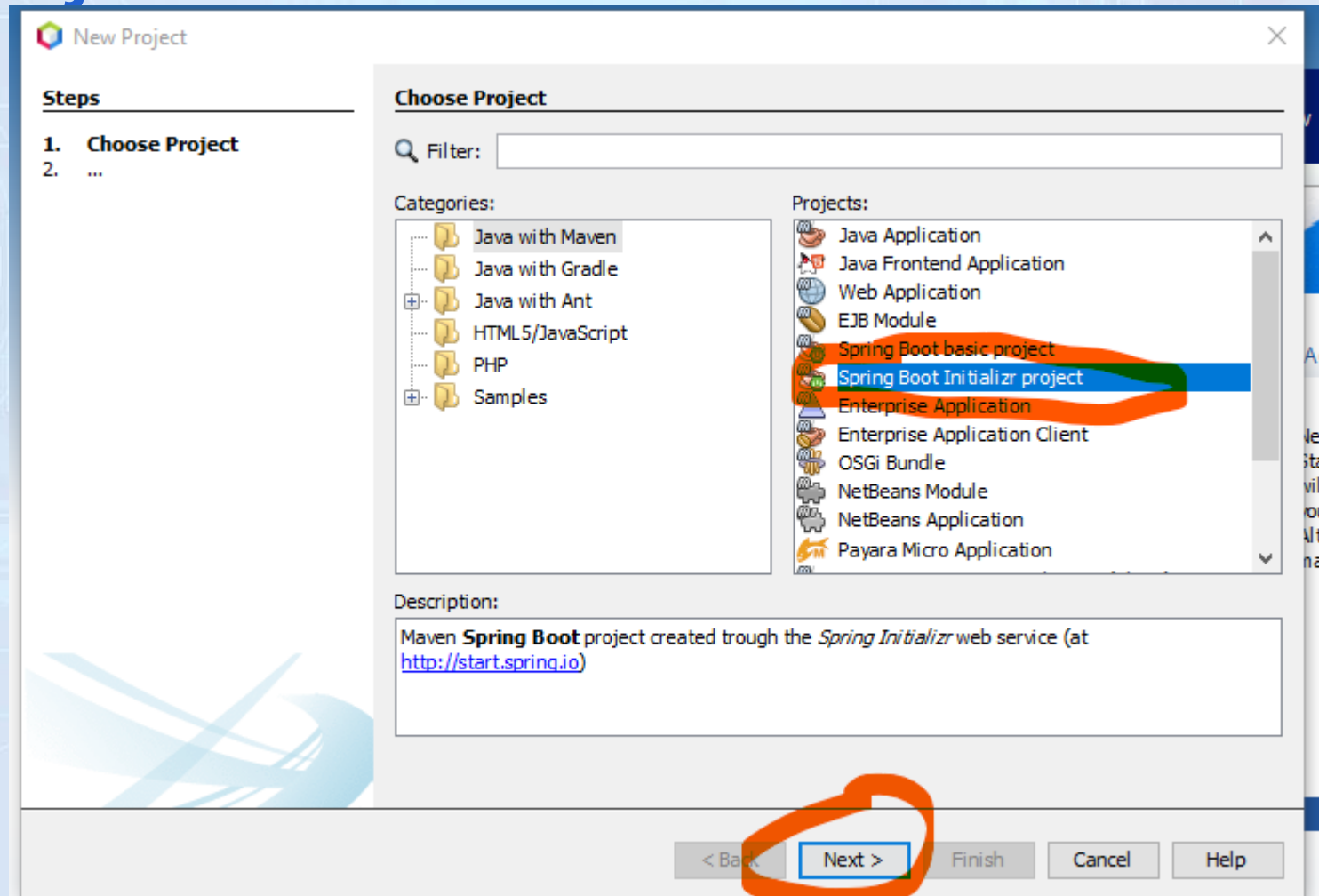




# Ahora a crear nuestro proyecto

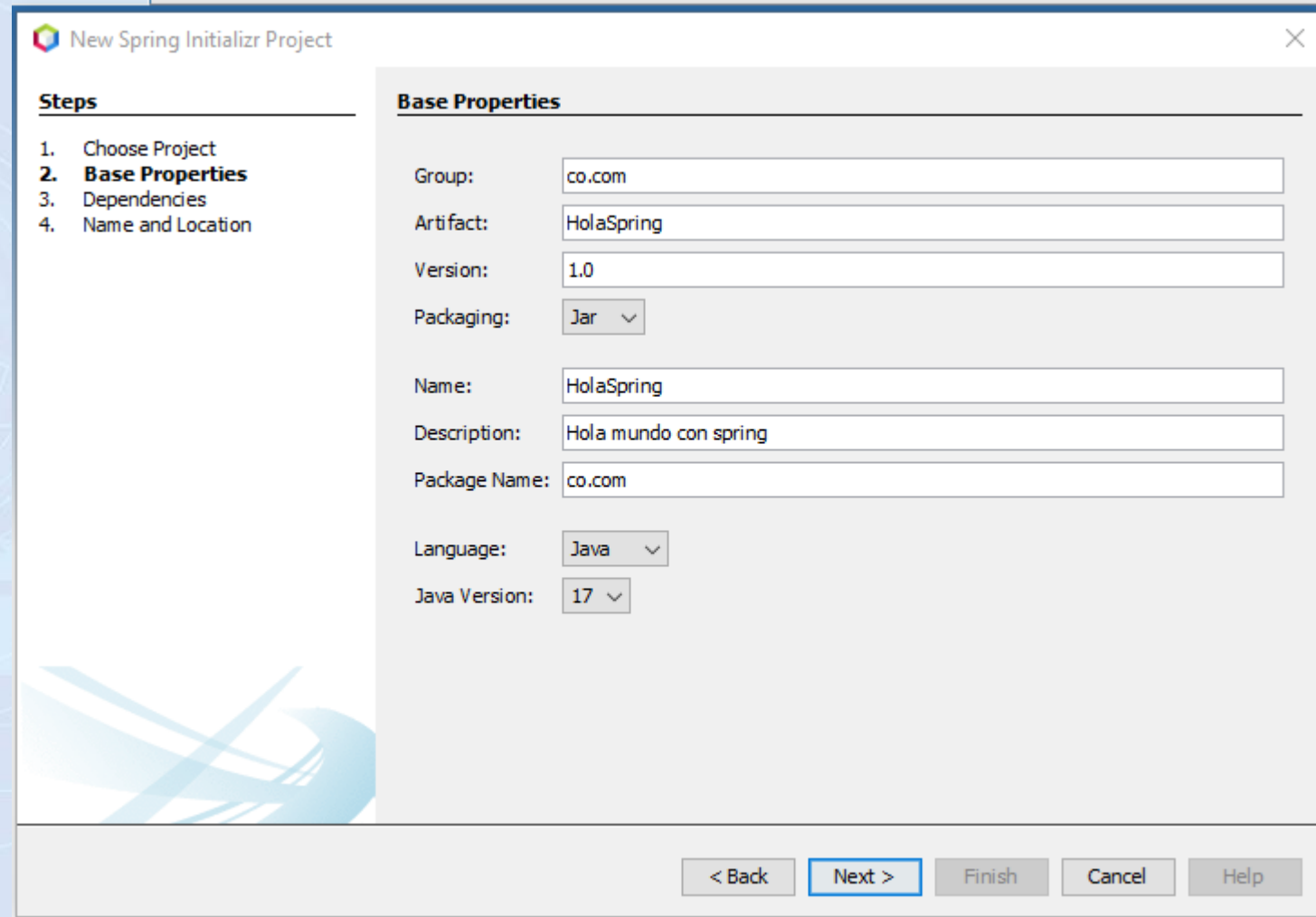
# Pasos para crear un proyecto con Spring

## 1. Luego de instalar Spring abrimos Netbeans y le damos en crear New Project



# Pasos para crear un proyecto con Spring

## 2. Llenar los campos



New Spring Initializr Project

**Steps**

1. Choose Project
- 2. Base Properties**
3. Dependencies
4. Name and Location

**Base Properties**

Group:

Artifact:

Version:

Packaging:

Name:

Description:

Package Name:

Language:

Java Version:

< Back   **Next >**   Finish   Cancel   Help

# Pasos para crear un proyecto con Spring

## 3. Escoger las siguientes opciones básicas

New Spring Initializr Project

**Steps**

1. Choose Project
2. Base Properties
3. **Dependencies**
4. Name and Location

**Dependencies**

Spring Boot Version: 2.7.3 Filter

**Developer Tools**

- ☐ Spring Native [Experimental] Incubating support for compiling Spring applications to native executables using the GraalVM native-image compiler.
- ☒ Spring Boot DevTools Provides fast application restarts, LiveReload, and configurations for enhanced development experience.
- ☒ Lombok Java annotation library which helps to reduce boilerplate code.
- ☐ Spring Configuration Processor Generate metadata for developers to offer contextual help and "code completion" when working with custom configuration keys (ex.application.properties/.yaml files).

**Web**

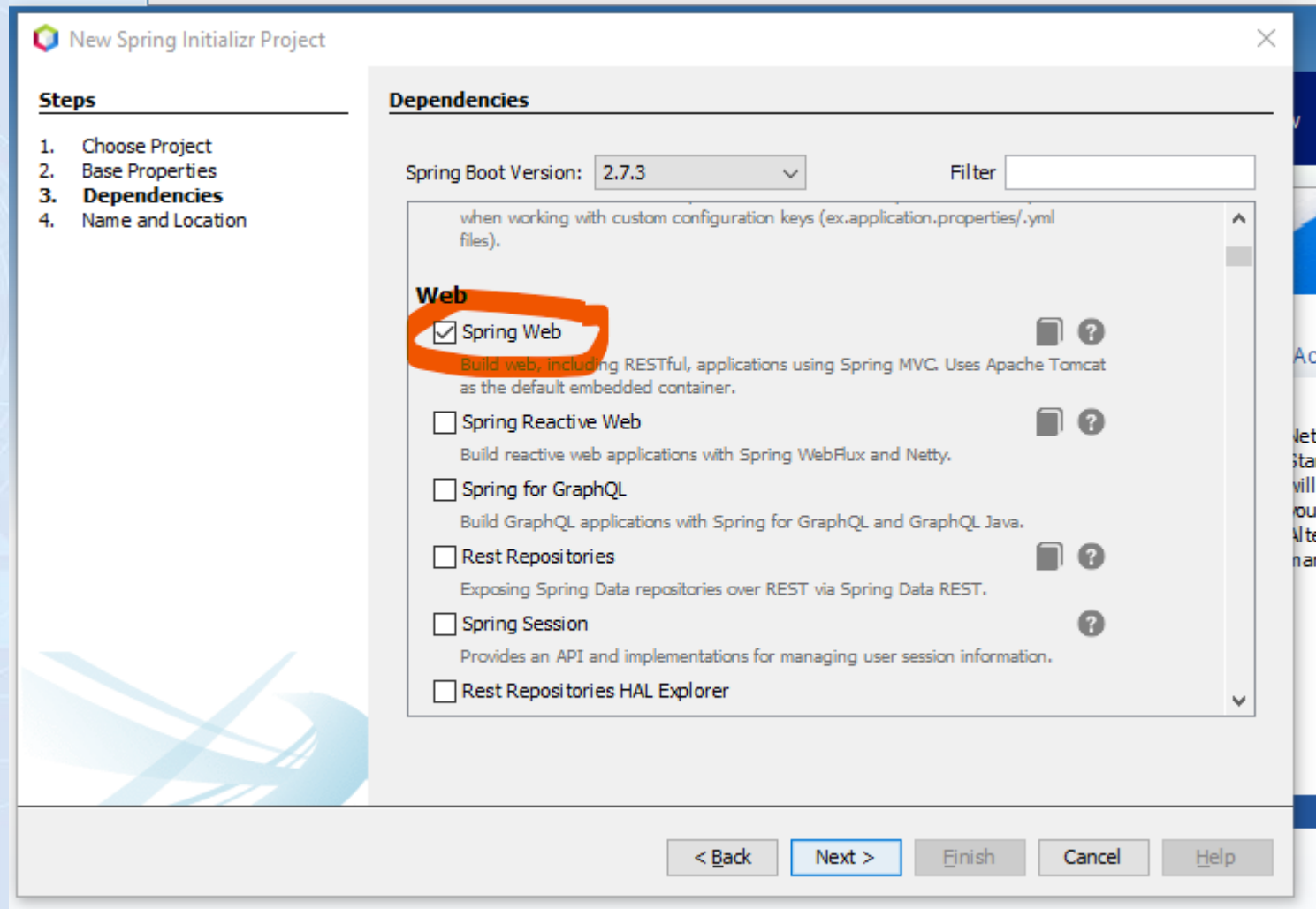
- ☐ Spring Web Build web, including RESTful applications using Spring MVC. Uses Apache Tomcat

< Back Next > Finish Cancel Help



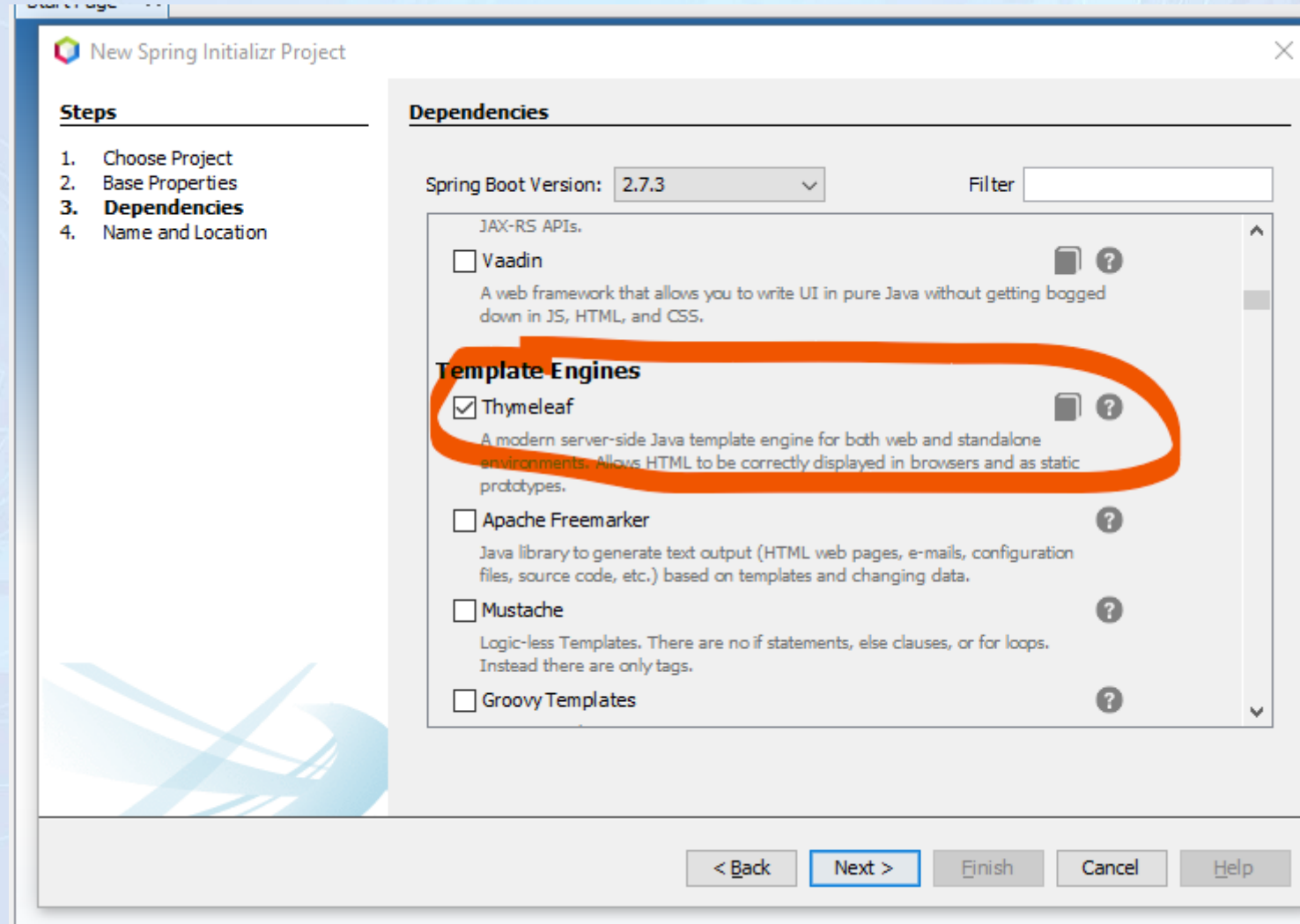
# Pasos para crear un proyecto con Spring

## 4. Escoger las siguientes opciones básicas



# Pasos para crear un proyecto con Spring

## 5. Escoger las siguientes opciones básicas



**New Spring Initializr Project**

**Steps**

1. Choose Project
2. Base Properties
3. **Dependencies**
4. Name and Location

**Dependencies**

Spring Boot Version:  Filter

**Template Engines**

- ☐ Vaadin  
A web framework that allows you to write UI in pure Java without getting bogged down in JS, HTML, and CSS.
- ☒ Thymeleaf  
A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.
- ☐ Apache Freemarker  
Java library to generate text output (HTML web pages, e-mails, configuration files, source code, etc.) based on templates and changing data.
- ☐ Mustache  
Logic-less Templates. There are no if statements, else clauses, or for loops. Instead there are only tags.
- ☐ Groovy Templates

< Back Next > Finish Cancel Help

# Pasos para crear un proyecto con Spring

## 6. Crear Proyecto

OJO: crear una carpeta lo más cercana a C o donde esté instalado el SO y sin espacios ni caracteres especiales

New Spring Initializr Project

**Steps**

1. Choose Project
2. Base Properties
3. Dependencies
4. **Name and Location**

**Name and Location**

Project Name: HolaKy

Project Location: C:\proyectospring Browse...

Project Folder: C:\proyectospring\HolaKy

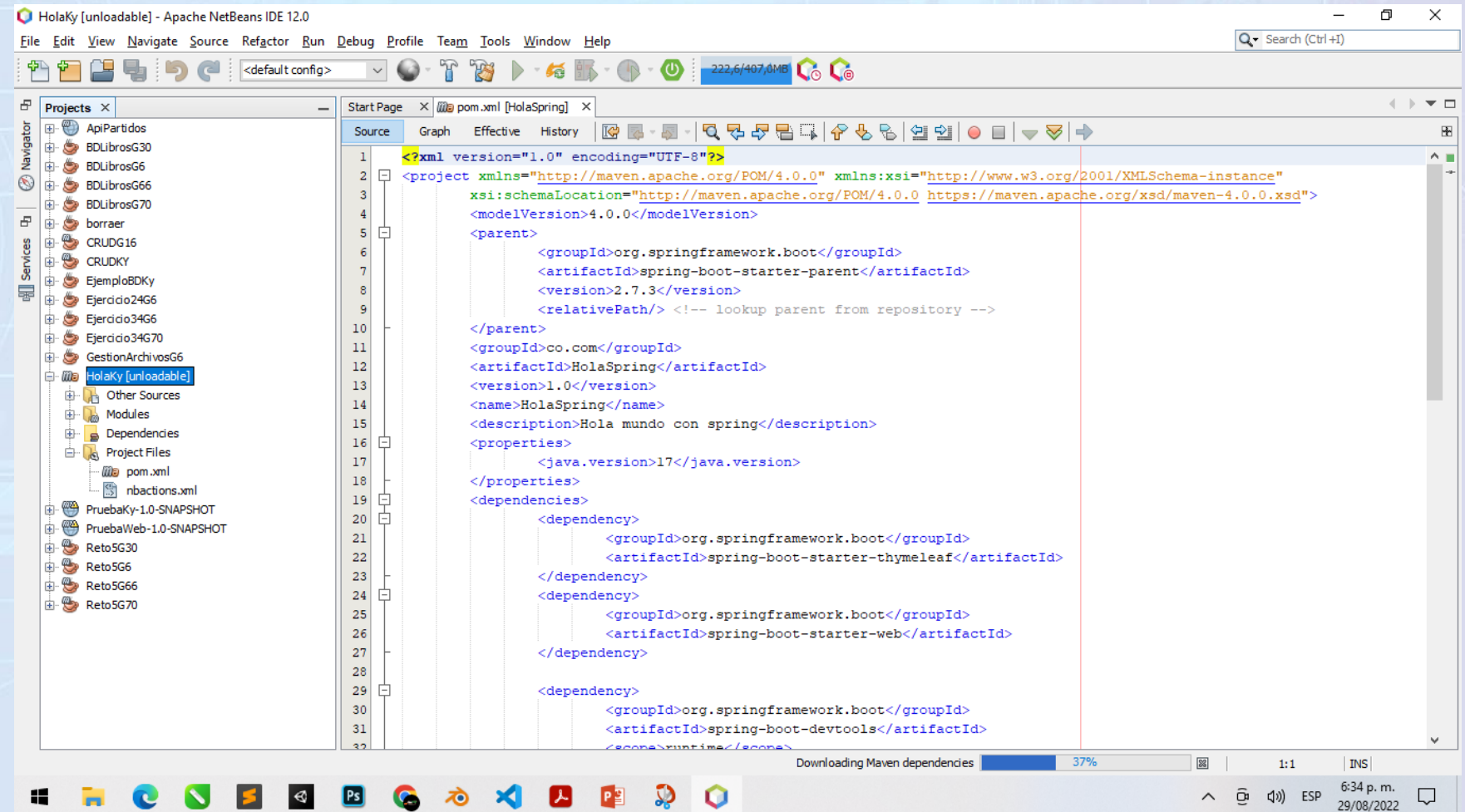
☒ Run/Debug trough Spring Boot maven plugin

☒ Remove Maven Wrapper

< Back Next > Finish Cancel Help

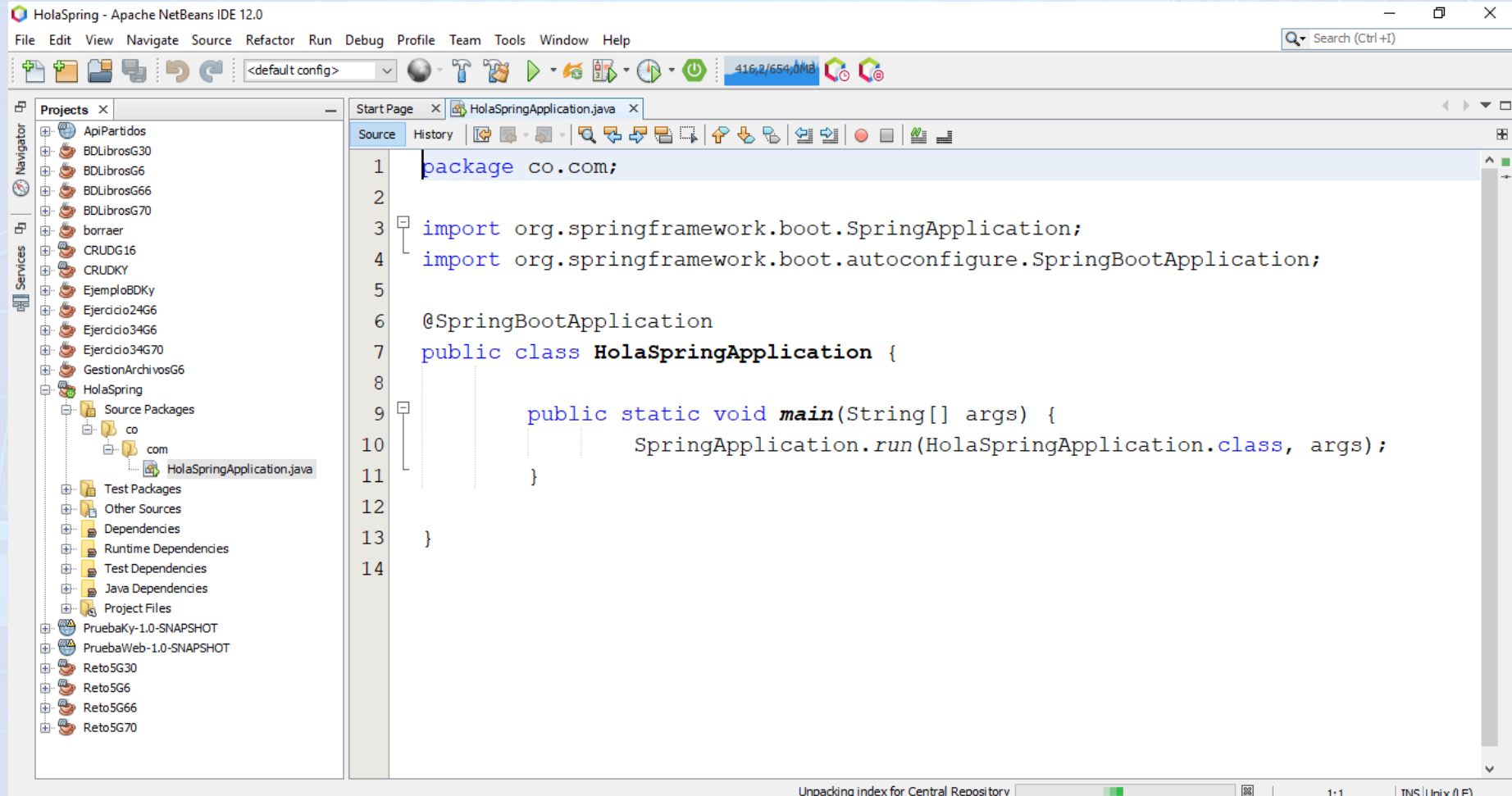
# Pasos para crear un proyecto con Spring

## 7. Debo esperar que descargue las dependencias, sino la descarga automáticamente le damos un Clean and Build



# Pasos para crear un proyecto con Spring

## 8. Listo ya debe estar creado el proyecto con la clase principal





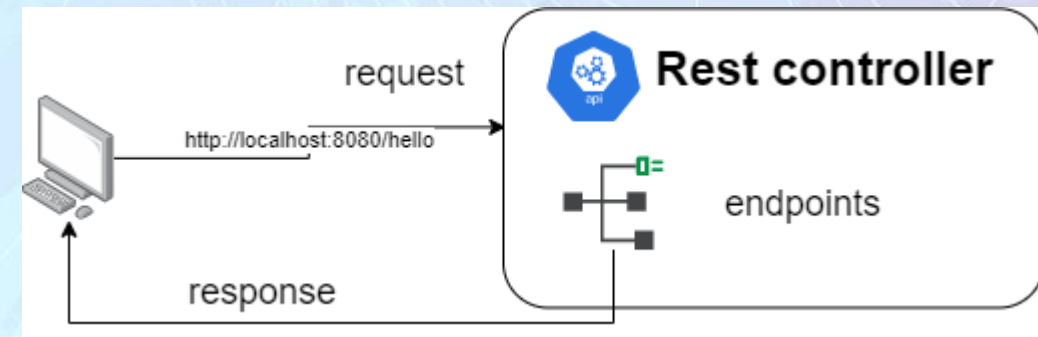
# Creando nuestro primer “Hola Mundo”

## Utilizar un Controlador tipo REST

Un *controller* es un componente de Spring capaz de recibir peticiones http y responderlas.

Las clases que definimos como un controller es responsable de procesar las llamadas entrantes (request) que ingresan a nuestra aplicacion, validarlas y dar una respuesta (response).

Un *rest controller* es un tipo de *controller* que reciben peticiones con un formato de específico que cumple con formatos de solicitud RESTful habitualmente y mayormente en [JSON](#) , aunque a veces se usan otros como HTML, XML, o simplemente texto.



# Creando nuestro primer “Hola Mundo”

## Cómo crear un RestController con Spring Boot?

El primer paso para crear un ‘controlador rest’ es anotar la clase que con `@RestController`.

Con esto Spring ya sabe que esa clase será un componente encargado de recibir llamadas.

```
@RestController  
public class HelloWorldController {  
}
```

# Creando nuestro primer “Hola Mundo”

## Método de respuesta

Una vez que hemos anotado la clase podemos definir el método y la ruta en la cual recibirá la llamada externa.

Anotamos el método que da respuesta con `@GetMapping` y le indicamos la ruta en la cual responderá .

```
@RestController
public class HelloWorldController {

    @GetMapping("/hello")
    public String sayHello() {
        return "Hello dev";
    }

}
```

# Creando nuestro primer “Hola Mundo”

## Tipos de mapeos en un rest controller de Spring Boot

En los métodos de un mapeo podemos usar:

- Get: para solicitar información de un recurso.
- Post: para enviar información a fin de crear o de actualizar un recurso.
- Put: para enviar información a fin de modificar un recurso.
- Patch: actualiza una parte del recurso.
- Delete: elimina un recurso específico.

¿Cuál es la diferencia entre Post , Put, Patch?

Habitualmente la diferencia entre Post y Put radica en que Post lo usamos para **añadir** un recurso y Put lo utilizamos para **modificar** un recurso en particular.

Patch también lo utilizamos para actualizar un recurso pero solo una **parcialidad** del mismo.

Mapping

GetMapping

PostMapping

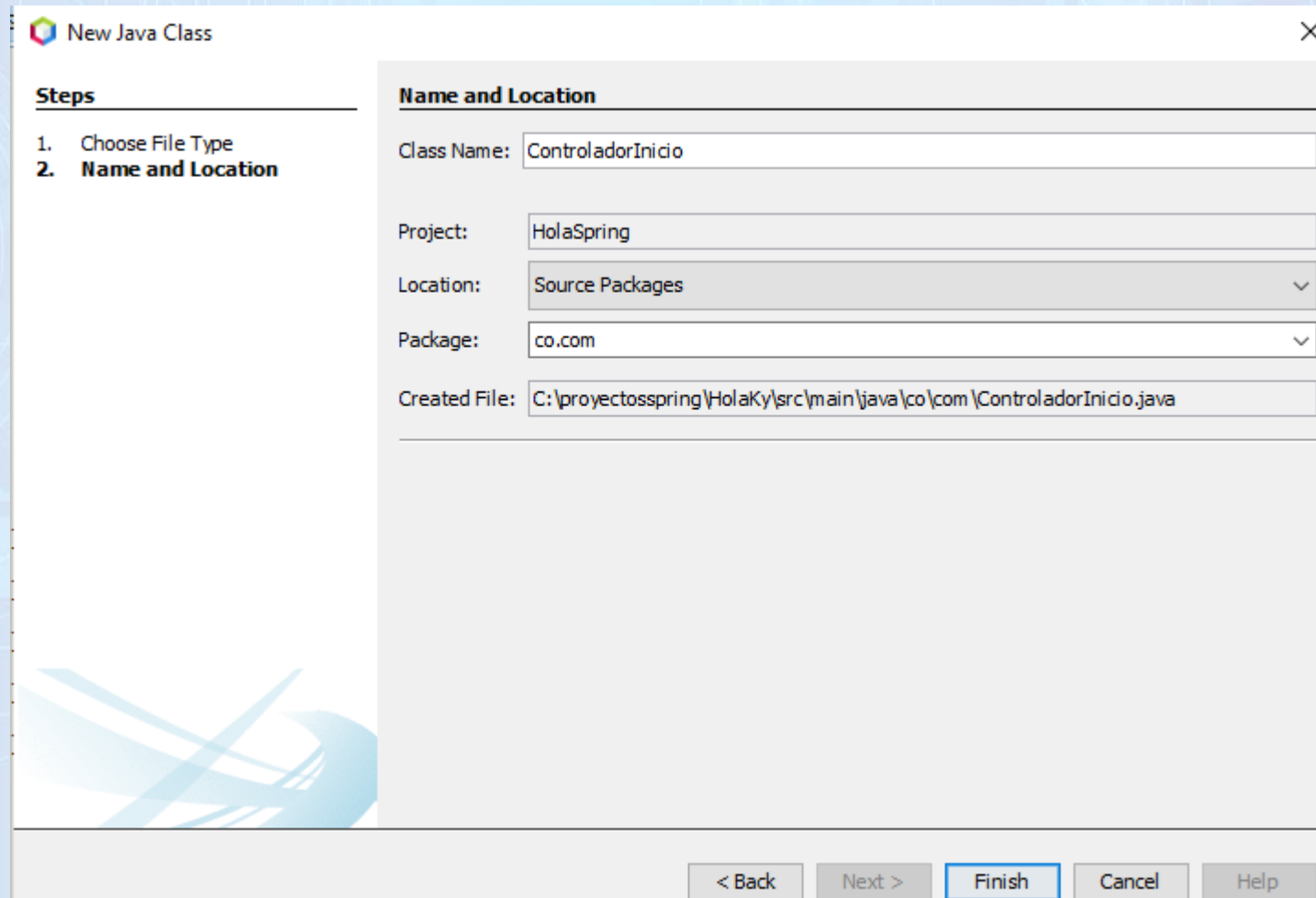
PutMapping

PatchMapping

DeleteMapping

# Pasos para crear un “Hola Mundo” con Spring

1. Creamos una clase controlador inicio, dentro del mismo paquete de la clase principal o dentro de un subpaquete



New Java Class

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

Class Name:

Project:

Location:

Package:

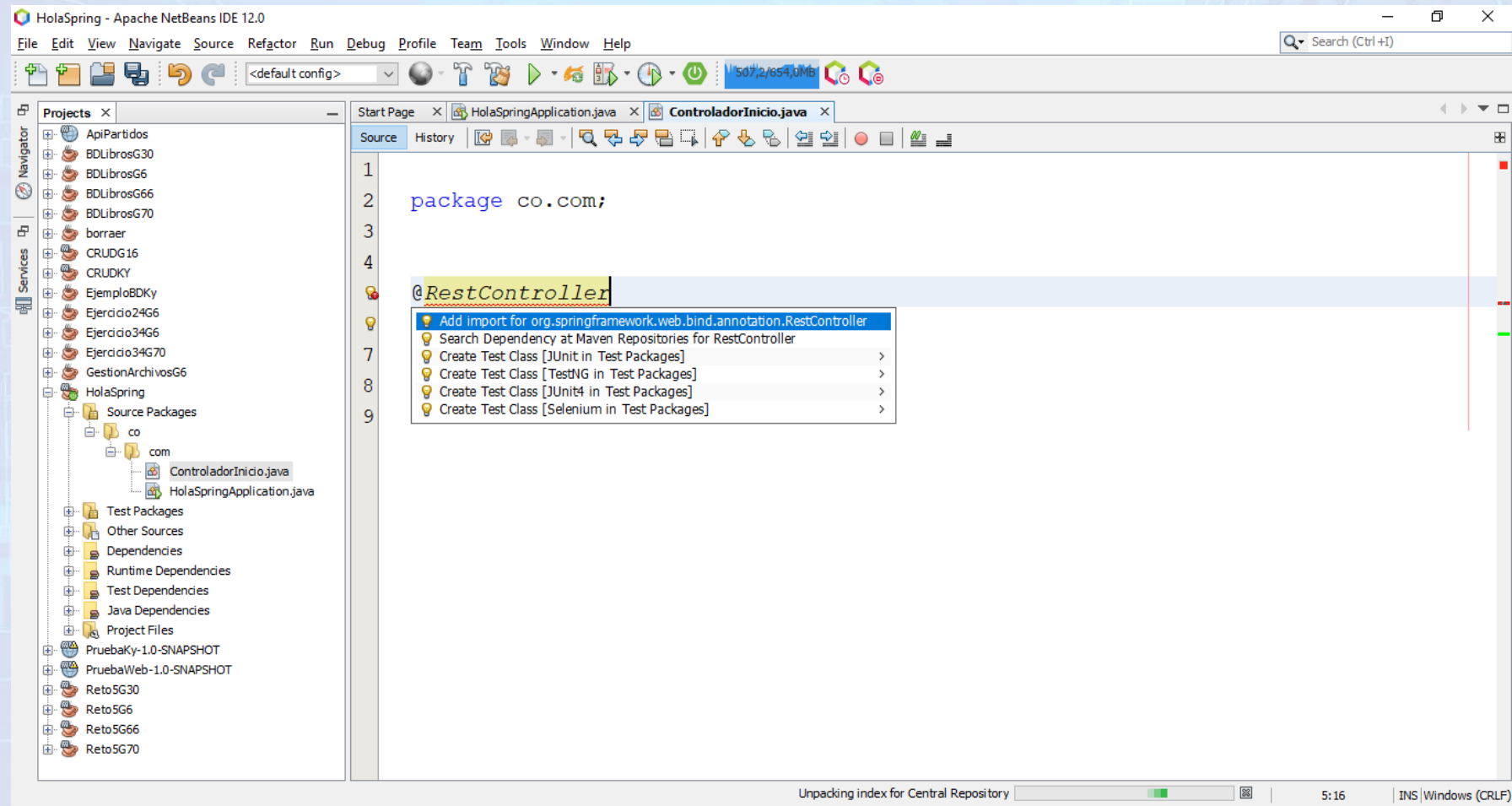
Created File:

< Back   Next >   **Finish**   Cancel   Help



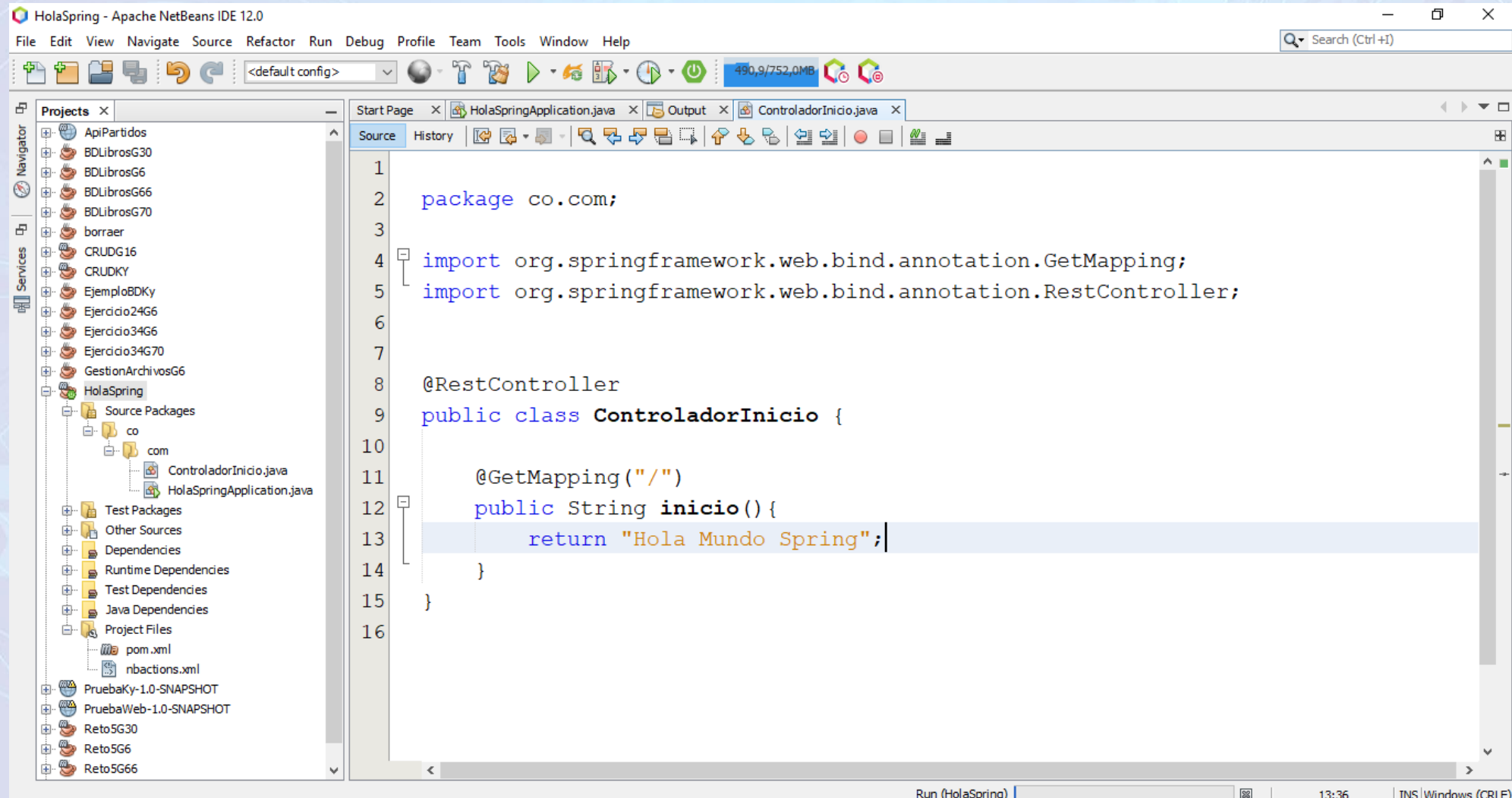
# Pasos para crear un “Hola Mundo” con Spring

## 2. Añadimos el RestController



# Pasos para crear un “Hola Mundo” con Spring

## 3. Creamos el contenido de la clase

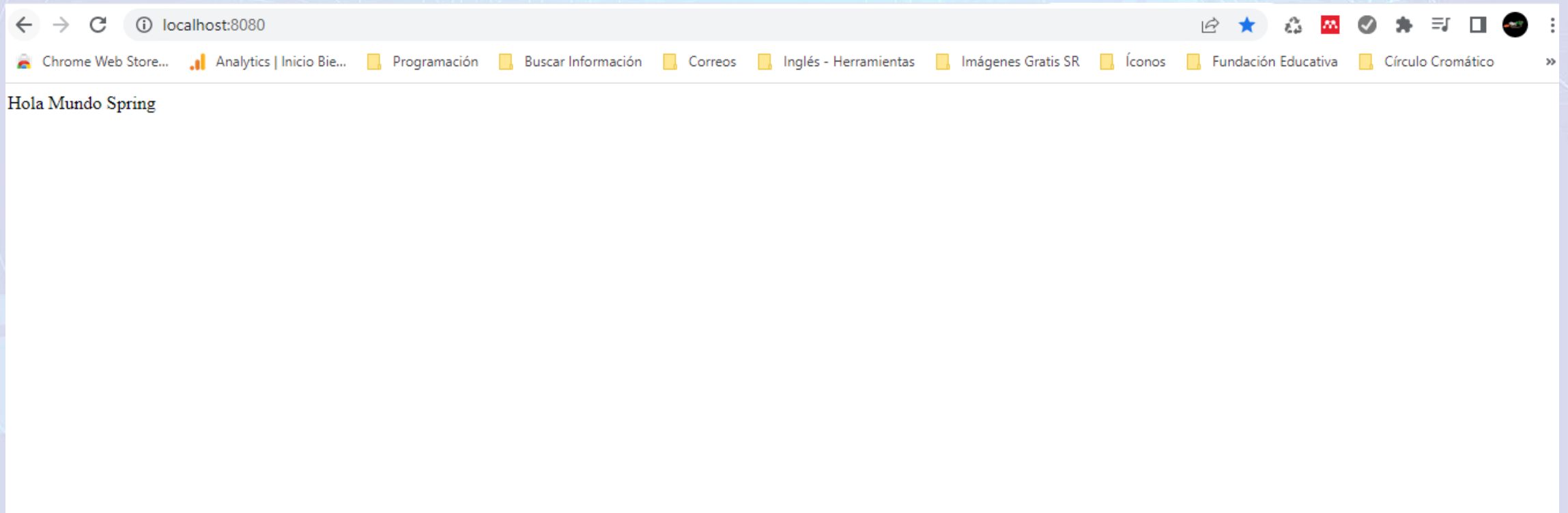



## 4. Ejecutamos



# Pasos para crear un “Hola Mundo” con Spring

5. En nuestro navegador favorito abrimos <http://localhost:8080/>



The background of the slide features a light blue and purple gradient. Overlaid on this are faint, stylized illustrations of mechanical gears and electronic circuitry, including lines representing wires and various geometric shapes like hexagons and rectangles, suggesting a technical or engineering theme.

**Listo, si has llegado hasta aquí, ya tienes configurado el Framework de Spring Boot en Netbeans....**



The background is a light blue gradient with faint, stylized white line art. It features several interlocking gears of different sizes, some with teeth and others with concentric circles. There are also circuit-like patterns, including straight lines, hexagons, and curved paths, suggesting a mechanical or technological theme.

**Gracias....**