< TODAS LAS GUÍAS

Guía de inicio rápido de Spring

Lo que vas a construir

Construirás un clásico "¡Hola mundo!" punto final al que cualquier navegador puede conectarse. Incluso puedes decirle tu nombre y te responderá de una manera más amigable.

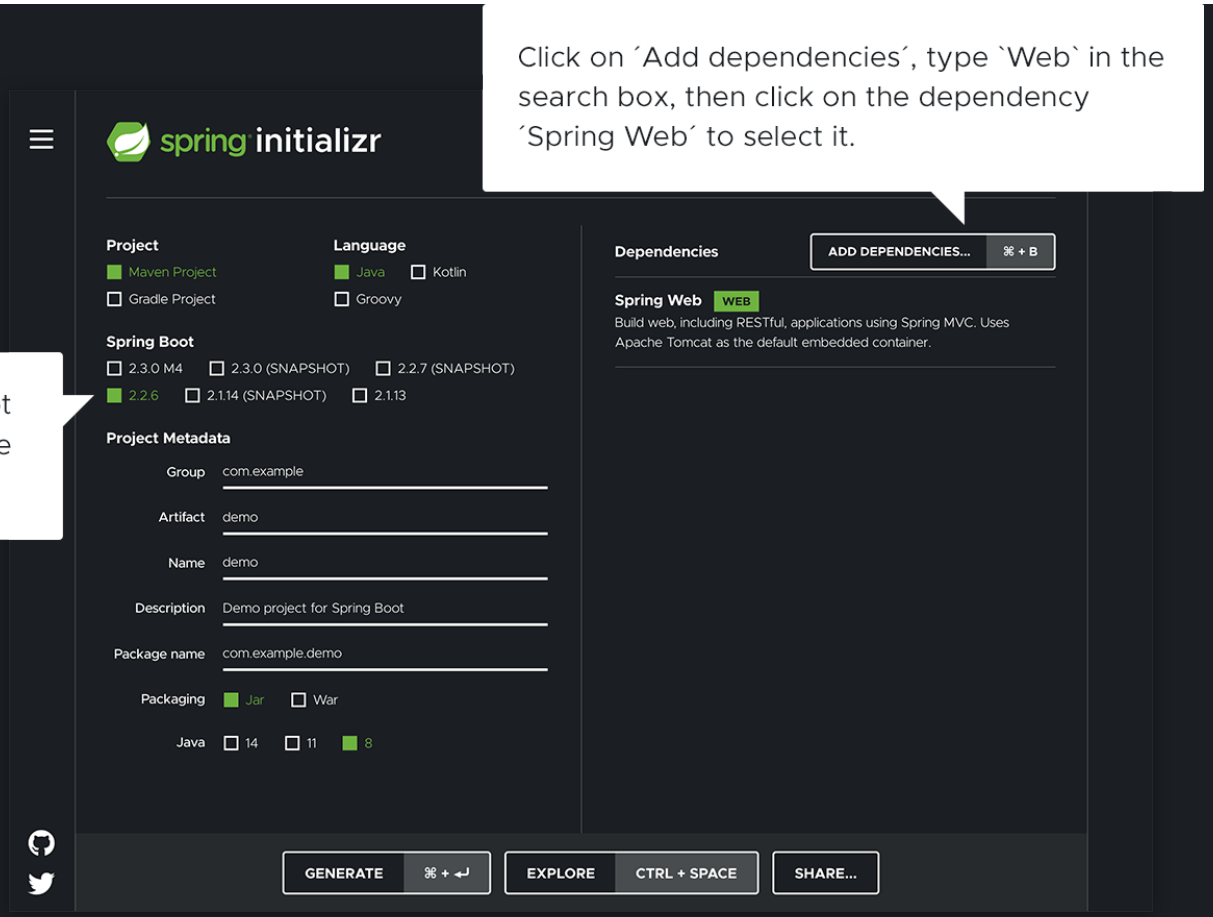
Lo que necesitarás

Un entorno de desarrollador integrado (IDE)
Las opciones populares incluyen IntelliJ IDEA, Herramientas de primavera, Código de Visual Studio, o Eclipse, y muchos más.

Un kit de desarrollo de Java ™ (JDK)
Nosotros recomendamos AdoptOpenJDK versión 8 o versión 11.

Paso 1: Inicie un nuevo proyecto de Spring Boot

Usar start.spring.iopara crear un proyecto "web". En el cuadro de diálogo "Dependencias", busque y agregue la dependencia "web" como se muestra en la captura de pantalla. Presione el botón "Generar", descargue el zip y descomprímalo en una carpeta en su computadora.



Proyectos creados por start.spring.io Contiene Bota de primavera, un marco que hace que Spring esté listo para funcionar dentro de su aplicación, pero sin mucho código o configuración requerida. Spring Boot es la forma más rápida y popular de iniciar proyectos de Spring.

Paso 2: agrega tu código

Abra el proyecto en su IDE y busque el `DemoApplication.java` archivo en la `src/main/java/com/example/demo` carpeta. Ahora cambie el contenido del archivo agregando el método adicional y las anotaciones que se muestran en el código a continuación. Puede copiar y pegar el código o simplemente escribirlo.

```
DUPDO

package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication
@RestController
public class DemoApplication {

   public static void main(String[] args) {
      SpringApplication.run(DemoApplication.class, args);
   }

   @GetMapping("/hello")
   public String hello(@RequestParam(value = "name", defaultValue = "World") String name) {
      return String.format("Hello %s!", name);
   }
}
```

Este es todo el código necesario para crear un servicio web sencillo "Hello World" en Spring Boot.

El `hello()` método que hemos agregado está diseñado para tomar un parámetro String llamado `name` y luego combinar este parámetro con la palabra `"Hello"` en el código. Esto significa que si establece su nombre en `“Amy”` en la solicitud, la respuesta sería `“Hello Amy”` .

La `@RestController` anotación le dice a Spring que este código describe un punto final que debería estar disponible en la web. Le `@GetMapping("/hello")` dice a Spring que use nuestro `hello()` método para responder a las solicitudes que se envían a la `http://localhost:8080/hello` dirección. Finalmente, le `@RequestParam` está diciendo a Spring que espere un `name` valor en la solicitud, pero si no está allí, usará la palabra "Mundo" por defecto.

Paso 3: Pruébelo

Construyamos y ejecutemos el programa. Abra una línea de comando (o terminal) y navegue hasta la carpeta donde tiene los archivos del proyecto. Podemos construir y ejecutar la aplicación emitiendo el siguiente comando:

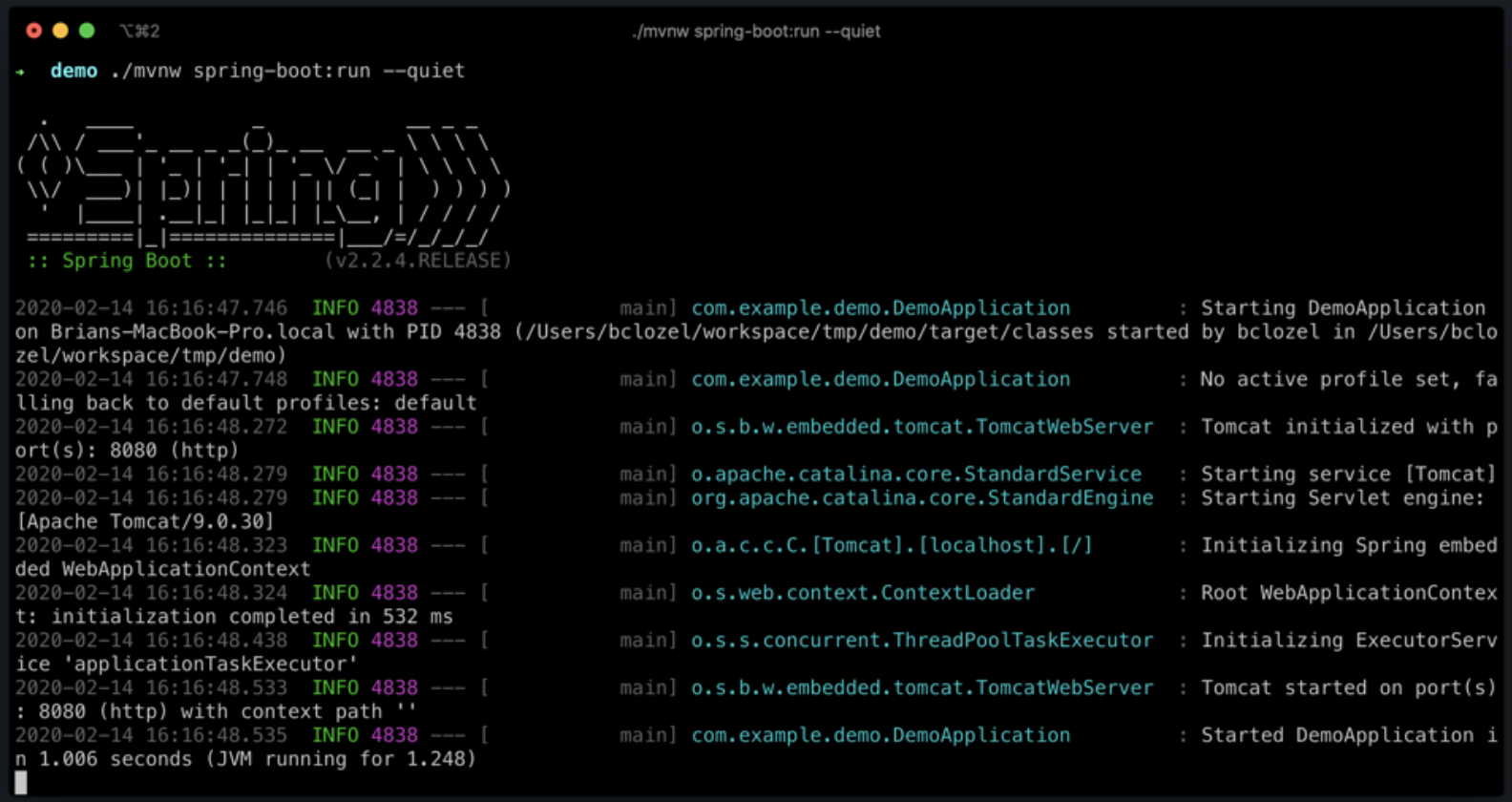
MacOS / Linux:

```
./mvnw spring-boot:run
```

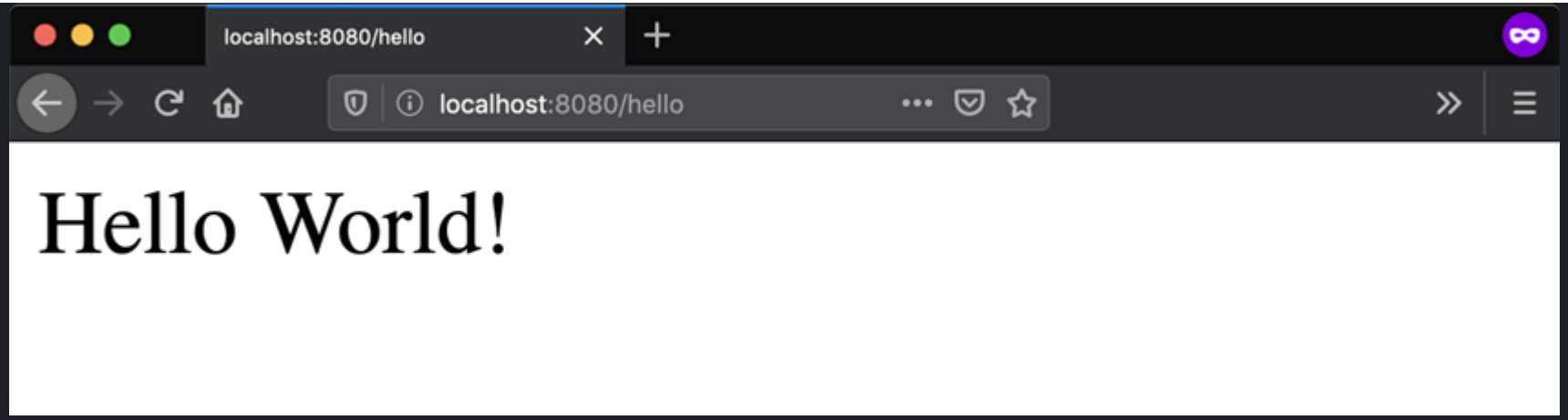
Ventanas:

```
mvnw spring-boot:run
```

Debería ver un resultado que se parece mucho a esto:



Las últimas dos líneas aquí nos dicen que la primavera ha comenzado. El servidor Apache Tomcat integrado de Spring Boot actúa como un servidor web y está escuchando solicitudes en el `localhost` puerto `8080`. Abra su navegador y en la barra de direcciones en la parte superior ingresehttp: // localhost: 8080 / hola. Debería obtener una respuesta agradable y amistosa como esta:



Examen sorpresa

¿Qué debería suceder si agrega `?name=Amy` al final de la URL?

A continuación, pruebe estas guías populares

Ya has visto lo simple que puede ser Spring, pero también es muy flexible. Hay miles de cosas que puede hacer con Spring, y tenemos muchas guías disponibles para llevarlo a través de las opciones más populares. ¿Por qué no seguir aprendiendo y probar una de estas guías adicionales?

Construyendo un servicio web RESTful

Continúe su aprendizaje creando un servicio web RESTful JSON en Spring

Consumir un servicio web RESTful

Aprenda a recuperar datos de páginas web con RestTemplate de Spring.

Acceder a datos con JPA

Aprenda a trabajar con la persistencia de datos JPA usando Spring Data JPA.