



Springboot – Contrôleur REST (Hello world)



Créez une nouvelle application Spring boot

On génère maintenant un projet correspondant à :

```
groupId com.demo
```

```
artifactId firstSpringBootTestApp
```

```
version 0.0.1-SNAPSHOT
```

```
java.version 11
```

Prévoir la dépendance (comme précédemment) correspondant à WEB

```
<dependency>
```

```
    <groupId>org.springframework.boot</groupId>
```

```
    <artifactId>spring-boot-starter-web</artifactId>
```

```
</dependency>
```



Rappel Spring

- Pré requis : quelles dépendances ?

Rappel Spring

- Pré requis : quelles dépendances ?

- On va :

- créer un package `...rest`

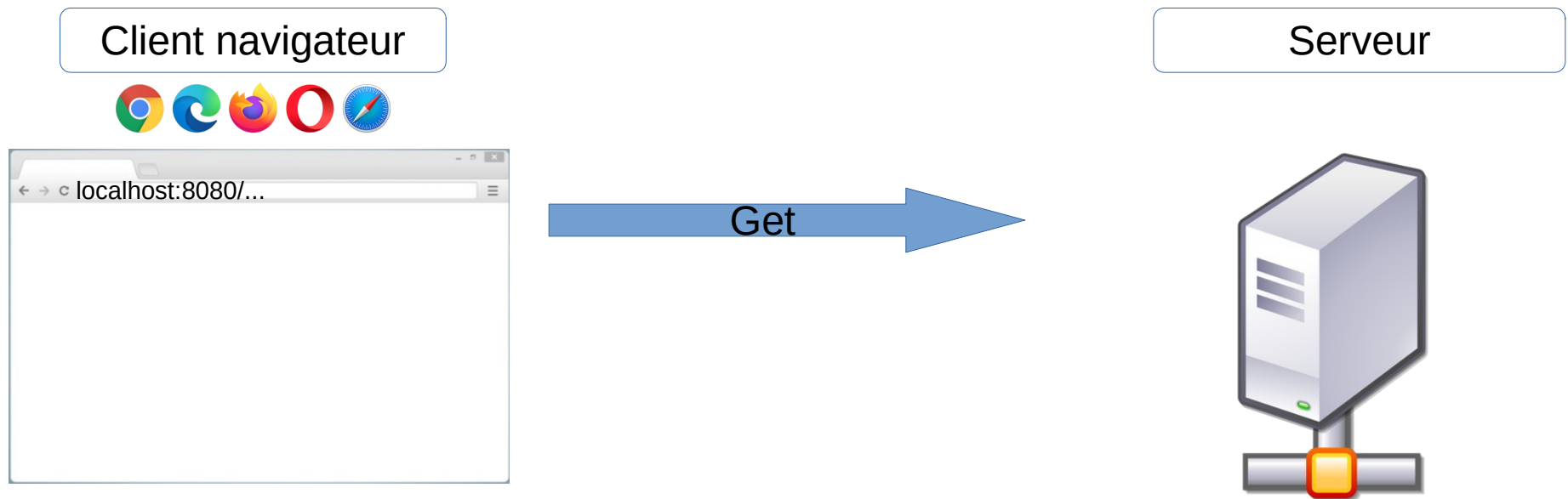


- créer une classe (le controleur REST) à l'intérieur



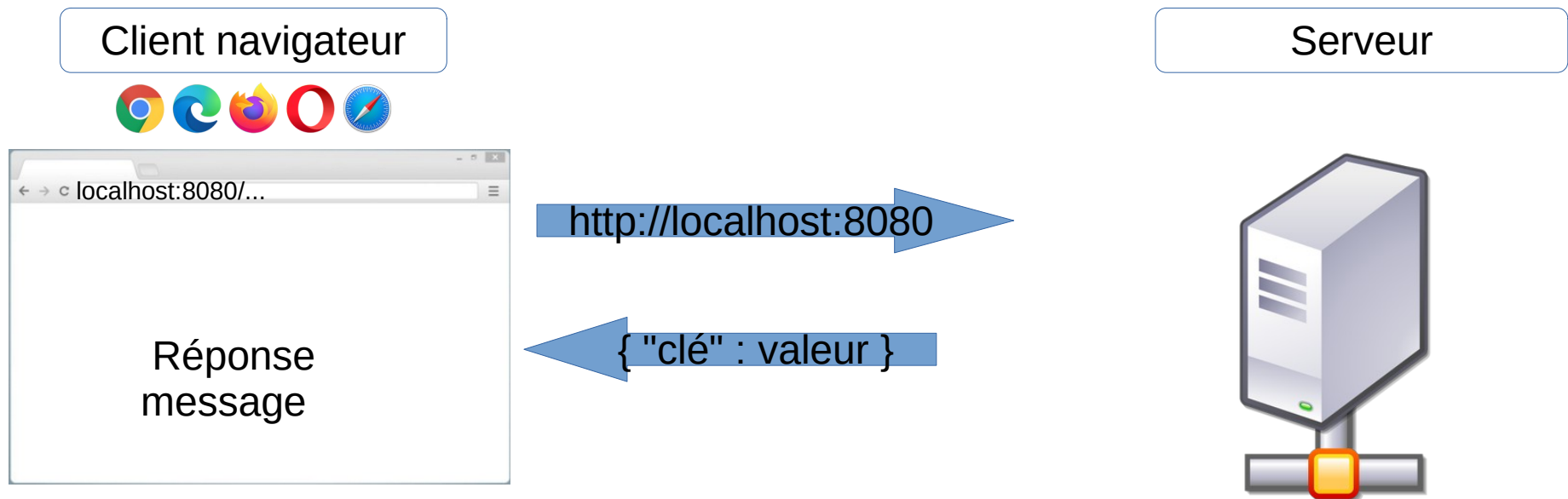
Rappel

un contrôleur REST avec Spring



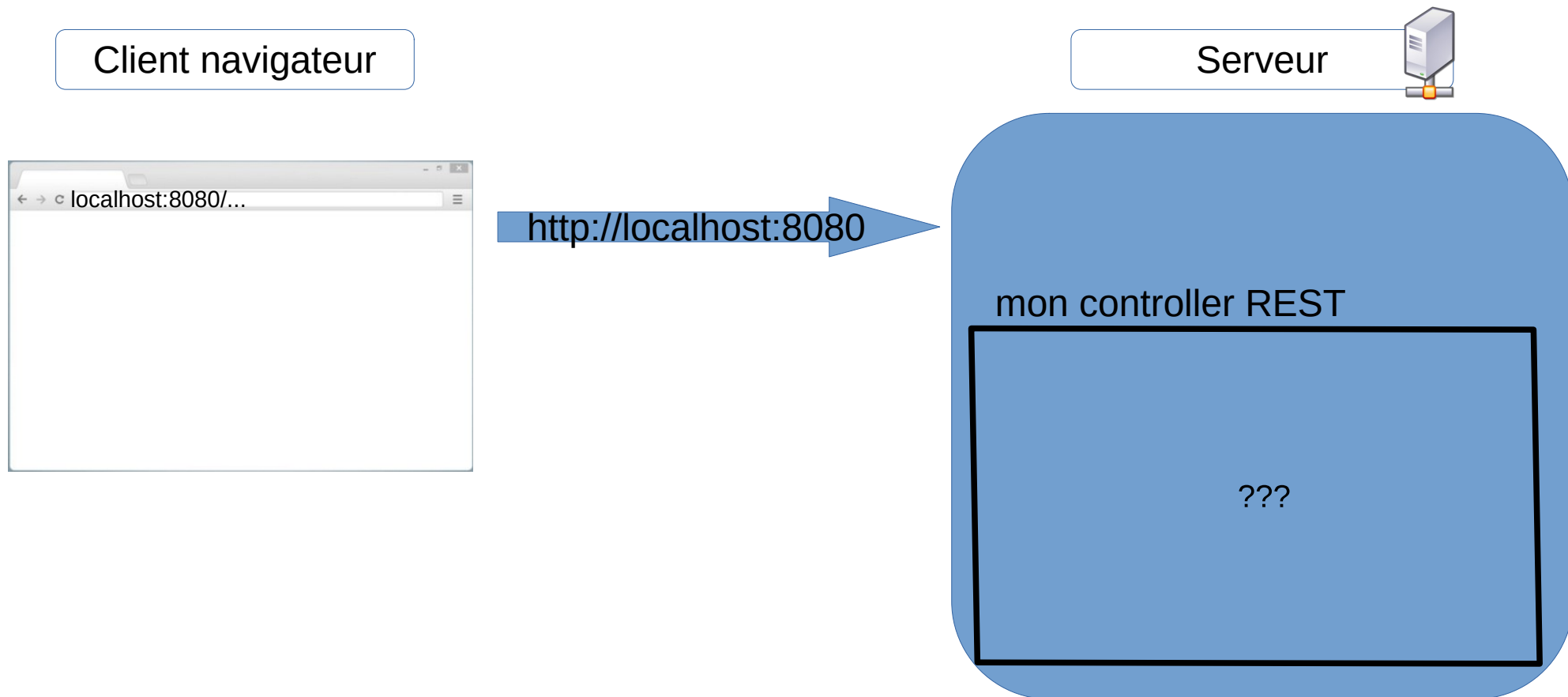
Rappel

un contrôleur REST avec Spring



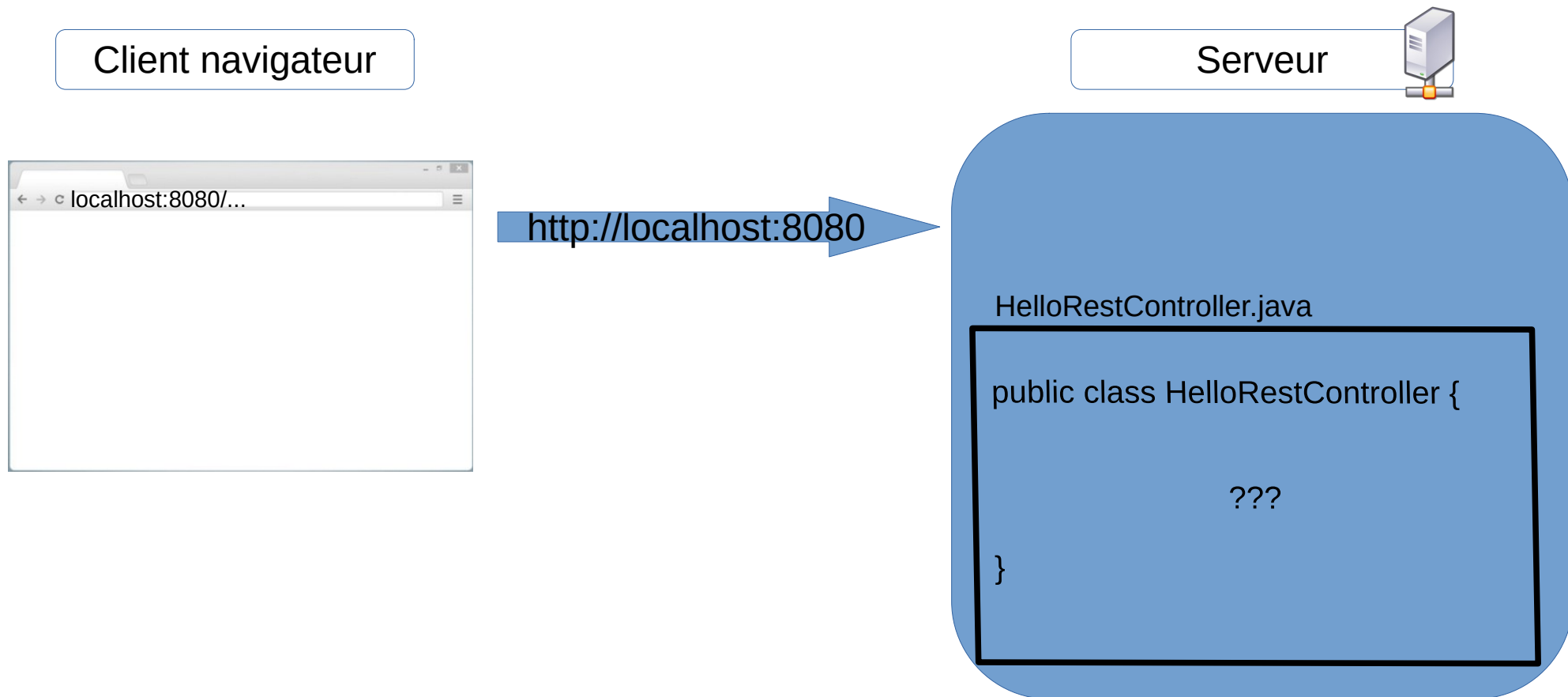
Rappel

un contrôleur REST avec Spring



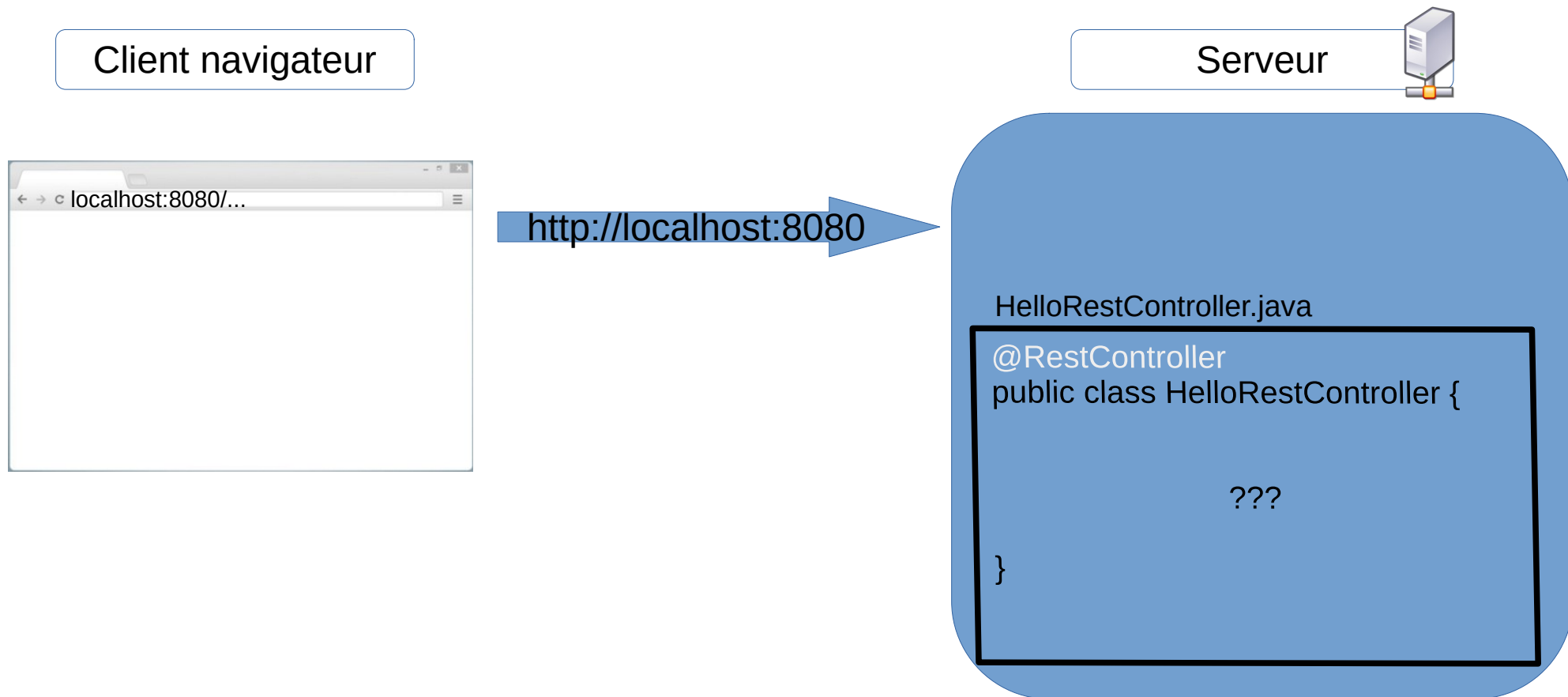
Rappel

un controleur REST avec Spring



Rappel

un controleur REST avec Spring



Rappel

un controleur REST avec Spring

Client navigateur



http://localhost:8080

Serveur



HelloRestController.java

```
@RestController
public class HelloRestController {
    ...

    public void sayHello(){
        System.out.println("Hello SpringBoot !") ;
    }

    ...
}
```

Rappel

un controleur REST avec Spring

Client navigateur



http://localhost:8080

Serveur

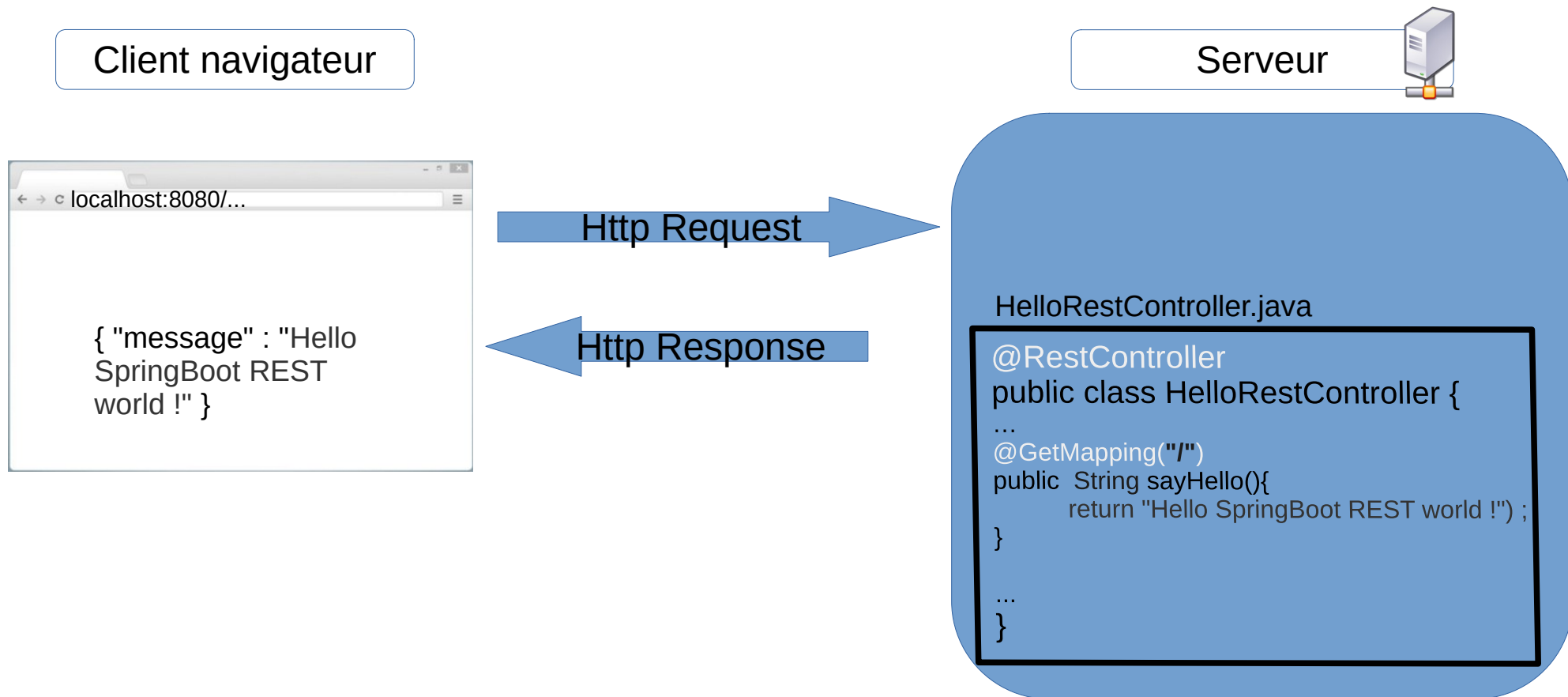


HelloRestController.java

```
@RestController
public class HelloRestController {
    ...
    @GetMapping("/")
    public void sayHello(){
        System.out.println("Hello SpringBoot !") ;
    }
    ...
}
```

Rappel

un controleur REST avec Spring



Rappel Spring

- Pré requis : quelles dépendances ?
- On va :
 - créer un package `...rest`
 - créer une classe (le controleur REST) à l'intérieur
- On va avoir besoin de :
 - `@RestController`
 - `@GetMapping`
 - `jackson data-bind`

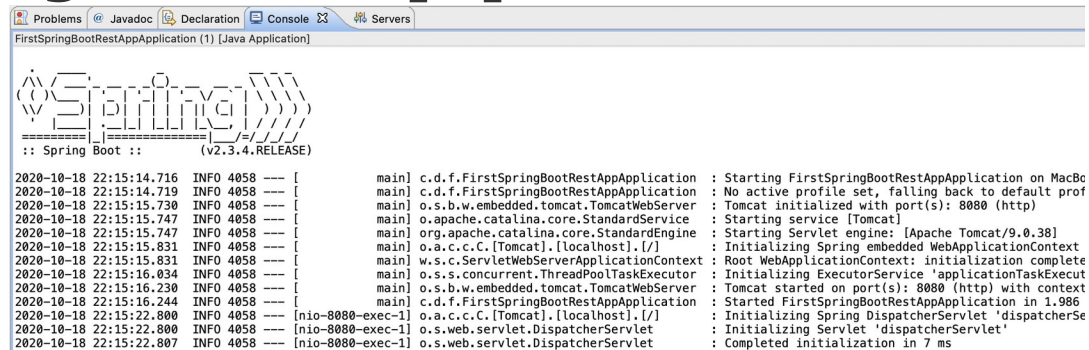
Démarrer l'application

HelloRestController.java

```
@RestController
public class HelloRestController {

    @GetMapping("/")
    public void sayHello() {
        System.out.println("Hello spring Boot Rest World !");
    }
}
```

Run as java application ...



```
FirstSpringBootTestAppApplication (1) [Java Application]
:: Spring Boot :: (v2.3.4.RELEASE)

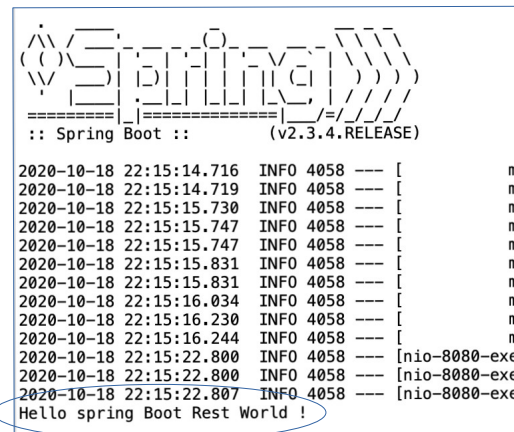
2020-10-18 22:15:14.716 INFO 4058 --- [main] c.d.f.FirstSpringBootTestAppApplication : Starting FirstSpringBootTestAppApplication on MacBe
2020-10-18 22:15:14.719 INFO 4058 --- [main] c.d.f.FirstSpringBootTestAppApplication : No active profile set, falling back to default prof
2020-10-18 22:15:15.730 INFO 4058 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2020-10-18 22:15:15.747 INFO 4058 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-18 22:15:15.747 INFO 4058 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-10-18 22:15:15.831 INFO 4058 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-10-18 22:15:15.831 INFO 4058 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization complete
2020-10-18 22:15:16.034 INFO 4058 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecut
2020-10-18 22:15:16.230 INFO 4058 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context
2020-10-18 22:15:16.244 INFO 4058 --- [main] c.d.f.FirstSpringBootTestAppApplication : Started FirstSpringBootTestAppApplication in 1.986
2020-10-18 22:15:22.800 INFO 4058 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherSe
2020-10-18 22:15:22.800 INFO 4058 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2020-10-18 22:15:22.807 INFO 4058 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 7 ms
```

Client navigateur



http://localhost:8080

Serveur



```

:: Spring Boot :: (v2.3.4.RELEASE)

2020-10-18 22:15:14.716 INFO 4058 --- [main] c.d.f.FirstSpringBootTestAppApplication : Starting FirstSpringBootTestAppApplication on MacBe
2020-10-18 22:15:14.719 INFO 4058 --- [main] c.d.f.FirstSpringBootTestAppApplication : No active profile set, falling back to default prof
2020-10-18 22:15:15.730 INFO 4058 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2020-10-18 22:15:15.747 INFO 4058 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-18 22:15:15.747 INFO 4058 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-10-18 22:15:15.831 INFO 4058 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-10-18 22:15:15.831 INFO 4058 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization complete
2020-10-18 22:15:16.034 INFO 4058 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecut
2020-10-18 22:15:16.230 INFO 4058 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context
2020-10-18 22:15:16.244 INFO 4058 --- [main] c.d.f.FirstSpringBootTestAppApplication : Started FirstSpringBootTestAppApplication in 1.986
2020-10-18 22:15:22.800 INFO 4058 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherSe
2020-10-18 22:15:22.800 INFO 4058 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2020-10-18 22:15:22.807 INFO 4058 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 7 ms

Hello spring Boot Rest World !
```

Retourner du json

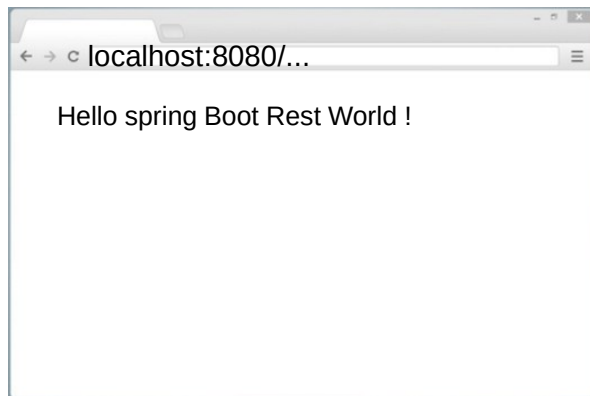
HelloRestController.java

```
@RestController
public class HelloRestController {

    @GetMapping("/")
    public String sayHello() {
        return "Hello spring Boot Rest World !";
    }
}
```

Après redémarrage de l'application Spring boot, on peut constater le fonctionnement comme attendu du controleur Rest de notre application Spring Boot

Client navigateur



http://localhost:8080

Response { ... }

Serveur



A retenir

- Nous n'avons pas configuré les dépendances
- Nous n'avons pas configuré beaucoup de choses
- Spring boot utilise l'autoconfiguration basée sur des fichiers de propriétés et le classpath standard
- Nous n'avons pas mis en route de server
- Nous n'avons pas surveillé les versions des différentes dépendances gérées par Maven

Dispatcher Servlet autoconfigurée

- NB l'une des principales fonctions de spring boot est l'autoconfiguration. L'autoconfiguration de Spring Boot enregistre et configure la DispatcherServlet automatiquement. Par conséquent il est inutile de déclarer manuellement un bean , ou une servlet DispatcherServlet.
- Par défaut, le spring-boot-starter-web configure DispatcherServlet sur l' URL pattern `"/"`.
- Il est toutefois possible de spécifier une URL pattern différente avec la clé `server.servlet.*` dans le fichier `application.properties` :

```
server.servlet.context-path=/demo
```

```
spring.mvc.servlet.path=/blabla
```

- Ainsi configuré, url pattern= `/blabla`, et le root context-path = `/demo`, la dispatcher servlet écoutera les requêtes sur :

```
http://localhost:8080/demo/blabla
```