

19/05/2023

Version 4



SPRING

JÉRÉMY PERROUULT

A decorative wavy line in light blue and white, running vertically along the left side of the slide.

SPRING TEST

SPRING TEST AVEC JUNIT

SPRING TEST MVC

- Une classe qui va simuler l'exécution d'une « vraie » classe
 - **Mock MVC** va simuler nos classes contrôleurs
 - Sans avoir besoin de démarrer le serveur d'application

SPRING TEST MVC

- On injecte `WebApplicationContext`

```
@Autowired  
private WebApplicationContext ctx;
```

- On construit `mockMvc` (de type `MockMvc`) avec `MockMvcBuilders`
 - A partir du contexte web

```
private MockMvc mockMvc;  
  
@BeforeEach  
public void beforeEach() {  
    this.mockMvc = MockMvcBuilders.webAppContextSetup(this.ctx).build();  
}
```

SPRING TEST MVC

- Ou bien, on injecte le contrôleur qu'on test

```
@Autowired  
private HomeController ctrl;
```

- On construit mockMvc (de type MockMvc) avec MockMvcBuilders
 - A partir du contrôleur (dans ce cas, seuls les mappings du contrôleur sont accessibles)

```
private MockMvc mockMvc;  
  
@BeforeEach  
public void beforeEach() {  
    this.mockMvc = MockMvcBuilders.standaloneSetup(this.ctrl).build();  
}
```

SPRING TEST MVC

- Vérifier que l'adresse "/hello" est accessible (GET)

```
@Test
public void shouldStatusOk() throws Exception {
    this.mockMvc
        .perform(MockMvcRequestBuilders.get("/hello"))
        .andExpect(MockMvcResultMatchers.status().isOk());
}
```

- Vérifier que "/home" est accessible (GET) et retourne la vue "home"

```
this.mockMvc
    .perform(MockMvcRequestBuilders.get("/hello"))
    .andExpect(MockMvcResultMatchers.status().isOk())
    .andExpect(MockMvcResultMatchers.view().name("home"));
```

SPRING TEST MVC

- Vérifier que l'adresse `/json` (GET) retourne
 - Est un statut OK
 - Un flux **JSON**
 - Qui existe
 - Qui est un « objet »
 - Et dont l'attribut « demo » est égale à « test mvc »

```
this.mockMvc
    .perform(MockMvcRequestBuilders.get("/json"))
    .andExpect(MockMvcResultMatchers.status().isOk())
    .andExpect(MockMvcResultMatchers.content().contentType(MediaType.APPLICATION_JSON))
    .andExpect(MockMvcResultMatchers.jsonPath("$").exists())
    .andExpect(MockMvcResultMatchers.jsonPath("$").isMap())
    .andExpect(MockMvcResultMatchers.jsonPath("$.demo").value("test mvc"));
```

SPRING TEST MVC

- Il faut ajouter 2 dépendances supplémentaires
 - hamcrest
 - json-path

EXERCICE

- Rédiger un test pour
 - Afficher la liste des produits
 - Ajouter un produit