Java MVC Application

Part III Custom MVC Framework – Understanding Spring MVC

Copiati proiectul https://github.com/ZeroToHero-2015/JavaMVCApp prin comanda **fork**:



Clonati proiectul obtinut prin comanda fork cu ajutorul comenzii **git clone**.

1. Adaugare dependinte framework Spring MVC

<spring.version>4.1.6.RELEASE</spring.version>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

## Pentru integrarea aplicatie web cu framework-ul Spring, configurati DispatcherServlet-ul.

## Creati clasa *EmployeeController*, cu doua metode: afisarea tuturor angajatilor si afisarea unui angajat dupa id-ul de angajat primit pe request.

## Configurati framework-ul Spring indicandu-i pachetul de clase in care se afla Application Controllerele (utilizati un fisier *mvc-dispatcher-servlet.xml*, pe care il salvati in WEB\_INF).

## Configurati metoda de afisare a tuturor angajatilor din clasa *EmployeeController* astfel incat sa raspunda la requestul */employees/all*, metoda HTTP – GET.

## Introduceti dependentele specific e librariei de Jackson in proiect:

## <dependency>

## <groupId>com.fasterxml.jackson.core</groupId>

## <artifactId>jackson-core</artifactId>

## <version>2.2.3</version>

## </dependency>

## <dependency>

## <groupId>com.fasterxml.jackson.core</groupId>

## <artifactId>jackson-databind</artifactId>

## <version>2.2.3</version>

## </dependency>

## <dependency>

## <groupId>com.fasterxml.jackson.core</groupId>

## <artifactId>jackson-annotations</artifactId>

## <version>2.2.3</version>

## </dependency>

## Configurati metoda getAllEmployees din clasa *EmployeeController* asftel incat sa returneze lista de angajati in format JSON.

## Care este diferenta dintre un RequestParam si un PathParam?

## Configurati metoda de afisare a unui angajat dupa id, din clasa *EmployeeController*, astfel incat sa primeasca id-ul angajatului sub forma de RequestParam. Metoda va raspunde la url-ul */employees/one.* Cum va fi influentat url-ul la care raspunde metoda de catre PathParam-ul necesar?

## Configurati metoda de afisare a unui angajat dupa id, din clasa *EmployeeController*, astfel incat sa primeasca id-ul angajatului sub forma de PathParam. Metoda va raspunde la url-ul */employees/one.* Cum va fi influentat url-ul la care raspunde metoda de catre PathParam-ul necesar?

## Modificati clasa EmployeeController astfel incat sa proceseze urmatoarele url-uri:

- /employees (GET - find All)

               - /employees/{id} (GET - find One)

               - /employees (POST - create, primeste pe request body un obiect JSON cu informatiile despre angajatul care se creaza)

               - /employees (PUT - update, primeste pe request body un obiect JSON cu informatiile despre angajatul care se updateaza)

               - /employees/{id} (DELETE - delete)

## Creati clasa *DepartmentController* si metodele necesare, care vor procesa urmatoarele url-uri:

               - /departments (GET - find All)

               - /departments/{id} (GET - find One)

               - /departments (POST - create, body - Obj JSON)

               - /departments (PUT - update, body - Obj JSON)

               - /departments/{id} (DELETE - delete)

## Creati clasa *JobController* si metodele necesare, care vor procesa urmatoarele url-uri:

               - /jobs (GET - find All)

               - /jobs/{id} (GET - find One)

               - /jobs (POST - create, body - Obj JSON)

               - /jobs (PUT - update, body - Obj JSON)

               - /jobs/{id} (DELETE - delete)

## Creati clasa *LocationController* si metodele necesare, care vor procesa urmatoarele url-uri:

               - /locations (GET - find All)

               - /locations/{id} (GET - find One)

               - /locations (POST - create, body - Obj JSON)

               - /locations (PUT - update, body - Obj JSON)

               - /locations/{id} (DELETE - delete)