API Document

Back-end

Class:

Class Application extends SpringBootServletInitializer{}

Function: entrance of project. Start project by run Application.

Class:

Class AppConfiguration{}

Function: All configuration set here.
Parameters:
@Configutation: Declare that is configuration file.
@EnableTransactionManagement: Support Transaction management

Method Summary:

Method Summary	
DataSource	dataSource()
LocalContainerEntityManagerFactoryBean	entityManagerFactory ();
persistenceExceptionTranslationPostProcessor	exceptionTranslation();
JpaTransactionManager	transactionManager();

Method Details

Public DataSource dataSource()

Function:DataSource Definition for database connection, setting are read from the application. Properties file(Using the env object)
Parameter:
Datasource: All configuration for connecting database.

Public LocalContainerEntityManagerFactoryBean entityManagerFactory ();

Function: Declare the JPA entity manager factory
Parameters:

entityManagerFactory.setDataSource(data): provide dabase connection parameter.
entityManagerFactory.setPackagesToScan: classpath scanning of @Component, @Service, etc annotated class.
vendorAdapter: vendor adapter.
AdditionalProperties: Hibernate configutation

Public persistenceExceptionTranslationPostProcessor exceptionTranslation();

Function: PersistenceExceptionTranslationPostProcessor is a bean post processor which adds and advisor to any bean annotated with Repository so that any platform-specific exceptions are caught and then treated as one Spring's DataAccessException.

Public JpaTransactionManager transactionManager();

Function: Declare the transaction manager

Class:

Class EmployeeController{}

Parameters:

@Controller: Declare the controller
@RequestMapping: Redirect to given controller

Method Summary:

Method Summary	
String	Test()
List<Employee>	getList(String columnSort, String sortOrder)
Employee	getEmployee(String name)

Method Details

public String direct()

functions: Redirect to the given file when visit http://localhost:8080/

Parameters:

@RequestMapping: Direct to given method.

public Employee getEmployee(@RequestParam("name") String name)

function: get Employee from service by name get by URL bind data.

Parameters:

String name: name bind with URL.

@RequestMapping: Get parameters in the URL.

Employee: Object data what front end query.

public List<Employee> getList(@RequestParam("sortColumn") String sortColu

mn, @RequestParam("sortOrder") String sortOrder)

function: get list of employee through by name(“sortColumn”) in order (“sortOrder”).

Parameters: sortColumn: the name of order according to.

sortOrder: the rule of order(asc or desc);

List<Employee>: A list of employee with order what front end need.

@RequestParams: the data in the URL what condition for query.

Class:

Class EmployeeService{}

parameters:

@Service: Declare this is service.

Method Summary:

Method Summary:

List<Employee>	getList(String columnSort, String sortOrder)
Employee	getEmployee(String name)

Method Details

Public List<Employee> getEmployeeList(String sortColumn, String sortOrder)
function: get list of employee through by name(“sortColumn”) in order (“sortOrder”).
Parameters: sortColumn: the name of order according to.
sortOrder: the rule of order(asc or desc);
List<Employee>: A list of employee with order what front end need.
Return: List<Employee>

Public Employee getEmployee(String name)
function: get Employee from service by name.
Parameters:
String name: name bind with URL.
Employee: Object data controller need.
Return: Employee

Interface:

Interface EmployeeRepository extends JpaRepository<Employee, Integer>{}
Function: Provide varies list of ordered employee according to the varies condition.

Method Summary:

Method Summary	
List<Employee>	findAllByOrderByNameAsc();
List<Employee>	findAllByOrderByNameDesc();
List<Employee>	findAllByOrderByHiredateAsc();
List<Employee>	findAllByOrderByHiredateDesc();
List<Employee>	findAllByOrderBySalaryAsc ();
List<Employee>	findAllByOrderBySalaryDesc();
Employee	findByName();

Method Details

Public List<Employee> findAllByOrderByNameAsc();
Function: get list of employee by name ascending order.
Return: List<Employee>

Public List<Employee> findAllByOrderByNameDesc();
Function: get list of employee by name descending order.
Return: List<Employee>

Public List<Employee> findAllByOrderByHiredateAsc();
Function: get list of employee by hiredate ascending order.
Return: List<Employee>

Public List<Employee> findAllByOrderByHiredateAsc();
Function: get list of employee by hiredate descending order.
Return: List<Employee>

Public List<Employee> findAllByOrderBySalaryAsc();
Function: get list of employee by salary ascending order.
Return: List<Employee>

Public List<Employee> findAllByOrderBySalaryDesc();
Function: get list of employee by name descending order.
Return: List<Employee>

Public Employee findByName(String name);
Function: get specific employee by querying name.
Return: Employee

Front-end:

Function URL list:

Resource name:	URL	method
redirect	http://localhost:8080	get
getListEmployee	http://localhost:8080/getlist?sortColumn={{sortColumn}}&sortOrder={{order}}	get

getEmployeebyname	http://localhost:8080/getEm?name={{parms}}	get