9-4 規劃 JapRepository類別



Spring Data JPA Repository架構



Repository<T,K>主要用來設定CRUD Method,且方便建立一個Spring Bean進入Spring Container中

CRUDRepository<T,K>介面,規劃CRUD操作方法

PagingAndSortingRepository<T,K>介面,附加方法進行翻頁與排序用

JpaRepository<T,K>介面,實現JPA規格

- org.springframework.data.repository.Repository<T,ID>
 - org.springframework.data.repository.CrudRepository<T,ID>
 - org.springframework.data.repository.PagingAndSortingRepository<T,ID>

org.springframework.data.jpa.repository

Interface JpaRepository<T,ID>

All Superinterfaces:

CrudRepository<T,ID>, PagingAndSortingRepository<T,ID>, QueryByExampleExecutor<T>, Repository<T,ID>



設計配合規劃好的Entity 的JpaRepository 介面



建立介面繼承介面JpaRepository

泛型設定相對的Entity Class

配合相對的Method與泛型,可以自動產生相對的Native SQL進行資料庫CRUD操作

需要藉助application.properties中配置的SQL Dialect類型

自訂配合Key查詢的方法規格

• findByXxxxx(Key),必須配合Entity類別中規劃的Property Name(setXxxx/getXxxx)命名規則

使用@Repository進行描述

Spring Boot需要設定EnableRepositories Annotation進行package scan



建立JpaRepository介面



```
### Public interface CustomersRepository extends JpaRepository **Customers, String > {

//取出所有客戶資料
List **Customers ** findAll();

Customers **EindByCustomeria** (String key);
```

```
@Entity
@Table(name="Customers")
public class Customers {
    @Id
    @Column(name="customerid",nullable=false,length=5)
    private String customerId;
    @Column(name="companyname",nullable=false)
    private String companyName;
    @Column(name="address",nullable=true)
    private String address;
    @Column(name="phone",nullable=true)
    private String phone;
    @Column(name="country",nullable=true)
    private String country;
    public String getCustomerId() {
```



Spring Boot Main Program啟動Repository掃描



透過@EnableRepositories annotation掃描package

啟動自訂的Repository

```
import org.springframework.boot.SpringApplication;

//配置自動掃描元件註冊到Spring Container

@SpringBootApplication

@ComponentScan(basePackages= {"com.tibame.controller", "com.tibame.component"})

@EntityScan(basePackages="com.tibame.entity")

@EnableJpaRepositories(basePackages="com.tibame.component")

public class MywebApplication {

   public static void main(String[] args) {

        SpringApplication.run(MywebApplication.class, args);
   }

}
```



Spring MVC Controller注入JapRepository



在MVC Controller注入JpaRepository物件

進行客戶資料查詢作業

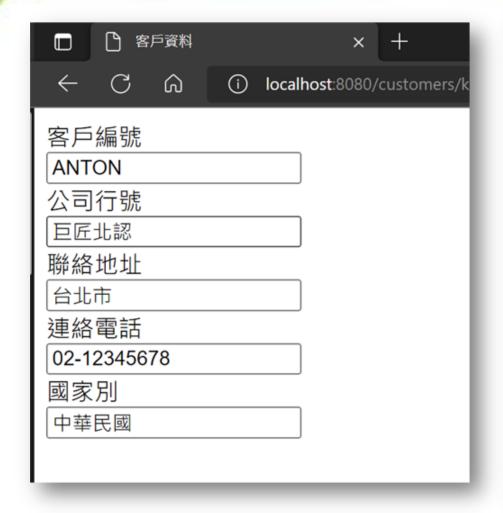
```
//客戶資料維護控制器
@Controller
@RequestMapping(path="/customers")
public class CustomersController {
    //依照型別注入依賴物件DataSource
    @Autowired
    private SQLServerDataSource datasource;
    //注入Jpa Repository
    @Autowired
    private CustomersRepository customerRep;
```

```
@GetMapping(path="/key/{cid}")
public String customersByCid(@PathVariable(name="cid")String customerId
       ,Model model) {
       com.tibame.entity.Customers customers=
                customerRep.findByCustomerId(customerId);
       model.addAttribute("customers", customers);
```



Spring MVC Controller注入JapRepository Demo









總結:9-4 規劃JapRepository類別

了解如何配設計JPA Repository設計與應用架構之後,接下來我們正式進入Sprint REST服務設計領域

0

