6. Configuration & RestTemplate

這地方會用到兩個 annotation，分別是 `@Configuration` 和 `@Bean` 來進行，先前在介紹 RESTful API 時有使用過，當初是設定 swagger 來測試自己建好的 API，來回頭看一下：

```
@Configuration
@EnableSwagger2
public class SwaggerConfig {

    @Bean
    public Docket api() {
        return new Docket(DocumentationType.SWAGGER_2)
            .apiInfo(apiInfo())
            .ignoredParameterTypes(Errors.class)
            .select()
            .apis(RequestHandlerSelectors.basePackage("com.example.demo"))
            .paths(PathSelectors.any())
            .build();
    }

    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
            .title("TEST")
            .description("TEST API")
            .version("v1")
            .build();
    }
}
```

@Component只能放在class上方，通常用在該class是你建立時(可接觸到)

根據new出來的實例是否是你可接觸到的來決定使用@Component或是@Bean

@Configuration通常會搭配@Bean一起使用，(可以有效管理不同config，而不是都放在xml檔裡面
Ex:SwaggerConfig, DataSourceConfig

在 class 上面，用了一個 `@Configuration` annotation，它主要是用來設定 Spring Boot 會用到的環境配置，像是這邊配置的 swagger、或是多個 DB 連線、還有下段會介紹用來打別人 API 的 RestTemplate 都是在這邊進行設定。而 `@Configuration` 通常都會和 `@Bean` 一起搭配使用，這樣標註為 `@Bean` 的 method，才會被 Spring Boot 的 `@ComponentScan` 給掃到放進 Bean Pool 裡。

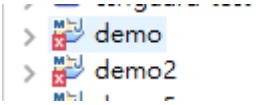
Service 層除了自己的業務邏輯外，有時資料會透過別人做好的 API 來取得，那表示要在程式裡完成在 Postman 所做的事情，Spring Boot 這邊有提供一個類別叫做 RestTemplate，這個物件目的是用來打 API。首先在 `@Configuration` 進行配置：

```
|--com.example.demospringboot
   |--DemospringbootApplication.java
|--com.example.demospringboot.configuration
   |--SwaggerConfig.java
   |--RestConfiguration.java // 新增的檔案
|--com.example.demospringboot.controller
   |--TestController.java
   |--ProductController.java
|--com.example.demospringboot.model
   |--Product.java
|--com.example.demospringboot.service
   |--ProductService.java
|--com.example.demospringboot.service.impl
   |--ProductServiceImpl.java
```

```java
@Configuration
public class RestConfiguration {

    @Bean
    public RestTemplate getRestTemplate() {
        return new RestTemplate();
    }
}
```

再來將前面建好的專案，直接複製一份出來：



改一下 demo2 專案的 `ProduceController.java` ，讓 demo2 透過 RestTemplate 打 demo 的 API：

```java
@RestController
public class ProductController {
    @Autowired
    private RestTemplate restTemplate;

    @GetMapping(value = "/products2")
    public ResponseEntity<Object> getProduct2() {
        return new ResponseEntity<>(restTemplate.getForObject("http://localhost:8081/products"
            , String.class), HttpStatus.OK);
    }

}
```

RestTemplate 其實提供了很多種打 API 的方法，這邊只是採用其中一種而已，有興趣的人可以自行去查詢還有哪些方法可以使用。再來分別執行 demo 專案跟 demo2 專案的 main 方法，執行的過程中應該會出現一個錯誤：

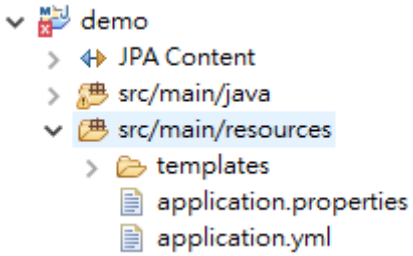
```
Description:

Web server failed to start. Port 8080 was already in use.

Action:

Identify and stop the process that's listening on port 8080 or configure this application to listen on another port.
```

console 會告訴你，目前 8080 的 port number 已經被佔用了，所以來改一下 demo 專案的 port number，讓它不會噴 port number 占用錯誤。在底下的圖片中，會看到兩個特別的檔案，一個是 `application.yml` 、一個是 `application.properties` ：



這兩個檔案可以對 Spring Boot 專案做一些參數上的設定，只是編寫的方式不太一樣而已，兩種其實都可以用，如果是寫 `application.properties` 時，編寫的方式如下：

```
server.port = 8081
```

如果是 `application.yml` ，編寫方式則如下：

```
server:
  port: 8081
```

改完以後，啟動專案就不會有問題，最後使用 Postman進行測試，這次是打 demo2 專案的 API：

Untitled Request

POST http://localhost:8080/products2

Params Authorization Headers (10) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize Text


```
1 [{"id":"1","name":"Honey"},{"id":"2","name":"Almond"}]
```