1.التعليقات التوضيحية لـ Spring Boot (Annotations)

التعليقات التوضيحية لـ Spring Boot هي شكل من أشكال البيانات الوصفية التي توفر بيانات حول البرنامج. بمعنى آخر، يتم استخدام التعليقات التوضيحية لتوفير معلومات تكميلية حول البرنامج. إنه ليس جزءًا من التطبيق الذي نطوره. ليس له تأثير مباشر على تشغيل الكود الذي يعلقون عليه. لا يغير عمل البرنامج المترجم.

جميع الـ annotations مرفقة في المرجع التالي:

https://www.javatpoint.com/spring-boot-tutorial

---

2. ملفات JSON (JavaScript object notation)

سميَ بهذا الاسم لمشابهته طريقة إنشاء الكائنات بلغة JavaScript.

2.1 ما هو JSON: هو ملف نصي يستخدم لتمثيل البيانات وتبادلها بين التطبيقات المختلفة، ومن ذلك نستنتج انه فقط طريقة لتمثيل البيانات وتناقلها وليس لغة برمجة بحد ذاته. كما يكون امتداد الملف json.

- **مثال :** fileName.json

2.2 مميزاتها :استخدمت ملفات JSON على نطاق واسع لسهولة كتابتها واستخدامها. ومن استخداماتها الاساسية أن تكون حلقة ربط بين الخادم ولغة البرمجة بحيث يتم تناقل البيانات بينهم بهذه الصيغة، ويتم كتابة الملف باستخدام الأقواس المتعرجة { }.

2.3 أنواع البيانات المتاحة :تقبل ملفات json أنواع بيانات محددة وهي String, numbers, null, boolean, : arrays and objects.

نصية (String) : سلسلة من الأحرف تمثل قيمة ما.

- **مثال :** {"title": "Hello World"}

رقم (Number) : هو رقم صحيح أو حقيقي او عشري وقد يحتوي على اشارة.

- **مثال :** {"age": 20}

قيمة منطقية :(Boolean) : قيمة قد تكون اما صحيحة (true) او خاطئة (false) وهي مستخدمة بكثرة لاعتماد الحاسب عليها.

- **مثال :** {"isAnimal": true}

قيمة فارغة (Null) : عدم وجود قيمة فعلية للمتغير.

- **مثال :** {"job" : null}

كعدم وجود وظيفة للشخص.

مصفوفة :(Array) هي سلسلة من القيم المترابطة مخزنة في متغير.

- **مثال :** {"oddNumbers" : [1, 3, 5, 7, 9]}

الكائن :(object) مجموعة من اسم للمتغيرات والقيم الخاصة بها.

مثال: {"employee":{ "id" : 100, "name" : "Sami" }}

2.4 المفتاح والقيمة

المفتاح :يمثل اسم خاص لقيمة البيانات ويتم وضعه بين علاماتي التنصيص " " ويجب أن يكون متبوعا بـ نقطتان رأسيتان : ليفصل بين الاسم وقيمة البيانات. كما تكون المفاتيح عباره عن نصوص (string) مثل Book : ,Company, Pen.

القيمة :تمثّل البيانات ويمكن أن تكون اكثر من نوع: اسم, رقم, مصفوفه... *يجب تكون القيم أنواع متاح استخدامها مثل : (string, number, object, array, Boolean or null)

- **مثال :** { "Name": "Ahmed" }

وبين أزواج المفاتيح والقيم المتعددة يتم الفصل بينهم بوضع علامة فاصله(,)

- **مثال :** {" Name": "Ali" , "Age": 23 , "id": 4433}

2.5 كائنات JSON المتداخلة (Nested JSON Objects) :

أي يمكن ان يحتوي الكائن على كائن آخر بداخلة كما في المثال التالي:

- **مثال :** { "Book" : { " BookName": " Java programming", " price": 200, " AuthorName": { " firstName": " xxxxxx", " secondName": " xxxxxx", "AuthorMajor": ["CS", "IT", "IS"] } }

مثال شامل : لنأخذ الطالب كمثال نعرض به ما تعلمناه بطريقة ملفات JSON.

{"name":{ "FristName":"Ghaida" ,"LastName":"Albohairy" }, "id":437005 , "major":"Information Technology", "courses": [2,1] , "isSenior":False , "GPA": null }

3. وحدة التحكم (Controller Layer):

هي واجهة الخادم (Server) التي من خلالها يمكن استقبال الطلبات عبر البوابات وارسال النتيجة، ولكي يقوم بمهمته لابد أن تحتوي كل بوابة على ثلاثة أشياء :

1.تحديد نوع الدالة: تحديد نوع الطلب GET, POST, PUT , DELETE باستخدام التعليق التوضيحي @....mapping

2. تحديد URL: الذي من خلاله يمكن طلب الصفحة

3. تحديد End Point: الدالة المسؤولة عن استقبال الطلب وارسال النتيجة

```
@RestController
@RequestMapping("/api/v1/message")
public class TodoController {

    @GetMapping("/message")
    public String getMessage(){
        return "Hey from Spring Boot";
    }
```

@RestController	يوضح أن هذا class نوعه controller ويقوم بإرجاع نتيجة من نوع JSON
@RequstMapping	لوضع API محدد لهذا controller
@GetMapping	لتحديد نوع Method
("/message")	تحديد URL للوصول لهذه الصفحة