Spring Boot 練功場
2020/08/16 筆記

請關注 Spring Boot 練功場活動：https://tw.kotlin.tips/dojos/springtw

本次教學內容

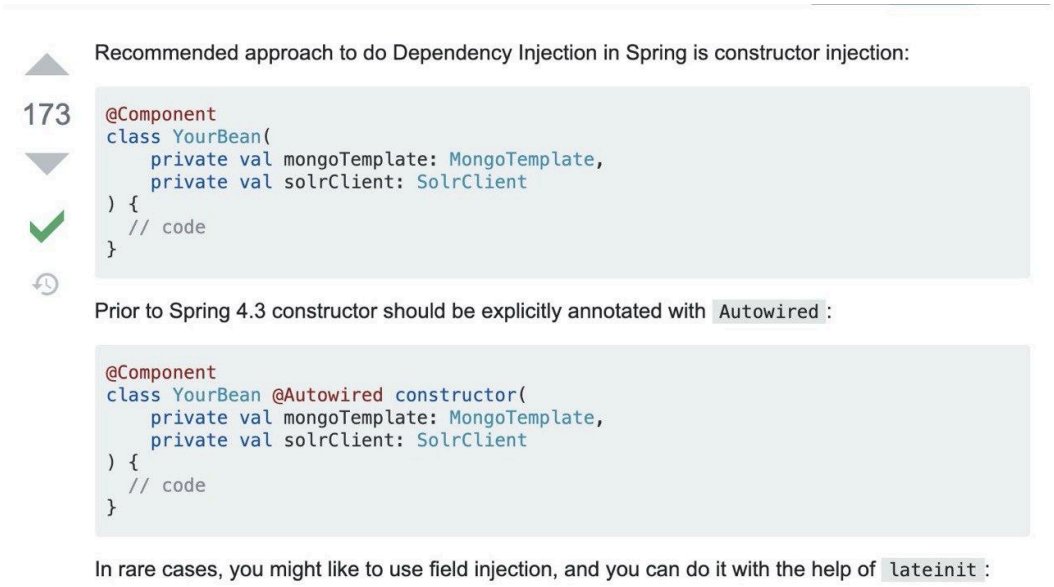
- 按圖施工，保證成功：https://github.com/b2etw/event-2020-1-kotlin/issues/7

社群朋友們的筆記

- https://medium.com/@yunghsincc/spring-%E7%B7%B4%E5%8A%9F%E5%A0%B4-2%E7%AD%86%E8%A8%98-%E7%9F%A5%E8%AD%98%E9%87%8F%E7%88%86%E7%99%BC-b944f5a3a0ec

補充資料

- @RestController 與 @Controller 的差別？
  - https://www.baeldung.com/spring-controller-vs-restcontroller
- 三種傳值方式 (https://www.baeldung.com/spring-requestparam-vs-pathvariable)
  - @RequestBody
  - @RequestParam
  - @PathVariable
- @Component、@Service、@Controller、@Repository 的差別
  - https://javarevisited.blogspot.com/2017/11/difference-between-component-service.html
- DI 與 @AutoWired
  - https://stackoverflow.com/a/35480801/90909
  - https://www.baeldung.com/spring-annotations-resource-inject-autowire



- 什麼是 Data Definition Language (DDL)？
  - https://whatis.techtarget.com/definition/Data-Definition-Language-DDL
- application.properties 怎麼設定？
  - spring.datasource.url=jdbc:mysql://localhost:3306/test
  - spring.datasource.username=root
  - spring.datasource.password=rootroot
  - spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
  -
  - spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
  - spring.jpa.hibernate.ddl-auto=update
  - spring.jpa.properties.hibernate.show_sql=true
  - spring.jpa.properties.hibernate.format_sql=true
- data migration
  - https://docs.spring.io/spring-boot/docs/2.1.1.RELEASE/reference/html/howto-database-initialization.html
- Kotlin 裡的 Scope Function
  - https://julianchu.net/2018/05/05-kotlin.html
  - https://blog.codecentric.de/en/2019/07/lets-also-apply-run-with-kotlin-scope-functions/
- 怎麼設定 MySQL 的 Docker？文件有詳細說明
  - https://hub.docker.com/_/mysql
- 用 Kotlin 的 Scope Function 這樣寫，會不會很難寫測試？
  - 答：你是測 method 的皮，不看他的內容怎麼寫。寫法只影響風格。人工字打得越少，錯誤越少 XD
- Java Mapping Framework
  - https://www.baeldung.com/java-performance-mapping-frameworks
- dto to entity 可以用 bean utils 複製

    - https://matthung0807.blogspot.com/2019/10/spring-beanutilscopyproperties-to-copy.html
- Spring JPA
    - **To configure the data source using a properties file, we have to set properties prefixed with *spring.datasource*:**

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.username=mysqluser
spring.datasource.password=mysqlpass
spring.datasource.url=
  jdbc:mysql://localhost:3306/myDb?createDatabaseIfNotExist=true
```

    - https://www.baeldung.com/the-persistence-layer-with-spring-and-jpa
- entity 可能的來源之一。
    - ER模型 [編輯]

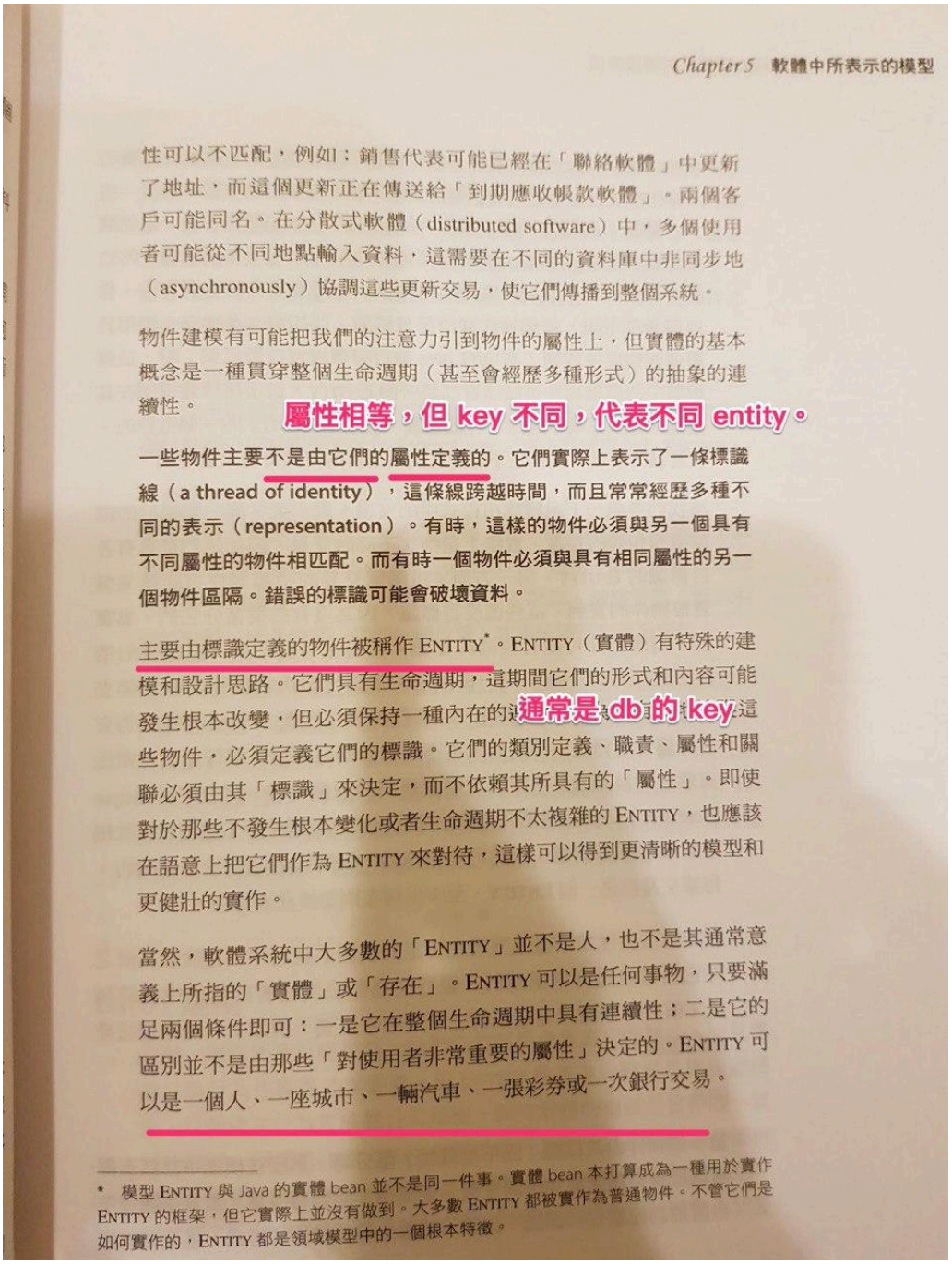
      維基百科，自由的百科全書

      提示：本條目的主題不是ER隨機圖。

      ER模型，全稱為**實體聯絡模型**、**實體關係模型**或**實體聯絡模式圖（ERD）**（英語：Entity–relationship model）由台裔美籍電腦科學家陳品山發明，是概念資料模型的高層描述所使用的資料模型或模式圖。

      ER模型常用於資訊系統設計中；比如它們在概念結構設計階段用來描述資訊需求和／或要儲存在資料庫中的資訊的類型。但是資料建模技術可以用來描述特定論域（就是感興趣的區域）的任何本體（就是對使用的術語和它們的聯絡的概述和分類）。在基於資料庫的資訊系統設計的情況下，在後面的階段（通常叫做邏輯設計），概念模型要對映到邏輯模型如關係模型上；它依次要在物理設計期間對映到物理模型上。注意，有時這兩個階段被一起稱為「物理設計」。

      實體聯絡模式圖（ERD）有一些約定。本文的餘下部分描述經典概念，並且主要與概念建模有關。有一些概念更加典型的在邏輯和物理資料庫設計中採用，包括資訊工程、IDEF1x（ICAM DEFinition Language）和空間建模。
    - https://zh.wikipedia.org/wiki/ER%E6%A8%A1%E5%9E%8B
- entity pattern from DDD
    - https://docs.spring.io/spring-boot/docs/current/reference/html/appendix-application-properties.html#data-properties



性可以不匹配，例如：銷售代表可能已經在「聯絡軟體」中更新了地址，而這個更新正在傳送給「到期應收帳款軟體」。兩個客戶可能同名。在分散式軟體（distributed software）中，多個使用者可能從不同地點輸入資料，這需要在不同的資料庫中非同步地（asynchronously）協調這些更新交易，使它們傳播到整個系統。

物件建模有可能把我們的注意力引到物件的屬性上，但實體的基本概念是一種貫穿整個生命週期（甚至會經歷多種形式）的抽象的連續性。

屬性相等，但 key 不同，代表不同 entity。

一些物件主要不是由它們的屬性定義的。它們實際上表示了一條標識線（a thread of identity），這條線跨越時間，而且常常經歷多種不同的表示（representation）。有時，這樣的物件必須與另一個具有不同屬性的物件相匹配。而有時一個物件必須與具有相同屬性的另一個物件區隔。錯誤的標識可能會破壞資料。

主要由標識定義的物件被稱作 ENTITY*。ENTITY（實體）有特殊的建模和設計思路。它們具有生命週期，這期間它們的形式和內容可能發生根本改變，但必須保持一種內在的連續性。對於這些物件，必須定義它們的標識。它們的類別定義、職責、屬性和關聯必須由其「標識」來決定，而不依賴其所具有的「屬性」。即使對於那些不發生根本變化或者生命週期不太複雜的 ENTITY，也應該在語意上把它們作為 ENTITY 來對待，這樣可以得到更清晰的模型和更健壯的實作。

通常是db的key

當然，軟體系統中大多數的「ENTITY」並不是人，也不是其通常意義上所指的「實體」或「存在」。ENTITY 可以是任何事物，只要滿足兩個條件即可：一是它在整個生命週期中具有連續性；二是它的區別並不是由那些「對使用者非常重要的屬性」決定的。ENTITY 可以是一個人、一座城市、一輛汽車、一張彩券或一次銀行交易。

* 模型 ENTITY 與 Java 的實體 bean 並不是同一件事。實體 bean 本打算成為一種用於實作 ENTITY 的框架，但它實際上並沒有做到。大多數 ENTITY 都被實作為普通物件。不管它們是如何實作的，ENTITY 都是領域模型中的一個根本特徵。

---

data.entity-> 跟table有關，會幫你create table
Repository直接繼承JpaRepository就有實作好的方法了

@RequestParam是查詢用的，API設計上是回傳多個結果，但id預期是唯一的

泛型的第2個類別是根據User的id嗎
對喔

應該可以用@Primary