# Link học chính của spring boot là link doc của nó:

<https://docs.spring.io/spring-boot/docs/2.1.0.M1/reference/html/howto-database-initialization.html>

# Dưới đây là các tên + link học java sắp xếp có vẻ không đúng thứ tự.

<https://www.tutorialspoint.com/java_dip/java_buffered_image.htm>

## Lập trình đa luồng với CompletableFuture trong Java 8

<https://viblo.asia/p/lap-trinh-da-luong-voi-completablefuture-trong-java-8-6J3ZgBMLKmB>

<https://gpcoder.com/3565-lap-trinh-da-luong-voi-callable-va-future-trong-java/>

\*\*

<https://stackjava.com/java8/completablefuture-la-gi-code-vi-du-java-completablefuture-java-8.html>

## Từ khóa Transient

<https://viettuts.vn/java-io/tu-khoa-transient-trong-java>

**Từ khóa transient trong java** được sử dụng trong serialization. Nếu bạn định nghĩa bất kỳ thành viên dữ liệu nào là transient, nó sẽ không được đánh dấu là tuần tự (serialize).

## Từ khóa Seriable trong java

<https://stackjava.com/java/java-serializable-la-gi-serialization-va-deserialization-trong-java.html>

* Serialization trong Java là cơ chế chuyển đổi trạng thái của một đối tượng (giá trị các thuộc tính trong object) thành một chuỗi byte sao cho chuỗi byte này có thể chuyển đổi ngược lại thành một đối tượng.
* Quá trình chuyển đổi chuỗi byte thành đối tượng gọi là deserialization.
* Một object có thể serializable (có thể thực hiện Serialization) nếu class của nó thực hiện implements interface java.io.Serializable
* [Home](https://stackjava.com/) » [Java](https://stackjava.com/category/java) » Java Serializable là gì? Serialization và Deserialization trong Java

Java Serializable là gì? Serialization và Deserialization trong Java

* Posted on [*Tháng Hai 8, 2018*](https://stackjava.com/java/java-serializable-la-gi-serialization-va-deserialization-trong-java.html)
* Java Serializable là gì? Serialization và Deserialization trong Java.
* Khi lập trình với Java chắc hẳn bạn đã bắt gặp khái niệm serialize nhất là khi đọc ghi object ra file, mapping với cơ sở dữ liệu…

Java Serializable là gì?

* Serialization trong Java là cơ chế chuyển đổi trạng thái của một đối tượng (giá trị các thuộc tính trong object) thành một chuỗi byte sao cho chuỗi byte này có thể chuyển đổi ngược lại thành một đối tượng.
* Quá trình chuyển đổi chuỗi byte thành đối tượng gọi là deserialization.
* Một object có thể serializable (có thể thực hiện Serialization) nếu class của nó thực hiện implements interface java.io.Serializable

Tại sao cần Serialization?

* Trong Java, khi trao đổi dữ liệu giữa các thành phần khác nhau (giữa các module cùng viết bằng Java) thì dữ liệu được thể hiện dưới dạng byte chứ không phải là đối tượng. Do đó ta cần có một cơ chế để hiểu các đối tượng được gửi và nhận.

## ****Khái niệm ThreadPool và Executor trong Java****

<https://kipalog.com/posts/Kha-i-nie--m-ThreadPool-va--Executor-trong-Java>

## ServletUriComponentsBuilder

## Vi-du-cac-loai-dependency-scope-trong-maven

<https://stackjava.com/maven/vi-du-cac-loai-dependency-scope-trong-maven.html>

## Cách convert mappstruct field này qua field khác

\*<https://stackoverflow.com/questions/48521903/map-custom-method-mapper-to-mapstruct>

<https://mapstruct.org/documentation/stable/reference/html/#mapping-method-resolution>

## Đọc ghi file ảnh trong java

<https://www.tutorialspoint.com/java_dip/java_buffered_image.htm>

## ResultMapping trong java( nó là jpa đấy nhé)

<https://stackoverflow.com/questions/25179180/jpa-joining-two-tables-in-non-entity-class/25184489#25184489>

Trong jpa thì bạn nên Select ntn nếu phải join thì lên join băng map hoặc new dto(b.x, b.a)

<https://stackoverflow.com/questions/36328063/how-to-return-a-custom-object-from-a-spring-data-jpa-group-by-query>

## ServletRequestAttribute và RequestContextHolder

RequestAttributes attrs = RequestContextHolder.*getRequestAttributes*();

Assert.*state*(attrs instanceof ServletRequestAttributes, "No current ServletRequestAttributes");  
return ((ServletRequestAttributes)attrs).getRequest();

**Có thể tóm tắt như sau:**

1. ServletRequestAttribute là 1 class chứa cả request(HttpServletRequest)

* Và reponse (HttpServletReponse)
* Hàm khởi tạo nó là 1 HttpServletRequest ( tức là đưa request đầu vào và nó phân tích cho bạn thôi)

1. RequestContextHolder: là 1 lớp context do spring tạo ra có thể lưu trữ các attr và convert nó thành 1 ServletRequestAttribute

* ((ServletRequestAttributes) RequestContextHolder.currentRequestAttributes()).getRequest()

**Các lớp hỗ trợ liên quan:**

* ServletUriComponentsBuilder (Cái này giúp lấy ra url của reponser cho request này
* UriComponentsBuilder

## Sự khác biệt giữa [RequestContextHolder , currentRequestAttributes() and getRequestAttributes()?](https://stackoverflow.com/questions/47586707/what-is-the-difference-between-these-methods-of-requestcontextholder-currentre)

<https://stackoverflow.com/questions/47586707/what-is-the-difference-between-these-methods-of-requestcontextholder-currentre>

## Cấu hình Swagger cho microService

<https://piotrminkowski.com/2017/04/14/microservices-api-documentation-with-swagger2/>

## Khái niệm Interceptor trong spring boot

<https://hocspringboot.net/2021/04/17/spring-boot-interceptor/>

<https://openplanning.net/11689/spring-boot-interceptor>

## LOGBACK

<https://howtodoinjava.com/logback/rollingfileappender/>

<https://viblo.asia/p/spring-spring-logback-config-GrLZDBgJ5k0>

Đây là lý do mà log méo thể ghi ra nhiều folder

<https://stackoverflow.com/questions/22188936/timebasedrollingpolicy-not-rolling-unless-there-are-new-logs>

## 15 Spring AUTO CONFIG

<https://www.baeldung.com/spring-boot-custom-auto-configuration>

[**https://docs.spring.io/spring-boot/docs/2.0.0.M3/reference/html/boot-features-developing-auto-configuration.html**](https://docs.spring.io/spring-boot/docs/2.0.0.M3/reference/html/boot-features-developing-auto-configuration.html)

## 16 Cần hiểu sâu hơn về Servlet Nó có thể lấy cả ip host

<https://stackoverflow.com/questions/20969513/java-httprequest-getremoteaddr-allways-return-127-0-0-1>

<https://stackoverflow.com/questions/12491773/why-does-request-getremoteaddr-equals127-0-0-1-when-accessing-from-a-remot>

## 17 [Sự khác nhau giữa Rest-template và FeignClient trong Spring Boot](https://daynhauhoc.com/t/su-khac-nhau-giua-rest-template-va-feignclient-trong-spring-boot/117430)

<https://daynhauhoc.com/t/su-khac-nhau-giua-rest-template-va-feignclient-trong-spring-boot/117430>

## 18 DomParing\_Java Soap

<https://stackoverflow.com/questions/42279563/parsing-a-soap-message-with-dom-library-in-java>

## 19 Ghi file Java

<https://gpcoder.com/3046-huong-dan-su-dung-luong-vao-ra-nhi-phan-trong-java/>

## 20 Regex Nâng Cao

<https://viblo.asia/p/regular-expressions-regex-khong-he-kho-nhu-nhung-gi-ban-thay-ii-L4x5xg3YlBM>

21.Lý thuyết oauth2

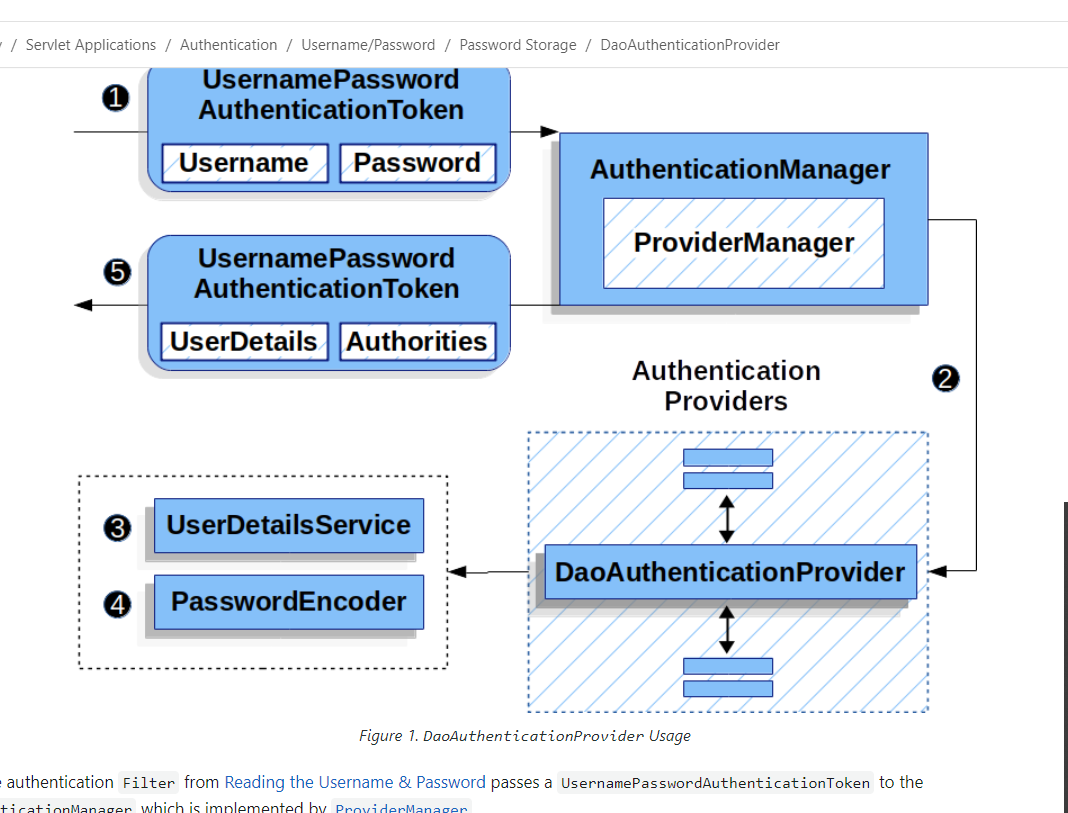
// authorization cái sinh ra token

<https://viblo.asia/p/cau-hinh-security-voi-oauth2-trong-spring-boot-4P856Nja5Y3>

## 21. Bản chất Spring Security và sự khác biệt Bảo mật web, và bảo mật phương thức Bean

<https://stackoverflow.com/questions/42316643/spring-security-configureauthenticationmanagerbuilder-auth-vs-authentication>

<https://docs.spring.io/spring-security/reference/servlet/authentication/passwords/dao-authentication-provider.html>



Chỉ cần inject 1 Cái AuthenticationManagerBuilder hoặc DaoAuthenticationProvider ,

Cái DaoAuthenticationProvider là kế thừa từ AuthenticationManagerBuilder