Spring boot

# Chap 1: rest api

## REST API

* Tài liệu

<https://www.programcreek.com/java-api-examples/?class=javax.servlet.http.HttpSession&method=removeAttribute>

<https://www.geeksforgeeks.org/arraylist-sublist-method-in-java-with-examples/>

* Spring boot:

<https://loda.me/java/>

<https://loda.me/spring-boot>

<https://kipalog.com/posts/Ta-i-sao-ne-n-ho-c-Java-va--Spring>

<https://kipalog.com/posts/Spring-Hu-o--ng-da--n-la--p-tri-nh-Spring-ca-n-ba-n-cho-ngu-o--i-mo--i>

<https://o7planning.org/vi/10127/huong-dan-lap-trinh-spring-cho-nguoi-moi-bat-dau#a7981>

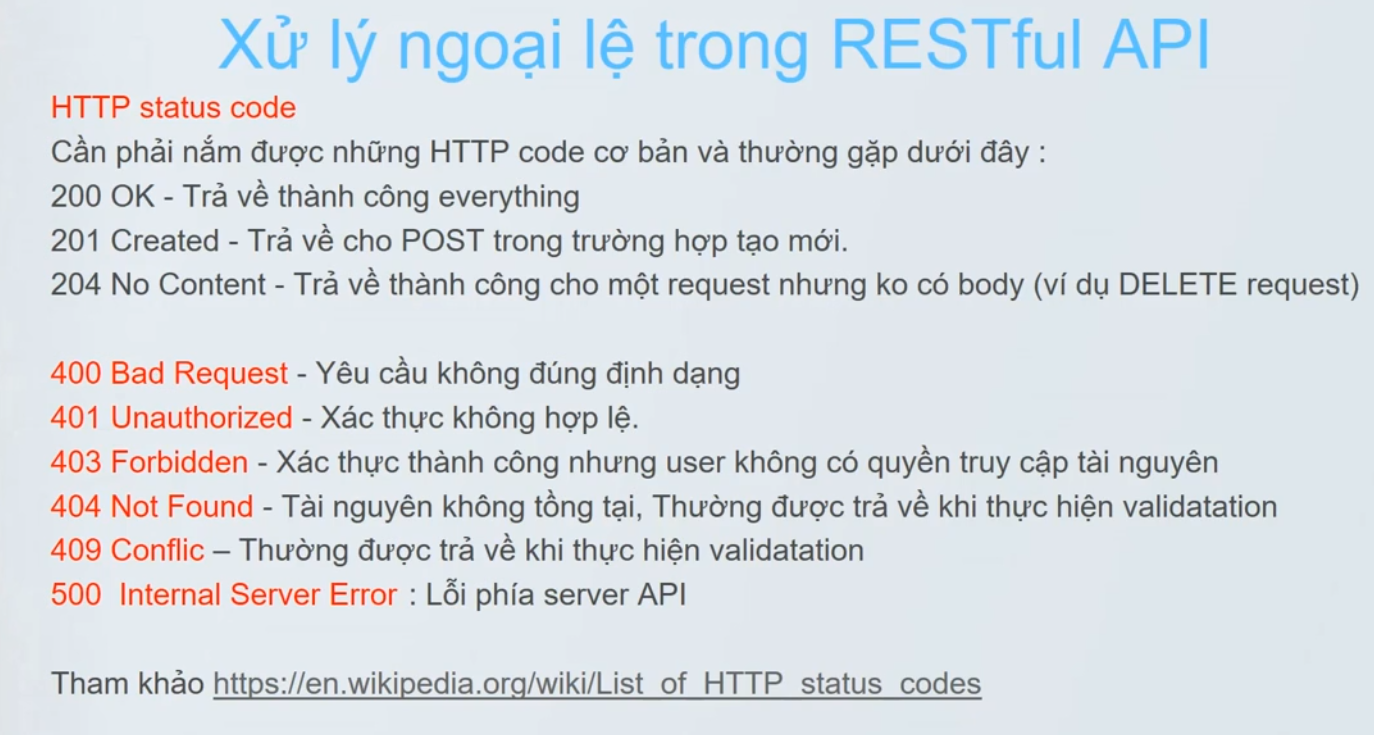
* Lombok

<https://stackjava.com/library/lombok-la-gi-sinh-code-tu-dong-tren-eclipse-intellij.html>

* Git

<https://github.com/loda-kun/spring-boot-learning>

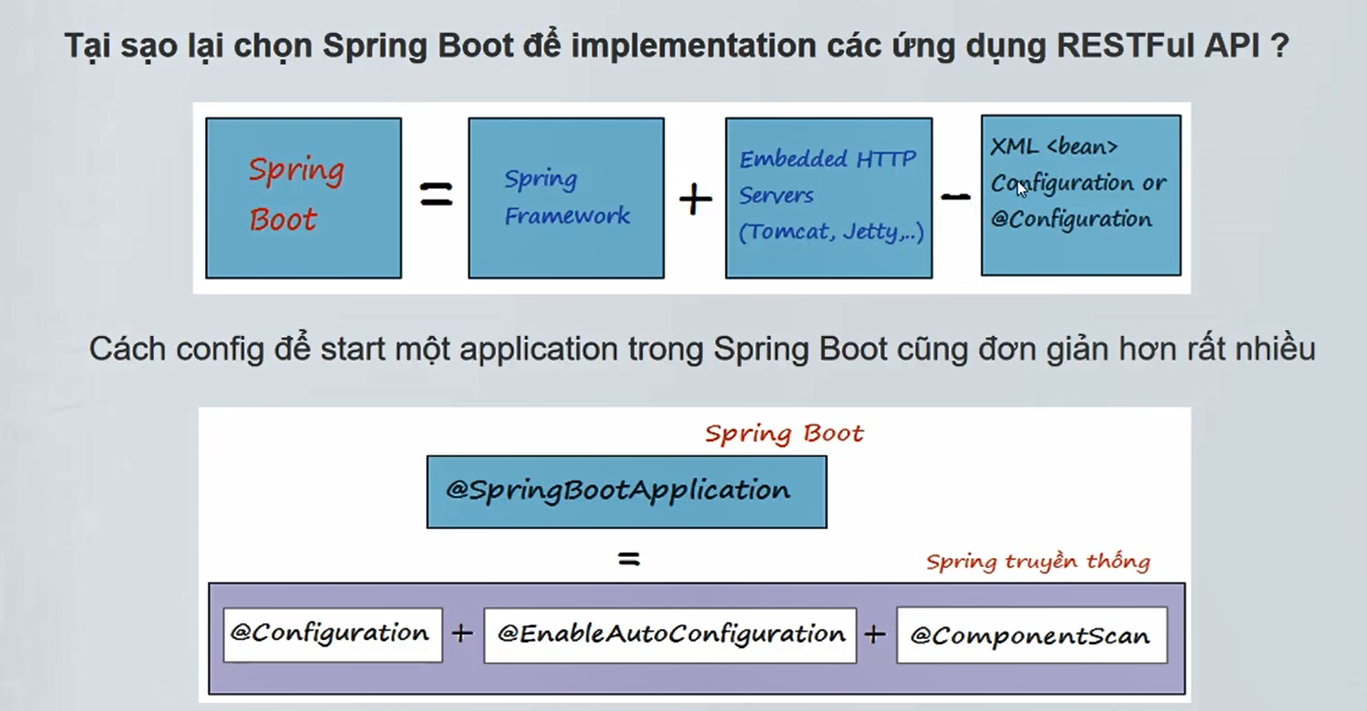
* Các ứng dụng được thiết kế để tái SD, dễ dàng chia sẻ và phát triển độc lập => chia UD lớn thành các UD nhỏ hơn
* Khái niệm
* Các nguyên tắc cơ bản thiết kế REST API
* **Web service** là 1 dịch vụ web, giúp client-server giao tiếp với nhau thông qua giao thức https/http
* Có 2 loại web service: SOAP & RESTFULL
* Web service giúp các ứng dụng giao tiếp với nhau không phụ thuộc vào nền tảng và ngôn ngữ
* **SOAP service (Simple Object Access Protocol)** là một giao thức dựa trên xml để truy cập các webservice
* Ưu bảo mật tốt, nhược chậm vì xml phải dk phân tích cú pháp
* **RESTful** **web service** được viết dựa trên kiến trúc REST chứ k phải protocol
* Restful web service nhanh chiếm ít bang thông và tài nguyên(có thể trả về nhiều loại dl như JSON)
* Nguyên tắc khi thiết kế restful api:
* Self-documenting: nhìn vào biết làm gì
* Flexible
* Thống nhất cấu trúc và cách đặt tên
* Clear err msg
* Không sử dụng danh từ số nhiều để đặt tên methods : vd /tickets => OK
* Style ngày tháng năm phải thống nhất
* Nguyên tắc thiết kế versioning : api/v1/user
* <https://swagger.io/> : tạo tài liệu api
* Bảo mật trong restful API: Restful API là stateless là nó không lưu trữ trạng thái giữa các lần client request lên server
* Do vậy cần xây dựng cơ chế authentication cho các public API nếu cần
* Ngày nay mọi người hay sử dụng web app hay mobile app nên không thẻ sửa dụng cookie hay session => SD phuong thức xác thực bằng token
* Có thể sử dụng JSON WEB TOKEN hay OAuth 2(gg, fb) để implement cho các Authentication cho các API
* **Xử lý ngoại lệ:**
* Sử dụng mã lỗi 4xx cho Http và 5xx cho internal error
* Trong một số TH http code là không đủ, chúng ta cần thiết kế để API trả về cho tiết lỗi của Application level error codes(gồm err code và msg)





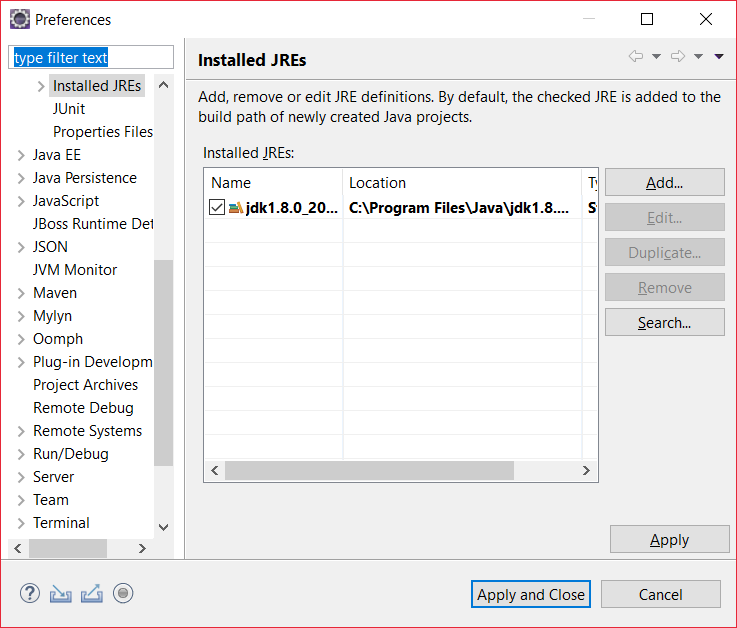
## Cấu hình

* Cài jdk 8 và cấu hình
* Lý do chọn spring boot: giúp phát triển ứng dụng nhanh mà ít cấu hình hơn spring mvc.
* Dễ tích hợp với hệ sinh thái của spring như jdbc, ORM, spring data, spring security
* Build ứng dụng như 1 standalone application, có thể chạy ứng dụng ngay mà không cần public lên server vì nó SD embedded server tức là k cần đóng gói ra file war
* Có thể dùng lệnh java -jar để chạy

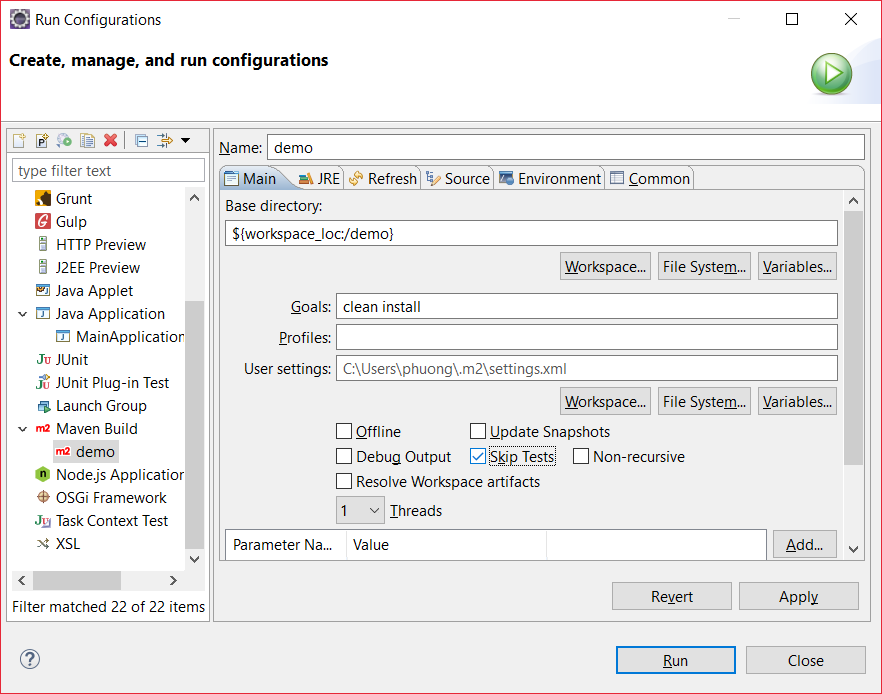


## Tạo project

* Tạo <https://start.spring.io/> (chọn packaging là jar và dependencies là web, thymeleaf)
* Sau khi import chuột phải chọn Maven/Update project để tải các package về
* Run as java application để test ok
* Lưu ý: vào java/ Install JRE chọn jdk và nhớ xóa jre nếu có để run được
* Vào properties sửa lại install JRE luôn, maven clean -> install
* Run as java application



* Restful thường chia làm 3 layer:
* Rest controller
* Service
* DAO
* Nếu có lỗi cùng port gõ cmd ở trong ổ C để xem list port: C:\>netstat -a -o -n
* Để stop port gõ: taskkill /F /PID 11340
* Tự cấu hình để clean install ở run configuration:



* S
* S