Mục Lục

[**1.** **Giới thiệu Ajax :** 1](#_Toc510835782)

[**2.** **Thực hành :** 1](#_Toc510835783)

[**3.** **Gọi Ajax Có Tham Số - dùng Method POST** 2](#_Toc510835784)

[**4.** **Gọi Ajax trả về 1 text :** 3](#_Toc510835785)

[**5.** **ObjectMapper : Lớp chuyển đối đối tượng java => json** 4](#_Toc510835786)

[**6.** **Phân Trang Ajax :** 8](#_Toc510835787)

1. **Giới thiệu Ajax :**

\_ Asynchronous Javascript & XML : giao tiếp bất đồng bộ với server.

\_ Dữ liệu nhận được : Text ( chuỗi, string ), HTML, XML, JSON.

\_ Ích lợi : không tải lại trang ( nhanh, mượt – trải nghiệm người dùng tốt hơn ).

\_ Tác hại : SEO

\_ Nhiều framework hỗ trợ : Angular, jquery.

\_ Nên sử dụng ajax khi nào :

+ Kiểm tra dữ liệu nhập : validation input form.

+ Tính năng comment.

+ Bộ lọc, sắp xếp mang lại trải nghiệm người dùng. VD : Excel.

+ Tính năng bình chọn.

1. **Thực hành :**

\_ B1 : Tạo AjaxController trả về trang index.jsp.

\_ B2 : Tại trang index.jsp bổ sung :

**<base href=”${pageContext.servletContext.contextPath}/” />**

Nhúng thư viện jquery => log kiểm tra.

Tạo button => khi click vào gọi ajax

\_ B3 : gọi Ajax :

Cú pháp : **$.ajax({ đối tượng });**

Đối tượng có các giá trị :

**+ url** **:** **‘ajax/hello.html’** địa chỉ mà website sẽ tương tác – action sẽ gọi.

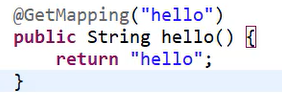
**+ success : function(response){ } :** hàm xử lý kết quả trả về từ phía server.

**+ data : { id : 1, name : ‘nghiepuit’ }** tham số sẽ gửi lên cho server. Trên server có thể dùng 4 phương pháp để nhận dữ liệu lại.

**+ type : ‘POST’** phương thức gọi ajax. Mặc đình là GET. Server viết gì dùng đó.

**+ dataType : ‘json’** kiểu dữ liệu nhận từ phía server.Mặc định là text.

\_ B4 : Xử lý action ở phía server :

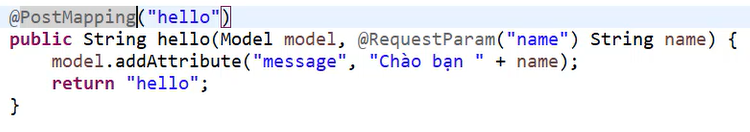


* Kết quả trả về từ server sẽ là 1 trang HTML.

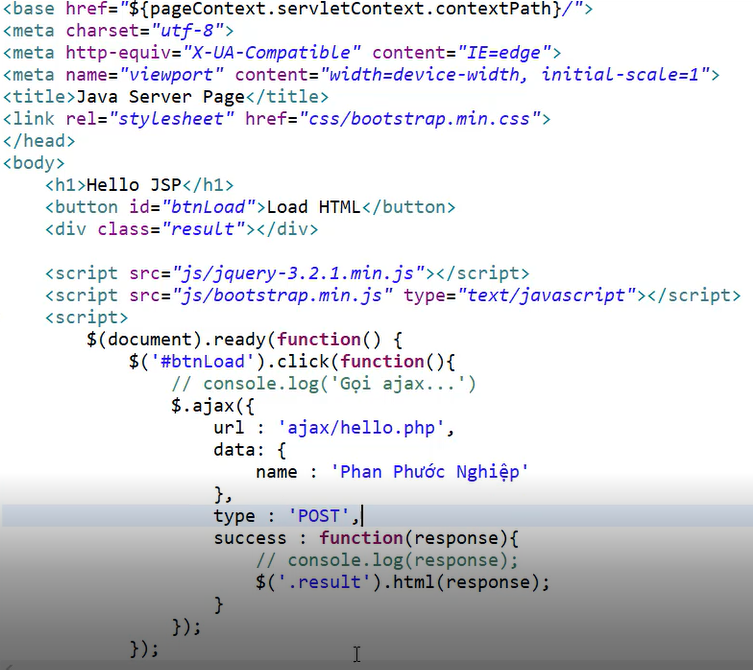
**Nâng Cao :** Làm chức năng loading. Start khi load và dừng loading khi **success** được gọi.

1. **Gọi Ajax Có Tham Số - dùng Method POST**

B1 : Server nhận bằng @RequestParam(“name”) String name : sử dụng **POST**

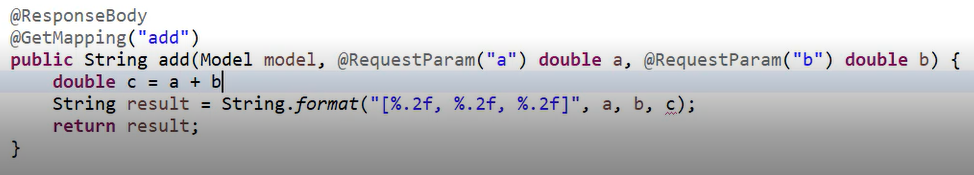


B2 : Client nhận :



1. **Gọi Ajax trả về 1 text :**

\_ Sử dụng annotation @ResponseBody để trả về 1 text với action method.

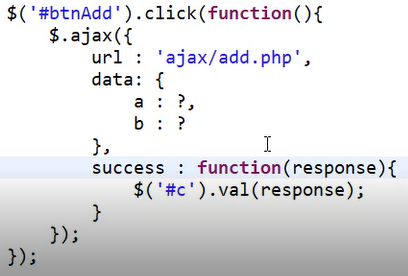


Gõ trực tiếp : /add.html?a=1&b=2 để xem kết quả.

\_ Tạo form gọi ajax :



\_ Xử lý ajax : truyền 2 tham số cho server nhận



\_ Mong muốn kết quả nhận được từ server là JSON. Ajax Engine tự động convert sang JSON cho mình. Kết quả nhận được là 1 array do ta khai báo => ta lấy phần tử đúng vị trí. Nên log ra kiểm tra. ( Lúc nãy ta trả về **[ a, b, c ]** => đây là mảng 3 pt)

1. **ObjectMapper : Lớp chuyển đối đối tượng java => json**

**\*** Không cần khai báo dataType JSON bên phía client javascript.

**\_** Map => JSON OBJECT

Ex : { “name” : “nghiepuit”, “age” : 1 }

\_ Bean => JSON OBJECT

Ex : { “name” : “nghiepuit”, “age” : 1 }

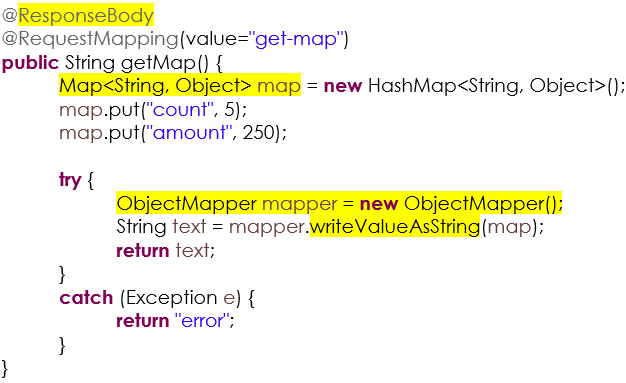
\_ List<Object> => JSON ARRAY

Ex :

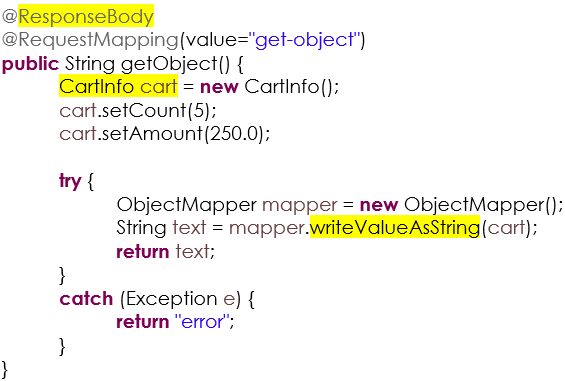
[{ “name” : “nghiepuit”, “age” : 18 }, { “name” : “a”, “age” : 1 }]

\_ Json là các cặp key ( tên thuộc tính ) – value ( giá trị thuộc tính )

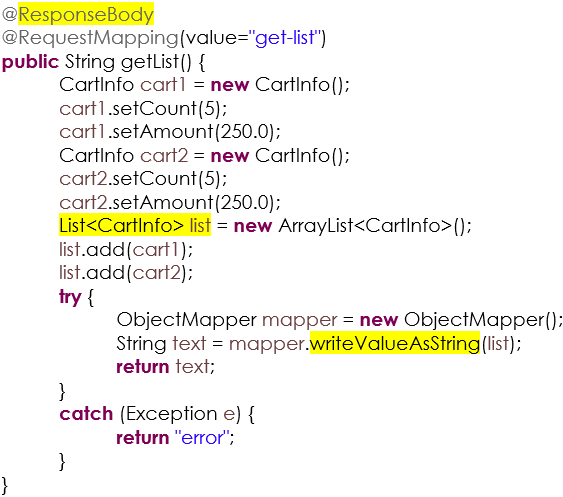
\_ Ví dụ **Map => JSON**



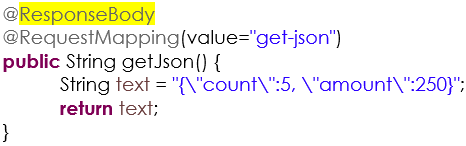
\_ Vì dụ **Bean => JSON**



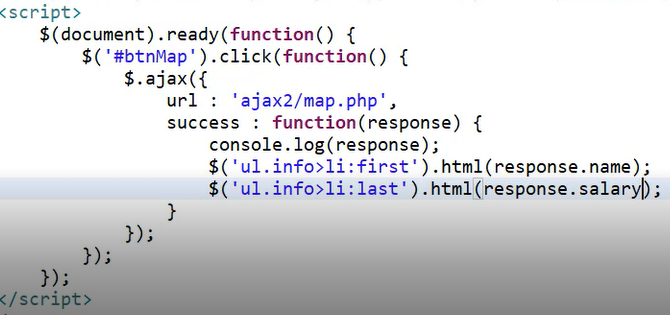
\_ Ví dụ **List Object => JSON ARRAY**



\_ Ví dụ cách thủ công :

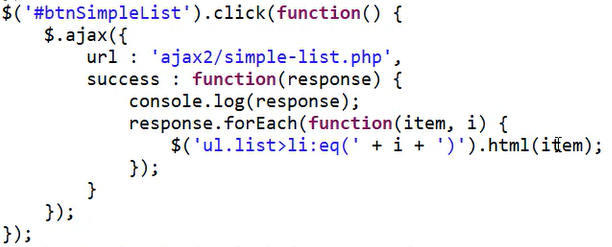


\_ Nhận dữ liệu và xử lý :



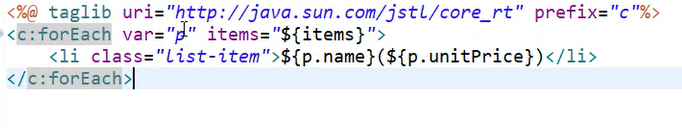
Bean tương tự.

Json Array :



1. **Phân Trang Ajax :**

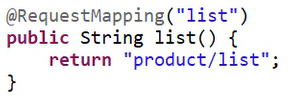
**\_** Tạo template danh sách sản phẩm trên 1 trang ( page.jsp )



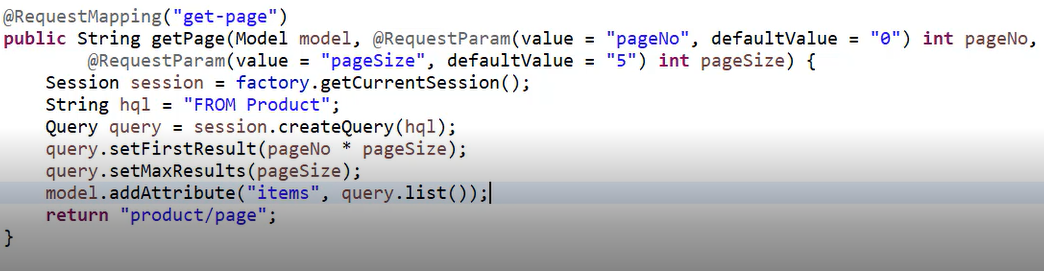
\_ Trang danh sách tạo 1 vùng để đổ nội dung vào. ( trang list.jsp )

\_ Tại controller :

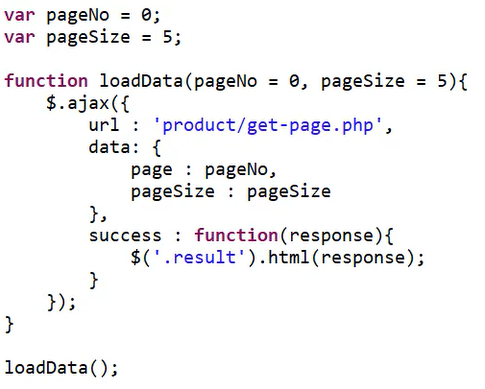
+ Trang hiển thị cho người dùng :



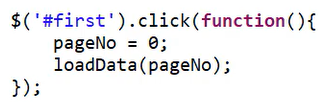
+ Ajax sẽ gọi action này :

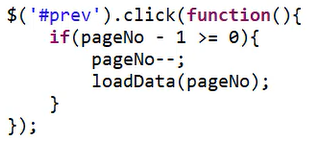


\_ Xử lý ajax tại **list.jsp** :

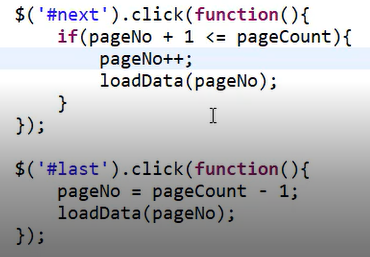


\_ Xử lý nút : first, prev, next, last

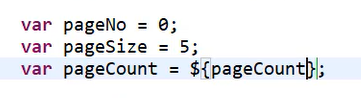




\_ Đối với next và last cần tổng số trang để biết được trang cuối hay chưa ?



* Tính pageCount



Truyền biến pageCount từ controller thông qua model :

