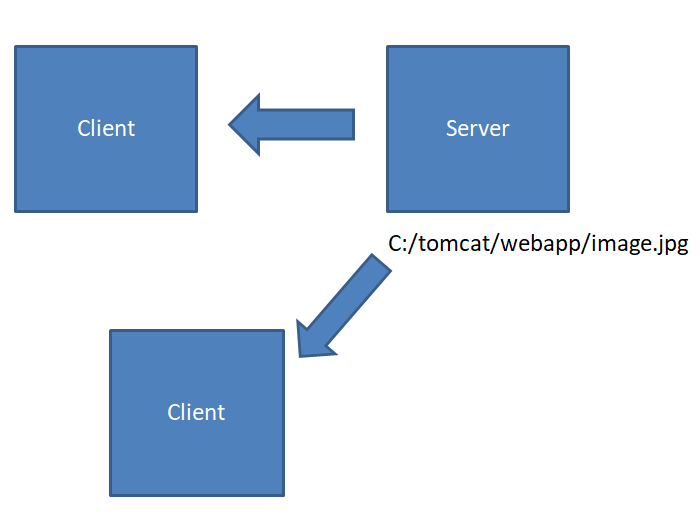
# Kiến thức cơ bản

Trong tài liệu này sẽ đề cập tới việc Upload File trong Spring MVC với Spring Boot Projet

Như các bạn đã biết việc upload file là cơ bản trong việc chia sẻ tài nguyên các máy tính trong mạng

## Tại sao Upload File



Các client sẽ đọc được ảnh từ server

Người quản trị là người đăng nhập máy tính của mình và quản lý các ảnh trong trang web. Máy tính này là client

Gỉa sử nếu người ấy muốn them một ảnh lên trang web, tốt nhất là không phải lên server để thêm ảnh. Điều đó cũng tương tự vài chuyện chỉnh sửa của thành viên của trang web nếu họ cần thay đổi avatar.

Ví dụ như Facebook các bạn chắc hẳn đã quá quen với thay đổi ảnh đại diên và hình nền của mình!

Vậy thì sao phải upload file, chắc hẳn mọi người đã biết, nhưng hiểu đơn giản như sau

Bản chất upload file là chia sẻ tài nguyên client với server và các client khác. Các client thì sẽ đọc được tài nguyên của mình thông qua server.

## Các cách upload file

Upload file thì khá nhiều cách, với Web thì liệt kê vài cách sau:

* Sử dụng web service và add file vào Body của request
* Sử dụng Form để put file lên

Nói chung là sử dụng phương thức mạng và đưa file vào để truyền qua

## Nhắc lại về Form

Form có thể hiểu là cách truyền thông tin tới một URL

|  |
| --- |
| <form action=*"*${pageContext.request.contextPath}*/category/create"* method=*"POST"* enctype=*"multipart/form-data"*>  <p> Name  <input type=*"text"* name=*"name"*>  </p>  <p> Image  <input type=*"file"* name=*"file"*>  </p>  <input type=*"submit"* value =*"Submit"*>  </form> |

* action là URL mà dữ liệu cần gửi tới
* method phương thức gửi dữ liệu. Phương thức GET là mặc định của HTTP nên với Form cũng không ngoại lệ như vậy
* enctype dạng dữ liệu sẽ gửi đi. Với mô tả enctype chỉ sử dụng được với method dạng POST.

# Upload File trong Spring MVC - Boot

## Tạo View

|  |
| --- |
| <form action=*"*${pageContext.request.contextPath}*/category/create"* method=*"POST"* enctype=*"multipart/form-data"*>  <p> Name  <input type=*"text"* name=*"name"*>  </p>  <p> Image  <input type=*"file"* name=*"file"*>  </p>  <input type=*"submit"* value =*"Submit"*>  </form> |

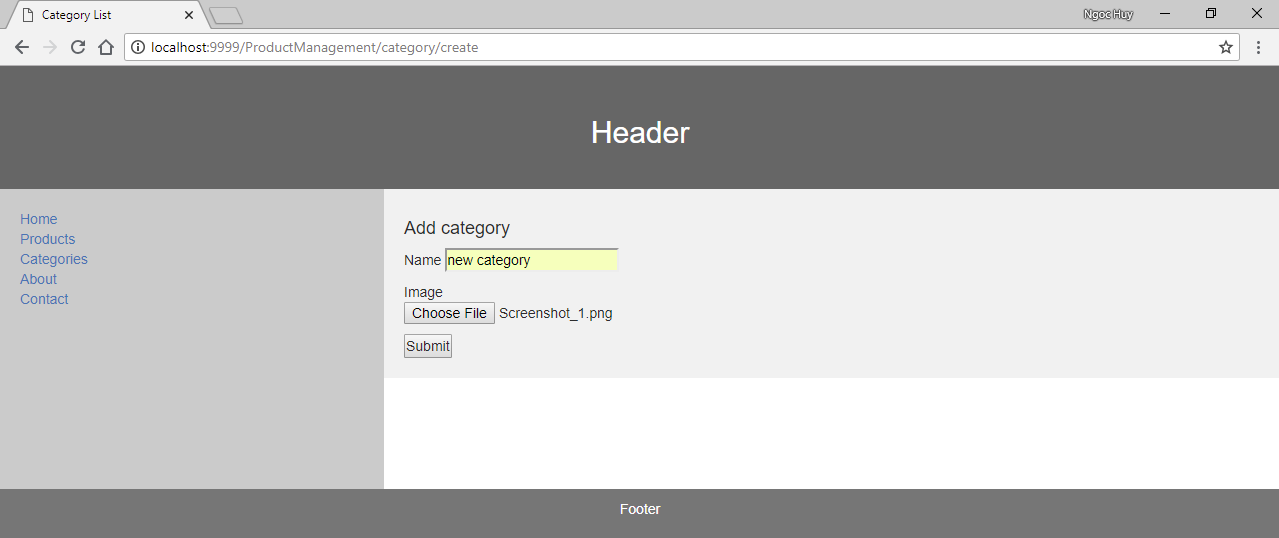
## Tạo Controller

### Method GET show View

Để người dùng thấy view để chọn file upload, thì ta xây dựng một method tương ứng này

|  |
| --- |
| @RequestMapping(value = "category/create", method = RequestMethod.***GET***)  **public** ModelAndView index() {  ModelAndView m = **new** ModelAndView("createCategoryPage");  **return** m;  } |

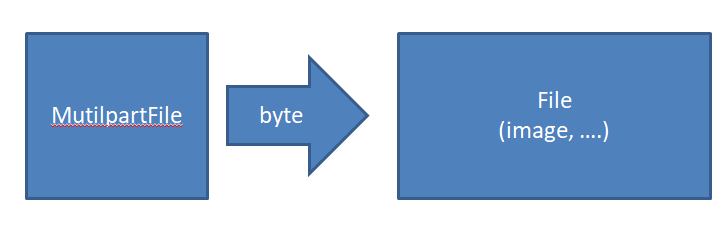
Kiểm tra view



Form upload file

## Method POST upload file

Như các bạn biết form sẽ POST dữ liệu lên với dạng kiểu *multipart* giờ chúng ta phải sử dụng đối tượng tương ứng MutipartFile của Spring để nhận file.



Cách thức xử lý chuyển dạng byte thành file

|  |
| --- |
| @Value("${upload.path}")  **private** String pathFolder;  @RequestMapping(value = "/category/create", method = RequestMethod.***POST***)  **public** ModelAndView createCategory(@RequestParam("name") String name, @RequestParam("file") MultipartFile file) {  **try** {  **if** (!file.isEmpty()) {  Category category = **new** Category();  category.setName(name);    **byte**[] bytes = file.getBytes();  Path path = Paths.*get*(pathFolder + file.getOriginalFilename());  Files.*write*(path, bytes);    category.setImages("images/" + file.getOriginalFilename());    categoryService.createCategory(category);  **return** **new** ModelAndView("redirect:/category");  }  } **catch** (Exception e) {  e.printStackTrace();  }  **return** **new** ModelAndView("redirect:/");  } |

## Cấu hình application.properties

|  |
| --- |
| upload.path=src/main/resources/static/images/ |

## Giải thích method Post

1. Đường dẫn cho việc upload

|  |
| --- |
| @Value("${upload.path}")  **private** String pathFolder; |

Đọc giá trị biến upload.path trong application.properties. ở đây

pathFolder = src/main/resources/static/images/ => Chúng ta đưa file tớ thư mục ảnh trong static của Spring Boot

1. Nhận file từ Form

Form truyền dữ liệu qua method POST dạng MutipartFile (mã hóa file thành phần nhỏ) truyền đi. Vậy controller cũng phải nhận tương ứng vậy

|  |
| --- |
| @RequestMapping(value = "/category/create", method = RequestMethod.***POST***)  **public** ModelAndView createCategory(@RequestParam("name") String name, @RequestParam("file") MultipartFile file) { |

1. Lưu file

Nhận file từ form thì phải lưu lại đâu đó trên server chứ!

|  |
| --- |
| **byte**[] bytes = file.getBytes();  Path path = Paths.*get*(pathFolder + file.getOriginalFilename());  Files.*write*(path, bytes); |

Vậy ở trên chúng ta lưu file ở đường dẫn pathFolder + file.getOriginalFilename()tương đương chúng ta lưu nó lại ở thư mục static với tên file là tên file cũ từ client gửi đi. Nếu bạn có thể lưu nó với tên khác tùy bạn ☺

1. Lưu thông tin file trong CSDL để gọi (Phụ)

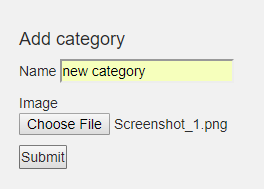
|  |
| --- |
| category.setImages("images/" + file.getOriginalFilename());  categoryService.createCategory(category); |

Các bạn biết là src/main/resources và folder static là nguồn của trang web.

Như vậy khi lưu “images/" + file.getOriginalFilename()” nó sẽ đọc từ nguồn của trang web để tìm tới thư mục images và tìm tới tên file mà các bạn lưu

## Test

1. Chọn file với view



1. Kiểm tra file

