**Controller**

**I-** **Introduction**  
**Controller** là các lớp điều khiển luồng ứng dụng, tiếp nhận yêu cầu người dùng thông qua HTTP header, sau đó chuyển tiếp nó đến các lớp phụ trách trực tiếp xử lý yêu cầu. Tùy theo cách thiết kế lớp mà chúng ta thường thấy Controller gồm:  
  
+ **Front Controller**. Là một controller xử lý tất cả các yêu cầu người dùng cho website. Front Controller có nhiệm vụ hợp nhất tất cả các xử lý yêu cầu vào một kênh yêu cầu thông qua một đối tượng.  
  
+ **Dispatcher**: Lớp điều phối hướng các điều khiển đi mức cao hơn

+ **Action**: lớp xử lý các sự kiện chính, nơi dẫn đến application flow chủ yếu (Main Event Handler)  
+ **Request**: xử lý một phần dữ liệu đầu vào ở mức GET, POST và PUT  
+ **Session**: xử lý một phần dữ liệu đầu vào ở mức SESSION   
Tùy theo dữ liệu đầu vào, Controller sẽ thực hiện các phép lọc (với dịch vụ lấy từ Model), các tính toán lựa chọn (Action Mapping) dựa trên kiến trúc và cấu hình nhằm xác định thành phần lớp chính sẽ thực hiện yêu cầu của người dùng. Hiểu một cách đơn giản, Controller là thành phần trung gian giữa View và Model. Nó nhận dữ liệu nhập vào qua View, sau đó gọi Model tương ứng rồi lấy kết quả trả về từ Model này. Tiếp theo, một View thích hợp sẽ được lựa chọn. Controller sẽ chuyển tiếp dữ liệu vào view để nó xử lý.

* **Một số hoạt động thường thấy của Controller:**  
  - Tạo form, gửi tin nhắn đến form để yêu cầu kiểm tra dữ liệu  
  - Tạo các dịch vụ liên quan đến nghiệp vụ ứng dụng, yêu cầu các lớp dịch vụ tương tác với nguồn dữ liệu để trả về hay thay đổi trạng thái dữ liệu: thực hiện các thao tác chuyển đổi dữ liệu, kiểm tra quyền truy cập trên một hoạt động cụ thể, tương tác với database, tương tác với các web services.  
  - Tạo đối tượng view, gán các nguồn dữ liệu lấy được từ đối tượng dịch vụ vào cho view.

**II- App controller**  
 **- AppController** là một lớp cha của tất cả các lớp controller trong ứng dụng. Bản thân AppController kế thừa lớp Controller trong thư viện CakePHP. AppController được định nghĩa trong **src/Controller/AppController.php**:

*namespace* App\Controller;

*use* Cake\Controller\Controller;

*class* AppController *extends* Controller

{

}  
Vì lớp AppController là lớp cha của các lớp controller trong ứng dụng, do đó mà các thuộc tính và các phương thức trong lớp AppController sẽ có thể được chia sẽ và sử dụng bởi các lớp con.  
  
**III- Pages Controllers**  
CakePHP hỗ trợ chúng ta Pages Controller (cake/libs/controller/pages\_controller.php). Trang chủ( home page ) mà bạn thấy sau khi cài đặt là được tạo ra bằng cách dùng controller này. Page controller thường được dùng để tạo ra các trang tính.  
Ví dụ: bạn tạo trang About. ( bạn tạo 1 file view /app/views/pages/about\_us.ctp, bạn truy cập nó bằng cách sử dụng đường dẫn http://example.com/pages/about\_us  
để sử dụng Pages controllers, chúng ta chỉ cần copy pages\_controllers.php từ core của CakePHP đến ứng dụng của mình.  
**IV- Các thuộc tính của controller  
1- $name**  
→ được dùng để thiết lập tên của controller   
→ thông thường thì đây là hình thức số nhiều của tên model mà controller sử dụng  
Ví dụ:  
# $name controller attribute usage example  
  
class RecipesController extends AppController {  
var $name = 'Recipes';  
}  
  
**2- $components, $helpers and $uses**  
ví dụ:  
class RecipesController extends AppController {  
var $name = 'Recipes';  
  
var $uses = array('Recipe', 'User');  
var $helpers = array('Ajax');  
var $components = array('Email');  
}  
**3- $layout**  
→ sử dụng $layout để chọn layout cho chương trình /app/views/layouts( $layout sẽ lưu tên của layout trong /app/views/layouts )   
→ nếu giá trị này ko được thiết lập thì layout mặc định sẽ được sử dụng cho ứng dụng của bạn: default.ctp  
ví dụ:  
// Using $layout to define an alternate layout  
class RecipesController extends AppController {  
function quickSave() {  
$this->layout = 'ajax';  
}  
}  
**4- $params**  
→ Được sử dụng để cung cấp quyền truy cập vào các thông tin về request hiện tại:  
4.1- form : → $this->params['form'];  
4.2- admin : → $this->params['admin'];  
4.3- bare : → $this->params['bare'];  
4.4- form : → $this->params['isAjax'];  
**V- Các method của controller**  
**1- Tương tác với View**  
**1.1- set( string $var, mixed $value)**  
chuyển dữ liệu từ controller tới view.  
Ví dụ:  
$this->set('title\_for\_layout', 'This is the page title');  
**1.2- render**  
render(string $action, string $layout, string $file)   
Dùng để render các view tương ứng  
Ví dụ:  
class RecipesController extends AppController {  
function search() {  
// Render the view in /views/recipes/search.ctp  
$this->render();  
}  
}  
  
**2- Flow control**   
**2.1- redirect**  
redirect(mixed $url, integer $status, boolean $exit)   
ví dụ:  
function placeOrder() {  
  
//Logic for finalizing order goes here  
  
if($success) {  
$this->redirect(array('controller' => 'orders', 'action' => 'thanks'));  
} else {  
$this->redirect(array('controller' => 'orders', 'action' => 'confirm'));  
}  
}  
  
**2.2- flash**

flash(string $message, string $url, integer $pause, string $layout)   
được dùng để redirect đến 1 trang. Phương pháp này khác với phương pháp redirect() ở chỗ là nó hiển thị 1 message trước khi chuyển tới 1 url khác  
  
**3- Callbacks**  
**3.1- beforeFilter()**Được thực thi trước mỗi action của controller  
**3.2- beforeRender()**  
Được thực thi sau mỗi action của controller( nhưng trước khi view được gọi )  
**3.3- afterFilter()**  
được gọi sau mỗi action và sau rendering được hoàn tất  
**3.4- Các callback trong scaffold**  
\_beforeScaffold($method)  
\_afterScaffoldSave($method)  
afterScaffoldSaveError($method)  
\_scaffoldError($method)  
**4- Một số phương thức khác**  
- constructClasses(), referer(), disableCache(), postConditions(), paginate(), requestAction(), loadModel().