**1, Ví dụ.**

-Ví dụ như module Query Builder của Laravel chúng ta co thể gọi được như sau:

DB::table('users')->where('name', 'John')->first();

-Hay cả với Query Builder Class của Codeigniter họ cũng có có thể gọi được như vậy.

$this->db->select('\*')->from('blogs')->join('comments', 'comments.id = blogs.id');

**2, Xây dựng call multiple methods.**

-Thực ra để làm được như các ví dụ trên thì cũng hết sức đơn giản, các bạn chỉ cần **return lại chính đối tượng đó** ($this) sau mỗi methods được.

**VD**:

<?php

class ConNguoi

{

private $name;

public function setName($name = 'Con Nguoi')

{

$this->name = $name;

//trả về đối tượng hiện tại

return $this;

}

public function getName()

{

return $this->name;

}

}

//Khởi tạo class

$connguoi = new ConNguoi();

//gọi nhiều phương thức trên một dòng

echo $connguoi->setName('Vũ Thanh Tài')->getName();

//Kết quả: Vũ Thanh Tài

- Bây giờ chúng ta sẽ thử bỏ this đi xem nó có chạy được không nhé.

<?php

class ConNguoi

{

private $name;

public function setName($name = 'Con Nguoi')

{

$this->name = $name;

}

public function getName()

{

return $this->name;

}

}

//Khởi tạo class

$connguoi = new ConNguoi();

//gọi nhiều phương thức trên một dòng

echo $connguoi->setName('Vũ Thanh Tài')->getName();

//Kết quả: Fatal error: Call to a member function getName() on null

-Như các bạn cũng đã thấy đó. Chương trình đã báo lỗi.

-Vậy tiếp tục chúng ta để return $this và thử gọi phương thức setName() 2 lần xem có được không nhé!

<?php

class ConNguoi

{

private $name;

public function setName($name = 'Con Nguoi')

{

$this->name = $name;

//trả về đối tượng hiện tại

return $this;

}

public function getName()

{

return $this->name;

}

}

//Khởi tạo class

$connguoi = new ConNguoi();

//gọi nhiều phương thức trên một dòng

echo $connguoi->setName('Vũ Thanh Tài')->setName('Vũ Văn A')->getName();

//Kết quả: Vũ Văn A

-oh, Chương trình vẫn chạy và nó nhận setName() cuối cùng. Như thế chúng ta có thể đưa ra được cá kết luận rồi.

**Kết Luận**

- Khi chương trình sử dụng return $this  thì cũng đồng nghĩa với phương thức đó **trả về chính đối tượng mà nó đang được thực thi** và lúc này **các xử lý của phương thức đó cũng được add vào đối tượng**. Để chứng minh chúng ta sẽ cùng var\_dump() dữ liệu trước và sau khi thực thi hàm xem nhé.

<?php

class ConNguoi

{

private $name;

public function setName($name = 'Con Nguoi')

{

$this->name = $name;

//trả về đối tượng hiện tại

return $this;

}

public function getName()

{

return $this->name;

}

}

$connguoi = new ConNguoi();

//var\_dump đối tượng sau khi khởi tạo

var\_dump($connguoi);

/\*

//kết quả:

object(ConNguoi)#1 (1) {

["name":"ConNguoi":private]=>

NULL

}

\*/

var\_dump($connguoi->setName('Vũ Thanh Tài'));

/\*

Kết quả:

object(ConNguoi)#1 (1) {

["name":"ConNguoi":private]=>

string(14) "Vũ Thanh Tài"

}

\*/

-Kết quả như ý muốn đúng không các bạn?, Và vì nó trả về mọt object như thế nên các bạn có thể gọi luôn được đến các phương thức tiếp theo trong đối tượng.