**1. Khai báo tên thuộc tính để nhận dạng cấp độ truy cập**

Cái tiêu đề nghe hơi lại nhưng nó thực sự rất hữu ích trong quá trình làm team cho các bạn đó. Nếu trong một team có những quy ước đặt tên [biến](https://freetuts.net/khai-bao-bien-va-hang-so-trong-php-2.html), tên hàm riêng thì nhìn vào dòng code tuy nhiều người nhưng tưởng chừng như là chỉ có một người làm duy nhất. Bởi vậy mình đặt ra tiêu đều này để hướng dẫn các bạn cách đặt tên biến, tên hàm trong một lớp. Tuy nhiên không phải người nào cũng áp dụng như vậy nên các bạn đừng chém nhé, **các bạn đọc nhưng có quyền không làm theo**.

**Cách chọn từ đặt cho tên biến và tên hàm**

Thông thường tên biến (thuộc tính) phải là một danh từ hoặc tình từ, còn tên của hàm (phương thức) bắt đầu bằng một động từ.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | class A  {      public $username;      public $password;        public function getUsername()      {        }        public function getPassword()      {        }        public function checkLogin()      {        }  } |

**Khai báo tên biến dạng private**

Thông thường nếu biến ở dạng private thì tên biến các bạn nên đặt hai dấu gạch dưới rồi mới đến tên biến.

**Ví dụ:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | class A  {      private $\_\_private;      private function \_\_func\_private()      {      }  } |

**Khai báo tên biến dạng protected**

Thông thường nếu ở dạng protected thì các bạn nên đặt một dấu gạch dưới rồi mới tới tên biến

**Ví dụ:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | // Lớp A  class A  {      protected $\_protected;      protected function \_func\_protected()      {      }  } |

**Khai báo tên biến dạng public**

Nếu ở dạng này bạn chỉ cần khai báo tên bình thường

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | class A  {      public  $public;      public function func\_public()      {      }  } |

**2. Hàm Set và hàm Get các thuộc tính**

Các thuộc tính trong đối tượng thường có hai thao tác là gán dữ liệu và lấy dữ liệu. Có những thuộc tính ta muốn khi gán giá trị hoặc khi lấy giá trị thì sẽ có một đoạn code tác động vào. Ví dụ như tôi muốn khi thiết lập mật khẩu cho người dùng thì mật khẩu sẽ được tự động thêm một số ký tự đặc biệt và mã hóa md5 để an toàn và bảo mật hơn, như vậy mỗi lần gán giá trị cho thuộc tính mật khẩu tôi phải thêm một đoạn như thế này:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | class A  {      public $username;      public $password;  }    $a = new A();  $a->password = md5('thehalfheart' . 'ky\_tu\_muon\_them'); |

Giả sử trong quá trình lập trình tôi có sử dụng phép gán này 100 lần thì đôi lúc sẽ bị nhầm lẫn ký tự thêm vào, chưa tính đến chuyện sau này tôi muốn sửa ký tự thêm vào thành một chuỗi khác hoặc sử dụng hàm mã hóa khác thì tôi phải vào 100 dòng code đó để sửa.

Giả sử tôi muốn khi lấy giá trị của thuộc tính username thì kết quả trả về sẽ có thêm một đoạn là "*Xin Chào + Username*", như vậy mỗi lần tôi lấy giá trị username phải bổ sung chữ xin chào vào. Có 100 dòng code thì phải thực hiện thao tác này 100 lần, điều này thật là phiền toái.

Qua những lý do đó tôi nghĩ có giải pháp là ta tạo những hàm SET và GET dùng cho việc gán và lấy dữ liệu cho các thuộc tính username và password để sau này nếu thay đổi thì chỉ cần vào hai hàm này sửa là được.

Quy tắc đặt tên cho Get và Set như sau: setThuoctinh(), getThuoctinh().

**Ví dụ:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | class A  {      private $\_\_username;      private $\_\_password;        function getUsername()      {          return 'Xin chào ' . $this->\_\_username;      }        function setUsername($username)      {          $this->\_\_username = $username;      }        function getPassword()      {          return $this->\_\_password;      }        function setPassword($password)      {          $this->\_\_password = md5($password.'ky\_tu\_muon\_them');      }  }    // Sử dụng  $a = new A();    $a->setUsername('TheHalfheart');    echo $a->getUsername();    $a->setPassword('matkhau');    echo $a->getPassword(); |

**3. Khi nào sử dụng cấp độ truy cập private**

Thông thường để an toàn dữ liệu các thuộc tính đều ở dạng private, nhưng điều này rất phiền vì ta phải tạo thêm các hàm SET và GET nên các lập trình viên cũng ít khi sử dụng private. Tuy nhiên có những trường hợp sau ta bắt buộc phải dùng ở dạng private để an toàn cho đối tượng.

* Những thuộc tính có tính biến đổi dữ liệu khi nhập và lấy dữ liệu như hai ví dụ về Username và Password ở trên.
* Những phương thức chỉ dùng trong nội bộ trong lớp đó, không có sử dụng bên ngoài lớp.

Đó là 2 đặc điểm cơ bản để bạn chọn độ truy cập là private.

**4. Khi nào sử dụng cấp độ truy cập protected**

Protected thường được dùng khi bạn biết chắc là có lớp khác sẽ kế thừa lớp này và những phương thức, thuộc tính đó chỉ được dùng trong lớp kế thừa nó.

Giả sử bạn khai báo lớp Động Vật, trong đó có hàm lưu dữ liệu động vật vào database, hàm này dùng chung cho tất cả các lớp con kế thừa lớp Động Vật. Và để bảo mật nên tôi không muốn ở bên ngoài lớp có thể sử dụng được, vì vậy tôi khai báo là protected.

Đó là những ví dụ cơ bản, chứ thực tế thì cũng tùy vào từng bài toán cụ thể mà bạn lựa chọn

**5. Khi nào sử dụng cấp độ truy cập public**

Public là cấp độ thoáng nhất, nó có thể gọi ở mọi nơi từ trong nội bộ của lớp đến lớp kế thừa nó, thậm chí cả bên ngoài lớp cũng gọi được.

Những hàm được khai báo public thường được dùng để lấy dữ liệu và xuất dữ liệu ra bên ngoài, hoặc là những hàm mang tính chất là hàm thao tác cuối cùng với người coder. Ví dụ như các hàm SET và GET được để ở dạng public.