**1. Lớp trừu tượng abstract**

Nếu bạn đã từng học qua lập trình hướng đối tượng một ngôn ngữ bất kỳ khác như C++, Java thì abstract khá quen thuộc, và trong php đây cũng là một tính chất của lập trình hướng đối tượng nhưng nó không được xếp vào tính chất quan trọng, tính chất này giống như tính kế thừa theo tầm nhìn bề ngoài. Lớp Abstract sẽ định nghĩa các hàm (phương thức) mà từ đó các lớp con sẽ kế thừa nó và Overwrite lại (tính đa hình). Tất cả các phương thức của lớp abstract đều phải được khai báo là abstract và phải ở mức protected và public, không được ở mức private. Lớp Abstract có thể có [thuộc tính](https://freetuts.net/lop-thuoc-tinh-phuong-thuc-cua-doi-tuong-28.html) nhưng thuộc tính không được khai báo là abstract, và bạn không thể khởi tạo một biến của lớp Abstract được.

Để khai báo một **lớp Abstract** ta dùng cú pháp sau:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | abstract class BaseClass  {      // phương thức ở mức protected      abstract protected function hello();        // Phương thức ở mức public      abstract public function hi();  } |

Trong lớp Abstract các phương thức bạn khai báo ở dạng Abstract đều phải tuân theo cú pháp trên, tức là bạn không được định nghĩa thêm dòng lệnh nào bên trong nó. Như ví dụ dưới này là sai.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | abstract class BaseClass  {      // Phương thức này sai vì hàm hello là      // hàm abstract nên không được code bên trong nó      abstract protected function hello()      {          // dòng code          echo 1;      }    } |

Như tôi nói ở trên bạn không thể tạo một biến đối tượng mới của lớp Abstract, như ví dụ dưới này là sai:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | abstract class BaseClass  {      abstract protected function hello();  }    // Sai vì BaseClass là lớp Abstract nên không  // khởi tạo mới được  $base = new BaseClass(); |

Mức truy cập các hàm của Abstract phải ở public hoặc protected để lớp kế thừa có thể định nghĩa lại và các thuộc tính của lớp Abstract không được khai báo Abstract. Các bạn xem ví dụ dưới đây:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | abstract class BaseClass  {      // Đúng      public $name;        // Sai vì các thuộc tính không được để ở dạng abstract      abstract public $title;        // Đúng      abstract protected function hello();        // Sai vì hàm abstract không thể ở private      abstract private function hi\_there();  } |

Lớp kế thừa từ lớp Abstracth phải Rewrite lại tất cả các hàm Abstract trong lớp Abstract, nếu không sẽ bị báo sai. 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 | abstract class Person  {      protected $ten;      protected $cmnd;      protected $namsinh;        abstract public function showInfo();  }    // Lớp này sai vì chưa viết lại hàm showInfo  class CongNhan extends Person  {    }    // Lớp này đúng vì ta đã khai báo, viết lại  // đầy đủ các hàm abstract  class SinhVien extends Person  {      public function showInfo(){        }  } |

**2. Hàm và lớp final**

Lớp Final là lớp được khai báo là lớp cuối cùng, không một lớp nào có thể kế thừa nó. Tương tự như hàm Final trong Abstract hoặc trong kế thừa chỉ để gọi sử dụng, không được viết lại (Override).  Để dễ hình dung các bạn xem ví dụ sau đây và thông qua phần ghi chú tôi đã giải thích.  
**Ví dụ lớp final:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | // Lớp Filnal  final class Person  {      protected $ten;      protected $cmnd;      protected $namsinh;      public function showInfo()      {          echo 'freetuts.net';      }  }    // Hàm này sẽ bị báo lỗi vì lớp SinhVien  // đã kế thừa một lớp Final, điều này là không thể  class SinhVien extends Person {  }    // Đoạn code này đúng vì lớp Final được  // sử dụng bình thường như các lớp khác  // chỉ có điều là không được kế thừa  $person = new Person;  $person->showInfo(); |

Ví dụ hàm final:

|  |  |
| --- | --- |
| 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 | class Person  {      protected $ten;      protected $cmnd;      protected $namsinh;      final public function showInfo()      {          echo 'freetuts.net';      }  }    // Lớp này đúng vì lớp Person không phải  // là một lớp final  class SinhVien extends Person {        // Hàm này sai vì hàm showInfo      // là hàm final trong lớp Person      // nên không thể Override lại      public function showInfo(){        }        public function Go()      {          // Đoạn code này đúng vì hàm final được          // sử dụng bình thường          $this->showInfo();      }  } |