**1. Mức truy cập private**

Đây là thành phần chỉ dành riêng cho nội bộ của lớp, nghĩa là ta không thể truy xuất tới thành phần private ở lớp con hoặc ở bên ngoài lớp.

Mức truy cập private thường được sử dụng với:

* Các thuộc tính dữ liệu nhằm bảo vệ chúng, tránh sự truy cập tự do từ bên ngoài. Các thuộc tính này sẽ có những hàm SET và GET gán và lấy dữ liệu.
* Các phương thức trung gian tính toán trong nội bộ của đối tượng mà ta k muốn bên ngoài có thể can thiệp vào.

**Ví dụ không kế thừa:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | // Lớp Xe  class Xe  {      private $loaixe;        function getLoaixe()      {           return $this->loaixe;      }        function setLoaixe($loaixe)      {              $this->loaixe = $loaixe;      }  }    // Khởi tạo một lớp đối tượng xe  $xe = new Xe();    // Gán giá trị cho thuộc tính loại xe  $xe->setLoaixe('Wave S');    // Lấy giá trị thuộc tính loại xe  echo $xe->getLoaixe();  // kết quả xuất ra Wave S |

Ở ví dụ trên muốn truy xuất giá trị thuộc tính loaixe thì phải thông qua phương thức setLoaixe và getLoaixe chứ không thể trỏ thẳng trực tiếp đến thuộc tính loaixe vì thuộc tính loaixe đang ở mức **private**.  
  
**Như bài này là sai:**

|  |  |
| --- | --- |
| 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 | // Lớp Xe  class Xe  {      private $loaixe;        function getLoaixe()      {          return $this->loaixe;      }        function setLoaixe($loaixe)      {          $this->loaixe = $loaixe;      }  }    // Khởi tạo một lớp đối tượng xe  $xe = new Xe();    // Gán giá trị cho thuộc tính loại xe  // nhưng sai tại vì loaixe đang ở mức private  // nên không thể truy xuất bên ngoài  $xe->loaixe = 'Wave S';    // Lấy giá trị thuộc tính loại xe  // cũng sai vì loaixe là private  echo $xe->loaixe; |

**Ví dụ có kế thừa:**

|  |  |
| --- | --- |
| 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  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67 | // Lớp Xe  class Xe  {      private $loaixe;        var $tenxe;        function getLoaixe()      {          return $this->loaixe;      }        function setLoaixe($loaixe)      {          $this->loaixe = $loaixe;      }        private function xoaLoaixe()      {          echo 'Hàm xóa loại xe';      }  }    // Kế thừa từ lớp xe  class XeHonda extends Xe  {      function hienThiThongTin()      {          // lệnh này đúng          echo $this->tenxe;            // lệnh này sai vì thuộc tính loaixe là          // private trong lớp cha          echo $this->loaixe;            // lệnh này đúng          $this->setLoaixe('Wave S');            // lệnh này đúng          echo $this->getLoaixe();            /// Lệnh này sai vì hàm xoaLoaixe là          // private trong lớp cha          $this->xoaLoaixe();      }  }    // -------------------  // Chương trình chính//  // -------------------    // Khởi Tạo mới lớp xe hon da  $xehonda = new XeHonda();    // Gọi hàm hienThiThongTin trong lớp XeHonda  // Các bạn kiểm tra trong hàm này để xem các  // lỗi mà tôi đã ghi chú  $xehonda->hienThiThongTin();    // Lệnh này đúng vì lớp XeHonda kế thừa lớp Xe nên  // nó được kế thừa các thuộc tính và hàm của lớp cha  $xehonda->setLoaixe('Suzuki');  echo $xehonda->getLoaixe();    // Lệnh này sai vì hàm xoaLoaixe trong lớp Xe là  // private nên lớp XeHonda không được kế thừa  $xehonda->xoaLoaixe(); |

Phần ghi chú tôi đã giải thích rõ rồi nên tôi không giải thích gì thêm.

**2. Mức truy cập protected**

**Mức truy cập protected** chỉ cho phép truy xuất nội bộ trong lớp đó và lớp kế thừa, riêng ở bên ngoài lớp sẽ không truy xuất đc. Mức protected thường được dùng cho những phương thức và thuộc tính có khả năng bị lớp con định nghĩa lại (*overwrite*).

**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  37  38  39 | // Lớp Xe  class Xe  {      protected $loaixe;  }    // Kế thừa từ lớp xe  class XeHonda extends Xe  {      function hienThiThongTin()      {          // Lệnh này đúng vì loaixe đang ở          // mức protected nên nó được kế thừa          $this->loaixe = 'Wave S';      }      protected function suaChuaXe()      {          // Lệnh      }  }    // ------------------//  // Chuong Trình Chính//  // ------------------//    $honda = new XeHonda();    // Lệnh này sai vì loaixe đang  // ở mức protected nên ở bên ngoài lớp  // không được truy xuất vào  $honda->loaixe = 'Wave Tàu';    // Lệnh này đúng  $honda->hienThiThongTin();    // Lệnh này sai vì: function suaChuaXe đang  // ở mức protected nên ko thể truy xuất  // từ ngoài lớp XeHonDa  $honda->suaChuaXe(); |

**3. Mức truy cập public**

Đây là mức truy cập thoáng nhất bởi vì bạn có thể truy cập tới các phương thức và thuộc tính ở bất cứ đâu, dù trong nộ bộ của lớp hay ở lớp con hay cả bên ngoài lớp đều được.

**Khi khai báo thuộc tính** là public ta có thể dùng từ khóa **var** để thay thế cho public như những ví dụ các bài trước.

**Ví dụ:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | class Xe  {      private $loaixe; // Mức private      protected $tenxe; // Mức protected      var $sokhung; // Mức public      public $soseri; // Mức public  } |

**Khi khai báo với hàm** là public nếu ta không đặt từ public ở đầu thì php sẽ tự hiểu hàm này là public.

**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 | class Xe  {      // Ở dạng public vì ta không có từ khóa gì ở trước      function showTenXe()      {          // lệnh      }        // Ở dạng public vì có từ khóa public ở trước      public function showHangXe()      {          // lệnh      }        // Ở dạng private vì có từ khóa private ở trước      private function setTenxe()      {          // lệnh      }        // Ở dạng protected vì có từ khóa protected ở trước      protected function getTenXe()      {          // lệnh      }  } |