

Как создать класс в PHP?

Класс – это набор данных и методов, имеющих общую, целостную, хорошо определенную сферу ответственности.

```
class имя_класса {  
    описание_класса  
}
```

```
class Person {  
    /* описание класса */  
}
```

Как объявить свойство класса?

Данные (свойства) – переменные, объявленные внутри класса.

Термины-синонимы: атрибуты, поля, данные, свойства-члены.

Необязательный компонент класса: класс может включать только методы, предоставляющие целостный набор услуг.

```
class Person {  
    private $uid = 0; # свойство со значением по-умолчанию  
    public $name;  
    public $age;  
}
```

Как создать объект в PHP?

Объекты – создаются на основе заранее определенных классов при помощи оператора **new**.

```
$linus = new Person();
```

```
$linus->name = 'Linus Torvalds'; # устанавливаем значение свойства
```

```
$linus->age = 45; # устанавливаем значение свойства
```

```
echo $linus->name; # получаем значение свойства
```

```
echo $linus->age; # получаем значение свойства
```

Для доступа к свойствам и методам объекта используется оператор **'->'**.

Объекты всегда **передаются** в функцию **или присваиваются** переменной **по ссылке**. Копию объекта можно создать с помощью оператора **clone**.

Как объявить метод класса?

Методы – это функции, объявленные внутри класса.

Методы позволяют объектам выполнять задачи.

```
class Person {  
    private $uid = 0;  
    protected $name;  
    protected $age;  
  
    public function __construct($name, $age) {  
        $this->uid = uniqid(); # устанавливаем значение свойства  
        $this->name = $name;   # устанавливаем значение свойства  
        $this->age = $age;     # устанавливаем значение свойства  
    }  
  
    public function getid() {  
        return $this->uid;    # получаем значение свойства  
    }  
}
```

Псевдо-переменная **\$this** доступна **только в контексте объекта** и является **ссылкой на вызываемый объект**.

Зачем нужен конструктор?

```
void __construct ( [ mixed args [ , ... ] ] )
```

Конструктор вызывается автоматически при создании экземпляра объекта и **выполняет инициализацию объекта**.

```
class Person {  
    private $uid = 0;  
    protected $name;  
    protected $age;  
  
    public function __construct($name, $age) {  
        $this->uid = uniqid();  
        $this->name = $name;  
        $this->age = $age;  
    }  
}
```

```
$linus = new Person('Linus Torvalds', 45);  
$richard = new Person('Richard Stallman', 62);
```

Что такое наследование?

Наследование – это механизм, позволяющий описать новый класс на основе уже существующего (родительского, базового) класса.

Класс, полученный в результате наследования, называют **потомком** или **дочерним** классом.

Дочерний класс наследует все свойства и методы своего **родителя**.

Дочерний класс расширяет функциональность родительского класса.

В PHP для указания наследования в описании класса используется ключевое слово **extends**.

PHP **не поддерживает** множественное наследование.

```
class Person {  
    private $uid;  
    protected $name;  
  
    public function __construct($name) {  
        $this->uid = uniqid();  
        $this->name = $name;  
    }  
  
    public function getId() {  
        return $this->uid;  
    }  
  
    public function getName() {  
        return $this->name;  
    }  
}
```

```
class Developer extends Person {  
    protected $skills = [];  
  
    public function __construct($name, array $skills) {  
        parent::__construct($name);  
        $this->skills = $skills;  
    }  
}
```

```
$developer = new Developer('Linus Torvalds', ['C++', 'Bash']);
```

Модификаторы доступа

PUBLIC – Открытый тип доступа.

Разрешает доступ из любого контекста.

PROTECTED – Защищенный тип доступа.

Разрешает доступ только внутри класса и его потомков, но запрещает доступ извне.

PRIVATE – Закрытый тип доступа.

Разрешает доступ только в пределах текущего класса.


```
class Person {  
    private $uid;  
    protected $name;  
  
    public function __construct($name) {  
        $this->uid = uniqid();  
        $this->name = $name;  
    }  
  
    public function getId() {  
        return $this->uid;  
    }  
}
```

```
class Developer extends Person {  
    protected $skills = [];  
  
    public function __construct($name, array $skills) {  
        parent::__construct($name);  
        $this->skills = $skills;  
    }  
  
    public function getName() {  
        return $this->name;  
    }  
}
```

```
$developer = new Developer('Linus Torvalds', ['C++', 'Bash']);
```

Как объявить константу класса?

Аналогичны обычным константам, только объявляются внутри класса.

```
class Developer extends Person {  
    const SKILL_PHP = 1;      # объявление  
    const SKILL_JAVASCRIPT = 2; # объявление  
  
    /* other code */  
  
    public function isPhpDeveloper() {  
        return in_array(self::SKILL_PHP, $this->skills); # доступ  
    }  
}  
  
echo Developer::SKILL_JAVASCRIPT; # доступ
```

В PHP $\geq 7.1.0$ для констант можно задать модификатор доступа.

Как объявить статическое свойство или метод?

Свойства и методы, доступ к которым можно получить без создания экземпляра объекта, называются **статическими**.

Объявляются с помощью ключевого слова **static**.

Шаблон проектирования — единственный экземпляр объекта

```
class Singleton {  
    private static $instance;  
  
    private function __construct() {}  
  
    public static function getInstance() {  
        if (!self::$instance) {  
            self::$instance = new self();  
        }  
        return self::$instance; # доступ к статическому свойству  
    }  
}
```

```
$obj = Singleton::getInstance(); # вызов статического метода
```

Оператор разрешения области видимости

"Двойное двоеточие" – это лексема, позволяющая обращаться к статическим свойствам, константам и перегруженным методам класса.

parent:: – ссылка на переопределенный родительский метод.

self:: – ссылка на текущий класс.

static:: – позднее статическое связывание (PHP \geq 5.3.0).

Ссылка на вызываемый класс в контексте статического наследования.

В контексте класса можно создать новый объект через

new self() и **new parent()**.

Как типизировать аргумент?

В PHP 5 добавили возможность уточнять тип данных для аргументов функций и методов.

```
class Team {  
    public function addDeveloper(Developer $developer) {  
        array_push($this->members, $developer);  
    }  
}  
  
$team = new Team();  
  
$linus = new Developer('Linus Torvalds', 45);  
$team->addDeveloper($linus);  
  
$team->addDeveloper(new Developer('Richard Stallman', 62));
```

Уточнение можно использовать для классов, интерфейсов, массивов (**array**) и с PHP >= 5.4.0 функций обратного вызова (**callable**).

С PHP >= 7.0.0 - для элементарных типов.

Но нельзя использовать для трейтов.