Search

Keyboard Shortcuts

?
: This help

j
: Next menu item

k
: Previous menu item

g p
: Previous man page

g n
: Next man page

G
: Scroll to bottom

g g
: Scroll to top

g h
: Goto homepage

g s
: Goto search
(current page)

/
: Focus search box

Типизированные перечисления »
« Обзор перечислений

- Руководство по PHP
- Справочник языка
- Перечисления

Change language: Russian

# Основы перечислений

Перечисления похожи на классы и работают с теми же пространствами имён, что и классы, интерфейсы и трейты. Их как и классы можно загружать автоматически. Перечисления определяют новый тип с фиксированным ограниченным количеством возможных допустимых значений.

```
<?php

enum Suit
{
case Hearts;
case Diamonds;
case Clubs;
case Spades;
}
?>
```

Это объявление создаёт новый перечислимый тип с именем `Suit`, у которого четыре и только четыре допустимых значения: `Suit::Hearts`, `Suit::Diamonds`, `Suit::Clubs` и `Suit::Spades`. Переменным может быть присвоено одно из этих допустимых значений. Функция может проверить тип аргумента на соответствие типу перечисления и тогда можно передать только значения заданного типа.

```
<?php

function pick_a_card(Suit $suit)
{
/* ... */
}

$val = Suit::Diamonds;

// OK
pick_a_card($val);

// OK
pick_a_card(Suit::Clubs);

// TypeError: pick_a_card(): Argument #1 ($suit) must be of type Suit, string given
pick_a_card('Spades');
?>
```

В перечислении может быть ноль или более определений `case`, максимальное количество не ограничено. Перечисление без вариантов синтаксически корректно, хотя и бесполезно.

Для вариантов перечисления работают те же правила синтаксиса, что и для любой метки в PHP, смотрите Константы.

По умолчанию варианты не поддерживаются скалярным значением. То есть `Suit::Hearts` не равно «`0`». Вместо этого каждый вариант поддерживается одноэлементным объектом с таким именем. То есть:

```
<?php

$a = Suit::Spades;
$b = Suit::Spades;

$a === $b; // true

$a instanceof Suit; // true
?>
```

Это также означает, что значения перечисления не будут < или > друг с друга, поскольку эти сравнения не имеют смысла для объектов. Сравнения всегда будут возвращать **`false`** при работе с вариантами перечисления.

Тип варианта без связанных данных называется «Чистый вариант» (Pure Case). Перечисление, которое содержит только чистые варианты, называется чистым перечислением (Pure Enum).

Все чистые варианты реализованы как экземпляры своего типа перечисления. Тип перечисления внутренне представлен как класс.

У всех вариантов есть доступное только для чтения свойство `name` — это и чувствительное к регистру имя самого варианта.

```
<?php

print Suit::Spades->name;
// prints "Spades"
?>
```

Можно также пользоваться функциями defined() и constant() для проверки существования или чтения регистра перечисления, если имя получено динамически. Однако поступать так не рекомендовано, поскольку в большей части ситуаций лучше работать с типизированными перечислениями.



## User Contributed Notes

There are no user contributed notes for this page.