Типы в языке

Они здесь

Булев тип *(boolean)* — Выражает истинность значения. Может быть либо **TRUE**, либо **FALSE**.

Целые числа *(integer)* — Множество натуральных чисел. Например: 42, 1337.

Числа с плавающей точкой *(float, double, real)* — Множество вещественных чисел. Например: 25.17, 4.20.

Строки *(string)* — Это набор символов. Строка определяется либо одинарными, либо двойными кавычками. Есть также редко используемые способы определения строки: heredoc и nowdoc.
В отличие от синтаксиса двойных кавычек и heredoc, переменные и управляющие последовательности для специальных символов, заключенных в одинарные кавычки, *не обрабатываются*.

Массивы *(array)* — Некие упорядоченные структуры, которые устанавливают соответствие между значением и ключом.
Создаются двумя способами: языковой конструкцией **array()** либо с помощью квадратных скобок [] (Начиная с PHP 5.4), приняв в качестве параметров пары ключ => значение(key => value), разделенные запятой.
Примеры:

```
array = array( "foo" => "bar", "bar" => "foo", );
array = [ "foo" => "bar", "bar" => "foo", ];
```

Объекты *(object)* — Экземпляры класса. Создаются с помощью ключевого слова *new*.

Ресурсы *(resource)* — Специальные переменные, которые содержат ссылку на внешний ресурс. Ресурсом может быть открытый файл, соединение с базой данных, изображение и т.д.

null — Переменная без значения. Переменная считается *null*, если ей было присвоено значение **NULL**, ей не было присвоено никакого значения или она была удалена функцией **unset().**

callable *(или callback)* — Функции обратного вызова.

Подробнее

Часто используемые функции

Работа со строками

explode(delimiter, string) — Разбивает строку с помощью разделителя

implode(glue, pieces) — Объединяет элементы массива в строку

trim(string) — Удаляет пробелы(или другие символы) из начала и конца строки

rtrim(string) — Удаляет пробельные(или другие символы) из конца строки

ltrim(string) — Удаляет пробельные(или другие символы) из начала строки

md5(string) — Возвращает MD5-хэш строки

sha1(string) — Возвращает SHA1-хэш строки

nl2br(string) — Вставляет HTML-код разрыва строки перед каждым переводом строки

str_replace(search, replace, subject) — Заменяет все вхождения строки поиска на строку замены

str_ireplace(search, replace, subject) — Регистронезависимый вариант функции str_replace

strip_tags(string) — Удаляет HTML и PHP-теги из строки

strlen(string) — Возвращает длину строки

strrev(string) — Переворачивает строку задом наперед

strpos(haystack, needle) — Возвращает позицию первого вхождения подстроки без учета регистра

strtolower(string) — Преобразует строку в нижний регистр

strtoupper(string) — Преобразует строку в верхний регистр

substr(str, start) — Возвращает подстроку

strstr(haystack, needle) — Находит первое вхождение подстроки

htmlspecialchars(string) — Преобразует специальные символы в HTML-сущности

htmlspecialchars_decode(string) — Преобразует специальные HTML-сущности обратно в соответствующие символы

htmlentities(string) — Преобразует все возможные символы в соответствующие HTML-сущности

wordwrap(string) — Переносит строку по указанному количеству символов

Подробнее

Работа с массивами

count(array_or_countable) — Подсчитывает количество элементов массива или что-то в объекте

array_diff(array1, array2) — Вычисляет расхождение массивов

array_intersect(array1, array2) — Вычисляет схождение массивов

array_key_exists(key, array) — Проверяет, присутствует ли в массиве указанный ключ или индекс

array_keys(array) — Возвращает все или некоторое подмножество ключей массива

array_values(array) — Выбирает все значения массива

array_merge(array1) — Сливает один или большее количество массивов

array_rand(array) — Выбирает одно или несколько случайных значений из массива

array_reverse(array) — Возвращает массив с элементами в обратном порядке

compact(varname1) — Создает массив, содержащий названия переменных и их значения

extract(array) — Импортирует переменные из массива в текущую таблицу символов

arsort(array) — Сортирует массив в обратном порядке, сохраняя ключи

asort(array) — Сортирует массив, сохраняя ключи

sort(array) — Сортирует массив

rsort(array) — Сортирует массив в обратном порядке

array_combine(keys, values) — Создает новый массив, используя один массив в качестве ключей, а другой в качестве соответствующих значений

array_search(needle, haystack) — Осуществляет поиск данного значения в массиве и возвращает ключ первого найденого элемента в случае удачи

array_shift(array) — Извлекает и возвращает первый элемент массива

array_unique(array) — Убирает повторяющиеся значения из массива

array_unshift(array, value) — Добавляет один или несколько элементов в начало массива

array_flip(array) — Меняет местами ключи с их значениями в массиве

array_pop(array) — Извлекает и возвращает последний элемент массива

array_push(array, value) — Добавляет один или несколько элементов в конец массива

in_array(needle, haystack) — Проверяет, присутствует ли в массиве значение

list(var1, var2) — Присваивает переменным из списка значения подобно массиву

Подробнее

Работа с датой

checkdate(month, day, year) — Проверяет корректность даты по григорианскому календарю

date(format, timestamp) — Форматирует вывод системной даты/времени

getdate(timestamp) — Возвращает информацию о дате/времени

mktime(hr, min, sec, month, day, yr) — Возвращает метку времени Unix для заданной даты

strftime(formatstring, timestamp) — Форматирует текущую дату/время с учетом текущих настроек локали

strtotime(str) — Преобразует текстовое представление даты на английском языке в метку времени Unix

time() — Возвращает текущую метку времени Unix

Подробнее

Форматы даты и времени

Y — 4-значный год(2007)

y — 2-значный год(07)

F — Полное название месяца(January)

M — Короткое название месяца(Jan)

m — Месяц(от 01 до 12)

n — Месяц(от 1 до 12)

D — Короткое название дня(Mon)

l — Полное название дня(Monday)

d — День(от 01 до 31)

j — День(от 1 до 31)

h — 12-часовой формат(от 01 до 12)

g — 12-часовой формат(от 1 до 12)

H — 24-часовой формат(от 00 до 23)

G — 24-часовой формат(от 0 до 23)

i — Минуты(от 00 до 59)

s — Секунды(от 00 до 59)

w — День недели(от 0 до 6)

z — День года(от 0 до 365)

W — Неделя года(от 1 до 53)

t — Дней в месяце(от 28 до 31)

Подробнее

Работа с файлами

clearstatcache() — Очищает кэш состояния файлов

copy(source, dest) — Копирует файл

fclose(handle) — Закрывает открытый дескриптор файла

fgets(handle, len) — Читает строку из файла

file(file) — Читает содержимое файла и помещает его в массив

filemtime(file) — Возвращает время последнего изменения файла

filesize(file) — Возвращает размер файла

file_exists(file) — Проверяет наличие указанного файла или каталога

fopen(file, mode) — Открывает файл или URL

fread(handle, len) — Бинарно-безопасное чтение файла

fwrite(handle, str) — Бинарно-безопасная запись в файл

readfile(file) — Выводит файл

file_get_contents(file) — Читает содержимое файла в строку

file_put_contents(file, data, flag) — Пишет строку в файл

move_uploaded_file(file, destination) — Перемещает загруженный файл в новое место

Подробнее

Режимы работы с файлами

r — Чтение, указатель в начале

r+ — Чтение и запись, указатель в начале

w — Запись, указатель в начале. Файл обрезается до нулевой длины

w+ — Чтение и запись, указатель в начале. Файл обрезается до нулевой длины

a — Запись, указатель в конце

a+ — Чтение и запись, указатель в конце

PDO

Соединение с базой данных

$pdo = new PDO(*Строка для соединения с базой данных*);

Строки для соединения

MySQL: ("mysql:host=hostname;dbname=mysql", "username", "password")

SQLite: ("sqlite:/path/to/database.db") или ("sqlite::memory:")

PostgreSQL: ("pgsql:dbname=pdo;host=hostname", "username", "password")

Oracle: ("OCI:dbname=mydatabase;charset=UTF-8", "username", "password")

Выполнение запроса без выборки

```
$sql = "INSERT INTO users(name, email) VALUES('john', 'john@smith.com')";
$result = $pdo->exec($sql);
```

Выборка данных

```
$sql = "SELECT id, name FROM users";
$stmt = $pdo->query($sql);
```

Обработка результата

```
$res = $stmt->fetch();
$res = $stmt->fetch(PDO::FETCH_NUM);
$res = $stmt->fetch(PDO::FETCH_ASSOC);
$res = $stmt->fetch(PDO::FETCH_OBJ);
$res = $stmt->fetchAll();
```

Подготовленные запросы

```
$sql = 'SELECT email FROM users WHERE id = :id AND name = :name';
$stmt = $pdo->prepare($sql);
$stmt->execute( [':id' => 5, ':name' => 'John'] );
$res = $stmt->fetchAll();
```

Регулярные выражения

preg_grep(pattern, arr) — Возвращает массив вхождений, которые соответствуют шаблону

preg_match(pattern, str) — Выполняет проверку на соответствие регулярному выражению

preg_match_all(pattern, str, arr) — Выполняет глобальный поиск шаблона в строке

preg_replace(pattern, replace, str) — Выполняет поиск и замену по регулярному выражению

preg_split(pattern, str) — Разбивает строку по регулярному выражению

Подробнее

Синтаксис регулярных выражений

^ — Начало строки

$ — Конец строки

. — Любой символ, кроме переноса строки

(a|b) — a или b

(...) — Группа

[abc] — Диапозон (a, b или c)

[^abc] — Не a, не b и не c

[a-z] — Буква между a и z

[A-Z] — Буква в верхнем регистре между A и Z

***** — 0 или больше

***?** — 0 или больше, нежадный

+ — 1 или больше

+? — 1 или больше, нежадный

{3} — Ровно 3

{3,} — 3 или больше

{3,5} — от 3 до 5

{3,5}? — от 3 до 5, нежадный

Модификаторы шаблонов

g — Глобальный поиск

i — Регистронезависимый поиск

s — Считать текст одной строкой

m — Многострочный текст

x — Разрешить комментарии и пробелы в шаблоне

e — Выполнение подстановки

U — Нежадный шаблон