

PHP 快速参考

语法、数组、OOP、数据库与文件 I/O 要点

基础

Hello World

```
<?php
echo "Hello, World!\n";
// PHP 代码必须在 <?php ... ?> 标签内
```

运行 PHP

```
php script.php # 执行文件
php -r 'echo "hi\n";' # 内联执行
php -S localhost:8000 # 内置开发服务器
```

注释

```
// 单行注释
# 也是单行注释
/* 多行
   注释 */
```

变量与类型

变量

```
$name = "PHP"; // 字符串
$version = 8.3; // 浮点数
$count = 42; // 整数
$active = true; // 布尔
$item = null; // null
```

类型检查

gettype(\$x)	以字符串返回类型
is_string(\$x)	是否为字符串
is_int(\$x)	是否为整数
is_array(\$x)	是否为数组
is_null(\$x)	是否为 null
isset(\$x)	是否已设置且不为 null
empty(\$x)	是否为空 (假值)

类型转换

```
$n = (int) "42"; // 42
$s = (string) 3.14; // "3.14"
$b = (bool) ""; // false
$a = (array) $obj; // 对象转数组
```

常量

```
define("MAX_SIZE", 100);
const API_VERSION = "v2";
echo MAX_SIZE; // 100
```

字符串

字符串基础

```
$name = "World";
echo "Hello, $name!"; // 变量插值
echo 'Hello, $name!'; // 字面字符串 (无插值)
echo "Value: {$arr['key']}"; // 复杂表达式
```

字符串函数

strlen(\$s)	字符串字节长度
mb_strlen(\$s)	字符串字符长度 (多字节安全)
strtolower(\$s)	转小写
strtoupper(\$s)	转大写
trim(\$s)	去除首尾空白
str_replace(a, b, \$s)	将 \$s 中的 a 替换为 b
substr(\$s, 0, 5)	从位置 0 截取长度 5 的子串
strpos(\$s, 'find')	查找子串位置 (未找到返回 false)
explode(' ', \$s)	将字符串分割为数组
implode(' ', \$a)	将数组合并为字符串

Heredoc 与 Nowdoc

```
$html = <<<HTML
<p>Hello, $name</p>
HTML;
$raw = <<<'TEXT'
No $interpolation here
TEXT;
```

数组

索引数组与关联数组

```
$nums = [1, 2, 3]; // 索引数组
$user = ["name" => "Alice", "age" => 30]; // 关联数组
$num = 4; // 追加
echo $user["name"]; // 访问
```

数组函数

count(\$a)	元素数量
array_push(\$a, \$v)	追加到末尾
array_pop(\$a)	移除并返回最后一个元素
array_merge(\$a, \$b)	合并两个数组
in_array(\$v, \$a)	检查值是否存在
array_key_exists(\$k, \$a)	检查键是否存在
array_map(\$fn, \$a)	对每个元素应用函数
array_filter(\$a, \$fn)	按回调过滤元素
sort(\$a)	原地排序 (重新索引)
array_keys(\$a)	返回所有键

遍历

```
foreach ($users as $user) { echo $user; }
foreach ($map as $key => $value) {
    echo "$key: $value\n";
}
```

函数

基础函数

```
function add(int $a, int $b): int {
    return $a + $b;
}
echo add(3, 5);
```

默认参数与命名参数

```
function greet(string $name, string $greeting = "Hello"): string {
    return "$greeting, $name!";
}
greet("Alice");
greet(greeting: "Hi", name: "Bob"); // 命名参数 (PHP 8+)
```

箭头函数

```
$double = fn(int $x): int => $x * 2;
$num = array_map(fn($n) => $n * 10, [1, 2, 3]);
```

闭包

```
$factor = 3;
$multiply = function(int $x) use ($factor): int {
    return $x * $factor;
};
echo $multiply(5); // 15
```

类与对象

类定义

```
class User {
    public function __construct(
        private string $name,
        private int $age = 0,
    ) {}
    public function greet(): string { return "Hi, {$this->name}"; }
}
```

继承与接口

```
interface Printable {
    public function toString(): string;
}
class Admin extends User implements Printable {
    public function toString(): string { return "Admin"; }
}
```

访问控制

public	任何地方均可访问
protected	本类及子类可访问
private	仅本类内部可访问
readonly	只能赋值一次 (PHP 8.1+)
static	属于类本身, 非实例
abstract	必须由子类实现

Trait

```
trait Timestamped {
    public function createdAt(): string {
        return date('Y-m-d H:i:s');
    }
}
class Post { use Timestamped; }
```

错误处理

try / catch / finally

```
try {
    $result = riskyOperation();
} catch (InvalidArgumentException $e) {
    echo "Bad input: " . $e->getMessage();
} catch (Exception $e) {
    echo "Error: " . $e->getMessage();
} finally { cleanup(); }
```

自定义异常

```
class ApiException extends RuntimeException {
    public function __construct(string $message, private int $statusCode = 500) {
        parent::__construct($message, $statusCode);
    }
}
```

Null 安全 (PHP 8+)

```
$len = $user?->address?->zip; // 空安全运算符
$name = $input ?? "default"; // null 合并
$data ??= []; // null 合并赋值
```

PHP 快速参考

文件 I/O

读写文件

```
$content = file_get_contents("data.txt");
file_put_contents("out.txt", $content);
$lines = file("data.txt", FILE_IGNORE_NEW_LINES);
```

文件句柄

```
$f = fopen("log.txt", "a");
fwrite($f, "entry\n");
fclose($f);
```

文件函数

file_exists(\$path)	检查文件是否存在
is_dir(\$path)	检查路径是否为目录
mkdir(\$path, 0755, true)	递归创建目录
unlink(\$path)	删除文件
glob('*.*txt')	按模式查找文件
realpath(\$path)	解析为完整绝对路径

数据库

PDO 连接

```
$pdo = new PDO(
    "mysql:host=localhost;dbname=app",
    "user", "password",
    [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]
);
```

预处理语句

```
$stmt = $pdo->prepare("SELECT * FROM users WHERE id = :id");
$stmt->execute([':id' => 42]);
$user = $stmt->fetch(PDO::FETCH_ASSOC);
```

插入与更新

```
$stmt = $pdo->prepare("INSERT INTO users (name, email) VALUES
(?, ?)");
$stmt->execute(["Alice", "alice@example.com"]);
$id = $pdo->lastInsertId();
```

PDO 获取模式

fetch()	获取单行
fetchAll()	获取所有行
FETCH_ASSOC	以关联数组返回
FETCH_OBJ	以匿名对象返回
FETCH_CLASS	以指定类实例返回

常用函数

JSON

```
$json = json_encode(["name" => "Alice", "age" => 30]);
$data = json_decode($json, true); // true = 关联数组
$data = json_decode($json); // 对象
```

日期与时间

```
echo date("Y-m-d H:i:s"); // 2026-03-26 12:00:00
$ts = strtotime("+1 week");
$dt = new DateTime("2026-01-01");
echo $dt->format("D, M j"); // Thu, Jan 1
```

数学与随机数

abs(\$n)	绝对值
round(\$n, 2)	四舍五入到 2 位小数
ceil(\$n) / floor(\$n)	向上 / 向下取整
min(\$a, \$b) / max(\$a, \$b)	最小值 / 最大值
random_int(1, 100)	密码学安全的随机整数
number_format(\$n, 2)	格式化数字 (带千位分隔符)

正则表达式

```
preg_match('/^[a-z]+$/i', $str, $matches);
preg_match_all('/\d+/', $str, $all);
$result = preg_replace('/\s+/', ' ', $str);
```