PHP Riferimento Rapido

Sintassi, array, OOP, database, I/O su file essenziali

Basi

Hello World

```
<?php
echo "Hello, World!\n";
// Il codice PHP deve essere dentro i tag <?php ... ?>
```

Eseguire PHP

```
php script.php          # esegui un file
php -r 'echo "hi\n";'  # esegui codice inline
php -S localhost:8000   # server di sviluppo integrato
```

Commenti

```
// commento su una riga
# anche su una riga
/* commento
   multi-riga */
```

Variabili e Tipi

Variabili

```
$name = "PHP";          // stringa
$version = 8.3;         // float
$count = 42;            // int
$active = true;         // bool
$items = null;          // null
```

Verifica del Tipo

`gettype($x)`	Restituisce il tipo come stringa
`is_string($x)`	Verifica se è una stringa
`is_int($x)`	Verifica se è un intero
`is_array($x)`	Verifica se è un array
`is_null($x)`	Verifica se è null
`isset($x)`	Verifica se è impostato e non null
`empty($x)`	Verifica se è vuoto (falsy)

Cast dei Tipi

```
$n = (int) "42";       // 42
$s = (string) 3.14;    // "3.14"
$b = (bool) "";        // false
$a = (array) $obj;     // oggetto in array
```

Costanti

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

Stringhe

Basi delle Stringhe

```
$name = "World";
echo "Hello, $name!";         // interpolazione variabile
echo 'Hello, $name!';         // letterale (senza interpolazione)
echo "Value: {$arr['key']}";  // espressione complessa
```

Funzioni su Stringhe

`strlen($s)`	Lunghezza stringa in byte
`mb_strlen($s)`	Lunghezza in caratteri (sicura per multibyte)
`strtolower($s)`	Converti in minuscolo
`strtoupper($s)`	Converti in maiuscolo
`trim($s)`	Rimuovi spazi da entrambi i lati
`str_replace(a, b, $s)`	Sostituisce **a** con **b** in **$s**
`substr($s, 0, 5)`	Sottostringa dalla posizione 0, lunghezza 5
`strpos($s, 'find')`	Trova posizione sottostringa (false se non trovata)
`explode(',', $s)`	Divide la stringa in array
`implode(',', $a)`	Unisce l'array in stringa

Heredoc e Nowdoc

```
$html = <<<HTML
<p>Hello, $name</p>
HTML;
$raw = <<<'TEXT'
Nessuna $interpolazione qui
TEXT;
```

Array

Indicizzati e Associativi

```
$nums = [1, 2, 3];                // indicizzato
$user = ["name" => "Alice", "age" => 30];  // associativo
$nums[] = 4;                       // aggiunge in fondo
echo $user["name"];                // accesso
```

Funzioni su Array

`count($a)`	Numero di elementi
`array_push($a, $v)`	Aggiunge alla fine
`array_pop($a)`	Rimuove e restituisce l'ultimo elemento
`array_merge($a, $b)`	Unisce due array
`in_array($v, $a)`	Verifica se il valore esiste
`array_key_exists($k, $a)`	Verifica se la chiave esiste
`array_map($fn, $a)`	Applica funzione a ogni elemento
`array_filter($a, $fn)`	Filtra elementi per callback
`sort($a)`	Ordina in-place (reindizza)
`array_keys($a)`	Restituisce tutte le chiavi

Iterazione

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

Funzioni

Funzione di Base

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

Argomenti di Default e Nominati

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

Funzioni Freccia

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

Closure

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

Classi e Oggetti

Definizione di Classe

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

Ereditarietà e Interfacce

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

Visibilità

`public`	Accessibile da qualsiasi punto
`protected`	Accessibile dalla classe e dalle sottoclassi
`private`	Accessibile solo all'interno della classe
`readonly`	Può essere assegnato solo una volta (PHP 8.1+)
`static`	Appartiene alla classe, non alle istanze
`abstract`	Deve essere implementato dalla sottoclasse

Trait

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

Gestione degli Errori

Try / Catch / Finally

```
try {
    $result = riskyOperation();
} catch (InvalidArgumentException $e) {
    echo "Input non valido: " . $e->getMessage();
} catch (Exception $e) {
    echo "Errore: " . $e->getMessage();
} finally { cleanup(); }
```

Eccezioni Personalizzate

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

Null Safety (PHP 8+)

```
$len = $user?->address?->zip;  // operatore nullsafe
$name = $input ?? "default";   // null coalescing
$data ??= [];                  // assegnazione null coalescing
```

PHP Riferimento Rapido

I/O su File

Leggi e Scrivi File

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

Handle di File

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

Funzioni sui File

`file_exists($path)`	Verifica se il file esiste
`is_dir($path)`	Verifica se il percorso è una directory
`mkdir($path, 0755, true)`	Crea directory in modo ricorsivo
`unlink($path)`	Elimina un file
`glob('*.txt')`	Trova file corrispondenti al pattern
`realpath($path)`	Risolve il percorso assoluto completo

Database

Connessione PDO

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

Istruzioni Preparate

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

Insert e Update

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

Modalità Fetch PDO

`fetch()`	Recupera una riga singola
`fetchAll()`	Recupera tutte le righe
`FETCH_ASSOC`	Restituisce come array associativo
`FETCH_OBJ`	Restituisce come oggetto anonimo
`FETCH_CLASS`	Restituisce come istanza della classe specificata

Funzioni Comuni

JSON

```
$json = json_encode(["name" => "Alice", "age" => 30]);
$data = json_decode($json, true);  // true = array associativo
$data = json_decode($json);        // oggetto
```

Data e Ora

```
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
```

Matematica e Casualità

`abs($n)`	Valore assoluto
`round($n, 2)`	Arrotonda a 2 decimali
`ceil($n) / floor($n)`	Arrotonda per eccesso / difetto
`min($a, $b) / max($a, $b)`	Minimo / massimo
`random_int(1, 100)`	Intero casuale crittograficamente sicuro
`number_format($n, 2)`	Formatta con separatore delle migliaia

Espressioni Regolari

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