

Lösungen zu den Aufgaben

Kapitel 1

Aufgabe 1

Tippen Sie *date* in das Suchfeld oben rechts ein, der erste Treffer führt Sie zur Dokumentation der Funktion. Als Abkürzung können Sie einen beliebigen Funktionsnamen auch direkt an die URL anhängen:

<https://www.php.net/date>.

Aufgabe 2

Der Browser versucht sein bestes zur Anzeige der verschiedenen Dateitypen. Da der Browser kein PHP interpretiert, zeigt er den Quelltext der PHP-Datei an. TXT wird ebenso als reiner Text dargestellt. Da der Browser HTML interpretiert, wird die HTML-Datei wie eine Webseite dargestellt. Eine Darstellung der ZIP-Datei ist nicht möglich, sie wird zum Download angeboten. Für PDFs haben Browser ein eingebautes Betrachtungsprogramm, sie werden damit angezeigt.

Aufgabe 3

Zwischen den mittleren beiden Zeilen könnte man einen Absatz erwarten. *Endet* eine Zeile jedoch mit einem schließenden PHP-Tag `?>`, so »schluckt« PHP den folgenden Zeilenumbruch.

Aufgabe 4

<code>php -h</code>	Zeigt Hilfetext zur Verwendung des <code>php</code> -Befehls.
<code>php -m</code>	Zeigt die aktivierten PHP-Erweiterungen (<i>Extensions</i>) an.
<code>php -i</code>	Gibt Informationen zur PHP-Installation aus, u.a. alle Einstellungen, die das Verhalten von PHP beeinflussen.

Kapitel 2

Aufgabe 1

Die Schlüssel bleiben unberührt, es entsteht eine Lücke.

```
$planets = ['Merkur', 'Venus', 'Erde'];  
unset($planets[1]); var_dump($planets);
```

Die Ausgabe:

```
array(2) {  
    [0]=> string(6) "Merkur"  
    [2]=> string(4) "Erde"  
}
```

Aufgabe 2

Für den Wert `true` steht der String `'1'` bzw. die Zahl 1, für `false` steht der leere String bzw. die Zahl 0:

```
var_dump((string) true); // => string(1) "1"  
var_dump((string) false); // leerer String! => string(0) ""  
var_dump((int) true); // => int(1)  
var_dump((int) false); // => int(0)
```

Aufgabe 3

Die Konstante enthält *die größtmögliche darstellbare Ganzzahl* der PHP-Installation, siehe www.php.net/reserved.constants. Für größere Zahlen wechselt PHP auf eine Gleitkommadarstellung, d.h. größere Zahlen sind nicht mehr als Integer darstellbar.

```
> php -r 'var_dump(PHP_INT_MAX);'  
int(9223372036854775807)  
> php -r 'var_dump(PHP_INT_MAX + 1);'  
float(9.223372036854776E+18)
```

Kapitel 3

Aufgabe 1

Der leere String konvertiert bei Auswertung der Bedingung zu `false`:

```
$name = '';
echo $name ? $name : 'Unbekannt'; // => Unbekannt
```

Alle anderen String-Werte konvertieren zu `true`:

```
$name = 'Lea';
echo $name ? $name : 'Unbekannt'; // => Lea
```

Für solche Fälle kennt der ternäre Operator eine Abkürzung:

```
echo $name ?: 'Unbekannt';
```

Oder allgemein ausgedrückt:

```
Bedingung ?: Wert; // ist eine Abkürzung für
Bedingung ? Bedingung : Wert;
```

Falls *Bedingung* den Wahrheitsgehalt `true` hat, ist die Bedingung selbst Rückgabewert des Ausdrucks, ansonsten *Wert*.

Aufgabe 2

```
$letters = ['A', 'B', 'C'];
foreach ($letters as $key => $letter)
    if ($key > 0) { // ab zweitem Durchlauf erfüllt!
        echo '-';
    }
    echo $letter;
} // => A-B-C
```

Aufgabe 3

```
<?php $planets = ['Merkur', 'Venus', 'Erde']; ?>
<?php foreach ($planets as $planet): ?>
    Willkommen auf Planet <?= $planet; ?>.
<?php endforeach; ?>
```

Kapitel 4

Aufgabe 1

```
$jackpot = (int) readline('Jackpot (€): ');
$winners = (int) readline('Anzahl Gewinner: ');
if ($winners > 0) {
    echo '€ je Gewinner: ' . ($jackpot / $winners);
} else {
    echo 'Keine Gewinner';
}
```

Aufgabe 2

Windows: Aktivierung in der php.ini durch Entfernung der Kommentarzeichen (;) vor den Zeilen:

```
extension=gd ; Bildbearbeitung (Kapitel 7)
extension=fileinfo ; Dateierkennung (Kapitel 8)
```

Linux: Unter Ubuntu ist *fileinfo* bereits vorinstalliert, *gd* installieren Sie mit:

```
> sudo apt install php-gd
```

Prüfung mit `php -m` (oder `phpinfo()` im Browser; Kapitel 5):

```
> php -m
... gd fileinfo ...
```

Unter macOS sollten beide Erweiterungen bereits standardmäßig aktiviert sein.

Aufgabe 3

Die Direktive heißt `log_errors` (= On bzw. Off). Vorsicht: Fehlermeldungen sind komplett unsichtbar, wenn Fehleranzeige *und* Logging ausgeschaltet sind.

Kapitel 5

Aufgabe 1

Die Wurzel aus 13, gerundet auf zwei Dezimalstellen, ergibt 3,61:

```
> php -r 'echo round(sqrt(13), 2);'  
3.61
```

Aufgabe 2

Die Funktion `filter_var()` »filtert« Variablen auf gewünschte Kriterien, siehe www.php.net/filter_var. Das erste Argument ist eine fragliche Variable, das zweite gibt den gewünschten Filter als Konstante an. Die Funktion gibt `false` zurück, wenn der Filter nicht zutrifft. Die Prüfung auf eine E-Mail-Adresse erfolgt mit der Filter-Konstanten `FILTER_VALIDATE_EMAIL`:

```
$filter = filter_var('max@beck.de', FILTER_VALIDATE_EMAIL);  
var_dump($filter); // Gültig => string(11) "max@beck.de"  
$filter = filter_var('maxbeck.de', FILTER_VALIDATE_EMAIL);  
var_dump($filter); // Ungültig => bool(false)
```

Aufgabe 3

```
function formatDate(string $date): string {...}  
echo formatDate('2022-03-30'); // OK  
echo formatDate([]); // Fatal error!
```

Beachten Sie auch die Hinweise in den Quellcode-Downloads zu dieser Aufgabe!

Kapitel 6

Aufgabe 1

Bei Aufrufen von `wikipedia.de:80` (HTTP) und `wikipedia.de:443` (HTTPS) entfernt der Browser automatisch die Port-Nummern, da sie Standard für den Abruf von Webseiten sind. Moderne Webseiten kennen nur noch verschlüsselte HTTPS-Verbindungen, weshalb Wikipedia Port 80 auf HTTPS weiterleitet. Unter `wikipedia.de:3306` ist kein Dienst zu erreichen, die Verbindung wird abgewiesen.

Aufgabe 2

Die Ausgabe erfolgt in einer Zeile. In HTML muss der Zeilenumbruch durch das Tag `
` angegeben werden:

```
Hallo <strong><?php echo 'PHP'; ?></strong>!<br/>
Heute ist der <?php echo date('d.m.Y H:i:m'); ?>.
```

Aufgabe 3

index.php:

- `http://localhost:8000/index.php`
- `http://localhost:8000/`
- `http://localhost:8000`

index.html:

- `http://localhost:8000/index.html`
- `http://localhost:8000/` (falls keine `index.php` vorhanden!)
- `http://localhost:8000` (falls keine `index.php` vorhanden!)

Aufgabe 4

Durchsuchen Sie die Ausgabe der Funktion `phpinfo()` mit der Suchfunktion des Browsers ((Strg)+(F)) nach den Direktiven.

- `memory_limit` hat den Standardwert 128M (=128MB).
- `date.timezone` hat keinen expliziten Standardwert (entspricht der Weltzeit UTC); für die deutsche Zeit als Standard können Sie die Direktive auf *Europe/Berlin* setzen.

Aufgabe 5

Die Direktive `default_mimetype` bestimmt den Wert des Content-Type-Headers, falls er nicht explizit gesetzt wird. Der Standardwert der Direktive lautet: `text/html`.

Kapitel 7

Aufgabe 1

```
> cd /Pfad/zu/Code/Download/kapitel-07
> php -S localhost:8000 -t art-project/v1-static
```

Aufgabe 2

Googles URLs zu Suchergebnissen enthalten zahlreiche Parameter. Den Suchbegriff *PHP* finden Sie im GET-Parameter `q` (**query**, engl. für Anfrage). Dieser reicht für ein Suchergebnis aus, daher:

- `https://www.google.de/search?q=PHP`
- `https://www.google.de/search?q=MySQL`

Aufgabe 3

Der Content-Type lautet `application/x-www-form-urlencoded`. Dies bedeutet, dass der Inhalt im Rumpf der Anfrage im Format eines Query-Strings übermittelt wird (statt in der URL wie bei einer GET-Anfrage):

```
POST /script.php HTTP/1.1
Content-Type: application/x-www-form-urlencoded
...
formularfeld1=eingabe1&formularfeld2=eingabe2
```

Enthält das Formular einen Dateupload, muss der Content-Type auf `multipart/form-data` umgeschaltet werden (Kapitel 8).

Kapitel 8

Aufgabe 1

Dateien, die nicht direkt öffentlich abrufbar sein müssen, sollten außerhalb der Document-Root liegen, v.a. sensible Daten.

```
~/website <= Projekt-Verzeichnis
emails.txt <= Nicht-öffentlicher Ablageort
/public <= Document-Root
newsletter.php <= öffentlich, via Browser erreichbar
```

Aufgabe 2

`FILE_APPEND` enthält die Ganzzahl 8:

```
echo FILE_APPEND; // => 8
```

Hinter vordefinierten Konstanten, die PHP als Optionen für Funktionen anbietet, um deren Verhalten zu beeinflussen, verbergen sich meist Ganzzahlen. Ein Aufruf von `file_put_contents()` könnte also ebenso gut lauten:

```
file_put_contents('...', '...', 8); // statt:
file_put_contents('...', '...', FILE_APPEND);
```

Den Namen einer Konstante kann man sich jedoch viel besser merken, er ist im Quelltext aussagekräftiger und leicht von anderen Optionen zu unterscheiden.

Aufgabe 3

```
upload_max_filesize = 15M
post_max_size = 50M
```

Mit maximal 50MB für POST-Daten kann das Formular bis zu drei 15MB große Fotos übermitteln und lässt noch 5MB Platz für andere Textfelder.

Aufgabe 4

Angenommen, eine mp3-Datei `happy.mp3` liegt im Benutzerverzeichnis, so kann der nötige PHP-Code in einem Skript oder auch direkt mit dem `php`-Befehl ausgeführt werden:

```
> php -r "echo mime_content_type('~/' . 'happy.mp3');"
audio/mpeg
```

Der MIME-Type der mp3-Datei lautet `audio/mpeg`.

Kapitel 9

Aufgabe 1

```
setcookie('name', 'Michael');
setcookie('has_accepted_cookie_banner', 'yes');
setcookie('tracking_id', '12345');
```

Prüfen Sie in der Browser-Konsole. Beachten Sie, dass HTTP keine Datentypen kennt, Cookie-Werte sind für PHP wie URL- und Formular-Parameter immer Strings.

Aufgabe 2

Ja, ein Cookie-Wert lässt sich überschreiben, wie Aufrufe mit dynamischer Uhrzeit zeigen:

```
var_dump($_COOKIE);
setcookie('time', 'Uhrzeit: ' . date('H:i:s'));
```

Aufgabe 3

Dies regelt die `php.ini`-Direktive `session.gc_maxlifetime` mit einem Standardwert von 1440 Sekunden (= 24 Minuten). Ist eine Session für diesen Zeitraum ungenutzt, besteht die Möglichkeit, dass PHP die Session-Daten auf dem Server löscht. »gc« steht für »garbage collection« (engl. für Abfallentsorgung).

Kapitel 10

Aufgabe 1

Das Flag heißt `JSON_THROW_ON_ERROR`. Im Fehlerfall wirft `json_decode()` dann eine Ausnahme vom Typ `JsonException` statt `null` zurückzugeben:

```
$json = '["a", "b", "c"; // Eckige Klammer fehlt!
try {
    $data = json_decode($json, true, 512, JSON_THROW_ON_ERROR);
} catch (JsonException $e) {
    echo 'JSON ist ungültig: ' . $e->getMessage();
}
```

Um das vierte Argument setzen zu können, muss auch das dritte Argument gesetzt werden, obwohl es hier nicht von Interesse ist (im Beispiel mit seinem Vorgabewert 512). Seit PHP 8 können ungenutzte Argumente mit einem »benannten Argument« auch elegant übersprungen werden:

```
$data = json_decode($json, true, flags: JSON_THROW_ON_ERROR);
```

Der Name `flags` entspricht dem Parameternamen aus der PHP-Dokumentation, das Argument lässt sich damit passend zuordnen.

Aufgabe 2

Es enthält das aktuelle Datum und die aktuelle Uhrzeit der eingestellten Zeitzone:

```
$datetime = new DateTime();
echo $datetime->format('d.m.Y H:i:s');
// z.B. => 27.05.2022 09:03:32
```

Kapitel 11

Aufgabe 1

Falls ein MySQL-Server Zugriff von außen zulässt (z.B. bei einem Webhoster), ist eine Verbindung unter zusätzlicher Angabe des Hostnamens mit der Option `--host` möglich:

```
> mysql --host example.com --user Username --password
```

Aufgabe 2

Mit dem Suchbegriff »mysql truncate table« finden Sie die passende Stelle in der MySQL-Dokumentation. Der `TRUNCATE`-Befehl ist die effizienteste Art, *alle* Tabellendaten zu löschen:

```
TRUNCATE TABLE employee;
```

Im Unterschied zu `DELETE` besteht keine Möglichkeit, nur bestimmte Datensätze zu löschen. Die Tabellenstruktur bleibt intakt.

Kapitel 12

Aufgabe 1

```
SELECT * FROM movie ORDER BY length DESC LIMIT 5;
```

Aufgabe 2

```
SELECT NOW();
```

Aufgabe 3

```
SELECT COUNT(phone) AS '# mit Telefon' FROM user;
```

Kapitel 13

Aufgabe 1

`PDO::query()` liefert ein Objekt vom Typ `PDOStatement` zurück. Von diesem Objekt erhalten Sie mit der Methode `PDOStatement::fetchAll()` die abgefragten Datensätze:

```
$stmt = $pdo->query('SELECT * FROM user');  
$users = $stmt->fetchAll(); // Array mit Benutzer-Datensätzen
```

Bei einem Prepared Statement liefert die Methode `PDO::prepare()` ebenfalls ein Objekt vom Typ `PDOStatement` zurück. Die Datensätze erhalten Sie auch mit der Methode `PDOStatement::fetchAll()` – jedoch erst nach der expliziten Ausführung des SQL-Befehls mit der Methode `PDOStatement::execute()`:

```
$stmt = $pdo->prepare('SELECT * FROM user');  
$stmt->execute();  
$users = $stmt->fetchAll(); // Array mit Benutzer-Datensätzen
```

`PDO::query()` ist bequemer, es führt Vorbereitung und Ausführung in einem Schritt durch. Jedoch können keine Parameter per Platzhalter in den SQL-Befehl übergeben werden, was zum Schutz vor SQL-Injections wichtig ist.

Aufgabe 2

`PDOStatement::fetchAll()` liefert alle Ergebnisdatensätze in einem Array zurück, z.B.:

```
$users = $stmt->fetchAll(); // => [User1, User2]
```

`PDOStatement::fetch()` liefert nur den jeweils nächsten Datensatz der Ergebnismenge; am Ende `false`:

```
$user1 = $stmt->fetch(); // => User1
$user2 = $stmt->fetch(); // => User2
$user3 = $stmt->fetch(); // => false
```

Je nach Anwendungsfall kann die eine oder andere Variante nützlicher sein. Bei einem SQL-Befehl, der nur einen einzigen Datensatz liefern kann (z.B. beim Einsatz einer SQL-Aggregatfunktion, siehe Abschnitt 12.5.2), ist es bequemer, ein Array zu vermeiden und den Datensatz direkt mit `PDOStatement::fetch()` auszulesen.

Kapitel 14

Aufgabe 1

Gute Indikatoren sind eine weite Verbreitung, gute Dokumentation und regelmäßige Veröffentlichungen neuer Versionen. Aufschluss zur Verbreitung geben die Download-Anzahl und GitHub-Sterne. Auf Packagist finden Sie Links zu Dokumentationen bzw. GitHub, wo die meisten Projekte ihren Quellcode pflegen. Dort können Sie den Rhythmus neuer Veröffentlichungen einsehen. Achten Sie darauf, dass ein Paket aktiv weiterentwickelt wird.

Aufgabe 2

Die Validierung prüft, ob ...

- ... die gewählte Kategorie einen gültigen Wert aus dem Konfigurations-Array `BLOG_CATEGORIES` enthält.
- ... der Titel mindestens 5, der Inhalt mindestens 15 Zeichen enthält.

Validierungsfehler werden unter dem Feldnamen als Array-Schlüssel zurückgegeben, um Fehler einem Feld zuzuordnen.

Aufgabe 3

Eine Internet-Suche nach »PHP statische Methode« führt Sie schnell zur passenden Stelle in der PHP-Dokumentation:

<https://www.php.net/manual/language.oop5.static>

Eine statische Methode kann ohne Instanziierung eines Objekts aufgerufen werden – wie eine gewöhnliche Funktion. Da es kein Objekt gibt, hat der Methodenrumpf keinen Zugriff auf Objekt-Eigenschaften oder die Sondervariable `$this`. Im Vergleich zu einer normalen Funktion verleiht die Zugehörigkeit zu einer Klasse einen thematischen Kontext. Im Symfony Mailer wird `Transport::fromDns()` genutzt, um durch Analyse der übergebenen DSN-Zeichenkette ein passendes, spezialisiertes Transport-Objekt zu erzeugen und zurückzugeben, z.B. für den Versand via SMTP.