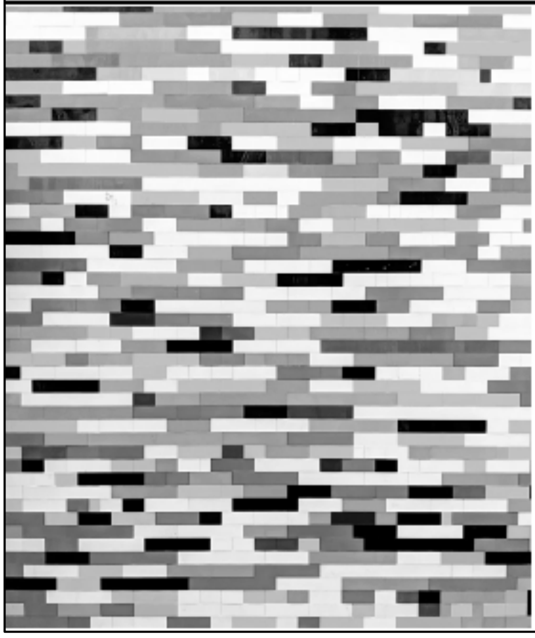
DEEPer

PHP & Libraries
Week 4 Session 1

x

TODAY'S SESSION



01 **Further PHP**

02 **PHP Libraries**

03 **PSR**

Committing & Pushing

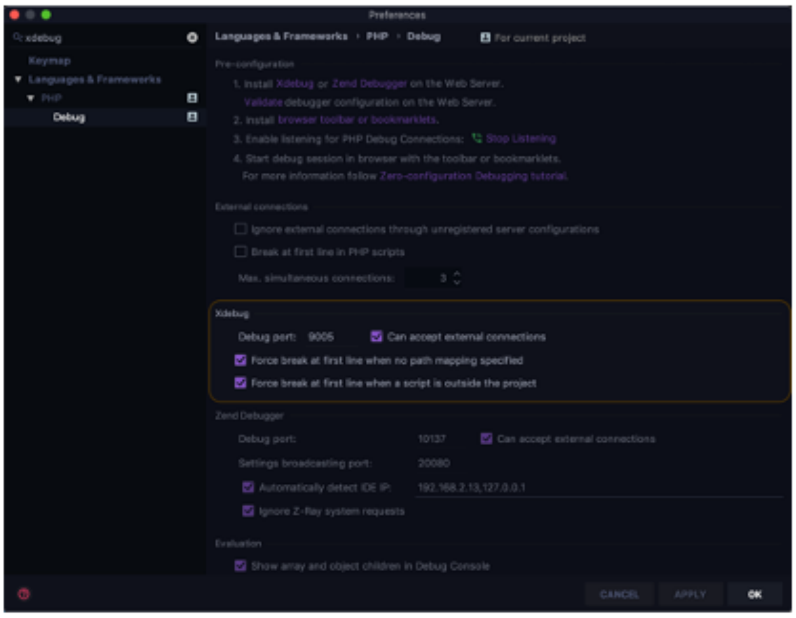
- It's important to push work!
- Pushing means we can see your code on BitBucket, and we can track your progress and help when you're stuck
- Think of it like a backup of your work – we can see a history of changes and easily revert if something breaks

```
git add . && git commit -m "Task complete" && git push
```

Setting Up XDebug

- Install browser extension:
  - https://www.jetbrains.com/help/phpstorm/browser-debugging-extensions.html
  - IDE Key: PHPSTORM

Setting Up XDebug



Pre-configuration

1. Install Xdebug or Zend Debugger on the Web Server.
   Validate debugger configuration on the Web Server.
2. Install browser toolbar or bookmarklets.
3. Enable listening for PHP Debug Connections: Stop Listening
4. Start debug session in browser with the toolbar or bookmarklets.
   For more information follow Zero-configuration Debugging tutorial.

External connections

- [ ] Ignore external connections through unregistered server configurations
- [ ] Break at first line in PHP scripts

Max. simultaneous connections: 3

Xdebug

Debug port: 9005 [✓] Can accept external connections
- [✓] Force break at first line when no path mapping specified
- [✓] Force break at first line when a script is outside the project

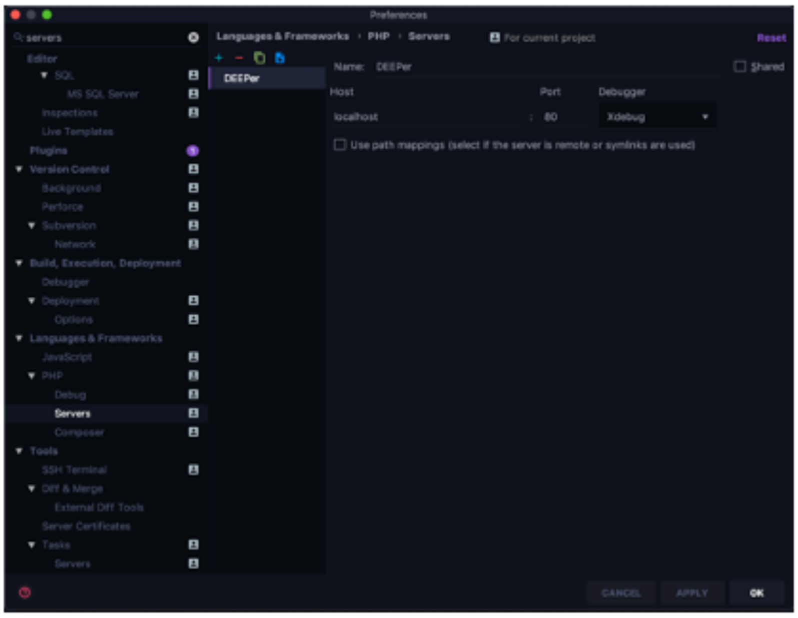
Zend Debugger

Debug port: 10137 [✓] Can accept external connections
Settings broadcasting port: 20080
- [✓] Automatically detect IDE IP: 192.168.2.13,127.0.0.1
- [✓] Ignore Z-Ray system requests

Evaluation

- [✓] Show array and object children in Debug Console

Setting Up XDebug

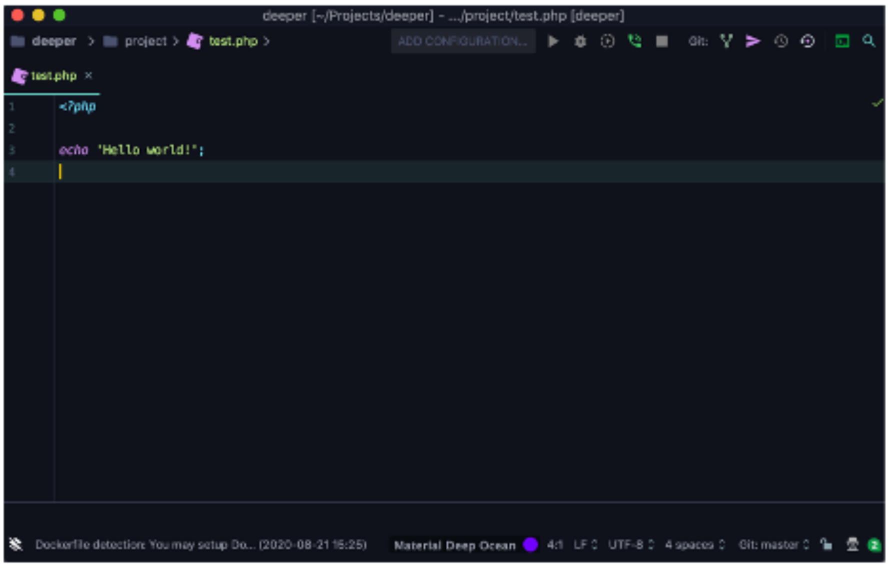


Name: DEEPer

Host	Port	Debugger
localhost	80	Xdebug

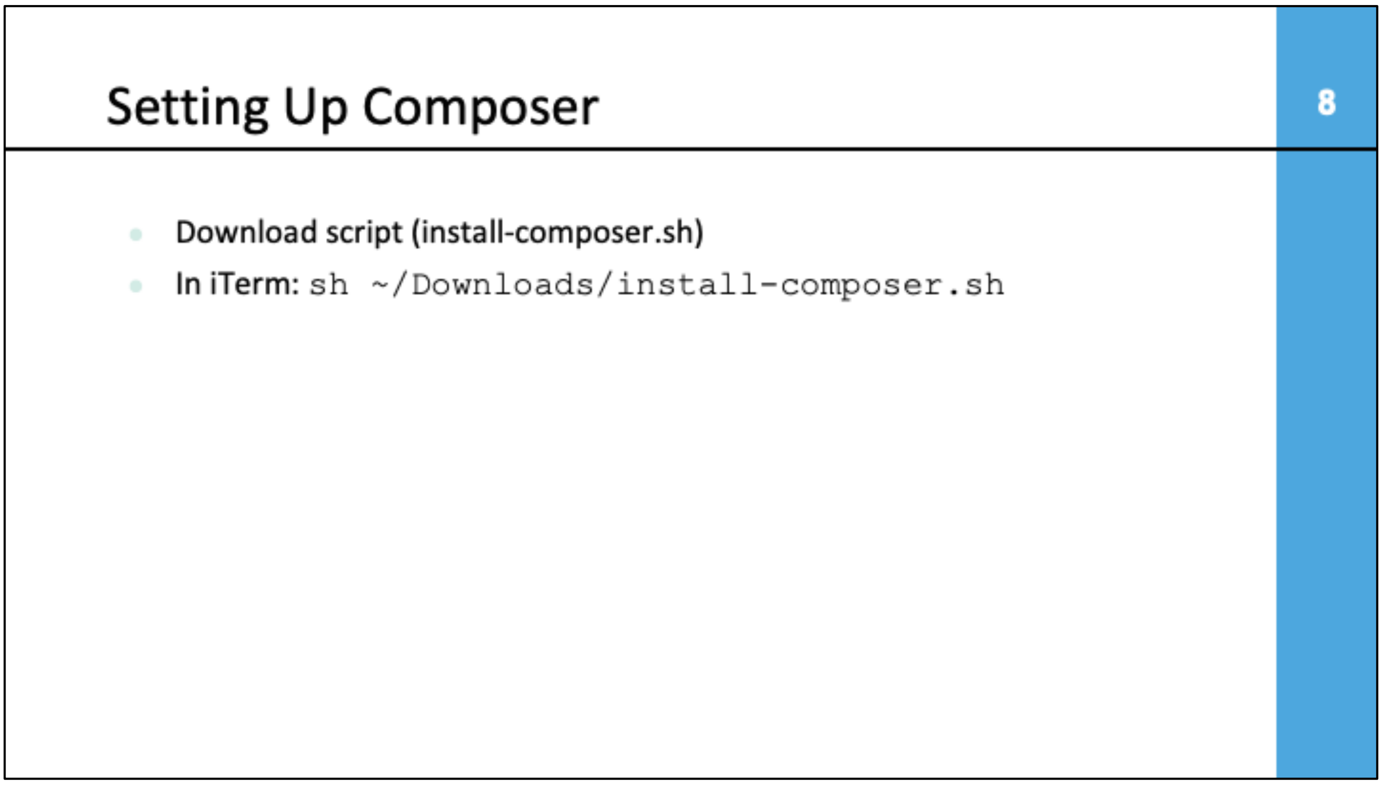
☐ Use path mappings (select if the server is remote or symlinks are used)

Setting Up XDebug


deeper [~/Projects/deeper] - .../project/test.php [deeper]
deeper > project > test.php >
ADD CONFIGURATION...
Git:
test.php ×

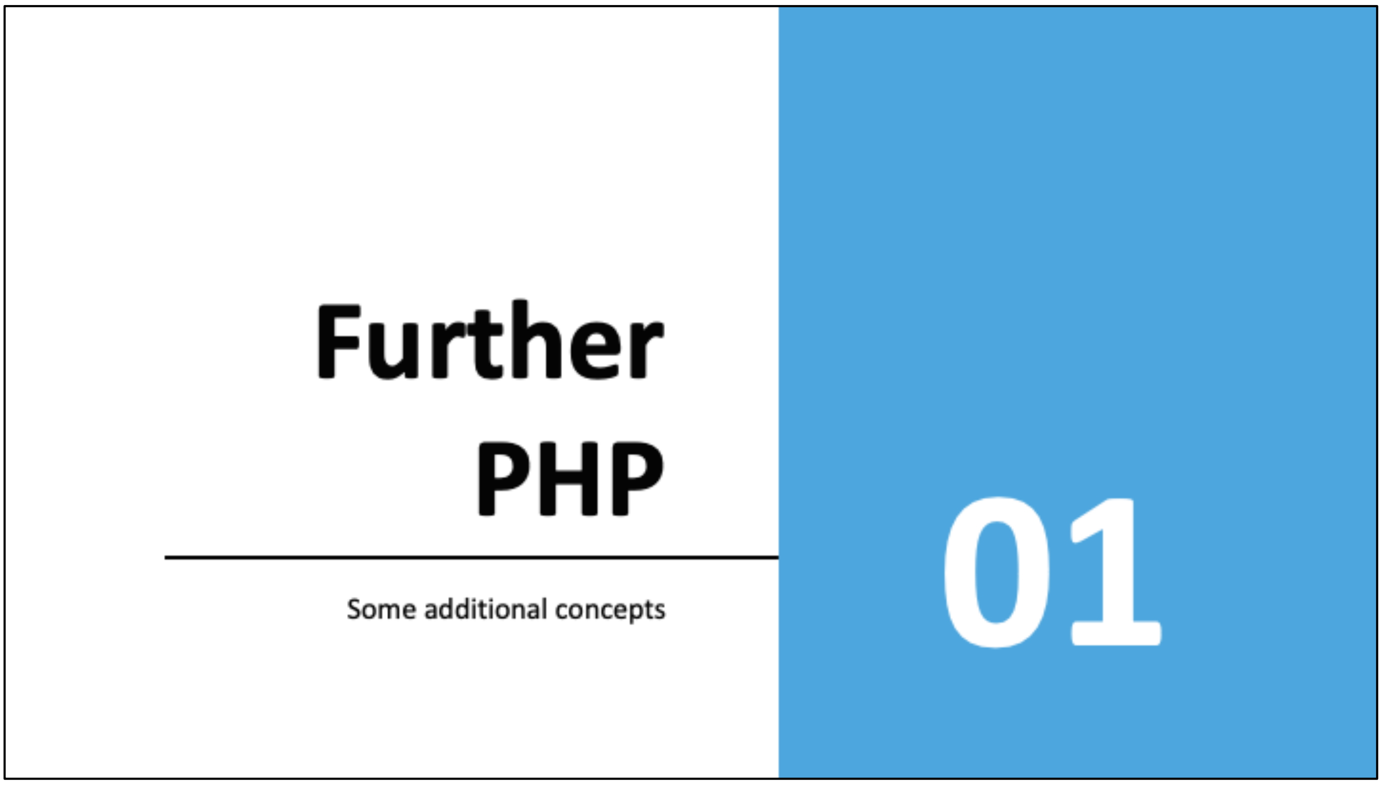
```php
<?php

echo 'Hello world!";
```

Dockerfile detection: You may setup Do... (2020-08-21 16:25)
Material Deep Ocean
4:1
LF
UTF-8
4 spaces
Git: master



Setting Up Composer

- Download script (install-composer.sh)
- In iTerm: `sh ~/Downloads/install-composer.sh`



Further PHP

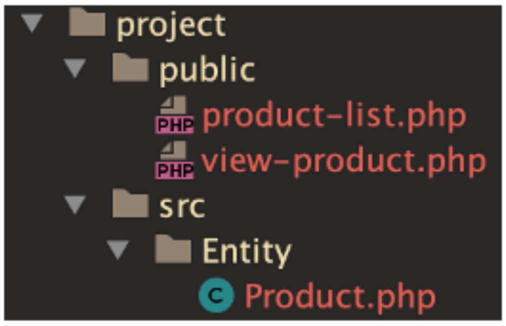
Some additional concepts

01

Structuring Projects

- So far in the DEEPer course project we have included files in the root of the project folder
- All files within this folder are accessible via the browser
- As projects grow however there are some files we do not want to expose directly to users
- This is achieved by creating a directory for the publicly accessible files
- The web server is then configured to only serve from this directory
- Files within this directory can still include files from others

Structuring Projects



```
▼ project
  ▼ public
      product-list.php
      view-product.php
  ▼ src
    ▼ Entity
        Product.php
```

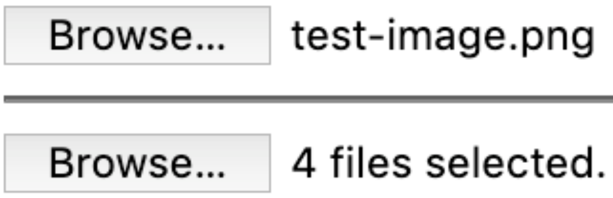
- **public** – the directory of files accessible by the public
- **src** – all application code separated into contextual subdirectories

File Uploads

- We have learned how to create and write data to files using PHP
- PHP also allows us to upload regular files submitted via the browser
- There are many different methods in the UI of allowing a user to select a file
- The simplest is an `<input/>` with `type="file"`

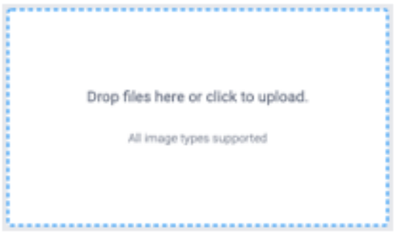
File Uploads

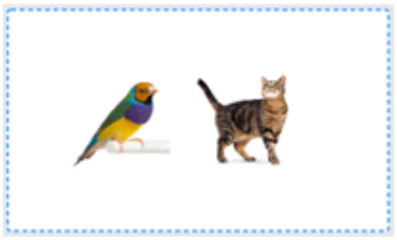
```
<input type="file" name="myFile" id="my-file">
<hr>
<input type="file" name="myFile" id="my-file" multiple>
```

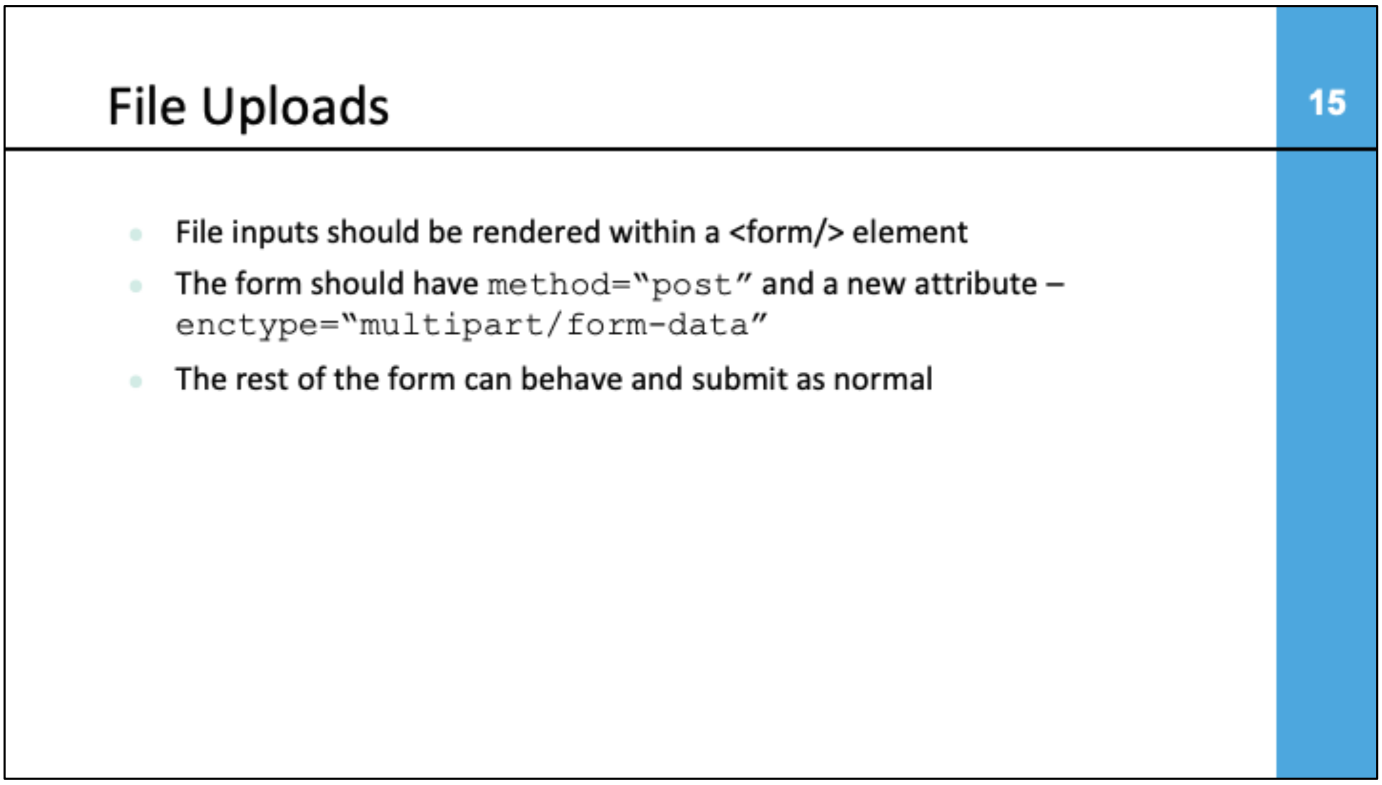


File Uploads

- More user-friendly interface components are available through the use of JavaScript
- One example of many is DropzoneJS, but there are **many**


Drop files here or click to upload.
All image types supported

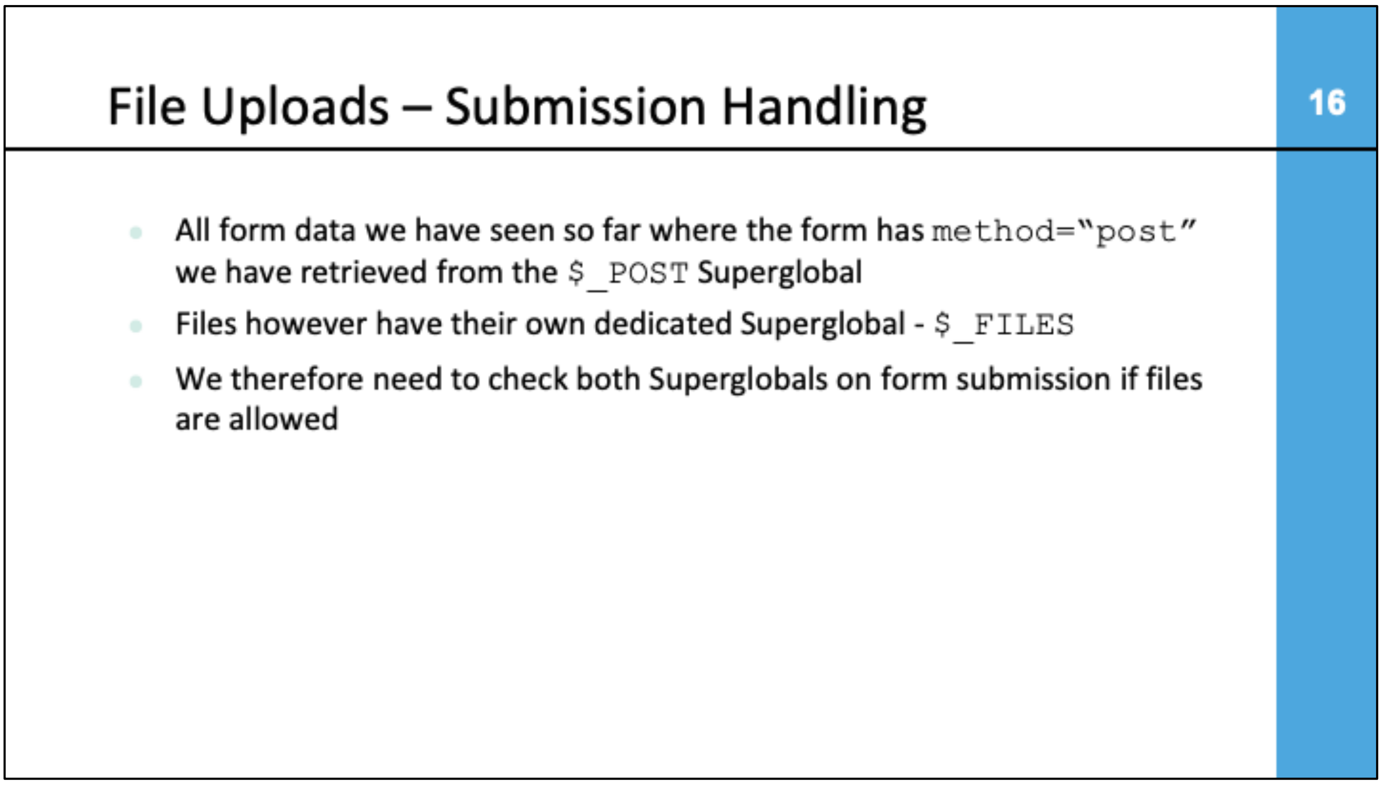




File Uploads

- File inputs should be rendered within a <form/> element
- The form should have `method="post"` and a new attribute – `enctype="multipart/form-data"`
- The rest of the form can behave and submit as normal

- CA Next

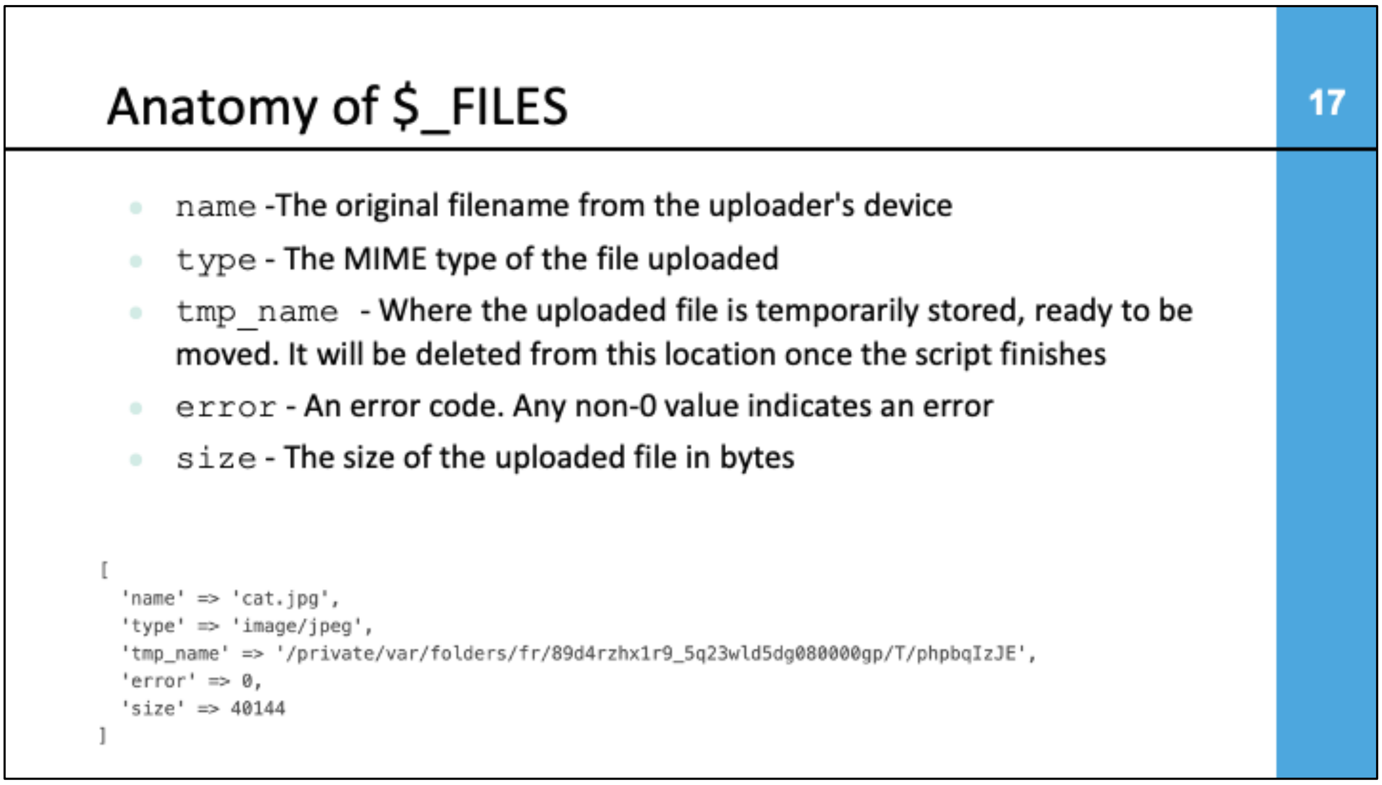


File Uploads – Submission Handling

16

- All form data we have seen so far where the form has `method="post"` we have retrieved from the `$_POST` Superglobal
- Files however have their own dedicated Superglobal - `$_FILES`
- We therefore need to check both Superglobals on form submission if files are allowed

- CA Next



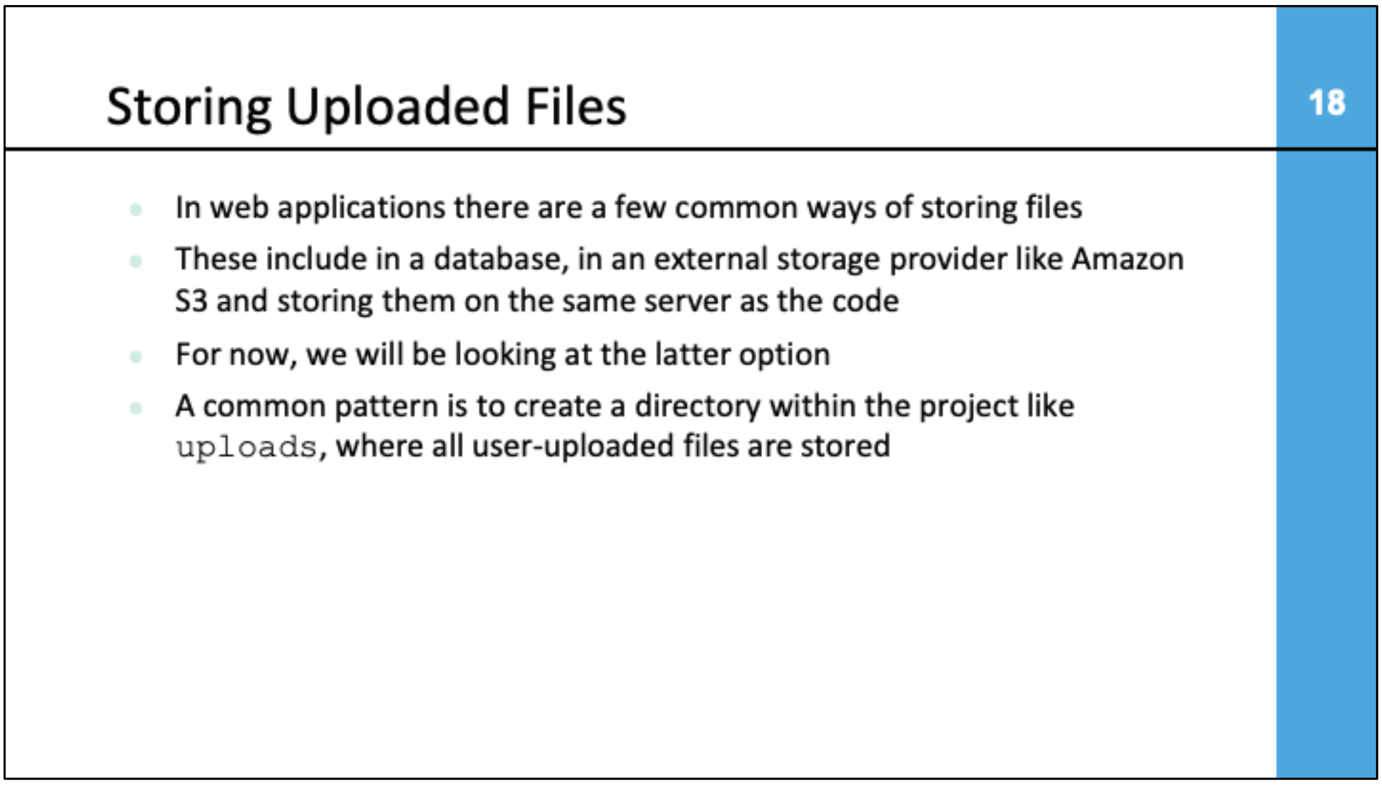
Anatomy of $_FILES

17

- `name` -The original filename from the uploader's device
- `type` - The MIME type of the file uploaded
- `tmp_name` - Where the uploaded file is temporarily stored, ready to be moved. It will be deleted from this location once the script finishes
- `error` - An error code. Any non-0 value indicates an error
- `size` - The size of the uploaded file in bytes

```
[
  'name' => 'cat.jpg',
  'type' => 'image/jpeg',
  'tmp_name' => '/private/var/folders/fr/89d4rzhx1r9_5q23wld5dg080000gp/T/phpbqIzJE',
  'error' => 0,
  'size' => 40144
]
```

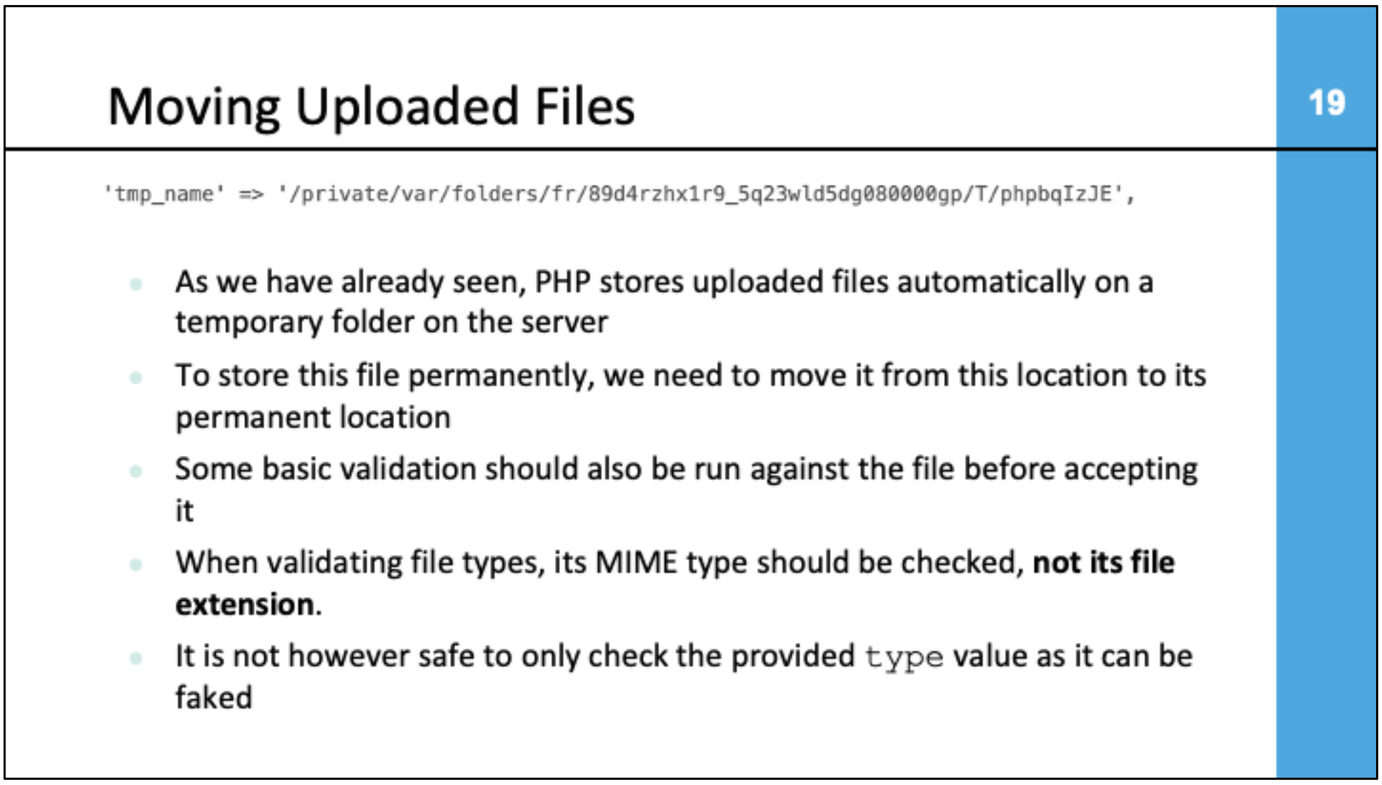
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types
- https://www.php.net/manual/en/features.file-upload.errors.php



Storing Uploaded Files

- In web applications there are a few common ways of storing files
- These include in a database, in an external storage provider like Amazon S3 and storing them on the same server as the code
- For now, we will be looking at the latter option
- A common pattern is to create a directory within the project like `uploads`, where all user-uploaded files are stored

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types
- https://www.php.net/manual/en/features.file-upload.errors.php



Moving Uploaded Files

```
'tmp_name' => '/private/var/folders/fr/89d4rzhx1r9_5q23wld5dg080000gp/T/phpbqIzJE',
```

- As we have already seen, PHP stores uploaded files automatically on a temporary folder on the server
- To store this file permanently, we need to move it from this location to its permanent location
- Some basic validation should also be run against the file before accepting it
- When validating file types, its MIME type should be checked, **not its file extension**.
- It is not however safe to only check the provided `type` value as it can be faked

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types
- https://www.php.net/manual/en/features.file-upload.errors.php

- You can think of the concept of temporary files and moving them like downloading a file to Downloads, then moving it somewhere else
- CA Next

Logging

- Logging is the process of storing useful events and system data somewhere for later review
- Typically logs may be written to a dedicated .log file, with one entry per line
- Common things to log include;
  - Debug data like inbound requests
  - Important system events like user authentication, authentication failure, admin changes
  - Errors to assist with debugging, especially in production
- Sensitive data should not be logged to avoid security issues, or legal issues like GDPR

Logging – Basic Example

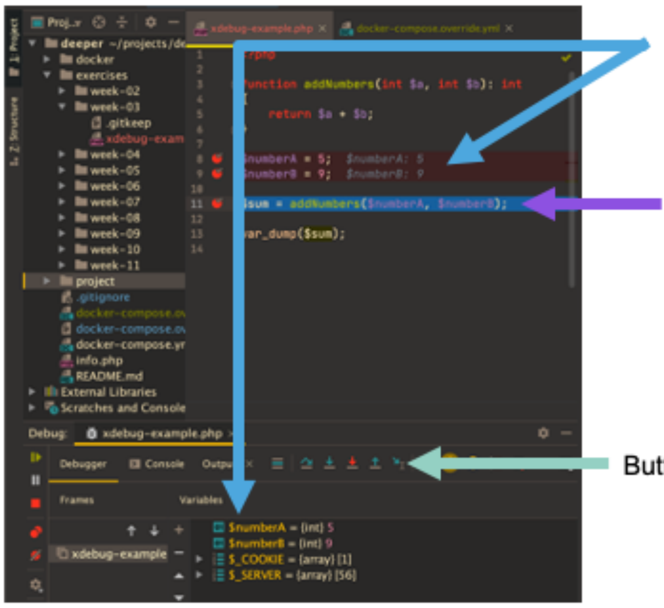
```php
<?php

if (!empty($_POST)) {
    $name = $_POST['name'];
    $email = $_POST['email'];

    $logMsg = time() . ' - NEW SUBSCRIPTION: ' . $name . ', email: ' . $email . PHP_EOL;
    file_put_contents('logfile.log', $logMsg, FILE_APPEND | LOCK_EX);

    // Continue with system logic
}
```

```
1598115073 - NEW SUBSCRIPTION: Joe Bloggs, email: joebloggs@email.com
1598115194 - NEW SUBSCRIPTION: Jane Doe, email: janedoe@email.com
```

Xdebug

- Xdebug is a PHP extension used in development to assist with debugging PHP code
- The main benefit is the ability to manually step through PHP code line-by-line and inspect the data
- Xdebug also upgrades the output from var_dump() to improve its formatting
- A profiler is also provided, which times how long individual components of a script take to measure performance

Xdebug



```
deeper ~/projects/de
  docker
  exercises
    week-02
    week-03
      .gitkeep
      xdebug-exam
    week-04
    week-05
    week-06
    week-07
    week-08
    week-09
    week-10
    week-11
  project
  .gitignore
  docker-compose.o
  docker-compose.o
  docker-compose.y
  info.php
  README.md
External Libraries
Scratches and Console
```

```php
<?php

function addNumbers(int $a, int $b): int
{
    return $a + $b;
}

$numberA = 5;  $numberA: 5
$numberB = 9;  $numberB: 9

$sum = addNumbers($numberA, $numberB);

var_dump($sum);
```

```
$numberA = {int} 5
$numberB = {int} 9
$_COOKIE = {array} [1]
$_SERVER = {array} [56]
```

Current variable states are available

The current line of execution is highlighted

Buttons to progress through the script

Xdebug

```
$array = [
    'key' => 'value',
    'anotherKey' => 'anotherValue',
    'intKey' => 123,
    'nestedArray' => ['orange', 'apple', 'blueberry'],
];

var_dump($array);
```

```
array (size=4)
  'key' => string 'value' (length=5)
  'anotherKey' => string 'anotherValue' (length=12)
  'intKey' => int 123
  'nestedArray' =>
    array (size=3)
      0 => string 'orange' (length=6)
      1 => string 'apple' (length=5)
      2 => string 'blueberry' (length=9)
```

Namespaces

- Allow us to create a collection or library of classes and functions
- Provide a way of organising code
- Reflects the directory structure
- Help to prevent conflicting class names
- The same class name can appear in multiple namespaces
- Namespaces can be nested
- Each level of nesting is observed by a backslash
- Classes can be imported from a namespace with a `use` command
- `use App\ExampleNamespace\MyClass;`

Namespaces

```php
<?php

use Path\To\BaseClass;
use Different\Path\To\BaseClass as AliasedBaseClass;

class MyClass extends BaseClass {}

class MyOtherClass extends AliasedBaseClass {}
```

cURL

- Stands for Client URL
- Allows us to retrieve data via numerous protocols, e.g.
  - HTTP
  - FTP
- Enabled via a PHP extension
- Interaction is similar to `fopen` / `fclose`, but with some additional options

cURL

- Key functions:
  - `curl_init` – initialises a cURL connection
  - `curl_setopt` – sets an option for the cURL resource
  - `curl_exec` – executes the cURL request
  - `curl_close` – closes the cURL resource

cURL

```
<?php

$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, 'https://api.jokes.one/jod');
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

$response = curl_exec($ch);

curl_close($ch);

$decodedResponse = json_decode($response);

$joke = $decodedResponse->contents->jokes[0]->joke->text;

echo nl2br($joke);
```

02

PHP Libraries

Why reinvent the wheel?

Why Libraries?

- There are a lot of common functions which others have already done a million times over
- Instead of reinventing the wheel every time we need to do something, there's often a library to do the heavy lifting for us
- Prime examples include logging and making API requests – which we'll be doing today!

Composer

- Composer is a package management utility for PHP libraries
- Enables easy installation (and updates!) of dependencies
- Allows us to define libraries and versions within our code
- Generates a centralised `autoload.php` file, which can be included to pull in all required packages
- Common commands:
  - `composer require vendor/package` – installs a package
  - `composer update` – updates packages
  - `composer install` – install all required packages
  - `composer remove vendor/package` – uninstalls a package

How to Choose a Library

- How many stars / watchers on GitHub?
- When was the last commit?
- Are there any reported issues which may affect what we want to do?
- How long has the project been alive – is it still in its infancy?
- Does the library *actually* do what we want it to do?
- What does the library offer on top of what we already have?

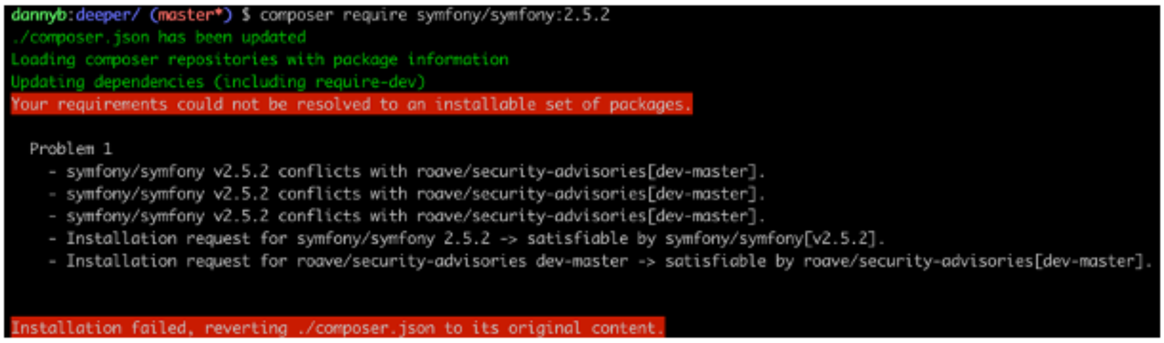
Today's Live Code Exercises

- Create the following folder:
- `~/projects/deeper/exercises/week-04/lecture/`
- All files will be situated in that directory
- All commands will be run within that directory, so before running anything, in Iterm, change directory:
- `cd ~/projects/deeper/exercises/week-04/lecture/`

Security Advisories

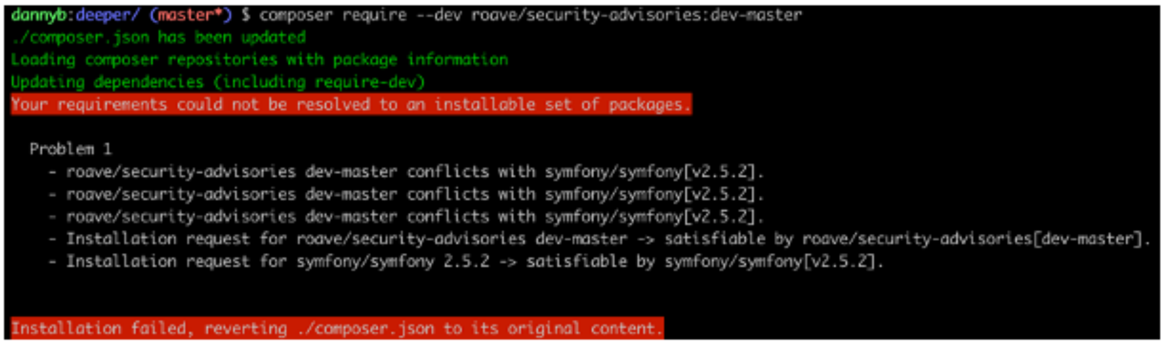
- Commonly, before we install a package, we want to know if there are any security vulnerabilities
- There's a convenient library which will scan any added library for security advisories
- `composer require --dev roave/security-advisories:dev-master`
- Any future attempts at installing a package which has known security vulnerabilities will be blocked

Security Advisories

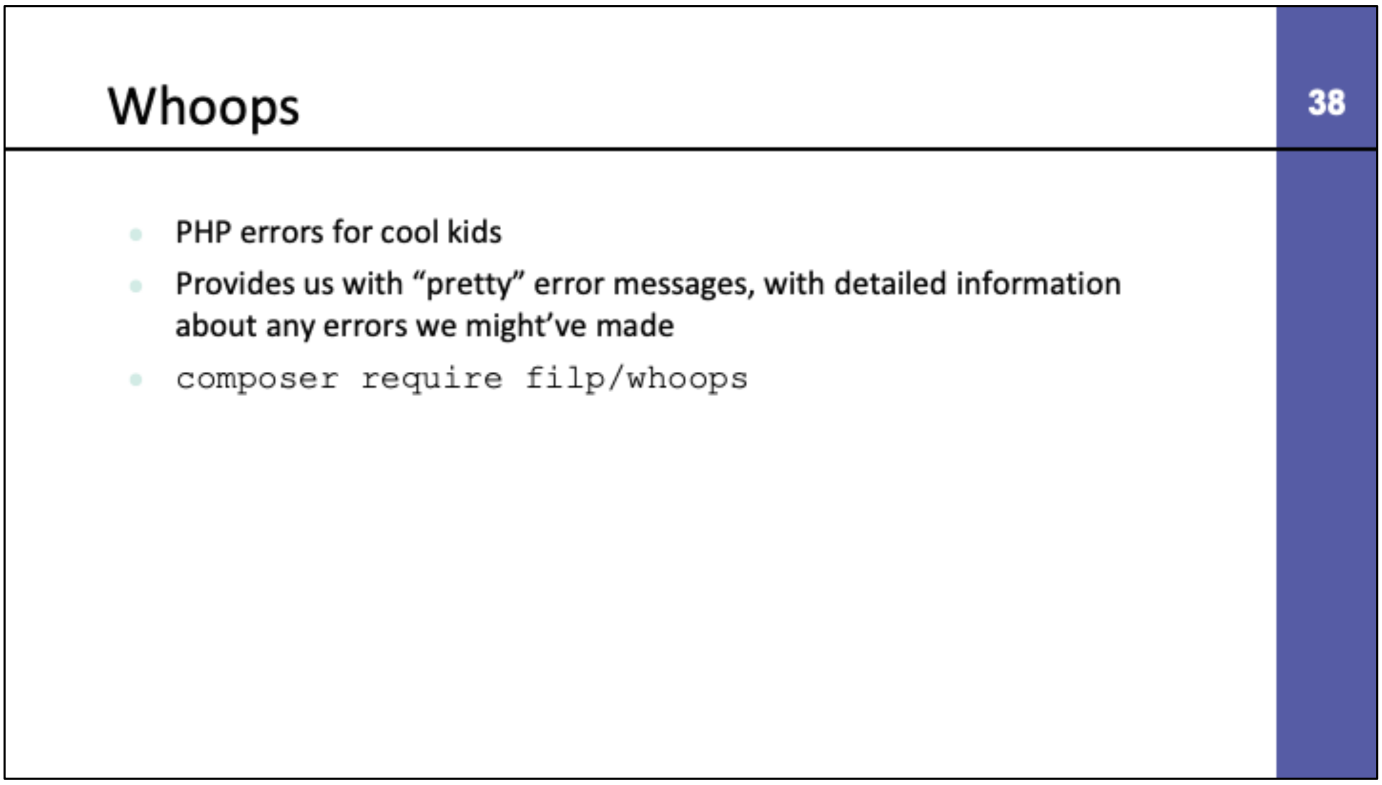


```
dannyb:deeper/ (master*) $ composer require symfony/symfony:2.5.2
./composer.json has been updated
Loading composer repositories with package information
Updating dependencies (including require-dev)
Your requirements could not be resolved to an installable set of packages.

  Problem 1
    - symfony/symfony v2.5.2 conflicts with roave/security-advisories[dev-master].
    - symfony/symfony v2.5.2 conflicts with roave/security-advisories[dev-master].
    - symfony/symfony v2.5.2 conflicts with roave/security-advisories[dev-master].
    - Installation request for symfony/symfony 2.5.2 -> satisfiable by symfony/symfony[v2.5.2].
    - Installation request for roave/security-advisories dev-master -> satisfiable by roave/security-advisories[dev-master].


Installation failed, reverting ./composer.json to its original content.
```

Security Advisories



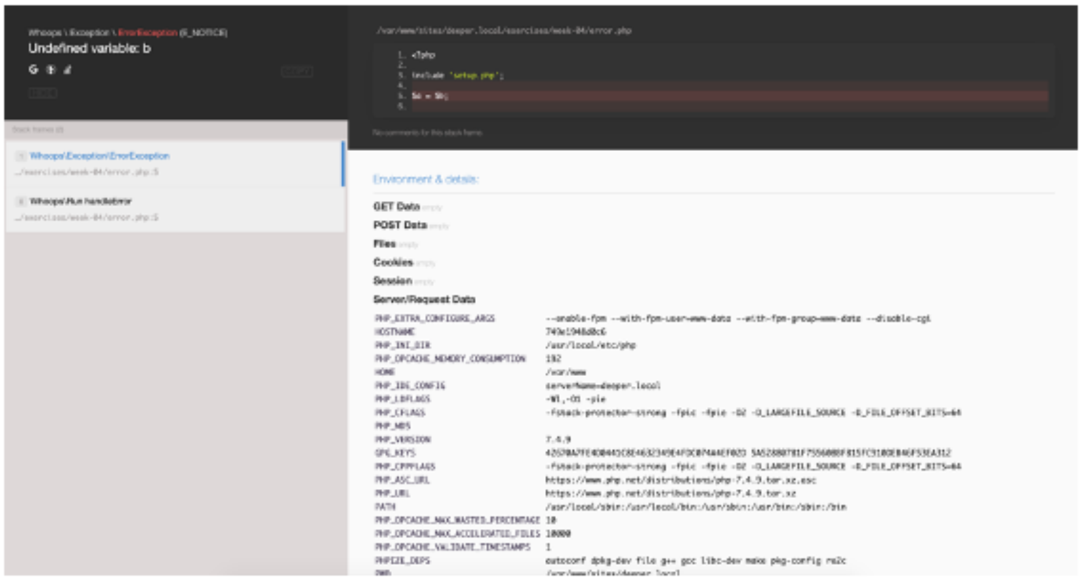
```
dannyb:deeper/ (master*) $ composer require --dev roave/security-advisories:dev-master
./composer.json has been updated
Loading composer repositories with package information
Updating dependencies (including require-dev)
Your requirements could not be resolved to an installable set of packages.

  Problem 1
    - roave/security-advisories dev-master conflicts with symfony/symfony[v2.5.2].
    - roave/security-advisories dev-master conflicts with symfony/symfony[v2.5.2].
    - roave/security-advisories dev-master conflicts with symfony/symfony[v2.5.2].
    - Installation request for roave/security-advisories dev-master -> satisfiable by roave/security-advisories[dev-master].
    - Installation request for symfony/symfony 2.5.2 -> satisfiable by symfony/symfony[v2.5.2].


Installation failed, reverting ./composer.json to its original content.
```


Whoops

- PHP errors for cool kids
- Provides us with “pretty” error messages, with detailed information about any errors we might’ve made
- `composer require filp/whoops`

CA Next

Whoops



Whoops \ Exception \ ErrorException (E_NOTICE)

Undefined variable: b

Stack frames (2)

Whoops\Exception\ErrorException
…/exercises/week-04/error.php:5

Whoops\Run handleError
…/exercises/week-04/error.php:5

/var/www/sites/deeper.local/exercises/week-04/error.php

```
<?php

include 'setup.php';

$a = $b;

```

No comments for this stack frame.

Environment & details:

GET Data empty

POST Data empty

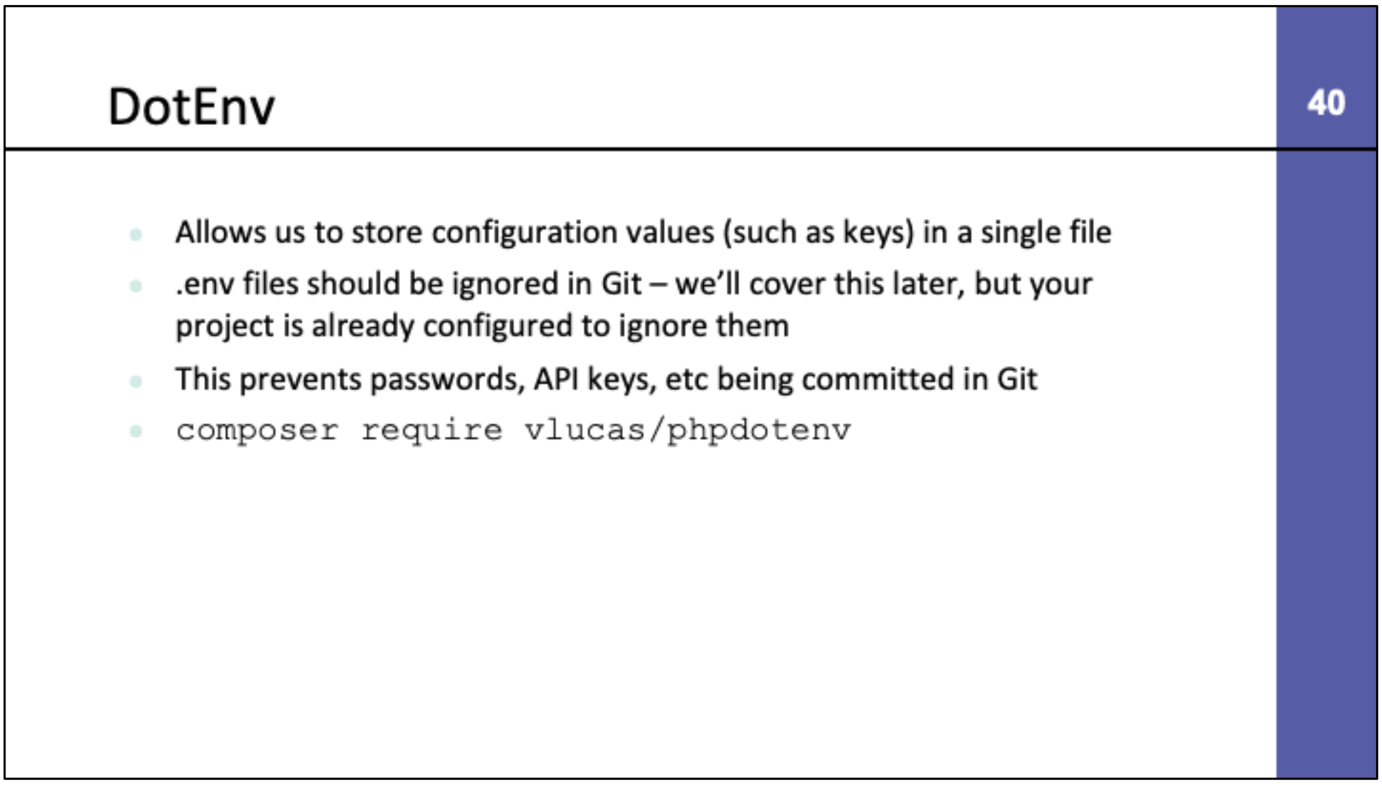
Files empty

Cookies empty

Session empty

Server/Request Data

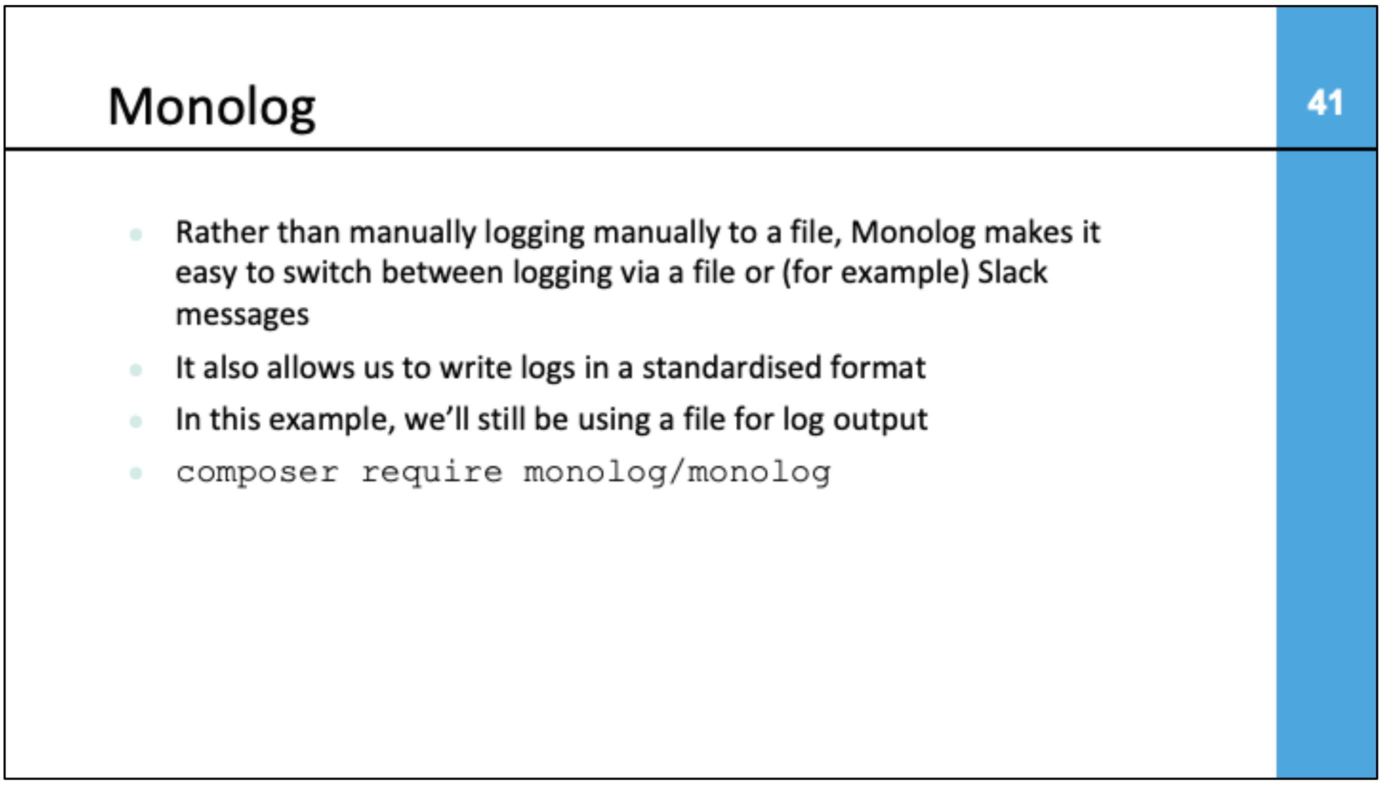
```
PHP_EXTRA_CONFIGURE_ARGS           --enable-fpm --with-fpm-user=www-data --with-fpm-group=www-data --disable-cgi
HOSTNAME                           740e1948d8c6
PHP_INI_DIR                        /usr/local/etc/php
PHP_OPCACHE_MEMORY_CONSUMPTION     192
HOME                               /var/www
PHP_IDE_CONFIG                     serverName=deeper.local
PHP_LDFLAGS                        -Wl,-O1 -pie
PHP_CFLAGS                         -fstack-protector-strong -fpic -fpie -O2 -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64
PHP_MD5
PHP_VERSION                        7.4.9
GPG_KEYS                           42670A7FE4D0441C8E4632349E4FDCBF44AEF023 5A52880781F755608BF815FC910DEB46F53EA312
PHP_CPPFLAGS                       -fstack-protector-strong -fpic -fpie -O2 -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64
PHP_ASC_URL                        https://www.php.net/distributions/php-7.4.9.tar.xz.asc
PHP_URL                            https://www.php.net/distributions/php-7.4.9.tar.xz
PATH                               /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
PHP_OPCACHE_MAX_WASTED_PERCENTAGE 10
PHP_OPCACHE_MAX_ACCELERATED_FILES 10000
PHP_OPCACHE_VALIDATE_TIMESTAMPS    1
PHPIZE_DEPS                        autoconf dpkg-dev file g++ gcc libc-dev make pkg-config re2c
PWD                                /var/www/sites/deeper.local
```



DotEnv

- Allows us to store configuration values (such as keys) in a single file
- .env files should be ignored in Git – we'll cover this later, but your project is already configured to ignore them
- This prevents passwords, API keys, etc being committed in Git
- `composer require vlucas/phpdotenv`

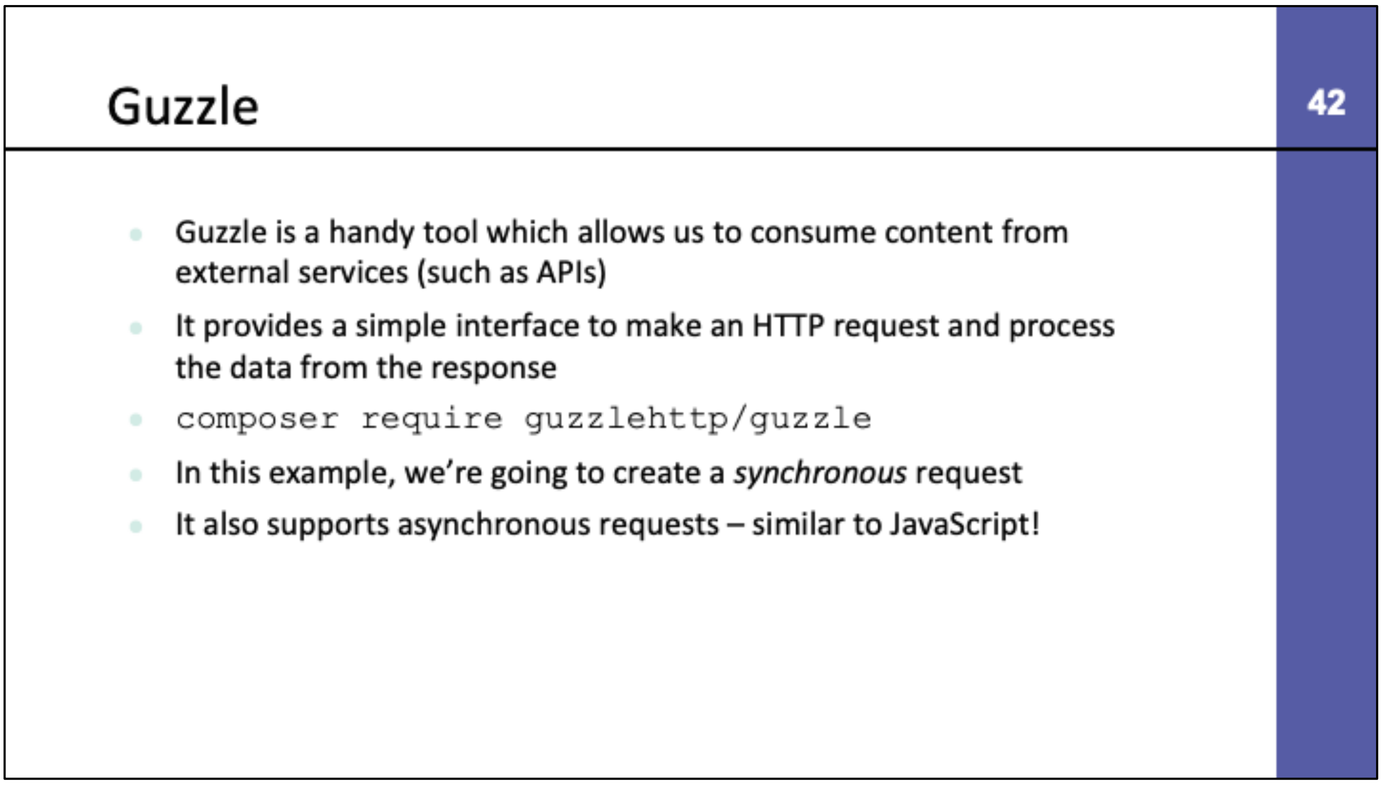
CA Next



Monolog

- Rather than manually logging manually to a file, Monolog makes it easy to switch between logging via a file or (for example) Slack messages
- It also allows us to write logs in a standardised format
- In this example, we'll still be using a file for log output
- `composer require monolog/monolog`

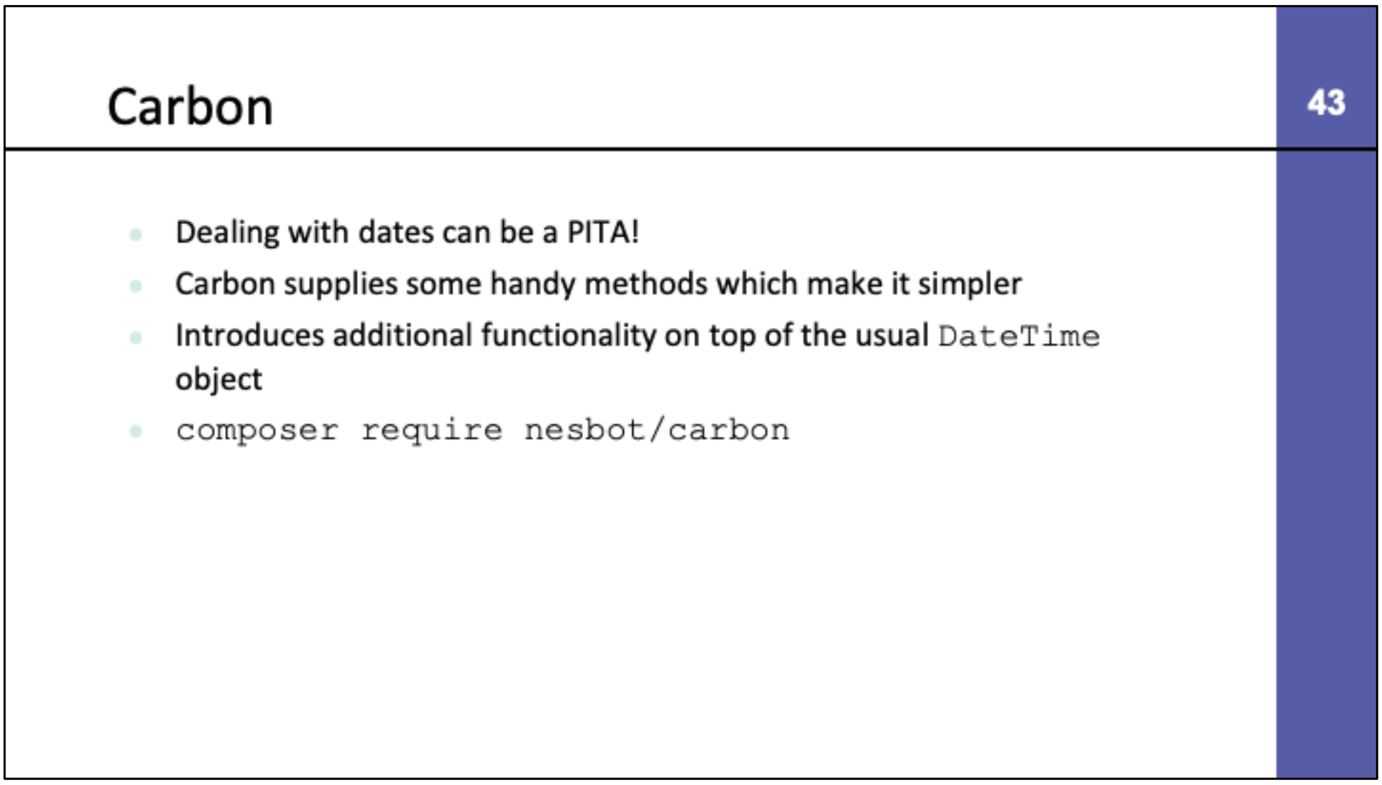
CA Next



Guzzle

- Guzzle is a handy tool which allows us to consume content from external services (such as APIs)
- It provides a simple interface to make an HTTP request and process the data from the response
- `composer require guzzlehttp/guzzle`
- In this example, we're going to create a *synchronous* request
- It also supports asynchronous requests – similar to JavaScript!

CA Next



Carbon

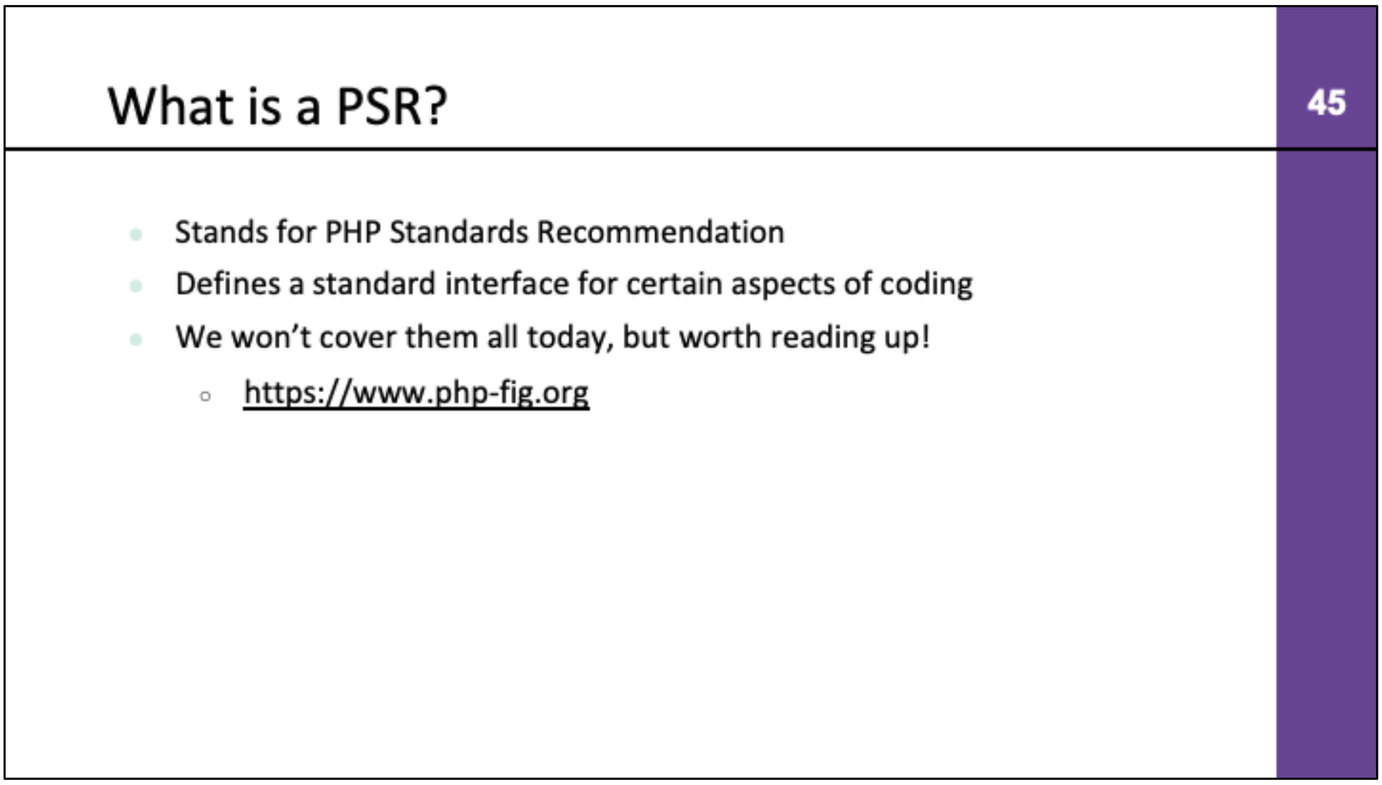
- Dealing with dates can be a PITA!
- Carbon supplies some handy methods which make it simpler
- Introduces additional functionality on top of the usual `DateTime` object
- `composer require nesbot/carbon`

CA Next

PSRs

Standards

03



What is a PSR?

- Stands for PHP Standards Recommendation
- Defines a standard interface for certain aspects of coding
- We won’t cover them all today, but worth reading up!
    - https://www.php-fig.org

- TODO: Find or make a diagram

PSR-12

- Defines a standard “style guide” for PHP
- Best practice which keeps code easily readable at a glance
- If only everyone conformed!
- https://www.php-fig.org/psr/psr-12/

PSR-12

```php
<?php

use Carbon\Carbon;
use Some\Namespace\With\A\Class as ImportedClass;

class MyClass extends ImportedClass
{
    public const SOME_CONSTANT = 10;

    public int $someProperty = 12;

    public function __construct(?int $someProperty)
    {
        if (!is_null($someProperty)) {
            $this->someProperty = $someProperty;
        }
    }

    public function output(string $message): void
    {
        echo $message;
    }
}
```

✓

PSR-12

```php
<?php

use Carbon\Carbon;
use Some\Namespace\With\A\Class as ImportedClass;

class MyClass extends ImportedClass
{
    public const SOME_CONSTANT = 10;

    public function __construct(?int $someProperty)
    {
        if ( !is_null( $someProperty ) ) {
              $this->someProperty = $someProperty;
        }
    }

    public function output(string $message): void
        {
            echo $message;
        }

    public int $someProperty = 12;
}
```