INFO1-CE9224: Introduction to PHP Programming

Session 7
August 1, 2012

Resources

http://davehauenstein.com/nyu/INFO1-CE9224-2012-Summer

Username: nyuscps
Password: $nyuscps$

Class 7 Agenda

- Complex Validation using Regular Expressions
    - PHP’s Regular Expression Functions
    - Writing Simple Regular Expressions
- File Handling
    - Reading from Files
    - Writing to Files
    - Uploading Files

Regular Expressions

What are Regular Expressions?

- A pattern describing a certain amount of text.
- Can be used for finding entities (ex: email addresses, a link w/in a web page, etc...) within a large amount of text.
- Can be used for validating user input.
- Can be used for finding and replacing patterns within a large amount of text.

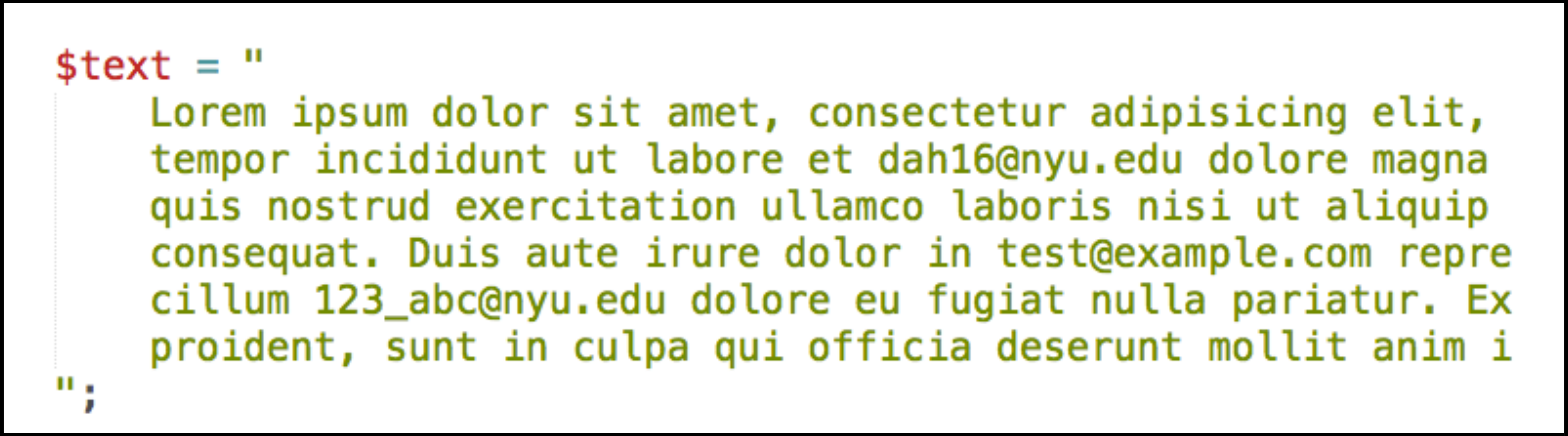
When to use Regex

- Finding URLs within text and replacing them with links.
- Screen scraping HTML and pulling out relevant data (ex: images, links, etc...).
- Validating that a username and/or password fit your acceptance criteria.
- Finding words with different spellings (grey, gray) using only one line of code.

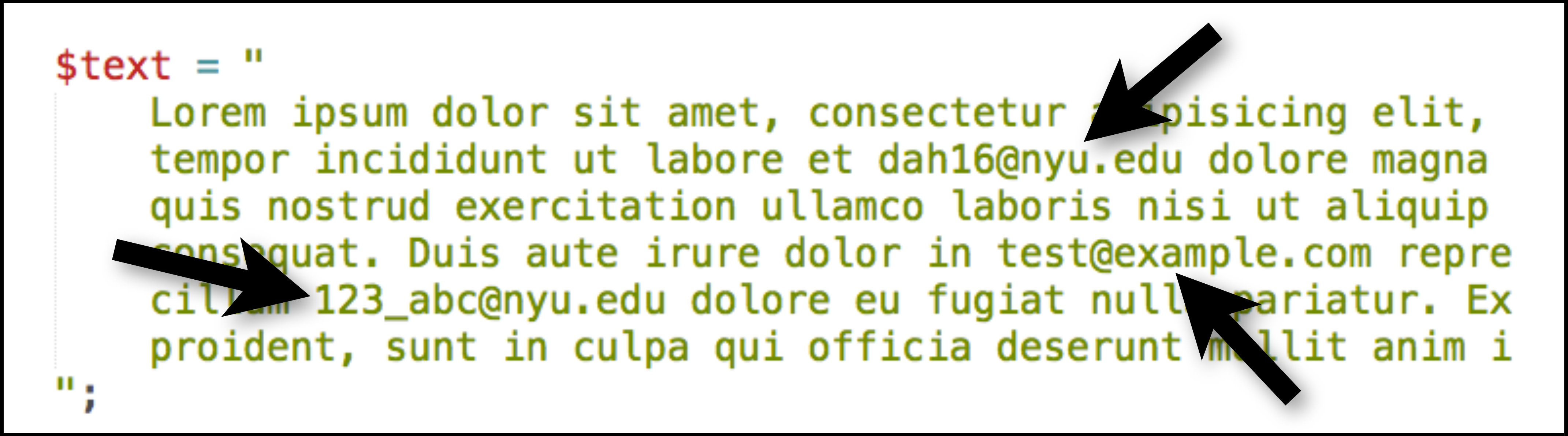
When to use Regex

Think of twitter and writing a tweet...

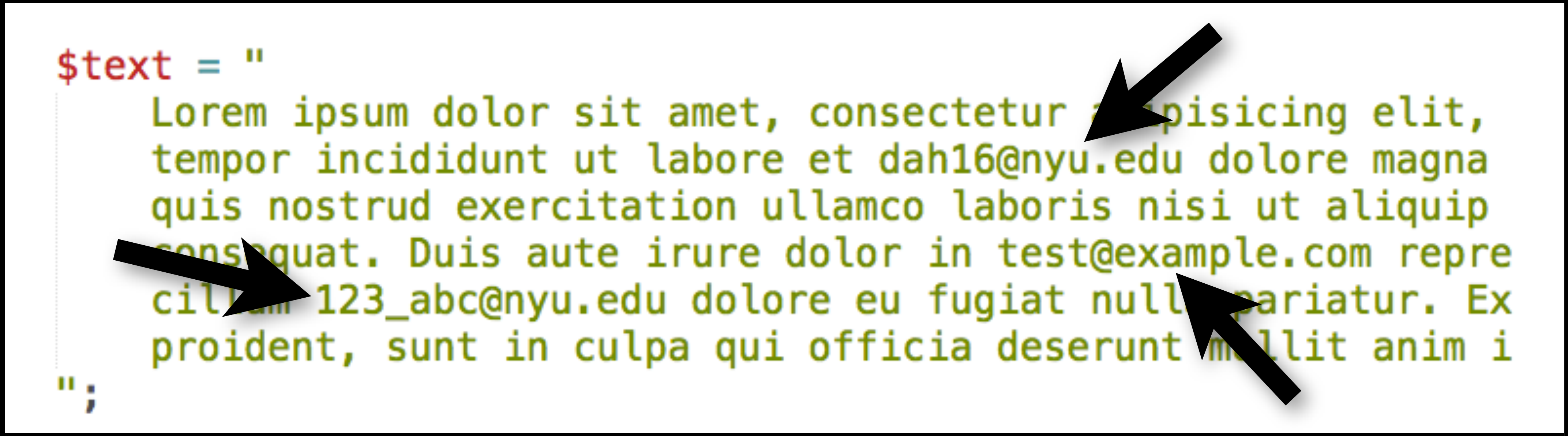
- hash tags are automatically found and linked to a search for other tweets with those hash tags.
- @username are automatically found and linked to a user’s account page.



```
$text = "
    Lorem ipsum dolor sit amet, consectetur adipisicing elit,
    tempor incididunt ut labore et dah16@nyu.edu dolore magna
    quis nostrud exercitation ullamco laboris nisi ut aliquip
    consequat. Duis aute irure dolor in test@example.com repre
    cillum 123_abc@nyu.edu dolore eu fugiat nulla pariatur. Ex
    proident, sunt in culpa qui officia deserunt mollit anim i
";
```



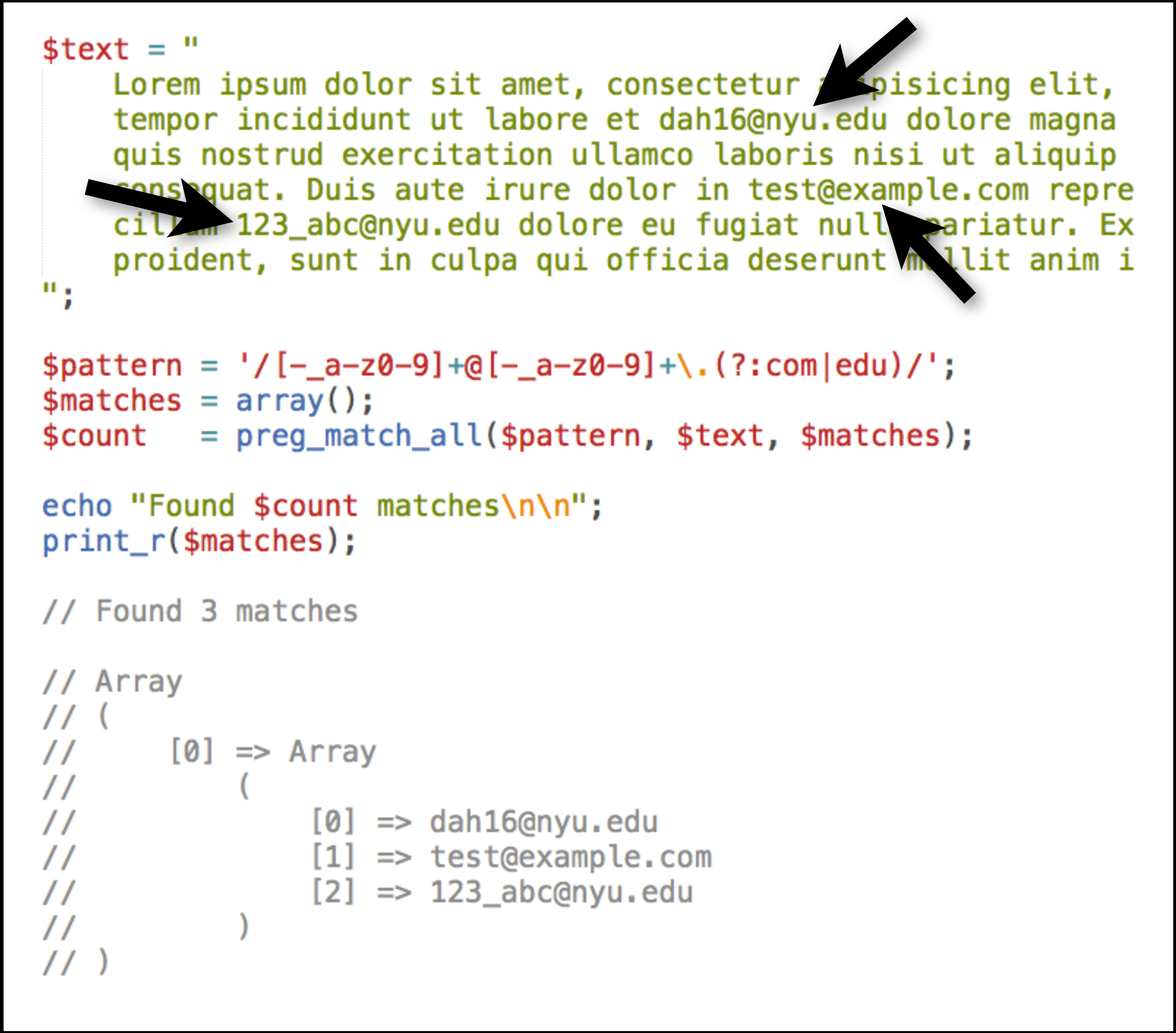
```
$text = "
    Lorem ipsum dolor sit amet, consectetur adipisicing elit,
    tempor incididunt ut labore et dah16@nyu.edu dolore magna
    quis nostrud exercitation ullamco laboris nisi ut aliquip
    consequat. Duis aute irure dolor in test@example.com repre
    cillum 123_abc@nyu.edu dolore eu fugiat nulla pariatur. Ex
    proident, sunt in culpa qui officia deserunt mollit anim i
";
```



```
$text = "
    Lorem ipsum dolor sit amet, consectetur adipisicing elit,
    tempor incididunt ut labore et dah16@nyu.edu dolore magna
    quis nostrud exercitation ullamco laboris nisi ut aliquip
    consequat. Duis aute irure dolor in test@example.com repre
    cillum 123_abc@nyu.edu dolore eu fugiat nulla pariatur. Ex
    proident, sunt in culpa qui officia deserunt mollit anim i
";
```

This text can come from anywhere...

- A website we've screen scraped.
- User input.
- Data from a database.
- Data from a web service.

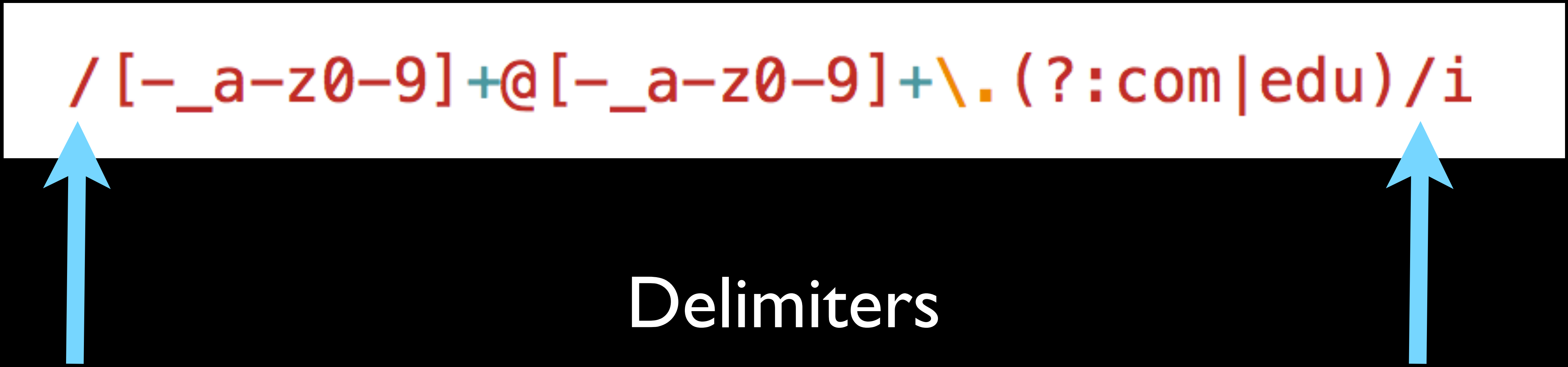


```
$text = "
    Lorem ipsum dolor sit amet, consectetur adipisicing elit,
    tempor incididunt ut labore et dah16@nyu.edu dolore magna
    quis nostrud exercitation ullamco laboris nisi ut aliquip
    consequat. Duis aute irure dolor in test@example.com repre
    cillum 123_abc@nyu.edu dolore eu fugiat nulla pariatur. Ex
    proident, sunt in culpa qui officia deserunt mollit anim i
";

$pattern = '/[-_a-z0-9]+@[-_a-z0-9]+\.(?:com|edu)/';
$matches = array();
$count   = preg_match_all($pattern, $text, $matches);

echo "Found $count matches\n\n";
print_r($matches);

// Found 3 matches

// Array
// (
//     [0] => Array
//         (
//             [0] => dah16@nyu.edu
//             [1] => test@example.com
//             [2] => 123_abc@nyu.edu
//         )
// )
```

Regex Engines

- PCRE RegEx (preg_*) - Perl Compatible Regular Expressions
- POSIX RegEx (ereg_*) - Portable Operating System Interface
- Different Syntax, PCRE is faster, more powerful, and more common.
- As of PHP 5.3 POSIX is deprecated.

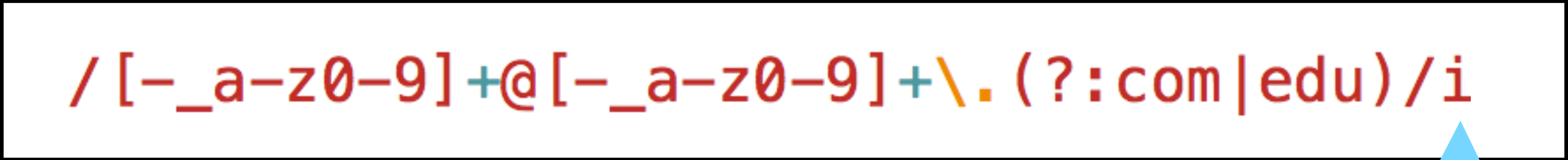
Regex Syntax



```
/[-_a-z0-9]+@[-_a-z0-9]+\.(?:com|edu)/i
```

Delimiters

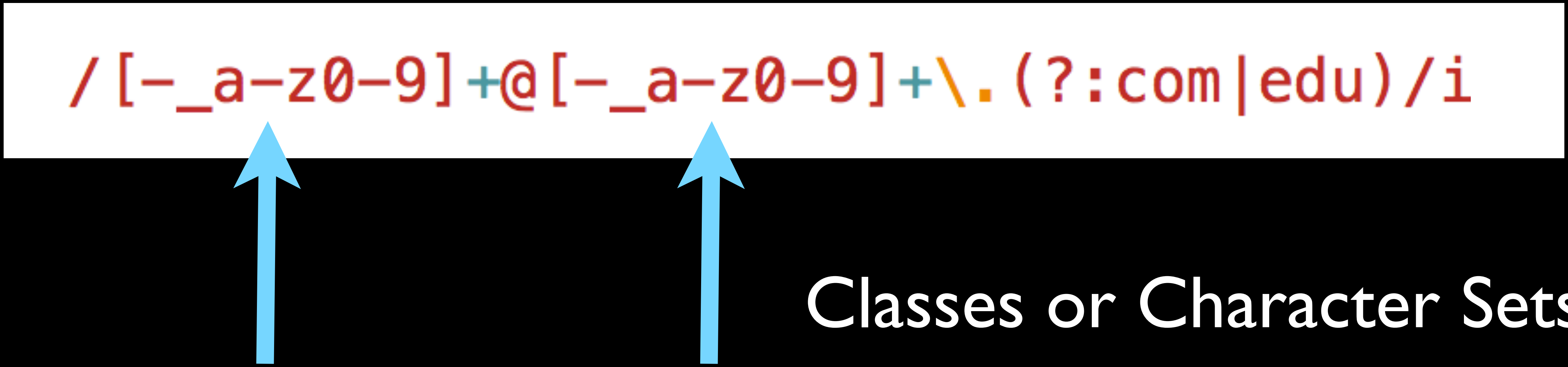
Regex Syntax



```
/[-_a-z0-9]+@[-_a-z0-9]+\.(?:com|edu)/i
```

Modifiers

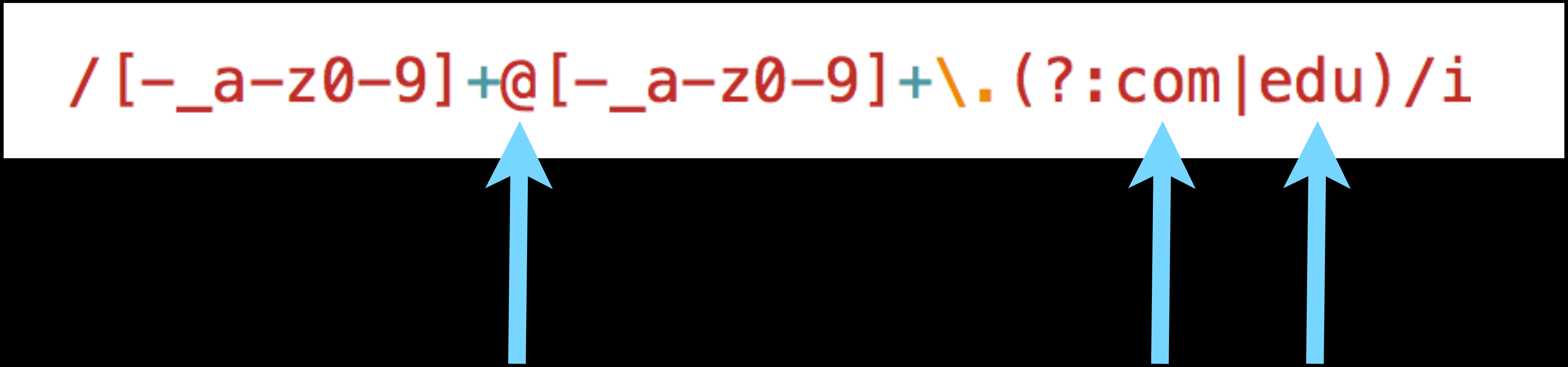
Regex Syntax



```
/[-_a-z0-9]+@[-_a-z0-9]+\.(?:com|edu)/i
```

Classes or Character Sets

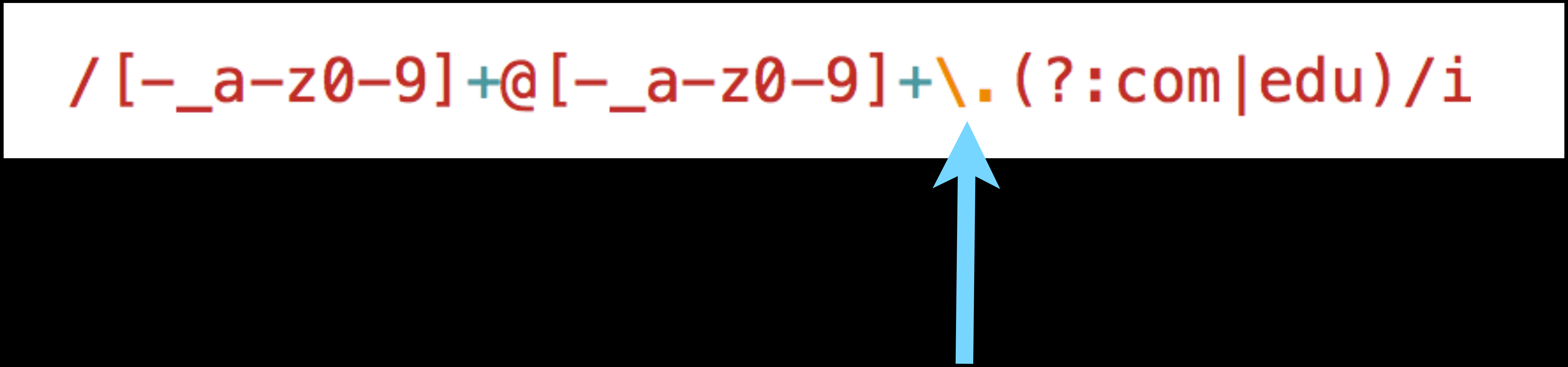
Regex Syntax



```
/[-_a-z0-9]+@[-_a-z0-9]+\.(?:com|edu)/i
```

@, com, edu are Literals

Regex Syntax

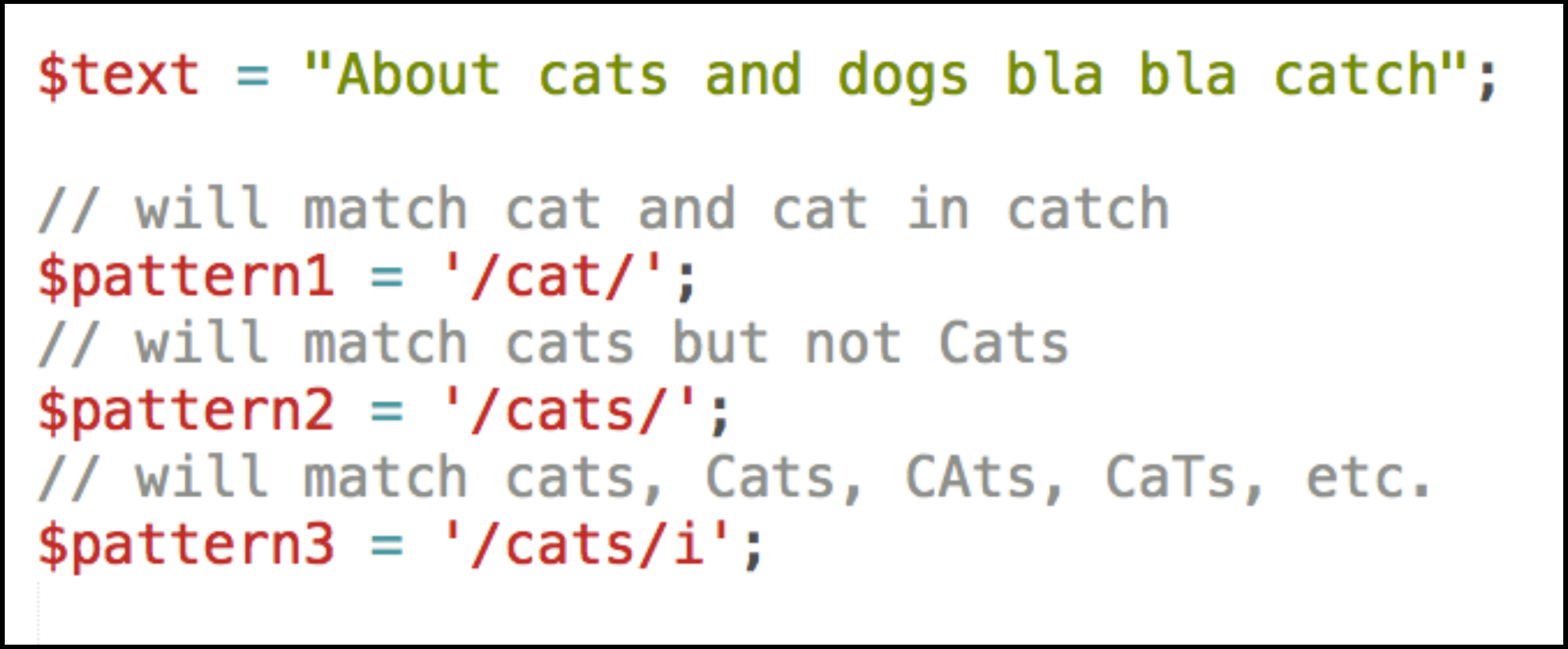


```
/[-_a-z0-9]+@[-_a-z0-9]+\.(?:com|edu)/i
```

Backslash used for escaping the special character, period

Regex Syntax

Literal Characters



```
$text = "About cats and dogs bla bla catch";

// will match cat and cat in catch
$pattern1 = '/cat/';
// will match cats but not Cats
$pattern2 = '/cats/';
// will match cats, Cats, CAts, CaTs, etc.
$pattern3 = '/cats/i';
```

Regex Syntax

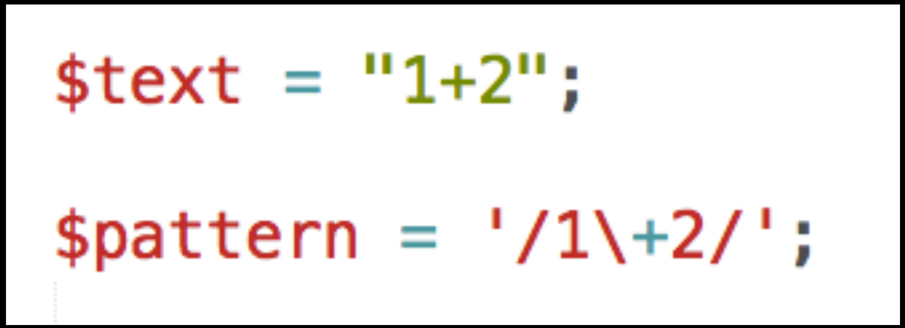
Special Characters

- square brackets [and]
- round brackets (and)
- curley braces { and }
- the backslash \
- the caret ^
- the dollar sign $
- the period or dot .
- the pipe symbol |
- the question mark ?
- the asterisk or star *
- the plus sign +

Regex Syntax

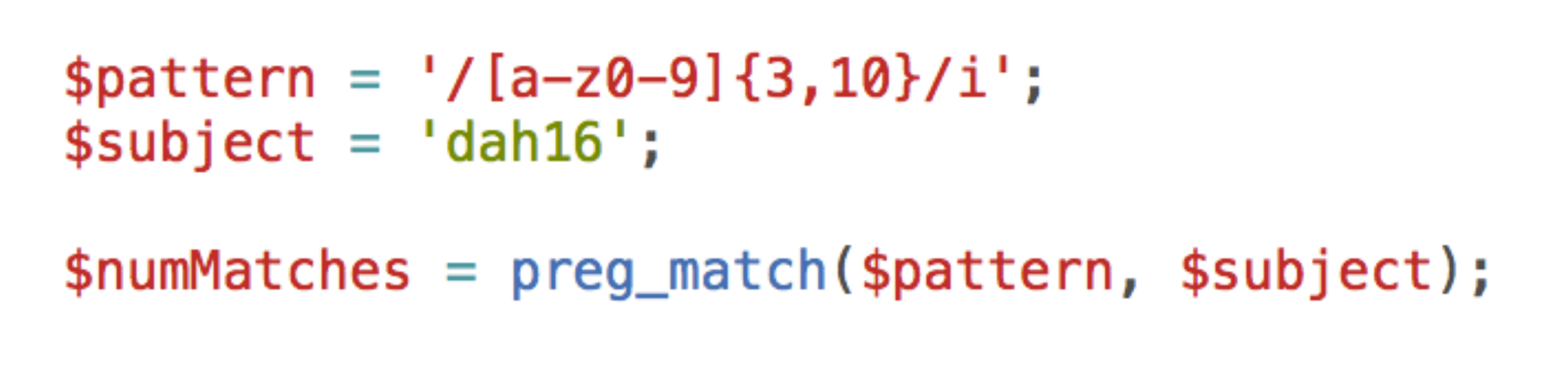
Special Characters

While trying to match the literal version of one of the special characters, they must be escaped using the backslash.



```
$text = "1+2";

$pattern = '/1\+2/';
```

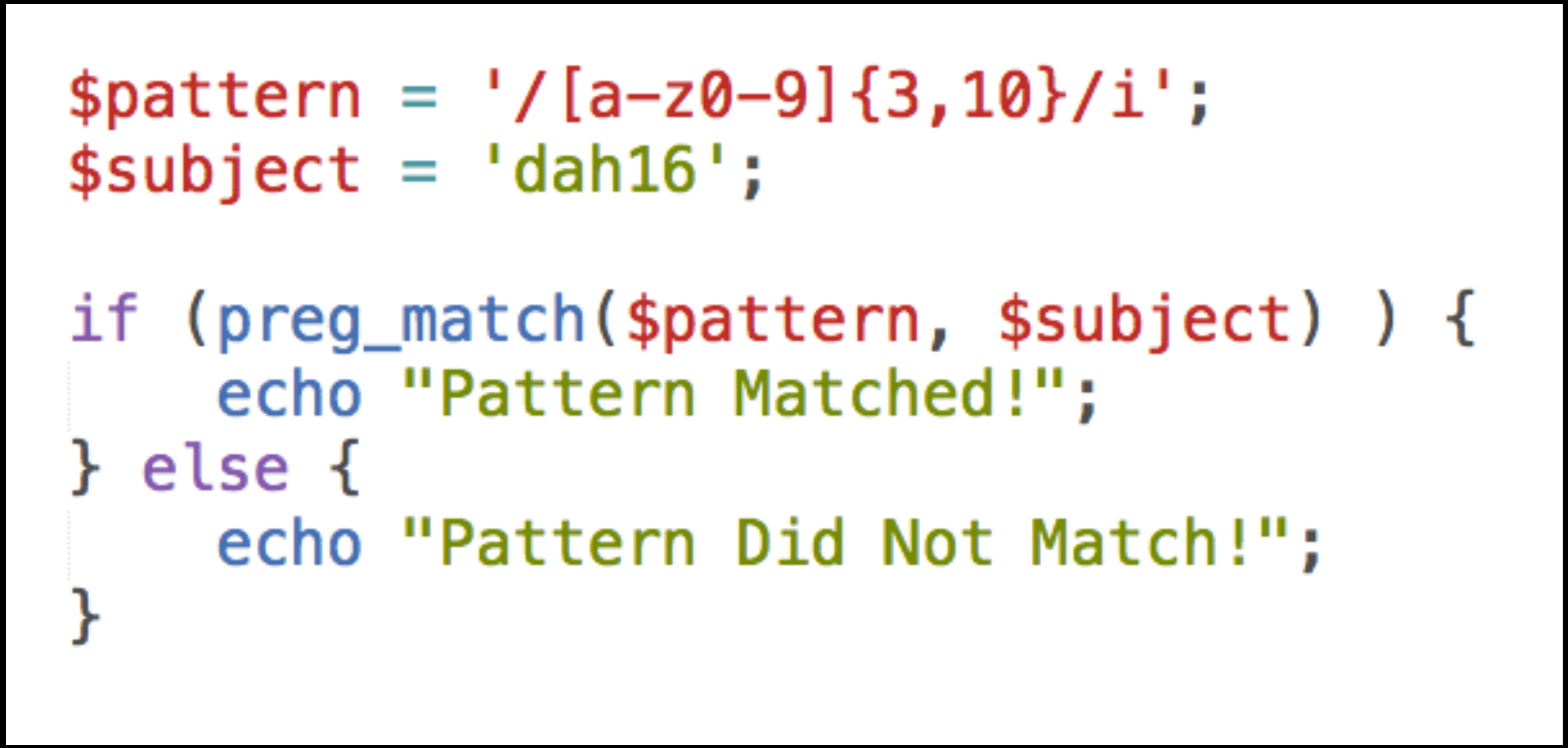
Regex for Validation



```
$pattern = '/[a-z0-9]{3,10}/i';
$subject = 'dah16';

$numMatches = preg_match($pattern, $subject);
```

- preg_match returns number of times pattern was matched.
- preg_match only matches the first occurrence of the pattern.
- will always only return 0 or 1

Regex for Validation



```
$pattern = '/[a-z0-9]{3,10}/i';
$subject = 'dah16';

if (preg_match($pattern, $subject) ) {
    echo "Pattern Matched!";
} else {
    echo "Pattern Did Not Match!";
}
```

- 1 evaluates to true
- 0 evaluates to false

Regex for Validation

- There is some specific ruleset
- Input should match the ruleset
- Input cannot be predicted
- Input can be absolutely anything

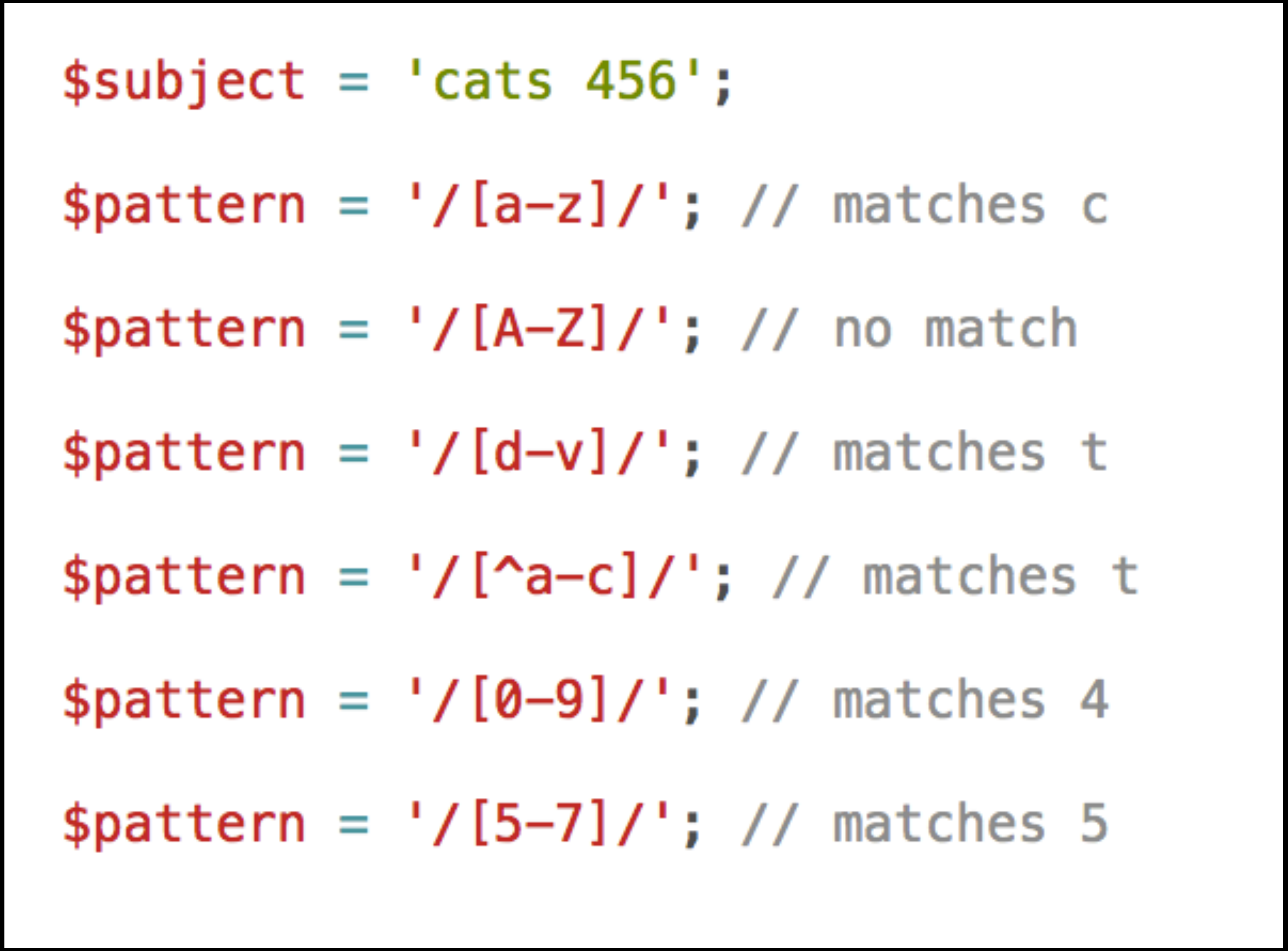
Regex for Validation

Our Ruleset and requirements for usernames:

- Must be at least 7 characters long
- Must be at most 17 characters long
- May only contain letters, numbers, dashes, underscores
- May contain both capital and lowercase letters
- Cannot have spaces

Regex - Character Sets

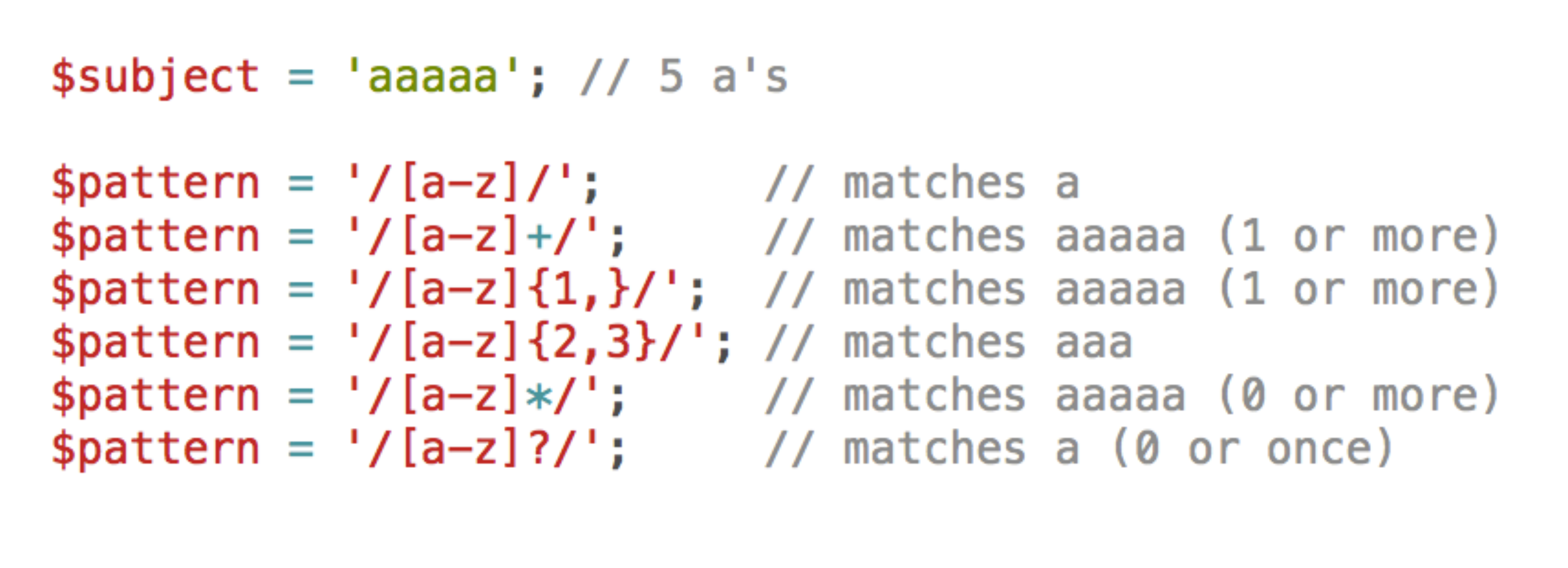
Matching a single character



```
$subject = 'cats 456';

$pattern = '/[a-z]/'; // matches c

$pattern = '/[A-Z]/'; // no match

$pattern = '/[d-v]/'; // matches t

$pattern = '/[^a-c]/'; // matches t

$pattern = '/[0-9]/'; // matches 4

$pattern = '/[5-7]/'; // matches 5
```

Regex - Repetition

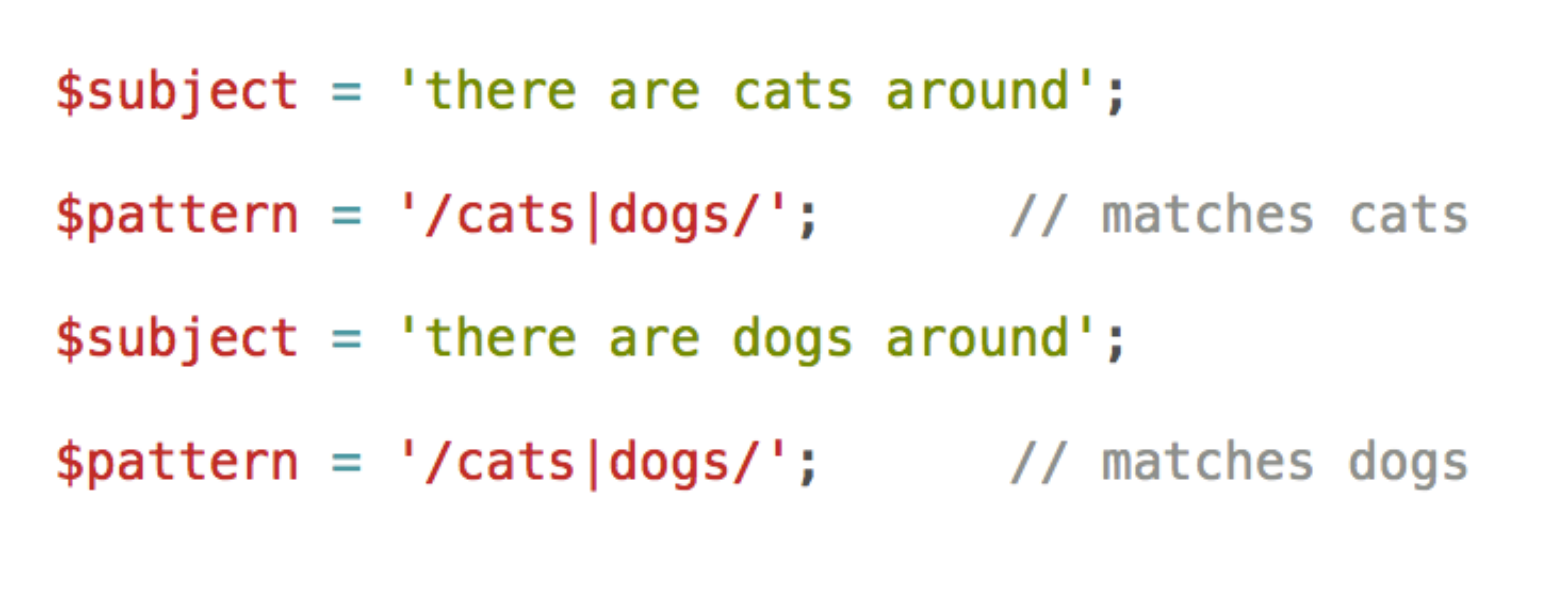
Matching 0, 1, more...



```
$subject = 'aaaaa'; // 5 a's

$pattern = '/[a-z]/';      // matches a
$pattern = '/[a-z]+/';     // matches aaaaa (1 or more)
$pattern = '/[a-z]{1,}/';  // matches aaaaa (1 or more)
$pattern = '/[a-z]{2,3}/'; // matches aaa
$pattern = '/[a-z]*/';     // matches aaaaa (0 or more)
$pattern = '/[a-z]?/';     // matches a (0 or once)
```

Regex - Matching *this* or *that*

Pipe operator for either/or



```
$subject = 'there are cats around';

$pattern = '/cats|dogs/';       // matches cats

$subject = 'there are dogs around';

$pattern = '/cats|dogs/';       // matches dogs
```

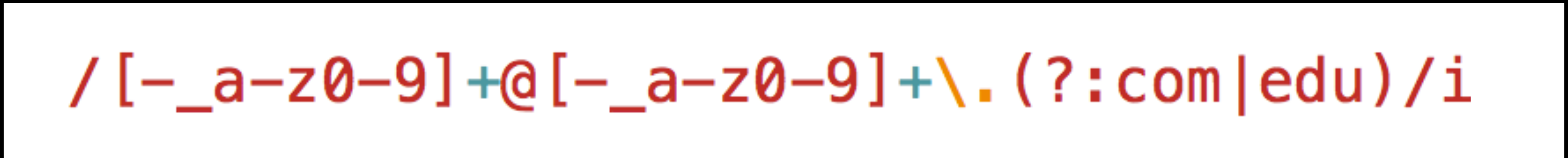
Regex - Beginning and End

- ^ matches the beginning of the string
- $ matches the end of the string

Open Your Browsers

- http://davehauenstein.com/nyu/INFO1-CE9224-2012-Summer/code/regex.php

Regex Syntax



```
/[-_a-z0-9]+@[-_a-z0-9]+\.(?:com|edu)/i
```

Revisit the email Regex

Some References

- http://www.codinghorror.com/blog/2008/06/regular-expressions-now-you-have-two-problems.html
- http://www.amazon.com/Mastering-Regular-Expressions-Jeffrey-Friedl/dp/0596528124/ref=sr_1_1?ie=UTF8&qid=1335396847&sr=8-1
- http://www.regular-expressions.info/
- http://www.solmetra.com/scripts/regex/index.php
- http://net.tutsplus.com/tutorials/other/8-regular-expressions-you-should-know/

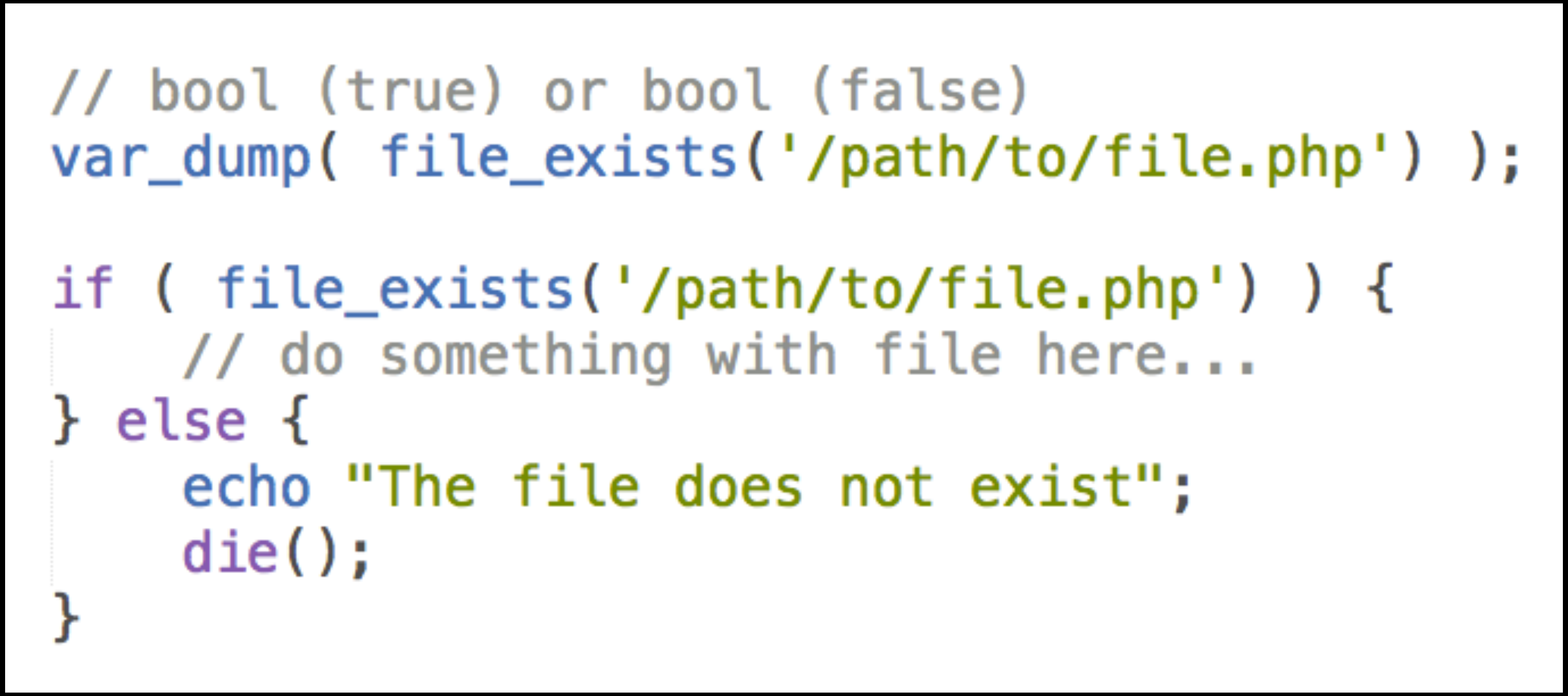
File Handling

Getting Info on a File

- Checking for a file's existence
- Retrieving the file's size in bytes
- Retrieving created/last modified times
- Retrieving the file name out of the path

Checking if a File Exists

Before doing anything with a file,
you must check if it exists first.

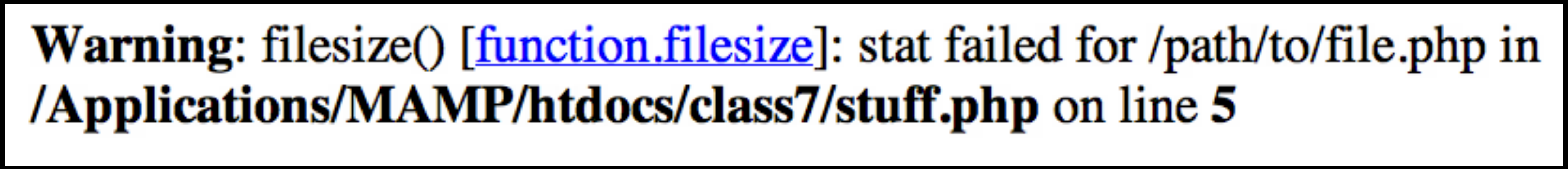


```
// bool (true) or bool (false)
var_dump( file_exists('/path/to/file.php') );

if ( file_exists('/path/to/file.php') ) {
    // do something with file here...
} else {
    echo "The file does not exist";
    die();
}
```

Checking if a File Exists

Before doing anything with a file,
you must check if it exists first.

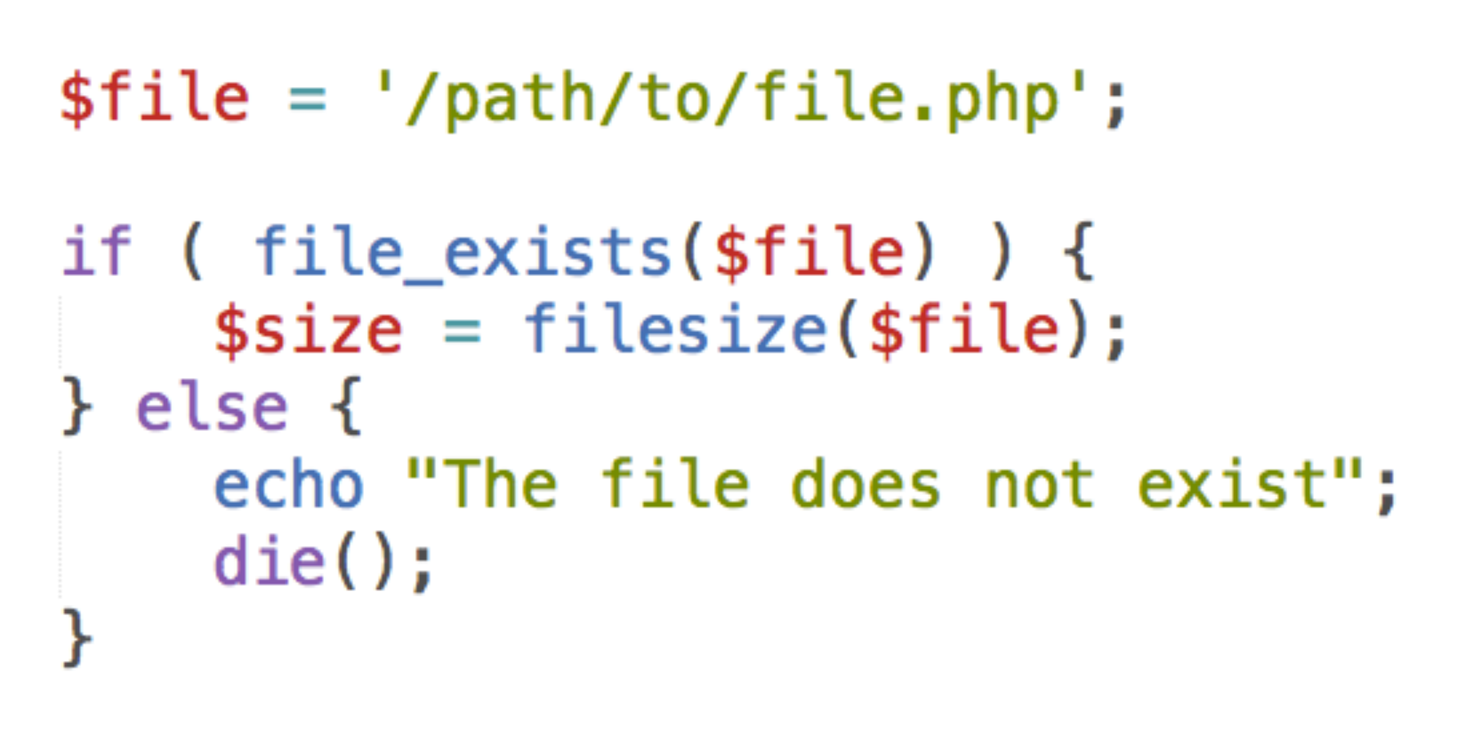
Otherwise the script will give PHP warnings. Think of this as using isset() only on files not variables.



Warning: filesize() [function.filesize]: stat failed for /path/to/file.php in **/Applications/MAMP/htdocs/class7/stuff.php** on line **5**

Retrieving a File’s Size

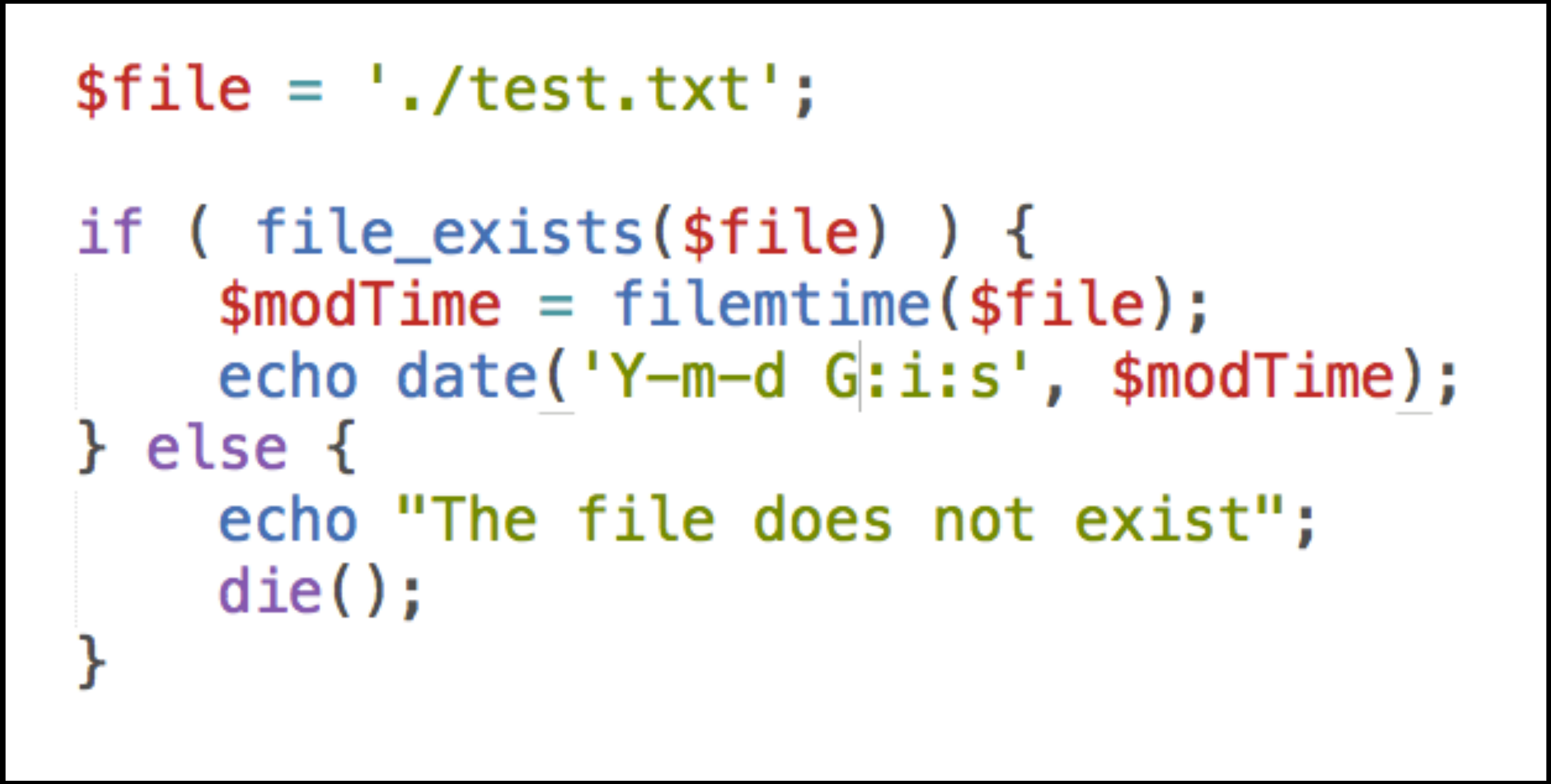
Use PHP’s function filesize(‘/path/to/file.txt’) to determine a file’s size in bytes. This function returns an integer value.



```
$file = '/path/to/file.php';

if ( file_exists($file) ) {
    $size = filesize($file);
} else {
    echo "The file does not exist";
    die();
}
```

Created/Modified Time

Use PHP’s function filemtime(‘/path/to/file.txt’) to determine when a file was either created or last modified. It returns a UNIX timestamp.

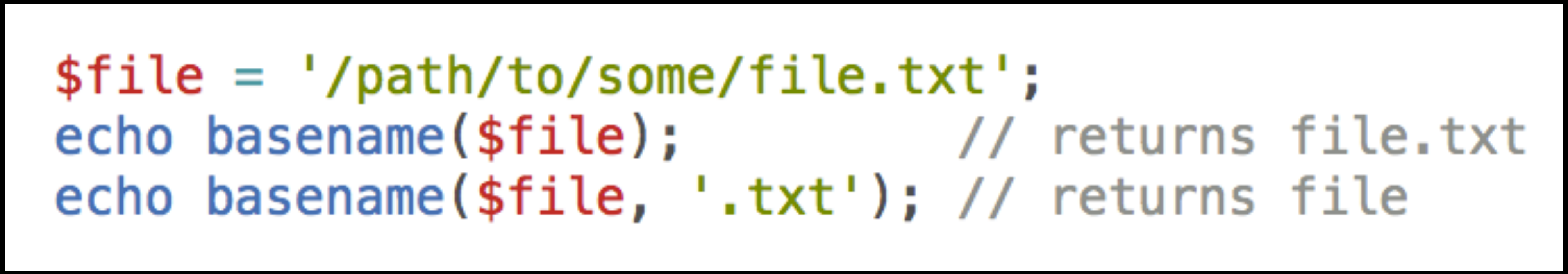


```
$file = './test.txt';

if ( file_exists($file) ) {
    $modTime = filemtime($file);
    echo date('Y-m-d G:i:s', $modTime);
} else {
    echo "The file does not exist";
    die();
}
```

Retrieving a File’s Name From a Path

Use PHP’s function basename(‘/path/to/file.txt’) to determine a file’s base name. This function returns a string.

Not operating on the file itself, just the string name of the file. Doesn’t require file_exists call.



```
$file = '/path/to/some/file.txt';
echo basename($file);          // returns file.txt
echo basename($file, '.txt'); // returns file
```

Opening, Writing, and Closing Files

Opening a File - fopen()

- PHP’s built in function fopen($path, $mode)
- $path is the full/relative path to the file
- $mode is what you would like to do with the file to be opened

Opening a File - fopen()

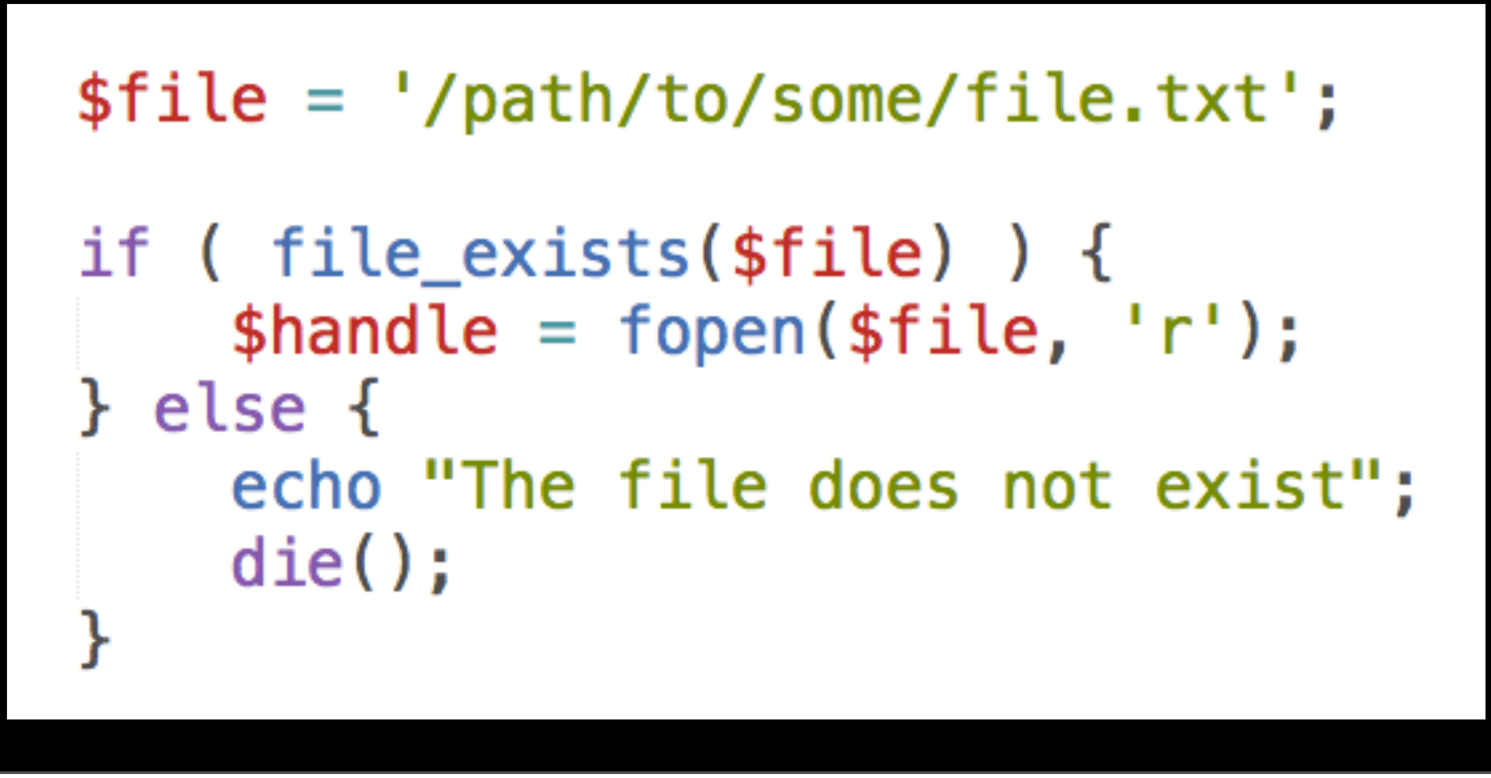
Available Modes

- r - Open for reading only. Pointer at beginning.
- r+ - Open for reading and writing. Pointer at beginning.
- w - Open for writing only. Existing content lost. If file doesn't exist, attempts to create it.
- w+ - Open for reading and writing. Existing content lost. If file doesn't exist, attempts to create it.
- a - Open for appending only. Data written to end of existing file. If file doesn't exist, attempts to create it.
- a+ Open for reading and appending. Data written to end of existing file. If file doesn't exist, attempts to create it.

Opening a File - fopen()

fopen() returns a *resource*. This is a type we haven’t talked about yet.

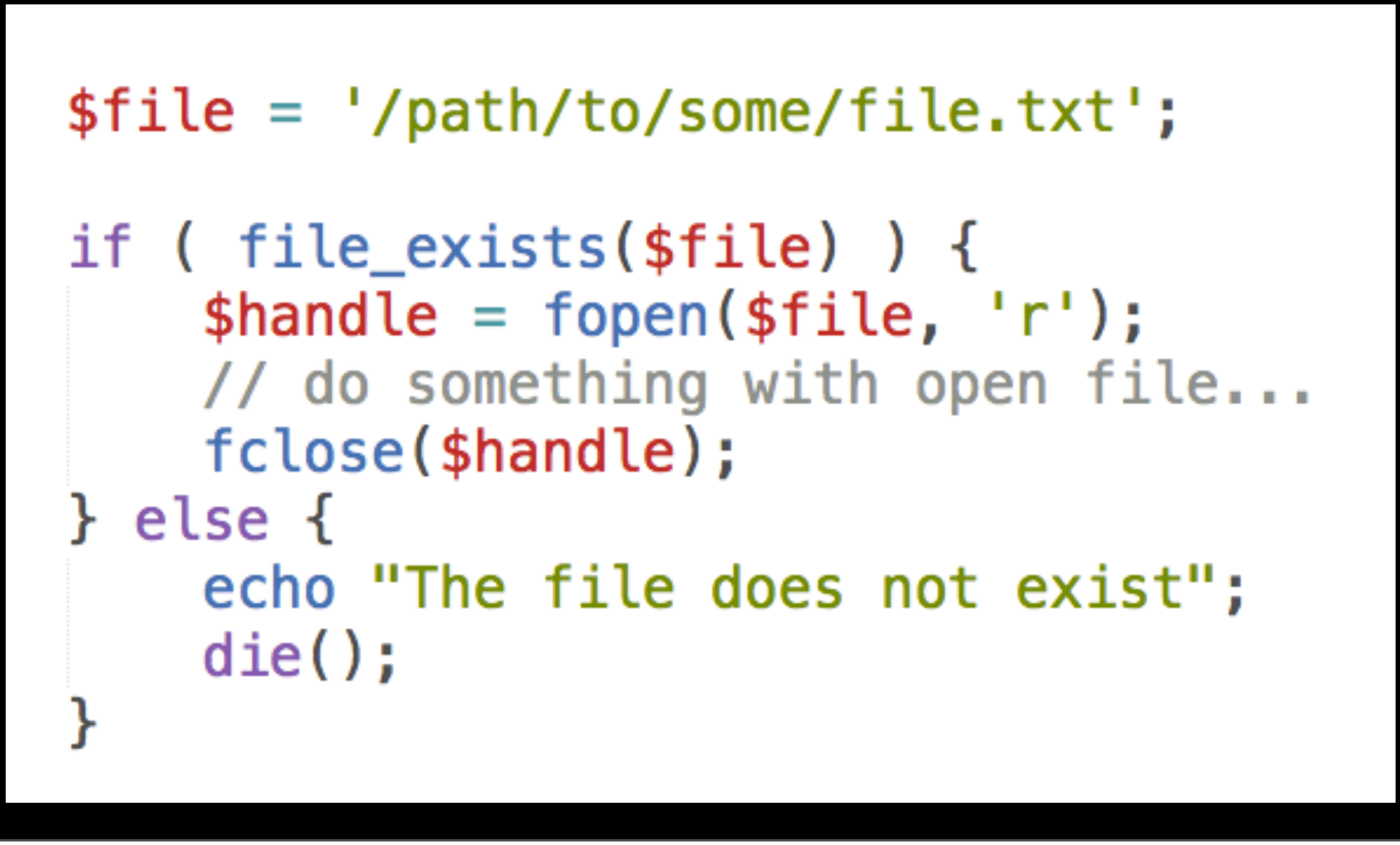
A resource is a special variable, holding a reference to an external resource.



```
$file = '/path/to/some/file.txt';

if ( file_exists($file) ) {
    $handle = fopen($file, 'r');
} else {
    echo "The file does not exist";
    die();
}
```

Closing a File - fclose()

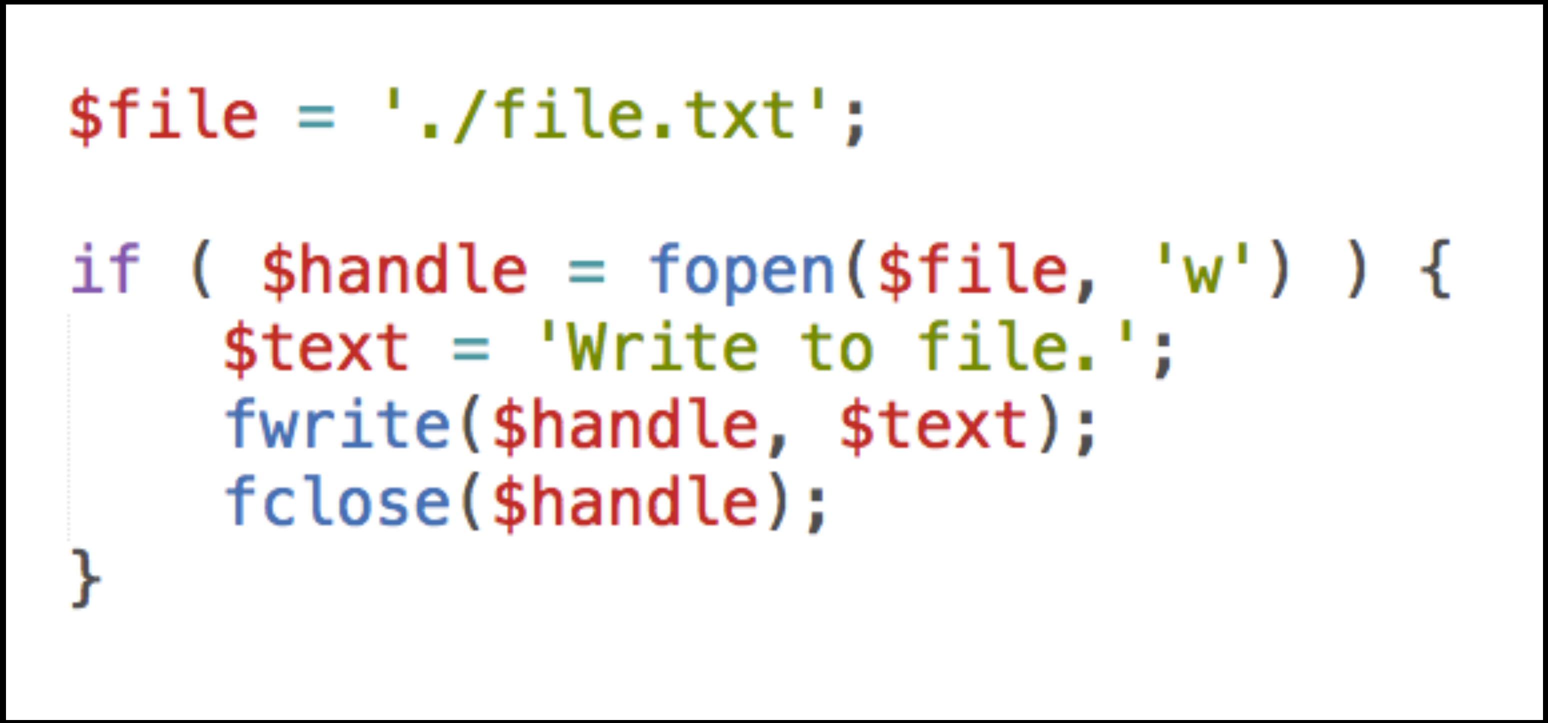
fclose() takes one argument; the file Resource. PHP will do this automatically, but it's a good idea to do it in your scripts: it frees up file quicker so it can be used by other scripts.



```php
$file = '/path/to/some/file.txt';

if ( file_exists($file) ) {
    $handle = fopen($file, 'r');
    // do something with open file...
    fclose($handle);
} else {
    echo "The file does not exist";
    die();
}
```

Writing to a file - fwrite()

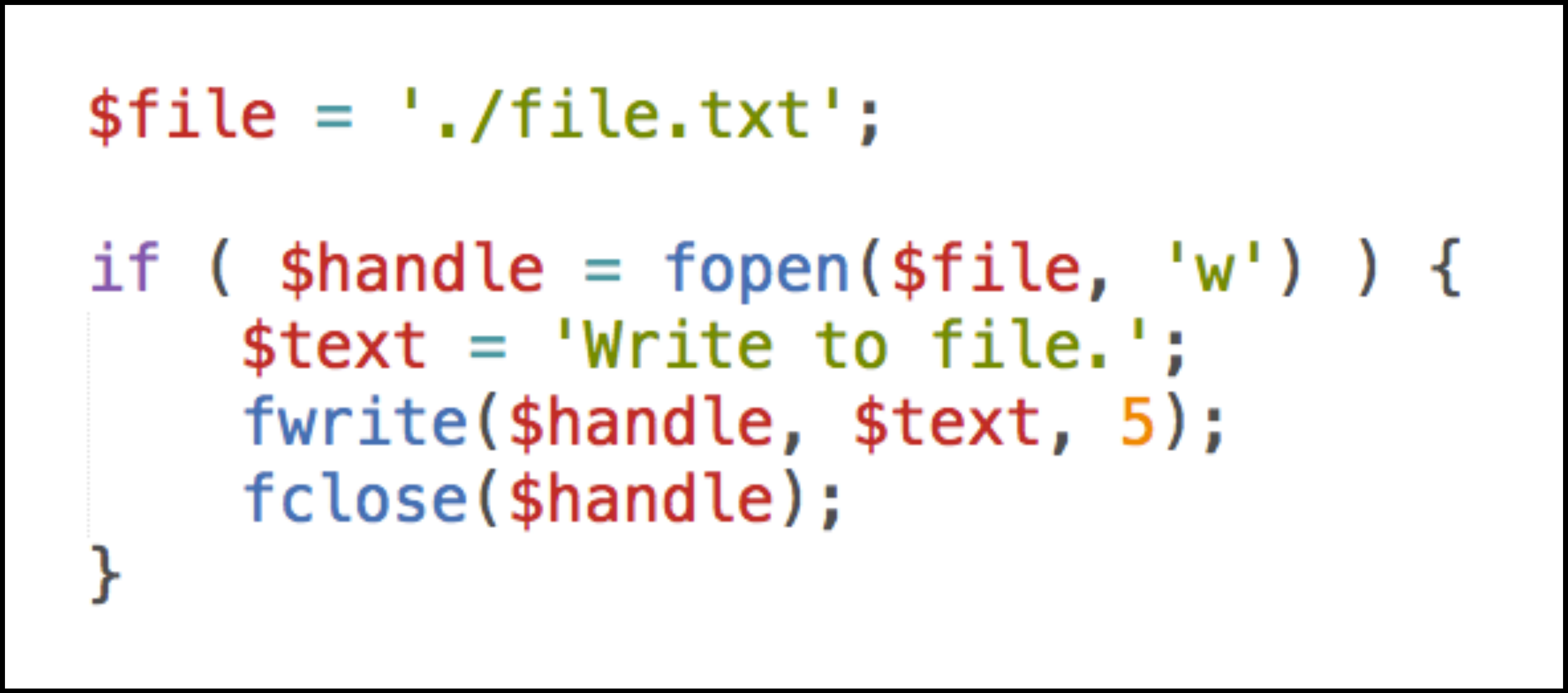
Use mode ‘w’. If file doesn’t exist, PHP creates it.



```
$file = './file.txt';

if ( $handle = fopen($file, 'w') ) {
    $text = 'Write to file.';
    fwrite($handle, $text);
    fclose($handle);
}
```

Writing to a file - fwrite()

Specifying a 3rd argument to specify the number of bytes to write.



```
$file = './file.txt';

if ( $handle = fopen($file, 'w') ) {
    $text = 'Write to file.';
    fwrite($handle, $text, 5);
    fclose($handle);
}
```

Writing to a file - file_put_contents()

```
file_put_contents($path, $data, $flags=0);
```

This function is identical to calling fopen(), fwrite() and fclose() successively to write data to a file.

If filename does not exist, the file is created.

Otherwise, the existing file is overwritten, unless the FILE_APPEND flag is set.

Returns number of bytes written, false on failure.

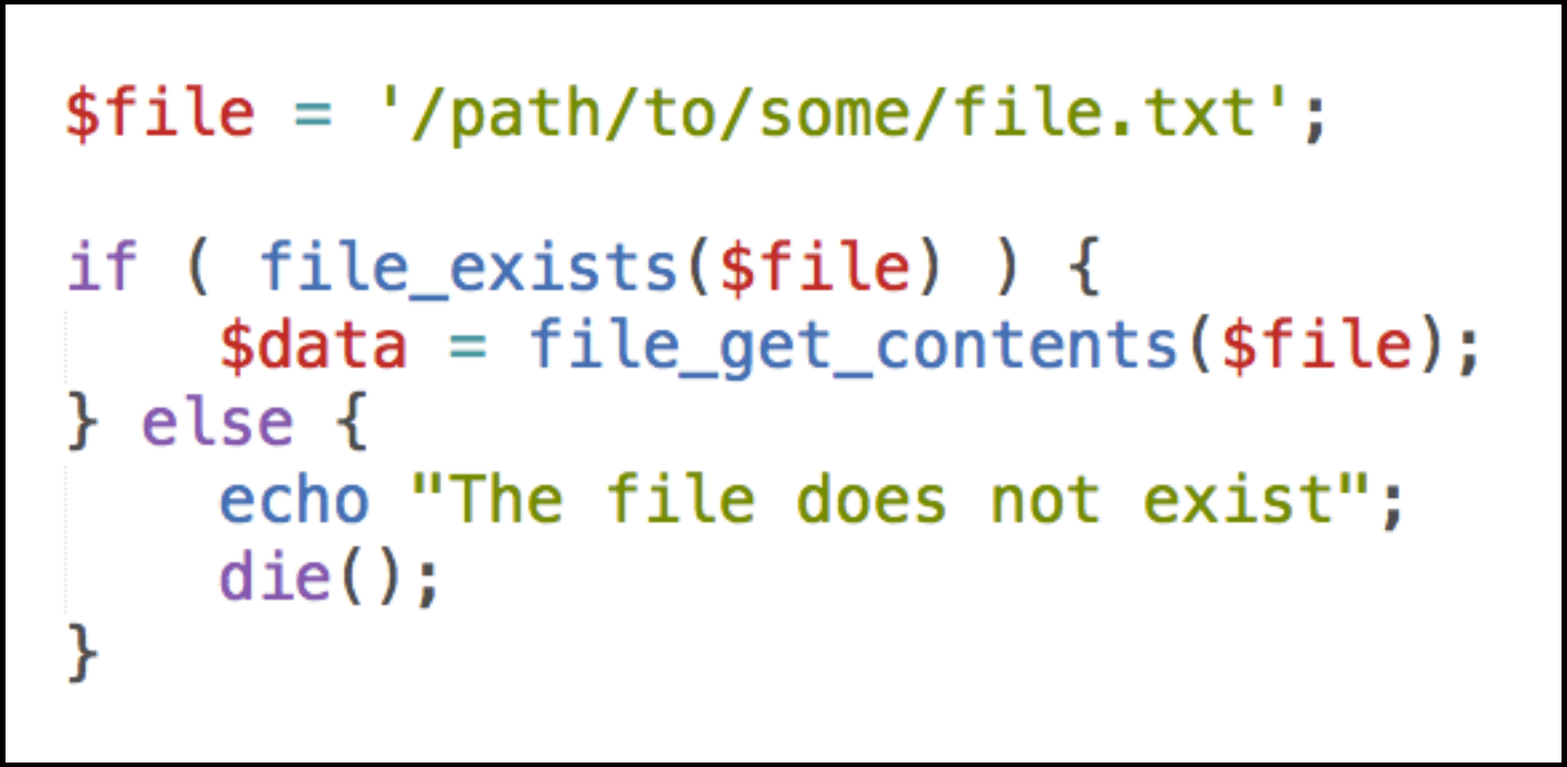
Reading From Files

Reading From Files

- Scenario 1: Reading entire file into a variable (either string or array).
- Scenario 2: Reading chunks of a file at a time into a variable.

Reading Entire File - file_get_contents()

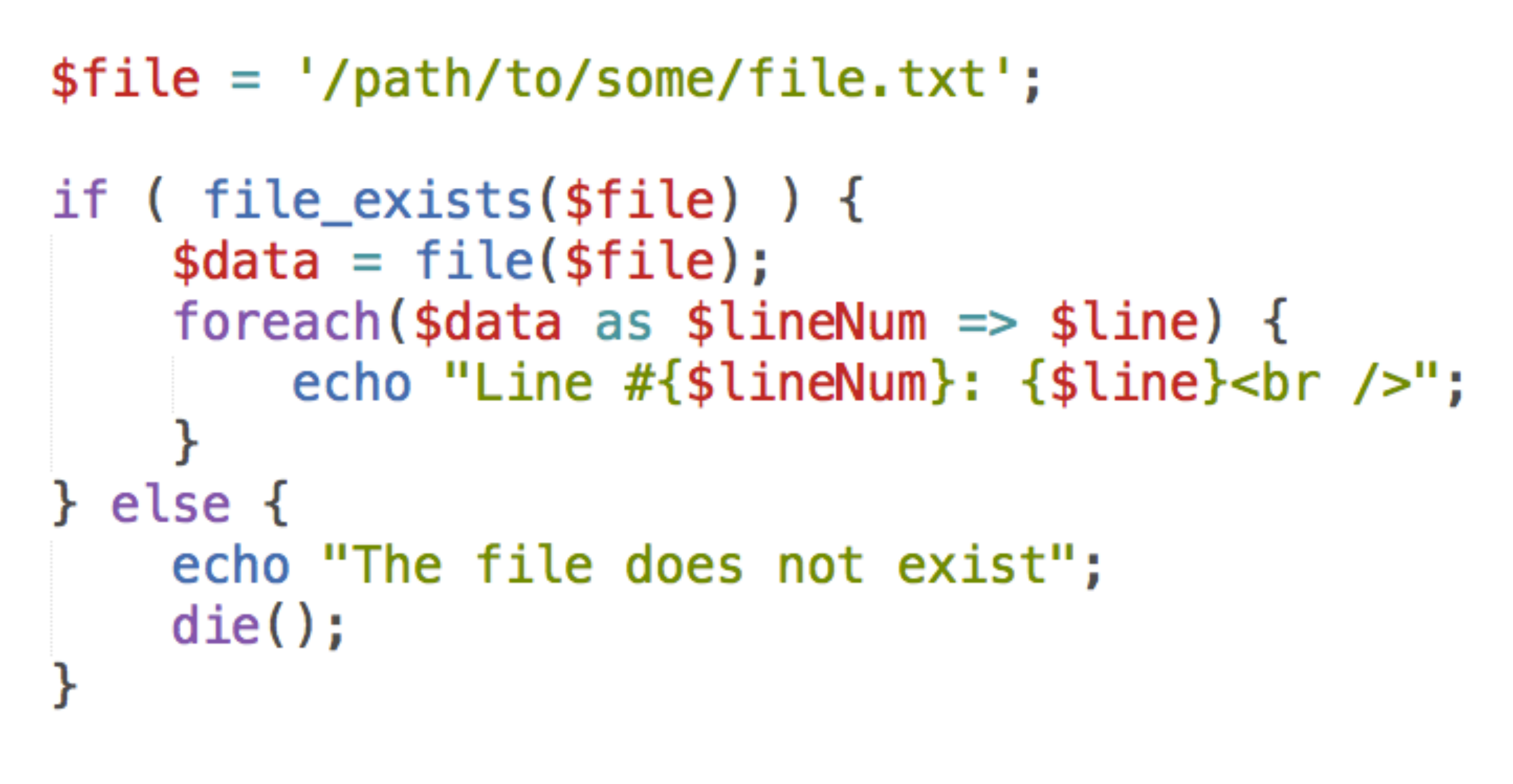
file_get_contents($path);



```php
$file = '/path/to/some/file.txt';

if ( file_exists($file) ) {
    $data = file_get_contents($file);
} else {
    echo "The file does not exist";
    die();
}
```

Reading Entire File - file()

file($path);

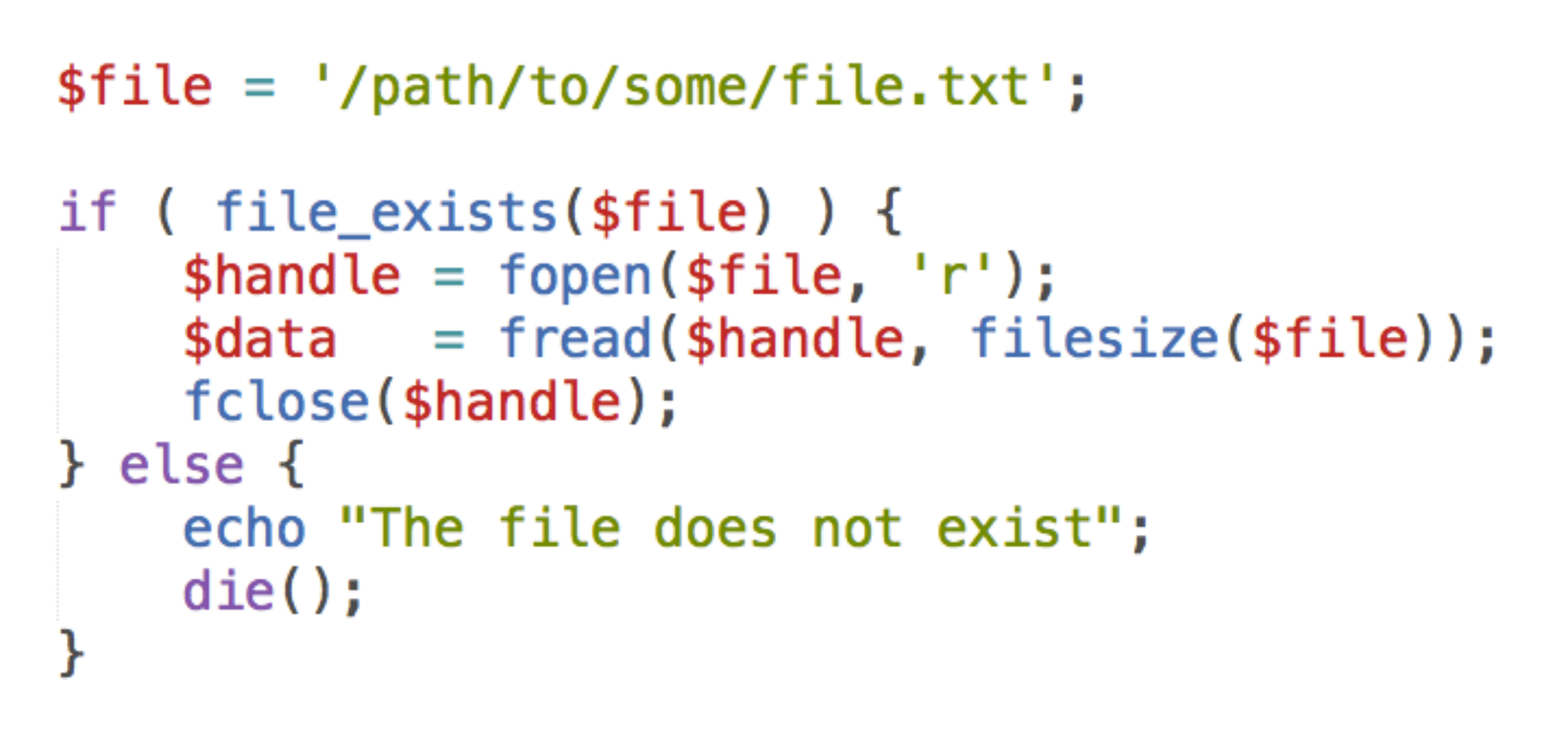


```
$file = '/path/to/some/file.txt';

if ( file_exists($file) ) {
    $data = file($file);
    foreach($data as $lineNum => $line) {
        echo "Line #{$lineNum}: {$line}<br />";
    }
} else {
    echo "The file does not exist";
    die();
}
```

Reading Entire File - fread()

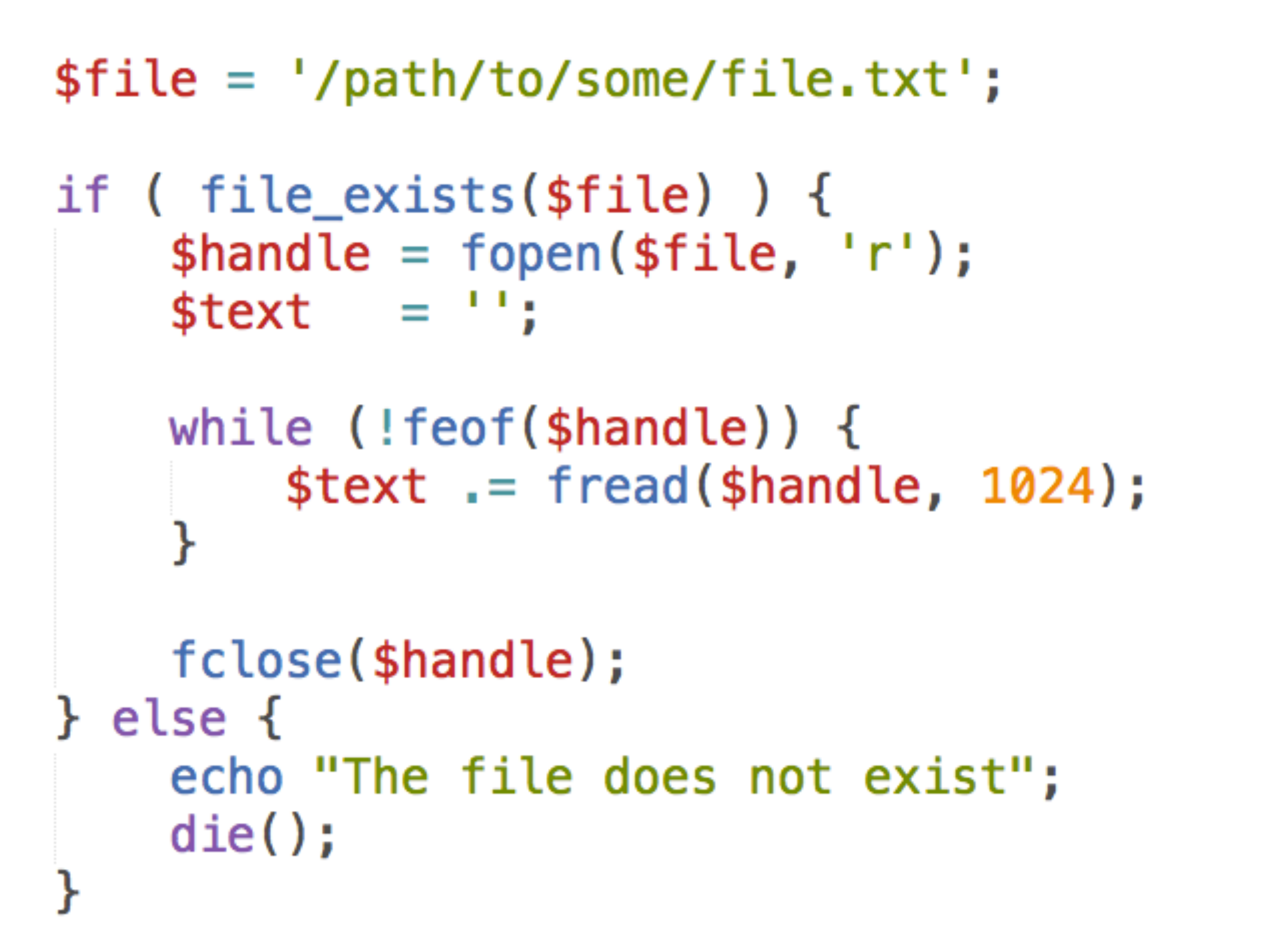
fread($handle, $length);

Note: file_get_contents() has better performance.



```
$file = '/path/to/some/file.txt';

if ( file_exists($file) ) {
    $handle = fopen($file, 'r');
    $data   = fread($handle, filesize($file));
    fclose($handle);
} else {
    echo "The file does not exist";
    die();
}
```

Reading Chunks of File - fread() and feof()

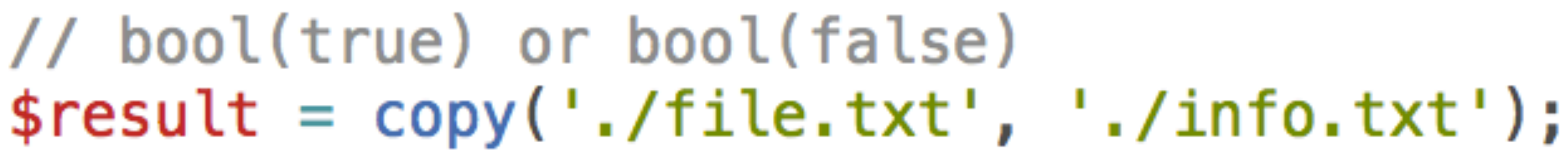


```php
$file = '/path/to/some/file.txt';

if ( file_exists($file) ) {
    $handle = fopen($file, 'r');
    $text   = '';

    while (!feof($handle)) {
        $text .= fread($handle, 1024);
    }

    fclose($handle);
} else {
    echo "The file does not exist";
    die();
}
```

Additional File Operations

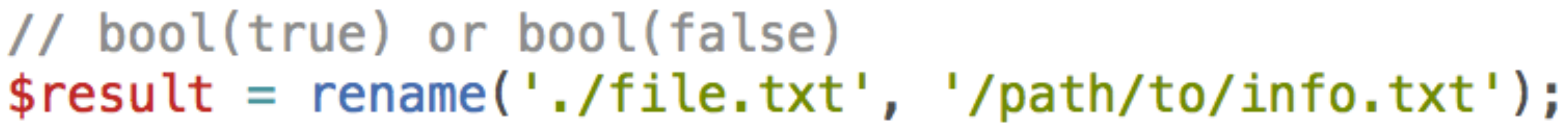
Copy, Rename, Delete

Copy a file to another file



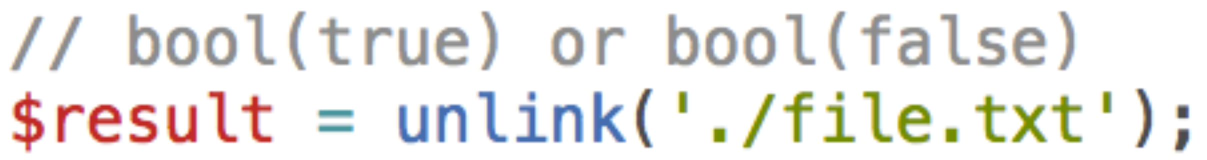
```php
// bool(true) or bool(false)
$result = copy('./file.txt', './info.txt');
```

Rename or move a file



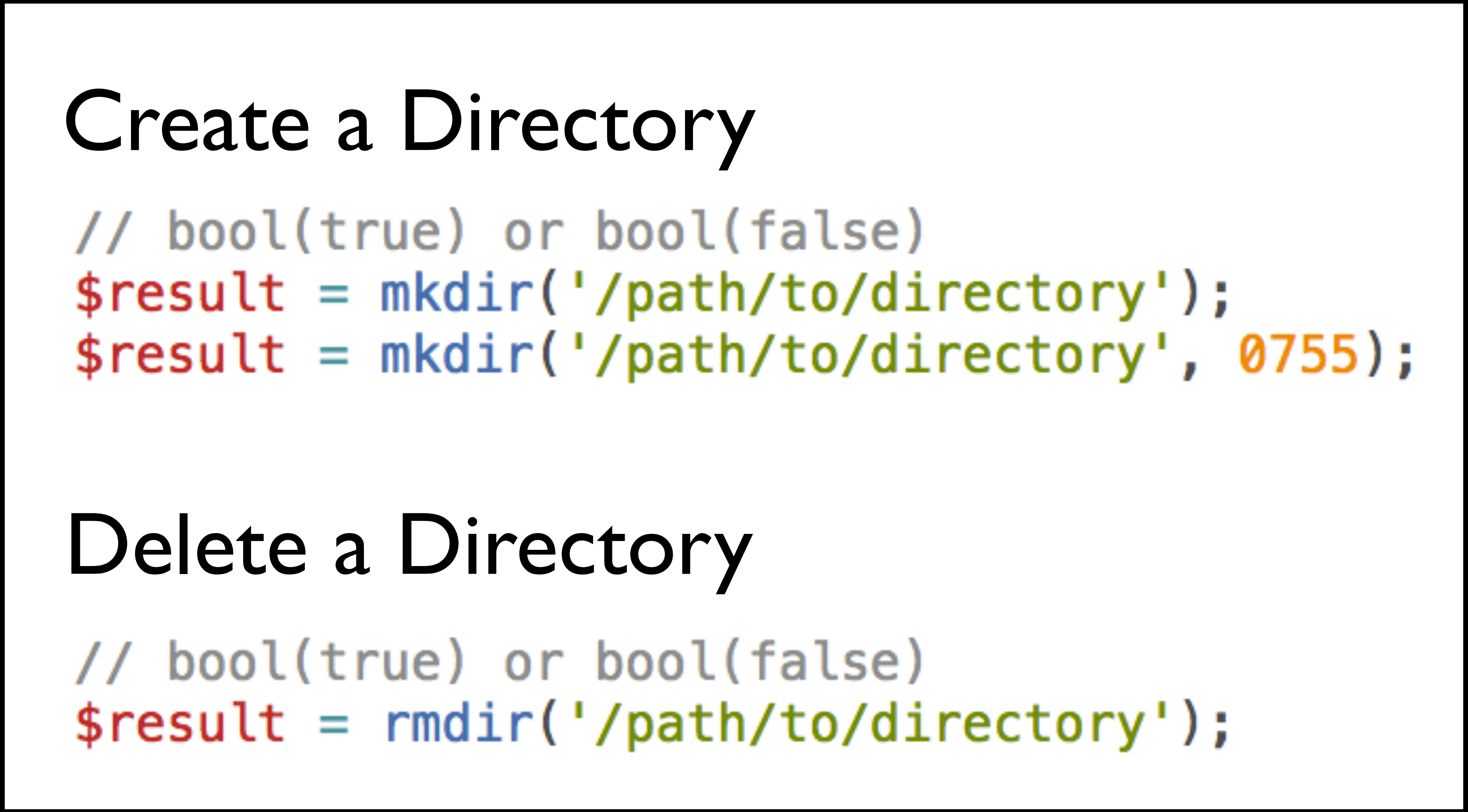
```php
// bool(true) or bool(false)
$result = rename('./file.txt', '/path/to/info.txt');
```

Delete a file



```php
// bool(true) or bool(false)
$result = unlink('./file.txt');
```

Directories/Folders



Create a Directory

```
// bool(true) or bool(false)
$result = mkdir('/path/to/directory');
$result = mkdir('/path/to/directory', 0755);
```

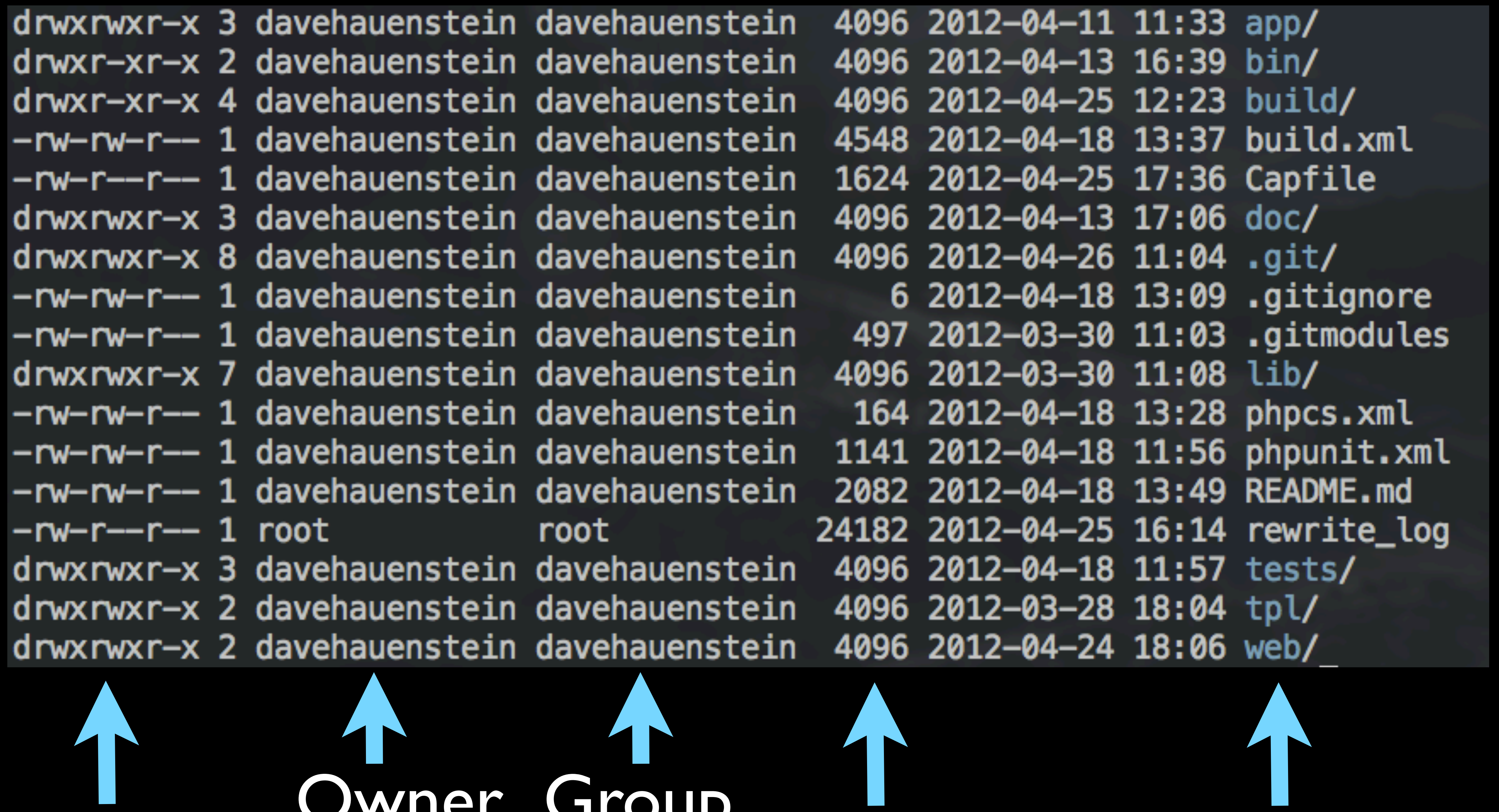
Delete a Directory

```
// bool(true) or bool(false)
$result = rmdir('/path/to/directory');
```

Notes on File Permissions

- Operating on files requires you to have proper permissions, otherwise PHP will give you errors.
- Apache runs PHP, which means the Apache user on the OS must have read/write access to files being operated on.
- File permissions can be manipulated with PHP, most languages, or on the command line.

File Permissions



```
drwxrwxr-x 3 davehauenstein davehauenstein  4096 2012-04-11 11:33 app/
drwxr-xr-x 2 davehauenstein davehauenstein  4096 2012-04-13 16:39 bin/
drwxr-xr-x 4 davehauenstein davehauenstein  4096 2012-04-25 12:23 build/
-rw-rw-r-- 1 davehauenstein davehauenstein  4548 2012-04-18 13:37 build.xml
-rw-r--r-- 1 davehauenstein davehauenstein  1624 2012-04-25 17:36 Capfile
drwxrwxr-x 3 davehauenstein davehauenstein  4096 2012-04-13 17:06 doc/
drwxrwxr-x 8 davehauenstein davehauenstein  4096 2012-04-26 11:04 .git/
-rw-rw-r-- 1 davehauenstein davehauenstein     6 2012-04-18 13:09 .gitignore
-rw-rw-r-- 1 davehauenstein davehauenstein   497 2012-03-30 11:03 .gitmodules
drwxrwxr-x 7 davehauenstein davehauenstein  4096 2012-03-30 11:08 lib/
-rw-rw-r-- 1 davehauenstein davehauenstein   164 2012-04-18 13:28 phpcs.xml
-rw-rw-r-- 1 davehauenstein davehauenstein  1141 2012-04-18 11:56 phpunit.xml
-rw-rw-r-- 1 davehauenstein davehauenstein  2082 2012-04-18 13:49 README.md
-rw-r--r-- 1 root           root           24182 2012-04-25 16:14 rewrite_log
drwxrwxr-x 3 davehauenstein davehauenstein  4096 2012-04-18 11:57 tests/
drwxrwxr-x 2 davehauenstein davehauenstein  4096 2012-03-28 18:04 tpl/
drwxrwxr-x 2 davehauenstein davehauenstein  4096 2012-04-24 18:06 web/
```

Owner Group

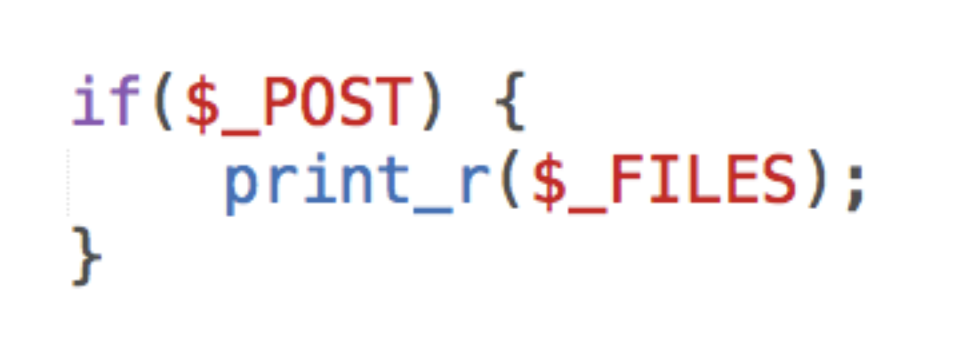
Permissions File Size File/Dir Name

Uploading Files

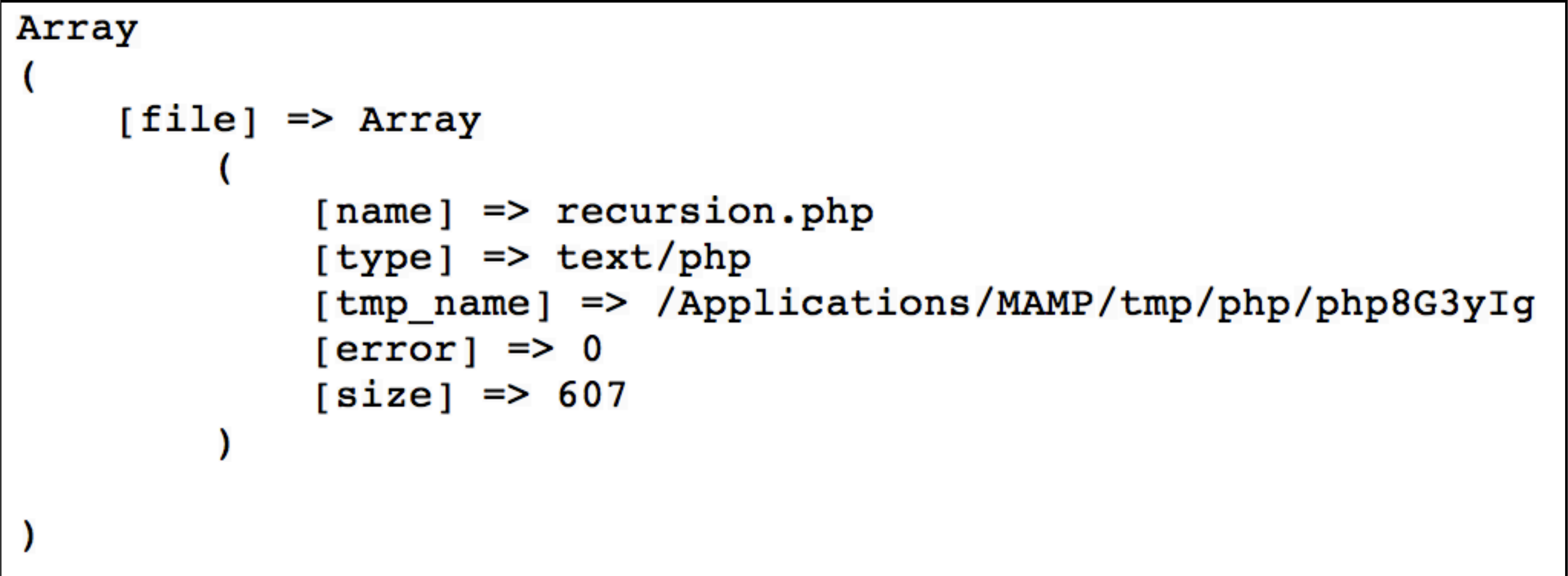
Uploading Files

- Set Up Form
    - <form enctype="multipart/form-data" ...>
    - <input type="file" name="a_file" id="a_file" />
- Use PHP to determine if file was successfully uploaded.
- If submitted without error, file is moved to a temporary place on the server.
- Use PHP to copy file from temporary location to permanent location.
- Use file for whatever purpose application needs it.

Uploading Files

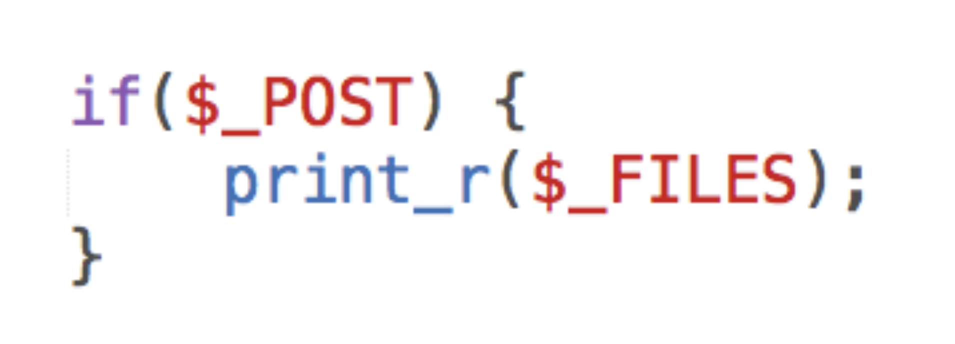


```
if($_POST) {
    print_r($_FILES);
}
```



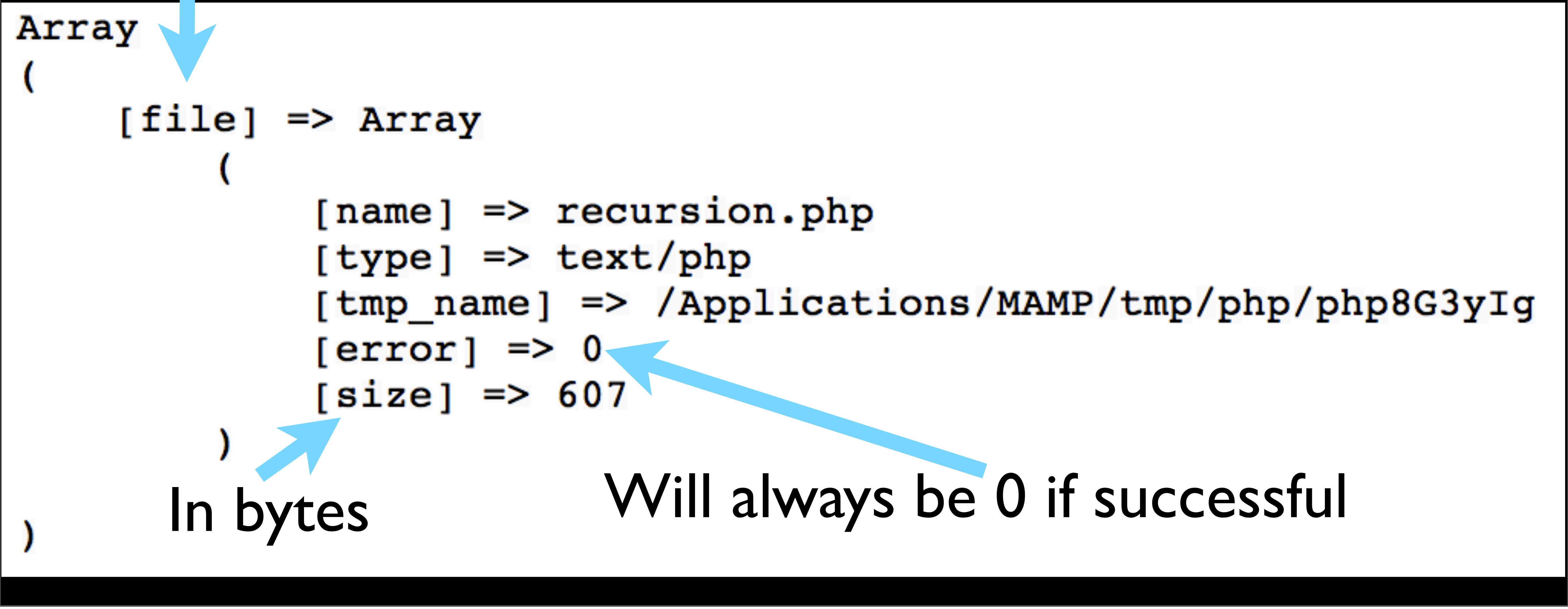
```
Array
(
    [file] => Array
        (
            [name] => recursion.php
            [type] => text/php
            [tmp_name] => /Applications/MAMP/tmp/php/php8G3yIg
            [error] => 0
            [size] => 607
        )

)
```

Uploading Files



```
if($_POST) {
    print_r($_FILES);
}
```

Name of form field



```
Array
(
    [file] => Array
        (
            [name] => recursion.php
            [type] => text/php
            [tmp_name] => /Applications/MAMP/tmp/php/php8G3yIg
            [error] => 0
            [size] => 607
        )

)
```

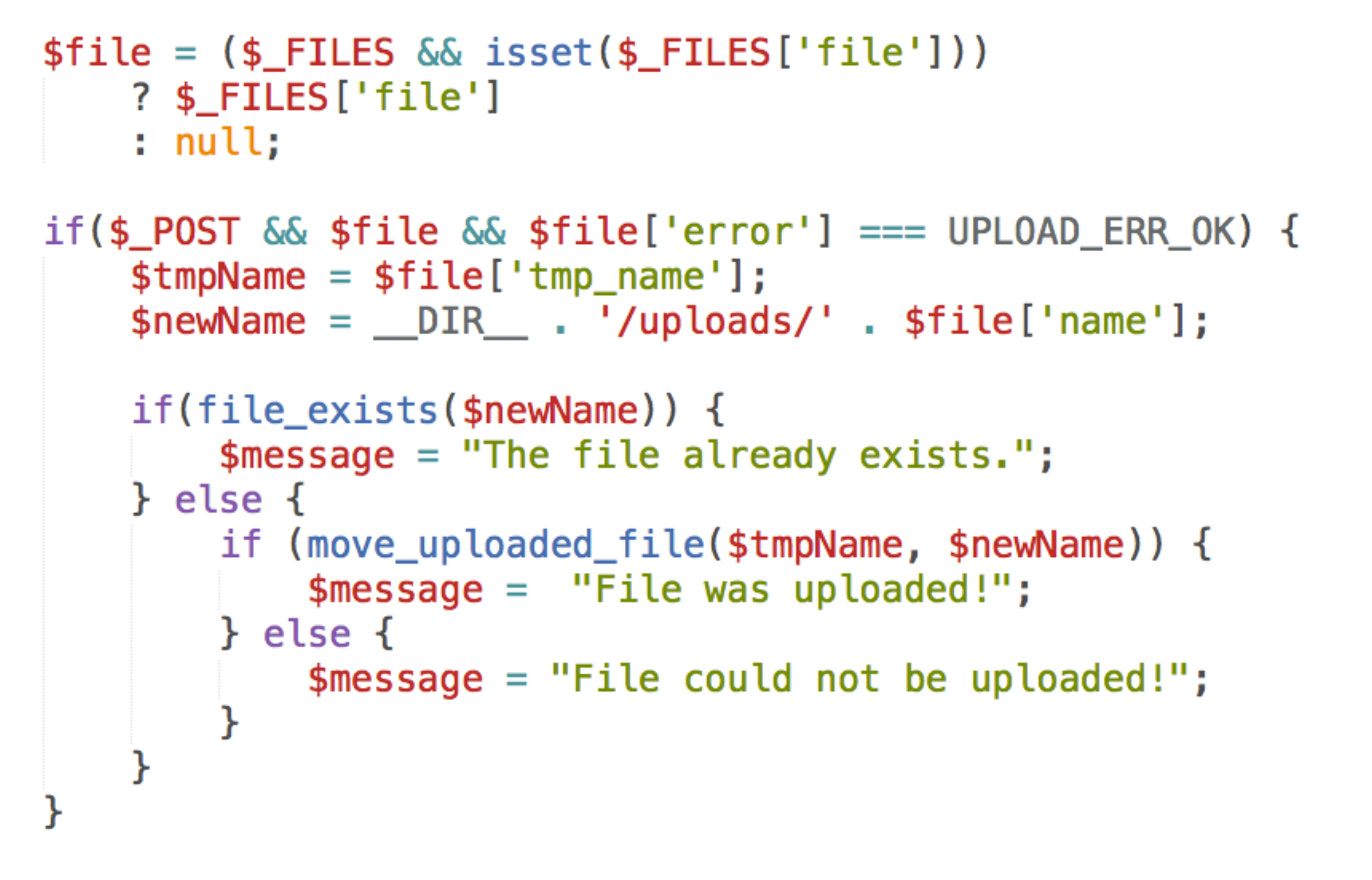
In bytes

Will always be 0 if successful

Errors

PHP Constant	Value	Description
UPLOAD_ERR_OK	0	There is no error, the file uploaded with success.
UPLOAD_ERR_INI_SIZE	1	The uploaded file exceeds the upload_max_filesize directive in *php.ini*
UPLOAD_ERR_FORM_SIZE	2	The uploaded file exceeds the MAX_FILE_SIZE directive that was specified in the HTML form.
UPLOAD_ERR_PARTIAL	3	The uploaded file was only partially uploaded.
UPLOAD_ERR_NO_FILE	4	No file was uploaded.
UPLOAD_ERR_NO_TMP_DIR	6	No file was uploaded.
UPLOAD_ERR_CANT_WRITE	7	Failed to write file to disk. Introduced in PHP 5.1.0.
UPLOAD_ERR_EXTENSION	8	A PHP extension stopped the file upload.

Uploading Files



```php
$file = ($_FILES && isset($_FILES['file']))
    ? $_FILES['file']
    : null;

if($_POST && $file && $file['error'] === UPLOAD_ERR_OK) {
    $tmpName = $file['tmp_name'];
    $newName = __DIR__ . '/uploads/' . $file['name'];

    if(file_exists($newName)) {
        $message = "The file already exists.";
    } else {
        if (move_uploaded_file($tmpName, $newName)) {
            $message =  "File was uploaded!";
        } else {
            $message = "File could not be uploaded!";
        }
    }
}
```

Uploading Files Review

1. Check for POST
2. Check to see if $_FILES['your_file'] is set
3. Check to see if $_FILES['your_file']['error'] is equal to 0 or the constant UPLOAD_ERR_OK
4. Assume everything else is ok, copy the tmp file, to the new location of the uploaded file.
5. Additionally...
    1. check if file exists before overwriting
    2. Check that the destination folder exists and is writeable