

Web Development Foundations

Web Application Servers. PHP Language and Environment.

Assoc. Prof. Andrei PANU – profs.info.uaic.ro/andrei.panu/

* many thanks to Prof. Sabin-Corneliu Buraga for the content on these slides

Table of Contents

1. How do we use a Web application server
2. Essential information about PHP
3. PHP as a procedural programming language
4. PHP features regarding Web interaction
5. Useful tools for Web developers
6. Case studies of Web applications developed in PHP
7. Live PHP code examples

How is an application server used for developing a Web application?

Web Application Server

Purpose:

increasing the efficiency of all processes
involved in the development of Web applications

Web Application Server

Integrated in one/many Web servers

can also provide its own Web server
or runtime environment

Web Application Server

advanced

Can encourage or enforce an architectural vision
regarding Web application development

typical situation:
MVC or variants

Web Application Server

Simplifies the invocation of programs (scripts)
concerning a Web application

-> dynamic content generation on the server side

Web Application Server

Aspects of interest:

programming language(s)

core API

persistent storage of data models

Web interaction

cookies and sessions

development environments + frameworks, components,...

specific characteristics

Web Application Server

Persistent storage

in relational databases – using **SQL**

examples:

ADO.NET for ASP.NET

Java – **JDBC (Java DataBase Connectivity)**

PHP – predefined functions/modules, plus included libraries (**SQLite + mysqli**) or various extensions

Web Application Server

advanced

Persistent storage

tree-based models: **XML**

(semi)structured data

transformations in other formats: XPath, XSLT

processing: DOM, SAX, SimpleXML,...

data validation: DTD, XML Schema, RELAX,...

querying: XQuery

Web Application Server

advanced

Persistent storage

using other non-relational paradigms
(based on graphs and/or key-value)

distributed on Internet, scalable – **NoSQL**

github.com/ericleung/awesome-nosql-guides
dbdb.io

github.com/igorbarinov/awesome-data-engineering#databases

examples:

Cassandra, MarkLogic, MongoDB, Neo4j, OpenLink Virtuoso, Redis

Web Application Server

advanced

Web interaction

facilitated by specific controls specified in the
source code invoked on the server

Web Application Server

advanced

Web interaction

facilitated by specific controls specified in the source code invoked on the server

can emulate HTML form fields and/or provide new interactive controls – e.g., calendar, slideshow,...

-> generation of processable code at client level (front-end)

Web components (HTML + CSS + JavaScript) executed by the browser

Web Application Server

advanced

Web interaction

Web template system

using specifications of content presentation (a Web template), persistent data (e.g., retrieved from a database) is used by a processor (template engine) in order to generate HTML documents or other formats

Web Application Server

advanced

```
<!-- HTML template -->
<h1 class="profile">[@username] profile</h1>

<div class="identity">[@firstName] [@lastName]</div>
<div class="location">[@location]</div>

<!-- PHP program: template processing -->
$profile = new Template ('templates/profile.tpl');
$profile->set ('username') = 'Tux';
$profile->set ('photoURL') = 'imgs/tux.svg';
$profile->set ('firstName') = 'Tuxy';
$profile->set ('lastName') = 'Pinguinesscool';
$profile->set ('location') = 'Romania';
```



example:

the specification of content presentation (HTML template – .tpl) includes variable names – here, using `[@variable]` syntax – that will be substituted by actual values obtained from the program as a result of the Web template system invocation

Web Application Server

advanced

Web interaction

Web template system

on the server

Apache FreeMarker (Java), Blade (PHP), Haml (Ruby),
Mustache (C++, JS, PHP, Python, Scala,...), Pug (Node.js),
Razor (.NET), Smarty (PHP), Tonic (PHP), XSLT (XML)

Web Application Server

advanced

Web interaction

Web template system

on the client

available for JavaScript:

HandleBars, Mustache.js, Nunjucks,...

github.com/sorrycc/awesome-javascript#templating-engines

Web Application Server

advanced

Web interaction

cookie and session management

via data structures/types – examples:

HttpSession class (ASP.NET), **HttpSession** interface (Java servlets),
HTTP::Session (Perl), **session** (Flask – Python framework), **web.session** (web.py),
HttpFoundation (Symfony component – PHP framework),
SessionComponent class (CakePHP), **session** array (Ruby on Rails),
play.mvc.Http.Cookie (Play for Java/Scala), **sessions** (Gorilla – Go),
cookie-parser and **express-session** (Node.js modules for Express)

Web Application Server

advanced

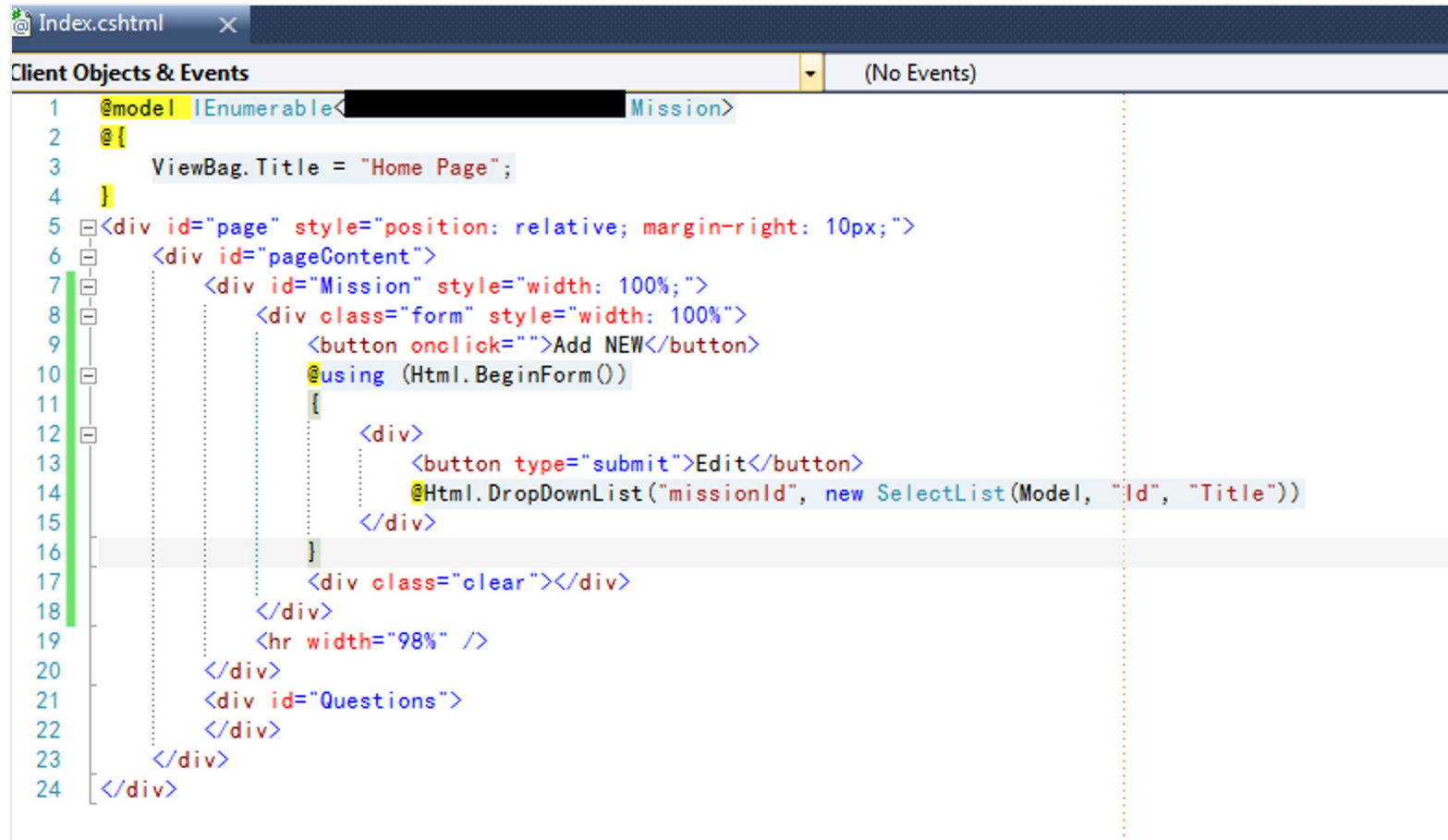
Support for software engineering

encouraging/enforcing the use of **design patterns** (architectural, behavioral, structural, targeting concurrent and/or distributed processing and others)

AMI (asynchronous method invocation), broker,
CBD (components-based development), DAO (data access object),
DDD (domain-driven design), DTO (data transfer object), façade,
MOM (message-oriented middleware), microservice, MV*,
publish-subscribe, SOA (service-oriented architecture), singleton,...

Web Application Server

advanced



```
1 @model IEnumerable<Mission>
2 @
3 ViewBag.Title = "Home Page";
4 }
5 <div id="page" style="position: relative; margin-right: 10px;">
6   <div id="pageContent">
7     <div id="Mission" style="width: 100%;">
8       <div class="form" style="width: 100%;">
9         <button onclick="">Add NEW</button>
10        @using (Html.BeginForm())
11        {
12          <div>
13            <button type="submit">Edit</button>
14            @Html.DropDownList("missionId", new SelectList(Model, "Id", "Title"))
15          </div>
16        }
17        <div class="clear"></div>
18      </div>
19      <hr width="98%" />
20    </div>
21    <div id="Questions">
22    </div>
23  </div>
24 </div>
```

example: **Spaghetti Code anti-pattern**

usually, specific to Web applications that mix business logic with content presentation (view) and data model access mechanism

Web Application Server

Integrated into a software stack

software stack (servers, tools,...) offering support
for developing complex Web applications

available – usually, open source –
for a specific platform (operating system, Web server,
database server, application server, programming language)

Web Application Server

Integrated into a software stack

LAMP

(Linux, Apache HTTP Server, MariaDB/MongoDB, Perl/PHP/Python)

alternatives:

FAMP (FreeBSD), **MAMP** (macOS),
WAMP (Windows), **XAMPP** (multi-platform)

www.apachefriends.org

Web Application Server

Integrated into a software stack

based on JavaScript – full stack Web development

MEAN (MongoDB, Express, Angular, Node.js)

MERN (MongoDB, Express, React, Node.js)

Web Application Server

Integrated into a software stack



complementary approaches:











LAPP (Linux, Apache, PostgreSQL, Perl/PHP/Python)



TALL (Tailwind CSS, Alpine.js, Laravel, Livewire)









tallstack.dev

Web Application Server

What runs httparchive.org?  



Analytics	Font Script
 Google Analytics UA	 Font Awesome
 SpeedCurve	 Google Font API
Web Framework	Web Server
 Bootstrap	 gunicorn 19.7.1
Javascript Graphics	Programming Language
 Highcharts	 Python
	Font
	 Open Sans
	 Open Sans, Sans Serif












What runs codepen.io?  

Analytics	Font Script
 Google Analytics UA	 Google Font API
Web Framework	Web Server
 Ruby on Rails	 Phusion Passenger
Programming Language	CDN
 Ruby	 CloudFlare
Advertising	Tag Managers
 BuySellAds	 Google Tag Manager

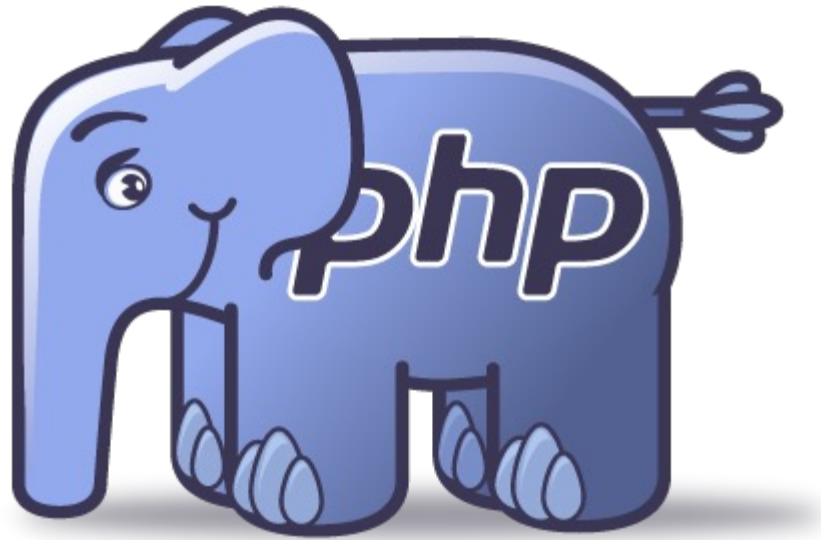
inspecting the technologies used by a Web application
with the instruments **Wapplyzer** and **WhatRuns**

Web Application Server

What runs unsplash.com?  

Analytics	Web Framework
 Google Analytics UA	 Cowboy
 comScore	Programming Language
 Scorecardresearch	 Erlang
 SpeedCurve	 Node.js
Cache	Javascript Frameworks
 Varnish	 Express
Build CI Systems	 React
 webpack	Font
	Font Apple System
	Font Family Apple System, Blinkmacsystemfont, San

inspecting the technologies used by a Web application
with the instrument **WhatRuns**



Essential information about PHP

PHP

History

Important characteristics

PHP as programming language

PHP as Web development platform

PHP – History

Personal Home Page Tools (1995)

Rasmus Lerdorf

PHP 3 (1998)

developed by Zend – Zeev Suraski & Andi Gutmans

PHP 4 (2000)

support for object-oriented programming

PHP 5 (2004) – most recent version: **PHP 5.6 (2014)**

new features inspired by Java

PHP 7 (2015), PHP 7.2 (2017), PHP 7.4 (2019)

strong typing, support for Unicode, performance,...

PHP 8 (2020), PHP 8.1 (2021), PHP 8.4 (2024)

major update: real-time compilation, new data types and syntactic constructs, improvements, etc.

latest minor release: **PHP 8.5 (20 Nov. 2025)**

new features like the URI extension, Pipe operator, support for modifying properties while cloning, etc.

PHP - Characteristics

Web application server

provides a script-based interpreted programming language

can be directly included into HTML documents

PHP - Characteristics

PHP is a procedural language, offering support for other programming paradigms (object-oriented and, more recently, functional)

can be used as a general purpose language, too

reference book:

Josh Lockhart, *PHP: The Right Way*, 2025

phptherightway.com

PHP - Characteristics

Syntax inspired by C, Perl, and Java – case sensitive

whitespaces (space character, Tab, New Line) have no effect
on the execution of the program

usually, the files which contain PHP source code
have the extension **.php**

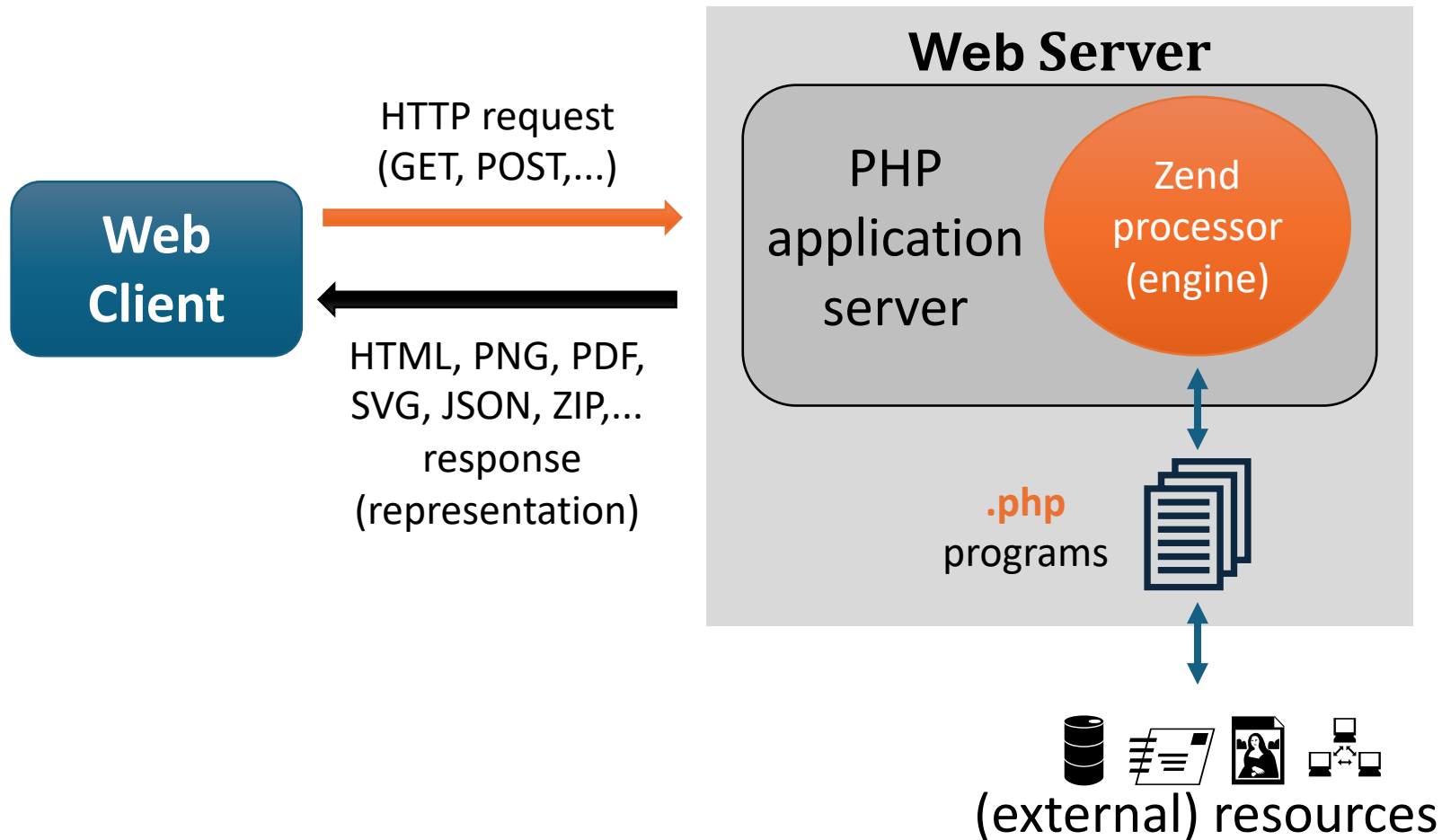
PHP - Characteristics

Freely available – open source – for various platforms
(FreeBSD, Linux, Windows, macOS, etc.)
and Web servers: Apache, IIS, Nginx,...

www.php.net
www.zend.com

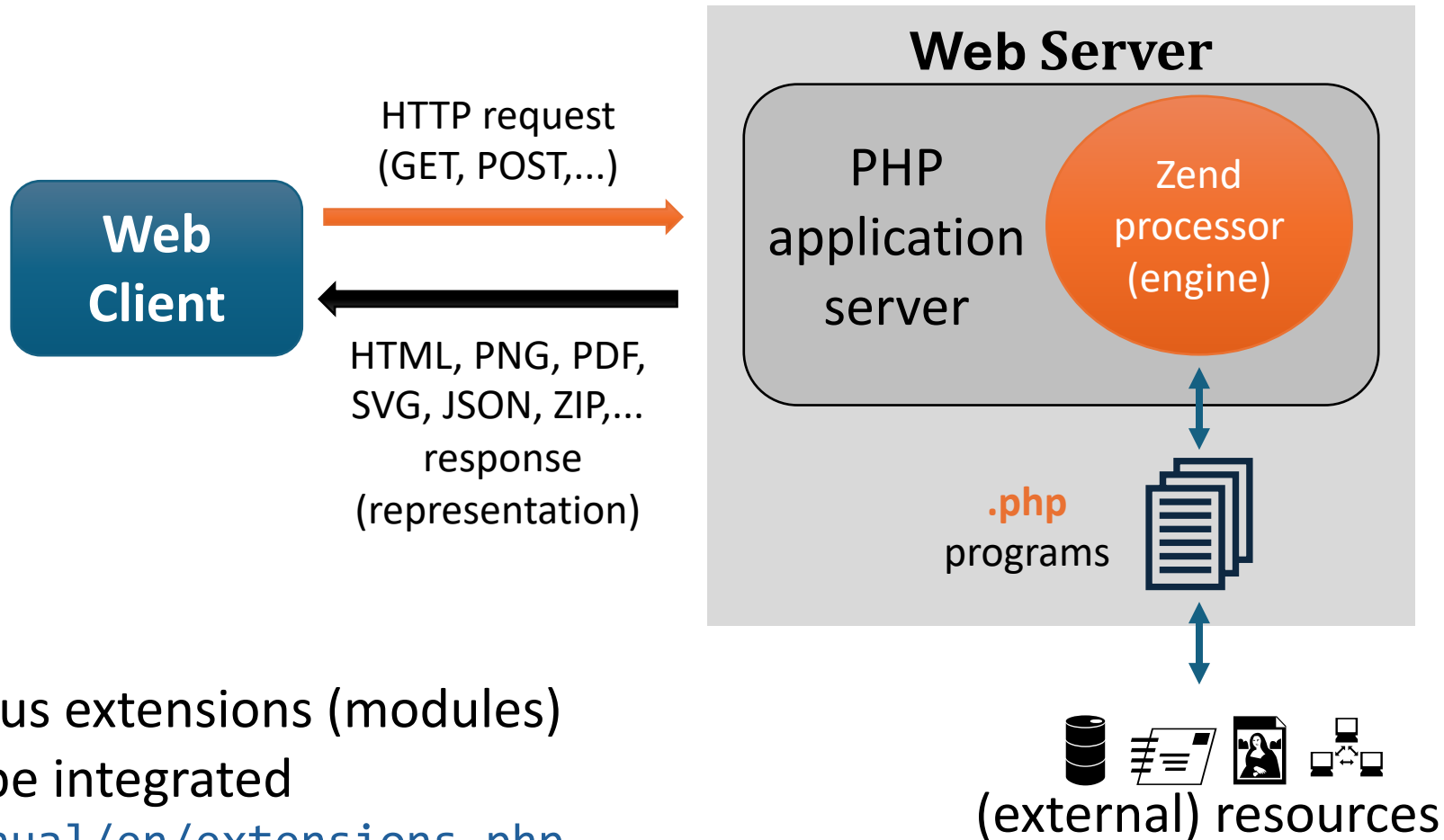
PHP - Characteristics

PHP processor (engine) mode of operation



PHP - Characteristics

PHP processor (engine) mode of operation



eventually, various extensions (modules)
can be integrated

www.php.net/manual/en/extensions.php

PHP - Characteristics

advanced

A PHP program is interpreted by the Zend Engine 2 (PHP 5), Zend Engine 3 (PHP 7), Zend Engine 4 (PHP 8) which generate internal instructions – opcodes

J. Pauli, N. Popov, A. Ferrara, *PHP Internals Book*, 2024

www.phpinternalsbook.com

PHP - Characteristics

advanced

```
precision = 14          ; precision for float values – details at php.net/precision
safe_mode = Off        ; processing control – study php.net/safe-mode
disable_functions = "allow_url_fopen, eval" ; unallowed functions (e.g., security reasons)
max_execution_time = 30 ; max. number of seconds allocated to program execution
memory_limit = 128M    ; max. size of the memory allowed for a script
post_max_size = 8M     ; max. size of data transmitted via POST method
default_mimetype = "text/html" ; default MIME type for the content generated by a script
file_uploads = On      ; file uploads are permitted
upload_max_filesize = 32M ; max. size of an uploaded file
session.use_cookies = 1 ; Web sessions will be stored in cookies
session.name = PHPSESSID ; default cookie name regarding Web sessions
...
; specifying the extensions loaded during the PHP server application initialization
extension_dir = "./php/extensions" ; directory including extensions
extension=curl ; HTTP + other Internet protocols using libcurl library
extension=pdo_sqlite ; support for SQLite via PDO (PHP Data Objects)
extension=mysqli ; support for MySQL
...
```

various behaviors of the PHP platform, including loaded extensions (.so/.dll shared libraries), can be globally configured by using the [php.ini](#) file

PHP - Characteristics

PHP program execution behavior can be adjusted via declare directive – possibly, for a block of code

www.php.net/manual/en/control-structures.declare.php

```
// charset encoding used for generated content  
declare (encoding='UTF-8');
```

```
// strict datatype checking – PHP 7+  
declare (strict_types=1);
```

PHP - Characteristics

User interaction:

retrieving the values entered into Web form fields

cookies

sessions

user authentication

access to global variables – created “on the fly”

PHP - Characteristics

Features regarding Web technologies:

URL processing

HTTP support – including cURL

 caching via memcached

Web services development – e.g. REST APIs

...and others

PHP - Characteristics

Support for database access:

abstract layers

DBAL (DataBase Abstraction layer)

iODBC (Independent Open DataBase Connectivity)

PDO (PHP Data Objects)

www.phptherightway.com/#databases_abstraction_layers

PHP - Characteristics

Support for database access:

specific to a database server

relational: DB2, MySQL, Oracle, PostgreSQL, SQLite,...

NoSQL – e.g., MongoDB

consult www.phptherightway.com/#databases

PHP - Characteristics

Resource content processing:

audio files – via various libraries: ktaglib, oggvorbis, etc.

bzip2, LZF, RAR, ZIP, ZLIB archives

PDF documents

graphical content in various formats

JSON files

XML documents – creating, processing, validating, etc.

information regarding credit cards

...

PHP - Characteristics

Support for Internet + system resources:

file systems, including FTP
processes – with Libevent, PCNTL, pthreads,...
event processing – via Event
sockets
e-mail – e.g., IMAP, POP3

...and many others

PHP as procedural programming language

PHP - Characteristics

Procedural programming

paradigm based on procedure calls (routines, functions) containing a series of steps for making the calculations

procedural languages are imperative, using instructions (commands) that change the state of the program

examples: FORTRAN (1954), ALGOL (1958),
BASIC (1964), Pascal (1970), C (1972), Ada (1978)

PHP – Data Types: scalars

boolean

true or false

PHP – Data Types: scalars

int

integer values specified in...

10 (decimal)	3734
16 (hex)	0xE96
8 (octal)	07226
2 (binary)	0b111010010110

PHP – Data Types: scalars

float

real numbers

usually, conforming to IEEE 754 (double precision)

the size of the representation is platform dependent
(the precision can be adjusted in the config file php.ini)

www.php.net/manual/en/language.types.float.php
floating-point-gui.de

PHP – Data Types: scalars

string

ASCII character strings

(starting with PHP 7, native Unicode support exists)

www.php.net/manual/en/language.types.string.php

PHP – Data Types: scalars

string

ASCII character strings

(starting with PHP 7, native Unicode support exists)

escape characters can be used, like

`\n` (New Line) `\r` (Carriage Return) `\t` (Tab)

`\\` (Backslash) `\$` (Dollar) `\"` (Double Quote) `\'` (Quote)

PHP – Data Types: scalars

string

common delimiters:

" or '

PHP – Data Types: composed

array

association between values (of any type)
and keys (of integer or string types)

www.php.net/manual/en/language.types.array.php

PHP – Data Types: composed

array

there is no clear distinction
between indexed and associative arrays

an array could represent various data structures:

list (vector), associative array - hash (implementation of an association of values - mapping), dictionary, collection, stack, queue,...

PHP – Data Types: composed

array

```
// an indexed array (vector of values) – initial syntax
$gifts = array ("watch", "cookie", "necklace", "axe");

// associative array – <key, value> pairs
array ( "name" => "Tux", "size" => 17, "offer" => TRUE );

// simplified syntax (starting with PHP 5.4)
[ "name" => "Tux", "size" => 17, "offer" => TRUE ];
```

PHP – Data Types: special

null

specifies the `null` value
representing a variable that has no value

useful functions:

`is_null()`

`unset()`

PHP – Variables

Variables have names composed by letters, digits, and _ characters prefixed by the \$ symbol

can store values – having a specific data type – or references (specified with &)

www.php.net/manual/en/language.variables.php

PHP – Variables

```
$nume = 'Tux';
```

```
// different behavior regarding the substitution of the actual  
// value of a variable depending on the delimiters used  
// for strings
```

```
echo "Salut $nume!\n";  
echo 'Salut $nume!\n';
```

```
-> Salut Tux!  
    Salut $nume!\n
```

</> source code

```
1 <?php  
2 $nume = 'Tux';  
3  
4 echo "Salut $nume!\n";  
5 echo 'Salut $nume!\n';
```

input  Output

```
Success #stdin #stdout 0.02s 24312KB  
Salut Tux!  
Salut $nume!\n
```

online execution of PHP code via [Ideone](#) Web tool

PHP – Variables

Useful predefined functions:

`var_dump()`

`settype()`

`is_bool()`, `is_int()`, `is_float()`, `is_array()`, `is_string()`

`is_scalar()`, `is_numeric()`

...and others

PHP – Variables

Scope (variables visibility)

in order to be accessed in the entire program,
the variables should be declared as **global**

php.net/manual/en/language.variables.scope.php

PHP – Variables

```
$score = 33;
```

```
function getScore () {  
    echo "Current score: " . $score;  
}
```

```
getScore();
```

- ▶ **Undefined variable:**
score in prog.php on line 4

```
$score = 33;
```

```
function getScore () {  
    global $score;  
    echo "Current score: " . $score;  
    // similar to $GLOBALS["score"]  
}
```

```
getScore();
```

- ▶ **Current score: 33**

PHP – Variables

advanced

Allocated memory is automatically freed
(garbage collection)

www.php.net/manual/en/features.gc.php

PHP – Predefined Variables

Variables available in the whole program
(superglobals)

`$GLOBALS[]`

an associative array containing references to
all variables defined as global

PHP – Predefined Variables

Web server
specific data

`$_SERVER[]`

`$_GET[]` `$_POST[]` `$_FILES[]` `$_REQUEST[]`

data about the
Web session

`$_SESSION[]`

client specific
HTTP requests

`$php_errormsg`

reported
error message

...

PHP – Constants

Specified with **define** ()

globally available in the program

```
define ( string $name , mixed $value) : bool
```

```
define ("MIN_DIMENS", 13);
```

www.php.net/manual/en/function.define.php

PHP – Predefined Constants

advanced

Examples:

```
PHP_VERSION  
PHP_OS  
PHP_EOL  
PHP_INT_MAX  
PHP_INT_SIZE  
DIRECTORY_SEPARATOR  
true  
false  
null
```

PHP – Predefined Constants

advanced

Error reporting control:

E_ERROR	fatal errors (script execution is ended)
E_WARNING	warnings
E_PARSE	code processing errors (parsing)
E_NOTICE	notifications at runtime
E_STRICT	suggestions on code improvement
E_DEPRECATED	notifications about deprecated aspects

www.php.net/manual/en/errorfunc.constants.php

www.phptherrightway.com/#errors_and_exceptions

PHP – Predefined Constants

advanced

Runtime environment offers access to “magical” constants

their values can be used in a given program

__LINE__
__FILE__
__DIR__

PHP – Operators

arithmetic: + - * / % ++ --

value assignation: = and => (for arrays)

reference: &

comparisons: == === != <> !== < > <= >= ?: ?? <=>

logic: and or xor ! && ||

string (concatenation): . .=

...and many others

PHP – Operators

advanced

Starting with PHP 7, new operators can be used:

`<=>` (spaceship)

comparing expressions (of a scalar type),
and returning -1, 0 or 1

```
echo 15.5 <=> 15.5; // 0 (equality)
echo 15.5 <=> 16.5; // -1 (right value is greater)
echo 17.5 <=> 15.5; // 1 (left value is greater)
```

PHP – Operators

advanced

Starting with PHP 7, new operators can be used:

`??` (null coalescing)

offers the value of first operand if exists and it's not NULL,
otherwise returns the value of second operand

```
// as user name, we'll use the value provided by a Web form
// (retrieved with GET or POST);
// if it doesn't exist, the value is 'tux'
$username = $_GET['user'] ?? $_POST['user'] ?? 'tux';
```

PHP – Control Statements

`if`, `switch`, `while`, `do`, `for`, `break`, `continue`
similar to the C instructions

```
if (!$name) {  
    echo ("No name was given..");  
} else {  
    echo ("Welcome, " . $name . "!\n");  
}
```

PHP – Example

```
<?php
// filling up an array with values from 1 to 10
for ($index = 1; $index <= 10; $index++) {
    $values[$index] = $index;
}
// computing the sum of values
$sum = 0;
foreach ($values as $item)
    $sum += $item;
/* showing the obtained sum on the standard output
   to be sent to the Web client */
echo ("<p>Sum of first 10 numbers is <strong>" .
        $sum . "</strong>.</p>");
?>
```

PHP – Example

Invoking (running) the PHP program from the command line:
saving source code into a text file – **values.php**
calling the PHP interpreter from the command line interface

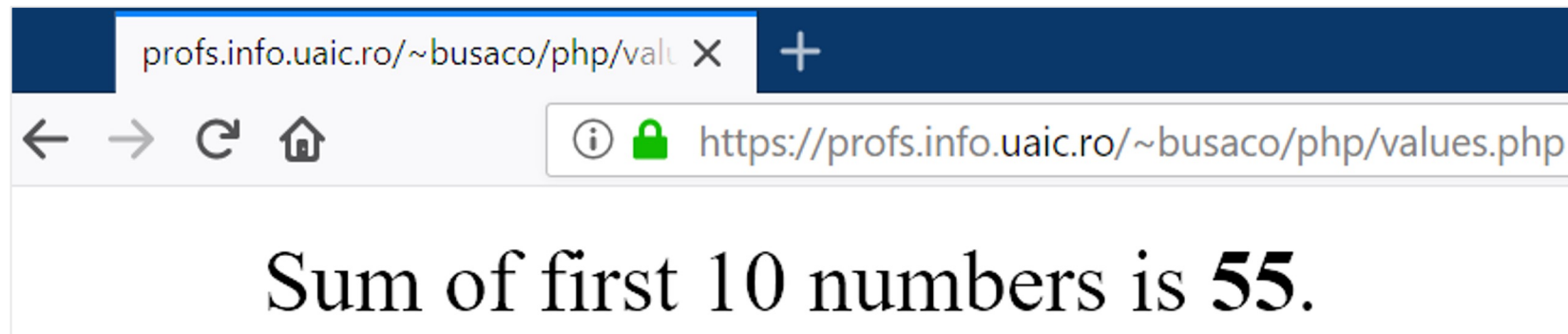
```
php values.php
```

```
<p>Sum of first 10 numbers is <strong>55</strong>.</p>
```

PHP – Example

Invoking (running) the PHP program on the Web server:
placing the source code – having read and execution rights

in the browser, specifying the URL to the program
to be invoked via the GET method of the HTTP protocol



result generated
by the script

PHP – Control Statements

Including source code from other files
(support for modularization)

include

searches the source file in predefined directories specified via
`include_path` and evaluates it

if the file does not exist, a warning is generated

include_once – to include it only once

PHP – Control Statements

Including source code from other files
(support for modularization)

require

searches the source file in predefined directories specified via
`include_path` and evaluates it

if the file does not exist, a fatal error is generated

require_once – to include it only once

PHP – Functions

User defined functions:

```
function sendMessage ($to="", $from="", $subject="Web") {  
    // body...  
}
```



parameter with
default value

php.net/manual/en/language.functions.php

PHP – Functions

```
define ('MAX', 10);           // maximum number of values

function square ($number) {  // a function to square a number
    return $number * $number;
}

$number = 0;
while ($number < MAX) {
    $number++;                // incrementing...

    if ($number % 2)         // is odd number...
        continue;           // continuing to next iteration
    // it is an even number, so we show its square
    echo "Square $number is " . square ($number) . "\n";
} // end of while
```

PHP – Functions

User defined functions:

a function name is case-insensitive

parameters can be specified by reference – prefixed by **&**

a variable number of parameters is indicated by ...

php.net/manual/en/functions.arguments.php

PHP – Functions

```
<?php  
declare (strict_types=1);
```

```
// arguments should be integers, returned value must be of integer type
```

```
function add (int ...$numbers): int {  
    $sum = 0;  
    foreach ($numbers as $number) {  
        $sum += $number;  
    }  
    return $sum;  
}
```

```
echo add (7, 3, 74, 1);  
echo add (pi (), '?');  
>
```

85

Fatal error: Uncaught TypeError: Argument 1 passed to add() must be of the type integer, float given
Next TypeError: Argument 2 passed to add() must be of the type integer, string given

PHP 7+: data type can be specified for each argument and for the value returned by the function (scalar type declarations)

PHP – Predefined Functions

mathematical & conversion
character string manipulation
array processing
access to resources and file processing
database manipulation
regarding network connections
cryptographic
XML, PDF, JPEG,... resource processing
operating system specific
general purpose

details at php.net/manual/en/funcref.php

PHP

advanced

Embedded Web server

run with:

```
php -S localhost:8000 -t phpwebapp/
```

www.php.net/features.commandline.webserver

How about the features regarding Web interaction?

PHP – Web Interaction

Data transmitted by the client (browser) is stored into (global) predefined associative arrays:

`$_GET[]` – data transmitted via GET

`$_POST[]` – data transmitted via POST

`$_COOKIE[]` – received cookies

`$_REQUEST[]` – data retrieved from client
(content of **`$_GET`**, **`$_POST`** and **`$_COOKIE`**)

`$_SESSION[]` – session data

PHP – Web Interaction

Other useful global variables:

`$_SERVER[]`

offers info about the Web server

`$_SERVER['PHP_SELF']` indicates the PHP script name

`$_SERVER['REQUEST_METHOD']` – used HTTP method

`$_SERVER['HTTP_REFERER']` – URL referring the resource

`$_SERVER['HTTP_USER_AGENT']` – client details

www.php.net/manual/en/reserved.variables.server.php

PHP – Web Interaction

Other useful global variables:

`$_ENV[]` – data provided by the runtime environment

`$_FILES[]` – data about the uploaded files

www.php.net/manual/en/features.file-upload.php

PHP – Web Interaction

```
<!-- a Web form modeled in HTML -->  
<form action="display.php" method="post">  
    <input type="text" name="name" />  
    <input type="text" name="age" />  
    <input type="submit" value="Send" />  
</form>
```

PHP – Web Interaction

```
<!-- a Web form modeled in HTML -->
<form action="display.php" method="post">
  <input type="text" name="name" />
  <input type="text" name="age" />
  <input type="submit" value="Send" />
</form>

<?php
// display.php - a program invoked via POST
if (!$_REQUEST["name"])
  echo ("Name is not specified!");
else
  echo ("Name is" . $_REQUEST["name"]);
?>
```



each name field in the form represents a key in **\$_REQUEST []** associative array (depending on the HTTP method, it could be consulted by using **\$_GET** or **\$_POST**)

PHP – Cookies: creation

- 🍪 Setting a cookie via `setcookie ()` function

```
setcookie ( string $cookie_name  
[, string $value = "" [, int $expiration_time = 0  
[, string $path = "" [, string $domain = ""  
[, bool $secure_transf = false  
[, bool $just_http = false ]]]]] ) : bool
```

PHP – Cookies: creation

- 🍪 Setting a cookie via `setcookie ()` function

```
setcookie ( string $cookie_name  
[, string $value = "" [, int $expiration_time = 0  
[, string $path = "" [, string $domain = ""  
[, bool $secure_transf = false  
[, bool $just_http = false ]]]]] ) : bool
```

```
// persistent - set that the cookie will expire in 10 days  
setcookie ('color', 'tan', time() + 60 * 60 * 24 * 10);  
// non-persistent  
setcookie ('authenticated', 'true');
```

why?

PHP – Cookies: expiration




Deleting a cookie:
cancel value and time;
eventually, also other cookie attributes

```
setcookie ($cookie_name, "", 0, "/", "");
```

www.php.net/manual/en/function.setcookie.php

PHP – Cookies: access


 A cookie is specified (accessed) like a variable of type associative array – `$_COOKIE ['cookie_name']`

```
<!-- the background color is given by the previously set
      cookie value -->
<style>
    body {
        background: <?php echo $_COOKIE[ 'color' ]; ?>
    }
</style>
```

PHP – Web Sessions

Session management: `session_start()`, `session_register()`, `session_id()`, `session_unset()`, `session_destroy()` functions

```
session_start (); // start a Web session
if (!isset ($_SESSION['visits'])) {
    $_SESSION['visits'] = 0;
} else {
    $_SESSION['visits']++;
}
```



the variable
`visits` attached
to the session

www.php.net/manual/en/book.session.php

Useful tools for Web developers?

Tools - Frameworks

advanced

Features:

MV* and various design patterns

database access (ORM, DAO, ActiveRecord,...)

input data validation and filtering

authentication + access control

cookie & session management

Tools - Frameworks

Features:

data presentation – templating

performance – i.e. caching

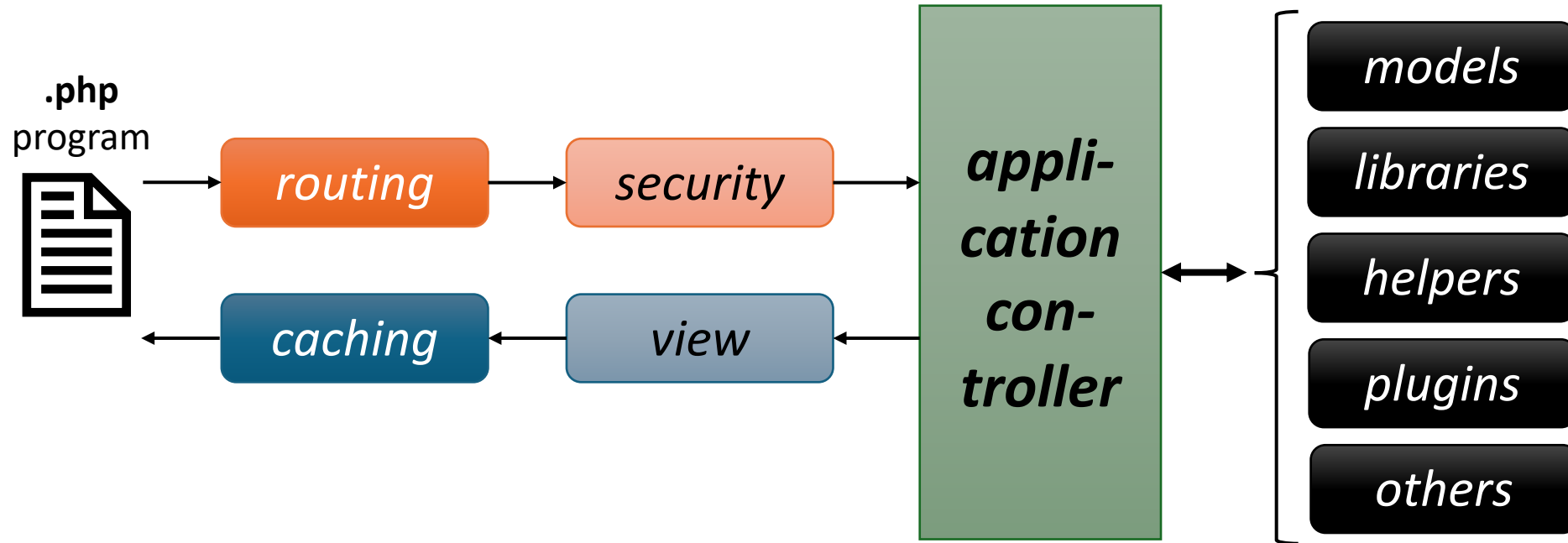
asynchronous data transfer (Ajax, WebSocket)

support for Web services and REST APIs

extensibility – e.g., user-defined modules, managed with the
[Composer](#) tool

Tools - Frameworks

advanced



workflows performed by a framework

Tools - Frameworks

CakePHP – cakephp.org

CodeIgniter – codeigniter.com

FuelPHP – fuelphp.com

Laravel – laravel.com

Nette – nette.org

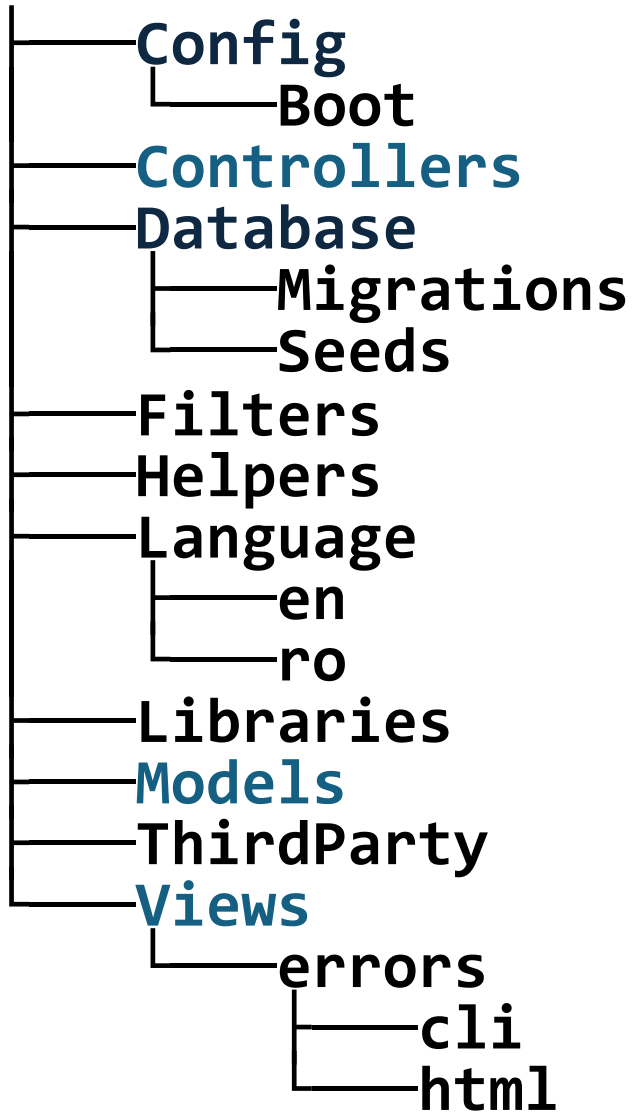
Phalcon – phalcon.io

PRADO – www.pradoframework.net

Symfony – symfony.com

Yii – www.yiiframework.com

Tools - Frameworks



directory structure of
a Web application developed
by using an MVC framework

CodeIgniter 4

codeigniter.com/user_guide/

Tools - Frameworks

Some frameworks like **Laravel** can benefit from the facilities offered by a virtualization platform like **Docker**

packaging an (Web) application and its dependencies in a virtual container that can run on any operating system installed on a host (on premises) and/or configured by a vendor – usually, in cloud

laravel.com/docs/
laracasts.com

Tools – Micro-frameworks

advanced

A micro-framework represents
a minimalist Web framework

Tools – Micro-frameworks

A micro-framework represents
a minimalist Web framework

does not include complex features

often, is focused on a single aspect regarding
Web development – e.g., creating an API, microservice,...

Tools – Micro-frameworks

advanced

Fat-Free Framework (F3)

fatfreeframework.com

Flight

flightphp.com

Fresh Squeezed Limonade

github.com/yesinteractive/fsl

Leaf PHP

leafphp.dev

Slim

www.slimframework.com

Tools – Packages

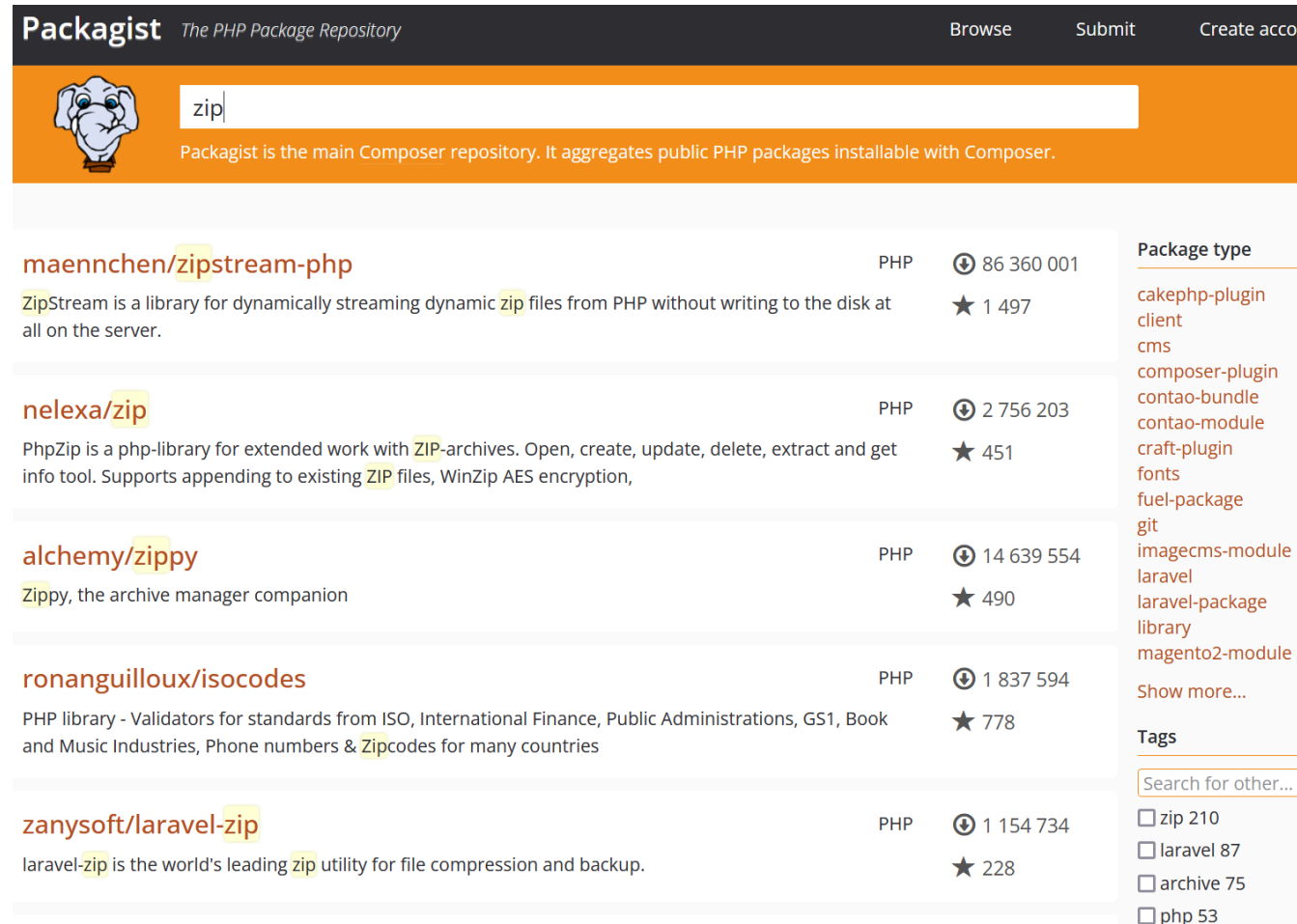
Managing dependencies
between libraries and packages

Composer
getcomposer.org

www.phptherightway.com/#dependency_management

Tools – Packages

advanced



The screenshot shows the Packagist website interface. At the top, there's a navigation bar with 'Packagist The PHP Package Repository', 'Browse', 'Submit', and 'Create account'. Below this is a search bar containing 'zip' and a description: 'Packagist is the main Composer repository. It aggregates public PHP packages installable with Composer.' The main content area displays a list of search results for 'zip'.

Package Name	Language	Downloads	Stars	Package Type
maennchen/zipstream-php	PHP	86 360 001	1 497	cakephp-plugin, client, cms, composer-plugin, contao-bundle, contao-module, craft-plugin, fonts, fuel-package, git, imagecms-module, laravel, laravel-package, library, magento2-module
nelexa/zip	PHP	2 756 203	451	
alchemy/zippy	PHP	14 639 554	490	
ronanguilloux/isocodes	PHP	1 837 594	778	
zanysoft/laravel-zip	PHP	1 154 734	228	

Additional features visible in the screenshot include a 'Tags' section with a search input and a list of tags: zip 210, laravel 87, archive 75, and php 53.

public packages can be downloaded from the Web from
Packagist – package repository: packagist.org

Tools – Packages

advanced

Dependency management
between libraries and packages

alternative – older:

PEAR (PHP Extension and Application Repository)

pear.php.net

+

extensions offered by third parties:

PECL (PHP Extension Community Library)

pecl.php.net

Tools – Development Environments

Pre-configured Web development environments

Web server + PHP + database server(s) + tools

Tools – Development Environments

Apache, PHP 8, MySQL/MongoDB...

AMPPS

www.ampps.com

XAMPP

www.apachefriends.org

with support for various configurations of Web systems like Drupal, Joomla!, MediaWiki, Magento, Moodle, WordPress

Tools – Documenting

advanced

Automatic documentation generators

Daux – daux.io

phpDocumentor – www.phpdoc.org

phpDox – phpdox.net

github.com/ziadoz/awesome-php#documentation

Tools – Code Analysis

advanced

PHP source-code static analysis

for discovering various programming bugs, verifying the conformance to various coding standards, automatic correction (fixers), determining code metrics: complexity, number of lines,...

github.com/exakat/php-static-analysis-tools

Tools – Code Analysis

advanced

PHP Standards Recommendations – www.php-fig.org/psr/
written by PHP FIG (Framework Interop Group)

PSR1: *Basic Coding Standard*, PSR3: *Logger Interface*,
PSR4: *Autoloading*, PSR6: *Caching Interface*,
PSR7: *HTTP Message Interface*, PSR11: *Container Interface*,
PSR12: *Extended Coding Style Guide*,
PSR13: *Hypermedia Links*, PSR14: *Event Dispatcher*,
PSR15: *HTTP Handlers*,...

tools: phan, PHP_CodeSniffer, PHP-CS-Fixer, Rector

Tools – Code Analysis

advanced

PSR12: Extended Coding Style Guide – example rules:

All PHP files MUST use the Unix LF (linefeed) line ending only.

The closing ?> tag MUST be omitted from files containing only PHP.

Lines SHOULD NOT be longer than 80 characters.

There MUST NOT be more than one statement per line.

All PHP reserved keywords and types MUST be in lower case.

Visibility MUST be declared on all properties and methods.

There MUST be one space after the control structure keyword.

www.php-fig.org/psr/psr-12/

Tools – Major Critic 🥲

advanced

In PHP, we can easily create applications that “adopt” the **Spaghetti Code** anti-pattern

study Unix Sheikh et al., *PHP The Wrong Way*, 2023
phpthewrongway.com

Tools – Major Critic 🥲

advanced

```
<h1>My Users <a class="btn btn-primary" href="new_user.php" role="button">New User</a></h1>
<table class="table table-condensed">
  <tr>
    <th>Id</th>
    <th>Name</th>
    <th>Age</th>
    <th>Email</th>
    <th>Action</th>
  </tr>
  <?php
```

HTML
Web
interface

PHP for data
access and
processing

PHP for
data presentation

```
// Get all users
$stmt = $dbh->prepare("SELECT * FROM users");
$stmt->setFetchMode(PDO::FETCH_ASSOC);
if ($stmt->execute()) {
    while ($row = $stmt->fetch()) {
```

```
        ?>
        <tr>
          <td><?php echo $row['id']; ?></td>
          <td><?php echo $row['name']; ?></td>
          <td><?php echo $row['age']; ?></td>
          <td><?php echo $row['email']; ?></td>
```

```
        <td>
          <div class="btn-group" role="group" aria-label="...">
            <a href="edit.php?id=<?php echo $row['id']; ?>" class="btn btn-default btn-sm">Edit</a>
            <a href="index.php?delete=<?php echo $row['id']; ?>" class="btn btn-default btn-sm">Delete</a>
          </div>
        </td>
      </tr>
    <?php
```

github.com/hanafiah/masterphp/tree/master/spaghetti

A few case studies of Web applications developed in PHP?

Case Study – Wikipedia

The image shows a screenshot of the Wikipedia article titled "World Wide Web". The page layout includes the Wikipedia logo and search bar at the top, a table of contents on the left, and the main article text. Annotations are present: a grey box labeled "Tim" points to a screenshot of a web browser interface; a grey box labeled "can be" points to the text "A web application is application software that is created with web technologies and runs via a web browser"; a grey box labeled "ough" points to the text "Web applications emerged during the late 1990s and allowed for the server to dynamically build a response to the request, in contrast to static web pages"; a grey box labeled "kup" points to the text "Web applications are web pages that function as application software"; a grey box labeled "plex user" points to the text "The information in the Web is transferred across the Internet using HTTP"; and a grey box labeled "resources." points to the text "Multiple web resources with a common theme and usually a common domain name make up a website".

WIKIPEDIA
The Free Encyclopedia

Search Wikipedia

World Wide Web

150 languages

Article Talk

Read View source View history Tools

From Wikipedia, the free encyclopedia

This article is about the global system of pages accessed via HTTP. For the worldwide computer network, see [Internet](#). For the web browser, see [WorldWideWeb](#).

"WWW" and "The Web" redirect here. For other uses, see [WWW \(disambiguation\)](#) and [The Web \(disambiguation\)](#).

The **World Wide Web** (**WWW** or simply **the Web**) is an [information system](#) that [content](#) sharing over the [Internet](#) through user-friendly ways meant to appeal to beyond IT specialists and hobbyists.^[1] It allows documents and other [web reso](#) be accessed over the Internet according to specific rules of the [Hypertext Trans Protocol](#) (HTTP).^[2]

The Web was invented by English computer scientist [Tim Berners-Lee](#) while at 1989 and opened to the public in 1993. It was conceived as a "universal linked information system".^{[3][4][5]} Documents and other media content are made avail accessed by programs such as [web browsers](#). Servers and resources on the W character strings called [uniform resource locators](#) (URLs).

The original and still very common document type is a [web page](#) formatted in H language supports [plain text](#), [images](#), embedded [video](#) and [audio](#) contents, and interaction. The HTML language also supports [hyperlinks](#) (embedded URLs) w

[Web navigation](#), or web surfing, is the common practice of following such hyperlinks across multiple websites. [Web applications](#) are web pages that function as [application software](#). The information in the Web is transferred across the Internet using HTTP. Multiple web resources with a common theme and usually a common [domain name](#) make up a [website](#). A single web server may provide multiple websites, while some websites, especially the most popular ones, may be provided by multiple servers. Website content is provided by a myriad of companies, organizations, government agencies, and [individual users](#); and comprises an enormous amount of educational, entertainment, commercial, and government information.

The Web has become the world's dominant [information systems platform](#).^{[6][7][8][9]} It is the primary tool that billions of people worldwide use to interact with the Internet.^[2]

Tim

can be

ough

kup

plex user

resources.

founders: Jimmy Wales & Larry Sanger (2001)

Case Study – Wikipedia

Aim: providing open content by using
a group of collaborative Web applications – wikis

Case Study – Wikipedia

Aim: providing open content by using
a group of collaborative Web applications – wikis

Wikipedia Foundation

also maintains Wikipedia, Wiktionary, Wikibooks, Wikiquote,
Wikivoyage, Wikisource, Wikimedia Commons, Wikispecies,
Wikinews, Wikiversity, Wikidata

en.wikipedia.org/wiki/Wikimedia_Foundation

Case Study – Wikipedia

advanced

Complex version of the LAMP technology stack

MediaWiki - *wiki* system used for all services
implemented in PHP (8.1+ versions) + JavaScript

MariaDB (main storage solution; alternatives: **SQLite**, **PostgreSQL**)

ImageMagick, **DjVu**, **TeX**, **rsvg**, **ploticus**, etc.

(for graphical content processing within MediaWiki)

Apache HTTP Server + **NGINX** (Web servers)

also provides an API to be used by Web developers:

www.mediawiki.org/wiki/API:Main_page

Case Study – Wikipedia

advanced

PHP-FPM (FastCGI Process Manager for PHP)

Apache Traffic Server + Varnish (proxy + caching)

Memcached (caching for database queries)

Elasticsearch (textual search –Java implementation)

gdnssd (C++ solution for DNS)

Apache Kafka (event streaming – Java + Scala)

Linux Virtual Server (load balancing); **PyBal** (local monit.)

Debian GNU/Linux (OS)

Icinga (monit. the state of the systems): www.wikimediastatus.net

Phabricator (bug tracking)

meta.wikimedia.org/wiki/Wikimedia_servers

Case Study – Wikipedia

Various Web sites use Content Management Systems
(CMS) developed in PHP

general:

Drupal, Joomla, WordPress, etc.

Case Study – Wikipedia

Various Web sites use Content Management Systems
(CMS) developed in PHP

wiki:

DokuWiki, MediaWiki, pmWiki, etc.

Case Study – Wikipedia

Various Web sites use Content Management Systems
(CMS) developed in PHP

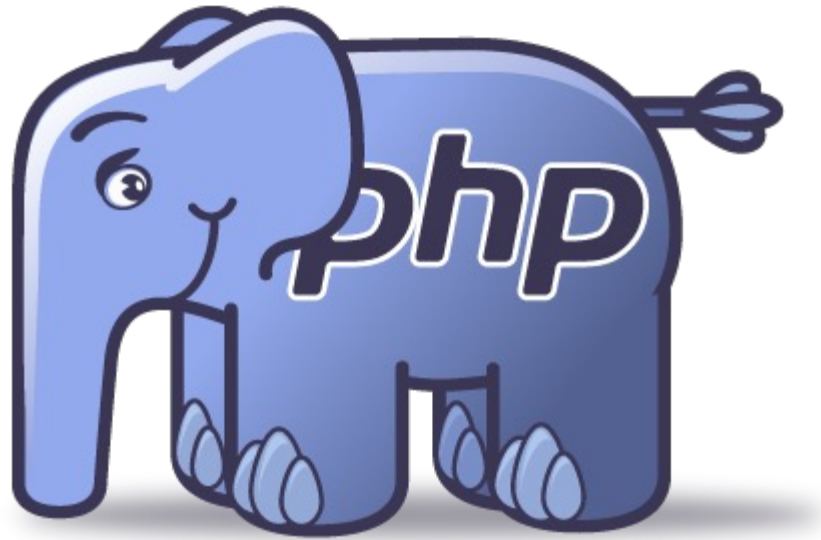
for e-commerce:

Magento, OpenCart, PrestaShop, etc.

Case Study – Wikipedia

Various Web sites use Content Management Systems
(CMS) developed in PHP

facilitating online discussions (message board, Web forum):
bbPress, esoTalk, phpBB, etc.



Live PHP code examples

Summary