Faculty of Science,
Technology
and Medicine

Web Programming

Volker Müller
University of Luxembourg

General Information

Lecture given every Monday, 10:30 – 12:00, no explicit exercise session, but a lot of practical homework

I will inform you on Moodle whether we will have remote sessions or hybrid teaching mode – might change at any time

Grading

At least 4 “small” exercises (duration 1-2 weeks) – 10% each (there also might be bonus exercises)

1 final “large” assignment (duration at least 3 weeks) – 40%

1 final (remote??) exam – 20%

Docker

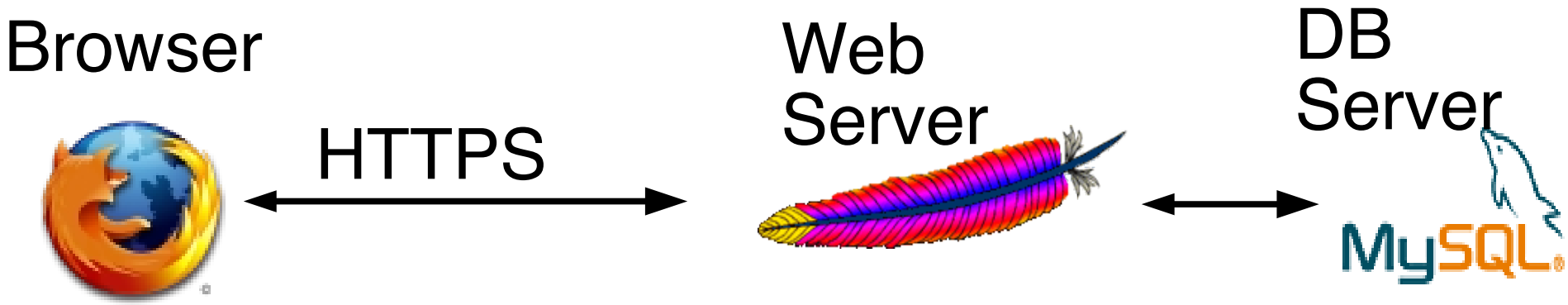
All examples and assignments will use Docker

Docker is a lightweight software containerization platform

Allows developers, sys-admins etc. to easily deploy their applications in a sandbox (“container”)

I will give you a practical introduction now

A Traditional Web Application – Three Tiers



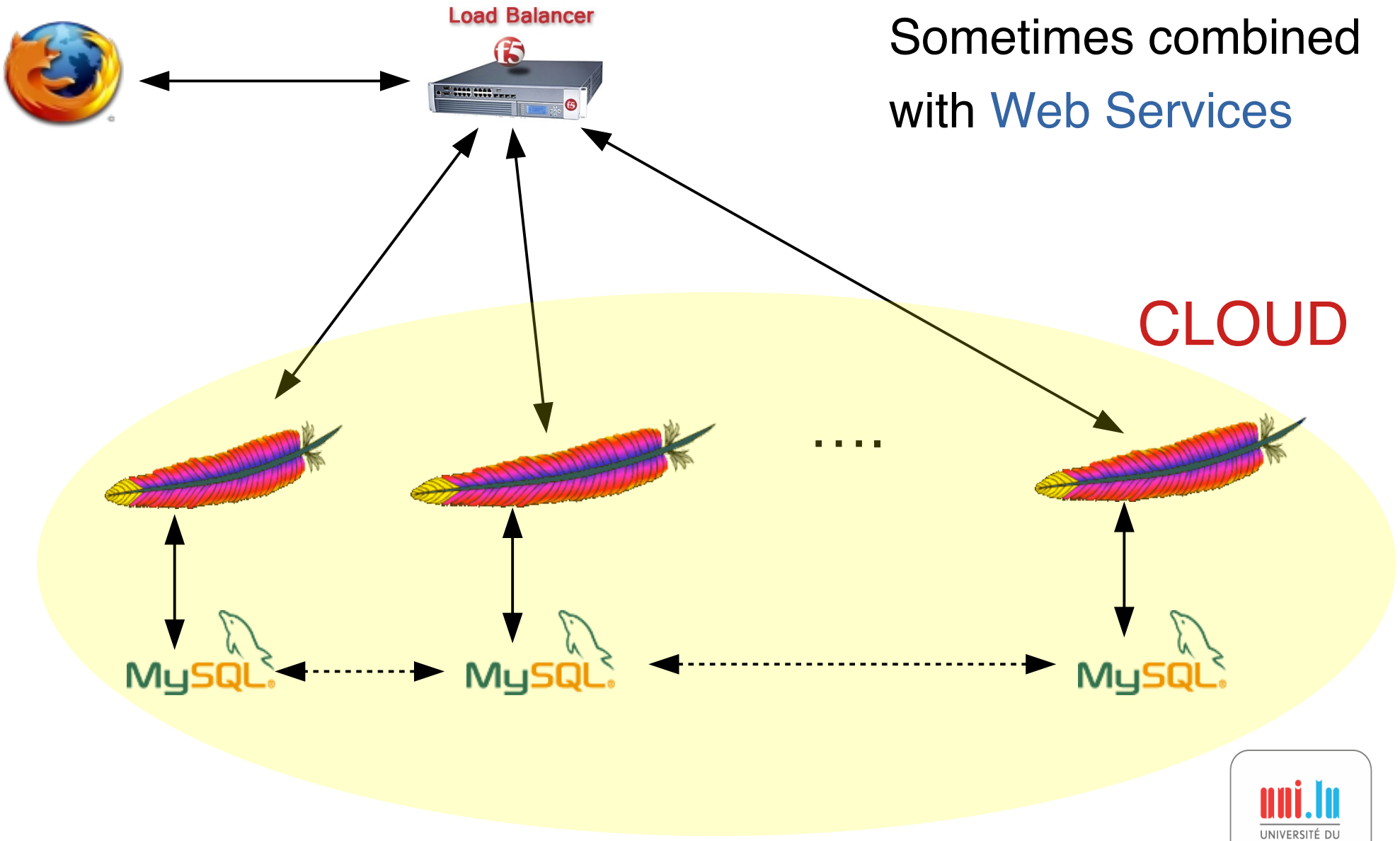
Javascript
Typescript
Flash
Silverlight

ASP / .NET (C#)
PHP
Java Servlets / JSP / JSF

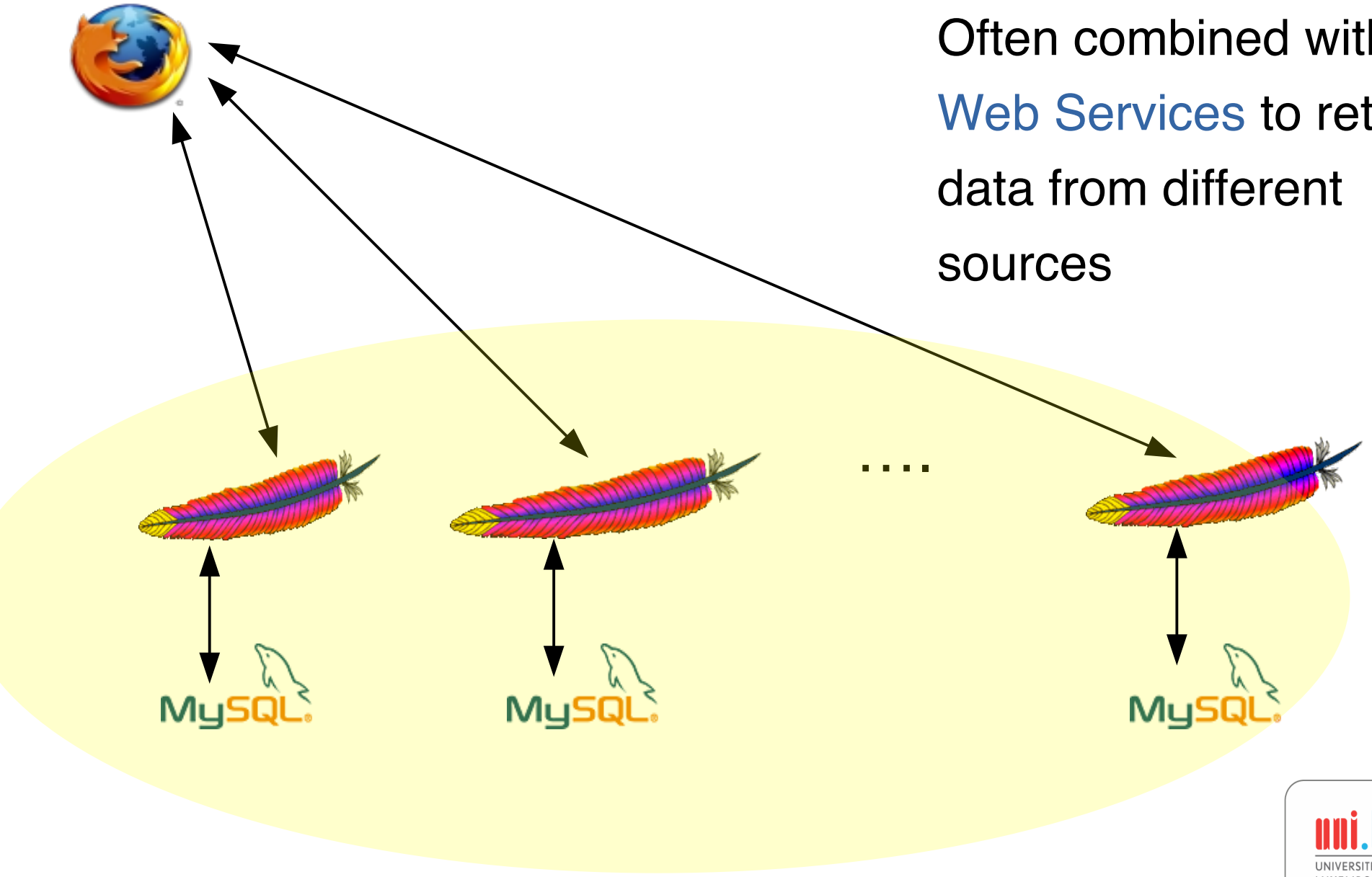
Client-side scripting

Server-side scripting

More Popular Cloud-Based Architecture



Also common: Many Data Sources



Often combined with Web Services to retrieve data from different sources

Web Services

Software system designed to support interoperable machine-to-machine interaction over a network

Often working over HTTP (port 80) or HTTPS (443)

Clients and servers exchange messages in XML format (SOAP – "old fashioned") or some other text-based format (JSON)

Examples later and in course “Java for Enterprise Applications”

Other Popular Technologies

Client-Side: Javascript-based Frameworks (JQuery, React.js, Vue.js, ...)

Server-Side: Javascript-based server (Node-JS), PHP-Frameworks (Symfony), Python with Django, Go, Java-Framework SpringBoot

Database: MariaDB, PostgreSQL, No-SQL DB like Cassandra, MongoDB, Redis

HTML 4.01 (1999)

Publishing language of the World Wide Web

Specification available at http://w3.org

Uses predefined tags to define page & element layout, no programming language

“Special tags”:

 ...

 ...

XHTML (2002, 2010)

Reformulation of HTML 4 as XML 1.0 application

Main differences to HTML:

- documents must be well-formed
- elements and attributes in lower case
- end tags required
- attributes must be quoted and must have value

HTML5 / XHTML5

"Next generation HTML", published on 28.10.2014

Main changes:

- Adds new tags and attributes: <figure>, <audio>, <video>, <progress>, <canvas>...
- Introduces large number of new JS APIs that help in creating Web applications: API for playing of video and audio, offline Web apps, Editing API, Drag & drop API, history, ...

Latest revision: HTML5.2, December 2017

Tutorials on HTML5

W3 School:

http://www.w3schools.com/html/html5_intro.asp

HTML5-Tutorial:

http://www.html-5-tutorial.com

Tutorialspoint:

http://www.tutorialspoint.com/html5

Specification: http://www.w3.org/TR/html5/

Cascading Style Sheets (CSS)

Style Sheet Language that allows authors to attach styles to structured documents (html, xml)

Separation of presentation style (css) and content (html)

CSS 2.1 published in June 2011

New version CSS 3 defined in different modules (new selectors, new combinator, new pseudo-elements), some finished, some not yet

Browsers (unless very old) support CSS 3

CSS (II)

CSS file = list of formatting rules

Each rule consists of one or more selectors and a declaration block

```
a.local { color: green; }
```

Pseudo-classes can define further behavior:

```
:hover :first-line :first-letter :link :visited
```

Absence of expressions / variables → CSS pre-processors exist: less (http://lesscss.org), sass (http://sass-lang.com)

CSS Frameworks

Many frameworks exist providing predefined (responsive) layouts which simplify web design significantly:

- Bootstrap (http://getbootstrap.com/)
- Foundation (http://get.foundation)
- Skeleton (http://getskeleton.com)
- Yaml (http://www.yaml.de) .....

Interesting References for CSS 3

Tutorials:

http://www.w3schools.com/css/css3_intro.asp

http://www.css3-tutorial.net

150 Amazing CSS3-JS Effects:

https://1stwebdesigner.com/css-effects/

CSS3-JS Animations:

http://www.minimamente.com/example/magic_animations/

Helpful Tools

Hundreds of WYSIWYG HTML editors available, but some produce “bad” code (lots of unnecessary or even empty tags)

Firefox/Chrome extension for web development:

Web Developer – Page Inspector, Web Console, JS Debugger, ...

HTML and CSS can be checked for correctness:

http://validator.w3.org

http://jigsaw.w3.org/css-validator/

Responsive Web Design

Web design approach aims at crafting sites to provide an optimal viewing experience across wide range of devices

Uses combination of CSS and Javascript

CSS media query:

```
@media screen and (min-width:500px) { ... }

@media tv and (min-width: 700px) and (orientation: landscape) { … }
```

Web Application Performance

CSS (and Javascript) have a large impact on loading time, since rendering blocked until complete CSS code received

→ Define media types in HTML (<link href = "o.css" rel = "stylesheet" media = "(min-width: 40em)">)

→ Minify and cleanse CSS (many tools exist)

More info:
https://developers.google.com/web/fundamentals/performance

PHP: Hypertext Preprocessor

HTML is static, no programming features

PHP = widely-used Open Source general-purpose scripting language

PHP code embedded in HTML

Syntax very similar to Perl (C, Java)

Documentation at www.php.net

PHP 7.x

New release since December 2015

Latest version: PHP 7.4.10 (3.9.2020)

Some changes compared to PHP 5.x:

- Better performance + less memory (up to -25%)
- Scalar Type Hints & Return Types
- “Spaceship” Operator <=>
- Null Coalescing Operator: $id = $_GET[‘id’] ?? null

Example: PHP embedded in HTML

```
<?php   if ($_POST["user"] != "vmueller")  {      ?>
        <b>You are not allowed to login. </b>
<?php exit ( );   }
          else {                                          ?>
                <b>Welcome. </b>
<?php   session_start ( );
          $_SESSION["login"] = true;   }      ?>
```

Example
PHP 5.x Function

```
function slotnumber ( )
 {
   srand( time() );
   for ($i = 0; $i < 3; $i++)
      {  $random = (rand() % 3);  $slot[$i] = $random;  }
   print("<td width=\"33%\">$slot[0]</td>");
   print("<td width=\"33%\">$slot[1]</td>");
   print("<td width=\"33%\">$slot[2]</td>");
....
}
```

Example: Typed functions in PHP 7.x

```
function sum (int $a, int $b) : int
```

```
function sum (… $numbers) : int
```

```
function sum (int … $numbers) : int
```

You can loop over arbitrary number of parameters with foreach loop

Usually wrong-typed input parameters casted to desired type

Strict mode forces error: declare (strict_types=1);

Sessions

Way to preserve certain data across subsequent accesses (note: HTTP is stateless !)

Session identified by session id (often stored in a cookie)

Session data can be stored in a file on the server / user-defined (e.g. in a DB) with a SessionHandler

Accessing Files

PHP can read / write files:

```
$fp = fopen ("test.txt", "r");

while (! feof ($fp))

   $buffer = fgets ($fp, 4096);

fclose ($fp);
```

fopen even works for URLs (usually read-only)

Regular Expressions

Way to describe a set of strings based on common characteristics shared by each string in the set

Combination of characters sets and quantifiers

Examples:

[abc]	= a OR b OR c
[a-z0-9]	= small character OR digit
a*	= a block with ≥ 0 a's

What can we do with REs?

Check whether a input string has a substring of a special form (defined by a RE)

Extract such substrings

Replace such substrings by something else

Some Tutorials for Regular Expressions

www.regular-expressions.info/tutorial.html

regexone.com

www.tutorialspoint.com/php/
php_regular_expression.htm

Character Sets (1)

[abc] a, b, or c (set of characters)

[^abc] any character except a, b, c (negation)

[a-zA-Z] a through z, OR A through Z (range)

[a-z&&[def]] d, e, or f (intersection, AND)

[a-z&&[^bc]] a to z, except for b, c (subtraction)

Character Sets (2): Simplifications

.	Any character
\d	A digit: [0-9]
\D	A non-digit: [^0-9]
\s	A whitespace character: [\t\n\x0B\f\r]
\S	A non-whitespace character: [^\s]
\w	A word character: [a-zA-Z_0-9]
\W	A non-word character: [^\w]

Escaped Characters

Characters which are used for defining REs can itself be represented by adding a backslash before:

RE for a dot	\.
RE for [	\[
RE for \	\\

Quantifiers

X | Y	X OR Y
X?	X, once or not at all
X*	X, zero or more times
X+	X, one or more times
X{n}	X, exactly n times
X{n,}	X, at least n times
X{n,m}	X, at least n, not more than m times

Boundaries

^	The beginning of a line or input (default)
$	The end of a line or input (default)
\A	The beginning of the input
\z	The end of the input

Greedy versus Reluctant

Input: RE .*b with input string aabaaaab

Greedy: find the longest substring that matches RE

Reluctant: find the shortest substring that matches RE

RE Quantifiers are per default greedy

To make then reluctant, add ? after quantifier

Using RE: preg_match (1)

Return whether input string contains substring of some form:

```
$subject = "abcdef";

$pattern = '/^def/';

echo preg_match ($pattern, $subject);
```

Note: Regular expressions given inside / /

Using RE: preg_match (2)

Can also be used to extract strings:

```
preg_match ( '/^http:\/\/([^\/]+)/i',
                    "http://www.php.net/index.html",
                    $matches);
```

matches[0] = complete string that matches
matches[i] = substr. in i-th bracket pair (i > 0)

Why are RE useful?

Input can be checked for correct form (e.g. email address)

Information can be extracted from sources:

- local files
- other web pages

Information can be changed on the fly

PHP and MySQL

MySQL is popular open-source Relational DBMS (http://www.mysql.com)

PHP provides (many) functions for directly using MySQL, i.e. connect to MySQL server, choose database, send query, access result of query, etc.

Old mysql methods have been replaced in PHP7 with improved framework mysqli

Improvements in mysqli

Object-oriented interface

Support for Persistent Connections

Support for Prepared Statements

Support for Multiple Statements

Support for Transactions

Enhanced debugging capabilities

Example: Connect to DB

```
$I = new mysqli ('localhost', 'USER', 'PWD', 'DB');

if ($I->connect_errno)  { die ('Could not connect'); }

    // HERE ARE THE QUERY COMMANDS

$I->close ( );
```

Persistent Connections

Persistent connection = connection between client process and database can be reused by client process, rather than being created and destroyed multiple times

Reduces overhead of creating fresh connections every time one is required, as unused connections are cached

Open persistent connection: prepend “p:” to hostname when connecting

Accessing Form Data

Users input information in HTML in a form:

```
<form action="t.php" method="post">
    <input type="password" name="pwd" />
    <button type="submit">Submit</button>
</form>
```

Key used in dictionary

Entered information accessible in t.php in dictionaries $_GET[ ], $_POST[ ], or $_REQUEST[ ] (depending on HTTP method)

Example: Check Login Info (Basic Idea)

```
// assume: form data in var. $account, $pwd given

$query = "SELECT * FROM loginTBL WHERE
account = '" . $account . "' AND pass = '" . $pwd . "' ";

If (! ($res = $l->query ($query)))
    die ("ERROR in query");

if ($res->num_rows == 0)
    print ("Login failed"); …..
```

NOTE: contains security flaws !!

Some Security Aspects (I)

Passwords should never be stored in cleartext in a database

Better: encrypted or salted hashed

Appropriate hash function password_hash available in PHP, several alternatives can be set in MySQL

Security (II): Example SQL Injection

```
$query = "SELECT * FROM loginTBL WHERE
account = '" . $account . "' AND pass = '" . $pwd . "' ";
```

Assume input: $account = vmueller

$pwd = test' or '1' = '1

Query becomes

```
SELECT * FROM loginTBL WHERE
    account = 'vmueller' AND pass = 'test' or '1' = '1'
```

Result: if “vmueller” is valid account, then at least one row returned, even if valid password not known

Measures against SQL Injections

Countermeasure: Always escape SQL special characters in a string before use

PHP function: mysqli::real_escape_string

Rule: Always apply this function to the parts of query based on user input (from forms, cookies, URL parameters, …)

Alternative (preferred): prepared statements, use stored procedures

Next Week

More on PHP and MySQL