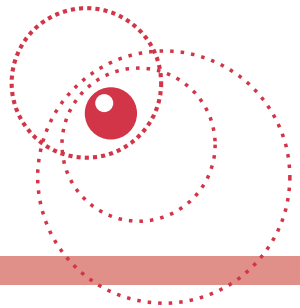

COMP7306: Web technologies

Server-side Web programming

6 February 2013

Pierre Senellart

TELECOM ParisTech

Outline

Outline

Web forms

- Allow interaction with the user through the input of information
- In HTML: only the interface of the form
- Most of the work will be carried out by the script that will process the form submission, on the server side

<form> tag

- An HTML form is the content of a `<form>` tag
- Following attributes:

action	URL of the script that will process the form
method	HTTP method, either `"get"` or `"post"`
enctype	HTTP parameter encoding `"application/x-www-form-urlencoded"` (by default) or `"multipart/form-data"`

Example (Elementary form)

```
<form action="action.php" method="get">
  <div><input type="submit"></div>
</form>
```

Field sets

In HTML, impossible to put form fields (inside elements) directly inside a `<form>`. They have to be put inside blocks first:

- Inside `<p>` tags if form fields are naturally inside text paragraphs (rare)
- Inside `<fieldset>` to regroup semantically close form fields
- Inside `<div>` blocks without any specific semantics otherwise

Example (Field set)

```
<fieldset>
  <legend>Size</legend>
  <input type="text" name="height">
  <input type="text" name="width">
</fieldset>
```

Labels

- Most fields are naturally accompanied by their labels (`<label>`).
- The label can be anywhere, usually directly to the left or to the right of the field
- Its `for` attribute references the `id` attribute of the corresponding field
- Inside a Web browser, clicking on the field label allows focusing on the field

Example (Label)

```
<label for="width">Width:</label>
<input type="text" name="width" id="width">
```

Outline

Input fields

- The `<input>` tag has a vast range of use in Web forms. It represents a form field
- The `type` attribute is the type (text, password, confirmation button, etc.) of field.
- The `name` attribute (name of the HTTP parameter) is compulsory (except for types `"reset"` and `"submit"`); it is used to specify to the server which parameter a value corresponds to

Example (Text field to enter comments)

```
<input type="text" name="comment">
```

Single-line text input

- `type ="text"` is used for inputting text fitting on a single line
- The `value` attribute can be used to specify the default value (optional)
- The maximum length of the string to input can be set with the `maxlength` attribute (optional)

Example

```
<input type="text" name="fname" value="John" maxlength="50">
```

Password input

- `type ="password"` is used for inputting a text whose characters should be masked: generally used for entering passwords. The password is still transmitted to the server in plain text!
- The `value` attribute can be used to specify the default value (optional)
- The maximum length of the string to input can be set with the `maxlength` attribute (optional)

Example

```
<input type="password" name="pwd" value="12345678">
```

Multiple choices among a list

- `type ="checkbox"` allows choosing several elements among a list
- Appears as boxes to check
- The returned value is necessarily given with the `value` attribute
- `checked ="checked"` specifies that the check box should be pre-checked
- In PHP, multiple choice input field parameter names should end with []

Example

```
<input type="checkbox" name="ad[]" value="site"
  checked="checked" id="ad-site">
<label for="ad-site">Receive offers from our site</label>

<input type="checkbox" name="ad[]" value="external"
  id="ad-external">
<label for="ad-external">Receive offers from other sites</label>
```

Single choice among a list

- `type ="radio"` allows choosing a single element among a list
- Displayed as radio buttons
- The returned value is necessarily given with the `value` attribute
- `checked ="checked"` specifies that a given radio button should be pre-selected

Example

```
Receive advertisements:
<input type="radio" name="ad" value="yes" id="ad-yes"
  checked="checked">
<label for="ad-yes">yes</label>

<input type="radio" name="ad" value="no" id="ad-no">
<label for="ad-no">no</label>
```

File upload

- `type ="file"` allows attaching a file to the form submission
- Because of the size of the request necessary to upload the file, one has to use the POST method and the `multipart/form-data` parameter encoding

Example

```
<label for="picture">Picture:</label>
<input type="file" name="picture" id="picture">
```

Hidden field

- `type ="hidden"` hides to the client some fields, but their content is still sent with the form
- It can be used to add some fixed information to the form, using the `value` attribute
- Attention: nothing prevents the client to send different values than the ones provided!

Example

```
<input type="hidden" name="currency" value="EUR">
```

Resetting a form

- `type ="reset"` provides a button that allows resetting the form fields to their default values
- The `value` attribute allows changing the text of the corresponding button

Example

```
<input type="reset" value="Clear everything">
```

Submitting a form

- `type ="submit"` provides a button for submitting the form
- The client sends the form at the URL provided in the `action` attribute of the `<form>` tag
- The `value` attribute allows changing the text of the corresponding button

Example

```
<input type="submit" value="Send">
```

Multiple-line text input

- For multiple-line text input, one uses the `<textarea>` tag
- The text inside this element is the default value of the field
- The closing tag is compulsory even if the field is empty
- The `rows` and `cols` attributes (compulsory) specify the height and width of the field, in number of characters

Example

```
<textarea name="bio" cols="40" rows="5">
Lorem ipsum dolor sit amet,
consectetur adipisicing elit
</textarea>
```

Selection menus

- The `<select>` tag describes a selection list:
  - The optional `size` attribute is the number of choices appearing simultaneously on the Web page. By default, 1
  - The `multiple ="multiple"` attribute allows multiple selections. In this case, in PHP, one always use a parameter name ending with `[]`.
- Choices are indicated with the `<option>` tag:
  - The `selected ="selected"` attribute specifies the default choice(s)
  - The `value` attribute is the parameter value associated with this choice

Example

```
<select name="age">
  <option value="20">under 20</option>
  <option value="35" selected="selected">21 to 35</option>
  <option value="50">36 to 50</option>
  <option value="51">over 51</option>
</select>
```

Outline

Role of server-side programs

- The Web is not just a set of static HTML documents
- Server-side programs allow:
  - processing form submissions;
  - displaying in a standard form the set of pages of a Web site;
  - proposing interactive applications;
  - the user to add or modify content on a Web site;
  - performing complex computation and queries and displaying the result as a Web page;
  - etc.

Programming languages

- CGI (Common Gateway Interface): normalized interface allowing the Web server software and a program running on the server to communicate;
- CGI makes it possible to use any programming language (compiled such as C, C++, Java, or interpreted such as Perl, Python, Ruby, etc.) to write server-side programs
- But some languages are more adapted to server-side programs:
  - include features dedicated to HTTP, HTML, etc.;
  - integrate more conveniently and more efficiently with the Web server software (via plugins);
  - have a syntax especially designed for Web programming, mixing HTML code sent as such and programming language instructions
- Whatever the technology used, the program code or binary is not available to the Web client, only the result of its execution

Server-side languages

PHP: one of the most popular languages, integrated very easily with Apache (free)

ASP and ASP.NET: meant to be used with IIS (Microsoft, commercial)

ColdFusion (Adobe, commercial)

JSP (Java Server Pages): allow mixing Java instructions and HTML code; requires a Java application server (e.g., Tomcat) in addition to Apache (Oracle, free)

Java servlets: true Java programs, better for complex applications on the server-side with few client interactions; requires a Java application server in addition to Apache (Oracle, free)

Web application frameworks

- Languages presented so far remain quite basic and general-purpose
- Do not necessarily encourage clear organization of the Web application code
- Framework: regroups programming languages, function library, external tools, guidelines to follow...
- Allows abstracting out Web application programming
- Usually follows the MVC paradigm (see next slide)
- Sometimes include client-side JavaScript generation to directly create a full dynamic Web application (e.g., form validation); AJAX integration as well
- Strongly recommended for complex applications... but also complex to master!

MVC paradigm

- Software engineering principle, used in other areas
- Especially well-suited to Web applications!
- Clean separation between:

  Model: data manipulated by the application and data manipulation function; reusable for other (Web) application that follow the same business logic

  View: data presentation; easily interchangeable to modify the appearance and the structure of a site

  Controller: controls the way the user interacts, through the view, with the model data and functions

Most popular server-side frameworks

ASP.NET:	DotNetNuke
ColdFusion:	Model-Glue, Fusebox
Java:	Struts, Spring, JavaServer Faces, Google Web Toolkit
Perl:	Catalyst
PHP:	CakePHP, Symphony, Zend
Python:	Django
Ruby:	Ruby on Rails (very influential!)
Smalltalk:	Seaside

. . . and many others!

CMS

- CMS (Content Management System)
- Allow creating Web sites without any need for software development
- Features:
  - simplified editing of a Web page (wiki or bbcode syntax, or JavaScript rich-text control);
  - adding external content (images, additional documents, etc.);
  - user management, access control, etc.;
  - forum or blog modules;
  - ready-to-use graphical designs;
  - version control.
- Depending on the CMS, many existing plugins.
- Some specialized CMSs: blogs (Dotclear, Movable Type, TypePad), e-commerce (PrestaShop, Magento), forums (phpBB, MyBB), etc.

Most commonly used CMSs

Number of weekly downloads:

WordPress	PHP	433 767
Joomla!	PHP	189 429
Drupal	PHP	62 500
Umbraco	.NET/XSLT	5 670
eZ Publish	PHP	5 612
CMS Made Simple	PHP	4 903
SilverStripe	PHP	2 500
e107	PHP	2 242
Xoops	PHP	1 209
TikiWiki	PHP	373
phpWebSite	PHP	347
Typo3	PHP	100
Alfresco	Java	57
DotNetNuke	ASP.NET	?
Jahia	Java	?
Liferay	Java	?
modx	PHP	?
OpenCMS	Java	?
Plone	Python	?
TextPattern	PHP	?

CMS and Web programming

Even when using a CMS, it is useful to know basic Web technologies (HTML, CSS, JavaScript, server-side languages):

- to create custom CSS styles (almost compulsory);
- to develop complex site-specific applications;
- to develop or adapt plugins;
- to understand what happens when something goes wrong;
- to ensure Web pages verify some constraints (W3C validity, accessibility).

CMS and Wikis

Fuzzy frontier between the two... Wikis put the focus on:

- by default, everything editable by a large number of users; no notion of “ownership” of a page or document
- insistence on version controls

Numerous wikis: MediaWiki, TWiki, Dokuwiki...

Choosing a CMS

Items of importance:

- license (most popular CMSs are free, but not all)
- underlying programming language, important for customizing or modifying the software, also important for deploying it
- availability of site-specific features: blogs, forums, video support, etc. Possible to use external components for each task, but easier to have all integrated.

Potential disadvantages

- Slower than a typical site; cache features that can improve performance, but can cause issues in some settings
- Security issues, code that one does not fully control
- Orphan software
- Migration towards another system potentially costly
- Not adapted to all use cases

Detecting server-side technologies

Impossible to access the source code, so how to know which technologies were used on the server side of a given site?

- URL: (`.php` for PHP, `.jsp` for JSP, `.asp` for ASP, `/servlet/` for a servlet...). Everything configurable, but most often the default configuration is used.
- Can be written on the site (often for a CMS)
- Look at the comments in HTML or CSS code
- File and directory layout, loaded JavaScript libraries or CSS stylesheets, etc.
- Cookies, in particular session identifiers

Outline

Outline

PHP and HTML

- PHP script: HTML document (for example), in which PHP code is incorporated
- The PHP code is inside pseudo-tags `<?php ... ?>` (or `<? ... ?>`, or `<?= ... ?>` which is a shortcut for `<? echo ... ?>`).

Example

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">

<html>
  ...
  <body>
  <h1><?php echo 2+2; ?></h1>
</body>
</html>
```

Syntax

- Syntax similar to (and inspired by) C, Java:
  - instructions terminated with semicolons
  - base statements: `if`, `while`, `for`...
  - operators: `=`, `==`, `++`, `<=`...
  - comments: `//`, `/* */`, but `#` also works
  - similar literals (but see interpolation later on)
  - etc.
- But important differences:
  - new instructions (e.g., `foreach`), new operators (`.` is string concatenation)
  - untyped and undeclared variables, default scope the whole function
  - variables always prefixed with a “$” sign

Interpolation

Character strings in PHP can be:

- enclosed with single quotes: `'Hello world'`. Behave like Java character strings but escape sequences (`\n`...) are not available.
- enclosed with double quotes: `"Hello\n$world"`. Escape sequences possible, and variable names are interpolated (replaced with their value). For complex variables, use the syntax `${var}`: `"The value is {$matrix[$i][$j]}"`

Arrays (1/3)

- Arrays can contain values of different types: integers, strings, etc.

Example

```
$tab[0] = "1st item";
$tab[1] = "2nd item";
$tab[2] = 120;
```

Arrays (2/3)

- In PHP, assignment of an index to an item is automatic: it will be put at the end of the array.

Example

```
$tab2[] = "1st item";
$tab2[] = "2nd item";
$tab2[] = 120;
```

- Initialization can also be made with the `array` function

Example

```
$tab3 = array("1st item","2nd item",120);
```

Arrays (3/3)

- Numerical indices can also be replaced with character strings named keys
- For a given array, all keys must be distinct
- The `array` function can also be used to initialize the array

Example

```
$fname["Belmondo"] = "Jean-Paul";
$fname["Delon"] = "Alain";
$fname["Deneuve"] = "Catherine";
$fname = array("Balasko" => "Josiane", "Bourvil" => "");
```

Iterating an array (1/3)

- Iterating an array with named keys is more complex than an array index with integers: one cannot just use index order to do a simple loop
- One can use a cursor to the array: pointer to the area of the array currently browsed
- `next` and `prev` functions move the cursor initially positioned on the first item of the array
- `key` and `current` functions return, respectively, the key and the value of the current item

Iterating an array (2/3)

Example (Display all persons of the $fname array)

```
echo "Last name=".key($fname).
     " First name=".current($fname)."\n";

while(next($fname)) {
  echo "Last name=".key($fname)." First name=".current($fname);
}
```

Iterating an array (3/3)

- But easier to use the `foreach` statement

Example

```
foreach ($fname as $key => $value) {
  echo "Last name=$key First name=$value\n";
}
```

- Can also be used for integer-indexed arrays

Example

```
foreach ($array as $value) {
  ...
}
```

Functions

- To ease readbility and automate repetitive tasks, one can use functions
- The `list ( ... )` construction can be used to retrieve in different variables the return value of a function that is an array

Example

```
function Kenshin() {
  return array ("Kenshin", "Nobuhiro Watsuki", 28);
}

function addition($x,$y) {
  $sum = $x+$y;
  return $sum;
}

$z=addition($x,$y);
list ($a,$b,$c)=Kenshin();
```


But also...

- Very (too?) rich a language
- Object-oriented programming: `class`, `$object->field`... often used in PHP frameworks
- Very large standard function library, and numerous third-party library available
- Tip: `http://php.net/function` returns the documentation of the *function* function

Validation

- A PHP script with PHP code incorporated inside HTML code is not valid HTML code!
- What needs to ba validated is HTML pages produced by the script
- No PHP “validator”, but syntax errors will cause errors at runtime

Outline

$_GET and $_POST

- HTTP parameters can be retrieved in PHP with the key–value arrays `$_GET` and `$_POST`.
- Values of this array are either simple values or integer-indexed arrays (for multiple-choice list parameters whose name is suffixed with `[]` in the HTML code)
- Attention to the use of these values (controlled by the user)! Use htmlspecialchars to protect HTML special characters when using these values in HTML code

Example

```
echo "<p>Your login is: ".htmlspecialchars($_POST["login"])."</p>";
echo "<p>You marked the genres: ";
for($i=1;$i<=count($_POST['genre']);$i=$i+1) {
  echo htmlspecialchars($_POST['genre'][$i]) . " ";
}
echo "</p>";
```

File uploads (1/5)

- Information on uploaded files is available in the key–value array $_FILES:
  - keys are form fields the file was uploaded from
  - values are property sets (represented with a key–value array) describing the file received by the server, together with an *error* property that gives status information

File uploads (2/5)

Example (in a file *FormTransfer.html*)

```
<form enctype="multipart/form-data"
  action="TransferFile.php" method="post">
  ...
  <div>
    <label for="myPhoto">Choose a file:</label>
    <input type="file" name="myPhoto" id="myPhoto" />
  </div>
  ...
</form>
```

File uploads (3/5)

name	is the local name of the file on the client machine
tmp_name	is the name of a temporary file containing the file on the server
size	is the size, in bytes
type	is the MIME type of the file, e.g., “image/gif”

Example (in a script *TransferFile.php*)

```
$file=$_FILES['myPhoto'];
echo "Name client file: ".$file['name']."<br />";
echo "Name server file: ".$file['tmp_name']."<br />";
echo "File size: ".$file['size']."<br />";
echo "File type: ".$file['type']."<br />";
```

File uploads (4/5)

UPLOAD_ERR_OK no error, everything went fine

UPLOAD_ERR_INI_SIZE uploaded file is above maximum allowed file size

UPLOAD_ERR_PARTIAL file only partially transferred

UPLOAD_ERR_NO_FILE no file transferred

Example

```
$errorCode = $_FILES['myPhoto']['error'];

if($errorCode!=UPLOAD_ERR_OK) {
  echo "<p>Error while uploading the file.</p>";
}
```

File uploads (5/5)

- The PHP `copy($source,$destination)` function copies a file *source* to *destination*. Important because the temporary file will be destroyed at the end of the script.
- Attention: the program should have write access to the directories in which the files are copied

Example

```
// Copy the file in the PHOTOS directory
copy($_FILES['toto']['tmp_name'],"./PHOTOS/$id.jpg");
```

Outline

Definitions

session: information kept throughout an interaction with the different pages of a Web site

authentication: mechanism allowing to associate a login to a Web site user, so as to allow custom content or access right management on a Web application; usually, authentication is made with a password

Sessions

- Simple authentification method available in HTTP, but requires a modification of the configuration of the Web server; somewhat heavy, hard to connect with a DBMS
- No session management in HTTP; Alternatives:
  - HTTP parameters inside the URL (GET). Heavy, since all links should be updated to include these parameters.
  - Cookies.

Cookies in PHP

- Informations, in the form of key–value pairs, that a Web server requests a Web client to keep and return for each subsequent HTTP request to the same site
- `setcookie($name,$value)` : requests storage of the cookie with name $name and value $value
- `$_COOKIE` is a key–value array of all cookies the client sent

Sessions in PHP

PHP provides an abstraction of session management (uses a cookie, but no need to manipulate it directly)

session_start() opens an existing session, or creates a new one if none exists; to be put at the very top of a PHP script, or before anything else is written in the returned page, so that HTTP headers can be modified

session_destroy() terminates the current session

session_id() provides a unique identifier for the current session

$_SESSION contains the set of session parametes (key–value array) available in the current session, throughout the site

Authentication through cookies

- A form requests a login and a password
- A form processing script checks these are correct (e.g., using a query to a DBMS):
  - If yes, a PHP session is created (`session_start();`), and some parameter is added to it, e.g., named valid_user (`$_SESSION['valid_user']=1;`). We redirect to another page.
  - If not, redirection back to the form with an error message
- Other pages of a site (accessible only to valid users) start with a `session_start();` and check the user is identified with (`if($_SESSION['valid_user']==1) { ... }`) and otherwise redirect to the identification page
- A disconnection page calls `session_destroy();`

Outline

DBMS

- DBMS: DataBase Management System
- Provides efficient methods to manage data that correspond to a given structure (a schema)
- Search for data: queries
- Add, delete, modify data: updates
- Fast processing of very large data quantities
- Transaction management (do not give two customers the same seat)

Relational model

- Most common and classical data model
- Data are organized into tables, each column representing an attribute of the data

Example

First name	Last name	Birth date
John	Chan	1967-08-07
Amanda	Wood	1981-09-12
Jack	Smith	NULL

- Each attribute (column) is typed
- SQL (Structured Query Language): standard language for querying and updating relational data (small variants depending on the DBMS)

Most common DBMSs

Oracle	commercial, the most popular for business data
IBM DB2	commercial
Microsoft SQL Server	commercial
Microsoft Access	commercial, poor in features
MySQL	free, widely used on the Web
PostGreSQL	free, richer and more complex than MySQL

Data types

(MySQL types)

INT: integer (`42`)

REAL: floating point number (`3.14159`)

VARCHAR(N): character string with at most N characters; values are enclosed with single quotes (`'This is a string'`)

TEXT: long character string, no size limit

DATE: date (`2005-11-08`)

TIME: time (`09:30:00`)

NULL

- NULL: special value
- Denotes the absence of a value
- Different from `0`, `''` ...
- A standard comparison (`=`, `<>`) with `NULL` always returns FALSE
- `IS NULL`, `IS NOT NULL` can be used to test NULL-ity of a value
- Each attribute can be declared as allowing or not the `NULL` value

- Standard language to interact with a DBMS
- Example query:

```
SELECT a.name, COUNT(*)
FROM actors a, casting m
WHERE a.id=m.actor_id
GROUP BY a.name
```

- Example update:

```
UPDATE availabilities
SET nb_available=nb_available-1
WHERE train_id=1234
```

Databases for the Web

- DBMS very useful in many Web applications (directories, catalogs, reservation systems, newsgroups, blogs, etc.)
- Possible to access DBMSs more or less simplified depending on the server-side technology:
  - In PHP, function libraries for each DBMS (e.g., `mysql` or `mysqli` for MySQL).
  - In JSP, JDBC library with scriptlets, and `<sql:*>` tag with JSTL, to interface with any DBMS

Example: Using MySQL with PHP

```
if(!mysql_pconnect("server","login","password"))
  { echo "<p>Sorry, connection impossible</p>"; exit; }
if(!mysql_select_db('database'))
  { echo "<p>Sorry, cannot access database</p>"; exit; }

$result = mysql_query("SELECT * FROM Films");
if($result) {
   while($film=mysql_fetch_assoc($result)){
     echo "<p>".htmlspecialchars($film["Title"])." released in ".
       $film["Year"]." </p>";
  }
} else {
   echo "<p>Error while processing the query.</p>";
   echo "<p>MySQL message: ".mysql_error()."</p>";
}
```

Outline

Conclusion

What you should remember

- Variety of server-side technologies
- CMSs: useful in certain circumstances
- Otherwise, need to choose a language and a framework to develop with; better to use an MVC approach
- PHP, popular and easy to set up, but not necessarily the best choice for everything
- DBMSs: popular choice for storing and accessing data from Web applications
- Not covered in this course: many security issues to be aware of

References

- Any text editor for most server-side programming technologies
- A Web server (e.g., Apache)
- Server-side interpreter for the chosen server-side language (e.g., PHP and mod_php for Apache)
- For JSP, Servlets: a Java application serveur (e.g., Tomcat)
- A DBMS (e.g., MySQL); under Windows, EasyPHP bundles Apache+PHP+MySQL
- Most of the time, a way to transfer programs to the remote Web server (SSH, FTP, Web application. . .)
- `http://www.php.net/`
- `http://dev.mysql.com/doc/`

Licence de droits d'usage

Contexte public } avec modifications

Par le téléchargement ou la consultation de ce document, l'utilisateur accepte la licence d'utilisation qui y est attachée, telle que détaillée dans les dispositions suivantes, et s'engage à la respecter intégralement.

La licence confère à l'utilisateur un droit d'usage sur le document consulté ou téléchargé, totalement ou en partie, dans les conditions définies ci-après et à l'exclusion expresse de toute utilisation commerciale.
Le droit d'usage défini par la licence autorise un usage à destination de tout public qui comprend :
– le droit de reproduire tout ou partie du document sur support informatique ou papier,
– le droit de diffuser tout ou partie du document au public sur support papier ou informatique, y compris par la mise à la disposition du public sur un réseau numérique,
– le droit de modifier la forme ou la présentation du document,
– le droit d'intégrer tout ou partie du document dans un document composite et de le diffuser dans ce nouveau document, à condition que :
  – L'auteur soit informé.

Les mentions relatives à la source du document et/ou à son auteur doivent être conservées dans leur intégralité.

Le droit d'usage défini par la licence est personnel et non exclusif.
Tout autre usage que ceux prévus par la licence est soumis à autorisation préalable et expresse de l'auteur : sitepedago@telecom-paristech.fr