

Chapter 2: Interactive Web Applications

2.1 Interactivity and Multimedia in the WWW architecture

2.2 Server-Side Scripting (Example PHP, Part I)

2.3 Interactivity and Multimedia for Web Browsers

2.4 Interactive Server-Side Scripting (Example PHP, Part II)

2.5 Interactive Client-Side Scripting (Example JavaScript)

2.6 Data Storage in Web Applications

2.7 Asynchronous Interactivity in the Web

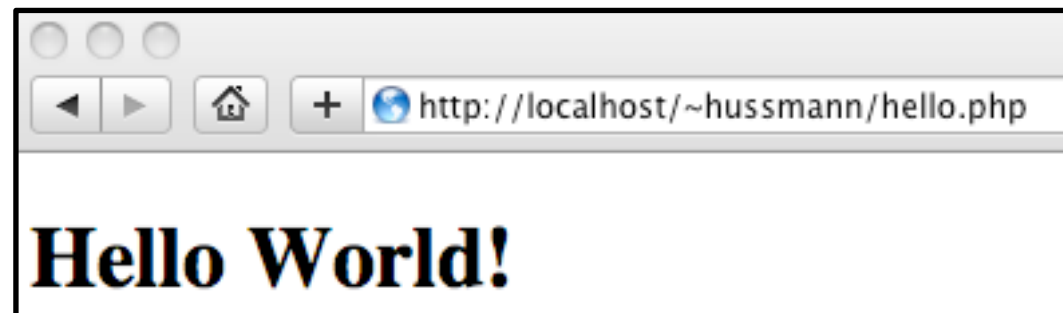
HTTP Basics

- HTTP = HyperText Transfer Protocol, see <http://www.w3.org/Protocols/>
- Client-Server communication:
 - Client opens (TCP) connection to server (usually on port 80)
 - Client sends request (as text lines)
 - Server sends response (as text lines)
 - Client closes connection (HTTP is *stateless*)
- Format of all HTTP messages (requests and responses):
 - Initial line*
 - Header lines (zero or more)*
 - Blank line*
 - Message body (optional)*
- Example HTTP request:

```
GET /lehre/ws1112/mmn/index.html HTTP/1.1
Host: www.medien.ifi.lmu.de:80
<blank line!>
```

Sample HTTP Request (GET)

```
GET /~hussmann/hello.php HTTP/1.1
ACCEPT: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
ACCEPT_ENCODING: gzip, deflate
ACCEPT_LANGUAGE: en-us
CONNECTION: keep-alive
HOST: localhost
USER_AGENT: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_2) AppleWebKit/536.26.14 (KHTML, like Gecko) Version/6.0.1 Safari/536.26.14
CONTENT_TYPE:
```



HTTP Server Responses

- Message sent back from HTTP server always contains an initial response line which gives the *status* of the request processing.
- Example (success):
`HTTP/1.1 200 OK`
- Example (error):
`HTTP/1.1 404 Not found`
- Status codes:
 - 1xx: Informational message
 - 2xx: Success of some kind
 - 3xx: Redirection to other URL
 - e.g. 303: See other URL (given in Location: header)
 - 4xx: Client side error
 - 5xx: Server side error
 - e.g. 500: Server error

Example HTTP Response

- Experimenting manually with HTTP client/server dialogues:
 - “telnet <host> 80” in UNIX shell
- Retrieving a HTML page:

```
HTTP/1.1 200 OK
Date: Wed, 24 Oct 2012 08:45:07 GMT
Server: Apache/2.2.22 (Unix) DAV/2 PHP/5.3.15 with
Suhosin-Patch mod_ssl/2.2.22 OpenSSL/0.9.8r
Content-Location: index.html.en
Vary: negotiate
TCN: choice
Last-Modified: Mon, 30 Jul 2012 09:33:21 GMT
ETag: "18664f-2c-4c608c0a63a40;4cc483d0fcd40"
Accept-Ranges: bytes
Content-Length: 44
Content-Type: text/html
Content-Language: en

<!DOCTYPE html> ... <html> ... </html>
```

Passing CGI-Style Parameters in GET Request

- Convention for passing parameter values to server-side programs
 - Introduced by the *Common Gateway Interface (CGI)*
 - Not part of the HTML protocol!
 - Interpreted by server programs, e.g. PHP module
- Syntax:
 - Parameter data stream is appended to URL after a “?”
 - Keyword/value pairs, separated by “=”, e.g. “`fibinput=12`”
 - Multiple parameter groups are separated by “&”
 - Spaces in strings are replaced by “+”
 - Non-ASCII characters (and special characters “&”, “+”, “=”, “%”) are replaced by “%xx” (hexadecimal code of character in used character set)

Example GET Request with Parameter

- Request:

```
GET /~hussmann/fibonacci2b.php?fibinput=12 HTTP/1.1
Host: localhost
```

- Response:

```
HTTP/1.1 200 OK
Date: Wed, 26 Oct 2011 11:57:45 GMT
Server: Apache/2.2.20 (Unix) mod_ssl/2.2.20 OpenSSL/0.9.8r
DAV/2 PHP/5.3.6
X-Powered-By: PHP/5.3.6
Content-Length: 338
Content-Type: text/html
```

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

```
<html>
<head> ... fib(12) = 144 ... </html>
```

GET and POST Methods in HTTP

Hypertext Transfer Protocol (HTTP) supports two methods for passing parameter values to called documents/scripts:

- GET Method:
 - Values of variables coded and transmitted within URL:
`http://host.dom/pfad/fibonacci2.php?fibinput=12`
 - Parameters can be passed just by creating a certain URL (without forms)
 - Suitable for simple requests
- POST Method:
 - Values of variables coded and transmitted in the HTTP message body data
 - Values of variables not visible in URL
 - Web server reads parameter values from message (like browser reads HTML text)
- Variable encoding is not part of HTTP (but specified for HTML forms)
 - For POST requests, the coding method is given in the Content-Type header
 - » application/x-www-form-urlencoded (CGI conventions)
 - » multipart/form-data (segmented data, better for large data blocks)

Example POST Request with Parameter

- Request:

```
POST /~hussmann/fibonacci2b.php HTTP/1.1
```

```
Host: localhost
```

```
Content-Type: application/x-www-form-urlencoded
```

```
Content-Length: 11
```

```
fibinput=12
```

- Response:

```
HTTP/1.1 200 OK
```

```
Date: Wed, 26 Oct 2011 14:06:35 GMT
```

```
...
```

```
Content-Type: text/html
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//  
EN"
```

```
"http://www.w3.org/TR/html4/loose.dtd">
```

```
<html>
```

```
<head> ... fib(12) = 144 ... </html>
```

Fibonacci Function in PHP: Using Request Data

```
<body>
  <h1>
    Fibonacci Function (Result)
  </h1>
  <h2>
    <?php
      $fibinput = $_REQUEST['fibinput'];
      function fib($n){ as in version 1 };
      echo "fib($fibinput) = ";
      echo fib($fibinput);
      echo "<br>";
    ?>
    <br>
    <a href="fibonacci2a.html">New Computation</a>
  </h2>
</body>
```

fibonacci2b.php

Variables, Parameter Passing and Security

- Global arrays `$_REQUEST`, `$_GET`, `$_POST`
 - for accessing external values determined at call time (like form input)
 - `$_REQUEST` contains all parameters given in request, `$_GET` and `$_POST` contains all parameters passed by the resp. method
 - Obtaining individual variable values by array lookup:
`$_REQUEST['var'];`
- Older PHP versions (up to 4.2.0):
 - Huge security hole by not distinguishing between external parameters (e.g. input from HTML forms) and local variables
 - » External values were directly accessible through variables (like `$fibinput`)
 - Manipulations of URL (GET parameter values) may enable setting of internal variables (e.g. `$authorization_successful`...!)
 - Old behavior can still be enabled by PHP server configuration

HTML Reminder: Forms

- User input in HTML:

- `<form>` Element

- Sub-element:

- `<input type=ty name=name>`

- Allowed types (*ty*) (selection):

<code>checkbox</code>	Check box (Attribute <code>checked</code>)
<code>radio</code>	Radio button (Attribute <code>checked</code>)
<code>text</code>	Textzeile
<code>textarea</code>	Multi-line text input area
<code>password</code>	Text input area not displaying the input
<code>file</code>	File selection
<code>button</code>	General button
<code>submit</code>	Button to send form contents
<code>reset</code>	Button to reset form contents

- `<select name=name>` Pop-up menu for selection from options

- List of options: Sub-elements `<option>`

- `<option selected>` defines "pre-selected" values

HTML Form Example

```
<body>
  <form action="test.php"
    method="GET"
    enctype="application/x-www-form-urlencoded">
    <label>Name <input type="text" name="name"
      maxlength="10" /></label><br>
    Sex:
    <input type="radio" name="sex"
      value="male"></input> male
    <input type="radio" name="sex"
      value="female"></input> female <br>
    <input type="checkbox" name="married"
      value="yes">Married</input><br>
    <input type="submit" value="Submit" />
  </form>
</body>
```

```
/test.php?name=Max+Muster&sex=male&married=yes
```

HTML Forms and Server-Side Scripts

- HTML page containing forms usually calls separate script page and transfers form data as variable values
- **action** attribute for HTML tag `<form>`
 - Specifies the server page to process the input
 - Can contain embedded script
- **method** attribute for HTML tag `<form>`
 - Specifies the HTTP method to be used to transfer form data to the server
 - Possible values: GET (default), POST
- **enctype** attribute for HTML tag `<form>`
 - Specifies the encoding method to be used for form data
 - Possible values:
 - » application/x-www-form-urlencoded (CGI conventions) (default)
 - » multipart/form-data (segmented data)

Example: POST Request with Multipart Encoding

- HTML:

```
<form action="test.php"  
      method="POST" enctype="multipart/form-data">
```

- Generated HTTP request:

```
POST /test.php HTTP/1.1  
Host: localhost ...  
Content-Type: multipart/form-data;  
boundary=-----103832778631715  
Content-Length: 355  
  
-----103832778631715  
Content-Disposition: form-data; name="name"  
  
Max Muster  
-----103832778631715  
Content-Disposition: form-data; name="sex"  
  
male  
-----103832778631715  
Content-Disposition: form-data; name="married"  
  
yes  
-----103832778631715--
```

Fibonacci Function in PHP (Version 2): Input Form Calling PHP Script

```
<body>
  <h1>
    Fibonacci Function (Input)
  </h1>
  <h2>
    Please enter number:
    <form name="fibform" action="fibonacci2b.php">
      <input type="text" name="fibinput"
        value="0"><br>
      <input type="submit" value="Compute">
    </form>
  </h2>
</body>
</html>
```

fibonacci2a.html

Combination of Input and Result Pages

```
<body>
  <h1>
    Fibonacci Function
  </h1>
  <h2>
    <?php
      function fib($n){ as above };
      $eingabe = $_REQUEST['fibinput'];
      echo "fib($eingabe) = ";
      echo fib($eingabe);
      echo "<br>";
    ?>
    <br>
    Please enter number:
    <form name="fibform" action="fibonacci2.php">
      <input type="text" name="fibinput" value="0"><br>
      <input type="submit" value="Compute">
    </form>
  </h2>
</body>
```

action="fibonacci2.php" can be omitted

fibonacci2.php

Embedding Media in HTML

- Media embedding requires:
 - Media data (a file)
 - Player software
- Typical media data:
 - Sound files (e.g. .wav, .mp3, .ogg, .midi)
 - Movie files (e.g. .avi, .mov, .ogv, .flv)
 - Programs to be executed on a virtual machine (“universal player”), e.g.:
 - » Java applets
 - » Flash runtime code (Shockwave Flash, .swf)
 - » Silverlight application packages (.xap)
- Browser integration:
 - Built-in: Browser "knows" about player for media type
 - Plug-in: Flexible association between player and media type
- Video on the Web is currently dominated by universal multimedia formats (in particular Flash)

Embedding a YouTube Video

```
<object width="500" height="315">  
<param name="movie" value=  
"http://www.youtube.com/v/_oBuE66majc&hl=de&fs=1&rel=0&border=1">  
</param>  
<param name="allowFullScreen" value="true"></param>  
<param name="allowscriptaccess" value="always"></param>  
<embed src=  
"http://www.youtube.com/v/_oBuE66majc&hl=de&fs=1&rel=0&border=1"  
type="application/x-shockwave-flash" allowscriptaccess="always"  
allowfullscreen="true" width="500" height="315"></embed></object>
```

- Redundant information
 - Nested “object” and “embed” tags
- Adobe Flash runtime code referenced
 - MIME type “application/x-shockwave-flash”
 - Movie player program, parameterized

<embed> Tag in HTML

- <embed> tag refers to browser *plugin*
 - Introduced by Netscape with browser version 2.0
 - **Outdated, not** part of the HTML standard
- Example:
`<embed src="yippee.wav" width="140" height="60">`
- Plugin:
 - Separate program to handle special file types
 - » E.g. Flash player plugin handles .swf files
 - Located on client
- Important attributes:
 - **src**: Data to be embedded (URI or local file)
 - **width, height** etc.: Control of appearance
 - **autostart**: Determines whether playback starts immediately
 - **pluginspage**: Where to find information on the plugin software
 - **pluginurl**: Where to find the plugin software

<object> Tag in HTML

- **<object>** : Generic solution to embed arbitrary data files
 - Part of HTML 4.0 and XHTML 1.0 standards, supported by Microsoft
 - Supports media files, files to be opened with separate application software, files to be opened with plugin software, executable programs (e.g. Java applets or ActiveX controls)
 - Not well supported in all browsers

- Example (modern standard-conform style):

```
<object data="nibbles.swf"  
  type="application/x-shockwave-flash"  
  width="600" height="400">  
  <param name="movie" value="nibbles.swf">  
  <param name="quality" value="high">  
</object>
```

- Important attributes:
 - **data**: Data to be embedded (URI or local file)
 - **width**, **height** etc.: Control of appearance
 - **type**: MIME type of data
- Nested tag **<param>** to convey arbitrary name/value pairs

selfhtml.org

More on the <object> Tag in HTML

- Further attributes:
 - **classid**: May be used to specify the location of an object's implementation via a URI. It may be used together with, or as an alternative to the `data` attribute, depending on the type of object involved.
 - » Specifies the version of the player software to be used
 - » In practice often platform specific, e.g. ActiveX registry values
 - **codebase**: Specifies the base path used to resolve relative URIs specified by the `classid`, `data`, and `archive` attributes. When absent, its default value is the base URI of the current document.
 - » In practice, misused to specify the location of the player software (like `pluginurl`)
 - **codetype**: Specifies the content type of data expected when downloading the object specified by `classid`.
 - » MIME type for code of player (not data)
- <object> tag with child tags in its body:
 - Uses the inner HTML code as display alternative

<http://www.alistapart.com/articles/flashsatay/>

Combining `<embed>` and `<object>`

- Problems:
 - Older browsers:
 - » Microsoft IE ignores `<embed>`
 - » Netscape/Mozilla ignores `<object>`
 - Current browsers:
 - » `<object>` as shown above works on all platforms
 - » However, Microsoft IE does not allow streaming of the data (but loads all data first)
- Pragmatic solution:
 - Enclosing an `<embed>` tag in an `<object>` tag (see above)
 - Recommended for Flash, stable
 - Not (X)HTML standard conform!
- Complex solution for Flash, standard conform:
 - Use portable `<object>` code from above
 - Load a container movie which then loads the target movie

<http://www.alistapart.com/articles/flashsatay/>

HTML 5

- HTML Version 5
 - Draft W3C standard (most recent draft 19 October 2010!)
 - Developed in parallel to XHTML 1.0
 - » XHTML 2.0 development has been stopped
- HTML 5 is partially supported already by many modern browsers
- HTML 5 contains standardized and simple media embedding tags
 - audio
 - video
 - embed

Audio Embedding in HTML 5

- Example:

```
<html> ...  
  <body>  
    ...  
    <audio src="nightflyer.ogg" autoplay>  
      Your browser does not support the <code>audio</code> element.  
    </audio>
```

- Attributes (examples):
 - autoplay: Playback starts automatically
 - controls: Control UI elements are made visible
 - loop: Plays in an endless loop
 - preload: Hints about preloading expectations
- Subelement <source>:
 - Alternative way to specify data source
 - Multiple occurrence is possible, first supported version is taken

Video Embedding in HTML 5

- Example:

```
<html>
  <body>
    <video controls>
      Your browser does not support the <code>video</code> element.
      <source src="big_buck_bunny_480p_stereo.ogg" type="video/ogg">
      <source src="big_buck_bunny_480p_surround-fix.avi">
    </video>
```

- Additional Attributes compared to <audio> (examples):
 - height, width: Dimensions of video image
 - poster: Image to be shown until first frame becomes available
- Events (can be handled e.g. with JavaScript, examples):
 - empty
 - canplay
 - ended
 - abort
 - volumechange


<embed> in HTML 5

- HTML 5 contains a standardized version of the `<embed>` element
- Purpose:
 - Embed arbitrary content played back via plug-in software
- Examples:
 - Flash content
 - Java applets
- Not intended for media playback

Side Remark: HTML 5 vs. Flash

- HTML 5 establishes a clear alternative to Flash:
 - Simple audio and video playback
 - » Makes usage of Flash video for video portals unnecessary
 - Test version of YouTube portal for HTML5 video exists already
 - » Other video Web sites are more hesitating
- Still open issues:
 - File format/compression (see next slide)
 - Javascript interaction (e.g. for switching to full screen)
 - Content protection
 - Bandwidth adaptivity
- Alternative players to browser built-in player
 - Video JS
 - SublimeVideo
 - ...



 YouTube HTML5 Video Player

You are in a trial for HTML5 video on YouTube. Some use HTML5 is a new browser technology that allows us to play

Video Codecs and HTML5 Video

- HTML5 Working Group: All browsers should support at least one common video format
 - Good quality & compression, hardware-supported, royalty-free!
- Problems with mainstream formats:
 - Patents on H.264
 - Fear of hidden patents for Ogg Theora
- Google:
 - Release of WebM to the public (after purchase of On2)
 - VP8 format with Vorbis audio in Matroska container
- Still no simple common solution for the key manufacturers available

Current Situation in Codec Support

Browser	Operating system	Latest stable release	Video formats supported		
			Theora	H.264	VP8 (WebM)
Android browser	Android	4.1.2 "Jelly bean" (October 9, 2012; 14 days ago) ^{[27][28]}	2.3 ^[29]	3.0 ^[29]	2.3 ^[29]
Chromium	All supported	N/A	r18297 ^[30]	Manual install ^[note 1]	r47759 ^[32]
Google Chrome	All supported	22.0.1229.94 (October 10, 2012; 13 days ago)	3.0 ^{[33][34]}	3.0 ^{[34][note 2]}	6.0 ^{[36][37]}
Internet Explorer	Windows	9.0.10 (September 21, 2012; 32 days ago)	Manual install ^[note 3]	9.0 ^[39]	Manual install ^[note 4]
	Windows Phone	9.0 (February 14, 2011; 19 months ago)	No ^[citation needed]	9.0 ^[citation needed]	No ^[citation needed]
Konqueror	All supported	4.9.2 (2 October 2012; 21 days ago) ^[42]	4.4 ^[note 5]		
Mozilla Firefox	Windows 7+	16.0.1 (October 11, 2012; 12 days ago) ^[44]	3.5 ^[45]	4.0, Manual install ^[note 6]	4.0 ^{[47][48]}
	All other supported			No ^[note 7]	
Opera	All supported	12.02 (August 30, 2012; 54 days ago)	10.50 ^[50]	No	10.60 ^{[51][52]}
Safari	iOS	6.0 (July 25, 2012; 2 months ago)	No	3.1 ^{[53][54]}	No
	MacOS X		Manual install ^[note 8]		Manual install ^[56]
	Windows		5.1.7		
Web (previously Epiphany)	All supported	3.6 (September 26, 2012; 27 days ago) ^[57]	2.28 ^[note 9]		

removal pending

en.wikipedia.org

Chapter 2: Interactive Web Applications

2.1 Interactivity and Multimedia in the WWW architecture

2.2 Server-Side Scripting (Example PHP, Part I)

2.3 Interactivity and Multimedia for Web Browsers

2.4 Interactive Server-Side Scripting (Example PHP, Part II)

2.5 Interactive Client-Side Scripting (Example JavaScript)

2.6 Data Storage in Web Applications

2.7 Asynchronous Interactivity in the Web

Permanent Storage of Information

- Displayed content very often comes from server or client side storage
 - E-Commerce, E-Government, ...
 - Personalized pages
 - Discussion fora
 - ...
- Server-side storage:
 - Huge amounts of data (database)
 - » or simple files!
 - Data update by external software
 - Integration with arbitrary software systems
- Client-side storage:
 - Small amounts of data
 - Security-based restrictions (local information under user control)

Sessions and States

- HTTP is stateless
 - Server does not “remember” any data from previous transactions
- Linking several transactions to a “session” with common data storage
 - Client-side: Storing all data on client and re-transmit for every transaction
 - Server-side: Storing all data on server, client has to identify the session
- Common solution:
 - Server-side software offers session support
 - » E.g. session support in PHP
 - Client stores “session id”
 - Methods for linking request to session id:
 - » Variable/value pair in GET or POST request
 - » HTTP “Cookie”

Cookies in HTTP

- Small data units stored in the browser storage area, controlled by browser
- Cookie contains:
 - *Name* (String), also called *key*
 - *Value* (String)
 - *Expiration date*
 - optional: domain, path, security information
- HTTP transfers cookies between client and server
 - In response, server can include header line “Set-Cookie:”
 - » Further information: name + value pair, expiration time
 - Cookie is stored by the browser
 - In further requests to the same server, client includes header line “Cookie:”
 - » Further information: name + value pair
 - Only cookies related to the requested server are transferred

Types of Cookies

- Session cookie
 - Deleted on browser termination
 - No expiration date given = session cookie
- Persistent cookie
 - For tracking, personalization
- Secure cookie
 - Only transmitted when secure connection to server is used
- HttpOnly cookie
 - Access only for HTTP, not for script APIs
- Third party cookie
 - Cookies set for different domain than currently visited server
 - Used for tracking and cross-domain advertising

Cookies in PHP: Screenshot

Current Time: 18:31:59

Cookies currently set:

cookie2=another text
cookie1=text for cookie 1

<input type="text" value="name"/>	Cookie Name
<input type="text" value="text"/>	Cookie Content
<input type="text" value="10"/>	Lifetime (minutes)

Accessing Cookies

Displaying a list of all cookies currently set (for this application) by reading from global array `$_COOKIE`:

```
<html>
  <?php
    date_default_timezone_set('Europe/Berlin');
    echo "Current Time: ", date("G:i:s"), "<br><br>\n";
    echo "<b>Cookies currently set:</b><br><br>\n";
    while (list($k, $v) = each($_COOKIE))
      echo $k, "=", $v, "<br>\n";
  ?>
  ...
</html>
```

HTML Form for Setting a Cookie

```
<form>
```

```
  <input type="text" name="key" value="name">  
    Cookie Name<br>
```

```
  <input type="text" name="val" value="text">  
    Cookie Content<br>
```

```
  <input type="text" name="tim" value="10">  
    Lifetime (minutes)<br>
```

```
  <input type="submit" name="set"  
    value="Set Cookie"><br>
```

```
</form>
```

- Page loaded via **action** is identical to page containing the form ("cookietest.php") – omitting the **action** attribute is sufficient.
- Due to server-side execution, the actual setting action can only be carried out when the next page is loaded!
- "**name**" attribute of **submit** button required for distinction to other buttons ("refresh" in the example).

Setting the Cookie

```
<?php
    if ($_GET['set']) {
        $key = $_GET['key'];
        $val = $_GET['val'];
        $tim = $_GET['tim'];
        $exp = time() + $tim * 60;
        setcookie($key, $val, $exp);
    }
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
    Transitional//EN" "http://www.w3.org/TR/html4/loose">
<html>
...
```

- **"name"** attribute of **submit** button (**'set'**) used to decide whether **"set"** button was pressed
- **setcookie()** call has to be very first output of page, to be transmitted together with the headers (HTTP requirement).

A Simple Discussion Forum (1)

- Interactive submission of text contributions
- Display of all submissions available on server
- Server uses simple text file for storage
- Altogether approx. 50 lines of HTML+PHP !

Discussion Forum

New Contribution:

Name:

Contribution (one line):

Current discussion:

3 contributions

Contribution # 1:

Name: Max
Text: I have an idea

Contribution # 2:

Name: Peter
Text: I like this idea

Contribution # 3:

Name: Janet
Text: I don't like it

A Simple Discussion Forum (2)

Contents of file "forum.txt":

- Each two consecutive lines represent one contribution.
- First line: Name
- Second line: Text

Max

I have an idea

Peter

I like this idea

A Simple Discussion Forum (3)

Display of the full content of the file 'forum.txt'

- Used file function:
 - `file()`: Converts file content to string array
- Used array function:
 - `count()`: Length of array

```
<h2>Current discussion:</h2>
```

```
<?php
```

```
    $content = file("forum.txt");
    echo "<h3>", count($content)/2, " contributions</h3>";
    echo "<hr>";
    $i = 0;
    while ($i < count($content)) {
        echo "<h3>Contribution # ", ($i+2)/2, " :</h3>";
        echo "<b>Name: &nbsp;</b>", $content[$i++], "<br>";
        echo "<b>Text: &nbsp;</b>", $content[$i++], "<br>";
        echo "<hr>";
    }
}
```

```
?>
```

forum.php

A Simple Discussion Forum (4)

Extending the file 'forum.txt' with a new contribution

- Parameter `$newcontrib` indicates whether the "enter contribution" button was pressed
- Used file functions:
 - `fopen()`, `fclose()`: Open file ("a"=append), close file
 - `fputs()`: Write string to file

```
<?php
    $newcontrib = $_REQUEST['newcontrib'];
    $name = $_REQUEST['name'];
    $contrib = $_REQUEST['contrib'];
    if ($newcontrib != "" && $name != "" && $contrib != "") {
        $file = fopen("forum.txt", "a");
        if ($file) {
            fputs($file,$name . "\n");
            fputs($file,$contrib . "\n");
            fclose($file);
        }
    }
?>
```

Potential Enabled by Server-Side Scripts

- Receive and store user input
 - In various forms of persistent storage
 - » Plain text files, XML files, data base
- Process input and compute results
 - Depending on various information available on server side
- Create output suitable for being displayed in Web browsers
 - HTML, may include JavaScript
- Make use of advanced features offered by Web browsers
 - Examples: Cookies, user agent identification

Applications to Multimedia

- PHP is not directly multimedia-related, but HTML-oriented
- HTML allows media embedding
- The combination of HTML + PHP + media embedding enables the creation of new digital media
- Examples for interactivity added to media playback, realizable by PHP scripts
 - Selection of media, e.g. search functions
 - » Using forms and backend data base
 - User-specific recommendations
 - » Using cookies
 - Aggregating (explicit and implicit) user input
 - » Frequency of use for individual media (charts)
 - » Correlation of use across media (collective recommendation)
 - » Tagging

Examples for PHP Multimedia Scripts

PHP: Multimedia Scripts and Programs

Scripts

Sort by: [PageRank](#) | [Newest](#) | [Hits](#) | [Alphabetical](#) | [Ranking](#)

PR: 2



YouTube Video Organizer Script

The script allows you to create custom categories for your YouTube Videos on your own site - [Read more](#)

not rated yet

(0 Reviews. Rating: Total Votes:)

N/A



iScripts Visualcaster

iScripts VisualCaster is a video hosting script that could be used to provide video hosting service to your customers. It is a turnkey solution to provide services like youTube.With millions of... - [Read more](#)

not rated yet

(0 Reviews. Rating: Total Votes:)

[Ads by Google](#)

[Script](#)

[Perl Hosting](#)

[CGI PHP](#)

[PHP CRM](#)

PR: 2



phpMDB - The music sharing database

phpMDB is a web based file sharing platform, featuring a comprehensive administrative panel to simplify the management of system settings, user accounts, file categorization and verification.... - [Read more](#)

not rated yet

(0 Reviews. Rating: Total Votes:)

N/A



TopMediaScript

Build your own media sharing site in minutes, with TopMediaScript. Allowing for the uploading and sharing of videos, games and images; as well as publishing embedded videos from sites such as... - [Read more](#)

not rated yet

(0 Reviews. Rating: Total Votes: 0)

www.webscriptsdirectory.com

Multimedia Functions in PHP Library (1)

- See e.g. Multimedia chapter of tutorial "Practical PHP Programming"
<http://www.tuxradar.com/practicalphp/11/0/0>

- Example: Creating an image

```
<?php
    $image = imagecreate(400,300);
    // do stuff to the image
    imagejpeg($image, '', 75);
    imagedestroy($image);
```

?> File: picture1.php

```
<HTML>
    <TITLE>PHP Art</TITLE>

    <BODY>
        <IMG SRC="picture1.php" />
    </BODY>
</HTML>
```

- Computer graphics functions, like:

```
$white = imagecolorallocate($image, 255, 255, 255);
imagefilledrectangle($image, 10, 10, 390, 290, $white);
```

Multimedia Functions in PHP Library (2)

```
<?php
    $image = imagecreate(400,300);
    $gold = imagecolorallocate($image, 255, 240, 00);
    $white = imagecolorallocate($image, 255, 255, 255);

    imagefilledrectangle($image, 0, 0, 400, 300, $gold);

    for ($i = 4, $j = 3; $i < 400; $i += 8, $j += 6) {
        imagefilledrectangle($image, $i, $j, 400 - $i, $j+3, $white);
    }

    imagepng($image);
    imagedestroy($image);
?>
```

