Multimedia im Netz

Wintersemester 2011/2012

Part I

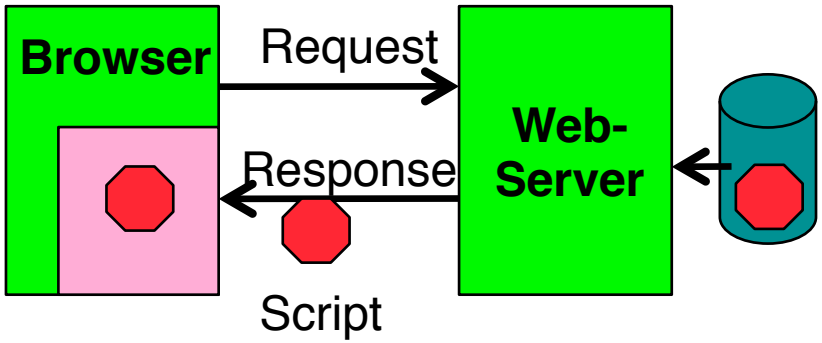
Web Technologies for Interactive Multimedia

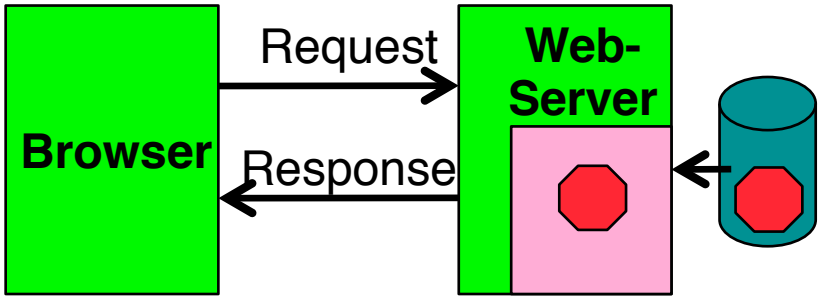
Chapter 2: Interactive Web Applications

Dynamic Web Contents

- Content shown to user in browser is dependent on some external variables
- Examples of external variables:
  - Date and time
  - Contents of an information archive (e.g. recent news)
  - Actions of the user
    - » Pointing to elements
    - » Clicking at a certain position
    - » Filling out forms
- Wide-spread applications:
  - E-Commerce
  - Interpersonal communication media (forums, discussion boards)
  - Mass media (news and other information services)

Server-Side vs. Client-Side Realisation

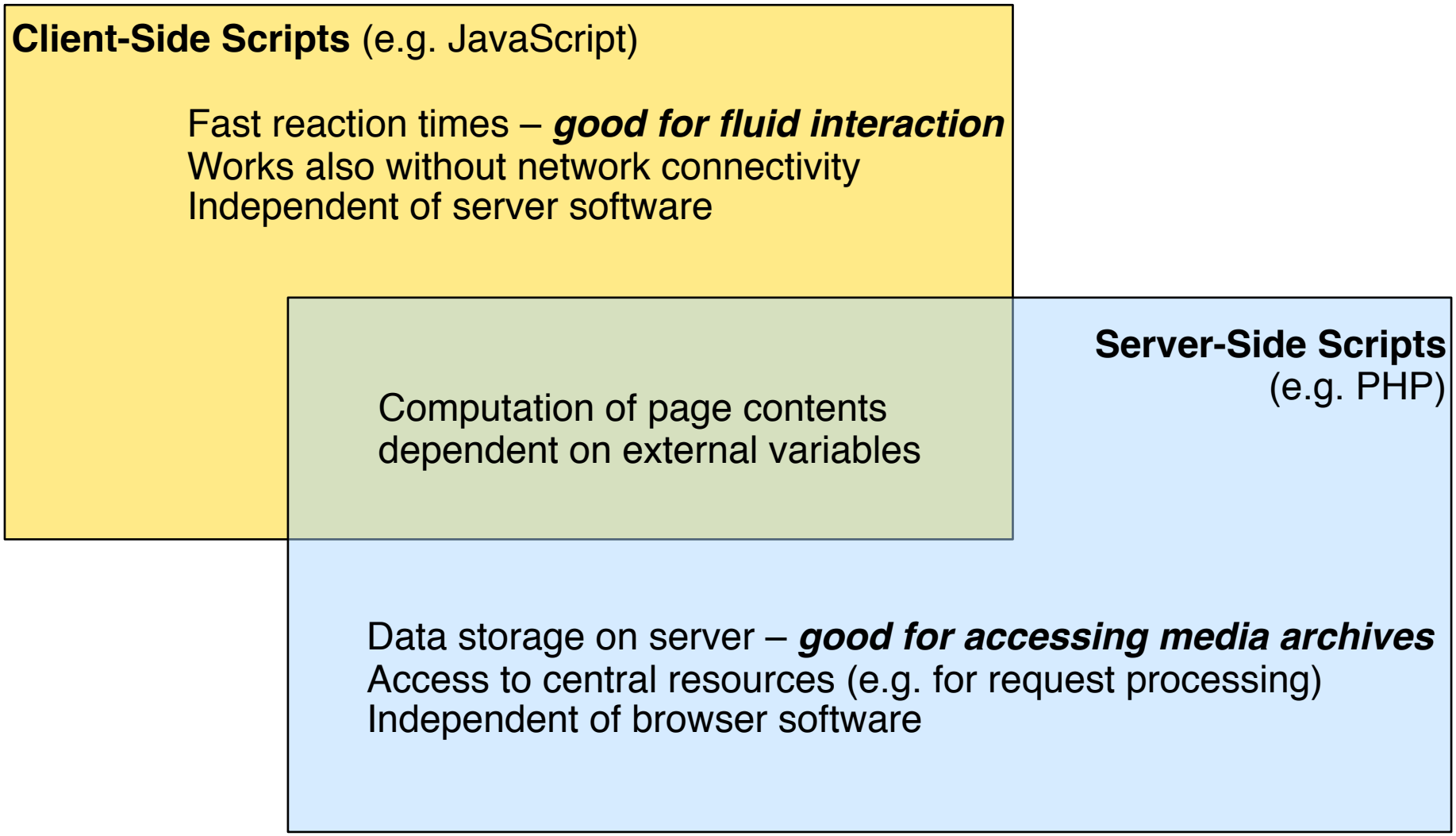




- Client-side realisation:
  - Browser contains execution engine for scripts
  - Web server does not need to execute scripts
  - Script ist sent to client as part of server response
  - Example: JavaScript

- Server-side realisation:
  - Web server contains execution engine for scripts
  - Browser does not need to execute scripts
  - Script is executed on server and computes response to client
  - Example: PHP

Server Scripts vs. Client Scripts



Common Gateway Interface (CGI)

- A request can identify an executable command on the server
  - Command is executed
  - Parameters are passed to it via environment variables (e.g. QUERY_STRING)
- Informal standard, by a developer community in 1993
  - Current standard (1.1) is documented at NCSA (http://hoohoo.ncsa.illinois.edu/cgi/)
  - IETF RFC 3875
- CGI programs can be written in any executable language:
  - Programming languages (e.g. C/C++, Java)
  - Scripting languages (e.g. Unix shells, Perl, TCL)
- Typical locations on server file system:
  - **`/cgi-bin`**
  - **`/cgi-src`**

Principles of Writing CGI Code

- Passing parameters to the CGI program:

  `http://www.example.com/cgi-bin/example.cgi?paraminfo`

  – Program example.cgi is executed

  – String “paraminfo” is made accessible for the program in the environment variable QUERYSTRING

- Passing information to the browser:

  – The CGI program has to write the data in a form displayable by the browser

  – Always the first line is a MIME type specification, e.g.:

  ```
  Content-type: text/html
  ```

- Example for a very simple CGI program:

```
#!/bin/sh
echo "Content-Type: text/plain"
echo ""
echo "Hello, world."
```

Drawbacks of CGI

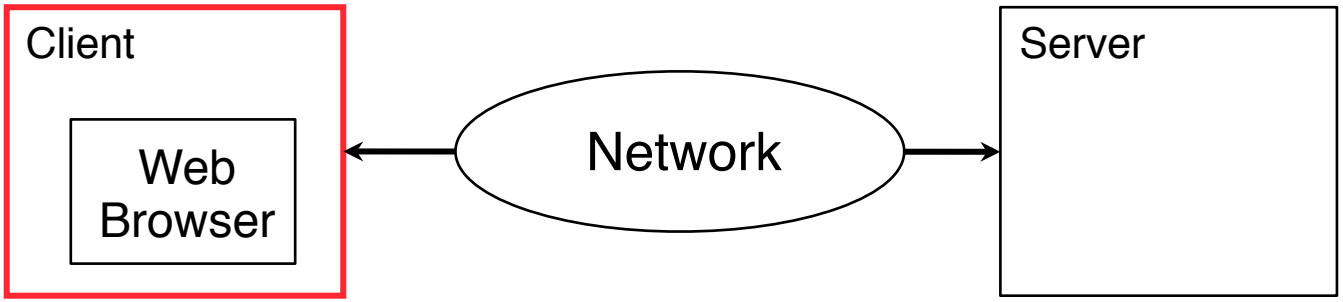
- High danger of security problems:
  - Injection of malicious script code (through program errors)
- Calling a CGI command is expensive:
  - Creating a new process (in Unix)
  - Sometimes on demand compilation
  - Generally not suitable to high load situations

- Alternatives to CGI:
  - SCGI (Simple CGI)
  - FastCGI (single persistent process to handle queries)
  - WSGI (Web Server Gateway Interface) for Python
  - Microsoft Internet Server Application Programming Interface (IISAPI)
  - Server modules
    - » E.g. script language modules for Apache

Modern Web Architectures for Interactivity

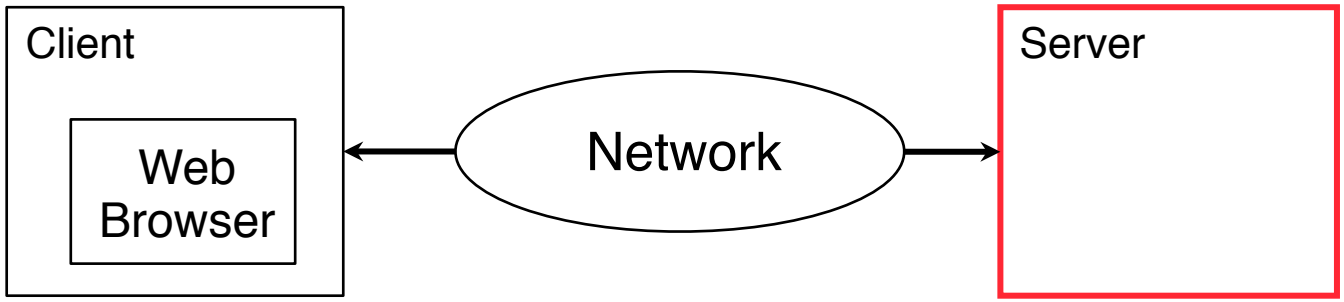
- Web server software add-ons
  - Interfaces to common scripting and programming languages
  - e.g. Perl, Ruby, Java
- Web server software integrated with specific execution environments (“Application Server”)
  - Highly optimized for good throughput
  - Complex, many configuration options
  - e.g. Java Enterprise Edition, Microsoft .NET framework
- Scripting languages specifically designed for Web application development
  - e.g. PHP
  - see later

Media Support – Functions of Client Only



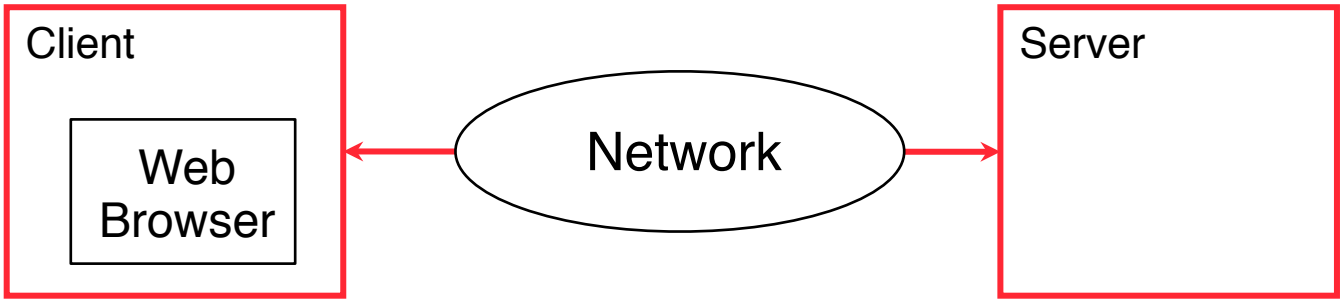
- Media rendering:
  - Recognition of media file types
    - » MIME registry of browser
  - Local media playing software
    - » Plugins or separate programs
- Interactivity:
  - Local interactions
    - » Highlighting, dynamic menus etc.

Media Support – Functions by Server Only



- Media rendering:
  - Storage of media files and meta-information
  - Indexing and querying
- Interactivity:
  - Interactions with server-side effect
    - » E.g. database updates (registration, buying, ...)
  - Interactions with global effect for all users
    - » E.g. adding a comment, uploading a video

Media Support – Functions by Client & Server



- Media streaming:
  - Playback of incomplete content in client
  - Playout in defined order from server
  - Synchronization, rate control, buffering
  - Flow control (stop, start, pause)
  - Adaptation to network conditions
- Interactivity:
  - Near real-time interactions
    - » E.g. status notifications, data ticker

Chapter 2: Interactive Web Applications

Literature:

A. Trachtenberg, D. Sklar: PHP Cookbook, O'Reilly 2006

R. Lerdorf, K. Tatroe, P. MacIntyre: Programming PHP, 2nd. ed., O'Reilly 2006

Server-Side Script Language PHP

(Only an example for a server-side script language!)

- PHP:
  - **P**ersonal **H**ome **P**age Toolkit
    - » 1995, Rasmus Lerdorf
    - » 2003, new by Zeev Suraski, Andi Gutmans
  - **P**HP **H**ypertext **P**reprocessor (recursive acronym, backronym)
- Current version: 5.3.8 (August 2011), 6 in preparation (currently stalled)
- OpenSource project:
  - see www.php.net
  - Can be used and modified freely (PHP license)
- Syntax loosely oriented towards C
  - Variations of possible syntax
- Extensive function library
  - being extended by community

Prerequisites for Using PHP in Practice

- Always (even if using just one computer)
  - Installation of a Web server
    - » OpenSource: *Apache*
    - » Microsoft *Internet Information Server*
  - Invocation of PHP always indirectly by loading pages from server (http://...)
    - » Loading from local computer: http://localhost/...
- Installation of PHP software as plug-in for used Web server
- Very often also installation of a data base system (e.g. MySQL)
- Frequently used acronyms for specific configurations:
  - LAMP: Linux, Apache, MySQL, PHP
  - WIMP: Windows, Internet Information Server, MySQL, PHP
  - MOXAMP: MacOS X, Apache, MySQL, PHP

Activation of PHP Module in Apache

- Example (MacOS 10.5):
  - Apache + PHP module are pre-installed
  - Configuration needs to be updated (remove a comment sign)
- /etc/apache2/httpd.conf:

```
# This is the main Apache HTTP server configuration file.  It contains the
# configuration directives that give the server its instructions.
# See <URL:http://httpd.apache.org/docs/2.2> for detailed information.
...
LoadModule bonjour_module     libexec/apache2/mod_bonjour.so
LoadModule php5_module        libexec/apache2/libphp5.so
#LoadModule fastcgi_module     libexec/apache2/mod_fastcgi.so
```

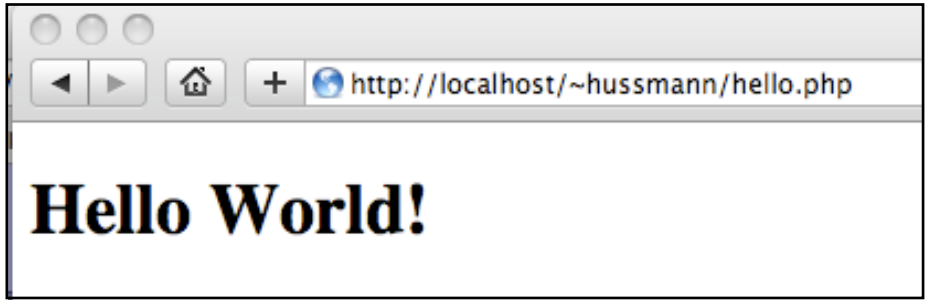
Hello World in PHP

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


<html>
<head>
  <title>Hello World with PHP</title>
</head>


<body>
  <h1>
     <?php echo "Hello World!"; ?>
  </h1>
</body>
</html>
```

File hello.php
in Web server directory



Embedding of PHP into HTML

- XML style (used here):
  - Like *Processing Instructions* in XML

```
<?php    PHP Text   ?>
```

- SGML style:
  - Widely used in older scripts
  - Not really recommendable: PHP language not specified

```
<?    PHP Text   ?>
```

- HTML style:
  - Using HTML tag:

```
<script language="php">    PHP Text   </script>
```

PHP Syntax (1)

- Inheritance from shell scripts
  - Variables start with "$"
  - Some UNIX commands part of the language, e.g.:

```
echo "Hello";
```

- Control statements exist in different versions, e.g.:

```
if (bedingung1)
  anw1
elseif (bedingung2)
  anw2
else anw3;

if (bedingung1):          anwfolge1
elseif (bedingung2):      anwfolge2
else:                     anwfolge3
endif;
```

PHP Syntax (2)

- Various comment styles:
  - One-line comment, C style:
    ```
    echo "Hello"; // Hello World
    ```
  - One-line comment, Perl style / Unix shell style:
    ```
    echo "Hello"; # Hello World
    ```
  - "One line" ends also at end of PHP block
  - Multi-line comment, C-style:
    ```
    echo "Hello"; /* Comment
      spreads over multiple lines */
    ```
  - Do not create nested C-style comments!

- Instruction must always be terminated with ";"
  - Exception: end of PHP block contains implicit ";"

PHP Type System

- Scalar types:
  - boolean, integer, float (aka double), string
- Compound types:
  - array, object
- Special types:
  - resource, NULL
  - Resource type: refers to external resource, like a file

- "The type of a variable is not usually set by the programmer; rather, it is decided at runtime by PHP depending on the context in which that variable is used."

  (PHP Reference Manual)

Arrays in PHP (1)

- An array in PHP is actually an ordered map
  - Associates values to keys
  - Keys can be integer or string (even mixed in same array)
  - Multi-dimensional arrays (arrays of arrays) are supported
- Multiple use of the array data structure for array, list, hash table, dictionary, stack, queue, …
- Creating arrays (examples):

```php
<?php
  $arr = array("foo" => "bar", 12 => true);
  echo $arr["foo"]; // bar
  echo $arr[12];    // 1
?>
<?php
  $arr = array("somearray" => array(6 => 5, 13 => 9, "a" => 42));
  echo $arr["somearray"][6];    // 5
  echo $arr["somearray"][13];   // 9
  echo $arr["somearray"]["a"];  // 42
?>
```

Arrays in PHP (2)

- Arrays with strictly numerical keys
  - Implicit position numbers as keys

```
$array = array( 7, 8, 0, 156, -10);
  // this is the same as array(0 => 7, 1 => 8, ...)
```

- Arrays as collections

```
$colors = array('red', 'blue', 'green', 'yellow');
foreach ($colors as $color) {
    echo "Do you like $color?\n";
}
```

- Assignment operations on arrays always mean copying of values!

Object-Orientation in PHP

```php
<?php
  class SimpleClass {

     // property declaration


  public $var = 'a default value';

     // method declaration
     public function displayVar() {
        echo $this->var;
     }
  }
```

Property access with "->" operator

Visibilities:
public, private, protected

```php
$instance = new SimpleClass();
$instance->var = 'property value';
$instance->displayVar();
```

Further Object-Oriented Concepts in PHP

- Static class properties and methods
  - "static" keyword
- Class Inheritance:
  - "extends" keyword in class definition
- Class Abstraction:
  - "abstract" keyword in class definition
- Scope Resolution operator ("::"):
  - Access to static, constant or overridden properties or methods of a class

```php
<?php
        class MyClass {
                const CONST_VALUE = 'A constant value';
        }
        $classname = 'MyClass';
        echo $classname::CONST_VALUE; // As of PHP 5.3.0
?>
```

  - In combination with "self" and "parent" keywords (denoting classes): Possibility to access overridden version of a method (cf. "super" in Java)

Example: Fibonacci Function in PHP (Version 1)

```
<body> ...
  <h2>
     <?php
          function fib($n){
               if ($n==0)
                    return 0;
               else
                    if ($n==1)
                         return 1;
                    else
                         return fib($n-1)+fib($n-2);
          };
          echo "fib(3) = ", fib(3), "<br>";
          echo "fib(8) = ", fib(8), "<br>";
     ?>
  </h2>
</body>
</html>
```

fibonacci1.php