UNIT-VI: Alternative Paradigm: Scripting Languages

Faculty In-charge
Aaysha Shaikh
Assistant Professor (IT Dept.)
email: aayshashaikh@sfit.ac.in
Academic Year: 2021-22

What Is A Scripting Language

- Modern scripting languages have two principal sets of

ancestors.

- command interpreters or “shells” of traditional batch and “terminal” (command-line) computing
  - IBM’s JCL, MS-DOS command interpreter, Unix sh and csh
- various tools for text processing and report generation • IBM’s RPG, and Unix’s sed and awk.

•From these evolved

- Rexx, IBM’s “Restructured Extended Executor,” which dates from 1979
- Perl, originally devised by Larry Wall in the late 1980s, and now the most widelyused general purpose scripting language. • Other general purpose scripting languages include Tcl (“tickle”), Python, Ruby, VBScript (for Windows) and AppleScript (for the Mac)

What Is A Scripting Language

- Scripting on Microsoft platforms
  - As in several other aspects of computing, Microsoft tends to rely on internally developed technology in the area of scripting languages
  - Most scripting applications are based on VBScript - dialect of Visual Basic • Microsoft has also developed a very general scripting interface (Windows Script)

that is implemented uniformly by the operating system, the web server, and the Internet Explorer browser

- A Windows Script implementation of JScript, the company's version of JavaScript, comes pre-installed on Windows machines, but languages like Perl and Python can be installed as well, and used to drive the same interface.
- Many other Microsoft applications use VBScript as an extension language, but for these the implementation framework (Visual Basic for Applications [VBA]) does not make it easy to use other languages instead

What Is A Scripting Language

- Scripting on Microsoft platforms
  - Given Microsoft's share of the desktop computing market, VBScript is one of the most widely used scripting languages
    - It is almost never used on other platforms
  - Perl, Tcl, Python, PHP, and others see significant use on Windows
    - For server-side web scripting, PHP currently predominates: as of February 2005, some 69% of the 59 million Internet web sites surveyed by Netcraft LTD were running the open source Apache web server, and of them most of the ones with active content were using PHP
    - Microsoft's Internet Information Server (IIS) was second to Apache, with 21% of the sites, and many of those had PHP installed as well.
    - For client-side scripting, where chrome controls about 70% of the browser market, most web site administrators need their content to be visible to the

other 30%

- Explorer supports JavaScript (JScript), but other browsers do not support VBScript

Difference: Programming Language and Scripting

1	A scripting language is a language that uses a naive method to bring codes to a runtime environment	A Programming language is a language which is used by humans to navigate their communication with computers.
2	These are made for a particular runtime environment.	Programming languages are of three types -: low-level, Middle-level and High-level
3	They are used to create dynamic web applications	Programming languages are used to write computer programs.
4	Example -: Bash, Ruby, Python	Example -: C, C++, Java.
5	Scripting languages can be easily ported among various operating systems.	Programming languages are translation free languages
6	These languages requires a host.	These languages are self executable.
7	Do not create a .exe file.	These generate .exe files.

8	Most of the scripting languages are interpreted	Most of the programming languages are compiled .
9	All the scripting languages are programming languages.	All the programming languages are not scripting languages.

Examples of Scripting Languages

- Bash
- • Node.js
- • PowerShell
- • Ruby
- • PERL
- • CGI
- • Tcl
- • Python
- • PHP

- ASP
- JSP
- ASP.NET • VB-Script • Java-Script

Advantages and Applications of Scripts

- Advantages:
- • Easy Learning
- • Fast Editing
- • Interactive
- • Flexibility
- • Rapid development
- • Validating contents
- • Include many libraries to create new functionalities
- • Better utilization of resources

Applications

- Used in web applications

as server-side (perl, php) and client-side scripting (javaScript, ajax, jQuery)

- Use

d in system-administration (shell, perl, python)

• Used in Game applications and multimedia

• Used to create plugins and extensions for existing applications.

• Controlling remote applications

• Command line interface

Common Characteristics of Scripts

- 1. Both batch and interactive use
- 2. Economy of expression (Short sentences)

- 3. Lack of declarations; simple scoping rules.
- 4. Flexible dynamic typing
- 5. Easy access to other programs
- 6. Sophisticated pattern matching and string manipulation
- 7. High level data types (lists, tuples dictionary etc.)

Common Characteristics:

- Both batch and interactive use
  - While a few languages (e.g. Perl) have a compiler that requires the entire source program, almost all scripting languages either compile or interpret line by line

- Many “compiled” versions are actually completely equivalent to the interpreter running behind the scenes (like in Python).

Common Characteristics:

- Economy of expression
  - Two variants: some make heavy use of punctuation and short identifiers (like Perl), while others emphasize “English-like” functionality
- Either way, things get shorter. Java versus Python (or Ruby or Perl):

```
class Hello {
 public static void main(String[] args) {
     System.out.println(“Hello, world!”);
 }

print “Hello, world!\n”
```

Common Characteristics:

- Lack of declarations; simple scoping rules.
  - While the rules vary, they are generally fairly simple and additional syntax is necessary to alter them.
  - In Perl, everything is of global scope by default, but optional parameters

can limit the scope to local

- In PHP, everything is local by default, and any global variables must be explicitly imported.
- In Python, everything is local to the block in which the assignment appears, and special syntax is required to assign a variable in a surrounding scope.

Common Characteristics:

- Flexible dynamic typing
  - In PHP, Python and Ruby, the type of a variable is only checked right before use
  - In Perl, Rexx, or Tcl, things are even more dynamic:

```
$a = "4"
print $a . 3 . "\n"
print $a + 3 . "\n"
```

Outputs the following:

```
43
7
```

Common Characteristics:

- Easy access to other programs
  - While all languages provide support for OS functionality, scripting languages generally provide amazing and much more fundamental built in support.
  - Examples include directory and file manipulation, I/O modules, sockets, database access, password and authentication support, and network communications.

Common Characteristics:

- Sophisticated pattern matching and string manipulation
  - Perl is perhaps the master of this, but it traces back to the text processing sed/awk ancestry.
  - These are generally based on extended regular expression.

Common Characteristics:

- High level data types
  - In general, scripting languages provide support for sets, dictionaries, lists and tuples (at a minimum).
  - While languages like C++ and Java have these, they usually need to be imported separately.

- Behind the scenes, optimizations like arrays indexed using hash tables are quite common.
- Garbage collection is always automatic, so user never has to deal with heap/stack issues.

Problem domains for using scripting

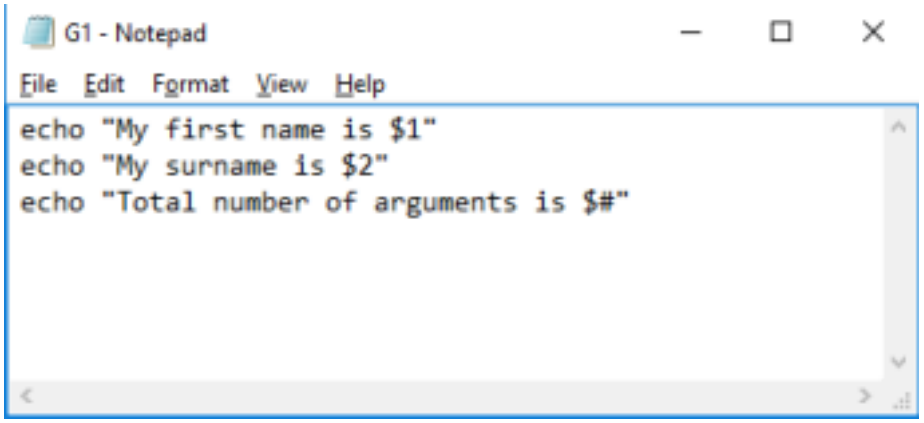
- Many scripting languages are general purpose as they support features like modules, separate compilation etc.
- • But they are intended for use in three main domains
- 1. Shell languages
- 2. Text processing
- 3. Glue Languages

1. Shell Languages

- Shell languages are heavily string oriented
- They are used to perform system level tasks and finding patterns in file.
- All variables are string valued.
- Common scripting languages include: – Shell scripts - sh, bash, csh, tcsh
- First set of scripting languages are those designed to execute terminal commands in batch mode. E.g bash scripts i.e. .bat files for MSDOS
- With shell programming – Repeated tasks can be done much faster

- Useful tasks in shell programming – Renaming sets of files – Changing the format of a set of files (e.g. png to tiff)
- Shell languages can not be used as text editors

Bash Script(Shell Script)



Save this file as G1.sh, set execute permission on that file by typing **chmod a+x name.sh** and then execute the file like this: **./name.sh**.

OUTPUT:

$ chmod a+x name.sh $ ./name.sh Hans-Wolfgang Loidl
My first name is Hans-Wolfgang
My surname is Loidl
Total number of arguments is 2

Scripts: The !# syntax

- To run a script in a file:

`. my_script`

- This reads the input line by line - but it's not an executable. •

Most version of UNIX can make it a script:

- Mark it as executable: i.e. `chmod +x my_script`
- Begin the script with a control sequence telling it how to run it:

`#!/bin/bash`

- This syntax is not just for bash - used also for Perl, Python, etc.

Bash Script Example-2

Bash is a Unix shell and command language written by Brian Fox. • It is a command line interpreter that runs in a text window where user can interpret commands to carry out various actions.

- myScript.sh

#!/bin/sh#char sequence followed by path to the interpreter
echo Hello World# echo to print.

# Set the script executable permission by running chmod command:
$ chmod 755 my-script.sh

Run or execute the script

```
$ ./my-script.sh
sh my-script.sh
Hello World
$
```


Bash Script Example

```
vivek@nixcraft:/tmp$ vi hello.sh
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ chmod +x hello.sh
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ ls -l hello.sh
-rwxr-xr-x 1 vivek vivek 31 Jan 21 15:08 hello.sh
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ ./hello.sh
Hello World
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ bash hello.sh
Hello World
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ sh hello.sh
Hello World
vivek@nixcraft:/tmp$ cat hello.sh
#!/bin/bash
echo "Hello World"
vivek@nixcraft:/tmp$
```


Bash Script Example-2

- #!/bin/sh for filename in *.nii.gz ; do fname=`$FSLDIR/bin/remove_ext${filename}`

- fslmaths ${fname} -s 2 ${fname}_smooth2
- mv ${fname}.nii.gz ${fname}_smooth0.nii.gz
- done
- What it does:
- • For each image (*.nii.gz) it smooths it to make a new one of the same name but ending in _smooth2 and also renames the unsmoothed image to end with _smooth0 • How this works:
- • The variable filename is used in a for loop to go through each name matching *.nii.gz
- • The variable fname is set to the filename with the ending (e.g. .nii.gz) removed. • • ${filename} and ${fname} are used to get the values (contents) of the variables • • fslmaths is used to do the smoothing.
- • mv is used to do the renaming (notice that .nii.gz is needed here, but not for the fsl tools, as they work with or without the .nii.gz endings).

2. Text Processing Languages

- Other set of scripting languages was developed for text processing

and report generation. E.g. sed (Stream-Editor) and awk (Alfred, Weinberger and Kernighan)

- Sed and awk are special online text editors
- These text editors are used to account for interactive features such as insertion, deletion, replacement, bracket matching etc.
- Sed is very powerful and can write complex scripts – It is used for one line programs
- Awk is data driven script language and is modified version of sed which looks more like a programming language
- AWK is a line-oriented language

Sed Text Processing Example

- Common syntax

sed[OPTION]…..[-f][file]

sed -n ‘/hello/p’ file1

- This command will display all the lines which contains hello sed ‘s/hello/HELLO/’ file1

- This command will substitute hello with HELLO everywhere in the file.

sed ‘/hello/,+2d’ file1

- This command will delete the two lines starting with the first match of ‘hello’

awk Text Processing Language

Common syntax

BEGIN { …. initialization awk commands …}

{ …. awk commands for each line of the file…}

END { …. finalization awk commands …}

Example:

```
#!/usr/bin/awk -f
#only print out every 3rd line of input file
BEGIN {skip=3}
{for (i=1; i<skip;i++)
        {getLine};
Print $0}
```


awk Text Processing Example

- An awk program operates on each line of an input file.
- For each line of the input file, it sees if there are any pattern-matching instructions, in which case it only operates on lines that match that pattern, otherwise it operates

An awk program operates on each line of an input file. • For each line of the input file, it sees if there are any pattern-matching instructions, in which case it only operates on lines that match that pattern, otherwise it operates on all lines. • The awk commands can do some quite sophisticated maths and string manipulations, and awk also supports associative arrays. • AWK sees each line as being made up of a number of fields, each being separated by a 'field separator'. By default, this is one or more space characters, so the line: this is a line of text • It contains 6 fields. Within awk, the first field is referred to as $1, the second as $2, etc. and the whole line is called $0. • The field separator is set by the awk internal variable FS, so if you set FS=”:” then it will divide a line up according to the position of the ':’ . • Other useful internal variables are NR which is the current record number (ie the line number of the input file) and NF which is the number of fields in the current line.

awk Control Statements

- Control Statements:

if (condition) statement [else statement]

while (condition) statement

do statement while (condition)

for (expr1; expr2; expr3) statement

for (var in array) statement

break

continue

atan2(y, x)
cos(expr)
exp(expr)
int(expr)
log(expr)
Rand()
sin(expr)
sqrt(expr)
srand([expr])

Inbuilt functions:

- These are inherited from shell languages and text-processing languages.
- Glue language is very useful in rapid prototyping environments where multiple software utilities are glued together quickly before being developed in a single programming language or framework.
- The gluing of multiple software utilities provides enhanced features and functionality to the base software/solution
- VBScript, AppleScript, Ruby, Python, Perl and PHP are popular examples of glue languages.
- JavaScript is also a glue language but is less popular as a glue language.

Python as a Glue Language

- Python is in use at many places as an integration language, used to glue together existing components.
- The large software components (OS, game, web browser etc) written in C/C++ available to the Python programmer.
- Python extension modules (written in C/C++) that make the functionality of large software components available to the Python programmer.

- Using Python, better applications can be developed because different kinds of programmers can work together on a project.
- For example, when building a scientific application, C/C++ programmers can implement efficient numerical algorithms, while scientists on the same project can write Python programs that test and use those algorithms.
- The scientist doesn't have to learn a C/C++ programming language, and the C/C++ programmer doesn't need to understand the science involved in python.
- Without Python, large amounts of C/C++ code have to be written to provide a flexible input mechanism so that scientists can feed the program its data, but with python, a much more flexible input mechanism in a much shorter time.

Example of Scripting languages

- Part 1: Static document formats for the web
  - Document forms: HTML and CSS

- Data forms: XML, DTDs and Schemas, XSL
- High-end graphics forms: VRML, SVG

•Part 2: Client-side interactive web pages

- Client-Side Scripting languages: JavaScript, VBScript
- Client-Side embedded applications: Java applets, ActiveX,

Flash •Part 3: Server-side web page creation

- Scripting languages: CGI and Perl, PHP, ColdFusion
- High-level frameworks: Servlets and JSP, ASP, ASP.NET

•Part 4: Web service architectures

- WSDL, SOAP

Client Side: Scripting

Languages JavaScript, VBScript, DHTML

Client Side Script

- Client-side scripting is used when the client-side interaction within a web page is required.
- • Client-side scripting is programming the behavior of the client's browser where browser is interpreting the script.
- • Client browser must have script interpreter
- • The client-side scripts are firstly downloaded at the client end and then interpreted and executed by the browser (default browser of the system).
- • The client-side scripting is browser-dependent. i.e., the client-side browser must be scripting enabled in order to run scripts

Client Side Script

- Some example uses of client-side scripting may be
- – To get the data from user's screen or browser.
- – For playing online games.
- – Customizing the display of page in browser without reloading or reopening the page.
- • Adding dynamic contents to the web page
- – Validation of Form data, image rollovers, time sensitive or random page elements
- – Document animation and automation
- – Handling cookies.
- • Defining programs with web interfaces .
- – Utilize buttons, textboxes, clickable images, prompts etc.
- • Basic document intelligence

Example: Client-Side Scripts

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
<script>
document.write("Hello Everyone!");
</script>
</head>
<body>
<h1>My First Heading</h1>
<p>My first paragraph.</p>
</body>
</html>
```

Client-Side Scripts

- Advantages:
- Allows for more interactivity
- Can perform actions quickly without going to server.
- Easier to use for those whose browser does
- Many free resources are available
- Platform independent

Disadvantages:

- If browser is out of date website wont display properly
- Different browsers support script differently so quality assurance testing is required
- Some browser will disable the active contents telling user that they may be harmful.
- Not secure as it is visible in the code
- Scripts are not fully featured and have limitations on them due to security

JavaScript

- The most common scripting language
  - Originally supported by Netscape, eventually by IE
- Typically embedded in HTML page
  - Executable computer code within the HTML content •Interpreted at runtime on the client side
- Can be used to dynamically manipulate an HTML document
  - Has access to the document's object model
  - Can react to events
  - Can be used to dynamically place data in the first place •Often used to validate form data
- Weak typing

JavaScript Syntax

- Code written within <script> element
  - e.g., `<script type="text/javascript">`
    `document.write("Hello World!")`
    `</script>`
  - Use `src` attribute for scripts in external files
- Placement determines execution time
  - Scripts in header must be invoked explicitly
    - e.g., during events
  - Scripts in body executed when that part is being processed.

JavaScript Syntax

- User can declare variables
  - e.g., `var name = "user";`
  - Variables can be global to the page
- User can declare functions
  - `function func(`*`argument1,argument2,…`*`)`
    `{` *`some statements`* `}`
  - Function can return values with `return`
- Standard conditionals
  - `if..then..else, switch, ?: operator`
- Standard loops
  - `while, do..while, for`

JavaScript Syntax

- JavaScript has built-in "Object" types

- Variety of operators and built-in functions
- Arrays, Booleans, Dates, Math, Strings

- Direct access to the HTML DOM model
- HTML Elements have script-specific event attributes • e.g., `<body onmousedown="whichButton()">` • e.g., `<input type="button" onclick="uncheck()" value="Uncheck Checkbox">`

VBScript

- Microsoft's answer to JavaScript

- Never been supported by Netscape
- Less in use now

- Use `<script type="text/vbscript">`
- Similar to JavaScript
  - Follows Visual Basic look and feel
  - Possible to declare variables
    - Use “option explicit” to force declaration
  - Separates procedures and functions

DHTML

- DHTML is a marketing buzzword
  - It is not a W3C standard

- Every browser supports different flavour
- It is HTML 4 + CSS stylesheets + scripting language with access to document model

Server side: Scripting and low-level languages

CGI, Perl, PHP, Python, ColdFusion

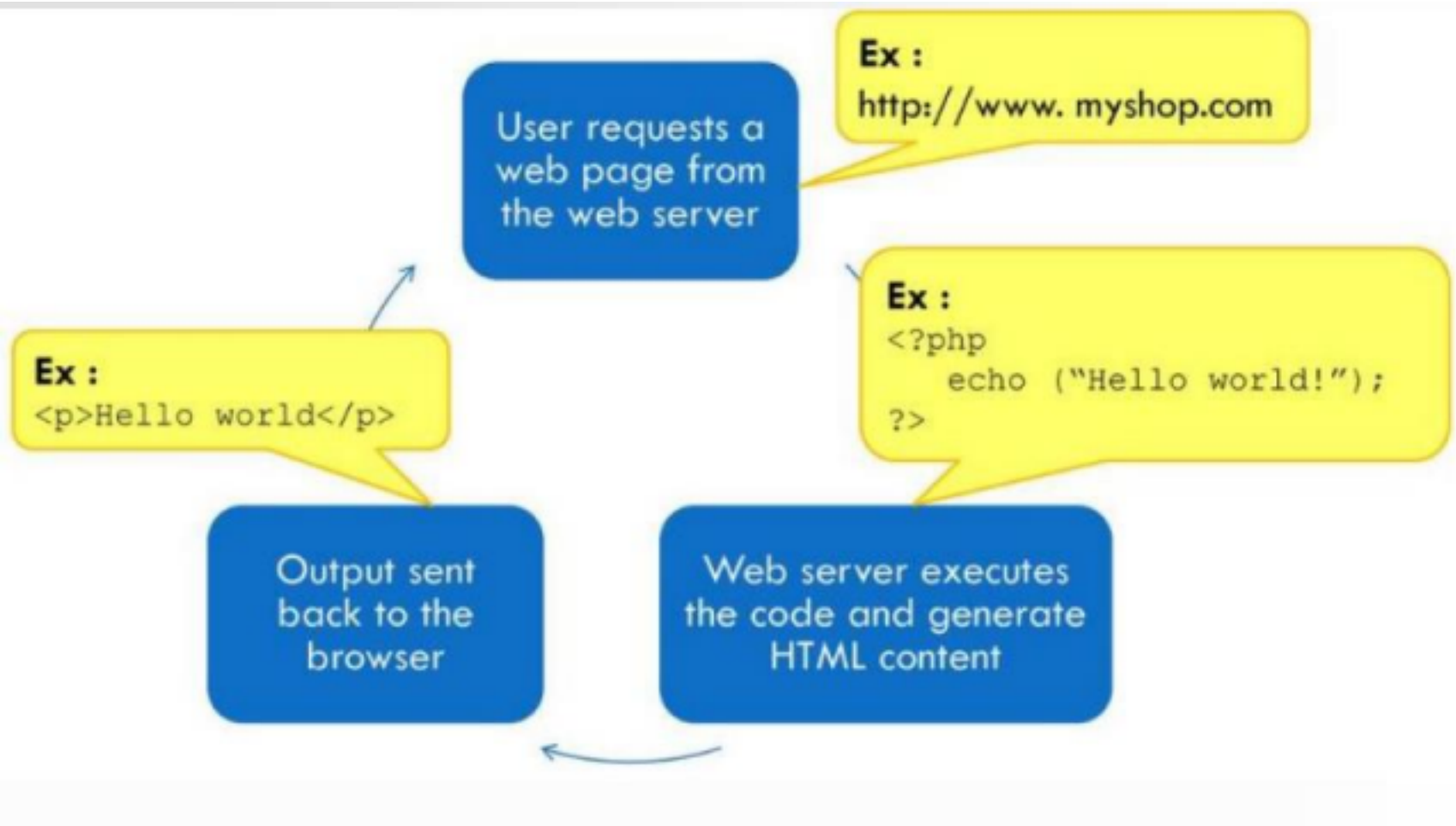
Server-Side Scripts

- Server-side scripting is responsible for the completion or carrying out a task at the server-end and then sending the result to the client-end. • In server-side script, it doesn't matter which browser is being used at client-end, because the server does all the work.
- • Server-side scripting is mainly used when the information is sent to a server for processing at the server-end.
- • Some sample uses of server-scripting can be :
- – Password Protection.
- – Browser Customization (sending info as per the requirements of client-end

browser). Respond to user queries or data submitted in HTML forms. • – Customize a web page to make it more useful for individual users – Form Processing • – Building/Creating and displaying pages created from a database (Retrieve) • – Dynamically editing changing or adding content to a web-page.

- • Here are some popular server-side scripting languages PHP, Perl, ASP(Active Server Pages), JSP(Java Server Pages).

Server-Side Scripts



Server-side Scripts

Types of Server-side-scripts

• Microsoft's ASP (Active Server Pages) or ASP.net which runs on windows server. • PHP is Hypertext Pre-processor which can be interpreted by any server but mostly used by Apache web servers
• Perl, JSP, Ruby, ColdFusion, Python and CGI

Server-Side Scripts

- **Advantages of Server-Side-Scripts:**
- • Does not require the user to download plugins

- • Page load times are faster than Client-Side-Scripts
- • You can create single website template; each new dynamic page that you create will automatically use it
- • You can configure a site to use CMS (Content Management System), which simplifies editing, publishing, adding of images and creation of web applications
- • Your scripts are hidden from the view, user sees only HTML output
- • User able to include external files to save coding
- **Disadvantages of Server-Side-Scripts:**
- • Sever-Side-Scripts can be used by attacker to gain access to the server
- • Scripts respond to URL input, changing URL to something can give users access to the server
- • To combat these attacks, the system admins need to use firewalls and need to keep these scripts updated with their security version.

Difference: Server side and Client-side

script 47

Common Gateway Interface (CGI)

- CGI Scripts

- The original mechanism for server-side web scripting is the Common Gateway Interface (CGI)
- A CGI script is an executable program residing in a special directory known to the web server program
- When a client requests the URI corresponding to such a program, the server executes the program and sends its output back to the client
  - this output needs to be something that the browser will understand: typically HTML.
- CGI scripts may be written in any language available
  - Perl is particularly popular:
    - its string-handling and “glue” mechanisms are suited to generating HTML
    - it was already widely available during the early years of the web

Common Gateway Interface (CGI)

- Standard interface allowing web server to delegate page creation

to external programs

- Arguments passed via environment variables

```
#!/usr/bin/perl

print "Content-type: text/html\n\n";

$host = `hostname`; chop $host;
print "<HTML>\n<HEAD>\n<TITLE>Status of ", $host,
      "</TITLE>\n</HEAD>\n<BODY>\n";
print "<H1>", $host, "</H1>\n";
print "<PRE>\n", `uptime`, "\n", `who`;
print "</PRE>\n</BODY>\n</HTML>\n";
```

Figure 13.10 **A simple CGI script in Perl.** If this script is named **status.perl**, and is installed in the server's **cgi-bin** directory, then a user anywhere on the Internet can obtain summary statistics and a list of users currently logged into the server by typing *hostname/cgi-bin/status.perl* into a browser window.

PERL based CGI scripts

- Practical Extraction and Report Language [1987]
  - Popular interpreted language among system administrators • Aimed at string processing
- Messy yet powerful language
  - Scripted, Procedural, Object Oriented

- OO support very sketchy
- Mixed notions from UNIX, C, Basic, SED/AWK
- Weak typing system
- Rapid built-in data structures
- Relies heavily on regular expressions

- Variety of extensions and libraries
  - CGI module facilitates web programming
    - e.g., easy access to arguments from URL

CGI Limitations

- Not appropriate for busy servers
  - Each program instance is a separate process
- Security risks
  - Only web-master has install privileges
  - Bad code can cause serious trouble
  - Explanation:Though widely used, CGI scripts have several disadvantages:
  - The web server must launch each script as a separate program, with

potentially significant overhead

- Though, CGI script compiled to native code can be very fast once running • Scripts must generally be installed in a trusted directory by trusted system administrators
- they cannot reside in arbitrary locations as ordinary pages do • The name of the script appears in the URI, typically prefixed with the name of the trusted directory, so static and dynamic pages look different to end users
- Each script must generate not only dynamic content, but also the HTML tags that are needed to format and display it
- This extra “boilerplate” makes scripts more difficult to write
- Most web servers now provide a “module loading” mechanism that allows interpreters for one or more scripting languages

Example: Server-Side Script- PHP

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport"
content="width=device-width,  initialscale=1.0">
```

```
<title>PHP - Hello, World!</title>
</head>
<body>
<h1> <?php echo 'Hello, World!'; ?> </h1>
</body>
</html>
```

Web page:

PHP - Hello, World!

PHP

- Personal Home Page tools
  - Open-source language for server-side scripting
    - Commercial 3rd party optimizers available
  - Adopted in popular large-scale web-applications
    - PhPBB bulletin board system
    - Software behind Wikis and WikiPedia
  - Some standalone rich-client applications

- Built-in facilities for popular protocols and services
- Shifts towards OOP
- Requires special server support
  - Web master must allow php scripts

Python

- A popular multi-paradigm language [’90]
  - Considers itself a “dynamic programming language” rather than a scripting language
  - Used to build some large scale applications
  - Inherent object oriented programming
- Variety of built-in data types
- Extensible

- Some support for functional programming
  - Interactive mode a-la LISP
- Indentation is used to indicate blocks
  - No semicolons or curly braces
  - Whitespace can destroy a program

ColdFusion

- Macromedia's server-side scripting language
  - Based on Allaire's software early prototype
  - Tag-based access to databases
- Tag based
  - Easier to learn than other languages
  - Extensible
- Targeted for the enterprise market

- Security and scalability features
- Interacts with variety of protocols and services
- Visual tools for rapid development
- Recent versions can run on J2EE application servers

Questions?
Thank you !