**UNIT-V**

World wide web, web browsers, web servers, uniform resource locator, Home pages, Basics of HTML, creating links, Anatomy of URL and kinds of URLs, HTML assignments, Editors and converters, New features of HTML, creating tables, Using images, Using external media, writing and designing web pages, Introduction to CGI scripts.

**1. World Wide Web (www)**

**Evolution**

**World Wide Web** was created by **Timothy Berners Lee** in 1989 at **CERN** in **Geneva.** World Wide Web came into existence as a proposal by him, to allow researchers to work together effectively and efficiently at **CERN.** Eventually it became **World Wide Web.** The following diagram briefly defines evolution of World Wide Web:



**Overview**

**WWW** stands for **World Wide Web.** A technical definition of the World Wide Web is : all the resources and users on the Internet that are using the Hypertext Transfer Protocol (HTTP).A broader definition comes from the organization that Web inventor **Tim Berners-Lee** helped found, the **World Wide Web Consortium (W3C).**

The World Wide Web is the universe of network-accessible information, an embodiment of human knowledge. In simple terms, The World Wide Web is a way of exchanging information between computers on the Internet, tying them together into a vast collection of interactive multimedia resources.

**Note: Internet** and **Web** is not the same thing: Web uses internet to pass over the information.

## 2. Web Browser

**Web Browser** is application software that allows us to view and explore information on the web. User can request for any web page by just entering a URL into address bar. Web browser can show text, audio, video, animation and more. It is the responsibility of a web browser to interpret text and commands contained in the web page. Earlier the web browsers were text-based while now day’s graphical-based or voice-based web browsers are also available. Following are the most common web browser available today:

|  |  |
| --- | --- |
| **Browser** | **Vendor** |
| Internet Explorer | Microsoft |
| Google Chrome | Google |
| Mozilla Firefox | Mozilla |
| Netscape Navigator | Netscape Communications Corp. |
| Opera | Opera Software |
| Safari | Apple |
| Sea Monkey | Mozilla Foundation |
| K-meleon | K-meleon |

### Starting Internet Explorer

Internet explorer is a web browser developed by Microsoft. It is installed by default with the windows operating system however; it can be downloaded and be upgraded.

To start internet explorer, follow the following steps:

* Go to **Start** button and click **Internet Explorer.**



The **Internet Explorer** window will appear as shown in the following diagram:



### Accessing Web Page

Accessing web page is very simple. Just enter the URL in the address bar as shown the following diagram:



### Navigation

A web page may contain **hyperlinks.** When we click on these links other web page is opened. These hyperlinks can be in form of text or image. When we take the mouse over an hyperlink, pointer change its shape to hand.



**Key Points**

* In case, you have accessed many web pages and willing to see the previous webpage then just click back button.
* You can open a new web page in the same tab, or different tab or in a new window.

**3. Web servers**

**Web servers** are computers that deliver (serves up) **Web** pages. Every **Web server** has an IP address and possibly a domain name. For example, if you enter the URL http://www.webopedia.com/index.html in your browser, this sends a request to the **Web server** whose domain name is webopedia.com. The server then fetches the page named index.html and sends it to your browser.

Any computer can be turned into a Web server by installing server [software](http://www.webopedia.com/TERM/S/software.html) and connecting the machine to the [Internet](http://www.webopedia.com/TERM/I/Internet.html). There are many Web server software applications, including public domain software and commercial packages.

**Web Server Working**

Web server respond to the client request in either of the following two ways:

* Sending the file to the client associated with the requested URL.
* Generating response by invoking a script and communicating with database



**Key Points**

* When client sends request for a web page, the web server search for the requested page if requested page is found then it will send it to client with an HTTP response.
* If the requested web page is not found, web server will the send an**HTTP response: Error 404 Not found.**
* If client has requested for some other resources then the web server will contact to the application server and data store to construct the HTTP response.

# Web - Server Types

**1. Apache HTTP Server**

This is the most popular web server in the world developed by the Apache Software Foundation. Apache web server is open source software and can be installed on almost all operating systems including Linux, UNIX, Windows, FreeBSD, Mac OS X and more. About 60% of the web server machines run the Apache Web Server.

**2. Internet Information Services (IIS)**

The Internet Information Server (IIS) is a high performance Web Server from Microsoft. This web server runs on Windows NT/2000 and 2003 platforms (and may be on upcoming new Windows version also). IIS comes bundled with Windows NT/2000 and 2003; Because IIS is tightly integrated with the operating system so it is relatively easy to administer it.

**3. Sun Java System Web Server**
This web server from Sun Microsystems is suited for medium and large web sites. Though the server is free it is not open source. It however, runs on Windows, Linux and UNIX platforms. The Sun Java System web server supports various languages, scripts and technologies required for Web 2.0 such as JSP, Java Servlets, PHP, Perl, Python, and Ruby on Rails, ASP and ColdFusion etc.

**4. Uniform Resource Locator**

The central idea in the development of web was the URL. A URL is a web address that uniquely identifies a document on the web such a document can be an image, a HTML file, a program etc. The URL has three parts.

**How://Where/What**

**Example:** <http://www.google.com/index.html>

**HTTP(How):** It defines the protocol or scheme by which to access the page. In this case the protocol is hypertext transfer protocol. This protocol is the set of rules by which HTML document is transferred over the web.

[**www.google.com(Where)**](http://www.google.com(Where))**:** It identifies the domain name of the computer where the pages reside. The computer is web server capable of satisfying page results.

**Index.html(What):** It provides the local name(Usually a file name) uniquely identifying the specific page. If no name is specified, the web server where the page is located may supply a default file.

## 5.Web Page (Home Pages)

**Web page** is a document available on World Wide Web. Web Pages are stored on web server and can be viewed using a web browser. A web page can contain huge information including text, graphics, audio, video and hyperlinks. These hyperlinks are the link to other web pages.

**Note:** Collection of linked web pages on a web server is known as **website.** There is unique **Uniform Resource Locator (URL)** is associated with each web page.

### Static Web page

**Static web pages** are also known as flat or stationary web page. They are loaded on the client’s browser as exactly they are stored on the web server. Such web pages contain only static information. User can only read the information but can’t do any modification or interact with the information.

Static web pages are created using only HTML. Static web pages are only used when the information is no more required to be modified.



### Dynamic Web page

**Dynamic web page** shows different information at different point of time. It is possible to change a portion of a web page without loading the entire web page. It has been made possible using **Ajax** technology.

#### SERVER-SIDE DYNAMIC WEB PAGE

It is created by using server-side scripting. There are server-side scripting parameters that determine how to assemble a new web page which also includes setting up of more client-side processing.

#### CLIENT-SIDE DYNAMIC WEB PAGE

It is processed using client side scripting such as JavaScript. And then passed in to **Document Object Model (DOM).**



**6. Basics of HTML**

HTML is a **markup** language for **describing** web documents (web pages).

* HTML stands for **H**yper **T**ext **M**arkup **L**anguage
* A markup language is a set of **markup tags**
* HTML documents are described by **HTML tags**
* Each HTML tag **describes** different document content

## A Small HTML Document

<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h1>My First Heading</h1>
<p>My first paragraph.</p>
</body>
</html>

### Example Explained

* The **<!DOCTYPE html>** declaration defines this document to be HTML5
* The text between **<html>** and **</html>** describes an HTML document
* The text between **<head>** and **</head>** provides information about the document
* The text between **<title>** and **</title>** provides a title for the document
* The text between **<body>** and **</body>** describes the visible page content
* The text between **<h1>** and **</h1>** describes a heading
* The text between **<p>** and **</p>** describes a paragraph

## Web Browsers

The purpose of a web browser (Chrome, IE, Firefox, Safari) is to read HTML documents and display them.

The browser does not display the HTML tags, but uses them to determine how to display the document:



## HTML Page Structure

Below is a visualization of an HTML page structure:

****

**Note:** Only the content inside the <body> section (the white area above) is displayed in a browser.

## HTML Headings

HTML headings are defined with the **<h1>** to **<h6>** tags.

<h1> defines the most important heading. <h6> defines the least important heading:

### Example

<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>

## HTML Paragraphs

HTML paragraphs are defined with the **<p>** tag:

### Example

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

## HTML Links

HTML links are defined with the **<a>** tag:

### Example

<a href="http://www.w3schools.com">This is a link</a>

## HTML Images

HTML images are defined with the **<img>** tag.

The source file (**src**), alternative text (**alt**), and size (**width** and **height**) are provided as **attributes**:

### Example

<img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142">

# HTML Text Formatting

Formatting elements were designed to display special types of text:

* <b> - Bold text
* <strong> - Important text
* <i> - Italic text
* <em> - Emphasized text
* <mark> - Marked text
* <small> - Small text
* <del> - Deleted text
* <ins> - Inserted text
* <sub> - Subscript text
* <sup> - Superscript text

## HTML Comment Tags

You can add comments to your HTML source by using the following syntax:

<!-- Write your comments here -->

HTML Marquee Tag

This tag displays the scrolling text in a marque style.

<marquee>Some text here to write</marquee>

This tag has some attributes tags:

1.Align 2. Behaviour 3.Bgcolor 4. Direction

**Example:**<marquee align=”middle” behavior=”alternate” bgcolor=”blue” direction=”up”>This is an example of marquee tag</marquee>

**HTML Lists**

## Unordered HTML List

An unordered list starts with the **<ul>** tag. Each list item starts with the **<li>** tag.

The list items will be marked with bullets (small black circles) by default:

### Example

<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>

## Unordered HTML List - Choose List Item Marker

The CSS **list-style-type** property is used to define the style of the list item marker:

|  |  |
| --- | --- |
| **Value** | **Description** |
| disc | Sets the list item marker to a bullet (default) |
| circle | Sets the list item marker to a circle |
| square | Sets the list item marker to a square |
| none | The list items will not be marked |

### Example - Disc

<ul style="list-style-type:disc">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>

## Ordered HTML List

An ordered list starts with the **<ol>** tag. Each list item starts with the **<li>** tag.

The list items will be marked with numbers by default:

### Example

<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>

## Ordered HTML List - The Type Attribute

The **type** attribute of the <ol> tag, defines the type of the list item marker:

|  |  |
| --- | --- |
| **Type** | **Description** |
| type="1" | The list items will be numbered with numbers (default) |
| type="A" | The list items will be numbered with uppercase letters |
| type="a" | The list items will be numbered with lowercase letters |
| type="I" | The list items will be numbered with uppercase roman numbers |
| type="i" | The list items will be numbered with lowercase roman numbers |

### Numbers:

<ol type="1">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>

# HTML Tables

* Use the HTML **<table>** element to define a table
* Use the HTML **<tr>** element to define a table row
* Use the HTML **<td>** element to define a table data
* Use the HTML **<th>** element to define a table heading
* Use the HTML **<caption>** element to define a table caption

**Example:**

<table border=”1”>

<tr>

<td> row1,cell1</td>

<td>row1,cell2</td>

</tr>

<tr>

<td>row2,cell1</td>

<td>row2,cell2</td>

</tr>

**Headings in a Table**

Headings in a table are defined with the <th> tag

<table border=”1”>

<tr>

<th> Heading1 </th>

<th>Heading 2 </th>

</tr>

<tr>

<td> row1,cell1</td>

<td>row1,cell2</td>

</tr>

<tr>

<td>row2,cell1</td>

<td>row2,cell2</td>

</tr>

**The Col span and Row span attributes to TD and TH**

<html>

<head>

<title>Table spans</title>

</head>

<body>

<table border width=40%>

<tr>

<td>1.1

<td>1.2

<td>1.3

<td>1.4

</tr>

<tr>

<td>2.1

<td colspan=2>2.2

<td>2.4

</tr>

<tr>

<td>3.1

<td>3.2

<td rowspan=3>3.3

<td>3.4

</tr>

<tr>

<td>4.1

<td>4.2

<td> 4.4

</tr>

<tr>

<td>5.1

<td>5.2

<td>5.4

</tr>

</table>

</html>



**HTML frameset Tag**

HTML frames are used to divide your browser window into multiple sections where each section can load a separate HTML document. A collection of frames in the browser window is known as a frameset. The window is divided into frames in a similar way the tables are organized: into rows and columns.

 To use frames on a page we use <frameset> tag instead of <body> tag. The <frameset> tag defines how to divide the window into frames. The **rows** attribute of <frameset> tag defines horizontal frames and **cols** attribute defines vertical frames. Each frame is indicated by <frame> tag and it defines which HTML document shall open into the frame.

### Example

Following is the example to create three horizontal frames:

<!DOCTYPE html>

<html>

<head>

<title>HTML Frames</title>

</head>

<framesetrows="10%,80%,10%">

<framename="top"src="topframe.html">

<framename="main"src="mainframe.html">

<framename="bottom"src="bottomframe.html">

<noframes>

<body>

 Your browser does not support frames.

</body>

</noframes>

</frameset>

</html>

This will produce following result:



### Example

Let's put above example as follows, here we replaced rows attribute by cols and changed their width. This will create all the three frames vertically:

<!DOCTYPE html>

<html>

<head>

<title>HTML Frames</title>

</head>

<framesetcols="25%,50%,25%">

<framename="left"src="topframe.html">

<framename="center"src="mainframe.html">

<framename="right"src="bottomframe.html">

<noframes>

<body>

 Your browser does not support frames.

</body>

</noframes>

</frameset>

</html>

This will produce following result:



**HTML Forms**

HTML Forms are required when you want to collect some data from the site visitor. For example during user registration you would like to collect information such as name, email address, credit card, etc.

A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc. The back-end application will perform required processing on the passed data based on defined business logic inside the application.

There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML **<form>** tag is used to create an HTML form and it has following syntax:

<form action="Script URL" method="GET|POST"> form elements like input, textarea etc.</form>

## HTML Form Controls

There are different types of form controls that you can use to collect data using HTML form:

* Text Input Controls
* Checkboxes Controls
* Radio Box Controls
* Select Box Controls
* File Select boxes
* Hidden Controls
* Clickable Buttons
* Submit and Reset Button

## Text Input Controls

There are three types of text input used on forms:

* **Single-line text input controls -** This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML **<input>** tag.
* **Password input controls -** This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTMl<input> tag.
* **Multi-line text input controls -** This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML **<textarea>** tag.

## Single-line text input controls

This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag.

### Example

Here is a basic example of a single-line text input used to take first name and last name:

<!DOCTYPE html>

<html>

<head>

<title>Text Input Control</title>

</head>

<body>

<form>

First name: <inputtype="text"name="first\_name"/>

<br>

Last name: <inputtype="text"name="last\_name"/>

</form>

</body>

</html>

This will produce following result:

Top of Form

First name: 
Last name: 

**Bottom of Form**

### Attributes

Following is the list of attributes for <input> tag for creating text field.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for text input control it will be set to **text**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | This can be used to provide an initial value inside the control. |
| size | Allows to specify the width of the text-input control in terms of characters. |
| maxlength | Allows to specify the maximum number of characters a user can enter into the text box. |

## Password input controls

This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML <input> tag but type attribute is set to **password**.

### Example

Here is a basic example of a single-line password input used to take user password:

<!DOCTYPE html>

<html>

<head>

<title>Password Input Control</title>

</head>

<body>

<form>

User ID : <inputtype="text"name="user\_id"/>

<br>

Password: <inputtype="password"name="password"/>

</form>

</body>

</html>

This will produce following result:

Top of Form

User ID :   
Password: 

**Bottom of Form**

### Attributes

Following is the list of attributes for <input> tag for creating password field.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for password input control it will be set to **password**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | This can be used to provide an initial value inside the control. |
| size | Allows to specify the width of the text-input control in terms of characters. |
| maxlength | Allows to specify the maximum number of characters a user can enter into the text box. |

## Multiple-Line Text Input Controls

This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML <textarea> tag.

### Example

Here is a basic example of a multi-line text input used to take item description:

<!DOCTYPE html>

<html>

<head>

<title>Multiple-Line Input Control</title>

</head>

<body>

<form>

Description :<br/>

<textarearows="5"cols="50"name="description">

Enter description here...

</textarea>

</form>

</body>

</html>

This will produce following result:

Top of Form

Description :


**Bottom of Form**

### Attributes

Following is the list of attributes for <textarea> tag.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| rows | Indicates the number of rows of text area box. |
| cols | Indicates the number of columns of text area box |

## Checkbox Control

Checkboxes are used when more than one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to**checkbox**.

### Example

Here is an example HTML code for a form with two checkboxes:

<!DOCTYPE html>

<html>

<head>

<title>Checkbox Control</title>

</head>

<body>

<form>

<inputtype="checkbox"name="maths"value="on">Maths

<inputtype="checkbox"name="physics"value="on"> Physics

</form>

</body>

</html>

This will produce following result:

Top of Form

 Maths  Physics

**Bottom of Form**

### Attributes

Following is the list of attributes for <checkbox> tag.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for checkbox input control it will be set to **checkbox**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | The value that will be used if the checkbox is selected. |
| checked | Set to *checked* if you want to select it by default. |

## Radio Button Control

Radio buttons are used when out of many options, just one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to **radio**.

### Example

Here is example HTML code for a form with two radio buttons:

<!DOCTYPE html>

<html>

<head>

<title>Radio Box Control</title>

</head>

<body>

<form>

<inputtype="radio"name="subject"value="maths">Maths

<inputtype="radio"name="subject"value="physics"> Physics

</form>

</body>

</html>

This will produce following result:

Top of Form

 Maths  Physics

**Bottom of Form**

### Attributes

Following is the list of attributes for radio button.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for checkbox input control it will be set to **radio**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | The value that will be used if the radio box is selected. |
| checked | Set to *checked* if you want to select it by default. |

## Select Box Control

A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options.

### Example

Here is example HTML code for a form with one drop down box

<!DOCTYPE html>

<html>

<head>

<title>Select Box Control</title>

</head>

<body>

<form>

<selectname="dropdown">

<optionvalue="Maths"selected>Maths</option>

<optionvalue="Physics">Physics</option>

</select>

</form>

</body>

</html>

This will produce following result:

Top of Form

  

**Bottom of Form**

### Attributes

Following is the list of important attributes of <select> tag:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| size | This can be used to present a scrolling list box. |
| multiple | If set to "multiple" then allows a user to select multiple items from the menu. |

Following is the list of important attributes of <option> tag:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| value | The value that will be used if an option in the select box box is selected. |
| selected | Specifies that this option should be the initially selected value when the page loads. |
| label | An alternative way of labeling options |

## File Upload Box

If you want to allow a user to upload a file to your web site, you will need to use a file upload box, also known as a file select box. This is also created using the <input> element but type attribute is set to **file**.

### Example

Here is example HTML code for a form with one file upload box:

<!DOCTYPE html>

<html>

<head>

<title>File Upload Box</title>

</head>

<body>

<form>

<inputtype="file"name="fileupload"accept="image/\*"/>

</form>

</body>

</html>

This will produce following result:

**Top of Form**

**Bottom of Form**

### Attributes

Following is the list of important attributes of file upload box:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| accept | Specifies the types of files that the server accepts. |

## Button Controls

There are various ways in HTML to create clickable buttons. You can also create a clickable button using <input> tag by setting its type attribute to **button**. The type attribute can take the following values:

|  |  |
| --- | --- |
| **Type** | **Description** |
| submit | This creates a button that automatically submits a form. |
| reset | This creates a button that automatically resets form controls to their initial values. |
| button | This creates a button that is used to trigger a client-side script when the user clicks that button. |
| image | This creates a clickable button but we can use an image as background of the button. |

### Example

Here is example HTML code for a form with three types of buttons:

<!DOCTYPE html>

<html>

<head>

<title>File Upload Box</title>

</head>

<body>

<form>

<inputtype="submit"name="submit"value="Submit"/>

<inputtype="reset"name="reset"value="Reset"/>

<inputtype="button"name="ok"value="OK"/>

<inputtype="image"name="imagebutton"src="/html/images/logo.png"/>

</form>

</body>

</html>

This will produce following result:

Top of Form

   

**Bottom of Form**

## Hidden Form Controls

Hidden form controls are used to hide data inside the page which later on can be pushed to the server. This control hides inside the code and does not appear on the actual page. For example, following hidden form is being used to keep current page number. When a user will click next page then the value of hidden control will be sent to the web server and there it will decide which page has be displayed next based on the passed current page.

### Example

Here is example HTML code to show the usage of hidden control:

<!DOCTYPE html>

<html>

<head>

<title>File Upload Box</title>

</head>

<body>

<form>

<p>This is page 10</p>

<inputtype="hidden"name="pagename"value="10"/>

<inputtype="submit"name="submit"value="Submit"/>

<inputtype="reset"name="reset"value="Reset"/>

</form>

</body>

</html>

This will produce following result:

Top of Form

This is page 10

 

# 7.HTML Editors

## Write HTML Using Notepad or TextEdit

Web pages can be created and modified by using professional HTML editors.

However, for learning HTML we recommend a simple text editor like Notepad (PC) or TextEdit (Mac).

We believe using a simple text editor is a good way to learn HTML.

Follow the four steps below to create your first web page with Notepad or TextEdit.

## Step 1: Open Notepad (PC)

**Open Notepad in Windows 8 or later:**

Open the **Start Screen** (the window symbol at the bottom left on your screen). Type **Notepad**.

**Open Notepad in Windows 7 or earlier:**

Click **Start** (bottom left on your screen). Click **All Programs**. Click **Accessories**. Click **Notepad**.

## Step 2: Write Some HTML

Write or copy some HTML into Notepad.

<!DOCTYPE html>
<html>
<body>
<h1>My First Heading</h1>
<p>My first paragraph.</p>
</body>
</html>



## Step 3: Save the HTML Page

Save the file on your computer. Select **File > Save as** in the Notepad menu.

Name the file **"index.htm"** and set the encoding to **UTF-8** (which is the preferred encoding for HTML files).



You can use either .htm or .html as file extension. There is no difference, it is up to you.

## Step 4: View the HTML Page in Your Browser

Open the saved HTML file in your favorite browser (double click on the file, or right-click - and choose "Open with").

The result will look much like this:



**8. Introduction to CGI scripts**

# What is CGI ?

* The Common Gateway Interface, or CGI, is a set of standards that define how information is exchanged between the web server and a custom script.
* The CGI specs are currently maintained by the NCSA and NCSA defines CGI is as follows: The Common Gateway Interface, or CGI, is a standard for external gateway programs to interface with information servers such as HTTP servers.
* The current version is CGI/1.1 and CGI/1.2 is under progress.

## Web Browsing

To understand the concept of CGI, lets see what happens when we click a hyper link to browse a particular web page or URL.

* Your browser contacts the HTTP web server and demand for the URL ie. filename.
* Web Server will parse the URL and will look for the filename in if it finds that file then sends back to the browser otherwise sends an error message indicating that you have requested a wrong file.
* Web browser takes response from web server and displays either the received file or error message.

However, it is possible to set up the HTTP server so that whenever a file in a certain directory is requested that file is not sent back; instead it is executed as a program, and whatever that program outputs is sent back for your browser to display. This function is called the Common Gateway Interface or CGI and the programs are called CGI scripts. These CGI programs can be a PERL Script, Shell Script, C or C++ program etc.

## CGI Architecture Diagram

****

## Web Server Support & Configuration

Before you proceed with CGI Programming, make sure that your Web Server supports CGI and it is configured to handle CGI Programs. All the CGI Programs be executed by the HTTP server are kept in a pre-configured directory. This directory is called CGI Directory and by convention it is named as /cgi-bin. By convention PERL CGI files will have extention as **.cgi**.

## First CGI Program

Here is a simple link which is linked to a CGI script called [hello.cgi](http://www.tutorialspoint.com/cgi-bin/hello.cgi). This file is being kept in /cgi-bin/ directory and it has following content. Before running your CGI program make sure you have chage mode of file using **chmod 755 hello.cgi** UNIX command.

|  |
| --- |
| #!/usr/bin/perlprint "Content-type:text/html\r\n\r\n";print '<html>';print '<head>';print '<title>Hello Word - First CGI Program</title>';print '</head>';print '<body>';print '<h2>Hello Word! This is my first CGI program</h2>';print '</body>';print '</html>';1;If you click hello.cgi then this produces following output:Hello Word! This is my first CGI program |

## Simple FORM Example: GET Method

Here is a simple example which passes two values using HTML FORM and submit button. We are going to use same CGI scripthello\_get.cgi to handle this input.

|  |
| --- |
| <FORM action="/cgi-bin/hello\_get.cgi" method="GET">First Name: <input type="text" name="first\_name"><br>Last Name: <input type="text" name="last\_name"><input type="submit" value="Submit"></FORM> |

Here is the actual output of the above form, You enter First and Last Name and then click submit button to see the result.

Top of Form

First Name: 
Last Name:  

# Passing Information using POST method:

A generally more reliable method of passing information to a CGI program is the POST method. This packages the information in exactly the same way as GET methods, but instead of sending it as a text string after a ?in the URL it sends it as a separate message. This message comes into the CGI script in the form of the standard input.

|  |
| --- |
| <FORM action="/cgi-bin/hello\_post.cgi" method="POST">First Name: <input type="text" name="first\_name"><br>Last Name: <input type="text" name="last\_name"><input type="submit" value="Submit"></FORM> |

Here is the actual output of the above form, You enter First and Last Name and then click submit button to see the result.

Top of Form

First Name: 
Last Name:  

# Passing Checkbox Data to CGI Program

Checkboxes are used when more than one option is required to be selected.Here is example HTML code for a form with two checkboxes

|  |
| --- |
| <form action="/cgi-bin/checkbox.cgi" method="POST" target="\_blank"><input type="checkbox" name="maths" value="on">Maths<input type="checkbox" name="physics" value="on"> Physics<input type="submit" value="Select Subject"></form> |

The result of this code is the following form

Top of Form

 Maths  Physics 

# Passing Radio Button Data to CGI Program

Radio Buttons are used when only one option is required to be selected.Here is example HTML code for a form with two radio button:

|  |
| --- |
| <form action="/cgi-bin/radiobutton.cgi" method="POST" target="\_blank"><input type="radio" name="subject" value="maths">Maths<input type="radio" name="subject" value="physics"> Physics<input type="submit" value="Select Subject"></form> |

The result of this code is the following form

Top of Form

 Maths  Physics 

# Passing Text Area Data to CGI Program

TEXTAREA element is used when multiline text has to be passed to the CGI Program.

Here is example HTML code for a form with a TEXTAREA box:

|  |
| --- |
| <form action="/cgi-bin/textarea.cgi" method="POST" target="\_blank"><textarea name="textcontent" cols=40 rows=4>Type your text here...</textarea><input type="submit" value="Submit"></form> |

The result of this code is the following form

Top of Form

 

**9. Using External Media(CSS)**

**Types of style sheets**

## 1.External Style Sheet

With an external style sheet, you can change the look of an entire website by changing just one file!

Each page must include a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section:

**Example**

<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>

An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension.

Here is how the "myStyle.css" looks:

body {
    background-color: lightblue;
}

h1 {
    color: navy;
    margin-left: 20px;
}

## 2.Internal Style Sheet

An internal style sheet may be used if one single page has a unique style.

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

### Example

<head>
<style>
body {
    background-color: linen;
}
h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>

## 3. Inline Styles

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

The example below shows how to change the color and the left margin of a <h1> element:

**Example**

<h1 style="color:blue;margin-left:30px;">This is a heading.</h1>

**Program that demonstrates the types of CSS**

**one.css**

body

{

background-color:lightblue;

}

H2

{

text-align:venter;

color:blue;

font-family:"verdana"

}

p

{

font-family:"sans serif";

color:red;

}

**original one.html file**

<html

<head>

<link rel="sytlesheet" type="text/css" href="one.css">

<style type="text/css">

hr

{

color:sienna

}

p

{

margin-left:20px

}

body

{

background-image:"URL"

}

</style>

</head>

<body>

<h1 style="margin-right:50px; text-decoration:underline; font-family:times"> This is inline CSS</h1>

<h2>This is new header</h2>

<hr>

<p> See the effects of internal css here</p>

</body>

</html><h1 style="margin-right:50px; text-decoration:underline; font-family:times"> This is inline CSS</h1><h1 style="margin-right:50px; text-decoration:underline; font-family:times"> This is inline CSS</h1></style><style type="text/css">**<style type="text/css">Bottom of Form**

**Bottom of Form**

**Bottom of Form**

**Bottom of Form**

**Student Registration Form**

<html>

<head>

<script type="text/javascript" src="validate.js"></script>

</head>

<body>

<form action="#" name="StudentRegistration" onsubmit="return(validate());">

<table cellpadding="2" width="20%" bgcolor="99FFFF" align="center"

 cellspacing="2">

 <tr>

 <td colspan=2>

 <center><font size=4><b>Student Registration Form</b></font></center>

 </td>

 </tr>

 <tr>

 <td>Name</td>

 <td><input type=text name=textnames id="textname" size="30"></td>

 </tr>

 <tr>

 <td>Father Name</td>

 <td><input type="text" name="fathername" id="fathername"

 size="30"></td>

 </tr>

 <tr>

 <td>Postal Address</td>

 <td><input type="text" name="paddress" id="paddress" size="30"></td>

 </tr>

 <tr>

 <td>Personal Address</td>

 <td><input type="text" name="personaladdress"

 id="personaladdress" size="30"></td>

 </tr>

 <tr>

 <td>Sex</td>

 <td><input type="radio" name="sex" value="male" size="10">Male

 <input type="radio" name="sex" value="Female" size="10">Female</td>

 </tr>

 <tr>

 <td>City</td>

 <td><select name="City">

 <option value="-1" selected>select..</option>

 <option value="New Delhi">NEW DELHI</option>

 <option value="Mumbai">MUMBAI</option>

 <option value="Goa">GOA</option>

 <option value="Patna">PATNA</option>

 </select></td>

 </tr>

 <tr>

 <td>Course</td>

 <td><select name="Course">

 <option value="-1" selected>select..</option>

 <option value="B.Tech">B.TECH</option>

 <option value="MCA">MCA</option>

 <option value="MBA">MBA</option>

 <option value="BCA">BCA</option>

 </select></td>

 </tr>

 <tr>

 <td>District</td>

 <td><select name="District">

 <option value="-1" selected>select..</option>

 <option value="Nalanda">NALANDA</option>

 <option value="UP">UP</option>

 <option value="Goa">GOA</option>

 <option value="Patna">PATNA</option>

 </select></td>

 </tr>

 <tr>

 <td>State</td>

 <td><select Name="State">

 <option value="-1" selected>select..</option>

 <option value="New Delhi">NEW DELHI</option>

 <option value="Mumbai">MUMBAI</option>

 <option value="Goa">GOA</option>

 <option value="Bihar">BIHAR</option>

 </select></td>

 </tr>

 <tr>

 <td>PinCode</td>

 <td><input type="text" name="pincode" id="pincode" size="30"></td>

 </tr>

 <tr>

 <td>EmailId</td>

 <td><input type="text" name="emailid" id="emailid" size="30"></td>

 </tr>

 <tr>

 <td>DOB</td>

 <td><input type="text" name="dob" id="dob" size="30"></td>

 </tr>

 <tr>

 <td>MobileNo</td>

 <td><input type="text" name="mobileno" id="mobileno" size="30"></td>

 </tr>

 <tr>

 <td><input type="reset"></td>

 <td colspan="2"><input type="submit" value="Submit Form" /></td>

 </tr>

</table>

</form>

</body>

</html>

Bottom of Form

**Bottom of Form**