MINISTRY OF EDUCATION

DIPLOMA IN

INFORMATION COMMUNICATION TECHNOLOGY

KENYA INSTITUTE OF CURRICULUM DEVELOPMENT

STUDY NOTES

Internet Based Programming

MODULE III: SUBJECT NO 5

Contents

TOPIC 1: INTRODUCTION TO INTERNETBASED PROGRAMMING

T1.1) Internet-based programming

What is Internet programming

Internet programming it a process of web development. It is where _ programming procedures are designed and added into the internet _ websites.

Internet-based, In general, an organization (with or without a physical or brick-and-mortar presence) that has its own website with information content, e-mail, and e-commerce facilities.

T1.2) World Wide Web

The **World Wide Web** (abbreviated **WWW** or **the Web**) is an information space where documents and other web resources are identified by Uniform Resource Locators (URLs), interlinked by hypertext links, and can be accessed via the Internet.

An information system on the Internet that allows documents to be connected to other documents by hypertext links, enabling the user to search for information by moving from one document to another.

Collection of internet resources (such as FTP, telnet, Usenet), hyperlinked text, audio, and video files, and remote sites that can be accessed and searched by browsers based on standards such as HTTP and TCP/IP. Also called the web, it was created in 1989 by the UK physicist Tim Berners-Lee while working at the European Particle Physics Laboratory (called CERN after its French initials Conseil Europeen de Reserches Nucleaires) in Switzerland, as an easier way to access information scattered across the internet.

Website

A location connected to the Internet that maintains one or more pages on the World Wide Web.

A set of interconnected webpages, usually including a homepage, generally located on the same server, and prepared and maintained as a collection of related information by a person, group, or organization. (The website content is stored under a common name, the website name)

Virtual location on WWW, containing several subject or company related webpages and data files accessible through a browser. Each website has its own unique web address (see uniform resource locator) which can be reached through an internet connection. The opening page of a website is usually called homepage which contains hyperlinks to other

pages on the same or other site(s). A single web server may support multiple websites and a single website may reside on multiple web servers, sometimes thousands of miles apart.

URL

Computers

1. Uniform Resource Locator: a protocol for specifying addresses on the Internet.

2. An address that identifies a particular file on the Internet, usually consisting of the protocol, as **http**, followed by the **domain name**.

Short for ***Uniform Resource Locator***. An Internet address (for example, *http://www.hmco.com/trade/* hometrade.html), usually consisting of the access protocol (*http*), the domain name (*www.hmco.com*), and optionally the path to a file (hometrade.html) or resource residing on that server (*trade*).

Domain name/website name

A domain name is a unique name that identifies a website. For example, the domain name of the Tech Terms Computer Dictionary is "techterms.com." Each website has a domain name that serves as an address, which is used to access the website.

Note: All domain names have a domain suffix, such as .com, .net, or .org. The domain suffix helps identify the type of website the domain name represents. For example, ".com" domain names are typically used by commercial website, while ".org" websites are often used by non-profit organizations. Some domain names end with a country code, such as ".dk" (Denmark) or ".se" (Sweden), which helps identify the location and audience of the website.

When you access a website, the domain name is actually translated to an IP address, which defines the server where the website located. This translation is performed dynamically by a service called DNS.

"**Domain Name System**." Domain names serve as memorizable names for websites and other services on the Internet. However, computers access Internet devices by their IP addresses. DNS translates domain names into IP addresses, allowing you to access an Internet location by its domain name.

Thanks to DNS, you can visit a website by typing in the domain name rather than the IP address. For example, to visit the Tech Terms Computer Dictionary, you can simply type "techterms.com" in the address bar of your web browser rather than the IP address (67.43.14.98). It also simplifies email addresses, since DNS translates the domain name (following the "@" symbol) to the appropriate IP address.

To understand how DNS works, you can think of it like the contacts app on your smartphone. When you call a friend, you simply select his or her name from a list. The phone does not actually call the person by name, it calls the person's phone number. DNS works the same way by associating a unique IP address with each domain name.

Unlike your address book, the DNS translation table is not stored in a single location. Instead, the data is stored on millions of servers around the world. When a domain name is registered, it must be assigned at least two nameservers (which can be edited through the domain name registrar at any time). The name-server addresses point to a server that has a directory of domain names and their associated IP addresses. When a computer accesses a website over the Internet, it locates the corresponding name-server and gets the correct IP address for the website.

A name owned by a person or organization and consisting of an alphabetical or alphanumeric sequence followed by a suffix indicating the top-level domain: used as an Internet address to identify the location of particular Web pages:

(Computing) a unique name, corresponding to one or more numeric IP addresses, used to identify a particular web page or set of web pages on the internet

Http

Hypertext transfer protocol: the standard protocol for transferring hypertext documents on the World Wide Web.

Webpage

A document on the World Wide Web, consisting of an HTML file and any related files for scripts and graphics, and often hyperlinked to other documents on the Web. The content of webpages is normally accessed by using a browser.

Electronic (digital) document created with HTML and, therefore, accessible with a browser. In addition to text and graphics, webpages may also contain downloadable data files, audio and video files, and hyperlinks to other pages or sites. A website is usually a collection of webpages.

Browser

A *browser* is an application program that provides a way to look at and interact with all the information on the World Wide Web. The word "*browser*" seems to have originated prior to the Web as a generic term for user interfaces that let you browse (navigate through and read) text files online.

A web *browser* (commonly referred to as a *browser*) is a software application for retrieving, presenting and traversing information resources on the World Wide

Computer program (The two most popular *browsers* are Microsoft Internet Explorer and Firefox. Other major *browsers* include Google Chrome, Apple Safari and Opera) that enables internet users to access, navigate, and search World Wide Web sites. Browsers interpret hypertext links ('hotlinks') and allow documents formatted in hypertext markup language (HTML) to be viewed on the computer screen, and provide many other services including email and downloading and uploading of data, audio, and video files. Also called web browser.

T1.3) Role of web sites in organizations

The Importance of Having a Website for Businesses, Communities and Public Organizations

Not Only for Individuals and Corporations

The importance of having a website cannot be understated whether it's a website for a business, a community or a public organization. Websites have a huge payoff in a variety of ways without a huge investment of time or money. In fact, having a website doesn't even need a ton of technical expertise. What is important is to get a website up and running to begin reaping the rewards.

Advertising

One of the things that highlight the importance of having a website is advertising. A website allows a user to quickly update addresses, phone numbers, products, services, events and any other relevant details. These changes are far less expensive and less time consuming than physically reprinting all sorts of different paper marketing materials. Online advertising on websites also has no limits on size – there can be a huge amount of information included in full detail and color without worrying about added costs of printed materials.

Market Expansion

Having a website allows the market to increase substantially. In fact, the internet knows no limits! People all over the world can view information about the company, group or organization being represented on a website. There will be no limits on the market that can be reached because of time zones, countries, borders or even languages. Many web browsers will translate websites to the language of the user's choice at the click of a button. All of this market expansion is possible at a cost much less than other forms of advertising such as print mailers, newspaper ads or television commercials.

Client Expectations

We are living in an ever increasingly digital world. People expect automatic answers to questions and honestly assume that nearly everyone has their own website. In fact, the internet has largely taken over directory assistance, yellow pages or even local phone books for contact information. If your information cannot be found online, it is almost as

though you don't exist. It is also important that a website is fully functional from mobile devices such as smart phones for clients who are constantly on the go.

Legitimacy of the Group or Business

Because so many different groups – both for profit and non-profit groups—and businesses have their own website, those without lack a certain amount of legitimacy. People who are interested in receiving more information may get the wrong impression on the size, scope and professionalism of a group without a website. Websites for all manner of groups, companies and even individuals are so common that potential clients may be more than put off by groups lacking one. These groups may be seen as technologically stymied at best and inferior quality at worst.

24/7 Availability

Most businesses and groups simply cannot run as consistently as a website can. A website can offer information to interested parties 24 hours a day, 7 days a week with no regard to vacation time or even statutory holidays. People often visit websites outside of normal working hours and this may even be the only time that they have available to find out about your organization. Information simply must be available at all times when people live in such a busy world and a website is the most cost effective way to provide an unlimited amount of information at all times.

Client Relations

Part of the importance of having a website is to strengthen client relations. The website itself helps to develop and strengthen relations with current and prospective customers. The website can proactively target marketing efforts to specific people at certain times. It can let customers become familiar with a company's brand, marketing materials, products, core beliefs, philosophies, history and even employees. It opens up an entirely different venue for communication and dialogue between a group and the people which it serves.

Additional Sales

Lastly, and perhaps most importantly, a website provides a venue for additional sales. Websites can be set up to sell products and accept payments with the help of e-commerce technologies. Even if the group seeking a website is not a physical product, potential clients can research what the group does have to offer and develop more avenues to increase the group's size and reach.

Whether a website is being sought for a public organization, community or business, it is imperative to establish an online presence as soon as possible. Without a website, the group is losing out on a variety of opportunities that other organizations are not just letting slip through their fingers. Jump onboard the internet bandwagon and see what it can do for you.

What are some other benefits of having a business website?

Cost Effective
You know exactly how much your website is going to cost you and it's ongoings – a brick and mortar store, on the other hand, is susceptible to many out of the ordinary occurrences which could blow out the costs such as leaving the lights on, theft, damage, extra staff etc.

A strategically developed website and online presence solution provides tremendous benefits and costing outlines.

Accessible around the clock
Your website and social media accounts are accessible 24/7/365. Imagine that you want to buy from a store. You put in all the effort required to go to the store, but when you get there, it's closed. We all know how irate we feel in that situation. You'll think twice about going back given the bad taste its left (ok might have been your fault for not checking but hey, this is proving the point here!). You will just find another store that is more easily accessible.

Since your website is operational around the clock, from the convenience of the local coffee shop, their couch or their bed, your customers and clients can easily access your website and services.

Convenient
What is more convenient: driving outside to look for different stores that are available to shop in, or sitting in the comfort of your own home and shopping for the products you're looking for? Pretty obvious answer, unless you like aimlessly driving around. Smart businesses realise this and thus have their own website housing their products and services so that potential customers can browse online for the products they want to purchase.

Credibility
By building a website you are giving your business the opportunity to tell consumers why they should trust you and the testimonials and facts to back up those opportunities. Believe it or not, most people will search the internet for a product or service before the purchase to check the credibility first. When you provide good service or product, positive word-of-mouth about your business is likely to spread. Which in turn, delivers more repeat and new business.

People tend to trust a business after they have done business with it. Using your website, you can continuously serve consumers online and increase your credibility as a business owner.

Sales
Without sales, or selling more than you spend, your business is doomed. By having an online presence you allow for the sale of your products or services around the clock to whoever whenever with no or hardly any limitations; Unless you run out of stock or overworked, but that's a good problem to have right! Giving your business the online presence it deserves is crucial to your brand and accountants smile.

In short, being visible worldwide means you are very likely to gain more customers. The more customers and visitors you have, the more sales you will generate. The more sales you generate the happier you and your shareholders will be!

Marketing
Having a website and online presence strategy allows you to market your business online. There are lots of marketing strategies you can use to advertise and market your business. All online marketing strategies have been proven to be effective. Which ones you choose depends on the type of business you are in. Speak to us to see which are best for your business.

TOPIC 2: WEB PROGRAMMING

T2.1) Explanation of Web programming

Web programming refers to the writing, markup and coding involved in Web development, which includes Web content, Web client and server scripting and network security. Or Writing the necessary source code to create a website

Web development refers to building, creating, and an maintaining websites. It includes aspects such as web design, web publishing, web programming, and database management.

T2.2) Approaches to web programming

Web programming can be briefly categorized into client and server coding. The client side needs programming related to accessing data from users and providing information. It also needs to ensure there are enough plug ins to enrich user experience in a graphic user interface, including security measures.

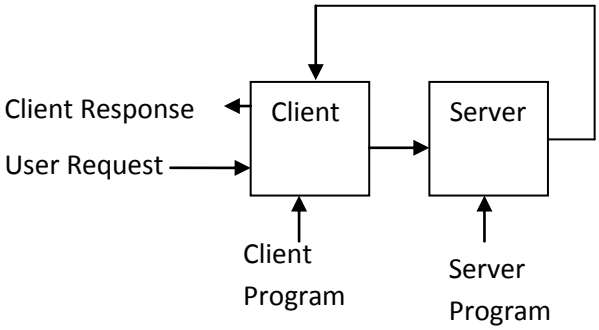
Web development is all about communication. In this case, communication between two (2) parties, over the HTTP protocol:

- The **Server** - This party is responsible for **serving** pages.
- The **Client** - This party *requests* pages from the **Server**, and displays them to the user. In most cases, the client is a **web browser**.
    - The **User** - The user *uses* the **Client** in order to surf the web, fill in forms, watch videos online, etc.

Each side's programming, refers to code which runs at the specific machine, the server's or the client's.

Basic Example

1. The **User** opens his web browser (the **Client**).
2. The **User** browses to http://google.com.
3. The **Client** (on the behalf of the **User**), sends a request to http://google.com (the **Server**), for their home page.
4. The **Server** then acknowledges the request, and replies the client with some meta-data (called *headers*), followed by the page's source.
5. The **Client** then receives the page's source, and *renders* it into a human viewable website.
6. The **User** types `Stack Overflow` into the search bar, and presses `Enter`
7. The **Client** submits that data to the **Server**.
8. The **Server** processes that data, and replies with a page matching the search results.
9. The **Client**, once again, renders that page for the **User** to view.



Server-side Programming

Server-side programming, is the general name for the kinds of programs which are run on the **Server**.

Uses

- Process user input.
- Display pages.
- Structure web applications.
- Interact with permanent storage (SQL, files).

Example Languages

- PHP
- ASP.Net in C#, C++, or Visual Basic.
- Nearly any language (C++, C#, Java). These were not designed specifically for the task, but are now often used for application-level web services.

Client-side programming

Much like the server-side, Client-side programming is the name for all of the programs which are run on the **Client**.

Uses

- Make interactive webpages.
- Make stuff happen dynamically on the web page.
- Interact with temporary storage, and local storage (Cookies, localStorage).
- Send requests to the server, and retrieve data from it.
- Provide a remote service for client-side applications, such as software registration, content delivery, or remote multi-player gaming.

Example languages

- JavaScript (primarily)
- HTML*

- CSS*
- Any language running on a client device that interacts with a remote service is a client-side language.

T2.3) Web programming languages

There are several languages that can be used for server-side programming:

- PHP
- ASP.NET (C# OR Visual Basic)
- C++
- Java and JSP
- Python
- Ruby on Rails and so on.

There are many client-side scripting languages too.

- JavaScript
- VBScript
- HTML (Structure)
- CSS (Designing)
- AJAX
- jQuery etc.

(Some other languages also can be used on the basis of the modeling/designing /graphics/animations and for extra functionalities.)

T2.4) Criteria for choosing a web programming language

How to Choose the Right Programming Language for a Web App

What really motivates people to go with one programming language or another when they start product development? Actually, it's not an easy choice and it's especially difficult for non-technical people who usually have to rely on their partners and advisors in this. This choice might even define the success of your business, so it's definitely worth weighing up all the pros and cons of each option.

Criteria

First, you should consider the following criteria:

- Efficiency of the language itself;
- Available platforms and frameworks;
- Community support;

- How difficult the language is to learn. Mind you, if the language syntax and rules are complex, you'll get more capable developers as only the best ones can get through. On the other hand, there might be the scarcity of developers, which we see with the languages like Scala, Erlang, Lisp, Prolog, Golang and others.
- Strategic issues: which tech stack your potential acquisition company is using, or which technology stack they would prefer to have.

Factors to consider when choosing a programming language

1. **Popularity**. This is a very important one. A good place to start is the Tiobe index. You are more likely to find people to collaborate with if you use a popular language. You are also more likely to find reference material and other help. Unfortunately, the most popular language globally may not be a good match for your problem domain.
2. **Language-domain match**. Choose one that matches your problem domain. You can do this by looking at what other people in your field are using (after adjusting for popularity, so don't think the match with Java is good simply because a lot of people are using Java) or by looking at some code that solves problems you are likely to have and seeing how natural the mapping is**.**
3. **Availability of libraries**. Some would argue that this is the same as the point above, but I don't think so. If there's a library that solves your problem well, you'll put up with some ugly calling conventions or hassle in the language.
4. **Efficiency**. Languages aren't fast - compilers are efficient. Look at the efficiency of compilers or interpreters for your language. Be aware that interpreted code will run an order of magnitude slower than compiled code as a rule of thumb.
5. **Expressiveness**. The number of lines of code you create per hour is not a strong function of language, so favour languages that are expressive or powerful
6. **Project-size**. Do you want to be programming in the large or programming in the small? Choose a language that supports your use case.
7. **Tool support**. Popularity usually buys tool support (and some languages are easier to write tools for). If you are a tool-oriented user, choose a language with good tool support. Just read this article on tool mavens vs language mavens before you make a choice.

PHP

It's hard to argue against the popularity of PHP, the scripting language which is used by many and has a vibrant developer community. It is used not only for WordPress development but also for complex systems so even Facebook utilizes PHP, which says a lot about its level of credibility.

Since PHP does not use too much of a system's resources in order to run, it operates quite well compared with other scripting languages. Hosting it is also very easy as lots of hosts provide support for PHP.

Pros of PHP:

- It is highly accessible, there are a lot of frameworks written in it like Symfony, Yii, Laravel, CakePHP, Kohana, Zend Framework, PhalconPHP, to name a few.
- It enables fast implementation of complex solutions. And the faster a new application enters the market, the higher is your cost-efficiency and the greater is your competitive advantage.
- Thanks to the popularity of PHP, it offers great flexibility during and after the initial project, as the number of resources is continuously growing.
- It has sound host support.
- PHP can be embedded in HTML.
- PHP7 was released in December 2015 ahead of deadlines and its performance has grown substantially. Totally different architecture sits in its kernel, so it changes the way how the interpreted code is being processed on operation system level, and this actually causes this increase in performance.
- There are performance accelerators available for PHP. For example, Facebook came up with HipHop, a compiler of PHP into C which then turned into executable files. Eventually, HipHop project came up with a virtual machine transforming PHP script into bytecode and doing its own internal performance optimization. There are also such solutions as APC, ionCube, Turck MMCache, Zend Opcache, XCache etc.
- Websites powered by PHP: Facebook, WordPress, Wikipedia, Mailchimp, Flickr, Yahoo!, Tumblr etc.

Cons of PHP

- When someone claims that they know PHP, it's hard to assess their level of development skill, so you'll need to dig deeper. PHP is so easy to learn on the basic level that people without proper computer science background could go and start developing, not knowing much about algorithms, time complexity, application performance optimization, advanced database querying, systems scalability and other important aspects.
- It's not as cross-platform as some of its competitors. The language is compatible with UNIX based OS and Windows OS.

Python

Pros of Python

- It requires less time to develop than, for instance, in Java, as programs are shorter. So faster development and also deeper prototyping would give you a competitive advantage.
- Python has a quite simple syntax and easy-to-use data structures.
- It performs well across different platforms.
- The language has good scalability.
- Apps powered or supported by Python include Instagram, Pinterest, Django, Google, NASA, Yahoo, etc. You can also check out more organizations using Python.

Cons of Python

- Performance could be better. Programs in Python are slower than in Java, for example. This is obviously because it is one of the interpreted languages, which are normally slower than compiled languages.

Ruby

According to Stack Overflow 2015 Developer Survey, Ruby is the technology which pays best in Western Europe, which says a lot about its adoption scale.

Pros of Ruby

- Ruby prefers convention over configuration which makes applications easier to develop and understand. Clear syntax makes the programming process much faster and more efficient.
- RoR framework has very simple structure, it's easy for developers.
- Don't Repeat Yourself, or DRY Principle. By not writing the same information over and over again, the code is more maintainable, more extensible, and less buggy.
- You can develop MVP very fast on Ruby, it takes less time than on other languages.
- It has big development community.
- Testing plays a big role in development. Minitest library goes together with Ruby, so when you develop something, it's ready for testing. As a result, you get a neat secure product. There's also a very powerful testing framework for ROR called RSpec.
- Platforms that are supported by Ruby include Hulu, Shopify, Twitter, Scribd, Crunchbase, Groupon, etc.
- There are many libraries, gems for any case, so when you are writing the code, you can use a ready-made solution, so you don't need to write it from scratch. The main repository contains over 100,000 gems. https://rubygems.org/stats

Cons of Ruby

- The performance is not as fast as on PHP or JavaScript.
- As it's the technology that pays really well, development might turn out to be more expensive than in other languages.

Java

Java Pros

- The core value proposition of the Java platform says "Write once, run anywhere". So the most important promise of Java technology is that you only have to write your application once and then you'll be able to run it anywhere. JVM is a universal engine, you can use it for anything, it will work everywhere. It's the most cross-platform language.
- It was arguably built with security in mind, so security features are one of Java's advantages.
- It has a big and dedicated development community.
- Outstanding performance.

Java Cons

- Bigger projects can be difficult to compile and build.

- Development is more expensive than in PHP or Python.

If you are still sitting on the fence with your choice, go through this fun flowchart and find out which language is better for your product And remember, you just need to find a good team with talented developers. Then no matter which programming language you go for, you can be confident in the final result.

T2.5) Common web programming interfaces

Application program interface (API)

An ***a**pplication **p**rogram **i**nterface* (**API**) is a set of routines, protocols, and tools for building software applications. Basically, an API specifies how software components should interact. Additionally, APIs are used when programming graphical user interface (GUI) components. A good API makes it easier to develop a program by providing all the building blocks. A programmer then puts the blocks together.

A **web API** is an application programming interface (API) for either a web server or a web browser. It is a web development concept, usually limited to a web application's client-side (including any web frameworks being used), and thus usually does not include web server or browser implementation details such as SAPIs or web browser engine APIs unless publicly accessible by a remote web application.

A **web framework** (**WF**) or **web application framework** (**WAF**) is a software framework that is designed to support the development of web applications including web services, web resources, and web APIs.
Server Application Programming Interface (**SAPI**) is the direct module interface to web servers such as the Apache HTTP Server, Microsoft IIS, and Oracle iPlanet Web Server.

CGI - Common Gateway Interface

CGI is the abbreviation of ***C**ommon **G**ateway **I**nterface.* It is a specification for transferring information between a World Wide Web server and a CGI program. A CGI program is any program designed to accept and return data that conforms to the CGI specification. The program could be written in any programming language, including C, Perl, Java, or Visual Basic.

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.

CGI Programs

CGI programs are the most common way for Web servers to interact dynamically with users. Many HTML pages that contain forms, for example, use a CGI program to process the form's data once it's submitted. Another increasingly common way to provide dynamic feedback for Web users is to include scripts or programs that run on the user's machine rather than the Web server. These programs can be Java applets, Java scripts, or ActiveX controls. These technologies are known collectively as *client-side* solutions, while the use of CGI is a *server-side* solution because the processing occurs on the Web server.

One problem with CGI is that each time a CGI script is executed, a new process is started. For busy websites, this can slow down the server noticeably. A more efficient solution, but one that it is also more difficult to implement, is to use the server's API, such as ISAPI or NSAPI. Another increasingly popular solution is to use Java servlets.

Web Browsing

To understand the concept of CGI, let's see what happens when we click a hyperlink to browse a particular web page or URL.

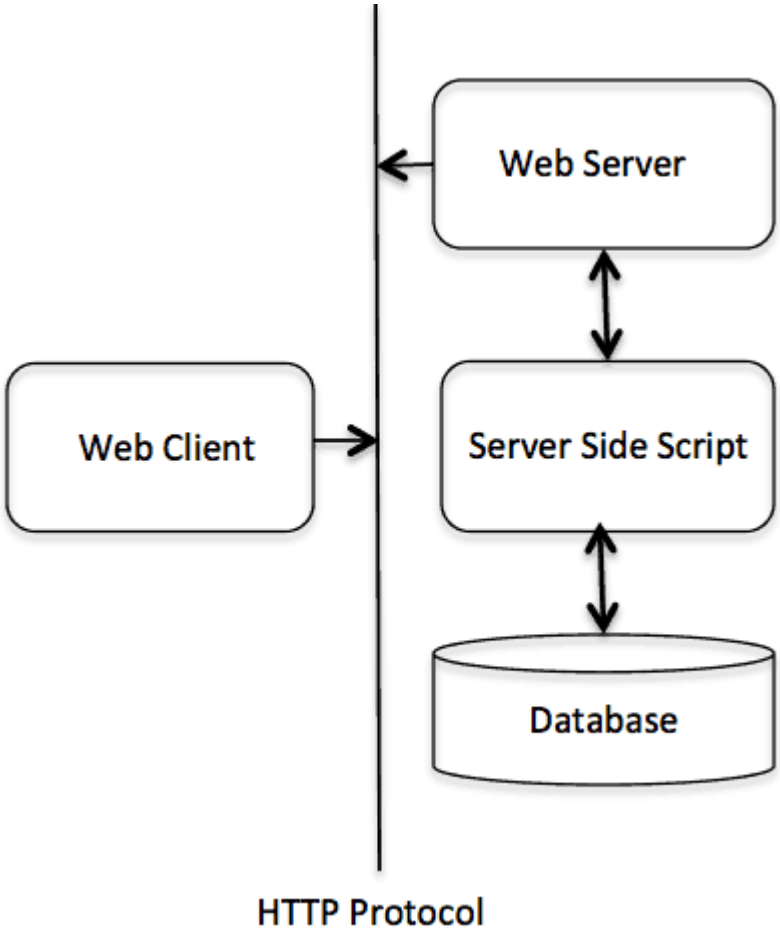
- Your browser contacts the HTTP web server and demand for the URL ie. filename.
- Web Server will parse the URL and will look for the filename. If it finds requested file then web server sends that file back to the browser otherwise sends an error message indicating that you have requested a wrong file.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

However, it is possible to set up the HTTP server in such a way that whenever a file in a certain directory is requested, that file is not sent back; instead it is executed as a program, and produced output from the program is sent back to your browser to display.

The Common Gateway Interface (CGI) is a standard protocol for enabling applications (called CGI programs or CGI scripts) to interact with Web servers and with clients. These CGI programs can be a written in Python, PERL, Shell, C or C++ etc.

CGI Architecture Diagram

The following simple program shows a simple architecture of CGI −



Web Server 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 to 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 /var/www/cgi-bin. By convention CGI files will have extension as **.cgi**, though they are C++ executable.

By default, Apache Web Server is configured to run CGI programs in /var/www/cgi-bin. If you want to specify any other directory to run your CGI scripts, you can modify the following section in the httpd.conf file −

```
<Directory "/var/www/cgi-bin">
   AllowOverride None
   Options ExecCGI
   Order allow,deny
   Allow from all
</Directory>

<Directory "/var/www/cgi-bin">
   Options All
</Directory>
```

Here, I assume that you have Web Server up and running successfully and you are able to run any other CGI program like Perl or Shell etc.

Common Client Interface (CCI)

The Common Client Interface (CCI) of the Connection Architecture provides a standard interface that allows developers to communicate with any number of Enterprise Information Systems (EISs) through their specific resource adapters, using a generic programming style. The CCI defines a set of interfaces and classes whose methods allow a client to perform typical data access operations.

TOPIC 3: HTML CODING

T3.1) Introduction to HTML

What is HTML?

HTML is a language for describing web pages.

- HTML stands for **H**yper **T**ext **M**arkup **L**anguage
- HTML is not a programming language, it is a **markup language**
- A markup language is a set of **markup tags**
- The purpose of the tags are to **describe page content**

It is the standard protocol for formatting and displaying documents on the World Wide Web.

Note

HTML (Hypertext Markup Language) is the set of markup symbols or codes inserted in a file intended for display on a World Wide Web browser page. The markup tells the Web browser how to display a Web page's words and images for the user. Each individual markup code is referred to as an element (but many people also refer to it as a tag). Some elements come in pairs that indicate when some display effect is to begin and when it is to end.

HTML is a formal Recommendation by the World Wide Web Consortium (W3C) and is generally adhered to by the major browsers, Microsoft's Internet Explorer and Netscape's Navigator, which also provide some additional non-standard codes.

Markup refers to the sequence of characters or other symbols that you insert at certain places in a text or word processing file to indicate how the file should look when it is printed or displayed or to describe the document's logical structure. The markup indicators are often called "tags." For example, this particular paragraph is preceded by a:

A **tag** is a generic term for a language element descriptor. The set of tags for a document or other unit of information is sometimes referred to as markup

T3.2) HTML structure

HTML Page Structure

Below is a visualization of an HTML page structure:

```
<html>
<body>
<h1>This a Heading</h1>

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

This is another paragraph.

</body>

</html>

Example Explained

What you just made is a skeleton html document. This is the minimum required information for a web document and all web documents should contain these basic components. The first tag in your html document is `<html>`. This tag tells your browser that this is the start of an html document. The last tag in your document is `</html>`. This tag tells your browser that this is the end of the html document.
The text between the `<head>` tag and the `</head>` tag is header information. Header information is not displayed in the browser window.
The text between the `<title>` tags is the title of your document. The `<title>` tag is used to uniquely identify each document and is also displayed in the title bar of the browser window.
The text between the `<body>` tags is the text that will be displayed in your browser.
The text between the `<b>` and `</b>` tags will be displayed in a bold font.

HTM or HTML Extension?

When you save an HTML file, you can use either the .htm or the .html extension. The .htm extension comes from the past when some of the commonly used software only allowed three letter extensions. It is perfectly safe to use either .html or .htm, but be consistent. **mypage.htm** and mypage.html are treated as different files by the browser.

How to View HTML Source

A good way to learn HTML is to look at how other people have coded their html pages. To find out, simply click on the View option in your browsers toolbar and select Source or Page Source. This will open a window that shows you the actual HTML of the page. Go ahead and view the source html for this page.

T3.3) HTML Tags

Tags, Attribute and Elements

Html Tags

A **tag** is a generic term for a language element descriptor. The set of tags for a document or other unit of information is sometimes referred to as markup.

HTML tags are the hidden *keywords* within a web page that define how your web browser must format and display the content.

HTML markup tags are usually called HTML tags

- HTML tags are keywords (tag names) surrounded by **angle brackets** like <html>
- HTML tags normally **come in pairs** like and
- The first tag in a pair is the **start tag,** the second tag is the **end tag**
- The end tag is written like the start tag, with a **forward slash** before the tag name
- Start and end tags are also called **opening tags** and **closing tags**

HTML Elements

"HTML tags" and "HTML elements" are often used to describe the same thing.

But strictly speaking, an HTML element is everything between the start tag and the end tag, including the tags:

HTML Element:

This is a paragraph.

HTML Attributes

Attributes provide additional information about HTML elements.

HTML Attributes

- HTML elements can have **attributes**
- Attributes provide **additional information** about an element
- Attributes are always specified in **the start tag**
- Attributes come in name/value pairs like: **name="value"**

Attribute Example

HTML links are defined with the <a> tag. The link address is specified in the **href attribute**:

Example

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

Attributes allow you to customise a tag, and are defined within the opening tag, for example:
`<img src="image1.jpg">` or `<p align="center"> ... </p>`

Attributes are often assigned a value using the equals sign, such as `border="0"` or `width="50%"`, but there are some that only need to be declared in the tag like this: `<hr noshade>`.

Most attributes are optional for most tags, and are only used when you want to change something about the default way a tag is displayed by the browser. However, some tags

such as the `<img>` tag has required attributes such as `src` and `alt` which are needed in order for the browser to display the web page properly.

HTML tags Types

1. Types

HTML tags can be of two types. They are

1. Paired Tags
2. Unpaired Tags

Paired Tags:

A tag is said to be a paired tag if the text is placed between a tag and its companion tag. In paired tags, the first tag is referred to as *Opening Tag* and the second tag is referred to as *Closing Tag.*

Example

```
<i>This text is in italics. </i>
```

Note: Here is called opening tag. and is called closing tag.

Unpaired Tags:

An unpaired tag does not have a companion tag. Unpaired tags are also known as *Singular* or *Stand-Alone* Tags.

Example

```
<br> , <hr>
```

etc. These tags does not require any companion tag.

2. Tags Based on their utility

We can differentiate tags based on the purpose they used. Basically we have three types here

Formatting tags

We manage the size of the font, underline part of the text, make the text bold etc by using tags like <font>, <u>,<b> etc.

Page Structure tags

Description, title, head, body etc. are part of the page structure tags. They are part of the basic html page and does not directly affect the formatting of text or image.

Control tags

Form tags, Script tags, Radio buttons etc are part of the control tags

HTML tags classification

1. Essential HTML Tags

There are four sets of HTML tags that are needed to form the basic structure for every HTML file:

- <html></html>
- <head></head>
- <title></title>
- <body></body>

Definition - `<html> </html>`
This basically defines the document as web page. It also identifies the beginning and end of the HTML document. All other tags must fall between the `html` tags.

Header - `<head> </head>`

The header contains information about the document that will not appear on the actual page, such as the title of the document, the author, which stylesheet to use and also meta tags.

Title - `<title> </title>`

The `title` tag defines the title that will appear in the title bar of your web browser. The `title` must appear between the `head` tags.

Body - `<body> </body>`

The `body` tags contain all the information and other visible content on the page. All your images, links and plain text must go between the `<body>` and `</body>` tags.

These four tags are special. There must only be one set of each and they must be in the correct order like in the example below. The fun and creative part comes when using the basic tags for adding content and headings.

Example:
Below is a basic html document. Notice that everything falls between the `html` tags, the `title` appears within the `head` of the document, and that the `body` comes after the `head`.

```
<html>
 <head>
  <title>My Page Title</title>
 </head>
 <body>

  This is where all my web page content goes!

 </body>
</html>
```

2. HTML Basics

Once you have the essential tags sorted out, there are a number of basic tags you will use a lot in order to add content such as:

- Headings and Subheadings
- Paragraphs of Text
- Links to Other Pages
- Images or Photographs

All of these must appear between the `<body>` and `</body>` tags, and you can learn more about them by following the links shown.

Headings

```
<h1> A Heading </h1>
```

Use headings for titles and subtitles, and make some text stand out from others. See text formatting tags.

Paragraphs

```
<p> Some text </p>
```

Most content on a simple web page will appear in paragraphs or sections. See division tags.

Links

```
<a href="home.html"> My homepage </a>
```

Links are necessary for users to jump from one page to another. See linking tags.

Images

```
<img src="photo.jpg">
```

Adding your holiday photos or other images to your web page is fairly simple. See image tags.

Example:

Below is an example of some of the basic tags explained above

```
<html>
 <head>
  <title>this is the title</title>
 </head>
```

```
 <body>
  <h1>My Heading</h1>
  <p>This is the first paragraph of text.</p>
  <p>This is the second paragraph of text.</p>
  <p>An image: <img src="photo.jpg"> </p>
  <p>A link: <a href="http://www.simplehtmlguide.com"> html guide
</a></p>
 </body>
</html>
```

3. Sections, Divisions & Lines

These are the tags used to divide your page up into sections. Effective use of these tags will mean that the page has a good structure and layout, making it more user friendly and easier to read.

Division - `<div> </div>`

The `div` tag defines a section or division within a HTML file. It typically contains headings, paragraphs, tables or other elements that need to be grouped together. Commonly used with css by setting the `<div class="?">` attribute to set the look and feel of a section of your web page.

Paragraph - `<p> </p>`

Used to define paragraphs of text, much like you would see in a book. A lot of text can appear within the `<p>` and `</p>` tags, and browsers will automatically wrap the text onto the next line once it reaches the edge of the screen. When another `<p>` tag is used to start the next paragraph, the browser will add some blank space between the paragraphs. It has the following attributes:

`align="?"` - Alignment of text in the paragraph: `left`, `center` or `right` (*)

`width="?"` - Paragraph will occupy a fixed width or percentage of the page, default 100%

Span - `<span> </span>`

Used to group inline elements together, such as a few words within a sentence, in order to apply a css style to those words only. The `span` tag can be used within `div` and `p` tags as it does not create a new horizontal block boundary.

Line Break - `<br>`

Equivalent to one carriage return, it is used to start text on a new line. Multiple `<br>` tags in a row will create a large vertical space on a web page.

Horizontal Line - `<hr>`

The horizontal rule, often referred to as the HTML line separator tag, creates a horizontal line commonly used to visually separate sections on a page. It has the following attributes:

`width="??%"` - The line will occupy a fixed width or percentage of the default 100% width.

`color="#??????"` - Colour of the line (*)

`noshade` - Prevent the 3D 'etched' look and create a flat, solid line separator.

No Breaks - `<nobr> </nobr>`

If for some reason you want text to continue in one straight line, and not to wrap at the edge of the screen screen, you can use the `nobr`. Note: this will force a user to scroll to the right to see the rest of the line, which is considered bad design.

Example:

Below is an example of some of the tags explained above

```
<html>
 <body>
  <p align="right">The first paragraph of text, aligned to the
right.</p>
  <p>This text is now in the second paragraph.
  <br>A new line, but still part of the second paragraph.</p>
  <p>A third paragraph, and then a horizontal line</p>
  <hr>
  <p>Some modified horizontal lines:</p>
  <hr width="50%" size="8" align="center">
  <hr noshade>
 </body>
</html>
```

4. Text Formatting Tags

The following HTML tags are used to format the appearance of the text on your web page. This can jazz up the look of the web page, *however*, too much variety in the text formatting can also look displeasing.

Header - `<h?> </h?>`

There are 6 levels of headings available, from `h1` for the largest and most important heading, down to `h6` for the smallest heading.

Bold - `<b> </b>`

The text in between the tags will be bold, and stand out against text around it, the same as in a word processor.

Italic - `<i> </i>`

Also working the same way as a word processor, italics displays the text at a slight angle.

Underline - `<u> </u>`

Again, the same as underline in a word processor. Note that html links are already underlined and don't need the extra tag.

Strike-out - `<strike> </strike>`

Puts a line right through the centre of the text, crossing it out. Often used to show that text is old and no longer relevant. Also works by using `<s> </s>` instead.

Preformatted Text - `<pre> </pre>`

Any text between the `pre` tags, including spaces, carriage returns and punctuation, will appear in the browser as it would in a text editor (normally browsers ignore multiple spaces)

Source Code - `<code> </code>`

Text is displayed in a fixed-width font, commonly used when showing source code. I have used it on this site, along with stylesheets, to show all tags.

Typewriter Text - `<tt> </tt>`

The text appears to have been typed by a typewriter, in a fixed-width font. (*)

Block Quote - `<blockquote> </blockquote>`

Defines a long quotation, and the quote is displayed with an extra wide margin on the left hand side of the block quote.

Small - `<small> </small>`

Instead of having to set a font size, you can use the `small` tag to render text slightly smaller than the text around it. Useful for displaying the 'fine-print'.

Font Colour - `<font color="#??????"> </font>` (*)

Change the colour of a few words or a section of text. The 6 question marks represent the hex color code, see this list of colours and codes for some samples. (*)

Font Size - `<font size="?"> </font>`

Replace the ? with a number from 1 to 7 to change the size of the font. One being the smallest and seven the largest. (*)

Font Size Change - `<font size="+/-?"> </font>`

For an immediate change of font size with respect to the font size preceding it, this tag increase or decreases the size of the font by the number you specify. Eg: `<font size="-1">`Some Text`</font>` (*)

Change Font Face - `<font face="?"> </font>`

To show text in a particular font, use the font name such "Helvetica" or "Arial" or "Courier". Be aware that using some fancy font from your computer means that the person viewing that page must also have that font installed on their computer too, otherwise it will look totally different to them. (*)

Centre - `<center> </center>`

A useful tag, as it says, it makes everything in between the tags centred (in the middle of the page). (*)

Emphasis - `<em> </em>`

Used to emphasize text, which usually appears in italics, but can vary according to your browser.

Strong Emphasis - `<strong> </strong>`

Used to emphasize text more, which usually appears in bold, but can vary according to your browser.

5. Images

Images are used in HTML documents to one: make the page visually effective and two: display information. Images can also be used as links, but this is discussed in the topic on linking.

Although images are good for a number of things, a page with too many images often looks too cluttered and can take too long to load, which can be frustrating, and as a business aspect it could lose clients.

An image - `<img src="url">`

To display an image you need to specify the URL of the image using the `src` attribute, replacing `url` with the filename of your image. There are several ways this can be done:

`src="picture.jpg"` - the filename if the image is in the same directory as the html file.

`src="images/picture.jpg"` - a relative path when the image is in another directory.

`src="http://www.simplehtmlguide.com/images/photo.jpg"` - a full URL can also be used.

Alternate Text - `<img ... alt="?">`

The `alt` attribute defines the text shown in place of an image when the image cannot load. This is actually a required attribute for valid html, and should briefly describe what the image normally would.

Image Size - `<img ... width="?" height="?">`

An image will normally be shown actual size, but by using the `width` and `height` attributes you can change the displayed size. You can specify the size in pixels or as a percentage. Tip: specify the size using the actual size of the image in pixels to force browsers to allocate space for the image before it is even loaded, ensuring you page layout remains the same with or without images displayed.

Border - `<img ... border="?">`

Add a border by specifying the thickness in pixels. You can also set `border="0"` to remove the border added when images are used as links. (*)

Image Alignment - `<img ... align="?">`

By default an image appears at the place specified in the html code(as with any other tag). However, you can align an image with the surrounding text or paragraph by setting any of `align="left | right | top | bottom | middle"`. (*)

Spacing - `<img ... vspace="?" hspace="?">`

Adjust the whitespace (or runaround space) around an image, in pixels. Use `vspace` to adjust the vertical spacing above and below, or `hspace` for the left and right sides. (*)

Example:
Show an image using html

```
<html>
 <body>
  <p><img src="photo.jpg" alt="parakeet"></p>
 </body>
</html>
```

6. HTML Linking Tags

Learn how to create links on your web page. Links allow you to jump from one page to another by clicking on the link text. You can also jump to places on the same page (called fragments), to different sections of your site, or to another web site alltogether.
Basic Link - `<a href="url">link text</a>`

There are two main parts to a link tag: the text a user can click, and the web address they go to if they click it. The bit between the `<a>` and `</a>` tags is the link text, and is generally displayed in blue and underlined by web browsers. The `href="url"` part is the web address, where `url` can be set in several ways:

`href="example.html"` - another page in the current directory

`href="example/page.html"` - a relative location

`href="http://www.example.com/page.html"` - a full address (URL).

Link to a Fragment - `<a href="#fragment">link</a>`

It is often usefull to link to other places on the same webpage, such as other sections or chapters further down the page. The technical term for this is linking to a Fragment, where browsers will automatically try and scroll to that part of the page.
Fragments first need to be defined somewhere in a webpage by giving them a name, for example `<a name="fragment_name">`, then links to this fragment are created by using the hash (#) character: `<a href="#fragment_name">Link</a>`. To link to a fragment on another page you would simply append the fragment name to the address, for example: `href="example.html#fragment_name"`.

Target Window - `<a href="url" target="???">link</a>`

You may not always want to link to a page and have it load up over the one you are currently viewing. Thats where the `target` attribute comes in handy. By setting the `target="_BLANK"` the page you link to will load up in a new window (or new tab in some newer browsers). Similary, `"_self", "_parent", or "_top"` will open the link in the current window, the parent window (used with frames) or the top level window, respectively.

Image as a Link - `<a href="url"><img ...></a>`

By placing an image tag between the <a> and tags, you can turn an image into a link, and clicking on that image will then load the referenced page. You may notice that the image gets a blue border just as link text became underlined. This can be resolved by setting the border="0" attribute of the image, or using css.

Email Link -

A special kind of link, the mailto notation link instructs the browser to compose and email to the specified address using the default email program. It but does not actually send any emails automatically. You can also set a subject for the email by using email me. You may notice that I have used this type of link over on the contact me page.

7. HTML Lists

Learn how to create lists on a web page. Lists are the preferred way to display items one after the other, instead of using <br> tags. Lists have a tag to start and end the list itself, as well as a tag for each item in the list.

There are three types of lists, *ordered*, *unordered* and *definition* lists.

Unordered Lists

An unordered list is a bulleted list, similar to the menu on the right (although the menu has been altered using stylesheets to use images instead of the standard bullets.)
Define Unordered List - ...

Use the <ul> tags to define the start and end of an unordered list. A number of list items (li elements) will go within the ul tags.

Unordered List Item - some item

Add the text for each item in between some <li> and tags. Each list item must have its own li tags.

Bullet Type <ul type="disc | circle | square">

By default a browser will show a round bullet. This can be changed by using the type attribute of the ul tag, which will change the bullet type for the entire list.

Item Type <li type="?">

You can set the type of bullet for an item in the middle of the list by setting the type attribute of an li tag.

Example:

```
<html>
 <body>
  <ul>
   <li>An item</li>
   <li>Another item</li>
  </ul>
 </body>
</html>
```

Ordered Lists

This list is used to create and indexed list, such as a numbered or alphabetical list.
Define Ordered List - `<ol> ... </ol>`

Use the `<ol>` tags to set the start and end of the list. A number of list items will then go between the ordered list tags.

Ordered List Item - `<li> an item </li>`

Each item must use the `<li>` tags the same as with an unordered list. But this time the browser will number each item automatically, instead of showing bullets.

List Type `<ol type="A | a | I | i | 1">`

Set the type of list index by using the `type="?"` attribute. The default style is numeric, and you can also choose from upper or lowercase, alphabetic or roman numerals.

List Starting Position `<ol start="?">`

Set the starting number (or letter) if you dont want the list to start at `1` or `A`.

Item Value `<li value="?">`

You can set the value of an item in the middle of the list manually, if you do not want it to follow the previous letter or number. Simply set the `value` attribute of the item you wish to change. Note: subsequent items will follow the new value.

Example:

```
<html>
 <body>
  <h3> Ordered List </h3>
  <ol>
   <li>Item one</li>
   <li>Item two</li>
  </ol>
  <h3> Modified Ordered List </h3>
```

```
  <ol type="A" start="6">
   <li>List item 1</li>
   <li>List item 2</li>
   <li value="12">List item 3</li>
   <li>List item 4</li>
  </ol>
 </body>
</html>
```

Definition Lists

This type of list is used to define and describe terms, much like a dictionary. Typically an entry in the list consists of a term, and a definition of that term. A browser will usually bold the term, and indent the definition.

Define a Definition List - `<dl> </dl>`

Set the start and end of a definition list. All entries go within the `dl` tags. Each entry will usually consist of one `dt` and one `dd` element.

Definition Title - `<dt> </dt>`

The title of a term being defined. Note: you may have a term with no definition, or multiple terms with the same definition.

Definition Description - `<dd> </dd>`

The definition of a term. Note: you can have multiple definitions for a single term.

Example:

```
<html>
 <body>
  <dl>
   <dt>Term 1</dt>
   <dd>Definition of term 1</dd>
   <dt>Term 2</dt>
   <dd>Definition of term 2</dd>
  </dl>
 </body>
</html>
```

8. HTML Tables

Table tags are used for displaying spreadhseet-like data neatly formatted in rows and columns. They can also be used to design page layouts by placing content into invisible rows and columns of a 'table'.

Table - `<table> ... </table>`

Used to define a table, it contains all row and column tags along with their content. Think of it like the `body` tag, although there must always be at least one row in a table. It has some attributes to define the table layout.

`border="?"` - The size of the border (in pixels) surrounding the table

`cellspacing="?"` - The space (in pixels) between each cell, eg. between rows or columns

`cellpadding="?"` - The space, or margin, between the content of a cell and its border

Table Row - `<tr> </tr>`

To start a table row, the `tr` tags must appear within the `table` tags.

Table Cell - `<td> </td>`

A table cell is where the content goes. Cells must exist within rows, where the number of cells in a row determines the number of columns in the table. Cell properties can be set using the attributes:

`align="?"` - Alignment of text in the cell: `left`, `center` or `right` (*)

`valign="?"` - Vertical alignment of the cell: `top`, `middle` or `bottom`.

`width="?"` - Specify a fixed with of a cell, by default they will only take up as much space as they need.

`colspan="?"` - Column spanning allows a cell to take up more than one column, in order to match layouts of other rows. Replace `?` with the number of columns to span.

`rowspan="?"` - Row spanning, similar to column spanning, forces a cell to occupy more than one row.

`nowrap` - No text in the cell will be wrapped onto the next line. Similar to the `nobr` tag for paragraphs

Header Cell - `<th> </th>`

Similar to a table cell, a header cell must appear within a table row. Normally found in the first row, header cells are usually shown in bold and centered by the browser.

Example:

A simple table with three rows and two columns.

```
<html>
 <body>
  <table border="1">
   <tr>
    <th>Header 1</th>
    <th>Header 2</th>
   </tr>
   <tr>
    <td>Cell A1</td>
    <td>Cell B1</td>
   </tr>
   <tr>
    <td>Cell A2</td>
    <td>Cell B2</td>
   </tr>
  </table>
 </body>
</html>
```

9. HTML Frames

Frames allow you to have multiple sections of the browser window, called frames, each showing their own .html file within the frame. This used to be common practice when trying to show separate sections of a site in separate sections of the browser window, such as a header at the top, navigation at the side, and the rest was page content that someone could scroll down without making the header and navigation disappear.

Frame sets are rarely used these days, as the introduction of server side scriping languages such as php and asp allow you to create content pages dynamically. The introduction of HTML5 has also provided new methods of doing page layouts without having to use frames.

Frame Set - `<frameset> ... </frameset>`

Used instead of the `body` tag, the `frameset` tag defines a group of frames. Setting the `rows` and `cols` attribute allow you to create the number of frames needed for your layout.

`rows="??,??"` - To set up multiple frames in rows, replace the question marks by the size of each row, either in pixels or as a percentage. A `*` can be used as a wild card, for instance: `rows="100,*"` would give you a top frame of 100 pixels high, and a bottom frame using the rest of the screen.

`cols="??,??"` - Similar to rows, a number of frames can be set out in columns.

`border="?"` - Frame border thickness in pixels.

`bordercolor="?"` - Colour of border between frames. (*)

Frame - `<frame>`

Each frame within a set will need a `frame` tag to tell it which web page to load in the frame. It uses the attribute:

`src="url"` - Filename or URL of page to show in the frame

`noresize="noresize"` - The frame will not be able to be resized by a visitor

`scrolling="auto"` - Each frame will have vertical and horizontal scroll bars appear automatically when needed. You can change this by setting the `scrolling` attribute to `yes`, `no`, or `auto`.

`frameborder="auto"` - Individual Frame Border. Set to `0, 1` to specify whether or not that frame must have a border.

Unframed Content - `<noframes> ... </noframes>`

Very old browsers are unable to display frames, and in this case we need to specify what these browsers should display instead of the frames. Even though this is not much of a problem anymore, it is still suggested that you specify unframed content when using frames. Anything between the `noframes` tags will not be shown in modern browsers that show framed content.

Example:

A frame example with one header row, a left & right side column, and the content in the center.

```
<html>
 <head>
  <title>Example - Frames</title>
 </head>
 <frameset rows="100,*">
   <frame name="top" src="frames_top.html">
   <frameset cols="50,*,50">
     <frame name="left" src="frames_left.html">
     <frame name="mid" src="frames_middle.html">
     <frame name="right" src="frames_right.html">
   </frameset>
   <noframes>
     <i>error to display to those who cannot see frames</i>
   </noframes>
 </frameset>
</html>
```

TOPIC 4: WEB AUTHORING AND DESIGN TOOLS

T4.1) Description of web design tool

Web design is the process of creating websites. It encompasses several different aspects, including webpage layout, content production, and graphic design. While the terms web design and web development are often used interchangeably, web design is technically a subset of the broader category of web development.

Websites are created using a markup language called HTML. Web designers build webpages using HTML tags that define the content and metadata of each page. The layout and appearance of the elements within a webpage are typically defined using CSS, or cascading style sheets. Therefore, most websites include a combination of HTML and CSS that defines how each page will appear in a browser.

Tools of web-design

Some web designers prefer to hand code pages (typing HTML and CSS from scratch), while others use a "WYSIWYG" editor like Adobe Dreamweaver. This type of editor provides a visual interface for designing the webpage layout and the software automatically generates the corresponding HTML and CSS code.

Another popular way to design websites is with a content management system like WordPress or Joomla. These services provide different website templates that can be used as a starting point for a new website. Webmasters can then add content and customize the layout using a web-based interface.

While HTML and CSS are used to design the look and feel of a website, images must be created separately. Therefore, graphic design may overlap with web design, since graphic designers often create images for use on the Web. Some graphics programs like Adobe Photoshop even include a "Save for Web…" option that provides an easy way to export images in a format optimized for web publishing.

T4.2) Features of web design tools

Basic Features of Web Design Tools

A perfect web design tool should incorporate most and preferably all the features listed below:

- **Freedom to create.** I need a tool where I don't feel crippled, where I can realize my own vision. Templates are great, but I also need to tweak and change things the way I want.
- **Responsive design for any screen.** Phones, tablets, watches, 4K monitors - screens are becoming more fragmented every day. Many tools seem to ignore this completely, others do it very hard to use.

- **SEO support.** Very important, and needs to be updated as best practices change. E.g. keywords are no longer relevant to Google.
- **Rich content.** Video, animations, interactive widgets. The web is an interactive medium and should be treated as such.

Details of Features of Web Design Tools

Feature #1: Views

Most web authoring software provides multiple views of the web page you're working on.

- **Standard, normal, or design view** - This is typically the default view, which is a blank screen on which you type, paste, or insert content. This is very similar to a word processor screen.
- **Code view -** Allows you to view and work directly with the HTML code.
- **Split -** Both of the above views are displayed simultenously in separate windows.

Examples from common applications

- In Macromedia Dreamweaver, you can switch between views using the View menu.
- In Microsoft FrontPage, you can switch between views using tabs that appear in the lower left corner of the application window.
- In Netscape Composer, you can switch between views using either of the above methods.

Activity

- Find how to switch between views in your web authoring software. Does the software provide more than one way to do this? Try typing something on the screen in Normal or Design View, then switch to Code View to see the HTML that was generated by the web authoring tool.
- Ask your instructor for instructions on how to open and save files with your web authoring software in your school's computing environment.
- Now open your portfolio file *unit6.htm* in your web authoring software. At this point the page should have a banner, a navigation menu, and a pair of W3C icons. Practice switching between views and exploring your page using your web authoring tool.

Feature #2: Creating Headings and SubHeadings

In Normal or Design View, Web authoring software is very similar to word processing software. One or more toolbars appear across the top of the application window. A text formatting toolbar typically includes buttons for bolding and italicizing text, and probably additionally includes some means of identifying a heading or subheading.

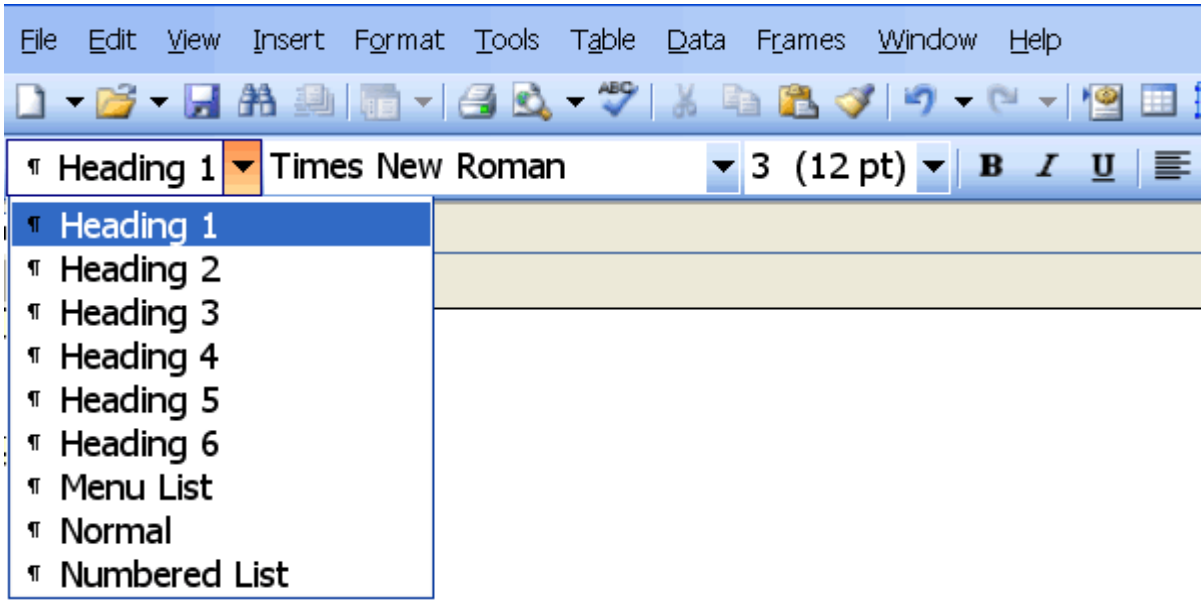
Examples from common applications

Example #1

The following is a sample toolbar from Macromedia Dreamweaver. To create an <h1> heading in Dreamweaver, simply select the button labeled **h1**, type your heading text, then press Enter.

Example #2

The following is a sample toolbar from Microsoft FrontPage. To create an **<h1>** heading in FrontPage, use the dropdown menu to select Heading 1, type your heading text, then press Enter.



Activity

- Add a main heading "Unit 6: Web Authoring Software" to your web page, just beneath the menu. Since you already know how to do this with source code, try doing it using the graphic view, then check your source code to see what code was inserted by the software.
- Now add a subheading "My Web Authoring Software" beneath this.
- As in previous lessons, add a named anchor to this heading with name="software". In some web authoring tools, this feature may be called a bookmark. See if you can figure out how to add it using your software's menu system or graphic interface.
- Beneath this subheading, write a short paragraph identifying what web authoring software you're using, including the version number.
- After you've added this content to your web page, switch to Code View to see the HTML that was generated by the web authoring tool. Does it differ at all from the HTML you would have used if you were coding this page manually?

Feature #3: Inserting Links

In many web authoring software products, you add a link to a document by selecting *Insert* from the menu, then *Link* or *Hyperlink*. A dialog box will appear, prompting you for the link text that you want to display, the destination of the link, and possibly other information. Most web authoring software tools additionally provide a button or icon that allows you to quickly insert a link.

Activity

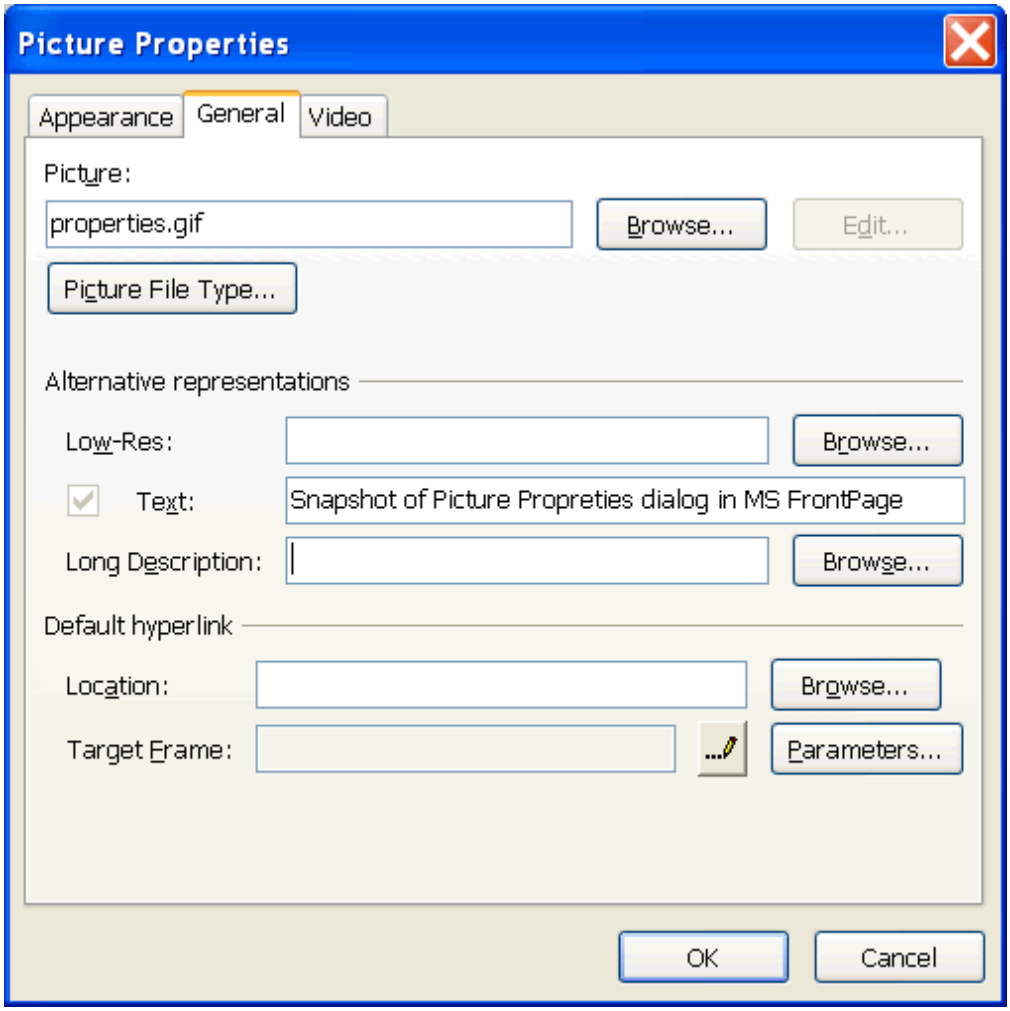
- Explore your software, and find out how many ways there are to add a link to your web page.
- Create a new section in your web page with subheading "List of Links", and a named anchor "links"
- In this new section, use your software's graphic interface to add an unordered list of links to a few of your favorite websites.
- Now switch to Code View to see the HTML that was generated by the web authoring tool.
- Open your portfolio home page (*index.htm*) in your web authoring software. See if you can figure out how to link to a specific named anchor within a document. In the appropriate places on your home page, convert items to links to the new sections in your Unit 6 document.

Feature #4: Inserting Images

In many web authoring software products, you add an image to a document by selecting *Insert* from the menu, then *Image* or *Picture*. A dialog box will appear, prompting you for the location of the image. After you have inserted the image into your webpage, you can edit its attributes in a Properties dialog box or panel. For example, you can change the image's height and width, put a border around it, make it into a link, and add alternate text for users who can't see the image. Image Properties are typically accessed by right clicking on the image and selecting Properties from the popup menu (if you are a Mac user, you can achieve the same result by pressing CTRL and clicking on the image to see the popup menu).

Examples from common applications

The following is the Picture Properties dialog from Microsoft FrontPage:



Activity

- Explore your software, and find out how to insert a picture.
- Create a new section in your web page with heading "Inserting an Image" and named anchor "image".
- In the new section, insert the 3D Box image that you created in the lesson on Basic Image Manipulation.
- Explore the various options that are available for image properties. Try chang some of these properties to see what happens. Be sure to add alternate text to the image for the benefit of those who can't see it.
- After changing image properties, switch to Code View to see the HTML that was generated by your changes.

T4.3) Web authoring protocols

The Internet relies on a number of *protocols* in order to function properly. A protocol is simply a standard for enabling the connection, communication, and data transfer between two places on a network. Here are some of the key protocols that are used for transferring data across the Internet from website to users.

HTTP
HTTP stands for Hypertext Transfer Protocol. It is the standard protocol for transferring web pages (and their content) across the Internet.

When you browse a web page, the URL might be preceded by `http://`. This is telling the web browser to use HTTP to transfer the data. Most browsers will default to HTTP if you don't specify it. You can test this by typing in say... `www.quackit.com` (instead of `http://www.quackit.com`).

HTTPS
HTTPS stands for Hypertext Transfer Protocol over Secure Socket Layer. Think of it as a secure version of HTTP. HTTPS is used primarily on web pages that ask you to provide personal or sensitive information (such as a password or your credit card details).

When you browse a web page using HTTPS, you are using SSL (Secure Sockets Layer). For a website to use HTTPS it needs to have an *SSL certificate* installed on the server. These are usually issued by a trusted 3rd party, referred to as a Certificate Authority (CA).

When you browse a web page using HTTPS, you can check the details of the SSL certificate. For example, you could check the validity of it. You could also check that the website does actually belong to the organization you think it does. You can usually do this by double clicking on the browser's padlock icon. The padlock icon only appears when you view a secure site.

Here's an example of Firefox's padlock icon from Firefox's address bar:

Firefox also displays a padlock at the bottom right of the browser window, along with the common name of the organization that the certificate was issued to:

FTP
FTP stands for File Transfer Protocol. It is used to transfer files across the Internet. FTP is commonly used by web developers to publish updates to a website (i.e. to upload a new version of the website).

Where HTTP is used for displaying the file in your browser, FTP is used simply to transfer the file from one computer to a specified location on another computer. You can use FTP to transfer the files from your computer to a remote computer (such as a web server), or to transfer from the remote computer to your local computer.

Internet Protocol Address (IP Address):
A numeric code that uniquely identifies a particular computer on the Internet. 127.0.0.1 is an example of an IP address. The browser uses IP addresses to find websites on the Internet.

Internet Service Provider (ISP):
Is a business or organization that offers users access to the Internet and related services , examples being BT, Virgin Media, Talk Talk.

Domain Name:
The name that identifies a website. For example, Facebook.com.

Domain Name System (DNS):
A naming system that converts domain names into IP addresses. We use domain names to navigate to websites because they're easier to remember, but computers require IP addresses to communicate with each other.

Doctype:
The doctype declaration specifies which version of HTML is used in a document. It has a direct effect on whether your HTML will validate , an example of this would be:

<!DOCTYPE html PUBLIC

T4.4) Characteristics of a good web design

Despite the innumerable variables, there are specific and telling features/characteristics of impeccably built websites, no matter the kind of business or industry, which any company can use a barometer to measure against. Below are some of the most important features/ characteristics of well-built web design of which to take note. These features are absolutely necessary for every website in order to deliver the exceptional results that site visitors and owners expect.

1. Quality Web Content. There's one primary reason people use search engines and browse websites, and that is to search for information. People desire information every day, and want if delivered in a fast and reliable fashion. Whether to entertain, entice or educate, superior content is a must in every frequently visited website, especially if search engine optimization is part of the website's overarching marketing strategy.

For business websites, content should include important information and come in the forms that are pertinent to the business. Retail sites for example, need high quality pictures of their products, while consulting firms are more apt to highlight client testimonials. A best practice for most search engine optimization gurus is also ensuring the most relevant content is prominent on the webpages.

2. Clear, User-friendly Navigation. A stellar web design must contain a user-friendly navigation scheme that allows visitors to quickly find the information needed. Important links must be easy to find and given logical, simple, and include easy-to-understand labels. Calls to action are placed in conspicuous spots within the navigation's scheme. If there is a plethora of content, then a search box is suggested to make it faster to reach more specific pages within a website.

3. Simple and Professional Web Design. Bells and whistles may seem nice in concept, but they rarely add much value to an effectively constructed and sensible web design. Typically, the websites best at effectively converting site visitors into buying customers, maintain an attractive layout, but keep it clean and simple. Google is an excellent example of such a site. Actually, users found Google's initial design over simplified during the initial testing phases. Thus in order to keep a simple interface, but prevent the appearance of site constructed without much thought, Google added the 'I'm Feeling Lucky' button underneath the search box. Despite the fact that hardly anyone clicks on this button, its addition balances the layout in such a way that delivers a better user experience.

To keep websites simple, without making them look bland such as in Google's case, a balanced distribution of content and graphics is required and the use of slightly contrasting colours and clear fonts is key. Colours that are scream, are overtly contrasting, and font sizes that are difficult to read will put a strain on visitors' eyes. Also, one should break up sizeable blocks of text with either spacing or images as appropriate.

4. Webpage Speed. People inherently lose patience quickly, and that holds true when visiting a website. The longer a website takes to load, the more likely a person will leave before it fully renders. Beautiful graphics and substantial content become useless if a site's speed hampers its ability to deliver content quickly.

Several factors affect site speed, including server speed, the number of graphics, website traffic, etc. A web design company must make sure to minimize all controllable factors slowing down site speed by using reliable site hosting, proper website code, and optimized graphics.

5. Search Engine Optimization. A well-designed website generally will receive many visitors, and one method to attract visitors is search engine optimization. This entails the insertion of search keywords in website content, an appropriate link profile, social media signals, and over 200 other factors.

6. Web Compatibility. The variety of browser and platforms which one can now be view web design can present a challenge to developers, but talented ones are accustomed to handling such factors. A site should easily render on various resolutions, screen sizes, and browsers; and with the increasing popularity of mobile devices, websites should function properly on the plethora of these types of devices.

When it comes to web design, remember that not all that glitters is gold. Know what truly makes a well-built site and you'll soon find your website quickly on its way to attracting and retaining customers

T4.5) Description of authoring tools

Web development can range from developing the simplest static single page of plain text to the most complex web-based internet applications,

A *platform* is the underlying programming language that the site will utilize. The most popular development platforms are HTML, PHP, .NET, and JSP.

Dream weaver is one of the applications that help manage webpages developed in the above platforms… and so are synonymously regarded as web platforms too though are only web development tools

Introduction to Adobe Dreamweaver

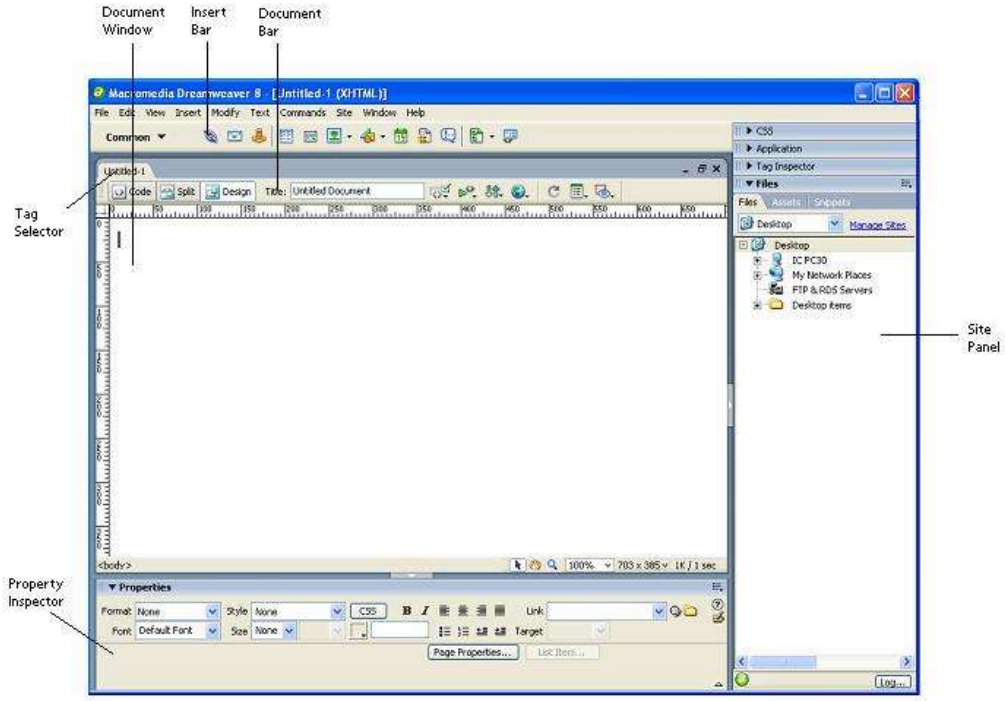
What is Dreamweaver? Dreamweaver helps you to create web pages while it codes html (and more) for you. It is located on the bottom tray  or in the **start** menu, under **Macromedia**.

Manage a website in Dreamweaver

Before you begin: *Webspace*: your webpage must be on your webspace to be accessible from the internet. store all components of the website in one folder. It is recommended that you create a separate "images" folder within the main one to keep track of your images. The main folder must be on your webspace. *Planning*: it helps if you know how you want your webpage to look *before* using Dreamweaver. Think about colors, uniformity among pages and organization of links and topics.

Creating a new page:

Under “File”, select “New”. Make sure it is set on “Basic Page” and “HTML”.



The Dreamweaver Windows

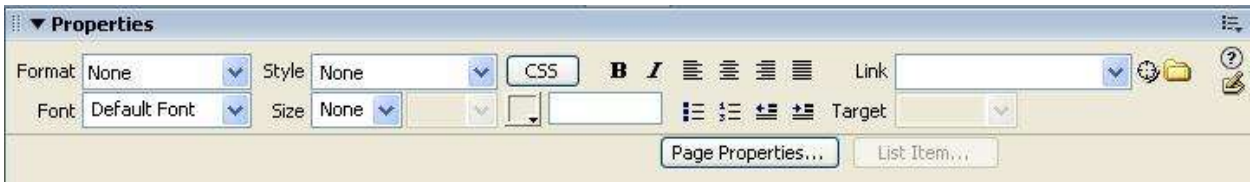
The **Document window** displays the current, editable page.

The **Site Panel** allows you to easily access, view and manage the files and folders that make up your site. This is **optional**, but useful.

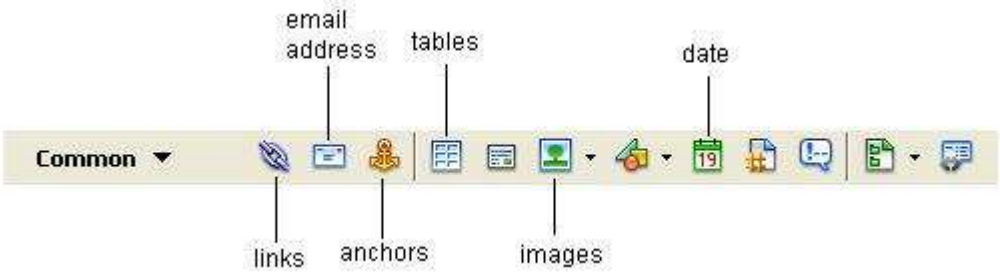
From the top menu, select **Site**⇨**New Site**⇨click the “**Advanced**” tab. Fill in the **Site Name** (“bob’s site”), the **Local Root Folder** (the folder where all pages of your website are contained), and the **http address** (the exact online location of your site—http://www.mtholyoke.edu/~bob/main_folder_name).

The two commonly used tool bars are the **Insert bar** and the **Properties inspector**. If they are not visible on your screen, pull down these options under **Windows (Insert** and **Properties)** at the top, or press **F4** on your keyboard.

The Property Inspector allows you to view and change properties of selected objects or texts.

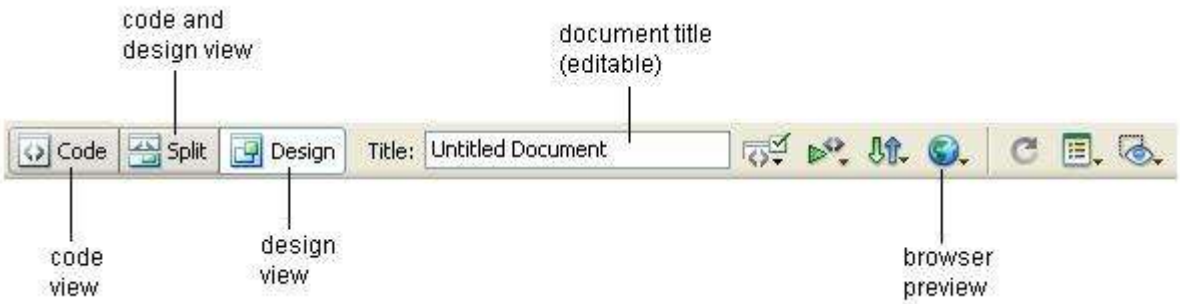


The **Insert bar** contains buttons for inserting various types of objects, e.g. images, tables and text into your page.



The **Document toolbar** contains buttons and pop -up menus that provide different views of the Document window (e.g. Design view, Code view) and gives you access to references and a preview of your page in the browser of your choice. If the document toolbar is not already visible, go to **View➪ Toolbars➪ Document** in the top menu.

To preview your page in a browser click on the Browser Preview on the Document toolbar or choose **File ➪ Preview in Browser**. You must save the page first to see changes.

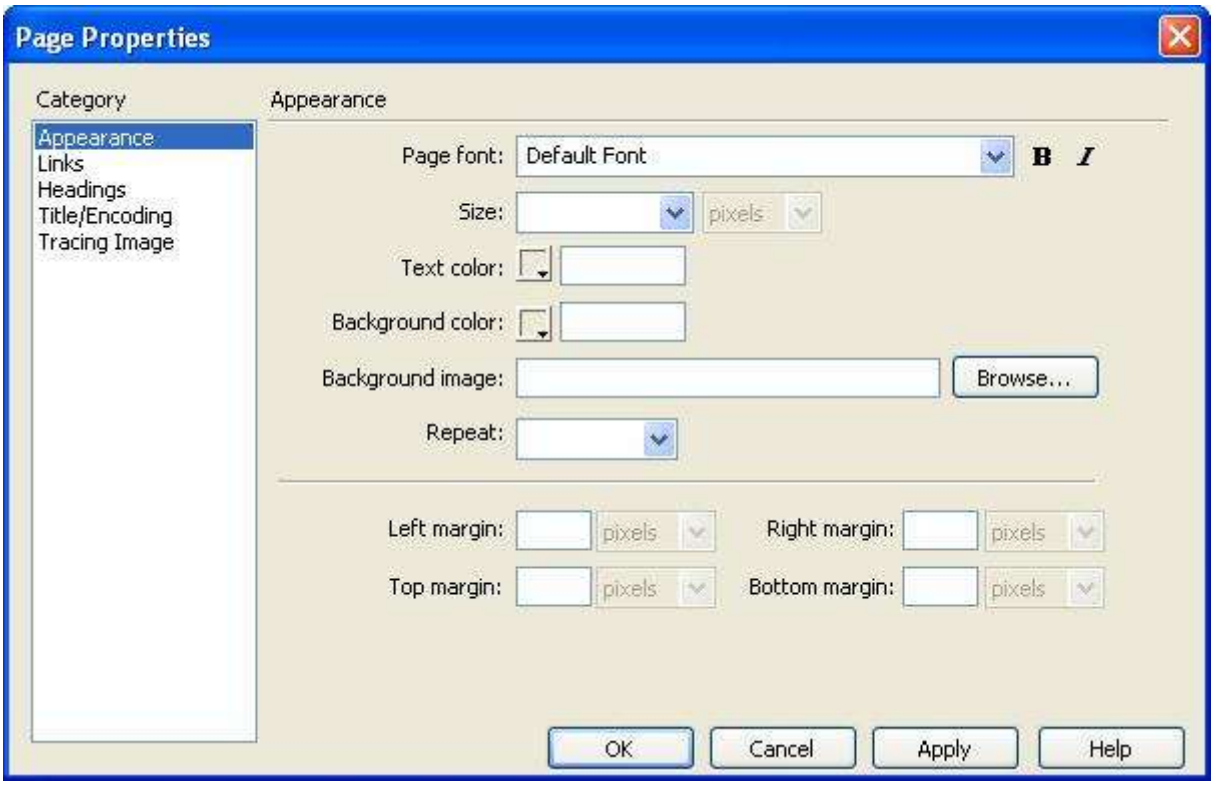


Page Title and Page Properties

Page Title: It is important to title new web pages. A good habit to adopt is to title your page before you do anything else. The page title appears in the top title bar of the browser window and if bookmarked, the title of the bookmark. Without a title, "Untitled document" will appear. You can yype in the page title in the Title box located in the **Document toolbar**.

Page Properties

The default appearance of text, background color, page margins, color of links, and other properties can be changed in the **Page Properties** window. Go to **Modify** ⇨ **Page Properties** or press Control + J on your keyboard to open the Page Properties dialog box. It is important to establish defaults on text as different browsers may have different defaults.



Creating Basic Web Page Elements

Dreamweaver's basic authoring functions are quite straightforward though manipulating the properties of basic elements, such as tables and images, may be a little less obvious.

In this section, to cover the basic functions, we will create a simple personal home page with the following elements:

- Text
- Table
- Image
- Hyperlink
- Mailto: link
- Background color

Note: SAVE YOUR PAGE NOW. Saving a page before inserting elements helps avoid prompts for defining full web sites and will help keep the code references calling for links and images relative to the document.

Backgrounds:

You can set your background color here. It is important to consider readability—good contrast between the text and background color. Neons are bad!

If you have an image to use as background on your page, select 'Browse' and click on the image you want to use as your background. The image will tile (repeat the image) over the entire page.

Working with Text

Text - Inserting

As with most web authoring programs, the essential method of inserting text with Dreamweaver is this

To insert text:

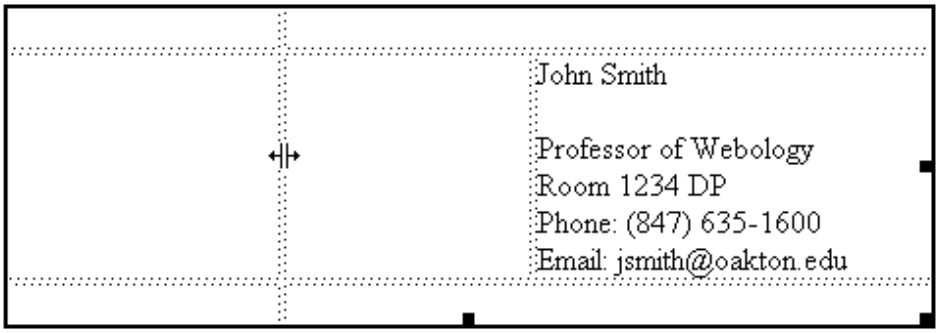
1. Click somewhere on the page.
2. Type.
3. When finished, stop typing.

You can also copy and paste from most any other text-based document but the result is a pretty boring chunk-o-text. Text must be formatted to achieve a more interesting look and easy-to-read layout.

For the home page exercise and use the table we created to contain the 'header' information. Type the following text (substitute your personal information if desired) in the **far right cell** of the 2nd row:

> ***Note on Line Breaks:*** *Pressing Enter will give you a double-space between lines. Shift-enter will single-space.*
>
> John Smith, PhD.
>
> Professor of Webology
>
> Room 1234
>
> Phone x5678
>
> Email: jsmith@oakton.edu

Your page should now look something like this (don't worry if the cell widths don't match... yet).



As it is with word processing, what you type will appear to be in Dreamweaver's default font-- generally Times Roman or Arial. This does not impact how the text will ultimately look in a web browser. If text is left unformatted, a web browser will apply its own default. (Even after formatting, it is still possible for a browser to override.)

Text - Formatting

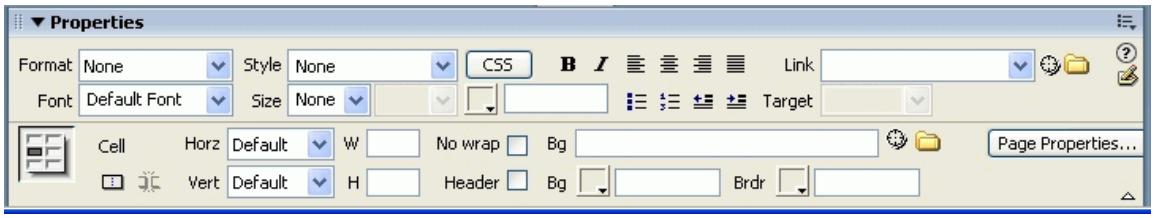
Text formatting (e.g., size, typeface, color, etc.) can be accomplished in a number of ways. More sophisticated web sites will use Style Sheets for much of this. Style Sheets are a set of commands that control the look and layout of web pages. Depending on how they are used, they can be applied to multiple pages or a portion of a single page. Though Style Sheets are beyond the scope of this introductory document, they are becoming standard and are worth learning as your skills and interest progress.

The most common way to format text in simple web pages is to apply formatting commands to individual words, phrases or lines of text. This is done by selecting the text and using the Properties palette to make changes.

To format text:

1. Drag the mouse across the desired text to select it.

2. Use the Properties Palette to change the text appearance. If you are formatting text in a table cell, the Properties palette will show text commands on the upper half of the palette and table/cell commands in the lower half.

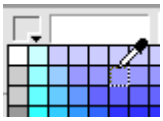


Key to formatting commands on the Properties palette:

- Paragraph/Heading. If you are formatting a line of text that is essentially a headline introducing a section of the document, choose one of the Heading sizes from the drop menu. The appearance will be similar to choosing a text size, but headings provide structural markup to the document-- which browsers for disabled users depend on.

- Font style. "Default" applies no style and leaves the appearance up to the browser when the page is displayed. Because the success of declaring a font style requires that other people's computers have that font installed, the five main choices on this list are pretty safe for most browsers. Each of the five selections has a 2nd and 3rd choice that will be applied if the preferred font is unavailable.

- Text color. Type a 6-digit hex code into the text field or click on the small down arrow near the upper left to reveal a color palette. The mouse pointer becomes an "eyedropper" that can choose a color from the color palette or can be pointed at an element (text, image, etc.) on the web page to duplicate its color.



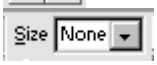
- Bold and Italics. Functions like most word processors. Choose one or both to apply that style to selected text.

- Alignment. Selected text can be left-justified, centered or right-justified.

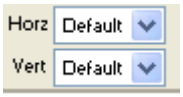
- Bullets and numbers. Select multiple lines of text then click one of these to make the list bulleted or numbered. To add to the list later, place the cursor at the end of a line and press the Enter key. Note that if you wish to format numbers and bullets, you may do so by going to the Text Menu and clicking on "List."

- Indent/Outdent. Selected text will be moved left or right.

- Text size. Drop menu of relative text sizes.



- Alignment within a cell. Click in the desired cell and use the Horizontal and Vertical Alignment drop menus on the Properties panel to set how everything in the cell aligns. Commonly, text will be aligned 'top' and 'left.'



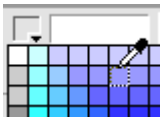
- (You can also drag the mouse from one corner cell to the opposite corner, selecting all cells, and THEN set the alignments for all the cells at the same time.

Continuing our little exercise...

Suggested formatting for the Home Page header text.

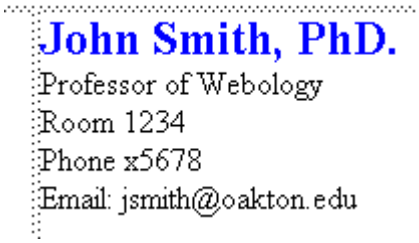
1. Select "John Smith, PhD." by dragging the mouse across the text.
2. Click on the Size drop menu on the Properties palette and choose "+2"

3. Click on the Color palette and use the eyedropper to choose a color



4. Click on the Bold button on the Properties palette

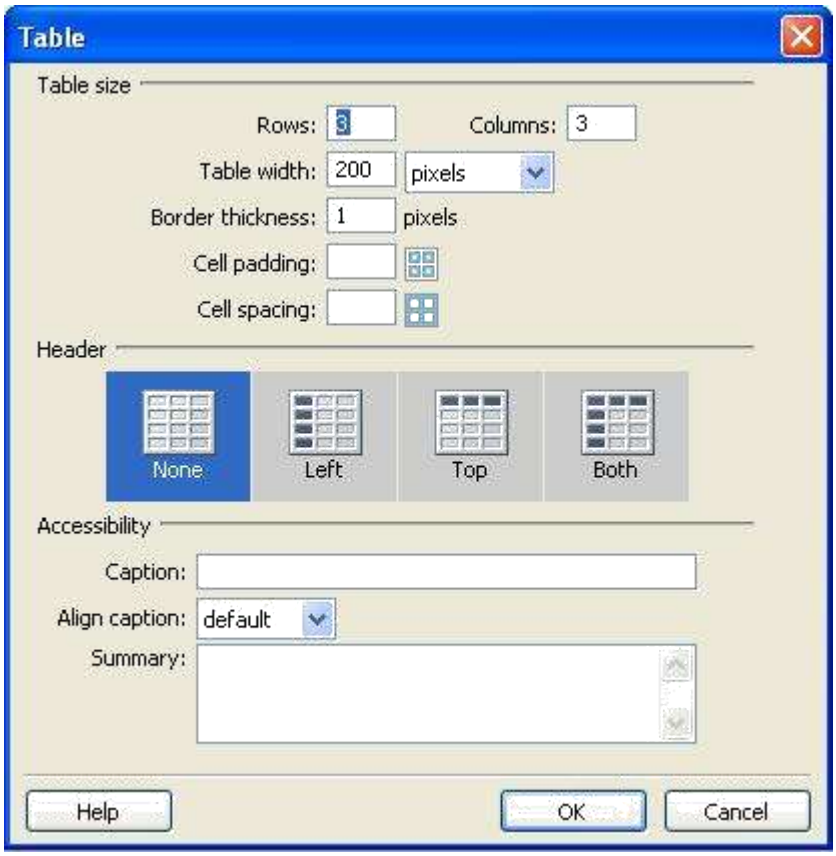
The selected text should now be a little larger, bolder, and have a different color than the rest of the text in the cell.



Tables

Tables are great for organizing and giving structure to the page.

1. Place cursor where you want the table.
2. Choose the Table icon on the Insert bar or at the top choose Insert➪Table.
3. Specify the number of columns and rows.



Cell Padding – specifies the amount of spacing between the content and the cell edge (wall) **Cell Spacing** – specifies the amount of spacing between each table cell, not including the border

Table Width – specifies the width of the table in pixels or as a percentage of the browser window. For precise layout of text and images, it is recommended that you specify tables in pixels.

> **Nested tables**: When creating nested tables, i.e. table(s) within table(s), designate specific pixels on the outside table. 700 is one standard. Another option is to the outside table at 100%.

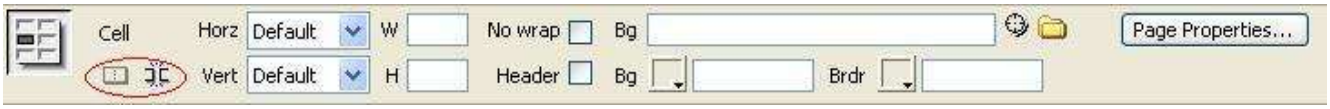
For tables inside main table percentages are generally used rather than pixels.

Border thickness: specifies the width of the table border. A border of "0" means the lines will appear invisible.

Background Color of Table, Cell or Border:

Select the entire table or a cell and click the Background color or Border button in the Property Inspector.

Split/Merge Cells: You can split (divide the selected cell) or merge (unify selected cells) by selecting the appropriate cell(s) and clicking either the split or merge icon in the properties inspector.



Inserting Images

It is recommended that your image be the size you want it to appear on your Web page, rather than enlarging or shrinking the image in your HTML editor. You should set the image resolution to 72 dpi (close to standard computer display resolution) to hold down image file size. Photoshop can help you do this (use the "Save Images for the Web" option).

To insert an image:

- Place the cursor where you want the image to appear on the page and then click the Image icon on the Insert bar, or go to **Insert** ⇗ **Image** on the top menu.
- In the dialog box that appears, click Browse to choose a file. It must be in your website folder.
- Click 'OK'

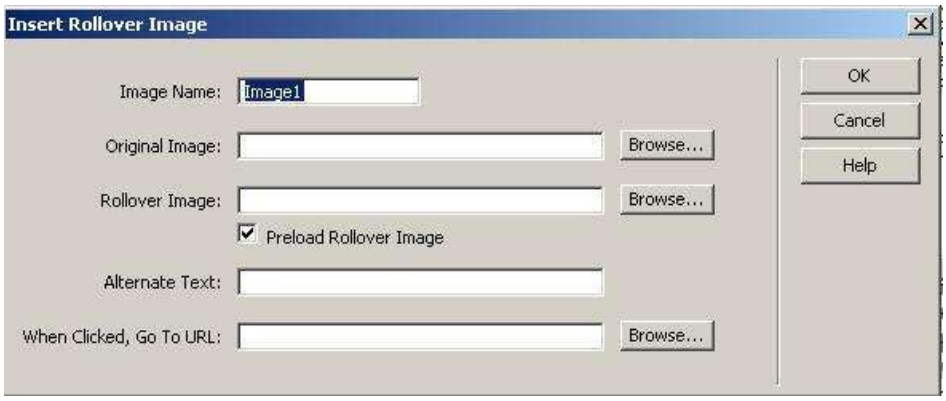
Inserting Rollovers page animation

A rollover is an image that changes when the mouse cursor moves across it. A rollover consists of two images: the primary image (the image displayed when the page first loads) and the rollover image (the image that appears when the cursor rolls over the primary image).

To create a rollover:

It is essential that both images are exactly the same size or the images will not appear correctly. You can do this in a drawing program.

- In Dreamweaver, place the cursor in the Document window where you want the rollover to appear.
- On the Insert bar, click the Rollover icon or go to **Insert** ⇗ **Image Objects** ⇗ **Rollover Image**.
- The Insert Rollover Image dialog box will appear. Browse to select an image file for the original and rollover image.



4. Make sure the Preload Rollover Image option is checked so that the rollover image loads when the page opens in a browser. This ensures a quick transition between the images.
5. To create link, in the 'When clicked go to URL' box, browse to select a file.

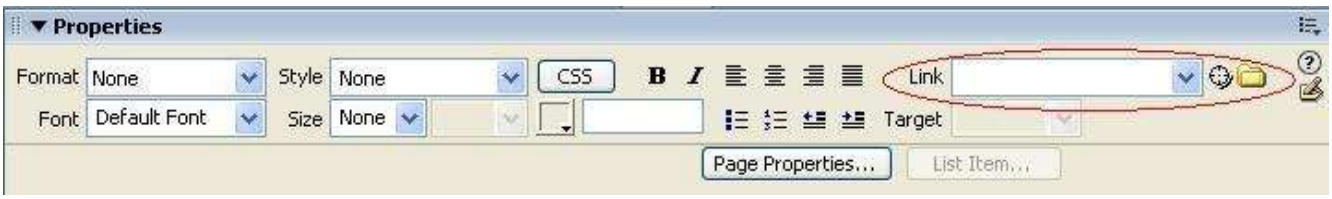
6. Click OK.

Creating Links and Email Links

Creating Links

Links are used to navigate between pages in your site and to introduce material from other web sites.

To create a link



- Highlight text or an image in the Document window.
- To link to a document inside your site, click the folder icon to the right of the Link field to browse and select a file. The path to the linked document appears in the URL field.
- To link to a document outside your site, enter an absolute path including the http:// (i.e., for a link to NASA, the absolute path is http://www.nasa.gov)

Creating Email Links

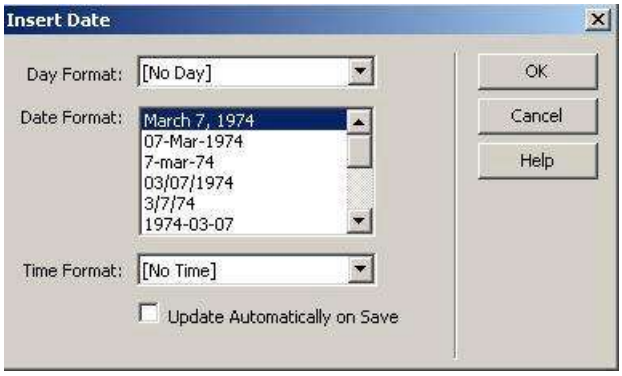
An email link opens a new message window (using the mail program associated with the user's browser) when clicked. In the email message window, the To: field is automatically filled with the address specified in the email link.

To create an email link:

- In the Document window, position the insertion point where you want the email link to appear, or select the text you want to appear as the email link.
- On the Insert bar, select the Email icon (looks like an addressed envelope). Or you can go to Insert ⇗ Email Link in the top menu.
- In the Text field of the Email Link dialog box, type or edit the text to appear in the document as an email link.
- In the Email field, type the email address that will appear in the To: field.
- Click OK.

Inserting a Last Updated Date

- Place cursor where you want the date inserted.
- From the Insert bar, select the Date icon.
- A dialogue box will appear, choose the day, date and time format you desire. Check the box for "update automatically on save" to have the date change whenever you save the web page file.



Inserting Anchors

Anchors mark specific positions in a page(s). Use named anchors to jump to a marked position in the current document or to link to a marked position in a different document.

To create a named anchor:

- In the Document window, place the cursor where you want the named anchor.
- Click on the Named Anchor icon on the Insert bar or go to Insert ⇗ Named Anchor from the top menu.
- Enter a name (no spaces) for the anchor in the dialog box that appears. (If the anchor marker doesn't appear at the cursor location, choose **View** ⇗ **Visual Aids**⇗ **Invisible Elements**.)

To link to a named anchor:

- Select text or an image in the Document window.
- In the Property inspector, enter a pound sign (#) and the name of the anchor in the Link box. For example to link to an anchor named "two" in the current file, type #two.

To link to an anchor named "two" in a different file in the same folder, type filename.html#two.

Saving

It is good practice to save web pages without capitals, spaces, or funny characters. This will ensure that the page will appear on all browsers, as well as make it easier to type in the address correctly.

The homepage of your website should be saved as "index". This means that instead of getting a file directory (where everyone can see ALL the files in your folder) your homepage automatically loads. In any folder, the file named "index" is the homepage.

You can tell if a page hasn't been saved or not if the tag selector has an asterisk. "index.html"*

A quick and dirty way to use a page as a template is to either create a copy with a different name or copy and paste the code.

NOTES:

Photoshop and Illustrator are great for creating banners and images. We have workshops for that!

Resources: Don't forget the Dreamweaver Help and tutorials. There are also numerous websites for html. Using a search engine to find a solution to a specific problem works pretty well. Consultants hold hours in the Info Commons. Come bother us.

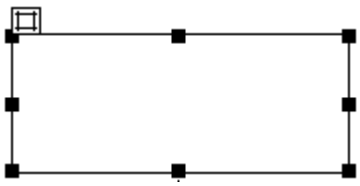
Working with Layers

We're going to a **look at** some of the characteristics of **layers**, and then work with them and apply some behaviors to them.

Introduction

Layers are squares that can be placed wherever you want in the Page, we can insert HTML content in them. Those layers can be hidden and overlapped between them. This gives you a great variety to design.

Layers can be moved from their positions by just clicking on the the white square in its lerft top corner, and whilst keeping the mouse pressed, dragging it to the new position.



They can also can be resized by clicking on the black squares and dragging them until the size you want is reached.

You can insert text, tables, images, and flash animations inside the layer's square, and also all the elements that an HTML file can contain.

This icon is used to select the layer when you click on it, and when you clear it, you are also clearing the layer.

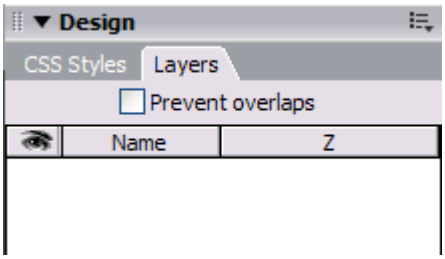
Inserting a layer

Layers can be inserted in a Page through the **Insert** menu, and then selecting the **Design object** option, and **Layer**.

Once the layer is inserted, you can edit its attributes by selecting it.

You can select layers through many different ways. One of them is clicking on the icon. This is not useful when there are many layers in the same file because all layers have an image like this one associated, and it's very easy to select the wrong one.

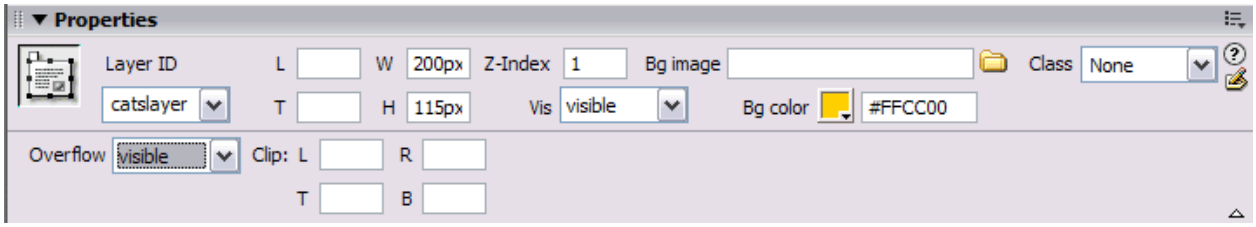
When there are many layers in the same file, it's recommendable to select them through the **Layers** tab in the Design panel, which can be opened through the **Window** menu, and then the Layers option. If the **Layers** option isn't in the menu, it may possibly be in the **Others** option. Also, it can be opened just by pressing **F2**.



In the panel are the names of all existing layers in the actual file, and you only need to click on the layer name to select it.

Layer format

Layer's properties are shown through its **properties inspector**



Layer ID is the layer's name. You can change it through the **Layers** panel, and double-clicking it.

L and **T** indicate the pixels distance between left and top limits of the document and the layer.

W and **H** indicate the layer's width and the height.

Z-Index is the depth order of the layer. This value can be changed through the **Layers** panel. A layer will be overlapped by others which greater Z-index.

Vis indicates the inital view of the layer. Visibility can accept four types.

Default (Browser view),

Inherit (the layer of the Page that is being displayed is shown)

Visible (It shows the layer although the Page isn't being seen)

Hidden (the layer is hidden).

You can also change the view through the **Layers** panel by clicking on the eye image. The open eye indicates **Visible**, and the closed eye indicates **Hidden**.

Through **Bg Image** and **Color** you can indicate an image or a background color for the layer.

Overflow controls how the layers appear in a browser when the content exceeds the specified size of the layer.

Visible indicates that the layer would be amplified to make its content fit.

Hidden specifies the additional content will not be shown in the browser.

Scroll specifies that the browser should add scrolling bars to the layer although they are or not needed.

Auto makes the browser show the scrolling bar when they are needed (when the layer content is bigger than its limit).

Using Frames

We are going to **see** what **Frames** are and when they must be used. We will also see how to insert a simple frame into a website and how to work with it.

Introduction

Frames are used to distribute the data in a web site. They help to keep some parts straight such as they are, as the logotype and the browser bar, while the others can change. In addition they usually improve the appearance.
Each page's frame has its own HTML file. For example, in the image on the right you can see a page with two frames. The left frame has the **menu.htm** file, and the right frame has the **dogs.htm** file.
To see the page this way, we have opened the browser's file **frames.htm**, which is the page that has grouped the frames.

It is possible to edit the frame's files from the page that containts the set of frames. This simplifies the work as it is easier to figure what the final page is going to look like.

This is something that you cannot do if you edit each framed file individualy.

Working with frames can be a bit complicated, and even more so in the begining. We are not going to go too much into the theme, and we will only look at some of the basic concepts with a few easy examples.

Creating frames

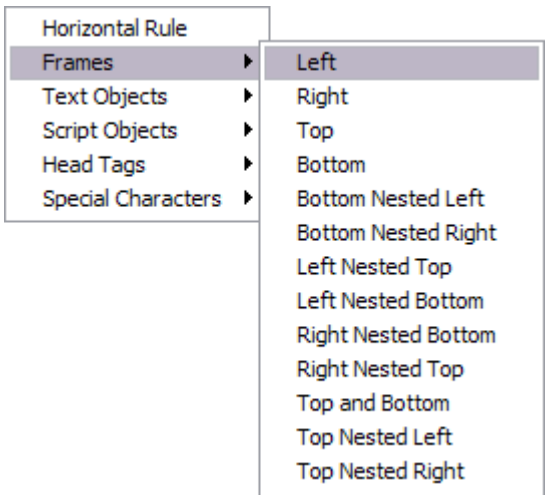
There are many ways of creating a frame. We will only learn one.

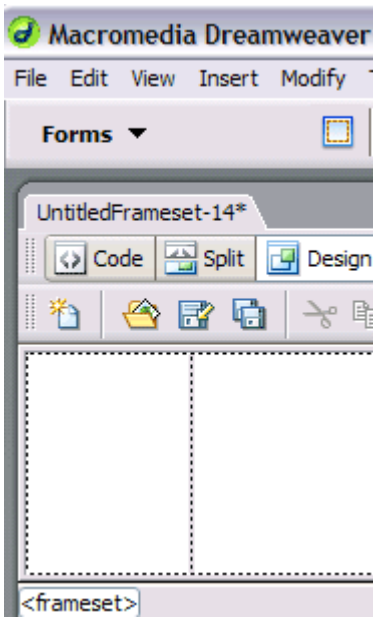
To create a frame, you first need to open a file. It can be new or an existing one.

After this, you need to go to the **Insert** menu**, HTML, and Frames**. Through this option you can select the type of frame you want to create.

We are going to look at the **Left** frame option.

If we click on **Left**, a new frame will be created in the left side of the current file.





As you can see in the image, there is a line splitting the document.

In this case we will have three files: the left one, the right one, and the one which has the set of frames. The right one is the file we had at the start. It is in the frame known as **Main Frame.**

To select the file that has the set of frames you have to click on the line that splits the frames. This is only possible in case of not having been previously saved.

In case of inserting a right frame instead of a left frame, the empty frame will be shown to the right of the original file.

In this image you can see an example of a right frame.

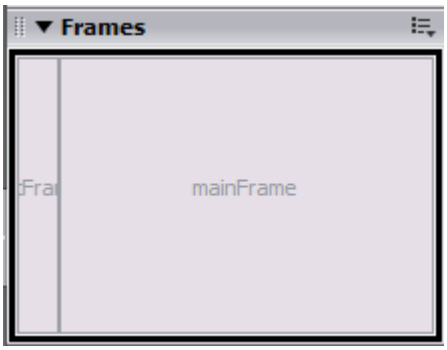
A right frame has been inserted over an existing file, **menu.htm**.

As in the previous case, we have three files: the left, the right, and the one which has the set of frames. In this case the file we had at the begining is the left one, while the previous was the right one. For this reason **Main frame** will be at the left.

Main frame is the frame that is always in the initial file, into which the rest of the frames have been inserted.

Selecting frames

If you want to select the frames of a document you have to go to the **Frames** panel which can be opened through the **Window** menu. If the **Frame** option isn't in this menu, it may be in **Others** option. You can also open the **Frames** panel by pressing the combination of **Shift+F2** keys.



The frames that are in the frame document are found in the **Frames** panel, and we can go through them by clicking on them in the panel. You can also select the frame Page by clicking on the border that sorrounds the frames.

Select the frames to edit the document they may contain.

Select the frames to specify the specific properties of each one of them.

Saving Frames

All documents that have frames must have a Page inside each one of them. This is why when you create a frame, new pages are loaded by default in each one of them (apart from the frame that has the original page).

These new pages can be replaced later by other existing pages as we'll soon see.

If you save the page that has the set of frames, each one of the pages included in their frames we'll be saved too.

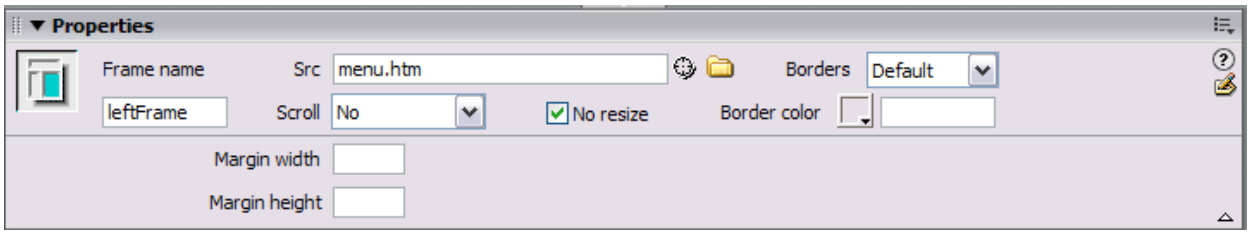
It is not recommendable to save the frames for the first time using the **Save all** option, because we can make mistakes when naming the new files.

It's preferible saving each file (one by one) unless all the frames had an already existing page, because in this case the unique file that will have to be named will be the one that has the set of frames.

To save a file that has a set of frames, you have to previously select it.

To save each one of the documents, you just have to place the cursor on one of them and click on the **Save** button.

Setting up Frames
After selecting a frame through the **Frames** panel, its properties can be established through the **properties inspector**.



Each frame has a name assigned that can be changed through the **Frame name** option. The name cannot have blank spaces.

Src is the HTML file name that is displayed in the frame.

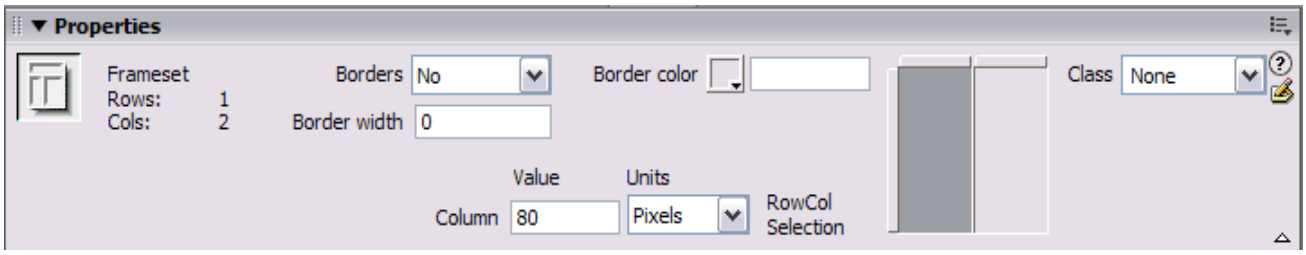
In **Borders** you can select a line that splits the frame from the rest of the frames. In case of showing the border, a color for this can be specified through **Border color**. You can also establish a width for the border through the **Width** option.

Scroll will indicate whether scrolling bars will be shown or not when the frame file cannot be completely seen.

If the **No resize** option is activated, it indicates that users won't be able to change the frame size in the browser.

The **Margin width** and **Margin height** set the separation between the content of the frame and its right-left and top-bottom borders.

If all the sets of frames are selected, then the **Properties inspector** will display the following options:



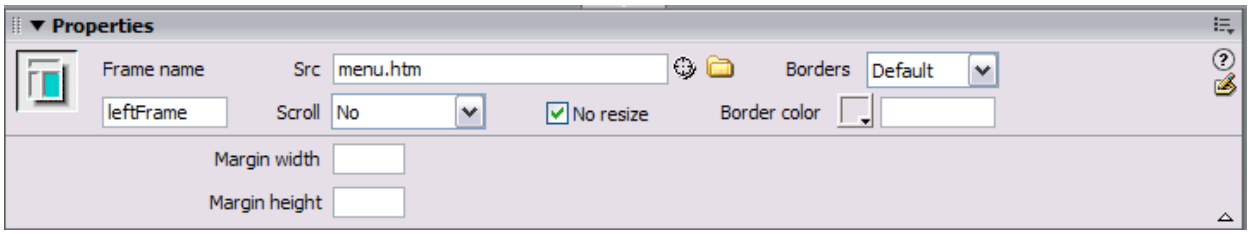
In **Borders** you can add a line to split the frame from the rest of the frames. In case of showing the border, a color for this can be specified through **Border color**. You can also establish a width for the border through the **Width** option.

The **Column** field (or **Row**) is used to set the frame size, and it can be introduced as **Pixels**, in **Percent** (of the window) or **Relative** (proportional to the rest of the frames).

You will usually use two frames, one of them with size in **Pixels**, which will contain the browser bar, and the other frame with a **Relative** size, to adjust it to the rest of the browser window.

Frame content

As you have seen, the frame content can be established through the **Src** field of the **properties inspector**.



When we work with frames, our aim is to load different files in at least one of them.

Through the **Src** field of the **properties inspector**, it's only possible to specify the file that will be loaded in the frame in first time, but we can change this later using links.

As you remember, in the hyperlinks unit we learnt the possible links destinies. Those destinies could be **_blank**, **_parent**, **_self**, and **_top**. Let's see again what are they used for. You now know how to work with frames and they will be easier to understand.

- **_blank:** Opens the file linked in a new browser window.

- **_parent:** Opens the linked file in the frame window that has the link or the set of main frames. As you know, the **main frame** is the frame where the initial file is and where the rest of the frames have been inserted.

- **_self:** Is the predetermined option. It opens the linked file in the same frame or window as the link.

Opening the linked file in the whole browser window means that the window frames will disappear when you open the link in it.

In addition to these destinies, you can also set the name of one of the frames in the Page as one of them, so the page will be loaded in that frame.

We can add all these destinies to any element of the Page that has any link, it can be a text, an image, an image map, a flash element, etc.

Thanks to this we can make our links work as we please, loading Pages in some of the frames, in a new window, in the whole window, etc.

This task can be difficult and complicated at the begining, but it leads to good final results.

teacherClick courses can be seen on the net using a top frame with pull-down panels

Using behaviors

In this unit we're going to **learn** the basic characteristics of the **behaviors**, and also look at a pair of examples of possible applications.

Introduction

behaviors are actions that happen when the userdoes something over an object, for example moving the mouse over an image, clicking on a text, double-clicking on an image map, etc.

behaviors don't exist as HTML code, they are programmed with JavaScript. Dreamweaver allows you to insert them through the **Behaviors** panel, so it is not necessary to write a JavaScript code line to program them.

The image below has a behavior associated. Place the mouse over it to see what happen.

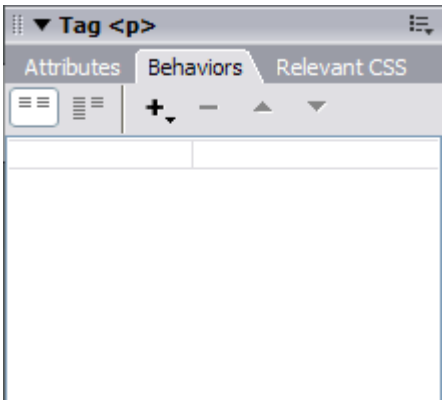
The image has two behaviors associated to show and hide the layer. We'll look at this kind of behavior later.

Other behaviors you have seen are the ones applied to sustitution bars and browser bars, they are predefined, and for this reason it isn't necessary to introduce the set of behaviors through **Behavior** panel.

To validate forms you saw some of the caracteristics of **Behavior** panel. Let's remember which we need to insert behaviors later.

The **Behavior** panel can be opened through the **Window** menu, and then the **Behaviors** option, also by pressing **Shift+F3.**

In this panel you need to click on the  button, and in **Show events** for selecting a version from the browser list.



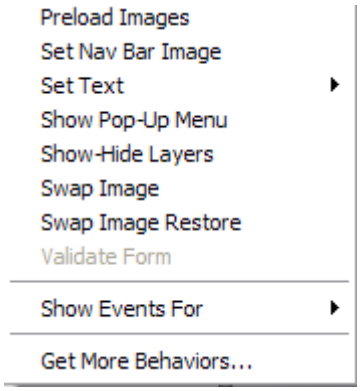
Some behaviors don't work with some browsers. Depending on the selected browser, you will see or not some of the possible behaviors.

There are many behaviors for Internet Explorer but they don't work with Netscape. As most users use Internet Explorer, let's select this browser. You can select from one of its latest versions: **IE 5.5** o **IE 6.0**.

Once the browser is selected it's not necessary to re-select it the following time you want to insert any behavior.

Inserting a behavior

When there is an established browser, you can insert behaviors.

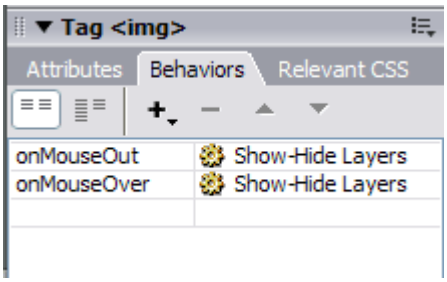


The first thing to do is to select the object on which the behavior is going to be applied, it can be an image, a text fragment, etc.

When you click on the button of the **Behavior** panel you will see the **Show Events for** option. Also, you will see a list of all possible actions in the browser previously selected, so you can select one.

Depending on the element on which you want to apply the behaviuor, you can select some actions, and others not.

In this case the **Validate form** action cannot be selected because there isn't any form in the Page.



After choosing any action, the corresponding behavior appears in the **Behaviors** panel. In this case, two behaviors have been inserted.

As you can see, each behavior has an action and an event associated to it.

Actions are the ones which have been selected in the previous list, and the event indicates the action by itself (when it is done).

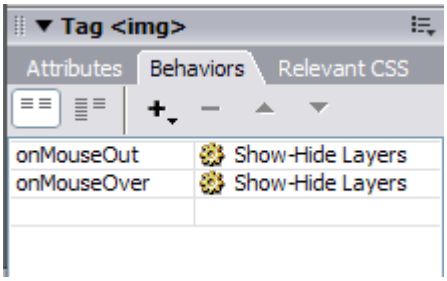
To clear any behavior, you need to select it in the **Behaviors** panel and click on the button. You can also change the order of the behaviors applied to an object, by selecting them and arranging their order through the buttons.

Show and Hide layer behavior

One of the most habitual and interesting behaviors is **Show and Hide layers**. It is obvious that when you want to apply this behavior there need to be layers present in the document.

In the previous Page you had an example of this type. Let's see which events and actions you need to establish in order to produce the behavior.

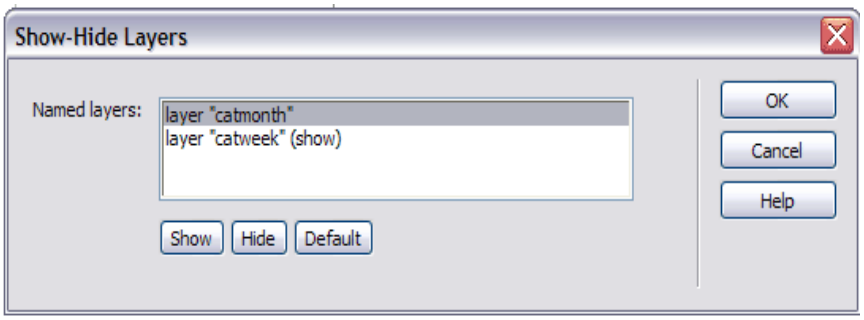
After selecting the image you need to select an action from the list by clicking on the + button. In this case the action has to be **Show-Hide layers**



After selecting the action, you must specify which layers have to be shown and which hidden for the event. For this you need to click twice on the action in the **Behavior** panel.

A new window is then shown, with the list of all the layers in the page, where you can indicate the view variation for each one of the layers.

You can indicate whether they are going to be Shown, Hidden, or if they are going to acquire the initial view (Predetermined). To determine the state of each one of the layers it's necessary to select them one by one, indicating the state through the buttons at the bottom of the window.



It's not necessary to apply a different view to all the layers of the page, only the ones you want to change at the moment of producing the event. For example, in this case the "**catmonth**" layer has not been applied a different view, because we don't want it to change when the event plays out.

If you want to clear an established view for any of the layers, you only need to click again on the button of the applied view. For example, to clear the view applied to the "**catweek**" layer, we would have to click again on the **Show** button.

You need to be careful about what you want - in this case it is that when you place the mouse on the image the layer is shown, and when the mouse is out of the image the layer

is hidden again. For this you need to insert two **Show-Hide layers** behaviors with the action. One of them will show the layer for the event **onMouseOver** (when the mouse is over), while the other will hide it for the event **onMouseOut** (when the mouse is out).

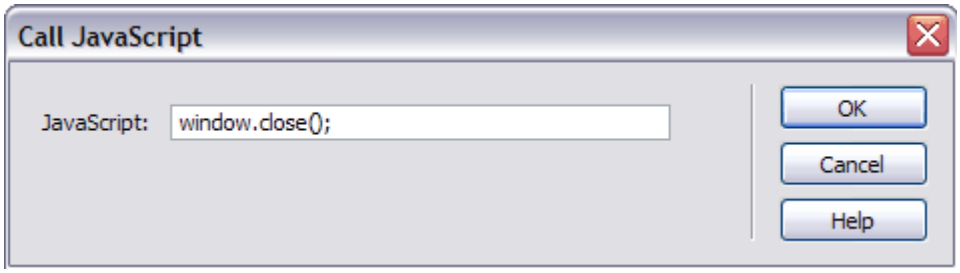
Using JavaScript Code

Another behavior that can be applied on any object is **Call JavaScript**. This behavior allows us to insert JavaScript code inside the file.

For example, it's possible to make the browser window close when you click on an object. To do this you need to insert a "**window.close();**" JavaScript line.

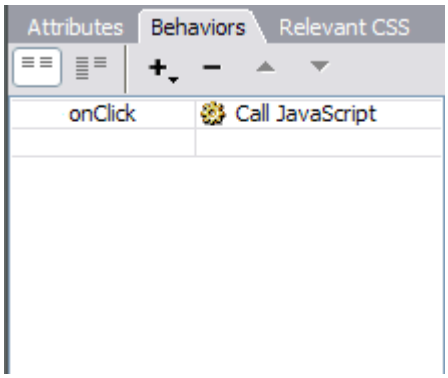
The same as the layers example, the first thing to do is to select the object on which the behavior is going to be applied. After this you need to select the **Call JavaScript** action through the  button.

Then a new window shows up and you now need to introduce the JavaScript line.



Once the behavior is inserted in the **Behavior** panel, you need to establish an **onclick** event to produce the call to JavaScript when clicking on the object.

By clicking twice on the action the previous window is opened again, and it's possible to modify the line code.



Introduction to Adobe Flash

Adobe Flash (formerly called Shockwave Flash, often just called Flash) is a multimedia software platform used for production of animations, rich Internet applications, desktop applications, mobile applications and mobile games. Flash displays text, vector graphics and raster graphics to provide animations, video games and applications. It allows streaming of audio and video, and can capture mouse, keyboard, microphone and camera input.

Introduction to Macromedia Flash
Macromedia Flash Mx/8 is a powerful Web authoring application used to create animation, interactive enviroments and data driven applications for use on Web sites. Flash Mx/8 is one of the leading Web authoring programs for creating vector-based animations or "Flash movies" for Web sites. A working Flash document should have the file extension of .fla. A compiled Flash document exported to web is abreviated as a .swf file.

Flash creates vector-based graphics rather than bitmap graphics. Vector graphics are created using lines and curves, whereas bitmap graphics are described by pixels of color. Bitmap images are resolution dependent; therefore, resizing a bitmap image can lead to distortion. Vector-based graphics, on the other hand, are drawn using mathematical formulas. Resizing a vector-based image means the formula is recalculated, resulting in a scaled version of the original image and no distortion.

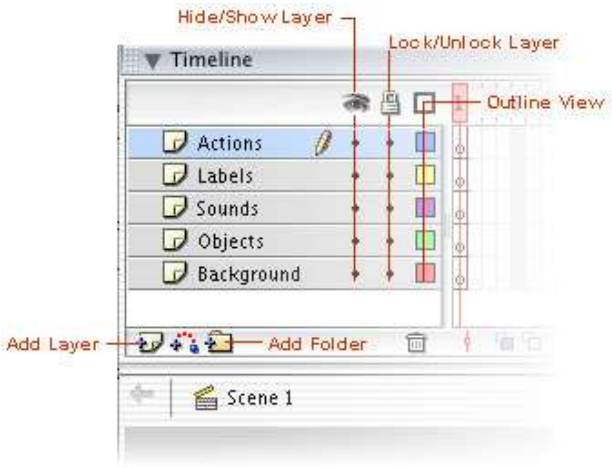
Flash MX supports the importing of bitmap graphics. This feature gives the Web developer a great deal of flexibility. The addition of a Flash movie to a Web site invites interactivity and provides an opportunity for visitors to have a more engaging experience.

Using Layers in Flash Mx/8
Layers are like clear sheets of acetate or glass. Layers can be stacked on top of each other. Each layer can have its own artwork, sound or action on it. Use as many layers as you need to keep everything separate and organized. Be consistent with the way you order and name layers, so that if you have to put the .flaw away and work on it down the road, you won't be totally lost. This is a good practice to adopt. Suggested layer order and layer names are:

1. Actions --- any frame actions assigned can all be here.
2. Labels --- Name keyframes that have actions
3. Sounds --- Any sounds can be placed in keyframes here.
4. Object 1 (whatever you have on this layer)
5. Background

Name layers with short meaningful names that you will be able to identify at a glance. Such as "red ball bounce," or "button home." These aren't read by Flash, so they just have to have meaning to you.

NOTE: You can have as many layers as you want (or as your system can handle). Layers do not add to file size.



Creating New Layers

There are several ways to create a new layer:

- Click on the "+" button in the lower left corner of the Timeline.
  or,
- From the **Insert** menu, choose **Layer**.
  or,
- Right–click (Windows) or Control+Click (Macintosh) on an existing layer and choose Insert Layer from the dropdown menu.

Create a guide layer by choosing **Modify > Layer** and selecting the "guide" option. A guide layer is used to help in drawing. Any layer can be made into a guide layer. The artwork appears in your work area, but not in the published movie. When you see this icon next to the layer name, then it is a guide layer. 

Layer Attributes

You can change the order of the layers by dragging them up or down in the Timeline. The icons across the top of the Timeline control different attributes of the layers.

- The first changes the visibility of the layer to Off or On. This is helpful if you don't need to see that layer to work on a different layer. Even if you have the visibility of the layer turned Off, it will still show up in the published movie. If you want the layer to be visible, but not show up in the published movie, change the layer to a guide layer. The page icon on the layer will then change to a "t-square shape"
- The Lock icon locks the layer so that you can't make changes on it.
- The Box will change whatever artwork is on the layer into an outline. It will show up as the artwork, not as an outline in the movie.

You can also change the height of the layer. This may be necessary if you have a sound file on a frame and you want to see the whole sound wave representation. To change the height, Control+Click on the layer (Macintosh) or right-click (Windows) and choose Properties. Adjust the layer height in the Layer Height box at the bottom of the menu box.

Exercise:

1. Create an object on a layer, click on the eye and test movie. You should see that even though the layer is invisible, the object shows up. (In order to test a movie, it must be saved first.)
2. Change the object to an outline, and do the same thing. It should look the same.

Change the layer type to a guide and test. Even though the object is visible, it won't show up in the test movie.

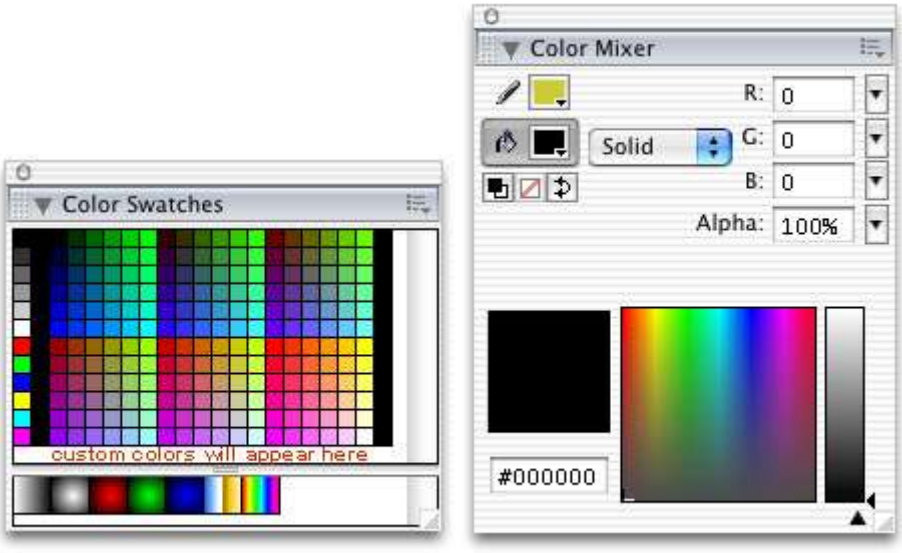
Creating Colors in Flash MX

There are several ways to select color with Flash MX. The Swatches panel displays the 216 Web safe colors and also stores custom mixed colors created by the Mixer panel. To open the Swatches and Mixer panel, from the **Windows** menu and then **Swatches** and then **Mixer.**

When working with colors in Flash, you may want to organize the Swatches, Mixer and Fill panels into one window. These three panels are used together very often with colors, and you can save and recall this panel layout.

Adding a New Color to the Swatches Panel

- Click on the Mixer panel tab. Select the color swatch next to the bucket symbol. This is the Fill color.



Select another color from the drop down swatches menu or by using the sliders from the RGB sliders on the right. These sliders can be switched from RGB (Red, Green, Blue) to HSL (Hue, Saturation, Luminance) or HEX (Hexidecimal) using the left pointing arrow on the Mixer panel.

- Adjust the transparency of the new color with the Alpha text box or slider.

- When the color and transparency is set, select Add Swatch from the right pointing arrow on the Mixer panel.
- Switch to the Swatches panel. The color you created is at the bottom of the color swatches. Now you have a custom swatch palette that you can save and call into other Flash sites.

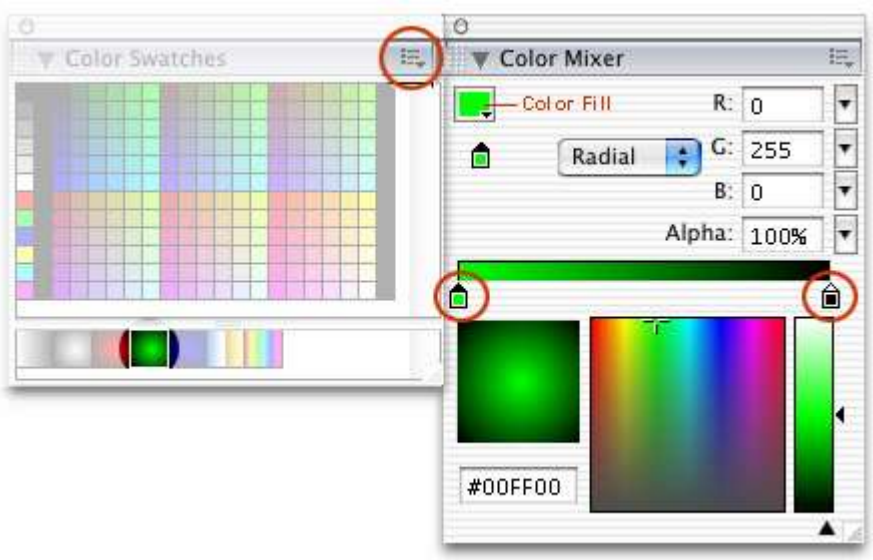
Saving a Custom Swatches Panel

Select the Swatches panel. Use the pull-down menu from the top right arrow and choose Save Colors. Save your swatches in your chosen directory.

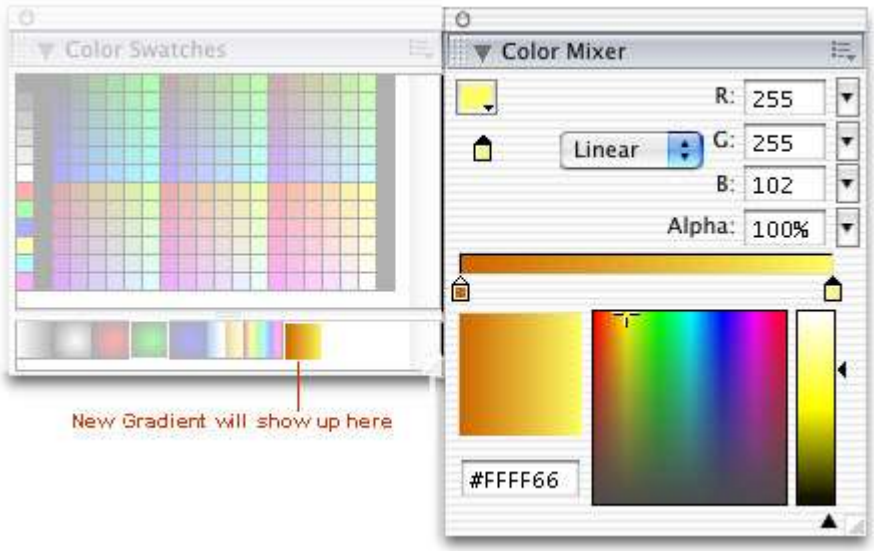
Adding a New Gradient Fill

Let's make a new gradient that has one solid color and one transparent color.

- From the Swatches panel, choose the green radial gradient type. (You can change the type of gradient later.)
- From the upper right arrow choose Duplicate Swatch.



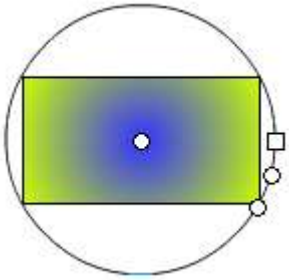
- To alter a portion of the gradient, select one of the color boxes under the bar that defines the gradient.
  Next, use the "color fill" box above to choose another color.
- Now select the other color and choose a new one. You can change the gradient type from the pull-down menu to create various types of gradients.
- Change the gradient type to Linear.
- Now look at your swatches and you'll see the new gradient you made at the end of the gradients.
- Gradient color swatches can only be used in fills. Transparent fills can only be applied to new objects. They will not replace a fill already there. You will have to select and delete the fill before refilling with the transparent color. This might have to be a two-step process. If there is no stroke on your object, add a temporary one with the inkwell tool. Use that to define the area to be filled, then you can delete the stroke again.



Modifying the gradient's size, position and rotation inside of a filled object.

- Make a shape and fill it with your newly created gradient.
- Select the "fill transform" tool on the Toolbox.

Click on your filled shape on the stage.



A center point appears in the center of the shape and with a Radial fill a circular transform modifier surrounds the shape. The square handle is an adjustment handle. Linear and Bitmap fills will a display a center point and two vertical lines on either side of the shape. On the right line, the square adjustment handle appears.

- The Transform Fill handles allow you to change the size, position and rotation of the gradient fill inside the object. When you drag from the center point, the position of the fill will move. The square on the side changes the width, the circle directly below controls scale, and the other circle below that one controls the rotation of the fill. The mouse pointer will change when hovering over one of the handles to indicate the function. Try this with a linear fill to see the difference. In a linear fill, you can modify the same properties as well as skew or slant by dragging from one of the corners.

Bitmap Fills

In the Fill panel, choose Bitmap from the Fill Menu. Click on a bitmap that appears in the Fill panel. You can modify a Bitmap fill the same as a Linear fill.

Drawing in Flash MX

If you're familiar with Photoshop or other drawing, image manipulation and paint programs, then Flash tools will look familiar. Let's go over some of the drawing methods unique to Flash.

Overlapping Shapes

When one shape overlaps another and the two shapes are on the same layer and and are not selected, these objects merge into one object. You can still select one section of the object, but when you move it, the part of the other object that was overlapped will be erased.

Exercise:
Draw a shape on the stage in one color. Draw an overlapping shape in another color. Select one of the color areas and drag the shape. You see that the part of the shape that was covered by the overlapping shape is gone. Keep this in mind when you're creating graphics. If you don't want this to happen, put the shapes on different layers, group them or make them symbols.

Pencil Tool

The Pencil tool works like a regular pencil tool with some cool options. The Pencil tool draws in three modes: Straighten, Smooth and Ink. These modes appear in the Options section of the Toolbox.

Choose Straighten and draw a box. The lines you draw have been straightened. Try this with the other two modes. Smooth makes nice curves averaged from the curve you drew. Ink leaves the line as you've drawn it.

Selecting edges and lines in Flash MX

- To select a line segment or the fill of an object, use the Arrow tool and click once over the line or fill.
- Double clicking on a line will select all the curves in that line.

- Choose the selection (Arrow) tool and deselect any shapes or lines that are on the stage. With the selection arrow, move it over one of the shape edges. The pointer arrow changes when you're over the edge of a shape that hasn't been grouped or made into a symbol. If it's over a curved line or curved edge of a fill, it is a pointer with a curve next to it and will move the curve if you press and drag. Over a corner or endpoint, it is a pointer with a corner, and will move the point or corner maintaining the straight lines. This will pull both stroke and fill alike.
- To reshape lines and fills, select them. Use the Smooth or Straighten mode from the Modify Menu or from the Options section of the Toolbox. (You can adjust the amount of smoothing or straightening by choosing **Preferences** from the **Edit** menu.)
- Control-drag (Windows) or option-drag (Macintosh) a line to create a new corner point.

Optimizing Shapes

The fewer curved sections in a shape, the lower the amount of data Flash has to maintain on that shape. Optimizing shapes is highly recommended to reduce file size and increase playback speed of the Flash movie. To reduce the amount of curves in an object, choose **Optimize** from the **Modify** menu. The Optimize Curves dialog box allows you to reduce the number of curves contained in a shape. Observe the shape as you smooth it and stop before it changes the shape unacceptably. If it's a simple shape, the number of curves may not change at all. Optimizing all objects in a movie, can significantly reduce the movie file size and increase the playback speed.

Exercise:

Draw a shape. Select it and choose **Optimize** from the **Modify** menu. Use the slider to adjust the amount of smoothing to apply to the curve. (You can always choose **Undo** from the **Edit** menu.) The option 'Use multiple passes' will repeat the smoothing process until no more optimizing is possible. This is the same thing as choosing **Optimize** from the **Modify** menu over and over again.

Pen Tool

If you are familiar with Adobe Illustrator or Photoshop, Macromedia Freehand or Fireworks, you've seen this tool before. The Pen tool allows you to place points and create curves and lines that are repositionable.

Exercise to create a straight line:

- Select the Pen tool.
- Click once anywhere in the stage window.
- Click again, and a straight line is created between the two points.

Exercise to create a curved line:

Position the Pen icon where you want it to begin on the stage. Press and hold the mouse and slide in the

Making Buttons in Flash MX

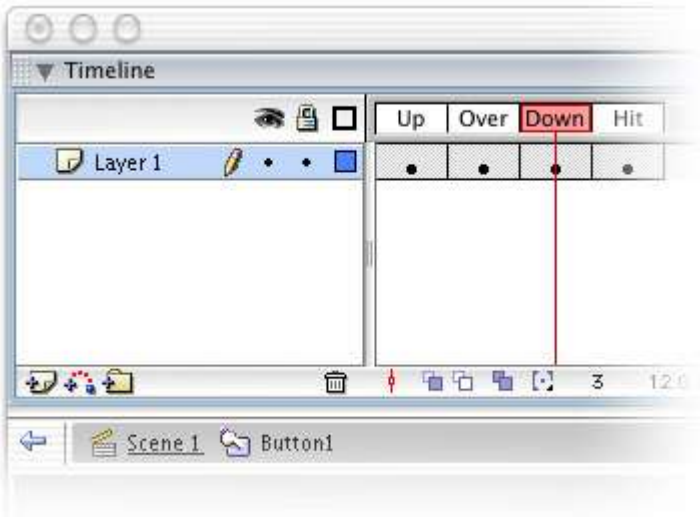
One of the most powerful features of Flash MX is the interactivity you can build into your Web site. The button symbol is something you'll use often. Buttons can be as simple or as complex as you want. Buttons can contain movies and actions inside.

Buttons are actually interactive movies consisting of four frames. The first three frames define three states of the button, while the fourth frame defines the hit area, or the area that is "hot." The four frames are:

1. Up -- what button looks like in inactive state, when the pointer is not over the button;
2. Over -- what the button looks like when the pointer is over it;
3. Down – what the button looks like when the mouse is clicked on it; and
4. Hit -- the area that is “hot” that will trigger the button states.

Exercise to build a simple button:

- From the **Insert** menu, choose **New Symbol**. In the Symbol Properties window that opens, choose Button as the Behavior type. Name it: b, simple button. Click OK.
- This opens up the symbol editing window.



You will see 4 labeled frames in the Layers and Timeline area. These frames represent the 4 states for your button: Up, Over, Down, and Hit. This is where you will create the artwork for the different states for your button.

- Click in the “Up” state frame.
- On the stage, draw a blue filled circle. The Up state is done.
- To create the "Over", "Down" and "Hit" states, select each frame (state) and from the **Insert** menu choose **Keyframe**. This will put the last keyframe (Up state frame) in the selected frame. Then you can go back to each new frame and change the button.
- We want the "Over" state of the button to look like the "Up" state except with a different fill color.

- Select the "Over" state frame. If you click directly on the frame, it will select all objects in that frame. If you click over the frame, on the state label, it will go to the correct frame, but not select the objects in that frame. If the object is not selected, choose the selection arrow and select the blue filled circle on the “Over” state. While it is selected, choose a new color by the fill bucket to change the fill color.
- Now select the "Down" state frame. Let's make the Down state a little different. Change the fill color to yellow the same way as in the Over state. Add some little dashes like radiating rays from the sun around the circle.
- Let's leave the "Hit" state the way it is.
- Click on the Scene 1 icon in the upper left hand corner of the work area.
- If the Library window is not open, select Window/Library. (The key command will be indicated in the pull-down menu, if you'd rather use keys than the pull-down menus all the time.)
- The simple button you made shows up in the Library list.
- Select the button and the "Up" state of the button will appear in the upper portion of the Library window along with a right pointing arrow. Click this right pointing arrow and you'll see all four states of the button played.
- Drag this button onto your stage.
- To see your button in action, from the **Control** menu, select **Enable Simple Buttons**. Rollover your button and click on it. It should turn a different color when you roll over it, and turn into a yellow sun when you click on it.

Exercise to create a button to perform an action:

One of the most common button actions is to call up a different URL. We'll put this action on the button we just made and placed on the stage.

- From the **Control** menu turn off **Enable Simple Buttons** so you can interact with the button on the stage without testing the movie.
  (Enable Simple Buttons will not show you movies in keyframes.)
- From the **Window** menu, choose **Actions**. This will open up the Object Actions window, one of the most powerful tools in Flash MX. Check and see that it is set on Normal mode and not Expert Mode by clicking on the arrow in the upper right corner. (Normal mode gives you clues and blanks to fill in with the proper syntax. Expert mode allows you to type in your own syntax.) For our purposes, we will choose Basic Actions from the list on the left hand side. Scroll through the list and select “Get URL”.Either double click on it, or drag it over to the right hand side. It will show up with the proper syntax, and an area to fill in on the bottom of the Object Actions window. (This action defaults to the “on release” action. If you want to change this, click on that line in the Object Actions window, and you'll see a new set of options for that line on the bottom.) With "Get URL" selected, Flash asks you for the URL you want this button to direct the user to. If you don't have a URL to use, type in this one: http://www.utexas.edu/
- Now you're ready to test your button. There are several ways to do this:
    1. from the **Control** menu, choose **Test Movie** or
    2. from the **File** menu, choose **Publish Preview/HTML** or

3. from the **File** menu, choose **Publish Preview/Flash**.

Test the button by rolling over it and clicking on it.

Exercise to create a fancier button:

- Select the b, simple button in the Library window. Click on the Options arrow and from the drop-down menu select Duplicate. Name this button 'copy b, movie button'. Leave the behavior as Button. Click OK.
- Create a movie to place in the Down state of the button.

- First, let's copy the sun and rays from the Down state of the simple button.
-  Click on the Symbols icon in the upper right hand corner of the stage, scroll down and choose your old button. The symbol editor opens.
- Click on the Down frame, select the sun and rays, and from the **Edit** menu, choose **Copy**.
- From the **Insert** menu, choose **New Symbol**. Select Movie Clip and name it 'm,sunny'. Click OK. The symbol editor is now open.
- From the **Edit** menu, choose **Paste in Place**. You should see your sun from the "Down" state of the simple button.
- From the **Insert** menu, choose **Keyframes** at frame 5 and frame 10.
- Go to frame 5. Scale the sun up a little (i.e., make the image bigger). From the **Modify** menu, choose **Transform** and then **Scale**. Grab one of the square handles and make the image a little bigger. Now lets make a tween between keyframe 1 and 5, and 5 and 10.
- Click on a frame between 1 and 5. The area on the Timeline turns black.
- Now open the Frames panel. From the **Window** menu, choose **Panels** and then **Frame**. Choose Shape Tweening from the pull-down menu. (Shape tweening is for objects that have not been turned into symbols or grouped together. You can tell if it's a raw graphic by the selection. It will select and fill the object with black and white noise if its still raw. Grouped objects and symbols will have a bounding box instead. The shape tween is represented by a light green background with a solid line arrow on top. Dashed lines means something is not right.)
- The tween should make a smooth upward scale between frame 1 and 5. Do the same thing between frame 5 and 10.

Let's make this even more fun.

- Add a layer above the scaling sun layer. Name it 'spiral.' Double-click on the Layer to change the layer name.
- Insert a keyframe on the first frame.
- Select the first frame on the spiral layer and draw a darker yellow or orange spiral on top of the sun.
- Select the spiral and make it a symbol by choosing **Convert to Symbol** from the **Insert** menu. Name it 'm,spiral movie clip.' The behavior is movie clip.
- Create a Keyframe in frame 10.
- Select frame 10 which still contains the unconverted graphic. Select the image, delete it and drag the spiral symbol from the Library window into its place. You

can use the arrow keys to line up the center crosshairs. It should be in registration with the first one on the first frame.

- Let's make the spiral rotate one complete revolution for 10 frames. Select the area between frames 1 and 10 and open the Frame panel (from the **Window** menu, chooose **Panels** and then **Frame**). In the Tweening box select Motion. In Motion Tweening, you can indicate the direction of the Rotation. Select CCW for counter-clockwise. In the "times" box, put 1 for 1 revolution. Motion tweens work on symbols. The tween on the Timeline will look blue, with a solid line arrow.
- Create a new layer and move it to the top. Call it 'actions.'
- On the actions layer, select frame 10 and insert a keyframe by choosing **Keyframe** from the **Insert** menu.
- Choose Window/Actions. The Frame Actions window opens.
- Select Basic Actions and “Go To." It will be an action on this frame sending it to a frame number or label. It defaults to frame 1 which is fine in this case. Now this movie will loop endlessly. (The timelines in movies are independent of the Timeline on the main stage, so frame 1 is unique to this movie.)

Putting the movie into the button

- Click on the symbols icon in the upper right hand corner and select 'copy b, movie button.' You are still in the symbol editor, but you are back in the duplicate of the first button you made.
- Go to the Down state (frame) and delete the sun graphic.
- From the library, drag in the copy b, movie button you just finished making. You can line up the center crosshairs with the arrow keys.
- Go back to Scene 1 (upper left). Now your movie button has an animated down state.
- Test your movie.

Using Symbols in Flash MX

One of the most efficient ways to use Flash MX is to use symbols whenever you need more than one occurrence of a graphic, or animation or sound. A symbol is a graphic, button, movie clip, or sound that you create once, then reuse in your movie or other movies. Flash allows you to have an item saved in the Library that you can pull out onto your stage over and over again without busting your file size budget. You can also modify certain parameters on each instance such as scale, rotation, alpha and overall color. If you modify the symbol, it will change all the instances, but if you modify the instance, then just that instance is affected.

Symbol Types

- **Movie Clip** - A symbol that can include animation(s), buttons, graphics, other symbols, and operates independently of the main stage Timeline.
- **Button** - A symbol that allows interactivity and that responds to mouse clicks or rollovers. Buttons can also have movies placed in one of the states.
- **Graphic** – A symbol that is used for static images or is controlled by main stage Timeline. Interactive controls and sounds won't work in this symbol.
- **Sound** - The Sound symbol is created automatically when you import a sound.

- **Font** - Create this type of symbol from the Library window options so you don't have to embed fonts in your site.

Naming conventions

The Library stores symbols in alphabetical order. In order to keep symbol types grouped together, name each symbol using the first letter of the type of symbol that is it first, followed by a comma, then the name of the symbol. When you are working with many symbols, this helps you find the particular symbol quickly and consistently.

Example:

- g, balloon
- m, balloon movie
- s, balloon squeaky sound
- b, balloon button
- ac, balloon movement action script in movie

Exercise:

1. Create a simple shape on the stage, such as a red balloon.
2. Let's save this shape into the Library by making the balloon a graphic symbol.
    - Using the Arrow tool, draw a selection box around the whole balloon.
    - With it selected, from the **Insert** menu, choose **Convert to Symbol**.
    - A Symbol Properties dialog box appears with 3 Behavior options. Flash defaults to the Movie Clip behavior.
    - Select the Graphic behavior.
    - Name your symbol: g, balloon. Click OK.
3. Delete the balloon from the stage.
4. From the **Window** menu, choose **Library**. You'll see your balloon symbol in the Library window.
5. Drag the balloon symbol (g, balloon) out of the library onto the stage.
6. Now you have an “instance” of the balloon on the stage.
7. Let's change the appearance of the balloon instance.
    - Using the properties menu, change the alpha to 50%, change the tint, and change the scale and rotation.
8. You'll see the original symbol is unaffected. You can keep pulling in the same symbol over and over again but change each instance any way you want. These attributes can also be changed for different keyframes.

Note: If you plan on animating each instance independently, put each instance on a separate layer. Animation requires "tweening", which will only work on one symbol on one layer at a time.

Making a movie symbol from the balloon graphic symbol

- From the **Insert** menu, choose **New Symbol**.

- Choose Movie Clip as the Behavior if it's not already selected, and name your symbol: "m, balloon movie." Click OK. Movie symbols have their own timeline. This can get confusing.
- You are now in the symbol editor, which looks just like the stage (because it has its' own independent timeline). One way you can tell you are in symbol editing mode is by looking at the upper left hand corner, which will have the movie symbol you just named after the scene name.
- In the Library window, drag the "g, balloon" symbol out onto the m, balloon movie stage.
- Select frame 10 on your timeline and then from the **Insert** menu, choosing **Keyframe** from the top menu. (Keyframe will duplicate the last keyframe with something in it.)
- Move the animation scrubber to the second keyframe and select the balloon or click on that frame (this will pick everything in that keyframe).
- Modify the balloon instance by rotation, scale, alpha and/or color.
- Now we need to create a "tween" between the first keyframe and the second keyframe. To do this:
    - Click in the Timeline somewhere between the first and second keyframes.
    - In the property inspector, select the Motion option from the Tween dropdown menu. You can only use the motion tween on symbols. You can also add automatic rotation, easing, and other options at this point in the frame window.
    - There should be an arrow displayed in the Timeline area between the two keyframes. If you select something else, the motion tween should show up on a light blue field. (Shape tweens are green.)
    - Now if you scrub through the frames, you'll see the balloon changing smoothly from frame to frame.
    - Let's make this action loop continuously. To do that, let's assign an action to the last keyframe, telling it to return to the first frame of its Timeline again.
    - Click on the last frame and choose window>actions, the Frame Actions dialog box appears.
        - Make sure you are in normal mode in the Frame window. (check upper right pull-down arrow)
    - Click on the "Actions" book display action catagories, select the "Movie Controls" book and find GoTo. It will default to frame 1, which is fine. Now, the movie will loop continuously.

Button Symbol

- Now we will make a button using both the g, balloon and the m, balloon movie. The button symbol has 4 default states: Up, Over, Down and Hit. These states reflect what's happening when the button is doing nothing, when the mouse is over it, when the mouse clicks on it, and the hit zone for the button.
- Go back to the main stage.
- From the **Insert** menu, choose **New Symbol**.

- Choose Button option and name it: 'b, balloon.' You are now in the symbol editing mode again.
- Select the “Up” frame. Drag the g, balloon symbol from the Library onto the stage. This is what the button will look like “at rest.”
- For the "Over" state, we want the exact same position, but another color, so we're going to copy the “Up” frame and paste it into the “Over” frame (or, click to select the "over" frame and hit F6, or you can also just choose Keyframe from the Insert menu, and it will copy the contents of the last keyframe).
- change the color of the balloon using the property inspector.
- Now when you rollover this button, it will change color.
- Next, select the “Down” keyframe area. From the **Insert** menu, choose **Keyframe** and drag in m, balloon movie from the Library.
- For the “Hit” area, it doesn't show at all, so you can use the symbol or draw a hit area in this state.

Your button is created now. Let's test it.

- Go back to Scene 1 and drag your new “b, balloon” out onto the stage.
- Test your button by choosing **Test Movie** from the **Control** menu.

Congratulations! You've made a movie, and a button with a movie in it!

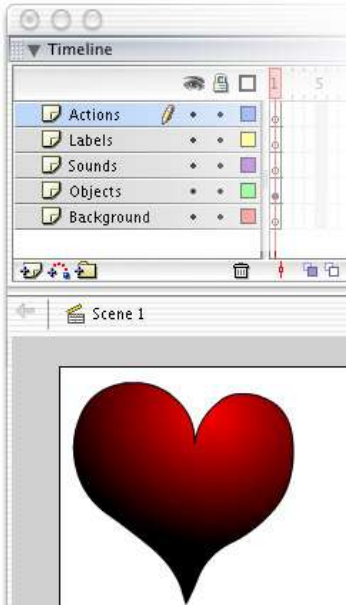
Animating in Flash

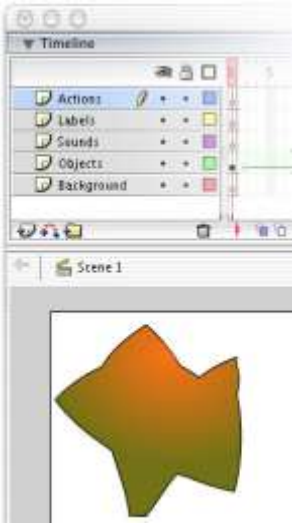
Animation in Flash is easy! Let's look at a couple of different ways to achieve animation in Flash.

We built tweens in the “Button and Symbols making" tutorial, but let's do a quick review.

Shape Tween

- Draw a shape at frame 1. (Try a simple shape, like a heart.)
- Insert a blank keyframe at frame 30 in the Timeline. Draw another shape (like a star) in a different color at frame 30. The two shapes should be on the same layer. Click on the area between the two frames in the Timeline. The area will turn black.





- Now, open the Frames panel and choose Shape Tween. A light green bar with a solid arrow will appear between the two frames. When you scrub back and forth in the Timeline, the heart morphs to the star, and changes color.

Motion Tween

- Create a symbol or pull one out of the Library onto a new layer on frame 1.
- Insert a keyframe at frame 30. This will keep the symbol on frame 30.
- Modify the symbol by changing the position, scale, rotation, and alpha. (Use the Effects panel for the alpha). Select the area between the frames on the timeline and it will turn black.
- Open the Frames Panel and choose Motion Tween. A light blue bar with a solid arrow will appear between the two frames. When you scrub back and forth in the Timeline, the symbol will move, scale, rotate and the alpha will change.
- Motion Tween has other options. You can choose the direction of the rotation from the pull-down menu in the Frames Panel and have the tween automatically rotate the object the number of times you specify.

Animations can be reused. For example, let's say you create an animation in a movie symbol of a bouncing ball. You could pull that ball out of the Library as many times as your computer can handle. You can put them all on independent layers, animate the position, scale, tint, alpha, etc.

Animating along a Path

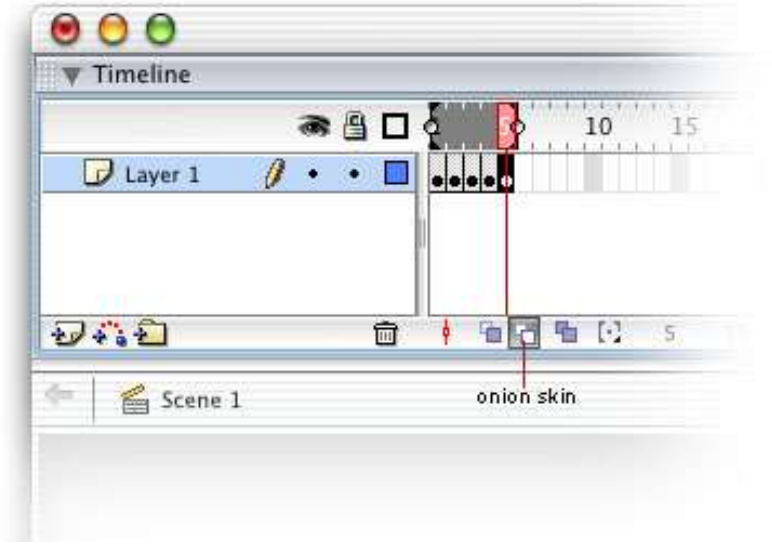
Let's say you want a small airplane to follow a path from the ground up to the clouds.

- Create a movie symbol of a little airplane with an animated propeller.
- Drag the symbol out onto the stage. You may want to add a runway and some clouds in the sky to make your animation more complete.
- Select the airplane layer and from the **Insert** menu, choose **Motion Guide**. This creates a new layer called a Guide Layer on top of the airplane layer.
- Insert another keyframe in the Guide:airplane layer at the point where you want the motion to stop.
- Using the Pencil tool, draw a curve that you want the airplane to follow from the runway to the clouds. (It won't show up in the movie)
- Select the first keyframe in the airplane layer. Select the Arrow tool and move the airplane to the beginning of the curve.

- Now select the last keyframe. Move the airplane along the curve to the end of the curve.
- Double-click the first keyframe to open the Frame panel. Choose Motion from the Tweening menu and check Snap (snaps the registration point to the path).
- Scrub along the Timeline, and the airplane will follow the curve from first to last frame. When you publish, the plane's propellers will move. (Movie animations do not display in Flash on the stage.)

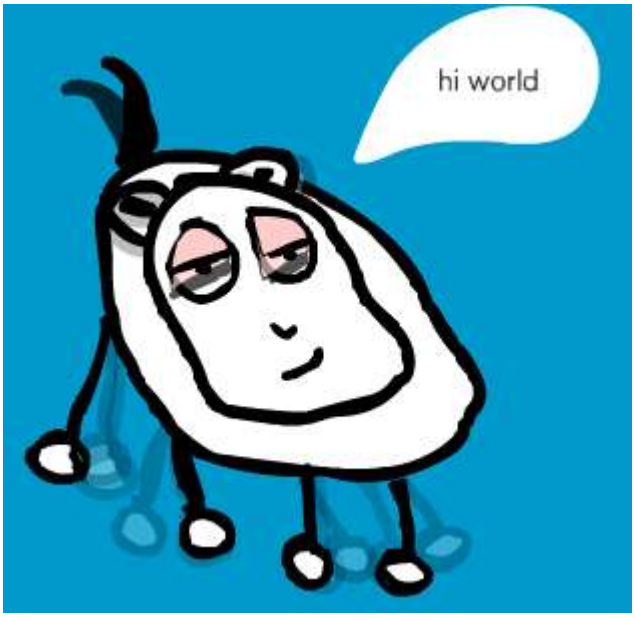
Frame by Frame Animation

This is the most file-size intensive method, but sometimes it's the only way to get the effect you want.You can turn on a nifty feature called Onion Skinning to see surrounding frames so you can more accurately draw your frame.



Exercise:

- In frame one, draw a dog.
- In frame seven, from the **Insert** menu, choose **Keyframe**. This copies the dog from frame one through frame seven.
- Insert a new layer above the dog layer. You can call it "feet."
- Click on the Onion Skin icon from the options under the Timeline. You will see a faded version of the first frame.
- Click and move each of the dog's legs.



- If you like, you can continue to do this for another few frames for the eye lids and the tail.
- Turn off the Onion Skin button.
- Now scrub through your animation. You now have a dancing, tail-wagging winking dog.

Using Sound in Flash MX

There are several ways to use sound in Flash MX. You can loop sounds to play continuously, independent of the timeline, or synchronize your sounds to an animation. You can import and edit your sounds in the Sound Panel.

Importing

Import sounds into Flash by choosing **Import** from the **File** menu. The following audio file formats can be imported into Flash:

- WAV
- AIFF
- MP3

If you have QuickTime 4 or higher, you can also import these file formats:

- Sound Designer II (Macintosh)
- Sound Only QuickTime Movies (Macintosh and Windows)
- Sun AU (Macintosh and Windows)
- System 7 Sounds (Macintosh)
- WAV (Macintosh and Windows)

Flash stores audio files in the Library as a symbol along with the graphics, movies and buttons. Sound files are indicated by a loudspeaker icon. Because the sound is a symbol,

you only need one copy of it and you can use it in many ways in your Flash movie. In the Library and on the Timeline, audio files are represented as sound waves.

Two types of sounds in Flash -- event sounds and stream sounds

- Event sounds must download completely before they start playing and they will continue playing until explicitly stopped (usually by a stop action). Event sounds are associated with an event such as a mouse click, and are independent of the Timeline.
- Stream sounds begin playing as soon as there's enough data to play. These sounds are synchronized to the timeline. Flash forces the animation to keep in sync with the sound. If it can't draw frames fast enough, Flash will drop frames to keep the sound in sync.

Editing with the Sound Panel

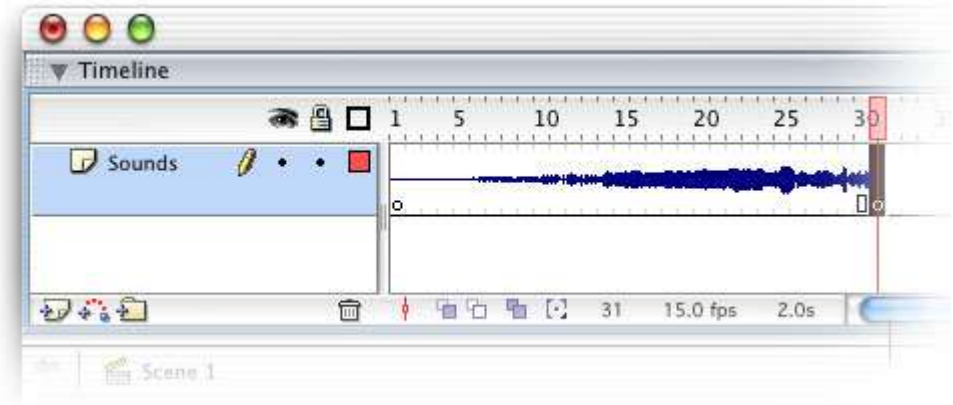
The Sound Panel allows you to modify the properties of the selected sound, for example:

- specify if the sound is an Event or Stream
- start a new sound, or Stop the sound
- apply effects to the sounds, such as None (which strips any previous effects off the sound), Fade In or Out, Left to Right, etc.
- If you choose Custom, you can modify the in and out points of the fade, the duration, etc. in the window that opens up. Click on the boxes at the corners of each channel and also add points along the line by clicking.

Exercise:

- Import a sound file by choosing Import from the File menu. You can download a sample .wav file named slide_whistle_3x.wav. If you have Flash you can download the finished sound.fla file. If you don't have Flash, a 30-day trial version is available at Macromedia.

- Open the Library Panel, and click on the sound symbol you just imported. You will see it in the Library window with a play arrow. You can listen to the sound you just imported by clicking on the arrow at the top right of the Library window.
- Name and select layer: Sounds
- Add keyframes at frame 1 and frame 45 (or somewhere on the Timeline past the duration of the sound). It's not necessary to do this, but it gives you a look at the file in the Timeline.
- Drag the sound onto the stage. You should see it on the layer as a waveform. You can get a better look at this by increasing the height of your layer.
- From the **Modify** menu, choose **Layer**.



- At the bottom of this window, you can change the layer height. Change it to 200% and click OK.

- Open the Sound Panel. Select the sound on the Timeline. The information about that sound file will show up in the Sound Panel. It will also be visible in the top pull-down menu. If you have other sounds imported, they will be in the menu also. You can change the sound file you're editing from here. Directly under that you will see the sound file information, MHz and file size.

You can apply different effects to your sound

1. None - choosing this will remove any previous effects
2. Left channel
3. Right channel
4. Fade left to right
5. Fade right to left
6. Fade in
7. Fade out
8. Custom – when you choose this, another window will open, allowing you to edit both audio channels.

The edit button opens up the same audio channel editor. Choose each effect, open the editor and look at each effect. If you click in one of the channels, it will change to Custom, but these effects can be used as starting points to get the effect you want or use them as is.
The best way to get a feel for the sound editor is to apply each effect to your sound, listen to it and try some things in the custom effect editor.

Below the effects panel is the sync menu. It has the following options:

1. Event – associated with specific actions, not tied to the timeline (like a sound accompanying a mouse click)
2. Start – same as Event, but will create a new instance of that sound if it is already playing.
3. Stop – stops the specified sound
4. Stream – syncs the sound with the animation, forces the animation to play at the same rate as the sound, dropping frames if necessary.

You can loop the sound the number of times you type in the box. Don't loop streaming sound; it will increase your file size dramatically.

Have fun!

Saving and Publishing in Flash MX/8

Saving Flash movies

To save your working site with all its layers, animations, etc. from the **File** menu, choose **Save**. Flash will save it as a Flash movie with a .fla extension. This is similar to Photoshop where you save your working site with all its layers, channels and text as a .psd file. Archive this file. When you're ready to show your creation to the world, you

must publish or export your .fla file into a file format that will play back on the Web or be viewed as a still image or a series of images.

Publishing Flash movies on the World Wide Web

One of the most popular ways to make your multimedia project available to the world is by putting it on the World Wide Web. An easy way to accomplish this is by choosing **Publish Settings** from the **File** menu. The Publish command creates the Flash Player file (SWF) and the HTML document that inserts your Flash Player file into a Web browser.

To publish your Flash project, first click **Publish** then **OK**.

This dialog box gives you the opportunity to create different file formats -- GIF, JPEG, PNG, and QuickTime -- and the HTML needed to display them in the browser window. These alternative formats enable a browser to display your movie's animation and interactivity for users who don't have the Flash Player installed.

Introduction to Adobe Fireworks

Pictures and image working are best handle by design programs and fireworks to be specific is the popular for web images

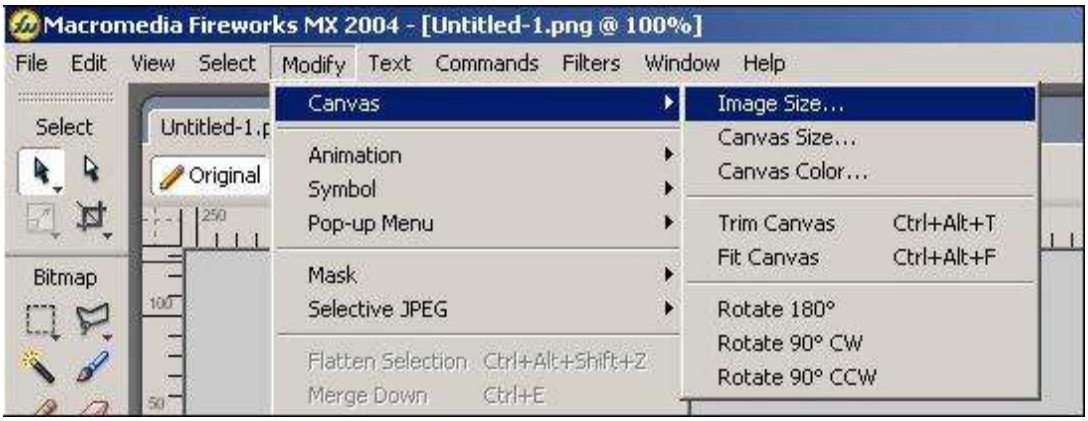
How to edit images in Macromedia Fireworks

Image Size and Canvas Size

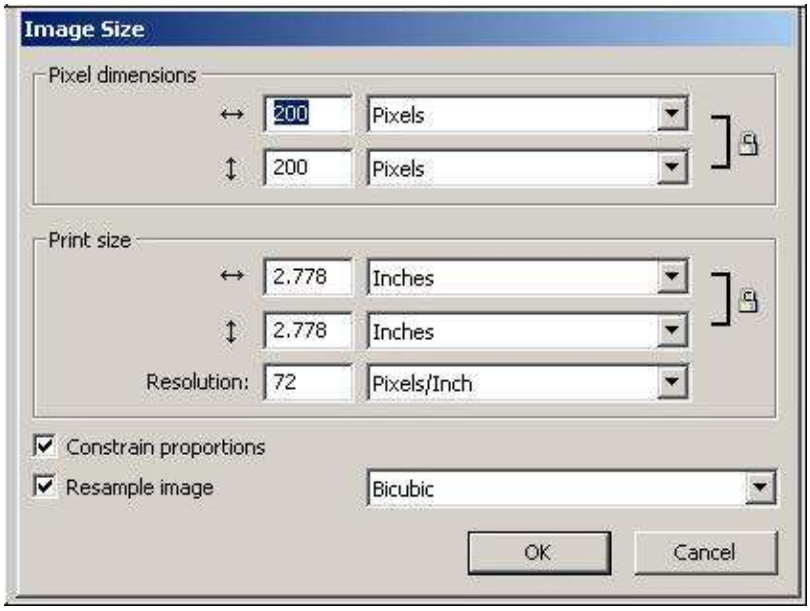
These are two different things. If you adjust the image size it will change the size of your image. If you adjust the Canvas Size it will change the size of the canvas or background that your image is on.

To adjust the image size:

- Go to **Modify – Canvas – Image Size**
- You should see…



- Adjust to the size you want to use the finished image at.



To adjust the Canvas Size:

To adjust the **Canvas Size**, do as above but select **Canvas Size** instead of Image Size.

Cropping

Sometime there is part of an image that you don't want included or perhaps you just want to change the shape of your image.

- Open your image in Macromedia Fireworks
- Select the Crop Tool

- Click and drag over the area you want. You can adjust this by moving the black square handles that appear (see below).

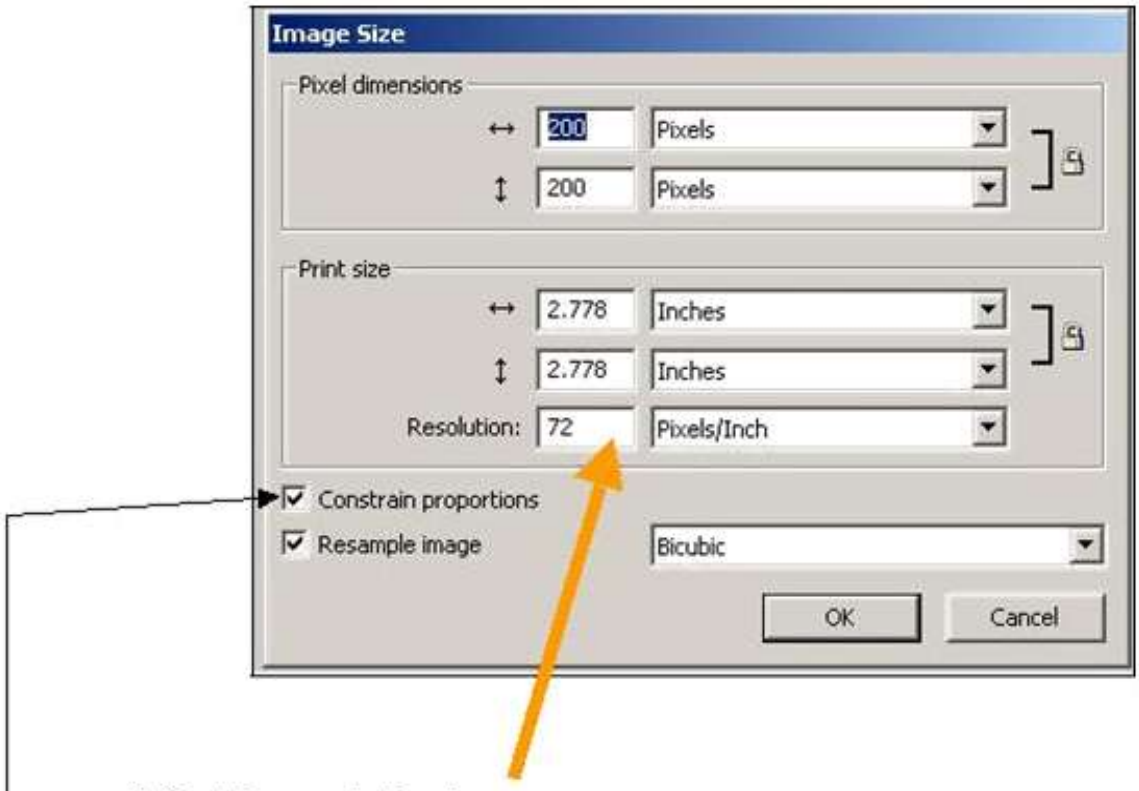
- When you are happy with your selection, **Double Click inside the area you have selected** to crop it.

- Go to **File – Save As – Call it 'Cropped image of ……..'**
- Using **Save As** means you can keep the original picture as well in case you want to use it again.

Resolution

To change the resolution of an image:

- Got to **Modify – Canvas – Image Size**



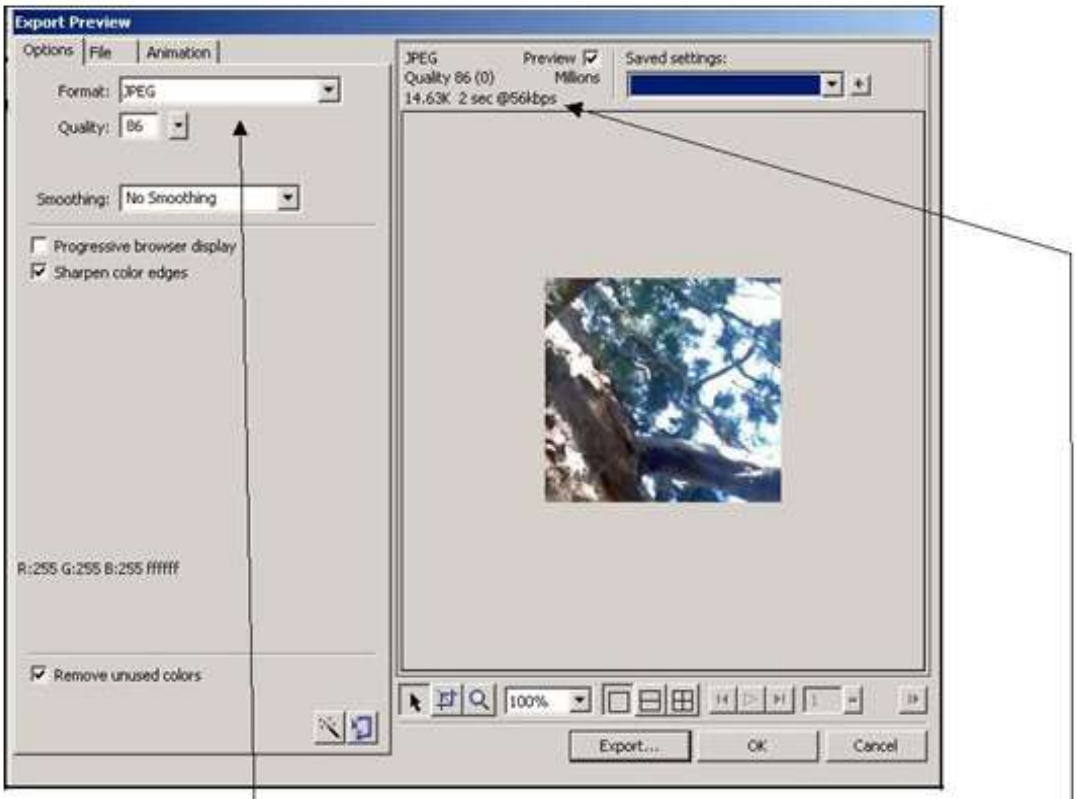
- Adjust the resolution here.
- If the Constrain proportions box is checked and you make the resolution smaller, the image dimensions will get smaller as well.

4. Exporting

To export or change the file format:

- Open the image you want to use.
- Got to **File – Export Preview**

- You should see...



- Use the **Format Box** to select the **File Format** you want
- You can make other adjustments to the quality of the image by adjusting the Quality box and the smoothing box, all of which will have an effect on the overall file size, which is shown above the image.

- Use the **Format Box** to select the **File Format** you want
- You can make other adjustments to the quality of the image by adjusting the Quality box and the smoothing box, all of which will have an effect on the overall file size, which is shown above the image.
- When you are happy with the settings click on **Export** and save the image.

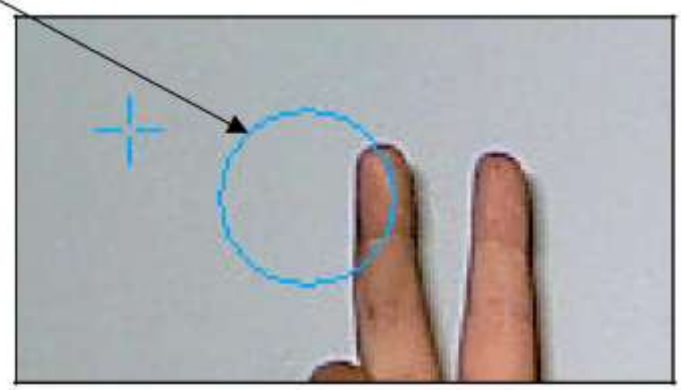
5. Cloning

This tool is really handy for editing out unwanted items from a picture:

- Open the image you want to edit
- Select the Cloning Tool

- Select an area of the picture that you want to clone or copy from.
- In this example I want to get rid of the hand behind the head so I have selected the plain white colour in the background here…

- When you select an area it then shows you the brush size you are about to use.



- This can be changed here…

- Use the tool like an airbrush cover up the section you don't want.
- If you want to change the area you are getting the sample from use the **ALT Key** on the keyboard and left click the mouse on the area you want to sample or select the clone tool again.

- Go to file **Save As** and save your edited image.

- Using **Save As** means you can keep the original picture as well in case you want to use it again.

Adding Text/type on image

To add text to an image:

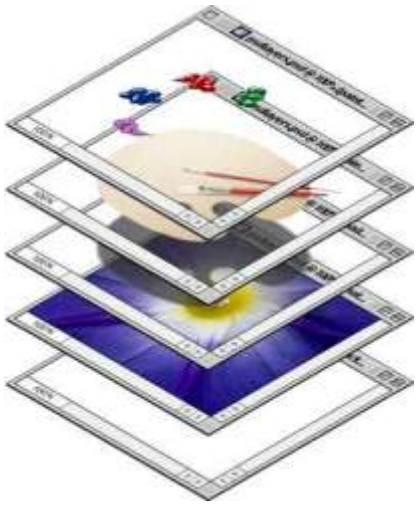
- **Open** the image that you want to add text to.
- Select the **Text Tool** and click on the image

- Type out the text you want
- You can make changes to the colour, size, font, effects etc in the properties box at the bottom of the screen

- Remember to **select the text** to get the **Properties Box**

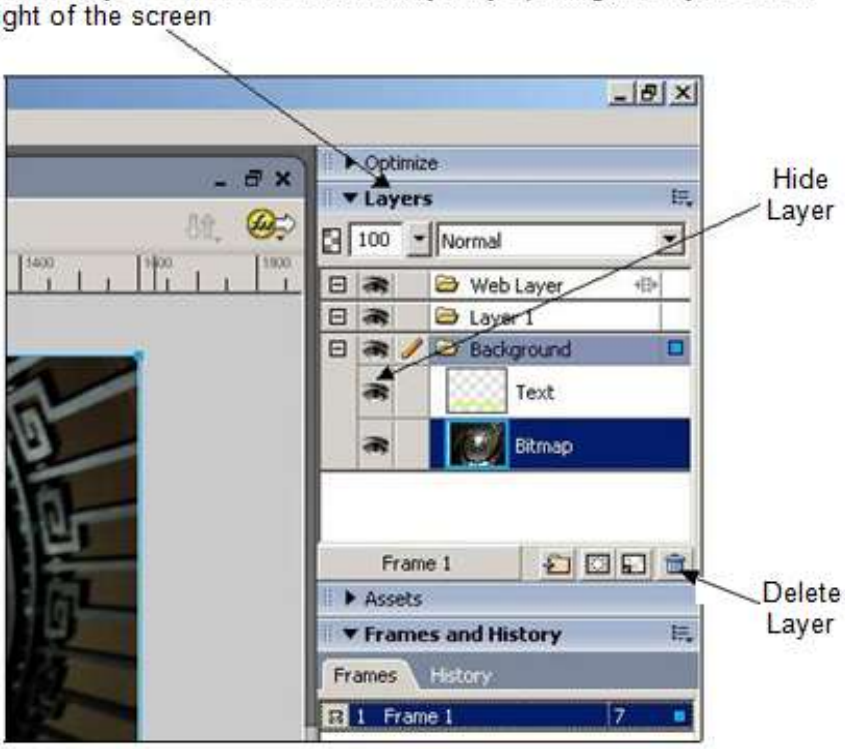
- When you are happy with you text-effect, go to **File** and **Save As**, give it a sensible name and save it into the correct folder.
- Using **Save As** means you can keep the original picture as well in case you want to use it again.

Layers



- As you add text or other images on top
- of your original image you are creating
- new layers. Think of it as lying down
- see-through sheets on top of each other.

- In Fireworks, you can see the different layer by opening the Layers tab on the right of the screen



- As each object is added to the original it adds a new layer. Each layer can be changed without effecting the other layers.

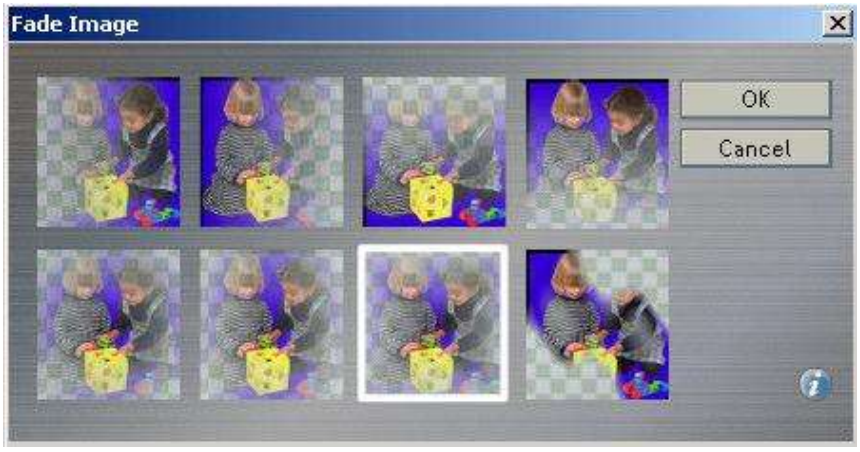
- You can select an object by selecting its layer in the Layers panel.

Image Effects

1. Fading Images

This is a nice effects and blends images into the page nicely:

- Open the image you want to fade
- Got to **Commands – Creative – Fade Image**

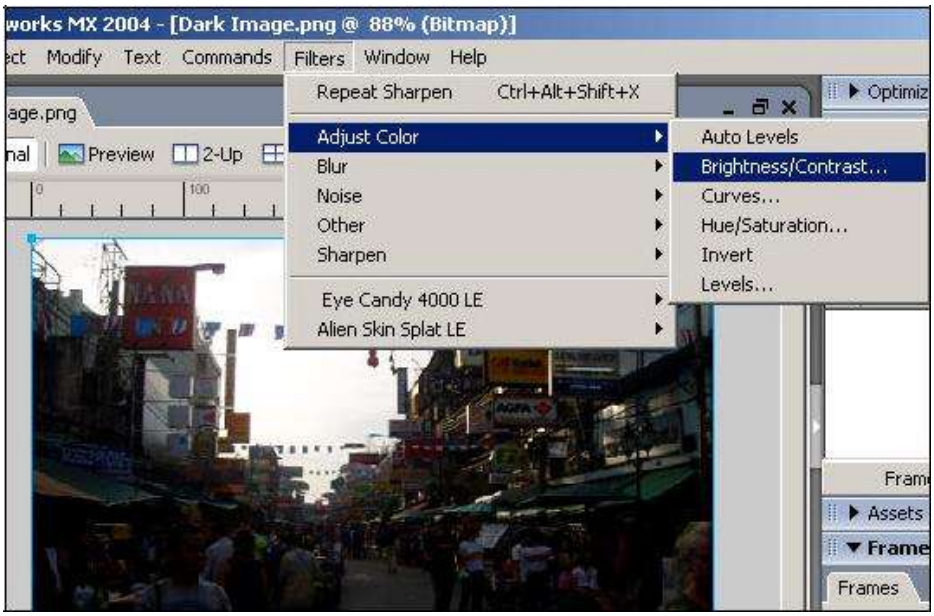


- Select the edges you want to fade – **click OK**
- You can adjust the amount of fade by moving the black handles

2. Brightness and contrast

This is really useful for lightening up dark images or giving your flat images more definition.

- Open the picture you would like to lighten up
- Go to **Filters – Adjust Colour – Brightness/Contrast**



- Use the sliding bars to adjust the setting until you are happy with the image.

- Go to **Save As** – **Save** you picture in the correct folder with a suitable name.

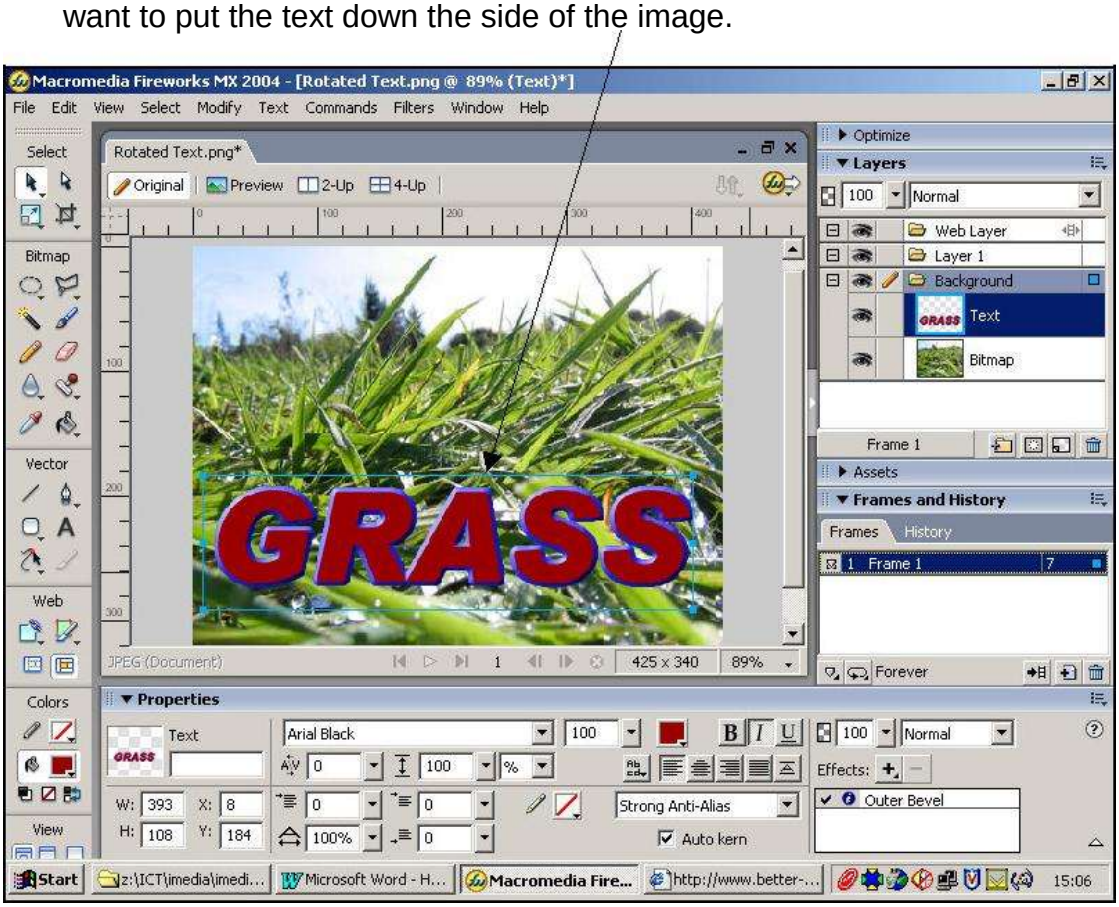
Example:

Original Image

Adjusted Brightness and Contrast

3. Rotating Objects

- Open the image and add the object you wish to rotate. In this example I want to put the text down the side of the image.



- Select the object that you would like to rotate
- You can either:

  Hold down **Ctrl + T** and use the mouse to click and drag the shape around or…

  Got to **Modify – Transform – Free Transform** (or chose to rotate by 90° or 180° etc)

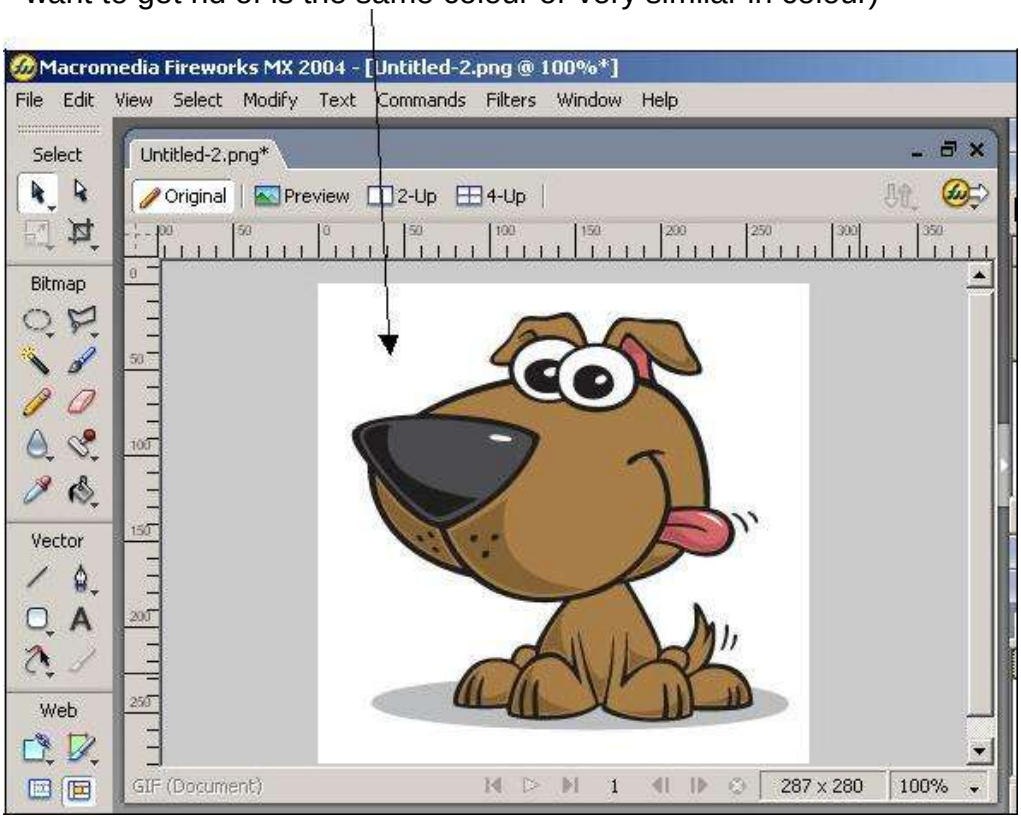
When you are happy with it go to **File – Save As** and save into the correct folder using a sensible file name.

4. Magic Wand Tool

This tool is really good for removing the backgrounds or unwanted sections of images.

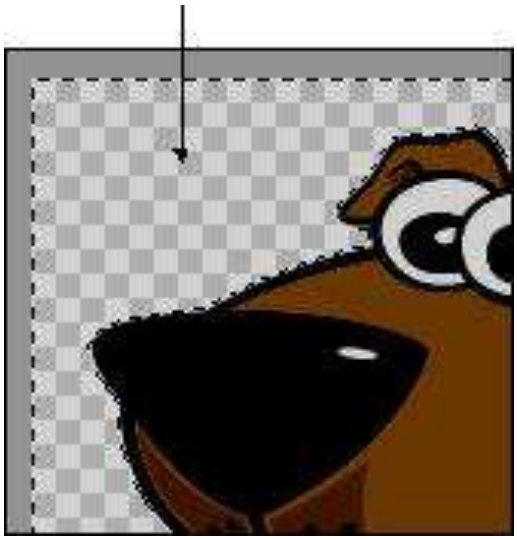
- Open the image that you want to change.
- Select the Magic Wand Tool

- Click on the area that you want to get rid of (it only works if the section you want to get rid of is the same colour or very similar in colour)



- You will notice that the section has been selected. Press Delete.
- To make sure it has been deleted – Got to **Modify – Canvas – Canvas**

  Colour – set to Transparent. (Checked Patern)



- You can make the background transparent when it has been put onto another layer.
- Paste your object onto another layer as shown below…

- Make sure the top layer (dog) is selected.
- Use the **Magic Wand Tool** to select the white area (or area you would like deleted).
- Press **Delete**
- Go to File - **Save As –**

Save your image in the correct folder with a sensible file name.

TOPIC 5: JAVA SCRIPT AND ACTIVE SERVER PAGES

T5.1) Describing Java Script and ASP

JavaScript is a programming language commonly used in web development. It was originally developed by Netscape as a means to add dynamic and interactive elements to websites.

JavaScript is a client-side scripting language, which means the source code is processed by the client's web browser rather than on the web server. This means JavaScript functions can run after a webpage has loaded without communicating with the server. For example, a JavaScript function may check a web form before it is submitted to make sure all the required fields have been filled out. The JavaScript code can produce an error message before any information is actually transmitted to the server.

Like server-side scripting languages, such as PHP and ASP, JavaScript code can be inserted anywhere within the HTML of a webpage. However, only the output of server-side code is displayed in the HTML, while JavaScript code remains fully visible in the source of the webpage. It can also be referenced in a separate .JS file, which may also be viewed in a browser.

Below is an example of a basic JavaScript function that adds two numbers. The function is called with the parameters 7 and 11. If the code below were included in the HTML of a webpage, it would display the text "18" in an alert box.

```
<script>
  function sum(a,b)
  {
    return a + b;
  }
  var total = sum(7,11);
  alert(total);
</script>
```

JavaScript functions can be called within <script> tags or when specific events take place. Examples include onClick, onMouseDown, onMouseUp, onKeyDown, onKeyUp, onFocus, onBlur, onSubmit, and many others. While standard JavaScript is still used for performing basic client-side functions, many web developers now prefer to use JavaScript libraries like jQuery to add more advanced dynamic elements to websites.

ASP has two different meanings in the IT world: 1) Application Service Provider, and 2) Active Server Page.

1) Application Service Provider
An Application Service Provider is a company or organization that provides software applications to customers over the Internet. These Internet-based applications are also known

as "software as a service" (SaaS) and are often made available on a subscription basis. This means ASP clients often pay a monthly fee to use the software, rather than purchasing a traditional software license. Some SaaS applications can be accessed via a web browser, while others operate over a proprietary secure port.

2) Active Server Page
An Active Server Page, commonly called an "ASP page," is a webpage that may contain scripts as well as standard HTML. The scripts are processed by an ASP interpreter on the web server each time the page is accessed by a visitor. Since the content of an ASP page can be generated on-the-fly, ASP pages are commonly used for creating dynamic websites.

ASP is similar to other scripting platforms, like PHP and JSP, but supports multiple programming languages. While the default ASP language is VBScript, ASP pages can include other programming languages as well, such as C# and JavaScript. However, alternative languages must be defined before the script code using the following declaration:

<%@ Page Language="C#"%>

ASP pages are part of the ASP.NET web application framework developed by Microsoft. Therefore, ASP pages are most often found on Windows-based web servers that run Microsoft Internet Information Services, or IIS. You can tell if you are accessing an ASP page in your browser if the URL has an ".asp" or ".aspx" suffix.

PHP Stands for "Hypertext Preprocessor." (It is a recursive acronym, if you can understand what that means.) PHP is an HTML-embedded Web scripting language. This means PHP code can be inserted into the HTML of a Web page. When a PHP page is accessed, the PHP code is read or "parsed" by the server the page resides on. The output from the PHP functions on the page are typically returned as HTML code, which can be read by the browser. Because the PHP code is transformed into HTML before the page is loaded, users cannot view the PHP code on a page. This make PHP pages secure enough to access databases and other secure information.

T5.2) Data Input procedures

Input & data entry design

To develop user-friendly interface and process for getting quality data into the information system in a timely and accurate fashion

Documenting Data Entry Procedures

Objectives: generally, most work done on improving the standards of data entry covers the entry of data for operational, financial or administrative applications. These are either

developed for the Department or acquired as Commercial products. This Activity presents some Good Practices for ensuring that all data entry procedures are properly documented.

Scope of usage: generally, this would apply to data entry for operational, administrative and financial application software. However, in some cases, there are data entry procedures that are covered by Office Technology products.

Risks: not having proper data entry procedures would lead to the following:

- Improper entry of data causing errors, rework and possibly damages with citizens and other users of the data.
- Erroneous or slow entry of data leading to loss of time and re-entry
- More rigorous training of new staff

Standard Operating Procedure:

1) For each data entry form in each software application, a detailed Data Entry Procedure is to be prepared. Generally, the software would have such procedures if it is well documented. In such a case, the Department should check if such procedures are sufficient, otherwise, they would be used as the basis for a redeveloped Data Entry Procedure.

   The Data Entry procedure would be used in the following cases:

   - ✓ Training
   - ✓ Day to day entry tasks
   - ✓ Validation and consistency checking
   - ✓ Auditing purposes
   - ✓ Refreshing the user

   A Sample Data Entry Procedure is presented in the **Appendix of Supplementary Material**.

2) On preparing a Data Entry Procedure, it should be tested to ensure that all data elements are clearly explained and that the procedure is easy to understand and use.

3) Whenever the Department is developing its own systems or requesting such systems to be developed by third parties, it is essential to follow some Data Entry standards or Good Practices. These are presented in the **Appendix of Supplementary Material**.

4) **Error analysis** should be carried out. When errors are reported, they should be registered and totaled. If specific forms are found to generate more errors than others, then there is the possibility that the form is not being properly understood or that it may have programming problems.

5) **Use Audit Trails** whenever possible. Audit Trails are rigorous ways of verifying proper data entry. Throughout the years, the term got distorted to mean the following:

a list of transactions as entered. This is not correct. Audit trails technically mean the following. A batch of vouchers is entered. One of its fields is manually accumulated, say the amount on the voucher. The batch is controlled as one single group of vouchers by the system. The batch or Audit Trail is entered into the system. The batch is not updated or posted to the database unless the computed total equals the entered total. Along with this checking, the system must be able to produce a validation list of all vouchers before committing the entry.

6) **Data Integrity:** various systems are designed without proper data integrity. For example, there are many Inventory Management systems that allow the quantity to go negative. There are cases where the accounting vouchers posted from an operational module do not correspond to the totals recognized by the accounting system. For example, purchases in the Inventory Management system do not correspond to the total inventory in the Accounting System. Such areas should be investigated and checked on a regular basis to ensure that there is Data Integrity.

Input design

Objectives	**Guidelines**
• Effective • Accurate • Ease to use • Consistent • Attractive • Simple	• Input forms should be easy to fill out, purposeful, facilitate accurate completion, and attractive • Screens should be simple, consistent, facilitate navigation, and attractive • Web-forms should use a variety of input methods, provide clear instructions, demonstrate a logical entry sequence, provide a feedback screen, separate a lengthy form into simpler forms on separate pages

Web-form input methods

1. Text-box
2. Check-box
3. Option button
4. Drop-down list
5. Sliders
6. Image maps
7. Message box
8. Command button
9. Tab control dialog box

Data entry design

Objectives	Guidelines
• Effective coding • Efficient data capture • Effective data capture • Effective input validation	Reduce data input volume • Keep codes concise, stable, unique, sortable, simple, uniform, modifiable, and meaningful (Figure 19.11) • Input necessary data only • Do not input data that can be computed/retrieved from the system • Provide default values when appropriate • Allow look up of value Reduce data input error • Validate input transactions to avoid submitting wrong or unauthorized data and to prevent unauthorized action • Validate input data to prevent missing data, incorrect field length, type, range, value, cross-reference, un-match data.

Types of codes

Purpose	Codes	Example
Tracking information	• Sequence codes	• Order number at Atlanta Bread
	• Alphabetic derivation codes	• U-connect account name
Classifying information	• Classification codes	• Letter grade
	• Block sequence codes	• IP address
Concealing information	• Cipher codes	• Morse code
Revealing information	• Significant-digit subset codes	• Course number system at UK
	• Mnemonic codes	• Acronyms
Requesting action	• Function codes	• Menu choices

Date-entry methods

1. Keyboards
2. Optical character recognition
3. Magnetic ink character recognition
4. Mark-sense forms
5. Bar codes
6. Intelligent terminals

Validation tests

1. Data type/class test – data are of proper type

2. Combination test – data from two or more fields are consistent or reasonable when considered together

3. Expectancy test – data are anticipated, e.g., in some predetermined sequence

4. Existence test – data must be input, e.g., a required field

5. Range test – data are within proper range of values

6. Domain test – data are of proper situation or come from set of standard values

7. Size test – data are of proper length

8. Validity test – data must have certain values, e.g., referential integrity

9. Batch control test – use record count/hash totals to verify batch input

10. Check digit test – add an extra digit to a numeric field to verify its accuracy (Figure 19.19)

User interface design

Objectives	Guidelines
• Task matching • Efficient • Feedback provision • Productivity improvement	Minimize user frustration • Communicate • Minimal user action • Standard operation and consistency • Context-sensitive help Provide feedback on • Acknowledgement • Error/warning • Status • Reassurance • Availability of further assistance

Types of user interface

1. Natural language, e.g., www.askjeeves.com (search engine)
2. Question-and-answer, e.g., dialog box
3. Menu, e.g., Menu bar in Access
4. Form-fill, e.g., registration form as www.yahoo.com members
5. Command-language, e.g., DOS operating systems
6. Graphical, e.g., Icon bar in Access
7. Touch-screen, e.g., ATM machine

Database design

Objectives	Guidelines
• Purposeful information retrieval • Efficient data storage • Data availability • Efficient data update and retrieval • Data integrity	• Use ER diagram for data modeling • All tables are normalized

Data integrity

- Entity integrity: the primary key of a table cannot have a null value
- Referential integrity: all foreign keys in a child table must have a matching record in the parent table
- Domain integrity: table entries must be of the same type, limit, range and other validation checks.

Basic constructs of the E-R model

Concept	Definition	Examples
Entity	A person, place, object, event or concept	Employee, department, building, sale, account
Relationship	An entity that serves to interconnect two or more entity types	Assignment (Employee-Department)
Attribute	A property or characteristic of an entity/relationship type	Employee_name, department_location, sale_date

Types of relationships

1. one-to-one, e.g, STUDENT-Assign-PARKING
2. one-to-many, e.g., DEPARTMENT-Offer-COURSE
3. many-to-many, e.g., STUDENT-register-COURSE

E-R models → relational models

Elements	Relational model
Entity	A table
Attribute	A column of the table
One to one relationship	• One table is created for each entity • The key of either one of the tables is placed as the

	foreign key in the other table
One to many relationship	• One table is created for each entity • The key of the table on the "one" side of therelationship (parent) is placed as the foreign key in the table representing the "many" side of the relationship (child)
Many to many relationship	• One table is created for each entity • One table is created for the relationship itself

T5.3) Data output procedures

Output design

To deliver the right information to the right people in the right format at the right time

Output design

Objectives	Guidelines
• Purposeful • Meaningful • Adequate • Appropriate distribution • Timely • Appropriate medium	• Consider usage factors before choosing an output technology/medium • Avoid output bias • Consider functional and stylistic attributes in designing printed reports • Keep screen output simple, consistent, easy to navigate, and attractive

Types of output technology/medium

1. Print
2. Screen
3. Audio
4. Microform
5. CD-ROM or DVD
6. Electronic
7. Web-based documents

Usage factors in making output technology decisions

1. Users
2. Number of users
3. Physical destination
4. Purpose
5. Speed
6. Frequency
7. Longevity
8. Legal requirements
9. Costs
10. Environmental requirements

Sources of output bias

1. Sorting
2. Setting limits
3. Misleading graphics

T5.4) Implement Java Script and ASP

Implementing JavaScript

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript - Syntax

JavaScript can be implemented using JavaScript statements that are placed within the **<script>... </script>** HTML tags in a web page.

You can place the **<script>** tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the **<head>** tags.

The <script> tag alerts the browser program to start interpreting all the text between these tags as a script. A simple syntax of your JavaScript will appear as follows.

```
<script ...>
   JavaScript code
</script>
```

The script tag takes two important attributes −

- **Language** − This attribute specifies what scripting language you are using. Typically, its value will be javascript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.
- **Type** − This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javascript".

So your JavaScript segment will look like −

```
<script language="javascript" type="text/javascript">
   JavaScript code
</script>
```

Your First JavaScript Script

Let us take a sample example to print out "Hello World". We added an optional HTML comment that surrounds our JavaScript code. This is to save our code from a browser that does not support JavaScript. The comment ends with a "//-->". Here "//" signifies a comment in JavaScript, so we add that to prevent a browser from reading the end of the HTML comment as a piece of JavaScript code. Next, we call a function **document.write** which writes a string into our HTML document.

This function can be used to write text, HTML, or both. Take a look at the following code.

```
<html>
   <body>
      <script language="javascript" type="text/javascript">
         <!--
            document.write("Hello World!")
         //-->
      </script>
   </body>
</html>
```

This code will produce the following result −

```
Hello World!
```

Whitespace and Line Breaks: JavaScript ignores spaces, tabs, and newlines that appear in JavaScript programs. You can use spaces, tabs, and newlines freely in your program and you are free to format and indent your programs in a neat and consistent way that makes the code easy to read and understand.

Semicolons are Optional: Simple statements in JavaScript are generally followed by a semicolon character, just as they are in C, C++, and Java. JavaScript, however, allows you to omit this semicolon if each of your statements are placed on a separate line. For example, the following code could be written without semicolons.

```
<script language="javascript" type="text/javascript">
   <!--
      var1 = 10
      var2 = 20
   //-->
</script>
```

But when formatted in a single line as follows, you must use semicolons −

```
<script language="javascript" type="text/javascript">
   <!--
      var1 = 10; var2 = 20;
   //-->
</script>
```

Note − It is a good programming practice to use semicolons.

Case Sensitivity: JavaScript is a case-sensitive language. This means that the language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.

So the identifiers **Time** and **TIME** will convey different meanings in JavaScript.

NOTE − Care should be taken while writing variable and function names in JavaScript.

Comments in JavaScript: JavaScript supports both C-style and C++-style comments, Thus −

- Any text between a // and the end of a line is treated as a comment and is ignored by JavaScript.
- Any text between the characters /* and */ is treated as a comment. This may span multiple lines.
- JavaScript also recognizes the HTML comment opening sequence <!--. JavaScript treats this as a single-line comment, just as it does the // comment.
- The HTML comment closing sequence --> is not recognized by JavaScript so it should be written as //-->.

Example

The following example shows how to use comments in JavaScript.

```
<script language="javascript" type="text/javascript">
   <!--

      // This is a comment. It is similar to comments in C++

      /*
      * This is a multiline comment in JavaScript
      * It is very similar to comments in C Programming
      */

   //-->
</script>
```

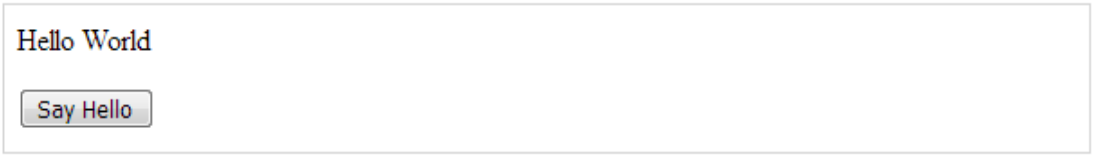
JavaScript in <body> and <head> Sections

You can put your JavaScript code in <head> and <body> section altogether as follows −

```
<html>
   <head>
      <script type="text/javascript">
         <!--
            function sayHello() {
               alert("Hello World")
            }
         //-->
      </script>
   </head>
```

```
   <body>
      <script type="text/javascript">
         <!--
            document.write("Hello World")
         //-->
      </script>

      <input type="button" onclick="sayHello()" value="Say Hello" />

   </body>
</html>
```

This code will produce the following result −



JavaScript in External File

As you begin to work more extensively with JavaScript, you will be likely to find that there are cases where you are reusing identical JavaScript code on multiple pages of a site.

You are not restricted to be maintaining identical code in multiple HTML files. The **script** tag provides a mechanism to allow you to store JavaScript in an external file and then include it into your HTML files.

Here is an example to show how you can include an external JavaScript file in your HTML code using **script** tag and its **src** attribute.

```
<html>

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

   <body>
      .......
   </body>
</html>
```

To use JavaScript from an external file source, you need to write all your JavaScript source code in a simple text file with the extension ".js" and then include that file as shown above.

For example, you can keep the following content in **filename.js** file and then you can use **sayHello** function in your HTML file after including the filename.js file.

```
function sayHello() {
   alert("Hello World")
}
```

JavaScript Datatypes

One of the most fundamental characteristics of a programming language is the set of data types it supports. These are the type of values that can be represented and manipulated in a programming language.

JavaScript allows you to work with three primitive data types −

- **Numbers,** eg. 123, 120.50 etc.
- **Strings** of text e.g. "This text string" etc.
- **Boolean** e.g. true or false.

JavaScript also defines two trivial data types, **null** and **undefined,** each of which defines only a single value. In addition to these primitive data types, JavaScript supports a composite data type known as **object**. We will cover objects in detail in a separate chapter.

Note − Java does not make a distinction between integer values and floating-point values. All numbers in JavaScript are represented as floating-point values. JavaScript represents numbers using the 64-bit floating-point format defined by the IEEE 754 standard.

JavaScript Variables

Like many other programming languages, JavaScript has variables. Variables can be thought of as named containers. You can place data into these containers and then refer to the data simply by naming the container.

Before you use a variable in a JavaScript program, you must declare it. Variables are declared with the **var** keyword as follows.

```
<script type="text/javascript">
   <!--
      var money;
      var name;
   //-->
</script>
```

You can also declare multiple variables with the same **var** keyword as follows −

```
<script type="text/javascript">
   <!--
      var money, name;
   //-->
</script>
```

Storing a value in a variable is called **variable initialization**. You can do variable initialization at the time of variable creation or at a later point in time when you need that variable.

For instance, you might create a variable named **money** and assign the value 2000.50 to it later. For another variable, you can assign a value at the time of initialization as follows.

```
<script type="text/javascript">
   <!--
      var name = "Ali";
      var money;
      money = 2000.50;
   //-->
</script>
```

Note − Use the **var** keyword only for declaration or initialization, once for the life of any variable name in a document. You should not re-declare same variable twice.

JavaScript is **untyped** language. This means that a JavaScript variable can hold a value of any data type. Unlike many other languages, you don't have to tell JavaScript during variable declaration what type of value the variable will hold. The value type of a variable can change during the execution of a program and JavaScript takes care of it automatically.

Implementing ASP

ASP stands for **A**ctive **S**erver **P**ages

ASP is a development framework for building web pages.

ASP supports many different development models:

- Classic ASP
- ASP.NET Web Forms
- ASP.NET MVC
- ASP.NET Web Pages
- ASP.NET API
- ASP.NET Core

The ASP Technology

ASP and ASP.NET are server side technologies.

Both technologies enable computer code to be executed by an Internet server.

When a browser requests an ASP or ASP.NET file, the ASP engine reads the file, executes any code in the file, and returns the result to the browser.

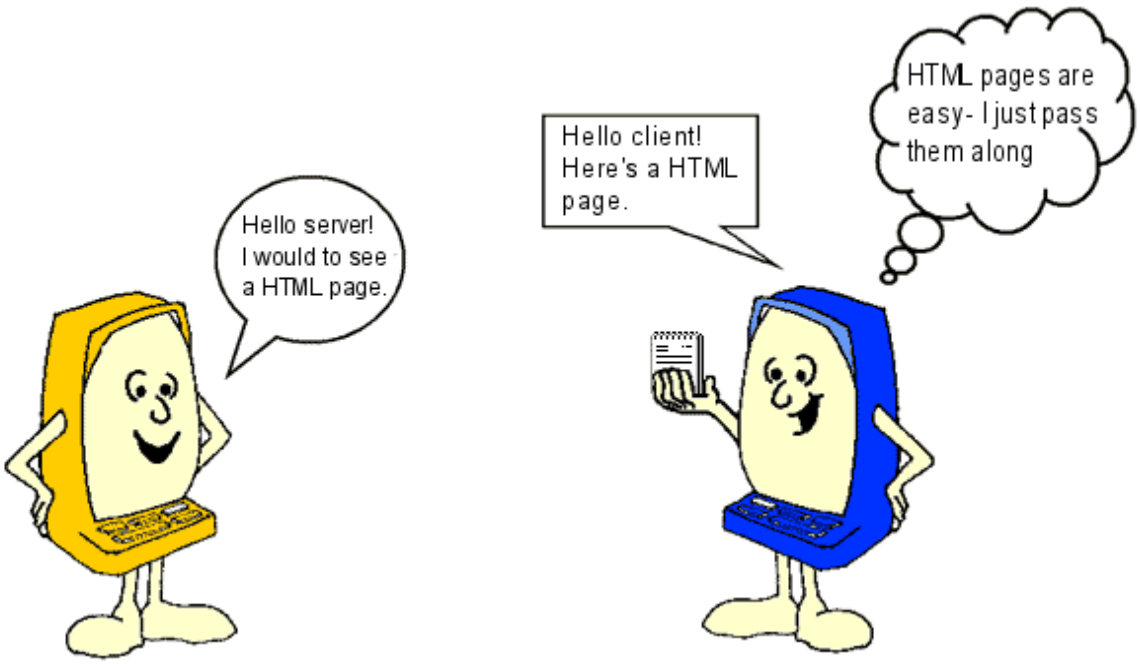
An Active Server Page (ASP) is an HTML page that includes one or more scripts (small embedded programs) that are processed on a Microsoft Web server before the page is sent to the user. An ASP is somewhat similar to a server-side include or a common gateway interface (CGI) application in that all involve programs that run on the server, usually tailoring a page for the user. Typically, the script in the Web page at the server uses input received as the result of the user's request for the page to access data from a database and then builds or customizes the page on the fly before sending it to the requestor.

ASP is a feature of the Microsoft Internet Information Server (IIS), but, since the server-side script is just building a regular HTML page, it can be delivered to almost any browser. You can create an ASP file by including a script written in VBScript or JScript in an HTML file or by using ActiveX Data Objects (ADOs) program statements in the HTML file. You name the HTML file with the ".asp" file suffix. Microsoft recommends the use of the server-side ASP rather than a client-side script, where there is actually a choice, because the server-side script will result in an easily displayable HTML page. Client-side scripts (for example, with JavaScript) may not work as intended on older browsers.

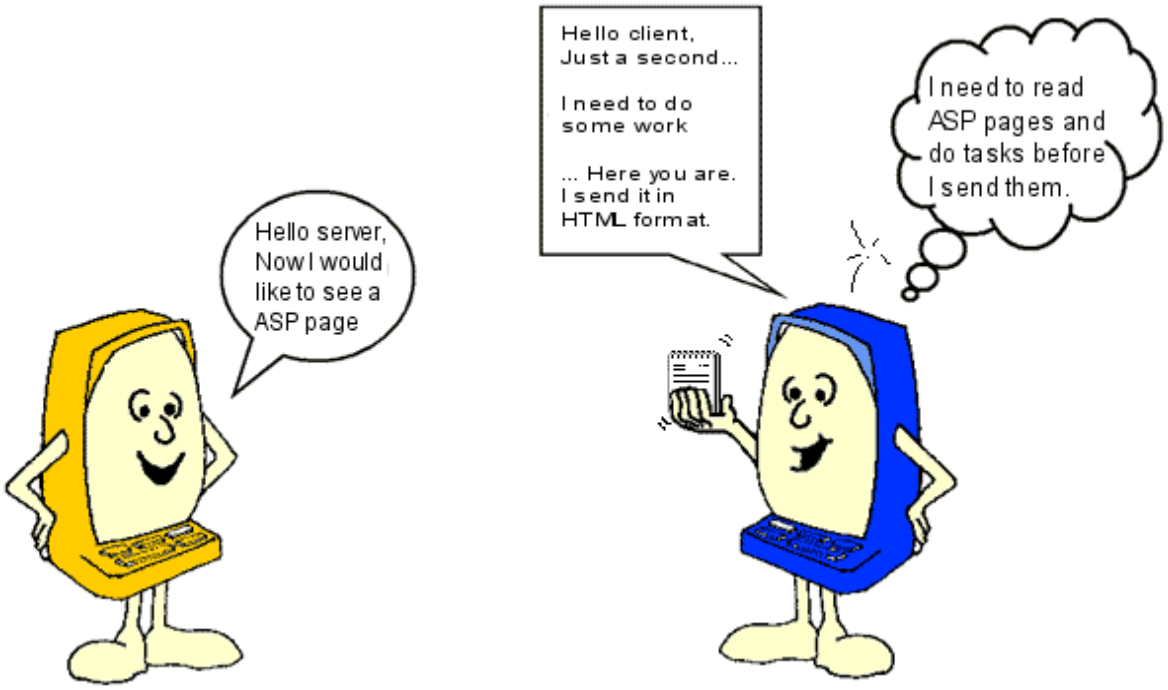
Important!
ASP is no longer being further developed by Microsoft - it has been replaced by ASP.NET. But ASP is still a fully functional scripting language and you may choose to continue to learn and use ASP (i.e., if your server only supports ASP, you need to maintain an ASP website, etc.). If it's not necessary for you to use ASP, we recommend you learn PHP instead:

How does ASP work?

The best way to explain how ASP works is by comparing it with standard HTML. Imagine you type the address of an HTML document (eg. **http://www.mysite.com/page.htm**) in the address line of the browser. This way you **request** an HTML page. It could be illustrated like this:



As you can see, the server simply sends an HTML file to the client. But if you instead type **http://www.mysite.com/page.asp** - and thus request an **ASP page** - the server is put to work:



The server first reads the ASP file carefully to see if there are any tasks that need to be executed. Only when the server has done what it is supposed to do, the result is then sent to the client. It is important to understand that the client only sees the **result** of the server's work - not the actual instructions.

This means that if you click "view source" on an ASP page, you do not see the ASP codes - only basic HTML tags. Therefore, you can not see how an ASP page is made by using "view source". You have to learn ASP in other ways, for example, by reading this tutorial.

What you learn in this tutorial is to write commands to a server!
So, the first thing you need to get hold of is... a server! But don't worry - you don't need to buy a new computer. You just need to install some software on your computer that makes it function as a server. Another option is to have a website on a hosted server that supports ASP. Then you just need to be online while coding.

The next lesson is about how to get your computer to act as a server.

Servers

ASP is a *server-side* technology. Therefore, you need to have a server to run ASP. But it doesn't need to cost you anything to make this upgrade and there are several options for doing so.

Since you ultimately only need to choose one option this lesson is divided into different parts. First comes a little introduction on the different options (just choose the one that suits you best). When your server is up and running, we'll pick up with Lesson 3 to make your first ASP page.

If you use Windows 7: Windows 7 includes Internet Information Server 7.5 (IIS) and can therefore run ASP.

If you use Windows XP Professional: Windows XP Professional includes Internet Information Server 5.1 (IIS) and can therefore run ASP.

If you use Windows XP Home Edition: Internet Information Server (IIS) is not included in this edition of Windows XP. However, according to this Usenet post it is possible to install IIS on Windows XP Home Edition. You may want to try this approach. But we recommend you use Windows 7 or Windows XP Professional instead.

If you use Linux: It is possible to run ASP on Linux, but we recommend that you use PHP on Linux instead. With PHP you can do pretty much the same as ASP.

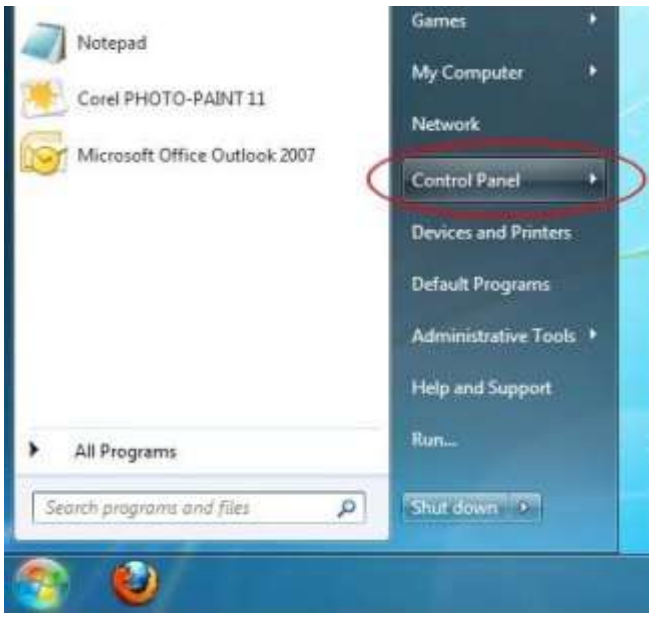
If you use Macintosch: We know of no option for running ASP on a Macintosh.

If you have a website on a hosted server: You can choose to have a website on hosted server which run ASP (like AspHost4Free (free host)), then you are ready to go and can just ftp your ASP pages to the server.

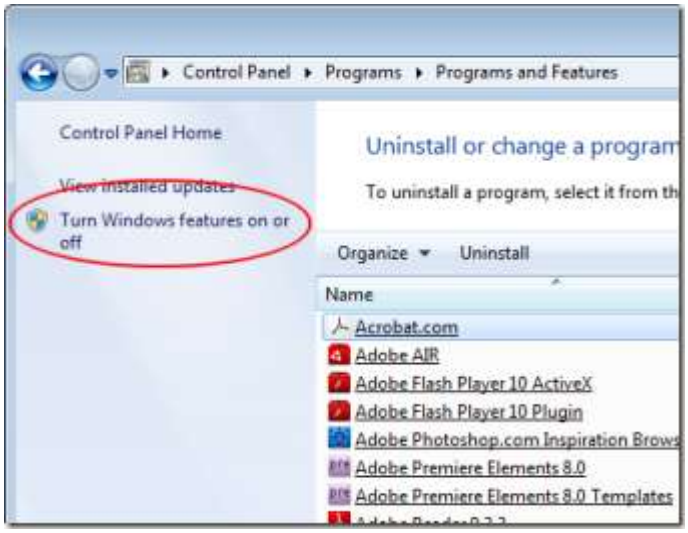
a) Internet Information Server (IIS) on Windows 7

By default, IIS is not installed on Windows 7. But you can easily install and configure IIS by following the instructions below:

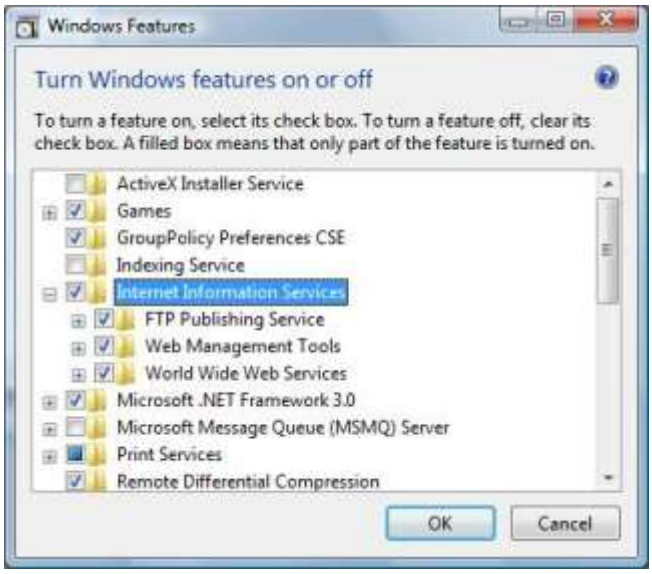
1. Click Start and then click Control Panel:



2. In Control Panel, click Programs and then click Turn Windows features on or off:



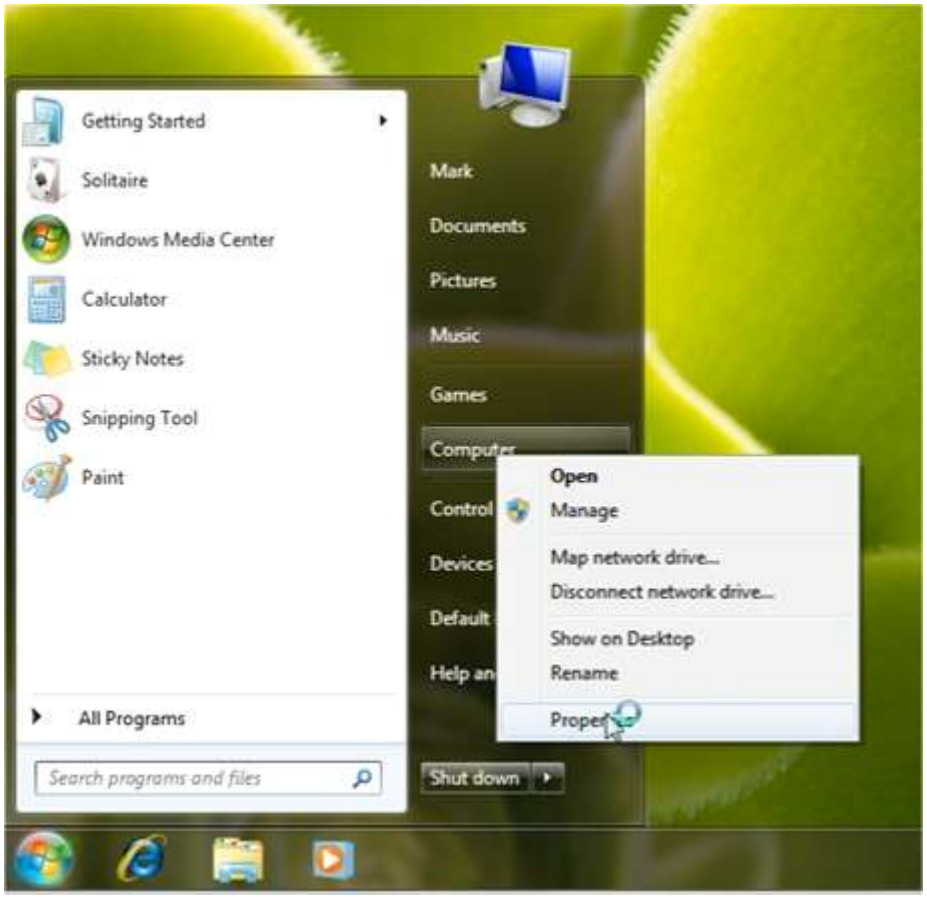
3. In the Windows Features dialog box, click Internet Information Services and then click OK:



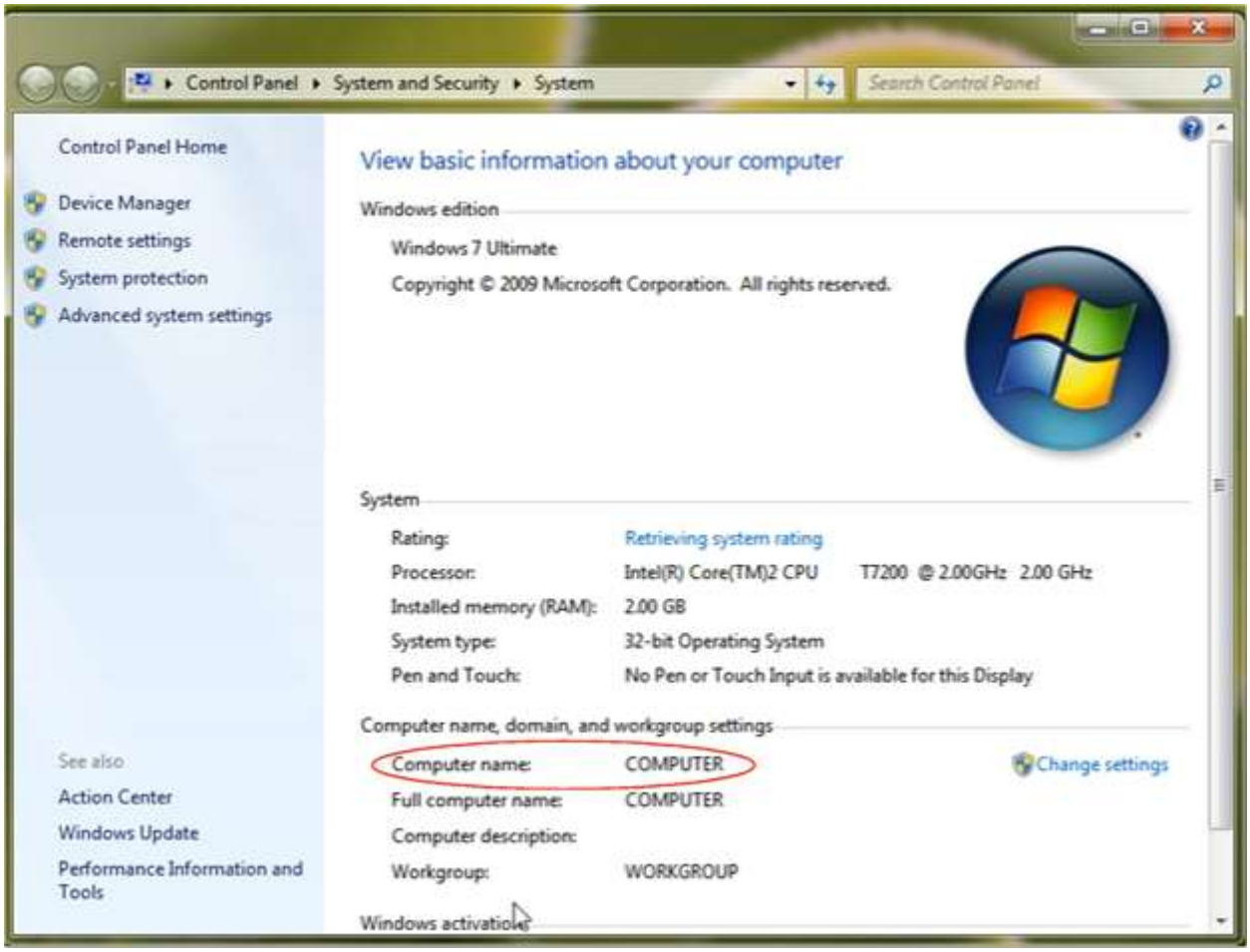
Congratulations! Your computer is now a web server!

Browsing your server

Next, you need to find your computer name. You'll find it by clicking the Start button, right-clicking Computer, and then clicking Properties.



Under Computer name, domain, and workgroup settings, you can find your computer name. I have here stated my computer's name in red - it's called COMPUTER (but yours are probably called something else).



Now you simply type the computer name in the address bar of your browser...

... and when you click return you will see a page like the one pictured below. It means that you now have contact to your server. Now we just need to look at where you need to save your ASP pages.

How to work with ASP and HTML files on the server

The last thing you need to know before you are ready to create your first ASP pages is where to save them.

The root of your website is c:\inetpub\wwwroot. This means that the file c:\inetpub\wwwroot\default.asp can be accessed in your browser at the address http://COMPUTER/default.asp (replace "COMPUTER" with the name of your computer (look above)).

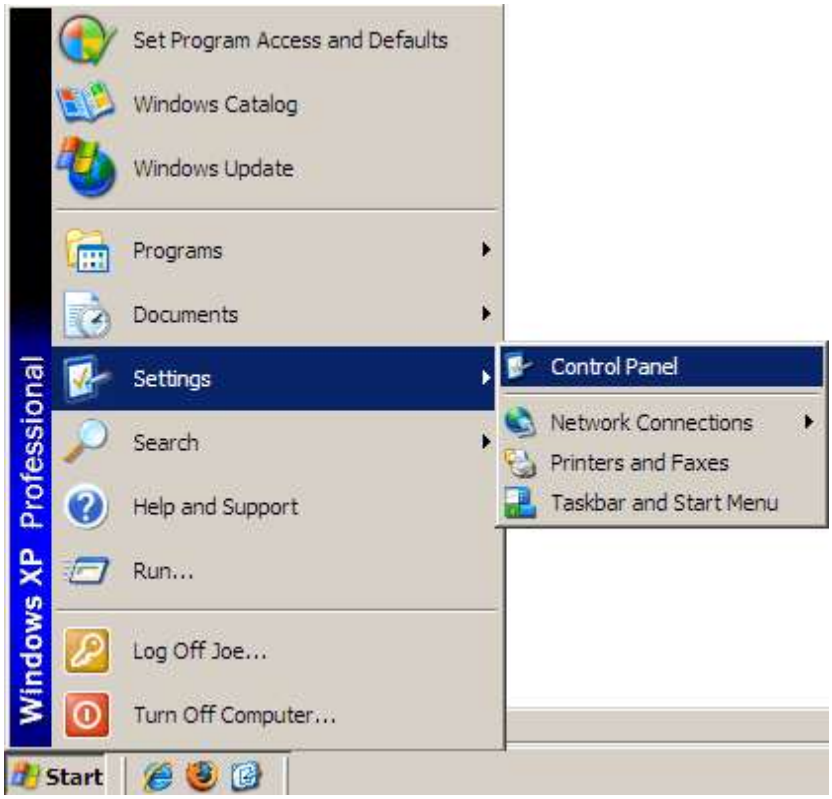
And in the same way, the folder c:\inetpub\wwwroot\images will be available in your browser at http://COMPUTER/images.

That's it! Now are you ready to code your first ASP page. Hurry on to Lesson 3!

b) Internet Information Server (IIS) on Windows XP Professional

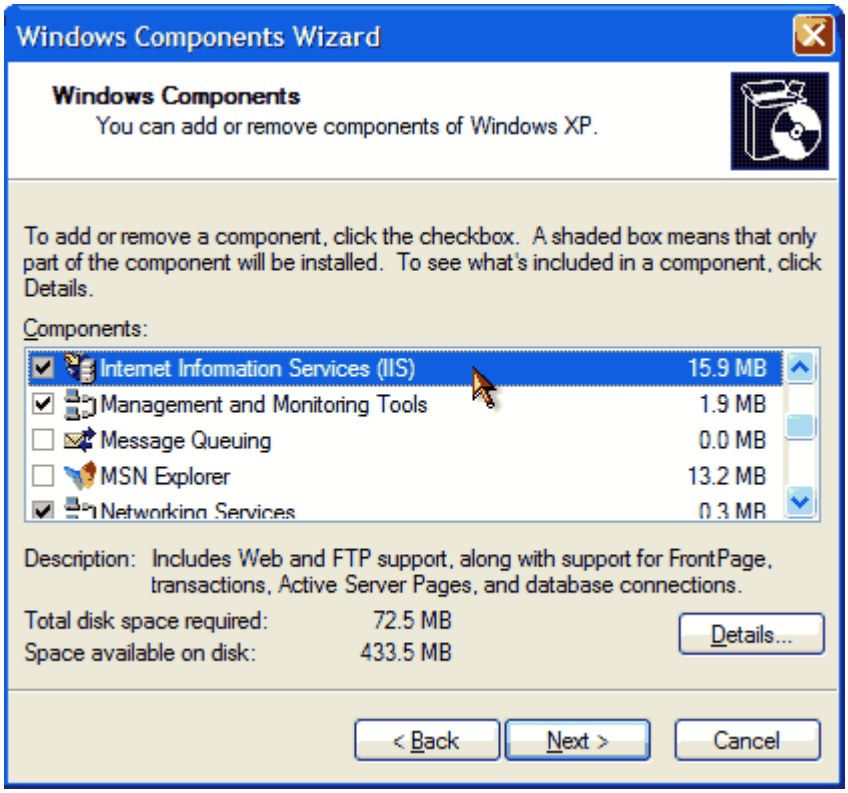
By default, IIS is not installed on Windows XP Professional. But you can easily install and configure IIS by following the instructions below:

1. Click Start, click Control Panel, and click Add or Remove Programs:



2. Click Add/Remove Windows Components. The Windows Components Wizard appears:

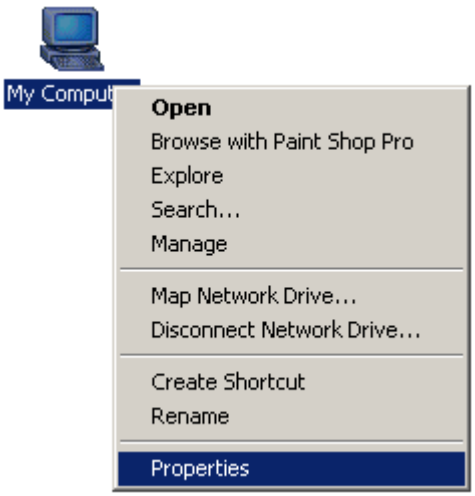
3. Follow the on-screen instructions to install, remove, or add components to IIS:



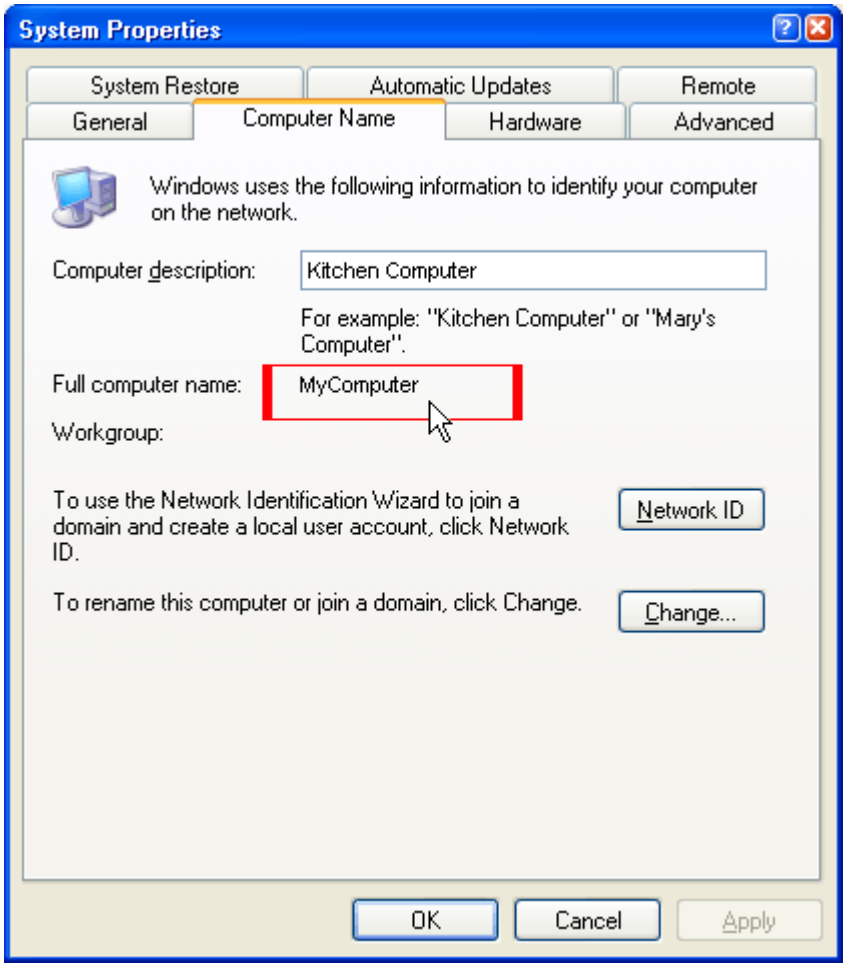
Congratulations! Your computer is now a web server!

Browsing your server

Next, you need to find your computer name. You'll find it by right clicking "My Computer" on your desktop and select "properties":



In the dialog box, you find your computer name under the tab "Network Identification". I have here stated my computer's name in red - it's called MyComputer (but yours are probably called something else).



Now you simply type the computer name in the address bar of your browser...

... and when you click return you will see a page like the one pictured below. It means that you now have contact to your server. Now we just need to look at where you need to save your ASP pages.

How to work with ASP and HTML files on the server

The last thing you need to know before you are ready to create your first ASP pages is where to save them.

The root of your website is c:\inetpub\wwwroot. This means that the file c:\inetpub\wwwroot\default.asp can be accessed in your browser at the address http://MyComputer/default.asp (replace "MyComputer" with the name of your computer (look above)).

And in the same way, the folder c:\inetpub\wwwroot\images will be available in your browser at http://MyComputer/images.

That's it! Now are you ready to code your first ASP page. Hurry on to Lesson 3!

Your first ASP page

You now know a little about what ASP is, and you've installed (or have access to) a server. Now we are ready to begin making our first ASP page. We keep it simple and easy, but after you have gone through this lesson, you will understand much more about what ASP is and what you can do with it.

Basically, an ASP file is a text file with the extension **.asp** which consists of:

- Text
- HTML tags
- ASP Scripts

You already know what text and HTML tags are. So let's look a little more at ASP scripts.

ASP Scripts

ASP scripts can be written in different languages. The examples in this tutorial are written in Microsoft Visual Basic Scripting Edition (VBScript), but could also be written in another language - eg. JScript.

Let's get started with your first ASP page.

Example: Hello World!

Start by making an ordinary HTML document. But name the file *default.asp* and place it in the root of the site. On your computer (which is now a server), the path is c:\inetpub\wwwroot\default.asp.

The HTML code should look like this:

```
<html>
<head>
<title>My first ASP page</title>

</head>
<body>

</body>
</html>
```

As you probably remember from lesson 1, ASP is all about **writing commands to a server.** So let's write a command to the server.

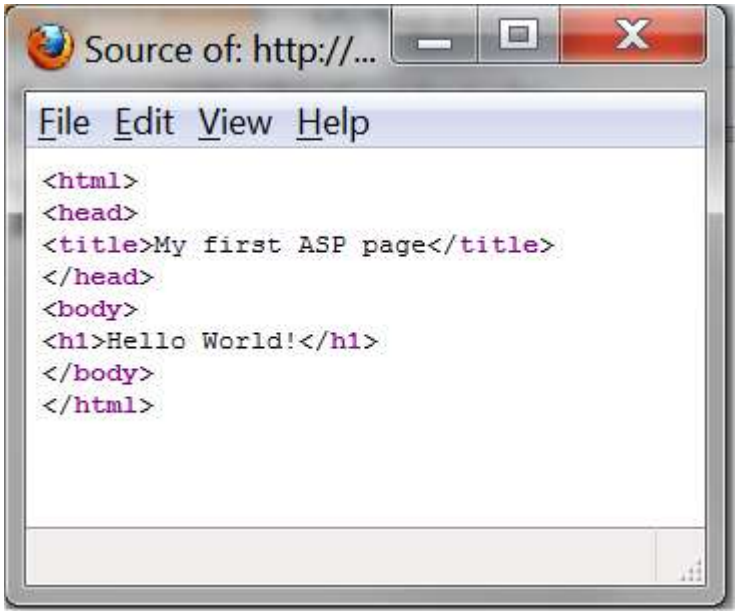
First, we need to tell the server when the ASP will **start** and **end.** In ASP you use the tags **<%** and **%>** to mark the start and end for the ASP codes that the server must execute.

Now try to add the following simple code snippet to your HTML code:

```
<html>
<head>
<title>My first ASP page</title>
</head>
<body>
<%
Response.Write "<h1>Hello World!</h1>"
%>
</body>
</html>
```

When we look at the ASP document in a browser, it should look like this:

But it gets interesting when you look at the HTML code in the browser (by selecting "view source"):



The ASP codes are gone! As you may remember from lesson 1, it is only the server that can see the ASP codes - **the client (the browser) only sees the result!**

Let's look at what happened. We asked the server to write <h1> Hello World!</h1>. In a more technical language, one would say that we used the object Response and the method Write to write a specified string to the client. But do not worry! In this tutorial we try to keep the technical language at a minimum.

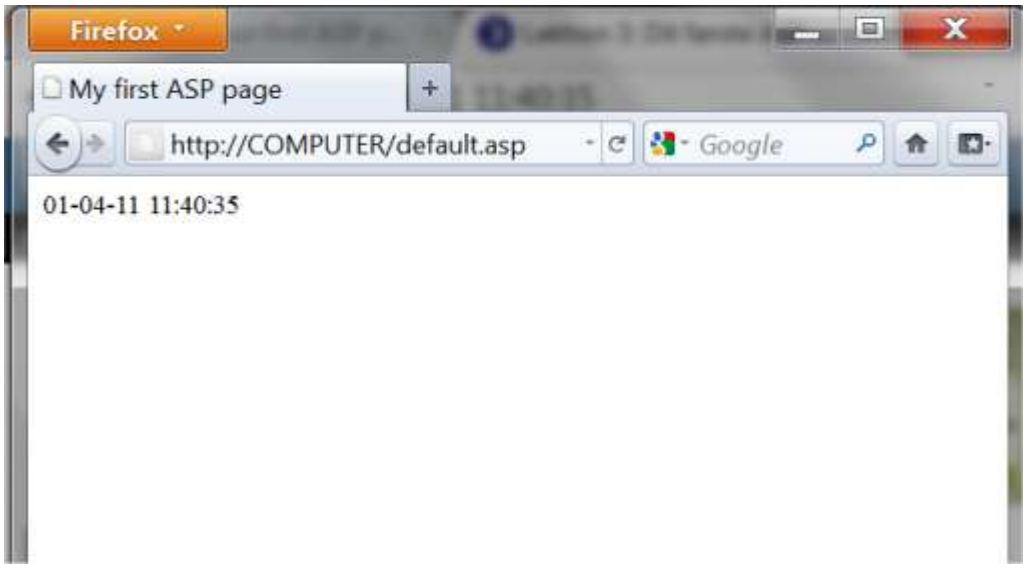
Our first example is obviously not particularly exciting. But just wait! From now on, it's only going to be more and more interesting. Let's look at another example.

Example: Now!

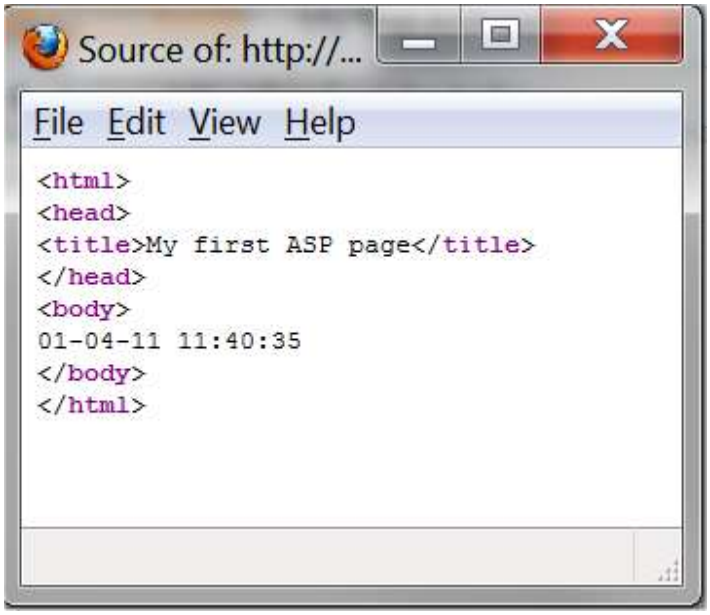
Let's make the server write something else. We could, for example, ask it to write the current date and time:

```
<html>
<head>
<title>My first ASP page</title>

</head>
<body>

<%
Response.Write Now
%>

</body>
</html>
```

Will look like this in the browser:



And the corresponding HTML code:



Now things are getting interesting, right?

We make the server write the date and time when the ASP page is displayed. Note that if you refresh the page in the browser, a new time is written. The server writes the current date and time each time the page is sent to a client.

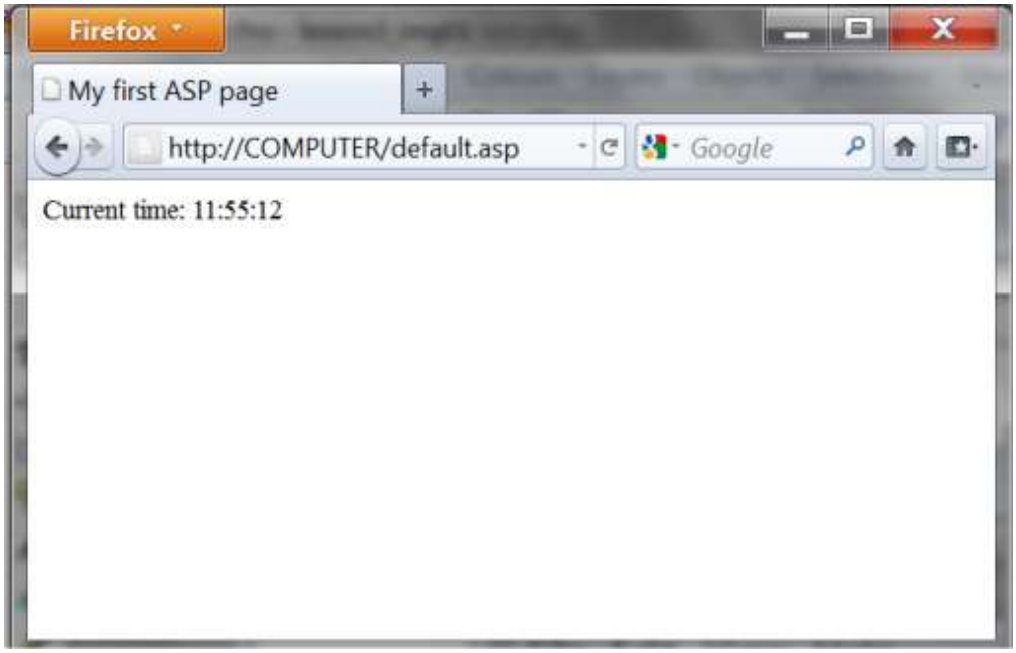
It is also important to note that the HTML code contains only the date - not the ASP codes. Therefore, the example is not affected by which browser is used. Actually, all functionalities that are made with *server-side* technologies always **work in all browsers!**

In the example, we used Now - which is a function that returns the current date and time on the server.

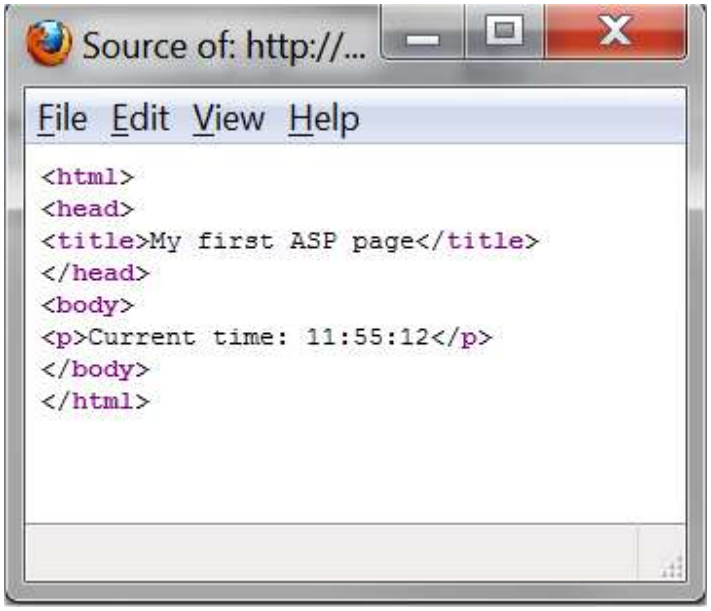
Let's try to extend the example by writing both a *string* and a *function* - separated by **&** - it's done like this:

```
<html>
<head>
<title>My first ASP document</title>
</head>
<body>

<%
Response.Write "<p>Current time: " & Time & "</p>"
%>

</body>
</html>
```

Will look like this in the browser:



And the corresponding HTML code:



As you can see, the function Time returns the current time. There are several functions that relate to time and date. And that's exactly what we'll look at in the next lesson.

TOPIC 6: WEB SECURITY

T6.1) Explaining web security

Definition of Web Security

1. Web application security is the process of securing confidential data stored online from unauthorized access and modification. This is accomplished by enforcing stringent policy measures. Security threats can compromise the data stored by an organization is hackers with malicious intentions try to gain access to sensitive information.
2. a set of procedures, practices, and technologies for assuring the reliable, predictable operation of web servers, web browsers, other programs that communicate with web servers, and the surrounding Internet infrastructure.

The aim of Web application security is to identify the following:

- Critical assets of the organization
- Genuine users who may access the data
- Level of access provided to each user
- Various vulnerabilities that may exist in the application
- Data criticality and risk analysis on data exposure
- Appropriate remediation measures

Web application security aims to address and fulfill the four conditions of security, also referred to as principles of security:

- Confidentiality: States that the sensitive data stored in the Web application should not be exposed under any circumstances.
- Integrity: States that the data contained in the Web application is consistent and is not modified by an unauthorized user.
- Availability: States that the Web application should be accessible to the genuine user within a specified period of time depending on the request.
- Nonrepudiation: States that the genuine user cannot deny modifying the data contained in the Web application and that the Web application can prove its identity to the genuine user.

The process of security analysis runs parallel with Web application development. The group of programmers and developers who are responsible for code development are also responsible for the execution of various strategies, post-risk analysis, mitigation and monitoring.

T6.2) Identifying web security issues

What are the different types of **website security issues, risks or threats**, and what can make your business and website an attractive or susceptible target? Many small businesses feel they do not represent a worthwhile target to attackers, but as you will read, this assumption is plain wrong. All online entities face a variety of security risks and threats that should be understood and assessed.

For all too many companies, it's not until *after* a breach has occurred that web security becomes a priority. An effective approach to IT security must, by definition, be proactive and defensive. Toward that end, this discussion is aimed at sparking a security mindset, hopefully injecting the reader with a healthy dose of paranoia.

In particular, this guide focuses on common and significant web security pitfalls to be aware of, including recommendations on how they can be avoided.

A little web security primer before we start – authentication and authorization

There usually is a confusion regarding the distinction between authorization and authentication. And of course, the fact the abbreviation *auth* is often used for both helps aggravate this common confusion. This confusion is so common that maybe this issue should be included in this post as "Common Web Vulnerability Zero".

So before we proceed, let's clearly the distinction between these two terms:

- **Authentication:** Verifying that a person is (or at least appears to be) a specific user, since he/she has correctly provided their security credentials (password, answers to security questions, fingerprint scan, etc.).
- **Authorization:** Confirming that a particular user has access to a specific resource or is granted permission to perform a particular action.

Stated another way, *authentication* is knowing who an entity is, while *authorization* is knowing what a given entity can do.

1. Common Mistake #1: Injection flaws

Injection flaws result from a classic failure to filter untrusted input. It can happen when you pass unfiltered data to the SQL server (SQL injection), to the browser (XSS – we'll talk about this later), to the LDAP server (LDAP injection), or anywhere else. The problem here is that the attacker can inject commands to these entities, resulting in loss of data and hijacking clients' browsers.

Anything that your application receives from untrusted sources must be filtered, preferably according to a whitelist. You should almost never use a blacklist, as getting that right is very hard and usually easy to bypass. Antivirus software products typically provide stellar examples of failing blacklists. Pattern matching does not work.

Prevention: The good news is that protecting against injection is "simply" a matter of filtering your input properly and thinking about whether an input can be trusted. But the bad

news is that *all* input needs to be properly filtered, unless it can unquestionably be trusted (but the saying "never say never" does come to mind here).

In a system with 1,000 inputs, for example, successfully filtering 999 of them is not sufficient, as this still leaves one field that can serve as the Achilles heal to bring down your system. And you might think that putting an SQL query result into another query is a good idea, as the database is trusted, but if the perimeter is not, the input comes indirectly from guys with malintent. This is called Second Order SQL Injection in case you're interested.

Since filtering is pretty hard to do right (like crypto), what I usually advise is to rely on your framework's filtering functions: they are proven to work and are thoroughly scrutinized. If you do not use frameworks, you really need to think hard about whether not using them really makes sense in your environment. 99% of the time it does not.

2. Common Mistake #2: Broken Authentication

This is a collection of multiple problems that might occur during broken authentication, but they don't all stem from the same root cause.

Assuming that anyone still wants to roll their own authentication code in 2014 (what are you thinking??), I advise against it. It is extremely hard to get right, and there are a myriad of possible pitfalls, just to mention a few:

1. The URL might contain the session id and leak it in the referer header to someone else.
2. The passwords might not be encrypted either in storage or transit.
3. The session ids might be predictable, thus gaining access is trivial.
4. Session fixation might be possible.
5. Session hijacking might be possible, timeouts not implemented right or using HTTP (no SSL), etc…

Prevention: The most straightforward way to avoid this web security vulnerability is to use a framework. You might be able to implement this correctly, but the former is much easier. In case you do want to roll your own code, be extremely paranoid and educate yourself on what the pitfalls are. There are quite a few.

3. Common Mistake #3: Cross Site Scripting (XSS)

This is a fairly widespread input sanitization failure (essentially a special case of common mistake #1). An attacker gives your web application JavaScript tags on input. When this input is returned to the user unsanitized, the user's browser will execute it. It can be as simple as crafting a link and persuading a user to click it, or it can be something much more sinister. On page load the script runs and, for example, can be used to post your cookies to the attacker.

Prevention: There's a simple web security solution: don't return HTML tags to the client. This has the added benefit of defending against HTML injection, a similar attack whereby the attacker injects plain HTML content (such as images or loud invisible flash players) – not high-impact but surely annoying ("please make it stop!"). Usually, the workaround is simply converting all HTML entities, so that `<script>` is returned as `&lt;script&gt;`. The other

often employed method of sanitization is using regular expressions to strip away HTML tags using regular expressions on < and >, but this is dangerous as a lot of browsers will interpret severely broken HTML just fine. Better to convert all characters to their escaped counterparts.

4. **Common Mistake #4: Insecure Direct Object References**

This is a classic case of trusting user input and paying the price in a resulting security vulnerability. A direct object reference means that an internal object such as a file or database key is exposed to the user. The problem with this is that the attacker can provide this reference and, if authorization is either not enforced (or is broken), the attacker can access or do things that they should be precluded from.

For example, the code has a `download.php` module that reads and lets the user download files, using a CGI parameter to specify the file name (e.g., `download.php?file=something.txt`). Either by mistake or due to laziness, the developer omitted authorization from the code. The attacker can now use this to download any system files that the user running PHP has access to, like the application code itself or other data left lying around on the server, like backups. Uh-oh.

Another common vulnerability example is a password reset function that relies on user input to determine whose password we're resetting. After clicking the valid URL, an attacker can just modify the `username` field in the URL to say something like "admin".

Incidentally, both of these examples are things I myself have seen appearing often "in the wild".

Prevention: Perform user authorization properly and consistently, and whitelist the choices. More often than not though, the whole problem can be avoided by storing data internally and not relying on it being passed from the client via CGI parameters. Session variables in most frameworks are well suited for this purpose.

5. **Common Mistake #5: Security misconfiguration**

In my experience, web servers and applications that have been misconfigured are way more common than those that have been configured properly. Perhaps this because there is no shortage of ways to screw up. Some examples:

1. Running the application with debug enabled in production.
2. Having directory listing enabled on the server, which leaks valuable information.
3. Running outdated software (think WordPress plugins, old PhpMyAdmin).
4. Having unnecessary services running on the machine.
5. Not changing default keys and passwords. (Happens way more frequently than you'd believe!)
6. Revealing error handling information to the attackers, such as stack traces.

Prevention: Have a good (preferably automated) "build and deploy" process, which can run tests on deploy. The poor man's security misconfiguration solution is post-commit hooks, to prevent the code from going out with default passwords and/or development stuff built in.

6. **Common Mistake #6: Sensitive data exposure**
   This web security vulnerability is about crypto and resource protection. *Sensitive data should be encrypted at all times, including in transit and at rest. No exceptions.* Credit card information and user passwords should *never* travel or be stored unencrypted, and passwords should always be hashed. Obviously the crypto/hashing algorithm must not be a weak one – when in doubt, use AES (256 bits and up) and RSA (2048 bits and up).

   And while it goes without saying that session IDs and sensitive data should not be traveling in the URLs and sensitive cookies should have the secure flag on, this is very important and cannot be over-emphasized.

   Prevention:

   - *In transit:* Use HTTPS with a proper certificate and PFS (Perfect Forward Secrecy). Do not accept anything over non-HTTPS connections. Have the secure flag on cookies.
   - *In storage:* This is harder. First and foremost, you need to lower your exposure. If you don't need sensitive data, shred it. Data you don't have can't be stolen. Do not store credit card information *ever*, as you probably don't want to have to deal with being PCI compliant. Sign up with a payment processor such as Stripe or Braintree. Second, if you have sensitive data that you actually do need, store it encrypted and make sure all passwords are hashed. For hashing, use of bcrypt is recommended. If you don't use bcrypt, educate yourself on salting and rainbow tables.

   And at the risk of stating the obvious, *do not store the encryption keys next to the protected data*. That's like storing your bike with a lock that has the key in it. Protect your backups with encryption and keep your keys very private. And of course, don't lose the keys!

7. **Common Mistake #7: Missing function level access control**
   This is simply an authorization failure. It means that when a function is called on the server, proper authorization was not performed. A lot of times, developers rely on the fact that the server side generated the UI and they think that the functionality that is not supplied by the server cannot be accessed by the client. It is not as simple as that, as an attacker can always forge requests to the "hidden" functionality and will not be deterred by the fact that the UI doesn't make this functionality easily accessible. Imagine there's an `/admin` panel, and the button is only present in the UI if the user is actually an admin. Nothing keeps an attacker from discovering this functionality and misusing it if authorization is missing.

   Prevention: On the server side, authorization must *always* be done. Yes, always. No exceptions or vulnerabilities will result in serious problems.

8. **Common Mistake #8: Cross Site Request Forgery (CSRF)**
   This is a nice example of a confused deputy attack whereby the browser is fooled by some other party into misusing its authority. A 3rd party site, for example, can make the user's browser misuse it's authority to do something for the attacker.

In the case of CSRF, a 3rd party site issues requests to the target site (e.g., your bank) using your browser with your cookies / session. If you are logged in on one tab on your bank's homepage, for example, and they are vulnerable to this attack, another tab can make your browser misuse its credentials on the attacker's behalf, resulting in the confused deputy problem. The deputy is the browser that misuses its authority (session cookies) to do something the attacker instructs it to do.

Consider this example:

Attacker Alice wants to lighten target Todd's wallet by transfering some of his money to her. Todd's bank is vulnerable to CSRF. To send money, Todd has to access the following URL:

http://example.com/app/transferFunds?amount=1500&destinationAccount=4673243243

After this URL is opened, a success page is presented to Todd, and the transfer is done. Alice also knows, that Todd frequently visits a site under her control at blog.aliceisawesome.com, where she places the following snippet:

```
<img
src="http://example.com/app/transferFunds?amount=1500&destinationAccount=46
73243243" width="0" height="0" />
```

Upon visiting Alice's website, Todd's browser thinks that Alice links to an image, and automatically issues an HTTP GET request to fetch the "picture", but this actually instructs Todd's bank to transfer $1500 to Alice.

Incidentally, in addition to demonstrating the CSRF vulnerability, this example also demonstrates altering the server state with an idempotent HTTP GET request which is itself a serious vulnerability. HTTP GET requests *must* be idempotent (safe), meaning that they cannot alter the resource which is accessed. Never, ever, ever use idempotent methods to change the server state.

Fun fact: CSRF is also the method people used for cookie-stuffing in the past until affiliates got wiser.

Prevention: Store a secret token in a hidden form field which is inaccessible from the 3rd party site. You of course always have to verify this hidden field. Some sites ask for your password as well when modifying sensitive settings (like your password reminder email, for example), although I'd suspect this is there to prevent the misuse of your abandoned sessions (in an internet cafe for example).

9. Common Mistake #9: Using components with known vulnerabilities

The title says it all. I'd again classify this as more of a maintenance/deployment issue. Before incorporating new code, do some research, possibly some auditing. Using code that you got from a random person on GitHub or some forum might be very convenient, but is not without risk of serious web security vulnerability.

I have seen many instances, for example, where sites got owned (i.e., where an outsider gains administrative access to a system), not because the programmers were stupid, but because a 3rd party software remained unpatched for years in production. This is happening all the time with WordPress plugins for example. If you think they will not find your hidden `phpmyadmin` installation, let me introduce you to dirbuster.

The lesson here is that software development does not end when the application is deployed. There has to be documentation, tests, and plans on how to maintain and keep it updated, especially if it contains 3rd party or open source components.

Prevention:

- *Exercise caution.* Beyond obviously using caution when using such components, do not be a copy-paste coder. Carefully inspect the piece of code you are about to put into your software, as it might be broken beyond repair (or in some cases, intentionally malicious).
- *Stay up-to-date.* Make sure you are using the latest versions of everything that you trust, and have a plan to update them regularly. At least subscribe to a newsletter of new security vulnerabilities regarding the product.

10. Common Mistake #10: Unvalidated redirects and forwards

This is once again an input filtering issue. Suppose that the target site has a `redirect.php` module that takes a URL as a `GET` parameter. Manipulating the parameter can create a URL on `targetsite.com` that redirects the browser to `malwareinstall.com`. When the user sees the link, they will see `targetsite.com/blahblahblah` which the user thinks is trusted and is safe to click. Little do they know that this will actually transfer them onto a malware drop (or any other malicious) page. Alternatively, the attacker might redirect the browser to `targetsite.com/deleteprofile?confirm=1`.

It is worth mentioning, that stuffing unsanitized user-defined input into an HTTP header might lead to header injection which is pretty bad.

Prevention: Options include:

- Don't do redirects at all (they are seldom necessary).
- Have a static list of valid locations to redirect to.
- Whitelist the user-defined parameter, but this can be tricky.

T6.3) Challenges of web security

Overreliance on Web gateways is putting data, users, customers, organizations, and reputation in harm's way.

Once upon a time, organizations primarily used Web gateways to prevent employees from wasting time surfing the Web — or worse, from visiting gambling, adult, and other unauthorized websites.

A few decades later, Web gateways do much more than enforce regulatory compliance and HR policies. Organizations rely on them to thwart Internet-borne threats in three ways:

- Advanced URL filtering, which uses categorization, reputation analysis, and/or blacklists to control access to categories of malicious or suspicious websites.
- Anti-malware protection, which uses various capabilities (such as antivirus, sandboxing, advanced threat protection, content inspection, etc.), to guard against infections caused by various kinds of malware (including rootkits, worms, Trojans, viruses, ransomware, spyware, adware, etc.).
- Application control capabilities, which manage and limit what users are allowed to do in specific applications.

However, although Web gateways have been around for decades and continue to evolve, they aren't bulletproof, and overreliance on them is putting data, users, customers, organizations, and reputation in harm's way. Here are five of the biggest Web gatway security challenges:

1. Filtering out malicious sites
Although URL categorization sounds appealing, this approach is actually very limited. To categorize malicious sites with 100% accuracy, Web gateways need to know how to identify even the most advanced threats. Unfortunately, the attackers' rate of innovation combined with frequent zero-day exploits are leaving Web gateways behind the curve.

To make things worse, it's also hard to keep up when 571 new websites are created every second, which generates a high volume of domains and increases the chance that some will be missed by security controls. It’s difficult for filters to detect the malicious URLs that attackers use for three reasons: URLs may be triggered only by the target organization and remain stealthy during categorization, they’re short lived (less than 24 hours), and they use dynamic domains that are harder to thwart than static ones.

2. Protecting against uncategorized websites without compromising productivity
Employees need access to information to be productive. However, many organizations block access to uncategorized sites because of security concerns, and in the process they reduce end user productivity. Not only does this practice hinder end users, but security teams are forced to deal with an onslaught of support tickets for users who legitimately need to access information. As a result, security teams find themselves maintaining a growing number of policies and rules. This is a major Web security problem because 1% to 10% of URLs can't be classified because of a lack of information.

3. Fighting infections from websites considered safe
The belief that infections occur only through websites that are categorized as suspicious or malicious is false. Websense estimates that 85% of infections occur through websites considered legitimate and safe. It's becoming increasingly common for so-called safe websites to knowingly serve malicious content.

A good example is "malvertizing," which injects malicious ads into legitimate online advertising networks later served by publishers that don't know that ads are malicious. These malicious ads may not even require any user interaction to infect unsuspecting victims. A recent example is the large-scale malvertising attacks that occurred in June and July this year against several Yahoo properties. To circumvent ad blockers' ability to separate banner and display ads, some publishers are integrating ads into their general content. Others, including *GQ* publisher Condé Nast, insist that users disable their ad blockers in order to access content.

Then there's the fact that many seemingly safe websites use common content management systems that are vulnerable to zero-day exploits and can therefore be compromised by attackers to serve malicious content. In July, thousands of websites running WordPress and Joomla — which account for about 60% of all website traffic — served ransomware to all their visitors. And you may remember that back in early 2015, Forbes.com was breached by Chinese hackers who served malicious code via its "Thought of the Day" Flash widget.

4. Identifying malicious files and keeping them out
Although some Web gateways integrate antivirus engines and other file-scanning services, antivirus scanners detect only 20% to 30% of malware.

Leveraging sandboxes to detect malware requires time to run and analyze files. To avoid affecting user experience, Web gateways often pass files to users while sandboxes complete their analysis in the background — which essentially means users are exposed to attacks. Moreover, with the proliferation of sandbox evasion techniques and as malware is often target-specific, sandboxes are proving to be less effective.

5. Neutralizing malware on infected machines
Web gateways only analyze network traffic, not what users are actually doing. As such, gateways have a hard time differentiating between legitimate and malicious traffic, and detecting and neutralizing malware on infected machines. In fact, some advanced threats can be active for weeks or even months without being detected.

Indeed, recent research has found that 80% of Web gateways failed to block malicious outbound traffic. Remote access Trojans are another example of how Web gateways can't detect and stop malicious traffic.

T6.4) Explaining web security measures

A. Basic System Security Measures

The *Basic System Security Measures* apply to all systems at NYU, regardless of the level of their *System Classification*. It is a baseline, which all systems must meet. Note that for most personal workstations, these are the only Measures that apply. The requirements are:

1. **Password Protection**: All accounts and resources must be protected by passwords which meet the following requirements, which must be automatically enforced by the system:
    1. Must be at least eight characters long
    2. Must NOT be dictionary or common slang words in any language, or be readily guessable
    3. Must include at least three of the following four characteristics in any order:upper case letters, lower case letters, numbers, and special characters, such as*!@#$%^&*.
    4. Must be changed at least once per year.
2. **Software Updates**: Systems must be configured to automatically update operating system software, server applications (webserver, mailserver, database server, etc), client software (web-browsers, mail-clients, office suites, etc), and malware protection software (anti-virus, anti-spyware, etc). For *Medium* or *High Availability* systems, a plan to manually apply new updates within a documented time period is an acceptable alternative.
3. **Firewall**: Systems must be protected by a firewall which allows only those incoming connections necessary to fulfill the business need of that system. Client systems which have no business need to provide network services must deny all incoming connections. Systems that provide network services must limit access those services to the smallest reasonably manageable group of hosts that need to reach them.
4. **Malware Protection**: Systems running Microsoft or Apple operating systems must have anti-virus software installed and it must be configured to automatically scan and update.

B. Intermediate System Security Measures

The *Intermediate System Security Measures* define the Security Measures that must be applied to *medium criticality* and *high criticality* systems. Note that except under special circumstances, they do not apply to desktop and laptop computers. The requirements are:

1. **Authentication and Authorization**
    1. **Remove or disable accounts upon loss of eligibility**: Accounts which are no longer needed must be disabled in a timely fashion using an automated or documented procedure.
    2. **Separate user and administrator accounts**: Administrator accounts must not be used for non-administrative purposes. System administrators must be provisioned with non-administrator accounts for end-user activities, and a separate administrator account that is used only for system-administration purposes.
    3. **Use unique passwords for administrator accounts**: Privileged accounts must use unique passwords that are not shared among multiple systems.

Credentials which are managed centrally, such as the NetID/password combination, are considered a single account, regardless of how many systems they provide access to.
    4. **Throttle repeated unsuccessful login-attempts**: A maximum rate for unsuccessful login attempts must be enforced. Account lockout is not required, but the rate of unsuccessful logins must be limited.
    5. **Enable session timeout**: Sessions must be locked or closed after some reasonable period.
    6. **Enforce least privilege**: Non-administrative accounts must be used whenever possible. User accounts and server processes must be granted the least-possible level of privilege that allows them to perform their function.
2. **Audit and Accountability**
    1. **Synchronize system clock**: The system clock must be synchronized to an authoritative time server run by NYU (currently tick.nyu.edu and tock.nyu.edu) at least once per day.
    2. **Enable system logging and auditing**: The facilities required to automatically generate, retain, and expire system logs must be enabled.
    3. **Follow an appropriate log retention schedule**: System logs must be retained for 30-90 days and then destroyed unless further retention is necessary due to legal, regulatory, or contractual requirements.
    4. **Audit successful logins**: Generate a log message whenever a user successfully logs on.
    5. **Audit failed login attempts**: Generate a log message whenever a user attempts to log on without success.
    6. **Audit when a system service is started or stopped**: Generate a log message when a system service is started or stopped.
    7. **Audit serious or unusual errors**: Generate a log message when a serious or unusual error occurs, such as crashes.
    8. **Audit resource exhaustion errors**: Generate a log message when a resource exhaustion error occurs, such as an out-of-memory error or an out-of-disk error.
    9. **Audit failed access attempts**: Generate a log message when an attempt to access a file or resource is denied due to insufficient privilege.
    10. **Audit permissions changes**: Generate a log message when the permissions of a user or group are changed.
    11. **Include appropriate correlation data in audit events**: For each audit event logged be sure to include sufficient information to investigate the event, including related IP address, timestamp, hostname, username, application name and/or other details as appropriate.
3. **Configuration and Maintenance**
    1. **Security Partitioning**: Systems may share hardware and resources only with other systems that have similar security requirements, regardless of their *criticality* classification. Systems which share similar security requirements have user communities of similar size and character, similar firewall profiles, and similar technical requirements. For example:

        1. Multiple systems of the same *criticality* may be aggregated together to share hardware and resources provided they have similar security requirements.
        2. *Medium criticality* systems may share hardware and resources with *low criticality* systems provided that all systems meet the *intermediate systems Security Measures*, and share similar security requirements.
    2. **Follow vendor hardening guidelines**: This document cannot be comprehensive for all systems available. Follow basic vendor recommendations to harden and secure systems.
    3. **Disable vendor default accounts and passwords**: Many systems come with default accounts which are publicly known. These accounts should be disabled.
    4. **Disable all unnecessary network services**: Processes and services which are not necessary to complete the function of a system must be disabled.
4. **Additional Requirements**
    1. **Report potential security incidents**: Potential security incidents must be reported to the IT Office of Information Security.
    2. **Security review**: During the design of the technical architecture, a review of the system must be requested from the NYU IT Office of Information Security.
    3. **Vulnerability assessment**: Before system deployment, a vulnerability assessment must be requested from the NYU IT Office of Information Security.
    4. **Physical access**: The system must reside in a locked facility, to which only authorized personnel have access.
    5. **Documentation**: Create and maintain documentation summarizing the business-process, major system components, and network communications associated with a system.

C. Advanced System Security Measures

The A*dvanced System Security Measures* define the Security Measures that must be applied to *high criticality* systems. The requirements are:

1. **Audit and Accountability**
    1. **Enable process auditing or accounting**: Enable process auditing or accounting, which generates logs information about the creation of new processes and their system activity.
    2. **Audit privilege escalation or change in privilege**: Generate a log message whenever a user changes their level of privilege.
    3. **Audit firewall denial**: Generate a log message when the host-based firewall denies a network connection.
    4. **Audit all significant application events**: Log all significant application events.
    5. **Write audit events to a separate system**: System logs must be written to a remote system in such a way that they cannot be altered by any user on the system being logged.
2. **Configuration and Maintenance**

1. **Follow advanced vendor security recommendations**: This document cannot be comprehensive for all systems and applications available. Conform to best practices and recommendations outlined in vendor security whitepapers and documentation.
2. **Host-based and network-based firewalls**: Systems must be protected by both a host-based and a network-based firewall that allows only those incoming connections necessary to fulfill the business need of that system.
3. **Configuration management process**: Configuration changes must be regulated by a documented configuration and change management process.
4. **Partitioning**: Systems may share hardware and resources only with other systems that have similar security requirements, regardless of their *criticality* classification. Systems which share similar security requirements have user communities of similar size and character, similar firewall profiles, and similar technical requirements. For example:
    1. Multiple systems of the same *criticality* may be aggregated together to share hardware and resources provided they have similar security requirements.
    2. *High criticality* systems may share hardware and resources with *medium* and *low criticality* systems provided that all systems meet the *advanced systems Security Measures*, and share similar security requirements.

3. **Additional Requirements**
    1. **Physical access**: The system must reside in a secured, managed data-center.

D. Data Handling Security Measures

These *Data Security Measures* define the minimum security requirements that must be applied to the data types defined in the *Reference for Data and System Classification*. Some data elements, such as credit card numbers and patient health records, have additional security requirements defined in external standards. In addition, access and use of University Data is covered by the *Administrative Data Management Policy*. Please be sure to consult all appropriate documents when determining the appropriate measure to safeguard your data.

The best way to safeguard sensitive data is not to handle it at all, and business processes that can be amended to reduce or eliminate dependence on *restricted data* should be corrected. For example, the University ID number can often be substituted for a social security number and poses much less risk if accidentally disclosed.

1. **Requirements for Handling Confidential Data**
    1. **Access control**: Access to *confidential data* must be provided on a least-privilege basis. No person or system should be given access to the data unless required by business process. In such cases where access is required, permission to use the data must be granted by the *data steward*.
    2. **Sharing**: *Confidential data* may be shared among the NYU community. It may be released publicly only according to well-defined business processes, and with the permission of the *data steward*.
    3. **Retention**: *Confidential data* should only be stored for as long as is necessary to accomplish the documented business process.

2. **Requirements for Handling Protected Data**
    1. **Access control**: Access to *protected data* must be provided on a least-privilege basis. No person or system should be given access to the data unless required by business process. In such cases where access is required, permission to use the data must be granted by the *data steward*.
    2. **Sharing**: *Protected data* may be shared among University employees according to well-defined business process approved by the data steward. It may be released publicly only according to well-defined business processes, and with the permission of the *data steward*.
    3. **Retention**: *Protected data* should only be stored for as long as is necessary to accomplish the documented business process.
    4. **Incident Notification**: If there is a potential security incident that may place protected data at risk of unauthorized access, the NYU IT Office of Information Security must be notified.
3. **Requirements for Handling Restricted Data**
    1. **Collection**: Restricted data should only be collected when all of the following conditions are met:
        1. The data is not available from another authoritative source, and
        2. The data is required by business process, and
        3. You have permission to collect the data from the appropriate *data steward*; or
        4. If the data is requested by the Office of General Counsel in response to litigation.
    2. **Access control**: Individuals must be granted access to restricted data on a least-privilege basis. No person or system may access the data unless required by a documented business process. In such cases where access is required, permission to use the data must be granted by the *data steward*.
    3. **Access auditing**: Enable file access auditing to log access to files containing restricted data.
    4. **Labeling**: Portable media containing restricted data should be clearly marked.
    5. **Sharing**: Access to restricted data can be granted only by a *data steward*. No individual may share restricted data with another individual who has not been granted access by a data steward.
    6. **Idle Access**: Devices which can be used to access *restricted data* must automatically lock after some period of inactivity, through the use of screensaver passwords, automatic logout, or similar controls.
    7. **Transit encryption**: Restricted data must be encrypted during transmission with a method that meets the following requirements.
        1. Cryptographic algorithm(s), the list of approved security functions.
        2. Cryptographic key lengths meet best-practices for length, given current computer processing capabilities.
        3. Both the source and destination of the transmission must be verified.
    8. **Storage encryption**: Restricted data must be encrypted using strong, public cryptographic algorithms and reasonable key lengths given current computer processing capabilities. Keys must be stored securely, and access to them provided on a least-privilege basis (see ISO 11568 for recommendations on

securing keys). If one-way hashing is used in lieu of reversible encryption, salted hashes must be used.
    1. Encrypt files containing restricted data using different keys or passwords than those used for system logon.
    2. Encrypt data stored in databases at the column-level.
    3. In addition to file and/or database encryption, implement full-disk encryption on portable devices containing restricted data.
9. **Retention**: Restricted data should only be stored for as long as is necessary to accomplish the documented business process.
10. **Destruction**: When restricted data is no longer needed it should be destroyed in accordance with applicable policies, using methods that are resistant to data-recovery attempts such as cryptographic data destruction utilities, on-site physical device destruction, or NAID certified data destruction service.

TOPIC 7: EMERGING TRENDS IN INTERNET BASED PROGRAMMING

T7.1) Emerging trends in internet-based programming

As social media, app stores and global availability become standard, many companies are looking to enhance the online customer experience. And while retail and other transactions via Internet are customary, more than ever companies are simplifying the ways in which customers interact with their website and ultimately make online purchases. Here are eight trends happening right now in global e-commerce that seek to enhance the user experience:

1. Micro-payments: Among the most revolutionary changes in the coming months—not years—is the use of micro-payment systems from a variety of financial firms, e.g., Paypal, Visa, WesternUnion, among others, including banks. This trend is facilitated by the W3C working group that approved these protocols and technical standards for the interworking. These systems will change not only how we carry money but how we value money and think about purchases. (Consider how a purchase of $4.99 feels in a mobile app store vs. at Dunkin' Donuts.) Payment systems that make it easier to buy online, coupled with mobile technologies will accelerate the usage of global e-commerce applications.

2. Mobile technologies: More people access the Internet on their mobile devices than on any other device. We are rapidly approaching the time (if we are not already there) where designs must be created for the mobile web first, and for the desktop second. Mobile technologies facilitate comparison shopping; with the advent of barcode reader apps and price-comparison databases, a consumer could snap a bar code in Walmart and quickly reference product reviews and prices on walmart.com (or compare prices with Walmart competitors). Mobile technologies also facilitate impulse buys – especially with the advent of micro-payments tied to the mobile device. Just recently, Starbucks customers can not only place an order with their Smartphone, but also make a purchase.

3. Social media: As Facebook has become the most visited site on the Web, the role of social media, including Facebook and its local clones such as Twitter, is increasingly important. Social media sites increasingly act as points of entry to e-commerce sites, and vice versa, as e-commerce sites build rating, loyalty and referral systems tied to social media. Group buying (e.g., Groupon) is also gaining mainstream ground, with many "deal of the day" sites competing for an increasingly savvy consumer base, but improvements lie ahead as the social aspects and user experience are refined.

4. Fulfillment options: I believe that users will want to have multiple fulfillments and return options when interacting with a vendor: ship to address, courier, pick-up in store, return to store, etc. Having many fulfillment options is how customers view their overall customer experience. Some companies have made a business proposition online by being exceptional in service to the online channel (e.g., Zappos).

5. Global availability: Increasingly, consumers want the availability to buy products from foreign sites and have them delivered locally. Thus, currency and customs will be of growing

concern to many online retailers. Along with this, there will be concerns with local privacy laws and restrictions on related data collection and storage.

6. Localization: While the trend is to globalize, what's often more important is to localize. User Centric's (now GfK's User Experience team) research clearly shows that sites that 'feel' local – with proper imagery, language, time/date, weights/measures, currency, etc. – resonate far more than sites that seem culturally distant or sterile.

7. Customizability: Consumers want control, and want to be able to design the details of the items they purchase.

8. Time-based availability: Some of the hottest and most successful sites are those that have a time-critical response component. Sites like Groupon, Gilt and others capitalize on the perception of limited-time availability. Creating a sense of urgency drives traffic and purchase behavior.

T7.2, T7.3) Challenges & Solution of emerging trends in internet-based programming

1. Job Losses

Computers, robotics, and automation are driving more and more of production. In turn this is leading to an enormous impact on the number and type of jobs. An Australian report released in June 2015 found that 40 per cent of the Australian workforce – or around 5 million jobs – are at high risk of being replaced by computers in the next 10-15 years. This backs up the Oxford Martin School's 2013 study finding 47 per cent of jobs in the United States are at risk of being automated using artificial intelligence. We need to move urgently from a discussion about protecting the jobs of today, to creating the jobs of the future.

2. Commercialization and innovation

There are significant emerging opportunities and challenges for commercialization and innovation resulting from technological changes to becoming a more sustainable, broad-based economy:

1. Reducing the tyranny of distance, boosting trade and creating new business models but also promoting outsourcing of work overseas.
2. Internationalizing labour markets are expanding the skilled labour pool.
3. Developing commercially functional goods and services from new technologies often takes a lot longer than expected.
4. Leveraging clean technologies to improve sustainability.
5. Fostering entrepreneurship and addressing constraints for Kiwi companies.
6. Addressing slow uptake of new technology due to redundancy risks or ease of sticking with the status quo and supporting workforce mobility.
7. Managing business change in a disruptive and dynamic business environment.

4. On-demand economy

Stable, permanent fulltime jobs are increasingly being replaced by an anywhere, anytime work model, facilitated by digital technology which is resulting in a shift towards more contract work and a more rootless and flexible workforce.

The "on-demand economy" is the result of pairing that workforce with smartphones and other devices, which now provide far more computing power than the desktop computers which reshaped companies in the 1990s, and reach far more people.[2]

The on-demand economy is starting to revolutionize commercial behavior in cities around the world. Fast-moving tech companies competing in this arena have developed new models – such as Uber, Handy and Air B&B – that are transforming industries which have been historically slow to innovate. Transportation, grocery, restaurant and personal service industries are seeing hyper-growth in the on-demand world.[3]

However this means a growing gap emerging between workers and their ultimate boss. Ensuring workers retain their voice within their company is crucial to ensuring new business models remain responsive. Emerging technology provides more ways than ever to ensure that this remains possible.

The on-demand economy gives consumers more choice. Consumers may be winners, as can workers who value flexibility over security such as younger workers, those with portable skills in demand who attract higher wages, or those who don't want to work fulltime. But those who value security over flexibility, have families or have mortgages are all threatened. In addition, there are inequities for those who work in the on-demand economy but do not qualify for superannuation and other benefits. Care is needed to minimize the impact of change on employment rights and health, safety and environmental protections.[4]

Smart policy makers can't stand in the way of change. We can't outlaw on-demand firms. But we can improve the ways in which we measure employment and wages, and we must stop treating contractors and freelancers as second class citizens. In effect every contractor is a small business with the insecurities, demands and potential that goes with that title.

5. Redefining work

Increasing use of digital business models alongside automation and computerization of jobs will see organizations shift to a smaller number of highly skilled people with scarce skills working in very different patterns, in order to enhance their competitiveness. Risk and change management will be crucial to ensuring success here. This has implications not just on how we manage work but also on the quality of life for our workforce. An AUT study into mobile technology found it contributes to irregular patterns of work, amplifies social pressures making boundaries between work and non-work indistinguishable, brings more work into personal time, and speeds up the way organizations function. Defining when a person is working and when they are not will be an increasing challenge.

6. Accessibility

Cheap computing power is transforming the way consumers and workers access technology as even more sophisticated and powerful hand-held smartphones become available. This

eliminates some of the barriers for how work is done. Complex tasks such as programming a computer or writing a legal brief can now be divided in component parts and subcontracted to specialists around the world. It also gives greater flexibility – providing an opportunity for workers and workplaces to create flexible working arrangements.

7. Big data

Big data is changing the way big business operates. Big data involves data collection and mining to ascertain consumer preferences and behaviour trends that assist companies to customise their offerings and specifically target their markets. Prompts on Amazon.com for related book titles are one example of this.

Big data creates new markets and new opportunities. It also drastically increases privacy risks and raises issues of resilience of cloud based applications and storage to hacking and other vulnerabilities.

8. Education and Training

Our education system will need to adapt. There is a tension between a model of education which is aligned to current industry demands churning out work-ready drones who will 'hit the ground running' and a model which enables rounded professional development and boosts worker's capacity to learn and think in a world where creative and critical skills are at a premium.

We are not currently training our workforce to be adaptable enough to changes in technology or providing proper lifelong education solutions for retraining. More needs to be done to prepare our workforce for the changes to come including looking at universal teaching computing and coding in schools and improving how we teach technology.

9. Infrastructure

New Zealand is rolling out ultrafast broadband which is transformational. However there are road blocks to UFB roll-out and uptake. There are huge opportunities for smaller geographically distant countries like New world where IT has reduced the tyranny of distance. This raises important issues around getting our infrastructure right inside New world and reducing the digital divides that exist. We must have robust, resilient and future proofed affordable international connectivity.

10. Digital Divide

Many people and businesses still lack basic access to broadband particularly rural, communities. The cost of access to the internet, digital devices, and big data also means many small businesses and lower income households miss out. This means some of our kids don't get the education they need because they can't access the internet at home, our small businesses are held back and too many New Zealanders miss out on their right to enjoy access to emerging technologies. Technology is now essential to modern life and learning. This divide applies not just within society but between countries and New world must ensure it remains on the winning side of that divide. We are failing to deal with the growing digital divide. We are losing ground because of lag in the ultra-fast broadband rollout. Schools, businesses and homes need access to high speed internet as soon as possible while free WiFi access in more community areas would do much to bridge the digital divide. Options like

Google’s Project Loon to use high-altitude balloons to deliver internet to rural areas also need to be explored further.