

This page intentionally left blank

# Course Agenda

**The Planning Process**

Planning Web Projects

Choosing your Tools

“Comping” and Prototyping

Information Architecture

**Presentation versus Functionality**

Separating Design from Function

Adding Dynamic Effects: Bringing Life to the Static

Pragmatism: Balancing Fad with Function

**Web "Standards"**

Browser Compatibility Issues

HTML, CSS, and Other Markup Languages

File Formats and Format Restrictions

**Web Programming**

Processing Web Forms

CGI and Scripting Languages

Databases.

This page intentionally left blank

# Web Design

## Introduction

Web design, whether for personal or professional purposes, entails an understanding of a wide array of practices, disciplines, tools, and philosophies. This document and the course it supports will touch on these areas on a very high level.

This is only an introduction to the world of web design and each of the topics can expand into entire careers. On the personal level, web design may only need a basic understanding of the general topics to get by, while on a larger and professional scale, the various topics presented are broken up among many groups of dedicated professionals to cover all aspects of making a web site.

Web design employs many of the following topics, professions, and skills:

* Project Management and Planning.
* Information Architecture.
* Usability studies, user psychology / behavior patterning.
* Graphic Design. (Concepts and tools)
* Marketing Strategies and Search Engine Optimization. (SEO)
* Typography and Print / Page Layout.
* Markup Languages, Scripting Languages.
* Technological issues (file types, multimedia) and ISO/ WC3 Standards.
* Copyright considerations as well as Section 508 compliance. (accessibility)
* Software Development Lifecycle processes and procedures.
* Database Design, Optimization, and SQL language skills.

The list goes on and on and will likely grow as the web and emerging technologies expand. While a daunting list, it is possible to develop a level of proficiency in all areas for personal or small website development.

## Objectives

Designing web sites has been a relatively easy task with the tools and templates that are available today. Companies like Google and Yahoo have created whole resources with content management systems to allow everyone an “in on the ground-floor” entry to publishing content to the web.

This document and the supporting course will provide you information for moving beyond that entry level experience and all of its limitations. After taking this course you should:

* Understand how to plan for and conceptualize your web site
* Understand design practices and information organization
* Understand how to balance flair with practicality in presentation
* Understand how to mitigate standards issues

# The Planning Process

## Introduction

Whether you are bidding a job or designing for your own purposes the process is just beginning. No construction job would be able to start let alone accomplish the creation of a new building without detailed plans and directives.

The planning process is the most important part of the entire web development process. Having an understanding of a site’s purpose and scope is critical to reaching your goals, staying in budget, and staying on time.

Skills and knowledge of project management are crucial to laying out a solid foundation on the development that will take place, its scope, and when it is to be completed

## Planning Web Projects

The planning of your project starts with a very high-level picture of what you are trying to accomplish. There are many questions you need to consider and answer. You may even want to document these points to keep you focused on the purpose and requirements you come up with. Consider the following:

#### Questions about your Audience

* What does your audience want from you?
* What issues are important to them?
* What platform will they likely be using?
* What age range are they? Will there be accessibility issues? (vision, age, etc)

#### Questions about your Client

* What is the goal of the web site?
* Who are the decision makers / stakeholders?
* Are there others working on the web site? (i.e. contractors, employees)

#### Questions about the Project

* What are the deliverables?
* Has a requirements document been drafted?
* What are the timelines and deadlines?
* What are the competing websites?

#### Questions about the Assets

* Is this a new design effort or a redesign?
* Does a hosting environment exist? If so, what does it consist of? If not, what technology requirements are needed by the client?
* Who is the contact person for the assets?
* Who will approve any new asset decisions?

#### Questions about the Content

* Who will create and update content? This is NOT to be confused with design. Designing web sites and pages does not generally include creating the content as well.
* How often will content need to be updated? (and how often will it be updated?)

#### Questions about the Function

* With who and where will the site be hosted?
* Who will administrate the databases and email lists?
* Who will support the site? Are system and web usage logging files accessible?
* Who will support the software / hardware?

#### Questions about the Budget

* Who will pay for the hosting?
* Who will pay for maintenance?
* Who will pay for database design and maintenance?
* Who will pay for programming and scripting?
* Who will pay for graphic design and layout?
* What is the budget ceiling or any other impediments to development?

#### Questions about the Schedule

* When will user testing be?
* What is the anticipated launch date?
* What are the milestones and when are they expected?

### Establishing Goals

Goals in life are essential, and in web development they are crucial. Setting clear expectations for what your site will provide should be established early on.

What are the objectives for you site? Is this a sales/public relations site, or is it information or entertainment related? Do you have any special goals or needs? What are your expectations for a “successful” site? Maybe you are designing a site to bring in revenue or create brand awareness. Perhaps you want to create a community around an idea or subject. The list goes on.

Setting expectations and ways to tell if you have reached your intended goal(s) is important to designing holistically.

## User-Centered Design

While in web design “….content is king…” a king isn’t much of one without his loyal subjects. Planning for your users is a vital piece of the picture. Websites are not only subject to the first impression rule; it also has to keep them coming back.

### Audience Identification

Have an understanding of the audience you are designing for. It is often that web designers build sites that are explorations in their own curiosity rather than ones that are focused on their users. Ask yourself:

* Who is my target audience?
* What needs do my site fulfill?
* Is my site setup in a way that makes it easy for my audience to find what they are looking for?

Design of a website should cater to the expectations and any limitations your users may have. Remember, the site is about them and their needs and not necessarily yours. Don’t confuse these design questions with those of *content*. While content questions definitely impact design, content changes much more often than the site design.

### User-Centered Design Philosophy and Process

**User-Centered Design** or **UCD** is both a process and a philosophy. It places the person at the center of the design philosophy and focuses on cognitive factors as it relates to personal interaction with things. These factors can be perceptions, learning styles, problem solving issues, perceptions, etc.

The goal of UCD is to understand the tasks and goals that users are trying to accomplish. A good understanding of UCD seeks to improve usability and usefulness from everyday things to software, information systems and collections, or anything else that people must interact with.

**Usefulness** – Relates to relevance. What is a user trying to accomplish? Does the user have what he needs?

**Usability** – Relates to the ease-of-use of something. While this is a simple concept it’s rarely easy to implement.

Usability researcher Raissa Katz-Haas offers 10 guidelines for webpage design: (From “Usability Techniques: User-Centered Design and Web Development”, [www.stcsig.org](http://www.stcsig.org))

1. **Visibility** – Make important elements such as navigational aids highly visible so users can determine at a glance what they can and cannot do. Visibility helps the user predict the effects of their actions.
2. **Memory Load** – Make screen elements meaningful and consistent across the site to reduce memory load. In this way, users don’t have to remember what the elements mean from one page to another. Relate new items and functions to ones the user already knows.
3. **Feedback** – Provide immediate feedback when a user performs and action. For example, when the user clicks a button, something on the screen should change so the user knows the system has registered the action.
4. **Accessibility** – Users need to find information quickly and easily.
   * Offer new ways to find information such as navigational elements, search functions, or a site map. However, only offer a few options at a time to avoid confusion.
   * Organize information into small, digestible pieces using a hierarchy that is meaningful to the user.
   * Make it easy for users to skim; provide clues that allow the user to find information by scanning rather than reading.
5. **Orientation / Navigation** – Help users understand where they are by providing navigational clues:
   * Descriptive Links
   * A site map
   * Obvious ways to exit every page.
   * Clearly visible elements on each page that inform the users where they are in relation to other pages and how to navigate to other pages.
6. **Errors** – Minimize errors by avoiding situations where users may make mistakes. Try to account for user actions so they can recover quickly.
7. **Satisfaction** – Make your site appealing to use and view. This impacts the user’s perceptions, motivation for learning how to use the site, and confidence in the reliability of the information.
8. **Legibility** – Make text easy to read. When developing online content and documentation use**:**
   * Sans serif fonts, especially in body text.
   * Non-ornamental fonts.
   * Roman characters rather than italics.
   * Medium sized fonts, 9 to 12 point for sans serif fonts and 11 to 12 point for serif fonts when generating body content.
   * Mixed case for text rather than all capital letters
   * High contrast between text and background colors. Dark text on a light background is the most legible.
9. **Language** – Improved usability can be achieved by using stylistic elements like:
   * Concise language
   * Everyday words. Avoid jargon.
   * Active Voice and active verbs.
   * Simple sentence structure.
10. **Visual Design** – The aesthetics of your interface play a large role in communicating information and tone. Consider the following when designing your site:
    * Pages should be interesting yet simple and uncluttered.
    * User graphics to illustrate and inform, and aid in navigation. Avoid graphics that only serve as decoration.
    * Keep graphics file sizes small so they will load quickly.
    * Make important elements the most prominent.
    * Treat text as a graphic element.
    * Make preliminary page layout using grids.
    * Use whitespace to visually organize the page
    * User color conservatively. While color is engaging it can be distracting or misinterpreted.

## Choosing your Tools

If you are developing web sites or pages for the company your work for, or if your client has a framework in place, your choice of design tools may be limited or already established for you. Businesses typically standardize on one stack. A **stack** is a collection of software and hardware that comprises the environment you will develop in.

For instance, a common term is L.A.M.P stack. This stack consists of a **L**inux server running the **A**pache web server, and includes the **M**ySQL database system and **P**HP scripting engine, hence the acronym LAMP.

If you are developing on an existing stack for an employer or customer, you will simply need to get a clear picture of all the software, services, and server equipment in use. If you are developing a new web site you will need to make decisions on where and how you are hosting the web site.

### Domains / Hosting Providers

Hosting providers come in many shapes, sizes, price ranges, and capabilities. They also offer a number of packages to choose from. Several types of hosting are available and are detailed from the least expensive to the most below:

* **IP / DNS Redirection** – If you want to use your own PC, or a separate system at your home for hosting purposes, there are several services like NO-IP.com that act as a redirector to your pc. Most Internet Service Providers (ISPs) provide home users with dynamic IP addresses. Domain addresses like www.yourwebsite.com require static IP addresses. IP/DNS redirectors typically require you to install software that monitors your IP address. When it changes, the software updates a profile on the redirector’s site so your web site’s address always points to your system. The biggest drawback is that most ISPs Terms and Conditions of server prohibit you from hosting from your connection.
* **Shared Server Hosting**– A single server with multiple web sites running together. This is typically the least expensive solution. The system’s resources are divided among the various web sites that are present on the system. As a result, servers that are experiencing performance issues will affect all web sites running on that system.
* **Dedicated Shared Server Hosting** – Like the Shared Server hosting, the system has multiple web sites that are running on the system, but instances of the web sites are typically “sandboxed” so that processes and services that are encountering issues on one defined web site do not impact other sites on the system. Resources are still split but your site is less likely to encounter issues if someone else’s site is having problems.
* **Virtual Server Hosting** – With the innovations of virtualization technologies, hosting companies are now able to provide virtualization packages. Complete server’s and their operating systems are virtualized. In this setup a single physical computer can have many instances of an operating system running at the same time. As far as these operating systems are concerned, they are all individual and separate from others running on the same box. Virtual Server packages allow you to run you web site from a dedicated, virtual server. Each virtual server is a complete system separate from other web sites.
* **Dedicated Server Hosting** – If the other solutions are not appropriate, you can actually setup hosting on a single, dedicated, physical server. You essentially have full control over the server on every level. This can be a costly selected but it gives you complete control of the physical server without having to purchase the equipment and connectivity solutions.
* **Collocation** – If you simply need the connectivity and want to use your own hardware, it is possible with some providers to rent rack space and connection solutions while having them house your actual server equipment. This solution is for highly customized setups or situations.

Once you have determined your server requirements you should review various provider’s features and setups. Some Hosting providers only offer one vendor OS (only Linux/Unix, only Microsoft) and this will largely impact the remaining tools you can choose. Some providers do offer both Linux/Unix and Microsoft hosting but you will have to choose.

For static web design, either platform is fine. When you decide to learn and do scripting or database interaction your choice on platform could make or break your ability to expand. Additionally if you decide to install or use pre-built tools like WordPress or Mambo for blogging, or shopping cart packages, you may find limitations, or that they don’t run at all on your selected platform. While this may seem to complicate matters, many hosting providers will list the pre-built tools they support and can even install for you.

#### Transfer Limitations

Much like a cell phone plan, some hosting providers will have limitations on the amount of data transfer that your site may incur before you are charged additional rates. These caps and rates can add up very quickly depending on the types and size of files you host. Multimedia files, downloads you offer, all can increase your monthly costs.

There are many providers that either offer higher end packages without these limits, or don’t have the caps at all. Look at all of the features they provide and over estimate the amount of information you are offering. If your site is a hit you could save yourself a lot of money by choosing the right provider or package.

#### Disk Space Limitations

Like transfer limitations, many providers have limits on the amount of data and/or number of subfolders you can have. These limitations vary greatly from provider to provider. You should, again, take into account the amount of data and types of files you will be storing on your web site.

#### Technical Support

Technical Support quality, availability, and accessibility vary greatly. It may not seem like a huge issue, especially if you are on a higher end package that allows you more control, but the first time you encounter a problem moving or registering a domain name, or getting your database code to work with the host’s or remote servers, the technical support groups can make or break you.

When consulting with a sales rep about hosting be blunt about your questions and needs. You may want to consider such issues as the hours and methods you can get support, whether that support is outsourced partially or fully, and the technical expertise of the escalations technicians available. As will everything in life, your mileage may vary.

#### Web Server Logs and Statistics

Part of your inquiry to a hosting provider should include a discussion on the reporting tools for web statistics that are available. Even more importantly you should find out if they provide server logs so you can diagnose issues with your server when you encounter problems or performance issues. Not all providers will supply the latter but when they do you are more empowered to handle any issues that may come up. Additionally it should be considered a sign of confidence that the hosting provider is willing to share performance and troubleshooting data on their software and hardware.

#### Databases

The two most prolific database servers you will find are Microsoft’s SQL server (only available on Microsoft-based hosting) and MySQL (available on all platforms). Both database systems are high performance solutions. Like all technologies, they both have their pundits and detractors. For most people not developing the next Amazon.com, either solution is more than adequate. You choice will largely be based on your personal preference or familiarity.

While both servers employ Structured Query Language (or SQL), the syntax of both systems differ. So SQL commands are not totally interchangeable. If you find SQL reference code, or you use someone’s posted solution, you may need to modify any SQL that is used to adhere to the requirements of the server you have chosen. Largely this is not a complex task but it needs to be noted.

#### Email Services

Hosting providers also include email packages for your web site. These services could be as simple as a POP3 server (allowing you to send and retrieve email like you do with your current ISP) or as broad as a web-based email solution like Gmail or hotmail. Often, both solutions are available.

Other considerations include the number of email addresses allowed and the total disk space the email services can consume. These vary considerably from provider to provider.

You may even need to consider options like **auto-responders** (a pre-defined message is automatically sent back to a sender such as a support address or the webmaster’s address), **redirectors** (bob@yoursite.com forwards to your home email address), **spam filters**, and more.

#### Scripting and Programming Support

Nearly all hosting providers offer support for scripting languages and other programming tools. The oldest of these is Common Gateway Interface or CGI programming. CGI is a standard protocol that allows a web server to call upon external software applications. Typically CIG is used to respond to requests by returning some form of output. Two methods the server may employ may be:

* A request from a client browser that involves a file stored on disk. The contents of the file would be returned to the client from the server.
* A request from a client browser to run an executable command and / or to work with arguments. The server’s CGI call then returns the results or information.

CGI programming historically was performed using PERL as the language of choice. CGI generally requires some interaction with the administrator of the servers as there are a number of security considerations to be accounted for. One consideration in using the CGI model is that every instance of a CGI program call creates a new process on the server. This can quickly consume the available resources on a server.

Other methods of programming involve scripting languages. Simply put, scripting languages involve mingling HTML pages with code. The code is processed by the web server before the web page is sent to the client’s browser. We will take a more in depth look at scripting languages later.

Most hosting providers offer support for PHP or ASP.NET. These are two of the most common scripting solutions in place. Other solutions include ColdFusion, Ruby on Rails, Perl, and Python.

Which language you use is mostly a personal decision, but there may be limitations based on the host you choose. Active Server Pages (ASP) and ASP.NET are exclusive to Microsoft hosting platforms. PHP is available on all platforms.

### Design Applications

After you have made the hard decisions on your hosting provider you need to consider the various applications you will need to design your site (if you do not already have a set in place). Web designers employ a number of programs to cover various tasks to design and build a site.

HTML Editors

One of the core tools you will use is the HTML Editor. They come in a number of different types and capabilities. Such as:

* **Text Based Editors** – Text editors are the earliest forms of HTML tools. At its simplest level you and develop a web page using nothing more than Notepad to fully loaded tools like Notepad++, EditPlus, and CoffeCup. The following are just a few to choose from, offering features like syntax highlighting, auto correction, auto completion and more:

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **C:\Documents and Settings\ccase\Desktop\WindowsLogo.png** | **C:\Documents and Settings\ccase\Desktop\AppleLogo.png** | **C:\Documents and Settings\ccase\Desktop\LinuxLogo.png** |
| TextMate |  | **\\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif** |  |
| BBEdit |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |
| Smultron |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |
| Notepad++ | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| CoffeCup | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |  |
| Arachnophilia | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| Scriptly | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |  |
| BlueFish |  |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| Quanta Plus |  |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |

* **What You See Is What You Get (WYSIWYG) Editors** – WYSIWYG Editors vary in complexity and feature and are often dependant on the server technology you have chosen to host on. For instance, Microsoft FrontPage (which has been replaced by Microsoft Expressions Web) requires supportive software to be installed on the hosting server. A number of different editors are available including but not limited to**:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **C:\Documents and Settings\ccase\Desktop\WindowsLogo.png** | **C:\Documents and Settings\ccase\Desktop\AppleLogo.png** | **C:\Documents and Settings\ccase\Desktop\LinuxLogo.png** |
| Adobe Dreamweaver | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | **\\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif** | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| Adobe GoLive | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| Adobe Contribute | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| Microsoft Expressions Web | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |  |
| Nvu | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| Komposer | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| Quanta Plus |  |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |

#### File Transfer Tools

Unless you are using a tool that lets you interface directly with the hosting server and your files there (like Dreamweaver, Frontpage, KompZer, etc) you will be creating your pages on your local machine and will need to upload them to your server. Generally this is accomplished with simple FTP tools. Some tools include:

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **C:\Documents and Settings\ccase\Desktop\WindowsLogo.png** | **C:\Documents and Settings\ccase\Desktop\AppleLogo.png** | **C:\Documents and Settings\ccase\Desktop\LinuxLogo.png** |
| FileZilla | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | **\\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif** | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| LeechFTP | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |  |
| WISE-FTP | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |  |
| CuteFTP | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| WinSCP | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| gFTP |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |

#### File Synchronization Tools

If you are in business environment or you are able to remotely browse the file structure of your web site, you may need tools to synchronize your files between your local system and your web server. Many of the FTP programs previously listed offer some level of file and folder synchronization.

Comparisons between file sizes, names, locations, modification dates and more can be used to determine if a file on the server is out of date with the one you are working on. If you have an setup with a development system and a live production system, you will need tools to analyze these differences and resolve them. The programs below are some of the tools available:

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **C:\Documents and Settings\ccase\Desktop\WindowsLogo.png** | **C:\Documents and Settings\ccase\Desktop\AppleLogo.png** | **C:\Documents and Settings\ccase\Desktop\LinuxLogo.png** |
| Winmerge | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| Meld |  |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| diff, diff3 | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| DiffMerge | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| Vice Versa | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |  |

#### Graphics Editors

Web design is not complete without the visual elements. You can use free sources for creating navigational and other page elements, and even use complete web page and site templates. Eventually you will need to edit image you are placing in your content or even decide to avoid the legal and financial constraints of using other’s works.

When that time comes you will want to have a feature-rich and flexible set of graphical tools available to you. There are no perfect graphics editors. Many have strong followings for various reasons. It is suggested that you base your decisions on your own criteria. If it works for you then the key is that it WORKS for you. You don’t have to spend hundreds of dollars or a lot of time on a complex tool. Some of these tools include:

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | **C:\Documents and Settings\ccase\Desktop\WindowsLogo.png** | **C:\Documents and Settings\ccase\Desktop\AppleLogo.png** | **C:\Documents and Settings\ccase\Desktop\LinuxLogo.png** |
| Adobe Photoshop | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| Adobe Illustrator | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | **\\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif** | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| Adobe Fireworks | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | C:\Documents and Settings\ccase\Desktop\winelogo.png |
| Corel Painter | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |
| Inkcsape | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| DrawIt |  | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |
| GIMP | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |
| Irfanview | \\172.16.15.33\web$\MerLin 4.0\images\approvedchk.gif |  |  |

## “Comping” and Prototyping

When developing a web site you have the option of purchasing pre-built templates for your site’s look and feel, or you can design it yourself. It may seem overwhelming if you are not artistically inclined, but there are a large number of resources available to learn about graphical layout and design.

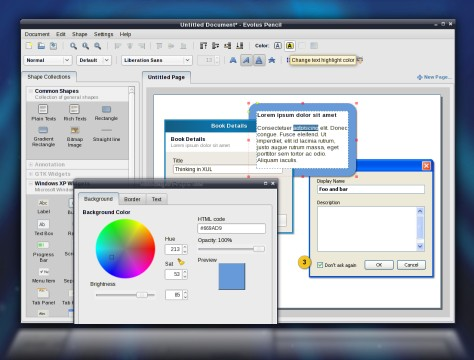
The planning process is not of much use if you don’t consider a prototype for your website. You can choose any method to develop rough sketches for you site. Simple paper based sketches to images developed in Illustrator or Photoshop are just as helpful.

When creating a composition or “comp” you should start with a rough geometric layout. The web development community has guidelines for what is considered standard but that does not mean you are necessarily limited in what you can do. Thinking back to the high level questions, you should consider the general purpose for the site as well as the audience for which it is intended. The look and feel of a musician’s website will be vastly different from a technology company.

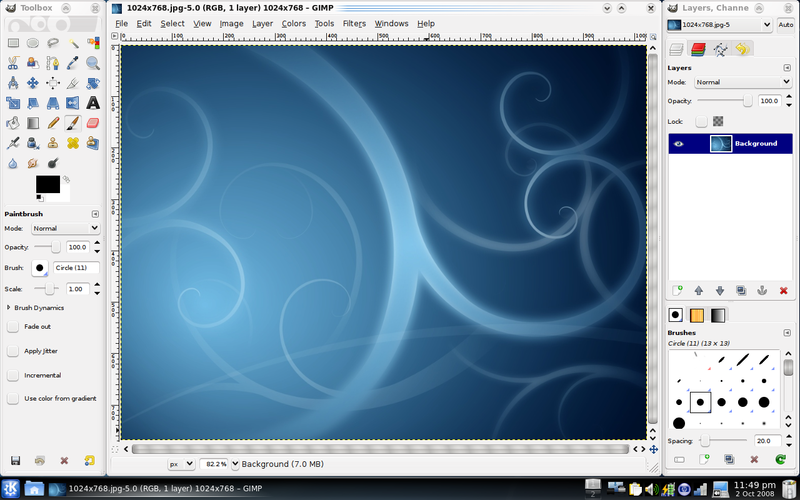
### Tools for Prototyping

Two of the most common tools used to create prototypes are Adobe Photoshop and Illustrator. These are widely used in the business world. They are very flexible tools but not the only alternatives available. The following tools are also available; many are open source and consequently free to use:

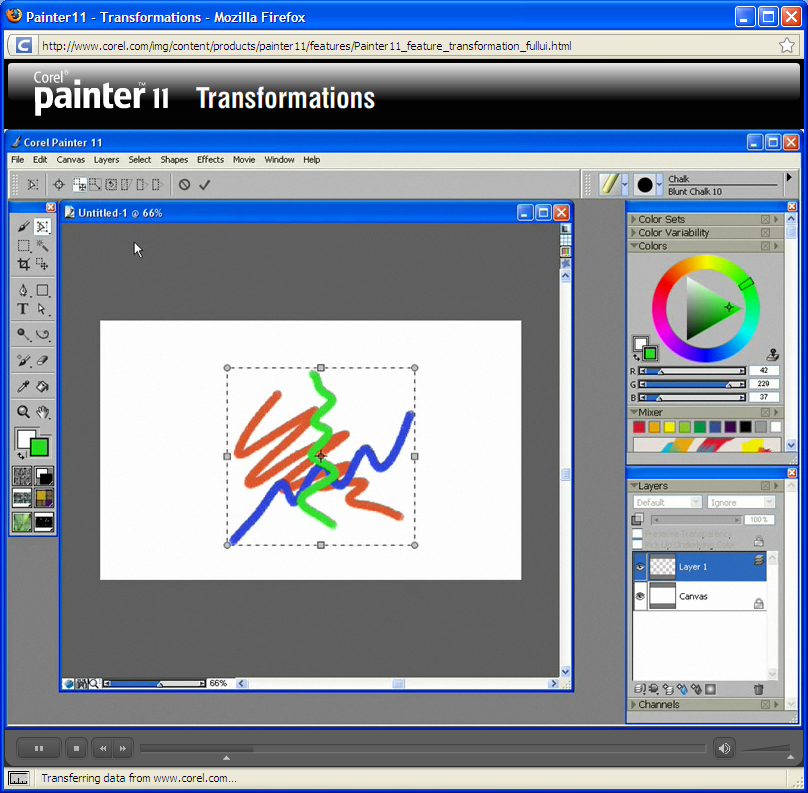
* **Pencil Project** – With the power of the underlying Mozilla Gecko engine, Pencil turns your excellent Firefox 3 browser into a sketching tool with just a 400-kilobyte installation package. Pencil is released under the GPL v3.0 and is completely free. Pencil can run on virtually all platforms that Firefox 3 supports.



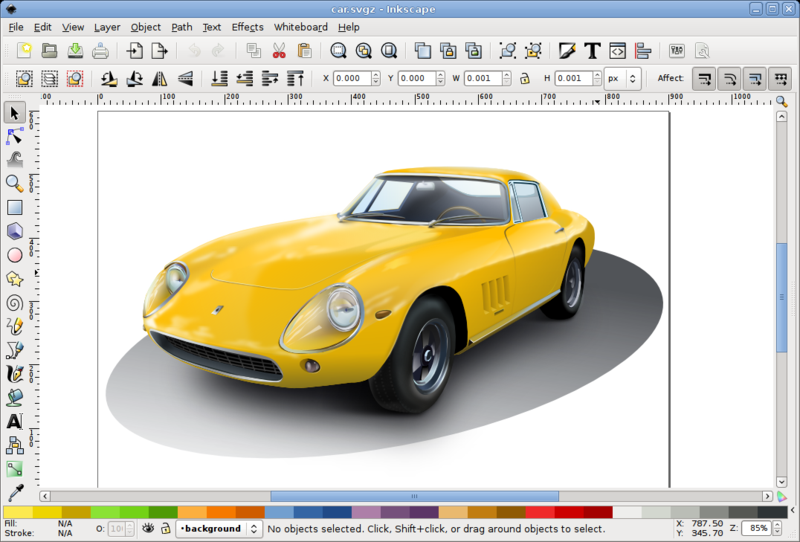
* **GIMP** (The GNU Image Manipulation Program) – A photo editor with many of the same basic functions as Adobe Photoshop. This is not a replacement but will handle most needs. The GIMP is open source and runs on all platforms. Additionally there is a version available call **GIMPShop** with menus and toolbars that have been moved and changed to mimic Photoshop’s interface as closely as possible.



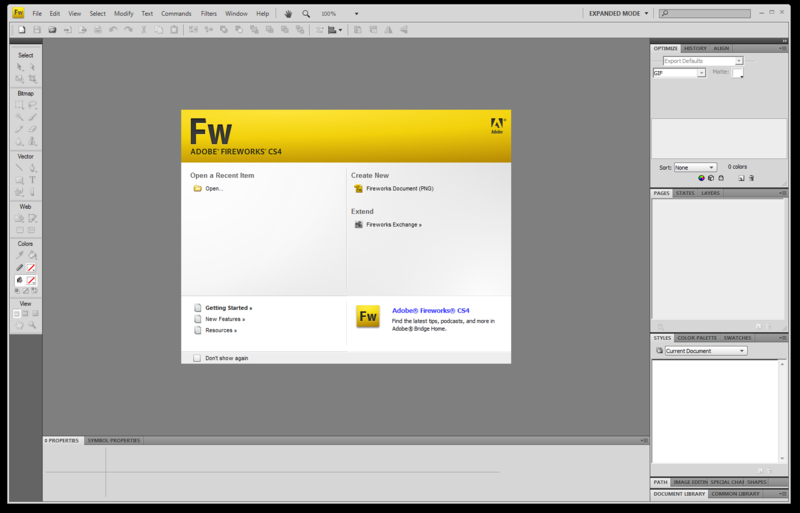
* **Corel Painter** – is a raster-based digital art application created to simulate as accurately as possible the appearance and behavior of traditional media associated with drawing, painting, and printmaking. It is intended to be used in real-time by professional digital artists as a functional creative tool.



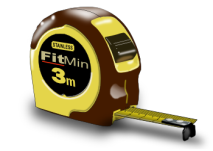
* **Inkscape** – is a vector graphics editor application. It is distributed under a free software license, the GNU GPL. Its stated goal is to become a powerful graphics tool while being fully compliant with the XML, SVG, and CSS standards.



* **Adobe Fireworks (formerly Macromedia Fireworks)** - Known as Fw for short, is a bitmap and vector graphics editor. It was originally developed by Macromedia, which Adobe acquired in 2005, and aimed at web designers (with features such as: slices, the ability to add hotspots etc.).   
    
  It is designed to integrate easily with other former Macromedia products, such as Macromedia Dreamweaver and Macromedia Flash. It is available as a standalone product or bundled with CS4. Previous versions were bundled with Macromedia Studio.



### Page Size Considerations

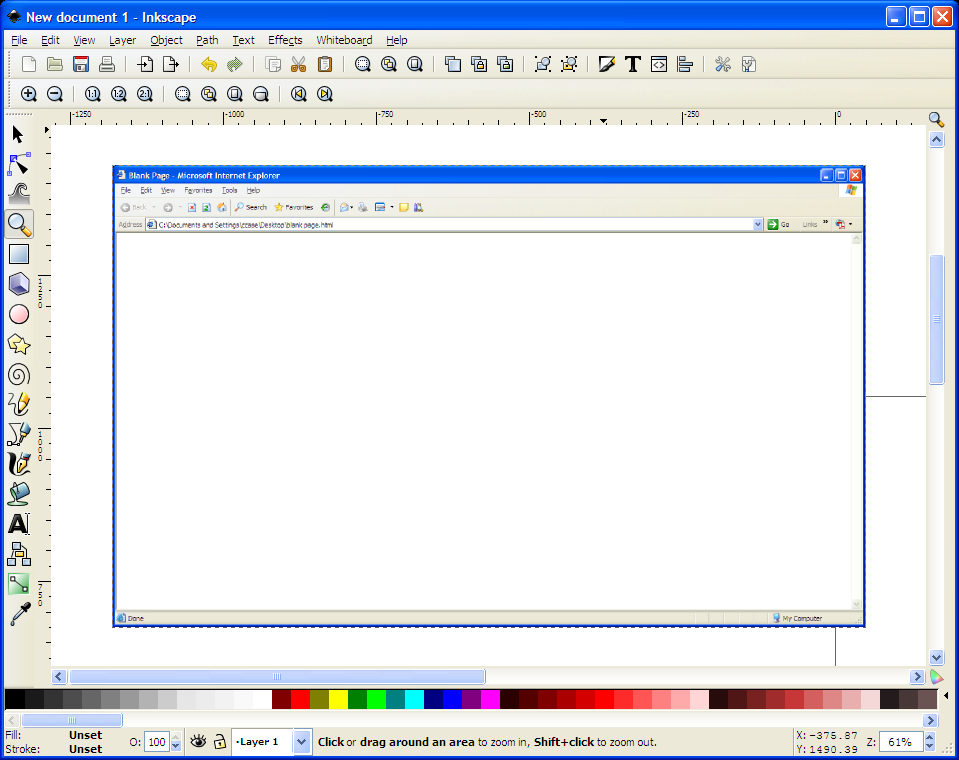
One problem in creating prototypes for web design is that your work needs to be a 1:1 representation of the final product. You need to make some initial decisions on the target size for your pages. Web sites have been historically limited to a resolution of 800 x 600 pixels. As time, technology, and monitor sizes have evolved, resolutions have been moved up to 1024 x 768. While these figures seem small, the best course of action is to design to the lowest common denominator.

To some degree size considerations can be mitigated with the use of percentage base design using Cascading Style Sheets. This will be discussed in further detail later in this guide.

### Browser Chrome

Browser Chrome is simply the browsers application interface. This User Interface consumes the amount of presentation area you have remaining for you web page, The lower the resolution, the less space for viewing your page’s design. The bottom of the viewable area is typically referred to as the “fold”. So content that can only be seen after scrolling down is referred to as being “below the fold”.

When creating compositions and prototypes, you may want to import a screenshot of an empty browser as a canvas for your sketches. This can help give you a better understanding of the perspective and visible area of the browser’s content area.



### Sketches and Wireframes

Rough out your ideas on paper before committing the time and effort to using a graphics program. The sketches don’t have to be perfect. Sketching is best for getting a very general design sense and for the geometry and balance of a page’s layout. We may have the best content or products but without a balanced and professional design the site can fail.

#### Visual Organization

According a research group with Stanford University, “We find that people quickly evaluate a site by visual design alone.” —Stanford Guidelines for Web Credibility, 2002

All content on a page has a visual weight, even text. Visual weight is the ability of content, or elements to visually draw the eye towards it. Visual weight can be created and manipulated through the use of color and/or contrast.

Page layout and design falls into one of two categories; the overall **Personality** and the **Visual Organization**.

Creating the personality or look and feel of your site is accomplished with use of colors, typefaces, images, and with shapes and patterns. Visual organization involves looking at the similarities and differences in ***what we are seeing***. All elements of your page have relationships with each other. These relationships allow us to distinguish objects while at the same time giving those objects ***meaning***.

A difference in color or size can imply two distinct objects or parts of the same object. The differences in sizes can suggest perspectives where one object is closer than another. Keeping visual traits in mind will help in developing a better understanding of what are seeing.

This practice of looking at elements of our design and their attributes plays into the **principles of perception**. These principals discuss the ways that various objects or elements are perceived in relation to other objects or elements.

These principles of perception are used to group objects in a number of ways:

|  |  |
| --- | --- |
| **Principle of Perception** | **Example** |
| **Proximity** Spatial or temporal proximity of elements may induce the mind to perceive a collective or totality | **C:\Documents and Settings\ccase\Desktop\Proximity.png** |
| **Similarity**  The mind groups similar elements into collective entities or totalities. This similarity might depend on relationships of form, color, size, or brightness. | C:\Documents and Settings\ccase\Desktop\Similarity.png |
| **Continuity** or **Continuance**  The mind continues visual, auditory, and kinetic patterns. | C:\Documents and Settings\ccase\Desktop\Continuance.png |
| **Closure**  The mind completes interpolates or extracts an object that is perceived but not literal or complete | C:\Documents and Settings\ccase\Desktop\Closure.png |

These principals are used in graphical layout and design whether it is for print, artistic media (sculpting, painting, etc.) or for web design. When applied to our web pages to address the Visual Organization, it becomes easier to balance the content on the page and create an aesthetically pleasing page. This goes beyond face value as it impacts the way the information the page is viewed and understood.



Above: Principles of Perception in use in page design

For further information on these concepts you may want to search Google for **Principles of Perception** or **Gestalt psychology**.

## Information Architecture

Information Architecture is the process of organizing your content into discernable groups. The process includes determining the number of categories for your content and what they will be called. The goal of information architecture is to make you content easy to find.

Libraries in the U.S. use classifications systems such as the Library of Congress or Dewey Decimal systems. These systems assign unique codes and locations based on their respective subjects. These classification systems work because they are so prevalent. People who have used the system know what to expect.

The books within these libraries have their own conventions and techniques for classifying and organizing the information contained within. They offer tables of contents, indexes, glossaries, bibliographical information and more.

These two systems are long established and people know what to expect. The web is another story. Classifying and organizing information can be a very large task with web content. How and where things are placed, and even what they are called can be subjective.

For your personal development exercises in classification and organization can be very straight forward. In a business environment you will need to work with all stake holders to come to an agreement on the architecture.

### Controlled Vocabularies

One of the first steps in information architecture is to develop a **Controlled Vocabulary**. A controlled vocabulary (CV) is defined as “organized lists of words and phrases, or notation systems, that are used to initially tag content, and then to find it through navigation or search.”

Controlled vocabularies are forms of **Metadata** (data about data). A controlled vocabulary is a way to display relationships between parts of your site to increase the ability to find information.

One way to look at it is with a label system. If your system is consistent and if you refer to things or concepts with the same terms every time you are using a simple controlled vocabulary.

Controlled Vocabularies can:

* Help keep your categories distinct and aids in category analysis in general.
* Define and provide you with your navigation.
* Aid in personalization features.
* Prepare for the implementation of Content Management Systems or knowledge management projects.
* Unify the language of the organization or user base that uses the system. (Better and more accurate communication.)
* Provide an organization with a better understanding of what the site covers and provides a site map.

### Taxonomies

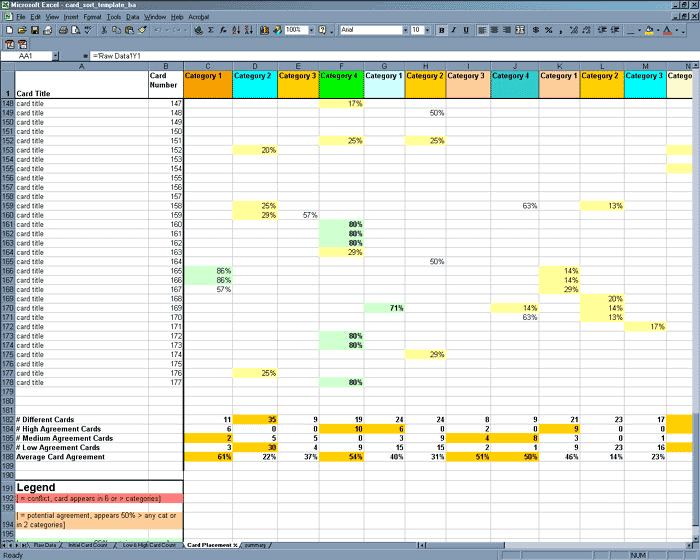
As you develop your controlled vocabularies you will start to notice the relationships between the terms you define. These relationships will turn into what’s referred to as a Taxonomy.

Taxonomies developed from the scientific world’s classifications in biology. With the use and study of Taxonomies to better understand the relationships of information, the term has been broadened to include Hierarchical relationships as a whole.

Take, for instance, a website devoted to clothing. Say we need a systematic way to map out the different types of pants available. You need to develop the hierarchy showing broader terms (BT), and the narrower terms (NT). Your controlled vocabulary will start to reveal the relationships of the information your site provides. If we were to configure this in a chart it might look something like this:

**Card Sorting**

 One method for organizing your controlled vocabulary is to perform exercises in card sorting. This method involves writing down the terms you have defined and then sorting them on the table like the tree structure above. This is a simple and quick way of exploring your organizational options. For instance, the tree above would look very different if you sorted by the casual/dress/sports nomenclature rather than by gender and age.

This exercise can also reveal things you have missed, how hard it will be to graft in new categories in the future and can also be applied in a usability study.

In business environments it is customary to have a team of users in different roles or departments perform card sorting exercises. As you record the results you can then compare and contrast the way others group information. With enough of a sample you can even develop a sort of a **Heat Map** that shows the categories grouped in the most common ways.

Card sorting can be performed in one of two ways:

**Open Card Sorting** – Give participants cards depicting content with no pre-established grouping. Ask them to sort the cards into groups they feel are appropriate and to describe each of the groups. This is useful for gathering input on new sites, products, or subjects.

**Closed Card Sorting** – Give your participants cards showing the site content with an established set of *primary* groups. Then ask them to place the cards into these groupings. Closed card sorting is good for adding new content to an existing site or for getting feedback after performing an Open Card Sorting exercise.

Card sorting does have its pro’s and con’s. Keeping the following in mind will help you know whether this technique will help you in making decisions about your design.

**Pros**

* **It’s simple** – The activity is easy for both the participants and the designer.
* **It’s inexpensive** – Typically all that’s needed are index cards, post-it notes, pens and pencils and time.
* **It’s quick** – You can run several card sorts in a short period of time yielding a lot of useful information.
* **It’s established** – The practice of card sorting has been used for over a decade and is common in the practice of information architecture.
* **It’s interactive** – Card sorting exercises involve real user feedback. It’s not limited to your bias or preference since it relies on a broader sampling of experience, understandings, and cultures.
* **It’s foundational** – While not the grand solution to all problems, it provides a solid starting point for structuring your design.

**Cons**

* **It doesn’t consider tasks** – This technique is clearly central to content. It does not take into account what the users are trying to accomplish and may result in architecture that is not usable where users are attempting to perform real tasks. The additional use of a Needs Analysis is required to be sure content is being sorted to user’s needs and that they are able to perform their tasks.
* **Results may not be consistent** – There is no guarantee that the results of the card sort might vary greatly and not yield a structure.
* **It can be overly time consuming** – While the act of card sorting is a relatively quick task, compiling and then analyzing the results can take a significant amount of time.
* **It may only capture shallow characteristics** – The participants my not consider what the purpose of the content is or how it might be used to complete a task. They may just sort it base on obvious, surface characteristics like what type of document it is.

#### When should card sorting be used?

Card sorting is a user-centered, formative technique. It should be used as an input to:

* designing a new site
* designing a new area of a site
* redesigning a site

Card sorting is not an evaluation technique and will not tell you what is wrong with your current site.

Card sorting is not a silver bullet to create an information structure. It is one input in a user-centered design process and should complement other activities such as information needs analysis, task analysis, and continual usability evaluation. It is most effective once you have completed:

* research into what users need out of the site
* a content (functionality) audit/inventory (for an existing site) or detailed content list (for a new site). For an existing site, it is crucial that the content inventory is examined carefully to include only content that is needed by users.

# Presentation Versus Functionality

## Introduction

It is the old axiom of **Form vs. Function**. This struggle for balance between the presentation or aesthetics and functionality must always be observed. In the case of web design the biggest mistakes generally happen on the design or presentation side. Many design their sites in flashy and glossy ways, but neglect the needs of the user to find information. Or their navigation systems are clumsy or not intuitive.

The presentation – functionality struggle is complicated by the technologies involved. A web designer must keep change in mind, and as such must design thing in a modular way in order to make future changes easy and less time consuming.

There are many ways to accomplish this from technological means such as Cascading Style Sheets (CSS) and scripting. Those technologies will be discussed later while this section will cover the conceptual side of the balancing act.

## Separating Design from Function

Prior to the introduction of Cascading Style Sheets, Hypertext Markup Language or HTML was exclusively used to both build the structure and content of a web page as well as to define the look and feel. HTML contains a number of defined tags such as the Bold tag (**<b>**), the Right justification tag (**<right>**), the Font tag (**<font>**) and more.

Tags like **Font** also allow you define properties like the color and typeface of the font you are marking up. This worked well for a time, but as web pages grew and as content started to change more frequently, designers found themselves having to scour through all of their code to make changes to these tags.

As scripting became more prevalent this only added to the complexity. Now a simple HTML page was full of tags that dictated the look and feel and scripting code was littered throughout the page as well. There has to be an easier way to maintain and update all of this code.

With the introduction of Cascading style sheets and the maturation of scripting and programming languages for web development, it has become possible to remove a large part of the non-HTML code from the web pages. Now design changes can be made to the CSS files and programming changes can be made to other external files that are called by the HTML page.

## Adding Dynamic Effects: Bringing Life to the Static

Before long you will find that you want your website to be more interactive, to offer the user a more personal experience, or just liven up your page. Your web pages can include elements with animated effects, audio and video, and other features. Some of these effects can had a practical purpose while others allow you to spice up the user experience. There are several technologies that you can take advantage of:

**Animated Gifs** – One of the first animated elements on the web were animated Gif images. These images are raster-based images that include an initial frame, and many additional frames. To keep the file sizes of animated Gifs down, every subsequent frame only contains the information on the pixels that changed from the previous frame. This technology is best for very small animations or banner ads.

**Dynamic HTML (DHTML)** – This technique is not really a new technology but is a mixture of JavaScript and standard HTML. Often this mixture is used to create effects like expanding menus, animated clocks, and moveable page elements.

**Flash** – Flash based content allows a very rich and dynamic user experience. Flash can be used to create whole pages, user navigation, animated banners or splash screens, interactive product consoles, and more. Flash is a very powerful tool that opens a world of possibilities.

This technology requires that your visitors have the appropriate plug-in installed in order to view and interact with the content. Best practices should be employed to offer alternative content for those who do not have the plug-ins, or are on slower connections or computers. Flash content can require high bandwidth and moderately to high powered computer systems.

**AJAX** – Much like DHTML, Ajax is not a new technology, but rather a new application of several existing technologies. Ajax stands for Asynchronous JavaScript and XML. Ajax is used with great frequency. It is in widespread use in many of Google’s web based tools such as Gmail, Google Calendar, Google Earth and more.

Ajax allows a client’s web browser to send and receive information without refreshing or navigating away from the page the user is on. For instance, when you are typing in a search keyword on Google, a list of suggested terms appears below the text field you are typing in. This list updates with every keystroke. Every time you press a key, it is transmitted to Google and a list of words is generated and sent back to your browser. This all happens without navigating away from the page.

## Pragmatism: Balancing Fad with Function

Have you ever watched one of those old TV shows that made the transition from black and white to color? Color was such a new and exciting thing. And the move to color inspired the creators of film and television to splash us with every hue that existed. If these explorations of new capabilities didn’t induce a seizure, it at least gave you a headache and caused temporary blindness!

This **TECHNICOLOR** assault went on for quite some time but while we have largely tamed that beast, the almost obsessive need to abuse new technology or capabilities will always remain. You still see it today:

* 3-D movies that spend so much time on the effects that the story suffers and the translation to the 2-D world of DVD’s land’s the multimillion dollar effort in the $5 bin at Wal-Mart.
* Buzzwords like **green**, **web 2.0**,or other marketing words creeping into every commercial and every product. Companies stretching any subject to fall under the buzzword of the day. (Remember the heights of the “Lite” marketing craze in the food industry?)
* Bandwagoning like the increase in vampire-love stories after the success of the movie Twilight. (The CW is starting a new series that is practically a clone of the main storyline in Twilight)

All of these are examples of **Fads** and not **Trends**.A fad is defined by the Business Futures Group as “…[S]hort-lived. Think Tickle Me Elmo [dolls], foam parties, and [the TV program] South Park... Trends, on the other hand, have staying power. Think the revival of interest in religion and the use of lifestyle drugs. Trends take longer to build and their effects may be felt for years, even decades.”

### Fads in Web Design

The information is valuable when we are thinking of our content, but what are we looking for when it comes to our *design*? While our content may be related to fads and trends, your web site should avoid fads and trends in design. Some of the fads of the past are more blatant that those of today. For instance, the over-use of Adobe plug ins to generate page curls, and drop shadows. Some others include:

* **Contrast Issues** such as gray typeface or other washed out colors. Particularly when it is used for main body content.
* **Splash pages**, most often consisting of Flash based content. If you are lucky they provide you with a “skip intro” link.
* **Developing** **Communities** around products. The infancy of social networking tools and sites saw the rise of businesses or individuals jumping on the Community bandwagon. This is a classic “solution looking for a problem”.
* **MIDI audio, animated page elements** like dividers, icons and other elements that distract from the main content. Embedded audio that you have no control over was / is particularly troublesome.
* **Awards** littering the entire page in an attempt to validate the worth of the site. Often these consisted of “site of the week” pats on the back with no real weight, lasting power, or worth.
* **Guest books** have gone the way of the dodo bird. These gems really offered no real purpose or value, but were often in place. Often it was the result of a page creator’s first attempts at creating and understanding web forms.

# Web “Standards”

## Introduction

In a perfect world, web standards would be understood, accepted, and implemented universally. Difference in display, functionality, and support would never be seen anywhere outside of a beta test for a new browser. Unfortunately this isn’t a perfect world.

With the creation of the web came many companies looking to make their mark and to create a perfect set of **Golden Handcuffs** locking their users to their products. The term “Golden Handcuffs” is used frequently by sales and marketing individuals to refer to a product or feature that people want that, once purchased or used, would limit either the desire or even the capability of the customer from no longer using the product or service. It is purely about retention.

In the world of the web it has created a number of serious issues and differences that web developer have to deal with and keep in mind at all times. Not having an awareness of the varying support for web standards can shut out users based on the computer they use, the operating system, and the browser. It also can contribute to a web site or page that simply doesn’t work or doesn’t look the way you intended.

This section discusses some standards and issues that must be considered when developing your web site.

## Browser Compatibility Issues

One of the fist and biggest considerations comes down to the browser the client is using to view your site. The World Wide Web has a group that maintains and governs specifications for HTML, Cascading Style Sheets (CSS), XML, and more. This group is called the **World Wide Web Consortium** or **W3C**. The W3C is an international organization where member organizations maintain full-time staff for the purpose of working together in the development of standards.

While there has been a long standing specification for HTML, companies like Microsoft and Netscape introduced new markup tags in every new version of their browsers. These tags were not added to the international standard. Reasons for this ranged from the desire to introduce new capabilities without having to wait for the W3C to approve and incorporate the new tags, to creating a sense of “stickiness” to their product.

Much of this kind of cavalier introduction of proprietary solutions has stopped but there are still differences in browser technologies that can create real headaches for developers.

## HTML, CSS, and Other Markup Languages

A **Markup language** is a system for annotating text files in a way that is distinguishable from the main text. The markup is typically acted upon by a program such as a browser or web server. The markup is omitted from the resulting document or information that is displayed to the end-user. There are three general categories of markup language:

* **Presentational markup** – This traditionally has been used by word processing systems to produce the What You See Is What You Get (WYSIWYG) view. Typically this markup is designed to be hidden from users, even authors and editors.
* **Procedural markup** – Provides instructions to programs that process text. The programs run through the text from beginning to end following the instructions provided by the markup.
* **Descriptive markup** – Used to label parts of a document rather than to provide instructions on processing.

The distinction of these types can be blurred by the fact that markup like HTML may be procedural but can also contain presentation attributes. These points are more important when attempting to separate presentation from structure and functionality but are not critical to beginners.

### Hypertext Markup Language (HTML)

Hypertext Markup Language is the core of your web design. It defines the page, and it’s content. I can contain information telling the browser *how* to display the information such as the typefaces, font sizes, and more, but it is suggested that you avoid setting presentational information in your HTML file.

HTML pages should define the *structure* of your web page. The page defines the header information describing the document and its contents, and the page’s body which includes elements like tables, images, and textual content.

If you are a beginner and you are relying on a WYSIWYG editor, you will not see this markup language unless you specifically decide to view it. You can get by not knowing the markup language, but you gain greater capabilities to correct errors, and to customize your page. Additionally you will need to be proficient in HTML if you decide to introduce scripting or custom tags.

While the HTML specification contains a large number of tags, you will find that you don’t need to know them all. Rather you will likely use a small subset of the tags available to you. The important thing to understand is the **Document Object Model** or **DOM**. The HTML DOM is a standard that defines the *structure* of the HTML document. Just as a Word file has a header, footer, and the content, an HTML document has a structure that must be followed. Many HTML Editors will give you the basic structure when you create a new file.

#### A Complete HTML 4.01 Document

An HTML 4.01 document begins with a **DOCTYPE** declaration that declares the version of HTML to which the document conforms. The **HTML** element follows and contains the **HEAD** and **BODY**. The **HEAD** contains information about the document, such as its title and keywords, while the BODY contains the actual content of the document, made up of block-level elements and inline elements. A basic HTML 4 document takes on the following form:

<!doctype html public "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

<html>  
 <head>  
 <title>The document title</title>  
 </head>  
   
 <body>  
 <h1>Main heading</h1>  
 <p>A paragraph.</p>  
 <p>Another paragraph.</p>  
   
 <ul>  
 <li>A list item.</li>  
 <li>Another list item.</li>  
 </ul>  
 </body>  
</html>

#### HTML Tags

Tags are very logical and simple. All tags are placed inside a greater that (**<**) and less than (>) symbols. Tags normally come in pairs with an opening tag and a closing tag. Any text that appears between those tags will be affected by the function the tag performs. For instance If you want bold text to appear in you document you would use a tag like the one below:

The quick brown <b>fox</b> jumps over the lazy <b>dog</b>.

This markup would produce text that looks like:

The quick brown **fox** jumps over the lazy **dog**.

In the old HTML standards, we used to use B for bold, and I for italics, etc. Best practices now suggest that you separate content from presentation. This is accomplished by setting the structure of your document in HTML, and control how it displays using a CSS file.

#### Can I Use Two Tags at Once?

Yes. Just make sure to begin and end both. Like so:

<b><i>Bold and Italic</b></i>

(gives you Bold and Italic )

If you do use multiple tags to alter text, make a point of not getting the end tags out of order. Look at this:

<b><i>Bold and Italic</b></i>

In terms of format, the example above is not correct. The file will display properly in Internet Explorer but there is no guarantee that it will in other browsers or versions of IE. Additionally it will fail an audit if you run an HTML validation tool against it.

The end tags are out of order in relation to the start tags. Always set the beginning and end tags at the same time, always placing them on the farthest end of the item being affected.

This is an example of the previous code in its corrected form:

<b><i>Strong and emphasis</i></b>

Notice the strong tags are on the far ends. Next in line are the emphasis tags. Just keep setting commands at the farthest ends each time you add them and you'll stay in good form.

#### HTML Headings

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

<h1>This is a heading</h1>  
<h2>This is a heading</h2>  
<h3>This is a heading</h3>

#### HTML Paragraphs

HTML Paragraphs are defined with the **<p>** tag.

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

#### HTML Links

HTML links (or hyperlinks) are defined with the **<a>** tag.

<a href=”http://www.google.com”>This is a link.</a>

The href entry is known as an **attribute**. Adding the **target** attribute allows you to tell the browser to open the link in the current window, or a new window.

#### HTML Images

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

<img src=”images/ml\_logo.gif” width=”100” height=”150” />

The file name and path as well as the height and width are seen as attributes of the image tag.

**Tip**HTML tags are not case sensitive (<P> means the same thing as <p>). Many web sites have upper case characters in their HTML. Generally though, it is better to always code in lower case. While it is not required now, future versions of the HTML specification will require lower case for all tags

#### HTML Attributes

HTML elements (or tags) can have attributes as you saw in previous examples. These attributes provide additional information about the element. Attributes always appear in the starting or opening tag. Attributes come in name/value pairs like: **name = ”value”**

Attributes should always be enclosed in quotes. Double quotes are the most commonly used but single quotes are allowed. Be as consistent in your usage as you can. When introducing scripting languages such as JavaScript, Active Server pages, or PHP, your attention to your quote usage is required.

#### HTML Comment tags

Comments can be added to your HTML code to make it more readable and understandable. Comments are ignored by web browsers and will not be displayed on the page. The following is an example of an HTML comment:

<!-- This is an HTML comment. The browser will ignore this line. -->

#### HTML Tables

The HTML table element or tag was created to simple display information in tables. Early on as the limitations of HTML were found, Tables were being used for far more than they were intended. This is also the case with MS Word documentation. The human ability to adapt tools capabilities to work around real or perceive limitations resulted in using tables to accomplish a higher degree of control over layout of a document.

HTML is very fluid in the way it displays content in the browser. Creating columns of text and images does not come easy. Anyone who has had to use tables in Word to accomplish a particular layout (two columns, three-column newspaper-styled layout, etc) will find that web pages are much less rigid. The code to create a table with three columns and two rows looks like this:

<table border=”1” cellpadding=”1” cellspacing=”1”>  
 <tr>  
 <td>Cell A1 (Column A, Row 1)</td>  
 <td>Cell B1 (Column B, Row 1)</td>  
 <td>Cell C1 (Column C, Row 1)</td>  
 </tr>

<tr>  
 <td>Cell A2 (Column A, Row 2)</td>  
 <td>Cell B2 (Column B, Row 2)</td>  
 <td>Cell C2 (Column C, Row 12)</td>  
 </tr>  
</table>

You may be wondering why the tag for the cells is not **<tc>** since the **<table>** tag is obvious and the **<tr>** tag is clearly a table row. The letters in the <td> tag stand for **table data** which is the content of a data cell. Data cells can contain text images, lists paragraphs, forms, horizontal rules, other tables, etc.

If you are using headers in the first row, you would use the **<th>** or table header tag. The text will be bold by default.

When using a table for layout you typically set the border, cell padding and cell spacing attributes to zero.

Table cells with no content will cause issues with the display. Depending on the browser you are using, the cell’s borders will not display. To override this issue you can fill the cell in HTML using a **non-breaking space**. In HTML this is written like this: **&nbsp;** This special character will allow the browser to display the cell properly.

#### Lists

HTML defines three different types of lists:

**Unordered Lists** – These lists mark each item with a bullet. The list uses the **<ul>** tag and the **<li>** tag. The following is the code for an unordered list:

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

This code produces the following:

* Coffee
* Milk

**Ordered Lists** – These lists mark each item numerically. The list uses the <ol> and <li> tags. The following is the code for an ordered list:

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

This code produces the following:

1. Coffee
2. Milk

**Definition Lists** – These lists contain a list of items (or terms) with a description of each item.

* The definition starts with a <dl> tag. (definition list)
* Each term starts with a <dt> tag. (definition term)
* Each description starts with a <dd> tag. (definition description)

<dl>  
 <dt>Coffee<dt>  
 <dd>Black hot drink</dd>  
 <dt>Milk<dt>  
 <dd>White cold drink</dd>  
</dl>

This code produces the following:

Coffee  
 Black hot drink

Milk  
 White cold drink

#### Forms

Forms are an area that contains form elements such as text fields and drop down selection boxes. Forms allow the user to post data to the web server for processing.

#### Input

Input tags are the most commonly used. The type of input is specified in the input tags **type** attribute. The most common types used are:

**Text Fields** – Accepts textual and numeric entry by the user.

<form>  
 First Name:  
 <input type=”text” name=”firstname”/><br>  
 Last Name:  
 <input type=”text” name=”lastname”/><br>  
  
</form>

**Radio buttons** – Allows the user to select **one** of a number of choices:

<form>  
 <input type=”radio” name=”gender” value=”male”/>Male<br>  
 <input type=”radio” name=”gender” value=”female”/>Female<br>  
</form>

**Checkboxes** – Allows the user to select one or more options from a number of choices:

<form>  
 I have a bike:  
 <input type=”checkbox” name=”vehicle” value=”bike”/>Bike<br>  
  
 I have a car:  
 <input type=”checkbox” name=”vehicle” value=”car”/>Car<br>  
  
 I have a boat:  
 <input type=”checkbox” name=”vehicle” value=”boat”/>Boat<br>  
</form>

#### The Form’s Action Attribute and the Submit Button

When the user clicks the submit button, the content of the form is sent to the server. The form’s action attribute defines the name of the file to send the content to. The file defined in the action attribute usually does something with the received input. In the example below the browser will send your entries to a page called **submitform.asp**. That page then shows you the input received:

<form name=”input” action=”submitform.asp” method=”get”>  
 User Name:  
 <input type=”text” name=”user” /> <br>  
 <input type=”submit” value=”submit”/>  
</form>

### XHTML

XHTML stands for Extensible Hypertext Markup Language. XHTML is considered an attempt to merge some of XML’s discipline with HTML’s more loose conventions. XHTML, for our purposes, is mostly a stricter version of HTML.

Generally web browsers are very forgiving when it comes to HTML pages that do not always “follow the rules”. While there are tags that are required to be “paired”, or having both an opening and closing tag, there are many other tags that are not as rigid. Forgetting to set a closing tag in those instances do not cause issues with the rendering of the page or its functionality. XHTML attempts to enforce a more strict usage.

You may be asking, if HTML is more forgiving why would I want to create a variant that requires your work to be more complete and exact? XHTML is generally employed where more automation may be used. If a technology has to work with the HTML or interpret page information, a stricter page would be required. Programming technology that interacts with web content would have to account for every anomaly which is much more complex that it would otherwise need to be.

### Cascading Style Sheets

When the web really started taking off it became clear early on that maintaining those pages took more and more time. Changes to the theme of web sites, the layout and other presentational considerations were very intensive efforts. Much of the time it resulted in designers having to start at square one. Designers and programmers found that there needed to be a separation of ***what*** is displayed (the content) from ***how*** it is displayed. Cascading Style Sheets or CSS has become the solution to this problem.

Without CSS, changing the overall look and feel of your website would often require you to edit every page in your site which involves reading through all of the code and making the changes manually.

A stylesheet **lists rules** for presentation. Each rule consists of a **selector** and **declaration block**:

selector { property: value;}

#### The Type Selector

**Type selectors** correspond to defined HTML elements or tags. Using selectors you can affect the presentation of all content using a particular tag. Let’s say we want to define the presentation of all paragraphs (<p>) in out page or site:

p {  
 color: #1662d8;  
 background-color: #ffffff;  
 padding: 10px;  
 border: 5px solid #1bc8fe;  
 }

This code produces the following:

|  |
| --- |
| Here is an example paragraph with a styled font, color, padding and border. |

#### The Class Selector

**Class selectors** allow you to define custom styles that are not specifically tied to HTML tag types like type selectors are. Class selectors allow you to override the effects of type styles that have been defined. For instance, if you want a paragraph to appear with a different style from all other paragraphs, you would define a class and then apply that class to the paragraph in question:

<p>  
 This is a normal paragraph, nothing   
 exciting going on here  
</p>  
  
<p class=”alert”>  
 But this is a really important  
 paragraph – pay attention!  
</p>

These paragraphs will take on different styles when they have been defined in CSS. If they are not, the class attribute will be ignored. So let’s define the **type selector** and the **class selector** for these two paragraphs:

p {  
 color: gray;  
 font-size: 12px;  
 }  
  
p.alert {  
 color: red;  
 font-size: 18px;  
 font-weight: bold;  
 }

This code produces the following:

|  |
| --- |
| This is a normal paragraph, nothing exciting going on here.  **But this is a really important paragraph – pay attention!** |

#### The ID Selector

The ID selector allows you to change the styles of an HTML element based on an ID attribute:

**HTML**

<ul id=”contents”>  
 <li>Chapter 1</li>  
 <li>Chapter 2</li>  
 <li>Chapter 3</li>  
</ul>

**CSS**

#contents {  
 font-weight: bold;  
 font-size: 18px;  
 }

#### Descendant Selectors

Selectors that are nested in other tags are referred to as **descendant selectors**. Take the following example:

<p>  
 This is a stand-alone paragraph.  
</p>  
  
<blockquote>  
 <p>  
 A paragraph inside our blockquote.  
 </p>  
 <p>  
 And another blockquoted paragraph.  
 </p>  
</blockquote>

It is possible in this case to apply styles to all paragraphs that are nested in blockquotes without affecting others. This is accomplished by creating a style for descendant selectors:

blockquote p {  
 color: blue;  
 font-weight: bold;  
 border-left: 3px solid blue;  
 padding-left: 10px;  
}

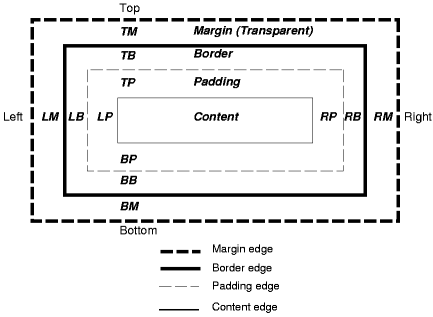
This code produces the following:

|  |
| --- |
| This is a stand-alone paragraph.  A paragraph inside our blockquote.  And another blockquoted paragraph. |

#### The Box Model

The CSS **box model** describes rectangular boxes that are generated for elements (including tags) in the document and laid out according to the visual formatting model.

Each box has a content area (e.g., text, an image, etc.) and optional surrounding padding, border, and margin areas; the size of each area is specified by properties defined below. The following diagram shows how these areas relate and the terminology used to refer to pieces of margin, border, and padding:



The margin, border, and padding can be broken down into top, right, bottom, and left segments (e.g., in the diagram, "LM" for left margin, "RP" for right padding, "TB" for top border, etc.).

The perimeter of each of the four areas (content, padding, border, and margin) is called an "edge", so each box has four edges:

**content edge or inner edge**

The content edge surrounds the rectangle given by the width and height of the box, which often depend on the element's rendered content. The four content edges define the box's content box.

**padding edge**

The padding edge surrounds the box padding. If the padding has 0 width, the padding edge is the same as the content edge. The four padding edges define the box's padding box.

**border edge**

The border edge surrounds the box's border. If the border has 0 width, the border edge is the same as the padding edge. The four border edges define the box's border box.

**margin edge or outer edge**

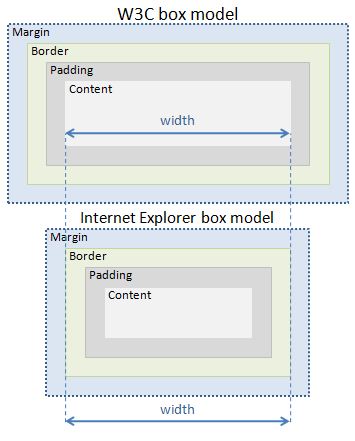
The margin edge surrounds the box margin. If the margin has 0 width, the margin edge is the same as the border edge. The four margin edges define the box's margin box.

Each edge may be broken down into a top, right, bottom, and left edge.

The dimensions of the content area of a box — the content width and content height — depend on several factors: whether the element generating the box has the 'width' or 'height' property set, whether the box contains text or other boxes, whether the box is a table, etc. Box widths and heights are discussed in the chapter on visual formatting model details.

The background style of the content, padding, and border areas of a box is specified by the 'background' property of the generating element. Margin backgrounds are always transparent.

The following illustration shows the W3C “Web Standards” box model and the way Microsoft Internet Explorer interpreted the standard. This single misinterpretation has created a world of workarounds and hacks to ensure that Internet Explorer is rendering the page in the same way as other browsers.



**File based CSS** – One or more files contain the styles used throughout your web pages. Defined HTML tags such as headers (<h1></h1>), table rows (<tr></tr>) can be defined directly or you can apply a defined class from the CSS file:

<h1 class=”myStyle”><h1>

**Inline CSS** – Styles can be defined manually inside the HTML. This method allows you to define “one-off” or custom styles where you want to override a defined style, or do not want to include a full CSS file.

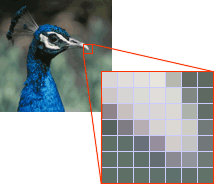
While this method has its place and uses, it does not separate presentation from structure. In effect it is reintroducing the problem associated with the presentation code in HTML. In wide use it defeats the purpose of cascading style sheets. The following is an example of a paragraph with an inline style being used to override the default style it would otherwise display:

<p style=”magin-left: 10px; font-family: arial; font-size: 10;”>  
  
 Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.  
  
</p>

## File Formats and Format Restrictions

You’ve been told about the endless possibilities, now you hear about the endless limitations, or so it seems. You really do have an enormous amount of flexibility in creating your web sites and designs, but there are some things to be aware of.

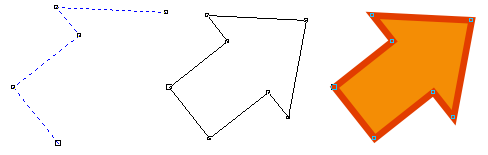
### Bitmaps vs. Vector

**Bitmap** graphics are composed of pixels. Each pixel contains specific color information. A pixel is minutely small; a single image may be composed of hundreds of thousands of individual pixels. Much like cells revealed from a piece of tissue when seen under a microscope, these pixels are only clearly and individually visible when the image is magnified.

A graphic composed entirely of pixels each with its own color properties is ideal for photographic images where there are thousands, even millions of different colors. Complex fills, shading and gradient effects can easily be rendered. The Bitmap image offers as much freedom as an empty canvas.

When a Bitmap graphic is saved, the computer is really saving an exact visual picture of the image: this pixel goes here and is this color; this pixel goes there and is that color, and so on and so on.

Making an existing Bitmap smaller requires the reduction of pixels by removing them from the image until it fits the new size. An image can be sized smaller repeatedly and still maintain the same quality, up until the point where there are not enough pixels available to reproduce the image clearly.

**Vector** graphics consist of points, lines, and curves which, when combined, can form complex objects. These objects can be filled with solid colors, gradients, and even patterns.

Vector graphics are mathematical creations. Programs that are used to create vector images save **instructions** on how the image should be drawn, rather than how it looks. Because the computer has a description of how the image should look, it can be redrawn at any size, in any position, without losing any quality

Vector graphics are ideal for producing artwork which frequently needs to be presented in different sizes or colors. Logos especially fall into this category. A logo produced with a vector application can be blown up to fit on a billboard or scaled down to adorn a letterhead with no loss of quality.

### Lossy and Lossless Compression

**Lossy** compression simply eliminates "unnecessary" bits of information, tailoring the file so that it is smaller. This type of compression is used a lot for reducing the file size of bitmap pictures, which tend to be fairly bulky.

**Lossless** compression is based on the idea of breaking a file into a "smaller" form for transmission or storage and then putting it back together on the other end so it can be used again.

### Reducing Image File Sizes

Reducing the file sizes of the images and graphics of your website will dramatically improve the load times of your pages. It also reduces the amount of bandwidth you are using. To keep the sizes of your files down, try the following:

* Use lossy formats like Jpeg for photographs or images where there are a lot of transitions in the colors.
* Use the Gif image file type for line art, comics, or images with a reduced palette.
* When editing images in your graphics editors, take advantage of tools that reduce the number of colors in the palette of your image. Many times, reducing the number of palette colors will significantly reduce the file size, and load times for your graphics without a dramatic or even noticeable change in image quality.

This page intentionally left blank

# Web Programming

## Introduction

The key to unlock the full potential of your site lies in web programming. There are hundreds of tools, platforms, and solutions. They range in complexity, scope, and cost. You can build your own test server or host it from home, or you can develop the next Amazon.com.

The key is moving beyond simple presentation and structure provided by CSS and HTML, or other markup languages and to move into scripting or complete application development.

The need to process and respond to user submitted forms quickly outgrew the Common Gateway Interface or CGI solutions which traditionally involved working with a system administrator and programmer to develop, and install the program.

Scripting technologies like Active Server Pages and PHP open up your ability to customize your website. Scripting, along with a database gives you the ability to store, and retrieve any kind of information the user may need. It is a powerful solution.

## Processing Web Forms

Whether it’s a guest book, or a fully featured shopping cart application, web forms allow the user to post and request information from the web server. Web servers, in turn, interact with database servers that store the information. There are several methods that can be used to process user input.

## CGI and Scripting Languages

In the most basic form a web page is served in the following fashion:

1. A user enters an URL which is looked up by a DNS server. The server directs the browser or application to the web server requested.
2. The web server calls up the requested web page and simply transmits the page to the user’s browser or application.
3. The browser or application interprets the HTML page and displays it to the user.



In web programming the server takes an active role. HTML pages with embedded scripting must be processed by the server before it sends the final web page to the client.

In this model, the scripting is referred to as **Server-Side** programming. That is, the code is processed by the server and the results replace the code. The user sees a simple HTML page without the processing code.



Scripting that occurs on the user’s system is referred to as **Client-Side** programming. Client side almost always consists of JavaScript code that is visible in the final HTML page or in a **.js** file that the browser downloads when rendering the page.

The example above demonstrates the processing of a PHP embedded HTML document. There are many different programming solutions and PHP is only one of them. Script appears within the HTML file stored on the web server. The code is only acted upon when the user requests the page.

#### Active Server Pages (Classic ASP, ASP 1.0)

Active Server pages, otherwise known as Classic ASP, is a Microsoft scripting language for generating dynamic web content. It is supported by default on Microsoft’s Internet Information Server (IIS).

Classic ASP source code consists of VBScript which is loosely based on Visual Basic. This technology has largely been superseded by ASP.NET which is integrated into the Visual Studio suite.

The Apache web server also contains some limited support for ASP but is limited in compatibility and features.

**PHP: Hypertext Preprocessor** or **PHP** is a general purpose scripting language for web development to create dynamic web pages. PHP source code is embedded in HTML and must be processed by a web server. The resulting page is a standard HTML page with the scripting processed leaving only the results and the HTML. PHP runs on all operating systems.

#### Scripting Examples

Blocks of PHP scripting fall between a start character set **<?php** and a closing character set **?>**. ASP’s opening and closing characters are similar. The opening set is **<%** and the closing is **%>**. In either case, when the server retrieves the file request for processing, it sees these tags and understands that what fall between the opening and closing characters consists of scripting code it must act upon.

Consider the following HTML file:

<html>  
 <body>  
 **<%  
 response.write "<h2>You can use HTML tags to format the text!</h2>”  
 %>  
  
 <%  
 response.write "<p style='color:#0000ff'>”  
 response.Write “This text is styled with the style attribute!</p>"  
 %>**  
 </body>  
</html>

The code above is essentially a full html file. When requested, the server reviews the contents of the file prior to sending it to the user. If it finds blocks of code like the ASP above, it will process the statements before it returns anything to the user. The statements above simply tell the server to write the HTML listed and to send the altered file to the browser. Once done the code is completely replaced by the HTML listed.

The code above produces the following results:

## You can use HTML tags to format the text!

This text is styled with the style attribute!

And the HTML that appears in the user’s browser now looks like this:

<html>  
 <body>  
  
 <h2>You can use HTML tags to format the text!</h2><p style='color:#0000ff'>This text is styled with the style attribute!</p>  
  
</body>  
</html>

## Databases

Databases come in any number of types, sizes, capabilities, and platforms. Understanding **Structured Query Language** or **SQL** is required if you want to interact with any of them.

SQL is generally a uniform way of retrieving, updating, adding, or deleting data from a database. Every database system has variations in the language or syntax so statements written for one engine won’t necessarily work for another. While frustrating, the differences are small, but you must be aware that they exist.

### Parts of a Database

Many terms to describe parts or a database are often misused. The term **database**, many times, is used interchangeably with **table**. In casual conversation for a single data source this is fine. But in more complex systems you should be more formal and fully understand the parts and concepts of a database.

**Database** – A collection of system and user-defined tables.

**Table** – A set of records arranged in rows and columns

**Field** – A single point of data that resides in a cell, or intersection of a row and column.

### Relational Database Management Systems (RDBMS)

Two of the most common database systems you will find are Microsoft’s SQL Server (MS SQL), and MySQL. Hosting providers that have both Windows and Linux/Unix packages will have both of these options. Often, on Microsoft servers you can choose MS SQL and/or MySQL. On Unix and Linux systems, you will not have the MS SQL option.

MySQL and MS SQL are very powerful systems. Data is stored in the form of tables and the relationship among the data is also stored in the form of tables.

### Normalization of Data

Normalization is a practice of designing and populating data in a way that is best suited to searching and free from duplicated information between tables or disconnects in the way data is stored or referred to.

As an example, let us say we have two tables of data. A list of staff members and their contact information, and in a second table, we have shipping addresses for our customers. It is inefficient to store information on cities, states, or postal codes in each table. During the design of the database, the tables would point to individual tables for cities, states, or postal codes. In Microsoft Access, these tables would be referred to as lookup tables.

Normalization covers a number of other factors such as:

* Free the database of modification anomalies
* Minimize redesign when extending the database structure
* Make the data model more informative to users
* Avoid bias towards any particular pattern of querying

### Other Data Sources

Large and complex database systems are not your only option. Many Unix, Linux, and even Microsoft web servers offer lite alternatives. Additionally, these web servers are able to read and write XML files as if they were databases. Microsoft servers can even treat an Excel spreadsheet like a database since those drivers are already loaded and available.

### Retrieving Data with Scripting

In the example below, the server reads this ASP embedded HTML page, builds a connection to a data source and retrieves the contents that match the SQL query.

<html>  
 <body>  
 **<%  
 set conn = Server.CreateObject("ADODB.Connection")  
 conn.Provider = "Microsoft.Jet.OLEDB.4.0"  
 conn.Open(Server.Mappath("/db/northwind.mdb"))  
 set rs = Server.CreateObject("ADODB.recordset")  
  
 sql= "SELECT Companyname, Contactname “ & \_  
 “FROM Customers WHERE CompanyName>'E'"  
  
 rs.Open sql, conn  
 %>**  
  
 <table border="1" width="100%">  
 <tr>  
 **<%  
 for each x in rs.Fields  
 response.write("<th>" & x.name & "</th>")  
 next  
 %>**

</tr>  
   
 **<%  
 do until rs.EOF  
 %>**  
 <tr>  
 **<%  
 for each x in rs.Fields  
 %>**  
   
 <td>**<%Response.Write(x.value)%>**</td>  
 **<%  
 next  
 rs.MoveNext  
 %>**  
 </tr>  
 **<%  
 loop  
   
 rs.close  
 conn.close  
 %>** </table>  
 </body>  
</html>

#### So What’s Happening?

The block of ASP code that is between the <body> tag and the start of the <table> creates a connection object, calls on the Microsoft Jet ODBC database driver, and then opens an Access database file.

Next the code creates a record set object to store the results of the query. Then the SQL is created to return information. This query is asking for the contact and company names for any companies that start with the letter **E**. Once the SQL is executed, the results are stored in the record set object (**rs**).

The second block of ASP code that is in the first row (**<tr>**) steps through each record and gets the names of each of the fields using a **For Next** statement. (Companyname and Contactname) The subsequent block of code then loops through each record and pulls the values for each of the fields and displays the data. The **Do Until** loop will keep running until it reaches **EOF** or **End of File**.

Finally, after the data is displayed, the recordset (**rs**) and connection (**conn**) objects are closed.

### The SQL SELECT Statement

The **SELECT** statement is used to select data from a database. The result is stored in a result table, called the result-set. SQL is not case sensitive. SELECT is the same as select.

#### SQL SELECT Syntax

SELECT column\_name(s)  
FROM table\_name

and

SELECT \* FROM table\_name

### The WHERE Clause

The **WHERE** clause is used to extract only those records that fulfill a specified criteria.

#### SQL WHERE Syntax

SELECT column\_name(s)  
FROM table\_name  
WHERE column\_name operator value

#### WHERE Clause Example

The "Persons" table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **P\_Id** | **LastName** | **FirstName** | **Address** | **City** |
| 1 | Hansen | Ola | Timoteivn 10 | Sandnes |
| 2 | Svendson | Tove | Borgvn 23 | Sandnes |
| 3 | Pettersen | Kari | Storgt 20 | Stavanger |

Now we want to select only the persons living in the city "Sandnes" from the table above. We use the following SELECT statement:

SELECT \* FROM Persons  
WHERE City='Sandnes'

The result-set will look like this:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **P\_Id** | **LastName** | **FirstName** | **Address** | **City** |
| 1 | Hansen | Ola | Timoteivn 10 | Sandnes |
| 2 | Svendson | Tove | Borgvn 23 | Sandnes |

This page intentionally left blank

# Appendix A: Design Resources

## Introduction

The resources that appear here are only a small number of tools and tips available to your on the web. They have many different aims and goals and therefore should not be considered the only source or the final authority on developing for the web.

## Design and Layout Websites

[](http://www.alistapart.com/)   
(www.alistapart.com)  
**“For people who make websites”**A List Apart Magazine (ISSN: 1534-0295) explores the design, development, and meaning of web content, with a special focus on web standards and best practices.

**Steal our code? Copy our content?**ALA’s content is protected by copyright shared jointly by the magazine and its writers, but our source code is freely available to all. We also welcome translation.

**Maybe you can be one of us...**...the few, the proud, the ALA contributing authors. A List Apart is written by the community it serves: designers, developers, architects, producers, project managers, and assorted specialists. Publishing in ALA confers prestige and has helped some of our authors gain book deals or find favor with the editors of print magazines. Interested in writing for us? See the Contribute page for guidelines.

[Boxes and Arrows](http://boxesandarrows.com/)  
(www.boxesandarrows.com)

Boxes and Arrows is devoted to the practice, innovation, and discussion of design; including graphic design, interaction design, information architecture and the design of business. Since 2001, it’s been a peer-written journal promoting contributors who want to provoke thinking, push limits, and teach a few things along the way.

If you’re interested in improving the way information architecture is done, if you find yourself sparking provocative conversation on interaction design topics in your spare time, and if you go out of your way to help everyone in your office think differently about everything from the design process to software, we want to work with you.

[](http://www.smashingmagazine.com/)  
(www.smashingmagazine.com)

Founded in September 2006, Smashing Magazine delivers useful and innovative information to Web designers and developers. Our aim is to inform our readers about the latest trends and techniques in Web development. We try to convince you not with the quantity but with the quality of the information we present. We hope that makes us different. Smashing Magazine is, and always has been, independent.

Perhaps the most remarkable yet overlooked aspect of the design community is its friendly, enthusiastic spirit. Every day, literally thousands of talented, hard-working folks out there gain new insight from their work, come up with brilliant ideas and then share their experience with fellow designers.

Nourished by the gratitude of its benefactors and powered by the reach of social networking, this community has produced a wide variety of high-quality articles, resources and tools, available to everybody. Every single contribution supports the entire community, and the community supports these contributors with traffic and word-of-mouth advertising: the networking effect at its best.

[](http://www.webpagesthatsuck.com/)  
([www.webpagesthatsuck.com](http://www.webpagesthatsuck.com))

Looking at bad web site design is valuable because it gives us the opportunity to learn from other people's mistakes without having to make them ourselves.

Winston Churchill once said, "All men make mistakes, but only wise men learn from their mistakes." If you go through this site and learn from all the mistakes you see, you will be a very, very wise web designer and be able to take a bad web site and change it into a good web site.

Great web design is an art and occurs when design and content are seamless and you don't notice its greatness. With great web design, it's easy to find the information you need. The content makes you want to return again and again and, most importantly, great design gives credibility to the company/organization.

## Books

**<designing web graphics.4>**by Lynda Weinman

Publisher: New Riders Press; 4th edition (January 10, 2003)   
ISBN-10: 0735710791  
ISBN-13: 978-0735710795  
512pp

**From the Back Cover**The most influential web design book ever written, completely updated to cover motion graphics, broadband interactive design, and more. Since it was first published in 1995, designing web graphics has been the seminal resource for web designers to learn the basics and then the nuances of solid design for the web.

Lynda Weinman has been updating the book to reflect changes in the technologies affecting web design, but dwg.4 has been largely rewritten from the ground up. Included is coverage of motion graphics made possible by the Flash phenomenon, broadband-enabled graphics issues, usability, and more. Lynda is rewriting the book so that all coverage of specific tools is focused on the essential functionality of these programs (Photoshop, Dreamweaver, Flash, etc.) and not on version-specific attributes of the software, making the book relevant longer for more users.



**Creating Killer Websites**  
by David Siegel

Publisher: New Riders  
Pub. Date: October 1997  
ISBN-13: 9781568304335  
320pp

**Synopsis**Creating Killer Web Sites was the first true design book for the web. It shows, in practical terms, the fundamentals of design applied to the web. Now completely updated! Every chapter has new material! In this expanded second edition, you'll find the techniques and principles you need to build sites for today's 4.0 and the coming 5.0 browsers. Much of the book has been rewritten with new tips, new tools, updated HTML, and an emphasis on cross-browser compatibility. There are three new chapters on strategies designers will need to know going forward, including a detailed style-sheet tutorial.

This page intentionally left blank

# Appendix B: Code and Scripting Resources

## Introduction

Whether you want to simple learn more about HTML, or you want to explore the widest range of web languages and technology, this section will provide a number of resources to streamline your learning and whet your feet to all of the possibilities. From Active Server Pages, to PHP, from XML to JavaScript you can find almost everything online before ever needing to buy a book. For your convenience some book options are listed as well.

## Tutorials and Reference Websites

[W3Schools.com](http://www.w3schools.com/)  
(www.w3schools.com)

This site is one of the widest and most comprehensive sites for just about any web based development language or tool….and it’s free! W3shools provides tutorials, reference documentation, and examples. W3shools also offers certification for a number of subjects. Some subjects they provide include:

* HTML / XHTML
* Cascading Style Sheets (CSS)
* TCP/IP
* Active Server Pages (Classic ASP/ASP.NET)
* PHP, AJAX
* SQL, MySQL, ADO
* Flash, SVG
* And more….

W3schools contains learning information for beginners through professional developers. In many cases there are even “Try it now” examples to allow you to change code and see the results of your changes in real time. This is one of the most powerful and frequently used sites in my toolset.

### Code Repositories

  
(www.hotscripts.com)

Hot Scripts is a comprehensive Web directory of more than 41,000 web development and programming-related resources. Hot Scripts serves webmasters, developers, business owners, marketers, coders and programmers, who are looking to build, enhance or improve their Web sites and intranets.

Hot Scripts is filled with dynamic development tools, tons of programming scripts for more than 16 programming languages as well as informational resources including books and tutorials. With hundreds of thousands of Web development and programming resources available today, Hot Scripts' mission is to provide you with a single destination to get the highest quality and hottest performing programming resources.

## [Head First PHP & MySQL (Head First Series) by Lynn Beighley: Book Cover](http://search.barnesandnoble.com/booksearch/imageviewer.asp?ean=9780596006303)Programming Books

**Head First PHP & MySQL (Head First Series)** by Lynn Beighley, Michael Morrison

Publisher: O'Reilly Media, IncorporatedPub. Date: December 2008ISBN-13: 9780596006303774pp

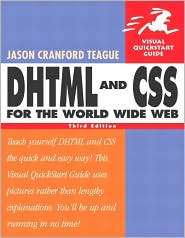
**Synopsis**

If you're ready to create web pages more complex than those you can build with HTML and CSS, Head First PHP & MySQL is the ultimate learning guide to building dynamic, database-driven websites using PHP and MySQL. Packed with real-world examples, this book teaches you all the essentials of server-side programming, from the fundamentals of PHP and MySQL coding to advanced topics such as form validation, session IDs, cookies, database queries and joins, file I/O operations, content management, and more.

Head First PHP & MySQL offers the same visually rich format that's turned every title in the Head First series into a bestseller, with plenty of exercises, quizzes, puzzles, and other interactive features to help you retain what you've learned.

Use PHP to transform static HTML pages into dynamic web sites Create and populate your own MySQL database tables, and work with data stored in files Perform sophisticated MySQL queries with joins, and refine your results with LIMIT and ORDER BY Use cookies and sessions to track visitors' login information and personalize the site for users Protect your data from SQL injection attacks Use regular expressions to validate information on forms.Dynamically display text based on session info and create images on the fly. Pull syndicated data from other sites using PHP and XML.

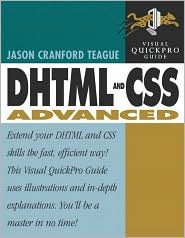
Throughout the book, you'll build sophisticated examples -- including a mailing list, a job board, and an online dating site -- to help you learn how to harness the power of PHP and MySQL in a variety of contexts. If you're ready to build a truly dynamic website, Head First PHP & MySQL is the ideal way to get going.

**[](http://search.barnesandnoble.com/booksearch/imageviewer.asp?ean=9780321199584)DHTML and CSS for the World Wide Web: Visual QuickStart Guide**   
by Jason Cranford Teague

Publisher: Peachpit Press  
Pub. Date: February 2004  
ISBN-13: 9780321199584  
630pp

**Synopsis**

If you can't afford to let the Web get ahead of you, you can't afford not to have this guide. In this best-selling Visual QuickStart Guide, you'll find all the friendly, step-by-step instructions you need to start using DHTML and CSS to add visually sophisticated, interactive elements to your Web sites. Completely updated to cover the new browsers, standards, and DHTML and CSS features that define the Web today, the one thing that hasn't changed in this edition is its task-based visual approach to the topic.

[](http://search.barnesandnoble.com/booksearch/imageviewer.asp?ean=9780321266910)Using loads of tips and screenshots, veteran author Jason Cranford Teague covers a lot of ground--from basic and advanced dynamic techniques to creating effects for newer browsers, migrating from tables to CSS, and creating new DHTML scripts. If you're new to DHTML and CSS, you'll find this a quick, easy introduction to scripting, and if you're a more experienced programmer, you'll be pleased to find practical, working examples throughout the book.

DHTML and CSS Advanced: Visual QuickPro Guide   
by Jason Cranford Teague

Publisher: Peachpit Press  
Pub. Date: December 2004  
ISBN-13: 9780321266910  
400pp

**Synopsis**

Once you discovered the dynamic, sophisticated sites you could create using DHTML and CSS, you wanted more--more interaction, more accessibility, more animation, more interesting combinations of text and graphics, more everything!

This book provides the key. By picking up where his enormously popular Visual QuickStart Guide left off, best-selling author Jason Cranford Teague explores the advanced DHTML and CSS programming techniques you need to take your Web sites to the next level.

Covering everything from standards and accessibility to separating content from style; creating multicolumn layouts; combining text and graphics; creating controls, menus, forms, and special effects; working with other Web languages; and much more, this friendly, task-based volume is a must-have if you're serious about Web design.

Step-by-step instructions and a strong visual format make complex topics understandable, even fun--especially when coupled with the loads of tips, screen shots and practical, working examples that Jason has provided.

# Appendix C: Checklists

### Check List 1: 148 ways you're killing your site

#### First Impression / Big Picture

* We've designed our site to meet our organization's needs (more sales/contributions) rather than meeting the needs of our visitors.
* Our site tries to tell you how wonderful we are as a company, but not how we're going to solve your problems.
* It takes longer than four seconds for the man from Mars to understand what our site is about.
* The man from Mars cannot quickly find the focal point of the home page.
* The man from Mars cannot quickly find the focal point of the current page.
* Our site doesn't make us look like credible professionals.
* Our site doesn't make visitors feel they can trust us.
* Our home page — or any page — takes more than four seconds to load.
* Quickly scanning the page doesn't tell our visitors much about its purpose.
* We don’t put design elements where our visitors expect them.
* We have not eliminated unnecessary design items.
* We don't know which design items are not necessary.
* Our site breaks when visited with the JavaScript turned off.
* Our site breaks because of back-end coding errors.
* We say "Welcome to..." on our home page.
* Our site is Flash-based (and this is what our site looks like to people without Flash.)
* Our site's navigation is Flash-based.
* Our site uses a splash page (unless it's a liquor, porn, gambling, adult, tobacco, or a multi-lingual / multinational site).
* Our site makes visitors register before they can enter.
* Our site uses two or more splash pages.
* Our site's TITLE tag is something like "New Document", "Index" and not the name of your company or other search-engine friendly terms.
* Our site has a sound file automatically play in the background when a web page loads, but we're not a record label or musician.
* I don't know if our site looks the same in the major browsers.
* Our site doesn't look the same in different browsers.
* The important content does not fit in the first screen.
* Our pages have too much/too little white space.
* Our site uses pop-up windows.
* Our site forces visitors to install weird plugins.
* Our site has "Download latest browser" text or buttons.
* Our site prominently displays what hardware and software was used to create the site.
* Our site's design was "borrowed" from another site.
* Our site doesn't provide clear instructions on how to perform tasks like ordering, filling out forms, etc.
* Our site disables a visitors right-click mouse button because we're crazy enough to think we have content worth stealing and that our visitors are too stupid to figure out how to bypass our code.
* Our site is based on a template that's bloated with ugly code, is difficult to maintain, and is, quite frankly, broken.
* We don't identify PDF files with an icon.
* We don't analyze our log files.
* We never conduct user testing.

#### Text and Links

* Our site mixes and matches text sizes on the page.
* Our site mixes text colors on the page.
* Our site's text requires people to have special (unusual) fonts on their computers to correctly view our text.
* Visitors can't read our text because it's too small.
* Our SITE USES LOTS OF WORDS IN ALL-CAPS.
* Our site uses scrolling, blinking, fading, or moving text.
* The color contrast analyzer says there isn't enough contrast between text/links and the background.
* **AccessColor** says there isn't enough contrast between our stylesheet and our page. (<http://www.accesskeys.org/tools/color-contrast.html>)
* Our site uses centered text on more than just headlines.
* We use justified text.
* We use browser-specific tags like <MARQUEE>.
* We use font faces that are not appropriate for our audience — like Comic Sans on a senior citizen site.
* Our site has text in the status bar — moving Javascript text.
* We have too much/too little text on a page.
* Our site uses underlined text. (Only links should be underlined.)
* Our site has sideways text.
* Visited links don't change color.
* Our links are not clearly labeled.
* Our site has too many links in one area.
* Our site has too many links.
* Our site has lots of dead links.
* Our site has lots of dead links and/or no 404 pages.
* Our site has lots of complex URLs.
* Our site has links consisting of 10-20 words.
* Our links are not informative.

#### Graphics, Video, Audio

* Our logo does not look like it was professionally made.
* Our logo is a bad scan of a business card.
* Logo is not above the fold. (Yes, this does happen.)
* Logo is not on the top of every page and clicking it doesn't lead to the home page.
* If your site has banner ads (especially near the top of the page), keep graphics away from them. People tend to ignore ads and they'll ignore your graphics.
* Our site uses cheap clip art instead of high-quality web graphics.
* Our site uses divider bars.
* Our site uses large (file-size) graphics.
* Our site uses graphics that detract from the page.
* Our site automatically loads movies instead of using YouTube's method of only showing movies when people click.
* Our site uses a trailing cursor.
* Our site uses IE page transitions.
* Our site uses "Under Construction" graphics.
* Our site doesn't physically reduce graphics using Photoshop (or other program). Instead, we take a 1200 x 800 pixel photo and manually changing the width and height attributes of the IMG tag to a smaller size.
* Our site uses graphics for text.
* Our site has an ugly color scheme (red and green, for example).
* Our sites' symbols are not logical. Our shopping cart symbol doesn't look like a shopping cart.
* Instead of calling it a shopping cart, we call it a basket or other silly term.
* Our site uses animations gratuitously.
* Our site uses animated GIFs.
* Our site uses 3-D graphics.
* Our site uses gradient images .
* Our site uses beveled images.
* Our site uses images with shadows.
* Our site has flashing graphics that might cause seizures.
* Our site uses a background graphic that repeats itself on large-screen monitors.
* Our background graphic doesn't contrast well against the text, making it hard to read.
* Our graphics don't have ALT= attribute text filled in and doesn't use "" for graphics that are empty.
* Our site's graphics are confusing — they look like ads.
* Our site uses moving graphics — falling snowflakes, flying birds — stupidly rendered by DHTML.
* Our site doesn't use color to convey meaning — red text signifies "this is important."
* No one has spent the time figuring out if our color scheme alienates our international users.
* Our site has multiple colored areas on the page.
* Our site has blocks of ugly colors next to each other (red next to green).
* Each page on our site is one big imagemap.
* Our site has graphics that suffer from the "halo effect" — dithering that leaves an ugly halo around the image.

#### Navigation

* We created our site's navigational system to meet our needs, not our visitors' needs.
* We understand how our site's navigation works so everybody else probably understands how it works.
* We have a page or a popup explains how our navigation works.
* A site's navigation should tell you where you are, where you're going to go, and how to get back to the home page. Our site's navigation doesn't.
* Our site uses Mystery Meat Navigation.
* Our site uses Flash navigation.
* A man from Mars could not quickly understand our site's navigation.
* Although people don't want to learn a new navigational system, we've created our own anyway.
* Our site uses JavaScript for navigation and it doesn't degrade naturally for visitors who come with JavaScript turned off.
* Our site uses sideways navigation . Here's a screen capture of another site with sideways navigation.
* We use stupid terms like "stuff" for our navigational links or "Beginning" for "Home."
* Our site doesn't have shortcuts on the main page to the popular content.
* Our site's navigation is not in the top screen.
* Instead of being predictable, our site's navigation is inconsistent.
* Instead of being predictable, our site's navigational placement is inconsistent from page to page.
* Our site's content is not divided into logical categories and subcategories.
* The names of the categories and subcategories are not clear and mutually exclusive on our site.
* Our links aren't clearly labeled and don't tell you where you'll end up.
* Our links aren't clearly labeled, don't tell you where you'll end up, and say "Click Here." (May not be suitable for work NSFW.)
* Some/all of our pages require visitors to scroll horizontally.
* Our site uses Java navigation.
* Navigation graphics are not the same size and/or color.
* On our site, you may have to click four or more times to get to the information you want.
* We keep people from signing up for expensive conferences.

#### Content

* We don't know what content is popular.
* Our content is not organized to meet our visitors' needs.
* Our content is not broken down into logical categories.
* The content is not understandable by humans and is full of marketing-speak, or jargon, or unexplained acronyms.
* The content is not engaging, or relevant, or accurate, or fair and impartial.
* I don't know if our content is appropriate for our audience.
* The content really isn't appropriate for our audience.
* We don't identify non-HTML documents like PowerPoint or Excel.
* Our site's content is not written for the web, but for print media (or other media) and we just transferred it to the web.
* Our site's content is written at a higher or lower readability level than our visitors' knowledge level.
* Our pages are too long. We forget that people skim.
* Our site doesn't have Heroin Content.
* Some of our site's content — graphics and/or text — is considered offensive.
* Some of our content — graphics and/or text is considered racially or politically incorrect.
* Some of our content — graphics and/or text — is considered offensive to international audiences.
* We didn't hire editor to proofread spelling, grammar, capitalization and content.
* We haven't checked to see we've eliminated all "Lorem ipsum" text or other placeholder text and graphics from site — especially from document titles.
* Our site uses content that our visitors don't need to know.
* Our site may contain sensitive information the public and/or competitors shouldn't see.
* Our site has a mission statement or a link to a mission statement on the front page (non-profit's are exempt).
* Our site has sections that are under construction and the public can access them.
* Our site has different looks on different pages or sections.
* Our site doesn't have a privacy or legal statement page.
* Our site's content came from Microsoft Word, Powerpoint, etc., documents and then converted to HTML.
* Our site has outdated calendar information.
* Our site has too many words on a line, making it hard to read.
* Our site has too few words on a line, making it frustrating to read.

### Check List 2

#### 82 Potential Mortal Sins

* Our site isn't accessible to visitors with physical disabilities.
* Our site doesn't use white or off-white color for background.
* Our site doesn't use black text.
* Our site uses Frames.
* Our site uses the wrong doctype.
* Our site doesn't use CSS.
* Our site uses CSS, but it's inline CSS. (Note: I'm going to remove this because of some recent articles from Google about speeding up page load time).
* Our sites logo is not at top left corner.
* Our site has no search engine.
* Our search engine is not at top-right of page.
* Our site's search engine box isn't long enough for visitors to see what they're typing.
* Our site has a search engine, but not on every page.
* Underlined text is not link text.
* Our site uses invalid HTML attributes (marginwidth, etc.).
* Our site uses graphics as links.
* We haven't validated our site's HTML code.
* We haven't validated our site's CSS.
* We haven't checked to see if our site's links are broken.
* Our site uses deprecated markup (like the CENTER tag).
* Our site doesn't use a style switcher.
* We haven't checked to see how our site looks in grayscale.
* We haven't checked our site using the Firefox plugin "Web Developer".
* We haven't checked our site using the "The Web Page Analyzer".
* Our site uses the CSS hover attribute on text.
* We don't optimize our site's CSS or HTML files.
* Our site's accounting figures don't line up on the decimal point.
* Our site's logo is disproportionate for top screen.
* Corporate logo is not at top left of screen.
* Logo links to home page on sub-pages.
* There are links to the home page on every page -- but the home page.
* Putting dates on material that's not updated regularly.
* Our site's content is full of jargon.
* Our site's content is full of acronyms.
* Our site's use of abbreviations is inconsistent.
* We don't use a consistent tone throughout. We switch back and forth from colloquial to clinical etc.
* Our font sizes are fixed and can't be resized.
* We put more than one idea in a paragraph.
* Not declaring a background color.
* Our site uses unencoded ampersands.
* Our site is <span> and/or <div> happy.
* Hit counters.
* Contact information is not available.
* Our site doesn't provide multiple methods to be contacted. (Contact Us form, phone number, address, e-mail, etc.)
* We use dates and times on a site that's not about dates and times.
* Our site has pictures of boring white executives.
* Our site has pictures of boring white executives -- and if you click the picture, you get to see them up close and personal .
* Some of our pictures are not recent -- we have a 20 year-old photo of some employees.
* Our site uses JPG when it should use GIF images and vice-versa.
* We use SVG file format because we can.
* We over-optimize our images.
* Our site has graphics that look like a link, but aren't.
* We don't check to see if scanned images are scanned on dirty scanners.
* Check on an outside machine if all your sites' graphics actually load.
* There's no reason for a globe image. We get that you're global.
* Our site uses an ugly background image.
* We use Dynamic HTML navigation.
* We don't have a site map.
* We have a site map, but it's graphics based.
* Our site uses liquid design.
* Our site uses fixed-width design. (You can't win. Liquid is wrong on wide-screen monitors because you have line lengths that are hard to read — and vice versa.)
* We use table borders on content other than accounting-type data.
* OUR HEADINGS ARE CAPITALIZED.
* Our text is full of superlatives like "This product will solve every problem you have."
* Our paragraphs have too much text. (We're a web of scanners.)
* Our site bolds a lot of text.
* Our site italicizes a lot of text.
* Our site's font sizes are fixed and can't be resized.
* Our site uses three or more font faces on a page.
* We don't have a CSS file for printing pages.
* We don't use logical file names for documents or graphics.
* Our site uses graphics that don't enhance the page.
* Our graphics and text don't match up .
* Some of our documents run across multiple pages, but we don't give the option to see it all on one page.
* Back button doesn't go back because we've disabled it.
* Our site doesn't provide a feedback mechanism.
* Our forms mix up checkboxes and radio buttons.
* Our site has pull-down (drop list) menus.
* We don't use bullets to organize information.
* We don't manually spell check TITLE, ALT tags, etc. (spell checkers don't check these).
* Our content is not timely.
* Our content is not updated frequently.
* Our site doesn't separate style from content.

#### The List of the Top 30 Web Design Mistakes

|  |  |  |
| --- | --- | --- |
| **Rank** | **Votes** | **Top 30 Web Design Mistakes** |
| 1 | 630 | Our site tries to tell you how wonderful we are as a company, but not how we're going to solve your problems. |
| 2 | 617 | We've designed our site to meet our organization's needs (more sales/contributions) rather than meeting the needs of our visitors. |
| 3 | 289 | We say "Welcome to..." on our home page. |
| 4 | 275 | It takes longer than four seconds for the "Man from Mars" to understand what our site is about. |
| 5 | 244 | The "Man from Mars" cannot quickly find the focal point of the home page. |
| 6 | 222 | Our site doesn't make us look like credible professionals. |
| 7 | 177 | Our home page — or any page — takes more than four seconds to load. |
| 8 | 177 | We never conduct user testing. |
| 9 | 174 | We don't analyze our log files. |
| 10 | 171 | We don't know which design items are not necessary. |
| 11 | 162 | Quickly scanning the page doesn't tell our visitors much about its purpose. |
| 12 | 159 | We have not eliminated unnecessary design items. |
| 13 | 157 | The "Man from Mars" cannot quickly find the focal point of the current page. |
| 14 | 129 | We don't know if our site looks the same in the major browsers. |
| 15 | 128 | Our pages have too much/too little white space. |
| 16 | 124 | Our site mixes text colors on the page. |
| 17 | 123 | We don 't put design elements where our visitors expect them. |
| 18 | 121 | Logo is not on the top of every page and clicking it doesn't lead to the home page. |
| 19 | 118 | Visited links don't change color. |
| 20 | 113 | Our site doesn't make visitors feel they can trust us. |
| 21 | 113 | We don't identify PDF files with an icon. |
| 22 | 110 | Our site uses divider bars. |
| 23 | 108 | Our logo does not look like it was professionally made |
| 24 | 107 | No one has spent the time figuring out if our color scheme alienates our international users. |
| 25 | 103 | Our site breaks when visited with the Javascript turned off. |
| 26 | 101 | Our site doesn't have a privacy or legal statement page. |
| 27 | 100 | Our site mixes and matches text sizes on the page. |
| 28 | 97 | We don't know what content is popular. |
| 29 | 95 | Our site's design was "borrowed" from another site. |
| 30 | 93 | The important content does not fit in the first screen. We use justified text. |

