

The HTML Tutorial



by [little_v](#) member of [Black Sun Research Facility](#)

This tutorial will teach you how to create web pages using the Hyper Text Markup Language.

[Lesson 1: Introduction](#)

[Lesson 2: Structure and Method](#)

[Lesson 3: Basic Tags and Formatting](#)

[Lesson 4: Adding Links](#)

[Lesson 5: More on Formatting your Text](#)

[Lesson 6: Working with Fonts](#)

[Lesson 7: Creating Lists](#)

[Lesson 8: Adding Images to our Pages](#)

[Lesson 9: Creating Image Maps](#)

[Lesson 10: Getting Down and Dirty with Tables](#)

[Lesson 11: Oh my god, I've been framed!](#)

[Summary of this tutorial](#)

[Back to Blacksun's Mainpage](#)



Get up to \$80 in mail-in rebates
when you buy a \$99.99 Ericsson R289LX phone.
With select plans

[click here](#)

The FTP Tutorial / written by yours truly, R a v e N (blacksun.box.sk)

version 2.0, 27/6/99

Note: whenever you see something like this: blah(1) it means that if you don't understand the meaning of the word blah there's an explanation for it just for you, located on the newbies corner on section 1. Note 2: if you're having a hard time reading this page because you have to scroll to the right whenever a long line comes, it's probably because you're not using "word wrapping". Most UNIX text editors and advanced Windows editors (and some less advanced ones like Wordpad) do this by themselves. To do word wrapping on Microsoft Notepad, simply go to Edit and then click on "Word wrapping".

Author's notes

This file is basically intended for newbies, but gurus can benefit from it too (read everything, even the newbies corner. You might come across something you've missed when you first started studying). The next tutorials will be mostly for gurus, so bear with us. If you have any comments or questions regarding this tutorial (no flames(10) or spam, please) Email me at barakirs@netvision.net.il. Visit blacksun.box.sk for more tutorials, free hacking/programming/unix books to download and much more.

Disclaimer

We do not encourage any kinds of illegal activities. If you believe that breaking the law is a good way to impress someone, please stop reading now and grow up. There is nothing impressive or cool in being a criminal.

Contents

What Is FTP and What Is It Good For?

- * What does the acronym FTP stands for?
- * What can I do with FTPs anyway? What are they good for anyway?

-----FTP Commands-----

- * How to use FTP with raw FTP commands
- * How to use FTP with a GUI (Graphical User Interface) / text client(5)

-----FTP Hacking-----

- * Finding out information about your target and finding security holes using that info
- * Example FTP-related security holes

The Stupid Bug Corner

- * An "elite" bug

Newbies Corner

- * What is a protocol
- * What is a port

- * What is a mirror site
- * What is a path (complete path + relative path)
- * What is a client program and what is a server program
- * How to find information about remote hosts
- * What is a daemon
- * What is root
- * What is a core dump
- * What is a DoS attack
- * What is DUN
- * What is an ISP
- * What is flaming

Other Tutorials

- * FTP Hacking.
- * Overclocking.
- * Ad and Spam Blocking.
- * Sendmail.
- * Phreaking.
- * Advanced Phreaking.
- * Phreaking II.
- * IRC Warfare.
- * Windows Registry.
- * Info Gathering.
- * Proxy/Wingate/SOCKS.
- * Offline Windows Security.
- * ICQ Security.

Bibliography

What Is FTP and What Is It Good For?

The word FTP (see footnote 1 below) stands for File Transfer Protocol(1).

FTP servers will let you to both download (retrieve a file from the server) and upload (send a file to the server) files from the server with great ease (if you have permission to do so). You browse through a remote FTP site the same way you browse through your own computer's files and directories (of course, you don't have read and/or write access to every file on the system, and some files you can't even see).

FTP Commands

The following are several basic FTP commands. To communicate with FTP daemons(7), connect to port(2) 21 and then use the following commands (see footnote 2 below) to communicate with the FTP server:

cd change directory (on the server)

lcd change local directory (when sending a file, the path(4) of the specified file will be the path you specify on lcd)

dir,ls directory listing

binary change mode to binary transfer

get retrieve a file
 mget retrieve many files
 put send a file
 mput send many files
 pwd print working directory on the server

Footnotes

1. For thousands of computer-related acronyms and abbreviations head to blacksun.box.sk and download the file called across.txt from the projects page.
2. If you don't feel like typing stupid commands, there are lots of FTP clients(5) who will do all the work for you, but fortunately some will still show you all the commands they use so you'll be able to learn new commands.

You can download FTP clients for every Operating System from TUCOWS. Simply go to the nearest TUCOWS mirror site(3) or go directly to www.tucows.com.

FTP Hacking

Since there are so many FTP holes for so many FTP server programs and so many Operating Systems, I decided that the best way it simply to explain to you how to find information about security holes by yourself.

I will also introduce several interesting FTP security holes near the end of this section.

To find FTP exploits, try searching the following websites (or join the BugTraq mailing list at www.securityfocus.com):

CERT (Computer Emergency Response Team) - <http://cert.org>

X-Force Search (simplest) - http://www.iss.net/cgi-bin/xforce_index.pl

Packet Storm - packetstorm.genocide2600.com

BugTraq Archives - <http://www.securityfocus.com/level2/bottom.html?go=search>

Fyodor's Exploit World - <http://www.insecure.org/spl0its.html>

Spikeman's Denial Of Service Website (for DoS(9) attacks against FTP servers) - <http://www.genocide2600.com/~spikeman/>

RootShell - <http://www.rootshell.com>

Slashdot - <http://www.slashdot.org>

Data - <http://www.hideaway.net/data.html>

(Please report all dead links to barakirs@netvision.net.il)

Note: one might think that the above sites are considered illegal, since they feature explanations about security holes and how to exploit them.

Well, screw one. These things are called "advisories" and they allow you to find holes on your own PC and fix them. Whether you use this information to secure yourself or hack others is your own choice. It's the difference between legitimate and illegal.

After you get to one of the following search sites (I recommend the BugTraq Archives) search for the keywords you want. For example: you find out(5) that your target is using this OS with this FTP server and this Webserver program etc'. Try combining all of those pieces of information and I'm sure you'll find the holes that fit you the most. You can also try searching holes on your own computer. Speaking

about holes, we will explain about many security holes on the upcoming Sendmail tutorial (see blacksun.box.sk). Now, for several selected FTP holes.

Selected FTP Holes

The following FTP holes aren't new or extraordinary or incredibly fantastic or anything of that sort of matter. They're just good for learning. I picked some interesting FTP holes and written a small explanation about them just to get the newbies started. Note: the sites I got these from aren't "evil hacking sites". These explanations are called advisories and they are meant to be used by people who want to fix bugs on their systems. Whether you use them for that purpose or others is none of our business.

1. Some FTP daemons allows a premature PASV command, which can cause some FTP daemons to crash with a core dump(9). FTP core dumps can be used to salvage encrypted passwords, bypassing any shadow password scheme. It is not known exactly which servers are immune to this and which are not, and the only workaround right now is to get a newer FTP server program. Also see <http://www.genocide2600.com/~spikeman/bisonware3.html> for a DoS(9) attack against BisonWare FTP Server 3.5 similar to this hole.
2. FTP Bounce Attack (too long, see <http://www.netSPACE.org/cgi-bin/wa?A2=ind9507B&L=bugtraq&P=R1425> (From BugTraQ))
3. Local bug in FTP Daemon (too long, see <http://www.netSPACE.org/cgi-bin/wa?A2=ind9507B&L=bugtraq&P=R1345> (From BugTraQ))
4. (Quotes in part from BugTraQ) Impact: Anybody from outside can shutdown your pc ftp server. And if u are under win3.1 the system will crash.

Program: WinQVT/NET

Version: All versions.. 16 and 32 bits

Solution.. dont use it or upgrade

Exploit: Just Send a OOB (Out of Band) to port 21,

Exploit for dummies: Take any winnuke, start it, and when u find a "139" change it to "21" instead.

OK, I know this is stupid..... :P. But maybe somebody will need it.. who knows...

Note: A patched version of NT 4.0 isn't vulnerable to this running MS's FTP server. I haven't had a chance to test an unpatched server, but IIRC, I did check the FTP port when the OOB problem was first reported and it didn't cause a crash.

I would suspect that this could be a DOS/Win problem in general, and might not be specific to the WinQVT package.

I hope this helped you learn how to find holes. There will be much more examples in the Sendmail tutorial.

The Stupid Bug Corner

I found this on an "elite" website made by a bunch of "elite" "hackers".

They said that in order to "hack an FTP" you need to connect to it and send the following commands:

```
quote user ftp
```

```
quote cwd ~root
```

```
quote pass ftp
```

Basically, what the so-called hacker is trying to do here is to enter a username to get into the system, change the user to root(7) and then enter a password for the username.

This only works on VERY badly-configured FTP servers (the author mentioned that "this doesn't work on every FTP server". Well, I've got news for you - this doesn't work. Period. Unless you're talking about some 5 years old boy who just got a computer and clicked on some buttons and accidentally set up an FTP server).

Appendix A: the SYST command

Entering the SYST command while connected to an FTP server often reveals valuable information on a system, such as the OS, which version and information about the FTP server.

Get access to an FTP server somehow (by using a username and a password you know or by using anonymous login - login: anonymous password:your-email-address@your.isp. You could also enter someone else's Email address, the server doesn't actually verifies the address you send or anything) and then type the SYST command.

Newbies Corner

1. Protocol - a set of rules and regulations, similar to a language. When two computers know the same protocol, they can use it to communicate with each other.
2. Port - (for the more technical explanation of what ports are, see the end of this explanation) ports are like holes that enable things (data, in this case) to come in or out of them. There are physical ports and software ports on your computer. Physical ports are those slots on the back of your computer, your monitor etc'. Now, software ports are used when connecting to other computers. For example: I just bought a new computer and I want to turn it into a webserver (I want to enable people to access selected web pages, pictures, cgi and java scripts or applets, programs etc' that are located on my computer). In order for that to happen, I need to install a webserver software. The webserver software opens a port on my computer and names it port 80. Then it listens to incoming connections on that port. When someone starts his Internet browser (Netscape, Lynx, Microsoft Explorer etc') and surfs to my website, his browser connects to my computer on port 80 and then sends HTTP commands that my webserver program can understand into it. My webserver program quickly picks up the incoming data and then sends it back into a port that the surfer's browser opened on the surfer's computer. The browser will listen on that port and wait for the data (the HTML page, the picture, the program etc') to come in through it. There are different ports for different services (we'll get to that) so data won't mix up. Imagine your browser getting data your FTP client was supposed to get. I hope you got the main idea of what a port is. Now, there are three kinds of ports: well-known ports, registered ports and dynamic/private ports. The well known ports are those from 0 through 1023. These are default ports for several services (a webserver is a service because it listens for connections from remote computers and then sends something back). For example: the default port for web servers is 80. Else, how would your browser know which port he has to access? Now, the registered ports are those from 1024 through 49151. These ports are reserved for several programs. For example: ICQ (www.icq.com) reserves a port and listens to incoming messages on it. The dynamic and/or private ports are those from 49152 through 65535, and can be used by anyone for any given purpose.

"Techy Explanation" - To grant simultaneous access to the TCP module, TCP provides a user interface called a port. Ports are used by the kernel to identify network processes. These are strictly transport layer entities (that is to say that IP could care less about them). Together with an IP address, a TCP port provides an endpoint for network communications. In fact, at any

given moment *all* Internet connections can be described by 4 numbers: the source IP address and source port and the destination IP address and destination port. Servers are bound to 'well-known' ports so that they may be located on a standard port on different systems. For example, the telnet daemon sits on TCP port 23, the FTP daemon sits on TCP port 21, the rlogin daemon sits on TCP port 513 etc'.

Important note about well-known ports: services (daemons waiting for incoming connections that serve people in some way) on these ports can be only ran by root, so inferior users won't start messing up with important ports.

3. Mirror site - a website which is an exact copy of the original website which is hosted by a different server. Mirror sites can be used to speed up downloads/uploads. For example: instead of downloading/uploading from/to the main tucows webserver, located somewhere distantly from my home, I can simply do it from one of their Israeli mirrors (mirror site located in Israel, my country) and that way the downloads/uploads would go faster.
4. Path - UNIX example: if a file is located at /etc/passwd, the file's path would be /etc.
DOS/Windows example: if a file is located at c:\windows\win.exe, the file's path would be c:\windows. There are two kinds of paths: a complete path and a relative path. Complete path on DOS/Windows: if the file is located on c:\program files\quickview plus\ then this is the file's complete path. Complete path on UNIX: if the file is located at /usr/local/sbin then this is the file's complete path. Relative path on DOS/Windows: if the current directory (the directory you are on at the moment) is c:\windows and the target file is located at c:\windows\temp then the relative path to this file is temp. Relative path on UNIX: if the current directory is /usr/nobody and the file is located at /usr/nobody/public_html/cgi-bin then the file's relative path is public_html/cgi-bin.
5. Client / Server programs - A client program is a program that uses a resource offered by another program/computer. A server program is a program that supplies resources to client programs. Example: Client=Netscape Navigator. Server=Apache version 1.6.6 (a webserver, meaning a program that lets people who use Internet browsers to download specific web pages, pictures, files etc' from the computer it is installed on).
6. How to find out information about remote hosts - the best way to find out information is too look at daemon(6) banners. Daemon banners are small pieces of information some daemons return when connected to in order for the remote machine (the one connecting to the daemon) to know how to interact with them better. Try connecting to port 80 (webserver) and sending some commands like get and then looking at the banner. You may also try Sendmail (see next tutorial) on port 25, Telnet on port 23, FTP on port 21 or whatever you can come up with.
7. Daemon - a program that listens for incoming connections from remote machines on a specified port(2) and interacts with them.
8. Root - also referred as superuser, because his permissions are endless. His UID (User ID number, an identification number and user on a UNIX system has) and GID (Group ID. You can create groups and give them several permissions. For example: everyone from the accounting department can read and execute all the files on this directory, etc') are always 0 (except on very altered boxes). Once you are root, you can do practically anything on a system. Core Dump - when a program crashes it dumps all the core (all the info it handles that isn't saved on disk, meaning all of the program's stuff that are on the RAM chip) into a temporary file.
9. DoS - Denial of Service. A nuke in dummies language. Some kind of an attack that causes the

target computer to deny some/all kinds of services to the users of that computer (including remote users). For example: Winnuke (also known as OOB), the simplest DoS in the world. (Taken from Spikeman's DoS site) This denial of service program affects Windows clients by sending an "Out of Band" exception message to port 139, which does not know how to handle it. This is a standard listening port on Windows operating systems. Users of Win 3.11, Win95, and Win NT are vulnerable to this attack. This program is basically a nuisance program, but it is being widely circulated over the internet now. It has become a bother in chatrooms and on IRC. By using your IP# and sending OOB data to port 139, malicious users can disconnect you from the net, often leaving you with low resources and the blue tinted screen. Some of you may have been victims already. If this happens to you on Win 95, you will see a Windows fatal error message similar to the following: Fatal exception 0E at 0028: in VxD MSTCP(01) + 000041AE. This was called from 0028: in VxD NDIS(01) + 00000D7C. Rebooting the comp should return it to normal state.

Patches ("fixes") For WinNuke (OOB)

Additional Information on WinNuke

<http://support.microsoft.com/support/kb/articles/Q168/7/47.asp>

Windows 95 Patches

<http://support.microsoft.com/download/support/mslfiles/Vipup11.exe>

<http://support.microsoft.com/download/support/mslfiles/Vipup20.exe> (for Winsock 2.0*)

<http://www.theargon.com/defense/nuke/index.html>

Please read notes referring to 95 patches before installing.

Which version of Winsock do you have on your Windows 95 PC?

<http://premium.microsoft.com/support/kb/articles/Q177/7/19.asp>

<http://www.theargon.com/defense/nuke/index.html>

Windows NT 4.0 Patch

<http://support.microsoft.com/support/kb/articles/Q143/4/78.asp>

<http://www.theargon.com/defense/nuke/index.html>

Please read notes referring to Windows NT patches before installing.

More info on DoS attacks can be found at Spikeman's DoS site:

<http://www.genocide2600.com/~spikeman/main.html>

* I do not know if it will work on newer versions of Winsock, so you'd better downgrade to Winsock 1.1 (the version that comes with Windows 95) by going to Control Panel, Network and removing TCP/IP and Dial Up Adapter(11) and then readding them (click add, choose protocol and in the company frame choose Microsoft and you'll find TCP/IP. For DUN do the same but choose adapter instead of protocol).

After you finish downgrading reupgrade to Winsock 2.0, apply the patch (Vipup20.exe) and then upgrade to newer versions of Winsock.

10. Flames - the action of flaming someone (send him angry mail about things he has done, opinions he has etc' which you do not agree with).
11. DUN - Dial Up Adapter. Basically it's the Windows program that dials to your ISP(12).
12. ISP - Internet Service Provider. A company that provides Internet services, such as Internet

connectivity, web hosting, Email services etc'.

13. Distro - Distribution. Since UNIX is not a registered patent, trademark, copyrighted or whatever there are many distributions (software packages) of it. Every distro has it's own advantages and disadvantages (example: Redhat is the best for beginners).

Next Tutorials

The next tutorial will be about Sendmail, the buggiest daemon on earth - what is Sendmail, Sendmail commands, how to hack through Sendmail, how to send completely untracable mail, a newbies corner (what is a daemon, how to trace mail etc') and much much more. If this tutorial scores 7 points out of 10, then the Sendmail tutorial with score 12. First of all, it's gonna be veery looong and it'll have lots of side tips and thorough explanations about security holes and tips and tricks and tons of cool stuff I havn't thought of yet. Besides, I did this tutorial in a rush 'cause I didn't have much time to work on it*, but summer vacation is coming up so I'll have plenty of time to work on the Sendmail tutorial. The 3rd tutorial will be probably about UNIX Shell Programming. I don't wanna give away any details right now, and besides - I'm not so sure about this title. Maybe I'll change it to an "All you wanted to know about IRC wars and never had the guts to ask" tutorial. Who knows. I'll set up a electronic poll soon so you'll be able to vote on that subject or suggest other titles (subscribe to the mailing list and you'll be notified when it's ready. To subscribe, go to blacksun.box.sk and go to the Mailing List page). For more information, head down to blacksun.box.sk. Don't forget to drop us a line!

* Just installed Redhat 6.0. Yeah, yeah, I know, it's not exactly the best Linux distro(10) out there (I'm trying not to offend all of you Redhat users out there), but I wanted to see how it looks and everything. I gotta tell you, the installation is EEE-ZZZ comparing to other distros, and it's great for beginners.

Note: before I'll release the Sendmail tutorial I will send out some mini-tutorials, such as "Buffer Overflows", "Overclocking", "RM Networks" etc'.

Other Tutorials

Overclocking.

RM Networks Hacking.

Ad and Spam Blocking.

Sendmail (creating fake mails and hacking servers that run Sendmail).

Get them all at blacksun.box.sk, or join the mailing list at blacksunresearch.listbot.com.

Bibliography

BugTraq Archives - <http://www.securityfocus.com/level2/bottom.html?go=search>

RootShell - <http://www.rootshell.com>

Fyodor's Exploit World - <http://www.insecure.org/sploits.html>

Packet Storm - <http://packetstorm.harvard.edu>

X-Force Search (simplest) - http://www.iss.net/cgi-bin/xforce/xforce_index.pl

Slashdot - <http://www.slashdot.org>

Spikeman's Denial Of Service Website - <http://www.genocide2600.com/~spikeman/>

PC Magazine - <http://www.pcmagazine.com>

Other Tutorials

- * FTP Hacking.
- * Overclocking.
- * Ad and Spam Blocking.
- * Sendmail.
- * Phreaking.
- * Advanced Phreaking.
- * Phreaking II.
- * IRC Warfare.
- * Windows Registry.
- * Info Gathering.
- * Proxy/Wingate/SOCKS.
- * Offline Windows Security.
- * ICQ Security.



Ad and Spam Blocking for Neophytes / written by yours truly, R a v e N (blacksun.box.sk)

version 1.8, 9/9/1999

Converted to HTML by [Penguin](#)

Note: whenever you see something like this: blah(1) it means that if you don't understand the meaning of the word blah there's an explanation about it just for you, located on the newbies corner on section 1.

Author's Notes

If you have any comments or questions regarding this file (no flames(9) or spam, please) Email me at barakirs@netvision.net.il.

Visit blacksun.box.sk for more tutorials, free hacking/programming/unix books to download and much more.

Disclaimer

We do not encourage any kinds of illegal activities. If you believe that breaking the law is a good way to impress someone, please stop reading now and grow up. There is nothing impressive or cool in being a criminal.

Content

- [What is Ad Killing?](#)
Why would I want to kill commercial ads on the Internet?
What do I have to lose?
- [Get To Work](#)
What do I need in order to do some ad killing?
Killing Banner Ads
Killing Banner Ads from Free ISPs
Killing Popups
Killing Spam
- [Newbies Corner](#)
What is the /etc/hosts or the c:\windows\hosts File?

What is DNS Lookup / Reverse DNS Lookup?

What are Popups?

What is a Bandwidth?

- [Appendix A: junkbusters.com](#)
- [Appendix B: more ad blocking](#)
- [Bibliography](#)

What is Ad Killing?

If you've been using the Internet for some time, you should be aware of those annoying commercials Popups(3), commercial banner ads, unsolicited commercial mail (spam) etc'. There are simple and difficult ways to kill those, according to the sophistication level of the advertiser.

Okay, so popups are annoying, but why would I want to kill regular banner ads which just appear within a page? Well, you have to download those things, right? Some ads could be 3Ks big, some could be 20Ks big. The point is - they chew up bandwidth.

Okay, so a banner ad could be 7Ks big... so what? Well, suppose your computer is a part of a Local Area Network (LAN, a bunch of computers who are located very near to each other (same room, same building etc') and are connected to each other so they can exchange files through the fast network cable, share resources etc'). The LAN has one connection to the Internet which is enough for 10 people (say, 100Ks per second, so each user gets about 10Ks per second). Now, imagine that 5 of the 10 people are browsing the web, and each one is downloading a 7Ks big banner ad. That means you lose 35Ks per second. Now what if those people won't have to download those ads? And what if the problem would be on a bit larger scale... like a 10Ks banner, or a bigger network, or more users downloading ads etc'. See my point?

Now, I myself do not recommend killing banner ads, because some might turn out to be useful (for example: an ad about a store that sells a new A-class state-of-the-art computer for a very cheap price with no catches). Popups, on the other hand, are annoying and in my experience they never yield any useful pieces of information, so I recommend killing those. But it's still worth knowing (if you're a sysadmin and you don't want any stupid ads to chew up your bandwidth(4))

Get To Work

Killing Banner Ad's

First make a list of computers that host banner ads programs. Suppose you decide that [www.foobar.com](#) is an ad haven. Next thing you add this line to the hosts(1) file:

```
127.0.0.1 www.foobar.com
```

Now, whenever any Internet application will try to access something from [www.ads-r-us.com](#) it will try the equivalent on 127.0.0.1. For example: [http://www.ads-r-us.com/stupid-banner-ad.gif](#) = [http://127.0.0.1/stupid-banner-ad.gif](#).

Whether you have a picture called stupid-banner-ad.gif on your computer, it will not chew up any bandwidth because 127.0.0.1 means self (as in me, as in my own computer. For example: try to hack 127.0.0.1 and you'll realize that this host is suspiciously similar to your own computer... hmm... maybe they hacked your computer and downloaded everything... lol).

There is a way to work around this, though. If you put the advertiser's IP address instead of his hostname in the part of the html code on your website that tells the browser to download the ad, it will go directly

to that IP (for example: if www.ads-r-us.com's IP is 123.7.14.139 then putting 123.7.14.139/stupid-banner-ad.gif instead of www.ads-r-us.com/stupid-banner-ad.gif will work around our trick). If you know of any better tricks please let me know at barakirs@netvision.net.il.

Killing Banner Ads from Free ISPs

There are some Internet Service Providers that give you free surfing (phone bills not included in most cases) in exchange for you using a stupid program that displays banner ads on the corner of your screen while you surf. These ads are not only annoying, but they also chew up your bandwidth(4). If you want to completely remove those things you need to find some sort of a crack for it. I'm not going to tell you how to crack every free ISP in the world, but I am going to tell you how to block those ads. First, go find a good firewall (try <http://www.theargon.com>, they have some). Then, run it and wait for a new banner ad to come from your ISP. Then the firewall will warn you about the incoming connection. You can either tell your computer to ignore these things manually, or configure a rules file for your firewall that will do so (consult your firewall's help files).

Killing Popups

There are programs that do this for you. They look for some special text in the title bar of the popup (for example: killing all popups with the text "Welcome to a Geocities Member Page" in their title will kill those annoying Geocities popups) or kill the popup by the size of the window (or both). Although I've been searching for a good one for a long time now, I still haven't found one that is good enough. Surf In Peace is pretty good, though. Go to www.download.com, www.cnet.com or www.zdnet.com for more information and programs.

Killing Spam

Yes, spam. Unsolicited commercial mail. Again, to kill those, you need programs called Spam Filters. Go to www.tucows.com and find the Anti-Spam category. There you will find lots of Spam Killing programs.

Basically, what spam filters do is to decide if an incoming message is spam or not by it's subect line and the body of the message. If they do find spam, they will delete it immedietly. If those won't work, call your ISP and tell them to block all incoming mail from the spammers' Email addresses.

Note: some ISPs have began implementing gigantic and ultra advanced spam filters on their mail servers. These are supposed to stop 90%-95% of incoming spam. These ISPs include Earthlink, Netscape and Usa.net.

Oh, btw, here's an interesting piece of information. A research called "Who Spams You" has been conducted lately. Here are the results:

First place: "get rich quick" scams.

Second place: adult websites.

Third place: website promos.

Fourth place: other.

Fifth place: software promos.

Newbies Corner

1. What is the /etc/hosts or the c:\windows\hosts File? - most OSs (OS=Operating System) I know have a

hosts file. UNIX usually stores it at /etc/hosts, Windows stores it at c:\windows\hosts and Windows NT stores it at c:\WinNT\system32\drivers\etc\hosts (thanks to Teolicy for the WinNT tip). The hosts file is used instead of wasting time to do a DNS Lookup(2). I mean, why waste time on DNS Lookup when you already know the IP but you just don't feel like typing it down and you'd rather remember the hostname. The hosts file should look like this:

```
# This is a comment line.
# Anything beginning with # will be disregarded by your computer.
# You don't have to put comment lines if you don't want to, but they make reading easier.
IP-address hostname
```

For example: on most hosts files you will see this line:

```
127.0.0.1 localhost
```

Anything directed to 127.0.0.1 is directed to self, meaning to your own computer.

This hosts file entry allows you to type localhost instead of 127.0.0.1 in browser windows or any other network application.

Note: some OSs do DNS Lookup first, and then, if DNS Lookup fails they go for the hosts file, but you can always reconfigure that somewhere (take MacOS for example: all you have to do is to put the line set use_hosts_first=1 somewhere in the config.sys file).

2. What is DNS Lookup / Reverse DNS Lookup? - the command nslookup hostname (Unix only. For a Windows version see <http://www.samspace.org>) gives you the IP address of that hostname.

How does it work?

Every computer which is connected to the Internet is assigned with an IP address, whether it accepts connections or not. If you want to connect to that certain computer, you have to know it's IP address, whether you like it or not. But what if you're senile and you don't feel like remembering IP addresses? This is what hostnames are for. Hostnames are simple names for IP addresses in the human language. For example: if you wanna surf over to Yahoo and you don't remember their IP address or you don't feel like finding it out, you can go to www.yahoo.com instead. www.yahoo.com is an alias to Yahoo's IP. Now, this is what DNS is for. DNS stands for a Domain Name Server. These servers store those aliases and their IPs.

A DNS Lookup means to find the IP of a given hostname. A reverse DNS Lookup is to do the exact opposite (IP==>hostname).

For more info, type man nslookup on Unix for nslookup's manual page or go to Sam Spade's library (see <http://www.samspace.org>).

Nslookup is a wonderful probing tool, and in fact it's one of the best ways to find out information about a certain host, so go and learn it.

3. Popup - another browser window that "pops up" by itself. Try going to any Geocities homepage and you'll see a good example of a popup window, because Geocities insert these things in every homepage hosted by them (except GeoPlus members, but they pay for that... :().

4. Bandwidth - the total speed a certain connection can achieve. Chewing up bandwidth means wasting some of the bandwidth, hence making surfing slower.

Appendix A: junkbusters.com

Here's an Email I got from someone called James Slater:

Hi there, I just read your tutorial, and thought you might like to make some mention of 'The Internet Junkbuster' (<http://www.junkbusters.com>). It's a proxy server that you can install on your Win9x/Un*x system that can be configured to block unwanted cookies, adverts etc. depending on a set of rules. You might think it's a bit out of the scope of the tutorial, but I thought I'd suggest it.

Well, there you have it. Junkbusters.com. C'mon, go give it a whirl!

Appendix B: blocking ads

I was told by a friend who wishes to stay anonymous that you can use the following command:

```
route add -host [spam server] reject
```

(replace 'spam server' with the IP or hostname of the server that has these banners on it. For example: if the banners come from banners-r-us.com, replace 'spam server' with banners-r-us.com)

This command orders the Linux kernel (no, it won't work on Windows. That's one of the things I hate about Windows - it's not sophisticated enough for me... lol) reject anything coming from this IP.

This does the same as that thing I did with /etc/hosts, but... I dunno, it's a little nicer... ;-)

Bibliography

Sam Spade's library (<http://www.samspade.org>)

Welcome to the HTML tutorial!

Soon you shall be on your way to building great Webpages and vast Websites, but first lets go over a thing or two about what a "Web Page" is, how they work, and what we can do with them.

For this tutorial you will need:

- An Internet ready Computer
- A Web Browser

Because you are reading this on a Web Page, I will assume that you have both of these things.

Let's jump right in to the Introduction!

In the beginnings of the Internet, it was very hard to exchange data. So with great vision, Tim Berners-Lee created a way to connect text on the Internet through Hypertext Links (References to other text on the Internet). This was'nt a new idea, but his Hypertext Markup Language (HTML) was very popular, and caught on better than other developer's projects.

HTML was not a "Programming Language" per se, but rather a *Scripting Language* that *marks up* the page with formatting commands. Your Web Browser then reads these commands and shows the accessed page on your screen.

Due to the popularity of the Web, some programmers wrote Web Browsers that could view graphics, and a wide range of content. Thousands of people started to create web pages, which ranged from personal "homepages" to business information pages.

Today, millions of people access the web. There is now a diverse medium of content on the web. Before going on to the next lesson, I suggest that you go out and view many pages that are out there on the Web. As you are viewing them, to view the HTML that they are made of click View|Source, if you're using Microsoft Internet Explorer or View|Document Source with Netscape Navigator.

[Click here to go to lesson 2](#)

[Back to Blacksun's Mainpage](#)

Lesson 2: Structure and Method

I'll bet you're thinking "Structure and Method? What is this... some kinda textbook???" Well, no, in this lesson you'll be learning about the **Structure** of HTML and the **Method** that is used to make them.

HTML is not coded with some special "HTML tool", and you do'nt even need some special program to make HTML pages, like Microsoft FrontPage (In fact, I discourage their use until you know the ins and outs of HTML code). All that you DO need is a simple text editing program like Notepad.

You're probably thinking "Wait just one second, you're telling me I can code up another [Yahoo!](#) with my puny little Notepad? Yes! That's part of the beauty of HTML! How do we do this? Keep reading!

When you make a Web page the first thing you need to do is gather your content. For our first page ever, we'll be making a informative page about ourselves. For example, here is mine:

Welcome to Justin's Web Page!

Hi, My name is Justin. I built this web page because I love coding in HTML! I could do it all day long!

I am a lover of programming languages, and love to design and produce web content.

Thanks for visiting my page!

Yours Truly,
-Justin

Go ahead and think up a few paragraph's like this, and meet me at the [next lesson](#).

[Back to Blacksun's Mainpage](#)

Lesson 3: Basic Tags and Formatting

So you're probably thinking "Okay, I knew that stuff. Teach me something I don't know." Okay, be patient. I think this lesson will give you your fill.

HTML is a language that is coded with *tags*. They are called tags because they tag parts of a webpage for formatting in a browser. HTML tags are very easy to spot in web page source. They are the things shown that start with a < and end with a >. Most HTML tags have an opening tag (<tag>) to start formatting text and a closing tag (</tag>) to end the formatting.

There are some HTML tags that are absolutely necessary in an HTML page. Those tags are as follows:

<HTML>' and </HTML>': Usually put at the top (<HTML>) and bottom (</HTML>) of an HTML page. This tag tells the browser that this page is an HTML page.

<HEAD> and </HEAD>: This is where information about the entire page is placed.

<TITLE> and </TITLE>: This tag needs to be put between the <HEAD> and </HEAD>. This tag gives a name for your page. The name won't be shown in the the text of the page, but rather in the top of the browser window.

<BODY> and </BODY>: This tag defines the body portion of the page. Later we will learn how to use the <BODY> tag to add background colors, text colors, and margins.

Now that we know the required tags, I'm going to put my text that I gathered in the [last lesson](#) in the proper web page format.

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Justin's Homepage</TITLE>
```

```
</HEAD>
```

```
<BODY> Welcome to Justin's Web Page!
```

Hi, My name is Justin. I built this web page because I love coding in HTML! I could do it all day long!

I am a lover of programming languages, and love to design and produce web content.

Thanks for visiting my page!

Yours Truly,

-Justin

```
</BODY> </HTML>
```

But wait 1 second! I typed that, and opened it with my browser, and it doesn't come out like that! What? Does it come out like this:

Welcome to Justin's Web Page! Hi, My name is Justin. I built this web page because I love coding in HTML! I could do it all day long! I am a lover of programming languages, and love to design and produce web content. Thanks for visiting my page! Yours Truly, -Justin

Yes? Ohmygod! We forgot the formatting tags!

The <p> tag can add paragraph spacing to your page. The
 tag adds a single line break to your page.

These tags do not need a <p> and </p>, or
 and </BR>, unlike the other tags.

So now our page is coded like this:

```
<HTML>
<HEAD>
<TITLE>Justin's Homepage</TITLE>
</HEAD>

<BODY>
```

Welcome to Justin's Web Page!<p>

Hi, My name is Justin. I built this web page because I love coding in HTML! I could do it all day long!<p>

I am a lover of programming languages, and love to design and produce web content.<p>

Thanks for visiting my page!<p>

Yours Truly,

-Justin

```
</BODY>
</HTML>
```

Great! We have format! Now i'll bet you want that first line to be large, right? This can be accomplished by what's called a "header". The header is made by putting in <Hx> "header text goes here" </Hx>, where the "x" is a number from 1-6, the bigger the number the bigger the header. So in our page it looks like <H2>Welcome to Justin's Web Page!</H2>. Note that i removed the <p> tag, since header are already paragraph spaced!

Go ahead and make your page, save it with a ".html" extension at the end (HTML pages all need either .htm or .html extensions) open it with your browser and meet me at the [next lesson!](#)

[Back to Blacksun's Mainpage](#)

Lesson 4: Adding Links

Great! We have a functional page of text! Are you satisfied? No? Well, me either. Ok, let's expand our page by adding some links to other web pages on it!

The ability to link to other web pages is exactly what makes HTML *hypertext*. Hyper means outside of, and when you link to another web page, you link outside of your own page.

Linking to another page is easy. In fact, you only need to use one tag! This tag is the <A>, or *anchor* tag. The <A> tag uses the HREF argument to specify the site to link to. So it looks like this This is my link!. Note the *This is my link!* part in between the <A> and . This is the part where you enter the text that'll show up in the browser. This text is underlined in some browsers. When Joe Surfer click on this text it'll take you to the link in <A HREF>, which is http://www.mylink.com in this case.

But wait, sometimes it's easier still! Sometimes you might want to do *relative linking*. This is when one page on your site links to another page on the same site. So if you're in http://mysite.com/index.html and want to link to http://mysite.com/page2.html you'd simply add a relative link. The format in this case would be Go to page2. The browser will then look in the current directory for that page.

If you want to get a page one directory up from the current one, you simply insert a ../ before the filename of the page. For example, if you're in http://mysite.com/newstuff/main.html and need to link to http://mysite.com/index.html, this is what you'd do:

Go to the index. You can add as many ../'s as you want to get to progressively higher directories.

If you want a link that people can click on to send email to, you just add mailto: before your email address. So it would look like this:

Mail me! Kapish? Ok great! Let's move on to the [next lesson!](#)

[Back to Blacksun's Mainpage](#)

Lesson 5: More on Formatting your Text Hey! We're really getting somewhere! By now i'll bet that you've got a very informative page with a beautiful link to someplace (this site? please?! awww....) on it. That's great, but i'll bet you're hungry to make it look even more beautiful than it already is (or maybe your boss demands it!). Well, read on!

Do you like your name to stand out on your pages? I do! On every page i make sure that *little v* is in boldface. How did I do this? HTML has some special tags that allow us to change the way our text is shown. The one i just used is the or boldface tag. You can also make text in italics with the <I> or *italics* tag. Similarly, you can underline text with the <U> or underline tag. These tags need both an opening tag and a closing tag, so the format is <I, B, or U>text to be formatted</I, B, or U>.

In addition to these regular formatting tags, theres lots of special formatting tags. Heres a list of them

Special formatting tags

- 1. <SMALL> - Small text**
- 2. <BIG> - Big text**

3. **<SUPER> - Superscript**

4. **<SUB> - Subscript**

5. **<STRIKE> - Strikethrough text (text with a line through it)**

6. **<TT> - Monospaced (typewriter style) text**

7. **<PRE> - Preserves all format and line breaks in source HTML**

Yep, *text formatting is great!* Now here's how to align your text.

When you align your text, it lines up with that portion of the window. So left aligned text (the default) is lined up with the left side of the window just like using a word processor and clicking the left align button.

However, there is no align tag. What HTML has is what's called an attribute. An attribute is an argument that is put into a tag to change the way that tag works. The align attribute can be put into many different tags to format paragraphs, or blocks of text (pictures too-more on this later). So if we wanted to make a paragraph aligned to the center of our window, we'd just add the align attribute to the <P> tag at the beginning of that paragraph. So it

would look like this:

<P ALIGN="center">

Hi, this paragraph is aligned to the center. This was accomplished by using the align attribute.

And here's how it would show up in the browser window:

Hi, this paragraph is aligned to the center. This was accomplished by using the align attribute.

There is also a way to format whole blocks of text. The way we do this is by using the **<DIV>**, or division tag. The division tag really doesn't do anything without the align attribute. In fact, it's useless with no attributes!

The **<DIV>** tag is used like this:

<DIV ALIGN="left">

This is the text to be left aligned. **<P>** I can align lots of text with the division tag!

</DIV>

If we put this into our web page, it shows up in a browser like this:

This is the text to be left aligned.

I can align lots of text with the division tag!

The align tag can also be used in headers. If you want your header to be eye catching, align="center" it!

All this stuff is great, but i bet you're worrying about the small margin space in your pages. Well, that can be fixed with one easy tag: the <BLOCKQUOTE> tag! Simply put a <BLOCKQUOTE> right under that <BODY> tag, and a </BLOCKQUOTE> right above the </BODY>, and you'll have beautiful margins in your page with minimal work!

Now you should be an expert in text format and alignment. Take a few minute to absorb what you've just learned, and surf to the [next lesson](#).

[Back to Blacksun's Mainpage](#)

Lesson 6: Working with Fonts

Yeah, HTML is good. We've gotten pretty deep into text control, but there's still more ahead so let's trudge on.

We can control the font in HTML using, what else, the tag! We can use the tag to control size using the SIZE attribute. The SIZE attribute is used like this:

```
<FONT SIZE="x">
```

This text font size x.

```
</FONT>
```

Where x is a number, from 1 to 7. The size that the formatted text is depends on the viewers preference settings and screen resolution. Generally though, 1 is really small and 7 is really big. Just in case you're curious, the default font size is 3. There is also a tag called the <BASEFONT> tag, which only can take the SIZE attribute, but is made to change the size of the text on the entire page.

We can also add color to our text using the tag. Color is added to text using, duh, the COLOR attribute to the tag. We use the COLOR attribute the same way as the SIZE attribute:

```
<FONT SIZE="4" COLOR="blue">
```

This is colorful text in font size 4!

```
</FONT>
```

To our page's viewers, it would look like this:

This is colorful text in font size 4!

The COLOR attribute can use these standard colors: black, white, green, red, yellow, blue, aqua, fuchsia, gray, lime, maroon, purple, navy, olive, silver or teal. What? You need more control? Then I suggest you learn the hexadecimal color codes. Hex color codes are used like this:

```
<FONT COLOR="#0033FF">
```

This text is purple.

```
</FONT>
```

For more information on Hex color codes, I suggest you try [this site](#).

The last attribute to we'll learn in this lesson is FACE. By defining a font's face, you can control the appearance of that font. The FACE attribute is used like this:

```
<FONT FACE="arial" size="5" color="blue">
```

This text is a stunning arial size 5 in blue!

```
</FONT>
```

To the viewer, it would look like this:

This text is a stunning arial size 5 in blue!

Yes, this is great, but there's a catch: for the viewer to see this font change he needs to have the font *on his computer*. How does it get there? Different computer's come with different font's installed. For example, the Arial font is used on PCs, but Macs use a similar font called Helvetica. We can get around this by asking for backup choices in our FACE attribute. This is done like this:

```
<FONT FACE="arial,helvetica">
```

This text is either Arial or Helvetica.

```
</FONT>
```

This will show up as arial on computers that have arial installed, or helvetica if they do'nt have arial but do have helvetica. If they do'nt have either, the font does'nt change. However, you can specify as many back up choices as you want.

Yeah, simple text is great, but what if we need to put some special characters into our page? Well, once again HTML to the rescue! The format for this is & followed by the Numeric Code of the special character, or the *mnemonic entity* of that character, followed by a ;. Here's a list of the important special characters:

Character	Numeric Code	Mnemonic Entity	Character Name
"	#34	quot	Quotation mark
&	#38	amp	Ampersand
<	#60	lt	Less Than sign
>	#62	gt	Greater Than sign
¢	#162	cent	Cent sign
£	#163	pound	Pound sterling
	#166	brkbar	Broken Vertical bar
§	#167	sect	Section sign
©	#169	copy	Copyright
®	#174	reg	Registered trademark
°	#176	deg	Degree sign
±	#177	plusmn	Positive or Negative
²	#178	sup2	Superscript two
³	#179	sup3	Superscript three
·	#183	middot	Middle dot
¹	#185	sup1	Superscript one
¹ / ₄	#188	frac14	Fraction one-fourth
¹ / ₂	#189	frac12	Fraction one-half
³ / ₄	#190	frac34	Fraction three-fourths
Æ	#198	AElig	Capital AE ligature
æ	#230	aelig	Small ae ligature
É	#201	Eacute	Accented capital E
é	#233	eacute	Accented small e

×	#215		Multiply sign
÷	#247		Division sign

I'll be putting up a complete list of these signs on the [main site](#), so come back and check for it later.

Hey! Here's a little secret: to change the Background color of our web page, simply insert the BGCOLOR attribute into your <BODY> tag. It's used like this:

<BODY BGCOLOR="color or hex number code">

A little review: How would we put the copyright sign (©) on your website? If you answered & followed immediately by #169;, you're correct! You deserve to go on to the [next lesson](#)!

[Back to Blacksun's Mainpage](#)

Lesson 7: Creating Lists

We've just about learned everything there is to know about text formatting, but there's still one very important thing that we have to learn: Lists.

Lists are everywhere. We post them on our refrigerators and take them to the grocery store. Lists are a very efficient way to organize information. Naturally, HTML has a few tags to help you make lists. HTML has not 1, not 2, but 3 different types of lists that you can add to your pages! They are the **ordered**, **unordered**, and **definition** lists.

Ordered lists are exactly what the name implies: lists that follow a numerical order. Ordered lists begin with the `` tag and end with a `` tag. When we want to put an item into this list, we need to put a ``, or *list item* tag before that item.

Here's an example of the ordered list syntax:

What do I need from the store today?`<p>`

```
<OL>
<LI>Bread
<LI>Cheese
<LI>Milk
<LI>Butter
</OL>
```

And heres how it looks to our viewers:

What do I need from the store today?

1. Bread
2. Cheese
3. Milk
4. Butter

Sure thing, but what if we do'nt want our list items to be numbered? That's when we use *Unordered Lists*. These are also called *Bulleted Lists*. Bulleted lists begin with the `` tag, and end with the `` tag. They look exactly like ordered lists, except the item numbers are replaced with special characters called *bullets*. Here's an example of how bulleted lists are used:

The Tutorial Underground is:`<p>`

```
<UL>
<LI>Cool!
<LI>Free!
<LI>Informative!
</UL>
```

And here's how it would look to Joe Browser (our viewer!):

The Tutorial Underground is:

- Cool!
- Free!
- Informative!

Think that's cool? Try adding the *TYPE* attribute to the *UL* tag! With the *TYPE* attribute, we can change the type of bullet that we want to use! The *TYPE* attribute takes three arguments: "disc", "square" or "circle". So our new ** tag with a circle bullet would look like this: *<UL TYPE="circle">*. The *TYPE* attribute can also be used in the ** tag to change from numbers to letters (capital[*TYPE*="A"] or small[*TYPE*="a"]), or roman numerals (uppercase[*TYPE*="I"] or lowercase[*TYPE*="i"]). If we want to make an ordered list with uppercase roman numerals, it looks like this:

```
<OL TYPE="I">
```

The ** tag also has an attribute: *VALUE*. With the *VALUE* attribute, we can change the value of a list item! Take a guess, what result would the following code result in:

```
<OL TYPE="A">
<LI VALUE="2">Think hard now!
</OL>
```

If you guessed "B. Think hard now!" you're right? Why? Check out the combination of the *TYPE* and *VALUE* attributes!

Heres a little trick: we can start an ordered list with any number (or letter, if we use *TYPE*) with the *START* attribute. It looks like this:

```
<OL START="3">
<LI>This is item number 3! </OL>
```

And to our viewers at home, it looks like this:

3. This is item number 3!

The last list that we can use is the *Definition List*. Definition lists are normally used when we need to define terms. The definition list starts with the *<DL>* tag and ends with the *</DL>* tag. Each term to be defined in a definition list uses the *<DT>* or *Definition Term* tag. Every definition in a definition list needs a *<DD>*, or *I do'nt know what DD stands for :)* tag in front of it. It probably looks alot like this:

```
<DL>
<DT>HTML <DD>Hypertext Markup Language
<DT>XML <DD>eXtensible Markup Language
</DL>
```

Our friend Joe Browser see's it like this:

HTML

Hypertext Markup Language

XML

eXtensible Markup Language

Note that you can create really cool effects if you use the formatting tags inside of lists. It's really neat when you have **Bold** terms and *Italic* definitions (in my opinion anyway)!

Great! Now we can list all our family members and their cats on our page! What's next? [Click here for the next lesson!](#)

[Back to Blacksun's Mainpage](#)

Lesson 8: Adding Images to our Pages

Well, text is very important, but it can only take you so far. I'm sure that up until now, if anyone has seen your pages they're asking "Where are the pictures?". Yeah, I know Uncle Bob is dying to see the latest picture of you and the family on your website, so this lesson is about adding images to your pages.

Putting images on a web page is simple. It's probably even simpler than it was for you to get the certain image onto your computer! The ``, or *Image* tag is used when we want an image on our web page. When we use ``, we don't need to close it with a ``. To tell the browser where to load this image from, we use the SRC, or source, attribute. It looks alot like this:

```
<IMG SRC="jussmall.gif">
```

And it loads on to the page, just like this:



Note that we can use relative linking just like with in the `<A>` tag, like `SRC="../jussmall.gif"` if the image was in the directory above the current one, or `SRC="pictures/jussmall.gif"` if the image was in the pictures directory.

Because some browsers don't load images, and some people turn them off, we need a way to show them what this image is. The solution? The ALT attribute! ALT is an optional (but highly recommended) attribute. When a browser doesn't load an image, or when they are turned off, the text in ALT will be shown instead. In our `` tag it's used a little like this:

```
<IMG SRC="mypic.gif" ALT="Check this out - It's a picture of me!">
```

Need more control over your image's positioning? Try using the ALIGN attribute in it! The ALIGN attribute can be used to put an image in the left, right, middle, top, or bottom of a page. Is this more choices than you're used to? Let me explain. If we add `ALIGN="top"` to our `` tag, the browser will align the top of our image to the top of the current line. `ALIGN="bottom"` aligns the image to the bottom of the current line, and `ALIGN="middle"` aligns our image to the middle of the current line. Aligning our image to the left or right aligns it to the left margin, or right margin of the page. Got it? Ok, here's a little pop quiz. What will the following code do?

```
<IMG SRC="myimg.jpg" ALT="My image!" ALIGN="left">
```

If you said it adds the image myimg.jpg to the current page with alternate text "My image" and left aligns it, you're wrong! Heh heh, just a joke, you're correct ;).

All right! The image looks great but...Hey! What's this border doing around my picture? Oh yeah! We forgot about the BORDER attribute!

The BORDER attribute takes a number as an argument. This number will be the width of the border around your image. Quick, how do we get rid of the border with the BORDER attribute? Easy, we just

set !

Yuuup, we've got an image on the page. But wait: why does the browser wait to load the image before displaying the rest of the page? Well, the browser doesn't automatically know how big your image is. You can give it this information (and make your pages load faster!) with the WIDTH and HEIGHT attributes. We give to the WIDTH and HEIGHT attributes the width and height of our image in pixels. So it looks like this:

```
<IMG SRC="newimg.gif" ALT="A new image" HEIGHT=120 WIDTH=200>
```

Another benefit of specifying the WIDTH and HEIGHT in the tag is that you can make sure that the proper space is left for your image, even if the viewer has images turned off.

Is the space around your image a little cramped? Try adding the HSPACE and VSPACE attributes to your tag. These attributes add horizontal and vertical spacing around your image.

Want an image for the background of your page? Try adding the BACKGROUND attribute to the <BODY> tag. It's used a little like the SRC attribute to the tag. Here's an example:

```
<BODY BACKGROUND="mybackground.gif">
```

This would take mybackground.gif, and tile it in the background of our page. Be warned though, use the wrong background image and your viewers may be straining to see your text!

Think you're an HTML wizard? Ok hotshot, how do we make images into clickable links? Easy...surround them with an anchor tag! For example:

```
<A HREF="lesson9.html">  
<IMG SRC="nextlesson.gif">  
</A>
```

Results in this (click on it!):



[Back to Blacksun's Mainpage](#)

Lesson 9: Creating Image Maps

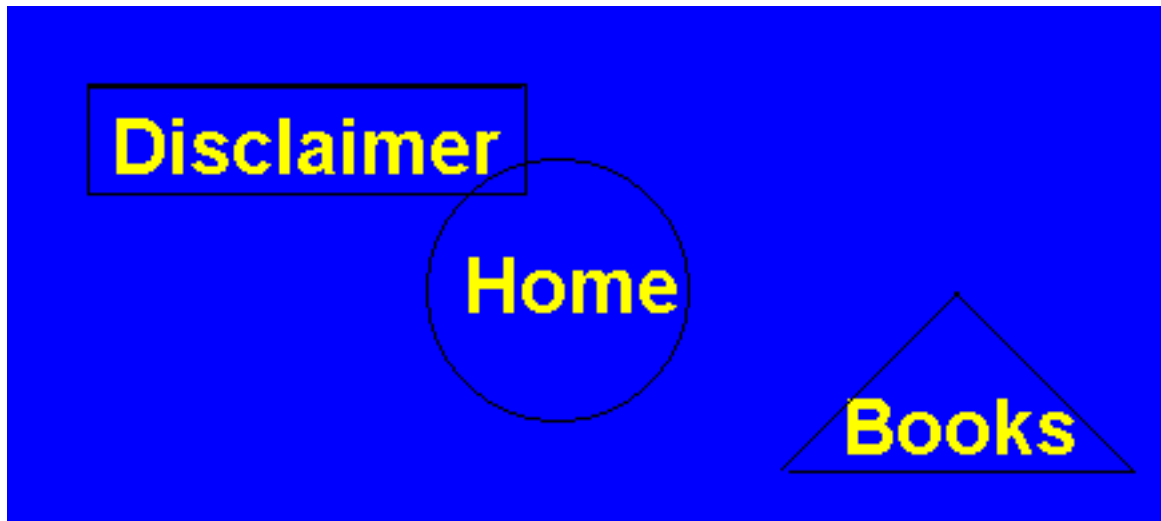
I'll bet you've surfed around the web for a while, and seen more than a few images on a variety of pages. Ever seen one of those images that you can click on different regions of to get to different pages? Think that's cool? In this lesson we'll be learning how to make them!

Any image that is subdivided into regions that point to different pages is called a *Image Map*. There are two ways that you can put an image map on your page: by using Client Side image maps or Server Side image maps. Client side scripting is when the coordinates for the image map is stored on an HTML page. Server side image maps store the coordinates for the image map on the page's web server. Because Client Side image map's are quicker loading and easier to understand, we will only explain how to implement Client Side image maps.

When we want to create an image map we use the `<MAP>` tag. The `<MAP>` tag, when put in the body part of our page, is our way of telling the browser "I'm going to put an image map on this page". The `<MAP>` tag takes one attribute: the NAME attribute. The NAME attribute gives our image map a name that we can call it with. It's just like calling a dog, you do'nt call it "dog" it's whole life, you name it something like "spot" and call it by name! Our basic `<MAP>` tag looks like this:

```
<MAP NAME="newmap">
```

Now let's say that I have an image that I want to use on my site as an image map, maybe one that looks something like this:



How do I make it into an imagemap? Well, there's really no physical way to change it into an image map. All that you can do is divide it into areas that link to different pages. We do this by using the `<AREA>` tag. The `<AREA>` tag takes three arguments: SHAPE, COORDS, and HREF. SHAPE is easy: it tells the browser what shape the said area is. SHAPE can be given three arguments, "rect" for rectangle areas, "circle" for areas that are circles, and "poly" for polygons that are not rectangles. So far, our `<AREA>` tag looks like this:

```
<AREA SHAPE="circle, rect or poly">
```

So how do we define the actual coordinates of our shape? As you've probably guessed, we use the COORDS attribute. For a rectangle, we have to pass the COORDS attribute the top left and bottom right

corners of our linking area.

How do we find these coordinates? Most image editing programs, such as [Paint Shop Pro](#) (free to try, my personal favorite!) show you what coordinates you're pointing at on the image. The first coordinate is the "x" or horizontal coordinate, and the second is the "y" or vertical coordinate (just like in algebra class!). I have already opened up my image in Paint Shop Pro and collected that the top left coordinates in the rectangle area of my image (the one surrounding "Disclaimer") are 30 (the "x" coordinate), and 30 (the "y" coordinate). I also got the bottom right coordinates, 194, 69. Now that we've done the hard part (getting the coordinates) all we have to do is pass them to the COORDS attribute. In our area tag, it looks like this:

```
<AREA SHAPE="rect" COORDS="30,30 , 194,69">
```

Whew! That's the most work we've done in this whole tutorial! But we're not done yet, we still need to pass where we want this section of our image to point to.

If we want our rectangle area to point somewhere, we're going to have to use one more attribute: HREF. The HREF attribute simply takes the *Hypertext Reference* (also called a link) that you want to use in this area as an argument. So our complete <AREA> tag looks like this:

```
<AREA SHAPE="rect" COORDS="30,30 , 194,69" HREF="../disclaimer.html">
```

(Note: Because our Disclaimer page is in the directory above the current one, if we use relative linking we need to add a '../' before our page name.)

So far we've flagged off the rectangular portion surrounding "Disclaimer" in our image to point to the page "disclaimer.html". But what about that circular portion around "Home"? This one's a simple switch, but let's take it one attribute at a time to make sure we get it.

The SHAPE attribute's argument has to change from "rect" to "circle", since we're no longer flagging off a rectangular area and now using a circular area. To get our coordinates, we're going to have to fire up the old graphics editing program. This time there are no corners to find, so we need to get the coordinates of the center of the circle and its radius (in pixels). For those of you who didn't take geometry in high school (shame on you!) the radius of a circle is the distance from the center of the circle to any point on the edge of the circle. Paint Shop Pro tells me that the center of my circle's coordinate's are 204, 106. It also tells me that the circle's radius is 57. Knowing this, we can change our <AREA> tag to look like this:

```
<AREA SHAPE="circle" COORDS="204,106 , 57" HREF="../index.html">
```

(Note: The "index.html" page, like the "disclaimer.html" page, is in the directory above the current one.)

You may notice that the circle and rectangle overlap a little bit. When there are overlapping area's in an image map, the area that is defined in the source HTML first is considered "on top". So if you click on the overlapping area and our rectangle was the first <AREA> tag in the source HTML, you would be taken to the Disclaimer page.

Wait just one minute! The last area in our image is a triangle? But the SHAPE attribute doesn't take "triangle" as an argument! How do we get around this? With the use of the "poly" argument of course!

The "poly" (or polygon) argument to the SHAPE attribute is used to define areas that are not rectangles or circles. In fact, you can define up to a 100 cornered polygon with the "poly" argument! The poly argument simply takes the coordinates of all the corners in the polygon, and connects them, dot to dot style. Since our lovely triangle has three corners, it has three coordinate pairs: 355,108 (top), 291,174 (bottom left) and 421,174 (bottom right). If we put them into our complete <AREA> tag, it looks like this:

```
<AREA SHAPE="poly" COORDS="355,108 , 291,174 , 421,174" HREF="../books.html">
```

Note that the browser automatically connects the last coordinate pair to the first coordinate pair to complete our polygon.

Now we have a complete image map, but what if we want the parts of our image that don't point anywhere yet to lead somewhere? That's where the "default" argument to the SHAPE attribute comes in. We could make "default" point somewhere, like this:

```
<AREA SHAPE="default" HREF="../magazines.html">
```

This would make the area not in the circle, rectangle or triangle point to "magazines.html" in the directory above the current one. But since I don't want this part of the image to point anywhere right now, I use the NOHREF attribute, which simply says "no link in this area!".

Our completed image map code looks like this:

```
<MAP NAME="newmap">
<AREA SHAPE="rect" COORDS="30,30 , 194,69" HREF="../disclaimer.html">
<AREA SHAPE="circle" COORDS="204,106 , 57" HREF="../main.php3">
<AREA SHAPE="poly" COORDS="355,108 , 291,174 , 421,174" HREF="../books.html">
<AREA SHAPE="default" NOHREF>
</MAP>
```

To put our image map on the page we use an tag with the USEMAP attribute. The USEMAP attribute tells the browser "use this image map". In this example, we're going to pass USEMAP the map that we made and named "newmap", like this:

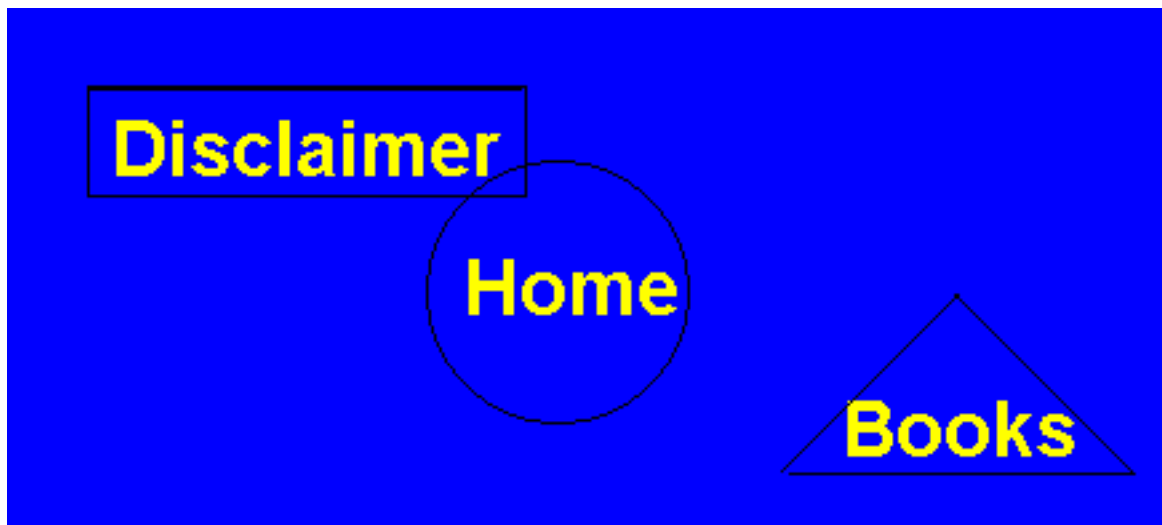
```
<IMG SRC="imagemap1.gif" USEMAP="#newmap">
```

Where imagemap1.gif is the name of my image. Take a look at how we had to put a "#" in front of the name of our map. This is because when you name a map, you flag it off so we can call it later. The "#" tells the browser that we're going to be using a certain part of an html page. Knowing this, we can use image maps from other pages like this:

```
<IMG SRC="imagemap1.gif" USEMAP="lesson9.html#newmap">
```

Where lesson9.html (this page) is the page that newmap is on.

Well that wasn't too hard! We've got our image map all divvied up, and it looks like this:



Not bad huh? I'll show you even cooler tricks in the [next lesson](#)!

[Back to Blacksun's Mainpage](#)

Lesson 10: Getting Down and Dirty with Tables

Format, format, format. HTML is all about format. HTML is all about format. Just the other day, a friend asked me "Hey little v, is there a way to format a part of my page to look like it's a spreadsheet?" I told him "Heck yeah! Just make a table!".

Here's a little description of tables from a great book, HTML 4 Unleashed by Rick Darnell: "Tables are kind of like lists. We're introduced to them at an early age through the mirth of games such as tic-tac-toe and checkers. Later in life, someone forces us to use a spreadsheet, and suddenly tables aren't so fun anymore...".

Tables were put in to HTML for the exact reason my friend wanted to use them for - spreadsheets and database information. Later, their features were expanded by developers to include attributes that allow much more than just that. Let's get started with creating a basic table, then get funky with some more advanced techniques!

So how do we start our own HTML spreadsheet? We use the `<TABLE>` tag!

The `<TABLE>` tag creates a blank table. It tells the browser "This is the start of a table". There is only one attribute to the `<TABLE>` tag that we need to worry now: the `BORDER` attribute. The `BORDER` attribute tells the browser how wide a border you want around your table data. If you don't include the `BORDER` attribute into your `<TABLE>` tag, the browser automatically sets it to `BORDER=0`, which means there will be no border around your table data.

So our basic table with no data looks like this:

```
<TABLE BORDER=1>
</TABLE>
```

What do we have to do to get some data in there? Add cells of course! A *cell* is a rectangular part of a table that can hold text, HTML tags, and even images! Those of us who use spreadsheets are already familiar with cells. When we make *rows* and *columns* in our table, we divide our table into many cells.

To create a row in a table, we use the `<TR>`, or table row tag. To divide our row into columns, we use the `<TD>`, or table data tag. You place your cells information after the table data tag.

The HTML for a table with 3 rows and 3 columns is like this:

```
<TABLE BORDER=1>
<TR><TD>Row 1, Col 1 <TD>Row 1, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 2, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 3, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
</TABLE>
```

And the table comes out looking like this:

Row 1, Col 1	Row 1, Col 2	Row 3, Col 3
Row 2, Col 1	Row 2, Col 2	Row 3, Col 3
Row 3, Col 1	Row 2, Col 2	Row 3, Col 3

Need a little explanation for your table? Maybe a title for your column? Try a table header! The `<TH>` tag is used to put a bold header (or headers!) on the top row of your table. It's used like this:

```
<TABLE BORDER=1>
<TR><TH>9 Cell Table</TR>
<TR><TD>Row 1, Col 1 <TD>Row 1, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 2, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 3, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
</TABLE>
```

And your browser shows it like so:

9 Cell Table		
Row 1, Col 1	Row 1, Col 2	Row 3, Col 3
Row 2, Col 1	Row 2, Col 2	Row 3, Col 3
Row 3, Col 1	Row 2, Col 2	Row 3, Col 3

We can change our table by using a variety of tags. Formatting a table combines parts of page format with image formatting.

Since we already know how to handle size in images, we can directly apply our knowledge to making tables. In our `<TABLE>` tag, we can add the `WIDTH` and `HEIGHT` attributes, just like in an image. For example, if we wanted a table 300 pixels wide and 300 pixels high, we'd code its `<TABLE>` tag like this:

```
<TABLE WIDTH="300" HEIGHT="300">
```

We can also change the height and width of the individual cells by adding the `HEIGHT` and `WIDTH` attributes to the `<TD>` tags. There's one other way to change our cells size though - we do this by passing the percentage of the table we want our cell to take up. For example, `<TD HEIGHT="50%" WIDTH="50%">` makes that cell take up 50 percent of the table's total height and 50 percent of the table's total width.

Following this trend of similarities between images and tables, we can also align data in a table in a similar fashion as an image. We just need to add the `ALIGN` or the `VALIGN` (vertical align) attributes to our `<TR>` tags. `ALIGN` can align our row to the "left", "right", or "center" and `VALIGN` can align our row to the "top", "middle" or "bottom". If we wanted a cell to take up what would normally be a few different cells, we use the `COLSPAN` attribute. `COLSPAN` takes as an argument the number of cells that this cell should take up. If I wanted to get a little crazy, I might align a row like this:

```
<TD ALIGN="right" VALIGN="bottom" COLSPAN="3">
```

And this would make a cell spanning 3 normal cells, and align our text in that cell to the bottom right portion of our row. How can `COLSPAN` and `ALIGN` improve our table? Take a look at our new, improved table and you'll see!

9 Cell Table		
Row 1, Col 1	Row 1, Col 2	Row 3, Col 3
Row 2, Col 1	Row 2, Col 2	Row 3, Col 3

Row 3, Col 1	Row 2, Col 2	Row 3, Col 3
--------------	--------------	--------------

As you can see, centering our top heading made our table look just a little better.

Does it get any better than this? Of course! We can give our whole table - or even individual rows or cells - their own background color. Can you guess how (heres a hint - remember how formatting a table is like formatting a page)? If you guessed "Use the BGCOLOR or BACKGROUND attributes" you're right! Do'n't be discouraged - you just learned how! These attributes are just inserted exactly as they would be in the <BODY> tag. So if i wanted the background of my sample table to be blue, and the top row to be red, I would just change the code to look like this:

```
<TABLE BGCOLOR="blue" BORDER=1>
<TR BGCOLOR="red" ALIGN="center"><TH COLSPAN="3">9 Cell Table</TR>
<TR><TD>Row 1, Col 1 <TD>Row 1, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 2, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 3, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
</TABLE>
```

And if we take another look it comes out like this:

9 Cell Table		
Row 1, Col 1	Row 1, Col 2	Row 3, Col 3
Row 2, Col 1	Row 2, Col 2	Row 3, Col 3
Row 3, Col 1	Row 2, Col 2	Row 3, Col 3

Notice that space is a little tight in those cells? Looks like we gotta add a little space between the borders and the text. How is this done you ask? We need to set the CELLSPACING and CELLPADDING in our table! The CELLSPACING attribute sets the amount of space between cells in our table, in pixels. The CELLPADDING attribute is used to set the amount of space between the cell borders and the cell data. If we add CELLPADDING="0" to our table, it makes the table's data align as closely as possible to the borders. Got it? Let's update our table!

```
<TABLE BGCOLOR="blue" BORDER=1 CELLSPACING=5 CELLPADDING=10>
<TR BGCOLOR="red" ALIGN="center"><TH COLSPAN="3">9 Cell Table</TR>
<TR><TD>Row 1, Col 1 <TD>Row 1, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 2, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 3, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
</TABLE>
```

Yields this result:

9 Cell Table		
Row 1, Col 1	Row 1, Col 2	Row 3, Col 3

Row 2, Col 1	Row 2, Col 2	Row 3, Col 3
Row 3, Col 1	Row 2, Col 2	Row 3, Col 3

Let's say we want our table to have a little caption that identifies what it is. We can do this by adding a `<CAPTION>` to our table. It's use is pretty simple, so i'll just show you how it's done.

```
<TABLE BGCOLOR="blue" BORDER=1 CELLSPACING=5 CELLPADDING=10>
<CAPTION>A sample table</CAPTION> <TR BGCOLOR="red" ALIGN="center"><TH
COLSPAN="3">9 Cell Table</TR>
<TR><TD>Row 1, Col 1 <TD>Row 1, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 2, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
<TR><TD>Row 3, Col 1 <TD>Row 2, Col 2 <TD>Row 3, Col 3</TR>
</TABLE>
```

Makes our table look like so:

A sample table

9 Cell Table		
Row 1, Col 1	Row 1, Col 2	Row 3, Col 3
Row 2, Col 1	Row 2, Col 2	Row 3, Col 3
Row 3, Col 1	Row 2, Col 2	Row 3, Col 3

Captions are great if you need to say "This is my table" or something along that line, but are completely optional.

Is your head spinning with all the information? Here's a few things to help you regain your senses:

1. Mail me at little_v@your-house.com with any questions.
2. [Go on to the next lesson!](#)

[Back to Blacksun's Mainpage](#)

Lesson 11: Oh my god, I've been framed!

Silly titles aside, frames are one of the most interesting and beautiful things any page can have. Be warned though - they can also be a design disaster, namely when a browser does not support frames (pre-Netscape 2.0 browsers do'nt).

A *frame* is a distinct section in a browser window. This section can be independent of the other sections in the window, and if we allow it, it can be resized. Like just about everything we've covered so far, making a web site with frames is easy. In fact, many people think creating tables are harder!

Creating frames is just a little different from making normal web pages. To make a frame, we need to create a *frame layout page* that links to the divisions of the page. On the layout page we put the <FRAMESET> and <FRAME> tags that tell the browser how to display the frames.

The <FRAMESET> tag has 3 attributes, ROWS, COLS and FRAMEBORDER. The ROWS attribute tells the browser "I (number) rows of frames, this size" by putting it into the <FRAMESET> tag like this: <FRAMESET ROWS="33%, 33%, 34%">. This would put 3 different frames on the page, separated like rows in a table. The COLS attribute works similarly, except it divides the page into columns instead of rows.

The FRAMEBORDER attribute specifies whether you want your frame to have borders or not. This attribute's use is a bit restricted though: you can only pass it either 0 or 1 as an argument. FRAMEBORDER="0" eliminates borders, and FRAMEBORDER="1" turns borders on.

So far, our bare-bones frame layout page looks like this:

```
<HTML>
<HEAD>
<TITLE>Welcome to Little v's frame demo!</TITLE>
</HEAD>

<FRAMESET COLS="40%, 60%">
</FRAMESET>

</HTML>
```

Notice anything missing? Yep, the BODY! However, this is justified: since this is just a frame *layout* and not an actual page with information, it really does'nt have a body. Hmm, that did'nt sound too healthy. Oh well, now for the business of displaying our actual frames.

To define the content of the two frames that we just created, we need to use the <FRAME> tag. The FRAME tag has a few attributes that give us extensive control of our frames. The most important attribute is the SRC attribute. This attribute gives the location of the HTML file to be displayed in the frame. Basically, <FRAME SRC> works for frames the same as does for images.

After the mandatory SRC tag, there are a few more attributes to the <FRAME> tag that we can use to give us control over how our frame is displayed. We can toggle the borders on or off in our individual frames by putting the FRAMEBORDER attribute in the <FRAME> tag. We can change the margins in our frame with the MARGINHEIGHT and MARGINWIDTH attributes. The proper way to set a margin

is like this:

```
<FRAME MARGINWIDTH="x">
```

Where x is any number. What if we want our frame to be fixed so the browser cannot resize it? We could tape up the browser's keyboard, or we could use the NORESIZE attribute. This attribute takes no arguments, and when placed in the <FRAME> tag, it locks out the resize option in the browser. If we want to get really specific as to the display of our frame, we can use the SCROLLING attribute to set whether the browser shows a scrollbar. We can either pass SCROLLING a yes, no, or an auto (sorry, no maybe's!). Auto allows the viewer's browser to determine whether or not it needs a scrollbar. The default is auto.

When we apply what we just learned and update our "bare-bones" frame layout, the code looks like this:

```
<HTML>
<HEAD>
<TITLE>Welcome to Little v's frame demo!</TITLE>
</HEAD>

<FRAMESET COLS="40%, 60%" FRAMEBORDER="0">
<FRAME SRC="leftframe.html"> <FRAME SRC="rightframe.html"> </FRAMESET>

</HEAD>
</HTML>
```

Now if we crank out one page for the left frame and one for the right frame, we'll have something like [this \(click here for the example\)](#).

But what about if our casual browser is surfing with frames off, or even worse, what happens if his browser doesn't even support frames? Always ready with a work-around, HTML offers us the <NOFRAMES> tag. It works like this: anything between the <NOFRAMES> and </NOFRAMES> tags on the frame format page will be shown to browsers that cannot display frames, or have frames turned off. A clever use for this tag would be to add a message with a link to the main page of your site, so these visitors can navigate your site without use of the frames. Something along these lines would work:

```
<NOFRAMES>
I see you can't see my frames, but you can still navigate my site the old fashioned way! <A
HREF="mainpage.html">Click Here!</A>
</NOFRAMES>
```

That just about sums it up for frames. Go and read the [summary](#) for advice on applying what you've learned, and how to advance your knowledge of html.

[Back to Blacksun's Mainpage](#)

Summary

Wow! You've made it a long way! Pat yourself on the back, and take a vacation. You deserve it!

Furthering your knowledge

There are many great HTML books and periodicals. Go to the [blacksun books](#) page to see the list.