

Absolute and Relative URLs

by Joel McKinney

URLs are the addresses that servers use to find a page on the web. Every page has its own unique URL. In building your web site you may have to define how the browser is to find specific pages within your site. The way the URL is assembled tells the browser where to look.

The first part of the URL is called the “scheme,” and this tells the browser whether to look on a server, or a folder on your computer, or an ftp site for downloading something. It also tells which server to look on.

For your web site you will use Hypertext Transfer Protocol, which reads: “http://”

You will notice that while working on your web site offline the URL will begin with “file:/// (your computer name/... etc.)”

The scheme, then will be “http://”

The next part of the URL is the “server name” and the University server is “usd.edu” This will be different if your site is hosted by a private company, for instance “deltanet.com” or some such.

The server name is “usd.edu” which brings us to “http://usd.edu”

The next part of the URL is called the “path” and this part tells the browser where on the server to look. Here at USD your path will be “/~” followed by your e-mail account name: “/~jmckinne” for example. Off campus you may not have to use the tilde “~”.

The path is “/~jmckinne” which brings you to: “http://usd.edu/~jmckinne”

This is enough to open a page on your browser because browsers, by default, look for a file named “index.html” (or “index.htm”). If no such file name exists you will probably see a list of files that are at that location, which doesn’t look very professional. This is why we name our “splash page” “index.html”.

Inside your path you may want divisions, or folders. There are a number of reasons for this, most of which are beyond the scope of this tiny tutorial, but in my case the need to keep track of hundreds of files made it necessary for me to use folders. One is named “blackhawk” and in order to find that I have to add to our existing URL “/” which means “find folder named” and the name of the folder to find.

The folder name is “blackhawk” which brings us to: “http://usd.edu/~jmckinne/blackhawk”

There is a file in “blackhawk” named “index.html” so I don’t need to type any file name, but if I wanted to send the browser to another page, for instance the page named “exhibitions.html” I would use the file name, “exhibitions.html” in this case, and the “/” to tell the browser to search in this folder for this file.

The file name is “exhibitions” which brings us to “http://usd.edu/~jmckinne/blackhawk/exhibitions.html”

Up to now we have been writing an “Absolute URL” but once we are inside a site we do not need this much information to navigate around inside that site. Inside we can use “Relative URLs.” If I want to go to another page in “blackhawk” “ceramics.html” for example, I need only specify “ceramics.html” because the browser is already in “blackhawk”. This is a relative URL, relating to where we already are (using this scheme, in this server, this path, and this folder).

So we link to “ceramics.html”

If we need to link to another folder, but still this server and path, we would type “../” to tell the browser to look for a different folder, but one that is located in the same folder that contains the folder where we are at the moment. I know that sounds confusing, so imagine we are within the path “/~jmckinne” and are in the folder “/blackhawk” which is in the folder “www”. There are three folders in “www”, one is “blackhawk” one is “edfoundry” and one is “artwork”. From a page in “blackhawk” to get to a page in “edfoundry” we would type a link thus: “../edfoundry/index.html” This tells the browser to stay in “www” but find a folder named “edfoundry” and find the file named “index.html” inside.

“../edfoundry/index.html”

The shortened links “ceramics.html” and “../edfoundry/index.html” are relative links. To go outside “/~jmckinne” to Ruth’s site for instance, we would have to specify an absolute URL: “http://usd.edu/~rmckinne”