CoffeeCup

Absolute vs. Relative Paths/Links

Still today, one of the more tricky and confusing things about HTML is linking to other pages and sites, especially when absolute and relative paths come into play. But worry not! Creating links — relative and absolute alike — is actually fairly easy. Read on, and by the end of this article, you'll know the difference between these two types of links, as well as when and how to use them.

Of course, it's still important to understand how relative and absolute links work, so read on...

First off, as you may or may not know, you would use the following code to create a link in HTML:

```
<a href="linkhere.html">Click Me</a>
```

linkhere.html would be the page you want to link to, and **Click Me** would be the blue, underlined link that the page displays.

In the example above, we used a relative path. You can tell if a link is relative if the path isn't a full website address. (A full website address includes **http://www.**) As you may have guessed, an absolute path does provide the full website address. Here are a few basic examples of relative and absolute paths:

Relative Paths

- index.html
- /graphics/image.png
- /help/articles/how-do-i-set-up-a-webpage.html

Absolute Paths

- http://www.mysite.com
- http://www.mysite.com/graphics/image.png
- http://www.mysite.com/help/articles/how-do-i-set-up-a-webpage.html

The first difference you'll notice between the two different types of links is that absolute paths *always* include the domain name of the website, including **http://www.**, whereas relative links only point to a file or a file path. When a user clicks a relative link, the browser takes them to that location on the current site. For that reason, you can only use relative links when linking to pages or files within your site, and you must use absolute links if you're linking to a location on another website.

So, when a user clicks a relative link, how does their browser know where to take them? Well, it looks for the location of the file *relative* to the page where the link appears. (That's where the name comes from!) Let's get back to our first example:

```
<a href="linkhere.html">Click Me</a>
```

This link points to a filename, with no path provided. This means that **linkhere.html** is located in the same folder as the page where this link appears. If both files were located in the root directory of the Website **http://www.website.com**, the actual website address the user would be taken to is **http://www.website.com/linkhere.html**. If both files were located in a subfolder of the root directory called **files**, the user would be taken to **http://www.website.com/files/linkhere.html**.

How about another example? Let's say we our http://www.website.com domain had a subfolder called **pictures**. Inside the pictures folder is a file called **pictures.html**. The full path to this page would be:

```
"http://www.website.com/pictures/pictures.html"
```

Still with us? Good. Let's say in this pictures.html file, we have a link:

```
<a href="morepictures.html">More Pictures</a>
```

If someone clicked that, where do you think it would take them? If you said **http://www.website.com/pictures/morepictures.html**, you'd be right! You probably know why it would take them there: because both files are saved in the **pictures** subfolder.

Now, what if we wanted to use a relative link to show a page in another folder? If you want to link to a file in a subfolder of the current folder, provide the file path to that file, like so:

```
<a href="/pictures/tahiti-vacation/tahiti.html">Read about my Tahiti vacation.</a>
```

In this example, you're telling the browser to look in the current folder (**pictures**) for a subfolder (**tahiti-vacation**) that contains the file you want the user taken to (**tahiti.html**). You can link to as many subfolders as you need using this method.

What if you want to link to a file in a folder above the current folder? You have to tell the browser to move up one folder in your relative link by putting two periods and a slash (**../**) in front of the filename or path:

```
<a href="../about.html>Learn more about my Website.</a>
```

When the browser sees **../** in front of the filename, it looks in the folder above the current folder. You can use this as many times as you need to. You can also tell the browser to look in a subfolder of the directory above the current one. Using the same example website from above, let's say we wanted to create a link that would take the user to a page called **stories.html** located in another folder called **stories**. This folder is located in the root directory, one folder up from the current folder, **pictures**. Here's how a relative link to this file would look:

```
<a href="../stories/stories.html">Read Stories</a>
```

Now, let's talk about absolute paths. Like we mentioned earlier, absolute paths provide the complete website address where you want the user to go. An absolute link would look like this:

```
<a href="http://www.coffeecup.com">Click here to visit CoffeeCup Software.</a>
```

You *must* use absolute paths when linking to another Website, but you can also use absolute paths within your own website. This practice is generally frowned upon, though. Relative links make it easy to do things like change your domain name without having to go through all your HTML pages, hunting down links and changing the names. As an added bonus, they force you to keep your site structure neat and organized, which is always a good idea.
