Absolute vs Relative Path - Which Should You Be Using? - KeyCDN

By Cody Arsenault

Updated on August 18, 2017

Absolute vs Relative Path - Which Should You Be Using?

Creating and keeping up with web links can be a challenge, especially if you don't understand the difference between absolute and relative paths. These conflicting approaches to connecting webpages can have a significant impact on a website's performance and SEO value. This guide will explain the two types of links and help you determine the right path for any situation.

Relative path vs absolute path: What do they mean?

Absolute paths contain a complete URL, which includes a protocol, the website's domain name and possibly a specific file, subfolder, or page name. For example:

```
<a href="https://trends.google.com/trends/">Google Trends</a>
```

The URL here, `https://trends.google.com/trends/`, can be entered into a browser's search bar, and you'll be taken where you want to go. While your personal browser may let you omit the protocol, `https://`, you should always include the scheme (e.g. `http://` or `https://`) when coding absolute links to make sure they work for all visitors.

Conversely, a relative link only includes the name of a specific file or page, which is relative, to the current path. If you keep all of your website's files in a single directory, you can establish links between pages as follows:

```
<a href="page2.html">Next page</a>
```

In this case, the link is only valid within the directory that `page2.html` is located. When clicked, the browser searches the current page's directory for `page2.html` and displays it. Of course, if you just pasted `page2.html` into a browser's search bar, it wouldn't take you where you wanted to go. Therefore, the link's path is **relative to the current document being displayed by the browser**, hence the term relative path.

You would need to include a protocol and your domain name within the link to turn the same example as above into an absolute path link:

```
<a href="https://yourwebsitename.com/pages/page2.html">Next page</a>
```

Although some developers are moving away from using relative links to limit potential content duplication, they can still be useful in some situations.

Creating a relative path for connecting website elements

Sometimes, you need to include more information than a file's name to create a relative path. Going back to the last example, if `page1.html` resides in the root directory while `page2.html` resides in a subdirectory named `folderA`, a relative link from `page1.html` to `page2.html` would have to **include the folder name** followed by a forward slash:

```
<a href="folderA/page2.html">Next Page</a>
```

Keep in mind that folder and file names are **always case sensitive in URLs**! Domain names, however, are never case sensitive.

Let's say you want to create a link on `page2.html` that takes you back to `page1.html`. To create a relative path to a higher directory, you simply tack on two periods followed by a forward slash at the beginning of the link:

```
<a href="../page1.html">Previous Page</a>
```

The addition of `../` tells the browser to look further up the folder hierarchy to find the desired file.

Now, imagine a folder structure with a root directory that contains `folderA`,

and `folderA` contains `folderB`. If `page1.html` is in the root directory, and `page3.html` is in `folderB`, you could create a link from `page3.html` to `page1.html` as follows:

```
<a href="../../page1.html">Back to Page 1.</a>
```

You can just keep adding `../` every time you want to move up one directory, but what if you need move up and then move down? For example, imagine this same root directory also has a folder called `subfolder`, which contains the file `contactpage.html`. If `page2.html` is in `folderA`, and you want to create a relative path from `page2.html` to `contactpage.html`, you would use the following link format:

```
<a href="../subfolder/contactpage.html">Contact US</a>
```

Creating an absolute path for connecting different websites

While relative links can be useful for linking internally, absolute links are useful for creating links both between separate websites and internally. If linking to an external website, it's best to target a blank browser window so that the link opens in a separate window rather than directing visitors away from your website. That way, the user will still see your site when they close the linked page. To instruct the browser to open a link in a new window, you must add the `target="_blank"` attribute. For example, you would make a link to the Google homepage like this:

```
<a href="https://www.google.com" target="_blank">Google Homepage</a>
```

Whether the link opens in a tab or a window depends on the user's browser settings. On the other hand, relative links connecting pages within your own website should always load in the same browser page.

Whether you use relative paths or absolute paths there are pros and cons to each method. If you're just linking pages that are all on the same website, traditional wisdom holds that there is no need to use absolute links. If you ever change your domain name, keeping all of your links relative will make the transition much easier. However, using absolute URLs does have some **SEO advantages**, which will be discussed later on.

What about relative protocol URLs?

Certain websites allow users to request assets both over HTTP and HTTPS. If the current page is being viewed over HTTPS then an asset using a relative protocol will be delivered over HTTPS. On the other hand, if the site is being viewed over HTTP then the asset will also match that protocol and be delivered over HTTP. The following is an example of what a link using a relative protocol looks like:

```
<a href="//yourwebsite.com/images/image.jpg">My Image</a>
```

Relative protocols can be useful if you have scripts or images that exist in both secure and non-secure areas of your website. However, since **HTTPS is now considered standard**, you should no longer require the use of relative protocols. Check out our complete guide on how to migrate from HTTP to HTTPS if you haven't already.

Furthermore, relative protocol URLs are not ideal for linking between pages, nor should they be used in conjunction with canonical tags. Doing so may create duplicate content problems, which could cause web crawlers to crawl the same page in both HTTP and HTTPS modes. For websites that are transitioning exclusively to HTTPS, it's best to just include the full protocol such as:

```
<a href="https://yourwebsite.com/images/image.jpg">My Image</a>
```

Relative path vs absolute path: Which is better for SEO?

When it comes to SEO, consistent use of **absolute URLs is preferable** even for internal links. If you have a separate staging environment, you can likely configure your CMS to generate absolute URLs dynamically based on the current server environment. For example, WordPress automatically generates absolute URLs for links that are not embedded into content.

If parts of your staging site are publicly accessible, it's very important to avoid using relative URLs for your site navigation links. If a single incorrect link gets deployed to production, then it may open up your entire staging site to being crawled, indexed and displayed in search results. Having multiple copies of your website floating around in the search results can pose a serious security threat on top of duplicate content issues.

Relative links are also problematic for RSS feeds. Consistent use of absolute links ensures that content always displays correctly, and it gives you better protection against scrapers. Using absolute links for images allows you to assert ownership in the image search results, which reduces the likelihood of duplicate content popping up on a competitor's website.

Canonizing your links

If you're having issues with duplicate content due to relative links, and you don't have the time or resources to fix all of your links, canonizing all of your pages can alleviate the problem. The canonical tag indicates that a URL leads to the master copy of a webpage, which is the version that displays in search results:

canonical header example

The canonical tag also comes in very handy when implementing a CDN. Since CDNs hold a cached copy of your website's assets and deliver them from a separate URL in most cases, crawlers may see this as duplicate content. However specifying the canonical tag alleviates this problem and **properly points crawlers to the original resource**.

KeyCDN offers the canonical header feature to protect you from any duplicate content issues caused by using a CDN.

Summary

If you already have a website full of relative links, you should consider working towards changing them to absolute links for the reasons mentioned above. In some cases such as WordPress, you can use a plugin such as Better Search Replace to modify certain links. For instance, this would be useful if your site contains many relative protocol links and you would like to change them to use

the absolute protocol URLs. Although be careful with this approach as if you are linking to an external site which doesn't support HTTPS, your users will receive an error when clicking on said link.

replace links plugin

As for relative paths, these would need to be changed manually in most cases. Although relative paths are not something that needs to be modified urgently, it's important to take their pitfalls into consideration moving forward.