# Demo Script: 1.1.7 Links and Paths

* [Replit Demo](https://replit.com/@mrrocampbell/1-1-7-links-and-paths#index.html)
* [Word Version of Script](https://chegg-my.sharepoint.com/:w:/p/rocampbell/Ef_MHRAcm79Op_f2lCuv1kgBET7hCkh6xHrAfoUXEbMGWg?e=uZAbVI)
* [Contentful lesson](https://overview.thinkful.com/preview/JS-MODULAR/version/1/introduction-to-web-development/guild-b2b-eng-module-introduction-html-css-fundamentals/modeng-links-and-paths)
* Jira Tickets:
  + [TFENG-2316: 1.1.7\_DemoVideo\_intro\_html\_css\_fundamentals\_links\_paths](https://chegg.atlassian.net/browse/TFENG-2316)
  + [TFENG-1910: Add video - Links & Paths](https://chegg.atlassian.net/browse/TFENG-1910)

## Script Start

Hey, this is Roger from Thinkful. In this video, we will learn how to utilize file paths and web URLs to link to web pages and display images within our HTML websites.

Let us start by creating a link that, when clicked, navigates our users to an external webpage, such as the Apple.com homepage. **Note:** I will show students the link working for context in replit\_

<a href='apple.com'>Apple Homepage</a>

We will accomplish this by utilizing the anchor or a tag.

***Note:*** *Show the students a fully developed a tag for reference before actually coding one out for them to see step by step*

<a href='apple.com'>Apple Homepage</a>

The anchor tag has a few different parts that allow us to accomplish this task. The first is the opening and closing tags.

<a></a>

Next, we need to provide a title for the link rendered on our website. This goes in the space between the opening and closing tag. Let’s make the title Apple Homepage:

<a>Apple Homepage</a>

Now if we look at our website, we can see that Apple Homepage renders on screen as normal text.

***Note:*** *I will show the students in replit how the site renders in its current state*

To make this anchor tag a true link, we must provide it with a href attribute, or hypertext reference attribute, which goes within the opening anchor tag:

<a href=""></a>

The href attribute is where you define the path or destination that the anchor leads to. Let’s set the path for this link to direct the user to the website Apple.com:

<a href="https://www.apple.com/"></a>

https://www.apple.com/ is an absolute link path because it is a full URL path that links a user to content outside of the current website.

***Note:*** *Highlight the “https://www.apple.com/” portion of the code*

Now if we view our webpage, you will notice that the link’s normal text Apple Homepage has changed. It now renders as a blue underlined word. This change is because the href attribute tells our browser to render our HTML anchor tag as a clickable link.

***Note:*** *Highlight the link in the display portion of replit so students can see how it displays on the website*

If we click the link, we are then navigated to the Apple homepage.

***Note:*** *I will demonstrate this in replit so the students can see the link work in real-time.*

Now that we know how to use an absolute link path to navigate users to an external webpage, how would we handle creating a link that guides users to a webpage that is a part of the website we are creating?

This is where a relative link path comes into play. Relative link paths are used to connect one file to another based on their location within a server.

Let’s say you are developing a portfolio website to show all the awesome projects you have been creating while learning how to code.

You have decided that your homepage will have a link that visitors can click to view your portfolio project webpage.

***Note:*** *I will show students a basic site that has this functionality as I talk about it.*

<!-- index.html -->  
<!DOCTYPE html>  
<html>  
  
<head>  
 <meta charset="utf-8">  
 <meta name="viewport" content="width=device-width">  
 <title>replit</title>  
 <link href="style.css" rel="stylesheet" type="text/css" />  
</head>  
  
<body>  
 <a href="./portfolio.html">Portfolio Page</a>  
  
 <h1>Welcome to my home page!</h1>  
 <script src="script.js"></script>  
</body>  
  
</html>

<!-- portfolio.html -->  
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta http-equiv="X-UA-Compatible" content="IE=edge">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Document</title>  
</head>  
<body>  
 <a href="./home.html">Home Page</a>  
  
 <h1>Welcome to my portfolio page</h1>  
</body>  
</html>

This would be an opportune time to deploy a link that uses a relative link path to navigate users from one HTML file to another within your website.

So, to start, let’s look at our current file structure. We have two files created with some started code.

The first is an index.html file which is set up as our homepage. It has an h1 tag that says, “Welcome to my portfolio home page!” and above the h1 tag, there is an anchor tag with the title “Portfolio Page”.

Our second HTML file is portfolio.html, which also has an h1 tag with a title.

So how do we tell the link in our index.html file to navigate users to the portfolio.html file? We do this by providing the anchor tag with a relative link path.

By modifying our href with the relative link paths ./portfolio.html, our browser can locate the portfolio.html file within our portfolio project file structure.

The ./ tells our browser to look within the current folder we are within for the portfolio.html file.

Now let’s look at this in action. If we click the “Portfolio Page” link, we are navigated from the home page to the portfolio page! How cool is that?!

Now that we know how to use relative link paths and Absolute link paths to navigate users to internal and external webpages, it is time to discuss how to use them to display images.

We utilized the same logic with anchor tags but used the img tag instead.

We want to render an image we found on google on our website. This is a prime opportunity to use an Absolute link path.

<img src="https://pixadunes.com/wp-content/uploads/2019/10/PIXA002017-U.jpg"/>

As you can see, rather than providing the src attribute with an absolute link path to a different webpage, we have provided it with one that leads to the location on the internet where the picture lives. Which then allows our browser to render it on screen.

***Note:*** *I will show how the code above renders within replit*

The last thing we will look at is how to render an image that lives within our website’s file structure. Let’s say we saved a copy of our image within a folder called assets.

***Note:*** *Show students the image saved within the folder structure*

How would we write a relative link path so that the browser could render the image? We can do this by providing the following path ./assets/groot.png.

As you can see, we can now render the Groot image with a relative link path.

Now take what you have learned to add links and images to a website. Thanks for watching! See you in the next video.