**[The Front End (OLD)](https://www.theodinproject.com/paths/foundations/courses/foundations" \l "the-front-end-old)**

Skip this section. These lessons will be removed shortly. We've left them up for a short time as a reference for people who were in progress when the new material was added.

[1. Introduction to the Front End](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end)

Top of Form

Bottom of Form

Top of Form

Bottom of Form

[2. HTML and CSS Basics](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics)

Top of Form

Bottom of Form

Top of Form

Bottom of Form

[3. Developer Tools](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools)

Top of Form

Bottom of Form

Top of Form

Bottom of Form

[4. **Project: Google Homepage**](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css)

Top of Form

Bottom of Form

# [Foundations](https://www.theodinproject.com/paths/foundations/courses/foundations)

## Introduction to the Front End

[Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#introduction)

[Learning Outcomes](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#learning-outcomes)

[Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#assignment)

[Additional Resources](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#additional-resources)

[Knowledge Check](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#knowledge-check)

### [Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#introduction)

The “front-end languages” live in the browser. After you type an address into the address bar at the top and hit the enter/return key, your browser will receive at least an HTML file from the web server. That file will likely tell the browser to request a CSS file and a JavaScript file as well (probably many more than one, but we’ll keep it simple).

Each of these languages performs a separate but very important function and they work harmoniously together to determine how the web page is STRUCTURED (HTML), how it LOOKS (CSS), and how it FUNCTIONS (JavaScript). And keep in mind that your browser handles figuring out how to make these files into a functioning web page (not the server).

Front-end web development is NOT design (you won’t be playing around in Photoshop or anything), but a front-end developer does apply the work of designers to the web page by translating their well-designed layouts into real code. The front-end developer stands between the designer on one end and the back-end developer on the other, translating the design into code and plugging the data from the back-end developer into the right spots. They must also handle all the possible interactions that the user may need to make with the page.

On the front end, you will need to be highly conscious of who your user is and how they will be interacting with your web page, because you are building their gateway to your page or product. This may mean gaining a strong understanding of accessibility and things like responsive development down the line, but first you need to build up your toolkit and pick up the fundamentals of the front-end languages.

In the following lessons, you’ll get a healthy understanding of each of the three front-end languages. To get warmed up, we’ll start at the high level.

### [Learning Outcomes](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#learning-outcomes)

Look through these now and then use them to test yourself after doing the assignment:

* What is the role of HTML in a web page?
* What is the role of CSS in a web page?
* What is the role of JavaScript in a web page?

### [Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#assignment)

1. Read a brief intro to front-end development from this [interview with Nick Schaden](https://web.archive.org/web/20200601022721/https:/generalassemb.ly/blog/what-is-front-end-web-development/).
2. Read about the [role of HTML](https://webapps-for-beginners.rubymonstas.org/html.html) from this Ruby Monstas page.
3. Get a high-level overview of how all three languages work together in MDN’s [Getting started with the web](https://developer.mozilla.org/en-US/Learn/Getting_started_with_the_web). Learners should only read: [Dealing with files](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/Dealing_with_files), [HTML basics](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics), [CSS basics](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics) and [JavaScript basics](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics). There is no need to implement the website; simply read about the process at this point.
4. Get an overview of what you can do with these documentation pages: [HTML](https://developer.mozilla.org/en-US/docs/Web/HTML/Element), [CSS](https://developer.mozilla.org/en-US/docs/Web/CSS/Reference#Keyword_index), and [JavaScript](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference). You will not understand most of what is there just yet, but bookmark the pages for later reference.
5. Bookmark [DevDocs.io](https://devdocs.io/). Read the “Welcome” message. Massive API documentation collection that even works offline. Essential collection of reference material for everything covered and more. (Maintained by [FreeCodeCamp](https://www.freecodecamp.org/))

### [Additional Resources](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#additional-resources)

This section contains helpful links to other content. It isn’t required, so consider it supplemental for if you need to dive deeper into something.

* Watch [this awesome video](https://www.youtube.com/watch?v=gT0Lh1eYk78) explaining the big picture of the three core web technologies: HTML, CSS and JavaScript.
* [This video](https://www.youtube.com/watch?v=BANChTkxYYY&list=PLwqG3V3cExWpCgHOcLEKg6z-InpjHr7MB) is another great introduction to how the various front-end technologies interact.
* [Skills of a Successful Front-End Web Developer](https://web.archive.org/web/20151110193658/https:/www.drupal.org/node/1245650) from Drupal (a CMS based on PHP).

### [Knowledge Check](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/introduction-to-the-front-end#knowledge-check)

This section contains questions for you to check your understanding of this lesson. If you’re having trouble answering the questions below on your own, review the material above to find the answer.

* Which language is responsible for the semantic structure of a document?
* Which language can change the font, text-size, or background-color of an element?
* Name a few behaviors that javascript can be responsible for on a website.
* What is the purpose of an opening tag and closing tag in an HTML element?
* What is the selector in a CSS ruleset?
* What is the property and property value in a CSS ruleset?
* What is the declaration in a CSS ruleset?
* How do you reference a file in the same directory as your HTML file?
* How do you reference a file in a directory below your HTML file?
* How do you reference a file in the directory above your HTML file?

# [Foundations](https://www.theodinproject.com/paths/foundations/courses/foundations)

## HTML and CSS Basics

[Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#introduction)

[Learning Outcomes](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#learning-outcomes)

[Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#assignment)

[Additional Resources](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#additional-resources)

[Knowledge Check](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#knowledge-check)

### [Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#introduction)

HTML is the markup that contains all the actual stuff that a web page has. All the text on this page you’re reading right now lives inside HTML tags that tell your browser how to order the content on the page. Go on, right click any element on the page and choose “Inspect Element” to open up your browser’s Developer Tools and it will show you the structure of the page.

[CSS](https://skillcrush.com/2012/04/03/css/) tells the browser if you want to display any of those tags a particular way, for instance, turning its background blue and pushing it a little to the left. In your Developer Tools, you can see the CSS styles in another panel, usually showing which specific properties were inherited from which lines of CSS.

### [Learning Outcomes](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#learning-outcomes)

Look through these now and then use them to test yourself after doing the assignment:

* Why do we separate HTML and CSS?
* What are classes and IDs (and how are they different)?
* What are elements?
* What are tags?
* What are attributes?
* What are forms?
* What is a div?
* What are selectors?
* What are properties?
* What are values?
* What is the “query string” in a URL and what does it do?
* What is the difference between “pixels” and “ems”?
* How do CSS styles for a particular element get inherited? i.e. how does an element get its “default” styles?
* What are two CSS attributes you can change to push an element around on the page?
* What are the three different ways to include CSS in your project or use CSS to style a particular element?
* What is the “default stylesheet” or “user agent stylesheet”?
* Why use a CSS reset file?

### [Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#assignment)

1. Dive right into learning HTML and CSS with [freeCodeCamp’s interactive tutorials](https://www.freecodecamp.org/learn/responsive-web-design/) to introduce you to these new concepts. Do only the courses/lessons in the linked section, do not do any of the other sections/certifications such as Javascript Algorithms”. It’s long and, if you’re new, might take you at least a few days to get through. **This resource is meant to help you develop familiarity, so do not expect to memorize everything.** Instead, expect to google these concepts in the future when you need to use them. You do not need to do the projects at the end of the section, you’ll have plenty of time to practice your skills in future lessons. (A large part of their proposed ‘300 hours’ for completion comes from these projects, so it probably won’t take you that long if you only do the exercises).

**Note:** If you are using a browser add-on such as Dark Reader (or a similar add-on that creates a “dark mode” for a website), it may interfere with certain exercises. If your solution isn’t accepted, try disabling the browser add-on and submitting it again.

1. Optional: Learn about your browser’s default stylesheet and CSS resets [in this video](https://www.youtube.com/watch?v=14Vb6tZCjEY) (resets are mentioned starting at the 2:00 mark). This is why there are some spaces that show up in your layout even if you haven’t specified CSS. Real developers almost always use a CSS reset to blow away the default stylesheet and let them work from scratch. If you’re still curious, here’s a [longer video](https://www.youtube.com/watch?v=HqRFPLP7Ffs) on resets.

### [Additional Resources](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#additional-resources)

This section contains helpful links to other content. It isn’t required, so consider it supplemental for if you need to dive deeper into something.

* A good page to bookmark for later reference is [Centering in CSS: A Complete Guide](https://css-tricks.com/centering-css-complete-guide/) from CSS Tricks.
* [W3 Schools’ CSS tutorial](https://www.w3schools.com/css/)
* [OverAPI’s CSS Cheat Sheet](https://overapi.com/css) and [OverAPI’s HTML Cheat Sheet](https://overapi.com/html)
* [CSSreference.io](https://cssreference.io/) and [HTMLreference.io](https://htmlreference.io)
* [Learn CSS Layout](https://learnlayout.com/)
* [CSS Flexbox](https://flexbox.io/) and [CSS Grid](https://cssgrid.io/)
  + The [Complete Guide to Flexbox](https://css-tricks.com/snippets/css/a-guide-to-flexbox/) and the [Complete Guide to Grid](https://css-tricks.com/snippets/css/complete-guide-grid/) from CSS Tricks are both excellent references.
  + These [CSS FlexBox](https://youtu.be/FTlczfR82mQ) and [CSS Grid](https://youtu.be/EFafSYg-PkI) videos by [Dev Ed](https://www.youtube.com/channel/UClb90NQQcskPUGDIXsQEz5Q) might be helpful for you as well!
  + Hop on over to [Flexbox Froggy](https://flexboxfroggy.com/) or dig into [Grid Garden](https://cssgridgarden.com/) to learn some concepts with a fun game!

### [Knowledge Check](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-and-css-basics#knowledge-check)

This section contains questions for you to check your understanding of this lesson. If you’re having trouble answering the questions below on your own, review the material above to find the answer.

* What is the difference between HTML and CSS?
* For accessibility in HTML, what is the attribute used to describe an image (on screen readers or if it fails to load)?
* What is the difference between CSS Grid and Flexbox?
* For a responsive website, should it be designed mobile-first or desktop-first?
* Describe the components of the CSS Box Model.
* In CSS, what is a breakpoint?
* What is a div and how are they used?
* What are the two main groups of CSS properties that control typography style?
* What is the “query string” in a URL and what does it do?
* What is the difference between “pixels” and “em”?
* How does inheritance work for CSS styles, i.e. how does an element get its “default” styles?
* What are two CSS attributes you can change to push an element around on the page?
* What are the three different ways to include a CSS stylesheet in your project or use CSS to style a particular element?
* What is the “default stylesheet” or “user agent stylesheet”?
* What is the purpose of a CSS reset file?
* What is the !important rule in CSS?

# [Foundations](https://www.theodinproject.com/paths/foundations/courses/foundations)

## Developer Tools

[Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#introduction)

[Learning Outcomes](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#learning-outcomes)

[Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#assignment)

[Knowledge Check](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#knowledge-check)

### [Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#null)

All of the most popular modern web browsers include a suite of tools designed to help with the development of websites. They are sometimes referred to as the web inspector. These tools allow you to inspect the HTML, CSS, JavaScript and much more on websites you visit and build.

### [Why do you need to know how to use developer tools?](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#null)

Knowing how to use your browser’s developer tools is an indispensable skill for a web developer. They allow you to easily and quickly track down the cause of issues and bugs, like problems with layout or other CSS hiccups. What might have taken an hour or even a few hours to track down and fix without the use of these tools often takes a matter of minutes or even seconds with the help of developer tools.

As you will come to see in the assignments, much, much more can be done with developer tools other than debugging CSS issues. They provide you with an incredibly powerful assortment of useful tools that will save you a lot of frustration and time as you develop your own websites.

Although they offer you so many useful capabilities, learning how to use the developer tools isn’t that difficult. It can feel clunky the first few times you use them, but with a little practice, you will soon find yourself comfortably using them to experiment and fix bugs with ease on your web pages.

### [Learning Outcomes](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#null)

Make sure you can do each of the following once you have finished the assignments:

* You can open the developer tools in your browser.
* You can select a specific HTML element on your page with your browser’s developer tools.
* You can use developer tools to experiment with the CSS on specific elements of a web page.

### [Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#null)

1. Watch [this 10-minute video](https://www.youtube.com/watch?v=wcFnnxfA70g) that goes over the most useful features of the Chrome DevTools in some detail.
2. Watch this [awesome tutorial](https://www.youtube.com/watch?v=Z3HGJsNLQ1E) from LearnCode.academy on how to use developer tools effectively when working with your CSS. It talks a bit about Bootstrap, which you may not know about yet. Don’t worry about it at this point; just check out the stuff he’s doing to CSS in the DevTools. In particular, editing CSS in the browser in real time is a serious productivity booster compared to using your text editor and continuously refreshing to see the changes.
3. Skim through the [official Chrome DevTools docs](https://developers.google.com/web/tools/chrome-devtools/) [(or Firefox’s!)](https://developer.mozilla.org/en-US/docs/Tools). Chrome and Firefox are constantly enhancing and updating their developer tools. The basic functionality is going to be the same of course, but it’s not unusual for things to move around or change appearance as the tools evolve. These docs should be your up-to-date resource. If your version of DevTools doesn’t look like the videos above, reference these docs to find out where everything went.

### [Knowledge Check](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/developer-tools#null)

This section contains questions for you to check your understanding of this lesson. If you’re having trouble answering the questions below on your own, review the material above to find the answer.

* How do you open developer tools in your browser?
* How do you select a specific element on your page with your browser’s developer tools?
* How do you change CSS in real time on specific elements of a web page with your browser’s developer tools?
* What does a strikethrough in a CSS element mean in your browser’s developer tools?
* How do you check every inherited style for an element in your browser’s developer tools?
* How do you edit HTML in real time in your browser’s developer tools?
* How do you toggle responsive design mode in your browser’s developer tools?
* Are changes made in your browser’s developer tools permanent?
* How do you open the console in your browser’s developer tools?
* What is the best resource for learning about your browser’s developer tools?

# [Foundations](https://www.theodinproject.com/paths/foundations/courses/foundations)

## Project: Google Homepage

[Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#introduction)

[Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#assignment)

[Additional Resources](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#additional-resources)

### [Introduction](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#introduction)

For this mini-project, you’ll deconstruct an existing web page and rebuild it. Don’t worry if the links don’t go anywhere and the search box doesn’t do anything when you submit it. The goal is to start thinking about how elements get placed on the page and roughly how they get styled and aligned. For some of you, this may be the first time you’ve actually tried to “build” something in HTML (very exciting!).

Using the browser’s developer tools (right-clicking something on the page and clicking “inspect element” will get you there) will be your best friend. Build the page in a .html text file and open it in your browser to check it out.

### [Try These Before Starting](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#try-these-before-starting)

These skills will be helpful for you when you start building. Either try them yourself or at least make sure you know how to do it:

1. Two ways to move a div around on the page
2. Stick a div onto the bottom or top of the page
3. Identify the background color of an existing webpage
4. Grab the URL for an image from an existing webpage
5. Center an element horizontally
6. Identify three ways you can include your CSS styles in a page
7. Understand how to use classes and ids to target CSS at specific elements on the page
8. Build a very basic form (even if it doesn’t “go” anywhere)

### [Setting Up Your Project’s GitHub Repository (optional)](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#setting-up-your-projects-github-repository-optional)

As mentioned in the [introduction to Git](https://www.theodinproject.com/courses/foundations/lessons/introduction-to-git), you’ll want to organize all your projects like a portfolio and link them to GitHub so it can be seen by others.

If you do not know how to set up a repository, follow the instructions found in [Git Basics](https://www.theodinproject.com/courses/foundations/lessons/git-basics) to learn how.

1. If you haven’t already, create a folder on your computer called the\_odin\_project and cd into it. This folder will house all the projects you do at Odin.
2. Create a new repo for this project on GitHub.com and call it google-homepage (instead of git-test).
3. Then move that repository onto your local machine. The command should look like: git clone git@github.com:username/google-homepage.git (use SSH).
   * [Setting up SSH on GitHub](https://www.theodinproject.com/courses/foundations/lessons/setting-up-git#step-2-configure-git-and-github).
4. Now cd into the google-homepage project directory that is now on your local machine; set up your README.md file and write a brief introduction for what the current project is and what skills you have demonstrated once you have completed it. (You can do this as a self-reflection at the end of the project which is a good way to review what you have learned.)

If you are having trouble:

* Ensure you followed the steps [here on Step 2.3](https://www.theodinproject.com/courses/foundations/lessons/setting-up-git#step-2-configure-git-and-github) to clone from GitHub with SSH.
* Refer to the [workflow](https://www.theodinproject.com/courses/foundations/lessons/git-basics) in the Git Basics Lesson.

Note: All Git commands need to be run from inside your project’s folder (did you forget to cd into the google-homepage folder?)

When you’re building your project, you will probably end up doing several *git add* + *git commit* cycles before being ready to push it up to GitHub with *git push origin main*.

When writing code, it’s considered best practice to commit early and often. Commit every time you have a meaningful change in the code. This will create a timeline of your progress and show that your finished code didn’t appear out of nowhere.

If you have entered git push origin main and typed out your username and password, if you refresh your GitHub page, you should see new files added onto GitHub.

Okay, that’s enough Git for the moment – time to actually build stuff!

### [Warning About Looking at Other People’s Finished Project Code Before You have Completed Your Own](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#warning-about-looking-at-other-peoples-finished-project-code-before-you-have-completed-your-own)

All of our projects have a “Student Solutions” section at the bottom for people to post their completed code and live websites for the projects. These are here for you to compare your **completed** code to and see what other routes people have chosen to take with their project code and designs.

We know it may be tempting to look at the solutions when you get stuck or don’t know where to start, but **don’t** do it! Here are some reasons why:

* All of the submitted projects are other student solutions. This means they are written by other people who are also still learning and may not necessarily be representative of “best practice” or quality code.
* Looking at solutions robs you of the chance to develop your problem solving and researching skills. In the real world, there will not always be “solutions” for you to look at and you will have to come up with your own.
* There are many different ways to achieve the same end result, it is not “one size fits all”.
* An Analogy: “Looking at completed code is like learning how to make pizza dough from looking at a baked pizza.” There is a lot more that goes into it than you would have assumed by only seeing the final product and not the process that lead to it.

Once you have completed your project feel free to look at the other solutions to see alternative ways the project was completed.

Note: It is okay to check the “Live Preview” to get an idea of layout and styling before starting but not the code files.

**DO NOT PEEK. Come ask for help in our discord server!**

### [Assignment](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#assignment)

#### Easy Version: Build the [Google.com](http://www.google.com) homepage

(the simple one with just a search box).

Inside your project folder, create your index.html file.

1. Tips:
   * DON’T BE A PERFECTIONIST! You’re just trying to make it look like google.com, not actually function like it and it doesn’t have to be spaced exactly the same way to the pixel. Any dropdown menus or form submissions or hover-highlighting should be ignored.
   * USE GOOGLE! You’ll probably run into roadblocks where you can’t figure out how to do something so do what all good devs do… Google it!
   * Now is a good time to [set up the live server extension](https://youtu.be/mGORIVStWWc) in VSCode, if you haven’t already. This extension will save you from the hassle of refreshing the browser window repeatedly to see changes made in your code and will automatically load the most recent change in the browser window.
   * If you’re frustrated with trying to get buttons or inputs to style the way you want (for instance, they seem to just not respond to any styles), look into the CSS property -webkit-appearance: none; or -moz-appearance if you’re using Firefox.
   * [Use this image in place of the actual Google logo to avoid getting flagged as malicious](https://cdn.statically.io/gh/TheOdinProject/curriculum/284f0cdc998be7e4751e29e8458323ad5d320303/foundations/the_front_end/project_html_css/imgs/00.png)
   * [Here’s a cached page of the Google home page for reference in case the original logo is not there.](https://web.archive.org/web/20191130234759/https:/www.google.com/)
   * [Here’s a link to the Google Material Icons page. This is a great resource for free to use open source icons.](https://fonts.google.com/icons)
2. Start with just putting the main elements on the page (the logo image and search form), then get them placed horizontally. You can either download the Google logo or link directly to its URL on the web in your <img> tag.
3. Next do the navbar across the top, first building the content and then trying to position it. Check out [how to build a horizontal CSS navbar](http://www.w3schools.com/css/css_navbar.asp) if you’re lost.
4. Finally, put in the footer, which should be very similar to the top navbar.
5. In general, do as much on your own as you can before relying on the developer tools (or viewing the page’s source code) to help you along.
6. Push your project to GitHub using the instructions above! Then consider sharing your project with the community by submitting a link to your repository in the Solutions section below.

#### Difficult Version (optional): Build the [Google.com search results page](https://www.google.com/search?q=build+this+webpage)

You should be able to reuse much of your code from before if you started with that project. Again, don’t worry about links to nowhere and forms that won’t submit and hard coding the search results (which you’ll have to do of course), just focus on placement and order of items on the page.

Note: All the classes and id’s and names of elements that you inspect on Google’s home page are nonsensical strings (like <div class='srg'>). This is because the code was **Minified** ([see the Wikipedia entry here](http://en.wikipedia.org/wiki/Minification_(programming))), which removes or shortens unnecessary characters and names to help the page load faster. The HTML (or JavaScript or CSS) file will be smaller but the browser can still read it just fine.

### [Viewing your project on the web](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#viewing-your-project-on-the-web)

If you want to show your work (the project) to others, or submit a solution below, you will need to publish your site so that others can access it from the web, rather than just on your local machine. The good news is that if you have your project on GitHub (as described above) doing this is incredibly simple.

GitHub allows you to publish web projects directly from a GitHub repository. Doing this will allow you to access your project from your-github-username.github.io/your-github-repo-name

There are a couple of ways to go about doing this, but the simplest is this:

* make sure that the main HTML file of your project is called index.html. If it is not, you will need to rename it.
* go to your GitHub repo on the web and click the **Settings** button
* click on **Pages** on the left side bar
* change the **Source** from none to main branch and click Save.
* it may take a few minutes (the GitHub website says up to 10) but your project should be accessible over the web from your-github-username.github.io/your-github-repo-name (obviously substituting your own details in the link)

### [Additional Resources](https://www.theodinproject.com/paths/foundations/courses/foundations/lessons/html-css#additional-resources)

This section contains helpful links to other content. It isn’t required, so consider it supplemental for if you need to dive deeper into something.

If you still feel shaky on your understanding of HTML and CSS, that’s okay! You don’t need to be an expert by any means yet. These resources should help if you want to shore up your understanding of things:

* If you want to see the art of CSS, check out the [Style Stage](https://stylestage.dev/), which collects submissions that use identical HTML and modify only the CSS to create wildly different (and beautiful) sites.
* Read through [Shay Howe’s HTML&CSS Tutorial](http://learn.shayhowe.com/html-css/terminology-syntax-intro). Lesson 1 gives a solid overview and you can do the whole thing for a more in-depth understanding.
* Learn more about GitHub using [this tutorial](https://try.github.io) or read more in [this article](http://readwrite.com/2013/09/30/understanding-github-a-journey-for-beginners-part-1).