## **PROJECT: GOOGLE HOMEPAGE**

### **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**

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**

As mentioned in the introduction to Git lesson, 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 Project: 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.
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 to clone from GitHub with SSH.
* Refer to the cheat sheet in the Git Basics Lesson.
* Review the workflow in the Git Basics Project.

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**

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:

* Other solutions might be created by other students or people learning. 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 leads 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 on our discord server!**

### **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.
   * [Here’s a link to the Google Logo just in case they temporarily replace it with a Doodle!](https://www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.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**

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 the right sidebar on Pages
* 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**

This section contains helpful links to other content. It isn’t required, so consider it supplemental 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 Page](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).

### **Examples**

* <https://michalosman.github.io/google-homepage/>
* <https://nuketurtle.github.io/google-homepage/>
* <https://tarekvisch.github.io/google-homepage/>
* <https://catqueencodes.github.io/google-homepage2/>