

Web Development Course

Project 4: News Site



Project: <https://news-website-ooo.glitch.me/>

Code: <https://glitch.com/edit/#!/news-website-ooo>

Big Idea: Learn how to customize the layout of a website!

Time: 35-45 minutes

- | | |
|----------------------|---|
| 5 minutes | Introduce project, More advanced CSS, layouts |
| 5-10 minutes | Design a layout |
| 20-25 minutes | News Sites project |
| 5 minutes | Sharing and Discussion |

Background

CSS and Layouts

CSS is the language we use to style an HTML document and it describes how HTML elements should be displayed.

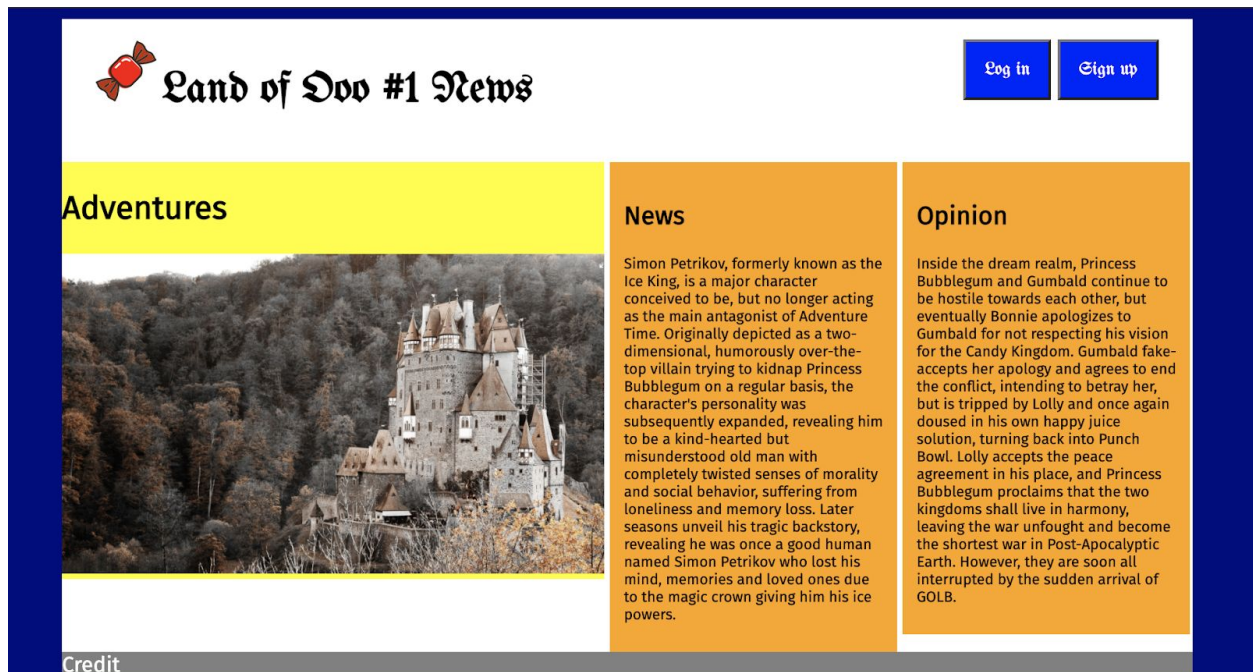
This project will focus on using CSS to edit the layout of the site, for example centering things, adding columns, and placing elements like images and buttons around the site.

It will also review expanding font options with Google Fonts.

Teacher Reference:

[The CSS Display Property](#)

Project Planning



- Site example 1: <https://news-website-ooo.glitch.me/>
- Site example 2: <https://theinknews.glitch.me/>

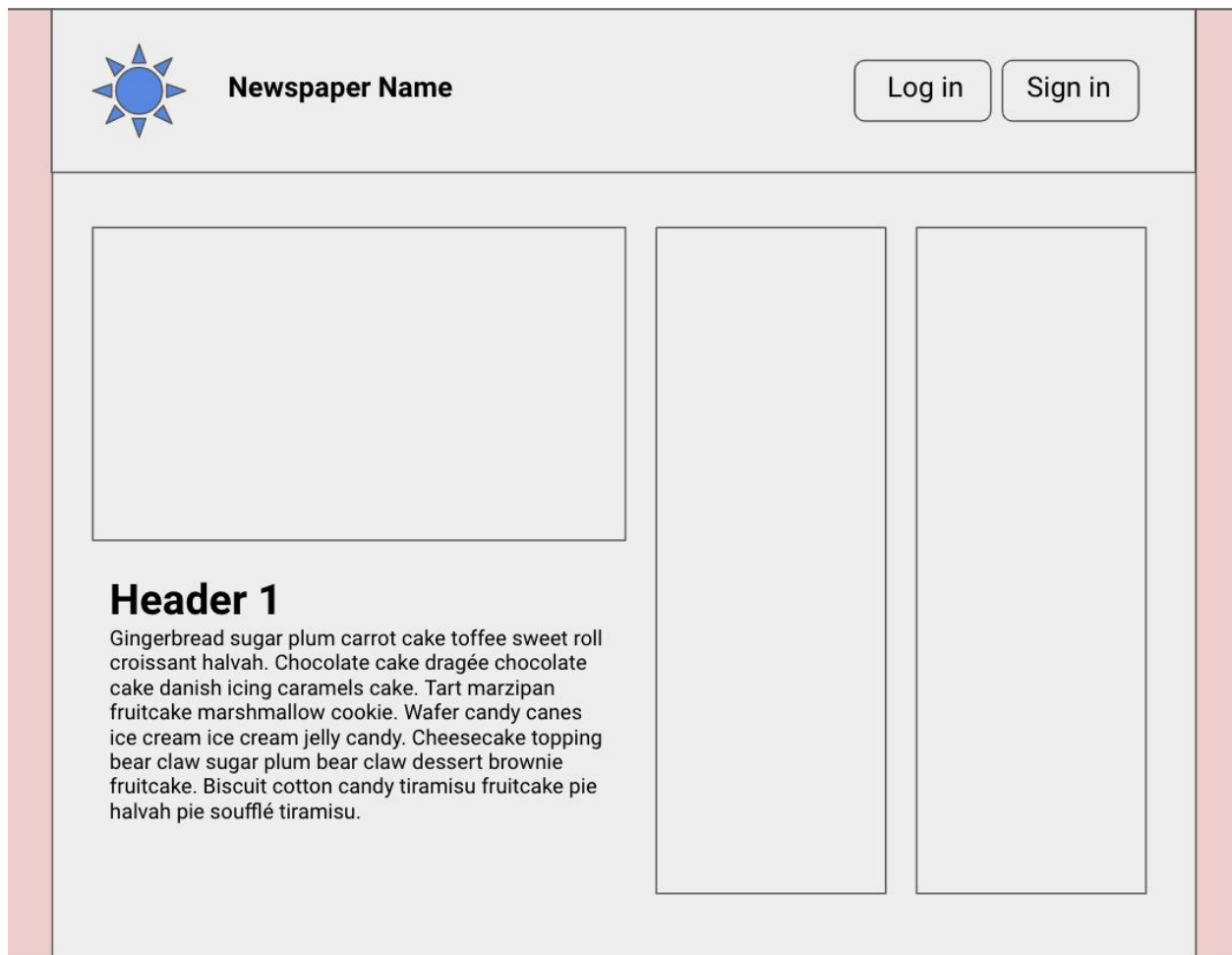
Instructions

1. Come up with a name for your news site and start to design the layout.
2. Include some tabs at the top of your news site to configure your web page's navigation.
3. Make sure to include some images and a variety of text for some fun.

Step 1: Design and Planning

There are a lot of ways to do different things with HTML and CSS. We're introducing one way here, but there are multiple ways to create buttons or layouts in CSS. None are wrong.

To start, students should **plan out** their news sites. Even advanced web developers can have trouble designing directly in HTML and CSS. They can use paper and pencil or Google Slides to plan. Draw a header and the columns. For inspiration, look up news sites with your students and encourage them to do the same.



Step 2: Coding HTML & CSS Columns

Add Columns

Add **divs** to your site that will serve as columns. Add class names, like **half**, **quarter**, or **third** to your columns, depending on how wide you want them to be.

HTML:

```
<div class="half">
  <h1>Title of Section</h1>
</div>
```

CSS:

```
.half {
  background-color: orange;
  display: inline-block;
  width: 48%;
}
```

Let's break down the code above:

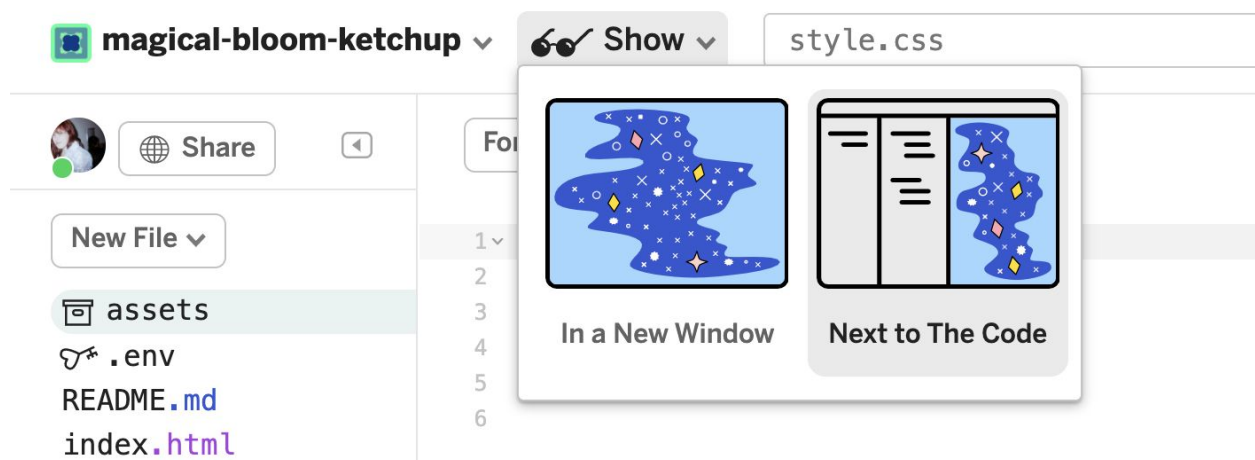
Display

Elements can either be **inline**, **block**, or **inline-block**. Block elements push things onto another line, inline elements line up next to each other. Inline elements are like pictures, or links, they don't push things onto new lines. Blocks are like paragraphs or divs, making a new one starts a new line. Inline-block is special, because elements can exist on the same line, but still have width set to them. Elements you use as columns should be inline-block.

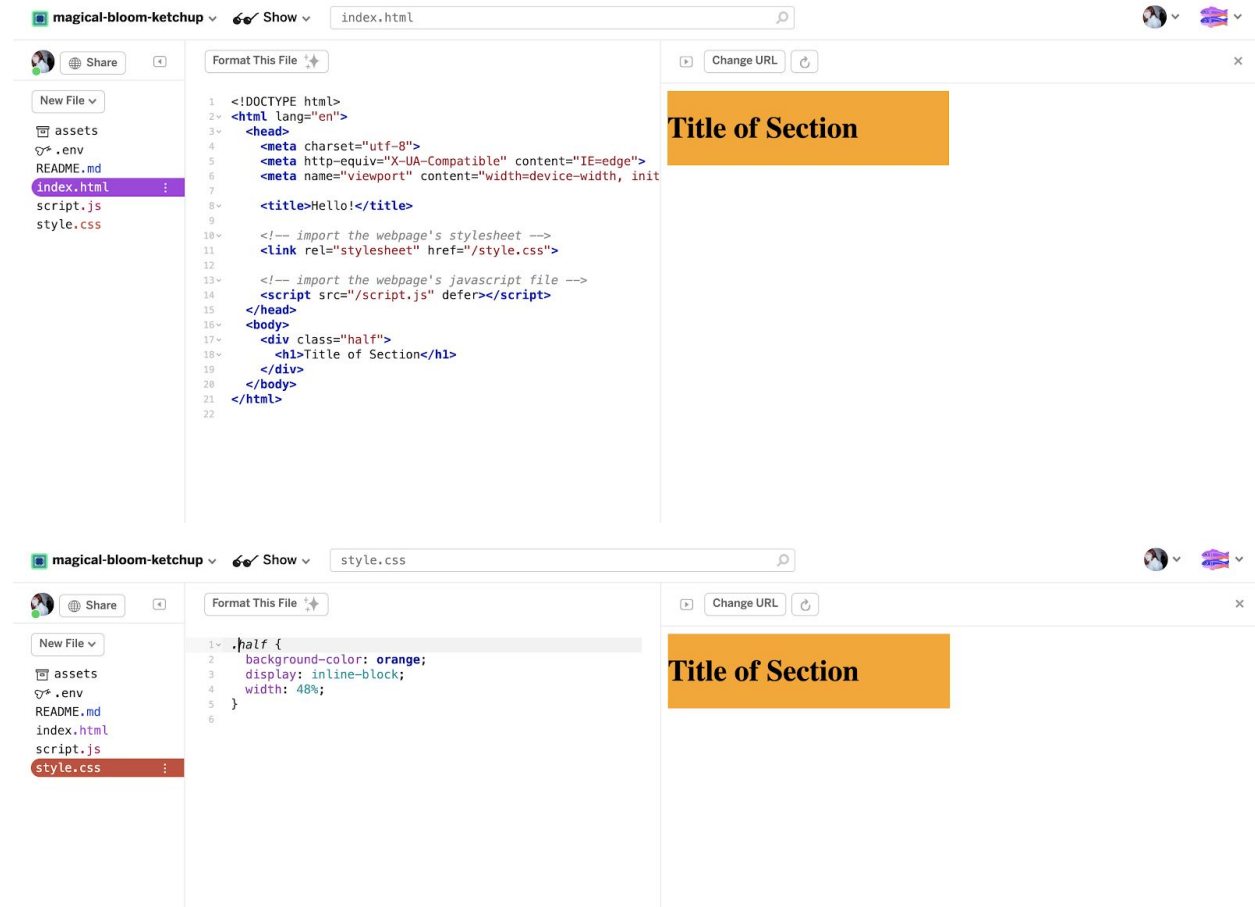
Width

Width sets the div to be a **percentage** of the full page-width.

Play around with widths, and get your columns set up. If they're not showing up, make sure they have either a height set to them, or content, such as images or text, added to them in HTML. **You can view your webpage next to the code to see changes live.**



The screenshot shows a web development environment with a sidebar on the left containing a 'Share' button and a file explorer with items like 'assets', '.env', 'README.md', and 'index.html'. The main area shows a code editor with a 'style.css' file open. A live preview window is overlaid on the code, showing two columns. The left column is titled 'In a New Window' and the right column is titled 'Next to The Code'. The right column contains a blue abstract image with white and yellow patterns.



The first screenshot shows the editor with the `index.html` file open. The code includes a `<div class="half">` containing an `<h1>` with the text "Title of Section". The visual output shows an orange rectangular box with the text "Title of Section" inside.

The second screenshot shows the editor with the `style.css` file open. The code defines a `.half` class with the following properties: `background-color: orange;`, `display: inline-block;`, and `width: 48%;`. The visual output shows the same orange box with "Title of Section" text.

Float

The other way to position elements is to add a "float" property to them.

```
<div class="half">
  <h1>Title of Section</h1>
</div>
```

```
half {
  background-color: orange;
  float: left;
  width: 48%;
}
```

Step 3: Customizing Fonts

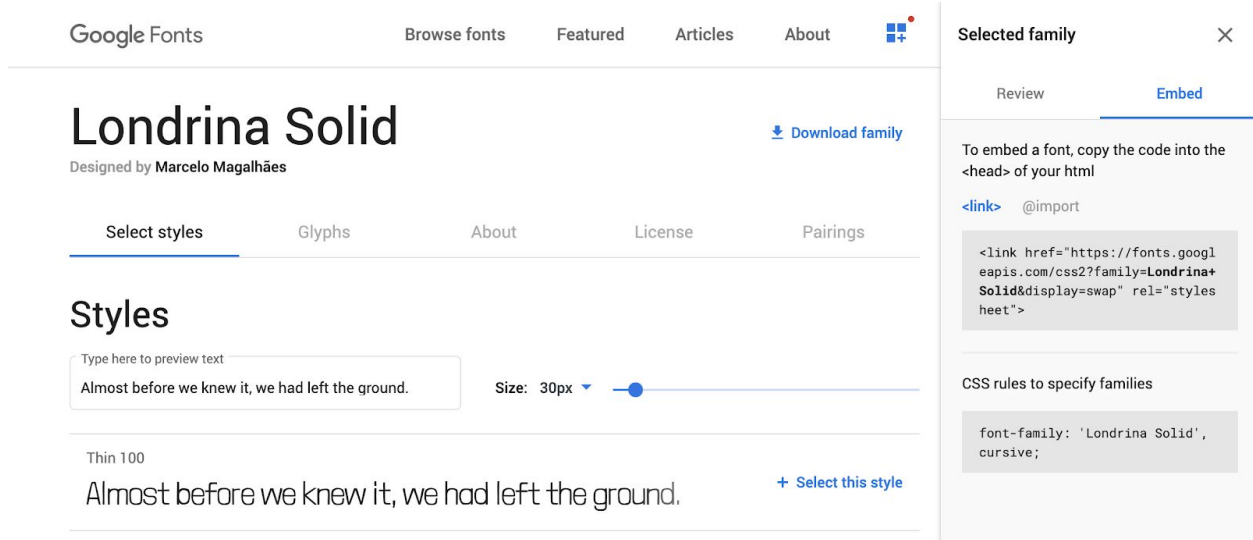
Google Fonts

Ready to expand the fonts available on your site? One way to have an almost unlimited number of fonts to choose from is to use Google Fonts!

To start, visit fonts.google.com and search for a font you like. You can browse by categories, or search by name.

When you've found a font that you like and want to add to your website, click into it. Press "+Select this Style" on the font-weight you want to use.

Once you've clicked that, a menu will open up on the right. Switch to the "Embed" tab. These are special codes that you're going to use to **link** this font to your website.



The screenshot shows the Google Fonts interface for the 'Londrina Solid' font. The main area displays the font name, designer (Marcelo Magalhães), and various tabs like 'Select styles', 'Glyphs', 'About', 'License', and 'Pairings'. Under 'Select styles', a preview of the font is shown with the text 'Almost before we knew it, we had left the ground.' and a size of 30px. A '+ Select this style' button is visible. On the right sidebar, the 'Embed' tab is active, showing instructions to copy code into the <head> of an HTML document. Two code blocks are provided: one for linking the font and another for CSS rules to specify the font family.

There are two parts to this (see the two boxes of code). The first is the **link** to add to the **head** of your index.html file. Copy it, and post it **inside the <head> and </head> elements on your index.html page**. Your code might look like:

```
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <title>Land of Ooo News</title>

  <!-- import the webpage's stylesheet -->
  <link rel="stylesheet" href="/style.css">

  <!-- import the Fire Sans font from Google Fonts -->
  <link href="https://fonts.googleapis.com/css2?family=Fira+Sans&display=swap"
rel="stylesheet">

  <script src="/script.js" defer></script>
</head>
```

That's great, the font is connected! You still have to tell the site where to use it. Use CSS for this.

Copy the second box, CSS rules to specify families, and attach it to an element in your CSS. For example,

```
h1 {  
    /*copy and paste the line from the Google fonts site here*/  
    font-family: 'Londria-Solid', cursive;  
}
```

Use **Google Fonts** to customize your news site. You can use multiple fonts by adding them the same way you added the first one. Mix and match 3-4 different fonts to make your news site unique!

Step 4: Adding Buttons



There's a button element in HTML. You can type:

```
<button>Click here!</button>
```

This has some pretty cool default style, but you can add to it with:

```
button {  
    cursor: pointer;  
    font-size: 16;  
    background-color: blue;  
    color: white;  
    font-family: 'UnifrakturMaguntia', cursive;  
    padding: 14px;  
}
```

Notice how there's no . before **button** above. That means it will affect **all** the buttons on your webpage, regardless of classes you set.

Step 5: Hover States

Add hover states by adding a **hover** condition to the elements. For example, if you wanted a link to change color from red to green on hover, you could write in your CSS document:

```
a {  
    color: red;  
}  
  
a: hover {  
    color: green;  
}
```

To test, view your webpage and **hover** over the element you've chosen with your cursor.

Discussion Points

- More advanced CSS layouts
Instead of the basic website layout that was introduced in the last project,, you will now be able to learn how to customize the layout of your website.
- Designing a layout
What were you thinking about when designing your site initially? What were some of the ways you had to change it when it came time to code it? What would you do differently next time?
The purpose of any page one may design is for it to communicate information in the most effective and clear manner. The layout of a page usually involves lots of rearranging and formatting of elements, but there are several tips to ensure a balanced layout.
 - Use a grid
 - Choose a single focal point
 - Use the Rule of Thirds
 - Use white space
 - Repeat design elements
 - Use hierarchy
 - Emphasize and scale
- How to add style with classes
One way to style an element is by creating a class in CSS and adding it: `<div class="...">`. What are some of the ways you decide what to call class names?