Schedule

- Review Core 1 Skill Assessment
- Project 1 Proposals
- HTML & CSS Basics
- Typography on the Web

Homework

- Assignment 1: Set up GitHub Page
- In-class exercise: Upload Project 1 proposal draft by the end of class
- Optional video (password: interaction)
  - Demo: HTML/CSS Basics
  - Demo: Typography on the Web

Notes from Student Work

- Sound effects - MDN Audio in HTML
- Scolling based on image movements

Review Core 1 Skills Assessment Test

Here are some common trends and interests:

1. Different sections in Core 1 are using different tools to code.

2. Everyone have basic understanding of HTML and CSS but little experience in JavaScript.

3. Interactive web design, game making, typography are some commonly shared interests.

4. Confusions in languages, software etc.

- HTML, CSS, JavaScript, Python are **programming languages**.

- Visual Studio Code, Sublime, Brackets, Vim are **text editors**.

- Glitch is a web application for people to build simple web applications collaboratively using JavaScript

- Processing is a software while p5.js is a JavaScript Library.

- **Java and JavaScript are 2 completely different languages.** Read more about the differences

GitHub

- GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.
- GitHub Pages is a static site hosting service that takes HTML, CSS and JavaScript files straight from a repository on GitHub, optionally runs the files through a build process, and publishes a website.
- GitHub is also a great platform to learn from the open-source community.

Relationship between HTML, CSS, JavaScript

- HTML - Hypertext Markup Language, to describe contents and layout a rough structure of the webpage
- CSS - Cascading Style Sheets, a style sheet language to shape and style website content in terms of colors, typography, spacing, borders, and more.
- JavaScript - a dynamic programming lanaguage that makes the website interactive

HTML Basics

Recap - What we had last week

```
<!DOCTYPE html>
<html>
<head>
  <title>Week 1 Demo</title>
  <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
  <h1>This is a Heading</h1>
  <p>This is a paragraph.</p>
  <img src="images/img.jpg">
</body>
</html>
```

- `<!DOCTYPE html>` declaration tells the web browser which version of HTML is being used
- `<html>` element is the container for all the other elements on the page.
- `<head>` element contains meta information about the document, such as the title, which is displayed in the browser's title bar or tab.
- `<body>` element contains the visible content of the web page.
- `<h1>` and `<p>` elements are used to create headings and paragraphs, respectively.
- `<img>` element is used to embed images. The src attribute specifies the URL of the image file, and the alt attribute provides alternative text for the image, which is displayed if the image cannot be loaded.

Opening and Closing Tags

- Tags are enclosed in angle brackets <> and always comes in pairs `<p></p>`
- Closing tags always ends with a `/`

Document Structure

Opening Tag	Closing Tag	Description

Container Tags

Opening Tag	Closing Tag	Description

HTML5 Sematic Tags

Opening Tag	Closing Tag	Description

Non-containter Tags

Tag	Description
`<img>`	Insert an image
`<br />`	Line break

Tables

Opening Tag	Closing Tag	Sample Attributes	Description

Nesting

- Nesting in HTML referes to the precess of placing one HTML element inside another. It allows you to create more complex and structured layout for the web pages.
- `<div>` stands for a division or a section, it's a container for a block of codings with different opening and closing tags

CSS Basics

Recap - what we had last week

```
body {
    background-color: #f0f0f0;
    font-family: "Gill Sans", sans-serif;
    margin: 0;
    padding: 0;
}

h1 {
    color: #333;
    font-size: 2em;
    margin: 0;
    padding: 0.5em;
    text-align: center;
}

p {
    color: #555;
    font-size: 1.2em;
    line-height: 1.5em;
    margin: 1em 0;
    padding: 0;
}

img {
    max-width: 100%;
}
```

CSS History

- CSS is the standard language/format for styling web pages, which specifies what the page's HTML will look like in the browser.
- It was proposed by Håkon Wium Lie in 1994, a researcher at CERN, as a way to separate the presentation of a document from its structure and content.
- CSS allowed designers and developers to create style sheets to be reused across multiple pages, making it easier to maintain and update the appearance of a website.

Background Properties

background property is used to set the background color or image of an HTML element. There are several properties that you can use to customize the background of an element, including:

1. background-color: Specifies the background color of an element. You can use any valid CSS color value, such as a color name, hex code, or RGB value.

   ```
   body {
       // background-color: brown;
       // background-color: #FFC500;
       background-color: rgb(100, 200, 250);        // ranged from 0 to 
   255
   }
   ```

   HTML Color Names

2. background-image: Specifies the background image of an element. You can use a URL to reference an image file.

   ```
   body {
       background-image: url("image.jpg");
   }
   ```

3. background-repeat: Specifies if and how a background image should repeat. Possible values are "repeat", "repeat-x", "repeat-y", and "no-repeat".

```
body {
    background-image: url("image.jpg");
    background-repeat: no-repeat;
}
```

4. background-position: Specifies the position of a background image. You can use any valid CSS length value or a keyword value, such as "center", "top", or "bottom".

```
body {
    background-image: url("image.jpg");
    background-position: center;
}
```

5. background-size: Specifies the size of a background image. You can use any valid CSS length value or a keyword value, such as "cover" or "contain".

```
body {
    background-image: url("image.jpg");
    background-size: cover;
}
```

6. background-attachment: Specifies if a background image should be fixed or scroll with the rest of the content. Possible values are "scroll" and "fixed".

```
body {
    background-image: url("image.jpg");
    background-attachment: fixed;
}
```

Border Properties

Borders in CSS are used to add visual separators between elements on a web page. The border property in CSS provides several options to customize the appearance of borders, including width, style, and color.

In this example, a div element will have a 5-pixel-wide white solid border.

```
div {
    border-width: 5px;
    border-style: solid;
    border-color: white;
}
```

- The `border-width` property sets the width of the border. You can specify the width in pixels, ems, rems, or a percentage of the containing element's width.
- The `border-style` property sets the style of the border. Common border styles include `solid`, `dotted`, `dashed`, `double`, `groove`, `ridge`, `inset`, and `outset`.
- The `border-color` property sets the color of the border. You can specify the color using a color name, hexadecimal value, RGB value, or RGBA value.
- In addition to these properties, there are also individual properties for controlling the width, style, and color of each border side: `border-top`, `border-right`, `border-bottom`, and `border-left`. These properties can be used to specify different values for each side of the border, giving you even more control over the appearance of borders in your CSS.

Margin and Padding

- Margin and padding are CSS properties used to control the space around an HTML element.
- The `margin` property specifies the space outside of an element's border. It creates clear space around an element, separating it from other elements on the page. For example, you can use the margin property to add space between two `div` elements so they don't appear to be touching.
- The `padding` property, on the other hand, specifies the space inside an element's border. It creates space between the element's content and its border. For example, you can use padding to add space between an element's text and its border.

List Styling

The CSS `list-style` property is used to style the appearance of ordered (numbered) and unordered (bulleted) lists in HTML. It allows you to control the markers that appear before each list item, as well as the spacing between items and other aspects of the list's appearance.

```css
ul {
    list-style-type: square;
    list-style-position: inside;
    list-style-image: url("img.jpg");

    // list-style: square inside url("img.jpg");
}
```

- `list-style-type`: Specifies the type of marker to use for the list items, such as a disc, circle, square, or decimal.
- `list-style-image`: Specifies an image to use as the marker for the list items. This can be a URL to an image file, or a data URI.
- `list-style-position`: Specifies the position of the marker relative to the list item. The options are inside, which means the marker will appear inside the list item, and outside, which means the marker will appear outside the list item.

Nesting

- Nesting in HTML refers to the precess of placing one HTML element inside another. It allows you to create more complex and structured layout for the web pages.

- `<div>` stands for a division or a section, it's a container for a block of codings with different opening and closing tags

```
<div>
    <p>This is a paragraph.</p>
    <!-- p is a child of the nesting element -->
</div>
<!-- div is a parent element -->
```

Class and ID

In HTML and CSS, classes and IDs are used to select elements and apply styles to them. The main difference between the two is their scope and the number of times they can be used on a single page.

Class:

- Classes are defined using the `.` symbol in CSS and are applied to elements using the `class` attribute in HTML.
- Classes can be used multiple times on a single page to select and style different elements.
- Classes are ideal for styles that are shared between multiple elements.

```
<div class="header">This is the header</div>
<div class="header">This is another header</div>
```

```
.header {
  background-color: blue;
  color: white;
  padding: 10px;
}
```

ID:

- IDs are defined using the `#` symbol in CSS and are applied to elements using the `id` attribute in HTML.
- IDs can only be used once on a single page.
- IDs are ideal for styles that are unique to a single element.

```
<div id="header">This is the header</div>
```

```css
#header {
  background-color: blue;
  color: white;
  padding: 10px;
}
```

Typography

Web Fonts

Web Fonts are fonts that are specifically designed for the web. They are hosted on a server and can be downloaded and used on a website.

To use a web font in CSS, you'll need to first include a reference to the font file in the HTML, and then specify the font in the CSS file using the font-family property.

There are 4 common web font formats:

1. WOFF (Web Open Font Format): WOFF is the most widely supported web font format, it provides good compression and supports font features such as OpenType features, kerning and variable font weights.
2. WOFF2 (Web Open Font Format 2.0): WOFF2 is an improved version of WOFF and provides better compression, faster download times and better security.
3. EOT (Embedded OpenType): EOT is a Microsoft font format that was created for Internet Explorer, it provides good font feature support and is well suited for use with Internet Explorer.
4. TTF/OTF (TrueType/OpenType Font): TTF and OTF are font formats that are supported by most desktop applications, they are not as widely supported by web browsers as WOFF or WOFF2, but they can be used as web fonts by embedding them in web pages.

Customized Font

Customized fonts, on the other hand, are fonts that you create yourself or that have been created by others and then customized for your specific needs. Custom fonts can be created using tools like Adobe Illustrator or FontLab, and then converted into a web-ready format like WOFF or TTF. To use a customized font in CSS, you'll need to include a reference to the font file in the HTML and then specify the font in the CSS file using the @font-face rule.

1. Choose a font: Choose the font that you want to use on your website. You can either use a free font or a paid font.
2. Host the font file: You need to host the font file on a server. You can either host it on your own server or use a third-party service such as Google Fonts.
3. Reference the font in CSS: To reference the font in your CSS, you can use the @font-face rule. The @font-face rule specifies the location of the font file and the font-family name to be used.
4. Apply the font: To apply the font to an element, you can use the font-family property in CSS.

5. Test the font: Test the font in different browsers to ensure that it works as expected.

CSS Properties for Fonts

In CSS, text styling refers to the appearance and formatting of text content in HTML elements. There are several properties that you can use to style text, including:

1. font-family: Specifies the font family to be used for text. You can specify multiple font families, separated by commas, in case the first font is not available.

   ```
   p {
       font-family: Arial, sans-serif;
   }
   ```

2. font-size: Specifies the font size of text. You can use any valid CSS length value, such as "12px" or "1em".

   ```
   p {
       font-size: 16px;
   }
   ```

- Units for font-size in CSS
  - Pixel (px) - Pixels are absolute units and are the same size on all devices.
  - Ems (em): Ems are relative units, and the font size is based on the parent element's font size. 1em is equal to the current font size.
  - Rems (rem): Rems are similar to Ems, but instead of being based on the parent element's font size, they are based on the root (HTML) element's font size.
  - Percentage (%): Percentage is a relative unit that specifies the font size as a percentage of the parent element's font size.
  - Points (pt): Points are absolute units that are commonly used in print design.

3. font-weight: Specifies the boldness of text. You can use keywords such as "bold" or "normal", or numeric values such as "400" or "700".

   ```
   p {
       font-weight: bold;
   }
   ```

4. font-style: Specifies the font style of text. Possible values are "normal" and "italic".

   ```
   p {
       font-style: italic;
   }
   ```

5. color: Specifies the text color. You can use any valid CSS color value.

   ```
   p {
       color: cornsilk;
   }
   ```

6. text-align: Specifies the horizontal alignment of text. Possible values are "left", "center", "right", and "justify".

   ```
   p {
       text-align: center;
   }
   ```

7. text-decoration: Specifies the text decoration. Possible values are "none", "underline", "overline", and "line-through".

   ```
   a {
       text-decoration: none;
   }
   ```

8. text-transform: Specifies the text capitalization. Possible values are "none", "uppercase", "lowercase", and "capitalize".

   ```
   p {
      text-transform: uppercase;
   }
   ```