



CSCI-UA-4-005

# Intro to Web Design + Computer Principles

## Page Layout

Professor Emily Zhao



## Agenda

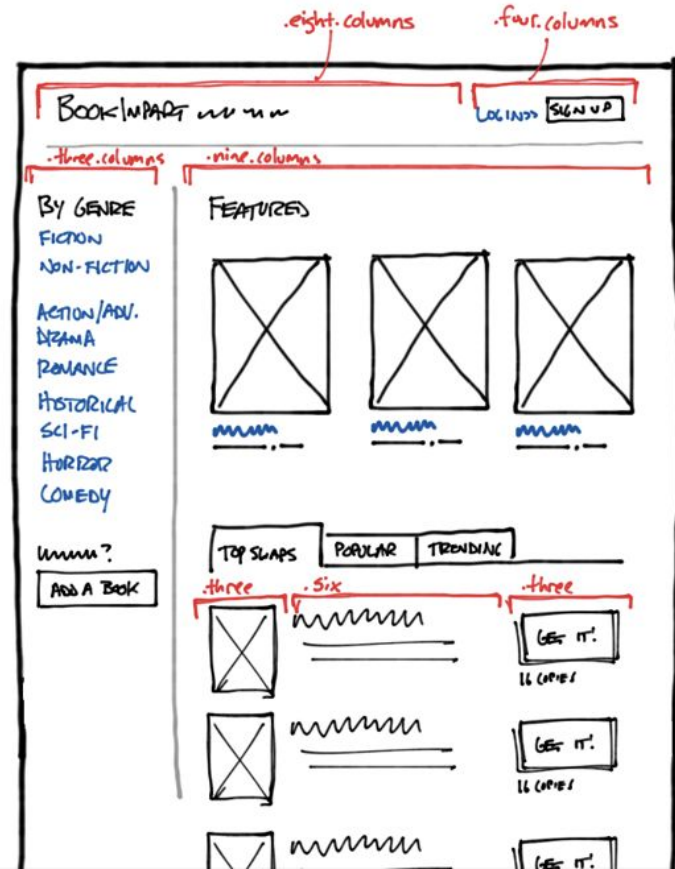
- Wireframing
- CSS Grid Basics
- Using a custom font
- Assignment #6

# Wireframing

# Wireframing

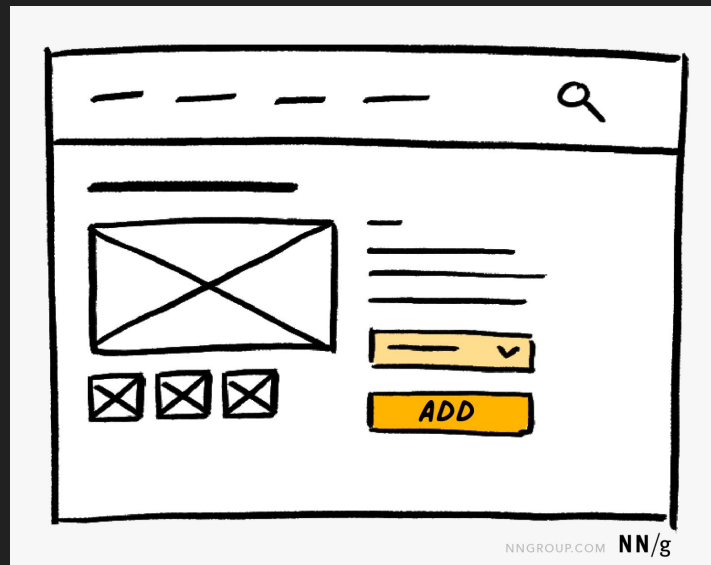
a visual representation or blueprint of a web page, mobile app, or user interface, created during the early stages of the design process

## BOOKIMPACT HOMEPAGE



# Key Principles of Wireframing

- Keep wireframes ***simple and uncluttered***. Use lines and basic shapes to represent elements. Avoid excessive details and overdesigning!
- Define the ***hierarchy of content*** – prioritize and differentiate elements like headings, body text, images, and calls to action
- Always ***design with the user in mind***. Consider the user's needs, goals, preferences, and abilities



# Wireframing Tools

## Non-Digital:

- Pen and paper

## Free Websites:

- Figma
- Adobe XD
- Wireframe.cc
- Canva



## Wireframes





## **Wireframing Exercise: Designing a Login Page**

Wireframe a login page for a social media platform or website. The login page should cater to both new and returning users. Consider what elements, interactions, and design choices you need to incorporate to create a seamless and secure login experience.

## Wireframing Exercise: Designing a Login Page

Possible elements:

- Username/email and password input fields
- Sign-in button
- Links to password recovery or account creation
- Social media login options (if applicable)
- Any additional security features (e.g., CAPTCHA, Two-Factor)
- Error messages and feedback



Look at examples/  
how other people have done it!



## Log In to Your NYU Account

NetID (e.g., aqe123)

ez560

Password

.....

Be cybersecurity aware:  
[Learn how to confirm that  
this is the legitimate NYU  
Login page.](#)

*By logging in, you agree to abide  
by the [Policy on Responsible Use  
of NYU Computers and Data.](#)*

Login

[Reset Password](#) [Forgot NetID](#) [Activate NetID](#) [Accessibility](#)

To securely log out of your NYU account, quit your browser,  
especially when using a shared computer.

**Need help?** Contact the [NYU IT Service Desk](#), open 24x7 for support  
by email or phone.

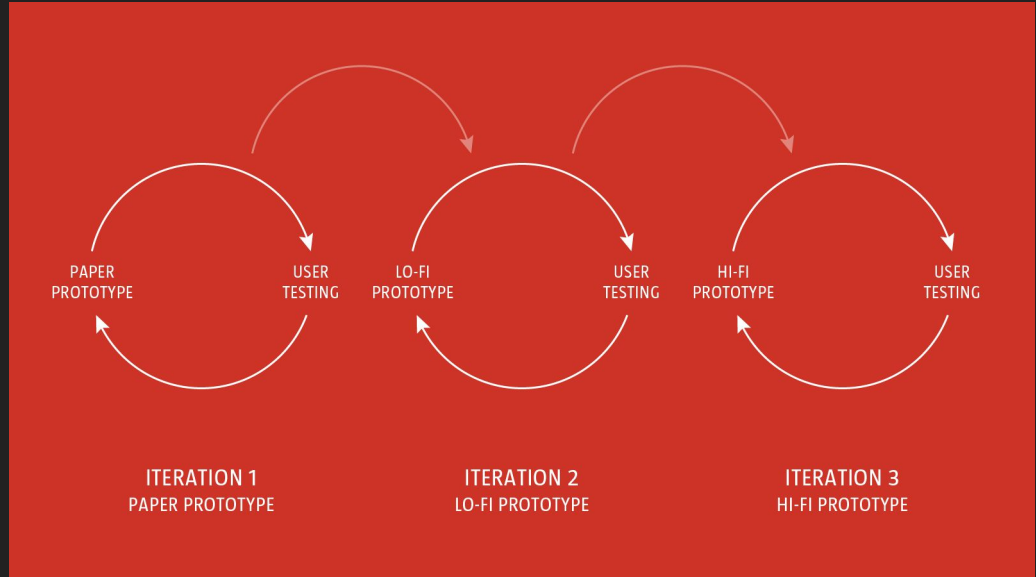
**[Demo] Digitizing your wireframe with Figma**

# Wireframing is an iterative process

Here is an approach to wire-framing that can be adapted to a variety of design projects:

- Think
- Design
- Implement
- Revise

This sequence can be looped through as necessary.



# Page Layout with CSS

## Page Layout with CSS

There are several ways to design the layout of a web page with CSS

- CSS `float` property
- CSS `position` property
- CSS flexible box module (Flexbox)
- CSS grid module

## CSS Float Property

The CSS float property allows you to position block elements inline.

This means that any element, block or inline, can be positioned alongside another element.

The CSS float property is an ***outmoded*** technique of web page layout.

# CSS Positioning

The CSS position property specifies the type of positioning used for an element on a page.

**static**

Default document flow

**absolute**

Element is positioned relative to its first positioned (not static) parent element

**fixed**

Element is positioned relative to the browser window

**relative**

Element is positioned relative to its normal position

**sticky**

Positioned based on the user's scroll position

## CSS FlexBox

Use the CSS Flexible Box Layout Module (Flexbox) for arranging items along **one axis**.

Flexbox consists of flexible containers and flexible items within.

A flex container expands items to fill available free space or shrinks them to prevent overflow.

In practice, flexbox can accommodate different screen sizes and different display devices more easily than the CSS float property.



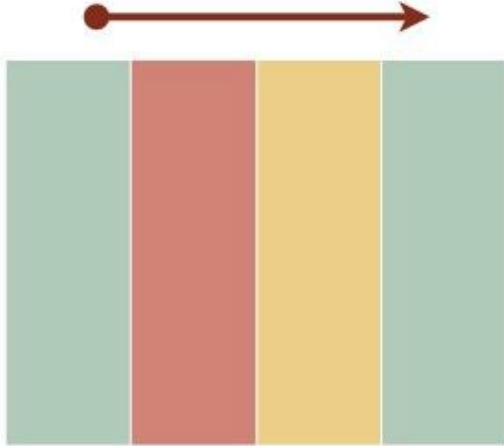
## CSS Grid

Web pages are often laid out using grid systems.

CSS grids are intended to make this process more intuitive by defining a grid and then specifying where content should be placed within it.

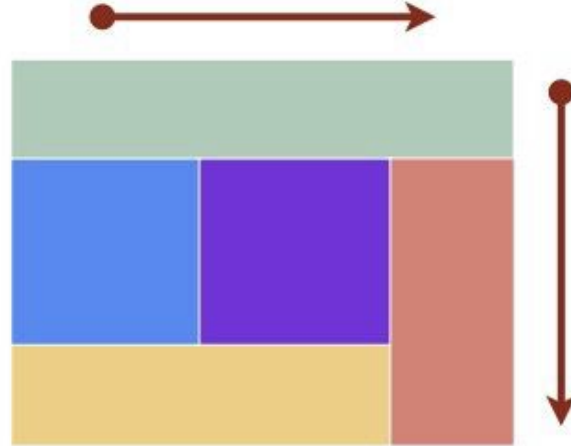
The CSS Grid Layout Module can be used for the **overall structure of a page.**

# FlexBox vs Grid



**Flexbox**  
One Dimensions

VS



**CSS Grids**  
Two Dimensions

# CSS Grid Properties

## **grid-template-columns**

Specifies the number of columns and their widths

```
grid-template-columns: 60px 60px;
```

## **grid-template-rows**

Specifies the number of rows and their heights

```
grid-template-rows: 40px 4em 40px;
```

## **grid-gap**

Set the gap between rows and columns

```
gap: 10%; /* same spacing for both */
```

```
gap: 10px 50px; /* row space col space */
```

## CSS Grid Properties

### `grid-template-areas`

Specifies a unique grid using a series of nicknames for each cell of the layout

```
grid-template-areas:
```

```
    "a a a"
```

```
    "b c c"
```

```
    "b c a";
```

## CSS Grid

Web pages are often laid out using grid systems.

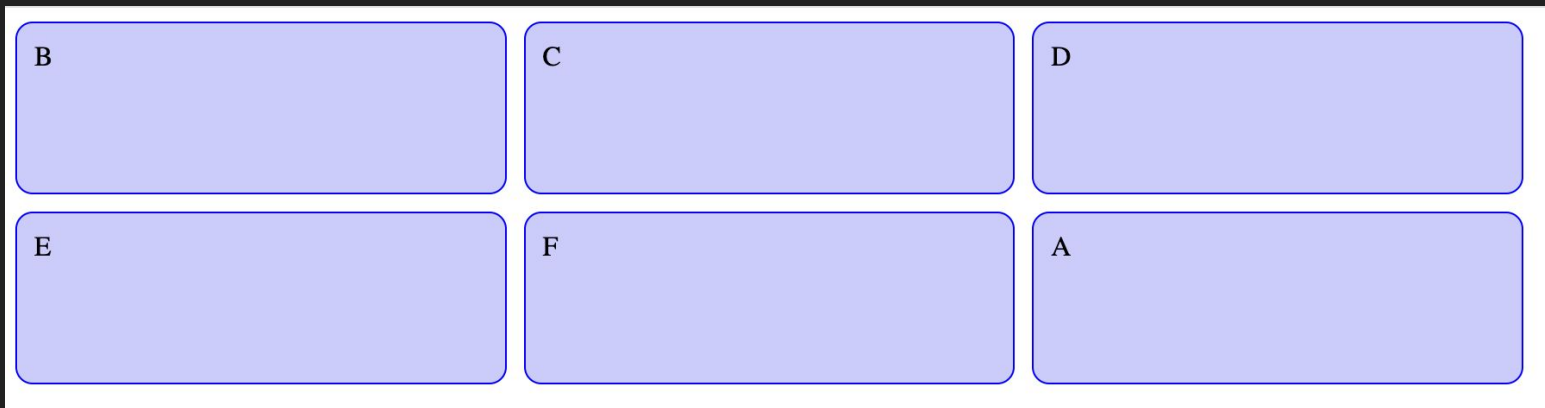
CSS grids are intended to make this process more intuitive by defining a grid and then specifying where content should be placed within it.

The CSS Grid Layout Module can be used for the **overall structure of a page.**

## Grid Exercise #1

Let's build the following layout using CSS grid:

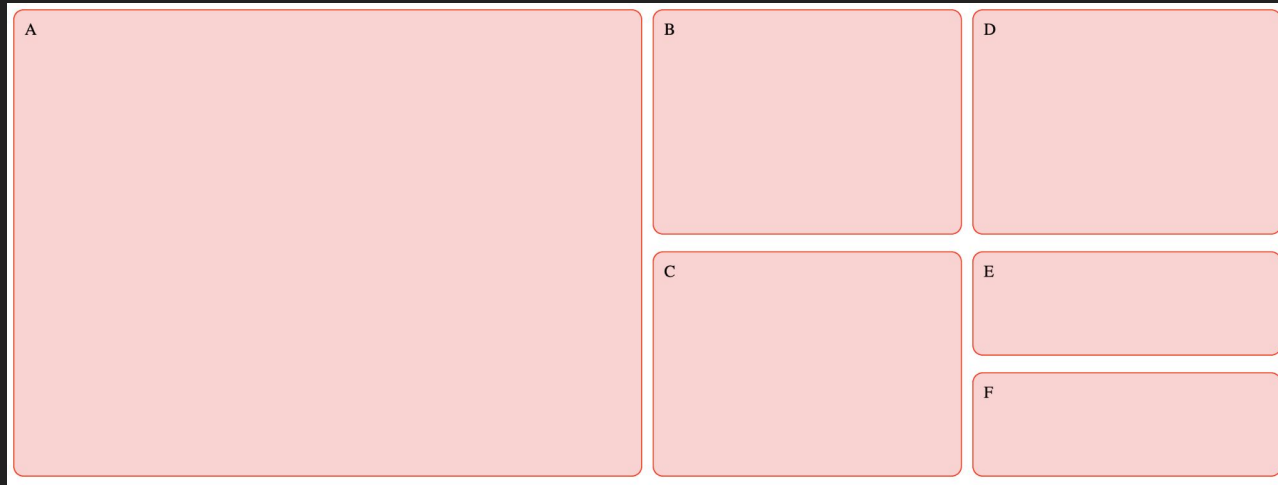
*There are starter files available on the class website.*



## Grid Exercise #2

Try to build this one on your own!

*There are starter files available on the class website.*



# What does CSS Grid look like in practice?

✕ Follow @rachelandrew

## Grid by Example

Everything you need to learn CSS Grid Layout

Start Here

Examples

Patterns

Video tutorial

Resources

### CSS Grid Layout

This site is a collection of examples, video and other information to help you learn CSS Grid Layout. Developed and maintained by [Rachel Andrew](#).

### The Video Tutorial

A collection of short and to the point videos, demonstrating various parts of the CSS Grid Layout specification.

### Get Started Guide

A structured guide to resources that will help you to start learning CSS Grid Layout.

New!

### The Examples

Small examples of the CSS Grid Layout specification. Each demonstrates a feature of the specification. Includes new Subgrid examples!

### Patterns

Grab & Go. A set of example patterns with fallbacks for older browsers.



# Importing your own Fonts

## Importing Your Own Fonts

1. Choose a custom font
  - a. [Google Fonts](#)
  - b. [Adobe Fonts](#)
  - c. [Font Squirrel](#)
  - d. [Dafont](#)
2. Host the font
  - a. Upload the font to your website's server
  - b. Use a third-party service that hosts for you (i.e. Google Fonts)
3. Add CSS rule that references the font
4. Apply the custom font to your site

# Hosting Your Own Font

```
/* CSS Rule that references font */
@font-face {
  font-family: 'CustomFont';
  src: url('path-to-font/customfont.ttf');
  font-weight: normal;
  font-style: normal;
}

/* Applying custom font to site */
h1, h2, h3 {
  font-family: 'CustomFont', sans-serif;
}
```

# Third-Party Font Hosting – Google Fonts

## Quickstart guides

Copy and paste this HTML into a file:

```
<html>
<head>
  <link rel="stylesheet"
        href="https://fonts.googleapis.com/css2?family=Crimson+Pro">
  <style>
    body {
      font-family: 'Crimson Pro', serif;
      font-size: 48px;
    }
  </style>
</head>
<body>
  <div>Making the Web Beautiful!</div>
</body>
</html>
```

Font(s)	Request
Crimson Pro (default)	<code>https://fonts.googleapis.com/css2?family=Crimson+Pro</code>
Crimson Pro Bold	<code>https://fonts.googleapis.com/css2?family=Crimson+Pro:wght@700</code>
Crimson Pro Regular & Bold	<code>https://fonts.googleapis.com/css2?family=Crimson+Pro:wght@400;700</code>
Crimson Pro Bold & Bold Italic	<code>https://fonts.googleapis.com/css2?family=Crimson+Pro:ital,wght@0,700;1,700</code>

→ [Google Fonts Quickstart Guide](#)

# Third-Party Font Hosting – Adobe Fonts

To use these fonts on a web page, copy this code into the <head> tag of your HTML.

```
<style>
  @import url("https://use.typekit.net/ndg2ebt.css");
</style>
```



The [default embed code](#) is simplest to implement and ideal for most websites.

---

## Fonts Added:

Panel Sans Regular 

```
font-family: "panel-sans", sans-serif;
font-weight: 400;
font-style: normal;
```

**What are you thinking of  
making your website about?**



# Assignment #6

## **Homework**

- Midterm Reflection
- Assignment #6