# Class overview & looking ahead

CISC-2350-R01 | Fall 2017 | Week 15-1

Ruta Kruliauskaite

## Today's Agenda

- Attendance
- Course evaluation
- Web review presentations: Danielle, Julian
- Finals: working on a presentation
- Looking ahead: what else you should learn

## Course evaluation

## Web review presentations: Danielle, Julian

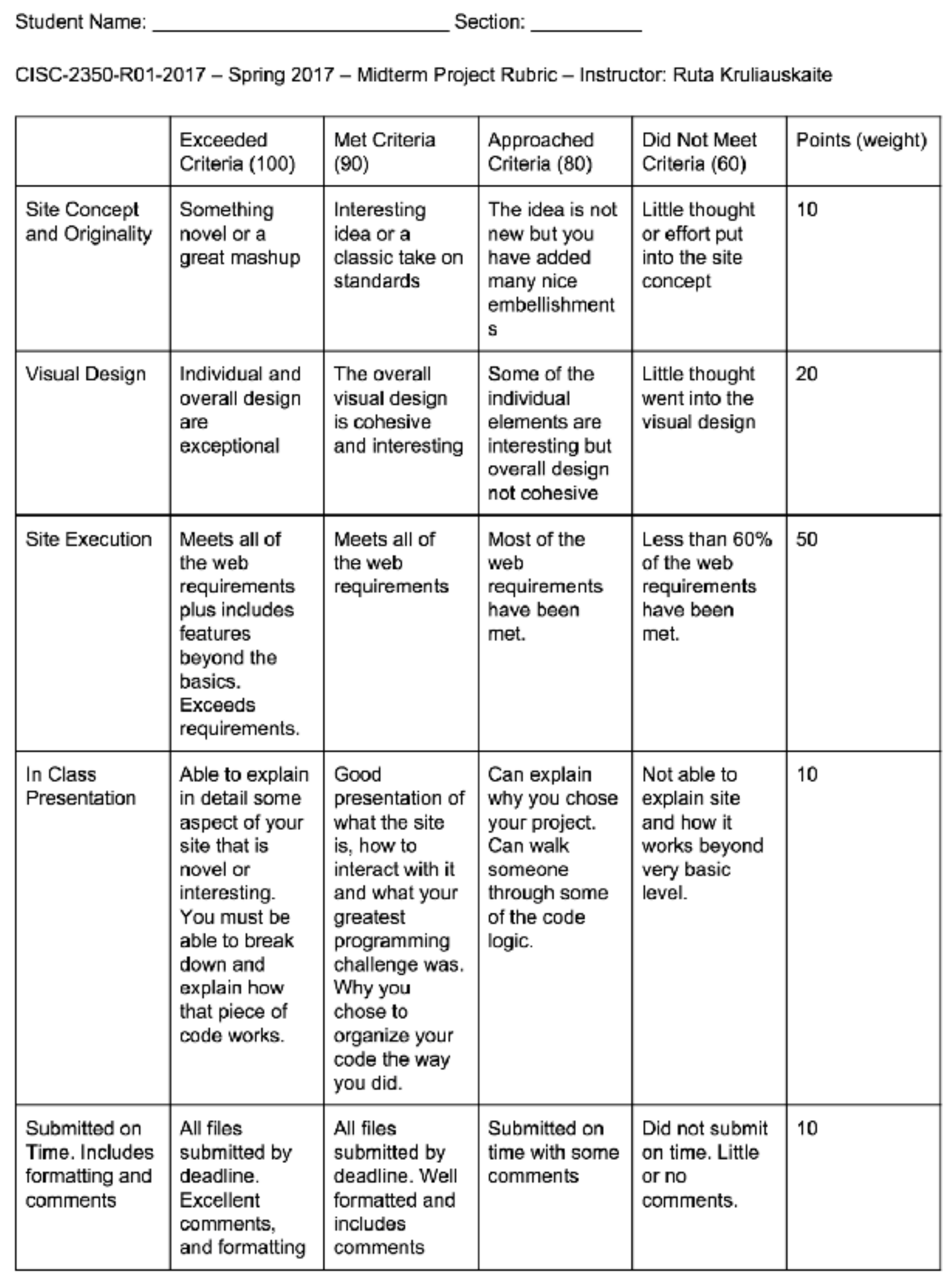
## Finals: presentation

## In class presentation 1

1. Thursday, December 7, 5:30-8:15pm.
2. Five minute presentation to the class.
3. There will be guest critics.
4. You will be graded according to a rubric on concept, execution, visual design and effective communication.

Link to description: https://github.com/rutaitp/CISC-2350-R01-2017/wiki/Final-project-assignment-(due-December-7)

# Final Project Grading



Student Name: _________________________ Section: __________

CISC-2350-R01-2017 – Spring 2017 – Midterm Project Rubric – Instructor: Ruta Kruliauskaite

| | Exceeded Criteria (100) | Met Criteria (90) | Approached Criteria (80) | Did Not Meet Criteria (60) | Points (weight) |
|---|---|---|---|---|---|
| Site Concept and Originality | Something novel or a great mashup | Interesting idea or a classic take on standards | The idea is not new but you have added many nice embellishment s | Little thought or effort put into the site concept | 10 |
| Visual Design | Individual and overall design are exceptional | The overall visual design is cohesive and interesting | Some of the individual elements are interesting but overall design not cohesive | Little thought went into the visual design | 20 |
| Site Execution | Meets all of the web requirements plus includes features beyond the basics. Exceeds requirements. | Meets all of the web requirements | Most of the web requirements have been met. | Less than 60% of the web requirements have been met. | 50 |
| In Class Presentation | Able to explain in detail some aspect of your site that is novel or interesting. You must be able to break down and explain how that piece of code works. | Good presentation of what the site is, how to interact with it and what your greatest programming challenge was. Why you chose to organize your code the way you did. | Can explain why you chose your project. Can walk someone through some of the code logic. | Not able to explain site and how it works beyond very basic level. | 10 |
| Submitted on Time. Includes formatting and comments | All files submitted by deadline. Excellent comments, and formatting | All files submitted by deadline. Well formatted and includes comments | Submitted on time with some comments | Did not submit on time. Little or no comments. | 10 |

https://github.com/rutaitp/CISC-2350-R01-2017/blob/master/week6/grading_rubric.pdf

# In class presentation 2

**Presentation structure:**

1. Introduction & context (roughly 30sec):
    1. Your name
    2. What is it that you made?
    3. Why this project?
2. Tour of a website (roughly 2min):
    - overview of different pages and elements utilized
    - visual choices - why did you choose these colors and fonts?
    - how you want user to interact with your site?
    - if something doesn't work, what is it supposed to do?
3. Code overview (roughly 1.5min):
    - explain your code - structure, html elements, css, javascript
    - Find a specific part of code that you think is interesting , took you some time, or you're proud of
    - What's the logic of the code for this portion between the HTML, CSS and JAvaScript? If there's transitions, how does it work?
4. Closing (roughly 30sec):
    - what did you learn?
    - what did you wish you did differently?
    - how you would take this project further

**Five minutes is not a long time - rehearse, rehearse, rehearse!**

## Class Overview

HTML CSS JS

## What else should I know?

## 1. Web development

## What else should I know?

If you want to continue understanding web design, you’ll need to be able to at least recognize and know how the following works:

- jQuery
- Server side programming
- Web hosting solutions
- Terminal
- Git

...and a bunch of other stuff...

## jQuery

**jQuery** is a JavaScript library that simplifies scripting for HTML pages using DOM element selectors and a set of animations, transitions and other manipulations based on CSS and HTML. It can accomplish tasks quickly and consistently across all browsers:

- Select elements
- Perform tasks
- Handle events

## jQuery

**jQuery** is:

- Cross-browser, no need to write fallback code
- Selectors work like CSS and require less code
- Event handling is easier and one method works on all browsers
- Once you have made a selection, you can apply multiple methods to it

The basic syntax: $(“selector”).action() - **selector can be anything**.

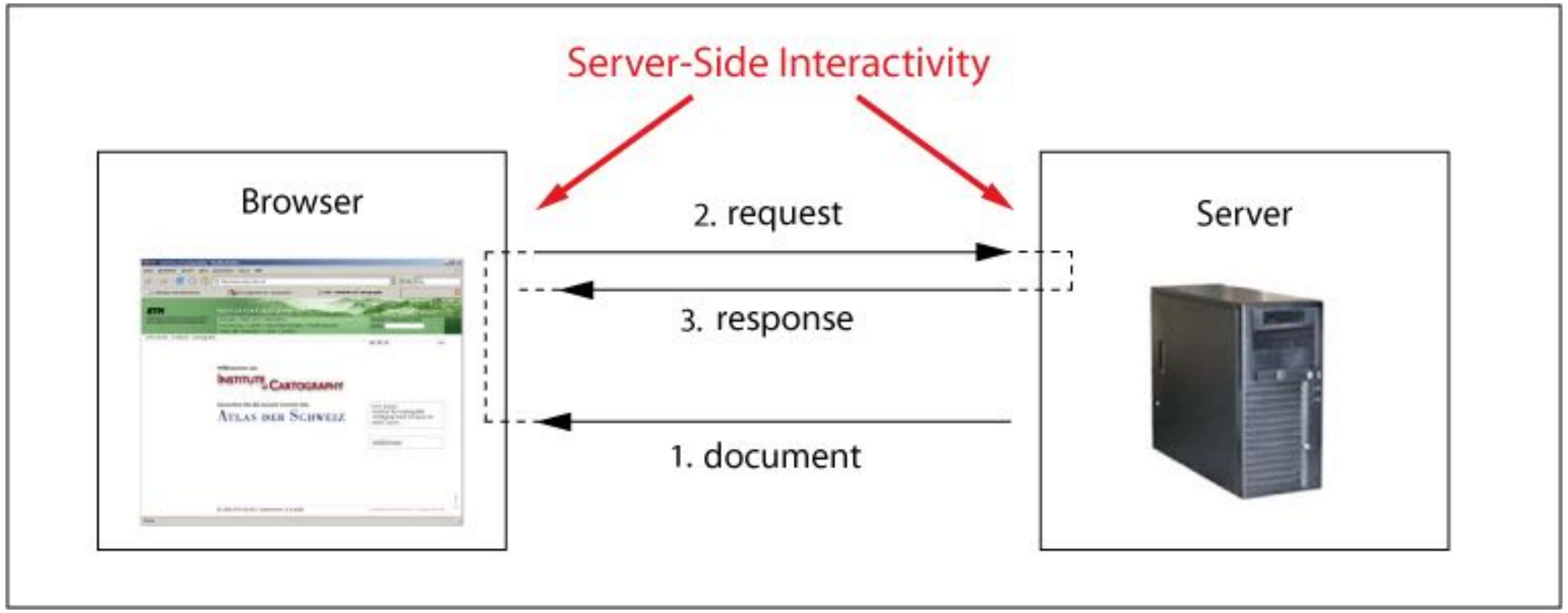
## jQuery

- http://jquery.com/download/
- https://www.w3schools.com/jquery/
- https://www.codecademy.com/learn/learn-jquery

# Server-side programming

Everything we've dealt with in the class was client side.

We've hosted all the files locally, but in theory a server renders these pages and "serves" them up for clients on request



## Server-side programming frameworks

There are a lot of back-end programming languages you can use. Here are a few:

- **Django (Python):** https://www.djangoproject.com/
- **Node.js**, a JavaScript framework: https://nodejs.org/en/
- **Express.js**, a Node.js module for web applications (handling routes, etc.): http://expressjs.com/

## Web hosting

Two things you will need to host websites:

- To have a server online
- And software to handle it

Digital Ocean is a cheap cloud server system ($5/month) that is very easy and quick to set up and runs of node.js.
CyberDuck is an open source client for FTP and SFTP to handle your Digital Ocean server (to upload files, edit code on the server, etc.).

## Terminal

To work with server-side programming you’ll have to run most of the commands from your terminal, also known as “command line”.

**Terminal** is an interface in which you can type and execute text based commands and control your computer.

- https://www.wired.com/2010/02/learn_enough_unix_for_your_resume/
- http://mally.stanford.edu/~sr/computing/basic-unix.html
- https://www.learnenough.com/command-line-tutorial
- https://www.codecademy.com/learn/learn-the-command-line

## Git

If you know how to use terminal, you also need to know how to use Git.

**Git** is a version control system for tracking changes in computer files and coordinating work on those files among multiple people.

- https://try.github.io/
- https://www.codecademy.com/learn/learn-git

## 2. JavaScript libraries

## socket.io

**WebSockets** is another way to make a communication between the user's browser and a server possible.

The connection between the client and server runs persistently and if there is any change or new data, it can be instantly sent from client to server or server to client with very little latency.
It enables real-time bidirectional event-based communication.

**Project examples include:**

- Chat application (text or video using WebRTC)
- Collective drawing application

## socket.io tutorials

Example:

- https://socket.io/demos/whiteboard/

Tutorials:

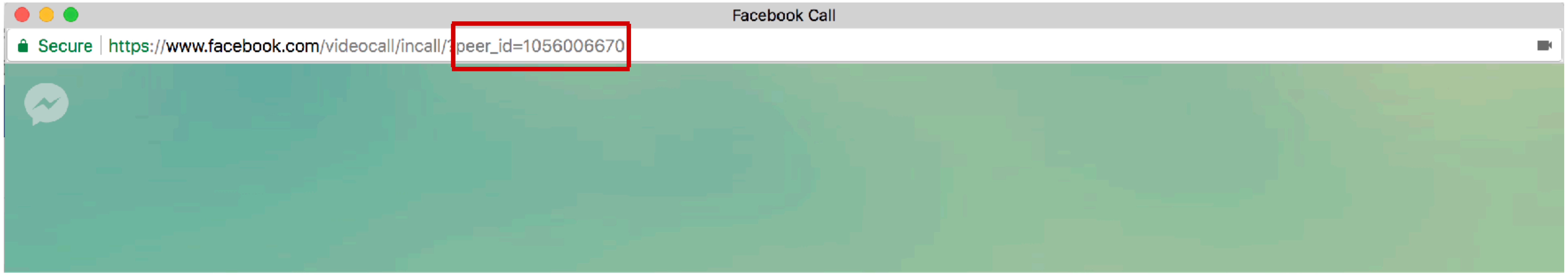
- https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API
- https://socket.io/
- https://socket.io/get-started/
- https://github.com/socketio

## peer.js

**If you want to create live peer2peer applications.**

**Project examples include:**

- Video chat applications


Facebook Call
Secure
https://www.facebook.com/videocall/incall/?peer_id=1056006670

## peer.js

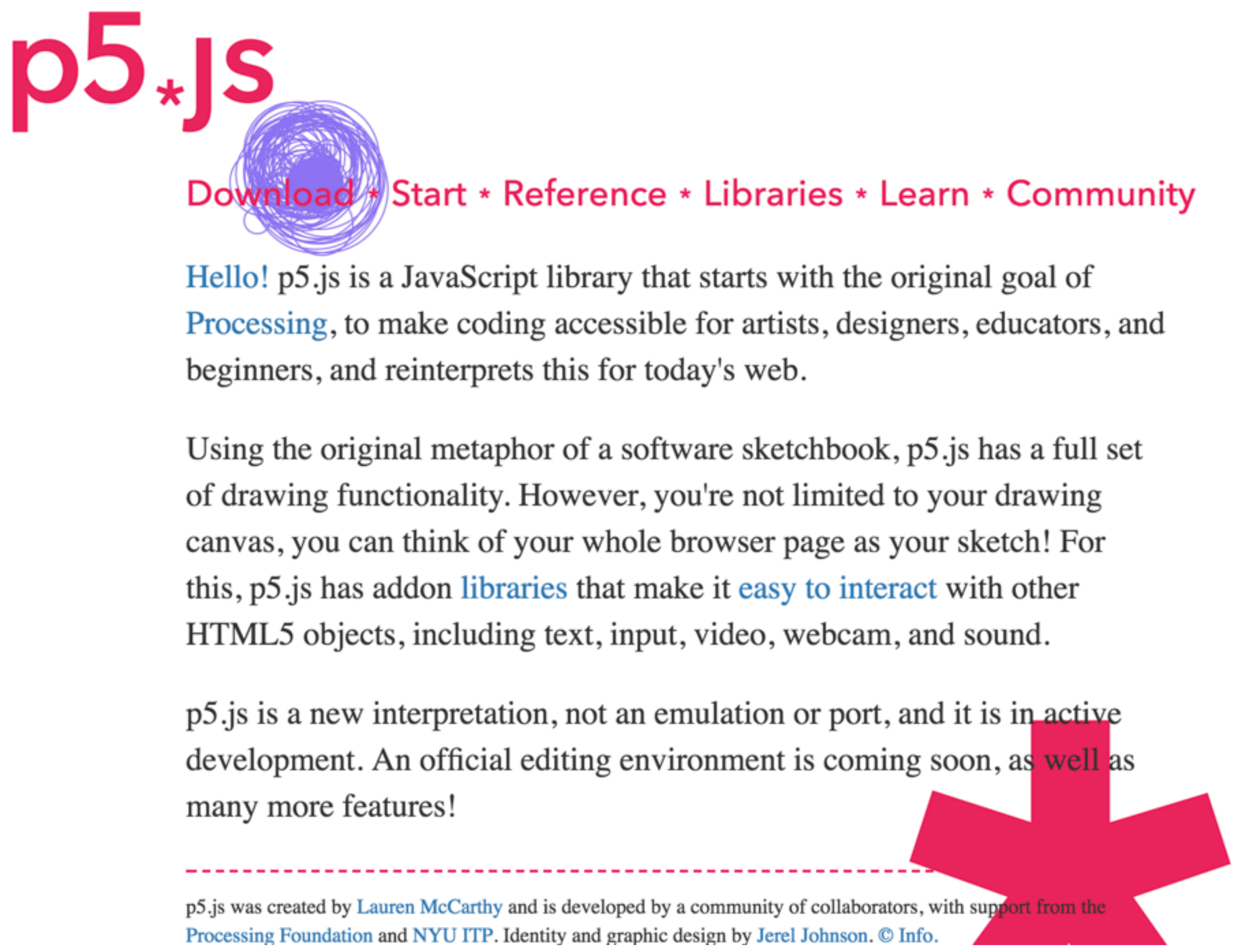
Example: http://cdn.peerjs.com/demo/videochat/

Tutorials:

- http://peerjs.com/docs/

## p5.js

**p5js** is a JavaScript library for creating coding: it makes drawing and creating graphics much easier. It's meant to make code more accessible to artists, designers, educators and everyone who wants to code.


p5.js
Download * Start * Reference * Libraries * Learn * Community

Hello! p5.js is a JavaScript library that starts with the original goal of Processing, to make coding accessible for artists, designers, educators, and beginners, and reinterprets this for today's web.

Using the original metaphor of a software sketchbook, p5.js has a full set of drawing functionality. However, you're not limited to your drawing canvas, you can think of your whole browser page as your sketch! For this, p5.js has addon libraries that make it easy to interact with other HTML5 objects, including text, input, video, webcam, and sound.

p5.js is a new interpretation, not an emulation or port, and it is in active development. An official editing environment is coming soon, as well as many more features!

p5.js was created by Lauren McCarthy and is developed by a community of collaborators, with support from the Processing Foundation and NYU ITP. Identity and graphic design by Jerel Johnson. © Info.

P

## p5.js

It’s built on drawing on a Canvas principle, which we learned last week.

It has its own web editor:

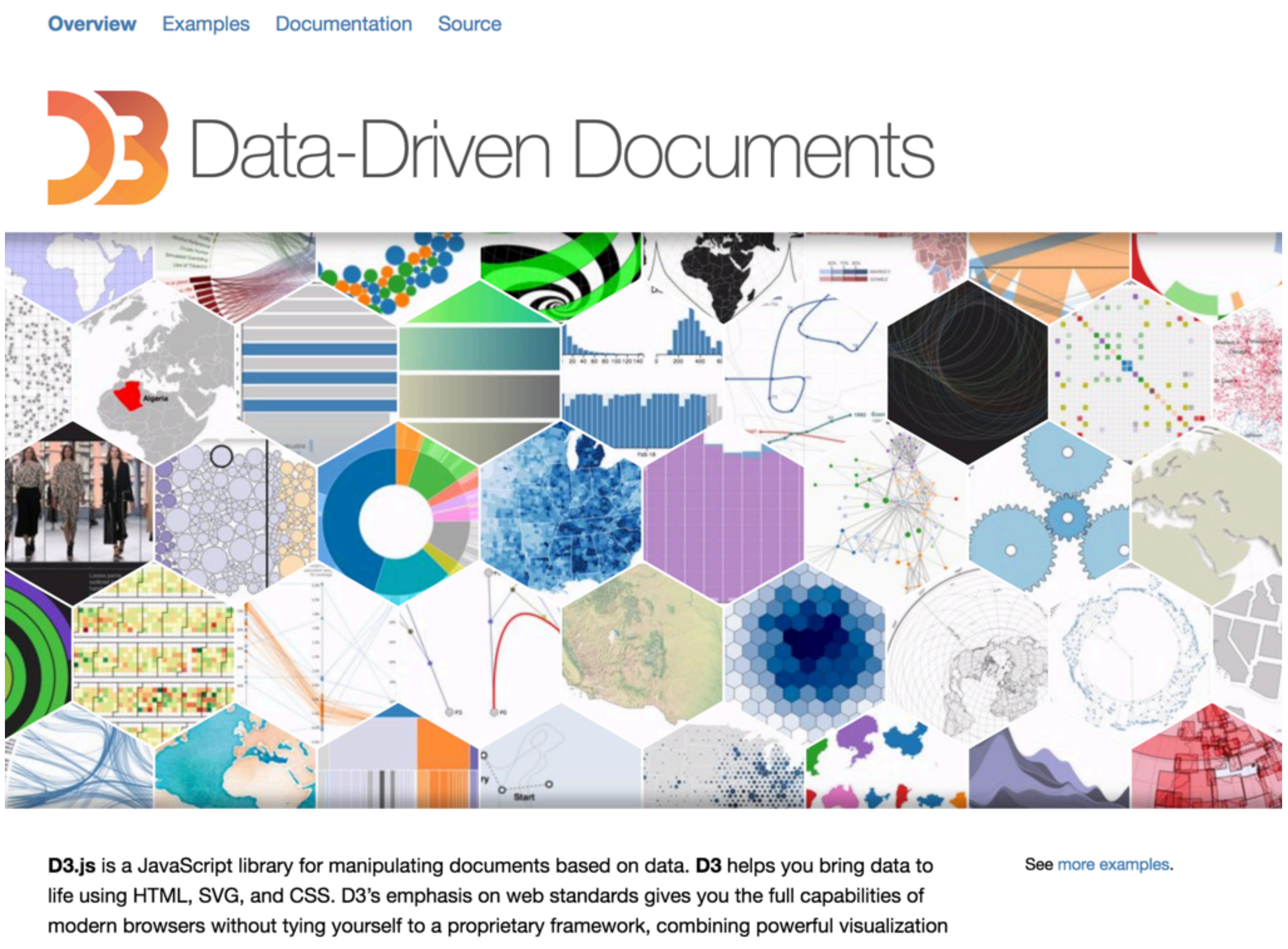
- https://alpha.editor.p5js.org/

Documentation:

- https://p5js.org/reference/

# d3.js

**D3.js** is a JavaScript library for producing dynamic, interactive data visualizations in web browsers.



**D3.js** is a JavaScript library for manipulating documents based on data. **D3** helps you bring data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization

See more examples.

# d3.js

Examples:

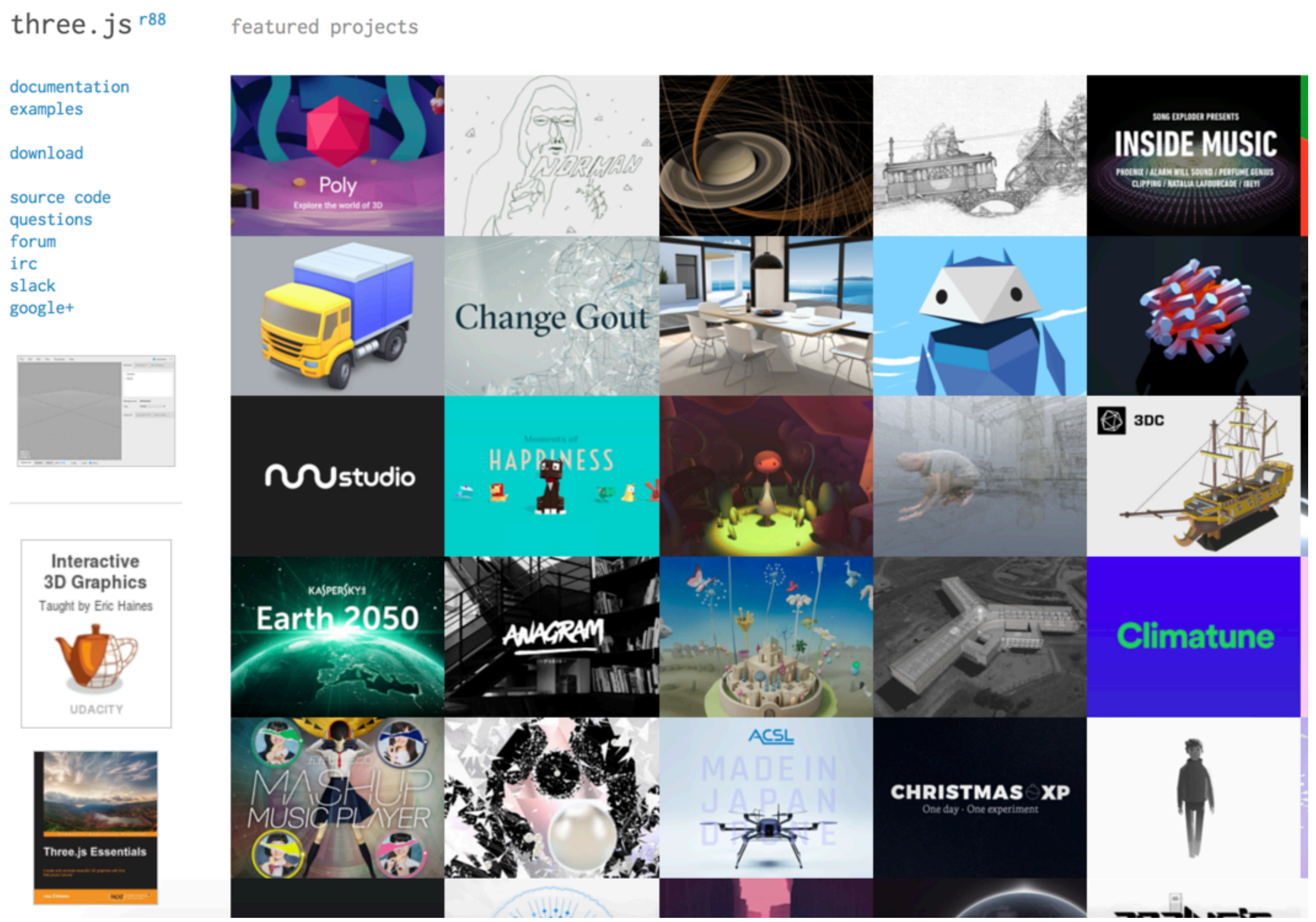
- https://github.com/d3/d3/wiki/Gallery

Documentation:

- https://github.com/d3/d3/wiki

# three.js

**Three.js** is a cross browser JavaScript library used to create and display animated 3D computer graphics in a web browser.



## three.js

Examples:

- https://threejs.org/examples/

Documentation:

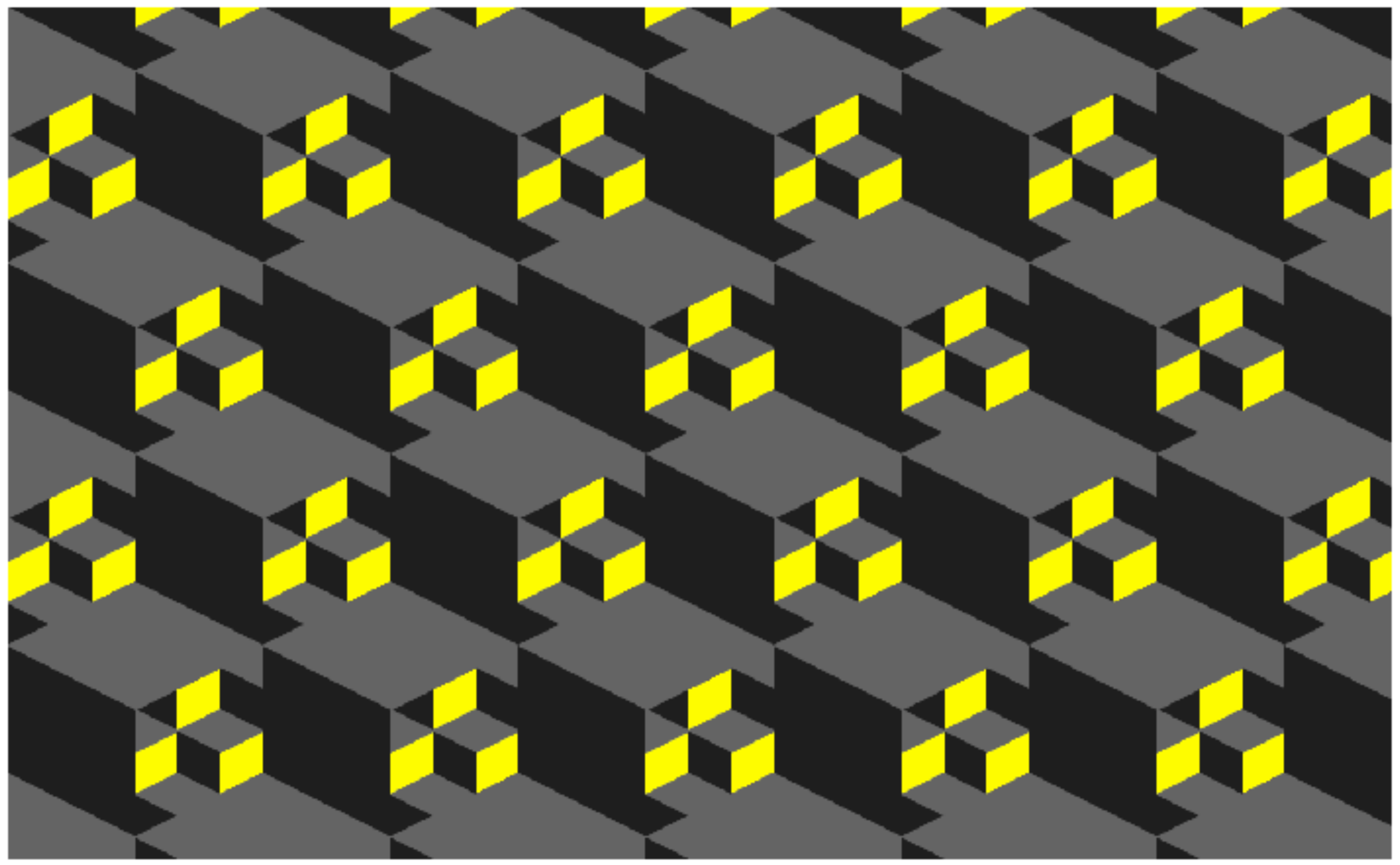
- https://threejs.org/docs/index.html#manual/introduction/Creating-a-scene

## rune.js

**Rune.js** is a JavaScript library for programming graphic design systems in a browser.

### Rune.js

Rune.js is a JavaScript library for programming graphic design systems with SVG in both the browser or node.js. It features a chainable drawing API, an unobtrusive scene graph, and a range of features aimed specifically at graphic designers: native support for color conversion, grid systems, typography, pixel iteration, as well as an expanding set of computational geometry helpers. Oh, and it uses virtual-dom under the hood.



## rune.js

Examples:

- http://printingcode.runemadsen.com/examples/

Documentation:

- http://runemadsen.github.io/rune.js/documentation.html

## More creative coding toolkits

Processing: **https://processing.org/** (Java)
OpenFrameworks: **http://openframeworks.cc/** (C++)
Cinder: **https://libcinder.org/** (C++)

## Homework assignment

## Homework

1. **Finish working on your final projects:**
   - You should have fully functioning websites ready
2. **Prepare 5min presentation:**
   - no need to send slides in advance
3. Office hours available after class for final questions before presentations on Thursday, December 7