Syllabus

Class: Advanced Creative Coding, WebGL | DM-UY 4913 C

Term: Spring 2022

Instructor: Andrew Cotter

Location: 370 Jay Street, Room 310, Brooklyn Campus

Instructor Email: andrew.cotter@nyu.edu

Class Time: Mondays & Wednesdays, 8:00AM - 9:50AM

Prerequisites

Students should already be very familiar with JavaScript, and/or have previous experience with strongly typed languages and scene-graph style frameworks like Unity. Students don't need to be experts in JS, but should already know how to make their own functions, classes, or extended classes as well as have some familiarity with or willingness to be quickly familiar with git and the terminal.

Course Description

WebGL is a popular tool in creative coding which can be used to create experiences ranging from website animations and games to full-scale interactive installations.

This course will focus primarily on two WebGL frameworks: Three.js, and Pixi.js, tools to create interactive 3D and 2D web experiences. We'll then learn how to bring these experiences to life using animation libraries, make networked desktop experiences with Electron, and cover how to use popular transpiled languages like Typescript to write more robust JavaScript programs.

Program Learning Objectives

In this course students will:

- develop **conceptual thinking skills** to generate ideas and content in order to solve problems or create opportunities.
- develop **technical skills** to realize their ideas.
- develop **critical thinking skills** that will allow them to analyze and position their work within cultural, historic, aesthetic, economic, and technological contexts.
- gain knowledge of **professional practices and organizations** by developing their verbal, visual, and written communication for documentation and presentation, exhibition and promotion, networking, and career preparation.
- develop **collaboration skills** to actively and effectively work in a team or group.

Course Learning Objectives

In this class, students will:

- Design, program, and author their own 2D and 3D software with Three.js and Pixi.js
- Develop an understanding how to compose and architect their software with TypeScript and Electron
- Be comfortable talking about their project ideas and presenting their work

Statement On Inclusion

The NYU Tandon School values an inclusive and equitable environment for all our students. I hope to foster a sense of community in this class and consider it a place where individuals of all backgrounds, beliefs, ethnicities, national origins, gender identities, sexual orientations, religious and political affiliations, and abilities will be treated with respect. It is my intent that all students' learning needs be addressed both in and out of class, and that the diversity that students bring to this class be viewed as a resource, strength and benefit. If this standard is not being upheld, please feel free to speak with me.

Important Dates

- First day of classes : Mon, Jan 24
- Last day of Drop/Add : Sun, Feb 6
- No Classes: Mon, Feb 21
- Spring Break (No Classes): Mar 14-20
- Midterm grades due (undergrads only): Mon, Apr 4
- Last day to withdraw with a 'W' : Tues, May 3
- Last day of classes : Mon, May 9

Course Outline

Week	Topic	Homework
1	Intro: Syllabus Review, Reintroduction to JS, a primer on Typescript	Primer readings on JS, TS, and WebGL
2	Pixi.js 1: Scene-graph workflow, Sprites/Graphics, Polymorphism with TS	Iterative Pattern Assignment and Readings
3	Pixi.js 2: Interaction, Events, and Composing Scenes with Model-View-Control Pattern	Clock Assignment and Recommended Readings
4	Pixi.js 3: Animations and Animation Libraries (GSAP)	Waves Assignment
5	Pixi.js 4: Electron/Node.js	Midterm Assigned
6	Pixi.js 5: Networking: OSC, MIDI, Serial	Work on Midterm
7	**Midterm Presentations**	Primer Readings on Three.js
8	**Spring Break**	
9	Three.js 1: Review Scene-graph workflow review, Meshes and Importing Models	Still Life Assignment
10	Three.js 2: Interaction and GUI	Interaction Assignment
11	Three.js 3: Electron/Node Review, Networking Review	Physical Interface Assignment
12	Three.js 4: Animation and Animation Libraries	Final Assigned
13	Three.js 5: Fragment Shader Review, Vertex Shaders	Work on finals
14	Semester Review	Finish Finals
15	**Final Presentations**	
16	**Final Presentations**	

Materials and Supplies

Required Reading

- Programming Typescript by Boris Cherny (Provided on course site)
- WebGL Fundamentals
- Three.js Manual
- Learning Pixi

Recommended Reading

- [Eloquent JavaScript][https://eloquentjavascript.net/]
- The Typescript Handbook
- Mozilla Docs on WebGL

Watch/Listen

- Getting started with Typescript, Ladybug Podcast
- Pixi.js Tutorial, Wael Yasmina
- Three.js Getting Started, Design Course

Academic Integrity

Plagiarism

In programming, it is often difficult to strictly define "plagiarism". In this course, we'll be using a number of different open source, copyable, editable, code resources created by others. You will be expected to cite all code contributions which you have not written yourself, or are directly inspired by another source. This includes recreations of artworks, digital media, completed assignments, and the work of your peers.

It's extremely important to cite any tutorials, examples, or inspiration that you use for your homework. I am less strict about proper MLA formatting as long as the creator and a link to their work is included. If you choose to use a library, tutorial, or a snippet of code to figure out a specific technical implementation that is a small part of your idea for the assignment and cite that resource in your class blog and in comments in your source code, you should be in the clear. If, however, you are using a tutorial or existing code as the template for your assignment, things can get murky very quickly.

Things to watch out for: It sometimes happens that an artist or developer places the entire source code for their sketch, app, or artwork online, as a resource for others to borrow and learn from. The assignments professors give in new-media arts courses are often similar; you may discover the work of a student in some other class or school, who has posted code for a project which responds to a similar assignment. *You should probably avoid this code*. At the very least, you should be very, very careful about approaching such code for possible re-use.

If it is necessary to do so, it is best to extract components that solve a specific technical problem, rather than those parts which operate to create a unique experience. **Your objective, if and/or when you work with others' code, is to make it your own**. Simply taking an artwork from someone's page on GitHub, Glitch, OpenProcessing, etc., and just changing the colors would be lazy. Doing so without proper citation would be plagiarism. If you don't know how the code you're reusing works, that can be a red flag to re-consider whether or not to use the code in the first place. If you do know how the code works, it's a healthy exercise to try re-implementing it on your own without copying to try avoiding this grey area.

Collaborations

Collaborations must be approved prior to work being submitted. If you'd like to create work with another individual, you must propose this collaboration prior to starting work on your assignment. You must also describe how both individuals will contribute to the project. Other students can absolutely assist you with your work, but they should not be writing code for you. A

good test is if you are writing the code yourself and FULLY understand how the code you are using works.

It is categorically forbidden to copy code from another source without citing it, to “receive help” from another student in the form of copied or directly provided code, or to present a work that you have not created as your own. This includes paying another individual to create work for you.

Ask First

If you have a question as to whether or not the work you are attempting to submit is plagiarism, ask first, do not submit first. I am happy to discuss the nuances in detail.

Logistics

Assignments

You will be expected to create a class repo and create a folder in it for each assignment completed. Assignments will have various expectations, including photo and video documentation of your work, photos of sketches and preparatory materials, and links to functioning projects / code used to create those projects. It's good practice to make a readme file in each assignment's folder to keep your documentation handy and readable.

Example folder structure:

[studentName]

|--> [Week 1]

| |--> [source code]

| |--> readme.md with documentation/process

| --[Week 2]

| |-- [source code]

| |-- readme.md with documentation/process

... etc

Code

Most assignments will be submitted using an instructor-accessible repo hosted on a site like Github or Gitlab.

Class Communication

Email/Brightspace

Assignment and lecture updates as well as class announcements will be posted via Brightspace.

Slack

Questions, comments, assignment, and lecture updates may also be posted to a classroom slack channel. Please stay tuned for more details.

Office Hours

Weekly

Each week, I'll aim to have at least 2 hours of open time where anyone can jump into a video channel and ask questions. This can be used for help with assignments, or review. Attending these office hours with questions is very helpful, but not required. You should ensure that you have reviewed the weeks asynchronous material prior to joining the office hours however so that we can make the most of our time!

By Request

If you are unable to attend the stated office hours, you may request additional help. Send me a message and we'll work out additional times that work best!

Grading & Attendance Policy

Breakdown

Weekly Assignments: 30%
Midterm Project: 30%
Final Project: 30%
Class Participation: 10%

Midterm Project

Your midterm project is designed to demonstrate competency in the core programming skills addressed by the course. In this section, you will also be expected to add a personal touch to your midterm project. More details to come towards the middle of the semester.

Final Project

Your final project will be a self directed exploration of creative coding concepts and techniques. You will be expected to design, prototype, and execute a functioning project that deeply explores various new media techniques, demonstrates mastery of the skills you have been learning, and stands on its own as a compelling work. More details to come toward the end of the semester.

Weekly Assignments

Each week, you'll be expected to create at least one software sketch that demonstrates core programming concepts and/or creative coding techniques. You'll be expected to document your work, any problems you encountered, and any questions you have. We will spend time in class reviewing these assignments, and the concepts covered.

Additionally, there may be weekly readings, accompanying videos, or other materials to reinforce or help you explore beyond the materials covered in class.

Late Work

You must submit your work by the assigned due date. You will lose a letter grade for any work that is submitted up to 1 day after the deadline. Any work submitted after over 24 hours late will receive an automatic failure, unless you have an excused assignment, or assignment extension granted by the professor.

Documentation

Documentation is expected to cover both the work itself, and your process of creating the work. Each assignment will be accompanied by an online blog post showing the final work, in progress sketches, photos, and videos, and commentary on your process. You absolutely should be documententing your work as you go. For interactive projects, you'll be expected to have a section of your documentation that describes how your project works, and how someone might use it.

Do not save this until the last minute. Document as you go. Some assignments may ask for specific documentation elements. The quality of your documentation will impact your final assignment grade, so make sure that text, photos, and video documentation are of decent quality.

Critique and Review

The work we create weekly will be reviewed and critiqued by your classmates. These critiques should be seen as an opportunity for constructive feedback to be provided and received on all aspects of the work we create. This process may be done both over video chat, and through asynchronous comments on your work, and will contribute to your final class participation score.

Academic Accommodations

If you are a student with a disability who is requesting accommodations, please contact New York University's Moses Center for Students with Disabilities at 212-998-4980 or mosescsd@nyu.edu. You must be registered with CSD to receive accommodations. Information about the Moses Center can be found at http://www.nyu.edu/csd. The Moses Center is located at 726 Broadway on the 2nd floor.

If you are experiencing an illness or any other situation that might affect your academic performance in a class, please email the Office of Advocacy, Compliance and Student Affairs: eng.studentadvocate@nyu.edu.

Semester Schedule

Below you will find a list of weekly topics. Examples and other information will be sent to the class via Slack. (Note: This schedule is likely to change as the semester continues.)

Week 1: January 24 & 26

Monday

- Syllabus Review
- Class Introductions
- Semester overview

Wednesday

- Typescript primer
- Workflow setup

Assignments
Due: 1/30 by 11PM ET

Reading:

- Programming Typescript, Chapters 1-3
- Optional: If you're less familiar with JS or need a refresher, review [Eloquent JS][7] chapters 1-6, and/or [this Coding Train Series][8] on the basics of JS with p5
- Optional: Listen to this podcast episode on [Getting started with Typescript][9]

Week 2: January 31 & February 3

Monday

- Getting started with Pixi
- Scene-graph workflow
- Graphics and Sprite Objects

Wednesday

- Filters and Fragment Shaders
- Classes with Typescript

Assignments
Due: 2/6 by 11PM ET

Iterative Pattern: write code to generate a tiling pattern or textural composition. Give consideration to aesthetic issues like symmetry, rhythm, color; detail at multiple scales; precises control of shape; and balance between organic and geometric forms. your pattern should be designed so that it can be infinitely repeated or extended. It may be helpful to use structures like for of/in/each loops to take care of repetitions.

Reading:

- Programming Typescript: Chapter 4
- Optional: The Book of Shaders, Chapters 0-4

Week 3: February 7 & 9

Monday

- Interaction/Events
- Adding a GUI

Wednesday

- Composing Scenes, Typescript Enums
- Model-View-Control Pattern

Assignments
Due: 2/13 by 11PM ET

Clock: Design a 'visual clock' that displays a novel or unconventional notion of time. Your clock should appear different at all times of the day, and it should repeat its appearance every 24 hours (or other relevant cycle, if desired). Challenge yourself to convey time without numerals.

Week 4: February 14 & 16

Content

- Animation with LERP-ing

Wednesday

- Animation Libraries (GSAP)

Assignments
Due: 2/20 by 11PM ET

Waves: Watch the music video for Waves by Max Cooper and try to recreate a scene or sequence from it. It may be helpful to use your knowledge of timeline animation and/or sinusoidal movement to emulate the animations from the video

Week 5: February 21 & 23

Content

Monday (Presidents' Day, No Class)

Wednesday

- Electron Overview
- Node.js Install/Setup

Assignments
Due: 2/28 by 11PM ET

- **Midterm:** Your midterm assignment is designed to demonstrate competency in the fundamentals of WebGL programming we have been learning. You will make your own interactive, screen-based artwork. Your sketch needs to contain at least a few objects (they can be from the same class) that behave independently from one another or compose together to facilitate the intended experience. Taking into consideration responsiveness, accessibility, and aesthetics there should be some form of interaction for your audience. This could be through keypresses, mouse movements, GUI objects, etc. Your code *must* be clearly and thoroughly commented throughout - indicating what is your original contribution and what you have gotten from elsewhere.

Week 6: February 28 and March 2

Content

- Networking: OSC, MIDI, Serial, Local JSON

Assignments
Due: 3/6 by 11PM ET

- Work on Midterms

Week 7: March 7 & 9

Monday/Wednesday

- Midterm Presentations

Assignments
Due: 3/13 by 11PM ET

Recommended Reading:

- Three.js Manual, Basics (Fundamentals - Setup) and Fundamentals (Primitives - Cameras)

Week 8: March 14 & 16

(Spring Break, No Class)

Week 9: March 21 & 23

Monday

- Three.js Basics
- Review TS/Electron Workflow with Three.js

Wednesday

- Importing models and textures

Assignments
Due: 3/27 by 11PM ET

Still Life: Create a scene which visually composes multiple 3D objects together. These can be primitive Three.js shapes, or models that you have made and imported into Three.js.

Recommended Reading:

- Three.js, Load an .OBJ file and Load a .GLTF file

Week 10: March 28 & 30

Monday & Wednesday

- Interaction/Events and GUI
- Importing Models

Assignments
Due: 4/3 by 11PM ET

Interaction: Create a sketch that adds objects based on interaction (mouse press, keystroke, etc) that interact with each other (collision, repulsion, etc.) *and/or* create a sketch that makes objects which have multiple dimensions of variability.

Week 11: April 4 & 6

Monday

- Electron/Networking Review

Wednesday

- Persistent state management via local JSON

Assignments
Due: 4/10 by 11PM ET

Physical Interface: Build a sketch with a physical interface. Take one of your earlier sketches in Three.js or make a new one and make it controllable via an Arduino circuit or midi controller

Week 12: April 11 & 13

Monday & Wednesday

- Animation & Animation Libraries

Assignments
Due: 5/1 by 11PM ET

Final: Your final project is wide open. This is your opportunity to create work that leverages the various tools, techniques and ideas we have explored throughout the semester. This includes interaction, data, design, interfaces, games, or any number of alternative forms of expression. You should create work that is engaging and compelling for your audience. Use this as an excuse to ask a question using computation, or to scratch an itch you've always wanted to explore. We will spend several weeks at the end of the semester working through your ideas on these projects.

Week 13: April 18 & 20

Monday

- Shaders Review for Three.js

Wednesday

- Vertex Shaders

Assignments
Due: 12/8 by 12PM ET

Recommended Reading:

- The Book of Shaders

Week 14: April 25 & 27

Monday & Wednesday

- Semester Review

Week 15: May 2 & 4

Monday

- Semester Review

Wednesday

- Final Presentations

Week 16: May 9

Monday

- Final Presentations