Game Development Experience
Where you are

Mostly narrow, played for fun, as a consumer

Playing experience

Design
Appreciation
Game Studies
History

Programming
Language
Data Structures
Environments

You are here

80K

12B

120

16?

17?

Teamwork
Ideation
Deployment
Finishing

Engines
Adv. Rendering
Efficiency
New Interfaces

Broad, play for study & analysis, as design, dev, and producer

Playing experience

You are here
Why You Are Here

➔ Learn the **basic principles** of game programming and put them into practice

➔ Learn how to do the low-level implementation so we can turn **ideas into working games**

➔ Learn how **technology and teamwork** affect game design (**PLO 7 & 8**)
The Team

Isaac Karth

➔ he/him
➔ Teaching CMPM 120
➔ ikarth@ucsc.edu
➔ Computational Media PhD student (I do procedural generation stuff)

Tad Leckman

➔ Teaching AGPM 120
Who are you?

If you think of yourself as a programmer:
What is one artwork that you like?

If you think of yourself as an artist:
What is one programming element that you like?
Who are you?

If you think of yourself as a programmer:
Albrecht Dürer, *Young Hare*

If you think of yourself as an artist:
(map (reduce '(:plus-python [list-comprehension])))
Some Bad News

Programming Videogames is Difficult

Especially in 10 weeks, in Summer, and working in a team.
var `Environment` = {
  code: ['HTML5', 'CSS', 'JavaScript'],
  framework: 'Phaser',
  collaboration: ['git', 'GitHub'],
  editor: ['Sublime Text', 'Atom', 'Chrome', 'etc'],
  server: 'Python'
}`
Desktop and Mobile HTML5 game framework

A fast, free and fun open source framework for Canvas and WebGL powered browser games.

DOWNLOAD & GET STARTED
Download or Fork via Github

PHASER FEATURES

WEBGL & CANVAS
PRELOADER
PHYSICS
SPRITES
GROUPS
ANIMATION
PARTICLES
CAMERA

INPUT
SOUND
TILEMAPS
DEVICE SCALING
PLUGIN SYSTEM
MOBILE BROWSER
DEVELOPER SUPPORT
WEB FIRST
Why Phaser?

➔ Free!
➔ Fast
➔ Because I said so!
➔ Actively supported and documented
➔ Well-structured and (generally) genre agnostic
➔ Gives us lots of game-specific functions “for free” (e.g., game loop, state management, physics, input handling, etc.)
➔ Lots and lots and lots of community resources
➔ Used in high-quality, actual, real-life, professional games
➔ Will help you learn to love again despite the deep void in your heart

Ten years ago, we would have used Flash

Nathan told me this, so it must be true
Phaser CE is the Community Edition of Phaser. In short, it’s a version of Phaser that you, the community, have direct control over. We started it off with the 2.7.0 release and then handed it over to you.

Phaser 2 was a massive milestone for us, and we’re still constantly amazed at all the cool things you’ve created, and continue to create with it. Thank you to everyone who has contributed to Phaser in one way or another, and helped shape the framework into the fantastic tool it is today.
Why not Phaser 3?

➔ Probably *will* be using 3 next Spring
  ◆ Nathan is in the process of updating his slides
➔ A few major features are still not in place
➔ The tutorials are in the process of being converted
➔ Phaser CE is not "worse" than Phaser 3
➔ Phaser CE is still actively supported
  ◆ Don't underestimate stability
➔ This is **not** the last time that you will encounter this situation in your career
There is no perfect...

...no perfect language
...no perfect framework
...no perfect engine

The sooner you learn this, the better.

You should be grateful I’m not forcing you to learn ClojureScript.
Your game's scope:
small
Your game's scope:

tiny
No
No
No
No
No
No
Ambitiously small  
Mechanic-centric  
Well-structured  
Expressive  
Idea-driven  

Achievable

2d :)
Some Good News

Every one of you can program a videogame

(And we have proof)
Lazy River: an endless drunker
Table Manners: a game about stealing sushi
Gravobot: a 2D puzzle platformer with a gravity orb mechanic
Tiny Steps: a storybook game wherein you play as the CUTEST mouse
rLDQ: a minigame collection of mundane daily tasks
The Light: a survival typing game (!?)
Some More Good News

Any style or genre you choose is fine with us*

*As long as you do so with creativity, thoughtfulness, and professionalism
The Runs: a very mature game
StumpJumper: a commercial UCSC game
Team of six, 10-minute game, 6 months, $25K Budget
Foreshadowing Your Future
Schedule Overview*

Week 1: Intro, Web Dev, Phaser Intro
Week 2: Loops, States, Assets, Pong
Week 3: JavaScript, Prefabs, Input, Collision
Week 4: Debugging, Camera, State Machines
Week 5: Tilemaps, P2 Physics, Runner Showcase
Week 6: Particles, git, Time
Week 7: Text, Fonts, CSS
Weeks 8–9: Audio, Guest Talks, Open Topics
Week 10: Final Presentations aka The TRIAL of WILL

*This will inevitably change a bit
There are no labs in the summer

However, I will have office hours:

Engineering 2, room 256

- Wednesday, 10am - 11am
- Thursday, 11am - 12pm
Canvas Resources

Syllabus: https://canvas.ucsc.edu/courses/26569
Policies & Expectations
Attendance

Respect your classmates' time

Attendance is mandatory for presentations
Lateness

This is a fast class and you need to stay on track.

Presentations need to happen in person
Devices

Respect the time and attention of those around you.

(And respect yourself too.)
Readings

Designed to help you learn the material
Slides & Source Code

These will be available in our class Drive folder a day or two after class
Collaboration & Help

You are encouraged to help each other, but don't shortcut your own programming.
The Citation Model

Link to your sources!

I trust you to program ethically and responsibly
Grading

25% Readings & Small Assignments

25% Endless Runner Project

50% Final Game
Disability Resources

Please let me know how I can help
Communication

Talk to us early and often.

Don't let problems pile up.
Respect & Honesty

A reciprocal operation.
We're excited you're here!

Any questions?
Your first reading assignment

"Learning Web Design"

There's a quiz.

It is supposed to be easy if you know the subject, and helpful if you don't.

You can retake it as many times as you want.

It is due by the end of Friday
Modern Web Development

HTML + CSS + JS
Apple is seeking a Front-End Developer to drive user experience innovations for apple.com. This developer will not only be responsible for defining the architectural strategy for front-end technologies, including HTML5, CSS3 & JavaScript, but for evangelizing that technology across the team and Apple as a whole.

Key Qualifications

- Comfortable with source version control software and package managers (SVN, Git, NPM)
- Well-versed in fundamental visual and interactive design discipline
- Strive to use web standards to build solutions using semantic markup, templates (Handlebars) and SASS
- Understanding of all major browsers and the special considerations required for all various quirks
- Competent JavaScript programmer who doesn’t need to rely on libraries to accomplish innovative interactions
- **Aware of the interplay between JavaScript and HTML & CSS, and can dynamically create, modify, and style elements on a page with ease.**
- Experience with WebGL is a plus.

Description

Lead development efforts on large scale web-based projects, ensuring robust and lasting solutions are implemented. Awareness of Apple’s mobile platform with the ability to build solutions that take advantage of the latest iOS features while remaining performant on the latest iOS devices. Maintain existing JavaScript libraries, making sure they support the engineering and creative needs of apple.com. Mentor team members. Educate on software development best practices and new technologies, especially HTML5 & CSS3. Innovate: Build things that people will blog and Twitter about.
HTML: semantic layer
(how a page is described)

CSS: presentation layer
(how a page looks)

JavaScript: interaction layer
(how a page behaves)
Before we start...

What the heck is the Internet?
On the Internet, but not actually *the* Internet...
Also not the Internet...
Still not the Internet...
This *is* The Internet (but not the Internet we're talking about)
The Internet is a diverse network of connected computers that use a variety of standardized protocols to send and receive information.
The World Wide Web is just one (of many) ways to send and receive information via the Internet.

It uses HTTP as its communication protocol, HTML as a language to describe and structure information, browsers to interpret HTML, and hypertext to link documents together.
What's this stuff?
Uniform Resource Locator (URL)

- **User Agent** (browser)
- **protocol**
- **subdomain**
- **domain**
- **top-level domain**

More stuff can go here.
protocol://subdomain.domain.tld:port-number/path?parameters
A basic model of how the web works
"Client-side" or "front end" applications run on our local machine.

- Limited by the resources of the local machine.
"server-side" or "back end" applications run on a remote machine

- Limited by the resources of the remote machine
- Are often virtual machines
  - Several can share a host machine
  - Or can be running on a cluster of host servers

May not even be a file: the html might be generated dynamically
You can serve websites locally
The easiest way to run a local server is with Python

Python 2:

`python -m SimpleHTTPServer`

Python 3:

`python -m http.server`

If you don't have Python on your machine, install it.

[https://www.python.org/downloads/](https://www.python.org/downloads/)

[https://www.anaconda.com/distribution/](https://www.anaconda.com/distribution/)
Technologies

**Client**
- HTML
- CSS
- JavaScript

**Remote**
- PHP
- Python
- Ruby
- Clojure
- node.js
HTML = ???
HTML = Hypertext Markup Language
Nathan Altice
Santa Cruz, CA

Dear Mom,

I had a fun time at summer camp today. I stole a kid's swimsuit and set it on fire. He didn't mind because we are friends.

When you have the time, please send me a birthday cake in the mail. It isn't my birthday, but I like cakes with my name on them. Please do not send cake in a poster tube like last time.

OK, that's all for now. Please don't touch the things in my room.

Love,
Nathan
Nathan Altice
Santa Cruz, CA

Dear Mom,

I had a fun time at summer camp today. I stole a kid's swimsuit and set it on fire. He didn't mind because we are friends.

When you have the time, please send me a birthday cake in the mail. It isn't my birthday, but I like cakes with my name on them. Please do not send cake in a poster tube like last time.

OK, that's all for now. Please don't touch the things in my room.

Love,
Nathan
Marking up text gives it structure and meaning.

- Language
- Semantic
HTML uses standardized tags to markup text

Tags provide semantic meaning to content

<!DOCTYPE>
<html>
</html>
All major browsers have developer tools

But every browser renders webpages slightly differently.

The browser wars! A period of civil war. Rebel open-source startups, striking from a hidden base, have won their first victory against the evil non-standards-compliant Internet Explorer...
View Source

The 90s web designer's best friend
In Safari: Preferences > Advanced > Show Develop Menu...

Welcome!

Here’s some text in a paragraph tag. If there’s lots and lots of text on the page, it will extend to the edge of the screen and look awful unless you change the actual size of the browser window.

Here’s a second-level header tag

This paragraph will be really short.

Here’s yet another paragraph. Notice how the browser automatically created a space between this paragraph and the paragraph above.

I’m about to make a list of music genres with silly names:

- Rock ‘n’ Roll
- Dubstep
- Vaporwave
- Chillwave
- Electronic
- Electro

Some inline tags

I can also nest tags inside of tags. So this paragraph can have nested <em>em</em> or <strong>strong</strong> text or even preformatted text for code examples.

And of course, it wouldn’t be the web unless you could link to an external page.
Developer Tools Keyboard Shortcuts

**Command+Option+I**

*OsX*

**F12 or Control+Shift+I**

*Windows*
CSS = ???
CSS = Cascading Style Sheets
CSS: Let me sum up

- CSS separates presentation from structure
- CSS is a separate language with its own syntax
- CSS statements are called rules
- Rules contain a selector and a declaration
- Style rules "cascade" downward
- CSS definitions may live in a `<style>` tag (usually bad) or be linked externally (much better)
```css
body {
  font-family: "Arial";
  font-size: 14px;
  background-color: #facade;
}

h1 {
  border: 1px dotted red;
}

h2 {
  font-variant: small-caps;
}

.green {
  color: green;
}

/* h1 {
  font-family: serif;
  font-size: 5em;
} */
```
But what about... What is HTML5?
HTML5 is the latest evolution of the standard that defines HTML. The term represents two different concepts:

- It is a new version of the *language* HTML, with new elements, attributes, and behaviors,
- and a larger set of technologies that allows more diverse and powerful Web sites and applications. This set is sometimes called *HTML5 & friends* and often shortened to just *HTML5*.

Designed to be usable by all Open Web developers, this reference page links to numerous resources about HTML5 technologies, classified into several groups based on their function.

- *Semantics*: allowing you to describe more precisely what your content is.
- *Connectivity*: allowing you to communicate with the server in new and innovative ways.
- **Offline and storage**: allowing webpages to store data on the client-side locally and operate offline more efficiently.
- *Multimedia*: making video and audio first-class citizens in the Open Web.
- *2D/3D graphics and effects*: allowing a much more diverse range of presentation options.
- *Performance and integration*: providing greater speed optimization and better usage of computer hardware.
- *Device access*: allowing for the use of various input and output devices.
- *Styling*: letting authors write more sophisticated themes.
An HTML element that allows us to draw graphics using scripting (i.e. JavaScript).
HTML5 provides a target container for our games
HTML, CSS, and JS combined
Introduction to the DOM

This section provides a brief conceptual introduction to the DOM: what it is, how it provides structure for HTML and XML documents, how you can access it, and how this API presents the reference information and examples.

What is the DOM?

The Document Object Model (DOM) is a programming interface for HTML and XML documents. It provides a structured representation of the document and defines a way that the structure can be accessed from programs so that they can change the document structure, style and content. The DOM provides a representation of the document as a structured group of nodes and objects that have properties and methods. Essentially, it connects web pages to scripts or programming languages.

A Web page is a document. This document can be either displayed in the browser window, or as the HTML source. But it is the same document in both cases. The Document Object Model (DOM) provides another way to represent, store and manipulate that same document. The DOM is a fully object-oriented representation of the web page, and it can be modified with a scripting language such as JavaScript.

The [W3C DOM] and [WHATWG DOM] standards form the basis of the DOM implemented in most modern browsers. Many browsers offer extensions beyond the standard, so care must be exercised when using them on the web where documents may be accessed by various browsers with different DOMs.

For example, the standard DOM specifies that the `getElementsByTagName` method in the code below must return a list of all the `<p>` elements in the document:

```javascript
var paragraphs = document.getElementsByTagName("p");
// paragraphs[0] is the first <p> element
// paragraphs[1] is the second <p> element, etc.
alert(paragraphs[0].nodeName);
```
“A Web page is a document. This document can be either displayed in the browser window, or as the HTML source. But it is the same document in both cases. The Document Object Model (DOM) provides another way to represent, store and manipulate that same document. The DOM is a fully object-oriented representation of the web page, and it can be modified with a scripting language such as JavaScript.”

MDN
JS = JavaScript
First developed in 1995 at Netscape (for Navigator 2.0)
Not really related to Java
Actually a scripting language (domain-specific for web environment)
Relies on host for input/output (e.g., browser)
Multi-paradigm (e.g., procedural, functional, OOP, etc.)
Dynamic (i.e., executes at runtime)
Loosely typed
Standardized as ECMAScript
Historically maligned/praised for its flexibility
JavaScript Types

- number
- string
- Boolean
- Object
  - Function
  - Array
  - Date
  - RegExp
- null
- undefined
// number
var year = 2019;
var course_number = 120;

// string
var name = "Isaac Karth";

// Boolean
var ownsCar = false;

// Object (Function)
var addNumbers = function(a, b) {
  return a + b;
}

// Object (Array)
var favGames = ['Thief: The Dark Project', 'SimCity 2000', 'Heaven's Vault', 'Crusader Kings 2', 'Pathologic 2']

// Object (Date)
var today = new Date(2019, 6, 25);

// Object (RegExp)
var re = new RegExp('\w');

var the_abyss = null;

undefined // it's complicated
Next Class:

Bring your laptop!
(if you have one)