Hacking The Browser

ITP, Spring 2020

Course Description

Web browsers were originally used only for displaying simple HTML pages, but over the years they have become supercharged all-powerful web execution machines. In this class we'll explore experimental new features and HTML5 APIs that allow browsers to communicate with the OS and their environment. APIs that will be covered may include: Battery Status, Geolocation, notifications, accelerometer usage, video access, speech recognition, and text-to-speech. We'll cover the mechanics of bookmarklets and Chrome extensions, with a sustained multi-week focus on building extensions and exploring Chrome's extensions APIs. Class workshops will include projects such as building an ad blocker, programmatically replacing text and images on a website, and making sites that respond to external events. Students will give weekly in-class presentations on web capabilities, complete small weekly assignments. This class leans heavily on web technologies, and experience with HTML, CSS and modern JavaScript (ICM with p5.js or Commlab Web/Networked Media) is required. Introduction to Computational Media is required.

Instructor

Cory Forsyth <cf831@nyu.edu>

Mandatory Tisch Syllabus Statements

STATEMENT OF ACADEMIC INTEGRITY

Plagiarism is presenting someone else's work as though it were your own. More specifically, plagiarism is to present as your own: A sequence of words quoted without quotation marks from another writer or a paraphrased passage from another writer's work or facts, ideas or images composed by someone else.

STATEMENT OF PRINCIPLE

The core of the educational experience at the Tisch School of the Arts is the creation of original academic and artistic work by students for the critical review of faculty members. It is therefore of the utmost importance that students at all times provide their instructors with an accurate sense of their current abilities and knowledge in order to receive appropriate constructive criticism and advice. Any attempt to evade that essential, transparent transaction between instructor and student through plagiarism or cheating is educationally self-defeating and a grave violation of Tisch School of the Arts community standards. For all the details on plagiarism, please refer to page 10 of the Tisch School of the Arts,

Policies and Procedures Handbook, which can be found online at:
http://students.tisch.nyu.edu/page/home.html

STATEMENT ON ACCESSIBILITY

Academic accommodations are available for students with documented disabilities. Please contact the Moses Center for Students with Disabilities at 212 998-4980 for further information.

STATEMENT ON COUNSELING AND WELLNESS

Your health and safety are a priority at NYU. If you experience any health or mental health issues during this course, we encourage you to utilize the support services of the 24/7 NYU Wellness Exchange 212-443-9999. Also, all students who may require an academic accommodation due to a qualified disability, physical or mental, please register with the Moses Center 212-998-4980. Please let your instructor know if you need help connecting to these resources.

STATEMENT ON USE OF ELECTRONIC DEVICES

Laptops will be an essential part of the course and may be used in class during workshops and for taking notes in lecture. Laptops must be closed during class discussions and student presentations. Phone use in class is strictly prohibited unless directly related to a presentation of your own work or if you are asked to do so as part of the curriculum.

Work

Weekly Project Work

Each week we will cover a new browser capability in class and you will have a homework assignment to experiment with what we learned in class and be prepared to demonstrate our work the following week.

Weekly Student Presentations

Each week we will have 2-3 student presentations. Each class, 2-3 students will sign up for a presentation slot at the next class. For your presentation, you'll select one of the topics on the list of presentation topics (or optionally suggest your own). The presentation should be 5-7 minutes long (please pay attention to time as I may cut you short if your presentation runs much longer than 10 minutes). It should include slides or code, as needed, to best explain your topic to the class. Every student *must* give a presentation.

Participation

Please be prepared each week to pay attention, be focused on the materials presented, and be ready and willing to join in group discussions with the class. You are expected to help create an

environment where we can all learn from and teach one another. Make an effort to *arrive to class on time*.

Laptop Policy

A lot of the work we'll be doing in class will involve sharing code samples and live coding together in class. When I am in front, you may have your laptop open when it makes sense to follow along with what we are working on. **When your classmates are presenting**, **laptops should be closed.**

Grades

- 40%: In-class participation
- 30%: In-class presentation
- 30%: Homework

Class Conduct

- This class meets for one two-hour-and-55-minute session per week
- Assignment dates are hard due dates
- If you know in advance that you will not be able to attend a class, let me know beforehand
- More than 1 unexcused absences may result in failing the class
- There is a lot to learn, so come on time

Important Links

This syllabus is the only link you need to remember. Everything else should be linked to from this syllabus, somewhere. Look at the list immediately below or look at the links for each week below. Links to slides presented in class will be with each week's section below.

- This syllabus is also at http://hackingthebrowser.com/
- The class github repository: https://github.com/ITPNYU/hacking-the-browser/
    - Code samples will be added to this repo
    - This repo also has a wiki that has code snippets and the presentation topics list
- Office Hours link

Class Schedule

Week 0: Prerequisites and Supplemental Material

Supplemental Readings:

- If unsure, start here:
    - Browser, DOM and Events chapters of Eloquent JavaScript
    - How the Web Works — Chapters 1 and 3
    - MDN: What is JavaScript
- If you are uncomfortable with HTML or CSS:
    - Follow the MDN CSS Intro
    - Follow the MDN HTML Intro
- If you are uncomfortable with JavaScript:
    - Read this JavaScript Primer or JavaScript for Cats
    - Follow the additional exercises and readings in the MDN What Is JavaScript guide
    - Short JavaScript Primer or JavaScript for Cats (<-- this one is fun if you like cats)
    - JavaScript Basics Course
    - JavaScript Fundamentals from our friends at Mozilla
- If you are uncomfortable with jQuery:
    - jQuery Fundamentals
    - How jQuery Works
- Intro to HTML/CSS Course
    - More advanced material:
        - How does the Internet Work? and HTTP Made Easy are both good background, a bit less relevant to our class though.
- Resilient Web Design ebook — Philosophical thoughts about where the web came from and where it is going.

Week 1: Introduction and Overview, HTML5 APIs

- Introduction to browser capabilities
    - “Browser wars”
    - How the web and browsers are changing today
- “HTML5”, JavaScript, APIs
    - What do these mean? What is an API?

    - How do we use HTML5 and Browser APIs?
- Why not native?
    - Why build for the web and the mobile web?
- JavaScript Review
    - What are JavaScript callbacks?
    - What are JavaScript events?
    - Is JavaScript synchronous or asynchronous? (What's the difference?)
- HTML, CSS, DOM Review
    - What is the DOM?
    - How do we interact with the DOM?
    - How does the browser create the DOM?
- Types of JavaScript: DOM API, "pure" JavaScript, Node.js
- Student Introductions
- HTML5 APIs:
    - Battery Status
    - Geolocation
    - Devicemotion
    - Orientation

Week 1 Slides

Week 1 Links

- In-class demos:
    - Battery API Demo
        - Works on Chrome desktop and Android, not Safari (desktop or iOS) or Chrome iOS
    - Geolocation API Demo
        - See also Trip Meter, GeoLocation API Demo, Device Orientation API
    - Other demos on codepen
- Web Browser Capabilities: https://whatwebcando.today/ and https://davidwalsh.name/more-html5-apis
- Links to look at in class:
    - hoooooooooooooooooooooooooooooooooooooooooooooooooooooooooorse
    - The Useless Web
    - Eel slap!
    - Browser Pong
        - and http://weareinstrument.com/ball/
    - DEFENDER OF THE FAVICON by @p01
    - Redditors design worst volume sliders possible – Designer News
    - CSS Zen Garden: The Beauty of CSS Design
    - Pure CSS minesweeper

    - Chromium Development Calendar and Release Info - The Chromium Projects
    - Principle of Least Power
    - Atwood's Law
    - DOM Tree
    - Checkwave
    - Web Bluetooth / Device Info Sample
    - HTML elements reference - HTML | MDN
    - BreakDOM
    - Mozilla Annual Report 2018
    - Only works offline: https://chris.bolin.co/offline/

Week 1 Homework

- Reading:
    - DevTools: Inspect Styles
    - DevTools: Using the Console
- Doing:
    - DevTools CSS tutorial
    - DevTools JavaScript Debugging Tutorial
    - Find an interesting web page, chrome extension
- Making:
    - Create a “responsive” web site (or codepen). Use one of the new web APIs that we discussed in class (battery, online/offline, location, DeviceMotion, window size, foreground detection). Look also at the MDN Event Reference to see other events your page could respond to.
- **Homework submission form**

“Responsive” is a keyword in web design today. It usually means that a web site looks/acts differently when viewed on a different screen size (especially mobile browser screen size).

Create a “responsive” web page or codepen that responds to some **unexpected** aspect of the browsing experience. You can build on one of the demos we worked on in class or come up with something else. Possible examples are a page whose display changes based on battery level, or distance moved since opening the page, or proximity to some physical location.

Week 2: Bookmarklets

- Review
    - JavaScript, CSS, DOM API
- JavaScript in the Address Bar

- What's a bookmarklet?
    - How does it work?
    - What can it do?
    - What *can't* it do?
- Examining the DOM, interacting with text
- In-class: Building bookmarklets

Week 2 Slides

Week 2 Links

Useful links relevant to class or shown in class:

- Example of responsive site:
    - Offline Only
- Tutorials
    - Bookmarklets overview notes by Casey Watts
    - Bookmarklet Tutorial from BetterExplained
- Creating bookmarklets:
    - Bookmarklet Snippets on the class wiki
    - Bookmarklet creator: bookmarklet.hackingthebrowser.com
- Debugging and Things that can go wrong with Bookmarklets:
    - Mixed-Content Warning (loading http from an https site)
    - Content Security Policy
- Reverse-engineering:
    - JS NICE: Statistical renaming, Type inference and Deobfuscation
    - URL Encoder/Decoder
- Bookmarklet examples
    - 100 bookmarklets — Not all work, some load scripts over http
    - Jesse's bookmarklets — These all mostly work, they are usually pure JS (no script)
    - http only (no https):
        - fontbomb (can be fixed to be https)
        - katamari hack
    - kickass
    - Enable Paste
- Other
    - javascript in browser bar: 822745 - document.title not correctly set with JavaScript in omnibox - chromium - Monorail
    - How to Use Bookmarklets in Chrome for Mobile
    - The Simple, Serendipitous Joy of Browser Extensions - The New York Times
    - Intent to Deprecate and Remove: Top-frame navigations to data URLs - Google Groups

    - Red Dot Data URL
    - MDN Data URL docs
    - Make a notepad: data:text/html,<body contenteditable="true"><h1>Hi</h1></body>
- Discussed in class
    - Airpods Website: http://airpods-pro.webflow.io/ https://lqez.github.io/js/airpodsvf/
- In-Class Exercises
    - Here are some samples of how you might have constructed your bookmarklets during the in-class group work.

Week 2 Presentations

-

Week 2 Homework

1. Make a bookmarklet that works *against* the user. Be clever, be devious. It might make links harder to see/click (change color, move them around), remove form elements, put image/div on top of page, etc, make it less informative/useful. Interpret this how you like. It can be a bookmarklet that works generically on (most) sites, or one that is tailored for a specific site.
    a. Ideas for brainstorming:
        i. Think about the events that you can add listeners for
        ii. Think about how the techniques covered in the class group work for modifying the DOM
        iii. Watch this JS Conf presentation by Feross Aboukhadijeh.
2. **Optional**: Find an interesting bookmarklet in the wild from the lists linked to in this syllabus, or find your own. Reverse-engineer it using the techniques discussed in class and write up a short description of how you think it works. See the section below for a description of how to reverse engineer a bookmarklet.
3. Readings (not optional)
    a. The DOM and Events chapters of Eloquent JavaScript

Week 2 Homework Submission Form

Submit the homework using this **Homework Submission Form**.

How to Reverse Engineer a Bookmarklet

Bookmarklets, like web pages, cannot help but reveal their source code, so, like web pages, they are a great learning tool. See an interesting bookmarklet and wonder how it was done? Go investigate it yourself!

1. Get the source: Control-click on the Bookmarklet link and select “Copy link address…” from the context menu.
2. Paste the result into the URL Encoder/Decoder, *delete the* “javascript:” at the beginning, and click “Decode”. This will fix any percent-encoded values like “%20” into their JavaScript equivalents.
3. The code may still be “obfuscated” or “uglified”. If needed, paste the decoded result from the previous step into a tool like JS Nice / de4js and use it to “prettify” the “uglified” (yes, these are real terms that are used) code.

Now, use your JavaScript sleuthing skills to read through the code and see if you can figure out what it does. Remember, you can copy and paste sections of it into the console if you need to. Of special note to look out for is some code that injects a script (look out for “document.createElement(‘script’)” and some following code that sets the “src” attribute of that script element). If you see that, look at the source code for that script as well. You may need to “prettify” it.

Publishing Bookmarklets

To publish a bookmarklet so that others can install it on their own browser, you need to create a web page (or blog post, etc. — anywhere that you can edit the HTML) with a link for your bookmarklet. The link must have contents of your bookmarklet (the JavaScript code) as its “href” attribute. For example, if your bookmarklet’s JS code is `alert('hello, world')`, the link would look something like this:

```
<a href="javascript:function(){ alert('hello, world'; )}()">My Bookmarklet</a>
```

Note that because this is HTML, if you use double-quote characters inside your JS code they can interfere with the double-quotes surrounding the contents of the “href” attribute. Other characters, like “>” and “<” can also mess up your HTML. For this reason your JavaScript should be encoded properly. You can do this yourself using the “encodeURI” JavaScript function, or you can use the bookmarklet creator, as described in the following paragraph.

In order to simplify this process, you can use the Bookmarklet Creator Website to get the snippet of HTML that should go on your website. After you’ve used the bookmarklet creator to

make your bookmarklet, look at the bottom-right of the page for the “Publish your bookmarklet” section and copy/paste the HTML code there.

Week 3: Chrome Extensions

- What capabilities does a Chrome extension have that a bookmarklet does not?
- Student Presentations
- Creating Chrome extensions Overview
- Chrome extensions components:
    - manifest.json
    - browser_action
    - content_scripts
    - background
- Chrome capabilities overview:
    - “persistent” bookmarklet via content scripts
    - browser action icon
    - popup
- In-class examples:
    - turn a bookmarklet into a chrome extension

Week 3 Student Presentations

-

Week 3 Slides

Week 3 Links

- Chrome Developer Documentation
    - Manifest.json docs
    - Content Scripts (linked to from Manifest.json docs)
    - Match Patterns (linked to from Content Scripts docs)
    - Chrome Extensions Getting Started and Overview and debugging
    - Chrome Extensions API Index (docs on the chrome.tabs.executeScript API can be found under the “tabs” section)
    - Background Pages
- View your extensions at: chrome://extensions
- Sample Extensions That We Built in Class
    - Hello World with a Content Script
    - Use click listener on browser action to inject content script
    - Other class example extensions on github

- Other
    - Security: Google's XSS Game
    - Cross Origin Request Blocking (sometimes applies to bookmarklets)
    - "500 Miles" bug
    - Async vs Sync:
        - This "Loupe" tool, and the linked presentation, help explain the JavaScript event loop and why async-vs-sync matters with JavaScript: http://latentflip.com/loupe/
- Extensions Tutorials
    - Thoughtbot
    - Tutspus
    - Creating my first Chrome Extension
- Chrome Extension Source Viewer

Week 3 Homework

1. **Create**
    a. Turn your bookmarklet from last week into an extension. You can zip it up or put the code on github or some other web site of your choosing. *Or*, if you are inspired by the new Chrome Extension APIs you learned about and want to create an extension that uses one of those instead, do that!
2. **Readings**
    a. Complete the Chrome Extension debugging tutorial.
    b. Review either this Extension Tutorial or this one (or both)
3. **Reverse-Engineer**
    a. Look through the class extension examples — Find one that seems interesting and look at the code. Try running it locally. Find an API that it uses that we didn't yet discuss in class.

Week 3 Homework submissions

Use this **homework submission form**.

Week 4: Chrome Extensions Continued

- Reverse Engineering Chrome Extensions
- Content - Background script communication
- Chrome API docs review

Week 4 Slides

Week 4 Links

- List of Chrome URLS: chrome://chrome-urls/
- Class Presentations:
  - DevTools and the Debugger
  - Chrome Special Pages
- Chino's code:
  - Chino's Tumblr of 100 days of Chrome Extensions
  - Code for the extensions on github
  - MutationObserver API
- Extensions we built in class
  - Convert "Hello World" content script extension into one that is activated when you click on browserAction:
    - Original
    - Converted to use browserAction.onClicked
  - Convert the extension ^ to pass messages from background script to content script:
    - "on/off" extension
- Mentioned In Class
  - Shiffman Chrome Extensions
    - p5 Content Script
  - Solutions to "Broken Examples" puzzles:
    - Fix Example A
    - Fix Example B
    - Fix Example C
    - Fix Example D
- Class Presentations
  - CSS Animations
  - Chrome Extension Docs Presentation: Links:
    - An Overview of Chrome Extensions: https://developer.chrome.com/extensions/overview
    - An Overview of APIs for Chrome Extensions: https://developer.chrome.com/extensions/api_index
    - Since every Chrome Extension must have a manifest.json file, this is a good page to know: https://developer.chrome.com/extensions/manifest
    - Debugging Tutorial for Chrome Extensions:

https://developer.chrome.com/apps/tut_debugging
    - A Guide of Options to Develop Extensions: https://developer.chrome.com/extensions/devguide

Week 4 Homework

1. **Create an extension that uses a Chrome Extension API that you haven't used yet**
    a. Suggestions:
        i. alarms
        ii. tabs.* (onCreated, onUpdated, create, etc.)
        iii. browserAction.setBadgeText
        iv. or choose your own!
    b. Either
        i. create your extension from scratch, or
        ii. look through the class example extensions or Chrome Official sample extensions (beware: not all of them work) and pick one to use as a starting point
2. Reverse-Engineering: Use the Chrome Extension Source Viewer to view the source of an extension that you have installed on your browser
    a. Look at its manifest.json file and find something that we didn't cover in class but that you're curious about.
    b. Use the manifest.json docs to research it. What does the property do? Why is it there?
    c. Analyze the code — how do you think it works? Does it use content scripts, a background script, both or neither?
    d. What permissions does it require?
3. Complete the Chrome Devtools Network panel tutorial
4. Readings for next class. You do not need to read these exhaustively, but skim them to get a sense for how HTTP works.
    a. Understanding HTTP Basics, HTTP Overview, Intro to HTTP, MDN HTTP Headers

Week 4 Homework Submissions

The submission form is **here**.

Week 5: Web Request Blocking, Distribution

- Distributing Your Chrome Extensions
    - Packaging
    - Chrome Developer Store
- Web Request modifications

- Final Project review/discussion
- Class small group work

Week 5 Slides

Week 5 Presentations

-

Week 5 Links

- Chrome Extension Distribution:
  - Chrome Developer Dashboard
  - Chrome Docs: Distribution Options
  - Week 5 Slides about Distribution
- Chrome Extension Examples Shown or Mentioned in Class:
  - Web Request Kitchen Sink — All APIs mentioned in class are used here, with detailed comments.
  - ad blocker — blocks scripts it thinks are ads
  - Google en español — make 'accept-language' request header 'es' so Google replies in spanish
  - Unloginable — removes 'set-cookie' response header so you can't log in anywhere
  - Request Blocker — multiple examples of blocking
- Group Work:
  - Chrome Extension Puzzles
- Mentioned in Class
  - Browser Fingerprinting
  - Evercookie
  - Chrome ScrollToTextFragment deep-linking and privacy concerns

Week 5 Homework

Create a "mini final project" Chrome Extension. Either extend and polish the one last week, or create a new one.

Week 5 Homework Submission form

Use this form to submit homework.

Week 6: Special Topics

- Chrome security discussion
- Realtime Communication with Firebase
- Browser discussion

Week 6 Links

- Mentioned in class:
    - WAT Talk about JS: https://www.destroyallsoftware.com/talks/wat

Week 6 Slides

Week 6 Presentations