Programming web-based educational media

Matthew X. Curinga

Antonios Saravanos

Table of Contents

**EDT 603 *Programming web-based educational media*, Spring 2019**

<!DOCTYPE html>  
<html>  
 <body>  
 <blockquote>  
 Anyone who has lost track of time when  
 using a computer knows the propensity  
 to dream, the urge to make dreams come  
 true and the tendency to miss  
 lunch.<br>  
 <strong>Tim Berners-Lee</strong>  
 <em>, inventor of the world wide web</em>  
 </blockquote>  
 </body>  
</html>

**Description:** In this course students learn techniques of web programming to develop interactive, educational media. Using the Javascript programming language and related web development technologies (HTML5, CSS, SVG), students gain practice in the programming and design of interactive software.

**Key words:** computer science, web development, mobile web, interaction design, html, html5, css, javascript, OOP, mobile first

# Office Hours

**Matt Curinga, Alumnae Hall, Room 226A**

* Monday, 11-1:00PM
* Tuesday, 2:30-4:30PM
* Thursday, 3-5PM
* *office hours by appointment*

# Goals and Objectives

This course builds on CSC 602 to move beyond basic programming concepts; students will gain expertise in building more complex computer programs, over several iterations. At the end of the course, students will be able to design educationally sound web-based learning media, solve moderately complex problems using object oriented and functional programming paradigms, and collaborate on team programming projects.

Specific teaching and learning goals include:

* designing web-based interactions and multimedia to support learning
* coding effective user interfaces for learning
* implementing Universal Design goals for accessible web sites
* identifying effective methods for teaching more advanced programming concepts and web design skills

Specific software development goals include:

* modeling real world problems with software
* iterative software development
* testing and debugging
* Object oriented programming concepts:
  + Abstraction
  + Encapsulation
  + Objects & Classes
  + Composition
  + Inheritance
  + Polymorphism

# Course textbook

Curinga, M. Peter Wentworth, P., Elkner, J., Downey, A, and Meyers, C. (2018). [Think Javascript](https://mcuringa.github.io/think-js/). [free open textbook]

# Online Documentation

* [Mozilla Developer Network Javascript Docs](https://developer.mozilla.org/en-US/docs/Web/javascript)
* [Mozilla Developer Network HTML Docs](https://developer.mozilla.org/en-US/docs/Web/HTML)
* [Mozilla Developer Network CSS Docs](https://developer.mozilla.org/en-US/docs/Web/CSS)
* [React Framework](https://reactjs.org/docs/getting-started.html)
* [Bootstrap](https://getbootstrap.com/docs/4.2/getting-started/introduction/)

# Required Software/Online Accounts

* Software
  + [Slack](http://slack.com) (recommend desktop and mobile clients)
  + [Firefox web browser](https://www.mozilla.org/en-US/firefox/new/)
  + Chrome or [Chromium](https://www.chromium.org/Home) web browser
* Accounts
  + [Repl.it](https://repl.it)
  + [AU Ed Tech #code](https://auedtech.slack.com/signup)

# Recommended Books

*Not required, but a good book for the basics of HTML and CSS*

Duckett, J. T. (2011). [*Html & css: design and build websites*](http://htmlandcssbook.com/). Indianapolis, IN: Wiley Pubishing, Inc.



# Schedule

|  |  |  |
| --- | --- | --- |
| Module | Topic | Due |
| 0 | Preparing for the class |  |
| 1 | Structure, data, style, logic |  |
| 2 | Mobile First, Styles, & Bootstrap |  |
| 3 | Advanced CSS | *Resume* Due |
| 4 | Objects and React State | *Mini App* Pitch |
| 5 | Forms and storage |  |
| 6 | Routes & Navigation |  |
| 7 | *Studio* |  |
| 8 | Mini App UX Testing | *Mini App* Due |
| 9 | 1:1 meetings | *self-eval due* |
| 10 | Async calls & remote data | *App* Pitch |
| 11 | REST APIs and CRUD |  |
| 12 | User authorization |  |
| 13 | Client-side interaction |  |
| 14 | *Studio* |  |
| 15 | App Reviews | *Final App* & Presentation |

# Assignments & Grading

|  |  |
| --- | --- |
| Assignment | Points |
| Multimedia Resume | 30 |
| Mini App | 30 |
| Self Evaluation | 10 |
| Web Application | 30 |

### Self-evaluation (10 points)

At the beginning of module 9, you will complete a self-evaluation. During your one-on-one meeting with the instructor, you will discuss your self-evaluation in order to make sure you get the most out of the remainder of the class.

### Software Project Evaluation Rubric

*This marking guide will be used to evaluate the three software project* *assignments required for this course.*

1. **React/Javascript (10 points)**
   * 1-3 points: the code runs with errors, is incomplete, or a very close copy of the example project
   * 4-7 points: code is organized into functions and uses parameters, code is well organized and well styled, can be improved by writing more general/reusable functions and parameters, being more flexible, or using Javascript idioms efficiently and correctly
   * 8-10 points: code is well thought out and reusable functions create core parts of the site, functions are organized so that changes and new features can be easily implemented, code meets our style guides and clear/accurate names are given to all identifiers (variables, functions). It is clear that the program goes beyond the example project.
2. **Data modeling (5 points)**
   * 1-2 points: most content is hard-coded in the Javascript code, model is an exact copy of example, and/or model does not support the goals of the site
   * 3-4 points: data model supports the goals of the specific site, but may not be flexible enough for a different presentation or to handle new data
   * 5 points: data model supports the site, and can support other uses without modification to the model, new content can be easily accommodated
3. **HTML (5 points)**
   * 1-2 points: content is presented with little structure, or different HTML tags would better describe the content
   * 3-4 points: there is a good fit between the tags used and the data they contain
   * 5 points: there is a good fit between the tags used and the data they contain *and* the code correctly and appropriately uses tags that were not part of the example code
4. **CSS (5 points)**
   * 1-2 points: little or no styles, styles are defined that are not used or do not take effect because of errors
   * 3-4 points: code demonstrates understanding of box-model, color, typography, and/or images
   * 5 points: advanced layout are used to interesting effect (flex, grids, etc), code exceeds example projects
5. **User Experience (5 points)**
   * 1-2 points: site is hard to read, understand, and use; labels and text are not well edited, items are not clearly organized, essential information is missing, and/or the design significantly impedes the usability of the site
   * 3-4 points: site is easy to use and information is well organized, presentation is clear and clean
   * 5 points: code, layout, and css combine to create an interesting effect, site has a high quality, professional feel

### Online Resume (30 points)

This first assignment is designed to get you up and running with the key technologies we will use this semester. You will create a data-driven web page as your own online resume or portfolio.

For this project, you will:

1. Keep all of the *data* for your resume in .json data files. These files will contain the lists of your work experience, education background, technical skills, and other relevant information.
2. Code Javascript source files using the React framework and HTML5 to structure your web page.
3. Create or find any media assets (images, audio, video, etc) necessary for your resume.
4. Code CSS files to achieve your desired aesthetic and usability goals

The portfolio site will be hosted on repl.it and the link to your final project will be submitted through the course website.

Your project will be assessed both on the quality of your code and the success of the project, using the following criteria:

*The resume is an individual project*

### Mini-app (30 points)

The mini app is a data driven web application. It is “mini” because it has a constrained, singular focus. This application stands alone – it does not need any networked resources. It goes beyond the resume project, though, in that it’s *interactive*. Buttons, text boxes, and other form elements enable the user to alter data and the way the app functions. Data created is either stored locally (in the user’s client) or only exists during the session.

Successful projects will use React *state* and React *life cycle functions* to achieve their results.

*This mini-app is an individual project*

### Web application (30 points)

For the final project, you will work in a team to create a fully functional web application. You will learn how to save data in a remote data store and how to handle user authentication and authorization to create a secure, multi-user app.

*This web application is a group project. All team members will receive the same grade.*

# Books and online resources

### Documentation & Reference websites

* [Mozilla Developer Network](https://developer.mozilla.org/en-US/)
* [World Wide Web Consortium](http://w3.org)
* [W3 Schools](http://www.w3schools.com/)
* [Regular Expressions](http://www.regexr.com/)

### Books

* [HTML and CSS: Design and Build Websites](http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118008189.html), our textbook
* [JavaScript & jQuery: Interactive Front-End Web Development Hardcover](http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118871650.html), also J. Duckett, same series
* [Dive into HTML 5](http://diveintohtml5.info/) [free online]
* [The Elements of Typographic Style Applied to the Web](http://webtypography.net/toc/) [free online]
* [Mastering Regular Expressions](http://shop.oreilly.com/product/9780596528126.do)

### Tutorial websites & online learning

* [Code Academcy](http://www.codecademy.com/)
* [P2PU School of webcraft](https://p2pu.org/en/schools/school-of-webcraft/)
* [Treehouse](http://teamtreehouse.com/) [paid]
* [Thinkful](http://www.thinkful.com/)
* [GeekCamp::HTML5 Tutorial](http://www.geekchamp.com/html5-tutorials/1-html5-overview)
* [SkilledUp::Learn Web Design](http://www.skilledup.com/learn-web-design-guide/)

### Design, accessibility, UX

* [A List Apart](http://alistapart.com/topic/html)
* [Smashing Magazine](http://www.smashingmagazine.com/)
* [Adobe Kuler](https://color.adobe.com/create/color-wheel/)
* [Nielsen/Norman Group](http://www.nngroup.com/articles/)
* [United States Section 508](http://en.wikipedia.org/wiki/Section_508_Amendment_to_the_Rehabilitation_Act_of_1973)
  + <https://www.section508.gov/>
  + <http://webaim.org/standards/508/checklist>
* [Usability.gov](http://www.usability.gov/index.html)
* [Research-Based Web Design & Usability Guidelines](http://www.usability.gov/guidelines/guidelines_book.pdf)
* [hex/html color chart](http://www.december.com/html/spec/color.html)

### Online Tools

* [w3c HTML Validation Service](http://validator.w3.org/#validate_by_uri+with_options)
* [w3c CSS Validation Service](http://jigsaw.w3.org/css-validator/)

### Media Resources

* [Creative Commons Search](http://search.creativecommons.org/), for images, music, etc
* [Wikimedia Commons](http://commons.wikimedia.org/wiki/Main_Page), images and other media (including stuff from Wikipedia), curated
* [Open Clip Art](https://openclipart.org/), free vector graphics
* [Creative Commons Music](http://creativecommons.org/music-communities)
* [Fossil Bank](http://fossilbank.wikidot.com/)
* [Colour Lovers Palettes](http://www.colourlovers.com/)
* [Google Fonts](https://fonts.google.com/)