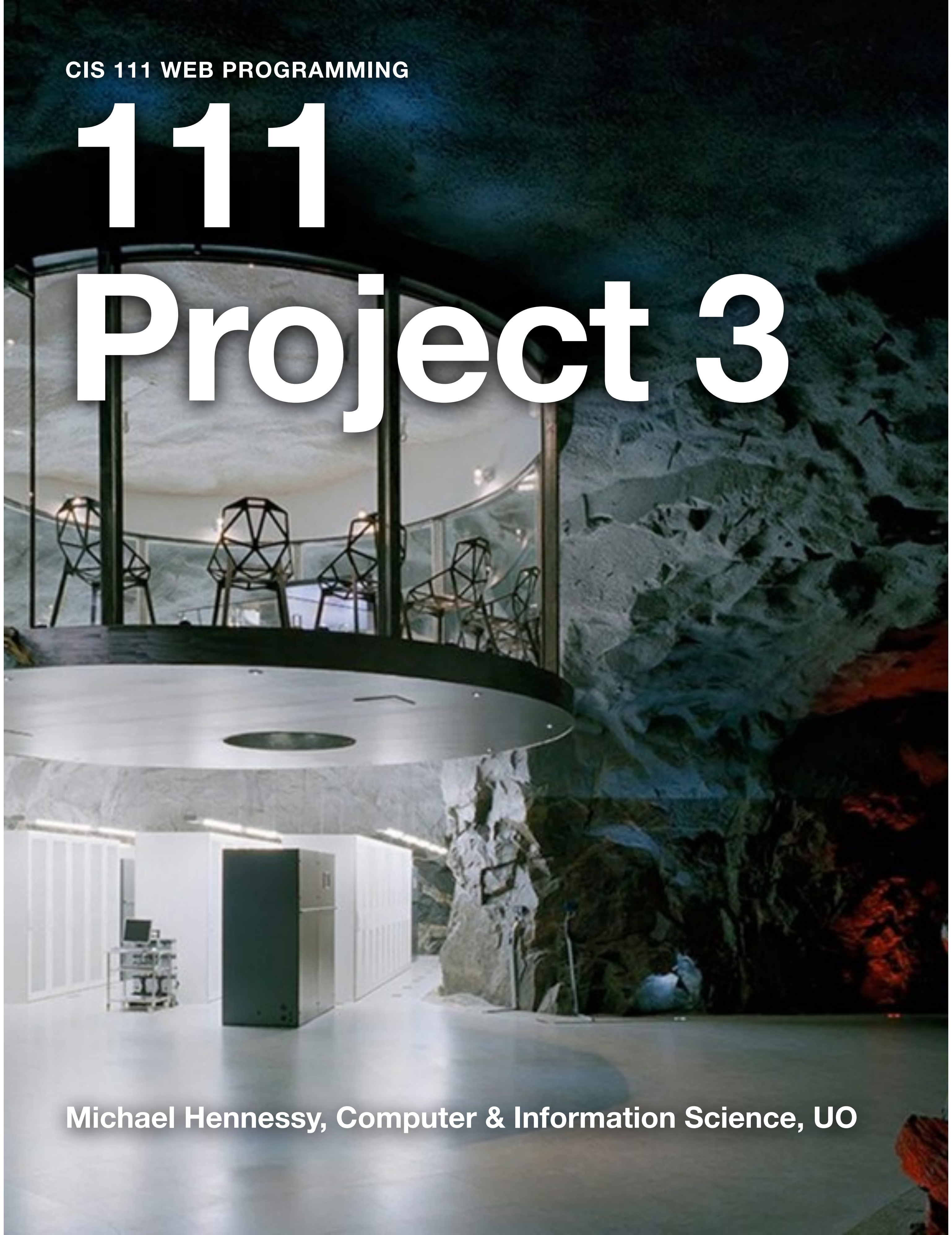
CIS 111 WEB PROGRAMMING

111 Project 3



Michael Hennessy, Computer & Information Science, UO

CIS 111 Project 2

Programming JavaScript

Due 1700 Mon 7/14.

In this project you will solve problems using functions, selection statements, loops and arrays.

Let's get started..

Project Requirements

Due: 1700 Mon 7/14.

PROJECT REQUIREMENTS

For the most efficient use of your time, solve the following problems in the order shown.

All the exercises on this project will be covered in class, with all your questions answered.

1. **[20 pts] 111/js/timelyMsg.js and timelyMsg.html.**

   P. 20 in JumpStart JavaScript shows the logic to generate a time-dependent greeting. Use this flowchart to write a value-returning function named timelyMsg, that does not accept an argument, and uses the JavaScript Date object to return a message as shown.

   (A) Command-line JavaScript: Use the DevTools JavaScript Console to test your function.

   For example, if you execute this function call at 1:14pm, the message, Good afternoon! will be displayed:

   timelyMsg() => Good afternoon!

   Store the function definition in 111/js/timelyMsg.js.

   (B) Client-Side JavaScript: Create a simple web page named timelyMsg.html, and store it in your 111/p3/ folder.

   The web page should contain an h2 header, with content Timely Messenger.

   The web page should contain a click button, labeled Display Message.

   The web page should contain no JavaScript statments. All JavaScript is contained in timelyMsg.js. Connect timelyMsg.html to timelyMsg.js.

   When the button is clicked, timelyMsg is called, and the returned value is inserted into a h3 element on the web page.

2. **[20 pts] 111/p3/numToWord.html, and 111/js/numToWord.js.**

   Write a function named toWord that accepts a number between 1 and 4 inclusive, and returns a word representation of the number.

   Examples:

   toWord(2) => "two"
   toWord(1) => "one"
   toWord(7) => "too large"
   toWord(0) => "too small"

   (A) Command-line JavaScript: Use the DevTools JavaScript Console to test your function. Run the tests shown above.

   Store the function definition in 111/js/numToWord.js.

   (B) Client-Side JavaScript: Create a simple web page named numToWord.html, and store it in your 111/p3/ folder.

   The web page should contain an h2 header, with content Number to Words Translator.

   The web page should contain a click button, labeled Display Word.

   The web page should contain no JavaScript statments. All JavaScript is contained in numToWord.js. Connect timelyMsg.html to numToWord.js.

   When the button is clicked, toWord is called, and the returned value is inserted into a h3 element on the web page.

3. **[20 pts] Dice Rollers, Part I.**

   Client-Side JavaScript: Follow the directions on Loops in JavaScript to create roll-v1.html and roll-v1.js; roll-v2.html and roll-v2.js; roll-v3.html and roll-v3.js. Store the .html files in 111/p3/. Store the .js files in 111/js/.

4. **[20 pts] Dice Rollers, Part II.**

   Client-Side JavaScript: Follow the directions on Loops in JavaScript to create roll-v4.html and roll-v4.js; roll-v5.html and roll-v5.js; roll-v6.html and roll-v6.js. Store the .html files in 111/p3/. Store the .js files in 111/js/.

5. **[20 pts] 111/js/array-functions.js.**

   (A) Command-Line JavaScript. Write a function named arrAvg that accepts an array of numbers and returns their average.

Use the DevTools JavaScript Console to test the function. Store the function in 111/js/array-functions.js.

(B) Command-Line JavaScript. Write a function named arrMax that accepts an array of numbers and returns the largest element in the array. Use the DevTools JavaScript Console to test the function. Store the function in 111/js/array-functions.js.

(C) Command-Line JavaScript. Write a function named arrHasOneEven that accepts an array of numbers and returns true if the array contains at least one even number. Use the DevTools JavaScript Console to test the function. Store the function in 111/js/array-functions.js.

How to Turn In your Project

How to Turn In your Project

<u>All you Have to Do</u> is Make Sure your web pages are uploaded to the server by the Due-Date.

When your web pages are on the server, they can be graded.

You do not have to submit this project in Blackboard, nor do you have to notify your instructor in any way.

Just make sure you complete the project by the Due-Date, and do not upload or edit the files after the due-date. If you change the web page files in any way after the due-date, this will change the time-stamp of the files on the server, and your project will be late (zero points).

Project Grading Checkpoints

HOW YOUR PROJECTS WILL BE GRADED

- **There is no "submit your project in Blackboard" step for 111 projects.** When you have uploaded your web pages and tested them on the server, you are done turning in the project for grading.
- **The instructional staff has complete access to your project files for grading.** Your job is to make sure the files are on the server on time, and that you have tested them to make sure they are correct.
- **The files you upload to the server by the due-date are what will be graded**, so be sure to test your web pages on the server to make sure they are correct.
- **There are no second chances. Why? We do not have the time or the resources to grade your work twice.** Therefore make sure that what you upload to the server is correct. Test your web pages on the server after uploading them.
- **Time-Stamps are Crucial.** When you upload a file to the server, it is stamped with the exact time of the upload. This time-stamp must be no later than the project due-date. Your project is on-time only if the time-stamps show that it was uploaded to the server on time.
- **Do not re-upload any of your project files after the due-date.** If you do, this will change the time-stamp and your project will be late (0 pts).
- **Do not use Sublime's Sync feature, as this will change the time-stamp on all your files on the server.**
- **Your 111 folder on the server must be .htaccess password-protected.** If it is not, your project score will be zero (0). See your instructor or GTF for assistance if necessary.
- **Know the 111 Late Policy.**

Keeping these points in mind will help ensure you get full credit on your projects. Ask questions in class if anything is not clear.

Meeting the Deadline

How to Handle the 17:00 Deadline

- Start working on your project early.
- Friday Office and Help hours are jammed, and may end before you get assistance. Plan on completing all your projects before the deadline.
- Turn in what you have by the deadline-- partial credit is better than none.
- Piazza is good for answering verbal questions, but limited in terms of debugging help. For debugging, you need F2F help, which is the gold standard
- For Gold Standard Help:

  See *Contacts* in Blackboard.

  See *111 Help Hours* in Blackboard.

WebDev Workflow

Here is the CIS 111 Web Development Workflow:

1. **Edit**. Use the Sublime Text editor to create a web page (.html and .js files) on your computer.
2. **Preview**. Open the web page on your computer using Chrome. When it is perfect, and not before, go to the next step.
3. **Upload**. Move all project files (.html, .js, .png, etc.) to the server using an SFTP client (Sublime, CyberDuck, Aptana). This is also known as publishing the web page.
4. **Test**. Use Chrome to open your web page that is on the server. Do not use CyberDuck; do not use Aptana. Make sure that this web page is correct, because that is what will be graded.

Related Glossary Terms

Drag related terms here

Index Find Term