SYST10199Web Programming

Sheridan College

# Dr Abstract (Dan Zen)

**Office Hours:** video or audio chat can be scheduled.

**Email:** Please feel free to email me any time at [dan.zen@sheridancollege.ca](mailto:dan.zen@sheridancollege.ca). Please include your name and day of your class in all correspondence. Slate messages are checked just before class.

# Course Outline

The official course outline is available at <https://shorturl.at/fIL07>

# Course Description

This course is about building web apps such as information apps, games, simulations, and data visualizations. Web apps are computer programs that run over the web. In a web app (like Gmail or Facebook) the user interface is a web page retrieved from a server and displayed on a browser running on a “client” computer. But the data for the app is housed on a remote server and the processing is done partly on the client side and partly on the server side using programming languages specifically tailored for each purpose.

In this course you will build on the HTML and CSS skills you learned in your Web Development class. You will learn how to make web pages dynamic and responsive by writing programs in JavaScript that run in the user’s browser. You will create client-side web apps using JavaScript. You will learn how to write programs on the server to store and retrieve data.

This course will provide you with the resources and understanding to continue exploring web app development on your own, in your future courses and/or on the job.

# Software

A text editor with syntax highlighting like VS Code and a Web browser such as Chrome.

# Learning Materials

Most of the code is detailed in PowerPoints in lessons at <http://imm.sheridanc.on.ca/webprogramming/>

with reference to Mozilla ([https://developer.mozilla.org](https://developer.mozilla.org/)). Other resources such as w3Schools (<http://www.w3schools.com>), StackOverflow, etc. may be helpful.

# Course Content

## Module 1: Client Side Scripting

JavaScript basics: variables, functions, conditionals, loops, arrays, objects, events taught with coding on the Canvas. Document object model (DOM) and modifying HTML and CSS properties with JavaScript.

## Module 2: Forms

Client-side web apps using HTML Forms, Validation, ARIA, localStorage, and JSON.

## Module 3: Server Side Scripting

NodeJS – Node Package Manager, Express (server), storing and retrieving files, AJAX, sessions.

# Evaluation Plan

Note that in order to pass the course you must get a mark of 50% or more on the weighted combination of all evaluation elements, as well as on the weighted combination of the tests and exams.

Midterm Exam 25%

Final Exam 25%  
Checkoffs (4) 10%

Assignments (4) 40%

## The Exams

One to two hours long. The midterm covers module 1 and the final covers modules 2 and 3.

## The Assignments

There will be four assignments - two each term. There are four checkoffs due the week before the assignment in class - to look at the code you have been doing in class.

## Accommodations

If you are entitled to any accommodations, whether in the classroom, in course materials, on tests, or on assignments, please don’t hesitate to let the instructor know.

## AI

Do not use AI for coding in this class – we want you to learn what is going on. You learn by puzzling things out. Do not use AI for solving the puzzle. Plus, AI might insert things that you have not learned yet, or that are incorrect, etc. Teachers generally know when your code is not your own.

Calendar is on next page

# Calendar of Events

Please note that all dates are subject to CHANGE according to schedule in Slate.

**Module 1: Client Side Scripting**

**Module 2: Forms**

|  |  |  |
| --- | --- | --- |
| Week |  | Event |
| 1 | Canvas |  |
| 2 | Canvas |  |
| 3 | Canvas | Checkoff 1 (2.5%) |
| 4 | DOM | Assignment 1 Due (10%) |
| 5 | DOM | Checkoff 2 (2.5%) |
| 6 | Review | Assignment 2 Due (10%) |
| 7 |  | Midterm Exam (25%) |
| -------------------------- BREAK WEEK ------------------------- | | |
| 8  9 | Takeup HTML / JS | Checkoff 3 (2.5%)  **Module 3: Server Side Scripting** |
| 10  11 | NodeJS NodeJS | Assignment 3 Due (5%) |
| 12  13 | NodeJS Study Period | Checkoff 4 (2.5%)  Assignment 4 Due (10%) |
| 14 |  | Final Exam (25%) |

# Due Dates

Assignments are due in the drop box on the day shown in the calendar.

You can hand in late assignments with no penalty, but you have a limit of 8 late days for all assignments cumulative. If there are special or unforeseen circumstances (illnesses, family emergencies, etc.) let me know as soon as you can, and I will take that into account.

The last day of class is the final due date for everything in the course, regardless of how many late days you have left.

# Missed Evaluations

If you miss a test, quiz, or in-class activity for any reason, you should contact the teacher immediately to explain your absence and be prepared to produce a doctor’s note or other documentation. If a make-up test is allowed, the Registrar’s Office will charge you a fee to write it.

# Advice for Students

You learn computer programming by doing it. To do well in this course, you need to be actively engaged. This means coming to class on time, paying attention, taking notes, using class time to complete exercises and assignments, working on exercises and assignments outside of class time, and actively seeking help from the teacher, other students, on-line resources, and textbooks. If you miss a class for any reason, you should look at SLATE as soon as you can and contact the instructor and/or other students to find out what you missed.

# Working Together and Helping Each Other

Students are encouraged to work together and help each other with assignments. When one student helps another in a productive way, they both end up understanding it better. But some ways of helping are more productive than others, and some ways of “helping” are no help at all.

### Good ways to help…

* Talking things over with someone to help them understand a concept
* Helping someone find the information they need
* Testing another student’s program to look for mistakes
* Sitting with someone to advise them while they write or debug a program something they are having trouble with

### Bad ways to help…

* Writing a part of somebody’s code for them.
* Mailing somebody your program so they can use it as a template, cut and paste parts of it, or change it slightly and hand it in as their own.

### Plagiarism & Academic Dishonesty

Plagiarism happens whenever somebody takes credit for somebody else’s work. Plagiarism in a computer program is as easy to spot as if two students handed in the same essay, even if the program has been changed a bit to make it look different. Copying a program or part of a program from another student or from an online source is an act of plagiarism.

When plagiarism happens, all students involved are equally guilty, whether the plagiarism was intentional or not. The first time you are caught plagiarising you will receive a zero on the assignment and be reported to the faculty. Consequences for subsequent offences can be quite severe, possibly including withdrawal from the course or expulsion from the college. Protect yourself: don’t write somebody else’s program for them, don’t use code from on-line sources, and never send anybody your own program to look at.

Please refer to Sheridan’s policy on Academic Honesty for more information: <https://policy.sheridanc.on.ca/dotNet/documents/default.aspx?docid=673&mode=view>