

JavaScript Rocks!



Syllabus

JavaScript Rocks! is a great introduction to programming. This course will give you the skills for building interactive web pages. And, you will learn essential skills used by professional programmers.

This course uses up-to-date teaching techniques allowing you to learn by reading, listening, experimenting, and writing your own code. Each module has specific learning activities designed to help you learn in an interesting and fun manner.

JavaScript is one of the top 10 programming languages in demand today. Even better, knowing JavaScript will give you the edge on learning other languages such as C#, Java, PHP, and Perl.

Course Details

| | High School | South Central College |
|--------------------------|--|-------------------------------------|
| Course Title | JavaScript Rocks! | COMP1130 - Programming Fundamentals |
| Credits | .5 high school credits | 4 college credits |
| Description | This course is an introduction to programming using JavaScript. It covers all the essential programming concepts including variables, data types, control statements, debugging, loops, functions, arrays, and frameworks such as jQuery. The learning activities and project work emphasize time management, problem solving, and professional communications. | |
| Prerequisite: | It is recommended that you have a minimum typing speed (touch typing) of at least 20 wpm. A basic knowledge of HTML and CSS will be helpful. This free site will get you started: https://www.codecademy.com/learn/web | |
| Format: | Online http://www.SocratesOnline.org/ | |
| Weekly work time: | 15-20 hours (approximate) | |

Textbooks are optional but highly recommended:

- **JavaScript by Example**, Second Edition by Ellie Quigley Prentice Hall 978-013-70548-93
This book has lots of great explanations and code examples.
- **A Smarter Way to Learn JavaScript**, by Mark Myers, Amazon Digital Services.
Mark Myers has come up with a new form of text book. The book itself gives very basic explanations to give your brain an overview. Then the learning starts as you start working with the the hundreds of simple examples. If you like to learn-by-doing than this is the "book" for you. Download the free Kindle reader for your computer or tablet.

Required Software:

- **A text editor with text color-coding**
 - Windows: [NotePad++](#)
 - Mac: [Text Wrangler](#)
 - Windows, Mac, and Linux: [Atom Text editor](#).
- **Web Browsers**
 - [Google Chrome](#)
 - [FireFox](#)
- **Register at <http://www.TypingTest.com/>** for the weekly keyboard challenges.

Quick Reference Charts

- [JavaScript Cheat Sheet](#) - A 1-page summary with functions and keywords.
- [HTML5 Element Summary](#) - A 1-page summary list of the most popular elements.
- [HTML5 Element - Complete List](#) - A 4-page summary of all the HTML5 elements from Smashing Magazine.
- [CSS3 Summary Sheet](#) - A 5-page chart from Smashing Magazine

Course Goals

1. **Think like a programmer.** Algorithmic thinking: being able to read and write a formal language.
2. **Solve problems.** Learning how to communicate complex ideas and how to break down problems into logical pieces.
3. **Communicate as a professional.** Speaking with precision, writing in a professional manner, and being able to listen.
4. **Demonstrate successful time management.** Complete projects early or on time based on client specifications.
5. **Improve your typing speed** by 50% or more by the end of the course.

Student Learning Outcomes

Write algorithms and following an organized design process.

- Define an algorithm.
- Write an algorithm in English describing the steps a program follows.
- Describe the strengths and weaknesses of different design processes.
- Demonstrate the difference between how our minds work and how a computer program works.

Create a web page and view the file in a browser.

- Create an HTML document using the essential HTML commands in the correct order.
- Display the web page in a browser without accessing the Internet.
- Incorporate effective commenting as part of the web page code.
- Incorporate a header comment block to self-document every page of code.
- Create an HTML template to speed up web page development.
- Utilize at least three different heading tags in a web page to establish an information hierarchy.
- Establish a workflow developing HTML code, alternating between editing the HTML code and viewing the results in a browser.

Use variables to remember things for the browser.

- Use at least three different data types in a program.
- Demonstrate the difference between declaring a variable and initializing it.

Use debugging techniques stepping through a program and observing the objects and their values.

- Write special print statements to display contents of variables.
- Use selective commenting to determine where bugs are located in code.
- Use developer tools set breakpoints, step through a program, and view the variables as they change.

Write functions that the browser can use to do things

- Declare a function in the <head> element of a web page.
- Call the function from the <body> element of the web page.
- Write a function uses parameters to customize the results.
- Call a function passing parameters into it.
- Write a function that returns information to the calling line of code.
- Call a function that returns information, capturing it in a variable using the assignment operator "=".

Utilize operators and expressions.

- Describe the difference between equality operator "==" and the assignment operator "="
- Demonstrate use of the relational operators
- Demonstrate an overloaded operator such as "+" showing when it concatenates and when it will do addition.

Demonstrate the use of control statements.

- Make decisions in your program using an **if/else** statement
- Demonstrate the proper use of the **switch** statement in a program.
- Demonstrate use of a counting loop using a **for** loop in a program.
- Demonstrate use of an indefinite loop using a **while** loop in a program.
- Demonstrate use of the **foreach** loop in a program.
- Demonstrate the **scope** of variables both local and global.

Utilize Arrays to keep track of lists of things

- Explore data structures by building a list of things.
- Use a for loop to display the contents of any single-dimensional array
- Utilize the built-in mutator methods to dynamically change the contents of an array.
- Declare and populate a two-dimensional array.
- Display the contents of a two-dimensional array using the HTML <table> element and nested for loops.
- Use parallel arrays.

Incorporate Events and Event Handling

- **onclick()** - Write an event handler that responds to user input such as a tap or click.
- **onfocus()** - Write an event handler that validates user input as the user tabs out of a text box.
- **onblur()** - Write an event handler that runs as soon as the user tabs into a specific form element such as a text box.
- Use the **onLoad** event to run JavaScript code only after the entire page has loaded into the browser's memory.
- Describe the difference between loop processing and event processing.

Demonstrate the Prototype/Object model used in JavaScript.

- Create a JavaScript object having properties and functions.
- Utilize the JavaScript object in a program using the keyword "new."
- Demonstrate use of **String objects** in a program.
- Demonstrate at least five string methods to parse data input by the user.
- Demonstrate the use of **Math objects** in a program.
- Create random numbers within a certain range
- Demonstrate the use of **Date/Time objects** in a program.

Demonstrate creating interactive forms using JavaScript.

- Create a web page that includes all the HTML form elements.
- Validate user input for a form textbox.

Demonstrate using the built-in DOM functions.

- Use the DOM function `getElementById()` to locate any element on the page by its unique id attribute.
- Build an array of similar objects on a page using the DOM function `getElementsByTagname()`.
- Create a hyperlink on a page using DOM functions.
- Change the attribute of a HTML element using the DOM.

Utilize a JavaScript library such as jQuery.

- Decide which process to use, local library file or a CDN link.
- Use a library function to determine if the DOM is completely loaded in memory.
- Use a library function to select any element on the page by tag or id.
- Handle events using library functions and callbacks.
- Animate objects on the page using library functions.

Expectations

1. **You are responsible for your own actions** (and inactions).
2. **Do the work.** It is expected that you have done all of the Learning Activities. Be smart and set up a regular study schedule as if you were attending a face-to-face class. The learning activities are presented in a progressive order. Complete them in order because they build on each other. (Please, do not try to complete this course by trying to learn everything in a single day or two.)
3. **Meet the deadlines.** Use an application such as Asana (<https://asana.com>) to remember what needs to be done and when.
4. **Understand that learning is moving out of your comfort zone.** Learning is about change and change means being willing to move into new areas you haven't explored before both emotionally and technically.
5. **Communicate as a professional.** In the professional world you are judged by your emails.
 - Always write using proper grammar and spelling. Use an app such as Grammarly (<https://www.grammarly.com>) is highly recommended.
 - Use paragraphs to make your writing more readable.
 - Format your content using headlines, bold text, and hyperlinks.
 - When sending emails, be smart and always include a copy of your code (as a zip file) as well as any screenshots or graphics that will clarify what you need.
6. **Stay connected.** Have all your csp.edu emails forwarded to your personal email address and check your email at least once each day. Install the Slack app on your smartphone and monitor the communications especially in the #general channel.

Course Policies

Deadlines

Projects: Build on the bonus points by turning in projects early. There is also a 48-hour window (with a 20% penalty) after the due date. After that no projects are accepted.

Quizzes: You have one week to take each graded quiz. Miss that huge window of opportunity and you have lost out on those points. There will be no makeup opportunities at the end of the semester.

Due to the time-relationship between the self-quizzes (as learning activities) and the graded-quizzes (as assessment activities) there will be no graded quizzes allowed after the established deadline.

Cheating

All graded projects must be your own work. Cheating or plagiarism is a serious breach of academic ethics and could lead to sanctions including expulsion from college. When taking tests, protect your answers from others. If cheating is discovered, all participants will be penalized. Don't be a victim. If someone asks for help on a graded project just say "no" or offer to help them using the learning activities that are not graded.

Special Needs

If you have a disability and need accommodations to participate in the course activities, please contact the academic support services person for your school.

The materials in this course including videos, tutorials, and other learning tools have been written to be accessible to all students.

The Bottom Line

I am here to help you learn and understand the material presented in this course. Please let me know if you do not understand any concepts or projects in this class. Feel free to clarify any questions you may have during class, at my office, or via email. I am also open to any suggestions you may have that would make this course more interesting and/or useful.



Your Instructor

Peter K. Johnson - pjohnson@mnSCSC.org

Website: <http://PeterKJohnson.com>

Mobile: 507-317-3572

Office Hours: 8 a.m.–5 p.m. Monday–Friday. Other meetings by request.