**Coding for GUIs (JavaScript edition)**

**LESSON 1: Structure, style and function**

Please refer to the online lesson plan on the DT Hub to access all website links and additional resources.

**Year levels**: **7–8**

This is the first in a series of lessons to incorporate Graphical User Interfaces (GUIs) into your General Purpose Programming. It follows on from the Visual To Text Coding lesson series.

Included videos can be used by a beginner teacher and/or students to see how to code each of the simple programs step-by-step with HTML, CSS and JavaScript.

This lesson may take two to three 45-minute periods. It introduces the way HTML joins together with CSS and JavaScript to make GUIs possible.

# **Curriculum links**

Links with Digital Technologies Curriculum Area

| **Strand** | **Year** | **Content Description** |
| --- | --- | --- |
| Processes and Production Skills | Year 7-8 | Design the user experience of a digital system (AC9TDI8P07) Generate, modify, communicate and evaluate alternative designs (AC9TDI8P08) Design algorithms involving nested control structures and represent them using flowcharts and pseudocode (AC9TDI8P05) Trace algorithms to predict output for a given input and to identify errors (AC9TDI8P06) Implement, modify and debug programs involving control structures and functions in a general-purpose programming language (AC9TDI8P09) |

# **Assessment**

Students can undertake a self-reflection of the programming task. The teacher can use the completed self-assessments to assist in summative assessment.

* Download the self-assessment sheet in Word or PDF format.

In assessing code in languages like JavaScript, consider a rubric that brings in important skills for General Purpose Programming.

* Download a sample rubric in Word or PDF format.

# **Learning hook**



*The Macintosh 128K released in 1984 (image CC-BY-SA-2.5-it All About Apple museum)*

Students will start this lesson by taking a trip back in time to try a computer their parents may have used! They'll do this by testing 3 virtual computers running in the web browser. (Note, these websites have been tested on computer / laptop, but they may not work on iPad or Chromebook.)

* Windows 3.1 (released 1992)
* Windows 95 (released 1995)
* Macintosh (released 1984)

Students may work in pairs to visit one or more of the virtual computers, and report back to the class on the following:

* What do you notice about the colours available?
* What symbols do you recognise?
* Explain one way that your experience of the interface is different to the modern equivalent (Windows or Mac).
* Explain one way that your experience of the interface is thesame as the modern equivalent (Windows or Mac).

The students have been testing some of the earliest mouse-driven **Graphical User Interfaces** (GUIs). Prior to the arrival of Mac and Windows operating systems, most computer users had only a keyboard and text commands to make everything happen, from creating and deleting files to playing games.

In this lesson series, we'll start each lesson by exploring GUI design principles, as well as **conventions -** things that GUIs have in common. Many of these conventions have been around for decades.

# **Learning map and outcomes**

In this lesson, students will:

1. access an online programming environment for **JavaScript** alongside **HTML** and **CSS**,
2. create a colourful pH scale using HTML and CSS alone, without JavaScript,
3. create your first interactive JavaScript program that looks good: a times table generator,
4. take on a fresh coding challenge to make a simple JavaScript program with output on a webpage.

# **Learning input**

Let's start with some terms. Read these carefully, then answer the questions below.

|  |  |
| --- | --- |
| **HTML:** HyperText Markup LanguageHTML is one of the earliest languages of the Internet. It defines the **structure** of a webpage, the static content that will appear on the page when it first loads. It also allows for links. You can make a webpage purely in HTML, but it's hard to make it look good!You might think of HTML as the **skeleton, muscles and nerves** of a webpage. |  |
| **CSS:** Cascading Style SheetsCSS is a script that defines the **style** of a webpage, allowing for webpages to have interesting and modern look-and-feel: colours, fonts, curved borders, etc.You might think of CSS as the **skin and** **clothes** of a webpage. | Picture of a woman's head wearing pink sunglasses |
| **JavaScript**JavaScript defines the **functionality** of a webpage, allowing interactivity beyond just links. It is the only true general purpose programming language of the three. With JavaScript, webpages can have all the functionality of applications; with loops, decisions, functions and data structures.You might think of JavaScript as the **brain** of a webpage. |  |

**QUESTIONS:**

Out of HTML, CSS and JavaScript…

1. Which is a true general purpose programming language?

2. Which defines the structure of a webpage?

3. Which allows for loops, decisions and functions for a webpage?

4. Which helps to define the look-and-feel of a webpage?

**ANSWERS**

1. JavaScript

2. HTML

3. JavaScript

4. CSS

# **Learning construction**

**STEP 1: SETUP**

The video below explains different online environments for creating webpages with HTML, CSS and JavaScript. It also explains an offline method to create the files on your own computer with a plain text editor like Notepad.

View these videos on setting up coding environments: CodePen, JSFiddle and repl.it.

In this course, different environments will be selected based on their suitability for each demonstrated project.

All final code and solution code will be provided as:

* a link to the project created in the environment used in the demonstration, *and / or*
* a link to the project created in the repl.it online environment.

SIDEBAR – What about Python?

The Visual To Text Coding lesson series introduced both JavaScript and Python in parallel, but it is necessary to part ways when we introduce GUIs.

While the principles for good GUI design are universal, JavaScript and Python require very different coding approaches to implement GUIs. The common approach with Python is to use specific frameworks that bring their own modules.

Some current popular frameworks (as of 2020) for GUIs in Python are:

* TKinter
* PySimpleGUI: A code "wrapper" designed to simplify the use of frameworks like TKinter above.
* PyGame: A framework designed for game-making, sometimes used for GUIs.

**STEP 2: A NON-INTERACTIVE WEBPAGE**

The video below demonstrates creating a webpage with only HTML and CSS. This demonstration is done in the CodePen environment.

View this video on pH scale.

Try it yourself using this pre-written tutorial before checking the solution code.

*Solution code: CodePen environment, repl.it environment*

**STEP 3: TINKER TASK**

Make the following changes to the pH scale website by editing the HTML *and/or* the CSS:

1. Change the text colour for "08 Neutral" to black.
2. Change the background colour for the entire page to a light grey: rgb(200, 200, 200)
3. Add a short subtitle under the heading but above the scale itself: **pH was introduced as a concept in 1909.**Like the main heading, the subtitle should be aligned to the left. It should have a font size of 20 pixels.

*Solution code: CodePen environment, repl.it environment*

**STEP 4: A JAVASCRIPT PROGRAM THAT LOOKS GOOD**

The video below demonstrates creating a webpage with content generated by JavaScript code. This demonstration is done in the JSFiddle environment.

View this video on time table generator.

Try it yourself before checking the solution code.

*Solution code: JSFiddle environment, repl.it environment*

**STEP 5: TINKER TASK**

Make the following changes to the Times table generator by editing the HTML, CSS or JavaScript:

1. Change the blackboard to look like a whiteboard. Board colour should be white and text should be black.
2. Change the times table to a power table, eg.
2 to the power of 0 = 1
2 to the power of 1 = 2.
2 to the power of 2 = 4.
…

*Hint: In JavaScript, use* \*\* *to do a power operation. eg. 5 \*\* 2 gives 25.*
3. Change the main heading text from TIMES TABLES to POWER TABLE.

*Solution code: JSFiddle environment, repl.it environment*

# **Challenge**

These challenges use the skills covered so far. By writing or modifying their own programs, students have an opportunity to demonstrate Application and Creation.

1. Using your preferred online environment, create a website **using only HTML and CSS** to show a First Nations seasonal calendar with broad, coloured bands.

The solution screenshot below shows the traditional seasons used by the Wurundjeri, an Australian First Nations people of the Kulin nation. The information was obtained from this website, and the specific colours were picked from the image there too. Note that the information source is also linked under the heading at the top of the finished page.



*Solution code: JSFiddle environment, repl.it environment*

1. Create a webpage that prompts the user for a number to start a countdown, then displays the countdown from that number to 1, followed by "BLASTOFF!". The page should have a dark red background with a black border, and white text.

You'll need to use HTML, CSS and JavaScript for this.

The solution screenshot below shows what is presented if the user enters 16 when prompted.



*Solution code: JSFiddle environment, repl.it environment*

# **Resources**

* Online environments for creating webpages with HTML, CSS and JavaScript:
	+ CodePen – quick prototyping with automatic refresh of webpage
	+ JSFiddle – simple interface that hides linking HTML code, also used in Visual To Text Coding lesson series
	+ repl.it – shows complete HTML to reflect offline approach, and allows uploading of images and other files for use in webpages
* Introductory courses
	+ HTML tutorial – An introduction to HTML
	+ CSS tutorial – An introduction to CSS
	+ Visual To Text Coding lesson series – The predecessor to this learning sequence introduces JavaScript as well as Python.
* JavaScript CheatSheet (Tip: Press the little blue tabs to move **Variables**, **Basics**, **Strings** and **Data Types** to the top.)