**Coding for GUIs (JavaScript edition)
LESSON 5: GUIfy my program!**

This is the fifth in a series of lessons to incorporate Graphical User Interfaces (GUIs) into your General Purpose Programming. The series follows on from the [Visual To Text Coding lesson series](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/visual-to-text-coding-index-page).

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 steps through the complete upgrading of a Heads or Tails game from text-only to GUI with animation. Two other completed GUI programs are also provided for students to examine.

**Year levels**: 7–8

# **Curriculum links**

Links with Digital Technologies Curriculum Area

| **Strand** | **Year** | **Content Description** |
| --- | --- | --- |
| Processes and Production Skills | 7-8 | Design the user experience of a digital system [(AC9TDI8P07)](https://v9.australiancurriculum.edu.au/f-10-curriculum.html/learning-areas/digital-technologies/year-7_year-8/content-description?subject-identifier=TECTDIY78&content-description-code=AC9TDI8P07&detailed-content-descriptions=0&hide-ccp=0&hide-gc=0&side-by-side=1&strands-start-index=0&subjects-start-index=0&view=quick)Generate, modify, communicate and evaluate alternative designs [(AC9TDI8P08)](https://v9.australiancurriculum.edu.au/f-10-curriculum.html/learning-areas/digital-technologies/year-7_year-8/content-description?subject-identifier=TECTDIY78&content-description-code=AC9TDI8P08&detailed-content-descriptions=0&hide-ccp=0&hide-gc=0&side-by-side=1&strands-start-index=0&subjects-start-index=0&view=quick)Design algorithms involving nested control structures and represent them using flowcharts and pseudocode [(AC9TDI8P05)](https://v9.australiancurriculum.edu.au/f-10-curriculum.html/learning-areas/digital-technologies/year-7_year-8/content-description?subject-identifier=TECTDIY78&content-description-code=AC9TDI8P05&detailed-content-descriptions=0&hide-ccp=0&hide-gc=0&side-by-side=1&strands-start-index=0&subjects-start-index=0&view=quick)Trace algorithms to predict output for a given input and to identify errors [(AC9TDI8P06)](https://v9.australiancurriculum.edu.au/f-10-curriculum.html/learning-areas/digital-technologies/year-7_year-8/content-description?subject-identifier=TECTDIY78&content-description-code=AC9TDI8P06&detailed-content-descriptions=0&hide-ccp=0&hide-gc=0&side-by-side=1&strands-start-index=0&subjects-start-index=0&view=quick)Implement, modify and debug programs involving control structures and functions in a general-purpose programming language [(AC9TDI8P09)](https://v9.australiancurriculum.edu.au/f-10-curriculum.html/learning-areas/digital-technologies/year-7_year-8/content-description?subject-identifier=TECTDIY78&content-description-code=AC9TDI8P09&detailed-content-descriptions=0&hide-ccp=0&hide-gc=0&side-by-side=1&strands-start-index=0&subjects-start-index=0&view=quick) |

# **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](http://www.digitaltechnologieshub.edu.au/docs/default-source/Lesson-Ideas/coding-for-guis/lesson-5/self-assessment---coding-for-guis---lesson-5.docx) or [PDF](http://www.digitaltechnologieshub.edu.au/docs/default-source/Lesson-Ideas/coding-for-guis/lesson-5/self-assessment---coding-for-guis---lesson-551cf4c9809f96792a599ff0000f327dd.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](https://www.digitaltechnologieshub.edu.au/docs/default-source/Lesson-Ideas/visual-to-text-coding/rubric-example---software-design-and-development-with-general-purpose-language6baf4a9809f96792a599ff0000f327dd.docx) or [PDF](https://www.digitaltechnologieshub.edu.au/docs/default-source/Lesson-Ideas/visual-to-text-coding/rubric-example---software-design-and-development-with-general-purpose-language.pdf) format.

# **Learning hook**

You and your friend are co-developing an application with the ability to save progress.

As a helpful feature, your friend proposes a pop-up for when the user tries to exit the program without saving their work:



As a class, discuss if you see any problem with the above UI design for this pop-up. For comparison, bring up a similar pop-up when exiting Microsoft Word, Notepad or another application like Audacity.

SAMPLE ANSWER

There is a serious problem with this design. It does not conform with user expectations for the placement of the three buttons.

**The convention for pop-ups of this kind is for the Cancel button to be on the right.** A user pressing the right-most button in this dialog may be expecting to return to their work without exiting, but instead they would be confirming to exit *and* lose their work.

*The activity above touches on the last of 4 principles for good User Interface design (distilled from* [*this article*](https://xd.adobe.com/ideas/process/ui-design/4-golden-rules-ui-design/) *hosted by Adobe):* ***Make user interfaces consistent****. A GUI should:*

* *employ visual consistency (same style, fonts, colours throughout)*
* *functional consistency (no surprises in the way the UI behaves)*
* *conform with user expectations (follow conventions and avoid reinventing)*

# **Learning map and outcomes**

In this lesson, students will:

1. access an online programming environment for **JavaScript** alongside **HTML** and **CSS**,
2. upgrade a text-only **heads or tails game** to a full GUI with animation,
3. examine two other completed GUI applications upgraded from text-only.

# **Learning input**

The main example in this lesson is a **heads or tails game**.

First, review and test the text-only version of the game in JavaScript [here](https://repl.it/%40digitechhub/Lesson-5-Text-only-game#script.js). *(The game was first introduced and built in* [*Lesson 3 of the course previous to this one*](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/visual-to-text-coding-lesson-3-heads-or-tails/)*).*

Next, access the [starter project](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-starter-project) for the new GUI version here. It contains skeleton code that will be built on as the lesson progresses.

Finally, view [this video](https://publish.viostream.com/player/download/bxixurbdnmjy8d). It gives an overview of the whole lesson and explains the starter project.

# **Learning construction**

**STEP 1: Setup**

View this [video](https://publish.viostream.com/player/download/bxixurbdnosebo) on setting up.

In this course, different environments are selected based on their suitability for each demonstrated project. This lesson uses the repl.it environment exclusively.

For more on the setup and environments used, see [Lesson 1](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/coding-for-guis-lesson-1-structure-style-and-function/#learning-construction).

**STEP 2: Constructing the Heads Or Tails game**

This game is the most complex of the GUI applications made in this course so far. Each video below tackles a part of the coding process.

First, set up the function for user button presses.

Watch the video: [Heads or tails: Get user input](https://publish.viostream.com/player/download/bxixurbdnmjy8g)

*Completed code so far:* [*repl.it environment*](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-Partway-A)

Next, begin the code for flipping the coin.

Watch the video: [Heads or tails: Prepare coin flip](https://publish.viostream.com/player/download/bxixurbdnmjy8k)

Watch the video: [Heads or tails: Flip the coin](https://publish.viostream.com/player/download/bxixurbdnmjy88)

Watch the video: [Heads or tails: Set interval](https://publish.viostream.com/player/download/bxixurbdnmjy8f)

*Completed code so far:* [*repl.it environment*](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-Partway-B)

Next, flip the coin images and stop after a random number of flips.

Watch the video: [Heads or tails: Swap images](https://publish.viostream.com/player/download/bxixurbdnmjy8r)

Watch the video: [Heads or tails: Stop coin flipping](https://publish.viostream.com/player/download/bxixurbdnmjnmm)

Watch the video: [Heads or tails: How long to spin the coin](https://publish.viostream.com/player/download/bxixurbdnmjy8n)

*Completed code so far:* [*repl.it environment*](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-Partway-C)

Now, complete the basic game by checking if the user's guess matches the coin after it stops.

Watch the video: [Heads or tails: Determine a match](https://publish.viostream.com/player/download/bxixurbdnmjy8m)

Watch the video: [Heads or tails: Basic styling](https://publish.viostream.com/player/download/bxixurbdnmjy8j)

*Completed code so far:* [*repl.it environment*](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-Partway-D)

**STEP 3: Improving the Heads Or Tails Game**

First, we'll add a scoring system.

Watch the video: [Heads or tails: Adding scores](https://publish.viostream.com/player/download/bxixurbdnmjnkh)

*Completed code so far:* [*repl.it environment*](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-Partway-E)

Next, keep track of how many turns it takes the user to win.

Watch the video: [Heads or tails: How many turns](https://publish.viostream.com/player/download/bxixurbdnmjy8e)

*Completed code so far:* [*repl.it environment*](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-Partway-F)

Finally, improve the buttons and fonts.

Watch the video: [Heads or tails: Better buttons](https://publish.viostream.com/player/download/bxixurbdnmjibb)

Watch the video: [Heads or tails: Google fonts](https://publish.viostream.com/player/download/bxixurbdnmjibg)

*Completed code so far:* [*repl.it environment*](https://repl.it/%40digitechhub/Lesson-5-Heads-or-Tails-Final)

**STEP 4: Two other GUI applications**

Before the major project, here are two more completed GUI applications for you to look at.

Test them out first, then take the time to step through the HTML, CSS and JavaScript to see if you can follow how they work. Tinker with some of the code to change the behaviour of the programs.

* [Magic 8 Ball](https://repl.it/%40digitechhub/Magic-8-Ball-standard) (original text-only program from [this lesson](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/visual-to-text-coding-lesson-6-magic-8-ball/))

*(Students may also wish to test an experimental mobile version of this webpage, which responds to tilting the mobile device. To do so, access* [*this link*](https://digitaltechnologieshub.s3-ap-southeast-2.amazonaws.com/magic_eight/magic_eight.html) *on a mobile device. Unfortunately, repl.it does not reliably support these features at time of writing, so students cannot edit the code.)*

* [Scissors Paper Rock](https://repl.it/%40digitechhub/Scissors-Paper-Rock) (original text-only program from [this lesson](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/visual-to-text-coding-lesson-4-scissors-paper-rock))

# **Challenge**

This lesson does not include a fresh coding challenge. Students can undertake a final project as set out on [this page](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/coding-for-guis-final-project-a-complete-application-with-gui/).

# **Resources**

* Online environments for creating webpages with HTML, CSS and JavaScript:
	+ [JSFiddle](https://jsfiddle.net) – simple interface that hides linking HTML code, also used in [Visual To Text Coding lesson series](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/visual-to-text-coding-index-page)
	+ [repl.it](https://repl.it) – shows complete HTML to reflect offline approach, and allows uploading of images and other files for use in webpages
* [Visual To Text Coding lesson series](https://www.digitaltechnologieshub.edu.au/teachers/lesson-ideas/visual-to-text-coding-index-page) – The predecessor to this learning sequence introduces JavaScript as well as Python.
* JavaScript [CheatSheet](https://htmlcheatsheet.com/js/) (Tip: Press the little blue tabs to move **Variables**, **Basics**, **Strings** and **Data Types** to the top.)
* Articles and lessons on User interface principles:
	+ [UI Design principles distilled](file:///C%3A%5CUsers%5CNathan%5CDocuments%5CCurrent%20Projects%5CDigital%20Technologies%20Hub%5CCoding%20for%20GUIs%20Lesson%20Ideas%5CJavaScript%20edition%5CLesson%202%20-%20Controls%20for%20input%5Cxd.adobe.com%5Cideas%5Cprocess%5Cui-design%5C4-golden-rules-ui-design) (Adobe)
	+ [Topic page](file:///C%3A%5CUsers%5CNathan%5CDocuments%5CCurrent%20Projects%5CDigital%20Technologies%20Hub%5CCoding%20for%20GUIs%20Lesson%20Ideas%5CJavaScript%20edition%5CLesson%202%20-%20Controls%20for%20input%5Co%09www.digitaltechnologieshub.edu.au%5Cteachers%5Ctopics%5Cuser-interface) on Digital Technologies Hub
	+ [Classic CS Unplugged lesson](file:///C%3A%5CUsers%5CNathan%5CDocuments%5CCurrent%20Projects%5CDigital%20Technologies%20Hub%5CCoding%20for%20GUIs%20Lesson%20Ideas%5CJavaScript%20edition%5CLesson%202%20-%20Controls%20for%20input%5Cclassic.csunplugged.org%5Chuman-interface-design%5C)
	+ [code.org lesson](file:///C%3A%5CUsers%5CNathan%5CDocuments%5CCurrent%20Projects%5CDigital%20Technologies%20Hub%5CCoding%20for%20GUIs%20Lesson%20Ideas%5CJavaScript%20edition%5CLesson%202%20-%20Controls%20for%20input%5Ccurriculum.code.org%5Cpwc%5Cayp%5C6%5C)

**Discuss as a class:** How can students ensure that images they are using in their GUIs are not restricted due to copyright?

**Suggested answer:**By understanding [Creative Commons](https://creativecommons.org.au/learn/education/) and Public Domain licenses, and learning about the places to find them, students can use images appropriately, giving attribution when necessary.