CODE WARRIORS

HAKITZU BATTLES

TEACHERS GUIDE

OVERVIEW

What is Code Warriors?

Code Warriors is a game designed to introduce children and young people to programming JavaScript, one of the world's most commonly used programming languages. The aim of Code Warriors is to provide an exciting introduction to the world of coding and help players understand some simple concepts.

Gameplay?

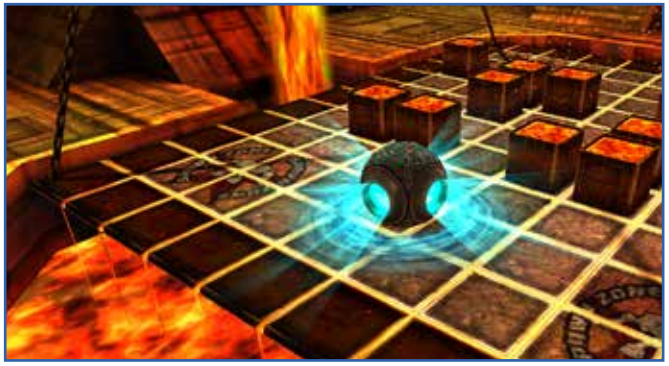
Code Warriors is a turn-based strategy game for two players, similar to chess. Players start at opposing ends of the board with two programmable battle robots, known as 'Code Warriors' and a Power Core to defend. The aim for each player is to reach their opponent's Power Core and destroy it, whilst protecting their own. The first player to destroy their opponent's Power Core wins the match. Code Warriors carry various weapons and can be programmed to battle each other as they move around the board. Players can also choose Single Player Challenges to hone their programming skills between matches.

Code Warriors

Mighty battle robots. Choose between the slow and powerful Tank class or the lighter, more agile Scouts. Code Warriors can be customised in their appearance and weapons.

Battle Arena

The 8x10 board where Code Warriors matches take place. There are four different battle arenas to choose each with different obstacles.



Power Core

Situated at each end of the board, Players must defend their own Power Core whilst attempting to destroy their opponents.

STRATEGY VIEW

There are two main viewpoints in Code Warriors: the strategy view and the arena view. This is the strategy view, where players plan their moves and write their code.

Function Libary

This area shows available moves and the code required to perform them.

Battle Arena

The overhead view can be zoomed by pinching and dragging on the screen to help plan strategy.

Code Editor

The player writes their code in this window

Action Points

The player writes their code in this window

Execute

The player presses execute to finish their turn. This will take them to the arena view where they will watch their programme play out.

Code Warrior Status

Displays health of currently selected Code Warrior. Code Warriors destroyed in battle sit out three game turns before returning to the field.

Arena View

The Arena View is the other main viewpoint in Code Warriors. This is where players watch their programmes as they execute, and their opponent's turns play out.

Function Libary

This area shows available moves and the code required to perform them.

Codewalker

The players Code Warrior will act out moves as programmed.

The Chop Shop

The Chop Shop is a garage area where players can customise their Code Warriors before battle. Players can spend Coding Credits earned through in-game achievements on a wide variety of body parts, weapons and colour schemes.

Code Warriors

The player can store up to four Code Warrior configurations in these slots.

Class

Choose between the powerful Tank class or the more agile Scout Code Warriors.

Credits

Coding Credits are earned by winning matches and other in-game achievments. Playing at harder Coding Ranks earns more Coding Credits.

Parts and Weapons

Coding Credits can be spent on a wide range of body parts, weapons and paint styles.

The Chop Shop (cont.)

Code Warriors can be equipped with a wide variety of weapons that do different amounts of damage. Each Code Warrior can carry one Melee (hand-to-hand combat) and one Ranged (firing) weapon at a time. Part of the strategic gameplay of Code Warriors is in choosing your weapons wisely.

Players can unlock more powerful weapons by completing in-game achievements, such as winning matches or solving challenges.

Weapon Stats

Weapons inflict different amounts of damage and cost different amounts of Action Points to use. Ranged weapons fire over certain distances. Some weapons, such as the Rocker Launcher, also have Splash Damage, meaning they will affect nearby squares when they explode.

CODING

Coding Ranks

Code Warriors is designed to progress the player through four coding levels or ranks. All players start at the simple 'tap and play' Beginner rank and progress through to writing code with auto-complete help, to the final Warrior rank where all code is typed by hand. Each coding rank awards a different amount of Coding Credits for victories, which can then be spent in the Chop Shop. Each rank is explained in detail over the next few pages.

Beginner

The player is not required to write any code. The player presses the icon for the type of move they want and then taps on the board.

Coder

This rank is similar to Junior Coder but has less auto-complete assistance.

Junior Coder

In this level, the player begins to type the code themselves. However auto-complete makes for faster input.

Warrior

At this level the player has no auto-complete support. This is equivalent to coding using a basic text editor.

CODING RANK

Beginner

In the Beginner coding rank, the player is not required to write any code.The player simply presses the icon for the type of move they want and then taps on the board. The code will then automatically be generated as follows:

Functions

The player taps the icon for the desired move.

Move Preview

Available moves are highlighted on the board. The player taps the board to select where they want to move or attack.

Code editor

The player observes the JavaScript code that is generated.

Execute

The player presses execute to finish their turn.

CODING RANK

Junior Coder

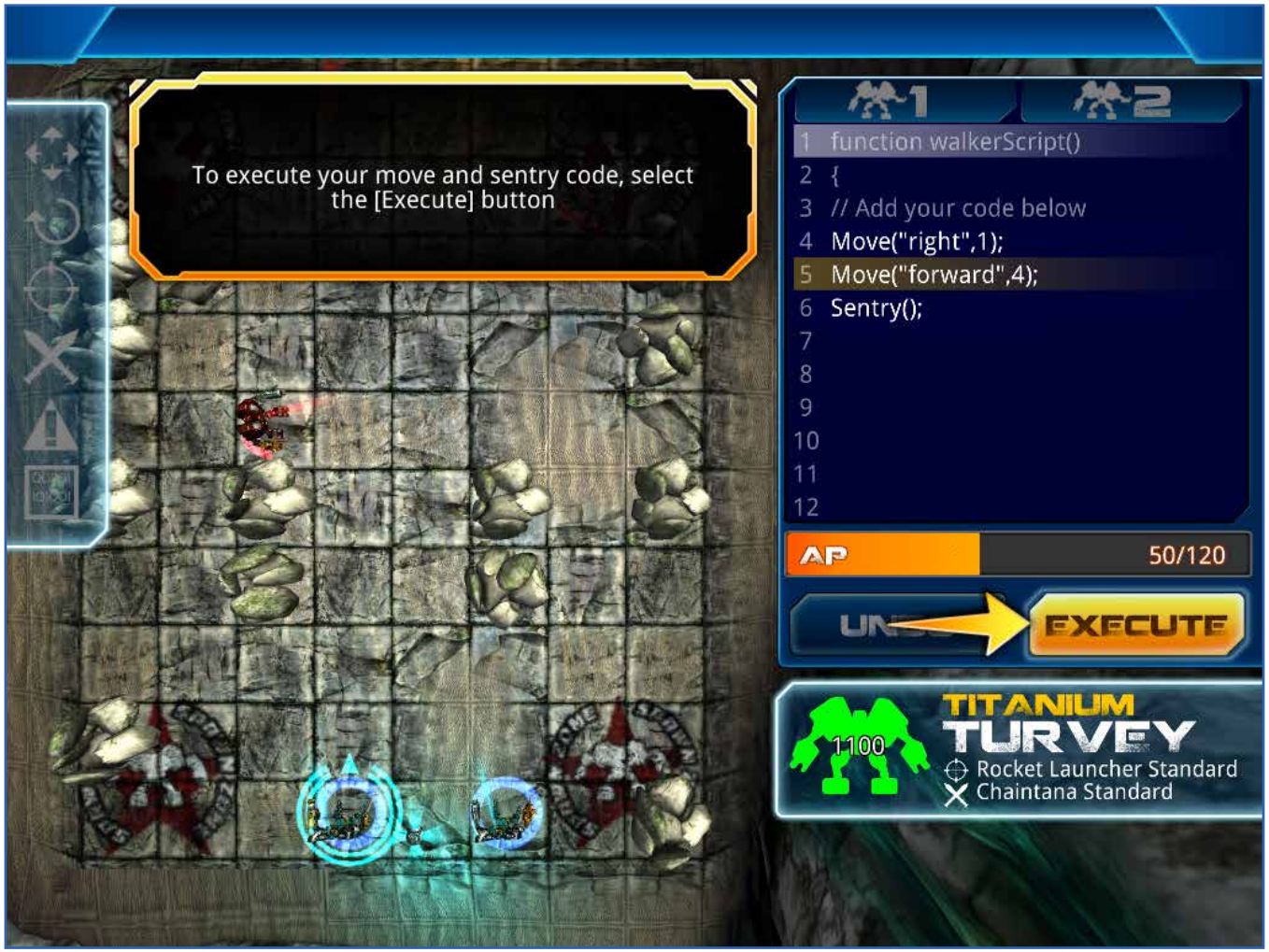
When playing in Junior Coder rank, the player begins to type the code themselves but is assisted by the Auto-Complete system. For example, if the player wishes to move, the player needs to type “m” which will bring up the suggested input of “Move”. The player can then tap the word that appears in the Code Editor window or the spacebar, to accept the auto-complete suggestion.

Functions

The player taps the icon for the desired move.

Auto Complete

In Junior Coder rank, the auto-complete system will fill in the majority of the code for the player.



Code editor

The player can alter basic parameters such as how many squares forwards or backwards.

Keyboard

The special keyboard has commonly used coding symbols on the surface layer, to enable faster coding

Coder and Warrior

Coder rank is similar to Junior Coder, but with a reduced amount of Auto-Complete help. Warrior is the hardest rank of all with no Auto-Complete available. At all coding ranks the Code Editor window will highlight any errors in the code and help the player correct them.

Error Highlight

At all Coding Ranks, errors in the code are highlighted with guidance on how to correct.

Functions

The following is a summary of all the available functions in Code Warriors. Players may refer to the Function Library at any time whilst in the strategy view to remind themselves of the functions available and the code required for each type of move.

Move

Code Warrior moves to a selected square in the arena.
Example code: **Move("forward",2);**

Turn

Rotates a Code Warrior 90 degree per turn.
Example code: **Turn("left",1);**

Attack Ranged Weapon

Code Warrior attacks using whichever ranged weapon it is carrying. Ranged weapons, such as the Rocket Launcher fire across the board. Example code: **FireRocket();**

Attack Melee Weapon

Code Warrior attacks using whichever melee weapon it is carrying. Melee weapons, such as the Laser Axe, are for close combat.
Example code: **Slash();**

Sentry

In this defensive mode, the Code Warrior will fire automatically only if an enemy Code Warrior crosses its line of sight.
Example code: **Sentry();**

Delete

This special function is used to destroy an opponent's core and can only be played when adjacent and facing the core.
Example code: **Delete();**

Action Points

Each command issued to a Code Warrior costs Action Points. Players have a set number of Action Points available per turn to spend between their two Code Warriors. Part of the strategic gameplay of Code Warriors is deciding how best to spend Action Points on each turn.

GETTING STARTED

BOOTCAMP TUTORIAL

On launching Code Warriors, the player is taken through an initial guided tutorial using the Beginner ‘tap and play’ approach. Completing this tutorial unlocks both Multiplayer mode as well as the Junior Coder guided tutorial. Completing the Junior Coder tutorial unlocks the Combat and Stealth Single Player Challenges.

Guided Tutorial

Players simply follow the on-screen instructions to complete the tutorial.

All the key parts of the game, from moving, attacking and Deleteing a Power Core are covered in the tutorial.

INVITING FRIENDS

After selecting 'Multiplayer' from the main menu, players can choose to challenge their friends to a game of Code Warriors via email, Game Center or Facebook. Players can also choose to be matched against a random opponent.

Friends List

Displays existing friends

Search

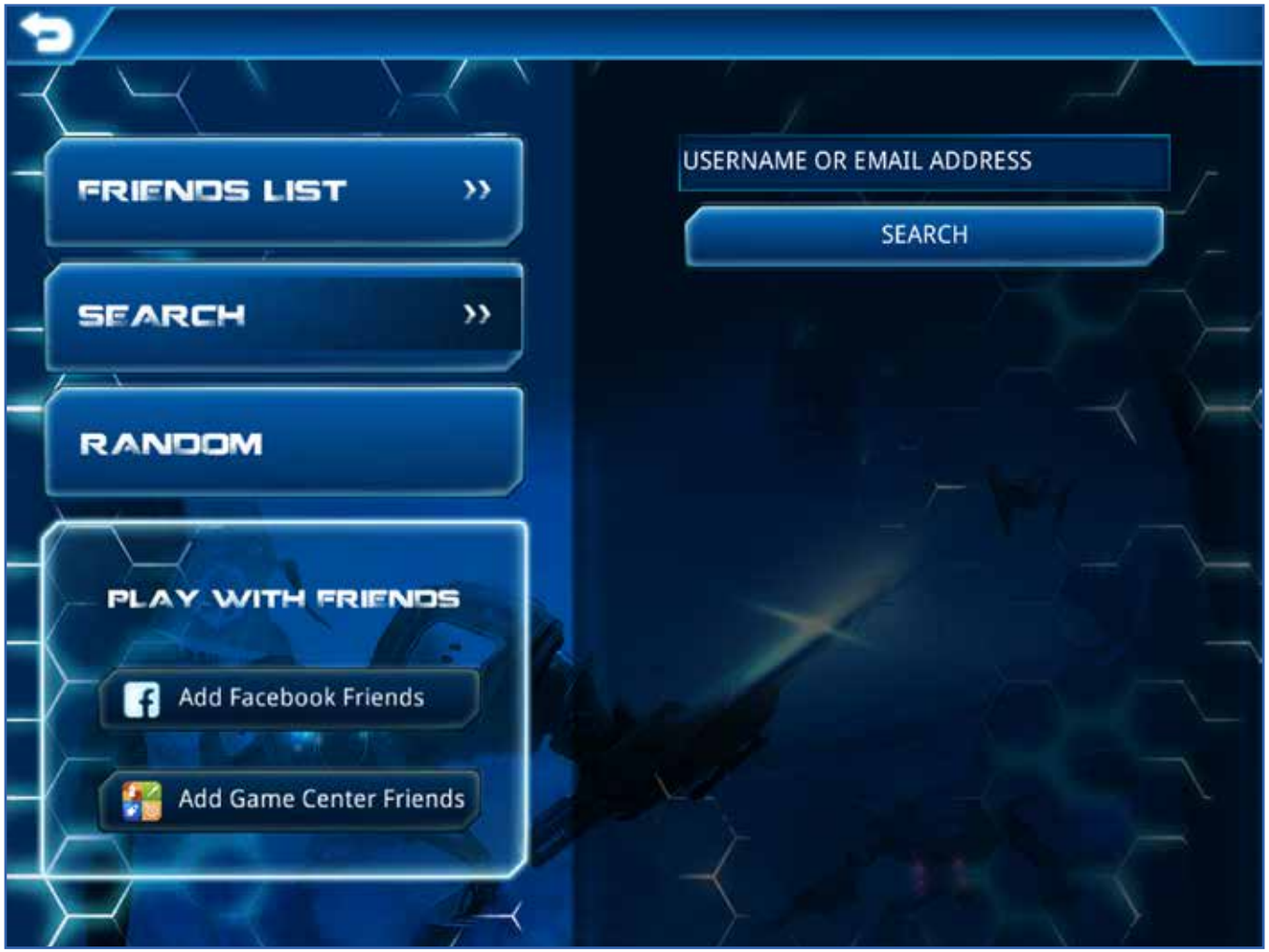
Searches for friends by username or email address.

Random

Matches user against a random opponent.

Play with Friends

Add existing Facebook or Game Center friends.



MULTIPLAYER OPTIONS

The Match Options screen allows you to choose your Code Warriors, arena and coding rank when setting up a new multiplayer game.

Code Warriors

The player is able to save up to four Code Warrior configurations in the Chop Shop. The player selects their chosen Code Warriors in this window.

Coding Rank

The player selects the coding rank they wish to use for this match.

Arena Select

The player can choose from four different battle arenas.

HELP SYSTEMS

At any time during the game, the player can access the in-game help by pressing the “?” icon in the upper right hand corner of the screen. This will display further small ‘?’ symbols over any important buttons or areas of the game. Pressing these will bring up detailed information about the chosen item.

Help Buttons

Press on any icon to view its help information.

Help System

Press here to activate the help system. Any important areas will be highlighted with a smaller “?” icon.

LEARNING

DASHBOARD

GROUP PERFORMANCE

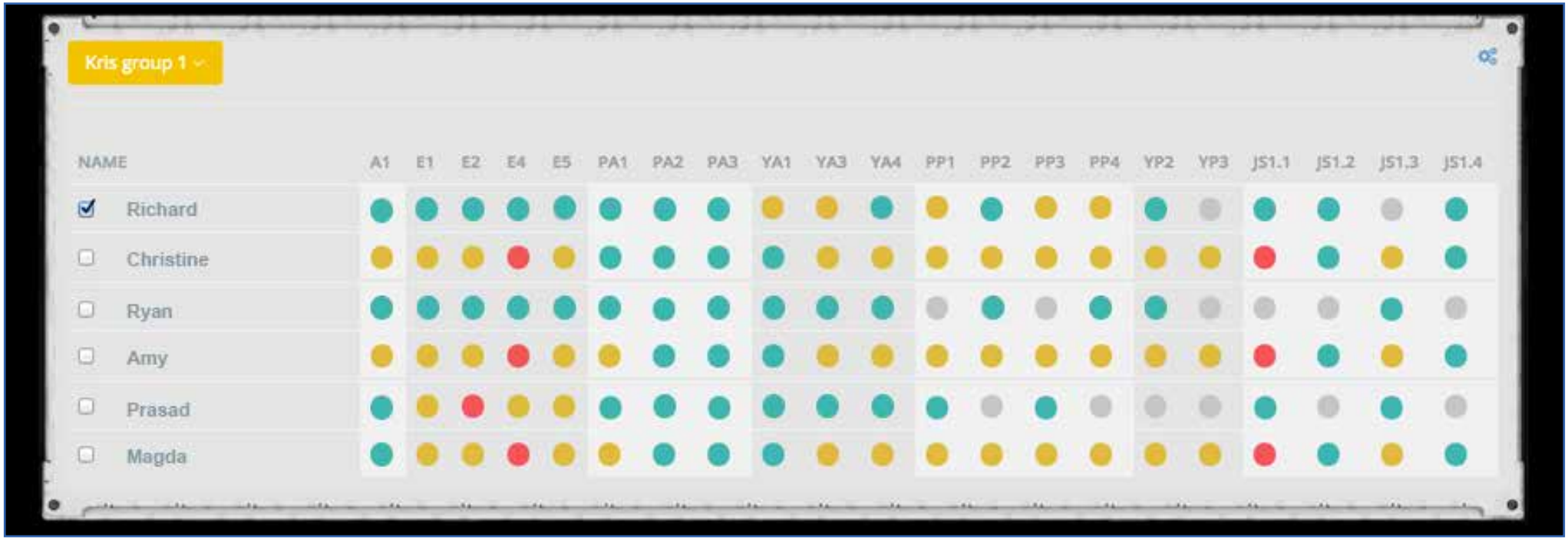
The Group Performance chart displays a simple visual summary of individual / group performance against key Learning Outcomes.

- Green indicates fluency with a specific learning outcome
- Amber indicates on track but fluency not yet achieved
- Red indicates no longer on track to achieve fluency

The Graph view allows teachers to monitor progress within each Learning Outcome on a session by session basis.

Choose your class

Learning Outcomes



A concept has yet to be introduced.

Not on track to achieve fluency.

Achieved fluency in a specific learning outcomes.

Introduced to concept

DASHBOARD

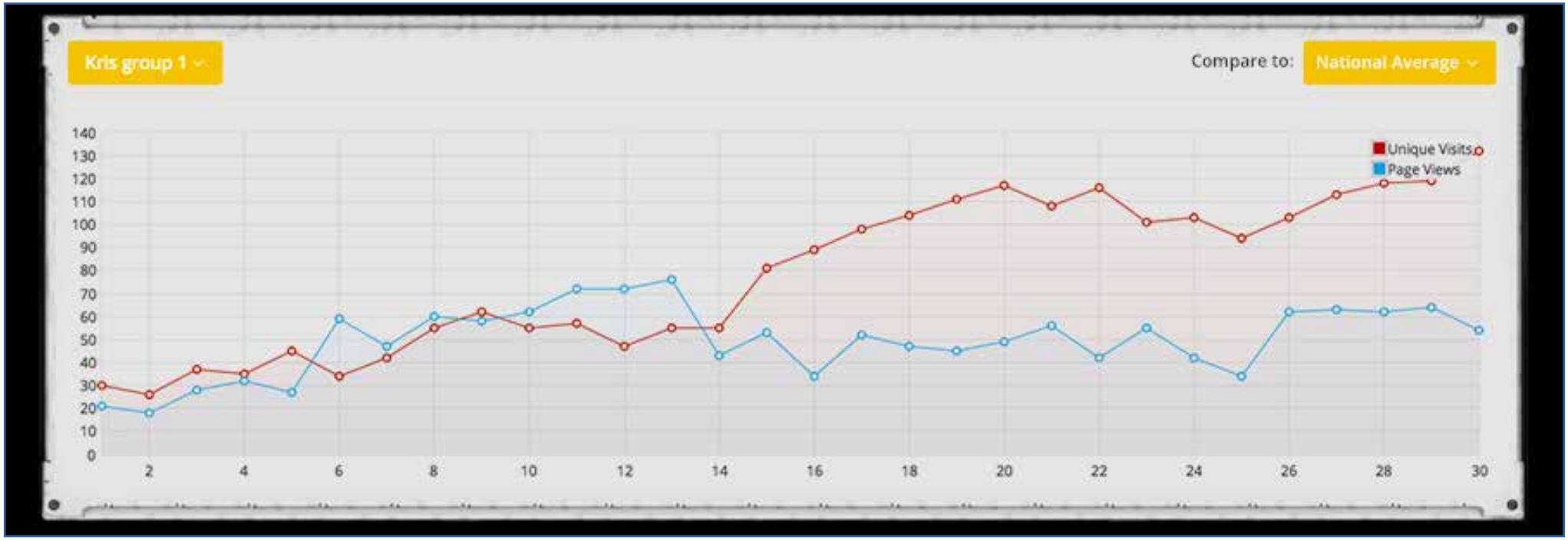
GRAPH

The Graph view allows teachers to monitor progress within each Learning Outcome on a session by session basis. Select an individual student (or group of students) and learning outcome to filter the graph. Student performance can be measured against class, national or global average.

Choose your Learning Outcome

Progress across multiple sessions.

Compare against class/national/global average



LEARNING

Code Warriors offers players across the age range 8 to 14 an introduction to JavaScript, one of the world's most popular programming languages. As well as learning the rudiments of a real programming language, through play, children will learn about algorithms and computational thinking.

The Learning Outcomes defined in the Dashboard are based on the Computing at School Progression Pathways document found at http://computingatschool.org.uk/, and at the Progression Pathways website - http://www.progression-pathways.co.uk/

Coding Ranks

The following sections - BEGINNER, JUNIOR CODER, CODER and WARRIOR - give oversight on the learning outcomes as children play through the different levels of Code Warriors.

BEGINNER

At beginner level, the focus is on developing an understanding of algorithms and computational thinking.

There are 10 Combat, 10 Stealth, 10 Maze, and 10 Comprehension Challenges. Each of these encourages familiarity with the concept that computer programs are based on algorithms - and that algorithms are a set of precise instructions, upon which a computer can process and act.

Maze Challenges

Students select functions from the Function Library on the left - Move, Turn, Fire, Smash, Sentry, and Delete - that will enable them to achieve their mission - to Delete the Core, or attack other Warriors. The player simply highlights the appropriate square to indicate their move. When a selection is made, the code is revealed in the Code Editor.

Comprehension Challenges

Students are asked to examine a block of code, and decide if the code will run successfully - will it complete the mission? There are logic errors and syntax errors which the player must look out for in order to complete the challenge successfully. This is a right/wrong challenge, with no star weighting.

The Combat Challenges and Stealth Challenges have a star weighting. Three stars suggests a student has solved the challenge in the most efficient way - that is, they have used the fewest Action Points to solve the challenge. One star suggests they have successfully solved the challenge, but that a more efficient solution is possible.

BEGINNER (cont.)

A range of learning outcomes derived from the Computing at School Progression Pathways are targeted at BEGINNER level.

Evaluation

- E1 Assessing whether an algorithm is fit for purpose
- E2 Assessing whether an algorithm does the right thing (functional correctness)
- E4 Assessment whether the performance of an algorithm is good enough

Algorithms

- PA2 I know that computers need precise instructions
- YA1 I know that algorithms are implemented on digital devices as programs
- YA3 I can use logical reasoning to predict outcomes

Programming and Development

- PP1 I can create a simple program
- PP2 I can run, check and change programs
- PP3 I know that programs run by following precise instructions
- YP2 I can use logical reasoning to predict the behaviour of programs
- YP3 I can find and correct simple semantic errors i.e. debugging, in programs

Links to UK national curriculum

Students will learn to:

- Understand what algorithms are and how they are implemented on digital devices; and that programs execute by following precise and unambiguous instructions
- Create and debug simple programs
- Use logical reasoning to predict the behaviour of simple programs
- Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs

LEARNING

JUNIOR CODER

In this level, students begin to write their own code with the aid of autocomplete. In addition to Combat, Maze and Comprehension Challenges, students can also access Parameter and Debug Challenges, which drill more deeply into students' fluency across a range of Learning Outcomes.

Parameter Challenges

Find and destroy the enemy! These challenges consolidate players' understanding of constant and variable parameters. Players must insert the correct parameters in each line of code to ensure that the code executes successfully.

Debug Challenges

Students are presented with lines of code that, if executed, should allow them to reach the Core or destroy enemy robots. However, there are errors! Their challenge is to locate and fix the bugs.

JUNIOR CODER reinforces students' understanding of algorithms, but they are now encouraged to think like a coder: running programs, using precise instructions, checking for errors... JUNIOR CODER also gives players the opportunity to write simple lines of JavaScript.

LEARNING

JUNIOR CODER (cont.)

Learning Outcomes at JUNIOR CODER:

Algorithmic Thinking

- AL1 Writing instructions that, if followed in a given order, achieve a desired output

Evaluation

- E1 Assessing whether an algorithm is fit for purpose
- E2 Assessing whether an algorithm does the right thing (functional correctness)
- E4 Assessment whether the performance of an algorithm is good enough

Algorithms

- PA1 I know what an algorithm is and can express simple algorithms using symbols
- PA2 I know that computers need precise instructions
- PA3 I can show care and precision to avoid errors
- YA1 I know that algorithms are implemented on digital devices as programs
- YA3 I can use logical reasoning to predict outcomes

Programming and Development

- PP1 I know that users develop their own programs, and that I can create a simple program
- PP2 I can run, check and change programs
- PP3 I know that programs run by following precise instructions
- YP3 I can find and correct simple semantic errors i.e. debugging, in programs

Javascript

- JS1.1 I understand how to execute basic JavaScript
- JS1.2 Constant parameters: I understand how different parameters affect a function's output
- JS1.3 I understand how to use JavaScript functions to achieve my aims
- JS1.4 Variable parameters: I understand how parameters affect the output of a function

Links to UK national curriculum

Students will learn to:

- understand what algorithms are and how they are implemented on digital devices; and that programs execute by following precise and unambiguous instructions
- create and debug simple programs
- use logical reasoning to predict the behaviour of simple programs
- use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs

LEARNING

CODER AND WARRIOR

In these levels, autocomplete is removed, forcing students to write their own lines of code. Precision and care become increasingly important. Playing the challenges in CODER and WARRIOR, students will consolidate the Learning Outcomes achieved in BEGINNER and JUNIOR CODER, but here players are introduced to two new outcomes:

- E5 Comparing the performance of algorithms that do the same thing
- YA4 I can find and correct errors i.e. debugging, in algorithms

CODER AND WARRIOR (cont.)

The full range of Learning Outcomes at CODER and WARRIOR:

Algorithmic Thinking

- AL1 Writing instructions that, if followed in a given order, achieve a desired output

Evaluation

- E1 Assessing whether an algorithm is fit for purpose
- E2 Assessing whether an algorithm does the right thing (functional correctness)
- E4 Assessment whether the performance of an algorithm is good enough
- E5 Comparing the performance of algorithms that do the same thing

Algorithms

- PA1 I know what an algorithm is and can express simple algorithms using symbols
- PA2 I know that computers need precise instructions
- PA3 I can show care and precision to avoid errors
- YA1 I know that algorithms are implemented on digital devices as programs
- YA3 I can use logical reasoning to predict outcomes
- YA4 I can find and correct errors, i.e. debugging, in algorithms

Programming and Development

- PP1 I know that users develop their own programs, and that I can create a simple program
- PP2 I can run, check and change programs
- PP3 I know that programs run by following precise instructions
- YP3 I can find and correct simple semantic errors, i.e. debugging, in programs

Javascript

- JS1.1 I understand how to execute basic JavaScript
- JS1.2 Constant parameters: I understand how different parameters affect a function's output
- JS1.3 I understand how to use JavaScript functions to achieve my aims
- JS1.4 Variable parameters: I understand how parameters affect the output of a function

SAMPLE LEARNING ACTIVITIES

Combat Challenges 1-5

Programming and Development

- PP1 I can create a simple program
- PP2 I can run, check and change programs

Javascript

- JS1.1 I understand how to execute basic JavaScript

Stealth Challenges 6-10

Algorithmic Thinking

- AL1 Writing instructions that, if followed in a given order, achieve a desired output

Algorithms

- PA2 I know that computers need precise instructions
- PA3 I can show care and precision to avoid errors

Javascript

- JS1.2 I understand how parameters can affect the output of a function

Parameter Challenges 1-5

JavaScript

- JS1.2 I understand how parameters can affect the output of a function
- JS1.4 I understand the difference between different types of values

Debug Challenges 6-10

Programming and Development

- PP2 I can run, check and change programs
- YP3 I can find and correct simple semantic errors i.e. debugging, in programs

Maze Challenges 1-5

Algorithmic Thinking

- AL1 Writing instructions that, if followed in a given order, achieve a desired outp

Algorithms

- PA1 I know what an algorithm is and can express simple algorithms using symbols
- PA3 I can sho

JavaScript

- JS1.3 I understand how to use JavaScript functions to achieve my aims

Comprehension Challenges 6-10

Algorithmic Thinking

- AL1 Writing instructions that, if followed in a given order, achieve a desired output

Algorithms

- YA3 I can use logical reasoning to predict outcomes

Javascript

- JS1.3 I understand how to use JavaScript functions to achieve my aims

LEARNING OUTCOMES

The Learning Outcomes mapped in Code Warriors are derived from the Computers At Schools Progression Pathways. More information can be found at the Computing at School website, and on the new Progression Pathways website - http://www.progression-pathways.co.uk. The Learning Outcomes below represent a subset of the full Progression Pathways.

AL1.
The focus here is on understanding that:
- algorithms are step-by-step sets of instructions that, if followed precisely, will achieve a desired output or effect
- a computer program tells the computer what steps to perform, in what exact order, in order to carry out a specific task

Game activities - JUNIOR CODER: Maze Challenges, Comprehension Challenges

E1.
The focus here is on understanding that:
- algorithms are designed to achieve a specific output
- if there is something wrong with the sequence, the algorithm will not execute successfully

Game activities - BEGINNER: Combat Challenges; JUNIOR CODER: Maze Challenges

E2.
The focus here is on understanding that:
- algorithms are designed to achieve a specific output
- if there is something wrong with the sequence, the algorithm will not execute successfully

Game activities - BEGINNER: Stealth Challenges

E4.
The focus here is on understanding that:
- an algorithm is a set of instructions or rules that are followed to complete a task
- algorithms can be improved by removing unnecessary steps, or looking for a more efficient sequence of steps

Game activities - BEGINNER: Stealth Challenges; JUNIOR CODER: Maze Challenges

LEARNING OUTCOMES

E5. Comparing the performance of algorithms that do the same thing
The focus here is on understanding that:

- different algorithms may achieve the same end, but that some algorithms may be more 'efficient' than the others. In Code Warriors, more efficient means fewer Action Points used
- it is good practice to aim for the most efficient algorithm. In Code Warriors the best algorithms will receive three stars, whereas less efficient algorithms will receive one or two

Game activities: CODER and WARRIOR: Comprehension Challenges; Parameter Challenges

PA1. I know what an algorithm is and I can express simple algorithms using symbols
The focus here is on understanding that:

- an algorithm is a set of instructions that, if followed precisely, step-by-step, will achieve an anticipated effect or result
- computer algorithms are written using specific symbols that are part of the programming language - for example JavaScript, which is a language that computers understand.

Game activities - JUNIOR CODER: Maze Challenges, Comprehension Challenges

PA2. I know that computers need precise instructions
The focus here is on understanding that:

- an algorithm can be expressed as a program in many different programming languages
- programming languages are very precise
- computers need more precise instructions than humans
- without precise instructions, computer programs will not perform tasks correctly

Game activities - JUNIOR CODER: Maze Challenges, Debug Challenges

LEARNING OUTCOMES

PA3. I can show care and precision to avoid errors
The focus here is on understanding that:
- programming languages have their own 'vocabulary' and 'grammar' - their syntax.
- precision is key when writing in computer language
- information is presented in a way that the computer can understand

Game activities - JUNIOR CODER: Maze Challenges, Comprehension Challenges

YA1. I know that algorithms are implemented on digital devices as programs
The focus here is on understanding that:
- algorithms are key to the way that computers process information
- algorithms tell computers what specific steps to perform, in what specific order
- computer programs carry out specific tasks

Game activities - BEGINNER: Stealth Challenges; JUNIOR CODER: Maze Challenges

YA3. I can use logical reasoning to predict outcomes
The focus here is on understanding that:
- everything a computer does is controlled by logical instructions
- the outcome of a set of instructions can be predicted
- by reading code, it is possible to work out the result of running that code

Game activities - BEGINNER: Combat Challenges; JUNIOR CODER: Comprehension Challenges

YA4. I can find and correct errors in algorithms. This is called debugging.
The focus here is on understanding that:
- computer programs are written by humans called programmers!
- programmers can make mistakes in their code, and these mistakes are called 'bugs'
- bugs can be found and corrected by searching lines of code for errors
- an important part of programming is testing and 'debugging'

Game activities: CODER and WARRIOR: Debug Challenges; Parameter Challenges

LEARNING OUTCOMES

PP1. I know that users can write their own programs
The focus here is on understanding that:
- computer programs are written by programmers
- computers, phones and games consoles all work because of programming

Game activities - JUNIOR CODER: Maze Challenges, Comprehension Challenges

PP2. I can create a simple program
The focus here is on understanding that:
- the player can program their robots to perform simple activities, Move, Turn, Attack...
- programming involves lines of code written in a specific order, with specific parameters
- computers need to be programmed so that they know what to do

Game activities - JUNIOR CODER: Maze Challenges, Combat and Stealth Challenges

PP3. I can run, check and change programs
The focus here is on understanding that:
- when the player has finished writing their code, they run it to see whether it works
- because precise language is so important, checking code is essential for it to run correctly
- code can be altered at any time, using undo or backspace

Game activities - JUNIOR CODER: Debug Challenges, Parameter Challenges

PP4. I know that programs run by following precise instructions
The focus here is on understanding that:
- computer code is a set of rules or instructions
- code is made up of words and symbols (numbers, brackets, semicolons...)
- code tells your computer what to do

Game activities - JUNIOR CODER: Debug Challenges, Parameter Challenges

LEARNING OUTCOMES

YP2. I can use logical reasoning to predict the behaviour of programs
The focus here is on understanding that:
- everything a computer does is controlled by logic
- the outcome of a sequence of instructions can be predicted
- by reading code, it is possible to work out what steps are being followed

Game activities - BEGINNER: Stealth Challenges; JUNIOR CODER: Comprehension Challenges

YP3. I can find and correct simple semantic errors i.e. debugging, in programs
The focus here is on understanding that:
- computer programmers can make mistakes when writing code
- mistakes in code are called 'bugs'
- bugs can be in syntax or in logic
- a syntax error could be a spelling mistake or use of a wrong symbol
- a logic bug is when a program runs, but doesn't achieve the intended result!

Game activities - JUNIOR CODER: Debug Challenges, Parameter Challenges

JS1.1 I understand how to execute basic JavaScript
The focus here is on understanding that:
- once lines of code have been written, the execute button makes the program run
- Code runs in a specific sequence

Game activities - JUNIOR CODER: Maze Challenges, Combat and Stealth Challenges

JS1.2 Constant parameters: I understand how different parameters affect a function's output
The focus here is on understanding that:
- constant parameters must always appear in a specific form
- in Code Warriors, the parameters 'Right', 'Left', 'Forward', 'Backward' are examples of constant parameters that do not change

Game activities - JUNIOR CODER: Debug Challenges, Parameter Challenges

LEARNING OUTCOMES

JS1.3 I understand how to use JavaScript functions to achieve a desired output
The focus here is on understanding that:

- the right instructions must be completed in the correct order, with appropriate parameters for the desired output to be achieved
- computers execute code very literally, so their programs must be exactly correct to achieve the desired results

Game activities - JUNIOR CODER: Maze Challenges, Combat and Stealth Challenges

JS1.4 Variable parameters: I understand how parameters may affect the output of a function
The focus here is on understanding that:

- parameters can be variable, which means they can change when being executed. For example, numeric parameters can be of different values

Game activities - JUNIOR CODER: Debug Challenges, Parameter Challenges

WHERE NEXT?

To join the growing Code Warriors community and to stay informed of other support materials, updates and other Kuato Studios products, visit us at:

Facebook - Kuato Studios - www.facebook.com/kuatostudios
Facebook - Code Warriors Elite - www.facebook.com/Code Warriors
Twitter - www.twitter.com/kuatostudios
YouTube - www.youtube.com/kuatostudios

For students who have developed an interest in coding through playing Code Warriors, the following resources can help you take things further:

Code Clubs
Code Club is a nationwide network of free volunteer-led after-school coding clubs for children aged 9-11.
https://www.codeclub.org.uk/
http://codeclubworld.org/
CodeDojo:http://coderdojo.com/

Scratch
http://scratch.mit.edu/

Code Academy
http://www.codecademy.com/

Raspberry PI
http://www.raspberrypi.org/

YRS - Young Rewired State:
https://youngrewiredstate.org/

Stemettes:
http://stemettes.org/

IN THE CLASSROOM

LESSON PLANS

The following three session plans have been designed to help educators use Code Warriors in the classroom. They are intended as a general guide only and may be adapted or altered as you wish. By the end of the sessions students should have a basic understanding of the structure of functions.

Session 1

Discussion and recording of students' current awareness of coding, what it is used for and their interest levels regarding it
Introduction to Code Warriors and learning basic gameplay.

Session 2

Completion of Single Player Challenges in small groups to learn the basics of JavaScript
Students present and discuss their coding solutions to the challenges

Session 3

Students engage in multiplayer tournament to determine the class coding champion(s)
Celebration of the winners
Recap of original coding discussion
Signposting to further coding resources

IN THE CLASSROOM

SESSION 1

For students new to programming, the following represents a simple three session course during which the player will be introduced to some rudimentary principles of JavaScript. The sessions culminate in a group Code Warriors tournament. The game sessions should also provide ample opportunities for class discussions around programming and where to learn more.

ACTIVITY	RESOURCES	LENGTH
Introduction: As a group, have a short discussion about code, e.g What is code? Who uses code and why? Why might it be useful to learn how to code? (See additional prompt sheet) Students' responses may be recorded for future discussion.	Whiteboard	5min
Discussion: Explain that over the next three sessions we will be learning more about code by playing Code Warriors. You may wish to show the Code Warriors trailer at this point, which can be found at www.youtube.com/kuatostudios	Code Warriors Video Trailer	5min

SESSION 1 (cont.)

ACTIVITY	RESOURCES	LENGTH
Bootcamp: Code Warriors Bootcamp challenges are designed to take players through the basic rules of the game. At this stage, the game writes the code; students should observe and notice particular features. Classes may be divided into groups or pairs to work through the Bootcamp challenges.	Devices with Code Warriors Elite installed	5min
Plenary Class discussion about their first experience of the game. Students should be encouraged to record their observations, both about the game, and about their exposure to JavaScript. Explain that in the next session they will start coding themselves!	Whiteboard	5min

Differentiation:

Students who complete the Bootcamp challenges quickly can either move on to the next set of 'Combat Challenges' and 'Stealth Challenges' at Beginner rank. Alternatively, they can retry the 'Bootcamp' section using the Junior Coding rank.

IN THE CLASSROOM

SESSION 2

In this session, students work in pairs or small groups to undertake Code Warriors the single player level in Junior Coding rank. At this level, the game will provide auto-complete help. However, players must type the code themselves, using the Function Library as reference. By the end of this session students should have a basic understanding of the structure of functions and how to alter string and numerical data.

ACTIVITY	RESOURCES	LENGTH
Introduction: Recap of previous session 'Welcome to Code Warriors'. Explain to the students that in this session, students will play in Junior Coder rank, i.e., they will start to type code for themselves. Auto-Complete will provide support.	N/A	5min
Combat Challenges: Divide the students into pairs or small groups. Students should work through the Combat Challenges by first planning their moves, writing code, debugging and then executing. Code Warriors will award up to 3 stars for successfully completed challenges based on how efficient the solution was in its use of Action Points.	Device with Code Warriors Elite Installed	30min

SESSION 2 (cont.)

ACTIVITY	RESOURCES	LENGTH
Evidencing: Once students have found their best solution for a challenge they should take a screenshot of their code and store it as a record of their work.	N/A	10min
Plenary Bring the students back together as a class. Ask each group to present a solution to a challenge. Discuss the structure of the code they have been writing. What observations have they made? Draw out from this discussion the difference between numerical and string data.	Devices with Code Warriors Elite installed	5min

Differentiation:

Students who find the challenges difficult can try planning their moves first using the 'Beginner' mode. In this mode, the code is written for the player as they tap on the board. Students should observe the code that is generated in this mode and try to replicate it in Junior Coder rank. Students who complete all the challenges quickly can attempt to optimise solutions to the challenges by using fewer Action Points. Alternatively, students can try the challenge again in a more advanced coding mode, such as 'Coder' or Warrior'.

IN THE CLASSROOM

SESSION 3

In this final session, classes will use the Multiplayer feature of Code Warriors to hold a group tournament. Awards may be given for winning the game and for the coding style used.

ACTIVITY	RESOURCES	LENGTH
Introduction: Explain that in this session we will be putting our coding knowledge to the test with a multi player Code Warriors tournament.	N/A	5min
Explore the ChopShop: Students may be given time to customise their Code Warrior ready for battle. **Tournament:** Students should then play each other in groups to determine an overall champion. The structure of the tournament will depend on class size and length of session.	Device with Code Warriors Elite Installed	30min

SESSION 3 (cont.)

ACTIVITY	RESOURCES	LENGTH
Celebration: Bring the class together as a group and award small prizes (e.g. Code Warriors stickers). Recognition may be given to match winners and to the team who played in the hardest coding style.	Stickers or other small prizes	10min
Recap: The class may like to review their responses from Session 1. What have they learned through the three sessions?	Discussion notes from Session 1	15min
Signposting: Depending on your class, talk through suitable sources for further exploration of coding. See the 'Further Coding' factsheet for suggestions.	'Where Next' factsheet (page X)	10min

Differentiation:

Students should be encouraged to play the tournament in Junior Coder style or above, but students working at a slower pace can use Beginner if needed. Advanced students can attempt to play in one of the harder styles such as Coder or Warrior.