Unit 2.1: We are astronauts

Programming on screen in ScratchJr

Software: ScratchJr (alternative: Scratch)
Hardware: iPads (alternatives: Android tablets, laptop/desktop/Chromebook computers, Bee-Bots, Blue-Bots)

Overview

In this unit, pupils program a **sprite** (such as a spaceship) to move around the screen. In:

- **Session 1** they take part in playground activities, planning movement between 'planets'
- **Session 2** they are introduced to **ScratchJr** and **program** sprite movement
- **Session 3** they are introduced to **output** and use multiple sprites
- **Session 4** they are introduced to message passing and **input**
- **Session 5** they are introduced to **repetition**
- **Session 6** they create new 'costumes' for their sprites.

Alternatives

The unit sessions give step-by-step guidance on using ScratchJr. ScratchJr is a free app for iPads or Android tablets and Chromebooks. However, this unit could also be carried out using Scratch, Bee-Bots or Blue-Bots.

Knowledge, skills and concepts

In this unit, pupils will learn to:

- plan a sequence of instructions to move **sprites** in **ScratchJr**
- create, test and **debug** programs for sprites in ScratchJr
- work with **input** and **output** in ScratchJr
- use **repetition** in their **programs**
- design costumes for sprites.

Progression

In Key Stage 1:

- Pupils programmed Blue-Bots in **Unit 1.1: We are treasure hunters** and programmed on-screen in ScratchJr in **Unit 1.5: We are rhythmic**.
- Pupils will be introduced to Scratch in **Unit 2.2: We are games testers** as they explore some pre-built games and work out the rules.

In Key Stage 2:

- **Unit 3.1: We are programmers** continues pupils' programming development as they make a scripted animation in Scratch.

Assessment – by the end of the unit:

All pupils can:

- plan a route from one hoop to another in the playground
- create a sequence of move instructions on screen
- record audio and add an instruction to play audio
- create a costume for a **sprite**.

Most pupils can:

- plan a return route in the playground
- create multiple sequences of move instructions
- add instructions to display a sequence of texts
- use different events to launch **code**
- create costumes for multiple sprites.

Some pupils can:

- plan a route visiting multiple hoops in the playground
- predict correctly what a sequence of instructions will do
- have events launch multiple programs in parallel
- use internal messages to control the behaviour of sprites
- use repetition in their programs
- create a background scene.

Background information

- The building-block approach for **programming** in **ScratchJr** is very flexible and will characterise how pupils program throughout Key Stage 2 and beyond into Key Stage 3.
- The programs that pupils write involve creating sequences of instructions for '**sprites**' – on-screen characters that can be independently controlled. Each sequence of instructions is carried out when a particular **event** occurs, such as the green flag being clicked, or sprites being tapped or interacting with each other.
- Pupils are introduced to a simple model of how computers operate:
  - They accept **input** (iPad taps).
  - They follow sequences of instructions.
  - They produce **output** (in this case, movement on screen, but also text and recorded audio).
- Pupils are also introduced to the idea of **repetition:** that computers can be programmed to run the same **code** repeatedly.
- In programming, encourage pupils to adopt the plan – code – predict – test – **debug** sequence, which is recommended in this unit.
  - No matter how user-friendly the programming environment, it is worth pupils having a clear idea of what they want the computer to do before they start building their programs; it can be helpful to sketch these ideas.
  - Programmers frequently make mistakes – called **bugs** – and much time can be spent fixing these. By first predicting what the program should do, and then testing it, you encourage pupils to compare the predicted and actual behaviour, making it a little easier to find out where the program has gone wrong.

Key vocabulary

Abstraction: computational thinking approach to managing complexity by simplifying things through identifying what is important, and what detail can be hidden

Algorithm: a sequence of precise instructions or steps (sometimes a set of rules) to achieve an objective

Bug: an error or mistake in a program or algorithm, causing the computer or robot to behave in a manner that was not originally intended

Code: instructions (or sometimes rules) that can be understood by a computer

Debug: correct mistakes in a program or algorithm

Event: something that happens within a computer program to cause some particular code to be run, such as an internal message being received or a sprite being tapped by the user

Input: data supplied to a computer, in this case, tapping on the screen of a tablet

Output: information produced by a computer – in this case, moving sprites on a screen, text and audio

Parallel processing: when programs run (or appear to run) simultaneously

Program: sequence of instructions (or sometimes a set of rules) that can be followed by a computer

Repetition: programming construct which allows a group of instructions to be repeated a number of times, or until a certain condition is met

Scratch: simple, block-based programming language in which programs for characters are built by snapping together code blocks

Sprite: a graphical character in a program that can be given its own sequence of instructions

Differentiation

See each session (pages 13–18) for ways to increase support and add challenge to this unit.

The ideas in this unit can be explored more simply using Blue-Bots or other floor turtles, or at a more sophisticated level using **Scratch**.

Pupils learning EAL can view the source code of Scratch games in their first language using the globe icon in the Scratch editor.

Cross-curricular opportunities

English: Pupils could explore some aspects of space travel through reading and creative writing.

History: Includes references to early space travel.

Maths: Pupils use simple arithmetic to work out how far the sprites must move, and use the language of position, movement and estimating distances. They think about how to approximate circular movement on a coordinate grid.

Science: The unit could be extended as part of a wider, cross-curricular topic exploring space.

Preparation for teaching the unit

Things to do

- Check you have access to **ScratchJr** (or the alternative you are using), and that it is available on the tablets or Chromebooks. You will need to be able to share your tablet screen with the class by connecting it to the display screen/interactive whiteboard, or use the walkthroughs provided.
- Read pages 10–11 to get an overview of the unit.
- Read the steps in the unit sessions (pages 13–18) and look at the associated online resources, printing out the worksheets as required.
- Work through the unit yourself so you know what is expected of the pupils.
- Make sure the tablets or Chromebooks are labelled in some way, as pupils will need to use the same devices throughout the unit.

Resources needed

- **Software:** ScratchJr
- **Hardware:** iPads
- See *Alternatives* on page 10

Online resources provided

Session resources

- Worksheet 2.1a: Instruction cards
- Worksheet 2.1b: End-of-unit quiz
- Worksheet 2.1c: Pupil self-assessment
- Teaching slides: 2.1a–2.1f
- Walkthrough videos: 2.1a–2.1f
- Interactive end-of-unit quiz 2.1

Additional resources

- CPD video: Instructions for sprites

Online safety

- ScratchJr is an offline **programming** environment and therefore has few online safety risks associated with it. Be vigilant about other apps, including the web browser, which pupils have access to while using tablets and ensure that the usual filters and monitors for Internet access are in place.
- Pupils can be encouraged to experiment with ScratchJr independently if they have compatible devices at home. Remind parents/carers about their responsibility to monitor their children's use of technology, and advise them to set sensible limits on the amount of screen time pupils have.

Collaboration

Session 1 involves group work on the playground or in the school hall. Mixed ability pairs can work well for this, if pupils are briefed on their responsibility to ensure that both partners contribute fairly to the project work and both understand the ideas involved. You might find it helpful to ring a bell every few minutes for partners to switch 'driver' and 'navigator' roles. Subsequent sessions on programming use pair work.

Useful links

Software and tools

- ScratchJr: **www.scratchjr.org**

Online tutorials

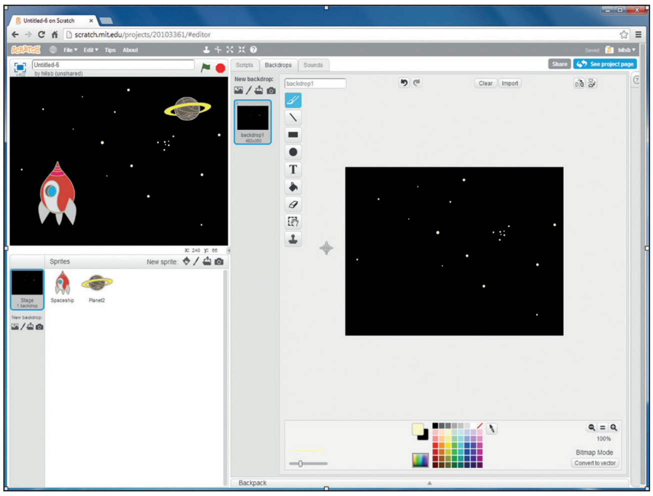
- ScratchJr: **www.scratchjr.org/learn/interface**
- ScratchJr introduction: **www.youtube.com/watch?v=tEWFDJSmWcw**

Information and ideas

- Apollo 11 launch sequence video: **www.youtube.com/watch?v=UExTN3_UOIY**
- Apollo 11 guidance computer error video: **www.youtube.com/watch?v=z4cn93H6sM0I**
- Apollo 13 manual burn clip: **www.youtube.com/watch?v=Wm628c3sgt8**
- Solar system animation: **www.youtube.com/watch?v=z8aBZZnv6y8**
- Audio and video of famous moments in space travel from NASA: **www.history.nasa.gov/40thann/videos.htm**
- Teaching ideas: **www.scratchjr.org/teach/activities**
- PBS Kids on space exploration in ScratchJr: **www.pbskids.org/learn/scratchjr/activities/space-exploration**

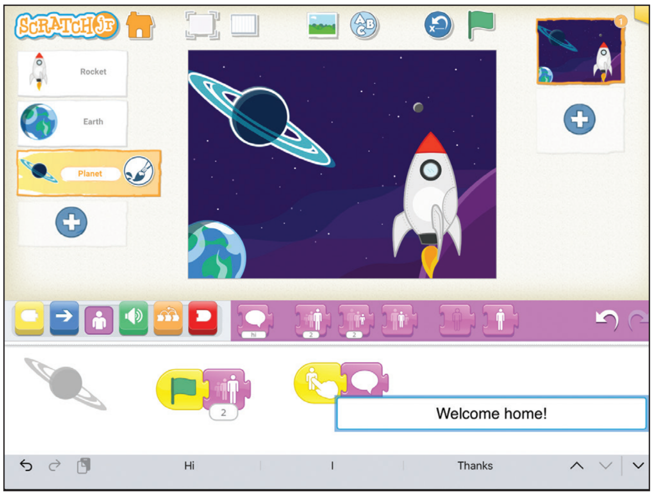
Unit outcomes

Below are some examples of the outcomes you could expect from this unit.

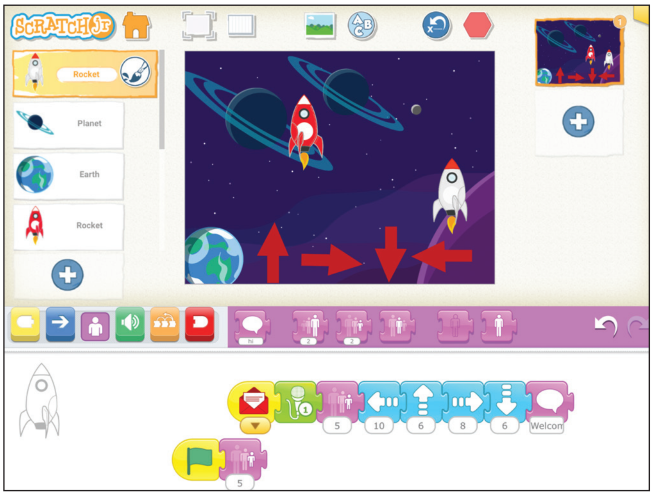


Session 1: Planning the algorithm needed to move from one planet (hoop) to another

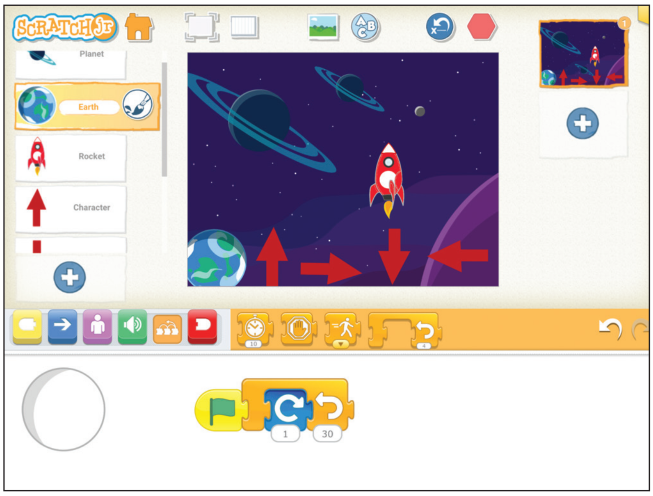
Session 2: Programming a spacecraft navigation program in ScratchJr



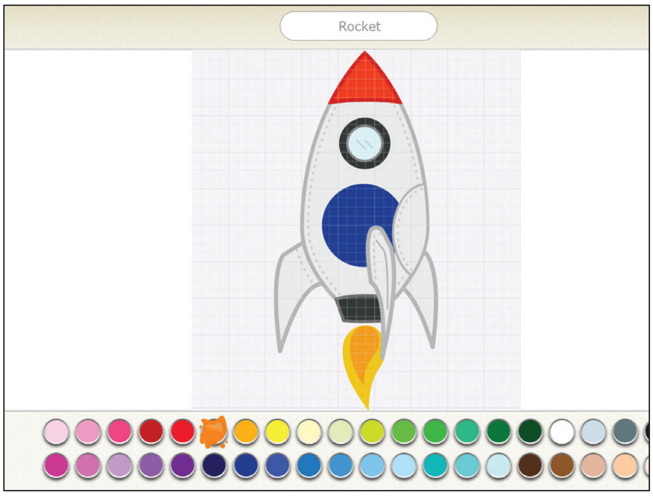
Session 3: Adding audio and messages to ScratchJr programs



Session 4: Adding control sprites to programs in ScratchJr



Session 5: Adding a repeating code to programs in ScratchJr



Session 6: Using the paint editor in ScratchJr