Year 1 – Programming B – Programming animations

Unit introduction

Learners will be introduced to on-screen programming through ScratchJr. Learners will explore the way a project looks by investigating sprites and backgrounds. They will use programming blocks to use, modify, and create programs. Learners will also be introduced to the early stages of program design through the introduction of algorithms.

Software and Hardware requirements

All the lessons in this unit require access to ScratchJr. ScratchJr is available as an App for tablets and Chromebooks. An unofficial version of ScratchJr is available for computers (Scratch Jr for Desktop - Open Source Community Port).

If you've adapted this unit to better suit your school, please share your adapted resources with fellow teachers in the STEM community. Alternatively, if this unit isn't quite right for your school, why not see if an adapted version which better suits has already been shared?

Overview of lessons

Lesson	Brief overview	Learning objectives
1 Comparing tools	During this lesson learners will become accustomed to the ScratchJr programming environment. They will discover that they can move characters on-screen using commands, and compare ScratchJr to the Bee-Bots used in the previous unit.	To choose a command for a given purpose • I can find the commands to move a sprite • I can use commands to move a sprite

		• I can compare different programming tools
2 Joining blocks	During this lesson learners will discover that blocks can be joined together in ScratchJr. They will use a **Start** block to run their programs. They will also learn additional skills such as adding backgrounds and deleting sprites. Learners will follow given algorithms to create simple programs.	To show that a series of commands can be joined together • I can use more than one block by joining them together • I can use a **Start** block in a program • I can run my program
3 Make a change	During this lesson learners will discover that some blocks in ScratchJr have numbers underneath them. They will learn how to change these values and identify the effect on a block of changing a value.	To identify the effect of changing a value • I can find blocks that have numbers • I can change the value • I can say what happens when I change a value
4 Adding sprites	During this lesson learners will be taught how to add and delete sprites in ScratchJr. They will discover that each sprite has its own programming area, and learn how to add programming blocks to give instructions to each of the sprites.	To explain that each sprite has its own instructions • I can show that a project can include more than one sprite • I can delete a sprite • I can add blocks to each of my sprites
5 Project design	During this lesson learners will choose appropriate backgrounds and sprites for a 'Space race' project. They will decide how each sprite will move, and create an algorithm based on the blocks available in ScratchJr that reflects this.	To design the parts of a project • I can choose appropriate artwork for my project

		• I can decide how each sprite will move • I can create an algorithm for each sprite
6 Following my design	During this lesson learners will use their project designs from the previous lesson to create their projects on-screen in ScratchJr. They will use their project design, including algorithms created in the previous lesson, to make programs for each of their rocket sprites. They will test whether their algorithms are effective when their programs are run.	To use my algorithm to create a program • I can use sprites that match my design • I can add programming blocks based on my algorithm • I can test the programs I have created

Subject knowledge and CPD opportunities

The unit focuses on developing learners' understanding of computer programming. It highlights that algorithms are a set of clear, precise, and ordered instructions, and that a computer program is the implementation of an algorithm on a digital device. The unit also introduces reading 'code' to predict what a program will do. Learners will engage in aspects of program design, including outlining the project task and creating algorithms.

When programming, there are four levels that can help describe a project, known as levels of abstraction. Research suggests that this structure can support learners in understanding how to create a program and how it works:
- Task – what is needed
- Design – what it should do
- Code – how it is done
- Running the code – what it does

Spending time at the 'task' and 'design' levels before engaging in code writing aids learners in assessing the achievability of their programs, and reduces a learner's cognitive load during programming.

Learners will move between the different levels throughout the unit.

Continual Professional Development

Enhance your subject knowledge to teach this unit through the following free CPD:

- [Getting started in Year 1 – short course](#)
- Introduction to primary computing [remote](#) or [face to face](#)
- [Teaching programming using Scratch and ScratchJr](#)
- [Teaching programming to 5- to 11-year-olds](#)

Teach primary computing certificate

To further enhance your subject knowledge, enrol on the [teach primary computing certificate](#). This will support you to develop your knowledge and skills in primary computing and gain the confidence to teach great lessons, all whilst earning a nationally recognised certificate!

Progression

This unit progresses learners' knowledge and understanding of programming and follows on from 'Programming A – Moving a robot', where children will have learned to program a floor robot using instructions.

Common misconceptions

This is likely to be the first time pupils have experienced on-screen programming, so addressing the difference between floor robots and on-screen programming is important to prevent misconceptions forming. In Scratch Jr, a sprite can move forwards, backwards, up and down. The turn block changes the sprites appearance, but not how the sprite moves. This is different to a Bee-bot, which can only move forwards and backwards, and which turns on the spot.

Curriculum links

Computing

- Understand what algorithms are, how they are implemented as programs 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

Maths

Measure

- Sequence events in chronological order using language [for example, before and after, next, first, today, yesterday, tomorrow, morning, afternoon and evening]

Geometry - position and direction

- describe position, direction and movement, including whole, half, quarter and three-quarter turns

Assessment

Formative assessment

Assessment opportunities are detailed in each lesson plan. The learning objective and success criteria are introduced in the slide deck at the beginning of each lesson and then reviewed at the end. Learners are invited to assess how well they feel they have met the learning objective using thumbs up, thumbs sideways, or thumbs down.

Summative assessment

Please see the assessment rubric document for this unit. The rubric can be used to assess learning and highlights whether the pupil is approaching (emerging), achieving (expected), or exceeding the expectations in this unit.

Resources are updated regularly — please check that you are using the latest version.

Attribution statement

This resource was created by Raspberry Pi Foundation and updated by STEM Learning for the National Centre for Computing Education.

The contents of this resource are available for use under the Open Government License (OGL v3) meaning you can copy, adapt, distribute and publish the information. You must acknowledge the source of the Information in your product or application, by attributing Raspberry Pi Foundation and STEM Learning as stated here and are asked to provide a link to the OGL v3.

The original version can be made available on request via info@teachcomputing.org.