

Year 6 – Programming A – Variables in games

Unit introduction

This unit explores the concept of variables in programming through games in Scratch. First, pupils will learn what variables are, and relate them to real-world examples of values that can be set and changed. Pupils will then use variables to create a simulation of a scoreboard. In Lessons 2, 3, and 5, which follow the Use-Modify-Create model, pupils will experiment with variables in an existing project, then modify them, then they will create their own project. In Lesson 4, pupils will focus on design. Finally, in Lesson 6, pupils will apply their knowledge of variables and design to improve their game in Scratch.

There are two Year 6 programming units:

- Programming A – Variables in games
- Programming B – Sensing

This is unit A, which should be delivered before unit B.

Overview of lessons

Lesson	Brief overview	Learning objectives
1 Introducing variables	In this lesson, pupils will be introduced to variables. Pupils will see examples of real-world variables (score and time in a football match), then they will explore them in a Scratch project. Pupils will then design and make their own project including variables. Finally, pupils will identify that variables are named and can be letters (strings) as well as numbers.	To define a ‘variable’ as something that is changeable • I can identify examples of information that is variable • I can explain that the way that a variable changes can be defined

		• I can identify that variables can hold numbers or letters
2 Variables in programming	In this lesson, pupils will understand that variables are used in programs, and that they can hold a single value at a time. Pupils will complete an unplugged task that will demonstrate the process of changing variables. Next, they will explore why it is important to name variables, then they will apply their learning in a Scratch project in which they will make, name, and update variables.	To explain why a variable is used in a program • I can identify a program variable as a placeholder in memory for a single value • I can explain that a variable has a name and a value • I can recognise that the value of a variable can be changed
3 Improving a game	In this lesson, pupils will apply the concept of variables to enhance an existing game in Scratch. They will predict the outcome of changing the same change score block in different parts of a program, then they will test their predictions in Scratch. They will also experiment with using different values in variables, and with using a variable elsewhere in a program. Finally, they will add comments to their project, explaining how they have met the objectives of the lesson.	To choose how to improve a game by using variables • I can decide where in a program to change a variable • I can make use of an event in a program to set a variable • I can recognise that the value of a variable can be used by a program
4 Designing a game	This lesson focuses on the design elements of programming. For the majority of the tasks, pupils will be working at the algorithmic level of abstraction. Pupils will first design the sprites and backgrounds for their project, then they will design their algorithms to create their program flow.	To design a project that builds on a given example • I can choose the artwork for my project

		• I can explain my design choices • I can create algorithms for my project
5 Design to code	In this lesson, pupils will implement the algorithms that they created in Lesson 4 as code. In doing this, they will identify variables in an unfamiliar project and learn the importance of naming variables. They will also have the opportunity to add another variable to enhance their project.	To use my design to create a project • I can create the artwork for my project • I can choose a name that identifies the role of a variable • I can test the code that I have written
6 Improving and sharing	This lesson gives pupils the opportunity to build on the project that they created in Lesson 5. As the lesson develops, the scaffolding is gradually removed, so that the last main activity is without constraints. Finally, pupils will evaluate each other's projects, identifying features that they like, and features that could be improved further.	To evaluate my project • I can identify ways that my game could be improved • I can extend my game further using more variables • I can share my game with others

Progression

This unit assumes that pupils will have some prior experience of programming in Scratch. Specifically, they should be familiar with the programming constructs of sequence, repetition, and selection. These constructs are covered in the Year 3, 4, and 5 National Centre for Computing Education programming units respectively. Each year group includes at least one unit that focuses on Scratch.

Please see the learning graph for this unit for more information about progression.

Curriculum links

National curriculum links

- Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts
- Use sequence, selection, and repetition in programs; work with variables and various forms of input and output
- Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs

Assessment

Formative assessment

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

We recommend the use of teacher accounts in Scratch to help with assessment throughout this unit. For guidance on setting up teacher accounts, please [visit the Scratch website](https://scratch.mit.edu/educators/faq) (scratch.mit.edu/educators/faq).

Summative assessment

Please see the assessment question and answer documents for this unit.

Subject knowledge

This unit focuses on developing pupils' understanding of variables in a new programming language. It highlights where variables can be used and how they can be set and changed through the running of a program. This unit also develops pupils' understanding of design in programming, using the approach outlined below.

When programming, there are four levels that can help describe a project (known as 'levels of abstraction'). Research suggests that this structure can support pupils 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 writing code can aid pupils in assessing the 'do-ability' of their programs. It also reduces the cognitive load for pupils during programming.

Pupils will move between the different levels throughout the unit, and this is highlighted within each lesson plan.

During this unit, pupils are required to save their work in Scratch. We recommend the use of teacher and pupil accounts to manage this process. You can find detailed guidance on setting up and managing accounts in Scratch on the [Scratch website](https://scratch.mit.edu/educators/faq) (scratch.mit.edu/educators/faq).

Enhance your subject knowledge to teach this unit through the following training opportunities:

Online training courses

- [Raspberry Pi Foundation online training courses](#)

Face-to-face courses

- [National Centre for Computing Education face-to-face training courses](#)

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

This resource is licensed under the Open Government Licence, version 3. For more information on this licence, see [ncce.io/ogl](https://www.ncce.io/ogl).