# Introduction to variables

In this tutorial, you’ll:

* Learn what a variable is and why it is useful
* Identify a variable within a script
* Identify the value assigned to a variable
* Practice reading instructions in a script

When you’ve finished, you’ll be ready to create your own variable.

## [**Working with the Creator Kit: Beginner Code scripts**](https://learn.unity.com/tutorial/introduction-to-variables?uv=2020.3&projectId=5d48025dedbc2a0020e85be2#5d4c242eedbc2a001f636ec8)

In this tutorial, you’ll identify specific instructions in the SpawnerSample script.

Remember:

* You won’t be able to understand everything in the script straight away.
* It may take you a little time to find information in the file.

This is completely normal. In this Creator Kit, you’ll learn how to approach an unfamiliar script and identify key elements, rather than how to understand everything in game-specific scripts.

## [**Find an instruction in the script**](https://learn.unity.com/tutorial/introduction-to-variables?uv=2020.3&projectId=5d48025dedbc2a0020e85be2#5d4c242eedbc2a001f636ec9)

In your code editor, find the **10th line** of the script:

int angle = 15;

This is the base building block of programming: an instruction. An instruction is a single task you give to the computer.

The instruction ends with a semicolon (;). This tells the compiler that the instruction is finished and the computer should now complete it. Compilers don’t care about the ends of lines — a semicolon is what signifies the end of an instruction for them. However, humans also need to be able to read C# code too; to improve readability, it’s good practice to put each instruction on a separate line.

Before you progress to the next step, consider:

* The purpose of this script in the game
* The word ‘angle’

What could this instruction be telling the computer to do?

## [**Introduction to variables**](https://learn.unity.com/tutorial/introduction-to-variables?uv=2020.3&projectId=5d48025dedbc2a0020e85be2#5d4c242eedbc2a001f636eca)

The instruction int angle = 15; declares and assigns a variable.

A variable is a container that stores a value. You can think of it like a trunk with a name written on it, where you can store information you want to retrieve later. As its name suggests, the information stored in the variable can change (or vary).

Before you can use a variable, you need to declare it by stating its name and type. You can also assign it an initial value.

In the image below, a variable (trunk) called gems has been assigned the value 2.



## **Identify the type and name of the variable**

There are two things you need to identify when declaring (or creating) a variable:

1. The type of information that will be stored, which tells the computer how much space to save in its memory to store the information. All variables need a type.

This is an int variable. These variables store integers (whole numbers).

2. The name of the variable, which you will use every time you want the computer to retrieve the variable’s value.

The name of this variable is angle.

## **Identify the value assigned to the variable**

You can assign a value to a variable using the equals sign (=). This means that you copy the value to the right of the equals sign into the variable on its left.

int angle = 15; tells the computer to store the integer value 15 inside the variable called angle.

Every time you use the variable angle in your code, the computer will substitute the integer value.

## **Identify where the variable is assigned a new value**

In the SpawnerSample script, find the next two lines where the int variable angle is assigned a different value.

In each line, what is different from the first time the variable was assigned a value?

## **Review the two instructions**

The angle variable is assigned different values on lines 17 and 22 of the SpawnerSample script.

Unlike the first time a value was assigned to angle, its type is not included. This is because the computer already knows that the variable exists and how much memory space it needs.

Since instructions in a script are sequential (completed one at a time), the changed value for the variable will only be used in instructions after it has been assigned.

In the image below, the example variable gems has now been assigned the value 4.



# Summary

In this tutorial, you learnt:

* That scripts are made up of instructions
* That variables are containers for values in code
* How to declare and assign a variable
* That one variable can have different assigned values within the same script, because the computer executes instructions sequentially

In the next tutorial, you’ll apply your learning and write an instruction to declare and assign a new variable.