**KS2 Use Selection in Programs: Kodu Game**

**Recommended Year Group:** Any KS2  
**Activity Duration:** 1 hour 10 mins+  
Cross curricular links: [Art](#)

---

### Concepts and approaches

**Programming**  
**Selection**

---

### Overview

In this activity pupils create a simple game in Kodu. Pupils start by creating a design for their game, which includes rule-based algorithms describing how it will be played and a sketch of the Kodu world it will be played in. Pupils then create the Kodu world, implement their algorithms as code and play and evaluate each others’ games.

### Pupil objectives

- I can design a game
- I can code a game in Kodu
- I can use selection in Kodu

### Before you start

- If required, view the [Introduction to Kodu video](#).
- Pupils should be familiar with Kodu – this could be by completing [Kodu tinkering](#).

---

### Introduction

- Lead a discussion to explore what computer games pupils enjoy playing and why.
- Explain that that in this lesson pupils will be designing and coding their own game and that they will get the chance to play, and evaluate, each others’ games.
- Ask pupils what we could use to create a computer game? A *programming language such as Scratch or Kodu*. Explain that in this lesson we will be using Kodu – what can pupils recall about this programming language? *Guide a discussion to elicit pupils’ current understanding.*
- Introduce the learning objectives on slide 2 of the presentation, if this is your normal practice.

### Main activity: designing the game (15+ mins)

- Ask pupils to think about the computer games they like to play – what features are common to many of these? Feedback as a class and construct a class list *e.g: a character which the player controls, objects/characters in the game which must be collected/avoided, a scoring system, a way to win or lose the game.*
- Explain that pupils’ will be creating single-player games in which the player controls a bot and has to collect and avoid objects/bots in their world.
Hand out the game design sheets. Talk through the example game design on slide 3 highlighting pupils must include details of:
- The main bot which is controlled by the player.
- The other objects/bots in the game.
- Rules (algorithm) describing what happens when the main bot touches the other objects/bots (e.g. How does it change the score? Does it end the game?)
- Rules (algorithm) about how the game is won or lost.

Give pupils 10 mins to complete their game design sheets. Note: pupils will need to look at the menus in Kodu to remind themselves of the bots/objects available. If required, invite a pupil to demonstrate how these are accessed before pupils start their designs.

Main activity: Creating a Kodu world (20 mins+)
- Open Kodu and ask pupils to do the same.
- Invite a pupil to recap for the class the tools used to create new worlds in Kodu: paintbrush, hills/valleys tool etc.
- Explain pupils’ next task is to design a world in which their game will be played. Pupils should have 10 mins to sketch their world in their art sketch books. They should annotate their sketches, including details about the shape and colours of features in their land. More confident pupils might make several sketches and select a favourite – or create a final design which is an amalgamation of their ideas.
- After they have completed their sketches, give pupils time (10+ mins) to create their world in Kodu.

Main activity: Coding their game (20+ mins)
- Invite a pupil to recap how to add objects and program these within Kodu.
- Highlight to pupils that when we program in Kodu, commands are placed in ‘When… Do…’ frames, as shown below. Can pupils recall what we call it when commands are run ‘When’ a condition is met? Selection.

Commands are placed in ‘When…Do…’ frames within Kodu

Give pupils 20 + mins to code their games. If pupils require additional support, you may choose to lead a discussion with the class on the code required to get their main bot to move using the arrow keys, as shown below.

The code required to move a character using the arrow keyboard keys

Can pupils recall why it is important to regularly run their program as they are coding it? To check it for bugs and debug it if required.
Optional mini-plenaries:

- **Common bugs:** Lead a discussion to explore the bugs pupils have encountered whilst coding their game. Where there any that many pupils encountered? Why was this? Did all pupils fix them in the same way? What have pupils learnt from fixing these bugs?
- **Design review:** Give pupils the opportunity to review their designs and make any changes based on their experience of starting to code the game. What have they changed? Why have they made these changes?
- Invite a selection of pupils to share their games as they are coding them. Encourage pupils to explain which commands they have used and why. Can pupils help explain to less confident pupils the code they have used?

Plenary (10 mins)

- Give pupils to the opportunity to play a selection of each other’s games. As they do they should take turns in evaluating the games by saying two things they like about the game, and 1 thing they think would improve it. They can use the class list of the features of computer games to help them in this evaluation.
- Provide pupils who completed the extension activity an opportunity to share what they learnt with the class.
- **Optional:** by setting up an account on the Kodu website (http://www.kodugamelab.com) pupils could upload their games and these can be downloaded and played by pupils from across the world.

Differentiation

- **Support:** A Kodu command help sheet has been provided to support less confident pupils. This includes details of the commands which may be useful in coding their game, and hints on how they can be used. Less confident pupils may also work in small group with additional adult support if available.
- **Stretch and challenge:** Challenge pupils to add additional bots to their game which are controlled by the computer. These might be evil bots which chase the player for example. An extension challenge sheet has been provided to support pupils with this challenge.

Assessment opportunities

- Informal, teacher assessment of progress during main task, class discussions and plenary. Key pupil knowledge and skills to identify:
  - Can pupils write the algorithms for their game?
  - Can pupils implement their algorithms as code within Kodu?
  - Can pupils test and debug their games in Kodu?
  - Can pupils evaluate each other’s games?
- Formal, summative assessment of Kodu games if required.
- Peer assessment and feedback on each others’ games during the plenary.
Teaching Notes

Concepts and approaches

**Programming**
In this activity pupils create a simple game in Kodu. In doing so they use their developing knowledge of the Kodu programming language to implement the algorithms they created as they designed their games. Programming is the process of implementing an algorithm as code.

**Selection**
Selection is used to write algorithms and programs which can select which commands to run depending on particular conditions or events. Whether commands run depends on whether certain conditions are met. In this activity pupils' programs use selection as the commands in Kodu are placed in ‘When… Do…’ frames i.e. the commands after ‘Do’ are run ‘When’ the preceding condition is met.

Curriculum links
**Computing:**
- use selection in programs, design
- write and debug programs that accomplish specific goals

**Art:**
- create sketch books to record observations and use them to review and revisit ideas
- improve mastery of art and design techniques, including drawing.

Resources (download from [webpage](#))
- Lesson presentation
- Game design sheet
- Kodu command help sheet
- Kodu challenge sheet

Related activities / further links
**KS2: Kodu Tinkering**
**Designing games with Kodu lab**