Published on *Le site de la Fondation La main à la pâte* (https://www.fondation-lamap.org)

1, 2, 3, code ! - Cycle 3 activities - Sequence II: Simulate the mission in Scratch

This sequence is dedicated to **programming** a video game to simulate our space exploration mission, following on from what was done in Sequence I.

Sequence II is mainly carried out on the computer. However, there are a few **unplugged exercises** (not using the computer) aimed at fostering understanding of certain concepts such as variables and logical operators. These unplugged exercises should ideally take place outside of time dedicated to programming (e.g., math or English classes), so as to avoid interrupting the project.

Some tips before beginning a programming activity

Programming a video game is highly motivating for students, but a few precautions are necessary to ensure it goes well.

Sc?ratch, an ideal environment for learning to code

There are a number of tools available for learning programming. We have chosen to use the software ***Scratch*** on the basis of its remarkable quality, its simplicity of use, it being free of charge, and it having a very active community, including in the education field (elementary and middle school). *Scratch* is available either online (without prior installation, but this requires a good Internet connection, at the address: https://scratch.mit.edu) or offline (requiring prior installation, after downloading the software at the address: https://scratch.mit.edu/scratch2download).

This choice is, however, arbitrary. Teachers can teach the same concepts using another environment (such as *Snap* which is very similar to *Scratch* and is also free, or *Kodu* and *Tangara* which must be purchased).

We also propose an alternative to *Scratch* “alone” in classes that have a robot whose programming uses the same concepts but applied to a physical object. The robot *Thymio*, for example, is programmed using visual programming languages like Aseba/VPL and/or *Scratch* (in this regard, see Cycle 2, Sequence III).

Ensure mastery of some basic ICT skills

Programming requires interaction with a computer, meaning some basic skills are needed:

- Using **keyboard and mouse** (this is generally, but not always, the case taught from grade nine)
- Launching programs by double-clicking on their icon
- Saving work in a file, and saving the file in a folder
- Opening work saved earlier

If certain students do not have these basic skills, they can learn them through this programming project but may end up falling behind during the first exercises.

Working in half-groups

Ideally, there should be **one computer for two students** (3 at most). To make that possible, and so as to make it easier to manage classes during programming activities (during which teachers are very busy), we advise working in half-groups: half of the class programs while the other half works on something else independently.

Preparing the working environment

In order to save time, it is useful to prepare the working environment in advance:

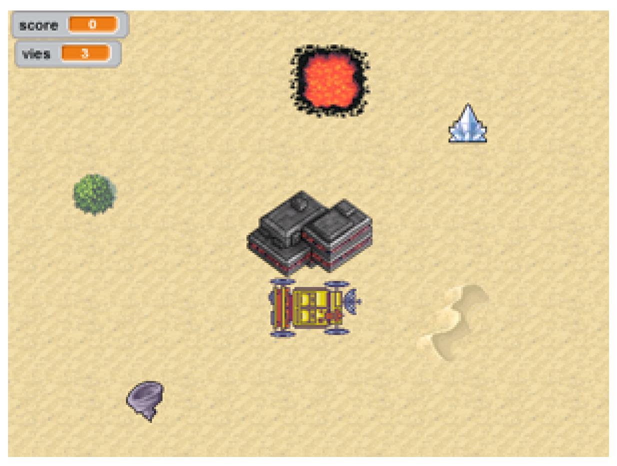
- *Scratch* needs to be installed on all computers (or accessible online, see footnote on previous page)
- A **shortcut** to *Scratch* should be placed on the desktop
- Similarly, a dedicated folder for the project (and class) should be easily accessible, either on the desktop or on a USB flash drive for the group. This folder should contain the files needed for the project (images to import, copies of programs, etc.). We will provide all the files required on the website.
  - The most advanced users will be able to put the useful files directly into the sub-folders of the *Scratch* application for example:

    <*Scratch*>/Medias/Costumes/MissionMars and <*Scratch*>/Medias/Background/MissionMars, where <*Scratch*> is the application’s installation path.

Doing the project yourself first

This is probably the most important tip, even if it seems obvious. It is essential for the teacher to take two or three hours of preparation time BEFORE the first class exercise to get used to *Scratch* and carry out the tasks that the students will have to perform during the project. Otherwise, they may not be able to help the students when they need it. This is not difficult (you can follow the process described in this sequence), and it is even quite fun!

Note : vous pouvez **tester le jeu** en ligne (ou en faire la démonstration aux élèves), en cliquant sur l'image ci-dessous. Récoltez de l'eau et de la nourriture (végétaux) tout en évitant les pièges !



Steps for the project

Please note: In most Cycle 3 classes, carrying out the whole project **will require six or seven one-hour sessions**. Some pairs will produce a more comprehensive and complex game than others, but they will all put together a satisfactory and fulfilling project.

We recommend holding **two sessions per week**, at least at the beginning of the project, to avoid the students forgetting what they have learned from one session to the next, as programming is a genuinely new activity for them.

Unless otherwise stated, all the steps are plugged activities. The durations given are averages.

Levels of difficulty

To help give an idea of the difficulty of the various steps and activities, we use color-coded symbols:

🟢	**Green activity:** easy. All students should manage without difficulty.
🔵	**Blue activity:** medium difficulty. Most students should manage on their own, but some may need a little guidance.
🔴	**Red activity:** difficult. Most students will need some guidance (more or less depending on ability).
⚫	**Black activity:** very difficult. All students will need guidance. These are optional activities.

Listing of the tasks

	Step	Titre	Tâche	
	Step 1	**Introduction to the *Scratch* programming environment**	Activity 0: Demonstration of the final game by the teacher *(5 minutes)*	🟢
			Activity 1: Opening *Scratch* and getting to know its interface (10 minutes)	🟢
			Activity 2: Exploring *Scratch* independently (15 minutes)	🟢
			Activity 3: Short exercises (20 minutes)	🔵
	Step 2	**Customizing the environment and saving all work**	Activity 1: Changing the sprite (5 minutes)	🟢
			Activity 2: Changing the backdrop (5 minutes)	🟢
			Activity 3: Saving your *Scratch* program (5 minutes)	🟢
	Step 3	**Operating the rover**	Activity 1: Making the rover move to the left (10 minutes)	🔵
			Activity 2: Making the rover move in any direction (5 minutes)	🟢
			Activity 3: Driving the rover using the arrow keys (15 minutes)	🔵
			Activity 4: Bouncing off the edges (5 minutes)	🔵
			Activity 5: Reset the position of the rover (5 minutes)	🟢
			Activity 6: Understanding the X and Y coordinates of the rover (20 minutes)	🔵
	Step 4	**Gathering resources and managing scores**	Activity 1: Importing a resource (ice) in the form of a new sprite (5 minutes)	🟢
			Activity 2: Making the resource say “Well done!” when touched by the rover (20 minutes)	🔴
			Activity 3: Making the resource disappear when touched (10 minutes)	🟢
			Activity 4: Creating a “score” variable (5 minutes)	🔴
			Activity 5: Increasing the score when a resource is gathered (10 minutes)	🟢
			Activity 6: Resetting the score to zero (10 minutes)	🟢
			Activity 7: Making resources reappear in random positions (15 minutes)	🔴
			Activity 8: Importing a new resource (plant) and repeating the same tasks as for the ice (20 minutes)	🔵
	Step 5	**Plugged and unplugged activities to better understand certain algorithmic concepts** This step, which is optional, does not deal with programming the video game and should not interrupt it (it should be done as a parallel activity, such as during a math or language arts class).	Activity 1: Formative assessment on the loop concept (plugged activity, 10 to 20 minutes)	🟢
			Activity 2: Set of cards to consolidate the notion of variable (unplugged activity, 1 hour)	🔴
			Activity 3: A card game to work on logical operators (unplugged activity, 1 hour)	🔴
			Activity 4: Understanding that an algorithm is not always perfect: the traveling salesman game (unplugged activity, 1 hour)	🔴
	Step 6	**Avoiding obstacles and managing player lives**	Activity 1: Adding new sprites (5 minutes)	🟢
			Activity 2: Creating and initializing a “number of lives” variable (5 minutes)	🟢
			Activity 3: Losing a life when the rover touches the lava (30 minutes)	🔴
			Activity 4: Repeat Activity 3 for the dune (10 minutes)	🟢

	Step 7	**Ending the game: "Game over"**	Activity 1: Make "game over" appear when there are no more lives (15 minutes)	🔵
			Activity 2: Stopping the game when "game over" appears (15 minutes)	🔵
	Step 8	**Adding challenges**	Activity 1: Make a countdown appear when the game starts (15 minutes)	🔵
			Activity 2: Limiting the game duration (15 minutes)	🔵
			Activity 3: Add a tornado that moves around randomly (15 minutes)	🔴
			Activity 4: Make the tornado bigger (15 minutes)	🔵
			Activity 5: Making the tornado go faster and faster (20 minutes)	⚫
			Activity 6: Simulate a torus world (joining the edges of the backdrop) (20 minutes)	🔴
			Activity 7: Preventing resources and traps from overlapping (20 minutes)	⚫
	Step 9	**Further study in*Scratch***	At this stage, the project is complete. Here we offer a few suggestions to explore other *Scratch* features, which could serve to support students' future personal projects.	

<< Sequence I   Cycle 3 activities   Sequence III >>