

SilverCoders

DIGITAL LITERACY IMPROVEMENT THROUGH EFFECTIVE
LEARNING EXPERIENCES FOR ADULTS



CHALLENGE 22 **ADVANCED GEOMETRY HUNTER**

**CODING TRAINING PROGRAMME
FOR +55 ADULTS**



SILVER CODERS

ERASMUS+ No. 2020-1-SE01-KA227-ADU-092582



**Co-funded by
the European Union**

This document reflects only the author's view and the National Agency and the European Commission are not responsible for any use that may be made of the information it contains

STRUCTURE OF THE CHALLENGE

DESCRIPTION

You were provided with a setup that is meant for you to recall the most important elements of the Gdevelop environment: the scene and the sheet of events. The events available allow the player to move the monster and catch the geometric pieces that, now are falling. You will be asked to improve the game, making it more dynamic (bombs also appear and may kill your monster).

GENERAL GOAL

In this challenge we are going to improve the Geometry Hunter game, making it more interactive and frenzy.

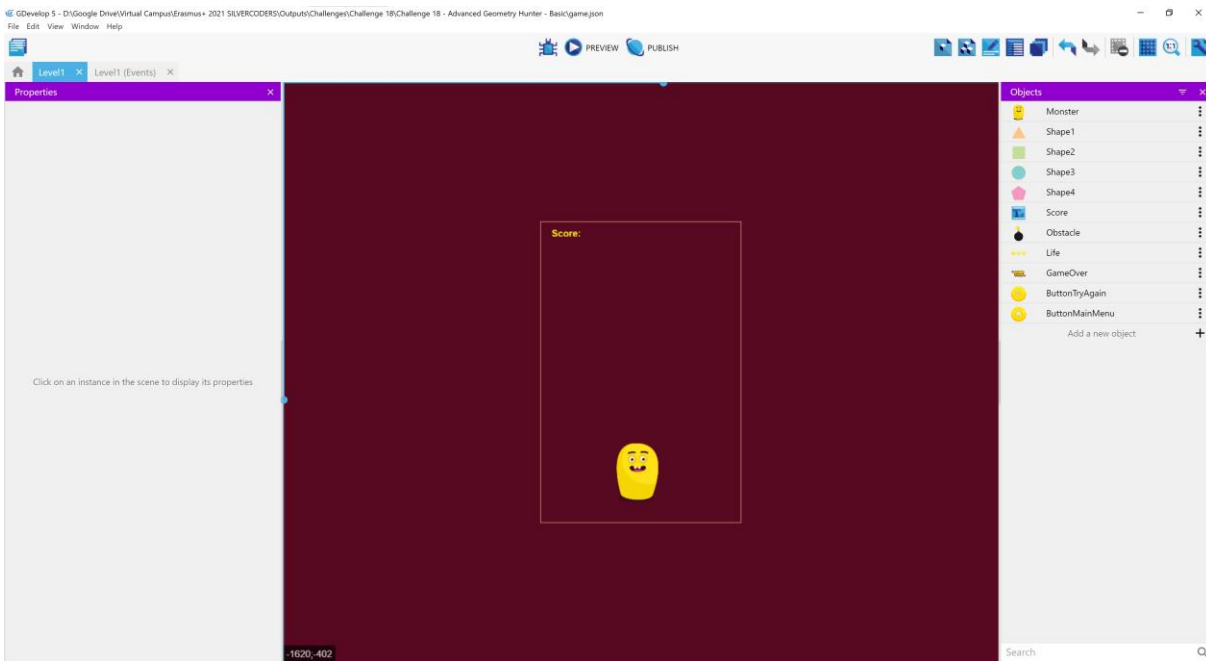
LEARNING OBJECTIVES

In the end of this challenge, you will be able ...:

- To understand how code is treated by a computer and what is the role of a compiler.
- To be familiar with the concept of low and high level languages and understand what their differences are and what is required to code in either of them.
- To have experience with a visual programming suite and be able to code standard small piece of software with it.
- Know what statements and command lines are and what they mean for a compiler.
- To be able to write instructions using correct syntax and with minimal errors.
- Know what operators are, what they do and which symbols stand for which operators.
- To be able to understand the assignment of values to variables and how to change them.
- To know all the basic arithmetic operations and how to use them.
- Recognize and know how to use all the data structures related to numbers.
- To know the structures linked to the use of text, such as strings and characters.
- To be able to use If statements correctly to execute code according to a certain defined fixed condition.
- To know how to use the Gdevelop editor
- To understand the concepts of scenes, events and objects

INSTRUCTIONS

- Start by opening the Gdevelop editor.
- Use the **File** menu to **open** the Challenge 22 – Basic game
- This should be what you get



- Press the **Preview** button to play the game. You can move the monster to the left and to the right with the arrow keys in your keyboard and you should catch the geometrical shapes that now are falling. For every piece you catch, one point is added to your score.
- Repeat the game as many times as you want. To repeat you have to close the game window and press the **Preview** button in the editor.
- Now that you know what are the game mechanics (what you can do in the game) let's see how it is done. We will focus on the differences to the last challenge.

| GAME START | |
|-------------------------------|--|
| At the beginning of the scene | Start (or reset) the timer "ShapeCreation" |
| Add condition | Add action |

- When the game starts we create a **Timer** called **ShapeCreation** which is an object that is always counting the time in seconds.

SHAPES

⌚ The timer "ShapeCreation" is greater than 2 seconds

Add condition

🎨 Among objects **Shapes**, create object named "Shape" + ToString(RandomInRange(1,4)) at position RandomInRange(80, 640-80);-100 (layer:)

📐 Change the angle of **Shapes**: set to RandomInRange(0,360)

📐 Change the scale of **Shapes**: set to RandomFloatInRange(0.8, 1.6)

⌚ Start (or reset) the timer "ShapeCreation"

Add action

- When the **Timer ShapeCreation** reaches 2 seconds a new shape is created and can be randomly one of the four different shapes. To make the game more fun these shapes are scaled and rotated. The we reset the Timer to 0 to start counting again.

Move shape according to the game speed

Add condition

📌 Add to **Shapes** an instant force, angle: 90 degrees and length: Variable(GameSpeed) pixels

🔄 Rotate **Shapes** at speed 90 deg/second

Add action

- The shape »falls« by adding a vertical force to it.
- Let's make the game a little bit more difficult. We are going to drop also some bombs that may remove lives of the monster if it gets hit. For that we are going to use the **Obstacle** object

OBSTACLE

⌚ The timer "ObstacleCreation" is greater than 5 seconds

Add condition

🎨 Create object **Obstacle** at position RandomInRange(80, 640-80);-100 (layer:)

⌚ Start (or reset) the timer "ObstacleCreation"

Add action

Move obstacle according to the game speed

Add condition

📌 Add to **Obstacle** an instant force, angle: 90 degrees and length: 1.5*Variable(GameSpeed) pixels

📄 Change the z-order of **Obstacle**: set to 4

Add action

- The code for the **Obstacle** object is similar to the code for the shapes. We have a timer that controls it and we make it fall.

Repeat for each instance of Obstacle:

📌 **Obstacle** is in collision with **Monster**

Add condition

✖ Delete **Obstacle**

💥 Damage **Monster**, removing 1 from its health

🔊 Play the sound killed.wav, vol.: , loop: no

Add action

- But if there is a Collision with the Monster, it is damaged and he gets 1 less health point



www.silvercoders.eu

3

PROPERTIES

BEHAVIORS

VARIABLES

EFFECTS

Flash

Half period (time during which object is invisible), in seconds
0,1

Health

Damage cooldown (in seconds) (0 for no cooldown)
0,8

Health
3

Maximum health (0 for no maximum)
3

StayOnScreen

Bottom margin, in pixels
0

Left margin, in pixels
0

Right margin, in pixels
133

Top margin, in pixels
0

?

HELP

▶

RUN A PREVIEW

+ ADD A BEHAVIOR

CANCEL

APPLY

- Health is a **Behaviour**, a standard property that we can associate to the objects. The maximum health is 3.

The screenshot shows the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows a 'Life' object under a 'HEALTH' parent. The Inspector panel on the right shows the 'Life' object's properties, including four animations (Life3, Life2, Life1, Life0) and a console log.

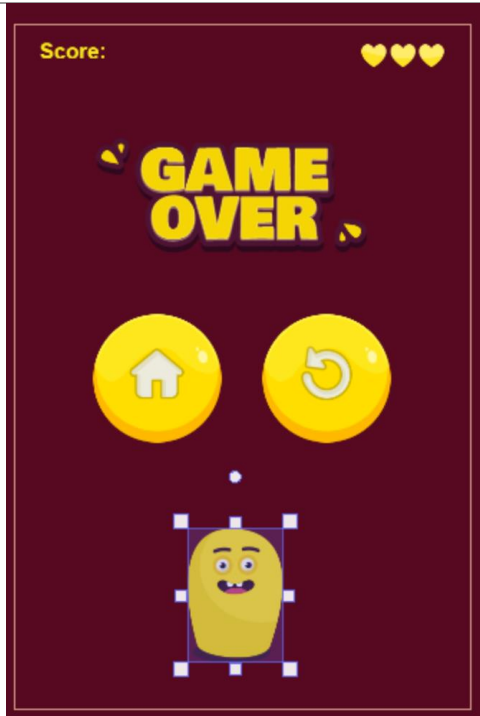
The 'Life' object has four animations, each with a duration of 0.08 seconds and a loop checkbox. The animations are labeled 'Animation #0 Life3', 'Animation #1 Life2', 'Animation #2 Life1', and 'Animation #3 Life0'. Each animation has a 'PISKEL' icon and a 'PREVIEW' button.

At the bottom of the Inspector panel, there are buttons for 'EDIT COLLISION MASKS', 'EDIT POINTS', 'ADVANCED OPTIONS', and '+ ADD AN ANIMATION'. There are also buttons for 'HELP', 'RUN A PREVIEW', 'CANCEL', and 'APPLY'.

The console log at the bottom shows a message: 'Monster has just been damaged'. Below this message, there are two actions: 'Set animation of Life to "Life" + ToString(Monster.Health:Health())' and 'Make Monster blink for 1.5 seconds'.

- To represent visually the health of the monster we will use the **Life** object. This Sprite has 4 frames, each representing a health status.

- When the monster is damaged the Sprite moves to the next frame.
- Now, when the monster is without lifes, the monster is dead. We must show the GameOver button and we will create two buttons, one to restart the game and another to leave the game. This what the initial scene looks.



-
- We don't want to see these objects in the beginning so we hide them.

| GAME OVER | |
|--|--|
| <ul style="list-style-type: none"> At the beginning of the scene Add condition | <ul style="list-style-type: none"> Hide GameOver Hide ButtonTryAgain Hide ButtonMainMenu Add action |

- And we show them when the monster is dead.

| | |
|---|---|
| <ul style="list-style-type: none"> Monster is dead Add condition | <ul style="list-style-type: none"> Set animation of Life to "Life0" Set animation of Monster to "MonsterDead" Delete Shapes Delete Obstacle Show GameOver Show ButtonTryAgain Show ButtonMainMenu Add action |
|---|---|

- We now check which button the player clicked on



RESOURCES

Challenge 22 (Basic)