algorithmics

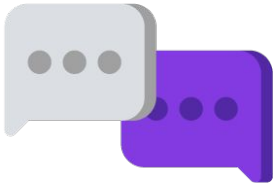
Module 5. Lesson 1.

Basics of game creation

Link to guidelines 

Discussion:

Game creation



Let's move on to game creation!

The ProTeam company has decided to start creating and promoting its own commercial products — computer games.

The PyGame library has been selected as the main tool.

Cole,
senior developer

Discussing work tasks

Introduction to game creation

What do you know about creating games? Let's discuss:

What genres of games are there?

How is a game designer different from a game developer?

What is a game loop?

Emily,
project manager

Discussing work tasks

Your role in the team is game developer

Development management

Product manager

Project manager

Producer

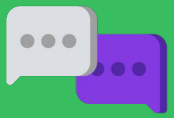
Development team

Developer

Game designer

Artist

Tester

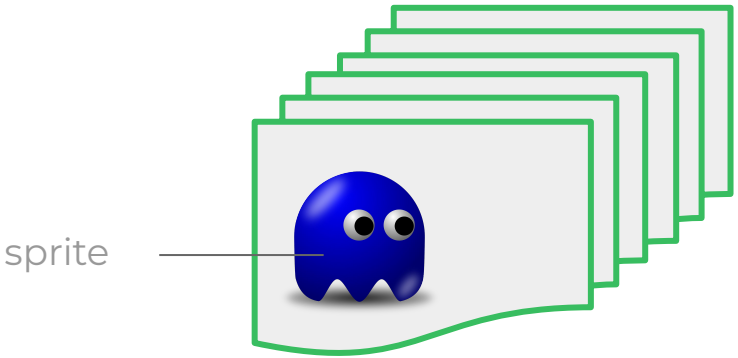


Discussing work tasks

Game loop development

A game loop is a loop, at each step ("frame") of which there is:

- ➔ analysis and processing of events;
- ➔ rendering of the background and characters;
- ➔ a countdown.

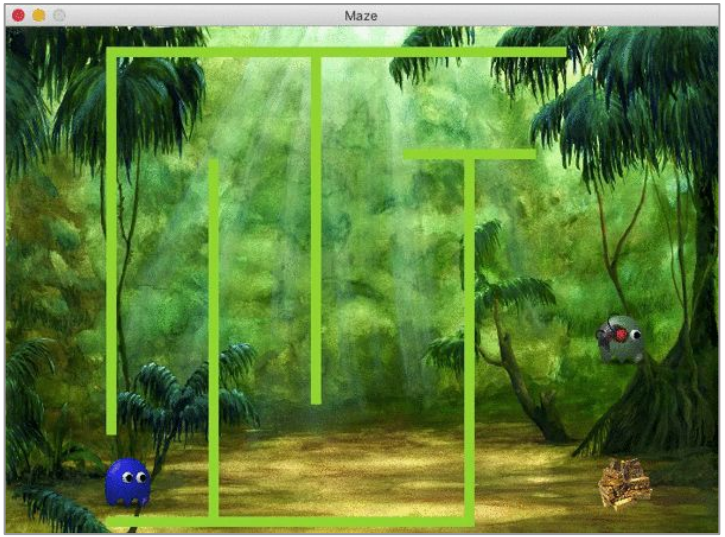


Developers call the game space a scene, and one step of the game loop is a frame.

Discussing work tasks

Training required!

The first ProTeam game product will be the interactive Maze game.



To create such a game, you will need to be comfortable with the basics of PyGame!

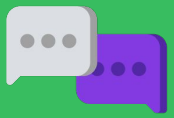
Discussing work tasks

The goal of the work day is

to learn and practice the basics of game creation using PyGame.

Today you will :

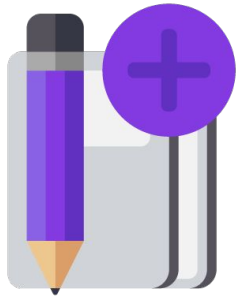
- explore the PyGame library for game creation;
- learn how to control sprites using the keyboard;
- program the "Catch" mini project.



Discussing work tasks

Brainstorming:

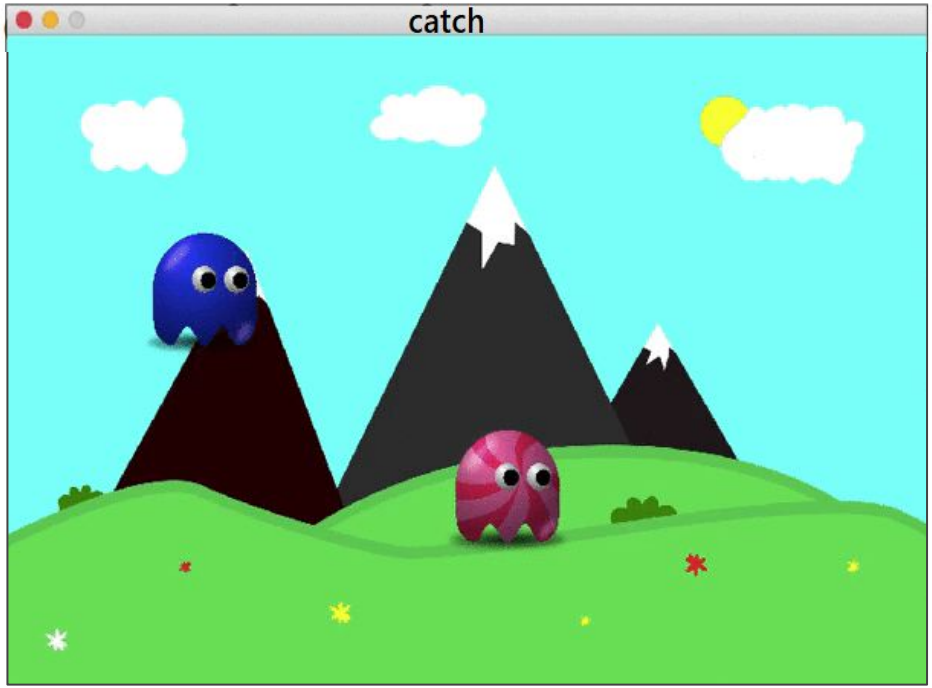
Game creation using PyGame

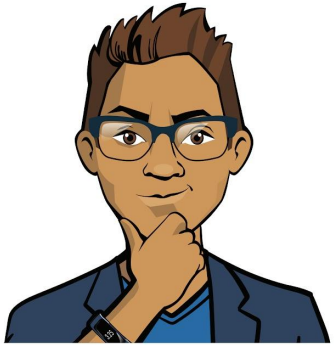


Game creation

Let's study the appearance of the "Catch" game.

What do we need to know to program a game like this?

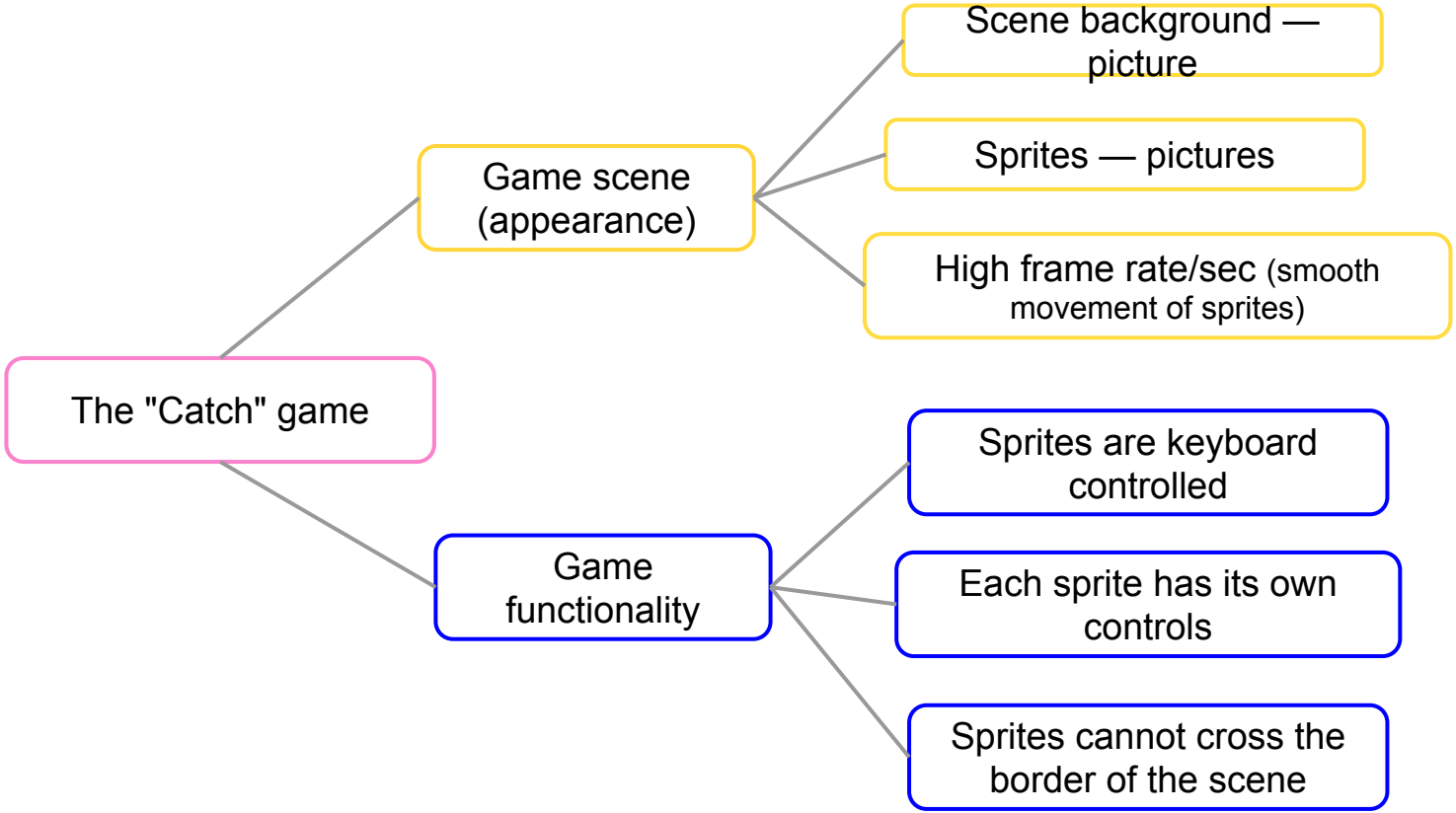




Brainstorming

The "Catch" game

Let's discuss the content of the game using a mind map:



Brainstorming

PyGame

is a library for game creating

The PyGame library is hierarchical.

There are lots of modules in it with ready-made tools for:

- ➔ creating sprites;
- ➔ processing in-game sprite events;
- ➔ processing events from the outside world;
- ➔ game timer settings;

and more.

The ygame.transform module

Functions for transforming surfaces
(e.g. background to fit the window)

The pygame.event module

Outside world events
(methods and required constants)

. . .

Brainstorming

PyGame

is a library for game creating

The PyGame library is hierarchical.

There are lots of modules in it with ready-made tools for:

- ➔ creating sprites;
- ➔ processing in-game sprite events;
- ➔ processing events from the outside world;
- ➔ game timer settings;

and more.

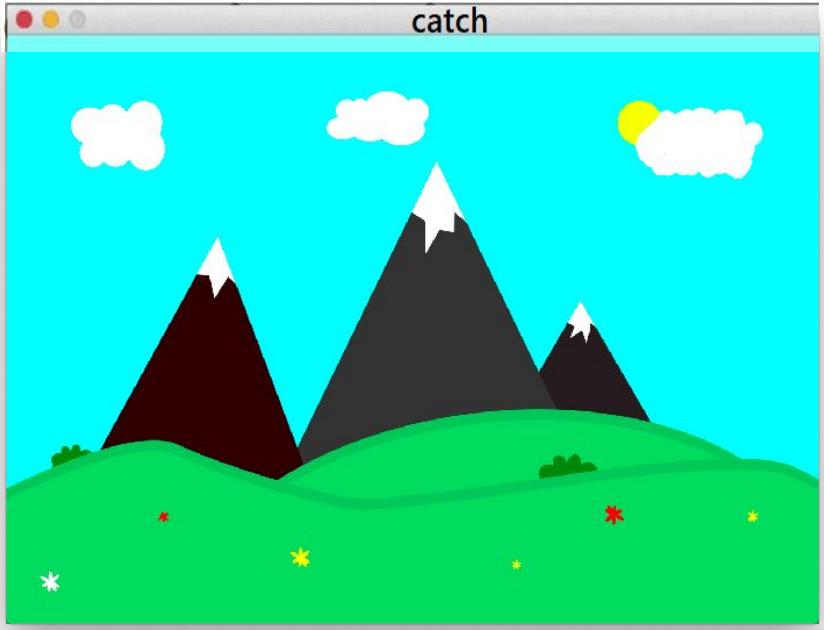
Command	*Purpose*
`from pygame import *`	Importing all the functionality of the PyGame library

Brainstorming

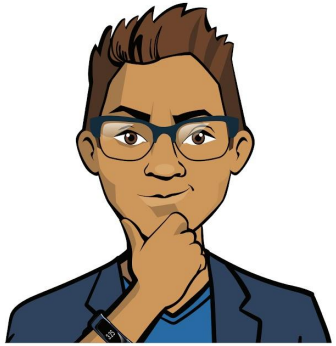
1. Creating a blank with a background

Let's program a blank for a game with a picture background.

Note. The picture must be in the project folder.



What do we need to know
to create such a scene?

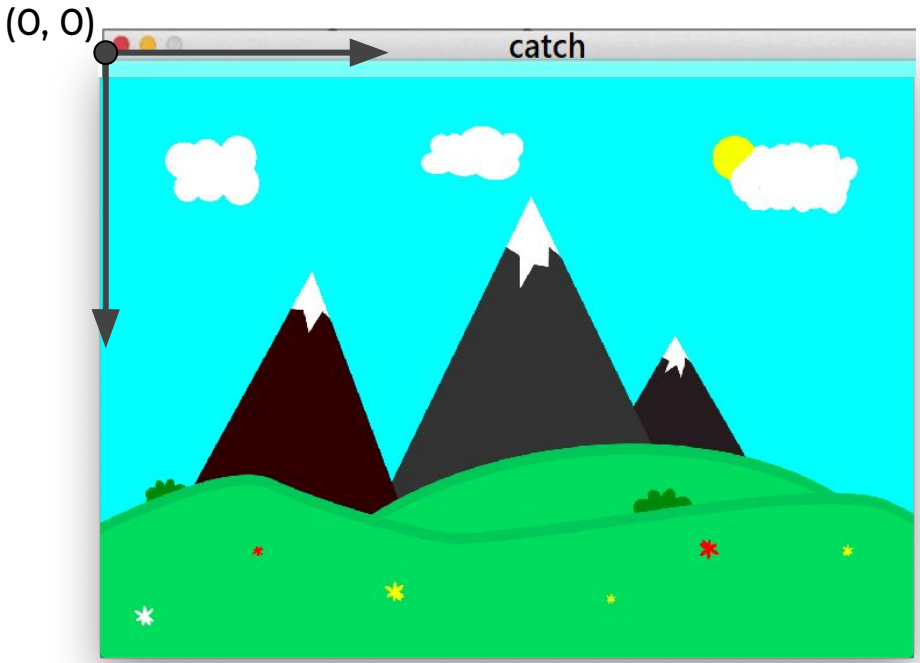


Brainstorming

1. Creating a blank with a background

By default, the origin is located in the upper left corner of the window.

The size of the window is determined by the developer.



Brainstorming

1. Creating a blank with a background

Let's program a blank for a game with a picture background.

Note. The picture must be in the project folder.

Command	*Purpose*
`window = display.set_mode((700, 500))`	Create the window size: (width, length).
`display.set_caption("Catch")`	Set the window title.
`background = transform.scale( image.load("background.png"), (700, 500) )`	Create a picture object, adapt the picture size to the window parameters.
`window.blit(background,(0, 0))`	Display the background picture in the window.

Brainstorming

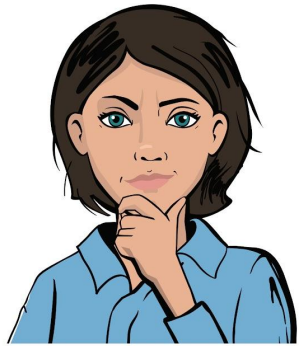
1. Creating a blank with a background

Let's program a blank for a game with a picture background.

```
from pygame import *

window = display.set_mode((700, 500))
display.set_caption("Catch")
background = transform.scale(image.load("background.png"), (700, 500))

window.blit(background,(0, 0))
```



If we create and run such a program, it will start and then exit immediately! Why?

Brainstorming

1. Creating a blank with a background

Let's program a blank for a game with a picture background.

```
from pygame import *


window = display.set_mode((700, 500))
display.set_caption("Catch")
background = transform.scale(image.load("background.png"), (700, 500))

window.blit(background,(0, 0))
```

There is no game loop in the program!

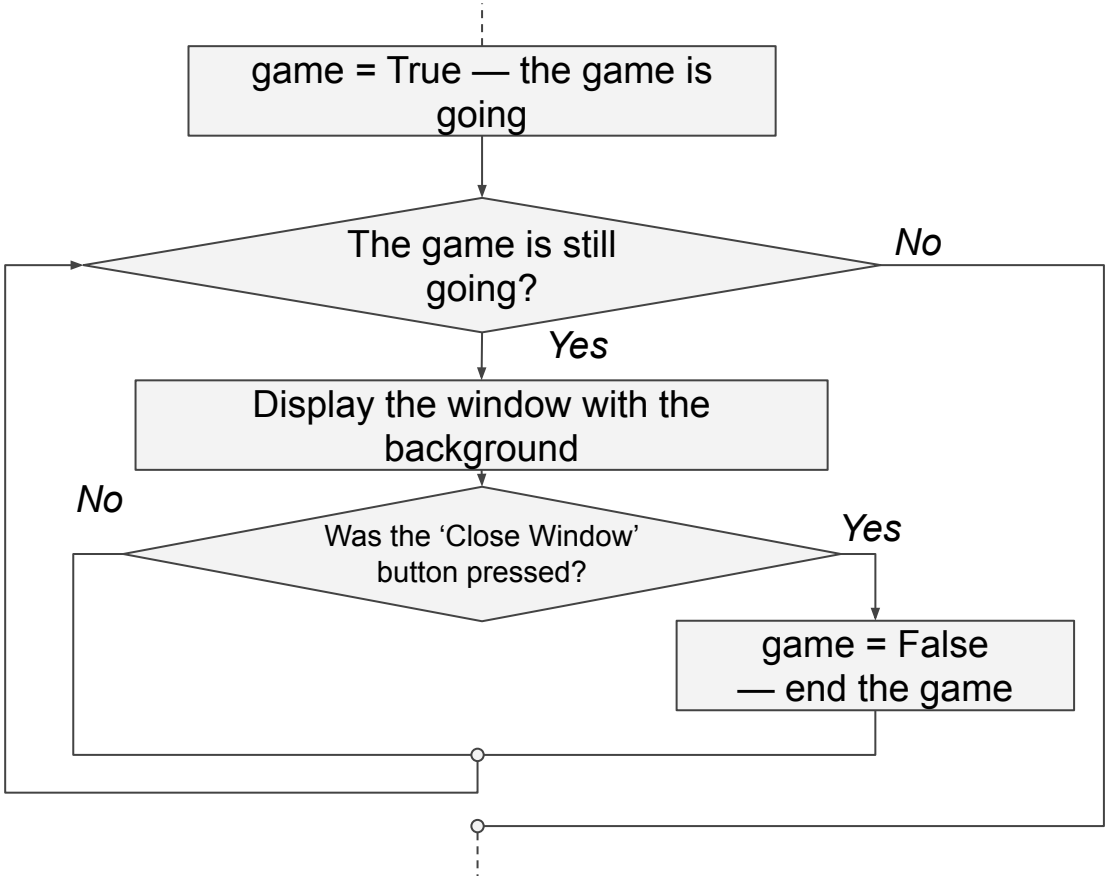
The window is displayed for a moment and then disappears.

What do we need to add so that the window is always displayed?

Brainstorming

1. Creating a blank with a background

The simplest game loop that displays the window until it is closed.



Brainstorming

1. Creating a blank with a background

New methods related to conditions that end the game.

Command	*Purpose*
`events = event.get()`	Returns a list of events that have occurred (events are instances of the ready-made Event class).
`events[0].type`	Each event object has a type property — **event type** (for example, "keydown").

Type examples:

QUIT — the "Close window" button is pressed (red x).

KEYDOWN — any key is down.

Program names for keys:

K_LEFT — ‘Left Arrow’ button pressed.

K_a — the ‘Letter A’ (Latin) button is pressed.

Brainstorming

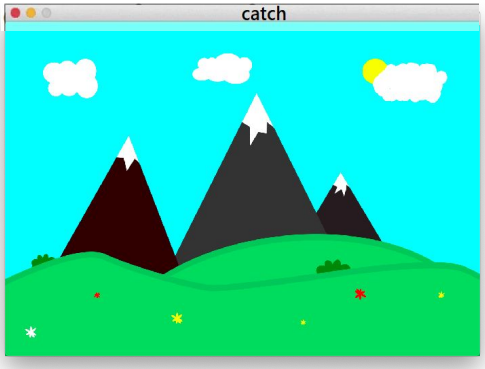
1. Creating a blank with a background

Let's program a blank for a game with a picture background.

```
from pygame import *
window = display.set_mode((700, 500))
display.set_caption("Catch")
background = transform.scale(image.load("background.png"), (700, 500))

game = True
while game:
    window.blit(background,(0, 0))

    for e in event.get():
        if e.type == QUIT:
            game = False
```



`display.update()`

The frames that appear on the screen with each step of the loop must be updated.

Brainstorming

2. Creating and placing sprites

Objects for image sprites are created in the same way as the background.

Command	*Purpose*

`sprite1 = transform.scale(`
`  image.load('sprite1.png'),`
`  (100, 100)`
`)`
	Create a picture object and fit it into a 100x100 square.

`window.blit(sprite1, (x1, y1))`	Place the sprite in the window at the point (x1, y1).

How do we add sprites to an existing program?

Brainstorming

Combining fragments in a program

We get a game with inactive sprites for now.

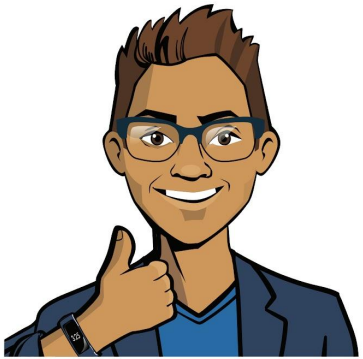
Importing PyGame modules

Creating objects for the background and sprites

The game = True variable (game started)

Game loop:

- Placing sprites on the scene
- End the game if the "Close window" button is pressed
- Scene update (the next frame of the game loop)



Brainstorming

Conclusions:

- Games can be created using the **PyGame library**. There are lots of ready-made modules for different tasks in it.

- All games are based on **game loops**. The rendering of the game scene, the processing of all the events, and the end of the game are described in the loop.

Brainstorming

Module 5. Lesson 1. Basics of game creation

Platform:

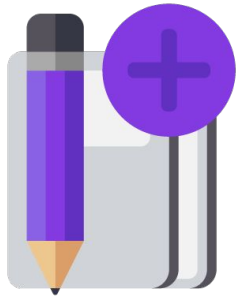
VSC PyGame: Catch

Complete the tasks in VS Code

PyGame: Catch, tasks 1 and 2

Brainstorming:

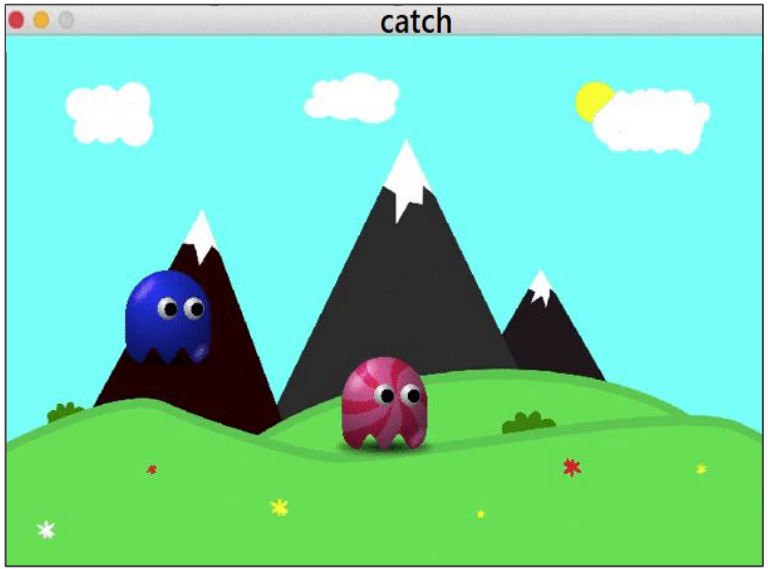
Keyboard event handling

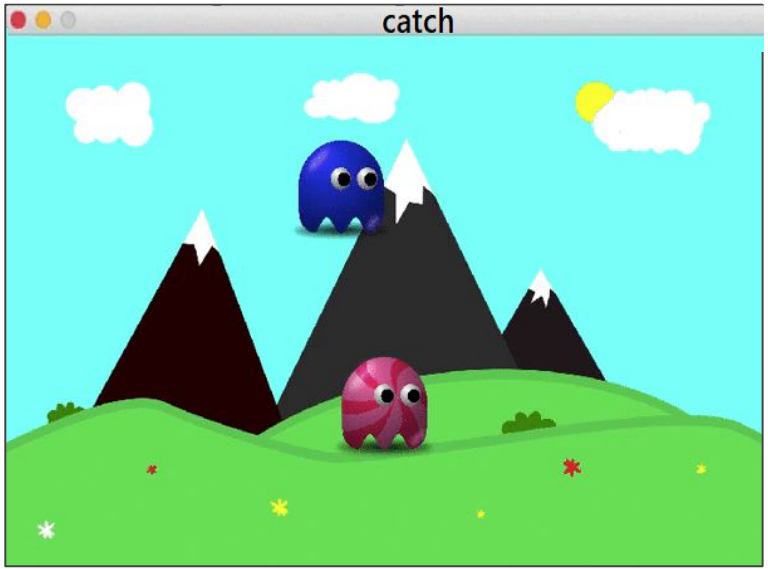


Controlling sprites by using the keyboard

Compare the two games. The speed the sprites move at is the same.

Why does the sprite move more smoothly and quickly in the game on the right?



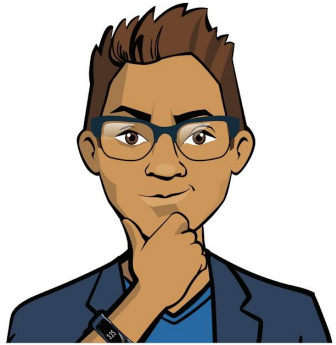


Brainstorming

Controlling sprites by using the keyboard

Controlling sprites is directly related to updating the scene on which the sprites are located.

A **high update rate** **looks better**, but it requires higher computer performance (other things being equal).

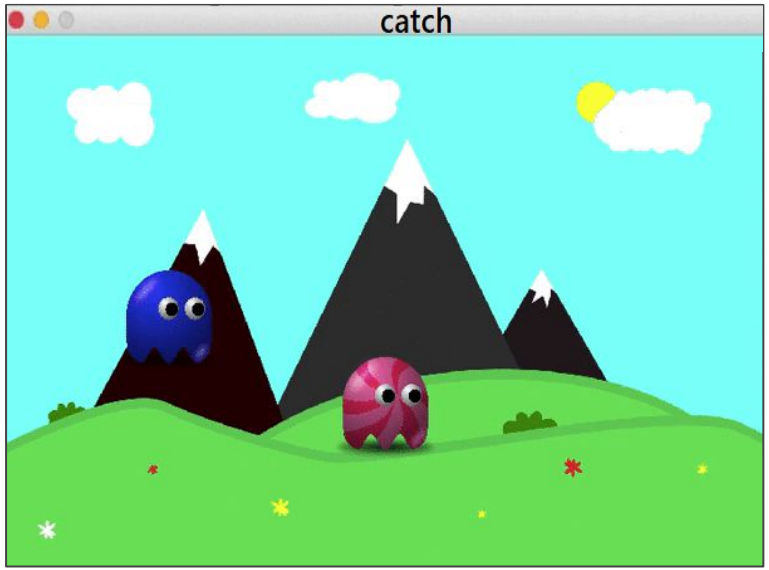


Brainstorming

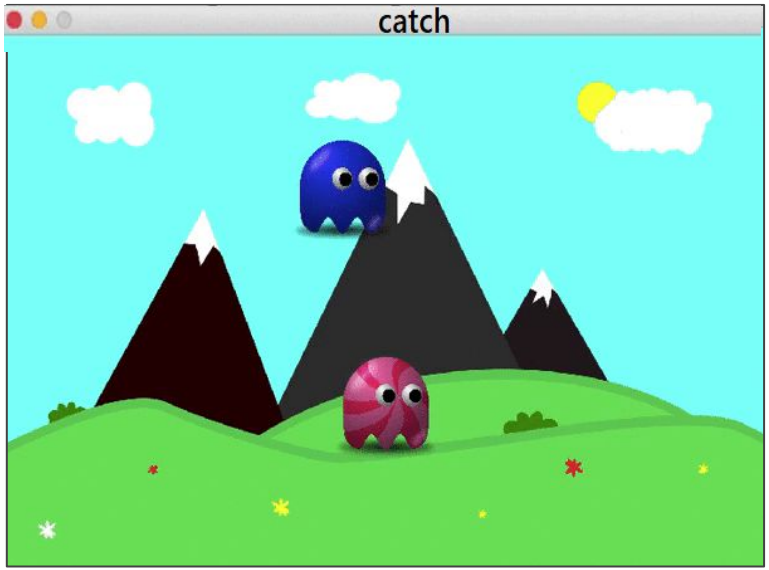
FPS (frames per seconds)

is the number of frames displayed in one second.

A comfortable frame rate for movies is 24, for games it’s 50-60.



The FPS is 5 frames/sec



The FPS is 60 frames/sec

Brainstorming

1. Setting the frame rate

Let's create a special Clock() object and give it the rate we want.

Command	*Purpose*
`clock = time.Clock()` `FPS = 60`	Create a "clock" object that keeps track of time. Let's immediately create the FPS constant and set the desired frame rate.
`clock.tick(FPS)`	In each frame, a second will be divided by 60. There will be a delay of 1/60th of a second.

Let's place it in the game loop.

Brainstorming

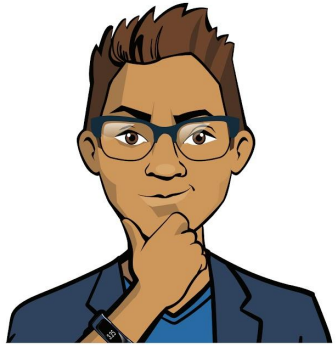
2. Keyboard events handling

Let's control sprite1 using the arrow keys.

Let’s have a keystroke move the sprite 10 pixels.

How do we process the “‘up arrow’ key pressed” event?

Wouldn't the conditional statement iterating all the control keys turn out to be too large?



Brainstorming

2. Keyboard events

Let's take a look at the functions that simplify keyboard event processing.

Command	*Purpose*
`keys_pressed = key.get_pressed()`	Returns a structure with the current state of the keys (True is down, False is up).
`if keys_pressed[K_UP]:` `  y1 -= 10`	If the ‘Up Arrow’ key is down, decrease the Y coordinate of Sprite1 by 10 pixels.
`if keys_pressed[K_s] and y2 < 395:` `  y2 += 10`	If the "S" key is down and the bottom of the screen has not been reached, increase the Y coordinate of Sprite2 by 10 pixels.

Brainstorming

2. Keyboard events

The game loop after adding keyboard event processing:

```
while game:
    #...
    keys_pressed = key.get_pressed()

    if keys_pressed[K_LEFT] and x1 > 5:
        x1 -= speed
    if keys_pressed[K_RIGHT] and x1 < 595:
        x1 += speed
    if keys_pressed[K_UP] and y1 > 5:
        y1 -= speed
    if keys_pressed[K_DOWN] and y1 < 395:
        y1 += speed

    #...
```

You need to check if the sprite has left the scene every time.

Brainstorming

Your tasks:

- ➔ Set the frame update rate to 50-60 in the Catch mini game.
- ➔ Program your sprite controls. Let Sprite1 be controlled by the arrow keys and Sprite2 by the letters "A" ... "D".
- ➔ Run and evaluate the game.

Brainstorming

Module 5. Lesson 1. Basics of game creation

Platform:

VSC PyGame: Catch

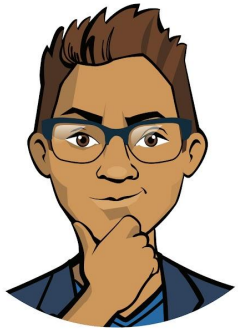
Complete the tasks in VS Code

PyGame: Catch, task 3

Wrapping up the work day

Let’s end the work day by answering these technical questions:

1. Which library contains tools for game creating? What functionality do you remember?
2. How do we create a game? What is the basis of any game?
3. What is FPS? What does this parameter affect?



Cole,
senior developer

Emily,
project manager

Excellent work!

Colleagues,

Today you learned the basics of game creation using PyGame.

On our next work day, we will be able to start creating the Maze game!

Wrapping up the work day