PROJECT DESCRIPTION

OBJECT - ORIENTED PROGRAMMING FOR GAME DEVELOPMENT WITH P5.JS

PREPARED FOR: PROCESSING FOUNDATION
PREPARED BY: JUAN OLAYA

INTRODUCTION

This document describes the project **Object-Oriented Programming for Game Development with p5.js,** this is an online book that explains how to develop 2D video games using the JavaScript library p5.js. It has 20 lessons that shows examples focus on game development such as: player movements, collision detection, gravity simulation, jumps on platforms and shoot bullets.

The beginning of the book shows how to use a basic class with its parts, how to create an instance of a class and how to invoke its methods. Then, it is presented how to use polymorphism and inheritance, which are especially useful to display different video game characters with attributes and methods in common. At the end of the book, the collision detection algorithms are used to develop different video games mechanics.

In this online book, each of 20 lessons has:

- A text that describes the subject
- An example in p5.js with its documentation code. In order to identify which part of the script code is related to the class code, the code documentation has been divided by different colors in the background, making easily visible the structure of the code and easily understandable the links between the script and classes (as can be seen in the **Lesson 1** and **Lesson 3**).
- A class diagram that describe the attributes and methods of each class and its relationship with the main script.
- And also, an embedded sketch running on OpenProcessing.

DEVELOPED PRODUCTS

Below I present the links of the products related with this project, Object-Oriented Programming (OOP) for Game Development:

1. **OpenProcessing Class:** An OpenProcessing Class with the structure of the curriculum of this project with teacher examples and student projects.
2. **Github Repository:** A repository of Github where you can find the structure of this curriculum, code documentation and some class diagrams. In addition, the students can find useful links of p5.js and Processing.
3. **Itch.io:** In the following list you can find the links of my student video games published on itch.io. These video games were developed using the curriculum of this project with Processing. For future courses the video games will be publish using p5.js as programming language:

    - **Rot & Blau**
    - **Animals vs Aliens**
    - **Painting Punks**
    - **Pixel Dancers**
    - **Space Balls**
    - **Alien Fight**

- **Tanques**
- **Running Pixel Girls**

4. **More Students Projects:** There are also, some student video games published on OpenProcessing by my students. These video games are developed in p5.js or Processing:

    - **Navecitas v2.3**
    - **BallMaze (Gyroscope-Phone)**
    - **Play 2**
    - **Game**
    - **Cube**
    - **Lineas Game**
    - **Space Ship**
    - **Pez.k**
    - **Inversiones Semánticas**
    - **Arkanoid**
    - **TransmiKiller**
    - **Esquivalo**
    - **Car Obstáculos**
    - **Rana**
    - **Saltando Obstaculos**
    - **Invaders 4 Real**
    - **Come Bolitas**

EXPERIENCE

For this project I am going to use the curriculum, products and knowledge that I have built and acquired working with Processing and p5.js as university educator since 2016. I am a Computer Engineering and I am currently teaching subjects such as: Introduction to Programming, Object-Oriented Programming, Data Structures and Software Engineering in the bachelor Digital Arts Engineering in the university Escuela de Artes y Letras (School of Arts and Letters) in Bogotá, Colombia.

In this respect, I also have taught video game subjects, using the game engine Unity 3D, such as: Virtual Reality and also Artificial Intelligence for Video Games. Therefore, for this project I am using the experience and knowledge that I acquired with Unity 3D, to implement game mechanics in p5.js that allow build functional 2D video games. The general curriculum and the students results of these subjects using Unity 3D, can be seen in the following links:

- **Virtual Reality**
- **Artificial Intelligence for Video Games**

In addition, I studied the master's degree Cognitive Systems and Interactive Media in the Pompeu Fabra in Barcelona, Spain where I started developing interactive apps using

Processing. In this master's I carried out the thesis *XIM expressions: Socializing with a non-anthropomorphic robot*. This thesis was elaborated using a full-body experiment which was developed in Unity 3D as can be seen in this link.

THEORY

The object-oriented programming paradigm allows to reuse code and to have the control of each object on the screen. It means, this paradigm allows to store the values of the properties of each object such as: position, size, color, velocity, acceleration, or availability status. These properties can be managed through methods that represent object behaviors such as: change color, change size, move up or move in any direction and bounce on the edges of the screen.
During this curriculum, the 4 characteristics of object-oriented programming are used: abstraction, encapsulation, inheritance and polymorphism.

EXPECTED RESULTS

The result of this project is an online book that shows 20 lessons. Each lesson has the code of the examples, a code documentation, a class diagram and embedded sketch running on OpenProcessing. The 20 lessons are listed below:

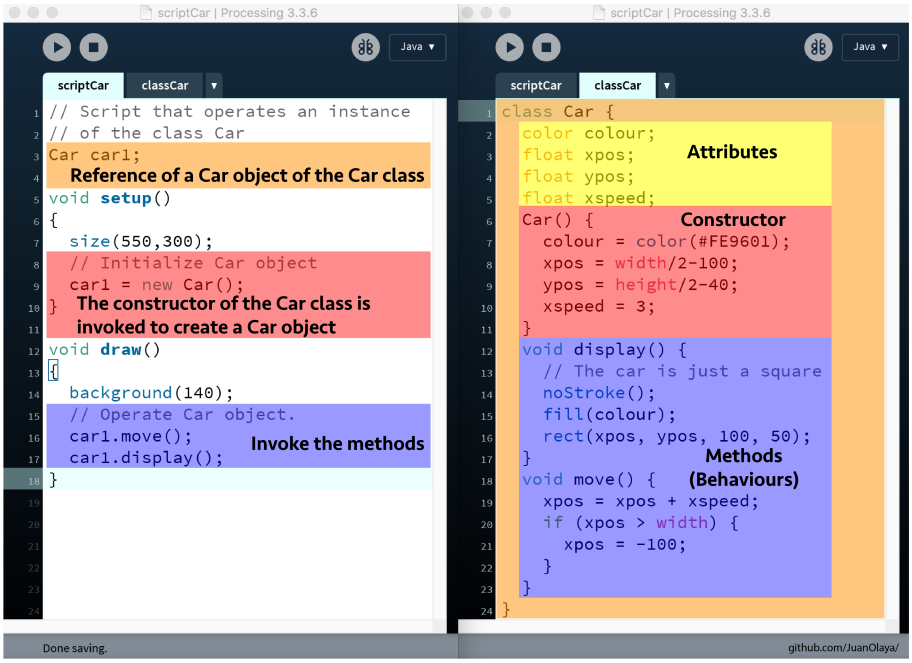
Lesson 1: One class, one object
Lesson 2: One class, two objects (and the constructor with arguments)
Lesson 3: One class, multiple objects (and how to store them on arrays)
Lesson 4: Using vectors
Lesson 5: Gravity
Lesson 6: Inheritance and polymorphism
Lesson 7: Player movement with arrow keys
Lesson 8: Collision detection
Lesson 9: Two players movement and stop player by obstacle
Lesson 10: Jump
Lesson 11: Jump on platform
Lesson 12: Bullet
Lesson 13: Bullet with delay
Lesson 14: Screens and buttons
Lesson 15: Using sprites for character animations
Lesson 16: Playing sounds
Lesson 17: Storing the score with JSON
Lesson 18: Sorting the ranking array
Lesson 19: More examples
Lesson 20: Publication on Itch.io

Some of the examples used in this project are developed in Processing and others in p5.js due to the fact that when I started teaching I chose Processing and now I am teaching my courses with p5.js. However, in the execution of this project I am going to translate the

examples from Processing to p5.js using the ECMAScript 6 which is the latest version supported by OpenProcessing and the p5.js Web Editor. I described each lesson below:

Lesson 1: One class, one object

Based on the Car class developed by Daniel Shiffman, I described in this lesson, the parts of a class such as: Constructor and Methods. I also described how to create an object in the class Car using the constructor and how to invoke its methods. See the example sketch on OpenProcessing (p5.js)

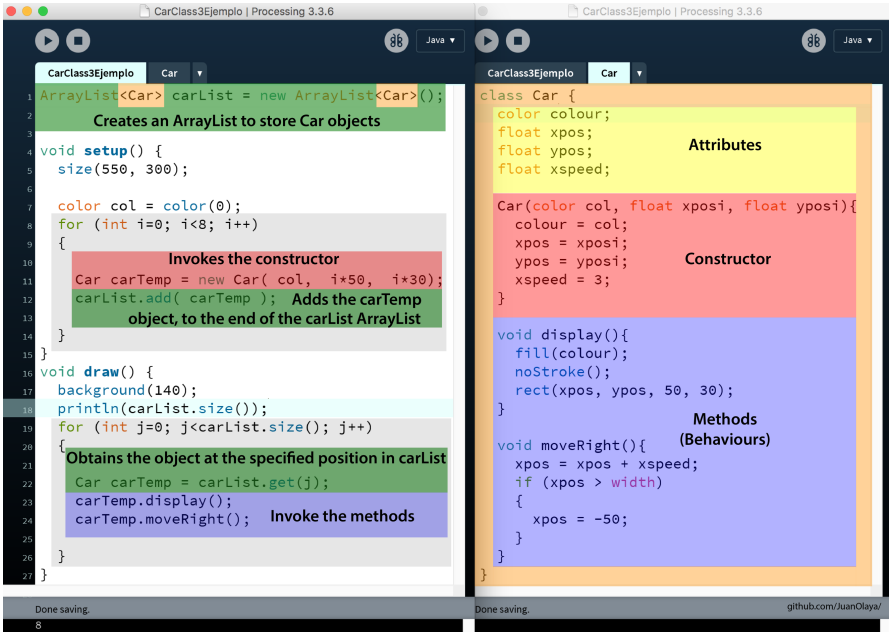


Lesson 2: One class, two objects (and the constructor with arguments)

This lesson shows how to create two objects from a single class, the Car class. The constructor with arguments is also used to be able to have two objects with different colors and different lanes (heights). See the example sketch on OpenProcessing (p5.js)

Lesson 3: One class, multiple objects (and how to store them on an array)

This lesson shows how to create multiple objects of a single class, the Car class and how to store these objects in an array. See the example sketch on OpenProcessing (p5.js)



Lesson 4: Using vectors

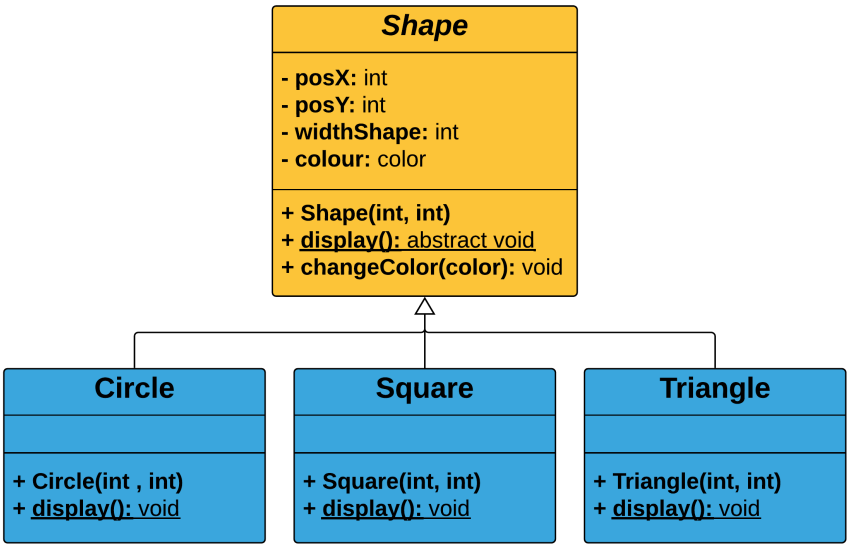
This lesson shows how to implement the class p5.vector to store a two dimensional vector inside of the Car class. See the example sketch on OpenProcessing (p5.js)

Lesson 5: Gravity + border rebound

This lesson shows the class Ball and how to develop the methods of the Ball to simulate the gravity using also the class PVector to store the: location, velocity and acceleration in x-axis and y-axis. See the example sketch on OpenProcessing (Processing).

Lesson 6: Inheritance and polymorphism

This lesson explains the object-oriented programming concepts about inheritance and polymorphism through a superclass Shape and with the subclasses Circle, Square and Triangle. See the example sketch on OpenProcessing (Processing).



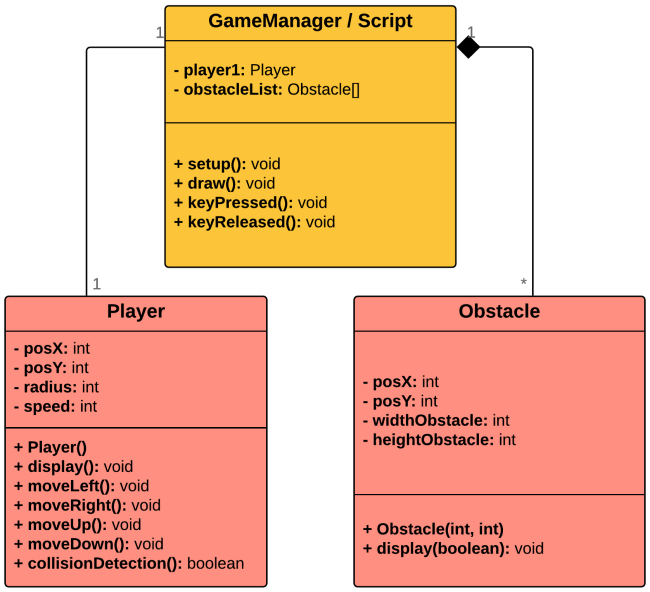
Lesson 7: Player movement with arrow keys

This lesson shows the class Player with its movement behaviors such as: moveLeft(), moveRight(), moveUp and moveDown. Then, it explains how to create an object of class Player and how to invoke the player movements from the script using the keyIsDown() p5.js function. See the example sketch on OpenProcessing (p5.js)

Lesson 8: Collision Detection

This lesson shows the algorithm about how to detect collisions between a circle and a square. This code is based on Jeff Thompson's algorithm and to be used in the lessons of this curriculum was added a method to detect the side of the collision between the circle and the square. It is able to detect the collision in 4 sides of the square. See the example sketch on OpenProcessing (Processing)

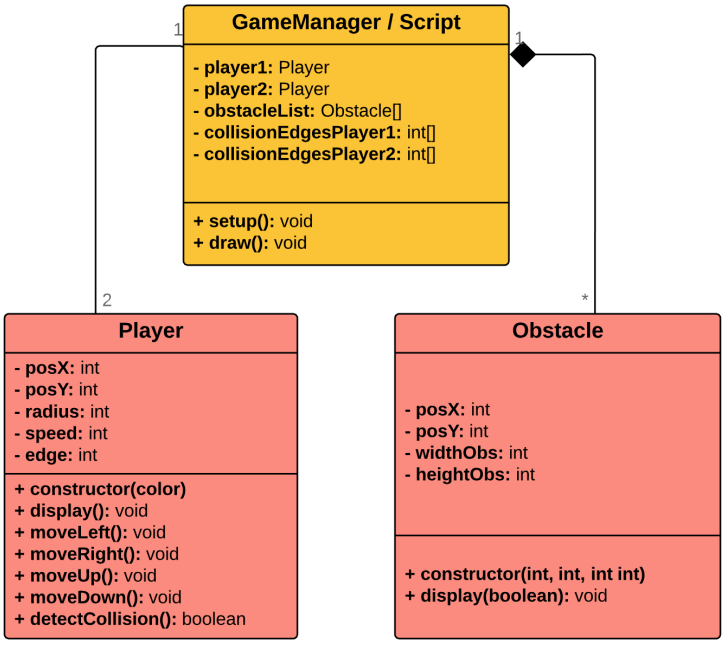
In addition, this lesson presents how to communicate two classes to detect the collision between the player and the obstacles. See the example sketch on OpenProcessing (Processing)



Lesson 9: Two players movement and stop player by obstacle

This lesson also presents how to communicate between two classes to detect the collision between the player and the obstacles and how to stop the player when it found an obstacle. See the example sketch on OpenProcessing (p5.js)

For this lesson was added a method that is able to detect the collision in 8 sides of the square in order to be more accurate and implement more realistic behaviors. See the example sketch on OpenProcessing (Processing)



Lesson 10: Jump

This lesson will show a basic implementation of the jumping algorithm without endless procedural platforms. This jump implementation does not allow jumping in the air, that is, the jump can only be executed when the player is on the ground. See the example sketch on OpenProcessing

Lesson 11: Jump on platform

This lesson shows how to develop an algorithm that can simulate a player jumping on platforms. This jump implementation does not allow jumping in the air, that is, the jump can only be executed when the player is on the ground or on a platform. To simulate the player movement on the air this algorithm is using the concepts of gravity taught in the lesson 5. This sketch also shows how to generate endless procedural platforms. See the example sketch on OpenProcessing (Processing)

Lesson 12: Bullet

This lesson shows an outline with three classes: Bullet, Enemy and Player. Bullets use the player's position when they are created. Then, the bullets move to the right side. The collision detection algorithm compares each bullet against each enemy in search of impacts, whether there is an impact, then the enemy disappears. See the example sketch on OpenProcessing (Processing)

Lesson 13: Bullet with delay

This lesson explains how to use the millis() function of p5.js to develop a delay between each fired bullets. See the example sketch about millis() function on OpenProcessing

Lesson 14: Screens and buttons

This lesson explains how to change screens through buttons using p5.js. See the example sketch on OpenProcessing

Lesson 15: Using sprites for character animations

This lesson uses the example Animated Sprites of Processing to use an external set of images to represent animations for video game characters. The example of Processing can be seen in this link. A functional example of this algorithm can be seen in the student project **Rot & Blau** developed with the curriculum of this project.

Lesson 16: Playing sounds

This lesson uses the example of p5.js to load sound and play them. The example of p5.js can be seen in this link.

Lesson 17: Storing the score with JSON

This lesson explains how to store the score of each player in a JSON file as can be seen in the image below.

```
{"players": [
  {
    "score": 157,
    "password": "1234",
    "name": "Martha",
    "id": 1
  },
  {
    "score": 294,
    "password": "1234",
    "name": "Amelia",
    "id": 2
  },
  {
    "score": 233,
    "password": "1234",
    "name": "Joe",
    "id": 3
  }
]}
```


Lesson 18: Sorting the ranking array

This lesson explains how to use bubble sort algorithm to organize a list, which store the score of each player, from highest to lowest.

```
void bubbleSortAlgorithm() {
  for (int i=0; i<playersList.size()-1; i++) {
    for (int j=0; j<playersList.size()-1; j++) {
      if(playersList.get(j).score < playersList.get(j+1).score){
        Player playerAux1 = playersList.get(j);
        Player playerAux2 = playersList.get(j+1);
        playersList.set(j,playerAux2);
        playersList.set(j+1,playerAux1);
      }
    }
  }
}
```

Lesson 19: More examples

- **Maze**
  This example implements a maze using the player movement of the **Lesson 7** and the collision detection of the **Lesson 8** of this curriculum.
  See the example sketch on OpenProcessing (Processing)

- **Collision + Rebound + Shake**
  This example implements behaviors to the objects such as: shake from bottom to top or shake from left to right depending on side collided. See the example sketch on OpenProcessing (p5.js)

- **12 sides collision detection**

In order to detect the collisions more accurate was necessary to enhance the code to be able to recognize which is the side that is colliding. This algorithm can detect a collision between a circle and a square. It is able to detect the collision in 12 sides of the square. See the example sketch on OpenProcessing (Processing).

Lesson 20: Publication on Itch.io

The activity for this lesson is create a user account on Itch.io and publish the video game on this platform. Is relevant upload a gameplay video. A list of some video games created by my students can be found on my itch.io profile in the link below:
https://juanolaya.itch.io/

EXPANDING POSSIBILITIES OF P5.JS

As seen in the inspirational video tutorials of Daniel Shiffman is possible to develop several 2D video games using Processing and p5.js. However, there are many game mechanics that have not yet been explored. In fact, there is not yet a central website or book that focus on the game development using Processing or p5.js.

Therefore, there is an opportunity to build and publish a curriculum that allow to teachers, students, educational institutions and enthusiasts, teach and learn how to develop 2D video games using p5.js, and as a result, position p5.js as a relevant 2D game engine.

In addition, it is relevant to define a programming paradigm in p5.js, such as the object-oriented one, as a programming strategy to develop game mechanisms and game managers which allow the implementation of comprehensible and scalable solutions for game developers in training.

TARGET AUDIENCIES

This online book is focused on the field of Computer Science Education for teachers and students who want to teach or learn the concepts about object-oriented programming using creative coding, exploring the concepts about the physics simulations and 2D game development.

This kind of teachers and students are found in conventional high schools and interdisciplinary bachelors or master's degrees about new media, electronic arts or human-computer interaction which have students with different backgrounds that must to learn the object-oriented programming paradigm to develop functional 2D video games with educational and entertainment approaches.

This project takes advantage of the student motivation and their meaningful experience playing video games, to invite them to create their own video games and turn into reality their video game ideas.

This curriculum focuses on advanced beginner students who have studied the basic topics of Processing or p5.js. These basic topics can be reviewed in the following featured books:

- Shiffman, D. (2009). ***Learning Processing****: a beginner's guide to programming images, animation, and interaction*. Morgan Kaufmann.

- McCarthy, L., Reas, C., & Fry, B. (2015). ***Getting Started with P5. js****: Making Interactive Graphics in JavaScript and Processing*. Maker Media, Inc.

CONCLUSIONS AND FUTURE WORKS

This curriculum is a gentle manner to teach and learn the object-oriented programming paradigm while developing 2D video games. The development of this project with the support of Processing Foundation could position p5.js as a relevant 2D game engine.

In addition, in order to be more robust and more affordable this project it would interesting to consider develop the following ideas:

- Implement more game mechanics
- Implement game templates
- Develop a user sessions management for video games in p5.js
- Translate this online book to Spanish in order to be easily used by the Spanish speakers in Latin America, Spain and the United States.
- Develop a multiplayer manager using WebSockets in p5.js. Currently, I am using for my classes the library oscP5 with Processing as can be seen in the student project Painting Punks. However, it must be implemented using WebSockets to be compatible with p5.js and to be included in this project in the future.