Introduction to Game Programming

Session 6 – Some additional improvements

Nacho Iborra

IES San Vicente

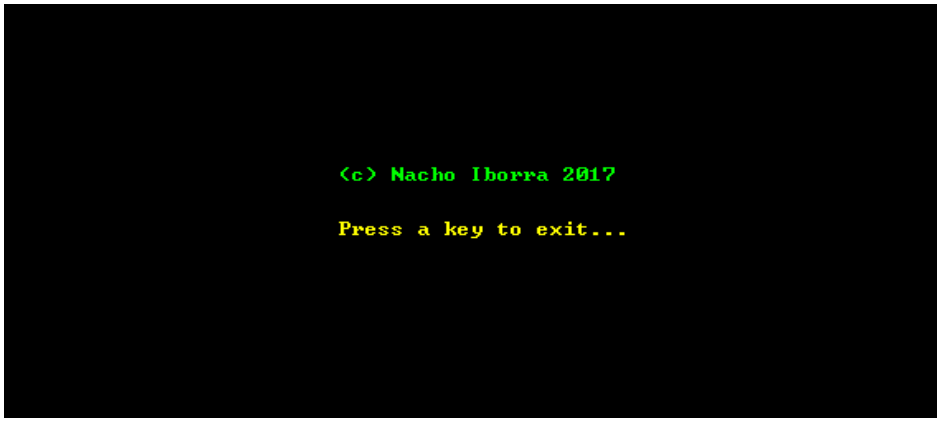
Index of contents

1. Introduction

In this last session of our console video game, we are going to add some final improvements to it:

- A welcome screen at the beginning of the game, so that we will not start playing until we press a key.

- A credits screen when finishing the game, so that we will not exit the game until we press a key



- A multi-level video game, in which we define, at least, 3 different levels. After finishing these levels, we will move to credits screen to exit the game.

2. Welcome and credits screens

To define these screens, we just need to define two functions (one for welcome screen and another one for credits screen).

```
public static void WelcomeScreen()
{
    ...
}

public static void CreditsScreen()
{
    ...
}
```

The first one should be called at the beginning of Main method, before drawing anything on the screen. The last one must be called after the game loop. You can set them as the example images shown in previous section.

3. Multiple levels

In order to set multiple levels, we need to define an array of levels, this is, a bi-dimensional array of bricks, in which every row of the array will correspond to a level. You could replace the CreateBricks and DrawBricks functions with new functions that return and use a bi-dimensional array of 3 level (rows). In this example we define a first level with just one brick, a second level with 10 bricks (code is omitted), and a third level with 40 bricks (code is also omitted). Try to adapt this functions to your preferred level designs:

```
public static brick[][] CreateBricks()
{
    brick[][] bricks = new brick[3][];

    bricks[0] = new brick[1];
    bricks[0][0].x = 0;
    bricks[0][0].y = 0;
    bricks[0][0].destroyed = false;
    bricks[0][0].numberOfHits = 1;
    bricks[0][0].color = ConsoleColor.Red;
    bricks[0][0].score = 10;

    bricks[1] = new brick[10];
    …
    bricks[2] = new brick[40];
    …
    return bricks;
}
```

Regarding DrawBricks, we would need to specify which level we want to print, and then print just this level within the function:

```
public static void DrawBricks(brick[][] bricks, int level)
{
    ...
}
```

Within the game loop, you may need an additional integer variable to store current level, and then just refer to this row of the bricks array to check collisions and destroy bricks. You may also need an additional function to check if all the bricks of a given level have been destroyed, and then move to next level (or exit the game loop if there are no more levels in the array).