AWB

Course: Web Animation

Project title	DanBreakoid
Hours required for completion:	20 hours
Project code	420-AWB-ID Project
Revision date	2021-04-21

Project

Instructor(s)

AWB - DanBreakoid

INTRODUCTION

You have just finished the course and, without knowing it, you have already set up the bases for your project. This project is a little different because it will give you the chance to integrate several concepts learned during the previous exercises and to apply some of these concepts the basics of the brick-breaking game seen at the end of the course. The goal here is to make this game seem more alive and attractive for the players. You have to envision this project as something that could be presented to a child aged between 7 and 77 years old when it's done. If you did your job right, that kid should be able to easily play the game and unconsciously understand all the usefulness of the graphical elements that you added. Also, this person should especially be impressed by the fluidity of your web animations which should make the game even more captivating! The full version of the game will give you the opportunity to perfect your abilities to use CSS, HTML, but above all JavaScript and the different types of animations that can be integrated into a website and certainly in this little game which could be hosted on the web. The game will be called ***DanBreakoid*** which is a tribute to two things. First, the classic **Arkanoid** game created by Taito in 1986 and second, one of our strengths as a programming instructor at CDI College and a huge fan of this kind of classic arcade game.

OBJECTIVES

The objectives of this project are:

- To integrate different types of Web animations in a website/game.
- To create HTML, CSS and JavaScript documents.
- To create links and use JavaScript libraries.
- To use the JavaScript animation engine *Anime.js*
- To sketch attractive graphical interfaces that make use of images and animations.
- To use web elements to clearly display information.
- To create animations that will spark interest and guide the users.
- To correctly structure your website's files and folders.
- To style your web pages using CSS and JavaScript.
- To create JavaScript functions that will make the game/web site simple but efficient.
- To use event handlers that will let the JavaScript interact with the user's actions.
- To simply have fun using your creativity.

Allowed time

You have 20 hours to complete this project.

AWB - DanBreakoid

Key Elements

Here are the requirements of the game; they will be classified into three categories, **mandatory items**, **additional content** and **bonus items**. Help from instructors will only be given for those items that are mandatory for the project and even there, help is still likely to cost you points. Note that you will not get help for add-ons and bonuses. Here is the list of the components that must or can be added to your ***DanBreakoid*** game. You are always free to add further functionalities as time permits without compromising the following fundamentals of the game:

Mandatory Items Section (70 pts)

- The website/web game should only contain one HTML page named **index.html**. All the scenes of the game and even the presentation takes place on a single page. This doesn't mean that your site won't have several other files like CSS, JavaScript or images and library directories. It only means that there will be only **one** page with the .html extension in your site.

- First of all, the index.html page must display a presentation page commonly called a "**splash screen**" in games. On the Web, a "**home page**" is the page on a website that serves as the introduction page before the main page is showed. In our case, this page will serve as information to encourage the player to start a game but it will also help to prepare the environment for the game which will start after the ***Start*** button is clicked. Think of this page as when you are playing a video game and at the very beginning there is a short presentation of the title with an animation that tells us to press "**Start**".

    - This page must contain and focus on showing the name of the game. (A single title at the top of the page or included in an image).
    - A background image, pattern or color must be used for the entire HTML page to make the game look nicer. We will not let our canvas stand on a white page.
    - A canvas of appropriate dimension surrounded by a **very** visible border will contain the game at a later point.
    - A ***START GAME*** button to let the player start the game.
    - First, the canvas should appear sliding across the screen to create an effect of dynamism and surprise. From top to bottom or left to right, it's up to you to decide what effect you want to demonstrate. One thing is certain however, is that the canvas must end its course centered on the page.

- Then, an animation on the button should be made. The button should appear after the canvas has come to a stop and should shake or inflate and deflate. In short, it should produce an animation to grab the player's attention so they know they should click on it to start the game. The movement of the button must be perpetual over time and repeated in a loop every 3 to 5 seconds.

***Note**: You have an example of this page named "**Splash screen**" in the appendix at the end of this document. You do not have to use the images provided with this project. You can find your own images, as long as you follow the demands of the project. Creativity is always welcome.*

□ When the player clicks on the «**Start Game**» button, he or she must then see an animation that represents a fake page loading. You can use any images or techniques that you have learned in the course to achieve that. This page loading must last **4** seconds before the game starts. You could use a rotation effect using a spinning circle or hourglass. If you prefer to go ***old school***, a loading bar that fills itself ¼ more at each second would be fine.

***Note**: You have an example of this page named "**Loading**" in the appendix at the end of this document. You can take a different approach than this, but keep in mind that the goal is to give the player the impression that it's waiting for the first level of the game to load. So use an appropriate animation!*

□ Then, in the next step, the first level of the game must be displayed. The game must let the player try to break at least 30 bricks using a ball moving inside the canvas.

- This page must open on a new background image in the canvas (bg_level1.png) and must contain a paddle with a ball placed in its center. **Warning!** Unlike in the original exercise, the game does not start immediately. The paddle and the ball are stationary and centered at the bottom of the game. The paddle can be moved at any time from left to right with the keyboard's **A** and **D** keys or by moving the mouse pointer, even if the game has not started yet. The ball remains stuck on the paddle as long as the player does not decide to throw the ball by pressing the **space bar**. This is when the ball takes off towards the bricks and the game begins.

- There must be at least 30 bricks in the first level (10 X 3). But you are free to add more. You can adjust the size of the bricks to give your game a certain look, but each one should be the same size compared to the others. Basically, they can be drawn without using images or sprite sheets. They can be created as in the course exercise. You will have the opportunity to improve this aspect in additional parts of the project.
- Same thing for the ball and paddle. You can draw them for now in the same way they were drawn in the original exercise (you will have the opportunity to use images later in the project). For now, you just have to worry about the game being functional. Of course, more points will be given for work done using more advanced aesthetics.

***Note**: You have an example of this page named "**First Level**" in the appendix at the end of this document. As it was said before, your game can look a bit different than this example but you absolutely need a minimum of 30 bricks and the paddle should be able to move with the ball stuck to it in its middle until the player presses the space bar.*

□ During game play, the ball must always remain within the limits of the canvas and bounce off bricks and walls except when it reaches the lower edge of the playfield. At this point the ball disappears and reappears in the center of the paddle, the player has just died. At this moment the two elements (paddle and ball) are reset and are found at the bottom in the middle of the game window ready to start another round.

- The player has 3 lives at his disposal. The number of lives remaining should be displayed outside of the canvas at the top right of the screen. Each time the player misses the ball with the paddle and it goes beyond the lower limit of the canvas, the player loses a life and the number of lives displayed on the screen must be updated. The ball then automatically reappears on the paddle, stuck in the center, and the player should be able to throw the ball back by pressing the space bar again. Now very important, when the player has 3 lives, ***the number of lives*** display color should be green, when 2 lives remain, the color changes to orange and red when only one life remains. But at this point, the display should also flash to create additional suspense and force the player to increase its stress level, which should make him or her miss the ball more easily!
- Each time the ball hits a brick, the brick should disappear and the player scores 10 points thus visually increasing its score on screen automatically. The score is located just below the player's health bar and not on the canvas. Since each brick is worth 10 points and the first level must have a minimum of 30 bricks, if the player manages to destroy every bricks on screen, the score displayed should be at least 300 at the end of the first level.

- The particularity of this game is to let the player believe that the ball has a natural behavior when bouncing off objects. Create one or more functions to direct the ball at an angle as soon as it comes in contact with a brick, a wall or the paddle. The ball should move at a constant speed for the basic version of the game.

***Note**: You have an example of this page named "**game**" in the appendix at the end of this document. As mentioned before, your result can be a little different than the example but keep in mind that the ball must bounce off the walls, bricks and the player's paddle who must try to keep the ball in play for as long as possible to make all the bricks disappear. The score and number of lives remaining must be updated in real time during the game.*

□ The paddle board must be controlled by the **A** and **D** keys for players who prefer to use the keyboard, but it must also be controlled by the mouse pointer. However, in this version of the game, the paddle will only react when **the pointer is above the canvas** and not elsewhere in the page. The paddle will respond instantly to movement of the mouse pointer over the playing surface but will remain motionless when and where the pointer leaves the canvas. Also, be careful when moving sideways, either by using the keyboard or the mouse pointer. When the paddle approaches the walls, make sure that it doesn't exceed the limit of the playing surface and remains entirely in the canvas. In the original exercise version of the game, half of the paddle disappeared outside when reaching a wall. That is not allowed, and that is not what we want here, we want refinement.

□ The game must announce to the player its success or failure. **No JavaScript alerts are allowed!** The game must come alive with its visuals and, above all, intuitive messages. The player should be able to easily tell that it has passed the level when all the bricks have been broken. A transition appears with a ***congratulations!*** message. However, he should just as well distinguish during the game that he lost one of his 3 lives or even all of his lives before he destroyed all the bricks. At this time, a negative transition occurs in the page. Use animations on the life bar and score. Also, animate images or text right in the middle of the playing field to inform about the success or failure of the player.

- If it's a success, the first level must reload itself while keeping the same score and number of lives remaining

***Note**: You have an example of this page named "**success**" in the appendix at the end of this document. As usual, your result might be a little different from the example but keep in mind that the player must see text or an animated image informing him of his success. This message should be displayed at least 5 seconds and then the first level reloads, keeping the same score and lives.*

- If it is a failure, the famous **GAME OVER** is displayed and you offer the player the possibility of retrying the experience with the "**START GAME**" button from the beginning. You then reload a new game, the first level with the score back to 0 and the 3 available lives.

***Note**: You have an example of this page named "**Fail**" in the appendix at the end of this document. The player must realize that the game has been lost. The game must display text or an animated image informing the player of the loss and making fun of the situation to stay in the spirit of the games created at the time. The **START GAME** button should reappear after a small delay. The player can then choose to play again, but with a reset in its score and available lives.*

Additional Content Section (30 pts)

- □ An Image is used to generate the ball and paddle. (5 pts)
- □ A sprite sheet is used to display the bricks. (5 pts)
- □ A sprite sheet is used to display the life bar counter. (4 pts)
- □ The score display moves each time the player cumulates 100 points. (4 pts)
- □ If the player succeeds in breaking all the bricks, the game offers more levels (up to three) with 3 different background images and 3 different brick patterns showed to the player at the start of each level. (4 pts)
- □ The space bar, aside from its primary function of launching the ball, can also pause the game by freezing the ball and the paddle. When pressing it again, the game resumes its course. (2 pts)
- □ Bricks of different colors have different strengths. As an example: A black brick could take up to three hits by the ball before being destroyed, a gray brick could take two hits and the rest of the colors only one. (2 pts)
- □ The angle of the ball varies considerably depending on the angle at which it is struck by the paddle. (4 pts)

Bonus Items Section (8 pts)

- □ A reward or penalty system involving special bricks is put in place. When these special bricks are destroyed in the game, the paddle doubles in width or just shrinks to half its size. These magic bricks could also slow down or speed up the ball. The bricks must be easily recognizable by the player. (5 pts)

- □ Using your own images and theme. (3 pts)

Files

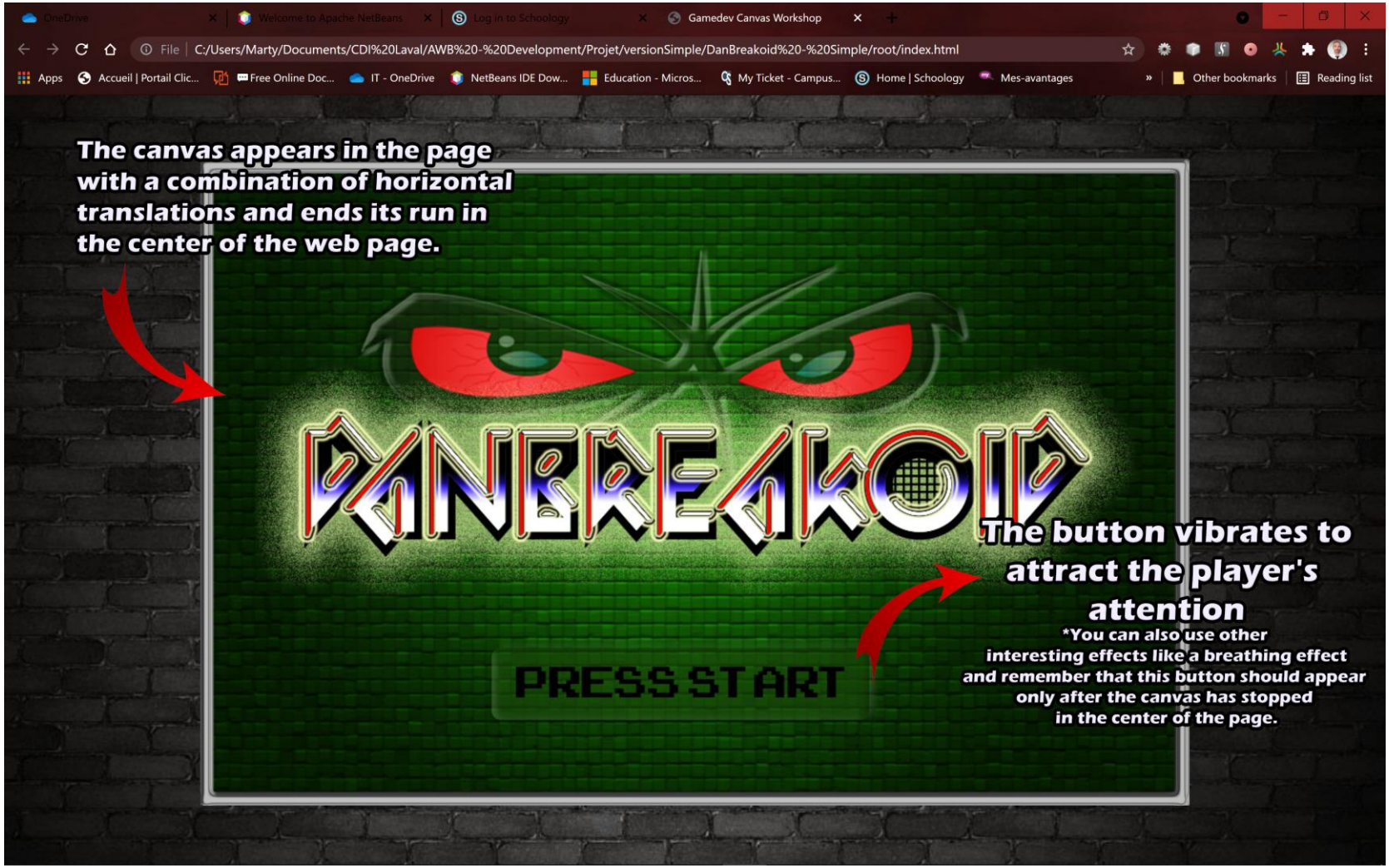
The file which contains the images necessary for the realization of this project has been given to you. On the other hand, it is strongly recommended that you use your own images which will coordinate more nicely with the theme and style chosen for your **DanBreakoid** animated game on the web. You will surely need the **Anime.js** library to help make more complex animations in this project so the **anime.min.js file** has been included in the available resources.

Be creative and good luck!

Annexe

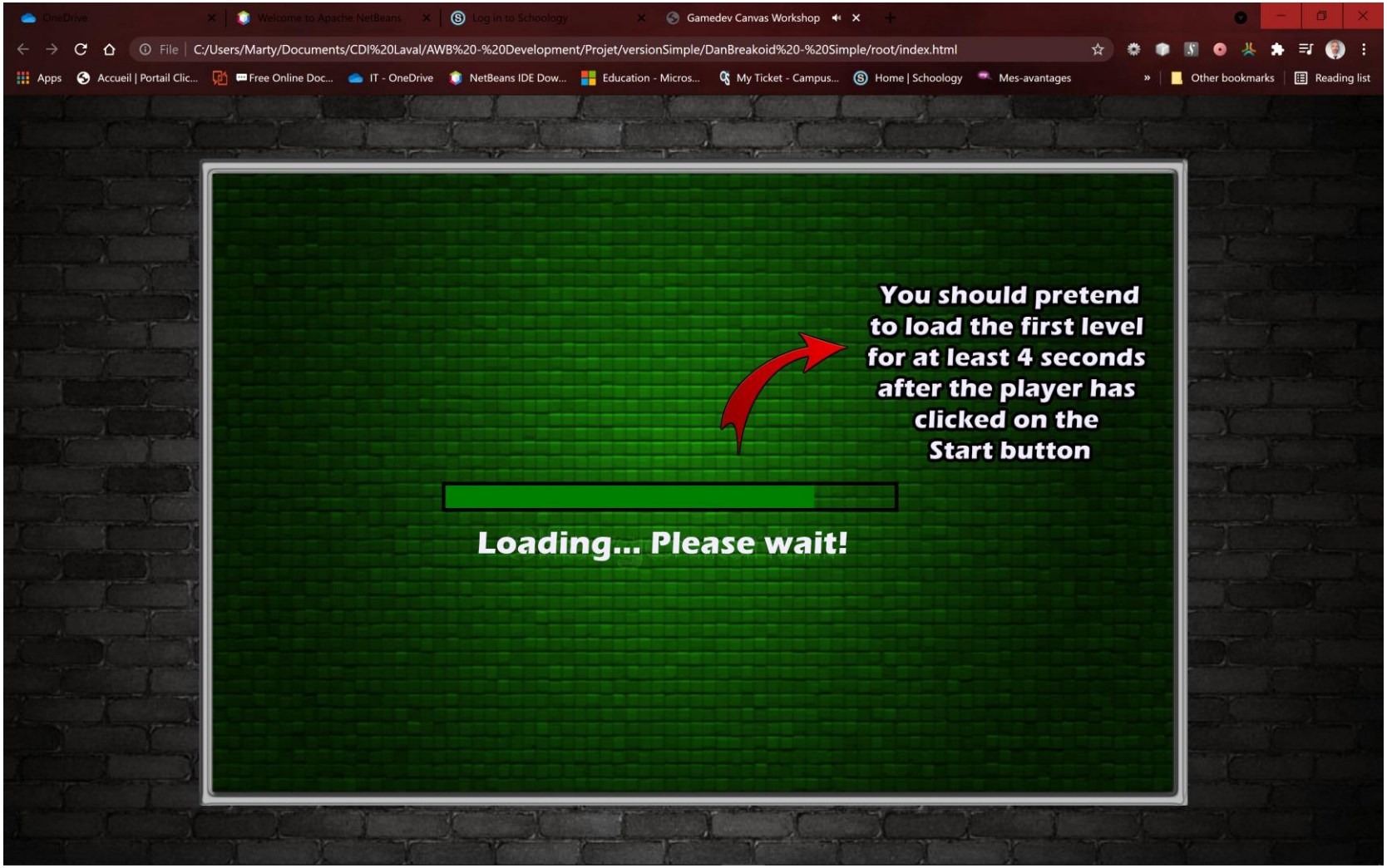
AWB - DanBreakoid

Splash screen



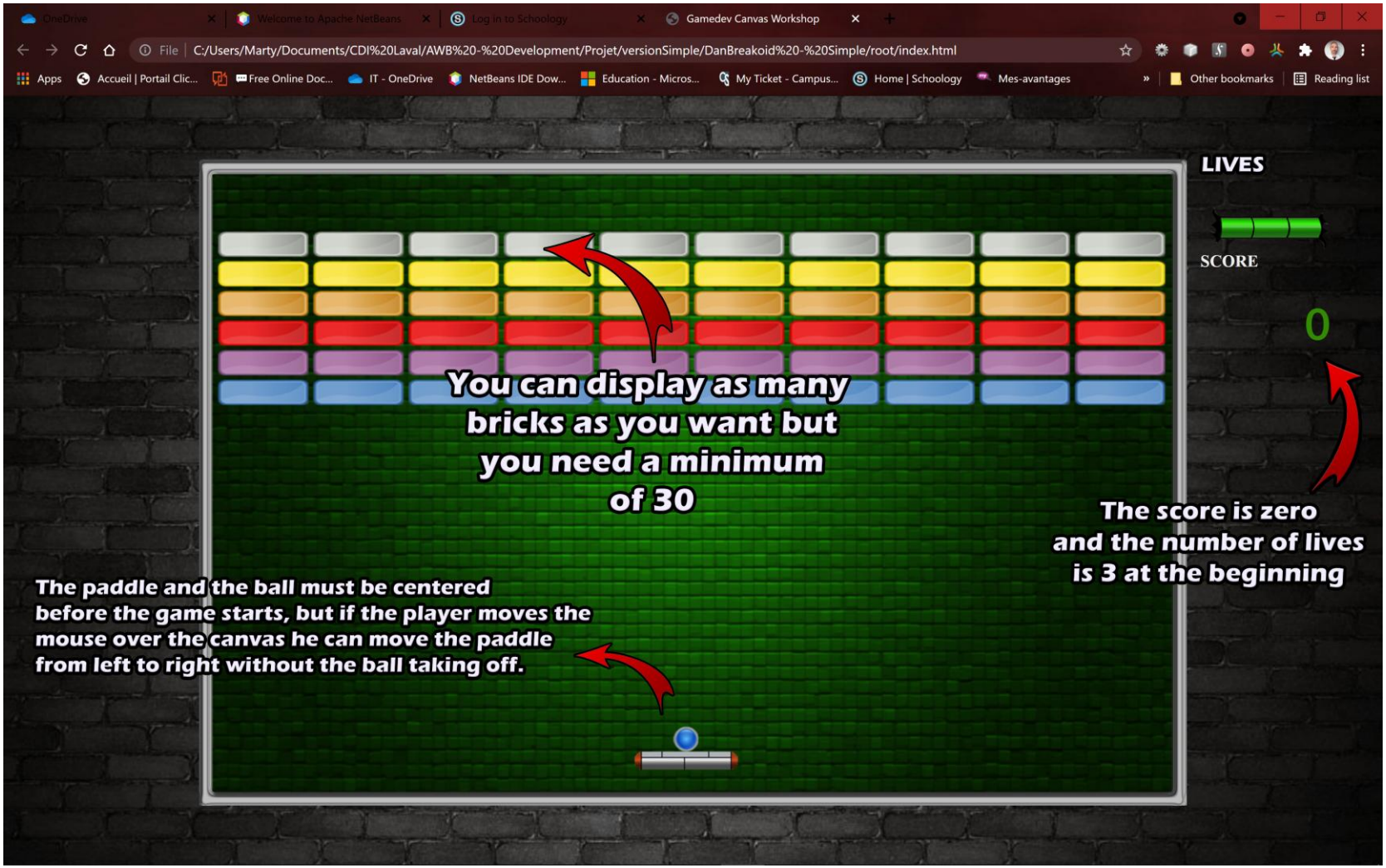
AWB - DanBreakoid

Loading



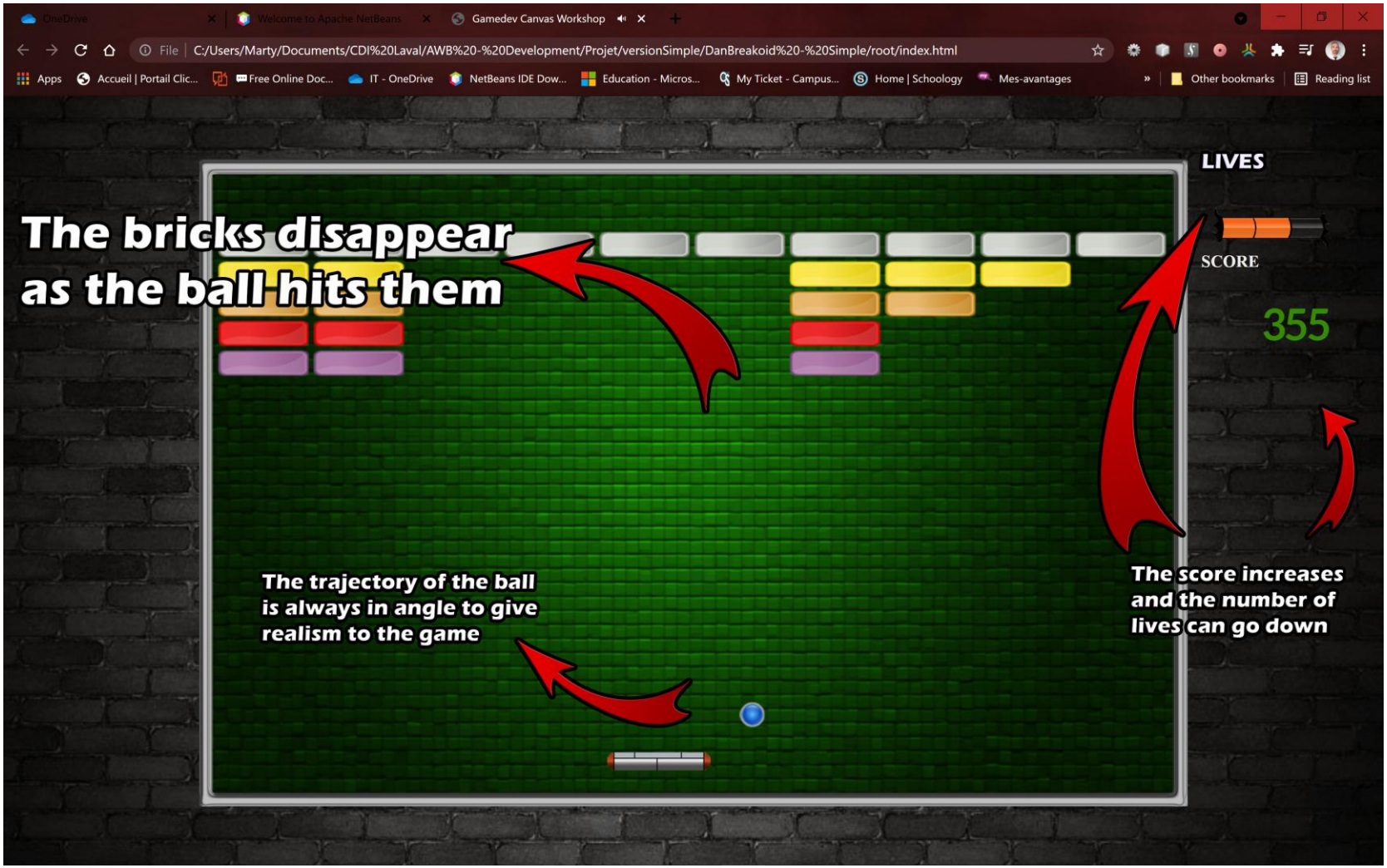
AWB - DanBreakoid

First level



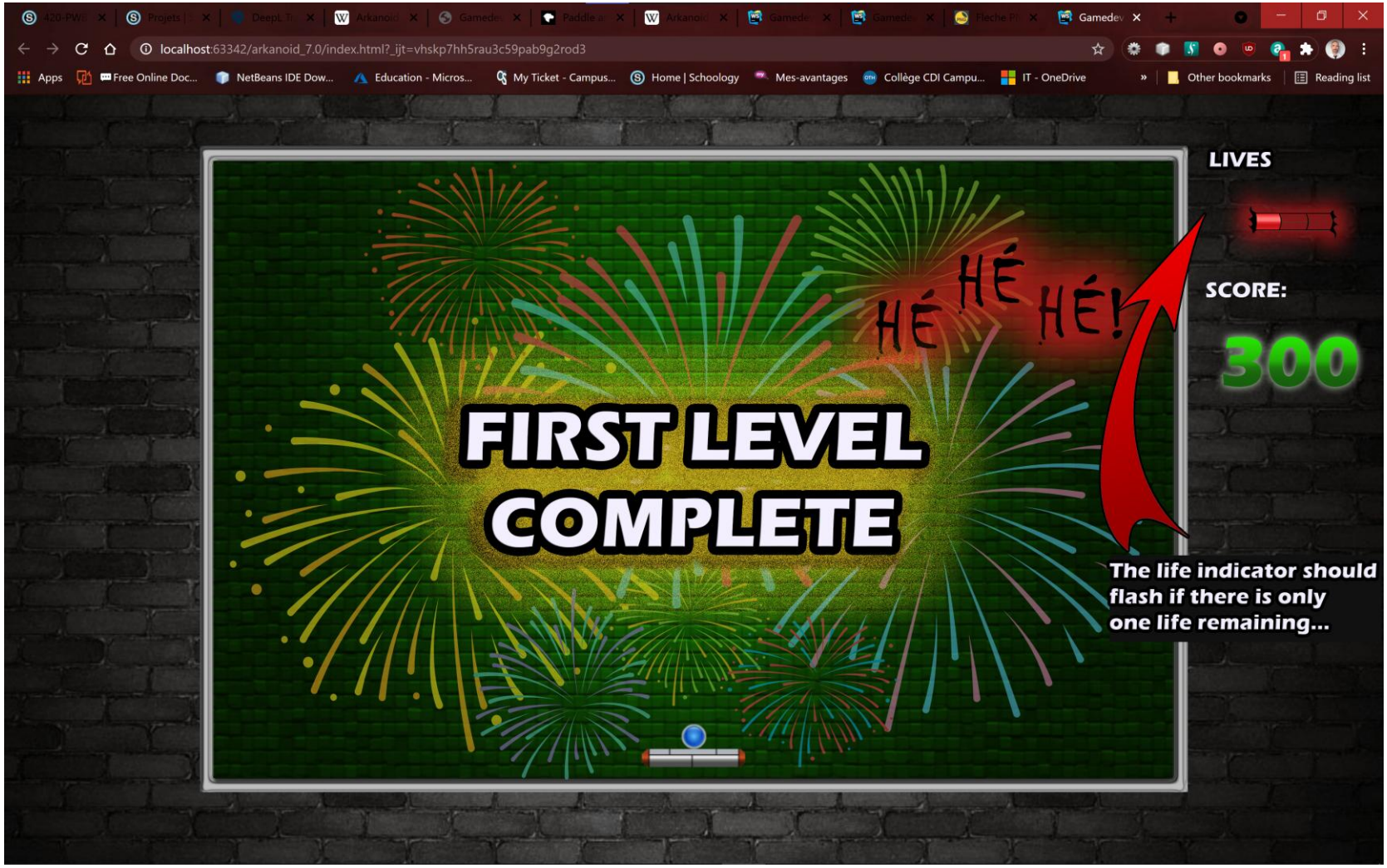
AWB - DanBreakoid

Game



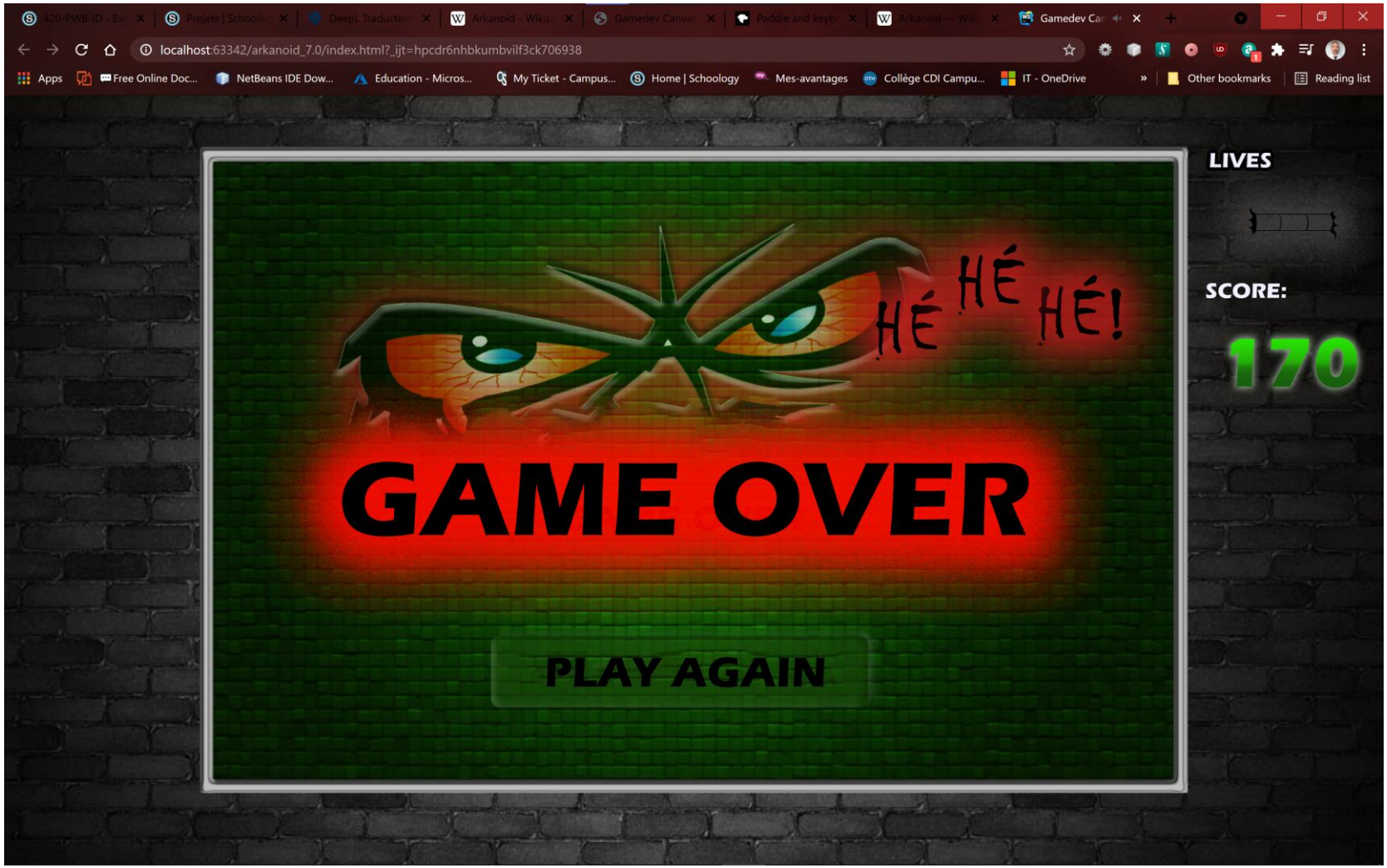
AWB - DanBreakoid

Success



AWB - DanBreakoid

Fail



AWB - DanBreakoid

Mandatory items (70%)

(Everything mentioned in this section should be completed and functional before going to the supplementary options)

1. *All the files are organized in separate folders (HTML, CSS, JS, lib…)* ***(3 pts)***
2. *Only one index.html file is used for the game and everything else, including the* ***splash screen****.* ***(3 pts)***
3. *On the* ***splash screen****, the canvas appears using one or many translations.* ***(4 pts)***
4. *On the splash screen, a button appears on the page after the canvas has stopped moving. Then, the button itself starts moving to grab the player's attention by shaking, inflating and deflating, changing colors, etc. This animation doesn't stop as long as the player doesn't click the button.* ***(4 pts)***
5. *The* ***Start*** *button triggers a fake animated loading effect that lasts 4 seconds.* ***(4 pts)***
6. *Displaying the background image of the first level and a minimum of 30 bricks.* ***(4 pts)***
7. *Displaying the paddle with the ball sticking on top at the middle of the paddle.* ***(4 pts)***
8. *The ball doesn't leave the paddle if the player doesn't press the space bar.* ***(4 pts)***
9. *The paddle is controlled using the keyboard or mouse pointer. When using the mouse, the paddle only moves if the pointer remains within the canvas. Also, the paddle should remain entirely inside the canvas and not have part of it disappear behind a wall.* ***(4 pts)***
10. *The balls remain within the limits of the game and bounces off walls, bricks and the paddle.* ***(4 pts)***
11. *The score and player lives counters are displayed in the top right section of the page, outside of the canvas.* ***(4 pts)***
12. *The score and lives are initialized to 0 points and 3 lives.* ***(4 pts)***
13. *The ball moves at an angle.* ***(4 pts)***
14. *The ball can destroy bricks in the game.* ***(4 pts)***
15. *Each destroyed brick adds 10 points to the player's score.* ***(4 pts)***
16. *If the paddle misses the ball and it reaches the bottom of the canvas, the player loses a life.* ***(4 pts)***
17. *A success screen displays in the canvas if a player finishes a level. The player also keeps its score and lives when the level starts anew.* ***(4 pts)***
18. *A failed animated screen appears in the canvas if the player loses its 3 lives. The button to let the user play again must reappear after the animation is done, and when the level reloads, the score and lives counters are reset.* ***(4 pts)***

Additional content (30%)

1. An Image is used to generate the ball and paddle. (5 pts)
2. A sprite sheet is used to display the bricks. (5 pts)
3. A sprite sheet is used to display the life bar counter. (4 pts)
4. The score display moves each time the player cumulates 100 points. (4 pts)

5. If the player succeeds in breaking all the bricks, the game offers more levels (up to three) with 3 different background images and 3 different brick patterns showed to the player at the start of each level. (4 pts)

6. The space bar, aside from its primary function of launching the ball, can also pause the game by freezing the ball and the paddle. When pressing it again, the game resumes its course. (2 pts)
7. Bricks of different colors have different strengths. As an example: A black brick could take up to three hits by the ball before being destroyed, a gray brick could take two hits and the rest of the colors only one. (2 pts)
8. The angle of the ball varies considerably depending on the angle at which it is struck by the paddle. (4 pts)

Bonus items (8%)

1. A reward or penalty system involving special bricks is put in place. When these special bricks are destroyed in the game, the paddle doubles in width or just shrinks to half its size. These magic bricks could also slow down or speed up the ball. The bricks must be easily recognizable by the player. (5 pts)

2. Using your own images and theme. (3 pts)

Penalties

- *5 % deducted for each day late.*
- *For any delay of more than three days up to 5 days, the maximum score will be 60%*
- *The project is failed automatically at the 6th day of lateness.*