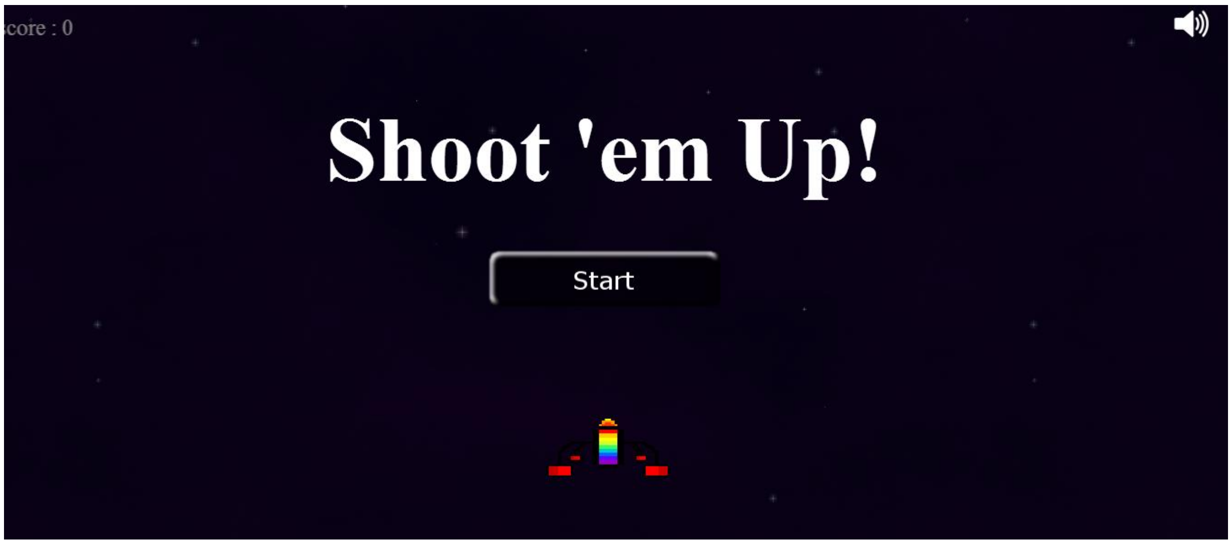
MMS-Project 2
Game Development using HTML5

Prepared by :

1) Ahmed AlaaElDeen Abdel Fattah	828
2) Mazen Mohamed Melouk	1067

i) Complete Description of game logic:

The game starts with the main menu scene along with the background music as a start, the page waits for the user to click "fire a bullet at the start button to initiate the game.



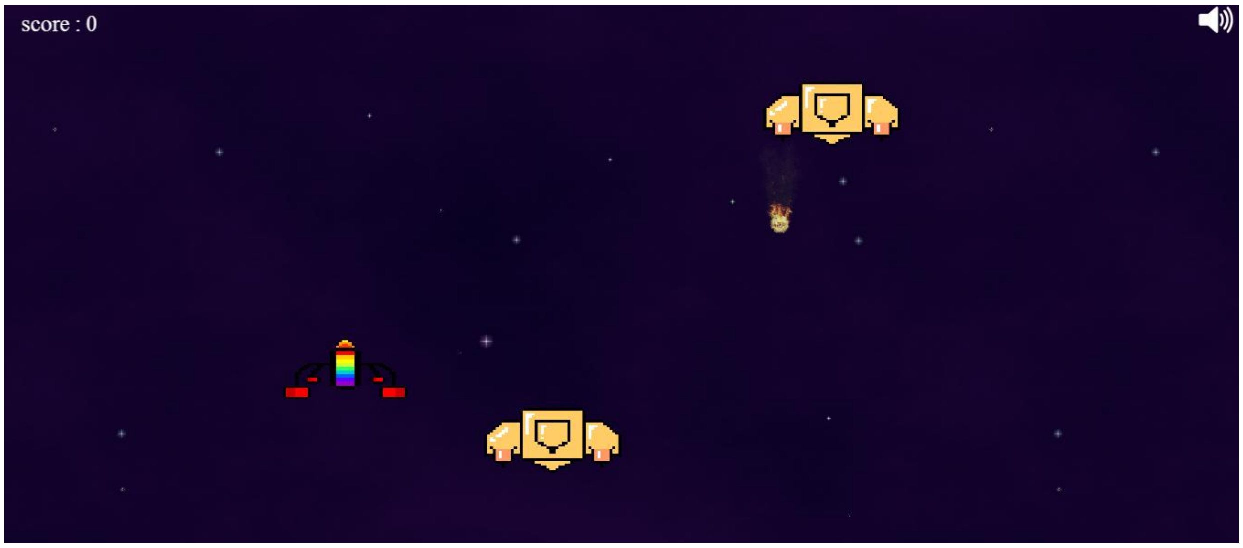
Then the game start with basic level 1 which includes the following:

1- Special sliding background
2- Special background Music
3- Simple Enemies that just travel vertically downwards, player loses if a collision occurs
4- Falling FireBalls, these are fire balls that fall at a high speed & could destroy the player
5- Rewards that occurs every multiple of the score these includes
   - A shield that is valid until the player colloides with an enemy or a fireball
   -A round of Multi-Bullets, fire 3 bullets instead of one with a higher score for special hits
   -A blast bomb, this blasts all enemies currently on screen.

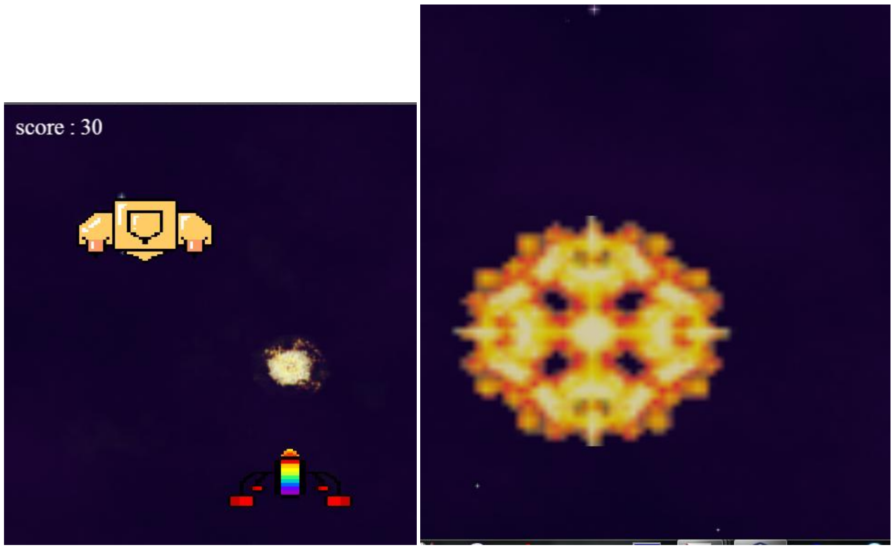
Generally, the game can be paused/unpaused, mute/unmute, exit to main menu via keyboard buttons "p","m" & "c" respectivly.
However the game automatically pauses when navigating away from the game page/tab.

After collecting a certain score the player is moved to level 2 which has same features as level 1 & additional features:

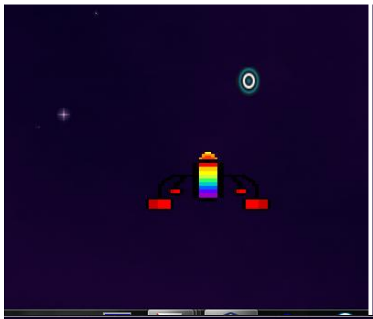
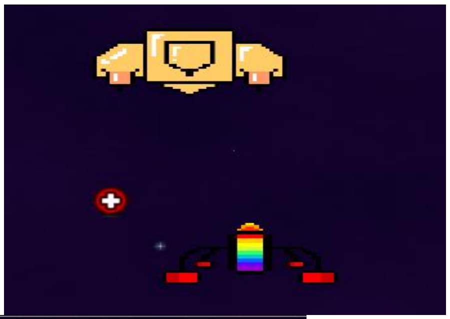
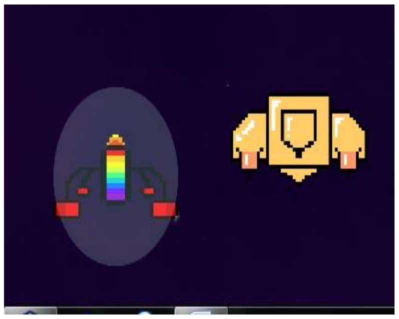
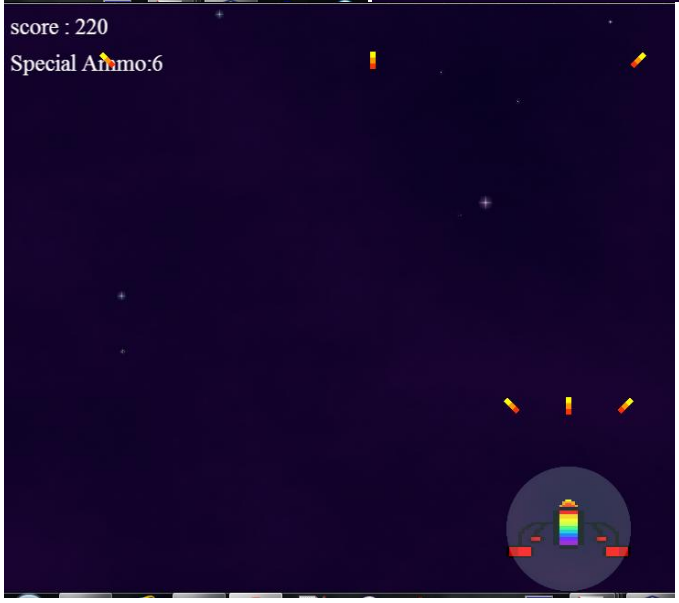
1) Fireballs & enemies are created & move faster
2) Smarter enemies which move to towards the player's position increasing the difficulty
3) Smarter enemies need 2 hits to die



A scene from level 1 where there is a player, 2 enemies & a fire ball

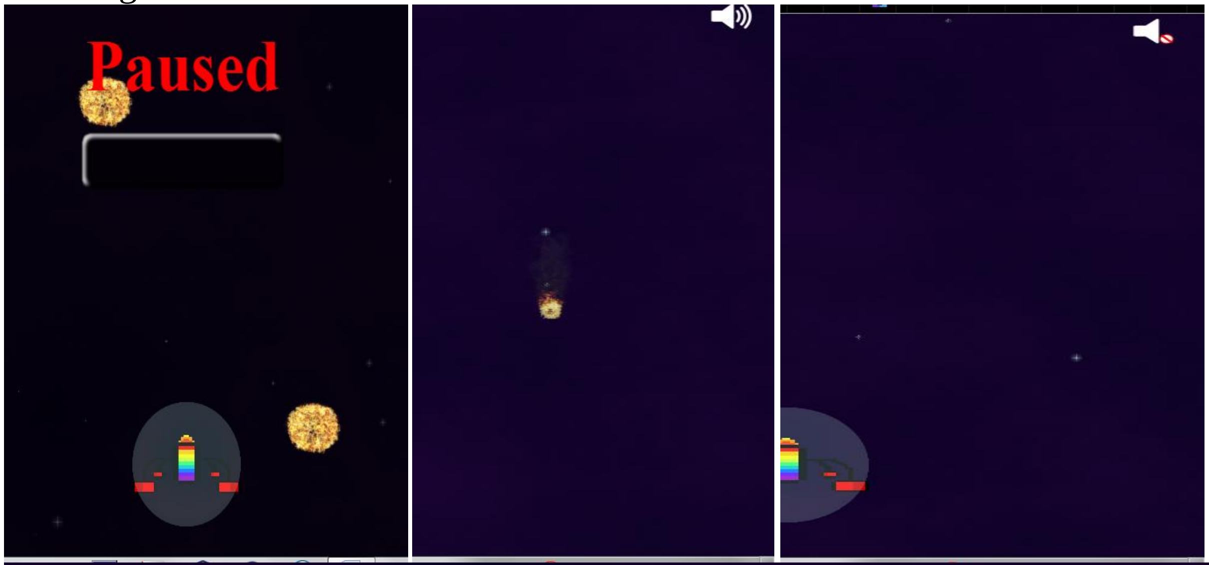


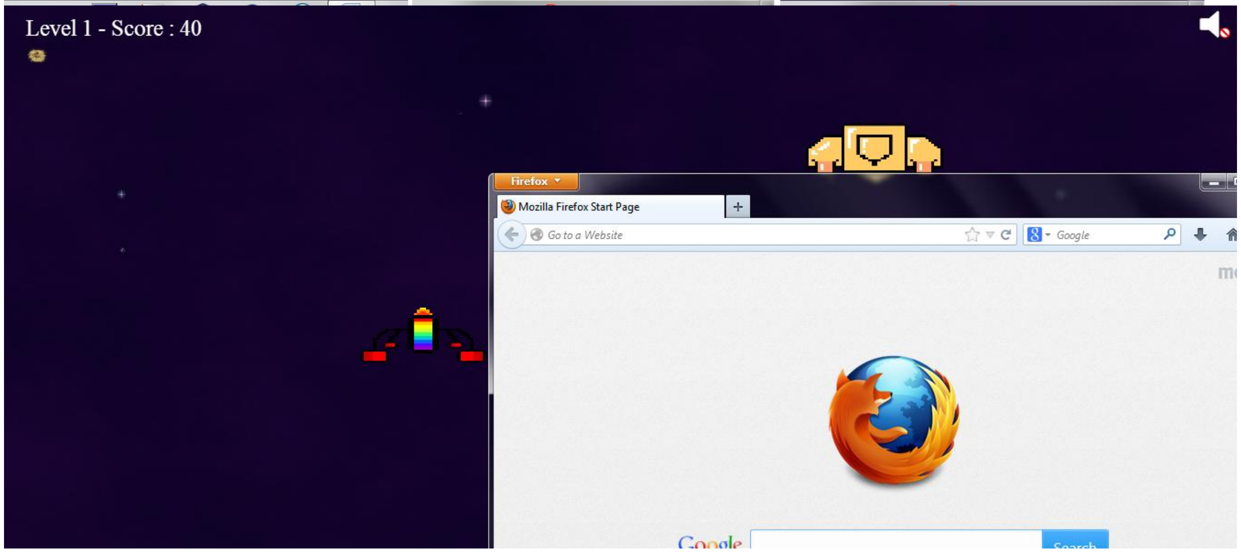
A scene showing the blast caused by hitting an enemy , on left & it is animated by varying multiple images with a increasing width & height while decreasing transparency "alpha" till disappearance

These image show rewards falling & their effects, the blue one would blast all enemies on screen, red one would shield the player & protect it from losing if collided with an enemy or fireball. Finally the multi-bullet function allows the player to fire 3 bullets with an angle & a definite number as shown below the score.

Another aspect is the game controls, when pausing a black screen with a small alpha pops with paused in the middle & a resume button, also muting/unmuting could be observed via the speaker icon in the top right corner .
Also navigation away from the window shall pause the game until the cursor is back to game screen.





ii) Technology used:

As required " pixi.js " as a 2D webgl renderer was used

https://github.com/GoodBoyDigital/**pixi**.js

Also "jQuery.js" is used for some extra features & events as timer function & clicks

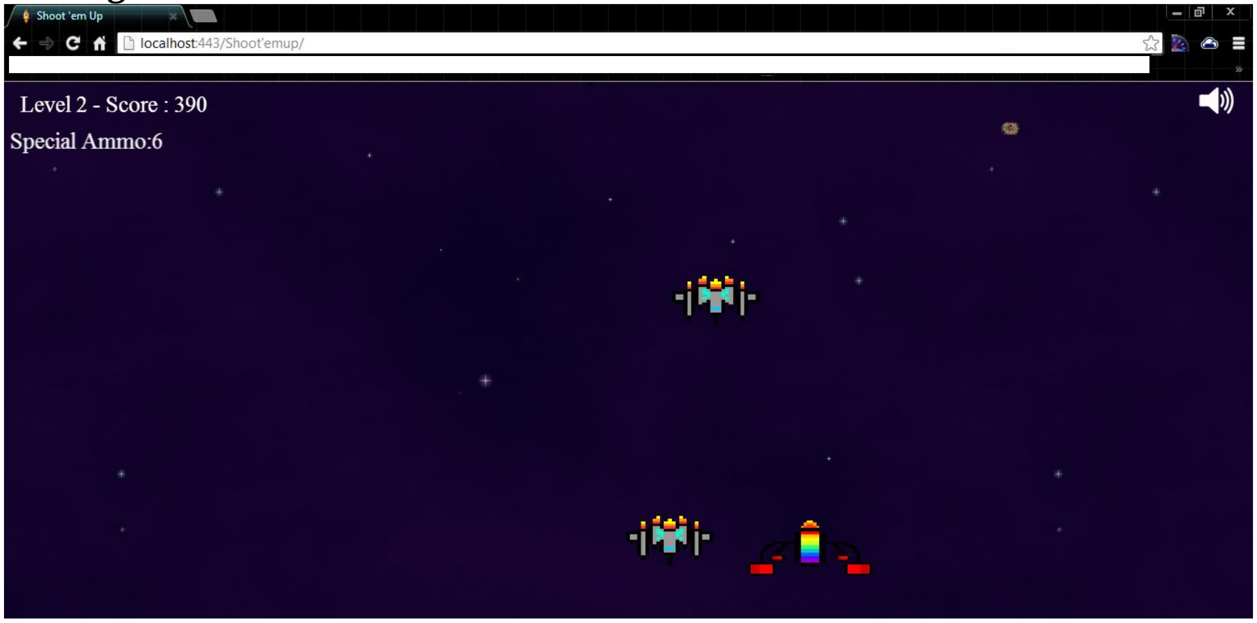
http://jquery.com/download/

NetBeans as IDE for a smooth coding experience

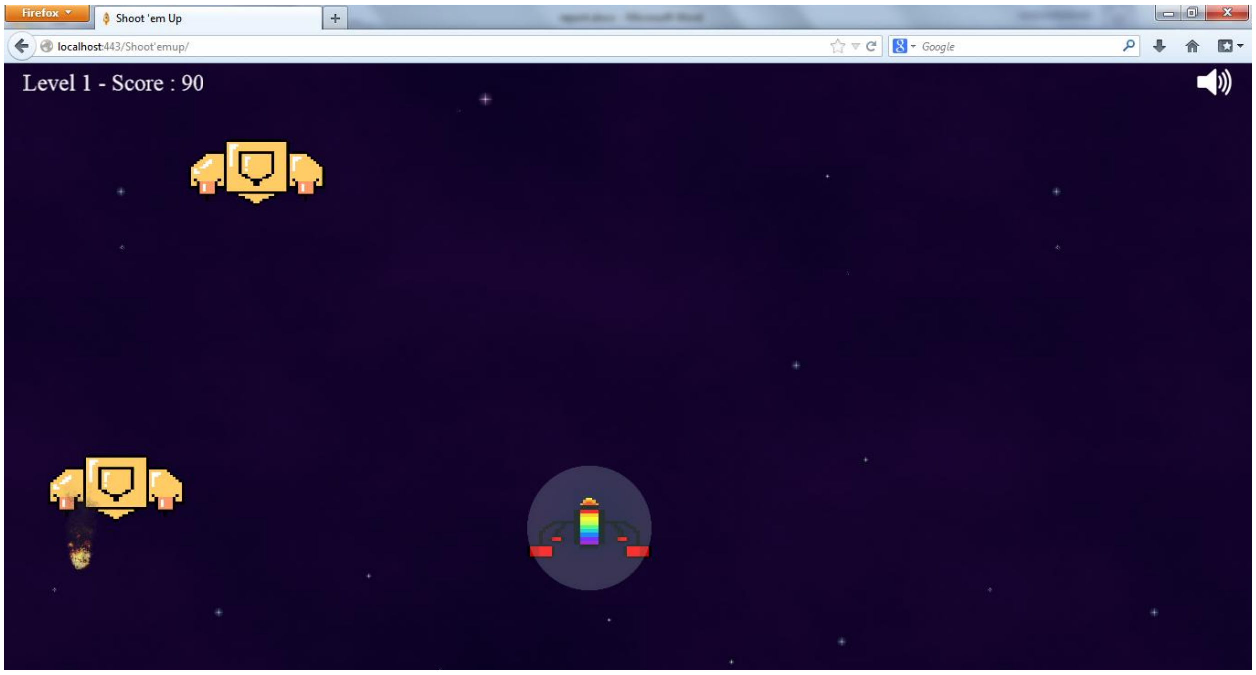
Google Chrome Debugging tools

iii) Testing on different browsers :

a- Google Chrome :



b-Fire Fox :



iv) Code Snippets :

The project folder is divided into folders according to contents that is js for javascript files , css for style sheets, img for images ....etc

Most of the work is through java scripts, apart from downloaded scripts "pixi.js" & "jQuery.js" , other files are labeled to give a hint about what their function is for example :
global.variables.js , would have all global variables defined
sound.js, would have all functions controlling sound & volume.

Most of the code is "event.detections.js" which detects events such bullets hitting an enemy, player colliding with an enemy, code snippets from the file are shown below:

```
function detectPlayerFireBallCollision() {
   for (i = 0; i < fireballs.length; i++) {
      if (detectCollisionFireBall(fireballs[i].sprite, Sprites.player)) {
         if (shielded) {
            shielded = false;
            stage.removeChild(Sprites.shield);
            stage.removeChild(fireballs[i].sprite);
            fireballs.splice(i, 1);

            return;
         }
         dead = true;
         $("body").css("cursor", "auto");
         playSound("bigBlast");
         Sprites.playerBlast.position = getCenter(Sprites.player);
         stage.removeChild(fireballs[i].sprite);
         fireballs.splice(i, 1);
         stage.addChild(Sprites.playerBlast);
         stage.removeChild(Sprites.player);
         return;
      }
   }
}
```

detectPlayerFireBallCollision is a function to check if player collides with fireball, if player is shielded ,then remove shield else blast to animate players death

```
function detectBulletEnemyCollision(bulletsArray) {
    var i, j;
    for (j = 0; j < enemies.length; j++)
        for (i = 0; i < bulletsArray.length; i++) {
            if (enemies[j].injuries >= enemyTypes.maxInjuries)
                return;
            if ((detectCollision(bulletsArray[i], enemies[j].sprite) &&
getBottomRight(enemies[j].sprite).y > 5)) {
                var enemyBlast = {
                    animIndex: 0,
                    associatedEnemy: enemies[j],
                    deltaXY: getCenter(enemies[j].sprite)
                };
                enemyBlast.sprite = new PIXI.Sprite(new
PIXI.Texture.fromFrame(cacheIndices.explosion1.start));
                enemyBlast.sprite.anchor.x = enemyBlast.sprite.anchor.y = 0.5;
                enemyBlast.sprite.position = getCenter(bulletsArray[i]);
                enemyBlast.sprite.width = enemyBlast.sprite.height = enemies[j].sprite.width / 2.5;
                score += 10 * (currentLevel + 1) * (enemyTypes[enemies[j].type].scoreFactor)
                  * (enemies[j].injuries+1);
                if (score>=Level[currentLevel].scoreStep)
                    LevelUp();
                playSound("blast");
                enemyBlasts.push(enemyBlast);
                stage.addChild(enemyBlast.sprite);
                stage.removeChild(bulletsArray[i]);
                bulletsArray.splice(i, 1);
                enemies[j].injuries++;
                enemies[j].oscNo = 0;
                enemies[j].oscDir = 'right';
                enemies[j].oscPos = 0;
                if (enemies[j].injuries < 2) {
                    switch(enemies[j].type){
                        case 0:
                            enemies[j].redMask = new PIXI.Sprite(Textures.enemy1Red);
                            break;
                        case 1:
                            enemies[j].redMask = new PIXI.Sprite(Textures.enemy2Red);
                            break;          }
                    enemies[j].redMask.alpha = 0;
                    enemies[j].sprite.addChild(enemies[j].redMask);              }
                if (enemies[j].injuries >= enemyTypes[enemies[j].type].maxInjuries)
                    enemies[j].state = 'dying';
                else {                  enemies[j].state = 'hurting';              }
                if ( (score - prevScore)>=Level[currentLevel].bonusStep && sendGift) {
                    giftIsActive = true;
                    sendGift = false;
                    spawnPowers();
                    prevScore = score;
                }
                else {
                    sendGift = true;
                }
            }
        }
}
```

Some function from "event handelers.js" which handles events as mouse clicking or navigation away from screen are shown below:

```
function clickFunction() {
    if (cantClick || dead || paused)
        return;

    var bullet = new PIXI.Sprite(Textures.playerBullet);
    bullet.width = Sprites.player.width * 0.04;
    bullet.height = Sprites.player.height * 0.3;
    bullet.anchor.x = 0.5;
    bullet.anchor.y = 0.5;
    bullet.position.x = getCenter(Sprites.player).x;
    bullet.position.y = getTopLeft(Sprites.player).y - bullet.height / 2.0;

    stage.addChild(bullet);
    bullets.push(bullet);

    playSound("bullet");

    if (multigunned) {
        bonuslimit--;
        var right, left;
        right = new PIXI.Sprite(Textures.playerBullet);
        left = new PIXI.Sprite(Textures.playerBullet);
        right.width = left.width = Sprites.player.width * 0.04;
        right.height = left.height = Sprites.player.height * 0.3;
        right.anchor.x = left.anchor.x = 0.5;
        right.anchor.y = left.anchor.y = 0.5;
        right.position.x = left.position.x = getCenter(Sprites.player).x;
        right.position.y = left.position.y = getTopLeft(Sprites.player).y - bullet.height / 2.0;
        right.rotation = 0.78532981625;
        left.rotation = -0.78532981625;
        stage.addChild(right);
        stage.addChild(left);
        bulletsR.push(right);
        bulletsL.push(left);
        if (bonuslimit < 0)
        {
            multigunned = false;
            stage.removeChild(Texts.counterText);
        }
    }

    cantClick = true;
    $.timer(function() {
        cantClick = false;
    }).once(clickDelay);
}
```

v)Github link:

project files are uploaded on Github @
https://github.com/mazenmelouk/HTML5GameDev