Project 02

Superman : The Man Of Steel

Made by:

1. Nour El Deen Mohamed Nabil 1240
2. Samar Khaled Azzam 966

Game Logic Section:

The game is divided into two parts : single player and multiplayer.

1.single player:

Superman the hero saves the city from the invasion of the enemy's spaceships.

The main player (superman) has to avoid incoming spaceships,and destroy as many as he could,with initial Health Points (HP) of 3 and with possibility of collecting 1Ups to increase his HP,A stage is finished when a set number of enemies is destroyed using Superman's eye beams (activated using space-bar or left click). And in the 2nd level superman's speed increases and the fighting is taken up to a higher level, the moon.The second stage consists of harder and faster enemies with more HP of their own .

2.Multiplayer:

This is a contest between two challengers.each one is given the same HP and stage and left to destroy as many enemies before dying ,the winner is the player with most enemy kills.

*Game is fully operational with both mouse and key board controls.

Game Controls:

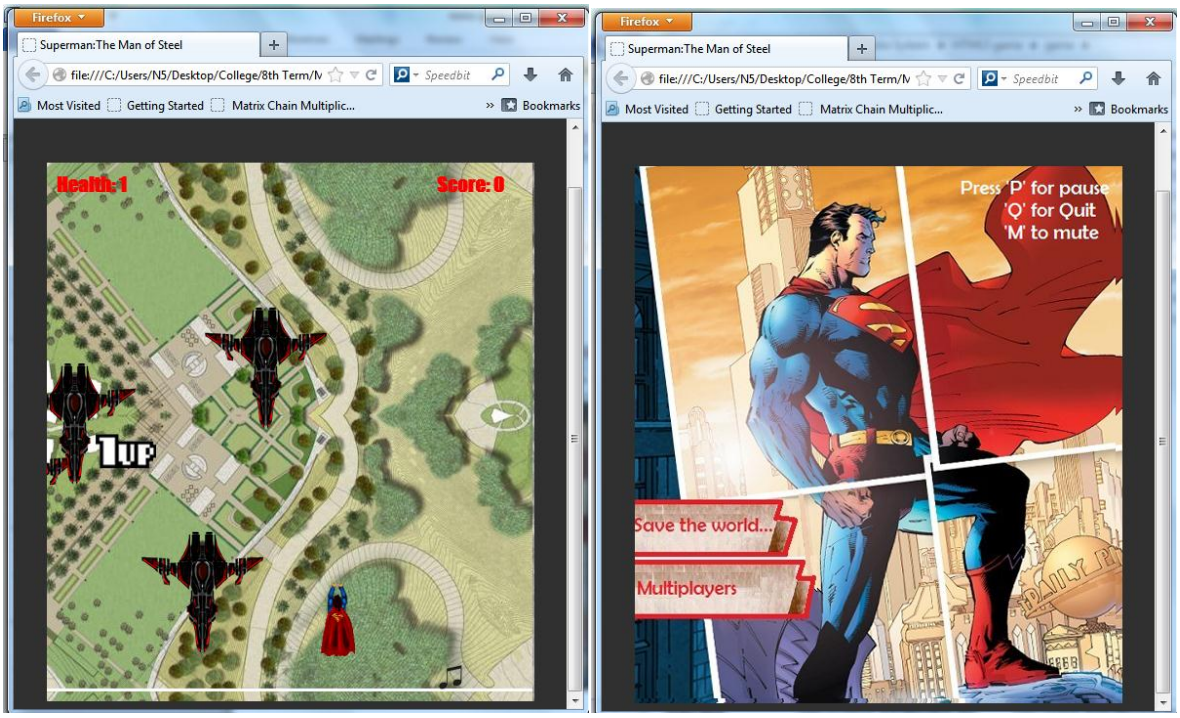
1. Mouse, arrow keys,wasd to control Superman.
2. Space or Left-Mouse button for firing.
3. M to mute music.
4. Q to quit to main menu at any time.
5. P to pause the game at any time.

Technologies used:

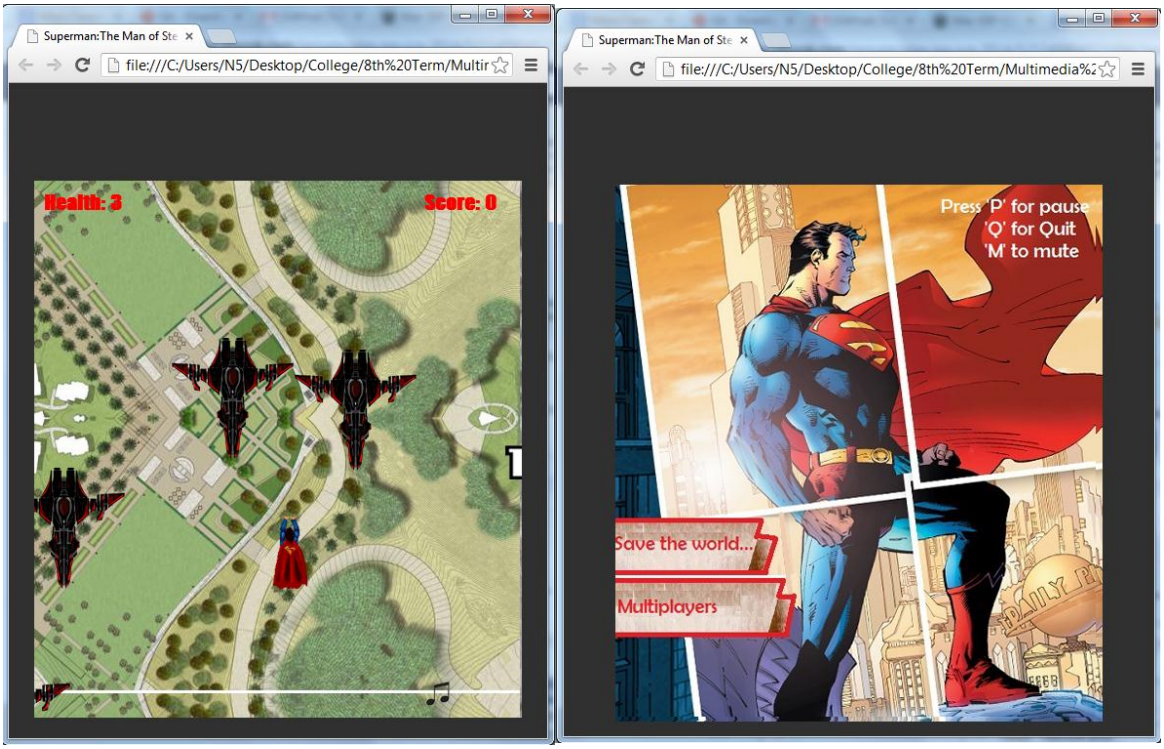
Basic HTML5 and Javascript with canvases.

Screenshots:

Firefox:



Google Chrome:



Code snippets :

Moving Background:(using two backgrounds behind each other and each is moved by a set amount to simulate motion)

```
//intializations for moving background
var bgDrawY1=0;
var bgDrawY2=-550;
var bgSrc1=0;
var bgSrc2=0;

//actual movement of background to make the motion look smoother
function moveBg()
{
  bgDrawY1+=5;
  bgDrawY2+=5;
  if(bgDrawY1 >=550 )
  {
    bgDrawY1=-550;
  }
  if(bgDrawY2 >=550 )
  {
```

```
function drawBg()
{
  ctxBg.clearRect(0,0,gamewidth,gameheight);
  var srcX=0;
            var srcY=0;
            var drawX=0;
            var drawY=0;
            ctxBg.drawImage(imgSprite,bgSrc1,0,gamewidth,gameheight,0,bgDrawY1,gamewidth,
gameheight);

ctxBg.drawImage(imgSprite,bgSrc2,0,gamewidth,gameheight,0,bgDrawY2,gamewidth,gamehe
ight);

}
```

Main Character object :

```
function Plane()
{
            this.srcX=1003;
            this.srcY=0;
            this.planeWidth=35;
            this.planeHeight=75;
            this.drawX=(gamewidth-this.planeWidth)/2;
            this.drawY=gameheight-this.planeHeight;
            this.speed=2;
  this.hp=3;
  this.score=0;
  //where the bullets will spawn
  this.noseX =this.drawX+17.5 ;
  this.noseY = this.drawY;
            this.isUpKey=false;
            this.isRightKey=false;
            this.isLeftKey=false;
            this.isDownKey=false;
  this.isSpacebar=false;
  this.isShooting=false;
  this.rockets= [];
  this.currentRocket=0;
  this.mainExplosion=new Explosion();
  for(var i=0;i<8;i++)
```

Enemy object: (with two enemy types for every level and different HP values)

```
function Enemy(hp)
{
 //enemies spawning in level 1
 if(isLevelOne){
 this.srcX=1000;
 this.srcY=77;
 this.enemyWidth=125;
 this.enemyHeight=133;
 this.speed=3;}
 //enemies spawning in level2
 else if(isLevelTwo)
 {
 this.srcX=1130;
 this.srcY=75;
 this.enemyWidth=150;
 this.enemyHeight=185;
 this.speed=2;
 }

 this.drawX=Math.floor(Math.random()*gamewidth-this.enemywidth);
 this.drawY=-1*Math.floor(Math.random()*gameheight);
 this.rewardPoints = 5;
 this.hp=hp;
 this.maxHp=hp;
}
```

Mouse motion function: (controls direction of motion of main plane,change it's sprite position to simulate turning and keep the object within the canvas.

```
function mouseMoved(e)
{
 if(e.pageX-canvasBg.offsetLeft<=0)
    mainPlane.drawX=0;
  else if(e.pageX-canvasBg.offsetLeft>=465)
    mainPlane.drawX=465;
  else{if(mainPlane.drawX<e.pageX-canvasBg.offsetLeft)
         mainPlane.srcX=1040;
       else if(mainPlane.drawX>e.pageX-canvasBg.offsetLeft)
         mainPlane.srcX=1080;
        else
         mainPlane.srcX=1000;
    mainPlane.drawX=e.pageX-canvasBg.offsetLeft;
      }
 if(e.pageY-canvasBg.offsetTop <=0)
  mainPlane.drawY=0;
 else if(e.pageY-canvasBg.offsetTop>=475)
  mainPlane.drawY=475;
else
mainPlane.drawY=e.pageY-canvasBg.offsetTop;
}
```

Rocket Collision function:(to check outgouing rockets against nearby enemies)

```
Rocket.prototype.checkHitEnemy=function()
{
  for(var i=0;i<enemies.length;i++)
  { if(this.drawX>=enemies[i].drawX && this.drawX<enemies[i].drawX+enemies[i].enemyWidth
    && this.drawY>=enemies[i].drawY &&
this.drawY<=enemies[i].drawY+enemies[i].enemyHeight )
    {  enemies[i].hp--;
      this.recycle();
      if(enemies[i].hp==0)
      {
      this.explosion.drawX=enemies[i].drawX;
      this.explosion.drawY=enemies[i].drawY+30;
      this.explosion.hasHit=true;
      mainPlane.updateScore(enemies[i].rewardPoints,enemies[i]);
      enemies[i].destructor();
     }
      }
      else if(this.drawX+this.width>=enemies[i].drawX &&
this.drawX+this.width<enemies[i].drawX+enemies[i].enemyWidth
    && this.drawY+this.height>=enemies[i].drawY &&
this.drawY+this.height<=enemies[i].drawY+enemies[i].enemyHeight )
     {
      enemies[i].hp--;
      this.recycle();
      if(enemies[i].hp==0)
      {
      this.explosion.drawX=enemies[i].drawX;
      this.explosion.drawY=enemies[i].drawY+30;
      this.explosion.hasHit=true;
      mainPlane.updateScore(enemies[i].rewardPoints,enemies[i]);
      enemies[i].destructor();
      }
    }

    }
```