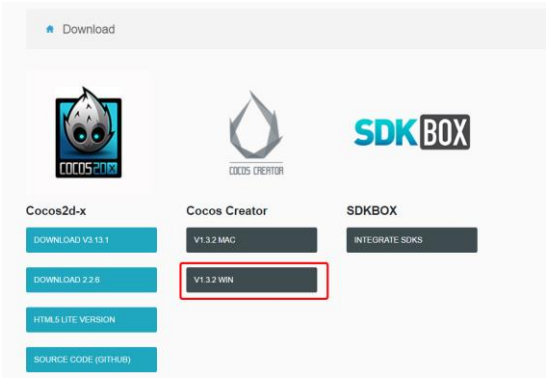
Pasos para empezar a generar un nuevo juego en Cocos Creator

Instalando Cocos Creator

Lo primero que hay que hacer es bajar la versión más reciente de Cocos Creator, en mi caso fue la V1.3.2 para Windows.

Link: http://cocos2d-x.org/filedown/CocosCreator_v1.3.2_win

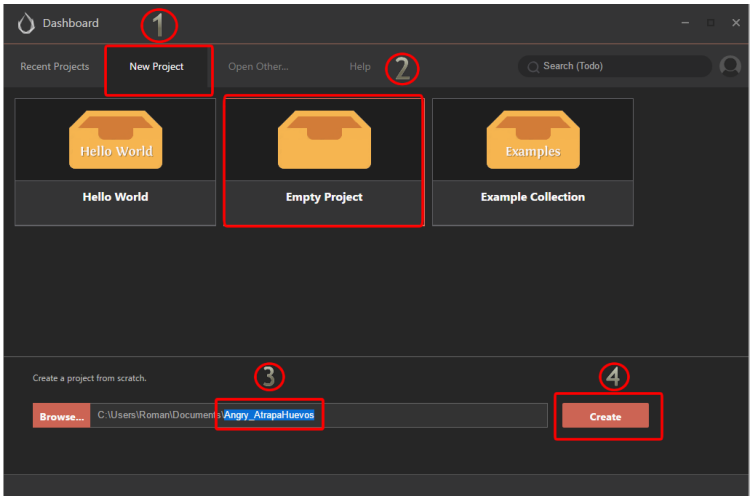


Al instalar Cocos, nos pregunta que, si queremos instalar Visual Studio y python, en mi caso seleccione si ya que yo uso el compilador C++, para ustedes depende de que compilador quieran usar.

Generando proyecto

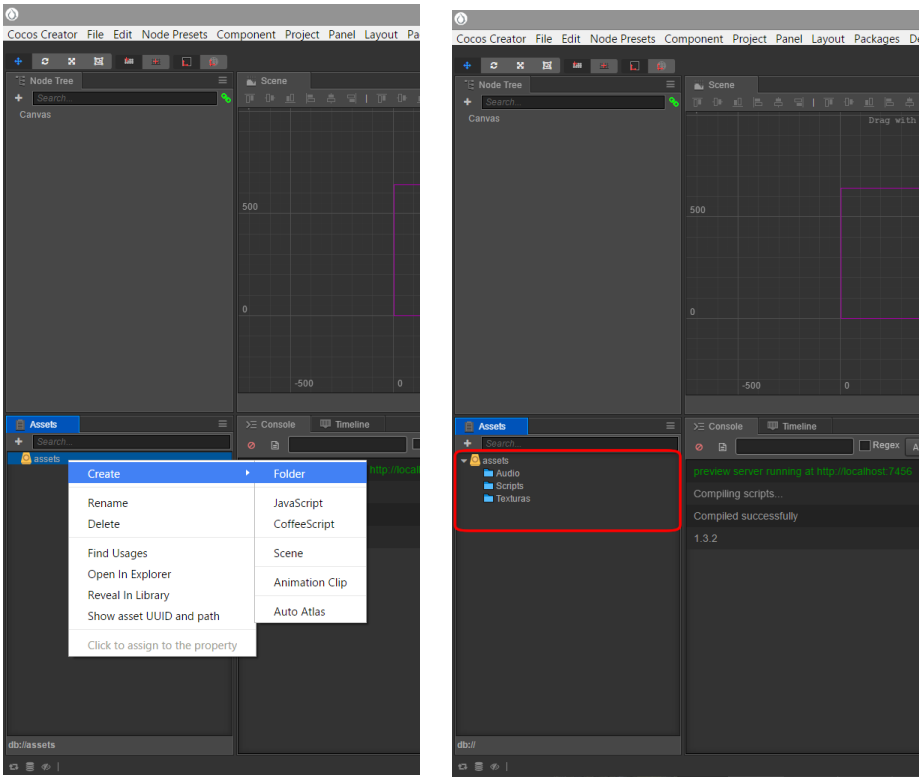
Abrimos Cocos Creator, en su ventana de proyectos (**Dashboard**),

1. Seleccione **New ProProject**.
2. Seleccione **Empty Project**.
3. Asigne el nombre del proyecto.
4. De clic al botón **Create**.

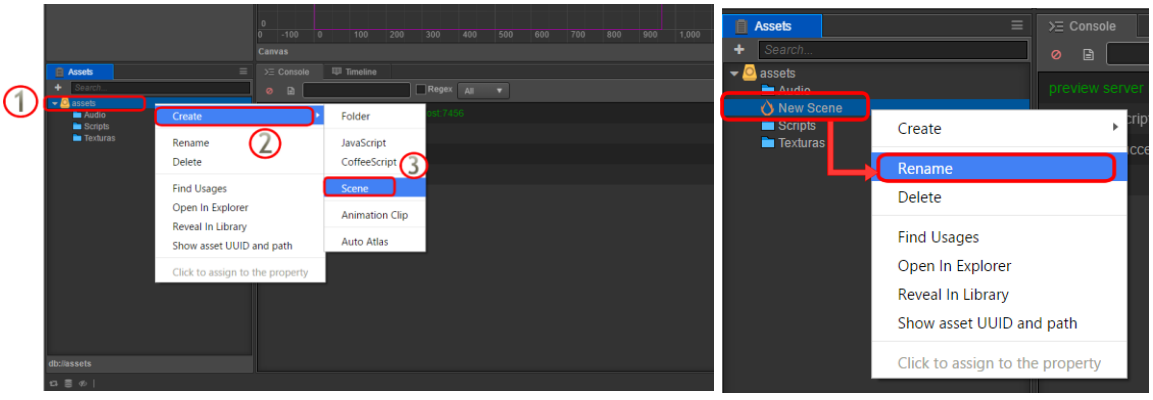


En el entorno d edición de su proyecto en la parte inferior izquierda vamos a agregar los directorios del proyecto (texturas, scripts, audio), para ello, con el mouse, de click derecho en **assets** y en el

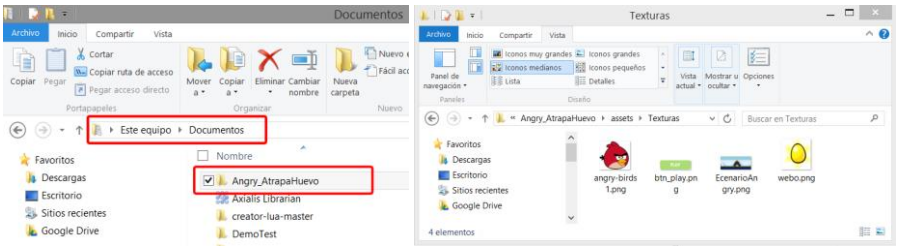
menú emergente seleccione la opción **Create**, seguido a ello seleccione **Folder**, al directorio que se genera, renómbrelo a uno de los indicados y así sucesivamente hasta terminar de generar los directorios necesarios.



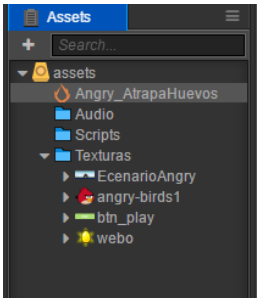
Ahora vamos a generar una nueva escena para nuestro proyecto, para ello, seleccionamos assets y sobre el damos click con el botón derecho del mouse y en el menú emergente no posicionamos sobre **Create** y finalmente damos click en **Scene**, y renombramos la nueva escena (Ver imagen).



Lo siguiente es agregar nuestras imágenes al proyecto, para ello nos vamos a nuestros documentos, ahí encontraremos nuestro proyecto ya que por defecto Cocos los genera ahí. No vamos a la ruta assets\Texturas y en esta colocamos nuestras imágenes de proyecto.



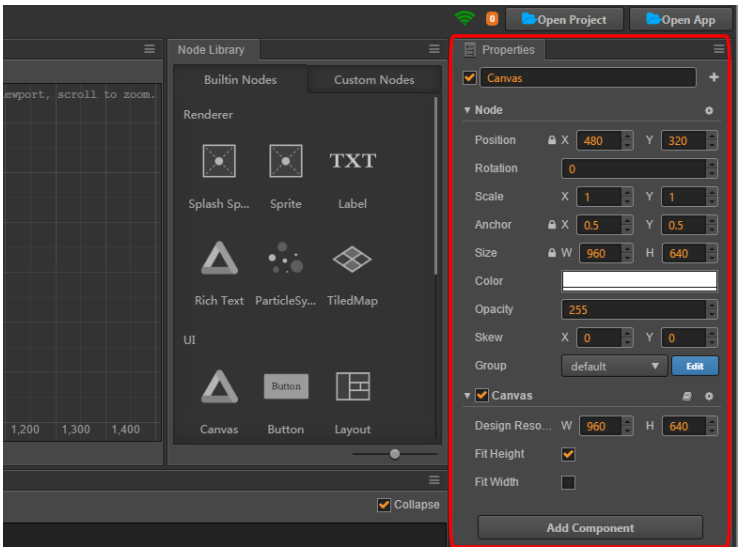
Automaticamente se cargaran en Cocos



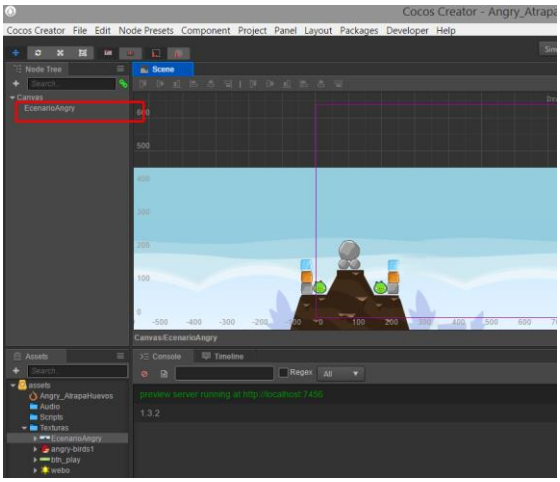
Ahora que ya tenemos todo listo, para empezar, damos doble clic e nuestra escena, para mi caso (Angry_AtrapaHuevos).

Para entender canvas (lona)

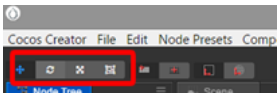
Después de abrir la escena, todos los nodos (y sus relaciones jerárquicas) de la escena actual se muestran en la del árbol de nodos. La escena recién creada tiene sólo un nodo llamado Canvas. El Cual puede ser llamado el nodo de lona o de la prestación nodo raíz. Haga clic en la lona (Canvas), sus propiedades se mostrarán en el Propiedades del panel Izquierdo de su proyecto.

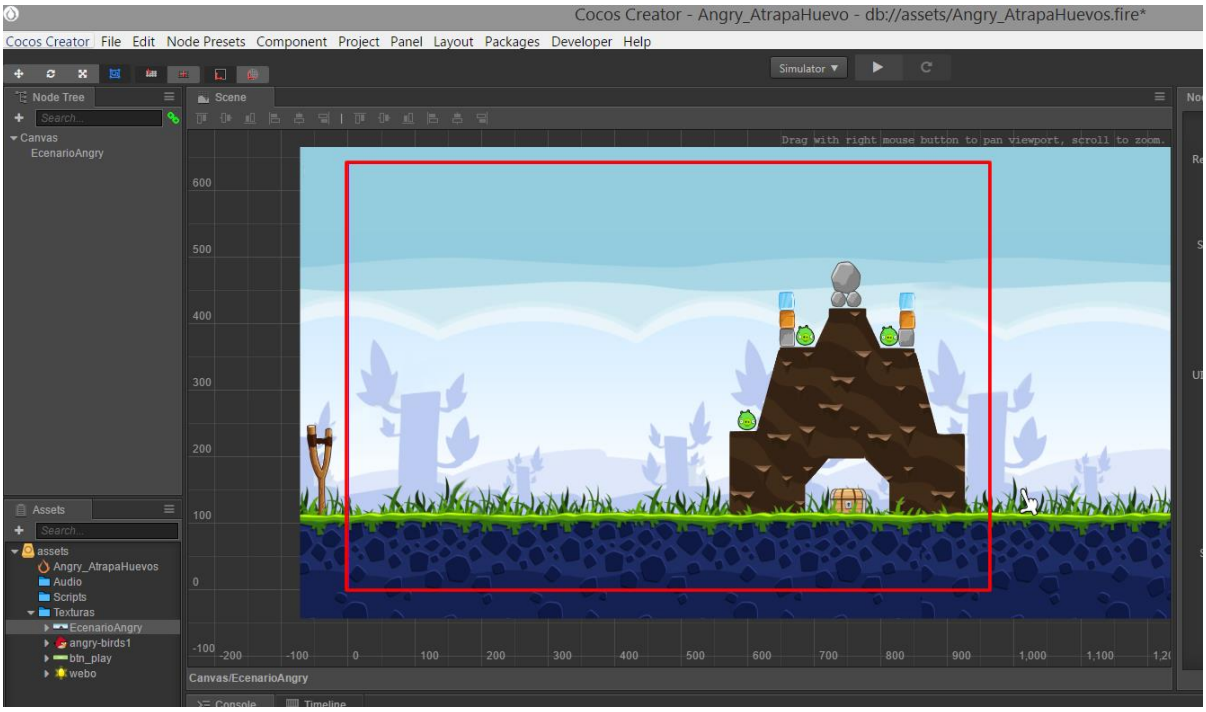


Como primer paso vamos a agregar a Canvas un fondo (Escena), para ello arrastramos la imagen que será nuestro fondo (Para mi caso es EcenarioAngry.png).



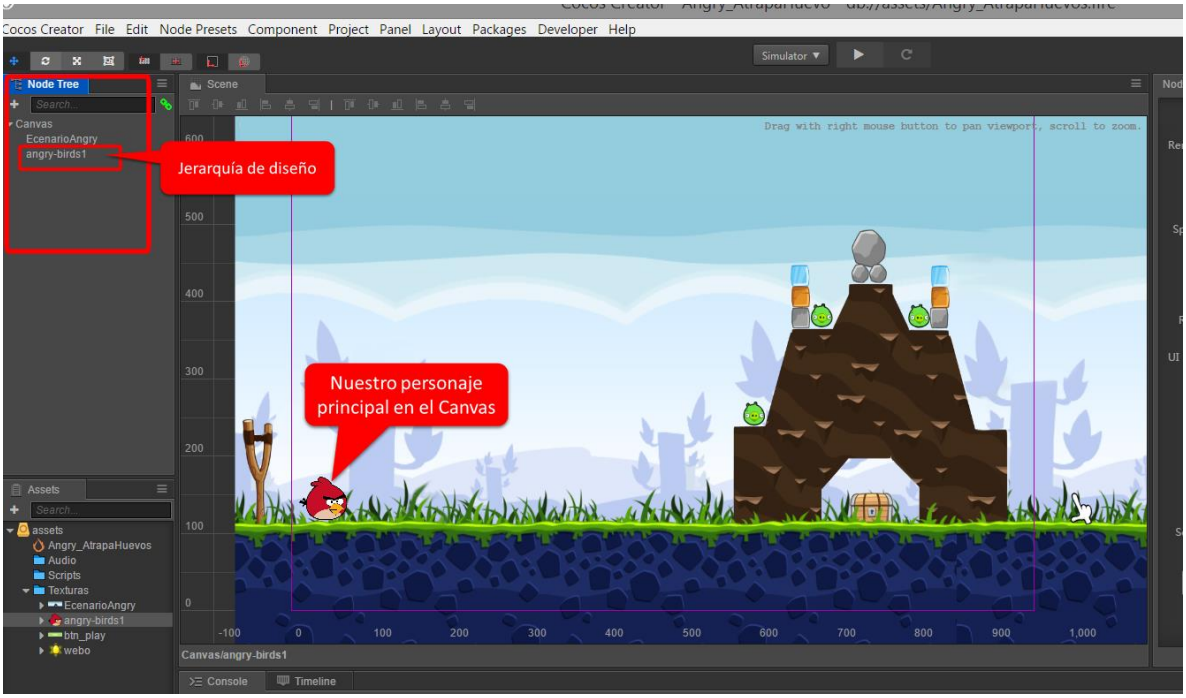
Como vemos, ya tenemos un fondo para nuestro juego, lo siguiente es centrar nuestra imagen en nuestro entorno Canvas, para ello la seleccionamos y en el panel superior izquierdo existen funciones para mover, rotar o cambiar de tamaño.



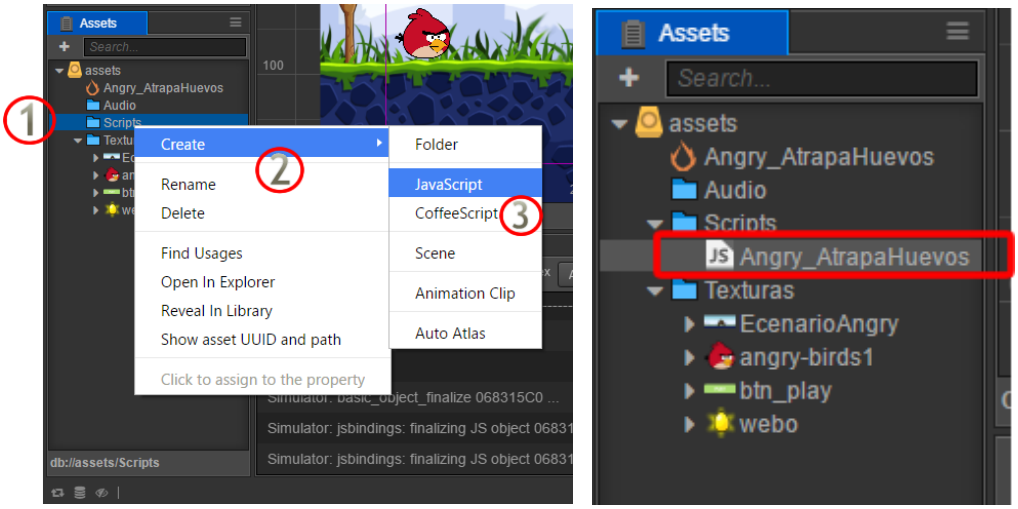


Lo siguiente es agregar nuestro personaje principal (Para mi caso es angry-birt1.png), para ello lo arrastramos al Canvas debajo de nuestro fondo (Recordemos que todo lo que agreguemos es

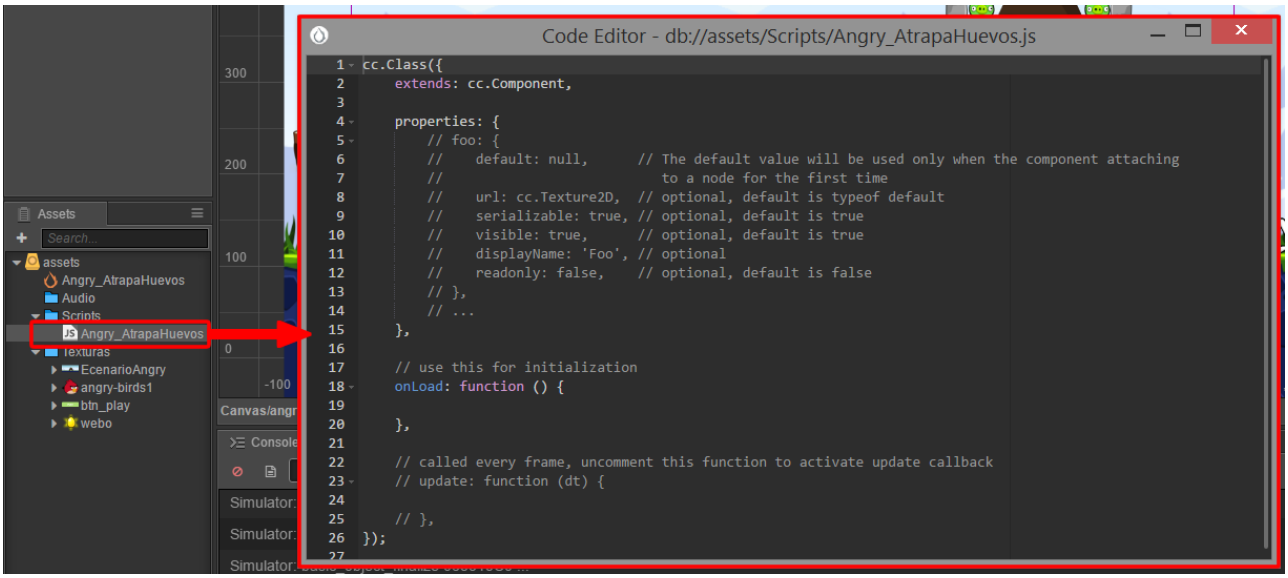
jerárquico por lo que si la imagen principal queda arriba de nuestro fondo quedara por debajo y no se mostrara).



Ahora viene lo bueno, que es agregar script a nuestro personaje principal, para ello nos posicionamos en la carpeta **Script** de nuestro proyecto y sobre ella damos un clic derecho con el mouse y buscamos la opción JavaScript (Menu>Create>JavaScrip), una vez se genere, le cambiamos el nombre.

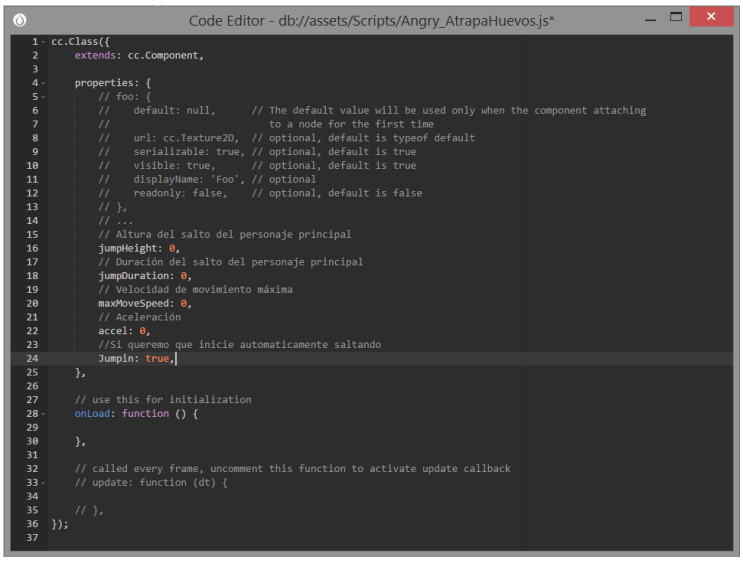


Damos doble clic al script generado, observemos que ya hay código, este lo genera Cocos por defecto y estos códigos son la estructura necesaria para la escritura de un componente.

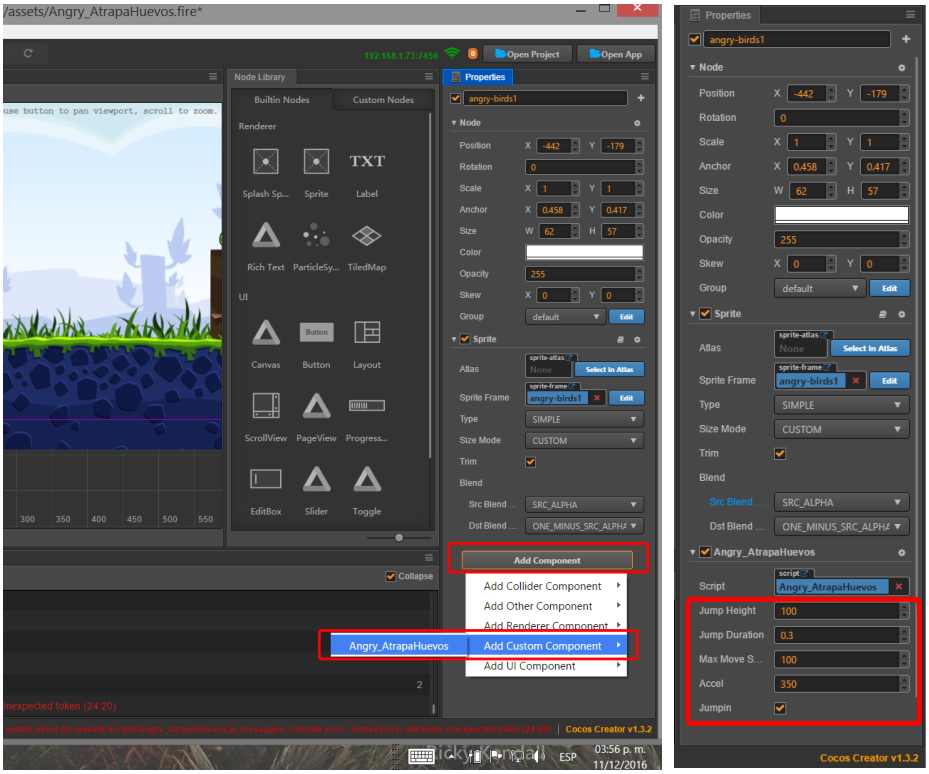


En la sección propiedades agregamos el siguiente código y presionamos la combinación de teclas **CTRL + S** para guardar los cambios.

```
// Player.js
        //...
        properties: {
                // main character's jump height
                jumpHeight: 0,
                // main character's jump duration
                jumpDuration: 0,
                // maximal movement speed
                maxMoveSpeed: 0,
                // acceleration
                accel: 0,
                //Si queremo que inicie automaticamente saltando
                Jumpin: true,
        },
        //..
```



Lo siguiente es ir al **Canvas** y dar doble clic a nuestro personaje principal (En mi caso angry-birds1.png), en la barra de menú derecho aparecerán las propiedades de nuestro personaje principal, en estas propiedades damos clic al botón **Add Component** y en el menú emergente seleccionamos **Add Custom Component** y seguido de ello seleccionamos nuestro componente creado en el script previo (Para mi caso **Angry-AtrapaHuevos**.js). En las propiedades fijamos los campos como lo indica la imagen (Jump Height = 100, Jump Duration = 0.3, Max Move Speed=100 y Accel = 350).



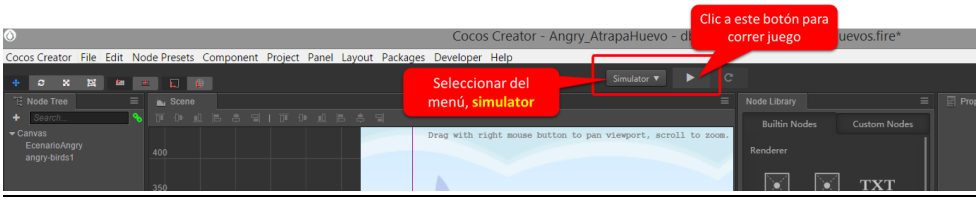
Regresamos a nuestro código y agregamos en la sección **onLoad** el siguiente código y presionamos la combinación de teclas **CTRL + S** para guardar los cambios.

```
        // use this for initialization
        onLoad: function () {
            this.Jumpin = true;
            /*Iniciando código para hacer que nuestro angry brinque sin parar*/

            // inicializando acción de salto (jump action)
            this.jumpAction = this.setJumpAction();
            this.node.runAction(this.jumpAction);
            /*Finalizando código para hacer que nuestro angry brinque sin parar*/

        },
    setJumpAction: function () {
            // Salto arriva (jump up))
            var jumpUp = cc.moveBy(this.jumpDuration, cc.p(0, this.jumpHeight)).easing(cc.easeCubicActionOut());
            // Salto abajo (jump down)
            var jumpDown = cc.moveBy(this.jumpDuration, cc.p(0, -this.jumpHeight)).easing(cc.easeCubicActionIn());
            // Repetir(repeat)
            if(this.Jumpin == true)
            {return cc.repeatForever(cc.sequence(jumpUp, jumpDown));}
            else {return cc.sequence(jumpUp, jumpDown);}
        },
```

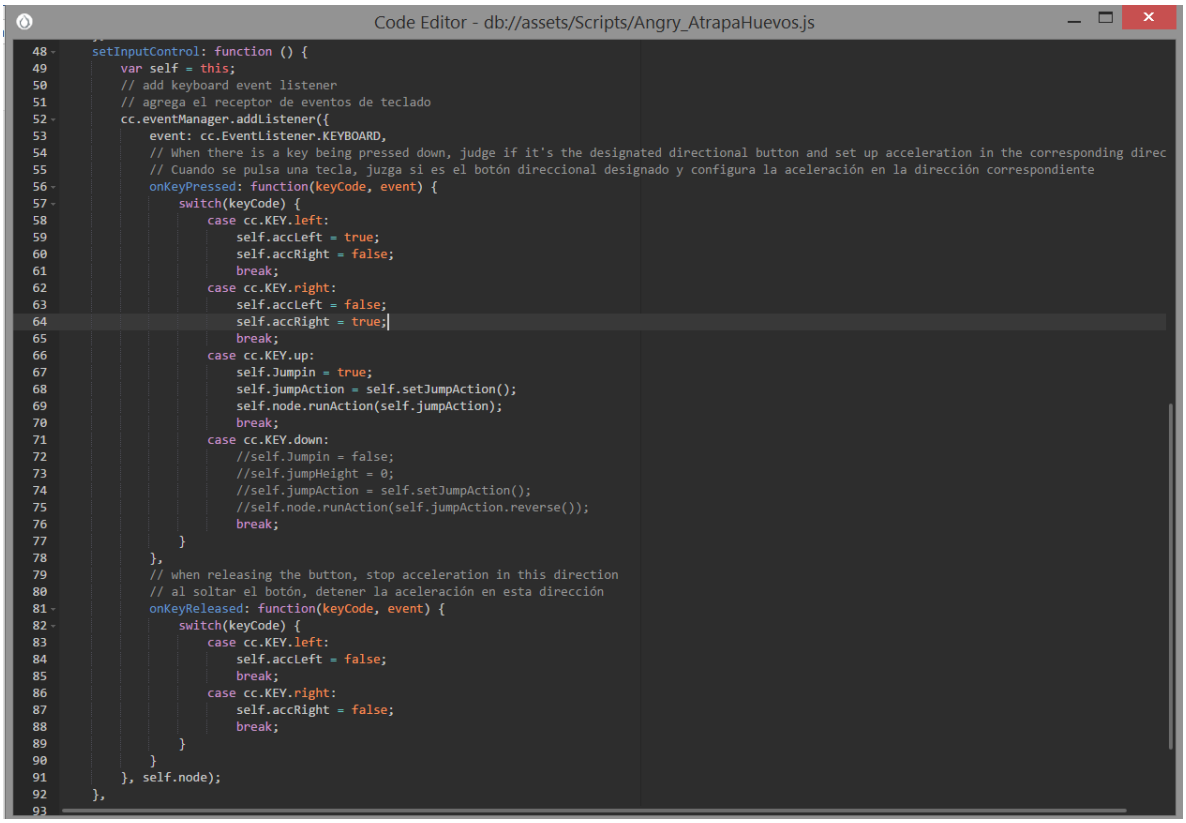
El código que agregamos, es para hacer brincar a nuestro personaje principal, ahora corramos nuestro juego y el personaje principal deberá estar brincando.



Lo siguiente es hacer que nuestro personaje se mueva ya que dejarlo en un solo lugar no sería lógico, para ello regresamos a nuestro código para agregar eventos de movimiento **Izquierda-Derecha** de teclado y acciones en estos eventos.

Agregamos el siguiente código y presionamos la combinación de teclas **CTRL + S** para guardar los cambios.

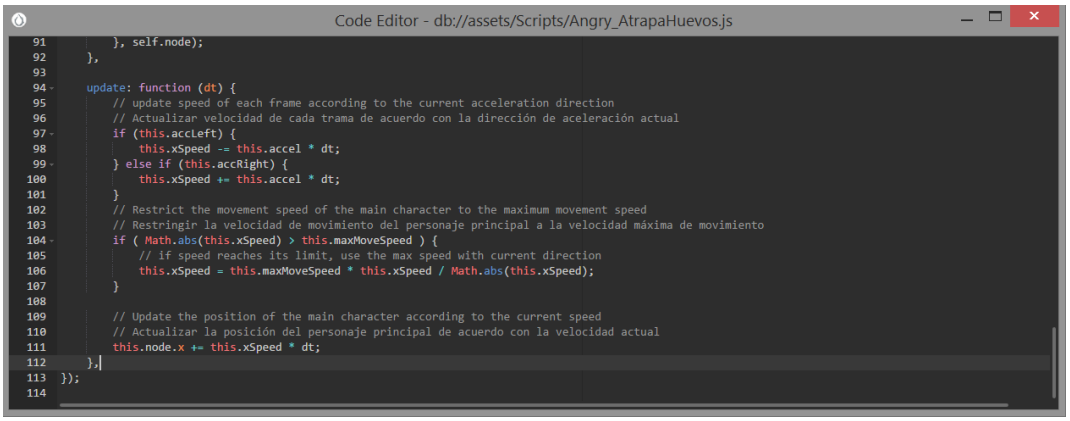
```
setInputControl: function () {
        var self = this;
        // add keyboard event listener
        // agrega el receptor de eventos de teclado
        cc.eventManager.addListener({
                event: cc.EventListener.KEYBOARD,
                // When there is a key being pressed down, judge if it's the designated directional
                //button and set up acceleration in the corresponding direction
                // Cuando se pulsa una tecla, juzga si es el botón direccional designado y configura
                //la aceleración en la dirección correspondiente
            onKeyPressed: function(keyCode, event) {
                switch(keyCode) {
                  case cc.KEY.left:
                     self.accLeft = true;
                     self.accRight = false;
                     break;
                  case cc.KEY.right:
                     self.accLeft = false;
                     self.accRight = true;
                     break;
                  case cc.KEY.up:
                     self.Jumpin = true;
                     self.jumpAction = self.setJumpAction();
                     self.node.runAction(self.jumpAction);
                     break;
                  case cc.KEY.down:
                     //self.Jumpin = false;
                     //self.jumpHeight = 0;
                     //self.jumpAction = self.setJumpAction();
                     //self.node.runAction(self.jumpAction.reverse());
                     break;
                }
            },
            // when releasing the button, stop acceleration in this direction
            // al soltar el botón, detener la aceleración en esta dirección
            onKeyReleased: function(keyCode, event) {
               switch(keyCode) {
                  case cc.KEY.left:
                     self.accLeft = false;
                     break;
                  case cc.KEY.right:
                     self.accRight = false;
                     break;
               }
            }
         }, self.node);
      },
```



Después agregaremos código para actualizar los eventos que se generen al cumplirse el evento de teclado, agregamos el siguiente código y presionamos la combinación de teclas **CTRL + S** para guardar los cambios.

```
update: function (dt) {
    // update speed of each frame according to the current acceleration direction
    // Actualizar velocidad de cada trama de acuerdo con la dirección de aceleración actual
    if (this.accLeft) {
        this.xSpeed -= this.accel * dt;
    } else if (this.accRight) {
        this.xSpeed += this.accel * dt;
    }
    // Restrict the movement speed of the main character to the maximum movement speed
    // Restringir la velocidad de movimiento del personaje principal a la velocidad máxima de movimiento
    if ( Math.abs(this.xSpeed) > this.maxMoveSpeed ) {
        // if speed reaches its limit, use the max speed with current direction
        this.xSpeed = this.maxMoveSpeed * this.xSpeed / Math.abs(this.xSpeed);
    }

    // Update the position of the main character according to the current speed
    // Actualizar la posición del personaje principal de acuerdo con la velocidad actual
    this.node.x += this.xSpeed * dt;
},
```

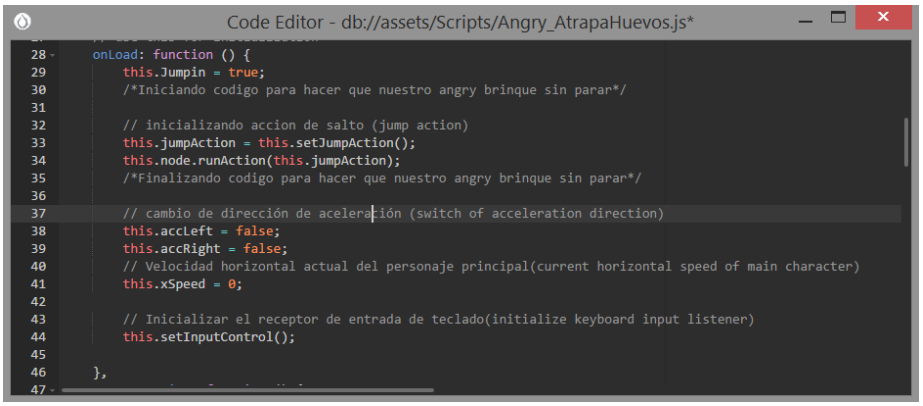


Lo siguiente es inicializar estas funciones y para ello nos vamos a la seccione de código **onLoad** y agregamos el siguiente código y presionamos la combinación de teclas **CTRL + S** para guardar los cambios.

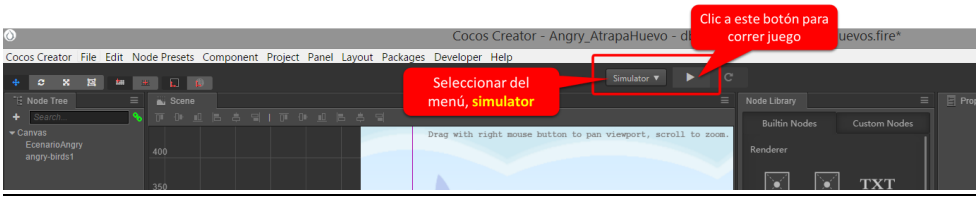
```
onLoad: function () {
    //.....
    // cambio de dirección de aceleración (switch of acceleration direction)
    this.accLeft = false;
    this.accRight = false;
    // Velocidad horizontal actual del personaje principal (current horizontal speed of main character)
    this.xSpeed = 0;

    // Inicializar el receptor de entrada de teclado (initialize keyboard input listener)
    this.setInputControl();

},
```



Corramos nuevamente nuestro juego y el personaje principal deberá estar brincando y también ahora ya podemos moverlo de izquierda a derecha con las teclas del teclado.



Referencias

Inc., C. T. (2016). *http://cocos2d-x.org/*. Obtenido de http://cocos2d-x.org/: http://cocos2d-x.org/docs/editors_and_tools/creator-chapters/getting-started/quick-start/index.html

Inc., C. T. (2016). *http://cocos2d-x.org/*. Obtenido de http://cocos2d-x.org/: http://cocos2d-x.org/docs/editors_and_tools/creator-chapters/asset-workflow/prefab/index.html