Race Cloud

Manual del programador

24/05/2016

Francisco Javier Gómez Vázquez

José Eduardo Loyola Donjuan

PROPUESTA DE PROYECTO

Semestre 2015-2016/II

Contenido

[No de proyecto 3](#_Toc451870154)

[Nombre del proyecto 3](#_Toc451870155)

[Integrantes del equipo 3](#_Toc451870156)

[Objetivo del proyecto 3](#_Toc451870157)

[Descripción del proyecto 3](#_Toc451870158)

[Descripción e imágenes de cada nivel 3](#_Toc451870159)

[● Nivel Principiante 3](#_Toc451870160)

[● Nivel Intermedio 4](#_Toc451870161)

[● Nivel Avanzado 5](#_Toc451870162)

[Diagrama de clases UML 5](#_Toc451870163)

[Características y comportamiento de cada clase 6](#_Toc451870164)

[Herencia y polimorfismo 8](#_Toc451870165)

[Cronograma de actividades (plan de trabajo) 8](#_Toc451870166)

[Bitácora de actividades (historial) 8](#_Toc451870167)

# 

# No de proyecto

163

# Nombre del proyecto

RACE COULD.

# Integrantes del equipo

Francisco Javier Gómez Vázquez 240669

José Eduardo Loyola Donjuan 239915

# Objetivo del proyecto

El objetivo del juego, es completar los tres niveles de los cuales se compone el mismo, asi también como recolectar la mayor cantidad de puntos posibles agarrando objetos asignados para eso y evitando colisionar con otros que le restaran vida, si su vida llega a cero, el juego termina .

# Descripción del proyecto

En el juego, tomamos el papel de un personaje, el cual esta volando ayudado de una nube, nuestra misión es esquivar una serie de obstáculos junto con disparos de un enemigo y agarrar la mayor cantidad de capsulas posibles, el juego se gana cuando el escenario se termine, el verdadero reto es juntar la mayor cantidad de puntos, por el contrario, se pierde cuando nuestra vida llega a cero, que puede ser colisionando con los obstáculos o por una bala enemiga.

# Descripción e imágenes de cada nivel

El juego deberá tener por lo menos tres niveles de dificultad, pero se pueden tener más si así lo requiere el proyecto. Para cada uno de los niveles, además de las imágenes del escenario, se debe describir claramente las dificultades que el jugador tiene que superar para pasar al siguiente nivel. Incluir como parte del escenario las estadísticas de cada nivel, por ejemplo, puntos, vidas, tiempo, nivel, etc, según sea requerido en el proyecto.

* Nivel Principiante

Nivel 1: Se iniciar con 4 vidas y 0 puntos, el único objetivo para avanzar al siguiente nivel es junta 500 puntos, los cuales se pueden obtener de dos formar, conforme avanzamos acomulamos puntos lentamente o si agarramos capsulas, las cuales cada una nos dara 50 puntos. Tambien se deberá evitar colisionar con los obstáculos o los disparos del enemigo.

* Nivel Intermedio

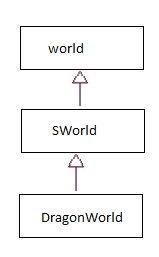
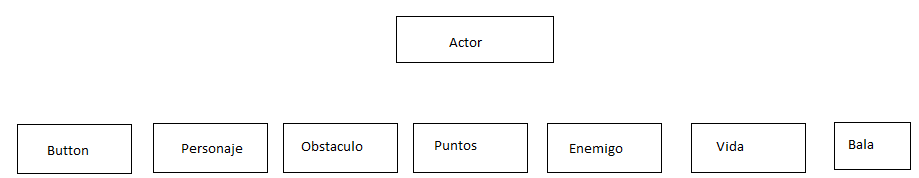
Nivel 2: El fondo y nuestro enemigo cambian, la velocidad del juego aumenta lo cual hace mas difícil recolectar capsulas y esquivar los obstaculos y las balas. Se avanzara al siguiente nivel juntando 1000 puntos.

* Nivel Avanzado

Nivel 3: En este punto, de nueva cuenta la velocidad del juego aumenta, al igual que el escenario y el enemigo, se vuelve aun mas difícil esquivar los obstáculos y las balas, asi como también el agarrar las capsulas, la tarea en este nivel es juntar la mayor cantidad de puntos posibles, antes de ganar el juego, que sucede cuando el escenario se acaba.

# Diagrama de clases UML

Elaborar un diagrama de clases colocando únicamente el nombre de las clases y sus relaciones de uso y herencia. Los nombres de los métodos y las características serán descritos en la siguiente sección.



# Características y comportamiento de cada clase

|  |  |
| --- | --- |
| Nombre de la clase: | Sworld |
| Características: | Desplaza una imagen de fondo |
| Utiliza un actor main para desplazarse según la acción de este |
|  |
| Comportamiento: | Permite manejar el scroll dentro de juego creando un efecto de |
| movimiento sobre el mapa en donde se desplaza el personaje principal |
| de nuestro juego |

|  |  |
| --- | --- |
| Nombre de la clase: | DragonWorld |
| Características: | Contiene los escenarios y distintas pantallas |
| Mantiene funcionando el juego |
| Selecciona el personaje en turno |
| Comportamiento: | Crea un menú inicio, dependiendo de la selección del usuario llama los |
| métodos correspondientes a dicha selección, permite seleccionar al |
| personaje con que arrancara el juego y lo inserta cuando inicia |
| el juego |

|  |  |
| --- | --- |
| Nombre de la clase: | Personaje |
| Características: | Le da imagen a un personaje |
| Mueve el personaje dentro del escenario |
| Detecta si está en contacto con otro objeto |
| Comportamiento: | Recibe el personaje que selecciono el jugador para darle una imagen |
| Si inicia el juego comienza a desplazarlo en la pantalla y si es golpeado |
| o recoge capsulas manda la reacción necesaria |

|  |  |
| --- | --- |
| Nombre de la clase: | Button |
| Características: | Crea una imagen para el boton |
|  |
|  |
| Comportamiento: | Envía la imagen que posee para que con esto determinemos |
| que botón selecciono el jugador |
|  |

|  |  |
| --- | --- |
| Nombre de la clase: | Enemigo1 |
| Características: | Crea una imagen para el enemigo |
| Se mueve a la par del personaje principal |
| Dispara a la posición del personje |
| Comportamiento: | Recibe que enemigo que atacara según el nivel, aparace a la par del |
| personaje y comienza a lanzarle proyectiles que de impactarle |
| causaran una baja en su nivel de vida |

|  |  |
| --- | --- |
| Nombre de la clase: | Obstaculo |
| Características: | Le da una imagen al objeto |
| Mueve el objeto |
|  |
| Comportamiento: | Crea un obstáculo en el mundo que si es tocado por el personaje este |
| recibirá daño al contacto . |
|  |

|  |  |
| --- | --- |
| Nombre de la clase: | Puntos |
| Características: | Aparece en el mundo |
| Mueve el objeto |
|  |
| Comportamiento: | Se crea una capsula en una posición aleatoria y avanza en contra |
| del personaje |

|  |  |
| --- | --- |
| Nombre de la clase: | Vida |
| Características: | Aparece en el mundo |
| Se mueve con el personaje |
| Comportamiento: | Crea la barra de vida |

|  |  |
| --- | --- |
| Nombre de la clase: | Bala |
| Características: | Mueve el objeto |
| Comportamiento: | Crea un proyectil en el mundo que si es tocado por el personaje este |
| recibirá daño al contacto . |
|  |

# Herencia y polimorfismo

Utilizamos herencia en la clase DragonWorld quien es hija de SWorld, de esta manera los métodos utilizados en la clase padre so heredados a la hija y de esta manera logramos hacer que el escenario e moviera junto con el personaje.

# Cronograma de actividades (plan de trabajo)

|  |  |  |
| --- | --- | --- |
| Fecha de Inicio | Fecha de Término | Actividad por realizar |
| 23/05/2016 | 23/05/2016 | Manual de Usuario |
| 20/05/2016 | 23/05/2016 | Manual de programador |
| 14/02/206 | 20/05/2016 | Codigo |
| 21/05/2016 | 22/05/2016 | Video |

# Bitácora de actividades (historial)

Esta parte será llenada durante la elaboración del proyecto. Aquí se deben describir cada una de las actividades realizadas desde la propuesta hasta la entrega del proyecto.

|  |  |
| --- | --- |
| Fecha (dd/mm/aa) | Descripción de la actividad realizada |
| 14/02/2016 | Iniciamos con la idea del proyecto creando al personaje principal con movimientos básicos |
| 11/03/2016 | Implementamos el scroll en el mundo aunque el personaje no parecía moverse solo. Creamos puntos en el escenario |
| 28/03/2016 | Logramos que el personaje se moviera automáticamente aparece el primer enemigo y elimina los puntos. |
| 15/04/2016 | El enemigo dispara a la posición del personaje principal, ahora puedes seleccionar entre 3 posibles personajes |
| 31/04/2016 | Se crea el personajes se refactoriza el código. Se empiezan a crear niveles |
| 20/05/2016 | Se crean los niveles y se acopla todo. |