ESCUELA COLOMBIANA DE INGENIERÍA
PROGRAMACIÓN ORIENTADA A OBJETOS
POOBird 2017-02

Propuesta: Nicolás García Rey – Carlos Gómez

Flappy Bird

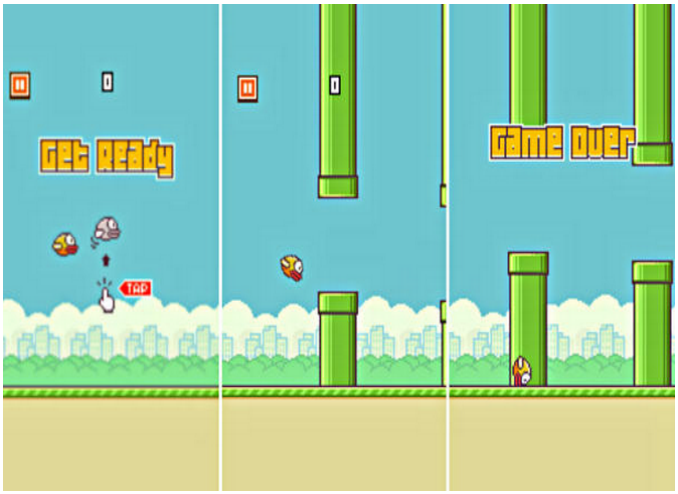
Flappy Bird es considerado uno de los juegos para dispositivos móviles más adictivos.

El jugador controla un pájaro que intenta volar entre filas de tuberías verdes sin estrellarse, sí en uno de sus intentos choca, se acaba el juego.

La escena se va desplazando lateralmente de derecha a izquierda y el único movimiento del pájaro es volar hacia arriba cuando el jugador toca la pantalla.

El objetivo del juego es atravesar el mayor número de tuberías sin chocar.

http://flappybird.io/



POOBird

Las novedades de POOBird son las siguientes:

- Ofrece dos modos de juego: solitario o en pareja.
- Incluye un pájaro automático con diferentes perfiles.
- Se puede seleccionar la velocidad del juego
- Los pájaros no mueren con los golpes, sino que van perdiendo su fortaleza.
- Existen diferentes clases de tubos cada uno con un comportamiento específico.
- Pueden aparecer monedas mágicas que dan a los pájaros fortaleza o inmunidad contra los diferentes enemigos.
- Pueden aparecer dragones inmunes a los tubos volando en cualquier dirección.

El juego termina cuando los pájaros de juego pierden su vida.

POOBirds

De cada **POOBird** se tiene la siguiente información:

1. Efectividad: número de tubos atravesados.
2. Fortaleza: el nivel de fortaleza actual (0 a 100%). Los pájaros mueren cuando este nivel es del 0%.
3. Poderes: si tiene actualmente algún poder.

El mejor **POOBird** es el que atraviesa más tubos manteniendo alto su nivel de fortaleza.

Los **POOBird** automáticos pueden asumir diferentes perfiles, entre ellos:

1. Tímido: busca terminar el juego lo antes posible estrellándose con los tubos.
2. Equilibrado: busca mantener siempre su nivel de altitud.
3. Astuto: si prioridad es evitar chocar con los tubos
4. Mimo: imita el movimiento de su pareja

TUBOS

Los comportamientos de los tubos son los siguientes:

1. **Verdes:** los originales
2. **Rojos:** se cierran y abren después de determinado tiempo. La apertura siembre es en el mismo punto
3. **Azules:** se mueven de arriba a abajo junto con su abertura.
4. **Rojos:** se mueven de abajo a arriba junto con su abertura.

El daño causado depende del tipo de tubo.

DRAGONES

Los dragones van a la misma velocidad de las aves.

1. **Verdes:** van en dirección contraria de los pájaros
2. **Rojos:** van la dirección de los pájaros, escupen bolas de fuego que avanzan a velocidades aleatorias.

El encuentro con un dragón es mortal.

MONEDAS

Las monedas aparecen en el suelo. Un pájaro no pueden tener más de una moneda activa.

Los poderes que dan las monedas son los siguientes:

1. **Oro:** se vuelven inmunes a un número determinado de tubos, dependiendo del valor de la moneda.
2. **Plata:** se vuelve inmune a los mini-dragones. Sólo se puede usar una vez.
3. **Algodón:** recupera el 50% de la fortaleza perdida.

REQUISITOS FUNCIONALES

La aplicación debe:

- ➔ Permitir generar aleatoriamente un escenario de juego
- ➔ Permitir seleccionar los diferentes elementos para el juego
- ➔ Permitir importar un escenario de juego
- ➔ Permitir abrir y salvar el estado de un juego

- ➔ Permitir seleccionar el tipo de juego
- ➔ Permite seleccionar el tipo de oponente, bien sea persona o un tipo específico de máquina
- ➔ Permitir a los jugadores elegir el color deseado

- ➔ Permitir realizar los movimientos
- ➔ Ilustrar permanentemente el estado del de juego y el estado de cada uno de los jugadores

- ➔ Permitir que el jugador termine el juego en cualquier momento
- ➔ Terminar el juego y comunicar su causa

REQUISITOS DE DISEÑO

De extensión

➔ Permitir generar nuevas versiones de la aplicación que incluyan otros tipos de tubos, monedas, dragones y otros perfiles de jugador computador (por ejemplo, en la competencia, el nuevo perfil con la estrategia)

De visualización

➔ El tablero debe tener una representación gráfica adecuada que permita conocer el estado del juego.

De manejo de excepciones

➔ Deben definir mínimo una nueva clase excepción para manejar las excepciones propias.

➔ Cuando ocurra una excepción no esperada o una propia grave se debe escribir esta información en el log de errores para los programadores y terminar la ejecución del mismo.

REQUISITOS DE ENTREGA

Revisión inicial		CAPA DE PRESENTACIÓN Boceto de la interfaz gráfica CAPA DE APLICACIÖN Diagrama de clases	A par S12:Jueves 3 de noviembre
Versión uno **Presentación**	Generar escenario Juego de un jugador Tubo verde Volar	**CAPA PRESENTACIÓN** **Boceto del la interfaz gráfica** **Diagrama de clases** **Código** CAPA APLICACIÓN Diagrama de clases Diagramas de secuencia Código programa Código pruebas JUnit	A par S14: Jueves 17 de abril
Versión dos **Persistencia**	Seleccionar elementos del juego Leer tablero Salvar guardar estado de juego Juego dos jugadores Todos los tubos Dragón verde Volar	**CAPA PRESENTACIÓN** **Boceto del la interfaz gráfica** **Diagrama de clases** **Código** CAPA APLICACIÓN Diagrama de clases Diagramas de secuencia Código programa Código pruebas JUnit	A par S16: Jueves 1 de diciembre
Versión tres	Funcionamiento logrado Explicitar mini-ciclos de desarrollo **PRUEBAS DE ACEPTACIÓN**	CAPA PRESENTACIÓN Boceto del la interfaz gráfica Diagrama de clases Código CAPA APLICACIÓN Diagrama de clases Diagramas de secuencia Código programa Código pruebas	A evaluador Inicial Miercoles 6 de diciembre Final Miercoles 13 de diciembre
COMPETENCIA	Es requisito para participar en la competencia que el equipo se haya presentado a todas las revisiones de pares y que todas las entregas del proyecto hayan sido aprobadas. El equipo ganador tiene 5.0 en la nota del tercer tercio.		Viernes 15 de diciembre