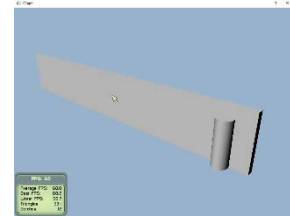
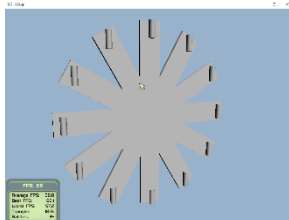


INFORMÁTICA GRÁFICA 2
Grado en desarrollo de videojuegos
Curso 2020-21
Práctica 1

(Práctica 1.1. Apartados 1-8)

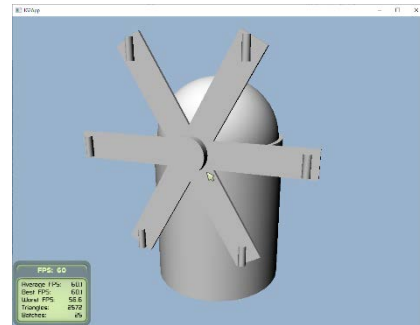
1. En **setupScene()**, construye un aspa que termina en un cilindro. Este aspa formará parte después de lo que será un molino. Las aspas se renderizan como lo que se muestra en la captura adjunta. Los nodos que las forman se estructuran en un nodo **aspaNode** del que cuelgan un **tableroNode** y un **cilindroNode**. Estos nodos se llaman “**aspa**”, “**tablero**” y “**adorno**”.



2. En **setupScene()**, construye **num** aspas como las que se muestran en la captura adjunta. En todas ellas, el cilindro de adorno tiene que estar vertical. La escena que se pide en este apartado está estructurada en un nodo “**aspas**” del que cuelgan **num** aspas, cuyos nodos y nombres están definidos como en el apartado anterior, respetando la unicidad de nombres. Es decir, hay nodos “**aspa_1**”, “**tablero_1**”, “**adorno_1**”, ...
- 
3. Añade un evento en la tecla **g** de **IG2App** de forma que, cuando se pulse, las aspas giren en sentido anti-horario, con todos los adornos cilíndricos de las mismas manteniendo la verticalidad en todo momento.
 4. Define la clase **Aspa**. Los objetos de esta clase son como los del apartado 1. Esta clase tiene pues atributos **SceneNode* aspaNode**, **tableroNode**, **cilindroNode**.
 5. Define la clase **AspasMolino**. Los objetos de esta clase son como los del apartado 2. Una forma de definir esta clase es únicamente con dos atributos **SceneNode* aspasNode** e **int numAspas**. Hazlo así. Fíjate que esto supone que la creación de los nodos para cada aspa, su tablero y su cilindro se hace explícitamente en la constructora de **AspasMolino**.
 6. Añade el evento **g** que gira las aspas al **keyPressed()** de la clase **AspasMolino** cuando la constructora de esta clase se define como en el apartado anterior.
 7. Haz lo mismo que en el apartado 5, pero define la constructora de forma que se construyan **numAspas** de la clase **Aspa**, guardándolas en un atributo de la clase **AspasMolino** llamado **Aspa** arrayAspas**, que es un array de tamaño **numAspas** cuyas componentes son de tipo **Aspa**.
 8. Añade el evento **g** que gira las aspas al **keyPressed()** de la clase **AspasMolino** cuando la constructora de esta clase se define como en el apartado anterior.

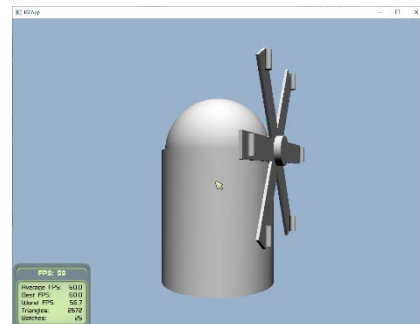
(Práctica 1.2. Apartados 9-17)

9. Define la clase **Molino** cuyos objetos se renderizan como en la captura adjunta. Su **mNode** tiene tres hijos: la esfera del techo, el cilindro del cuerpo del molino y las aspas. Las aspas se construyen como un objeto de la clase **AspasMolino**. Fíjate que ahora esta clase, aparte de los tableros con adorno cilíndrico, tiene un nodo más para el cilindro central. Evidentemente el evento **g** que giraba las aspas, manteniendo verticales los adornos, debes hacer que siga funcionando.

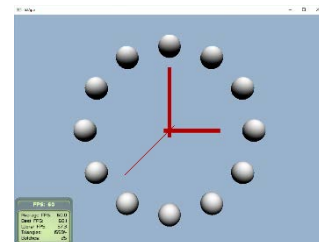


10. Añade el evento **c** a la clase **AspasMolino** de manera que el cilindro central se retire hacia atrás cuando la tecla es pulsada.

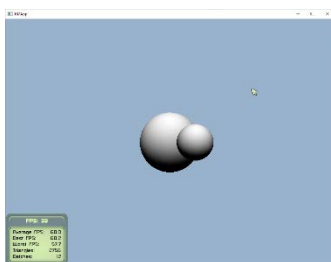
11. Añade el evento **h** a la clase **Molino** de manera que las aspas con su cilindro central roten alrededor del (techo del) molino. En la captura adjunta las aspas se han girado y han pasado de mirar de frente a mirar a la derecha. Implementalo de dos formas diferentes: usando la técnica del nodo ficticio y la del truco.



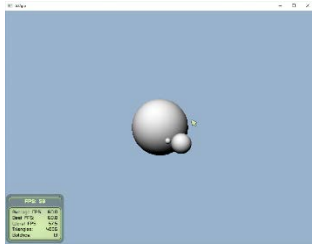
12. Retoma la escena del reloj de la **Práctica 0**. Aplica transformaciones adecuadas de las tres fundamentales (escalación, traslación, rotación) con respecto a alguno de los sistemas posibles (global, paterno, local) que permitan dejar la aguja de los segundos como aparece en la captura adjunta. El color de las agujas de tu escena debe ser gris, no rojo, por supuesto.



13. Añade el evento de teclado **h** que gira únicamente la aguja de los segundos alrededor del reloj, en sentido horario, claro.

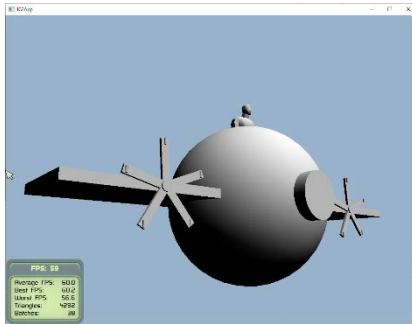


14. Define una escena con dos nodos, uno tiene una esfera que representa el sol y el otro una esfera más pequeña que representa la tierra. Añade el evento de teclado **j** que rota la tierra alrededor del sol, pero suponiendo que los dos nodos son independientes, el del sol no se mueve y no se usan nodos ficticios ni reposicionamiento de la tierra.



15. Define una escena con tres esferas para el sol, la tierra y la luna. Redefine el evento de teclado del apartado anterior para que la tierra rote alrededor del sol en horario y la luna alrededor de la tierra en anti-horario.

16. Define la clase **Avion** cuyos objetos son como el que se muestra en la captura adjunta, a la izquierda. El **mNode** de un avión tiene, como hijos, los siguientes nodos:



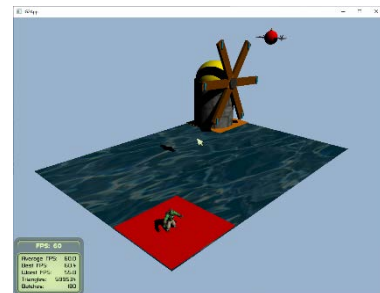
- **cuerpoNode** para la esfera de la nave
- **alaINode** y **alaDNode** para las alas izquierda y derecha y que tienen, **cada una**, un cubo escalado apropiadamente, es decir, no es correcto representar las dos alas del avión mediante un listón único que atraviesa la esfera
- **frenteNode** para el cilindro de la parte delantera del avión
- **pilotoNode**, para el piloto ninja
- dos **heliceNode**'s que son elementos de la clase **AspasMolino** y que están situados en la mitad del borde de las alas.

17. Define el evento de la tecla **g** de manera que roten las aspas de las hélices del avión, a la vez y en sentido horario.

(Práctica 1.3. Apartados 18-23)

18. Construye la malla de un plano y agrégala como entidad al **setupScene()** de **IG2App** dentro de un nodo de la escena.
19. Extiende tu proyecto con la clase **EntidadIG** que se define tal como se explica en las transparencias.
20. Refactoriza las clases **Aspa**, **AspasMolino**, **Molino** y **Avión** de forma que hereden solamente de **EntidadIG**.
21. Define la clase **Plano** que hereda de **EntidadIG** de forma que el plano que creabas antes en **IG2App** pase ahora a ser creado en la clase **Plano** y, a partir de él, crea una entidad que irá asociada al nodo principal de esta clase.
22. Tras los cambios de los tres apartados anteriores, asegúrate de que el evento de la tecla **g** siga haciendo lo mismo.

23. Define una escena que muestre los elementos que aparecen en las capturas adjuntas, sin colores ni texturas ni sombras, por supuesto. Se trata de un molino sobre un plano en una esquina de otro plano. En la esquina opuesta hay un Simbad sobre otro plano. Hay tres planos en total, pues. Y flotando sobre todos ellos, hay un avión. Son nodos principales creados en el **setupScene()** de **IG2App**, para esta escena, el del molino, el del avión, el que contiene Simbad y el de cada uno de los tres planos.



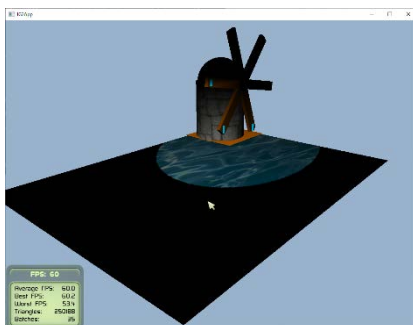
(Práctica 1.4. Apartados 24-29)

24. Añade el método:

```
virtual void frameRendered(const Ogre::FrameEvent& evt) {}
```

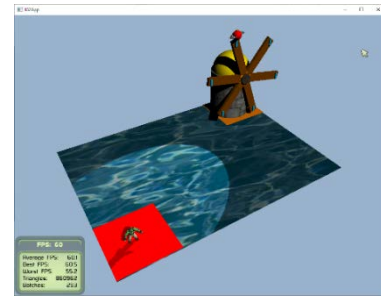
a **EntidadIG**. Haz que **Molino** implemente este método de forma que las aspas del molino giren sin parar.

25. Añade el método **frameRendered()** a la clase **Avión** de forma que las hélices del avión de la escena roten sin parar y que el avión esté dando vueltas continuamente sobre el plano del río, sin cruzarse con las aspas del molino que, por el apartado anterior, también rotan sin parar.
26. Añade un foco al avión de forma que, en ausencia de la luz direccional inicial de la escena, el foco gire con las rotaciones del avión. La dirección de emisión del foco debe ser **(1, -1, 0)**. Las capturas adjuntas, que no tienen sombras, intentan mostrar esto, con la escena vista desde esquinas diametralmente opuestas de la misma.

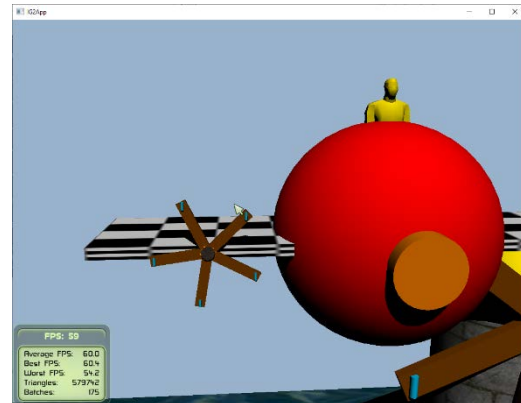
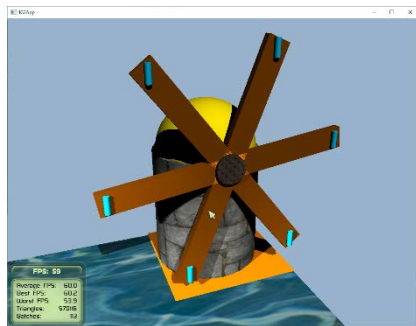


27. Añade sombras a la escena como se muestran en la captura adjunta.

28. Define la clase **Simbad**, que hereda de **EntidadIG**, uno de cuyos objetos es el situado encima del plano rojo de la escena. Recuerda que la malla de la entidad de esta clase es `Sinbad.mesh`.



29. Añade material a la escena mediante un script `Practica1.material`. Deduce de las siguientes capturas las texturas y colores de los elementos que aparecen en la escena.



(Práctica 1.5. Apartados 30-38)

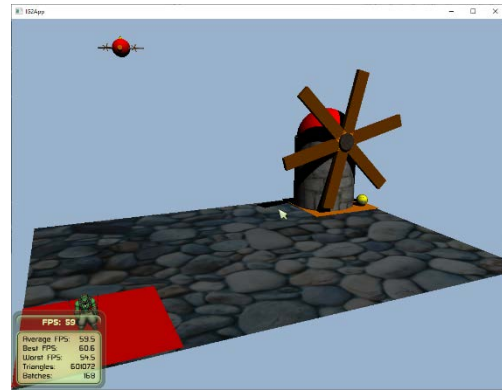


30. En el `setupScene()`, añade una cabeza que vigila todo y que se encuentra situada al lado del molino, tal como se muestra en la captura adjunta. Se trata de una esfera que tiene la textura que aparece en la captura, pero con efecto billboard.

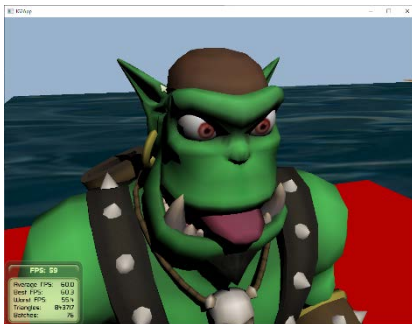


31. Programa el evento que se describe a continuación, mediante el envío de eventos apropiados a través del vector estático `appListeners`. Al pulsar la tecla **r** se producen los siguientes acontecimientos cuyo resultado final es el que se muestra en la captura adjunta:

- el agua del plano se para y pasa a verse el fondo del río que está lleno de piedras
- las aspas del molino se detienen y el techo se vuelve rojo
- los adornos de las aspas desaparecen
- el avión se detiene, pero no el giro de sus hélices
- la luz del foco del avión se apaga.



32. Anima los objetos de la clase **Sinbad** con el método `frameRendered()` usando **Dance** para crear el estado de animación. Abajo se muestra uno de los frames de la animación.



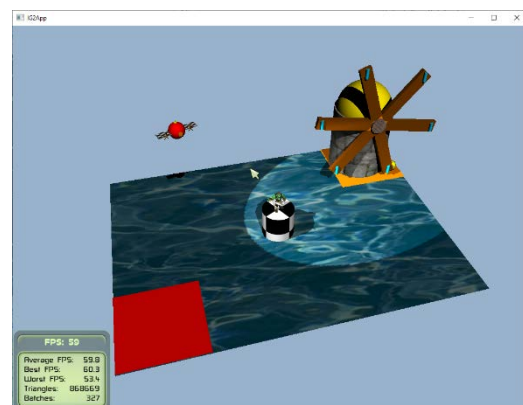
33. Anima **Sinbad** con los estados de animación **RunBase** y **RunTop**.

34. Haz que con la tecla **c** se pueda cambiar la animación de **Sinbad** de bailar a correr y viceversa, pulsando otra vez la tecla. Cuida que, al cambiar de animación, **Sinbad** haga exactamente lo que debe en su estado de animación y no se quede en una mezcla de ambos.

35. Haz que con la tecla **e** se pase de adjuntar una espada a **Sinbad**, de la mano derecha a la mano izquierda y viceversa, de la izquierda a la derecha, pulsando otra vez la tecla.

36. Define la clase **Boya**, que hereda de **EntidadIG**, cuyos objetos tienen una entidad con la malla **Barrel.mesh** asociada que se mostrará con **checker.png** como textura.

37. Programa la animación de nodo **AnimVV** explicada en las transparencias, que mueve el cilindro de la clase **Boya** de arriba abajo, con giro de 45 grados con respecto al eje **Y**. El cilindro está en medio del río como muestra la captura adjunta. No tengas en cuenta el Sinbad que hay sobre la boya.



38. Programa la animación de nodo que manda a **Sinbad** desde el centro de su plano rojo al centro del río, lo hace volver y vuelta a empezar. **Sinbad** se encuentra en el estado de animación **RunTop/RunBase**, y corre blandiendo una espada en su mano derecha.

No muestres la boya del apartado anterior. Cuando va por el río, **Sinbad** anda sobre las aguas, que para algo es Sinbad el marino. En las capturas adjuntas se le ve alejarse y volver.

