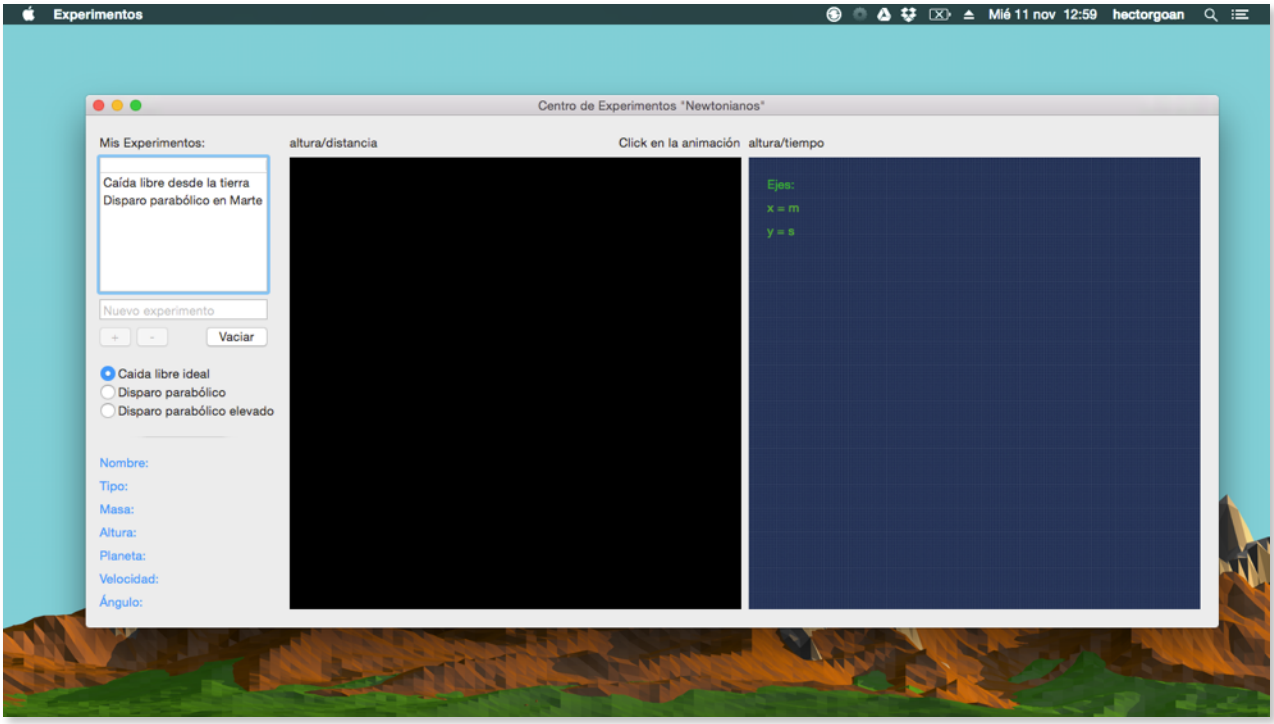
EXPERIMENTOS

Manual de usuario: Experimentos

Al inicial el programa nos encontramos con la ventana principal del mismo.

En esta ventana, visualizamos una lista de experimentos situada en la parte izquierda superior. Si disponemos de experimentos de otras sesiones, estos serán mostrados en la lista de manera automática. Los cambios realizados sobre la lista de experimentos se guardan automáticamente.

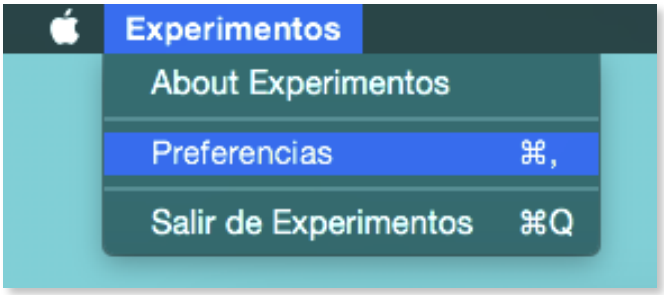


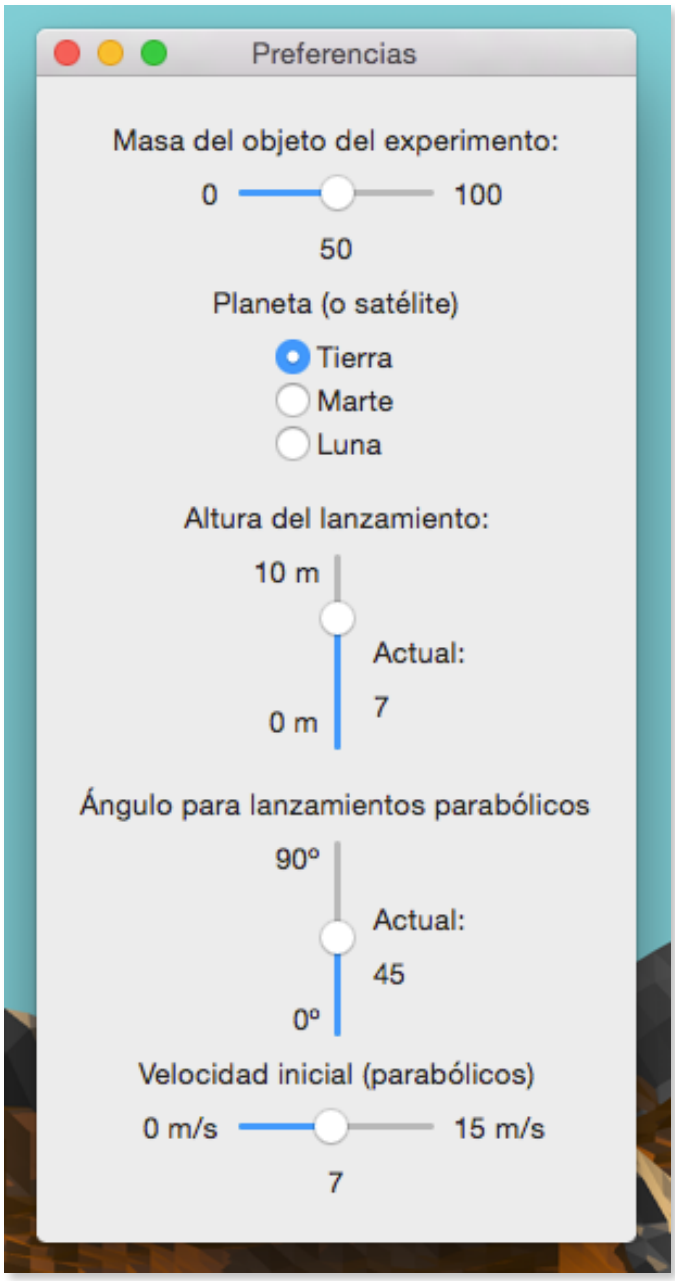
Justo por debajo de la lista de experimentos, nos encontramos con un campo de texto, acompañado de varios botones. En el campo de texto introduciremos el nombre del nuevo experimento. El botón “ + “ lo añade a la lista, “ - “ eliminará el experimento que se tenga seleccionado en ese momento, y “ Vaciar “ nos quita todos los elementos.

Si usted intenta de crear un experimento con el mismo nombre que otro ya existente, se le notificará, y tendrá que probar con otro nombre.

El siguiente set de botones nos permiten elegir de una manera rápida el tipo de experimento que queremos generar.

Si queremos configurar otros parámetros, tendremos que abrir el panel de preferencias. (⌘,)





En el panel de preferencias podremos configurar diferentes parámetros para los experimentos que se generen a partir de los cambios aquí realizados, y hasta que se vuelvan a cambiar o se produzca un reinicio de la aplicación, ya que carga los valores por defecto.

El primero es la masa del objeto con el que se va a experimentar.

Esto, aunque no afecte a los experimentos, se hace para que el usuario llegue a la conclusión de que en condiciones ideales, la masa no afecta al experimento.

El siguiente parámetro es el planeta en el que realizar el experimento a generar. Nos afectará cambiando la gravedad en las demostraciones.

Después configuraremos la altura del lanzamiento, la cual afecta a caída libre, y a tiro parabólico con elevación inicial.

A continuación se configura el ángulo de lanzamiento en los tiros parabólicos.

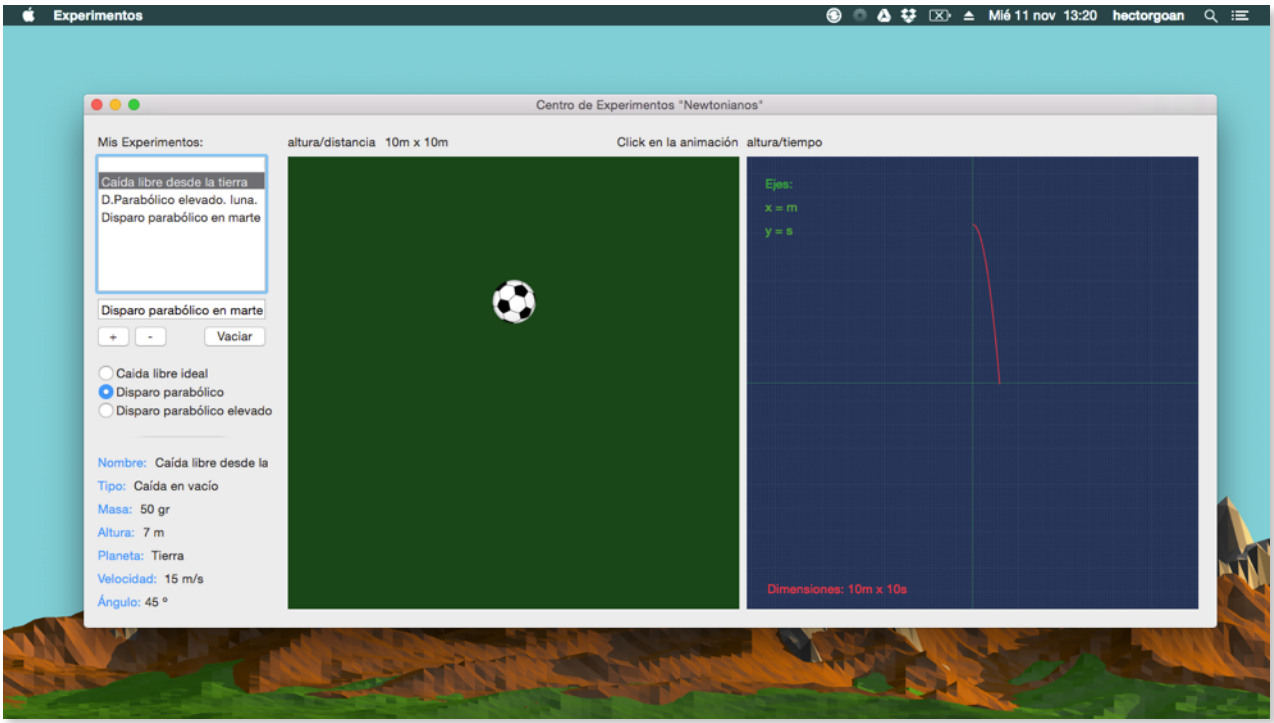
Para finalizar, podremos dar un valor a la velocidad inicial con la que se realiza el lanzamiento en los tiros parabólicos.

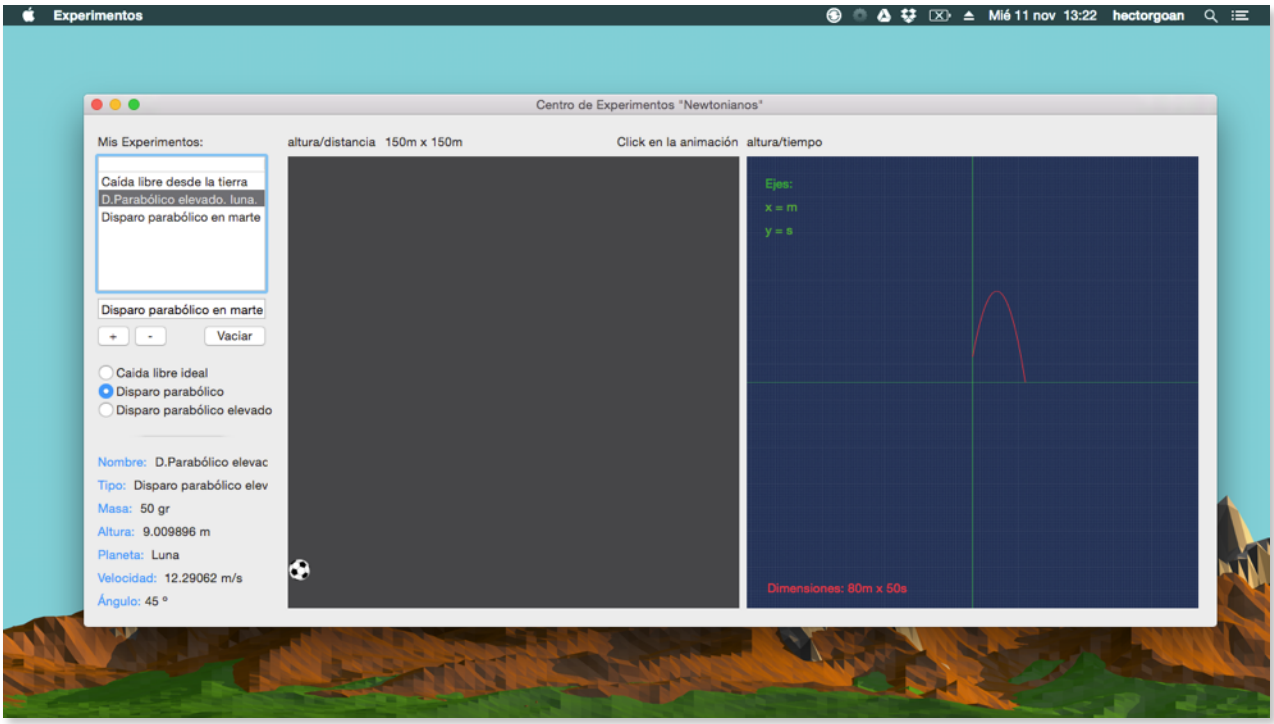
De vuelta en la ventana principal, nos encontraremos con 7 etiquetas que, cuando un experimento de la lista sea seleccionado, nos mostrarán información relevante al mismo.

Para finalizar, tenemos los elementos principales de esta aplicación, los cuadros de visualización.

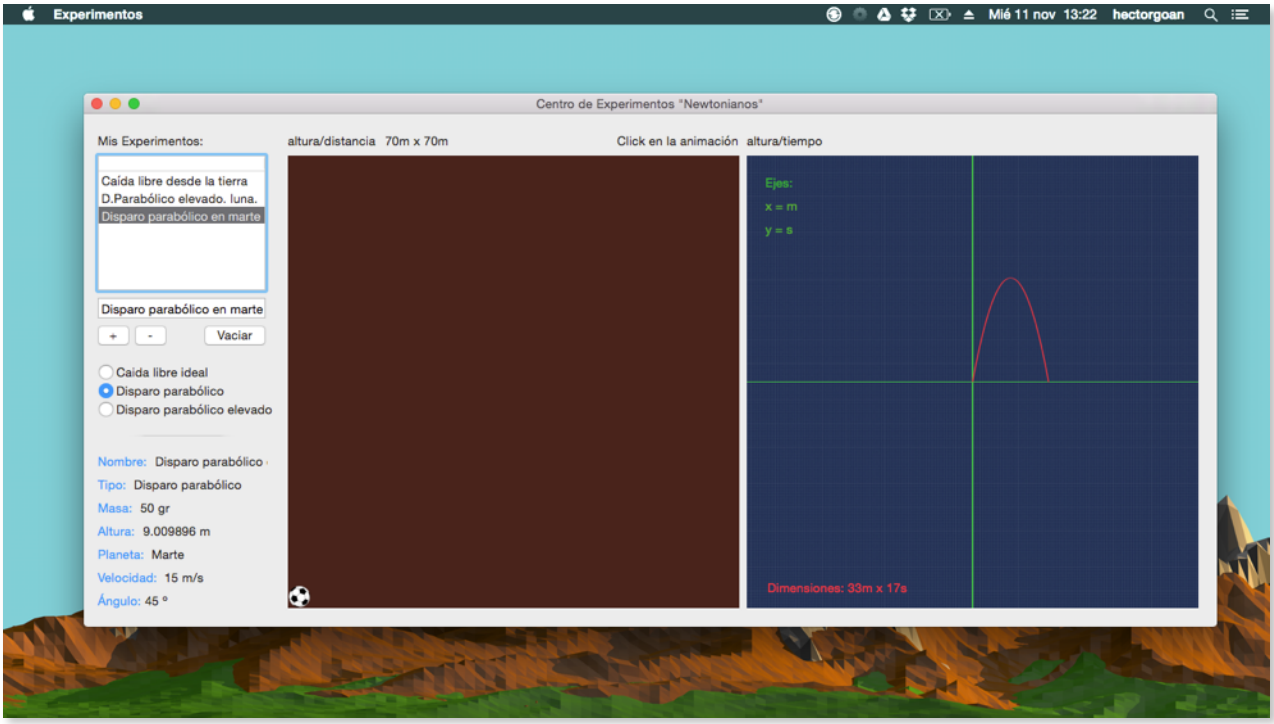
El primero nos mostrará una animación altura/distancia del experimento seleccionado, la cual activaremos haciendo click en el cuadro. Dependiendo del planeta en que nos encontremos, tendremos una respuesta visual inmediata, cambiando el fondo de la animación a verde si estamos en la Tierra, rojizo si estamos en Marte, y grisáceo si nos encontramos en la Luna.

El segundo muestra una gráfica altura/tiempo para que nos podamos hacer una idea del tiempo de vuelo de cada experimento.





La escala de ambas visualizaciones variará según sea el tipo del elemento a visualizar. La escala se visualiza inmediatamente en la interfaz.



Manual del programador y explicación de app : Experimentos

Esta práctica planteaba el reto inicial de crear un sistema con algunos de sus componentes comunicados mediante notificaciones, otros (o los mismos) haciendo uso intensivo de la delegación, manteniendo un sistema de carga y guardado de datos, y, como punto final dibujado de gráficas y/o animaciones.

Desde el punto de vista técnico, lo ideal sería que lo primero fuera crear todo el sistema de arquitectura, y el de diseño de clases, pero, desde el punto de vista del programador, decidí empezar afrontando por separado los que sabía que iban a ser los componentes del sistemas que mas tiempo/aprendizaje me iban a llevar.

Para empezar, me puse con el tema del guardado de datos, y, tras haber creado un sistema precario de carga/lectura de archivos y parseados JSON, la cosa se iba complicando a la hora de recuperar las propiedades de los objetos. De esta guinda me encontré con el sistema de "user defaults", el cual nos permite hacer todo este trabajo aplicando un par de líneas donde queramos cargar datos y otro par de ellas donde queramos guardarlos (actualizando estos user defaults).

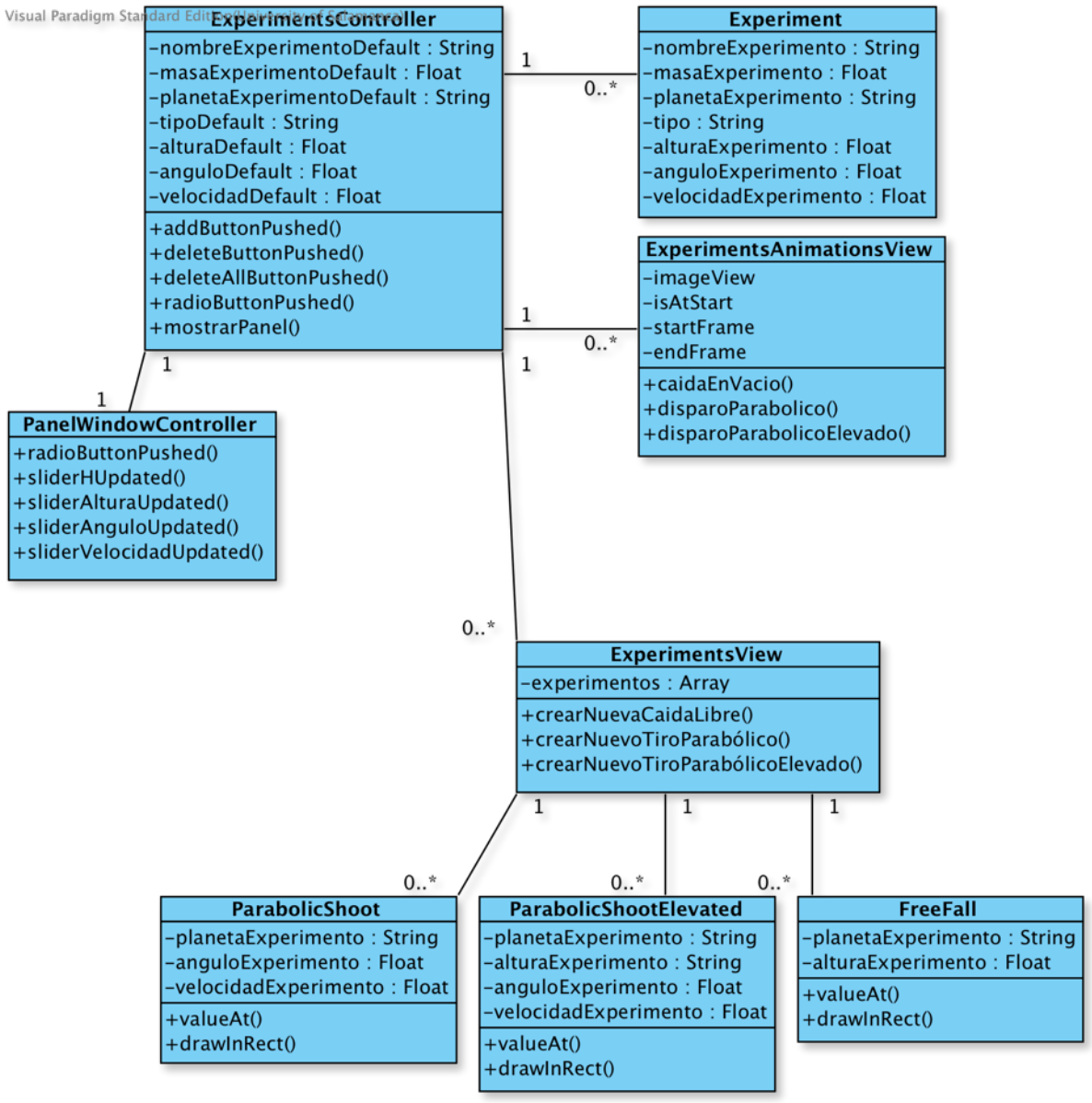
El siguiente reto fue el de realizar las gráficas altura/tiempo de los experimentos. Para esto, seguí la guía vista en clase, y lo que hago es dibujar los ejes para la gráfica en color verde, y luego me sitúo en el origen de esa gráfica, para esbozar un "Bezier Path", y, simulando en el eje de las x un sistema de tiempo, voy calculando el valor de la altura para cada uno de esos momentos. Cabe destacar que en todos los visualizaciones se han utilizado las ecuaciones reales para estos experimentos en condiciones ideales, esto es en el vacío.

Para finalizar, lo más costoso sin duda ha sido el tema de las animaciones. Empecé a hacerlas como en el ejemplo visto en clase, y tras haber hecho solo la animación de la caída libre, el código se complicaba demasiado, por tanto decidí buscar alguna alternativa para realizar animaciones en Cocoa. *El proyecto hasta este punto se adjuntará en la entrega bajo el nombre de "Experimentos Incompleto"*.

Esta alternativa tenía que ser sin utilizar bibliotecas externas a lo que es Cocoa. De esta manera, me puse a indagar, y encontré el sistema Core Animation, que mediante la inclusión del framework QuartzCore permitía la realización de las animaciones de una forma que a mi me pareció mas sencilla. Además, debido al sistema de suavizado de la transición punto a punto que presenta Core Animation se permite, con el uso de menos puntos de muestreo, una fluidez de animación asombrosa.

Aprender Core Animation fue sencillo, y no me llevó mas de un par de horas gracias al siguiente repositorio de Github: "https://github.com/lucasderraugh" y sus correspondientes explicaciones en YouTube: "https://youtu.be/8WqVCNkM7VM"

Diagrama de clases:



Explicación de las clases:

Experiment:

Esta clase es la que nos va a definir el tipo de objetos a manipular.

Consta de los métodos constructores (proporcionados por @property), además de los métodos init, y de los necesarios para poder codificar el tipo de objeto para que se pueda guardar en los user defaults.

experimentosController

Delegado de ventana, de tabla y de campo de texto. También es fuente de datos de la tabla.

A continuación se explican los métodos mas importantes.

-init:

Utilizamos dos listas en este método, una para los experimentos, y otra para los nombres de los experimentos. Se inicializan ambas y se establecen los valores por defecto para los experimentos a generar, nos suscribimos a las notificaciones que nos interesan, y cargamos los datos del user defaults.

-awakeFromNib:

Activación o desactivación de botones según convenga.

-addButtonPushed:

Se añadirá un nuevo experimento con los parámetros marcados como default en ese momento (si se hubiera cambiado mediante el panel, ya se habrían actualizado). Se activan o desactivan botones y se actualizan los user defaults.

-deleteButtonPushed

Se elimina de la lista el elemento seleccionado en la tabla, se vacían las etiquetas de mostrar características de experimento, se manda notificación para limpiar gráficas, y se actualiza el user defaults.

-deleteAllButtonPushed

Se eliminan todos los elementos de la lista se vacían las etiquetas de mostrar características de experimento, se manda notificación para limpiar gráficas, y se actualiza el user defaults.

-radioButtonPushed

Se cambia el tipo del experimento a crear.

-mostrarPanel

Se activa con las preferencias y nos lanza el panel que nos permite cambiar ajustes para los experimentos.

Lo siguiente son métodos a implementar por la utilización de patrones.

PanelWindowController

Maneja las preferencias de los experimentos. Cuando se cambia algún parámetro, se notifica a experimentosController para que lleve a cabo las acciones oportunas.

ExperimentsView

Esta clase nos controla la creación de gráficas para el experimento seleccionado en la tabla. Establece el fondo, y:

Si el elemento seleccionado en la tabla es de tipo caída libre, se llama a FreeFall que nos dibujará la gráfica de una caída libre, teniendo en cuenta los parámetros reales, y mediante la ecuación de caída libre ideal.

Si el elemento seleccionado en la tabla es de tipo tiro parabólico, se llama a ParabolicShoot que nos dibujará la gráfica de un tiro parabólico, teniendo en cuenta los parámetros reales, y mediante la ecuación de tiro parabólico ideal.

Si el elemento seleccionado en la tabla es de tipo tiro parabólico elevado, se llama a ParabolicShootElevated que nos dibujará la gráfica de un tiro parabólico elevado, teniendo en cuenta los parámetros reales, y mediante la ecuación de tiro parabólico elevado ideal.

ExperimentsAnimationView

Recibe señales de petición de dibujado de gráfica dependiendo del elemento de la tabla seleccionado, y cuando las recibe, utiliza sus métodos (consultar en el diagrama de clases), para generar la animación utilizando Core Animation.

Las animaciones se generan utilizando las fórmulas físicas reales en condiciones ideales para esos experimentos.