Interfaces Graficas de Usuario

Aplicación para la representación gráfica de funciones en Objective-C con Cocoa

Francisco Pinto Santos

Manual de usuario

En la Figura 1.1 esta presentadas las ventanas que se despliegan al ejecutar, con los elementos numerados:

- Ventana de representación: se trata del elemento 1. Es donde aparecerá la representación que configure el usuario en la ventana contigua.
  Cada vez que el usuario añada, elimine, modifique, … la representación cambiara en tiempo real de forma automática.
- Parámetros de configuración de la representación: Se trata del elemento 4, y son los parámetros que definen los limites de la representación.
  El botón “save”, sirve para guardar dichos parámetros, y está habilitado únicamente cuando estos son válidos.
  Una vez que el usuario pulse dicho botón, estos parámetros se actualizarán, y se verá reflejado en la representación.
  Nota: Un efecto colateral de pulsar este botón, es que realiza un reset del zoom de la representación.
- Tabla de gestión de funciones: Es el elemento 2, y aquí se irán desplegando filas con funciones, según estas sean añadidas por el usuario.
- Botones de gestión de funciones: Son el elemento 3 y son 3 que serán explicados comenzando por la izquierda:
  - El primero se elimina todas las funciones añadidas.
  - El segundo elimina el conjunto de funciones seleccionadas por el usuario.
  - El tercero despliega una ventana para añadir función.

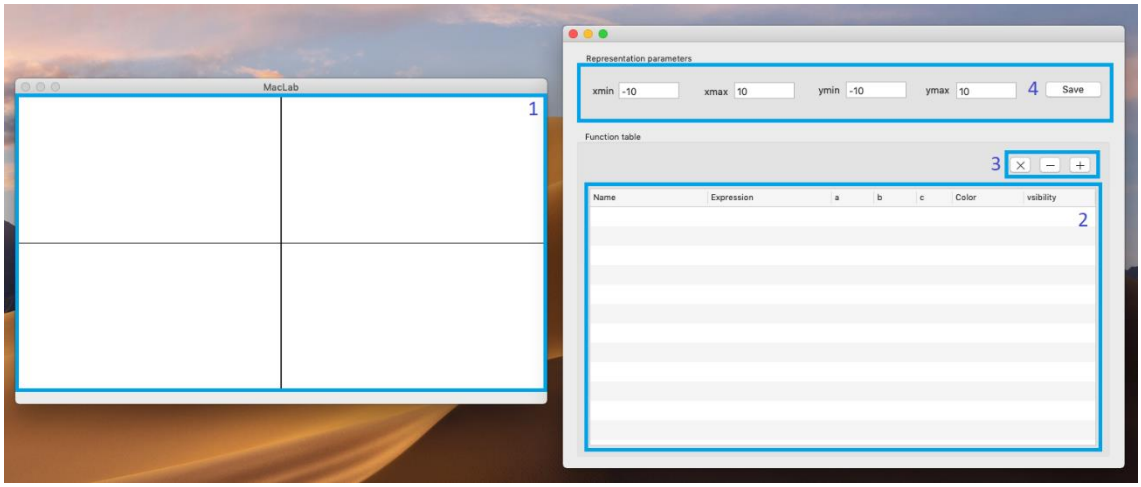

Figura 1.1

Para añadir una función, se tendrá que pulsar el botón añadir, el cual esta situado como el primero desde la derecha en el elemento 3 de la Figura 1. Esto desplegara la ventana que se puede apreciar en las figuras 1.2 y 1.3.
Esta ventana contiene los campos necesarios para crear una función, y hasta que estos estén rellenados debidamente y con datos correctos, el botón de añadir estará deshabilitado.
En el caso de que el usuario salga sin rellenar todos los campos, el programa lanzará una advertencia de que, si realiza esto, perderá la información introducida.

Nótese que en la figura 1.3 se ha desplegado un campo adicional, esto es porque este campo, se despliega únicamente cuando es necesario, y en nuestro caso solo es necesario para la función hipérbola.

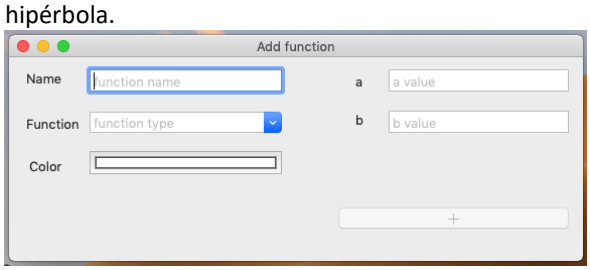

Figura 1.2

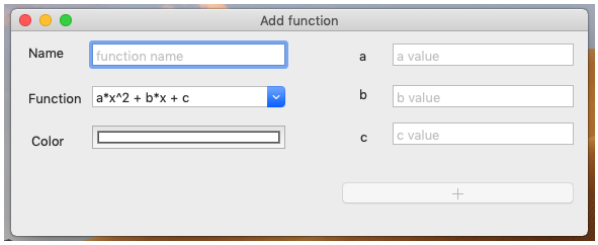

Figura 1.3

En cuanto a las opciones del menú, se encuentran todas las añadidas a la aplicación en las figuras 1.4 y 1.5.
Concretamente, en la figura 1.4, se observa el menú correspondiente a File, en el cual nos encontramos las siguientes opciones:

- Open: abre un fichero binario, el cual contiene funciones del programa, y borra las funciones creadas en el contexto actual.
  En el caso de fracasar lanza un mensaje de error para indicarle al usuario.
- Save As: guarda las funciones añadidas actualmente a un fichero binario, y al igual que en el caso anterior, en el caso de fracasar avisa al usuario.
- Print: exporta la representación a una imagen png, y si falla advierte al usuario.

En cuanto a la figura 1.5, solo cabe destacar la opción “reset zoom,” la cual reestablece los últimos límites de representación establecidos por el usuario.

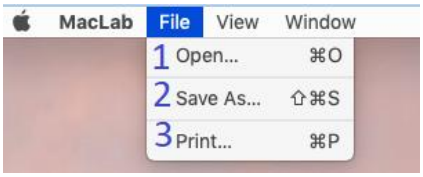

Figura 1.4

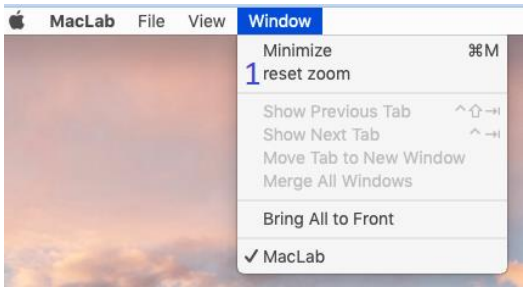

Figura 1.5

En la Figura 1.6 se aprecia un ejemplo de uso en el que se han añadido 5 funciones, las cuales son todos los tipos de funciones disponibles para el usuario.

Caben destacar los siguientes aspectos de la representación y la forma de trabajar con la aplicación:

- El primero de ellos es que, para editar las funciones, basta con cambiar cualquiera de los controles de la tabla de funciones, y esto se verá reflejado automáticamente en la representación.
- En el caso de que se quieran cambiar los parámetros a, b o c; estos solo cambiaran en el caso de que lo introducido sea un número, ya que están asociados con un validador de formato. Además, cuando la función no necesite usar el parámetro c, este será deshabilitado.
- Cuando ponemos el ratón sobre la representación, en la parte inferior de la ventana de representación, aparecerán las coordenadas del ratón, y cuando el ratón salga de la representación, estas desaparecerán.
- Por último, cabe destacar, que se puede realizar zoom (se puede apreciar el zoom en la Figura 1.7), pinchando en un punto y arrastrando hasta soltar en otro punto. Cuando se realice esto, se hará zoom sobre el recuadro que ha creado el usuario al moverse entre esos 2 puntos. Claro está, si el punto inicial o el final no se encuentran en la representación, no se realizará zoom.

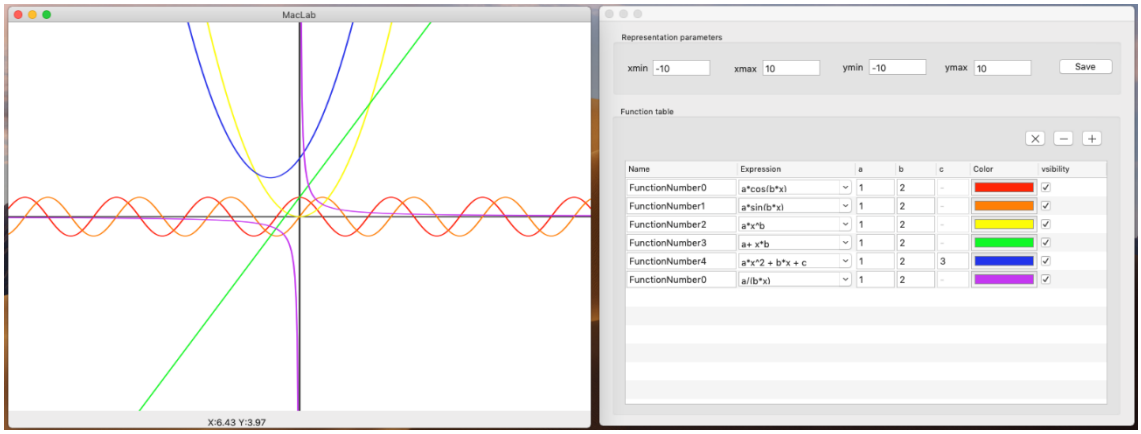

Figura 1.6

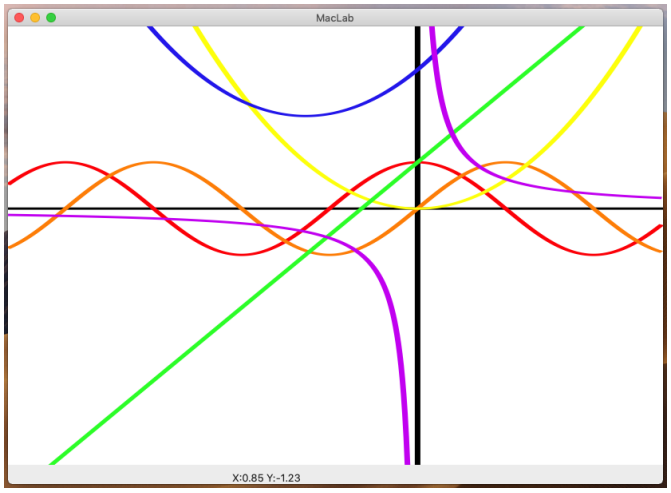

Figura 1.7

Manual del programador

Function
Se trata de un Bean que porta la información necesaria para realizar la representación de una función y los cálculos asociados a dicha representación.
Entre los métodos a destacar están valueAt que calcula un valor de y, para un punto x indicado. Y drawInBoundsWIthParametersWithGraphicsContext, el cual cuando es invocado es el encargado de realizar el dibujado de la función.
Esta clase tiene una enumeración asociada llamada FunctionType, la cual define el tipo de la función.

Model
Es el modelo, el cual almacena un conjunto de Function en un NSMutableArray, e implementa un CRUD sobre dicho conjunto de Function.
Es una clase singletone, es decir, de la cual se puede obtener una única instancia mediante el método defaultModel.
Cabe destacar que presenta dos métodos adicionales: exportFile e importFile, los cuales dada una ruta de un fichero .bin, realizan el exportado de los objetos Function almacenados; o el borrado de los objetos Function almacenados y el importado, respectivamente.

AddFunctionUIController
Se trata del controlador de la ventana para añadir funciones, es decir, el controlador asociado a el fichero AddFunctionUI.xib.
Al tratarse de una ventana secundaria, se trata de una instancia de un panel, que hereda de NSWindowController.
Implementa los protocolos NSComboBoxDelegate y NSTextFieldDelegate para poder deshabilitar el botón de añadir en el caso de que los datos necesarios para instanciar un objeto Function estén incompletos o sean incorrectos.
Cuando se pulsa el botón de añadir, se instancia el objeto Function, se guarda en el modelo, y por ultimo se envía una notificación, la cual será recibida por la clase FunctionTableUIController, para advertirle a esta de que se ha añadido un Function al modelo y tiene que refrescar sus datos. A su vez dicha clase también notificara a PlotrepresentationUIController mediante notificación de que los datos han sido actualizados en el modelo para que actualice la representación.
Entre sus métodos destacados están isFormCompleted que nos indica si el formulario esta completo y correcto mediante un BOOL; El método takeDataFromFormulary, el cual toma los datos del formulario e instancia un objeto Function el cual devuelve como parámetro; y el método cleanAndDeactivateFields, el cual limpia los campos y desactiva el botón de añadir, limpiando así el formulario, y dejándolo preparado para que el usuario cuando entre la próxima vez, no vea los datos del ultimo objeto que añadió.

FunctionTableUIController
Esta clase se encarga de hacer 3 tareas principales: lanzar la ventana de añadir función, gestionar la tabla de funciones, y notificar cuando el usuario cambia los parámetros de representación.

En cuanto a la ventana de añadir función, este controlador la instancia cuando se pulsa el botón de añadir función, en el caso de que no este instanciada; tras esto la lanza, y si el usuario introduce y guarda una función, a FunctionTableUIController le llegara una notificación de dicha ventana para que este refresque su tabla de funciones, y a su vez, FunctionTableUIController avisara por notificación a PlotrepresentationUIController para que este actualice su representación también.

En cuanto a notificar cuando el usuario cambia los datos de la representación: cuando el usuario cambia los parámetros de representación, si son correctos (todos son números y xmin<xmax e ymin<ymax), este los toma, y los mete en un diccionario para mandárselos a PlotRepresentationUIController a través de notificación.

Por último, la tarea más importante de esta clase es ser el delegado de la tabla de funciones, la cual se gestiona por delegación.
Al tratarse de una tabla basada en vistas, hay que implementar un método que devuelva el número de filas, y otro que es llamado una vez por cada celda, en el cual se toma el control que reside en la celda pertinente, y se le introduce el valor correspondiente. Además, para identificarlos, se les pone de tag el numero de fila, y se vinculan a un action especifico de su columna y se les pone de target la instancia de FunctionTableUIController que reside en la propia clase para gestionar cuando el usuario cambia uno de estos controles, para que la información de la función que representa ese control sea actualizada en el modelo.

NSPlotView
Se trata de un objeto que hereda de NSView, el cual es la vista personalizada, utilizada para representar las funciones almacenadas en el modelo.

Esta clase instancia FunctionTableUIController en su init, de tal forma que aparezcan las 2 ventanas a la vez; además, FunctionTableUIController, le manda notificaciones cada vez que cambian los parámetros de representación, o que se modifica alguna representación, para que esta actualice la representación.

Esta clase es la encargada de almacenar los limites lógicos (xmin,xmax,ymin e ymax) de representación, y los límites físicos (self.bounds), además de los límites actuales (los que se están usando ahora mismo debido al zoom), y si el zoom esta activo.

Al heredar de NSView cuando se llama a setNeedsDisplay sobre un objeto de esta clase, se llama al método drawRect para que se realice el dibujado sobre la vista. Dicho método se ha sobrescrito para que ponga un fondo blanco, dibuje los ejes, y llame a un datasource (implementa el ptotocol NSPlotViewDatasource), para que le diga el número de elementos; y luego una vez por cada elemento.

Esta vista también gestiona el zoom mediante los eventos de ratón mouseDown y mouseUp, y también avisa a el delegado NSPlotViewMouseEventsDelegate, cuando el ratón entra, sale o se mueve por la pantalla; para lo cual ha sido necesario usar un NSTracking área y sobrescribir el método updateTrackingAreas de la vista.

Entre los métodos destacados están reloadData, que sirve para encapsular el setNeedDisplay; resetZoom que sirve para resetear el zoom, y drawAxysInBoundsWithParametersWithGraphicsContext que se encarga de dibujar los ejes.
También cabe destacar setParameters que pone nuevos parámetros y resetea el zoom, y exportViewToPath, que dada una ruta exporta todo lo representado en ese momento en la vista a un fichero con formato png.

PlotRepresentationUIController
Este controlador es el encargado de gestionar PlotRepresentationUI, y su función principal es ser el delegado de tipo NSPlotViewDatasource y NSPlotViewMouseEventsDelegate del NSPlotView almacenado el dicho fichero; además es el controlador de la ventana principal.
Como delegado datasource del NSPlotView, recibe llamadas para pedir el numero de elementos, y para que cada uno de estos sea representado. Y como delegado de tipo mouseEvents, recibe las llamadas necesarias por medio de este protocolo, para gestionar la label de posiciones del borde inferior PlotRepresentationUI.

También cabe destacar que tiene implementados los métodos para gestionar el menú principal, los cuales son para resetear el zoom, para exportar e importar funciones y exportar la vista.
Por último, cabe destacar que tiene redefinido en método widowShouldClose para que realice un terminate sobre la aplicación, cerrando así con el programa.

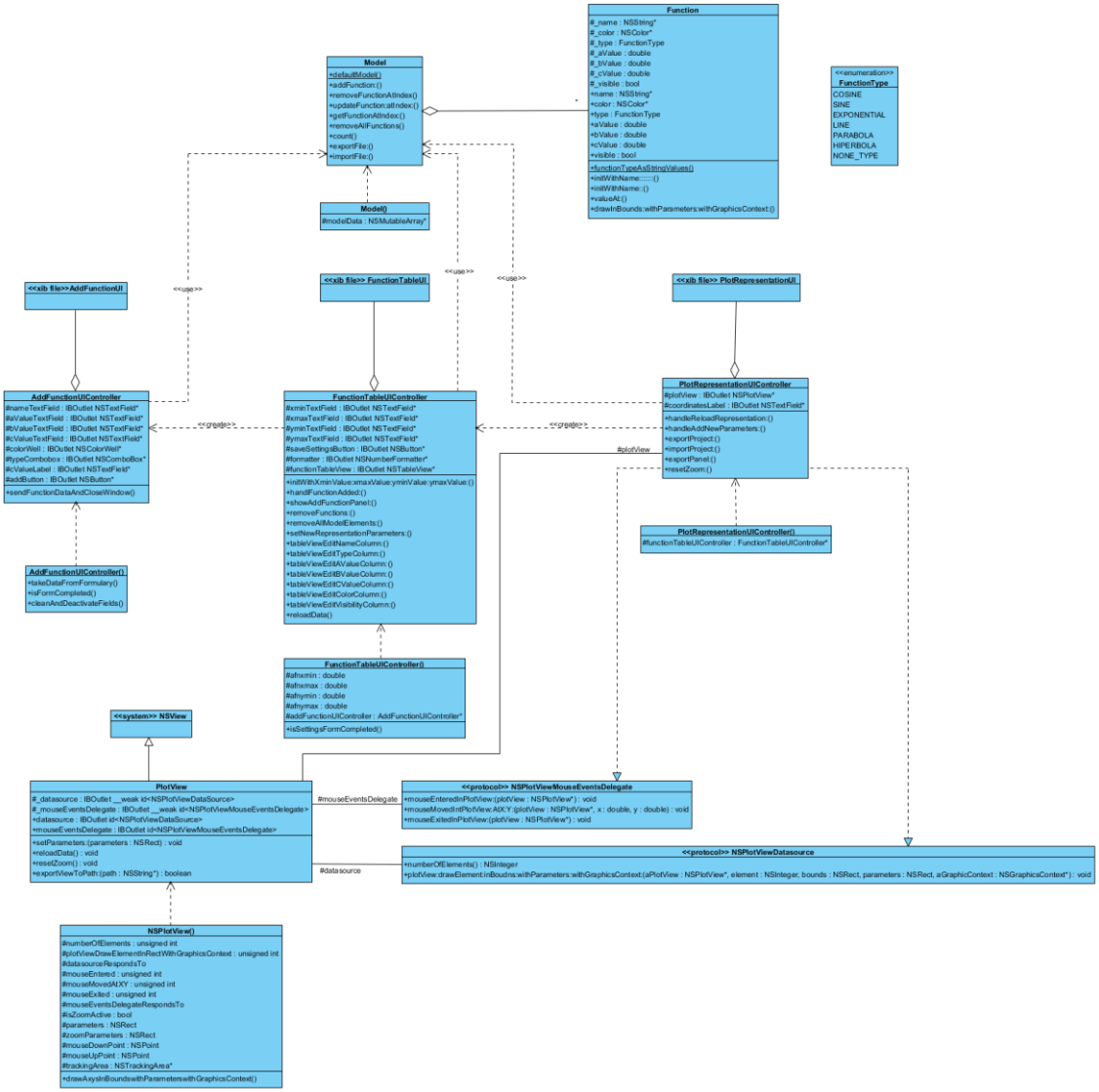

Figura 2.1. diagrama de clases de la aplicación

Referencias

- Apuntes de Cocoa de Interfaces Graficas de Usuario.
- Aaron Hillegass Cocoa Programming for MacOSX (4th edition): entre otros he consultado:
    - Buenas prácticas para cocoa (pag. 33) y (pág. 87)
    - Uso de NSCoder y el protocol NSCoding (pág. 261)
    - Uso de NSKeyedArchiver (pág. 269)
    - Creación de protocolos (pág. 279)
    - Uso de Paneles (pág. 307)
    - Uso y buenas practicas de las notificaciones (pág. 351)
    - Uso de eventos de ratón (pág. 427)
    - Uso del NSTrackingArea (pág. 467)
    - …
- Documentación de Apple: https://developer.apple.com/documentation/
- Stacks overflow: ha sido una de mis fuentes principales de consulta, y entre otros he consultado:
    - https://stackoverflow.com/questions/4639379/how-to-use-nstrackingarea
    - https://stackoverflow.com/questions/5544551/how-to-find-the-location-of-the-mouse-in-objective-c
    - https://stackoverflow.com/questions/435685/how-to-make-a-transparent-nsview-subclass-handle-mouse-events
    - https://stackoverflow.com/questions/4427370/mouse-moved-events-are-not-detected-by-nsview
    - https://stackoverflow.com/questions/3199888/the-status-of-a-checkbox-in-cocoa
    - https://stackoverflow.com/questions/9332667/cocoa-without-xcode
    - https://stackoverflow.com/questions/172598/best-way-to-define-private-methods-for-a-class-in-objective-c
    - https://stackoverflow.com/questions/4289511/nsopenpanel-set-file-type
    - https://stackoverflow.com/questions/3773180/how-to-get-indexes-from-nsindexset-into-an-nsarray-in-cocoa
    - https://stackoverflow.com/questions/47561382/nscolorwell-in-an-nstableview-in-swift
    - https://stackoverflow.com/questions/2879154/change-color-of-nstableviewcell
    - https://stackoverflow.com/questions/22692423/how-to-set-nstableview-background-color-to-clear-in-xcode
    - https://stackoverflow.com/questions/9440646/cocoa-programming-setting-the-delegate
    - https://stackoverflow.com/questions/28281045/view-based-nstableview-editing
    - https://stackoverflow.com/questions/2204572/nstableview-with-multiple-columns
    - https://stackoverflow.com/questions/7811776/validating-strings-using-nsnumberformatter
    - https://stackoverflow.com/questions/2670653/create-custom-button-with-image-in-interface-builder

    - https://stackoverflow.com/questions/29433487/create-an-nsalert-with-swift
    - https://stackoverflow.com/questions/2670653/create-custom-button-with-image-in-interface-builder
    - https://stackoverflow.com/questions/2477516/how-to-hide-the-label-cclabel-after-a-certain-time-in-cocos2d
    - https://stackoverflow.com/questions/172598/best-way-to-define-private-methods-for-a-class-in-objective-c
    - https://stackoverflow.com/questions/6817869/how-to-draw-graph-of-mathematic-function-in-objective-c
    - …
- Por último cabe destacar algunas webs:
    - http://beensoft.blogspot.com/2011/09/xcode-snippet-2-archiving-objects-with.html
    - https://gist.github.com/marteinn/540cfa4282add987ee6b
    - http://iosbrain.com/blog/2018/02/01/tutorial-delegates-and-delegation-in-objective-c/
    - https://www.ios-blog.com/tutorials/objective-c/how-to-create-an-objective-c-delegate/
    - https://spin.atomicobject.com/2014/02/03/objective-c-delegate-pattern/
    - https://mycodetips.com/objective-c/constructors-creating-objects-objective-c-1734.html
    - https://gist.github.com/soffes/812796
    - https://blog.teamtreehouse.com/beginners-guide-objective-c-classes-objects
    -
    - …