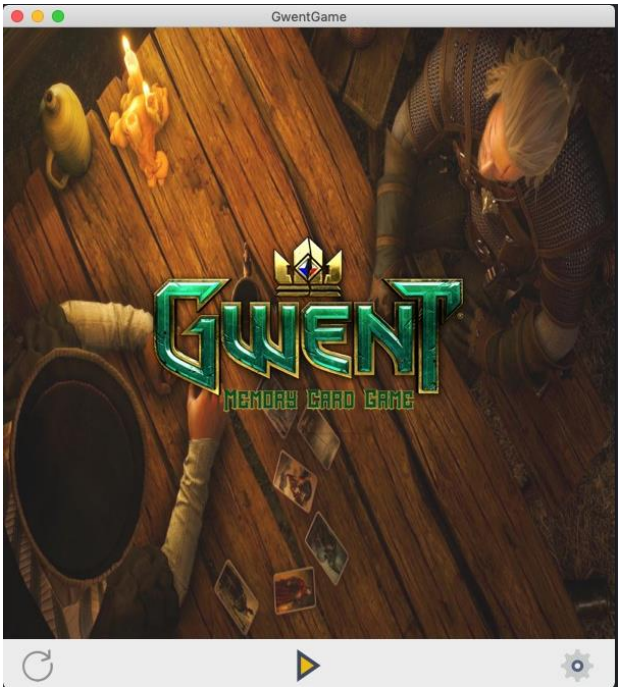
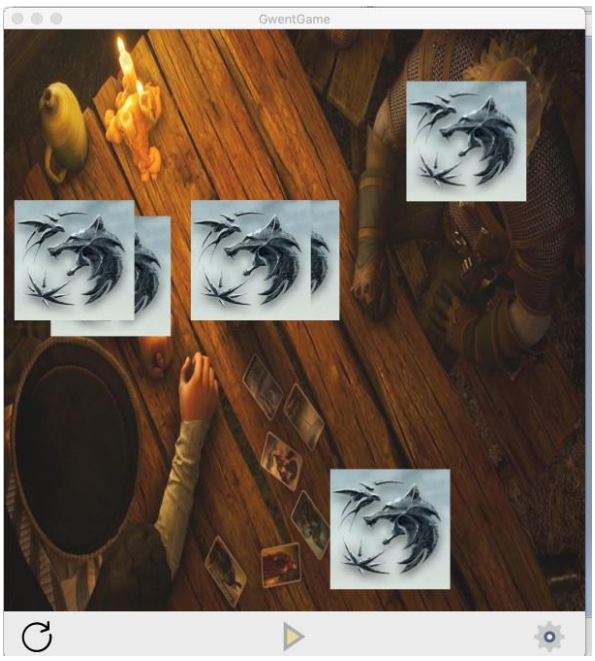
# MANUAL DE USUARIO

Al arrancar la aplicación se nos mostrará la ventana principal que tendrá el siguiente aspecto:



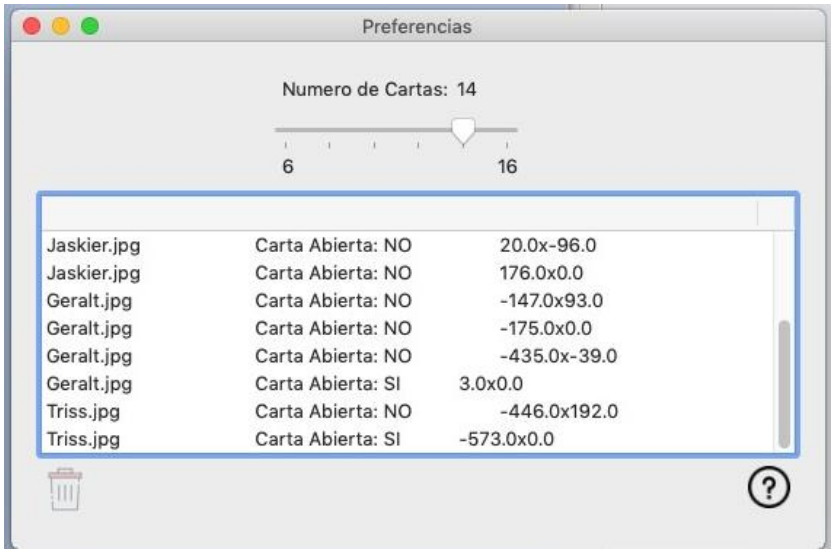

El botón más a la izquierda, el de refrescar, nos permitirá que el juego vuelva a comenzar en medio de una partida. El botón del medio arranca el juego. Y el botón más a la derecha, de preferencias, permite abrir el panel de preferencias con la tabla. También se puede hacer al panel de preferencia desde la barra superior: GwentGame → Preferences.

Una vez arrancado el juego, tenemos el siguiente aspecto:

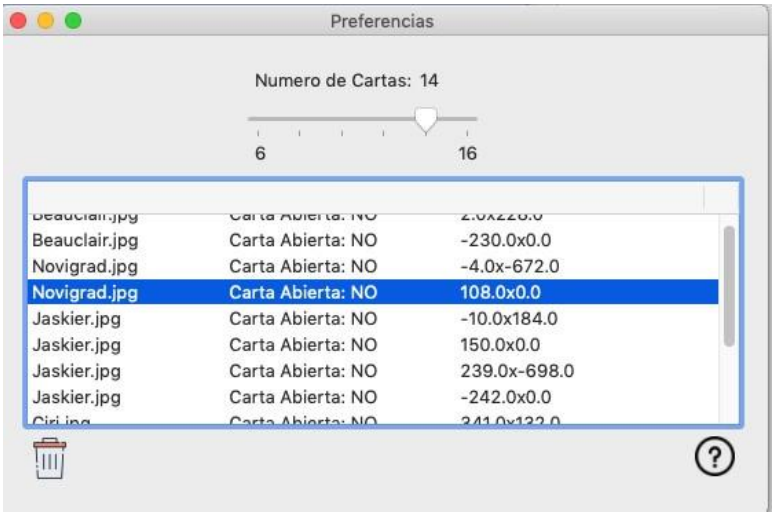



El juego consiste en encontrar la pareja de cada carta. Para ello al hacer click en una, la carta se abre, se da la vuelta, permitiéndote darle la vuelta a otra para saber si es la pareja o no. Si las dos cartas son la misma, se eliminan del tablero. En caso contrario, las cartas se dan la vuelta.

La tabla del panel de preferencias muestra las cartas que hay en el tablero, si están abiertas o no, así como las coordenadas donde se encuentran. También podemos cambiar el número de cartas con las que queremos jugar, la nueva configuración se aplicara al darle al botón de refrescar o al comenzar una nueva partida.

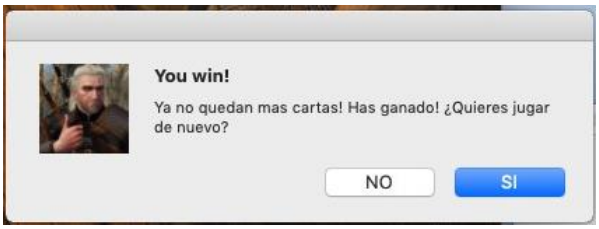




Si seleccionamos un elemento de la tabla, el botón de eliminar se activa, permitiendo borrar del tablero la carta seleccionada:

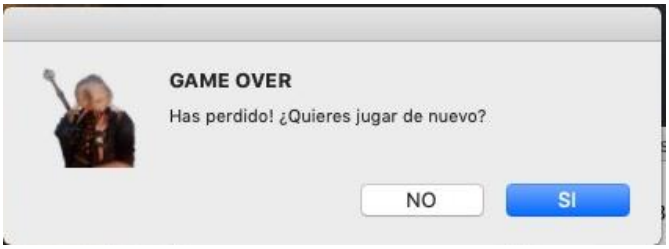


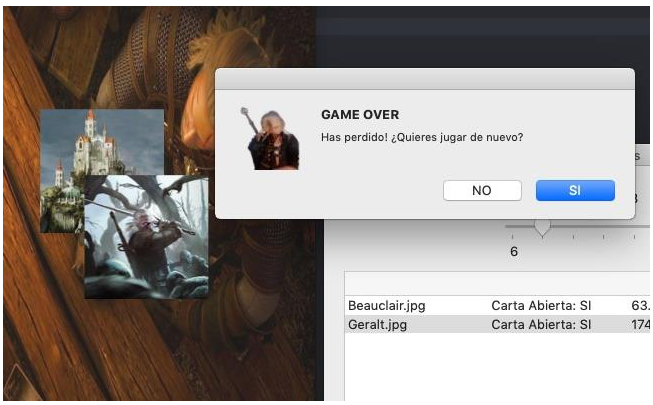
El botón de información del panel de preferencias nos abre un mensaje con información relativa al juego:

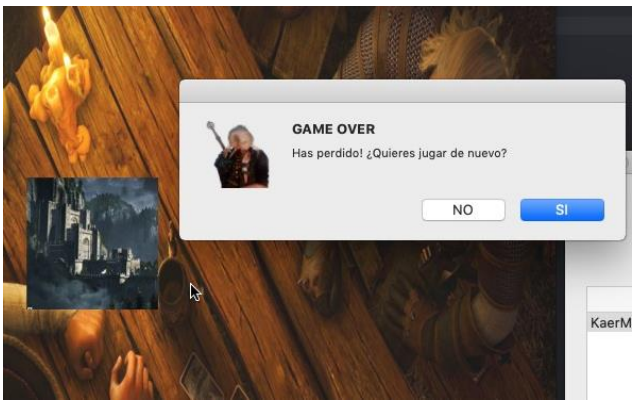

Si conseguimos encontrar las parejas de todas las cartas, nos saldrá un mensaje avisándonos y preguntando si queremos jugar otra partida. En caso afirmativo, comenzara otra partida, y en caso negativo, regresa a la pantalla principal.




Si borramos cartas desde la tabla puede darse el caso de que nos quedemos con una carta, o con dos cartas que no son pareja, entonces en ese caso el programa le dará la vuelta a las cartas que quedan y nos mostrara un mensaje de *game over* con una funcionalidad parecida al anterior:



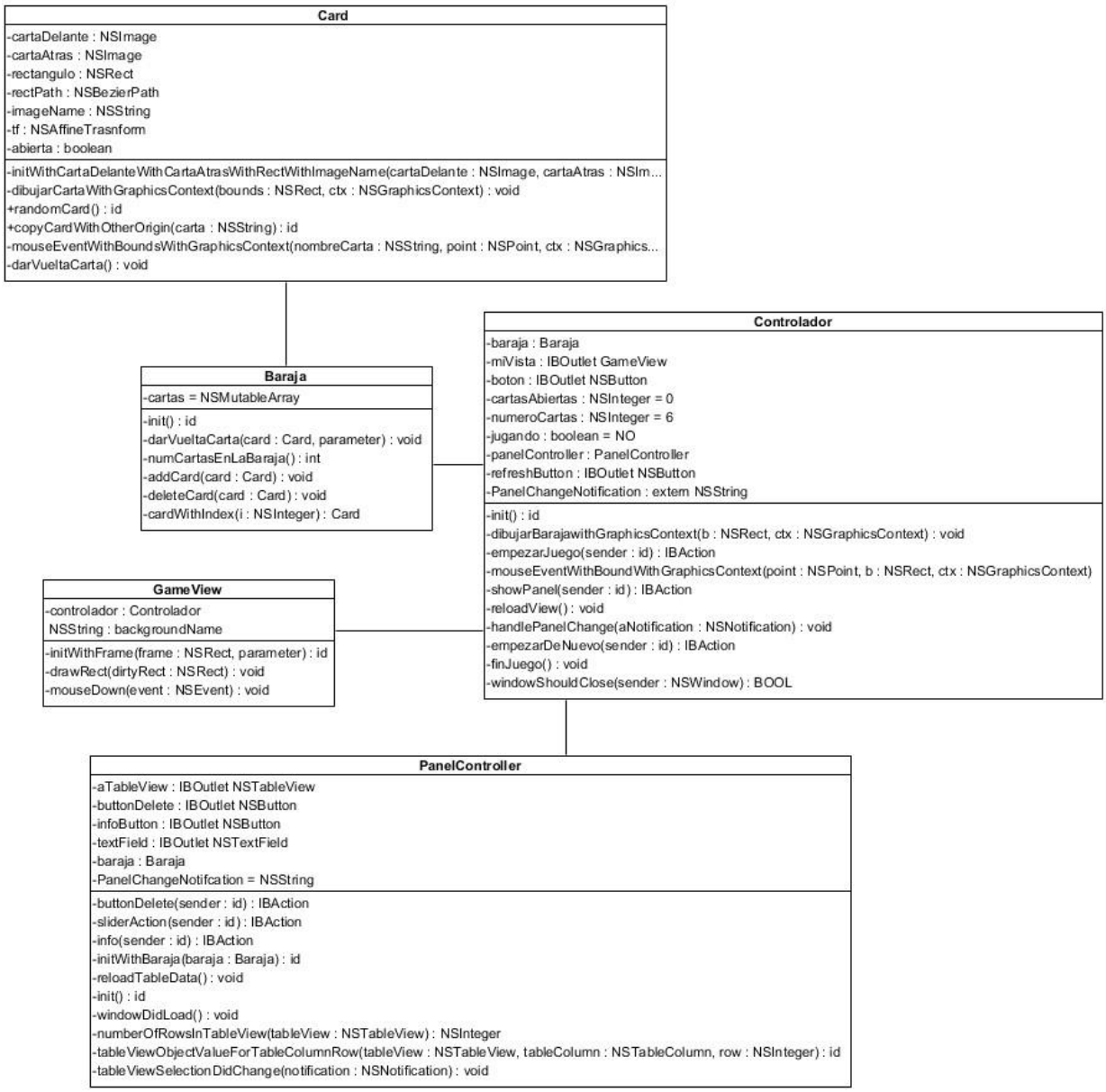



Por último, si damos a cerrar la ventana en la ventana principal nos saldrá un mensaje preguntándonos si queremos salir de la aplicación. En caso afirmativo el juego se cerrará.

# MANUAL DE PROGRAMADOR

Las clases del programa, con sus atributos y métodos son las siguientes:

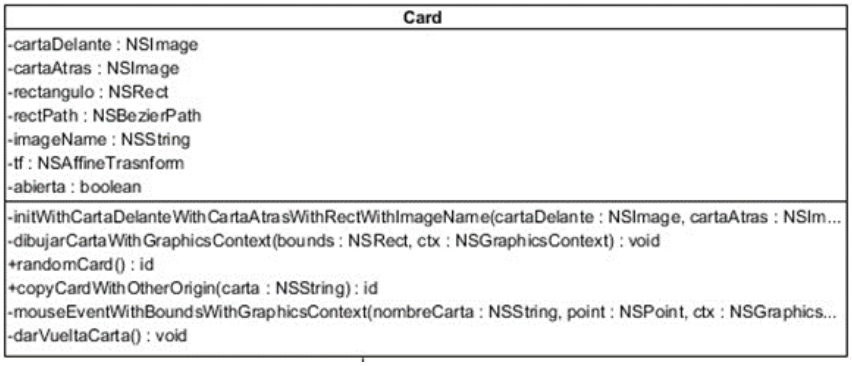



Las clases Card y Baraja representan el modelo de la aplicación.

Controlador y PanelController son las clases controladoras de ventana principal y del panel de preferencias, respectivamente.

GameView representa la clase asociada a la customView.

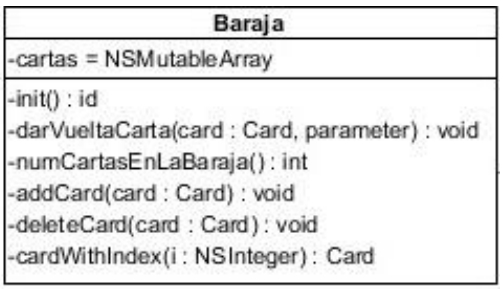
**Clase Card**:




Destacan los siguientes métodos:

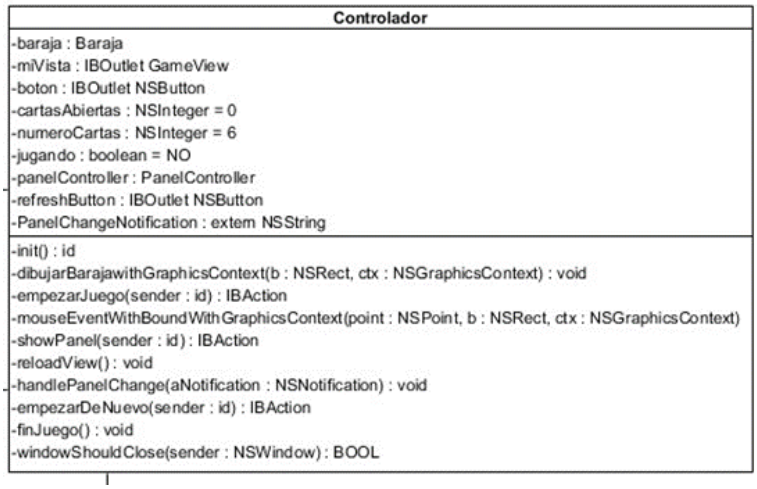
- dibujarCartasWithGraphicsContext recibe el contexto grafico actual de la customView y realiza las conversiones afines necesarias para dibujar las cartas correctamente.
- randomCard crea una instancia de la clase dándole un valor aleatorio al atributo ImageName, que será el nombre de la foto que representa a la carta, y también le da un valor aleatorio al atributo Origin del rectángulo para que se dibuje en una posición aleatoria del tablero.
- copyCardWithOtherOrigin recibe el atributo ImageName de una carta ya creada, y crea una instancia de Card con el nombre de la carta pasado y posiciones aleatorias en el origin del rectángulo.
- mouseEventWithBoundsWithGraphicsContext: recibe las coordenadas donde se encontraba el ratón al ser pulsado, las transforma al espacio de coordenadas del programa, y comprueba si esa posición se encuentra en el rectángulo de la carta, devolviendo el resultado en un booleano.
- darVueltaCarta: Para dibujar la carta se usa el atributo “cartaAtras”, entonces para representar que se le ha dado la vuelta una carta, se intercambia el valor que tiene el atributo “cartaAtras” por el que tiene “cartaDelante”.

**Clase Baraja**




Los métodos que se encuentran en la clase baraja son métodos que manejan el NSMutableArray de la clase.

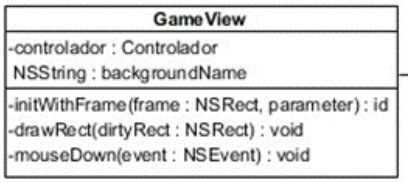
### Clase Controlador




Los métodos más destacables son:

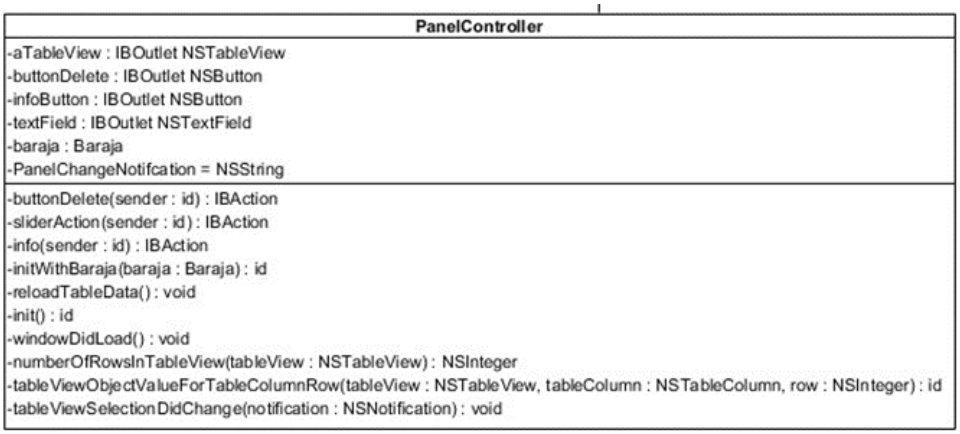
- empezarJuego: Este método es ejecutado cuando el usuario le da al botón “Jugar”, y se encarga de crear las instancias de la clase Card, creara el número de cartas que este indicado en numeroCartas. También avisara a la vista y a la tabla que necesitan actualizar la información.
- mouseEventWithBoundWithGraphics: Recorre la baraja llamando al método de la clase Card del mismo nombre buscando si la posición del ratón al ser pulsado corresponde con alguna carta. Si es así, le da la vuelta a la carta. Si detecta que hay dos cartas abiertas, comprueba si son iguales mediante el atributo “ImageName” de la clase Card, en caso de que sean iguales, las elimina del array. Si detecta que el usuario ha pulsado a una tercera carta, le da la vuelta a las dos anteriores, y abre la pulsada por el usuario. Al final del método, avisa a la vista y la tabla para que se actualicen.
- handlePanelChange: maneja las dos notificaciones que le llegan del panel controlador. La primera de ellas sirve para actualizar el atributo “cartasAbiertas” en caso de que se haya borrado desde la tabla una carta que estaba abierta. Y la segunda de ella actualiza el valor “numeroCartas” que será el valor seleccionado por el usuario en el slider del panel de preferencias.
- finJuego: Comprueba las cartas que quedan en el array, y según sean 0, 1 o 2 lanza los mensajes de ganar o perder según corresponda.

### Clase GameView




Destaca el método mouseDown que se activa al hacer click con el ratón. En el método extraemos el punto donde el usuario ha pulsado, para usarlo en los diferentes métodos mencionados anteriormente.

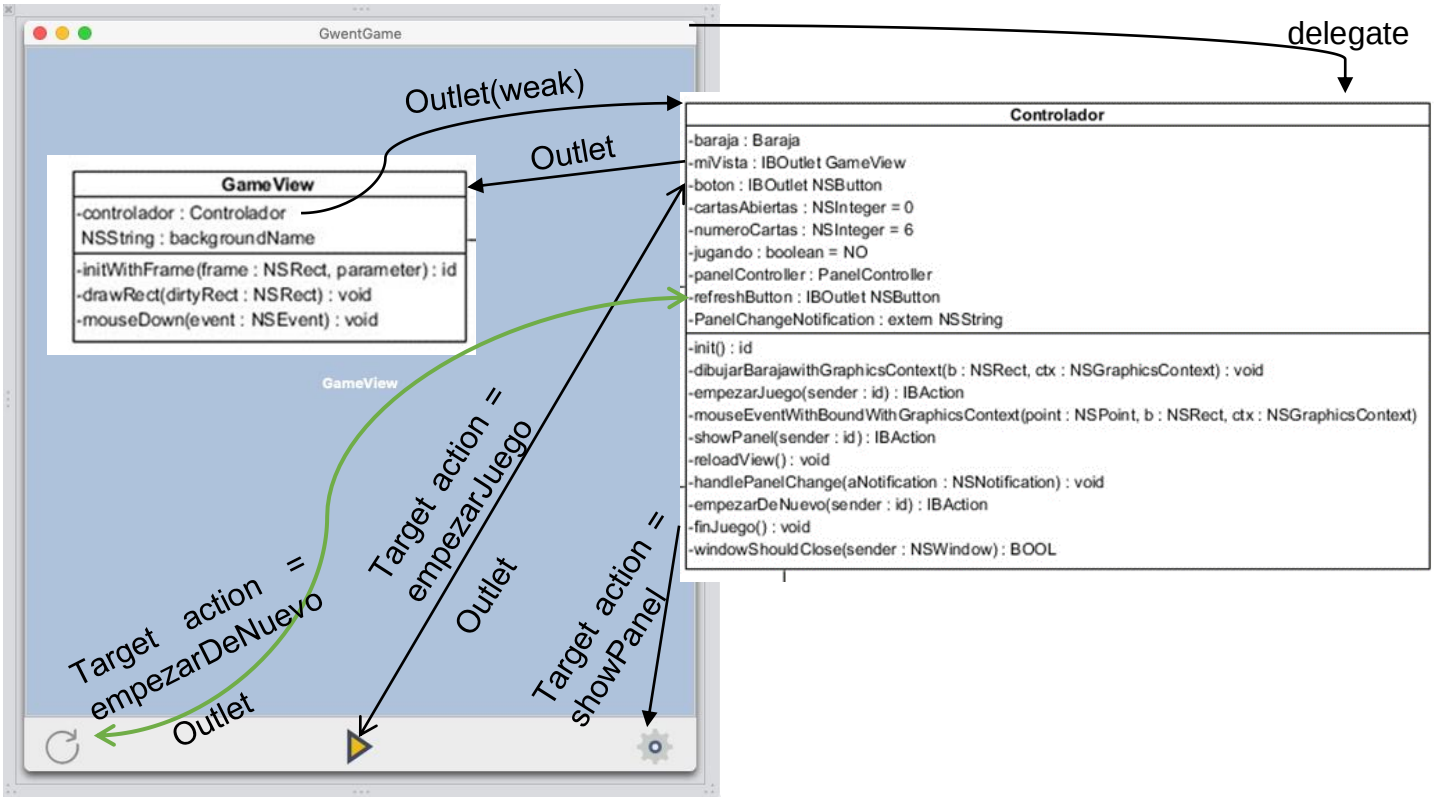
### Clase PanelController

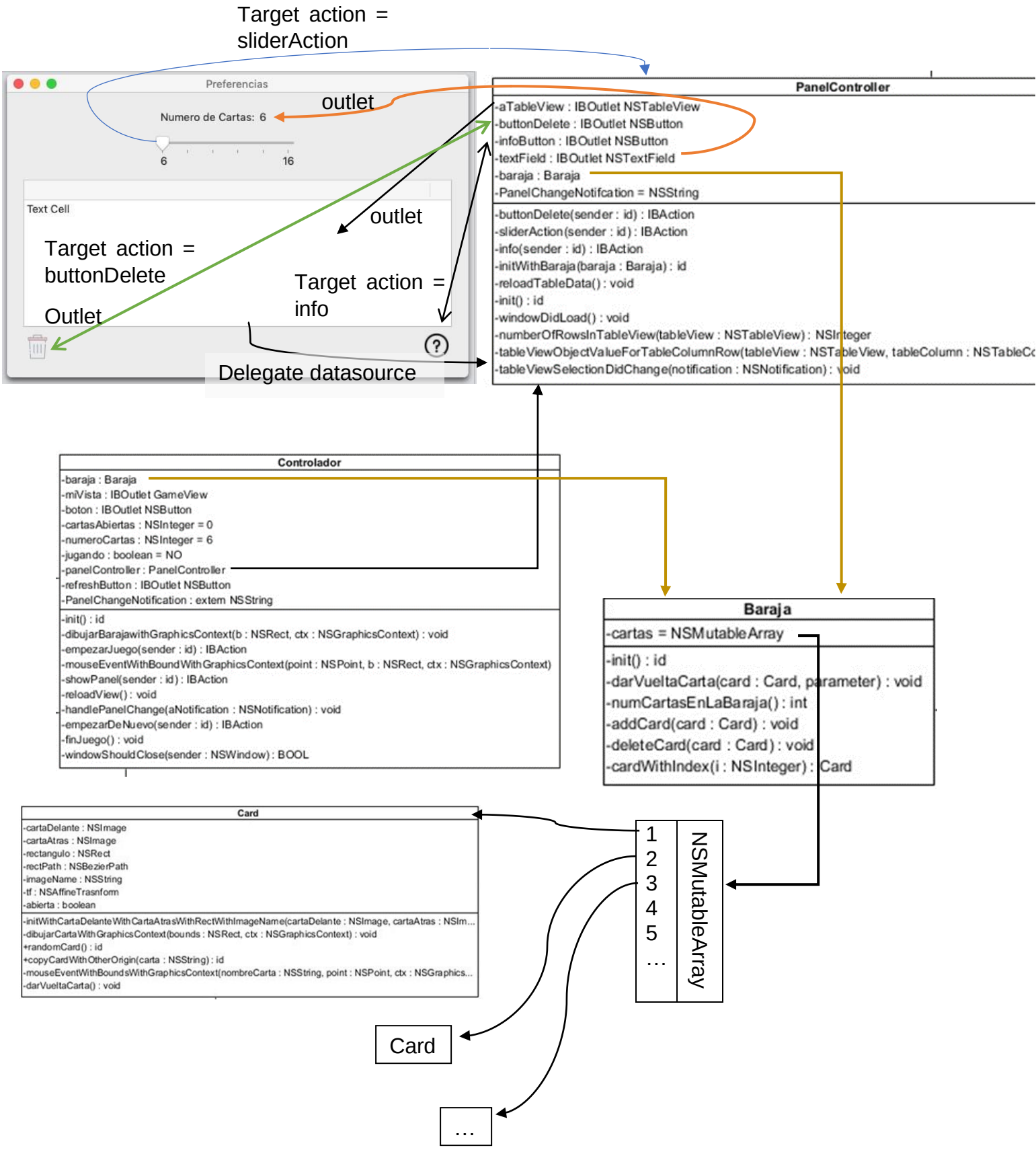



Los métodos más destables son:

- buttonDelete, sliderAction y info son los métodos asociados al funcionamiento del botón borrar, slider y botón de información, respectivamente. En los dos primeros, envían una notificación al controlador, cuyo contenido se ha mencionado anteriormente.
- initWithBaraja: Este método es llamado al crear la instancia del controlador del panel, y permite pasarle la instancia de baraja creada en el controlador al panel, con el fin de conectar el modelo con los dos controladores.

Las conexiones con él xib quedarían de la siguiente manera:




Target action = sliderAction
Preferencias
Numero de Cartas: 6
6
16
Text Cell
outlet
outlet
Target action = buttonDelete
Outlet
Target action = info
Delegate datasource
PanelController
-aTableView : IBOutlet NSTableView
-buttonDelete : IBOutlet NSButton
-infoButton : IBOutlet NSButton
-textField : IBOutlet NSTextField
-baraja : Baraja
-PanelChangeNotifcation = NSString
-buttonDelete(sender : id) : IBAction
-sliderAction(sender : id) : IBAction
-info(sender : id) : IBAction
-initWithBaraja(baraja : Baraja) : id
-reloadTableData() : void
-init() : id
-windowDidLoad() : void
-numberOfRowsInTableView(tableView : NSTableView) : NSInteger
-tableViewObjectValueForTableColumnRow(tableView : NSTableView, tableColumn : NSTableCo
-tableViewSelectionDidChange(notification : NSNotification) : void
Controlador
-baraja : Baraja
-miVista : IBOutlet GameView
-boton : IBOutlet NSButton
-cartasAbiertas : NSInteger = 0
-numeroCartas : NSInteger = 6
-jugando : boolean = NO
-panelController : PanelController
-refreshButton : IBOutlet NSButton
-PanelChangeNotification : extern NSString
-init() : id
-dibujarBarajawithGraphicsContext(b : NSRect, ctx : NSGraphicsContext) : void
-empezarJuego(sender : id) : IBAction
-mouseEventWithBoundWithGraphicsContext(point : NSPoint, b : NSRect, ctx : NSGraphicsContext)
-showPanel(sender : id) : IBAction
-reloadView() : void
-handlePanelChange(aNotification : NSNotification) : void
-empezarDeNuevo(sender : id) : IBAction
-finJuego() : void
-windowShouldClose(sender : NSWindow) : BOOL
Baraja
-cartas = NSMutableArray
-init() : id
-darVueltaCarta(card : Card, parameter) : void
-numCartasEnLaBaraja() : int
-addCard(card : Card) : void
-deleteCard(card : Card) : void
-cardWithIndex(i : NSInteger) : Card
Card
-cartaDelante : NSImage
-cartaAtras : NSImage
-rectangulo : NSRect
-rectPath : NSBezierPath
-imageName : NSString
-tf : NSAffineTrasnform
-abierta : boolean
-initWithCartaDelanteWithCartaAtrasWithRectWithImageName(cartaDelante : NSImage, cartaAtras : NSIm...
-dibujarCartaWithGraphicsContext(bounds : NSRect, ctx : NSGraphicsContext) : void
+randomCard() : id
+copyCardWithOtherOrigin(carta : NSString) : id
-mouseEventWithBoundsWithGraphicsContext(nombreCarta : NSString, point : NSPoint, ctx : NSGraphics...
-darVueltaCarta() : void
1
2
3
4
5
…
NSMutableArray
Card
…

# REFERENCIAS


- Apuntes y ejercicios de clase
- Webs y documentación: (Consultadas el 08/11/2019):
    - NSImage:
        - https://developer.apple.com/documentation/appkit/nsimage?language=objc
        - https://developer.apple.com/documentation/appkit/nsimage/1520015-imagenamed?language=objc
        - https://developer.apple.com/documentation/appkit/nsimage/1519863-drawinrect?language=objc
    - Captura del click del ratón:
        - https://developer.apple.com/documentation/appkit/nsresponder/1524634-mousedown?language=objc
        - https://developer.apple.com/documentation/foundation/nsaffinetransform/1411619-transformpoint?language=occ
    - NSAlert:
        - https://stackoverflow.com/questions/18417432/how-to-show-alert-pop-up-in-in-cocoa
        - https://stackoverflow.com/questions/27877216/nsalert-multiple-buttons
    - Formatear Bool en String:
        - https://stackoverflow.com/questions/2603802/objective-c-formatting-string-for-boolean
    - Alinear en columnas NSString:
        - https://stackoverflow.com/questions/1671531/is-it-possible-to-use-format-strings-to-align-nsstrings-like-numbers-can-be
    - NSNumber:
        - https://developer.apple.com/documentation/foundation/nsnumber?language=objc
        - https://developer.apple.com/documentation/foundation/nsnumber/1407580-initwithint?language=objc
        - https://developer.apple.com/documentation/foundation/nsnumber/1412554-integervalue?language=objc