Ejercicios de vistas

Índice

1. Creación de la vista con Interface Builder

Vamos a crear la pantalla con los detalles de una película utilizando Interface Builder. En la plantilla nos ha creado una vista para los detalles en el fichero `UADetailViewController.xib`. Abriremos dicho fichero y realizaremos los siguientes pasos:

a) Vamos a borrar el contenido actual de la vista (la etiqueta `UILabel` incluida por la plantilla). Configuraremos la vista principal para que simule tener una barra de navegación en la parte superior y una barra de pestañas en la parte inferior (esto puede hacerse desde el inspector de atributos). Con esto podremos diseñar la pantalla sabiendo el espacio con el que contaremos realmente.

b) Creamos una pantalla con:

- `UILabel` para el *Título*.
- `UITextField` para el título. Debe mostrar como texto *placeholder* `Título de la película`, que es lo que se indicará cuando el usuario no haya introducido nada en este campo. La introducción de texto por defecto pondrá en mayúsculas la primera letra de cada palabra, y el teclado tendrá la tecla *Done* para indicar que hemos terminado de introducir el texto. Todos estos elementos deben especificarse mediante el editor de atributos.
- `UILabel` para el *Director*.
- `UITextField` para el director. Tendrá los mismos atributos que el del título, y como texto de *placeholder* `Director de la película`.
- `UILabel` para *Calificación de edad*.
- `UISegmentedControl` para seleccionar la calificación de edad de la película. Tendrá 4 segmentos: TP, NR7, NR13 y NR18.
- `UILabel` para la *Valoración*
- `UISlide` para especificar la valoración.
- `UILabel` para indicar el valor especificado en la barra de valoración de forma numérica. Estará junto a dicha barra.
- `UIButton` para *Guardar* los datos y para *Ver cartel* de la película.

El aspecto de esta pantalla deberá ser el siguiente:

Vista de detalles de la película

c) Probar que la pantalla se ve correctamente al ejecutar la aplicación. Al editar el texto del título o del director, ¿puedes cerrar el teclado? Solucionaremos este problema en los próximos ejercicios.

2. Conexión de componentes mediante outlets

Vamos a conectar los elementos de la interfaz creados en el ejercicio anterior con nuestro código Objective-C. Se pide:

a) Crea una serie de *outlets* en `UADetailViewController` para:

- Los `UITextField` del título y el director.
- El `UISegmentedControl` de la calificación de edad.
- El `UISlider` de la puntuación y el `UILabel` que hay junto a él para indicar el valor numérico.

b) Conecta en Interface Builder las vistas anteriores con los *outlets* que acabamos de crear.

c) Haz que al pasar a la pantalla de detalles de la película se rellenen los campos con los valores almacenados para la película. Para ello básate en el método `setDetailItem` de la

plantilla, y modificalo para que lo que se proporcione sea una película (puedes cambiar el nombre al método y a la propiedad). En `configureView` rellenaremos los elementos de la interfaz a partir de los datos de la película. En `UIMasterViewController` localiza el método `tableView:didSelectRowAtIndexPath:` y haz que se proporcione al objeto `UIDetailViewController` los datos de la película de la fila seleccionada antes de mostrar dicha pantalla.

3. Controles y acciones

Vamos a tratar los diferentes eventos de los controles de la pantalla mediante acciones. Se pide:

a) En los campos de texto hemos visto que el teclado no se cierra cuando terminamos de editar el texto (cuando pulsamos el botón *Done*), lo cual es un problema. Para cerrar el teclado asociado a un campo de texto debemos hacer que el campo de texto deje de ser *First responder*, mediante el siguiente método:

```
[campo resignFirstResponder];
```

Esto deberemos hacerlo cuando se termine de editar el texto. Esto corresponde al evento *Did end on exit* del `UITextField`. Añade la acción correspondiente en `UADetailViewController`, y conéctala con el correspondiente evento en los dos campos de texto (título y director). Comprueba que ahora el teclado se cierra correctamente.

Alternativa

También podemos hacer esto creando un delegado de `UITextField` para el campo de texto (por ejemplo adoptando el protocolo `UITextFieldDelegate` en la misma clase `UADetailViewController`). El delegado deberá implementar el método `textField:shouldReturn:`, y en él devolveremos `YES` para indicar que se debe cerrar el teclado.

b) Vamos a añadir un evento al botón *Guardar* para que almacene en el objeto los nuevos valores que hayamos introducido en los campos de texto. Crea una acción para hacer esto en `UADetailViewController` y conéctala con el evento apropiado del botón. Comprueba ahora que los datos quedan almacenados correctamente cuando pasamos de una película a otra.

c) Si guardamos los datos y volvemos a la lista, veremos que el título no cambia, aunque lo hayamos modificado en la ficha con los datos de la película. Para conseguir que cambie deberemos recargar la tabla que muestra los datos. Una forma sencilla de hacer esto es recargar la tabla cada vez que se muestra la pantalla, implementando el siguiente método en `UAMasterViewController`:

```
- (void)viewWillAppear:(BOOL)animated
{
    [super viewWillAppear:animated];
    [self.tableView reloadData];
```

```
}
```

4. Imágenes

Vamos a añadir imágenes a la aplicación. Para ello puedes utilizar las proporcionadas en las plantillas o descargarlas de Internet. Se pide:

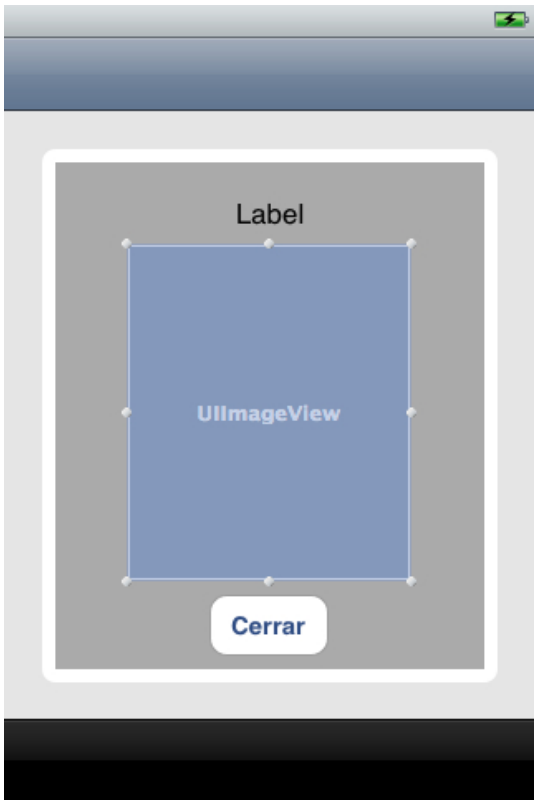
a) En primer lugar añadiremos una propiedad de tipo `UIImage` a las películas. Al crear la lista de películas, asignaremos a cada película la imagen que corresponda.

b) Vamos ahora a mostrar en la tabla la imagen asociada a cada película. Utilizaremos para ello la propiedad `imageView.image` del objeto `UITableViewCell` al configurar cada celda.

5. Carga de componentes del NIB

Ahora vamos a dotar de funcionalidad el botón *Ver cartel* que tenemos en la pantalla con los datos de las películas. Se pide:

a) Crea la vista con el cartel de la película en un NIB independiente, con el siguiente aspecto:



Vista del cartel de la película

De este NIB sólo nos interesa el recuadro gris, que contiene como subvistas una etiqueta con el título de la película (`UILabel`), una imagen con su cartel (`UIImageView`), y un botón para cerrar la ventana. Por lo tanto, será este recuadro el que tendremos que conectar con el código de la aplicación mediante un *outlet*.

b) Haz que al pulsar el botón *Ver cartel* se cargue la vista anterior del NIB y se muestre en pantalla (añadiéndola como subvista). Deberá quedar por encima de los datos de la película. Para rellenar los datos del título de la película y de su imagen puedes asignar *tags* a cada uno de estos componentes, y en el código hacer un `viewWithTag` desde la vista que los contiene.

c) Haz que al pulsar el botón *Cerrar* de la subvista con la imagen, ésta se elimine de su supervista.

d) Cambia el atributo necesario de `UIImageView` para que la imagen ocupe el máximo espacio posible dentro del área del componente, sin deformarse ni recortarse.

6. Almacenamiento (*)

Sobre el ejercicio anterior, vamos a hacer que cada vez que la aplicación pase a segundo

plano se almacene la lista de películas actual en disco para así evitar perder los cambios realizados si la aplicación se cerrase. Se pide:

a) ¿Podemos guardar la colección de películas directamente en disco mediante el método que tenemos en las colecciones para ello? ¿Por qué? En caso negativo, ¿cómo podríamos guardarlas?

b) Implementa en la clase `UAPelicula` el protocolo necesario para almacenar las películas. Para simplificar, por el momento vamos a guardar únicamente el título y el director (el resto de datos los guardaremos más adelante).

c) Implementa en `UAMasterViewController` un método `save` que se encargue de guardar la lista de películas utilizando los métodos de serialización definidos en el apartado anterior (podemos guardar toda la colección mediante una única operación de guardado, no hace falta guardar cada película individualmente).

d) Haz que cuando la aplicación pase a segundo plano se ejecute el método `save` anterior. Esto deberemos implementarlo en el método adecuado de `UAAppDelegate`. Para poder hacer esto, en el *app delegate* deberemos guardarnos una referencia a `UAMasterViewController` (por ejemplo podemos guardarla como propiedad de la clase). Comprueba que el guardado se realiza correctamente cuando pulsamos el botón HOME del teléfono, por ejemplo escribiendo un *log* que nos indique el resultado de la operación de guardado (`YES` o `NO`).

e) En el inicializador de la clase `UAMasterViewController` haz que intente leer la colección del fichero en disco. Si el fichero existe, cogerá la colección almacenada en él, si no, inicializará la colección por defecto como hacíamos anteriormente. Comprueba que si se modifica el título o el director de alguna película la próxima vez que carguemos la aplicación esos cambios se conservan.

f) Vamos a guardar ahora también el resto de campos de las películas. El campo más complejo es la imagen, ya que no es un tipo de datos que podamos guardar directamente. Para solucionar esto, podemos guardar la imagen como un objeto de tipo `NSData` (datos binarios) que contenga su representación en PNG. Podemos obtener dicha representación con:

```
NSData *datos = UIImagePNGRepresentation(self.cartel);
```

Para crear de nuevo una imagen a partir de los datos binarios de su representación en PNG, podemos utilizar el siguiente inicializador:

```
UIImage *imagen = [UIImage imageWithData: datos];
```

7. Creación programática de componentes (*)

Vamos a crear componentes de forma programática. Añadiremos a cada elemento de la tabla (en `UIMasterViewController`) un interruptor que nos permitirá cambiar el color de

la fila. Se pide:

a) En el método `tableView:cellForRowAtIndexPath:` de `UAMasterViewController`, tras la instanciación de las celdas, crea también un nuevo objeto de tipo `UISwitch` utilizando el inicializador vacío, y asígnaselo a la propiedad `accesoryView` de la celda.

b) En el mismo lugar, añade una acción de forma programática al `UISwitch` para que al cambiar de valor cambie el color de fondo y de texto de la celda. Cuando el interruptor esté activado el fondo de la celda será gris claro, y el texto rojo:

Lista de películas con interruptor