



Desarrollo de Aplicaciones iOS

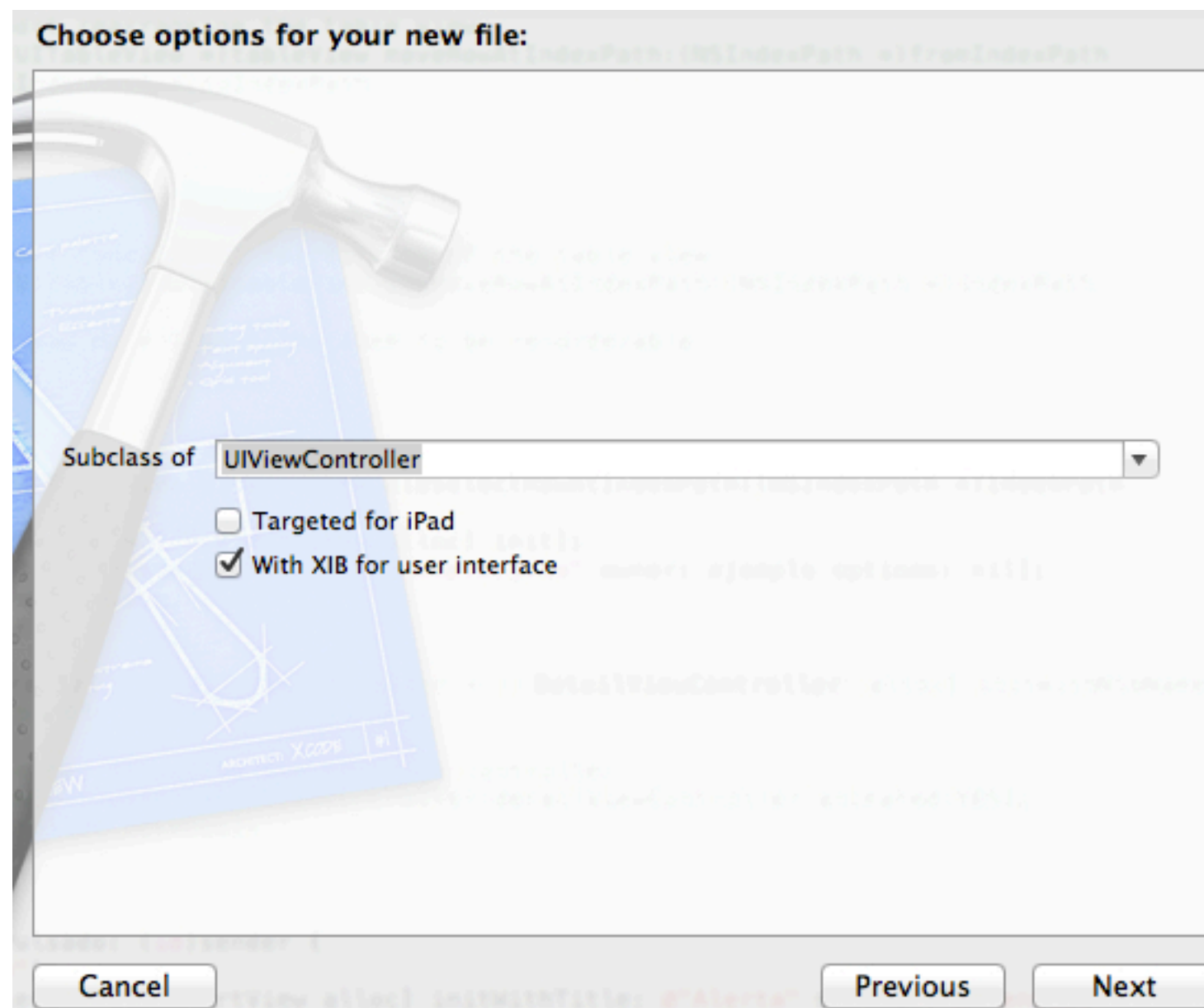
Sesión 5: Controladores

Puntos a tratar

- Creación de controladores
- Gestión de tablas
- Modo de edición
- Controlador de búsqueda

Controladores

- Clases que derivan de `UIViewController`
 - Se crean con *File > New File ... > iOS > Cocoa Touch > UIViewController subclass*



Gestión de la vista

- Cargan y gestionan la vista
 - Hacen de *File's Owner* de las vistas

```
EjemploViewController *ejemploViewController =  
    [[EjemploViewController alloc]  
        initWithNibName:@"EjemploViewController" bundle:nil];
```

- La vista principal se asigna a la propiedad `view` del controlador
- Podemos mostrar la vista en pantalla asignando el controlador a la propiedad `rootViewController` de la ventana clave

```
self.window.rootViewController = ejemploViewController;
```

- En versiones previas a iOS 4 esta propiedad no existe, se debe añadir la vista del controlador como subvista de la ventana

Ciclo de vida

- Inicialización de la vista

```
- (void)viewDidLoad {  
    descripcionView.text = asignatura.descripcion;  
    descripcionView.editable = NO;  
}
```

- Se ejecuta tras cargar la vista en `loadView`
 Sólo sobrescribir `loadView` para crear vista de forma programática
- Debemos deshacer lo anterior en `viewDidUnload`
- Eventos del ciclo de vida

```
- viewWillAppear:  
- viewDidAppear:  
- viewWillDisappear:  
- viewDidDisappear:
```

Control de la orientación

- Definimos el siguiente método

```
- (BOOL)shouldAutorotateToInterfaceOrientation:  
    (UIInterfaceOrientation)interfaceOrientation
```

- Recibimos como parámetro la orientación solicitada
- Devolvemos YES si aceptamos el giro a dicha orientación

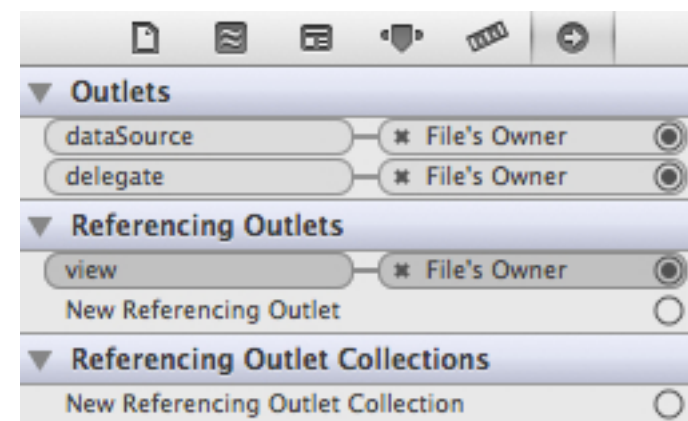
- Eventos de la rotación

```
- (void)willRotateToInterfaceOrientation:duration:  
- (void)willAnimateRotationToInterfaceOrientation:duration:  
- (void) didRotateFromInterfaceOrientation:
```

- La orientación inicial se configura en Info.plist

Tablas

- Para gestionar las tablas (UITableView) utilizamos un controlador especial: UITableViewController
- Además de heredar de UIViewController, implementa
 - UITableViewDataSource
Se encarga de dar contenido a las filas de la tabla
 - UITableViewDelegate
Trata los eventos de la tabla
 - Seleccionar una fila
 - Borrar una fila
 - Reordenar filas





Fuente de datos de las tablas

Devuelve el número de filas

Busca una celda en el *pool*

Instancia una nueva celda si no hay ninguna disponible

Configura y devuelve la celda en la fila `indexPath.row`

```
- (NSInteger) tableView:(UITableView *)tabla
numberOfRowsInSection: (NSInteger)section {
    return [asignaturas count];
}

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    static NSString *CellIdentifier = @"Cell";

    UITableViewCell *cell =
        [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
    if (cell == nil) {
        cell = [[[UITableViewCell alloc]
            initWithStyle:UITableViewCellStyleDefault
            reuseIdentifier:CellIdentifier] autorelease];
    }

    // Configuramos la celda
    cell.textLabel.text =
        [[asignaturas objectAtIndex: indexPath.row] nombre];

    return cell;
}
```


Eventos de las tablas

- Lo más común es realizar alguna acción al seleccionar una celda
 - Por ejemplo mostrar los datos del *item* seleccionado

```
- (void)tableView:(UITableView *)tableView
    didSelectRowAtIndexPath:(NSIndexPath *)indexPath {

    UASignatura *asignatura =
        [asignaturas objectAtIndex: indexPath.row];
    UIAlertView *alert =
        [[UIAlertView alloc] initWithTitle:asignatura.nombre
                                     message:asignatura.descripcion
                                     delegate: nil
                                     cancelButtonTitle: @"Cerrar"
                                     otherButtonTitles: nil];

    [alert show];
    [alert release];
}
```

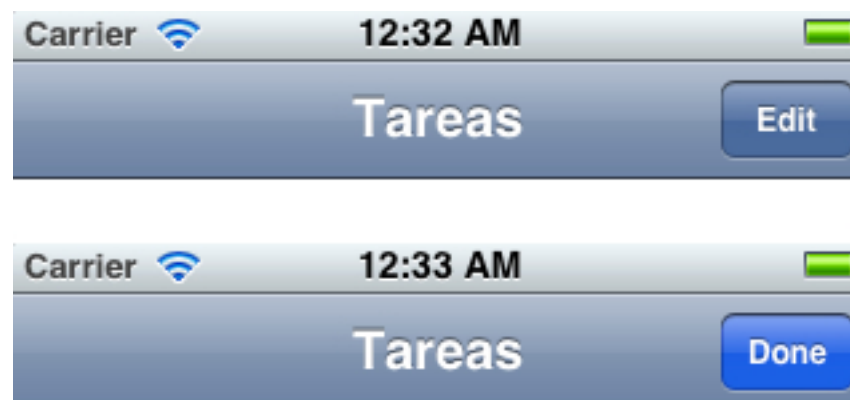
Modo de edición

- Las tablas pueden entrar en modo de edición
- Tenemos un botón del sistema para pasar a este modo

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    // Uncomment the following line to display an Edit button in the
    // navigation bar for this view controller.
    self.navigationItem.rightBarButtonItem = self.editButtonItem;

    ...
}
```



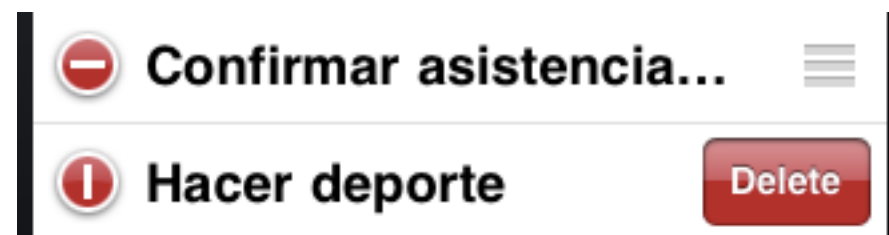
Borrado de filas

- En modo de edición por defecto podemos borrar filas
- Debemos implementar `commitEditingStyle` en el delegado

```
- (void)tableView:(UITableView *)tableView
    commitEditingStyle:(UITableViewCellEditingStyle)editingStyle
    forRowAtIndexPath:(NSIndexPath *)indexPath {
    if (editingStyle == UITableViewCellEditingStyleDelete) {
        [self.arrayTareas removeObjectAtIndex:indexPath.row];
        [tableView deleteRowsAtIndexPaths: [NSArray arrayWithObject:indexPath]
                        withRowAnimation:UITableViewRowAnimationFade];
    } else if (editingStyle == UITableViewCellEditingStyleInsert) { }
}
```

Elimina el elemento internamente

Borra la fila mediante una animación



- Si implementamos este método, también se puede borrar mediante un gesto de barrido lateral (*swipe to delete*)

Reordenación de filas

- Podemos permitir al usuario reordenar las filas en modo edición implementando los siguientes métodos del delegado

```
// Override to support rearranging the table view.
- (void)tableView:(UITableView *)tableView
    moveRowAtIndexPath:(NSIndexPath *)fromIndexPath
    toIndexPath:(NSIndexPath *)toIndexPath
{
    NSString *item = [self.arrayTareas objectAtIndex:fromIndexPath.row];
    [self.arrayTareas removeObjectAtIndex:fromIndexPath.row];
    [self.arrayTareas insertObject:item atIndex:toIndexPath.row];
}

// Override to support conditional rearranging of the table view.
- (BOOL)tableView:(UITableView *)tableView
    canMoveRowAtIndexPath:(NSIndexPath *)indexPath
{
    // Return NO if you do not want the item to be re-orderable.
    return YES;
}
```

Cambiamos el orden internamente

Indicamos qué filas son reordenables

Inserción

- La forma más sencilla de insertar tareas es utilizar un botón estándar en la barra de navegación



- Al pulsar el botón, hacemos la inserción

```
- (void)addTarea:(id)sender {  
    NSIndexPath *indexPath = [NSIndexPath indexPathForRow:[self.arrayTareas count]  
                                                                inSection:0];  
    [self.arrayTareas addObject:@"Tarea nueva"];  
    [self.tableView insertRowsAtIndexPaths: [NSArray arrayWithObject:indexPath]  
                             withRowAnimation:UITableViewRowAnimationTop];  
}
```

Fila de inserción

- También podemos añadir una fila de inserción a la propia tabla

	Ir al banco	
	Llamar a Juan	
	NUEVA	

- Este caso es más complejo que el anterior, deberemos
 - Devolver una fila adicional en el *data source* cuando estemos en modo de edición
 - Insertar o eliminar la fila de inserción cuando entremos o salgamos de dicho modo
 - Evitar que la fila pueda ser reordenada

Fila de inserción en el *data source*

```
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section {
    if(self.editing) {
        return [self.arrayTareas count]+1;
    } else {
        return [self.arrayTareas count];
    }
}

- (UITableViewCell *)tableView:(UITableView *)tableView
    cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    static NSString *CellIdentifier = @"Cell";

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
    if (cell == nil) {
        cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault
            reuseIdentifier:CellIdentifier];
    }

    if(indexPath.row == [self.arrayTareas count]) {
        cell.textLabel.text = @"NUEVA";
    } else {
        cell.textLabel.text = [self.arrayTareas objectAtIndex:indexPath.row];
    }
    return cell;
}
```

Modo edición



Modo edición, última fila

Transición al modo de edición

- Podemos sobrescribir el método `setEditing:animated:` del controlador para realizar las acciones oportunas cuando cambiemos entre modo de edición y modo normal


```
- (void)setEditing:(BOOL)editing animated:(BOOL)animated {  
    [super setEditing:editing animated:animated];  
  
    NSArray *rows = [NSArray arrayWithObject:  
        [NSIndexPath indexPathForRow:[self.arrayTareas count]  
        inSection:0]];  
  
    if(editing) {  
        [self.tableView insertRowsAtIndexPaths:rows  
            withRowAnimation:UITableViewRowAnimationTop];  
    } else {  
        [self.tableView deleteRowsAtIndexPaths:rows  
            withRowAnimation:UITableViewRowAnimationTop];  
    }  
}
```


Estilo de edición de filas

- Podemos especificar el tipo de edición para cada fila
- La fila de edición deberá mostrar  en lugar de 
- Definimos el siguiente método del *data source*

```
- (UITableViewCellEditingStyle)tableView:(UITableView *)tableView
    editingStyleForRowAtIndexPath:(NSIndexPath *)indexPath {
    if(indexPath.row < [self.arrayTareas count]) {
        return UITableViewCellEditingStyleDelete;
    } else {
        return UITableViewCellEditingStyleInsert;
    }
}
```

Ejecución de la inserción

- Debemos implementarlo en el método `commitEditingStyle` del delegado
- Este método se invocará al pulsar sobre 

```
- (void)tableView:(UITableView *)tableView
    commitEditingStyle:(UITableViewCellEditingStyle)editingStyle
    forRowAtIndexPath:(NSIndexPath *)indexPath
{
    if (editingStyle == UITableViewCellEditingStyleDelete) {
        [self.arrayTareas removeObjectAtIndex:indexPath.row];
        [tableView deleteRowsAtIndexPaths: [NSArray arrayWithObject:indexPath]
                        withRowAnimation:UITableViewRowAnimationFade];
    } else if (editingStyle == UITableViewCellEditingStyleInsert) {
        [self.arrayTareas addObject:@"Tarea nueva"];
        [tableView insertRowsAtIndexPaths:[NSArray arrayWithObject:indexPath]
                        withRowAnimation:UITableViewRowAnimationTop];
    }
}
```

Control de la reordenación

- Evitar que la fila de inserción sea reordenable

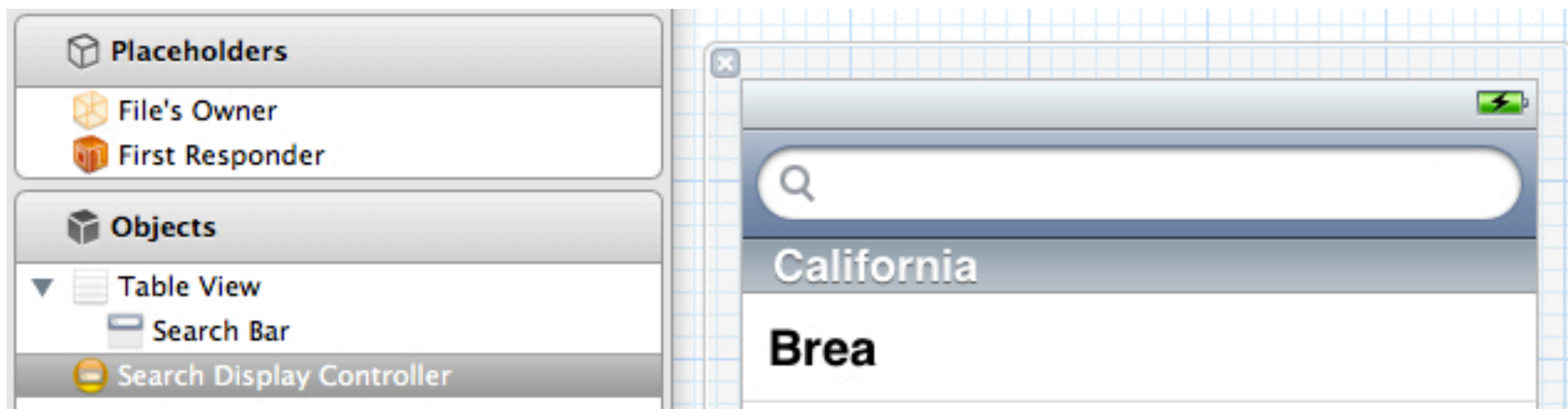
```
- (BOOL)tableView:(UITableView *)tableView
    canMoveRowAtIndexPath:(NSIndexPath *)indexPath
{
    // Return NO if you do not want the item to be re-orderable.
    return indexPath.row < [self.arrayTareas count];
}
```

- Impedir que una fila se pueda mover tras la fila de inserción

```
- (NSIndexPath *)tableView:(UITableView *)tableView
    targetIndexPathForMoveFromRowAtIndexPath:(NSIndexPath *)sourceIndexPath
    toProposedIndexPath:(NSIndexPath *)proposedDestinationIndexPath {
    if(proposedDestinationIndexPath.row >= [self.arrayTareas count]) {
        return [NSIndexPath indexPathForRow:([self.peliculas count]-1) inSection:0];
    } else {
        return proposedDestinationIndexPath;
    }
}
```

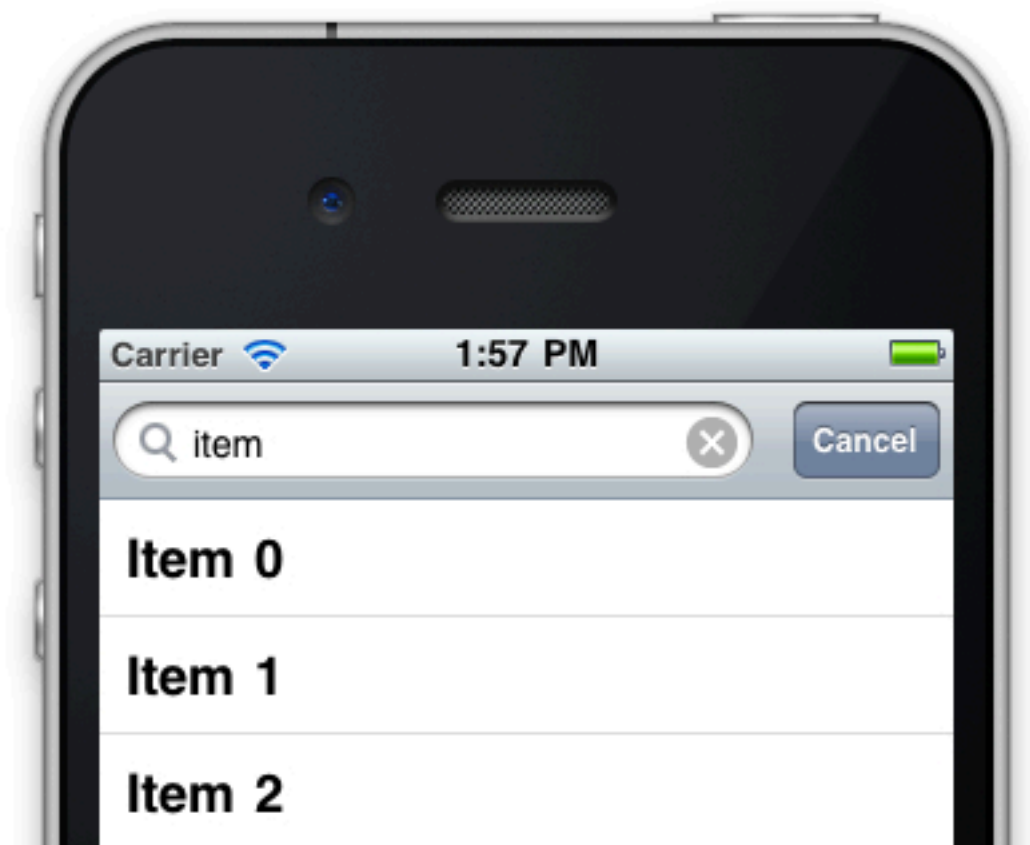
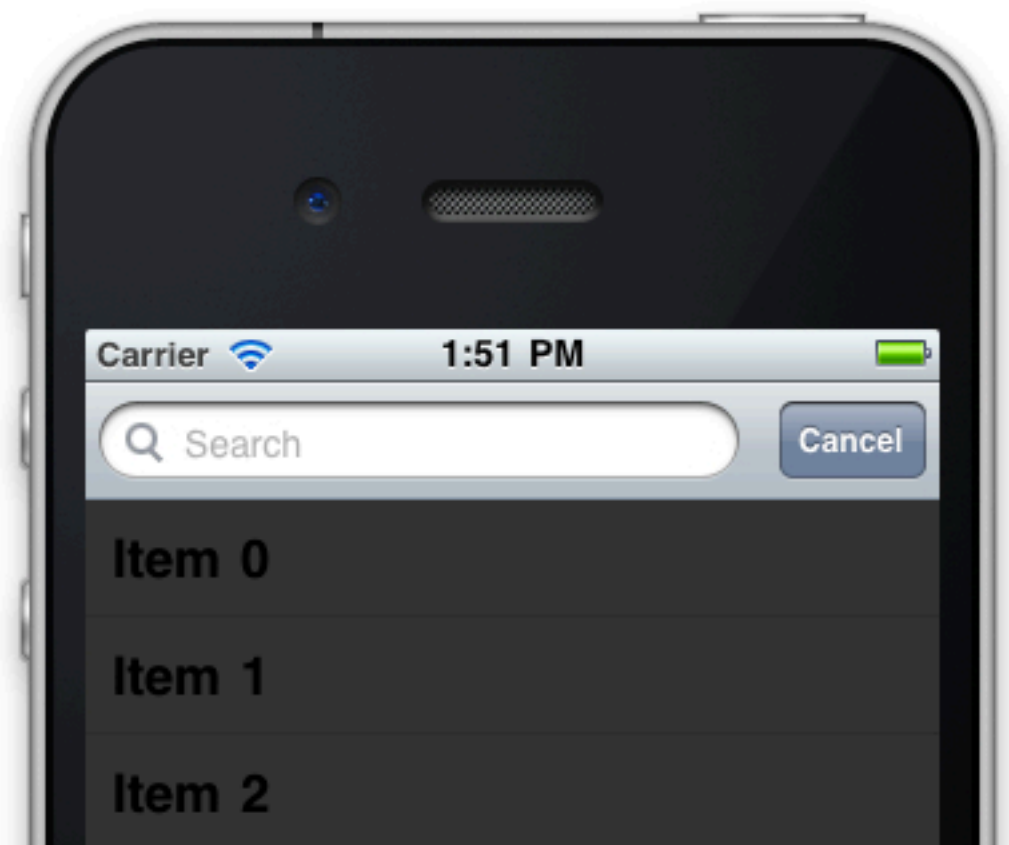
Controlador de búsqueda

- Permite realizar búsquedas en una pantalla
 - Normalmente se utilizan en tablas (UITableViewController)
 - Filtran los contenidos de la tabla según la cadena de búsqueda
- Necesita
 - Barra de búsqueda (UISearchBar)
 - Controlador de contenido (normalmente el de la misma tabla)
Se utilizará para obtener los resultados de la búsqueda



Funcionamiento de la búsqueda

- Al pulsar la barra la tabla queda en sombra
- Al introducir texto se muestran los resultados de la búsqueda
- ¡CUIDADO! La tabla de resultados la crea el controlador
 - Propiedad `self.searchDisplayController.searchResultsTableView`



Realizar la búsqueda

- Definimos un método de UISearchDisplayDelegate
 - Filtramos los resultados según la cadena de búsqueda

```
- (BOOL)searchDisplayController:
    (UISearchDisplayController *)controller
    shouldReloadTableForSearchString:(NSString *)searchString {

    self.itemsFiltrados = [self filtrarItems: _items
                                busqueda: searchString];

    return YES;
}
```

- Se ejecuta cada vez que modificamos la cadena
- Si la operación es lenta, dejamos una operación en segundo plano y devolvemos NO

Mostrar resultados

- Comprobamos si la tabla es la original o la de resultados

```
- (NSInteger)tableView:(UITableView *)tableView
    numberOfRowsInSection:(NSInteger)section {
    if(tableView == self.searchDisplayController.searchResultsTableView) {
        return [_itemsFiltrados count];
    } else {
        return [_items count];
    }
}

- (UITableViewCell *)tableView:(UITableView *)tableView
    cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    // Obtener celda del pool
    ...

    if(tableView == self.searchDisplayController.searchResultsTableView) {
        cell.textLabel.text = [_itemsFiltrados objectAtIndex: indexPath.row];
    } else {
        cell.textLabel.text = [_items objectAtIndex: indexPath.row];
    }

    return cell;
}
```



¿Preguntas...?