



POLITÉCNICA

ETSIT  
UPM

*dit*  
UPM

# Desarrollo de Apps para iOS Collection Views

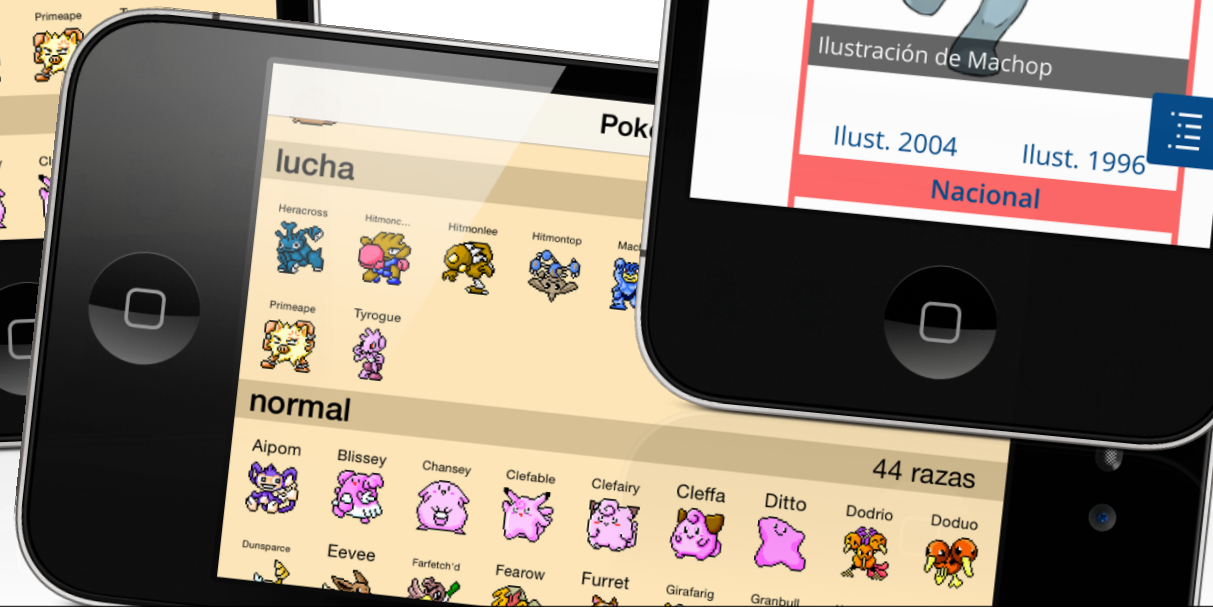
IWEB,LSWC 2014-2015

Santiago Pavón

ver: 2014.12.03

# Características de las CollectionViews

- Muestran los datos usando celdas dispuestas con diferentes layouts.
  - Flow Layout:
    - Coloca las celdas en una parrilla completando líneas horizontal o verticalmente.
  - Layouts personalizados
- Las celdas se puede agrupar en secciones.
  - Las secciones pueden tener un vistas complementarias
    - haciendo el papel de cabecera y pie de las secciones.
- El contenido de las celdas es siempre personalizado.
- Realiza un uso eficiente de celdas y vistas complementarias.
  - Reutilizando las no visibles.



# Clases y Protocolos

- Algunas clases:
  - **UICollectionView**: la vista que contiene las celdas.
  - **UICollectionViewCell**: la celda con los datos.
  - **UICollectionViewController**: un UIViewController conteniendo una UICollectionView ocupando todo el espacio.
  - **UICollectionViewFlowLayout**: clase del gestor de layout Flow.
  - **UICollectionViewLayoutAttributes**: clase con los atributos que aplican a las celdas o vistas complementarias.
- Algunos protocolos:
  - **UICollectionViewDataSource**: define el protocolo de obtención de datos del objeto dataSource.
  - **UICollectionViewDelegate**: define el protocolo para comunicarse con el delegado informando de sucesos.

# UICollectionView

- Propiedades:

**dataSource**  
**delegate**

**backgroundView**

**collectionViewLayout**

**allowsSelection**  
**allowsMultipleSelection**

- Algunos métodos:

```
init(frame:, collectionViewLayout:)
registerClass(:, forCellWithReuseIdentifier:)
registerNib(:, forCellWithReuseIdentifier:)
registerClass(:, forSupplementaryViewOfKind:, withReuseIdentifier:)
registerNib(:, forSupplementaryViewOfKind:, withReuseIdentifier:)
enqueueReusableCellWithReuseIdentifier(:, forIndexPath:)
enqueueReusableSupplementaryViewOfKind(:, withReuseIdentifier:, forIndexPath:)
setCollectionViewLayout(:, animated:, completion:)
reloadData()
reloadSections(:)
reloadItemsAtIndexPaths(:)
numberOfSections()
numberOfItemsInSection(:)
visibleCells()
insertItemsAtIndexPaths(:)
moveItemAtIndexPath(:, toIndexPath:)
deleteItemsAtIndexPaths(:)
indexPathsForSelectedItems()
selectItemAtIndexPath(:, animated:, scrollPosition(:))
deselectItemAtIndexPath(:, animated:)
indexPathForItemAtPoint(:)
indexPathForCell(:)
cellForItemAtIndexPath(:)
scrollToItemAtIndexPath(:, atScrollPosition:, animated:)
etc . . .
```

# UICollectionViewCell

- Propiedades:

**collectionView**

**collectionView**

**collectionView**

**collectionView**

**collectionView**

# UICollectionViewDataSource

```
func collectionView(_ collectionView: UICollectionView,  
    numberOfItemsInSection section: Int)  
    -> Int
```

```
optional func numberOfSectionsInCollectionView(  
    _ collectionView: UICollectionView)  
    -> Int
```

```
func collectionView(_ collectionView: UICollectionView,  
    cellForItemAtIndexPath indexPath: NSIndexPath)  
    -> UICollectionViewCell
```

```
optional func collectionView(_ collectionView: UICollectionView,  
    viewForSupplementaryElementOfKind kind: String,  
    atIndexPath indexPath: NSIndexPath)  
    -> UICollectionViewReusableView
```



# UICollectionViewDelegate

- Selección de celdas:

```
collectionView( : , shouldSelectItemAtIndexPath: )  
collectionView( : , didSelectItemAtIndexPath: )  
collectionView( : , shouldDeselectItemAtIndexPath: )  
collectionView( : , didDeselectItemAtIndexPath: )
```

- Destacar celdas:

```
collectionView( : , shouldHighlightItemAtIndexPath: )  
collectionView( : , didHighlightItemAtIndexPath: )  
collectionView( : , didUnhighlightItemAtIndexPath: )
```

- Eliminar vistas:

```
collectionView( : , didEndDisplayingCell: , forItemAtIndexPath: )  
collectionView( : , didEndDisplayingSupplementaryView: , forElementOfKind: , atIndexPath: )
```

- Transición entre layouts:

```
collectionView( : , transitionLayoutForOldLayout: , newLayout: )
```

- Gestión de acciones

```
collectionView( : , shouldShowMenuForItemAtIndexPath: )  
collectionView( : , canPerformAction: , forItemAtIndexPath: , withSender: )  
collectionView( : , performAction: , forItemAtIndexPath: , withSender: )
```

# UICollectionViewController

- Inicializador:

`init(collectionViewLayout:)`

- Propiedades:

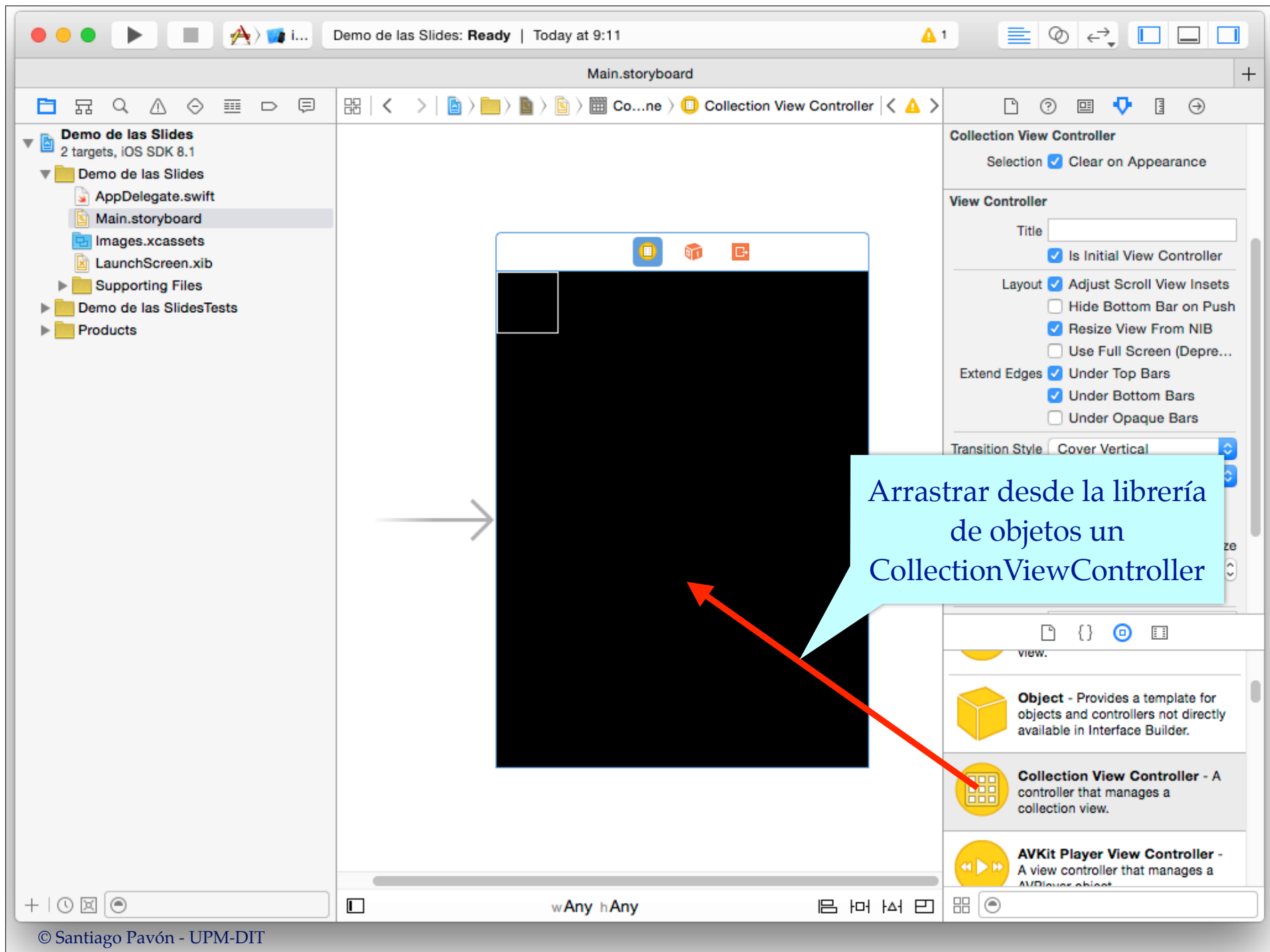
`collectionView`

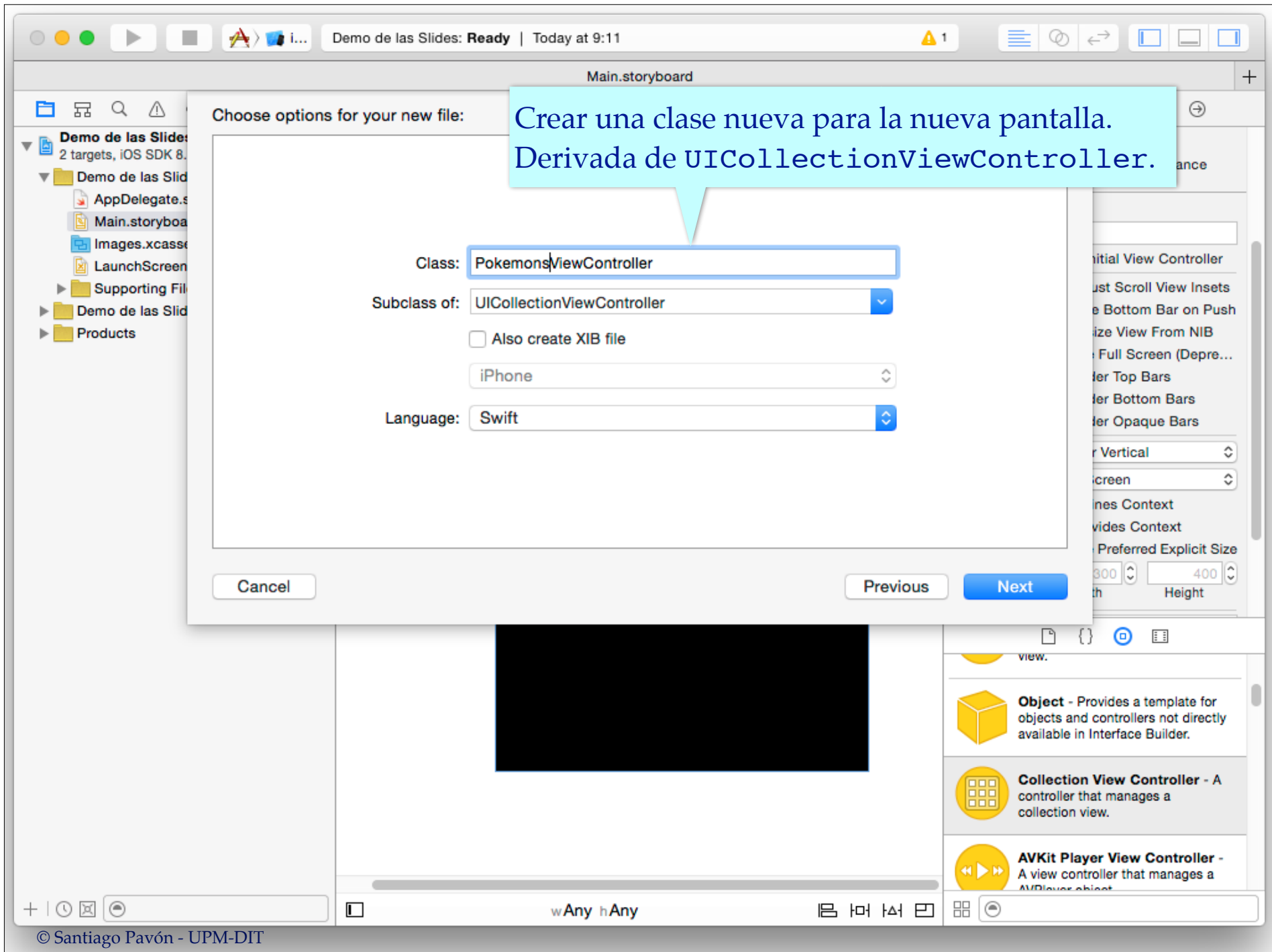
`collectionViewLayout`

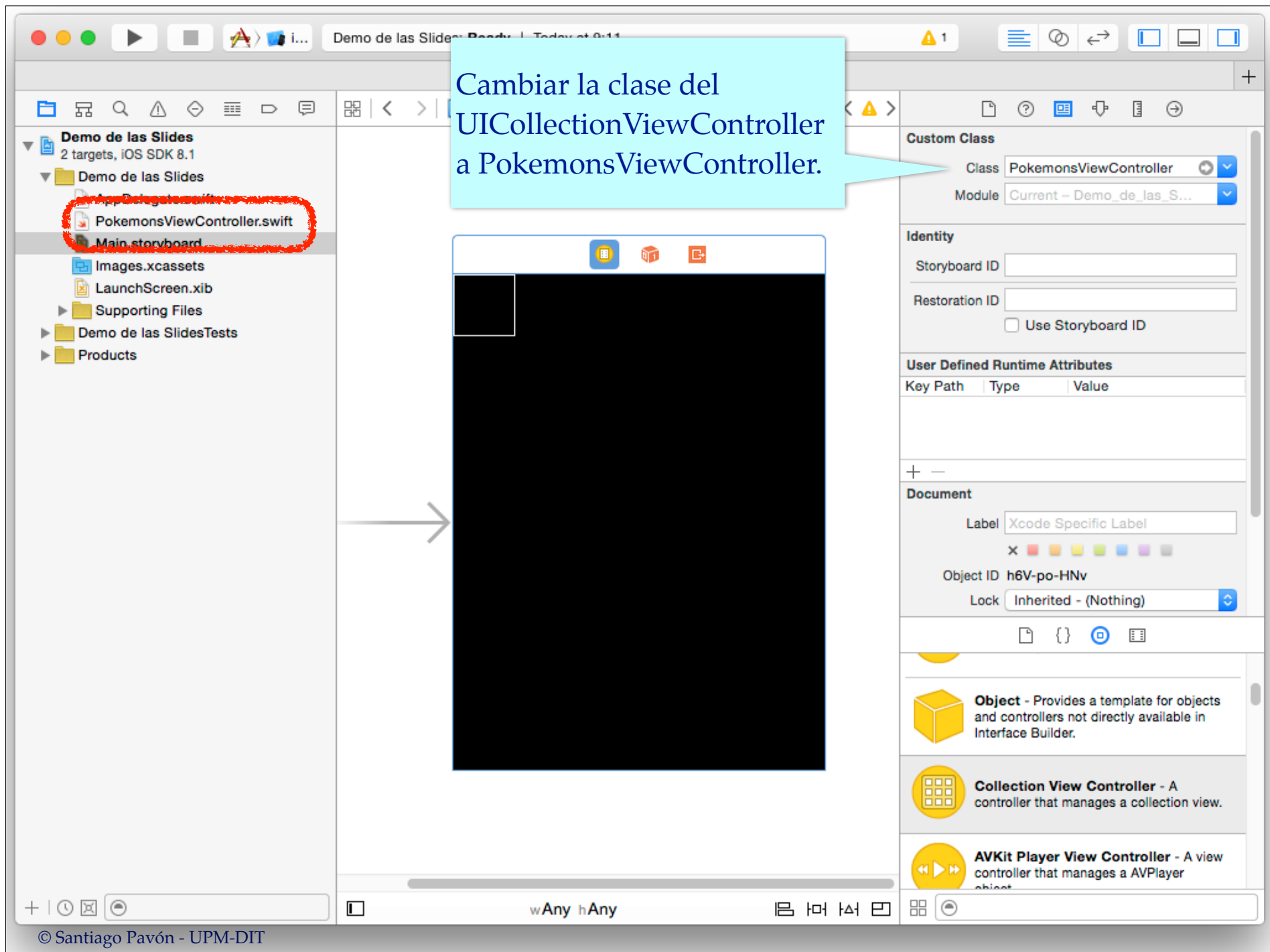
`clearsSelectionOnViewWillAppear`

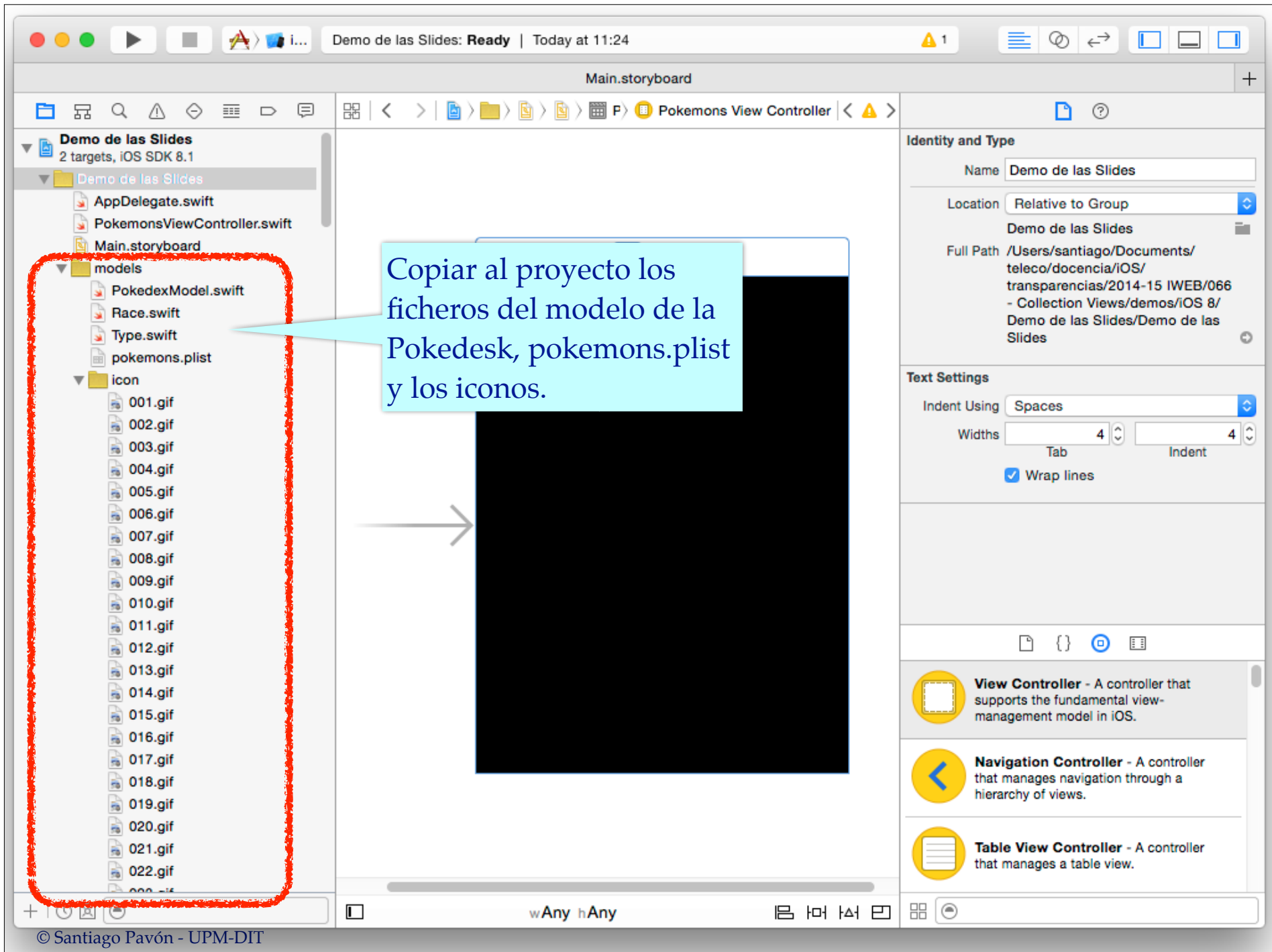
`useLayoutToLayoutNavigationTransitions`

# Demo









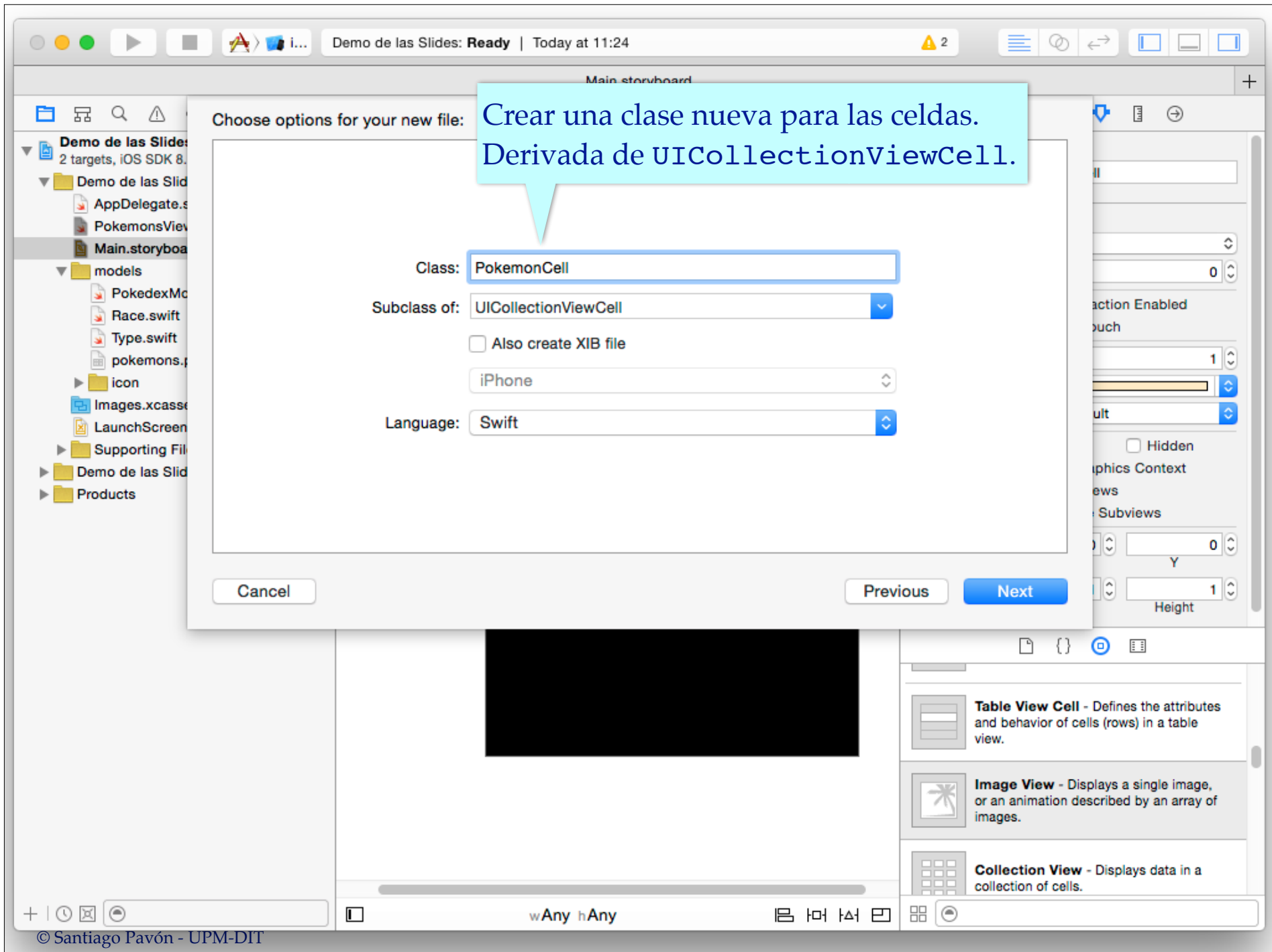
```
//  
// PokemonsViewController.swift  
// Demo de las Slides  
//  
// Created by Santiago Pavón on 3/12/14.  
// Copyright (c) 2014 UPM. All rights reserved.  
//  
import UIKit  
  
let reuseIdentifier = "Cell"  
  
class PokemonsViewController: UICollectionViewController {  
    private var pokedexModel = PokedexModel()  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        // Uncomment the following line to preserve selection  
        // between presentations  
        // self.clearsSelectionOnViewWillAppear = false  
  
        // Register cell classes  
        self.collectionView.registerClass(UICollectionViewCell,  
            self, forCellWithReuseIdentifier: reuseIdentifier)  
  
        // Do any additional setup after loading the view.  
    }  
  
    override func didReceiveMemoryWarning() {  
    }  
}
```

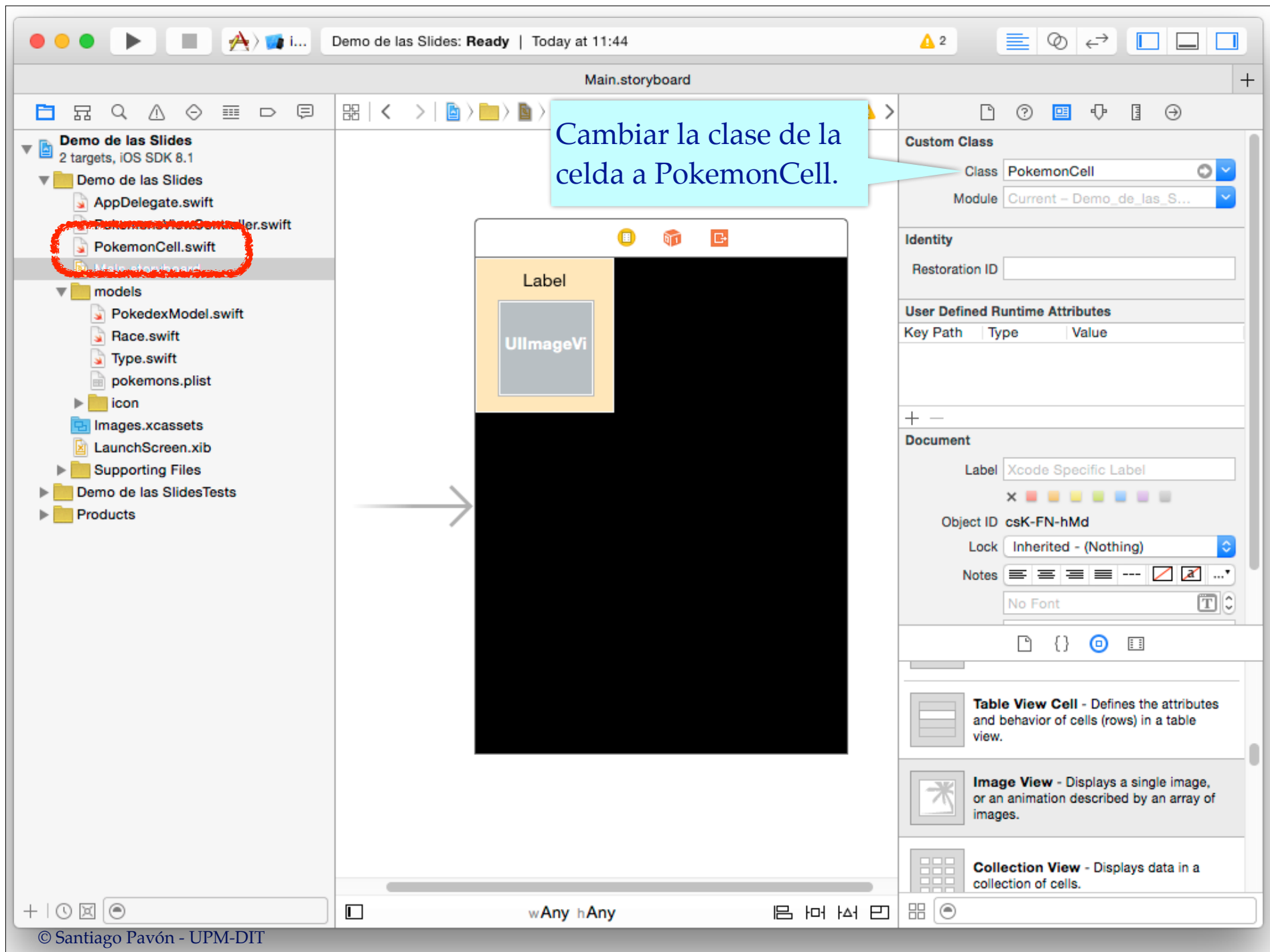


The screenshot shows the Xcode interface for editing a storyboard. On the left, the project navigator shows a project named "Demo de las Slides" with a "Main.storyboard" file. The storyboard canvas in the center shows a "Label" containing a "UIImageView" object. A red arrow points from the "UIImageView" object in the storyboard to the "Identifier" field in the right-hand inspector panel, which is set to "Pokemon Cell". The inspector panel also shows various view properties such as "Mode", "Tag", "Interaction", "Alpha", "Background", "Tint", "Drawing", "Stretching", and "Width/Height".

Poner un identificador de reutilización a la celda.

Diseñar la celda.  
Añado una UILabel y una UIImageView desde la librería de objetos.  
Añadir también las restricciones de autolayout.





Ctrl-B

Crear dos outlets en PokemonCell para la etiqueta y la imagen.

```
//  
// PokemonCell.swift  
// Demo de las Slides  
//  
// Created by Santiago Pavón on 3/12/14.  
// Copyright (c) 2014 UPM. All rights reserved.  
//  
import UIKit  
  
class PokemonCell: UICollectionViewCell {  
    @IBOutlet weak var nameLabel: UILabel!  
    @IBOutlet weak var iconImageView: UIImageView!  
}
```

Running Demo de las Slides on iPhone 6

PokemonsViewController.swift

Demo de las Slides > Demo de las Slides > PokemonsViewController.swift > viewDidLoad()

```
import UIKit

let reuseIdentifier = "Pokemon Cell"

class PokemonsViewController: UICollectionViewViewController {

    private var pokedexModel = PokedexModel()

    override func viewDidLoad() {
        super.viewDidLoad()

        // Uncomment the following line to preserve selection between presentations
        // self.clearsSelectionOnViewWillAppear = false

        // Register cell classes
        // self.collectionView.registerClass(UICollectionViewCell.self,
        // |                                     forCellWithReuseIdentifier: reuseIdentifier)

        // Do any additional setup after loading the view.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that
    }

    /*
    // MARK: - Navigation
```

Identificador de reutilización de las celdas.

Método para registrar que clase hay que usar para construir las celdas. Lo comentamos ya que hemos definido un prototipo en el storyboard.

Demo de las Slides

© Santiago Pavón - UPM-DIT

```
Running Demo de las Slides on iPhone 6
PokemonsViewController.swift
Demo de las Slides > Demo de las Slides > PokemonsViewController.swift > collectionView(_:cellForItemAtIndexPath:)
}
*/
// MARK: UICollectionViewDataSource

override func numberOfSectionsInCollectionView(collectionView: UICollectionView) ->
    Int {
    return pokedexModel.types.count
}

override func collectionView(collectionView: UICollectionView,
    numberOfItemsInSection section: Int) -> Int {
    return pokedexModel.types[section].races.count
}

override func collectionView(collectionView: UICollectionView,
    cellForItemAtIndexPath indexPath: NSIndexPath) -> UICollectionViewCell {
    let cell = collectionView.dequeueReusableCellWithReuseIdentifier(
        |reuseIdentifier, forIndexPath: indexPath) as PokemonCell

    let type = pokedexModel.types[indexPath.section]
    let race = type.races[indexPath.item]

    cell.nameLabel.text = race.name
    cell.iconImageView.image = UIImage(named: race.icon)

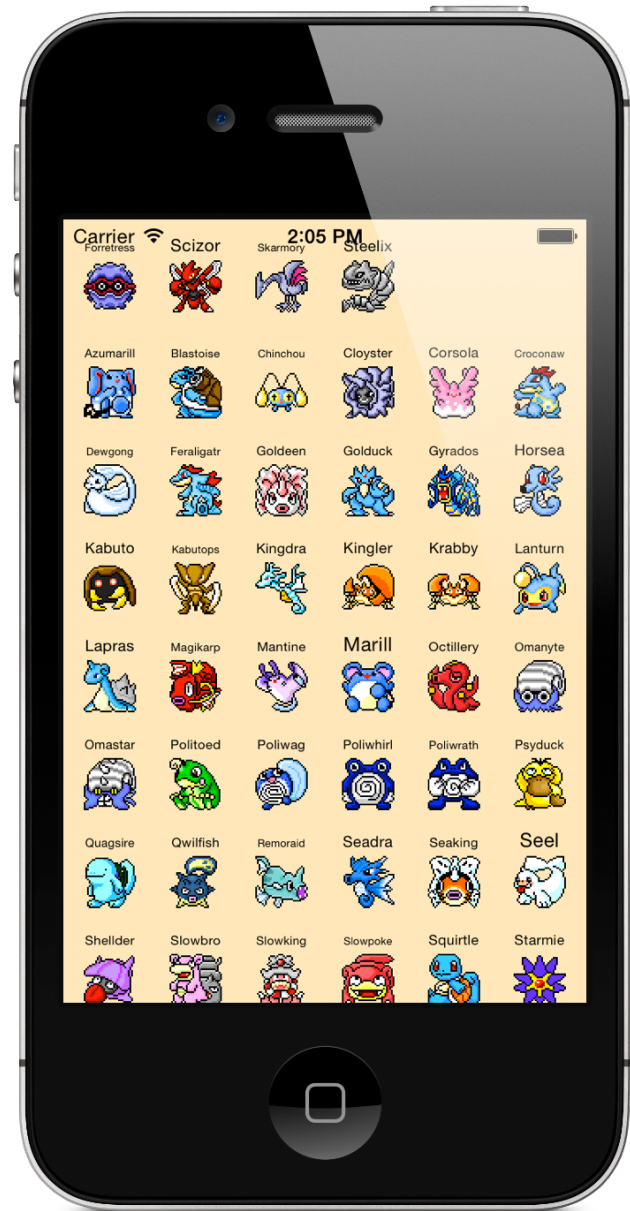
    return cell
}

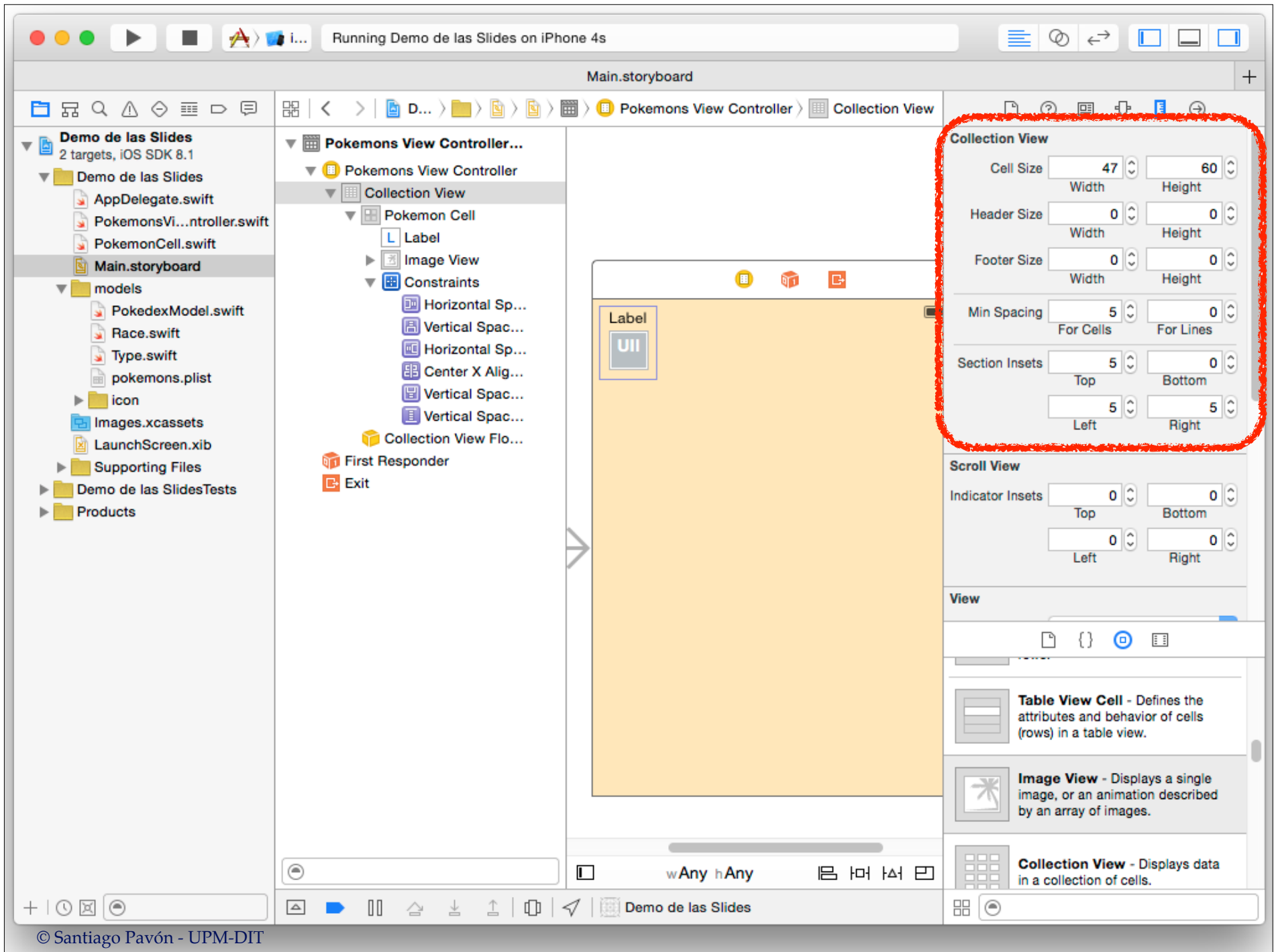
// MARK: UICollectionViewDelegate
```

Métodos del Data Source

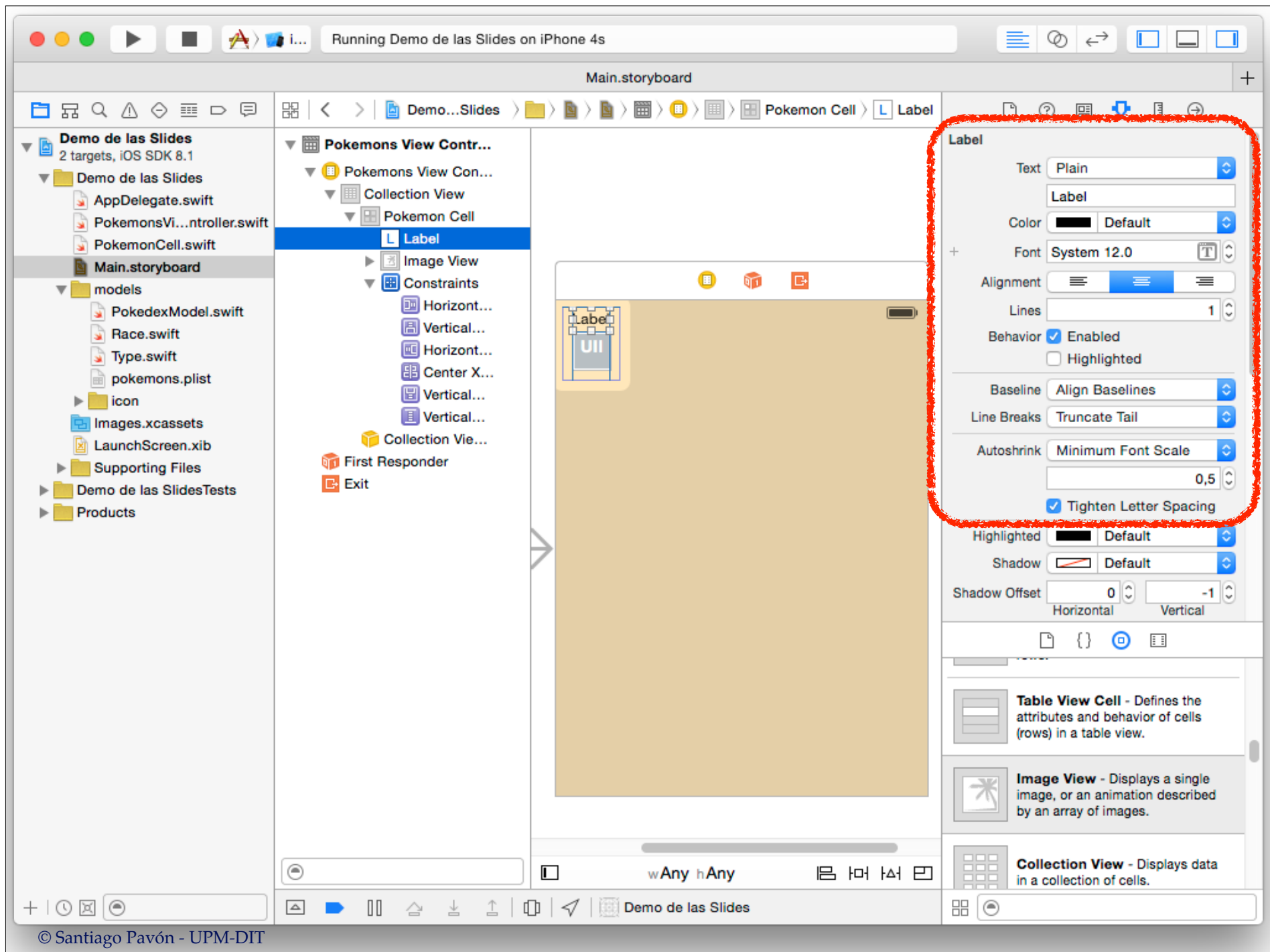


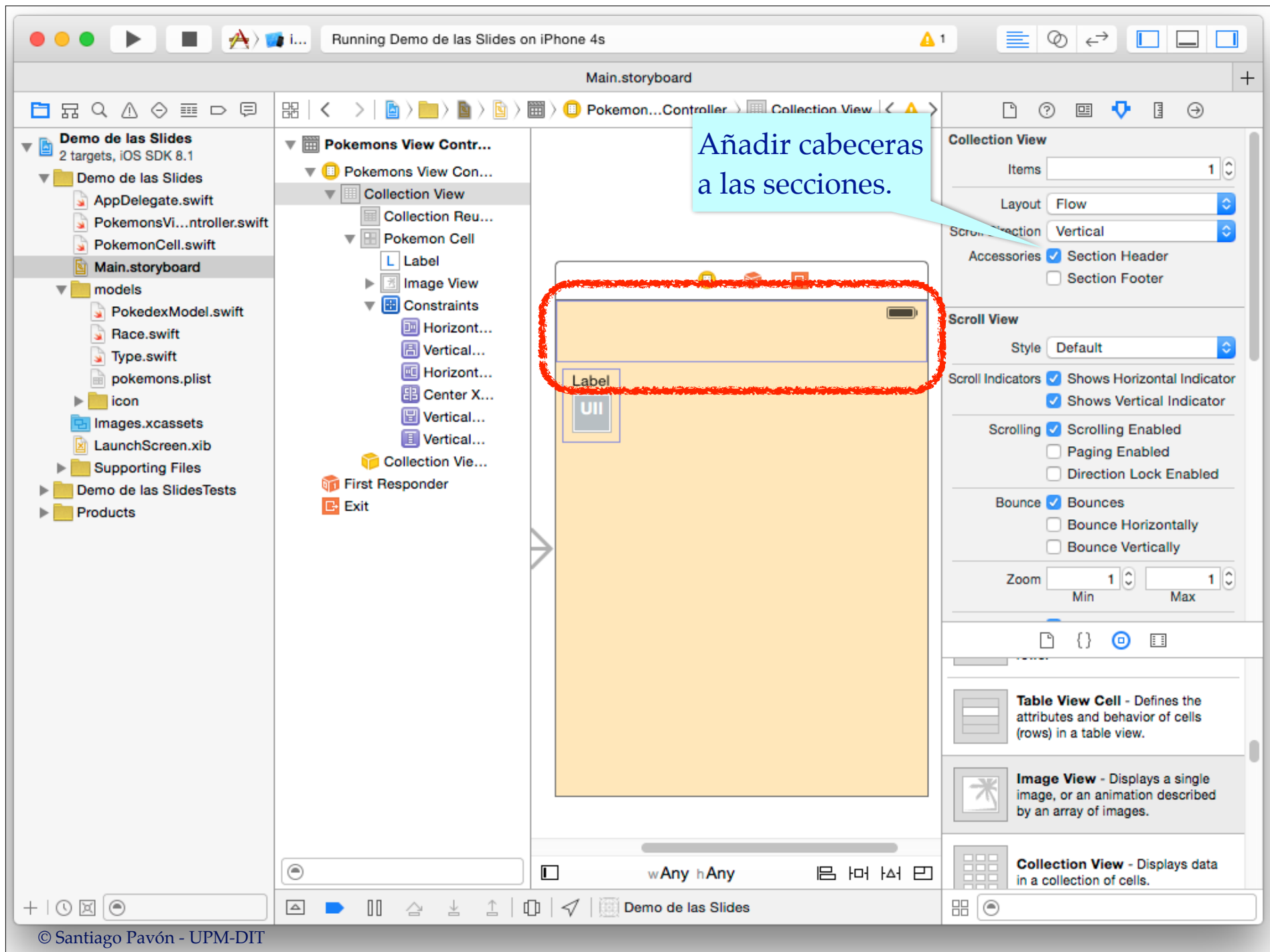
Retocar tamaños, colores, fonts en el inspector.

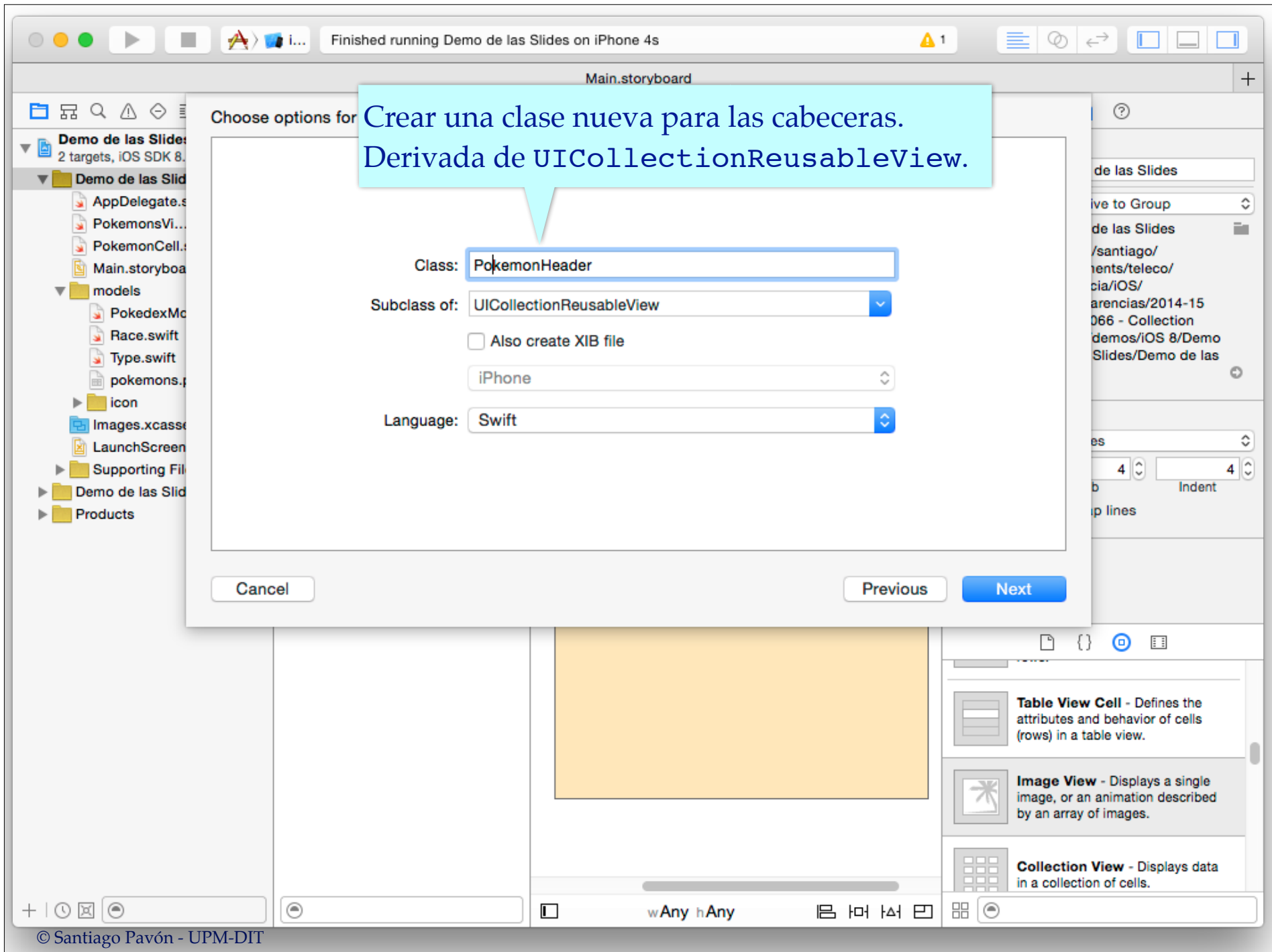




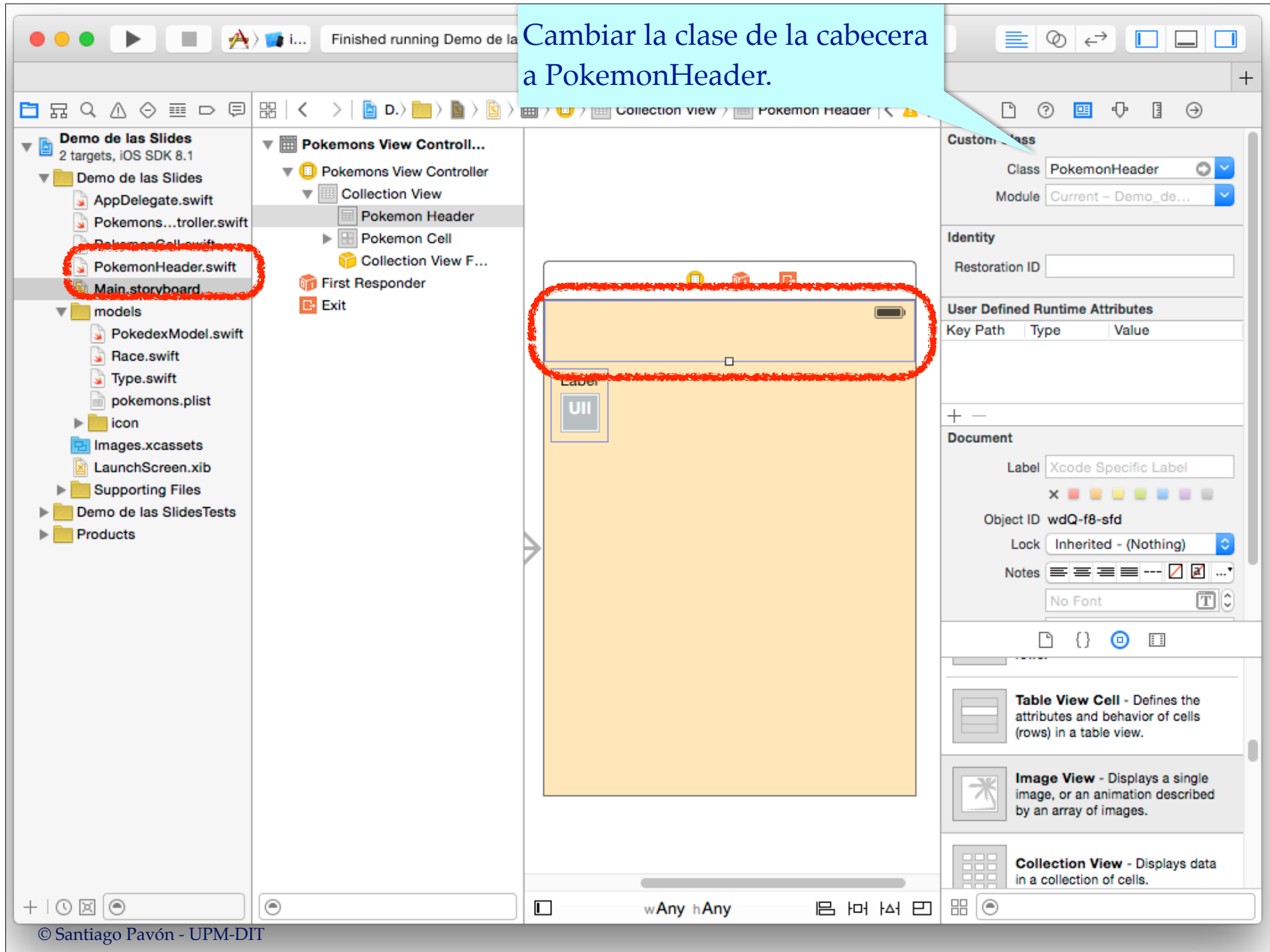


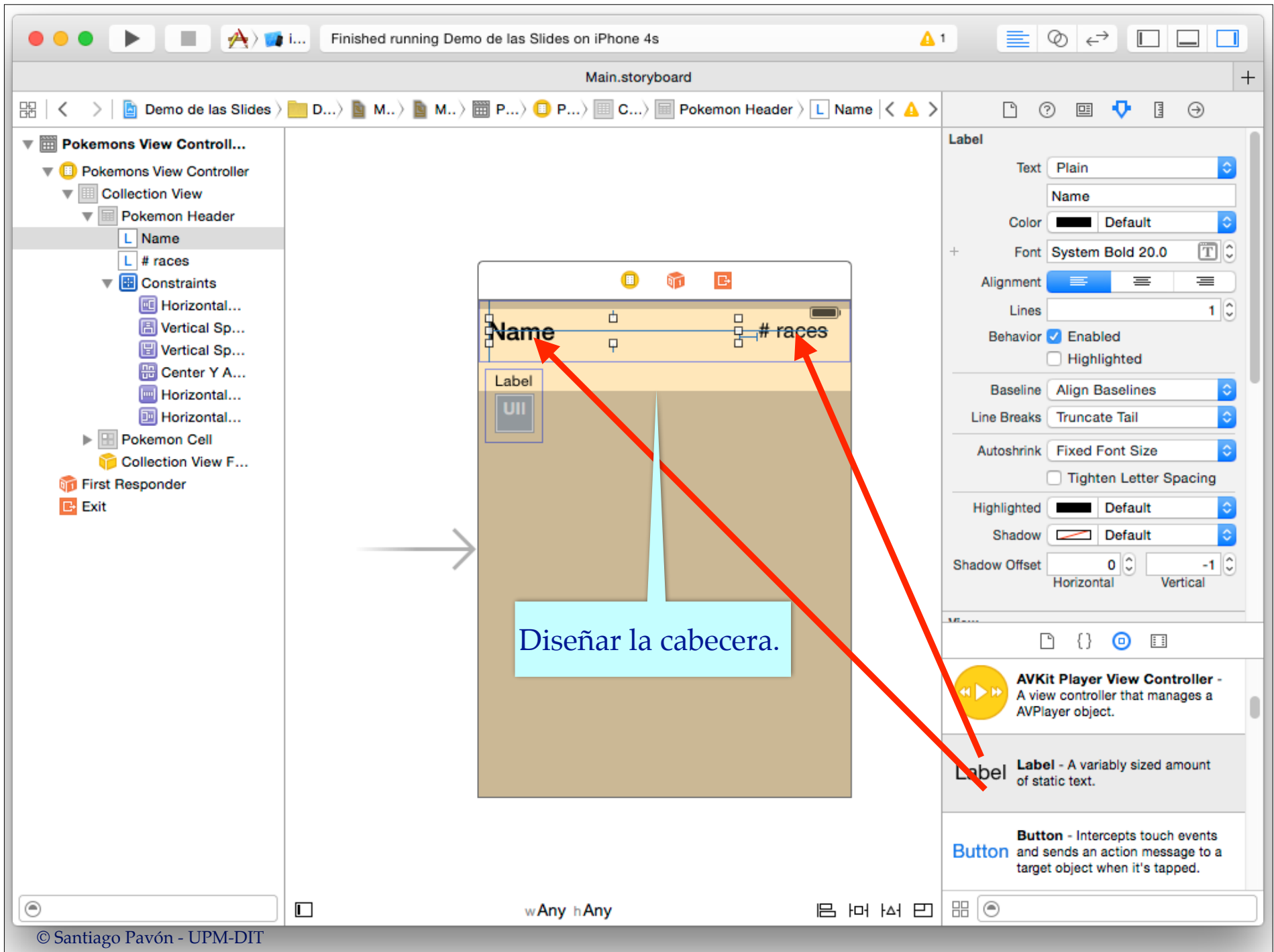


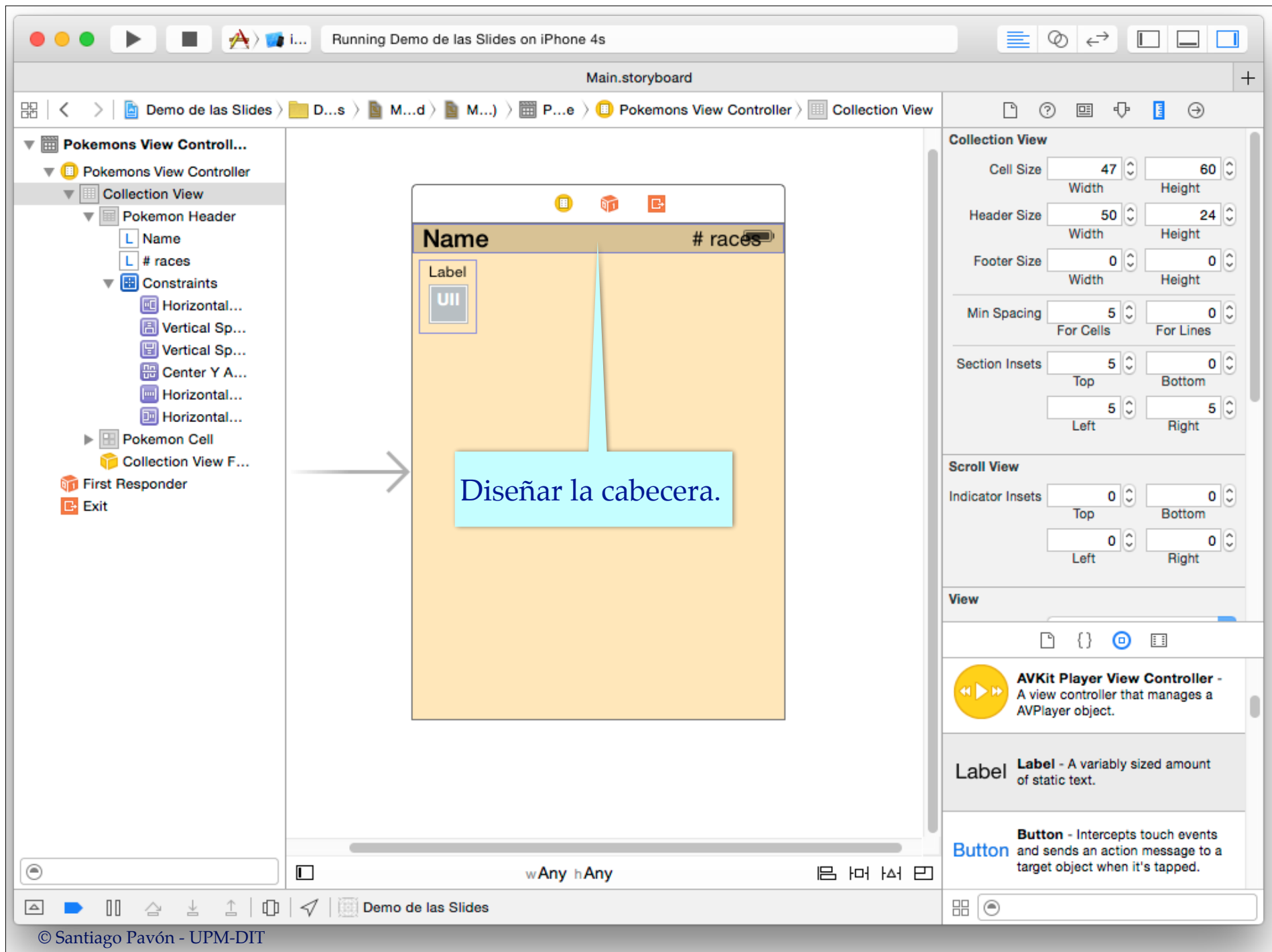


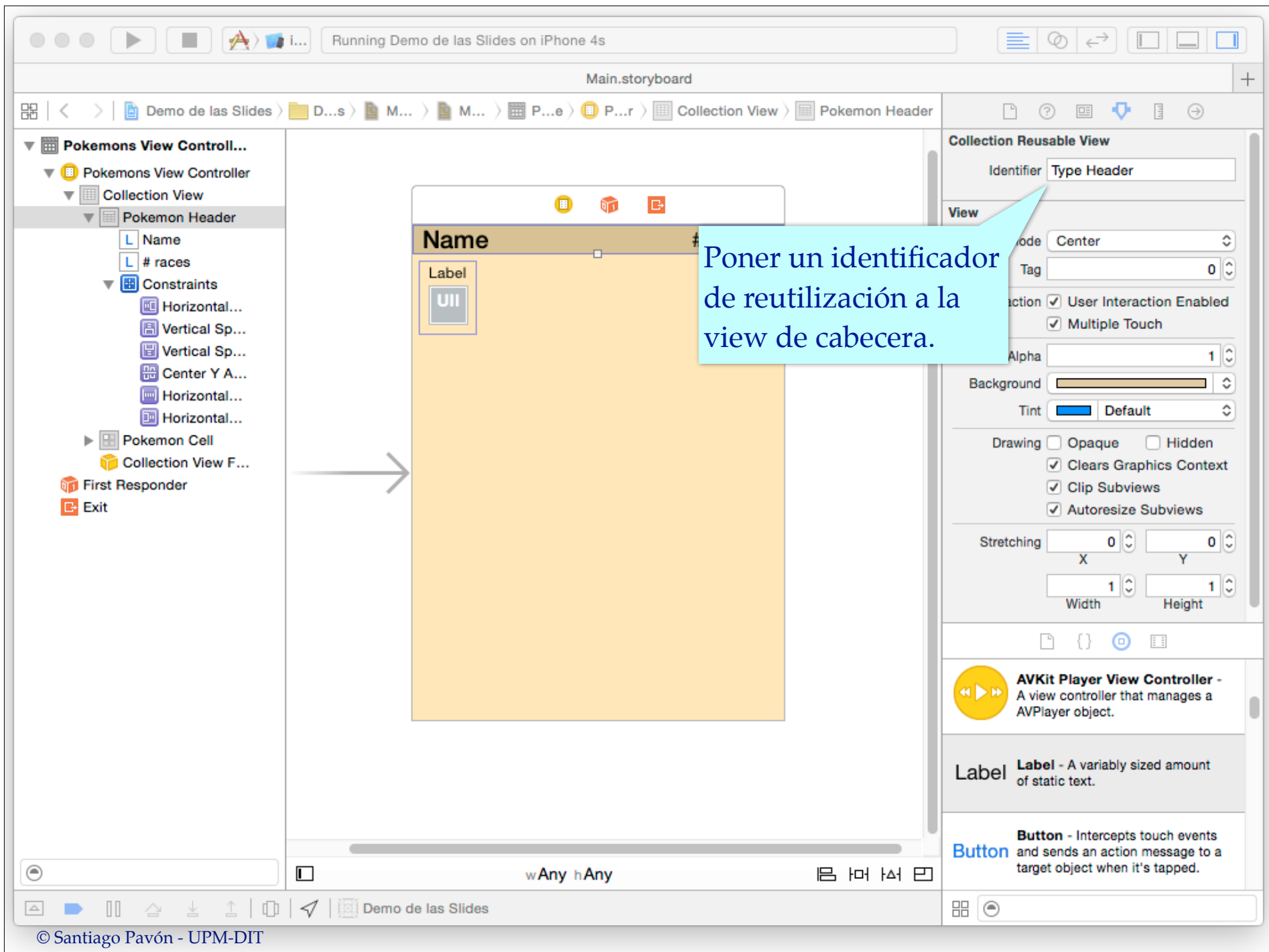


Cambiar la clase de la cabecera a PokemonHeader.









Running Demo de las Slides on iPhone 4s

Main.storyboard

Manual > Demo...lides > Demo...lides > PokemonHeader.swift > PokemonHeader

Ctrl-B1

Name # races

Label UI

Crear dos outlets en PokemonHeader para el nombre del tipo y el número de razas.

```
//  
// PokemonHeader.swift  
// Demo de las Slides  
//  
// Created by Santiago Pavón on 3/12/14.  
// Copyright (c) 2014 UPM. All rights reserved.  
//  
import UIKit  
  
class PokemonHeader: UICollectionViewReusableView {  
    @IBOutlet weak var nameLabel: UILabel!  
    @IBOutlet weak var infoLabel: UILabel!  
}
```

wAny hAny

Demo de las Slides



```
Running Demo de las Slides on iPhone 4s
PokemonsViewController.swift
Demo de las Slides > Demo de las Slides > PokemonsViewController.swift > PokemonsViewController

return cell
}

override func collectionView(collectionView: UICollectionView,
viewForSupplementaryElementOfKind kind: String,
atIndexPath indexPath: NSIndexPath) -> UICollectionViewReusableView {

    var cell: UICollectionViewReusableView! = nil
    if kind == UICollectionViewElementKindSectionHeader {

        cell = collectionView.dequeueReusableSupplementaryViewOfKind(kind,
            withReuseIdentifier: "Type Header", forIndexPath: indexPath) as
            PokemonHeader

        let type = pokedexModel.types[indexPath.section]

        (cell as PokemonHeader).nameLabel.text = type.name
        (cell as PokemonHeader).infoLabel.text = "\((type.races.count) razas"
    }
    return cell
}

// MARK: UICollectionViewDelegate
/*
```

Método del Data Source



The screenshot shows the Xcode storyboard editor for a project named "Demo de las Slides". The storyboard is titled "Main.storyboard" and contains three scenes: "Navigation Controller", "Pokedex", and "Title".

- Navigation Controller:** A gray scene representing a UINavigationController.
- Pokedex:** An orange scene representing a Pokedex view. It contains a table with two columns: "Name" and "# races". Below the table is a "Label" and a "UI" element.
- Title:** A blue scene representing a Title view. It contains a "Title" label and a "UIWebView" element.

Arrows indicate the flow of the storyboard: from the Navigation Controller to the Pokedex, and from the Pokedex to the Title. A callout box at the bottom explains the purpose of the Navigation Controller and the UIWebView.

Usar un Navigation Controller y otra escena con un UIWebView para mostrar información descargada de internet. Similar al ejemplo Pokedex.

```
Finished running Demo de las Slides on iPhone 4s
PokemonsViewController.swift
Demo de las Slides > Demo de las Slides > PokemonsViewController.swift > PokemonsViewController

override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
    // Dispose of any resources that can be recreated.
}

// MARK: - Navigation

// In a storyboard-based application, you will often want to do a little
// preparation before navigation
override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject?) {

    if segue.identifier == "Show Web Info" {

        if let ip = collectionView.indexPathForCell(sender as PokemonCell) {

            if let wvc = segue.destinationViewController as? WebViewController {

                let type = pokedexModel.types[ip.section]
                let race = type.races[ip.item]

                wvc.race = race
            }
        }
    }
}

// MARK: UICollectionViewDataSource
```

