



POLITÉCNICA

ETSIT
UPM

dit
UPM

Desarrollo de Apps para iOS Container View Controllers

IWEB,LSWC 2014-2015
Santiago Pavón

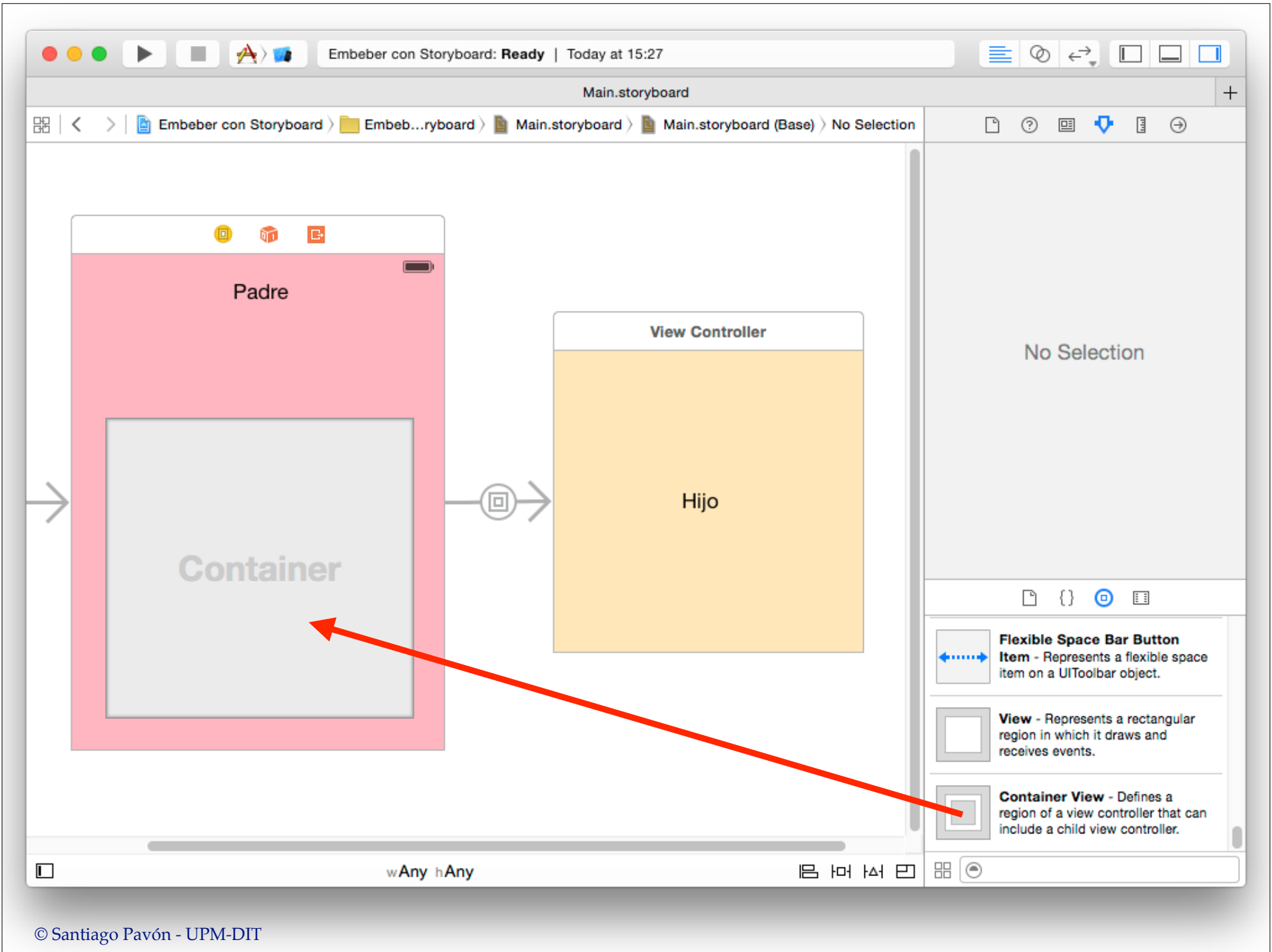
ver: 2014.11.03

¿En qué Consiste?

- Objetivo:
 - Que un View Controller muestre dentro de su vista las vistas de otros View Controllers.
 - Es interesante para evitar que el código de un VC sea muy grande / complejo. Parte de la funcionalidad se desarrolla en otro VC que luego se incluye en la vista. También para facilitar la reutilización.
- Tareas:
 - Además de construir la jerarquía completa de UIViews, es necesario construir la jerarquía de UIViewController para que los métodos del ciclo de vida de los VC incluidos se llamen adecuadamente:
 - al rotar el terminal, cuando aparece y desaparece el VC, al recalcular layouts, ...

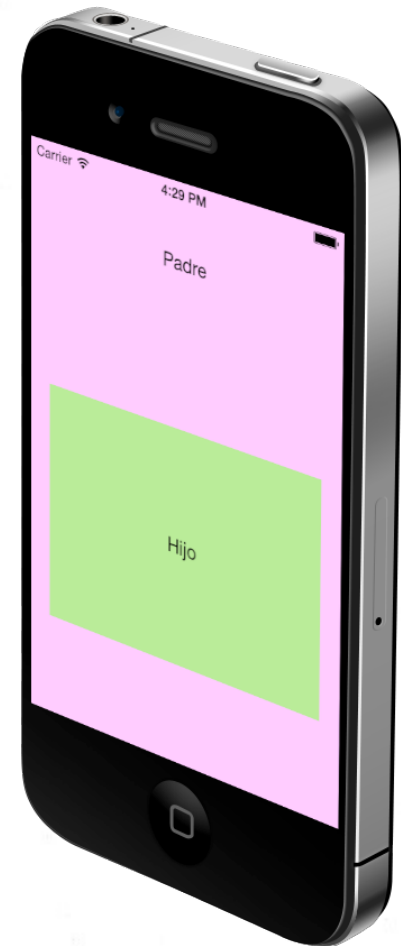
Con Storyboard - Interface Builder

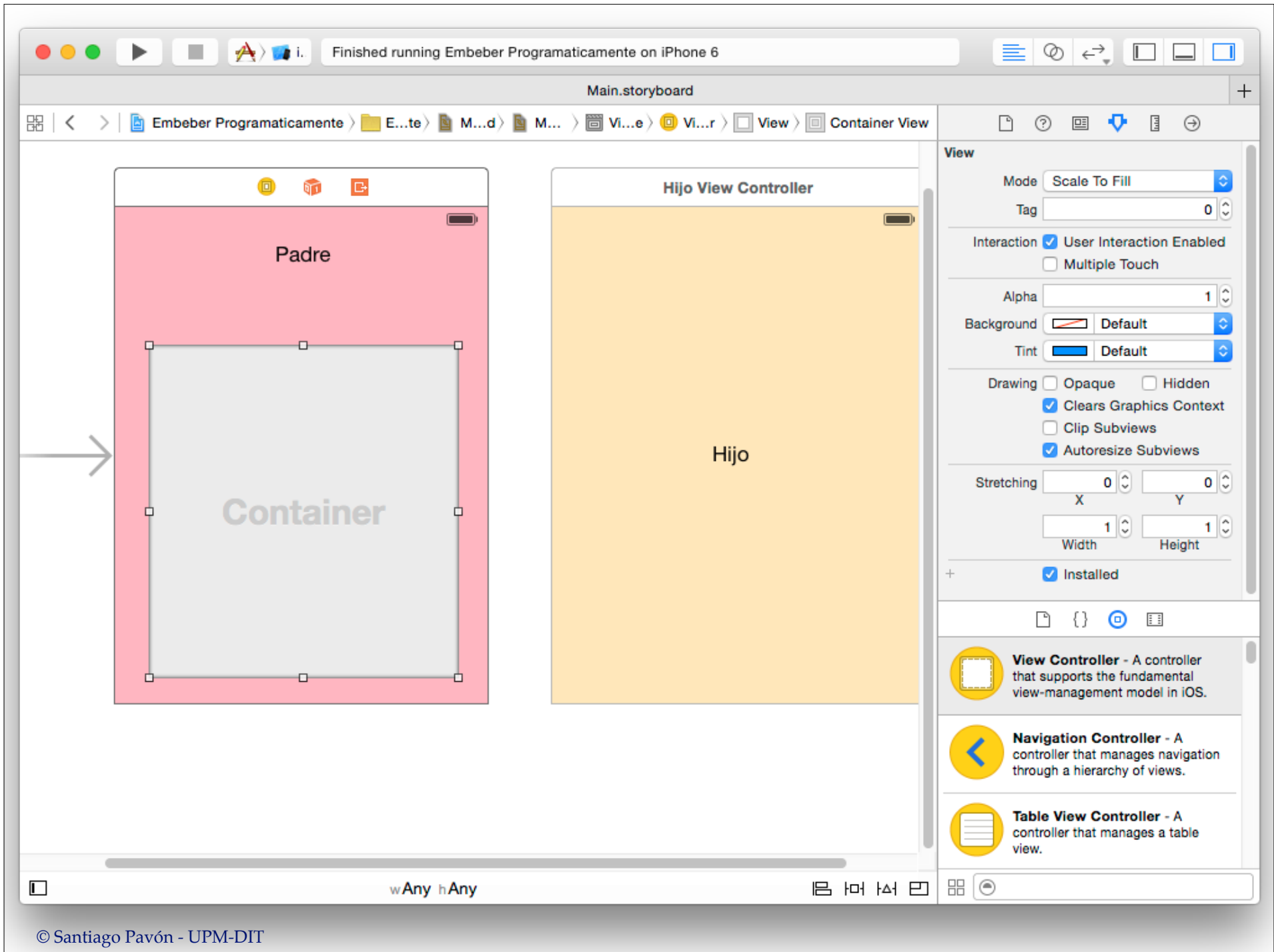
- Se usan Container Views
 - Son views cuyo contenido es la view de otro VC.
- Arrastrarlas desde la biblioteca de objetos hasta el Storyboard.
- Un Container View se enlaza con el View Controller que mostrará con **segues** de tipo **embed**.
 - Con los segues embed también puede sobrecribirse el método **prepareForSegue:sender:** del VC contenedor para configurar el VC embebido.
 - No olvidar asignar un identificador único al segue.



Programáticamente

- Crear la jerarquía de UIViews, que será una mezcla de las vistas de varios VC.
- Crear la jerarquía de VC.
 - Para añadir/eliminar un UIViewController como hijo:
addChildViewController:
removeFromParentViewController:
 - Antes de añadir o eliminar un VC hijo:
willMoveToParentViewController:
 - Este método ya se llama automáticamente desde addChildViewController:
 - Para hacer transiciones entre VC hijos:
transitionFromViewController:toViewController:
duration:options:animations:completion:
 - Después de añadir o eliminar un VC hijo:
didMoveToParentViewController:
 - No se llama automáticamente.
 - Debemos llamarlo después de addChildViewController:
 - y una vez hayan terminado las animaciones de las transiciones.
- Gestionar Trait Collection y cambios de tamaño:
 - Cambiar el TraitCollection de un View Controller hijo:
setOverrideTraitCollection:forChildViewController:
 - Especificar el tamaño preferido de un View Controller:
preferredContentSize
 - Informar sobre cambios de tamaño:
preferredContentSizeDidChangeForChildContentContainer:
sizeForChildContentContainer:withParentContainerSize:
viewWillTransitionToSize:withTransitionCoordinator:





```

class ViewController: UIViewController {

    @IBOutlet weak var container: UIView!

    var ivc: HijoViewController?

    override func viewDidLoad() {
        super.viewDidLoad()

        // Creo el objeto VC hijo:
        ivc = storyboard?.instantiateViewControllerWithIdentifier(
            "Hijo VC") as? HijoViewController

        if ivc != nil {

            // Construir jerarquia de views
            container.addSubview(ivc!.view)

            // Construir jerarquia de View Controllers
            addChildViewController(ivc!)
            ivc!.didMoveToParentViewController(self)
        }
    }

    override func viewDidLoadSubviews() {
        // Ajusto geometrias
        ivc?.view.frame = container.bounds;
    }
}

```

