ETSIT UPM

dit UPM

Desarrollo de Apps para iOS
Demo Game Top3

IWEB 2015-2016

Santiago Pavón

ver: 2015.11.25

Índice

- Desarrollo de un servidor web para usar en los ejemplos de este tema.
  - Guarda los puntos obtenidos por los usuarios en un juego.
  - El servidor tiene un API REST.
  - Desarrollo del servidor con Rails.
- Desarrollo de una aplicación que baje las tres mejores puntuaciones.
  - Paso 1: Desarrollar el GUI de la aplicación usando una TableView.
  - Paso 2: Implementar una descarga de datos JSON usando:
    - Opción 1: NSData(contentsOfURL:).
    - Opción 2: NSURLSession y DataTask.
- Realizar una aplicación que suba nuevas puntuaciones al servidor.

Desarrollar el Servidor Web Game Top-3

Desarrollo con Ruby on Rails

El Servicio Web

- Servicio Web
  - Almacena nombres y puntuaciones obtenidas en las partidas de un juego
  - Consultar las 3 mejores puntuaciones.
- Servicio Web RESTful implementado con Rails 4.
  - Suponemos que Ruby on Rails ya está instalado.

Crear la aplicación Rails

- Ejecutar:

```
$ rails new gametop3
$ cd gametop3
$ rails generate scaffold Score \
        name:string total:integer
$ rake db:migrate
```

- Editar el fichero **config/routes.rb**

```
resources :scores do
   collection do
      get "top3"
   end
end
```

- Añadir la acción **top3** al controlador **Scores**

```
def top3
    @scores = Score.find(:all,
        :limit => 3, :order => 'total desc')

    respond_to do |format|
       format.html {render action: "index"}
       format.xml  {render xml:  @scores}
       format.json {render json: @scores}
    end
end
```

app/controllers/scores_controller.rb

- Para eliminar la protección contra ataques CSRF (Cross-Site Request Forgery), editar el fichero **application_controller.rb**.

```
class ApplicationController < ActionController::Base

  protect_from_forgery with: :null_session

end
```

- Lanzar el servidor:

```
$ rails server
```

- Recuperar datos:
  - Desde un navegador:

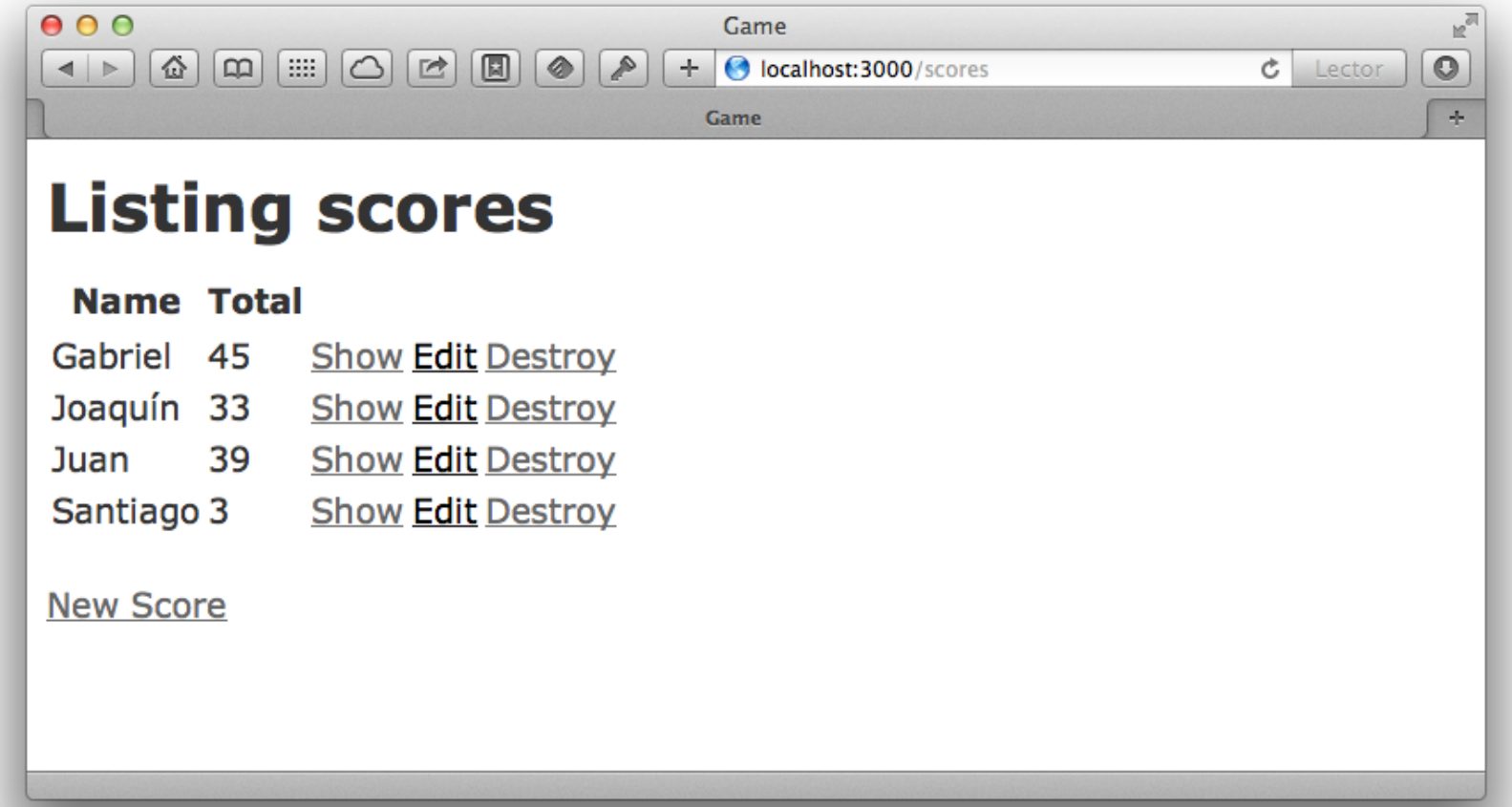
```
http://localhost:3000/scores/top3
```

  - Versión XML:

```
http://localhost:3000/scores/top3.xml
```

  - Versión JSON:

```
http://localhost:3000/scores/top3.json
```



```
<?xml version="1.0" encoding="UTF-8"?>
<scores type="array">
  <score>
    <created-at type="datetime">2012-09-24T08:15:13Z</created-at>
    <id type="integer">3</id>
    <name>Gabriel</name>
    <total type="integer">4</total>
    <updated-at type="datetime">2012-09-24T08:15:13Z</updated-at>
  </score>
  <score>
    <created-at type="datetime">2012-09-24T08:15:27Z</created-at>
    <id type="integer">4</id>
    <name>Jara</name>
    <total type="integer">4</total>
    <updated-at type="datetime">2012-09-24T08:15:27Z</updated-at>
  </score>
  <score>
    <created-at type="datetime">2012-09-24T08:15:01Z</created-at>
    <id type="integer">2</id>
    <name>Nuria</name>
    <total type="integer">2</total>
    <updated-at type="datetime">2012-09-24T08:15:01Z</updated-at>
  </score>
</scores>
```

```
[{"created_at":"2012-09-24T08:15:13Z",
  "id":3,
  "name":"Gabriel",
  "total":4,
  "updated_at":"2012-09-24T08:15:13Z"},
 {"created_at":"2012-09-24T08:15:27Z",
  "id":4,
  "name":"Jara",
  "total":4,
  "updated_at":"2012-09-24T08:15:27Z"},
 {"created_at":"2012-09-24T08:15:01Z",
  "id":2,
  "name":"Nuria",
  "total":2,
  "updated_at":"2012-09-24T08:15:01Z"}
]
```

- **Crear un registro nuevo:**
  - Desde un formulario:

```
$ curl -X POST
       -d 'score[name]=Nuria&score[total]=53'
       http://localhost:3000/scores
```

  - Con datos JSON:

```
$ curl -X POST
       -d '{"score":{"name":"Heliodora","total": 51}}'
       -H 'Content-type: application/json'
       http://localhost:3000/scores.json
```

- **Editar un registro:**
  - Desde un formulario:

```
$ curl -X PUT
       -d 'score[total]=153'
       http://localhost:3000/scores/11
```

  - Con datos JSON:

```
$ curl -X PUT
       -d '{"score":{"total": 21}}'
       -H 'Content-type: application/json'
       http://localhost:3000/scores.json
```

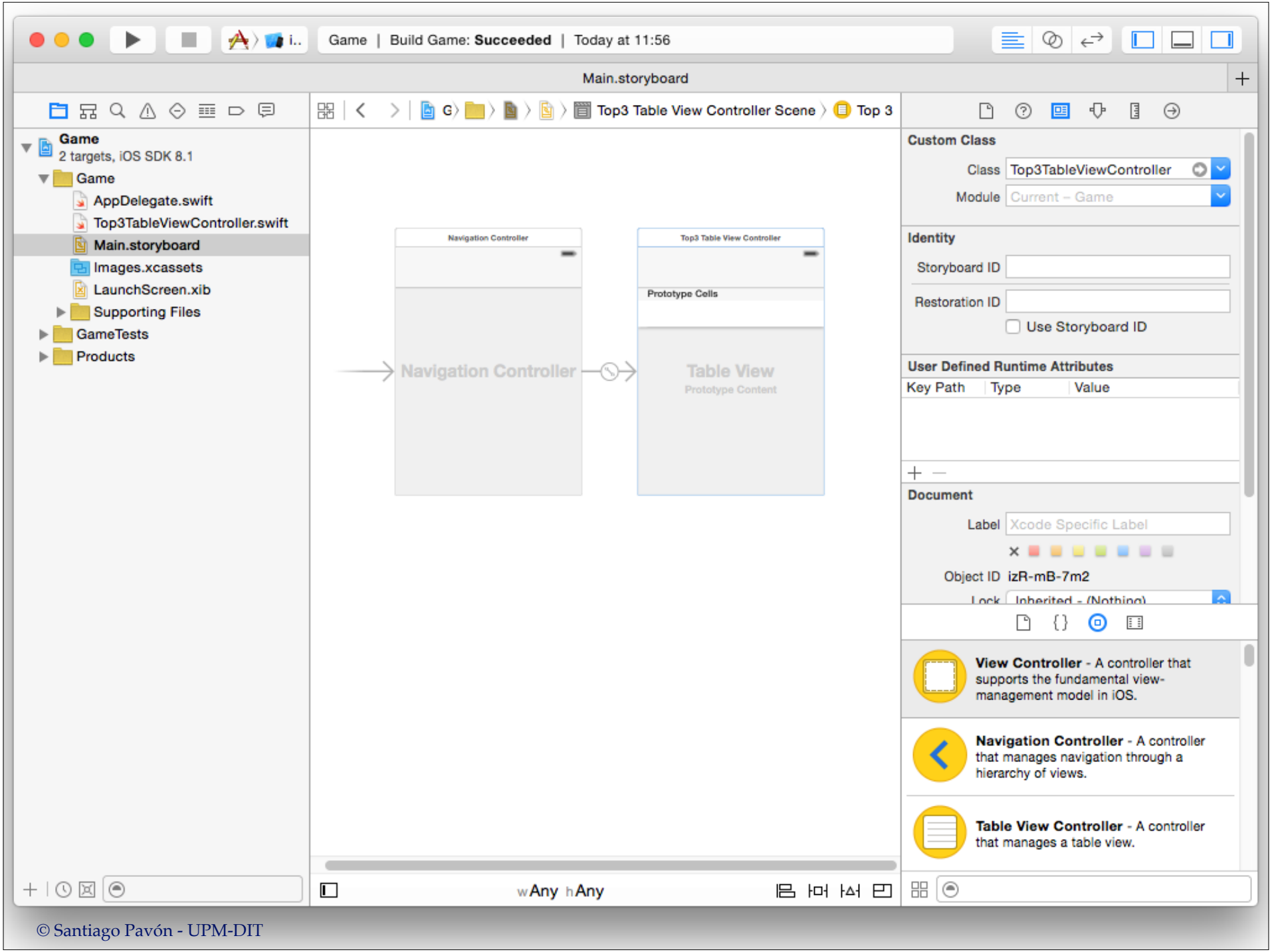
- **Borrar un registro:**

```
$ curl -X DELETE
       http://localhost:3000/scores/11
```

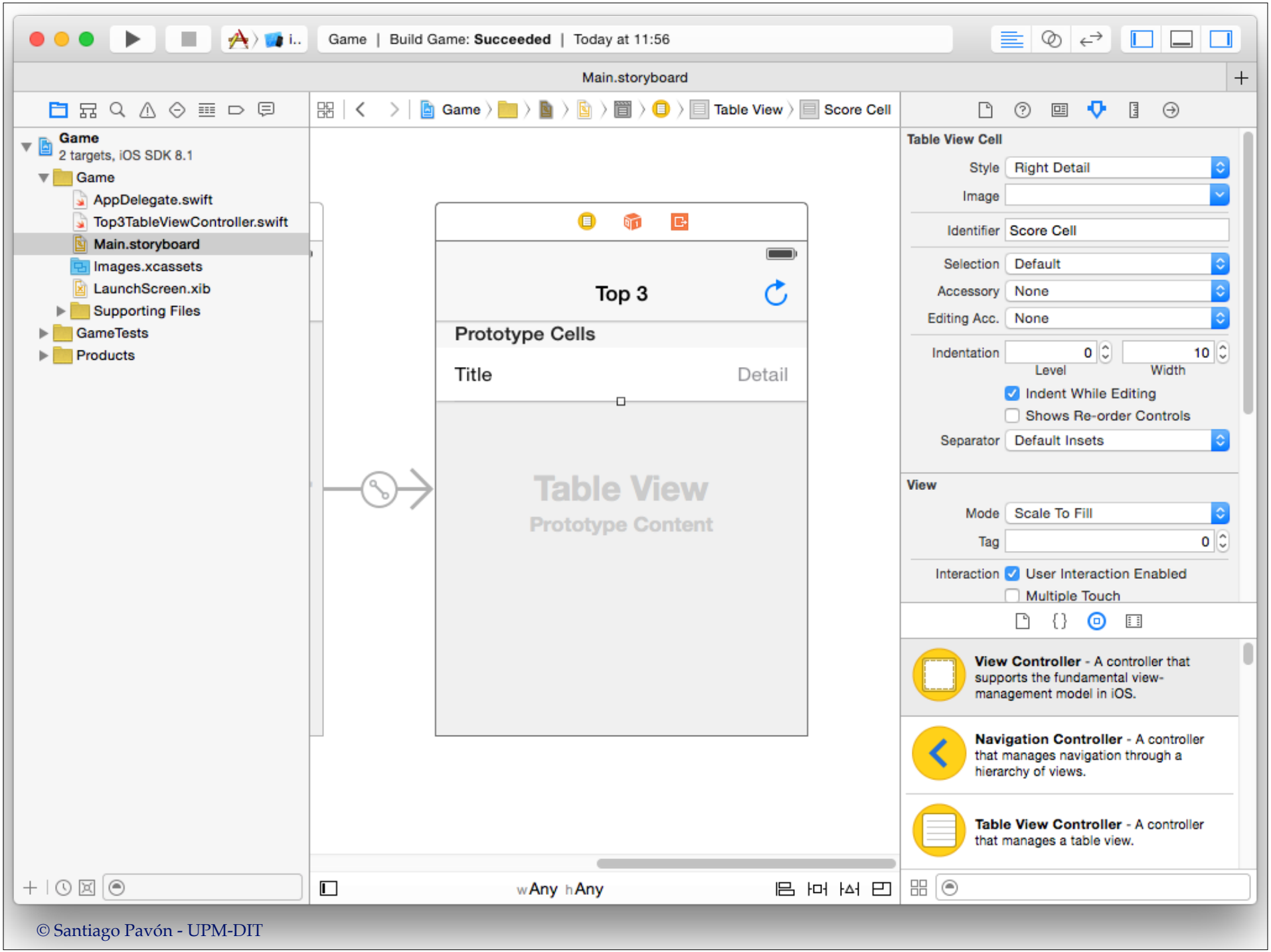
App para Mostrar el Top3

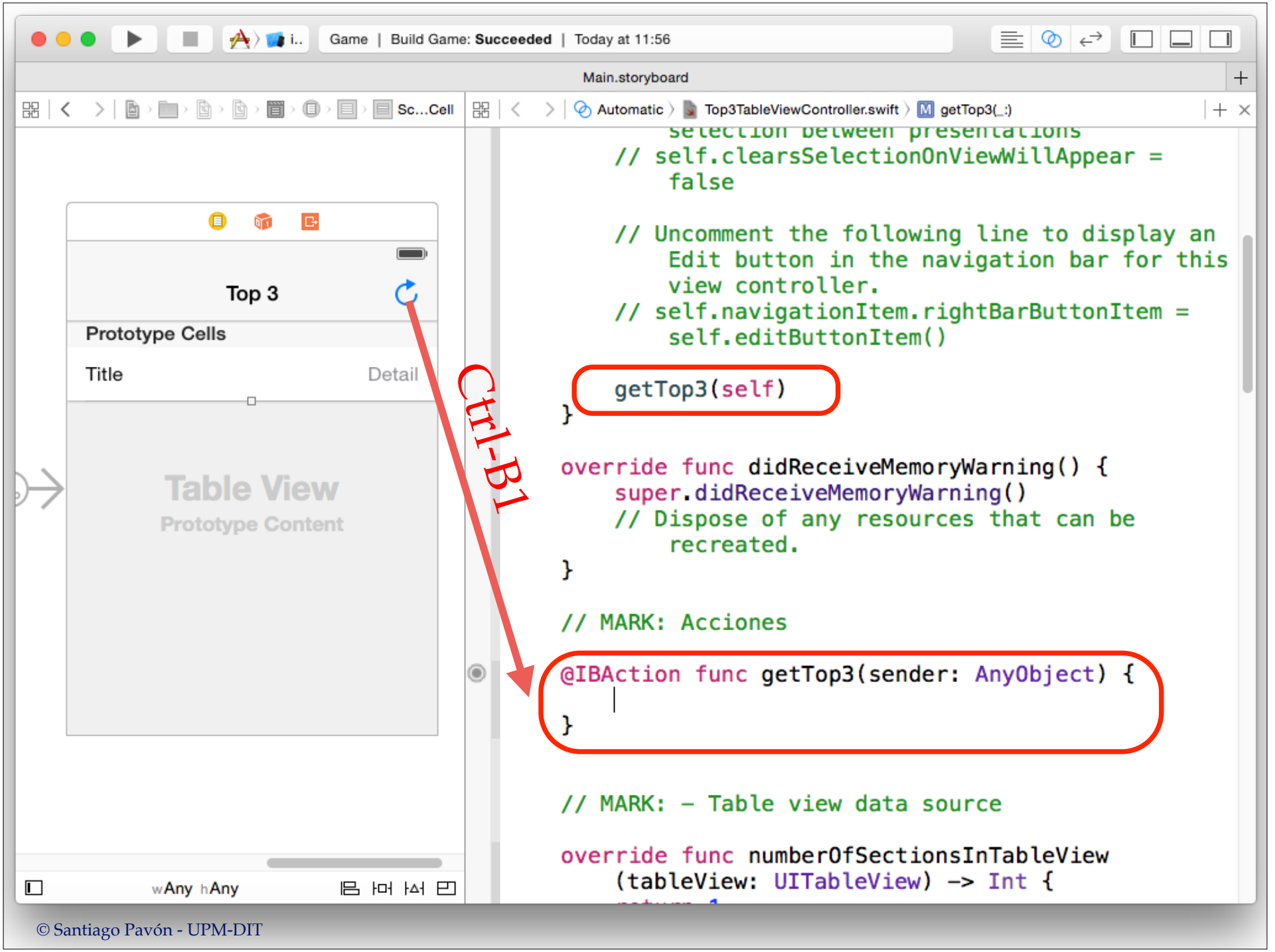
Crear Aplicación: Game

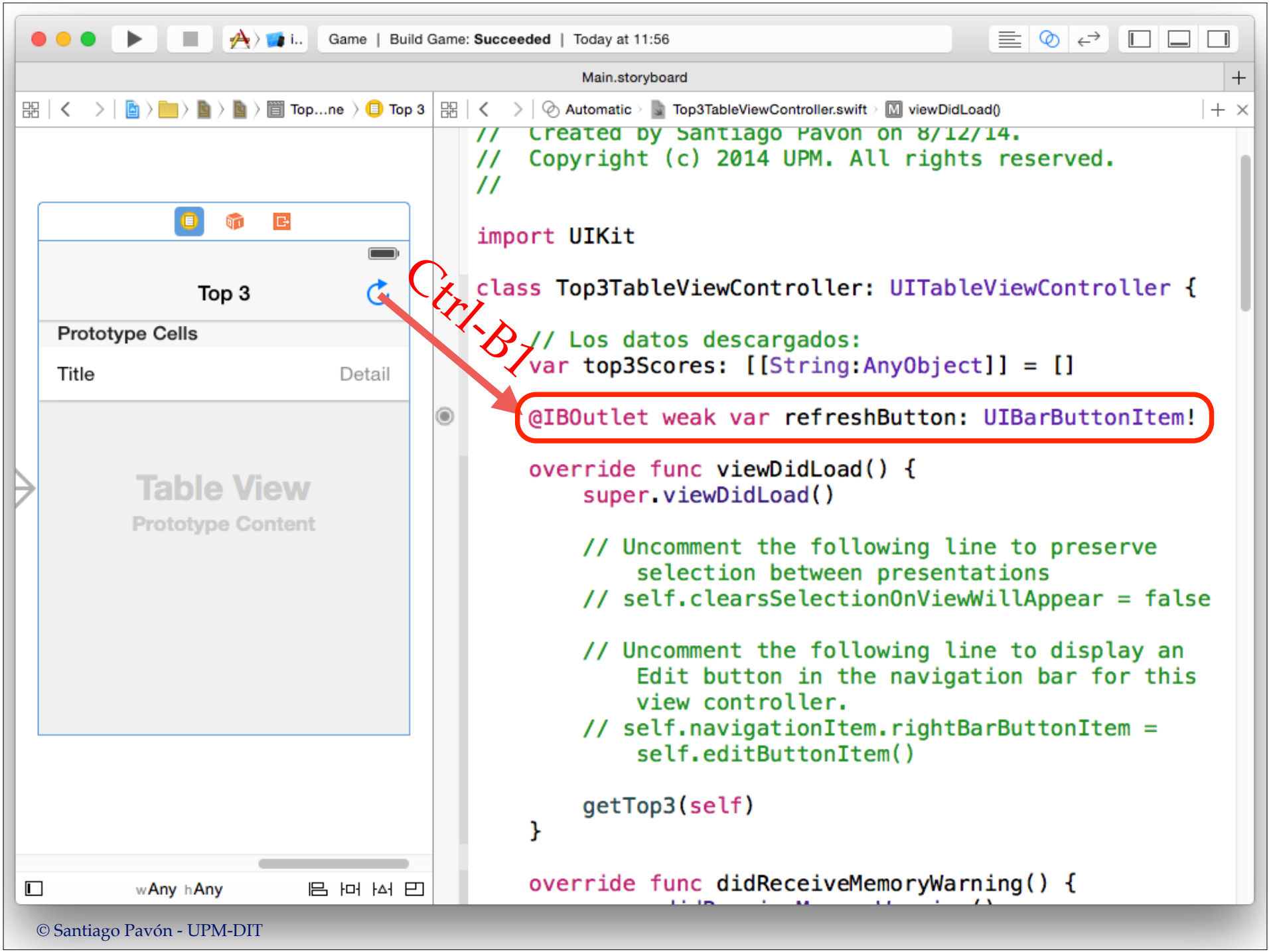
- En esta aplicación mostraremos los datos obtenidos al acceder al recurso top3 de nuestra aplicación web.
- Crear un nuevo proyecto iOS.
  - Usar plantilla Single View Application
    - Nombre de la aplicación: Game
    - La plantilla crea una clase ViewController
      - Borrar esta clase y su escena del storyboard.
    - Crear una clase nueva llamada Top3TableViewController
      - que derive de UITableViewController
    - Añadir al storyboard una escena TableViewController para la clase Top3TableViewController.
    - Meter esta escena en un Navigation Controller.



- Editar el storyboard:
  - Editar los atributos del prototipo de celda:
    - Cambiar el estilo de celda a **Right Detail**.
    - Cambiar el identificador de celda a "**Score Cell**".
  - Poner **Top 3** como título de la barra de navegación.
  - Añadir un Bar Button Item (con identificador **Refresh**) para recargar las puntuaciones.
    - Crear una IBAction para este botón llamada **getTop3**.
    - Llamar al método **getTop3** desde **viewDidLoad** para cargar los datos inicialmente.
    - Crear un outlet a ese botón para poder deshabilitarlo programáticamente.







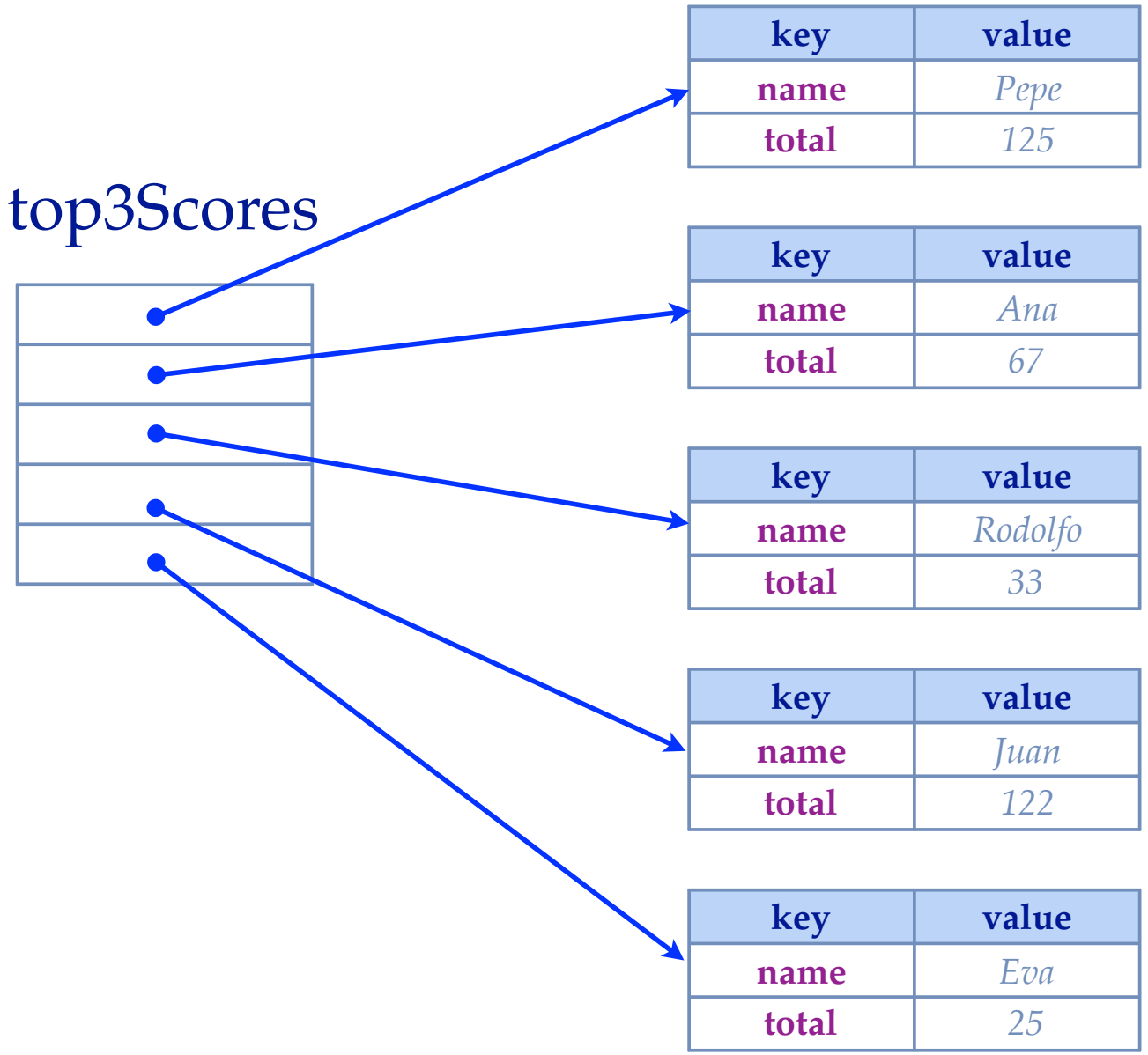
La propiedad top3Scores

- Añadir una propiedad en **Top3TableViewController** para almacenar los datos descargados del servidor web.

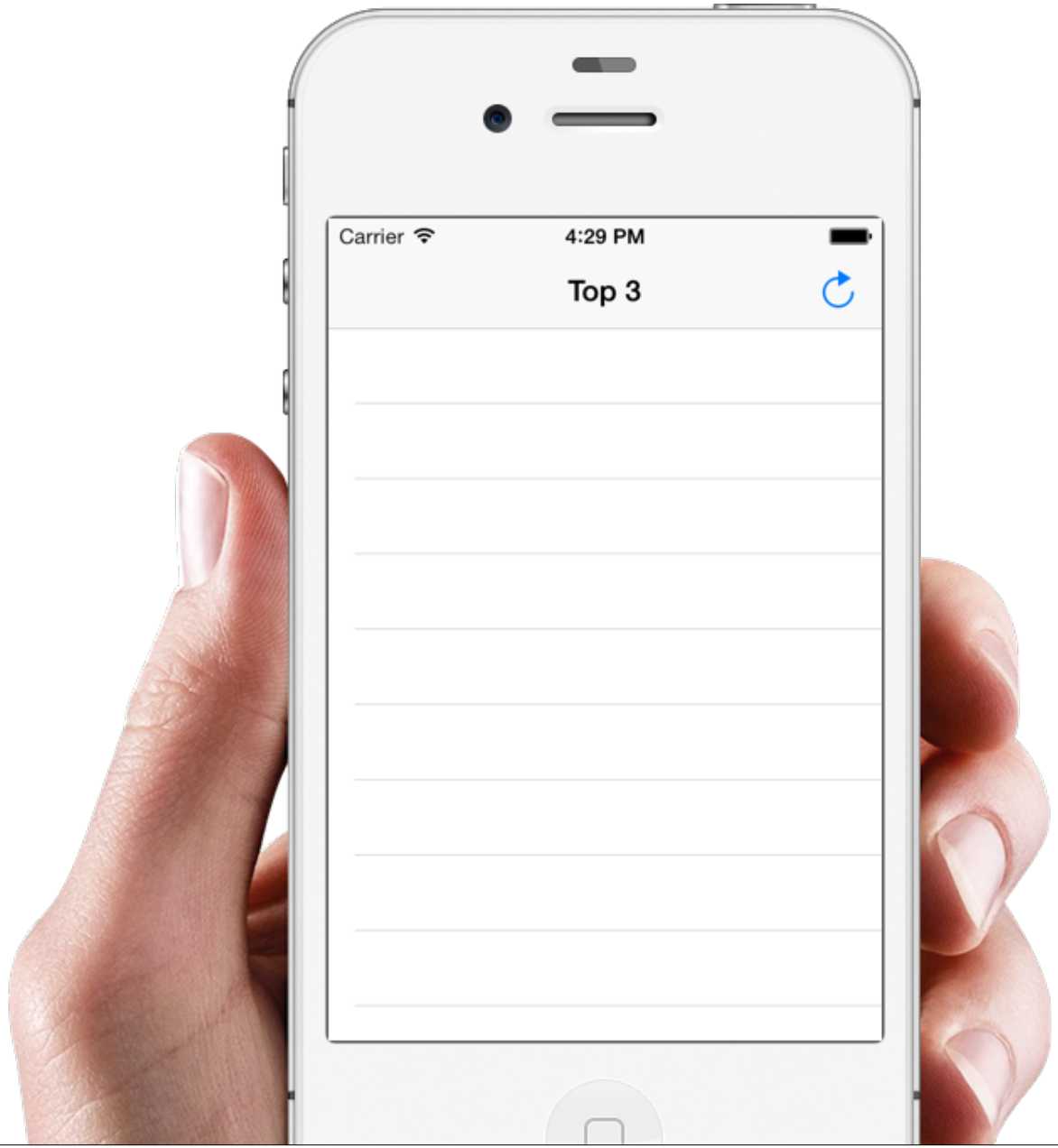
```
var top3Scores: [[String:AnyObject]] = []
```

  - Es un array de diccionarios.
  - Usado por el data source de la tabla.

- Los datos descargados son diccionarios.
  - Un diccionario para cada puntuación.
  - Cada diccionario almacena un nombre y una puntuación
    - Usar la clave "**name**" para acceder al nombre del jugador.
    - Usar la clave "**total**" para acceder a la puntuación.



```
override func numberOfSectionsInTableView(tableView: UITableView)
                                                    -> Int {
    return 1
}

override func tableView(tableView: UITableView,
                numberOfRowsInSection section: Int) -> Int {
    return top3Scores.count
}

override func tableView(tableView: UITableView,
                cellForRowAtIndexPath indexPath: NSIndexPath)
                                                    -> UITableViewCell {

    let cell = tableView.dequeueReusableCellWithIdentifier(
            "Score Cell", forIndexPath: indexPath)

    let dic = top3Scores[indexPath.row]

    cell.textLabel?.text = dic["name"] as? String

    let total = dic["total"] as! Int
    cell.detailTextLabel?.text = "\(total)"

    return cell
}
```



Descargar Datos JSON con **`NSData(contentsOfURL:)`**

Descargar JSON

- Para descargar el contenido JSON de una URL:

```
let GAME_URL = "http://localhost:3000/scores/top3.json"

let escapedURL = GAME_URL.stringByAddingPercentEscapesUsingEncoding(
                                                    NSUTF8StringEncoding)!

let url = NSURL(string: escapedURL)!

let jsonData: NSData? = NSData(contentsOfURL: url)
```

- Ya tenemos un buffer con los datos JSON y extraemos los datos:

```
var newTop3: [[String:AnyObject]]?
var err: NSError?

newTop3 = NSJSONSerialization.JSONObjectWithData(jsonData,
                            options: .allZeros,
                            error: &err) as [[String:AnyObject]]?
```

La acción getTop3

```
let GAME_URL = "http://localhost:3000/scores/top3.json"

@IBAction func getTop3(sender: AnyObject) {
    title = "Descargando..."
    refreshButton.enabled = false
    UIApplication.sharedApplication().networkActivityIndicatorVisible = true

    let escapedURL = GAME_URL.stringByAddingPercentEscapesUsingEncoding(
                                                        NSUTF8StringEncoding)!
    let url = NSURL(string: escapedURL)!

    if let jsonData = NSData(contentsOfURL:url) {
        do {
            var newTop3 = try NSJSONSerialization.JSONObjectWithData(jsonData!,
                                    options: []) as? [[String:AnyObject]]
            top3Scores = newTop3!
            tableView.reloadData()
            title = "Top 3"
        } catch let err as NSError {
            print("Error descargando = \(err.localizedDescription)")
            title = "Desactualizado"
        }
    }

    refreshButton.enabled = true
    UIApplication.sharedApplication().networkActivityIndicatorVisible = false
}
```

La acción getTop3 (GCD)

```
@IBAction func getTop3(sender: AnyObject) {
    title = "Descargando..."
    refreshButton.enabled = false
    UIApplication.sharedApplication().networkActivityIndicatorVisible = true

    let queue = dispatch_queue_create("download queue", DISPATCH_QUEUE_SERIAL)
    dispatch_async(queue, {
        let escapedURL = GAME_URL.stringByAddingPercentEscapesUsingEncoding(NSUTF8StringEncoding)!
        let url = NSURL(string: escapedURL)!
        if let jsonData = NSData(contentsOfURL:url) {
            do {
                var newTop3 = try NSJSONSerialization.JSONObjectWithData(jsonData!,
                                        options: []) as? [[String:AnyObject]]
                dispatch_async( dispatch_get_main_queue(), {
                    self.top3Scores = newTop3!
                    self.tableView.reloadData()
                    self.title = "Top 3"
                })
            } catch let err as NSError {
                dispatch_async( dispatch_get_main_queue(), {
                    print("Error descargando = \(err.localizedDescription)")
                    self.title = "Desactualizado"
                })
            }
        }
        dispatch_async( dispatch_get_main_queue(), {
            self.refreshButton.enabled = true
            UIApplication.sharedApplication().networkActivityIndicatorVisible = false
        })
    })
}
```

NSURLSession

Descargar Datos con un DataTask

Realizar una petición GET

- Objetivo:
  - Hacer una petición HTTP de tipo GET para descargar unos datos de una URL.
- Los pasos a seguir son:
  - Crear y configurar una sesión NSURLSession para todas la tareas que se creen en un futuro.
  - Crear la URL.
    - Previamente escapar caracteres conflictivos.
  - Cada vez que se quieran descargar los datos hay que crear una tarea NSURLSessionDataTask.
    - Al terminar, ejecutará el completionHandler pasado como parámetro.
    - La tarea empieza suspendida; hay que llamar a resume().

```
let GAME_URL = "http://localhost:3000/scores/top3.json"


// La session
var session: NSURLSession!


override func viewDidLoad() {
    super.viewDidLoad()

    // Crear la session
    let config = NSURLSessionConfiguration.defaultSessionConfiguration()
    session = NSURLSession(configuration: config)

   // Descarga inicial de datos
    getTop3(self)
}
```

```
@IBAction func getTop3(sender: AnyObject) {

    title = "Descargando..."
    refreshButton.enabled = false
    UIApplication.sharedApplication().networkActivityIndicatorVisible = true



    // Construir la URL
    let escapedURL = GAME_URL.stringByAddingPercentEscapesUsingEncoding(
                                                        NSUTF8StringEncoding)!
    let url = NSURL(string: escapedURL)!



    // Continua la funcion getTop3 ->
```

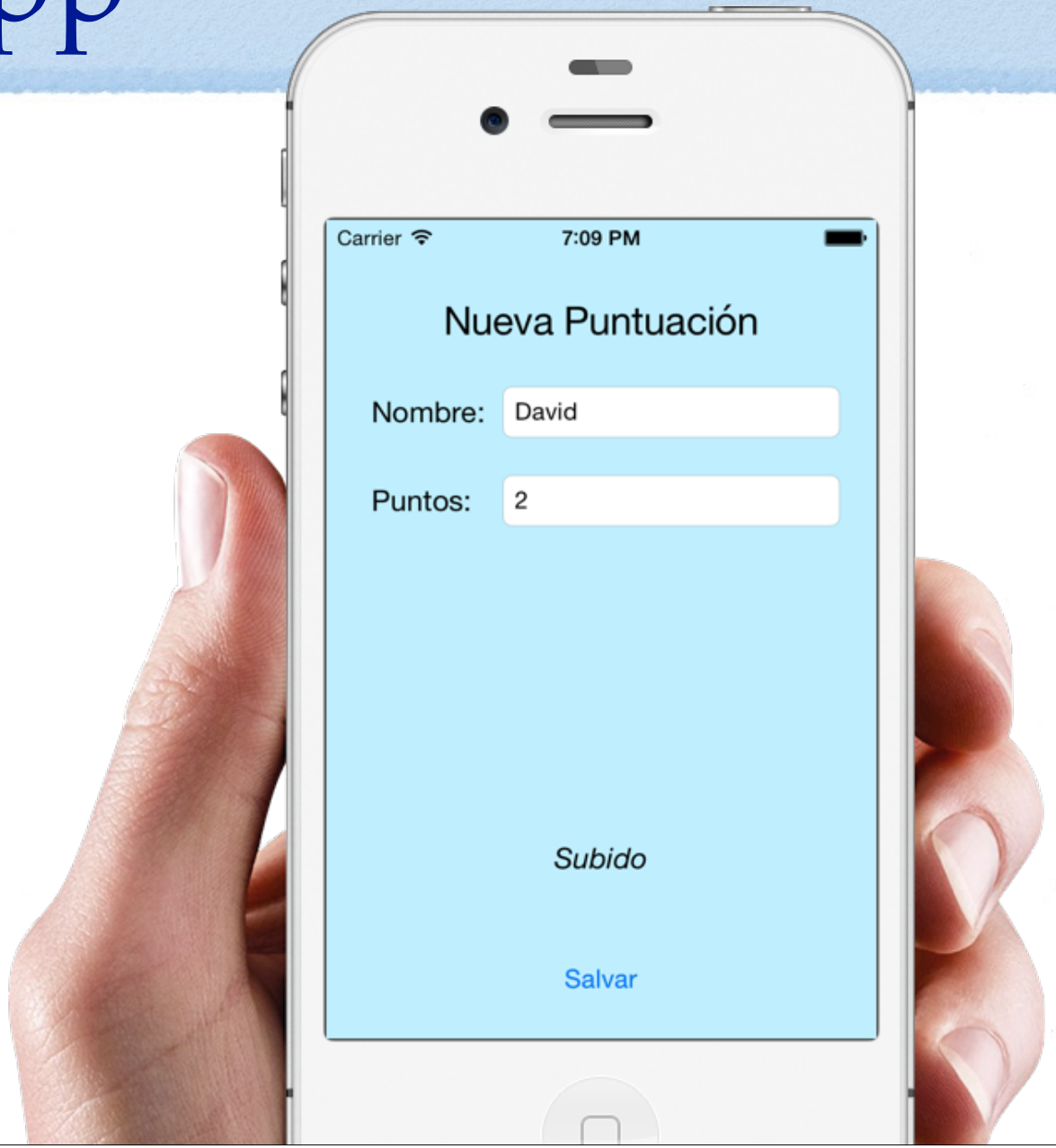
```
    // Crear Data Task
    let dataTask = session.dataTaskWithURL(url,
                        completionHandler: { (data: NSData?,
                                              response: NSURLResponse?,
                                              error: NSError?) in

        var newTop3: [[String:AnyObject]]?

        let res = response as! NSHTTPURLResponse
        if error == nil && res.statusCode == 200 {
            newTop3 = try! NSJSONSerialization.JSONObjectWithData(data!,
                                          options: []) as? [[String:AnyObject]]
        }

        // El completionHandler no corre en el Main Thread
        dispatch_async( dispatch_get_main_queue(), {
            if newTop3 == nil {
                self.title = "ERROR"
            } else {
                self.top3Scores = newTop3!
                self.tableView.reloadData()
                self.title = "Top 3"
            }
            self.refreshButton.enabled = true
            UIApplication.sharedApplication().networkActivityIndicatorVisible = false
        })
    })
    dataTask.resume()
}
```

NSURLSession

Subir Nueva Puntuación

JSON

Nueva App



Creamos un **NSURLSessionUploadTask**

- La **URL** donde subimos es **`http://localhost:3000/scores.json`**
- La petición HTTP:
  - Usa el **método** es **POST**
  - y pone la siguiente cabecera **Content-type**:

```
application/json; charset=utf-8
```

- El dato a subir es un NSData obtenido convirtiendo en JSON un objeto swift:

```
[ "score": [ "name": "Pedro",
             "total": 125 ]
]
```

- Creamos una **NSURLSession** y creamos una tarea **NSURLSessionUploadTask** para subir el NSData anterior.
  - Usamos un completionHandler que se invoca cuando la tarea ha terminado.
  - Inicialmente la tarea está suspendida y tenemos que reanudarla (resume).
  - Devuelve un JSON con todos los campos del registro creado.

```
@IBAction func save() {

    statusLabel.text = "Salvando Puntuación"
    UIApplication.sharedApplication().networkActivityIndicatorVisible = true

    //--- Datos a subir:
    let score = ["score":["name":nameTextField.text,
                          "total":totalTextField.text]]
    guard let dataScore = try? NSJSONSerialization.dataWithJSONObject(score,
                                        options: []) else {
       return
    }

    //--- URL destino:
    let SCORES_URL = "http://localhost:3000/scores.json"
    let url = NSURL(string: SCORES_URL)!

    //--- La peticion HTTP:
    let request = NSMutableURLRequest(URL: url)
    request.HTTPMethod = "POST"
    request.addValue("application/json; charset=utf-8",
            forHTTPHeaderField: "Content-Type")

    //--- La sesion:
    let sessionConf = NSURLSessionConfiguration.ephemeralSessionConfiguration()
    let session = NSURLSession(configuration: sessionConf)
```

```
    //--- La tarea para subir los datos:
    // Nota: el tercer parametro es el completionHandler, pero se esta usando
    // como una Trailing Closure.
    let uploadTask = session.uploadTaskWithRequest(request,
        fromData: dataScore) { (data: NSData?, response: NSURLResponse?,
                                error: NSError?) -> Void in
            let res = response as! NSHTTPURLResponse

            // El completionHandler no corre en el Main Thread
            dispatch_async(dispatch_get_main_queue(), {

                if error != nil {
                    self.statusLabel.text = error!.localizedDescription
                } else if res.statusCode != 200 && res.statusCode != 201 {
                    let code = res.statusCode
                    let msg = NSHTTPURLResponse.localizedStringForStatusCode(code)
                    self.statusLabel.text = msg
                } else {
                    self.statusLabel.text = "Subido"
                }
                UIApplication.sharedApplication().networkActivityIndicatorVisible
                                                                    = false

            })
    }
    uploadTask.resume()
}
```