ETSIT UPM

dit UPM

Desarrollo de Apps para iOS
Acceso a Servicios WEB

IWEB,LSWC 2013-2014
Santiago Pavón

ver: 2013.11.16

Índice

- Soporte proporcionado por Xcode.
- 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 muestre las tres mejores puntuaciones.
  - Paso 1: Desarrollar el GUI de la aplicación usando una TableView.
  - Paso 2: Desarrollar los métodos para descargar los datos.
    - Opción 1: Descargar los datos utilizando el método [NSData dataWithContentsOfURL:].
      - Descargar los datos en formato JSON.
      - Descargar los datos en formato XML.
    - Opción 2: Descargar los datos usando la clase NSURLConnection.
      - Descargar los datos en formato XML.
      - Realizando una conexión síncrona.
      - Realizando una conexión asíncrona.
- Realizar una aplicación que suba nuevas puntuaciones al servidor.
  - Realizando una conexión NSURLConnection síncrona.
- Versión SOAP.

Web Services

¿Qué Soporte Tenemos?

- Disponemos de clases para manejar URLs, Peticiones y Respuestas HTTP, Conexiones, Codificaciones, …
  - NSData dataWithContentsOfURL:
  - NSURLConnection
  - …
- JSON:
  - La clase **NSJSONSerialization** proporciona métodos para serializar y des-serializar JSON.
- XML:
  - Xcode no ofrece soporte para construir documentos XML.
  - Para parsear documentos XML disponemos de la clase **NSXMLParser**.

Desarrollar el Servidor Web Game Scores

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:

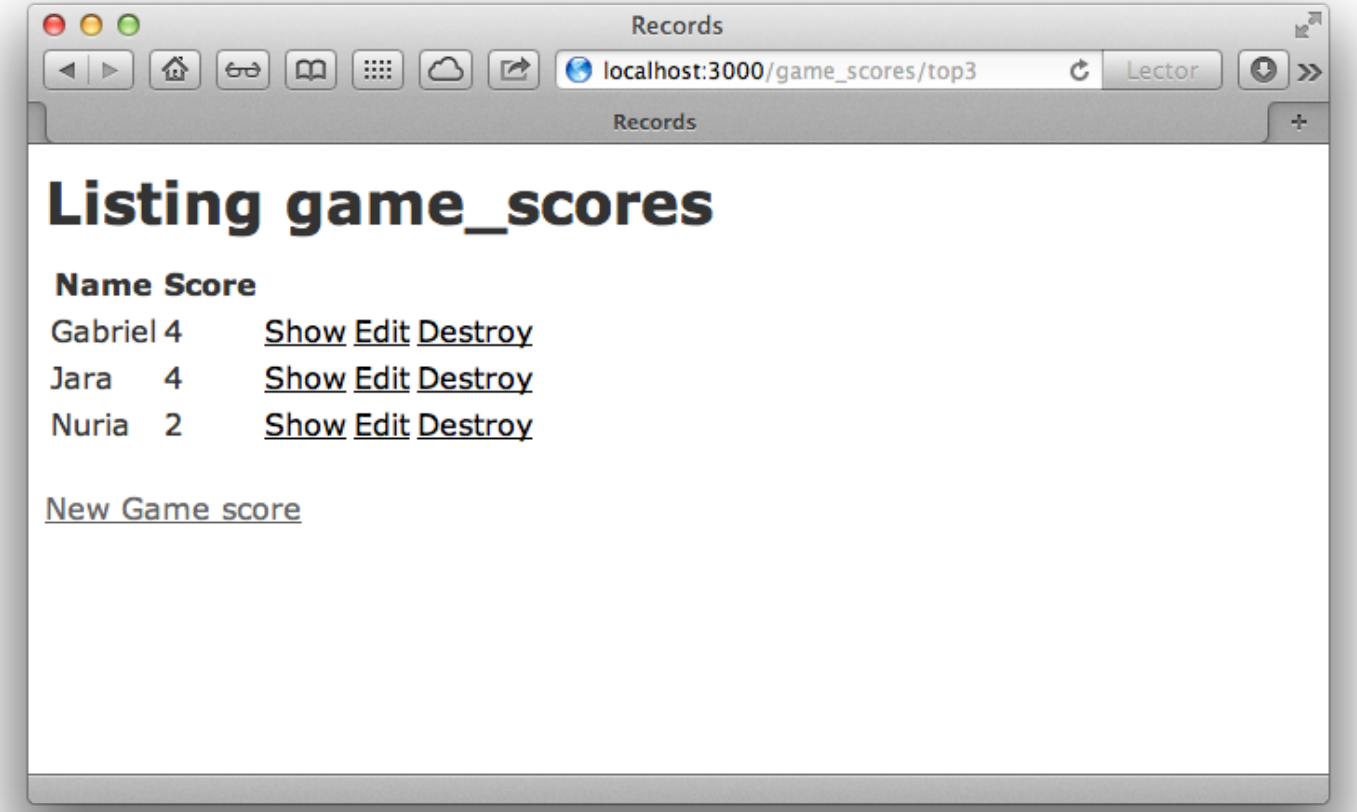
```
http://localhost:3000/scores/top3
```

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

  - Con datos JSON:

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

- Editar un registro:
  - Desde un formulario:

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

  - Con datos JSON:

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

- Borrar un registro:

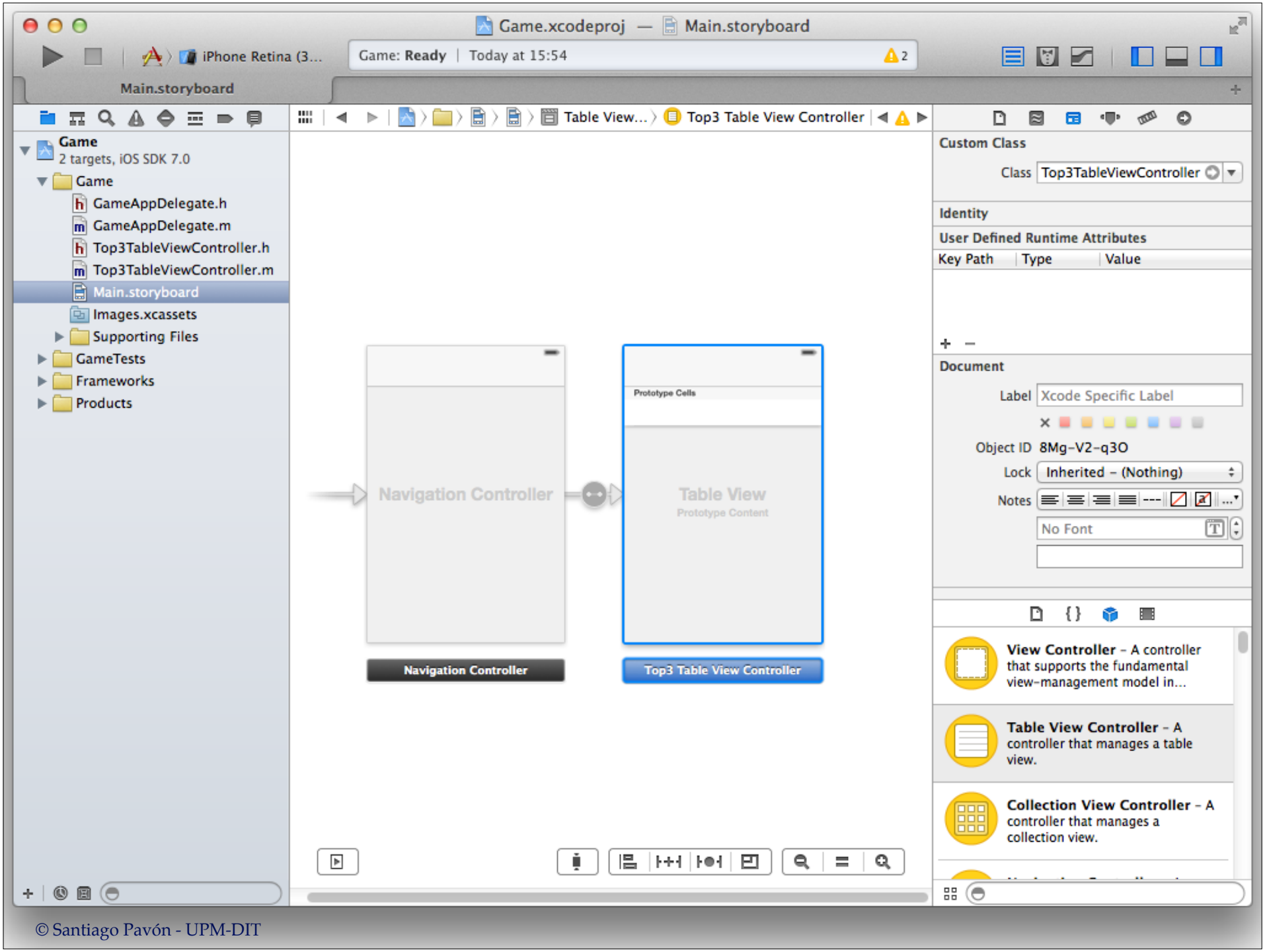
```
$ curl -request DELETE
       http://localhost:3000/scores/11
```

App para Mostrar el Top3

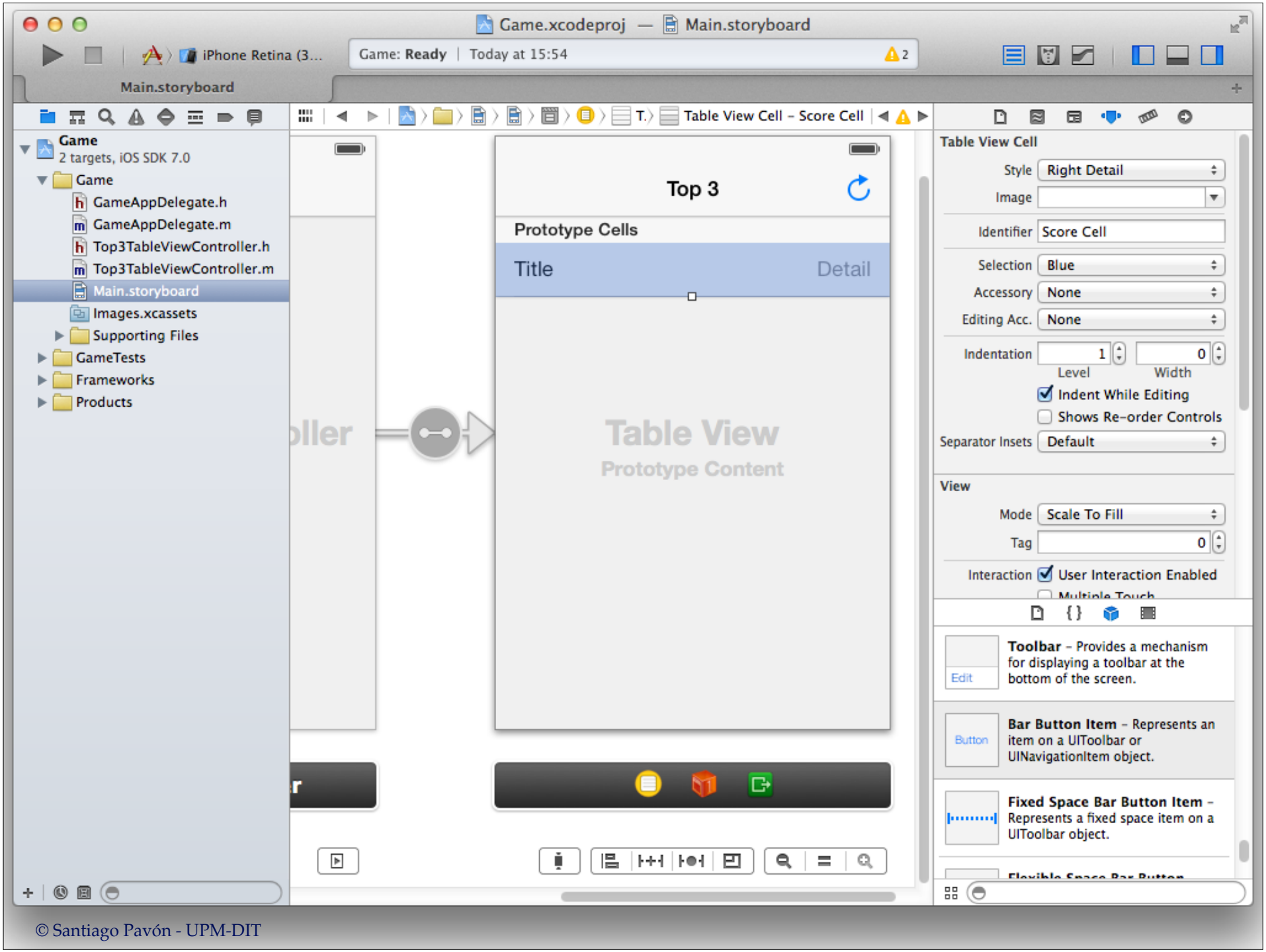
GUI

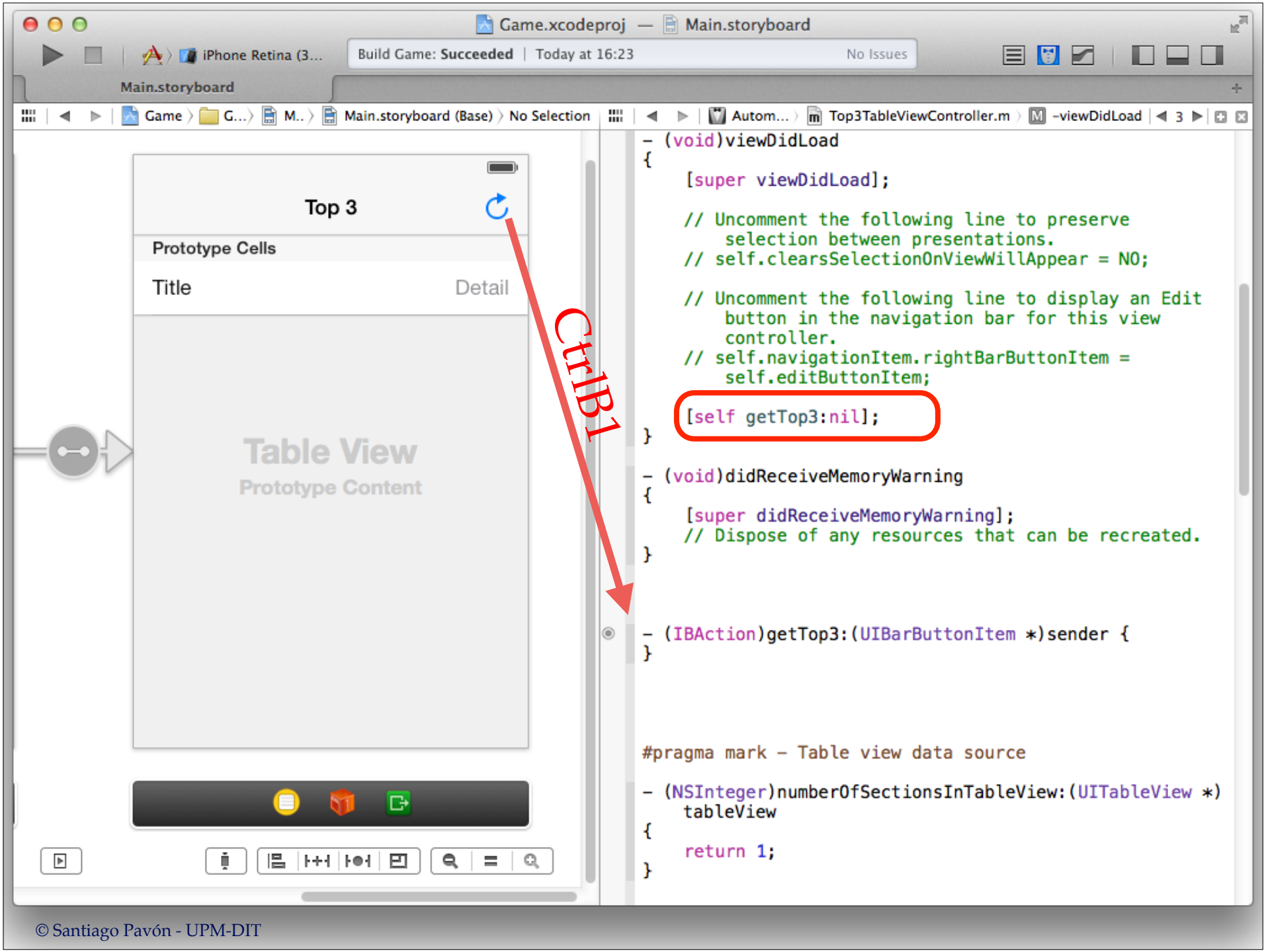
Crear Aplicación: Game

- En esta aplicación mostraremos los datos obtenidos al acceder al recurso top3 de nuestra aplicacion 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.





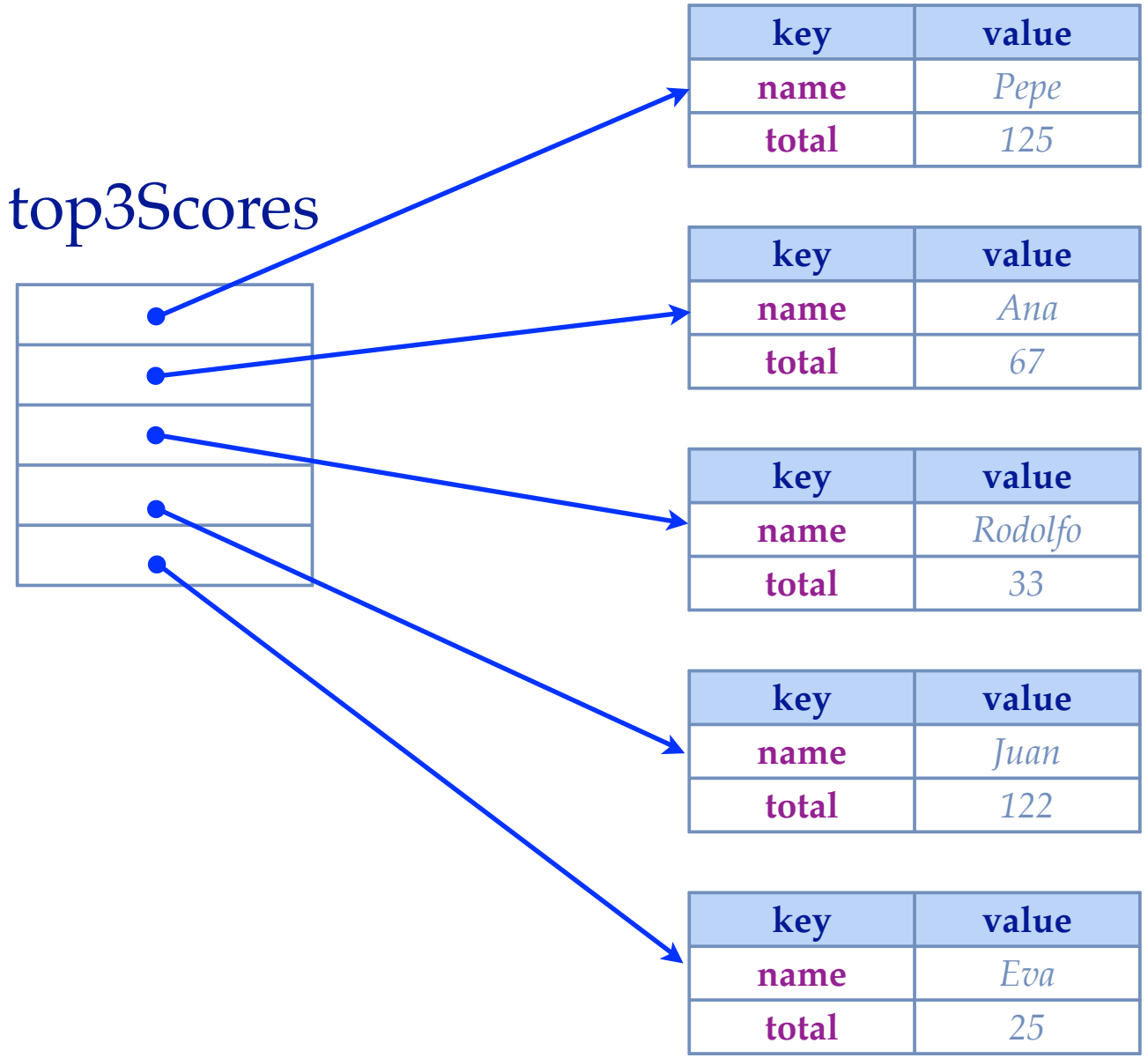
La propiedad top3Scores

- Añadir una propiedad en **Top3TableVC.m** para almacenar los datos descargados.

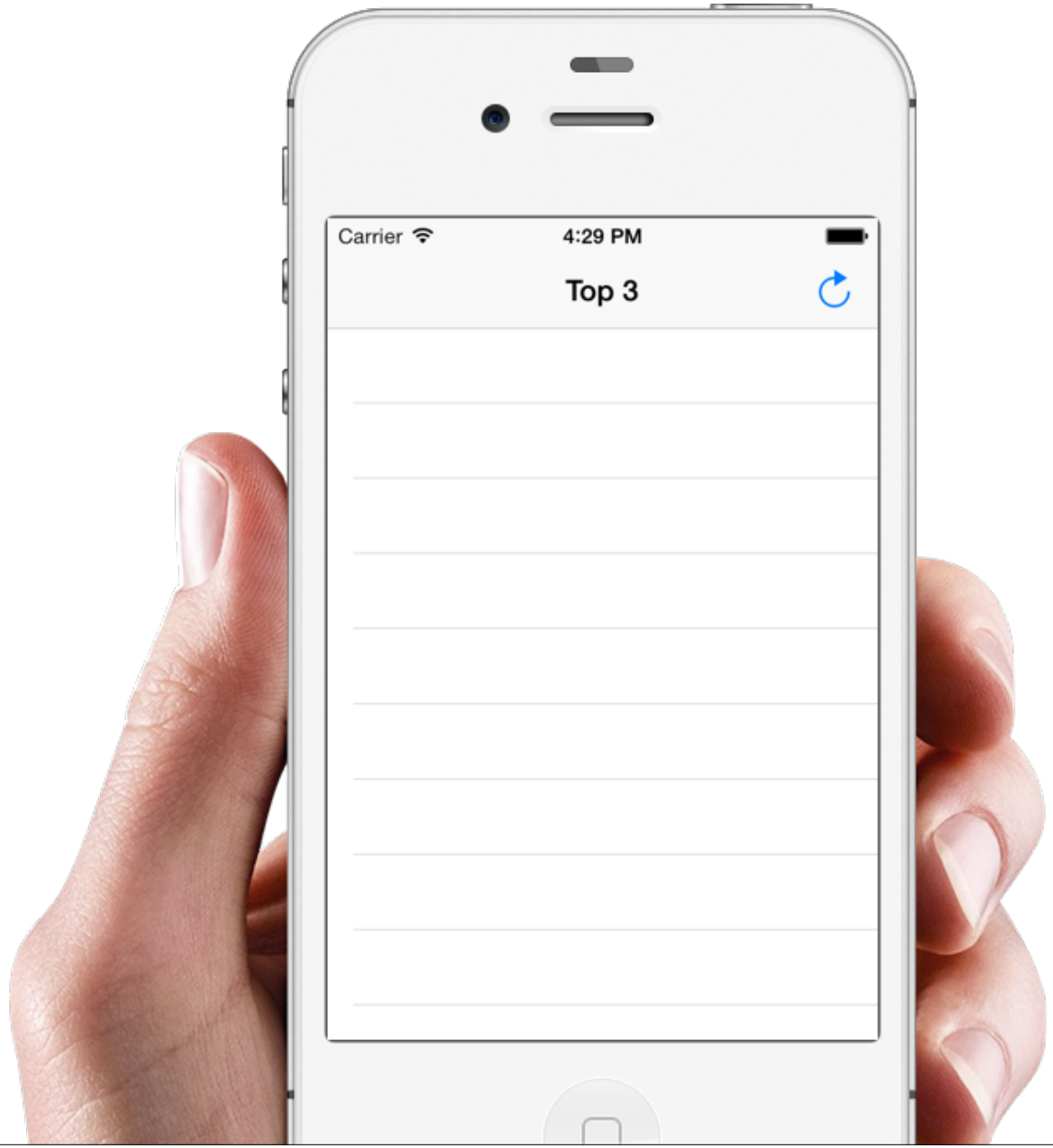
```
@property (nonatomic,strong)
                    NSArray * top3Scores;
```

  - Es un array donde se guardarán los datos descargados del servidor web.
  - 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.



```
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return 1;
}

- (NSInteger) tableView:(UITableView *)tableView
  numberOfRowsInSection:(NSInteger)section
{
    return [self.top3Scores count];
}

- (UITableViewCell *)tableView:(UITableView *)tableView
         cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *CellIdentifier = @"Score Cell";
    UITableViewCell *cell = [tableView
                    dequeueReusableCellWithIdentifier:CellIdentifier
                                         forIndexPath:indexPath];

    NSDictionary *dic = self.top3Scores[indexPath.row];

    cell.textLabel.text       = dic[@"name"];
    cell.detailTextLabel.text = [dic[@"total"] description];

    return cell;
}
```



Descargar y Parsear datos Versión JSON

Descargar JSON

- Para descargar el contenido JSON de una URL:

```
#define GAME_URL "http://localhost:3000/scores/top3.json"


NSString *escapedURL =
        [GAME_URL stringByAddingPercentEscapesUsingEncoding:
                                              NSUTF8StringEncoding];
NSURL *url = [NSURL URLWithString:escapedURL];


NSData *jsonData = [NSData dataWithContentsOfURL:url];
```

  - Obtenemos un buffer con los datos JSON.

Parsear JSON

- La clase **`NSJSONSerialization`** que permite generar y parsear datos JSON.

- Para parsear un buffer (**NSData**) de datos JSON y obtener un diccionario o array con los datos usaremos el método:

```
+ (id)JSONObjectWithData:(NSData*)data
                 options:(NSJSONReadingOptions)opt
                   error:(NSError**)error
```

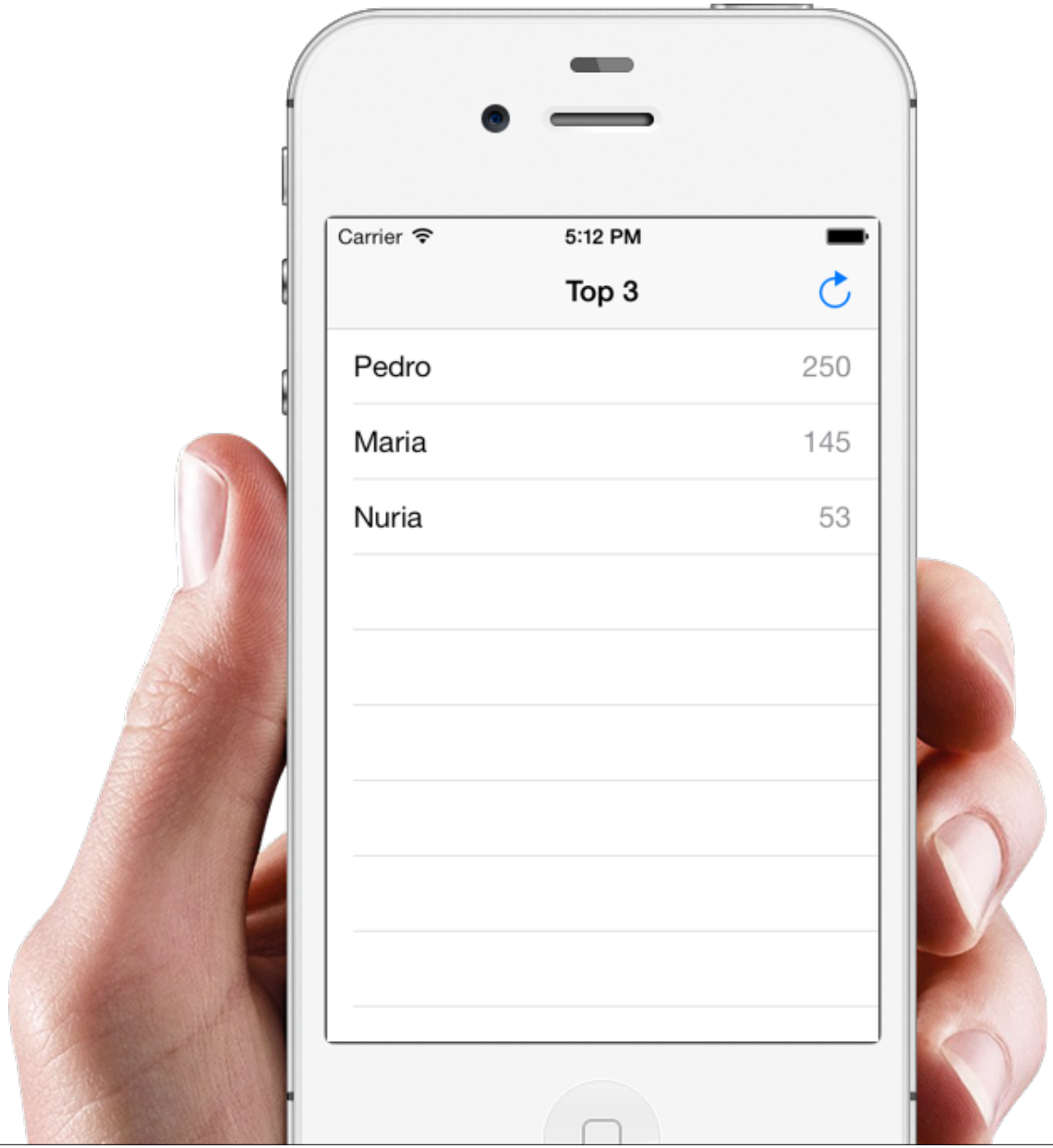
  - Devuelve **nil** se encuentra algún error.

La acción getTop3

```
#define GAME_URL @“http://localhost:3000/scores/top3.json"

- (IBAction)getTop3:(UIBarButtonItem *)sender {

    NSString *escapedURL =
             [GAME_URL stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
    NSURL *url = [NSURL URLWithString:escapedURL];

    NSData *jsonData = [NSData dataWithContentsOfURL:url];

    NSArray * newTop3;
    NSError * err;
    if (jsonData) {
        newTop3 = [NSJSONSerialization JSONObjectWithData:jsonData
                                                  options:0
                                                    error:&err];
    }

    if (!newTop3) {
        NSLog(@"Error parsing JSON = %@",[err localizedDescription]);
        return;
    }

    self.top3Scores = newTop3;

    [self.tableView reloadData];
}
```

La acción getTop3 (GCD)

```
- (IBAction)getTop3:(UIBarButtonItem *)sender {
    self.title = @"Descargando ...";
    self.refreshButton.enabled = NO;  // hay que crear este outlet
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:TRUE];

    dispatch_queue_t queue = dispatch_queue_create("download queue", NULL);
    dispatch_async(queue, ^{
        NSString *escapedURL = [GAME_URL stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
        NSURL *url = [NSURL URLWithString:escapedURL];
        NSData *jsonData = [NSData dataWithContentsOfURL:url];

        NSArray * newTop3;
        NSError * err;
        if (jsonData) {
            newTop3 = [NSJSONSerialization JSONObjectWithData:jsonData
                                                      options:0
                                                        error:&err];
        }
        dispatch_async( dispatch_get_main_queue(), ^{
            if (!newTop3) {
                NSLog(@"Error parsing JSON = %@",[err localizedDescription]);
                self.title = @"Desactualizado";
            } else {
                self.top3Scores = newTop3;
                [self.tableView reloadData];
                self.title = @"Top 3";
            }
            self.refreshButton.enabled = YES;
            [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:FALSE];
        });
    });
}
```



Descargar y Parsear datos Versión XML

NSXMLParser

- Implementación del SAX.
  - SAX = Simple API for XML.
- Lee un documento XML y va informando de los elementos, atributos, comentarios, etc. que encuentra.
  - El parser informa a su delegado de los elementos encontrados.
  - El VC del ejemplo debe ser conforme al protocolo **`NSXMLParserDelegate`**.

Parsear los Datos XML

```
#define GAME_URL @“http://localhost:3000/scores/top3.xml"

- (void) getTop3:(UIBarButtonItem *)sender {
    self.title = @"Descargando ...";
    self.refreshButton.enabled = NO;
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:YES];

    NSString *escapedURL =
        [GAME_URL stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
    NSURL *url = [NSURL URLWithString:escapedURL];

    NSXMLParser * xmlParser = [[NSXMLParser alloc] initWithContentsOfURL:url];

    [xmlParser setDelegate:self];
    [xmlParser setShouldResolveExternalEntities:NO];
    if ([xmlParser parse]) {
        self.title = @"Top 3";
        self.top3Scores = self.top3ScoresNew;
        [self.tableView reloadData];
    } else {
        self.title = @"Error datos incorrectos";
    }

    self.refreshButton.enabled = YES;
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:NO];
}
```

Propiedades usadas en el ejemplo

- Propiedades usadas por los métodos del delegado del **NSXMLParser**:
  - Array con las nuevas puntuaciones recibidas.
    - Contiene un diccionario para cada puntuación ya parseada.

```
NSMutableArray *top3ScoresNew;
```

  - Diccionario para almacenar la puntuación que estoy parseando
    - Las claves son **name** y **total**

```
NSMutableDictionary *scoreNew;
```

  - Clave de la puntuación que estoy parseando

```
NSString *keyNew;
```

  - Valor de la puntuación que estoy parseando

```
NSMutableString *valueNew;
```

Algoritmo

- Cuando llega el comienzo de:
  - **scores** - creo un nuevo array apuntado por top3ScoresNew.
  - **score** - creo en scoreNew un diccionario nuevo para la nueva puntuación.
  - **name** - guardo en keyNew el valor @"name" y los siguientes caracteres que se parseen serán el nombre del jugador para la nueva puntuación.
  - **total** - guardo en keyNew el valor @"total" y los siguientes caracteres que se parseen serán los puntos de la nueva puntuación.
  - en otro caso indico que no hay ni nueva clave ni nuevo valor.
- Cuando llegan caracteres y hay nueva clave (**`keyNew!=nil`**)
  - uso los caracteres para el nuevo valor, guardándolos en **valueNew**.
- Cuando llega el fin de
  - **scores** - ya he terminado, tengo las nuevas puntuaciones en el array top3ScoresNew.
  - **score** - Ya tengo una puntuación nueva en newScore y la guardo en el array top3ScoresNew.
  - **name** o **total** - Ya tengo una nueva pareja clave-valor de la puntuación y la guardo en scoreNew.

```
<scores>   -> Vacío el array

  <score>  -> Creo un diccionario
    <name>       -> Nueva clave es name
         Richard  -> Nuevo valor es Richard
    </name>      -> Meter clave y valor en diccionario

    <total>      -> Nueva clave es total
         666       -> Nuevo valor es 666
    </total>     -> Meter clave y valor en diccionario
     ...
  </score> -> Meter diccionario en array
   ...
</scores>  -> Termine. Refresco UI
```

Encontrado Comienzo de Elemento

```
- (void) parser:(NSXMLParser *)parser
didStartElement:(NSString *)elementName
   namespaceURI:(NSString *)namespaceURI
  qualifiedName:(NSString *)qName
     attributes:(NSDictionary *)attributeDict {

    self.keyNew = nil;
    self.valueNew = nil;

    if ([elementName isEqualToString:@"scores"]) {
        self.top3ScoresNew = [NSMutableArray array];
    } else if ( [elementName isEqualToString:@"score"] ) {
        self.scoreNew = [NSMutableDictionary dictionary];
    } else if ( [elementName isEqualToString:@"total"] ) {
        self.keyNew = @"total";
    } else if ( [elementName isEqualToString:@"name"] ) {
        self.keyNew = @"name";
    }
}
```

Encontrado el Texto de un Elemento

```
- (void) parser:(NSXMLParser *)parser
foundCharacters:(NSString *)string {

    if (self.keyNew) {
        if ( ! self.valueNew) {
            self.valueNew = [NSMutableString string];
        }
        [self.valueNew appendString:string];
    }
}
```

Encontrado Final de Elemento

```
- (void) parser:(NSXMLParser *)parser
  didEndElement:(NSString *)elementName
   namespaceURI:(NSString *)namespaceURI
  qualifiedName:(NSString *)qName {

    if ([elementName isEqualToString:@"scores"]) {
        // ya hemos parseado todo.
    } else if ([elementName isEqualToString:@"score"]) {
        // Guardar nueva puntuacion.
        [self.top3ScoresNew addObject:self.scoreNew];
    } else if ([elementName isEqualToString:@"total"] ||
        [elementName isEqualToString:@"name"])  {
        [self.scoreNew setValue:self.valueNew forKey:self.keyNew ];
    }
    self.keyNew = nil;
    self.valueNew = nil;
}
```

Alternativa: initWithData

- El método que hemos usado:

```
[[NSXMLParser alloc]
        initWithContentsOfURL:url];
```

  - toma como parámetro una URL, e internamente:
    - crea una petición GET HTTP y se baja el XML que va a parsear.

- XMLParser tiene más métodos inicializadores.
  - El inicializador **`initWithData`** toma como parámetro un NSData con el XML a parsear.
    - Este inicializador es útil si obtenemos un NSData con el XML por otros medios.

- El buffer XML lo podemos bajar usando una conexión **NSURLConnection**.
  - NSURLConnection se crea usando un objeto **NSURLRequest**.
    - que nos permite especificar el método HTTP a usar (**DELETE**, **PUT**, ..) y añadir cabeceras HTTP y un body.
  - Devuelve un **NSData** con el XML
- Con los datos bajados, parseamos usando el método:

```
[[NSXMLParser alloc] initWithData:data];
```

Truco: Depuración

```
(NSData *) data = ...

// Mostrar un Log por consola del NSData con el XML recibido
NSString *xml = [[NSString alloc] initWithBytes:[data bytes]
                                          length:[data length]
                                        encoding:NSUTF8StringEncoding];

NSLog(@"XML Recibido = %@",xml);
```

URL Connection

Conectar con el Servidor Web

- Pasos:
  - Crear URL
  - Crear URL Request
    - Poner método, cabeceras, body
  - Establecer la conexión URL
    - puede ser síncrona o asíncrona

NSURLConnection

- Conexión para una petición HTTP a un URL.
- Si la conexión es **asíncrona** se llama a los métodos del delegado:
  - Cuando se recibe la respuesta
  - Cuando van llegando datos
  - Cuando se termina
  - Cuando hay errores
- Si la conexión es **síncrona**, nos quedamos bloqueados hasta que se termina la operación.

Descargar un URL con Conexión Síncrona

```
#define GAME_URL @“http://localhost:3000/scores/top3.json"

- (IBAction)getTop3:(UIBarButtonItem *)sender {
   self.title = @"Descargando ...";
   self.refreshButton.enabled = NO;
   [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:YES];

   NSString *escapedURL =
      [GAME_URL stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
   NSURL *url = [NSURL URLWithString:escapedURL];

   NSURLRequest *req = [NSURLRequest requestWithURL:url];

   NSHTTPURLResponse * responseHTTP = nil;
   NSError * error = nil;
   NSData * data = [NSURLConnection sendSynchronousRequest:req
                                         returningResponse:&responseHTTP
                                                     error:&error];
   if (data) {
      NSInteger statusCode = [responseHTTP statusCode];
      NSLog(@"Codigo de respuesta = %d", statusCode);
      if ( statusCode != 200 ) {
         self.title = @"Error descargando Top 3";
      } else { //--- Parsear el JSON recibido ---
         [self parseTop3:data];
      }
   } else {  // No pudo realizarse la conexion o la descarga fallo.
      self.title = @"Error descargando Top 3";
      NSLog(@"Error Cargando Top 3: %@",[error localizedFailureReason]);
   }

   self.refreshButton.enabled = YES;
   [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:FALSE];
}
```

```
- (void)parseTop3:(NSData *)data {

    self.title = @"Analizando Top 3";

    NSArray * newTop3;
    NSError * err;
    if (data) {
        newTop3 = [NSJSONSerialization JSONObjectWithData:data
                                                  options:0
                                                    error:&err];
    }

    if (!newTop3) {
        NSLog(@"Error parsing JSON = %@",[err localizedDescription]);
        self.title = @"Desactualizado";
    } else {
        self.top3Scores = newTop3;
        [self.tableView reloadData];
        self.title = @"Top 3";
    }
    self.refreshButton.enabled = YES;
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:FALSE];
}
```

Descargar un URL con Conexión Asíncrona

Necesitamos dos Propiedades

- Objeto que gestiona la conexión HTTP.

```
@property (nonatomic,strong)
        NSURLConnection *urlConnection;
```

- Buffer donde almacenamos los datos recibidos.

```
@property (nonatomic,strong)
          NSMutableData *responseData;
```

```
- (IBAction) getTop3:(UIBarButtonItem *)sender {

    self.title = @"Descargando ...";
    self.refreshButton.enabled = NO;
    [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:YES];

    NSString *escapedURL =
       [GAME_URL stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
    NSURL *url = [NSURL URLWithString:escapedURL];

    NSURLRequest *req = [NSURLRequest requestWithURL:url];

    self.urlConnection = [[NSURLConnection alloc] initWithRequest:req
                                                         delegate:self];

    if (self.urlConnection) {
       self.responseData = [NSMutableData data];
    } else {
       self.title = @"Error de Conexion";
       self.refreshButton.enabled = YES;
       [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:NO];
    }
}
```

Métodos Delegados

- El VC de la app será el delegado de **NSURLConnection**.
  - Tiene que ser conforme al protocolo **`NSURLConnectionDelegate`**.

- Metodos del delegado:
  - `connection:canAuthenticateAgainstProtectionSpace:`
  - `connection:didCancelAuthenticationChallenge:`
  - `connection:didReceiveAuthenticationChallenge:`
  - `connectionShouldUseCredentialStorage:`
  - `connection:willCacheResponse:`
  - **`connection:didReceiveResponse:`**
  - **`connection:didReceiveData:`**
  - `connection:didSendBodyData:totalBytesWritten:`
    `totalBytesExpectedToWrite:`
  - `connection:willSendRequest:redirectResponse:`
  - **`connection:didFailWithError:`**
  - **`connectionDidFinishLoading:`**

Falla la Conexión

```
- (void) connection:(NSURLConnection *)connection
   didFailWithError:(NSError *)error {

   NSLog(@"Error de conexion = %@",[error localizedFailureReason]);

   self.title = @"Error Cargando Top 3";
   self.refreshButton.enabled = YES;
   [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:YES];

   self.urlConnection = nil;
   self.responseData = nil;
}
```

LLega la Respuesta

```
- (void) connection:(NSURLConnection *)connection
 didReceiveResponse:(NSURLResponse *)response {

   NSInteger code = [(NSHTTPURLResponse *)response statusCode];
   NSLog(@"Codigo de respuesta = %d", code);
   if ( code != 200 ) {
      self.title = @"Error Cargando Top 3";
      self.refreshButton.enabled = YES;
      [[UIApplication sharedApplication] setNetworkActivityIndicatorVisible:NO];

      [connection cancel];
      self.urlConnection = nil;
      self.responseData = nil;
   } else {
      [self.responseData setLength:0];
   }
}
```

- Pueden producirse varias llamadas a el método didReceiveResponse.
  - Si el tipo de datos es **multipart/x-mixed-replace**
- En estos casos hay que descartar todos los datos recibidos anteriormente, y prepararse para recibir nuevos datos.
- Ver documentación para detalles.

LLegan Datos

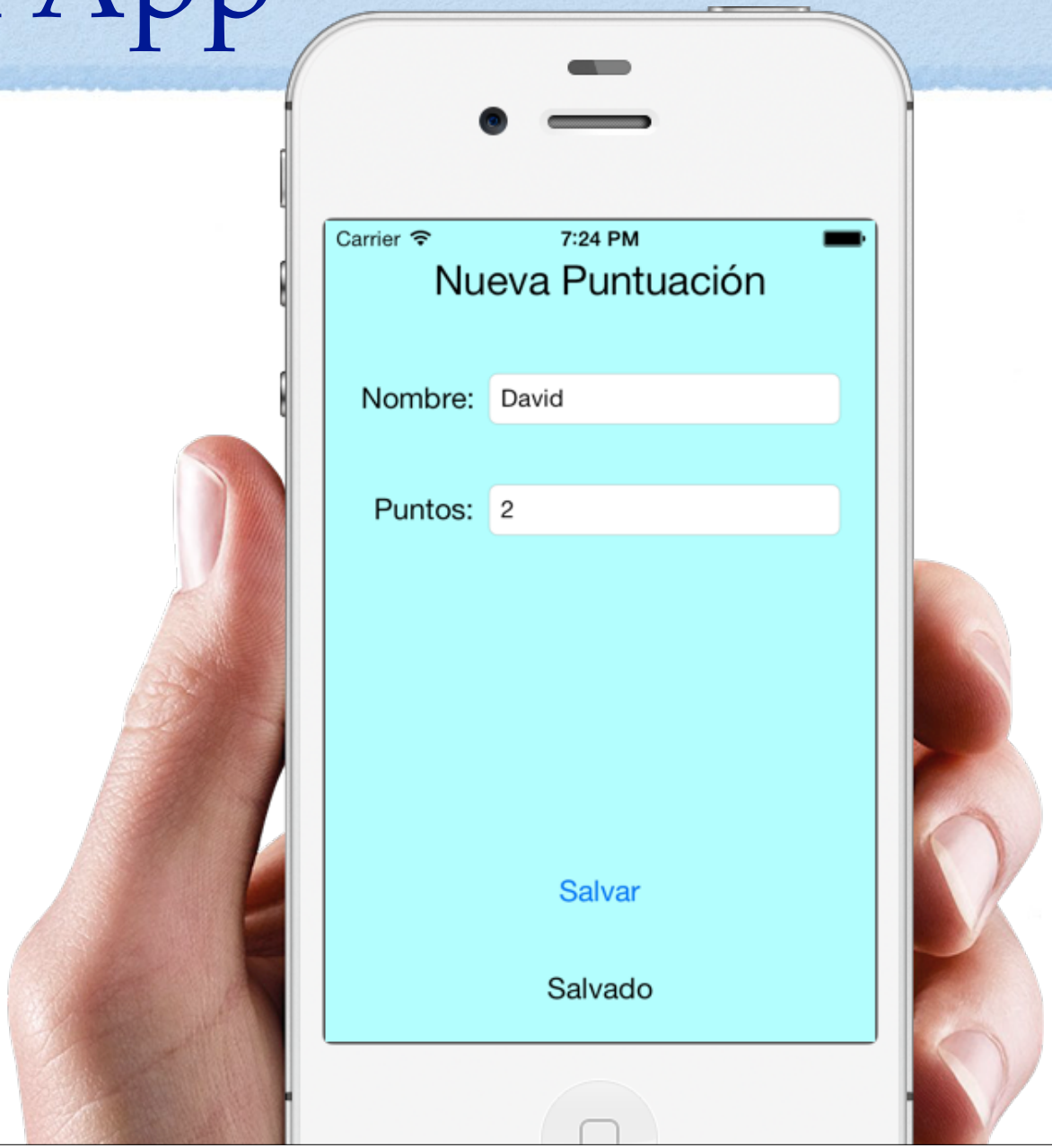
```
- (void) connection:(NSURLConnection *)connection
     didReceiveData:(NSData *)data {

   [self.responseData appendData:data];
}
```

Terminó la Descarga

```
- (void) connectionDidFinishLoading:
                              (NSURLConnection *)connection {

    //--- Parsear el JSON ---
    [self parseTop3:self.responseData];

    self.urlConnection = nil;
    self.responseData = nil;

}
```

Guardar Nueva Puntuación JSON

Nueva App



Petición HTTP

- La **URL** es `http://localhost:3000/scores.json`
- El **método** es POST
- La cabecera **Content-type** es:

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

- El **body** es:

```
{
  "name": "Pedro"
  "total": 125
}
```

- Devuelve un JSON con todos los campos del registro creado.

```
#define SCORES_URL @“http://localhost:3000/scores.json"

NSString *score = [NSString stringWithFormat:
       @"{\"name\":\"%@\",\"total\":%d}",
       @”Pedro”, 125];

NSString *escapedURL =
      [SCORES_URL stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
NSURL *url = [NSURL URLWithString:escapedURL];

NSMutableURLRequest *req = [NSMutableURLRequest requestWithURL:url];

[req setHTTPMethod:@"POST"];

[req addValue:@"application/json; charset=utf-8"
     forHTTPHeaderField:@“Content-Type"];

NSString *length = [NSString stringWithFormat:@"%d", [score length]];
[req addValue:length forHTTPHeaderField:@"Content-Length"];

[req setHTTPBody:[score dataUsingEncoding:NSUTF8StringEncoding]];

NSURLResponse * response = nil;
NSError * error = nil;
NSData * data = [NSURLConnection sendSynchronousRequest:req
                                      returningResponse:&response
                                                  error:&error];
```

Guardar Nueva Puntuación XML

Petición HTTP

- La **URL** es `http://localhost:3000/scores.xml`
- El **método** es POST
- La cabecera **Content-type** es:

```
text/xml; charset=utf-8
```

- El **body** es:

```
<score>
    <name>Pedro</name>
    <total>125</total>
</score>
```

- Devuelve XML con todos los campos del registro creado.

```
#define SCORES_URL @“http://localhost:3000/scores.xml"


NSString *score = [NSString stringWithFormat:
       @"<score><name>%@</name><total>%d</total></score>",
       @”Pedro”, 125];

NSString *escapedURL =
      [SCORES_URL stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
NSURL *url = [NSURL URLWithString:escapedURL];

NSMutableURLRequest *req = [NSMutableURLRequest requestWithURL:url];

[req setHTTPMethod:@"POST"];

[req addValue:@"text/xml; charset=utf-8" forHTTPHeaderField:@"Content-Type"];

NSString *length = [NSString stringWithFormat:@"%d", [score length]];
[req addValue:length forHTTPHeaderField:@"Content-Length"];

[req setHTTPBody:[score dataUsingEncoding:NSUTF8StringEncoding]];

NSURLResponse * response = nil;
NSError * error = nil;
NSData * data = [NSURLConnection sendSynchronousRequest:req
                                      returningResponse:&response
                                                  error:&error];
```

Y si fuera SOAP

Formato Petición SOAP 1.1

```
POST /tops3.asmx HTTP/1.1
Host: www.ecubicle.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://localhost:3000/webservices/Top3AsXml"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Top3AsXml xmlns="http://localhost:3000/webservices/"/
>
  </soap:Body>
</soap:Envelope>
```

Formato Respuesta SOAP 1.1

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Top3AsXmlResponse
      xmlns="http://localhost:3000/webservices/">
      <Top3AsXmlResult>xml result</Top3AsXmlResult>
    </Top3AsXmlResponse>
  </soap:Body>
</soap:Envelope>
```

Codificación y Escapado

Codificación y Escapado

- Por motivos de interoperabilidad, al manejar URLs y el protocolo HTTP, debemos **codificar** y **escapar** algunos datos.
  - Las RFCs 1808, 1738 y 2732 especifica como son las URLs.
    - Hay una serie de reglas para su codificación:
      - Solo pueden usarse unos cuantos caracteres ASCII (letras, números y algunos signos) en una URL, y los caracteres conflictivos debe escaparse.
      - Los caracteres (`$&+,/::=?@`) que tienen un significado especial en la sintaxis de la URL deben escaparse cuando se usan con otro significado.
      - …
    - Los caracteres de la URL se codifican en UTF-8, y los bytes que no son letras o números ASCII, o tienen un significado conflictivo, se sustituyen por un % seguido de dos dígitos hexadecimales (su código ASCII).
      - Ejemplo: **`El Camión`** -> **`El%20Cami%C3%B3n`**
  - En los datos de formularios se sustituyen los espacios en blanco por + (o por %20).
  - Los valores de algunas cabeceras de las peticiones y respuestas HTTP, también se codifican en UTF-8, Base-64,…

- En Xcode tenemos algunas clases y métodos para realizar estas tareas de codificación y escapado.

- Ya sabemos que:
  - Un NSString es una secuencia de caracteres Unicode.
  - Un NSData es una secuencia de bytes.
- Para convertir un NSString en una secuencia de bytes UTF-8:

```
NSString *str = @“El Camión”;
NSData * data = [str dataUsingEncoding:NSUTF8StringEncoding];
```

- Para reconstruir un NSString desde un NSData codificado en UTF-8:

```
NSString *str2 = [[NSString alloc] initWithData:data
                                       encoding:NSUTF8StringEncoding];
```

- Para construir una URL escapando con % y codificando en UTF-8:

```
NSString *origUrl = @“http://dominio/ruta?query”;
NSString *escapedURL = [origUrl
                  stringByAddingPercentEscapesUsingEncoding:
                                              NSUTF8StringEncoding];
NSURL *url = [NSURL URLWithString:escapedURL];
```

- **Codificar** en Base-64:
  - Crear un NSString codificado en Base-64 desde un NSData:

```
NSString *str64 = [data base64EncodedStringWithOptions:0];
```

  - Crear un NSData codificado en Base-64 y UTF-8 desde un NSData:

```
NSData *data64 = [data base64EncodedDataWithOptions:0];
```

- **Decodificar** en Base-64:
  - Crear un NSData desde un NSData codificado en Base-64 y UTF-8:

```
NSData *data = [[NSData alloc]
                    initWithBase64EncodedData:data64
                                      options:0];
```

  - Crear un NSData desde un NSString codificado en Base-64:

```
NSData *data = [[NSData alloc]
                    initWithBase64EncodedString:str64
                                        options:0];
```

- Consultar la documentación para ver las opciones de codificación y descodificación en Base-64.