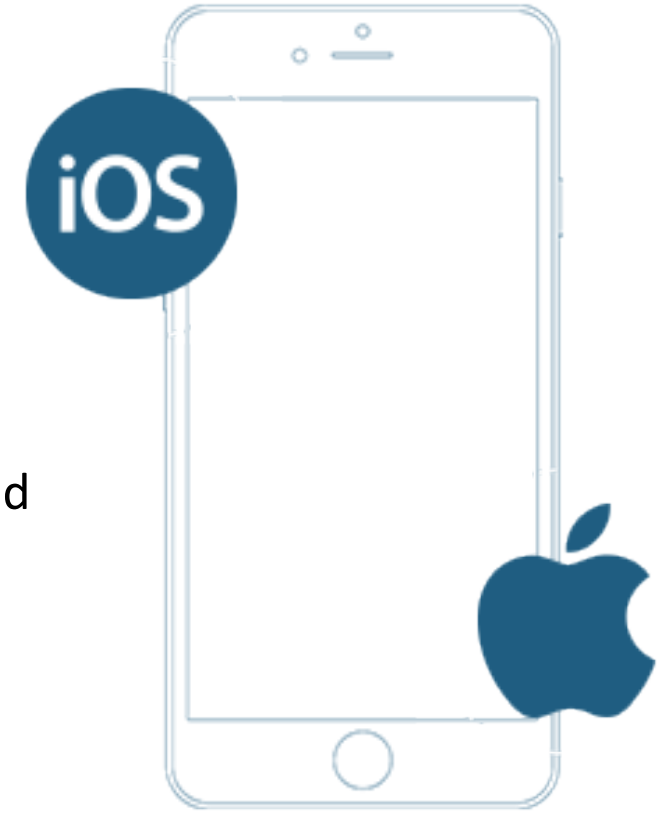
Developing IOS apps using objective-c

Presented By
Abdelrahman Sayed



Java™ Education and Technology Services

Objective-C

Java™ Education and Technology Services

Lecture 5

Objective-C

Agenda

- Networking.
- Synchronous Communication.
- Asynchronous Communication.
- Web Services.
- UITableView inside UIViewController

Networking

Networking

- Using networking in mobile applications is very essential due to the need of storing and retrieving data from / to a remote host.
- Remote hosts are used due to:
  - Providing security.
  - Data backup.
  - Heavy processing on data is much better on a remote host.
  - When data is needed to be shared with users, this data should be saved on a server.

iOS Basic Networking

- As iOS was built on UNIX kernel, so it supports networking and sockets.
- Sockets is the fundamental network programming interface on iOS; all of the higher-level frameworks are based on it.
- It is a good choice for maximum performance and flexibility.
- In iOS development there are many higher implementations for networking that minimizes our work for accessing resources, sending and receiving data over network connections.

Synchronous Communication

Main Thread

- Any application should have at least one thread called the main thread.
- Main thread is the thread responsible for:
  - Main application actions, including the GUI response.
- It is wrong to carry out long operation in the main thread as it will result in blocking the UI.

Downloading

- If you have a URL of file containing some text, and you need to load this data to your application
- In one step you can simply use:
  - stringWithContentsOfURL method

```
NSString *myData = [[NSString alloc] initWithContentsOfURL:url encoding:
                                     NSUTF8StringEncoding error:nil];
```

- But doing this will cause blocking of UI as it is done on the main thread

Asynchronous Communication

Asynchronous Communication

- In order to access web resources asynchronously , the combination of these classes are used:
  - NSURLRequest
  - NSURLConnection

NSURLRequest

- It is used to define the parameters of the request such as:
  - URL
  - Timeout
  - Encoding

- After creating the request , it should be sent to NSURLConnection object to execute it.

NSURLRequest Methods

- The following methods are used to create and return a URL request from a given NSURL

```
NSURLRequest *request = [NSURLRequest requestWithURL:url];
```

```
NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];
```

NSURL Methods

- NSURLs are created using the following methods

```
NSURL *url = [[NSURL URLWithString:@"jets.iti.gov.eg"];
```

```
NSURL *url = [[NSURL alloc] initWithString:@"jets.iti.gov.eg"];
```

NSURLConnection

- Objects from this class represent the connection to some server.
- It is used to execute the given requests.
- Connection class has a delegate which enables the developer to carry on tasks upon some actions such as:
  - Request sent
  - Result received
  - Connection error

NSURLConnection Methods

- Creates and returns an initialized URL connection and begins to load the data for the URL request.

```
NSURLConnection *connection = [[NSURLConnection alloc] initWithRequest:request
                                                       delegate:self];
```

- Causes the connection to begin loading data, if it has not already.

```
[connection start];
```

NSURLConnection Methods Cont.

- Sent as a connection loads data

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

- Sent when a connection fails

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

NSURLConnection Methods Cont.

- Sent as a connection loads data

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

- Sent when a connection fails

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

NSURLConnection Methods Cont.

- Sent after connection start

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

- Sent after a successful connection

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

NSURLConnectionData And NSURLConnection Delegate

- To handle asynchronous connection you should conform to:
  - NSURLConnectionDelegate
  - NSURLConnectionDataDelegate

Networking Demo

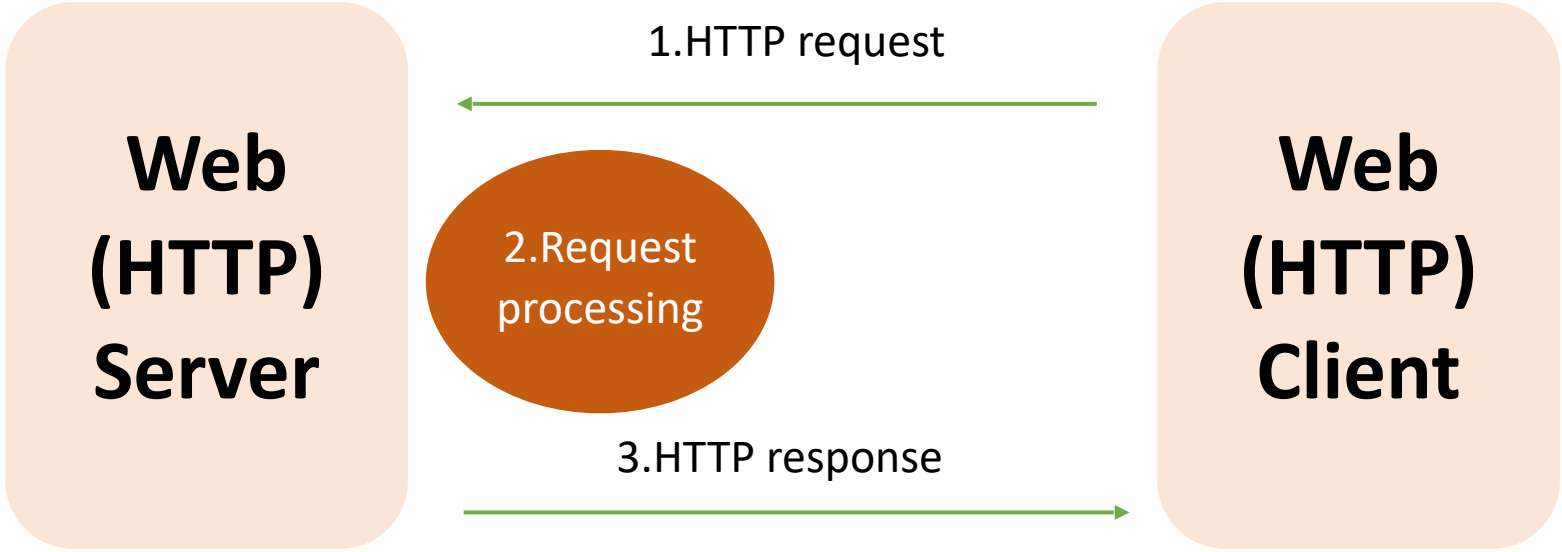
Java™ Education and Technology Services

Web Services

Objective-C

Web Access

- How it works:
  1. Web client sends to the server HTTP request
  2. The web server processes the request
  3. HTTP response is sent to the client



Web Service

- It is a web application without interface (No HTML code)
- Web service contains PHP, ASP or JSP files that returns data
- Data might be formatted as:
  - XML
  - JSON
  - Text

XML

- XML Response
  - No matter what kind of request you are sending, your response will always be binary data
  - Parsing this data should be performed

XML Example

```
<track name = “MADA”>

        <student name=”Ahmed” id=20 />

        <student name=”Mohamed” id=21 />

</track>
```

JSON

- JSON Request (HTTP Request)
  - The same as XML However: creating a JSON object and sending it with the request as service's parameters might be needed
- JSON Response (special data format)
  - The same as XML Remember (It's all about parsing)

JSON Example

- JSON could have one of three format:
  - Array
    - ➢[“first” , ”second” , “third”]
  - HashMap
    - ➢{“key” : ”value” , “key” : “value”}
  - Mix
    - ➢{“JavaTracks”:[“EWD”, “MAD”,“MADA”]}

Webservice Demo

Java™ Education and Technology Services

UITableView inside UIViewController

Objective-C

UITableView inside UIViewController

- You can create UITableView inside UIviewController and make it one of its componnets like UILabel, UITextField, UIImageView and so on.

UITableView inside UIViewController Cont.

Steps:

1. Drag & drop UITableView on your UIViewController.

UITableView inside UIViewController Cont.

2. In your view controller subclass, declare your intention to implement the appropriate delegate and data source protocol:

```
<UITableViewDelegate , UITableViewDataSource>
```

3. In the implementation file of your view controller, implement the methods you've defined.

UITableView inside UIViewController Cont.

4. Finally, in Interface Builder (or programmatically) set both the delegate and dataSource outlets of the table view to be equal to its superview's view controller (in IB, this view controller is File's Owner).

UITableView Demo

Lab Exercise

1.Synchronous & Asynchronous

- Create application which can retrieve data from https://www.facebook.com and display its contents in your application in UIWebView.

Use both Synchronous & Asynchronous connection

2.Register using Web Service

- Connect to the following Web Service and parse the returned JSON object

https://dummyjson.com/products

- Finally show the result in alert with ok button in the case of SUCCESS.
- In the case of FAILING you will show alert with Ok and Try again buttons:
  - Ok will dismiss the alert.