Communications

Mobile Application Development in iOS

School of EECS

Washington State University

Instructor: Larry Holder

Outline

- Already seen
  - MapKit LocalSearch, Directions
- Safari Services and WebKit
- HTTP requests
- APIs
- Other communications services

Safari Services

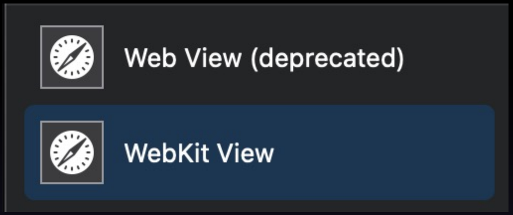
- Full browser functionality within app
- Import SafariServices
- Create URL
- Create SFSafariViewController (URL)
- Execute present (ViewController)

Safari Services

```swift
import SafariServices

class ViewController: UIViewController {

    let urlString = "https://school.eecs.wsu.edu"

    @IBAction func safariTapped(_ sender: UIButton) {
        let url = URL(string: urlString)
        let safariVC = SFSafariViewController(url: url!)
        present(safariVC, animated: true, completion: nil)
    }
}
```

WebKit

- View to display web content
  - Programmatic browser functions
- Web View vs. WebKit View
  - UIWebView deprecated; only choice before iOS 8
  - WKWebView only programmatically for iOS 8-10
    - Storyboard version works for iOS 11+
  - Use WKWebView
- Import WebKit

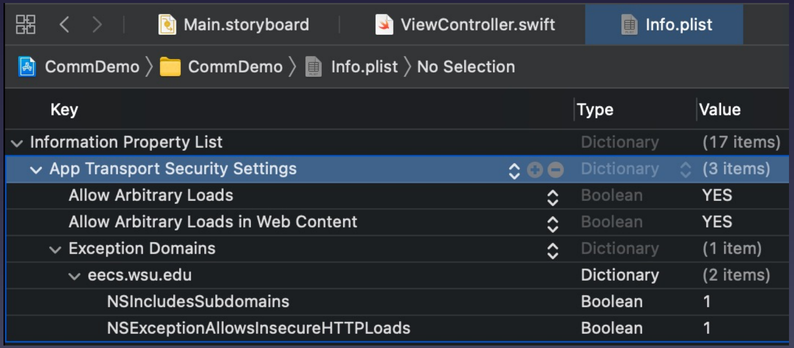


Web Kit View

```
import WebKit

class ViewController: UIViewController {

    let urlString = "https://school.eecs.wsu.edu"

    @IBOutlet weak var webView: WKWebView!

    @IBAction func webViewTapped() {
        let url = URL(string: urlString)
        let request = URLRequest(url: url!)
        webView.load(request)
    }
}
```

Web Security

- App Transport Security
  - Only HTTPS by default
- App Transport Security Settings
  - Allow Arbitrary Loads (iOS 9 or earlier)
  - Allow Arbitrary Loads in Web Content (iOS 10 or later)
  - Exception Domains
    - Dictionary for each domain
      - NSIncludesSubdomains
      - NSExceptionAllowsInsecureHTTPLoads



Key	Type	Value
Information Property List	Dictionary	(17 items)
App Transport Security Settings	Dictionary	(3 items)
Allow Arbitrary Loads	Boolean	YES
Allow Arbitrary Loads in Web Content	Boolean	YES
Exception Domains	Dictionary	(1 item)
eecs.wsu.edu	Dictionary	(2 items)
NSIncludesSubdomains	Boolean	1
NSExceptionAllowsInsecureHTTPLoads	Boolean	1

URL Sessions

- API for web data transfer tasks
  - Numerous delegates to monitor and control transfer
- URLSession.shared singleton for simple tasks
  - No delegates
- Create your own URLSession for more complex data transfer tasks
  - Allows assignment of various delegates

HTTP Requests

- Using URLSession.shared instance
- Create data task
  - URLSession.shared.dataTask(with: URL, completionHandler: @escaping (Data?, URLResponse?, Error?) -> Void)
  - URLSession.shared.dataTask(with: URLRequest, completionHandler: @escaping (Data?, URLResponse?, Error?) -> Void)
- Call resume() on data task

HTTP Requests

- Using new URLSession instance
  - myURLSession.dataTask(with: URL)   // calls delegates
  - myURLSession.dataTask(with: URLRequest)   // calls delegates
- Delegates
  - URLSessionDelegate
  - URLSessionTaskDelegate
  - URLSessionDataDelegate
  - URLSessionDownloadDelegate
  - URLSessionStreamDelegate
- Methods
  - didReceive, didFinish, …

HTTP Requests

```
let dateURLString = "https://eecs.wsu.edu/~holder/tmp/datetime.php"

func getDateTimeFromServer() {
    let url = URL(string: dateURLString)
    let dataTask = URLSession.shared.dataTask(with: url!,
                 completionHandler: handleResponse)
    dataTask.resume()
}
```

HTTP Requests: Server Side

```php
<?php

// datetime.php – return current date and local time in JSON format

date_default_timezone_set("America/Los_Angeles");
$myDate = date("Y-m-d");
$myTime = date("H:i:s");
$json = '{"date":"' . $myDate . '","time":"' . $myTime . '"}';
print $json;

?>
```

Handle HTTP Response (Yhprum's edition)

```
func handleResponse (data: Data?, response: URLResponse?, error: Error?) {
    let dataStr = String(data: data!, encoding: .utf8)
    print("success: response = \(dataStr!)")
}
```

Handle HTTP Response (Murphy's edition)

```swift
func handleResponse (data: Data?, response: URLResponse?, error: Error?) {
      // 1. Check for error in request (e.g., no network connection)
      if let err = error {
            print("error: \(err.localizedDescription)")
            return
      }
      // 2. Check for improperly-formatted response
      guard let httpResponse = response as? HTTPURLResponse else {
            print("error: improperly-formatted response")
            return
      }
      let statusCode = httpResponse.statusCode
      // 3. Check for HTTP error
      guard statusCode == 200 else {
            let msg = HTTPURLResponse.localizedString(forStatusCode: statusCode)
            print("HTTP \(statusCode) error: \(msg)")
            return
      }
      // 4. Check for no data
      guard let somedata = data else {
            print("error: no data")
            return
      }
      // 5. Check for improperly-formatted data
      guard let dataStr = String(data: somedata, encoding: .utf8) else {
            print("error: improperly-formatted data")
            return
      }
      // 6. Everything seems okay
      print("success: response = \(dataStr)")
}
```

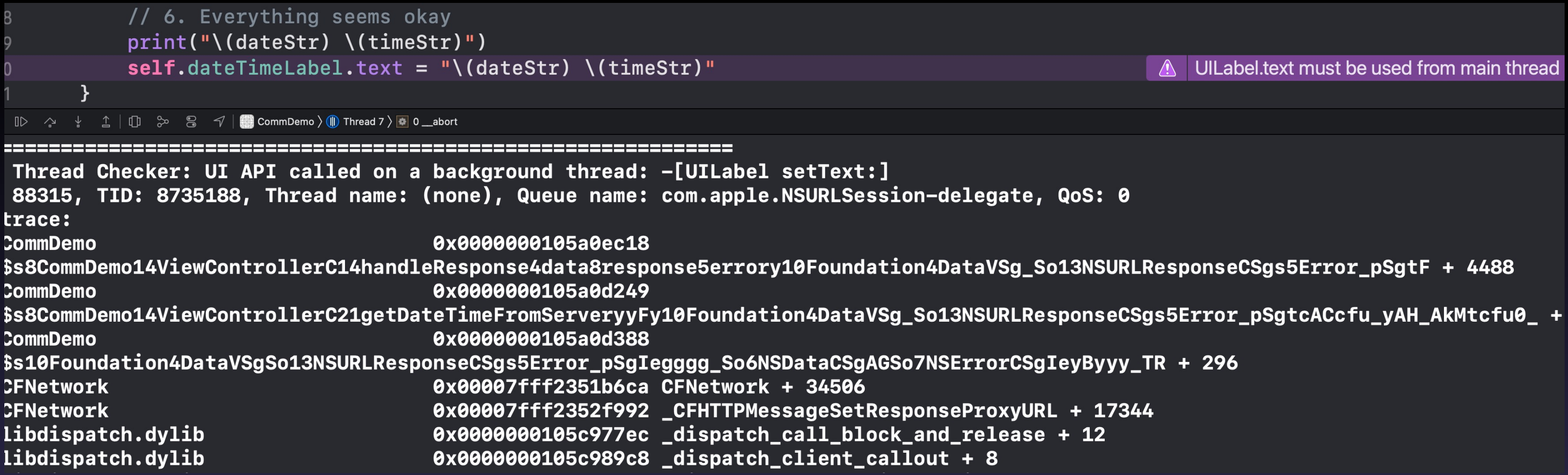
Handling JSON Responses

- If JSON data, then use JSONSerialization

```
func handleResponse (data: Data?, response: URLResponse?, error: Error?) {
    ...
    // 5. Check for properly-formatted JSON data
    guard let jsonObj = try? JSONSerialization.jsonObject(with: somedata),
            let jsonDict = jsonObj as? [String: Any],
            let dateStr = jsonDict["date"] as? String,
            let timeStr = jsonDict["time"] as? String else {
        print("error: invalid JSON data")
        return
    }
    // 6. Everything seems okay
    print("\(dateStr) \(timeStr)")
}
```

Handling Responses

- Completion handler called on background thread



```
        // 6. Everything seems okay
        print("\(dateStr) \(timeStr)")
        self.dateTimeLabel.text = "\(dateStr) \(timeStr)"
    }
```

UILabel.text must be used from main thread
CommDemo 〉 Thread 7 〉 0 __abort

```
=============================================================
 Thread Checker: UI API called on a background thread: -[UILabel setText:]
 88315, TID: 8735188, Thread name: (none), Queue name: com.apple.NSURLSession-delegate, QoS: 0
trace:
CommDemo                            0x0000000105a0ec18
$s8CommDemo14ViewControllerC14handleResponse4data8response5errory10Foundation4DataVSg_So13NSURLResponseCSgs5Error_pSgtF + 4488
CommDemo                            0x0000000105a0d249
$s8CommDemo14ViewControllerC21getDateTimeFromServeryyFy10Foundation4DataVSg_So13NSURLResponseCSgs5Error_pSgtcACcfu_yAH_AkMtcfu0_ +
CommDemo                            0x0000000105a0d388
$s10Foundation4DataVSgSo13NSURLResponseCSgs5Error_pSgIegggg_So6NSDataCSgAGSo7NSErrorCSgIeyByyy_TR + 296
CFNetwork                           0x00007fff2351b6ca CFNetwork + 34506
CFNetwork                           0x00007fff2352f992 _CFHTTPMessageSetResponseProxyURL + 17344
libdispatch.dylib                   0x0000000105c977ec _dispatch_call_block_and_release + 12
libdispatch.dylib                   0x0000000105c989c8 _dispatch_client_callout + 8
```

- If need to change view, dispatch to main thread

```swift
func handleResponse (data: Data?, response: URLResponse?, error: Error?) {
    ...
    DispatchQueue.main.async {
        self.dateTimeLabel.text = "\(dateStr) \(timeStr)"
    }
}
```

HTTP POST Requests

```swift
let caloriesURLString = "https://eecs.wsu.edu/~holder/tmp/calories.php"

func getCaloriesFromServer(foodname: String, servings: Int) {
    let jsonDict: [String: Any] = ["foodname": foodname, "servings": servings]
    if let jsonData = try? JSONSerialization.data(withJSONObject: jsonDict) {
        let url = URL(string: caloriesURLString)
        var request = URLRequest(url: url!)
        request.httpMethod = "POST"
        request.httpBody = jsonData
        request.setValue("application/json", forHTTPHeaderField: "Content-Type")
        let dataTask = URLSession.shared.dataTask(with: request,
                        completionHandler: handleCaloriesResponse)
        dataTask.resume()
    } else {
        print("error: invalid JSON arguments")
    }
}
```

HTTP POST Requests: Server Side

```php
<?php

// calories.php - Return calories for given food name and servings.

// Need more checks on the input here...
$json = file_get_contents("php://input");
$obj = json_decode($json);
$foodname = $obj->foodname;
$servings = $obj->servings;

$foods = array (array("pizza", 220), array("ice cream", 190), array("spaghetti", 150));

$calories = 0;
$message = "fail";

for ($foodIndex = 0; $foodIndex < count($foods); $foodIndex++) {
     if (strcasecmp($foods[$foodIndex][0], $foodname) == 0) {
          $calories = $foods[$foodIndex][1] * intval($servings);
          $message = "succeed";
          break;
     }
}

$response = array();
$response["message"] = $message;
$response["calories"] = $calories;
print json_encode($response);
?>
```

HTTP POST Requests: Handling Response

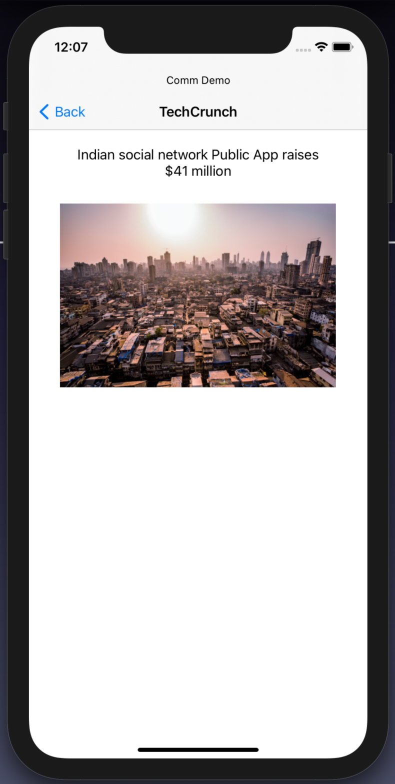
```swift
func handleCaloriesResponse (data: Data?, response: URLResponse?, error: Error?)
{
    // Checks 1-4 here...

    // 5. Check for properly-formatted JSON data
    guard let jsonObj = try? JSONSerialization.jsonObject(with: somedata),
              let jsonDict = jsonObj as? [String: Any],
              let messageStr = jsonDict["message"] as? String,
              let calories = jsonDict["calories"] as? Int else {
        print("error: invalid JSON data")
        return
    }
    // 6. Returned data seems okay
    if (messageStr == "succeed") {
        print("calories = \(calories)")
    } else {
        print("food not found")
    }
}
```

Application Programming Interfaces (APIs)

- News
  - newsapi.org
- Weather
  - openweathermap.org/api
- Food
  - spoonacular.com/food-api
- And many more (24,000+)
  - www.programmableweb.com

API Requests

```
// My newsAPIKey for newsapi.org is defined in another Swift file
let newsURLString = "https://newsapi.org/v2/top-headlines?sources=techcrunch&apiKey=\(newsAPIKey)"

func getNews() {
      // May not know exactly what's in the URL, so replace special characters with % encoding
      if let urlStr = newsURLString.addingPercentEncoding(withAllowedCharacters: .urlQueryAllowed) {
            if let url = URL(string: urlStr) {
                  let dataTask = URLSession.shared.dataTask(with: url,
                        completionHandler: handleNewsResponse)
                  dataTask.resume()
            }
      }
}
```



Indian social network Public App raises $41 million

Handle API Responses

```
func handleNewsResponse (data: Data?, response: URLResponse?, error: Error?) {
    // Checks 1-4 here...
    // 5. Check for properly-formatted JSON data
    guard let jsonObj = try? JSONSerialization.jsonObject(with: somedata),
            let jsonDict1 = jsonObj as? [String: Any],
            let articleArray = jsonDict1["articles"] as? [Any],
            articleArray.count > 0,
            let jsonDict2 = articleArray[0] as? [String: Any],
            let titleStr = jsonDict2["title"] as? String,
            let urlToImage = jsonDict2["urlToImage"] as? String else {
        print("error: invalid JSON data")
        return
    }
    print(jsonDict1)
    // 6. Everything seems okay
    self.loadNewsImage(urlToImage)
    DispatchQueue.main.async {
        self.newsTitleLabel.text = titleStr
    }
}
```

Handle API Responses

```swift
func loadNewsImage(_ urlString: String) {
     // URL comes from API response; definitely needs some safety checks
     if let urlStr = urlString.addingPercentEncoding(
                    withAllowedCharacters: .urlQueryAllowed) {
          if let url = URL(string: urlStr) {
               let dataTask = URLSession.shared.dataTask(with: url,
                    completionHandler: {(data, response, error) -> Void in
                    if let imageData = data {
                         let image = UIImage(data: imageData)
                         DispatchQueue.main.async {
                              self.newsImageView.image = image
                         }
                    }
               })
               dataTask.resume()
          }
     }
}
```

Other Communications Services

- CloudKit
  - Share data across devices and apps
- GameKit
  - Peer-to-peer for multi-player and voice
- Network Services
  - WiFi, Bluetooth
- Sockets

Resources

- Safari Services
  - developer.apple.com/documentation/safariservices
- Web Kit
  - developer.apple.com/documentation/webkit
- URLSession
  - developer.apple.com/documentation/foundation/urlsession