Networking

Dr.-Ing. Thomas Springer
M.Sc. Martin Weißbach
Example Project

https://www.rn.inf.tu-dresden.de/martin/iOS-Programming-17/
Network-Example.zip
Networking
• new APIs since iOS7
  • URLSession API
  • allows consecutive requests in a single session
  • each session contains its own configuration and cache
  • support for background execution built in

• APIs for iOS6 and earlier are still available and not deprecated
NSURLConnection Stack

1) (create a delegate adopting the URLSessionDelegate)

2) set up a configuration for the session
   (NSURLSessionConfiguration)

3) create the URLSession

4) create a URL object

5) create a URLSessionTask to actually create the request
to be made via this session
Main Classes & Protocols of the URLSession stack
• base class for tasks in an URLSession
  • created by the session in form of a subclass
    • URLSessionDataTask, URLSessionUploadTask,
      URLSessionDownloadTask, URLSessionStreamTask

• status and progress properties

• cancel, suspend, resume

• data and upload tasks provided (subclasses)
• **Data Task**
  • “intended for short, often interactive requests” [1]
  • return of data piecemeal or all at once
  • no data stored to a file → not supported in background sessions

• **Download Task**
  • data retrieval in form of a file
  • support background download while the app is not running

• **Upload Task**
  • send data (usually as a file) to a server
  • support background upload while the app is not running

• **Stream Task** *(since iOS9)*
  • TCP/IP connection (bidirectional, asynchronous communication)
  • maybe result of upgrading an HTTP connection
• base protocol for all delegates of URLSession
  • subprotocols following the principle of URLSessionTask subclasses

• session keeps strong reference

• connection level authentication

• URLSession(_:didBecomeInvalidWithError:)
  • error will be nil when session was successful
  • gets always called before the delegate is released
• protocol to handle task-level events in a URLSession

• Delegate adopting the URLSessionDelegate is recommended to implement this protocol as well

• URLSession(_:task:didCompleteWithError:)
  • error will be nil for successful requests
  • connection and transmission errors only
    • e.g. 404 will not be detected by this method
• custom sessions with private configuration

• invalidation required for session

• creates data, upload & download tasks

• async convenience APIs
  • cancelable
  • can share delegate for authentication
• **Default Session**
  - persistent disk-based cache
  - stores credentials in the user’s keychain

• **Ephemeral Session**
  - caches, credential stores etc. kept in RAM and are tied to the session
  - all data is purged when application invalidates the cache

• **Background Session**
  - similar to default session
  - separate process handles all data transfer
  - additional limitations
• Additional Limitations are:

  • session **must** provide a delegate for delivery callbacks
  • supports only HTTP(S)
  • supports only upload and download tasks
  • redirects are always followed
Background Session [2]

NSURLSession Identifier: “DownloadSession”

Create Download

Progress

Progress

NSURLSession Background Daemon
Background Session [2]

NSURLConnection Identifier: “DownloadSession”

Create Download

Progress

Progress

NSURLConnection Background Daemon
NSURLSession
Identifier: “DownloadSession”

NSURLSession
Background Daemon
Background Session [2]

NSURLSession Identifier: “DownloadSession”

Reconnect

Relaunched

Completed

NSURLSession Background Daemon
App Transport Security (ATS)

- by default, only secured network connections are allowed
- ATS converts HTTP requests automatically to HTTPS requests

Indicating Networking Tasks
• indicating network utilization is a voluntary task in iOS; the system does not inform the user by default
• UI widgets available to show task processing
UIApplication: var networkActivityIndicatorVisible: Bool
Custom Activity Indicator

- synchronous request that loads data
- asynchronous at application level to prevent UI from being blocked
- yet user sees a white or empty screen
  - graphical mean to depict load of content in progress
- UIActivityIndicatorView
• application is not waiting for data to be loaded
  • i.e. sending a message, uploading data
  • it might just be of interest that the message was transmitted successfully to the server

• usually indicated by a progress indicator – UIProgressView
References
