NSURLSession: New Features and Best Practices

Session 711

Jeff Jenkins Software Engineer, Internet Technologies
Agenda
Agenda

Evolving NSURLSession API
Agenda

Evolving NSURLSession API

Security
Agenda

Evolving NSURLSession API
Security
Best practices and tips
Networking API
Networking API

Foundation

NSURLConnection
NSNetServices
NSStream
NSURLSession Review
NSURLSession Example

Benefits
NSURLSession Example

Benefits

HTTP/1.1, SPDY, HTTP/2
NSURLSession Example

Benefits

HTTP/1.1, SPDY, HTTP/2

App Transport Security (ATS)
NSURLSession Example

Benefits

HTTP/1.1, SPDY, HTTP/2

App Transport Security (ATS)

HTTP Strict Transport Security (HSTS)
NSURLSession Example

Benefits

HTTP/1.1, SPDY, HTTP/2

App Transport Security (ATS)

HTTP Strict Transport Security (HSTS)

Cache, cookies, proxy, authentication
NSURLSession Example

Benefits

HTTP/1.1, SPDY, HTTP/2

App Transport Security (ATS)

HTTP Strict Transport Security (HSTS)

Cache, cookies, proxy, authentication

Configuration
NSURLSession Example

Benefits

NSURLSessionConfiguration
NSURLSession Example

Benefits

NSURLSessionConfiguration
Transport Layer Security (TLS) version
NSURLSession Example

Benefits

NSURLSessionConfiguration

Transport Layer Security (TLS) version

Prohibit cellular usage
NSURLSession Example

Benefits

NSURLSessionConfiguration

Transport Layer Security (TLS) version

Prohibit cellular usage

Network service type
NSURLSession Example

Benefits

NSURLSessionConfiguration

Transport Layer Security (TLS) version

Prohibit cellular usage

Network service type

Cookie policy
NSURLConnection Example

Benefits

- URLSessionConfiguration
- Transport Layer Security (TLS) version
- Prohibit cellular usage
- Network service type
- Cookie policy
- Cache policy
NSURLConnection Example

Benefits

- URLSessionConfiguration
- Transport Layer Security (TLS) version
- Prohibit cellular usage
- Network service type
- Cookie policy
- Cache policy
- Storage objects
NSURLSession Example

Benefits

NSURLSessionConfiguration

Transport Layer Security (TLS) version

Prohibit cellular usage

Network service type

Cookie policy

Cache policy

Storage objects

Request and resource timeouts
NSURLSession Example
Review
let config = NSURLSessionConfiguration.defaultSessionConfiguration()
NSURLSession Example

let config = NSURLSessionConfiguration.defaultSessionConfiguration()

let session = URLSession(configuration: config)
NSURLSession Example
Review
let url = NSURL(string: "https://www.example.com")!
let url = NSURL(string: "https://www.example.com")!

let task = session.dataTask(with: url) { (data: NSData?, response: NSURLResponse?, error: NSError?) in ...
}
let url = NSURL(string: "https://www.example.com")!

let task = session.dataTask(with: url) { (data: NSData?, response: NSURLResponse?, error: NSError?) in
    ...
}
let task = session.dataTask(with: url) { (data: NSData?, response: NSURLResponse?, error: NSError?) in

...}

task.resume()
NSURLSession Example
Review
NSURLSession Example

Summary: Three-step process
NSURLSession Example

Review

Summary: Three-step process

• Configuration
NSURLConnection Example

Review

Summary: Three-step process
- Configuration
- Session
NSURLSession Example

Review

Summary: Three-step process

• Configuration
• Session
• Tasks
NSURLConnection Example

Review

Summary: Three-step process
• Configuration
• Session
• Tasks

Task 1  →  Session 1
Task 2  →  Session 2
Task 3  →  Session 3
Review

Summary: Three-step process
• Configuration
• Session
• Tasks

Task 1 → Session 1
Task 2 → Session 2
Task 3 → Session 3

Task 1
Task 2 → Session 1
Task 3
Summary: Three-step process

- Configuration
- Session
- Tasks
NSURLSession API
HTTP/2
NSURLSession API
HTTP/2
NSURLConnection API
HTTP/2

- Apple
- Facebook
- Instagram
- Wikipedia
- Twitter
- YouTube
- Yahoo
NSURLSession API

HTTP/2

- Apple
- Facebook
- Instagram
- Wikipedia
- Twitter
- YouTube
- Yahoo
NSURLConnection API

HTTP/2

• Multiplexing and concurrency

• Apple
• Facebook
• Instagram
• Wikipedia
• Twitter
• YouTube
• Yahoo
NSURLConnection API

HTTP/2

- Multiplexing and concurrency
- Header compression

- Apple
- Facebook
- Instagram
- Wikipedia
- Twitter
- YouTube
- Yahoo
HTTP/2 Protocol
• Multiplexing and concurrency
• Header compression
• Stream priorities

NSURLSession API
• Apple
• Facebook
• Instagram
• Wikipedia
• Twitter
• YouTube
• Yahoo
HTTP/2 Protocol

• Multiplexing and concurrency
• Header compression
• Stream priorities
• Server Push

NAMESPACE

NEW

• Apple
• Facebook
• Instagram
• Wikipedia
• Twitter
• YouTube
• Yahoo
NSURLSession API

HTTP/2 Server Push
NSURLConnection API

HTTP/2 Server Push

Prevent network round trips
NSURLConnection API

HTTP/2 Server Push

Prevent network round trips
Server support required
HTTP/2 Server Push

NSURLConnection API

Prevent network round trips
Server support required
Now available in NSURLConnection
HTTP/1.1
Loading

Time

App  NSURLConnection  Server
HTTP/1.1

Latency

Time

App

NSURLSession

Server
HTTP/1.1

Loading

Latency

Time

App

NSURLSession

Server
HTTP/1.1

Loading

Latency

Time

App

NSURLSession

Server

GET index.html

Response
HTTP/1.1

Loading

Latency

Time

App

NSURLSession

Server

GET index.html

Response

index.html
HTTP/1.1

Loading

Latency

Time

index.html

App

NSURLConnection

Server

GET index.html

Response

GET style.css
HTTP/1.1

Loading

Latency

Time

index.html

style.css

App

NSURLSession

Server

GET index.html

Response

GET style.css

Response
HTTP/1.1

Loading

Latency

Time

App

NSURLSession

Server

GET index.html

Response

GET style.css

Response

GET background.jpg
HTTP/1.1 Loading

Latency

Time

App

NSURLSession

Server

GET index.html

Response

GET style.css

Response

GET background.jpg

Response
HTTP/1.1

Loading

Latency

Time

index.html

style.css

background.jpg

App

NSURLSession

Server

GET index.html

Response

GET style.css

Response

GET background.jpg

Response
HTTP/1.1

Loading

Latency

Time

index.html

style.css

background.jpg

finish

App

NSURLSession

Server

GET index.html

Response

GET style.css

Response

GET background.jpg

Response
HTTP/1.1

Latency

Time

HTTP/1.1

App

GET index.html
Response

GET style.css
Response

GET background.jpg
Response

Server

index.html

style.css

background.jpg

finish
HTTP/2 Server Push

Latency

HTTP/1.1

Loading

NEW
HTTP/2 Server Push

Loading

- Latency
- Time

HTTP/1.1

App

NSURLSession

GET index.html

Server
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Time

NSURLSession

NEW

HTTP/2 Server Push

App

NSURLSession

Server

GET index.html

Response
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

NEW

NSURLSession

PUSH 2, 3, n

GET index.html

Response

App

Server
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Time

HTTP/2 Server Push

App

NSURLSession

Server

GET index.html

Response

GET style.css

PUSH 2, 3, n

Latency

Time
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Get index.html

Response

Get style.css

Response

PUSH 2, 3, n
HTTP/2 Server Push

Latency

HTTP/1.1

Time

App

NSURLSession

Server

GET index.html
Response

GET style.css
Response

PUSH 2, 3, n

Loading
HTTP/2 Server Push

Loading

- GET index.html
- GET style.css
- GET background.jpg

HTTP/1.1 Latency

PUSH 2, 3, n

App → NSURLConnection → Server
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Time

index.html

style.css

GET index.html
Response

GET style.css
Response

GET background.jpg

NSURLSession

PUSH 2, 3, n

App

Server
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Time

index.html

style.css

background.jpg

GET style.css

Response

GET index.html

Response

PUSH 2, 3, n

GET background.jpg

NSURLSession

NEW

App

NSURLConnection

Server
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Time

App

NSURLSession

Server

GET index.html
Response

GET style.css
Response

GET background.jpg

PUSH 2, 3, n

index.html
style.css
background.jpg
finish

HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Time

App

NSURLSession

Server

GET index.html
Response

GET style.css
Response

GET background.jpg

PUSH 2, 3, n

index.html
style.css
background.jpg
finish
HTTP/2 Server Push

Loading

HTTP/2 Server Push

Latency

index.html

style.css

background.jpg

finish

HTTP/1.1

Latency

Time

NSURLSession

NEW

App

NSURLConnection

Server

GET index.html

Response

GET style.css

Response

GET background.jpg

PUSH 2, 3, n
HTTP/2 Server Push

Loading

HTTP/1.1

Latency

Time

HTTP/2 Server Push

index.html

style.css

background.jpg

finish

NSURLSession

NEW

PUSH 2, 3, n

GET index.html

Response

GET style.css

Response

GET background.jpg

Latency

Time

HTTP/2

Server Push

Loading
Demo
HTTP/2 Server Push

Andreas Garkuscha Software Engineer, Internet Technologies
Demo

HTTP/2 Server Push

Andreas Garkuscha Software Engineer, Internet Technologies
HTTP/2 Server Push

Summary
HTTP/2 Server Push

Summary

Server Push supported in NSURLSession
HTTP/2 Server Push

Summary

Server Push supported in NSURLConnection
No opt in, just works
HTTP/2 Server Push

Summary

Server Push supported in NSURLSession

No opt in, just works

Even better performance
NSURLSession API
Network statistics
NSURLSession API

Network statistics
Did you know?
Did you know?

98% of all statistics are made up!
Did you know?

98% of all statistics are made up!

Network statistics: Performance, bugs, understanding
Did you know?

98% of all statistics are made up!

Network statistics: Performance, bugs, understanding

Currently supported on iOS, macOS, and tvOS
NSURLSession API
Network statistics delegate
func urlSession(_ session: URLSession, task: URLSessionTask, 
didFinishCollecting metrics: URLSessionTaskMetrics)
NSURLSession API
NSURLSessionTaskMetrics
NSURLSession API
NSURLSessionTaskMetrics
NSURLSession API

NSURLSessionTaskMetrics

**taskInterval**: `NSDateInterval`
NSURLSession API
NSURLSessionTaskMetrics

taskInterval: NSDateInterval

redirectCount: Int
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>taskInterval</td>
<td>NSDateInterval</td>
</tr>
<tr>
<td>redirectCount</td>
<td>Int</td>
</tr>
<tr>
<td>transactionMetrics</td>
<td>[NSURLSessionTaskTransactionMetrics]</td>
</tr>
</tbody>
</table>
NSURLSession API
NSURLSessionTaskMetrics

- `taskInterval`: `NSDateInterval`
- `redirectCount`: `Int`
- `transactionMetrics`: `[NSURLSessionTaskTransactionMetrics]`
Four Categories of Metrics
NSURLSession API
NSURLSessionTaskTransactionMetrics
NSURLSession API

NSURLSessionTaskTransactionMetrics

1 - Request and Response
NSURLSession API

NSURLSessionTaskTransactionMetrics

1 - Request and Response

request: NSURLRequest
response: NSURLResponse?
NSURLSession API

NSURLSessionTaskTransactionMetrics

1 - Request and Response

request: NSURLRequest
response: NSURLResponse?
NSURLSession API
NSURLSessionTaskTransactionMetrics
NSURLSession API

NSURLSessionTaskTransactionMetrics

2 - Protocol and Connection
networkProtocolName: String?
2 - Protocol and Connection

**networkProtocolName**: String?

- http/1.1
- h2
- spdy/3, spdy/3.1
NSURLSession API

NSURLSessionTaskTransactionMetrics

2 - Protocol and Connection

networkProtocolName: String?

isProxyConnection: Bool

http/1.1
h2
spdy/3, spdy/3.1
NSURLSession API

NSURLSessionDataTaskTransactionMetrics

2 - Protocol and Connection

- `networkProtocolName: String?`
- `isProxyConnection: Bool`
- `isReusedConnection: Bool`

Protocols:

- `http/1.1`
- `h2`
- `spdy/3, spdy/3.1`
NSURLSession API
NSURLSessionTaskTransactionMetrics
NSURLSession API
NSURLSessionTaskTransactionMetrics

3 - Load Info
NSURLSession API

NSURLSessionTaskTransactionMetrics

3 - Load Info

resourceFetchType: NSURLSessionTaskMetricsResourceFetchType
NSURLSession API

NSURLSessionTaskTransactionMetrics

3 - Load Info

resourceFetchType: NSURLSessionTaskMetricsResourceFetchType

.networkLoad
.localCache
.serverPush
NSURLSession API
NSURLSessionTaskTransactionMetrics
NSURLConnection API

NSURLSessionTaskTransactionMetrics

4 - Connection Establishment and Transmission
fetchStartDate: NSDate?
domainLookupStartDate: NSDate?
domainLookupEndDate: NSDate?
connectStartDate: NSDate?
secureConnectionStartDate: NSDate?
secureConnectionEndDate: NSDate?
connectEndDate: NSDate?
fetchAllDate: NSDate?
domainLookupStartDate: NSDate?
domainLookupEndDate: NSDate?
connectStartDate: NSDate?
secureConnectionStartDate: NSDate?
secureConnectionEndDate: NSDate?
connectEndDate: NSDate?
requestStartDate: NSDate?
requestEndDate: NSDate?
responseStartDate: NSDate?
responseEndDate: NSDate?
NSURLSession API

NSURLSessionTaskTransactionMetrics
NSURLSession API
NSURLSessionTaskTransactionMetrics

fetchStart
Task
NSURLSession API
NSURLSessionTaskTransactionMetrics

fetchStart

Task
 NSURLSession API

NSURLSessionTaskTransactionMetrics

fetchStart

Task
NSURLSession API

NSURLSessionTaskTransactionMetrics

fetchStart  domainLookupStart

Task  DNS

domainLookupEnd
NSURLConnection API

NSURLSessionTaskTransactionMetrics

- `fetchStart`
- `domainLookupStart`
- `DNS`
- `domainLookupEnd`
Fetch Start

Domain Lookup Start

Domain Lookup End

Connection Establishment

Connect Start

Connect End
NSURLSession API
NSURLSessionTaskTransactionMetrics

Connection Establishment

connectStart

connectEnd

fetchStart  domainLookupStart

Task  DNS

domainLookupEnd
NSDataSession API
NSDataSessionTaskTransactionMetrics

Task
connectStart
DNS
domainLookupStart
domainLookupEnd
secureConnectionStart
secureConnectionEnd
connectEnd

fetchStart
TCP
TLS
 NSURLSession API
NSURLSessionTaskTransactionMetrics

secureConnectionStart

TLS

secureConnectionEnd

fetchStart domainLookupStart

Task DNS

domainLookupEnd

connectStart

Connection Establishment

TCP

connectEnd

NEW
NSURLSession API
NSURLSessionTaskTransactionMetrics

Task
fetchStart
domainLookupStart
domainLookupEnd

DNS

secureConnectionStart
TCP
secureConnectionEnd
TLS
Connection Establishment
connectStart
connectEnd
NSURLConnection API

NSURLSessionTaskTransactionMetrics

Fetch Start: deltaTime = connectStart - fetchStart

Domain Lookup Start: deltaTime = domainLookupStart - connectStart

DNS: deltaTime = domainLookupEnd - domainLookupStart

Connection Establishment: connectStart to secureConnectionEnd

TCP

TLS

HTTP

secureConnectionStart: connectEnd - secureConnectionEnd
NSURLSession API
NSURLSessionTaskTransactionMetrics

GET

requestStart

requestEnd

secureConnectionStart

secureConnectionEnd

Task

DNS

domainLookupStart

domainLookupEnd

connectStart

TCP

Connection Establishment

TLS

HTTP

fetchStart
NSURLSession API

NSURLSessionTaskTransactionMetrics

Task
- fetchStart
- domainLookupStart
- domainLookupEnd

DNS
- connectStart
- domainLookupEnd

Connection Establishment
- secureConnectionStart
- TCP
- TLS

GET
- requestStart
- requestEnd

HTTP
- responseStart
- 200 OK
- responseEnd

secureConnectionEnd
- connectEnd
NSURLSession API
NSURLSessionTaskTransactionMetrics

- `responseStart` to `responseEnd` with a status of `200 OK`
- Connection Establishment:
  - `connectStart`
  - `domainLookupStart` to `domainLookupEnd`
  - `secureConnectionStart`
  - `secureConnectionEnd`
- GET to HTTP:
  - `requestStart` to `requestEnd`
NSURLSession API
NSURLSessionTaskTransactionMetrics

Task

fetchStart
domainLookupStart
domainLookupEnd

DNS

customNameStart
domainLookupEnd

TCP

secureConnectionStart

Connection Establishment

secureConnectionEnd

GET

requestStart
requestEnd

HTTP

responseStart

200 OK

responseEnd
 NSURLSession API
NSURLSessionTaskTransactionMetrics

- Task
  - fetchStart
  - domainLookupStart
  - connectStart
  - domainLookupEnd

- DNS

- Connection Establishment
  - secureConnectionStart
  - secureConnectionEnd
  - connectStart
  - requestStart
  - connectEnd
  - requestEnd

- TCP
  - domainLookupStart

- TLS

- GET

- HTTP
  - responseStart
  - 200 OK

- responseEnd
**NSURLSession API**

**NSURLSessionTaskTransactionMetrics**

- **Cache**
  - `fetchStart`
  - `domainLookupStart`
  - `domainLookupEnd`

- **Task**
  - `connectStart`
  - `secureConnectionStart`
  - `TCP`
  - `TLS`
  - `Connection Establishment`
  - `connectEnd`
  - `secureConnectionEnd`

- **DNS**
  - `domainLookupStart`
  - `domainLookupEnd`

- **GET**
  - `requestStart`
  - `requestEnd`

- **HTTP**
  - `responseStart`
  - `responseEnd`
  - `200 OK`
NSURLSession API
NSURLSessionTaskTransactionMetrics
class MySessionDelegate: NSObject, NSURLSessionTaskDelegate {
    @objc(URLSession:task:didFinishCollectingMetrics:)
    func urlSession(_ session: URLSession, task: URLSessionTask, didFinishCollecting metrics: NSURLSessionTaskMetrics) {
        // metrics.redirectCount
        // metrics.taskInterval
        // metrics.transactionMetrics[0].connectStartDate ...
    }
}
class MySessionDelegate: NSObject, NSURLSessionTaskDelegate {
    @objc(URLSession:task:didFinishCollectingMetrics:)
    func urlSession(_ session: URLSession, task: URLSessionTask, didFinishCollecting metrics: NSURLSessionTaskMetrics) {
        //metrics.redirectCount
        //metrics.taskInterval
        //metrics.transactionMetrics[0].connectStartDate …
    }
}

let myDelegate = MySessionDelegate()
class MySessionDelegate: NSObject, URLSessionTaskDelegate {
    @objc(URLSession:task:didFinishCollectingMetrics:)
    func urlSession(_ session: URLSession, task: URLSessionTask, didFinishCollecting metrics: URLSessionTaskMetrics) {
        //metrics.redirectCount
        //metrics.taskInterval
        //metrics.transactionMetrics[0].connectStartDate …
    }
}

let myDelegate = MySessionDelegate()
let config = URLSessionConfiguration.defaultSessionConfiguration()
class MySessionDelegate: NSObject, URLSessionTaskDelegate {
    @objc(URLSession:task:didFinishCollectingMetrics:)
    func urlSession(_ session: URLSession, task: URLSessionTask, didFinishCollecting metrics: NSURLSessionTaskMetrics) {
        //metrics.redirectCount
        //metrics.taskInterval
        //metrics.transactionMetrics[0].connectStartDate …
    }
}

let myDelegate = MySessionDelegate()
let config = URLSessionConfiguration.defaultSessionConfiguration()
let myDelegateQueue = NSOperationQueue()
class MySessionDelegate: NSObject, NSURLSessionTaskDelegate {
    @objc(URLSession:task:didFinishCollectingMetrics:)
    func urlSession(_ session: NSURLSession, task: NSURLSessionTask, didFinishCollecting metrics: NSURLSessionTaskMetrics) {
        //metrics.redirectCount
        //metrics.taskInterval
        //metrics.transactionMetrics[0].connectStartDate …
    }
}

let myDelegate = MySessionDelegate()
let config = NSURLSessionConfiguration.defaultSessionConfiguration()
let myDelegateQueue = NSOperationQueue()

let session = URLSession(configuration: config, delegate: myDelegate, delegateQueue: myDelegateQueue)
class MySessionDelegate: NSObject, URLSessionTaskDelegate {
    @objc(URLSession:task:didFinishCollectingMetrics:)
    func urlSession(_ session: URLSession, task: URLSessionTask, didFinishCollecting metrics: NSURLSessionTaskMetrics) {
        //metrics.redirectCount
        //metrics.taskInterval
        //metrics.transactionMetrics[0].connectStartDate …
    }
}

let myDelegate = MySessionDelegate()
let config = URLSessionConfiguration.defaultSessionConfiguration()
let myDelegateQueue = NSOperationQueue()
let session = URLSession(configuration: config, delegate: myDelegate, delegateQueue: myDelegateQueue)

let url = NSURL(string: "https://www.example.com")!
let task = session.dataTask(with: url) { (data: NSData?, response: NSURLResponse?, error: NSError?) in
    ...
}
class MySessionDelegate: NSObject, URLSessionTaskDelegate {
    @objc(URLSession:task:didFinishCollectingMetrics:)
    func urlSession(_ session: URLSession, task: URLSessionTask, didFinishCollecting metrics: URLSessionTaskMetrics) {
        //metrics.redirectCount
        //metrics.taskInterval
        //metrics.transactionMetrics[0].connectStartDate …
    }
}

let myDelegate = MySessionDelegate()
let config = URLSessionConfiguration.defaultSessionConfiguration()
let myDelegateQueue = NSOperationQueue()
let session = URLSession(configuration: config, delegate: myDelegate, delegateQueue: myDelegateQueue)

let url = NSURL(string: "https://www.example.com/")!
let task = session.dataTask(with: url) { (data: NSData?, response: NSURLResponse?, error: NSError?) in
    …
}
NSURLSession API

Network statistics
NSURLSession API

Network statistics

Summary
NSURLSession API

Network statistics

Summary

Deep view into network loading
NSURLConnection API
Network statistics

Summary

- Deep view into network loading
- Faster debugging, easier profiling, better performance
NSURLConnection API

Network statistics

Summary

- Deep view into network loading
- Faster debugging, easier profiling, better performance
- Available in NSURLConnection
Security
Never an afterthought
Security

Transport Layer Security
Security
Transport Layer Security

Protects data
Security

Transport Layer Security

Protects data

Cipher—algorithm for encrypting/decrypting data
Security

Transport Layer Security

Protects data

Cipher—algorithm for encrypting/decrypting data

The RC4 cipher is no longer supported
Security

Transport Layer Security

Protects data

Cipher—algorithm for encrypting/decrypting data

The RC4 cipher is no longer supported

• Tuesday—Security session and labs

How iOS Security Really Works

Nob Hill
Tuesday 4:00PM

What's New in Security

Nob Hill
Tuesday 5:00PM
Security
nscurl
$ nscurl https://insecure.example.com/
Security

ns curl

$ ns curl https://insecure.example.com/

ns curl[1234:123456] NSURLConnection HTTP load failed
(kCFStreamErrorDomainSSL, -9824)
Security

$ nscurl --enable-rc4 https://insecure.example.com/
Security

nscurl

$ ns curl --enable-rc4 https://insecure.example.com/

Enabling RC4 cipher suites
<html><body><h1>It works!</h1></body></html>
App Transport Security

NSAllowsArbitraryLoadsInWebContent
NSRequiresCertificateTransparency
App Transport Security

How iOS Security Really Works

What's New in Security
Summary

NSURLSession
Summary

NSURLSession
  • HTTP/2 Server Push
Summary

NSURLSession

- HTTP/2 Server Push Performance!
Summary

NSURLSession

• HTTP/2 Server Push Performance!
• Network statistics
Summary

NSURLSession

- HTTP/2 Server Push: Performance!
- Network statistics: Faster and better analysis of applications
Summary

NSURLSession

• HTTP/2 Server Push: Performance!
• Network statistics: Faster and better analysis of applications
• Security
Summary

NSURLSession

• HTTP/2 Server Push
  Performance!

• Network statistics
  Faster and better analysis of applications

• Security
  Protect user data
Summary

NSURLSession

- HTTP/2 Server Push: Performance!
- Network statistics: Faster and better analysis of applications
- Security: Protect user data

NSURLSession in your applications!
More Information

https://developer.apple.com/wwdc16/711
<table>
<thead>
<tr>
<th>Related Sessions</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>What’s New in Foundation for Swift</td>
<td>Mission</td>
<td>Tuesday 4:00PM</td>
</tr>
<tr>
<td>What’s New in Security</td>
<td>Nob Hill</td>
<td>Tuesday 5:00PM</td>
</tr>
<tr>
<td>Engineering Privacy for Your Users</td>
<td>Pacific Heights</td>
<td>Wednesday 4:00PM</td>
</tr>
<tr>
<td>Networking for the Modern Internet</td>
<td>Pacific Heights</td>
<td>Thursday 3:00PM</td>
</tr>
<tr>
<td>Related Labs</td>
<td>Frameworks Lab B</td>
<td>Thursday 4:00PM</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Networking Lab 1</td>
<td>Frameworks Lab D</td>
<td>Friday 2:00PM</td>
</tr>
<tr>
<td>Networking Lab 2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>