

NSURLSession: New Features and Best Practices

[원문 링크](#)

소개

WWDC 2016에 발표된 NSURLSession API에 관해 몇 가지 새로운 기능과 향상된 기능을 알려줍니다. HTTP/2, Server Push 기능에 관한 지원, 네트워크 통계를 이용한 애플리케이션의 신속하고 효율적인 분석, ATS(App Transport Security)의 보안 기능과 RC4 암호의 사용을 중단하여 사용자 데이터를 더욱더 안전하게 보호하는 방법에 관해 설명합니다.

NSURLSession

HTTP/1.1 프로토콜, SPDY(구글에서 개발한 비표준 네트워크 프로토콜), HTTP/2(SPDY에 기반을 뒀서 개발)들을 지원하며 해당 프로토콜들을 고려하면서 개발해야 한다고 설명합니다. 그리고 HTTP Strict Transport Security (HSTS)의 기능을 이용해 사용자의 데이터와 개인정보를 보호할 수 있고 NSURLSession의 구성정보 (NSURLSessionConfiguration)를 이용해 네트워킹을 훌륭하게 조정하고 제어할 수 있다고 설명합니다.

NSURLSessionConfiguration

- Transport Layer Security (TLS) version
 - 애플리케이션에서 지원하고자 하는 TLS의 최소 및 최대 버전을 제어할 수 있습니다.
- Prohibit cellular usage
 - 애플리케이션으로 셀룰러 사용을 제어할 수 있습니다.
- Network service type
 - 네트워크 유형을 지정할 수 있습니다.
- Cookie policy
 - 세션 기간 또는 영구적으로 쿠키를 지정할 수 있습니다.
- Cache policy
 - 임시로 생성한 캐시와 영구적으로 캐시를 지정할 수 있습니다.
- Storage objects

- 다른 유형의 네트워킹 간에 캐시를 공유하고자 할 때 사용할 수 있는 공간입니다.

- Request and resource timeout

- 애플리케이션이 네트워크의 오류상태를 처리할 수 있도록 리소스를 설정하고 제한시간을 요청할 수 있습니다.

```
let url = NSURL(string: "https://www.example.com/")!

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

task.resume()
```

모범사례 : 성능상 많은 영향을 미치기 때문에 한 태스크와 한 세션의 모델(one-session to one-task)을 피하도록 강조 합니다. 많은 세션을 처리할 경우 메모리관리 및 OS리소스에 문제가 생길 수 있다고 설명합니다.

NSURLSession API

HTTP/2 프로토콜에 관해 설명하고 NSURLSession 내에서 이 프로토콜이 어떻게 수행하는지 설명합니다. 그리고 HTTP/2 Server Push를 이용해 HTTP/1.1과 HTTP/2의 성능 차이를 비교해 주고 네트워크 통계를 이용해 애플리케이션 내의 네트워크 성능을 분석하고 관련 버그를 찾아 수정할 수 있습니다. 또한 애플리케이션 내에서 네트워킹이 수행하는 작업을 더 잘 이해 할 수 있다고 설명합니다. iOS, macOS, tvOS 플랫폼에서 지원된다고 합니다.

HTTP/2 Protocol

- Multiplexing and concurrency

- HTTP/2의 가장 큰 특징 중 하나는 다중화 및 동시성 지원이라고 설명합니다.

- Header compression

- HTTP/1.1과는 달리 HTTP/2에서는 헤더의 중복성을 줄임으로 인해 네트워크 왕복을 줄일 수 있다고 설명합니다.

- Stream priorities

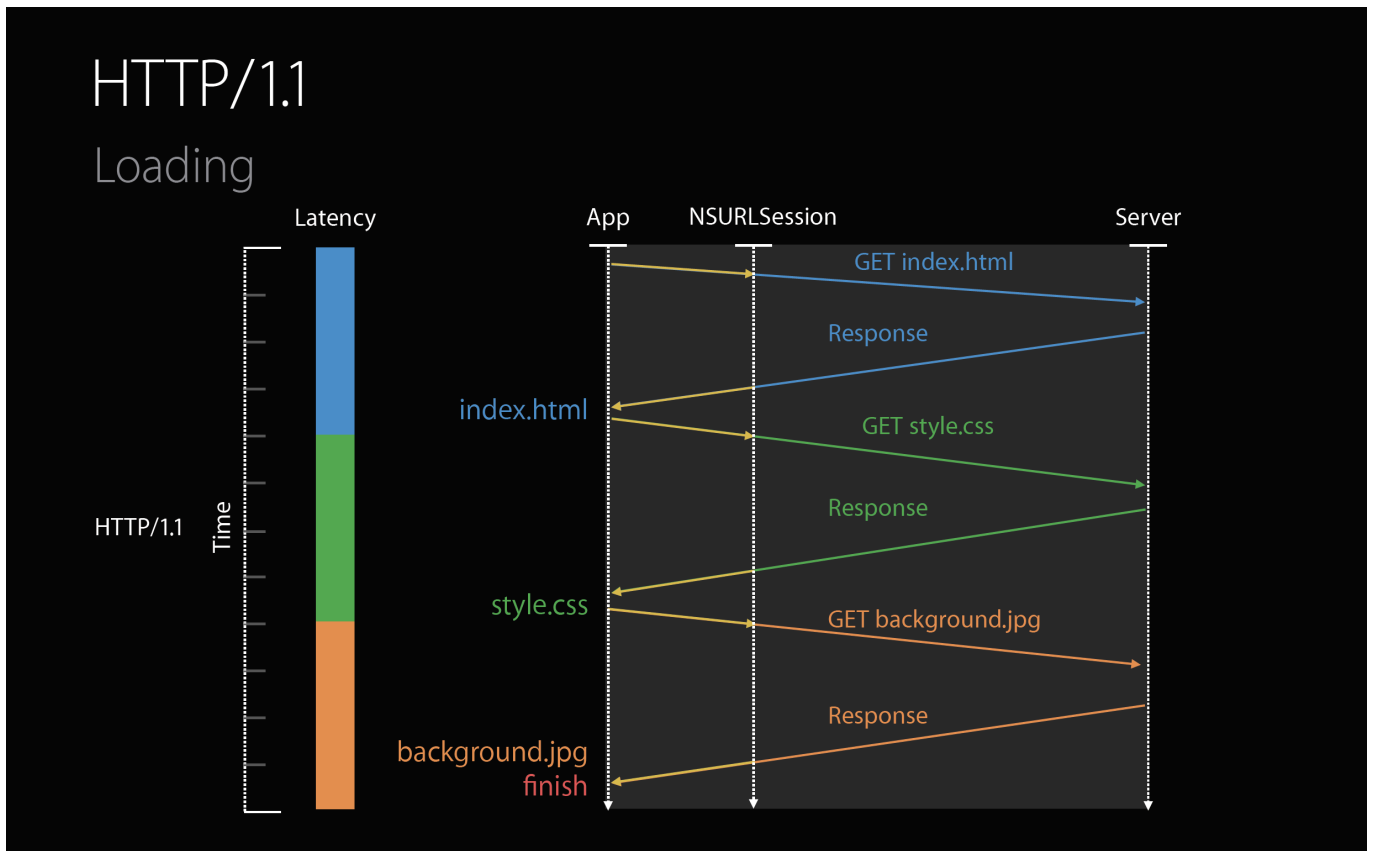
- 흐름(스트림)의 우선순위로 인해 서버로부터 리소스가 반환되는 우선순위를 나타낼 수 있다고 설명합니다.

- Server Push

- 클라이언트가 요청하면 서버가 응답하게 되는데 HTTP/1.1의 경우 동시에 클라이언트 추가 응답을 보낼 수가 없었습니다. HTTP/2에서는 동시에 추가 응답을 보낼 수 있게 되는데 이 기능이 서버 푸시 기능이라고 설명합니다.

HTTP/2 Server Push

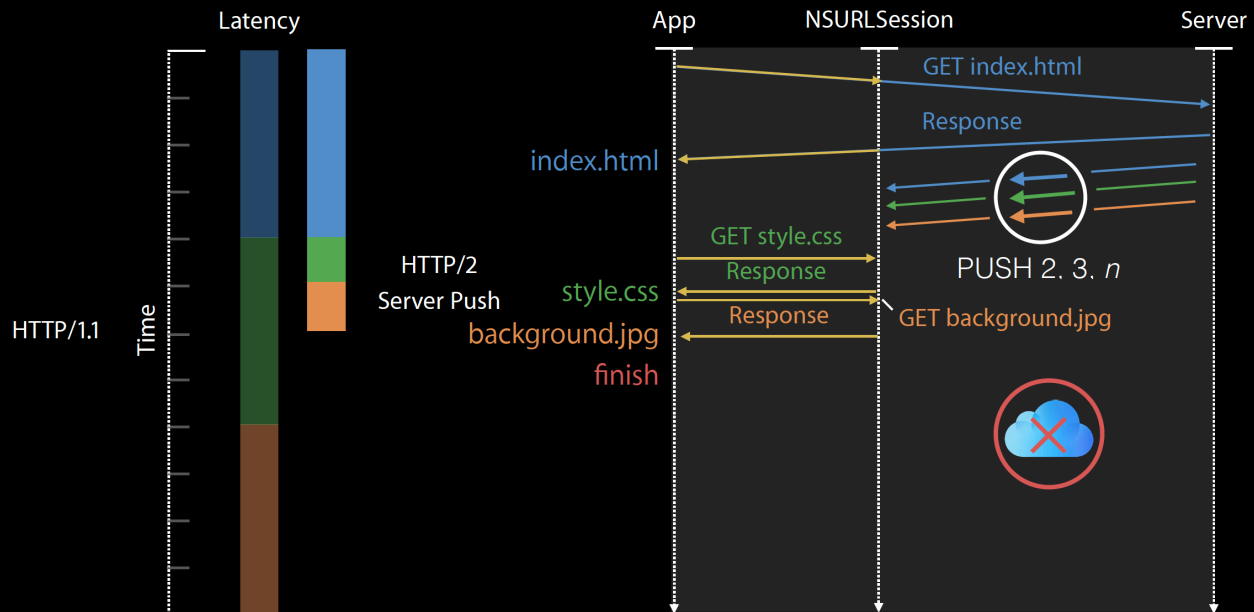
- Prevent network round trips
 - 클라이언트가 네트워크를 통해 개별 리소스를 가져오는 대신 서버는 원래 응답과 함께 클라이언트가 필요로 하는 추가 리소스에 대한 정보를 넣을 수 있습니다. 따라서 네트워크 왕복을 방지한다고 설명합니다.
- Server support required
 - 서버에서 HTTP/2를 지원해야 하며 서버 푸시(Server Push) 기능을 사용하도록 설명합니다.
- Now available in NSURLSession
 - 서버 푸시(Server Push)는 NSURLSession을 사용하는 모든 애플리케이션에서 사용 가능하다고 설명합니다.
- 스크린샷



HTTP/2 Server Push

NEW

Loading



Network statistics

- NSURLSessionTaskMetrics

- NSURLSessionTaskDelegate 클래스의 델리게이트 메서드를 구현해 측정기준(매트릭스)에 수집된 태스크(작업)와 새 클래스 객체가 전달됩니다. NSURLSessionTaskMetrics 클래스에는 taskInterval, redirectCount, transactionMetrics 의 프로퍼티가 있습니다.
- taskInterval 프로퍼티는 태스크(작업)생성에서 모든 통계가 수집되어 didFinishCollectingMetrics 델리게이트에게 전달 될 준비가 된 시점까지의 시간 간격입니다.
- redirectCount 프로퍼티는 작업 실행 중 HTTP 리다이렉션이 발생한 횟수입니다.
- transactionMetrics 프로퍼티는 NSURLSessionTaskTransactionMetrics 객체의 배열을 가지고 있다고 설명합니다.

- NSURLSessionTaskTransactionMetrics

1. Request and Response

- request, response 프로퍼티를 이용해 정말로 내가 요구한 것이 무엇인지 분석할 수 있게 도와준다고 설명합니다.

2. Protocol and Connection

- networkProtocolName 의 프로퍼티는 통계가 수집된 시간에 사용된 프로토콜 유형(HTTP/1.1, H2, SPDY/3, SPDY/3.1)을 알려준다고 설명합니다.
- isProxyConnection 프로퍼티는 통계가 수집되었거나 수집되는 동안 트랜잭션이 부분적인지 또는 프록시 연결의 여부를 알려준다고 설명합니다.

- `isReusableConnection` 프로퍼티는 리소스를 가져오는 중에 지속적인 연결이 사용된 경우 Yes로 설정된다고 설명합니다.

3. Load Info

- `resourceFetchType` 프로퍼티인 `networkLoad` 값은 리소스를 어떻게 얻어졌는지 알려 줍니다. `localCache` 에서 리소스를 가져와 볼 수 있습니다. 로컬로 접근할 경우 네트워크 트랜잭션이 필요하지 않다고 설명합니다. 그리고 `serverPush` 값은 서버 푸시(Server Push)요청을 했을때 리소스가 캐시 결과로 발견되었음을 알려주는 메시지라고 설명합니다.

4. Connection Establishment and Transmission

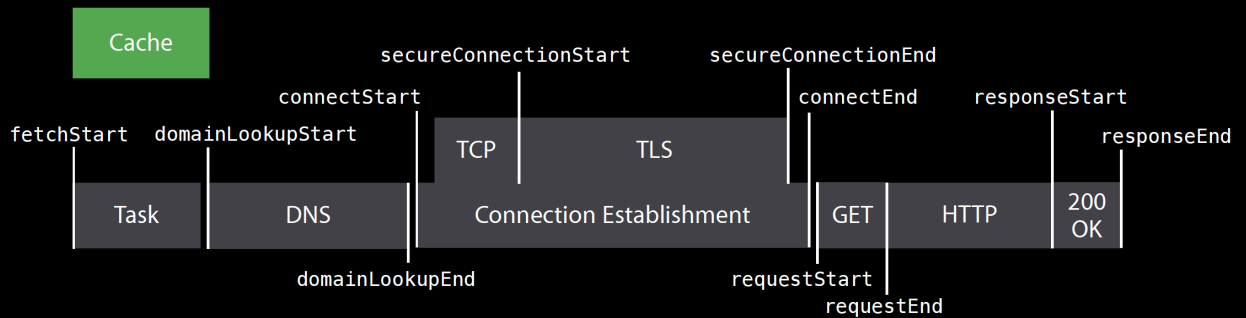
- `fetchStart` 는 애플리케이션이 리소스를 요청을 시작하는 시간입니다.
- `domainLookupStart` 은 리소스에 대한 이름검색이 시작되기 직전의 시간입니다. DNS쿼리입니다.
- `domainLookupEnd` 은 해당 조회가 완료되는 시점입니다.
- `connectStart` 는 애플리케이션이 원격 서버와 TCP 연결을 시작하거나 설정하기 직전의 시간입니다. (응답이 로컬 캐시에서 발견되면 값은 0이 될 수 있습니다.)
- `secureConnectionStart` 은 애플리케이션이 현재 연결을 보호하기 위해 보안 핸드셰이크(handshake)를 시작하기 바로 전의 시점입니다.
- `secureConnectionEnd` 는 보안 핸드셰이크(handshake)가 완료되었을 때입니다.
- `connectEnd` 는 보안 핸드셰이크(handshake)를 포함하여 애플리케이션이 원격 서버에 연결된 직후입니다.
- `requestStart` 는 애플리케이션이 로컬 캐시 또는 원격 서버에 리소스를 가져왔는지 여부에 관계없이 애플리케이션 리소스 요청을 시작하는 시간입니다.
- `requestEnd` 는 요청의 마지막 바이트가 네트워크에 기록된 시간입니다.
- `responseStart` 는 응답의 첫 번째 바이트가 서버로부터 수신된 시간입니다.
- `responseEnd` 는 애플리케이션이 요청된 리소스의 마지막 바이트를 수신한 직후의 시간입니다.

• 스크린샷

NSURLSession API

NEW

NSURLSessionTaskTransactionMetrics



Security

NSURLSession API의 보안 기능인 Transport Layer security (TLS) 와 App Transport Security에 관해 설명하고 몇가지 개선사항을 알려줍니다.

- Transport Layer security
 - 전송 계층 보안 또는 TLS는 네트워크를 통해 끝점에서 전송되는 데이터를 보호하는 프로토콜입니다. TLS의 암호는 한쪽에서 데이터를 스크램블 하고 다른 수신 측은 동일한 암호를 사용하여 해당 데이터를 해독해 이를 사용할 수 있도록 한다고 설명합니다. 그중 변경된 사항은 더이상 RC4 암호를 지원하지 않는다고 합니다. 더 자세한 내용은 보안 세션에서 참고 가능하다고 설명합니다.
- App Transport Security
 - 애플리케이션 전송 보안 또는 ATS에서 사용 권한을 부여하는 키를 이용하여 애플리케이션의 정책, 보안 정책을 설정할 수 있다고 설명합니다.