

# CoAP Observe: 예제 및 활용

2017년 1월

경북대학교 통신프로토콜연구실

정중화 ([godopu16@gmail.com](mailto:godopu16@gmail.com)), 최동규 ([supergint@gmail.com](mailto:supergint@gmail.com))

## 목 차

1. 서론 .....	2
1.1 CoAP 이란? .....	3
1.2 CoAP OBSERVE 란? .....	5
2. COAP OBSERVE 예제 구현 .....	6
2.1 JDK 설치 .....	6
2.2 CALIFORNIUM 라이브러리 가져오기 .....	8
2.3 CoAP OBSERVE 예제 가져오기 .....	8
2.4 CoAP OBSERVE 예제 테스트 .....	9

## 1. 서론

사물인터넷 (IoT : Internet of Things) 의 개념은 1999년 케빈 애쉬톤 (Kevin Ashton) 에 의해 최초로 사용되었다. 초기에 케빈 애쉬톤이 사용을 하던 개념은 RFID 태그를 활용한 시스템의 발전을 시작으로 개념이 조금씩 변화되어, 최근에는 유비쿼터스 컴퓨팅을 포함하여 생활 속 유무선 네트워크로 연결 할 수 있는 모든 사물들로 범주가 커졌다. 가전제품, 전자기기뿐만 아니라 헬스 케어, 스마트 홈, 스마트 자동차, 원격제어 등 많은 분야에서 사물을 네트워크로 연결해 편리하게 사람과 정보를 교환 할 수 있다. 기존에는 사물 본연의 목적만으로 단일적인 기능을 했다면 인터넷과의 융합으로 다양한 서비스를 제공한다. 구글의 구글 글라스, 나이키의 퓨어 밴드, 그레이트 리버 메디컬 센터 의료 플랫폼, 미국 벤처기업 코벤티스가 개발한 심장박동 모니터링 기기 등 모두 사물인터넷의 기반을 둔 예이다. 미국의 정보 기술 연구 및 자문 회사인 가트너에 의하면 아래의 그림 1과 같이 2020년까지 사물인터넷 시장 규모는 1.9조 달러 수준으로 성장 할 것으로 예상하고 있다.

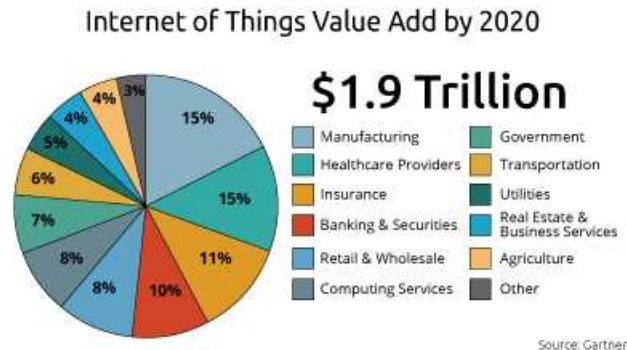


그림 1. 2020년 사물인터넷 시장 예상 도표

사물의 요소는 우리가 일반적으로 사용하는 전자기기 및 가전제품뿐만 아니라 경량화 된 CPU, 센서 등도 포함 하고 있기에 현재 인터넷 기술인 TCP/IP 프로토콜 스택을 맞추기에는 불가능하다. 따라서 낮은 성능의 사물인터넷 기기도 인터넷 연결이 가능하도록 인터넷 연결 기술도 현재보다 경량화 되어야 한다.

현재 국제적인 획일화 된 표준은 존재하지 않지만 ITU-T, oneM2M, 3GPP, IETF, IEEE 등의 다양한 표준화 기구에서 국제적인 표준을 확립하기 위해 글로벌한 IoT/M2M 서비스 기술을 연구하고 있다. ITU-T와 oneM2M 등은 주로 스마트 헬스케어나 스마트 홈과 같은 서비스 플랫폼 표준 기술을 개발 하고있으며, 3GPP, IETF, IEEE 등의 표준 기구는 네트워크 통신을 위

한 프로토콜을 주로 연구한다. 현재 표준화의 선두주자인 3GPP는 기대만큼의 수익이 나오지 않아 소강상태에 머무르고 있으며, IETF는 대표적인 프로토콜로 CoAP을 핵심적으로 사용하고 있다. IEEE는 원활한 M2M 통신을 제공하기 위해 IEEE 802.x 계열의 무선통신 기술을 확장하는 표준 개발이 이루어지고 있다. 국내에서의 표준 활동은 TTA를 중심으로 M2M 서비스 요구사항, 이동통신 무선 접속 기술, 표준 플랫폼 간의 인터페이스 등을 표준화 하는 작업을 ITU-T, oneM2M 등과의 국제표준화 공동협력을 추진 하고 있다.

## 1.1 CoAP 이란?

CoAP은 사물인터넷과 같이 대역폭이 제한된 통신 환경에 최적화하여 개발된 REST (REpresentational State Transfer) 기반의 자원 발견, 멀티캐스트 지원, 비동기 트랜잭션 요청 및 응답 등을 지원하기 위한 경량 메시지 전송 프로토콜이다.

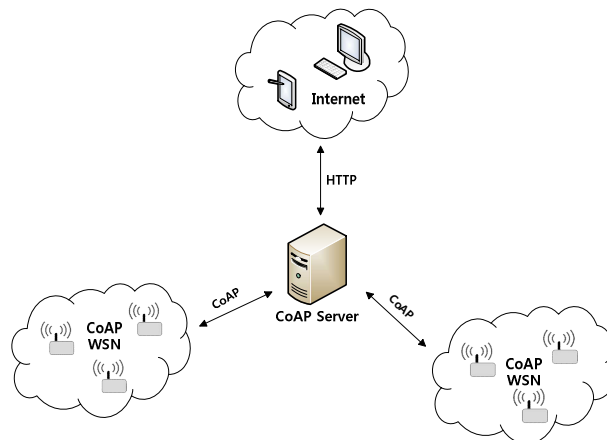


그림 2. CoAP 시스템 개요도

CoAP은 RESTful기반의 프로토콜이므로 기존의 HTTP (Hypertext Transfer Protocol) 웹 프로토콜과 연동이 쉽다. CoAP은 자원 관리를 위해 HTTP와 동일하게 GET, PUT, POST, DELETE의 메서드를 사용한다.

CoAP은 전송 계층에서 UDP (User Datagram Protocol) 를 사용하며 송신자 수신자 간에 정보를 주고받을 때 자료를 보낸다는 신호나 받는다는 신호를 보내지 않는 비동기식 방식의 전송방법이다. CoAP은 수신자 쪽에서 자료를 받았는지 받지 않았는지 확인 할 수 없는 UDP의 단점을 극복하기 위해 재전송 및 타이머 관리를 옵션으로 포함하고 있다. CoAP은 확인형 (Confirmable), 비확인형 (Non-confirmable), 승인 (Acknowledgement), 리셋 (Reset) 4가지의 메시지 형을 정의하고, 요청 (Request) 과 응답 (Response) 메시지의 상호작용으로 전달된다.

Application	CoAP		HTTP	
Transport Layer	UDP		TCP	
Internet Protocol Layer	IPv6	IP Routing	IPv4	IPv6
	6LoWPAN			
Network/Link Layer	MAC		MAC	
	PHY		PHY	

그림 3. CoAP과 HTTP 프로토콜 스택 비교

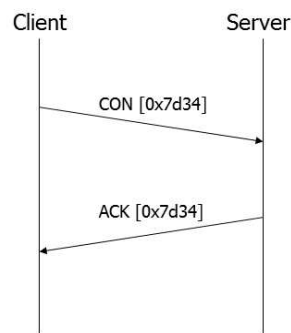


그림 4. CoAP 신뢰성 메시지 전송

그림 4는 신뢰성 메시지 전송 방법이다. 확인형 (CON) 메시지를 전송하고, 여기에 포함된 메시지 식별자 (Message ID) 는 승인 (ACK) 메시지에도 포함된다. 수신자가 자료나 요청을 잘 전달 받았으면 승인 메시지를 전송하고, 만약 수신자가 수신한 자료나 요청을 처리할 수 없을 때는 승인 메시지 대신 리셋 (Reset) 메시지를 전송한다.

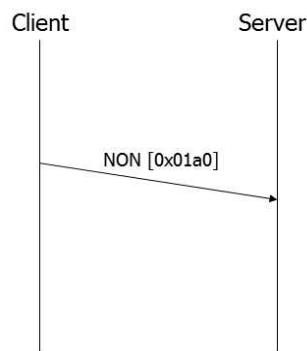


그림 5. CoAP 비신뢰성 메시지 전송

주기적으로 센서 자료 값을 얻어오는 것처럼 모니터링 방식에서는 메시지의 신뢰성을 요구하지 않는다. 이런 때에는 그림 5와 같이 비확인성 메시지 (NON) 을 전송한다. 이 때 메시지 ID는 메시지 중복 검사를 위해 사용 한다. 만약 수신자가 비확인성 메시지를 처리할 수 없을 때는 신뢰성 메시지 전송과 같은 방식으로 리셋 메시지를 전송한다.

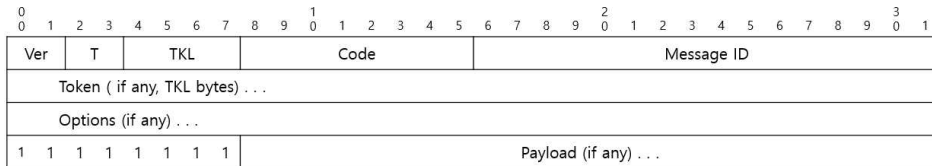


그림 6. CoAP 메시지 형식

CoAP 메시지는 그림 6과 같이 간단한 이진 형식으로 인코딩 되어 전송 된다. 고정적인 4 바이트와 0바이트에서 8바이트 길이의 토큰, 그리고 옵션과 페이로드 순으로 배치된다. 처음 Ver 부분은 2비트로 CoAP의 현재 버전을 나타낸다. 두 번째 T 부분은 2비트로 메시지 형을 의미하며 '0'은 확인형, '1'은 비확인형, '2'는 승인, '3'은 리셋을 나타낸다. TKL 부분은 4 비트로 토큰 필드의 길이를 나타내며 값의 단위는 바이트이고 0에서 8까지의 수를 사용 한다. 다음으로 Code 부분은 1바이트로 3비트는 클래스를 나타내고, 5비트는 자세한 내용을 나타낸다. 클래스 값의 경우 '0'은 요청, '2'는 성공적인 응답, '4'는 클라이언트 에러 응답, '5'는 서버 에러 응답이다. 요청의 경우, '0.01'은 GET, '0.02'는 POST, '0.03'은 PUT, '0.04'는 DELETE, '0.00'은 빈 메시지이다. 메시지 ID는 메시지의 중복 확인 및 확인성/비확인성 메시지에 대한 짝으로 승인/리셋 메시지에 사용된다. Token과 Options는 필요에 따라 포함될 수 있고 포함되지 않을 수 있다. Token과 Options의 끝은 Payload 마커로 알 수 있고, Payload 마커는 0xFF이다. 현재 정의된 Options는 Content-Format, ETag, Location-Path, Location-Query, Max-Age, Proxy-Uri, Proxy-Scheme, Uri-Host, Uri-Path, Uri-Port, Accept, If-Match, If-Non-Match, Size1이 있다.

## 1.2 CoAP Observe 란?

CoAP과 같이 요청/응답 모델은 클라이언트가 짧은 주기로 메시지를 반복해서 보내는 경우에는 적합하지 않다. 이러한 상황에서 기존의 HTTP를 사용한다면 자원이 제한 된 저성능의 기기에서는 큰 오버헤드와 복잡성 때문에 문제가 발생한다. CoAP은 이러한 문제를 해결 하기 위해 기존의 CoAP을 확장시켜 Observe라는 기능을 개발하였다. 이는 CoAP 서버에 주제 (Subject) 라는 공간을 할당하여 그 주제에 관심 이는 클라이언트가 주제를 통해 서버로

접근한다. 특정 주제에 클라이언트가 Observe 요청을 한다면 그 주제에 관한 내용이 바뀔 때 마다 별도의 요청 메시지 없이 서버로부터 변경 된 내용을 알림 (Notifications) 으로 받게 된다. 이러한 기능이 CoAP Observe 기능이다.

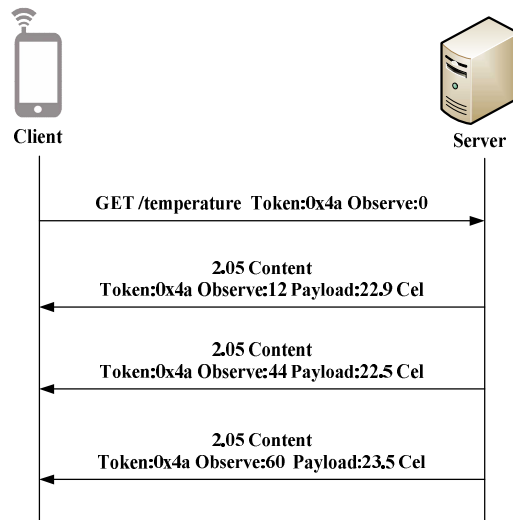


그림 7. CoAP Observe 기능 예시

그림 7은 온도에 관한 정보를 받는 Observe 기능의 예시이다. 우선 클라이언트는 특정 주제에 관한 내용을 Observe 하고 싶다고 서버에게 확장된 GET 메시지를 통해 요청한다. 서버는 Observer 리스트에 클라이언트의 정보를 저장하고, 그에 관한 응답으로 특정 주제의 원하는 정보를 담아 메시지를 보낸다. 이렇게 Observe 과정이 끝나게 되면, 특정 주제에 관한 상태가 변경 되었을 때 변경 된 정보를 서버가 별다른 요청 없이도 클라이언트에게 알려 준다.

## 2. CoAP Observe 예제 구현

CoAP Observe를 사용하기 위해서는 CoAP 라이브러리를 사용하여야 합니다. 여기에서는 Californium 이라는 JAVA용 라이브러리를 사용합니다.

### 2.1 JDK 설치

Californium 라이브러리는 JAVA용 라이브러리기 때문에 라이브러리를 사용하기 위해서는 JDK가 필요하다. JAVA는 Oracle이 소유하고 있으므로 Oracle 홈페이지로 이동하여 다운로드 받을 수 있다.

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

주소로 접속하여 JDK 설치 파일을 다운로드 한 후 설치한다.

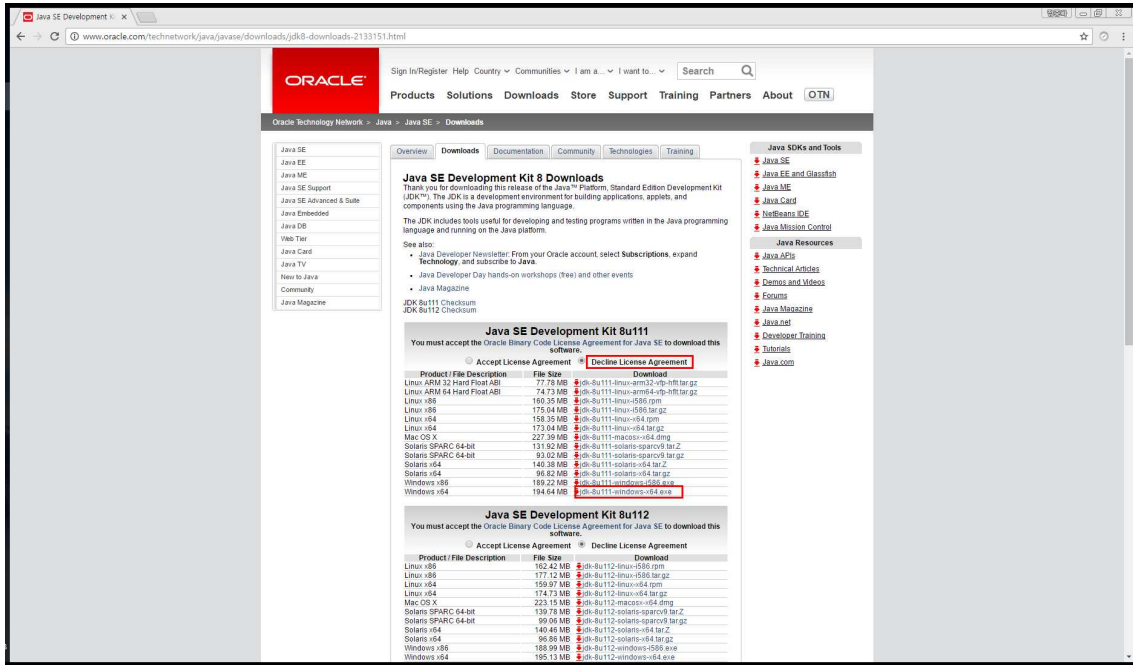


그림 8. JDK 다운로드 웹 페이지

## 2.2 Californium 라이브러리 가져오기

Californium 라이브러리를 사용하기 위해서는 Californium (Cf) Core 라이브러리를 다운로드 하여야 한다. 해당 라이브러리는 californium-core 라이브러리 파일과 element-connector 라이브러리 파일로 구성되어 있으며, californium-core 라이브러리 파일은

<http://mvnrepository.org/artifact/org.eclipse.californium/californium-core/2.0.0-M2> 로 이동하여

다운로드 할 수 있으며, element-connector 라이브러리 파일은

<http://mvnrepository.org/artifact/org.eclipse.californium/element-connector/2.0.0-M2> 로 이동

하면 다운로드 할 수 있다.

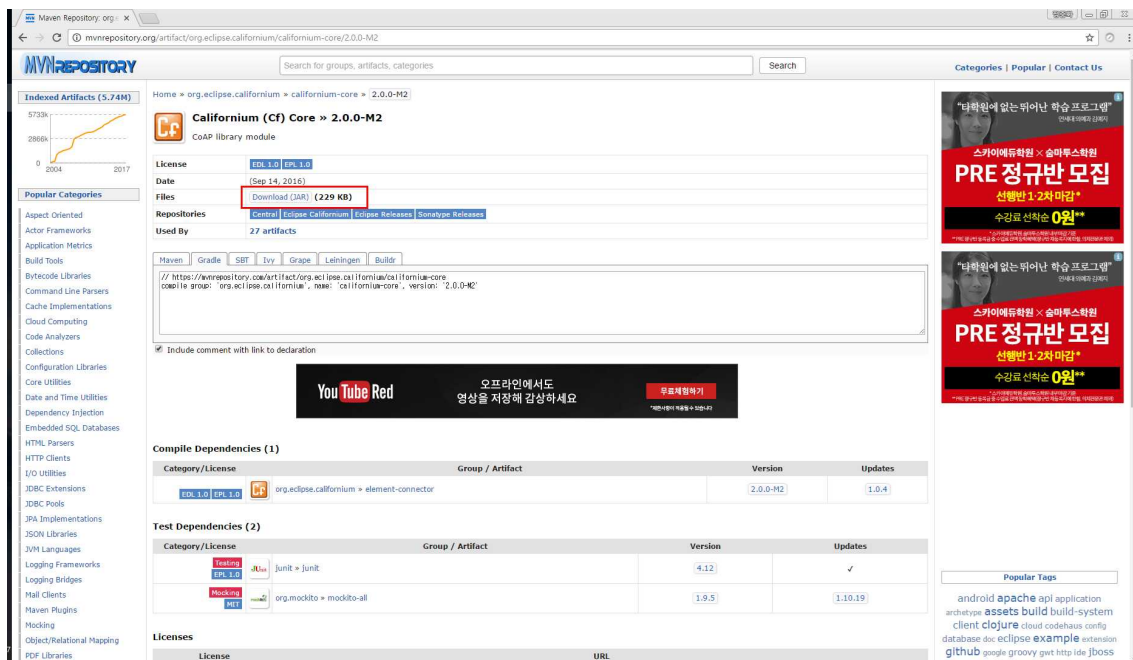


그림 9. Californium library 다운로드 웹 페이지

## 2.3 CoAP Observe 예제 가져오기

CoAP 라이브러리 Californium을 이용한 예제를 살펴보자. Eclipse를 이용하여 프로젝트를 Import 한 후 실행시켜 본다. 예제는 <http://protocol.knu.ac.kr> 에 방문하면 다운로드 받을 수 있다. 다운로드한 파일의 압축을 푼 후 해당 디렉토리를 Workspace로 하여 이클립스를 실행한다. 그 후 아래 그림 10의 절차를 따라하여 프로젝트를 import 한다.



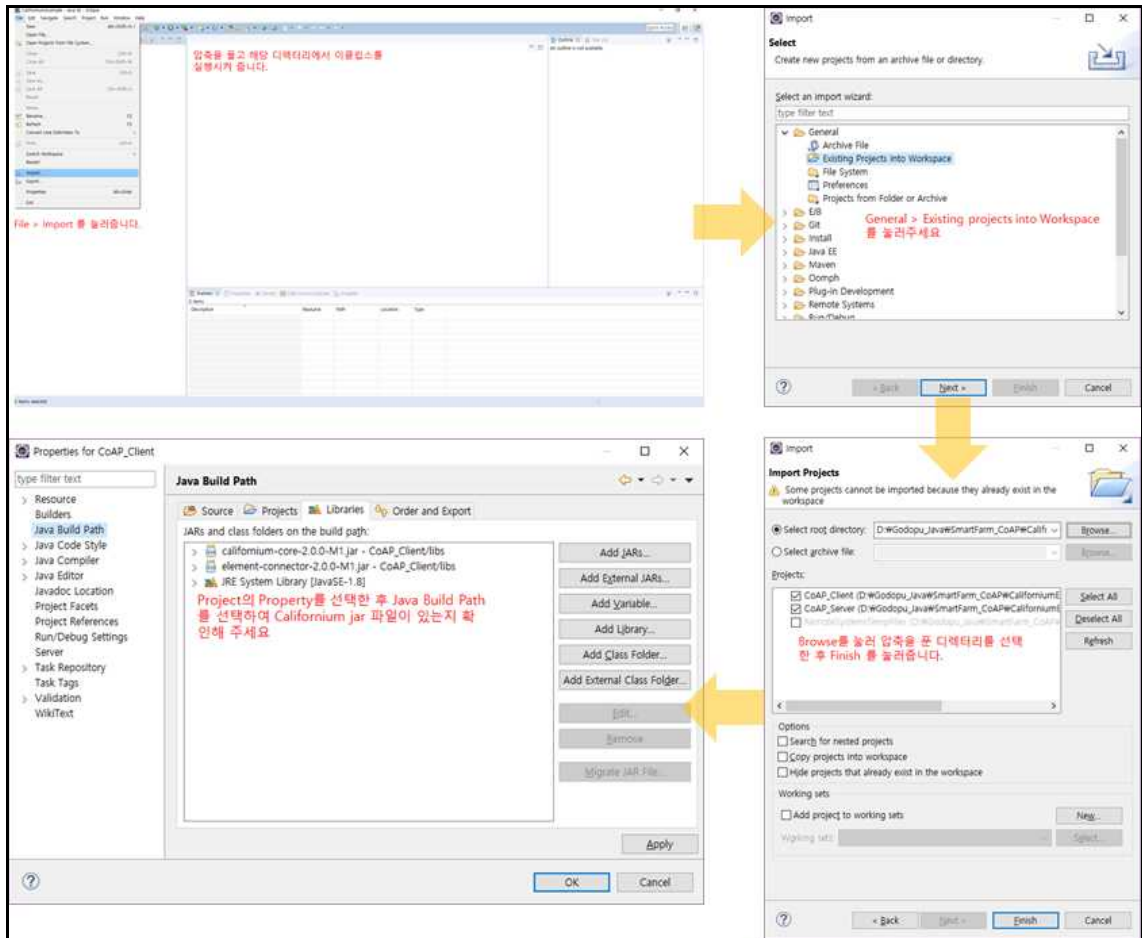


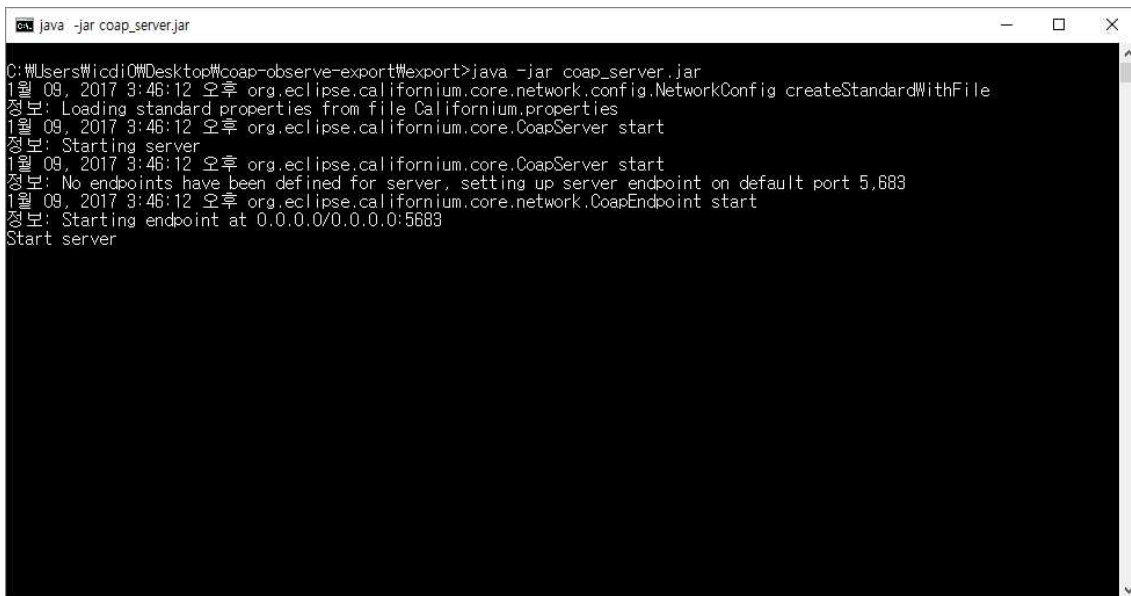
그림 10. 프로젝트 가져오기 절차

## 2.4 CoAP Observe 예제 테스트

앞서 다운로드한 폴더 안의 export 폴더로 이동한 뒤 파일 > 명령 프롬프트 열기를 눌러 해당 디렉터리에서 CMD창을 연다. 가장 먼저 브로커 역할을 하는 서버를 실행시켜야 한다. coap\_server.jar를 아래의 명령어와 함께 CoAP Observe Server 예제를 실행할 수 있다.

- 1) CMD를 실행합니다.
- 2) coap\_server.jar 를 실행합니다.  
→ java -jar coap\_server.jar

### coap\_server 실행



```
java -jar coap_server.jar
C:\Users#wicdio\Desktop#coap-observe-export#export>java -jar coap_server.jar
1월 09, 2017 3:46:12 오후 org.eclipse.californium.core.network.config.NetworkConfig createStandardWithFile
정보: Loading standard properties from file Californium.properties
1월 09, 2017 3:46:12 오후 org.eclipse.californium.core.CoapServer start
정보: Starting server
1월 09, 2017 3:46:12 오후 org.eclipse.californium.core.CoapServer start
정보: No endpoints have been defined for server, setting up server endpoint on default port 5683
1월 09, 2017 3:46:12 오후 org.eclipse.californium.core.network.CoapEndpoint start
정보: Starting endpoint at 0.0.0.0/0.0.0.0:5683
Start server
```

그림 11. coap\_server.jar 실행 화면

이제 위와 같이 두 개의 CMD 창을 더 실행시킨 뒤 한 개는 coap\_sub.jar를 실행하고 한 개는 coap\_pub.jar를 수행하여 subscriber가 publisher가 전송한 메시지를 잘 수신하는지를 확인해보자.

- 1) CMD를 실행합니다.
- 2) coap\_sub.jar 를 실행합니다.  
→ java -jar coap\_sub.jar localhost topic

### coap\_sub.jar 실행

```

C:\WINDOWS\system32\cmd.exe - java -jar coap_sub.jar 155.230.105.168 topic1
정보: Loading standard properties from file Californium.properties
1월 09, 2017 3:52:26 오후 org.eclipse.californium.core.network.CoapEndpoint start
정보: Starting endpoint at 0.0.0.0/0.0.0.0:0
1월 09, 2017 3:52:26 오후 org.eclipse.californium.core.network.EndpointManager createDefaultEndpoint
정보: Created implicit default endpoint 0.0.0.0/0.0.0.0:56136
==[ CoAP Response ]=====
MID      : 22861
Token    : 8cc5554ffc
Type     : ACK
Status   : 2.01
Options  : {}
Payload  : 0 Bytes
=====
Start Subscription topic - topic1
NOTIFICATION:

```

그림 12. coap\_sub.jar 실행 화면

Topic 과 함께 서버로 메시지를 전송한다.

- 1) CMD를 실행합니다.
  - 2) coap\_pub.jar 를 실행합니다.
- java -jar coap\_pub.jar localhost topic "Hello world"

### coap\_pub.jar 실행

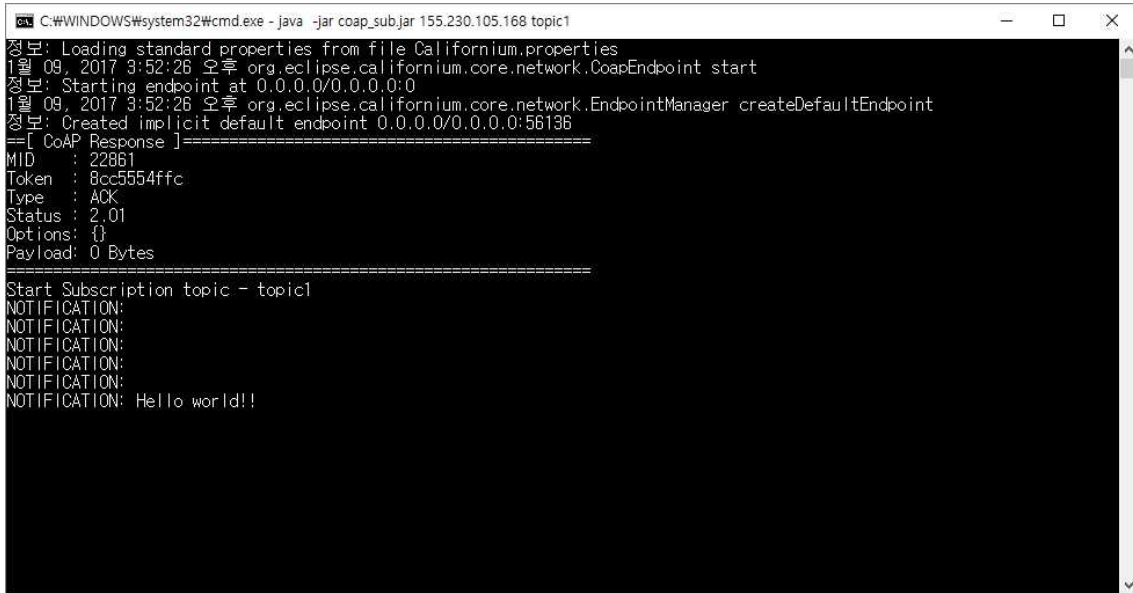
```

C:\WINDOWS\system32\cmd.exe
C:\Users\wicdi0\Desktop\coap-observe-export\export>java -jar coap_pub.jar 155.230.105.168 topic1 "Hello world!!"
1월 09, 2017 3:58:34 오후 org.eclipse.californium.core.config.NetworkConfig createStandardWithFile
정보: Loading standard properties from file Californium.properties
topic1
1월 09, 2017 3:58:34 오후 org.eclipse.californium.core.network.CoapEndpoint start
정보: Starting endpoint at 0.0.0.0/0.0.0.0:0
1월 09, 2017 3:58:35 오후 org.eclipse.californium.core.network.EndpointManager createDefaultEndpoint
정보: Created implicit default endpoint 0.0.0.0/0.0.0.0:57082
==[ CoAP Response ]=====
MID      : 15597
Token    : d08432b6
Type     : ACK
Status   : 2.04
Options  : {}
Payload  : 0 Bytes
=====
CANCELLATION
C:\Users\wicdi0\Desktop\coap-observe-export\export>
C:\Users\wicdi0\Desktop\coap-observe-export\export>

```

그림 13. coap\_pub.jar 실행 화면

coap\_pub.jar 실행 후 다시 coap\_sub.jar 실행 화면을 확인하면 그림 14 와 같이 publisher 가 전달한 메시지가 잘 전달되었음을 확인할 수 있다.



```
C:\WINDOWS\system32\cmd.exe - java -jar coap_sub.jar 155.230.105.168 topic1
정보: Loading standard properties from file Californium.properties
1월 09, 2017 3:52:26 오후 org.eclipse.californium.core.network.CoapEndpoint start
정보: Starting endpoint at 0.0.0.0/0.0.0.0
1월 09, 2017 3:52:26 오후 org.eclipse.californium.core.network.EndpointManager createDefaultEndpoint
정보: Created implicit default endpoint 0.0.0.0/0.0.0.0:56136
==[ CoAP Response ]=====
MID      : 22861
Token    : 8cc5554ffc
Type     : ACK
Status   : 2.01
Options  : {}
Payload  : 0 Bytes
=====
Start Subscription topic - topic1
NOTIFICATION:
NOTIFICATION:
NOTIFICATION:
NOTIFICATION:
NOTIFICATION:
NOTIFICATION: Hello world!!
```

그림 14. 메시지 수신 화면