



# 200

## Oracle 퍼블릭 클라우드 워크샵

Cloud Native Continuous  
Delivery of Java Microservices

2016년 11월 23일

## 도입

오라클 퍼블릭 클라우드 DevOps 클라우드 네이티브 마이크로 서비스 워크샵의 일부인 여러 Lab 중 두 번째 워크샵입니다. 이 워크샵에서는 여러 Microservices 를 만들고 사용하는 클라우드 네이티브 프로젝트에 대한 SDLC (Software Development Lifecycle)를 안내합니다.

첫 번째 lab 100 에서 프로젝트 관리자는 개발자 클라우드 서비스에 새 프로젝트를 만들고 프로젝트에 팀 구성원을 추가하고 이 응용 프로그램 개발자에게 작업을 생성 및 할당했습니다. 이 실습에서는 Java 개발자가 될 것입니다. Java 개발자는 필요한 프론트 엔드 또는 분석 구성 요소에 데이터를 제공하는 여러 가지 마이크로 서비스를 작성해야 합니다

## 목표

- Developer Cloud Service 접근
- 외부 Git Repository 의 코드 가져오기
- Eclipse 에 프로젝트 가져오기
- 개발자 클라우드 서비스 및 Oracle Application Container Cloud Service 를 이용하여 프로젝트 Build 및 Deploy

## 필수 요건

- 다음은 강사가 제공하는 Oracle Public(공용)클라우드 계정을 필요로 합니다. 당신은 최신 버전의 이클립스를 다운로드하고 설치해야 할 것이다.

## 개요

도입 .....	<b>2</b>
목표 .....	2
필수 요건 .....	2
개요 .....	3
<b>최초의 정적인 트위터 피드 만들기 .....</b>	<b>4</b>
Developer 클라우드 서비스 살펴보기 .....	4
초기 Git Repository 만들기 .....	11
기본 빌드 및 구축 프로세스 생성.....	15
Twitter 피드/트위터 구축을 확인합니다.....	23
<b>정적 트위터 피드 서비스에 필터 추가.....</b>	<b>28</b>
이클립스 IDE 프로젝트 클론 생성 .....	28
Local Cloned Service 테스트 .....	36
필터를 서비스에 추가.....	40
Local Filtered Service 테스트 .....	44
<b>새 Branch 및 Commit 코드 생성.....</b>	<b>45</b>
Branch 와 Commit 코드 생성 .....	45
병합 요청 생성 .....	48
리자 존스로 branch Merge.....	51
클라우드에서 테스트를 수행합니다.....	57
<b>Supplementary Assignment – Twitter Live Feed Credentials 60</b>	
트위터 앱 만들기 .....	60
<b>Appendix 2 – Installing Eclipse.....</b>	<b>69</b>
이클립스 다운로드 및 설치 .....	69
필요에 따라 구성 요소를 자동으로 구성합니다.....	69

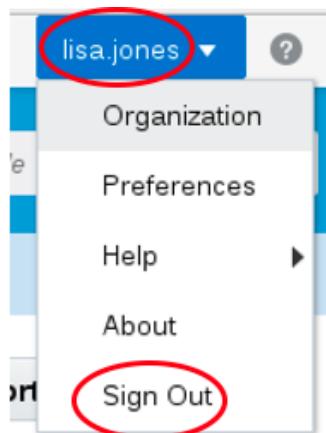
## 최초의 정적인 트위터 피드 만들기

### Developer 클라우드 서비스 살펴보기

참고: 여러 사용자(예. Lisa.Jones, Bala.Gupta, John.Dunbar)를 사용하여 이 랙을 실행하지 않는 경우 1 단계, 2 단계, 3 단계, 4 단계를 건너뛰고 5 단계로 이동합니다. 그러나 lab 100 을 완료한 상태인 사용자로 로그인 한 경우에도 개발자인 Bala.Gupta 로 로그인한 것으로 간주하셔야 합니다.

#### STEP 1: Oracle Cloud 에 Bala.Gupta 계정으로 로그인하십시오.

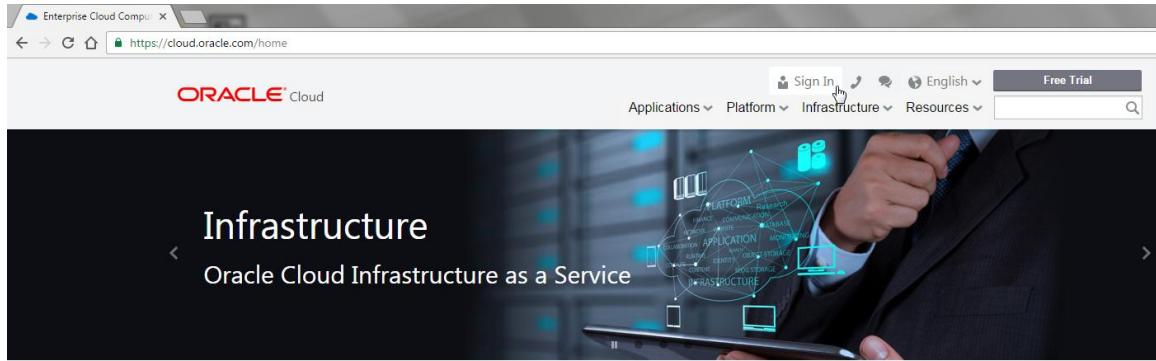
- lab 100 을 방금 완료했거나 Lisa.Jones 로 로그인 한 경우 이 lab 을 계속하기 전에 먼저 로그 아웃해야합니다. 화면 오른쪽 상단의 사용자 이름 (lisa.jones)을 클릭 한 다음 드롭 다운 메뉴에서 로그 아웃을 선택하여 로그 아웃합니다.



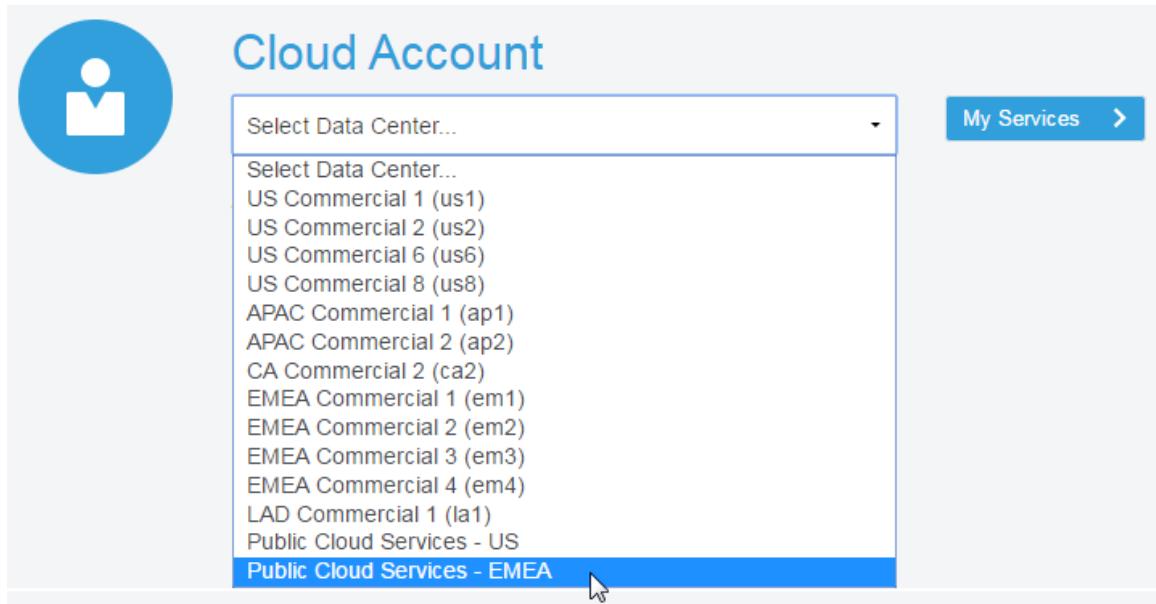
- 이제 다시 로그인할 수 있습니다. 브라우저에서 다음 URL 로 이동합니다.

<https://cloud.oracle.com>

- 브라우저의 오른쪽 상단 모서리에 있는 Sign In(로그인)을 클릭합니다.



- 중요-내 서비스에서 드롭 다운 목록에서 선택할 영역을 강사에게 요청하고 MyServices(내 서비스)버튼을 클릭합니다.



- ID 도메인을 입력하고 **Go(이동)**를 클릭합니다.

참고: 강사로 부터 사용자 이름, 사용자 이름 및 암호 값이 부여 받습니다.

Enter your Identity Domain

Go

- ID 도메인이 설정되면 사용자 이름과 암호를 입력하고 **Sign In(로그인)**을 클릭합니다.

참고 : 이 lab 의 첫 번째 부분에서는 Java Developer Bala Gupta 로 활동하게 됩니다.  
이전 lab 과 마찬가지로 여러 사용자를 지원할 수 없다면 지원되는 사용자로 로그인하고  
Java Developer 인 Bala Gupta 의 "논리적"식별을 가정합니다.

Welcome gse00002057 [change domain](#) ?



Can't access your account?

Sign In

- 이 계정에 사용 가능한 다양한 클라우드 서비스를 표시하는 Dashboard 가 표시됩니다.

Account gse00002055 • ?

0 Important Notifications

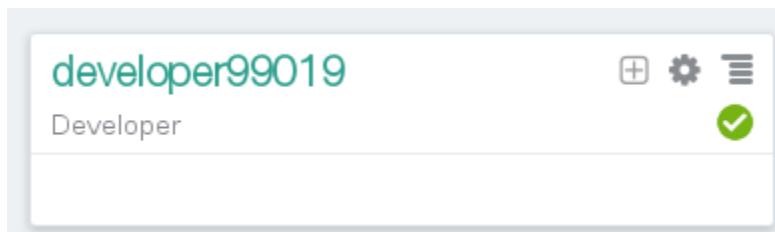
Create Instance Customize Dashboard

developer99019 Developer	Database Instances	Compute Instances	Java Instances
Application Container Instances	Database Backup Instances	Storage Instances	SOA

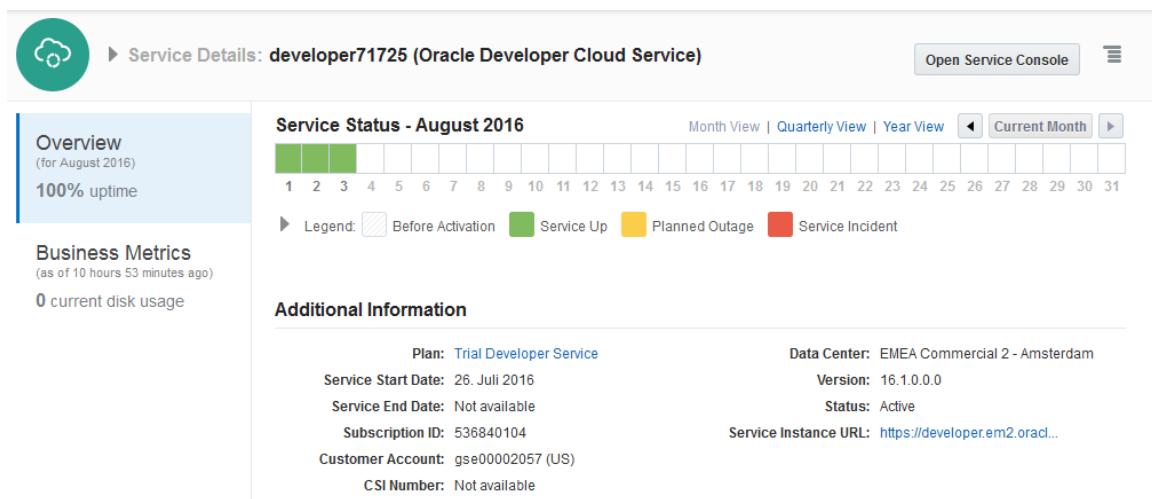
## STEP 2: 개발자 클라우드 서비스에 로그인

Oracle Developer Cloud Service 는 팀 개발 프로세스를 간소화하고 소프트웨어 제공을 자동화하는 완벽한 개발 플랫폼을 제공합니다. 통합 플랫폼에는 Issue 추적 시스템, 민첩한 개발 대시 보드, 코드 버전 관리 및 코드 검토 플랫폼, 지속적인 통합 및 전달 자동화는 물론 위키 및 라이브 활동 스트림과 같은 팀 공동 작업 기능이 포함됩니다. 풍부한 웹 기반 대시 보드와 널리 사용되는 개발 도구와의 통합을 통해 Oracle Developer Cloud Service 는 더 나은 애플리케이션을 보다 신속하게 제공 할 수 있도록 지원합니다.

- Cloud UI 대시 보드에서 **Developer** 서비스를 클릭합니다. 이 예제에서는 개발자 클라우드 서비스를 **developer99019** 로 하였습니다.



- 서비스 세부 정보 페이지에서는 서비스 상태를 한눈에 파악할 수 있습니다.



- Oracle 개발자 클라우드 서비스에 대한 **Open Service Console** 을 클릭합니다. 서비스 콘솔은 현재 여러분이 소속되어 있는 모든 프로젝트를 나열합니다.

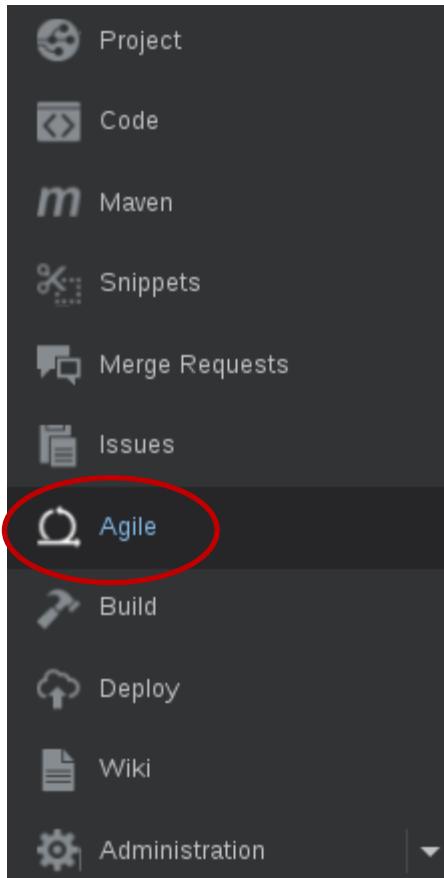
The screenshot shows the Oracle Developer Cloud Service interface. At the top, there are three tabs: 'Member', 'Favorites', and 'All'. A green button labeled '+ New Project' is located in the top right. Below the tabs is a search bar with the placeholder 'Filter Projects' and a magnifying glass icon. The main area displays a project card for 'Twitter Feed Marketing Project', which is described as 'Project to gather and analyze twitter data'. To the right of the project name is a blue star icon. The background shows a dark sidebar with various project management options like Project, Code, Maven, Snippets, Merge Requests, Issues, Agile, Build, Deploy, Wiki, and Administration.

- 프로젝트에 접근하려면 **Twitter Feed Marketing Project** 를 클릭하십시오.

The screenshot shows the detailed view of the 'Twitter Feed Marketing Project'. On the left is a sidebar with project navigation links. The main area has a header 'ORACLE® Developer Cloud Service' and the project name 'Twitter Feed Marketing Project'. Below the header is a 'TODAY' activity stream showing several recent actions by Lisa Jones. To the right is a 'REPOSITORIES' section with a message indicating no git repositories exist. At the bottom is a 'Maven' section with a link to the Maven repository URL.

**STEP 3: Agile Board 리뷰**

- Twitter Feed Marketing Project**에서 Agile을 클릭합니다.

**STEP 4: Microservices 보드 보기**

Microservices Sprint 1 목록이 표시되지 않으면 Sprint 1 작업 목록을 표시하려면 다음을 실행해야합니다. Sprint 1 작업 목록이 표시되면 다음 단계로 건너 뛕니다.

- Microservices 가 기본 보드가 아닌 경우, 현재 보드의 드롭 다운을 클릭하고 All 필터를 선택한 다음 Microservices 를 클릭하십시오.

## Find Board

The screenshot shows a user interface for finding boards. At the top, there are three tabs: 'Owned', 'Favorites', and 'All', with 'All' being highlighted and circled in red. To the right is a green 'New Board' button with a plus sign. Below the tabs is a search bar labeled 'Filter Boards' with a magnifying glass icon. Underneath the search bar are two board entries. The first board, 'Microservices', has a red circle around its name and a note 'No description'. The second board, 'System Board', is described as a 'System board'. Both boards have a blue star icon to their right.

### STEP 5: Active Sprints 표시

- Microservices Board 에서 **Active Sprints** 를 클릭합니다.

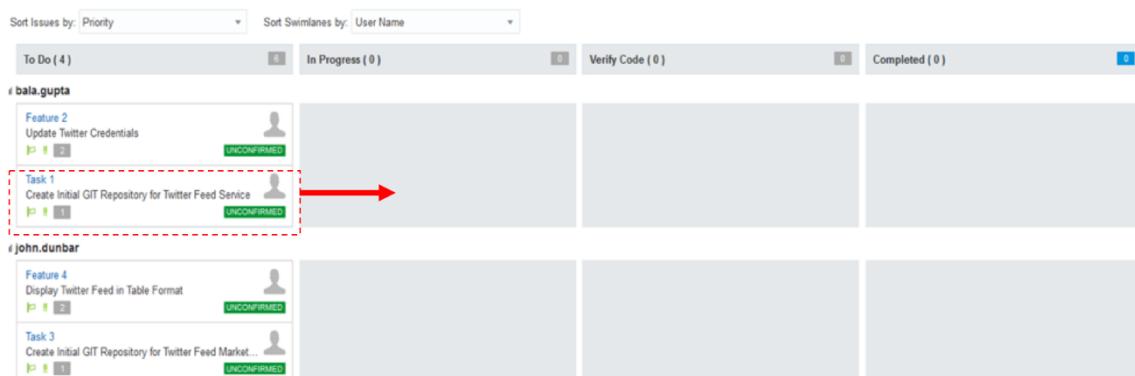
The screenshot shows the 'Active Sprints' view within the Microservices board. At the top, there is a navigation bar with tabs: 'Backlog', 'Active Sprints' (which is highlighted and circled in red), 'Reports', and 'Board'. Below the navigation bar are four status boxes: 'To Do (4)', 'In Progress (0)', 'Verify Code (0)', and 'Completed (0)'. The main area displays two user stories under the heading 'bala.gupta': 'Feature 2: Create Filter on Twitter Feed' and 'Task 1: Create Initial GIT Repository for Twitter F...'. Under the heading 'john.dunbar', there are two more stories: 'Feature 4: Display Twitter Feed in Table Format' and 'Task 3: Create Initial GIT Repository for Twitter F...'. Each story includes a progress bar and a 'UNCONFIRMED' status indicator.

## 초기 Git Repository 만들기

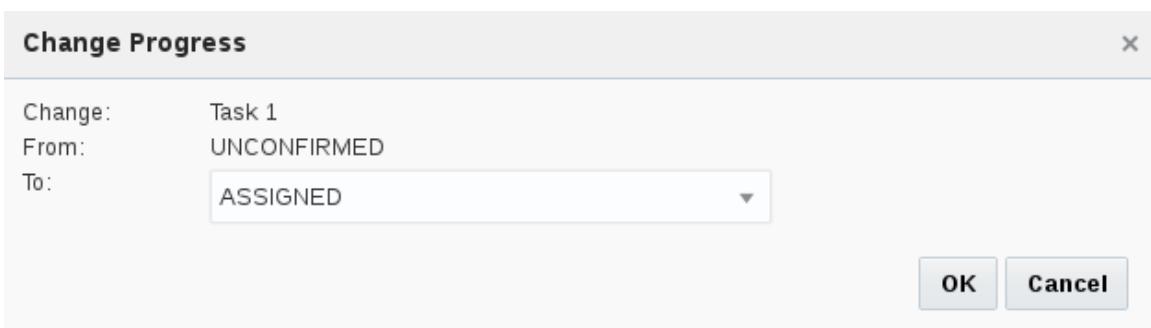
### STEP 6: 초기 Git Repository 만들기

트위터 피드 마이크로 서비스에서 개발을 시작하려면 처음부터 코딩을 시작할 수 있습니다. 그러나 이 프로젝트의 공식 시작 전에 개발자 할당의 타당성을 평가하기 위해 이미 개발자 클라우드 서비스 외부에서 개념 증명 개발을 시작했습니다. 개발자는 기존 코드를 개발자 클라우드 서비스에 가져와 마이크로 서비스의 출발점으로 삼고 싶습니다. 외부 GIT 저장소를 개발자 클라우드 서비스로 복제하면 됩니다. 첫 번째 단계는 애자일 보드를 사용하여 작업을 수락하는 것입니다.

- Task1 - Create Initial GIT Repository for Twitter Feed Service 을 In Progress 영역으로 Drag and Drop 으로 이동시킵니다**



- 진행 상황 확인 팝업에서 **OK(확인)**를 클릭합니다.



The screenshot shows a project management interface with two main sections: "To Do (3)" and "In Progress (1)".

- To Do (3):** Contains one task named "Feature 2" with the sub-description "Create Filter on Twitter Feed". It has a status icon showing 2 issues and is labeled "UNCONFIRMED".
- In Progress (1):** Contains one task named "Task 1" with the sub-description "Create Initial GIT Repository for Twitter Fe...". It has a status icon showing 1 issue and is labeled "ASSIGNED".

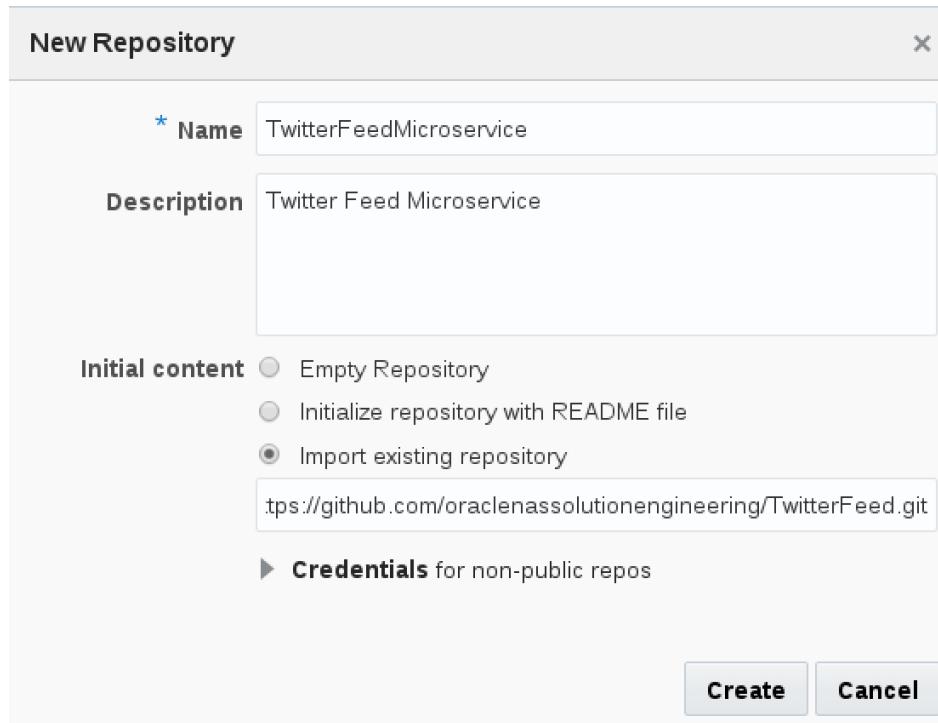
- 좌측 탐색 패널에서 **Project** 를 클릭합니다.
- 새 Git 을 만들려면 **New Repository** 를 클릭하시오.

The screenshot shows the "REPOSITORIES" section of a project management tool.

- Buttons:** "All" (selected), "Favorites".
- Actions:** "+ New Repository" button.
- Search:** "Filter Git Repositories" input field with a search icon.
- Display:** A large "No Repository" icon (a document with a slash) and the message "There are no git repositories to display". Below it, it says "No git repositories exist in this project."
- Maven:** A section showing Maven repository details: "HTTP DAV https://developer.em2.oraclecloud.com/profile/developer71725-gse00002057/s/developer7172".

- New Repository Wizard(새 마법사)마법사에서 다음 정보를 입력하고 Create(생성)을 클릭합니다.

```
Name = TwitterFeedMicroservice
Description = Twitter Feed Microservice
Initial content = Import existing repository
and enter the URL:
https://github.com/oraclenassolutionengineering/TwitterFeed.git
```



- 이제 기존 저장소를 기준으로 새 GIT 저장소를 생성했습니다.

The screenshot shows a GitHub repository interface. At the top, it says "Twitter Feed Marketing Project" with a blue checkmark icon. Below that, there are dropdown menus for "TwitterFeedMicroservice.git" and "master". A toolbar with "Description", "HTTP", "SSH", and "Twitter Feed Microservice" is visible. The main area displays a list of commits:

File/Type	Commit Message
.settings	fixed Eclipse project issues, updated maven plugin versions, added JUnit   philchung
src	Added , and CR to the end of the record in the Tweets list   oraclenassolutionengineering
.classpath	Modified code to support the Static Tweets in the SampleStream, pom,   oracle
.DS_Store	moved everything up a directory and removed jersey-example   philchung
.gitignore	moved everything up a directory and removed jersey-example   philchung
.project	moved everything up a directory and removed jersey-example   philchung
build.sh	moved everything up a directory and removed jersey-example   philchung
DEPLOY.md	moved everything up a directory and removed jersey-example   philchung
deploy.sh	moved everything up a directory and removed jersey-example   philchung
manifest.json	moved everything up a directory and removed jersey-example   philchung
pom.xml	Modified code to support the Static Tweets in the SampleStream, pom,   oracle
README.md	Updated README.md   philchung

## 기본 빌드 및 구축 프로세스 생성

### STEP 7: 기본 빌드 프로세스 생성

이제 관리되는 GIT 저장소에 소스 코드가 있으므로 마스터 branch에 커밋 될 때마다 생성되는 빌드 프로세스를 생성해야 합니다. 우리는 이 섹션에서 Maven 구축 과정을 설정할 것이다.

- 탐색 패널에서 **Build** 를 클릭하여 빌드 페이지에 액세스하고 **New Job** 을 클릭합니다.



You currently have no build jobs defined.



- 새 작업 팝업에서 작업 이름으로 **Twitter Feed Build** 를 입력 한 다음 저장을 클릭하십시오.

New Job

\* Job Name Twitter Feed Build

Create a free-style job  
 Copy existing job

**Save** **Cancel**

- 이제 작업 구성 화면에 배치됩니다.

[« Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters Source Control Triggers Environment Build Steps Post Build Advanced

\* Name Twitter Feed Build

Description

JDK Default (The default Java version in the executing environment)

- Disable build
- Execute concurrent builds if necessary
- Discard old builds

- Configure Build 화면의 메인 탭에서 JDK 를 JDK8 로 변경합니다.

JDK

- JDK 8
- Default (The default Java version in the executing environment)
- JDK 6
- JDK 7
- JDK 8

- Source Control 탭을 클릭합니다. Git 을 클릭하고 드롭 다운에서 **TwitterFeedMicroservice.git** 를 선택합니다.

[« Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters **Source Control** Triggers Environment Build Steps Post Build Advanced

To integrate the Build System with Source Control, select an option below and then configure the required settings.

None  
 Git

**Repositories** Add

\* Repository TwitterFeedMicroservice.git

► Advanced Repository Settings

**Branches** Add

► Advanced Git Settings

- Triggers** 탭을 클릭하고, **Based on SCM polling schedule**를 선택한 후 스케줄로 **\*/1\*\*\*\*\*** 를 추가합니다.

참고:상기의 문구는 매 분마다 설문 조사를 실시하여 변경 사항을 확인합니다. 변경 사항이 있으면 빌드가 트리거 됩니다.

[◀ Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters Source Control Triggers Environment Build Steps Post Build Advanced

When these jobs are built  
 Based on this schedule  
 Based on SCM polling schedule

Schedule `*/1****`

Polling will automatically trigger on commits to the SCM. Specify a schedule only for any additional polling required.

When Maven dependencies have been updated by Maven 3 integration

- Build Steps** 탭을 클릭합니다. **Add Build Step**(빌드 단계 추가)을 클릭하고 **Invoke Maven 3** 를 선택합니다.

[◀ Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters Source Control Triggers Environment Build Steps Post Build Advanced

Configure Build Steps

Add Build Step ▾

- Execute shell
- Invoke Ant
- Invoke Maven 2 (Legacy)
- Invoke Maven 3** (highlighted with a red circle)
- Invoke Gradle
- Invoke NodeJS
- Copy Artifacts

- Goals 를 clean assembly:assembly 로 변경합니다.

< Jobs Overview | Twitter Feed Build | Configure build job

Main Build Parameters Source Control Triggers Environment **Build Steps** Post Build Advanced

Configure Build Steps

Add Build Step ▾

Invoke Maven 3

Maven 3 (Bundled) ▾

Goals clean assembly:assembly

- Post Build(게시판)탭을 클릭합니다. **Archive the artifacts** 를 체크하고 Files to Archive 에 **\*\*/target/\*** 를 입력합니다. Compression Type 은 **GZIP** 으로 하고., **Publish JUnit test report** 는 체크되어 있어야 합니다. Test Report XMLs 에는 **\*\*/target/surefire-reports/\*.xml** 를 입력합니다. 각 빌드에 대한 테스트 결과에 대한 보고서를 제공합니다.

< Jobs Overview | Twitter Feed Build | Configure build job

Main Build Parameters Source Control Triggers Environment Build Steps **Post Build** Advanced

Aggregate downstream test results  
 Build other jobs  
 Archive the artifacts

\* Files To Archive **\*\*/target/\***  
 Enable auto validation for file masks

Excludes  
 Discard all but the last successful/stable artifact to save disk space

Compression Type **GZIP**

Record fingerprints of files to track usage  
 Publish Javadoc  
 Publish JUnit test result report

Test Report XMLs **\*\*/target/surefire-reports/\*.xml**  
 Retain long standard output/error

Archive Maven 3 artifacts  
 Record fingerprints of Maven artifacts  
 Git publisher  
 Notify that Maven dependencies have been updated by Maven 3 integration  
 Oracle Cloud Service Deployment

- Save(저장)를 클릭하여 구성을 완료합니다. Build now(지금 빌드)버튼을 클릭하여 빌드를 즉시 시작합니다. 잠시 후, 상태는 다음과 같이 변경됩니다.

## ◀ Jobs Overview | Twitter Feed Build

### Description

No description available

### Permalinks

[Last](#) | Successful | Failed | Completed | Unsuccessful | Stable | Unstable

### Notifications

On      Off

✉ CC Me

### Build History

New Build

N/A



Status	Build	Time	Duration	Console
	#1	Just now	N/A	

- 빌드가 시작되면 빌드가 완료되는 데 약 1 ~ 2 분이 걸립니다. 완료되면 Test Result Trend 섹션에서 성공적인 테스트 실행 횟수를 볼 수 있습니다. 다음 단계로 진행하기 전에 빌드가 완료 될 때까지 기다리십시오. 배포 구성을 완료하려면 빌드 아티팩트가 필요하므로 빌드가 시작되면 콘솔 아이콘을 클릭하여 빌드 로그 세부 사항을 모니터링 할 수 있습니다.

## ◀ Jobs Overview | Twitter Feed Build

Build Now | Configure | Disable | Delete

### Description

No description available

### Permalinks

[Last](#) | [Successful](#) | Failed | Completed | Unsuccessful | [Stable](#) | Unstable

### Notifications

On      Off

✉ CC Me

### Build History

Status	Build	Time	Duration	Console
	#1	Just now	1 min 51 s	

### Artifacts of Last Successful Build

target

(all files in zip)

### Build Trend

The chart displays the build duration in seconds for the first successful build. The Y-axis represents the duration in seconds, ranging from 0 to 120. The X-axis represents the build number, with a single data point at build number 1. The duration for this build is approximately 100 seconds.

Module /200

Oracle 퍼블릭 클라우드 워크샵

### STEP 8: 기본 구축 프로세스 생성

이제 프로젝트를 성공적으로 구축 했으므로 안정적인 빌드를 감시하고 테스트를 위해 새로운 Application Container Cloud Service 인스턴스에 배치하는 배포 구성을 만들어야합니다.

- 탐색 패널에서 **Deploy** 를 클릭하여 배포 페이지에 액세스합니다. **New Configuration** 을 클릭하십시오.



Define a new configuration plan for deploying an application in your project to an Oracle cloud service instance.

- 다음 데이터를 입력합니다.

```
Configuration Name = TwitterFeedMicroserviceDeploy
Application Name = JavaTwitterMicroservice
```

\* Configuration Name TwitterFeedMicroserviceDeploy

\* Application Name JavaTwitterMicroservice

- 배포 대상 오른쪽에 **New** 를 클릭하고 **Application Container Cloud** 를 선택합니다.

* Configuration Name	Twitter Feed Microservice Deploy
* Application Name	JavaTwitterMicroservice
* Deployment Target	Select a Java Service
Type	<input checked="" type="radio"/> On Demand <input type="radio"/> Automatic
* Job	Twitter Feed Build
* Build	1 (Success)
* Artifact	Select an Artifact

New ▾

Java Cloud Service - SaaS Extension...  
 Java Cloud Service...  
**Application Container Cloud...**

- 다음 데이터를 입력하고 Test Connection(테스트 연결)을 클릭합니다. 성공적이면 연결 사용을 클릭합니다.

```
Data Center = EMEA Commercial 2 - em2
** or your appropriate Data Center **
Identity Domain = <Your Identity Domain>
Username = bala.gupta (or your appropriate username
if running as single user)
Password = <Supplied Password>
```

**Deploy to Application Container Cloud**

* Data Center	EMEA Commercial 2 - em2
* Identity Domain	gse00002057
* Username	bala.gupta
* Password	*****
<span style="color: green;">✓ Successful</span> <span style="border: 1px dashed gray; padding: 2px;">Use Connection</span>	

- ACCS Properties 를 Runtime **Java** 및 Subscription **Hourly** 로 설정하십시오.  
**Automatic** 을 클릭하십시오. **Artifact**에 대해 **target/twitter-microservice-example-dist.zip** 을 선택하십시오.

ACCS Properties	Runtime <input checked="" type="radio"/> Java <input type="radio"/> Node	Subscription <input checked="" type="radio"/> Hourly <input type="radio"/> Monthly
Type	<input type="radio"/> On Demand <input checked="" type="radio"/> Automatic <input checked="" type="checkbox"/> Deploy stable builds only	
* Job	Twitter Feed Build	
* Artifact	target/twitter-microservice-example-dist.zip	

- 저장을 클릭하십시오.

New Deployment Configuration

Save Cancel

\* Configuration Name: TwitterFeedMicroserviceDeploy

\* Application Name: JavaTwitterMicroservice

\* Deployment Target: em2 / gse00002055 / bala.gupta

ACCS Properties

Runtime:  Java  Node

Subscription:  Hourly  Monthly

Type:  On Demand  Automatic  Deploy stable builds only

\* Job: Twitter Feed Build

\* Artifact: target/twitter-microservice-example-dist.zip

- 드롭 다운 메뉴를 클릭하고 **Start** 을 선택합니다.

Deployments

JavaTwitterMicroservice

Deploy to ACCS: em2 / gse00002057 / bala.gupta

Configuration: TwitterFeedMicroserviceDeploy

Job / Build: Twitter Feed Build / Latest Successful Build

Artifact: target/twitter-microservice-example-dist.zip

TwitterFeedMicroserviceDeploy: History

- ▶ Start
- ▶ Redeploy
- >Edit Configuration
- Delete Configuration

- 메시지가 **Last deployment succeeded** 으로 될 때까지 기다립니다.

## Deployments

JavaTwitterMicroservice

Deploy to ACCS: em2 / gse00002055 / bala.gupta

Configuration: TwitterFeedMicroserviceDeploy

Job / Build: Twitter Feed Build / Latest Successful Build

Artifact: target/twitter-microservice-example-dist.zip

 Starting application - Just now.

## Deployments

JavaTwitterMicroservice

Deploy to ACCS: em2 / gse00002055 / bala.gupta

Configuration: TwitterFeedMicroserviceDeploy

Job / Build: Twitter Feed Build / Latest Successful Build

Artifact: target/twitter-microservice-example-dist.zip

 Last deployment succeeded - Just now.

## Twitter 피드/트위터 구축을 확인합니다.

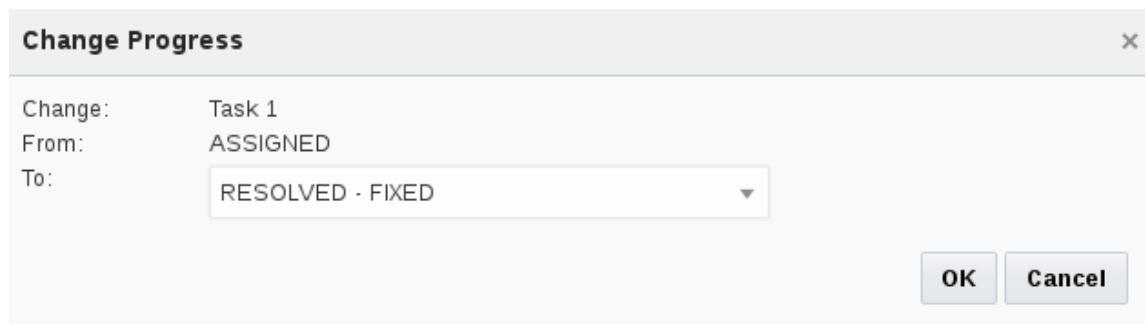
### STEP 9: Verified 상태로 전환

이제 Application Container Cloud Service에 빌드 아티팩트를 성공적으로 배포 했으므로 Agile 보드를 업데이트하여 해당 상태를 반영합니다. 다음 작업의 복잡성(검증)은 매우 간단하지만 새로운 기능을 수동으로 확인하기 전에 작업을 "Verify Code" 열로 이동합니다.

- 탐색 패널에서 **Agile** 을 클릭하고 **Active Sprints** 를 클릭하십시오. **Task 1** 을 **In Progress** 에서 **Verify Code** 열로 끌어다 놓습니다.



- 진행 상황 변경 팝업 화면에서 **OK(확인)**를 클릭합니다.



- 이제 완료되기 전에 확인을 위해 코드를 확인할 수 있습니다.



## STEP 10: Oracle 애플리케이션 컨테이너 클라우드 서비스에 로그인

- 아직 사용할 수 있는 경우 개발자 서비스 클라우드 대시 보드 탭으로 돌아가서 대시 보드 아이콘을 선택하여 Oracle Public Cloud Dashboard로 돌아갑니다. 참고 : 세션이 만료 된 경우 다시 로그인해야 할 수도 있습니다.

The screenshot shows the Oracle Developer Cloud Service dashboard for the service 'developer99019'. It includes a service status chart for August 2016, showing 100% uptime. A legend indicates 'Before Activation' (light blue), 'Service Up' (green), 'Planned Outage' (yellow), and 'Service Incident' (red). A red circle highlights the 'Dashboard' button in the top right corner of the header.

- OraclePublic(공용)클라우드 대시 보드가 표시되면 Application(애플리케이션)컨테이너의 오른쪽에 있는 메뉴를 클릭한 다음 Open Service 콘솔을 선택합니다.

The screenshot shows the Oracle Public Cloud Dashboard with several service cards: developer99019 (Developer), Database Instances, Application Container Instances, and Database Backup. The Database Backup card has a dropdown menu with 'View Details' and 'Open Service Console' options. A red circle highlights the 'Open Service Console' button.

- Application Container Cloud Service (ACCS) 서비스 콘솔에서 새로 생성 된 **JavaTwitterMicroservice** 를 비롯하여 배포 된 모든 응용 프로그램을 볼 수 있습니다. URL 을 클릭하면 새 브라우저 탭이로드됩니다.

The screenshot shows the Oracle Application Container Cloud Service (ACCS) Applications page. It lists the 'JavaTwitterMicroservice' application, which was last deployed on Aug 15, 2016 at 8:34:05 PM UTC. The URL for the application is highlighted with a red circle: <https://JavaTwitterMicroservice-gse00002057.apaas.em2.oraclecloud.com>.

- 브라우저의 URL 끝에 **/statictweets** 를 추가하고 return 을 누릅니다.

```
https://javatwittermicroservice-
gse00002055.apaas.em2.oraclecloud.com/statictweets
```

- URL 은 정적 Twitter 피드가 포함 된 JSON 배열을 반환해야합니다. 참고 : JSON 의 형식이 지정된 보기를 보려면 새 탭을 열고 Google 에서 'JSONViewer chrome plugin'을 검색하십시오. - Chrome 플러그인을 설치하고 URL 을 다시 제출하면 보다 읽기 쉬운 형식으로 JSON 을 볼 수 있습니다.



```
{"tweets" : [{"delete": {"status": {"id": 765578381903601664, "id_str": "765578381903601664", "user_id": 2186468947, "user_id_str": "2186468947", "timestamp_ms": 1519997185069305856, "id_str": "519997185069305856", "user_id": 16693090, "user_id_str": "16693090"}, "timestamp_ms": 1519997185069305856, "id": 765625802700513281, "id_str": "765625802700513281", "user_id": 72625334, "user_id_str": "72625334"}, "text": "The VICTORIOUS SESSIONS at Little Johnny Russel's Bookshop in Portsmouth, UK", "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "is_quote_status": false, "retweet_count": 0, "favorite_count": 0, "coordinates": null, "place": null, "contributors": null}, {"id": 23947154, "id_str": "23947154", "name": "Nick Courtney", "screen_name": "NickCourtney", "location": "Portsmouth, UK", "url": "http://www.book.events and Sweet Memories Vinyl Records", "protected": false, "verified": false, "followers_count": 1399, "friends_count": 2009, "utc_offset": 3600, "time_zone": "London", "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_background_image_url_https": "https://abs.twimg.com/images/themes/theme14/bg.gif", "profile_text_color": "#333333", "profile_use_background_image": true, "profile_image_url": "http://pbs.twimg.com/profile_images/697776971057594368/kMfUzOxw_normal.jpg", "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/1519997185069305856", "url": "https://t.co/9DGpEX0js", "expanded_url": "http://fb.me/7y5dSkFTD", "display_url": "fb.me/7y5dSkFTD", "indices": [1], "favorited": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "timestamp_ms": 1519997185069305856, "id": 765626469615702016, "id_str": "765626469615702016", "text": "#DolceAmoreHotSeat Wave to the crowd - #Push", "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "is_quote_status": false, "retweet_count": 0, "favorite_count": 0, "coordinates": null, "place": null, "contributors": null}, {"id": 724990427187671041, "id_str": "724990427187671041", "name": "LizQuenlurve", "screen_name": "LQuenlurve", "location": "Philippines", "url": null, "description": null, "protected": false, "verified": false, "followers_count": 16, "friends_count": 2016, "utc_offset": null, "time_zone": null, "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_background_tile": false, "profile_link_color": "#2B7BB9", "profile_sidebar_border_color": "#CODEED", "profile_sidebar_fill_color": "#F0F0F0", "profile_text_color": "#333333", "profile_use_background_image": true, "profile_image_url": "http://pbs.twimg.com/profile_images/724990726602268672/fAOZzR3o_normal.jpg", "profile_image_url_https": "https://pbs.twimg.com/profile_images/724990726602268672/fAOZzR3o_normal.jpg", "url": null, "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "is_quote_status": false, "retweet_count": 0, "favorite_count": 0, "coordinates": null, "place": null, "contributors": null}], "meta": {"total_count": 2, "next_max_id": 0, "next_max_id_str": "0", "prev_max_id": 0, "prev_max_id_str": "0"}}
```

### STEP 11: 작업 완료

우리는 이제 statictweets microservice 가 배치되었고 기능이 제대로 작동하고 있음을 확인했다. 이 부분을 완료하기 위해, 우리는 우리의 스프린트에서 완성된 이슈를 기록할 것입니다.

- 개발자 클라우드 서비스 창으로 돌아가려면 애자일(Agile)을 클릭하고 Active Sprints 를 클릭합니다.



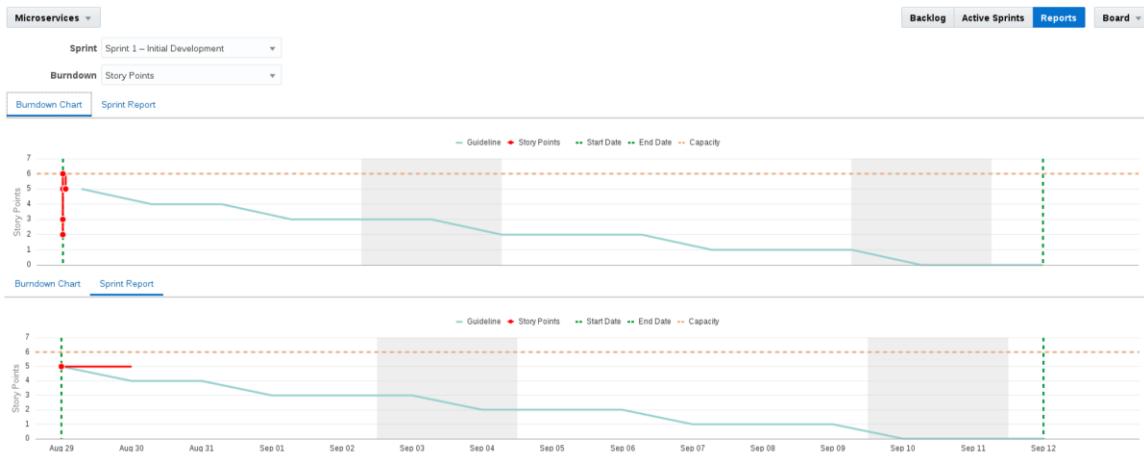
- Task1 에서 Task1 을 드래그 하여 완료합니다.
- Progress(진행)팝업 창에서 OK(확인)를 클릭합니다.



- 이제 스프린트는 다음과 같아야 한다.



- Report(리포트)버튼을 클릭하고 **Burndown Chart** 및 **Sprint Report**에서 진행 상황을 볼 수도 있습니다.



## 정적 트위터 피드 서비스에 필터 추가

이제 초기 정적 트위터 마이크로 서비스의 가져 오기, 빌드, 배포 및 확인을 완료 했으므로 내용을 기반으로 들어오는 트윗을 동적으로 필터링 할 수 있는 새로운 마이크로 서비스를 추가하여 프로젝트를 확장해야합니다. Eclipse IDE를 사용하여 관리되는 GIT 저장소를 로컬 워크 스테이션에 복제하고, 로컬 복사본을 테스트하고, 로컬 복사본에 필터링 기능을 추가합니다. 우리는 이클립스에서 새로운 기능을 테스트하고 이를위한 새로운 코드 브랜치를 만들고 브랜치를 커밋 할 것이다. 그런 다음 병합 요청을 만들고 해당 요청을 승인하기 위해 Project Manager로 전환합니다. 또한 Eclipse에서 직접 agile task의 상태를 관리하는 방법에 대해서도 알아볼 것입니다.

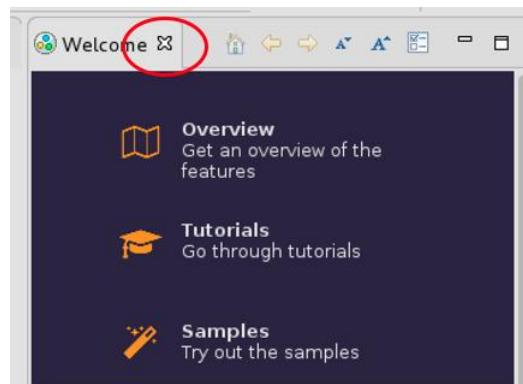
### 이클립스 IDE 프로젝트 클론 생성

#### STEP 12: 이클립스 IDE 로드

- 마우스 오른쪽 버튼을 클릭하고 바탕 화면 아이콘을 선택합니다.

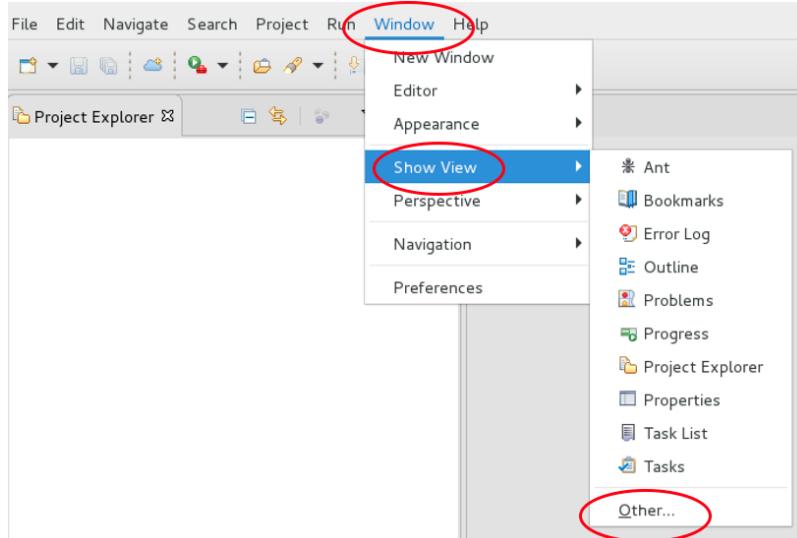


- 일단 Eclipse 가 로드되면, Welcome 창을 닫으세요.

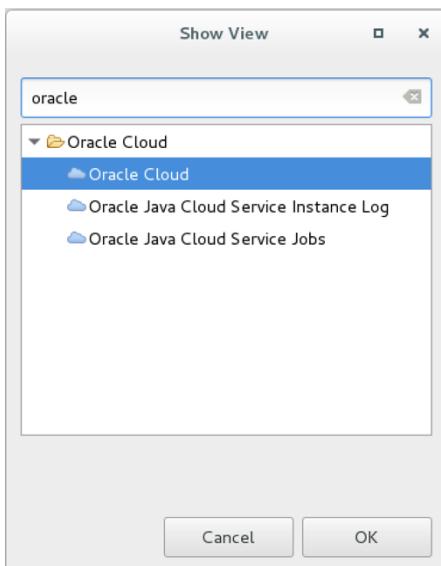


**STEP 13: Oracle 개발자 클라우드 서비스에 대한 연결 생성**

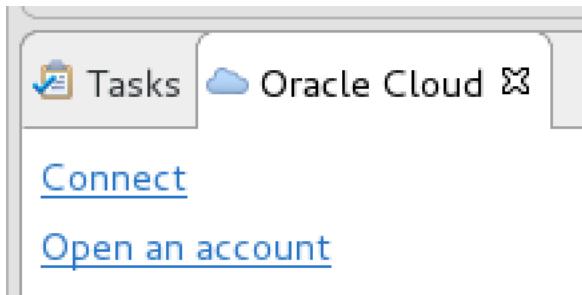
- 이제 개발자 클라우드 서비스에 대한 connection 을 생성하겠습니다. 이렇게 하려면 먼저 **Window -> Show View ->Other** 메뉴 옵션을 클릭합니다.



- 검색 필드에 oracle 을 입력하세요. **Oracle Cloud** 를 선택하고 **OK(확인)**를 클릭합니다.

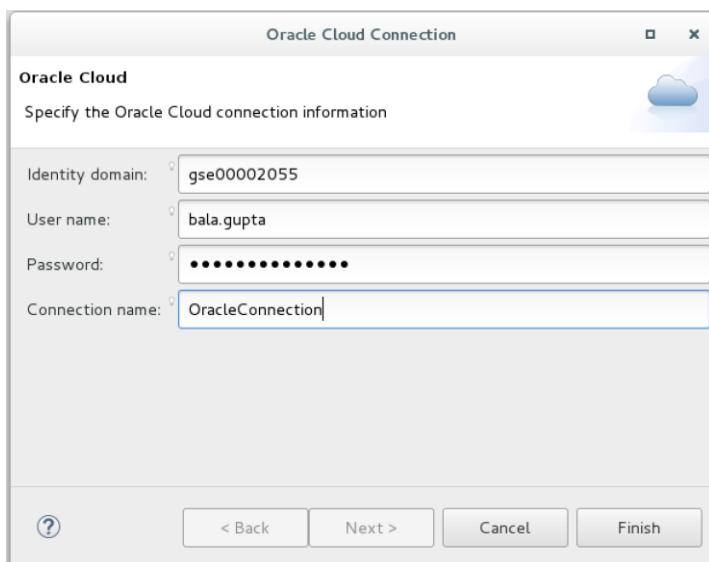


- Oracle 클라우드 탭에서 **Connect(연결)**를 클릭합니다.



- 다음 정보를 입력하십시오.

```
Identity Domain: <your identity domain>
User name:      bala.gupta (or your appropriate username,
                  if running as single user)
Password:       <your Identity domain password>
Connection Name: OracleConnection
```



- 메시지가 나타나면 Eclipse Secure Storage 의 마스터 비밀번호를 입력하고 확인하십시오.  
이 예에서는 oracle 암호를 사용합니다. 다음 확인을 누릅니다.

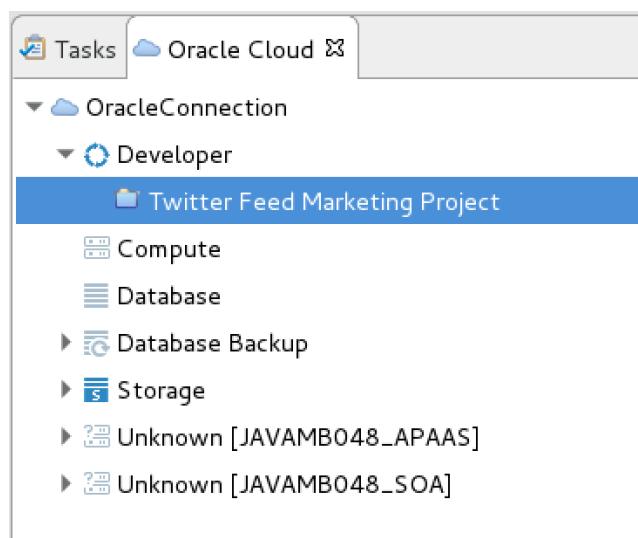


- 비밀 번호 힌트를 입력하라는 메시지가 표시되면 **No(아니오)**를 클릭합니다.

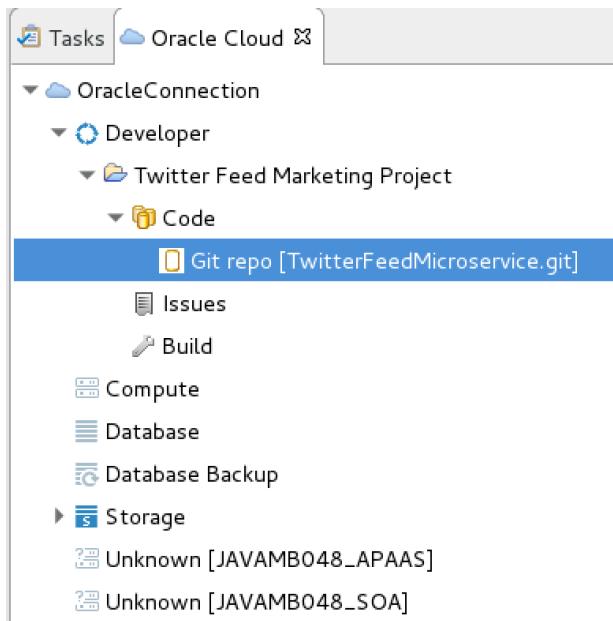


#### STEP 14: 저장소의 로컬 클론 생성

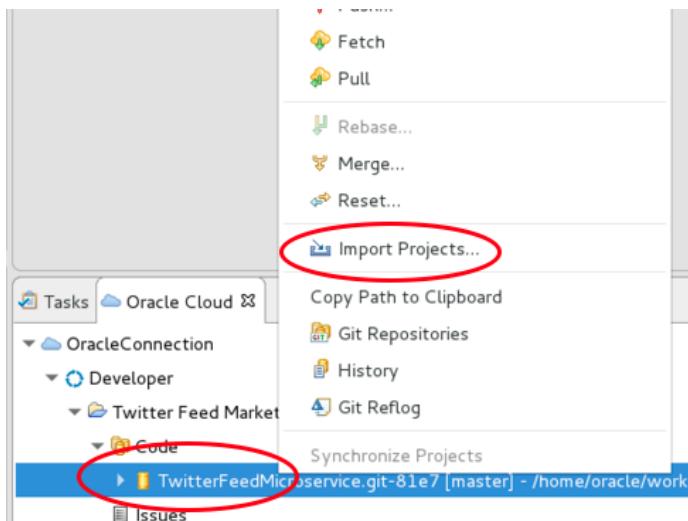
- **Developer** 를 클릭해서 확장하고 **Twitter Feed Marketing Project** 를 더블 클릭하여 프로젝트를 활성화하세요.



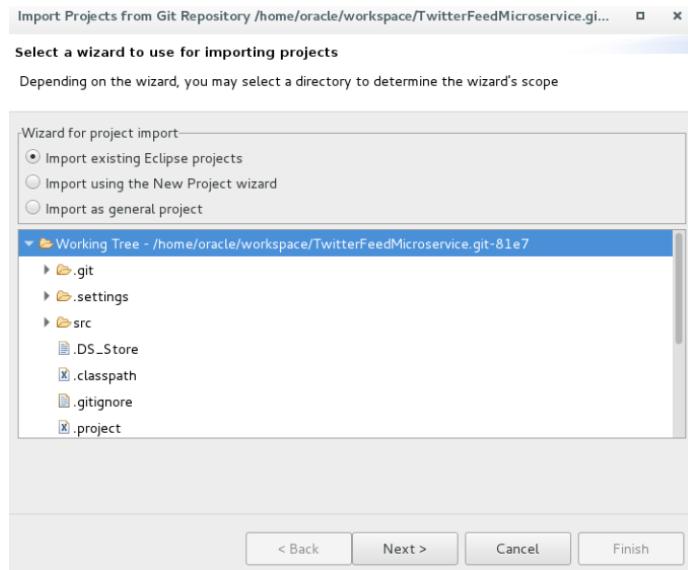
- Code** 를 확장하고 **Git Repo [TwitterFeedMicroservice.git]**를 더블 클릭하여 Repository 가 로컬로 복제되도록 한다.



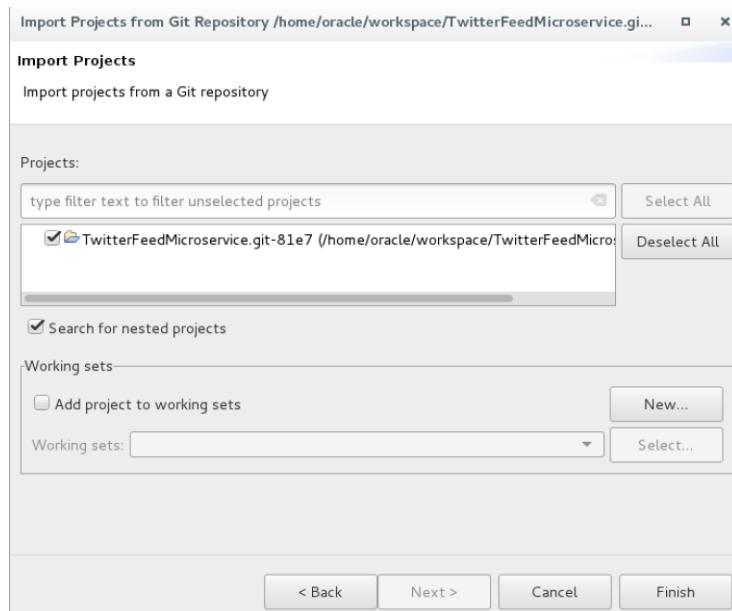
- TwitterFeedMicroservice** 를 마우스 오른쪽 단추로 클릭하고 **Import Project** 를 클릭합니다.



- 마법사를 기본 값으로 설정하고 **Next(다음)**을 클릭합니다.

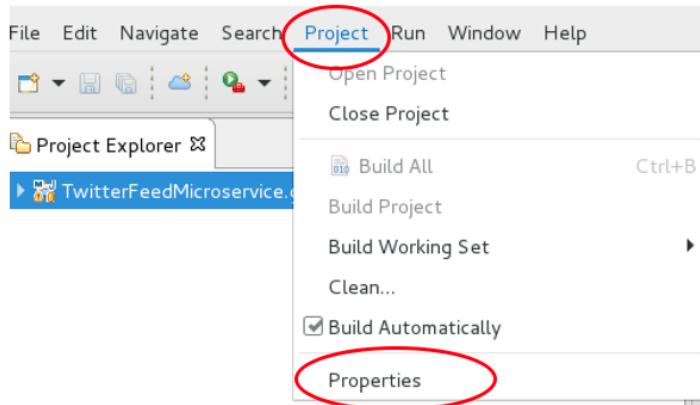


- 가져오기 기본 값을 수락하고 **Finish(마침)**을 클릭합니다.

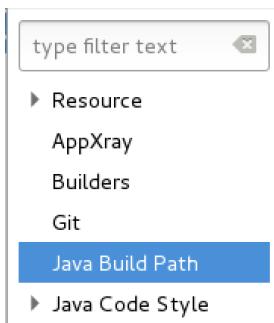


### STEP 15: 올바른 Java JDE 선택

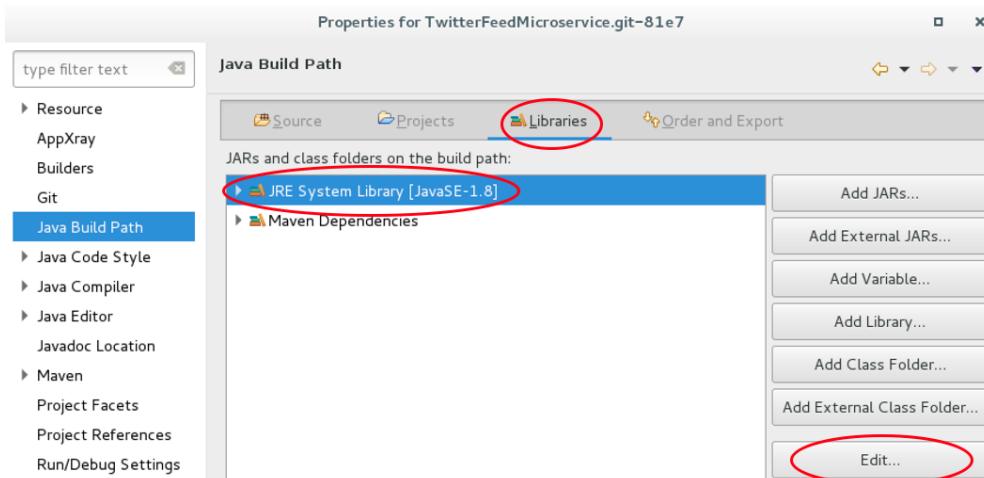
- TwitterFeedMicroservice Project** 를 클릭한 다음 상단 메뉴에서 **Project>Properties(프로젝트>속성)**를 선택합니다.



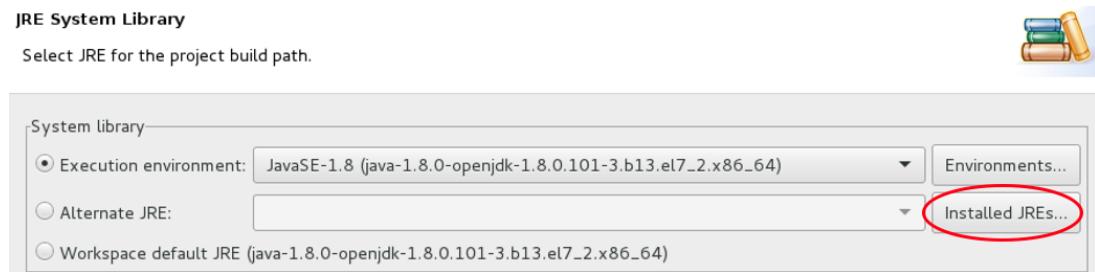
- Java Build Path 옵션을 선택합니다.



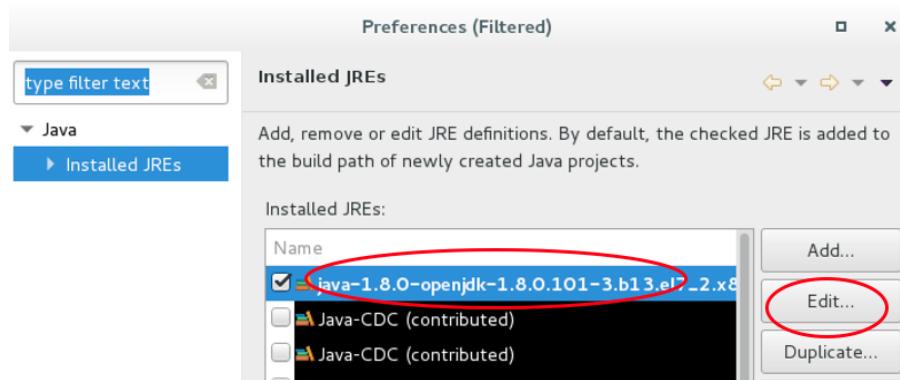
- 라이브러리 탭을 클릭한 다음 JRE System Library 를 선택합니다. 다음으로 Edit(편집)버튼을 클릭합니다.



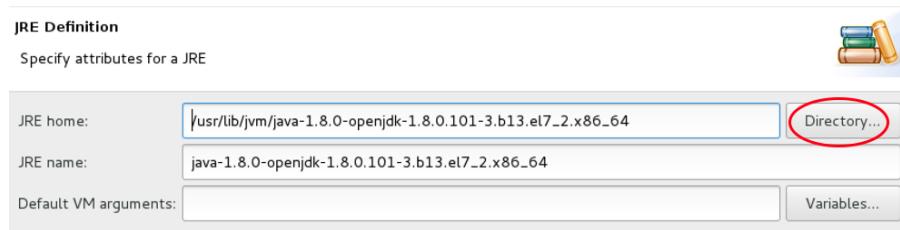
- Install JREs 버튼을 클릭합니다.



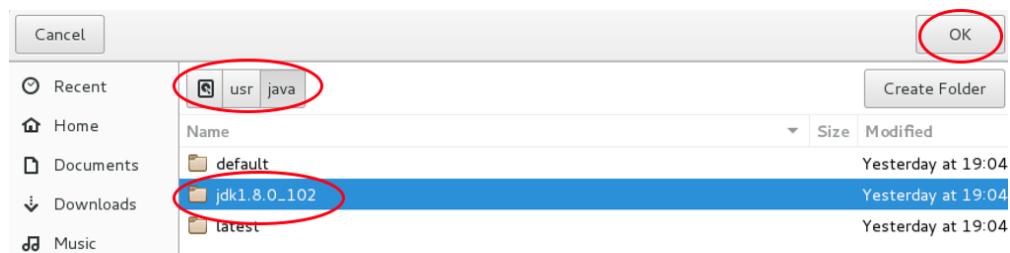
- 이 경우 Standard VM 을 선택합니다. 이 예에서는 **java-1.8.0-openjdk** 로 되어 있습니다. 그런후에 Edit(편집)을 클릭합니다.



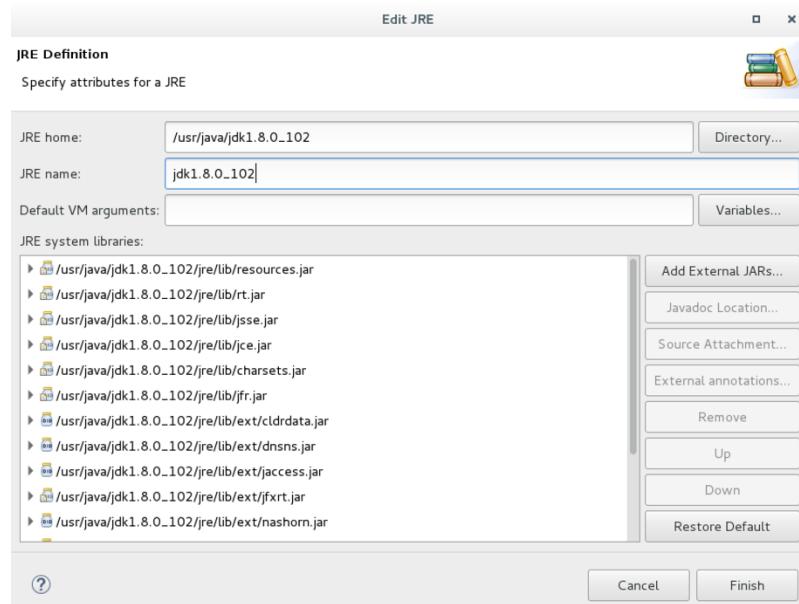
- JRE 품목:Directory 버튼 클릭



- usr/java** 로 이동해서, **jdk1.8.0\_102** 를 선택, OK 를 클릭합니다.



- JRE Name 을 jdk1.8.0\_102 로 변경하고 Finish(마침)을 클릭합니다.



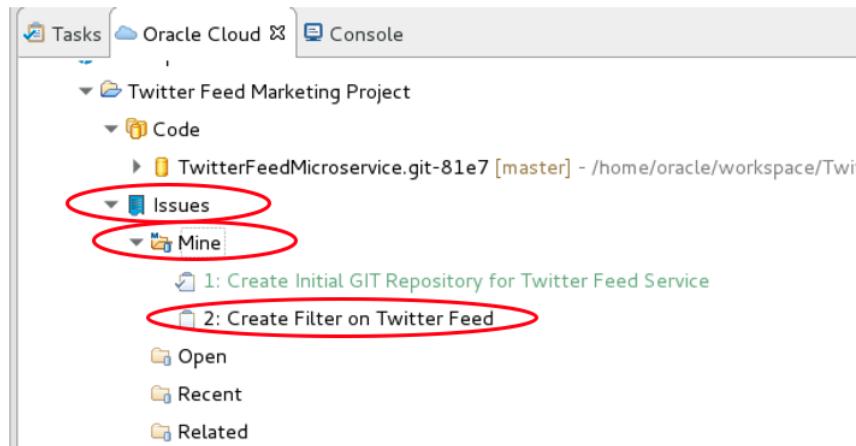
- OK(확인)를 클릭하여 다음 대화 상자에서 메시지가 표시되면 라이브러리의 변경 사항을 완료하고 OK(확인)를 클릭합니다.

## Local Cloned Service 테스트

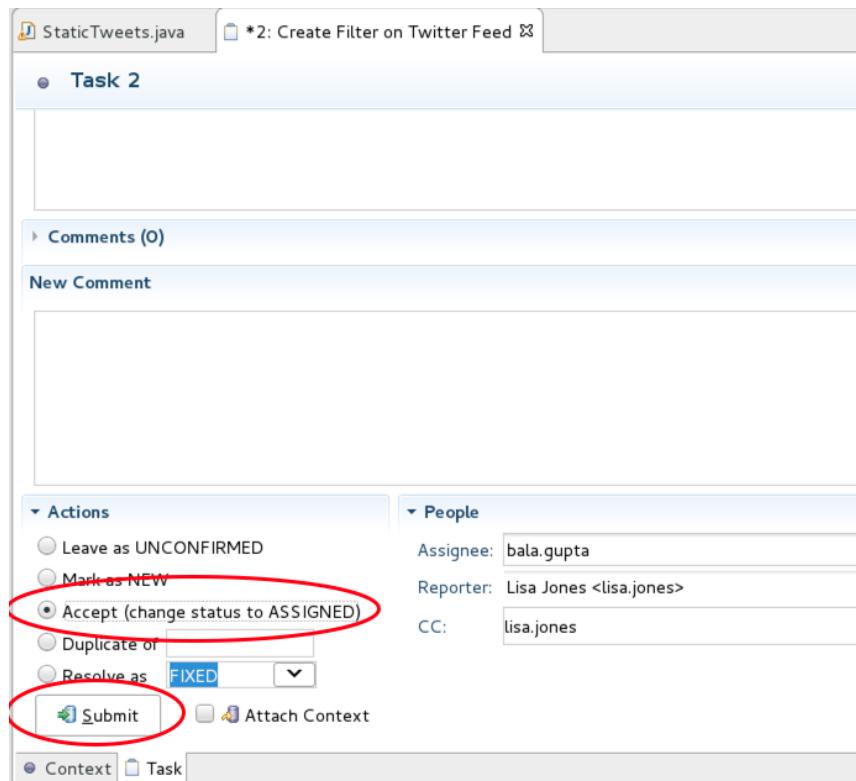
### STEP 16: Feature 2 상태를 진행 중으로 설정합니다.

이전 단계에서는 웹 인터페이스를 사용하여 개발자 클라우드 서비스에 대한 작업 할당을 Bala Gupta 로 업데이트했습니다. 이 단계에서 우리는 개발자 클라우드 서비스를 개발자 클라우드 서비스에 사용하여 Bala 의 task 를 업데이트할 것입니다.

- Oracle Cloud 연결 탭에서 두번 클릭하여 Issue 를 확장한 다음, **Mine** 을 두번 클릭하여 목록을 확장합니다. Issue 목록을 보면, **Create Filter on Twitter Feed** 를 더블 클릭합니다.



- **Create Filter on Twitter Feed** 의 하단으로 스크롤 합니다. **Actions(작업)섹션**에서 **Accept (change status to ASSIGNED)** 를 클릭하여 **Submit(제출)**을 클릭합니다.

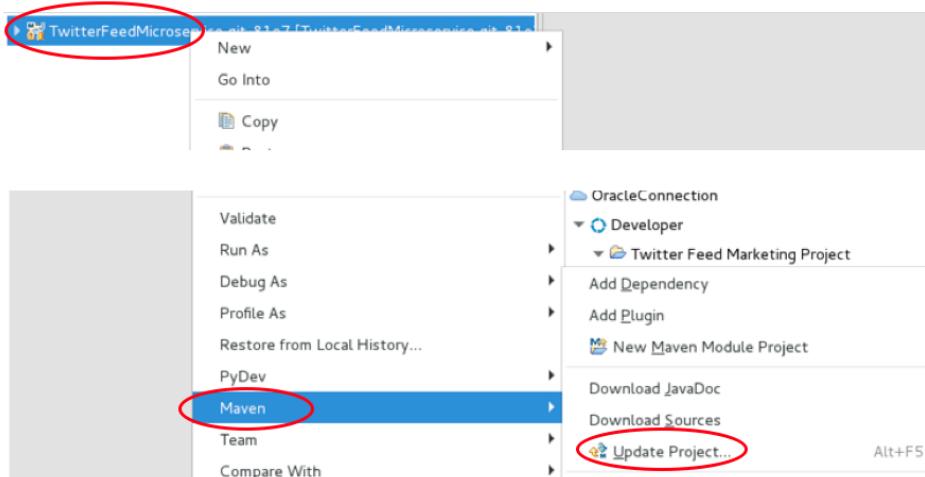


- 선택적으로, 개발자 클라우드 서비스 웹 인터페이스로 돌아 가면 Eclipse 인터페이스에서 Feature 2 가 Agile > Active Sprints 의 "In Progress" 열로 이동하게됩니다.

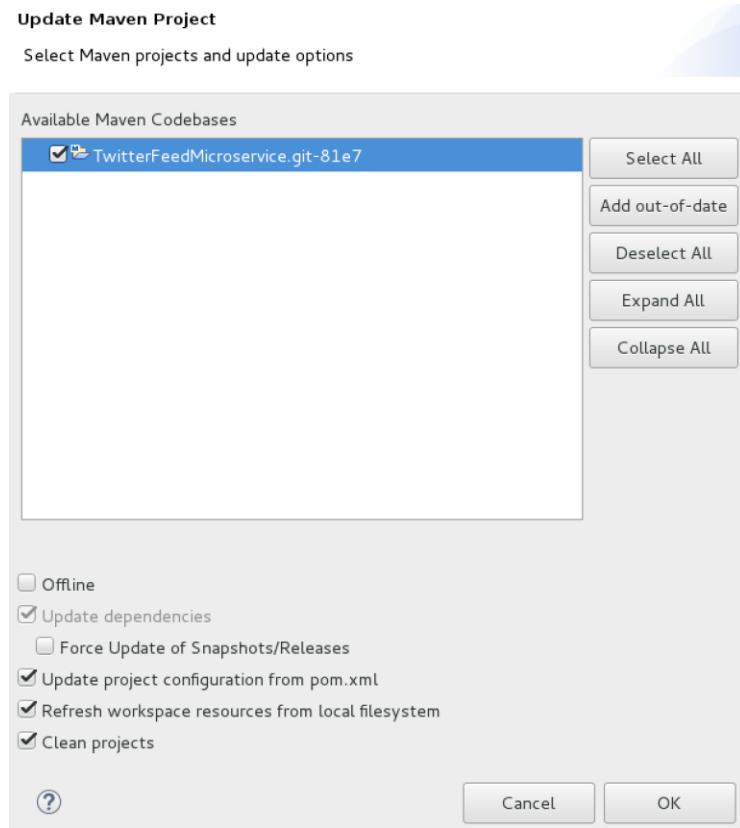
The screenshot shows the Microservices web interface. At the top, it says "Microservices". Below that, it shows a sprint: "Sprint 1 – Initial Development -- 11/2/2016 1:13 PM • 11/16/2016 12:13 PM". There is a "Complete Sprint" button. Below the sprint, there are two dropdown menus: "Sort Issues by: Priority" and "Sort Swimlanes by: User Name". Under "To Do (2)", there are three items. Under "In Progress (1)", there is one item: "Feature 2: Create Filter on Twitter Feed" assigned to "bala.gupta".

### STEP 17: TwitterFeedMicroservice 빌드 및 테스트

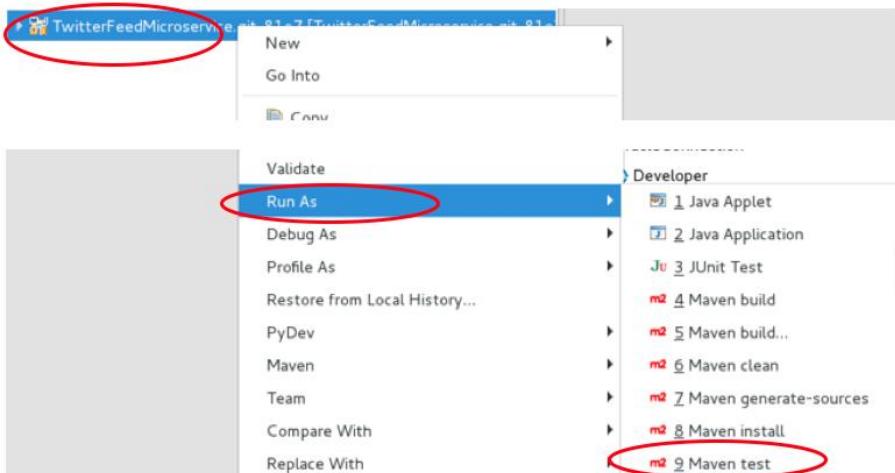
- 마우스 오른쪽 버튼을 클릭합니다. **Maven>Update Project** 선택



- 기본 설정을 유지하고 OK(확인)를 클릭합니다.



- 프로젝트 코드의 로컬 사본을 테스트하려면 **TwitterFeedMicroservice Project** 를 마우스 오른쪽 버튼으로 클릭한 다음 **Run as > Maven Test** 선택합니다.



- Console tab** 을 더블 클릭하면 창이 확장됩니다. 탭을 다시 더블 클릭하여 창을 최소화 할 수 있습니다. TwitterFeedMicroservices 테스트가 성공적으로 실행되면 콘솔 창에 "### Tweets in Static Tweets"라는 메시지가 표시됩니다. 또한 실패가 없음을 확인해야합니다.



```

<terminated> /usr/java/jdk1.8.0_102/bin/java (Aug 17, 2016, 4:15:37 PM)
[INFO] Downloaded: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/sure
----- T E S T S -----
Running com.example.MyServiceTest
Aug 17, 2016 4:15:52 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost,localhost:8080]
Aug 17, 2016 4:15:52 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer] Started.
{"delete":{"status":{"id":765578381903601664,"id_str":"765578381903601664","user_id":2
{"delete":{"status":{"id":765625802700513281,"id_str":"765625802700513281","user_id":7
{"delete":{"status":{"id":519997185069305856,"id_str":"519997185069305856","user_id":1
{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": 765626469619859456, "id_str": "76562
{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": 765626469615702016, "id_str": "76562
{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": 765626469582110721, "id_str": "76562
{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": 765626469607354368, "id_str": "76562
{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": 765626469615865856, "id_str": "76562
{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": 765626469586497536, "id_str": "76562
{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": 765626469590634497, "id_str": "76562
101 Tweets in StaticTweets
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost,localhost:8080]
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost,localhost:8080]
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer-1] Started.
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost,localhost:8080]
Aug 17, 2016 4:15:54 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost,localhost:8080]
Aug 17, 2016 4:15:54 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer-2] Started.
Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.593 sec
Aug 17, 2016 4:15:54 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost,localhost:8080]

Results :

Tests run: 3, Failures: 0, Errors: 0, Skipped: 0

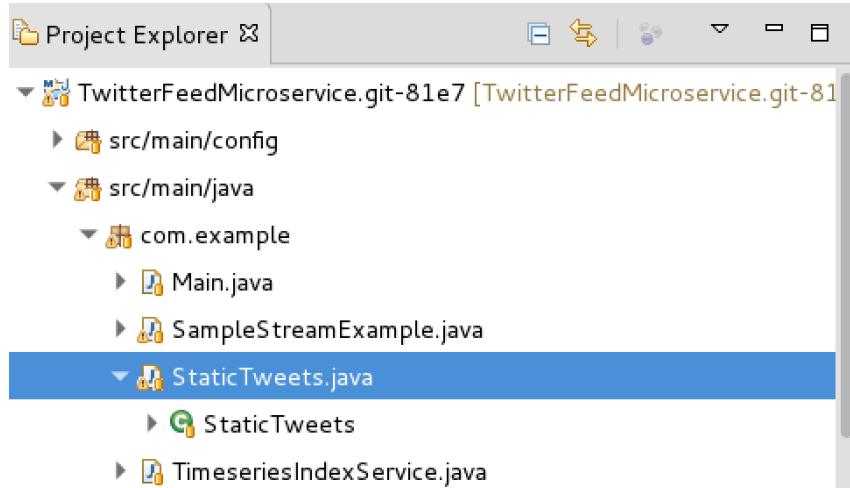
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 14.560 s
[INFO] Finished at: 2016-08-17T16:15:54-04:00
[INFO] Final Memory: 11M/46M
[INFO] -----
```

## 필터를 서비스에 추가

우리가 로컬에서 복제한 코드는 정적 트위터 피드를 필터링 하는 데 필요한 모든 소스를 포함하고 있습니다. 이 섹션의 섹션에서 코드의 주석을 해제하고 필터를 테스트합니다.

### STEP 18: 필터 추가

- Project Explorer에서 **TwitterFeedMicroservice > src/main/java > com.example** 를 확장하고 **StaticTweets.java** 를 더블 클릭하여 소스 코드를 엽니다.



- StaticTweets.java에서 두줄로 된 소스 파일을 스크롤하여 "--- Remove this comment"로 시작하는 두줄의 코드를 찾을 때까지 스크롤합니다. 이 두라인을 모두 삭제하여 정적 트위터 파일의 필터링을 일으킬 코드를 활성화합니다..

```
/*
 * Method handling HTTP GET requests. The returned object will be sent
 * to the client as "text/json" media type.
 *
 * @return String that will be returned as an application/json response.
 */
/* --- Remove this comment to allow for Filtered Searching */
@Path("/{search}")
@GET
@Produces(MediaType.APPLICATION_JSON)
public Response getItWithCount(@PathParam("search") final String search) {
    final ChunkedOutput<String> output = new ChunkedOutput<String>(String.class);
    runTask(output, search);
    return Response.ok()
        .entity(output)
        .header("Access-Control-Allow-Origin", "*")
        .header("Access-Control-Allow-Methods", "GET, POST, DELETE, PUT")
        .build();
}

/* --- Remove this comment to allow for Filtered Searching */
```

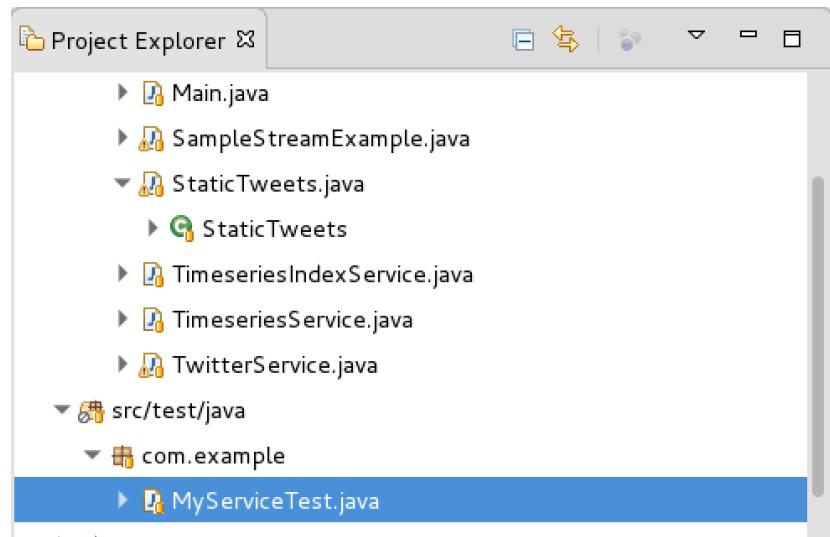
- 이제 이 코드는 다음과 같아야 합니다.

```

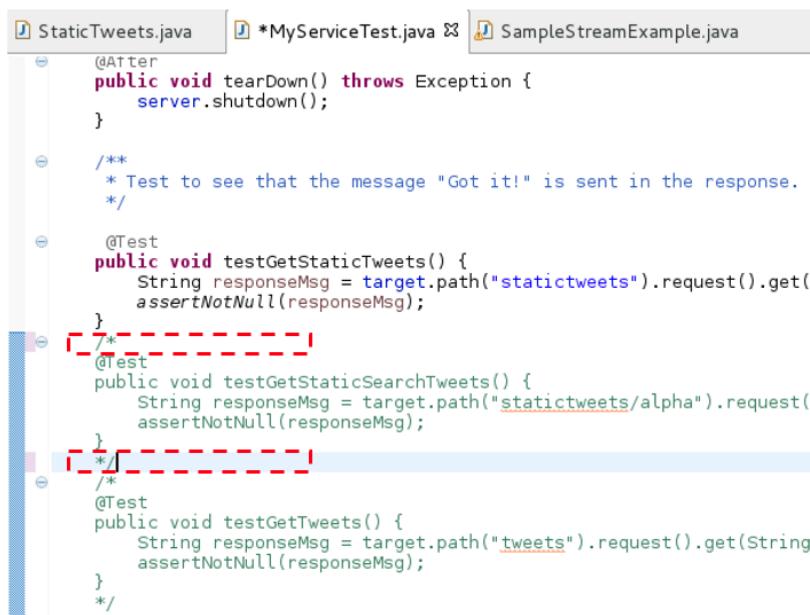

    /**
     * Method handling HTTP GET requests. The returned object will be sent
     * to the client as "text/json" media type.
     *
     * @return String that will be returned as an application/json response.
     */
    @Path("{search}")
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public Response getItWithCount(@PathParam("search") final String search) {
        final ChunkedOutput<String> output = new ChunkedOutput<String>(String.class);
        runTask(output, search);
        return Response.ok()
            .entity(output)
            .header("Access-Control-Allow-Origin", "*")
            .header("Access-Control-Allow-Methods", "GET, POST, DELETE, PUT")
            .build();
    }
}


```

- 다음으로 테스트 코드의 필터를 활성화하겠습니다. [src/test/java > com.example](#) 폴더를 확장하고 **MyServiceTest.java** 를 더블 클릭하여 소스 파일을 엽니다.



- MyServiceTest.java 소스파일에서 메서드 testGetStaticSearchTweets()를 찾은 다음, 코드의 섹션이 실행될 수 있도록 코멘트를 제거합니다.



```

    public void tearDown() throws Exception {
        server.shutdown();
    }

    /**
     * Test to see that the message "Got it!" is sent in the response.
     */

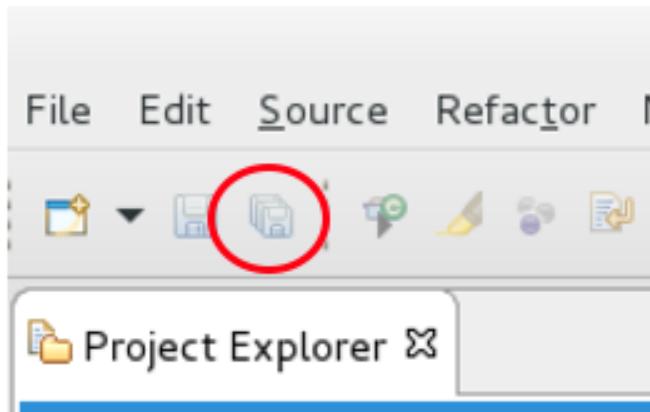
    @Test
    public void testGetStaticTweets() {
        String responseMsg = target.path("statictweets").request().get(
            assertNotNull(responseMsg);
    }

    /*-----*/
    @Test
    public void testGetStaticSearchTweets() {
        String responseMsg = target.path("statictweets/alpha").request(
            assertNotNull(responseMsg);
    }

    /*-----*/
    @Test
    public void testGetTweets() {
        String responseMsg = target.path("tweets").request().get(String
            assertNotNull(responseMsg);
    }
}

```

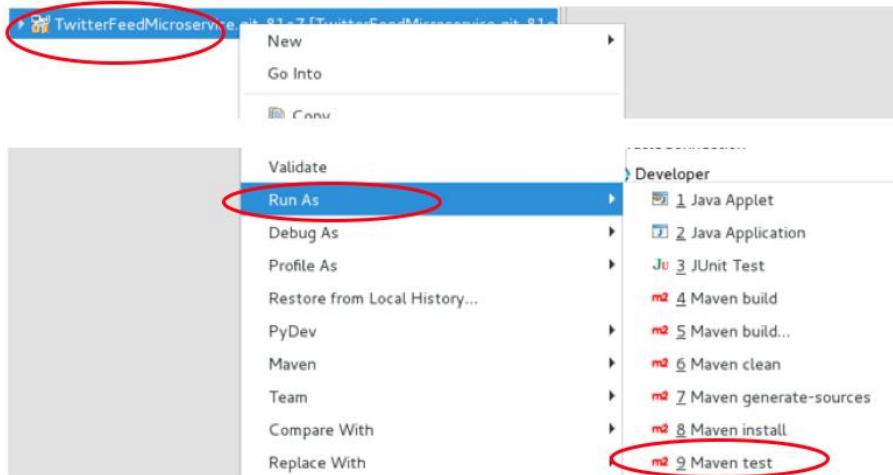
- SaveAll(모두 저장)아이콘을 클릭합니다.



## Local Filtered Service 테스트

### STEP 19: Run 테스트

- TwitterFeedMicroservice** 를 마우스 오른쪽 버튼으로 클릭하고 **Run As > Maven Test** 를 선택하여 실행합니다.



- 테스트가 완료되면 필터링 되지 않은 테스트와 필터링 테스트 양쪽 모두에 대해 반환되는 정적 트위터 피드가 표시됩니다. 실패가 표시되면 안됩니다.

```

Tasks Oracle Cloud Console

<terminated> /usr/java/jdk1.8.0_102/bin/java (Aug 17, 2016, 5:31:07 PM)
{"delete": {"status": {"id": "765578381903601664", "id_str": "218"}, "user_id": "2184668947", "user_id_str": "2184668947"}, {"delete": {"status": {"id": "765625802700513281", "id_str": "765625802700513281"}, "user_id": "72625", "user_id_str": "72625"}, {"delete": {"status": {"id": "519997185069305856", "id_str": "519997185069305856"}, "user_id": "16693090", "user_id_str": "16693090"}, {"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469619859456", "id_str": "765626469619859456", "text": "The V: ", "user_id": "16693090", "user_id_str": "16693090"}, {"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469615702016", "id_str": "765626469615702016", "text": "#Dolci ", "user_id": "16693090", "user_id_str": "16693090"}, {"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469582110721", "id_str": "765626469582110721", "text": "@Cody ", "user_id": "16693090", "user_id_str": "16693090"}, {"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469607354368", "id_str": "765626469607354368", "text": "RT @esl ", "user_id": "16693090", "user_id_str": "16693090"}, {"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469615865856", "id_str": "765626469615865856", "text": "@Kidr ", "user_id": "16693090", "user_id_str": "16693090"}, {"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469586497536", "id_str": "765626469586497536", "text": "RT @T ", "user_id": "16693090", "user_id_str": "16693090"}, {"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469590634497", "id_str": "765626469590634497", "text": "@Tupt ", "user_id": "16693090", "user_id_str": "16693090"}}, {"tweets": [{"text": "101 Tweets in StaticTweets", "user_id": "16693090", "user_id_str": "16693090"}]}, {"logs": [{"text": "Aug 17, 2016 5:31:16 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow", "level": "INFO"}, {"text": "INFO: Stopped listener bound to [localhost.localdomain:8080]", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:16 PM org.glassfish.grizzly.http.server.NetworkListener start", "level": "INFO"}, {"text": "INFO: Started listener bound to [localhost.localdomain:8080]", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:16 PM org.glassfish.grizzly.http.server.HttpServer start", "level": "INFO"}, {"text": "INFO: [HttpServer-1] Started.", "level": "INFO"}, {"text": "{"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469619859456", "id_str": "765626469619859456", "text": "Alpha ", "user_id": "16693090", "user_id_str": "16693090"}, {"text": "{"created_at": "Tue Aug 16 19:09:10 +0000 2016", "id": "765626473797545984", "id_str": "765626473797545984", "text": "I've l ", "user_id": "16693090", "user_id_str": "16693090"}}, {"text": "2 Tweets in StaticTweets", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:16 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow", "level": "INFO"}, {"text": "INFO: Stopped listener bound to [localhost.localdomain:8080]", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener start", "level": "INFO"}, {"text": "INFO: Started listener bound to [localhost.localdomain:8080]", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.HttpServer start", "level": "INFO"}, {"text": "INFO: [HttpServer-2] Started.", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow", "level": "INFO"}, {"text": "INFO: Stopped listener bound to [localhost.localdomain:8080]", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener start", "level": "INFO"}, {"text": "INFO: Started listener bound to [localhost.localdomain:8080]", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.HttpServer start", "level": "INFO"}, {"text": "INFO: [HttpServer-3] Started.", "level": "INFO"}, {"text": "Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 4.536 sec", "level": "INFO"}, {"text": "Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow", "level": "INFO"}, {"text": "INFO: Stopped listener bound to [localhost.localdomain:8080]", "level": "INFO"}], "level": "INFO"}}

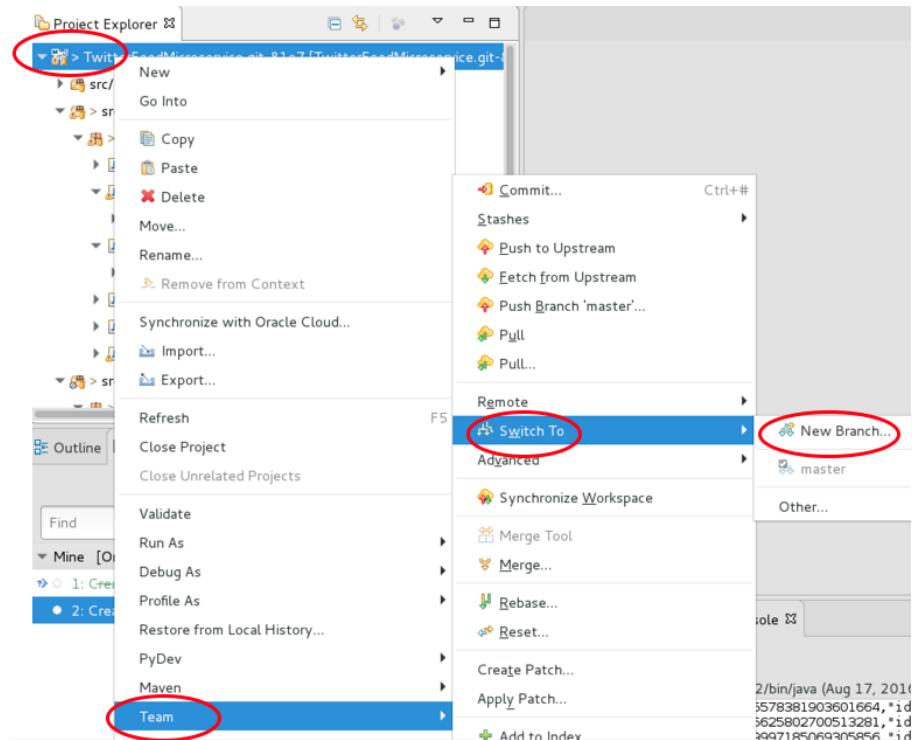
Results :
Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
```

## 새 Branch 및 Commit 코드 생성

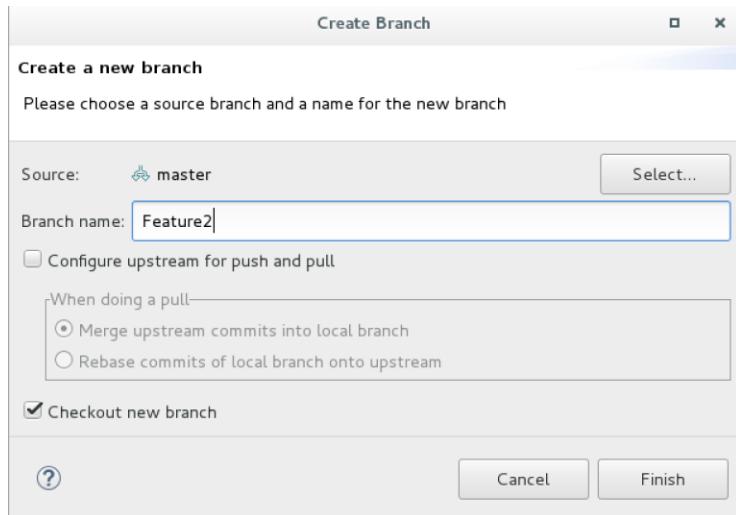
### Branch 와 Commit 코드 생성

#### STEP 20: 새 Branch 와 Commit 코드 생성

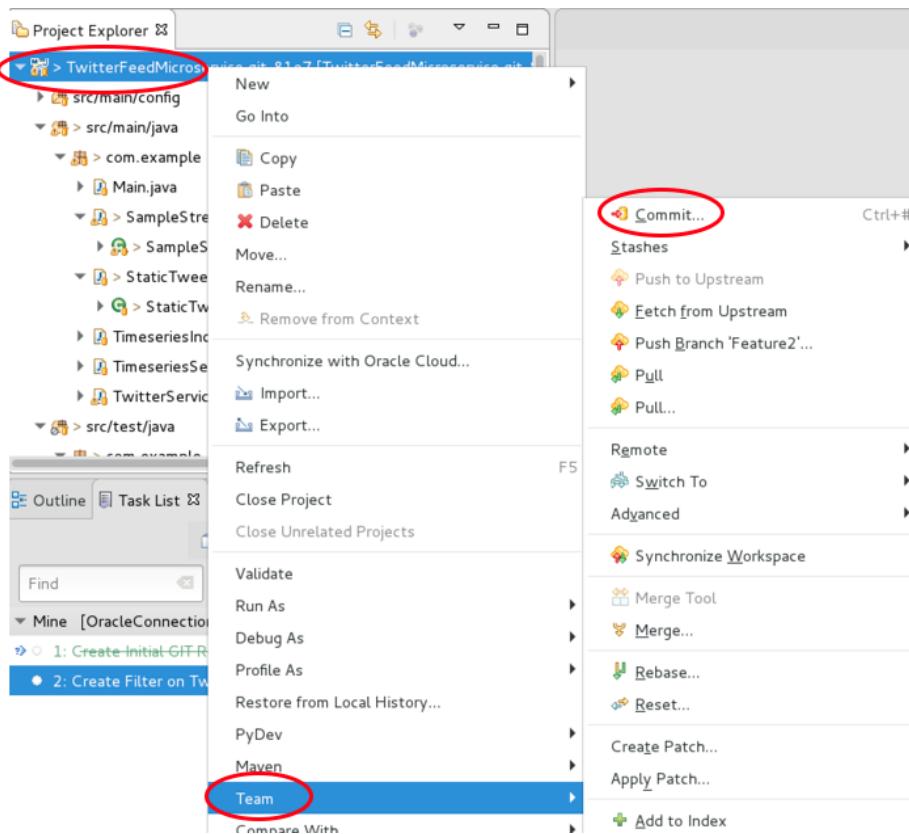
- Git 저장소에 새로운 branch 를 만들려면 **TwitterMicroservice** 의 오른쪽 클릭한 다음 **Team> Switch To > New branch** 를 선택한다.



- Branch 이름을 입력하고 Finish(마침)을 클릭합니다.

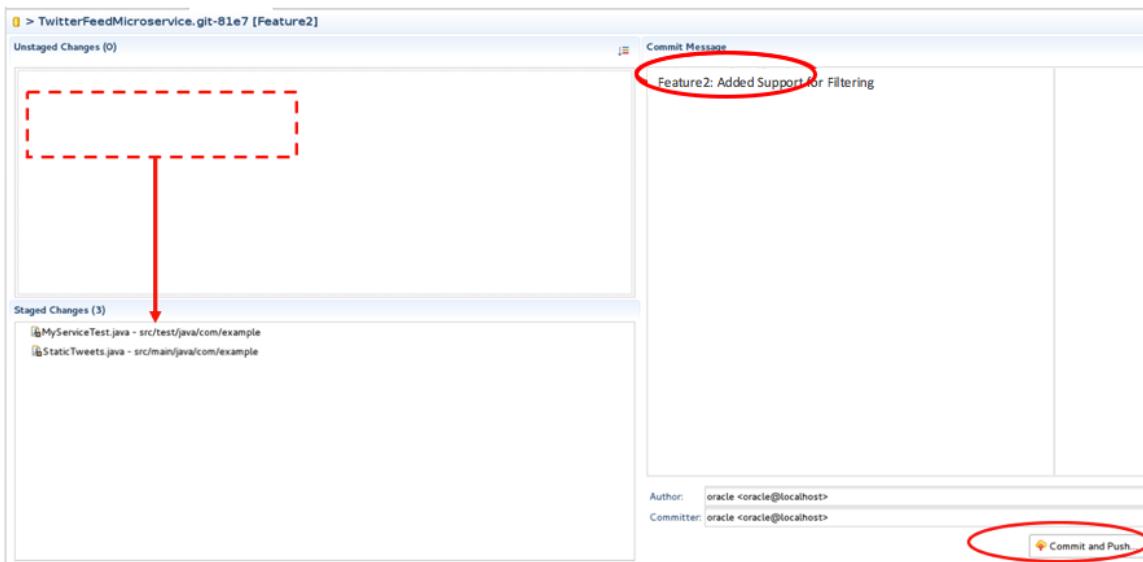


- 이제 **TwitterFeedMicroservice** 를 마우스 오른쪽 버튼으로 클릭한 다음 **Team> Commit** 을 선택하여 해당 지점에 코드를 커밋 할 수 있습니다.



- 커밋 메시지 상자에 "Feature2 : **Added Support for Filtering**"를 입력하고 변경된 파일을 **Staged Changes** 패널에 끌어다 놓은 다음 **Commit and Push** 를 클릭합니다.

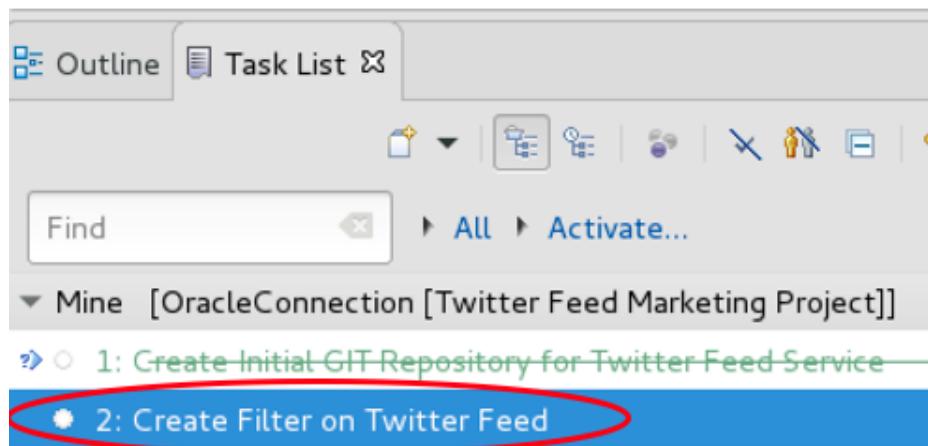
참고 : 기본 작성자와 커미터를 현재 "persona"와 일치하도록 변경할 수 있습니다. 그러나 이 랩 가이드를 위해 기본값을 그대로 사용합니다.



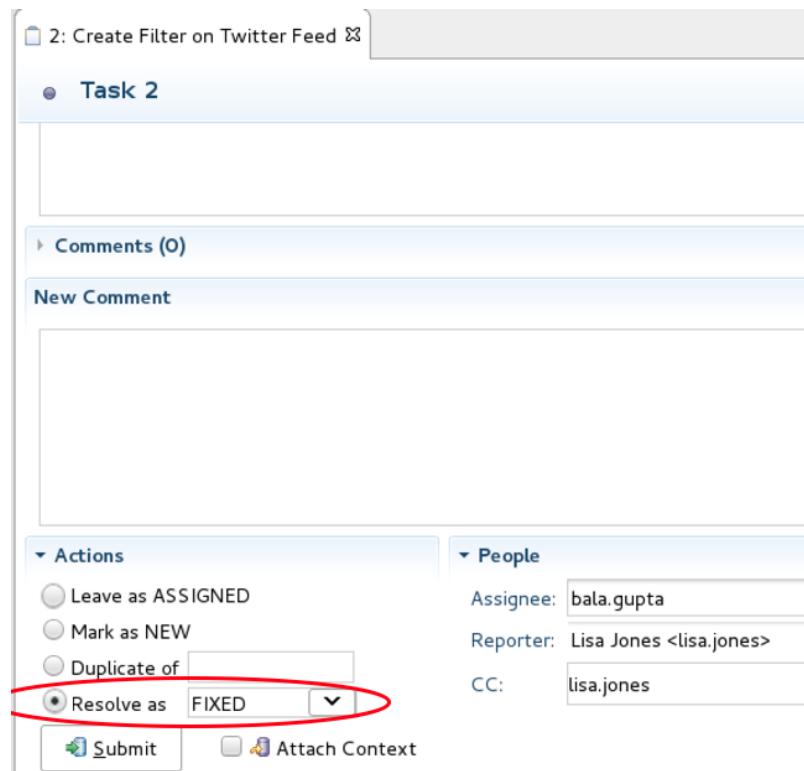
- Push Branch Feature2** 대화 상자의 Default(기본 값)를 수락하고 **Next(다음)**을 클릭합니다.
- Push Confirmation(확인) 대화 상자에서 **Finish(마침)** 버튼을 클릭합니다.
- OK(확인)** 결과 대화 상자에서 **OK(확인)**을 클릭합니다.

#### STEP 21: Create Filter 작업 완료

- Eclipse Task List(작업 목록)에서 **Create Filter on Twitter Feed**를 더블클릭합니다.



- Create Filter on Twitter Feed** 에서 Actions 섹션으로 스크롤 합니다. Click on **Resolve as FIXED**, and then click on the **Submit** button



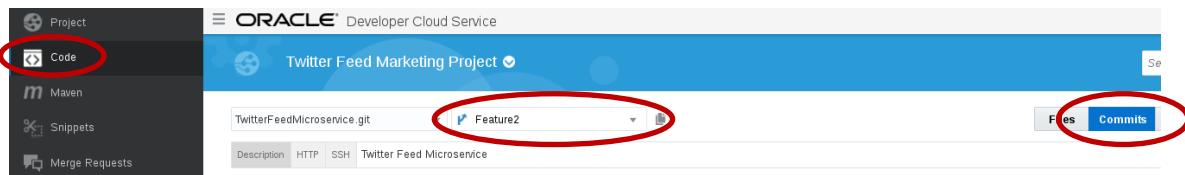
## 병합 요청 생성

### STEP 22: 스프린트 상태 검토 및 병합 요청 생성

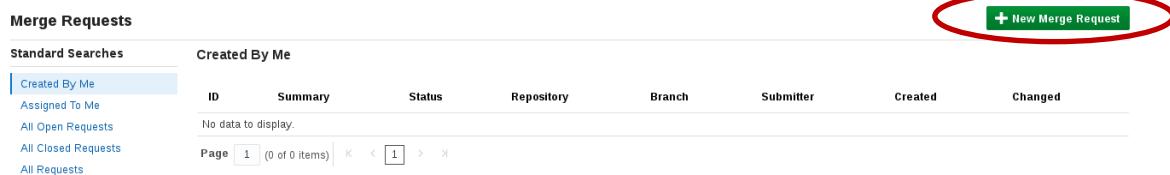
- 브라우저의 개발자 클라우드 서비스 대시 보드로 돌아가서 **Agile** 을 선택하십시오. 기본 게시판이 마이크로 서비스로 설정되지 않은 경우 Board 필터 찾기를 모두로 설정하고 마이크로 서비스 보드를 선택하십시오.
- "Active Sprints" 버튼을 클릭합니다. Feature 2 가 **Verify Code** 열에 있어야 합니다.

Swimlanes	Columns
Verify Code (1)	Completed (1)
Feature 2 Create Filter on Twitter Feed 	Task 1 Create Initial GIT Repository for Twitter Feed Service 
<small>RESOLVED : FIXED</small>	<small>VERIFIED : FIXED</small>

- 다음으로 탐색 패널에서 **Code** 를 클릭하고 **Feature2** Branch 을 선택한 다음 **Commits** 하위 탭을 클릭합니다. 최근 commit 내용을 확인합니다.



- Bala Gupta 는 검색 필터를 추가하는 작업을 완료한 후 병합 요청을 작성하여 Lisa Jones 에게 할당할 수 있습니다. **Merge Requests** 을 클릭한 다음 **New Request** 를 클릭합니다..



- 새 병합 요청에 다음 정보를 입력하고 **Next(다음)**을 클릭합니다.

Repository: TwitterFeedMicroservice.git  
 Target Branch: master  
 Review Branch: Feature2

New Merge Request

Branch      Details      Description

\* Repository: TwitterFeedMicroservice.git

\* Target Branch: master

\* Review Branch: Feature2

oracle Today  
 3b75f2a Feature2: Added Support for Filtering

- 다음 정보를 세부 정보로 입력하고 **Create(생성)**을 클릭합니다.

Summary: Merge Feature 2 into master  
 Reviewers: Lisa Jones (or current user in non-multi user env)

New Merge Request

Branch      Details      Description

Linked Issues: Search and Link Issues

Linked Builds: Search and Link Build Jobs

Summary: Merge Feature 2 into master

\* Reviewers: Lisa Jones

- Write 박스에서 다음 주석을 입력한 다음 Comment 버튼을 클릭한 후 다음을 입력합니다.

"I added the ability to add a filter request to the end of the URL - e.g. statictweets/alpha"

The screenshot shows a GitHub commit page for a pull request. At the top, there's a list of commits by Bala Gupta. Below that is a comment input field with tabs for 'Write' (which is selected), 'Preview', and 'Markdown Reference'. The 'Write' tab contains the text "I added the ability to add a filter request to the end of the URL - i.e. statictweets/alpha". A 'Comment' button is located at the bottom right of the input field.

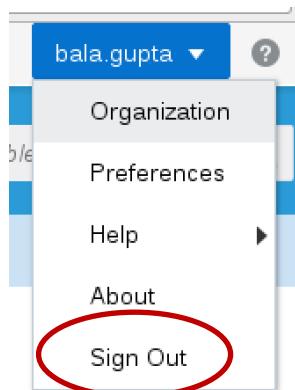
## 리자 존스로 branch Merge

다음 단계에서 "Lisa"는 "Bala"로 작성한 branch 를 마스터에 병합합니다.

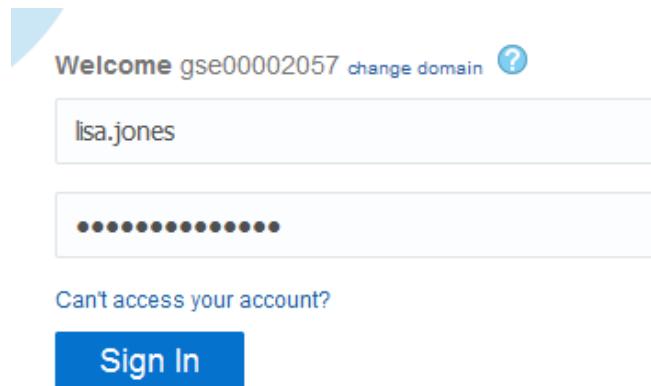
참고 : 단일 사용자 환경을 사용하는 경우 다음 단계를 건너 뛰고 다음 단계로 이동하십시오. "Merge Requests"

**STEP 23:** Bala Gupta 로 로그 아웃하고 Lisa Jones 로 로그인하십시오.

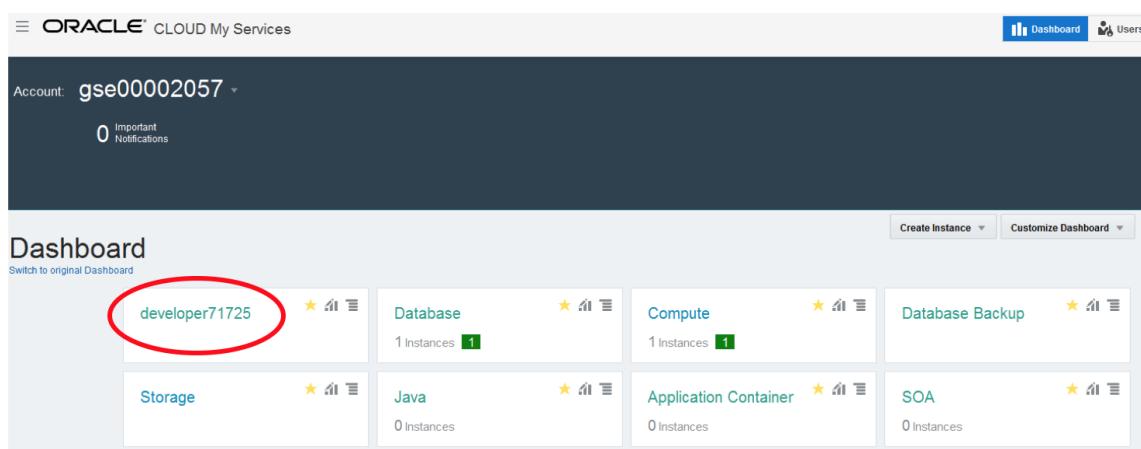
- 화면 오른쪽 상단 모서리에 있는 bala.gupta 드롭 다운 목록을 클릭합니다.  
SignIn(로그인)을 선택합니다.



- 이전에 문서화 된 단계에 따라 다음 URL 로 이동하십시오. <http://cloud.oracle.com>, 창의 맨 위 오른쪽 모서리에있는 **Sign In** 을 클릭하십시오. 올바른 데이터 센터를 선택하고 내 서비스 버튼을 클릭 한 다음 올바른 **Identity Domain** 을 입력하고 이동을 클릭하십시오.
- 사용자 이름을 입력하고 올바른 암호를 입력합니다. SignIn(로그인)을 클릭합니다.



- 대시 보드가 표시되면 **Developer Cloud Service** 를 클릭합니다.



- 개발자 클라우드 서비스 대시 보드에서 **Open Service Console** 버튼을 클릭합니다.

Service Details: developer71725 (Oracle Developer Cloud Service)

Overview (for August 2016)  
100% uptime

Business Metrics (as of 10 hours 53 minutes ago)  
0 current disk usage

Service Status - August 2016

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Legend: Before Activation (light blue), Service Up (green), Planned Outage (yellow), Service Incident (red)

Additional Information

Plan:	Trial Developer Service	Data Center:	EMEA Commercial 2 - Amsterdam
Service Start Date:	26. Juli 2016	Version:	16.1.0.0.0
Service End Date:	Not available	Status:	Active
Subscription ID:	536840104	Service Instance URL:	<a href="https://developer.em2.oracle.com">https://developer.em2.oracle.com</a>
Customer Account:	gse00002057 (US)		
CSI Number:	Not available		

- Twitter Feed Marketing Project 를 선택하세요

Member Favorites Owner All + New Project

Filter Projects

Twitter Feed Marketing Project  
Project to gather and analyze twitter data.

#### STEP 24: 병합 요청

- 앞으로 이동하기 전에 "리사 존스"는 Agile 를 클릭하여 Burn down 및 스프린트 보고서를 살펴볼 수 있습니다. 그리고 Reports 버튼을 클릭합니다.

Microservices ▾

Sprint Sprint 1 – Initial Development

Burndown Story Points

Burndown Chart Sprint Report

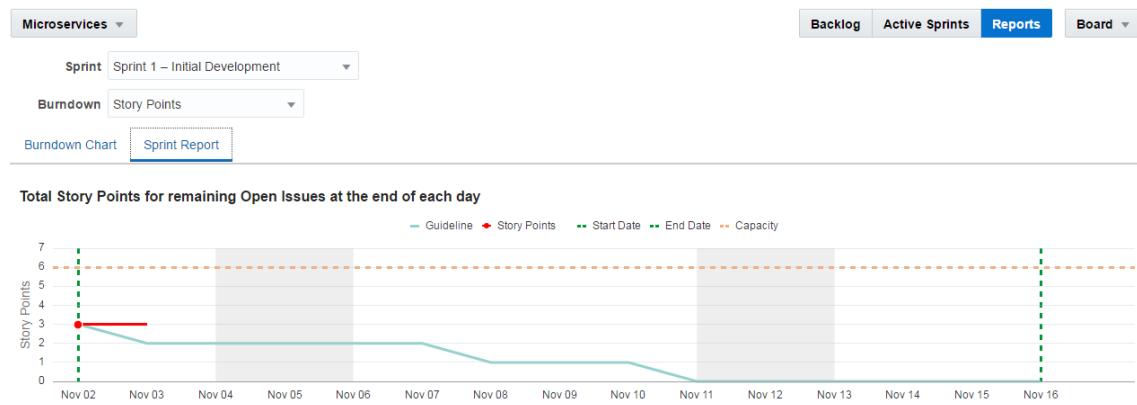
Total Story Points for remaining Open Issues after each event

Story Points

Nov 2016

Legend: Guideline (blue line), Story Points (red dots), Start Date (green dots), End Date (green dashed line), Capacity (orange dashed line)

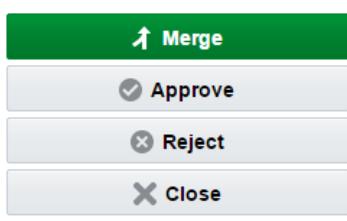
**Sprint Report** 클릭



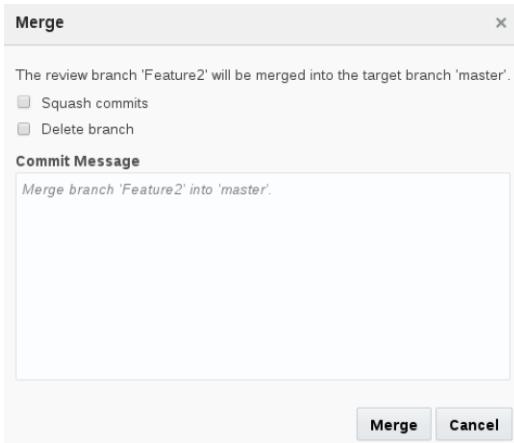
- 탐색 패널에서 **Merge Requests** 을 클릭합니다. **Assigned to Me** (할당된 위치)를 선택합니다. 검색이 완료되면 **Merge Feature 2 into master** 를 클릭하여 할당된 요청에 따라 선택합니다.

Created By Me	ID	Summary	Status
All Open Requests	5	Merge Feature 2 into master	OPEN
All Closed Requests			
All Requests			

- 요청이 로드되면 **Changed Files** 탭을 선택합니다. "리사"는 이제 branch 를 Approving, Rejecting 또는 Merging 하기 전에 변경 사항을 검토하고, 코멘트를 작성하고, 개발자에게 더 많은 정보를 요청할 수 있습니다.
- Merge** 버튼을 클릭합니다.

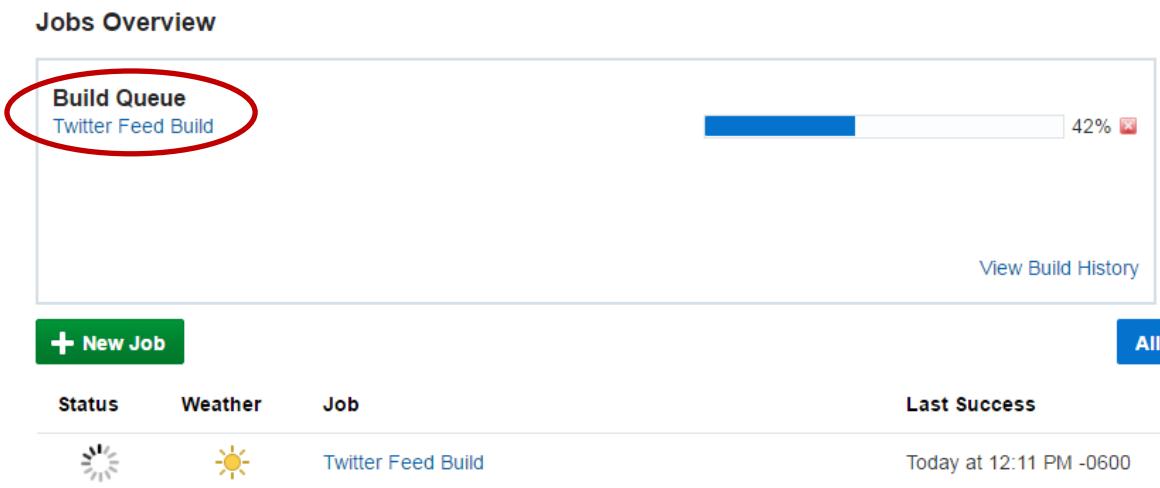


- 기본 값을 그대로 두고 확인 대화 상자에서 **Merge** 버튼을 클릭합니다.



- 이제 코드가 개발자 클라우드 서비스 저장소에 커밋 되었으므로 구축과 배포가 자동으로 시작됩니다. 탐색 패널에서 빌드를 클릭하고 큐에 있는 **Twitter Feed Build**를 확인해야 합니다.

**Jobs Overview**



Status	Weather	Job	Last Success
		Twitter Feed Build	Today at 12:11 PM -0600

- 빌드가 완료될 때까지 잠시 기다리거나 두번 클릭합니다. **Last Success** 은 build 가 완료될 때쯤 **Just Now** 될 것이다.

### Jobs Overview

The screenshot shows the 'Jobs Overview' section. At the top left is a 'Build Queue' placeholder. To its right is a 'View Build History' link. Below this is a green button labeled '+ New Job'. On the far right is a blue 'All' filter button. The main area is a table with columns: Status, Weather, Job, and Last Success. The first row shows a green checkmark icon, a sun icon, 'Twitter Feed Build', and 'Just now' circled in red. A blue 'All' filter button is located at the top right of the table.

Status	Weather	Job	Last Success
✓	☀	Twitter Feed Build	Just now

- Deploy** 를 클릭합니다. 배포 상태가 **Deployment update in progress** 상태를 변경할 때까지 기다린 후, **Last deployment succeeded – Just now** 로 변경됩니다.

### Deployments

The screenshot shows the 'Deployments' section. It lists a single deployment entry for 'JavaTwitterMicroservice'. The entry includes details: Deploy to ACCS em2 / gse00002921 / bala.gupta, Configuration TwitterFeedMicroserviceDeploy, Job / Build Twitter Feed Build / Latest Successful Build, Artifact target/twitter-microservice-example-dist.zip. Below this, a message 'Deployment update in progress' is shown, which is circled in red.

### Deployments

The screenshot shows the 'Deployments' section. It lists a single deployment entry for 'JavaTwitterMicroservice'. The entry includes details: Deploy to ACCS em2 / gse00002921 / bala.gupta, Configuration TwitterFeedMicroserviceDeploy, Job / Build Twitter Feed Build / Latest Successful Build, Artifact target/twitter-microservice-example-dist.zip. Below this, a message 'Last deployment succeeded -- Just now.' is shown, which is circled in red.

## 클라우드에서 테스트를 수행합니다.

- 서비스가 성공적으로 배포되면 JavaTwitterMicroservice 링크를 클릭합니다.

**Deployments**

JavaTwitterMicroservice

Deploy to ACCS em2 / gse00002921 / bala.gupta

Configuration TwitterFeedMicroserviceDeploy

Job / Build Twitter Feed Build / Latest Successful Build

Artifact target/twitter-microservice-example-dist.zip

Last deployment succeeded -- Just now.

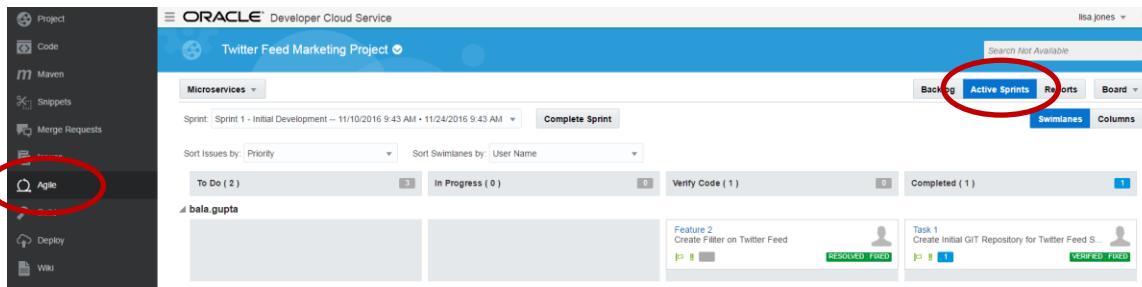
- 새 브라우저 탭 로드하면 URL의 끝에 **/statictweets** 추가하고 Enter 키를 눌러 원래 서비스를 테스트합니다.

```
{"tweets": [{"delete": {"status": {"id": 765578381903601664, "id_str": "765578381903601664", "user": {"id": 765625802700513281, "id_str": "765625802700513281", "user_id": 72625334, "user_id_str": "72625334", "user_name": "bala.gupta", "screen_name": "bala_gupta", "name": "Bala G", "location": "Mumbai, India", "description": "Software Engineer", "url": null, "profile_image_url": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_banner_url": "https://pbs.twimg.com/profile_banners/765625802700513281/1510000000", "profile_link_color": "#0080C0", "profile_sidebar_border_color": "#0080C0", "profile_sidebar_fill_color": "#E0FFFF", "profile_text_color": "#000000", "profile_use_background_image": true, "profile_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_text_color_hex": "#000000", "profile_use_background_image_hex": "#E0FFFF", "profile_image_width": 40, "profile_image_height": 40}, "text": "The VICTORIOUS SESSION: rel=nofollow>Facebook</a>, truncated": false, "in_reply_to_status_id": null, "in_reply_to_user_id": null, "in_reply_to_screen_name": null, "in_reply_to_favorited": false, "in_reply_to_retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "entities": {"hashtags": [{"text": "DolceAmoreHotSeat Wave"}, {"text": "LizQuenlurve"}], "symbols": [{"text": "#"}, {"text": "@"}], "urls": [{"url": "https://t.co/9DGgpxOjzs"}], "user_mentions": [{"name": "Nick Courtney", "screen_name": "NickCourtney", "id": 23947154, "id_str": "23947154", "indices": [1, 14]}]}, {"id": 765626469619859456, "id_str": "765626469619859456", "user": {"id": 23947154, "id_str": "23947154", "user_id": 23947154, "user_id_str": "23947154", "user_name": "Nick Courtney", "screen_name": "NickCourtney", "name": "Nick Courtney", "location": "Philippines", "description": "Software Engineer", "url": null, "profile_image_url": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/1510000000", "profile_link_color": "#0080C0", "profile_sidebar_border_color": "#0080C0", "profile_sidebar_fill_color": "#E0FFFF", "profile_text_color": "#000000", "profile_use_background_image": true, "profile_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_text_color_hex": "#000000", "profile_use_background_image_hex": "#E0FFFF", "profile_image_width": 40, "profile_image_height": 40}, "text": "Sweet Memories Vinyl Records", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_user_id": null, "in_reply_to_screen_name": null, "in_reply_to_favorited": false, "in_reply_to_retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "entities": {"hashtags": [{"text": "Sweet Memories Vinyl Records"}], "symbols": [{"text": "#"}, {"text": "@"}], "urls": [{"url": "http://www.book.events"}], "user_mentions": [{"name": "Nick Courtney", "screen_name": "NickCourtney", "id": 23947154, "id_str": "23947154", "indices": [1, 14]}]}]}
```

- 이제 URL 에 **/statictweets/alpha** 추가한 후 Enter 키를 누릅니다. 이렇게 하면 alpha 포함된 텍스트가 반환됩니다.



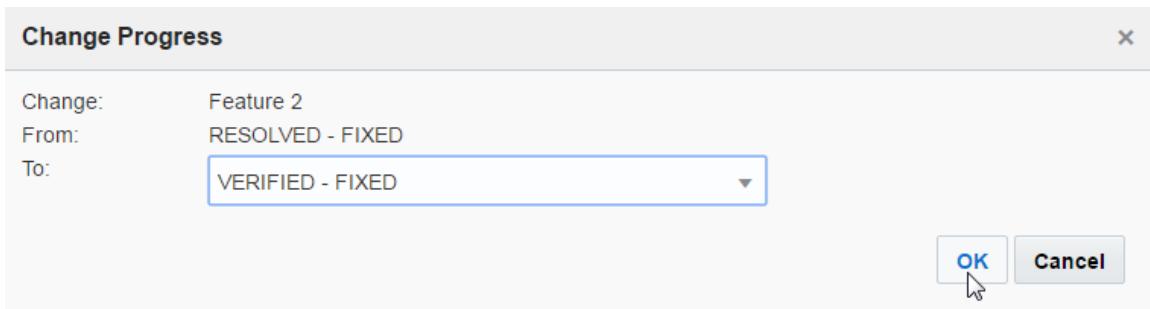
- 스프린트 기능을 완료하려면 좌측 탐색기에서 Agile 를 클릭하십시오. 그런 다음 **Active Sprints** 버튼을 클릭합니다.



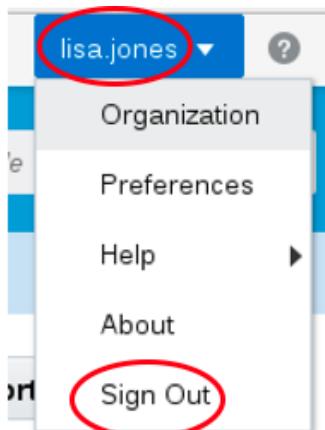
- Feature 2 를 Verify 에서 Completed 로 끌어다 놓으십시오.



- Set the Status to **VERIFIED - FIXED** and click **OK**



- 리사 존스 로그아웃



- 이제 lab 이 끝났습니다.

## 추가 실습과제 – Twitter Live Feed Credentials

### 트위터 앱 만들기

선택적인 과제이며, 정적 트위터 마이크로 서비스를 확장하여 라이브 트위터 데이터를 사용함으로써 개발자 클라우드 서비스에 대한 새로운 지식을 활용할 수 있습니다. 이 실습에서는 Twitter 응용 프로그램 자격 증명을 획득하고 이를 사용하여 마이크로 서비스의 라이브 트위터 피드를 조작합니다. 이 과제를 수행하기 위해 개인 계정을 사용하여 트위터에 로그인하고 자격 증명을 생성합니다. 그러나 이 실습의 응용 프로그램의 환경에서 이 계정은 제품 관리자에게 의해 제공된 것이고 제품 응용 프로그램의 자격을 의미하는 것으로 가정합니다.

버전 제어 시스템에서 코드 변경을 관리하기 위한 두 가지 옵션이 있습니다. 다중 사용자 워크 플로에 대해 더 실습하고 싶다면 이 기능에 대한 새 branch 를 시작하고 해당 branch 를 커밋하고 병합 요청을 만들고 병합을 승인 할 수 있습니다. 워크 플로에 익숙하다면 로컬 저장소에서 마스터로 전환하고 클라우드에서 최신 버전을 가져온 다음 이 실습을 위해 마스터에게 직접 커밋하고 직접 푸시 할 수 있습니다. 이것은 방법 B 가 될 것입니다.

#### STEP 25: 새로운 트위터 앱 만들기

마이크로 서비스에 대한 고유 한 트위터 자격 증명을 생성하려면 트위터에 로그인하여 이 프로젝트에 대한 새 응용 프로그램을 만든 다음 액세스 토큰을 생성해야합니다.

- <https://apps.twitter.com>.으로 이동합니다. SignIn 링크를 클릭합니다.

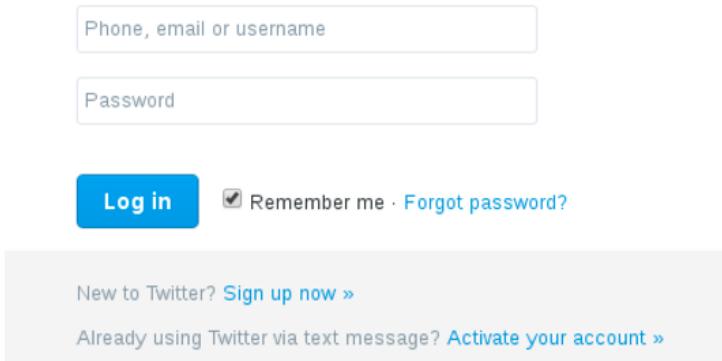


## Twitter Apps

Please [sign in](#) with your Twitter Account to create :

- 이미 트위터 사용자 인 경우 트위터 자격 증명을 사용하여 로그인하십시오. 그렇지 않으면 지금 가입하기 링크를 클릭하십시오.

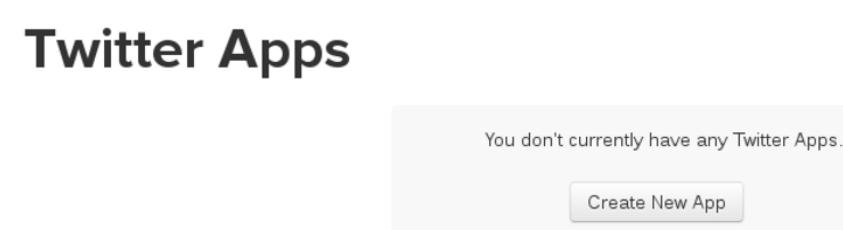
## Log in to Twitter



The screenshot shows the Twitter login interface. It features two input fields: 'Phone, email or username' and 'Password'. Below these is a blue 'Log in' button. To the right of the 'Log in' button are two links: 'Remember me' with a checked checkbox and 'Forgot password?'. Below the input fields, there are two links: 'New to Twitter? Sign up now >' and 'Already using Twitter via text message? Activate your account >'. The entire form is set against a light gray background.

- Once logged in, click on the **Create New App** button.

 Application Management



- 다음을 입력하고 Twitter 애플리케이션 생성 버튼을 클릭하십시오. 응용 프로그램 이름을 입력 할 때 이름 끝에 고유 한 것을 추가하십시오. 예 : 귀하의 이니셜 또는 이름.

Name :	<b>JavaTwitterMicroservice&lt;UniqueName&gt;</b>
Description :	<b>A Twitter Feed Microservice</b>
Website :	<b><a href="https://cloud.oracle.com/acc">https://cloud.oracle.com/acc</a></b>
Developer Agreement :	<b>Click Yes</b>

## Create an application

### Application Details

**Name \***


Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

**Description \***


Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

**Website \***


Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL will be used to sign tweets created by your application and will be shown in user-facing authorization screens.  
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

**Callback URL**


Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth\_callback URL on the request token step, regular applications from using callbacks, leave this field blank.

### Developer Agreement

Yes, I have read and agree to the Twitter Developer Agreement.

[Create your Twitter application](#)

- Keys and Access Tokens** 탭을 클릭합니다.

## JavaTwitterMicroservicePCD

[Details](#)[Settings](#)[Keys and Access Tokens](#)[Permissions](#)

### Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key) XXXXXXXXXXXXXXXXXXXXXXXXX

Consumer Secret (API Secret) XXXXXXXXXXXXXXXXX

Access Level Read and write ([modify app permissions](#))

Owner ownername

Owner ID 1234567

- 페이지 하단에 있는 하위 폴더가 보이지 않으면 **Create my access tokens button** 를 클릭합니다.

### Your Access Token

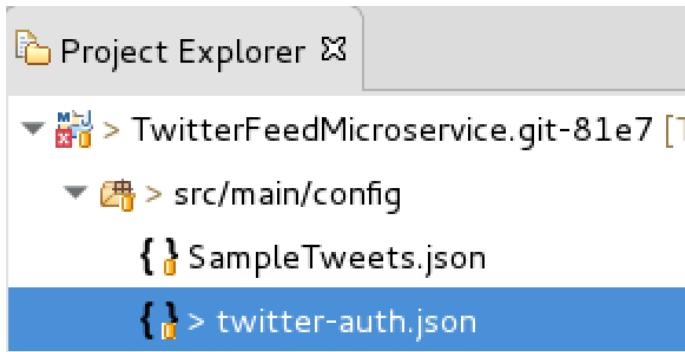
You haven't authorized this application for your own account yet.

By creating your access token here, you will have everything you need to use your application's current permission level.

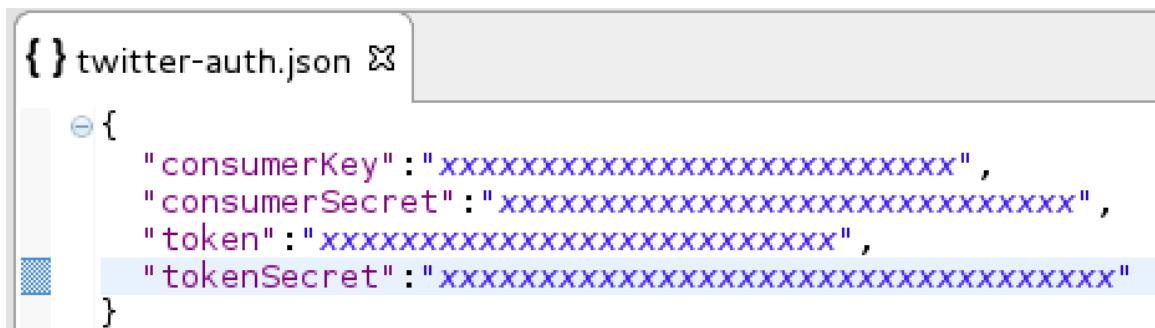
#### Token Actions

[Create my access token](#)

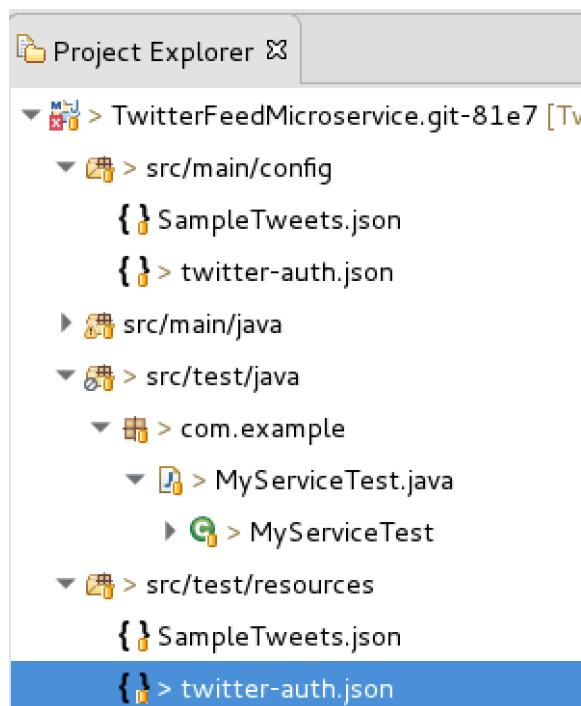
- 참고 : 방법 B 를 따르는 중이라면 Eclipse 에서 코드 수정을 시작하기 전에 마스터 branch 로 전환하고 원격 저장소에서 가져와야 합니다.
- Eclipse 로 돌아가서 프로젝트 탐색기 탭에서 **TwitterFeedMicroservices.git> src / main / config** 를 확장하고 **twitter-auth.json** 을 두 번 클릭하여 소스를 로드하십시오.



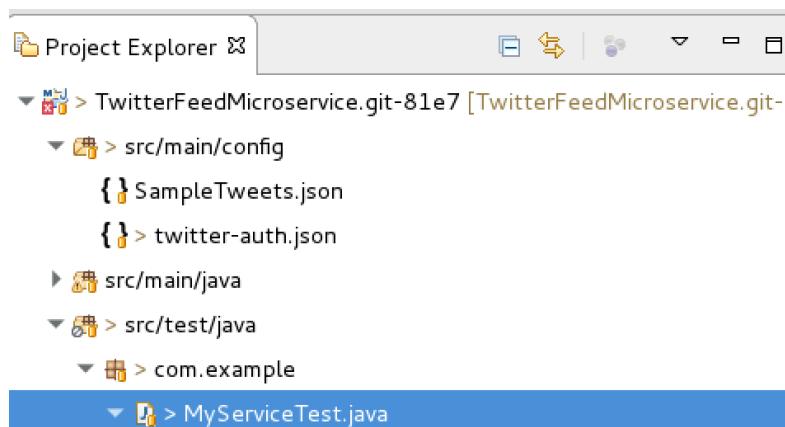
- 응용 프로그램 컨테이너 클라우드에 배포 할 파일입니다. consumerKey, consumerSecret, token 및 tokenSecred 의 xxx 를 Twitter 애플리케이션 관리 페이지에 있는 **Consumer Key** (API 키), **Consumer Secret** (API 비밀번호), **Access Token** and **Access Token Secret** 로 대체하여 이 파일을 편집하십시오.



-  이클립스에서는 SaveAll(모두 저장)아이콘을 클릭합니다.
- 로컬에서 테스트 할 수 있으므로 테스트 코드의 twitter-auth.json 자격 증명을 업데이트하여 동일한 단계를 반복하겠습니다. **TwitterFeedMicroservices.git> src / test / resources> twitter-auth.json** 에있는 파일을 열고 업데이트하십시오. 업데이트가 완료되면 모두 저장 아이콘을 클릭하십시오.



- 이제 온라인 트위터 피드 테스트를 허용하는 코드의 주석을 제거해 보겠습니다. 프로젝트 탐색기를 사용하여 **TwitterFeedMicroservice.git> src / test / java> com.example> MyServiceTest.java** 파일을 엽니다.



- MyServiceTest.java 파일에서 testGetTweets () 메소드를 찾고 해당 메소드를 둘러싼 주석을 제거하십시오.

```

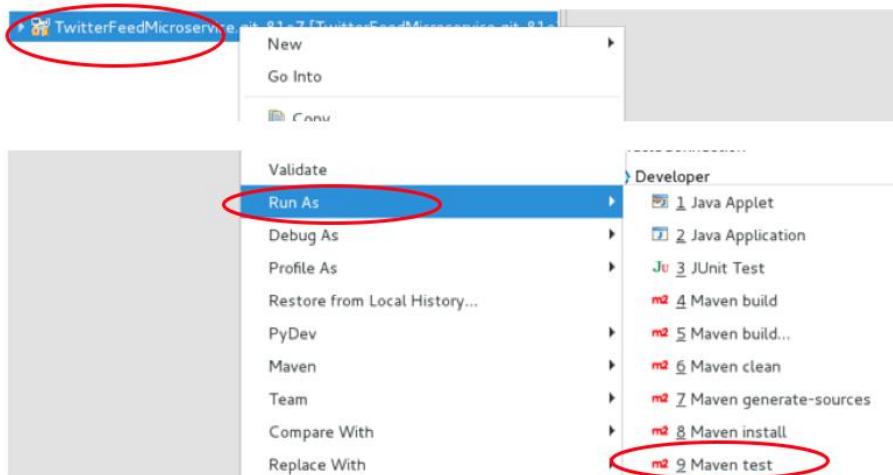
@Test
public void testGetStaticTweets() {
    String responseMsg = target.path("statictweets").request().get(String.class);
    assertNotNull(responseMsg);
}

@Test
public void testGetStaticSearchTweets() {
    String responseMsg = target.path("statictweets/alpha").request().get(String.class);
    assertNotNull(responseMsg);
}

/*
@Test
public void testGetTweets() {
    String responseMsg = target.path("tweets").request().get(String.class);
    assertNotNull(responseMsg);
}
*/

```

-  SaveAll(모두 저장)아이콘을 클릭합니다.
- TwitterFeedMicroservice 를 오른쪽 클릭하고 Run As> Maven Test 를 선택하여 테스트를 실행하십시오.



- 테스트를 실행 한 후 testGetTweets () 메서드는 "The client read 10 messages !,"메시지를 반환하고 모든 테스트가 성공적으로 완료됩니다.

```
-----  
TESTS  
-----  
Running com.example.MyServiceTest  
Aug 18, 2016 11:40:40 AM org.glassfish.grizzly.http.server.NetworkLister  
INFO: Started listener bound to [localhost.localdomain:8080]  
Aug 18, 2016 11:40:40 AM org.glassfish.grizzly.http.server.HttpServer  
INFO: [HttpServer] Started.  
[{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877038449459},  
 {"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877040546611},  
 {"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877039278899},  
 {"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877037605273},  
 {"delete": {"status": {"id": 678115516905574401, "id_str": "678115516905574401"},  
 {"delete": {"status": {"id": 335505648538234881, "id_str": "335505648538234881"},  
 {"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 766298770405482496},  
 {"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877040952115},  
 {"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877039296512},  
 {"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877039713075},  
The client read 10 messages!  
Aug 18, 2016 11:40:42 AM org.glassfish.grizzly.http.server.NetworkLister
```

- 방법 A 를 따르면서이 새로운 기능을 라이브 트위터 피드에 액세스 할 수 있게되었으므로이 문서에서 사용 된 이전 단계에 따라 코드를 클라우드에 적용 할 수 있습니다. 일단 커밋되면 개발자 클라우드 서비스를 사용하여 병합 요청을 만든 다음 해당 요청을 승인합니다. 마스터 branch 가 업데이트되면 응용 프로그램 컨테이너 클라우드 서비스에 대한 자동 빌드 및 배포가 수행됩니다. 계속 진행하기 전에 배포가 성공했는지 확인하십시오.
  - 방법 B 를 따르면서이 새로운 기능이 라이브 트위터 피드에 액세스 할 수 있게되었으므로이 문서에서 사용 된 이전 단계에 따라 코드를 클라우드에 적용 할 수 있습니다. 그러면 자동 빌드가 시작되고 응용 프로그램 컨테이너 클라우드 서비스 배포가 개발자 클라우드 서비스에 의해 수행됩니다. 계속 진행하기 전에 배포가 성공했는지 확인하십시오.
  - 두 가지 방법 모두 이제 응용 프로그램 컨테이너 클라우드 서비스 URL 끝에 추가 /tweets 을 추가하고 라이브 트윗을 가져올 수 있습니다.
  - 아래의 예제에서는 응용 프로그램이 반환된 후에 반환되는 라이브 트위트를 보여 줍니다.

```
← → ↻ https://javatwittermicroservice-gse00002055.apaas.em2.oraclecloud.com/tweets

{"tweets": [{"delete": {"status": {"id": 628210773559357440, "id_str": "628210773559357440", "user_id": 766301530232328192, "user_name": "Client\u003c/a\u003e", "created_at": "Thu Aug 18 15:51:36 +0000 2016", "text": "\u003cdiv style='background-color: #f8f8f8; padding: 5px; border-radius: 5px; font-family: sans-serif; font-size: 14px; margin-bottom: 10px;'\u003e\nClient\u003c/a\u003e", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_user_id": null, "source": "\u003ca href='http://twitter.com' rel='nofollow'\u003e\nClient\u003c/a\u003e\u003c/a href='http://twitter.com' rel='nofollow'\u003e", "protected": false, "verified": false, "followers_count": 8970, "friends_count": 2011, "utc_offset": 3600, "time_zone": "London", "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url": "https://pbs.twimg.com/profile_images/1720338750/537_normal.gif", "profile_banner_url": "https://pbs.twimg.com/profile_banners/766301530232328192/1534510000", "profile_link_color": "#000000", "profile_sidebar_fill_color": "#E0E0E0", "profile_text_color": "#000000", "profile_use_background_image": true, "profile_background_color": "#F0F0F0", "profile_background_image_url": "https://abs.twimg.com/images/themes/theme14/bg.gif", "profile_background_image_url_https": "https://abs.twimg.com/images/themes/theme14/bg.gif", "profile_link_color_hex": "#000000", "profile_sidebar_fill_color_hex": "#E0E0E0", "profile_text_color_hex": "#000000"}}, {"id": 442053406, "id_str": "442053406", "name": "irakliytovlies", "screen_name": "irakliytovlies", "location": "Russia", "url": null, "description": "I am a software developer, currently working at Oracle Cloud Infrastructure. I have been working with Java for over 10 years and I am also interested in machine learning and deep learning.", "profile_image_url": "https://pbs.twimg.com/profile_images/1720338750/537_normal.gif", "profile_banner_url": "https://pbs.twimg.com/profile_banners/766301530232328192/1534510000", "profile_link_color": "#000000", "profile_sidebar_fill_color": "#E0E0E0", "profile_text_color": "#000000", "profile_use_background_image": true, "profile_background_color": "#F0F0F0", "profile_background_image_url": "https://abs.twimg.com/images/themes/theme14/bg.gif", "profile_background_image_url_https": "https://abs.twimg.com/images/themes/theme14/bg.gif", "profile_link_color_hex": "#000000", "profile_sidebar_fill_color_hex": "#E0E0E0", "profile_text_color_hex": "#000000"}]}]
```

## Appendix 2 – Installing Eclipse

### 이클립스 다운로드 및 설치

이 부록에서는 Oracle Plugin 가 포함된 이클립스를 다운로드하여 설치합니다.

#### STEP 26: 이클립스 다운로드

- 다음 URL로 이동합니다. <http://www.oracle.com/technetwork/developer-tools/eclipse/downloads/index.html>
- licensing agreement** 을 수락한 후 운영 체제에 필요한 **Neon** 버전을 선택합니다.

The screenshot shows the Oracle Technology Network > Developer Tools > Enterprise Pack for Eclipse > Downloads page. On the left, there's a sidebar with various tools like JDeveloper, NetBeans, Application Testing Suite, etc. The main content area has tabs: Overview, Downloads (selected), Documentation, Community, Learn More. Below the tabs, it says 'Oracle Enterprise Pack for Eclipse (12.2.1.3.1)'. It asks for accepting the OTN License Agreement. Two radio buttons are shown: 'Accept License Agreement' (selected) and 'Decline License Agreement'. A red circle highlights the 'Accept License Agreement' button. Below that, under 'Network Installer', it says 'Create a custom installation of Eclipse with Oracle Tools, by choosing the desired OEPE release and the required capabilities.' A 'Launch Network Installer' link is present. Under 'Packaged Distributions', it says 'These distributions include Eclipse with Oracle Tools already installed. Just download and unzip.' A table lists distributions for Windows 64-bit, Linux 64-bit, and Mac OS X 64-bit, with columns for Mars and Neon. A red circle highlights the 'Neon' column header. In the table, the 'Neon' download links are circled in red.

	Mars	Neon
Windows 64-bit	<a href="#">Download</a>	<a href="#">Download</a>
Linux 64-bit	<a href="#">Download</a>	<a href="#">Download</a>
Mac OS X 64-bit	<a href="#">Download</a>	<a href="#">Download</a>

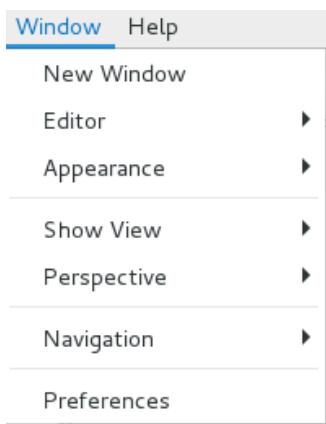
- eclipse 를 다운로드한 후 zip 파일을 압축하여 설치합니다.

### 필요에 따라 구성 요소를 자동으로 구성합니다.

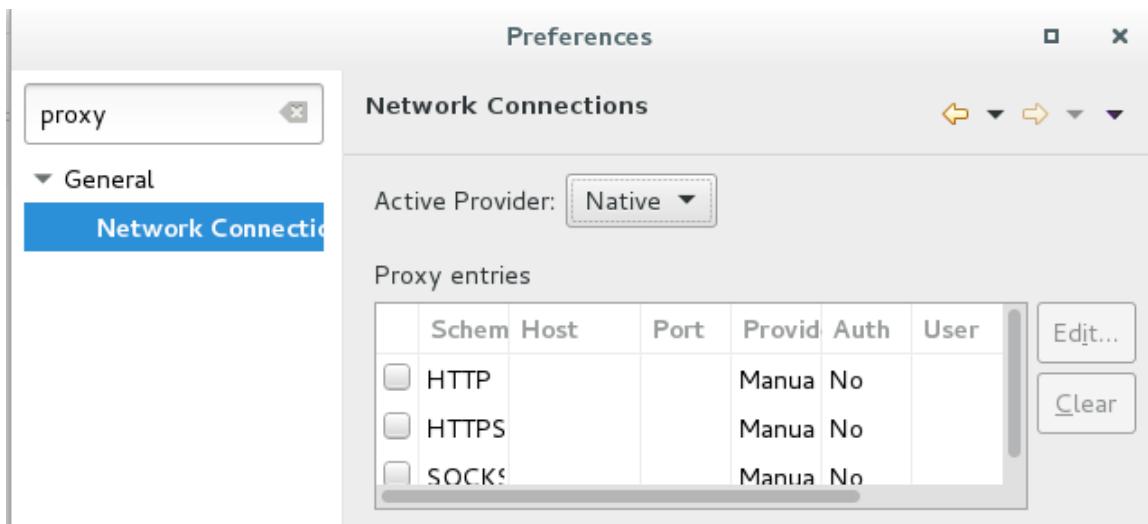
#### STEP 27: Configuring Proxies

방화벽 뒤에서 Eclipse 를 실행 중이고 프록시 설정을 구성해야하는 경우 몇 가지 사항을 업데이트해야합니다. 먼저, Eclipse 의 프록시가 설정되었는지 확인하고, 다음으로 Maven 프록시 설정을 업데이트해야하며 마지막으로 Oracle Plugin 이 프록시 설정과 함께 작동하는지 확인해야합니다.

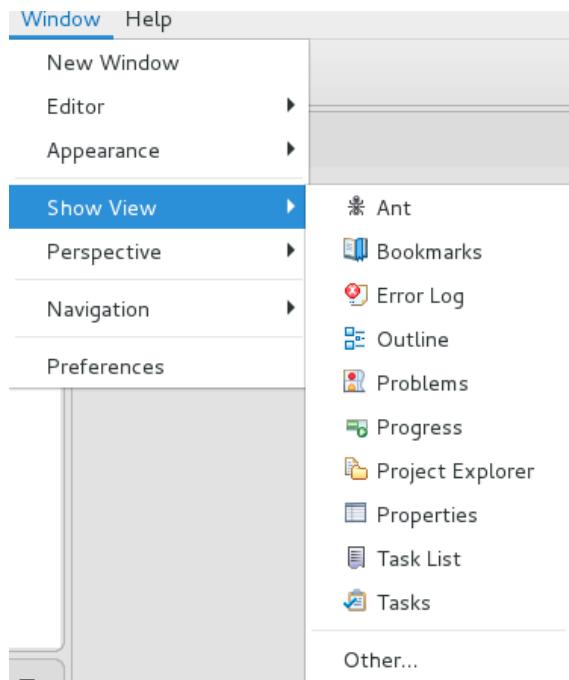
- Eclipse 구성을 설정하려면 Eclipse 를 열고 Properties 메뉴로 이동하십시오. 운영 체제에 따라, 드롭 다운은 Eclipse> 환경 설정 또는 윈도우> 환경 설정에서 찾을 수 있습니다



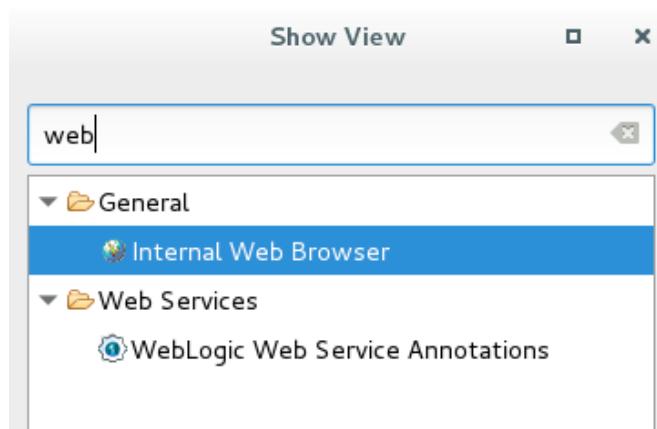
- 환경 설정 패널에서 검색 창에 "proxy"를 입력하고 **Network Connections** 을 클릭하십시오. 활성 공급자에 대해 **Native** 를 선택하십시오. 이 설정은 정상적으로 작동하지만 Eclipse 가 실행되는 시스템에서 프록시 설정을 올바르게 구성해야합니다. Windows, MAC OS 또는 Linux. 수동을 선택해도 작동하지만 일부 플러그인은 기본 운영 체제의 프록시를 구성해야합니다.



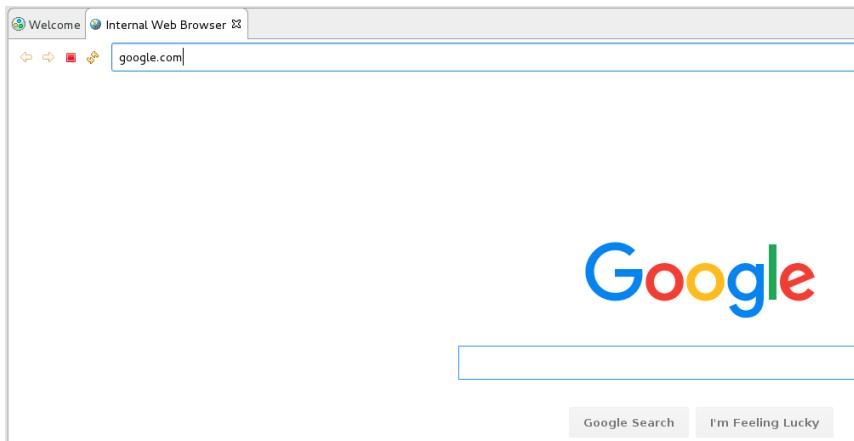
- 연결이 작동하는지 테스트하려면 메뉴>보기 표시> 기타 메뉴 옵션을 선택하십시오.



- 검색 필드에 "Web"을 입력하고 Internal Web Browser 를 선택하고 OK(확인)를 클릭합니다.

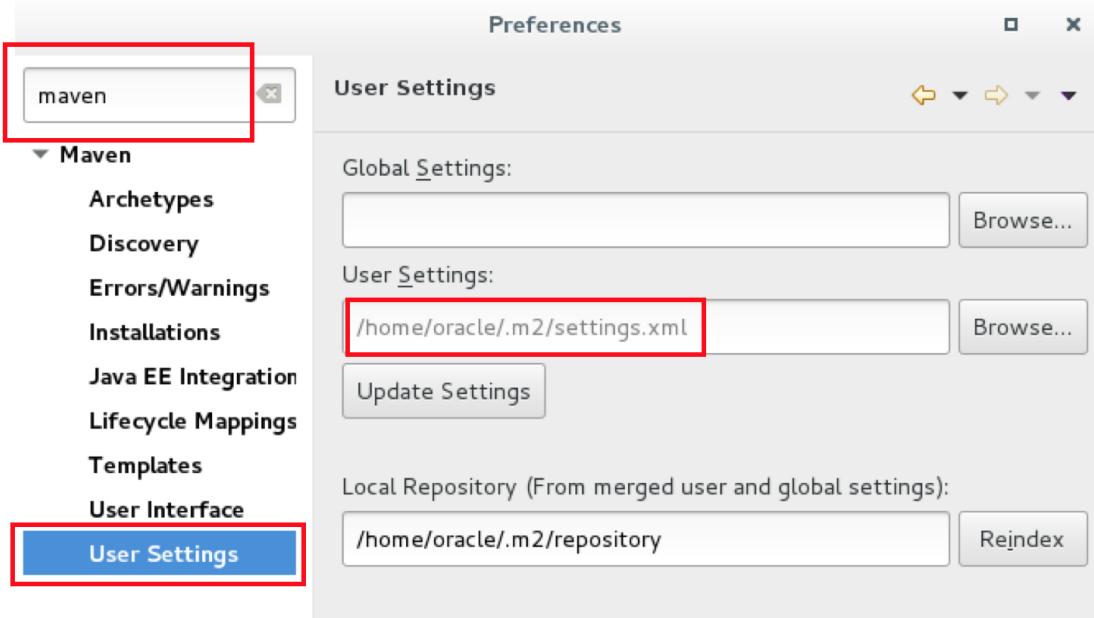


- URL 을 입력하고 Enter 키를 눌러 프록시 설정을 테스트합니다.



#### STEP 28: 이클립스/ Maven 프록시 업데이트

- Eclipse > Preference** 또는 **Window > Preferences**에서 검색 상자에 **Maven**을 입력하십시오. Maven 사용자 설정을 클릭하십시오. settings.xml 파일이 있는 디렉토리를 적어 두십시오. 아래 예에서 Maven 사용자 설정은 **/home/oracle/.m2** 디렉토리에 있습니다.



- 이클립스를 닫습니다.
- settings.xml 파일이 있는 디렉토리가 존재하지 않으면 디렉토리를 만듭니다. 이 예제에서는 .m2 디렉토리를 생성합니다. 또한 존재하지 않으면 settings.xml 파일을

만듭니다. `settings.xml` 파일에 다음을 추가하십시오 (참고 : 올바른 호스트, 포트, `nonProxyHosts`, 사용자 이름 및 비밀번호 설정을 사용해야합니다).

```
<?xml version="1.0" encoding="UTF-8"?>
<settings xmlns="http://maven.apache.org/SETTINGS/1.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.1.0
  http://maven.apache.org/xsd/settings-1.1.0.xsd">
  <proxies>
    <proxy>
      <active>true</active>
      <protocol>http</protocol>
      <username>proxyuser</username>
      <password>proxypass</password>
      <host>www-proxy.us.oracle.com</host>
      <port>80</port>
      <nonProxyHosts>local.net|some.host.com</nonProxyHosts>
    </proxy>
    <proxy>
      <active>true</active>
      <protocol>https</protocol>
      <username>proxyuser</username>
      <password>proxypass</password>
      <host>www-proxy.us.oracle.com</host>
      <port>80</port>
      <nonProxyHosts>local.net|some.host.com</nonProxyHosts>
    </proxy>
  </proxies>
</settings>
```

- 이클립스를 다시 로드합니다.