* 두 개 이상의의 컨테이너를 이용한 웹사이트 구축

* Link 사용해서 Mariadb 를 이용한 JSP 페이지

```
master@master-virtual-machine:~$ docker run -it -d -e MYSQL_ROOT_PASSWORD=123456 --name mariadb1 mariadb

master@master-virtual-machine:~$ docker run -it -d -p 8080:8080 --link mariadb1:lmariadb1 --name tomcat1 tomcat

master@master-virtual-machine:~$ docker exec -it tomcat1 /bin/bash
root@cdb727e4e342:/usr/local/tomcat# mkdir -p ./webapps/ROOT/WEB-INF

master@master-virtual-machine:~$ docker exec -it tomcat1 pwd
/usr/local/tomcat

master@master-virtual-machine:~$ docker exec -it tomcat1 ls
BUILDING.txt       NOTICE             RUNNING.txt  lib              temp       work
CONTRIBUTING.md    README.md bin      logs         webapps
LICENSE            RELEASE-NOTES      conf         native-jni-lib   webapps.dist

master@master-virtual-machine:~$ docker exec -it tomcat1 mkdir -p ./webapps/ROOT/WEB-INF
```

* 기본 웹페이지 구성

```
master@master-virtual-machine:~$ vi index.jsp
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%
	out.println( "Hello JSP" );
%>
```

```
master@master-virtual-machine:~$ docker cp ./index.jsp tomcat1:/usr/local/tomcat/webapps/ROOT/
```

* http://localhost:8080/

* JDBC 연동 웹페이지 구성

```
master@master-virtual-machine:~$ wget https://downloads.mariadb.com/Connectors/java/connector-java-2.6.2/mariadb-java-client-2.6.2.jar
--2020-09-24 14:37:05--  https://downloads.mariadb.com/Connectors/java/connector-java-2.6.2/mariadb-java-client-2.6.2.jar
Resolving downloads.mariadb.com (downloads.mariadb.com)... 172.67.32.229, 104.20.68.208, 104.20.67.208, ...
접속 downloads.mariadb.com (downloads.mariadb.com)|172.67.32.229|:443... 접속됨.
HTTP request sent, awaiting response... 200 OK
Length: 621278 (607K) [application/java-archive]
Saving to: 'mariadb-java-client-2.6.2.jar'

mariadb-java-client 100%[===================>] 606.72K  --.-KB/s    in 0.05s

2020-09-24 14:37:05 (11.5 MB/s) - 'mariadb-java-client-2.6.2.jar' saved [621278/621278]
```

```
master@master-virtual-machine:~$ ls
mariadb-java-client-2.6.2.jar  다운로드  바탕화면  사진  템플릿
```

공개 문서 비디오 음악

master@master-virtual-machine:~$ docker exec -it tomcat1 mkdir -p ./webapps/ROOT/WEB-INF/lib

master@master-virtual-machine:~$ docker cp ./mariadb-java-client-2.6.2.jar tomcat1:/usr/local/tomcat/webapps/ROOT/WEB-INF/lib

master@master-virtual-machine:~$ vi jdbc.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>

<%@ page import="java.sql.DriverManager" %>
<%@ page import="java.sql.Connection" %>
<%@ page import="java.sql.PreparedStatement" %>
<%@ page import="java.sql.ResultSet" %>
<%@ page import="java.sql.SQLException" %>

<%
	String url = "jdbc:mysql://lmariadb1:3306/mysql";
	String user = "root";
	String password = "123456";

	Connection conn = null;
	PreparedStatement pstmt = null;
	ResultSet rs = null;
```

```
        try {
                Class.forName( "org.mariadb.jdbc.Driver" );

                conn = DriverManager.getConnection( url, user, password );

                String sql = "select now()";
                pstmt = conn.prepareStatement( sql );

                rs = pstmt.executeQuery();
                if( rs.next() ) {
                        out.println( "현재시각 : " + rs.getString( "now" ) );
                }
        } catch( ClassNotFoundException e ) {
                System.out.println( "에러 : " + e.getMessage() );
        } catch( SQLException e ) {
                System.out.println( "에러 : " + e.getMessage() );
        } finally {
                if( rs != null ) rs.close();
                if( pstmt != null ) pstmt.close();
                if( conn != null ) conn.close();
        }
%>
```

```
master@master-virtual-machine:~$ docker cp ./jdbc.jsp tomcat1:/usr/local/tomcat/webapps/ROOT/
```

```
master@master-virtual-machine:~$ docker stop tomcat1
master@master-virtual-machine:~$ docker start tomcat1
```

FROM	기본 이미지를 지정한다. 최근에는 경량화된 운영체제인 Alpine LInux 를 많이 사용하기도 한다. FROM "사용할 이미지 이름" FROM centos:latest
MAINTAINER / LABEL	이미지를 생성한 저자에 대한 정보 및 라벨링을 할 수 있다. 최근에는 LABEL 을 많이 사용한다. MAINTAINER "이미지 생성한 사람 관련 정보" MAINTAINER WOO HYUNG CHOI whchoi98@gmail.com
RUN	패키지 인스톨 및 Shell 명령을 기본이미지를 실행 할때, 함께 수행할 수 있다. RUN "기본 이미지에서 스크립트 또는 명령을 실행시키는 명령어" RUN yum -y update RUN yum -y install net-tools
CMD	이미지를 통해 컨테이너가 만들어지고 나서 가장 처음 실행될 명령어를 선언 한다. 단 한번만 적용되기 때문에, 필요에 따라 지정해서 사용하면 된다. CMD "컨테이너에서 실행할 명령어를 실행시키는 명령" CMD touch /home/test.txt
EXPOSE	EXPOSE 는 docker run –expose 옵션과 동일하며 호스트와 연결될 포트 번호를 설정하는 명령이다. EXPOSE 호스트와 연결할 포트번호를 설정하는 명령 EXPOSE 8888 80
ENV	환경변수를 설정할 수 있다. Docker run -e 와 유사한 결과를 가질 수 있다. ENV "환경변수"

	ENV nginx_vhost /etc/nginx/sites-available/default
ADD	현재 폴더 내부의 파일을 이미지에 추가할 수 있으며 , 반드시 절대경로를 지정한다. ADD http://test.com/test.txt /home/test/ # Image 내부 /home/test 폴더에 test.txt 를 저장
COPY	ADD 와 유사하지만 URL, tar 에 대한 압축을 풀 수는 없다.
ENTRYPOINT	컨테이너가 실행될 때 사용할 명령어를 지정할 수 있다.
VOLUME	지정된 호스트 볼륨에 특정 디렉토리를 저장할 수 있도록 마운트 시킬 수 있다. VOLUME ["호스트 디렉토리", "이미지 내부 디렉토리"] VOLUME ["/home/whchoi", "/home/guest"]
USER	RUN, CMD, ENTRYPOINT 에서 실행될 계정을 선언한다. USER "User Name" USER admin
ONBUILD	ONBUILD 는 최초 이미지를 생성할 때는 동작하지 않으며, 최초에 생성된 이미지에 바인딩되어 이후 해당 이미지를 부모 이미지로 기반으로 자식 이미지들을 만들 때 자동으로 바인딩 된다. ONBUILD RUN touch /thisisremade.txt
WORKDIR	명령을 실행하게 되는 디렉토리를 설정하는 명령이다. 주로 RUN, CMD, ENTRYPOINT 에서 설정한 실행 파일이 실행될 디렉토리가 된다. WORKDIR "작업 디렉토리" WORKDIR "/home/whchoi"

ARG	커맨드로 docker image 를 빌드할때 설정 할 수 있는 옵션 들을 지정해준다. 예를 들어: ARG env ARG log_level=debug 위의 경우 docker build 커맨드로 빌드할때 --build-arg 옵션을 사용하여 env 와 log_level 값을 설정 해줄수 있다.
SHELL	디폴트로 지정되어 있는 shell 타입을 바꿀수 있게 해준다. Linux 의 default shell 은 ["/bin/sh", "-c"] 이다.