



# AWS 웹서버 구축하기

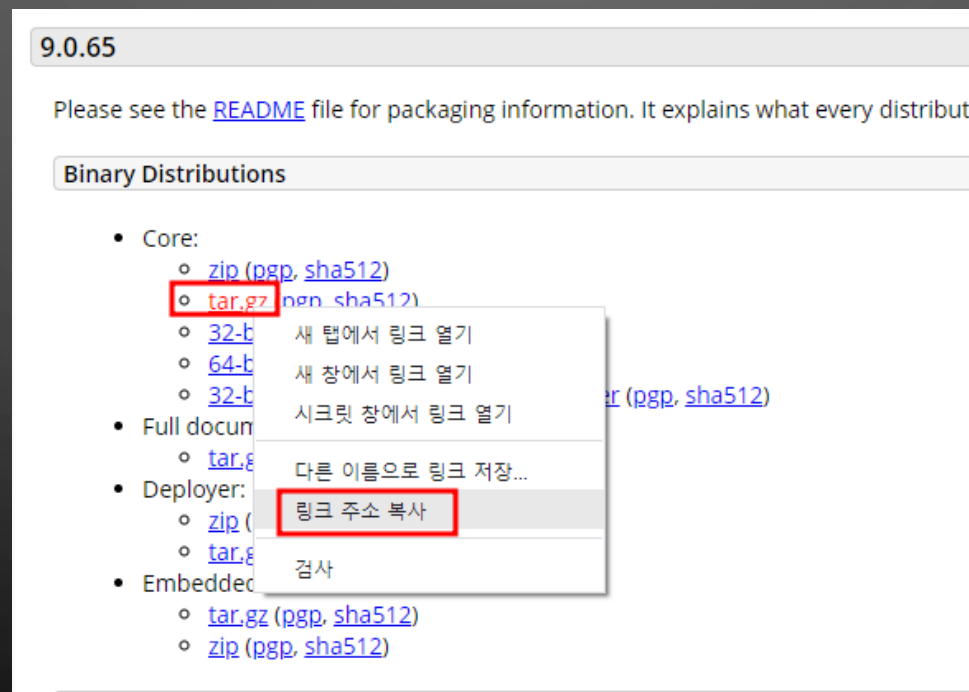
4강 - AWS 프로젝트 배포하기

LECTURED BY SOONGU HONG

AWS EC2에 프로젝트를 배포하기 위해 Tomcat을 설치합니다.

아래 링크로 이동하여 톰캣 다운로드 페이지로 가서 다운로드 주소를 복사합니다.

<https://tomcat.apache.org/download-90.cgi>



Putty로 EC2에 접속하여 아래 명령어를 입력하여  
톰캣을 설치합니다.

\$ wget [복사한 다운로드 링크]

```
[ec2-user@ip-172-31-35-93 ~]$ wget https://d1cdn.apache.org/tomcat/tomcat-9/v9.0.65/bin/apache-tomcat-9.0.65.tar.gz
--2022-09-02 10:52:00-- https://d1cdn.apache.org/tomcat/tomcat-9/v9.0.65/bin/ap
ache-tomcat-9.0.65.tar.gz
Resolving d1cdn.apache.org (d1cdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to d1cdn.apache.org (d1cdn.apache.org)|151.101.2.132|:443... connecte
d.
HTTP request sent, awaiting response... 200 OK
Length: 11593900 (11M) [application/x-gzip]
Saving to: 'apache-tomcat-9.0.65.tar.gz'

100%[=====>] 11,593,900  --.-K/s   in 0.06s

2022-09-02 10:52:00 (173 MB/s) - 'apache-tomcat-9.0.65.tar.gz' saved [11593900/1
1593900]
```

아래 명령어로 톰캣의 압축을 풀어줍니다.

\$ tar xvfz [압축파일명]

```
]$ tar xvfz apache-tomcat-9.0.65.tar.gz
```

```
[ec2-user@ip-172-31-35-93 ~]$ ls  
apache-tomcat-9.0.65  apache-tomcat-9.0.65.tar.gz
```

아래 명령어로 권한을 관리자로 변경한 후

```
$ sudo su
```

톰캣폴더를 만들고 해당 폴더로 압축을 푼 톰캣을 이동합니다.

```
$ mv apache-tomcat-9.0.65 /usr/local/tomcat
```

```
[root@ip-172-31-35-93 ec2-user]# mv apache-tomcat-9.0.65 /usr/local/tomcat
[root@ip-172-31-35-93 ec2-user]# cd /usr/local/tomcat/
[root@ip-172-31-35-93 tomcat]# ls
bin          CONTRIBUTING.md  logs          RELEASE-NOTES  webapps
BUILDING.txt lib              NOTICE       RUNNING.txt    work
conf         LICENSE          README.md    temp
```

톰캣 포트설정을 진행하기 위해

```
$ vi /usr/local/tomcat/server.xml
```

명령어를 통해 설정파일을 열어 포트번호를  
80으로 바꿔줍니다

Vi에디터에서는 i를 통해 입력하고  
ESC를 누르고 :wq! 명령어를 통해 빠져나옵니다.

```
-->  
<Connector port="80" protocol="HTTP/1.1"  
            connectionTimeout="20000"  
            redirectPort="8443" />  
  
!--  
<Connector executor="tomcatThreadPool"  
:wq!
```

톰캣이 설치되었으면 톰캣 서버를 실행시킵니다.

```
$ /usr/local/tomcat/bin/startup.sh
```

```
[root@ip-172-31-35-93 bin]# /usr/local/tomcat/bin/startup.sh
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /
Using CLASSPATH:       /usr/local/tomcat/bin/bootstrap.jar:/usr/local/
tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
```

EC2 대시보드에서 우리의 웹서버 주소를 확인합니다.

▼ 인스턴스

인스턴스

인스턴스 유형

시작 템플릿

스팟 요청

Savings Plans

예약 인스턴스

전용 호스트

용량 예약

퍼블릭 DNS(IPv4)

ec2-3-39-73-67.ap-northeast-2.compute.amazonaws.com

IPv4 퍼블릭 IP

■■■■

IPv6 IP

-

탄력적 IP

■■■■

가용 영역

ap-northeast-2c

보안 그룹

[gogugamgamja](#). [인바운드 규칙 보기](#). [아웃바운드 규칙 보기](#)

예약된 이벤트

예약된 이벤트 없음

AMI ID

amzn2-ami-kernel-5.10-hvm-2.0.20220805.0-x86\_64-gp2 (ami-01d87646ef)

서브넷 ID

subnet-034e7d176135f66c3




해당 주소를 브라우저에 입력해 우리 웹사이트에 접속합니다. 아래와 같은 화면이 뜨면 톰캣이 정상 실행된 것입니다.


ec2-3-39-73-67.ap-northeast-2.compute.amazonaws.com

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

## Apache Tomcat/9.0.65

 SOFTWARE FOUNDATION  
http://www.apache.org/

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 Recommended Reading:

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

Server Status  
Manager App  
Host Manager

### Developer Quick Start

- [Tomcat Setup](#)
- [First Web Application](#)
- [Realms & AAA](#)
- [JDBC Data Sources](#)
- [Examples](#)
- [Servlet Specifications](#)
- [Tomcat Versions](#)

#### Managing Tomcat

For security, access to the [manager webapp](#) is restricted. Users are defined in:

#### Documentation

- [Tomcat 9.0 Documentation](#)
- [Tomcat 9.0 Configuration](#)

#### Getting Help

[FAQ](#) and [Mailing Lists](#)

The following mailing lists are available:

본격적인 배포 준비를 위해 프로젝트설정을 변경합니다.  
먼저 DataSource설정에서 유저이름을 RDS 마스터 계정명  
jdbcUrl에 RDS 엔드포인트 이름을 적습니다.

```
@Bean
public DataSource dataSource() {

    HikariConfig config = new HikariConfig();

    config.setUsername("goguma");
    config.setPassword(" ");
    config.setJdbcUrl("jdbc:mariadb://goguma-dev-database.cbcszxd0fody.ap-northeast-2.rds.amazonaws.com:3306/spring4");
    config.setDriverClassName("org.mariadb.jdbc.Driver");

    return new HikariDataSource(config);
}
```

파일업로드 설정 변경을 위해 UploadController로  
이동합니다.

파일 루트 저장 경로를 아래와 같이 변경해주세요.

```
public class UploadController {  
  
    // 업로드 파일 저장 경로  
    4 usages  
    private static final String UPLOAD_PATH = "/home/ec2-user/sl_dev/upload";  
}
```

EC2에 접속해서 아래 경로대로 폴더를 생성해줍니다.

```
$ mkdir -p /home/ec2-user/sl_dev/upload
```

```
public class UploadController {
```

```
    // 업로드 파일 저장 경로
```

```
    4 usages
```

```
    private static final String UPLOAD_PATH = "/home/ec2-user/sl_dev/upload";
```

다음은 build.gradle 설정 변경입니다.  
스프링부트 내장톰캣과 충돌방지를 위해  
아래의 설정 코드를 추가합니다.

```
plugins {  
    id 'org.springframework.boot' version '2.7.1'  
    id 'io.spring.dependency-management' version '1.0.11.RELEASE'  
    id 'java'  
    id 'war'  
}  
  
bootWar.enabled = false; // bootWar와 충돌 방지  
war.enabled = true;  
  
group = 'com.project'  
version = '0.0.1-SNAPSHOT'  
sourceCompatibility = '11'  
  
configurations {  
    compileOnly {  
        extendsFrom annotationProcessor
```

Gradle이 빌드 실행시 테스트를 돌리는데  
우리가 만든 토이프로젝트는 테스트가 불완전할 수  
있으므로 테스트를 스킵하는 설정을 작성합니다.

그리고 깃허브에 push합니다.

```
}  
  
tasks.named('test') {  
    // useJUnitPlatform()  
    exclude '**/*'  
}
```

Putty로 EC2 인스턴스에 접속하여 git을 설치합니다.

```
$ sudo yum install git
```

Git 설치 후 톰캣의 webapps 폴더로 이동합니다.

```
$ cd /usr/local/tomcat/webapps
```

```
[root@ip-172-31-35-93 ~]# cd /usr/local/tomcat/webapps/  
[root@ip-172-31-35-93 webapps]# ls  
docs  examples  host-manager  manager  ROOT
```

이 안에 ROOT라는 이름의 폴더가 바로 톰캣이 실행시킬  
우리의 웹앱입니다.

기존 톰캣의 ROOT를 삭제합니다.

```
$ sudo rm -rf ROOT
```

```
[root@ip-172-31-35-93 webapps]# sudo rm -rf ROOT  
[root@ip-172-31-35-93 webapps]# ls  
docs  examples  host-manager  manager  
[root@ip-172-31-35-93 webapps]#
```



이제 github에서 우리의 프로젝트를 불러옵니다.

```
$ git clone [ git repository http 주소 ]
```

클론 후 해당 프로젝트 폴더로 이동합니다.

```
[root@ip-172-31-35-93 webapps]# ls
docs  examples  host-manager  manager  spring_webprj4
[root@ip-172-31-35-93 webapps]# cd spring_webprj4/
[root@ip-172-31-35-93 spring_webprj4]# ls
build.gradle  gradle  gradlew  gradlew.bat  settings.gradle  src
[root@ip-172-31-35-93 spring_webprj4]#
```

빌드프로그램 실행 권한을 부여 한 후 해당 프로젝트를 빌드합니다.

```
$ chmod 777 gradlew
```

```
$ sudo ./gradlew build
```

```
]# chmod 777 gradlew  
]# ./gradlew build
```

```
Starting a Gradle Daemon (subsequent builds will be faster)
```

```
BUILD SUCCESSFUL in 2m 46s  
5 actionable tasks: 5 executed
```

빌드가 완료되면 프로젝트 내부의 build/libs 폴더에  
War파일이 생성됩니다.

```
$ cd build/libs
```

```
[root@ip-172-31-35-93 spring_webprj4]# cd build/libs/  
[root@ip-172-31-35-93 libs]# ls  
web_prj-0.0.1-SNAPSHOT-plain.war  
[root@ip-172-31-35-93 libs]#
```

해당 war파일을 ROOT.war로 이름변경 후  
톰캣의 webapps폴더로 이동시킵니다.

```
$ sudo mv [현재 war] ROOT.war // 이름 변경
```

```
# sudo mv web_prj-0.0.1-SNAPSHOT-plain.war ROOT.war
```



```
[root@ip-172-31-35-93 libs]# ls  
ROOT.war
```



// webapps폴더 이동 후 ROOT.war 복사

\$ cd /usr/local/tomcat/webapps

\$ cp [ ROOT.war가 있는 경로 ] . // 맨뒤에 . 있음 주의



```
# cd /usr/local/tomcat/webapps/
```

```
# cp spring_webprj4/build/libs/ROOT.war .
```

배포 성공!!

EC2 아이피주소를 통해 접속

