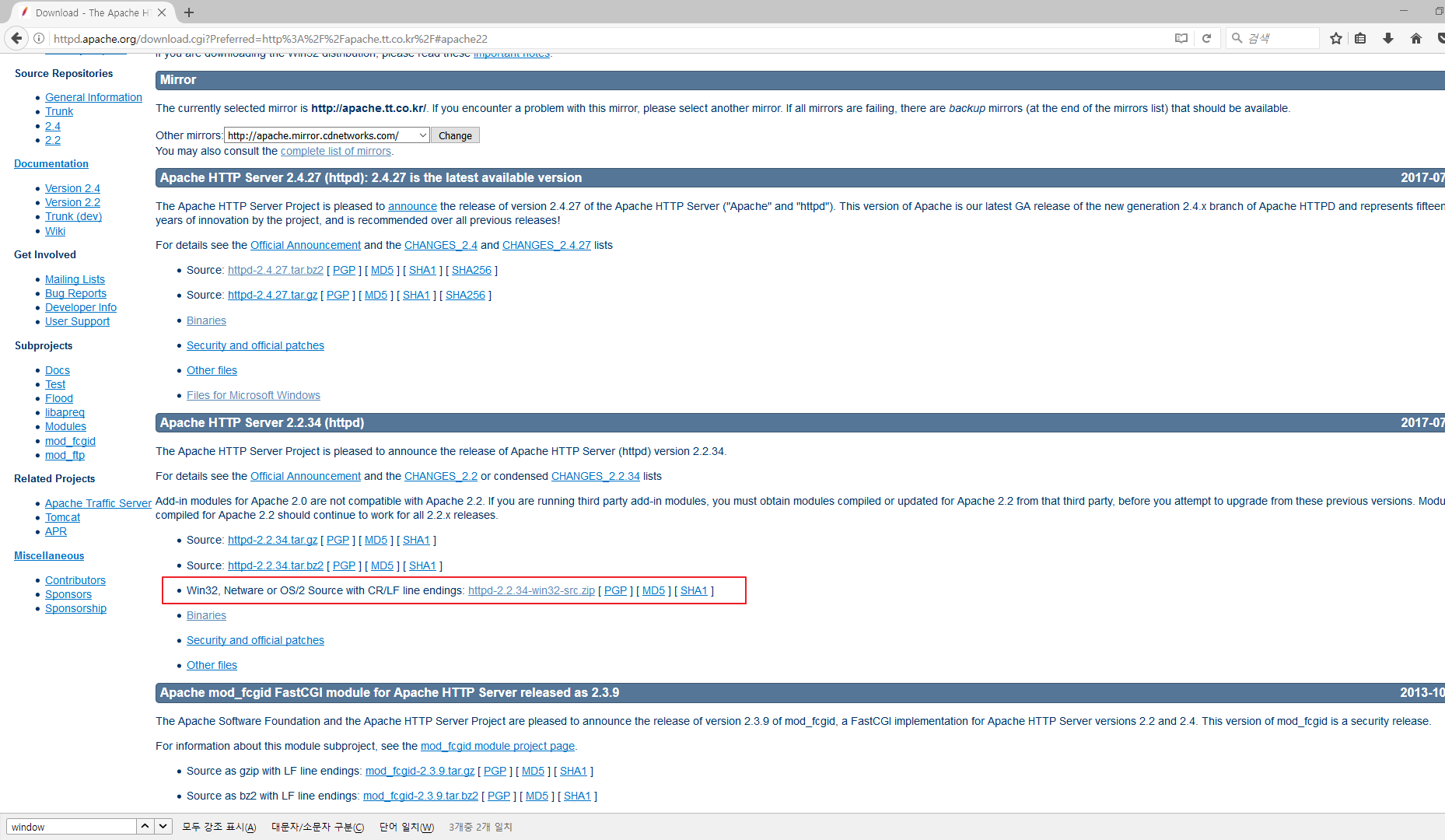
아파치설치

<http://httpd.apache.org/download.cgi?Preferred=http%3A%2F%2Fapache.tt.co.kr%2F#apache22>



<http://www.apachelounge.com/download/>

일단 본문내용은

<http://www.lynda.com/Apache-HTTP-Server-tutorials/Installing-Apache-HTTP-Server-24-Windows-New/77958/150487-4.html?vid=1>

여기있는 비디오 내용을 해석한거다

일단 아파치라운지라는 좋은곳이 있다. 공식홈페이지는 리눅스 위주로 최신버전을 제공하므로, 여기를 이용한다..

<http://www.apachelounge.com/download/>

**여기서 최신버전을 다운받는다**. 윈도우 XP, VISTA는 VC10 윈도우7은 VC11로 설치한다. 일단, 64비트 환경임에도 불구하고 32비트를 추천하고 있는데 이유를 모르겠다. 그러면 왜만들어..

어쨌든, 32비트 버전을 압축해제한 후에, c:\apache를 만들어 옮긴다. 물론 이건 니맘대로

***(선택)C:\Program Files\에 깔았을경우!***

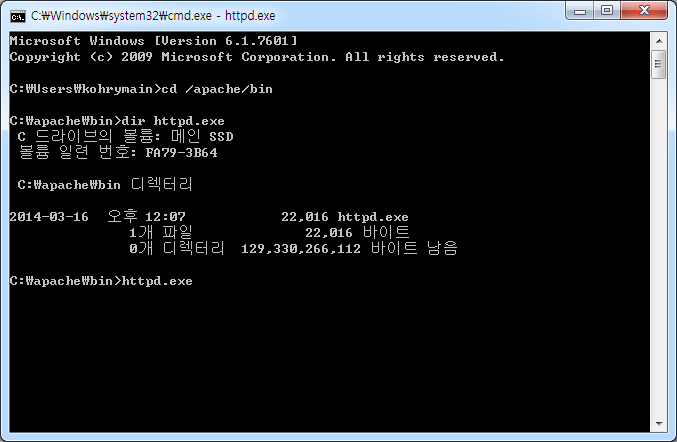
*만약에 이를 ProgramFiles폴더 밑에 설치했을 경우는, 수정이 직접 안되기때문에 밖에서 세이브하고 관리자 권한을 받아서 덮어쓴다. 위의 과정을 거쳤으면 이건 패스*

***(선택)c:\Apache24라는 이름 외에 다른 폴더에 깔았을경우!***

*어쨌든 폴더이름을 바꾸었으므로, c:\apache\conf\httpd.conf를 수정하면 되는데, c:/Apache24 를 찾아 모두바꾼다. 메모장에도 기능있다. 한글로 열고 깨지는 일은 없어야할거싱다.*

**자 이제 c:\apache\conf\httpd.conf 의 ServerName앞의 #의 주석을 풀고 localhost:80으로 수정해준다**. 포트번호는 니맘대로 해도 상관없는데 80포트번호가 http서버 포트이기 때문에 걍 하란대로 해라. 그리고 저장

이제 cmd가서 설정하기 시작한다



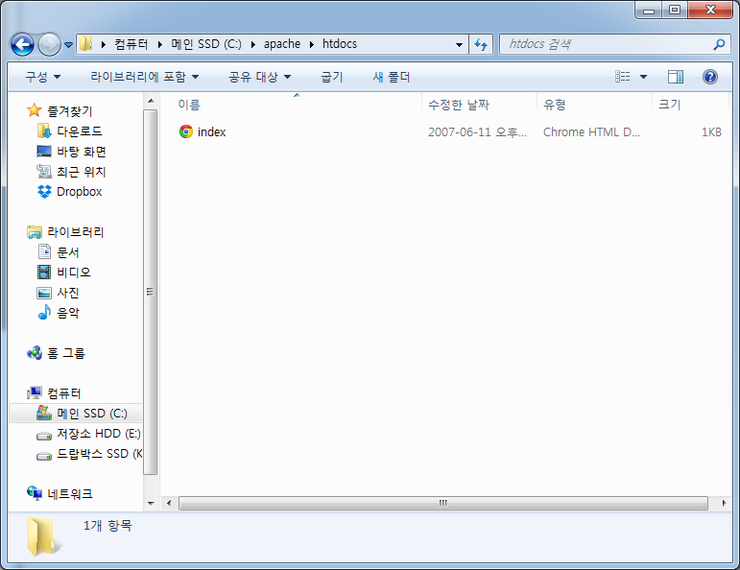
요로코롬 해준다.

방화벽설정이 나온다면 허용해주고,

이제 웹브라우저에 들어가서 localhost 또는 localhost:80 (설정한 주소와 소켓번호)

를 쳐주면 정상작동 하는것을 확인할수 있다.

It Works!가 어디 박혀있는가 하면 바로 여깄다.



출처: <http://gorakgarak.tistory.com/46> [먹거리 만드는 열정맨 고락가락]

## [아파치 톰캣 연동](http://gorakgarak.tistory.com/47)

[웹/PHP](http://gorakgarak.tistory.com/category/%EC%9B%B9/PHP) 2014.05.12 08:25

<http://blog.naver.com/freezyang?Redirect=Log&logNo=208526876>

인터넷에 나와있기는 한데, 따옴표도 빠지고 허접하다. 일단 여기가 깔끔하고 제일 잘 되어있는듯

일단 mod\_jk.so는 굉장히 버전에 취약하다. 따라서 버전에 맞는 거시기를 받는게 중요하다.

<http://www.apachelounge.com/>

여기의 mod\_jk.so가 제일 낫다

**Apache WebSever 설정**

**1. 커넥터 설치**

   - APACHE\_HOME/modules 디렉토리 아래에 커넥터(mod\_jk.so)파일을 추가.

**2. httpd.conf 설정**

   - APACHE\_HOME/conf/httpd.conf 파일을 수정.

|  |
| --- |
|  |

|  |
| --- |
| # 확인후 수정해야할 부분  ServerRoot "D:\Tools\Apache24"   <-- Apache가 설치된 경로(APACHE\_HOME)    DocumentRoot "D:\Tools\Apache24\htdocs"    <-- Apache가 설치된 경로아래 htdocs위치(APACHE\_HOME\htdocs)  LoadModule jk\_module modules/mod\_jk.so      <-- Tomcat Connector 로드    # 추가되는 부분  <IfModule mod\_jk.c>    JkWorkersFile conf/workers.properties    JkLogFile logs/mod\_jk.log              <-- 로그파일    JkLogLevel info    JkMountFile conf/uriworkermap.properties  </IfModule> |

**3. workers.properties 파일 생성**

  - APACHE\_HOME/conf 디렉토리 아래에 workers.properties 파일을 추가.

|  |
| --- |
| # 파일내용  worker.list=worker1 worker.worker1.port = 8009 worker.worker1.host = localhost worker.worker1.type = ajp13  worker.worker1.lbfactor=1 |

**4. uriworkermap.properties 파일 생성**

- APACHE\_HOME/conf 디렉토리 아래에 uriworkermap.properties 파일을 추가.

|  |
| --- |
| # 파일내용  /\*.do=worker1  /\*.jsp=worker1 |

출처: <http://gorakgarak.tistory.com/47> [먹거리 만드는 열정맨 고락가락]  
  
출처: <http://gorakgarak.tistory.com/47> [먹거리 만드는 열정맨 고락가락]

## 개요

Apache httpd web server 와 tomcat 을 연계하는 방법은 세 가지가 있다.

예전부터 많이 쓰던 방법은 [tomcat connector](http://tomcat.apache.org/connectors-doc)(mod\_jk)를 사용하는 방법이고 다른 하나는 mod\_proxy를 사용하여 reverse proxy 기능을 사용하는 방법, 마지막은 mod\_proxy\_ajp 를 사용하여 AJP Protocol을 reverse proxy 로 사용하는 방법이다.

mod\_proxy 가 mod\_jk 에 비해 설정이 간편하고 AJP 같은 특정 WAS 의존적인 프로토콜을 사용하지 않으므로  성능이 더 좋다고 하지만 mod\_jk 가 오랫동안 써왔고 친숙해서 mod\_jk 를 많이 사용하는 편인데 세 가지 방법 모두 정리해 본다.

## 관성의 법칙

## 톰캣 하나로 두 사이트를 띄우는 방법 본문

[**프로그램**](http://esckey.tistory.com/category/%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%A8)

### 톰캣 하나로 두 사이트를 띄우는 방법

관성의 법칙 박대감 2014.01.21 12:19

**1. 도메인으로 분류하는 방법**

TOMCAT\_HOME\conf\server.xml 을 열면 기본적으로 하나의 Service 엘리먼트가 있고  
그 하위에 Engine 엘리먼트가, 또 그 하위에 아래와 같은 하나의 Host 엘리먼트가 있다.

      <Host name="localhost"  appBase="webapps"  
            unpackWARs="true" autoDeploy="true"  
            xmlValidation="false" xmlNamespaceAware="false">  
      </Host>

아래와 같이 추가하려는 도메인으로 Host 엘리먼트를 하나 더 추가한다.

      <Host name="localhost"  appBase="webapps"  
            unpackWARs="true" autoDeploy="true"  
            xmlValidation="false" xmlNamespaceAware="false">  
      </Host>  
      <Host name="www.testdomain.com"  appBase="C:\testdomain"  
            unpackWARs="true" autoDeploy="true"  
            xmlValidation="false" xmlNamespaceAware="false">  
      </Host>

appBase는 이 컨텍스트의 물리적 영역이 되겠다.  
물론 실제의 물리적인 디렉터리도 아래와 같이 필요하다.

C:\testdomain\ROOT  
C:\testdomain\ROOT\WEB-INF

WEB-INF의 web.xml등은 TOMCAT\_HOME\webapps\ROOT\WEB-INF에서 복사한다.  
  
C:\testdomain\ROOT 디렉터리는 이 컨텍스트의 루트 디렉터리로 작동한다.

여기서 www.testdomain.com은 추가하려는 도메인이 되겠으며,  
실제로 도메인을 보유하지 않고 개발 PC에서 작업하는 경우,   
C:\Windows\system32\drivers\etc\hosts 파일을 열고  
마지막 줄에 다음을 추가한다.

127.0.0.1       www.testdomain.com

이제 웹브라우저를 열고 [www.testdomain.com:포트번호]에 접속하면 된다.

만일 C:\testdomain\ROOT 가 아닌 C:\testdomain 를 룰트 디렉터리로 사용하고자 하는 경우에는  
server.xml을 다음과 같이 작성한다.

      <Hostname="localhost"  appBase="webapps"

            unpackWARs="true" autoDeploy="true"

            xmlValidation="false" xmlNamespaceAware="false">

      </Host>

      <Host name="www.testdomain.com"  appBase="C:\testdomain"

            unpackWARs="true" autoDeploy="true"

            xmlValidation="false" xmlNamespaceAware="false">

<context path="" docBase="D:\Team\_ePro\StrutsPilot"></context>

      </Host>

**2. 포트 번호로 분류하는 방법**

톰캣에서도 MS의 IIS와 같이 포트 번호에 따라 호스팅하는 것이 가능하다.  
TOMCAT\_HOME\conf\server.xml 을 열고 아래와 같이 Service 엘리먼트를 추가한다.

  <Service name="testdomain">  
    <Connector port="8090" protocol="HTTP/1.1"   
               connectionTimeout="20000"   
               redirectPort="8443" />  
    <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />  
    <Engine name="testdomain" defaultHost="localhost">  
      <Realm className="org.apache.catalina.realm.UserDatabaseRealm"  
             resourceName="UserDatabase"/>  
      <Host name="localhost"  appBase="C:\testdomain"  
            unpackWARs="true" autoDeploy="true"  
            xmlValidation="false" xmlNamespaceAware="false">  
            <context path="" docBase="C:\testdomain"></context>  
      </Host>  
    </Engine>  
  </Service>

Service 엘리먼트의 name 속성은 추가하려는 컨텍스트의 이름이 되겠다.  
첫 Connector 엘리먼트의 port 속성은 사용하려는 포트 번호를 넣으면 된다.  
Host 엘리먼트와 context 엘리먼트, appBase에 대한 물리적 경로 생성은 1항의 설명과 같다.

이제 웹브라우저를 열고 [localhost:포트번호]에 접속하면 된다.

hosts

127.0.0.1 cms.geumcheon.com

127.0.0.1 api.geumcheon.com

**pache HTTP 서버와 Tomcat 서버의 연동**

Web Server인 Apache와 WAS 서버인 Tomcat 을 연동하는 방법에 대해 포스팅하겠습니다.

저는 Windows 7를 사용하였고, Apache는 2.2 버전, 톰캣은 7.0.8 버전을 사용하였습니다 (그리 중요하지 않습니다)

연동하기 앞서, 간단한 개념부터 살펴보도록 하겠습니다.

**Apache, Tomcat 연동의 이유**

Tomcat은 WAS 서버이지만 Web 서버의 기능도 갖추고 있는 WAS 서버입니다.

그러나 톰캣의 Web 서버 기능은 아파치보다 느린 속도처리를 보였고, 이로 인해

정적인 페이지는 Apache가 처리하고, 동적인 페이지를 Tomcat이 처리함으로써 부하를 분산하는 이유에서 Apache와 Tomcat을 연동하였습니다.

그러나 이는 옛날 얘기이고.. 지금은 Tomcat이 많이 발전해 Tomcat 내의 Web 서버가 아파치에 절대 뒤쳐지지

않을만큼의 역할을 수행합니다.

그럼에도 불구하고 아직도 Apache와 Tomcat을 연동하여 사용하는 이유는, 아파치 내에서만 설정할 수 있는 부분이라던가

아파치에서 제공하는 유용한 모듈들을 톰캣에서 사용할 수 없기 때문입니다.

**AJP, mod\_jk**

아파치와 톰캣이 연동하기 위해선**AJP**를 통해 서로 통신을 하여야 합니다.

AJP란 아파치가 웹서버와 외부 서비스(톰캣 등)과 연동하기 위해 정한 규약(프로토콜) 입니다. 현재 1.3까지 나와있습니다.

아파치는 이를 사용하여 80포트로 들어오는 요청은 자신이 받고, 이 요청중 서블릿을 필요로 하는 요청은 톰캣에 접속하여 처리합니다.

우리는 아파치 톰캣 연동을 위해 **mod\_jk** 라는 모듈을 사용할건데, 이는 AJP 프로토콜을 사용하여 톰캣과 연동하기 위해 만들어진 모듈입니다.

mod\_jk는 톰캣의 일부로 배포되지만, 아파치 웹서버에 설치하여야 합니다.

**처리 플로우**

1)아파치 웹서버의 httpd.conf 에 톰캣 연동을 위한 설정을 추가하고 톰캣에서 처리할 요청을 지정함.

2)사용자의 브라우저는 아파치 웹서버(보통 80포트)에 접속해 요청

3)아파치 웹서버는 사용자의 요청이 톰캣에서 처리하도록 지정된 요청인지 확인. 요청을 톰캣에서 처리해야 하는 경우

  아파치 웹서버는 톰캣의 AJP포트(보통 8009포트)에 접속해 요청을 전달

4)톰캣은 아파치 웹서버로부터 요청을 받아 처리한 후, 처리 결과를 아파치 웹서버에 되돌려줌

5)아파치 웹서버는 톰캣으로부터 받은 처리 결과를 사용자에게 전송

이제 필요한 mod\_jk 모듈을 다운받으러 가실까요.

톰캣 홈페이지의 tomcat-connector 부분에서 구할 수 있습니다.

<http://archive.apache.org/dist/tomcat/tomcat-connectors/jk/binaries/windows/>

여기서도 구하실 수 있습니다.(참고로 <http://archive.apache.org> 는 아파치 관련 파일들이 많이 있으니 자주 애용하시면 좋습니다.)

근데 위의 경로에서는 Apache 2.2 기준으로 64비트 관련한 모듈이 없더군요 ㅠ.ㅠ

그래서 경로를 하나 더 올리겠습니다. ㅠㅠ

<http://www.apachelounge.com/download/win64/>

여기서는 아파치 2.2, 2.4 버전과 그에 해당하는 모듈들을 전부 다운받을 수 있습니다. 유용한 사이트입니다.

이렇게 경로를 많이 올리는 이유는 **자신의 아파치 버젼과, 비트와, 컴파일 버전(VC10, VC11 등) 에 따라 각각**

**mod\_jk.so** **파일이 다 다르기 때문입니다.**. 자신의 아파치 버전에 맞지 않는 mod\_jk.so 파일을 로딩할 경우 아파치가 실행되지 않으니

잘 찾아보시고 자신에게 맞는 mod\_jk 모듈을 다운받으시기 바랍니다.

파일을 다운로드 받으셨으면 압축을 풀고 **mod\_jk.so** 파일을 Apache 폴더 내 modules 폴더안에 복사해 넣습니다.

이제 설정파일을 수정해 보겠습니다.

톰캣을 건드리지 않으셨다면 AJP 포트는 8009번일테지만, 확인한번 해보고 가는것도 나쁘지 않지요~

톰캣의 server.xml 파일을 열어서 아래의 구문이 있는가 확인해 줍시다.

**server.xml**

<!-- Define an AJP 1.3 Connector on port 8009 -->

**<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />**

주석처리 되어있으면 해제해주시고.. 확인되었으니 이제 톰캣에서는 건드릴 것이 없습니다.

아파치 쪽 설정으로 넘어가죠.

먼저, 연동할 톰캣의 목록을 적어줄 **workers.properties** 파일을 하나 만듭니다.

그리고 아래와 같이 작성해줍니다.

**workers.properties**

worker.list=worker1

worker.worker1.type=ajp13

worker.worker1.host=localhost

worker.worker1.port=8009

worker.list는 사용할 woker(톰캣) 인스턴스들의 리스트입니다. 이름은 임의로 지정해주시면 되고,

아래에서 **worker.woker이름.속성=값** 의 형태로 정의해 주시면 됩니다.

저희는 ajp13 프로토콜을 사용하며, host는 localhost이며, 포트는 8009(아까 확인하셨죠?)를 사용합니다.

그리고 만약 톰캣을 여러개 사용할 경우 아래와 같이 해주시면 됩니다.

**workers.properties**

worker.list=worker1,worker2 // 이름은 임의로 설정

worker.worker1.type=ajp13

worker.worker1.host=localhost

worker.worker1.port=8009

#worker.worker1.lbfactor=1

worker.worker2.type=ajp13

worker.worker2.host=localhost

worker.worker2.port=8010 // 포트만 바꿔주시면 됩니다. 사용할 두번째 톰캣의 server.xml 파일을 수정하셔야 됩니다.(위에서 보여드린 부분)

#worker.worker2.lbfactor=2

여기서 주석처리되었지만 추가된 부분이 있는데, **worker.worker명.lbfactor** 라는 구문입니다.

이는 작업할당량 비율이라고 보시면됩니다. worker1:worker2 = 1:2 의 비율로 작업을 처리한다고 보시면 됩니다.

이제 연동할 톰캣의 정보를 가진 properties 파일을 생성하였으니, 아파치가 실행할 때 참조하는 **httpd.conf** 파일에 이를 명시해줘야합니다.

**httpd.conf**

LoadModule jk\_module modules/mod\_jk.so    // 모듈 위치

JkWorkersFile conf/workers.properties    // workers.properties 파일 위치

JkLogFile logs/mod\_jk.log    // 로그파일 위치

JkLogLevel info    // 로그 레벨

JkMount /\* worker1

**LoadModule** 구문을 추가하여 아까 추가한 **mod\_jk.so** 파일의 위치를 명시하고,

**JkWorkerFile** 구문으로 **worker 프로퍼티 파일의 위치**를 적어줍니다.

log 관련 부분은 로그파일의 위치와 로그레벨을 명시합니다. (로그의 형태도 지정할 수 있습니다.)

이외에도 구문을 추가하여 설정하실 수 있으니 구글신에게 물어보시면 됩니다.

그리고 마지막으로 중요한 **JkMount** 구문입니다.

JkMount 뒤에 오는 **/\*** 는 모든 url의 요청을 의미합니다.

**즉, 모든 url의 요청에서 서블릿 관련 처리가 필요하다면 workers.properties 파일에 명시된 worker1로 넘기겠다는 의미가 됩니다.**

/\* 외에도 /\*.jsp, /\*.php 등등 적어주실 수 있습니다. 그리고 톰캣을 여러대 추가하셨다면 worker1 대신에

worker2, worker3 등을 작성해 다른 톰캣으로 넘기셔도 되지요.

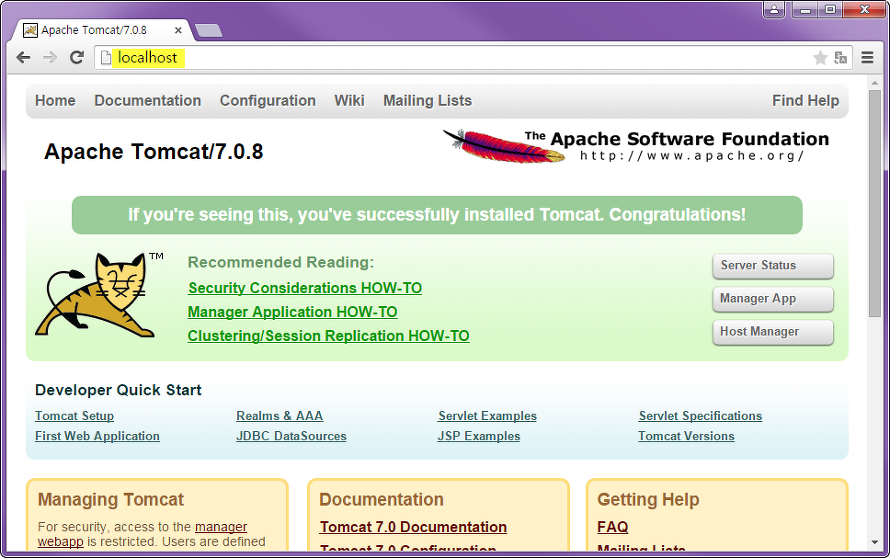
끝났습니다..

설정은 굉장히 간단합니다. 이제 확인한번 해보도록 하죠.

아파치의 포트는 80, 톰캣의 포트는 8080입니다. 두 서버를 연동하였기 때문에

**http://localhost 를 입력하였을때 톰캣 웹루트의 index.jsp가 떠야 합니다.**

아파치 띄우시고, 톰캣 띄우시고 주소창에 localhost 입력해보도록 하죠.



연동이 잘 되네요~~ 수고하셨습니다.

아, 그리고 참고로

httpd.conf에 VirtualHost 설정이 되어있으니 연동이 되지 않더군요..

VirtualHost 구문을 해제해주세요.

이상 포스팅을 마치겠습니다. 감사합니다.

출처: <http://joont.tistory.com/55> [Toward the Developer]  
  
출처: <http://joont.tistory.com/55> [Toward the Developer]  
  
출처: <http://joont.tistory.com/55> [Toward the Developer]

# Apache + Tomcat + mod\_jk 설정 후 AJP 연결하는 서버 환경 설정

By [webterror](http://webterror.net/author/webterror/) | 2015년 2월 9일

[0 Comments](http://webterror.net/apache-tomcat-mod_jk-%ec%84%a4%ec%a0%95%ed%9b%84-ajp%ec%97%b0%ea%b2%b0%ed%95%98%eb%8a%94-%ec%84%9c%eb%b2%84-%ed%99%98%ea%b2%bd-%ec%84%a4%ec%a0%95%ed%95%98%ea%b8%b0/#respond)

### 아파치 + 톰캣 설정하는 방법

#### Apache와 Tomcat을 연동하는 이유

Tomcat 서버는 본연의 임무인 서블릿 컨테이너의 역할만 하고, Apache HTTP Server는 웹서버의 역할을 하도록 각각의 기능을 분리하기 위해 연동을 할 수 있다. Apache HTTP Server에서 제공하는 편리한 기능을 사용하기 위해서 연동을 할수 있다. 대규모 사용자가 사용하는 시스템을 구축할 때 웹 서버인 아파치와 연동을 하면 부하 분산의 효과를 가질 수 있다. mod\_jk의 Load Balancing과 FailOver기능을 사용하여 안정적으로 운영 할 수 있다.

#### Apache와 Tomcat 연동하기

@@ mod\_jk 다운로드 및 설정  
http://www.apache.org/dist/tomcat/tomcat­connectors/jk/binaries/windows/  
에서 버전에 맞는 파일을 다운 받아 $APACHE\_HOME/modules 디렉토리에 복사한다.  
Apache HTTP 서버 2.2.x 버전을 설치하였다면 tomcat­connectors­1.2.37­windows­i386­httpd­2.2.x.zip 파일을 다운 받는다.  
httpd.conf 파일의 LoadModule 부분에 mod\_jk Module을 추가한다.

@@ 우분투에서 다음의 커맨드로 설치가 가능하다.



|  |  |
| --- | --- |
| 1 | sudo apt-get install libapache2-mod-jk |

@@ httpd.conf 파일  
  
# jk\_module 추가  
LoadModule jk\_module modules/mod\_jk.so

@@ 우분투에서 활성화, 그러나 패키지를 설치하면 자동으로 enable 되어 있을 것이다.



|  |  |
| --- | --- |
| 1 | sudo a2enmod |

#### workers.properties 파일 설정

Apache와 tomcat를 연동하기위해서는 workers.properties 파일을 설정해야 한다.  
$APACHE\_HOME/conf/workers.properties 파일을 아래 예제와 같이 생성한다.  
workers.properties 파일은 일반적으로 httpd.conf 파일과 같은 디렉토리에 위치하게 설정한다.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | worker.list=sample  # 톰캣 server.xml 파일 AJP/1.3 Connector의 Port를 입력한다.  worker.sample.port=8009  # 톰캣 server 호스트  worker.sample.host=localhost  # 아파치 + 톰캣 통신 프로토콜  worker.sample.type=ajp13 |

%% 참고  
톰캣 워커는(Tomcat worker) 웹서버로부터의 서블릿 요청을 톰캣 프로세스(Worker)에게 전달하여 요청을 처리하는 톰캣 인스턴스이다.  
대부분 하나의 worker를 사용하나, load 밸런싱이나 site 파티셔닝을 위해 여러개의 worker를 사용 할 수 있다.  
워커 타입에는 ajp12, ajp13, jni, lb 등이 있다.

#### workers.properties 경로 지정

httpd.conf 파일에 workers.properties 파일 경로를 지정한다.

@@ httpd.conf 파일



|  |  |
| --- | --- |
| 1  2 | # workers.properties 파일 추가  JkWorkersFile conf/workers.properties |

#### VirtualHost 설정 변경

@@ $APACHE\_HOME/conf/vhosts/extra/httpd­vhost.conf 파일의 VirtualHost의 DocumentRoot를 Tomcat 디렉토리로 변경하자

@@ JkMount 설정을 추가하자 httpd­-vhost.conf 파일 설정



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | NameVirtualHost \*:80    ServerName test.apache.org  DocumentRoot C:\dev\apache­tomcat­6.0.32webapps\ROOT    # URL중 jsp로 오는 Request만 Tomcat에서 처리 함  # sample은 workers.properties에서 등록한 worker이름 JkMount/\*.jsp sample  # servlet 예제 실행을 위해서 추가 JkMount /examples/\* sample |

#### Tomcat의 server.xml 수정

태그의 docBase 디렉토리를 Apache HTTP Server 설정과 동일하게 Tomcat 서버의 webapps/ROOT 디렉토리를 절대경로로 지정하자.  
원하는 디렉토리를 Document Root로 사용하면 된다.

### 우분투에서 mod-jk 설정



|  |  |
| --- | --- |
| 1  2 | $ apt-get install libapache2-mod-jk  $ a2enmod jk |

@@ jk.conf 파일 내용 확인하기



|  |  |
| --- | --- |
| 1  2 | cat /etc/apache2/mods-enabled/jk.conf  vi /etc/libapache2-mod-jk/workers.properties |

@@ Jsp Application이 설치된 곳을 가르키도록 설정



|  |  |
| --- | --- |
| 1  2  3  4 | vi /etc/apache2/sites-available/default    DocumentRoot /var/lib/tomcat7/webapps/ROOT/  JkMount /\*.jsp ajp13 |

@@ server.xml에서 인식하고 있는 HOST 설정.

아래처럼 appBase 경로 아래에는 ROOT가 있어야 한다.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | root@mainframe:/home/ryu/public\_html/root# tree  .  ├── build  ├── index.html  ├── logs  │   ├── log4j.log  │   └── log4j.log.1  ├── mlink\_erq.zip  ├── ROOT  │   ├── board  │   │   ├── board\_master\_edit.jsp  │   │   ├── board\_master\_insert\_success.jsp  │   │   ├── board\_master\_list.jsp  │   │   ├── board\_master\_success.jsp  │   │   └── success.jsp  │   ├── css  │   │   └── CVS  │   │   ├── Entries  │   │   ├── Repository  │   │   └── Root |



|  |  |
| --- | --- |
| 1 | vi /etc/tomcat7/server.xml |

우선적으로 커넥터가 2개가 있는데 다음과 같이 구분해서 생각해야 한다.

1. 톰캣에서 HTTP 프로토콜로 연결해 주는 서블릿 커넥터, 따라서 아래의 8080 포트로 웹페이지가 제대로 나와야 한다.  
2. AJP 프로토콜, 즉 아파치에서 /\*.jsp로의 요청이 왔을때에는 무조건 ajp13 워크로 톰캣에게 던진다. 따라서 DocumentRoot /var/lib/tomcat7/webapps/ROOT/ 해당 위치와는 별개로 생각해서 세팅해야 한다.

@@ 로그확인하기



|  |  |
| --- | --- |
| 1 | tail -f /var/log/apache2/mod\_jk.log |

#### 우분투에서 톰캣 환경 보는 방법



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | $ /usr/share/tomcat7/bin/version.sh  Using CATALINA\_BASE: /usr/share/tomcat7  Using CATALINA\_HOME: /usr/share/tomcat7  Using CATALINA\_TMPDIR: /usr/share/tomcat7/temp  Using JRE\_HOME: /usr  Using CLASSPATH: /usr/share/tomcat7/bin/bootstrap.jar:/usr/share/tomcat7/bin/tomcat-juli.jar  Server version: Apache Tomcat/7.0.35  Server built: May 24 2013 09:52:20  Server number: 7.0.35.0  OS Name: Linux  OS Version: 3.8.0-19-generic  Architecture: i386  JVM Version: 1.7.0\_51-b00  JVM Vendor: Oracle Corporation |

<http://netframework.tistory.com/entry/tomcat-httpd-%EC%84%9C%EB%B2%84-%EC%97%B0%EA%B2%B0-%EB%B0%A9%EB%B2%95>

톰캣 포트 다르게 올리기 server.xml

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <!--  Licensed to the Apache Software Foundation (ASF) under one or more  contributor license agreements. See the NOTICE file distributed with  this work for additional information regarding copyright ownership.  The ASF licenses this file to You under the Apache License, Version 2.0  (the "License"); you may not use this file except in compliance with  the License. You may obtain a copy of the License at  http://www.apache.org/licenses/LICENSE-2.0  Unless required by applicable law or agreed to in writing, software  distributed under the License is distributed on an "AS IS" BASIS,  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  See the License for the specific language governing permissions and  limitations under the License.  -->  <!-- Note: A "Server" is not itself a "Container", so you may not  define subcomponents such as "Valves" at this level.  Documentation at /docs/config/server.html  -->  <Server port="8005" shutdown="SHUTDOWN">  <Listener className="org.apache.catalina.startup.VersionLoggerListener" />  <!-- Security listener. Documentation at /docs/config/listeners.html  <Listener className="org.apache.catalina.security.SecurityListener" />  -->  <!--APR library loader. Documentation at /docs/apr.html -->  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />  <!-- Prevent memory leaks due to use of particular java/javax APIs-->  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />  <!-- Global JNDI resources  Documentation at /docs/jndi-resources-howto.html  -->  <GlobalNamingResources>  <!-- Editable user database that can also be used by  UserDatabaseRealm to authenticate users  -->  <Resource name="UserDatabase" auth="Container"  type="org.apache.catalina.UserDatabase"  description="User database that can be updated and saved"  factory="org.apache.catalina.users.MemoryUserDatabaseFactory"  pathname="conf/tomcat-users.xml" />  </GlobalNamingResources>  <!-- A "Service" is a collection of one or more "Connectors" that share  a single "Container" Note: A "Service" is not itself a "Container",  so you may not define subcomponents such as "Valves" at this level.  Documentation at /docs/config/service.html  -->  <Service name="geumcheon-cms-server">  <Connector port="8080" protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8443"  URIEncoding="UTF-8"  />  <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />  <Engine name="geumcheon-cms-engine" defaultHost="localhost">  <Realm className="org.apache.catalina.realm.LockOutRealm">  <Realm className="org.apache.catalina.realm.UserDatabaseRealm"  resourceName="UserDatabase"/>  </Realm>  <Host name="localhost" appBase="webapps/geumcheon-cms"  unpackWARs="true" autoDeploy="false" deployOnStartup="false">  <Context path="" docBase="cms.war" reloadable="true" >  </Context>  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"  prefix="localhost\_access\_log" suffix=".txt"  pattern="%h %l %u %t &quot;%r&quot; %s %b" />  </Host>  </Engine>  </Service>        <Service name="geumcheon-api-server">  <Connector port="8081" protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8444"  URIEncoding="UTF-8"  />  <Connector port="8010" protocol="AJP/1.3" redirectPort="8443" />  <Engine name="geumcheon-api-engine" defaultHost="localhost">  <Realm className="org.apache.catalina.realm.LockOutRealm">  <Realm className="org.apache.catalina.realm.UserDatabaseRealm"  resourceName="UserDatabase"/>  </Realm>  <Host name="localhost" appBase="webapps/geumcheon-api"  unpackWARs="true" autoDeploy="false" deployOnStartup="false">  <Context path="" docBase="api.war" reloadable="true" >  </Context>  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"  prefix="localhost\_access\_log" suffix=".txt"  pattern="%h %l %u %t &quot;%r&quot; %s %b" />  </Host>  </Engine>  </Service>        </Server> |

톰캣 도메인으로 같이 올리기

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <!--  Licensed to the Apache Software Foundation (ASF) under one or more  contributor license agreements. See the NOTICE file distributed with  this work for additional information regarding copyright ownership.  The ASF licenses this file to You under the Apache License, Version 2.0  (the "License"); you may not use this file except in compliance with  the License. You may obtain a copy of the License at  http://www.apache.org/licenses/LICENSE-2.0  Unless required by applicable law or agreed to in writing, software  distributed under the License is distributed on an "AS IS" BASIS,  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  See the License for the specific language governing permissions and  limitations under the License.  -->  <!-- Note: A "Server" is not itself a "Container", so you may not  define subcomponents such as "Valves" at this level.  Documentation at /docs/config/server.html  -->  <Server port="8005" shutdown="SHUTDOWN">  <Listener className="org.apache.catalina.startup.VersionLoggerListener" />  <!-- Security listener. Documentation at /docs/config/listeners.html  <Listener className="org.apache.catalina.security.SecurityListener" />  -->  <!--APR library loader. Documentation at /docs/apr.html -->  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />  <!-- Prevent memory leaks due to use of particular java/javax APIs-->  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />  <!-- Global JNDI resources  Documentation at /docs/jndi-resources-howto.html  -->  <GlobalNamingResources>  <!-- Editable user database that can also be used by  UserDatabaseRealm to authenticate users  -->  <Resource name="UserDatabase" auth="Container"  type="org.apache.catalina.UserDatabase"  description="User database that can be updated and saved"  factory="org.apache.catalina.users.MemoryUserDatabaseFactory"  pathname="conf/tomcat-users.xml" />  </GlobalNamingResources>  <!-- A "Service" is a collection of one or more "Connectors" that share  a single "Container" Note: A "Service" is not itself a "Container",  so you may not define subcomponents such as "Valves" at this level.  Documentation at /docs/config/service.html  -->  <Service name="Catalina">  <!--The connectors can use a shared executor, you can define one or more named thread pools-->  <!--  <Executor name="tomcatThreadPool" namePrefix="catalina-exec-"  maxThreads="150" minSpareThreads="4"/>  -->  <!-- A "Connector" represents an endpoint by which requests are received  and responses are returned. Documentation at :  Java HTTP Connector: /docs/config/http.html  Java AJP Connector: /docs/config/ajp.html  APR (HTTP/AJP) Connector: /docs/apr.html  Define a non-SSL/TLS HTTP/1.1 Connector on port 8080  -->  <Connector port="8080" protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8443"  URIEncoding="UTF-8"  />  <!-- A "Connector" using the shared thread pool-->  <!--  <Connector executor="tomcatThreadPool"  port="8080" protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8443" />  -->  <!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443  This connector uses the NIO implementation. The default  SSLImplementation will depend on the presence of the APR/native  library and the useOpenSSL attribute of the  AprLifecycleListener.  Either JSSE or OpenSSL style configuration may be used regardless of  the SSLImplementation selected. JSSE style configuration is used below.  -->  <!--  <Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"  maxThreads="150" SSLEnabled="true">  <SSLHostConfig>  <Certificate certificateKeystoreFile="conf/localhost-rsa.jks"  type="RSA" />  </SSLHostConfig>  </Connector>  -->  <!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443 with HTTP/2  This connector uses the APR/native implementation which always uses  OpenSSL for TLS.  Either JSSE or OpenSSL style configuration may be used. OpenSSL style  configuration is used below.  -->  <!--  <Connector port="8443" protocol="org.apache.coyote.http11.Http11AprProtocol"  maxThreads="150" SSLEnabled="true" >  <UpgradeProtocol className="org.apache.coyote.http2.Http2Protocol" />  <SSLHostConfig>  <Certificate certificateKeyFile="conf/localhost-rsa-key.pem"  certificateFile="conf/localhost-rsa-cert.pem"  certificateChainFile="conf/localhost-rsa-chain.pem"  type="RSA" />  </SSLHostConfig>  </Connector>  -->  <!-- Define an AJP 1.3 Connector on port 8009 -->  <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />  <!-- An Engine represents the entry point (within Catalina) that processes  every request. The Engine implementation for Tomcat stand alone  analyzes the HTTP headers included with the request, and passes them  on to the appropriate Host (virtual host).  Documentation at /docs/config/engine.html -->  <!-- You should set jvmRoute to support load-balancing via AJP ie :  <Engine name="Catalina" defaultHost="localhost" jvmRoute="jvm1">  -->  <Engine name="Catalina" defaultHost="localhost">  <!--For clustering, please take a look at documentation at:  /docs/cluster-howto.html (simple how to)  /docs/config/cluster.html (reference documentation) -->  <!--  <Cluster className="org.apache.catalina.ha.tcp.SimpleTcpCluster"/>  -->  <!-- Use the LockOutRealm to prevent attempts to guess user passwords  via a brute-force attack -->  <Realm className="org.apache.catalina.realm.LockOutRealm">  <!-- This Realm uses the UserDatabase configured in the global JNDI  resources under the key "UserDatabase". Any edits  that are performed against this UserDatabase are immediately  available for use by the Realm. -->  <Realm className="org.apache.catalina.realm.UserDatabaseRealm"  resourceName="UserDatabase"/>  </Realm>  <Host name="cms.geumcheon.com" appBase="webapps/geumcheon-cms"  unpackWARs="true" autoDeploy="false" deployOnStartup="false">  <Context path="" docBase="cms.war" reloadable="true" >  </Context>  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"  prefix="localhost\_access\_log" suffix=".txt"  pattern="%h %l %u %t &quot;%r&quot; %s %b" />  </Host>      <Host name="api.geumcheon.com" appBase="webapps/geumcheon-api"  unpackWARs="true" autoDeploy="false" deployOnStartup="false">  <Context path="" docBase="api.war" reloadable="true" >  </Context>  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"  prefix="localhost\_access\_log" suffix=".txt"  pattern="%h %l %u %t &quot;%r&quot; %s %b" />  </Host>  </Engine>  </Service>  </Server> |

아파치에서 톰캣연동하기. (mod\_jk사용)

|  |
| --- |
| 아파치 conf\httpd.conf  # workers.properties 파일 위치  **JkWorkersFile conf/workers.properties**  #로그파일 위치  **JkLogFile logs/mod\_jk.log**  #로그 레벨  **JkLogLevel info**  #넘길것  #JkMount /\* /\*  **LoadModule jk\_module modules/mod\_jk.so**  # Virtual hosts  **Include conf/extra/httpd-vhosts.conf** |

|  |
| --- |
| 아파치 conf\workers.properties  worker.list=cms,api  # 톰캣 server.xml 파일 AJP/1.3 Connector의 Port를 입력한다.  worker.cms.port=8009  # 톰캣 server 호스트  worker.cms.host=localhost  # 아파치 + 톰캣 통신 프로토콜  worker.cms.type=ajp13  worker.api.port=8010  # 톰캣 server 호스트  worker.api.host=localhost  # 아파치 + 톰캣 통신 프로토콜  worker.api.type=ajp13 |

|  |
| --- |
| 아파치 conf\extra\httpd-vhosts.conf  <VirtualHost \*:80>  ServerName cms.geumcheon.com  DocumentRoot D:\omnicns\install\apache-tomcat-8.5.12\webapps\geumcheon-cms\ROOT  JkMount /\* cms  </VirtualHost>  <VirtualHost \*:80>  ServerName api.geumcheon.com  DocumentRoot D:\omnicns\install\apache-tomcat-8.5.12\webapps\geumcheon-api\ROOT  JkMount /\* api  </VirtualHost> |

|  |
| --- |
| 아파치 ${APACHE\_HOME}\modules  mod\_jk.so 넣어놓기  **http://www.apachelounge.com/download/** |

|  |
| --- |
| 톰캣설정 conf\server.xml  <?xml version="1.0" encoding="UTF-8"?>  <!--  Licensed to the Apache Software Foundation (ASF) under one or more  contributor license agreements. See the NOTICE file distributed with  this work for additional information regarding copyright ownership.  The ASF licenses this file to You under the Apache License, Version 2.0  (the "License"); you may not use this file except in compliance with  the License. You may obtain a copy of the License at  http://www.apache.org/licenses/LICENSE-2.0  Unless required by applicable law or agreed to in writing, software  distributed under the License is distributed on an "AS IS" BASIS,  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  See the License for the specific language governing permissions and  limitations under the License.  -->  <!-- Note: A "Server" is not itself a "Container", so you may not  define subcomponents such as "Valves" at this level.  Documentation at /docs/config/server.html  -->  <Server port="8005" shutdown="SHUTDOWN">  <Listener className="org.apache.catalina.startup.VersionLoggerListener" />  <!-- Security listener. Documentation at /docs/config/listeners.html  <Listener className="org.apache.catalina.security.SecurityListener" />  -->  <!--APR library loader. Documentation at /docs/apr.html -->  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />  <!-- Prevent memory leaks due to use of particular java/javax APIs-->  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />  <!-- Global JNDI resources  Documentation at /docs/jndi-resources-howto.html  -->  <GlobalNamingResources>  <!-- Editable user database that can also be used by  UserDatabaseRealm to authenticate users  -->  <Resource name="UserDatabase" auth="Container"  type="org.apache.catalina.UserDatabase"  description="User database that can be updated and saved"  factory="org.apache.catalina.users.MemoryUserDatabaseFactory"  pathname="conf/tomcat-users.xml" />  </GlobalNamingResources>  <!-- A "Service" is a collection of one or more "Connectors" that share  a single "Container" Note: A "Service" is not itself a "Container",  so you may not define subcomponents such as "Valves" at this level.  Documentation at /docs/config/service.html  -->  <Service name="geumcheon-cms-server">  <Connector port="8080" protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8443"  URIEncoding="UTF-8"  />  <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />  <Engine name="geumcheon-cms-engine" defaultHost="localhost">  <Realm className="org.apache.catalina.realm.LockOutRealm">  <Realm className="org.apache.catalina.realm.UserDatabaseRealm"  resourceName="UserDatabase"/>  </Realm>  **<Host name="localhost" appBase="webapps/geumcheon-cms"**  **unpackWARs="true" autoDeploy="false" deployOnStartup="false">**  **<Context path="" docBase="cms.war" reloadable="true" >**  **</Context>**  **<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"**  **prefix="localhost\_access\_log" suffix=".txt"**  **pattern="%h %l %u %t &quot;%r&quot; %s %b" />**  **</Host>**  </Engine>  </Service>        <Service name="geumcheon-api-server">  <Connector port="8081" protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8444"  URIEncoding="UTF-8"  />  <Connector port="8010" protocol="AJP/1.3" redirectPort="8443" />  <Engine name="geumcheon-api-engine" defaultHost="localhost">  <Realm className="org.apache.catalina.realm.LockOutRealm">  <Realm className="org.apache.catalina.realm.UserDatabaseRealm"  resourceName="UserDatabase"/>  </Realm>  **<Host name="localhost" appBase="webapps/geumcheon-api"**  **unpackWARs="true" autoDeploy="false" deployOnStartup="false">**  **<Context path="" docBase="api.war" reloadable="true" >**  **</Context>**  **<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"**  **prefix="localhost\_access\_log" suffix=".txt"**  **pattern="%h %l %u %t &quot;%r&quot; %s %b" />**  **</Host>**  </Engine>  </Service>        </Server> |