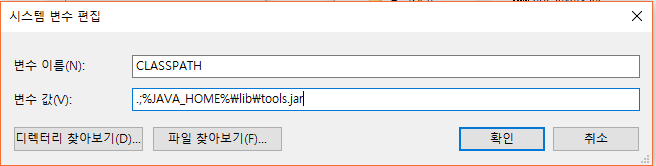
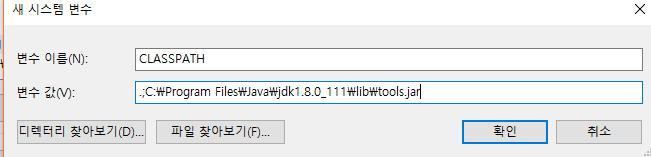
2017-07-03

1. 개발 환경 설정
2. Java 개발연도 & 개발된 이유
   1. 91년 개발 ex) 세탁기
   2. Software 재사용 (하드웨어 소프트웨어 개발)
      1. 시간을 절약하기 위해서
   3. 플랫폼에 독립적이다.
      1. 운영체제에 동일
   4. 네트워크
   5. ‘왜 만들어졌을까?’를 고민해볼 것
3. JAVA\_HOME 환경변수 설정 이유
   1. WAS실행 또는 다른 프로그램이 실행되기 위해서 필요
   2. 제일 상단으로 위치 변경
   3. 다른 Java버전을 사용 할 수 있기 때문이다.
4. CLASSPATH
   1. 라이브러리에 대한 위치를 설정
   2. 서버 배포시에는 반드시 CLASSPATH 환경변수 설정 필요
   3. eclipse에서는 프로젝트별 CLASSPATH설정
5. 윈도우 운영체제는 시스템변수 대소문자 구분X, 다른 운영체제에서는 시스템변수 대소문자 구분O
   1. 이후 필요한 path는 추가 가능

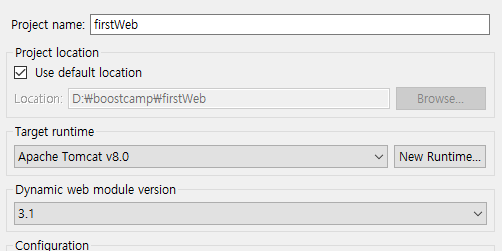


1. EUC-KR은 표현하지 못하는 문자가 존재 그러므로 UTF-8사용
2. MySql Workbench공부
3. Database와 DBMS의 차이점
   1. Database: 데이터를 저장 할 수 있는 Repository. (집에 있는 책)
   2. DBMS: Mysql이나 Oracle처럼 Database를 관리하는 시스템(엄마)
4. Oracle -> Role 공부
5. MySql 사용자 삭제
   1. mysql> drop user 사용자명@호스트;
6. Junit, xunit
7. test code는 배포하지 않기 때문에 디렉토리를 다르게 만든다
8. 패키지의 이름이 같으면 물리적인 다른 공간이여도 접근이 가능하다.
9. Junit 3 -> Test할 class를 상속.
   1. setup -> method -> teardown
10. Junit 4 -> 상속 대신 **annotation이용**
    1. 내부적으로 처리(중요)
11. JUnit 3과 4 동시 사용 시 충돌 발생
12. try hello world -> annotation 공부
13. Web Server와 WAS의 차이
14. ejb
15. 이전 Apache와 Tomcat둘 다 사용하는 경우 Apache:80, Tomcat:8080 포트로 apache->tomcat순서로 거쳐 들어오기 때문에 8080을 사용했었다.
16. tomcat\conf\server.xml



* 1. URIEncoding Get방식만 처리

1. Tomcat설정
   1. Preferences > Server > Runtime Environment
2. Web 은 main method가 없다. 내가 시작시켜주는 것이 아니다. 실행은 WAS Tomcat이 시작시켜준다.



* 1. module version 3.x 이상부터는 annotation을 사용

1. WAS 구성 <- <http://localhost/name/x.html> (접속 주소)
   1. Web Project 구조
      1. WEB-INF (외부에서 절대 접근 불가능)
         1. lib (WAS에 필요한 library를 반드시 내부적으로 포함해야 된다.)
         2. classes (eclipse에서는 굳이 노출시키지 않는다
         3. web.xml (배치 파일, 웹의 모든 정보)
      2. html
      3. css
      4. jsp (보안상의 이유로 WEB-INF로 옮기는 경우도 있다.)
2. MVC1 -> .jsp or .php, MVC2 -> .jsp가 없다.
3. Servlet은 클래스 그러므로 src에 생성
   1. Servlet Framework
   2. Servlet 내부 동작을 보면 Spring Framework 이해하는데 도움이 된다.
   3. init(), service(), destroy()
   4. servlet하나 마다 ServletConfig를 갖는다.
   5. ServletContext -> Servlet 전체에 대한 설정을 담고 있는 객체
   6. 생성시 classes에 생성된다. WEB-INF에 접근 불가하기 때문에 web.xml에 servlet에 대한 정보를 기술
   7. welcome-file list는 url을 입력하지 않아도 출력되는 파일.
   8. xml파일의 스키마는 xmlns:xsi의 url에 정의되어 있기 때문에 형식에 맞춰서 값을 입력해줘야 한다.
      1. Servlet하나 당 xml파일에 servlet, servlet-mapping 등록을 반드시 해야 한다.
      2. 3.x 이상부터는 annotation을 사용해서 대체 가능
      3. servlet-name은 아무거나 servlet, servlet-mapping 두 개의 이름만 맞춰주면 된다. 그러나 패키지 포함한 서블릿 이름을 중복방지를 위해서 주로 사용했다
   9. **dispatcher servlet**
   10. 설정이 필요한 이유를 아는 것이 중요
   11. Framework은 meta-file이 항상 존재한다. 정보를 알려주는 파일이 필요하다.
   12. servlet-class -> 반드시 패키지명을 포함한 class이름을 적어야한다.
   13. web.xml을 변경한 경우 반드시 server restart 환경 설정을 변경했기 때문이다.
   14. <http://localhost:8080> -> Tomcat
   15. ../firstWeb -> firstWeb Project
   16. ../hello -> web.xml의 내용을 토대로 Servlet을 찾는다.
       1. <servlet-mapping> -> /hello 찾고 servlet-name이 동일한 servlet-class를 찾는다.
   17. 처음 Servlet객체를 생성하면 init이 된다. 매번 다른 사용자가 요청을 하면 객체를 생성하지 않고 thread를 생성해서 사용
   18. Thread를 따로 명시해주지 않아도 되는가?
   19. 요청이 들어오면 HttpServlet Request와 HttpServlet Response를 두 개 자동으로 만들어 진다.
   20. Request에 Client의 정보를 가져온다.
       1. Client의 property를 보고 언어에 대해서 선택해서 그에 맞는 화면을 뿌려 줄 수 있다.
   21. Response에 응답할 정보를 넣어준다.
   22. query string
       1. …..url**?name=haha**
   23. service는 무조건 호출됨, doGet, doPost는 각 요청에 따라서 다른 응답을 하고 싶을 때 사용
   24. 부모(httpservlet) service에서 request의 method를 알아내서 실행 (Get인지 Post인지)
   25. GET사용 예) 타인과 인터넷 url공유하는 경우
   26. jsp -> Servlet으로 변경되어서 실행, jsp container가 변경해주는 역할 수행
   27. Jsp Container는 Jsp를 실행
   28. Servlet Container는 Servlet을 실행
   29. 이클립스 전에 톰캣을 사용하기 위해선
       1. C:\wbjava\apache-tomcat-8.0.44-windows-x64\apache-tomcat-8.0.44\webapps에 Dynamic Web Project 디렉토리 구조로 프로젝트를 구성해서 올려서 실행해야 한다.
   30. 내장 객체 ex) session