

		작업번호	페이지
		A01	1/81
작성자	확인자	승인자	작성일
작성자	담당자	담당자	2020-07-27
제목	파일명		
웹 프로그래밍 입문자를 위한 핵심 요약	[완료] 웹 프로그래밍 입문자를 위한 핵심 요약.hwp		

웹 프로그래밍

(백엔드, 질의어, 프론트엔드, 프레임워크)

입문자를 위한 핵심 요약

-
- 주 제 : 주제
 - 학 번 : 학번
 - 성 명 : 성명
 - 연 락 처 : 연락처
-

제목	작성일	페이지
웹 프로그래밍 입문자를 위한 핵심 요약	2020-07-27	2/81

<제 목 차 례>

1. 백엔드 (Back-end)	8
가. Java	8
나. JSP & Servlet	26
2. 질의어	32
가. MySQL & Oracle	32
나. Oracle	36
3. 프론트엔드 (Front-end)	55
가. HTML	55
나. CSS	58
다. JavaScript & jQuery	59
4. 프레임워크 (Framework)	66
가. 스프링 (Spring)	66
5. 기타	81
가. Eclipse Maven 프로젝트 개발환경 설정	81

제목	웹 프로그래밍 입문자를 위한 핵심 요약	작성일	2020-07-27	페이지	3/81
----	-----------------------	-----	------------	-----	------

1. 백엔드 (Back-end)

가. Java

<p>1. 자바의 특징</p> <p>가. 자바는 1996년 1월에 미국 선 마이크로 시스템사에서 제임스 고슬링과 아서 벤 호프와 같은 썬 개발자에 의해 개발된 언어이다.</p> <p>나. 자바는 운영체제에 독립적인 컴파일 언어이다.</p> <p>다. 최초 작성한 자바 언어는 파일명.java로 저장한다. 이 언어를 원시 언어라 한다. 원시 언어를 자바 컴파일 명령어인 javac.exe에 의해서 다음과 같이 컴파일 한다. javac 파일명.java 로 컴파일 하면 바이트 코드 클래스 파일이 만들어진다. 파일명.class</p> <p>라. 컴파일 된 클래스 파일에서 .class 확장자를 버리고, java 클래스명만 입력하면 자바 가상 머신 (JDK)에 로드되어 실행된다. 즉 운영 체제와 상관없이 JDK에 클래스 파일이 실행되는 것이 자바의 가장 큰 특징이다.</p> <p>참고)</p> <p>CMD에서 패키지 클래스 컴파일 하는 방법</p> <p>1. javac -d ./Test.java</p> <p>가. -d는 디렉토리 (패키지)를 뜻함. 이 옵션을 지정하면 package로 설정된 해당 폴더 (패키지)가 만들어 진다.</p> <p>나. .은 현재 디렉토리 경로</p> <p>CMD에서 패키지 클래스 파일 실행법</p> <p>1. c:>java -cp 패키지전까지경로 패키지명.클래스명</p>

<p>2. 자바 개발 환경 설정</p> <p>가. 먼저 www.oracle.com 회사 사이트로 접속해서 자바 가상 머신 JDK를 다운 받고 운영체제에 설치한다 (도서 참조).</p> <p>나. 그리고 JDK를 설치한 후 대부분 개발 툴로 이클립스를 사용하기 때문에 과거와 달리 path 운영체제 환경설정과 CLASSPATH 자바 클래스 환경설정을 각각 설정할 필요 없다.</p> <p>다. 자바 개발 툴인 이클립스를 www.eclipse.org/home/index.php로 접속해서 이클립스 압축파일을 다운받아 압축을 풀어준다. 그러면 eclipse라는 폴더가 생기고 해당 폴더 내에 eclipse.exe 실행 파일이 있다.</p> <p>라. 이 파일을 실행해서 작업 폴더 경로를 만든다. 이 폴더 안에 자바 프로젝트를 만들고 자바 클래스를 만들어 실행하면 된다. 이 과정에서 작업 폴더는 이클립스가 알아서 만들어 준다.</p> <p>마. 이클립스 실행 전 반드시 JDK를 먼저 설치해야 한다. 그렇지 않으면 이클립스가 실행시 에러를 발생하고 이클립스를 실행할 수 없다. 이는 이클립스가 실행될 때 JDK를 함께 로드되면서 실행하기 때문이다.</p>

제목	웹 프로그래밍 입문자를 위한 핵심 요약	작성일 2020-07-27	페이지 4/81
----	-----------------------	-------------------	-------------

3. 자바 기본형 (Data Type)과 참조형 (Data01.java)

- 가. 정수형 숫자형 : byte (1바이트), short (2바이트), int (4바이트), long (8바이트)
- 나. 실수형 숫자형 : float (4바이트), double (8바이트)
- 다. 논리형 (true or false) : boolean
- 라. 단일 문자 : char

자바 기본형 8개 외 참조형이 있다.

가. 참조형 : 클래스형, 배열형, 인터페이스형

1. 자바의 산술(수학) 연산자 (Data02.java)

가. + (덧셈), - (뺄셈), * (곱셈), / (나눗셈), % (나머지 연산)

2. 자바의 관계/비교 연산자 (Data03.java)

가. > (~보다 크다), >= (~보다 크거나 같다), < (~보다 작다), <= (~보다 작거나 같다)
== (같다 연산), != (같지 않다)

3. 자바의 논리연산자 (Data04.java)

- 가. && (논리곱) : 2개의 조건이 모두 true 이어야 결과값도 true이다.
- 나. || (논리합) : 2개의 조건중 하나라도 true 이면 결과값도 true 이다.
- 다. ! (not) : 입력값이 true 이면 결과값은 false이고, 반대로 입력값이 false이면 결과값은 true이다.

4. 자바 변수 개념, 변수 선언법, 대입연산자, 증감연산자 (Data05.java)

가. 변수란 자료형 값을 저장하는 장소를 뜻한다. 변수에 저장되는 값은 최종적으로 가장 마지막에 저장되는 값이 저장된다.

나. 자바 변수 선언법:

예) int a = 7;

여기서 =은 대입 연산자 (또는 할당 연산자)라 한다. 대입 연산자는 오른쪽의 값을 왼쪽 변수에 대입한다. ; (세미콜론)는 한 문장의 끝을 뜻한다.

다. 자바 증감연산자

++ (1씩 증가) : ++a (선행 증가), a++ (후행 증가)

-- (1씩 감소) : --a (선행 감소), a-- (후행 감소)

1. 자바 if ~ else 조건문 (Data06.java)

```

가. if (조건식) {
    조건식이 참이면 실행;
}
나. if (조건식) {
    조건식이 참이면 실행;
} else {
    조건식이 거짓이면 실행;
}
다. 다중 조건문
If (조건식 1) {
    조건식 1이 참이면 실행;
} else if (조건식 2) {
    조건식 2가 참이면 실행;
} else {
    조건식 1,2 모두 거짓일때 실행;
}

```

2. 다중 선택이 가능한 switch ~ case 조건문 (Data07.java)

```

형식) switch (정수식) {
    case 값2 : 처리할 문장 2; break;
    중략
    default: 처리할 문장;
}

```

정수식과 case문의 값과 일치하면 해당 처리할 문장이 실행된다. 그리고 문장 실행 후 break문을 만나면 switch ~ case문을 종료 한다. 그리고 default문은 해당 조건이 없을 때 실행된다.

참고) JDK 1.7부터는 switch 문의 조건식에도 문자열이 올 수 있다. case문의 값도 문자열이 올 수 있다.

1. for 반복문 (Data08.java)

형식)

```
for (초기치; 조건식; 증감식) {
    조건식이 참일 동안만 반복;
}
```

가) 처음엔 초기값을 실행한다. 초기값이란 변수에 최초의 값을 저장하는 것을 말한다.

나) 조건식을 평가하여 참일 동안 문장을 반복 실행한다.

다) 증감식을 실행한다. 다시 조건식을 평가하여 참이면 문장을 반복하고 거짓이면 반복문을 종료 한다.

2. 확장 for 반복문 (Data13.java)

가) 확장 for 반복문은 JDK 1.5부터 배열과 컬렉션에 저장된 요소에 접근할 때 기존 for 반복문보다 편리한 방법으로 처리할 수 있도록 새롭게 추가되었다.

나) 확장 for 반복문 형식

```
for (타입 변수명: 배열 또는 컬렉션) {
```

// 배열 또는 컬렉션에 저장된 값이 매 반복마다 하나씩 순서대로 읽혀져 변수에 저장된다.

```
}
```

다) 확장 for 반복문은 일반적인 for 반복문과 달리 배열과 컬렉션에 저장된 요소들을 읽어오는 용도로만 사용할 수 있다.

2. while 반복문 (Data09.java)

형식)

```
while (조건식) {
    조건식이 참일 동안만 반복;
    증감식;
}
```

3. do ~ while 반복문 (Data09.java)

형식)

```
do {
    조건식이 참일 동안만 반복;
    증감식;
} while (조건식);
```

주의 사항) 나중에 조건식을 검사하기 때문에 조건식이 거짓이라도 무조건 한번은 반복한다는 단점이 있다. 실제 자바 프로젝트 (웹 개발)에서는 do ~ while 반복문은 잘 사용되지 않는다.

제목	웹 프로그래밍 입문자를 위한 핵심 요약	작성일 2020-07-27	페이지 7/81
----	-----------------------	-------------------	-------------

4. 무한 루프란?

가. 무한 루프란 조건식이 무조건 참이어서 영원히 반복하는 반복문을 뜻한다.

5. break, continue문 (Data10.java)

가. break문: 반복문 내에서 break문을 만나면 반복문을 중단한다. break 자신과 가장 가까운 반복문을 벗어난다 (이중 for문일 경우 안쪽 for문부터 중단시킬 수 있음).

나. continue문: 반복문 내에서 continue문을 만나면 아래문장을 실행하지 않고 다음 반복을 위해서 반복문 처음으로 돌아가서 그 다음 반복을 수행한다. 즉 continue문과 반복문 블록의 끝 “}” 사이의 문장들을 건너뛰고 반복을 이어가는 것이다.

1. 배열의 특징과 생성법 (Data13.java, Data14.java)

가. 배열이란 동일한 자료 타입을 가진 자료들을 한꺼번에 저장하기 위해서 사용하는 것을 배열이라 한다. 배열은 고정된 크기이다.

나. 배열 생성법

new 키워드를 이용한 배열 생성법

형식) `int [] score = new int [3];` 배열 크기가 3인 정수형 배열 score 생성

배열 요소 초기화를 통한 배열 생성법

형식) `int [] score = {10,20,30};` 배열 요소값을 초기화 하면서 배열 크기가 3인 배열 score 생성

다. 배열 길이가 0인 배열도 생성 가능하다. 배열의 길이는 int 범위의 양의 정수 (0도 포함)이어야 한다.

라. 배열 길이는 배열이름.length를 통해서 알 수 있다.

2. 자바 클래스 구성 요소 (Data15.java)

가. 자바 클래스는 class 키워드로 정의한다.

나. 클래스 구성 요소

```
class Test { // 자바 한줄 주석문 기호
    int a = 10; // 변수 선언부 (멤버 변수)
```

생성자 영역 부분;

사용자 정의 메서드;

}

다. 사용자 정의 메서드 생성 문법

형식)

```
public void p ( ) { // void는 반환값이 없는 자료형 즉 리턴 타입이 없다는 뜻.
    실행 문장;
```

```
} // p ( )가 사용자 정의 메서드 명
```

사용자 정의란 자바 개발자에 의해서 임의로 정의 가능한 이름을 뜻한다.

3. 자바 주석문 기호

가. 주석문이란 소스에 대한 설명문으로 실행되지 않는다.

나. // 한줄 주석문 기호

```
/*
```

```
* 한줄 이상 주석문 기호
```

```
*/
```

4. 키워드와 return

가. 키워드는 예약어로서 약속된 명령어이다. 키워드를 가지고 변수명으로 정의할 수 없다.

나. return 키워드는 반환값을 되돌릴 때 사용한다.

1. package와 import (Data16.java)

가. package는 클래스 묶음을 뜻한다. 그러므로 이 키워드로 클래스 묶음을 저장할 수 있는 폴더를 생성할 수 있다.

나. import 키워드는 외부 패키지 클래스를 읽어 올 때 사용한다. 이클립스 import 단축키는 ctrl+shift+o (영문) 이다.

2. \n과 \t

가. \n은 수직으로 개행한다.

나. \t는 수평으로 개행한다.

3. 자바 클래스 정의와 객체 생성법

가. 자바 클래스 생성 문법

형식) **class** 클래스 이름 {

변수 선언;

생성자 () { }

public 자료형 메서드명 () {

}

} 라고 먼저 클래스를 정의하고 난 이후

나. **new** 키워드로 새로운 객체명을 생성한다.

형식) 클래스명 객체명 (객체주소를 가리킴) = **new** 클래스명();

4. 자바의 접근 지정자 (Data16.java)

가. **private** : 내 자신 클래스 내에서만 접근 가능하다 (자바의 보안성).

나. 기본 접근 지정자 (생략) : 같은 패키지의 내 자신클래스와 다른 클래스에서 접근 가능하다.

다. **protected** : 같은 패키지 내의 내 자신 클래스와 다른 클래스 또는 상속받은 자식 클래스에서 접근할 수 있다. 특히 상속받은 자식클래스에서 접근할 때는 외부 패키지에 서도 접근할 수 있다.

라. **public** : 같은 패키지뿐만 아니라 외부 패키지 다른 클래스에서도 접근 할 수 있다. 누구나 다 접근 할 수 있다.

1. 자바의 메서드 오버로딩 개념 (Data17.java)

가. 메서드 오버로딩이란 한 클래스 내에 같은 이름의 메서드를 여러 개 정의하는 것을 말한다.

나. 메서드 오버로딩 구분 요건

첫째. 메서드의 전달인자 개수를 다르게 한다.

둘째. 메서드의 전달인자 자료형을 다르게 한다.

셋째. 메서드의 전달인자 순서를 다르게 한다.

- 전달인자 개수가 다른 메서드 오버로딩이 되면 오버로딩된 메서드만큼 메서드명을 정의해야 되기 때문에 JDK 1.5 이후부터 가변인자 문법이 추가됨 (Data18.java).

2. 자바의 생성자 특징

- 가. 생성자 이름은 클래스 이름과 같다.
- 나. 생성자는 **new** 클래스명 (); 에 의해서 호출 ([Data19.java](#))
- 다. 생성자는 메서드의 일종이다.
- 라. 생성자의 자료형 타입이 없다.
- 마. 생성자의 주된 기능은 변수 초기화, 객체 생성 ([Data20.java](#))
- 바. 생성자는 오버로딩이 가능 ([ConsTest01.java](#))
- 사. 생성자는 상속이 되지 않는다.
- 아. 같은 클래스에서 다른 생성자 호출 법 ([ConsTest02.java](#))
예) **this** ();

3. 클래스 (정적) 메서드와 클래스 (정적) 변수 ([StTest03.java](#))

- 가. **static** 예약어로 정의된 메서드를 클래스 (정적) 메서드라 한다. 클래스 메서드는 객체 생성 없이 클래스로 직접 접근 한다. 물론 **new** 키워드로 생성된 모든 객체에 의해서 정적메서드는 공유되기 때문에 생성된 객체로도 접근할 수 있다. 클래스 메서드 내에 인스턴스 변수와 **this**를 사용할 수 없다.
- 나. **static** 예약어로 정의된 변수를 클래스 (정적) 변수라 한다. 정적 변수도 클래스로 직접 접근 한다.
 - 클래스 (설계도)의 소속 멤버변수는 정적 변수 (클래스로 직접 접근 가능, 속도 빠름)와 인스턴스 변수로 나눌 수 있음.
 - 삼성 Tv는 부산으로 보내고 (정적 변수) 아프리카로 보낼 경우 (인스턴스 변수) 아프리카에서 배달된 것을 알 수 있을까? 모름
 - 빠른 정적 변수로 설계할 경우 쓸모없는 변수까지 불러들이기 때문에 메모리 과부하 발생되나 인스턴스 변수는 JDK에서 자동으로 쓰지 않는 변수를 메모리에서 제외시킴.
 - 개발할 경우 **int** 속도가 제일 빠르고 **double**은 정밀도가 높아서 많이 사용함.

4. 자바 저장 빈 클래스 생성 문법 (InfoMain.java)

가. 빈클래스 생성 문법

형식)

```
public class 클래스명 {
    private String name;
```

중략

// 빈클래스는 변수명을 정의할 때 반드시 private으로 정의한다. 이유는 내 자신 클래스내에서만 접근 할 수 있기 때문이다 (데이터 보안성).

// String name은 매개변수 or 전달인자명

```
public void setName (String name) { // setter ( ) 메서드 정의 (값을 저장)
    this.name = name; // 이름이 같아서 저장하기 위해서 this로서 내 자신 클래스를
    가리킴.
}
public String getName ( ) { // getter ( ) 메서드 정의
    return name; // return 키워드로 값을 반환
}
```

나. 빈 클래스명은 중간에 자료를 저장하는 역할을 하기 위해서 사용한다.

다. 빈 클래스는 자바의 기본자료형 8개 외 레퍼런스 자료형인 클래스에 속한다.

1. 자바의 상속 (ExtendTest04.java)

가. 상속이란 자식이 부모가 가지고 있는 재산이나 권력을 물려받는다는 의미이다.

나. 상속문법

형식) class 자식클래스명 **extends** 부모클래스명 {

}

extends 키워드로 부모클래스로부터 상속을 받는다.

클래스 상속은 하나의 부모로 부터만 단일 상속만 가능하다.

- 클래스를 이혼을 허락하지 않기 때문에 부모님 2명에서 상속 받을 수 없음.
- 제임스 고슬링 (캐나다 출신 개발자)이 Java 설계할 때 최고 조상을 찾는 장유유서 (효도 정신)가 깃들여 있음.
- 조상 클래스에서 매서드를 찾아서 상속할 경우 불필요한 코드를 작성할 필요 없음.

2. 메서드 오버라이딩 (OverTest05.java)

가. 자손 클래스에서 부모 클래스의 기존 메서드와 시그니처 즉 전달인자의 이름, 자료형, 개수, 반환 타입을 동일하게 정의한다. 부모클래스로 부터 상속받은 메서드 내용을 자식 클래스에 맞게 변경하는 것을 오버라이딩이라 한다.

나. 메서드 오버라이딩 전제 조건

반드시 부모 자식간의 상속관계를 만들어야 한다.

- 30년 교육방식과 현재 교육방식이 다르기 때문에 이 경우 상속할 때 문제가 발생됨. 즉 30년 교육방식을 상속한 후에 현재 교육방식을 재정의해야 함. 단, 교육방식이 변화하지 않는다고 가정한다면 상속 받아도 괜찮음.

- 자바 웹개발에서 오버로딩보다 오버라이딩이 중요하고 상속받을 경우 오버라이딩하는 것이 좋음.

- 생성자 오버라이딩이 없음. 생성자는 상속되지 않기 때문에 오버라이딩 안됨. 그러나 오버로딩은 가능함.

3. super (SuperTest06.java, ThisTest07.java)

1. 상속관계에서 자식 클래스 영역에서 부모 클래스의 메서드가 오버라이딩된 경우 부모 클래스의 메서드를 호출하려면 **super.메서드();** 로 접근한다.

2. **super**는 자손 클래스에서 조상 클래스로부터 상속받은 멤버 변수를 참조하는 데 사용되는 참조변수이다.

3. **super**는 정적 메서드 내에서는 사용할 수 없고 인스턴스 메서드에서만 사용한다.

4. 상속관계에서 자식 클래스에서 부모 클래스부터 상속받은 멤버변수에 접근하려면 **super.변수명**으로 접근한다. 즉 조상 클래스 멤버 변수 이름과 동일한 이름을 자손 클래스에서 정의 했을 경우 자손 클래스에서 조상 클래스로부터 상속받은 멤버 변수에 접근할 때 **super.변수명**으로 접근해서 값을 가져온다. 자손 클래스 멤버변수에 접근할 때는 **this.변수명**으로 값을 가져온다.

- **super**은 정적 메서드에서 사용할 수 없고 인스턴스 변수에서 사용할 수 있음.

- 조상 클래스를 접근할 경우 **super**로 접근하고 자손 클래스는 **this**로 이용해야 함.

- 프로그램 내에 조상 및 자손 클래스에서 변수명 (int a)이 같은 경우 **super**, **this**를 구분해야 되고 **this**를 생략해도 됨.

1. 상속에서의 생성자 호출 문제 (ConsTest08.java, ConsTest09.java)

가. 전달인자가 없는 생성자를 기본 생성자라 한다.

나. 생성자가 오버로딩 되면 자바는 기본 생성자를 묵시적으로 제공하지 않는다.

다. 그러므로 생성자가 오버로딩 될 때 자식클래스에서 묵시적으로 제공하지 않는 부모 클래스의 기본 생성자를 호출하고자 할때 문제가 발생한다. 그러므로 부모 클래스에서 생성자가 오버로딩 되면 상속에서의 생성자 호출문제가 발생할 수 있으므로 명시적인 기본 생성자를 정의하는 것이 좋다 (웹 개발자는 99.99%는 기본 생성자를 정의하지 않고 경로만 변경함).

라. 생성자는 상속되지 않는다. 그러므로 자식 클래스에서 new 클래스명();에 의해서 생성자를 호출하면 자바는 먼저 부모 클래스의 기본 생성자를 먼저 호출하도록 설계되어 있다 (장유유서 및 호도 정신).

마. 자식 클래스에서 부모 클래스 기본 생성자를 호출하고자 할 때 super();로 한다.

- 자손이 조상을 찾게 되는 것은 제임스 고슬린의 호도 정신 (장유유서)을 엿볼 수 있음.

2. 업캐스팅과 다운캐스팅 (UpCastTest10.java)

가. 업캐스팅은 상속관계에서 자식타입의 참조변수가 부모타입의 참조변수로 변환하는 것을 말한다.

나. 다운캐스팅은 상속관계에서 부모타입의 참조변수가 자식타입의 참조변수로 변환하는 것을 말한다.

다. 다운캐스팅 전제 조건

첫째. 상속관계를 만들어야 한다.

둘째. 다운캐스팅을 하기 전 먼저 업캐스팅을 해야 한다. 이유는 업캐스팅을 한 것에 한해서만 다운캐스팅을 허용한다.

셋째. 캐스팅 연산자를 사용하여 다운캐스팅을 한다.

- 캐스팅은 참조 타입 간의 형변환을 말함.

- 자손끼리 상속되지 않기 때문에 자손간의 형변환은 안됨.

- 다운캐스팅의 경우 형변환 연산자 (캐스팅 변환자)를 사용하나 업캐스팅은 할 필요 없음.

3. 자바의 추상클래스 특징 (AbsTest11.java)

- 가. 추상클래스는 **abstract** 키워드로 정의한다.
- 나. 추상클래스는 **new** 키워드로 객체 생성을 못한다.
- 다. 추상클래스에는 추상메서드가 올수 있다. 추상메서드는 { }가 없고 실행문장이 없다. 그러므로 호출이 불가능하다.
- 라. 추상클래스를 상속받은 자식클래스에서 부모 추상클래스의 추상메서드를 오버라이딩해야 한다.
 - 일반 클래스와 달리 추상 클래스는 { }를 올 수 없음. 추상 메서드를 강제 오버라이딩하기 위해서 사용함. 이는 자바의 매개변수 다형성 문법을 적용하기 때문이다.
 - 다형성을 하지 않을 경우 메서드 100개 정의해야 하고 많은 시간 및 불필요한 코드가 많아지나 이를 활용하면 메서드 1개로 업캐스팅 할 수 있음.
 - 형변환 판단 연산자 (InstanceOfTest11.java)

4. final 키워드 특징 (Final12.java, Final13.java, Final14.java)

- 가. 변수를 **final**로 정의하면 수정할 수 없는 변수 즉 상수가 된다.
- 나. 클래스를 **final**로 선언하면 더 이상 상속을 허락하지 않는다.
- 다. 메서드를 **final**로 선언하면 더 이상 오버라이딩을 허락하지 않는다.

1. 자바의 인터페이스 특징 (InterTest15.java, InterTest16.java, InterTest17.java, model package)

- 가. 인터페이스는 **interface** 예약어로 정의한다.
- 나. 인터페이스는 하나 이상의 부모로부터 다중 상속이 가능하다.
- 다. 인터페이스에는 **public static final**로 인식되는 정적상수만 올수 있다.
- 라. 인터페이스는 **public abstract** 로 인식되는 추상메서드만 올수 있다.
- 마. 인터페이스는 자식클래스에서 **implements** 키워드로 상속을 받는다. 상속받은 자식클래스에서 부모 인터페이스의 추상메서드를 반드시 오버라이딩을 해야 한다. 그래야 자식클래스 객체 생성이 가능하다.
- 바. 인터페이스는 **new**로 객체 생성을 못한다.
 - 인터페이스는 빈껍데기로서 상속 대신 구현 상속이라고 함.
 - **public static**은 정적 변수이고 디폴트로 생략 가능.
 - **JDK 1.8**에서는 **default** 메서드 외 추상 메서드로 올 수 있음.
 - 추상 클래스는 불완전한 설계도이나 인터페이스의 경우 도화지라고 함.
 - 추상 클래스는 조상이 있으나 인터페이스의 경우 조상이 없음.
 - 추상 클래스를 상속 받는 것보다는 인터페이스에서 상속을 받음.

2. 자바의 상속방법 (InterTest18.java)

가. 클래스에서 인터페이스를 상속: implements

나. 클래스에서 부모클래스 상속: extends

다. 인터페이스에서 인터페이스 상속: extends

- 구글 >> java 8 API 검색 >> 자바 기술 분석서 >> class object (단군시조 클래스) + API 활용 (ObjectTest19.java, EqualTest20.java)

3. 자바의 래퍼 클래스 (InterTest21.java)

가. 래퍼 (Wrapper) 클래스는 자바의 기본 자료형 8개를 포장해서 클래스화 한것을 래퍼 클래스라 한다.

나. 래퍼 클래스 종류

기본 자료형	래퍼 클래스
byte	Byte
short	Short
int	Integer
long	Long
float	Floag
double	Double
boolean	Boolean
char	Character

4. 오토 박싱과 오토 언박싱 (InterTest21.java)

가. 오토 박싱 : 자바 기본 자료형을 래퍼 클래스형으로 바뀌는 것을 말한다. 오토 박싱은 자동 형변환이 이루어 진다.

나. 오토 언박싱 : 래퍼 클래스형이 기본 자료형으로 바뀌는 것을 말한다.

- 내부 컴파일 내에서 자동으로 변환함.

1. 자바 배열과 java.util 패키지의 컬렉션의 차이점

가. 배열의 특징

- 첫째. 배열은 단 하나의 자료형만 저장할 수 있다.
- 둘째. 배열은 고정된 크기이다.
- 셋째. 배열은 하나이상의 요소값을 저장할 때 사용한다.

나. 컬렉션 특징

- 첫째. 컬렉션은 복수개의 자료형을 동시 저장할 수 있다.
- 둘째. 컬렉션은 복수개의 요소값을 동시 저장이 가능하다.
- 셋째. 컬렉션은 가변적 배열이다.
- 컬렉션은 Set, List*, Web* 인터페이스
- 웹 개발에서 Set을 주로 사용하지 않음.

2. java.util의 Set 인터페이스와 List 인터페이스 특징 (Iterator08.java)

가. Set 인터페이스 (SetTest07.java)

- 첫째. 복수개의 요소값을 저장할 수 있다.
- 둘째. 요소값을 순서없이 저장한다.
- 셋째. 중복 요소값을 허용하지 않는다.
- 넷째. 가변적인 배열이다.
- 다섯째. 복수개의 타입을 동시 저장 가능하다.

나. List 인터페이스 (SetList06.java, Iterator08.java)

- 첫째. 복수개의 요소값을 저장할 수 있다.
- 둘째. 요소값을 순서있게 저장한다.
- 셋째. 중복 요소값을 허용한다.
- 넷째. 요소값 접근시, 배열처럼 인덱스 번호로 접근한다..
- 다섯째. 복수개의 자료형 타입을 동시 저장할 수 있다.

3. java.util의 Stack 클래스와 LinkedList 컬렉션 특징

가. Stack 클래스 (StackTest09.java)

- 첫째. 한 개 이상의 자료를 저장할 수 있다.
- 둘째. 순차적으로 저장된다.
- 셋째. FILO (First Input Last Output) 구조. 즉 먼저 입력된 것이 가장 나중에 나오는 구조이다. 이유는 입구와 출구가 같기 때문이다.
- 넷째. LIFO (Last Input First Output) 구조. 즉 가장 나중에 입력된 것이 가장 먼저 나오는 구조이다.

나. LinkedList 컬렉션 클래스 특징 (LinkedList10.java)

- 첫째. FIFO(First Input First Output) 구조이다. 먼저 입력된 것이 가장 먼저 나가는 구조이다.
- 둘째. 입구와 출구가 다르기 때문에 FIFO 구조이다.

1. 컬렉션 클래스 제네릭 (HashMap11.java)

가. 제네릭 배경 : 컬렉션은 복수개의 자료형을 동시 저장되다 보니 자료의 안정성이 떨어지고 원하는 자료 추출이 어렵다. 그러므로 지정한 자료형 타입 한개만 저장하기 위해서 제네릭을 이용한다.

나. 제네릭을 지정하면 원하는 자료형 추출이 쉽다.

다. List <String> city = new ArrayList <> ();

<String>이 제네릭이다. 즉 문자열만 저장 가능하다.

- 제네릭은 JDK 1.5에서 추가되었고 <>은 앞서 제네릭 타입을 생략하는 명령어로서 JDK 1.7 이후부터 사용할 수 있음.

- Java.util 패키지의 Map 인터페이스를 구현한 컬렉션 HashMap, Hashtable 클래스의 특징

1. 키, 값 쌍으로 저장하는 구조
2. 중복키는 허용하지 않고 중복값은 허용함.
3. 키를 기준으로 값을 빠르게 검색할 수 있음.

2. 자바의 멀티스레드

가. 멀티스레드는 하나의 프로그램 내에서 여러 개의 작은 작업이 동시에 수행되는 것을 말한다 (다중 작업).

나. 멀티스레드 구현 방법

첫째. Thread 클래스 상속받는 법 (Thread17.java)

Thread 클래스를 상속받아 멀티스레드를 구현하면 방법은 간단하나 단일 상속만 가능하다는 단점이 있다.

둘째. Runnable 인터페이스를 상속받는 법 (RunThread18.java)

다중 상속을 받을 수 있다는 장점이 있다. 그러나 멀티스레드를 시작하는 start () 메서드가 없기 때문에 인터페이스를 상속받은 자식클래스 객체를 Thread 클래스 생성자 인자값으로 넘겨 또 다시 객체를 생성해서 start () 메서드를 호출해야 한다는 단점이 있다.

다. 멀티스레드에서 start () 메서드를 호출하면 멀티스레드가 시작되면서 run () 메서드를 자동 호출한다. 명시적인 run () 메서드를 호출하지 않아도 start () 메서드만 호출하면 run () 메서드는 자동 호출하도록 자바는 설계 되어져 있다. 바로 자동 호출되는 run () 메서드 내에 멀티스레드 문장을 구현한다.

3. 자바의 예외 처리

가. 자바에서 예외는 정상적인 프로그램이 실행되는 동안 예상치 못했던 에러를 의미한다. 즉 컴파일된 바이트 코드 클래스 파일이 JVM이 로드되어져서 실행될 때 발생하는 예상치 못한 에러이다. 이 경우 자바에서 예외 처리 클래스를 통해서 처리한다.

나. 자바에서 대표적인 예외 에러는 숫자를 0으로 나눌 때에 발생한다.

다. 예외 처리 방법

첫째. try ~ catch문 ([TryCatch19.java](#))

```
try {
    정상구문이 실행되다 예외가 발생하면 제어가 catch문으로 옮겨진다.
} catch (예외 처리 클래스 e) {
    예외 처리할 문장;
} // catch문에서 예외를 예외 처리클래스로 처리한다.
```

둘째. throw 키워드 ([TryCater20.java](#))

throw는 예외를 일부러 발생시킬 때 사용하는 키워드이다.

형식) throw new 예외 클래스 생성자;

셋째. throws 키워드 ([NewThrow21.java](#))

throws 키워드는 발생된 예외를 자신이 직접 처리하는 것이 아니라 자신을 호출한 곳으로 떠넘기는 역할을 한다.

형식) public void 메서드명 (매개변수) throws Exception {.....}

넷째. finally 키워드

finally 키워드는 예외와 상관없이 무조건 마지막에 실행한다.

라. JDK 1.7 이후에 추가된 try-with-resources 문장

가. 자원은 프로그램이 종료되면 반드시 close () 메서드로 닫혀져야 한다. 그러나 try-with-resources 문을 사용하면 문장의 끝에서 자원들이 자동으로 닫혀지게 한다.

나. 형식)

```
try (리소스 자료형1 변수명1 = 초기값; 리소스자료형2 변수명2 = 초기값2;) {
    문장
}
```

다. 예시)

```
try (PrintWriter output = new PrintWriter("test.txt")) {
    for (String s:list){ // 향상된 for
        output.println(s.toLowerCase());
    } // for
}
```

이처럼 try 다음에 소괄호가 나오면 리소스로 취급한다. 즉 PrintWriter 객체가 try-with-resources 문으로 선언되었기 때문에 try 문장이 정상적으로 종료되든 예외가 발생하든 간에 무조건 닫혀진다. 즉 finally문에서 close (); 메서드로 명시적으로 닫지 않아도 자동으로 해당 객체는 닫힌다. 이 기능을 사용하려면 자원 객체가 [java.lang.AutoCloseable](#) 인터페이스를 구현하고 있어야 한다.

JDK 1.8부터 추가된 람다식 (LamDa15.java, LamDa16.java)

1. 람다식의 도입으로 인해 이제 자바는 객체지향언어인 동시에 **함수형 언어**가 되었다.
2. 람다식은 간단히 말해서 메서드를 하나의 식으로 표현한 것이다. 메서드를 람다식으로 표현하면 메서드의 이름과 반환값이 없어지므로 람다식을 **“익명 함수 (Anonymous function)”** 라고 한다.
3. 람다식은 메서드의 매개변수로 전달되어지는 것이 가능하고 메서드의 결과로 반환될 수 있다.
4. 람다식은 **익명함수**답게 메서드에서 이름과 반환타입을 제거하고 매개변수 선언부와 몸통 { } 사이에 “->”를 추가한다.

```
형식) 반환타입 메서드이름(매개변수 선언) {
    몸체 문장;
}
이를 람다식으로 바꾸면
(매개변수 선언) -> {
    몸체 문장;
}
```

- **내외부클래스** 관련
- **내부무명 클래스** (AnonyMous14.java)

java.io 입출력

1. InputStream은 바이트 입력 스트림의 최상위 추상클래스이다.
2. OutputStream은 바이트 출력 스트림의 최상위 추상클래스이다.
3. File 클래스는 파일 및 디렉토리를 관리할 수 있도록 기능을 제공해 주는 클래스이다. 주로 웹 애플리케이션 개발에서는 이진 파일을 업로드 하는 자료실을 만들 때 사용한다.
4. Reader는 문자 스트림을 읽어들이기 위한 최상위 추상클래스이다.
5. Writer는 문자 출력 스트림의 최상위 추상클래스이다.
6. InputStreamReader는 Reader로부터 상속된 클래스로서 바이트 스트림을 문자 스트림으로 변환하는 기능을 제공해 준다.
7. BufferedReader는 문자 및 입력 스트림으로부터 문자들을 읽을 때 버퍼링을 통해 텍스트를 효율적으로 읽는다.

java.net 네트워킹

1. 통신규약 (protocol) 이란 컴퓨터 상호간의 대화에 필요한 통신규약을 의미한다.
2. 소켓은 멀리 떨어져 있는 컴퓨터를 연결한 사용자 상호간에 데이터를 송수신하도록 하기 위해서 두 사용자를 연결시켜 주는 도구라 할 수 있다.
3. 정보나 서비스를 제공하는 쪽을 서버라고 하며 제공 받는 쪽을 클라이언트 (사용자)라 한다.
4. 메신저에서 클라이언트 프로그램이 서버 프로그램에 접속하려면 우선 서버 ip주소와 포트번호를 알아야 한다.
5. ServerSocket은 서버 측에서 클라이언트 접속을 기다리기 위해서 필요한 클래스이다.
6. Socket은 서버와 클라이언트 사이에 데이터 송수신하기 위해서 필요한 클래스이다.
7. 네트워크 프로그램을 작성하기 위해서는 서버 측 프로그램과 클라이언트 측 프로그램 모두 소켓 객체가 있어야 한다. 서버 측에서는 ServerSocket과 Socket 클래스 모두 있어야 하고 클라이언트 쪽에서는 Socket 클래스만 사용된다.
8. Socket 클래스 하위의 `getInputStream ()` 메서드로 입력 스트림 객체를 구하고 `getOutputStream ()` 메서드로 출력 스트림 객체를 구해서 데이터 송수신을 한다.

java.sql JDBC

1. SQL (Structured Query Language)의 약어로 관계형 데이터베이스에서 표준화된 질의 언어를 뜻한다.
2. JDBC 드라이버 클래스, 데이터 베이스 접속 주소, 접속 사용자, 사용자 비밀번호 (총 4가지)가 있을 경우 `Connection con`을 생성할 수 있다.
3. `Connection`은 특정의 데이터베이스와의 접속 (세션)을 표현한다.
4. `DriverManager` 클래스는 데이터베이스에 연결하기 위해 로드된 드라이버를 관리하는 역할도 하나 동시에 데이터베이스와의 연결을 위한 `Connection con`을 생성도 한다.
5. 따라서 `Connection`은 데이터베이스와의 연결을 의미하고 생성된 `Connection con`으로 `Statement st`를 생성해서 쿼리문을 실행한다.
6. `select` 검색 쿼리문은 `executeQuery ()` 메서드로 주로 실행하고 결과 레코드를 `ResultSet`으로 반환한다. 그 결과 레코드가 하나 이상 복수개이면 `while`문을 반복하고 하나이면 `if`문으로 레코드 값을 가져온다.
7. `insert`, `update`, `delete` 쿼리문은 `executeUpdate ()` 메서드를 사용하고 반환값은 수행된 레코드 행의 개수를 반환한다.
8. 저장 프로시저를 실행하기 위해서는 `CallableStatement`를 사용한다.

나. JSP & Servlet

JSP (Java Server Pages) 개요
1. Java를 이용하여 동적인 웹 페이지를 만들기 위해 선 마이크로 시스템社에서 개발한 서버 스크립트 언어이다.

HTML과 JSP 주석문 기호
1. html 주석문 : <code><!-- html 주석문 --></code> 2. jsp 주석문 : <code><%-- jsp 주석문 --%></code>

<code><%@ ... %></code>
1. <code><%@</code> 를 지시자 또는 지시어라 한다. JSP 파일 내에서 웹 서버에게 해당 페이지를 어떻게 처리할 것인지에 대한 정보를 주는 데 사용한다.

JSP에서 외부 패키지 import 방법 (jsp_data.jsp, header.jsp, footer.jsp, include_test.jsp)
1. <code><%@ page import = "java.util.*" %></code> : java.util 패키지의 모든 클래스 등을 import하여 읽어 들인다. 2. <code><%@ page import = "java.util.*, model.*" %></code> : java.util 패키지의 모든 클래스와 model 패키지의 모든 클래스 (복수개의 패키지)를 import 한다.

JSP에서 외부 jsp 파일을 읽는 방법 (header.jsp, footer.jsp, include_test.jsp)
1. <code><%@ include file = "header.jsp" %></code> : 외부 jsp 파일 (header.jsp)을 로드다. 2. <code><jsp:include page = "footer.jsp" /></code> : jsp:include 액션태그로 외부 jsp 포함 파일을 로드한다. - 상단/하단 방법의 결과는 동일하나 최근 하단 방법을 많이 사용함.

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 22/81
----	---------------	-------------------	--------------

JSP 선언문, 스크립트릿, 표현식 문법 (jsp_script.jsp)
<p>1. 선언문 :</p> <pre><%! 자료형 변수명 선언; 접근 지정자 자료형 사용자 정의 메서드 () { } %></pre> <p>형식과 같이 주로 변수 선언 또는 메서드 선언 용도로 사용된다. 즉 <%! 라고 시작하면 JSP에서는 선언문이라 한다.</p> <p>2. 스크립트 릿 :</p> <pre><% %></pre> <p>이 영역을 스크립트 릿이라 한다. 스크립트 릿은 자바 문법을 따라 간다. 주석문도 자바 주석문과 같다.</p> <p>3. 표현식: <% = 변수명 %>과 같은 방법으로 <% = 문법을 표현식이라 한다. 주로 표현식은 출력용으로 많이 사용한다.</p>

out.println (); (jsp_script.jsp)
1. out.pritnln (출력될 값); => <%= 표현식 문법과 같이 출력 메서드로 활용된다.

jsp:forward 액션 태그 (forwardTest.jsp, forwardTest01.jsp, forwardTest02.jsp)
<p>1. 다른 페이지로 프로그램의 제어하여 이동할 때 사용된다.</p> <p>2. 페이지의 흐름을 제어하는 것으로 jsp 페이지 내에서 forward 액션태그를 만나면 지정한 페이지로 이동한다. 사용자가 입력한 값에 따라 각 페이지로 이동해야 할 경우에 사용하면 좋다.</p> <p>3. 이 액션 태그를 잘 이해하면 모델 2에서 controller를 훨씬 더 쉽게 이해할 수 있다. 모델 2에서는 controller가 forward 액션 태그와 같이 페이지 흐름을 제어한다.</p> <p>4. 형식) <jsp:forward page = "이동할 페이지명" /></p> <p>주의사항) forward 액션 태그로 페이지를 이동하면 웹 브라우저 주소창에 보이는 주소값과 실제 보이는 본문 내용이 다르다. 이 방식은 모델 2와 같다.</p> <p>5. param 액션 태그를 사용해서 프로그램 제어가 이동할 페이지에 파라미터 값을 전달한다.</p> <p>형식) <jsp:forward page = "이동할 페이지명"></p> <pre><jsp:param name = "paramName01" value = "값" /> <jsp:param name = "paramName02" value = "값" /> </jsp:forward></pre>

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 23/81
----	---------------	-------------------	--------------

<p>Servlet 개념</p> <ol style="list-style-type: none"> 1. 서버 측에서 실행되는 자바 기반으로 이루어진 웹 프로그래밍 언어이다. 2. Sun Microsystems Inc.에서 자바 언어를 이용하여 쉽게 웹 어플리케이션을 개발하기 위해 만든 표준 API (라이브러리, 클래스의 집합)이다. 3. Servlet 클래스는 웹 브라우저 상에서 누구나 접속 가능하게 public 접근 지정자로 선언한다. 4. Servlet 클래스를 만들기 위해서는 HttpServlet Servlet 클래스로부터 상속을 받아야 한다 (Tomcat WAS 파일 내 포함). 5. get으로 접근할 때는 doGet () 메서드를 오버라이딩해서 호출한다. 6. post로 접근할 때는 doPost () 메서드를 오버라이딩해서 호출하면 된다. 7. Servlet 클래스를 호출해서 실행할 때 get 또는 post에 관계없이 실행하려면 service () 메서드를 오버라이딩해서 호출한다. 8. HttpServletRequest Servlet은 사용자 폼에서 입력한 정보를 서버로 가져오는 역할을 한다. 9. HttpServletResponse Servlet은 서버의 가공된 정보를 사용자 웹브라우저에 응답할 때 사용한다. 10. response.setContentType ("text/html; charset = UTF-8"); → 웹브라우저에 출력되는 파일형식과 언어코딩 타입을 설정 11. PrintWriter out = response.getWriter (); → 출력스트림 객체 out 생성 12. request.setCharacterEncoding ("UTF-8") 코드는 폼 태그에서 method = post 방식으로 서버로 넘어가는 한글 자료를 서버에서 받을 때 폰트 유지 역할을 한다. method = get 방식의 경우 폰트 유지 역할을 못한다. 13. method = get 방식으로 한글 자료가 서버로 넘어갈 때 인코딩 되어져서 깨져서 넘어간다. 한글 자료를 서버에서 받을 때 폰트 유지하려면 String클래스의 getBytes () 메서드를 사용하여 원래 언어코딩 타입인 UTF-8로 바꿔주면 서버에서 받을 때 폰트 유지한다. 14. RequestDispatcher Servlet 특징 (GuestCont.java) <ol style="list-style-type: none"> 가. request.setAttribute ("키 이름", 값); 메서드로 키 이름에 저장된 값을 유지하려면 RequestDispatcher를 사용해서 뷰 페이지 이동을 해야 한다. 만약 response.sendRedirect () 메서드로 이동하면 기존 url-pattern 주소값도 잃어 버리고 키 이름에 저장된 값도 잃어 버린다. 나. 로그인 인증할 때 사용하는 session.setAttribute ("키 이름", 값) 메서드로 키 이름에 저장된 값은 RequestDispatcher를 사용하나 response.sendRedirect () 메서드로 이동한다. 이 경우 키 이름에 저장된 값을 잃어 버리지 않는다.

표현 언어 (Expression Language : EL) 출력법 (EL_01.jsp)
\$ {출력할 값}

JSTL 라이브러리 다운로드 순서

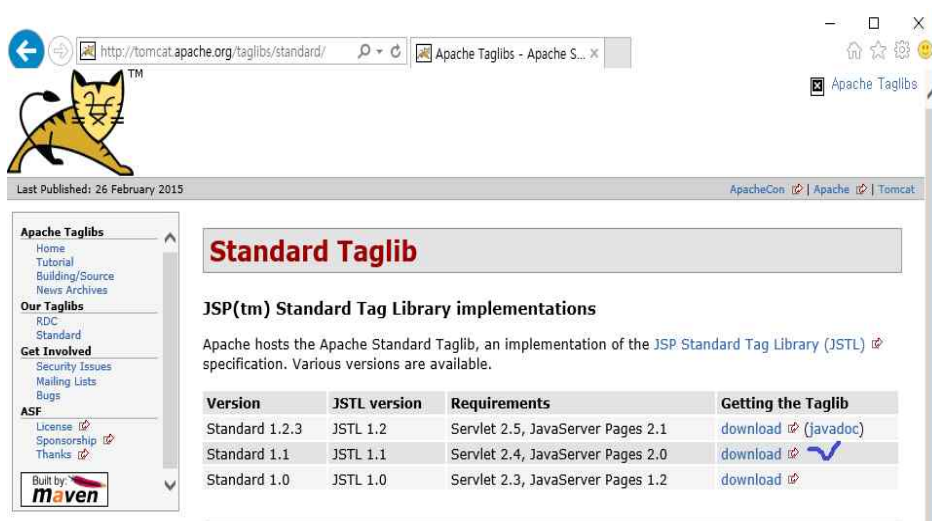
1. <http://jakarta.apache.org> 사이트로 접속한다.
2. 왼쪽 하단에서 6번째 Taglibs 메뉴를 선택한다.



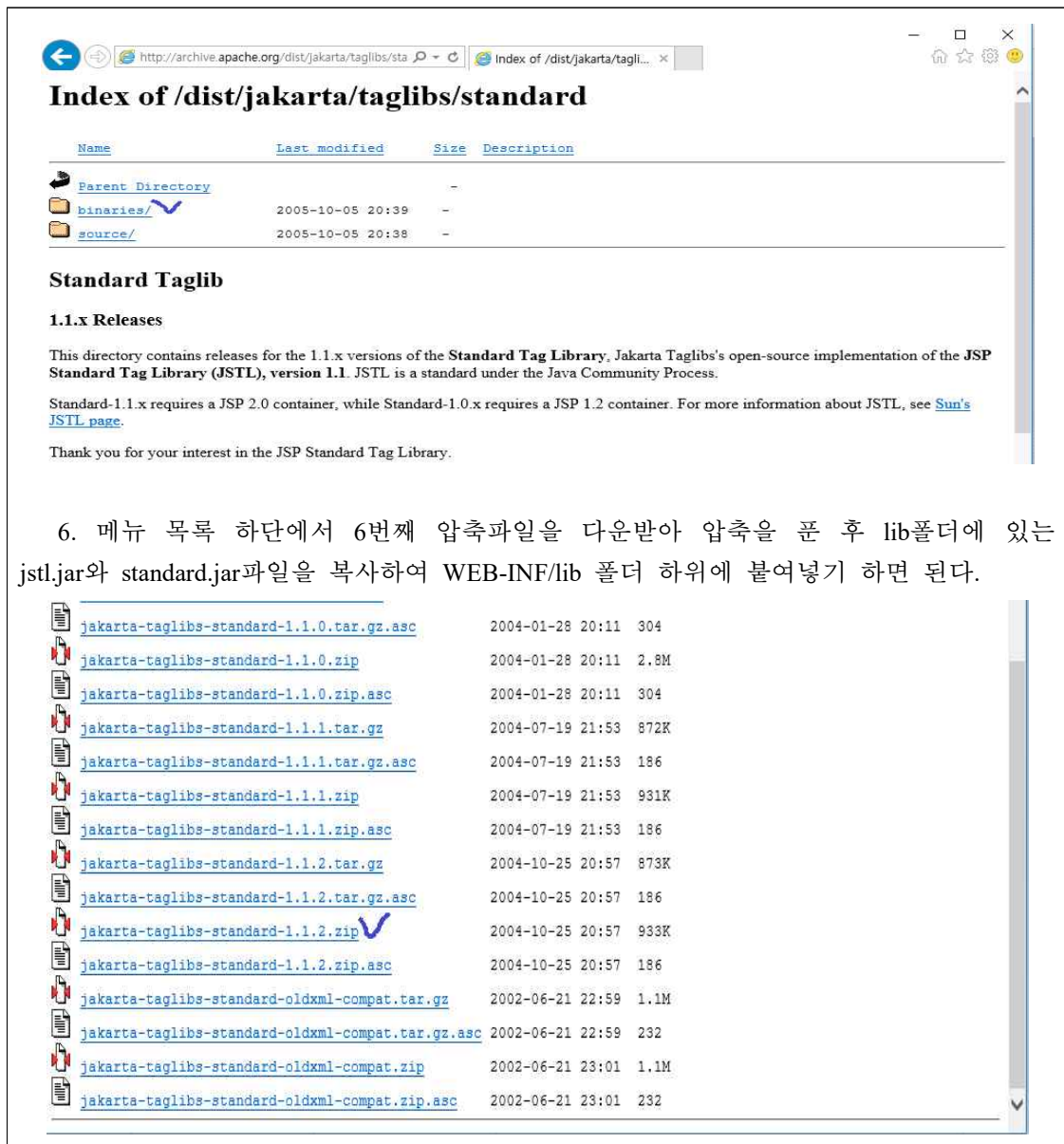
3. Apache Standard Taglib을 선택한다.



4. Standard 1.1에서 download를 선택한다.



5. binaries 폴더를 선택한다.



The screenshot shows a web browser window displaying the 'Index of /dist/jakarta/taglibs/standard' page. The browser's address bar shows the URL 'http://archive.apache.org/dist/jakarta/taglibs/standard'. The page title is 'Index of /dist/jakarta/taglibs/standard'. Below the title is a table with columns: Name, Last modified, Size, and Description. The table lists the following items:

Name	Last modified	Size	Description
Parent Directory		-	
binaries/	2005-10-05 20:39	-	
source/	2005-10-05 20:38	-	

Below the table, the page title 'Standard Taglib' is displayed. Underneath, the section '1.1.x Releases' is shown. The text explains that this directory contains releases for the 1.1.x versions of the Standard Tag Library, Jakarta Taglibs's open-source implementation of the JSP Standard Tag Library (JSTL), version 1.1. JSTL is a standard under the Java Community Process. It also mentions that Standard-1.1.x requires a JSP 2.0 container, while Standard-1.0.x requires a JSP 1.2 container. For more information about JSTL, it refers to Sun's JSTL page. A thank you message follows: 'Thank you for your interest in the JSP Standard Tag Library.'

6. 메뉴 목록 하단에서 6번째 압축파일을 다운받아 압축을 푼 후 lib폴더에 있는 jstl.jar와 standard.jar파일을 복사하여 WEB-INF/lib 폴더 하위에 붙여넣기 하면 된다.

The screenshot also shows a list of files for download, including various tar.gz and zip archives for different versions of the Jakarta Taglibs Standard. The files are listed with their names, last modified dates, and sizes. The 6th file in the list is 'jakarta-taglibs-standard-1.1.2.zip', which is marked with a checkmark.

JSTL 중요한 문법 구조 (JSTL_02.jsp, JSTL_03.jsp, JSTL_04.jsp, JSTL_05.jsp)

1. JSTL 출력법 : `<c:out value = "출력할 값" />`
2. c:if 조건문 형식 :
`<c:if test = "조건식">`
 조건식이 맞으면 실행;
`</c:if>`
 주의사항: else 문은 없다.
3. 다중 조건문 형식:
`<c:choose>`
 `<c:when test="조건 1">조건 1이 만족하면 실행 </c:when>`
 `<c:when test="조건 2">조건 2가 만족하면 실행 </c:when>`
 중략
 `<c:otherwise>해당 사항 없으면 실행</c:otherwise>`
`</c:choose>`
4. 반복문
 형식) `<c:forEach var = "변수명" begin = "1" end = "10" step = "2">`
 1부터 10 까지 2씩 증가하면 홀수 출력
`</c:forEach>`
5. 변수 : `<c:set var = "msg" value = "${'안녕하세요'}">`

web.xml Servlet 설정 형식 (WEB-INF/web.xml)

```

<servlet>
  <servlet-name>Test</servlet-name>
  <servlet-class>model.Test</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>Test</servlet-name>
  <url-pattern>/t</url-pattern>
</servlet-mapping>

```

1. servlet-name 에 지정한 Test는 서블릿 클래스명
2. servlet-class 에 지정한 model은 패키지명, Test는 Servlet 클래스명
3. /t는 웹 주소에서 실행되는 매핑주소.

결론적으로 웹 주소에서 /t 매핑주소가 실행되면 model 패키지의 Servlet Java 클래스 Test가 로드되어져서 웹 상에 실행된다.

cos.jar 다운로드 주소 : www.servlets.com/cos ([member_join.jsp](#))

2. 질의어

가. MySQL & Oracle

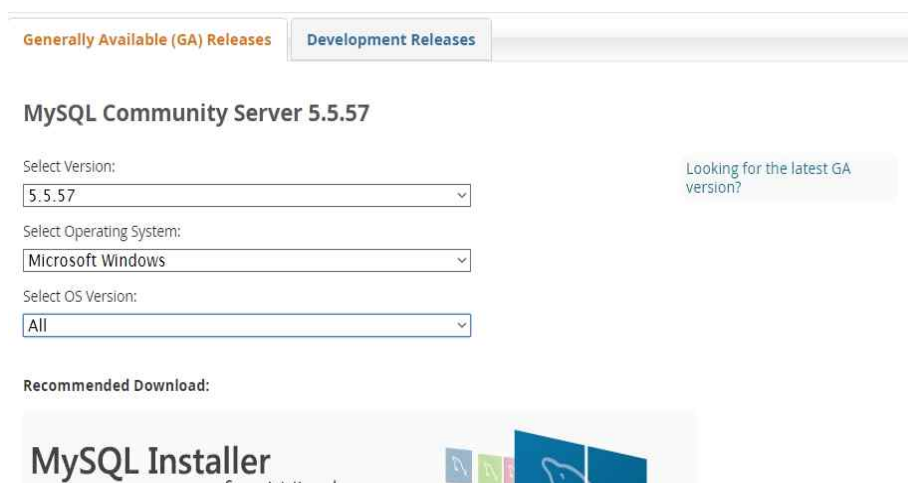
MySQL 설치 및 접속

MySQL 설치

1. 브라우저를 실행하고 <http://dev.mysql.com/downloads/> 사이트로 이동한다. 그리고 [MySQL Community Server]를 클릭한다.
2. 스크롤바를 내리면 아래 화면과 같은 화면이 나온다. 여기서 v표 한 곳을 클릭한다. 이것은 이전 MySQL 버전을 찾는 것이다.



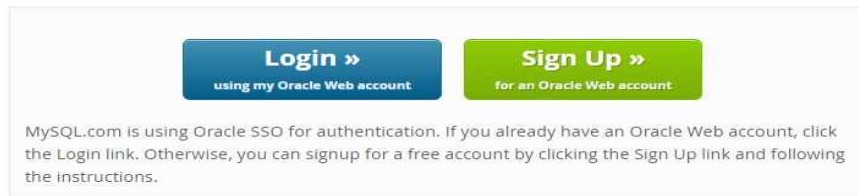
3. 아래 화면에서 Select Version에서 원하는 MySQL 버전, 운영체제, 비트 등을 선택한다.



4. 원하는 MySQL에서 다운로드 버튼을 누르면 로그인 하거나 회원가입하라는 내용

은 무시하고, [NO thanks,just start my download]를 클릭해서 다운로드 받는다.

- Download technical White Papers and Presentations
- Post messages in the MySQL Discussion Forums
- Report and track bugs in the MySQL bug system
- Comment in the MySQL Documentation



 No thanks, just start my download.

MySQL JDBC 드라이버 다운로드

1. 웹 브라우저를 실행하고 <http://dev.mysql.com/downloads/> 사이트로 이동한다. 그리고 [MySQL Community Server]를 클릭한다.
2. [MySQL Connectors] 화면으로 이동하면 왼쪽의 [MySQL Connectors] 의 하위 항목인 [Connector/J]를 선택한다.
3. [Download Connector/J] 화면으로 이동하면 스크롤바를 내려 [Generally Available (GA) Releases] 탭에서 [Platform Independent (Architecture Independent), ZIP Archive]의 [Download] 버튼을 클릭한다.
4. 로그인 하거나 회원가입하라는 내용은 무시하고, [NO thanks,just start my download]를 클릭해서 다운로드 받는다.

다운로드 받은 mysql-connector-java-5.1.43.zip 압축풀기

1. mysql-connector-java-5.1.43.zip 압축파일을 풀면 mysql-connector-java-5.1.43-bin.jar 파일이 생긴다. 이 파일이 바로 MySQL JDBC 드라이버 라이브러리 파일이다.

MySQL 5.7 Command Line Client root 계정으로 접속

1. MySQL 5.7 Command Line Client를 실행하고 비밀번호를 입력한다.
2. mysql> quit를 입력하고 종료한다.

데이터 베이스 생성

1. MySQL 5.7 Command Line Client를 실행하고 비밀번호를 입력한다.
2. create database **bbs**; 입력하면 데이터베이스 생성된다.

데이터베이스 확인

1. MySQL 5.7 Command Line Client를 실행하고 비밀번호를 입력한다.
2. show databases; 입력하면 create database **bbs**;를 입력하면 생성된 데이터베이스를 확인할 수 있다.
3. mysql> quit를 입력하고 빠져 나간다.

사용자 계정 추가와 권한 설정

1. MySQL 5.7 Command Line Client를 실행하고 비밀번호를 입력한다.

2. mysql> grant select, insert, update, delete, create, drop, alter on bbs.* to 'shlee'@'localhost' identified by 'shlee';

이 명령어에서 grant는 권한 설정 명령어이고 shlee 및 shlee는 각각 사용자 계정 및 비밀번호입니다. 그리고 select는 레코드 (데이터베이스에 저장된 한행의 자료 집합) 검색, insert 레코드 저장, update 레코드 수정, delete 레코드 삭제, create 테이블생성, drop 은 테이블 삭제, alter 테이블 수정, localhost는 내 컴퓨터를 의미합니다. 즉 내 컴퓨터에 설치된 mysql 데이터베이스 서버에 접속하겠다는 뜻.

3. mysql> quit

사용자 계정으로 데이터베이스 접속

1. cmd를 실행하고 다음과 같이 입력한다.

mysql -u jspid -p jsptest

2. Enter password:에서 비밀번호인 jsppass를 입력한다. 그러면 mysql> 로 접속한다.

3. mysql> show databases; 라고 입력하면 접근할 수 있는 데이터베이스에 jsptest가 있는 것을 확인할 수 있다.

- 생성된 테이블 확인하기

mysql> show tables;

Oracle 설치 및 접속

Oracle 사용자 생성

1. 오라클 관리자로 접속한다.

콘솔창에서 C:> `sqlplus "/as sysdba"`를 실행한다 (`sqlplus system/manager`).

계정과 패스워드가 없어도 sys 관리자로 접속이 가능하다. 관리자로 접속한 이유는 계정을 만들기 위함이다.

2. 계정에 해당하는 tablespace를 만든다. tablespace를 만들기 위해 파일 경로를 확인해야 할뿐만 아니라 기존에 만들어진 tablespace의 정보를 알아야 한다.

```
SQL> select tablespace_name, bytes, file_name FROM dba_data_files;
```

저장 경로를 확인 후 200 MB의 week 테이블을 만든다.

```
SQL> create tablespace oraclejava
```

```
datafile 'C:\oracle\oradata\XE\xe.dbf' size 200M;
```

참고) `create tablespace xe datafile 'C:\oracle\app\oracle\oradata\XE\xe.dbf' size 200M;` (Ver 11g)

3. 계정 생성

```
SQL> create user week // 계정 생성
```

```
2 identified by week // 패스워드 생성
```

```
3 default tablespace xe // 위에서 생성한 테이블스페이스 명
```

```
4 quota UNLIMITED ON xe; // 테이블 용량 무제한
```

참고) `create user week identified by week default tablespace xe quota UNLIMITED ON xe;`

4. 권한 부여

접속과 기타 기능을 사용할 수 있도록 GRANT를 이용해 할당한다. 권한은 root 개념으로 관리하는데 connect, resource을 할당한다.

```
SQL> conn /as sysdba
```

```
show user
```

```
SQL> GRANT connect, resource TO week;
```

```
connect week/week
```

```
show user
```

나. Oracle

◆ 테이블 구조 명령어

- desc (describe) 테이블명;

◆ 자료형

- number(10)은 정수형 10자리까지 저장한다.
- number(10, 2)은 실수형이다. 즉 소수점 둘째자리까지 포함하여 총 자리수 10자리실수형을 저장한다.
- Date는 날짜형으로써 sysdate 날짜값을 반환하는 함수와 주로 같이 사용한다. MySQL에서는 날짜/시간값을 반환하는 함수로 now() 함수를 사용한다.
- char은 고정 문자형 → 기억장소 낭비의 단점이 있다.
- varchar2은 가변형 문자형 → 기억장소 절약의 장점이 있다.

◆ 동일한 중복 레코드를 한번만 출력

- 형식) select distinct 필드명 from 테이블명; 즉 distinct 예약어를 사용한다.

◆ SQL Plus 편집 명령어

- List (L) : 버퍼의 내용을 나타내기 위한 명령어이다. 약자 (L)로 사용해도 된다. 그리고 바로 이전에 실행했던 쿼리문만 버퍼에 저장된다. 이 버퍼에 저장된 쿼리문을 출력할 경우 List를 사용한다.
- / : 이전에 버퍼에 저장된 쿼리문을 출력하지 않고 바로 실행한다.
- run (R) : 버퍼에 저장된 쿼리문을 출력하고 실행한다. 이 명령어는 “List + /” 랑 같다.

◆ SQL Plus 파일 명령어

- EDIT (ED) : 파일의 내용을 유닉스의 vi나 윈도우의 Notepad++와 같은 편집기를 읽어 사용한다.
 - 형식) EDIT (ED) 파일명
 - 실행했던 쿼리문 마지막 세미클론 (;)이 편집기에서는 /로 대체된다는 점에 주의한다. 저장되는 파일의 확장자는 *.buf 형식이 된다.
- host : Oracle을 종료하지 않고 도스 프롬프트로 종료할 때 사용한다. 다시 sql 프롬프트 돌아올려면 exit을 사용한다.
- save : *.sql 파일로 쿼리문을 저장하고자 할 때 사용한다. 파일명만 기술하고 확장자를 생략하면 기본확장자 sql로 지정된다. save 옵션으로 replace (이미 존재하는 파일에 새로운 내용을 덮어쓰기) 또는 append (이미 존재하는 파일 끝에 최근 실행한 명령어 추가)를 사용할 수 있다.
- @파일명 : 파일명에 저장된 쿼리문을 실행할 때 사용한다. 확장자를 생략하면 *.sql로 인식한다.
- SPOOL : 쿼리문 및 실행 결과 등을 파일에 저장하여 복습하거나 서브 노트를 작성할 때 주로 이용한다. SAVE 명령어는 SQL문을 저장하는 명령어이며 spoolS 명령어는 SQL문과 실행된 쿼리문 결과를 파일로 기록하는 명령어이다. 즉 화면에 보이는 내용을 갈무리해서 하나의 파일로 만든다.
 - 형식) SPOOL 파일이름
 - 기본 확장자는 *.LST이다. spool 해제를 위한 명령어로 SPOOL OFF를 사용한다. 이 명령어는 SPOOL 화면 갈무리 작업을 중단한다. 해제하기 전 까지 여러 쿼리문을 모두 저장한다. SPOOL 명령어를 사용할 때 화면을 갈무리한 내용을 저장하기 위해서는 반드시 SPOOL OFF를 해 주어야 한다. 만약 SPOOL OFF를 하지 않고 Oracle을 종료해 버리면 지금까지 갈무리한 내용이 저장되지 않는다.
- get : save 명령어를 이용하여 저장된 sql 명령어를 다시 불러올 경우 사용되는 명령어이다. 확장자를 기술하지 않으면 기본확장자로 .sql로 인식한다.
 - 형식) get 파일명

◆ LIKE 연산자와 와일드 카드 문자

- LIKE 연산자는 검색하려는 값을 정확히 모를 경우에도 검색할 수 있도록 와일드 카드 (*)와 함께 사용하여 원하는 내용을 검색한다.
 - 형식) 컬럼명 LIKE 형태;
- 와일드 카드 (*)
 - % : 하나 이상의 모르는 임의의 문자와 매핑 대응한다.
 - _ : 단 하나 모르는 문자와 매핑 대응한다.

◆ NULL의 의미

- NULL은 미확정으로 알 수 없는 값을 의미
- NULL을 = 연산자로 판단할 수 없다.
- 특정 필드 값이 NULL 값인지를 비교할 경우에는 IS 연산자를 사용한다. 즉 IS NULL로 판단
- 특정 필드 값이 NULL이 아닌 자료만 추출하기 위해서는 IS NOT NULL 연산자를 사용한다.

◆ 숫자 함수

- ABS 함수 : 절대값을 구하는 함수이다. 주어진 값이 음수인 경우 양수로 표현된다.
- FLOOR 함수 : 소수점 이하를 버리는 함수이다.
- ROUND 함수 : 지정한 자릿수 이하에서 반올림하는 함수이다.
 - 형식) ROUND (대상, 자릿수)
 - 예를 들면) ROUND (34.5678, 2) 하면 두번째 인자값이 2이면 소수점 이하 세번째 자리에서 반올림하여 소수점 이하 두번째 자리까지 표시된다.
- TRUNC 함수 : 지정한 자릿수 이하를 버리는 함수이다.
 - 예를 들면) TRUNC (34.5678, 2) 하면 두번째 인자값이 2이면 소수점 이하 3번째 자리에서 버림하여 소수점 이하 2번째 자리까지 표시된다. 최종적으로 34.56가 된다.
- MOD 함수 : 나눗셈을 하여 몫이 아닌 나머지를 구하는 함수이다.

◆ DECODE 함수 특징

- DECODE 함수는 프로그램 언어에서 가장 많이 사용하는 switch case문과 같은 기능을 한다. 즉 여러 가지 경우에 선택할 수 있게 한다.
 - 형식) DECODE (표현식, 조건1, 결과1,
조건2, 결과2,
조건3, 결과3,
기본 결과 n
)

◆ CASE 함수

1. 여러 가지 경우에서 하나를 선택하는 함수이다.
2. 다중 조건문 (if else)과 같은 기능을 한다.
 - 형식) CASE WHEN 조건1 THEN 결과1
WHEN 조건2 THEN 결과2
WHEN 조건3 THEN 결과3
ELSE 결과 N
END

	제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 34/81
--	----	---------------	-------------------	--------------

◆ 그룹 함수 종류

- SUM (합계), AVG (평균), COUNT (총 레코드 개수), MAX (최대값), MIN (최소값), STDDEV (표준편차), VARIANCE (분산)
- 그룹 함수를 사용하면 comm 필드 보너스 총합을 계산해도 다른 연산자와 달리 null을 제외하고 총합을 구한다. 즉 그룹함수는 null을 제외한다.
- count는 null에 대한 개수를 세지 않는다.

◆ group by문

- 어떤 특정 컬럼 값을 기준으로 그룹 함수를 사용하고자 할 경우 select문 이후에 group by문을 추가하여 해당 컬럼을 기술한다.
 - 예) 부서별 급여 평균을 구하는 경우 사용.

◆ where 조건식과 having 조건식 비교

- where 조건식
 - where절은 테이블에서 데이터를 가져올 때 특정 조건에 맞는 자료만을 검색할 때 사용한다.
- having 조건식
 - 그룹 함수에서 원하는 조건에 맞는 자료만 추출하고자 할 때 사용한다.
 - having문에는 그룹 함수를 적용한 컬럼이 조건식으로 온다.

◆ join

- 하나 이상의 테이블을 서로 합쳐서 데이터를 조회하기 위해서 사용되는 것을 조인(join)이라 한다.

◆ cross join

- 2개 이상의 테이블이 조인될 때 where절에 의한 공통되는 컬럼에 의한 결합이 발생하지 않는 경우를 뜻한다.
- cross join을 하면 테이블 전체 행에 대한 컬럼이 조인된다.

◆ equi join 특징

- equi Join은 가장 많이 사용하는 조인 방법으로 조인 대상이 되는 두 테이블에서 공통적으로 존재하는 컬럼값이 일치되는 행을 연결하여 결과를 생성하는 조인 기법이다.
- 서로 다른 테이블을 조인하려면 일치되는 공통 컬럼을 사용해야 한다. 컬럼 이름이 같아 혼동을 유발할 수 있기 때문에 컬럼 이름 앞에 테이블명을 명시해야 한다.

◆ Non Equi Join 특징

- 조인 조건의 특정 범위 내에 있는지를 조사하기 위해 where 조인 조건으로 = 연산자 이외의 비교 연산자를 사용한다.
- where절의 특정 범위를 급여등급으로 하는 SALGRADE 테이블을 생성한다.

grade필드 (급여등급)	LOSAL (최소급여)	ISAL (최대급여)
1	700	1200
2	1201	1400
3	1401	2000

- where 조건절에서 특정 범위내의 조건으로 검색하는 기법을 NON EQUI JOIN이라 한다.

◆ ANSI Cross Join문

- ANSI Cross Join문은 From절 이후에 쉼표없이 cross join을 사용해서 조인을 하는 방법이다.

◆ ANSI Inner join문

- Ansi Inner join문은 From절 이후에 Inner Join 이라는 명확한 단어를 사용하여 조인할 테이블명을 명시하고 on문에 조인 조건을 명시하는 조인 방법이다.
- 이 조인문은 두 테이블의 동일 컬럼을 기준으로 조인한다.
- using문을 사용한 조인 조건
 - 조인을 정의한 컬럼명이 두 테이블에서 모두 동일하다면 using 절에서 조인할 컬럼을 지정하여 보다 더 간단히 조인을 할수 있다.
 - using문에는 테이블명 또는 별칭을 사용할 수 없다.

◆ Natural Join문

- 조건절을 생략하고 Natural Join을 사용하면 자동적으로 모든 컬럼을 대상으로 공통 컬럼을 조사하여 내부 조인을 실행한다.

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 36/81
----	---------------	-------------------	--------------

◆ ANSI Outer Join 기법

- Outer Join은 어느 한쪽 테이블에는 해당하는 데이터가 존재하는데 다른 테이블에는 데이터가 존재하지 않는 경우 그 데이터가 출력되지 않는 문제점을 해결하기 위해 사용하는 조인 기법이다.
- ANSI Outer Join의 종류
 - LEFT Outer join, Right Outer Join, Full Outer join이 있다.
 - LEFT Outer Join은 From절 이후에 테이블명을 기술할 때 좌측, 우측에 기술하는데 우측 테이블명에 데이터가 없는 경우 사용하는 조인 기법이다.
 - Right Outer Join은 From 절 다음에 테이블명을 기술할 때 좌측, 우측에 기술하는데 좌측 테이블에 데이터가 없는 경우 사용하는 조인 기법이다.
 - Full Outer Join은 Left Outer Join과 Right Outer Join 기법을 결합한 형태이다.
- Outer Join도 inner join처럼 두 테이블 간의 조인 조건에 사용되는 컬럼명이 같다면 On문 대신 Using절문을 사용할 수 있다. 물론 이런 경우도 테이블명, 컬럼명 형태가 아닌 컬럼명만 명시한다.

◆ 서브 쿼리문

- 특정 테이블에서 검색한 결과를 다른 테이블에 전달하여 새로운 결과를 검색하는 경우에 사용한다.

◆ 서브 쿼리문 구조

- `select dname from dept where deptno = (select deptno from emp where ename = 'scott');`
 - 서브 쿼리는 하나의 select 문장 내에 포함된 select문이다. 그렇기 때문에 서브 쿼리를 포함하고 있는 쿼리문을 메인 쿼리라 하고 또 다른 쿼리를 서브 쿼리라고 한다.
 - 서브 쿼리문은 비교 연산자의 우측에 기술해야 하고 반드시 소괄호 내에 넣어야 한다. 서브 쿼리는 메인 쿼리가 실행되기 전 한번만 실행된다.
 - 서브 쿼리는 단일행 서브 쿼리와 다중행 서브 쿼리가 있다.

◆ 단일행 서브쿼리문

- 수행 결과가 오직 하나의 행만 반환된다.
- 하나의 행으로 반환된 결과값을 메인 쿼리로 보내고 메인 쿼리의 where절에는 단일행 비교 연산자를 사용한다.
- 단일행 비교 연산자 종류
 - = (같다), >, >=, <, <=, <> (같지 않다)

◆ 다중행 서브쿼리

- 서브 쿼리에서 반환되는 결과가 하나 이상의 행일 때 사용한다.
- 단일행 서브 쿼리 연산자(=, >, >=, <, <=, <>)를 사용할 수 없고 다중행 연산자를 사용해야 한다.
- 다중행 서브 쿼리 연산자 종류
 - IN 연산자 : 메인 쿼리의 비교 조건에서 서브 쿼리의 출력 결과와 하나라도 일치할 경우 메인 쿼리의 WHERE절이 참이 되는 연산자를 뜻함.
 - ALL 연산자 : 서브쿼리의 모든 결과값 중에서 최대값보다 크면 참이 되는 연산을 뜻한다.
 - ANY 연산 : 서브쿼리의 모든 결과값 중에서 최소값보다 더 크면 참이 되는 연산을 뜻한다.

◆ alter table문

- 기존 테이블 구조를 변경하기 위한 쿼리문이다.
- 기존 테이블에 새로운 컬럼 추가
 - alter table 테이블명
 - add (추가할 컬럼명 자료형 (크기));
- 기존 컬럼 속성 변경
 - alter table 테이블명
 - modify (필드명 자료형(크기));
- 기존 컬럼 삭제
 - alter table 테이블명
 - drop COLUMN 삭제할 컬럼명;
- 테이블 삭제
 - drop table 테이블명;
- 테이블의 모든 레코드행을 제거
 - truncate table 테이블명;
- 테이블 명 변경
 - rename 기존 테이블명 to 신규 테이블명
- 현재 접속중인 사용자로 사용할 수 있는 테이블명을 알고자 할 경우
 - select table_name from user_tables
 - order by table_name desc;

◆ insert문

- 테이블에 새로운 데이터를 저장하기 위해서 사용하는 데이터 조작 쿼리문이다.
 - 형식) insert into 테이블명 (컬럼명, 컬럼명, ...) values (값, 값, ...);
 - 테이블명 이후에 기술한 컬럼 목록 순서대로 values에 지정한 값들이 차례로 저장된다.
 - 테이블명 이후에 컬럼명을 생략하면 테이블 생성시 (create table) 지정한 컬럼명 순서대로 값들이 저장된다.
 - 테이블명 이후에 컬럼 목록 개수와 values 값의 개수가 일치해야 한다.

◆ insert ALL문

- 서브 쿼리의 결과를 조건없이 여러 테이블에 동시에 저장할 경우 사용한다.

◆ insert all when 조건식 then문

- 복수개의 테이블에 다중행 레코드를 저장할 때 when 조건식에 맞는 자료만 저장시킨다.

◆ update문

- 테이블에 저장된 자료를 수정하기 위해 사용하는 쿼리문이다.
- update문 문법구조
 - 형식) update 테이블명 set 필드명 = 변경값, 필드명 = 변경값, ... where 조건식;
 - 조건식 맞는 레코드만 수정
- update문에서 where 조건식을 주지 않으면 모든 레코드가 수정된다.

◆ 서브 쿼리를 사용한 데이터 수정하기

- update 수정문의 set절에 서브 쿼리문을 기술하여 수행한 결과로 자료를 변경한다. 이러한 방법으로 다른 테이블에 저장된 레코드를 이용하여 해당 컬럼값을 수정할 수 있다.
- 문법 형식) update 테이블명 set (컬럼명, 컬럼명, ...) = (서브 쿼리) where 조건식;

◆ merge문

- 합병이라는 의미이며 구조가 같은 2개의 테이블을 하나의 테이블로 합치는 기능을 한다.
- merge 명령어를 수행할 때 해당 테이블에 기존에 존재하는 레코드가 있다면 새로운 값으로 update 되고 존재하지 않으면 새로운 레코드로 insert된다.

<div>제목</div> <div>인터넷 키워드 정보 검색</div>	<div>작성일</div> <div>2020-07-27</div>	<div>페이지</div> <div>39/81</div>
--	--------------------------------------	---------------------------------

◆ delete문

- 테이블에 저장된 데이터를 삭제하는 쿼리문이다.
- 형식) delete from 테이블명 where 조건식;
- delete문에서 where절을 사용하지 않을 경우 테이블에 있는 모든 행이 삭제된다. 그러므로 이 명령어를 신중히 사용해야 한다.
- 주의사항) select 검색시 oracle은 영문 레코드는 영문 소문자를 구분한다.

◆ 트랜잭션

- 트랜잭션 정의
 - 트랜잭션은 데이터 처리의 한 단위를 의미한다. Oracle에서 발생하는 여러 개의 SQL문을 하나의 논리적 작업 단위로 처리하는 것을 말한다.
- commit 정의
 - 모든 sql문을 정상적으로 처리하겠다는 뜻
- ROLLBACK 정의
 - SQL문 작업을 취소하겠다는 뜻.
- AUTO COMMIT (자동 커밋) 되는 쿼리문 정리
 - create (테이블 생성문), alter (테이블 수정문), drop (테이블 삭제문), rename (테이블명 변경문), TRUNCATE (테이블 전체행 삭제문)
- insert,delete,update을 데이터 조작어라 한다. 데이터 조작어는 auto commit이 아니다. 그러나 delete문을 실행하고 commit을 하지 않아도 auto commit되는 create문을 실행시 commit을 사용하지 않아도 delete문까지 auto commit 되어 버린다.
- 데이터 조작어를 실행하고, auto commit되는 쿼리문을 실행할 때 예러가 발생해도 이전 데이터 조작어에 commit이 반영되어 rollback 처리가 안된다.
- SAVEPOINT는 제공하는 기본 트랜잭션 범위를 인위적으로 작게 분할하는 것을 말한다.
- 세이브 포인트에 의해서 지정한 세이브 포인트명까지 이동하려면 rollback to 세이브 포인트명으로 처리 할 수 있다. 그러면 이동한 세이브 포인트명까지 롤백 즉 쿼리문 취소를 할 수 있다.
- 세이브 포인트명 지정하는 형식
 - SAVEPOINT 세이브포인트명;
- 세이브 포인트명으로 지정된 곳 까지 이동 즉 되돌아 가는 형식
 - 형식) ROLLBACK TO 세이브 포인트명;

◆ 트랜잭션과 잠금

- Oracle은 여러명의 사용자가 동시에 하나의 테이블에 접근하여 DML (insert,update,delete) 조작시 특정 사용자가 자원을 독점하지 못하게 하기 위해서 잠금을 발생시킨다.
- 이러한 락을 해제 할려면 트랜잭션을 처리해야한다. 곧 commit, rollback 처리를 해야한다.
- 데드락 (Dead Lock) 정의
 - 서로 락 즉 잠금 상태가 되어져서 무한 대기 상태인 경우를 Dead Lock이라 한다. Oracle에서 dead lock이 걸리면 자원을 사용할 수 없는 상태가 된다.
 - 이러한 데드락이 발생하면 Oracle은 기본으로 Oracle을 사용하지 못하는 상황을 방지하기 위해서 다음과 같은 메시지를 보내면서 비 정상적인 종료를 하게 한다.
 - 메시지) ORA-00600 : 자원 대기중 교착 상태가 검출되었습니다.

◆ TRUNCATE와 delete문의 차이점

- TRUNCATE문의 특징
 - AUTO COMMIT문로서 트랜잭션에 의한 커밋을 처리하지 않아도 삭제 쿼리문이 반영된다.
 - 삭제 후 롤백에 의한 자료 복구가 불가능하다.
 - 전체 행 레코드는 삭제할 수 있지만 WHERE 조건절에 의한 조건에 맞는 레코드만 삭제하는 것은 불가능하다.
- delete문의 특징
 - 실행 후 반드시 커밋 또는 롤백에 의한 트랜잭션 처리를 해야한다.
 - 롤백에 의한 데이터 복구가 가능하다.
 - 전체 행 레코드 삭제도 가능하고 where 조건절에 의한 조건에 맞는 레코드만 삭제하는 것도 가능하다.

◆ 제약 조건

- 데이터 무결성 제약 조건이란 테이블 컬럼에 부적절한 자료가 저장되는 것을 방지하기 위해서 테이블 생성시 각 컬럼에 대해서 정의하는 여러가지 규칙을 말한다.
- 제약조건 종류
 - 기본키 (primary key): 중복 자료를 금지하고 null이 저장되는 것을 방지한다. 유일한 값을 가지고 반드시 자료를 입력해야 한다는 제약조건이다.
 - not null 제약조건 : 컬럼에 반드시 값을 저장해야 한다.
 - 즉 null이 저장되는 것을 방지한다.

◆ 제약 쿼리문

- 제약 조건명, 제약 조건의 유형, 제약 조건이 속한 테이블명을 확인하는 쿼리문
- `select constraint_name,constraint_type, table_name from user_constraints;`
- `constraint_name` 컬럼에는 제약조건명
- `constraint_type`에는 제약 조건 유형
- 제약조건 유형 의미
 - P : 기본키 (Primary key)
 - C : not null + check 제약 조건
 - U : unique (중복값이 없고 null을 허용하는 제약조건)
 - R : 외래키 (Foreign key)
- `table_name`은 제약 조건이 속한 테이블명

◆ unique 제약 조건 특징

- 특정 컬럼에 자료가 중복되지 않게 한다.
- null 저장되는 것은 허용한다.
- 중복 자료에서 null은 체크하지 않는다. 즉 null은 중복을 허용하는 특징이 있다.

◆ CONSTRAINT 키워드로 정의하는 사용자 정의 및 제약 조건명 설정법

- 컬럼 설계시 제약조건 유형을 설정하면 기본으로 제공하는 제약조건명을 자동으로 부여한다.
 - 기본 제약조건명 형식) SYS_숫자번호형식
- 사용자 직접 constraint 키워드로 사용자 정의
 - 제약조건명을 설정할 수 있다.
- 사용자 정의 제약조건명 설정 형식
 - 형식)
 - 컬럼명 자료형(크기) constraint
 - 사용자정의 제약조건명 제약조건유형(not null)
- 사용자 정의 제약조건명 명명 규칙
 - 형식)
 - 테이블명_컬럼명_제약조건유형

◆ CHECK 제약 조건

- check 제약 조건은 조건에 맞는 자료만 저장되게 한다.

	제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 42/81
--	----	---------------	-------------------	--------------

◆ 제약 조건 지정방법

- 컬럼 레벨 지정법 : 컬럼에 직접 제약조건을 지정한다.
- 테이블 레벨 지정법
 - 컬럼을 모두 정의하고 나서 테이블 정의를 마무리 짓기 전에 일괄적으로 한꺼번에 제약 조건을 지정하는 방법이다.
 - 하나의 테이블에 2개 이상의 기본키를 설정하는 것을 복합키라 한다. 이러한 복합키는 반드시 테이블 레벨 지정법으로 정의해야 한다.
 - not null은 컬럼 레벨 지정법으로 직접 컬럼 생성시 제약조건을 정의해야 한다.

◆ not null 제약 조건 추가

- not null 제약 조건을 추가하기 위해서는 add 대신 modify문을 사용한다. 이는 null을 허용한 상태에서 not null로 수정한다는 뜻이 있기 때문에 modify문을 사용한다.
- 형식)
 - alter table 테이블명
 - modify 컬럼명
 - constraint 사용자정의 제약조건명 not null

◆ 제약 조건 제거 형식

- alter table 테이블명
- drop constraint 제약조건명;

◆ 제약조건 비활성화 특징

- 제약 조건을 삭제하지 않고 비활성화하여 잠시 사용을 보류하는 것을 말한다.
- 제약조건 비활성화 형식
 - alter table 테이블명
 - disable constraint 제약조건명;

◆ 제약조건 활성화 형식

- 잠시 비활성화 되어져서 사용이 보류된 제약 조건을 다시 사용할려고 활성화하는 것을 말한다.
- 제약조건 활성화 형식
 - alter table 테이블명
 - enable constraint 제약조건명;

◆ CASCADE 옵션 특징

- 부모 테이블과 자식 테이블 간에 기본키와 외래키 참조 관계가 설정된 경우 부모 테이블의 기본키 제약 조건을 비활성화 시키면 연속적으로 자식 테이블의 외래키 제약 조건까지 함께 한번에 비활성화 시키는 방법이다.

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 43/81
----	---------------	-------------------	--------------

◆ 가상테이블 (뷰: VIEW)

- 실제 테이블에 근거한 논리적인 가상테이블이라 정의할 수 있다.
- 가상테이블 뷰에는 실제 자료가 저장되지 않는다. 마치 테이블 처럼 동일하게 뷰를 통해 실제 테이블의 레코드를 볼 수 있다.
- 가상테이블 뷰를 통해 사용자에게 실제 테이블 사용을 제한할 수 있다.
- 뷰 생성 문법 형식
 - create view 뷰이름
 - as
 - 서브 쿼리문;

◆ select view_name, text from user_views;

- view_name 컬럼에는 생성된 가상테이블 뷰 이름을 확인할 수 있다.
- text 컬럼에서는 create view에 의한 뷰 생성시 as문 다음의 서브 쿼리문이 저장된다. 그러므로 뷰에는 실제 자료가 저장되지 않고 뷰 생성시 작성한 서브 쿼리문이 저장된다.

◆ 가상테이블 뷰를 사용하는 이유

- 뷰를 사용하는 이유는 복잡하고 긴 쿼리문을 뷰로 정의하면 접근을 단순화 할 수 있기 때문이다.

◆ 사용자 생성법

- Oracle DB 사용자를 만들려면 dba 즉 system 관리자로 접속
 - SQL> CREATE USER 사용자명
identified by 비밀번호;
- 계정 (사용자, 비번)을 생성한다.
- 사용자만 생성해서는 오라클에 접속할 수 없다.
- 접속 권한
 - grant create session to 사용자명 :접속할 수 있는 권한을 설정
- 테이블 또는 가상 뷰 테이블 검색 권한 설정
 - grant select on 테이블명 (가상 뷰 테이블명) to 사용자명;

◆ 가상테이블 뷰 삭제 문법

- 형식)
 - drop view 뷰이름;

◆ 가상테이블 뷰 생성 옵션 문법

-
- 1. create or replace view 뷰이름;
- or replace옵션은 존재하지 않는 뷰생성시
- 새로운 뷰를 만들게 하고,기존에 존재하는 뷰인

	제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 44/81
--	----	---------------	-------------------	--------------

<ul style="list-style-type: none"> - 경우는 내용을 수정하게 한다. - - 2. create or replace FORCE view 뷰이름; - 가. FORCE옵션은 기존 테이블이 존재하지 않아도 뷰를 생성하게 한다. - - 나. 이 옵션을 생략하면 기본값이 NOFORCE가된다. - NOFORCE 옵션은 반드시 기존 테이블이 있어야 가상테이블 뷰를 생성할 수 있다. -

◆ 가상 뷰 생성시 사용하는 with check option
<ul style="list-style-type: none"> - 뷰를 생성할 때 조건 제시에 사용된 컬럼값을 변경 못하게 한다.

◆ with read only 옵션
<ul style="list-style-type: none"> - 뷰를 만들면 가상테이블 뷰를 통한 기존 테이블의 어떤 컬럼 레코드값도 수정 못하게 한다.

◆ ROWNUM 특징
<ul style="list-style-type: none"> - ROWNUM은 Oracle에서 내부적으로 제공하는 컬럼명이다. - ROWNUM은 테이블에 레코드 저장시 순차적으로 1씩 증가되는 번호값이 저장된다. 이 컬럼값은 자료가 입력되는 시점에 결정되기 때문에 정렬 검색할 때 바뀌지 않는다. - ORDER BY 정렬에 의해서 재정렬되어 ROWNUM 컬럼값을 변경하고자 할 경우는 새로운 테이블에서 다시 작업을 해야 한다. 이 경우는 테이블의 저장 공간이 필요하기 때문에 가상 뷰 테이블을 사용하면 효과적이다.이유는 가상 뷰 테이블은 저장 공간이 필요 없기 때문이다.

◆ 인라인 뷰
<ul style="list-style-type: none"> - SQL문에서 사용하는 서브 쿼리문로서 FROM절 이후에 마치 테이블처럼 사용된다. - 주 쿼리문의 SELECT문 FROM절 이후에 사용되는 서브 쿼리문을 말한다. - 형식) <ul style="list-style-type: none"> • SELECT ... • FROM ..(SELECT ..) • WHERE 조건식;

<div>제목</div> <div>인터넷 키워드 정보 검색</div>	<div>작성일</div> <div>2020-07-27</div>	<div>페이지</div> <div>45/81</div>
--	--------------------------------------	---------------------------------

<div>◆ 인덱스 특징</div> <div> <ul style="list-style-type: none"> - 인덱스는 보다 더 빠른 검색을 위해서 사용한다. - 기본키 또는 유일키 제약 조건으로서 해당 컬럼에 자동으로 인덱스를 생성해 준다. - set timing on : 인덱스에서 검색 속도를 파악할 수 있다. - 형식) <ul style="list-style-type: none"> • create index 인덱스명 • on 테이블명 (컬럼명); - 인덱스 삭제 문법 - 형식) <ul style="list-style-type: none"> • drop index 인덱스명; </div>

<div>◆ 고유 인덱스 특징</div> <div> <ul style="list-style-type: none"> - 기본키 또는 유일키와 같이 유일한 컬럼에 대해서 생성하는 인덱스를 뜻한다. - 중복 자료가 있는 컬럼에는 고유 인덱스를 설정 못한다. - 형식) <ul style="list-style-type: none"> • create unique index 인덱스명 on 테이블명(컬럼명); </div>
--

<div>◆ 비고유 인덱스 특징</div> <div> <ul style="list-style-type: none"> - 중복된 자료가 저장된 컬럼에 대해서 생성하는 인덱스를 비고유 인덱스라 한다. - 중복자료가 있는 컬럼에는 비고유 인덱스만 설정 가능하다. </div>
--

◆ 시퀀스

- 시퀀스는 중복번호가 없고 null이 발생되지 않는다.
- 시퀀스를 사용할 컬럼은 반드시 기본키 제약 조건과 정수형 타입으로 선언해야 한다.
- 형식)
 - 시퀀스 생성: create sequence 시퀀스명;
 - 시퀀스 생성: create sequence 시퀀스명;
 - 시퀀스 생성시 추가적인 옵션 명령어:
 - increment by 1 : 시퀀스 생성시 1씩 증가되는 시퀀스를 생성
 - start with 1 : 1부터 시작되는 시퀀스를 생성하는 추가 옵션
 - nocache : 이 옵션을 사용하면 시퀀스를 임시 메모리 상에서 사용하지 않겠다는 의미로서 기본값은 20, cache는 메모리에 시퀀스값을 미리 할당하고 minvalue는 최소값, maxvalue는 최대값, nocycle은 기본값으로 최소 또는 최대값에 도달하면 생성 중지, cycle은 증가는 최대값에 도달하면 다시 최소값부터 시작, 감소는 최소값에 도달하면 다시 최대값부터 시작
- 현재 만들어진 시퀀스 이름, 증가값을 확인
 - select sequence_name, increment_by from user_sequences;
 - sequence_name 컬럼값은 시퀀스 이름을 저장
 - increment_by 컬럼값은 각 시퀀스의 증가값에 대한 정보를 저장
- 현재 및 다음값에 대한 시퀀스 명령어
 - CURRVAL : 시퀀스로 부터 현재값을 가져온다.
 - nextval : 시퀀스로 부터 다음값을 가져온다.
- 새로 만든 시퀀스로 nextval을 하지않고 currval로 현재값을 가져오면 오류가 발생한다. 이는 nextval로 새로운 값을 생성하지 않았기 때문이다.
- CURRVAL 특징
 - currval은 nextval 이후에 같은 세션에서만 현재 시퀀스 번호를 가져오는 것이 가능하다.
 - nextval 을 사용하고 난 다음에 그 세션이 종료하기 전까지 currval로 값을 가져올 수 있다.
 - 시퀀스명.currval은 한 session에서만 존재하는 임시값이다.
 - currval은 세션에서 마지막으로 call한 nextval에 의해 리턴된 값에 의해 정의 된다.
 - 만일 세션에서 nextval이 아직 call 되지 않았다면 currval은 정의 되지 않는다.
 - 한 세션에서 nextval을 먼저 해야 currval을 할 수 있다.

◆ 사용자 생성 문법

- 형식:
 - create user 사용자명
 - identified by 비밀번호;

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 47/81
----	---------------	-------------------	--------------

◆ 사용자 권한 설정 형식 - 형식) <ul style="list-style-type: none"> • grant 권한 to 사용자명; - grant 권한 to 사용자명 with admin option; <ul style="list-style-type: none"> • 사용자 권한 설정시 부가옵션으로 with admin option 권한을 설정하면 권한이 설정이 된 사용자가 데이터베이스 관리자 (DBA)가 아니어도 자신이 부여 받은 권한을 다른 사용자에게 부여할 수 있는 권한도 함께 부여된다. - 데이터베이스 관리자가 설정하는 시스템 권한 종류 <ul style="list-style-type: none"> • 사용자 생성 후 grant create session to 사용자; <ul style="list-style-type: none"> · create session 권한을 설정해야 생성된 사용자가 Oracle에 접속할 수 있다. • grant create table to 사용자; <ul style="list-style-type: none"> · 생성된 사용자에게 create table 즉 테이블 생성권한을 할당해야 한다. 그러나 테이블 생성 권한만 주어서는 테이블 생성을 할 수 없다. 따라서 각 사용자에게 테이블 사용 공간인 테이블 스페이스를 할당해야 한다. - 각 객체 즉 테이블을 소유한 사용자가 객체의 모든 권한을 가지고 있다. <ul style="list-style-type: none"> • 객체 권한 설정형식 <ul style="list-style-type: none"> · grant 권한 on 테이블명 to 사용자명;
--

◆ select grantee, table_name, grantor, privilege from user_tab_privs_made; - grantee : 권한이 할당된 사용자 명 - table_name : 각 객체,가상테이블 뷰 이름 - grantor : 권한을 준 사용자명 - privilege : 권한 이름 테이블 select 권한

◆ 사용자에게 부여한 각 객체 권한을 철회하기 위한 명령어 형식 - 형식) <ul style="list-style-type: none"> • REVOKE 권한 ON 테이블명 from 사용자명;
--

◆ grant 권한 on 테이블명 to 사용자 with grant option; - 사용자에게 객체 권한을 with grant option과 같이 부여하면 부여받은 사용자는 그 권한을 다른 사용자에게 부여할 수 있는 권한도 함께 주어진다.

◆ 롤 (ROLE) - 사용자에게 보다 더 간편하게 권한을 부여할 수 있도록 여러 개의 권한을 묶어 놓은 권한의 집합을 뜻한다.

◆ 롤 (ROLE)의 생성 절차

- 롤을 생성하기 위해서 DBA (system)계정으로 로그인
- 롤을 생성
- 형식) create ROLE 롤 이름
- 생성될 롤에 권한을 부여
 - 형식) 부여되는 권한이 시스템권한(create session (오라클 연결 권한), create table,create view)일때는 DBA로 접속에서 부여
 - grant create session,create table,create view to 롤 이름;
 - 형식) 부여되는 권한이 객체 권한일 때는 객체 소유자로 접속해서 부여. grant 객체권한 to 롤 이름;
- 사용자에게 생성될 롤을 부여하는 작업 (DBA로 접속)
 - 형식) grant 롤이름 to 사용자;
- 사용자에게 생성된 롤을 확인 방법
 - 형식) select username,granted_role from user_role_privs;
 - username 컬럼에는 사용자명
 - granted_role 컬럼에는 사용자에게 설정된 롤 이름이 저장

◆ 롤 회수

- 롤 (ROLE) 회수는 특정 사용자에게 해당 롤을 사용할 수 없도록 회수하는 것을 말한다. 해당 롤은 존재하기 때문에 다른 사용자에게 롤을 부여할 수 있다.
- 형식) REVOKE 롤이름 FROM 오라클 사용자명;

◆ 롤 제거

- 롤 자체를 삭제하는 것을 말한다.
- 롤 자체가 제거 때문에 다른 사용자에게 롤을 부여할 수 없다.
- 형식) DROP ROLE 롤 이름;

◆ 저장 프로시저

- 복잡한 쿼리문을 매번 사용할 때 다시 입력할 필요 없이 간단하게 저장 프로시저로 정의해 놓고 호출해서 복잡한 쿼리문에 대한 실행 결과를 얻으려고 할 때 주로 사용한다.
- 저장프로시저를 사용하면 성능도 향상되고 호환성 문제도 해결된다.
- 저장프로시저 사용 문법
 - create or replace procedure sel_board13
 - → or replace는 같은 이름의 저장프로시저를 생성할 경우 기존 프로시저를 삭제하고 새롭게 기술한 내용으로 재생성 하는 옵션이다.
 - → sel_board13은 저장프로시저 이름 (vname out board13.name%TYPE, vtitle out board13.title%TYPE, vcont out board13.cont%type, vnum in board13.num%type)이다.
 - → mode는 매개변수라고 한다. mode 매개변수의 종류는 3가지가 있다. in은 값을 전달받을 때 사용, out는 DB 레코드값을 되돌려 받을 때 사용한다. 즉 출력 결과물을 받을 때 사용한다. inout는 두 가지 목적에 모두 사용할 경우 사용한다.
 - is begin select name, title, cont into vname, vtitle, vcont from board13 where num = vnum;
 - end;
 - /
 - → begin 과 end 사이에 실제 실행할 쿼리문 문장이 들어가면 된다.
- 저장프로시저 만드는 순서 (GuProFind.java)
 - sqlplus로 접속
 - sql> ed 저장프로시저로 작성할 sql 스크립트 파일명 fun02로 입력
 - 메모장이 열리면 저장프로시저 생성 쿼리문을 입력한다.
 - @fun02를 입력해서 저장 프로시저를 생성
 - sql> execute 저장프로시저이름 (전달될값); 으로 실행한다. 이전 실행 명령어를 취소하고 싶으면 트랜잭션의 rollback; 하면 쿼리문이 실행이 취소된다. 반대로 commit; 하면 저장프로시저 실행이 성공적으로 완료된다.

3. 프론트엔드 (Front-end)

가. HTML

HTML이란?

1. HTML(Hyper Text Markup Language)의 약어로 웹 페이지를 기술하기 위한 언어이다. 마크업 언어는 각종 태그를 이용하여 텍스트가 문서의 어디에 해당하는지를 기술하는 것이다.
2. 또 다른 의미로는 웹 페이지 뼈대 구조를 만드는 언어라 할 수 있다.
3. HTML은 태그들로 이루어진 HTML 요소(element)의 형태로 작성된다.
4. 태그란 <title>문서제목</title> 화살 괄호 내에 표현된 단어이다. 이처럼 태그는 일반적으로 쌍태그로 이루어진다. 즉 시작태그와 닫는 태그로 구성되어져 있다.

HTML 문서의 기본구조

```
<!DOCTYPE html>
<html>
<head>
  <title>문서제목</title>
</head>
<body>
  본문내용
</body>
</html>
```

1. HTML 문서는 <html>로 시작하여 </html>로 끝난다.
2. <head></head> 태그는 웹 페이지에 대한 정보를 표시한다.
3. <title></title> 태그는 웹브라우저 좌측 상단에 문서제목을 표시한다.
4. <body></body>태그는 본문 내용을 표시한다.
5. 요소는 시작 태그 + 내용 + 닫는 태그로 구성된다.
6. html 태그는 대소문자를 구분 안한다.
7. <p></p>태그는 문단태그이다.
8.
태그는 줄바꿈한다.
9. html 주석문 기호는 <!-- 주석문 -->
10. <h1>,<h2>,... <h6>까지 있다. 이 태그는 heading을 나타낸다. 즉 신문의 머리기사와 마찬가지로 페이지의 머리글을 의미한다. 주로 본문내용 제목을 지정할 때 많이 사용한다. <h1>이 글자크기가 가장 크고, <h6>이 가장 작다.
11. 태그는 이미지 즉 그림을 삽입한다.
12. 태그는 텍스트를 강조할 때 사용한다.

HTML 태그 종류

1. <pre>태그 : 입력한 그대로 화면에 나타나게 한다.
2. <hr> 태그 : 수평선을 표시
3. 은 한칸의 빈공백
4. 태그는 번호 없는 목록을 만든다. 각 목록 항목은 태그로 표현한다.
5. 태그는 번호 있는 목록을 만든다. 각 목록 항목은 태그로 표현한다.
6. 문자를 하이퍼링크 태그라 한다. href 속성 값으로 지정한 주소 또는 파일로 이동시킨다. 하이퍼 링크의 경우 밑줄뿐만 아니라 마우스 오버 시에 화살표 모양이 손가락으로 변경된다.

Table 태그

1. table은 표 형태의 데이터를 표시하는 데 사용한다.
2. 테이블은 하나의 행을 <tr></tr>로 표현한다.
3. <tr>태그 같은 행 내에서 데이터는 <td></td>안에 표현한다.
4. <th></th>도 같은 행 내에서 열을 표현하는 데 사용한다. 글자를 중앙 정렬하고 진하게 표현.
5. rowspan은 행을 합치고 colspan은 열을 합친다.

iframe 특징

1. iframe은 웹 페이지 안에서 다른 웹 페이지를 표시하고자 할 때 사용한다.
2. iframe은 inline frame의 약자이다. 흔히 광고를 위해 이것을 사용한다.
3. 하나의 웹페이지를 여러 프레임으로 나누어서 각각 다른 문서를 표시하고자 할 때도 사용한다.
4. iframe은 html 4.01부터 도입하였고, 거의 모든 브라우저가 지원한다.

div와 span 특징

1. div는 divide의 약자로서 웹페이지를 논리적인 섹션 영역으로 분리할 때 많이 사용한다.
2. div는 블록 수준 영역이기 때문에 한줄 전체 영역을 차지한다.
3. span은 인라인 요소이기 때문에 자신이 필요한 크기만 차지한다.
4. div는 웹 페이지의 공간을 분할하여 레이아웃을 작성하는 데 널리 사용한다.
5. 과거에는 table을 이용한 레이아웃을 많이 사용하나 현재는 테이블을 레이아웃에 사용하는 것은 올바른 방법이 아니다. 테이블은 단지 표 형태의 데이터를 표시하는 용도로만 웹 표준에서는 사용해야 한다.

input type 속성값

1. text : 한줄 입력박스를 만들어 준다.
2. password : 암호화된 값을 입력할 수 있는 한줄 입력박스를 만든다.
3. radio : 라디오 버튼을 만든다. 라디오 버튼은 단 하나만 선택가능하다.
4. checkbox : 복수 개를 선택할 수 있는 체크박스 버튼을 만든다.
5. file : 컴퓨터에 있는 파일을 선택할 수 있게 한다.
6. image : 이미지 전송 버튼을 만든다.
7. hidden : 사용자에게는 보이지 않지만 값을 서버로 넘길 때 많이 사용한다.
8. submit : 자료를 서버로 보내는 버튼을 생성한다.
9. reset : 입력박스 값을 초기화 하는 버튼을 생성한다.
10. button: 버튼을 생성한다.

입력박스

1. textarea : 여러 줄을 입력할 수 있는 입력박스를 만든다.
2. select : 여러 개의 목록항목을 보여주고 사용자로 하여금 선택하게 한다.

HTML5에서 새롭게 추가된 태그

1. header : 상단영역을 지정할 때 사용한다.
2. nav : 메뉴 구성할 때 주로 사용한다.
3. footer : 하단영역을 지정할 때 사용한다.
4. figure : 이미지나 사진 등을 보여줄 때 사용한다.
5. figcaption : figure와 함께 사용하며 그림에 대한 부연 설명을 할 때 사용한다.
6. audio : 오디오를 재생할 때 사용한다.
7. video : 비디오를 재생할 때 사용한다.

나. CSS

CSS 속성 기능
<ol style="list-style-type: none"> 1. css 주석문 : /* 주석문 */ 2. background-color : 배경색 3. margin-left:auto; margin-right:auto; : div 블록 영역을 좌우 중앙 정렬 4. border:1px solid blue; : 테두선 두께를 1픽셀, solid는 실선, blue는 선색을 파랑 5. a:link : 방문하지 않은 하이퍼링크에 css 효과 6. a:hover : 하이퍼링크에 마우스 오버 시 css 효과 7. a:visited : 하이퍼링크에 1회 이상 클릭한 경우 css 효과 (과거형) 8. a:active : 하이퍼링크에 클릭하는 순간 css 효과 (현재형) 9. font-weight : 글자를 굵게 10. text-align:center; :글자를 중앙 정렬 11. padding : 내부 여백 설정 12. margin : 바깥 여백 설정 13. background-image : 배경 이미지 14. text-shadow : css3에서 새롭게 추가된 속성으로 텍스트에 그림자 효과 15. border-radius : css3에서 새롭게 추가된 속성으로 테두리를 원형 처리 16. box-shadow : css3에서 새롭게 추가된 속성으로 div 박스 모델 그림자 효과 17. border-style : 경계선 속성 18. border-width : 경계선 폭 19. border-color : 경계선 색상 20. list-style:none; : ul li로 구성할 때 생성된 목록 앞에 기호 제거 21. float:left; : div에서 복수개의 div는 수직 정렬이나 이를 수평 정렬로 변경 그리고 ul li 태그로 생성된 수직 정렬 목록을 수평목록으로 바꾼다. 22. clear:both; : div 간의 잘못된 css 간섭 적용을 막아주는 역할을 한다. clear 속성 값에 "both"을 적용하면 float의 left, right 모두 제거된다. 23. display:none; : 웹 영역의 자리를 차지 않고 해당 요소 감추기 24. opacity : 0.0 (투명)-1.0 (불투명) 사이에서 투명도를 조절 25. visibility : 어떤 요소를 보여주기 및 감추기 26. hidden : 웹 영역의 자리를 차지하나 해당 요소는 감추기 27. visible: 해당 요소 보여주기

다. JavaScript & jQuery

1. JavaScript 소개

- 가. 동적인 웹페이지를 만들기 위해서 가장 많이 사용되는 자바스크립트이다.
- 나. 1995년 동적인 웹페이지를 만들기 위해서 네비게이터 웹 브라우저를 만든 넷츠게이프사와 선마이크로 시스템사에서 공동으로 개발된 사용자 스크립트 언어이다.
- 다. 자바 언어보다 훨씬 배우기 쉽고 사용하기 쉽다. 특히 변수의 자료형을 선언할 필요가 없다. 변수에 저장되는 값에 의해서 자료형이 결정된다.
- 라. 모든 명령어가 영문 대소문자를 구분한다.
- 마. 넷츠게이프사의 브랜든 아이크가 개발한 언어이다.

2. Java와 JavaScript 차이점

특징	Java	JavaScript
언어 종류	소스 코드를 컴파일 하여 실행하는 컴파일 언어	브라우저가 소스 코드를 직접 해석 하여 실행하는 스크립트 언어
실행 방식	자바 가상 머신 위에서 실행	브라우저 위에서 실행
작성위치	별도의 소스 파일에 작성	HTML 코드 내에 삽입
변수 선언	변수의 자료형을 미리 선언 해야 함.	변수 자료형을 미리 선언 하지 않음.

3. JavaScript 주석문 기호

- 가. 한줄 주석문 : //
- 나. 한줄 이상 주석문 : /* */

4. JavaScript 산술 연산자

- 가. + (덧셈), - (뺄셈), * (곱셈), / (나눗셈), % (나머지), ++ (증가), -- (감소)

5. 문자열 연결 연산자와 비교 연산자

- 가. + 값이 숫자이면 덧셈을 한다. 문자+문자는 문자열을 연결한다. 특히 숫자 + 문자인 경우는 숫자가 문자열로 치환이 되어져서 결합된 문자열로 반환한다.
- 나. 비교연산자 : == (같다), != (같지 않다), > (~보다 크다), < (~보다 작다), >= (~보다 크거나 같다), <= (~보다 작거나 같다)

1. JavaScript 논리연산자

가. && (논리곱) : 두 개의 조건이 모두 true이면 결과값도 true이다.
 나. || (논리합) : 2 개의 조건중 하나라도 true이면 반환값도 true이다.
 다. ! (부정) : 입력값이 true이면 결과값은 false 이고 반대로 입력값이 false이면 결과값은 true이다.

2. JavaScript 조건식 (if ~ else)

가. if (조건식) {
 조건식이 참이면 실행;
 }
 나. If (조건식) {
 조건식이 참이면 실행;
 } else {
 조건식이 거짓이면 실행;
 }
 다. If (조건식 1) {
 조건식이 1이 참이면 실행;
 } else If (조건식 2) {
 조건식 2가 참이면 실행;
 } else {
 조건식 1,2 모두 거짓일때 실행;
 }

3. JavaScript 다중 조건문 (switch ~ case)

가. switch (조건식) {
 case c01: 문장1; break; // 조건식과 c01이 같으면 문장 1을 실행
 중략
 default: 문장 n; break; // 해당조건이 없으면 실행
 }

4. JavaScript 반복문

가. for 반복문

```
형식) for (초기치; 조건식; 증감식) {
    조건식이 참일동안만 반복;
}
```

나. while (조건식) {

```
    조건식이 참일동안만 반복; 증감식;
}
```

다. do {

```
    조건식이 참일동안만 반복;
    증감식;
} while (조건식);
```

주의사항) 조건식을 마지막에 검사하기 때문에 조건식이 거짓이라도 무조건 한번은 반복한다는 단점이 있다.

1. JavaScript break문과 continue문

가. break문 : 반복문 내에서 break문을 만나면 반복문을 강제 중단한다.

나. continue문 : 반복문 내에서 continue문을 만나면 다음 문장을 실행하지 않고 반복을 위해서 처음으로 돌아가서 반복을 계속한다.

2. JavaScript 함수 선언 형식

가. 함수명은 function 키워드로 정의한다. 즉 임의로 함수명을 정의해도 된다는 뜻이다.

나. 함수명 선언 형식 :

```
function 함수명 ( ) {
    실행문장;
}
```

3. JavaScript 내장함수

가. alert () : 사용자에게 경고창을 만드는 함수이다.

나. confirm () : 사용자에게 어떠한 사항을 알려주고 확인/취소 버튼을 가진 윈도우 화면을 띄운다. 사용자에게 확인을 클릭하면 true를 반환하고 취소를 클릭하면 false를 반환한다. 주로 삭제 유무를 확인할 때 많이 사용한다.

다. prompt () : 사용자에게 어떤 사항을 알려주고 입력할 수 있는 입력박스를 가진 윈도우 창을 만들어 준다. 사용자가 입력한 내용을 문자열로 반환한다.

라. parseInt () : 문자열을 숫자로 변환한다.

<div>제목</div> <div>인터넷 키워드 정보 검색</div>	<div>작성일</div> <div>2020-07-27</div>	<div>페이지</div> <div>57/81</div>
--	--------------------------------------	---------------------------------

4. JavaScript 공지창 띄우는 법

가. 형식)

 window.open (공지창 경로와 파일명, 공지창 이름, 공지창 속성);

 window 객체의 open () 메서드로 새로운 공지창을 띄운다.

나. 생성된 공지창은 close () 메서드로 닫는다.

5. JavaScript location 객체

가. location = “이동할 주소 또는 파일명”; location 객체로 지정된 주소 또는 파일로 이동시켜 준다.

6. JavaScript 변수 선언 법

가. var 키워드로 변수 선언

 var a = 변수에 저장될 값;

나. var 키워드를 생략한 변수 선언:

 b = 변수에 저장될 값;

주의사항) JavaScript는 문장 끝을 의미하는 ; (세미콜론)을 생략해도 된다.

1. jQuery

1. 2006년 존 레식이 발표한 가장 인기 있는 JavaScript 프레임워크 라이브러리이다.
 2. JavaScript 프로그래밍 양을 상당히 줄일 수 있다.
 3. 사용할 경우 www.jquery.com으로부터 jQuery 라이브러리를 다운받아야 한다.
 4. jQuery 대신 \$로 표현할 수 있다.
 5. CSS의 선택자 문법을 사용하여 특정한 요소를 찾아서 선택한다.
 6. 웹페이지에서 JavaScript를 사용하는 것을 보다 쉽게 만들어졌다.
 7. jQuery를 사용하면 JavaScript의 여러 줄을 한 줄로 줄일 수 있다.
- 인터넷을 연결해서 사용하는 방법 (CDM)과 라이브러리를 다운받아서 쓸 수 있음.
 - JavaScript와 jQuery를 이용하여 가독성 좋게 코딩해야 함.

2. jQuery 함수 종류

1. text () : 선택된 요소의 문자를 반환하거나 변경한다 ([jQuery_02.html](#), [jQuery_08.html](#)).
2. html () : 선택된 요소의 html 태그와 문자를 반환하거나 변경할 수 있다 ([jQuery_08.html](#)).
3. val () : 입력박스의 입력값을 반환하거나 변경할 수 있다 ([jQuery_10.html](#)).
4. css () : 요소의 스타일 속성을 반환하거나 변경한다.
5. one () : 이벤트가 발생 시에 한번만 실행한다.
6. attr () : 선택된 요소의 속성값을 가져오거나 변경할 수 있다.
7. show () : 선택된 요소를 화면에 표시하게 한다 ([jQuery_04.html](#)).
8. hide () : 선택된 요소를 화면에서 숨긴다 ([jQuery_01.html](#)).
9. toggle () : 요소가 감추어져 있으면 표시하고 표시되어 있으면 감춘다.
10. slideUp () : 요소를 밀어 올린다 ([jQuery_06.html](#), [jQuery_07.html](#)).
11. slideDown () : 요소를 밀어 내린다 ([jQuery_06.html](#), [jQuery_07.html](#)).
12. append () : 선택된 요소의 끝에 새로운 내용을 추가한다.
13. prepend () : 선택된 요소의 처음에 새로운 내용을 추가한다.
14. after () : 선택된 요소의 뒤에 내용을 삽입한다.
15. before () : 선택된 요소의 앞에 내용을 삽입한다.
16. remove () : 선택된 요소와 함께 자식 요소들도 전부 삭제한다.
17. empty () : 선택된 요소는 그대로 두고 자식 요소들만 삭제한다.
18. addClass () : ([jQuery_11.html](#), [jQuery_12.html](#))
 - 가. 선택된 요소에 css클래스를 적용하는 것이다.
 - 나. 기존 요소의 스타일을 어떤 클래스 스타일로 순식간에 변경할 수 있다.
 - 다. 클래스는 css에서 미리 정의되어 있어야 한다.
19. removeClass () : 선택된 요소로부터 css 클래스를 삭제하는 것이다.
20. width () : 요소의 가로 크기를 반환한다. 여기에는 마진, 패딩, 경계는 포함되지 않는다.
21. height () : 요소의 세로 크기를 반환한다.
22. focus () : 선택된 요소의 포커스를 가진다 ([jQuery_03.html](#)).
23. blur () : 선택된 요소의 포커스를 잃어버린다 ([jQuery_03.html](#)).
24. animate () : 애니메이션 효과를 부여한다 ([jQuery_05.html](#)).

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 60/81
----	---------------	-------------------	--------------

3. Ajax (Asynchronous JavaScript and XML) 개요

가. 비동기식 (서버에 요청한 후 끝나는 것이 아니라 실시간 응답 및 요청하다는 것을 의미)으로 서버와 자료를 교환하는 기술이다.

나. 클라이언트 (사용자)가 서버와 적은 양의 자료를 교환하여 비동기적으로 HTML 페이지를 업데이트 할 수 있다.

다. 전체 페이지를 로드하지 않고 웹 페이지의 일부분만 업데이트 할 수 있다는 의미이다. 따라서 Ajax를 사용하면 빠르고 동적인 대화형 웹 페이지를 만드는 데 유용하다.

- 비동기식은 실시간 검색어 및 최신 뉴스는 특정 영역 내에서 업데이트 되는 것이고 동기식의 경우 전체 화면이 변경되는 것을 의미함. 즉 동기식은 불필요한 데이터까지 모두 바뀌기 때문에 오랜 시간이 소요됨 → 모니터를 장기간 볼 경우 시각 피로 증가

4. jQuery + Ajax 함수

1. load () : 서버로부터 데이터를 로드해서 선택된 요소에 반환된 자료를 출력해준다 ([ajax_load.html](#)).

2. \$.ajax () : get 또는 post 방식의 jQuery Ajax 비동기식 프로그램을 처리할 때 많이 사용한다 ([ajax2.html](#)).

3. \$.getJSON () : get 방식으로 JSON으로 표현된 자료를 읽을 때 많이 사용한다. JSON은 JavaScript Object Notion의 약어로 키, 값 쌍으로 가지는 구조이다 ([get_json.html](#), [item.json](#)).

4. \$.get () : get 방식으로 서버에 저장된 자료를 가져오기 위해서 서버와 통신하는 jQuery Ajax 함수이다 ([item.xml](#), [ajax_get.html](#)).

5. \$.post () : post 방식으로 서버에 저장된 자료를 가져오기 위해서 서버와 통신하는 jQuery Ajax 함수이다 ([ajax_post.html](#), [welcome.jsp](#)).

- 자바스크립트를 이용해서 Ajax 경우는 웹 스크립트 외 모바일 웹 페이지에서 활용하기 어려움 및 코드 라인 증가. 이러한 문제를 해결하기 위해 jQuery를 이용하여 Ajax 함수를 이용함 (코드 라인 감소, 유지보수 용이, 빠르게 비동기식 프로그램 활용성 증대).

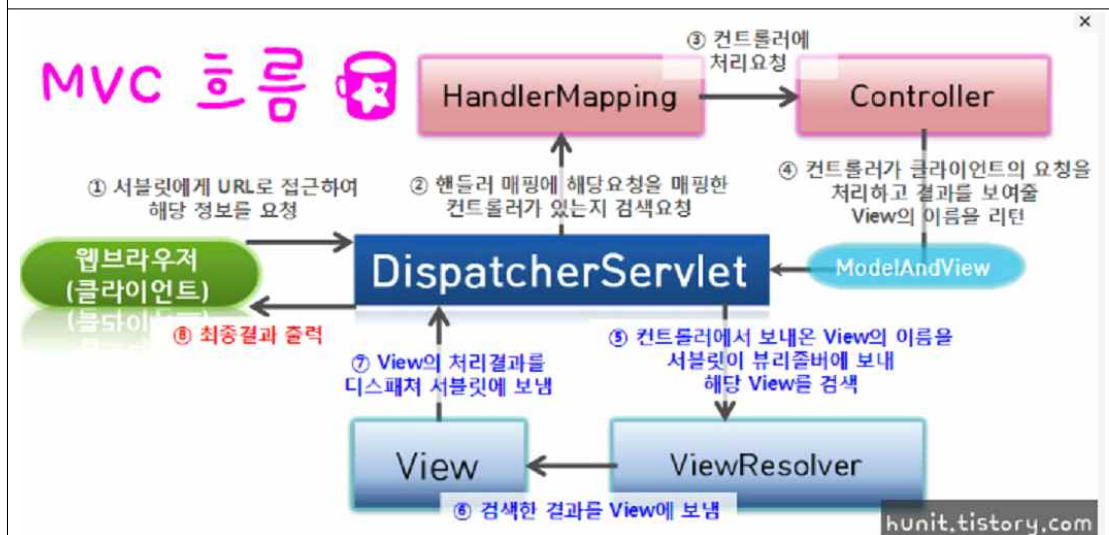
4. 프레임워크 (Framework)

가. 스프링 (Spring)

Spring 개요

1. 웹사이트 개발을 쉽고 편리하게 개발할 수 있도록 지원하는 오픈소스 (공개된 의미) 프레임워크 (외부 라이브러리에 의존)로 경량급 애플리케이션 프레임워크
2. 기존 Servlet MVC Java 언어 개발에 비해서 개발 기간이 단축되고 코드가 간소화되면서 유지 보수가 쉽다. 그러나 Spring은 워낙 프로젝트 분화가 되기 때문에 배우기 어려움.
3. 로드 존슨이 2003년 오픈소스 프레임워크 프로젝트로 개발하기 시작함.
4. MVC 패턴을 따르기 때문에 웹 개발자와 UI 개발자 영역이 철저히 분리됨. 그러므로 유지 보수가 용이하고 웹과 UI 개발자들과의 충돌 염려가 현격히 사라짐.
 - 2010년 중반 이후 Spring과 스트럭처 프레임워크를 사용하기 시작함. 현업에서 사용하다보니 학원 수업 강좌로 과정 개설됨. 2013년에는 다이어믹 프로젝트이나 2014년에 메이븐 프로젝트하다가 작년부터 Spring MVC 패턴을 사용함.

스프링 MVC 구조 (여기서는 MVC에서 컨트롤과 뷰만 표시됨)



- 핸들러 매핑은 매핑 주소를 한 곳에 모아서 관리하는 것
- 뷰리저는 뷰 매핑과 JSP 확장자를 설명해줌
- Help -> Eclipse markplace -> database development install
- STS에서 UTF-8로 변환
- XML 개발환경은 수동 변환 필요

1. Spring Maven과 ibatis 1단계 실습

1. web.xml에서 디스패처 Servlet 설정 (Spring)

```
<servlet>
<servlet-name>sp</servlet-name>
  <servlet-class>
    org.springframework.web.servlet.DispatcherServlet
  </servlet-class>
<servlet-mapping>
  <servlet-name>sp</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

가. url-pattern에 지정한 *.do는 웹주소에서 실행되는 매핑주소

나. 매핑주소가 실행되면 디스패처 Servlet Class에 의해서 Servlet 네임에 지정한 sp에 접미사 -servlet.xml이 결합되어 sp-servlet.xml 파일이 실행된다.

다. sp-servlet.xml은 매핑주소와 스프링 컨트롤 클래스와 모델 dao 영역 파일 내용이 있다.

라. sp-servlet.xml 파일 경로는 src/main/webapp/WEB-INF

2. sp-servlet.xml 파일 내용

```
<beans>
  <bean id = "gListAction" name = "/gu_list.do" class = "action.GListAction">
    <property name = "guService" ref = "guService" />
  </bean>
  <bean id = "guService" class = "dao.GuDAOImpl" />
</beans>
```

가. <bean></bean> xml 태그는 GListAction 컨트롤 클래스 빈 아이디 객체명 gListAction을 생성하게 한다.

나. 매핑주소인 /gu_list.do 가 웹주소에서 실행되면 해당 컨트롤 클래스 빈 아이디가 생성되면서 action 패키지의 GListAction 스프링 컨트롤 클래스가 실행된다.

다. <property name = "guService" ref = "guService" />에서 프로퍼티 네임 guService는 GListAction.java 컨트롤 클래스에서 정의할 dao 패키지의 GuDAO 인터페이스형으로 선언될 참조 변수명이다.

```
private GuDAO guService;
```

라. ref = "guService"는 바로 모델 dao의 빈아이디 명 guService를 가리킨다.

```
<bean id = "guService" class = "dao.GuDAOImpl" />
```

빈 아이디명 guService는 dao패키지의 GuDAOImpl.java의 객체명이다. 우선 dao패키지에 GuDAO.java 인터페이스를 만들고 이 인터페이스를 구현한 자손 클래스 GuDAOImpl.java를 만든다.

마. <property name = "guService" ref = "guService" />를 스프링에서는 setter () 메서드를 통한 DI 의존관계를 주입했다고 표현한다. 위의 XML을 자바 코드로 표현하면 이렇게 표현한다.

```
private GuDAO guService; // 이 부분이 property name = "guService"에 해당한다.
public void setGuService(GuDAO guService) {
    this.guService = guService;
}
```

이는 스프링의 DI 의존관계 설정부분이다. 의존관계를 주입하면 ref = "guService"에서 GuDAOImpl.java의 빈 아이디명 guService를 호출할 수 있다. setter () 메서드를 통한 의존성이 주입되면 this.guService가 생성되며 GuDAO.java 인터페이스를 구현한 GuDAOImpl.java의 오버라이딩 된 메서드를 호출할 수 있는 것이다. 이러한 오버라이딩 된 메서드에서 ibatis 쿼리문 실행 메서드로 guest.xml에 설정된 해당 아이디명을 호출해서 쿼리문을 실행한다.

3. ibatis + mybatis 개념

가. ibatis (mybatis)는 DB에 있는 레코드보다 더 편리하게 xml로 다루기 위한 프레임 워크이다.

나. ibatis (mybatis)도 프레임워크이기 때문에 외부 라이브러리에 의존한다.

다. insert, update, delete, select 쿼리문을 ibatis (mybatis) SQL 매퍼 XML에서 해당 아이디명을 호출해서 쉽게 쿼리문을 실행 시키는 구조이다.

라. ibatis (mybatis)를 쉽게 다루기 위해서 Java 저장 빈 클래스 변수명과 Oracle 테이블 필드명을 같게 해야 한다. 그러면 필드 레코드가 일대일 매핑이 되어져서 쉽게 빈 클래스 변수명에 저장된다.

4. ibatis로 Oracle DB 연결과 SQL 매퍼 xml을 읽어들이는 SqlMapConfig.xml 설정 (경로 : src/main/resources)

```
<sqlMapConfig>
  <transactionManager type="JDBC" commitRequired="false">
    <dataSource type="DBCP">
      <property name="JDBC.Driver" value="oracle.jdbc.OracleDriver" />
      <property name="JDBC.ConnectionURL" value="jdbc:oracle:thin:@127.0.0.1:1521:xe" />
      <property name="JDBC.Username" value="day" />
      <property name="JDBC.Password" value="day" />
    </dataSource>
  </transactionManager>
  <sqlMap resource="guest.xml" />
</sqlMapConfig>
```

가. sqlMapConfig는 ibatis 에서 환경설정 태그이다.

나. transactionManager는 트랜잭션에 관련된 값을 셋팅하고 하위 데이터 소스값을 지정하는 dataSource를 가진다.

다. Spring, ibatis에서 dataSource가 지정되면 보통 DB 연결해주는 부분이 설정된다.

라. type = "JDBC" DB 연결을 하겠다라는 뜻이다.

마. commitRequired = "false"로 설정되어 있는 경우 ibatis에서 내부적으로는 모든 쿼리를 수행하기 전에 conn.setAutoCommit (false) 모드로 동작한다.

바. type = "DBCP"는 DataSource API를 통해 connection pooling 서비스를 제공하기 사용한다.

사. 프로퍼티 네임 JDBC.Driver는 데이터베이스 JDBC 드라이버 패키지명과 드라이버 클래스명을 지정한다.

JDBC.ConnectionURL : 데이터베이스 접속 주소를 지정

JDBC.Username : 데이터베이스 접속 사용자

JDBC.Password : 데이터베이스 접속 비밀번호

5. guest.xml (경로:src/main/resources)

```
<sqlMap>
  <typeAlias alias = "g" type = "model.GuBean" />
  <select id="g_list" resultClass="g"> <!-- resultClass 속성은 반환자료형 타입 -->
    select * from gu14 order by g_no desc
  </select>
</sqlMap>
```

- 가. sqlMap 태그는 ibatis 쿼리문을 설정하겠다는 뜻
- 나. typeAlias 태그는 별칭을 지정할 때 사용
- 다. alias = "g"는 model 패키지의 GuBean 클래스의 객체명이다. GuBean 클래스는 자바 저장 빈 클래스이다.
- 라. select, update, delete, insert 태그는 검색, 수정, 삭제, 저장 쿼리문을 지정할 때 사용
- 마. dao 패키지의 GuDAOImpl.java에서 실행할 ibatis 쿼리문 실행 메서드 정리
 - 첫째. queryForList () 메서드는 하나 이상의 레코드를 검색해서 컬렉션 List로 반환
 - 둘째. queryForObject () 메서드는 단 한 개의 레코드만 검색해서 반환. 이러한 2개의 메서드는 검색 쿼리문 (select)에서만 사용
 - 셋째. insert, update, delete () 메서드는 레코드를 저장, 수정, 삭제할 때 사용
- 바. select 아이디명 g_list는 GuDAOImpl.java에서 쿼리문을 실행할 때 호출된다. 이 아이디명은 유일해야 한다
- 사. ibatis에서 parameterClass 속성은 전달인자 타입을 지정. resultClass 속성은 반환타입을 지정.

2. 스프링 메이븐과 mybatis 2 단계 실습

1. web.xml에서 dispatcher-servlet 설정과 ContextLoaderListener 설정 (스프링)

```

<servlet>
  <servlet-name>naver</servlet-name>
  <servlet-class>
    org.springframework.web.servlet.DispatcherServlet
  </servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>naver</servlet-name>
  <url-pattern>*.nhn</url-pattern>
</servlet-mapping>

<listener>
  <listener-class>
    org.springframework.web.context.ContextLoaderListener
    <!-- WEB-INF/applicationContext.xml 파일을 로드한다. -->
  </listener-class>
</listener>

```

가. Spring에서 지원하는 ContextLoaderListener 클래스에서 의해서 모델 dao 영역 xml을 설정할 WEB-INF/applicationContext.xml 파일을 로드해서 실행한다.

나. 1단계 실습과 비교해서 ContextLoaderListener가 추가된다.

2. naver-servlet.xml 파일 설정 내용 (경로 : src/main/webapp/WEB-INF)

가. SimpleUrlHandlerMapping 클래스로 매핑주소를 일괄적으로 관리하는 것이 추가

나. InternalResourceViewResolver 클래스로 뷰페이지 경로 폴더 설정과 뷰페이지 확장자 설정 부분 추가

```
<beans> <!-- 매핑처리 -->
```

```
<bean id="hm" class="org.springframework.web.servlet.handler.SimpleUrlHandler Mapping">
```

```
<!-- SimpleUrlHandlerMapping 클래스 특징
```

1. 웹주소 매핑처리와 컨트롤 매핑처리를 일괄관리
2. 매핑설정을 한곳에 모아서 관리

```
-->
```

```
<property name="mappings"> <!-- 프로퍼티네임에 설정된 mappings 네임 속성명은 바꿀수 없다. -->
```

```
<props>
```

```
<prop key="/gu_write.nhn">guAction</prop>
```

```
<prop key="/gu_write_ok.nhn">guAction</prop>
```

```
<prop key="/gu_list.nhn">guAction</prop>
```

```
</props>
```

```
</property>
```

```
</bean>
```

```
<!-- 컨트롤작업-->
```

```
<bean id="guAction" class="com.naver.action.GuAction" >
```

```
<property name="gService" ref="gService" />
```

```
</bean>
```

```
<!-- 뷰페이지경로작업-->
```

```
<bean id="it" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
```

```
<!-- InternalResourceViewResolver 기능
```

1. 숨겨진 WEB-INF/jsp/ 폴더의 jsp파일에 접근 가능하게 한다. -->

```
<property name="viewClass">
```

```
<!-- viewClass, prefix, suffix은 바꿀 수 없다. -->
```

```
<value>org.springframework.web.servlet.view.JstlView</value>
```

```
<!-- JSTL를사용해JSP를만드는것을지원해주는클래스가JstlView이다. -->
```

```
</property>
```

```
<property name="prefix">
```

```
<value>WEB-INF/jsp</value>
```

```
<!-- 뷰페이지폴더경로와폴더명지정-->
```

```
</property>
```

```
<property name="suffix">
```

```
<value>.jsp</value>
```

```
<!-- 뷰페이지확장자.jsp설정-->
```

```
</property>
```

```
</bean>
```

```
</beans>
```

3. ContextLoaderListener 클래스에 의해서 로드되는 applicationContext.xml 파일 내용 (경로 : src/main/webapp/WEB-INF)

```
<!-- Data Source -->
```

```
<bean id="dataSource" class="org.springframework.jdbc.datasource.SimpleDriverDataSource">
```

```
<!--
```

1. 스프링이 제공해주는 DataSource 구현 (오라클 데이터베이스 연결) 클래스 중에 간단히 사용할 수 있는 SimpleDriverDataSource 라는 것이 있다. 이 클래스를 사용하면 쉽게 오라클을 연결할 수 있다.

2. 프로퍼티네임 driverClass, url, username, password는 정해진 이름이다. 바꿀 수 없다.

3. driverClass, url, username, password 각 프로퍼티 속성에 value 속성값을 전달한다.

```
-->
```

```
<property name="driverClass" value="oracle.jdbc.OracleDriver" />
```

```
<property name="url" value="jdbc:oracle:thin:@127.0.0.1:1521:xe" />
```

```
<property name="username" value="day" />
```

```
<property name="password" value="day" />
```

```
</bean>
```

```
<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
```

```
<!--
```

1. mybatis SqlSessionFactoryBean 클래스의 빈 아이디 sqlSessionFactory 생성

2. dataSource, configLocation, mapperLocations

바꿀 수 없다. classpath도 바꿀 수 없다.

```
-->
```

```
<property name="dataSource" ref="dataSource" />
```

```
<!-- mybatis에서 dataSource 빈 아이디명을 호출해서 오라클 DB를 연결. -->
```

```
<property name="configLocation" value="classpath:com/naver/mybatis-config.xml" />
```

```
<!--
```

classpath 경로는 target/classes를 뜻함. com.naver 패키지의 mybatis-config.xml 파일을 읽어들임.

```
-->
```

```
<property name="mapperLocations" value="classpath:com/naver/mappers/**/*.xml" />
```

```
<!--
```

1. 이 설정은 com.naver.mappers 패키지 아래와 그 하위패키지를 모두검색해서 마이바티스 매퍼 XML 파일을 모두로드할 것이다.

2. JDBC 와 비교해 보면 마이바티스는 코드를 굉장히 단순하게 만들고 깔끔하게 만든다. 이해하기 쉬워서 유지보수도 편하게해준다.

```
-->
```

```
</bean>
```

```
<bean id="sqlSession" class="org.mybatis.spring.SqlSessionTemplate">
```

```
<!--
```

1. mybatis-spring-1.2.3.jar 가 제공하는 mybatis SqlSessionTemplate클래스로 쉽게 mybatis 쿼리문 실행객체 sqlSession을생성한다.

```
-->
```

```
<constructor-arg index="0" ref="sqlSessionFactory" />
```

```
<!--
```

1.constructor-arg index="0"은 생성자를 통한 의존성을 주입 하는 것이다. 스프링에서는 의존성을 주입하는 방법이 setter () 메서드를 통한 인젝션 의존관계를 만들 수 있고 생성자를 통해서 의존성을 만들 수 있다.

2. index="0"은 생성자의 매개변수 중 첫 번째 전달인자를 가리킨다. 만약 여기에서 <constructor-arg index="0" value="값" /> 라고 xml을 만들었다면 생성자를 통한 의존성을 주입하면서 index="0" 생성자의 첫 번째 전달인자에 value속성값을 저장 즉 전달하게 된다.

```
-->
```

```
</bean>
```

```
<!-- dao설정-->
```

```
<bean id="gService" class="com.naver.dao.GuestBookDAOImpl">
```

```
<property name="sqlSession" ref="sqlSession" />
```

```
</bean>
```

가. mybatis 설정부분 추가

4. mybatis-config.xml 파일 내용 (경로 : src/main/resources/com/naver)

```
<configuration>
  <typeAliases>
    <typeAlias alias = "g" type = "com.naver.model.GuBean" />
  </typeAliases>
</configuration>
```

가. mybatis-config.xml에서 개별적으로 GuBean 클래스의 객체명 g를 관리하기 때문에 편리하다.

나. typeAlias 태그는 별칭을 지정할 때 사용한다.

다. alias 속성 g는 GuBean클래스의 객체명이면서 별칭 이름이다.

라. type속성에는 자바 저장빈 클래스 패키지명 (com.naver.model)과 빈 클래스명 (GuBean)이 들어간다.

5. mybatis SQL 매핑 XML 파일 (경로 : src/main/resources/com/naver/mappers/guestbook/guestbook.xml)

```
<mapper namespace = "Gu">
  <insert id = "g_in" parameterType = "g">
    insert into gul4 values (g14_no_seq.nextval, #{g_name}, #{g_title}, #{g_pwd},
    #{g_cont}, 0, sysdate)
  </insert>
  <select id = "g_list" resultType = "g">
    select * from gul4 order by g_no desc
  </select>
</mapper>
```

가. mybatis에서는 sql문이 들어갈 xml 파일을 mapper 태그로 관리한다.

나. namespace = "Gu" 는 매핑 네임스페이스 이름을 지정

다. mybatis 쿼리문 실행 메서드는 com.naver.dao.GuestBookDAOImpl에서 코드해야 한다.

라. mybatis 쿼리문 실행 메서드

첫째. selectList () 메서드는 하나 이상의 레코드를 검색해서 컬렉션 List로 반환

둘째. selectOne () 메서드는 단일 레코드만 검색해서 반환하며 앞서 2개의 메서드는 select 검색 쿼리문에서만 사용 가능

셋째. insert () 메서드는 레코드를 저장할 때 사용

넷째. update () 메서드는 레코드를 수정할 때 사용

다섯째. delete () 메서드는 레코드를 삭제할 때 사용

마. insert 아이디명 g_in, select 아이디명 g_list는 GuestBookDAOImpl 클래스에서 해당 쿼리문을 실행할 때 호출한다. 중복 아이디명이 있으면 안된다.

바. parameterType 속성은 mybatis에서 전달인자로 넘어오는 자료형을 지정할 때 사용한다.

사. resultType 속성은 mybatis에서 반환 자료형을 지정할 때 사용한다.

```
예) public void updateHit(int no) {
    this.sqlSession.update("gu_hit",no); // gu_hit는 guest.xml에서 지정할 update
아이디명
    } // 조회수 증가
```

모델 DAOImpl 메서드가 void형으로 선언되면 resultType 속성을 사용하지 않는다. 그리고 update () 메서드 인자값으로 no가 넘어가는 경우는 parameterType 속성을 사용한다.

아. #{g_name}은 자바 코드로 표현하면 g.getG_name();와 같다. 즉 getter () 메서드를 호출해서 값을 반환한다.

자. mybatis SQL 매핑 (mapper) XML 태그

첫째. select 는 검색 쿼리문 (select)을 지정할 때 사용

둘째. insert는 저장 쿼리문 (insert)을 지정할 때 사용

셋째. update는 수정 쿼리문 (update)을 지정할 때 사용

넷째. delete는 삭제 쿼리문 (delete)을 지정할 때 사용

6. com.naver.action.GuAction 스프링 컨트롤 클래스에서 사용한 어노테이션

가. @Controller 어노테이션을 사용하면 컨트롤 클래스를 만들 때 implements Controller 인터페이스를 상속받지 않아도 쉽게 컨트롤 클래스를 만들 수 있다.

나. @RequestMapping (value = "/gu_write.nhn", method = RequestMethod.GET)은 get 방식으로 접근하는 매핑주소를 처리하는 리퀘스트 매핑 어노테이션이다.

다. @RequestMapping (value = "/gu_write_ok.nhn", method = RequestMethod.POST)은 post 방식으로 접근하는 매핑주소를 처리하는 리퀘스트 매핑 어노테이션이다.

라. @RequestMapping (value = "/gu_list.nhn")은 get or post에 상관없이 매핑주소를 처리하는 리퀘스트 매핑 어노테이션이다.

JSON를 이용하여 Ajax 연결

- JSON을 이용하지 않고 문자열 객체를 데이터를 빠르게 처리할 수 있음 (SampleController6.java).

- JSON 형태의 키, 값으로 단일 저장 (SampleVO.java, SampleController6.java)

<!-- JSON 라이브러리 추가 -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>2.9.0</version>

</dependency>

- List 형태의 키, 값으로 복수 저장 (SampleVO.java, SampleController6.java)

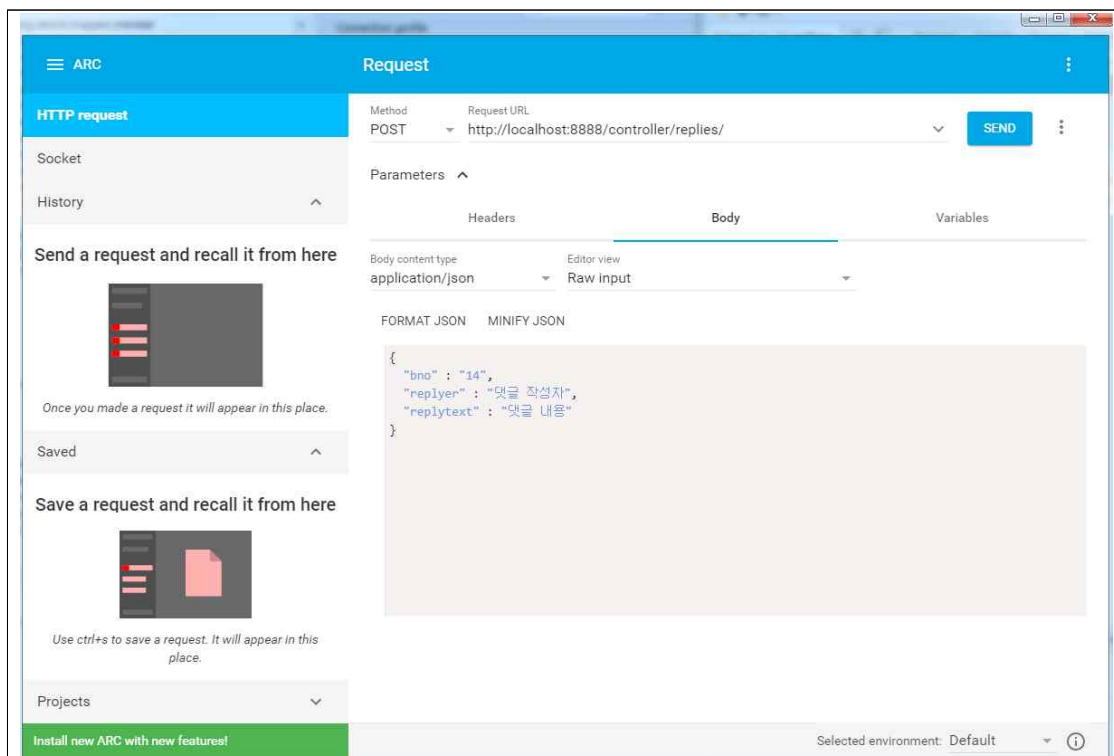
- Map 형태의 키, 값으로 복수 저장 (SampleVO.java, SampleController6.java)

- 400 Error (SampleController6.java)

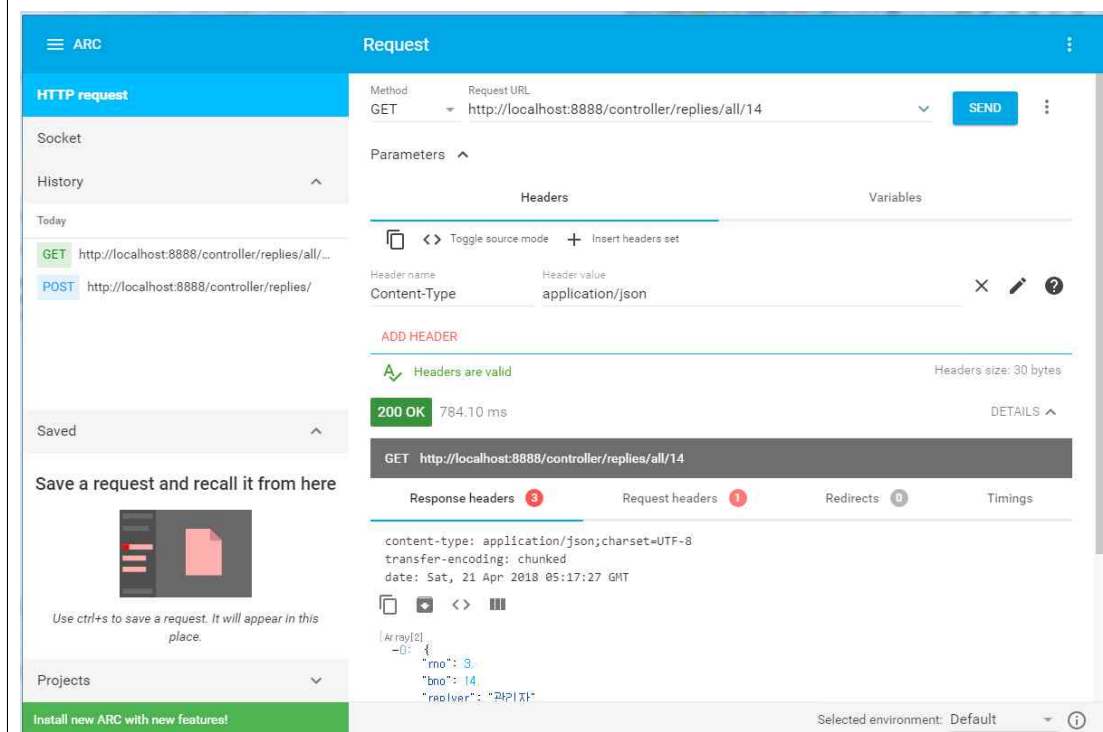
- 정상적인 JSON 데이터와 404 Error (SampleController6.java)

- Advanced REST client (Chrome 웹 스토어 → 윈도우 프로그램)

- 댓글 작성 (tbl_member.sql, ReplyVO.java, ReplyDAOImpl.java, ReplyDAO.java, org.zerock.mappers.reply, reply.xml, mybatis-config.xml, ReplyService.java, ReplyServiceImpl.java, ReplyController.java)



- 댓글 목록 (ReplyController.java, ReplyService.java, ReplyServiceImpl.java, ReplyDAO.java, ReplyDAOImpl.java, reply.xml)



- 댓글 수정 (ReplyController.java, ReplyService.java, ReplyServiceImpl.java,

ReplyDAO.java, ReplyDAOImpl.java, reply.xml)

ARC

HTTP request

Socket

History

Today

PATCH http://localhost:8888/controller/replies/2

GET http://localhost:8888/controller/replies/all/...

POST http://localhost:8888/controller/replies/

Saved

Save a request and recall it from here

Use ctrl+s to save a request. It will appear in this place.

Projects

Install new ARC with new features!

Headers

Body

Variables

Body content type: application/json

Editor view: Raw input

FORMAT JSON MINIFY JSON

```
{
  "replytext": "수정 댓글 내용"
}
```

200 OK 824.20 ms

DETAILS

PATCH http://localhost:8888/controller/replies/2

Response headers: 3

Request headers: 1

Redirects: 0

Timings

content-type: text/plain; charset=ISO-8859-1

content-length: 7

date: Sat, 21 Apr 2018 05:48:14 GMT

SUCCESS

Selected environment: Default

- 댓글 삭제 (ReplyController.java, ReplyService.java, ReplyServiceImpl.java, ReplyDAO.java, ReplyDAOImpl.java, reply.xml)

ARC

Request

Method: DELETE

Request URL: http://localhost:8888/controller/replies/2

SEND

Parameters

Headers

Body

Variables

Body content type: application/json

Editor view: Raw input

FORMAT JSON MINIFY JSON

```
{
  "replytext": "수정 댓글 내용"
}
```

200 OK 667.30 ms

DETAILS

DELETE http://localhost:8888/controller/replies/2

Selected environment: Default

- Ajax 댓글 목록, 수정, 삭제, 닫기 ([HomeController.java](#), [test.jsp](#))

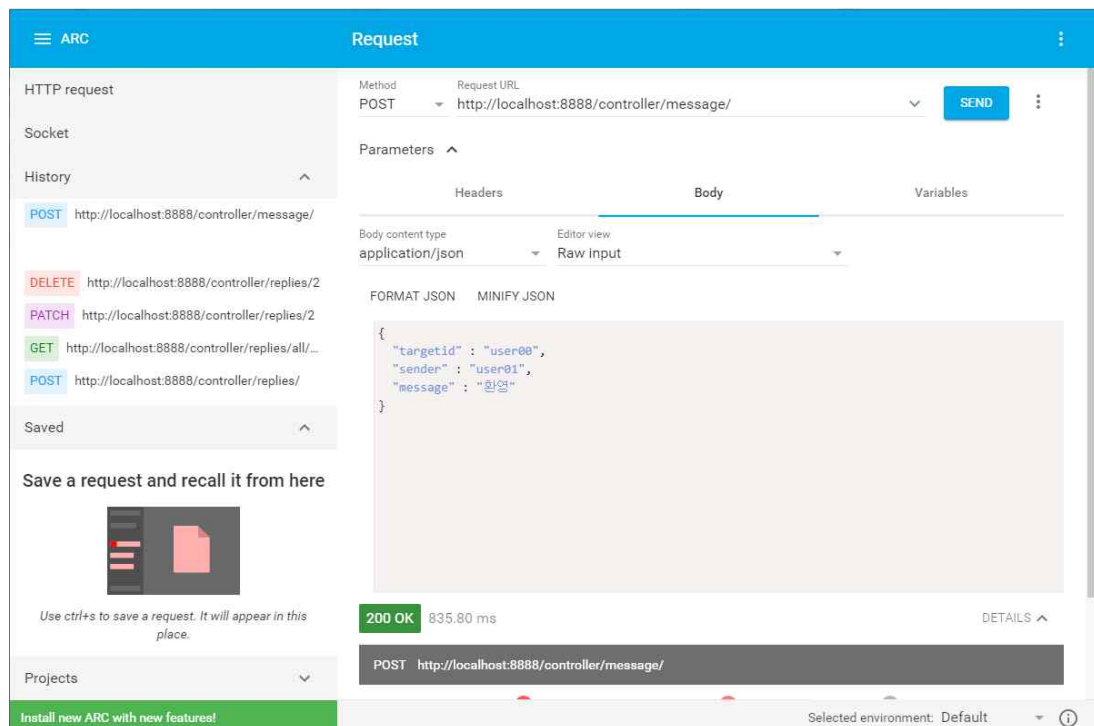
ex00에서 단축보기 → Run As → Run on Server 실행할 경우 HomeController는 get 방식으로 접근할 경우 날짜 포맷으로 /home으로 이동함. ex00/src/main/webapp/WEB-INF/views 경로에 homm.jsp가 호출해줌.

- AOP 자동 프록시, 트랜잭션 설정

(ex00/src/main/webapp/WEB-INF/spring/root-context.xml)

- 테이블 설정 (tbl_member.sql)

- UserVO.java, messageVO.java, PointDAO.java, PointDAOImpl.java, MessageDAO.java, MessageDAOImpl.java, message.xml, Point.xml, MessageService.java, MessageServiceImpl.java, MessageController.java



- 트랜잭션을 위한 코드

회원가입은 되나 포인트가 안될 경우 트랜잭션을 이용하여 롤백해야 됨. 그렇지 않을 경우 일관성이 떨어짐.

<div>제목</div> <div>인터넷 키워드 정보 검색</div>	<div>작성일</div> <div>2020-07-27</div>	<div>페이지</div> <div>76/81</div>
--	--------------------------------------	---------------------------------

5. 기타

가. Eclipse Maven 프로젝트 개발환경 설정

0. Eclipse는 UTF-8로 반드시 변환

1. Eclipse 작업환경 설정

eclipse.exe 실행하면 작업환경 (Workspace) 경로를 week_work로 한다. 새로운 경로를 설정하면 STS (Eclipse) 작업환경이 초기화된다.

2. Eclipse project 삭제 방법

Pivotal tc Server Developer Edition v3.0 → config STS에 있는 기본 웹서버가 설치된다. 이 서버를 STS로부터 삭제한다. 서버를 삭제할 때 반드시 Delete project contents on disk를 체크하고 삭제하세요. 이렇게 삭제해야 STS뿐만 아니라 내 컴퓨터 작업환경에서도 함께 삭제된다.

3. Tomcat 설치 및 로드

<http://tomcat.apache.org>로부터 코어 압축파일을 다운받는다. 다운 받는 경로는 작업환경 경로로 설정하고 압축 파일을 푼다. 그러면 apache-tomcat-9.0.6 폴더가 만들어진다.

4. Tomcat 서버를 STS에 로드

- 가. STS Servers → No servers are available. Click this link to create a new server...
- 나. New Server에서 Apache/Tomcat v9.0 Server → Next
- 다. Tomcat server에서 Tomcat installation directory → Browser → Tomcat server → next → finish

5. STS에서 Java, Java EE, Database Development 메뉴 설정

- 가. Window → Perspective → Open Perspective → Other...

6. STS에서 탐색기 폴더 구조가 동일하게 보는 방법

- 가. Window → Show View → Navigator

7. STS에서 Database Development 메뉴를 통해서 Oracle DB 연결

- 가. Database Development 선택
- 나. Data Source Explorer에서 Database Connections 오른쪽 단축메뉴 → New
- 다. New Connection Profile에서 Oracle 선택 후 Name (Oracle11g) 입력 → Next
- 라. New Connection Profile에서 New Driver Definition 아이콘 클릭 → Name/Type 탭에서 (Oracle Thin | Driver | Oracle 11) 선택, Jar List 탭에서 Driver files → ojdbc14.jar 선택 후 C:\oracle\app\oracle\product\11.2.0\server\jdbc\lib 경로 설정, Properties 탭에서 User ID 및 Password 입력
- 마. New Connection Profile에서 Database instance 변경 (db → xe), Host 변경 (server →

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 77/81
----	---------------	-------------------	--------------

127.0.0.1), User name (오라클 접속 사용자), Password (오라클 접속 비밀번호), Save password 체크, Test Connection 클릭 (Oracle DB 연결 여부 확인 → Ping succeeded!)

8. STS에서 Tomcat server.xml 포트 번호 및 xml 줄번호, 글꼴 설정

- 가. Window → Preferences → General → Editors → Text Editors → Show line numbers
- 나. STS Window → Preferences → XML → XML Files → Editor → Syntax Coloring → Sample text에서 xml → Tag Names → Foreground (글꼴 : 파랑, Bold)
- 다. Sample text에서 version 1.0 선택 → Attribute Names (속성) → Foreground (글꼴 : 빨강, Bold)
- 마. server.xml에서 포트번호 변경
 - 20번줄 : <Server port = "8005" shutdown = "SHUTDOWN"> 포트번호 수정
 - 65번줄 : <Connector connectionTimeout = "20000" port = "8080" protocol = "HTTP/1.1" redirectPort = "8443"/> Http 포트번호 8080에서 다른 포트번호로 수정
 - 85번줄 : <Connector port = "8009" protocol = "AJP/1.3" redirectPort = "8443"/> Connector port 번호를 수정

9. STS에서 글꼴 스타일 및 크기 설정

- 가. Window → Preferences → General → Appearance → Colors and Font → Basic → Text Font → Edit... → 글꼴, 스타일, 크기 설정

10. 자바 클래스, 프로퍼티 파일 언어 타입 설정

- 가. Window → Preferences → General → Workspace → Text file encoding → Other: UTF-8

11. Maven 프로젝트 만들기

- 가. STS Java → Navigator → Maven Project 선택 → New Maven project에서 Next → Select an Archetype에서 Artifact Id를 maven → archetype → webapp 선택 → Next
- 나. Group Id 입력창에서 영문자로 home 입력 (아무거나 입력)
- 다. Artifact Id는 Maven 프로젝트 명 : project (중요) Package 입력창의 home.project는 지운다. 주의할 것은 Maven 프로젝트를 처음 만들 때 반드시 인터넷이 연결되어 있어야 한다. 이유는 외부 인터넷으로부터 Maven 빌딩과정을 다운로드 받아야 한다.

12. Maven 프로젝트 시 JDK 1.5에서 JDK 1.8 수정

- 가. 내 컴퓨터에서 설치된 JDK 버전을 확인 → STS Java → Package Explorer → JRE System Library에서 확인
- 나. JRE System Library → 단축메뉴에서 → Properties → JRE System Library → Workspace default JRE (jdk1.7.0_60) 선택해서 JDK 버전을 수정
- 다. Maven project 단축메뉴 → Properties → Project Facets → Java → 1.5에서 1.7로 수정 → OK

13. Maven 프로젝트에서 Tomcat library import 방법

- 가. Maven 프로젝트 단축메뉴 → Properties → Java Build Path → Libraries → Add

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 78/81
----	---------------	-------------------	--------------

Library → Server Runtime → Next → Apache Tomcat v9.0 → Finish → OK

나. 위의 라이브러리를 import 해야 하는 경우는 간혹 해당 프로젝트의 jsp 또는 servlet java에서 Tomcat 홈 → lib 폴더에 있는 servlet → api.jar 라이브러리를 로드하지 못해 에러가 발생한다. 이 경우에 위의 메뉴를 사용한다. 왜냐하면 바로 이 라이브러리에 있는 servlet을 사용하기 때문이다.

14. 해당 프로젝트에서 JDK 라이브러리를 로드하지 못해 자바 클래스에서 에러가 발생하는 경우

가. STS Project → Properties → Java Build Path → Libraries → Add Library → JRE System Library → Next → Finish → OK

15. Maven project → src → main → webapp → WEB → INF → lib 폴더를 만든다.

16. Maven 프로젝트 폴더구조 설명

- 가. src/main/java : 패키지 이하 원본 *.java 파일이 들어간다.
- 나. src/main/resources : 패키지 이하 mybatis, *.xml 파일이 들어간다.
- 다. src/main/webapp : 폴더, jsp (view page), *.html, *.css, *.js, 이미지 파일 (*.gif, *.jpg, *.png)이 들어간다.
- 라. src/main/java와 src/main/resources 파일은 컴파일 되어서 target/classes으로 들어간다.
- 마. 라이브러리 관리는 pom.xml에서 한다.

17. Maven 프로젝트 src/main/webapp/WEB → INF/web.xml의 배포 서술자 수정

- 가. New Select a wizard (ctrl + N) → Dynamic Web Project → src/main/webapp/WEB/web.xml 내용을 복사해서 INF/web.xml 붙여넣기
- 나. web.xml의 display → name 프로젝트명을 Maven 프로젝트명으로 수정한다.

18. src/main/webapp/index.jsp는 삭제

19. STS에서 크롬 웹 브라우저를 기본으로 설정

- 가. STS Window → Preferences → General → Web Browser → New... → Add External Web Browser 창에서 (Name: chrome | Location: Browse... → C:\Program Files\Google\Chrome\Application\chrome.exe) 설정 → OK
- 나. STS Java EE → Window → Web Browser → chrome를 선택

20. JSP 언어 코딩, 글꼴, 템플릿 설정

- 가. STS Window → Preferences → Web → JSP Files → Encoding: ISO 10646/Unicode(UTF-8) 설정
- 나. STS Window → Preferences → Web → JSP Files → Editor → Syntax Coloring → Sample text page → Tag Names → Foreground (글꼴 : 파랑, Bold)
- 다. Sample text language → Attribute Names → Foreground (글꼴 : 빨강, Bold)
- 라. Sample text java를 → Attribute Values → Italic 해제, Bold 선택
- 마. jsp 템플릿 설정 → Window → Preferences → Web → JSP Files → Editor →

	제목	작성일	페이지
	인터넷 키워드 정보 검색	2020-07-27	79/81

Templates → New JSP File (html) → Preview에서 jsp 내용을 전체 복사 → New... → New Template에서 (Name: new jsp (html5) | Pattern: 붙여넣기 아래와 같이 수정) 설정 → OK

```
<%@ page contentType = "text/html; charset = ${encoding}"%>
<!DOCTYPE html>
<html>
<head>
<meta charset = "${encoding}">
<title> </title>
</head>
<body>

</body>
</html>
```

21. Tomcat에서 server.xml의 Host 사이에 project Maven 프로젝트를 Context path로 등록

가. src/main/webapp 단축메뉴에서 New → Other... → Wizarps 입력창에서 jsp라고 입력 → JSP File → Next → File name을 입력 → Next → Template → new jsp (html5)목록을 선택 → Finish버튼을 클릭

나. Run on Server로 크롬 실행 시 server.xml Host 사이에 Context path로 등록된다.

22. HTML 태그 언어 타입, 글꼴, 템플릿 설정

가. STS Window → Preferences → Web → HTML Files → Encoding: ISO 10646/Unicode (UTF-8)

나. STS Window → Preferences → Web → HTML Files → Editor → Syntax Coloring → DOCTYPE Name or Sample text에서 html 선택 → Foreground (글꼴 : 파랑, Bold)

다. Tag Names or Sample text에서 <html> 선택 → Foreground (글꼴 : 파랑, Bold), Attribute Names or Sample text에서 content 선택 → Foreground (글꼴 : 빨강, Bold), Attribute Values or Sample text에서 text/html 선택 → Italic 해제하고 Bold 선택

라. STS Windows → Preferences → Web → Html Files → Editor → Templates → New HTML File(5) → Edit → Edit Template에서 Pattern를 아래와 같이 설정 → OK

```
<!DOCTYPE html>
<html>
<head>
<meta charset="${encoding}">
<title> </title>
</head>
<body>

</body>
</html>
```

23. CSS 언어 타입, 글꼴 설정

<div>제목</div> <div>인터넷 키워드 정보 검색</div>	<div>작성일</div> <div>2020-07-27</div>	<div>페이지</div> <div>80/81</div>
--	--------------------------------------	---------------------------------

가. STS Window → Preferences → Web → CSS Files → Encoding: ISO 10646/Unicode (UTF-8)

나. Window → Preferences → Web → CSS Files → Editor → Syntax Coloring → URL or Sample text에서 url 선택 → Italic 해제하고 Bold 체크, Type Selector or Sample text에서 BODY 선택 → Foreground (글꼴 : 파랑), Property Name or Sample text에서 color 선택 → Foreground (글꼴 : 빨강, Bold), Property value or Sample text에서 black 선택 → Italic 해제하고 Bold 선택, Id Selector or Sample text에서 #content 선택 → Italic 해제하고 Bold 선택, Class Selector or Sample text에서 .links 선택 → Italic 해제하고 Bold 선택

24. java 클래스 글꼴 설정

가. STS Windows → Preferences → Java → Editor → Syntax Coloring → Element → Java → Classes → Enable (글꼴 : 빨강, Bold)

나. STS Windows → Preferences → Java → Editor → Syntax Coloring → Element → Keywords excluding 'return' → color : 파랑

25. javascript 글꼴 설정

가. STS → Windows → Preferences → JavaScript → Editor → Syntax Coloring → Element → JavaScript → Functions → Enable (글꼴 : 빨강, Bold)

나. STS → Windows → Preferences → JavaScript → Editor → Syntax Coloring → Element → JavaScript → Keywords excluding 'return' → color : 파랑

26. project Maven 프로젝트에 Oracle jdbc 라이브러리 설정

가. 윈도우 탐색기를 실행 → C:\oracle\app\oracle\product\10.2.0\server\jdbc\lib → ojdbc14.jar를 복사 → src/main/webapp/WEB-INF/lib에 붙여넣기

27. 스프링 + mybatis 라이브러리 설정

가. projectMaven → pom.xml을 연다.

나. 크롬에서 www.google.co.kr로 접속한다.

다. 구글 검색 폼에서 spring → webmvc maven으로 검색한다. spring → webmvc는 artifactId이다. 즉 artifactId maven검색한다.

웹에서

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring → webmvc</artifactId>
  <version>4.1.6.RELEASE</version>
</dependency>
```

검색된다. 이것을 pom.xml의 <dependencies> </dependencies> 사이에 붙여넣기 하면 된다. 나머지도 동일방법에 의해서 하면된다.

라. 인터넷이 연결된 상태에서 다운받아 지면서 빌딩이 된다.

마. STS → Java → Package Explorer → Maven Dependencies → 라이브러리가 다운받아져 있다.

제목	인터넷 키워드 정보 검색	작성일 2020-07-27	페이지 81/81
----	---------------	-------------------	--------------

28. Maven 프로젝트가 Tomcat 서버에 의해서 Tomcat 프로젝트로 변경

가. 윈도우 탐색기에서 작업환경 경로 → C:\project_work\metadata\plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps로 이동 → 프로젝트명 → WEB → INF/lib폴더에 가면 pom.xml에서 추가한 라이브러리를 확인 가능하다.

나. C:\project_work\metadata\plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps에서 Maven 프로젝트가 Tomcat 프로젝트로 변경된 것을 확인 가능하다.

29. src/main/webapp/WEB-INF/web.xml 환경설정

가. 스프링 디스패처 서블릿 설정

나. method=post 방식으로 넘어오는 한글 폰트 유지를 위해 xml 설정

다. mybatis xml 설정하는 applicationContext.xml 파일 로드되게 설정