JSP/Servlet

chap8 데이타베이스와 JDBC

데이터베이스 이용한 회원관리시스템 구축하기

학습 목표:

- 자카르타 DBCP API를 이용한 커넥션 풀 설정
- MVC2 모델 기반 회원관리시스템 구축

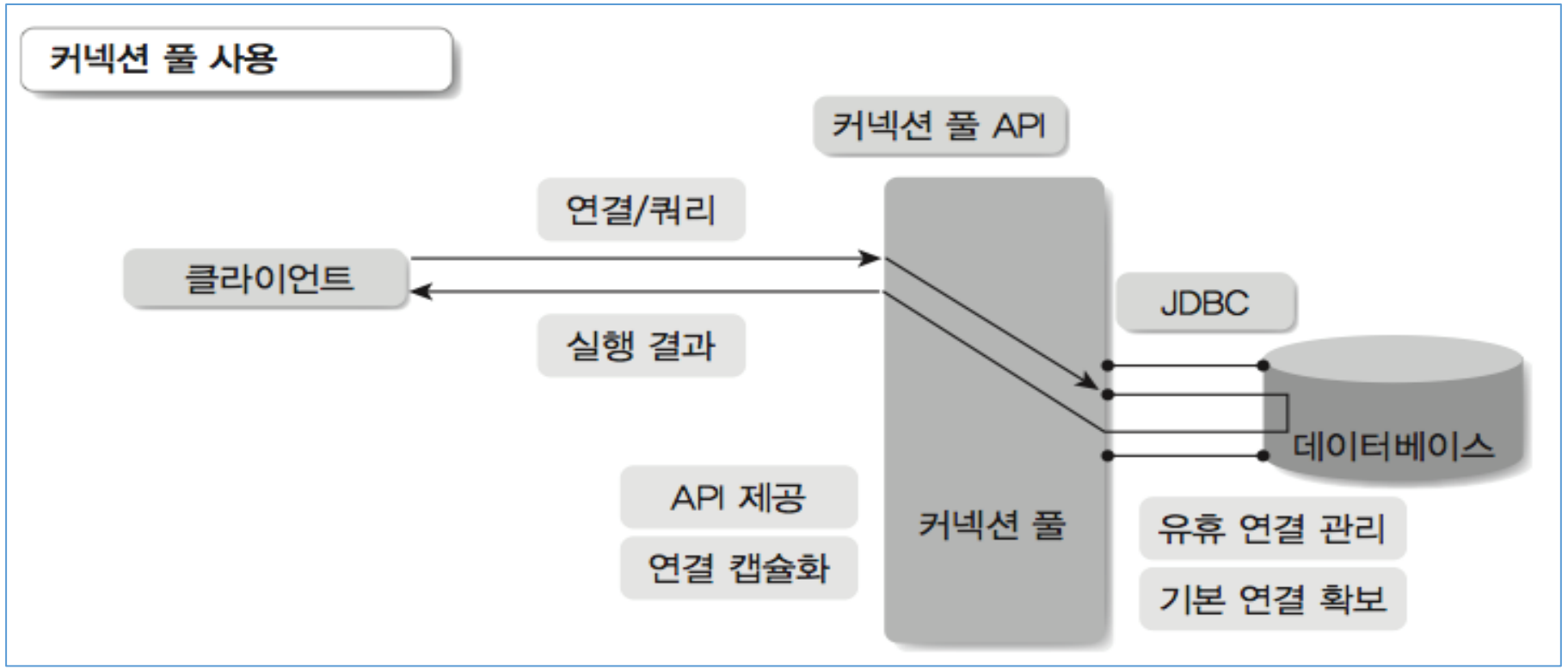
데이터베이스 커넥션 풀(DBCP: DataBase Connection pool)

커넥션 풀

- 애플리케이션에서 필요로 하는 시점에 커넥션을 만드는 것이 아니라 미리 일정한 수의 커넥션을 만들어놓고 필요한 시점에 애플리케이션에 제공하는 서비스 및 관리 체계
- 동시 접속 사용자가 늘어나면 프로그램에서 데이터베이스를 연결하고 종료하는 일련의 과정은 시스템에 많은 부하 발생 시키는 문제를 해결하기 위해 만들어진 전략
  - 커넥션 객체들 생성 후, 커넥션 객체가 필요한 경우 작성한 객체 할당, 사용 후 다시 커넥션 풀로 회수하는 전략을 취함
  - serive()메소드 당 1개씩 할당
  - 커넥션 수 제한함

자카르타 DBCP API를 이용한 커넥션 풀(connection pools) 설정

➢ 커넥션 풀 구조



자카르타 DBCP API를 이용한 커넥션 풀(connection pools) 설정

➢ 커넥션 풀 동작

❶ 웹 애플리케이션 서버가 시작될 때 일정 수의 커넥션을 미리 생성한다.

❷ 웹 애플리케이션 요청에 따라 생성된 커넥션 객체를 전달한다(JNDI 이용).

❸ 일정 수 이상의 커넥션이 사용되면 새로운 커넥션을 만든다.

❹ 사용하지 않는 커넥션은 종료하고 최소한의 기본 커넥션을 유지한다.

자카르타 DBCP API를 이용한 커넥션 풀(connection pools) 설정

■ **커넥션 풀 구현 유형**

유형	설명
직접 구현	개발자가 직접 javax.sql.DataSource 인터페이스를 구현하거나, 직접 새로운 형태의 커넥션 풀을 구현하는 방법이다. 학습을 위한 목적이 아니라면 일반적으로는 권장되지 않는다.
아파치 자카르타 DBCP API를 이용한 구현 (commons-dbcp)	아파치 그룹의 공개된 데이터베이스 커넥션 풀 API인 DBCP를 이용하는 방법이다. 프로그래밍에 자신이 있고 서블릿, 리스너 등의 활용에도 능숙하다면 이를 이용해 자신만의 커넥션 풀을 구성할 수 있다.
애플리케이션 서버 제공	애플리케이션에서 제공되는 커넥션 풀을 사용하는 방법이다. 최근의 웹 애플리케이션 서버들은 javax.sql.DataSource 인터페이스를 따르는 커넥션 풀을 제공하므로 호환에는 큰 문제가 없다. 이 경우 JNDI 네이밍 서비스(Naming Service)를 통해 커넥션 풀을 사용할 수 있다.
프레임워크 제공	애플리케이션 서버와는 별도로 스프링이나 스트러츠 같은 애플리케이션 프레임워크에서 제공하는 커넥션 풀을 사용하는 방법이다. 웹 애플리케이션 이외의 개발에도 사용할 수 있다는 점을 제외하고는 기본적으로 애플리케이션 서버가 제공하는 방식과 다르지 않으므로 개발 성격에 따라 프레임워크에서 제공되는 커넥션 풀을 사용하는 것도 괜찮은 방법이다. 자세한 내용은 해당 프레임워크 개발문서를 참고하도록 한다.

자카르타 DBCP API를 이용한 커넥션 풀(connection pools) 설정

- DBCP API를 사용해서 커넥션 풀을 사용하기 위한 순서

① DBCP API관련 jar파일 설치

② DBCP에 관한 정보 설정 - server.xml

③ JNDI 리소스 사용 설정 - web.xml

④ JSP페이지에서 커넥션 풀 사용

자카르타 DBCP API를 이용한 커넥션 풀(connection pools) 설정

① DBCP API관련 jar파일 설치

http://commons.apache.org/ 접속 후 관련 파일 다운로드

commons-collections-3.2.1.jar, commons-dbcp-1.4.jar , commons-pool-1.6.jar 파일을 톰캣홈\lib 폴더 및 이클립스의 [프로젝트]-[WebContent]-[WEB-INF]-[lib]폴더에 복사

- JDBC 커넥터인 mysql-connector-java-5.1.23-bin.jar파일을

톰캣홈\lib 폴더에 복사

https://commons.apache.org/proper/commons-collections/download_collections.cgi

1. DBCP API관련 jar파일 다운로드 및 설치

1-1.[Collections]: Pool API가 사용하는 자카르타 Collection API의 jar 파일

https://commons.apache.org/proper/commons-collections/download_collections.cgi

[DBCP] : 자카르타 DBCP API 관련 jar 파일

https://commons.apache.org/proper/commons-dbcp/download_dbcp.cgi

[Pool] : 자카르타(Jakarta) DBCP API가 사용하는 자카르타 Pool API의 jar 파일

https://commons.apache.org/proper/commons-pool/download_pool.cgi

1-2. [WebContent] - [WEB-INF] - [lib] 에 JAR DBCP API 관련 jar 파일 배치

2. DBCP에 관한 정보 설정 - server.xml

- 실제 서비스 환경
  - 톰캣홈\conf안에 있는server.xml, context.xml
- 이클립스 가상환경
  - [Servers]-[Tomcat v9.0 Server ~]안에 있는 server.xml, context.xml
- 설정 위치
  - <GlobalNamingResources>엘리먼트의 하위 엘리먼트로 설정
  - <Context>엘리먼트 하위의 하위 엘리먼트로 설정

DBCP에 관한 정보 설정(server.xml)

```
<GlobalNamingResources> --> 톰캣에서 전체 서버를 위한 글로벌 리소스들을 정의함
  <Resource name="jdbc/mysql"
  auth="Container"
  type="javax.sql.DataSource“    →웹에서 이 리소스 사용시 Datasource 타입으로 리턴됨
  maxWaitMillis="10000"
  username="root"
  password="test1234"
  driverClassName="com.mysql.jdbc.Driver"
  url="jdbc:mysql://localhost:3306/memberdb"/>
</GlobalNamingResources>
```

DBCP에 관한 정보 설정(context.xml)

```
<Resource name="jdbc/mysql"
auth = "Container"
driverClassName = "com.mysql.cj.jdbc.Driver"
url= "jdbc:mysql://127.0.0.1:3306/memberdb?serverTimezone=UTC"
username= "root"
password= "test1234"
type="javax.sql.DataSource"
maxActive="20"
maxWait="10000"
/>
</Context>
```

JNDI 리소스 사용 설정 - web.xml

- server.xml에 저장된 JNDI 리소스를 사용하려면 WEB-INF/web.xml 문서에 <resource-ref>엘리먼트 기술

```
  <resource-ref>
    <description>member db</description>
    <res-ref-name>jdbc/mysql</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
  </resource-ref>
</web-app>
```

JSP 코드

JSP페이지에서 커넥션 풀 사용

1. InitialContex객체 생성

   - Context initCtx = new InitialContext();

2. (Context) initCtx.lookup("java:comp/env")에서“”안에 기술된 이름을 lookup()메소드를 사용해서 찾음

   - Context envCtx = (Context) initCtx.lookup("java:comp/env");

JSP 코드

JSP페이지에서 커넥션 풀 사용

3. Context객체의 lookup 메소드 이용하여 "jdbc/mysql“ 관련 객체를 리턴. 이 때 DataSource 객체 타입으로 형변환함

- DataSource ds = (DataSource)envCtx.lookup("jdbc/mysql");

4. DataSource 객체의 getConnection() 이용하여 커넥션 풀로부터 커넥션 객체 할당 받음

- Connection conn = ds.getConnection();

DBCP 연동 테스트1

```
<%
    try{
        Context initContext = new InitialContext();
        Context envContext = (Context) initContext.lookup("java:/comp/env");
        DataSource ds = (DataSource) envContext.lookup("jdbc/mysql");
        Connection conn = ds.getConnection();
        if(conn!=null){
            out.println("DBCP 연동 성공");
        }else {
            out.println("DBCP 연동 실패");
        }
    }
    catch(Exception e){
      e.printStackTrace();
    }

%>
```

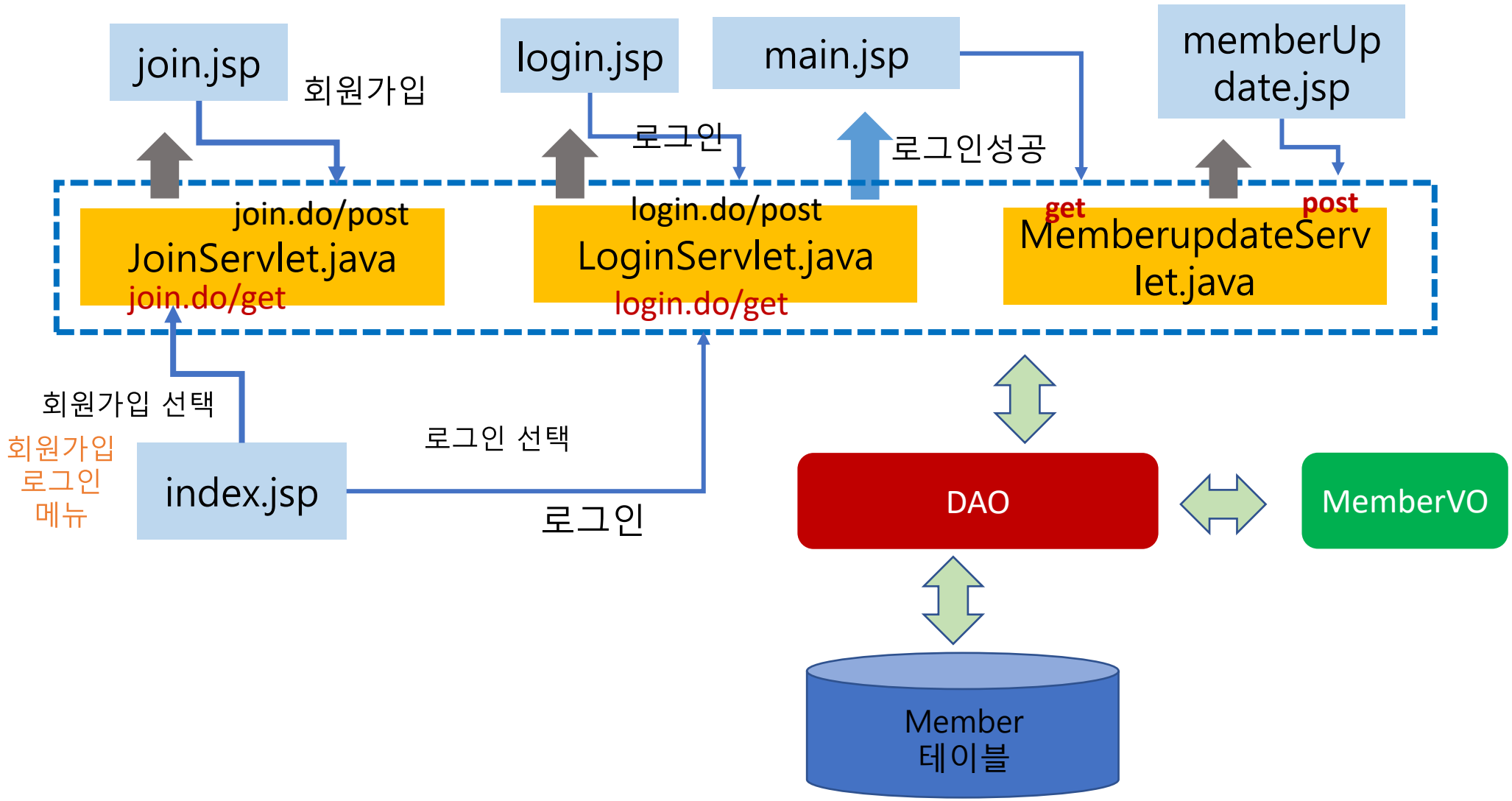
DBCP 연동 테스트1

allMember_dbcp.jsp

DBCP 이용한 회원 정보 보기

이름	아이디	암호	이메일	전화번호	권한(1:관리자, 2:일반회원)
홍길동	hkdong	1234	kdhong@naver.com	010-2344-1234	0
김윤승	light	1234	yoon100@naver.com	010-8978-2124	0
하상오	sang12	1234	ha12@naver.com	010-5678-1212	1

회원관리코드 설치 및 분석



9장 DBCP 테스트 및 회원관리코드.zip 자료 중

회원 관리 시스템 코드 실행 & 분석 & 업그레이드

회원 가입

'*' 표시 항목은 필수 입력 항목입니다.

이름 강철수 *

아이디 chsoo * 중복 체크

암 호 •••• *

암호 확인 •••• *

이메일 kcs@naver.com

전화번호 010-2345-2312

등급 ◉ 일반회원 ○ 관리자

확인 취소

join.jsp (실행은 join.do)

회원가입 후
바로 로그인 화면은 이동
(session에 userid 저장)

로그인

아이디 chsoo

암 호

로그인 취소 회원 가입

회원 가입에 성공했습니다.

login.jsp

회원 관리 시스템 코드 실행 & 분석 & 업그레이드

회원 가입

'*' 표시 항목은 필수 입력 항목입니다.

이름 강철수 *

아이디 chsoo * 중복 체크

암 호 •••• *

암호 확인 •••• *

이메일 kcs@naver.com

전화번호 010-2345-2312

등급 ◉ 일반회원 ○ 관리자

확인 취소

join.jsp (실행은 join.do)

회원가입 후
바로 로그인 화면은 이동
(session에 userid 저장)

로그인

아이디 chsoo

암 호

로그인 취소 회원 가입

회원 가입에 성공했습니다.

login.jsp

회원 관리 시스템 코드 실행 & 분석 & 업그레이드

로그인 성공할 경우
(session에 userid 저장)

msin.jsp

회원 전용 페이지

안녕하세요. 강철수(chsoo)님

로그아웃 회원정보변경

세션무효화

로그인

아이디
암 호
로그인 취소 회원 가입

회원정보변경

회원 수정

이름	강철수
아이디	chsoo
암 호	*
암호 확인	*
이메일	kcs@naver.com
전화번호	010-2345-2312
등급	일반회원 관리자

확인 취소

MVC란?

모델
Model

뷰
View

컨트롤러
Controller

MVC란?

모델
Model

어플리케이션의 정보, 데이터

사용자가 보게 될 결과 화면 출력

뷰
View

컨트롤러
Controller

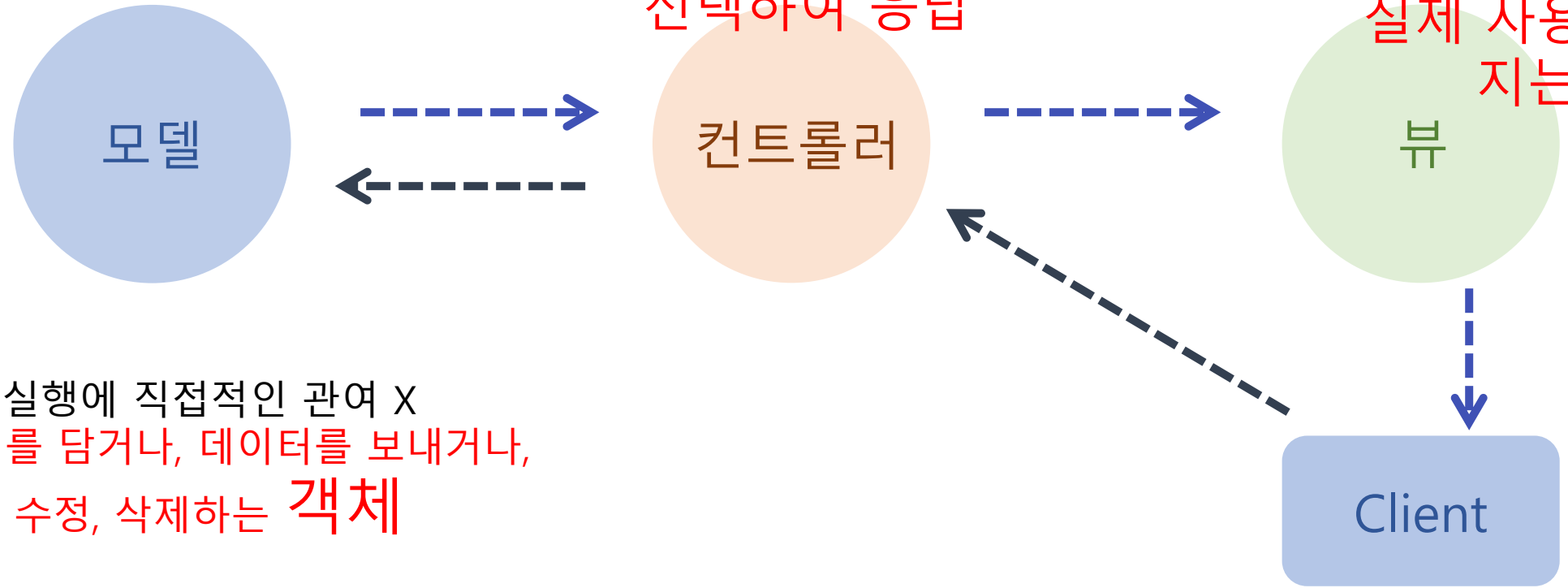
사용자의 입력처리와 흐름 제어 담당

MVC란?

세 영역으로 구분하고, 영역간 결합도를 최소화

어플리케이션의 행위를 정의
Logic 결과에 따라 적절한 View를
선택하여 응답

Model 데이터를 기반으로
실제 사용자에게 보여
지는 페이지



실행에 직접적인 관여 X
데이터를 담거나, 데이터를 보내거나,
수정, 삭제하는 객체

MVC

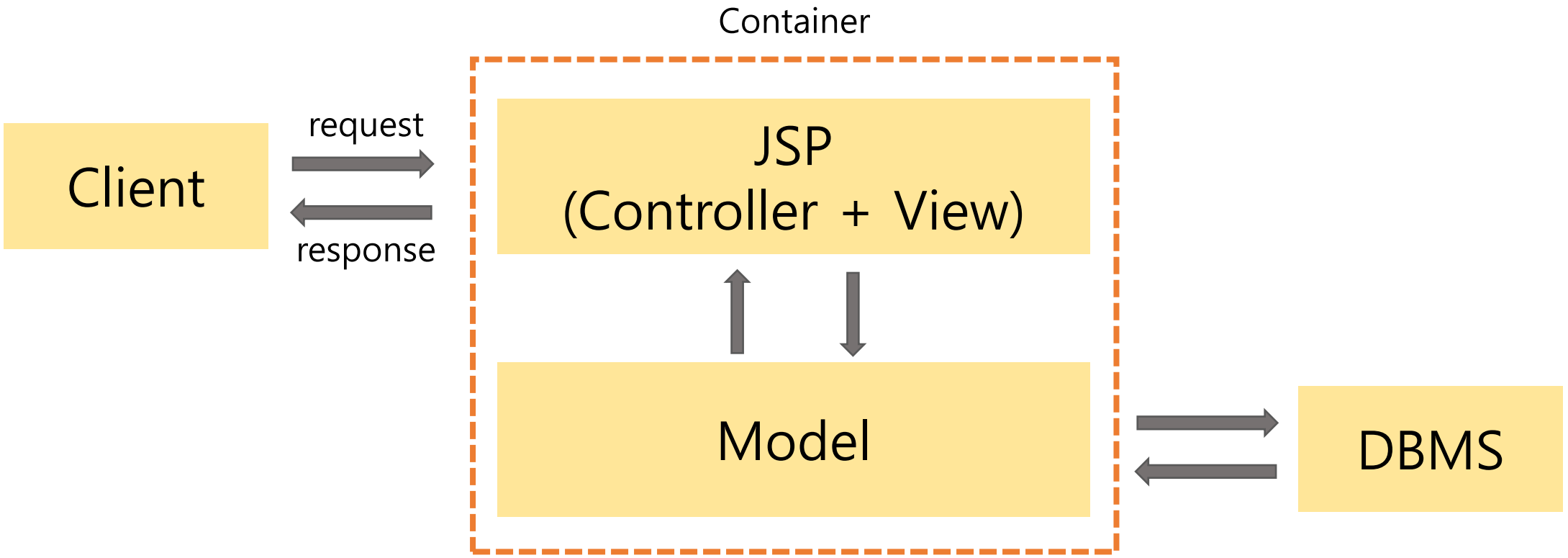
Model1

JSP에서 출력과 로직을
전부 컨트롤

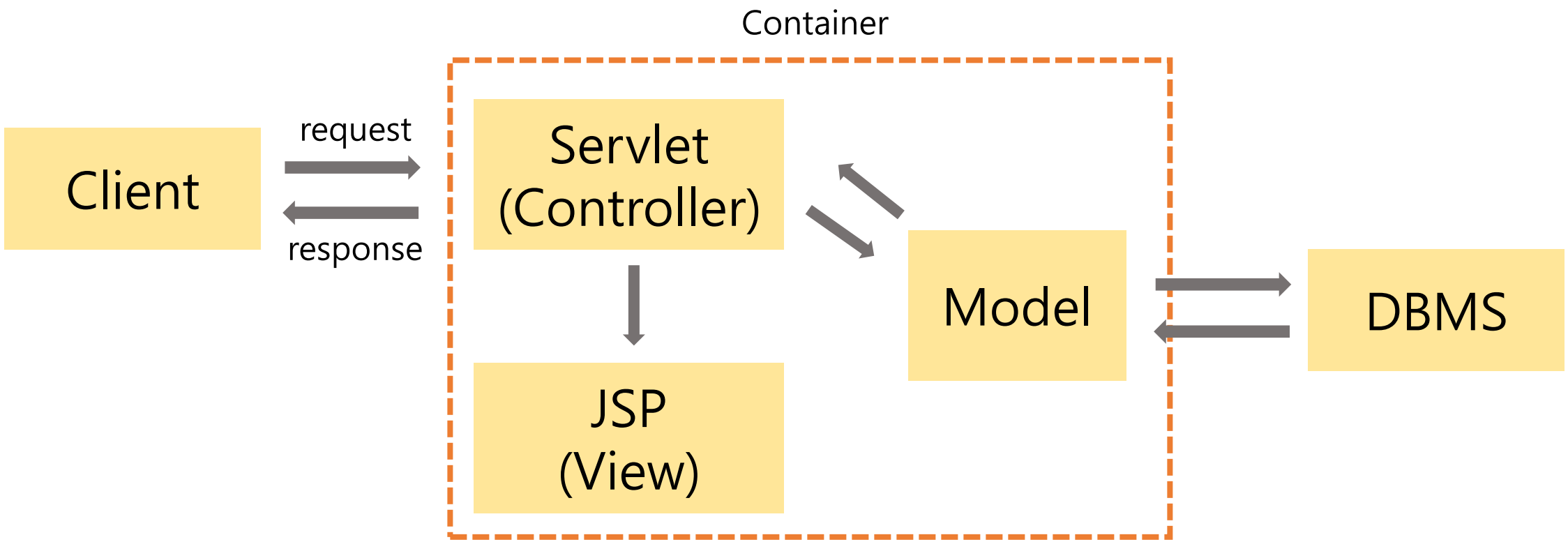
Model2

JSP에서는 출력만
처리

Model1


Container
Client
request
response
JSP
(Controller + View)
Model
DBMS

Model2


Container
Client
request
response
Servlet
(Controller)
JSP
(View)
Model
DBMS

Model1 vs Model2

Model 1	Model 2
View 와 Controller가 같은 JSP에서 실행	View와 Controller가 엄격히 구분 View는 어떠한 처리 Logic도 포함X
개발기간 단축	초기 구조 설계에서 많은 시간이 필요
유지 보수 어려움	유지 보수, 확장이 용이
디자이너와 개발자간 소통 필요	디자이너와 개발자의 작업 분리
간단한 웹 애플리케이션 구현에 용이	중 대형 프로젝트에 적합

싱글톤 패턴(Singleton Pattern)

-> 클래스 설계할 때 인스턴스가 오직 한 개만 생성되도록 설계하는 것
-> 메모리 낭비를 막기 위한 설계 방법

```
Public class MemberDAO{
        private MemberDAO(){
          //객체 초기화
        }

        private static MemberDAO instance = new MemberDAO();

        public static MemberDAO getInstance(){
            return instance;
        }
```