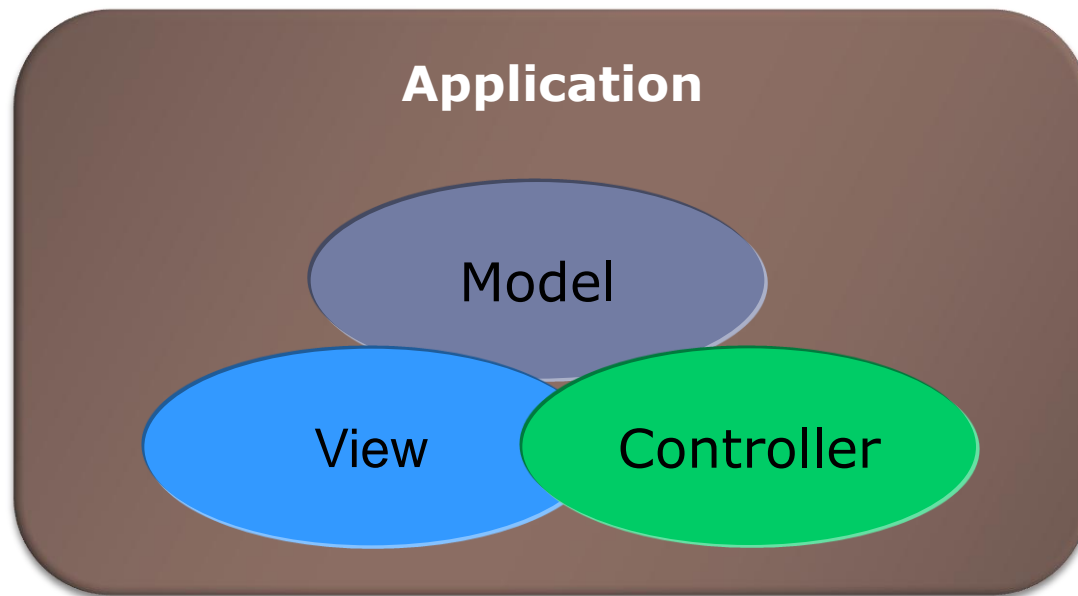

4. MVC Design Pattern

MVC Design Pattern

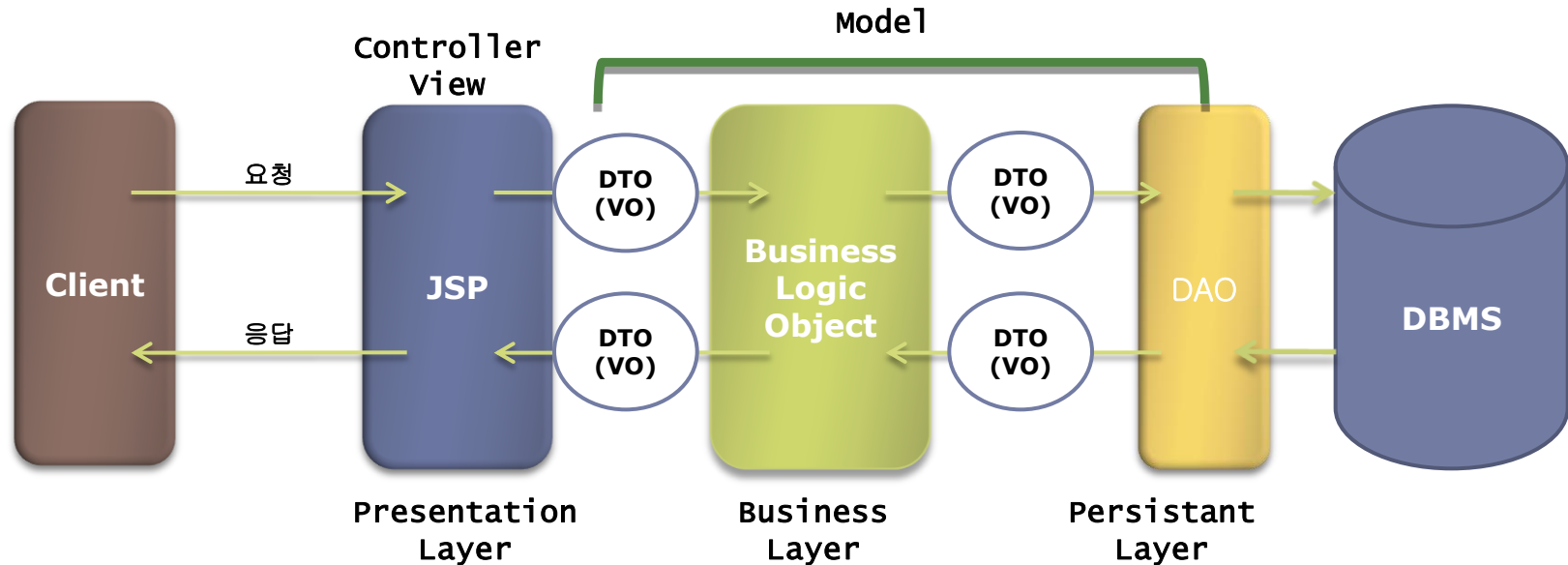
- ▶ 애플리케이션을 Model, View, Controller 3가지 영역으로 구분하고 영역간의 결합도를 최소화하여 개발하는 패턴



MVC Design Pattern

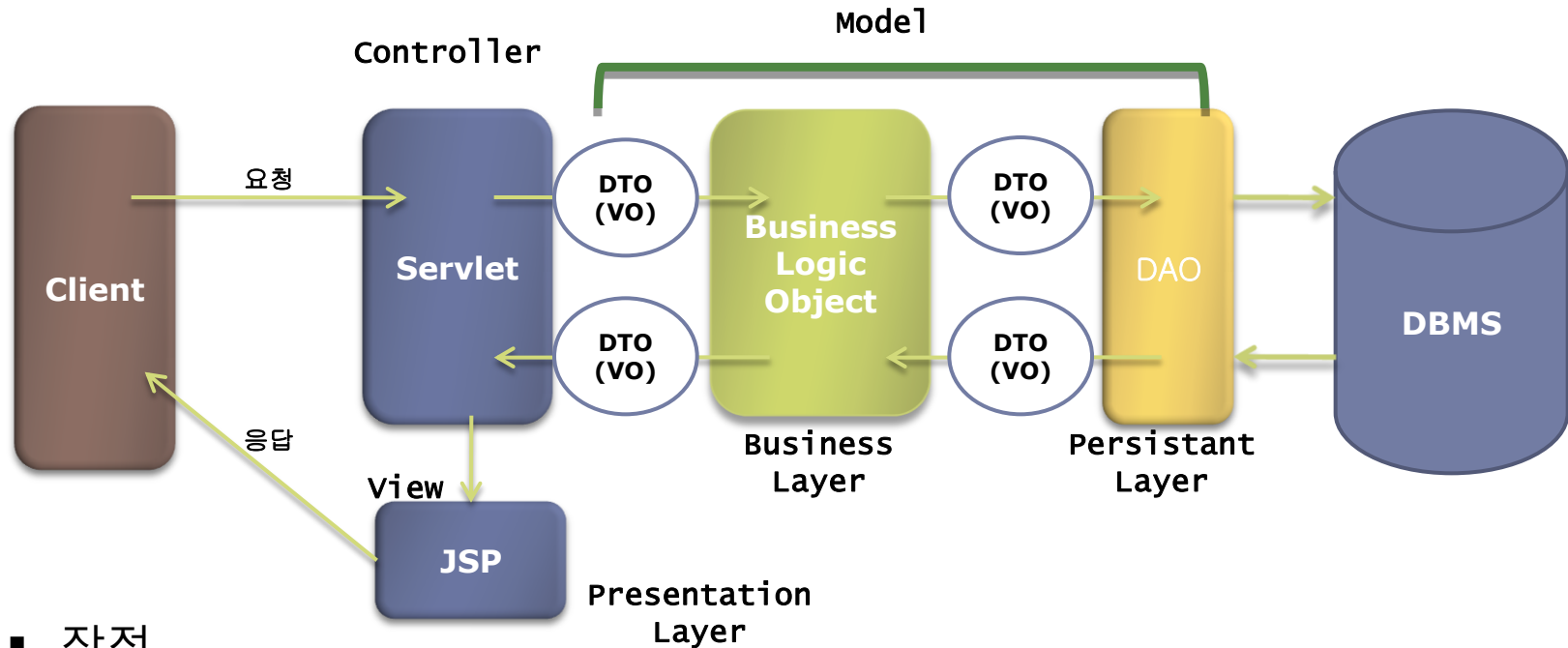
- ▶ Model
 - 애플리케이션에서 서비스하는 Business Logic , 사용되어지는 데이터를 다룬다.
- ▶ View
 - 사용자에게 보여질 Presentation Logic을 담당한다.
- ▶ Controller
 - 흐름을 관리하는 역할로써, 사용자의 요청을 받아 적절한 Model영역의 Business Logic을 호출하고 그 결과에 따른 적절한 View를 선택하여 보여질 수 있도록 한다.

MVC Model 1



- 장점
 - 쉬운 개발 방법으로 인한 개발 시간 단축
 - 성능면에서 유리
- 단점
 - View의 재사용이 어려움
 - Controller와 View의 코드가 섞여 있어 View의 변경이 용이하지 않음

MVC Model 2



- 장점
 - 비즈니스 로직의 결과에 따라 View 수정이 용이
 - Presentation 계층과 Business 계층의 독립성 보장
 - View의 재 사용성이 뛰어나고, 유지 보수 및 확장성이 유리
- 단점
 - 아키텍처에 대한 높은 이해 필요
 - Controller인 Servlet의 추가적인 개발이 필요

Page 이동 방식

▶ Redirect 방식

- 서버측에서 이동할 페이지의 정보를 브라우저로 응답을 보낸다.
- 브라우저는 응답정보에 있는 url로 다시 요청을 보낸다.
- 따라서, 브라우저의 URL이 변화한다.

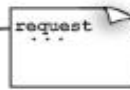
Page 이동 방식

▶ Redirect 방식

① 브라우저 주소 창에 URL을 입력합니다.



② 서버/컨테이너로 요청이 날아 갑니다.



③ 서버/컨테이너는 요청을 다른 URL로 보내야 하는 것임을 간파한 다음.



⑥ 브라우저는 응답을 받은 다음, 상태 코드가 301임을 확인한 다음 Location 헤더 값이 무엇인지 확인합니다.



⑤ HTTP Response에는 상태 코드 헤더에 301 값과 Location 헤더에 새로운 URL 값을 포함하고 있습니다.



④ Response 객체의 sendRedirect() 메소드를 호출합니다. 이것으로 서버/컨테이너는 끝.



response

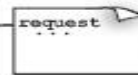
Page 이동 방식

▶ Redirect 방식

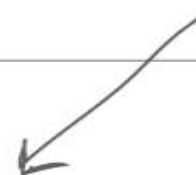
⑦ Location 값으로 받은 URL로 브라우저는 새로운 요청을 날립니다. 사용자도 브라우저의 주소창의 값이 바뀌는 것이 보이기 때문에 이 사실을 알 수 있습니다.



⑧ 사실 새로 발생한 요청이 처음 요청과 비교해 다른 점은 없습니다. 단지 방향 바꾸기에 의해 생성된 요청이라는 것만 빼면...



⑨ 서버는 요청을 접수합니다. 여기에도 뭐 별 특별한 것은 없습니다.



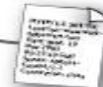
어떻게 해서 이리로 왔지?



⑪ 브라우저는 리턴받은 내용을 화면에 출력합니다. 놀라는 사용자 얼굴이 보이니까?



⑩ 다른 HTTP Response와 별반 다른 것이 없는 응답을 보냅니다. URL 자체가 사용자가 입력한 값이 아니라는 것만 빼고는...



Page 이동 방식

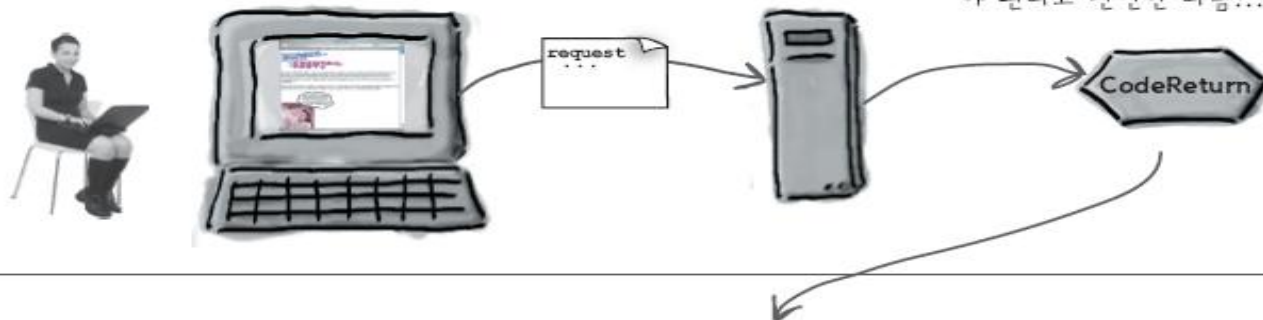
▶ Forward 방식

- 서버측에서 이동할 페이지의 정보를 브라우저로 응답을 보내지 않고 직접 페이지를 이동 시킨다.
- 이때, 이동되어지는 페이지는 기존 요청,응답을 그대로 전달받는다.
- 따라서, 브라우저의 URL은 변화없다.

Page 이동 방식

▶ Forward 방식

- 1 브라우저 주소창에 서블릿 URL을 입력합니다.
- 2 서버 컨테이너로 요청이 날아갑니다.
- 3 서블릿이 보기에 이 요청은 웹 애플리케이션의 다른 컴포넌트 (지금 예에서는 JSP)가 처리해야 된다고 판단한 다음...



- 5 어느 때와 마찬가지로 브라우저는 응답을 받고 화면에 출력합니다. 리다이렉트처럼 브라우저 주소창의 값이 바뀌지 않습니다. 브라우저는 JSP가 페이지를 만들었는지, 서블릿이 만들었는지 알 길이 없죠.
- 4 서블릿은 다음 코드를 실행합니다.

```
RequestDispatcher view =  
    request.getRequestDispatcher("result.jsp");  
view.forward(request, response);
```


이제 제어는 JSP에게 넘어 갔습니다.

