1. JSP 입문

목차

- 웹 애플리케이션의 개요
- JSP와 서블릿(Servlet)

1. 웹 애플리케이션의 개요

- 웹 프로그래밍
  - 웹 상에서 사용자와 기업 또는 사용자들간의 연결을 가능하게 하는 프로그래밍 언어
  - HTML : 웹 기반의 언어로 가장 먼저 개발되어 텍스트 기반의 웹 문서 작성이나 정적인 데이터들을 처리하는 데에는 편하지만 동적인 데이터를 처리할 수 없음
  - 동적인 데이터를 처리하기 위해 CGI, ASP, PHP, JSP 등의 기술 사용

1. 웹 애플리케이션의 개요

- 웹 프로그래밍
  - 동작 원리

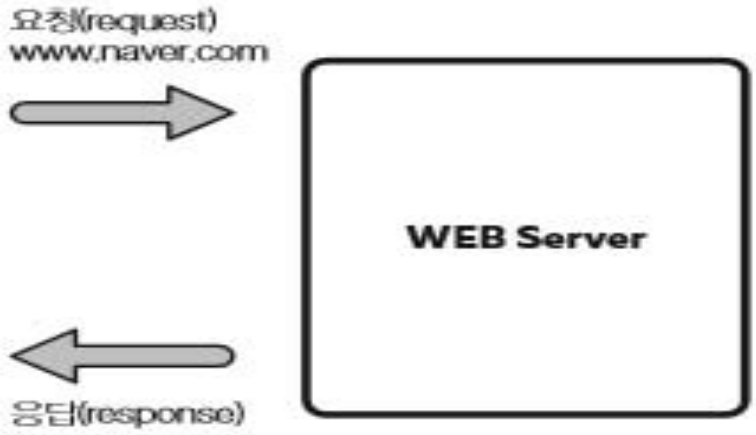


그림 1-1. 클라이언트/서버 구조

1. 웹 애플리케이션의 개요

- 웹 프로그래밍언어
  - CGI(Common Gateway Interface)
    - 응용 프로그램과 웹 서버 사이의 정보를 주고받는 방식이나 규약
    - 초기 웹 프로그래밍에 사용되었으나 데이터베이스와의 연동이 불편하고 익히기 어려운 단점이 있음
    - 현재 대부분의 웹 프로그래밍은 ASP, PHP, JSP 를 이용한 것이 대부분임

1. 웹 애플리케이션의 개요

- 웹 프로그래밍언어
  - ASP(Active Server Page)
    - 동적인 웹 페이지 구현을 위해 Visual Basic 언어로 만들어진 웹 프로그래밍 기술
    - 장점
      - Visual Basic을 기반하기 때문에 문법이 쉬워 개발 속도가 빠름
      - Active-x, DDL 컴포넌트를 사용하여 어느 정도 확장성도 보유
    - 단점
      - 웹 서버로 오직 Windows 기반의 IIS(Internet Information Server)만을 사용하므로 플랫폼에 비독립적
      - Java 기반의 JSP에 비해 시스템 자원의 효율성과 확장성이 떨어짐

1. 웹 애플리케이션의 개요

- 웹 프로그래밍언어
  - PHP(Personal Hypertext Preprocessor)
    - C 기반으로 만들어진 언어
    - 장점
      - C언어를 기반하는 언어이기 때문에 속도가 빠름
      - 개인적인 용도로 개발된 언어이기 때문에 100% 무료로 사용 가능
    - 단점
      - 서버 측의 지원 인프라가 매우 부족하여 확장성이 떨어짐
      - 기업형의 복잡한 구조에 적용하기 힘들며 보안상의 약점을 가짐

1. 웹 애플리케이션의 개요

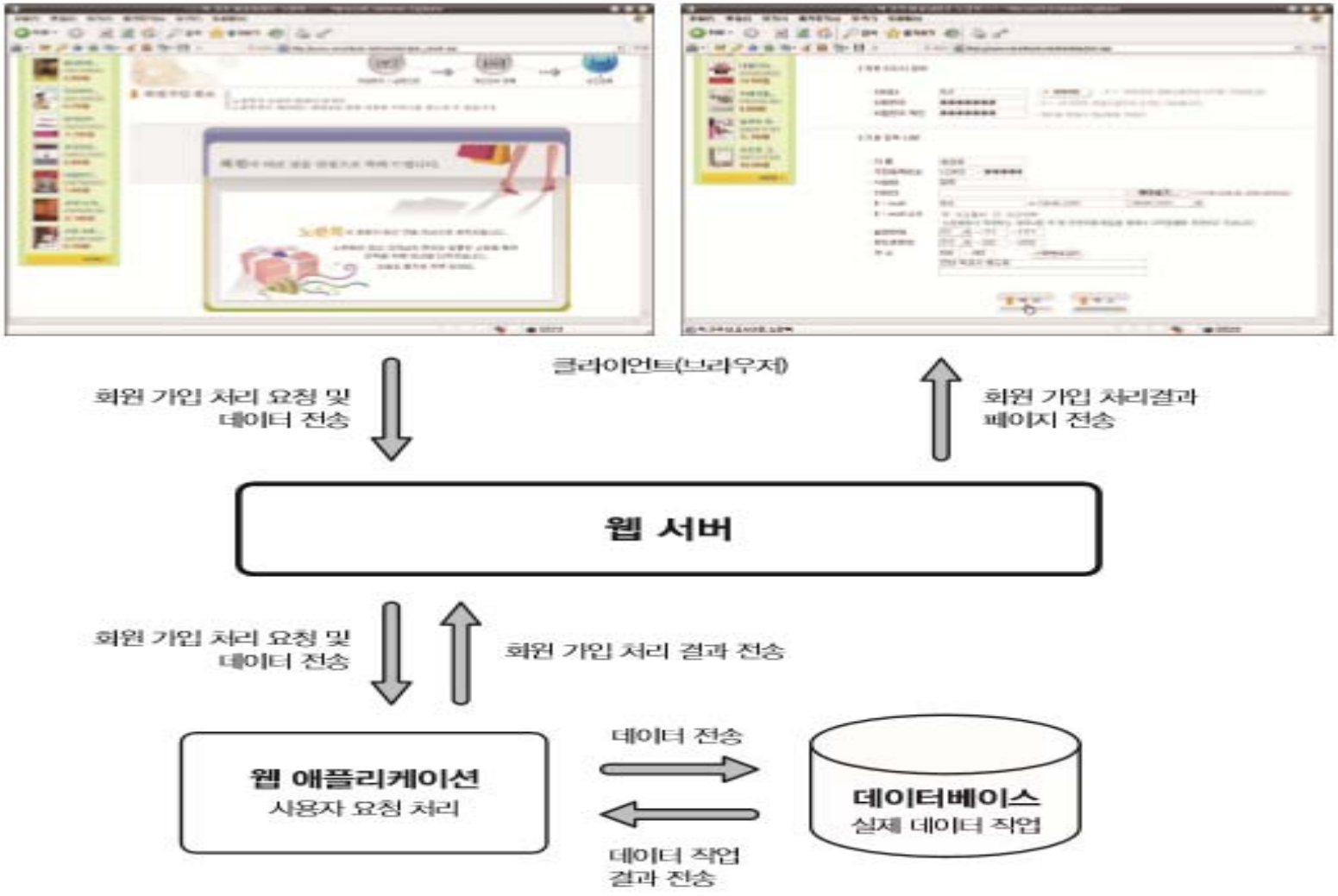
- 웹 프로그래밍언어
  - JSP(Java Server Page)
    - 유저인터페이스 구현이 쉬운 ASP의 장점을 수용하여 개발한 자바 기반 웹 프로그래밍 언어
    - 장점
      - ASP, PHP처럼 스크립트 기반하기 때문에 서버 페이지를 쉽게 작성할 수 있음
      - 서블릿과 함께 구동함으로써 서블릿의 기능을 그대로 사용할 수 있음
      - 자바빈즈(JavaBeans), EJB같은 기술로 보다 강력한 객체지향적 지원 가능

1. 웹 애플리케이션의 개요

- 웹 프로그래밍언어
  - JSP(Java Server Page)
    - 장점
      - JSTL을 지원하게 되면서 웹 프로그램의 가독성이 좋아지고 유지 및 보수가 훨씬 쉬워지는 장점
      - 규모가 클수록 장점이 부각되기 때문에 최근에는 일정 규모 이상의 웹사이트를 개발할 때 대부분 JSP를 이용

1. 웹 애플리케이션의 개요

▶ 웹 애플리케이션의 구조



1. 웹 애플리케이션의 개요

- 프로세스(process)와 스레드(thread)
- 프로세스 : 실행중인 프로그램
  - 초기 CGI 프로그램은 사용자 수에 따라 웹서버가 끊임 없이 프로세스를 생성해야 함
    => 같은 페이지에 대해 다른 자원을 할당해야 하므로 비효율적

1. 웹 애플리케이션의 개요

- 스레드
  - 하나의 프로세스 내에서 해당 프로세스가 할당 받은 자원을 공유하며 실행되는 독립된 작업 단위
  - 서블릿 : 스레드 기반
    - 같은 페이지 요청을 스레드 기반으로 처리하면 프로세스의 자원을 참조하면 되므로 성능이 향상됨
      - 새로운 프로세스 생성보다 기존 프로세스에 새로운 스레드를 하나 생성하는 것이 약 67배 빠름
      - 무한정 스레드를 생성할 수 없으므로 서블릿 컨테이너가 스레드 수와 프로세스 수를 적절히 조절함

2. JSP와 서블릿(Servlet)

- JSP(Java Server Page)의 개요
  - Java를 이용하여 동적인 웹 페이지를 만들기 위해 Sun Microsystems사가 개발한 기술
  - 현재 시간을 표시해주는 예시를 통한 요청 처리 흐름

그림 1-3. Jsp의 현재시간 요청 처리 흐름

2. JSP와 서블릿(Servlet)

- JSP(Java Server Page)의 특징
  - 강력한 이식성
    - JSP의 가장 큰 장점
    - JVM의 특성상 작성된 코드는 별다른 수정 없이 운영체제나 JSP 컨테이너의 종류에 관계없이 사용 가능함
  - 서버 자원의 효율적 사용
    - 스레드를 활용하여 자원을 효율적으로 사용함

2. JSP와 서블릿(Servlet)

- JSP(Java Server Page)의 특징
  - 간편한 MVC 패턴 적용
    - MVC(Model View Controller) 패턴
      - 사용자에게 보여지는 화면인 View 부분과 실제 비즈니스 로직이 들어가는 Model 부분, View와 Model을 연결시켜 주는 Controller 부분으로 구성됨
      - 디자인 패턴
      - JSP(View), 자바빈즈(Model), 서블릿(Controller)으로 쉽게 구현할 수 있어 프로젝트 규모가 커지더라도 View 부분과 Model 부분의 분업으로 훨씬 더 효율적인 개발 가능

2. JSP와 서블릿(Servlet)

- JSP(Java Server Page)의 특징
  - 간편한 MVC 패턴 적용

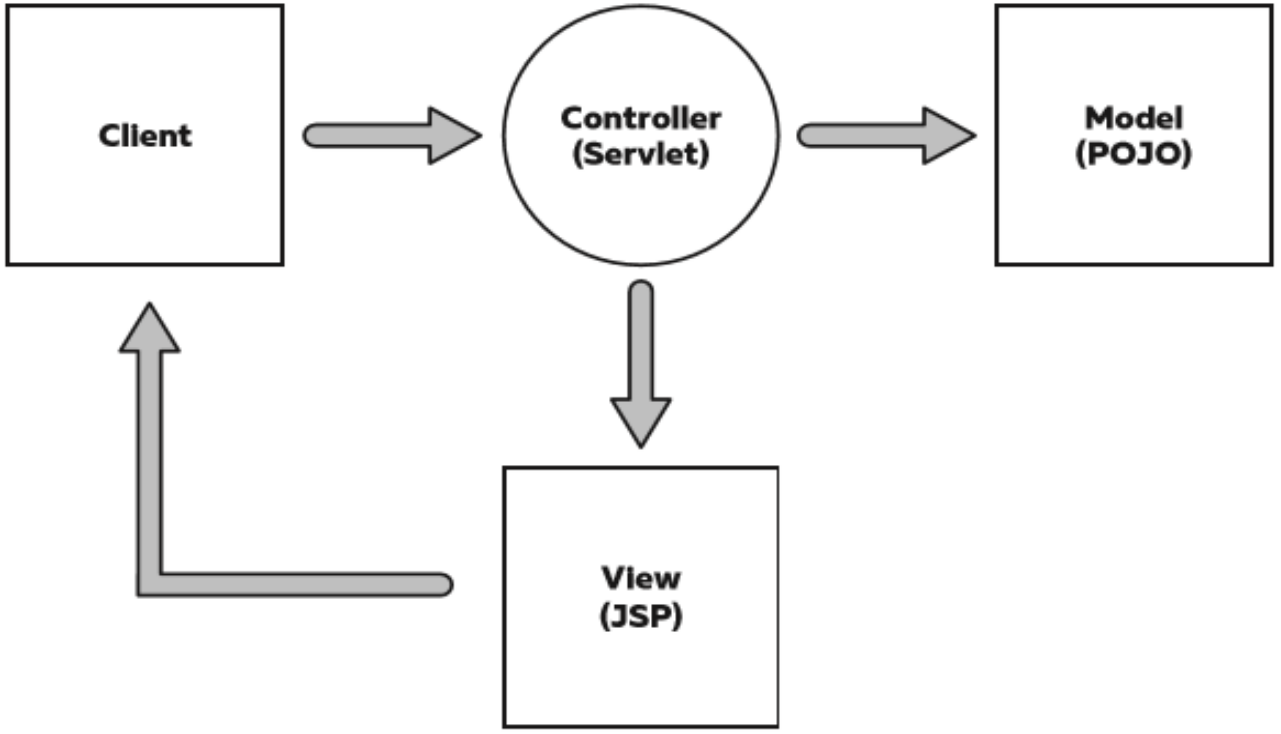


그림 1-5. Jsp를 이용한 MVC 패턴 구성도

2. JSP와 서블릿(Servlet)

- JSP(Java Server Page)의 특징
  - JSTL(JSP Standard Tag Library), 커스텀 태그 등을 이용한 개발 용이성
    - 자바 문법과 HTML 태그가 함께 작성되어 가독성이 떨어지는 단점이 있음
    - JSTL과 커스텀 태그에 대한 지원 강화로 자바 형식의 문법을 태그 라이브러리 파일로 만들어 JSP를 태그 만으로 구성할 수 있게 됨
    - 코드의 가독성 및 유지 보수의 효율이 대폭 향상됨

2. JSP와 서블릿(Servlet)

- JSP(Java Server Page)의 특징
  - 서블릿(Servlet)과 비교되는 장점
    - 전체 페이지가 Java 코드로 이루어짐
    - JSP
      - 전체 틀은 HTML 태그로 구성되고 JSP 코드가 필요한 부분은 <% %>안에 삽입함

2. JSP와 서블릿(Servlet)

- 서블릿(Servlet)의 개요
  - 웹 서버 측에서 사용자의 요구에 따라 자동으로 생성된 HTML형식의 페이지를 생산해 전송해 줄 수 있는 여러 기술이 개발되었고 그 중 자바 진영의 기술
  - 웹 서버 상에서 실행되는 자바의 클래스 파일이라고 할 수 있기 때문에 기본적으로 자바의 모든 API를 그대로 사용할 수 있으며 강력한 객체지향성 등 자바의 장점을 모두 가질 수 있음
  - 반드시 javax.servlet. Servlet 인터페이스를 구현(Implements)해서 작성해야만 함
  - 입력과 출력을 HTTP 프로토콜 의 요청(Request)과 응답(Response)의 형태로 다룸
  - **서버사이드의 자바 응용 프로그램!**

2. JSP와 서블릿(Servlet)

- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - 인터넷 통신 프로토콜 중 하나
  - 웹 브라우저 통신에 관한 프로토콜
  - HTTP 프로토콜의 구조
    - 요청(Request)과 응답(Response)의 형태로 이루어짐

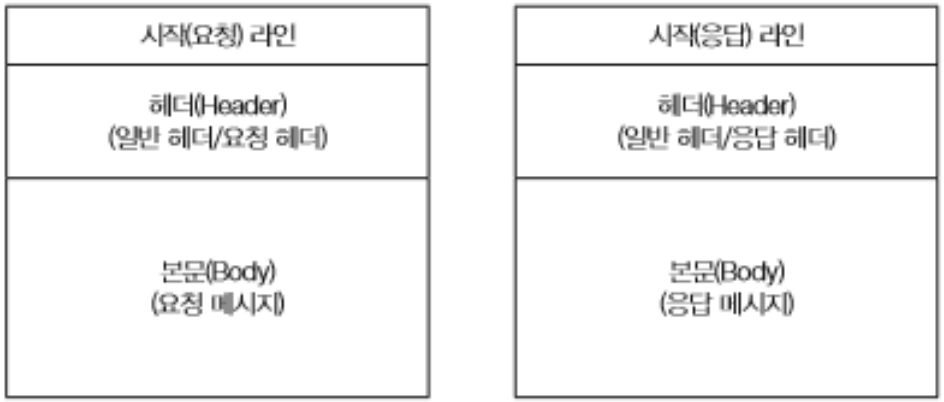


그림 1-6. HTTP 메시지의 구조

2. JSP와 서블릿(Servlet)

- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - HTTP 요청(Request) 메시지
    - HTTP 메소드(Method)와 접근할 주소(URL) 정보 그리고 서버에 전달할 데이터인 폼 파라미터로 구성
    - GET 메소드와 POST 메소드

2. JSP와 서블릿(Servlet)

- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - HTTP 요청(Request) 메시지
    - GET 메소드
      - Get 방식 요청
      - 전송할 파라미터 값들을 시작 라인의 URL 정보에 붙여서 같이 전송
      - 데이터의 크기 제약, 속도 빠름, 보안 취약

그림 1-7. GET 방식의 요청 메시지

2. JSP와 서블릿(Servlet)

- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - HTTP 요청(Request) 메시지
    - POST 메소드
      - Post 방식 요청
      - 파라미터 값들을 요청 메시지의 본문(Body)에 담아서 전송
      - 데이터 크기 제약 無, 속도 느림, 보안 강함

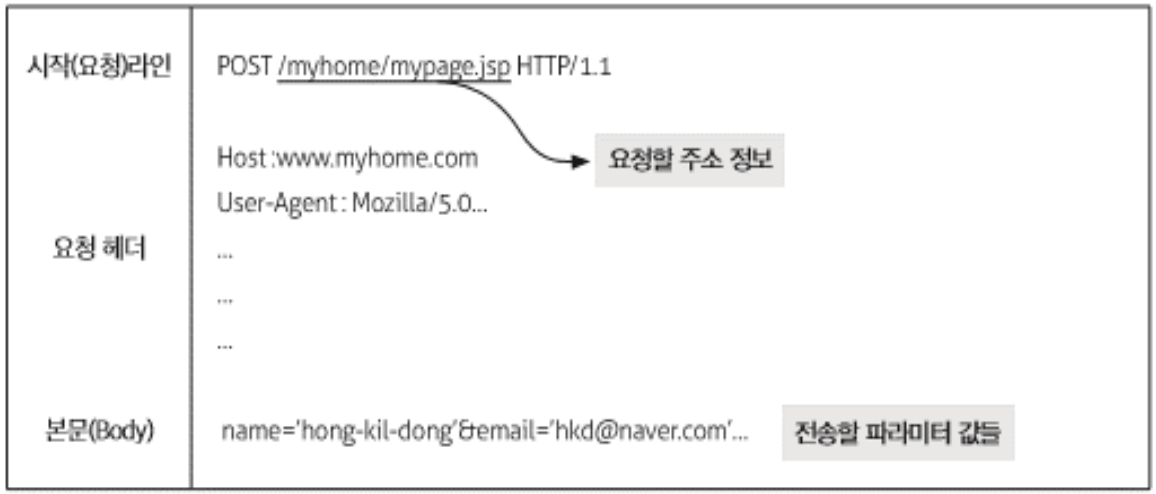


그림 1-8. POST 방식의 요청 메시지

2. JSP와 서블릿(Servlet)

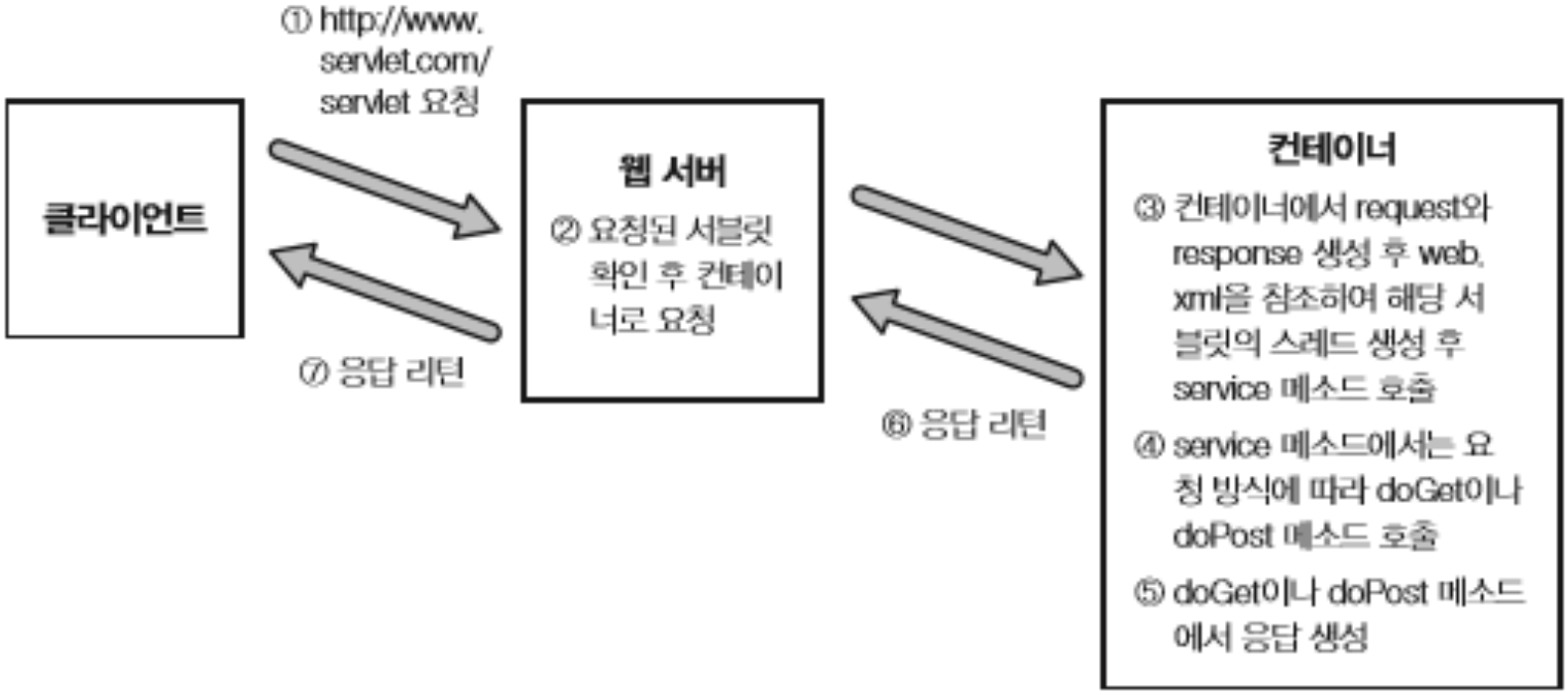
- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - HTTP 응답(Response) 메시지
    - 요청에 대한 서버의 처리 성공 여부를 표시하는 상태 코드(HTTP 404, 500 등) 번호와 웹 서버가 응답해주는 콘텐츠의 타입 정보(텍스트/HTML, 이미지 등), 콘텐츠의 내용으로 구성

2. JSP와 서블릿(Servlet)

- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - 웹 컨테이너
    - JSP 파일의 실행 요청을 처리
    - 웹 서버 내부에서 서블릿 클래스 또는 JSP 파일을 실행하기 위한 실행 환경을 제공하는 역할
    - HTTP 서버
      - 어떤 주소에(URL) 대한 요청 → HTTP 서버 → 그 주소에 미리 매핑되어 있는 콘텐츠(HTML 파일이나 이미지 등)를 사용자의 브라우저에 응답 형태로 전송
    - 웹 컨테이너
      - 요청된 URL이 서블릿 클래스 또는 JSP 파일일 경우 HTTP 서버 → 웹 컨테이너 → 요청된 URL에 맞는 서블릿 클래스 또는 JSP 파일을 실행하여 결과를 HTTP 서버에 넘겨줌 → 응답 메시지 형태로 사용자 브라우저에 전송

2. JSP와 서블릿(Servlet)

- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - 서블릿의 동작 원리
    - 사용자의 URL 요청 → request, response 객체 생성 → 서블릿 인스턴스와 스레드 생성 → service( ) 메소드 호출과 서블릿 클래스 실행 → 응답과 스레드 소멸



2. JSP와 서블릿(Servlet)

- HTTP(HyperText Transfer Protocol) 프로토콜의 이해
  - 서블릿의 라이프 서클(life circle)

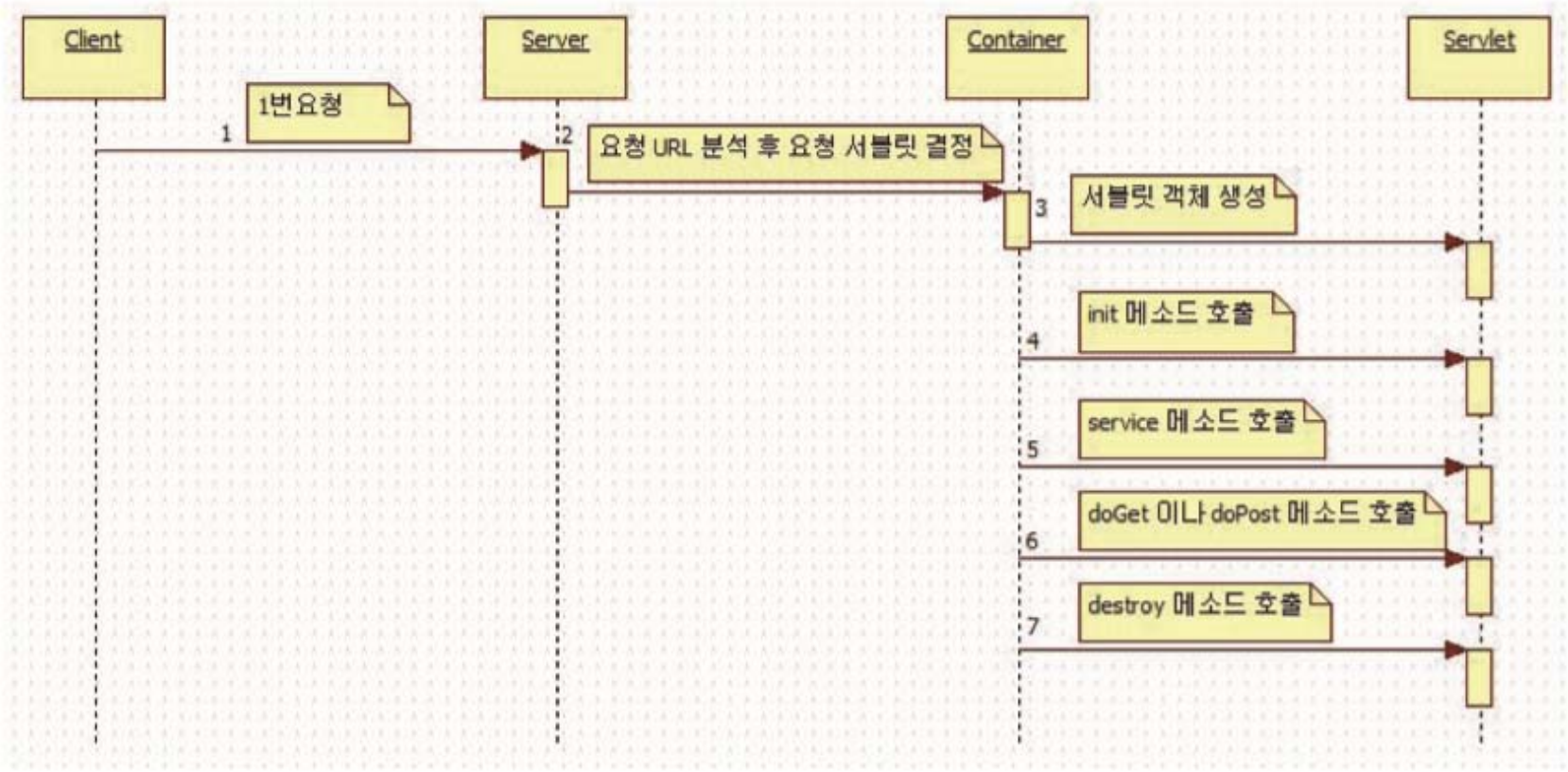


그림 2-54. 서블릿 객체 생성 소멸 단계