# 웹개발

## 웹 어플리케이션의 이해

* 데스크톱 어플리케이션에서 웹 어플리케이션이 개발되기까지 단계를 정리한 부분

## 데스크톱 어플리케이션

* 장점 : 실행속도가 빠름
* 단점 : 배포가 번거로움(모든 PC에 수동 배포 해야함)  
   보안에 취약함(DB가 개인 PC에 설치되어 있기 때문에 보안에 취약함)

## 클라이언트, 서버 어플리케이션(단일 스레드)

* 장점 : 비즈니스 로직 변경시 서버만 변경하면 되기 때문에 배포가 편리함
* 단점 : 단일 쓰레드로 하나의 요청에 하나의 응답만 가능함  
   여러 세션에서 요청시 처리순서를 기다려야 하는 문제점 발생  
   ( 앞 요청에서 오류가 발생한 경우 계속해서 대기를 해야함)

## 클리언트, 서버 어플리케이션(다중 클리아이언트)

* 장점 : 여러 클라이언트의 요청을 처리할 있다.

🡺C/S 프로그램(웹 제외)은 위와 같이 진화를 하였지만, 근본적으로 클라이언트 프로그램이 변경되면 PC마다 재설치를 해야한다는 단점을 가지고 있다.

## 웹 어플리케이션

* 웹 서버를 개설하여 클라이언트가 보는 화면을 통합 관리할 수 있다
* 어플리케이션 서버를 개설하여 비즈니스 로직을 통합 관리할 수 있다.
* 웹 서버와, 어플리케이션 서버를 통해서 수정된 화면과 비즈니스 로직을 쉽게 배포할 수 있게 되었다.

# 웹 프로그래밍 기초

# HTTP 프로토콜

* 웹브라우저와 **웹서버** 사이의 데이터 통신 규칙
* SOAP, RESTful : 클라이언트와 서버 서버 사이의 요청하고, 응답하는 방식을 의미  
  (두 방식 모두 HTTP 프로토콜을 응용하거나 확장한 기술)
* WebDAV(Web Distributed

## 요청 헤더(GET)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| GET | 공백 | / | 공백 | HTTP/1.1 |
| 메서드 |  | 요청 URI |  | HTTP 버전 |

* 메서드 : GET, POST, HEAD, PUT, DELETE, TRACE, CONNCET, OPTIONS…
* URI : 이미지 ,동영상, 어플리케이션 등의 가상 경로
* HTTP 버전 : 현재 요청 HTTP 문서의 버전

## HTTP 헤더 유형

|  |  |  |
| --- | --- | --- |
| 유형 | 헤더명 | 설명(추후 하나씩 기재 예정) |
| 일반 헤더 | Cache-Control |  |
| Connection |  |
| Date |  |
| Pragma |  |
| Trailer |  |
| Transfer-Encoding |  |
| Upgrade |  |
| Via |  |
| Warning |  |
| 요청 헤더 | Accept |  |
| Accept-Charset |  |
| Accept-Encoding |  |
| Accept-Language |  |
| Authorization |  |
| Except |  |
| From |  |
| Host |  |
| If-Match |  |
| If-Modified-Since |  |
| If-None-Match |  |
| If-Range |  |
| If-unmodified-Since |  |
| Max-Forwards |  |
| Proxy-Authorization |  |
| Range |  |
| Referer |  |
| TE |  |
| User-Agent |  |
| 응답 헤더 | Accept-Rages |  |
| Age |  |
| ETag |  |
| Location |  |
| Proxy-Authenticate |  |
| Retry-After |  |
| Server |  |
| Vary |  |
| WWW-Authenticate |  |
| 본문 헤더 | Allow |  |
| Content-Encoding |  |
| Content-Language |  |
| Content-Location |  |
| Content-MD5 |  |
| Content-Range |  |
| Content-Type |  |
| Expires |  |
| Last-Modified |  |
| 기타 확장 헤더 |  |

## 응답헤더

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HTTP/1.1 | 공백 | 200 | 공백 | OK |
| HTTP 버전 |  | 상태 코드 |  | 상태 설명 |

|  |  |
| --- | --- |
| **상태코드** | **상태설명** |
| 200 | 요청이 성공적으로 처리되었다 |
| 301 | 요청한 자원이 이동되었다.  헤더 정보에 이동 위치를 알려줄 테니 다시 요청하라 |
| 304 | 클라이언트가 임시 보관한 응답결과와 다르지 않다. |
| 400 | 잘못된 요청이다. |
| 404 | 요청한 자원을 못 찾았다.  서버 내부에서 오류가 발생하였다. |
| 500 |

# GET 요청

* GET 요청시 서버에 데이터를 전달하는 방법 기술

## GET 요청의 특징

* URL에 데이터를 포함 -> 데이터 조회에 적합
* 바이너리 및 대용량 데이터 전송 불가
* 요청라인과 헤드 필드의 최대 크기 : HTTP 사양에는 제한 사항 없음  
   대용량 URL로 인한 문제 발생-> 웹 서버에 따라 크기 제한 있음  
   (Microsoft IIS 6.0 : 16KB, Apache 웹 서버 : 8KB)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 서비스주소 | ? | V1 | = | 23 | & | V2 | = | 15 |
|  | 서비스와 데이터를 분리 | 매개변수명 |  | 값 | 데이터 구분자 | 매개변수명 |  | 값 |

## GET 요청의 문제점

* 웹 주소창에 사용자가 입력한 정보가 모두 노출되기 때문에 보안에 취약하다
* 바이너리 데이터를 전송할 수 없다.

: BASE64 인코딩 방식을 이용하여 바이너리 데이터(이미지,동영상)를 문자화해서 보낼 수는 있지만, 헤더에 용량제한으로 적합하지 않다.

## GET 요청시 HTML Form의 Method 속성 값 : get (기본값이 get 이므로 기재하지 않아도 무관)

**<form action=”LoginServlet” method = “get” >**

# POST 요청

## POST 요청의 특징

* URL에 데이터가 포함되지 않음->외부노출방지
* 메시지 본문에 데이터 포함->실행 결과 공유 불가
* 바이너리 및 대용량 전송 가능

## POST 요청시 HTML Form의 Method 속성 값 : post

**<form action=”LoginServlet” method = “post” >**

## POST 바이너리 데이터 전송(파일업로드)

* 파일 업로드 폼 소스
* Enctype 기본값 : ‘application/x-www-form-urlencoded’

|  |
| --- |
| <form action="FileUploadServlet" method="post"  enctype="**multipart/form-data**">  사진: <input type="file" name="photo"><br>  설명: <textarea name="description" cols="50" rows="3"></textarea><br>  <input type="submit" value="추가"><br>  </form> |

* 결과

|  |
| --- |
| * POST /Lesson02/FileUploadServlet HTTP/1.1 * Host: localhost:8080 * Content-Length: 879694 * Cache-Control: max-age=0 * Origin: http://localhost:8080 * Upgrade-Insecure-Requests: 1 * Content-Type: multipart/form-data; * boundary**=----WebKitFormBoundaryWCCbgrgoeullXWFH** * User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.84 Safari/537.36 * Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8 * Referer: http://localhost:8080/Lesson02/MultipartTest.html * Accept-Encoding: gzip, deflate, br * Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7 * **------WebKitFormBoundaryWCCbgrgoeullXWFH** * Content-Disposition: form-data; name="photo"; filename="Chrysanthemum.jpg" * Content-Type: image/jpeg * //사진 바이너리 데이터 * ���� * . * . * . * . * . * **------WebKitFormBoundaryWCCbgrgoeullXWFH** * Content-Disposition: form-data; name="description" * 국화다~ |

## POST 일반 방식(문자전송) 멀티파트(이미지,동영상) 전송의 차이점

* 일반 방식(문자전송)에 사용되는 매개변수들은 &를 사용하여 구분이 가능하나 사진, 동영상 같은 바이너리 데이터는 &로 매개변수를 구분할 수 없다.
* 따라서 멀티파트 전송 방식을 사용하여 3) 의 결과와 같이 파트 구분자를 써서 전송하는 방식을 사용한다..