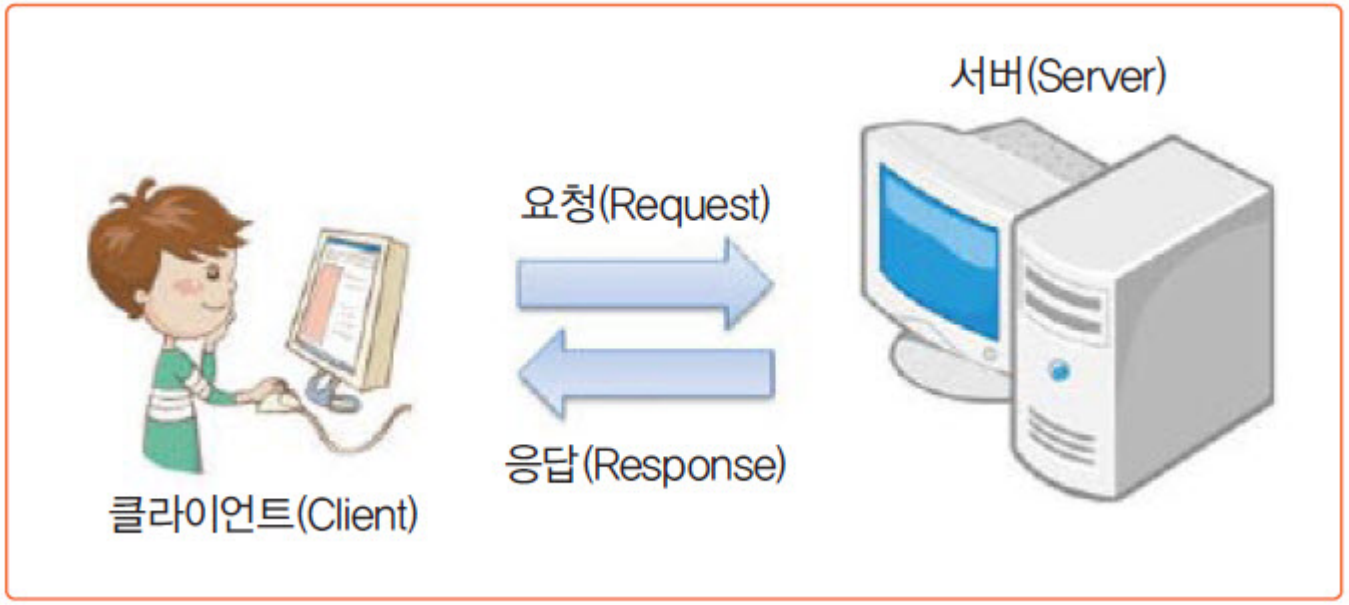
Web Programming

Web Programming

- Web
  - ✓ 인터넷에 연결된 컴퓨터를 통해 사람들이 정보를 공유할 수 있는 전 세계적인 정보 공간
  - ✓ 서비스에 대한 응답을 제공하기 위해서 서비스를 제공하는 서버(server)와 서비스를 요청하는 클라이언트(client)의 형태를 띄며 클라이언트가 웹 브라우저 나 별도의 애플리케이션을 통해 콘텐츠를 요청하면 서버는 클라이언트의 요청에 응답하는 형식으로 동작



Web Programming

- Web
  - ✓ 웹 애플리케이션은 수행되는 위치에 따라 Front End 와 Back End 로 분류하기도 함
  - ✓ Front End
    - Back End 와 사용자 사이에서 보여지는 부분을 만들기 위한 부분
    - 그래픽 디자인 과 이미지 제작과 같은 Visual Graphic 디자인 및 사용자 인터페이스 디자인 및 사용자 경험과 관련된 정보 디자인
    - HTML, CSS, JavaScript 언어를 이용해서 브라우저를 통해서 보여지도록 구현할 수 있는데 최근에는 라이브러리 형태로 제공되는 jQuery, Angular, react, vue, bootstrap 등을 사용해서 구현하는 경우가 많음
    - Python 이나 Java, Go 등 의 프로그래밍 언어를 이용해서 만든 PC 용 애플리케이션
    - Native Language(Java, Kotlin, Objective-C, Swift) 또는 JavaScript 나 Dart 와 같은 언어를 이용해서 Smart Device 에서 동작하는 애플리케이션 개발 가능
  - ✓ Back End
    - 애플리케이션을 동적이고 대화형으로 만들기 위한 부분
    - 클라이언트의 요청을 처리하기 위한 부분
    - PHP, Java, C#, JavaScript, Python, Ruby, Go 와 같은 프로그래밍 언어 와 Oracle, MySQL, Mongo DB 와 같은 데이터베이스 등의 기술 및 Spring, Django, Node 같은 프레임워크를 이용해서 구현

Web Programming

- Web
  - ✓ Web Application 의 동작 원리
    - Web Application은 최초 요청(request), 최초 응답(response), 추가 요청, 추가 응답의 과정을 거쳐 동작
    - 사용자가 주소 창에 페이지의 URL을 입력하는 행동으로 최초 요청을 서버에 전달하고 서버는 HTML 파일 형식으로 사용자의 컴퓨터(브라우저)에 응답을 전송하는데 HTML 파일에는 출력 내용의 구조만 담겨 있고 이미지 등의 자원을 포함하지는 않기 때문에 사용자의 컴퓨터는 다시 한번 서버에 추가 요청을 하고 서버에서는 필요한 내용을 CSS, JS, JPG(이미지)와 같은 여러 가지 파일을 전송하고 브라우저에서 해석해서 출력
    - 최근에는 서버에서는 문자열 형태의 데이터만 전달하고 이를 클라이언트에서 해석해서 출력하는 방법도 사용되고 있음

Web Programming

- ❖ 웹 2.0과 웹 3.0
  - ✓ 웹 3.0
    - 2010년 등장
    - 컴퓨터가 정보자원의 뜻을 이해하고 논리적 추론까지 가능한 시멘틱 웹 개념 등장
    - 속도와 플랫폼 변화
      - ◆ 10Mbps ~ 1Gbps 의 초고속 인터넷 환경과 4G LTE 등 초고속 무선 인터넷 서비스 보급
      - ◆ 인터넷 사용 장치가 PC → 스마트폰, 태블릿, 스마트TV 등으로 급격히 변화
    - 똑똑한 데이터와 인공지능의 향상
      - ◆ 컴퓨터가 정보 자원의 뜻을 이해하고 논리적 추론 까지 가능한 차세대 지능형 웹 기술
      - ◆ 구글 나우(Google Now), 애플 시리(Siri) 등 사용자의 상황이나 질문의 의도에 따라 지능화된 서비스 제공
    - 애플리케이션의 진화
      - ◆ 웹 2.0에서 시도되었던 open API, SOA 등이 새로운 플랫폼 등장으로 더욱 발전
      - ◆ 메쉬업은 컴포넌트화된 애플리케이션의 부분을 조합해 개인이나 그룹의 용도에 맞게 여러 서비스 장치를 사용하는 사용자들이 손쉽게 자신만의 컨텐츠나 정보를 구성할 수 있도록 해줌

Web Programming

- WOA(Web Oriented Architecture)
  - ✓ 기존 PC 중심의 사용자 환경에서 스마트폰, 태블릿, Smart TV, Smart Car 등 새로운 기기들이 출현하면서 한 사람이 여러 기기를 통해 동일한 서비스 와 정보로의 접근이 필요해 짐에 따라 여러 기기 간의 끊어짐이 없는(Seamless) 서비스가 요구되면서 One Source Multi Use(OSMU)를 위한 N-Screen 혹은 N-Device 서비스에 대한 요구 증대
  - ✓ OSMU 서비스를 위한 기술의 변화
    - 하드웨어 인프라: Cloud라고 불리는 대규모의 공용 컴퓨팅 서비스, 가상화 SW
    - 소프트웨어 방법론: WOA
  - ✓ WOA
    - 기존 SOAP(Simple Object Access Protocol) 기반의 SOA(Service Oriented Architecture)에서 REST(Representational State Transfer) 기반의 경량 웹 서비스 모델로 발전했는데 Restful 웹 서비스는 JAX-RS(JSR-311)로 자바 규격에 공식적으로 포함
    - 웹을 중심으로 전체 시스템 아키텍처를 설계해 나가는 기술

Web Programming

- 프레임워크(Framework)
  - ✓ 개발의 방향 변화
    - 프로그램의 규모 확대 -> 높은 생산성, 쉬운 유지 보수, 기능 확장이 용이한 개발 기술 필요
    - 개발방법론, 소프트웨어 디자인 패턴, 리팩토링, 프레임워크 등 소프트웨어 공학적 기술 등장
  - ✓ 프레임워크(Framework)는 무언가를 만들기 위한 틀
  - ✓ 목적에 맞게 잘 설계된 구조와 미리 구현된 라이브러리가 포함된 소프트웨어 형태.
  - ✓ 프레임워크를 사용하면 정해진 규격에 따라 프로그램 구조를 만들어야 하는데 개발자가 신경쓰거나 처리해야 할 많은 일과 이벤트 관리는 프레임워크를 통해 처리
  - ✓ 여러 유틸리티 라이브러리도 제공하기 때문에 개발자는 비교적 적은 노력으로도 고품질의 소프트웨어 개발이 가능해짐.
  - ✓ 대표적인 자바 프레임워크는 스프링 프레임워크로 웹 개발을 포함해 대규모 시스템 개발에 적합한 기술 구조를 제공

Web Programming

- 웹 프로그램의 미래
  - ✓ 애플의 아이폰에서 시작된 스마트폰 열풍은 기존 컴퓨터 사용 패턴을 변화시킴.
  - ✓ 전통적인 PC 시장의 변화 -> 대규모 PC 제조업체인 DELL 상장 폐지 -> MS 시장 지배력 약화
  - ✓ 상당수의 PC 소프트웨어들이 설치 형에서 웹 형태로 전환됨.
  - ✓ 다양한 스마트 기기, 안드로이드, iOS, 웹 OS 등 새로운 모바일 운영체제의 성장과 함께 고속의 무선 인터넷을 기반으로 한 소프트웨어 발전이 가속화될 전망.

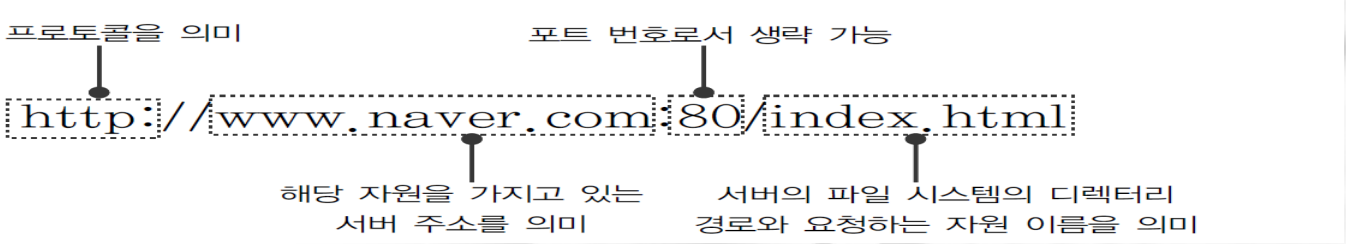
Web Programming

- 웹 프로그래밍
  - ✓ 웹 프로그래밍: TCP/IP 프로토콜로 통신하는 클라이언트와 서버를 개발하는 것
  - ✓ 웹 서버에 접속하는 방법
    - 웹 브라우저를 사용해서 요청
    - Linux 의 curl 명령을 사용하여 요청
    - 만들어진 클라이언트용 프로그램이나 직접 프로그램을 만들어서 요청
  - ✓ HTTP 메시지의 구조
    - Start Line: 요청 라인 또는 상태 라인 – 요청 방식과 URI 및 프로토콜 버전
    - Header
    - Blank Line: 헤더의 끝을 빈 줄로 식별
    - Body
    - Header 와 Body는 생략 가능

Web Programming

- URL(Uniform Resource Locator)
  - ✓ 클라이언트가 웹 서버에 존재하는 자원(정보, 파일 등)을 요청하는데 필요한 네트워크 서비스의 표현
  - ✓ 클라이언트의 웹 브라우저 주소란에 다음과 같은 형식으로 입력

    [프로토콜]://[호스트][:포트]/[경로]/[쿼리문자열]#Fragment
  - ✓ 프로토콜없이 //로 시작하는 주소는 상황에 따라 http 또는 https 프로토콜을 추가
  - ✓ 호스트는 컴퓨터를 구분하기 위한 이름으로 도메인이나 IP가 될 수 있음
  - ✓ 포트 번호는 애플리케이션을 구분하기 위한 번호로 프로토콜의 기본 포트를 사용하는 경우 생략 가능
  - ✓ 경로는 서버 애플리케이션에게 요청하는 경로 – 파일 경로가 아님
  - ✓ 쿼리 문자열은 주소 뒤에 추가로 붙는 정보로 일반적으로 parameter(매개변수)라고 불리는 것으로 클라이언트에서 서버에게 데이터를 전달할 목적으로 사용하는 것으로 요청 경로 뒤에 ?를 붙이고 이름 = 값 의 형태로 전달하며 2개 이상을 전달하고자 하는 경우에는 &로 구분해서 입력
  - ✓ Fragment는 # 으로 시작하는데 문서 내의 앵커와 같은 조각을 지정



Web Programming

- URL(Uniform Resource Locator)
  - ✓ URL을 바라보는 측면
    - URL은 웹 클라이언트에서 호출한다는 시점에서 보면 웹 서버에 존재하는 Application에 대한 API(Application Programming Interface)로 볼 수 있음
    - API의 명명 규칙을 정하는 방법을 두 가지로 분류할 수 있는데 하나는 URL을 RPC(Remote Procedure Call)로 바라보는 방식이고 다른 하나는 REST(Representational State Transfer)로 바라보는 방식
    - RPC
      - 클라이언트가 네트워크상에서 원격에 있는 서버가 제공하는 API 함수를 호출하는 방식
      - 이 방식의 배경에는 URL 설계와 API 설계를 동일하게 고려하여 URL의 경로를 API 함수 이름으로 하고 쿼리 스트링을 함수의 인자로 간주해서 웹 클라이언트에서 URL을 전송하는 것은 웹 서버의 API 함수를 호출한다고 인식하는 것
      - RPC 방식에서는 URL 경로의 대부분이 동사가 되는데 자바의 일반적인 함수나 메서드 이름이 동사인 것과 동일한 개념
      - RPC 방식은 웹 개발 초기부터 사용된 방식으로 REST 방식이 나오면서 사용 빈도가 줄어드는 추세이지만 여전히 많이 사용
      - RPC 방식은 다음과 같은 형태로 사용

        http://blog.example.com/search?q=test&debug=true

Web Programming

- URL(Uniform Resource Locator)
  - ✓ URL을 바라보는 측면
    - REST
      - ◆ 웹 서버에 존재하는 요소들을 모두 리소스로 정의하고 URL을 통해 웹 서버의 특정 리소스를 표현 한다는 개념
      - ◆ 리소스는 시간이 지남에 따라 상태(state)가 변할 수 있기 때문에 클라이언트 와 서버 간에 데이터의 교환을 리소스 상태의 교환(Representational State Transfer)으로 간주
      - ◆ 리소스에 대한 조작을 GET, POST, PUT, DELETE 등의 HTTP 메서드로 구분
      - ◆ 웹 클라이언트에서 URL을 전송하는 것이 웹 서버에 있는 리소스 상태에 대한 데이터를 주고받는 것으로 간주될 수 있음
      - ◆ REST 방식은 다음과 같은 형태로 사용

        http://blog.example.com/search/test

Web Programming

- URL(Uniform Resource Locator)
  - ✓ 간편 URL
    - 최근에는 REST 방식의 URL 개념을 기반으로 간단하면서도 사용자에게 친숙하게 URL을 표현하려는 노력이 진행되었고 주로 검색 엔진 분야에서 이러한 노력이 많이 진전되었는데 그 결과 기존의 길고 복잡한 URL에서 특수 문자 등을 제거하고 간결하게 만드는 방식인 간편(Clean) URL이 탄생
    - 기존 URL 방식에서 사용되는 문자(?, =, &, #)들은 웹 프로그래밍 언어 입장에서는 연산자 나 특수한 용도의 기호로 사용될 가능성이 높기 때문에 검색 엔진에서 이런 주소를 저장하고 표현하는 데 불편이 따름
    - 간편 URL은 쿼리 스트링 없이 경로만 가진 간단한 구조의 URL을 의미
    - 검색 엔진의 처리를 최적화하기 위해 생겨난 간편한 URL은 URL을 입력하거나 기억하기 쉽다는 부수적인 장점도 있어 검색 엔진 친화적 URL(search engine friendly url) 또는 사용자 친화적 URL(User friendly url)이라고 부르기도 함
      - http://example.com/index.php?page=foo -> http://example.com/foo
      - http://example.com/index.php?page=consulting/marketing -> http://example.com/consulting/marketing
      - l http://example.com/products?category=2&pid=25 -> http://example.com/products/2/25

Web Programming

- ❖ 전송 요청 방식
  - ✓ get
    - URL에 Query String을 이용해서 요청하는 방식
    - 이름 과 값은 = 으로 구분
    - 2개의 이상의 데이터를 전송하고자 할 때는 &로 구분해서 전송
    - 전송하는 데이터 길이에 제한이 있음
    - 한글을 전송하는 경우 RFC 2396 규약에 의해 인코딩 한 후에 전송
    - 폼에서 사용하면 처리가 지연되는 경우 재요청
    - 요청 방법
      - ◆ a 태그를 이용해서 페이지를 이동하는 경우
      - ◆ 자바스크립트에서 페이지를 이동하는 경우
      - ◆ ajax 요청을 get 방식으로 하는 경우
      - ◆ 폼에서 명시적으로 GET 방식으로 전송하는 경우

Web Programming

- 전송 요청 방식
  - ✓ post 방식
    - 데이터를 HTTP 요청의 본문(body)에 포함시켜 전송하며 폼 데이터나 파일 업로드와 같은 작업에 주로 사용
    - 요청 방법
      - ◆ 폼에서 명시적으로 POST 방식으로 요청
      - ◆ 자바스크립트에서 ajax 요청 시 명시적으로 post 방식으로 지정하는 경우
    - 데이터의 크기에 제한이 없으며 URL에 표시가 되지 않은 상태로 전송하므로 GET 방식보다 보안이 우수
  - ✓ PUT: 리소스 변경
  - ✓ PATCH: 리소스 일부 변경
  - ✓ DELETE: 리소스 삭제
  - ✓ HEAD: 리소스의 헤더 취득
  - ✓ OPTIONS: 리소스가 지원하는 메서드 취득
  - ✓ TRACE: Loopback 시험에 사용
  - ✓ CONNECT: 프록시 동작의 터널 접속으로 변경

Web Programming

- 상태 코드
  - ✓ HTTP 상태 코드는 클라이언트가 보낸 요청의 결과를 나타내는 세 자리 숫자
  - ✓ 이 코드는 HTTP 응답의 일부로 서버에서 클라이언트로 전송되며 클라이언트는 이를 통해 요청이 성공적으로 처리되었는지 또는 어떠한 문제가 발생했는지를 확인 가능
  - ✓ 상태 코드는 일반적으로 브라우저 개발자 도구 또는 서버 로그와 같은 도구를 사용하여 확인할 수 있는데 상태 코드는 HTTP 응답의 첫 번째 줄에 나타나며 예를 들어 HTTP/1.1 200 OK 와 같이 표시됨
  - ✓ 범주
    - 1xx: Informational(정보 제공) - 요청이 수신되었으며 프로세스를 계속 진행
    - 2xx: Success(성공) - 클라이언트의 요청이 서버에서 성공적으로 처리되었다는 의미
    - 3xx: Redirection - 완전한 처리를 위해서 추가적인 동작이 필요한 경우로 서버의 주소나 요 청한 문서가 이동된 경우
    - 4xx: Client Error - 없는 페이지를 요청하는 경우처럼 클라이언트의 요청이 잘못된 경우
    - 5xx: Server Error - 서버 측 사정에 의해서 메시지 처리에 문제가 발생한 경우

Web Programming

- ❖ 상태 코드
  - ✓ 주요 상태 코드
    - 200: OK
    - 201: Created - 요청은 처리되었고 새로운 리소스를 생성
    - 202: Accepted - 요청은 접수했지만 처리가 완료되지 않음
    - 301: Moved Permanently - 지정한 리소스가 새로운 URI로 이동
    - 303: See Other - 다른 위치로 요청
    - 307: Temporary Redirect - 임시로 리다이렉션 요청이 필요
    - 400: 잘못된 요청
    - 401: 권한이 없음
    - 403: 권한 처리 이외의 사유로 리소스에 대한 엑세스가 금지
    - 404: 지정한 리소스를 찾을 수 없는 경우
    - 500: 내부 서버 오류
    - 502: 게이트웨이나 프록시 서버 이상
    - 503: 서비스 제공 불가

Web Programming

- 웹 페이지 종류
  - ✓ 정적 페이지
    - 항상 동일한 내용을 표시하는 웹 페이지
    - 동일한 URL의 요청에 대해서는 동일한 내용을 페이지를 반환
    - HTML, CSS 만으로 구성된 경우가 많음
  - ✓ 동적 페이지
    - 동일한 리소스의 요청이라도 누가, 언제, 어떻게 요청했는지에 따라 각각 다른 내용을 반환하는 페이지
    - 현재 시간을 보여주는 페이지 라던가 쇼핑몰에서 사용자의 카트를 보여주는 페이지 등
    - 서버에 요청을 보내서 데이터를 받은 후 그 데이터를 기반으로 HTML을 생성해서 결과 페이지를 만들어 내는 방식으로 동작

Web Programming

- 요청 처리
  - ✓ Web Server
    - 웹 클라이언트의 요청을 받아서 요청에 해당하는 파일을 찾아서 전송하거나 처리를 수행하는 Application Server 나 CGI Application을 호출해서 요청을 처리한 결과를 웹 클라이언트에게 응답하는 소프트웨어
    - 정적 페이지인 HTML, 이미지, CSS, 자바스크립트 파일을 웹 클라이언트에 제공
    - Apache httpd, Nginx, lighttpd, IIS 등
  - ✓ CGI 방식
    - 정적인 HTML 문서 서비스의 한계를 극복하기 위해 등장
    - 서버 쪽에서 동적인 프로그래밍으로 실행되는 스크립트의 시초로 웹 서버가 직접 애플리케이션을 호출하여 결과를 출력하는 방식
    - 하나의 요청마다 하나의 프로세스를 생성해서 처리하며 일반적으로 컴파일 된 실행 프로그램을 호출
    - Perl, C기반 CGI 프로그램, ASP 등
  - ✓ Application Server
    - 웹 서버로부터 요청을 받아서 처리하고 그 결과를 웹 서버에게 반환하는 소프트웨어
    - 요청을 처리하기 위한 알고리즘 코드 와 데이터베이스 연동 등을 처리
    - Apache Tomcat, JBoss, WebLogic, WebSphere, Jetty, Jeus, mod_wsgi, uWSGI, Gunicom 등

Web Programming

❖ 요청 처리