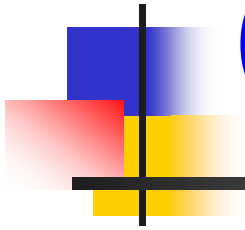
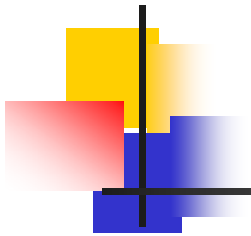


Chapter 8

데이터베이스 응용 개발

목 차

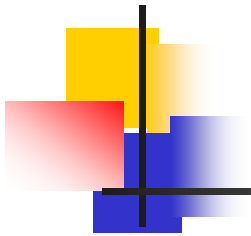


1. 사용자 인터페이스와 도구들

- 대부분의 데이터베이스 사용자들은 SQL을 사용하지 않음
  - 응용 프로그램: 사용자와 데이터베이스를 연결

- 데이터베이스 응용의 구조
  - Front-end
  - Middle Layer
  - Backend

- Front-end: User interface
  - Forms and graphical user interfaces
  - Report generators
  - Data analysis tools
  - Web 기반의 사용자 인터페이스가 널리 채택

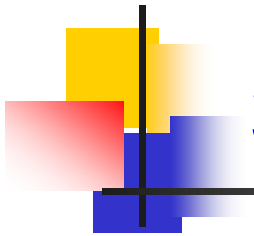


2. 웹 인터페이스와 데이터베이스

- Application architecture의 발전 과정
  - Mainframe (1960년대 ~ 1980년대 초반)
  - Personal computer (1980년대 ~ 1990년대 중반)
  - Web (1990년대 이후)

웹 인터페이스

- Web browser는 DB 응용에 대한 de-facto standard
  - Universal front end를 지원
    - 임의의 장소에 위치한 임의의 컴퓨터에서 접속 가능
    - 별도의 전용 코드를 설치할 필요가 없음
  - DB를 이용함으로써 동적 Web page 생성 가능
    - Web page = 고정된 구조(HTML) + 컨텐츠를 생성하는 프로그램(Javascript, Flash, Java Applet 등)
  - HTML과 Hyperlink의 장점 활용 가능
    - 깔끔하고 이해하기 쉬운 사용자 인터페이스
  - HTML 문서에 포함된 고사양의 프로그램 활용 가능
    - Javascript, Java Applet 등을 이용하여 HTML보다 고급의 사용자 인터페이스 구현 가능
    - Flash나 Shockwave 등을 이용한 애니메이션 기능 구현 가능



3. 웹 기초

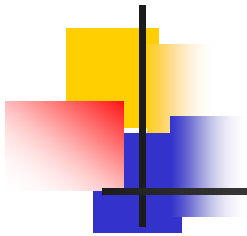
- Uniform Resource Locator (URL)
- HyperText Markup Language (HTML)
- Client-Side Scripting and Applets
- Web Servers and Sessions

3.1 Uniform Resource Locator (URL)

- URL: Web에서 포인터의 역할
- Example

  http://www.acm.org/sigmod

  - http – 문서를 액세스하는 프로토콜을 표시
  - www.acm.org – Internet에서 machine의 고유한 이름
  - URL의 나머지 부분: 그 machine에서 문서를 표시
    - Local identification
- Local Identification의 내용
  - Machine에서 문서가 위치하고 있는 경로, 또는
  - 프로그램의 이름 + 프로그램 실행을 위한 arguments
    - 예: http://www.google.com/search?q=cho



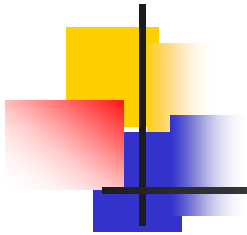
3.2 HTML

- Text formatting, hypertext link, 및 이미지 출력 등을 지원
  - 표나 style sheet 등의 기능 포함
  - 다양한 입력 기능 지원
    - Pop-up menus, radio buttons, check lists
    - Text boxes
  - 사용자의 입력을 서버에 전송하고, 서버에서 전송된 실행 결과를 사용자에게 다시 출력

- Web server와의 통신을 위하여 HTTP 사용
  - get method
    - http://www.google.com/search?q=cho
  - post method
    - http://www.google.com/ + parameter 값은 HTTP 프로토콜의 부분으로 전송

Sample HTML Source Text

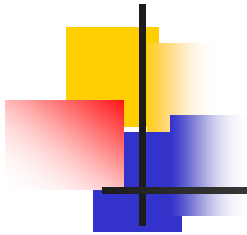
```
<html> <body>
<table border cols = 3>
      <tr><td>A-101</td><td>Downtown</td><td>500</td></tr>
      <tr><td>A-102</td><td>Perryridge</td><td>400</td></tr>
      <tr><td>A-201</td><td>Brighton</td><td>900</td></tr>
</table>
<center> The <i>account</i> relation </center>

 <form action="BankQuery" method=get>
      Select account/loan and enter number <br>
      <select name="type">
            <option value="account" selected> Account </option>
            <option> value="Loan"> Loan </option>
      </select>
      <input type=text size=5 name="number">
      <input type=submit value="submit">
</form>
</body> </html>
```



3.3 Client-Side Scripting

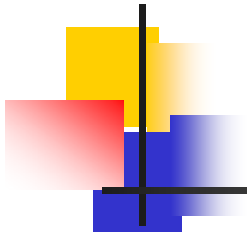
- Client의 web browser에서 실행되도록 고안된 언어
  - Javascript
  - Macromedia Flash and Shockwave for animation
  - VRML for 3D modeling
  - Java Applets

- Active web page의 구성이 가능
  - HTML의 제한을 넘어선 flexible UI 제공
    - 예: animation by executing programs at local site
  - 사용자 입력에 대한 빠른 응답이 가능
    - 입력된 값에 대한 client side의 정확성 검사
    - 대안: 입력된 모든 값들을 서버에 전송하여 검사?
  - Client-side processing을 극대화 → AJAX



3.4 웹 서버와 세션

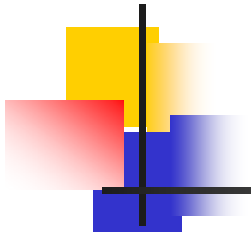
- Web Server
  - 서버에서 실행되는 프로그램
    - Web browser로부터 전송된 요청을 수신
    - 요청을 실행하기 위한 프로그램을 동작
    - 실행 결과를 HTML 형태로 browser에게 다시 반환
  - 다양한 정보 서비스에 대한 중계자 역할
    - 새로운 서비스를 지원하기 위하여 해당 프로그램을 작성한 후 등록하는 과정이 필요
  - 대표적인 web server: Apache, IIS 등
- Web server와 application program과의 연동
  - Common Gateway Interface (CGI)
  - via Application server (예: JSP/Servlet container)



HTTP와 세션(Session)

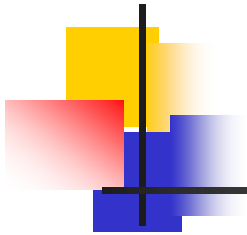
- HTTP는 비연결성 프로토콜
  - 서버가 요청에 대해 답한 후, 클라이언트와의 접속을 종료
    - 이유: 서버에 대한 부하를 줄이기 위하여
    - 동시에 접속 가능한 클라이언트 수에 제한
  - Login 정보나 JDBC/ODBC 연결의 경우, 동일한 연결에 대한 정보를 지속적으로 필요로 함
    - 모든 요청마다 login을 해야할 경우, 비효율적

- 해결 방법: Cookie



Session과 Cookie

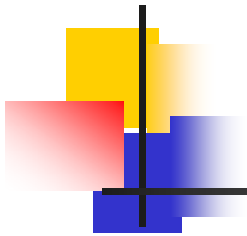
- Cookie: 문자열 정보
  - Server → Browser
    - 첫번째 접속시 생성된 세션에 관한 기록
  - Browser → Server
    - 동일한 세션에 대한 이후 interaction시 HTTP 프로토콜의 일부로 전송
  - Server는 cookie에 대한 정보를 저장
    - Request를 처리할 때 사용
    - 예: 사용자 ID 정보, 사용자 기호도 등



4. Servlets and JSP

- 데이터베이스를 이용한 Web 프로그래밍 기술
  - Java Servlets
  - Java Server Pages (JSP)
  - Active Server Pages (ASP)

- Java Servlet
  - HTTP 요청에 대한 응답을 동적으로 생성하는 서버쪽의 응용 프로그램
  - 주요 내용
    - HTML form에서 입력된 인자를 액세스
    - Business logic을 적용하여 어떤 응답을 생성할 지 결정
    - Browser에게 돌려보낼 HTML output 생성



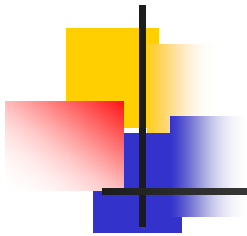
Application Server and Servlets

- Servlet은 application server 안에서 동작
  - Apache Tomcat, Glassfish, JBoss
  - BEA Weblogic, IBM WebSphere and Oracle Application Servers

- Application Server의 역할
  - 요청을 담당할 servlet 실행 및 결과를 client에게 반환
    - 각 servlet에 대한 요청 시 서버내에서 새로운 thread가 생성
  - Servlet의 배포 및 감시
    - Servlet code가 변경될 경우, recompile 후 다시 load

Example Servlet Code

```
Public class BankQueryServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse result)
        throws ServletException, IOException
    {
        String type = request.getParameter("type");
        String number = request.getParameter("number");
            ...code to find the loan amount/account balance ...
            ...using JDBC to communicate with the database..
            ...we assume the value is stored in the variable balance
        result.setContentType("text/html");
        PrintWriter out = result.getWriter( );
        out.println("<HEAD><TITLE>Query Result</TITLE></HEAD>");
        out.println("<BODY>");
        out.println("Balance on " + type + number + "=" + balance);
        out.println("</BODY>");
        out.close ( );
    }
}
```



Server-Side Scripting

- Server에서 browser에게 보낼 web page를 동적으로 생성
  - Web page: HTML + Embedded executable code
  - 해당 페이지가 요청될 때, web server는 embedded code를 실행하여 실제 HTML 문서를 생성

- Server-side scripting languages
  - JSP, PHP
  - General purpose scripting languages: VBScript, Perl, Python
  - ASP.net: Visual Basic or C#

Java Server Pages (JSP)

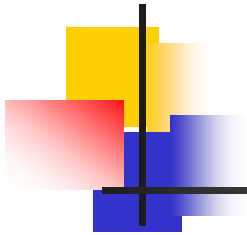
- A JSP page with embedded Java code

```
<html>
<head> <title> Hello </title> </head>
<body>
<% if (request.getParameter("name") == null)
       { out.println("Hello World"); }
else { out.println("Hello, " + request.getParameter("name")); }
%>
</body>
</html>
```

- JSP is compiled into Servlets
- Tag library의 개념 지원
  - JSP 페이지간 흐름제어(include, forward)
  - 다른 Java 객체 사용(useBean, setProperty, getProperty)

5. 대규모 웹 응용 개발

- Web application의 경우, DB 관련 작업보다 UI 구성에 많은 노력이 필요
- UI에 필요한 프로그래밍 작업을 줄이기 위한 방법들
  - 다양한 web page에 사용되는 공통된 기능들을 자동으로 생성하는 함수 제공
    - 예: 광역지자체 이름을 선택하는 메뉴
  - 날짜 입력이나 입력 데이터 검증에 사용되는 Javascript를 자동으로 생성하는 함수 제공
  - SQL query를 입력으로 받은 후, 결과를 표 형태로 출력하는 일반적인 함수 작성
  - 결과를 페이지화(pagination)하는 기능 작성



웹 서버의 성능 향상

- 인기 있는 web site의 경우, 성능이 주요 고려 대상
- 어떻게 사용자 요청을 빨리 처리할 수 있을까?

Caching

- 서버 사이트에서 수행 가능한 작업들
  - Servlet 요청들 사이에 JDBC 연결을 캐싱
    - connection pooling
  - 데이터베이스 질의의 결과를 캐싱
    - Base relation의 내용이 변경될 경우, cached result 도 수정(폐기, 재계산, incremental update)
    - SQL Cache Dependency at ASP.net
  - 생성된 HTML을 캐싱

- 클라이언트 네트워크에서 수행 가능한 작업
  - Web proxy에서 검색 페이지들을 캐싱