# サーバサイド Java を用いた Web データベースアプリケーションの試作

## A Prototype of a Web Database Application Using Server-Side Java

http://www.tani.cs.chs.nihon-u.ac.jp/g-2005/naotaka/

谷 研究室　涌井 直貴
Naotaka WAKUI

**概要**

デジタルアーカイブ・プロジェクトの一環として，Web データベースアプリケーションをサーバサイド Java を用いて試作する．尚，対象はアンケート支援システムである．

## 1　はじめに

2003 年 5 月，文理学部情報科学研究所のプロジェクト,「デジタルアーカイブ・インフラストラクチャの構築と高度利用」が，文部科学省から，2003 年度私立大学学術フロンティア推進事業の選定を受けた．これに伴い，研究拠点として，2004 年 4 月に文理学部広領域情報学研究センターが竣工された．

私立大学学術フロンティア推進事業とは，私立大学における研究基盤を強化し，日本の学術研究の推進に資するために，文部科学省が，私立大学の大学院研究科，研究所の中から，重点的研究領域ごとに，優れた研究実績をあげ，将来の研究発展が期待される卓越した研究組織のプロジェクトに対して選定し，総合的に支援するという事業のことである．

その研究目的であるデジタルアーカイブとは，有形・無形の歴史・文化資産をデジタル情報の形で保存，データベースとして蓄積し，その情報を次世代に継承を図るとともに，閲覧，鑑賞，研究のためにインターネットなどの情報ネットワークを通して情報発信するという研究である．

では，なぜデジタルアーカイブが必要なのかというと，資料の寿命が物理的条件に左右されるからである．例えば，紙に書かれている資料の寿命は，紙の寿命に依存する．また，原本はひとつしか存在しないため，閲覧が難しい．たとえ複製物を作ったとしても，それがアナログデータであるかぎり，同じ問題点を持つことになる．そこで，資料をデジタル化することによって，半永久的に利用でき，ネットワーク上のすべての人に閲覧が可能になるのである．

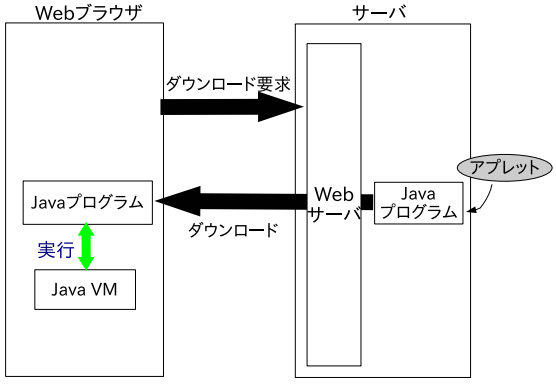
広領域情報学研究センターの基盤システムは，文理学部情報科学研究所が所有する日本語貴重資料，中国文書資料， 中国人の非言語伝達行動に関する資料，ポリグラフ検査，史的自然災害記録， 農林業・農山村社会の地域情報，大脳皮質の機能などに関わるデータを蓄積し，それを検索・ 閲覧することができる．これを利用するによって，他の研究に役立つのである．

本研究は，このプロジェクトの一環として，また，サーバサイド Java のノウハウの蓄積を目的として，デジタルアーカイブの基盤となるシステム，Web データベースアプリケーション（以下，Web アプリ）を，サーバサイド Java で試作する．尚，対象は，アンケートを集計，結果を表示させる，アンケート支援システムである．

## 2　サーバサイド Java

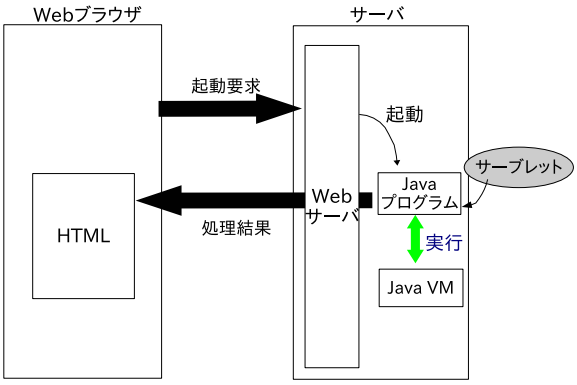
サーバサイド Java を分かりやすく説明するために，アプレットと比較して紹介する．

アプレットとは，Web ブラウザ上で動作する Java プログラムである．ネットワーク上のサーバに置かれた Java プログラム (アプレット) を，Web ブラウザが自動的にダウンロードし，Web ブラウザに組み込まれた Java Virtual Machine（Java プログラムを実行するために必要な仮想的なマシンのこと．以下，JVM）上で動作する．アプレットの動作には，JVM を組み込んだ Web ブラウザが必須である．




アプレットの動作概要

一方，サーバサイド Java とは，アプレットとは違い，クライアント側で動作するのではなく，クライアントからの要求を受け取ってサーバ側で動作する Java プログラムのことである．尚，サーバ側で動作するため，JVM が組み込まれていない Web ブラウザでも動作する．以下は，サーバサイド Java の一つであるサーブレットの動作概要である．




サーブレットの動作概要

サーバサイド Java で Web アプリを作成するにあたって，最低限必要なソフトウェアは以下の通りである．

・Java 開発環境 JDK
・Web サーバ
・アプリケーションサーバ（サーブレットコンテナ）
・リレーショナルデータベース管理システム（RDBMS）
・JDBC

この他，メール機能を利用するための API，JavaMail などがある．

この中で核となるものがアプリケーションサーバである．クライアントからの要求を受けると，Java クラスファイル（Java ソースコードをコンパイルしたもの．拡張子は ”.class”）を JVM に実行させ，処理結果をクライアントに返す．

尚，本研究では，Web サーバとして Apache，アプリケーションサーバとして Tomcat を使用する．

## 2.1 サーブレット（Servlet）

サーブレットとは，Java コードの中に HTML コードを記述する形式のサーバサイド Java である．見た目は Java プログラムそのものであることから，複雑な処理に適していて，逆に表示部が冗長になりやすい欠点を持つ．

```
import java.io.IOException;
import java.io.PrintWrier;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class ○○○ extends HttpServlet{
  public void doGet(HttpServletRequest req,
    HttpServletResponse res)
  throws ServletException, IOException{

  res.setContentType( “text/html;charset=UTF-8” );
  PrintWrier out = res.getWriter();
  out.println( “<html><body>” );
  out.println( “<h1>サーブレット</h1>” );
  out.println( “</body></html>” );

  out.flush();
  out.close();
  }
}
```

ここが冗長になりやすい

サーブレットのソースコード

アプリケーションサーバがサーブレットを実行するためには，人（サーバの管理者など）があらかじめコンパイルをし，サーバの所定の位置に配置しなければならない．

## 2.2 JSP（Java Server Pages）

一方 JSP は，HTML コードの中に JSP コード（JSP のタグや Java コード）を記述する形式のサーバサイド Java である．

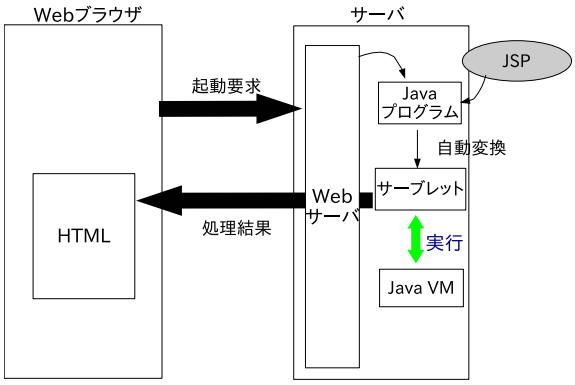
つまり，通常の HTML はそのまま記述し，動的に変更する部分を JSP コードで記述するのである．ただ，HTML と JSP コードが混在するため，複雑な処理を記述するとソースが見にくくなる．

```
<%@page contentType=” text/html; charset=UTF-8” %>

<html>
 <body>
   <% out.println( “JSP” ); %>
 </body>
</html>
```

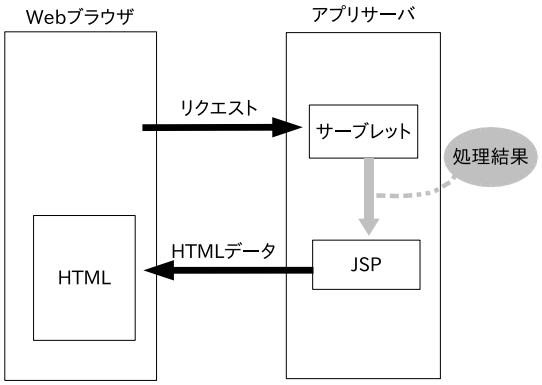
JSP のソースコード

JSP はサーバの所定の位置に配置さえすれば，アプリケーションサーバが一旦サーブレットに変換したのち，自動でコンパイル・実行をしてくれる．




JSP の動作概要

サーブレットと JSP は，プログラム間でのデータのやり取りが非常に容易である．そこで，互いの長所を合わせ，処理などのプログラム・ロジックはサーブレットで，結果の表示は JSP で行う，という役割分担が一般的である．




連携の概要

## 2.3 JDBC

Web アプリは，一度に多くのリクエストを処理することになる．その際，高速で信頼性の高いデータアクセスが必要である．それを実現する仕組みとして，一般にデータベースが用いられる．また，デジタルアーカイブにおいては，デジタルデータを蓄積するためにデータベースとの連携は必要不可欠である．

そのデータベースを操作するコマンドは，データベース管理システム（以下，DBMS）によって異なる．そのため，プログラム中からデータベースを操作するには，対象の DBMS に応じてコマンドを書き分ける必要が生じる．DBMS が変わるたびにプログラム中のコマンドを修正するのは面倒である．

そこで，サーバサイド Java には，JDBC（Java DataBase Connectivity）という API が存在する．DBMS に対応した JDBC ドライバを使えば，DBMS に依存した処理をプログラムに記述する必要はなくなる．また，DBMS を変更する場合，JDBC ドライバを変更するだけでよいのである．

## 2.4 利点

Web アプリの作成にサーバサイド Java を使ったときの利点を，PHP との比較で簡単な表にした．

| | PHP | **サーブレット** |
|---|---|---|
| 処理方式 | interpreter | compile ＋ interpreter |
| コンパイル | 不要 | 必要 |
| 1 度目の起動 | ふつう | 遅い |
| 2 度目以降 | ふつう | はやい |

PHP は，インタプリタ（逐次翻訳）型と呼ばれる処理方式で，実行のつど，逐一コードを解析し，翻訳されたものから順に実行されるというものである．この方式は，コンパイルが不要であるという簡便さはあるが，実行局面においては，決して効率の良いものではない．

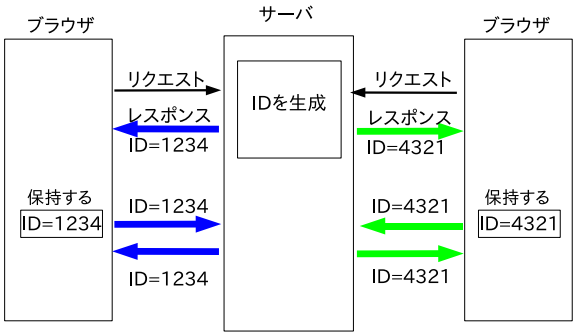
これに対し Java はコンパイル&インタプリタであり，さらに処理効率が悪いように思えるが，サーブレットは一度実行されるとメモリに常駐するため，高速な処理が可能である．

また，元々Java はネットワーク関連の機能が豊富であり，サーブレットにも Cookie やセッションなどのネットワークに関連した機能がクラスとして既に用意されており，他の言語と比べて Web アプリの作成が比較的容易である．

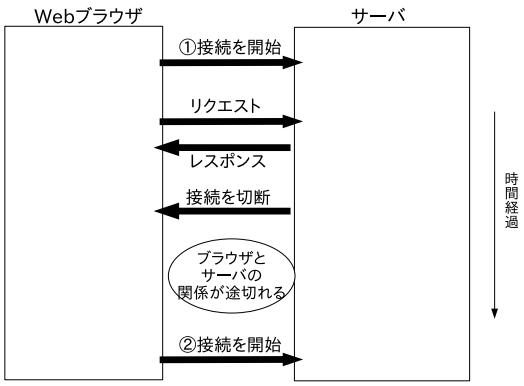
さらには，Java は『Write Once, Run Anywhere』を基本理念にしていて，一度書けばどこでも動くというように，JVM のおかげでプラットホームを選ばないので，Windows 用に開発したものがそのまま，UNIX/Linux や Macintosh，携帯電話などで動作させることができ，生産性が高い点も利点の一つである．

## 3　セッション管理

セッション（Session）とは，ある特定のクライアントがサーバと接続され，データを送受信している状態のことである．しかし，クライアントがサーバにリクエストを送り，サーバからレスポンスが返ってくると，その時点でセッションは終了し，それ以降は関係が維持されない．HTTP には，2 つ以上のリクエストが同じクライアントかどうかを確認する情報はないのである．




セッション管理の仕組み




セッションの様子
1 と 2 が同じクライアントかどうかは分からない

同じクライアントの複数のリクエストにまたがる処理を，一つのセッションとして成立させたい．また，クライアントとクライアントを区別したい．そこで，サーバがどのクライアントと接続しているかという情報を保持し，クライアント側にはクライアントごとにセッション ID とよばれる文字列を生成し，Cookie の形で保持させる．そうすることにより，複数のリクエストにわたって，一つのセッションを成立できる．このような仕組みのことをセッション管理という．

最初のリクエストが来たら，サーバは自動的に ID を生成し，レスポンスとともにクライアントに送信する．以降は，サーバ側の ID と，cookie の ID を照合してセッションを確立させる．

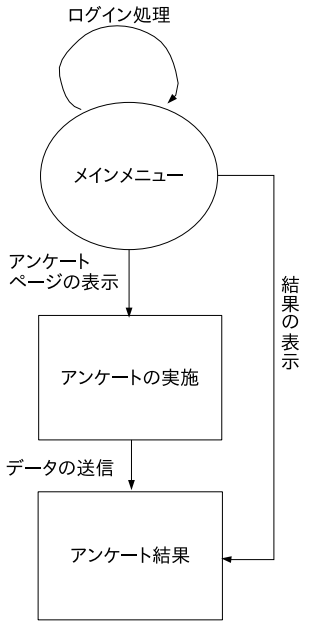
尚，サーバサイド Java は，このセッション管理も容易に実装できる．

## 4　試作物の紹介

今回，私が試作したアプリケーションは，同研究室の学生の支援システムである．

その学生は，キーワードを入力すると，そのキーワードに関する Web ページの URL が出力されるあるアルゴリズムの，改善しうる異なる 2 方式のアルゴリズムを提唱し，実装を行った．その 3 つのプログラムの実行結果と，検索エンジンの実行結果を足した 4 つの実行結果を比較し，自分のプログラムの評価をしたい．そこで，同研究室生を対象に，キーワードごとの各プログラムの実行結果に対して順位をつけるアンケートを行い，評価する．そのアンケートを支援する，アンケートページの作成・集計・結果表示をするシステムを製作した．

大まかな流れは以下の通りである．




試作物の流れ

## 参考文献

[1] デジタルアーカイブ推進協議会 http://www.jdaa.gr.jp/
[2] 株式会社アイ・ディ・ジー・ジャパン (発行) Java World for Beginners
[3] 山田 祥寛 (著) 10 日でおぼえる JSP/サーブレット入門教室
[4] 上ヶ迫 勝憲 (著) PostgreSQL & MySQL 逆引き大全 456 の極意
[5] 山田 祥寛 (著) JSP & サーブレット スーパーリファレンス