From Services to Servlets

Servlets are dynamically loaded Java classes/objects invoked by a Web server to process requests.

- Servlets are to servers as applets are to browsers.
- Servlet support converts standard Web servers into extensible “Web application servers”.
  
  designed as a Java-based replacement for CGI
  
  Web server acts as a “connection manager” for the service body, which is specified as pluggable servlets.
  
  interface specified by JavaSoft, supported by major Web servers

- Servlets could be used in a variety of kinds of servers.

Invocation triggers are defined by server; the servlet does not know or care how it is invoked.

Anatomy of a Servlet

network service

GenericServlet

Servlet

ServletRequest

ServletContext

String getServerInfo()
Object getAttribute(name)
String getMimeType(filename)
log(string)

init(ServletConfig config)
String getServletInfo()
service(....)
destroy()

(String getInitParameter(name)
ServletContext getServletContext()
Enumeration getInitParameterNames()
Invoking a Servlet

```
import java.io.*;
import javax.servlet.*;
public class HelloWorld extends GenericServlet {
    public void service(ServletRequest request, ServletResponse response)
        throws ServletException, IOException {
        ...}
    public String getServletInfo() {
        return "Hello World Servlet";
    }
}
```
HelloWorld Servlet (continued)

```java
public void service(ServletRequest request, ServletResponse response)
    throws ServletException, IOException
{
    ServletOutputStream output = response.getOutputStream();
    String fromWho = request.getParameter("from");

    response.setContentType("text/html");
    if (fromWho == null) {
        output.println("<p>Hello world!");
    } else {
        output.println("<p>Hello world from <em>" + fromWho + "</em>" + fromWho + "</em>");
    }
}
```

Example 1: Invoking a Servlet by URL

Most servers allow a servlet to be invoked directly by URL.
- client issues HTTP GET
  e.g., http://www.yourhost/servlet/HelloWorld
- servlet specified by HTTP POST
  e.g., with form data

```html
<FORM ACTION="http://yourhost/servlet/HelloWorld" METHOD="POST">
From : <INPUT TYPE="TEXT" NAME="from" SIZE="20">
<input type="submit" value="Submit"> </FORM>
```

generates a URL-encoded query string, e.g., "<servletURL>?from=me"
Example 2: Server-Side Include

Servlets might be invoked on the server side as a side effect of accessing an HTML page, transparently to the client.

```html
<HTML>
<HEAD><TITLE>Hello World SSI</TITLE></HEAD>
<BODY>

<H1>A Server Side Include Sample</H1>
<SERVLET NAME="HelloWorld">
<PARAM NAME="from" VALUE="me">
</SERVLET>

</BODY>
</HTML>
```

The servlet’s input is an empty stream; its output replaces the body of the `<SERVLET>` block in the HTML returned to the client.

Constant parameters are passed explicitly from HTML (init parameters may be specified in SERVLET block).

**HTTP Servlets**

- **GenericServlet**
  - `service(...)`
  - `doGet()`, `doHead()`, `doPost(...)`

- **HttpServlet**
  - `addCookie()`, `setStatus(code, msg)`, `setHeader(name, value)`, `sendRedirect()`, `encodeUrl()`

- **HttpServletResponse**
  - `getCookies()`, `getRemoteUser()`, `getAuthType()`, `getHeader(name)`, `getHeaderNames()`, `getSession()`

- **ServletRequest**
HTTP Sessions

TinyServer Overview
What You Will Do for Project #1

1. networking code
   listener code in TinyServer
   stream and port management in ServerConnection

2. servlet management
   loading/instantiating/caching servlets
   parameter methods in ServerConnection (HttpServletRequest)

3. session management
   generating sessions and keeping track in SessionManager

4. some sample servlets
   default servlet (for static pages), “interesting” servlet