Sun Certified Web Component Developer

The Servlet Model

1. HttpServlet must override doGet for GET, doPost for POST, doPut for PUT.
2. Params are HttpServletRequest and HttpServletResponse
3. GET – URL in browser, method="GET" in form, and when no method specified in form.
4. POST – when method="POST" in form, unlimited length.
5. HEAD – could bytes, headers only, not body. Could use to check if page has changed.
6. ServletConfig <<Interface>>
    - getInitParmeter(String) – String
    - getInitParameterNames() – enum
    - getServletContext() – ServletContext
7. ServletContext <<Interface>>
    - getAttribute(String) – String
    - getAttributeNames() – enum
    - setAttribute(String,Object)
    - removeAttribute(String) – same as setAttribute(String,null)
    - getInitParmeter(String) – String
    - getInitParameterNames() – enum
    - getRequestDispatcher() – RequestDispatcher
    - getResourceAsStream(String) – InputStream
    - getResource(String) – URL
    - log(String)
    - log(String,Throwable)
8. ServletRequest <<Interface>>
    - getParameter(String) – String
    - getParameterValues(String) – String[]
    - getParameterNames() – enum
    - getAttribute(String) – Object
    - getAttributeNames() – enum
    - setAttribute(String,Object)
    - removeAttribute(String) – same as setAttribute(String,null)
    - getInputStream() - ServletInputStream
    - getRequestDispatcher() - RequestDispatcher
9. HttpServletRequest <<Interface>> extends ServletRequest
    - getHeaders(String) – enum
    - getHeader(String) – String
    - getDateHeader(String) – long
    - getIntHeader(String) - int
    - getHeaderNames() – enum
    - getSession(boolean) or getSession() same as getSession(true) – HttpSession
10. ServletResponse <<Interface>>
    - getOutputStream() – ServletOutputStream, inherits from java.io.OutputStream
    - getWriter() – java.io.PrintWriter, flushing the PrintWriter commits it.
    - setContentType(String) – eg: "text/html;charset=ISO-8859-4"
11. HttpServletResponse <<Interface>>
    - addCookie(Cookie)
    - addHeader(String name,String value) – add not replace
    - addDateHeader(String,long)
    - addIntHeader(String,int)
    - encodeRedirectURL(String) – add session params
    - encodeURL(String)
    - sendRedirect(String) – can be relative or absolute, client side redirect
    - setDateHeader(String,long) – update not add
    - setHeader(String,String)
    - setIntHeader(String,int)

- sendError(int) – error code integer
- sendError(int,String) – error code and message
- setStatus(int) – set the status code

12. HttpSession <<interface>>
    - getAttribute(String) – Object
    - getAttrbuteNames() – Enum
    - getMaxInactiveInterval() – int, in seconds
    - invalidate() – kill session
    - removeAttribute(String) – same as setAttribute(String,null)
    - setAttribute(String,Object)
    - setMaxInactiveInterval(int) – in secs

Serlvet Lifecycle Methods

- init – once when loaded, like a static but not
- servce – per request
- destroy – when unloaded

Server Side Dispatching

```
// the URL must start with / and used before committing the response
RequestDispatcher rd = getServletContext().getRequestDispatcher("url");
rd.forward(req,res); // server side forward
rd.include(req,res); // server side include
```

The Structure and Deployment of Modern Servlet Web Applications

WAR file

- /xyz.jsp
- /abc.html
- /WEB-INF/web.xml
- /WEB-INF/classes/abc.class
- /WEB-INF/classes/com/abc/abc.class
- /WEB-INF/lib/abc.jar

web.xml

```
<servlet>
        <servlet-name>abc</servlet-name>
        <servlet-class>com.abc.Abc</servlet-class>
        <init-param>
                <param-name>cat</param-name>
                <param-name>apple</param-name>
        </init-param>
</servlet>

<servlet-mapping>
        <servlet-name>abc</servlet-name>
        <url-pattern>/at/*</url-pattern>
</servlet-mapping
```

The Servlet Container Model

```
<context-param>
        <param-name>xyz</param-name>
        <param-value>abc</param-value>
        <description>bla bla</description>
</context-param>
```

Init Param

- ServletContext.getInitParameter(String)
- ServletContext.getInitParameterNames() – enum

Listeners, specified in the web.xml

- ServletContextListener – chgs to servlet context, contextInitialized(Event), contextDestroyed(Event)
- ServletContextAttributeListener – chgs to attributes in servlet context, attributeAdded(Event), attributeRemoved(Event), attributeReplaced(Event).
- HttpSessionAttrbuteListener – chgs to session attributes, attributeAdded(Event), attributeRemoved(Event)
- HttpSessionListener – chgs to sessions, sessionCreated(Event), sessionDestroyed(Event)
- HttpSessionActivationListener – activation/passivation events, sessionDidActivate(Event), sessionWillPassivate(Event)
- HttpSessionBindingListener – when bound and unbound from session, valueBound(Event), valueUnbound(Event).

Register in web.xml

```
<web-app>
        <listener>
                <listener-class>com.apple</listener-class>
        </listener>

        <servlet>
                …
        </servlet>
</web-app>
```

[will register all listener I/Fs the class implements]

Distributed

```
<web-app>
        <distributable>
</web-app>
```

HttpSession and ServletContext are distributed, the Listener instances aren't distributed so don't store variables in them. Also, instance variables of servlets aren't either.

Designing and Developing Servlets to Handle Server-side Exceptions

In code

- HttpServletResponse.sendError(int) – the response must be uncommitted, it does the commit and sets the HTTP code.

- HttpServletResponse.sendError(int,String) – as above but creates a default "text/html" page with message in it. However, if <error-page> in web.xml it will use that instead.

- HttpServletResponse.setStatus(int) – normally for success, response.SC_OK or response.SC_MOVED_TEMPORARILY.

In web.xml

```
<error-page>
```

```
        <error-code>404</error-code>
        <location>/404.html</location>   - must start with /
    </error-page>

    <error-page>
        <exception-type>com.abc.AbcException</exception-type>
        <location>/abc.html</locatgion>
    </error-page>
```

[note: can use RequestDispatcher to forward to error page but need to set status_code, exception, request_uri, servlet_name in the request as attributes]

Logging

- GenericServlet.log(Stirng)
- ServletContext.log(String) – same as above
- GenericServlet.log(String,Throwable)
- ServletContext.log(String,Throwable) – same as above

Designing and Developing Servlets Using Session Management

See HttpSession methods earlier.

Timouts

HttpSession.getMaxInactiveInterval() – int, in seconds
HttpSession.setMaxInactiveInterval(int) – in seconds

Or in web.xml..

```
<web-app>
    <session-config>
        <session-timeout>30</session-timeout>  - in mins
    </session-config>
</web-app>
```

Maintaining the session

HttpServletResponse.encodeURL(String) – add's jsessionid param if no cookies
HttpServletResponse.encodeRedirectURL(String) – as above for client side re-direct responses.

Designing and Developing Secure Web Applications

Definitions

Authentication – who they claim to be
Authorization – allowed to do operation
Data Integrity – data not modified/seen, eg: use SSL
Auditing – log of activities
Mallicious Code – code that attacks client from a site, or code that attacks site, eg: buffer overflow
Web Site Attack – hackers.

Security Constraint

- Declaritive protection
- Web Resource Collection, URL patterns and HTTP methods to describe what protecting
- Authorization Constraint, roles with access
- User Data Constraint, transport layer, the requirements of client/server communication.

```
<security-constraint>
    <web-resource-collection>
        <web-resource-name>app</web-resource-name>
        <description>my app</description>
        <url-pattern>/*</url-pattern>
```

```
                <http-method>GET</http-method>
                <http-method>POST</http-method>
            </web-resource-collection>
            <auth-constraint>
                <description>arse</description>
                <role-name>mgr</role-name>
            </auth-constraint>
            <user-data-constraint>
                <description>bla bla</description>
                <transport-guarantee>NONE|INTEGRAL|CONFIDENTIAL</transport-guarantee>
            </user-data-constraint>
        </security-constraint>
```

None – no guarantees, Integral – data can't be chgd, Confidential – not observed or chgd.

Login

```
        <login-config>
            <auth-method>BASIC|DIGEST|FORM</auth-method>
            <realm-name>abc</realm-name>
        </login-config>
```

Security Role

```
        <web-app>
            <security-role>
                <role-name>manager</role-name>
            </security-role>

            <servlet>
                <security-role-ref>
                <role-name>MGR</role-name>
                <role-link>manager</role-link>
            </servlet>
        </web-app>
```

now can use 'MGR' in code and it refers to 'manager' role.

HTTP Basic

- Not secure, base64 encoded
- Authenticates user is in realm
- Browser caches auth for future
- Pop-up dialog box

HTTP Digest

- As above
- Password encrypted
- Not supported by may browsers

FORM

- Custom login/error screens
- Form action=j_security_check
- Form parameters=j_username, j_password
- Passwords as plain text
- Sessions must be cookies, encoded URLs or SSL sessions

```
    <form-login-config>
        <form-login-page>/login.jsp</form-login-page>
        <form-error-page>/err.jsp</form-error-page>
    </form-login-config>
```

CLIENT-CERT

- Uses SSL
- Public/private key encryption

Designing and Developing Thread-safe Servlets

Variables that are thread safe
- Local variables – yes
- Instance variables (attributes) – no unless SingleThreadModel
- Class variables – no and no if STM (above) too
- Request attributes – yes
- Session attributes – no
- Context attributes – no

SingleThreadModel
- No concurrent requests
- May poll lots of instances
- Not apply to objects such as HttpSession, ServletContext, etc
- Servlet – implement SingleThreadModel
- JSP - <%@page isThreadSafe="false" %>

The Java Server Pages (JSP) Technology Model

Directive
- <%@ bla bla %>
- <jsp:directive.xxxx> </jsp:directive.xxxx>, eg: <jsp:directive.include>…</jsp…>
- control translation and compilation.
- <%@page import="…"%> - import at compiler time
- <%@page session="true"%> - page is in a session
- <%@page errorPage="…"%> - uses error page
- <%@page isErrorPage="true"%> - this is an error page, therefore access to 'exception' variable.

Declaration
- <%! Bla bla %>
- <jsp:declaration> </jsp:declaration>
- declare variables and methods
- can override methods, eg: jspInit(), jspDestroy()

Scriptlet
- <% bla bla; %>
- <jsp:scriptlet> </jsp:scriptlet>
- code that is part of the _jspService() method

Expression
- <%= abc %> - must be suitable to place in out.println(….) brackets, ie: no ';'s.
- <jsp:expression> </jsp:expression>

Phases
1. translation
2. compilation
3. class loading
4. instantiation
5. call jspInit
6. call _jspService
7. call _jspDestroy

Objects
- request – javax.servlet.ServletRequest
- response – javax.servlet.ServletResponse
- out – JspWriter, outputStream

- session – javax.servlet.http.HttpSession, only if @page session="true" or not specified
- config – javax.servlet.ServletConfig
- application – javax.servlet.ServletContext
- page – instance of class for Jsp, 'this'.
- PageContext – environment, has servlet objects, handy-as!
- exception – current exception, only if @page isErrorPage="true"

Designing and Developing Reusable Web Components

Include directive
- <%@include file="…"%>
- translation time
- parsed by container

Include action
- <jsp:include page="…"%>
- request time
- not parsed by container, response simply added to jsp's response

Designing and Developing JSP pages Using JavaBean Components

Declare JavaBean

```
<jsp:useBean id=".." class=".." scope=".."/>
```

- class must always be fully qualified, even if included.
- Scope – defaults to page if omitted.
- Also can initialise in body, eg:

```
<jsp:useBean id=".." class="..">
        <jsp:setProperty name=".." value=".."/>
</jsp:useBean>
```

Set Attribute
```
<jsp:setProperty name=".." property=".." value=".."/>
```

Get Attribute
```
<jsp:getProperty name=".." property=".."/>
```

Designing and Developing JSP pages Using Custom Tags

Declaring
```
<web-app>
        <taglib>
                <taglib-uri>http://bla</taglib>
                <taglib-location>/WEB-INF/abc.tld</taglib-location>
        </taglib>
</web-app>
```

in JSP page..

<%@taglib uri="http://bla" prefix="abc"%>

Using

- empty - <abc:tagName />
- attrs - <abc:tagName2 attr="pete"/>
- body –

```
<abc:tagName3>
        content
</abc:tagName3>
```

- Nested

```
<abc:tagName4>
        <abc:tagName5/>
        <abc:tagName6/>
</abc:tagName4>
```

Designing and Developing a Custom Tag Library

TLD file

```
<tag>
        <name>msg</name>
        <tag-class>com.arse.msgTag</tag-class>
        <tei-class>com.arse.extraInfo</tei-class>
        <body-content>empty|JSP|tagdependent</body-content>
        <attribute>
                <name>msg</name>
                <required>true</required>
                <rtexprvalue>false</rtexprvalue>
                <type>java.lang.Date</type>
                <description>message value</description>
        </attribute>
</tag>
```

-tag-class = class that implements javax.servlet.jsp.tagext.Tag interface.
-tei-class = class that provides extra type information.
-body-content = tagdependent = passed verbatim, eg: SQL, jsp = JSP syntax, empty = no body.
-required = true|false|yes|no, false is default.
-rtexprvalue = run-time expression value, calculated at run-time.
-type = if rtexprvalue is false then java.lang.String, else any class.

Note: pre-1.2 DTD, 'body-content' was 'bodycontent'.

Tag lifecycle methods

```
<abc:myTag>         - doStartTag()
        body
<                   - doAfterBody()
/abc:myTag>         - doEndTag()
```

doStartTag

- EVAL_BODY_INCLUDE – deal with body
- SKIP_BODY – ignore body
- EVAL_BODY_BUFFERED – new body content

doAfterBody

- EVAL_BODY_AGAIN – iterate, re-evaluate tag
- SKIP_BODY – stop iteration

doEndTag

- EVAL_PAGE – carry on and evaluate page
- SKIP_PAGE – skip rest of page, eg: a forward

Methods

PageContext.getOut() – get the current JspWriter stream being written to.
PageContext.findAttribute(String) – find in all variables starting with most granular, eg: req, sess, app, etc.
PageContext.getAttributeNamesInScope() – enum
PageContext.getRequest() – ServletRequest
PageContext.getResponse() – ServletResponse
PageContext.getSession() – HttpSession
PageContext.getServletContext() – ServletContext

Tag.getParent() – get the enclosing Tag
Tag.findAncestorWithClass(Tag from, Class klass) – get the parent Tag specified

Patterns

- ValueObject – reduced network traffic, faster responses, less chatty
- MVC – Model/View/Controller, separate modelling issues from display, multi-views easily, greater extensibility, easier distribution.
- DAO – Data Access Object, abstract data sources, vendor independence, easier migration to CMP.
- DB – Business Deligate, decomple presenation and services tiers, exception translation, performance (caching), easy interface and expose client to remote I/F.

WEB.XML DTD

```
web-app
    icon?
        small-icon?
        large-icon?
    display-name?
    description?
    distributable?
    context-param?
        param-name
        param-value
        description?
    filter*
        icon?
            small-icon?
            large-icon?
        filter-name
        display-name?
        description?
        filter-class
        init-param*
            param-name
            param-value
            description?
    filter-mapping*
        filter-name
        (url-pattern | servlet-name)
    listener*
        listener-class
    servlet*
        icon?
            small-icon?
            large-icon?
        servlet-name
        display-name?
        description?
        (servlet-class | jsp-file)
        init-param*
            param-name
            param-value
            description?
        load-on-startup?
        run-as?
            description?
            role-name
        security-role-ref*
            description?
            role-name
            role-link?
    servlet-mapping*
        Servlet-name
        url-pattern
    session-config?
        session-timeout?
    mime-mapping*
        extension
        mime-type
    welcome-file-list?
        welcome-file+
```

```
    error-page*
        (error-code | exception-type)?
        location
    tag-lib*
        taglib-uri
        taglib-location
    resource-env-ref*
        description?
        resource-env-ref-name
        resource-env-ref-type
    resource-ref*
        description?
        res-ref-name
        res-type
        res-auth
        res-sharing-scope?
security-constraint*
    display-name?
    web-resource-collection+
        web-resource-name
        description?
        url-pattern*
        http-method*
    auth-constraint?
        description?
        role-name*
    user-data-constraint?
        description?
        transport-guarantee
login-config?
    auth-method?
    realm-name?
    form-login-config?
        form-login-page
        form-error-page
security-role*
    description?
    role-name
env-entry*
    description?
    env-entry-name
    env-entry-value?
    env-entry-type
ejb-ref*
    description?
    ejb-ref-name
    ejb-ref-type
    home
    remote
    ejb-link?
ejb-local-ref*
    description?
    ejb-ref-name
    ejb-ref-type
    local-home
    local
    ejb-link?
```