SUNBEAM

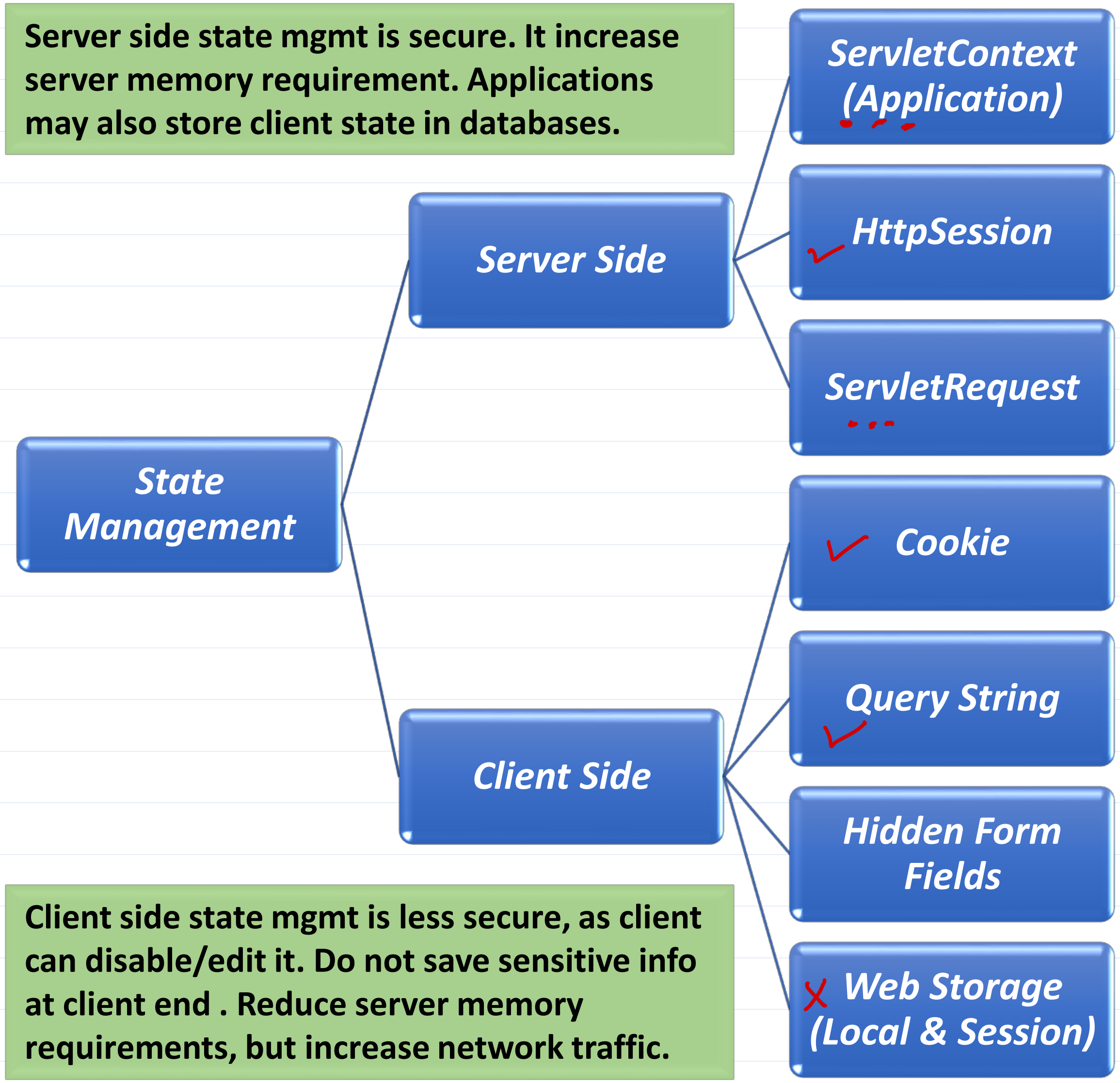
Java EE™

Advanced Java

Trainer: Nilesh Ghule

Java Web Applications – State management – at a glance

Server side state mgmt is secure. It increase server memory requirement. Applications may also store client state in databases.



State Management

Server Side

- ServletContext (Application)
- ✓ HttpSession
- ServletRequest

Client Side

- ✓ Cookie
- ✓ Query String
- Hidden Form Fields
- ✗ Web Storage (Local & Session)

Client side state mgmt is less secure, as client can disable/edit it. Do not save sensitive info at client end . Reduce server memory requirements, but increase network traffic.

ServletContext is created one for the application. It stores all meta-information like context param and attributes (set of key-value pairs). Accessible in all web components for all requests of all clients.

HttpSession is created one ~~for the application~~ each client.. It stores attributes (key-value pairs) for each client. Accessible in all web components for all requests from the same client. Session tracking is done using cookie or URL rewriting or SSL.

ServletRequest is created one for the application. It stores headers and attributes (key-value pairs) for the request. Accessible in all web component to which same request is forwarded/included (using RequestDispatcher).

Cookie is text data (key-value pair) sent from application to client. Client (browser) send it back to the server with each request. Cookies stored in browser memory and disappear when browser is closed. Persistent cookies are stored on client disk.

Query string is key-value pair appended to the URL like url?key=value. Commonly used in master detail pages to load details of selected items from the master.

Data can be stored in HTML <input type="hidden" ...>. It will be sent back to server when HTML form is submitted. Typically used to keep identity of currently selected/displayed item.

Application stores key-value data in client browser. LocalStorage has no expiration date, while SessionStorage expires when browser is closed. Both are managed in client-side JS code using window.localStorage & window.sessionStorage.

Session

HttpSession is key-value map stored on server side for each user.

To create/access session:

session = req.getSession();

To store key-value in session:

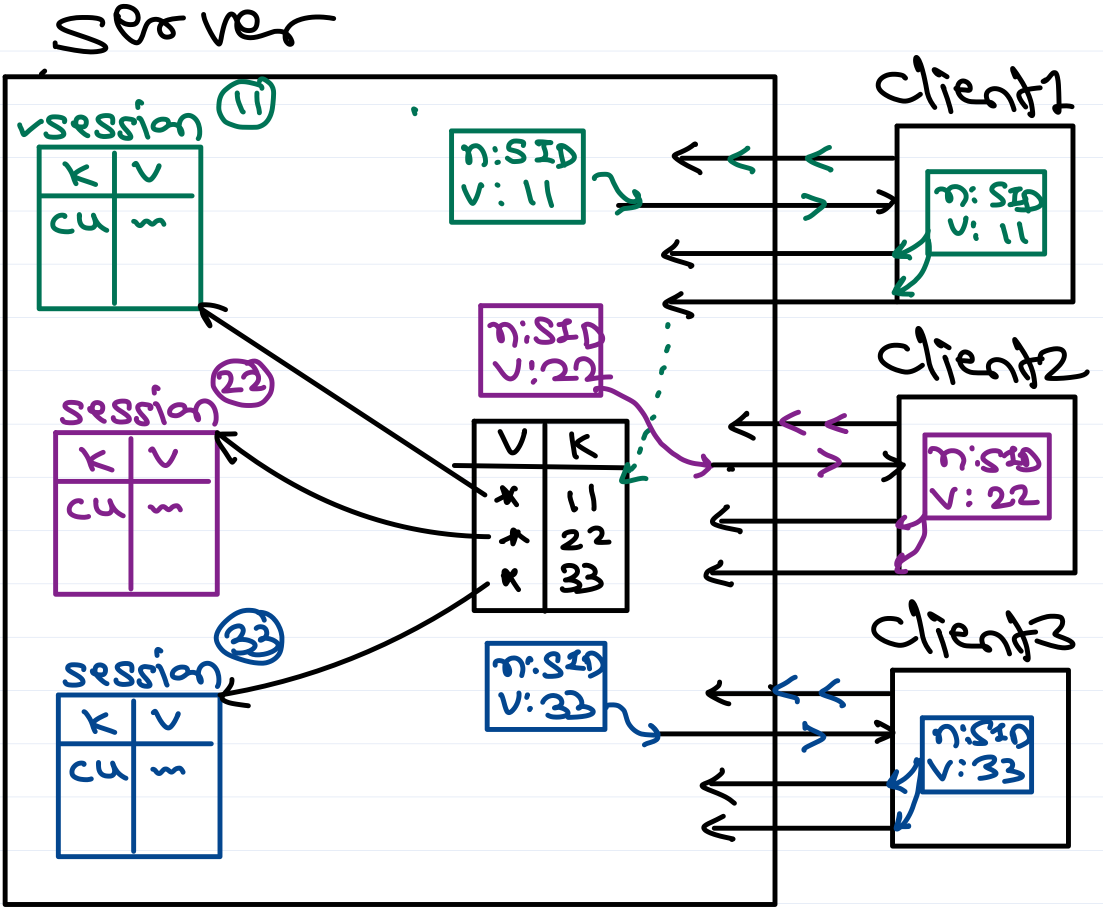
session.setAttribute("key", value);

To retrieve key-value from session:

value = session.getAttribute("key");

To destroy the session:

session.invalidate();



Session tracking using Cookie (JSESSIONID).

Session

session = req.getSession();

① check if session id cookie is present in current request. If available, find the session object corresponding to that id and return it.

② if session id cookie doesn't exist, create a new session and save it with new session id (in server internal mem). Create a new session id cookie and send it to client with next response.

Session tracking modes:

```
web.xml
    <session-config>
        <tracking-mode> COOKIE </tracking-mode>
                          or
                          URL
    </session-config>
```

Session tracking by URL rewriting

http://lh:8080/app/url

after re-writing

http://lh:8080/app/url;jsessionid=xxxxx

programmer need to rewrite each url in appln (form action, a href, resp.sendRedirect, ...) using

① reurl = resp.encodeURL(url);

OR

② reurl = resp.encodeRedirectURL(url);

QueryString

when req next page, the page url can be appeded with additional key-value pairs.

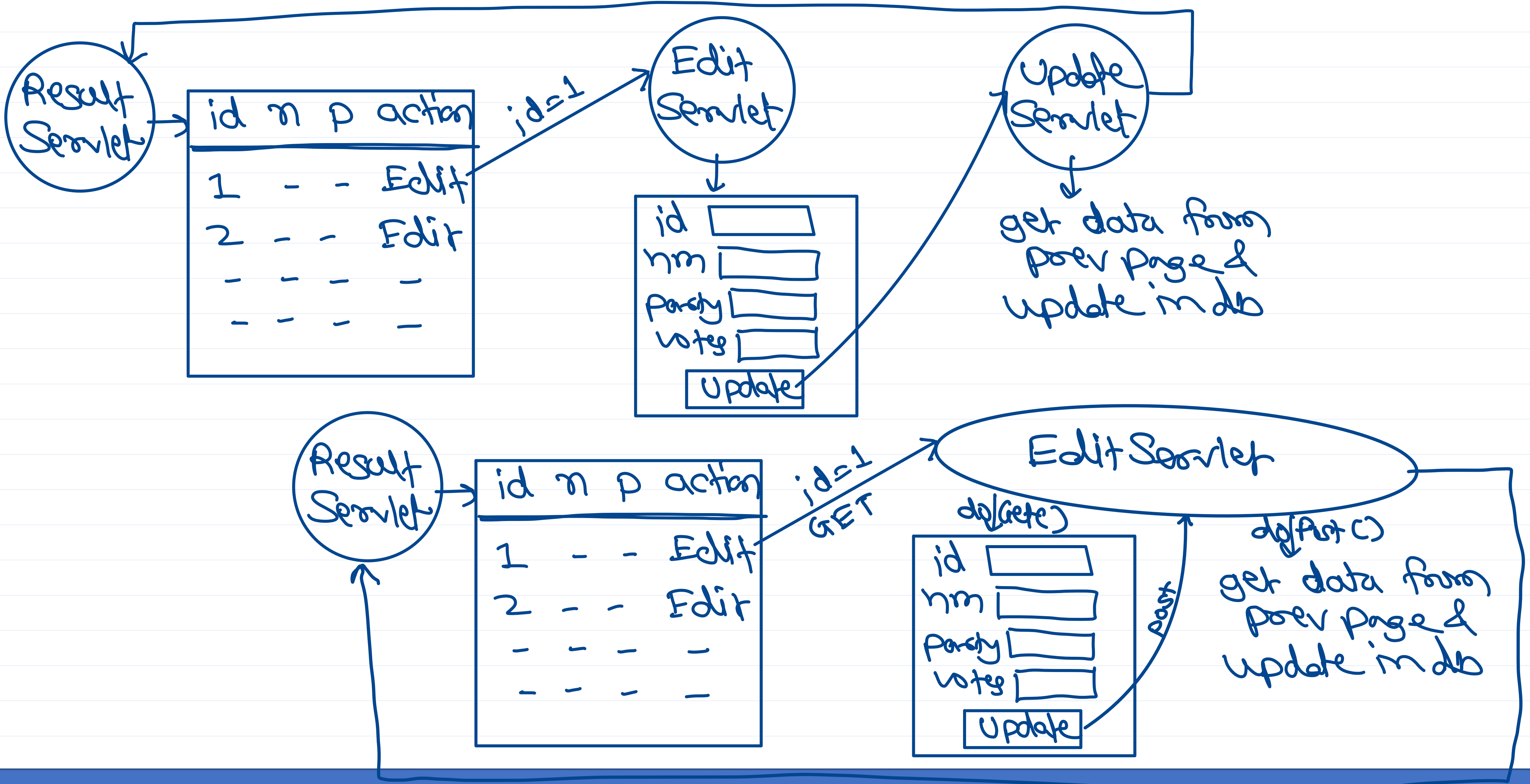
It is called as query string.

http://lh:8080/app/page?key1=val1&key2=val2

This data is accessible in that page as req. parameters.

```
String val1=req.getParameter("key1");
String val2=req.getParameter("key2");
```

Edit record


Result Servlet
id n p action
1 - - Edit
2 - - Edit
id=1
Edit Servlet
id
nm
party
votes
Update
Update Servlet
get data from prev page & update in db
Result Servlet
id n p action
1 - - Edit
2 - - Edit
id=1
GET
Edit Servlet
doGet()
id
nm
party
votes
Update
post
doPost()
get data from prev page & update in db

Req parameters vs Req attributes

① received from client.

② usually data from form controls or query string.

③ always string.

④ to access:

```
String v = req.getParameter("_");
```

or

```
String[] v =
    req.getParameterValues("_");
```

① sent from servlet1 to servlet2

② any data added by servlet1.

③ any java Object

④ servlet 1:

```
req.setAttribute("k", val);
```

servlet2:

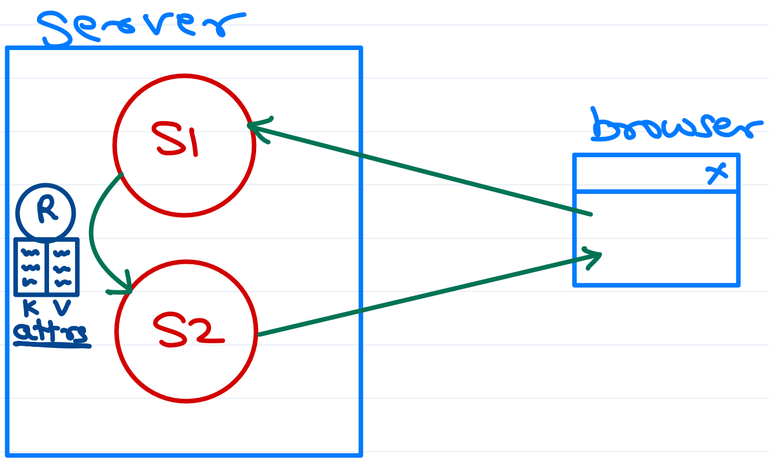
```
val = req.getAttribute("k");
```

⑤ Used only while forwarding/including request.

* both are destroyed when req processing is Completed i.e. response is generated

Attributes - Scopes

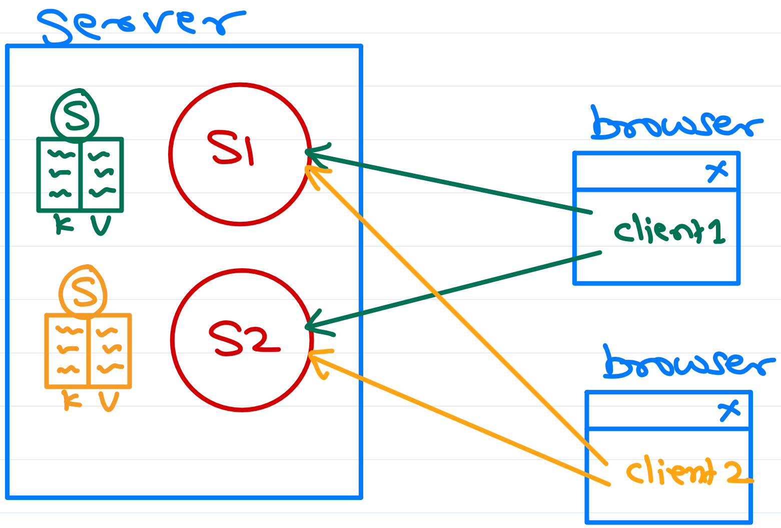
Request attributes



Req attributes will live only for current request - being forwarded or included.

"req" obj is arg to doGet(), doPost(),... methods.

Session Attributes



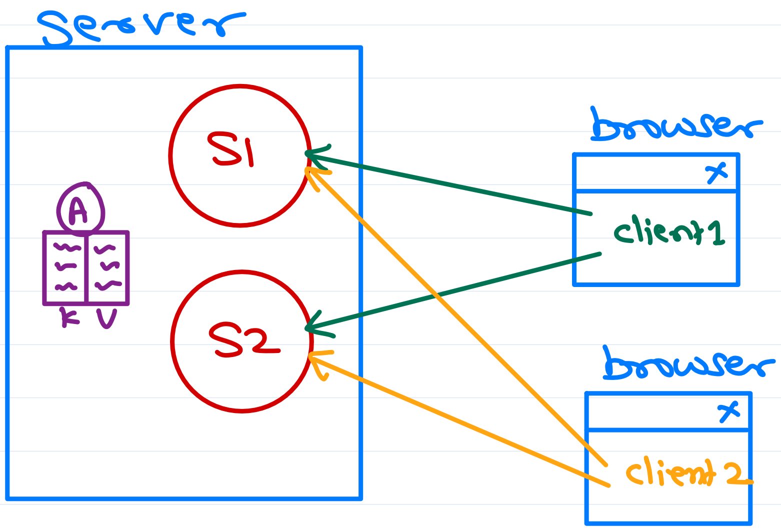
Session attributes are accessible across all requests for current user.

Separate session for each user.

To get current user session:

```
HttpSession session= req.getSession();
```

Servlet Context attributes



ServletContext - application object - one for whole appln.

ServletContext attributes are accessible across all requests for all users.

To get appln/ServletContext:

```
ServletContext appln = req.getServletContext();
```

JSP

JSP is outdated.

Servlets = business logic* + presentation logic

JSPs = presentation logic* + business logic

JSP is translated into Servlet while execution.

① JSP syntax (other than markup)

ⓐ <%@ .... %> → directives
↳ page, include, taglib

ⓑ <%! ... %> → declaration
↳ jspInit(), jspDestroy(), fields, ...

ⓒ <% .... %> → scriptlet
↳ java statements - req. handling

ⓓ <%= .... %> → expression
↳ java expression - req. handling
↳ output added to resp directly.

ⓔ <%-- ... --%> → server side comment

② JSP implicit objects

variables available in req handling stage.
internally - local vars of _jspService() method.

ⓐ request : HttpServletRequest
ⓑ response : HttpServletResponse
ⓒ config : ServletConfig
ⓓ session : HttpSession
ⓔ application : ServletContext
ⓕ page : Object (this pointer)
ⓖ pageContext : PageContext
ⓗ out : JspWriter
ⓘ exception : Throwables
avail only in err pages.

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>