

## Anotações dos Exercícios Práticos

Imports feitos implicitamente pela JSP:

- java.lang.\*
- javax.servlet.\*
- javax.servlet.jsp.\*
- javax.servlet.http.\*

Declarar o useBean com class="" e type="" iguais não gera erro.

Ao buscar um valor do header a partir do objeto implícito de EL, a chave deve estar em aspas caso tenha caracteres especiais como o (-) por exemplo: `${header[user-agent]}` **deve ser** `${header["user-agent"]}`

**Include** via **diretiva** `<%@ include file=""%>` é estático (insere o todo o código do arquivo no tempo de tradução) e via **standard action** `<jsp:include page=""/>` é dinâmico (insere a response do arquivo no tempo de execução). Já a JSTL `<c:import url="" />` é dinâmico e mais poderoso e flexível que a standard action podendo obter o arquivo mesmo estando fora do container.

Ao usar a EL `${}` com um atributo, ele irá varrer todos os contextos até achar um, se não encontrar nenhum, nada será exibido

Pode haver mais de um `<error-page>` no web.xml

Os filtros são encadeados do mais **genérico** para o mais **específico**

O arquivo TLD serve pra duas coisas principais: Custom Tags e EL Functions

A declaração da URI com o Location da tag antes de JSP2.0 era feito dentro de `<jsp-config>` `<taglib>` com `<taglib-uri>` e `<taglib-location>`

O body-content de uma SimpleTag pode ser **empty, scriptless, tagdependent e JSP**. Esse ultimo não funciona para **SimpleTag**, somente para **ClassicTag**

SimpleTag classificada como empty não pode ter um Enter e nem um Espaço em branco entre as tags.

SimpleTag: Classe Java Handler, arquivo .TLD dentro de /WEB-INF/ para fazer o link

TagFile: Arquivo .TAG dentro de /WEB-INF/tags

As Servlets são batidas do mais **específico** para o mais **genérico** seguindo a seguinte ordem (URL Exata, Diretório, Extensão).

A diretiva taglib é removida dos documentos XML, e declarada dentro do `xmlns:c=""`

Pode haver mais de uma `<security-constraint>` no DD.

Se deseja ter autenticação na aplicação, a tag `<login-config>` deverá existir.

Mesmo com `<role-name>*</role-name>` o usuário deve estar autenticado.

## Anotações do Mock do Final do Livro

- É criado uma variável local com o nome do bean informado no id="" do jsp:useBean no método \_jspService()
- Não existe uma diretiva @variable para TagFile
- O evento de HttpSessionBindingEvent não acessa diretamente o getAttribute(), deve ser recuperado a session primeiramente com getSession()
- O elemento <jsp:root> em um JSP Document é opcional, mas se declarado, deve existir um namespace para jsp:
- Se dentro de um comentário html (<!-- -->) houver uma expressão JSP, a mesma será avaliada
- O método padrão para um form HTML é o GET
- O algoritmo das regras do <welcome-file-list> é especificado e deve ser seguido pelo container.
- Após uma session ser invalidada, o único método que pode ser chamado sem haver um exceção é o getServletContext()
- Deixar alguns métodos como doGet() sincronized afeta diretamente no desempenho da aplicação Web.
- TagFile e SimpleTag: body-content pode ser [empty|scriptless|tagdependent]
- Classic Tag: body-content pode ser [empty|scriptless|tagdependent|**JSP**]
- Na SimpleTag handler para recuperar a request é necessário fazer um cast do JspContext para PageContext,
- O método resp.encodeRedirectURL(String) apenas codifica a URL, não faz o redirect
- Se o elemento <auth-constraint> for omitido, todos terão acesso aos resources
- Os métodos getHeader(), getHeaders() e getHeaderNames() pertencem a HttpServletRequest
- Não pode haver mais de um elemento <error-code> ou <exception-type> declarado em <error-page>
- Os atributos de contexto são os que possuem maior vida
- Em qualquer escopo, os atributos podem ser recuperados usando o método getAttribute()
- Role name são case sensitive
- As classes Servlet devem ficar dentro de /WEB-INF/classes ou se estiver em um JAR dentro de /WEB-INF/lib
- Multiplas requests feitas do mesmo browser pode acessar o mesmo session objeto
- Multiplas janelas do browser irão compartilhar a mesma sessão
- O diretório /META-INF é obrigatório quando o arquivo WAR é criado e o conteúdo desse diretório pode ser acessado intertamente pela classe ServletContext
- O método chamado pelo FilterChain para continuar o fluxo é o doFilter(req, res)
- A URI declarada na diretiva @taglib deve ser completa iniciando com /WEB-INF/
- O RequestDispatcher obtido pelo getServletContext() deve iniciar sempre com barra "/"
- O método res.encodeRedirectURL(String) prepara a URL para ser passada ao método res.sendRedirect()
- Uma classe para suportar EL Function deve ter um construtor sem argumentos e o método deverá ser public static
- O prefixo para a JSP Standard Action <jsp:> não precisa ser declarado em uma diretiva taglib, já as JSP Custom Action são necessárias
- Os conceitos comuns para todos os mecanismos de autenticação são: passwords e secure web resources
- A cookie simples é criada para manter o ID unico da session
- Transfer Object: pode criar dados desatualizados, reduz a duplicidade de código, aumenta a complexidade do código
- Business Delegate: usado para minimizar o acoplamento entre clientes e serviços de negócio, e para esconder a complexidade da implementação
- MVC: Apresentação em diversas views