Objetivo

Preparar profissionais da linguagem Java para obter sucesso no exame **1Z0-858 – Java Enterprise Edition 5 Web Component Developer Certified Professional**, através de aulas práticas, simulados e revisões dos principais pontos da prova. Curso totalmente prático e com foco no objetivo da prova através da realização de simulados com explicações exaustivas em cada uma das perguntas reunidas para cobrir todo o conteúdo e obter seu sucesso na Certificação Oracle. Obter o reconhecimento através da Certificação Oracle que estão entre as mais procuradas por sua credibilidade no mercado de Tecnologia da Informação.

Duração
40 horas/aula

Diferenciais X25

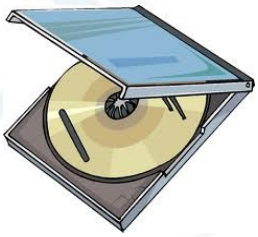
- Instrutores altamente qualificados
- Livros como Material Didático
- Coffee-break
- Computadores de última geração
- Salas com projetores multimídia
- Somente 01 aluno por computador
- Certificado diferenciado pelo aproveitamento do aluno
- Parcerias internacionais
- Treinamentos in-company
- Treinamentos revisados periodicamente

Pré-requisitos

Para obter um bom aproveitamento neste curso os alunos devem ter um conhecimento sólido da linguagem de programação JAVA e praticas da Web. Leitura técnica no idioma Inglês.

Público Alvo

Profissionais que possuam o conhecimento sólido da linguagem Java e desejam confirmá-lo através da certificação oficial da ORACLE. Isso inclui, mas não limita a: Programadores, Desenvolvedores, Analistas de Sistemas e Estudantes.



Materiais

Será distribuído material eletrônico (em língua Inglesa) contendo artigos, exemplos, dicas e questões de simulados.

Conteúdo Programático

Modelo de Tecnologia com servlet

- Descrever a finalidade de cada um dos métodos HTTP (como GET, POST, HEAD, e assim por diante) e suas características técnicas no protocolo HTTP, listas de *triggers* que um cliente poderia causar (geralmente através de um navegador da Web) pelo uso do método e identificar o método *HttpServlet* que corresponde ao método HTTP.
- Utilizar a interface *HttpServletRequest* para recuperar parâmetros de formulários HTML a partir de uma requisição HTTP, obter as informações do *header* e recuperar *cookies* da requisição.
- Utilizar a interface *HttpServletResponse* para definir um *header* de um *response* HTTP, definir o tipo de conteúdo da resposta, adquirir um fluxo de texto ou um fluxo binário para a resposta, redirecionar uma solicitação HTTP para outra URL e adicionar cookies a resposta.
- Descrever o propósito e a sequência de eventos do ciclo de vida do servlet.

Estrutura e Implementação de Aplicações Web

- Estruturar arquivos e diretórios de uma aplicação Web que pode conter: conteúdo estático, páginas JSP, servlet, *deployment descriptor*, bibliotecas TAG, arquivos JAR e de classe Java.
- Proteger arquivos de recursos de acesso HTTP.
- Propósito e a semântica do *deployment descriptor*.
- Construir a estrutura correta do *deployment descriptor*.
- Explicar a finalidade de um arquivo WAR e como seu conteúdo pode ser construído.

Modelo de Contêiner Web

- Acessar os parâmetros de inicialização do *ServletContext* e criar os elementos do *deployment descriptor* para declarar parâmetros de inicialização.
- Para os escopos fundamentais do servlet (*request*, *session* e *context*): adicionar, recuperar e remover atributos, identificar o âmbito adequado para um atributo e identificar problemas *multi-threading* associados a cada âmbito .
- Descrever o modelo do contêiner Web no processamento de uma requisição, criar e configurar um filtro, criar um pedido de resposta e como aplicar um filtro ou uma classe *Wrapper*.
- Descrever o modelo de eventos do ciclo de solicitações, sessões e aplicações Web, criar e configurar classes de escuta para cada ciclo de vida do escopo, criar e configurar classes de escuta escopo de atributos e identificar o atributo *listener* correto.
- Descrever o mecanismo de *RequestDispatcher*, criar um pedido, transmitir ou incluir o recurso de destino e identificar e descrever os adicionais de escopo de atributos fornecidos pelo contêiner para o recurso de destino.

Gerenciamento de Sessões

- Armazenar e recuperar objetos em uma sessão através de um *servlet*.
- Descrever as API utilizadas para acessar o objeto *session*, explicar quando este objeto foi criado e descrever os mecanismos utilizados para destruí-lo.
- Utilizar *listeners* de sessão para responder a um evento quando um objeto é adicionado a uma sessão ou quando um objeto de sessão migra para outra VM.
- Descrever qual mecanismo de gerenciamento de sessão do contêiner Web poderia ser empregado, como os *cookies* podem ser usados para gerenciar as sessões, como a regravação da URL pode ser usada para gerenciar sessões e escrever *servlet* para realizar a regravação de URL.

Web Application Security

- Comparar os seguintes mecanismos de segurança: autenticação, autorização, integridade dos dados e confidencialidade.
- Declarar no *deployment descriptor*: restrição de segurança, recurso Web, garantia de transporte, configuração de login e função de segurança.
- Comparar e contrastar os tipos de autenticação (BASIC, DIGEST, FORM, e CLIENT-CERT), descrever como funciona e selecionar um deles apropriadamente.

Modelo de Tecnologia JavaServer Pages (JSP)

- Identificar, descrever e escrever o código JSP para os seguintes elementos: modelo de texto, elementos de *script* (comentários, diretivas, declarações, *scriptlets* e expressões), *standard* e *Custom Actions* e elementos de expressão de linguagem.
- Escrever código JSP que usa as diretivas: *page* (com os atributos *import*, *session*, *contentType* e *isELIgnored*), *include*, e *taglib*.
- Escrever um documento JSP (documento baseado em XML) que usa a sintaxe correta.
- Descrever o propósito e a sequência dos eventos do ciclo de vida da página JSP, tradução da página JSP, compilação das páginas, carga da classe, criar a instância, chamar o método **jspInit**, chamar o método **_jspService** e chamar o método **jspDestroy**.
- Escrever código JSP usando os objetos implícitos apropriadamente: *request*, *response*, *out*, *session*, *config*, *application*, *page*, *exception* e *pageContext*.
- Configurar o *deployment descriptor* para declarar uma ou mais bibliotecas de tags, desativar a avaliação da linguagem e a linguagem de *script*.
- Incluir um segmento JSP em outra página, utilizar o mecanismo de inclusão mais apropriado (a diretiva ***include*** ou a ação padrão ***jsp:include***).

Construção de páginas JSP utilizando Expression Language (EL)

- Acessar as seguintes variáveis implícitas: *pageScope*, *requestScope*, *sessionScope*, *applicationScope*, *param*, *paramValues*, *header*, *headerValues*, *cookie*, *initParam* e *pageContext*.
- Utilizar os seguintes operadores de acesso: a propriedade (operador .) e a coleção (operador []).

Construção de Páginas JSP com *Custom Actions*

- Utilizar as seguintes *Custom Actions*: *jsp:useBean* (com atributos: 'id', 'escope', 'type' e 'class'), *jsp:getProperty*, *jsp:setProperty* (com todas as combinações de atributos), *jsp: attribute*, *jsp:include*, *jsp:forward* e *jsp:param*.

Construção de Páginas JSP com Tag Libraries

- Criar uma diretiva 'taglib' para uma página JSP para uma biblioteca de tags personalizadas ou uma biblioteca de *Tag Files*.
- Criar a estrutura de uma *Custom Tag* em uma página JSP.
- Utilizar adequadamente a biblioteca ”core” da *JSP Standard Tag Library* (JSTL v1.1).

Construir uma Biblioteca de Tag personalizada

- Descrever a semântica de uma *custom tag* "classic" quando cada método de evento (*doStartTag*, *doAfterBody* e *doEndTag*) é executado, explicar o significado dos valores de retorno de cada método e escrever uma classe *tag handler*.
- Usando a *API PageContext*, escrever código para acessar as variáveis JSP implícitas e atributos de acesso da aplicação Web.
- Dado um cenário, escrever código para acessar uma *tag parent* e *ancestor*.
- Descrever a semântica de uma *custom tag* "Simple" quando o método de evento (*doTag*) é executado, escrever uma classe *tag handler* e explicar as restrições ao conteúdo JSP dentro da *tag*.
- Descrever a semântica do modelo de arquivo tag, descrever a estrutura da aplicação Web para arquivos de tag, escrever um arquivo de tag e explicar as restrições ao conteúdo JSP no corpo da tag.

Padrões de Projeto Java EE

- Para a lista dos Padrões de Projeto aplicados na prova: *Intercepting Filter*, *Model-View-Controller*, *Front Controller*, *Service Locator*, *Business Delegate* e *Transfer Object*.
  - Selecionar um Padrão de Projeto para resolver determinado tipo de problema.
  - Descrever potenciais benefícios que se obtêm a partir do uso do padrão.