Java Server Pages

[01]

CESPE - 2010 - TRT 10

Nas páginas JSP, combinam-se modelos estáticos, incluindo fragmentos de HTML ou XML, com o código para gerar conteúdo dinâmico e compilar páginas JSP dinamicamente em *servlets*, quando solicitado.

CESPE - 2011 - MEC

A JSP é uma extensão da tecnologia dos servlets que permite simplificar o processo de criação de páginas, separando a apresentação do conteúdo.

[02] FCC - 2012 - TRE-CE

Embora as servlets sejam muito boas no que fazem, tornou-se difícil responder ao cliente com conteúdo no formato HTML.

PORQUE

Geralmente quem trabalha com o conteúdo HTML é o web designer que normalmente não é programador Java experiente. Ao misturar HTML dentro de uma servlet, torna-se muito difícil separar as funções de web designer e desenvolvedor Java. Além disso, é difícil fazer alterações no conteúdo HTML, pois para cada mudança, uma recompilação da servlet tem que acontecer. Para contornar as limitações da tecnologia Java Servlet a Sun Microsystems criou a tecnologia JavaServer Pages (JSP).

Acerca dessas asserções, é correto afirmar:

a) Tanto a primeira quanto a segunda asserções são proposições falsas.

b) A primeira asserção é uma proposição verdadeira e a segunda uma proposição falsa.

c) A primeira asserção é uma proposição falsa e a segunda uma proposição verdadeira.

d) As duas asserções são proposições verdadeiras, mas a segunda não é a justificativa correta da primeira.

e) As duas asserções são proposições verdadeiras e a segunda é a justificativa correta da primeira.

[03] CESPE - 2004 - SERPRO

Cada JSP passa por duas fases distintas. Na primeira, denominada translation time, o tradutor transforma um arquivo em um servlet. A segunda fase ocorre quando o servlet é executado para a geração da página. A manipulação dos comentários ocorre na primeira fase, o tradutor omite qualquer comentário fazendo que o servlet não o tenha que manipular.

[04] CESPE - 2006 - CENSIPAM

O seguinte cenário descreve os passos que são executados toda vez que uma página JSP é solicitada: o navegador solicita a página JSP ao servidor; o código de um servlet é gerado e compilado; nesse servlet, o HTML da página encontra-se convertido em enunciados **println**; o servlet é instanciado e são invocados os métodos **init** e **service**; o servlet recebe dados sobre a solicitação via uma instância de **ServletRequest** e envia dados via uma instância de **ServletResponse**; a página HTML produzida pelo servlet é enviada para o navegador; a página é apresentada pelo navegador.

[05] FGV - 2012 - SENADO

Assinale a alternativa que apresenta instruções corretas em sintaxe válida JSP para inicializar e imprimir o valor da variável *num*.

a) *<%int num = 2;%>*
<%=num; %>

b) *<%int num = 2*
num; %>

c) *<%int num = 2 %>*
<%=num %>

d) *<%int num = 2; %>*
<%=num %>

e) *<%int num = 2 %>*
<%=num; %>

[06]

CESPE - 2010 - MPU

O contêiner, que executa JSP, transforma o programa JSP em Servlet, assim, a expressão "<%= Math.Random()%>" se torna argumento para out.println().

CESPE - 2007 - CBM-DF

Uma página JSP pode conter métodos Java definidos entre os grupos de caracteres <%! e %>. Esses métodos não podem ser invocados a partir de expressões Java definidas entre <%= e %>. O resultado da execução de uma expressão é convertido em um int e incluído na página apresentada.

[07]

CESPE - 2011 - MEC

A JSP possui quatro componentes: chaves diretivas, ações, elementos de script e bibliotecas de tags. As ações são mensagens para o contêiner de JSP e os elementos de script permitem aos navegadores inserir códigos Java que interajam com os componentes JSP.

CESPE - 2013 - CNJ

Como forma de incluir dinamismo em páginas JSP, é possível incluir blocos de código Java conhecidos como *scriptlets*.

[08] FUMARC - 2011 - BDMG

Analise as seguintes afirmativas:

I. Uma página JSP é um documento texto que contém dados estáticos em formato HTML e XML, por exemplo, e elementos JSP que constroem o conteúdo dinâmico.

II. Os elementos JSP em uma página JSP podem ser expressos nas sintaxes padrão e XML, embora em um dado arquivo, pode-se usar somente uma das sintaxes.

III. Uma página JSP em sintaxe XML é um documento XML que pode ser manipulado pelas ferramentas e APIs para documentos XML. Além disso, pode ter um **jsp:root** como elemento raiz.

Marque a alternativa **CORRETA:**

a) apenas a I e II são verdadeiras. **b)** apenas a I e III são verdadeiras.

c) apenas a II e III são verdadeiras. **d)** todas são verdadeiras.

[09] FCC - 2012 - MPE-PE

Em uma página JSP, para importar uma classe de um pacote e para fazer referência a uma biblioteca (como, por exemplo, JSTL) podem ser utilizadas, respectivamente, as diretivas:

a) <%@page import="pacote.Classe"%> e <%@taglib uri="caminho/biblioteca" prefix="prefixo"%>

b) <%@include import= "pacote.Classe"%> e <%@taglib uri="caminho/biblioteca"%>

c) <%import= "pacote.Classe"%> e <%taglib uri="caminho/biblioteca"%>

d) <%@page include= "pacote.Classe"%> e <%@library uri="caminho/biblioteca"%>

e) <%@import class= "pacote.Classe"%> e <%@taglib urli="caminho/biblioteca"%>

[10] FCC - 2012 - TRE-CE

<%@ page atributo1="valor1" atributo2="valor2"... %> é a sintaxe típica da diretiva Page, em JSP. Um de seus atributos, se definido para true, indica o processamento normal do servlet quando múltiplas requisições podem ser acessadas simultaneamente na mesma instância de servlet. Trata-se do atributo

a) extends. b) import.

c) isThreadSafe. d) session.

e) autoFlush.

[11] ESAF - 2009 - ANA

O mecanismo de inclusão, que permite o conteúdo dinâmico ser incluído em uma JSP em tempo de solicitação, é denominado:

a) Ação <jsp:plugin>.

b) Ação <jsp:include>.

c) Diretiva include.

d) Diretiva Page.

e) Diretiva taglib.

[12] FCC - 2013 - ALE-RN

Em uma aplicação web desenvolvida utilizando a plataforma Java EE 6, há a seguinte classe Java:

```
package dados;
public class Cliente {
	private String nome;
	public Cliente() { }
	public String getNome() {
		return nome;
	}
	public void setNome(String nome) {
		this.nome = nome;
	}
}
```

Em uma página JSP da mesma aplicação, para instanciar um objeto desta classe pode-se utilizar a tag

a) <jsp:setBean name="cliente" class="dados.Cliente"/>

b) <jsp:setBean id="cliente" class="dados.Cliente"/>

c) <jsp:useBean name="cliente" class="dados.Cliente"/>

d) <jsp:useBean id="cliente" class="dados.Cliente"/>

e) <jsp:newInstance id="cliente" class= "dados.Cliente"/>

[13] CESPE - 2008 - MPE-RR

Cada vez que um novo pedido HTTP for enviado por um *browser* a essa página, será criada uma nova instância da classe sessions.DummyCart.

```
<html>
   <jsp:useBean id="cart" scope="session" class="sessions.DummyCart" />
   <jsp:setProperty name="cart" property="*" />
   <%
      cart.processRequest(request);
   %>
   <FONT size=5 color="#CC0000">
      <br> You have the following items in your cart:
      <ol>
         <%
            String[] items = cart.getItems();
            for (int i=0; i<items.length; i++) {
            %>
              <li><% out.print(util.HTMLFilter.filter(items[i])); %>
            <%
            }
         %>
      </ol>
   </FONT>
   <hr>
   <%@ include file ="carts.html" %>
</html>
```

[14]

CESPE - 2011 - PREVIC

O *container* JSP provê uma lista de objetos instanciados, chamados de objetos implícitos. É através do objeto aplicação *(application object)* que são rastreadas as informações de um cliente específico entre múltiplas requisições.

CESPE - 2011 - CBM-DF

O uso de Javabeans, o controle de transferência entre as páginas e o suporte independente de *applets* Java pelos *browsers* são possibilidades proporcionadas pela *action tag* da JSP.

[15] FCC - 2009 - TRT-MG

NÃO possui uma habilidade de armazenar e recuperar valores de atributos arbitrários o objeto implícito de JSP

a) *session*

b) *request*

c) *exception*

d) *application*

e) *pageContext*

Expression Language

[16]

CESPE - 2010 - TCU

O uso de expressões da *unified expression language* deve ser evitado dentro do código de classes Java, mas tais tipos de expressões são adequados e devem ser usados em páginas JSF, entre outras razões, por possibilitarem maior legibilidade ao código e constituírem alternativa mais simples ao uso de *tags* como <jsp:getProperty>.

CESPE - 2010 - MPU

Para que métodos estáticos de classes Java sejam executados a partir das funções da linguagem de expressão em JSP, é necessário que o nome da função coincida com o nome do método da classe Java.

[17]

CESPE - 2012 - TJ-AL [adaptada]

Em < h:inputText id="name" value= "#{customer.name}" /> o atributo value da tag h:inputText faz referência a uma expressão de avaliação tardia que aponta para a propriedade name do bean customer.

CESPE - 2012 - TJ-AL [adaptada]

As expressões que são avaliadas imediatamente pelo compilador podem atuar como expressões rvalue e lvalue.

JSP Standard Tag Library e Custom Tags

[18] FCC – 2010 – TRT-8ª

A biblioteca de tags padrão do Java Server Pages (JSTL) é uma coleção de tags padronizadas para tarefas comuns a muitas aplicações JSP. Estas tags estão divididas em 5 áreas funcionais:

a) Web, XML, Banco de Dados, Internacionalização (I18n) e Funções.

b) Core, XML, Banco de Dados, Internacionalização (I18n) e Funções.

c) Core, Estatística, Banco de Dados, Internacionalização (I18n) e Funções.

d) Web, Matemática, Banco de Dados, Internacionalização (I18n) e Funções.

e) Core, Web, Matemática, Internacionalização (I18n) e Funções.

[19] CESPE - 2009 - CEHAP-PB

Assinale a opção incorreta acerca de JSTL.

a) O JSTL não possui suporte nativo a SQL. Para tanto, é utilizada em conjunto com a biblioteca nativa SQL do J2EE.

b) Uma página JSTL pode ser definida como uma página JSP contendo um conjunto de tags JSTLs.

c) Em uma página JSTL, cada tag realiza determinado tipo de processamento.

d) O JSTL permite ao programador escrever páginas JSPs sem necessariamente utilizar códigos Java, facilitando a integração entre web designers e programadores.

[20] CESPE - 2008 - MPE-RR

Na linha 13 do trecho de código mostrado, é utilizada uma *tag* de uma biblioteca de *tags* padronizada JSLT.

```jsp
<%@ page contentType="text/html; charset=UTF-8" %>
<%@ taglib prefix="s" uri="/struts-tags" %>
<html>
<head>
<title>Sign On</title>
</head>

<body>
      <s:form action="Login">
            <s:textfield key="username" />
            <s:password key="password" />
            <s:submit />
      </s:form>
</body>
</html>
```

[21] CESPE – 2010 - MPU

As *tags* personalizadas são produzidas em arquivos TLD *(tag library description)*. O código a seguir é um exemplo de *tag* personalizada no JSF 1.0.

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE taglib
PUBLIC "-//sun Microsystems, Inc.//DTD JSP Tag Library 1.2//EN"
"http://java.sun.com/dtd/seb-jsptaglibrary_1_2.dtd">
<taglib>
      <tlib-version>1.0</tlib-version>
      <jsp-version>1.2</jsp-version>
      <short-name>teste</short-name>
      <uri>http://exemplojsf.com.br/teste</uri>
      <tag>
            <name>testeProva</name>
            <tag-class>br.com.exemplojsf.TesteProvaTag</tag-class>
            <attribute>
                  <name>separator</name>
            </attribute>
      </tag>
</taglib>
```

[22]

CESPE - 2012 - PERITO CRIMINAL-CE

A JSP permite introduzir tags customizadas à sua biblioteca e, assim, estender facilidades à linguagem. Entre outros benefícios dessa prática estão a eliminação de scriptles em aplicações JSP, reuso e sintaxe similar à do HTML.

CESPE - 2007 - CBM-DF

Uma página JSP pode conter tags que, quando encontradas, causam a execução de código Java. Essas tags possibilitam a inserção de lógica dentro das páginas e podem ser agrupadas em bibliotecas. As bibliotecas sendo usadas podem ser identificadas via tags no arquivo web.xml.

CESPE - 2011 - Correios

Ao se usar tag personalizada **JSP,** é suficiente carregar uma **URL** que indique a localização do arquivo **TLD** para a biblioteca que se deseja acessar.

JavaServer Faces

[23] FCC - 2011 - TRT - 14ª

Em relação a **frameworks Java**, considere:

I. Associa os eventos do lado cliente com os manipuladores dos eventos do lado do servidor.

II. Fornece separação de funções que envolvem a construção de aplicações *Web*.

III. Inclui um conjunto padrão de componentes de interface de usuário que possibilitam validação padronizada.

Os itens I, II e III, referem-se a

a) **EJB**, apenas. b) **JSF**, apenas. c) *Hibernate*, apenas.

d) **EJB** e **JSF**. e) **JSF** e *Hibernate*.

[24]

FCC - 2010 - TRT-PI [adaptada]

JavaServer Faces é um *framework* MVC utilizado no desenvolvimento de aplicações para a internet de forma visual, que utiliza o recurso de arrastar e soltar os componentes na tela para definir suas propriedades.

CESPE - 2010 - TRE-BA

Uma página JSF pode ser composta de conteúdos estáticos, como *tags* HTML e códigos dinâmicos.

[25]

CESPE - 2009 - SECONT-ES

O JSF é um framework web embasado em interface gráfica, capaz de renderizar componentes e manipular eventos em aplicações web no padrão Java EE, no qual os componentes JSF são orientados a eventos. O JSF fornece, ainda, mecanismos para conversão, validação, execução de lógica de negócios e controle de navegação.

FUMARC - 2014 - PBH [adaptada]

JSF é um *framework* para desenvolvimento de aplicações Web em Java, baseado no modelo MVC, para construção de interfaces com os usuários por meio de componentes visuais.

[26] FCC - 2012 - TJ-PE

No JSF, o componente *Controller do MVC* é composto por uma classe servlet, por arquivos de configuração e por um conjunto de manipuladores de ações e observadores de eventos. Essa *servlet* é chamada de

a) ControllerServlet. *b) Facelet.*

c) HttpServlet. *d) FacesConfig.*

e) FacesServlet.

[27] FCC - 2013 - TRT-PR

Uma aplicação utilizando o *framework* JSF e a IDE NetBeans gera automaticamente dois componentes essenciais assim descritos:

I. É responsável por receber requisições dos componentes *View* do MVC, redirecioná-las para os *beans* gerenciados (*managed beans*) do componente *Model* do MVC e responder a essas requisições.

II. É o arquivo principal de configuração de uma aplicação *web* que utiliza o *framework* JSF. É responsável por descrever os elementos e sub-elementos que compõem o projeto, tais como as regras de navegação, *beans* gerenciados, configurações de localização etc.

As descrições I e II referem-se, respectivamente, aos componentes

a) servlet Controller.java e ao arquivo faces_config.xml

b) FaceletServlet e ao arquivo web_config.xml.

c) FacesServlet e ao arquivo faces-config.xml.

d) servlet Controller e ao arquivo web-config.xml.

e) servlet Facelet e ao arquivo web.xml.

[28]

CESPE - 2013 - STF

A configuração do controlador do JSF é realizada no *servlet* contido no arquivo *web.xml*. Essa servlet é responsável por receber as requisições e delegá-las ao núcleo do JSF.

CESPE - 2014 - TJ-SE

Antes de uma aplicação *web* desenvolvida nos moldes da JSF executar sua primeira página *web*, uma instância *FacesServlet* é executada, a fim de gerenciar as requisições dessa aplicação.

[29] FCC - 2011 – TER-AP

Considere:
O JSF extrai todos os valores digitados pelo usuário e guarda esse valor nos seus respectivos componentes. Se o valor digitado não coincidir com o componente, um erro vai ser adicionado na classe FacesContext *e será mostrado na fase* Render Response Phase.
No ciclo de vida do JSF trata-se de um evento típico da fase

a) Process Validations Phase.

b) Restore View Phase.

c) Apply Request Values Phase.

d) Update Model Values Phase.

e) Invoke Application Phase.

[30] FCC - 2012 - TRE-CE

No ciclo de vida do *Java Server Faces* trata-se da fase na qual o componente deve primeiro ser criado ou recuperado a partir do *FacesContext*, seguido por seus valores, que são geralmente recuperados dos parâmetros de *request* e, eventualmente, dos cabeçalhos ou *cookies* gerados. Trata-se da fase

a) Restore View.

b) Apply Request Values.

c) Process Validation.

d) Update Model Values.

e) Invoke Application.

[31] IADES - 2014 - TRE-PA

O clico de vida do JavaServer Faces (JSF) é composto por seis fases. Acerca desse tema, assinale a alternativa correta.

a) A fase em que se atualizam valores do modelo só é alcançada após todos os componentes JSF serem considerados válidos na fase que processa as validações.

b) Na fase em que se aplicam os valores de requisição, cada componente extrai seu novo valor e armazena em si próprio; porém, caso esse valor não seja do tipo String, uma mensagem de erro é gerada e associada ao componente.

c) Quando é feita uma requisição a uma página JSF, sua implementação inicia a fase que processa as validações dos componentes.

d) Durante a fase de atualizar os valores do modelo, a implementação do JSF manipula eventos da aplicação, como ir para a próxima página por meio de um link.

e) Caso sejam encontrados erros nas fases em que as validações são processadas ou em que os valores do modelo são atualizados, tais erros são exibidos na página após a fase de renderizar a resposta, independentemente de a página conter um ou mais componentes <message/> ou <messages/>.

[32] CESPE - 2012 - ANAC

Com relação ao desenvolvimento Java, julgue os itens a seguir.

Na fase de submissão de valores via request do JSF, caso a conversão de um valor falhe, uma mensagem de erro associado com o componente é gerada, devolvida para FacesContext e exibida para o usuário, parando-se imediatamente o processamento a partir desse ponto.

[33] FCC - 2010 - MPE-SE

No Java Server Pages (JSP), BackBean

a) não seguem a convenção JavaBean, portanto não possuem getters e setters.

b) são métodos que acionam os Action Event Handlers, sem procurar nenhuma regra de navegação para voltar a mesma página.

c) são classes simples, não herdam de ninguém nem são obrigados a implementar nenhuma interface.

d) são componentes associados à página, onde são criadas as views de primeiro acesso.

e) são processos que executam os validadores existentes na view, para aplicação das regras a todos os valores digitados.

[34] FCC - 2013 - TRT-SC

Considere as instruções abaixo encontradas em um arquivo de uma aplicação que utiliza JSF

```
<managed-bean>
<managed-bean-name>func</managed-bean-name>
<managed-bean-class>bean.Funcionario</managed-bean-class>
<managed-bean-scope>session</managed-bean-scope>
</managed-bean>
```

Essas instruções indicam a existência de um *bean* gerenciado (classe Funcionario.java) no pacote *bean* que poderá ser referenciado nas páginas JSP por meio da palavra func . O arquivo correto no qual essas instruções são colocadas é o

a) context.xml. b) web-inf.xml. c) web.xml.

d) faces-config.xml. e) config - bean.xml.

[35] CESPE - 2009 - TCU

```
1<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
2<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
3<html>
4<body>
5    <div id="container">
6        <f:view>
7            <h:form>
8                <table>
9                    <tr><td>Email</td>
10                       <td><h:inputText id="email" value="#{flight.email}"/></td></tr>
11                   <tr><td><h:outputText id="departure" value="#{flight.departure}"/></td> </tr>
12                   <tr><td><h:outputText value="#{booking_messages['city']}"/></td>
13                       <td><h:selectOneMenu value="#{flight.city}" required="true" /></td>
14                   </tr>
15               </table>
16               <h:commandButton type="submit" value="#{booking_messages['reserve']}"
17                   action="#{flight.reserve}"/>
18               <h:messages/>
19           </h:form>
20       </f:view>
21   </div>
22</body>
23</html>
```

Na instrução ***value="#{flight.*}"***, o nome do *backing bean* é ***flight*** e o segundo valor representa o nome do campo definido no *bean.*

[36]

CESPE - 2013 - INPI

Quando registrado em JSF 2 (Java Server Faces), um *managed bean* permanece no escopo de *session.*

CESPE - 2012 - ANAC

A validação de dados de um componente pode ser uma das funções de um backing bean, em uma aplicação JSF.

CESPE - 2011 - BRB

No JavaServer Faces, para que as páginas de uma aplicação acessem as propriedades e operações de uma classe Bean, é necessário realizar um mapeamento da classe, que pode ser feito no arquivo faces-config.xml ou utilizando-se annotations

[37] FUMARC - 2011 - BDMG

Em relação aos conceitos da tecnologia *JavaServer Faces* (JSF), analise as seguintes afirmativas:

I. JSF fornece um conjunto de componentes de interface de usuário – componentes JSF – que ajudam na construção de páginas Web.

II. Os componentes JSF podem ser adicionados a páginas JSP por meio das bibliotecas de *tags* personalizadas *(tag libraries*).

III. Além dos componentes básicos, existem bibliotecas nativas de componentes JSF adaptados para interfaces Swing e AWT, por exemplo.

Marque a alternativa CORRETA:

a) apenas I e II são verdadeiras.

b) apenas I e III são verdadeiras.

c) apenas II e III são verdadeiras.

d) todas são verdadeiras.

[38] CESPE - 2012 - TJ-AC

A inclusão de ***inputText*** em uma página JSF permite validar o tamanho mínimo dos valores digitados por meio da utilização do seguinte código:

```
<h:inputText id="cpf" label="CPF" value="#{UsuarioBean.cpf}">
      <f:validateLongRange
            minimum="#{UsuarioBean.minimum}"
            maximum="#{UsuarioBean.maximum}" />
</h:inputText>
```

[39] CESPE - 2012 - TJ-AC

Acerca de sistemas transacionais, julgue os itens seguintes.
A tecnologia JSF suporta eventos de mudança de valores por meio da seleção de um link.

[40]

CESPE - 2014 - TJ-SE

Em aplicações *web* nos padrões da JSF, é possível utilizar recursos Ajax para criar páginas dinâmicas, como, por exemplo, por meio da *tag* f:ajax, conforme apresentado na sintaxe abaixo.

CESPE - 2010 - TCU

Para suportar a construção de aplicações com Ajax e JSF, recomenda-se aos desenvolvedores de páginas que usem a *tag* <f:ajax>, relacionada ao processamento de pedidos http assíncronos.

[41]

CESPE - 2010 - TCU

No desenvolvimento de conteúdos para apresentação, o uso de *facelets* traz vantagens em relação ao uso de JSP. Uma delas é a maior modularidade, com o uso de *templates* e componentes compostos *(composite)*.

CESPE - 2013 - CPRM

Facelets são utilizadas para desenvolver visões (views) JavaServer Faces (JSF) com linguagem HTML e XHTML, em conformidade com a *transitional document type definition*, sendo, ainda, compatível com a biblioteca de *tag* JSF.

CESPE - 2014 - TJ-SE

É possível utilizar XHTML no desenvolvimento de *facelets* para criar páginas *web* compatíveis com a JSF (*JavaServer Faces*) para apresentação dos dados. Na versão Java EE 7, essa forma de apresentação é mais indicada que a JSP (*JavaServer Pages*), uma vez que esta não suporta todos os novos recursos da versão Java EE 7.

[42] FCC - 2012 - TST

Para criar as páginas XHTML de uma aplicação JSF é possível utilizar um conjunto de bibliotecas de tags JSF. Algumas dessas bibliotecas são HTML, *Core e Facelets.* Considere os fragmentos de códigos abaixo, que utilizam *tags* dessas bibliotecas:

Fragmento de código **I**:

```
<h1>
   <ui:insert name="titulo">
      Título
   </ui:insert>
</h1>
```

Fragmento de código **II**:

```
<h:inputText id="usuário"
   value="#{usuarioBean.usuario.nome}"/>
```

Fragmento de código **III**:

```
<h:selectOneMenu id="lista">
   <f:selectItems
      value="#{optionBean.optionList}">
   </f:selectItem>
</h:selectOneMenu>
```

A correlação correta entre o fragmento de código e a biblioteca de *tags* utilizada é

a) I-*Facelets*, II-HTML e III-Core.

b) I-Core, II-*Facelets*, e III-HTML.

c) I-HTML, II-*Core*, e III-Facelets.

d) I-HTML, II-*Facelets*, e III-HTML.

e) I-HTML, II-HTML, e III-Core.

Java API for JSON Processing (JSON-P)

[43] CESPE - 2014 - ANATEL

Na plataforma JEE (Java Enterprise Edition) versão 6, não é possível encontrar bibliotecas da própria plataforma para o consumo dos serviços REST no formato JSON.

Gabarito

[01]	C-C
[02]	E
[03]	C
[04]	E
[05]	D
[06]	C-E
[07]	E-C
[08]	D
[09]	A
[10]	C
[11]	B

[12]	D
[13]	E
[14]	E-C
[15]	C
[16]	C-E
[17]	C-E
[18]	B
[19]	A
[20]	E
[21]	C
[22]	C-C-E

[23]	B
[24]	C-E
[25]	C-C
[26]	E
[27]	C
[28]	C-C
[29]	C
[30]	B
[31]	A
[32]	E
[33]	C

[34]	D
[35]	C
[36]	E-C-C
[37]	A
[38]	C
[39]	E
[40]	C-C
[41]	C-C-C
[42]	A
[43]	C