\*No imprimir documento, se contesta digitalmente.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombres** | David Albuja Quiñonez | **Calificación** | 28/44  **6.36/10** |
| **Fecha** | 05/03/2020 |

## **Prueba Práctica**

En la prueba siguiente se pretende evaluar la velocidad de aprendizaje y la capacidad de resolver nuevos temas; la persona que toma la prueba deberá explicar:

* Como probar xpath en notepad++ mediante plugin XMLTools
* Como probar xslt en notepad++ mediante plugin XMLTools
* Como diseñar un wsdl en eclipse
* Como diseñar un servicio en jdeveloper

## **Ejercicios de xpath**

Para todos los ejercicios siguientes pegar el código que realiza lo solicitado o la respuesta en caso respectivo.

Dado el siguiente xml responda el cuadro siguiente:

<?xml version="1.0" encoding="UTF-8"?>

<bookstore>

<book category="cooking">

<title lang="en">Everyday Italian</title>

<author>Giada De Laurentiis</author>

<year>2005</year>

<price>30.00</price>

</book>

<book category="children">

<title lang="es">Harry Potter</title>

<author>J K. Rowling</author>

<year>2005</year>

<price>29.99</price>

</book>

<book category="web">

<title lang="es">XQuery Kick Start</title>

<author>James McGovern</author>

<author>Per Bothner</author>

<author>Kurt Cagle</author>

<author>James Linn</author>

<author>Vaidyanathan Nagarajan</author>

<year>2003</year>

<price>49.99</price>

</book>

<book category="web">

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<year>2003</year>

<price>39.95</price>

</book>

</bookstore>

|  |  |  |  |
| --- | --- | --- | --- |
|  | Xpath | | Resultado |
|  | /bookstore/book[2]/year/text() | | 3 |
|  | /bookstore/book[last()-1]/author[1] /text() | | 3 |
|  | /bookstore/book[position()<2]/title/text() | | 3 |
|  | /bookstore/book[title[@lang='es']][2]/title/text() | 3 | |
|  | /bookstore/book[price>45.00]/title/text() | | 3 |
|  | //book[@category="children"]/price/text() | | 3 |
|  | Expresión para obtener el título del libro escrito por “J K. Rowling” | | /bookstore/book[2]/author |
|  | Expresión para obtener todos los autores del libro con categoría web y precio menor a 40$ | | /bookstore/book[@category='web'][price>40]/author |
|  | Expresión para obtener el libro cuyo precio es mayor a 40$ | | /bookstore/book[price>40] |
|  | Expresión para obtener la categoría del libro cuyo autor es “Erik T. Ray” | | /bookstore/book/author[text()='Erik T. Ray'] |

\*Cuadro a llenar por evaluador

|  |  |
| --- | --- |
| Calificación | 4/10 |
| Comentarios |  |

## **Ejercicios de XSLT**

Para todos los ejercicios siguientes pegar el código que realiza lo solicitado o la respuesta en caso respectivo.

Considerar para los ejercicion de esta sección el siguiente xml:

<?xml version="1.0" encoding="UTF-8"?>

<catalog>

<cd>

<title>Empire Burlesque</title>

<artist>Bob Dylan</artist>

<country>USA</country>

<company>Columbia</company>

<price>10.90</price>

<year>1985</year>

</cd>

…

</catalog>

1. Cual hoja de estilos permite tener la siguiente presentación:

<listabreve>

<libro>

<titulo>Empire Burlesque</titulo>

<año>1985</ año >

</libro>

…

</listabreve>

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<listabreve>

<xsl:for-each select="catalog/cd">

<libro>

<titulo>

<xsl:value-of select="title"/>

</titulo>

<año>

<xsl:value-of select="year"/>

</año>

</libro>

</xsl:for-each>

</listabreve>

</xsl:template>

</xsl:stylesheet>

1. Cual hoja de estilos permite tener la presentación del ejercicio anterior, evitando mostrar cd’s cuyos valores excedan los 15$

<xsl:for-each select="catalog/cd[price &lt; 15]">

1. Cual hoja de estilos permite tener la presentación del ejercicio anterior, colocando elemento un hermano después de “año” llamado “cualidad” que tomará el valor de “caro” si el precio supera los 15$ y “barato” si caso contrario.

Falta

\*Cuadro a llenar por evaluador

|  |  |
| --- | --- |
| Calificación | 4/6 |
| Comentarios |  |

## Ejercicios de wsdl

1. Cree un contrato que describa a un servicio “Prueba” que tenga el método sumar, donde ingresan 2 valores flotantes y responde 1 valor flotante.

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://www.example.org/miWsdl/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" name="miWsdl" targetNamespace="http://www.example.org/miWsdl/">

<wsdl:types>

<xsd:schema targetNamespace="http://www.example.org/miWsdl/">

<xsd:element name="sumar">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="param1" type="xsd:float" />

<xsd:element name="param2" type="xsd:float"></xsd:element>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

<xsd:element name="sumarResponse">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="out" type="xsd:float"/>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

</xsd:schema>

</wsdl:types>

<wsdl:message name="sumarRequest">

<wsdl:part element="tns:sumar" name="parameters"/>

</wsdl:message>

<wsdl:message name="sumarResponse">

<wsdl:part element="tns:sumarResponse" name="parameters"/>

</wsdl:message>

<wsdl:portType name="miWsdl">

<wsdl:operation name="sumar">

<wsdl:input message="tns:sumarRequest"/>

<wsdl:output message="tns:sumarResponse"/>

</wsdl:operation>

</wsdl:portType>

<wsdl:binding name="miWsdlSOAP" type="tns:miWsdl">

<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

<wsdl:operation name="sumar">

<soap:operation soapAction="http://www.example.org/miWsdl/sumar"/>

<wsdl:input>

<soap:body use="literal"/>

</wsdl:input>

<wsdl:output>

<soap:body use="literal"/>

</wsdl:output>

</wsdl:operation>

</wsdl:binding>

<wsdl:service name="Prueba">

<wsdl:port binding="tns:miWsdlSOAP" name="miWsdlSOAP">

<soap:address location="http://www.example.org/"/>

</wsdl:port>

</wsdl:service>

</wsdl:definitions>

1. Al ejercicio anterior que debe hacer para que los campos de entrada sean obligatorios

<xsd:element name=*"param1"* type=*"xsd:float"* maxOccurs=*"1"* minOccurs=*"1"*/>

<xsd:element name=*"param2"* type=*"xsd:float"* maxOccurs=*"1"* minOccurs=*"1"*></xsd:element>

1. Cree otro contrato de un servicio “Prueba” que tenga el método validarCuenta que reciba un número de cuenta bancaria, el tipo de cuenta (ahorros o corrientes) y el valor a debitar y la respuesta consiste de un texto que indica si puede o no debitar el valor de dicha cuenta. Todos los campos son obligatorios

<?xml version=*"1.0"* encoding=*"UTF-8"* standalone=*"no"*?>

<wsdl:definitions xmlns:soap=*"http://schemas.xmlsoap.org/wsdl/soap/"* xmlns:tns=*"http://www.example.org/Prueba/"* xmlns:wsdl=*"http://schemas.xmlsoap.org/wsdl/"* xmlns:xsd=*"http://www.w3.org/2001/XMLSchema"* name=*"Prueba"* targetNamespace=*"http://www.example.org/Prueba/"*>

<wsdl:types>

<xsd:schema targetNamespace=*"http://www.example.org/Prueba/"*>

<xsd:element name=*"validarCuenta"*>

<xsd:complexType>

<xsd:sequence>

<xsd:element name=*"nroCuenta"* type=*"xsd:string"* maxOccurs=*"1"* minOccurs=*"1"*/>

<xsd:element name=*"tipoCuenta"* type=*"xsd:string"* maxOccurs=*"1"* minOccurs=*"1"*/>

<xsd:element name=*"valorDeb"* type=*"xsd:double"* maxOccurs=*"1"* minOccurs=*"1"*/>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

<xsd:element name=*"validarCuentaResponse"*>

<xsd:complexType>

<xsd:sequence>

<xsd:element name=*"out"* type=*"xsd:string"*/>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

</xsd:schema>

</wsdl:types>

<wsdl:message name=*"validarCuentaRequest"*>

<wsdl:part element=*"tns:validarCuenta"* name=*"parameters"*/>

</wsdl:message>

<wsdl:message name=*"validarCuentaResponse"*>

<wsdl:part element=*"tns:validarCuentaResponse"* name=*"parameters"*/>

</wsdl:message>

<wsdl:portType name=*"Prueba"*>

<wsdl:operation name=*"validarCuenta"*>

<wsdl:input message=*"tns:validarCuentaRequest"*/>

<wsdl:output message=*"tns:validarCuentaResponse"*/>

</wsdl:operation>

</wsdl:portType>

<wsdl:binding name=*"PruebaSOAP"* type=*"tns:Prueba"*>

<soap:binding style=*"document"* transport=*"http://schemas.xmlsoap.org/soap/http"*/>

<wsdl:operation name=*"validarCuenta"*>

<soap:operation soapAction=*"http://www.example.org/Prueba/validarCuenta"*/>

<wsdl:input>

<soap:body use=*"literal"*/>

</wsdl:input>

<wsdl:output>

<soap:body use=*"literal"*/>

</wsdl:output>

</wsdl:operation>

</wsdl:binding>

<wsdl:service name=*"Prueba"*>

<wsdl:port binding=*"tns:PruebaSOAP"* name=*"PruebaSOAP"*>

<soap:address location=*"http://www.example.org/"*/>

</wsdl:port>

</wsdl:service>

</wsdl:definitions>

1. Cree otro contrato de un servicio “Prueba” que tenga los métodos antes descritos sumar y validarCuenta

<?xml version=*"1.0"* encoding=*"UTF-8"* standalone=*"no"*?>

<wsdl:definitions xmlns:soap=*"http://schemas.xmlsoap.org/wsdl/soap/"* xmlns:tns=*"http://www.example.org/Prueba2/"* xmlns:wsdl=*"http://schemas.xmlsoap.org/wsdl/"* xmlns:xsd=*"http://www.w3.org/2001/XMLSchema"* name=*"Prueba2"* targetNamespace=*"http://www.example.org/Prueba2/"*>

<wsdl:types>

<xsd:schema targetNamespace=*"http://www.example.org/Prueba2/"*>

<xsd:element name=*"validarCuenta"*>

<xsd:complexType>

<xsd:sequence>

<xsd:element name=*"nroCuenta"* type=*"xsd:string"* maxOccurs=*"1"* minOccurs=*"1"*/>

<xsd:element name=*"tipoCuenta"* type=*"xsd:string"* maxOccurs=*"1"* minOccurs=*"1"*/>

<xsd:element name=*"valorDeb"* type=*"xsd:double"* maxOccurs=*"1"* minOccurs=*"1"*/>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

<xsd:element name=*"validarCuentaResponse"*>

<xsd:complexType>

<xsd:sequence>

<xsd:element name=*"out"* type=*"xsd:string"*/>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

<xsd:element name=*"sumar"*>

<xsd:complexType>

<xsd:sequence>

<xsd:element name=*"param1"* type=*"xsd:string"* maxOccurs=*"1"* minOccurs=*"1"*></xsd:element>

<xsd:element name=*"param2"* type=*"xsd:string"* maxOccurs=*"1"* minOccurs=*"1"*></xsd:element>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

<xsd:element name=*"sumarResponse"*>

<xsd:complexType>

<xsd:sequence>

<xsd:element name=*"out"* type=*"xsd:float"*></xsd:element>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

</xsd:schema>

</wsdl:types>

<wsdl:message name=*"validarCuentaRequest"*>

<wsdl:part element=*"tns:validarCuenta"* name=*"parameters"*/>

</wsdl:message>

<wsdl:message name=*"validarCuentaResponse"*>

<wsdl:part element=*"tns:validarCuentaResponse"* name=*"parameters"*/>

</wsdl:message>

<wsdl:message name=*"sumarRequest"*>

<wsdl:part name=*"parameters"* element=*"tns:sumar"*></wsdl:part>

</wsdl:message>

<wsdl:message name=*"sumarResponse"*>

<wsdl:part name=*"parameters"* element=*"tns:sumarResponse"*></wsdl:part>

</wsdl:message>

<wsdl:portType name=*"Prueba2"*>

<wsdl:operation name=*"validarCuenta"*>

<wsdl:input message=*"tns:validarCuentaRequest"*/>

<wsdl:output message=*"tns:validarCuentaResponse"*/>

</wsdl:operation>

<wsdl:operation name=*"sumar"*>

<wsdl:input message=*"tns:sumarRequest"*></wsdl:input>

<wsdl:output message=*"tns:sumarResponse"*></wsdl:output>

</wsdl:operation>

</wsdl:portType>

<wsdl:binding name=*"Prueba2SOAP"* type=*"tns:Prueba2"*>

<soap:binding style=*"document"* transport=*"http://schemas.xmlsoap.org/soap/http"*/>

<wsdl:operation name=*"validarCuenta"*>

<soap:operation soapAction=*"http://www.example.org/Prueba2/validarCuenta"*/>

<wsdl:input>

<soap:body use=*"literal"*/>

</wsdl:input>

<wsdl:output>

<soap:body use=*"literal"*/>

</wsdl:output>

</wsdl:operation>

</wsdl:binding>

<wsdl:service name=*"Prueba2"*>

<wsdl:port binding=*"tns:Prueba2SOAP"* name=*"Prueba2SOAP"*>

<soap:address location=*"http://www.example.org/"*/>

</wsdl:port>

</wsdl:service>

</wsdl:definitions>

\*Cuadro a llenar por evaluador

|  |  |
| --- | --- |
| Calificación | 8/8 |
| Comentarios |  |

## **Ejercicios SOA**

Crear un Servicio web soa llamado Prueba con los siguientes métodos:

1. Método sumar; suma 2 valores flotantes y retorna un valor flotante.
2. validarCuenta: dado una cuenta, tipo de cuenta(ahorros o corrientes) y el valor a debitar indica si es posible debitar caso contrario muestra el mensaje de error, para esto considerar los siguientes insumos:

* sp de validación: base: cob\_cuentas sp: pruebaRecaudacion
* credenciales para conectarse a la base de datos: user: operadorSat clave: SAT\*2013
* string de conexión: jdbc:sybase:Tds:172.16.23.22:2030/cob\_cuentas
* conectarse con “generic jdbc” y driver “com.sybase.jdbc4.jdbc.SybDriver”

Procedimiento almacenado en caso de no estar compilado:

use cob\_cuentas

go

if exists (select 1 from sysobjects where name = 'pruebarecaudaciones' and type='P')

begin

drop procedure dbo.pruebarecaudaciones

if exists (select 1 from sysobjects where name = 'pruebarecaudaciones' and type='P')

PRINT '<<< drop procedure pruebarecaudaciones error -->>>'

else

PRINT '<<< drop procedure pruebarecaudaciones OK -->>>'

end

go

create procedure dbo.pruebarecaudaciones (

@e\_cuenta varchar(10),

@e\_tipocta varchar(3),

@e\_debito money

)

as

if @e\_cuenta = "0005247004"

return 0

else

return 25060

go

if exists (select 1 from sysobjects where name = 'pruebarecaudaciones' and type='P')

PRINT '<<< CREATE PROCEDURE pruebarecaudaciones OK -->>>'

else

PRINT '<<< CREATE PROCEDURE pruebarecaudaciones ERROR -->>>'

go

grant execute on pruebarecaudaciones to apl\_osb\_hpdes

go

\*Cuadro a llenar por evaluador

|  |  |
| --- | --- |
| Calificación | 12/20 |
| Comentarios | Conoce y tiene practica en tecnologías xml,xpath,wsdl,xslt  Ha trabajado con servicios web  Deseo de continuar.  Pudo concluir con una explicación de ayuda  Retentiva nivel medio-bajo  Se bloqueó un poco en la explicación  Podrá aprender con una ligera cantidad de explicaciones |