Disciplina: Linguagens e Tecnologias Web (edição 2013/14)
Data: Quarta-feira, 15 de Janeiro de 2014
Hora: 09h00m
Duração: 120 minutos
Salas: B116, B120 e B113
Nota: Com consulta de apontamentos em papel
Época: Normal
Docentes: JVV, AOR e TPF

1) Considerando o HTML representado abaixo responda às seguintes alíneas

a) (1v) Apresente um esboço da visualização no navegador assumindo que os ficheiros de CSS não existem.

```
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="1.css"/>
        <link rel="stylesheet" type="text/css" href="2.css"/>
    </head>
    <body>
        <div><ol><li>A</li><li>B</li><li>C</li><li>D</li></ol></div>
    </body>
</html>
```

b) (2v) Repita a alínea anterior assumindo agora a existência do ficheiro 1.css com o seguinte conteúdo

```
*{
    margin:0px;
    padding:0px;
}
html, body{
    width:100%;
    height:100%;
}
div{
    margin:0 auto;
    width:400px;
    position:relative;
    top:50%;
}
ol{overflow:hidden;}
li{
    display:inline-block;
    width:98px;
    height:25px;
    border:1px solid black;
    text-align:right;
}
li ~ li:not(:last-child){
    position:relative;
    top: 25px;
}
```

c) (2v) Assuma agora que o ficheiro 2.css também existe e tem o conteúdo representado abaixo. Compare a especificidade do único seletor de 2.css com a especificidade do último seletor de 1.css, indicando ambos os valores que obteve e, depois, esboce o resultado da visualização no navegador?

```
li:not(.a1):not(.a2){
    float:right;
    position:static;
}
```

2) Suponha que tem uma página HTML com um único formulário o qual pode conter vários elementos input e textarea os quais podem ser criados/eliminados dinamicamente. Utilizando jQuery implemente os seguintes handlers

   a) (1v) Um para guardar o valor atual do campo numa variável global. Este handler deve correr sempre que se inicia a edição do campo.

   b) (3v) Outro, que deve correr sempre que se termina a edição do campo, cujo objetivo é comparar o valor atual com o valor anterior e, se diferente, deve via ajax (função $.ajax), invocar o url ./update.php com os seguintes parâmetros: field, newvalue, oldvalue assumindo cada um o valor respetivo. Em caso de sucesso nada deve ser assinalado. Em caso de erro deve ser gerada a mensagem "ERRO no campo 'nome do campo'" num elemento span a inserir no topo do formulário

3) (3v) Usando PHP escreva o script param2json.php que recebe os parâmetros field, newvalue, oldvalue e devolve uma mensagem json com o seguinte formato '{"field":$field, "data":{"old":$oldvalue, "new":$newvalue}}'. Se, no pedido, algum destes campos estiver omisso (não é um erro os parâmetros newvalue e oldvalue serem nulos) deve ser gerada a resposta '{"missing":["nome do campo1 em falta",…, "nome do campoN em falta"]}'. Se o parâmetro field estiver definido mas for nulo a resposta deve ser '{"field": ""}'. As mensagens de resposta devem ser do tipo "application/json"

4) (4v) Considerando a seguinte informação, relativa a um armazém de produtos hortícolas, represente-a num documento XML válido segundo o XSD na página seguinte

Stock	
Produto	Preço
Batata	1.1€/kg
Cebola	0.9€/kg
Alho	2€/kg

Encomenda			
N°	Data	Produtos	Quantidade
1	15/1/2014	Batata	3000
1		Alho	200
2	14/1/2014	Cebola	400

5) Para o documento XML na página seguinte, relativo a produtos, apresente:

   a) (1v) Uma expressão Xpath para obter o peso total dos produtos em stock. (Ignore o espaço de nomes.)

   b) (1v) Uma expressão Xpath para obter o nome dos produtos ainda sem preço (Ignore o espaço de nomes.)

   c) (1v) O resultado da transformação do XML pelo XSL na última página, para simplificar, assuma que os dois últimos templates não existem

   d) (1v) Repita a alínea anterior mas considerando agora os templates todos

```xml
<schema
    xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:t="urn:feup.2014"
    targetNamespace="urn:feup.2014"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">
    <element name="produto">
        <complexType>
            <sequence>
                <element name="referencia" type="IDREF" form="unqualified"/>
                <element name="quantidade" type="nonNegativeInteger"/>
            </sequence>
        </complexType>
    </element>
    <element name="Encomenda">
        <complexType>
            <sequence>
                <element ref="t:produto" maxOccurs="unbounded"/>
            </sequence>
            <attribute name="data" type="date" use="required"/>
        </complexType>
    </element>
    <complexType name="pType">
        <attribute name="nome" type="string" use="required"/>
        <attribute name="preço" type="decimal"/>
        <attribute name="code" type="ID" form="qualified"/>
    </complexType>
    <element name="Stock">
        <complexType>
            <sequence>
                <element name="produto" type="t:pType" maxOccurs="unbounded"/>
            </sequence>
        </complexType>
    </element>
    <element name="Armazém">
        <complexType>
            <sequence>
                <element ref="t:Stock"/>
                <element ref="t:Encomenda" maxOccurs="unbounded"/>
            </sequence>
        </complexType>
    </element>
</schema>
```

```xml
<Produtos xmlns="urn:mercado.fruta">
    <!-- documento XML para a pergunta 5 -->
    <Stock>
        <produto nome="banana" quantidade="2000" un="kg"/>
        <produto nome="pêra" quantidade="3000"  un="kg"/>
        <produto nome="macã" quantidade="3500" un="kg"/>
        <produto nome="ananáz" quantidade="1000" un="kg"/>
    </Stock>
    <Preços>
        <produto nome="pêra" un="€">1.2</produto>
        <produto nome="macã" un="€">1.5</produto>
        <produto nome="banana" un="€">2</produto>
    </Preços>
</Produtos>
```

```xml
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/*">
        <xsl:copy>
            <xsl:apply-templates select="node()|@*"/>
        </xsl:copy>
    </xsl:template>
    <xsl:template match="node()">
        <xsl:apply-templates select="node()"/>
    </xsl:template>
    <xsl:template match="@*">
        <xsl:copy/>
    </xsl:template>
    <xsl:template match="t:produto" xmlns:t="urn:mercado.fruta">
        <xsl:copy>
            <xsl:apply-templates select="@*"/>
            <xsl:copy-of select="//*[@nome = current()/@nome]/text()" />
        </xsl:copy>
    </xsl:template>
    <xsl:template match="Stock/produto" priority="5">
        <fruta preço="Consultar elemento Preços">
            <xsl:apply-templates select="@*"/>
        </fruta>
    </xsl:template>
</xsl:stylesheet>
```