2016 - 2017

Programarea Clientului Web

conf dr. ing. Simona Caraiman

asist drd.inf. Tiberius Dumitriu

mailto: tiberiusdumitriu@yahoo.com

Universitatea Tehnica "Gheorghe Asachi" din Iasi
Facultatea de Automatica si Calculatoare

Tehnologii XML

Marcarea informatiilor pentru Web

Tehnologii XML

- Familia XML
  - XML – reprezentarea datelor semistructurate
  - Componente de baza
    - spatii de nume
    - transformarea documentelor XML: CSS, XSL
    - validarea documentelor XML: DTD, XML Schema
  - Limbaje bazate pe XML
- Procesari XML
  - Modelul DOM
  - Interfata SAX
- XML pentru servicii Web: REST, AJAX

XML - eXtensible Markup Language

- meta-limbaj de marcare
- descendent simplificat al SGML, utilizat in Web
- Versiuni:
  - XML 1.0 – Rec. W3C (2008, 5th edition)http://www.w3.org/TR/REC-xml/
  - XML 1.1 – Rec. W3C (2006, 2nd edition) http://www.w3.org/TR/xml11/
- proiectat pentru modelarea, transportul si stocarea datelor (nu pentru prezentare - HTML)
- marcajele nu sunt predefinite

XML - caracterizare

- format textual
  - human-readable & machine-readable
- marcaje descriptive: `<para>`, `<imagine/>`
- independenta datelor
  - comunicarea datelor intre sist. incompatibile
  - independenta hardware/software
  - extinderea marcajelor
- case-sensitivity

XML - trasaturi

- suport Web, implementare in toate limbajele de programare
- utilizare internationala
  - suport pentru Unicode
  - independent de codificare si limba
- meta-limbaj
  - permite definirea de noi limbaje, portabil
- solutie pentru reprezentarea continutului resurselor Web identificate prin URI

XML – structura documentelor

Constituenti:

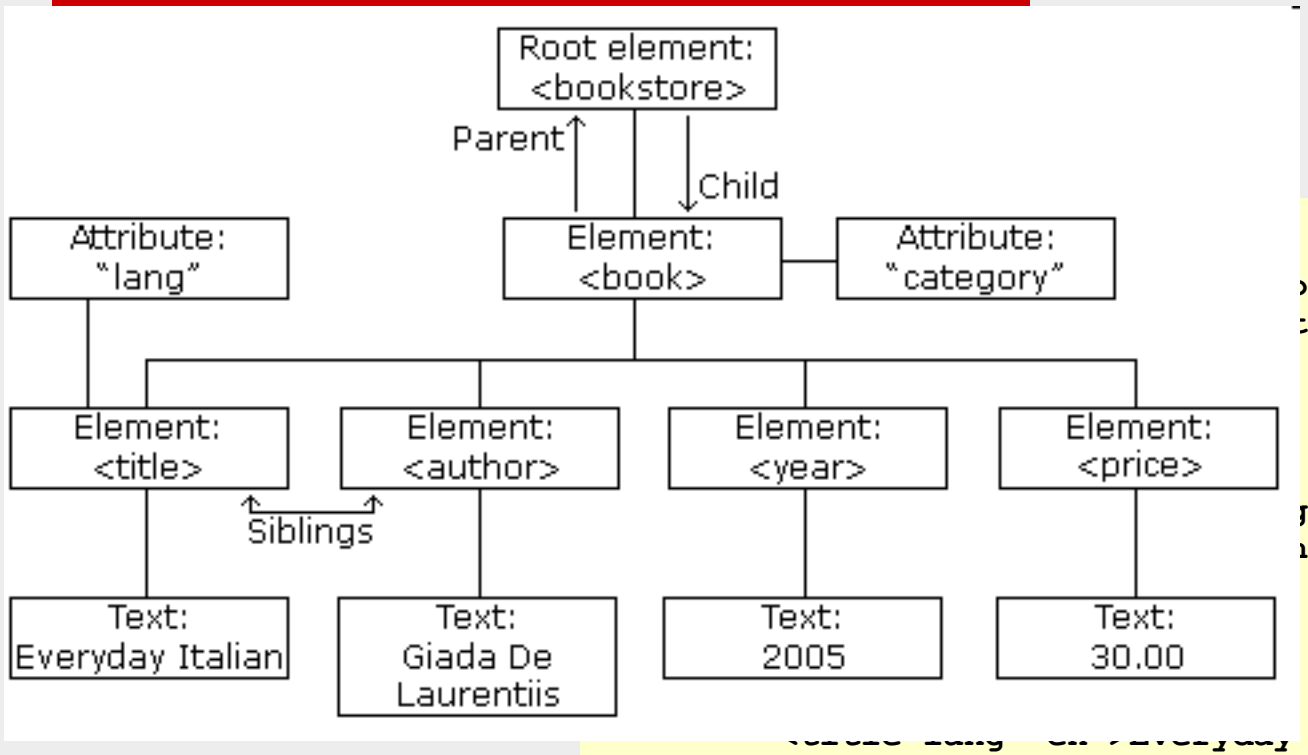
- declaratia xml
- elemente
- atribute
- entitati
- sectiuni de marcare
- instructiuni de procesare

XML – structura arborescenta

```
<root>
    <child>
        <subchild>.....</subchild>
    </child>
</root>
```

```
<bookstore>
    <book category="CHILDREN">
        <title lang="en">Harry Potter</title>
        <author>J K. Rowling</author>
        <year>2005</year>
        <price>29.99</price>
    </book>
    <book category="WEB">
        <title lang="en">Learning XML</title>
        <author>Erik T. Ray</author>
        <year>2003</year>
        <price>39.95</price>
    </book>
    <book category="COOKING">
        <title lang="en">Everyday Italian</title>
        <author>Giada De Laurentiis</author>
        <year>2005</year>
        <price>30.00</price>
    </book>
</bookstore>
```

XML – structura arborescenta




```
otter</title>
thor>

g XML</title>
hor>

                                    Italian</title>
        <author>Giada De Laurentiis</author>
        <year>2005</year>
        <price>30.00</price>
    </book>
</bookstore>
```

XML – Reguli de sintaxa

Document XML bine format (*well formed*):

- toate elementele trebuie sa aiba marcaje de sfarsit
- marcajele sunt case-sensitive
- elementele XML trebuie imbricate corect
- documentele XML trebuie sa aiba un element radacina
- valorile atributelor furnizate intre ghilimele

XML - constituenti

Declaratia XML

- specifica versiunea si codificarea documentului
- primul element al documentului
- apare o singura data
- trei atribute posibile:

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

XML - constituenti

Elementele

- componenta structurala a unui document XML (unitatea-text)
- specificate prin intermediul marcajelor de inceput si de sfarsit

  `<disciplina>PCW</disciplina>`
- pot avea continut vid

  `<disciplina></disciplina>` sau `<disciplina/>`
- **reguli** de formare a **numelor** elementelor:
  - contin litere, numere si alte caractere
  - nu pot incepe cu un numar sau caracter de punctuatie
  - numele incepand cu xml/XML sunt rezervate
  - nu pot contine spatii

XML - constituenti

Elementele

- trebuie sa fie inchise si imbricate corect
- case-sensitive
- pot contine text si/sau alte elemente

```
<facultate>
   AC are adresa
   <adresa>www.ace.tuiasi.ro</adresa>
   si este o facultate
</facultate>

<center><p><b>Salut!</b></p></center>
```

- sunt extensibile

XML - constituenti

Elementele

- trebuie sa fie inchise si imbricate corect
- case-sensitive
- pot contine

```
<bookstore>
    <book category="CHILDREN">
        <title lang="en">Harry Potter</title>
        <author>J K. Rowling</author>
        <ISBN>978-0590353403</ISBN>
        <year>2005</year>
        <publisher>Scholastic Press</publisher>
        <price>29.99</price>
    </book>
    …
</bookstore>
```

- sunt extensibile

XML - constituenti

Atribute

- furnizeaza informatii aditionale despre continut

```
<book category="CHILDREN">
  ...
</book>
```

- apar doar in tag-ul de inceput
- scrise intre ghilimele (simple sau duble)
- nu sunt acceptate atribute fara valoare
- case-sensitive
- evitarea atributelor:
  - nu pot contine valori multiple
  - nu pot contine structuri imbricate
  - nu pot fi usor expandate (pentru modificari ulterioare)
- metadate –> atribute; date -> elemente

XML - constituenti

Atribute

- furnizeaza infor
  <book category=
- ut
- e sau duble)
- ara valoare
- case-sensitive
- evitarea atributelor:
  - nu pot contine valori mu
  - nu pot contine structuri
  - nu pot fi usor expandate
- metadate –> atribute;

```
<message date="12/03/2009">
    <from>Tarzan</from>
    <to>Jane</to>
    <body>Me Tarzan, you Jane</body>
</message>
```

```
<message
    <date>12/03/2009</date>
    <from>Tarzan</from>
    <to>Jane</to>
    <body>Me Tarzan, you Jane</body>
</message>
```

```
<message
    <date>
        <day>12</day>
        <month>03</month>
        <year>2009</year>
    </date>
    <from>Tarzan</from>
    <to>Jane</to>
    <body>Me Tarzan, you Jane</body>
</message>
```

XML - constituenti

Atribute

- furnizeaza informatii ad
  ```
  <book category="CHILDR
  ...
  </book>
  ```
- apar doar in tag-ul de i
- scrise intre ghilimele (s
- nu sunt acceptate atrib
- case-sensitive
- evitarea atributelor:
  - nu pot contine valori
  - nu pot contine struct
  - nu pot fi usor expand
- metadate –> atribute; date -> elemente

```
<message id="101"
    <date>
        <day>12</day>
        <month>03</month>
        <year>2009</year>
    </date>
    <from>Tarzan</from>
    <to>Jane</to>
    <body>Me Tarzan. You Jane</body>
</message>
<message id="102"
    <date>
        <day>12</day>
        <month>03</month>
        <year>2009</year>
    </date>
    <from>Jane</from>
        <to>Tarzan</to>
        <body>Show me the
    jungle</body>
</message>
```

XML - constituenti

Referinte la entitati

- entitate XML = unitate de text (un singur caracter, un alt document)
- constructia sintactica:

`&nume_entitate;` sau `%nume_entitate;` sau `&#numar;`

- entitati predefinite:

Entitate	Referinta la entitatate
<	&lt;
>	&gt;
&	&amp
‘	&apos;
“	&quot;

XML - constituenti

Referinte la entitati

- entitate XML = unitate de text (un singur caracter, un

```
<!ENTITY s ”Simona">
<!ENTITY sc "&s; Caraiman">
```

- construcția sintactică:

&nume_entitate; sau &nume_entitate; sau &#numar;

- entitati predefinite:

Entitate	
<	
>	
&	
‘	
“	"

```
<!ENTITY chap1 SYSTEM "chapter-1.xml">
<!ENTITY chap2 SYSTEM "chapter-2.xml">
<!ENTITY chap3 SYSTEM "chapter-3.xml">

<?xml version="1.0" encoding="utf-8"?>
<!-- Pull in the chapter content: -->
&chap1;
&chap2;
&chap3;
```

XML - constituenti

Sectiuni de marcare

- anumite parti din document necesita procesari speciale:
  - CDATA (*character data*)– inhiba procesarea XML
  - ex.: includerea de cod sursa
  - sintaxa: `<![CDATA[…]]>`
  - sectiunile CDATA nu pot fi imbricate si nu pot contine sirul `]]>`

XML - constituenti

Sectiuni de marcar

- anumite parti dir
  procesari speciale
  - CDATA (*characte*
    procesarea XML
  - ex.: includerea de cod sursa
  - sintaxa: **<![CDATA[…]]>**
  - sectiunile CDATA nu pot fi imbricate si nu pot contine sirul **]]>**

```
<script>
<![CDATA[
function matchwo(a,b) {
  if (a < b && a < 0) then {
   return 1;
  }
  else {
   return 0;
  }
}
]]>
</script>
```

XML - constituenti

Instructiuni de procesare

- includ informatii privitoare la aplicatiile (externe) care urmeaza a fi executate pentru procesarea continutului
- **<?nume_apl …?>**
- ***nume_apl*** nu poate fi ***xml***
- ex.: invocare interpretor php

```
<script>
        <?php echo “<p>Salut!\n</p>”; ?>
</script>
```

- ex.: asociere foi de stiluri

```
<?xml:stylesheet type=“text/css” href=“stil.css” ?>
```

XML - utilizare

- separarea datelor de HTML
- simplificarea data sharing
- simplificarea transportului datelor
- simplificarea schimbarilor de platforma
- crearea de noi limbaje Internet

Tehnologii XML – Familia XML

- **XML specification**
  - **XML Infoset** – descrie o reprezentare abstracta a unui doc XML
  - **XPath Data Model** – adresarea unor parti ale unui doc XML
  - **DOM (Document Object Model)** – defineste modul in care datele sunt structurate, accesate si manipulate
  - **XQuery** – limbaj de interogare a colectiilor de date XML
- **XML Accessories**
  - extind capabilitatile specificate in XML
  - **XML Schema, XML Names**
- **XML Transformers (Transducers)**
  - transformarea documentelor XML in alte (tipuri de) documente (XML, XHTML, etc.)
  - **CSS**, **XSL (eXtensible Stylesheet Language)**
- **XML Applications**
  - limbaje bazate pe XML

Aplicatii XML

- Formatarea continutului
  - in cadrul navigatorului Web: *XHTML*
  - in medii mobile, fara fir: WML (*Wireless Markup Language*)
  - formulare electronice *XForms*
  - grafica vectoriala: SVG (*Scalable Vector Graphics*)
  - grafica 3D: X3D (*Extensible Three Dimensions*)

- Reprezentarea diferitelor tipuri de continut
  - expresii matematice: *MathML*
  - continut multimedia sincronizat: SMIL (*Synchronized Multimedia Integration Language*)
  - informatii vocale: *VoiceXML*
  - componente ale interfetei cu utilizatorul: XUL (*Extensible User-interface Language*), XAML (*Extensible Application Markup Language*)
  - stocarea informatiilor prelucrate de suite de birou (ex. OpenOffice): *OpenDocument*

Aplicatii XML (cont.)

- Descrierea resurselor Web
  - cadrul general: RDF (*Resource Description Frameweork*)
  - exprimarea vocabularelor de meta-date: RSS (*Really Simple Syndication*), Atom
  - exprimarea de ontologii: OWL (*Web Ontology Language*)
- Descrierea serviciilor Web
  - serializarea datelor transmise conform paradigmei RPC (Remote Procedure Call): *XML-RPC*
  - descrierea serviciilor Web: WSDL (*Web Service Description Language*)
  - exprimarea protocolului de transfer: SOAP (*Simple Object Access Protocol*)

Instrumente XML

- analizoare (Expat, libxml, MSXML, Apache Xerces)
- instrumente de vizualizare (Firefox, OpenOffice, <oXygen/>, XMLSpy, MS Visual Studio, etc)
- instrumente de formatare (FOP, Saxon, Xalan, XEP, etc)
- instrumente de convertire/arhivare (Tidy, OpenSP)
- sisteme de gestiune a bazelor de date orientate pe text (dbXML, eXist, etc)
- instrumente de modelare conceptuala (pOWL)

Familia XML – componente de baza

1. Spatiile de nume
2. Transformarea documentelor XML
3. Validarea documentelor XML

Spatii de nume

- date din diverse surse XML => **conflicte** de nume
- *spatiu de nume* (*namespace*): vocabular utilizat pentru identificarea in mod unic a elementelor si a atributelor

```
<!-- carti -->
<book>
    <title>The Godfather</title>
    <author>Mario Puzo</author>
    <year>1969</year>
    <genre>crime</genre>
</book>
<!- persoane -->
<person>
    <title>Mr.</title>
    <name>John Doe</name>
    <email>johndoe@mail.com</email>
</person>
```

Spatii de nume

- vocabularul poate fi desemnat de un URI
  - specificat prin atributul **xmlns**
- optional, se poate atasa un identificator unic fiecarui vocabular
  - *QName* (nume calificat): **`prefix:nume`**

```
<lentBook xmlns:b="http://www.library.com/books/"
xmlns:p="http://www.library.com/people/">
  <b:book>
    <b:title>The Godfather</b:title>
    <b:author>Mario Puzo</b:author>
    <b:year>1969</b:year>
    <b:genre>crime</b:genre>
  </b:book>
  <p:person>
    <p:title>Mr.</p:title>
    <p:name>John Doe</p:name>
    <p:email>johndoe@mail.com</p:email>
  </p:person>
<lentBook>
```

URI

ator

unic fiecare vocabular

- QNam

```
<lentBook>
  <b:book xmlns:b="http://www.library.com/books/">
    <b:title>The Godfather</b:title>
    <b:author>Mario Puzo</b:author>
    <b:year>1969</b:year>
    <b:genre>crime</b:genre>
  </b:book>
  <p:person xmlns:p="http://www.library.com/people/">
    <p:title>Mr.</p:title>
    <p:name>John Doe</p:name>
    <p:email>johndoe@mail.com</p:email>
  </p:person>
<lentBook>
```

Transformarea documentelor XML

- un document XML separa continutul de maniera de formatare/procesare
- pentru a prezenta utilizatorului datele XML, trebuie specificata o modalitate de redare (asa-numita foaie de stiluri – *stylesheet*)
- **Solutii:**
  - CSS (*Cascading Style Sheet*)
    - sintaxa non-XML, flexibilitate limitata, nu exista context
  - XSL (*Extensible Stylesheet Language*)
    - sintaxa XML, flexibilitate mai mare, procesare in functie de context, se pot opera modificari de structura XML,...

Transformarea documentelor XML

Strategii:

- 1 foaie de stiluri, N documente
  - se mentine consistenta formatului pentru documente multiple
  - usor de dezvoltat, aplicat si controlat
- N foi de stiluri, 1 document
  - se permit formatari diferite in functie de mediile de redare (ecran, imprimanta, etc) sau de preferinte (ex. skin-uri)
  - usor de produs documente derivate: selectii, sumarizari, indexari, cataloage, …

Transformarea documentelor XML

Prezentarea continutului XML via CSS

```
<!-- xml file -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="cd_catalog.css"?>
<CATALOG>
  <CD>
    <TITLE>Hide your heart</TITLE>
    <ARTIST>Bonnie Tyler</ARTIST>
    <COUNTRY>UK</COUNTRY>
    <COMPANY>CBS Records</COMPANY>
    <PRICE>9.90</PRICE>
    <YEAR>1988</YEAR>
  </CD>
  …
</CATALOG>
```

Trans

Prezen

cont

```
<!-- xml
<?xml ve
<?xml-st
<CATALOG
  <CD>
    <TIT
    <ART
    <COU
    <COM
    <PRI
    <YEA
  </CD>
  ...
</CATALO
```

```
<!-- css file -->
CATALOG {
  background-color: #ffffff;
  width: 100%;
}
CD {
  display: block;
  margin-bottom: 30pt;
  margin-left: 0;
}
TITLE {
  color: #FF0000;
  font-size: 20pt;
}
ARTIST {
  color: #0000FF;
  font-size: 20pt;
}
COUNTRY,PRICE,YEAR,COMPANY {
  display: block;
  color: #000000;
  margin-left: 20pt;
}
```

Empire Burlesque Bob Dylan
USA
Columbia
10.90
1985

Hide your heart Bonnie Tyler
UK
CBS Records
9.90
1988

Greatest Hits Dolly Parton
USA
RCA
9.90
1982

Still got the blues Gary Moore
UK
Virgin records
10.20
1990

Eros Eros Ramazzotti
EU
BMG
9.90
1997

One night only Bee Gees
UK

Transformarea documentelor XML

XSL – eXtensible Stylesheet Language

- Scopuri:
  - Transformarea structurii/continutului documentelor XML
  - Rescrierea documentelor XML => documente XML/XHTML/alte formate
- Inspirat din DSSSL (Document Style Semantics and Specification Language) folosit pentru SGML
- Trei componente:
  - XSLT (*XSL Transformations*) – limbaj pentru transformarea documentelor XML
  - XPath – limbaj pentru navigare in documente XML
  - XSL-FO (*XSL Formatting Objects*) – limbaj pentru formatarea documentelor XML pe baza unor obiecte de formatare
- Limbaj descriptiv **bazat pe reguli**, **orientat-evenimente**

Transformarea documentelor XML

XSL – eXtensible Stylesheet Language

- Document XML ≡ arbore de noduri
- Tipuri de noduri :
  - Radacina, Elemente, Text, Atribute, Spatii de nume, Instructiuni de procesare, Comentarii
    - pentru noduri de tip text caracterele rezervate trebuie rescrise cu entitati
- **Reguli** compuse dintr-un *pattern* (model) si o actiune
- Modelul este exprimat in XPath
- Actiunea este specificata in XSLT
- Transformarile se aplica recursiv tuturor nodurilor XML care satisfac modelul/sabloanele de reguli (*pattern-matching*)

XPath (modelul)

- Recomandare W3C (1999) **http://www.w3.org/TR/xpath**
- Permite adresarea unor parti dintr-un document XML
- Opereaza la nivelul structurii abstracte a documentelor XML (arborele XML)
- Contine o biblioteca de functii standard
  - siruri, valori numerice, date & time, manipularea nodurilor, Qname-urilor si secventelor, valori booleene, etc.
- Constructia de baza este **expresia XPath**
  - Utilizata pentru navigarea in documente XML

XPath (modelul)

Constructia de baza este **expresia XPath**

- evaluarea se realizeaza in functie de context:
  - un nod al documentului XML
  - pozitie
  - functie de biblioteca
  - declaratie a unui spatiu de nume
- in urma evaluarii expresiei este returnat un obiect:
  - multime de noduri (*node-set*)
  - boolean (*true*, *false*)
  - numar (*float*)
  - sir de caractere

XPath (modelul)

Operatori:

- descendent /
- traversare recursiva //
- wildcard *
- nodul curent .
- nodul parinte ..
- atribut @
- spatiu de nume ::
- filtru/index []
- pentru booleeni si numere: operatorii uzuali
  or and = != <= < >= >
  + - * div mod

XPath (modelul)

Exemple:

- **`table/tr[@align="center" or @valign="top"]`**
  - selecteaza elementele dintr-un element **`<tr>`** avand specificate atributele **`align="center"`** sau **`valign="top"`** din cadrul unui element **`<table>`**

- **`capitol/nume | capitol/autor`**
  - va furniza toate elementele **`<nume>`** si **`<autor>`** descendente ale elementului **`<capitol>`**

XPath (modelul)

Functii de baza:

- Noduri: `id()`, `position()`, `count()`, `name()`, `namespace-uri()`, `last()`, …
- Tipuri de noduri: `node()`, `text()`, `comment()`, `processing-instruction()`
- Siruri: `concat()`, `starts-with()`, `contains()`, `substring()`, `string-length()`, `translate()`, …
- Boolean: `not()`, `true()`, `false()`, …
- Numere: `sum()`, `round()`, `floor()`, `number()`, …

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<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="en">Harry Potter</title>
      <author>J K. Rowling</author>
      <year>2005</year>
      <price>29.99</price>
    </book>

    <book category="WEB">
      <title lang="en">XQuery Kick Start</title>
      <author>James McGovern</author>
      <author>Per Bothner</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>
```

/bookstore/book/title/text()

```
Everyday Italian
Harry Potter
XQuery Kick Start
Learning XML
```

/bookstore/book[1]/price/text()

```
30.00
```

/bookstore/book[price>35]/price

```
49.99
39.95
```

/bookstore/book[@category=
'WEB']/title

```
XQuery Kick Start
Learning XML
```

XSLT (actiunea)

XSLT – XSL Transformations

- Recomandare W3C (1999)

  http://www.w3.org/TR/xslt
- Transforma documentele XML in alte tipuri de continut (XML, HTML, text etc.)
  - documentul original nu este modificat
- Gandit pentru a fi parte din XSL

  (XSL ≡ XSLT + XSL-FO)

  http://www.w3.org/TR/xsl

  http://www.w3.org/TR/xml-stylesheet/
- Poate fi utilizat independent de XSL

XSLT

- pentru a putea fi folosite, constructiile XSLT trebuie sa apartina spatiului de nume desemnat de URI-ul: **http://www.w3.org/1999/XSL/Transform**

- o foaie de stiluri XSLT are drept element radacina **<xsl:stylesheet>** sau **<xsl:transform>**

XSLT

- include sabloane de transformare (macar un sablon la nivelul radacina)

  template rules : *instructions*

- pentru transformare se utilizeaza expresii **XPath** folosite la:
  - selectarea nodurilor dorite a fi procesate
  - specificarea conditiilor de procesare
  - generarea textului de iesire (ex. HTML)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="style.xsl"?>
<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="en">Harry Potter</title>
      <author>J K. Rowling</author>
      <year>2005</year>
      <price>29.99</price>
    </book>

    <book category="WEB">
      <title lang="en">XQuery Kick Start</title>
      <author>James McGovern</author>
      <author>Per Bothner</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>
```

```
<?xml version="1.0" enc
<?xml-stylesheet type="
<bookstore>

    <book category="CO
      <title lang="en">E
      <author>Giada De
      <year>2005</yea
      <price>30.00</pri
    </book>

    <book category="CH
      <title lang="en">H
      <author>J K. Rowl
      <year>2005</yea
      <price>29.99</pri
    </book>

    <book category="W
      <title lang="en">X
      <author>James Mc
      <author>Per Bothn
      <year>2003</yea
      <price>49.99</pri
    </book>

    <book category="W
      <title lang="en">L
      <author>Erik T. Ra
      <year>2003</yea
      <price>39.95</pri
    </book>
</bookstore>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My Book Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Author</th>
      <th>Year</th>
    </tr>
    <xsl:for-each select="bookstore/book">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="author"/></td>
      <td><xsl:value-of select="year"/></td>
    </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

My Book Collection

Title	Author	Year
Everyday Italian	Giada De Laurentiis	2005
Harry Potter	J K. Rowling	2005
Learning XML	Erik T. Ray	2003

XSLT

Modelul XSLT

- o lista de noduri sursa (*input*) este procesata pentru a genera un fragment de arbore de noduri destinatie (*output*)

- initial se proceseaza nodul radacina, la care se insereaza noduri copil generate de sabloane aplicate unei liste de noduri selectate (recursiv) prin *pattern-matching* – via expresii Xpath

XSLT

Reguli de aplicare XSLT

- regulile sabloanelor identifica noduri asupra carora se vor aplica transformari
- selectarea nodurilor se face prin XPath
- un sablon se defineste prin elementul **<xsl:template>**
- aplicarea unui sablon se realizeaza cu elementul **<xsl:apply-templates>**

XSLT

Crearea arborelui de iesire

- intr-un sablon, orice elemente ce nu apartin spatiului de nume XSLT sunt copiate (fara a fi operate modificari) in arborele de iesire
- pot fi generate si alte tipuri de noduri: **<xsl:element>**, **<xsl:attribute>**, **<xsl:text>**, **<xsl:comment>** etc.
- extragerea unor valori se face prin elementul **<xsl:value-of>**
- controlul iesirii: **<xsl:output>**

XSLT

Programe XSLT

- Constructii repetitive:
  - `<xsl:for-each>`
- Constructii de test:
  - `<xsl:if>, <xsl:choose>, <xsl:when>, <xsl:otherwise>`
- Copierea nodurilor:
  - `<xsl:copy>, <xsl:copy-of>`
- Sortarea nodurilor:
  - `<xsl:sort>`
- Includerea de alte foi de stiluri:
  - `<xsl:include>`
- Variabile si parametri:
  - `<xsl:variable>, <xsl:param>`

XSLT

Functii XSLT de baza:

- nodul curent:
  - `current()`
- verifica existenta unei functii:
  - `function-available()`
- formateaza valori numerice:
  - `format-number()`
- ofera informatii privitoare la sistemul de procesare:
  - `system-property()`

Validarea documentelor XML

Necesitati:

- informatiile marcate in XML sa poata fi regasite, reutilizate si partajate intre aplicatii
- cunoasterea:
  - elementelor/atributelor ce pot fi specificate
  - modului lor de structurare (*e.g.*, ordinea, numarul minim/maxim de aparitii,...)
  - tipului continutului
  - ce este valid si ce reprezinta eroare

Validarea documentelor XML

Solutie:

- specificarea *modelului structural* al documentului (multimea de elemente si atribute permise si regulile de marcare)
- realizata de:
  - companii (Adobe – XMP, Sun - JSP)
  - industrie (dispozitive mobile - WML)
  - persoane ce impartasesc un scop comun (dezvoltatori ai OpenOffice)
  - producatori de instrumente specifice (Microsoft, Oracle)
  - consortii, organizatii non-profit (W3C, OASIS)

Validarea documentelor XML

- Modelul structural
  - se aplica unei clase de documente XML, in vederea verificarii corectitudinii instantelor apartinand clasei
- Se au in vedere:
  - numirea elementelor/atributelor
  - definirea regulilor de utilizare a acestora
  - specificarea structurii si continutului
  - oferirea unui set de conventii de numire

Validarea documentelor XML

- specificarea unui set de constrangeri asociate documentelor
- modalitati de specificare:
  - descrieri – DTD, XMLSchema
    - “exista un element `<student>` avand un atribut `nume` care are continutul…”
  - reguli – Schematron
    - “orice element `<student>` va avea un atribut `nume`, iar continutul acestui atribut se va conforma regulii …”
  - sabloane – RELAX NG
    - “orice document din clasa *student* trebuie sa se potriveasca urmatorului sablon …”

EXEMPLU SCHEMATRON

```
<schema xmlns="http://www.ascc.net/xml/schematron" >
    <pattern name="Print positive result only">
      <rule context="AAA">
        <report test="BBB">BBB element is present.</report>
        <report test="@name">AAA contains attribute name.</report>
      </rule>
    </pattern>
    <pattern name="Print negative result only">
      <rule context="AAA">
        <assert test="BBB">BBB element is missing.</assert>
        <assert test="@name">AAA misses attribute name.</assert>
      </rule>
    </pattern>
</schema>
```

conforma regulil

- sabloane – RELAX
  - "orice document potriveasca urma

```
<AAA>
    <BBB/>
</AAA>
```

Output:

Pattern: **Print positive result only**
/AAA: BBB element is present.

Pattern: **Print negative result only**
/AAA: AAA misses attribute name.

EXEMPLU RELAX NG

```
<element name="book" xmlns="http://relaxng.org/ns/structure/1.0">
  <oneOrMore>
    <element name="page">
      <text/>
    </element>
  </oneOrMore>
</element>
```

- descriere – DTD, XMLSchema
  - "exista un element ... nume care are co...
- reguli – Schematro...
  - "orice element <... nume, iar continu... conforma regulii ..."
- sabloane – RELAX NG
  - "orice document din clasa *student* trebuie sa se potriveasca urmatorului sablon ..."

```
<book>
  <page>This is page one.</page>
  <page>This is page two.</page>
</book>
```

Validarea documentelor XML

DTD – Document Type Definition

- specificare formala a tipurilor de documente (constituenti + structura)
- documentele XML pot avea sau nu un DTD atasat
- daca DTD-ul lipseste, documentul trebuie sa respecte un numar minim de constrangeri (sa fie bine formate - *well formed*)

Validarea documentelor XML

DTD – Document Type Definition

- *intern* sau *extern* documentului
- regulile sintactice de specificare a meta-elementelor DTD provin de la SGML
- DTD-ul poate defini:
  - structura continutului
  - indicatori de aparitie
  - conectori
  - exceptii

Validarea documentelor XML

DTD – intern

- the document type declaration must be placed between the XML declaration and the first element (root element) in the document.
- the keyword DOCTYPE must be followed by the name of the root element in the XML document.
- the keyword DOCTYPE must be in upper case.

Validarea documentelor XML

DTD – intern

- the document typ
  between the XML
  (root element) in the document.

```
<!DOCTYPE root_element [

  Document Type Definition (DTD):
  elements/attributes/entities/notations/
  processing instructions/comments/PE references

]>
```

Exemplu

```
<?xml version="1.0" standalone="yes" ?>

<!--open the DOCTYPE declaration -
  the open square bracket indicates an internal DTD-->
<!DOCTYPE foo [

<!--define the internal DTD-->
  <!ELEMENT foo (#PCDATA)>

<!--close the DOCTYPE declaration-->
]>

<foo>Hello World.</foo>
```

Validarea documentelor XML

DTD – extern

- pot fi utilizate in documente multiple
- *privat:*

<!DOCTYPE root_element **SYSTEM** "DTD_location”>

- *public*:

<!DOCTYPE root_element **PUBLIC** "DTD_name" "DTD_location”>

- document XML ce contine elemente, atribute sau entitati referite sau definite intr-un DTD extern:

<?xml version=“1.x” standalone=“**no**”?>

Validarea documentelor XML

DTD – extern

- pot fi utilizate
- *privat:*

<!DOCTYPE root_eler

- *public*:

<!

-

ocation”>

te sau

tern:

?>

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE bookstore SYSTEM “Book.dtd">
<bookstore>
    <book>
        <title lang="en">Harry Potter</title>
        <author>J K. Rowling</author>
        <ISBN>978-0590353403</ISBN>
        <year>2005</year>
        <price>29.99</price>
    </book>
</bookstore>
```

```
<!DOCTYPE bookstore [
    <!ELEMENT bookstore (book+)>
    <!ELEMENT book (title, author, ISBN, year, price)>
    <!ELEMENT title (#PCDATA)>
    <!ELEMENT author (#PCDATA)>
    <!ELEMENT ISBN (#PCDATA)>
    <!ELEMENT year (#PCDATA)>
    <!ELEMENT price (#PCDATA)>
    <!ATTLIST title lang (en|fr|de|ro) “en”>
]>
```

Validarea documentelor XML

DTD – extern

- po
- *pr*

<!DOC

- *pu*

<!DOC

- do

en

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
  "http://www.w3.org/TR/REC-html40/loose.dtd">
<HTML>
<HEAD>
<TITLE>A typical HTML file</TITLE>
</HEAD>
<BODY>
  This is the typical structure of an HTML file. It follows
  the notation of the HTML 4.0 specification, including tags
  that have been deprecated (hence the "transitional" label).
</BODY>
</HTML>
```

Validarea documentelor XML

XML Schema

- alternativa la DTD, bazata pe XML
- descrie structura unui document XML
- *XML Schema Definition* (XSD)
- avantaje:
  - extensibile
  - mai bogate si mai puternice decat DTD
  - scrise in XML
  - suporta *tipuri de date*
  - suporta *spatii de nume*
- se foloseste spatiul de nume definit de http://www.w3.org/2001/XMLSchema

Validarea documentelor XML

XML Schema

Defineste:

- elementele ce pot aparea intr-un document
- atributele ce pot aparea intr-un document
- elementele care sunt de tip *child*
- numarul de elemente *child*
- daca un element este vid sau daca poate include text
- tipurile de date pentru elemente si atribute
- valori implicite si fixe pentru elemente si atribute

Validar

XMLSch

Defineste

- elemente ce pot apărea într-un document

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<book
xmlns=http://www.somebookstore.com
xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation="http://www.somebookstore.com/book.xsd">
        <title lang="en">Harry Potter</title>
        <author>J K. Rowling</author>
        <ISBN>978-0590353403</ISBN>
        <year>2005</year>
        <price>29.99</price>
</book>
```

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="book">
       <xs:complexType>
          <xs:sequence>
             <xs:element name="title" type="xs:string"/>
             <xs:element name="author" type="xs:string"/>
             <xs:element name="ISBN" type="xs:string"/>
             <xs:element name="year" type="xs:unsignedInt"/>
             <xs:element name="price" type="xs:float"/>
          </xs:sequence>
       </xs:complexType>
    </xs:element>
</xs:schema>
```

Procesarea documentelor XML

Procesoare (analizoare, parser) XML:

- *fara validare* – verifica doar daca documentul este bine-format (*expat*, *libxml*, *MSXML*,...)

- *cu validare* – verifica daca documentul este valid, folosind un DTD sau o Schema (*Apache Xerces*, *JAXP*, *MSXML*,...)

Procesarea documentelor XML

Tipuri de procesari XML:

- Procesare manuala
  (*e.g.*, expresii regulate)
- **Procesare obiectuala**
  (*e.g.*, DOM)
- Procesare condusa de evenimente
  (*e.g.*, SAX)
- Procesare simplificata
  (XML Reader/Writer, *e.g.*, .NET XmlTextReader/Writer)
- Procesare particulara (via API-uri specializate –
  e.g., RDF, RSS, SOAP, SVG)

Document Object Model (DOM)

- o reprezentare a documentelor (X)HTML/**XML** sub forma unui set de obiecte
- cross-platform
- language independent
- permite accesarea si modificarea *dinamica* a *continutului*, *structurii* si *stilului* unui document

Document Object Model (DOM)

- Standardizare (W3C) – 3 parti:
  - Core DOM – model standard pt. orice document structurat
  - **XML DOM – model standard pt. documente XML**
  - HTML DOM - model standard pt. documente HTML

- DOM Level 1 (1998)
  - core elements
- DOM Level 2 (2000)
  - getElementById
  - event model
  - suport pt. spatii de nume XML
  - suport pt. CSS
- DOM Level 3 (2004)
  - suport pt. Xpath
  - tratare evenimente tastatura
  - interfata pt. serializarea documentelor in format XML
- DOM Level 4 (Nov. 2015 – W3C Recomm.)

Modelul DOM

- **Scop**: procesarea obiectuala a documentelor XML/HTML
- Interfata abstracta de programare a aplicatiilor (API) pentru XML/HTML
- Independent de platforma & limbaj
- Defineste
  - structura logica arborescenta a documentelor XML
  - modalitatile de accesare si modificare a lor
- Document ≡ set de obiecte

Modelul DOM

Interfete DOM

- documentul XML = set de *obiecte nod*
  - nodurile pot fi accesate cu JavaScript sau alt lbj de programare
- interfata de programare a DOM este definita de un set standard de *metode* si *proprietati*
- ofera o modalitate de accesare si de modificare a reprezentarii interne a unui document XML
- nu implica o implementare concreta, particulara:
  - se ofera interfete de procesare independente de implementare
  - specificarea interfetelor: IDL (*Interface Description Language*)

Modelul DOM

Interfete DOM

- Proprietati XML DOM
  - x.nodeName - the name of x
  - x.nodeValue - the value of x
  - x.nodeType - the node type of x
  - x.parentNode - the parent node of x
  - x.childNodes - the child nodes of x
  - x.attributes - the attribute nodes of x

- Metode XML DOM
  - x.getElementsByTagName(*name*) - get all elements with a specified tag name
  - x.appendChild(*node*) - insert a child node to x
  - x.removeChild(*node*) - remove a child node from x

Modelul DOM

Interfete DOM

- Proprietati XML DOM
  - x.nodeName - the name of x
    - Read-only
    - nod element = tag name
    - nod atribut = attribute name
    - nod text = #text
    - nod document= #document
  - x.nodeValue – the value of x
    - nod element = undefined
    - nod text = textul in sine
    - nod atribut = valoarea atributului

Modelul DOM

Interfete DOM

- Proprietati XML DOM
  - x.nodeType - the node type of x
    - Read-only
    - nod element = 1
    - nod atribut = 2
    - nod text = 3
    - nod comentariu = 8
    - nod document= 9

Modelul DOM

- Traversarea arborelui DOM

```
x=xmlDoc.documentElement.childNodes;
for (i=0;i<x.length;i++)
  {
  document.write(x[i].nodeName);
  document.write(": ");
  document.write(x[i].childNodes[0].nodeValue);
  document.write("<br />");
  }
```

```
x = xmlDoc.documentElement.firstChild;
while(x){
  document.write(x.nodeName);
  document.write(": ");
  document.write(x.childNodes[0].nodeValue);
  document.write("<br />");
  x = x.nextSibling;
}
```

Modelul DOM

DOM - Implementari

- **domxml** – extensie pentru PHP
- **JAXP** – parte integranta din J2SE (javax.xml.*)
- **JDOM** – interfata de programare special construita pentru Java: **http://www.jdom.org**
- **libxml** – API oferit de GNOME: **http://xmlsoft.org**
- **MSDOM** – procesari XML pe partea client/server in C/C++, JScript si VBScript (MSIE, IIS+ASP, Windows, …) – inclus in MSXML SDK
- **Xerces DOM API** – platforma XML pentru C++ si Java: **http://xml.apache.org/**
- **XmlDocument** – clasa oferita de .NET Framework (C#, J#, VB.NET,...)
- **XML::DOM** – modul Perl pentru DOM 1, bazat pe Expat (XML::Parser)
- …

Procesarea documentelor XML

Tipuri de procesari XML:

- Procesare manuala
  (*e.g.*, expresii regulate)
- Procesare obiectuala
  (DOM & non-DOM)
- **Procesare condusa de evenimente**
  (SAX & non-SAX)
- Procesare simplificata
  (XML Reader/Writer)
- Procesare particulara (via API-uri specializate –
  e.g., RDF, RSS, SOAP, SVG)

Interfata SAX

SAX – *Simple API for XML*

- Caracterizare
- Modelul procesarii
- Implementari
- SAX *versus* DOM

Interfata SAX

- **Scop**:
  - manipularea documentelor XML fara ca in prealabil sa fie construit arborele de noduri-obiect

  ⇒ documentul nu trebuie stocat complet in memorie inainte de a fi efectiv prelucrat

- ofera o procesare XML secventiala (liniara), orientata-evenimente

Interfata SAX

- efort independent (de W3C) de standardizare a procesarii XML condusa de evenimente
  - SAX 1.0
  - SAX 2.0 (spatii de nume + extensii)

- larg acceptat ca standard industrial
  - http://www.megginson.com/SAX/
  - http://www.saxproject.org

Interfata SAX

Modelul procesarii

- pentru fiecare tip de constructie XML (inceput de *tag*, sfirsit de *tag*, continut, instructiune de procesare, comentariu,...) se va genera un eveniment care va fi tratat de o functie/metoda (*handler*)
  - functiile de tratare se specifica de catre programator, pentru fiecare tip de constructie in parte
- programul consuma si trateaza evenimente produse de procesorul SAX

Interfata SAX

Modelul procesarii

- pentru fiecare tip de constructie XML (inceput de *tag*, sfirsit de *tag*, continut, instructiune de

```
<?xml version="1.0"?>
<doc>
<para>Hello, world!</para>
</doc>
```

```
start document
start element: doc
start element: para
characters: Hello, world!
end element: para
end element: doc
end document
```

Interfata SAX

Modelul procesarii

- Minimal, trebuie definite urmatoarele functii/metode:
  - trateaza_tag_inceput (*procesor*, *tag*, *atrib*)
  - trateaza_tag_sfirsit (*procesor*, *tag*)
  - trateaza_date_caract (*procesor*, *date*)

Interfata SAX

Modelul procesarii

- se ataseaza pentru fiecare eveniment de aparitie a *tag*-ului de inceput, a *tag*-ului de sfirsit si a datelor-continut una dintre functiile de tratare oferite de SAX, respectiv:
  - set_element_handler (trateaza_tag_inceput, trateaza_tag_sfirsit)
  - set_character_data_handler (trateaza_date_caract)

Interfata SAX

- **implementarea de referinta SAX**: pachetul Java `org.xml.sax`
- 5 grupuri de clase si interfete:
  - interfete implementate de procesorul XML – analizorul XML se mai numeste si *SAX Driver*
  - interfete implementate de aplicatia care doreste sa prelucreze documentele XML via driverul SAX: `DocumentHandler`, `ErrorHandler, DTDHandler, EntityResolver` - optionale
  - clase SAX standard (atat pentru procesoare cat si pentru aplicatii): `inputSource, SAXException, SAXParseException` si `HandlerBase` – implementate in intregime de SAX
  - clase aditionale specifice Java, complet implementate: `Parser-Factory, AttributeListImpl` si `LocatorImpl`
  - clase demonstrative (in fapt, aplicatii Java): nu fac parte din specificatiile de baza ale SAX si pot sa nu apara in implementari SAX in alte limbaje

Interfata SAX

Implementari:

- **libxml** – API oferit de GNOME (C)
- **MSSAX** – procesari SAX in C/C++, JScript, VBScript – inclus in MSXML SDK (vezi si SAX Win32 AppWizard)
- **org.xml.sax** – API pentru Java
- **QSAX** – parte a Trillian Qt (C++)
- **Xerces SAX API** – platforma XML pentru C++ si Java: **http://xml.apache.org/**
- **XML::Parser** – modul Perl (bazat pe Expat)
- **xml_*()** – functii PHP

Interfata SAX - exemplu

```
<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="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</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>
```

Exemplu SAX - Java org.xml.sax

```
//clasa de tratare a evenimentelor de procesare SAX
class BooksSAX extends DefaultHandler {
  // stiva de elemente
  private Stack<String> stiva = new Stack<String> ();
  // metoda de tratare a evenimentului 'inceput de tag'
  public void startElement (String uri, String local, String qName,
                            Attributes atts) throws SAXException {
    …
  }
  // metoda de tratare a evenimentului 'final de tag'
  public void endElement (String uri, String local, String qName)
                               throws SAXException {
    …
  }
  // metoda de tratare a evenimentului 'continut text'
  public void characters (char buf [], int offset, int length)
                               throws SAXException {
    …
  }
}
```

Exemplu SAX - Java org.xml.sax

```
// metoda de tratare a evenimentului 'inceput de tag'

public void startElement (String uri, String local, String qName,
                          Attributes atts) throws SAXException {

    stiva.push (new String(qName)); // introducem in stiva

    if (qName.equals ("book")) { // afisam categoria
        String categ;
        categ = atts.getValue ("category");
        System.out.print(categ + ": ");
    }
}
```

Exemplu SAX - Java org.xml.sax

```
// metoda de tratare a evenimentului 'final de tag‘

  public void endElement (String uri, String local, String qName)
                                  throws SAXException {

    if (qName.equals ("book"))
        System.out.println();
    stiva.pop (); // eliminam elementul din stiva
  }
```

Exemplu SAX - Java org.xml.sax

```
// metoda de tratare a evenimentului 'continut text'

  public void characters (char buf [], int offset, int length)
                                  throws SAXException {
    Object top = stiva.peek ();  // preluam virful stivei
    if (!top.equals ("title"))   // nu e element 'title'
      return;
    // afisam continutul text
    for (int i = 0; i < length; i++)
      System.out.print (buf[offset + i]);
  }
}
```

Exemplu SAX - Java org.xml.sax

```
// clasa de procesare SAX
public class SAX {
  // argv[0] reprezinta URI-ul documentului XML
  public static void main (String argv []) {
    XMLReader    prod; // producatorul SAX
    BooksSAX  cons; // consumatorul SAX
    …
    // instantiem procesorul SAX
      prod = XMLReaderFactory.createXMLReader ();
    …
    // consumam toate evenimentele SAX
      cons = new BooksSAX ();
    // stabilim maniera de tratare a continutului
      prod.setContentHandler (cons);
    // stabilim maniera de raportare a erorilor
      prod.setErrorHandler (cons);
    …
    // startam procesarea SAX (producerea de evenimente)
      prod.parse (argv [0]);
    …
}}
```

Exemplu SAX - Java `org.xml.sax`

```
> java SAX books.xml

COOKING: Everyday Italian
CHILDREN: Harry Potter
WEB: Learning XML
```

DOM vs SAX

Cind trebuie folosit SAX?

- Procesarea unor documente de mari dimensiuni
- Necesitatea abandonarii procesarii (procesorul SAX poate fi oprit oricand)
- Extragerea unor informatii de mici dimensiuni
- Crearea unei structuri noi de document XML
- Utilizarea in contextul unor resurse computationale reduse (memorie scazuta, largime de banda ingusta,...)

DOM vs SAX

Cind trebuie folosit DOM?

- Accesul direct la datele dintr-un document XML
- Cautari complexe
- Necesitatea efectuarii de transformari XSL
- Filtrarea complexa a datelor via XPath
- Necesitatea modificarii si salvarii documentelor XML
- In contextul procesarii XML direct in cadrul navigatorului

DOM vs SAX

- DOM necesita incarcarea completa a documentului XML in vederea procesarii ca arbore
- SAX necesita pentru procesare existenta unor fragmente reduse din document, efectuindu-se o prelucrare liniara (sir de evenimente)
- SAX poate fi utilizat pentru generarea de arbori DOM; invers, arborii DOM pot fi traversati pentru a se emite evenimente SAX
- In cazul unor structuri XML sofisticate, modul de procesare SAX poate fi inadecvat
- Unele implementari SAX ofera suport pentru validari si transformari
- Uzual, se folosesc ambele API-uri

XML vs browser

- **XML Data Islands** (deprecated)
  - date XML incorporate in pagina HTML
  - doar in Internet Explorer
- ex:

```
<html>
<body>

<xml id="cdcat" src="cd_catalog.xml"></xml>

<table border="1" datasrc="#cdcat">
  <tr>
    <td><span datafld="ARTIST"></span></td>
    <td><span datafld="TITLE"></span></td>
  </tr>
</table>

</body>
</html>
```

XML vs browser

- IE behaviors
  - a way to add behaviors to XML (or HTML) elements with the use of CSS styles.
  - doar in Internet Explorer
- ex.:

```
<html>
<head>
<style type="text/css">
 h1 { behavior: url(behave.htc) }
</style>
</head>
<body>

<h1>Mouse over me!!!</h1>

</body>
</html>
```

XML (or HTML) elements with the use of CSS

behave.htc:

```
<attach for="element" event="onmouseover" handler="hig_lite" />
<attach for="element" event="onmouseout" handler="low_lite" />

<script type="text/javascript">
function hig_lite()
{
element.style.color='red';
}

function low_lite()
{
element.style.color='blue';
}
</script>
```

XML vs browser

- diferente in parsarea DOM
  - modul in care sunt tratate spatiile albe si liniile noi

```
<book>CR/LF
SPSP<title>Everyday Italian</title>CR/LF
SPSP<author>Giada De Laurentiis</author>CR/LF
SPSP<year>2005</year>CR/LF
SPSP<price>30.00</price>CR/LF
</book>
```

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.documentElement.childNodes;
document.write("Number of child nodes: " + x.length);
```

```
IE<9: 4
Alte browsere: 9
```

XML vs browser

- E4X = JavaScript for XML (deprecated)
  - o noua extensie a JavaScript
  - standardizat in 2004 (ECMA-357)
  - adauga in JavaScript suport direct pt. XML
  - XML = obiect JavaScript

```
var x = new XML()

var y = new Date()

var z = new Array()
```

  - usurinta in parsarea si manipularea unui document XML
  - doar in Firefox (deocamdata..)

```
<order>
    <date>2005-08-01</date>
    <customer>
       <firstname>John</firstname>
       <lastname>Johnson</lastname>
    </customer>
    <item>
       <name>Lamp</name>
       <qty>5</qty>
       <price>155.00</price>
    </item>
</order>
```

d)

- standa
- adaug
- XML =
- usurin
  docum
- doar i

```
var order = new XML(txt);

//calculare pret
var total=order.item.qty * order.item.price;

//adaugare item nou
order.item+=
<item>
<name>Chair</name>
<qty>10</qty>
<price>36.00</price>
</item>;

//calculare pret total
var price=0;
for each (i in order.item)
{
  price+= i.qty*i.price;
}
```

AJAX

AJAX – Asynchronous JavaScript and XML

Context:

- aplicatii Web ce ofera o interactiune bogata cu utilizatorul

AJAX

- NU este un limbaj de programare
- modalitate de utilizare a JavaScript
- o modalitate de a downloada date de la un server fara reincarcarea paginii
- permite prezentarea dinamica a datelor sau actualizarea paginii fara a incomoda desfasurarea actiunilor utilizatorului
- permite crearea de situri web user-friendly

AJAX

- Aplicatie web bogata: sit web care imita experienta unei aplicatii desktop
  - interactiune continua a utilizatorului
- Applicatii web de la Google:
  - Gmail, Google Maps, Google Docs and Spreadsheets
- Aplicatiile web pot folosi **Ajax** pentru a combate:
  - reactia lenta a interfetelor cu utilizatorul
  - lipsa unei interactiuni user-friendly
  - natura neplacuta a sablonului "**`click-wait-refresh`**"

AJAX

Caracterizare:

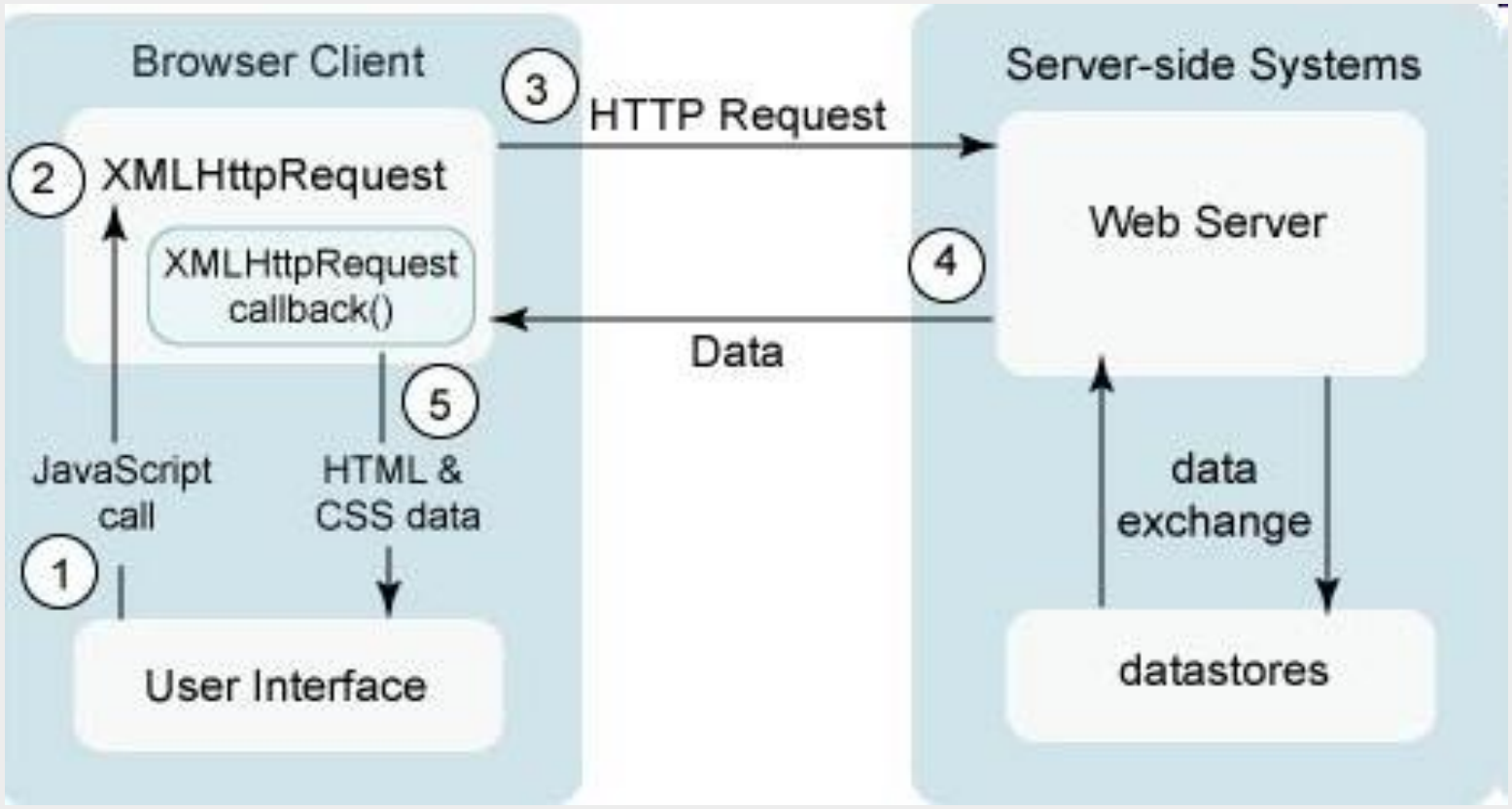
- reprezinta o suita de tehnologii deschise, incorporand:
  - limbaje standardizate de prezentare a datelor ((X)HTML, CSS)
  - redare si interactiune via DOM
  - interschimb si manipulare de date prin XML si/sau XSLT
  - procesare prin limbajul ECMAScript/JavaScript (ECMA)

AJAX

- Componenta de baza: obiectul **XMLHttpRequest** pus la dispozitie de browser-ul Web:
  - permite realizarea de cereri HTTP (ex. GET sau POST) dintr-un program ruland la nivel de client (*browser*) spre o aplicatie server, ***asincron***
  - continutul fisierului transferat poate fi accesat in pagina web prin intermediul DOM
  - rezultat:
    - pagina web a utilizatorului este actualizata dinamic fara a fi reincarcata in intregime
  - Implementarea depinde de navigator:
    - Firefox – obiect nativ
    - Internet Explorer 5,6 - instantiat drept obiect ActiveX
    - Safari 1.2+, Opera 8.0+

AJAX

Cerere AJAX tipica:

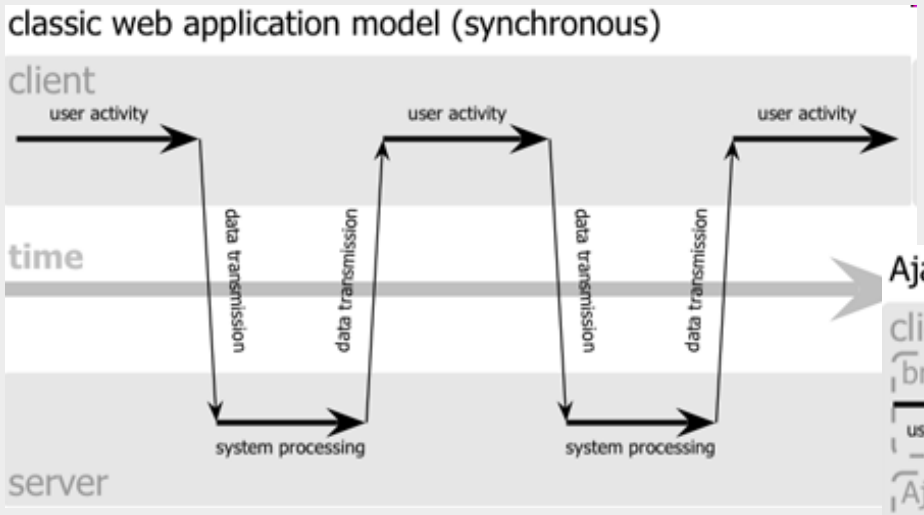


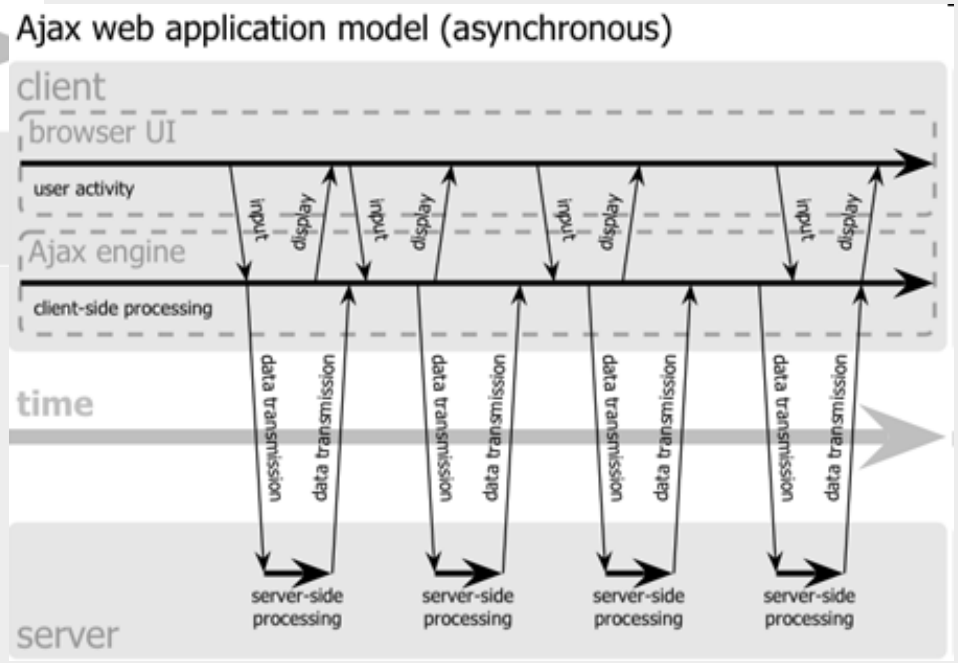
AJAX

Cerere AJAX tipica:

1. Utilizatorul activeaza un control, invocand o functie ce trateaza evenimentul respectiv
2. Codul JS al functiei creeaza un obiect XMLHttpRequest
3. Obiectul XMLHttpRequest cere un document de la un server web
   - Serverul obtine datele corespunzatoare si le trimite
4. XMLHttpRequest genereaza un eveniment pentru a anunta sosirea datelor cerute
   - Se poate atasa o functie de tratare a acestui eveniment (handler) pentru a notifica sosirea datelor
5. Functia handler proceseaza datele si le afiseaza

AJAX


classic web application model (synchronous)
client
user activity
user activity
user activity
data transmission
data transmission
data transmission
data transmission
time
system processing
system processing
server


Ajax web application model (asynchronous)
client
browser UI
user activity
input
display
input
display
input
display
input
display
Ajax engine
client-side processing
data transmission
data transmission
data transmission
data transmission
data transmission
data transmission
data transmission
data transmission
time
server-side processing
server-side processing
server-side processing
server-side processing
server

AJAX

Obiectul XMLHttpRequest

- Metode:
  - `abort`, `getAllResponseHeaders`, `getResponseHeader`, **`open`**, **`send`**, `setRequestHeader`
- Proprietati:
  - **`onreadystatechange`**, **`readyState`**, **`responseText`**, **`responseXML`**, **`status`**, `statusText`

AJAX

Obiectul XMLHttpRequest

- Metode:

open(Method, URL, Asynchronous, UserName, Password)

Method – metoda HTTP (GET, POST, HEAD, PUT, DELETE, OPTIONS)

URL – adresa resursei (in acelasi domeniu cu sursa)

Asynchronous – boolean (false blocheaza executia scriptului pana la finalizarea cererii)

Username, Password – parametri de autentificare

send(Data)

Data – orice tip (disponibil limbajului de scripting) ce poate fi serializat

AJAX

Utilizare:

```
//cod intr-o functie ce trateaza un eveniment
//provenit de la un control onscreen

var ajax = new XMLHttpRequest();
ajax.onreadystatechange = function;
ajax.open("GET", url, true);
ajax.send(null);
```

- se ataseaza o functie de tratare a evenimentului ***onreadystatechange***
- functia va fi apelata la o schimbare a starii cererii, ex. la terminare
- ***function*** contine codul ce trebuie executat la incheierea cererii
- ***url*** reprezinta resursa ce se doreste a fi adusa de pe server

AJAX

Proprietatea `readyState`

- mentine status-ul cererii
- valori posibile:
  - 0 – cerere neinitializata
  - 1 – set-up (dupa invocarea cu succes a metodei open)
  - 2 – sent (dupa invocarea cu succes a metodei send si primirea headerelor HTTP ale raspunsului)
  - 3 – in progress (la inceperea incarcarii continutului raspunsului HTTP)
  - **4 – complete** (la terminarea incarcarii continutului raspunsului HTTP)

AJAX

Utilizare:

```
var ajax = new XMLHttpRequest();
ajax.onreadystatechange = function() {
if (ajax.readyState == 4) {
      if (ajax.status == 200) {
            do something with ajax.responseText;
      }
      else {
            code to handle the error;
      }
};
ajax.open("GET", url, true);
ajax.send(null);
```

AJAX

Rapunsul HTTP:

- **`responseXML`**
  - contine un obiect *DOM document* daca
    - raspunsul serverului este un XML valid si
    - headerul Content-Type setat de server este un Internet media type pt. XML
- **`responseText`**
  - contine raspunsul serverului in format text, indiferent daca acesta este interpretat ca XML sau nu

AJAX

Restrictii de securitate:

- navigatoarele Web impun anumite restrictii de executie a programelor Javascript
- same origin policy:
  - nu pot fi realizate cereri HTTP via obiectul **`XMLHttpRequest`** decat asupra server-ului care gazduieste documentul din care provin cererile

    www.exemplu.com/a/b/c.html

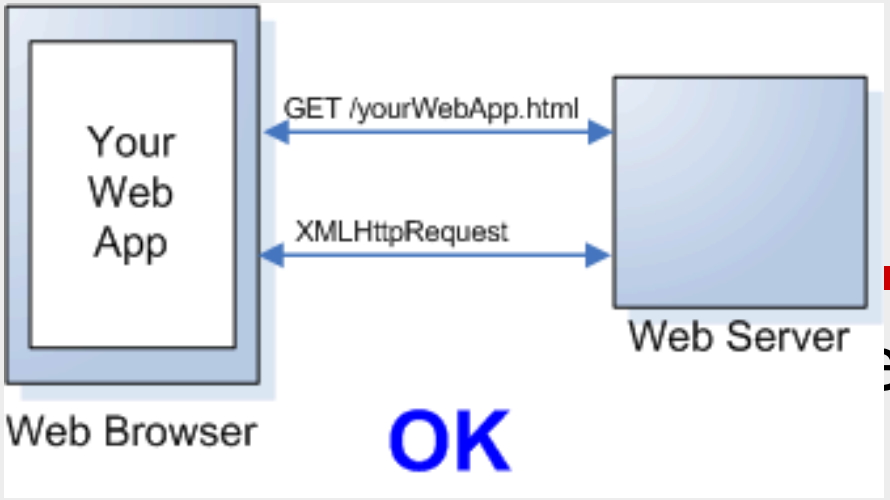
    poate transfera fisiere doar de pe

    www.exemplu.com

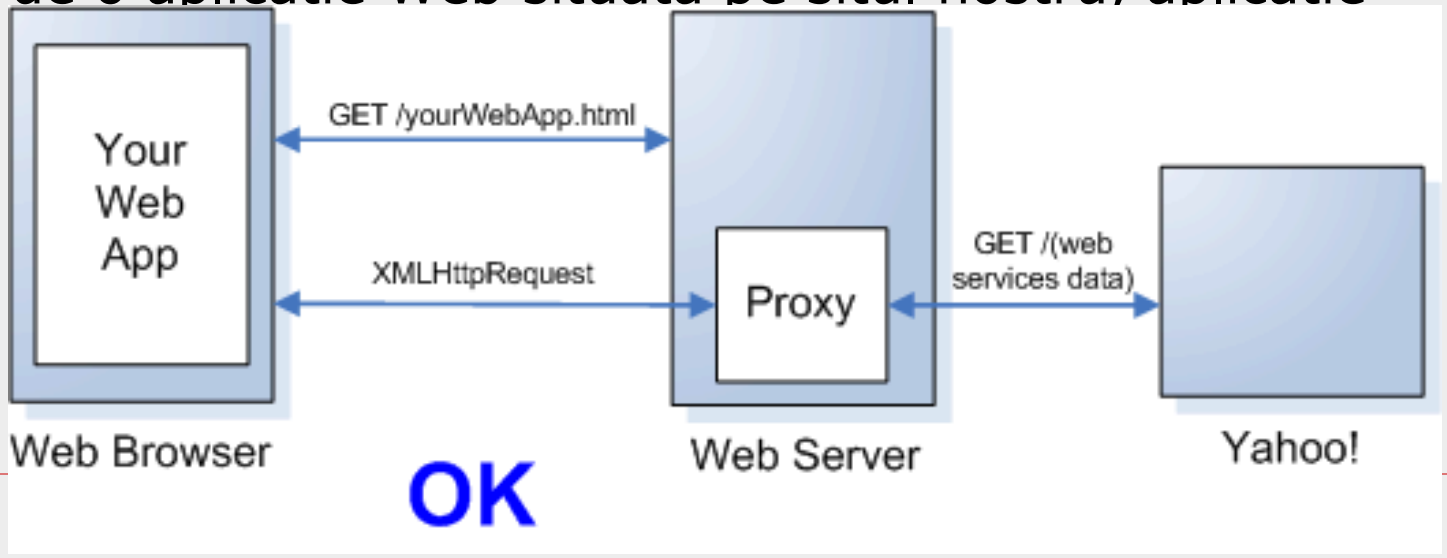
AJAX

Restrictii de securitate: solutii (1)

- folosirea unor aplicatii intermediare (*application proxies*)
  - cererile efectuate prin intermediul AJAX vor fi preluate de o aplicatie Web situata pe situl nostru, aplicatie care va realiza invocarea serviciului Web dorit si va returna rezultatele documentului care a realizat acea cerere



- folosirea unor aplicatii inter
  proxies)
  - cererile efectuate prin int
    de o aplicatie Web situata pe situl nostru, aplicatie



AJAX

Restrictii de securitate: solutii (2)

- ajustarea configuratiei serverului Web
  - orice cerere provenita via AJAX sa fie dirijata, in mod transparent, spre aplicatia dorita
  - Apache proxy
  - Apache `mod_rewrite` utilizand directiva `passthru`
    - http://www.xml.com/lpt/a/1627
  - IIS 7+ - addon Application Request Routing
    - http://thethoughtfulcoder.com/2009/09/09/Cross-Domain-AJAX-Call-Using-IIS-7-And-Microsoft-URL-Rewrite

AJAX

Restrictii de securitate: solutii (3)

- recurgerea la solutii alternative de transfer asincron
  - utilizarea JSON si a invocarii la cerere a programelor JavaScript via elementul XHTML **`<script>`**
  - nu necesita **`XMLHttpRequest`**

AJAX

Restrictii de securitate: solutii (3)

```
<html>
<head>
<title>How Many Pictures Of Madonna Do We Have?</title>
</head>
</body>
<script type="text/javascript">
function ws_results(obj) {
        alert(obj.ResultSet.totalResultsAvailable);
}
</script>
<script type="text/javascript"
        src="http://search.yahooapis.com/ImageSearchService/V1/
        imageSearch?appid=YahooDemo&query=Madonna&
        output=json&callback=ws_results">
</script>
<body></body>
</html>
```

AJAX

	XmlHttpRequest	Dynamic script Tag
Cross-browser compatible?	No	Yes
Cross-domain browser security enforced?	Yes	No
Can receive HTTP status codes?	Yes	No (fails on any HTTP status other than 200)
Supports HTTP GET and POST?	Yes	No (GET only)
Can send/receive HTTP headers?	Yes	No
Can receive XML?	Yes	Yes (but only embedded in a JavaScript statement)
Can receive JSON?	Yes	Yes (but only embedded in a JavaScript statement)
Offers synchronous and asynchronous calls?	Yes	No (asynchronous only)

AJAX

- Restrictii de securitate: solutii (4)
- CORS - Cross-Origin Resource Sharing
  - W3C Working Draft (iulie 2010)
  - defineste un mecanism de activare la nivelul clientului a cererilor cross-origin
  - implementari:
    - Firefox 3.5+
    - Safari 4+
    - IE 8 (obiectul XDomainRequest)

AJAX - CORS

- specificatie ce consta in interschimbarea de headere intre client si server
- permite efectuarea de cereri cross-site
- adauga noi headere HTTP ce permit serverelor sa trimita resurse unor domenii specificate
  - https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS

AJAX - CORS

Simple requests – allow:

- **Methods**: GET, HEAD, POST

- **Manually-set headers**: Accept, Accept-Language, Content-Language, Content-Type

- **Content-Type values**: application/x-www-form-urlencoded, multipart/form-data, text/plain

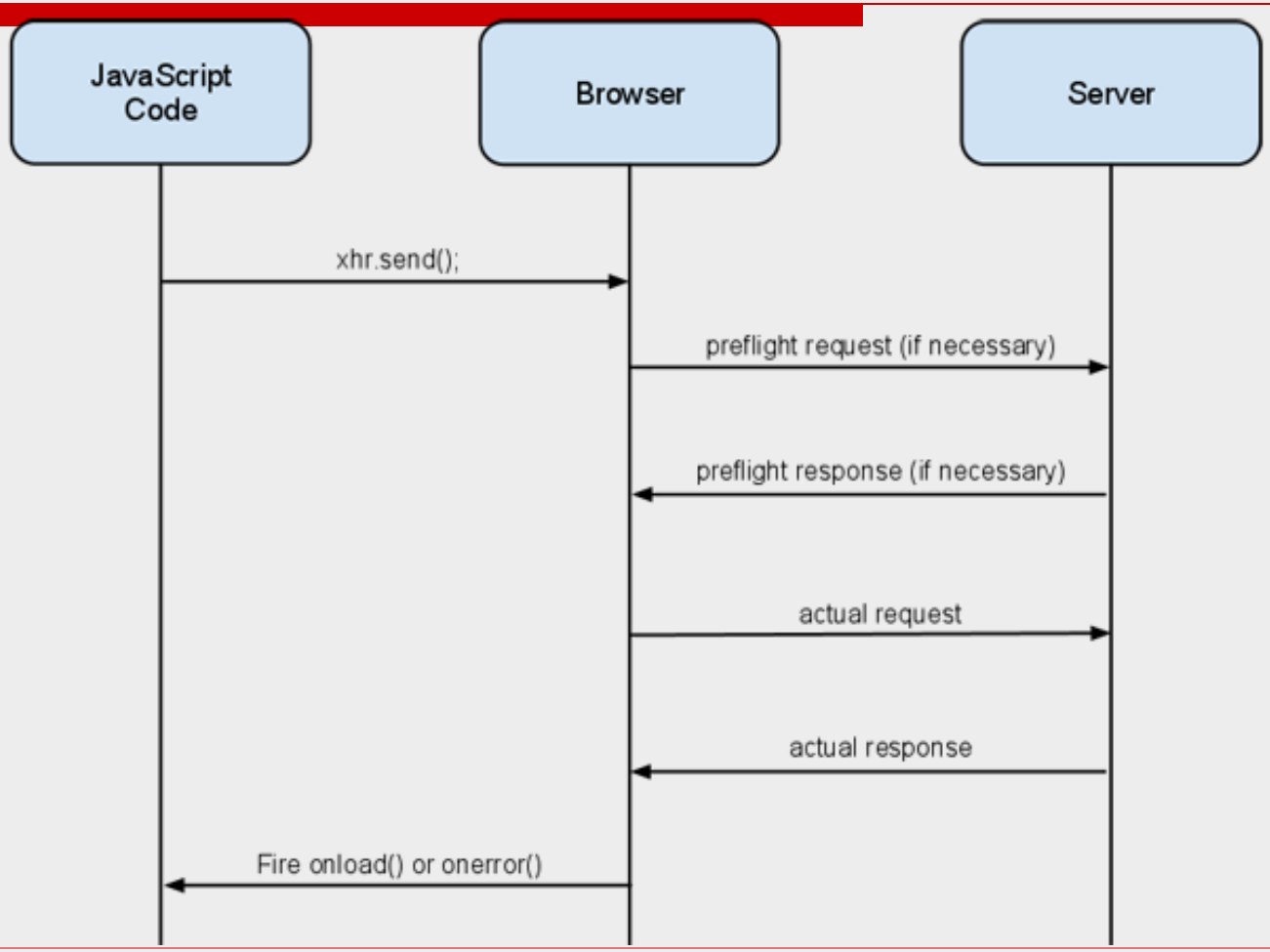
AJAX - CORS

- browser-ul trimite header-ul ORIGIN
  - schema (http:// sau https://)
  - domeniul paginii care face cererea
- serverul trebuie sa trimita inapoi headerul corect
  - Access-Control-Allow-Origin pt originea in cauza sau “*” pt toate domeniile daca resursa este publica

AJAX - CORS

- pt. cererile cu metode HTTP ce pot cauza efecte secundare asupra datelor utilizator (pt. alte met. decat GET sau pt. POST cu anumite MIME type-uri)
  - browser-ul tb. sa faca o cerere "preflighted":
    - se solicita metodele suportate de la server (cu HTTP OPTIONS)
    - la aprobarea serverului se trimite cererea cu metoda HTTP efectiva
- serverele pot notifica clientii daca este necesara transmiterea de "credentials" (cookies sau HTTP Authentication) impreuna cu cerearea

AJAX – CORS flow



AJAX - CORS

- **cereri simple**

Ex: http://foo.example efectueaza o cerere AJAX/CORS catre http://bar.other

CLIENT:

```
GET /publicNotaries/ HTTP/1.1
Referer: http://foo.example/notary-mashup/
Origin: http://foo.example
```

SERVER:

```
Access-Control-Allow-Origin: http://foo.example
Content-Type: application/xml
......
```

AJAX – CORS

- cereri simpl

```
var url = "http://bar.other/publicNotaries/"
if(XMLHttpRequest)
{
  var request = new XMLHttpRequest();
  if("withCredentials" in request)  {  // Firefox 3.5 and Safari 4
   request.open('GET', url, true);
   request.onreadystatechange = handler;
   request.send();
  }
  else if (XDomainRequest)  {  // IE8
   var xdr = new XDomainRequest();
   xdr.open("get", url);
   xdr.send();
   // handle XDR responses -- not shown here :-)
  }
 // This version of XHR does not support CORS
 // Handle accordingly
}
```

AJAX - CORS

- **cereri “preflighted”**
  - folosesc alte metode decat POST sau GET
  - utilizeaza headere custom
  - corpul cererii are alt MIME type decat text/plain
- NU este suportata de obiectul XDomainRequest din IE8
- mecanismul de “preflight”
  - in seama browser-ului
- Ex: http://foo.example efectueaza o cerere catre http://bar.other

CLIENT:

```
OPTIONS /resources/post-here/ HTTP/1.1
Origin: http://foo.example
Access-Control-Request-Method: POST
Access-Control-Request-Headers: X-PINGOTHER
```

SERVER:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://foo.example
Access-Control-Allow-Methods: POST, GET, OPTIONS
Access-Control-Allow-Headers: X-PINGOTHER
Access-Control-Max-Age: 1728000
......
```

```
var invocation = new XMLHttpRe
var url = 'http://bar.other/re
var body = 'Arun';
function callOtherDomain(){
//mecanism detectie capabilita
//…
if(invocation)
{
    invocation.open('POST', url, true);
    invocation.setRequestHeader('X-PINGOTHER', 'pingpong');
    invocation.setRequestHeader('Content-Type', 'application/xml');
    invocation.onreadystatechange = handler;
    invocation.send(body);
}
```

- mecanismul de "preflight"
  - in seama browser-ului
- Ex: http://foo.example efectueaza o cerere catre http://bar.other

AJAX - CORS

- **cereri cu credentials**
  - cookies
  - HTTP Auth
- se seteaza proprietatea withCredentials a obiectului XMLHttpRequest
  - NU este suportata in IE8
- Ex.: http://foo.example face o cerere catre http://bar.other si doreste sa transmita un Cookie
  - comportament asemanator cu cererile simple

AJAX - CORS

```
var request = new XMLHttpRequest();
var url = 'http://bar.other/resources/credentialed-content/';
function callOtherDomain(){
  if(request)
  {
   request.open('GET', url, true);
   request.withCredentials = "true";
   request.onreadystatechange = handler;
   request.send();
  }
```

- Ex.: http://foo.example face o cerere catre http://bar.other si doreste sa transmita un Cookie
  - comportament asemanator cu cererile simple

AJAX - CORS

- **Enabling CORS**
  - se adauga headerul HTTP Access-Control-Allow-Origin: "*"
- Apache
  - in fisierul .htaccess
    - Header set Access-Control-Allow-Origin "*"
- PHP
  - <?php header("Access-Control-Allow-Origin: *");…

AJAX - CORS

Enabling CORS

- se adauga headerul HTTP Access-Control-Allow-Origin: "*"

IIS 6

- Open *Internet Information Service (IIS)* Manager
- Right click the site you want to enable CORS for and go to *Properties*
- Change to the *HTTP Headers* tab
- In the *Custom HTTP headers* section, click *Add*
- Enter Access-Control-Allow-Origin as the header name
- Enter * as the header value
- Click *Ok* twice

AJAX - CORS

- **Enabling CORS**
  - se adauga headerul HTTP Access-Control-Allow-Origin: "*"
- IIS 7
  - in fisierul web.config:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.webServer>
    <httpProtocol>
      <customHeaders>
        <add name="Access-Control-Allow-Origin" value="*" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
</configuration>
```

AJAX - CORS

- **Enabling CORS**
  - se adauga headerul HTTP Access-Control-Allow-Origin: "*"
- ASP.NET
  - Response.AppendHeader("Access-Control-Allow-Origin", "*");