UNIVERSITÉ DE NANTES

Programmer avec XML

L'API SAX

E. Desmontils (Emmanuel.Desmontils@univ-nantes.fr)

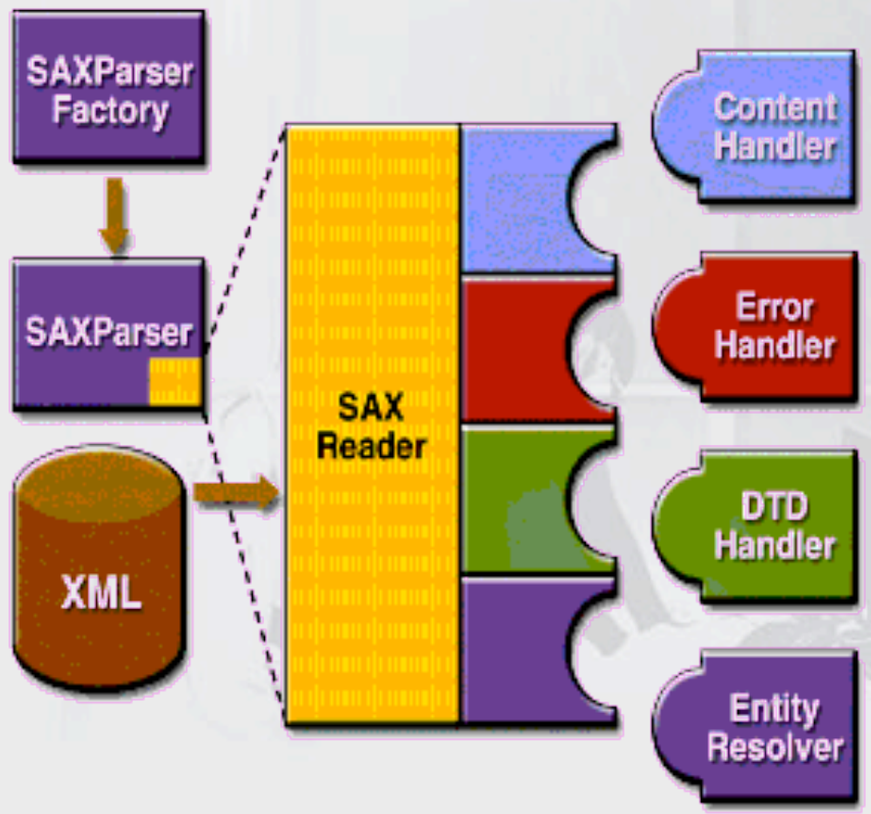
Université de Nantes

Faculté des sciences et des techniques

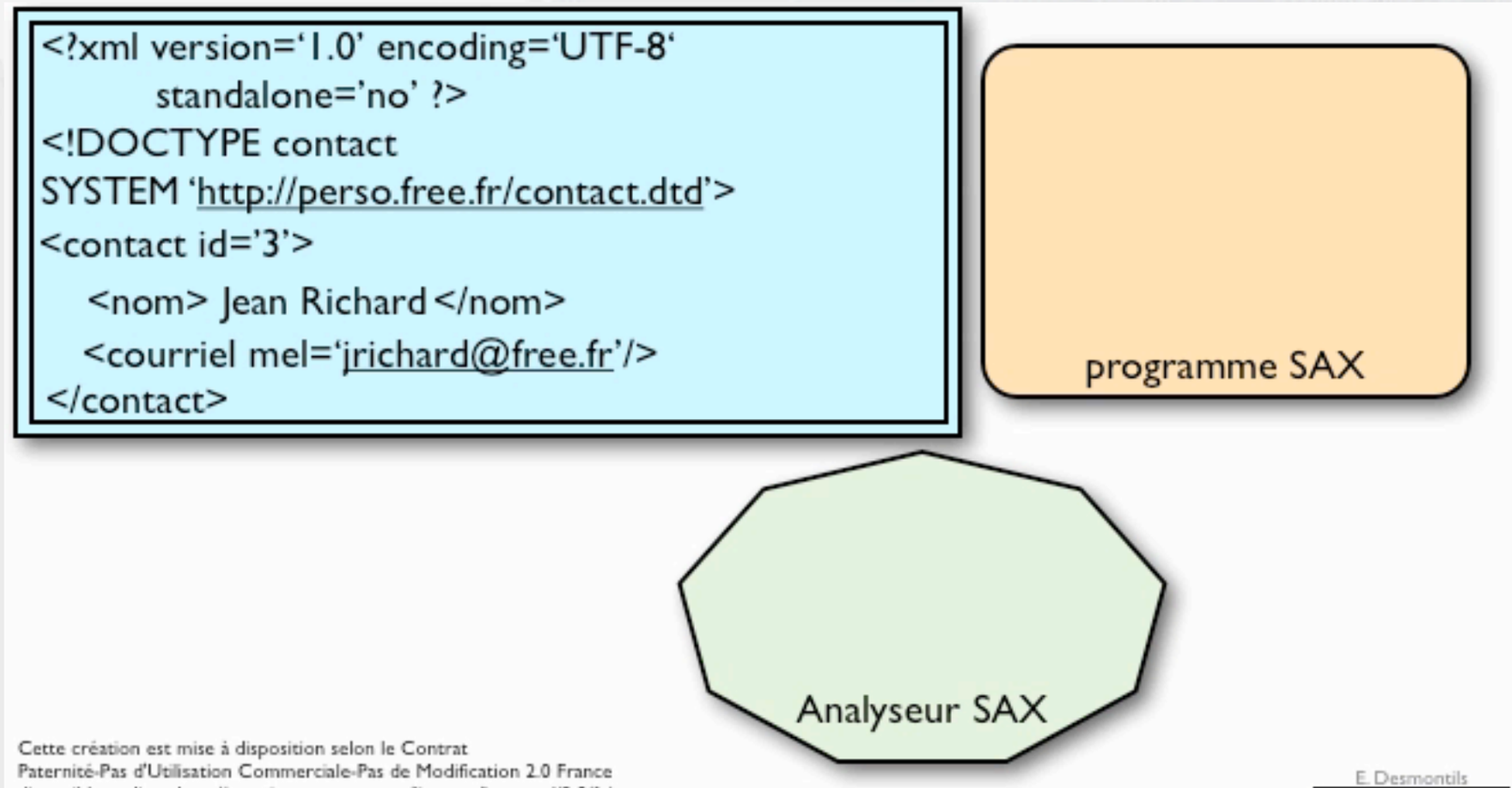
version 2.5

L'API Sax

- Simple API for XML
  - À l'origine, une API pour Java, mais actuellement C, C++, Perl, PHP...
- Méthode événementielle
- Traitement lié à l'analyse syntaxique du document XML
- L'analyse et le traitement ont lieu en même temps
- Événement : début ou fin d'un élément...

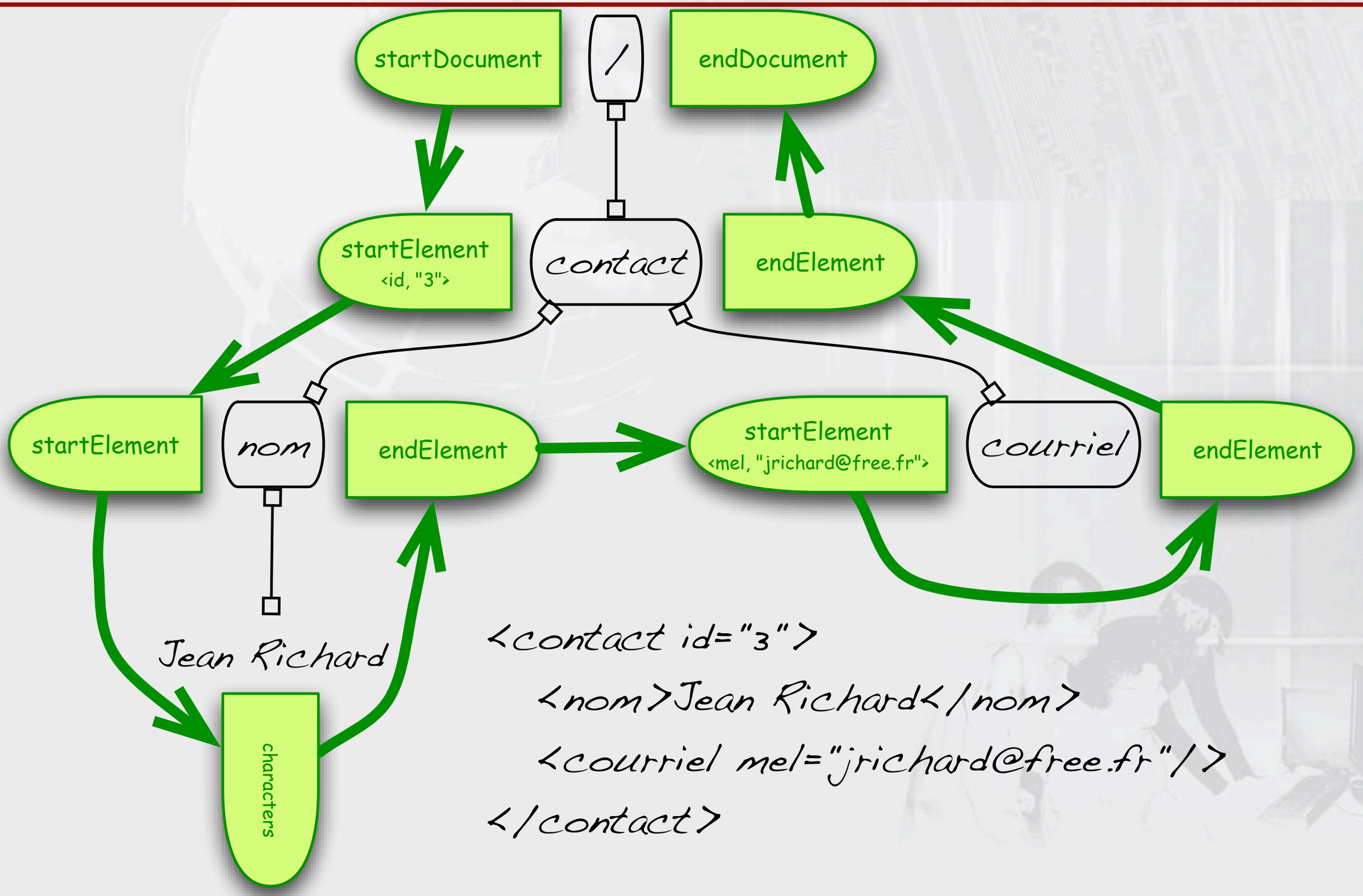


Flux d'événements SAX



E. Desmontils

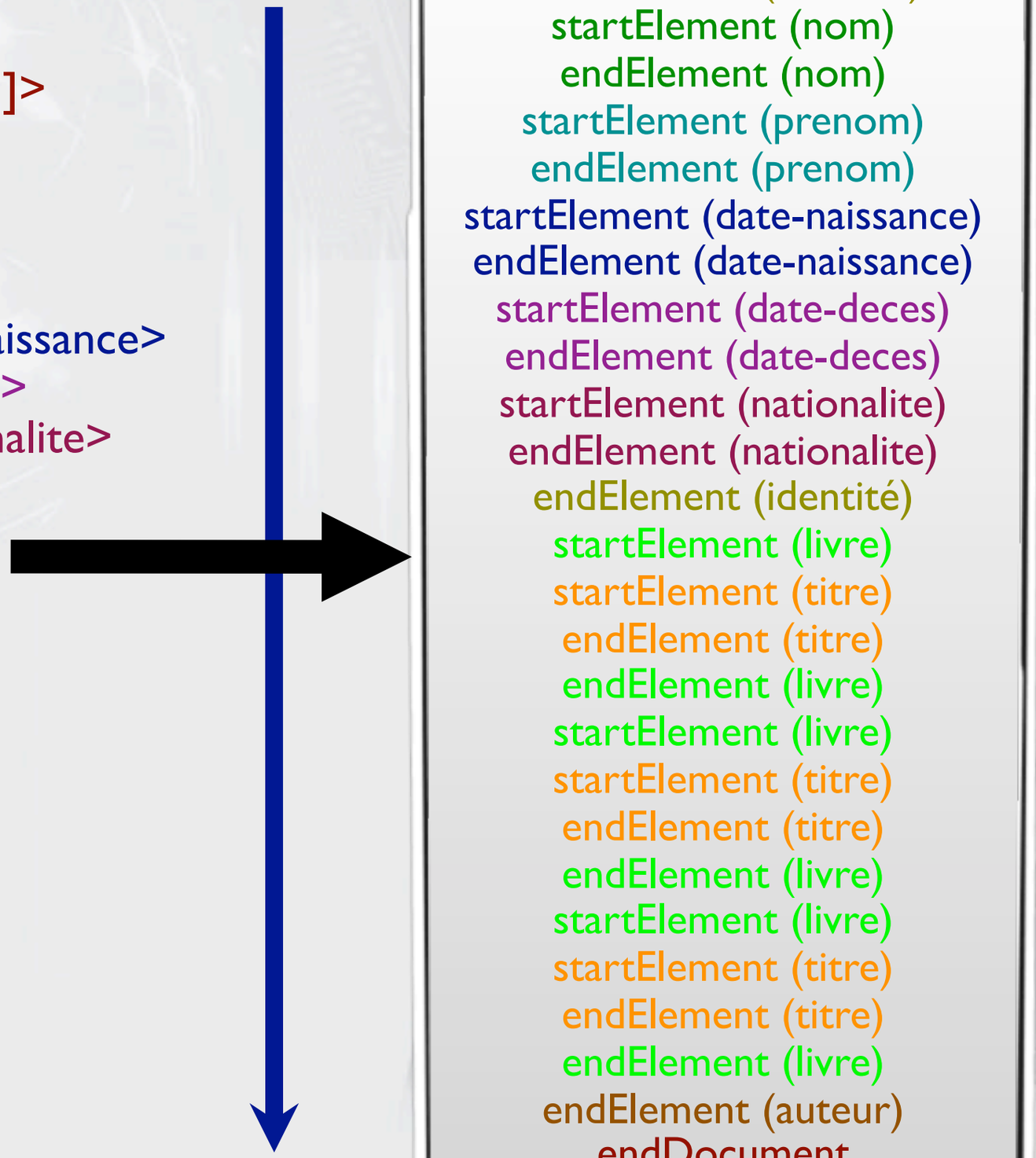
Flux d'événements SAX



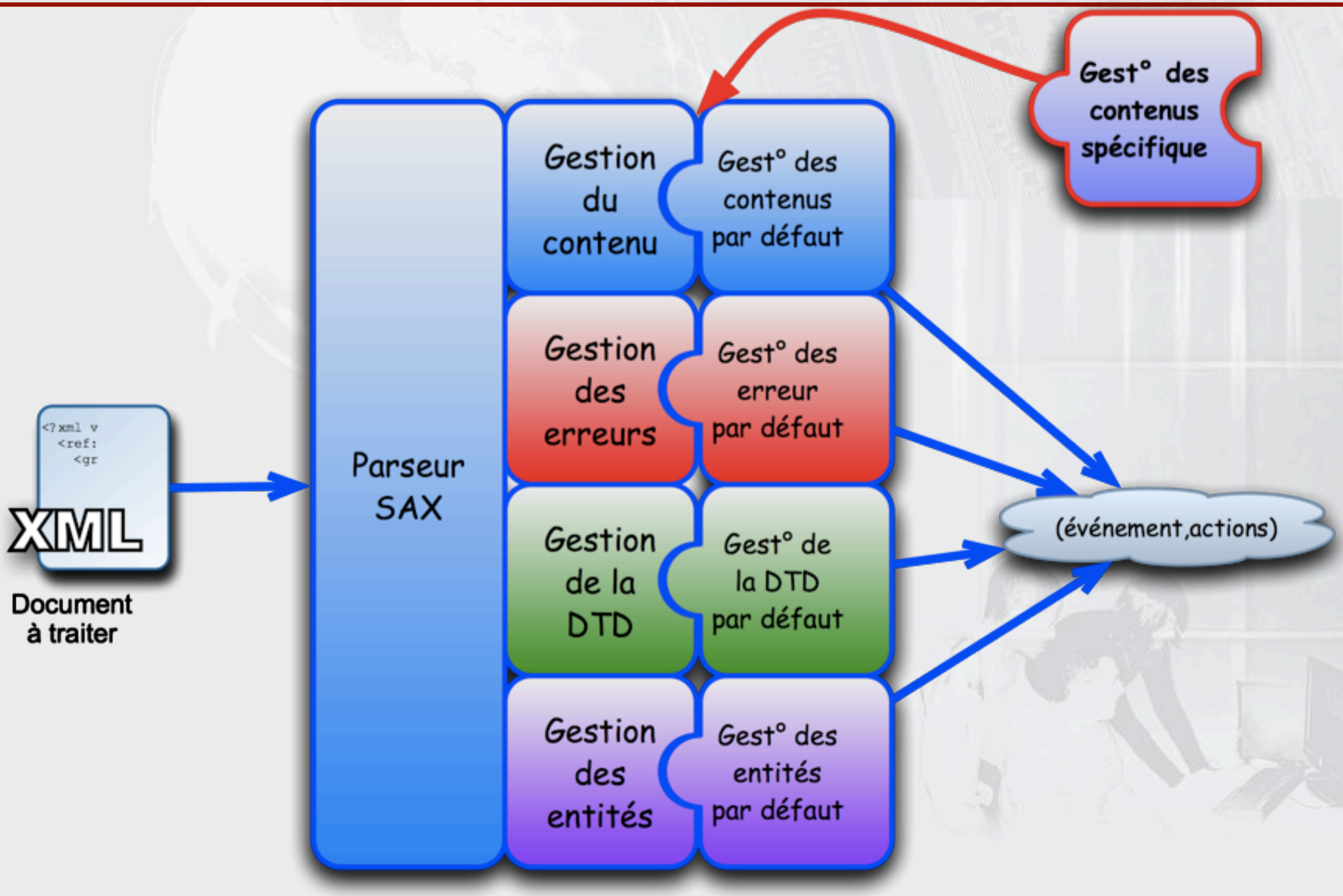

```
<contact id="3">
   <nom>Jean Richard</nom>
   <courriel mel="jrichard@free.fr"/>
</contact>
```

Flux d'événements SAX

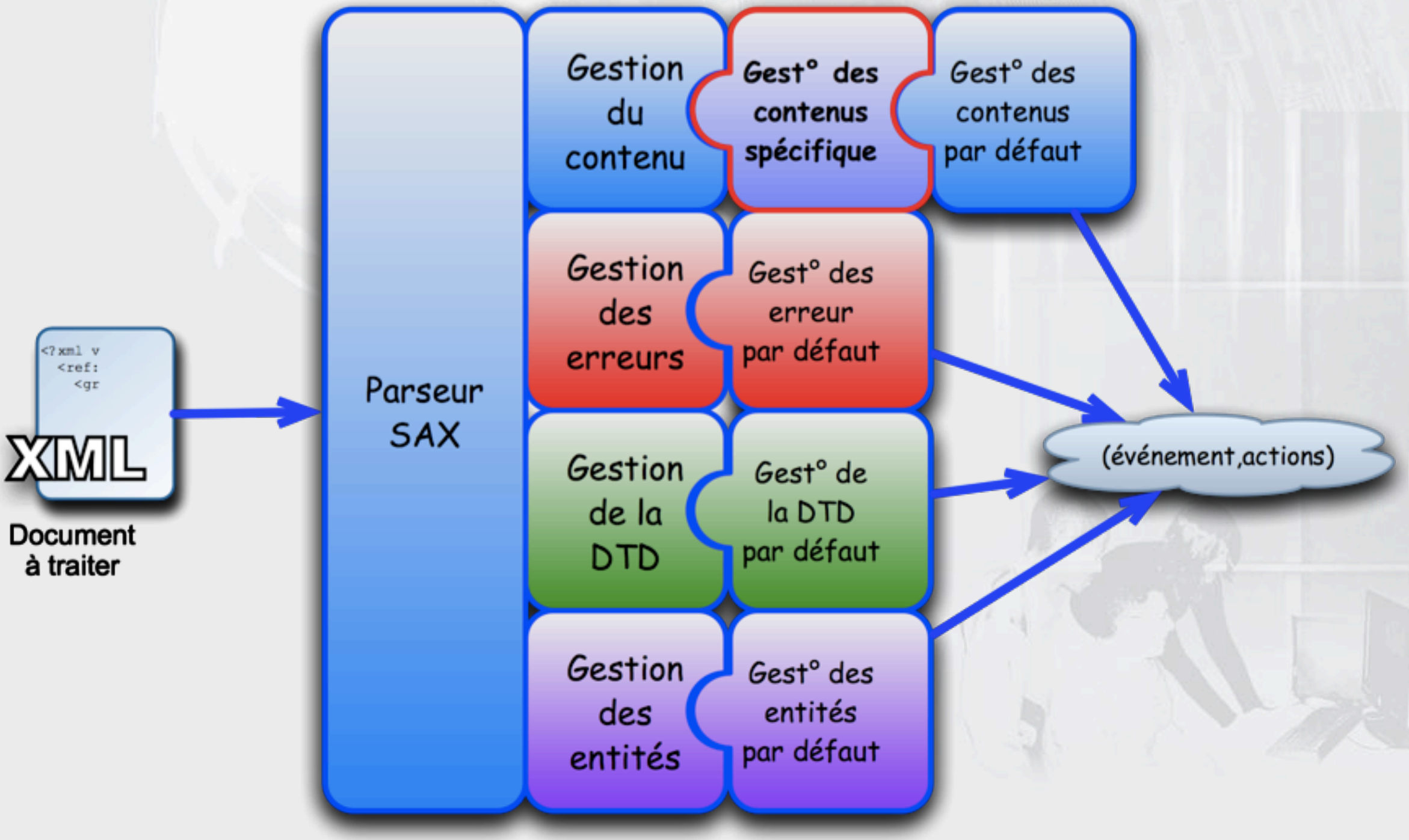
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE auteur [<!ENTITY jl "j'ai lu"> ]>
<auteur>
   <identité>
      <nom>Asimov</nom>
      <prenom>Isaac</prenom>
      <date-naissance>1920-01-02</date-naissance>
      <date-deces>1992-04-06</date-deces>
      <nationalite>Russe/Américain</nationalite>
   </identité>
  <!-- Liste des ouvrages -->
   <livre annee="1964" editeur="&jl;"
           reference="l542">
      <titre>Un défilé de robots</titre>
   </livre>
   <livre annee="1950" editeur="&jl;"
           reference="l453">
      <titre>I, robot</titre>
   </livre>
   <livre annee="2002" editeur="&jl;"
            reference="l2388">
      <titre>Le robot qui rêvait</titre>
   </livre>
</auteur>
```



Programmer en SAX : principe



Programmer en SAX : principe



Gestion des événements liés au document

Gest° des contenus par défaut

```
void startDocument() throws SAXException

void endDocument() throws SAXException

void startElement(String uri, String localName, String qName,  Attributes atts)
                  throws SAXException

void endElement(String uri, String localName, String qName)
                throws SAXException
```

- uri - L'URI de l'espace de noms (vide s'il n'y en a pas ou si la gestion des espaces de noms est désactivée)
- localName - le nom de l'élément (sans préfix) ou vide si la gestion des espaces de noms n'est pas activée.
- qName - le nom de l'élément avec le préfixe ou vide si la gestion des noms qualifés n'est pas activée.
- atts - les attributs de l'élément (objet Attributes).
- SAXException - une exception SAX quelconque.

Gestion des événements liés au document

void startDocur

void endDocum

void startEleme
thr

void endElemen
throv

- uri - L'URI de l'e
- localName - le r
- qName - le nom
- atts - les attribu
- SAXException -

Method Summary

La classe "org.xml.sax.Attributes"

int	**getIndex**(String qName) Look up the index of an attribute by XML qualified (prefixed) name.
int	**getIndex**(String uri, String localName) Look up the index of an attribute by Namespace name.
int	**getLength**() Return the number of attributes in the list.
String	**getLocalName**(int index) Look up an attribute's local name by index.
String	**getQName**(int index) Look up an attribute's XML qualified (prefixed) name by index.
String	**getType**(int index) Look up an attribute's type by index.
String	**getType**(String qName) Look up an attribute's type by XML qualified (prefixed) name.
String	**getType**(String uri, String localName) Look up an attribute's type by Namespace name.
String	**getURI**(int index) Look up an attribute's Namespace URI by index.
String	**getValue**(int index) Look up an attribute's value by index.
String	**getValue**(String qName) Look up an attribute's value by XML qualified (prefixed) name.
String	**getValue**(String uri, String localName) Look up an attribute's value by Namespace name.

Gestion des événements liés à des caractères ou à une instruction de traitement

Gest° des contenus par défaut

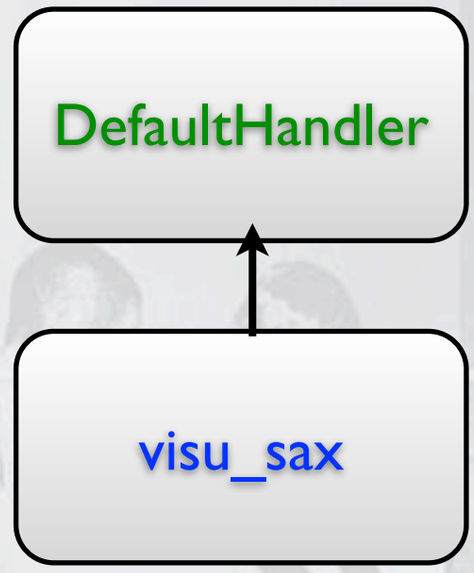
```
void characters(char[] ch, int start, int length)
                throws SAXException

void ignorableWhitespace(char[] ch, int start, int length)
                throws SAXException

void processingInstruction(String target, String data)
                throws SAXException
```

- ch - une série de caractères du document XML (#PCDATA)
- start - la position de départ dans la chaîne
- length - le nombre de caractères à lire dans la chaîne
- target - la cible de l'instruction de traitement
- data - les données de l'instruction de traitement.

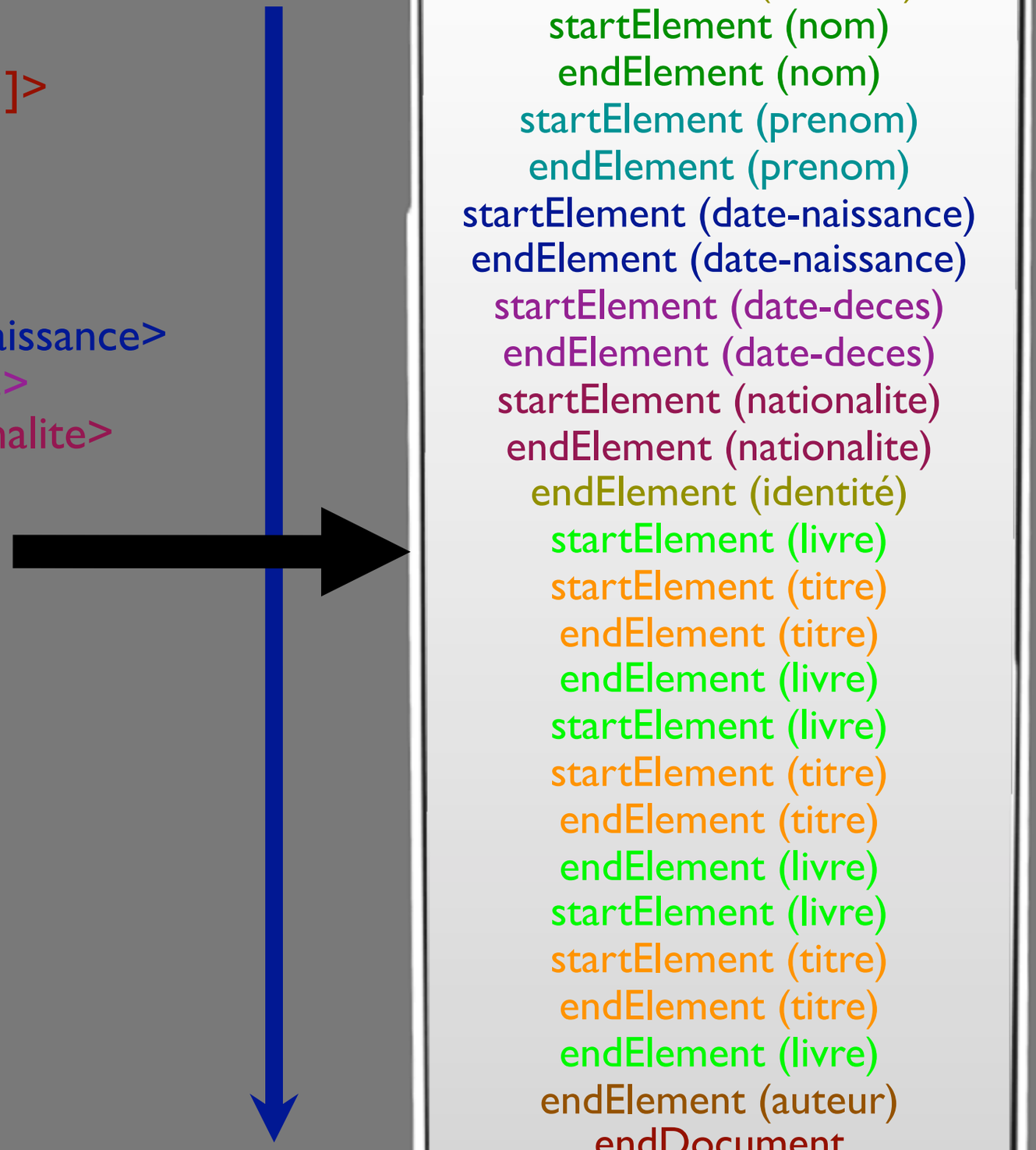
Un exemple de gestionnaire d'événements SAX


```
public class visu_sax extends DefaultHandler {
   public void startDocument() throws SAXException{
      System.out.println("startDocument"); }

   public void endDocument() throws SAXException{
      System.out.println("endDocument");}

   public void startElement(String namespaceURI, String sName,
                            String qName,  Attributes attrs)
   throws SAXException{
      System.out.println("startElement ("+qName+")");}

   public void endElement(String namespaceURI, String sName,
                            String qName)
   throws SAXException{
      System.out.println("endElement ("+qName+")");}

   public void processingInstruction(String target, String data)
   throws SAXException{
      System.out.println("processingInstruction ("+target+")");}
   ...
}
```

Gest° des contenus par défaut



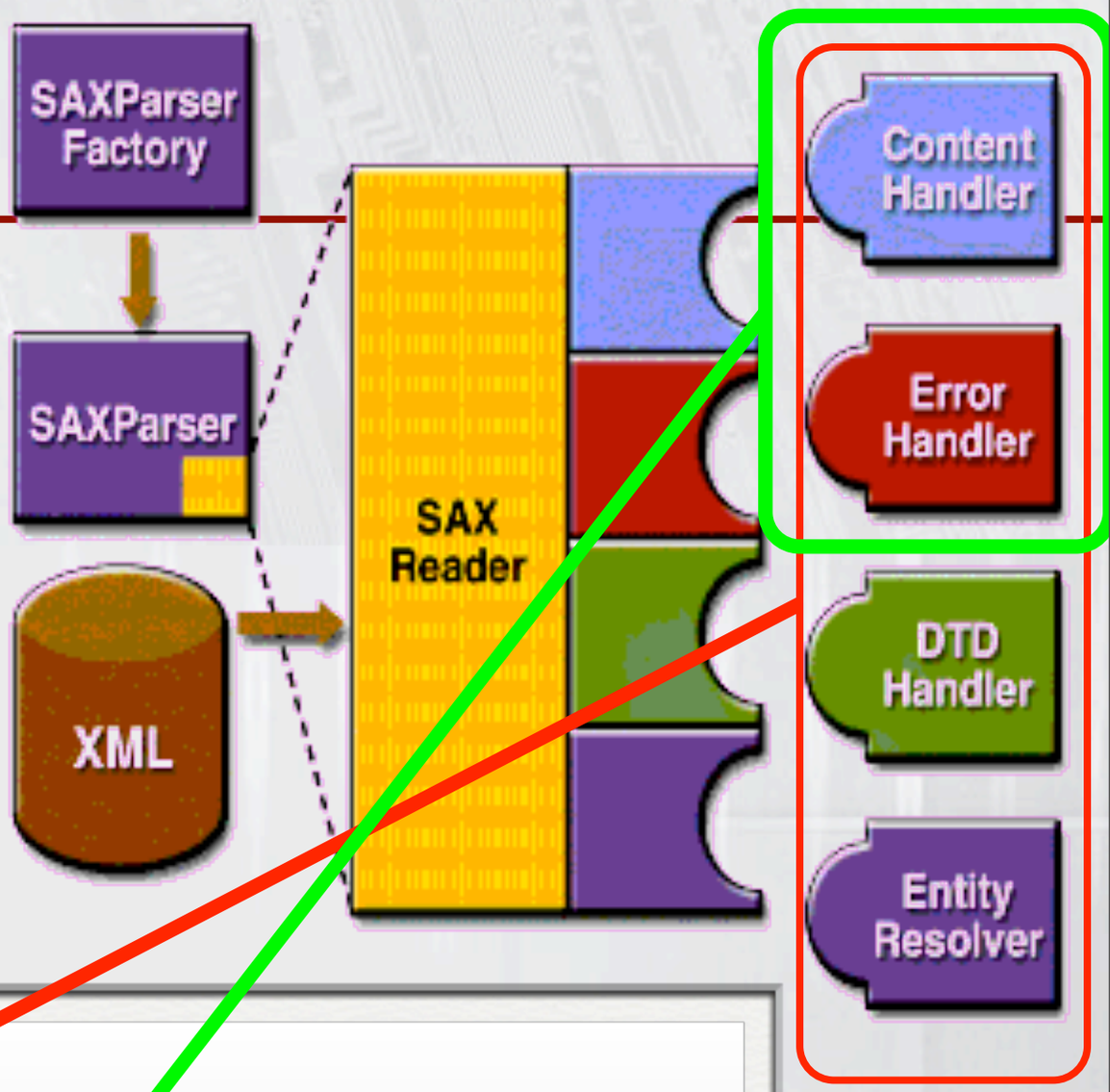
```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE auteur [<!ENTITY jl "j'ai lu"> ]>
<auteur>
   <identité>
      <nom>Asimov</nom>
      <prenom>Isaac</prenom>
      <date-naissance>1920-01-02</date-naissance>
      <date-deces>1992-04-06</date-deces>
      <nationalite>Russe/Américain</nationalite>
   </identité>
   <!-- Liste des ouvrages -->
   <livre annee="1964" editeur="&jl;"
          reference="1542">
      <titre>Un défilé de robots</titre>
   </livre>
   <livre annee="1950" editeur="&jl;"
          reference="1453">
      <titre>I, robot</titre>
   </livre>
   <livre annee="2002" editeur="&jl;"
          reference="12388">
      <titre>Le robot qui rêvait</titre>
   </livre>
</auteur>
```



startDocument
startElement (auteur)
startElement (identité)
startElement (nom)
endElement (nom)
startElement (prenom)
endElement (prenom)
startElement (date-naissance)
endElement (date-naissance)
startElement (date-deces)
endElement (date-deces)
startElement (nationalite)
endElement (nationalite)
endElement (identité)
startElement (livre)
startElement (titre)
endElement (titre)
endElement (livre)
startElement (livre)
startElement (titre)
endElement (titre)
endElement (livre)
startElement (livre)
startElement (titre)
endElement (titre)
endElement (livre)
endElement (auteur)
endDocument

Créer un parseur SAX en Java


```
...
import org.xml.sax.*;
import org.xml.sax.helpers.DefaultHandler;
import org.xml.sax.helpers.XMLReaderFactory;
import javax.xml.parsers.SAXParserFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.parsers.SAXParser;
...
```




```
...
  DefaultHandler handler = new visu_sax();
  try {
      XMLReader saxParser = XMLReaderFactory.createXMLReader();
      saxParser.setContentHandler(handler);
      saxParser.setErrorHandler(handler);
      saxParser.parse( argv[0] );
  } catch (Throwable t) {t.printStackTrace();}
...
```

Exemple (supplémentaire) de SAX en Java : Compter le nombre de livres depuis 1960

```
import java.io.*;
import org.xml.sax.*;
import javax.xml.parsers.SAXParserFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.parsers.SAXParser;
import org.xml.sax.helpers.DefaultHandler;

public class nbl_sax extends DefaultHandler {
        public int nb;
        public void startDocument() throws SAXException {nb=0;}
        public void endDocument() throws SAXException {System.out.println(nb);}
        public void startElement(String namespaceURI, String sName,
                                 String qName, Attributes attrs)  throws SAXException {
            if (qName == "livre")
                if (Integer.parseInt(attrs.getValue("annee"))>1960) nb = nb +1;
        }

        public static void main(String argv[]){...}
}
```

JDK 1.5 : extensions de SAX

- Deux nouveaux "handler" : "DeclHandler" et "LexicalHandler"
  - permet de gérer :
    - DeclHandler : la DTD
      - Déclaration d'attributs
      - Déclaration d'élément
      - Déclaration d'entité interne et externe
    - LexicalHandler
      - Début/Fin de DTD
      - Commentaires
      - Sections CDATA
      - Début/Fin d'entité
- Un nouveau gestionnaire par défaut : "DefaultHandler2"
  - hérite de "DefaultHandler"

Extensions de SAX

```
import org.xml.sax.ext.DefaultHandler2;
...
public class visu_sax2 extends DefaultHandler2 {
        public void comment(char[] ch, int start, int length) throws SAXException{
                System.out.println("comment");
        }
        public void startCDATA() throws SAXException {
                System.out.println("startCDATA");
        }
        public void endCDATA() throws SAXException {
                System.out.println("endCDATA");
        }
        public void startEntity(String name) throws SAXException {
                System.out.println("startEntity");
        }
        public void endEntity(String name) throws SAXException {
                System.out.println("endEntity");
        }
        public void startDTD(String name, String publicId, String systemId)
                        throws SAXException{
                System.out.println("startDTD");
        }
        public void endDTD() throws SAXException{
                System.out.println("endDTD");
        } ...
}
```

PHP et SAX

- Possibilité de gérer des flux SAX
  - http://www.php.net/manual/fr/ref.xml.php
  - Création/Libération
    - resource xml_parser_create ([string encoding])
    - resource xml_parser_create_ns ([string encoding])
    - bool xml_parser_free (resource parser)
  - Parse
    - void xml_set_object (resource parser, object &object)
    - bool xml_parse (resource parser, string data [, bool is_final])
    - mixed xml_parser_get_option (resource parser, int option)
    - bool xml_parser_set_option (resource parser, int option, mixed value)
      - XML_OPTION_CASE_FOLDING
      - XML_OPTION_TARGET_ENCODING
        - ISO-8859-1, US-ASCII ou UTF-8

Gestion des événements en PHP-Sax

- Gestion des éléments
  - bool xml_set_element_handler (resource parser, callback start_element_handler, callback end_element_handler)
    - start_element_handler (resource parser, string name, array attribs)
    - end_element_handler (resource parser, string name)
- Gestion des caractères
  - bool xml_set_character_data_handler (resource parser, callback handler)
    - handler (resource parser, string data)
- Gestion des instructions de traitement
  - bool xml_set_processing_instruction_handler (ressource parser, callback handler)
    - handler (resource parser, string target, string data)
- Gestion par défaut
  - bool xml_set_default_handler (resource parser, callback handler)
    - handler (resource parser, string data)

Exemple PHP-Sax

```
class visu_sax {
private $sax;
  function __construct() {
    $this->sax = xml_parser_create('UTF-8');
    xml_set_object($this->sax,$this);
    xml_parser_set_option($this->sax,XML_OPTION_CASE_FOLDING, FALSE);
    xml_set_character_data_handler($this->sax,'texte_node');
    xml_set_element_handler($this->sax,'deb_elem','fin_elem');
    xml_set_processing_instruction_handler($this->sax,'proc_ins');
    xml_set_default_handler($this->sax,'defaut');
  }
  function __destruct() {xml_parser_free($this->sax);}

  function parse($fic) {
    $fic=file_get_contents('ex_asimov.xml');
    xml_parse($this->sax,$fic,TRUE);
  }

  function texte_node($sax, $txt) {
    $txt = trim($txt);
    if (!(empty($txt))) echo "txt#".utf8_decode($txt)."#";
  }
  function deb_elem($sax, $nom, $att) {echo 'Début:'.utf8_decode($nom)."\n";}
  function fin_elem($sax, $nom) {echo 'Fin:'.utf8_decode($nom)."\n";}
  function proc_ins($sax, $cible, $contenu) {echo 'PI:'.$cible."\n";}
  function defaut($sax,$data) {echo '???'.$data."\n";}
}
```

Exemple PHP-Sax

```
class visu_sax {
private $sax;
  function __construct() {
    $this->sax = xml_parser_crea
    xml_set_object($this->sax,$t
    xml_parser_set_option($this-
    xml_set_character_data_hand
    xml_set_element_handler($th
    xml_set_processing_instructi
    xml_set_default_handler($thi
  }
  function __destruct() {xml_par

  function parse($fic) {
    $fic=file_get_contents('ex_asi
    xml_parse($this->sax,$fic,TRU
  }

  function texte_node($sax, $txt
    $txt = trim($txt);
    if (!(empty($txt))) echo "txt#"
  }
  function deb_elem($sax, $nom,
  function fin_elem($sax, $nom)
  function proc_ins($sax, $cible, $
  function defaut($sax,$data) {ec
}
```

```
class visu_sax {
private $sax;
  function texte_node($sax, $txt) {
    $txt = trim($txt);
    if (!(empty($txt))) echo "txt#".utf8_decode($txt)."#";
  }
  function deb_elem($sax, $nom, $att) {echo 'Début:'.utf8_decode($nom)."\n";}
  function fin_elem($sax, $nom) {echo 'Fin:'.utf8_decode($nom)."\n";}
  function proc_ins($sax, $cible, $contenu) {echo 'PI:'.$cible."\n";}
  function defaut($sax,$data) {echo '???'.$data."\n";}
}

function doSAX($fichier) {
    $sax = xml_parser_create('UTF-8');
    xml_set_object($sax, new visu_sax() );
    xml_parser_set_option($sax,XML_OPTION_CASE_FOLDING, FALSE);

    xml_set_character_data_handler($sax,'texte_node');
    xml_set_element_handler($sax,'deb_elem','fin_elem');
    xml_set_processing_instruction_handler($sax,'proc_ins');
    xml_set_default_handler($sax,'defaut');

    $fic=file_get_contents($fichier);
    xml_parse($sax,$fic,TRUE);

    xml_parser_free($sax);
}
```

Exemple PHP-Sax (suite)

```
<?php

$visu = new visu_sax();
$visu->parse("ex_asimov.xml");

?>
```

Ou

```
<?php

doSAX("ex_asimov.xml");

?>
```

```
Début:auteur
Début:identité
Début:nom
txt#Asimov#Fin:nom
Début:prenom
txt#Isaac#Fin:prenom
Début:date-naissance
txt#1920-01-02#Fin:date-naissance
Début:date-deces
txt#1992-04-06#Fin:date-deces
Début:nationalite
txt#Russe/Am#txt#éricain#Fin:nationalite
Fin:identité
???<!-- Liste des ouvrages -->
Début:livre
Début:titre
txt#Un d#txt#éfilé de robots#Fin:titre
Fin:livre
Début:livre
Début:titre
txt#I, robot#Fin:titre
Fin:livre
Début:livre
Début:titre
txt#Le robot qui r#txt#êvait#Fin:titre
Fin:livre
Fin:auteur
```

Autre Solution : Sax4PHP

- Outil GPL pour "simuler" Sax (version Java) en PHP
  - http://sax4php.sourceforge.net/

```
<?php header('Content-type: text/plain');
        include('Sax4PHP/Sax4php.php');
class visu_sax extends DefaultHandler {
  function characters($txt) {
    $txt = trim($txt);
    if (!(empty($txt))) echo "txt#".utf8_decode($txt)."#";
  }
  function startElement($nom, $att) {echo 'Début:'.utf8_decode($nom)."\n";}
  function endElement($nom) {echo 'Fin:'.utf8_decode($nom)."\n";}
  function processingInstruction($cible, $contenu) {echo 'PI:'.$cible."\n";}
  function node($data) {echo '???'.utf8_decode($data)."\n";}
}
$visu = file_get_contents('ex_asimov.xml');
try {
            $sax = new SaxParser(new visu_sax());
            $sax->parse($visu);
}catch(SAXException $e){ echo "\n",$e;
}catch(Exception $e) {echo "Capture l'exception par défaut\n", $e;}
?>
```

Exemple (supplémentaire) de SAX en PHP : Compter le nombre de livres depuis 1960

```php
<?php class nbl_sax {
private $sax;
public $nb;
  function __construct() {
    $this->sax = xml_parser_create('UTF-8');
    xml_set_object($this->sax,$this);
    xml_parser_set_option($this->sax,XML_OPTION_CASE_FOLDING, FALSE);
    xml_set_element_handler($this->sax,'deb_elem','fin_elem');
    xml_set_default_handler($this->sax,'defaut');
  }
  function __destruct() {xml_parser_free($this->sax);}
  function compter($fic_name) {
    $this->nb = 0;
    $fic=file_get_contents($fic_name);
    xml_parse($this->sax,$fic,TRUE);
  }
  function deb_elem($sax, $nom, $att) {
    if ($nom=='livre') if ($att['annee']>1960) $this->nb += 1;}
}
$nbl = new nbl_sax();
$nbl->compter("ex_asimov.xml");
echo $nbl->nb;
?>
```

Exemple (supplémentaire) de SAX en PHP : Compter le nombre de livres depuis 1960 (Sax4php)[

```php
<?php header('Content-type: text/plain');
include('Sax4PHP/Sax4php.php');

class nbl_sax2 extends DefaultHandler {
  public $nb;
  function startDocument() {$this->nb = 0;}
  function endDocument() {echo $this->nb;}
  function startElement($nom, $att) {
            if ($nom=='livre') if ($att['annee']>1960) $this->nb += 1;}
}

$fic = file_get_contents('ex_asimov.xml');
try {
            $nbl = new nbl_sax2();
            $sax = new SaxParser($nbl);
            $sax->parse($fic);
}catch(SAXException $e){
            echo $e;
}?>
```

SAX avantages et inconvénients

- Avantages
  - Ne nécessite pas d'avoir tout le document en mémoire
  - Le traitement de gros documents est aisé
  - Permet d'entamer le traitement (et la sortie du début d'un résultat) alors que l'ensemble du document n'est pas encore chargé.
  - Permet de remplir directement les structures de données spécifiques à une application.
- Inconvénients
  - Traitement devant être quasi-linéaires (Réf. avant/arrière difficile)
  - Détection tardive des erreurs de validation
  - Nécessite de définir une méthode spécifique pour la sérialisation.

Exercices

- Écrire un programme utilisant l'API SAX (en Java ou en PHP) pour générer une page HTML de type "Contact" avec le document 1.

```
<?xml version="1.0" encoding="iso8859-1"?>
<personne>
  <nom val="Dupont"/>
  <prénom val="Max"/>
  <courriel val="md@dm.net"/>
</personne>
```

- Faire la même chose avec le document 2

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
</personne>
```

Exercices

Mes contacts

Nom	Prénom	Courriel
Dupont	Max	md@dm.net

- Écrire un programme utilisant l'API SAX (en Java ou en PHP) pour générer une page HTML de type "Contact" avec le document 1.

```
<?xml version="1.0" encoding="iso8859-1"?>
<personne>
  <nom val="Dupont"/>
  <prénom val="Max"/>
  <courriel val="md@dm.net"/>
</personne>
```

- Faire la même chose avec le document 2

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
</personne>
```

Exercices

```php
<?php header('Content-type: text/html; charset=utf-8');
include('Sax4PHP/Sax4php.php');

class Contacts01 extends DefaultHandler {
  function startDocument() {
        echo '<html><head><title>Exo SAX 01</title></head><body>';
        echo '<h1>Mes contacts</h1><table border="1">';
        echo '<tr><th>Nom</th><th>Prénom</th><th>Courriel</th></tr>';
  }

  function endDocument() {echo '</table></body></html>'; }

  function startElement($nom, $att) {
        switch(utf8_decode($nom)) {
                case 'personne' : echo '<tr>'; break;
                case 'nom' : echo '<td>'.utf8_decode($att['val']).'</td>'; break;
                case 'prénom' : echo '<td>'.utf8_decode($att['val']).'</td>'; break;
                case 'courriel' : echo '<td>'.utf8_decode($att['val']).'</td></tr>'; break;
                default:;
        }
  }
}

$xml = file_get_contents('../xml/exo_sax_01_01.xml');
try {
        $sax = new SaxParser(new Contacts01());
        $sax->parse($xml);
}catch(SAXException $e){ echo "\n",$e;}
 catch(Exception $e) {echo "Capture l'exception par défaut\n".$e;}
?>
```

Mes contacts

Nom	Prénom	Courriel
Dupont	Max	md@dm.net

1.

```xml
<?xml version="1.0" encoding="iso8859-1"?>
<personne>
  <nom val="Dupont"/>
  <prénom val="Max"/>
  <courriel val="md@dm.net"/>
</personne>
```

```xml
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
</personne>
```

Exercices

Mes contacts

Nom	Prénom	Courriel
Dupont	Max	md@dm.net

- Écrire un programme utilisant l'API SAX (en Java ou en PHP) pour générer une page HTML de type "Contact" avec le document 1.

```
<?xml version="1.0" encoding="iso8859-1"?>
<personne>
  <nom val="Dupont"/>
  <prénom val="Max"/>
  <courriel val="md@dm.net"/>
</personne>
```

- Faire la même chose avec le document 2

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
</personne>
```

Exercices

```
class Contacts02 extends DefaultHandler {
private $texte;

  function characters($txt) {
    $txt_reduit = trim($txt);
    if (!(empty($txt_reduit))) $this->texte .= $txt;
  }

  function startDocument() {
        echo '<html><head><title>Exo SAX 01</title></head><body>';
        echo '<h1>Mes contacts</h1><table border="1">';
        echo '<tr><th>Nom</th><th>Prénom</th><th>Courriel</th></tr>';
  }

  function endDocument() {echo '</table></body></html>';}

  function startElement($nom, $att) { $this->texte = '';}

  function endElement($nom) {
        switch(utf8_decode($nom)) {
                case 'personne' : echo '</tr>'; break;
                case 'nom' : echo '<tr><td>'.utf8_decode($this->texte).'</td>'; break;
                case 'prénom' : echo '<td>'.utf8_decode($this->texte).'</td>'; break;
                case 'courriel' : echo '<td>'.utf8_decode($this->texte).'</td>'; break;
                default:;
        }
  }
}...
```

Mes contacts

Nom	Prénom	Courriel
Dupont	Max	md@dm.net

1.

```
<?xml version="1.0" encoding="iso8859-1"?>
<personne>
  <nom val="Dupont"/>
  <prénom val="Max"/>
  <courriel val="md@dm.net"/>
</personne>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
</personne>
```

Exercices

- Écrire un programme utilisant l'API SAX (en Java ou en PHP) pour générer une page HTML de type "Contact" avec le document 3, en indiquant l'institut et son adresse en taille de police assez petite ("x-small").

- Faire la même chose avec le document 4 où un contact peut avoir plusieurs prénoms (mais un seul prénom principal), mais il ne faut afficher que le prénom principal.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
 </personne>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom principal="vrai">Max</prénom>
  <prénom principal="faux">Antoine</prénom>
  <prénom principal="faux">Jérémie</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
</personne>
```

Exercices

Mes contacts

Nom	Prénom	Courriel	Institut	Adresse
Dupont	Max	md@dm.net	LINA	2 rue de la Houssinière, BP92208, 44322 Nantes Cedex 03

- Écrire un programme utilisant l'API SAX (en Java ou en PHP) pour générer une page HTML de type "Contact" avec le document 3, en indiquant l'institut et son adresse en taille de police assez petite ("x-small").

- Faire la même chose avec le document 4 où un contact peut avoir plusieurs prénoms (mais un seul prénom principal), mais il ne faut afficher que le prénom principal.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
 </personne>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom principal="vrai">Max</prénom>
  <prénom principal="faux">Antoine</prénom>
  <prénom principal="faux">Jérémie</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
</personne>
```

Exercices

```
class Contacts03 extends DefaultHandler {
private $texte='';
private $institut=false;

 function characters($txt) { $txt_reduit = trim($txt); if (!(empty($txt_reduit))) $this->texte .= $txt; }

 function startDocument() {
        echo '<html><head><title>Exo SAX 01</title></head><body>';
        echo '<h1>Mes contacts</h1><table border="1">';
        echo '<tr><th>Nom</th><th>Prénom</th><th>Courriel</th>
<th>Institut</th><th>Adresse</th></tr>';
 }

 function endDocument() {echo '</table></body></html>';}

 function startElement($nom, $att) {
        $this->texte = '';
        switch(utf8_decode($nom)) {
                case 'institut' : $this->institut = true; break;
                case 'personne' : echo '<tr>'; break;
                default:;
        }
 }

 function endElement($nom) {
        switch(utf8_decode($nom)) {
                case 'personne' : echo '</tr>'; break;
                case 'institut' : $this->institut = false; break;
                case 'adresse' : echo '<td style="font-size:x-small">'.
                                utf8_decode($this->texte).'</td>'; break;
                case 'nom' :
                        if ($this->institut) echo '<td style="font-size:x-small">'; else echo '<td>';
                        echo utf8_decode($this->texte).'</td>';
                        break;
                case 'prénom' : echo '<td>'.utf8_decode($this->texte).'</td>'; break;
                case 'courriel' : echo '<td>'.utf8_decode($this->texte).'</td>'; break;
                default:;
        }
 }
} ...
```

afficher que le prénom principal.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
</personne>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom principal="vrai">Max</prénom>
  <prénom principal="faux">Antoine</prénom>
  <prénom principal="faux">Jérémie</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
</personne>
```

Exercices

Mes contacts

Nom	Prénom	Courriel	Institut	Adresse
Dupont	Max	md@dm.net	LINA	2 rue de la Houssinière, BP92208, 44322 Nantes Cedex 03

- Écrire un programme utilisant l'API SAX (en Java ou en PHP) pour générer une page HTML de type "Contact" avec le document 3, en indiquant l'institut et son adresse en taille de police assez petite ("x-small").

- Faire la même chose avec le document 4 où un contact peut avoir plusieurs prénoms (mais un seul prénom principal), mais il ne faut afficher que le prénom principal.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
 </personne>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom principal="vrai">Max</prénom>
  <prénom principal="faux">Antoine</prénom>
  <prénom principal="faux">Jérémie</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
</personne>
```

Exercices

```
class Contacts04 extends DefaultHandler {
private $texte='';
private $institut=false;
private $prenom_principal=false;

 function characters($txt) { $txt_reduit = trim($txt); if (!(empty($txt_reduit))) $this->texte .= $txt;}

 function startDocument() {
        echo '<html><head><title>Exo SAX 01</title></head><body>';
        echo '<h1>Mes contacts</h1><table border="1">';
        echo '<tr><th>Nom</th><th>Prénom</th><th>Courriel</th>
<th>Institut</th><th>Adresse</th></tr>';
 }

 function endDocument() {echo '</table></body></html>'; }

 function startElement($nom, $att) {
        $this->texte = '';
        switch(utf8_decode($nom)) {
                case 'institut' : $this->institut = true; break;
                case 'personne' : echo '<tr>'; break;
                case 'prénom' : $this->prenom_principal = $att['principal'] == 'vrai'; break;
                default:;
        }
 }

 function endElement($nom) {
        switch(utf8_decode($nom)) {
                case 'personne' : echo '</tr>'; break;
                case 'institut' : $this->institut = false; break;
                case 'adresse' : echo '<td style="font-size:x-small">'.
                                utf8_decode($this->texte).'</td>'; break;
                case 'nom' :
                        if ($this->institut) echo '<td style="font-size:x-small">'; else echo '<td>';
                        echo utf8_decode($this->texte).'</td>';
                        break;
                case 'prénom' :
                        if ($this->prenom_principal) echo '<td>'.utf8_decode($this->texte).'</td>';
                        break;
                case 'courriel' : echo '<td>'.utf8_decode($this->texte).'</td>'; break;
                default:;
        }
 }
} ...
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
 </personne>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom principal="vrai">Max</prénom>
  <prénom principal="faux">Antoine</prénom>
  <prénom principal="faux">Jérémie</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
</personne>
```

Exercices

Mes contacts

Nom	Prénom	Courriel	Institut	Adresse
Dupont	Max	md@dm.net	LINA	2 rue de la Houssinière, BP92208, 44322 Nantes Cedex 03

- Écrire un programme utilisant l'API SAX (en Java ou en PHP) pour générer une page HTML de type "Contact" avec le document 3, en indiquant l'institut et son adresse en taille de police assez petite ("x-small").

- Faire la même chose avec le document 4 où un contact peut avoir plusieurs prénoms (mais un seul prénom principal), mais il ne faut afficher que le prénom principal.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom>Max</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
 </personne>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<personne>
  <nom>Dupont</nom>
  <prénom principal="vrai">Max</prénom>
  <prénom principal="faux">Antoine</prénom>
  <prénom principal="faux">Jérémie</prénom>
  <courriel>md@dm.net</courriel>
  <institut>
        <nom>LINA</nom>
        <adresse>2 rue de la Houssinière,
BP92208, 44322 Nantes Cedex 03</adresse>
  </institut>
</personne>
```

Bilan sur SAX

- Applications typiques
  - Chargement dans des structures spécifiques à l'application
  - Filtrages
  - Transformations simples
  - Traitement local des informations
  - Serveur de pages
  - Construction d'un DOM
- Méthode événementielle
- API légère permettant de traiter des données "à la volée"
- Facilite le chargement de données