Multimédia

- XML

Benoît Piranda
Équipe SISAR
Université de Marne La Vallée

XML l'avenir de HTML ?

- **Extensible Markup Language, entre HTML et SGML**
  - **tente de se servir des principes de simplicité du HTML et de la souplesse SGML**
  - **Richesse sémantique**
    - **plus structuré que HTML**
    - **langage de mise en forme**
    - **permet l'écriture de ses propres DTD**
  - **Adapté à la réalisation de documents pour Internet**
    - **moins lourds à mettre en œuvre que SGML**
    - **conserve la structure initiale de HTML (balises prédéfinies)**
- **XML 1.0 est recommandé officiellement par W3C depuis 1998.**

XML

- **Caractéristique importante :**
  - **la mise en forme des données est totalement séparée des données elles-mêmes.**
    - **Cela permet de séparer complètement l'information (le contenu) de son apparence (le contenant),**
    - **et donc de fournir plusieurs types de sortie pour un même fichier de données,**
      - ✓ **en fonction de l'utilisateur**
      - ✓ **ou de l'application (tableau, graphique, image, animation multimédia, fichier HTML, fichier PDF...).**

La norme XML

- **La DTD**
  - **si le document contient une DTD il doit la respecter**
  - **il doit respecter un formalisme plus rigoureux que HTML**
    - **pas de balise d 'ouverture sans fermeture (et vice et versa)**
- **Les feuilles de style**
  - **Structuration des données pour l'affichage**
  - **CSS : issus de l'HTML**
  - **XSL : avec langage de programmation**
- ***Encore en cours de développement***
  - **Des liens hypertextes étendus XLL**
    - **liens multidirectionnels**
      - ✓ **pour retourner au point de départ d'un lien**
    - **inclusion d'un sous-document lié dans le document courant**

Format XML

- **Entête**
  - **Version**
  - **Codage du jeu de caractères**
    - **ISO-8859-1 : europe de l'ouest**
  - **Référence DTD externe ou interne**
    - **Standalone="yes" : DTD dans la page**
    - **Standalone="no" : DTD dans un fichier externe**

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
```

La DTD

- **Document Type Definition**
  - **Grammaire permettant de vérifier la conformité d'un document**
    - **Document valide si vérifiant sa DTD**
    - **Document bien formé si il répond aux règles de XML**
  - **Emplacement de la DTD**
    - **Forme interne : dans le fichier**

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE élément-racine [déclaration des éléments]>
```

    - **Forme externe : dans un fichier accessible par son URL**

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE élément-racine SYSTEM "nom_du_fichier.dtd">
```

  - **Langage simple permettant de définir les règles de dérivation**

Règle de définition de la DTD

- **Déclarer un élément**
  - **<! ELEMENT Nom Modèle >**
  - **Nom : intitulé de l'élément**
  - **Modèle**
    - Type du contenu
      - ✓ ANY
      - ✓ #PCDATA
      - ✓ EMPTY
    - Liste des sous-balises possibles

```
<!ELEMENT personne (nom,prenom,telephone),email? >
<!ELEMENT nom (#PCDATA) >
<!ELEMENT prenom (#PCDATA) >
<!ELEMENT telephone (#PCDATA) >
<!ELEMENT email (#PCDATA) >
```

Exemple de http://fr.selfhtml.org/xml/dtd

Règle de définition de la DTD

- **Règle d'énumération**
  - **? : facultatif**
  - ***,+ : tuples**
  - **| : ou**
  - **, : séparateur de liste**

```
<!ELEMENT recettes (liste_ingredients, suite_instructions)>
<!ELEMENT liste_ingredients (ingredient)+>
<!ELEMENT ingredient (#PCDATA)>
<!ELEMENT suite_instructions (instruction)*>
<!ELEMENT instruction (#PCDATA)>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Exemples

```
<!ELEMENT adresses (adresse)*>
<!ELEMENT adresse (titre?,nom,(boite_postale|numero_rue),cp_ville)>
<!ELEMENT titre (#PCDATA)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT boite_postale (#PCDATA)>
<!ELEMENT numero_rue (#PCDATA)>
<!ELEMENT cp_ville (#PCDATA)>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE adresses SYSTEM "adresses.dtd">
<adresses>
<adresse>
<nom>Société Générale</nom>
<boite_postale>7001</boite_postale>
<cp_ville>13100 Aix en Provence</cp_ville>
</adresse>
<adresse>
<titre>Monsieur</titre>
<nom>Luc Minighetti</nom>
<numero_rue>112 rue de Lyon</numero_rue>
<cp_ville>13000 Marseille</cp_ville>
</adresse>
</adresses>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Exemple : ordre libre, contenu mixte

```
<!ELEMENT texte (#PCDATA | menace | rire | question | cynique)*>
<!ELEMENT menace (#PCDATA)>
<!ELEMENT rire (#PCDATA | clignant_oeil)*>
<!ELEMENT question (#PCDATA)>
<!ELEMENT clignant_oeil (#PCDATA)>
<!ELEMENT cynique (#PCDATA)>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE text SYSTEM "texte.dtd">
<texte> Quelquefois la conscience nous dit:
<menace>tu dois penser davantage aux femmes et aux hommes.</menace>
Alors, bien sûr, on se demande parfois,
<question>pourquoi il y a là tant matière à reflexion,</question>
mais quelquefois aussi, on obéit et réfléchit. La femme dit à
l'homme:
<rire>Oh chéri, <clignant_oeil>tu vaux ton poids
d'or!</clignant_oeil> </rire>
Et l'homme répond:
<cynique>Oui, parce que je pousse le chariot et que j'y ai mis une
pièce de dix francs!</cynique>
La femme rétorque: <rire>tu as tout compris!</rire>
</texte>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Exemple : ANY / EMPTY

```
<!ELEMENT anytext ANY>
<!ELEMENT anglais (#PCDATA)>
<!ELEMENT italiano (#PCDATA)>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE anytext SYSTEM "anytext.dtd">
<anytext> c'est un peu de texte qui signifie en anglais:
<anglais>this is some text</anglais>
et en italien:
<italiano>ciò è un certo testo</italiano>
</anytext>
```

```
<!ELEMENT lignes_texte (#PCDATA | nouvelle_ligne)*>
<!ELEMENT nouvelle_ligne EMPTY>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE lignes_texte SYSTEM "lignes_texte.dtd">
<lignes_texte> Ceci est le texte, mais où commence la
<nouvelle_ligne />
nouvelle ligne? </lignes_texte>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Les entités

- **Abréviations définies**
  - **Exemple HTML : **
- **Schéma de définition**

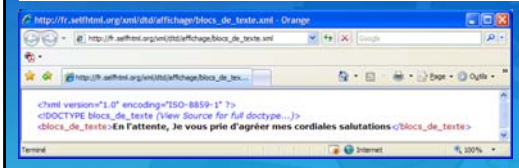
```
<!ENTITY [%] Nom [SYSTEM|PUBLIC] "valeur" [mentions supplémentaires] >
```

- **Utilisation**
  - **&Nom;**

Exemple d'entités

```
<!ELEMENT blocs_de_texte (#PCDATA)>
<!ENTITY jpa "Je vous prie d'agr&#233;er mes cordiales salutations">
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE blocs_de_texte SYSTEM "blocs_de_texte.dtd">
<blocs_de_texte> En l'attente, &jpa;
</blocs_de_texte>
```



Exemple de http://fr.selfhtml.org/xml/dtd

Exemple d'entité

```
<!ENTITY % article "numero_article, nom_article, quantite_article">
<!ENTITY % additif "description | classe_produit">
<!ELEMENT stock (entree | sortie)*>
<!ELEMENT entree (numero_entree, (%article;), (%additif;))>
<!ELEMENT sortie (numero_sortie, (%article;), (%additif;))>
<!ELEMENT numero_entree (#PCDATA)> <!ELEMENT numero_sortie (#PCDATA)>
<!ELEMENT numero_article (#PCDATA)> <!ELEMENT nom_article (#PCDATA)>
<!ELEMENT quantite_article (#PCDATA)>
<!ELEMENT description (#PCDATA)> <!ELEMENT classe_produit (#PCDATA)>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE stock SYSTEM "stock.dtd">
<stock>
<entree>
<numero_entree>452</numero_entree>
<numero_article>45-234</numero_arcicle>
<nom_article>fauteuil de bureau "ministre"</nom_article>
<quantite_article>10</quantite_article>
<classe_produit>C-III</classe_produit>
</entree>
<sortie>
<numero_sortie>318</numero_sortie>
<numero_arcicle>37-917</numero_arcicle>
<nom_article>armoire à glaces "Narcisse"</nom_article>
```

http://fr.selfhtml.org/xml/dtd

Règle de définition des attributs

- **Attributs : paramètres de la balise**
- **Déclaration d'attributs**
  - **<! ATTLIST Elément Attribut Type >**
- **Type représente le type de donnée de l'attribut, il en existe trois :**
  - **CDATA : une chaîne de caractères**
  - **ID : identifiant unique**
  - **IDREF : référence à un identifiant existant**
  - **Liste de valeurs possibles**
    - **<! ATTLIST Elément Attribut (Valeur1 | Valeur2 | ...) "val.def." >**
- **Niveau de nécessité de l'attribut :**
  - **#IMPLIED : optionnel**
  - **#REQUIRED : obligatoire**
  - **#FIXED : affectation d'une valeur par défaut s'il n'est pas défini**

Exemple d'attributs

```
<!ELEMENT autos (auto)*>
<!ELEMENT auto EMPTY>
<!ATTLIST auto
type              CDATA      #REQUIRED
annee_construction CDATA     #REQUIRED
km                CDATA      #REQUIRED
puissance         CDATA      #REQUIRED
prix_vente        CDATA      #REQUIRED
>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE autos SYSTEM "autos.dtd">
<autos>
<auto type="AUDI 80" annee_construction="1992" km="125000"
puissance="90" prix_vente="6250 Euros" />
</autos>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Exemple d'attributs

```
<!ELEMENT hotels (hotel)*>
<!ELEMENT hotel (#PCDATA)>
<!ATTLIST hotel
nom               CDATA           #REQUIRED
categorie         (I|II|III|IV|V) #REQUIRED
chambre_simple    (oui|non)       #IMPLIED
chambre_double    (oui|non)       "oui"
>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE hotels SYSTEM "hotels.dtd">
<hotels>
<hotel nom="Au lion d'or" categorie="IV">hôtel situé en lisière de
forêt, 150 lits, calme et cher. </hotel>
<hotel nom="Arabesque" categorie="II" chambre_double="oui"
chambre_simple="oui"> hôtel en ville simple, 400 chambres,
satisfaisant, sans confort particulier.</hotel>
<hotel nom="Lancelot" categorie="III" chambre_simple="non">hôtel
situé dans le centre, 100 lits, agréable, facilement accessible.
</hotel>
</hotels>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Exemple d'attributs avec identifiant

```
<!ELEMENT livres (livre)*>
<!ELEMENT livre (#PCDATA)>
<!ATTLIST livre
isbn    ID      #REQUIRED
titre   CDATA   #REQUIRED
auteur  CDATA   #REQUIRED
>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE livres SYSTEM "livres.dtd">
<livres>
<livre isbn="nr_3-90193-3949-7" titre="Fleurette facile"
auteur="Professeur Trouvetout"> Une introduction entre le génie et
la folie.
</livre>
<livre isbn="nr_3-90193-3950-2" titre="Fleurette facile II"
auteur="Professeur Trouvetout"> Une autre introduction entre le
génie et la folie.
</livre>
</livres>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Exemple d'attributs avec référence

```
<!ELEMENT liste (point)*>
<!ELEMENT point (#PCDATA)>
<!ATTLIST point
nom           ID       #REQUIRED
point_parent  IDREF    #IMPLIED
>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE liste SYSTEM "liste.dtd">
<liste>
<point nom="racine">contenu</point>
<point nom="introduction" point_parent="racine">introduction</point>
<point nom="histoire" point_parent="introduction">histoire</point>
<point nom="aujourd_hui" point_parent="introduction">etat
actuel</point>
<point nom="pas" point_parent="racine">premiers pas</point>
<point nom="exemple" point_parent="pas">un petit exemple</point>
</liste>
```

Exemple de http://fr.selfhtml.org/xml/dtd

Règle de définition de la DTD

- **Exemple de définition d'attributs**
  - **La balise img est une balise vide dont les attributs sont**
    - **src pour le lien (obligatoire)**
    - **alt pour le texte (obligatoire)**
    - **Longdesc (facultatif)**
    - **height (facultatif)**
    - **weight (facultatif)**
    - **usemap (facultatif)**
    - **ismap (facultatif)**

```
<!ELEMENT img EMPTY>
<!ATTLIST img
   %attrs;
   src        %URI;       #REQUIRED
   alt        %Text;      #REQUIRED
   longdesc   %URI;       #IMPLIED
   height     %Length;    #IMPLIED
   width      %Length;    #IMPLIED
   usemap     %URI;       #IMPLIED
   ismap      (ismap)     #IMPLIED
>
```

XML principe

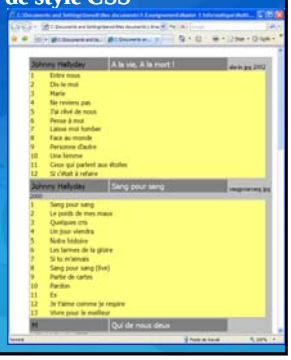
- **Un premier exemple simple**
  - **Liste d'informations avec champs facultatifs**
    - **Liste de musiques**
    - **Pas toujours de photo de l'album**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<music SUBJECT="XML">
<album>
<artiste>Johnny Hallyday</artiste>
<titre>A la vie, A la mort !</titre>
<photo>alavie.jpg</photo>
<date>2002</date>
<piste><numero>1</numero><chanson>Entre nous</chanson></piste>
<piste><numero>2</numero><chanson>Dis-le moi</chanson></piste>
<piste><numero>3</numero><chanson>Marie</chanson></piste>
…
<piste><numero>11</numero><chanson>Ceux qui parlent aux
étoiles</chanson></piste>
<piste><numero>12</numero><chanson>Si c'était à refaire</chanso
</album>
<album>
<artiste>Johnny Hallyday</artiste>
<titre>Sang pour sang</titre>
<photo>sangpoursang.jpg</photo>
<date>2000</date>
```

XML principe

- **Utilisation d'une feuille de style CSS**

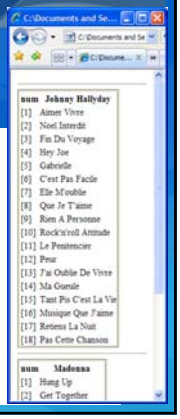
```
album
 { position:relative;
   display:block;
   width:620px;
   top:10px;
   left:40px;
   background-color:#C0C0C0;
   padding:2px;
 }
artiste
 { position:relative;
   width:200px;
   background-color:#808080;
   color:#000000;
   font-family:Tahoma,Arial,Helvetica,sans-serif;
   font-size:14pt;
   padding:5px;
 }
titre
 { position:relative;
   width:300px;
   background-color:#808080;
   color:#FFFFFF;
   font-family:Tahoma,Arial,Helvetica,sans-serif;
   font-size:14pt;
   padding:5px;
 }
```



XML Principe

- **Lien avec une feuille de style xsl**
  - **Affichage des données**
  - **Extraction des informations**

```
</head>
<body>
  <xsl:for-each select="music/album">
  <hr />
  <table class="type1">
  <tr><th>num</th><th><xsl:value-of select="@artiste" />
</th></tr>
  <xsl:for-each select="piste">
    <TR><TD>[<xsl:value-of select="numero" />]</TD>
    <TD><xsl:value-of select="chanson" /></TD>
    </TR>
  </xsl:for-each>
  </table>
  </xsl:for-each>
</body>
</html>
</xsl:template>

</xsl:stylesheet>
```



XML Principe

- **Plusieurs mises en forme des données**
  - **En changeant le ficher xsl associé**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href="model1.xsl" type="text/xsl" ?>
<music SUBJECT="XML">
```

XML Principe

- **Description de scène 3D**
  - **Problème assez complexe**
  - **Liste d'objets dont on donne des informations sur la géométrie, la radiométrie (couleur)**
    - **Géométrie : sphère, bloc, cylindre…**
      - ✓ **Chaque type a des paramètres propres**
        - ➢ **Rayon, hauteur…**
    - **Radiométrie**
      - ✓ **Composante diffuse, spéculaire**
        - ➢ **Peuvent être des couleurs, ou des textures**
      - ✓ **Objet miroir, transparent ?**

XML Principe

- **On l'affiche en HTML**

```
<html><head></head>
<body bgcolor=#777777>
<h2>liste des objets</h2>
<ul>
<li><i>sphere</i>, rayon 2, origine en (1.,0.,2.),
<font color=#CC4C99>couleur diffuse(0.8,0.3,0.6)</font>,
<font color=#E5E5E5>couleur spéculaire(0.9,0.9,0.9)</font>
</li>
<li><i>pavé</i>, coté 1, origine en (2.,-3.,2.), texture diffuse
(bois.png),
<font color=#E5E5E5>couleur spéculaire(0.9,0.9,0.9)</font>,
miroir</li>
<li><i>cylindre</i>, rayon 2, hauteur 2, origine en (1.,0.,0.),
<font color=#CC4C99>couleur diffuse(0.8,0.3,0.6)</font></li>
</ul>
</body>
</html>
```

XML Principe

- **Les balises ne servent qu'à la mise en page**
- **Pas d'information sémantique**
  - **Pas de lien entre les données d'une ligne**

liste des objets

- *sphere*, rayon 2, origine en (1.,0.,2.), couleur spéculaire(0.9,0.9,0.9)
- *pavé*, coté 1, origine en (2.,-3.,2.), texture diffuse (bois.png), couleur spéculaire(0.9,0.9,0.9) miroir
- *cylindre*, rayon 2, hauteur 2, origine en (1.,0.,0.),

XML Principe

- **Utilisons le langage XML pour décrire les informations**
  - **Données uniquement**
  - **Pas de mise en forme pour l'instant**
- **Règles**
  - **Les balises**
    - **Simples**
      - ✓ **<X>…</X>**
    - **Vides**
      - ✓ **Objet miroir : pas de paramètre**
    - **Avec paramètres : attributs**
      - ✓ **<OBJET id="1">…</OBJET>**
      - ✓ **Guillemets obligatoires**

XML Principe

- **Choix de structuration de la SCENE**
  - **OBJET**
    - **@id, @nom**
    - **GEOMETRIE**
      - ✓ **@type (sphere, bloc, cylindre)**
      - ✓ **Paramètres simple (VALEUR)**
        - ➢ **@type**
      - ✓ **Paramètres vecteur (VEC3D)**
        - ➢ **@type**
    - **RADIOMETRIE**
      - ✓ **@type (diffus, spéculaire, miroir)**
      - ✓ **Paramètre de COULEUR**
        - ➢ **R,V,B**
      - ✓ **Paramètre de TEXTURE**
        - ➢ **NOM**
        - ➢ **Transformations 2D**

DTD correspondante

```
<!ELEMENT objet geometrie,(radiometrie)+ >
<!ATTLIST objet
id      CDATA   #REQUIRED
nom     CDATA   #IMPLIED
>
<!ELEMENT geometrie (valeur|vec3d)* >
<!ATTLIST geometrie
type    (sphere|bloc|cylindre)        #REQUIRED
>
<!ELEMENT valeur (#PCDATA) >
<!ATTLIST valeur
type    CDATA   #REQUIRED
>
<!ELEMENT vec3d (x,y,z) >
<!ATTLIST vec3d
type    CDATA   #REQUIRED
>
<!ELEMENT radiometrie (texture|couleur) >
<!ATTLIST radiometrie
type    (diffus|speculaire|miroir)    #REQUIRED
>
…
```

XML Principe

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SCENE SUBJECT="XML">
<OBJET id="1" nom="boule">
<GEOMETRIE type="sphere">
<VALEUR type="rayon">1.</VALEUR>
<VEC3D type="origine"><X>1.</X><Y>0.</Y><Z>2.</Z></VEC3D>
</GEOMETRIE>
<RADIOMETRIE type="diffus">
<COULEUR><R>0.8</R><V>0.3</V><B>0.6</B></COULEUR></RADIOMETRIE>
<RADIOMETRIE type="spéculaire">
<COULEUR><R>0.9</R><V>0.9</V><B>0.9</B></COULEUR></RADIOMETRIE>
</OBJET>
<OBJET id="2" nom="pavé">
<GEOMETRIE type="bloc">
<VEC3D type="dimensions"><X>1.</X><Y>1.</Y><Z>3.</Z></VEC3D>
<VEC3D type="origine"><X>1.</X><Y>0.</Y><Z>2.</Z></VEC3D>
</GEOMETRIE>
<RADIOMETRIE type="diffus">
<TEXTURE><NOM>bois.png</NOM><VEC2D type="echelle"><S>1.</S><T>1.</T></VEC2D>
<VEC2D type="décalage"><S>0.</S><T>0.5</T></VEC2D></TEXTURE></RADIOMETRIE>
<RADIOMETRIE type="spéculaire">
<COULEUR><R>0.9</R><V>0.9</V><B>0.9</B></COULEUR></RADIOMETRIE>
<RADIOMETRIE type="miroir"/>
</OBJET>
</SCENE>
```

XML Principe

- **Visualisation dans un navigateur**
  - **Les balises ont une signification**
  - **Elles sont organisées hiérarchiquement**

XML Principe

- **Mise en page des données**
  - **Feuille de style**
    - **CSS**
    - **XSL : eXtensible Style Language**
      - ✓ **Mêmes propriétés que CSS**
      - ✓ **Langage de transformation XSLT**
      - ✓ **Vocabulaire XML pour la sémantique de formatage**

Exemple de XSL

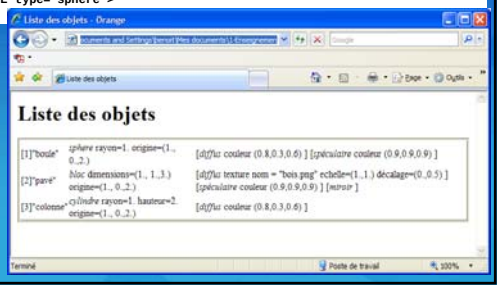
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<!-- Feuille de style pour la scène 3D -->
<xsl:template match="/">
<html><head>
<style>
table.type1
{ border-style:groove;
  border-collapse:collapse;
  background-color:White;
  font-family: Times,Helvetica;
  font-size: 14;
  line-height: 14pt;
}
</style></head>
<body><title>Liste des objets</title>
<h1>Liste des objets</h1>
  <table class="type1">
  <xsl:for-each select="SCENE/OBJET">
    <TR><TD>[<xsl:value-of select="@id" />]"<xsl:value-of select="@nom"
/>"</TD>
    <TD><xsl:apply-templates select="GEOMETRIE"/></TD>
    <TD><xsl:apply-templates select="RADIOMETRIE"/></TD>
    </TR>
  </xsl:for-each>
  </table></body></html>
</xsl:template>
```

Exemple de XSL (suite)

```
<xsl:template match="GEOMETRIE">
<i><xsl:value-of select="@type" /></i>
<xsl:apply-templates select="VALEUR"/>
<xsl:apply-templates select="VEC3D"/>
</xsl:template>

<xsl:template match="VEC3D">
<xsl:value-of select="@type" />=(<xsl:value-of select="X" />,<xsl:value-of
select="Y" />,<xsl:value-of select="Z"/>)
</xsl:template>

<xsl:template match="VALEUR">
<xsl:value-of select="@type" />=<xsl:value-of/>
</xsl:template>

<xsl:template match="TEXTURE">
texture nom = "<xsl:value-of select="NOM" />"
<xsl:apply-templates select="VEC2D"/>
</xsl:template>

<xsl:template match="RADIOMETRIE">
[<i><xsl:value-of select="@type" /></i> <xsl:apply-templates/>]
</xsl:template>

</xsl:stylesheet>
```

Combinaison XML avec XSL

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href="model1.xsl" type="text/xsl" ?>
<?xml version="1.0" encoding="ISO-8859-1"?>
<SCENE SUBJECT="XML">
<OBJET id="1" nom="boule">
<GEOMETRIE type="sphere">
…
```



Le langage XSLT

- Issus du langage XPATH
- Définir une règle associée à une balise
  - <xsl:template match="pattern"> ... </xsl:template>

```
<xsl:template match="VEC3D">
<xsl:value-of select="@type" />=(<xsl:value-of select="X"
/>,<xsl:value-of select="Y" />,<xsl:value-of select="Z"/>)
</xsl:template>
```

Le langage XSLT

- Exécuter une règle
  - <xsl:apply-template>
    - Relance récursivement l'application des règles
    - Option : select permet de filtrer le flux d'entrée

```
<xsl:template match="GEOMETRIE">
<i><xsl:value-of select="@type" /></i>
<xsl:apply-templates select="VALEUR"/>
<xsl:apply-templates select="VEC3D"/>
</xsl:template>
```

Le langage XSLT

- Afficher le contenu d'une balise
  - <xsl:value-of select="expression"/>
    - Option : disable-output-escaping="yes|no"
      - ✓ Mode d'affichage du caractère '<'

```
// affichage de l'attribut type
<xsl:value-of select="@type" />
// affichage du contenu de la balise VALEUR
<xsl:apply-templates select="VALEUR"/>
```

Le langage XSLT

- Réaliser une boucle
  - <xsl:for-each select="pattern">
    instructions...
    </xsl:for-each>
    - Répète instructions pour toutes les instances de « pattern » disponibles dans le flux courant

```
<xsl:for-each select="SCENE/OBJET" >
  <TR>
    <TD>[<xsl:value-of select="@id"/></TD>
    <TD><xsl:apply-templates select="GEOMETRIE"/></TD>
    <TD><xsl:apply-templates select="RADIOMETRIE"/></TD>
  </TR>
</xsl:for-each>
```

Le langage XSLT

- Instructions conditionnelles
  - <xsl:if test="condition"> Instructions... </xsl:if>
  - <xsl:choose>
    <xsl:when test="condition1"> instr1... </xsl:when>
    <xsl:when test="condition2"> instr2... </xsl:when> ...
    <xsl:otherwise> instr3... </xsl:otherwise>
    </xsl:choose>

Le langage XSLT

- Trier des balises
  - <xsl:sort
    select="pattern"
    lang="langue"
    data-type="text|number|nom"
    order="ascending|descending"
    case-order="upper-first|lower-first"/>
  - Uniquement dans les instructions <xsl:for-each> et <xsl:apply-template>

XML pour ses propres applications

- **Parser XML**
  - **Existe sur tous les systèmes**
    - **Libxml2 sous linux**
- **Exemple :**
  - **Description de scène 3D**

Réalisation de document multimédia

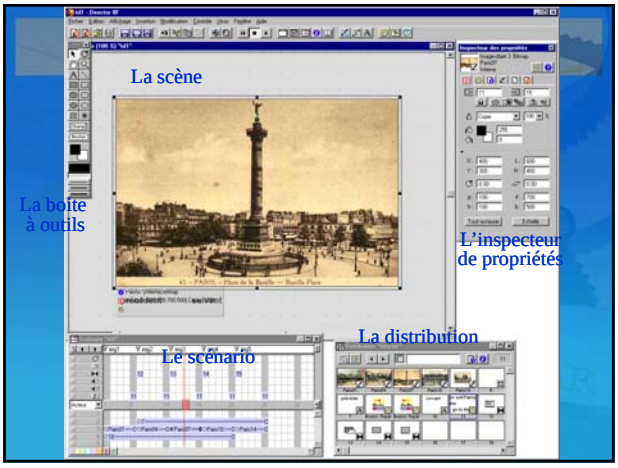
- **Un logiciel spécifique : Macromédia Director**
  - **structuration des média (les acteurs)**
    - **données générales**
      - ✓ **géométrie**
      - ✓ **instants d'apparition et de disparition**
      - ✓ **méthode d'insertion dans la scène (mélange des couleurs)**
    - **données spécifiques**
  - **organisation des acteurs en tenant compte du temps**
    - **placement géométrique**
      - ✓ **position,**
      - ✓ **orientation,**
      - ✓ **déformation**
    - **placement temporel**
      - ✓ **instant d'entrée et de sortie**
      - ✓ **informations géométriques et de couleurs pour plusieurs images clés**
      - ✓ **interpolation entre les images clés**

Macromédia Director

- **Gestion des média**
  - **incorporation d'un très grand nombre de format de média**
    - **images**
    - **images animées**
    - **séquences de films**
    - **sons**
  - **organisation des données en mémoire (les distribution)**
    - **chargement dynamique**
    - **optimisation des animations (chargement unique des média utilisés plusieurs fois)**
  - **exportation sous forme d'un fichier exécutable**
    - **MS Windows ou mac**
    - **prêt à graver pour créer un CD ROM**

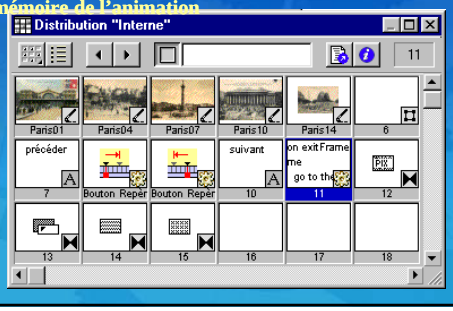
Macromédia Director

- **Interactivité très forte**
- **notion de comportement**
  - **rôle que l'on peut associer à un acteur**
  - **lié à un événement (souris, clavier…)**
  - **indépendant (mouvement aléatoire)**
  - **pilotage d 'un média de haut niveau (lecture, arrêt d 'une animation QuickTime, réglage du niveau sonore...)**
- **bibliothèque de comportement et langage de programmation (LINGO)**



La distribution

- **Regroupement des média utilisables dans l'animation**
  - **la mémoire de l'animation**



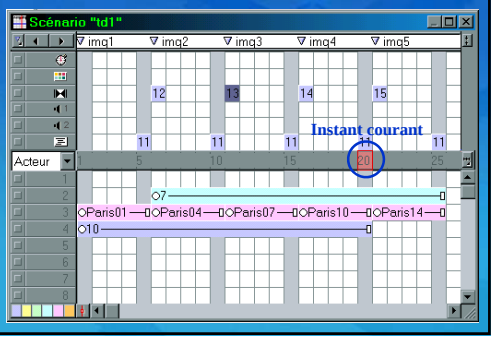
La scène

- **Partie visible de l'animation**
  - **interface avec l'utilisateur**
  - **organisation géométrique des acteurs**

Le scénario

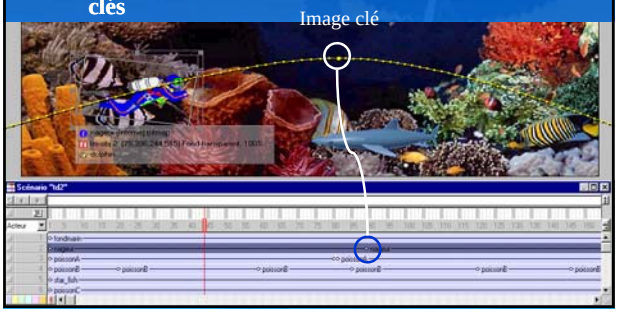
- **Organisation temporelle des acteurs**

Pistes
vitesse
palette
transition
son
script
Arrière plan
Profondeur des objets dans la scène
Premier plan



Le scénario

- **Interpolation des informations géométriques suivant une trajectoire dirigée par des images clés**



Les propriétés des acteurs

- **Propriétés générales**
  - **communes à tous les acteurs**
    - **géométriques,**
    - **temporelles,**
    - **colorimétriques**
  - **spécifiques au média**
    - **vitesse d'animation**
- **Comportements**
  - **associés à un événement**
  - **automatique**
  - **pilotage d'autres média**