Projet de Géolocalisation “Générique”
Phase 0.2 : sélection dans le fichier XML des objets choisis par l'internaute puis géolocalisation de ces objets en SVG sur le plan de la salle

Pierre Pompidor

Principe de la phase 0.2

La phase 0.2 met en oeuvre une feuille de style (plan_SVG.xsl) qui doit visualiser sur le plan les objets géographiques correspondants aux critères de sélection de l'internaute. Une première version de cette feuille de style va permettre de générer directement du code SVG. Elle sera exécutée par l'intermédiaire d'un script PHP relais appelé lors de la soumission du formulaire.

Script PHP relais

Le but principal du script PHP relais est de relayer les paramètres du formulaire à la feuille de style de géolocalisation :
formulaire → script PHP → feuille de style
(en effet la seconde feuille de style doit être paramétrée, et pour cela nous allons utiliser l'API XSLT de PHP).

Exemple :
L'internaute veut localiser toutes les **peintures** de **Picasso** se trouvant sur n'importe quel **mur** de n'importe quel **secteur** :
liste de paramètres (QUERY-STRING) émis par le formulaire :
`secteur-nom=&mur-nom=&oeuvre-type=peinture&oeuvre-auteur=Picasso`

```
header("Content-type:image/svg+xml");
...
$doc_xml = new DOMDocument();
$doc_xml->load('musee.xml');
...
$doc_xsl = new DOMDocument();
$doc_xsl->load('plan.xsl');
...
$proc = new XsltProcessor();
$proc->importStylesheet($doc_xsl);
...
$proc->setParameter(null, 'criteres', $parametres);
echo $proc->transformToXML($doc_xml);
```

Premier rôle de la feuille de style de géolocalisation : la construction de l'expression XPath

La feuille de style récupère dans un paramètre la liste des critères de recherche : `<xsl:param name="criteres" />` puis elle doit :
- parcourir la structure du document XML (que nous considérerons comme régulier (chaque étage de l'arbre est constitué de balises de mêmes noms))
- pour chaque balise, extraire de la liste des criteres ceux qui concernent cette balise et rajouter les conditions associées à l'expression XPath.

- analyse de la balise `secteur` : pas de critère correspondant → `//secteur`
- analyse de la balise `mur` : pas de critère correspondant → `//secteur/mur`
- analyse de la balise `oeuvre` : deux critères correspondants :
  → l'expression XPath devient : `//secteur/mur/oeuvre[@type=peinture][@auteur=Picasso]`

Attention :
Pour exploiter l'expression XPath ainsi construite, il faut utiliser une fonction `evaluate` qui permet d'évaluer une expression XPath contenue dans une variable (une fonction similaire mais moins souvent implémentée est `node-set`.
Cf l'exemple ci-dessous.

Second rôle de la feuille de style de géolocalisation : la génération du SVG

Puis la feuille de style doit :
- afficher l'image du plan de fond (que nous appellerons par exemple plan.jpg)
- afficher par dessus autant de carrés de couleur que d'objets géographiques à localiser (correspondants ici aux oeuvres).
  Je ne suis pas contre d'autres effets graphiques (bulles qui apparaîtraient lors du survol d'un carré pour donner toutes les informations liées à l'oeuvre...)

Fragments de code SVG :

```
<xsl:output method="xml" ... /> <!-- XML car nous générons du SVG -->

<xsl:template match="/">
  <svg xmlns="http://www.w3.org/2000/svg"
       xmlns:xlink="http://www.w3.org/1999/xlink"
       xmlns:dyn="http://exslt.org/dynamic>

    <!-- Appel du template-fonction qui va construire l'expression XPath -->
    <xsl:variable name="xpath">
        <xsl:call-template name="construit_xpath">
            <xsl:with-param name="criteres_a_analyser" select="$criteres" />
        </xsl:call-template >
    </xsl:variable>

    <image xlink:href="plan.jpg" x="..." y="..." width="..." height="..." />

    <xsl:for-each select="dyn:evaluate($xpath)">
       <xsl:rect x="{@x}" y="{@y}" width="..." height="..." style="..." />
   </xsl:for-each>

  </svg>
</xsl:template>

<xsl:template name="construit_xpath">
  A ECRIRE !!!
</xsl:template>
```

Vous remarquerez :
- que la feuille de style produit du xml (ici du code SVG) !
- la déclarations des deux espaces de noms dans la balise `svg` outrel'espace de nom par défaut (svg) :
  - **xlink** : qui permet la liaison de ressources externes
  - **dyn** qui permet l'utilisation de fonction étendue **evaluate**
- que le template-fonction construit_xpath va devoir extraire pour chaque balise différente du document XML, les critères de recherche correspondants avec les fonctions suivantes :
  - substring-before(string chaîne, string sous-chaîne) : retourne la chaîne avant la première occurrence de chaîne : exemple : `<xsl:value-of select="substring-before($criteres,':')" />`
  - substring-after(string chaîne, string sous-chaîne) : retourne la chaîne après la première occurrence de chaîne.