

## Rapport Technologies XML et Technologies WEB

### Sujet

# Les évènements organisés par les Clubs de l'EMI

**Réalisé par :**

ELMIMOUNI Mohamed  
EL ANNAOUI Imad  
LACHHAB Anas

**Sous l'intitulé de :**

Mme. EL KASSIRI Asmae

# Table de matière

I.	Introduction .....	3
II.	Technologies XML .....	3
1.	Définitions de quelques notions .....	3
2.	Analyse de projet.....	4
3.	Structure d'arbre.....	4
4.	Schéma XML .....	5
5.	DTD.....	8
6.	Document XML.....	9
7.	Langage XSLT .....	11
8.	Utilisation de XQuery .....	12
9.	Transformation en HTML.....	12
III.	Technologies WEB .....	14
1.	Aspect Web.....	14
a.	Réalisation de l'interface graphique.....	14
b.	Les balises de structuration.....	15
c.	Feuille de style CSS .....	17
d.	Les langues supportées.....	17
e.	Responsive design .....	18
2.	Aspect Application .....	19
a.	Installation du serveur web local Apache Tomcat .....	19
b.	Utilisation de JavaScript comme langage dynamique et DOM .....	21
c.	Hébergement du site web.....	21
IV.	Conclusion.....	23
V.	Webographie.....	23

## I. Introduction

Dans le cadre des cours technologies XML et technologies WEB, nous sommes amenés à réaliser une application web sur les événements organisés par les clubs de l'EMI en exploitant les notions vues dans ces deux cours.

Au début nous avons effectués une recherche sur les événements organisés par les clubs de l'EMI, ensuite nous avons structurés ces données sous forme d'arbre , après nous avons décrit la grammaire de notre document XML en utilisant les DTDs.

Ensuite nous avons transformés les données XML en une page web en utilisant la langage XSLT.

L'objectif de la partie XML est unifier les informations concernant les clubs et préparer les données que nous aurait eus besoins dans la partie technologies web.

Concernant la partie web, nous avons travaillé sur la page générée de langage XSLT en ajoutant des modifications au niveau de style et l'esthétique de la page et également l'ajout d'autres pages comme la page d'accueil.

Aussi nous avons inclus les balises de structurations pour bien structurer et organiser notre site, et intégrer deux langues (français et arabe) en donnant à l'utilisateur la possibilité de choisir.

Nous avons aussi inclus le principe de Responsive Design pour que notre site s'affiche bien dans toutes les dispositifs (PC, tablette, Smartphone, ...).

Encore nous avons utilisé JavaScript comme langage dynamique de notre site web, pour exploiter quelques données XML par DOM (car la plupart des données sont générées automatiquement par la langage XSLT).

Enfin nous avons déployés notre site en utilisant Apache Tomcat comme serveur local.

## II. Technologies XML

### 1. Définitions de quelques notions

**XML :** Est un langage d'échange de données structurés entre applications de différents systèmes d'informations. Les données d'un fichier XML sont organisée d'une manière hiérarchique.

**DTD :** Est une série de spécifications déterminant les règles structurelles des documents

XML qui lui sont associés.

**Schémas XML** : C'est aussi une série de spécifications déterminant les règles structurelles des documents XML comme le DTD mais avec plus de fonctionnalités (typage des données, héritage, ...).

**XSLT** : *eXtensive Stylesheet Language* : est un langage de feuille de style. Il est aussi un très puissant manipulateur d'éléments. Il permet de transformer un document XML source en un autre.

**XQuery** : Est un langage non XML, permet de traiter des ressources XML (fichier ou SGBD-XML) pour obtenir des structures XML.

**DOM** : (Document Object Model) : permet d'accéder et d'agir d'une manière directe sur le contenu et la structure de l'arbre XML.

## 2. Analyse de projet

Le projet se divise en deux parties la partie technologies XML et la partie Technologie WEB.

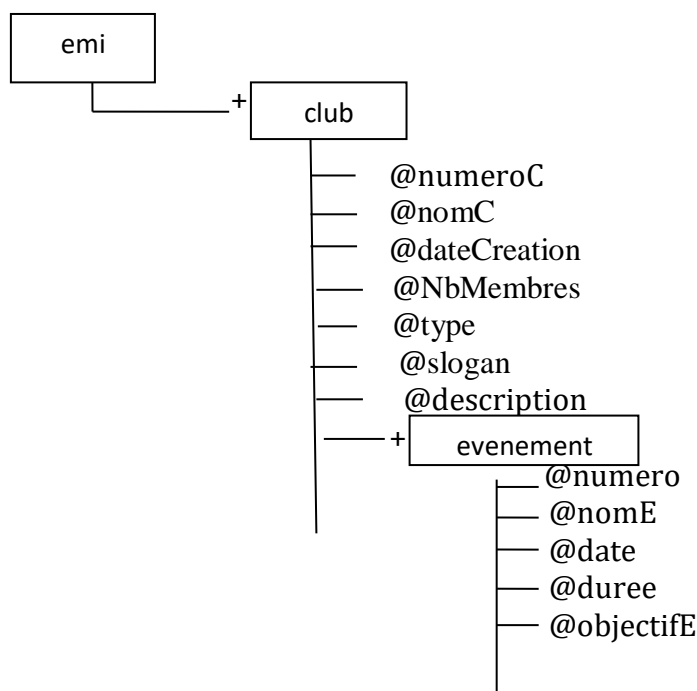
Il consiste à la mise en place d'un site web qui vise à présenter les clubs de l'école en citant les événements organisés par ces clubs de façon bien organisée.

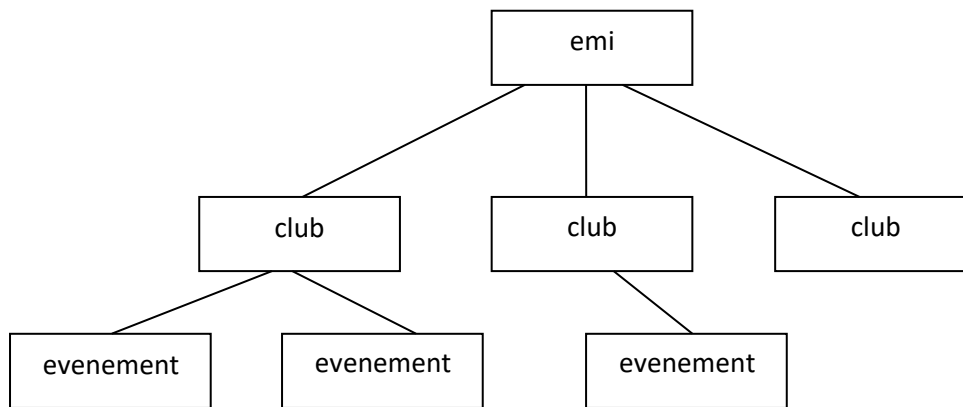
Nous avons mis les données dans un fichier XML et les exploitons par XSLT et DOM.

Concernant les données nous les avons recueillies à partir de sources fiables (Les pages officielles des clubs dans les réseaux sociaux, site officiel de l'EMI, ...).

## 3. Structure d'arbre

Nous avons nommé la racine de notre fichier XML 'emi'.





#### 4. Schéma XML

Pour déterminer les règles structurelles de notre fichier nous avons utilisés le schéma XML et le DTD.

Mais c'est le DTD que nous avons associés avec notre fichier XML (le schéma XML est juste une addition).

Voici un extrait du code 'schéma XML' sachant qu'on a utilisés [Oxygen XML Editor 19.0](#) pour créer les codes de tous les fichier XML :

```

• EmiEvents.xsd x
Utilisation de Schéma XML 1.0 pour le fichier courant.
En savoir plus
Modifier la version

xs:schema xs:element
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
3   <xs:element name="emi">
4     <xs:complexType>
5       <xs:sequence>
6         <xs:element name="club" maxOccurs="unbounded" type="TypeClub"/>
7       </xs:sequence>
8     </xs:complexType>
9   </xs:element>
10  <xs:complexType name="TypeClub">
11    <xs:sequence>
12      <xs:element name="evenement" maxOccurs="unbounded" type="TypeEvent"/>
13    </xs:sequence>
14    <xs:attribute name="numeroC" type="xs:NMTOKEN" use="required"/>
15    <xs:attribute name="nomC" type="xs:string" use="required"/>
16    <xs:attribute name="dateCreation" type="xs:string" use="required"/>
17    <xs:attribute name="NbMembres" type="xs:int" use="required"/>
18    <xs:attribute name="type" type="xs:string" use="required"/>
19    <xs:attribute name="slogan" type="xs:string" use="required"/>
20    <xs:attribute name="description" type="xs:string" use="required"/>
21  </xs:complexType>
22  <xs:complexType name="TypeEvent">
23    <xs:attribute name="numero" type="xs:NMTOKEN" use="required"/>
24    <xs:attribute name="nomE" type="xs:string" use="required"/>
25    <xs:attribute name="date" type="xs:string" use="required"/>
26    <xs:attribute name="duree" type="xs:int" use="required"/>
27    <xs:attribute name="objectifE" type="xs:string" use="required"/>

```

- Un Schéma XML commence par un prologue, et a un élément racine.

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

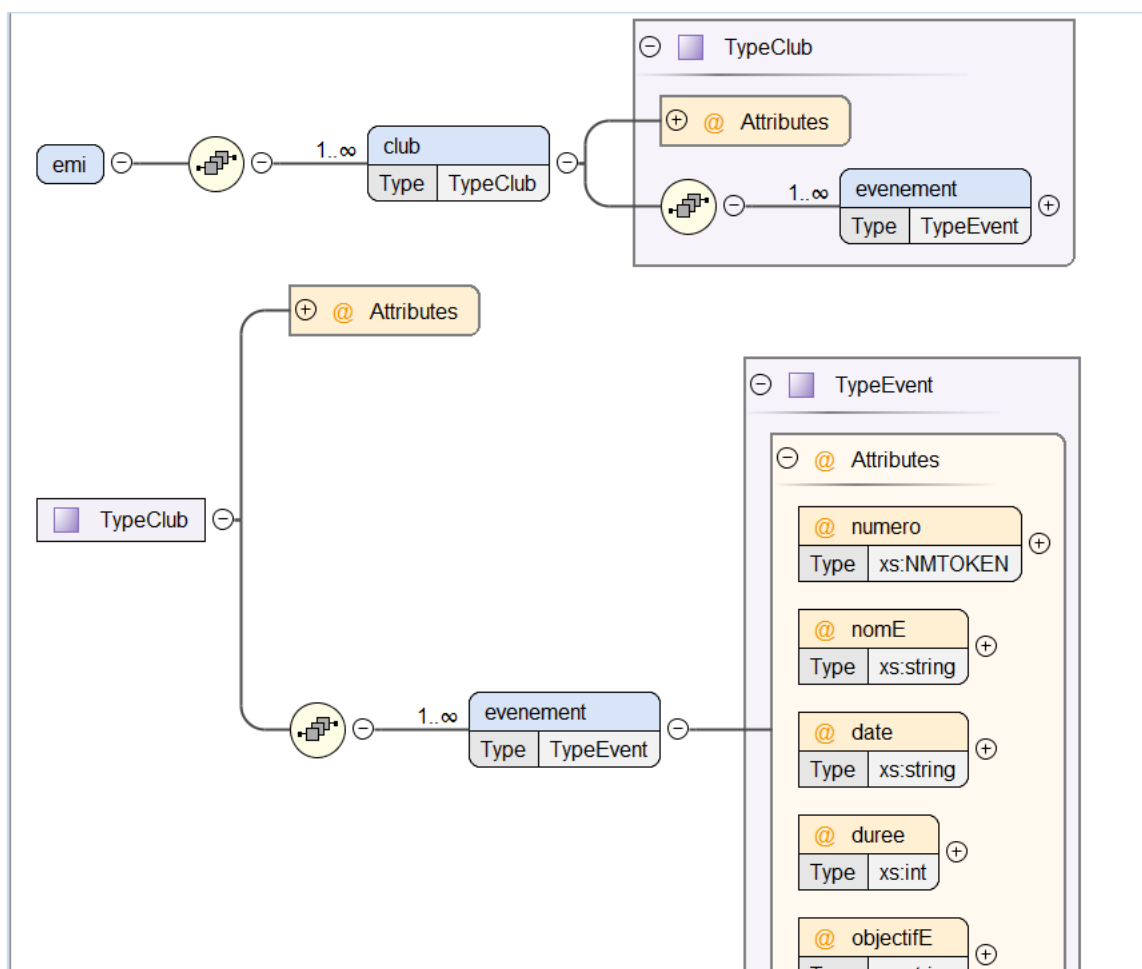
```
<xs:schema
```

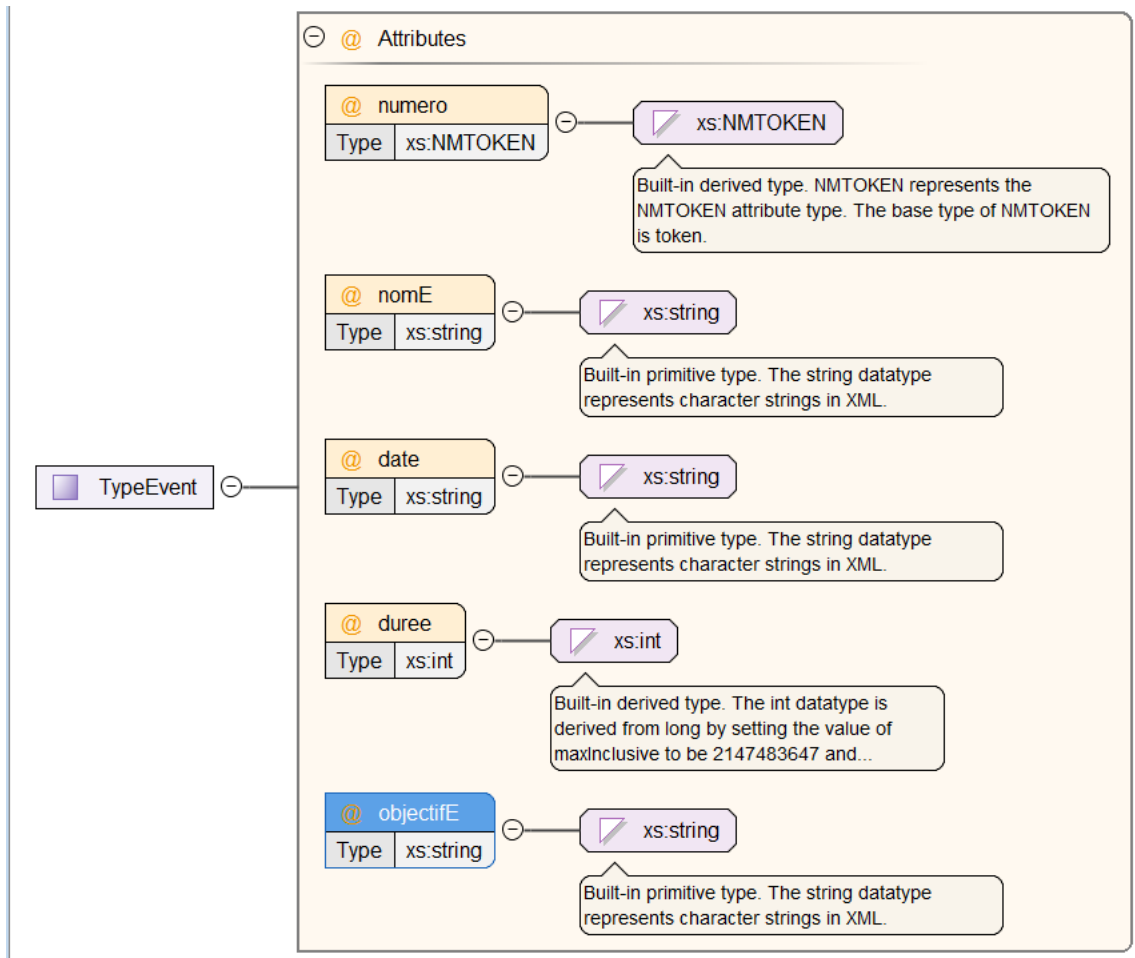
```
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
```

```
<!-- déclarations d'éléments, d'attributs et de types ici -->
```

```
</xs:schema>
```

- L'élément racine est l'élément **xs:schema**.
- Les éléments du XML schéma sont préfixés par xs. : Cela signifie que tous les éléments de XML schéma commencent par le préfixe xs (<xs:element>).
- Un élément, dans un schéma, se déclare avec la balise **<xs:element>**, il peut être de type simple ou de type complexe.
- Un attribut, dans un schéma, se déclare avec la balise **<xsd:attribute>**, et il ne peut être que de type simple.
- On peut aussi afficher le Schéma XML sous forme graphique (click sur 'Design'):





## 5. DTD

Nous avons choisi le DTD pour l'association avec le fichier XML car nous avons le trouvés plus simple que le schéma XML.

Voici un extrait du code 'DTD' :

```

• EmiEvents.dtd x
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!ELEMENT emi (club+) >
3 <!ELEMENT club (evenement*) >
4 <!ELEMENT evenement EMPTY >
5 <!-->
6     numeroC NMTOKEN #REQUIRED
7     nomC CDATA #REQUIRED
8     dateCreation CDATA #REQUIRED
9     NbMembres CDATA #REQUIRED
10    type CDATA #REQUIRED
11    slogan CDATA #REQUIRED
12    description CDATA #REQUIRED>
13 <!-->
14    numero NMTOKEN #REQUIRED
15    nomE CDATA #REQUIRED
16    date CDATA #REQUIRED
17    duree CDATA #REQUIRED
18    objectifE CDATA #REQUIRED>
19

```

Un DTD spécifie :

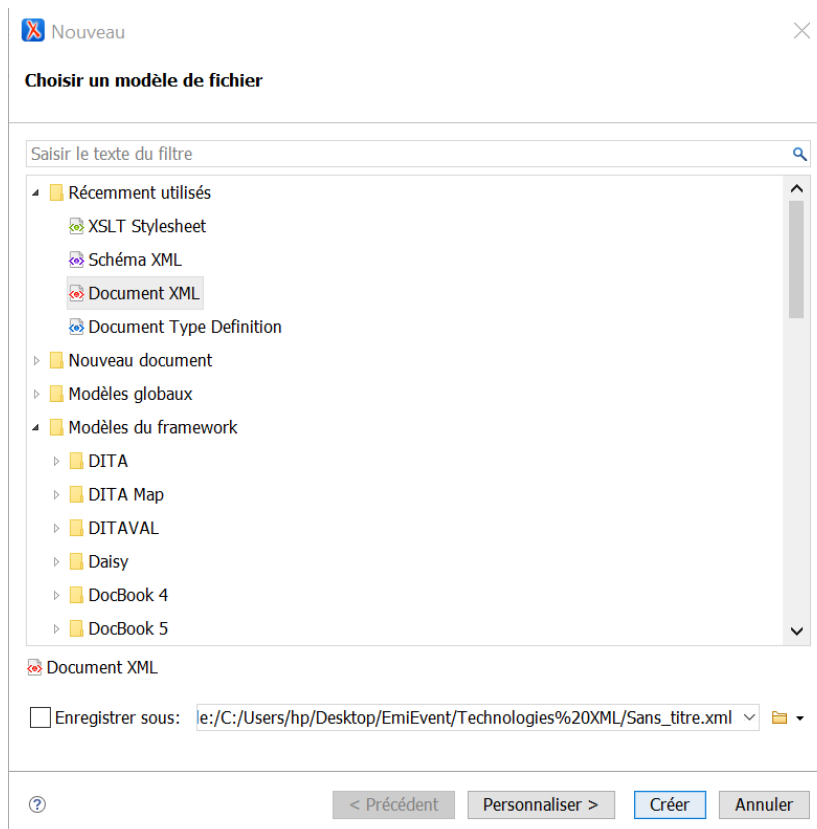
- ❖ Le nom des balises associées à tous les éléments.
  - ❖ Pour chaque balise, les attributs possibles et leur type.
  - ❖ Les relations contenant-contenu entre les éléments et leur cardinalité.
- 
- La déclaration d'une nouvelle balise se fait grâce à l'instruction ELEMENT. La syntaxe est:  
`<!ELEMENT nom contenu>.`
  - `club+` : Indique que club peut apparaître une ou plusieurs fois.
  - `evenement*` : Indique l'élément peut apparaître zéro, une ou plusieurs fois.
  - Le mot EMPTY permet de définir une balise vide `<!ELEMENT evenement EMPTY`  
`>`
  - La syntaxe de déclaration d'un attribut est :  
`<!ATTLIST nom-balise nom-attribut type-attribut présence >`
  - Les types d'attributs les plus courants sont:
    - Type `CDATA` : signifie que la valeur de l'attribut doit être une chaîne de caractères.
    - Type `NMTOKEN` : signifie que la valeur de l'attribut doit être une chaîne de caractères ne contenant pas d'espaces et de caractères spéciaux.
  - Le terme Présence permet de définir comment doit être gérée la valeur de l'attribut. Il existe 4 types de présences parmi les types on trouve:  
`#REQUIRED` Indique que l'attribut doit être présent dans la balise et que sa valeur doit être obligatoirement spécifiée.

## 6. Document XML

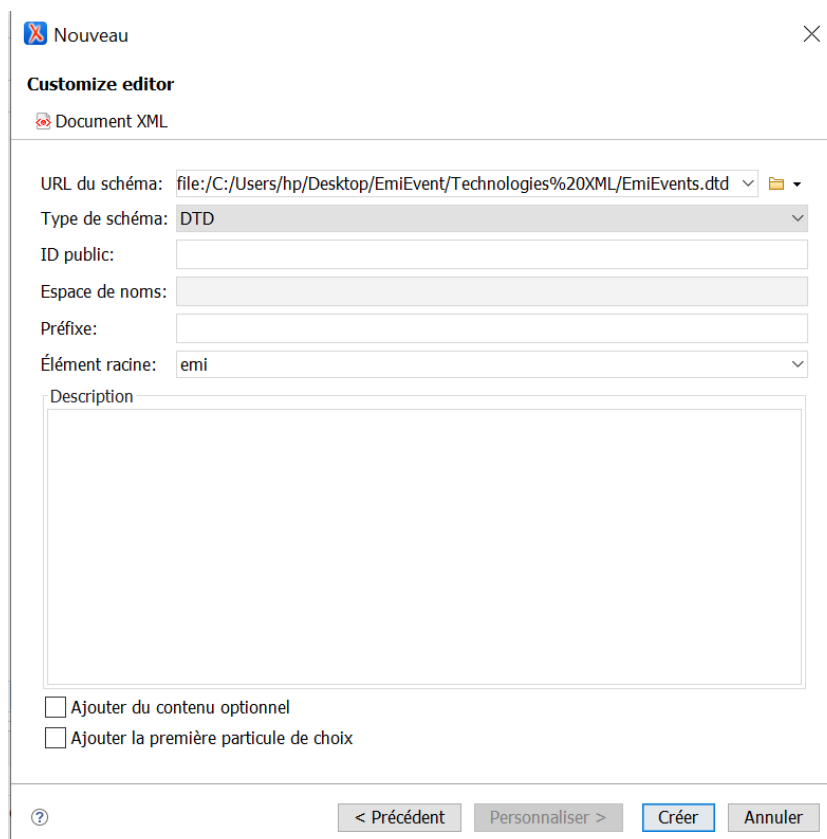
Pour créer un fichier XML qui respecte une grammaire, voici les étapes qu'il faut suivre :

- i. Aller dans `fichier` puis `Nouveau`.
- ii. Sélectionner `Document XML` puis `Personnaliser >` :





Mettre URL de la grammaire dans **URL du schéma** et la racine dans **Élément racine** puis **Créer** :



Après un fichier XML qui va être généré :

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE emi SYSTEM "file:/C:/Users/hp/Desktop/EmiEvent/Technologies%20XML/EmiEvents.dtd">
3 <emi>
4   <club numeroC="" nomC="" dateCreation="" NbMembres="" type="" slogan="" description=""></club>
5 </emi>
6
7
8
9
10
11
12
13
14
15
16
17
18
19

```

La valeur de l'attribut "" de type NMTOKEN doit être un NMTOKEN

Il donne une erreur car l'attribut numeroC ne respecte pas le type NMTOKEN.

La **déclaration de type de document** (DTD) permet de définir la structure logique du document et sa validité.

```
<!DOCTYPE emi SYSTEM "file:/C:/Users/hp/Desktop/EmiEvent/Technologies%20XML/EmiEvents.dtd">
```

Après en remplit le fichier par nos données :

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE emi SYSTEM "file:/C:/Users/hp/Desktop/EmiEvent/Technologies XML/EmiEvents.dtd">
3 <emi>
4   <club numeroC="1" nomC="EmiSport" dateCreation="2012" NbMembres="60" type="Loisir et renco
5     <evenement numero="1" nomE="Olympiades sportives" date="6 Mars" duree="3" objectifE="Lo
6     <evenement numero="2" nomE="mini journée" date="25 Novembre" duree="1" objectifE="Loisi
7     <evenement numero="3" nomE="tounoi de l'école" date="toute l'année" duree="toute l'anné
8   </club>
9   <club numeroC="2" nomC="Forum Emi-Entreprises" dateCreation="1994" NbMembres="28" type="Or
10     <evenement numero="1" nomE="Le forum" date="1 Avril" duree="2" objectifE="Faciliter la
11     <evenement numero="2" nomE="Gala" date="03 Avril" duree="1" objectifE="Loisir" />
12   </club>
13   <club numeroC="3" nomC="EmiKhayr" dateCreation="2012" NbMembres="60" type="Club social" sl
14     <evenement numero="1" nomE="Le caravane" date="28 Mars" duree="1" objectifE="Aider les
15     <evenement numero="2" nomE="Ftour" date="un jour de ramadan" duree="1" objectifE="Distr
16     <evenement numero="3" nomE="Achouraa" date="Le jour d'achouraa" duree="1" objectifE="So
17     <evenement numero="4" nomE="Paraolympique" date="06 Mars" duree="01" objectifE="Socio-s
18   </club>
19   <club numeroC="4" nomC="EMInence" dateCreation="1991" NbMembres="20" type="Club culturel"
20   ...
21 </club>
22 </emi>

```

Un fichier XML se compose : des balises(ex : <club>), des éléments(ex : EmiSport) et des attributs(ex : nomC).

Pour qu'un document XML soit bien formé il doit obéir à 4 règles :

- Un document XML ne doit posséder qu'une seule racine
- Tous ces éléments doivent être fermés.
- Les éléments contenus et contenant doivent être imbriqués.
- Les valeurs des attributs s'écrivent entre guillemets.

## 7. Langage XSLT

XSL est un outil privilégié de production de fichiers HTML à partir de sources XML.

Un fichier XSL étant un fichier XML, il doit respecter les normes de syntaxe de ce format.

Voici un extrait du code 'EmiEvents.xsl' :

```
<xsl:for-each select="@emi/club">
  <section class="container">
    <div class="card">
      <div class="card-header">
        <h4><strong><xsl:value-of select="@nomC"/></strong></h4>
      </div>
      <div class="card-body">
        <div class="group">
          <div class="float-left">
            <label><strong>Date de création : </strong></label> <xsl:value-of select="@dateCreation"/><hr/>
            <label><strong>Le nombre des membres : </strong></label> <xsl:value-of select="@NbMembres"/><hr/>
            <label><strong>Type : </strong></label><xsl:value-of select="@type"/><hr/>
            <label><strong>Slogan : </strong></label><i><xsl:value-of select="@slogan"/></i><hr/>
            <label><strong>Nombre de ses événements : </strong></label> <xsl:value-of select="count(eventement)"/>
            <label><strong>Les événements : </strong></label><p></p>
          </div>
        </div>
        <table class="table table-hover">
          <thead>
            <tr>
              <th scope="col">
                Nom
```

Sélection d'un élément:

- Syntaxe: **<xsl:value-of select="nom\_element" />**
- Il est également possible de "naviguer" dans les branches de l'arborescence du document XML, en utilisant les ressources du DOM.

**<xsl:for-each>**

- Crée une boucle dans laquelle sont appliquées des transformations.

**<xsl:value-of>**

- Cet élément permet d'insérer la valeur d'un nœud dans la transformation.

## 8. Utilisation de XQuery

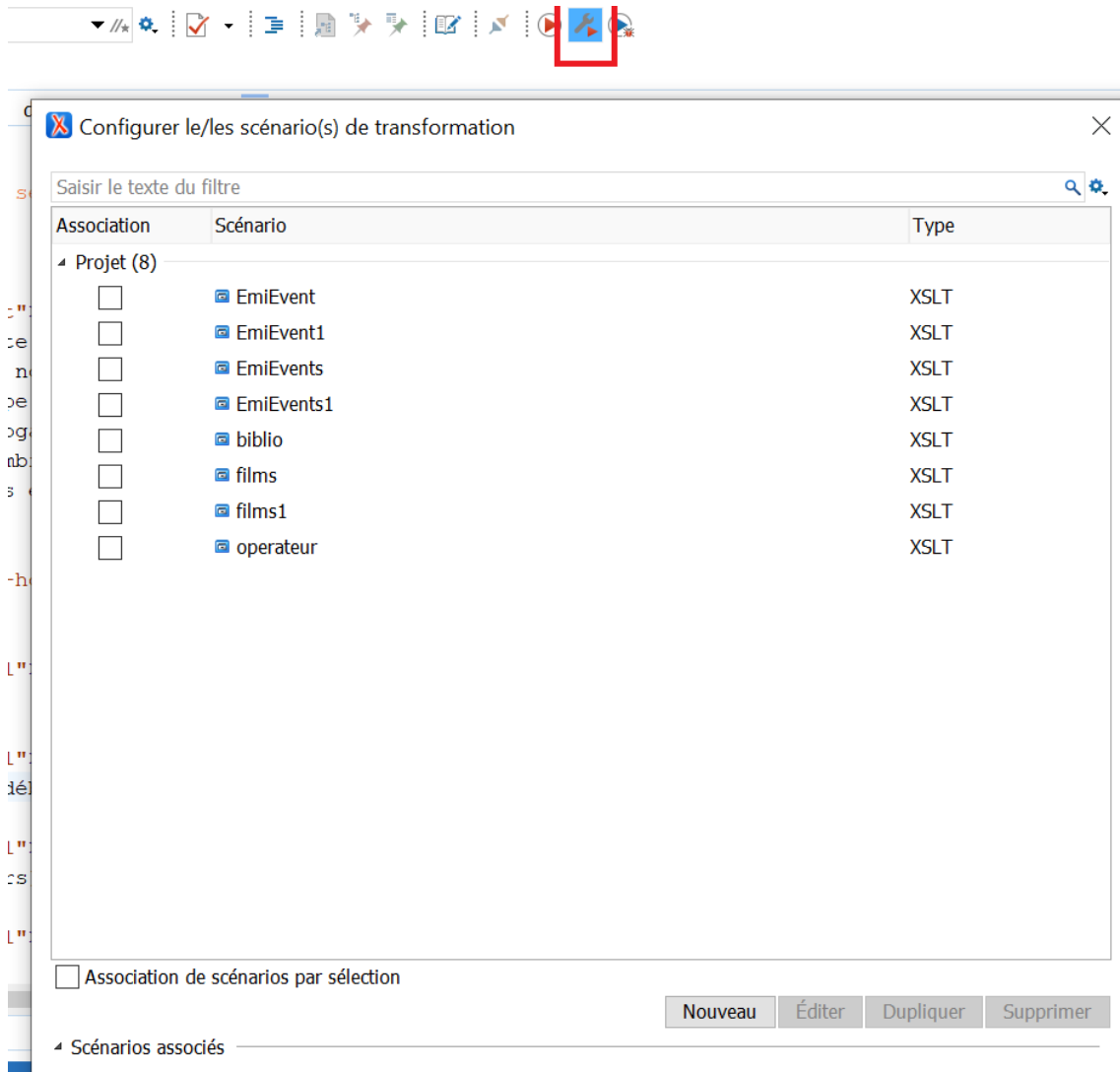
Durant l'établissement du projet nous n'avons pas besoin de XQuery qu'une seule fois (pour calculer le nombre des événements d'un club en utilisant count).

```
<label><strong>Nombre de ses événements : </strong></label> <xsl:value-of select="count(eventement)"/><hr/>
```

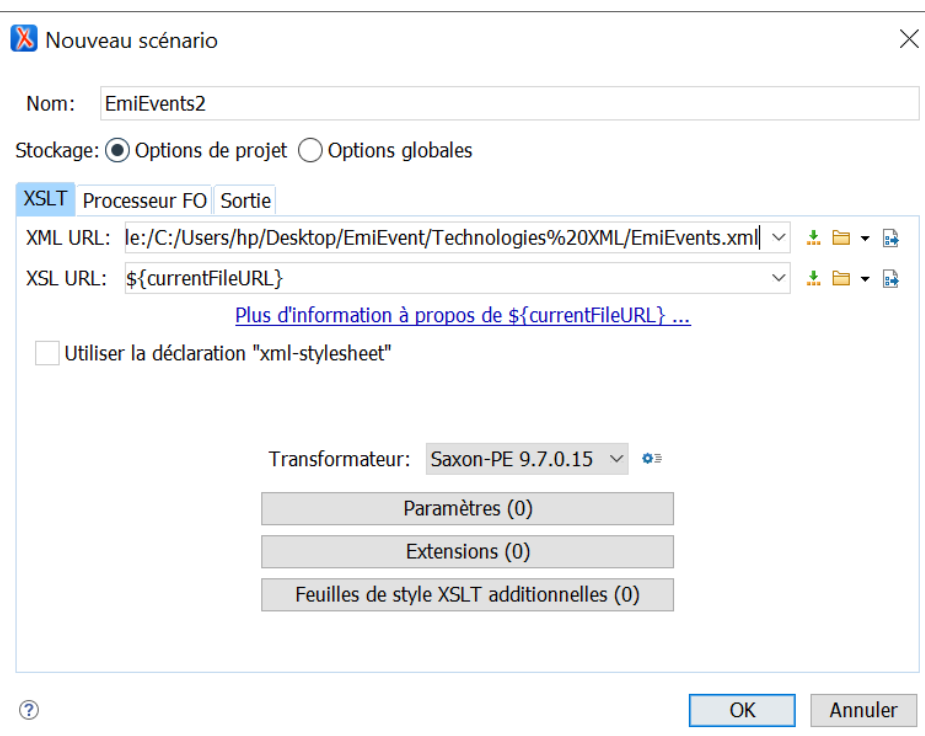
## 9. Transformation en HTML

Voici les étapes pour transformer un fichier .xsl en un fichier.html :

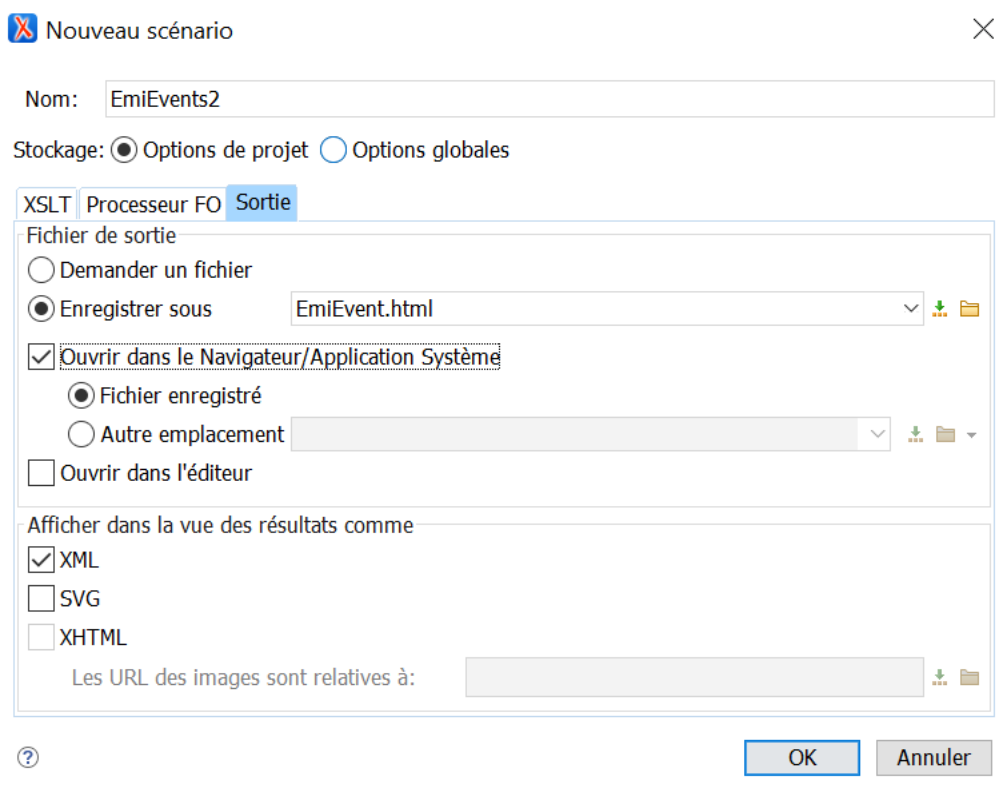
- Click sur l'icône entouré par le rouge puis choisir **Nouveau** et **XSLT transformation**



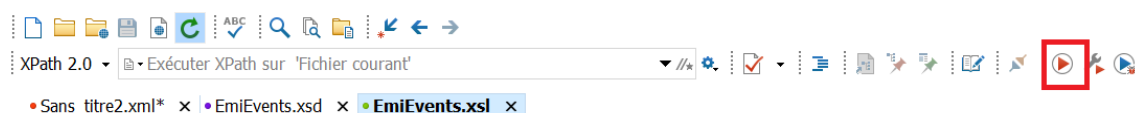
ii. Dans **XSLT /XML URL** : saisir l'URL du fichier XML puis click sur Sortie.



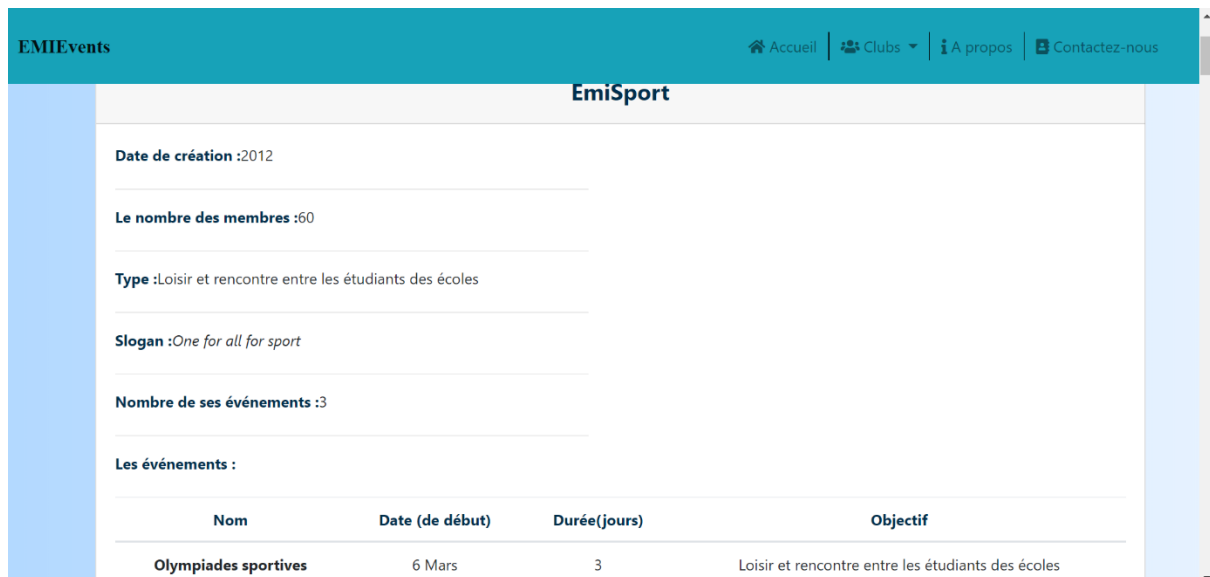
- iii. Dans **Enregistrer sous** : saisir le nom du fichier terminé par .html, puis cocher l'option '**Ouvrir dans le Navigateur**' et **OK**:



Pour afficher le résultat dans le navigateur il suffit de cliquer sur l'icône encadré ci-dessous :



Voici une partie du page généré qui contient les données d'un club :



Pour le style nous avons associés une feuille de style css à notre fichier xslt et utilisés le Framework Bootstrap et pour les icônes nous avons intégrés la bibliothèque 'font awesome' dans le fichier.

### III. Technologies de web

#### 1. Aspect Web

##### a) Réalisation de l'interface graphique

L'interface graphique de la page principale de notre site web est fait dans le fichier xslt, après la transformation du fichier nous avons obtenu un fichier html bien structuré. Dans la suite du projet nous avons travaillé sur ce fichier généré (EmiEvent.html) et créés deux autres fichiers html (index.html et EmiEvent-arabe.html).

##### Fichier index.html :

C'est la page d'accueil de notre site web, il contient des images de quelques clubs et de ses événements, les services fournis par le site et des informations à propos l'application.

##### b) Les balises de structuration

Les trois pages ont le même header et le même footer, la navigation entre ces pages se fait de façon simple en cliquant sur les liens du site web.

**Header** : ou en-tête / entête d'un fichier informatique ou d'un paquet transitant sur un réseau informatique, contient les données présentes au début de ce fichier ou du paquet.

**nav** : L'élément HTML `<nav>` représente une section d'une page ayant des liens vers

d'autres pages ou des fragments de cette page. Autrement dit, c'est une section destinée à la navigation dans un document (avec des menus, des tables des matières, des index, etc.).

Le header des pages EmiEvent.html et EmiEvent-arabe.html contient seulement un nav.



En plus de nav le header de la page [index.html](#) contient aussi le carrousel (class de Bootstrap).

Le carrousel est un diaporama pour parcourir une série de contenu, construit avec des transformations CSS 3D et un peu de JavaScript.



**Section :** L'élément HTML `<section>` représente une section générique d'un document, par exemple un groupe de contenu thématique. Une section commence généralement avec un titre.

Nous avons utilisé les sections pour regrouper les informations de chaque club :

**Forum Emi-Entreprises**

**Date de création :**1994

---

**Le nombre des membres :**28



---

**Type :**Organisateur de la forum

---

**Slogan :**Sans

---

**Réseaux sociaux :**  


---

**Nombre de ses événements :**2

---

**Les événements :**


Nom	Date (de début)	Durée(jours)	Objectif



Logo Forum Emi-Entreprises

**Article** : Section de contenu indépendante, pouvant être extraite individuellement du document ou syndiquée, sans pénaliser sa compréhension.

Nous avons utilisé article dans la page index.html :




**Services**

Découvrez avec nous tous les informations que vous voulez connaître sur les clubs de l'EMI.



**Actualités**

Notre plateforme offre toutes les actualités des clubs.



**A propos**

Notre plateforme permet de présenter toutes les informations nécessaires sur les clubs de EMI et ses événements

**Aside** : Section dont le contenu est un complément par rapport à ce qui l'entoure, qui n'est pas forcément en lien direct avec le contenu mais qui peut apporter des informations supplémentaires.

Nous avons utilisé aside dans la page des clubs pour montrer le changement des dates :

**⚠ Note:**

*Les dates des événements changent d'année en année et nous n'avons que des dates approximatives.*

Nous avons utilisé plusieurs balise HTML comme :



table, thead,tbody, ul ...

### c) Feuille de style CSS

Pour que notre site web soit bien organisé, nous avons associés aux pages html une feuille de style personnelle (EmiEvents.css)

Voici un extrait du code css :

```

1  body {
2      background: #3385ff;
3      background: linear-gradient(to right, #b3d9ff, #e6f2ff);
4  }
5
6
7  #mainNav
8  {
9      padding: 15px;
10     color: white;
11     height: 80;
12 }
13
14 #mainNav.navbar-scrolled {
15     background-color: #dedede;
16 }
17 .navbar-brand
18 {
19     font-family: "times new roman", times, serif;
20 }
21 #navbarNav{
22     margin-left: 300px;
23 }
```

### d) Les langues supportées

Les langues supportées par le site web sont le français et l'arabe. Le changement de la langue se fait de façon simple en cliquant sur la langue voulue dans une liste déroulante :

Choisir la langue:
 

Français ▼

اختر اللغة:
 

عربي ▼

Les deux langues sont supportées par une seule page (EmiEvent.html) par contre la

page index.html ne supporte que la langue française.

Voici une section en langue française :

**EmiKhayr**

**Date de création :**2012

---

**Le nombre des membres :**60



---

**Type :**Club social

---

**Slogan :**United we are

---

**Réseaux sociaux :**  


---

**Nombre de ses événements :**4

---

**Les événements :**

Nom	Date (de début)	Durée(jours)	Objectif
Le caravane	28 Mars	1	Aider les gens dans les villages oubliés



Logo Emikhayr

La même section en langue arabe :

**إيمي الخير**

**تاريخ الإنشاء :**2012

---

**عدد الأعضاء :**60



---

**النوع :**نادي اجتماعي

---

**متحدون نحن :**الشعار الخطي

---


**الشبكات الاجتماعية :**  

---

**عدد فعالياته :**4

---

**الفعاليات :**



شعار

### e) Responsive design

Le Responsive Web Design consiste à utiliser HTML et CSS pour redimensionner, masquer, rétrécir ou agrandir automatiquement un site Web afin de le rendre agréable sur tous les appareils (ordinateurs de bureau, tablettes et téléphones)

Il est nécessaire car une page Web devrait bien paraître sur n'importe quel appareil.

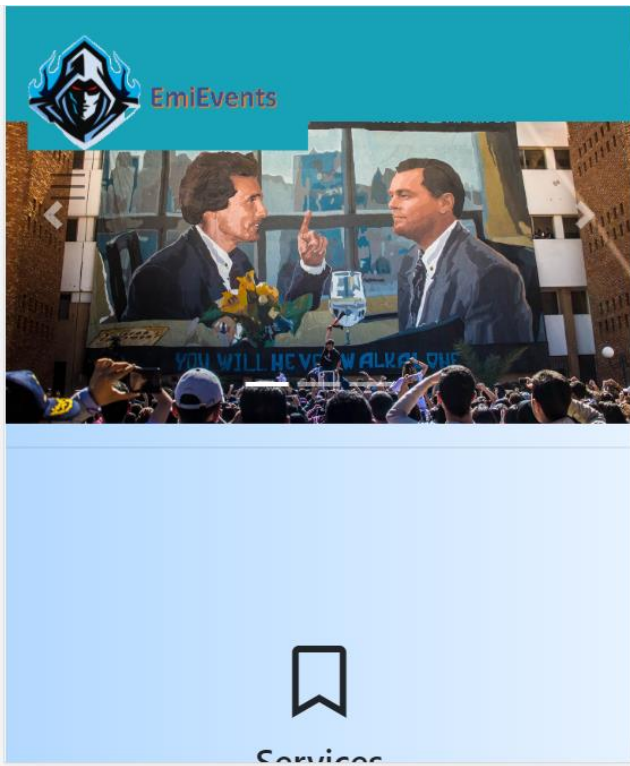
Pour ce faire on doit ajoutez l' **<meta>** élément suivant dans toutes nos pages Web:

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

C'est ce que nous avons faites dans le site web pour les 3 pages :

```
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```

En changeant la dimension d'une page dans le navigateur, on obtient :



## 1. Aspect Application

### a) Installation du serveur web local Apache Tomcat

Apache Tomcat est serveur d'application libre.

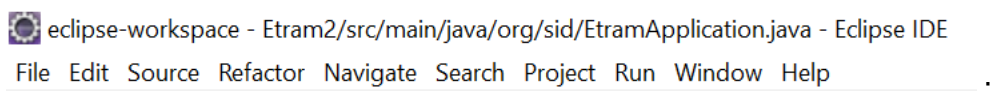
Nous avons le choisi pour déployer notre site web.



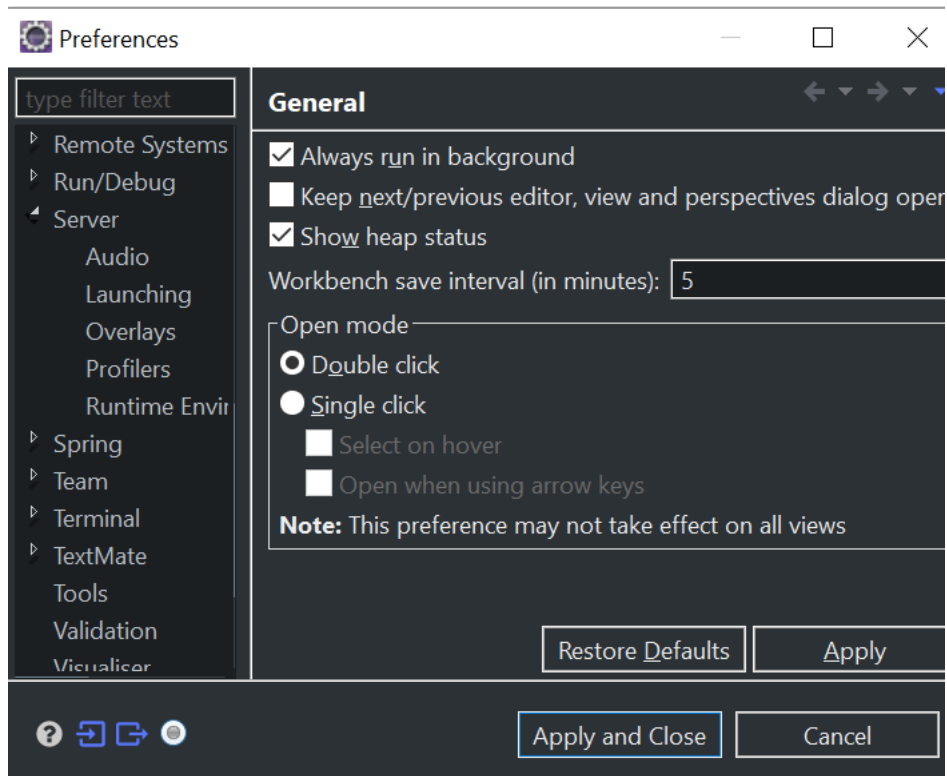
Nous avons le téléchargés l'installés puis l'intégrés avec eclipse.

Voici les étapes de l'intégration avec eclipse

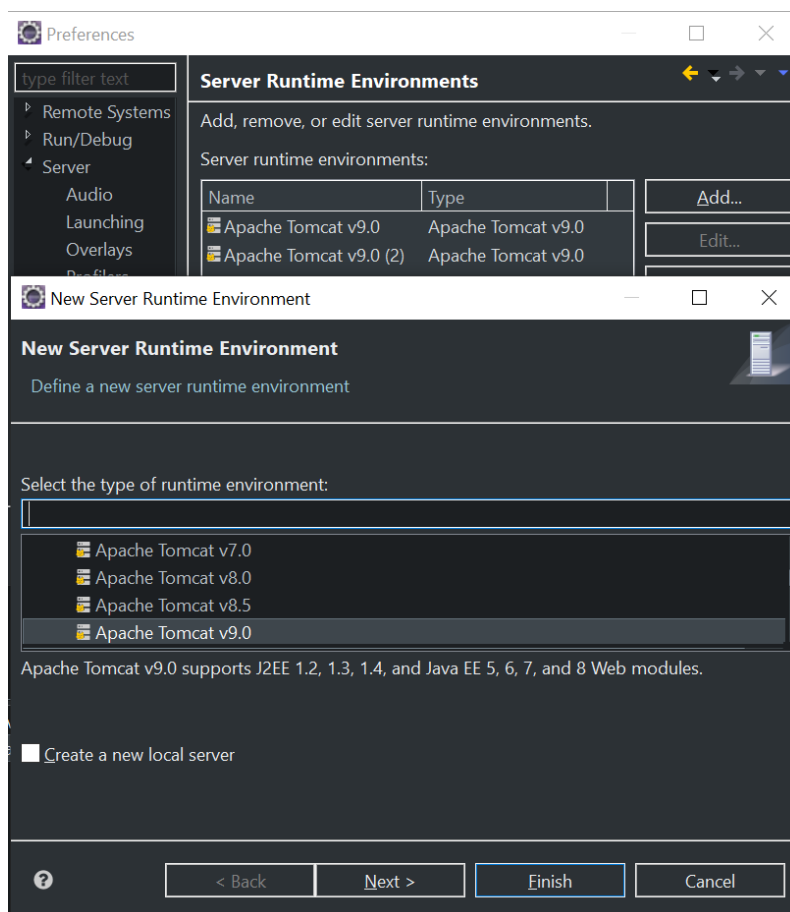
- i. Aller dans **Window** et **Preferences** :



ii. Aller dans **Server** et **Runtime Environments** :



iii. Choisir Apache Tomcat téléchargé(en tenant compte la version) en cliquant sur **Add** puis **Finish** et **Apply and Close** :



### b) Utilisation de JavaScript comme langage dynamique et DOM

Nous avons aussi utilisé le JavaScript avec DOM comme Parser XML pour obtenir quelques données (la description de chaque club) sachant que la plupart des données sont obtenus par la langage XSLT .

Voici un extrait du code JavaScript :

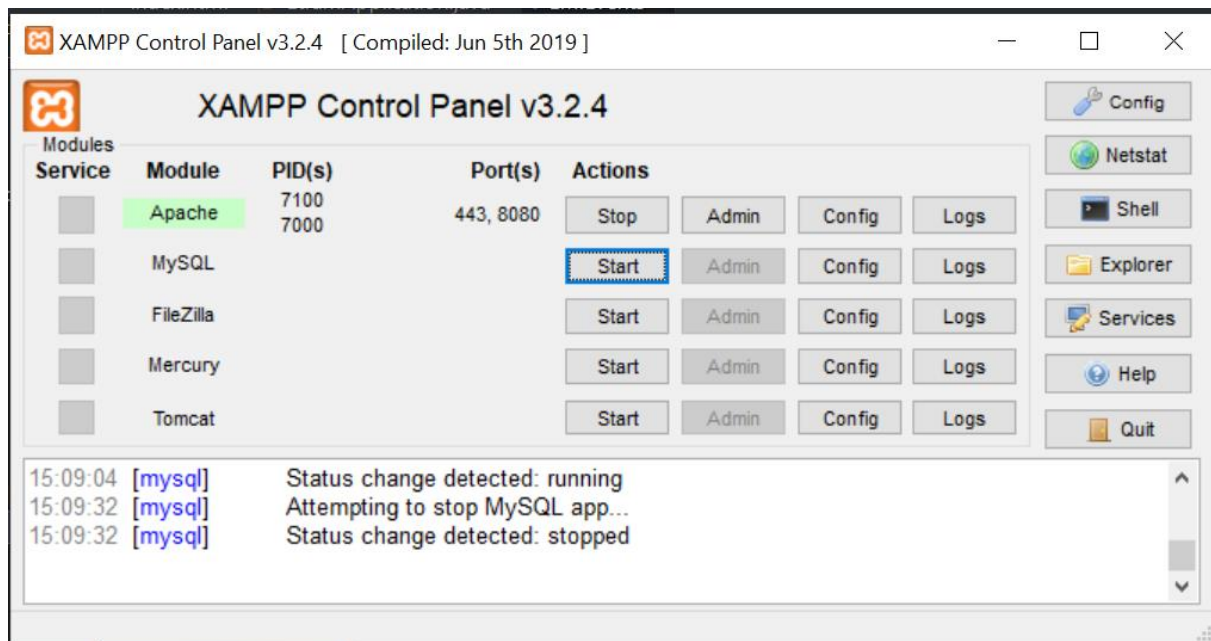
```
1  var xhttp = new XMLHttpRequest();
2  xhttp.open("GET","EmiEvents.xml",true);
3  xhttp.send();
4  function display1(){
5
6      var xm = document.getElementById("xm");
7      var x = xm.getElementsByTagName("club")[0];
8      var y = x.getAttributeNode("description");
9      var txt = y.nodeValue;
10
11      alert("Description : " + txt);
12  }
```

Résultat dans le navigateur en cliquant sur EmiSport :



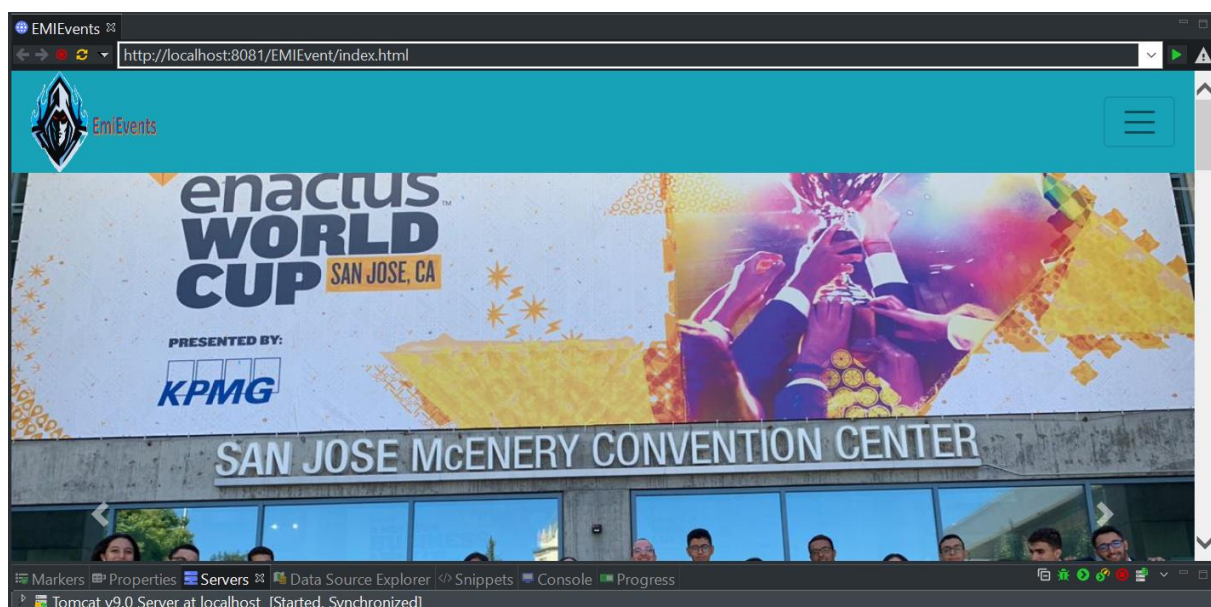
### c) Hébergement du site web

Pour déployer notre site web nous avons mis les fichier (html ,css,js,...) dans le dossier WebContent/WEB-INF après la création d'un web dynamique projet dans éclipse. Après l'instalation de XAMPP ( logiciels permettant de mettre en place un serveur Web local) en clique sur **start** Apache:



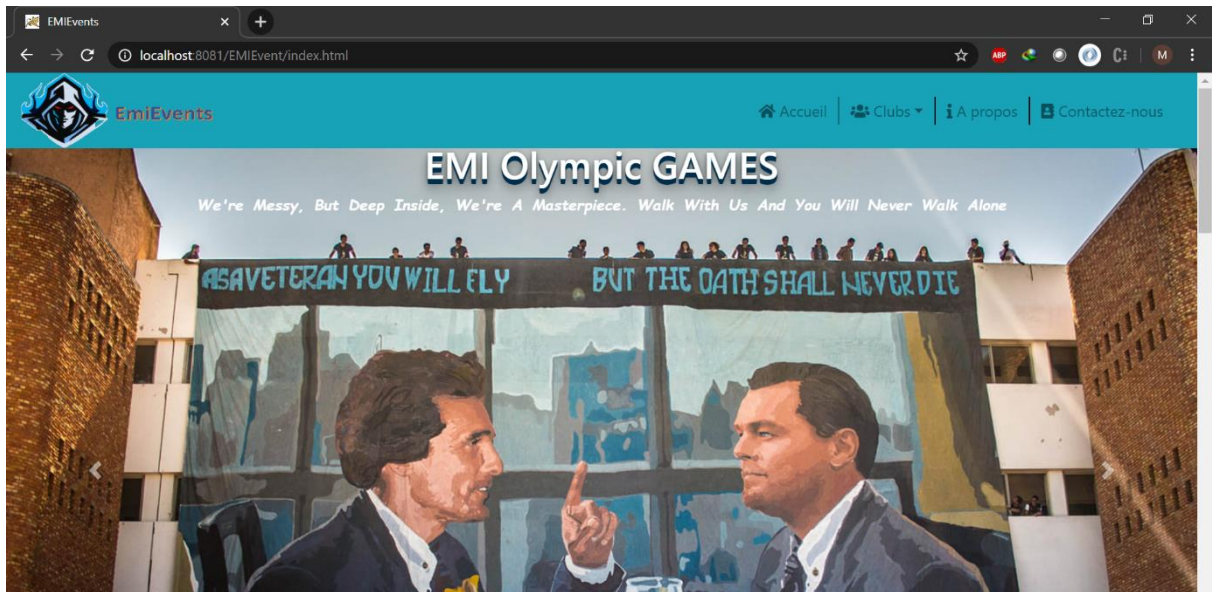
Ensuite clique droit de la page que nous voulons déployer puis **run on server** :

Voici le résultat dans le navigateur d'éclipse :



Dans le navigateur chrome :





#### IV. Conclusion

Ce rapport décrit tous les étapes que nous avons passées pour établir un site web qui a comme objectif principal la présentation des événements des clubs de l'EMI.

Le début était de recueillir les données liées au sujet.

Après nous avons organisés ces données dans un fichier XML et les exploités par XSLT et DOM.

Dans la partie technologies de web nous avons ajoutés des styles des images des icônes des liens et associé un fichier JavaScript avec html pour rendre le site un peu dynamique.

#### V. Webographie

<https://getbootstrap.com/>

<https://fontawesome.com/>

<https://fr.wikipedia.org/wiki/>

<https://www.logogenie.fr/>