République Algérienne Démocratique et Populaire

Ministère de l’Enseignement Supérieur et de la Recherche Scientifique

Université AMO de Bouira

Faculté des Sciences et des Sciences Appliquées

Département d’Informatique

Compte rendu de TP 4

Méthodes et Technologies d’Implémentation

Spécialité : Génie des Systèmes Informatiques

Programmation XML avec DOM

Réalisé par :

-Akli Meziane

-Ladjal Sarah

Encadré par :Mr.Taha zerrouki

Partie TP

**Introduction**

A l’objectif d’apprendre le XML DOM API. Il est demandé de réaliser un travail sur le language python qui réaliser le fonctionnement entre le language de balisage XML et le DOM

**La méthodologie du travail**

dans le premier partie de TP on a executer et tester les resultats des code et son role et apprender un novelle type de programmation il est DOM.

**Les outils**

Copy Editor .

PyCharm

**Teste et résultat**

Affichage de Fonction 1

|  |
| --- |
| **Capture.PNG** |

|  |
| --- |
| Affichage de fonction 2 display télephone  dislp.PNG |
| Affichage de focntion 3 Display téléphone personne  **tl.PNG** |
| Affichage de fonction Add\_id personne  dd.PNG |
|  |

**Partie 2 : Manipulation avec javaScript**

|  |  |
| --- | --- |
| **Question 1:Larbre DOM**  **js1.PNG** | |
| Qustion 2: ajouter le code de First Name et Last Name  **js2.PNG** | |
| Qustion 3:creer un code pour le num tel et la ville  **jstel.PNG**  **tel.PNG** | |
| Question 4:creation des elements HTML  **js3.PNG** | |

**Code annexe**

|  |
| --- |
| DATA\_FILE = 'annuaire-v0.2.xml'  import xml.dom.minidom as minidom  import sys  def treat\_doc(xmldoc):  annuaire = xmldoc.getElementsByTagName('annuaire')[0]  print(annuaire)  cpt = 0  for personne in annuaire.childNodes:  print("-" \* 40)  print("Personne n°", cpt)  print(personne.toxml())  cpt += 1  def display\_tel\_personne(xmldoc):  personnes = xmldoc.getElementsByTagName('personne')  print(personnes)  cpt = 0  # display telephone by personne  for personne in personnes:  print("-" \* 40)  print("Personne n°", cpt)  nom = personne.getElementsByTagName('nom')[0]  prenom = personne.getElementsByTagName('prenom')[0]  tels = personne.getElementsByTagName('telephone')  print("\*" \* 20)  print("Nom:\t", nom.firstChild.data)  print("Prénom:\t", prenom.firstChild.data)  for tel in tels:  print("-" \* 20)  print("N°:", tel.firstChild.data)  print("Type:", tel.getAttribute("type"))  cpt += 1  def add\_id\_personne(xmldoc):  # get the personnes list  personnes = xmldoc.getElementsByTagName('personne')  print(personnes)  cpt = 0  # display personne by personne  for personne in personnes:  print("-" \* 40)  print("Personne n°", cpt, personne.nodeValue, personne.nodeType,personne.setAttribute('id', str(cpt)))  cpt += 1  print(personne.toxml())  def display\_tel(xmldoc):  telephones = xmldoc.getElementsByTagName('telephone')  print(telephones)  cpt = 0  # display tel by tel  for tel in telephones:  print("-"\*40)  print("Tel n°", cpt)  print(tel.toxml())  print("N°:",tel.firstChild.data)  print("Type:",tel.getAttribute("type"))  cpt += 1  def main():  try:xmldoc = minidom.parse(DATA\_FILE)  except:  print("Can't Open the file",)  sys.exit()  #display\_tel(xmldoc)  #print(xmldoc.toxml())  #treat\_doc(xmldoc)  #display\_tel\_personne(xmldoc)  #add\_id\_personne(xmldoc)  return 0  if \_\_name\_\_ == '\_\_main\_\_':  **main()** |

Partie : Travail a domicile

**Première partie**

**Introduction**

A l’objectif d’apprendre le XML DOM API. Il est demandé de réaliser ce travail sur le fichier résultant du TP1.

**La méthodologie du travail**

On a récupérer le contenu du fichier.txt (liste des étudiants composé par nom, prénom, moyennes S1, S2, crédit) du TP précèdent avec ce ligne de code **DATA\_FILE = 'tr4.xml'**

, Ecrire un programme de conversion en XML, puis on a fait les calculs demandé en utilisent DOM.

**Les outils**

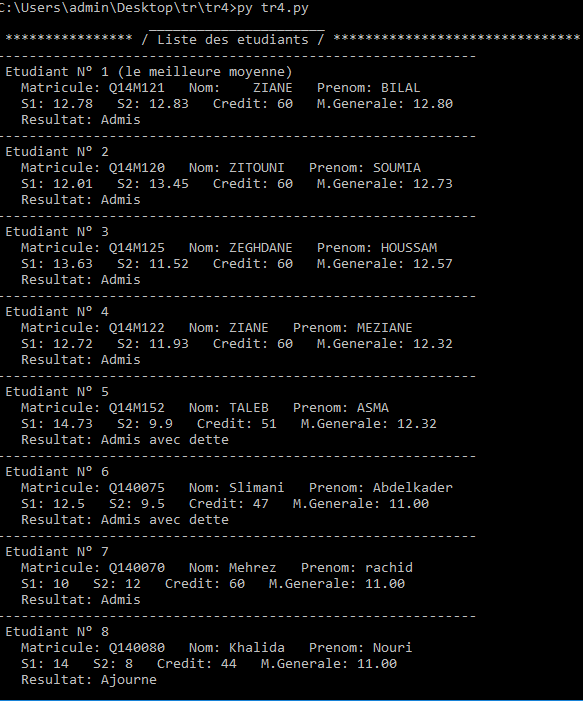
* XML Copy Editor.
* Un **éditeur XML** est un éditeur de [langage de balisage](https://en.wikipedia.org/wiki/Markup_language) avec des fonctionnalités supplémentaires pour faciliter l' [édition](https://en.wikipedia.org/wiki/Editing) de [XML](https://en.wikipedia.org/wiki/XML) . Cela peut être fait à l'aide d'un [éditeur de](https://en.wikipedia.org/wiki/Text_editor)[texte brut](https://en.wikipedia.org/wiki/Plain_text) , avec tout le code visible, mais les éditeurs XML ont ajouté des fonctionnalités telles que la [complétion des balises](https://en.wikipedia.org/wiki/Word_completion) et des menus et des boutons pour les tâches courantes dans l'édition XML, sur la base des données fournies avec la [définition de type de document](https://en.wikipedia.org/wiki/Document_type_definition) (DTD) ou l'arborescence XML.
* Pycharm.

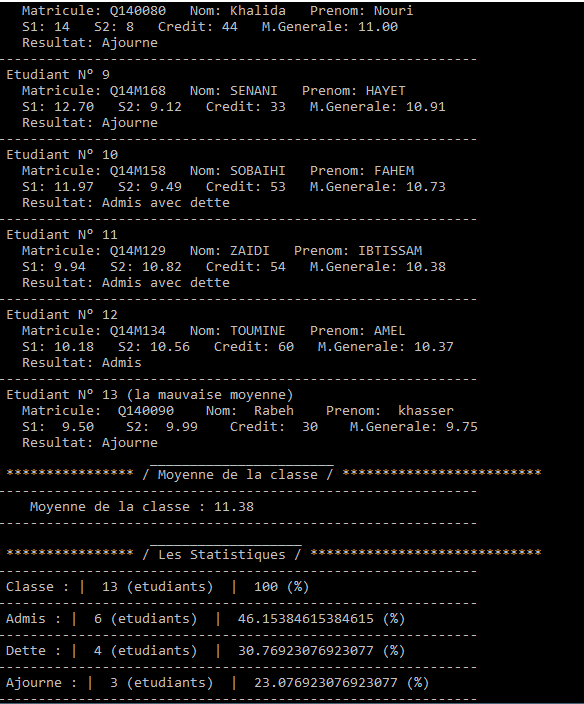
**PyCharm** est un [environnement de développement intégré](https://fr.wikipedia.org/wiki/Environnement_de_d%C3%A9veloppement) utilisé pour programmer en [Python](https://fr.wikipedia.org/wiki/Python_(langage)).

Il permet l'analyse de code et contient un débogueur graphique. Il permet également la gestion des tests unitaires, l'intégration de logiciel de gestion de versions, et supporte le développement web avec [Django](https://fr.wikipedia.org/wiki/Django_(framework)).

**Tests et résultats**

On a fait l’exécution avec le CMD :

****

****

**Annexe**

*#!/usr/bin/env python  
#-\*- coding: utf-8 -\*-*DATA\_FILE = **'tr4.xml'  
import** sys  
**import** xml.dom.minidom **as** minidom  
  
*#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* MAIN \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****def** main():  
 **try**:  
 xmldoc = minidom.parse(DATA\_FILE)  
 **except**:  
 **print** (**"Can't Open the file"**,DATA\_FILE)  
 sys.exit()  
 **return** xmldoc  
  
*#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* triter \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****def** triter(xmldoc):  
 liste=[]  
 admi=0  
 dett =0  
 ajou =0  
 cid=0  
 som=0  
  
 etudiants = xmldoc.getElementsByTagName(**"etudiant"**)  
 **for** etudiant **in** etudiants :  
 cid +=1  
 etudiant.setAttribute(**'id'**, str(cid))  
 m = etudiant.getElementsByTagName(**'matricule'**)[0].firstChild.data  
 n = etudiant.getElementsByTagName(**'nom'**)[0].firstChild.data  
 p= etudiant.getElementsByTagName(**'prenom'**)[0].firstChild.data  
 s1 = etudiant.getElementsByTagName(**'s1'**)[0].firstChild.data  
 s2 = etudiant.getElementsByTagName(**'s2'**)[0].firstChild.data  
 c =etudiant.getElementsByTagName(**'credit'**)[0].firstChild.data  
 mg = calcule(float(s1),float(s2))  
 res=result(float(s1),float(s2),float(c))  
 **if** res == **"Admis"**:  
 admi +=1  
 **elif** res == **"Admis avec dette"**:  
 dett +=1  
 **else** :  
 ajou +=1  
  
 som=som+mg  
  
 mgenerale=xmldoc.createElement(**'Mgenerale'**)  
 mtext=xmldoc.createTextNode(str(mg))  
 mgenerale.appendChild(mtext)  
 etudiant.appendChild(mgenerale)  
 resultat=xmldoc.createElement(**'resultat'**)  
 rtext=xmldoc.createTextNode(str(res))  
 resultat.appendChild(rtext)  
 etudiant.appendChild(resultat)  
   
 liste.append([mg,cid])  
  
 liste=sorted(liste,reverse=True)   
   
 tr2 = xmldoc.getElementsByTagName(**"tr2"**)[0]   
 **for** i **in** range(4):  
 nombre=xmldoc.createElement(**'nombre'**)  
 pourcen=xmldoc.createElement(**'pourcentage'**)  
 types=xmldoc.createElement(**'type'**)  
 **if** i==0:  
 typ=xmldoc.createTextNode(**"Classe"**)  
 nb=xmldoc.createTextNode(str(cid))  
 pr=xmldoc.createTextNode(**"100"**)  
 **if** i==1:  
 typ=xmldoc.createTextNode(**"Admis"**)  
 nb=xmldoc.createTextNode(str(admi))  
 pr=xmldoc.createTextNode(str(admi\*100/cid))  
 **if** i==2:  
 typ=xmldoc.createTextNode(**"Dette"**)  
 nb=xmldoc.createTextNode(str(dett))  
 pr=xmldoc.createTextNode(str(dett\*100/cid))  
 **if** i==3:  
 typ=xmldoc.createTextNode(**"Ajourne"**)  
 nb=xmldoc.createTextNode(str(ajou))  
 pr=xmldoc.createTextNode(str(ajou\*100/cid))  
   
 types.appendChild(typ)  
 nombre.appendChild(nb)  
 pourcen.appendChild(pr)  
  
 statisti=xmldoc.createElement(**'statistique'**)  
 statisti.appendChild(types)  
 statisti.appendChild(nombre)  
 statisti.appendChild(pourcen)  
 tr2.appendChild(statisti)  
   
 statistiques = xmldoc.getElementsByTagName(**"statistique"**)   
   
 mc=xmldoc.createElement(**'Mclasse'**)  
 m=xmldoc.createTextNode(str(som/cid))  
 mc.appendChild(m)  
 tr2.appendChild(mc)  
   
 mclasse = xmldoc.getElementsByTagName(**"Mclasse"**)[0]  
 moynneclass=mclasse.firstChild.data  
  
  
 affiche(etudiants,liste,statistiques,moynneclass)  
  
*#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* moyenne \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****def** calcule (x,y) :  
 mg=(x+y)/2  
 **return** mg  
  
*#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* resultat \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****def** result (x,y,z):  
 **if** x>=10 **and** y>=10 :  
 res=(**"Admis"**)  
 **elif** z>=45 :  
 res=(**"Admis avec dette"**)  
 **else** :  
 res=(**"Ajourne"**)  
 **return** res  
  
*#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* affiche \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****def** affiche(xm,lis,st,cl):  
 **print**(**""" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ """**)  
 **print**(**""" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* / Liste des etudiants / \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* """**)  
   
 **for** i **in** range(len(lis)):  
 x=lis[i][1]  
 **for** a **in** xm :  
 d=a.getAttribute(**'id'**)  
 **if** int(d) == x :  
 m = a.getElementsByTagName(**'matricule'**)[0].firstChild.data  
 n = a.getElementsByTagName(**'nom'**)[0].firstChild.data  
 p= a.getElementsByTagName(**'prenom'**)[0].firstChild.data  
 s1 = a.getElementsByTagName(**'s1'**)[0].firstChild.data  
 s2 = a.getElementsByTagName(**'s2'**)[0].firstChild.data  
 c =a.getElementsByTagName(**'credit'**)[0].firstChild.data  
 g =a.getElementsByTagName(**'Mgenerale'**)[0].firstChild.data  
 r =a.getElementsByTagName(**'resultat'**)[0].firstChild.data  
 **print** (**"-"**\*60)  
 **if** i+1 == 1:  
 **print** (**""" Etudiant N° %d (le meilleure moyenne)"""**%(i+1))  
 **elif** i+1 == len(lis):  
 **print** (**""" Etudiant N° %d (la mauvaise moyenne)"""**%(i+1))  
 **else** :  
 **print**( **""" Etudiant N° %d"""**%(i+1))  
   
 **print**(**""" Matricule: %s Nom: %s Prenom: %s"""**%(m,n,p))  
 **print**(**""" S1: %s S2: %s Credit: %s M.Generale: %.2f"""**%(s1,s2,c,float(g)))  
 **print**(**""" Resultat: %s"""**%r)  
  
 **print**(**""" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ """**)  
 **print**(**""" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* / Moyenne de la classe / \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* """**)  
 **print** (**"-"**\*60)  
 **print** (**""" Moyenne de la classe : %.2f"""**%float(cl))  
 **print** (**"-"**\*60)  
   
 **print**(**""" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ """**)  
 **print**(**""" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* / Les Statistiques / \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* """**)  
 **for** r **in** st :  
 typ=r.getElementsByTagName(**'type'**)[0].firstChild.data  
 nb=r.getElementsByTagName(**'nombre'**)[0].firstChild.data  
 pr=r.getElementsByTagName(**'pourcentage'**)[0].firstChild.data  
 br=(**"%"**)  
 **print** (**"-"**\*60)  
 **print**( **""" %s : | %s (etudiants) | %s (%s) """**%(typ,nb,pr,br))  
 **print** (**"-"**\*60)  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 triter(main())