Défi à réaliser : **concevoir une application**

"Disponibilités des places de parking de la ville de NANTES"

sur Processing 1.5

Plan des séances :

Séance 1 - L'OpenData : découverte d'un concept

1. Introduction à l'Opendata
2. L'Opendata et son vocabulaire :
3. Plateforme NOD : Nantes Ouvertures des Données
- Quelles données ?
- Fonctionnement de l'API
- Accès aux données sous quelle licence ?
- Création d'un compte sur la plateforme NOD

Séance 2 - Manipuler les données : XML et CSV

1. Fichier CSV
- Définition
- Créer un fichier CSV

2. CSV avec Processing
3. Format XML
4. XML avec Processing
- Mettre des données au format XMLSéquence 4 - Commander l'API de l'Open Data de Nantes avec Processing
- Exemples XML avec Processing

- Commander l'API de l'Open Data de Nantes avec Processing

1. Fonctionnement de l'API
2. Mise en œuvre de l'API avec Processing
3. Interprétation des données XML
4. Mettre en forme les données de l'API

Séance 3 - Développer une véritable application Open Data avec Processing

1. Afficher la carte de Nantes
2. Récupérer les coordonnées géographiques d'un parking
3. Localiser un parking sur la carte
4. Rendre l'application interactive
5. Réalisation de l'application finale

Ce défi sera réalisé sur Processing 1.5.
Copier-coller Processing 1.5 de l'ENT sur votre clé usb

Compétences visées:

- Référentiels par compétences
  - **C1 : Décrire et expliquer une situation, un système ou un programme**
    - **C1.1** Justifier dans une situation donnée, un codage numérique ou l'usage d'un format approprié, qu'un programme réalise l'action attendue...
    - **C1.2** Détailler le déroulement d'une communication numérique, le rôle des constituants d'un système numérique, le rôle des éléments constitutifs d'une page web, ...
  - **C2 : Concevoir et réaliser une solution informatique en réponse à un problème**
    - **C2.1** Analyser un besoin dans un système d'information, le fonctionnement d'un algorithme...
    - **C2.2** Structurer une formule logique, des données, une arborescence, une page web, une approche fonctionnelle en réponse à un besoin...
    - **C2.3** Développer une interface logicielle ou une interface homme-machine, un algorithme, un programme, un document ou fichier numérique...
  - **C4 : Communiquer à l'écrit et à l'oral**
    - **C4.1** Documenter un projet numérique pour en permettre la communication en cours de réalisation et à l'achèvement, tout en précisant le déroulement et la finalité du projet.
  - **C5 : Faire un usage responsable des sciences du numérique en ayant conscience des problèmes sociétaux induits**
    - **C5.1** Avoir conscience de l'impact du numérique dans la société notamment de la persistance de l'information numérique, de la non-rivalité des biens immatériels, du caractère supranational des réseaux, de l'importance des licences et du droit.
    - **C5.2** Mesurer les limites et les conséquences de la persistance de l'information numérique, des lois régissant les échanges numériques, du caractère supranational des réseaux.
    -

Savoirs associés du programme d'ISN :

- Référentiels par savoirs
  - **1 -** Représentation de l'information
    - **1.4** Formats
    - **1.6** Structuration et organisation de l'information
    - **1.8** Non-rivalité de l'information
  - **2 -** Algorithmique
    - **2.1** Algorithmes simples
  - **3 -** Langages et programmation
    - **3.1** Types de données
    - **3.2** Fonctions
  - **5 -** Réseaux
    - **5.4** Supranationalité

OPEN DATA (séance 1)

Présentation et cadre légal

Eléments du programme étudiés
• Représentation de l'information : formats, structuration et organisation de l'information, non-rivalité de l'information. • Architectures matérielles : réseaux
Compétences mise en œuvre :
C1 : **justifier**, dans une situation donnée, un codage numérique ou l'usage d'un format approprié... **Détailler** le déroulement d'une communication numérique … **C4 : documenter un projet numérique** pour en permettre la communication.. **C5 : avoir conscience de l'impact du numérique dans la société** notamment de l'importance des licences et du droit. **Mesurer les limites et les conséquences** des lois régissant les échanges numériques.

Certaines collectivités locales, les gouvernements, les grandes métropoles, les organismes publiques de recherche rendent accessible les données publiques.
Pour quelles raisons ?

- Assurer la transparence publique.
- Développer des applications innovantes pour améliorer les services.

L'objectif de cette séance est d'étudier les lois régissant l'Open Data et d'étudier les moyens d'accéder et d'utiliser ces données.

I) Concept et vocabulaire de l'Open Data.

A) Présentation.

Visualiser le spot suivant : http://www.youtube.com/watch?v=aHxv_2BMJfw&hd=1#
ou sur l'ENT (dossier : ressource-seance1-opendata-la-loupe)
et répondre aux questions suivantes :

1. Quelles sont les quatre catégories de données à disposition ?
2. L'Open Data est une plateforme mettant à disposition des données publiques. Qu'est-ce qu'une donnée privée ?
3. A qui s'adresse l'Open Data ?
4. Quelles sont les données permettant de « simplifier l'accès aux transports aux personnes à mobilité réduite » ?
5. Sur quelle type de support peut-on consulter l'appli pour simplifier l'accès aux transports aux personnes à mobilité réduite » ?

Consultation de la carte de l'Open Data :
http://libertic.wordpress.com/2012/01/02/carte-de-france-de-lopen-data-v4/?relatedposts_exclude=797

B) Langage.

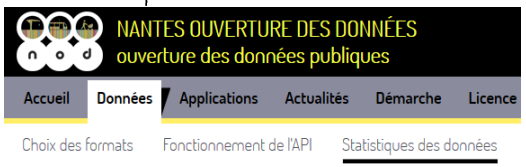
A l'aide de recherches documentaires sur le web, définir les termes suivants :

- API
- Crowdsourcing
- Data journalism
- Linked Data
- Interopérabilité
- Métadonnée
- Web 2.0

II) Présentation d'un exemple : plateforme NOD (Nantes Ouvertures des Données)

Ouvrir la plateforme de Nantes : http:\\data.nantes.fr

Exemples de données accessibles



1. Quelles sont les 4 principales catégories de données proposées ? les 4 principaux formats de données proposées ?
2. Quelles sont les données réactualisées en temps réel ? quotidiennement ? mensuellement ?

L' API

1. Relever la syntaxe d'un appel sur l'API v2.
2. Quel est le format par défaut des réponses sur l'API v2 ?
3. Mêmes questions pour l'API v1.

Licence de l'accès aux données

La loi du 17 juillet 1978 donne à toute personne un droit d'accès et de réutilisation des informations publiques.
La CADA (Commission d'Accès aux Documents Administratifs) s'occupe du respect de la loi.

1. Quel est le type de licence utilisée pour l'Open Data de Nantes. ?

Certaines données publiques restent payantes en France. Elles sont répertoriées sur le site :
http://www.data-publica.com/content/2012/09/les-donnees-publiques-payantes-en-france-ce-quil-faut-retenir/
A l'aide du fichier fourni, déterminer le montant de la licence permettant d'accéder aux prix des carburants pratiqués dans les stations-services.

2. Rechercher sur le web :

- la différence entre les licences libres et les licences gratuites.
- la définition de la licence **Creative Commun.** (vu lors d'exposés oraux)

En allant dans la rubrique COMPTE, créer un compte pour récupérer une clé (à noter) qui vous permettra d'accéder aux données de l'open data de NANTES

OPEN DATA (séance 2)

Exploitation des fichiers de données
Commander une API

Eléments du programme étudiés

- Représentation de l'information :
  formats, structuration et organisation de l'information.
- Algorithmes :
  algorithme de recherche d'un élément dans un tableau.
- Langages et programmation :
  types des données et fonctions

Compétence mise en œuvre :

C2 : **Concevoir et réaliser** un solution informatique en réponse à un problème donné

Pour assurer l'interopérabilité, on accède en général aux données par téléchargement de fichiers au format CSV, ou par accès direct via une API délivrant les informations sous format XML.
L'objectif de cette séance est d'étudier les caractéristiques des formats CSV et XML et les procédés pour accéder et traiter les données CSV et XML ?

I) Fichier CSV

A) Structure

1. Lancer Open Office Calc et recopier les données ci-contre
2. Enregistrer au format CSV sous le nom : liste
3. Ouvrir ce fichier avec notepad.
4. Commenter le résultat

	A	B	C	D
1	NOM	PRENOM	CLASSE	SEXE
2	Portier	Fabrice	TS1	G
3	Petit	Sandra	TS1	F
4	Le Roux	Pauline	TS2	F
5	Sandjak	David	TS1	G

B) Traitement des données avec Processing.

Voici le code permettant d'extraire les données d'un fichier CSV :

```
//.....................................
String [] lignes = loadStrings("liste.csv");
//.....................................
for(int i = 0; i<lignes.length;i++){
  println("lignes n°"+i+" : "+lignes[i]);
}
```

1. Faites un copier/coller de ce code et ajouter le fichier liste.csv au sketch (menu Sketch/Add file…). Exécuter ce programme.
2. Commenter ces lignes de codes
   (voir la documentation : astuce→ par exemple pour *loadStrings* le sélectionner à la souris puis clic droit > Find Reference)
3. L'objectif de la suite est d'extraire les éléments de chaque lignes séparés par « ; » :
   a) Quel est le rôle de la fonction split()
      (voir documentation : Help>Reference> …….. rubrique : **String Functions**)
   b) Compléter le code suivant à l'aide de cette fonction pour obtenir le résultat ci-après ;

```
String [] lignes =
loadStrings("liste.csv");
//affichage des lignes
for(int i = 0; i<lignes.length;i++){
  println("lignes n°"+i+" : "+lignes[i]);
   // à compléter avec la fonction split()
   ……..
  println(tableau);
  }
```

```
lignes n°0 : "NOM","PRENOM","CLASSE","SEXE"
[0] ""NOM""
[1] ""PRENOM""
[2] ""CLASSE""
[3] ""SEXE""
lignes n°1 : "Portier","Fabrice","TS1","G"
[0] ""Portier""
[1] ""Fabrice""
[2] ""TS1""
[3] ""G""
lignes n°2 : "Petit","Sandra","TS1","F"
[0] ""Petit""
[1] ""Sandra""
[2] ""TS1""
[3] ""F""
lignes n°3 : "Le Roux","Pauline","TS2","F"
[0] ""Le Roux""
[1] ""Pauline""
[2] ""TS2""
[3] ""F""
lignes n°4 : "Sandjak","David","TS1","G"
[0] ""Sandjak""
[1] ""David""
[2] ""TS1""
[3] ""G""
```

II) Fichier XML (eXtensible Markup Language)

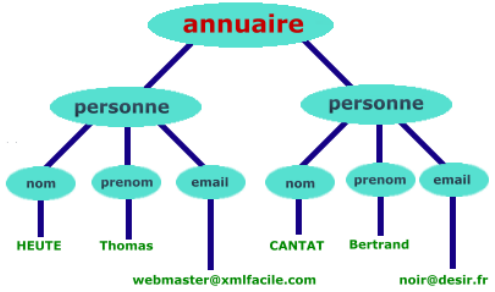
A) Présentation

Traduction : Langage à balises étendu, ou Langage à balises extensible.

Principe : c'est une méthode permettant créer des documents structurés. Pour cela, on utilise des balises (markup) : elles s'écrivent sous la forme **<balise>**

La structure d'un document XML peut être représentée par un arbre. Voici l'exemple d'un fichier annuaire composé de plusieurs personnes décrites par le nom , le prénom et l'email.

Structure arborescente



XML

```
<?xml version="1.0" encoding="UTF-8"?>
<annuaire>
    <personne>
        <nom>HEUTE</nom>
        <prenom>Thomas</prenom>
        <email>webmaster@xmlfacile.com</email>
    </personne>
    <personne >
        <nom>CANTAT</nom>
        <prenom>Bertrand</prenom>
        <email>noir@desir.fr</email>
    </personne>
</annuaire>
```

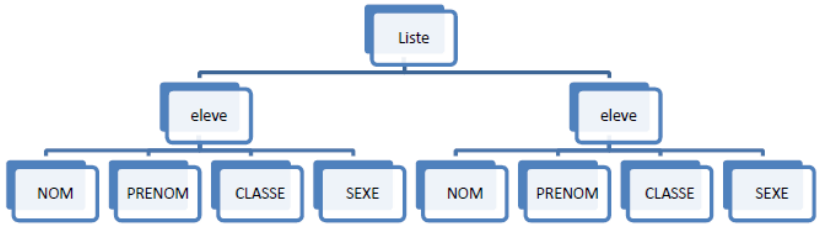
Ce document XML contient une 1ère ligne indiquant la version XML utilisée et l'encodage.
Il a une seule racine (**annuaire**), 2 éléments (**personne**), et la balise personne a 3 éléments (**nom**,**prénom**,**email**). Chaque bulle est appelé NŒUD de l'arbre.

On peut ajouter un attribut à une balise (ici associé à <personne> (**type**))

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<annuaire>
   <personne type="étudiant">
      <nom>HEUTE</nom>
      <prenom>Thomas</prenom>
      <email>webmaster@xmlfacile.com</email>
   </personne>
   <personne type="chanteur">
      <nom>CANTAT</nom>
      <prenom>Bertrand</prenom>
      <email>noir@desir.fr</email>
   </personne>
</annuaire>
```

B) Créer un fichier XML.

Nous allons structurer le fichier liste.csv du **I)** au format XML avec la structure suivante :



1. Ouvrir Notepad et sélectionner dans le menu « Langage » l'option XML.
2. Sur le modèle du fichier XML du **A)** (fichier annuaire) :
   créer le fichier d'auteurs au format XML (attention !!!! il faut introduire une première ligne de code : **< ?xml version ="1.0" encoding= »UTF-8" ?>**
3. Enregistrer avec l'extension XML sous l'intitulé liste.

C) Lecture des données avec Processing 1.5

Processing contient une librairie XML pré-implantée (on parle de librairie native).
(attention !!!! dans les versions 2+ de Processing, la librairie XML a été modifiée)
Voici un exemple permettant de comprendre les déclarations et certaines méthodes.

1. Copier le fichier XML ci-dessous dans Notepad et enregistrer sous l'intitulé **sites.xml**

```
<?xml version="1.0"?>
<websites>
    <site id="0" url="processing.org">Processing</site>
    <site id="1" url="mobile.processing.org">Processing Mobile</site>
</websites>
```

2. Copier/coller le code ci-dessous dans Processing 1.5 et enregistrer le fichier sous l'intitulé :
   Exemple_xml.pde

```
XMLElement xml;

void setup() {
  xml = new XMLElement(this, "sites.xml");
  int numSites = xml.getChildCount();
  for (int i = 0; i < numSites; i++) {
    XMLElement kid = xml.getChild(i);
    int id = kid.getInt("id");
    String url = kid.getString("url");
    String site = kid.getContent();
    println(id + " : " + url + " : " + site);
  }
}
```

3. Ajouter le fichier xml au scketch (menu Sketch/Add file…) et tester le programme.
Le type d'un objet XML est : XMLElement.

4. Commenter chaque ligne de code du programme Processing.
(Astuce : sélectionner avec la souris un élément du programme, clic droit puis sélectionner le menu Help l'option « Find in Reference »)

Voici une variante du code précédent :

```
XMLElement xml;

void setup() {
  xml = new XMLElement(this, "sites.xml");
  XMLElement []kids = xml.getChildren();
  for (int i = 0; i < kids.length; i++) {
    String site = kids[i].getContent();
    println(site);
  }
}
```

5. Tester ce programme (*s'assurer que le fichier sites.xml a été collé dans le sketch)*
Quelle est la différence entre les méthodes getChildren() et getChild() ?

Défi n°1:

Implémenter sous Processing 1.5 , un programme permettant de lire et d'afficher les données contenues dans le fichier liste.xml

III) Prise en main de l'API de l'Open Data de Nantes.

A) Commandes

L'Open Data de Nantes propose deux API pour accéder aux données.
L'API v1 donne accès aux données liées à la mobilité. Elles sont réactualisées en temps réel.

1. Lister les commandes donnant accès aux données en temps réel.
2. Quel est le format d'une requête sur l'API ?

B) Prise en main sur Processing.

Avec votre clé d'application créée sur l'Open Data de Nantes lors de la séance 1, vous allez implémenter un programme qui permet d'interroger le service avec son URL dont voici la constitution : **URL du service / catégorie de données / version de l'API /clé d'application**
La requête s'effectue avec l'instruction : **loadStrings**. Elle renvoie un tableau de chaîne de caractères (String) dont chaque entrée correspond à une ligne de la réponse de l'API de la ville de Nantes. La réponse est au format XML.

Tester le programme en l'adaptant avec votre clé :

```
String DataNantesCle="309NRRB5VI21JW6";
String DataNantesCommande="getDisponibiliteParkingsPublics";
String DataNantesVersion="1.0";

void setup(){
  // Adresse pour accéder aux données
  String  url =
"http://data.nantes.fr/api/"+DataNantesCommande+"/"+DataNantesVersion+"/" DataNantesCle;
  // Connexion à l'API et chargement des données
  String [ ] lines=loadStrings(url);
  // affichage du résultat dans la console
  for(int i=0;i<lines.length;i++){
    println(lines[i]);
  }
}

void draw(){
  // aucun dessin pour le moment
}
```

IV) Comment interpréter les données reçues ?

Plutôt que d'utiliser la fonction loadStrings, on va utiliser des fonctions spécifique à Processing permettant d'interpréter le format XML. Seules la fonction d'appel et le format du résultat changeront : **XMLElement xml = new XMLElement(this,url);**

1. Modifier le programme précédent.

L'objectif de ce qui suit est de récupérer le nombre de parkings dans la ville de Nantes.

Dans la structure d'arbre du format XML, le nombre est obtenu en accédant au nœud <Groupe_Parking>.
L'idée est donc de stocker dans un tableau d'éléments de type ***XMLElement*** tous les nœuds <Groupe_Parking>. On pourra le faire avec la fonction **getChildren(path)** en définissant le chemin d'accès aux données de <Groupe_Parking>.

2. Compléter et tester le programme suivant :

```
XMLElement[] parkings = xml.getChildren(. . . . . .);
println(parkings[1]);
int nombreParking = parkings.length;
println("Nombre de parkings sur Nantes : "+ nombreParking);
```

Maintenant, nous allons récupérer le nom du parking n°1 à l'aide des fonctions getChild("Grp_nom") et getContent().

3. Modifier votre programme pour afficher le nom du parking n°1 (TOUR DE BRETAGNE).

V) Mise en forme des données reçues (Défi n°2)

Créer un programme permettant de mettre en forme les données reçues sur les parkings dans la fenêtre graphique de Processing (il faudra afficher du **texte** et gérer des **polices** de caractère : vu dans les TP précédents).

OPEN DATA (séance 3)

Développement d'une APPLI…

Eléments du programme étudiés

- Représentation de l'information :
  formats, structuration et organisation de l'information.
- Algorithmes :
  algorithme de recherche d'un élément dans un tableau.
- Langages et programmation :
  types des données et fonctions

Compétence mise en œuvre :

C2 : **Concevoir et réaliser** un solution informatique en réponse à un problème donné

Nantes métropole a lancé dernièrement un appel à projets innovants :
« Rendez-moi la ville + facile »
Il s'agissait notamment de créer de nouveaux services en utilisant les données publiques.
L'accent était mis sur la créativité et l'ingéniosité des citoyens, des entreprises, des start-up, des étudiants, des développeurs, des associations, ……
Le prix du jury a été attribué à l'application mobile iPhone: **Statiophone**.

Elle permet de connaître en temps réel les places disponibles dans les parkings publics de Nantes. Elle vous gélocalise et affiche les parkings les plus proches de votre lieu.

Nous allons créer une application du même type (sans géolocalisation) avec Processing 1.5

Voici les questions auxquelles, il faudra répondre :

Comment afficher une carte ?

Comment récupérer les coordonnées géographiques des parkings ?

Comment localiser les parkings sur la carte ?

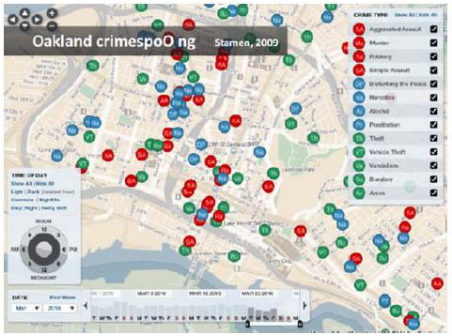
Comment rendre l'application interactive ?

I) Affichage de la carte de NANTES : cartographie avec Processing.

Pour afficher et récupérer des informations sur une carte avec Processing, on peut :

- utiliser le logiciel de création de carte **TileMil**
- utiliser la librairie **Unfolding**. Elle est très utilisée dans de nombreuses applications.

Nous allons donc utiliser cette 2ième méthode.



Localisation de crimes et le type de crimes commis dans une ville durant une période donnée.

Télécharger la dernière version de la librairie Unfolding : http://unfoldingmaps.org/
Décompresser le dossier et copier le dossier des librairies de Processing (dossier modes>java>librairies) puis installer cette librairie (add library …)
(*les cartes seront récupérées sur le net donc il est indispensable d'être connecté)*

II) Mise en œuvre sur Processing.

A) Un 1er exemple.

Copier-coller le code suivant et exécuter le:

```
//importation des librairies
import processing.opengl.*;
import de.fhpotsdam.unfolding.utils.*;
import de.fhpotsdam.unfolding.*;
import de.fhpotsdam.unfolding.geo.*;

//déclaration d'une variable globale : création d'un objet carte
UnfoldingMap carte;
void setup(){
  //dimension de la fenêtre
  size(800,600);
  //initialisation de l'objet "carte"
  carte = new UnfoldingMap(this);
  /*initialisation d'un gestionnaire d'événements : ajout de
  fonctions s'appliqauant à notre objet carte comme double-clique, zoom, glissement, .....*/
  MapUtils.createDefaultEventDispatcher(this,carte);
}

void draw(){
  //affichage de la carte
  carte.draw();
}
```

B) Fonctions de la librairie Unfolding

→ Obtenir la géolocalisation d'un point de la carte :

La fonction **getLocation(x,y)** permet d'obtenir la géolocalisation du point de coordonnées (x ; y)

```
void draw(){
  carte.draw();
  Location coordgeo = carte.getLocation(mouseX,mouseY);
  fill(0);
  text(coordgeo.getLat() + ", " + coordgeo.getLon(), mouseX, mouseY);
}
```

→ Choisir le format de la carte :

Unfolding Default

Google Terrain


Microsoft Aerial

Midnight Commander
(CloudMade)


Watercolor
(Stamen)


Buildings
(OpenStreetMap + TileMill)

Il faut commencer par importer la librairie provider :

```
import de.fhpotsdam.unfolding.providers.*;
```

Ensuite, modifier l'initialisation de l'objet carte en ajoutant un second attribut :

```
carte = new UnfoldingMap(this, new Microsoft.AerialProvider());
```

→ Zoomer et glisser :

On peut faire glisser la carte avec la souris ou avec les flèches de déplacement du clavier.
On peut aussi zoomer en utilisant la molette de la souris ou en appuyant sur les touches + ou – du clavier.
On peut paramétrer le niveau de zoom et la zone de déplacement pour délimiter la visualisation d'une zone géographique

Par exemple, voici le code pour zoomer sur Berlin afin de visualiser l'ensemble de la ville :

```
// déclaration et initialisation de l'objet
Location berlin = new Location(52.5f, 13.4f);
// zoom de Berlin
carte.zoomAndPanTo(berlin, 10);
// limitation du glissé à 30 km autour d'un point
carte.setPanningRestriction(berlin, 30);  // en km
```

Application à la carte de NANTES :

- Rechercher les coordonnées géographiques de NANTES sur le web : http://www.gpsfrance.net/adresse-vers-coordonnees-gps , par exemple.
- Sur Processing, implémenter un programme permettant d'afficher la cartes de NANTES (vue satellite) avec un glisser maximal de 10 km.

III) Réalisation de l'application.

A) Récupérer les coordonnées géographiques d'un parking.

Les données XML obtenues par l'API de NANTES lors de la séance 2 ne permettent pas de récupérer les informations sur la géolocalisation des parkings :

```
<Groupe_Parking>
  <Grp_identifiant>3</Grp_identifiant>
  <Grp_nom>TOUR DE BRETAGNE</Grp_nom>
  <Grp_statut>5</Grp_statut>
  <Grp_pri_aut>0</Grp_pri_aut>
  <Grp_disponible>268</Grp_disponible>
  <Grp_complet>20</Grp_complet>
  <Grp_exploitation>645</Grp_exploitation>
  <Grp_horodatage>04/02/2013 23:24:40</Grp_horodatage>
  <IdObj>299</IdObj>
</Groupe_Parking>
```

Comment obtenir ces informations sur la géolocalisation ?

Tous les équipements de la ville, chaque parking possèdent un identifiant <idObj> à partir duquel on peut obtenir les données de géolocalisation dans un fichier CSV :

« Equipement_publics_deplacement.csv »

- Récupérer ce fichier sur le site Open Data de Nantes.
- Ouvrez le avec Notepad et analyser son contenu et sa structure.
- Implémenter l'algorithme ci-dessous sur Processing permettant d'extraire de ce fichier les données de géolocalisation du parking « TOUR DE BRETAGNE » dont l'identifiant est « 299 ».
  Ces données doivent mises dans deux variables lat et lon de type String.

```
DEBUT

    Extraire chaque ligne du fichier csv et placer la chaîne de
    caractères correspondante dans un tableau.

    Pour i de 0 à nombre de lignes du fichier (c'est-à-dire la taille
    du tableau)Faire

        Décomposer la ligne en chaînes de caractère suivant le séparateur
        utilisé dans le fichier csv (caractère  ; ).

        Mettre en forme la chaîne de caractère correspondant au IdObj
        (exemple : «299,000 » => « 299 »)

        Si IdObj du CSV = IdObj du XML alors

            Récupérer la chaîne de caractère correspondant à la latitude.
            Récupérer la chaîne de caractère correspondant à la longitude.
            Afficher la variable lat et la variable lon

        Fin du Si
    Fin du Pour
FIN
```

B) Localiser un parking

Nous allons afficher sur la carte l'emplacement du parking signalé par un petit icône :

1. Récupérer les données XML du parking « TOUR DE BRETAGNE ».
2. Récupérer la géolocalisation du parking à partir du fichier csv et l'identifiant idObj (données XML).
3. Afficher la carte de NANTES.
4. Placer l'icône du parking sur la carte. Pour convertir les coordonnées de géolocalisation en coordonnées sur la carte, il faudra utiliser le code suivant :

```
int taille_icone =20;
float [] coordonnees;
Location position = new Location(lat, lon);
//initialisation de la position du parking sur la carte (en pixels)
coordonnees = carte.getScreenPositionFromLocation(position);
//affichage de l'icône de ce parking
image(datanantesPictoParking,coordonnees[0]-taille_icone/2,coordonnees[1] – taille_icone/2,
taille_icone,taille_icone);
```

Attention, les variables lat(latitude) et lon (longitude) sont de type String or l'instruction Location prend pour paramètres des float : Location (<float>,<float>).
Il va donc falloir effectuer une convertion : String en float. Pour cela, nous allons utiliser la fonction **split(***tableau de String***,** *séparateur***).** Elle permet de « casser » une chaine de caractère au niveau du séparateur. Par exemple pour la latitude 47,215 :

```
//on casse la chaine de caractère 47,215 au niveau du séparateur «  ,  » :  47 et 215
   String [ ] lat_casse = split(element[15], " , ") ;
// on reconstitue la chaine de caractère le séparateur «  .  » 47.215
 String lat_join = join(lat_casse, " . ") ;
//conversion (cast) de la chaine de caractère en type float
 float lat = float (lat_join) ;
```

Pour la longitude, on procède de manière identique en remplaçant element[15] par element[14].

Implémenter le programme complet en utilisant l'icône dans le fichier image fourni.

C) Rendre l'application interactive.

Implémenter un programme sur Processing permettant d'afficher le nombre de places disponibles dans le parking « TOUR DE BRETAGNE » lorsque l'utilisateur clique sur l'icône parking de la carte.

D) Finalisation

Implémenter un programme sur Processing permettant d'afficher la disponibilité de tous les parkings de NANTES lorsque l'utilisateur clique sur l'icône d'un parking de la carte.