

Martin DELSINNE


Gestionnaire Data / Développeur

martin.delsinne@geovendee.fr

PROGRAMMATION



Programmation

- 1 - Programmation orientée objet
- 2 - Environnement de développement
- 3 - Les variables
- 4 - Python 
- 5 - Les formulaires
- 6 – Les modules
- 7 – Les procédures
- 8 - Les choix : if...else
- 9 - Les répétitions
- 10 - Les tableaux - Les collections – Les listes
- 11 –PyQGIS

Langages de programmation

En informatique, un **langage de programmation** est une notation conventionnelle destinée à formuler des algorithmes et produire des programmes informatiques qui les appliquent.

Il existe de nombreux langages, ci-dessous, les plus populaires au 1^{er} janvier 2021 :

Rank	Change	Language	Share	Trend
1		Python	30.44 %	+1.2 %
2		Java	16.76 %	-2.0 %
3		JavaScript	8.44 %	+0.3 %
4		C#	6.53 %	-0.7 %
5	↑	C/C++	6.33 %	+0.3 %
6	↓	PHP	6.05 %	-0.2 %
7		R	3.87 %	+0.1 %
8		Objective-C	3.71 %	+1.2 %
9		Swift	2.14 %	-0.3 %
10		TypeScript	1.78 %	-0.0 %

Source : GitHub

Usages pour les Géomaticiens

Usage	Langage(s)
Prise en main de QGIS (console- autonome)	Python
Prise en main d'ArcGIS (console-autonome)	Python, VB.Net
Construction d'une page WEB	HTML5, JavaScript
Construction de pages WEB en interaction avec un BDD	PHP / JAVA
Manipulation de fichiers/dossiers	Python, C, VBScript, ...
Automatiser des tâches sur Excel	VBA
Création de fonctions (trigger) sur PostgreSQL	PL/pgSQL
Automatisation des tâches WINDOWS	MS-DOS - Tâche planifiée
Automatisation des tâches LINUX	Script shell - crontab

1 - Principes de base

1 – Programmation Orientée Objet

- * Pour créer/instancier un objet

var *monObjet* = new *ClasseDeReference*



*var monEtoile = new
Etoile*

Un objet est créé à partir d'un modèle nommé classe

- * Les propriétés (attributs) d'un objet

monObjet.nomPropriété

= données caractérisant l'objet



*monEtoile.couleur
→ retourne « Vert »*

- * Les méthodes (fonctions) d'un objet

monObjet.nomMéthode

= actions de l'objet



monEtoile.seDéplacer()

1 - Principes de base

1 – Programmation Orientée Objet

PYTHON	PLPGSQL	VB.NET
<ul style="list-style-type: none">Déclaration variable non nécessaire <code>V='Bonjour'</code>Le type de la variable sera défini lors de l'affectation de sa valeurUtilisation d'une méthode <code>V.lower()</code>	<ul style="list-style-type: none">Déclaration de la variable obligatoire <code>Declare V text; Begin V= 'Bonjour'; ...</code>Non orienté Objet	<ul style="list-style-type: none">Déclaration de la variable obligatoire <code>Dim v as String V = « Bonjour »</code>Utilisation d'une méthode <code>V.ToLower()</code>

1 - Principes de base

2 – Environnement de développement

IDE = Integrated Development Environment

Programme regroupant un ensemble d'outils pour le développement de logiciels.

Un EDI se compose généralement d'un éditeur de texte, d'un compilateur, d'outils automatiques et d'un débogueur.

PYTHON

- IDLE
- Wing IDE
- Sublime Text

= Langage interprété

VB.NET

- Visual Studio .Net
(Express Editions)
- SharpDevelop

= Langage compilé

1 - Principes de base

2 – Environnement de développement

IDE les plus populaire au 1^{er} Janvier 2021 :

Rank	Change	IDE	Share	Trend
1		Visual Studio	26.28 %	+3.4 %
2	↑	Eclipse	16.1 %	-1.4 %
3	↓	Android Studio	10.75 %	-8.6 %
4		Visual Studio Code	9.3 %	+3.4 %
5	↑	pyCharm	7.98 %	+2.4 %
6	↑	IntelliJ	5.91 %	+0.8 %
7	↓↓	NetBeans	5.51 %	-0.2 %
8		Xcode	4.31 %	-0.2 %
9	↑	Atom	3.78 %	+0.5 %
10	↓	Sublime Text	3.66 %	-0.2 %

Source : GitHub

1 - Principes de base

2 – Environnement de développement

Installation Python et EDI

Etape 1- Installation de Python

<https://www.python.org/>

<https://www.python.org/downloads/windows/>

- Personnaliser l'installation
- Définir l'emplacement de l'installation (location), par exemple : « *C:\python-3.8* »

Etape 2- Installation de WING IDE Python

<https://wingware.com/>

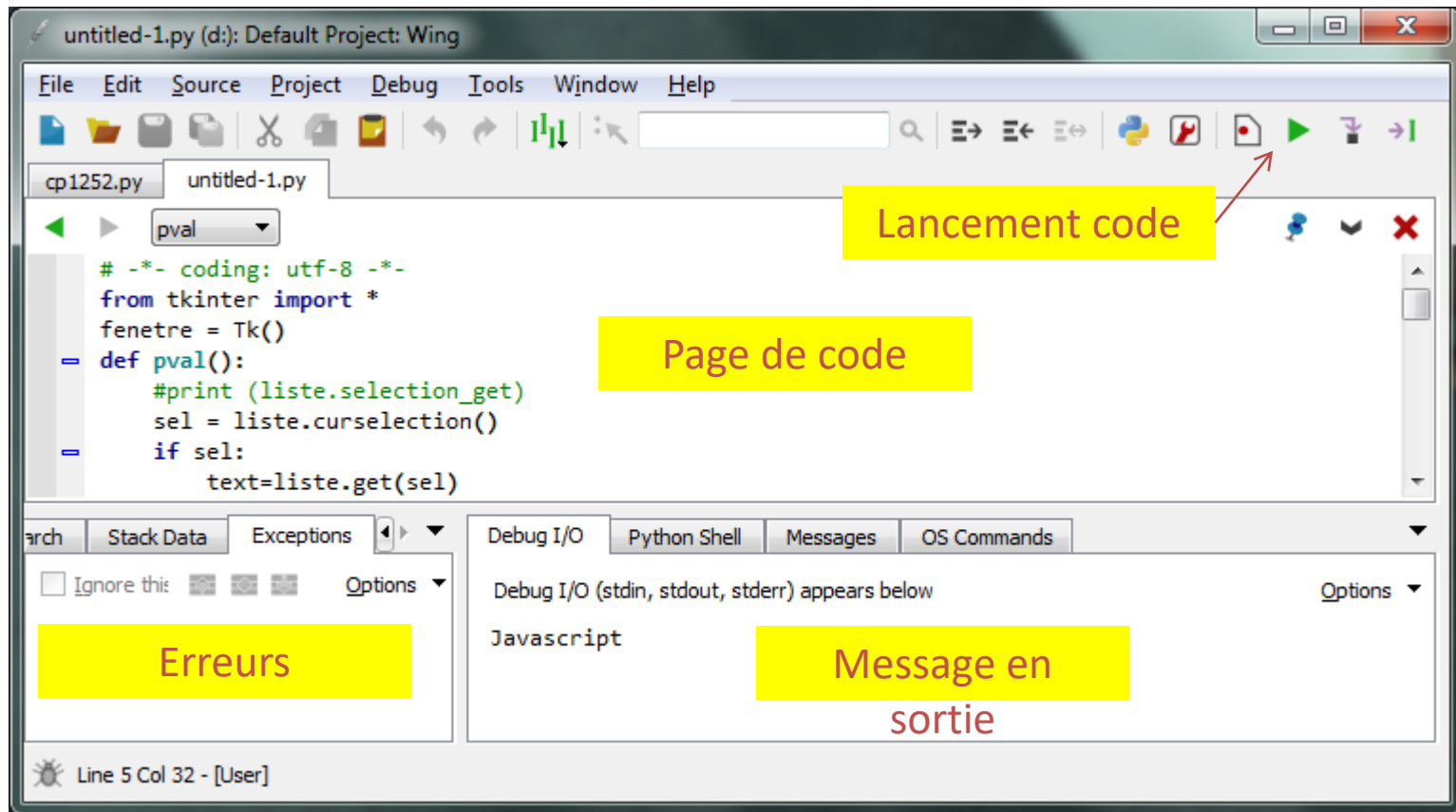
- Télécharger et décompresser l'archive zip (version « personal »)
- Lancer l'exécutable « *repertoire_install_wingide/bin/wing-personal.exe* »
- Définir l'emplacement de python



1 - Principes de base

2 – Environnement de développement

Interface WING IDE



1 - Principes de base

3 - les variables

Variable = espace de stockage pour un résultat

Les variables sont définies par un nom (identifiant) et un type (nature de l'information).

- Déclaration des variables

```
var i as integer  
var tab as array  
var txt as string
```

- Initialisation des variables

```
i= 10  
tab = [« 1er élément », « 2nd élément », false, 14.5684, « 5ième  
élément »]  
txt = « Bonjour »
```

1 - Principes de base

3 - les variables (types)

PYTHON	PLPGSQL	VB.NET
<ul style="list-style-type: none">• Entier• Réel• Chaîne de caractères• Liste	<ul style="list-style-type: none">• Integer• Booleen• Varchar• Text• Double precision• Numeric• Record (= enregistrement)	<ul style="list-style-type: none">• Type numérique : Integer - Double• Type alphanumérique : String - Char• Boolean• Date• Objet• Type complexe/structuré: Array - Collections

1 - Principes de base

4 - Python

Pas de déclaration des variables

L'affectation des variables définit le typage

$V = \text{« Bonjour »} \rightarrow \text{type texte}$

$V = 1 \rightarrow \text{type entier}$

$V = 1.2568 \rightarrow \text{type réel}$

$V = ['a', 'b', 'test'] \rightarrow \text{type liste}$

Une variable peut-être affectée par une autre variable. (exemple: $V = V2$)

1 - Principes de base

4 - Python

Print() → Fonction qui affiche des données en sortie

Les opérateurs de calcul	
+	Additionner
-	Soustraire
*	Multiplier
/	Diviser
Les opérateurs de comparaison	
==	Est égal
!=	Est différent
> , <	Est supérieur , Est inférieur

1 - Principes de base

4 - Python

Les fonctions de chaines de caractères	
+	Concaténer 2 chaînes d caractères
Len()	Longueur de la chaine
Int(), float()	Convertir Chaine en numérique
<i>TXT</i> [n° cdebut : n° cfin+1]	Extraction chaine
Quelques méthodes sur les chaines de caractères	
capitalize()	rsplit()
Isdecimal()	lower() / upper()
count()	find()
Isnumeric()	replace()
https://docs.python.org/fr/3.6/library/string.html	
Les caractères spéciaux	
\n	Est égal
\t	Est différent
\", \'	Est supérieur , Est inférieur

1 - Principes de base

4 - Python

- **Commenter** le code

python

Ligne pseudo-commentaire pour définir l'encodage du fichier de code:

```
# -*- coding:latin-1 -*-
```

```
# -*- coding:utf-8 -*-
```

- Indenter le code. **Indentation obligatoire** pour le langage python)

Condition:

..... Action

- Décomposer au maximum le code (dans sa compréhension et dans sa structuration)

EXERCICE 1

1. **Créer 7 variables** en affectant les valeurs suivantes. Utiliser la fonction `print()` pour restituer les valeurs des variables

Nom de la variable	Valeur
v1	Bonjour
v2	4
v3	10,25
v4	un , deux, trois, quatre, cinq
v5	L'addition de v2 et de v3
v6	été
v7	tout le monde

Inclure dans le code le pseudo-commentaire d'encodage `# -*- coding: utf-8 -*-` et des `#commentaires`.

2. **Manipuler les chaines de caractères**

- 21/ Remplacer le caractère « é » par le caractère « e » pour la variable v6
- 22/ Dans une nouvelle variable nommée v8, concaténer les variables v1 et v7 en ajoutant un espace entre les 2
- 23/ Afficher le nombre de caractères pour la variable v1
- 24/ Mettre en majuscule la variable v7
- 25/ Dans une nouvelle variable nommée v9, extraire du 7^{ème} caractère au 13^{ème} caractère de la variable v7
- 26/ Supprimer les espaces en début et en fin de chaine pour la variable v9
- 27/ Décomposer, au niveau de chaque espace, la variable v7 en liste. Utiliser la fonction `split()`

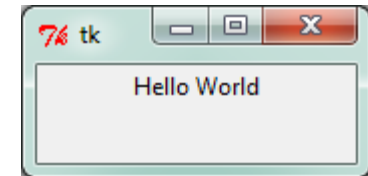
1 - Principes de base

5 - les formulaires

Plusieurs modules disponibles pour créer des interfaces:

Utilisation du module « tkinter »

```
# -*- coding: utf-8 -*-  
from tkinter import *  
fenetre = Tk()  
label = Label(fenetre, text="Hello World")  
label.pack()  
fenetre.mainloop()
```



<http://apprendre-python.com/page-tkinter-interface-graphique-python-tutoriel>

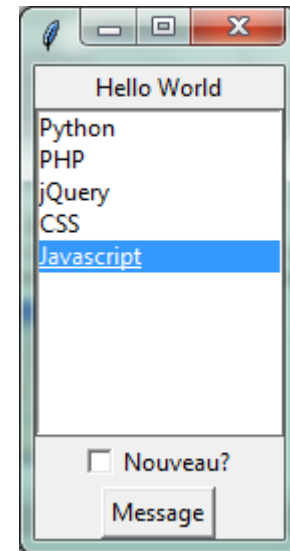
https://www.xavierdupre.fr/app/teachpyx/helpsphinx/c_gui/tkinter.html

<https://python-django.dev/page-tkinter-interface-graphique-python-tutoriel>

1 - Principes de base

5 - les formulaires

```
# -*- coding: utf-8 -*-
from tkinter import *
fenetre = Tk()
# label
label = Label(fenetre, text="Hello World")
label.pack()
# liste
liste = Listbox(fenetre)
liste.insert(1, "Python")
liste.insert(2, "PHP")
liste.insert(3, "jQuery")
liste.insert(4, "CSS")
liste.insert(5, "Javascript")
liste.pack()
# boutonchk
boutonchk = Checkbutton(fenetre, text="Nouveau?")
boutonchk.pack()
# bouton
bouton=Button(fenetre, text="Message", command=fenetre.quit)
bouton.pack()
fenetre.mainloop()
```

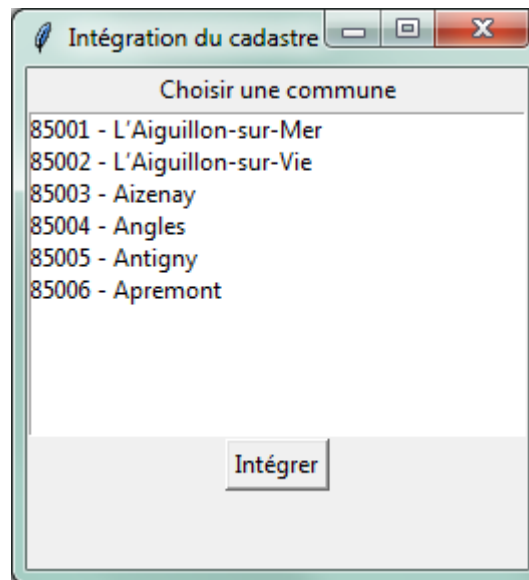


La méthode pack() permet d'ajouter le composant dans la fenêtre

La méthode mainloop() permet d'afficher le formulaire

EXERCICE 2

1. Créer le formulaire suivant avec le module « tkinter »



Complément: Utiliser les méthodes suivantes pour paramétrer le formulaire

`fenetre.title("...")`

`fenetre.geometry('LongueurxLargeur')`

1 - Principes de base



6 - les modules

* Sur Python, chaque module est stocké dans un fichier d'extension py.

* Code pour importer un module

```
import nomdumodule
```

OU

```
from nomdumodule import * / from nomdumodule import nom_classe
```

- Modules internes du langage Python (<http://www.jchr.be/python/modules.htm>)
- **1. Système**
 - 1.1 sys
 - 1.2 time
 - 1.3 os et os.path
- **2. Nombres**
 - 2.1 (c)math (trigo, log)
 - 2.2 random (hasard)
 - 2.3 decimal
- **3. Chaînes**
 - 3.1 string (chaînes)
 - 3.2 unicode (chaînes)
 - 3.3 re (expressions régulières)
 - 3.4 curses (affichage console)

1 - Principes de base



6 - les modules

Pour ajouter des modules complémentaires à l'installation initiale:

Possibilité 1

- Décompresser une archive & Copier le répertoire dans le chemin :
Rep_Install_Python\Lib\site-packages

Possibilité 2

- Installation avec le module PiP

Utilisation de l'interface MS DOS (cmd)

```
cd C:\Rep_install_python\Scripts  
Pip install nom_module
```

✓ Liste des modules complémentaires: <https://pypi.org/>

1 - Principes de base



6 - les modules

Modules couramment utilisés

module	description
sys	Paramètres et fonctions propres au système
os	Manipulation des fichiers et des dossiers
psycopg2	Interface avec PostgreSQL
urllib.request	Requêtes HTTP
zipfile	Compression / Décompression de fichier
shapefile	Shapefile librairie

EXERCICE 3

1. Manipuler les fichiers

11/ Tester la présence d'un chemin

12/ Tester si le chemin est un fichier et si le chemin est un répertoire

13/ Lister les fichiers et les répertoires à partir de la racine d'un répertoire

14/ Créer un répertoire

15/ Créer un fichier et intégrer la chaîne de caractères « Bonjour tout le monde »

16/ Lire le contenu du fichier précédemment créé

Complément: Pour manipuler les fichiers et les dossiers

`os.path.exists(«chemin»)` → Le chemin est-il existant ? TRUE/FALSE

`os.path.isdir(«chemin»)` → Le chemin est un répertoire

`os.path.isfile(«chemin»)` → Le chemin est un fichier

`os.listdir(«repertoire»)` → Liste les fichiers et répertoires à la racine du rép

`os.mkdir(«repertoire»)` → Créer un répertoire

`fw = Open(«chemin»,«w»)` → Ouvrir un fichier en mode d'écriture

`fw.write(«texte»)` → Ecrire dans un fichier ouvert en mode écriture

`fr = Open(«chemin»,«r»)` → Ouvrir un fichier en mode lecture

`fr.readline()` / `fr.read()` → Lire un fichier

1 - Principes de base

7 - les procédures

* Une procédure (ou fonction) est un sous-programme qui permet d'effectuer un ensemble d'instructions par simple appel dans le corps du programme principal.

→ exécute des instructions

→ Possède 0 ou plusieurs arguments en entrée

→ Retourne aucun ou 1 résultat en sortie

* Les procédures peuvent être appelée plusieurs fois dans un programme (+ pour la simplicité & la taille du code) et permettent de découper un problème complexe.

1 - Principes de base

7 - les procédures

* Déclaration d'une procédure

```
Function NomProcedure (Argument1, Argument2, ...)  
    Instructions  
End Function
```

* Exemple d'utilisation :

```
Function FrancEuro (prix as double) as double  
    return prix/6.55957  
End Function
```

1 - Principes de base



7 - les procédures

* Syntaxe en Python

```
def maFonctionAddition (arg1 , arg2) :  
    # mes instructions  
    value = arg1 + arg2  
    # valeur retournée  
    return value  
  
print (maFonctionAddition (2 , 7) )  
  
>>> 9
```

1 - Principes de base

7 - les procédures et les fonctions

PYTHON

- Créer une fonction et une procédure

```
def indique_mon_age():  
    return 30
```

- Appeler une fonction

```
indique_mon_age()
```

PLPGSQL

- Créer une fonction

```
CREATE OR REPLACE  
FUNCTION  
indique_mon_age()  
    RETURNS integer AS  
$BODY$begin  
return 30;  
end;$BODY$  
LANGUAGE plpgsql
```

- Appeler une fonction

```
SELECT indique_mon_age();
```

Possibilité de créer des fonction sur
pgAdmin



VB.NET

- Créer une fonction

```
Function  
indique_mon_age() as  
integer  
    indique_mon_age =3  
End Function
```

- Appeler une fonction

```
indique_mon_age()
```

EXERCICE 4

1. Créer une fonction permettant de convertir les francs en euros
2. Créer une fonction permettant de supprimer les caractères accentués dans une chaîne de texte
3. Créer une procédure permettant de récupérer, dans la liste du formulaire de l'exercice 2, le nom de la commune sélectionnée

Complément: Utiliser les méthodes suivantes pour interagir avec la liste

liste.curselection() → Récupère l'index de la liste sélectionnée

liste.get(index) → Récupère la valeur de la liste à partir d'un index

Complément: Pour associer une procédure au clic sur le bouton d'un formulaire

bouton=Button(fenetre, text="Message", command=nom_procedure)

1 - Principes de base

8 - les choix : if...else

Syntaxe (VB.NET)

If Condition Then

Instruction

Else

Autre instruction

End If

Définition

Si Condition est vrai Alors

Effectuer cette instruction

Sinon

Effectuer cette autre instruction

Fin Si

VB.NET

```
Dim numero as integer
numero = 4
If numero < 5 then
    numero = numero*2
Else
    numero = numero/2
End If
-----
numéro =
```

1 - Principes de base

8 - les choix - If

Syntaxe (Python) :

If Condition :

Instruction

Else:

Autre instruction

```
numero = 4
If numero < 5:
    numero = numero*2
Else:
    numero = numero/2
```

Python

Syntaxe (PL/pgSQL) :

If Condition **then**

Instruction;

Else

Autre instruction;

End If;

```
numero integer;
numero := 4;
If numero < 5 then
    numero = numero*2
Else
    numero = numero/2
End If
```

PL/pgSQL

1 - Principes de base

8 - les choix - Select Case

Syntaxe (VB.NET)

Select Case Valeur

Case Condition1

Instruction1

Case Condition2

Instruction2

End Select

Définition

Sélectionner

Le cas 1 si condition1 est vrai

Le cas 2 si condition2 est vrai

```
Dim i as integer
i = 12/4
Select Case i
    Case 0 to 2
        msgbox(«la valeur est
               comprise entre 0 et 2 »)
    Case 3 to 5
        msgbox(«la valeur est
               comprise entre 3 et 5 »)
End Select
```

VB.NET

Python

Syntaxe non existante

1 - Principes de base

9 - les répétitions - For (nombre min / max)

Syntaxe (VB.NET)

For i = 0 to 10

Instruction

Next i

Définition

Pour i allant de 0 à 10, Répéter

Instruction

Fin Répéter

VB.NET

Exemple :

Afficher successivement les valeurs de 0 à 10

```
Dim i as integer
For i = 0 to 10
    MsgBox(i)
Next
```

1 - Principes de base

9 - les répétitions - For (nombre min / max)

Syntaxe (Python)

For i in range (0,11):

Instruction

```
For i in range(0,11):  
    print(i)
```

Python

Syntaxe (PL/pgSQL)

for i in 0..10 loop

Instruction;

end loop;

```
begin  
j=0;  
for i in 0..10 loop  
    j= j+1;  
end loop;  
return j;  
end;
```

PL/pgSQL

1 - Principes de base

9 - les répétitions - For (sur un objet)

Syntaxe (VB.NET)

For Each Élément **in** objet

Instruction

Next

Définition

Pour chaque élément de l'objet

Instruction

Fin

Exemple:

Pour boucler sur les éléments d'une collection

```
Dim cl As new Collection
cl.Add("UN")
cl.Add("DEUX")
cl.Add("TROIS")
For Each val As String In cl
    msgbox(val)
Next
```

VB.NET

1 - Principes de base

9 - les répétitions - For (sur un objet)

Syntaxe (Python)

For Élément **in** objet:
Instruction

```
for letter in 'Python':  
    print 'Current Letter :', letter  
fruits = ['banana', 'apple', 'mango']  
for fruit in fruits:  
    print 'Current fruit :', fruit
```

Python

Syntaxe (PL/pgSQL)

For Élément **in** objet **Loop**
Instruction;
End Loop;

```
for r in SELECT nom from matable Loop  
    nom_agg = nom_agg || r.nom;  
End Loop;
```

PL/pgSQL

1 - Principes de base

VB.NET

10 - les tableaux

* Un tableau permet de stocker plusieurs variables de même type sous un même nom de variable, chaque élément étant repéré par un index ou indice.

Déclaration d'un tableau

Dim T(4) as integer

12

Affecter une valeur à un élément

36

T(1) = 36

58

Pour accéder à un élément du tableau

61

Dim v as integer = T(0)

(v=12)

Remarque: Le premier élément comporte l'index 0

Possibilité de déclarer des tableaux multidirectionnels

Dim T(4,2) as integer

1 - Principes de base

VB.NET

10 - les collections

* Une collection permet de stocker plusieurs variables ou objets, chaque élément étant repéré par un index ou indice. Les collections n'ont pas de nombre d'éléments précis au départ.

Déclaration d'une collection

Dim Col as new collection

titi

toto

lulu

dodo

Ajouter un élément

Col.add(« titi »)

Supprimer un élément

Col.remove(1)

Remarque: Le premier élément comporte l'index 1

1 - Principes de base



10 - les listes (list/array)

* Liste = variable dans laquelle on peut mettre plusieurs valeurs

Déclaration d'une liste

Liste = []

1

Affecter valeurs à la liste

2

Liste = [1,2,3]

3

Liste.append(21)

21

Pour accéder à un élément du tableau

v = Liste[0]

(v=1)

Remarque: Le premier élément comporte l'index 0

* Autres types de séquences python: tuples, dictionnaires ...

EXERCICE 5

1. Les conditions

Créer une fonction qui teste un chemin (placé en paramètre) et qui retourne les valeurs suivantes

- Si le chemin est existant
 - Si le chemin est un fichier → Le chemin est un fichier
 - Si le chemin est un dossier → Le chemin est un dossier
- Si le chemin est non existant → Le chemin est inexistant

2. Les Boucles

Lister les fichiers et les dossiers à la racine d'un répertoire.

Boucler sur les résultats, pour intégrer chaque valeur dans un fichier texte nommé listerep.txt

3. Gestion des listes

Créer une liste avec les valeurs 0,3,25,624

Ajouter à la liste la valeur 2365

Récupérer la dernière valeur ajoutée

1 - Principes de base

11 – PyQGIS

Console Python



```
Console Python
1 Console Python
2 Utilisez iface pour accéder à l'interface de l'API QGIS ou tapez hel
  p(iface) pour plus d'informations
3 Alerte de sécurité : saisir des commandes à partir d'une source non
  approuvée peut mener à la perte et/ou la fuite de données
4 >>> # Charger une couche vecteur SHAPEFILES
5 >>> vlayer = iface.addVectorLayer(r"C:\Users\martin.delsinne\Desktop
  \Cours Programmation\2020-2021\ressources
  \Communes", "ogr")
6
>>>
```

Ouverture d'un
shapefile sur la carte

<https://docs.qgis.org/3.4/pdf/fr/QGIS-3.4-PyQGISDeveloperCookbook-fr.pdf>

Charger une couche

```
layer = iface.addVectorLayer("C:\\communes.shp", "communes", "ogr")
```

Exporter une couche

```
export = QgsVectorFileWriter.writeAsVectorFormat(layer, r"c:\export-
communes.shp", "utf-8", QgsProject.instance().crs(), "ESRI Shapefile", False)
```

1 - Principes de base

11 – PyQGIS

<https://qgis.org/pyqgis/3.4/core/QgsProject.html>

map : QgsProject :

map = QgsProject.instance() # QGIS v3 ou supérieur

map.count() : retourne le nombre de couches (sous la forme d'un entier)

map.crs() : retourne le système de coordonnées (classe QgsCoordinateReferenceSystem)

map.mapLayers() : retourne une liste (dictionnaire) des couches de la carte

layer : QgsMapLayer :

extent() : retourne l'emprise de la carte (classe QgsRectangle)

Name() : retourne le nom de la couche

id() : retourne l'identifiant unique interne de la couche (classe QString)

isEditable() : retourne True si la couche peut être modifiée

isSpatial() : retourne True si la couche est pourvue d'une géométrie

isValid() : retourne False si la couche a un problème (mauvaise source de données par exemple)

maximumScale() : retourne l'échelle maximum de visibilité (sous la forme d'un float)

minimumScale() : retourne l'échelle minimum de visibilité (sous la forme d'un float)

source() : retourne la source (classe QString)

type() : retourne le type de couche

layer : QgsVectorLayer :

featureCount() : retourne le nombre d'éléments (sous la forme d'un entier)

fields() : retourne la liste des attributs (classe QgsFields)

geometryType() : retourne le type de géométrie (point, ligne, polygone, pas de géométrie)

selectedFeatures() : retourne la liste des éléments sélectionnés (classe QgsFeatureList)

Quelques liens

Généraliste

<http://www.developpez.com/>

<https://openclassrooms.com/fr/>

<https://github.com/>

VB.NET

<http://www.vbfrance.com/>

Python

<http://www.jchr.be/python/>

<http://apprendre-python.com/>

<https://openclassrooms.com/courses/apprenez-a-programmer-en-python>

<https://www.python.org/>

Python & QGIS

<http://www.geoinformations.developpement-durable.gouv.fr/qgis-le-coin-des-developpeurs-r727.html>

<https://qgis.org/pyqgis/3.4>

IDE

<https://code.visualstudio.com/>

<http://www.icsharpcode.net/opensource/sd/>

<http://www.microsoft.com/express/downloads/>

Retrouver l'ensemble du cours, ressources et exercices à l'adresse suivante :

<https://github.com/geovendee/formation-prog-20212022>

MERCI

