L'Association pour le Développement Informatique et la Communication LADIC.com

Formation Python / Django

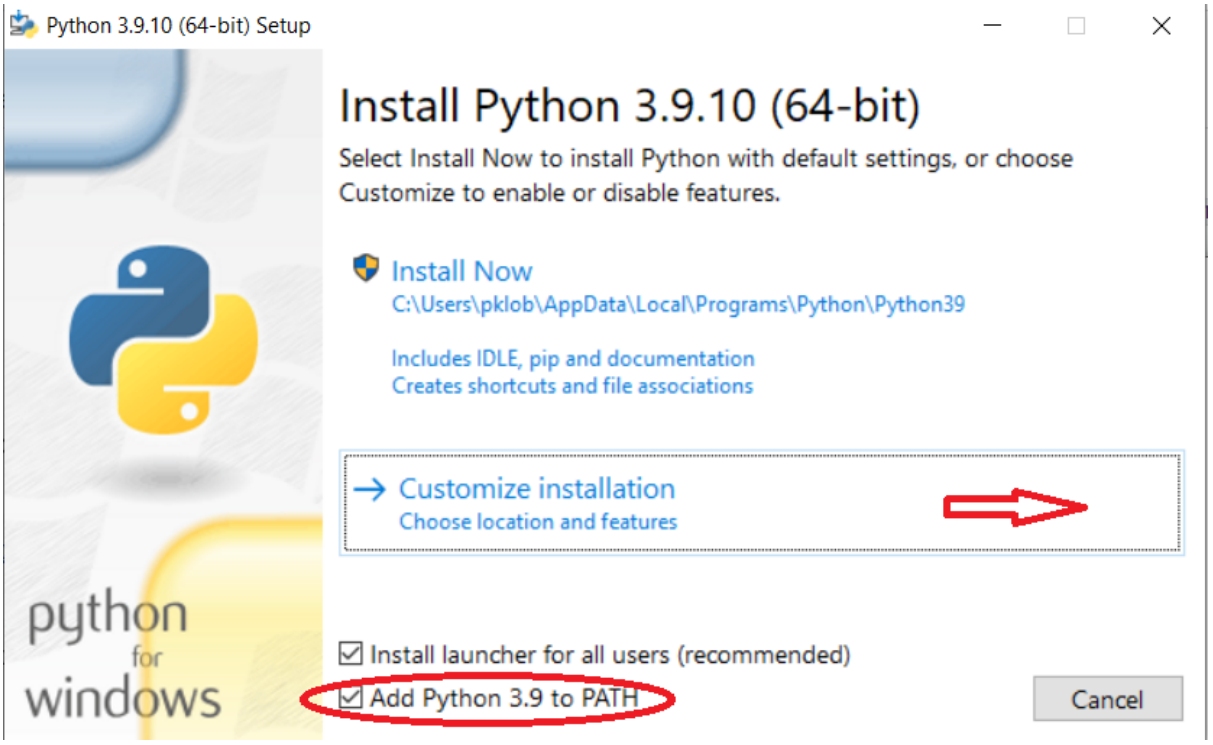
1) Préambule

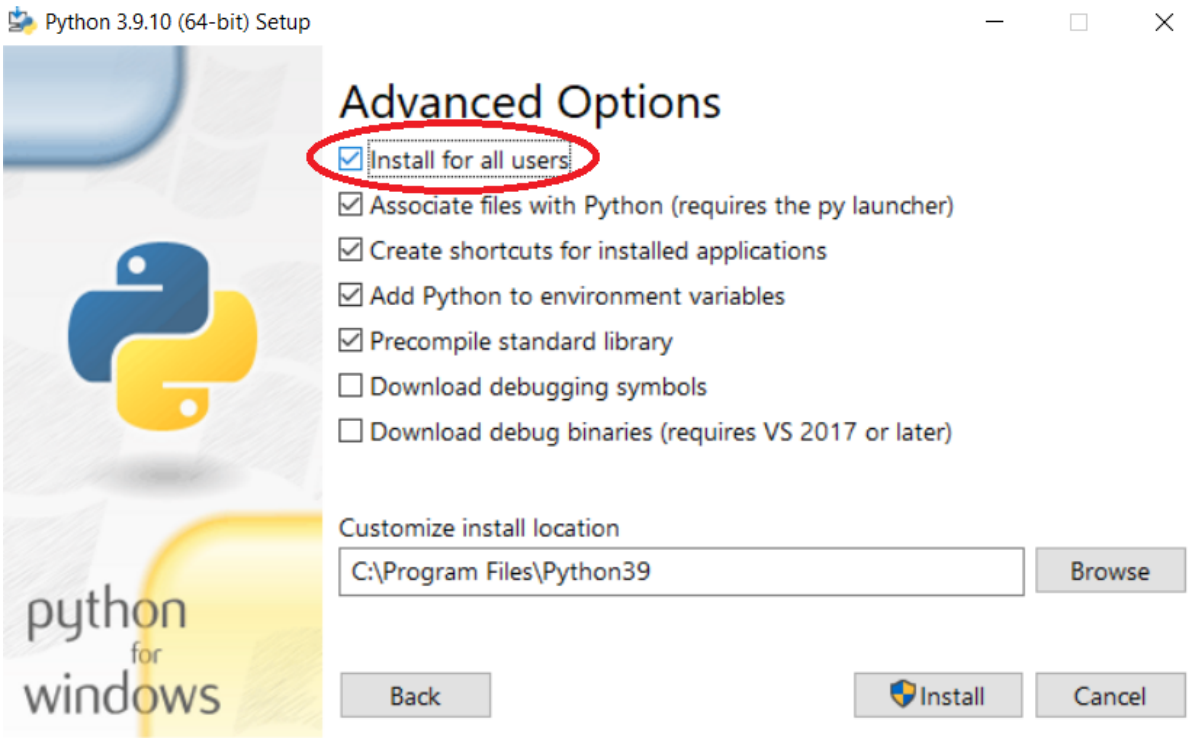
Cette formation, sous forme de tutoriel, vous permet d'acquérir les bases nécessaires en Python pour créer votre première application Django.

2) Installation de Python

Télécharger et installer Python **3.9.10**, c'est la version à utiliser pour cette formation :

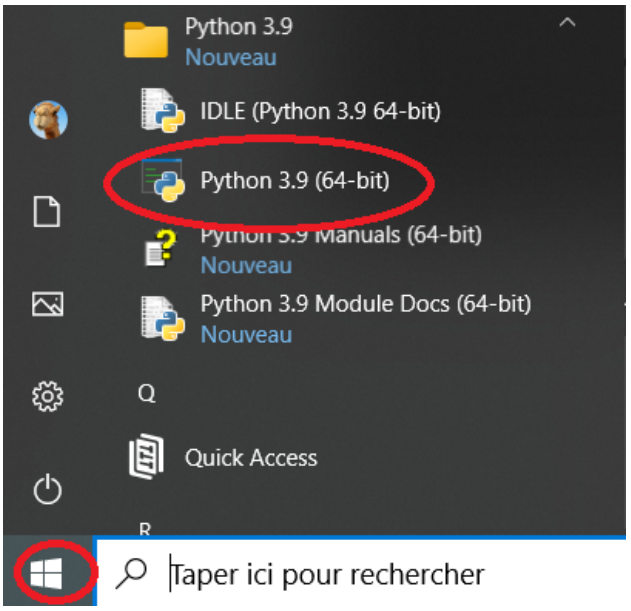
https://www.python.org/ftp/python/3.9.10/python-3.9.10-amd64.exe



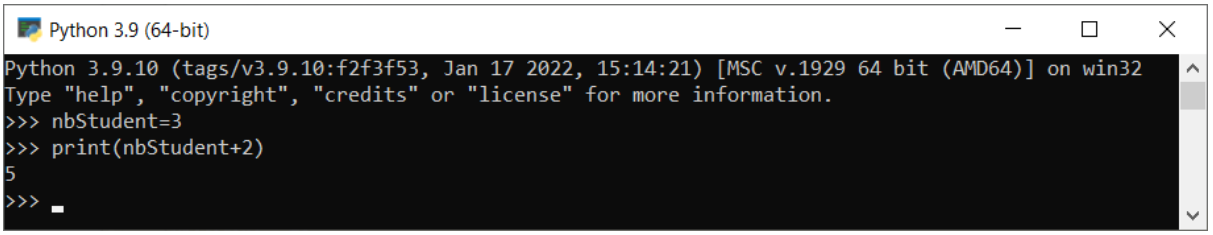


3) Les bases du langage Python

Une fois l'installation terminée, vous cliquez sur Windows "Démarrer" puis vous lancez Python :



Tapez des instructions Python :



Et suivez et exercez vous avec ce mini tuto de 13 minutes sur les notions de base de Python : https://www.youtube.com/watch?v=_Xu2W9nsmKg

Pour profiter pleinement de la formation, vous devez maîtriser les variables, les listes, les boucles et les conditions.

4) Objets et classes

Connaître les classes d'objets en Python :
https://www.youtube.com/watch?v=19yqBzPGLk8&t=862s

5) Exercice : convertir un montant en devises

a) En utilisant les boucles, terminer le programme suivant pour afficher le montant converti dans les 4 devises en fonction du taux :

```
listeDesDevises = ["Dollar USD", "Rouble RUB", "Livre sterling GBP", "Franc suisse CHF"]
listeDesTauxEur = [1.11        , 116.22      , 0.84               , 1.03 ]
montantEnEuros = input("Entrez le montant en euros à convertir en devises : ")
```

b) Même exercice en utilisant en plus une classe et sa méthode :

```
class Devise:
    def __init__(self, nom, tauxEur):
```

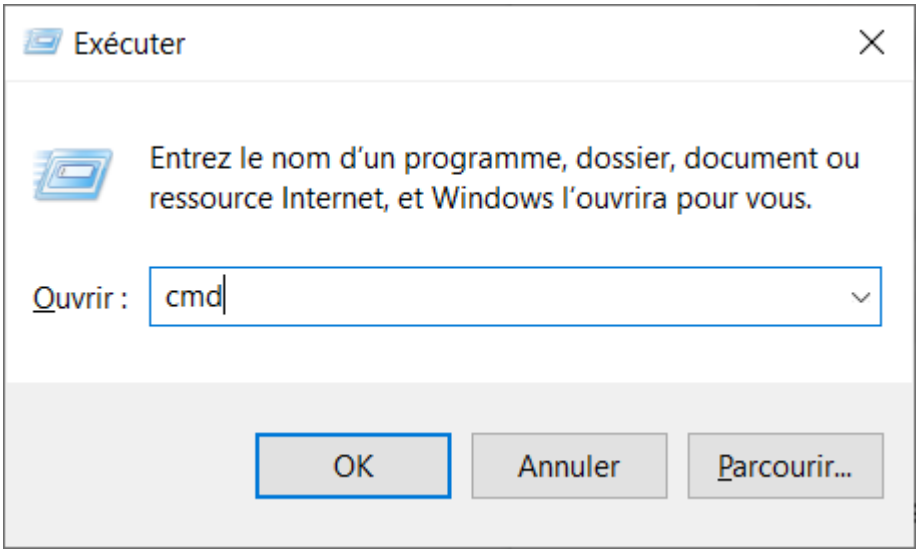
```
        self.nom = nom
        self.tauxEur = tauxEur
    def convertirEnDevise(self, montantEur):
# finir le programme...
```

c) Une solution :

```
listeDesDevises = ["Dollar USD", "Rouble RUB", "Livre sterling GBP", "Franc suisse CHF"]
listeDesTauxEur = [1.11        , 116.22      , 0.84                , 1.03 ]

montantEnEuros = input("Entrez le montant en euros à convertir en devises : ")

class Devise:
    def __init__(self, nom, tauxEur):
        self.nom = nom
        self.tauxEur = tauxEur
    def convertirEnDevise(self, montantEur):
        print(self.nom+' : ',float(montantEur)*self.tauxEur)

for d in listeDesDevises:
    i=listeDesDevises.index(d)
    print(listeDesTauxEur[i]*float(montantEnEuros),d)
    Devise(d,listeDesTauxEur[i]).convertirEnDevise(montantEnEuros)
```

6) Installation de Django

Après l'installation de Python, démarrez l'invite de commande (Windows+R cmd) puis tapez:



Vérifiez la version :

```
...\> py --version
```

Créez un dossier “pythonDjango”

```
...\> mkdir pythonDjango
```

```
...\> cd pythonDjango
```

Créez l’environnement virtuel

```
...\> py -m venv myproject
```

```
...\> .\myproject\Scripts\activate.bat
```

L’environnement virtuel sera activé et vous verrez « (myproject) » à côté de l’invite de commande pour vous en convaincre. Chaque fois que vous lancez une nouvelle invite de commande, vous devrez activer à nouveau l’environnement.

Installez django 4.0.3

```
...\> py -m pip install Django
```

7) Première application

```
...\> django-admin startproject mysite .
```

```
Attention au point “.” ci-dessus
```

```
...\> py manage.py runserver
```

Lancer http://127.0.0.1:8000/

```
...\> py manage.py startapp myapp
```

Écrivons la première vue. Ouvrez le fichier **myapp/views.py** et placez-y le code Python suivant :

```
from django.http import HttpResponse


def index(request):
    return HttpResponse("Hello, world. You're at the myapp index.")
```

Dans le fichier **myapp/urls.py**, insérez le code suivant :

```
from django.urls import path


from . import views


urlpatterns = [
    path('', views.index, name='index'),
]
```

L’étape suivante est de faire pointer la configuration d’”URL racine vers le module **myapp.urls**. Dans **mysite/urls.py**, ajoutez une importation **django.urls.include** et insérez un appel **include()** dans la liste **urlpatterns**, ce qui donnera :

```
from django.contrib import admin
from django.urls import include, path

urlpatterns = [
    path('myapp/', include('myapp.urls')),
    path('admin/', admin.site.urls),
]
```

Démarrez le server

```
...\> py manage.py runserver
```

puis http://127.0.0.1:8000/myapp/

8) Installation sur un serveur de production

Pensez à ajouter le nom de domaine du serveur dans mysite/settings.py :
ALLOWED_HOSTS = ["mon-domain.com","127.0.0.1","localhost"]

Suivez https://kb.planethoster.com/guide/astuces-techniques/installer-django-sur-world/ jusqu'à l'installation de Django puis copiez les dossiers mysite et myapp sur le server avec ce fichier passenger_wsgi.py à la racine :

```
Modifiez le fichier passenger_wsgi.py :
```
import mysite.wsgi
application = mysite.wsgi.application

```
Ajouter les lignes suivantes à la fin du fichier .htaccess :
```
PassengerFriendlyErrorPages on
PassengerAppEnv development

9) Créer un model

```
...\> py manage.py migrate
```

```
...\> py manage.py createsuperuser
```

Connecter vous avec le user: http://127.0.0.1:8000/admin/

Ajoutez ‘myapp’ dans la liste INSTALLED_APPS de mysite/settings.py

Créer le modèle Client dans myapp/models.py:

```
from django.db import models

# Create your models here.
class Client (models.Model):
    name = models.CharField(max_length=25)
    email = models.EmailField(max_length=40)
    phone = models.IntegerField()
```

Rendre le modèle visible dans l’admin myapp/admin.py:

```
from django.contrib import admin

# Register your models here.
from .models import Client
admin.site.register(Client)
```

```
...\> py manage.py makemigrations myapp
```

Vérifiez le model dans l’admin: http://127.0.0.1:8000/admin/

10) Créer un formulaire et enregistrer les données

Créer le fichier myapp/myform.py :

```
from django.forms import ModelForm
from .models import Client

class ClientForm(ModelForm):
    class Meta:
        model = Client
        fields = ['name','email','phone']
```

Modifiez les lignes suivantes dans mysite/settings :

```
import os
TEMPLATES = [
   {
      'BACKEND': 'django.template.backends.django.DjangoTemplates',
      'DIRS': [os.path.join(BASE_DIR, "templates")],
```

Modifier myapp/views.py :

```
from django.shortcuts import render, redirect
from django.http import HttpResponse
from django.template import loader
from .models import Client
from .myform import ClientForm

def index(request):

   if request.method == "POST":
      form = ClientForm(request.POST).save()
      return redirect('/myapp')

   else:
      form = ClientForm()
      return render(request,'index.html',{'form':form})
```

Créer le fichier templates/index.html :

```
<!DOCTYPE html>
<html>
<head>
<title>Title of the document</title>
</head>

<body>
<h1>Formulaire client</h1>
<form method="POST" class="post-form">
        {% csrf_token %}
        {{form.as_p}}
        <button type="submit" class="save btn">OK</button>
</form>


</body>

</html>
```

Ajoutez des clients : http://127.0.0.1:8000/myapp/

11) Lister les données

Pour lister les clients en dessous du formulaire, on peut ajouter une clé 'dataClients' dans myapp/views.py :

return render(request,'index.html',{'form':form**,'dataClients':Client.objects.all()**})

Puis parcourir ces données clients dans le templates/index.html :

{% for c in dataClients %}
**{{ c.name }} {{ c.email }} {{ c.phone }}
**
{% endfor %}

Analyse des données

On peut aussi voir les données en glissant le fichier db.sqlite3 dans https://inloop.github.io/sqlite-viewer/

Exercise :

1) écrire la requête SQL qui affiche le nom de domaine de chaque adresse email client.
2) écrire la requête SQL qui compte le nombre de clients par nom de domaine.

Solution :

1) SELECT SUBSTR(email,CHARINDEX('@',email)+1) AS Domain FROM 'myapp_client'
2) SELECT SUBSTR(email,CHARINDEX('@',email)+1) AS Domain, count(email) FROM 'myapp_client' GROUP BY Domain

12) Supprimer les données

Ajouter le path suivant dans myapp/urls.py :

path('delete_client/<client_id>', views.delete_client, name='delete-client'),

Ajouter delete_client dans myapp/views.py :

```
def delete_client(request,client_id):
    client=Client.objects.get(pk=client_id)
    client.delete()
    return redirect('/myapp')
```

Ajouter le lien suivant dans la boucle for dans templates/index.html :

```
<a href="{% url 'delete-client' c.id %}">X</a>
```

Vous pouvez télécharger l'ensemble des fichiers sources de cette formation à cette adresse : https://ladic.com/pythonDjango/pythonDjango.zip

13) Documentation utilisée pour créer la formation

Installation de Python :
https://docs.djangoproject.com/fr/4.0/howto/windows/

Install Python 3.9.10
https://www.python.org/downloads/release/python-3910/

Première app
https://docs.djangoproject.com/fr/4.0/intro/tutorial01/

Model
https://www.youtube.com/watch?v=aw2mlbQWXec

Template
https://www.youtube.com/watch?v=nCPuMQc7A-I

Form
https://www.youtube.com/watch?v=M8N6ud_ok6g

Lister les données
https://www.youtube.com/watch?v=iqux7GJtQzM

Supprimer les données
https://www.youtube.com/watch?v=u1j-_kDc6g0