



XWDB Excel **Windev** Python Java .NET C# Bases de données Divers

## Windev & Python

Depuis la version 27 de Windev, vous pouvez appeler des fonctions écrites en Python comme des fonctions natives de Windev. Cela ouvre de nouvelles possibilités, je pense en particulier à l'utilisation des bibliothèques de traitement des données comme numpy et pandas.

Mais avec la version 28, tout est fonctionnel et utilisable ?

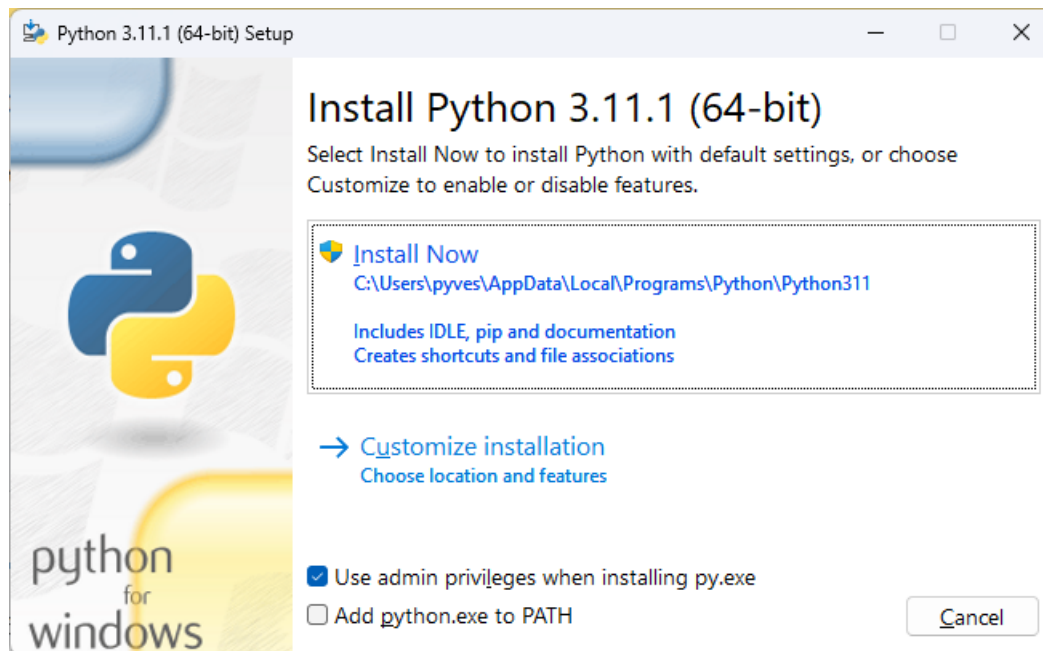
L'exemple "WD Python" fourni par PC Soft nous permet de très rapidement prendre en main cette nouveauté.

Vous trouverez dans le dossier "Exe" du projet le module "exemple.py".

## Installer l'interpréteur Python

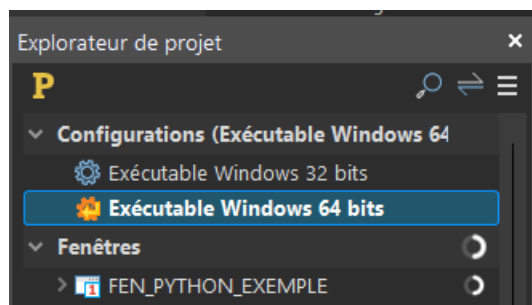
La première étape est d'installer sur votre PC si ce n'est pas déjà le cas, un interpréteur Python. Pour cela, je vous conseille de télécharger la version 3.11 à partir de: <https://www.python.org/downloads/release/python-3111/>

Notez le chemin d'installation dans le dossier système "AppData" de votre profil:

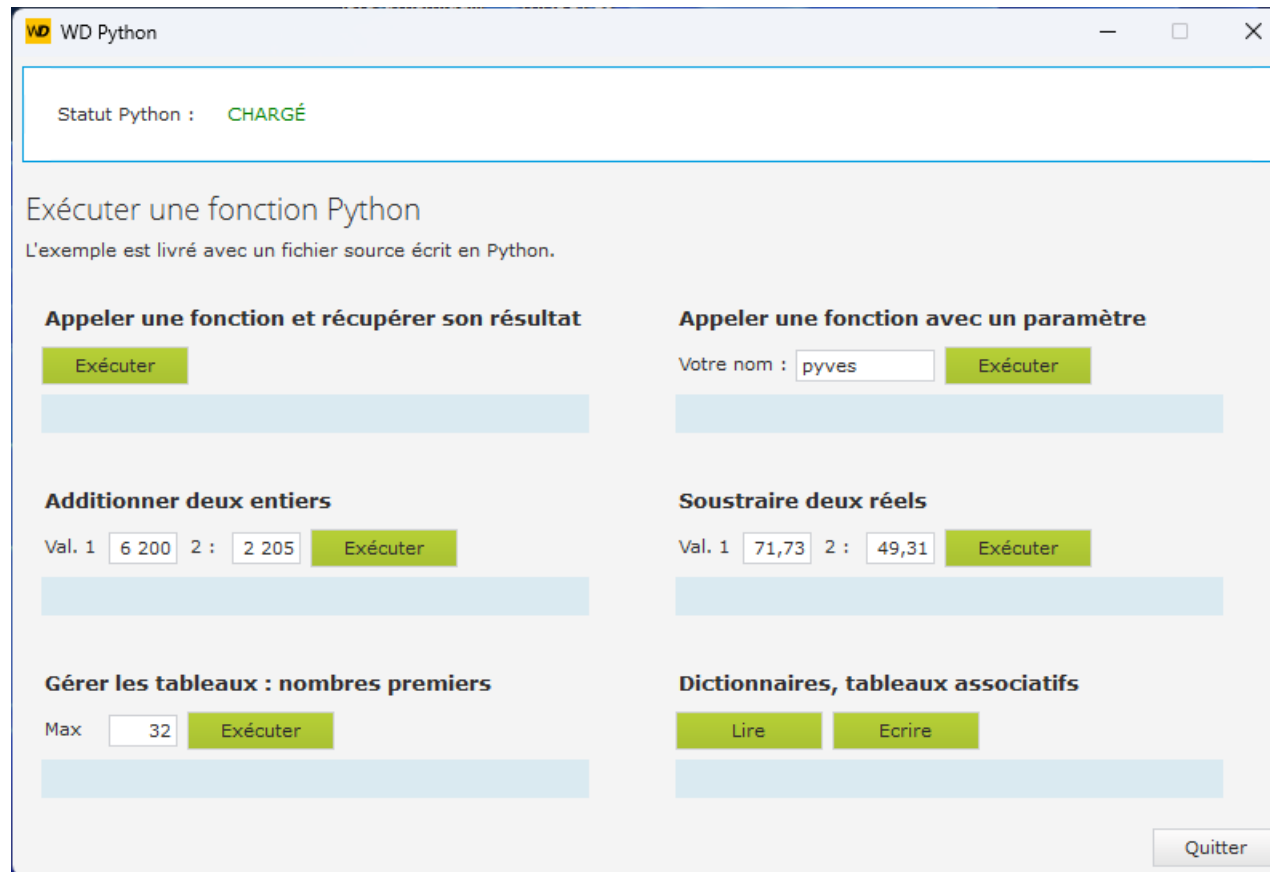


## Test l'exemple WD Python

Ouvrir l'exemple WD Python (testé avec la version 28 01F280051n) et basculez directement sur la configuration "Exécutable Windows 64 bits". En effet il est impératif de synchroniser l'environnement d'exécution de la dll python (**x64**) avec votre exécutable.

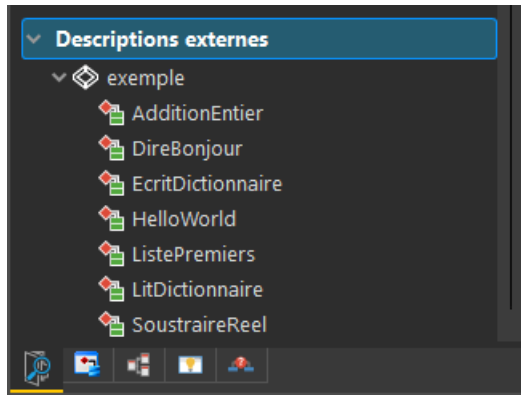


Vous pouvez alors lancer l'application en constatant les améliorations significatives apportées par Python dans Windev :)



## Principe de fonctionnement

Le module Python (fichier .py) doit être déclaré en tant que description de ressource "externe" dans l'explorateur de projet :



à partir de ce moment, vous pouvez appeler les méthodes du module, directement dans le code.

Par exemple, pour appeler la méthode "**HelloWorld**" de exemple.py :

```
// Appelle la fonction 'HelloWorld' du fichier exemple.py  
LIB_RESULTAT_HELLOWORLD = HelloWorld() // Equivalent à PythonExécute( "exemple", "HelloWorld" )
```

En théorie, si vous modifiez le code Python du module, vous devez simplement lancer une mise à jour des descriptions via le menu contextuel accessible via click droit sur l'item "Descriptions externes".

Un petit bug en version 28 51n empêche le rechargement, en réalité, **il faut fermer le projet et le ré-ouvrir**. Si la modification apporté dans le module Python n'apparaît pas, recompilez.

Mais attention, avant de pouvoir appeler ces méthodes, il est impératif que l'interpréteur Python soit initialisé avec la commande "**PythonInitialise**" qui accepte en paramètre le fichier "python3.dll"

Cela est réalisé dans l'exemple dans le code d'initialisation de la fenêtre.

Utilisez la fonction "**PythonTermine**" pour libérer les ressources associées.

## Méthode dynamique

Sachez que vous n'êtes pas obligé de déclarer le module Python en tant que ressource "externe". Grâce à la fonction "PythonExécute", vous pouvez spécifier le nom du module, la méthode à appeler et ses paramètres.

Par exemple:

```
PythonExécute("exemple2.py", "HelloWorld2")
```

Vous noterez la présence de l'extension du fichier (.py)

Attention, dans ce cas, vous n'avez pas l'assistance du compilateur Windev pour vous informer que la méthode n'existe pas ou attend des paramètres...

## Au delà...

Pour utiliser des modules Python qui ont des dépendances comme les bibliothèques pandas ou numpy, vous devez impérativement spécifier l'emplacement de ces dépendances (dossier nommée "site-packages" dans l'environnement virtuel "venv"). Pour cela utilisez la fonction **PythonAjouteChemin** :

```
//Ajout des dépendances des scripts python  
PythonAjouteChemin("C:\\Users\\pyves\\PycharmProjects\\test_regex\\venv\\Lib\\site-packages")
```

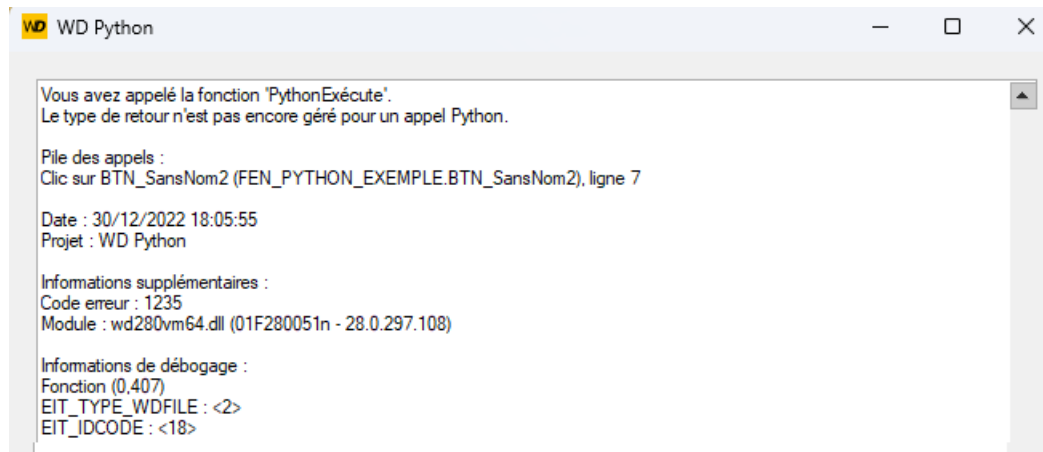


Malheureusement, le type de retour **DataFrame** de la bibliothèque **pandas** n'est pas supporté par Windev...

Exemple de script Python:

```
import pandas as pd  
  
def pandas_test():  
    data = {'col_1': [3, 2, 1, 0], 'col_2': ['a', 'b', 'c', 'd']}  
    df = pd.DataFrame.from_dict(data)  
    return df
```

Appelé avec : `PythonExécute("test_windev_usage.py", "pandas_test")`, provoque une **exception**:



Une solution est de transformer le DataFrame en chaîne afin de le renvoyer à Windev:

```
import pandas as pd

def pandas_test() -> str:
    data = {'col_1': [3, 2, 1, 0], 'col_2': ['a', 'b', 'c', 'd']}
    df = pd.DataFrame.from_dict(data)
    return df.__str__()
```

Une autre solution serait de créer en Python, un fichier Excel à partir du DataFrame avec la méthode "to\_excel".

## Synthèse

Utilisez **PythonInitialise** pour indiquer l'emplacement de votre interpréteur Python.

Utilisez **PythonAjouteChemin** pour indiquer l'emplacement des dépendances de vos modules Python (dossier "site-packages")

Utilisez **PythonAjouteChemin** pour indiquer l'emplacement de vos modules Python ou faites un drag and drop du fichier dans les "Descriptions externes" du projet.

Utilisez **"PythonTermine"** pour libérer les ressources associées.

A voir si vous ne craignez pas les serpents : [Intégrer du code Python dans une application WINDEV - YouTube](https://www.youtube.com/watch?v=...)

Information importante pour ceux qui veulent tester et afin de ne pas complexifier cet article, sachez qu'il est nécessaire d'utiliser l'interpréteur python en version 3.9 avec Numpy (le 30/12/2022).

Aussi, dans le code d'initialisation de la fenêtre, dans la procédure lambda que initialise l'interpréteur, ne prenez que le chemin qui contient '39' avec **ChaîneOccurence**:

:

```
{
    sEmplacement est une chaîne = sChemin + [ fSep ] + sRépertoire + [ fSep ] + "python3.dll"

    // Cherche si le fichier python3.dll existe
    SI fFichierExiste( sEmplacement ) AND ChaîneOccurrence(sEmplacement,"39")>0 ALORS

        // Charge la DLL
        SI PythonInitialise( sEmplacement ) ALORS

            //Ajout des dépendances des scripts python
            PythonAjouteChemin("C:\Users\pyves\PycharmProjects\test_regex\venv\Lib\site-packages")

            LIB_STATUT_PYTHON          = "CHARGÉ"
            LIB_STATUT_PYTHON..Couleur = VertFoncé
            GR_EXEMPLE..Grisé          = Faux
            RENVOYER Faux

        SINON
            Erreur("Echec de l'initialisation de Python", ErreurInfo())
        FIN
    FIN

    RENVOYER Vrai
}
```

Vous avez noté que c'est ici que je défini l'emplacement des dépendances ('site-packages') !

Pierre-yves / Articles sur Windev / 30 décembre 2022 / Affichages : 987

Précédent

Suivant

© 2020 [www.xwdb.fr](http://www.xwdb.fr)