Pourquoi apprendre à coder

Le codage, qu'est-ce que c'est ? Il s'agit simplement du langage de programmation informatique. Maîtriser cette nouvelle syntaxe permet de créer des actions et de commander aux machines. Un graphiste peut par exemple concevoir un site Internet, s'il sait coder il est également capable de lui donner vie sur la toile.

À quoi sert Python ? 7 utilisations concrètes de Python

Savoir programmer permet ainsi de :

1. Grossir son CV : savoir combiner cette double casquette constitue une véritable plus-value sur le marché de l'emploi. Le code enrichit vos compétences, vous rend plus précieux, plus utile, voire indispensable. Bref, vous sortez du lot.

2. Comprendre son environnement : jeux vidéo, applications, pages web… être initié au code c'est savoir décrypter les rouages de ces objets qui nous gouvernent.

3. Partager : contrairement aux idées reçues, le coding est régi par un esprit altruiste, c'est le fameux « open source ». Vous pouvez copier des codes existants mais aussi offrir vos créations. Bienvenue au programme des bisounours.

4. Aller au bout de vos idées : informé du champ des possibles, vous connaîtrez les contraintes pour mettre en œuvre vos concepts et ainsi vous assurer de leur aboutissement.

5. Simplifier les process : grâce à cette nouvelle compétence, un client peut vous confier un projet sans avoir recours à d'autres prestataires. Vous devenez ainsi l'interlocuteur dédié, c'est un gain de temps et de réactivité, et donc d'argent.

6. Inventer des solutions opérationnelles : savoir coder, permet de concevoir des créations réalistes, vouées à exister.

7. Mieux communiquer : connaître le code c'est aussi maîtriser les bons éléments de langage pour expliquer vos idées à un intégrateur ou un développeur. Et d'optimiser ainsi la gestion de votre projet.

8. Booster sa créativité : l'univers du code n'est pas uniquement technique, il offre de nombreuses opportunités créatives. Il s'agit de concevoir des programmes, mais aussi d'imaginer comment leur donner vie.

9. Se la raconter : le codage a le vent en poupe, le succès de l'Ecole 42 en témoigne (Lire aussi L'Ecole 42, élue meilleure Code Factory). Vous commandez aux machines, vous êtes dans la matrice, c'est jubilatoire.

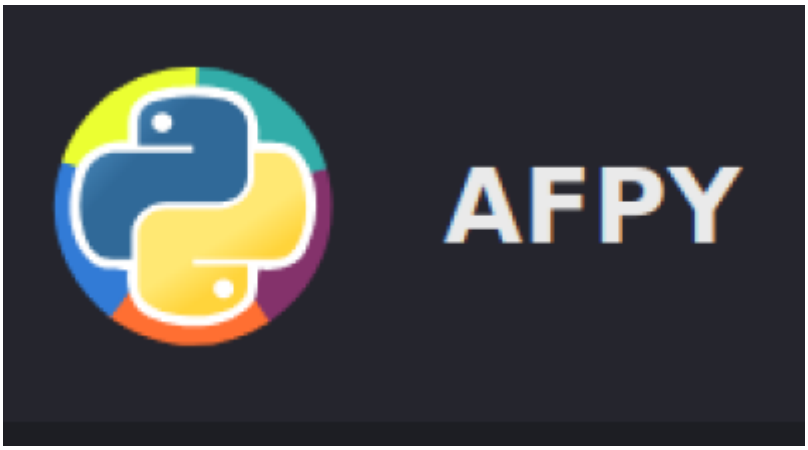
10. S'amuser : l'apprentissage du code consiste à comprendre une nouvelle langue, son alphabet, sa grammaire, ses modes d'expression… afin de créer des actions. Même les plus jeunes s'y mettent et peuvent créer leurs propres jeux grâce notamment aux Magic Makers ou aux coding-goûters.

Le Python

Introduction à Python

Python EN

Association Francophone Python



Fonction print en python FR print en python

Apprendre python 3 en ligne

Pour apprendre le python en ligne: Codecademy EN

Apprendre Python en ligne FR

Python pour les jeunes

FAQ Python

Python : Lancer un programme externe

Cours Python débutant FR

Python Facile FR

Comprendre et corriger les erreurs en Python

Python : Comprendre

Apprendre Python 3 en ligne avec Hackinscience FR

Pygame

WikiBooks: Pygame FR

Apprendre Pygame FR

Tutoriel : Interface graphique Pygame pour Python FR

Pygame : librairie Python pour créer des jeux vidéos FR

Tutoriels Pygame FR

Doc Pygame EN

IDE pour python

Geany

pour debuter avec geany

Geany, un éditeur de texte génial

Apprendre Geany

Geany colour scheme editor

Plugins for Geany

Geany est un éditeur de texte2 léger utilisant GTK+ et Scintilla et incluant les fonctions élémentaires d'un environnement de développement intégré. Il est pensé pour avoir peu de dépendances et démarrer rapidement, il est disponible pour plusieurs systèmes d'exploitation tel que Windows, Linux, Mac OS X3, BSD et Solaris. Il supporte, entre autres, les langages C/C++, Java, JavaScript, PHP, HTML, CSS, Python, Perl, Ruby, Pascal et Haskell.

Geany est plus puissant que SciTE tout en gardant la simplicité de celui-ci. Il n'atteint ni ne vise pour autant la sophistication d'Eclipse. Il peut remplacer sous Windows des éditeurs tels que NoteTab ou ConTEXT.

C'est un logiciel libre sous licence GNU GPL1.

Apprendre Python 3 en ligne avec Hackinscience FR

Visual Studio

Debuter avec Visual studio FR

Visual studio et Python 3 Tutoriel EN

Installation visual studio FR

Notepad++

ici Et ici

Thonny

Facile à démarrer. Thonny est livré avec Python 3.10 intégré, donc un seul installateur simple est nécessaire et vous êtes prêt à apprendre la programmation. (Vous pouvez également utiliser une installation Python distincte, si nécessaire.) L'interface utilisateur initiale est dépourvue de toutes les fonctionnalités qui peuvent distraire les débutants.

PyCharm

PyCharm

IDE Python3 en ligne

IDE Python 3 en ligne

Python le langage pour commencer

- "If else " en Python — If else " en Python
- un autre cours
- Une vidéo sur if-else en python
- Les tableaux en Python 28 mn FR *Les tableaux ou listes en Pyton FR
- Les boucles en python FR *Boucle For... While... Range... FR
- Les types « integer » et « long »
- Python3 et la programmation Objets : POO FR
- Apprendre Python 3 en ligne avec Hackinscience FR
- Quiz-Test-Python3 EN

Python et Minecraft

- Python et Minecraft en Anglais

- Python et Minecraft en Français

Doc Tuto - Livre

- Minecraft :Interagir avec les circuits électroniques Ch5
- le langage Python et MicroPython
- Doc python 3.x officielle FR
- pygame-pour-les-zesteurs.pdf FR
- Minecraft et le raspberry

Videos

Comment coder

Pour les parents

Les langages de programmation expliqués

Histoire et évolution des langages de programmation

Comment utiliser python sous windows

python sous windows

Python et les GPIO du raspberry

GPIO

Programmes d'essais en Python 3

Menu 1 en mode texte tres simple Python3

py menu002.py

```
menu_options = {
    1: 'Option 1',
    2: 'Option 2',
    3: 'Option 3',
    4: 'Exit',
}

def print_menu():
```

```
    for key in menu_options.keys():
        print (key, '--', menu_options[key] )

def option1():
     print('Handle option \'Option 1\'')

def option2():
     print('Handle option \'Option 2\'')

def option3():
     print('Handle option \'Option 3\'')

if __name__=='__main__':
    while(True):
        print_menu()
        option = ''
        try:
            option = int(input('Enter your choice: '))
        except:
            print('Wrong input. Please enter a number ...')
        #Check what choice was entered and act accordingly
        if option == 1:
           option1()
        elif option == 2:
            option2()
        elif option == 3:
            option3()
        elif option == 4:
            print('Thanks message before exiting')
            exit()
        else:
            print('Invalid option. Please enter a number between 1 and
4.')
```

Menu en mode texte python3

doc console-menu

py menu001.py

```
# Import the necessary packages
# pip install console-menu
# Modifier par GL le 15/01/2022
##############################

from consolemenu import *
from consolemenu.items import *

# Create the menu
menu = ConsoleMenu("Essais de menu", "sous menu")

# Create some items

# MenuItem is the base class for all items, it doesn't do anything when
selected
menu_item = MenuItem("Menu 1")

# A FunctionItem runs a Python function when selected
function_item = FunctionItem("Menu 2", input, ["Entrer une donnee "])

# A CommandItem runs a console command
command_item = CommandItem("Demarrer une console de commande",  "touch
hello.txt")

# A SelectionMenu constructs a menu from a list of strings
selection_menu = SelectionMenu(["item1", "item2", "item3"])

# A SubmenuItem lets you add a menu (the selection_menu above, for
example)
# as a submenu of another menu
submenu_item = SubmenuItem("Sous Menu item", selection_menu, menu)

# Once we're done creating them, we just add the items to the menu
menu.append_item(menu_item)
menu.append_item(function_item)
menu.append_item(command_item)
menu.append_item(submenu_item)

# Finally, we call show to show the menu and allow the user to interact
menu.show()
```

calculatrice python3 menu texte

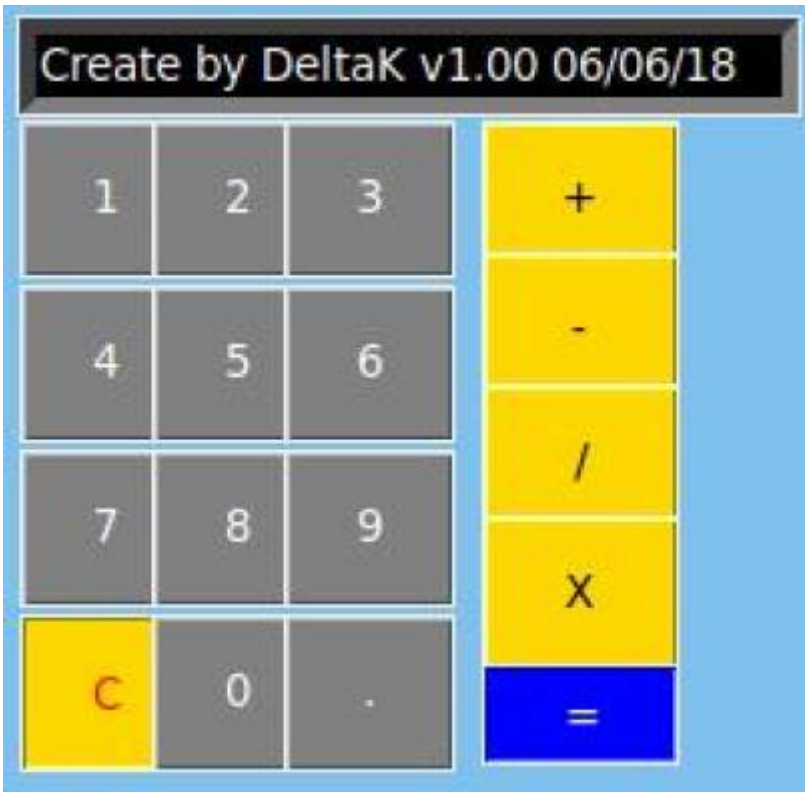
py calc3.py

```
## Programme de calculatrice tres simple pour demarrer en python
```

```
## Modifier par GL 01/2022
#############

def calculate():
   operation = input('''
SVP , Entrez le type d operation desire:
+ pour l addition
- pour la soustraction
* pour la  multiplication
/ pour la  division
''')

   number_1 = int(input('SVP entrer le premier nombre: '))
   number_2 = int(input('SVP entrer le deuxieme nombre: '))

   if operation == '+':
       print('{} + {} = '.format(number_1, number_2))
       print(number_1 + number_2)

   elif operation == '-':
       print('{} - {} = '.format(number_1, number_2))
       print(number_1 - number_2)

   elif operation == '*':
       print('{} * {} = '.format(number_1, number_2))
       print(number_1 * number_2)

   elif operation == '/':
       print('{} / {} = '.format(number_1, number_2))
       print(number_1 / number_2)

   else:
       print('Vous n avez pas chois une operation valide , veuillez
recommencer .')

   # Add again() function to calculate() function


def again():
   calc_again = input('''
Voulez vous calculer a nouveau?
SVP taper O pour Oui , et N pour Non.
''')

   if calc_again.upper() == 'O':
       calculate()
   elif calc_again.upper() == 'N':
       print('Merci, au revoir...')
   else:
       again()
```

```
calculate()
```

Une calculatrice en python3 mode graphique



py calc10.py

```
#////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
//////////////
#
CALCULATRICR (GRAPHIQUE)
#\\\\\\\\\\\\\\\\\\\

#--------------
#IMMPORTATION :
#--------------
from tkinter import * # Tkinter

#---------
# CLASSE :
#---------
class Calculator():
    def __init__(self): # Construction
        self.phase1 = 0 # Premier nombre
        self.phase2 = 0 # Deuxième nombre
        self.final = 0 # Valeur finale
```

```
        self.entry = StringVar() # Capte les valeurs écrit
        self.text = "" # Nombre écrir par l'utilisateur
        self.signe = "" # Type d'opération
        self.entry.set("Create by DeltaK v1.00 06/06/18")

    def init(self): # Initialisation
        self.phase1 = 0 # Premier nombre
        self.phase2 = 0 # Deuxième nombre
        self.final = 0 # Valeur finale
        self.text = "" # Nombre écrir par l'utilisateur
        self.signe = "" # Type d'opération

    def afficher_Nb(self): # Afficher les nombre sur écran
        self.entry.set(self.text)

    def operation(self): # Vérification du type d'opération
        try :
            if "+" in self.text:
                self.Plus()
            elif "-" in self.text:
                self.Sous()
            elif "/" in self.text:
                self.Div()
            elif "X" in self.text:
                self.Mult()
        except:
            self.entry.set("ERROR")
            self.init()

    def Plus(self): # Addition
        nb = self.text.split("+")
        self.phase1 = float(nb[0])
        self.phase2 = float(nb[1])
        self.final = self.phase1 + self.phase2
        self.entry.set(str(self.final))
        self.init()

    def Sous(self): # Soustraction
        nb = self.text.split("-")
        self.phase1 = float(nb[0])
        self.phase2 = float(nb[1])
        self.final = self.phase1 - self.phase2
        self.entry.set(str(self.final))
        self.init()

    def Div(self): # Division
        nb = self.text.split("/")
        self.phase1 = float(nb[0])
        self.phase2 = float(nb[1])
        self.final = self.phase1 / self.phase2
        self.entry.set(str(self.final))
```

```
        self.init()

    def Mult(self): # Multiplication
        nb = self.text.split("X")
        self.phase1 = float(nb[0])
        self.phase2 = float(nb[1])
        self.final = self.phase1 * self.phase2
        self.entry.set(str(self.final))
        self.init()

#------------
# FONCTIONS :
#------------
def Button1 (): # Actionnerle bouton 1
    calculatrice.text += "1"
    calculatrice.entry.set(calculatrice.text)

def Button2 (): # Actionnerle bouton 2
    calculatrice.text += "2"
    calculatrice.entry.set(calculatrice.text)

def Button3 (): # Actionnerle bouton 3
    calculatrice.text += "3"
    calculatrice.entry.set(calculatrice.text)

def Button4 (): # Actionnerle bouton 4
    calculatrice.text += "4"
    calculatrice.entry.set(calculatrice.text)

def Button5 (): # Actionnerle bouton 5
    calculatrice.text += "5"
    calculatrice.entry.set(calculatrice.text)

def Button6 (): # Actionnerle bouton 6
    calculatrice.text += "6"
    calculatrice.entry.set(calculatrice.text)

def Button7 (): # Actionnerle bouton 7
    calculatrice.text += "7"
    calculatrice.entry.set(calculatrice.text)

def Button8 (): # Actionnerle bouton 8
    calculatrice.text += "8"
    calculatrice.entry.set(calculatrice.text)

def Button9 (): # Actionnerle bouton 9
    calculatrice.text += "9"
    calculatrice.entry.set(calculatrice.text)

def Button0 (): # Actionnerle bouton 0
    calculatrice.text += "0"
```

```
    calculatrice.entry.set(calculatrice.text)

def ButtonF(): # Actionnerle bouton F
    calculatrice.text += "."
    calculatrice.entry.set(calculatrice.text)

def ButtonP (): # Actionnerle bouton P
    calculatrice.text += "+"
    calculatrice.entry.set(calculatrice.text)

def ButtonS (): # Actionnerle bouton S
    calculatrice.text += "-"
    calculatrice.entry.set(calculatrice.text)

def ButtonD (): # Actionnerle bouton D
    calculatrice.text += "/"
    calculatrice.entry.set(calculatrice.text)

def ButtonM (): # Actionnerle bouton M
    calculatrice.text += "X"
    calculatrice.entry.set(calculatrice.text)

def ButtonE (): # Actionnerle bouton E
    calculatrice.operation()

def ButtonC (): # Actionnerle bouton c
    calculatrice.entry.set("")
    calculatrice.init()

#----------
# FENETRE :
#----------
fen = Tk() # Création de a fenêtr 1e
fen.geometry("200x240") # Définition de la fenêtre
fen.title("Calculatrice v1.0") # Titre de la calculatrice
fen["bg"]= "SkyBlue2" # Coleur de la fenêtre
fen["relief"] = "raised" # Profondeur de la fenêtre
#------------
# PROGRAMME :
#------------
# Création instance
calculatrice = Calculator()

# ATTRIBUTS DE LA FENETRE
#########################
# // Ecran calculatrice //
ECRAN = Entry(fen, width=28, textvariable=calculatrice.entry, bg
="black", fg="white", relief=SUNKEN, bd=5).place(x=9, y=8)

# // Bouttons //
B1 = Button(fen, text="1", command=Button1, width=3, height=2,
```

```
bg="grey", fg="white").place(x=10, y=40) # Boutton 1
B2 = Button(fen, text="2", command=Button2, width=3, height=2,
bg="grey", fg="white").place(x=50, y=40) # Boutton 2
B3 = Button(fen, text="3", command=Button3, width=3, height=2,
bg="grey", fg="white").place(x=90, y=40) # Boutton 3
B4 = Button(fen, text="4", command=Button4, width=3, height=2,
bg="grey", fg="white").place(x=10, y=90) # Boutton 4
B5 = Button(fen, text="5", command=Button5, width=3, height=2,
bg="grey", fg="white").place(x=50, y=90) # Boutton 5
B6 = Button(fen, text="6", command=Button6, width=3, height=2,
bg="grey", fg="white").place(x=90, y=90) # Boutton 6
B7 = Button(fen, text="7", command=Button7, width=3, height=2,
bg="grey", fg="white").place(x=10, y=140) # Boutton 7
B8 = Button(fen, text="8", command=Button8, width=3, height=2,
bg="grey", fg="white").place(x=50, y=140) # Boutton 8
B9 = Button(fen, text="9", command=Button9, width=3, height=2,
bg="grey", fg="white").place(x=90, y=140) # Boutton 9
BC = Button(fen, text="C", command=ButtonC, width=3, height=2,
bg="gold", fg="red", relief=RIDGE).place(x=10, y=190) # Boutton C
(Clear)
B0 = Button(fen, text="0", command=Button0, width=3, height=2,
bg="grey", fg="white").place(x=50, y=190) # Boutton 0
BF = Button(fen, text=".", command=ButtonF, width=3, height=2,
bg="grey", fg="white").place(x=90, y=190) # Boutton = (égale)

BP = Button(fen, text="+", command=ButtonP, width=4, height=2,
bg="gold", fg="black", relief=GROOVE).place(x=150, y=40) # Boutton +
(addition)
BS = Button(fen, text="-", command=ButtonS, width=4, height=2,
bg="gold", fg="black", relief=GROOVE).place(x=150, y=80) # Boutton -
(soustacrtion)
BD = Button(fen, text="/", command=ButtonD, width=4, height=2,
bg="gold", fg="black", relief=GROOVE).place(x=150, y=120) # Boutton /
(division)
BM = Button(fen, text="X", command=ButtonM, width=4, height=2,
bg="gold", fg="black", relief=GROOVE).place(x=150, y=160) # Boutton X
(multiplication)
BE = Button(fen, text="=", command=ButtonE, width=4, height=1,
bg="blue", fg="white", relief=RIDGE).place(x=150, y=205) # Button =
(égale)

fen.mainloop() # Gestion de la fenêtre
```

lire un fichier texte

py testfich001.py

```
#! /usr/bin/env python
```

```
# coding: utf-8

menu_options = {
    1: 'Option 1',
    2: 'Option 2',
    3: 'Aide -- 3',
    4: 'Exit',
}

def print_menu():
    for key in menu_options.keys():
        print (key, '--', menu_options[key] )

def option1():
     print('Handle option \'Option 1\'')

def option2():
     print('Handle option \'Option 2\'')

def option3():
    f = open('aide001.txt', 'r')
    data = f.read()
    f.close
    print(data)

if __name__=='__main__':
    while(True):
        print_menu()
        option = ''
        try:
            option = int(input('Enter your choice: '))
        except:
            print('Wrong input. Please enter a number ...')
        #Check what choice was entered and act accordingly
        if option == 1:
           option1()
        elif option == 2:
            option2()
        elif option == 3:
            option3()
        elif option == 4:
            print('Thanks message before exiting')
            exit()
        else:
            print('Invalid option. Please enter a number between 1 and
4.')
```

Inserer le fichier texte “aide001.txt” dans le meme repertoire que le programme ci-dessus

aide001.txt

```
Ceci est un fichier d aide à completer suivant les besoins :

une introduction  :

Python (prononcé /pi.tɔ̃/) est un langage de programmation interprété,
multi-paradigme et multiplateformes.

Il favorise la programmation impérative structurée, fonctionnelle et
orientée objet.

Il est doté d'un typage dynamique fort, d'une gestion automatique de la
mémoire par ramasse-miettes et d'un système de gestion d'exceptions ;

il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl.

Le langage Python est placé sous une licence libre proche de la licence
BSD3 et fonctionne sur la plupart des plates-formes informatiques, des
smartphones aux ordinateurs centraux4,
de Windows à Unix avec notamment GNU/Linux en passant par macOS,  ou
encore Android, iOS, et peut aussi être traduit en Java ou .NET. Il est
conçu pour optimiser la productivité
des programmeurs en offrant des outils de haut niveau et une syntaxe
simple à utiliser.

Il est également apprécié par certains pédagogues qui y trouvent un
langage où la syntaxe, clairement séparée des mécanismes de bas niveau,
permet une initiation aisée aux concepts de base de la programmation5.
```

MicroPython

les bases de microPython FR

Apprendre Micropython en ligne avec Chrome FR

Lego MindStormsEducation et Micropython EN

LEGO® Education SPIKE TM Principal MicroPython EN

PDF : LEGO® Education SPIKE TM Principal MicroPython FR

micropython pour ESP32 FR

Doc FR en PDF MicroPython et ESP32

Micropython FR

MicroPython sur ESP8266 ou ESP32

Test Micropython avec un esp32 en Ligne sur Wokwi

CPython

CPython Guide EN

Python version CPython EN

CPython sur GitHub EN

CPython Guide 02 EN

Calculatrices utilisant Python

Numworks

TI83

Casio