# Walking directories in Python in search of files.

Python version: 2.7.5

Tkinter version: 8.5.2

Source: [file\_walker.py](https://doc-0c-3k-docs.googleusercontent.com/docs/securesc/cormtv1php2grf0ospqqeu63j685nr1q/hvisfphctu9b50j3o4455qah845ek0bp/1372140000000/14340645910421099495/14340645910421099495/0B1qmap1Ur9MvZ2RRRWNzZUh1VEU?e=download&h=16653014193614665626&nonce=vbpt3602dg0ec&user=14340645910421099495&hash=f37rrm9sigj2rfirb4kt88pkuqc9nhrb)

Brief:

This tutorial shows how to walk directory trees using python.  A simple Tkinter interface is used to and directory paths are displayed with files are found within that directory.  These directory paths could then be copied and pasted into the location bar of explorer to find these files.

**1. Walk directory trees in search of files**

Python provides a simple interface for dealing with file systems within the [os module](http://docs.python.org/2/library/os.html).  Of note is the os.getcwd() and os.walk().  getcwd returns the current working directory.  If you are using IDLE this is probably your python root directory.  os.walk() returns a [turple](http://docs.python.org/2/tutorial/datastructures.html#tuples-and-sequences) that can be gone through using a for loop.  The code below loops through the current working directory and prints out root directory paths that contain files.

import os  
  
for root, dirs, files in os.walk(os.getcwd()):  
    if files:  
        print root

**2. Build a simple Tkinter interface**

Tkinter is the included GUI library.  The documentation included with Tkinter in the Python documentation is real lacking.  I've found one of the best references on Tkinter, although outdated, to be the book [Python and Tkinter Programming](http://www.amazon.com/Python-Tkinter-Programming-Grayson-Ph-D/dp/1884777813/ref=sr_1_2?ie=UTF8&qid=1372138957&sr=8-2&keywords=Python+Tkinter) by John E Grayson Ph.D.  There is also a tutorial online at <http://www.tkdocs.com/tutorial/> if you are using Python 3.x.  Most of what I've learned has been through the above book, reading the source code in the python library, trial and error, and searching the internet.  Tkinter is relatively easy to use and full featured, there just isn't one good source of documentation on it.

Of note is the tkFileDialog module that provides access to the standard open file / directory dialog boxes within your operating system.

version = '0.1'  
  
import os  
from Tkinter import BOTH,LEFT,TOP,END,StringVar  
from ttk import Frame, Entry, Button, Label  
from ScrolledText import ScrolledText  
from tkFileDialog import askdirectory  
  
class FileWalkerWindow(Frame):  
    def \_\_init\_\_(self):  
        Frame.\_\_init\_\_(self)  
        self.pack(expand=True, fill=BOTH)  
        self.master.title("File Walker v" + version)  
        self.master.iconname("File Walker")  
  
        self.dir = StringVar() #tkinter does not work with standard python variables  
        self.dir.set(os.getcwd()) # set to current working directory  
          
        description = "This program walks directories looking " \  
        + "for files.  A directory path is outputed each time a file is " \  
        + "found.  These directory paths can be coppied into explorer to " \  
        + "view files."  
          
        row1 = Frame(self)  
        Label(row1, text="Root Directory:").pack(side=LEFT, pady=10)  
        self.dir\_ent = Entry(row1, width=80, textvariable=self.dir)  
        self.dir\_ent.pack(side=LEFT)  
        Button(row1, text="Browse", width=10, command=self.browse).pack(side=LEFT, padx=5)  
        row1.pack(side=TOP, ipadx=15)  
  
        row2 = Frame(self)  
        btn = Button(row2, text="Find Files", command=self.walk, width=15)  
        btn.pack(side=LEFT, padx=5, pady=10)  
        row2.pack(side=TOP)  
  
        self.output = ScrolledText(self, height=15, state="normal",  
                                   padx=10, pady=10,  
                                   wrap='word')  
        self.output.insert(END,description)  
        self.output.pack(side=LEFT, fill=BOTH, expand=True, padx=5, pady=5)  
  
  
        self.bind('<Key-Return>', self.walk) #bind enter press to walk  
  
    def browse(self):  
        dirpath = askdirectory(parent=self, title="Select Root Directory")  
        self.dir.set(dirpath)  
  
    def walk(self):  
        self.output.delete(1.0, END)  
        for root, dirs, files in os.walk(self.dir.get()):  
            if files:  
                self.output.insert(END, root+'\n')  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
    FileWalkerWindow().mainloop()