

Re: ultra newbie question (don't laugh)

Source: <http://coding.derkeiler.com/Archive/Python/comp.lang.python/2006-09/msg04107.html>

- *From:* Larry Bates <larry.bates@xxxxxxxxxxxx>
 - *Date:* Tue, 26 Sep 2006 09:38:03 -0500
-

John Salerno wrote:

Ok, I've decided to make a little project for myself which involves storing employee information in an XML file. I'm doing this partly to experiment with working with XML. The blocks in the file will look something like this:

```
<researcher id="salerjo01">
<first_name>John</first_name>
<last_name>Salerno</last_name>
<birth_country>United States</birth_country>
<birth_state>Texas</birth_state>
<birth_city>Houston</birth_city>
#etc.....
</researcher>
```

I also plan to make a GUI frontend with wxPython for entering the records. This will be fairly easy, but not as fun as writing the logic. For now I've decided to focus just on writing the logic, and not worry about the GUI yet.

So this is what I came up with so far, then I sat staring at the screen wondering how to proceed:

```
class LabXMLWriter(object):

def write_name(self, first, last, given=""):
```

Suddenly I realized I just have no idea how to start thinking about what I need to do. My first instinct was to use a class, as above, but then I wondered if that was even necessary, since all I need to do is get information from a user and write it to a file. Do I really need that information stored in an object?

Then I wondered if I needed an `__init__` method, and what could go in it? Or should I just make separate methods for each bit of information to write to the file (i.e., name, birth location, address, phone number, etc.). I thought maybe I could create the ID in the `__init__` method (salerjo01), but to do that I need the name first. I could do this:

Re: ultra newbie question (don't laugh)

```
def __init__(self, first_name, last_name, given_name=""):
# code to initialize name and create ID
```

But then I wondered if this detracts from the work that the class methods would do later.

So you see, what I'm asking for is very basic help, sort of along the lines of "what things do I need to consider before I even begin this?" Is OOP necessary here? Would utility functions work just as well for simply writing the information to a file?

Perhaps I should just take some kind of programming intro class! I read all these Python books, but when it comes time to write something non-trivial, I get stuck almost immediately with all the possibilities.

Thanks,
John

One reason to use OOP is to insulate your main code from things like storage backends or future changes you may decide to make. What if you change your mind and want to store the data into something other than XML? If you move all your storage code into an object and have it do the writing/reading then you can easily replace that object without disrupting your main program. You could even extend your program and allow someone to store in CSV, database, or other storage by adding additional storage objects. I had a project where I was reading data from a CSV file. Later in the implementation the client decided that the file that was to be processed would be a .ZIP file instead of a plain-text CSV. Because I had implemented a class to do the reading of all data from the CSV file that had implemented an iterator, I was able to insert code to open .ZIP file (using zipfile module), open a file inside the .ZIP and read using CSV module from the zipped file without ever actually unzipping the file. I didn't change any top level code and the change took a just a few minutes to implement. There was little chance of creating unintended problems and I was able to easily support both the old unzipped CSV files AND the new zipped files. If I had not isolated that code into a class, the changes would have been a LOT more difficult.

It can also be nice to provide for future expansion by supporting additional keyword arguments. Something like:

```
def __init__(self, first_name, last_name, **kwargs):
```

Then process the kwargs for additional fields. This way you can painlessly add a field to your code by just specifying an additional keyword argument. If the application is as simple as you describe it may not be worth the additional effort. Once you have done one of these it will become second nature to think of creating/using objects in places where you want to provide for maximum flexibility and code isolation.

At a minimum you should take a look at the elementtree module for handling your

Re: ultra newbie question (don't laugh)

Re: ultra newbie question (don't laugh)

XML. It is downloadable for Python < 2.5 and is part of the standard library starting with version 2.5.

Hope information helps at least a little.

–Larry Bates

.