MAIN
PLAY VIDEO
PREVIOUS
NEXT
HELP
Follow

Exercise 41: Learning To Speak Object Oriented

In this exercise I'm going to teach you how to speak "object oriented." What I'll do is give you a small set of words with definitions you need to know. Then I'll give you a set of sentences with holes in them that you'll have to understand. Finally, I'm going to give you a large set of exercises that you have to complete to make these sentences solid in your vocabulary.

Word Drills

class	Tell Python to make a new type of thing.
object	Two meanings: the most basic type of thing, and any instance of some thing.
instance	What you get when you tell Python to create a class.
def	How you define a function inside a class.
self	Inside the functions in a class, self is a variable for the instance/object being accessed.
inheritance	The concept that one class can inherit traits from another class, much like you and your parents.
composition	The concept that a class can be composed of other classes as parts, much like how a car has wheels.
attribute	A property classes have that are from composition and are usually variables.
is-a	A phrase to say that something inherits from another, as in a "salmon" is-a "fish."
has-a	A phrase to say that something is composed of other things or has a trait, as in "a salmon has-a mouth."

Take some time to make flash cards for these terms and memorize them. As usual this won't make too much sense until after you are finished with this exercise, but you need to know the base words first.

Phrase Drills

Next I have a list of Python code snippets on the left, and the English sentences for them:

class X(Y)	"Make a class named X that is-a Y."

class X(object): def __init__(self, J)
    "class X has-a __init__ that takes self and J parameters."
class X(object): def M(self, J)
    "class X has-a function named M that takes self and J parameters."
foo = X()
    "Set foo to an instance of class X."
foo.M(J)
    "From foo get the M function, and call it with parameters self, J."
foo.K = Q
    "From foo get the K attribute and set it to Q."

In each of these where you see X, Y, M, J, K, Q, and foo you can treat those like blank spots. For example, I can also write these sentences as follows:

> 1. "Make a class named ??? that is-a Y."
> 2. "class ??? has-a __init__ that takes self and ??? parameters."
> 3. "class ??? has-a function named ??? that takes self and ??? parameters."
> 4. "Set foo to an instance of class ???."
> 5. "From foo get the ??? function, and call it with self=??? and parameters ???."
> 6. "From foo get the ??? attribute and set it to ???."

Again, write these on some flash cards and drill them. Put the Python code snippet on the front and the sentence on the back. You *have* to be able to say the sentence exactly the same every time whenever you see that form. Not sort of the same, but exactly the same.

Combined Drills

The final preparation for you is to combine the words drills with the phrase drills. What I want you to do for this drill is this:

> 1. Take a phrase card and drill it.
> 2. Flip it over and read the sentence, and for each word in the sentence that is in your words drills, get that card.
> 3. Drill those words for that sentence.
> 4. Keep going until you are bored, then take a break and do it again.

A Reading Test

I now have a little Python hack script that will drill you on these words you know in an infinite manner. This is a simple script you should be able to figure out, and the only thing it does is use a library called `urllib` to download a list of words I have. Here's the script, which you should enter into `oop_test.py` to work with it:

```python
import random
from urllib import urlopen
import sys

WORD_URL = "http://learncodethehardway.org/words.txt"
WORDS = []

PHRASES = {
    "class %%%(%%%):":
      "Make a class named %%% that is-a %%%.",
    "class %%%(object):\n\tdef __init__(self, ***)" :
      "class %%% has-a __init__ that takes self and *** parameters.",
    "class %%%(object):\n\tdef ***(self, @@@)":
      "class %%% has-a function named *** that takes self and @@@ parameters.",
    "*** = %%%()":
      "Set *** to an instance of class %%%.",
    "***.***(@@@)":
      "From *** get the *** function, and call it with parameters self, @@@.",
    "***.*** = '***'":
      "From *** get the *** attribute and set it to '***'."
}

# do they want to drill phrases first
if len(sys.argv) == 2 and sys.argv[1] == "english":
    PHRASE_FIRST = True
else:
    PHRASE_FIRST = False

# load up the words from the website
for word in urlopen(WORD_URL).readlines():
    WORDS.append(word.strip())


def convert(snippet, phrase):
    class_names = [w.capitalize() for w in
                   random.sample(WORDS, snippet.count("%%%"))]
    other_names = random.sample(WORDS, snippet.count("***"))
    results = []
    param_names = []

    for i in range(0, snippet.count("@@@")):
        param_count = random.randint(1,3)
        param_names.append(', '.join(random.sample(WORDS, param_count)))

    for sentence in snippet, phrase:
        result = sentence[:]

        # fake class names
        for word in class_names:
            result = result.replace("%%%", word, 1)

        # fake other names
        for word in other_names:
            result = result.replace("***", word, 1)

        # fake parameter lists
        for word in param_names:
            result = result.replace("@@@", word, 1)

        results.append(result)

    return results


# keep going until they hit CTRL-D
try:
    while True:
        snippets = PHRASES.keys()
        random.shuffle(snippets)

        for snippet in snippets:
            phrase = PHRASES[snippet]
            question, answer = convert(snippet, phrase)
            if PHRASE_FIRST:
                question, answer = answer, question

            print question

            raw_input("> ")
            print "ANSWER:  %s\n\n" % answer
```

```
except EOFError:
    print "\nBye"
```

Run this script and try to translate the "object-oriented phrases" into English translations. You should see that the `PHRASES` dict has both forms and you just have to enter the correct one.

Practice English to Code

Next you should run the script with the "english" option so that you drill the inverse operation:

```
$ python oop_test.py english
```

Remember that these phrases are using nonsense words. Part of learning to read code well is to stop placing so much meaning on the names used for variables and classes. Too often people will read a word like "Cork" and suddenly get derailed because that word will confuse them about the meaning. In the above example, "Cork" is just an arbitrary name chosen for a class. Don't place any other meaning on it, and instead treat it like the patterns I've given you.

Reading More Code

You are now to go on a new quest to read even more code, to read the phrases you just learned in the code you read. You will look for all the files with classes, and then do the following:

> 1. For each class give its name and what other classes it inherits from.
> 2. Under that, list every function it has and the parameters they take.
> 3. List all of the attributes it uses on its self.
> 4. For each attribute, give the class this attribute is.

The goal is to go through real code and start learning to "pattern match" the phrases you just learned against how they're used. If you drill this enough you should start to see these patterns shout at you in the code whereas before they just seemed like vague blank spots you didn't know.

Common Student Questions

Q: What does `result = sentence[:]` do?

> That's a Python way of copying a list. You're using the list slice syntax `[:]` to effectively make a slice from the very first element to the very last one.

Q: This script is hard to get running!

> By this point you should be able to type this in and get it working. It does have a few little tricks here and there but there's nothing complex about it. Just do all the things you've learned so far to debug scripts. Type each line in, confirm that it's *exactly* like mine, and research anything you don't know online.

Q: It's still too hard!

> You can do this. Take it very slow, character by character if you have to, but type it in exactly and figure out what it does.

Video