Unit 9

Apply what you know

0. Study the programs in the *Learn something new* section until you can write them yourself from scratch without relying on this document or any other source of information. Here are the programs:

    0.1. Write a class for playing cards. Include methods to return the face value and suit of a card. Arrange for Python to print cards in a sensible format. Write a program to test the class.

    0.2. Write a class for decks of playing cards. A new deck should be initialized to contain a full set of cards in random order. Include methods to shuffle the deck, deal one card and return the number of cards that have not yet been dealt. Test the class with a program that deals from two decks, side by side and notes coincidences in which the same card is dealt from both decks.

    0.3. Write a specialized version of the deck class that includes an attribute for the name of the owner of the deck. When an instance of this kind of deck is printed, the owner's name should be part of the output. Test the new class, making sure that inherited methods work, as well as the new facility for printing.

    0.4. Define a new widget for 'Quit' buttons by specializing the `Button` class provided by `tkinter`. Test it by writing a minimal GUI program with a 'Quit' button.

    0.5. Define a new widget that combines an entry box, a label that tells the user what to type in it and a button to click when the typed input is ready for processing. This enhanced entry box should include methods to change the prompt, button text and button command. It should also provide easy access to the user input. Test it by writing a GUI program that accepts a text string from the user and then 'slides' it into a label, so that it appears to move smoothly into view from the left-hand side of the label.

1. Define a new class for bank accounts. Include methods to deposit money in the account, to withdraw money from the account and to display a statement of all transactions. Printing a bank account should display the owner's name and the current balance. Write a test program to demonstrate the use of all features of the class.

    In producing the bank statement you may want to right-justify strings. Just use the > character in a pattern string. If `word` is `'hello'`, then `'{0:8s}'.format(word)` is `'hello   '`, but `'{0:>8s}'.format(word)` is `'   hello'`.

2. Define a new sliding label widget based on the code for Program 9.0.5. Include a method that sets the text property of the label by sliding it into place from the left-hand side. Add this widget to `myWidgets.py` and use it to rewrite Program 9.0.5.

3. Define a "reverse" sliding label widget. It works just like the sliding widget in Program 9.2, but text slides in from the right-hand side of the label. Add the new widget to `myWidgets.py` and write a simple GUI program to demonstrate it in action.

   It may help you to know that `*` is defined to mean *repetition* for strings, so that `5*'ab'` is `'abababababab'`.

4. Add the enhanced entry widget of Program 9.0.5 to `myWidgets.py` and use it to rewrite the word-scramble-puzzle solver of Program 8.1.

5. Use the enhanced entry widget of Program 9.0.5 to rewrite the vocabulary learner of Program 8.2. Be sure that code for the enhanced entry widget is included in `myWidgets.py` before you start.

6. Define a new widget that initially displays a single entry box, along with two buttons. One button can be set to display any text and carry out any function, as in the enhanced entry widget.

   The other button is labeled 'More'. Clicking it causes an additional entry box to appear. By clicking it repeatedly, the user can produce as many entry boxes as desired and, thus, enter as many strings as he or she wants. Include a method that returns a list of all strings the user has entered.

   Add the new widget to `myWidgets.py` and test it with a program that asks for names and displays the first one with five or more letters, identifying it as the 'favorite'. If none has five or more letters, the first name is selected as the favorite. Challenge a friend to run the program and try to figure out which names are favored.

   Here is how the program might look before the user enters any names, after the 'More' button has been clicked four times and five names have been entered and after the 'Pick favorite' button has been clicked:


tk
Enter names:
more
Pick favorite
tk
Enter names:
Fred
Sue
Anthony
Ralph
Elizabeth
more
Pick favorite
tk
Enter names:
Fred
Sue
Anthony
Ralph
Elizabeth
more
Pick favorite
My favorite name is Anthony