Computers & the Humanities 281

Flash Card Application: Vocabulary Drill

These notes are designed to walk you through the process of creating a simple application that emulates a set of flash cards. Remember the purpose of flash cards: An item is presented, the user attempts to recall the associated word or phrase, and then turns the card over to check whether he/she was correct or not. We can emulate this behavior with a Revoultion stack. For the purposes of an example, we will do a vocabulary drill. Think carefully about what you would like to accomplish and that will determine how you create your stack.

Beginnings

In order to have an effective vocabulary drill, we need vocabulary words. We could start with two parallel fields of words to drill, one with English words, and one in the target language (French, for example. In order to work properly, the drill items must correspond line for line (e.g., the first line of one must correspond to the first line of the other, and so forth). We will need a field in which to display the vocabulary word being used as a prompt and another field for the corresponding word in the target language.

To begin with, it would be nice to have the ability to switch the drilling between the two languages in which we are working, to reinforce the vocabulary. This could be accomplished through a dialog box, allowing the student to pick have an English word given and then require the French equivalent to be given, or vice versa. We need to give some visual representation of their choice, perhaps by placing it directly above where the prompt will be presented.

We then need to create a button that will choose a prompt and its corresponding vocabulary word from the lists we have created and place them in the appropriate locations. We could move lockstep through the lists, but the drill would be enhanced (and the learning more complete) if the words were randomized. We can easily generate a random number between 1 and the number of words in the list, then use this random number to get the item located on that particular line from each list, and put each word into its corresponding field for display for the user.

This last function becomes a little tricky as we must determine what goes where based upon the decision of the user. One approach to this (among the hundreds possible) would be to create some custom properties of the card that correspond to the prompt language chosen and the response language. By employing custom properties, it becomes easy to determine which language holds which position in the drill (this could accomplished just as easily through global variables). We shouldn't forget another button that will initialize the card, some sort of start button.

There Must Be More...

Even though this works after a fashion, it requires too many clicks to make it all work. First let's combine the click to start with the click to pick new words. We could add commands to the script of the start button to automatically pick new words. This would use the same process as pick button does: determine how many lines, pick random number, copy that line from lists into flash card fields. Actually, we could just copy and paste code.

Wait a second. Does it bother you to have the same code in two places? What if we find a bug, or wanted to augment/amend the process? We would have to remember to fix it in both places. **Red Flag:** Identical code in more than one place is bad form. It is smarter to put the code in one place and invoke it from different places (much like a subroutine). To that end, let's make a Pick message handler on the card script. This would obviate the need for the Pick button and solve our dilemma with too many clicks. Remember that we can create our own messages and handlers for those messages. Hence, we can create an `on pick` handler. We could then invoke that handler from the Start button. Such maneuvers eliminate a number of clicks and tighten up the operation of the application.

Let's Try This...

Now although our flash card application works fairly well, another problem has emerged: We get a lot of duplicates. In fact, we often get the same item twice (or more) in a row, which would never really happen with a true set of flash cards. This is partially due to the small list we have, but primarily a result of how we randomly pick words. If we had a larger list, the possibility exists of never getting to some words at all. We evidently need a technique to guarantee that the words are still presented randomly, but once and only once, ensuring that all were presented.

It is easy to see the solution to this if we examine the metaphor with which we started. In a true deck of flash cards, once we have reviewed a card, we remove that card. That ensures that we don't encounter it again before the others are reviewed. It also guarantees that all cards are used because the deck gets smaller.

Let's implement this for our application. First we'll modify the Pick handler to delete lines from the lists after copying them. Since our random statement uses the number of lines in the field, it will continue to work as the number of lines is reduced (the range of numbers gets smaller). We also need to add an if-statement to check when we run out of words.

However, this has created another problem. Our application is not like a deck of cards in that we can't collect the used cards and reshuffle them to recreate the stack. The words are gone because we have deleted them and we'll never see them again. To solve this we need a method to preserve both lists of words while still maintaining the functionality we just created. One way to do this would be to have master word lists, then copy these lists to scratch fields at the beginning of each session with the flash cards. We just need to create duplicate fields, rename the originals, and change the Start button script. It would also be nice to improve the capability of changing the language to practice. One way to accomplish this would be to create a tabbed button. This button would need two tabs: one tab named "English" and the other "French." Then send this button to the back behind the other items on the card. We need to script the tabbed button to start the process. Then, we need to get the text of the selected tab to determine which the prompt language and which is the response language. If done properly, the result is that the English tab would allow us to review English words, and the French tab would allow us review French words. This gives users the ability to review words in two ways, much like traditional flash cards.

We also want to stop showing the response word. With traditional flash cards, we usually present a word and give ourselves the opportunity to respond with its equivalent in the target langauge. There are two approaches to this. We can either hide the word and then show it at the bequest of the user, giving the user the opportunity to determine if he/she remembered the word or not (much like traditional flash cards), or we could provide a field in which the

user could type what he/she thinks the correct response would be and have the computer check the answer (this latter approach is more of a quiz than a drill). A combination of these two is possible as well. You just need to be aware of your target audience and tailor the flash card activity to their needs.

And How About This...

When we're drilling vocabulary with real flash cards, we often take the cards with words we missed and replace them back in the deck (instead of laying them aside) in order to review them until we feel that we know them. This is a feature we would want to emulate in our application.

Your approach to this end would depend heavily upon how you decided to allow the user to determine if they responding correctly or to let the computer do it for them. If they are governing themselves, then you need to add a Right and a Wrong button so the users can indicate the ones they know and don't know. The Right button should just pick another word (we're not keeping stats or anything at this point). The Wrong button should put words back into work lists so they will come up again. If the computer is checking the answers, we would want it to do the same thing: move on with correct response, and retain words that were missed.

Since the words are already in containers (the fields displaying the words), we can use those to place the words back into the lists. We just have to make sure that when we put the words back, we maintain the same list organization, i.e., one word/phrase per line, with related words on corresponding lines. To guarantee there's only one word/phrase per line, we'll include a `return` with each word (remember that a `return` delimits lines). To ensure they're on corresponding lines, we'll put them `before` the other words in the fields, which will place them on the first line of each field. We'll then have to pick new words for review. This would effectively put the missed words back into the mix where they would come up again for review.

Administrative Details

Like any deck of flash cards, there will come a time when it will become desired to augment/alter the words being drilled. If this is done on a regular basis, it becomes tedious to have to manually show the fields. We would be better served instead by creating a button to show them for editing. This is a perfect situation for a check box button. The script would check the hilite of the button and then set the visible of the fields to match the hilite of the button. Remember that a check box button has different hiliting behavior than regular buttons. When the box is checked, the hilite is true; when the box is not checked, the hilite is false. Setting the visible of the master word list fields to match the hilite of the button will then alternately show the lists and hide them. Since the Start button hides the list fields it should also de-hilite this Edit Vocabulary button to avoid confusion.

Finishing Touches

One finishing touch would be to add scripting to empty the flash card fields when the list is exausted and indicate such to the user (probably in the Pick handler). This would make the final product clearer and more visually interesting. Also, we can also add some visual enhancements by changing some field properties to emphasize which language is the object of review.

And Now, For an Encore...

Your assignment, then, is to create a flash card activity in a language of your choice.

1. Create a stack and entitle it ***YourName*--FlashCard**. Give the stack an appropriate label (based on the language with which you are working).
2. Identify at least 10 words you would like to drill and incorporate them in your stack.
3. Put a copy of your finished stack in the **Assignments** folder.

There you have it. As before, this is only one of a myriad of ways of approaching this type of activity. This is not necessarily the most effective or most efficient, but it works and works well. Hopefully, this will inspire your minds as to the possibilities that exist with Revolution and spark some creativity of your own.

Course Schedule
Main Page

Sample Scripts

These are provided as an aid. Copying and pasting directly into a script tends to diminish the efficacy of this assignment. Use them at your discretion.

```
on mouseUp
  hide field "EngWords"
  hide field "FrenchWords"
  hide field "EngWords Master"
  hide field "FrenchWords Master"
  set the hilight of button "Edit Vocabulary" to false
  put field "EngWords Master" into card field "EngWords"
  put field "FrenchWords Master" into card field "FrenchWords"
  show btn "cover"
  pick
end mouseUp

on mouseUp
  put fld "English" & return before fld "EngWords"
  put fld "French" & return before fld "FrenchWords"
  show btn "cover"
  pick
end mouseUp

on pick
  put number of lines of field "EngWords" into lineCount
  if lineCount = 0 then
```

```
    put empty into field "English"
    put empty into field "French"
    answer "YouÕve finished all the words."
    exit to top
  end if
  put random(lineCount) into whichLine
  put line whichLine of field "EngWords" into field "English"
  delete line whichLine of field "EngWords"
  put line whichLine of field "FrenchWords" into field "French"
  delete line whichline of field "FrenchWords"
end pick

on pick
  if the selectedText of btn "selectionTabs" contains "English" then
    set the promptLanguage of this card to "French"
    set the responseLanguage of this card to "English"
  else
    set the promptLanguage of this card to "English"
    set the responseLanguage of this card to "French"
  end if
  put empty into field "prompt"
  put empty into field "response"
  put number of lines of field "EnglishWords" into lineCount
  if lineCount = 0 then
    answer "YouÕve finished all the words."
    exit to top
  end if
  put the promptLanguage of this card into pLang
  put pLang & "Words" into promptFld
  put the responseLanguage of this card into rLang
  put rLang& "Words" into responseFld
  put random(lineCount) into whichLine
  put line whichLine of field promptFld into field "prompt"
  set the correctResponse of field "response" to line whichLine of field responseFld
  delete line whichLine of field "EnglishWords"
  delete line whichline of field "FrenchWords"
  select the text of fld "response"
end pick
```