QCard DLL Lesson 3

Lesson 2 Lesson 4

-
Alyce

QCard DLL Lesson 3 | Creating a Game | An Array of Cards | Shuffle the Cards | Dealing the Cards | DEMO

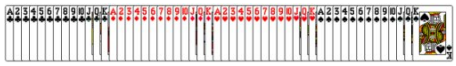
See Lesson 1 for QCard DLL and WAV files needed for the demo code.

Creating a Game

Now we start a real game! This program will be the game called Memory. Cards will be dealt in rows and columns face down, and the user will turn them face up in pairs. When he turns up a pari of identical cards they are removed from the table. Each turn counts as a point and the object is to get the fewest points possible in solving the puzzle. If the cards don't match they are turned face down for another try. We have a long way to go and we'll take it a step at a time.

An Array of Cards

The first step is to select the cards for this game and place them into an array. Qcard has two decks available. The first deck has indices of 1-52, while the second deck is comprised of 53-104. We can use any of these cards in our game. We can use a single deck or both decks, or selected cards. We'll use just the face cards (Jack, Queen, King) from both decks for the memory game. The face cards in the first deck have indices of (11,12,13), (24,23,25), (37,39,39), (50,51,52.) The cards in the DLL are arranged Ace through King in each suit, with clubs first, then diamonds, then hearts, then spades.



The second deck begins at card 53 and is arranged the same as the first deck. Adding 52 to each index in the first deck gives us the identical card in the second deck.

```
[fillCardArray]
    'fill card array
    'cards 1 to 52 are in the first deck
    'cards 53 to 104 are in the second deck
    'use cards Jack through King in each suit, first deck
    card(1)=11  'jack of clubs
```

```
card(2)=12  'queen
card(3)=13  'king
card(4)=24  'jack of diamonds
card(5)=25  'queen
card(6)=26  'king
card(7)=37  'jack of hearts
card(8)=38  'queen
card(9)=39  'king
card(10)=50  'jack of spades
card(11)=51  'queen
card(12)=52  'king

'now use second deck, to fill second half of array
for i = 1 to 12
    card(i+12)=card(i)+52
next
```

Shuffle the Cards

Now that we have the indices of the cards for this game in an array we must shuffle them. If we didn't shuffle them the cards would always be in the same place. That wouldn't be much of a game!

This method of shuffling is quite simple. For each element in the card array we select a random slot in the array and switch that card with the one in the random slot.

For each switch, first select a random array element. Since RND returns a number between 0 and 1, we multiply the result by 24 and take the integer value, then add 1 because we are not using item 0 in the array.

```
newIndex=int(rnd(0)*24)+1
```

We next fill a temporary variable with the contents of the array we'll be switching.

```
tempCard=card(i)  'temp var to allow switching values
```

We can now make the switch. We set the value of the array element equal to the contents of the randomly chosen element.

```
card(i)=card(newIndex)
```

```
'this index now contains value from random index
```

Now we fill the element at the random index with the value of the original index. We can do this because we placed it into a temporary variable before filling that array element with the new value.

```
    card(newIndex)=tempCard
'random index now contains value from other index
    'now card(i) has switched values with a random card in the array
```

We make this random switch for each card in the array by doing it in a FOR...NEXT loop.

```
[shuffleCards]
    playwave "shuffle.wav",async
    'now shuffle cards
    for i = 1 to 24
        newIndex=int(rnd(0)*24)+1
        tempCard=card(i)  'temp var to allow switching values
        card(i)=card(newIndex)
'this index now contains value from random index
        card(newIndex)=tempCard
'random index now contains value from other index

'now card(i) has switched values with a random card in the array
    next
    playwave "shuffle.wav",sync
```

Dealing the Cards

Previous tutorials discussed dealing the cards and setting their status to "face up" or "face down."

We've placed the dealing routine in a sub for easy access. The sub requires the handle of the graphicbox or graphics window, the index of the card to deal, and the x,y location to place the dealt card.

```
Sub DealCard hndle,nC,x,y
    'places card on window whose handle is hndle at x,y
    'nC is number of card - 1-52 in first deck and
    '53-104 in second deck, if used
    calldll #qc, "DealCard",hndle as ulong,nC as long,_
    x as long,y as long,r as void
```

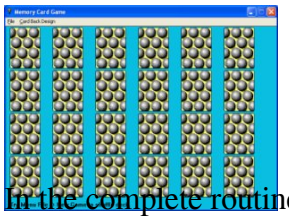
```
End Sub
```

Whenever we want to deal a card, we can call this sub. We need columns and rows of cards, so the easiest way to deal the cards is to do it in a loop. If we didn't use a loop, we'd have to deal each of the 24 cards individually. We set the x,y location of the first card:

```
'set xy location to start deal
x=10:y=2
```

We then deal the cards in a FOR...NEXT loop, incrementing the x and y positions for each card. Notice that we check the x location to see if we've reached the end of a row. If we have, we set x back to the first column and advance the y value to start the next row.

```
for i = 1 to 24
    Call DealCard hBox,card(i),x,y
    x=x+100
    if x>510 then   'move to next row
        x=10
        y=y+100
    end if
next
```



In the complete routine, we set each card to be displayed face down and play a wav that sounds like a card being dealt on a table. We also pause for a short time to slow the action so that it looks more realistic.

```
'set xy location to start deal
x=10:y=2
for i = 1 to 24
    'set status of all cards to 0, which is face down
    ' - we won't do this yet, so we can see the results
    'of our deal
    'call SetCardStatus card(i), 0

    Call DealCard hBox,card(i),x,y
    x=x+100
```

```
        if x>510 then   'move to next row
            x=10
            y=y+100
        end if
        playwave "card.wav",sync

       'pause 100 milliseconds between cards
        call Pause 100
        scan
    next
```

DEMO

See Lesson 1 for QCard DLL and WAV files needed for the demo code.

```
'An open project card game, begun by Alyce Watson, May 27, 2003.
'Uses Qcard32.dll, a freeware library of playing card images.
'DLL by Stephen Murphy.  Qcard32.DLL website:
'http://www.telusplanet.net/public/stevem/

'cards3.bas
'Now we start a real game!  This program will be the game
'called Memory.  Cards will be dealt in rows and columns,
'face down, and the user will turn them face up in pairs.
'When he turns up a pair of identical cards, they are
'removed from the table.  Each turn counts as a point and the
'object is to get the fewest points possible in solving the
'puzzle.  If the cards don't match, they are turned face
'down for another try. We have a long way to go and we'll
'take it a step at a time.

'new in cards3.bas:
'Card array is created and filled with Jacks, Queens,
'and Kings from both decks.
'Cards are shuffled for new game.
'Cards are dealt in 6 columns, 4 rows.


'new this time:
dim card(24)            'array to hold cards
gosub [fillCardArray]   'fill array with card values
newIndex=0              'used when shuffling
tempCard=0              'temp var used when shuffling
```

```
[varSetup]
i=0         'i will be our counter var in for/next loops
design=1    'default design is circles

nomainwin
    WindowWidth=640:WindowHeight=480
    UpperLeftX=1:UpperLeftY=1

    menu #1, "&File", "&New",[new],"E&xit", [quit]
    menu #1, "&Card Back Design","&Circles",[circles],"&Blue",[blue],_
    "&Red",[red],"&Mountain",[mountain],"&Purple",[purple],
"M&usic",[music]
    graphicbox #1.g, 0, 0, 640, 440
    open "Memory Card Game" for window_nf as #1
    #1 "trapclose [quit]"

    'get graphicbox handle
    hBox=hwnd(#1.g)

    'open the dll
    open "qcard32.dll" for dll as #qc
    'initialize the deck
    Call InitializeDeck hBox

[new]
    Call SetDefaultValues
    Call SetCurrentBack design

    'draw a nice background
    #1.g "down; fill 10 190 225"
    #1.g "backcolor 10 190 225"
    'temp message for this demo only
    #1.g "place 10 420"
    #1.g "\Try Menu File -> New Game to shuffle deck."
    gosub [shuffleCards]

    'set xy location to start deal
    x=10:y=2
    for i = 1 to 24
        'set status of all cards to 0, which is face down
        ' - we won't do this yet, so we can see the results
        'of our deal
        'call SetCardStatus card(i), 0

        Call DealCard hBox,card(i),x,y
```

```
        x=x+100
        if x>510 then   'move to next row
            x=10
            y=y+100
        end if
        playwave "card.wav",sync

       'pause 100 milliseconds between cards
        call Pause 100
        scan
    next

wait


'setting new card back doesn't restart game,
'so new back won't show until new game is started:
[circles] design=1:goto [setDesign]
[blue] design=2:goto [setDesign]
[red] design=3:goto [setDesign]
[mountain] design=4:goto [setDesign]
[purple] design=5:goto [setDesign]
[music] design=6:goto [setDesign]

[setDesign]
    Call SetCurrentBack design
    'design can be 1,2,3,4,5,6 for 6 possible designs
    wait


[fillCardArray]
    'fill card array
    'cards 1 to 52 are in the first deck
    'cards 53 to 104 are in the second deck
    'use cards Jack through King in each suit, first deck
    card(1)=11  'jack of clubs
    card(2)=12  'queen
    card(3)=13  'king
    card(4)=24  'jack of diamonds
    card(5)=25  'queen
    card(6)=26  'king
    card(7)=37  'jack of hearts
    card(8)=38  'queen
    card(9)=39  'king
    card(10)=50  'jack of spades
    card(11)=51  'queen
```

```
    card(12)=52  'king

    'now use second deck, to fill second half of array
    for i = 1 to 12
        card(i+12)=card(i)+52
    next
    RETURN


[shuffleCards]
    playwave "shuffle.wav",async
    'now shuffle cards
    for i = 1 to 24
        newIndex=int(rnd(0)*24)+1
        tempCard=card(i)  'temp var to allow switching values
        card(i)=card(newIndex)
'this index now contains value from random index
        card(newIndex)=tempCard
'random index now contains value from other index

'now card(i) has switched values with a random card in the array
    next
    playwave "shuffle.wav",sync
    RETURN

[quit] close #qc:close #1:end


''''''''''''''''''''
'subs and functions:
Sub Pause ms
    'pause ms number of milliseconds
    calldll #kernel32,"Sleep",_
    ms as long, re as void
    End Sub

Sub InitializeDeck hndle
    calldll #qc, "InitializeDeck",_
    hndle as ulong,r as long
    End Sub

Sub SetCardStatus nC,face
    'nC is number of card - 1-52 in first deck and
    '53-104 in second deck, if used
    'face:  0=facedown,1=faceup
    calldll #qc, "SetCardStatus",nC as long,_
```

```
    face as long,r as void
    End Sub

Sub DealCard hndle,nC,x,y
    'places card on window whose handle is hndle at x,y
    'nC is number of card - 1-52 in first deck and
    '53-104 in second deck, if used
    calldll #qc, "DealCard",hndle as ulong,nC as long,_
    x as long,y as long,r as void
    End Sub

Sub DrawBack hndle, nV, x, y
    'nV can be 1,2,3,4,5,6 for 6 possible designs
    'draws a cardback image on screen
    calldll #qc, "DrawBack",hndle as ulong,_
    nV as long,x as long,y as long,r as void
    End Sub

Sub SetCurrentBack nV
    'nV can be 1,2,3,4,5,6 for 6 possible designs
    calldll #qc, "SetCurrentBack",nV as long,r as void
    End Sub

Sub SetDefaultValues
    'reset all card properties back to their default values.
    calldll #qc, "SetDefaultValues",r as void
    End Sub
```

QCard DLL Lesson 3 | Creating a Game | An Array of Cards | Shuffle the Cards | Dealing the Cards | DEMO

Lesson 2 Lesson 4