## Keypad Lock part 2

**Create a cool keypad-like access to your secret pages. This is part 3 with more runtime script.**

**This article was written and published in December 2003, the second version of keypad script has been published in 4th of January 2004**

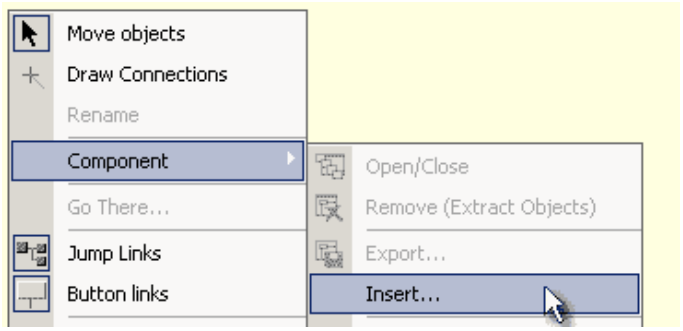
**Note: this text refer to a PRO version which is currently in beta.**
**A features presented here may not yet be available in the public beta version**

**A scripting support is added into 1.4 beta 2**

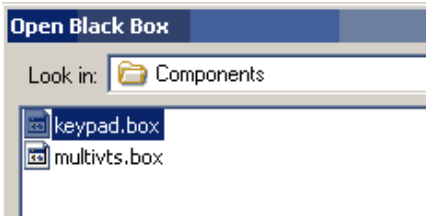
**In the Previous Part 1 we learned how to create the secret keypad lock and then we learned how to export it to a Component.**
**All is great except the component has always the same secret number (352 in our case) and if we want to change it we have to manually rewire the buttons.**

**But don't worry, since 1.4 beta 2 there is a script support in DVD-lab called lab-TALK. The Component can have its own Script which is executed after you insert the Component to Connections window.**
**Creating a script that re-wire the buttons is actually very easy (3 lines of code) plus one line to get the code from user!**

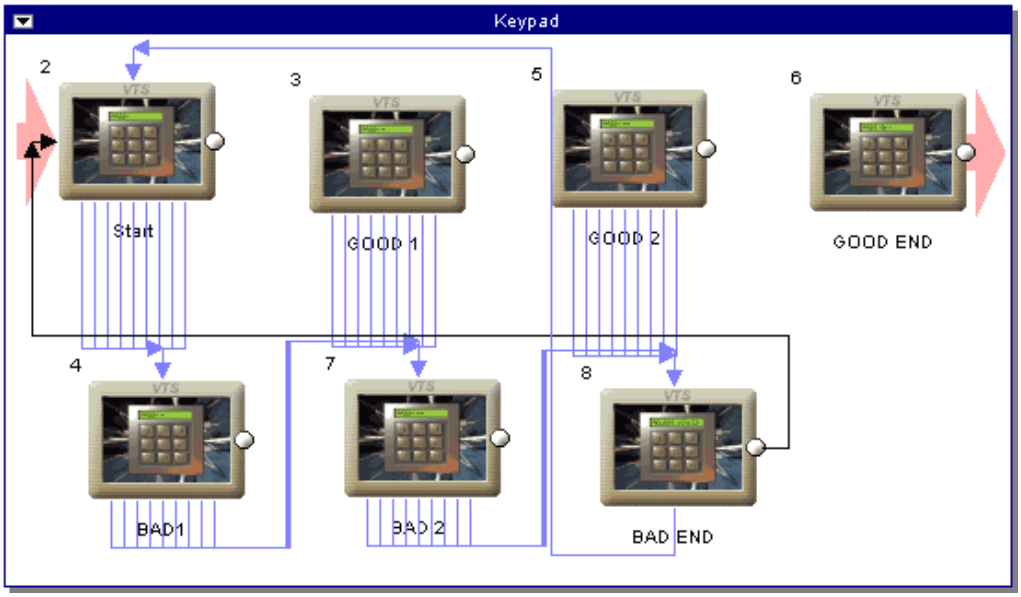
**First let's import our keypad.**
**Right click on the Connection window and select Component-Insert**




**Now select our previously exported keypad.box**

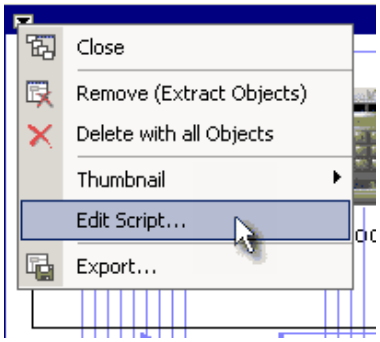



**Right now it is set for 352. We need an universal component so we will have to actually link all buttons from good menu (Button 3 on Start, Button 5 on GOOD1, Button 2 on GOOD 2) to the bad menus as the others buttons.**

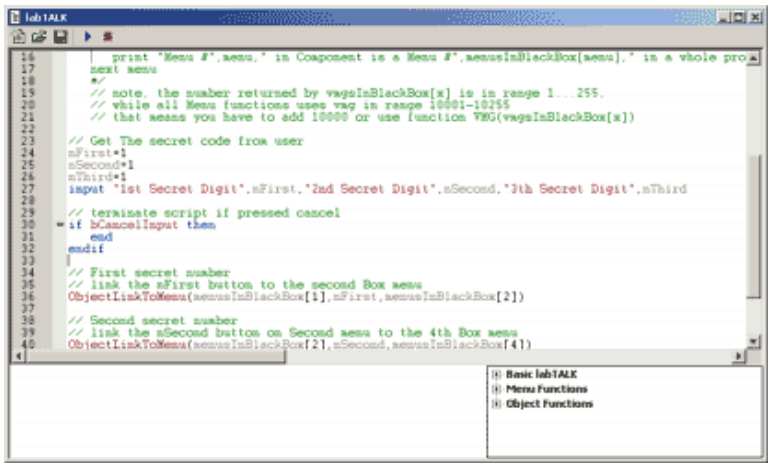



**Here it is, all buttons from Start links to BAD1, from GOOD 1 to BAD 2 and from GOOD 2 to BAD END. We didn't change anything else. Obviously right now the component is non-workable - an user will be never able to get to GOOD END - there is not even a link to it.**

**The script will create these good links for, so let's roll.**
**Click on the component arrow and select Edit Script.**




**You will get to a serious lab-TALK source code editor where you can type your commands. Don't worry, there is a small window at the right bottom with all commands and syntax listed.**



**lab-TALK language**

**The lab-TALK itself is a simple script based on BASIC. The core has just few commands for input, print, for loop, if condition, goto and gosub. .**
**The script editor is actually very pleasant to use - with syntax color coding and ability to hide long parts of the code.**

```
 1
 2    // simple code
 3    b=10
 4  - for i= 0 to 100
 5       //loop body
 6  -    if i>50 then
 7          b=b+2
 8  -    else
 9          b=b+1
10       endif
11       // now print this thing
12       print "Loop ",i," Value of b =",b
13    next i
14
```

**By clicking on the minus sign left of for we can hide the whole loop, which makes much easier to navigate a very large code**

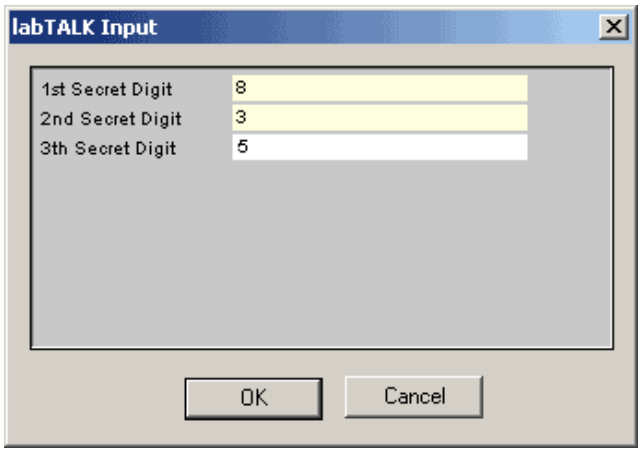
```
 1
 2    // simple code
 3    b=10
 4  + for i= 0 to 100
13    next i
14
```

**As you see the commands are easy and familiar but the real power is in the extension commands (specific to DVD-lab) that works with menus, objects even down to the pixel editing.**

**Let's code our case**
**The user input is easy:**

```
nFirst=1
nSecond=1
nThird=1
input "1st Secret Digit",nFirst,"2nd Secret Digit",nSecond,"3th Secret
Digit",nThird
```

**What this will do is to display a multiple input dialog box where user can type the numbers. The number will be assigned to our variables nFirst, nSecond and nThird. We can of course use just one input and then get each number with a little of math, but this is faster. Even now you can press Run command and this will be displayed (I already input the numbers 8,3,5)**

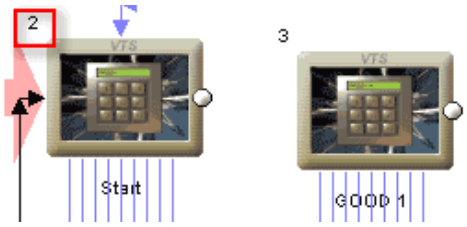



**Linking buttons on menus**

**As you look into the language helper box in Object section there is a simple command ObjectLinkToMenu(menufrom, object, menuto) to link an object(button) one menu to other menu. In lab-TALK all is based from 1 so first object is 1, first menu is 1 etc..**
**We know what are the objects, since our buttons on the menu are in order 1,2,3... corrensponding to the number written on them the object we are linking from on menu one will be nFirst.**

```
ObjectLinkToMenu(menuFrom,nFirst,menuTo)
```

**The question is now what are the menus. As you look at the image on very top of this page there is a small number on left top corner of each menu - this is the order of object in whole project.**




**Our Start menu has number 2. This is because I import the Component to a project where there already was a VTS menu (number 1), So in order for this work I will need to put a number menuFrom = 2 and menuTo = 3 like that:**

```
ObjectLinkToMenu(2,nFirst,3)
```

**Yes it will work, but only on project where I have already one menu. if I put such component to a project which has for example already 3 menus, then my Start menu would be number 4! So this code needs more work.**
**We need to know what is the number of a first menu in Component, then second etc...**

**Surprisingly, the Component itself will tell us this info ;-) by setting up few array variables for us: menusInBlackBox, vmgsInBlackBox, moviesInBlackBox**
**We can use these variables to map the box->project.**
**For example the number of all menus in the component is**

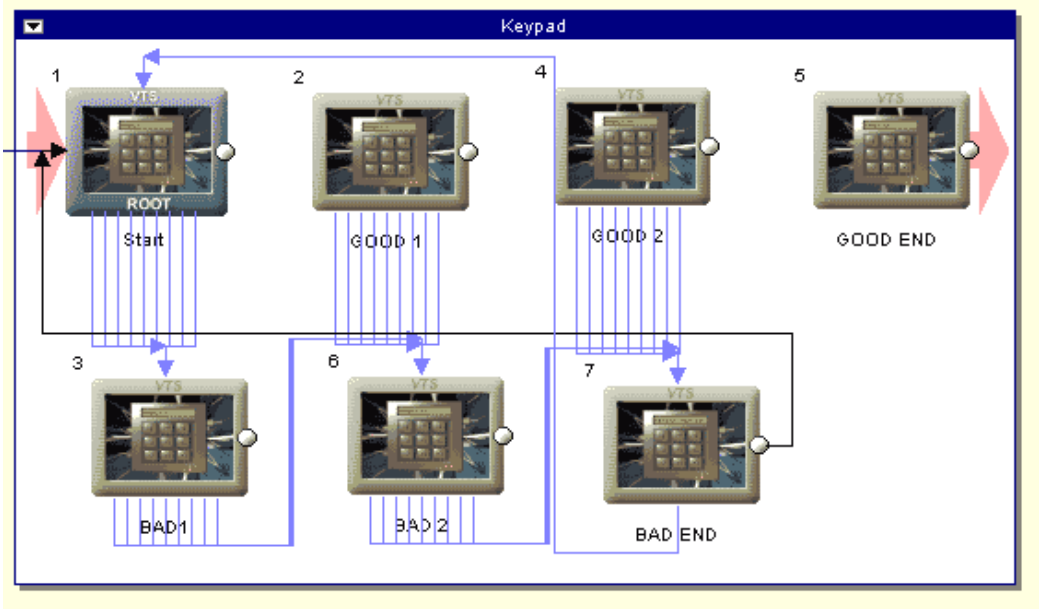
```
nMenus = menusInBlackBox
```

**then in lab-TALK we can use the very same variable as array**

```
menu1 = menusInBlackBox[1]
menu2 = menusInBlackBox[2]
etc..
```

**In our case the menu1 will have value of 2 because our first menu in Component is a second menu in project - but just in our case.**
**So instead of putting a constant numbers for menus we always use** `menusInBlackBox[number]` **which will do the mapping on runtime for us.**

**Just to see the correct number and avoid some mistake, I deleted the first VTS menu I had in the project so all numbers shifted and the first menu in keypad component is now also a first menu in project - therefore it has number 1. Again I already learned I have to use** `menusInBlackBox[number]`




**So from the image above, looking at the left top numbers I see that I need to create link from menu 1 to menu 2, from menu 2 to menu 4 and from menu 4 to menu 5. The code using the info from above will look like:**

```
// First secret number
// link the nFirst button to the second Box menu
ObjectLinkToMenu(menusInBlackBox[1],nFirst,menusInBlackBox[2])

// Second secret number
// link the nSecond button on Second menu to the 4th Box menu
ObjectLinkToMenu(menusInBlackBox[2],nSecond,menusInBlackBox[4])

// Third secret number
// link the nThird button on 4th menu to the 5th Box menu
```

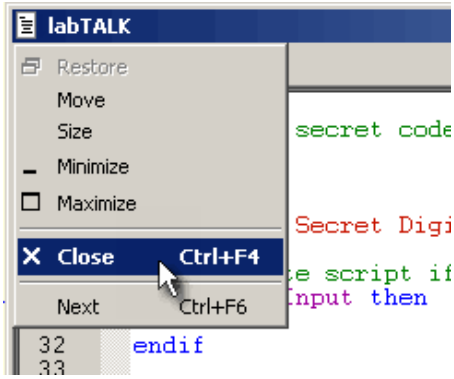
```
ObjectLinkToMenu(menusInBlackBox[4],nThird,menusInBlackBox[5])
```

**And yes this is the correct code that works.**
**My whole script looks like the one below, note I added ability to terminate the script if user press Cancel on the input box:**

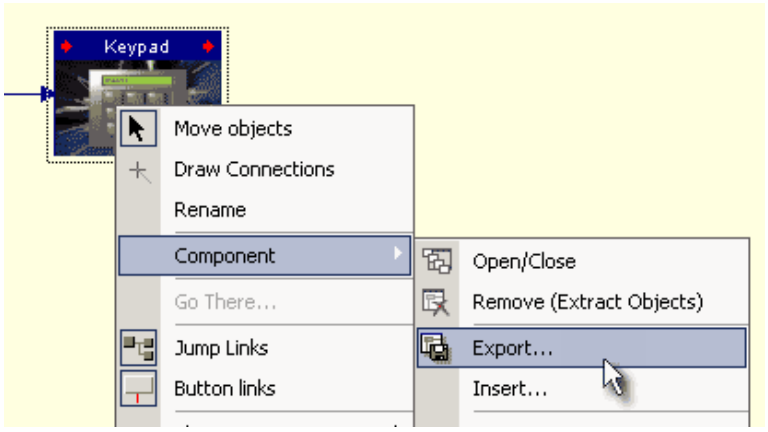
```
nFirst=1
nSecond=1
nThird=1
input "1st Secret Digit",nFirst,"2nd Secret Digit",nSecond,"3th Secret
Digit",nThird

// terminate the script if pressed cancel
if bCancelInput then
end
endif

// First secret number
// link the nFirst button to the second Box menu
ObjectLinkToMenu(menusInBlackBox[1],nFirst,menusInBlackBox[2])

// Second secret number
// link the nSecond button on Second menu to the 4th Box menu
ObjectLinkToMenu(menusInBlackBox[2],nSecond,menusInBlackBox[4])

// Third secret number
// link the nThird button on 4th menu to the 5th Box menu
ObjectLinkToMenu(menusInBlackBox[4],nThird,menusInBlackBox[5])
```

**Remember, when you will try the script by pressing Run button, the Component will change so before Exporting it you have to set it to the initial state - no links to good menus.**

**Now all I need to do is to close the script window so the script updates inside the component,**




**Then minimize the Component itself by dobleclicking on it, then right click on the thumbnail and select Component-Export. Again, once more, remember to export it in its initial state where no button links to a good menu!**




**Our first "smart" component is born. Now whenever you Insert the component to a project it will ask you for the secret code.**

---