Window objects:

The foundations of an application

This topic introduces you to the card and stack objects – the foundations of the different windows, dialog boxes and palettes that your application might need to use. You'll learn how to create stacks of cards, each with their own content, for these different purposes.

Key topics covered in this tutorial

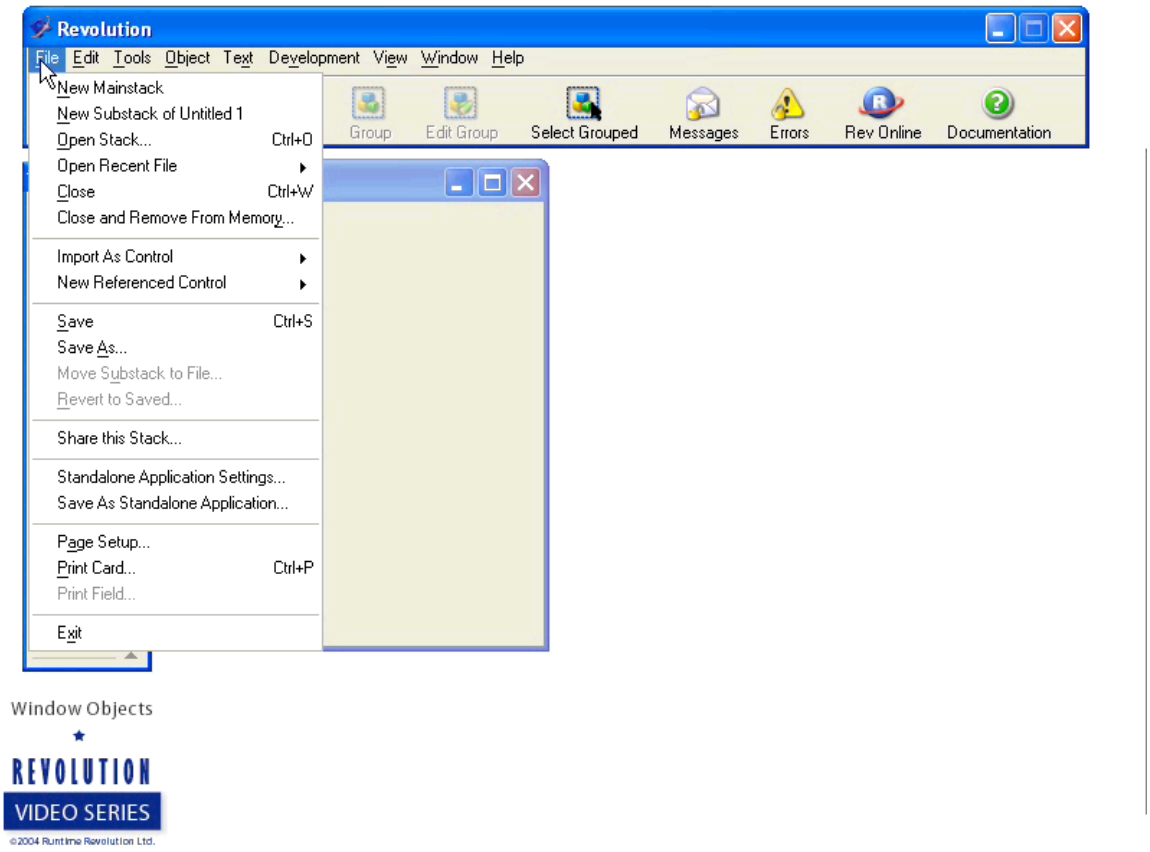
- Using stacks to create multiple windows in your application
- Using cards to display multiple views within a window
- Using backgrounds to place common objects on groups of cards
- Using fields on backgrounds
- Using backgrounds with the tabbed object

See also: Reference: mainStacks, subStacks and the organization of the stack file; groups and backgrounds; windows, palettes and dialog boxes

---

So far in these tutorials, we have used a stack with a single card for our application building. Usually, applications are more complicated than this – they display different things within a window, they might have multiple windows or need palettes or custom dialog boxes.

You can create multiple windows in your application by creating multiple stacks. The first stack you create is called a mainStack, it opens when your application starts and contains the other stacks, known as subStacks. You can use subStacks to create dialog boxes, palettes, custom menus and any other kind of window that your application might need. The mainStack and its subStacks will save into a single file, or create a single application to share online, or a standalone executable application if you have the Studio or Enterprise editions of Revolution.



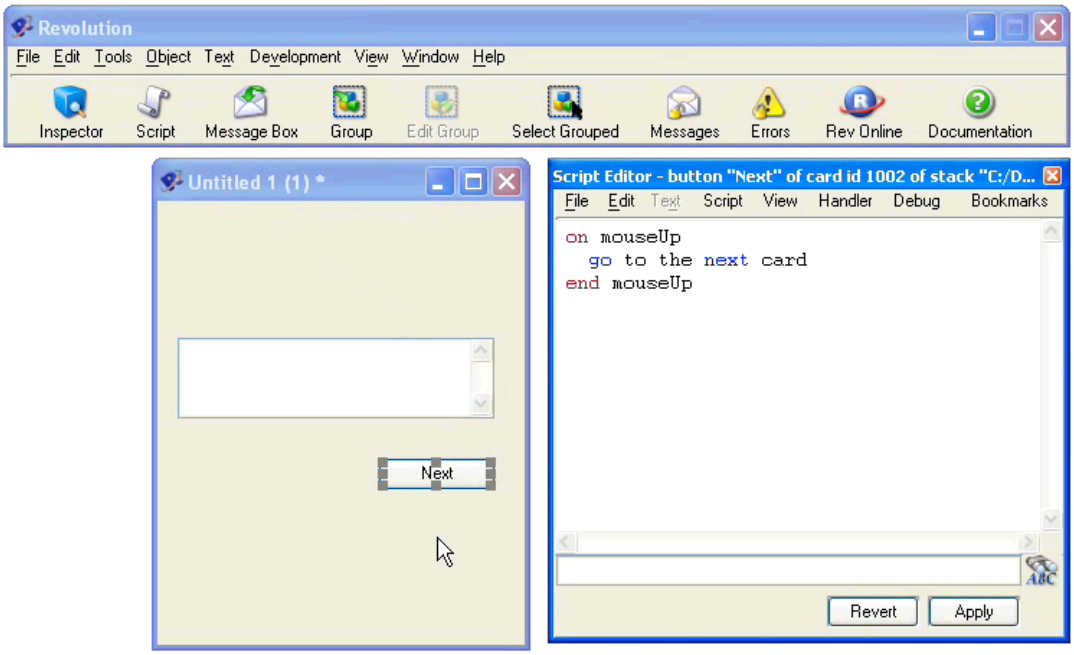
Window Objects

REVOLUTION
VIDEO SERIES

Let's try creating a stack with two cards to start with – first we'll lay out some objects on the first card, then let's create a new card (using the 'New Card' option in the 'Object' menu). This creates a new blank card that we can put more objects on. We can navigate between these cards using the 'View' menu. It's really easy to write a script and put it in a button so the user can switch between cards. We'll use the 'Properties Inspector' to call the first button 'Next'. and then we can add a short script:

```
go to the next card
```

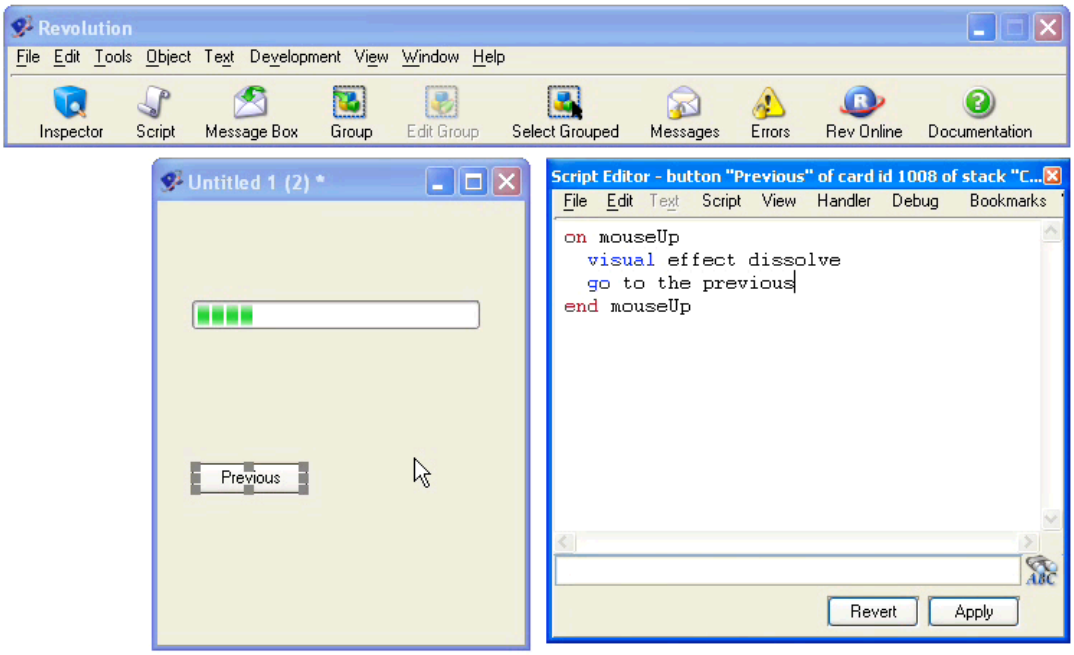
Now if we switch to run mode we can navigate to the next card.



.

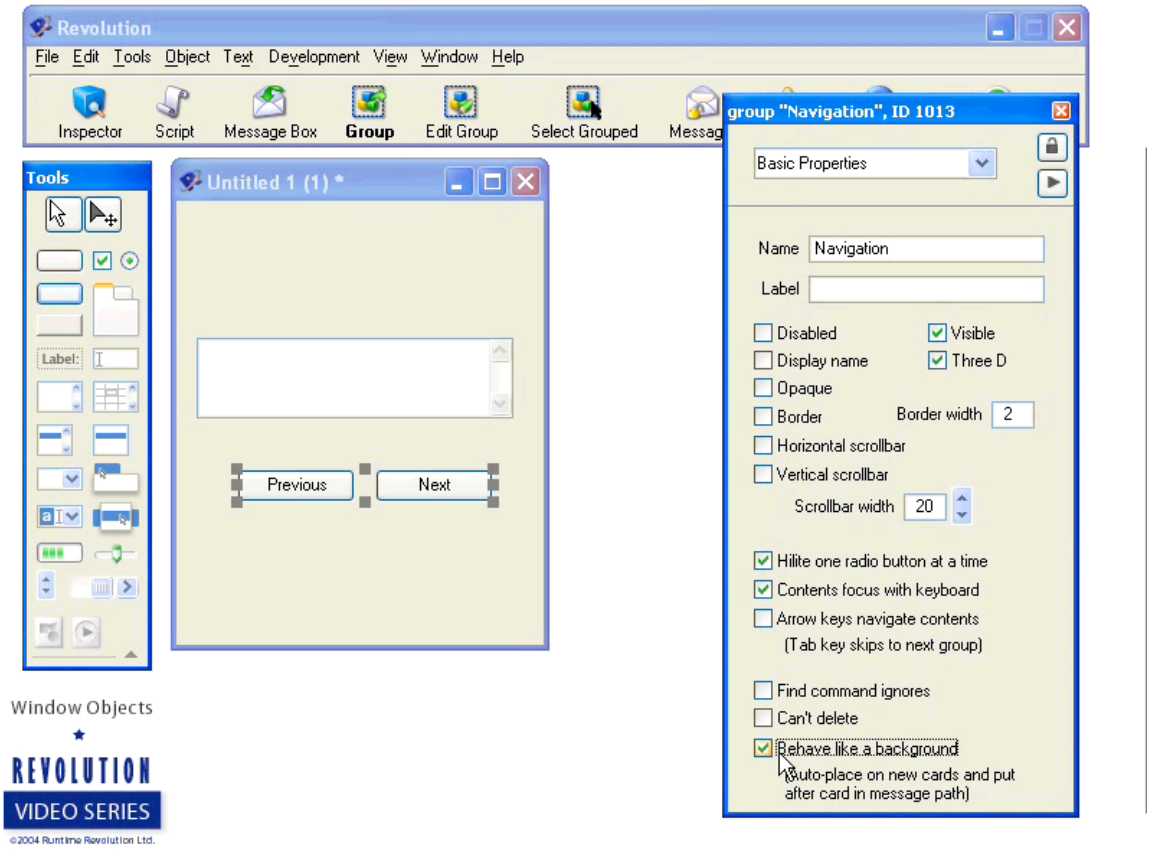
We can now edit the button on this card to allow us to go back. We start by changing the name of this button to 'back'. This time let's imagine we are producing an interactive presentation and add an effect to our script:

```
visual effect dissolve

go to the previous card
```

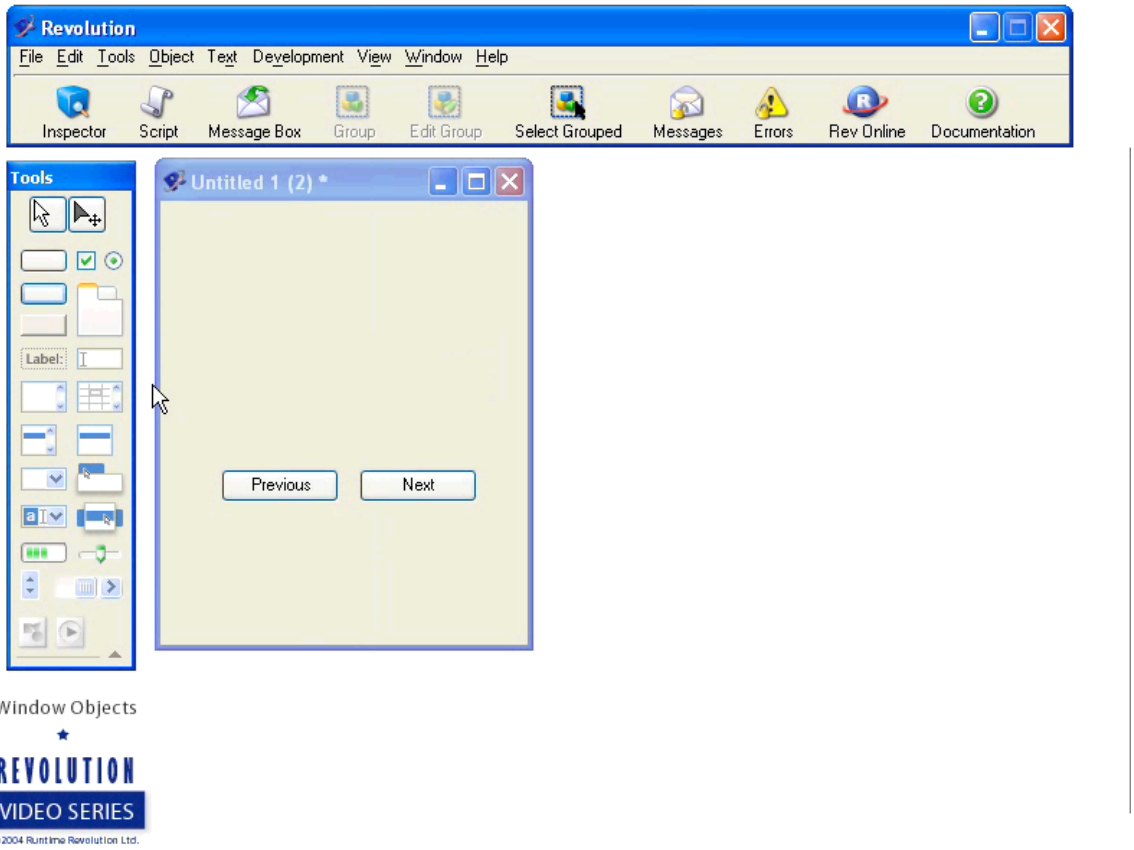


Cards can share groups containing objects with other cards. In this way, the same objects can appear on more than one card. In Revolution groups of objects which are shared between cards like this are also known as backgrounds. Let's say that we want our 'Next' and 'Previous' buttons to appear on both cards. First, let's get the objects together on the same card – as they are both on different ones at the moment. We'll do this by moving the 'Previous' button. First, select the 'Previous' button, then cut it. Go back to the first card and paste.

Now that the objects are both on one card, we'll put them in a group by selecting them both and clicking the 'Group' button in the Toolbar. Let's name this group 'Navigation' (using the properties palette). To place this group automatically onto new cards you create, you need to make it behave like a background. To do this click the 'Behave like a background' checkbox in the properties palette.



Whenever you create a new card in Revolution, any groups that have been set to behave like backgrounds will be placed on them. When we create a new card the 'Navigation' group appears.



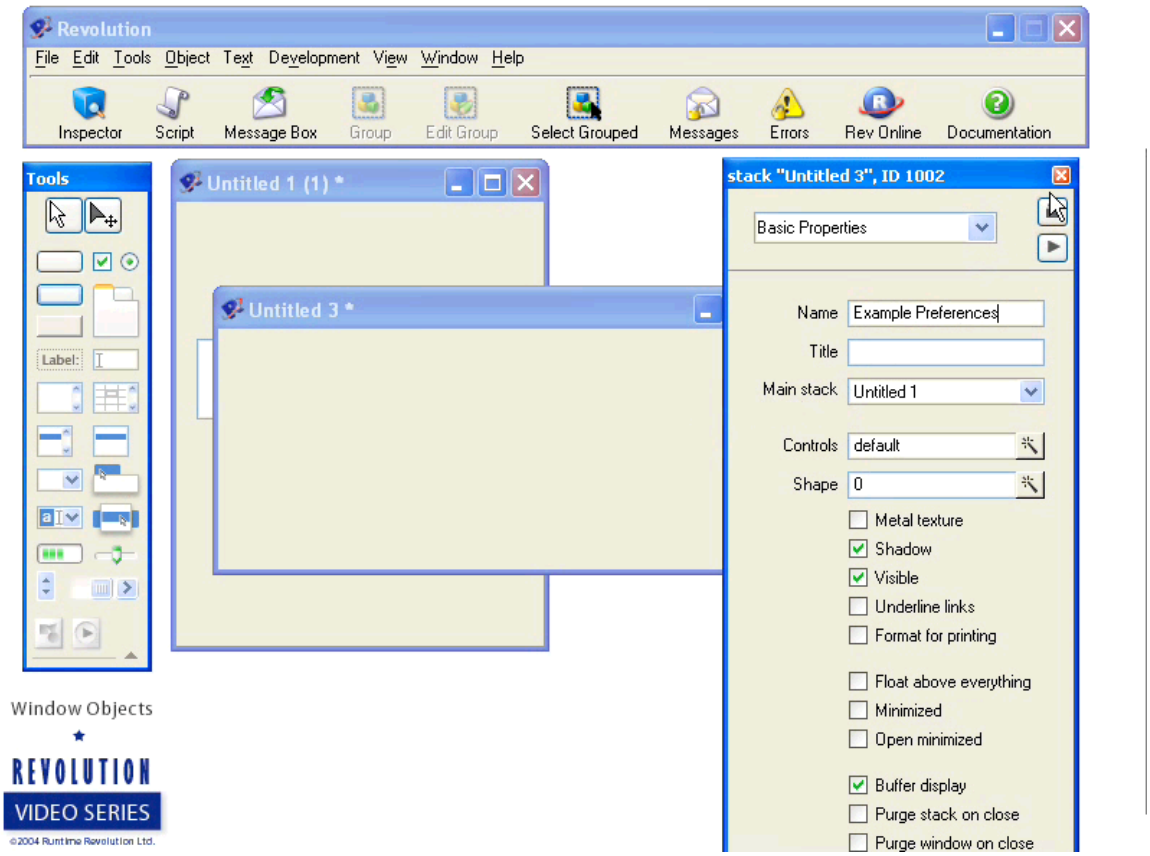
One thing to notice is that if we go to the final card you will see that the group isn't available here. The reason is that background groups only get placed automatically onto new cards created next to the card with the original group. You have to place the group on this card manually by choosing 'place group' from the object menu.

If we want to remove this group from a specific card, we can select the group, then choose 'Remove Group' from the 'Object' menu. The group is no longer present on this card, but is present on the next card. This makes it easy to place or remove groups of controls. You can create as many groups as you need, and place some of them on specific cards, or some of them on all cards.

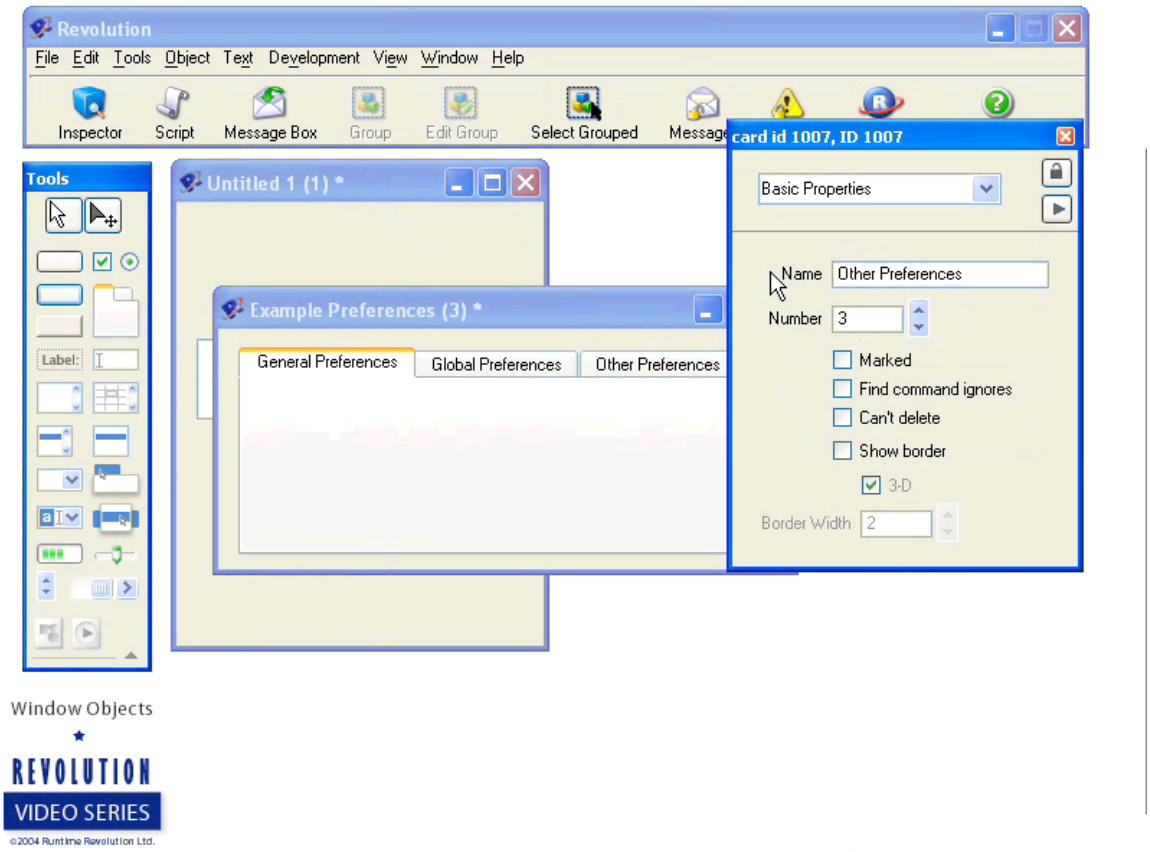
Text fields are even more flexible when they are used in background groups. Not only can a background field be used to display the same text on consecutive cards, it can be set to carry different text for each card it appears on.

A simple example would be an address book with two fields in the background. One might say 'My address book', which looks the same on every card, the other is a field to enter people's details into – which will have different text in it on each card for each person.

Let's take a look at another example of using a background group - using a tabbed interface. Suppose you are creating a preferences window in your application which contains three different screens, each accessible by tabs at the top. First, we can create a new substack for the preferences. Let's name the stack 'Example Preferences'. To do this you open the stack inspector and edit the name. Note that the title of the window has changed to reflect the name.



Now we'll add a tabbed button and change the tabs to 'General Preferences', 'Global Preferences' and 'Other Preferences'. To do this you need to open the properties inspector for this tabbed button. Let's name this card the same as the first tab by loading up the card properties inspector. We're going to make the tabbed button appear on three different cards, one for each set of options. Let's make a group with the tabbed button and set this group to behave like a background. Now, we can create two more cards and name them 'Global Preferences' and 'Other Preferences'. Note that our tabbed button appears on all of the cards.



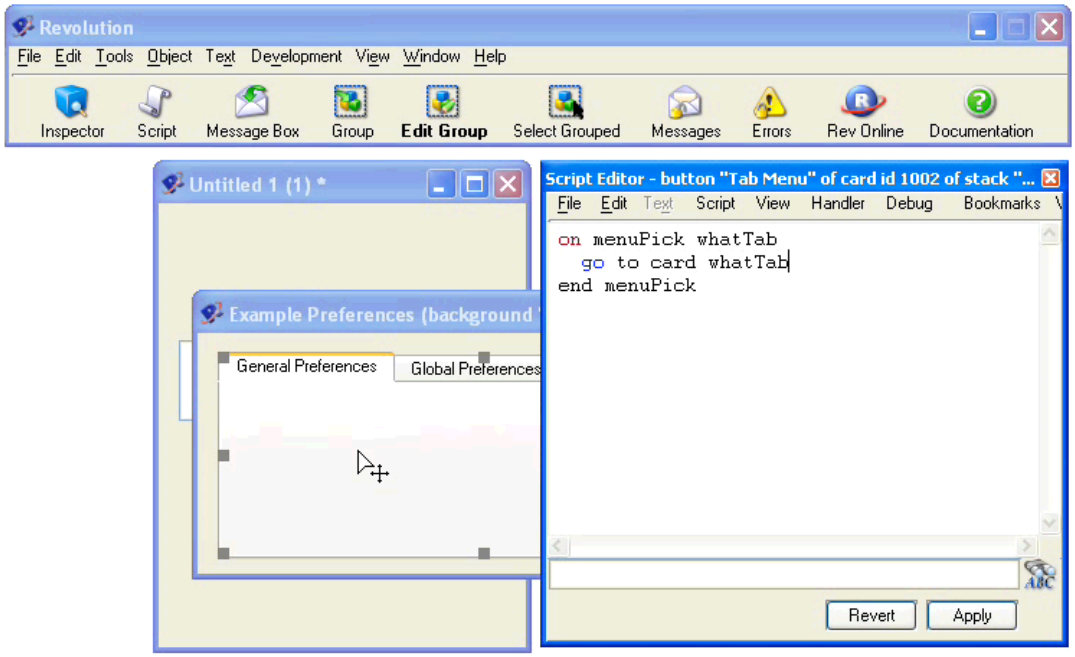
Window Objects

REVOLUTION VIDEO SERIES

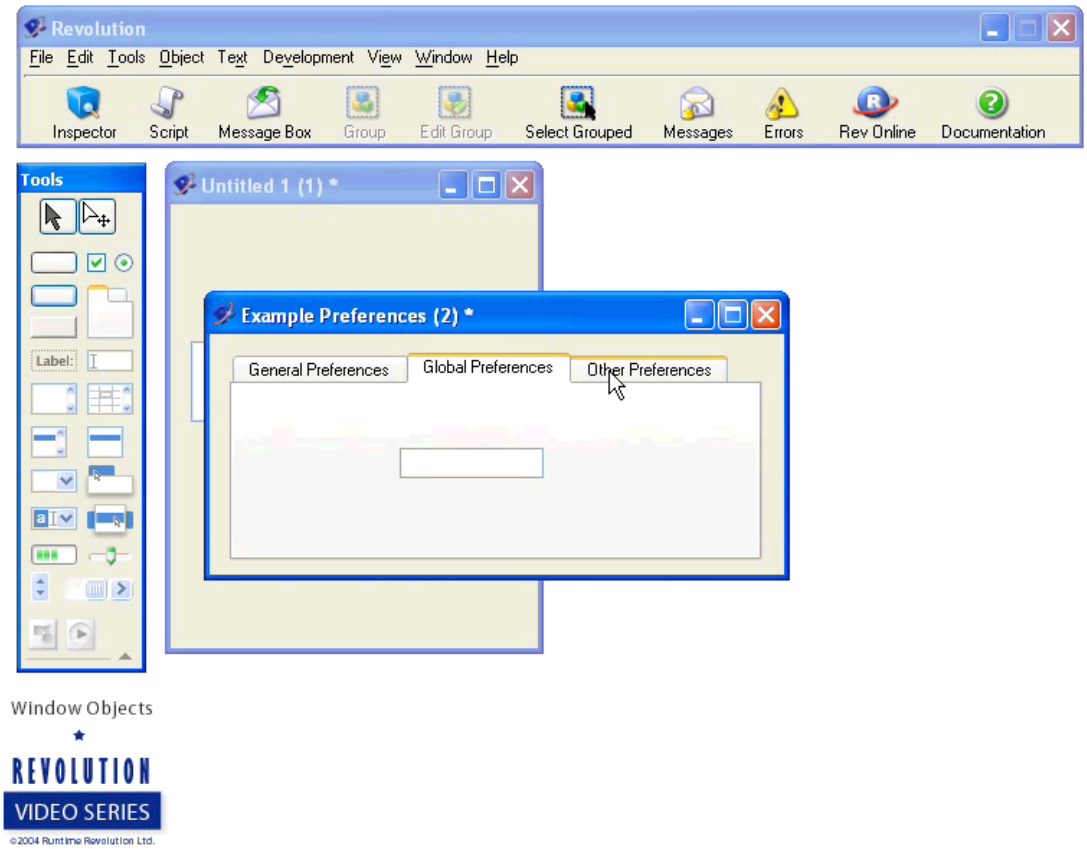
All we need to do now is add a simple script to navigate to each of the cards when we click on one of the tabs. We need to edit the script of the tabbed button. To do that, we'll select the group, then go into edit group mode (by clicking on the 'Edit Group' button in the Toolbar), then edit the script for the tabbed button. When a user clicks on a tab, a menuPick message is sent to the tabbed button. The menuPick message is sent with a parameter which contains the name of the tab clicked on. So the script we need is:

```
on menuPick whatTab

go to card whatTab

end menuPick
```

Remember that we have already named each card to be the same as the name of each tab. The word 'whatTab' will be replaced by the name of the tab the user clicks on. So if the user clicks on 'Global preferences' Revolution will run the script as if it was written to say 'go to card Global preferences'.

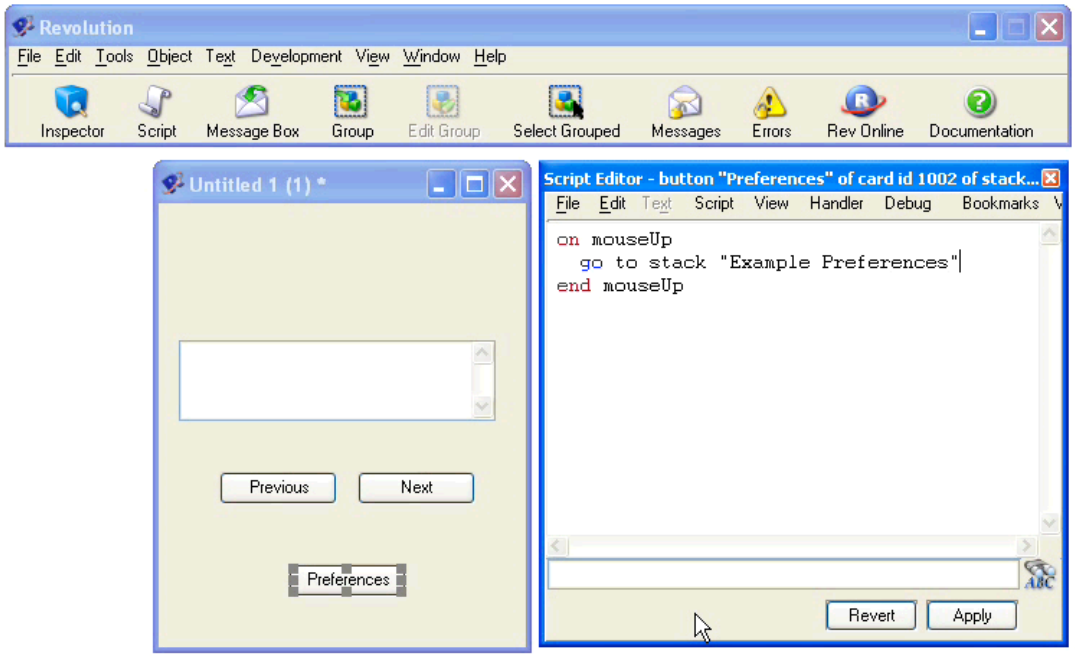


If we apply the script, and turn off edit group mode , and go into run mode, we can try it out. So that you can see that we are moving between different cards, let's add a different object to each card.



The final step is to be able to open the Preferences screen from the first stack we created. We need to switch back to the first stack, and add a new button naming it 'Preferences'. We can then add a simple script to this button:

go to stack "Example Preferences"



Appendix: Scripts used in this tutorial

```
on mouseUp
  go to the next card
end mouseUp

on mouseUp
  visual effect dissolve
  go to the previous card
end mouseUp

on mouseUp
  go to stack "Example preferences"
end mouseUp

on menuPick whatTab
  go to card whatTab
end menuPick
```