

---

## The Basics

We can use a `wxMultiChoiceDialog` object to present the user with a list of choices and allowing them to select one or more. This is a dialog and so we can use the `ShowModal` method to wait for the user to make her selection(s) and click Ok or Cancel.

```
dlg = wxMultiChoiceDialog(NULL, "Select an item", "my caption", LETTERS)
status = dlg$ShowModal()
```

Then we can query the results. We get back an integer vector giving the indices of the selected items in the `wxListBox` widget. Note that these are 0-based indices so we have to add 1 to them to identify the elements within our R vector.

```
ans = dlg$GetSelections()
print(LETTERS[ans + 1])
```

## Adding a Widget to a Dialog

Now, we want to go a little further with this example. When the user selects an item, we want to display some information about that in an adjacent widget. What we would like to be able to do is add a new widget to the `wxMultiChoiceDialog` display before it is displayed. Then we could arrange to put an event handler on the `wxListBox` to be called when the selection is changed and update the display in the newly added widget. To do this absolutely right is a little trickier than we may want, but we can do something quickly that displays the new widget. Basically, a dialog is a `wxWindow` and so we can find out what children it has with `wxWindow_GetChildren` or `dlg$GetChildren()`. When we do this, we see that there are 5 children: `wxStaticText`, `wxListBox`, `wxStaticLine`, `wxButton`, `wxButton`. These correspond to the title, the list of items, the line spacer between the list box and the buttons and the Ok and Cancel buttons. We can also ask for the `wxSizer` associated with this window, `dlg$GetSizer()`. If this is a `wxBoxSizer`, we can find its orientation with `sz$GetOrientation()`. And we can ask for its children and so find out a great deal about how things are laid out. What we would "like" to do is to insert a new widget, say a button for the moment, into the `wxSizer`. So we create a new button with the dialog as the parent. And then we use the `Insert` method for `wxSizer` and specify the position at which it is to be inserted. We want to put this after the `wxListBox` and before the `wxStaticLine`. So this will be the third widget and since we use 0-based indexing, this is position 2. `wxSizer_Insert` is very similar to `wxSizer_Add` and takes arguments controlling how the widget should be resized when the window changes size.

```
dlg = wxMultiChoiceDialog(NULL, "Select an item", "my caption", LETTERS)
sz = dlg$GetSizer()
btn = wxButton(dlg, wxID_ANY, "My Button")
sz$Insert(2, btn, 1, wxEXPAND, 0)
```

```
status = dlg$ShowModal()
```

There is nothing special about a `wxButton` in this example.

## Reorganizing the Widgets in the Dialog

At this point, we have done the basics and added a new widget. If we wanted to arrange the `wxListCtrl` and our new widget to be horizontally aligned, things are more challenging. We would need to create a new `wxBoxSizer` that was horizontally oriented. Then we could add the `wxListCtrl` and new widget to that and

---

then insert that new `wxBoxSizer` in position 2 (i.e. `index = 1` because of 0-based counting). But before we can do that, we would have to remove the `wxListCtrl` from the original `wxBoxSizer` associated with the `wxMultiChoiceDialog` window. We start by creating the `wxMultiChoiceDialog` in the usual way

```
dlg = wxMultiChoiceDialog(NULL, "Select an item", "my caption", LETTERS)
```

And then the following code does what we want. We get the sizer and remove the `wxListCtrl` widget from it.

```
sz = dlg$GetSizer()
listCtrl = dlg$GetChildren()[[2]]
sz$Remove(listCtrl)
```

Then we create the new sizer and add our widgets to it and then insert it into the dialog.

```
hsizer = wxBoxSizer(wxHORIZONTAL)
btn = wxButton(dlg, wxID_ANY, "My Button")
hsizer$Add(listCtrl, 1, wxEXPAND)
hsizer$Add(btn, 1, wxEXPAND)
sz$Insert(1, hsizer, 1, wxEXPAND, 0)
```

And now we can display the dialog and allow the user to select the items in the usual way.

```
status = dlg$ShowModal()
print(dlg$GetSelections())
```

Instead of reorganizing the widgets from an existing dialog class, it is often better to create a new type of `wxDialog`. You can do this using the `RwxDialog` class and implementing the `TransferDataToWindow` and `TransferDataFromWindow`. This is described in a separate "tutorial".