Dave's Development Blog

Software Development using Borland / Codegear / Embarcadero RAD Studio

Source Editor for the Active View

By David | July 13, 2019 0 Comment

While working through some bugs in one of my Open Tools API projects I came across some old code that I've written about here before that did not behave as I wanted it to.

Up until now the tools has been used with the Object Pascal language and as such the IDE has only needed to work with one file, the `.pas` file. However when working with C++ the tool wouldn't do want I wanted (insert comments in the `.h` file). The reasons for this was the use of `IOTASourceEditor.GetEditView()` / `IOTASourceEditor.EditViews[]`.

By using these you are specifically referencing an index into the view list which is usually one for Object Pascal but for C++ contains views for the `.cpp` and `.h` files.

So what I needed to use was `IOTAEditorServices.TopView` instead. Now that gives me the correct view for getting the cursor position but doesn't help me get the active editor code.

Historically I've used a function I wrote as below:

```
Function TBADIToolsAPIFunctions.ActiveSourceEditor : IOTASourceEditor;

Var
  MS : IOTAModuleServices;

Begin
  If Supports(BorlandIDEServices, IOTAModuleServices, MS) Then
    If Assigned(MS.CurrentModule) Then
      Result := SourceEditor(MS.CurrentModule);
End;
```

Notice that this function uses another function to get the `IOTASourceEditor` interface from the current module as follows:

```
Function TBADIToolsAPIFunctions.SourceEditor(Const Module : IOTAModule) :
IOTASourceEditor;

```

```
 Var
   iFileCount : Integer;
   i : Integer;

 Begin
   Result := Nil;
   If Not Assigned(Module) Then
     Exit;
   iFileCount := Module.GetModuleFileCount;
   For i := 0 To iFileCount - 1 Do
     If Module.GetModuleFileEditor(i).QueryInterface(IOTASourceEditor, Result) = S_OK
Then
       Break;
 End;
```

This code returns the **first** `IOTASourceEditor` interface from the module's list of editors and this is where the problem lies.

However if we look at the declaration of the methods of `IOTAEditorServices` where the `TopView` method is that we need to use (later), then you will see that there is a `TopBuffer` property which is of the type `IOTAEditBuffer` which is inherited from `IOTASourceEditor`. So we can write the following revised `ActiveSourceEditor` function to get the currently active editor view as follows:

```
Function TBADIToolsAPIFunctions.ActiveSourceEditor : IOTASourceEditor;

 Var
   ES : IOTAEditorServices;

 Begin
   Result := Nil;
   If Supports(BorlandIDEServices, IOTAEditorServices, ES) Then
     Result := ES.TopBuffer;
 End;
```

This now gives us the active source editor, so all we need to do to get the active view's cursor position is do something like the following:

```
...
Var
  EditorSvcs   : IOTAEditorServices;
   ...

Begin
  ...
  If Supports(BorlandIDEServices, IOTAEditorServices, EditorSvcs) Then
    Begin
```

```
      EV := EditorSvcs.TopView;
      ...
End;
```

Hopefully that helpful to you all.

regards
Dave.

Category: Browse and Doc It Delphi Open Tools API RAD Studio Tags: IOTAEditBuffer, IOTAEditorServices, IOTASourceEditor