PDFCreator 4.4.2 documentation

PREVIOUS | NEXT

Migrating from PDFCreator 1.7 to 2.0

This section provides an overview of the differences between the COM Interfaces of the old PDFCreator (1.7.3 and below) and the new PDFCreator version, starting with PDFCreator 2.0. The two different workflows will be described shortly and compared to each other. The description below will not go into detail what each object is capable of doing (have a look at the different manuals if you are interested in knowing all the different methods and properties), instead the concentration will be on the main workflow, i.e. printing files and applying the desired settings. At the bottom of this page, a flow chart demonstrates the two different workflows visually.

Modularity

The old COM Interface (v1.7.*) consisted of three different objects: the clsPDFCreator object, the clsPDFCreatorOptions and the clsPDFCreatorError. The first object leads the main conversion process, e.g. it starts the actual conversion on a file, sets the printer, sets the options and so on. Whereas the second one just sets the different settings options that PDFCreator offers to its users. The third object is just for success/failure report.

In opposite to that, the new COM Interface (v2.*) has five different objects: the Queue object, the PrintJob object, the PDFCreatorObj object, the Printers object, the OutputFiles object. We will only focus on the first three since they define the main workflow. The Queue object acts as a container for all jobs that should be converted and you have full control over that container using the Queue object. Each of these jobs is in turn wrapped into the PrintJob object, which allows you to set up specific settings for different jobs such as under which conversion profile each job should be converted. The PDFCreatorObj object allows to retrieve information about the current instance of PDFCreator, i.e. if it is running or not, but also to print files.

Differences

As we saw above the whole conversion process has been divided into many objects instead of just one object as it was in the old version. This allows the developers to be more flexible, e.g. you can print a file, enqueue it and simply postpone the conversion process whenever you like. The clsPDFCreator object of the old version, i.e. v1.7.* did the printing, the queueing as well as the conversion. Now, these main functionalities have been divided into: PDFCreatorObj object doing the printing, the Queue object doing the queueing and the PrintJob object doing the actual conversion.

When you wanted to change the settings in the old version, you had to use the clsPDFCreatorOptions object or again the clsPDFCreator object and set the PDFCreator settings to the desired values. As a clsPDFCreatorOptions object was created you had to assign this object to the clsPDFCreator object’s appropriate property. This kind of settings set up required to set the autosave property to the path that you wanted your file to be saved to. In the new version instead, the PrintJob object offers a method called SetProfileSetting(string name, string value) which allows to set different settings options. The key difference is that in the new version, the PrintJob object itself offers a ConvertTo(string path) method which has a new explicit workflow for the conversion of a file using the COM Interface. The old version just used the built-in autosave workflow, that’s why it is not reasonable to have the possibility to set the autosave setting as well as some other settings options on the PrintJob object.

Since we are working with C# and the .Net framework there is no need for an explicit object for error handling, if something goes wrong an exception is being thrown right away similarly to normal C# programming.

Advantages

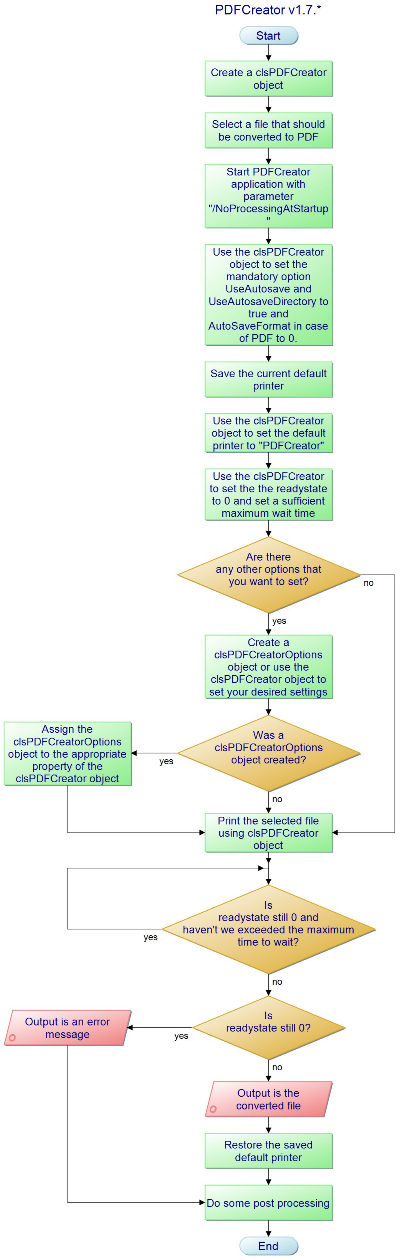
As mentioned above, the higher granularity of the new PDFCreator version makes a flexible use much easier. You could for instance wait for the arrival of 4 print jobs, after arriving let them wait in the queue and do other work of your software, then apply your desired settings to each individual print job and then again do a lot of your other work and eventually do the actual conversion.

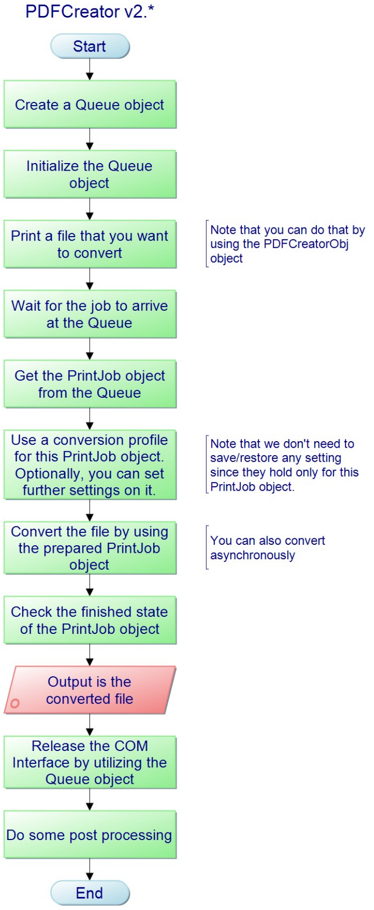
The newest version of PDFCreator will always be compatible with older versions (greater or equal to v2.2).

Further, we will drop support for PDFCreator 1.7.3 and below shortly.

Workflow Comparison

Below, you can see a flow chart describing the COM workflows of PDFCreator 1.7.x and PDFCreator 2.x. As you can see, the new workflow is much simpler while providing you the same flexibility in what you can achieve.


PDFCreator v1.7.*
Start
Create a clsPDFCreator object
Select a file that should be converted to PDF
Start PDFCreator application with parameter "/NoProcessingAtStartup"
Use the clsPDFCreator object to set the mandatory option UseAutosave and UseAutosaveDirectory to true and AutoSaveFormat in case of PDF to 0.
Save the current default printer
Use the clsPDFCreator object to set the default printer to "PDFCreator"
Use the clsPDFCreator to set the the readystate to 0 and set a sufficient maximum wait time
Are there any other options that you want to set?
no
yes
Create a clsPDFCreatorOptions object or use the clsPDFCreator object to set your desired settings
Was a clsPDFCreatorOptions object created?
yes
Assign the clsPDFCreatorOptions object to the appropriate property of the clsPDFCreator object
no
Print the selected file using clsPDFCreator object
Is readystate still 0 and haven't we exceeded the maximum time to wait?
yes
no
Is readystate still 0?
yes
Output is an error message
no
Output is the converted file
Restore the saved default printer
Do some post processing
End


PDFCreator v2.*
Start
Create a Queue object
Initialize the Queue object
Print a file that you want to convert
Note that you can do that by using the PDFCreatorObj object
Wait for the job to arrive at the Queue
Get the PrintJob object from the Queue
Use a conversion profile for this PrintJob object. Optionally, you can set further settings on it.
Note that we don't need to save/restore any setting since they hold only for this PrintJob object.
Convert the file by using the prepared PrintJob object
You can also convert asynchronously
Check the finished state of the PrintJob object
Output is the converted file
Release the COM Interface by utilizing the Queue object
Do some post processing
End

PREVIOUS | NEXT