

PDF to Text

Software Development Kit



INVESTINTECH.COM
P D F S O L U T I O N S



Copyright 2019 Investintech.com, Inc. All rights reserved

Word, Windows, Visual Basic, Visual C++, Visual C#, Visual Studio, .Net, PowerShell, Win32, Windows NT and Windows 10 are either Trademarks and/or Registered Trademarks of Microsoft Corporation (MS).

Adobe, Acrobat and Postscript are either Trademarks and/or Registered Trademarks of Adobe System Incorporated.

Contents:

1. Introduction	3
1.1. System Requirements	3
1.2. Getting more information	4
2. Installing PDF to Text SDK tools	5
3. Registration	10
3.1. Using Licence file	10
3.2. Using Password to activate licence	10
4. Using the Command Line Tool	11
4.1. Preparation	11
4.1.1. Adding Path variable (Optional)	12
4.2. Quick Single File conversion	13
4.3. Quick Multiple File Conversion	14
4.4. Command line arguments	15
4.4.1. Specifying input directory	18
4.4.2. Specifying output directory	19
4.4.3. Page ranges	21
4.4.4. Verbosity level of Console output	23
4.4.5. Converting password protected files	24
4.4.6. Ignoring Errors	25
4.4.7. Parallel Conversion	26
5. Using the Shared Library	28
5.1. Preparation	28
5.1.1. Licence Registration	28
5.2. Lib.c sample project	29
5.2.1. Resolving Dependencies	31
5.2.2. Running the sample project	32
5.2.3. Registration	34
5.3. Net C# sample project	34
5.3.1. Resolving Dependencies	35
5.3.2. Analyzing the SampleForm.cs	37
5.3.3. Running the Sample Application	38
6. Appendices	38
6.1. A -Troubleshooting: Command Line Tool	39
6.2. B - Troubleshooting: Shared Library	41

1. Introduction

The main purpose of this document is to provide you with a detailed guide on how to get started with the PDF2Text Command Line Tool and how to use Sample Files for developing an application using the PDF2Text Shared library.

PDF to Text SDK is a complete package that comes with the following components:

- ☐ PDF2Text Command Line tool
- ☐ Shared resources library
- ☐ Sample files

1.1. System Requirements

The following are the requirements for using the PDF2Text SDK tools comfortably:

- ☐ Windows 7, Windows 2008 or newer
- ☐ x86 Architecture CPU
- ☐ 512 MiB or more of system RAM
- ☐ At least 100 MiB of Storage Space

Besides these requirements, for SDK Dynamic Link Library Sample files, it is also recommended to have a software Development Environment such as Visual Studio, Visual Studio Code Eclipse, Borland Delphi or something similar.

1.2. Getting more information

Investintech.com, Inc. strives to provide the best possible technical support to its current and prospective customers. If you would like personal assistance, please feel free to call us or send in an email to our Customer Service or Technical Support teams.



Customer support

Telephone:	+1 416 920 5884
Hours:	9am - 6pm EST (GMT -5:00), Monday - Friday
Email:	cs@investintech.com



Technical support

Telephone:	+1 416 920 2539
Hours:	9am - 6pm EST (GMT -5:00), Monday - Friday
Email:	techsupport@investintech.com



Fax and Mailing Address

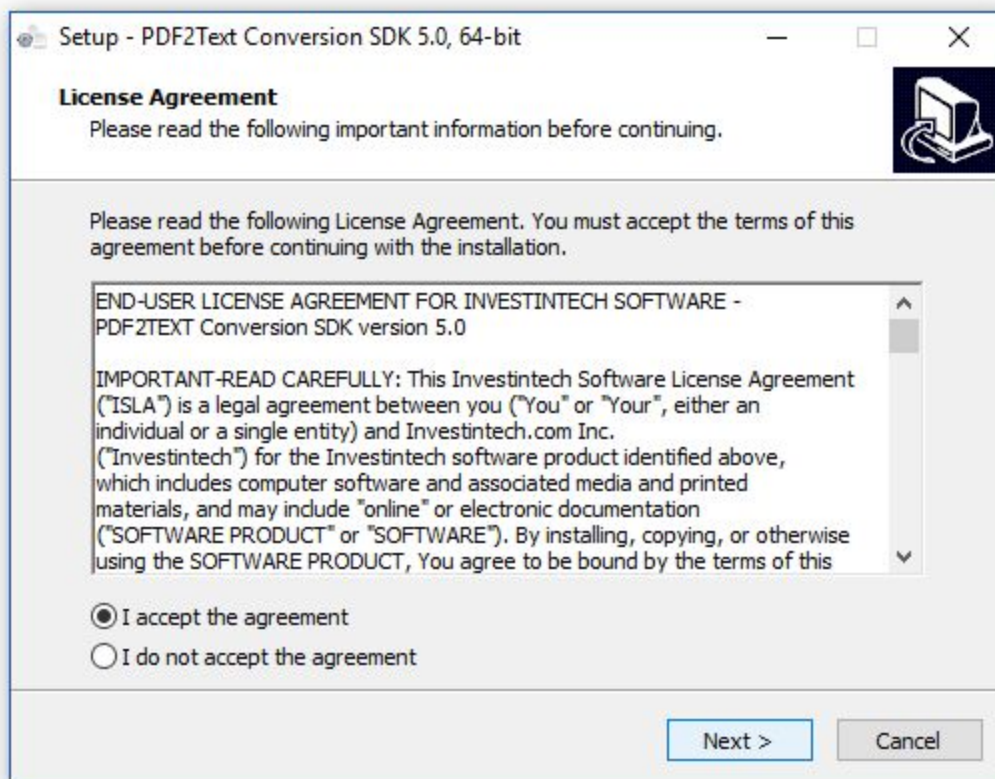
Fax:	+1 416 920 5848
Mail:	Investintech.com, Inc. 301 - 425 University Avenue Toronto, ON Canada M5G 1T6

2. Installing PDF to Text SDK tools

Thank you for choosing PDF to Text SDK tools from Investintech.com, Inc.

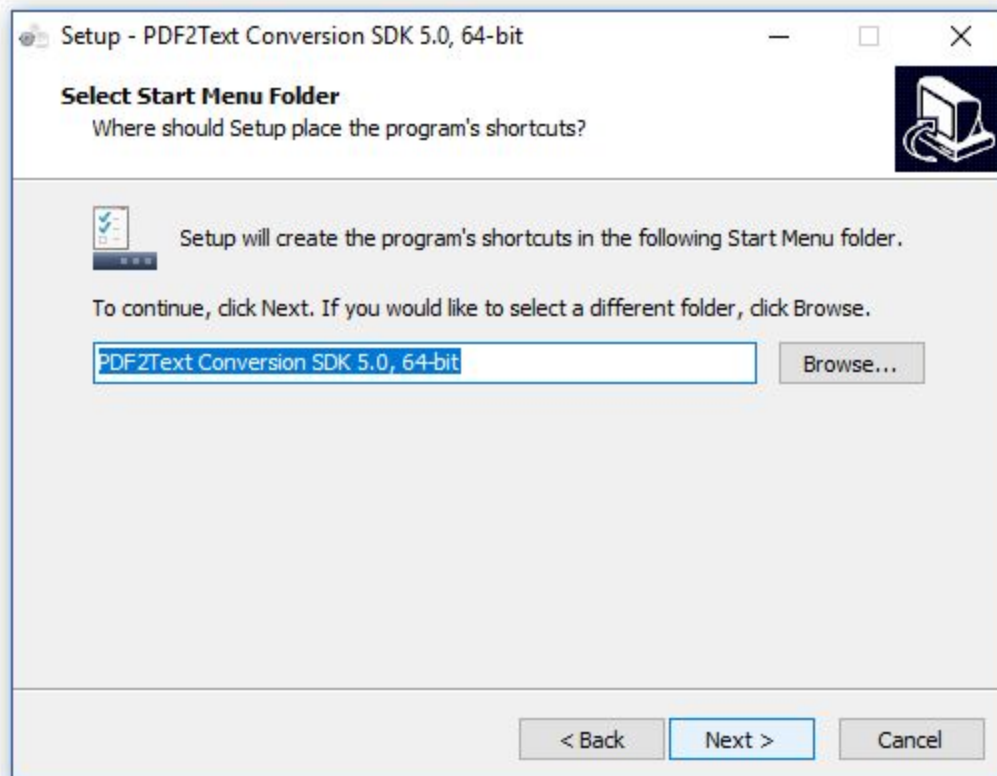
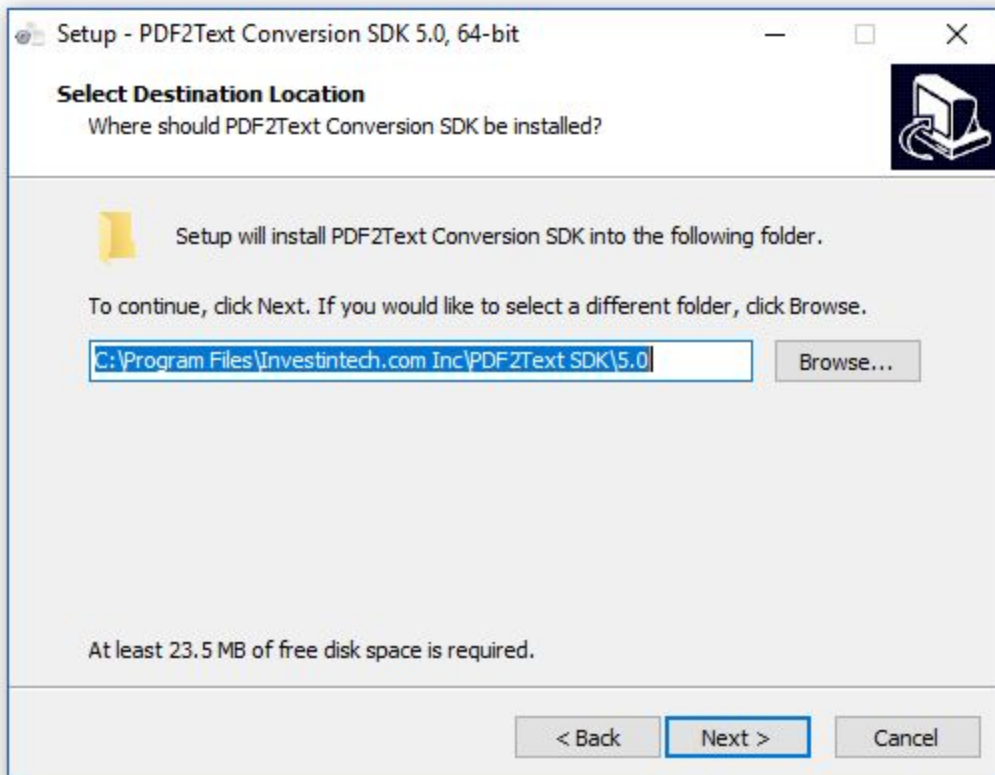
By following the steps below, you can quickly start using the PDF2Text SDK tool of your choice.

After downloading the executable from the Investintech website, you can run the installation by starting the executable. This will run the Installation Wizard and display the Licence Agreement screen:



Please read the Licence Agreement carefully. By choosing "I accept the agreement" and clicking on the Next button, you are accepting the agreement terms for usage of PDF to Text SDK.

In the next two windows, you can choose to change the location of the installation files and Start Menu folder. You are free to leave the default options as displayed in the screenshots below:



After completing this step, you will be presented with the choice of selecting the individual components of the SDK tools you would like installed on your machine.

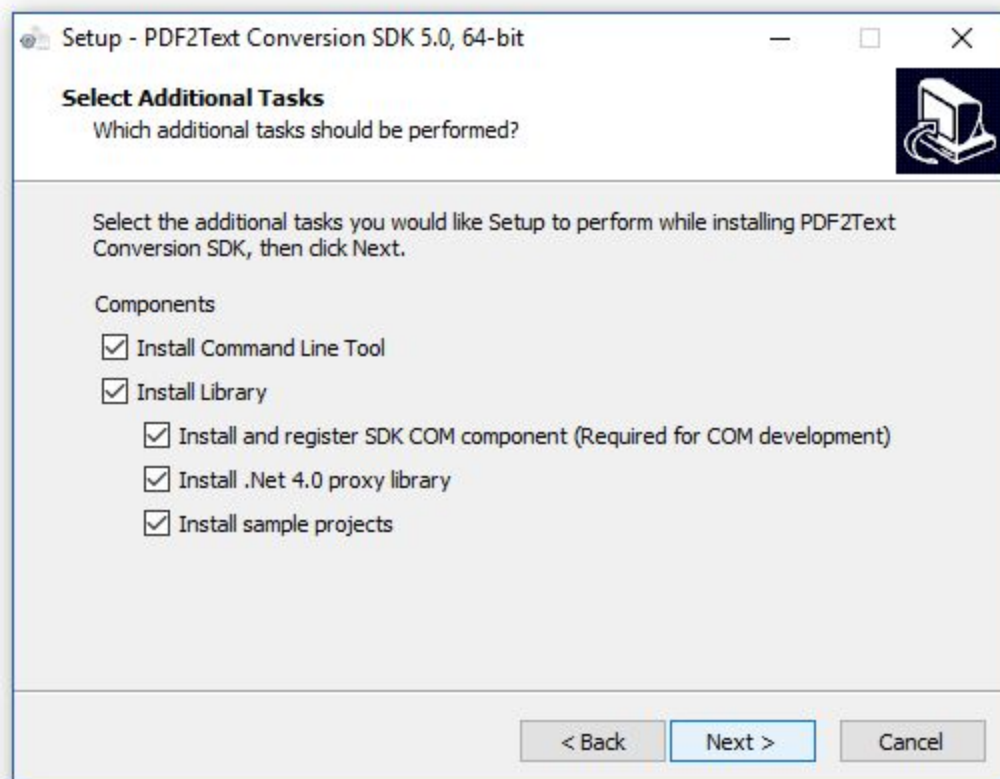
There are two major components of the SDK tools:

- ☐ Command Line Tool (CLT)
- ☐ PDF to Text Library

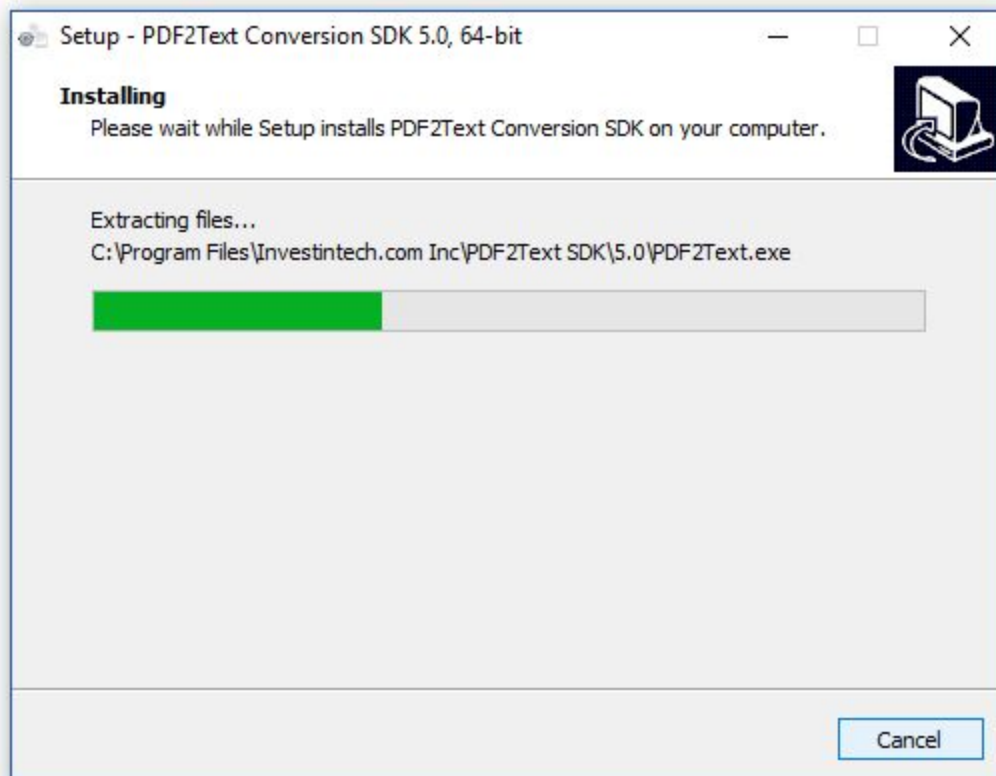
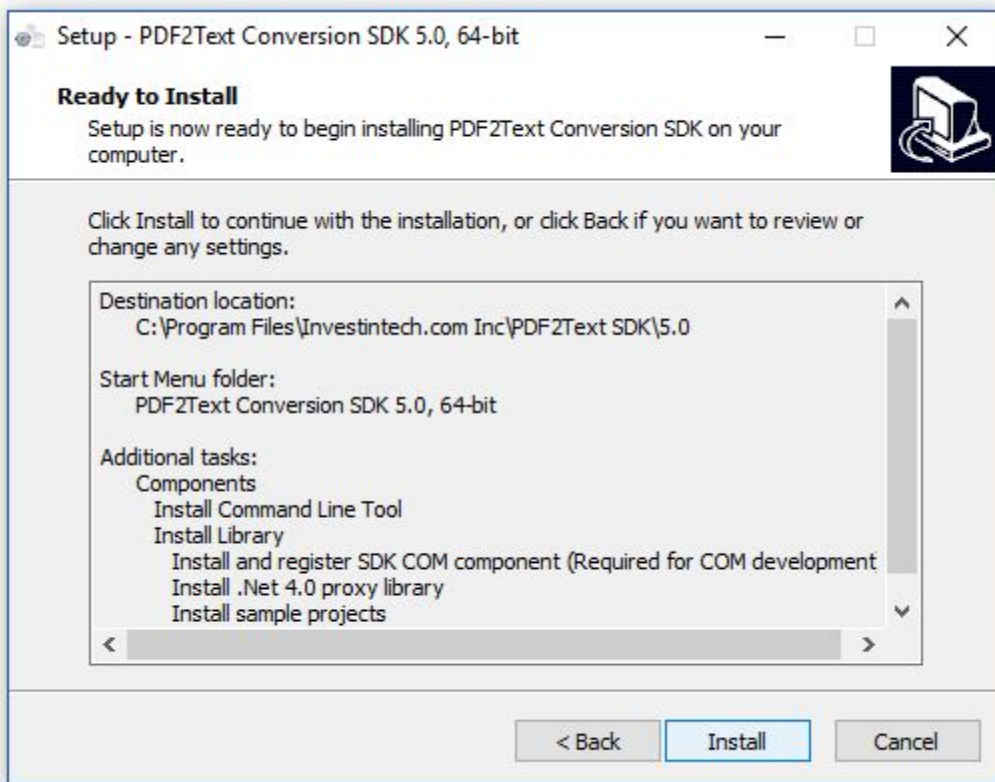
The Command Line tool usage is discussed in detail in Chapter 3 and the PDF to Text Library usage is discussed in Chapter 4.

By leaving the "Install Library" option checked you can also choose which of the individual proxy libraries to include with the installation. The inclusion of these libraries is strongly recommended as they significantly simplify the integration of the SDK into non native C/C++ software.

Our PDF to Text SDK tools also come with Sample Projects for Visual Studio. These sample projects will be copied to the "Documents" directory or more precisely, to the following location on a local machine: **%USERPROFILE%\Documents\PDF2Text Samples**



The last two steps involve a quick review of your chosen installation components and a short process of the installation itself.



After the installation finishes, simply close the installation dialog by clicking on the Finish button.



You are now ready to get started with PDF to Text SDK!

3. Registration

If you decide to purchase our PDF2Text SDK tools, you will have to perform a few more steps in order to be able to use the SDK tools without restrictions. Once you have purchased a licence through our website www.investintech.com or through one of our sales representatives, you will be provided with either a personalized Licence.lic file or a PIN number password for initializing the Libraries or both .

3.1. Using Licence file

To use the personalized License.lic file with either the Command Line Tool or Shared Libraries, simply copy the file in the same directory as the executables (binaries).

Note: When opened with any text editor, the licence file will contain your information. Any changes done to the file will invalidate its usage, causing the tool to fall back into Trial Mode. Thus, it is recommended to keep a copy of this file safe in case of accidental file corruption.

3.2. Using Password to activate licence

For full, unhindered usage of Shared Libraries conversion methods, in addition to the Licence.lic file you need to enter the password provided when purchasing the software.

This password is provided by passing an argument to the special Initialization method which needs to be called first, before any of the conversion methods. Syntax is dependent on the type of Shared Library in question, it can be summarized to the following call:

```
Initialize (String pass);
```

Note: Call to the native C++ function returns `PCSErrorCode` which is an alias of type integer (`int`) . This type is defined in `Lib.h` file.

4. Using the Command Line Tool

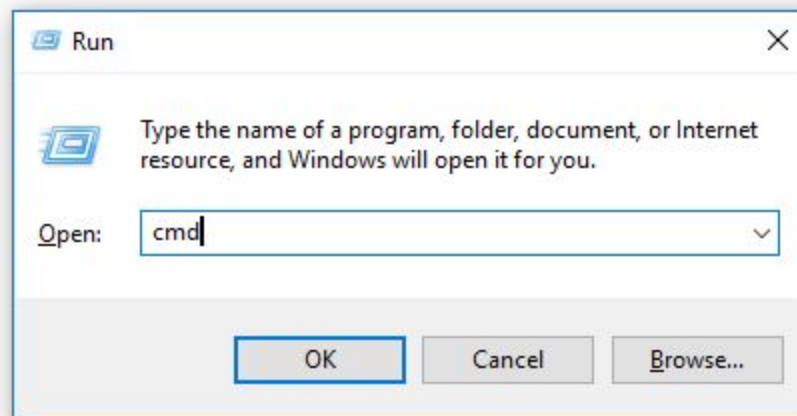
The Command line executable is a powerful, out-of-the-box tool that can be used for easy automation of conversion of PDF files into other file formats.

4.1. Preparation

In order to run the PDF to Text Command Line Tool on Windows OS, you first need to open it from the Command Prompt, Windows Powershell or some other Command Line Interface. This guide will focus on using the Command Prompt.

Open the Windows Command prompt by using any of the following techniques:

- Start typing "*command ...*" string in the Start Search bar , and clicking on the Command Prompt Desktop App when it appears.
- Start "Run" dialog using **Windows + R** keyboard shortcut and entering "cmd" on the keyboard and pressing "Enter" to run the cmd.exe

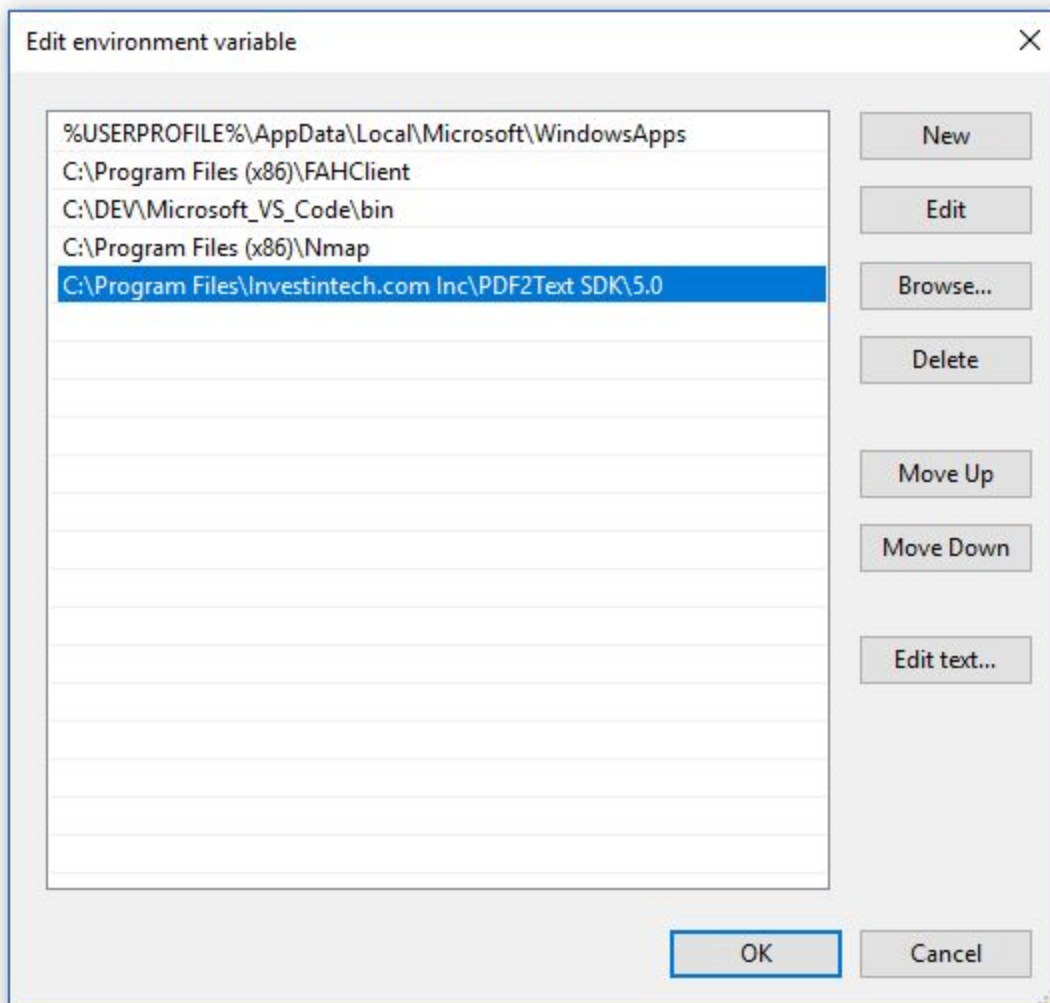


- Using File Explorer to navigate to the SDK installation directory and double-clicking on "Cmd.cmd" file. The installation directory is the one specified during installation.

4.1.1. Adding Path variable (Optional)

If you would like to avoid entering an explicit path location of the PDF2Text Console Application, you can add the location as a Path Environment Variable to your System Properties.

PATH environment variables on Windows are accessible from the Advanced tab in the System Properties window. In the screenshot below, the environment variable Path is referring to the default install location of PDF to Text SDK.



4.2. Quick Single File conversion

Once you have your favorite terminal application running, converting the first file with default properties comes down to running the *PDF2Text.exe* with an absolute file path as your first parameter.

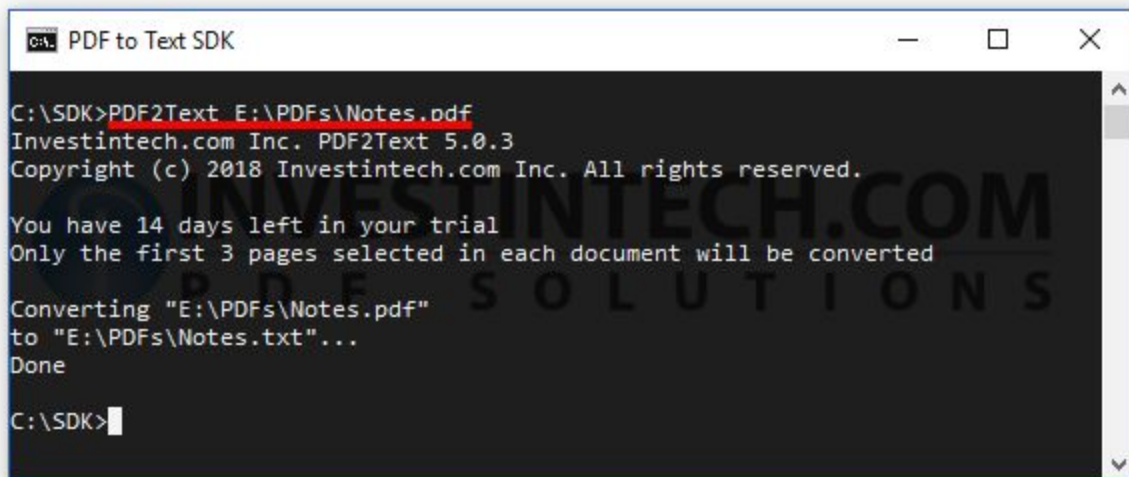
Note: In order for the System to find the correct executable make sure your current location in the Command Prompt is the installation directory of PDF to Text SDK. Optionally you could setup Environment Variables (refer to the [Section 4.1.1](#))

Filenames with spaces should be enclosed with double quotation marks (") as shown in the example that follows.

Example: To convert the file named "Notes.pdf" in the directory "E:\PDFs\" you can issue the following command:

```
>PDF2Text "E:\PDFs\Notes.pdf"
```

In the screenshot below, you can see the expected output when the installation directory path as well as the current command line directory is: C:\SDK\



```
C:\SDK>PDF2Text E:\PDFs\Notes.pdf
Investintech.com Inc. PDF2Text 5.0.3
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Notes.pdf"
to "E:\PDFs\Notes.txt"...
Done

C:\SDK>
```

During the conversion, some basic information of the process of conversion will be shown. The output will also display the location and name of the input and output files. When no additional command line arguments are provided for the output directory, the default output directory location is the same as the input directory, which in this example is "E:\PDFs\"

When the file conversion process is finished, "Done" will be displayed in the command line to mark the end of the conversion process. The Command Prompt will then fall back to its default state.

4.3. Quick Multiple File Conversion

The conversion of multiple files using PDF to Text SDK Command Line tools can be as easy as converting individual files.

The PDF to Text Console Application supports two types of wildcard characters that can be used for converting multiple files with a single command:

- An asterisk "*" character is used to match zero or more characters
- A question mark "?" character is used to match exactly one character

Example: Your goal is to convert all files starting with the string "Sales". For this purpose you could simply append an asterisk wildcard character to the end of a fixed string:

Sales*

Note that this would include all file types, regardless of the extension. If you would like to include only .pdf files to be converted, you could expand your string in the following way:

Sales*.pdf

Example usage of the question mark (?) wildcard character could include matching files containing exactly four characters:

????.pdf

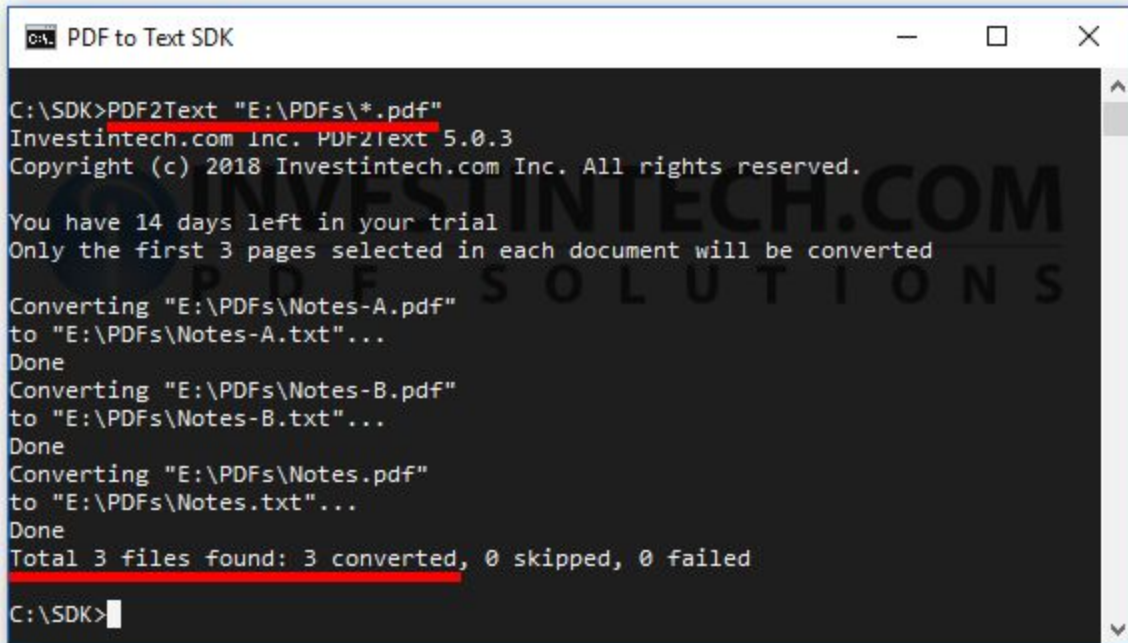
Finally, you could use the combination of both wildcards in order to match .pdf files containing three or more characters, by adapting the following syntax:

???.pdf

Using one or more of the wildcard characters when passing on the first mandatory <inputFilePath> argument can be used to control the conversion of multiple files.

Example: To convert all files with the .pdf extension in the directory "E:\PDFs" you can simply issue the following command:

```
> PDF2Text "E:\PDFs\*.pdf"
```



```
C:\SDK>PDF2Text "E:\PDFs\*.pdf"
Investintech.com Inc. PDF2Text 5.0.3
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Notes-A.pdf"
to "E:\PDFs\Notes-A.txt"...
Done
Converting "E:\PDFs\Notes-B.pdf"
to "E:\PDFs\Notes-B.txt"...
Done
Converting "E:\PDFs\Notes.pdf"
to "E:\PDFs\Notes.txt"...
Done
Total 3 files found: 3 converted, 0 skipped, 0 failed
C:\SDK>
```

The console output is giving you information about individual file conversions, as well as a summary of number of total converted files, skipped files and failed fails (if any).

The console output is giving you information about individual file conversions, as well as a summary of the total number of converted files, skipped files and failed files (if any).

Another applicable example could be the need to convert only the pdf files that start with a "Notes-" string, followed by exactly one other character.

```
> PDF2Text "E:\PDFs\Notes-?.pdf"
```

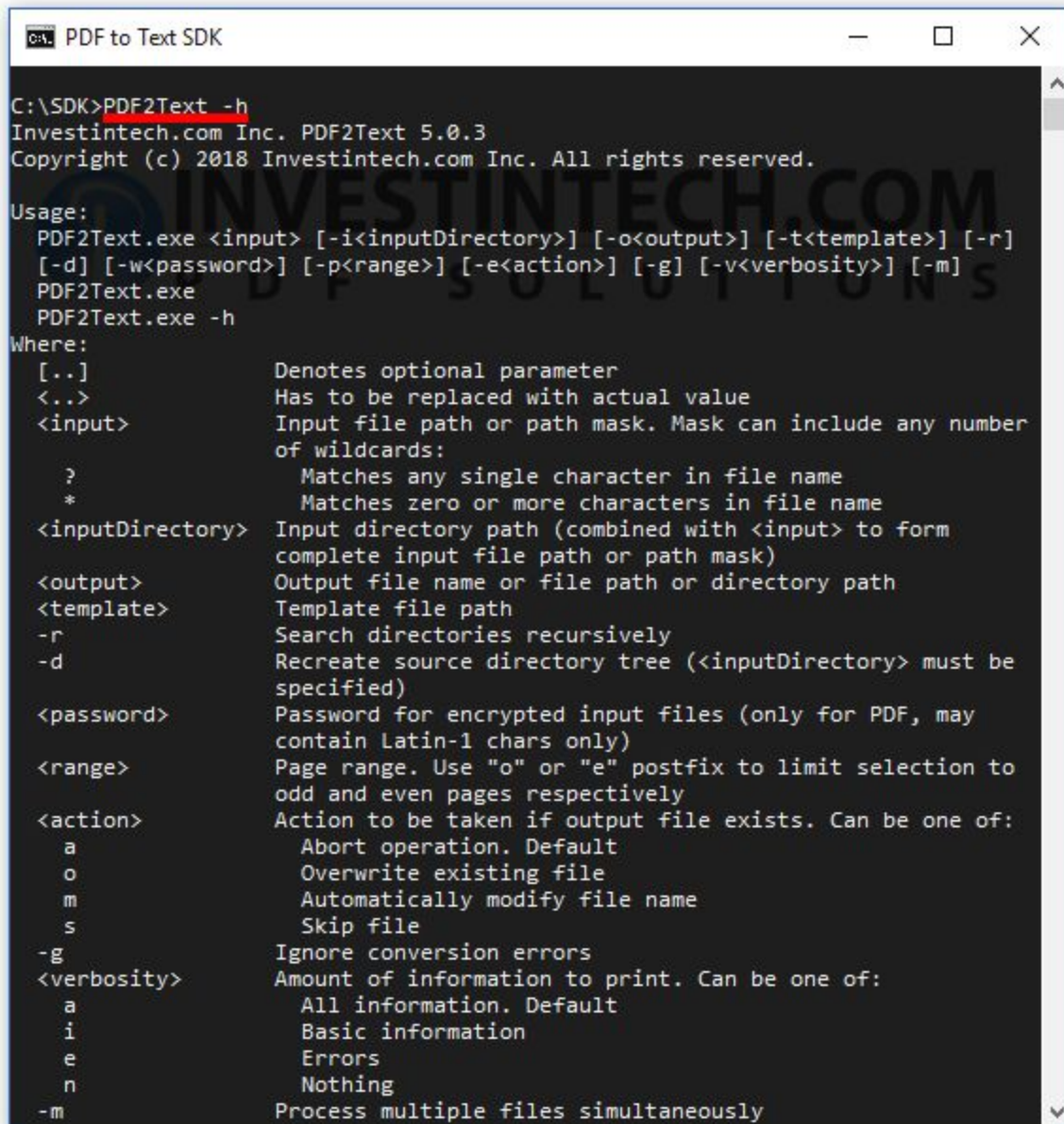
In the following chapters you will get familiar with other methods of converting multiple files using additional Command Line Arguments with the combination of wildcard characters.

4.4. Command line arguments

PDF to Text provides several command line arguments that can help with the conversion automatization fitted to the specific need of the task at hand.

To access the help menu directly from the Command Line interface simply run the PDF2Excel Console Application with the **-h** switch:

> PDF2Text -h

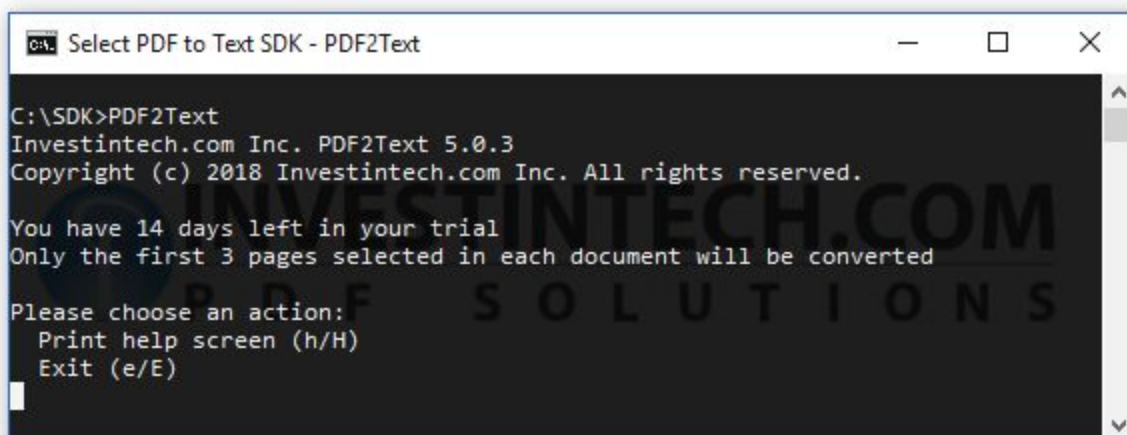


```
C:\SDK>PDF2Text -h
Investintech.com Inc. PDF2Text 5.0.3
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

Usage:
  PDF2Text.exe <input> [-i<inputDirectory>] [-o<output>] [-t<template>] [-r]
  [-d] [-w<password>] [-p<range>] [-e<action>] [-g] [-v<verbosity>] [-m]
  PDF2Text.exe
  PDF2Text.exe -h

Where:
[.]          Denotes optional parameter
<..>        Has to be replaced with actual value
<input>      Input file path or path mask. Mask can include any number
             of wildcards:
             ?    Matches any single character in file name
             *    Matches zero or more characters in file name
<inputDirectory> Input directory path (combined with <input> to form
             complete input file path or path mask)
<output>      Output file name or file path or directory path
<template>    Template file path
-r           Search directories recursively
-d           Recreate source directory tree (<inputDirectory> must be
             specified)
<password>    Password for encrypted input files (only for PDF, may
             contain Latin-1 chars only)
<range>       Page range. Use "o" or "e" postfix to limit selection to
             odd and even pages respectively
<action>      Action to be taken if output file exists. Can be one of:
             a    Abort operation. Default
             o    Overwrite existing file
             m    Automatically modify file name
             s    Skip file
-g           Ignore conversion errors
<verbosity>  Amount of information to print. Can be one of:
             a    All information. Default
             i    Basic information
             e    Errors
             n    Nothing
-m           Process multiple files simultaneously
```

It is also possible to display help text from a simple interactive menu by running PDF2Text.exe without any arguments, and then pressing "h" or "H".



The following table contains the summary for the usage of the switches:

Switch	Parameter description	Usage
-i	<dirPath>	Specify the input directory
-o	<dirPath>	Specify the output directory
-t	<pcvtTemplate>	Specify the conversion template
-r	NONE	Recursive search of directories
-d	NONE	Recreate Source directory tree
-w	<password>	Specify the password(s) for encrypted PDF files
-p	<range>	Define page range
-e	<action>	Action to be taken if output file exists
-g	NONE	Ignore Conversion Errors
-v	<verbosity>	Amount of information to print.
-m	NONE	Process multiple files simultaneously
-h	NONE	Display help screen

The switch parameters that are not required are marked by NONE in above table. All other parameters are mandatory and shortly described in the above table. The function of the specific switches will be discussed in more detail in the following chapters.

Note: When passing a parameter to a switch it must be entered immediately after the switch, that is, without spaces. It is allowed, however, to pass a parameter enclosed in double quotation marks for more readable code.

4.4.1. Specifying input directory

The Command line tool allows you to easily convert multiple files from a specified directory.

Switch **-i** is used to specify your source directory, after the mandatory `<inputFilePath>` argument.

When a directory is specified, the usage of wildcard characters is mandatory. This is required in order to match multiple files from the specified directory.

If you would like to convert all of the files in directory "E:\PDFs", you could use the following command:

```
>PDF2Text * -i"E:\PDFs"
```

The first argument passed is only one character. This is a simple "catch all" filter `*` which will match all files in the specified directory.

Caution: When specifying a filter that is too broad, like using the catch-all asterisk sign alone, the conversion might stop when it reaches an unsupported file for conversion. In order to make sure that the conversion of the file is successful, you can do one of the following: specify the appropriate file extension or use the **-g** switch to force the engine to ignore conversion errors.

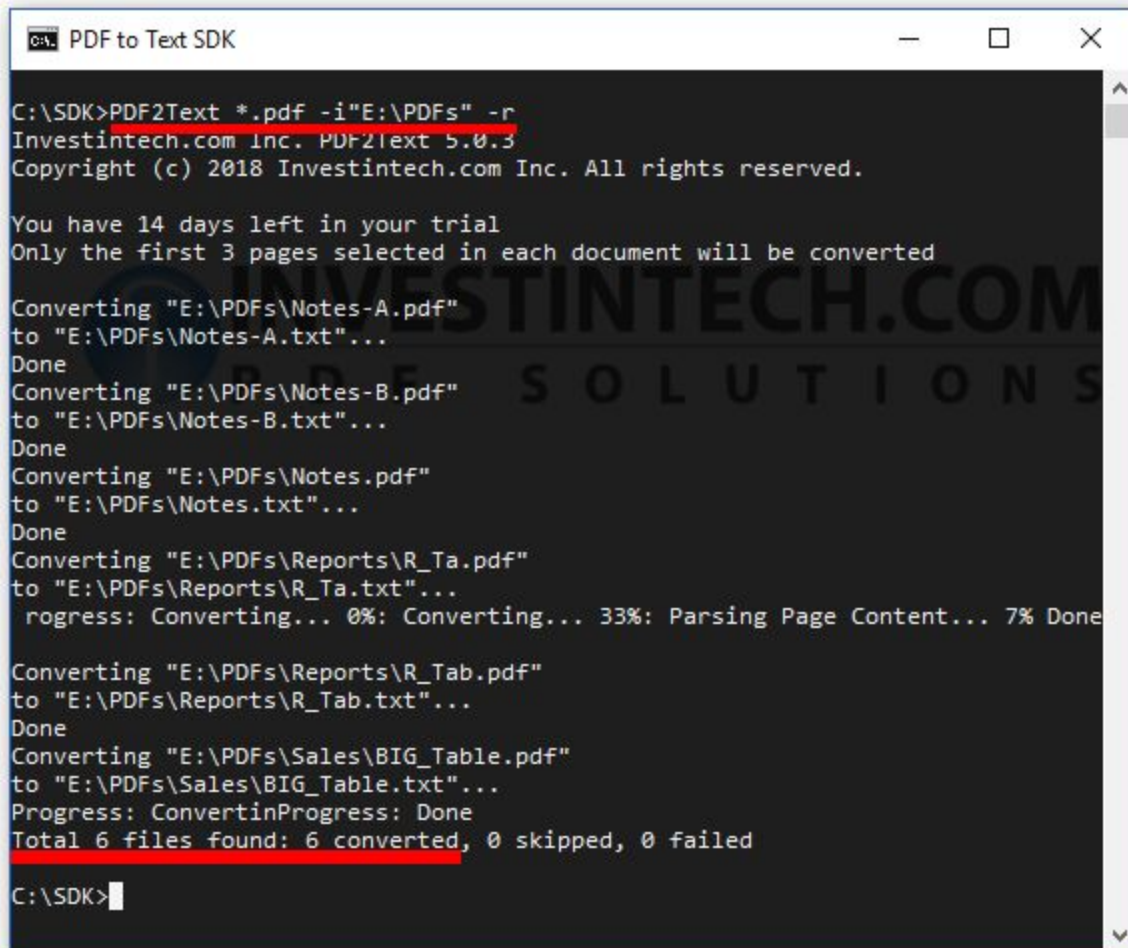
It is quite common that there are nested directories inside the specified input directory. In this case, if there are also files inside those nested directories that need to be included in the conversion output, you can use the **-r** switch to instruct the engine to perform a recursive search of files matching the pattern.

Example: If you have "E:\PDFs" as a directory which contains some other sub-directories, that in turn also contain files of interest, you can use the following command to easily traverse the file tree hierarchy:

```
>PDF2Text *.pdf -i"E:\PDFs" -r
```

The screenshot below shows the conversion result of the following tree hierarchy of the "E:\PDFs" directory that contains two directories named "Sales" and "Reports".

Two nested folders, "\Sales" and "\Reports", also contain PDF files, that is, files that are matching the specified mask: "***.pdf**".



```
C:\SDK>PDF2Text *.pdf -i"E:\PDFs" -r
Investintech.com Inc. PDF2Text 5.0.3
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Notes-A.pdf"
to "E:\PDFs\Notes-A.txt"...
Done
Converting "E:\PDFs\Notes-B.pdf"
to "E:\PDFs\Notes-B.txt"...
Done
Converting "E:\PDFs\Notes.pdf"
to "E:\PDFs\Notes.txt"...
Done
Converting "E:\PDFs\Reports\R-Ta.pdf"
to "E:\PDFs\Reports\R-Ta.txt"...
Progress: Converting... 0%: Converting... 33%: Parsing Page Content... 7% Done
Converting "E:\PDFs\Reports\R-Tab.pdf"
to "E:\PDFs\Reports\R-Tab.txt"...
Done
Converting "E:\PDFs\Sales\BIG_Table.pdf"
to "E:\PDFs\Sales\BIG_Table.txt"...
Progress: Converting... Done
Total 6 files found: 6 converted, 0 skipped, 0 failed

C:\SDK>
```

The PDF2Text Command Line Tool displays the individual absolute paths for each file that is being converted.

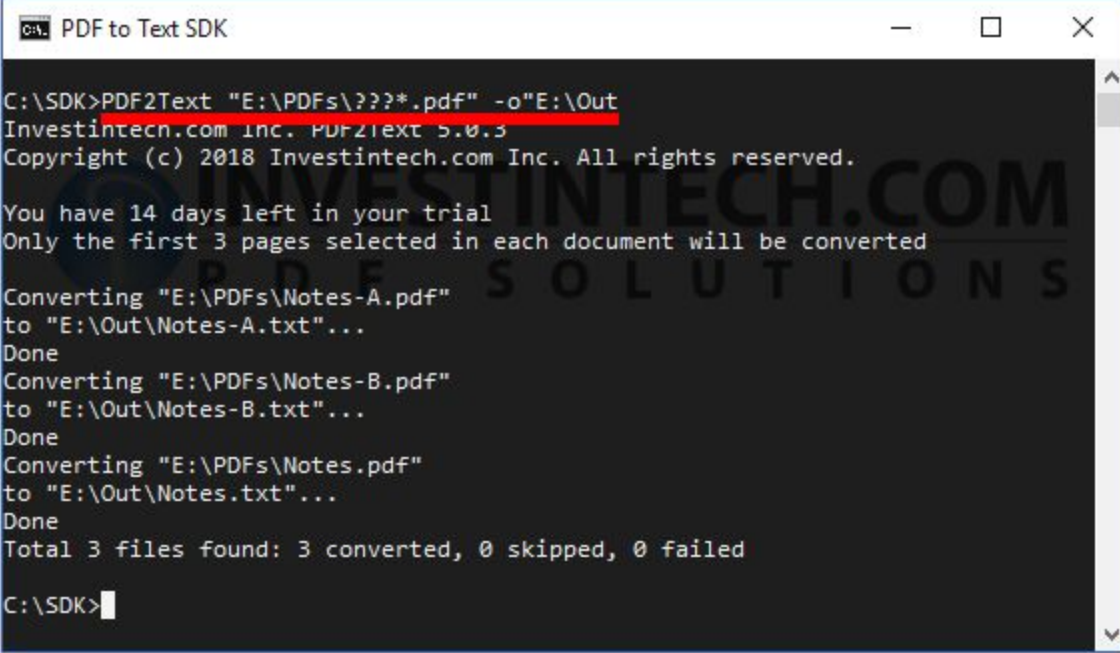
4.4.2. Specifying output directory

In addition to specifying the input folder, it is often convenient to specify the output directory to collect converted files in a different directory. By default, the converted files are placed in the same directory as the source files, which can quickly become inconvenient as the number of source file grows larger.

To save converted files in a location other than the source document directory, you can specify an absolute path of the output directory after the **-o** switch.

Example: In order to store converted files to "E:\Out" directory, simply declare the desired path after the **-o** switch, using similar syntax as in the following line:

```
>PDF2Text "E:\PDFs\???*.pdf" -o"E:\Out"
```



```
C:\SDK>PDF2Text "E:\PDFs\???*.pdf" -o"E:\Out"
Investintech.com Inc. PDF2Text 5.0.5
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Notes-A.pdf"
to "E:\Out\Notes-A.txt"...
Done
Converting "E:\PDFs\Notes-B.pdf"
to "E:\Out\Notes-B.txt"...
Done
Converting "E:\PDFs\Notes.pdf"
to "E:\Out\Notes.txt"...
Done
Total 3 files found: 3 converted, 0 skipped, 0 failed
C:\SDK>
```

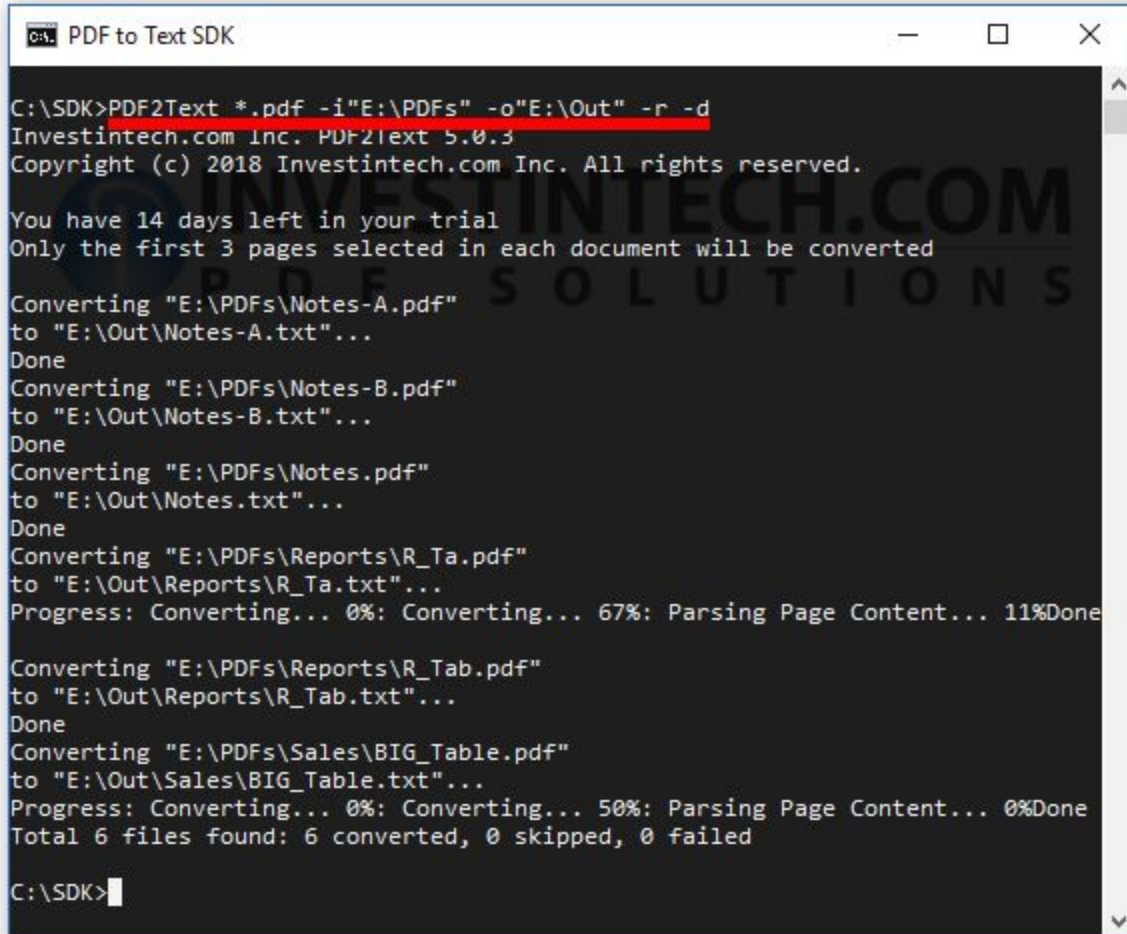
In the previous example the filter was specified for a file name using wildcards " **???*.pdf** " to match only files with the extension PDF and having at least 3 characters in the filename.

When using both <input> and <output> file paths together with the **-r** switch ([Section 4.4.1](#)) it is possible to recreate the directory tree in the output directory using the **-d** switch.

By following the simple directory tree from [Section 4.4.1](#) where we have directory "E:\PDF Files" containing two sub-directories, "\Sales" and "\Reports", we can see how the **-d** switch can be implemented.

Example: In the following example you can see how the **-d** switch is producing a conversion of all files from all three directories, while re-creating the directory tree in the output directory:


```
>PDF2Text *.pdf -i"E:\PDFs" -o"E:\Out" -r -d
```



```
C:\SDK>PDF2Text *.pdf -i"E:\PDFs" -o"E:\Out" -r -d
Investintech.com Inc. PDF2Text 5.0.3
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Notes-A.pdf"
to "E:\Out\Notes-A.txt"...
Done
Converting "E:\PDFs\Notes-B.pdf"
to "E:\Out\Notes-B.txt"...
Done
Converting "E:\PDFs\Notes.pdf"
to "E:\Out\Notes.txt"...
Done
Converting "E:\PDFs\Reports\R-Ta.pdf"
to "E:\Out\Reports\R-Ta.txt"...
Progress: Converting... 0%: Converting... 67%: Parsing Page Content... 11%Done
Converting "E:\PDFs\Reports\R-Tab.pdf"
to "E:\Out\Reports\R-Tab.txt"...
Done
Converting "E:\PDFs\Sales\BIG_Table.pdf"
to "E:\Out\Sales\BIG_Table.txt"...
Progress: Converting... 0%: Converting... 50%: Parsing Page Content... 0%Done
Total 6 files found: 6 converted, 0 skipped, 0 failed

C:\SDK>
```

Note: When using this **-d** switch to recreate a directory tree, only child-directories that contain files with names that match the filter specified in the first <inputFilePath> argument will be created. This is because the **-r** switch will ignore directories with no matching file mask.

Following the previous example, if directory "E:\PDFs" contained a folder named "Images" which contains only .png files, the "Images" folder would not be created in the "E:\Out" directory, since the provided pattern, *.pdf, only matches pdf files.

4.4.3. Page ranges

When dealing with large PDF files there is often no need to convert all of the pages in the document. Using Page Range switch, you can define which part of the document to convert.

With the Page Range switch `-p`, you can define a single page or a subset of pages to be converted. When defining a subset of pages, you can specify a start page number (`<startPageN°>`), end page number (`<endPageN°>`) or both. When specifying only starting page, followed by a hyphen (`-`), the assumed end page is the end of the document. Similarly, when specifying only last page number, the assumed starting page is the first page of the document.

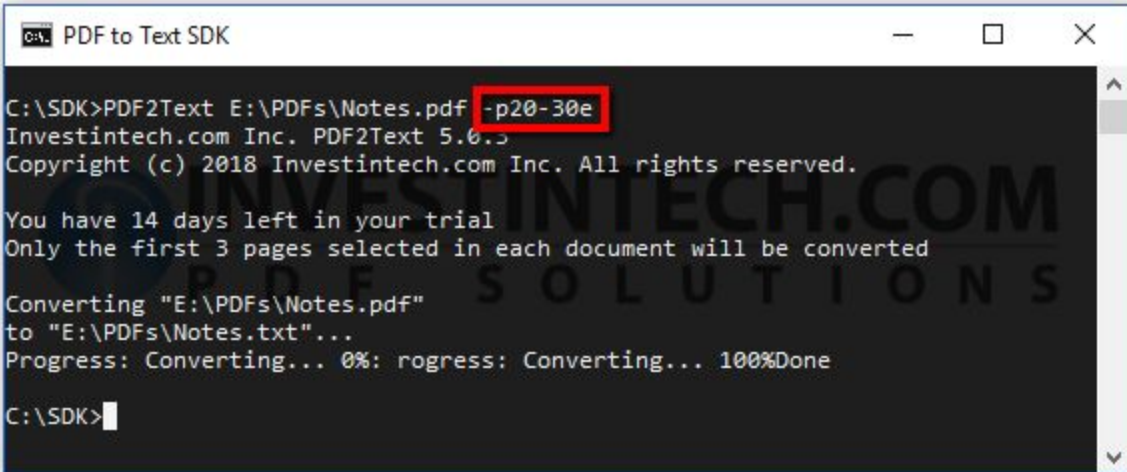
Additionally, you can decide to extract only odd or even pages of the document by using suffixes, `o` or `e` respectively. The table below provides you with an overview of the possible syntax for each case and what it does:

Syntax	What does it do?
<code>-p<SinglePageN°></code>	Convert one page of the document.
<code>-p<startPageN°>-<endPageN°></code>	Convert all pages starting with <code><startPageN°></code> and ending with <code><endPageN°></code>
<code>-p<startPageN°>-<endPageN°>e</code>	Convert even pages starting with <code><startPageN°></code> and ending with <code><endPageN°></code>
<code>-p<startPageN°>-<endPageN°>o</code>	Convert odd pages starting with <code><starPageN°></code> and ending with <code><endPageN°></code>

Note: Odd and even pages are "absolute", meaning that they are dependent on the real document page numbering, and not on `<starPageN°>`.

Example: Let's say you have a large PDF file, but you only require even pages from page 20 to page 30. You can use the following command to extract specific pages:

```
>PDF2Text E:\PDFs\Notes.pdf -p20-30e
```



```
C:\SDK>PDF2Text E:\PDFs\Notes.pdf -p20-30e
Investintech.com Inc. PDF2Text 5.0.5
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Notes.pdf"
to "E:\PDFs\Notes.txt"...
Progress: Converting... 0%: rogress: Converting... 100%Done
C:\SDK>
```

4.4.4. Verbosity level of Console output

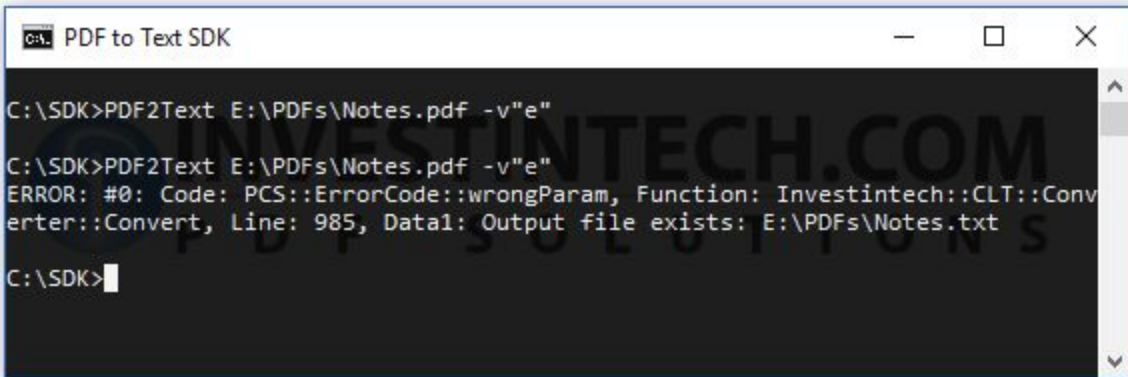
Using the Command Line Tool is often done in a fully automated manner. When this is the case, it is often unnecessary or even counterproductive to have all of the conversion outputs to be printed to the command line interface.

You can control the amount of output using the `-v` switch followed by one of the options. This is described in the table below ordered by the amount of output that will be displayed:

-v Switch parameter	What shall be printed out?
a	All of the information (default)
i	Only basic information
e	Only errors, if they occur
n	Nothing

Example: Converting the same file two times with verbosity level reduced only to errors. The output of repeated conversion of the file "Notes.pdf" is shown below:


```
>PDF2Text E:\PDFs\Notes.pdf -v"e"
```



```
C:\SDK>PDF2Text E:\PDFs\Notes.pdf -v"e"
C:\SDK>PDF2Text E:\PDFs\Notes.pdf -v"e"
ERROR: #0: Code: PCS::ErrorCode::wrongParam, Function: Investintech::CLT::Converter::Convert, Line: 985, Data1: Output file exists: E:\PDFs\Notes.txt
C:\SDK>
```

You can see from the screenshot that the first run of the command didn't produce any output at all, since no error occurred, while for the second conversion we have an error, and it is the only output displayed.

Important: Verbosity mode can have a very big impact on the performance of the Command Line Tool. Conversion time can be doubled when complete verbosity level is set to all, which is the default. When the tool is invoked by another app or script it is strongly recommended to turn off the verbosity level completely. Otherwise, you should generally use the scarcest verbosity level you require in order to get the fastest conversions.

4.4.5. Converting password protected files

PDF files are sometimes encrypted, and thus cannot be opened without providing a password. In order to be able to convert these files you can provide the password as parameter to the **-w** switch.

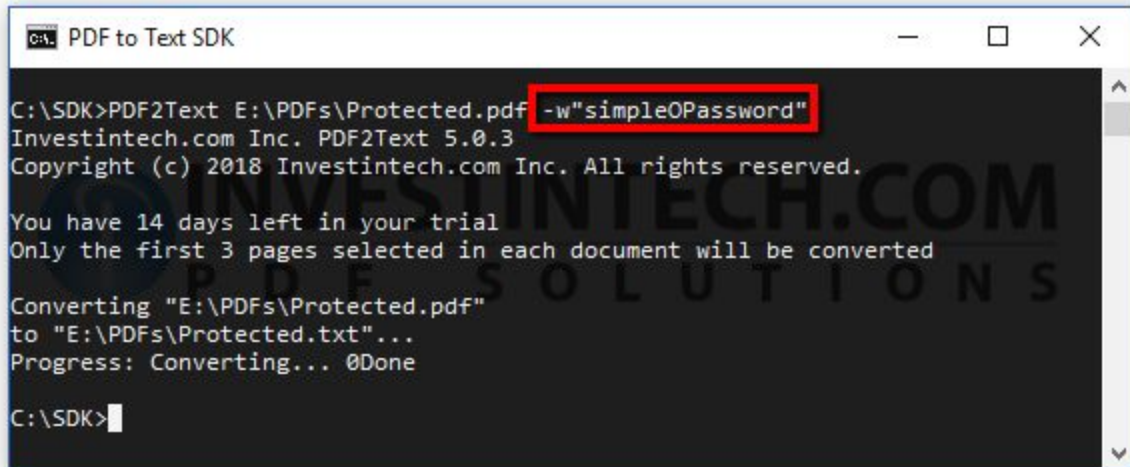
If you try to convert an encrypted file, you will get the following error:

```
ERROR: #0: Code: PDL::ERR_PDF_WRONG_PASSWORD
```

Since the conversion engine only requires reading of the document in order to convert it to another file format, you can supply either an "Owner" or "User" password to successfully convert the file.

Example: Here is a simple example of how to convert an encrypted file with an absolute file path "E:\PDFs\Protected.pdf", and password "simpleOPassword":

```
>PDF2Text E:\PDFs\Protected.pdf -w"simpleOPassword"
```



```
C:\SDK>PDF2Text E:\PDFs\Protected.pdf -w"simpleOPassword"
Investintech.com Inc. PDF2Text 5.0.3
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Protected.pdf"
to "E:\PDFs\Protected.txt"...
Progress: Converting... 0Done

C:\SDK>
```

4.4.6. Ignoring Errors

Having an error reporting feature is an important part of every application. There are some cases when the conversion process needs to proceed despite the errors, especially when large scale file conversions are in question.

The correct option to use in this case is the **-g** switch that instructs the PDF to Excel Command Line Tool to ignore conversion errors and continue the execution of the program.

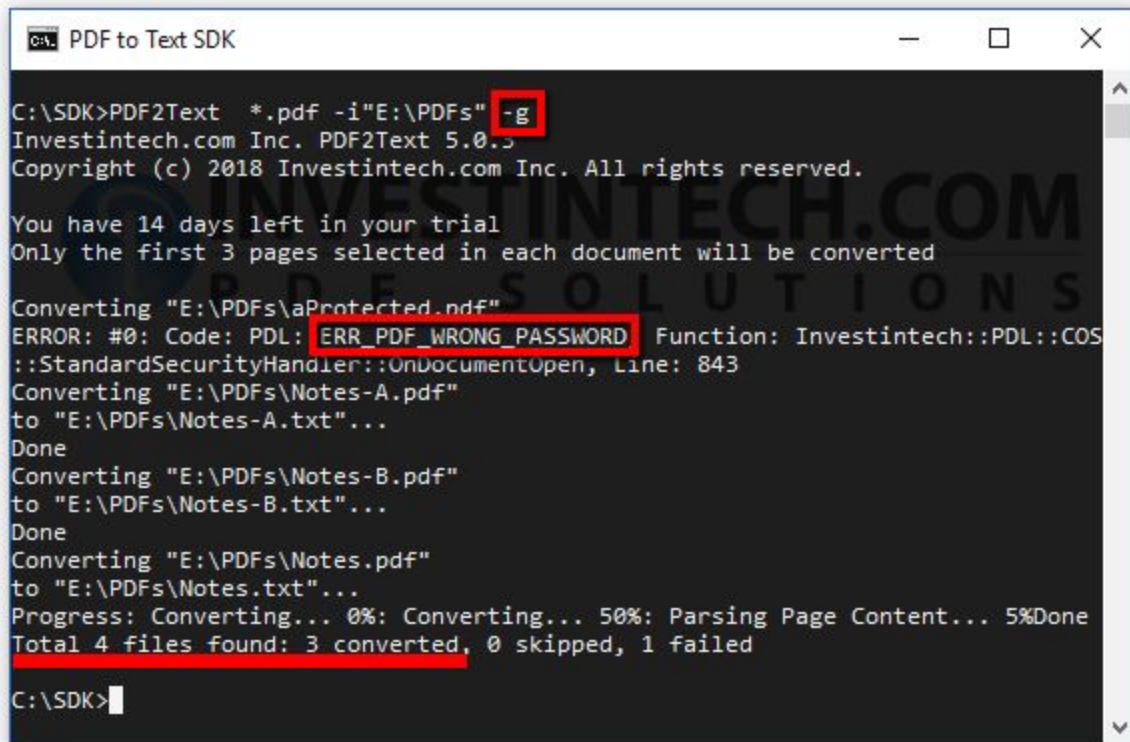
When converting multiple files using wildcard characters, it is possible that one of the files cannot be converted. When this happens, the process of conversion will stop and all the other files will not be converted.

Example: You have a directory named "E:\PDFs\" with several .pdf files, one of which is password protected. You can order the conversion of all .pdf files in the directory with the following command:

```
>PDF2Text *.pdf -i"E:\PDFs"
```

If conversion fails when it reaches the password protected file, then all the other files will not be converted. To avoid this from happening you can add the **-g** switch to the command:

```
>PDF2Text *.pdf -i"E:\PDFs" -g
```



```
C:\SDK>PDF2Text *.pdf -i"E:\PDFs" -g
Investintech.com Inc. PDF2Text 5.0.3
Copyright (c) 2018 Investintech.com Inc. All rights reserved.

You have 14 days left in your trial
Only the first 3 pages selected in each document will be converted

Converting "E:\PDFs\Protected.pdf"
ERROR: #0: Code: PDL: ERR_PDF_WRONG_PASSWORD Function: Investintech::PDL::COS
::StandardSecurityHandler::OnDocumentOpen, Line: 843
Converting "E:\PDFs\Notes-A.pdf"
to "E:\PDFs\Notes-A.txt"...
Done
Converting "E:\PDFs\Notes-B.pdf"
to "E:\PDFs\Notes-B.txt"...
Done
Converting "E:\PDFs\Notes.pdf"
to "E:\PDFs\Notes.txt"...
Progress: Converting... 0%: Converting... 50%: Parsing Page Content... 5%Done
Total 4 files found: 3 converted, 0 skipped, 1 failed
C:\SDK>
```

From the output in the console, you can see that Error has occurred when the password protected file was reached. Nevertheless, the conversion of all other files was completed successfully.

4.4.7. Parallel Conversion

One more option that can help you in the process of converting a large number of files is using the **-m** switch which forces the engine to convert multiple files simultaneously.

Since the immediate output of a big number of files can be overwhelming, this command is intended to be used together with the verbosity level (section 3.4.7) set to "none" using **-vn** (or **-v"n"**).

To simultaneously convert all of the PDF files in a directory named "E:\PDFs", use the following command:

```
>PDF2Text E:\PDFs\*.pdf -m -vn
```



Of course, due to verbosity level being set to "none", there will be no output on command line interface.

5. Using the Shared Library

The PDF to Text SDK comes with a complete set of files which allow for quick integration into your development environment. PDF2Text.dll is collection of methods compiled, linked and stored in a Dynamic Link Library. You can use the Dynamic Link Library PDF2Text.dll directly, or through one of the proxy libraries for COM and .NET for non C/C++ projects.

5.1. Preparation

After installing the PDF to Text SDK tool on your local machine you should have access to Sample Files for Visual studio in addition to libraries and executive binaries. These files can be accessed in your local "Documents" directory, the install directory relative to the User folder on your local machine, or by using the out-of the box environment variable:

%USERPROFILE%\Documents\PDF2Text Samples

The stated folder above contains subfolders with different Visual Studio sample projects.

Note: While instructions for using sample projects can be implemented using the default location in which they are stored, it is recommended to make a copy of this sample project to some other location so you can easily rollback to a default state.

These instructions will assume you have Visual Studio 2017 installed on the machine you intend to use for integrating the PDF2Text Shared Library.

5.1.1. Licence Registration

In order to use a fully licenced version of PDF2Text.dll in your application you will need two distinct components:

1. Personalised Licence.lic file
2. Password corresponding to the Licence file

To use the licenced application, the `Licence.lic` file needs to be present in the same directory as your executables. If you have a licence file you must pass an argument to the initialization function in PDF2Text class (C++):

PCSErrorCode PDF2TextInitialize(const char* pass)

If you do not have a Licence.lic file, you do not have to provide a password String, however, this will give you limited trial access to our SDK tools. In order to use the unlicensed SDKs, you can pass an empty string to the initialization function.

Note: Return type of PDF2ExcelInitialize() is PCSErrorCode. This is an alias of type integer (int) which is defined in the Lib.h file.

5.2. Lib.c sample project

This section will explain how to connect the Visual C++ sample project with PDF2Text.dll library.

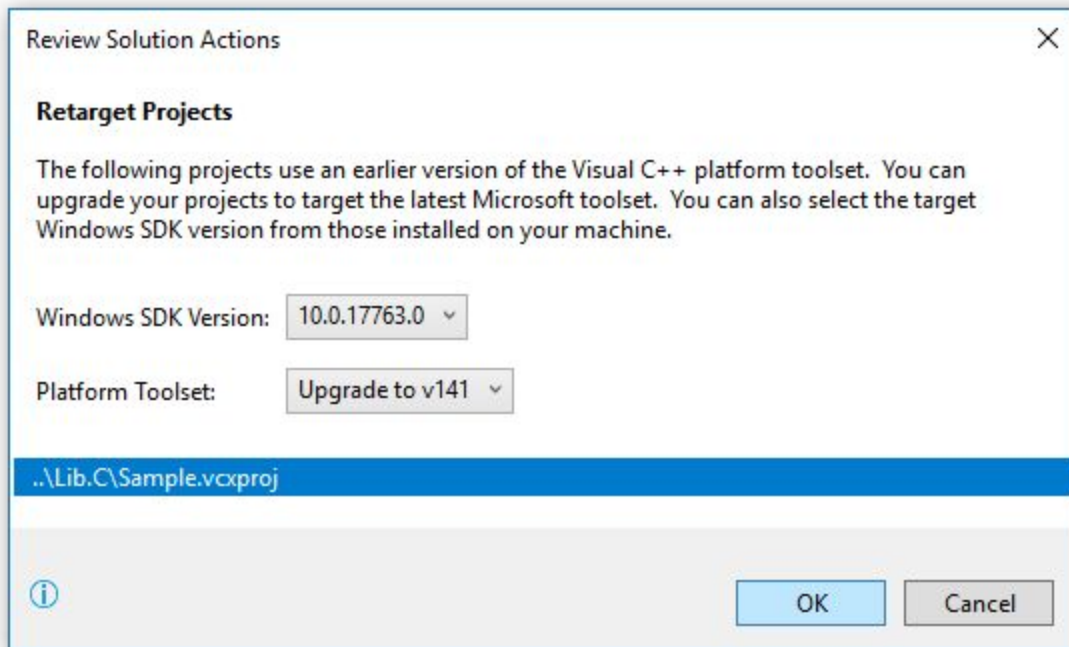
This sample project is placed by default in the %USERPROFILE%\Documents\PDF2Text\Samples\Lib.C directory.

Important: To make the process of resolving dependencies easier, it is best to copy all the files from the PDF2Text SDK install directory to the Lib.C directory.

Once you have copied the required files open Sample.sln either by directly clicking on the file itself or by opening the solution from Visual Studio by using File -> Open/Project Solution.

During the opening of the Solution, you will most likely be asked to retarget the Visual C++ platform toolset. It is recommended to accept this project upgrade and leave the default options recommended by Visual Studio.

The following screenshot displays this type of window for retargeting Projects.



Note: The actual Windows SDK Version and platform Toolset may depend on your local environment, and thus may differ somewhat from the screenshot above.

If the upgrade is successful, you should see several messages in the Output Window, one for every build configuration. The messages should be formatted in a manner similar to the following:

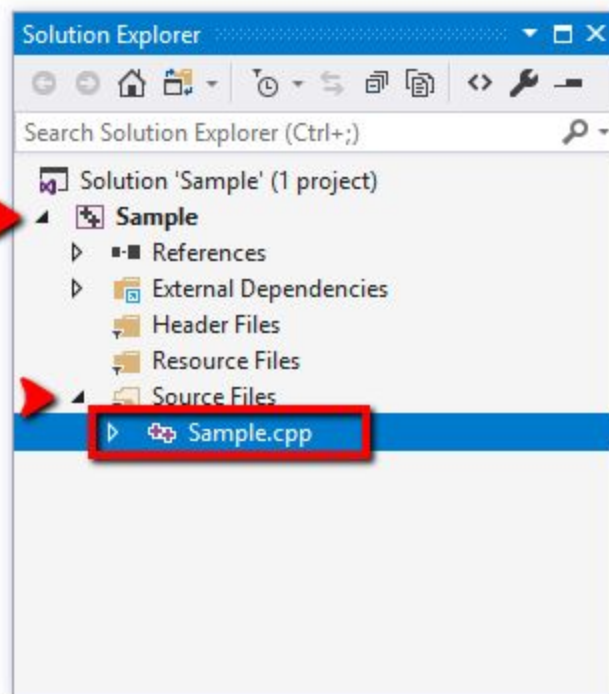
changing Platform Toolset to 'v141' (was 'v100')

If the Output Window is not visible you can open it from the Menu: View -> Output.

Visual Studio will not initially open any files. To open the source file of a Sample Project for editing in Visual Studio, look for the Solution Explorer, by default it is located on the right hand side. If the Solution Explorer window is not visible, you can open it from the Toolbar, by choosing View -> Solution explorer. Alternatively you can use the keyboard shortcut, CTRL + W, S.

Next, you might need to expand the "Sample" project and Sample files, if not already expanded. Then open Sample.cpp file in the editor by double-clicking, or using the right-click context menu and choosing Open.

You can refer to the following screenshot for quick reference:



5.2.1. Resolving Dependencies

Even if all of the necessary files from the PDF2Excel SDK are present in the same directory as Sample.sln file, some dependency issues might occur.

If the header file PDF2Excel.h is not resolved when the solution is opened, you might need to make a simple correction on how the preprocessor searches for the header. This file is included in the sample project in the following way:

```
#include <PDF2Text.h>
```

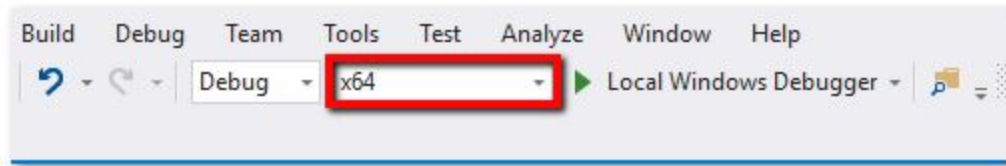
Simply changing the square brackets to parentheses will force the preprocessor to first search in the folder where the source file is located. Thus, you can replace the above line with the following:

```
#include "PDF2Text.h"
```

If this relative path cannot be resolved, it might be necessary to provide the absolute path of the PDF2Excel header file. In order to do this, navigate and note down the relative location of this header file. Add the absolute path instead of the relative one:


```
#include <C:\SDK\PDF2Text.h>
```

Before building the project, you should check if the correct CPU architecture is selected in Configuration manager. This option is usually quickly accessible directly from the toolbar, left of the "Run" button, as presented in the following screenshot.



Use the configuration that matches the target platform of the PDF2Text SDK tools.

5.2.2. Running the sample project

When all of the dependencies are resolved, and you can successfully build the Solution, running the Solution without any parameters set will fail with exit code 1.

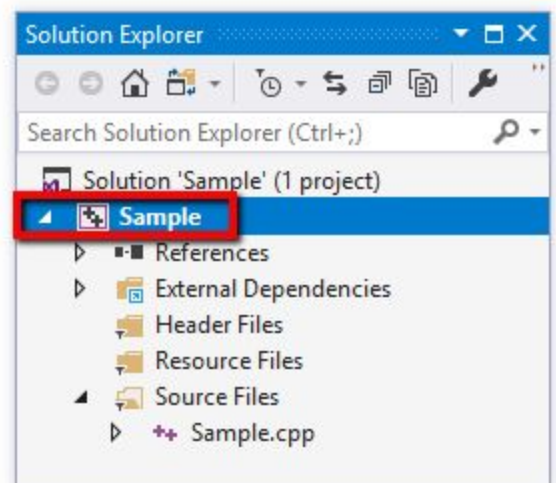
The out-of-the-box Sample.cpp has a check for the number of parameters passed as command line argument. For the conversion to be successful, you need to pass at least three arguments to the application: Input file path (PDF) , output directory and output file name.

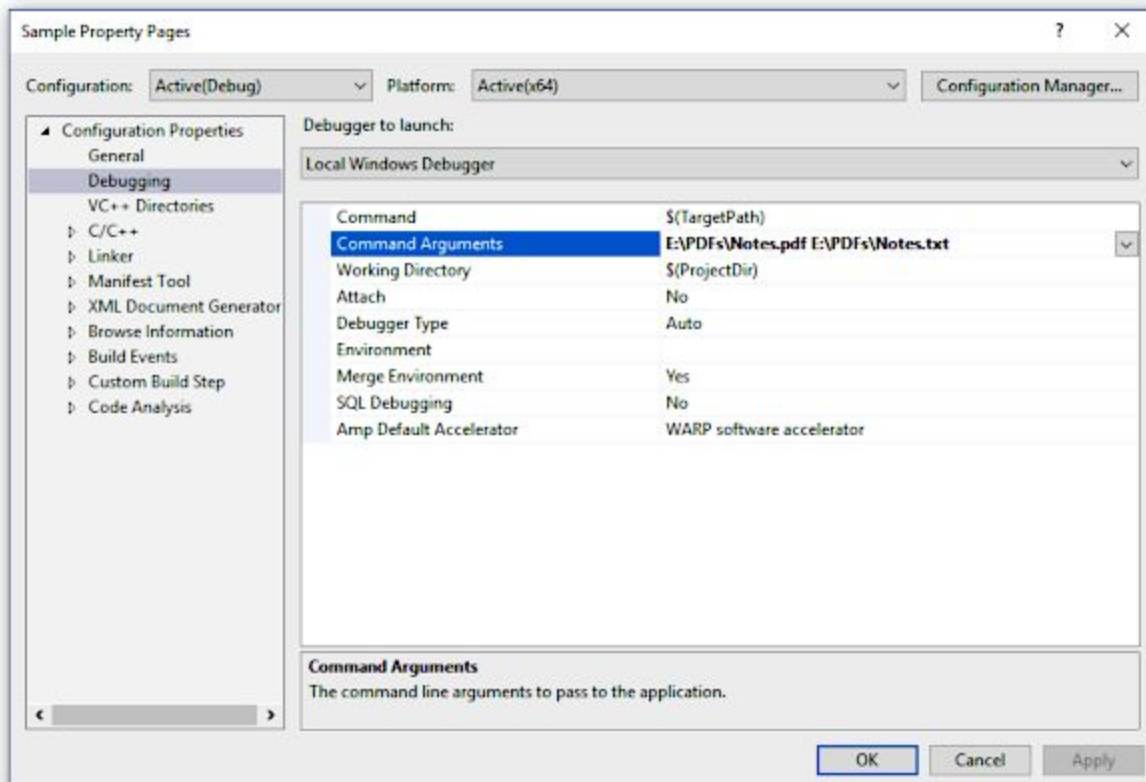
To pass on parameters you can use one of the following two approaches:

1. Directly run the ... \Debug\Sample.exe executable from Command Line, which Visual Studio built for you, and pass the required parameters separated by "white space".

2. Open the Sample Project Property page by right clicking on Sample project from the Solution Explorer window and choosing Properties from the context menu. You must select Project properties, not Solution properties, as marked in the screenshot.

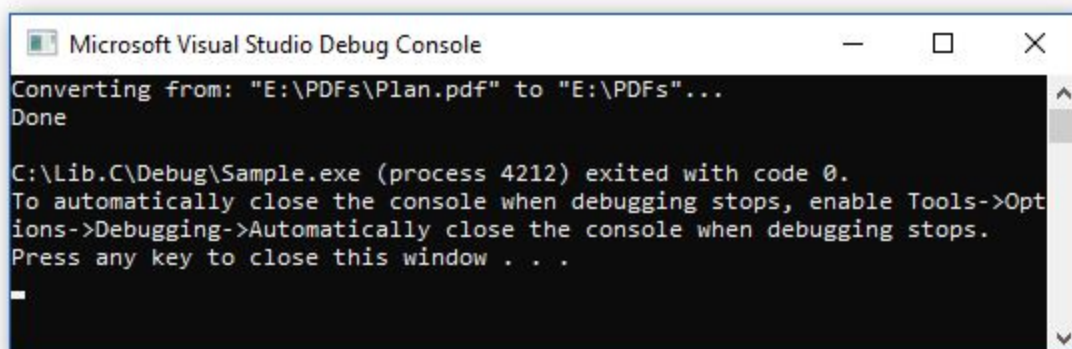
From the Sample Project Property dialog, on the left hand side, choose "Debugging" and fill in two paths for the Command Arguments field on the right. The first Argument is the path to the Input File that should be converted, while the second argument is the path for the output file, separated by "white space".





Note: The default output format for the conversion is plain text file. In order for the output file to be properly interpreted by the OS, it is strongly recommended for the output file extension to be .txt. This nomenclature can be seen in the screenshots. You also have the ability to change (or add) the extension after the conversion is complete.

Once you have successfully converted the file using the Sample application, you will see the confirmation in the Command Line window:



If no error occurred, the process will terminate automatically with an exit code 0. The default output format of the output file is Drawing Interchange format file.

5.2.3. Registration

PDF2Text.dll will convert only the first three pages of the document when running in trial mode. You can acquire the licence from the official Investintech website:

<https://www.investintech.com>

For successful and unlimited conversions, you will be provided with two components: a Licence.lic file and a personal password which should be passed on as an argument to the method:

```
PCSErrorCode PDF2TextInitialize( const char* pass)
```

The Licence.lic file should be placed in the same directory as the Sample.sln file. In order to initialize PDF2Text.dll, you need to pass your personal password String to the function PDF2TextInitialize(), replacing the NULL placeholder. You can use the following code snippet from Sample.cpp as a reference:

```
if((errorCode=PDF2TextInitialize( NULL ))!=IT_PCS_ERR_SUCCESS )
{
    wcout<< L"Initialization failed: "<< errorCode << endl;
    return errorCode;
}
```

As you can see, this snippet, included in the Sample file, provides basic error checking, giving the user information if the Licenced Initialization has failed.

5.3. Net C# sample project

This section is intended to simplify connecting the PDF2Text.NET.dll proxy library for Visual C# Sample project. First things first, you need to locate the sample project. This sample project is placed by default in the Documents directory, expressed through an environment variable in the location of:

```
"%USERPROFILE%\Documents\PDF2Text Samples\Net.C#"
```

It is strongly recommended to make a copy of Sample files on which you intend to work on, as this will make it easier to rollback changes if the need arises.

5.3.1. Resolving Dependencies

The first step in configuring sample files to release the power of PDF2Text.dll is running the "Sample.sln" file located in the ".\Net.C#" directory. This can be accomplished either by using Open Project/Solution from Visual Studio menu, or by simply double clicking on Sample.sln file.

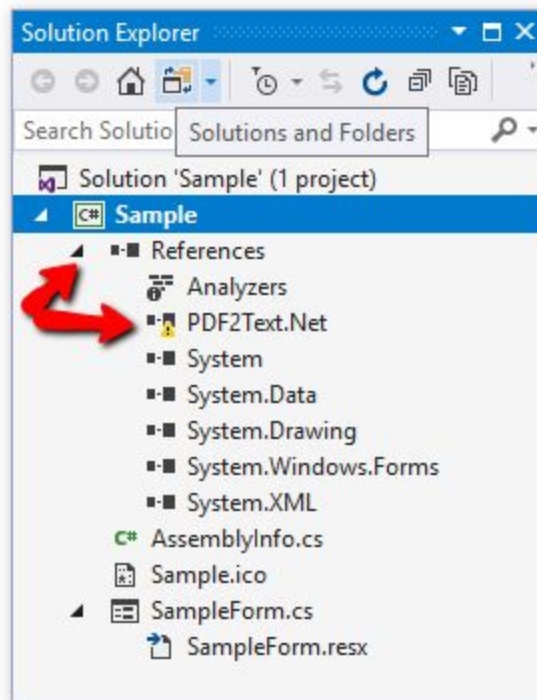
When the Solution is opened, no project files will be loaded at start, which is normal. However, by simply opening the Sample.sln solution, you'll see that Visual Studio has prepared additional directories required for successful running of the sample application, specifically "bin" and "obj" directories, placed in the same directory as Sample.sln file.

The directories that are of interest here are the sub-directories of the "bin" directory, which will store executables after the first build of the project. The name of the directory will depend on the Build Configuration used to build the application. Since the default configuration is "Debug" we will refer to this directory as the target one.

Next, using Windows Explorer, navigate to the installation directory of PDF to Text SDK, select all files and folders in the installation directory, and copy them in the ./bin/Debug/ directory of your sample project.

Once the files have been copied, you need to check if the proxy library PDF2Text.Net.dll dependencies has been resolved. Look for the Solution Explorer Window in your Visual Studio 2007. If Solution Explorer is not present, you can open it from the main menu: View -> Solution Explorer.

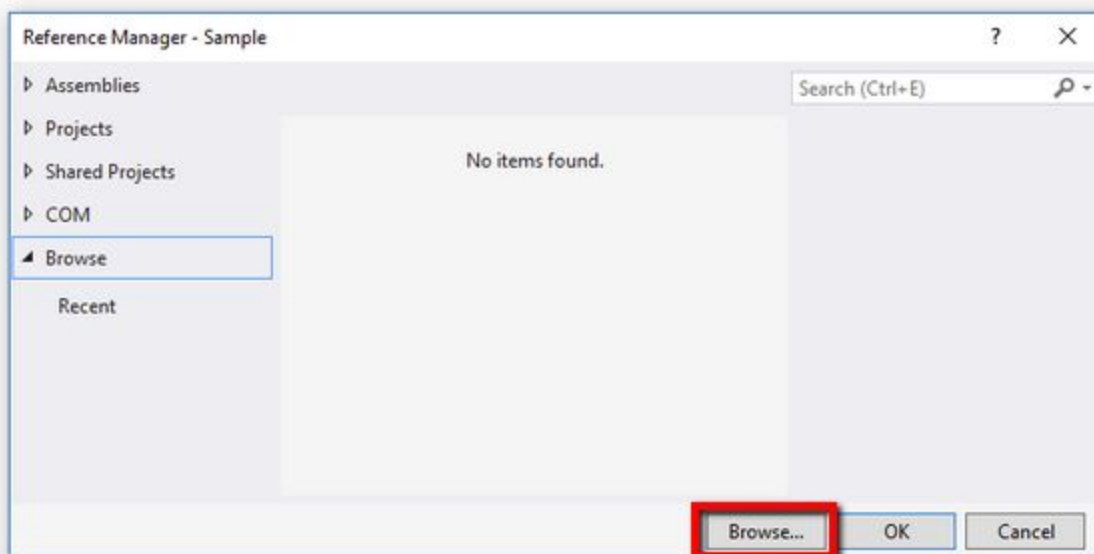
Inside Solution Explorer Windows, click the arrow-triangle left of "References" to see current project references. Solution Explorer should look similar to this:



If marked by a warning exclamation mark, PDF2Text.Net dll is not properly linked. This situation is shown in the screenshot above.

To fix this, right click on "References" to bring up context menu, and choose "Add Reference"

You should get the following window:



Note: It is quite common for the list of Recent files to be populated with other files if you worked with Visual Studio before. To avoid unnecessary clutter this list has been cleared, as it is shown in the screenshot.

On this screen click on the Browse... button and navigate to the location of your PDF to Text SDK tools. This is usually the installation directory chosen during the installation. Once you have navigated to the location of PDF to Text SDK tools, choose and add PDF2Text.Net.dll. After that, simply confirm the selection by clicking on the OK button.

5.3.2. Analyzing the SampleForm.cs

Once the References are resolved, you can open the `SampleForm.cs` from the solution explorer.

Because the `SampleForm` class is a subclass of `System.Windows.Forms.Form`, it will open by default in *design mode*. In order to access the code, you can either double click on design window, or simply press the F7 keyboard button, while `SampleForm.cs` is selected in Solution explorer.

With `SampleForm.cs` code visible, scroll down to main function. It should have the following structure:

```
static void Main()
{
    PDF2Text.Initialize("XXXXXXX");
    Application.Run(new SampleForm());
    PDF2Text.UnInitialize();
}
```

Argument of the method `public static void Initialize(string pass)` needs to be changed in order to use PDF2Text as a licenced product. This method accepts a String value that should be your personal password which is one of the two parts required to have a licenced SDK. For more details refer to Chapter 3.

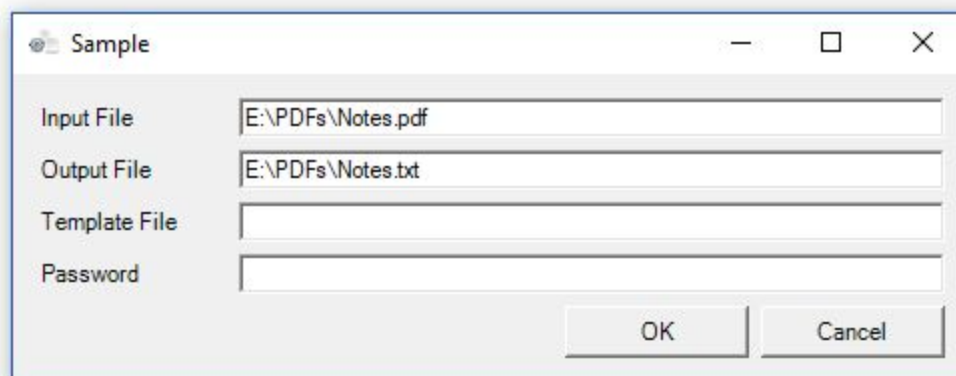
Here, the password has been replaced with "XXXXXX". To successfully run the sample application in non - trial mode, you will need to pass in the password provided after purchasing the licence.

Next, we need to concentrate on the function `okButton_Click()` which is triggered when the user clicks on the OK button on the Form. This part of the Sample application is controlling the actual call to conversion function.

Note: The current Sample solution doesn't automatically add an extension to the file names. It is up to the developer to set appropriate file extensions if needed.

5.3.3. Running the Sample Application

After you've successfully built the application and gave it a good test run, you should see the following window:



The last step needed before the PDF2Text engine can be seen in action is to supply at least two fields in the running form: Input File and Output file.

After the file paths have been provided, simply click on OK and after some time, depending on the size of the file, you'll have your first converted file using PDFtoText.dll

Note: If you're running a trial version of the SDK, a maximum of three pages of the input .pdf file will be converted.

Optionally you can choose different output formats for converted files, or use Template (Options) Files made using Able2Extract Desktop application.

Password field is mandatory only if the Input File is password protected. Currently, only ASCII / ISO 8859-1 (Latin-1) characters are supported for password.

6. Appendices

6.1. A -Troubleshooting: Command Line Tool

We, at Investintech, do our best to make our applications and development tools as stable as possible. Of course, no application is errorless, and when an error does happen it can often be avoided by consulting with the following table:

Problem	Possible Solutions
Input file not found: <filePath/Name>	<ol style="list-style-type: none">1. Use absolute file path example: "E:\PDFs\SomeFile.pdf"2. Check the spelling3. Use wildcard characters and/or recursion switch -r (section 3.4.1)
Output file exists: <filePath/Name>	<ol style="list-style-type: none">1. Specify a different output Directory (section 3.4.2)2. Delete the previously created file at the location <filePath>3. Use -eo or -em to overwrite or modify your output file name (section 3.4.6)
Unrecognized input file format: <filePath/Name>	<ol style="list-style-type: none">1. When using multiple file conversions with wildcards:<ol style="list-style-type: none">1.1. Increase restrictions on mask Example: use .pdf extension (*.pdf)1.2. Use -g switch to skip error (section 3.4.9)2. When using single file conversion:<ol style="list-style-type: none">1.1. File format is not supported.

Problem	Possible Solutions
SAXParseException: Tag mismatch in line 9 column 4	<ol style="list-style-type: none"> 1. Create or choose another template file (.pcvt) 2. Re-create an Option.pcv file in Able2Extract Desktop App
Initialization failed: 10	<ol style="list-style-type: none"> 1. Check for password and Licence integrity
PCS::ErrorCode::wrongParam	<ol style="list-style-type: none"> 1. Check if correct parameters are used. For reference see (Section 3.4) 2. Make sure no space is provided between a flag and the passed parameter.
The filename, directory name, or volume label syntax is incorrect.	<ol style="list-style-type: none"> 1. Make sure that the first argument is either the correct path to a file name, or a correctly formulated wildcard combination, followed by the -i switch with the correct folder location.

6.2. B - Troubleshooting: Shared Library

The table below contains the most common problems which users encounter when using Sample files. For a full list of possible errors, please refer to header file LIB.h, which can be used for reference when the conversion error is given in row value.

Problem	Possible Solutions
System.DllNotFoundException: 'Unable to load DLL 'PDF2Text.dll'	<ol style="list-style-type: none">1. Make sure that the required files and directories are in the same directory as binaries (see section 5.2.1)
Warning LNK4272 library machine type 'x64' conflicts with target machine type 'x86'	<ol style="list-style-type: none">1. Change your specified CPU platform architecture build from Configuration Manager in VS Section 5.2.12. Download the appropriate SDK build
Conversion failed: 3024 (File Save error)	<ol style="list-style-type: none">1. Change your output file name2. Delete the previous output file
Conversion failed: 5 Unrecognized File Type	<ol style="list-style-type: none">1. Check the input file name for spelling errors2. Include proper file extensions of the input file name3. Choose .pdf file for input file
Conversion failed: 2 Wrong Param	<ol style="list-style-type: none">1. Check the parameters passed2. Check if format and formatting are compatible (Section 5.2.5)
Conversion failed: 3022 File Create	<ol style="list-style-type: none">1. Change output file name to avoid file collision2. Delete previous output file