|  |
| --- |
| Witamy w SDK Tesseract.Net |

**Tesseract.Net SDK** to biblioteka klas oparta na [projekcie tesseract-ocr](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://code.google.com/p/tesseract-ocr/&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhgdVWhuKmxaynCbEyrXDaWDxZTANQ) . Może czytać szeroką gamę formatów graficznych i konwertować je na tekst w ponad 60 językach.

https://tesseract.patagames.com/help/icons/SectionExpanded.png Zgodność

Pakiet SDK Tesseract.Net jest dostępny dla .Net Framework 2.0 - 4.5 dla 32- i 64-bitowych systemów operacyjnych.

SDK został przetestowany pod Windows XP, Vista, 7, 8, 8.1 i 10 i jest w pełni kompatybilny z każdym z nich. Natywna biblioteka tesseract.dll dołączona do tego zestawu SDK jest dostarczana w wersjach 32-bitowej i 64-bitowej, więc Twoja aplikacja .NET może być "dowolnym procesorem".

https://tesseract.patagames.com/help/icons/SectionExpanded.png Pierwsze kroki

[Pobierz](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/downloads/&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhgtd-3KAqP_5n0NHkeOgCyovY3Ciw) pakiet instalacyjny, rozpakuj go i skopiuj następujące pliki do katalogu projektu

netXX \

Patagames.Ocr.dll

Patagames.Ocr.xml

x64 \

tesseract.dll

x86 \

tesseract.dll

tessdata \

configs \

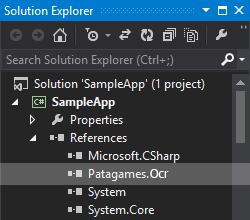
eng. \*

pdf.ttf

pdf.ttx

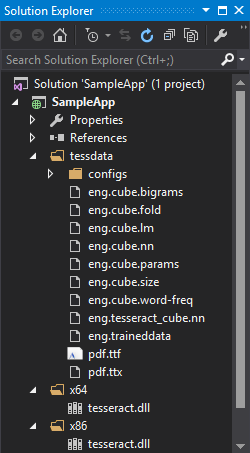
|  |
| --- |
| **Uwaga Uwaga** |
| Pliki **eng. \*** Odpowiadają językowi angielskim, który jest dostarczany w standardowym pakiecie. Jeśli chcesz używać innych języków, pobierz je [tutaj](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/langs/&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhhZ-3p-12pYjC_Ijl-kABMSIHqKEQ) osobno i umieść w folderze **tessdata** .  Jeśli chcesz skonfigurować silnik za pomocą plików konfiguracyjnych, umieść go w folderze **configs** . W przeciwnym razie ten folder nie jest potrzebny i można go usunąć. |

1. Aby użyć biblioteki, musisz najpierw dodać odniesienie do pliku **Patagames.Ocr.dll** specyficznego dla twojej wersji framework do twojego projektu.



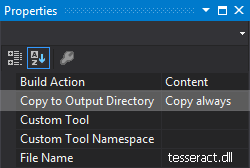
|  |
| --- |
| **Uwaga Uwaga** |
| Plik **Patagames.Ocr.xml** zawiera dokumentację XML interfejsu API. Tak długo jak jest w tym samym folderze co **Patagames.Ocr.dll** , Visual Studio wyświetla podpowiedzi do funkcji API. Nie musisz rozprowadzać tego pliku. |

1. Po dodaniu powyższej referencji musisz dodać następujące pliki do swojego projektu:
   * **x86 \ tesseract.dll** to 32-bitowa wersja biblioteki Tesseract;
   * **x64 \ tesseract.dll** to 64-bitowa wersja biblioteki Tesseract;
   * **tessdata** zawiera pliki wymagane do działania silnika Tesseract;



|  |
| --- |
| **Uwaga Uwaga** |
| Masz dwie opcje. Jeśli twoja aplikacja jest tylko 32-bitowa lub tylko 64-bitowa, możesz usunąć bibliotekę DLL, która nie będzie używana. Możesz pozostawić ten plik w katalogu x86 lub x64 lub przenieść go do katalogu głównego projektu. Biblioteka znajdzie bibliotekę DLL w obu przypadkach. |

1. Podczas budowania projektu biblioteki (biblioteki) **tesseract.dll** muszą znajdować się obok aplikacji, w podkatalogu głównym lub x86 lub x64. Folder **tessdata** musi być również umieszczony obok aplikacji w katalogu głównym.Najłatwiej to zrobić, zmieniając właściwości tych plików, zmieniając ustawienie **Kopiuj na katalog wyjściowy** na **Kopiuj zawsze** .



1. **To wszystko.** **Twój projekt jest gotowy do użycia SDK Tesseract.Net**

Teraz możesz utworzyć instancję klasy **OcrApi**

**DO#**

[Kopiuj](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/baa0aa10-7805-4ae6-b6e9-9df777c4678c.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhi2nLPCfWeNlSxsUqFPFTdMoaA0EA)

przy użyciu Patagames.Ocr;

...

var api = OcrApi.Create ();

|  |
| --- |
| **Ważna uwaga Ważny** |
| Jeśli masz problemy z tworzeniem (jeśli jest napisane coś w stylu "Nie można załadować biblioteki DLL" tesseract.dll "), spróbuj podać pełną ścieżkę do **pliku tesseract.dll** za **pośrednictwem** właściwości **OcrApi.PathToEngine** .Przeczytaj [tutaj,](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/e52e55c3-ed24-4b03-9833-40471554ed61.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhixKSXRlsHjsMeAOKhJIeP6WDqalw) aby uzyskać więcej informacji. |

https://tesseract.patagames.com/help/icons/SectionExpanded.png Używanie SDK Tesseract.Net w Twojej aplikacji

Dodanie funkcji OCR do aplikacji przy użyciu pakietu SDK Tesseract.Net jest łatwe. Główną klasą obejmującą wszystkie interfejsy API wysokiego poziomu biblioteki jest [OcrApi](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/T_Patagames_Ocr_OcrApi.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhhchweNvt3eTqBbvdW76MBx516gwQ) . Klasa [OcrResultRenderer](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/T_Patagames_Ocr_OcrResultRenderer.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhiREQDVZjvwQQZmXtz7aT_vjjB2sA) i jej childs służą do tłumaczenia wyniku rozpoznawania na określone formaty wyjściowe, w tym PDF, HTML i inne. Funkcje niskiego poziomu, które umożliwiają pracę z poszczególnymi akapitami, słowami, literami i parametrami czcionki, są implementowane w klasie [OcrResultIterator](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/T_Patagames_Ocr_OcrResultIterator.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhhN469rKsjRQG-fF9bF62fTD40ixQ) .

Szczegółowy diagram klas można znaleźć [tutaj](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/24482ca6-f2b1-4b6a-b024-d838a5562cc9.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhjd5tAkevdPiX6OfnvwpwuSnKe2VA) .

Oto typowy kod C # demonstrujący sposób wyodrębniania zwykłego tekstu z obrazu. Tutaj tworzymy instancję klasy OcrApi i inicjujemy ją dla języka angielskiego. Następnie po prostu otrzymujemy tekst z obrazu.

**DO#**

[Kopiuj](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/baa0aa10-7805-4ae6-b6e9-9df777c4678c.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhi2nLPCfWeNlSxsUqFPFTdMoaA0EA)

public void ExtractTextFromImage ()

{

using ( var api = OcrApi.Create ())

{

api.Init (Languages.English);

string plainText = api.GetTextFromImage ( "c: \ test\_image.png" );

}

}

Metoda GetTextFromImage nie tylko działa z plikami obrazów, ale może również rozpoznawać tekst na danej mapie bitowej, na przykład System.Drawing.Bitmap. Oto przykład takiego użycia:

**DO#**

[Kopiuj](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/baa0aa10-7805-4ae6-b6e9-9df777c4678c.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhi2nLPCfWeNlSxsUqFPFTdMoaA0EA)

public void ExtractTextFromBitmap ()

{

using ( var api = OcrApi.Create ())

{

api.Init (Languages.English);

using ( var bmp = Bitmap.FromFile ( "c: \ test\_image.png" ) jako Bitmap)

{

string plainText = api.GetTextFromImage (bmp);

}

}

}

Jeśli skompilujesz powyższe przykłady i dodasz do aplikacji następujący obraz:



daje następujący wynik zwykłego tekstu:

" Nie ma takiej rzeczy

jako zbyt zajęty.

Jeśli naprawdę chcesz coś,

zrobisz na to czas "

Jednak zwykle jesteśmy bardziej zainteresowani tworzeniem przeszukiwalnych plików PDF z tych zeskanowanych obrazów, a nie zwykłego tekstu. Potrzebna jest tylko drobna modyfikacja powyższego kodu, aby wygenerować plik PDF:

**DO#**

[Kopiuj](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/baa0aa10-7805-4ae6-b6e9-9df777c4678c.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhi2nLPCfWeNlSxsUqFPFTdMoaA0EA)

public void Tiff2Pdf ()

{

using ( var api = OcrApi.Create ())

{

api.Init (Languages.English);

// Utwórz renderer do pliku wyjściowego pliku PDF. Rozszerzenie zostanie dodane automatycznie

using ( var renderer = OcrPdfRenderer.Create ( " plik\_kopii\_wielok\_plikowej " , @ "c: \ YourApp \ tessdata \" ))

{

renderer.BeginDocument ( "Tytuł" );

api.ProcessPages ( @ "c: \ multipage.tif" , null , 0 , renderer);

renderer.EndDocument ();

}

}

}

Tutaj tworzymy renderer PDF i tworzymy strony procesu API obrazu źródłowego (multipage.tif) i zwracamy wynik rozpoznawania do renderera. W rezultacie powstaje plik output.pdf. Plik zawiera strony zeskanowanego obrazu reprezentowanego jako tekst, który można skopiować i przeszukać.

Jeśli podasz [ten wielostronicowy tiff](https://tesseract.patagames.com/help/Media/multipage.tif) do powyższego kodu, wygeneruje [następujący dokument PDF](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/Media/multipage.pdf&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhhglhWiQBeQancp-Pz5fbXbvJ0IWQ) .

Nie musisz konwertować obrazów na wielostronicowy TIFF przed utworzeniem pliku PDF z rozpoznanym tekstem. W rzeczywistości, nawet jeśli masz zakres pojedynczych obrazów, nadal możesz połączyć je wszystkie w jeden plik PDF.Aby to zrobić, należy dostarczyć zwykły plik tekstowy do metody ProcessPage zamiast TIFF. Ten zwykły plik tekstowy powinien zawierać nazwy plików wszystkich pojedynczych obrazów, po jednym w wierszu: "scanned.txt"

C: \ page1.jpg

c: \ page2.png

c: \ page3.bmp

Zmodyfikuj kod, jak pokazano poniżej, a otrzymasz 3-stronicowy, możliwy do przeszukiwania plik PDF z tych 3 pojedynczych obrazów.

**DO#**

[Kopiuj](https://translate.googleusercontent.com/translate_c?depth=1&hl=pl&rurl=translate.google.pl&sl=en&sp=nmt4&tl=pl&u=https://tesseract.patagames.com/help/html/baa0aa10-7805-4ae6-b6e9-9df777c4678c.htm&xid=25657,15700022,15700124,15700149,15700168,15700186,15700190,15700201,15700205&usg=ALkJrhi2nLPCfWeNlSxsUqFPFTdMoaA0EA)

pustka publiczna MultiplyImages2Pdf ()

{

using ( var api = OcrApi.Create ())

{

api.Init (Languages.English);

// Utwórz renderer do pliku wyjściowego pliku PDF. Rozszerzenie zostanie dodane automatycznie

using ( var renderer = OcrPdfRenderer.Create ( "output\_pdf\_file" , @ "c: \ YourApp \ tessdata \" ))

{

renderer.BeginDocument ( "Tytuł" );

api.ProcessPages ( @ "c: \ scanned.txt" , null , 0 , renderer);

renderer.EndDocument ();

}

}

}