OmniPage Capture SDK User's Guide
User's Guide
Version:

Date: 2019-10-21

KOFAX

Legal

END USER LICENSE AGREEMENT

IMPORTANT: PLEASE READ THIS END USER LICENSE AGREEMENT ("LICENSE AGREEMENT") CAREFULLY BEFORE INSTALLING OR USING THE SOFTWARE. THE SOFTWARE IS COPYRIGHTED AND LICENSED (NOT SOLD). BY INSTALLING OR USING THE SOFTWARE, YOU ARE ACCEPTING AND AGREEING TO BE BOUND BY THE TERMS OF THIS LICENSE AGREEMENT. EXCEPT TO THE EXTENT THE SOFTWARE IS SUBJECT TO A SEPARATE WRITTEN SOFTWARE LICENSE AGREEMENT BETWEEN YOU AND KOFAX, THIS LICENSE AGREEMENT WILL SUPERSEDE ANY AND ALL LICENSE AGREEMENTS GOVERNING ANY LICENSES OF THE SOFTWARE PREVIOUSLY GRANTED BY KOFAX (AND ITS PREDECESSORS IN INTEREST) TO YOU.

1. **Software License Grant.**

   a. **License Grant.** Subject to Your acceptance of the terms and conditions of this License Agreement and conditional on payment of all required fees, Kofax, Inc. ("Kofax") grants You non-exclusive nontransferable licenses to use the Software, including any upgrades and new version releases that may be provided to You from time to time (as and when available as part of Kofax's Software Maintenance and Support program), for Your internal use in object code form only and as otherwise provided in this License Agreement. Your licenses allow You to use the Software only for the purposes (production, evaluation, testing, demonstration, disaster recovery) and for the duration and extent for which You have paid the appropriate license fees, as evidenced by one or more valid order documents (a "Sales Order") between You and Kofax or between You and an authorized Kofax reseller or distributor identifying the specific software and accompanying hardware (if any) products licensed (the "Software"), the limitations on use of the Software (such as volume limitations or concurrent client module use limitation), and any other then current licensing policies for the Software. You agree to exercise the same level of care against unauthorized use by, or disclosure to, third parties as You use with respect to Your own proprietary information of comparable importance, provided that in no event will You use less than reasonable care.

   b. **Restrictions.** You will use the Software only for Your internal business purposes and only for Your direct benefit, and You will not attempt to use the Software, or any portion thereof, in excess of its licensed capacity. You will neither permit nor permit any third party to (i) reverse engineer, decompile, disassemble, decrypt, re-engineer, reverse assemble, reverse compile or otherwise translate or create, attempt to create the source code of the Software or perform any process intended to determine the source code for the Software, or (ii) modify, enhance or create derivative works based upon the Software or otherwise change the Software. Any modification, enhancement, derivative work or other improvement to the Software developed by You, whether with or without the consent of Kofax, will be the exclusive property of Kofax and subject to and governed by this License Agreement.

   c. **U.S. Government Entities.** If You are a U.S. Government entity, then Your use, duplication or disclosure of the Software is subject to the following restricted rights clause: The Software is a "Commercial Item," as that term is defined in 48 C.F.R. §2.101, consisting of "commercial computer Licensed Software" and "computer software documentation," as such terms are used in 48 C.F.R. §252.227-7014(a)(1) and 48 C.F.R. §252.227-7014(a)(5), respectively, and used

in 48 C.F.R. §12.212 and 48 C.F.R. §227.7202, as applicable, and all as amended from time to time. Consistent with 48 C.F.R. §12.212 and 48 C.F.R. §227.7202-1 through 227.7202-4, and other relevant sections of the Code of Federal Regulations, as applicable, and all as amended from time to time, all U.S. Government entities license the Software (i) only as Commercial Items, and (ii) with only the rights explicitly set forth in this License Agreement and the Sales Order.

d. (d) Third Party Software. Any portion of the Software that constitutes third party software, including software provided under a public license, is licensed to You subject to the terms and conditions of the software license agreements accompanying such third-party software, or as set forth in the thirdpartylicenses.txt file accompanying the Software.

2. **Copy of Software.** You may make one copy of the Software in machine-readable form for the purpose of backup in the event the installers or executables are damaged or destroyed; provided, that any backup copy of the Software must include all copyright, trademark, and other proprietary notices contained on the original.

3. S**oftware Maintenance and Support.** If You purchase Software Maintenance and Support for the Software, the Software Maintenance and Support will be provided as described in Kofax's then current Software Maintenance and Support Agreement (Schedule), available upon request, and which is incorporated herein by this reference. Kofax will have no liability to You arising from or related to Your cessation of Software Maintenance and Support, whether from Your failure to timely renew Software Maintenance and Support or otherwise. If You elect to reinstate Software Maintenance and Support following expiration of the Software Maintenance and Support for whatever reason, You will (a) pay a reinstatement fee equal to the sum of the current annual Software Maintenance and Support fees, any unpaid Software Maintenance and Support fees from the date of expiration to the date of reinstatement, and an amount equal to one additional year of Software Maintenance and Support fees, and (b) apply all upgrades, enhancements and new releases to the Software needed to bring Your Software current with Kofax's most current supported version of the Software. Software Maintenance and Support pricing will increase for renewal terms by an amount not to exceed 5% of the prior year term fee, provided that increases associated with additional software license purchases, if any, will be incorporated into the base for the purpose of calculation of each annual increase.

4. **Intellectual Property.** You acknowledge and agree that (i) the Software is licensed and not sold, (ii) by accepting the licenses set forth in this License Agreement, You acquire only the right to use the Software in accordance with the terms of this License Agreement, and that Kofax and/or its licensors will retain all rights, title, interest, including all associated patent, copyright, trademark, trade dress, trade secret and other proprietary rights in and to the Software, and (iii) the Software, including the source and object codes, logic and structure, constitute valuable trade secrets of Kofax. You agree to secure and protect the Software with the same degree of care which You employ to protect Your own intellectual property of a similar nature, but in no event less than a reasonable standard of care.

5. **Warranties.**

   a. **Warranties.** Subject to the limitations stated herein, Kofax warrants to You, the original end user, that, for a period of ninety (90) days from the date the Software is made available to You the Software, as delivered (a) will materially conform to Kofax's then-current documentation for such Software, and (b) does not contain any computer worms or viruses. To be eligible for a remedy under this warranty You must report all warranted problems to Kofax in writing within the warranty period. Your exclusive remedy, and Kofax's entire liability, under this warranty will be, at Kofax's option, to provide a correction or a workaround for any reproducible errors or other noncompliance, the replacement of the non-conforming Software, hardware key, media and/or documentation, or a refund of the license fees You paid for the affected Software,

subject to Your return of the Software. This limited warranty is void if You have modified or altered the Software, installed, operated, repaired or maintained the Software other than in accordance with the then-current documentation for such Software, subjected the Software to misuse, negligence, or accident, or cannot reasonably reproduce the error reported by You. Any replacement Software will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

b. **DISCLAIMER OF ADDITIONAL WARRANTIES.** THE EXPRESS WARRANTIES ABOVE ARE IN LIEU OF ALL OTHER WARRANTIES, AND KOFAX MAKES NO REPRESENTATIONS OR WARRANTIES CONCERNING THE SOFTWARE, EXPRESSED OR IMPLIED, EXCEPT AS EXPRESSLY PROVIDED HEREIN, AND EXPRESSLY DISCLAIMS TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW ANY AND ALL OTHER WARRANTIES, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT OR SKILL AND CARE. ANY IMPLIED WARRANTIES THAT BY LAW CANNOT BE DISCLAIMED ARE LIMITED IN DURATION TO THE GREATER OF (I) NINETY (90) DAYS FROM THE DATE OF THIS LICENSE AGREEMENT, OR (II) THE SHORTEST PERIOD PERMITTED BY LAW.

6. **Intellectual Property Indemnification.**

a. **Indemnification.** Kofax will indemnify and defend, at its own expense, any claim, suit or proceeding brought against You by a third party to the extent it is based upon a claim that Your use of the Software in the United States, Canada, Australia or the European Union pursuant to this License Agreement infringes upon any patent, copyright or trade secret of a third party. If You comply with the provisions hereof, Kofax will pay all damages, costs and expenses finally awarded to third parties against You in such action. If the Software is, or in Kofax's opinion might be, held to infringe as set forth above, Kofax may, at its option (i) acquire the right for You to continue to use the Software upon the terms of this Agreement, (ii) modify the Software to avoid or correct the infringement, or (iii) replace the Software. If none of such alternatives are, in Kofax's opinion, commercially reasonable, You will return the infringing Software to Kofax, and Kofax's sole liability, in addition to its obligation to pay awarded damages, costs and expenses as set forth above, will be to refund the license fees You paid to Kofax hereunder, depreciated on a 3-year, straight-line basis.

b. **Limitations.** The foregoing notwithstanding, Kofax will have no liability for any claim of infringement arising as a result of (i) Your use of the Software in combination with any items not supplied by Kofax, (ii) any modification of the Software by You or at Your request, (iii) use of other than the latest revision of the Software if use of the latest revision would avoid the infringement, (iv) use of the Software outside the scope of the granted licenses or otherwise in violation of the terms of this License Agreement, or (v) any other act or omission by You which is a breach by You of any term of this License Agreement

c. **Conditions to Indemnification.** Kofax will have the sole right to control the defense of, and to settle or compromise, any claim of infringement concerning the Software, and Kofax’s indemnification obligations are conditioned upon You (i) giving Kofax prompt written notice of any claim for which indemnity is sought, and (ii) fully cooperating in the defense or settlement of any such claim. Subject to the foregoing, however, You, at Your own expense, may participate, through its attorneys or otherwise, in the investigation, trial and defense of any such claim, demand or action and any appeal therefrom.

d. **Exclusive Remedy.** The foregoing states Kofax’s entire liability and Your exclusive remedy concerning infringement of intellectual property rights, including but not limited to, patent, copyright and trade secret rights.

7. **Limitation of Liability.** UNDER NO CIRCUMSTANCES WILL EITHER PARTY BE LIABLE TO THE OTHER FOR ANY PUNITIVE DAMAGES OR LOST PROFITS OR OTHER ECONOMIC LOSS, LOST OR DEGRADED DATA, INTERRUPTION OF BUSINESS, PROCUREMENT OF SUBSTITUTE PRODUCTS, OR FOR INDIRECT, SPECIAL, CONSEQUENTIAL, EXEMPLARY OR INCIDENTAL DAMAGES (INCLUDING WITHOUT LIMITATION ANY LOSS OF BUSINESS, REVENUE, GOODWILL OR USE), HOWEVER CAUSED AND REGARDLESS OF THEORY OF LIABILITY, ARISING OUT OF THE USE OF (OR INABILITY TO USE) THE SOFTWARE PROVIDED HEREUNDER, EVEN IF SUCH PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THESE LIMITATIONS APPLY TO ALL CAUSES OF ACTION IN THE AGGREGATE, INCLUDING CAUSES OF ACTION ARISING OUT OF TERMINATION OF THIS LICENSE AGREEMENT, BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, STRICT LIABILITY, MISREPRESENTATION, PRODUCT LIABILITY AND ANY OTHER TORTS. THE MAXIMUM AGGREGATE AMOUNT FOR WHICH EITHER PARTY MAY BE LIABLE UNDER THIS LICENSE AGREEMENT WILL BE LIMITED TO THE AMOUNTS ACTUALLY PAID OR PAYABLE BY YOU FOR THE SOFTWARE SUBJECT OF THE CLAIM FOR WHICH SUCH LIABILITY IS ASSERTED DURING THE EIGHTEEN (18) MONTHS PRECEDING THE CLAIM. THIS SECTION WILL NOT APPLY, HOWEVER, TO A PARTY'S BREACH OF CONFIDENTIALITY OR TO ANY CLAIM ARISING OUT OF YOUR BREACH OF THE LICENSE RESTRICTIONS SET FORTH IN THIS LICENSE AGREEMENT.
8. **Trademarks.** You recognize Kofax's ownership in and title to all trademarks and/or service marks owned by Kofax and set forth in the Software, including any and all common law and registered rights throughout the world (hereinafter the "Trademarks"). No right, license or interest in or to any of the Trademarks is granted hereunder, and You agree that You will assert no such right, license or interest with respect to such Trademarks. Furthermore, You will not contest the validity of any of the Trademarks, claim adversely to Kofax any right, title or interest in and to the Trademarks and will not use, register, apply to register or aid a third party in registering the Trademarks during the term of this License Agreement or any time thereafter.
9. **Term and Termination.**
   a. **Term.** The term of this License Agreement and Your licenses of the Software will commence as of the first to occur of the date of Your acceptance of this License Agreement or the date the Software is made available to You, and will continue until the termination or expiration of the term of all of the licenses of the Software, unless earlier terminated at the end of any timeframe specified in a Sales Order or as provided below.
   b. **Termination.** Kofax may terminate this License Agreement (i) effective ten (10) days after written notice to You in the event that You fail to pay when due any fees for the Software as provided in a Sales Order, or (ii) effective thirty (30) days after written notice to You in the event that You breach any other material provision of this License Agreement and You do not cure such failure to pay or breach within such thirty (30) day period.
   c. **Rights and Obligations upon Termination or Expiration.** Upon termination of this License Agreement, all rights granted to You hereunder will immediately cease and You will (i) immediately discontinue all use of the Software, and (ii) destroy all copies of the Software. Termination of this License Agreement for any reason will not excuse Your obligation to pay in full any and all amounts due for the Software, nor will termination result in a refund of any fees paid by You for the Software.
   d. **Continuing Obligations.** The terms and conditions in this License Agreement that by their nature and context are intended to survive any termination of this License Agreement, including, without limitation, Sections 4 (Intellectual Property), 6 (Intellectual Property Indemnification), 7 (Limitation of Liability), 8 (Trademarks), 9 (Term and Termination), 10 (Audit), and 11

(Miscellaneous), will survive such termination of this License Agreement for any reason and will be fully enforceable thereafter.

10. **Audit.** Kofax, upon thirty (30) days written notice to You and not more than once during each calendar year during the term of this License Agreement and once during the one (1) year period following the termination of this License Agreement, may enter upon Your premises during Your regular business hours to audit Your use of the Software. You agree to cooperate with Kofax's audit and provide reasonable assistance and access to Your systems and information. If pursuant to any such audit, Kofax discovers any excess or unlicensed use of the Software, You agree to pay within thirty (30) days of written notification an amount equal to the sum of (a) the license fees which Kofax would have received for the additional licenses necessary to license such excess or unlicensed use of the Software at Kofax's then current list pricing, and (b) if Your excess or unlicensed use of the Software exceeds 105% of the licensed use of the Software, all costs and expenses incurred by Kofax in conducting such audit. If You fail to pay such amounts within thirty (30) days of being invoiced for such amounts, Kofax may terminate this License Agreement, Your licenses of the Software, and any maintenance and support of the Software. You will be responsible for any of Your costs incurred in cooperating with any such audit.

11. **Miscellaneous.**

    a. **Notices.** All notices, demands or other communications under this License Agreement must be in writing and reference this License Agreement, and will be deemed effectively delivered to the party when delivered at the address for such party as last provided to the other, subject to modification by giving notice as provided herein. Notices may be delivered: (a) by email using a method that positively establishes receipt of the email by the recipient; (b) by personal, same or next day delivery; or (c) by commercial overnight courier with written verification of delivery. All notices so given will be deemed given upon the earlier of receipt or three (3) days after dispatch.

    b. **Governing Law.** This License Agreement will be construed and governed in accordance with the internal laws of the State of Delaware, without regard to any rules of conflicts or choice of law provisions that would require the application of the laws of any other jurisdiction. The foregoing notwithstanding, however, if You acquired the Software in a country which is a member of the European Union, the laws of that country will govern the interpretation of this License Agreement and any claims arising hereunder, regardless of choice of laws principles of any other jurisdiction. In each case, this License Agreement will be construed and enforced without regard to the United Nations Convention on the International Sale of Goods or the Uniform Computer Information Transactions Act.

    c. **Severability.** If any one or more of the provisions of this License Agreement is determined to be invalid, illegal, or unenforceable, the validity, legality, and enforceability of any of the remaining provisions or portions thereof will not be affected or impaired thereby and will nevertheless be binding between the parties. In the event any provision of this License Agreement is found to be invalid, illegal, or unenforceable, the parties will modify that provision in a manner that gives effect to the intent of the parties in entering into the License Agreement.

    d. **Waiver or Delay.** No failure to exercise or delay by a party in exercising any right, power, or remedy under this License Agreement operates as a waiver of such right, power, or remedy. A single or partial exercise of any right, power, or remedy does not preclude any other or further exercise of that or any other right, power, or remedy. A waiver is not valid or binding on the party granting the waiver unless made in writing.

    e. **Export Laws.** The Software is subject to United States export control jurisdiction, and may not be shipped, transferred, re-exported to any country or recipient, or used for any purpose

prohibited by any applicable international and national laws that apply to the Software, including the U.S. Export Administration Regulations as well as end-user, end-use, and destination restrictions issued by the United States and other governments. You will not export or re-export the Software without first obtaining the appropriate U.S. or foreign government export licenses.

f. **Entire Agreement.** This License Agreement (including the Kofax Licensing Policies) constitutes the entire understanding and agreement between the parties with respect to the subject matter of this License Agreement and supersedes all previous agreements and communications between the parties concerning such subject matter. No modifications may be made to this License Agreement except in writing, signed by both parties.

g. **Benefit of Agreement.** This License Agreement will bind and inure to the benefit of the parties and their respective permitted successors and assigns.

h. **Cumulative Remedies.** Except as otherwise provided in this License Agreement, all remedies of the parties hereunder are non-exclusive and are in addition to all other available legal and equitable remedies.

i. **Force Majeure.** Neither party will be liable or deemed to be in default for any delay or failure in performance under this License Agreement (except for payment obligations) resulting, directly or indirectly, from acts of God, civil or military authority, acts of the public enemy, war, riots, civil disturbances, insurrections, accidents, fire, explosions, earthquakes, floods, the elements, strikes, labor disputes or any causes beyond its reasonable control; provided that the party failing to perform in any such event will promptly resume or remedy, as the case may be, the performance of its obligations hereunder as soon as practicable.

j. **Construction of Agreement.** Each party acknowledges that it has had the opportunity to review this License Agreement with legal counsel of its choice and agrees that in the event that this License Agreement or any other documents delivered in connection with the transactions contemplated by this License Agreement contain any ambiguity, such ambiguity will not be construed or interpreted against the drafting party. The titles and headings herein are for reference purposes only and will not in any manner limit the construction of this License Agreement, which will be considered as a whole.

k. **Choice of Language.** The original of this License Agreement has been written in English, which will be the controlling language in all respects. Any translations into any other language are for reference only and will have no legal or other effect.

l. **Personal Data; Consent to Process and Transfer.** You agree to comply with all applicable laws and regulations which may govern Your use of the Software, including, but not limited to, laws pertaining to the collection and use of personal data and to the transfer of data over state or other jurisdictional lines. You agree that Kofax, its affiliates, and agents may collect and use information you provide in relation to any support services performed with respect to the Software and requested by You. Kofax agrees not to use this information in a form that personally identifies You, except to the extent necessary to provide such services. You agree that Kofax may transfer Your information to the United States or other countries for use in accordance with this Section.

Third Party Licenses/Notices

The following acknowledgments are presented in the alphabetical order of their titles.

Asian character recognition

Partner: Wintone Ltd.

Copyright notice: Asian OCR capabilities in this product are jointly developed by the Beijing Wintone Information Technology Corporation Ltd and Kofax Inc. All rights reserved.

Asian codepage conversion data files

Partner: ICU project

Copyright notice: International Components for Unicode (ICU) project Copyright (c) 1995-2009 International Business Machines Corporation and others.

Dictionaries from Proximity

Partner: Proximity Technology

Copyright notice: The language dictionary support of this software is partly from Proximity. The Proximity Dictionaries © 2000, all rights reserved Proximity Technology, Inc.

Dictionaries from Vantage Research

Partner: Vantage Research

Copyright notice: International CorrectSpell ™ spelling correction system © 1993 by Lernout & Hauspie

Dictionary for Esperanto language

Partner: Toon Witkam and Stefan MacGill

Copyright notice: Esperanto dictionary based on compilation by Toon Witkam and Stefan MacGill.

Dictionary for Slovenian language

Partner: Amebis d.o.o.

Copyright notice: Slovenian Speller Database, Copyright © 2002 Amebis d.o.o.

Export Options dialog controls

Partner: Allan Nielsen

Copyright notice: Supergrid control, copyright © 1999 Allan Nielsen.

File compression library

Partner: Jean-Loup Gailly and Mark Adler.

Copyright notice: Zlib copyright © 1995-2010 Jean-loup Gailly and Mark Adler.

Font file handling

Partner: The FreeType Team

Copyright notice: This software is based in part on the work of the FreeType Team. Copyright © 2012 The FreeType Project <www.freetype.org>.

Handprint recognition

Partner: re Recognition GmbH

Copyright notice: Alphanumerical handprint recognition module Copyright © 1993-2019 re Recognition GmbH

JPEG image file library

Partner: Independent JPEG Group

Copyright notice: This software is based in part on the work of the Independent JPEG Group. Copyright © 1991-2009, Thomas G. Lane, Guido Vollbeding.

JPEG 2000 image file library

Partner: Kakadu Software Pty Ltd

Copyright notice: The JPEG 2000 image read and write capability of this software was developed using the Kakadu software.

Licensing server access

Partner: Daniel Stenberg and many contributors

Copyright notice: This software is based in part on the libCURL project, licensed under a modified MIT license. Copyright (c) 1996 - 2012, Daniel Stenberg

OOXML Office document support

Partner: Florian Reuter and the libOPC hackers

Copyright notice: This software is based in part on the libOPC project, licensed under the BSD-3 license. Copyright (c) 2011 Florian Reuter.

PDF encryption

Partner: Dr Brian Gladman

Copyright notice: AES encryption/decryption is based on the BGAES code. Copyright (c) 2001, Dr Brian Gladman, Worcester, UK.

PNG image file handling

Partner: Glenn Randers-Pehrson and others

Copyright notice: The Portable Network Graphics (PNG) image file read and write capability of this software is based, in part, on the PNG Reference Library (libpng).

QR Code and DataMatrix Barcode reading

Partner: ZXing authors

Copyright notice: This software is based in part on the ZXing project, licensed under the Apache-2 license. Copyright 2008 ZXing authors.

XML handling in Office documents

Partner: Daniel Veillard and other Authors

Copyright notice: This software is based in part on the libXML2 project, licensed under the MIT license. Copyright (c) 1998-2003 Daniel Veillard.

Table of Contents

Preface

This guide provides an overview and instructions from purchasing the Omnipage Capture Software Development Kit to developing your own application.

Related documentation

The product documentation set for Capture Software Development Kit is available at the following location.

https://docshield.kofax.com/Portal/Products/en_US/OmniPageCaptureSDK/21.0.0_y3b7m3tg1v/OmniPageCaptureSDK.htm

Getting Help for Kofax Products

Kofax regularly updates the Kofax Support site with the latest information about Kofax products.

To access some resources, you must have a valid Support Agreement with an authorized Kofax Reseller/Partner or with Kofax directly.

Use the tools that Kofax provides for researching and identifying issues. For example, use the Kofax Support site to search for answers about messages, keywords, and product issues. To access the Kofax Support page, go to www.kofax.com/support.

The Kofax Support page provides:

- Product information and release news

  Click a product family, select a product, and select a version number.
- Downloadable product documentation

  Click a product family, select a product, and click **Documentation**.
- Access to product knowledge bases

  Click **Knowledge Base**.
- Access to the Kofax Customer Portal (for eligible customers)

  Click **Account Management** and log in.

  To optimize your use of the portal, go to the Kofax Customer Portal login page and click the link to open the *Guide to the* Kofax *Support Portal*. This guide describes how to access the support site, what to do before contacting the support team, how to open a new case or view an open case, and what information to collect before opening a case.
- Access to support tools

  Click **Tools** and select the tool to use.

- Information about the support commitment for Kofax products
  Click **Support Details** and select **Kofax Support Commitment**.

Use these tools to find answers to questions that you have, to learn about new functionality, and to research possible solutions to current issues.

Training

Kofax offers both classroom and computer-based training to help you make the most of your Omnipage Capture Software Development Kit solution. Visit the Kofax website at www.kofax.com for details about the available training options and schedules.

Chapter 1

Introduction

Welcome to the Omnipage Capture Software Development Kit, released by Kofax. This kit provides you with image management and recognition technologies for building 32-bit and 64-bit Microsoft® Windows® applications. We are confident it will be a productive tool in your development work. To those of you examining an evaluation copy, we hope your experience will be positive and lead to a purchase.

Overview of building and distributing applications

To guide you through the process from purchasing the Omnipage Capture Software Development Kit to developing your own application, see the following topics:

- Pre-installation steps and installation:
  - Prepare for installation
  - Installation steps
- Getting started with the CSDK:
  - Interface properties
  - Sample viewer application
- Distribution:
  - Prepare distribution file set
  - Test Applications Using Temporary Test Licenses
- Assistance:
  - Getting help
  - Introduction

Quick start with SDK

Prepare for installation

- It is not necessary to remove previous Capture Software Development Kit products. If you wish to do it, open the **Windows Control Panel** > **Add/Remove programs** . For earlier releases of the current version, check the *Release Notes* in the ReadMeEng.htm.
- Ensure that the installation package is available. Download the appropriate 32-bit or 64-bit package from the Kofax Network. The exact location is included in the fulfillment letter.
- Browse the downloaded and unzipped content for the latest information and the *Release Notes* (ReadMeEng.htm file).

- Check that your system meets the minimum requirements for memory and disk space and be ready to select install location. See System requirements.
- Ensure that internet access is available for product license activation. It is required to use the SDK:

  Internet access can be available on your developer machine (the computer where you are about to install the SDK), or on another one that you use for reaching the activation webpage. In case your developer machine has internet access, it is recommended that you activate your license directly online. If you choose to activate using a second machine, make sure you can transport files between this computer and your developer workstation, and use manual activation. For details see *General Information*.

Installation steps

Make sure that all other Windows programs are closed; we recommend that the anti-virus software be disabled.

The licensing service (Central Licensing Service) has to be added to the list of firewall exceptions on the machine you plan to use for activation.

You do not have to provide your license key during installation.

1. Ensure you have administrator rights on the installation computer.
2. Browse in the downloaded and unzipped content to the filesetup.exe and run it.
3. Accept the license agreement when prompted, and press Next.
4. Provide your User Name and that of your Organization.
5. Accept or define the destination path and a program folder for the program binaries and icons.
6. When installation is complete, click Finish.
   - The CSDK setup program installs all files to the developer system. Components like Form Template Editor, and Document Classifier Assistant can be selected during the setup procedure.
   - If you do not have the Microsoft .NET 4.7.1 Framework on your computer, it is installed silently during CSDK installation, because it is a prerequisite for several SDK components.
   - In some cases, the installer recommends a reboot. This have to be accepted.

Check installation

To check whether your CSDK installation was successful, you can perform some basic tests by means of the RecAPI Sample Viewer Application, see Sample viewer application.

This sample has additional information for you to check, for instance:

- the version number of the CSDK you have just installed

You can safely ignore error messages displayed under Module Information in this window. The program only generates these messages for demonstration purposes.

1. Launch the Sample Viewer, then run Sample 1.

   The module information test lets you check that all licensed modules are really available.

   The Output Window displays the results of initialization and Module Information. The initialization test confirms that installation and license activation were successful and also that the recognition engine is ready for use on your machine.

2. If initialization fails, check that the required activation steps (in particular, running the NCLT) were performed properly, for more information, see the *General Information* online help.
3. Check the additional information, for example the version number of the CSDK you have just installed.

Getting help

The Toolkit is shipped with four HTML help files:

- *General Introduction*: System requirements, Licensing, Testing and Distributing Applications, and so on
- *RecAPI*: Full reference of all functions, enums and parameters for C++ implementation. It also contains the full RecPDF API reference, and most of the overview and reference topics which are not repeated for IPRO or Visuals.
- *IPRO*: Full reference of the object model, events, methods and properties.
- *Visuals*: Full description of all visual components.

The help includes Tutorials, Sample Codes and Reference topics. Access this through the Windows Start menu or Help buttons in the programs. Readme files present latest information and solutions to recently reported problems.

All these items are stored in electronic form in the sub-folder `Documentation`. The full collection is accessible to those who are evaluating the Kit, and also to purchasers, regardless of which Kit or Add-ons are purchased.

Interface properties

A new object-oriented .NET API is introduced instead of the old PInvoke-based API. Developers are encouraged to utilize this for greater efficiency.

For evaluation purposes we recommend working with the RecAPI interface and possibly the supplied test application, ITest. Use the IPRO interface only if you need to access the functionality of the Visual Toolbox. For your development work both interfaces yield the same recognition accuracy. RecAPI is divided into two parts, so your evaluation and development choices are as follows:

RecAPI C/C++ interface (KernelAPI)

Best suited for

- high throughput single page processing tasks with plain text output
- applications where processing time needs to be minimized

RecAPI interface (RecAPIPlus)

- supports document-level and high-volume (possibly unattended) processing
- supports fully formatted output, one-step workflow-style processing and new features such as redaction and form data extraction
- leverages multithreading to deliver performance benefits on multi-core and hyper-threaded computers

IPROPlus ActiveX

- necessary for developers who wish to take advantage of the Visuals Toolbox
- possibility to create and use OmniPage-compatible workflows
- supports document-level processing, along with the ability to easily access all objects and properties involved

Sample viewer application

A sample viewer application is provided for each interface. These are designed to demonstrate the functionalities of the toolkit through nearly 50 working samples per platform. We recommend you view the sample indexes to determine which are closest to your configuration and development plans, and run those. For a general overview, we suggest you run samples 1-14, to test basic functionality.

Sample availability is determined by your license. Viewing samples is only possible after you have activated your license.

Launch the viewer of the interface you plan to use for your development.

- CSharp Objects Samples

  Start Menu > **All Programs** > **OmniPage Capture SDK 21 x64** > **Samples** > **CSharp Objects Samples (C# 10.0)**
- IPRO Sample Viewer Application

  Start Menu > **All Programs** > **OmniPage Capture SDK 21 x64** > **Samples** > **IPRO Sample Viewer Application**
- RecAPI Sample Viewer Application

  Start Menu > **All Programs** > **OmniPage Capture SDK 21 x64** > **Samples** > **RecAPI Sample Viewer Application**
- Visual Controls Sample Application

  Start Menu > **All Programs** > **OmniPage Capture SDK 21 x64** > **Samples** > **Visual Controls Sample Application**

The Sample Viewer window displays the following areas:

- Description window summarizes the purpose of the current sample.
- Console window displays status messages as the samples run to denote progress and to report any problems encountered.
- Function window lists all functions/objects involved in running the sample. Select a function or object, then press Help for detailed information on that item.

Note To use a scanning device with the CSDK, it must be already installed using its own installation software and functioning correctly. Please consult the scanning topics in the documentation.

Distribute the application

Prepare distribution file set

This section describes how to distribute the necessary Engine files with your application developed using the CSDK.

The Distribution Wizard is a separate executable which you can use to collect distribution file sets. Because it collects the files from the CSDK installation folder, it always gets the latest installed content, so if you have any patch or point release installed, the file set collection will reflect this updating.

The CSDK setup program installs all files to the developer system. Components, for example Form Template Editor, and Document Classifier Assistant can be selected during the setup procedure on the Customize installation screen.

These steps are described in detail in the General Information help.

1. Prepare an Engine configuration for distribution (create a distribution file set).
2. Include the file set in a setup program or installer you have written.
3. Copy these Engine files (together with your integrating application files) to your customer machine.
4. Read the hints in the Distribution Wizard log, and make sure the necessary system files are installed on the target computer.
5. Setup the user's system for scanning, if necessary.
6. Ensure that recognition modules (and the PDF output converter and other Add-ons) used by your integrating application get licensed.

Test Applications Using Temporary Test Licenses

After you have finished compiling the distribution file set, the next step is testing your application. This guide presents the basic outline: for details see the corresponding topic in the General Information Help.

The test computer can be any computer where the CSDK has not been installed, and no developer license has been activated.

All aspects of the application should be tested and passed through a QA procedure, to ensure correct interaction with the CSDK in the test environment.

1. Be aware which recognition modules are used by your application.
2. Check whether your distribution licensing supports all the RECOGNITIONMODULEs required by your application.
3. Ensure that your application checks the availability of required modules.
4. Select the distribution file set of the Engine.
5. Activate one or more temporary Seat Licenses on the test computer.
6. Check which Engine modules are licensed on the test computer.
7. Check which Engine modules are licensed, available and initialize correctly.
8. Fully test the application. Remove the log file (DistributionWizard.log) and the DISTR_TST.exe executable from the test folder.

Performance comparison of Engine combinations

CSDK supports multiple omnifont OCR engines for integration. This chart shows the speed/accuracy trade-off using selected Engine combinations and trade-off settings. Data was measured on a 3GHz Core Duo machine with 2GB RAM, with the following test set:

- About 5000 pages in single-page TIFF files, with 1.13 million words
- Mixed resolutions (200-300 dpi), both black-and-white and color
- Mixed languages (English, French, German, Italian, Hungarian, Polish, Portuguese)
- Mixed layout (magazines, business letters, faxes, business cards, and so on)
- Mixed scanners

For both speed and accuracy the 100% value is for the Plus2W configuration. That means the fastest configuration is slightly over twice as fast as the Plus2W configuration. Starting with CSDK version 21, the default configuration is Plus3W, that provides the highest accuracy. The percentage differences translate into word accuracy values as follows:

Most accurate (Plus 3W)	97.2%
Plus 2W	96.9%
Fastest (FRX + Legacy)	92.5%

Image size limits

For information on image size limits, see the *RecAPI Help*.

Uninstall the product

Before uninstalling the Capture Software Development Kit, it is recommended that you move your fixed-volume development license to a different computer. For this operation, see the *General Information* help.

Note Be sure to keep your license key and license codes in a safe place; you may need them to re-install the program or to install an upgrade. Some files may not be removed during uninstallation, for example, result files generated by running the sample viewers.

Go to the **Windows Control Panel** and select **Programs and Features** (Windows 7) and then choose **OmniPage Capture SDK**.

Chapter 2

System requirements

System requirements for computers used to develop a Capture SDK-enabled application

Supported operation systems:

- Windows Server 2008 R2
- Windows 7 x86/x64
- Windows Server 2012/R2
- Windows 8/8.1 x86/x64
- Windows 10 x86/x64
- Windows Server 2016
- Windows Server 2019

Note CSDK runs on Server type operating systems when the Server license feature is present in the CSDK license.

Windows RT is not supported in this release.

- Intel Core (higher CPU is recommended)
- 4 GB minimum RAM (6 GB recommended, more for working with grayscale or color images and more for multithreaded applications)
- 4 GB free disk space

System requirements for computers used to run a Capture SDK-enabled application

Supported operation systems:

- Windows Server 2008 R2
- Windows 7 x86/x64
- Windows Server 2012/R2, also Core
- Windows 8/8.1 x86/x64
- Windows 10 x86/x64
- Windows Server 2016, also Core in Docker containers

Note CSDK runs on Server type operating systems when the Server license feature is present in the CSDK license.

Windows RT is not supported in this release.

- Intel Core (higher CPU is recommended

- 2 GB minimum RAM (4 GB recommended, more for working with grayscale or color images and more for multithreaded applications)
- 450 MB free hard disk space (less if not all recognition modules are distributed)

Chapter 3

New Features and Changes

This document presents an overview of the innovations, improvements and changes in version 21 of the CSDK.

Screen capture OCR accuracy Improvement

CSDK contains a new preprocessing algorithm for recognizing screen captures.

If the images to be processed can include screen captures, you need to set Kernel.Img.ScreenCapture setting to SCR_AUTO or SCR_YES.

If SRC_AUTO option is selected, CSDK analyzes the image and uses the optimal algorithm.

Objects API for .NET Core supports Docker container environments on Windows (and later Linux)

Kofax.OmniPageCSDK.NETCore.Objects.dll, Kofax.OmniPageCSDK.NETCore.CAPI.dll, and Kofax.OmniPageCSDK.NETCore.RecPDF.dll assemblies have to be used in their .NET Core applications, that is .NET Core v2.1.

For CSDK usage in Docker, check the *Container section of Technical Note*.

RecAPI for Java

- It is the JNI implementation of the RecAPI. The current implementation is based on Java version 8.
- The csdk_omnipage.jar and csdk_omnipage.dll files can be in the `bin` folder.
- The complete Java Sample can be in `Samples\JavaSamples`. The Build.cmd builds the sample. Run it from an Admin command prompt. Run.cmd -x 1 to build sample #1.

Essay, New Output Format support

Essay mode concept is about exporting layout metadata unified at document-level, allowing the user to easily process further the exported document. It uses the following functions:

- Newly introduced support for styles and heading levels
- Improved Headers/Footers settings

Enable Styles, Heading levels, and Header/Footer features, and select one of the new Essay mode converter shortcuts (Converters.Text.DocXEssay, Converters.Text.Rtf2000Essay, or Converters.Text.Html50Essay) when exporting a document. The already existing export flags can be also used.

Improvement in accuracy

Significant accuracy improvements are achieved in mixed alphanumeric text recognition.

Form Template Datarules

User can assign data rules, for example wordlist, regular expression or logical expressions, to form fields. After the recognition process, the form fields and the letters of the fields are flagged if they fulfil the data rules or not. Users can quickly verify the erroneous fields. Using this feature increases the reliability and the reparability of forms.

Form Template Adaptation

Form Template Editor now supports Form Template Adaptation, a new operation to create a new form template based on an already existing similar template.

Changes in CSDK **Licensing**

CSDK Installer does not install License Service for CSDK. After the install, you need to use OPLicMgr.exe or OPLicMgrUI.exe from the `bin` folder to license the CSDK. Seat and OEM licenses should be managed using these tools.

The previous central licensing has been replaced by the new component, OmniPage Licensing Agent. OPLA functionality is very similar to NLS/NCLS and it is used for page-based licenses.

See more detailed information on the new licensing in the *Licensing section of the General Information* document.

The default Omnifont OCR engine configuration change to accurate one

The default OCR recognition engine is RM_AUTO. This engine used 2-way vote OCR on Latin based input images. From version 21 it is changed to 3-way vote where it is possible. The default OCR engine is more accurate and slightly slower. If an application requires the previous 2-way voting, set Kernel.OcrMgr.PreferAccurateEngine to false.

Kernel.Img.Resolution.Restore

The default value changed from FALSE to TRUE.

Assembly and Namespace changes in .NET APIs

In this version .NET assemblies and its namespaces were changed due to the rebranding to Kofax Inc. The new names are descriptive and easy to use. Applications must be changed a bit to refer to the assemblies and namespaces.

Generally Nuance.OmniPage.CSDK is replaced with Kofax.OmniPageCSDK. Check the samples for exact names.

Old IProPlus .NET support API and its Samples are retired and not supported

There is a newer version. The applications have to use the current IProPlus .NET API, Kofax.OmniPageCSDK.IproPlus.dll and Samples\HowCouldWeUseIt.

Revoked binaries	Commands to recreate
IproPlus.NET.dll	`tlbimp IproPlus.dll /out:IproPlus.NET.dll /sysarray /namespace:IproPlus`
IproPlus.SRV.NET.dll	`tlbimp IproPlus.exe /out:IproPlus.SRV.NET.dll /sysarray /namespace:IproPlus`
MediumWeightVisualsLib.dll	`tlbimp MediumWeightVisuals.dll /out:MediumWeightVisualsLib.dll /sysarray`
MediumWeightVisuals.NET.dll	`aximp MediumWeightVisuals.dll /out:MediumWeightVisuals.NET.dll /rcw:MediumWeightVisualsLib.dll`
VisualsLib.dll	`tlbimp Visuals.dll /out:VisualsLib.dll /sysarray`

Revoked binaries	Commands to recreate
Visuals.NET.dll	`aximp Visuals.dll /out:Visuals.NET.dll /rcw:VisualsLib.dll`

Old RecAPI P/Invoke API (CAPI_PInvoke.dll) is retired

The applications have to use the new assemblies, Kofax.OmniPageCSDK.CAPI.dll and Kofax.OmniPageCSDK.ArgTypes.dll.

Old ISAppMFC and VSAppMFC samples are retired

Study the Samples\HowDoWeUseIt. It is a complete sample for C/C++ applications.

MAX image file support is retired

Version 21 is not able to load MAX image files.

New features of earlier versions can be found in the documents of the given release.

Chapter 4

RecAPI

The Omnipage Capture Software Development Kit lets developers access a broad range of document processing algorithms and workflows for integrating them into their own applications. Its native C programming interface is RecAPI, which consists of three main parts:

- KernelAPI, that contains the page level operations.
- RecAPIPlus, that is an extension over KernelAPI for performing document based tasks.
- RecPDF, that is also an API extension, which can be used for managing direct page level manipulation of PDF files.

Throughout the documentation we frequently refer to CSDK as Engine meaning all the modules of RecAPI.

This chapter is intended to give you an overview of RecAPI, describing the following topics:

- Introducing RecAPI
- KernelAPI Modules
- RecAPIPlus
- Alternate Linking of RecAPI

Introduction to RecAPI

RecAPI is a traditional, standard native "C" API separated into two layers: KernelAPI and RecAPIPlus. This simplifies the architecture and also makes the offering of the toolkit more flexible: higher speed processing to simple text output, or more thorough processing with formatted output. RecAPI also provides a base for RecAPI P/Invoke assembly for .NET managed applications.

A higher layer set of services is provided by IPRO and RecIPRO - with an updated object model. IPRO serves as the base for a range of visual components. To access IPRO and MediumWeight Visuals documentation, use their own *Help* systems or see IPRO and Introduction to Visual Toolbox.

RecAPI modules

KernelAPI

This is a page-oriented interface performing OCR. The process includes image loading, pre-processing, page parsing, recognition, and simple output conversion. It can provide plain text (DirectTXT), XML, and searchable PDF output.

KernelAPIS

This is a wrapper for KernelAPI to provide it with alternate linking, so the application and the CSDK engine can be separated. It provides more robust functioning.

RecAPIPlus

This is a document-oriented extension to KernelAPI that provides a full range of formatted outputs. It includes document-level layout consolidation. It allows efficient multi-page high-volume processing, particularly on multi-core systems. It offers composite one-step functions to perform a succession of tasks from one command.

RecAPIPlus S

This is a wrapper for RecAPIPlus to provide it with alternate linking, so the application and the CSDK engine can be separated.

Object oriented RecAPI model support for .NET users

This is a bridge between native and managed applications. It allows the functionality of KernelAPI and RecAPIPlus to be presented through a unified interface.

The RecAPIPlus part of RecAPI contains a user interface based on a document-oriented approach. You can implement functionality similar to that of OmniPage Ultimate through just a handful of functions. When exporting multi-page documents you can use a series of output converters to provide layout retention quality equal to that in OmniPage Ultimate. Getting high quality page layout retention and document level formatting increases the total processing time.

The KernelAPI part of RecAPI is useful when processing speed is a critical factor and/or there is no need for layout retention. This is the basic API with page-level functions. Therefore, this is very similar to APIs of earlier CSDK versions. We recommend that you use it for applications that do not require total format retention and rich output formats. KernelAPI generates its simple outputs directly from HPAGE. Here you can choose TXT, CSV, two different XML types, or PDF (image on text). Since page formatting is not present here, page-oriented processing on the KernelAPI level gets faster.

KernelAPI modules

General operations module

This module contains the basic types and functions necessary to use the general services of the KernelAPI. Initialization is an essential prerequisite for utilizing KernelAPI functionality. User authorization can also be performed in this module. You can get useful information about the CSDK, for example accessible modules.

Most RecAPI operations work through Settings Collections. A Settings Collection acts as a kind of profile. It stores a set of operational settings, handled by the Settings Manager Module.

Distinct Settings Collections can store different values for given settings. A Settings Collection is authorized by an identifier that must be passed to all operations using that Settings Collection. Users can create and delete Settings Collections.

Error handling module

The majority of RecAPI operations have the return value type RECERR. This is an enumeration type collecting all possible error codes in CSDK. The integrating application should always check the return value of RecAPI functions to make sure that the function has successfully completed its operation.

Use this module to obtain information about any errors that occurred in the latest operation and to see their categories.

Image file handling module

This module supports loading and saving different image file formats. During loading, the system automatically detects the file format and compression method of the image files without using file extensions. Both this and the Image Handling Module support B/W, grayscale and color images. CSDK can read the following multi-page image file formats: DCX, MAX, PDF, TIFF and TIFF-FX. The input can be an image file, or the memory of the integrating application. After the file is loaded, the image is forwarded to the Image Handling Module, that can save the input.

To achieve faster image file handling for multi-page TIF and PDF files, call kRecOpenImgFile prior to calling kRecLoadImg. This way there is no need to lose time opening the same image file repeatedly for each page. The open, load, and close (kRecOpenImgFile, kRecLoadImg, kRecCloseImgFile) functions are connected by the HIMGFILE handle in an already familiar way. To load single-page images, or to load single pages from multi-page files, you can use the kRecLoadImgF function. In this case, it is not necessary to use the HIMGFILE handle.

Image handling module

The CSDK Engine is page oriented, the basic unit used in processing is typically the image of the page. Most functions require a page as an input parameter, but some API calls also operate on rectangular areas of the image (zones). The Image handling module enables the Engine to handle several pages at the same time. The Engine can only manage and process pages within its own memory space. The images must be loaded into the Engine by the Image file handling module, or the Scanning module. Possible image sources include files, scanning devices or the memory of the integrating application. The best resolution for B/W images is 300 or 400 dpi. The optimal recognition resolution for grayscale or color images is 150 to 300 dpi.

CSDK offers pre-recognition functions to apply image preprocessing procedures to images. These functions enhance the image quality, and yield more accurate auto-zoning (carried out by the Zone Handling Module) and recognition (carried out by the Recognition Module). Preprocessing can include any or all of the following steps:

- Inversion: automatic, programmed, or none
- Rotation: automatic, programmed, or none
- Deskewing: automatic, programmed, or none
- 3D DESKEW: automatic, programmed, or none
- Use of SET tools through the ActiveX Image Viewer Control (IVC)

In addition, image despeckling and resolution enhancements are performed internally to render better results, with or without the above transformations.

Additional tasks in this module include:

- User modification of the loaded images
- Detection and cutting book pages (when the image contains two book pages)
- Auto-detection of table location and structure for the Table recognition module
- Line information detection
- Auto-detection of border, box or frame location and styling

Scanning module

The scanning subsystem supports a wide selection of scanner models through TWAIN or WIA. For conditions of implementing ISIS support please contact the Kofax sales staff. The following conditions must be fulfilled to use the scanning subsystem:

- The scanning subsystem must be initialized for actual scanning. Any other scanning function can only be called after the successful initialization of the scanning subsystem.
- The scanner type must be set through the Scanner Setup Wizard. It can be accessed through the API.
- The Scanner Setup Wizard supports a majority of the popular scanner models. You can update your scanning system through the Wizard by downloading a hint file from the internet.
- The scanning subsystem requires an additional initialization besides the Engine initialization, done with kRecScanInit.
- Both the Scanner Setup Wizard and the entire scanning subsystem can be distributed along with your integrating application.
- When the scanning subsystem is initialized, images coming from the scanner device are forwarded to the Image Handling Module.

Note During scanning, the integrating application has to disable all functionalities that can interfere with the scanning. For example, the application should not allow the user to terminate it by any means.

The whole scanning module is thread safe. It is possible to initiate two scanning processes in parallel, using two scanners. Calls from two different threads to a single scanner are queued.

Settings manager module

This is the manager of settings at the KernelAPI level. The settings are organized into a tree. Every node of the tree has a symbolic name, which is determined by its path in the tree. These names can be divided into parts along their point separators. Every section of the name between two points (dots) corresponds to an individual node of the path. The sections of the name of a given node appear in the correct order to denote all the nodes in the path going from the root of the tree to the given node. The name of the root is an empty string. There may be non-setting nodes in the tree. Nodes, and therefore settings can be accessed via handles.

The settings of StsMan can have the following types:

- Integer, including int, bool and enum
- Double
- Integer array, including int, bool, enum and set array

- Double array
- String
- Unicode string

DevKit users can create their own settings, store them in the tree and maintain them in the same way as existing settings. Every setting has a default value, which cannot be changed during the whole lifetime of that setting. It can be set only when it is created. On the other hand, every setting has a current value, which can be different in different Settings Collections. The current value can be changed anytime in any collection. Settings values from any collection can be saved into files, and loaded from them later.

Zone handling module

The zone is a rectangular area or the union of specifically located rectangular areas in the page. The upper limit of its dimensions is full page size. It contains a feature that is interesting to the user. The union of rectangles must have a pizza box shape. It means the top of each rectangle in the union must touch the bottom of the rectangle above it. A pizza box-shaped zone is a compound and an irregular zone.

The image data covered by each zone is handled and processed separately, according to zone-specific parameters.

This module can handle two types of zones in separate zone lists of each page: user zones and OCR zones. Zones can be added to the proper zone list of any given HPAGE in the following ways:

- OCR zones: add zones automatically (auto-zoning)
- User zones:
  - add zones manually (by specifying the zone coordinates and attributes)
  - add zones from a zone file (created by the user through API functions)
  - add zones from a template library (created by Form Template Editor)

You can start auto-zoning directly, or wait for it to run automatically at the beginning of the recognition process provided that there are no OCR zones already defined.

You can also modify the method and performance of auto-zoning through different settings. When you use user-zones, you can specify whether to have auto-zoning also run or not. Individual user zones are transformed into one or more OCR zones prior to recognition. Recognition is always carried out on OCR zones, and the post-processing can also change the zones.

Based on their content, zones can be classified as follows:

- Flowing text (horizontal or vertical with three rotations)
- Table (handled by the Table Recognition Module)
- Graphics (no recognition performed)
- Form (handled by the Form Recognition Module)

Form recognition module

Analyze and activate empty forms

Analyze empty forms and convert them into an active (fillable) digital form document formats.

The current empty form recognition core is mainly based on the LFR technology which was the backbone of OmniForm, and is now an integrated part of the CSDK. LFR recognizes textual and non-textual elements in empty forms, for example check boxes, circle texts, comb fields, tables and cells, graphics, lines, and boxes. These detected form field zones cannot be modified programmatically, but the page can be viewed and modified in the TEC.

The aim of this module is to turn a form image into a fillable electronic form that can be distributed to respondents or posted on a website. The data in the returned forms can then be processed by any suitable means, outside the scope of the CSDK. The filled form data is returned electronically, so there is no need to use anchor zones.

1. Load a single-page image of a form.
2. Specify user zones manually.
   - Define the whole page as a single form.
   - Define one or more separate form type user zones.
3. Perform page decomposition.

   LFR runs inside the defined form zones, as part of the Zone handling module. It auto-detects form elements and creates active form controls.
4. Inspect the auto-detected form objects in the TEC visual control.

   It is possible to move, resize or delete auto-detected objects, and add new objects. TEC also lets you view and modify form object attributes.
5. Save the results to a format that supports active form elements: formatted XML, RTF (Microsoft Word) and PDF, optionally with Acro-Form-like fillable field definitions.

   Form object attributes can also be modified in the target application.

Process filled forms

Recognize the contents of filled forms based on pre-constructed form templates and send the respondent data contents to digital format that can be easily processed.

Recognize the contents of filled forms based on pre-constructed form templates and send the respondent data contents to digital format that can be easily processed. A form template is a set of form-field zones and anchor definitions. Form-field zones are normal input zones with some additional information attached. They identify the areas where fill-in data is required on a specific form, for example fill-texts, combs, and checkboxes. Form-field zones may contain data rules that specify the syntax of the form-fields. CSDK checks whether the extracted form data matches the data rule and returns information about it. Anchor definitions are specified as ignore zones during template definition. They provide additional information to assist the matching of the template to incoming forms.

Form Templates are organized into template libraries. Use the Form Template Editor tool to create and prepare form templates, organize them into template libraries and test how they match to the test forms. See the *About the Form Template Editor* section of the *Form Template Editor* Help for details. To process

filled forms, create and test form templates with the Form Template Editor, and export the form template library.

To process filled forms, perform the following steps in your application:

1. Load the exported template library.
2. Load the image of the form to be processed.
3. Pre-process the image.
4. Call `kRecFindFormTemplate` to find the matching template.
5. Call `kRecApplyFormTemplateEx` to apply the template onto the image.
6. Recognize the image.

Direct TXT Output Converter Module

This module allows you to convert recognized text. That is, you use the output of the recognition module as it is, without reading order and paragraph detection. Therefore the DirectTXT Outputs are simpler and faster to produce than the Layout Retention Output conversions that are available in RecAPIPlus, because DirectTXT Outputs do not include slow detection processes. The following DirectTXT output types are available:

- DirectTXT Text output: a simple text file.
- DirectTXT CSV output: a comma-separated text file, a simple format to represent tables. Microsoft Excel can read this format.
- DirectTXT Formatted Text output: This converter delivers plain text, but attempts to keep the page layout as detected in the original image. It creates a text file that simulates columns and boxes using tabulators.
- DirectTXT PDF output: contains the whole image of the original page and the text behind the image on a separate layer (image on text PDF). These PDF files suit the purpose of page archiving, because they contain both the original image and recognized text.
- DirectTXT XML output: typically used for further processing recognized data. You can easily parse, for example, to MSXML, or transform to XSLT the output XML file. The format of the XML output is specified by the same schema as the Layout Retention XML Output, see http://www.scansoft.com/omnipage/xml/ssdoc-schema3.xsd.
- DirectTXT Binary output: used for creating files directly from the recognition data without any character conversion and formatting. It is the most usable output format for barcodes containing binary data, for example Code128 or PDF417 barcodes, containing encrypted data.

When you specify an already existing file name, the TXT type outputs are appended.

The DTXT module can be especially useful for applications that do not require formatting but speed is an important factor, for example: indexing, archiving, or some form processing applications. When programming with KernelAPI, this is your only output choice. It is possible to purchase distribution licenses that exclude formatted output, see Prepare distribution file set.

Language, Character Set, and Code Page Handling Module

This module is required to be present for successful Engine initialization. This module handles language, character set and code page related settings and their combinations.

Recognition module

The Engine can load several recognition engines. Licensing determines which ones are available. You can select which available engine to run in a given user zone by specifying a particular filling method suitable for the zone content, or by specifying an actual recognition engine. In addition, you can also set which recognition module will be used by the Engine for the processing of the individual OCR zones that are created automatically. If no instruction or guidance is given, the recognition engine is selected automatically.

The following recognition engines are available:

- PLUS3W: 3-way voting omnifont engine for machine-printed text
- PLUS2W: 2-way voting omnifont engine for machine-printed text (default)
- MTX: M/TEXT omnifont engine for machine-printed text in 12 languages
- MOR: MOR multi-language omnifont engine for machine-printed text in over 120 languages
- FRX: Fireworx multi-language omnifont engine for machine-printed text in 56 languages
- DOT: engine for 9-pin or 24-pin draft dot-matrix printouts
- MAT: matrix matching engine for codified scripts (for example, OCR-A, and OCR-B)
- HNR: engine for hand-printed digits
- RER: re Recognition (third-party) engine for hand-printed alphanumerical characters
- OMR: engine for optical marks, for example checkmarks
- BAR: engine for bar codes

PDF recognition

PDF files can be separated into two basic classes, image-only PDF and normal PDF. Recognition of image-only PDF is exactly the same as that of any other image format. The normal PDF contains both image and text data. The CSDK can extract this textual information to boost OCR accuracy. In general, text in a PDF file is reliable, so accurate results are possible even when the quality of the imaged text is low. See New Features and Changes for improvements in PDF handling. The fullest description of PDF functionality in the CSDK is found in the *Related Pages* section of the *RecAPI online help* system.

Recognition data handling module

In addition to the usual output choices, you can also access raw data, providing significantly more information than the character codes and attributes. This data stores all information that can be obtained on characters. For example, it gives access to alternatives, such as character tips or word suggestions. It also shows recognition confidence data from the recognition engines that ran in conjunction with Spell Checking Module output.

Spell checking module

The spell checking module consists of the following separate parts:

- Spell checking: using language-specific dictionary elements. It uses third-party language-specific spell checkers. There are two kinds of spell checkers:
  - Language dictionaries (language checking)
  - Vertical dictionaries (professional dictionaries for a given set of languages)
- User-written checking: using user-written callback functions.

Note User dictionaries and user-written callbacks are deprecated.

You can combine any of these steps during the recognition process to determine the acceptability of words. The use of these checks can still be enabled or disabled separately at zone level.

The CSDK is delivered with over 20 different language dictionary files. These are generic language dictionaries and typically contain between 100,000 and 200,000 entries.

Vertical dictionaries are based on specific professions. They can be treated as extensions to the language dictionaries, but they can also be used when no language dictionary is specified. The CSDK is delivered with the following vertical dictionaries:

- Dutch Legal Professional Dictionary
- Dutch Medical Professional Dictionary
- English Financial Professional Dictionary
- English Legal Professional Dictionary
- English Medical Professional Dictionary
- French Legal Professional Dictionary
- French Medical Professional Dictionary
- German Legal Professional Dictionary
- German Medical Professional Dictionary

A User Dictionary is a list of words, and regular expressions, called UDitems. User Dictionaries can be created or modified by the user manually or through KernelAPI calls. User-written checking callback functions are parts of the integrating application. They receive the string to be checked and the index of the zone where this string comes from. The application must evaluate the string, express the acceptability of the recognized string, then return this to the Engine. The Recognition Module uses feedback from the Spell Checker module along with other data to make its assessment of recognition confidence.

Note The confidence reporting system works best when the 3-way voting engine runs. If other machine print recognition modules are used, confidence information is still available, but the ability of the system to report on confidence is reduced. This results in a higher level of false negative and false positive reporting of suspicious recognition results.

Table recognition module

This module detects tables on the page and links the recognition result and the table information to each other. A table is described by the list of its cells in reading order, from left to right and top to bottom,

ordered by their top-left coordinates. Cells cannot have blank spaces between them, they cannot intersect each other, and all cells must be rectangular.

Alternate linking of RecAPI

The KernelAPI and RecAPIPlus interfaces use dynamic linking. This means all required DLL files are loaded at initialization time. It also requires all DLL files to be placed at a known location. The CSDK searches them using the Standard Windows Search Order. The program and its DLL files need to be located at a predetermined path. It may constrain the distribution of the integrating application. The KernelAPIS and RecAPIPlusS modules solve this problem. These are thin interface layers for binding the Engine at run-time. They load KernelAPI or RecAPIPlus from a path specified in the initialization function and get all the entry points of the required DLL files. The actual Engine load happens when kRecInitS or kRecInitPlusS is called. Other DLL files are loaded when a function, that requires their services, is called.

Since these modules are wrappers on the corresponding modules KernelAPI and RecAPIPlus, they make it possible to separate the integrating application from the files of the CSDK without placing the CSDK folder into the PATH environment variable. That makes the application less vulnerable to processing issues raised by the CSDK, and avoids the risk of identically named program files in the application and the CSDK.

Depending on how the release or debug versions of the integrating application use the C run-time libraries, one of the following small `KernelAPIS` or `RecAPIPlusS` static libraries should be linked to the application:

Release version of KernelAPIS

KrnAPIS.LIB	single-threaded C run-time library	statically linked
KrnAPIS_MS.LIB	multi-threaded C run-time library	statically linked
KrnAPIS_MDyn.LIB	multi-threaded C run-time library	dynamically linked

Release version of RecAPIPlusS

RecAPIPlusS.LIB	single-threaded C run-time library	statically linked
RecAPIPlusS_MS.LIB	multi-threaded C run-time library	statically linked
RecAPIPlusS_MDyn.LIB	multi-threaded C run-time library	dynamically linked

The debug version of the integrating application requires a static library to be linked as follows:

Debug version of KernelAPIS

KrnAPISd.LIB	single-threaded C run-time library	statically linked
KrnAPIS_MSd.LIB	multi-threaded C run-time library	statically linked
KrnAPIS_MDynd.LIB	multi-threaded C run-time library	dynamically linked

Release version of RecAPIPlusS

RecAPIPlusSd.LIB	single-threaded C run-time library	statically linked
RecAPIPlusS_MSd.LIB	multi-threaded C run-time library	statically linked
RecAPIPlusS_MDynd.LIB	multi-threaded C run-time library	dynamically linked

RecAPIPlus

RecAPIPlus, as a part of RecAPI, contains a programming interface based on a document-oriented approach and provides formatted page output.

General service functions

These functions are responsible for the initialization and termination of CSDK, loading and saving RecAPIPlus settings, and setting the count of the OCR threads.

Simple multi-page document handling

With these functions the application of the user can implement document-based solutions. The application can create multi-page documents. It can insert recognized pages that have been processed using KernelAPI functions. It can also delete and move pages, and save and load the document from the proprietary OPD file type. When the document is ready, the application can export it into the output file, using the appropriate output format.

One-step functions

RecAPIPlus has two distinct one-step functions. The source code of the user contains only one call to the function that performs the whole workflow.

One of the functions uses the following 123-type workflows from OmniPage 18:

- Loading
- Image preprocessing
- Recognition
- Page and document formatting
- Exporting

It can process several multi-page image files or it can work from a scanner. During the process, it uses all settings as previously adjusted by the application.

The other function executes workflow files created by the Workflow Assistant, the Workflow Applet, the Batch Manager in OmniPage 18 or by the ITest component of the CSDK.

Layout retention output

RecAPIPlus provides complex and accurate layout retention outputs in several file formats like RTF, DOC, WordML, XLS, PDF, WP, and WAV. One-step functions also use the output converters of this module.

Different converters have different capabilities for layout retention. The following output levels are available:

- Plain Text (previously called No formatting)
- Formatted Text (previously called Retain Font and Paragraphs)
- True Page
- Flowing Page
- Spreadsheet

Not all converters can realize each level of output formatting. For example, an outputted Word document or a PDF document can use Flowing Page and True Page levels, which are very similar to the original layout. There are also simple text converters, which can retain only the text in Plain Text mode, or the text and its attributes in Formatted Text mode. In addition, final output results can be influenced by adjusting settings. For the lists of the supported formatting levels of each output converter, see the *Output Converters and Output Levels* in *RecAPI Help*.

Chapter 5

IPRO

ComKit Libraries

The ComKit 21 libraries are made to support programmers of COM-enabled environments. They replace the IPRO libraries of the version 12 SDK. The new version is tightly integrated with the latest CSDK Engine and is based on the OmniPage Ultimate architecture. The version 21 object model is largely backward compatible with that of version 16. As the CSDK OCR engine underwent a major revision between versions 12 and 15, the current object model is not backward compatible beyond version 16.

This document summarizes the COM object hierarchy of the SDK, and the interface details for the specific objects.

Since CSDK 15, this toolkit does not support Windows 9x platforms including Windows 95, Windows 98, Windows 98 SE and Windows Millennium editions. These OS platforms cannot be used as development platforms or runtime environments.

Potential deadlock situations

The IPRO is a highly multi-threaded architecture. It boosts performance on multi-processor, multi-core and multi-threaded systems. However, because it also massively overlays the Windows messaging architecture, that has limitations in multi-threaded environments, it can cause deadlock situations. You can find a set of hints below about how to avoid deadlock scenarios when using IPRO, and optionally the MediumWeight Visuals.

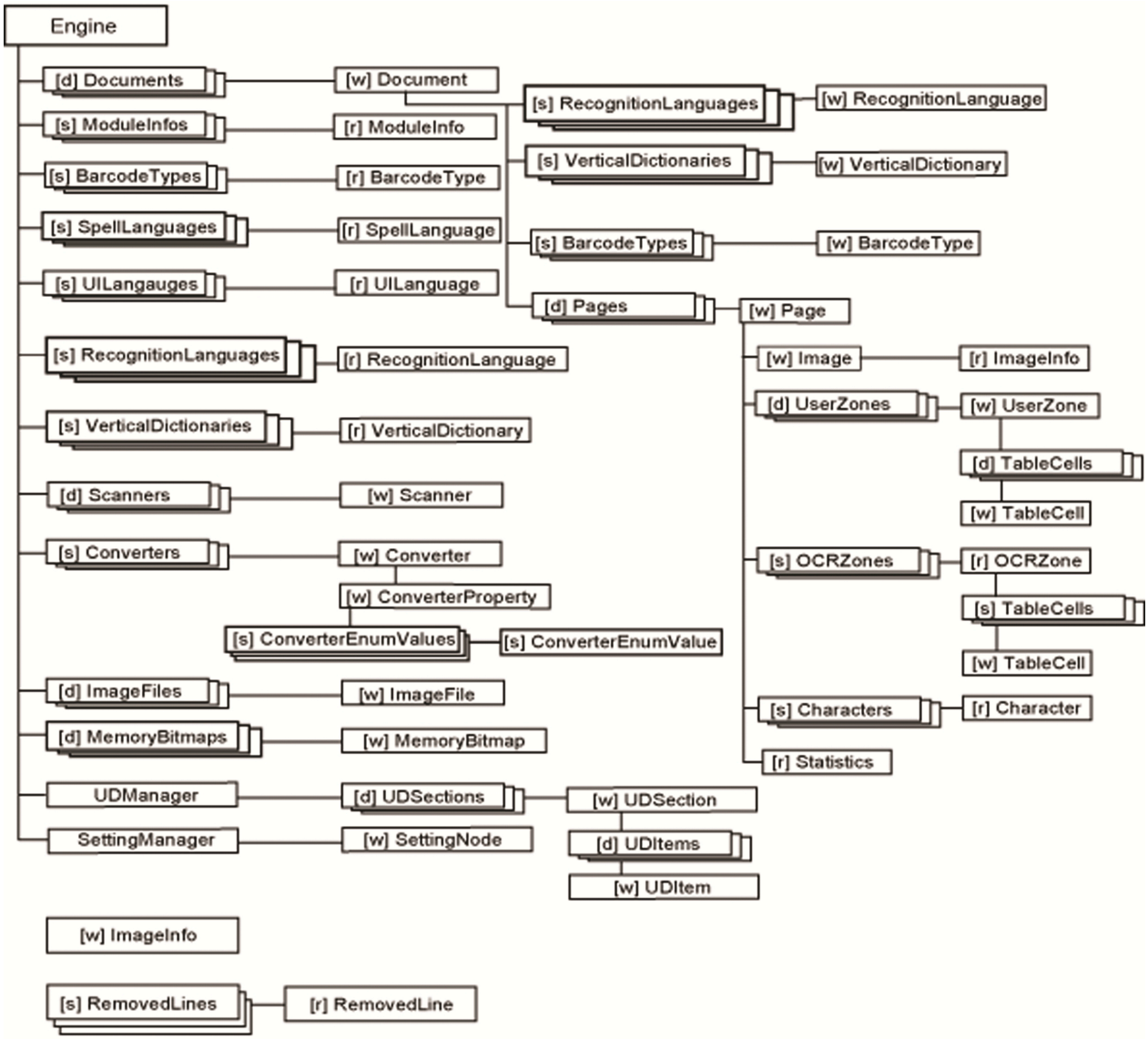
How to use IPRO and the MediumWeight Visuals in a safe way:

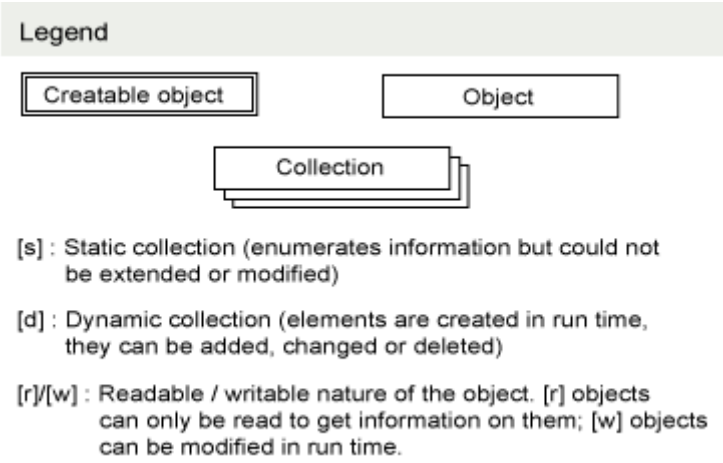
- Avoid calling back to IPRO in event handlers. Many IPRO events receive IPRO object references. If it is really important, use the received object reference to query basic properties, but never initiate massive processing which causes hard IPRO usage or additional event firing, even in different threading context.
- Avoid calling the SendMessage API targeting windows that were created in different threads. Never send messages from a worker thread to a window, which was created in the main UI thread. This is a typical reason of deadlocks in multi-threaded windows applications even without any IPRO, or other COM object usage.
- Avoid accessing objects that were created in a particular thread from a different thread. If possible, encapsulate all steps to create and process IPRO related objects into a single thread, or ensure proper interface marshalling if multiple thread usage is necessary and you need to access objects in a thread different than the creator.

- Ensure that during an IPRO call the program does not enter into another parallel call. This can happen, for example, if you have a UI button associated with some process and click it a second time before the process initiated by the first click has finished. The IPRO Workflow manager has its own message loop to enable the UI of the integrator application to be responsive and painted correctly, so this type of encapsulation is possible. Disable the related UI elements during any processing so that the user cannot click the button at the wrong time even if the interface is responsive.

IPRO object model

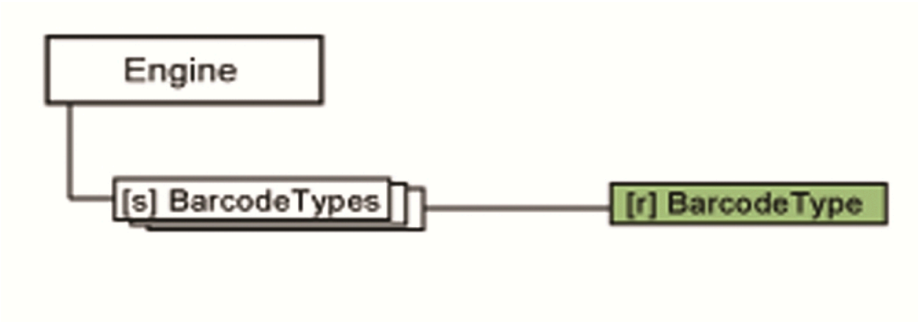
The IPRO object structure is a complex, multifunction hierarchical object model which is detailed in this documentation. Below is a comprehensive reference diagram.





Object summaries

BarcodeType / BarcodeTypes

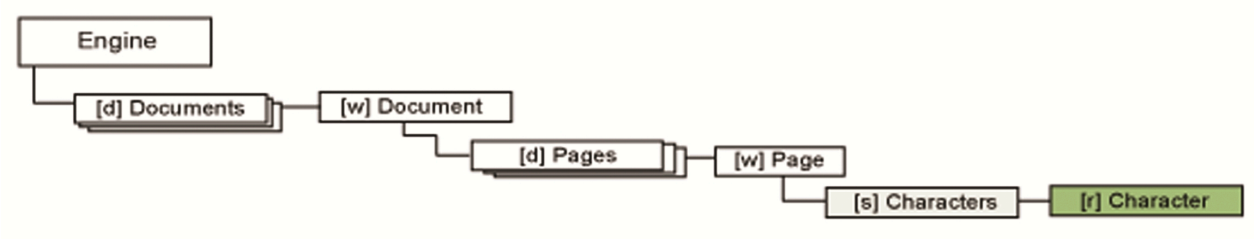


The BarcodeType object contains information about the particular barcode type.

The BarcodeTypes collection is a sorted set of BarcodeType objects. Each element represents a recognizable bar code type. Using its properties, the BarcodeTypes collection enables you to gather information about the supported bar code types.

This collection serves multiple purposes. Under the Engine objects it provides barcode type information for listing purposes. If the parent of this collection is a Document object, the contained barcodes can be enabled or disabled for recognition. By default the following bar code types are enabled: EAN, ITF, Code 39, Code 128 and Codabar. For the list of supported barcode types, see the *RecAPI Help*.

Character / Characters

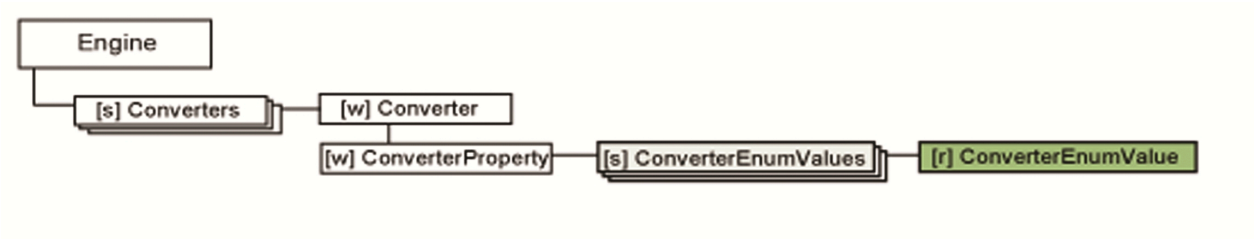


The Character object provides information about a single recognized character, such as the character code, coordinates of the character on the image, and confidence information. The character coordinates are treated as OCR image coordinates.

The Characters collection is a sorted set of Character objects, where each element represents a single character in the recognition result. By accessing its properties, the Characters collection enables you to gather information about the recognition result. The collection is sorted by the OCR zone indices (OCRZoneIndex property).

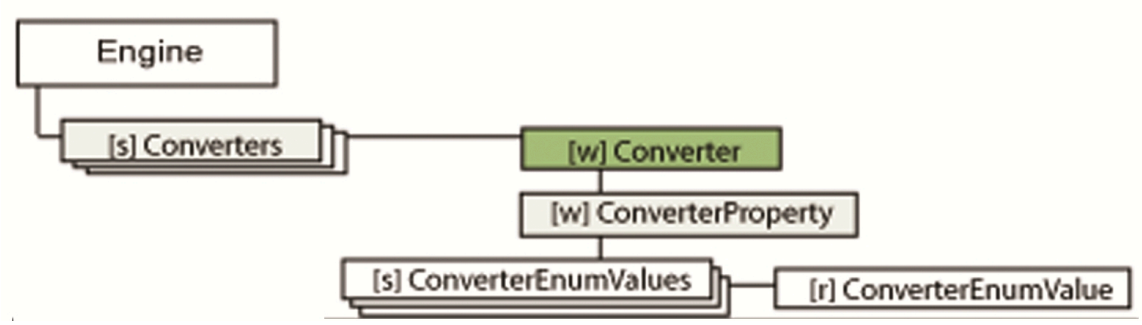
By default, this collection is empty. After the Recognize method of the Document or Page object is called, it is filled with the recognition result. The returned Characters can be filtered by zone index and cell number.

ConverterEnumValue / ConverterEnumValues



The ConverterEnumValue object represents an enum value of a ConverterEnumValues collection. The ConverterEnumValues collection contains the available enum values of a ConverterProperty object.

Converter / Converters



The Converter object provides properties and methods to customize the layout of the output files created by the ConvertResult method of the Document and Page objects. The converter settings are organized into a tree-structure, and they are available through the ConverterProperty object, which represent a setting node in the tree.

There are the following converter types:

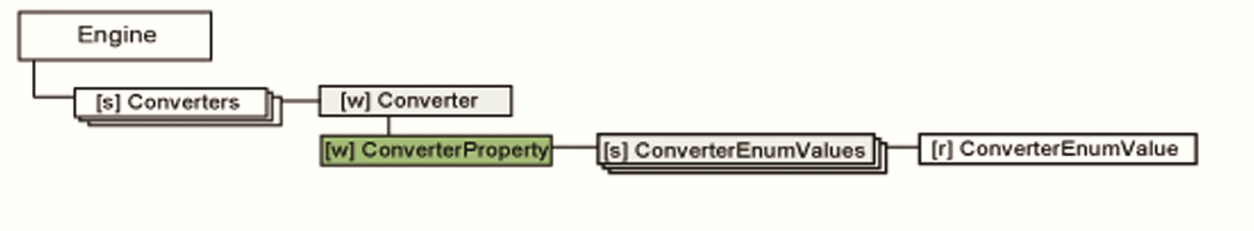
- Text converters create a formatted or unformatted text file from the recognition results.
- Image converters create image files from the input images, so they do not require the input pages to be recognized.
- Multiple converters are a combination of text and image converters and they create multiple output files when exporting.
- Form converters are used in form data extraction.

There are multiple possibilities to set converter parameters. Use ConverterProperty to customize them. To query them, use the Item method of the Converters collection. Converter names and parameter setting possibilities are detailed in the RecAPI documentation under List of all Converter settings of RecAPI.

The Converters collection is a sorted set of output Converter objects. Using its properties, the Converters collection enables retrieving information about the supported converters.

A converter can be cloned using the CloneConverter method. It makes an exact copy of the source converter and is useful to make customized converters for the same output format. The cloned converters can later be deleted using the RemoveClonedConverter method if no longer needed. Factory converters cannot be deleted.

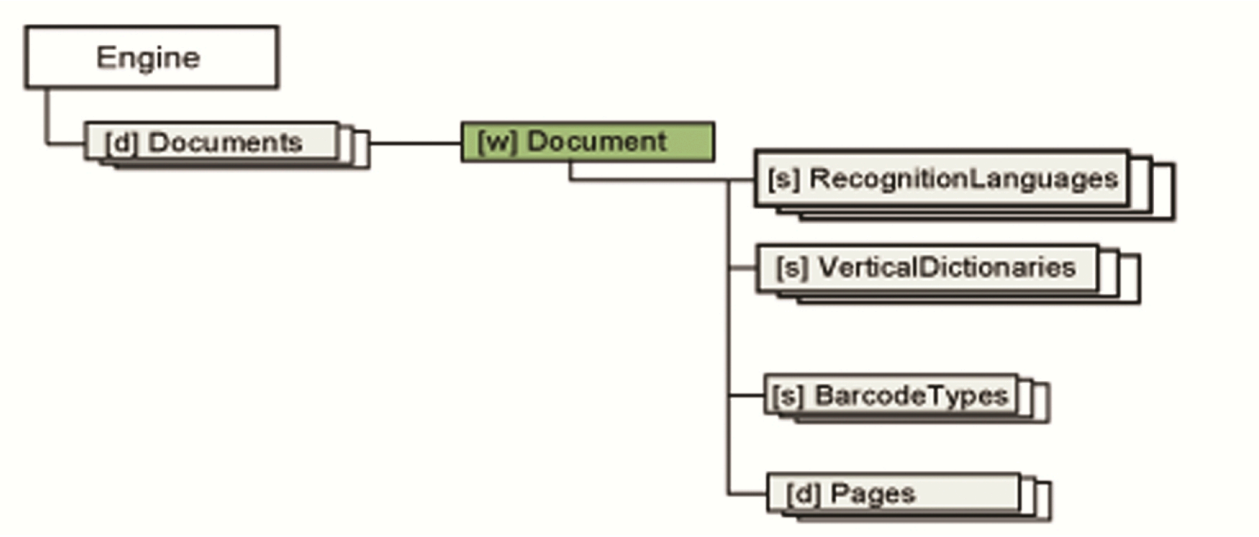
ConverterProperty



This object represents a single converter setting. The converter settings are represented by a hierarchical tree structure, isolating the different setting groups, so that each tree node represents a converter setting which can have an associated value and can have children settings. To read, or manipulate a setting, connecting to a particular ConverterProperty object, use Name, FullName, DisplayName, Type, Value and EnumValues properties.

Note EnumValues returns a valid object only if the related setting is of type PROPERTY_ENUM and it can be used to query the valid enum values the particular property supports. In order to access the children settings of the object, use the Count and the Item properties.

Document / Documents

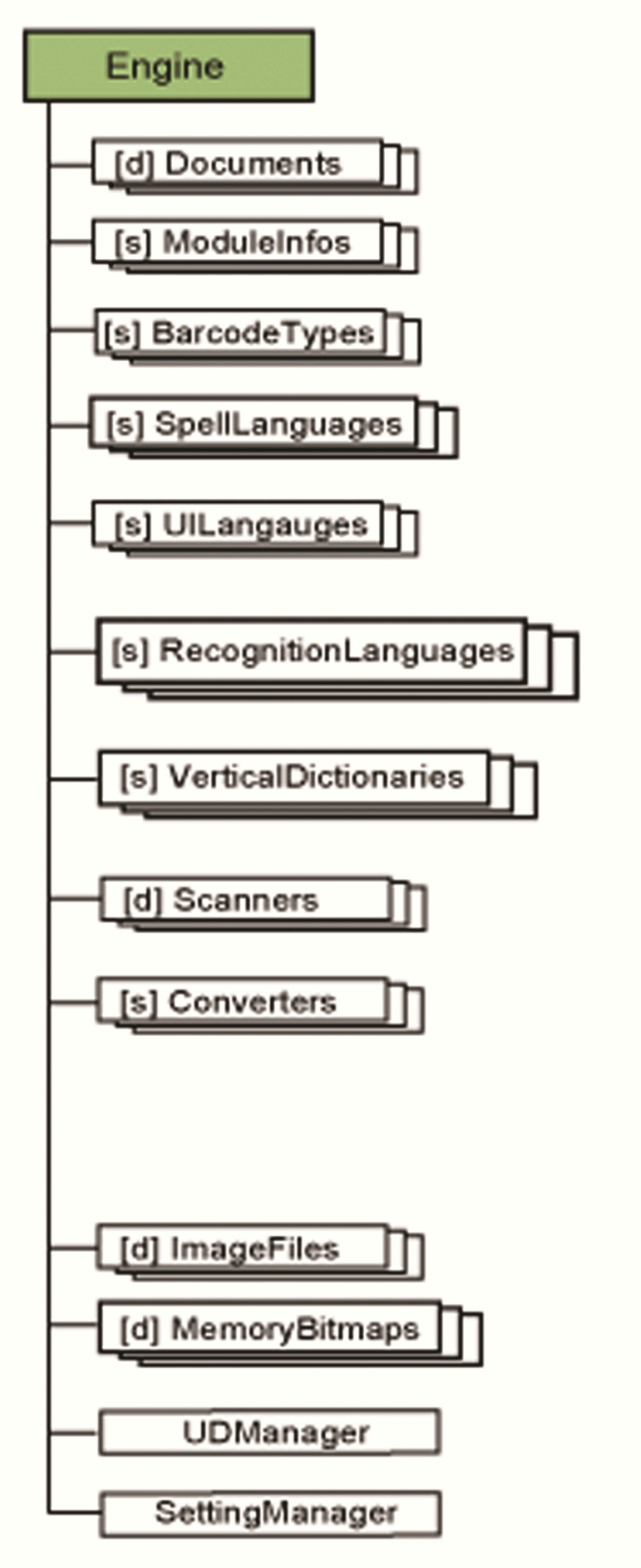


The Document object represents an IPRO document which encapsulates the data and the parameters required during OCR related procedures. To create a new document, invoke the Add method of the Documents collection. To open an existing document, invoke the Open method. To save a document for later reference, invoke the Save method. The Document object has numerous properties and settings. You can save or load all settings by using the SaveSettings and LoadSettings methods.

Note The Save method also saves all document level settings to the document file automatically.

The Documents collection is a sorted set of Document objects, where each element represents a document handled by the Engine. Through its properties, the Documents collection returns information about the currently open IPRO documents.

Engine


Engine
[d] Documents
[s] ModuleInfos
[s] BarcodeTypes
[s] SpellLanguages
[s] UILangauges
[s] RecognitionLanguages
[s] VerticalDictionaries
[d] Scanners
[s] Converters
[d] ImageFiles
[d] MemoryBitmaps
UDManager
SettingManager

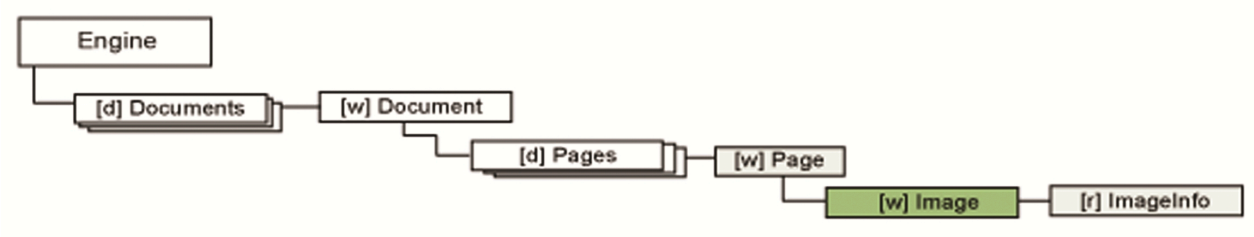
The Engine object represents the main OCR Engine class which serves many purposes such as scanning, image preprocessing, recognition, text formatting and output.

This object and its functions are referred to as the Engine.

The Engine object is the top of the object hierarchy and controls all other objects in the object hierarchy. The integrating application must always explicitly create an instance of the Engine object. When it is created, the Engine object analyzes the configuration of available sub-modules and performs initialization. The valid licensing of the required modules at this time is vital, otherwise either the Engine object creation fails, or one or more modules are not available.

The Engine must be initialized with the Init method. All properties and methods return error code if the proper initialization is not done.

Image



The Image object stores the original page image either scanned, loaded from image file or memory, together with the engine-managed additional, generated images. The generated images are the following:

- Primary image: the result of the image preprocessing and image manipulation, that occurs on the original image.
- OCR image: a black-and-white image created by the first operation that requires it, for example preprocessing, page-parsing, or recognition. The OCR image is only available after the recognition.
- Thumbnail image: a smaller version of the current image.

The role of this object is to facilitate access to the managed images, and to provide a basic operation set for image manipulation. A large image manipulation library is available through the ImageFile object.

Redaction is a special case for image handling. This operation writes back to primary, OCR and original images stored in the program. Only the original image, stored in file outside the program, remains untouched. In the case of scanning, no unredacted image is available. Therefore we suggest that you apply redaction to a copy.

ImageFile / ImageFiles

The ImageFile object represents an image file in the file system. Use this object to get information, manipulate, or convert image files using a wide set of image manipulation methods. Most image-processing methods are available only for particular multi-page TIFF images. To process other images using the available methods, save them in TIFF format first, perform the operations and convert them back to the original format.

The supported image resolution is in the range of 75-2400 DPI. When accessing bitmap pixel data, note that every stored image is stored in bottom-up line order and the pixel order is BGR.

The ImageFiles collection contains the list of the available ImageFile objects. Using this collection, the available images can be loaded, enumerated, or closed.

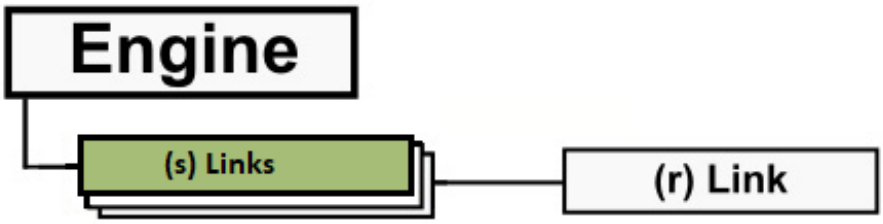
ImageInfo

The ImageInfo object provides basic image information about the page image, like size and resolution. Although this creatable object has read/write properties, these properties cannot be used to change the properties of an existing image or image file.

Images in IPRO are always retrieved in the following format:

- Byte order: BGR
- Line order: bottom-up
- Line padding: double word (4 bytes) aligned

Link / Links



The role of the Link object is to provide workflows with loading sources and saving targets that are not present on the local computer. These links to file sources and destinations include FTP locations and document management systems, such as different versions of SharePoint and ODMA.

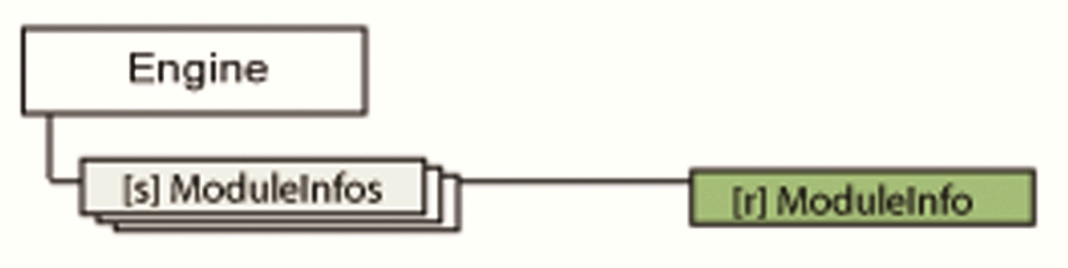
The Links collection contains the list of the available Link objects.

MemoryBitmap / MemoryBitmaps

The MemoryBitmap object represents an image stored in the memory. This object manages bitmaps as raw data. This object can be passed to the LoadBitmap method of the Document object. This allows the integrating code to load an image which is stored in the memory into a particular document. All bitmaps stored by the IPRO object model are bottom-up, BGR color order, and double-word aligned bitmaps.

The MemoryBitmaps collection is a sorted set of MemoryBitmap objects. Using this collection, bitmaps stored in memory can be created and deleted.

ModuleInfo / ModuleInfos

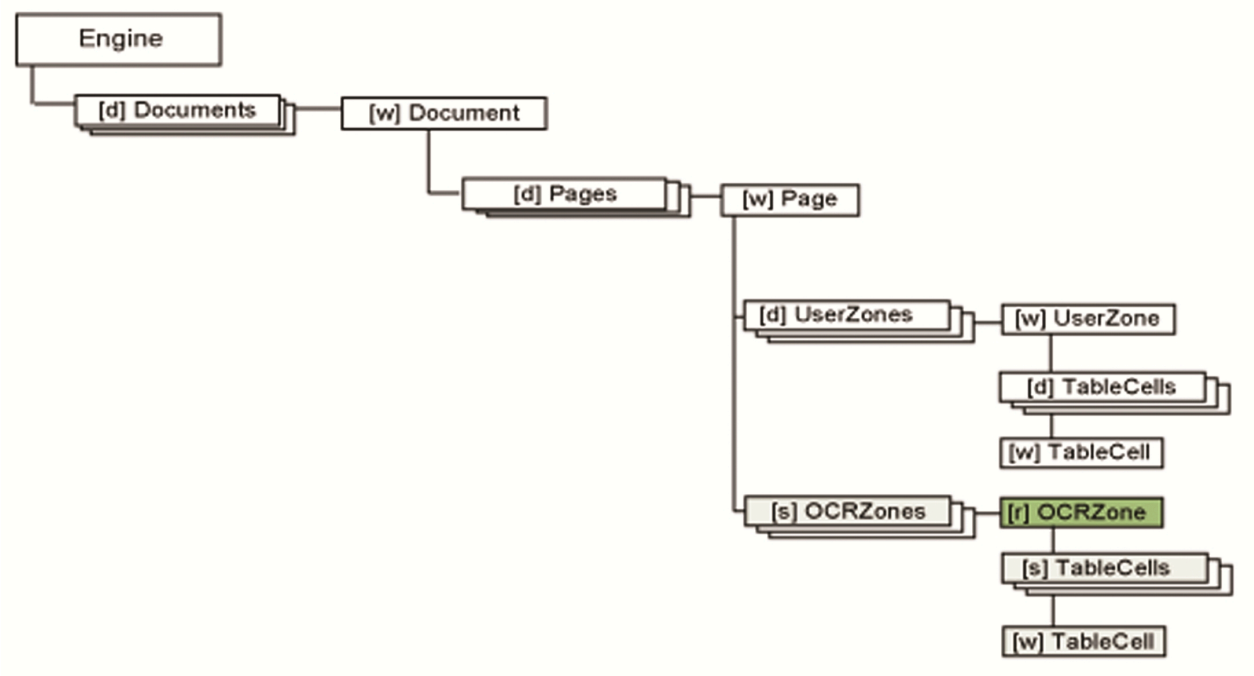


The ModuleInfo object contains information about a specific Engine recognition module even if the particular module is not available because of the lack of the appropriate licensing. Use the Version property to decide whether the module is available or not. Use the InitCode property to determine the reason if the module is unavailable.

The ModuleInfos collection is a sorted set of ModuleInfo objects. Each element in the collection represents an Engine module. Using these objects you can retrieve information about the available Engine modules.

Some modules must always be distributed with the integrating application, since their existence is a prerequisite of any Engine operation. Others, such as some recognition modules, the scanning module, and so on, must be distributed only if the integrating application requires their services. This flexibility allows the integrator to scale the capability of the distribution file set for disk footprint sensitive applications. This collection can be used to log diagnostic information that helps solving problems. Diagnostic information can provide users with run time feedback, when an application is designed to optionally use a variable set of Engine modules.

OCRZone / OCRZones

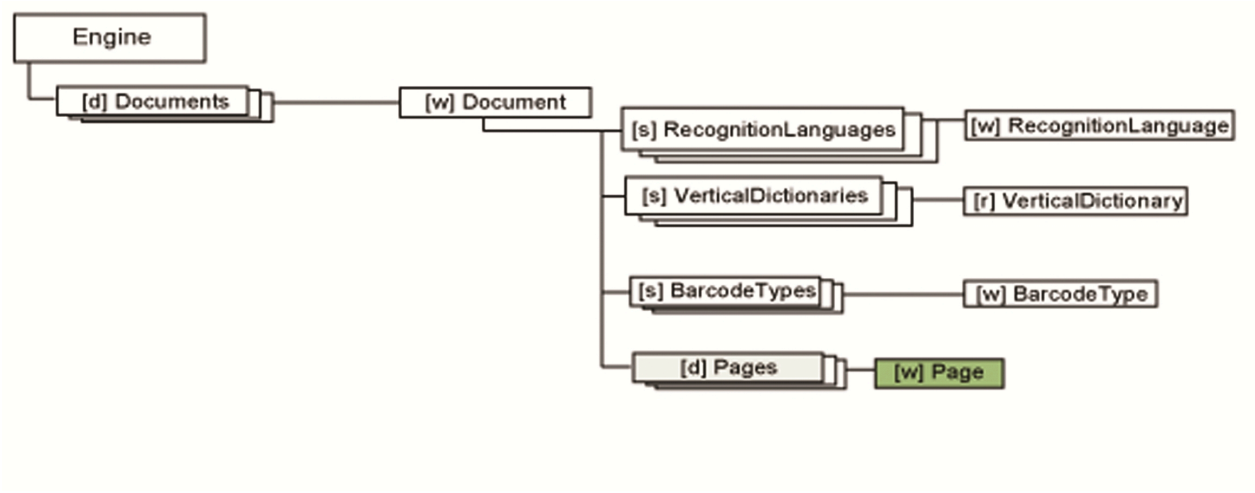


The OCRZone object contains information about a specific OCR zone in a page. The OCRZone object provides access to the coordinates of an OCR zone, its content type, and so on. All coordinates in this object refer to the OCR image.

The OCRZones collection is a sorted set of OCRZone objects. The order of the OCRZone objects are the reading order of the recognized paragraphs.

Note The reading order takes effect only when it is exported with converters using the Formatted Text formatting level, formerly called Retain Fonts and Paragraphs mode.

Page / Pages



The Page object represents a document page. In addition to a set of processing operations, you can use the Page object to access zone, character, statistical and image objects. The UserZones collection maintains the zone set available for modification and it provides the input zone set for the recognition. The OCRZones collection is created by decomposition and recognition processes, the OCR recognized character results are assigned to the OCR-zones.

After processing methods, such as LocateZones and Recognize are run, the UserZone collection is synchronized with the OCRZones collection. The OCRZone objects are read-only.

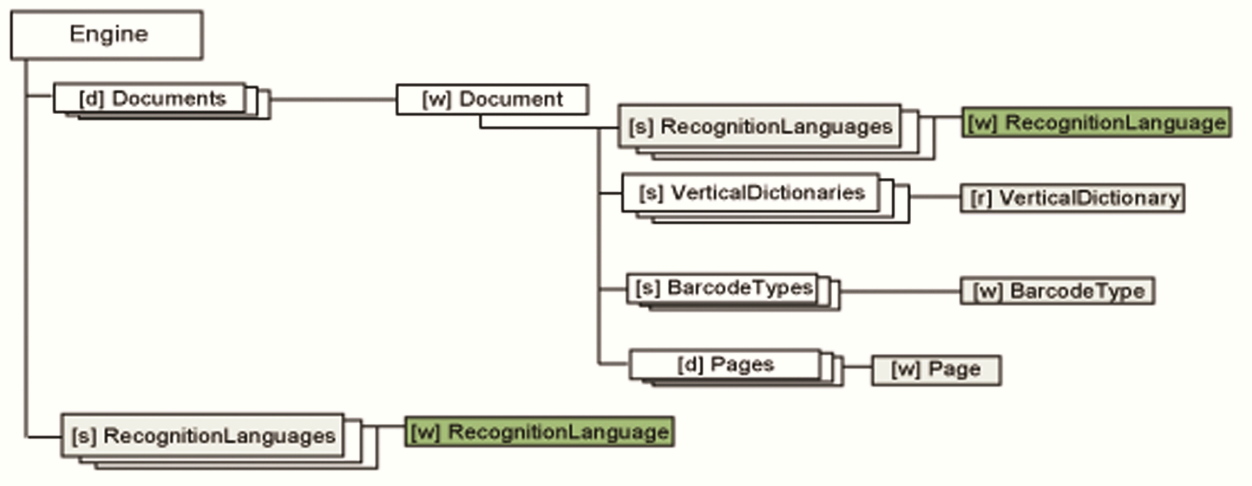
The Image object provides access to a set of image versions maintained by the engine, such as the original, the b/w and the thumbnail images.

The Characters collection provides access to the whole recognized and optionally filtered character set.

The Statistics object provides details about the recognition processes.

The Pages collection is a sorted set of Page objects, where each element represents a page in the document. Through its properties and methods, the Pages collection enables you to gather information about the document pages.

RecognitionLanguage / RecognitionLanguages



The RecognitionLanguage object contains information about the particular recognition language.

The RecognitionLanguages collection returns the collection of available recognition languages that can be switched on and off independently. The set of recognizable characters can be expanded by using the LanguagePlus property, and can be restricted by using the Filter property. By default, only the English language is used for recognition.

The Recognize method of the Document or Page objects use this setting.

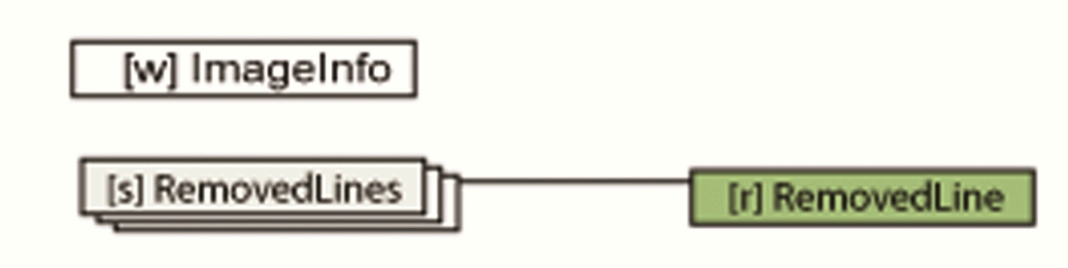
When the Asian OCR add-on is purchased, this language collection includes Traditional Chinese, Simplified Chinese, Japanese and Korean.

When the Asian Plus OCR add-on is purchased, this language collection includes Thai, Arabic, and Hebrew beside the Asian OCR languages.

The following restrictions apply for the languages of Asian Plus:

- Each language has to be set alone, or only with English
- Orientation detection is not supported
- Arabic recognizes English words only when English language is also set,
- Arabic does not support deskew detection,
- Arabic RecAPIPlus output converters are supported from version 20.1.

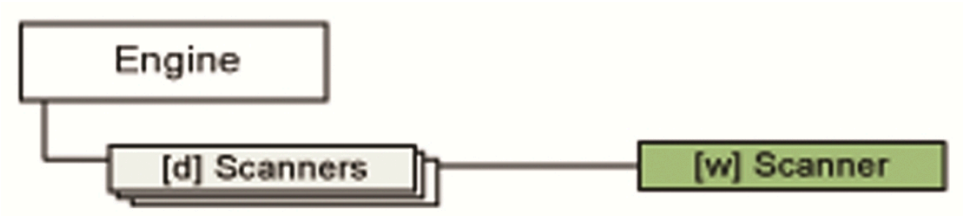
RemovedLine / RemovedLines



The RemovedLine object represents a line, which was removed from the BW version of the original image. Through this object, the position, and the orientation of the line can be accessed.

The RemovedLines collection is a set of RemovedLine objects representing the lines that were removed from the related BW image. The RemoveLines method of the Page object returns this collection.

Scanner / Scanners



Scanner objects represent a physical scanner. The Name property uniquely identifies the scanner. Prior to invoking scanner methods, or accessing scanner properties, the particular scanner must be initialized by calling the Init method. Depending on the scanner model, this may take several seconds. Each scanner manages its scanning properties individually. To load or save a scanning profile using external files, invoke the LoadSettings and SaveSettings methods.

To scan images into image files, use the ScanPages method. To scan pages directly into a particular document, pass the Scanner object to the ScanImage method of the Document object.

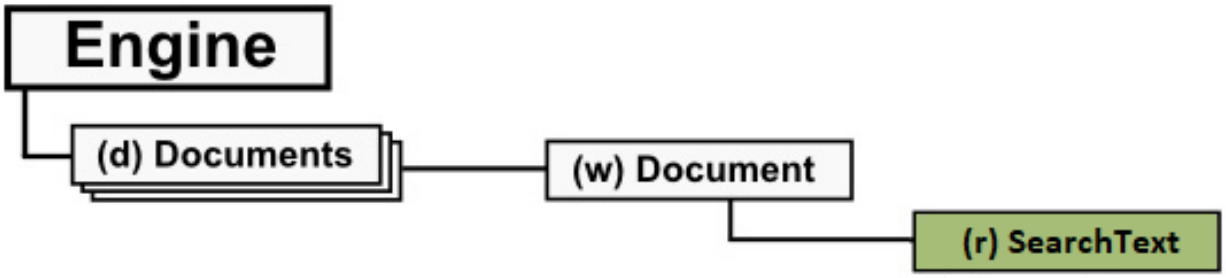
The Scanners collection lists the set of available Scanner objects. Each element represents a physical scanner. The scanner list is loaded from a .ini file. The .ini file can be altered with the Scanner Wizard. To run the wizard, invoke the ScannerWizard method. You can add, remove and parameterize the available scanners with the wizard. By default, the .ini file is called scanner.ini and stored in the application data of the user. The scanner.ini file can optionally be saved to the `All Users` application data folder, or into the `application` binary folder. To change the .ini file location, update the ConfigFileLocation property. If the ConfigFileLocation property changes, the actual scanner list is removed, and the new scanner.ini file is loaded. Use the SaveSettings and LoadSettings methods to save or restore scanner parameters, like resolution or color mode, stored in the external setting file.

During scanning, the scanner drivers may show their own progress dialogs. To specify a parent window for these dialog boxes, set the WindowHandle property of the Engine object accordingly. Providing a valid

window handle is highly recommended to avoid a window or message being displayed behind the main application window. The WindowHandle property should be set before initializing the scanning subsystem. If the parent window is subject to change after you initialized the scanning system, you need to re-initialize the scanning, so the new setting takes effect. To do so, update the value of the ConfigFileLocation property.

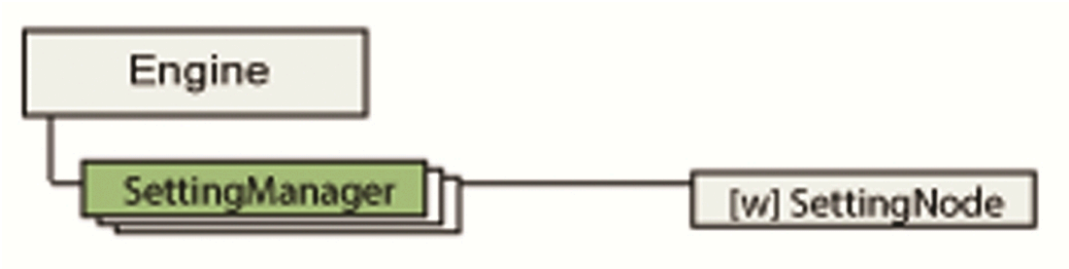
Note Changing the ConfigFileLocation property resets all scanner properties to their default value.

SearchText



The role of the SearchText object is to allow you to carry out parameterized text searches, such as finding whole words only, searching with case matching, searching from the beginning or the end of the document. Navigate through your results and process them using the text marking and redaction facilities.

SettingManager



The SettingManager object allows you to manage the settings of the Engine and the Documents. Settings are named values which control the behavior of different functions on the lower levels of the CSDK. They are organized in a tree-structure. The setting tree contains a large number of settings, but you should use the SettingManager only to manipulate those that are documented in the RecAPI online help.

In a setting tree, every setting is attached to a node. The nodes do not necessarily have associated setting values. For example, some nodes might be created to act as the parent of a set of child nodes having associated setting values. Individual settings are identified by their unique path name, which is the concatenated names of the nodes from the root node to the setting node. The node names are separated by dots.

Several setting trees can co-exist at a time. Upon its creation, the Engine creates the first setting tree. All other setting trees are created as copies of the current settings of the Engine. Each newly created

Document receives a copy of the Engine settings. This way the settings of the individual documents are independent from each other and from the settings of the Engine.

When a document is saved to an OPD file, the related Document settings are saved automatically. However, you should explicitly save the Engine settings if needed.

Scanner and Converter object settings present special cases. Although they do appear in setting trees, several of their individual setting values may depend on others. Therefore we suggest that you work with Scanner and Converter object settings through these objects themselves, and not through the SettingManager.

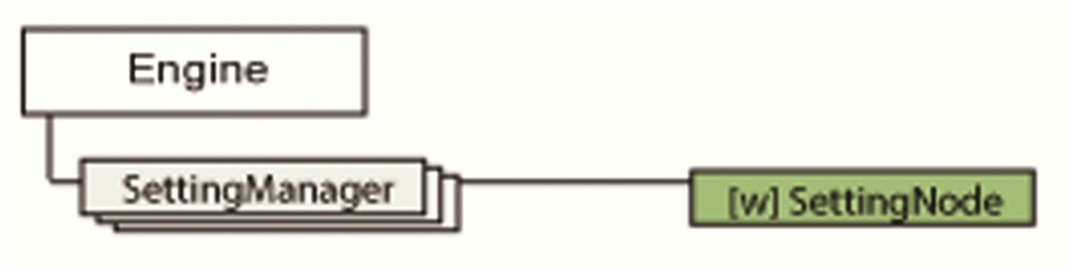
Scanner and Converter objects can be queried only from the Engine. These objects modify the settings of the Engine. To use document-specific Scanner and Converter settings, set SP_EXP_USEENGINECONVERTERS and SP_EXP_USEENGINESSCANNERS workflow step parameters to false.

The setting manager supports the following setting types:

- Integer
- Floating point
- Boolean
- String
- Integer array
- Floating point array
- Boolean array
- Enumerated

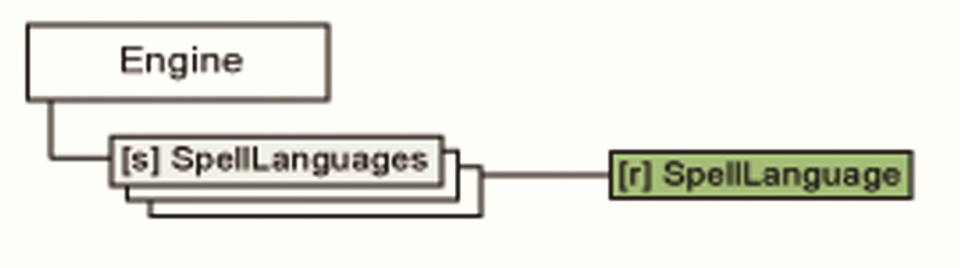
Saving and loading the entire setting tree, or an arbitrary sub-tree is also supported.

SettingNode



The SettingNode object provides access to a particular node and its setting if it exists. You can use it to get or set the value of a setting, retrieve information about setting flags and the default value, and to enumerate the child nodes.

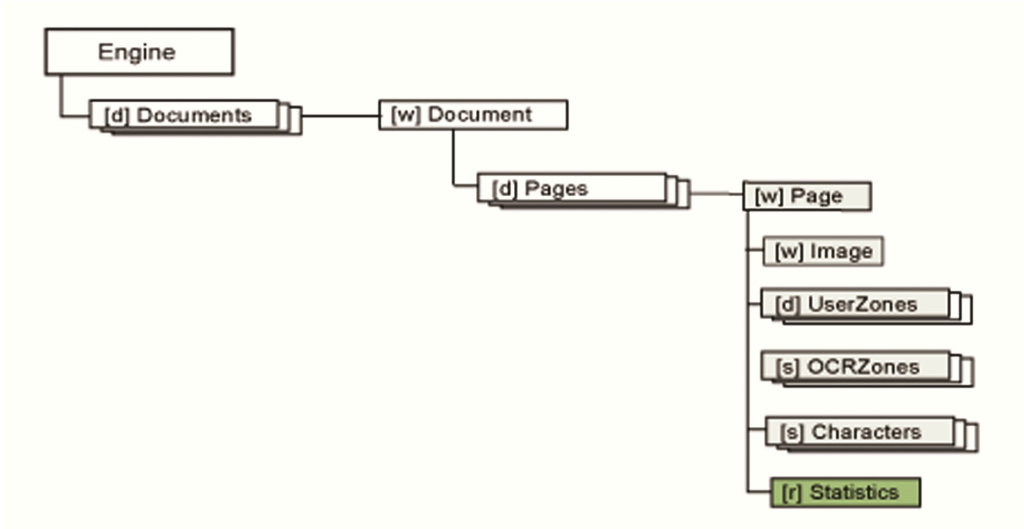
SpellLanguage / SpellLanguages



The SpellLanguage object contains information about a specific spelling language.

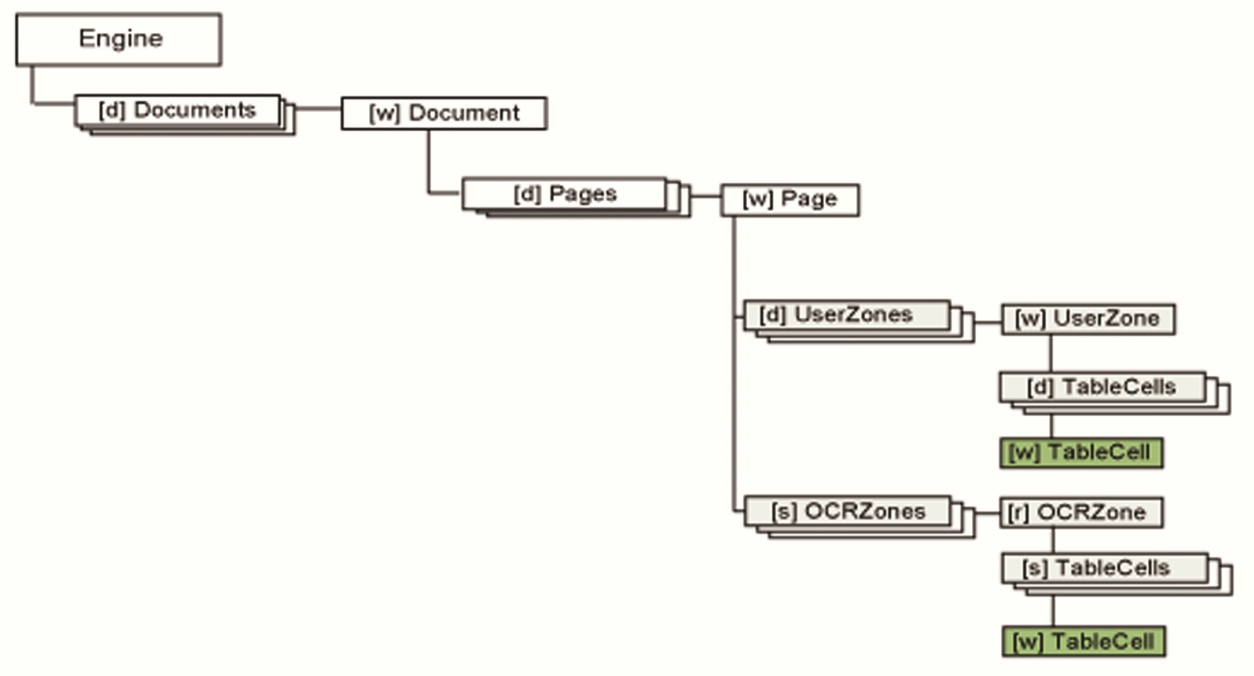
The SpellLanguages collection is a sorted set of SpellLanguage objects, where each item represents an available spelling language. It means the Language dictionary is available in the current engine configuration. Through its properties, the SpellLanguages collection enables you to retrieve information about the installed spelling languages.

Statistics



The Statistics object provides statistical information about the processing of the page. It includes general, accuracy and timing information. If the page has not been recognized, all properties of the related Statistics object return zero (0). The Statistics object does not reflect the time spent in the Progress or ProgressEx events fired by Document object.

TableCell / TableCells

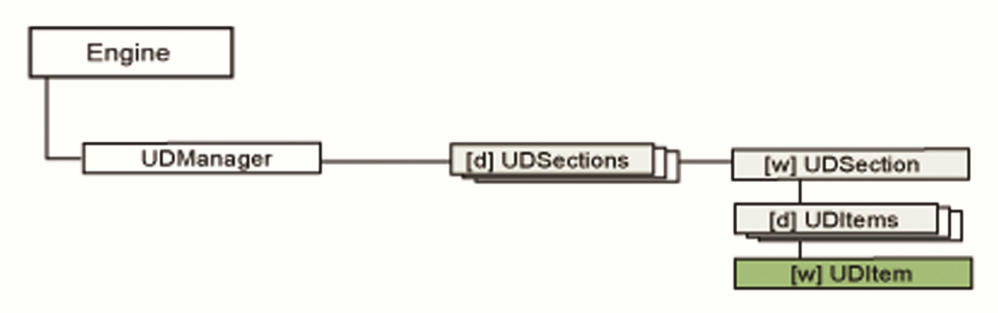


The TableCell object represents a single cell in a table.

The TableCells collection is a sorted set of TableCell objects.

If the parent of this collection is a UserZone object, the content of the collection can be modified. If the parent of this collection is an OCRZone object, the content of the collection is available for enumeration in read-only mode, but cannot be modified. The members of the collection are sorted by the top and left coordinates.

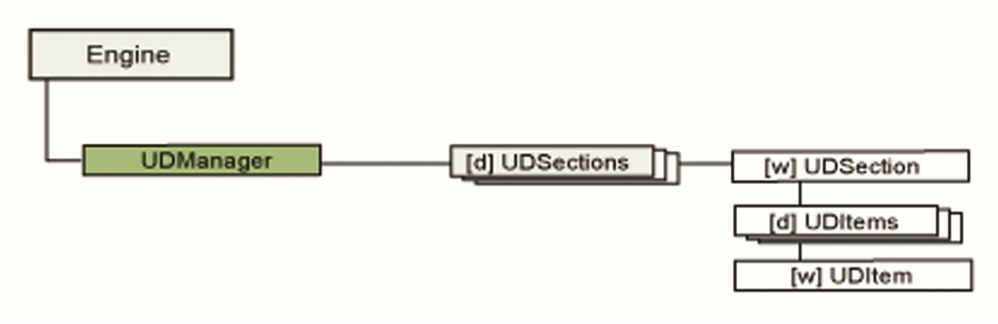
UDItem / UDItems



The UDItem object contains information about a specific user dictionary.

The UDItems collection is a sorted list of the UDItem objects. These objects can be string literals or regular expressions. Through its properties, the UDItems collection enables you to retrieve information about the underlying UDItem objects. The string literals contained by the collection are sorted alphabetically and followed by the regular expressions.

UDManager



The UDManager object is used for creating and maintaining user dictionaries. Only a single user dictionary can be opened for maintenance at a time. This user dictionary is referred to as active.

Each user dictionary contains a number of UDItems, which can be either a literal string or a regular expression. Each valid dictionary must contain at least one UDItem object.

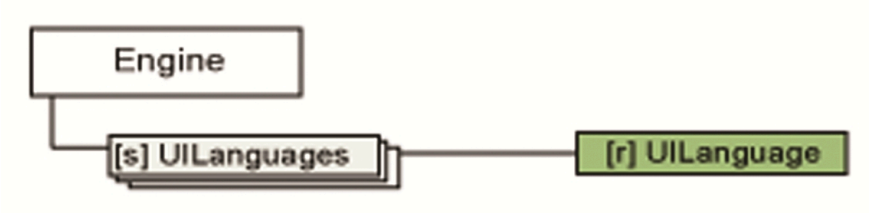
User dictionaries are used with the SetUserDictionary method of the Document object. Make sure that the checking subsystem is enabled.

Note The checking subsystem can only use a single user dictionary, optionally with more than one sections. A particular section can be specified while setting a user dictionary for a document or a zone.

UDSection / UDSections

The UDSection object is deprecated since CSDK 20.1.

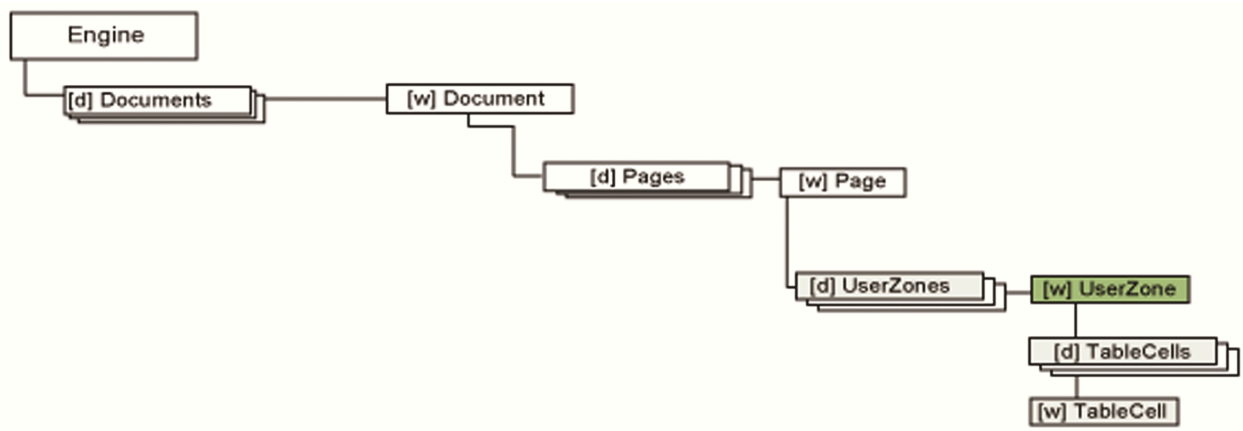
UILanguage / UILanguages



The UILanguage object contains information about a specific Visual Toolbox User Interface language.

The UILanguages collection is a sorted set of the available UILanguage objects. Each member item represents an available Visual Toolbox User Interface language, currently English, French, and German languages are available. Through its properties, the UILanguages collection enables you to retrieve information about the available user interface languages. This property primarily controls the Visual Toolbox UI language, although it also sets the IPRO related UI language for the ScannerWizard.

UserZone / UserZones



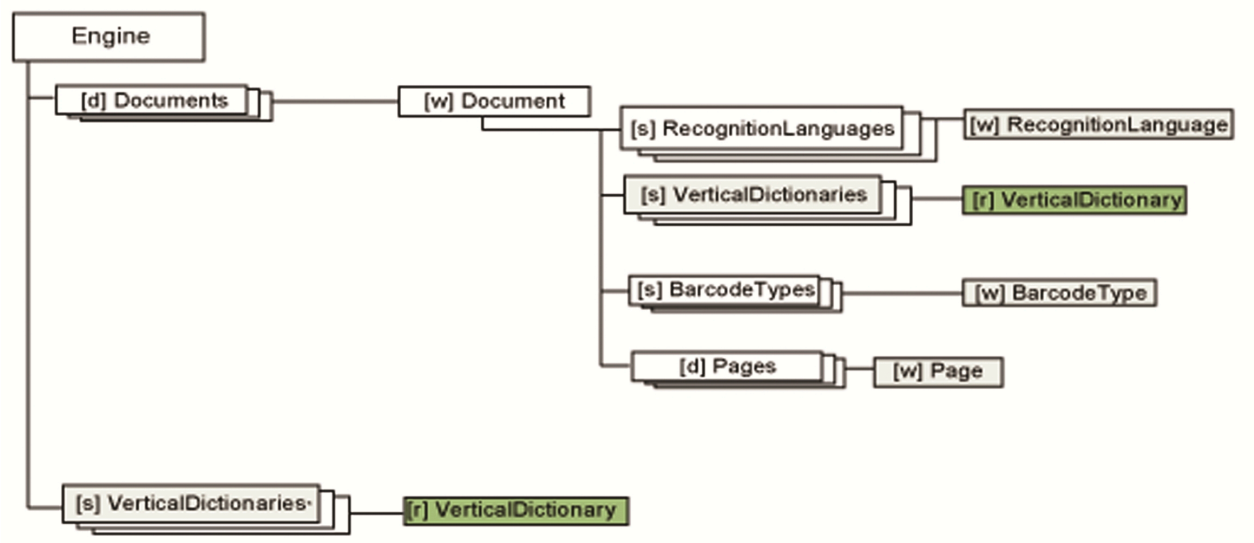
The UserZone object contains information about a particular user zone on the page. The UserZone object provides access to the coordinates of a user zone, its content type, and so on. All coordinates of this object refer to the original image.

The UserZones collection is a sorted set of UserZone objects, where each element represents a user zone on the related page. Through its properties, the UserZones collection enables you to access and modify the user zone set which provides input for recognition.

After zone location or recognition, the OCRZones collection is created and the UserZones collection is synchronized accordingly.

Note In processing methods that alter the user zone set, the collection is stored automatically for later reference. Use the RestoreUserZones method to recall the original UserZones set.

VerticalDictionary / Vertical Dictionaries



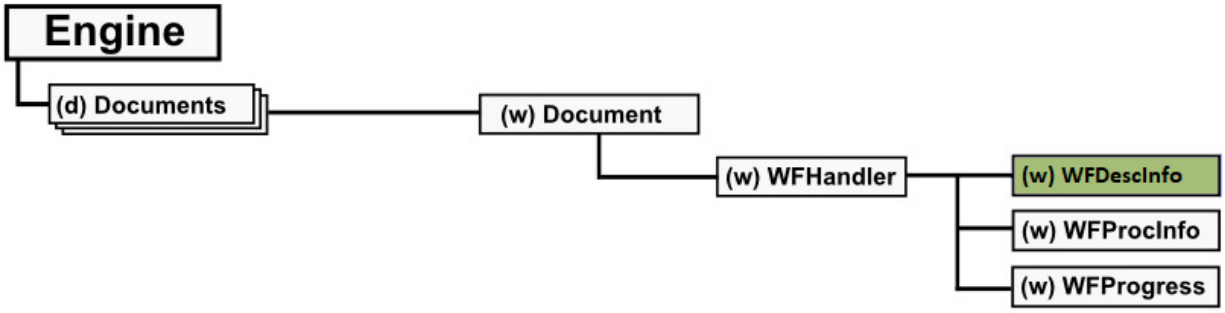
The VerticalDictionary object contains the name and the description of the particular vertical dictionary. Vertical dictionaries are alternatively called professional dictionaries. Each vertical dictionary can either be disabled or enabled, if the parent of the containing VerticalDictionaries collection is a Document object.

CSDK supports the following vertical dictionary set:

- Dutch Legal Professional Dictionary
- Dutch Medical Professional Dictionary
- English Financial Professional Dictionary
- English Legal Professional Dictionary
- English Medical Professional Dictionary
- French Legal Professional Dictionary
- French Medical Professional Dictionary
- German Legal Professional Dictionary
- German Medical Professional Dictionary

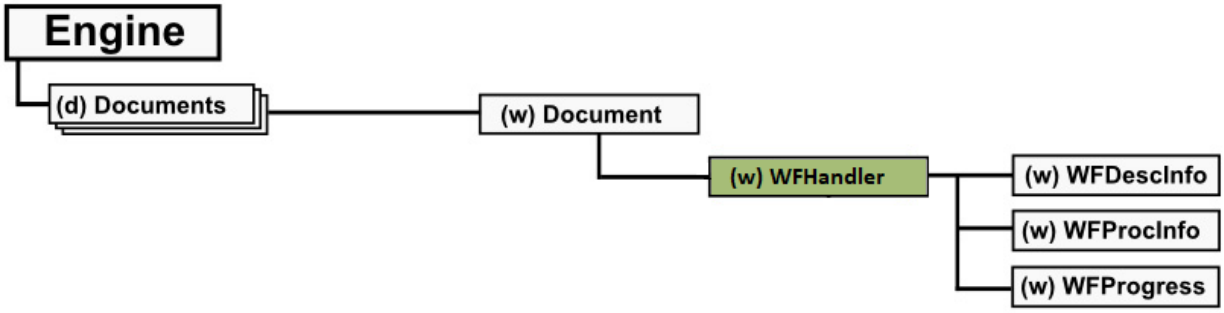
The VerticalDictionaries collection is a sorted set of the available VerticalDictionary objects. This collection can be accessed through the Engine and Document objects. Through the Engine object, you can only query the available dictionary set and its properties. Through the Document object, you can enable or disable an arbitrary combination of the available vertical dictionaries.

WFDescInfo



The WFDescInfo object is provided by the WFHandler object. It is used to acquire static information about the current workflow. Static information remains during the workflow.

WFHandler



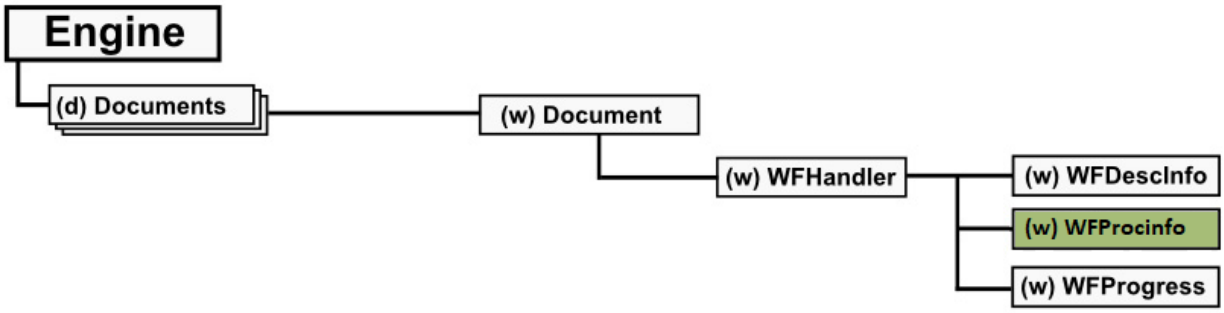
The WFHandler object contains a wide range of workflow handling related functionality. Its main purposes are:

- Running existing workflows created with the Workflow Assistant
- Programmatically compiling and running workflows.

In either case, you can specify parameters for individual workflow steps. In addition to the traditional steps, like loading, scanning, recognizing, and exporting, the current application version offers new ones, such as Form Data Extraction and Text Marking, including redaction.

To query dynamic and static information, you can also use WFProcInfo and WFDescInfo provided by the WFHandler object.

WFProcInfo



The WFProcInfo object is provided by the WFHandler object. It is used to acquire various kinds of dynamic information about the currently running workflow. Dynamic information changes during the workflow, for example the execution state.

Chapter 6

Introduction to Visual Toolbox

This help system provides a description of the Visual Toolbox, also known as Visuals, and a recently developed set of ActiveX controls named MediumWeightVisuals. These provide all visual controls necessary to create customized user interfaces for Windows applications using the IPRO interfaces. The IPRO provides invisible imaging, recognition and document management services. The ActiveX controls focus on typical user interface elements that require complex programming tasks to have the following functions:

- Display images
- Correct preprocessing and decomposition mistakes
- View and modify recognition results

The controls visualize the contents of the objects managed by IPRO, for example images, recognized texts, and so on. They provide different levels of presentation, adjusted to the underlying object hierarchy. The controls work in together with IPRO and with each other on multiple multi-page documents.

The interfaces of the Visual Controls are available in English, French and German. See UILanguage / UILanguages.

Overview

The ActiveX controls can be divided into the following main groups:

- Visuals
- MediumWeightVisuals

Legacy visual controls work only with an in-process IPRO server. MediumWeight Visuals enable the integration of and out-process IPRO server as well.

To have proper interworking, a legacy or MediumWeight control needs to be attached to the relevant IPRO object, either a Document or an Engine.

There are some issues to consider when attaching a control to an Engine or a Document:

- Controls providing document view, for example, Thumbnail and Details, can be attached only to a Document object.
- Controls attachable to both engines and documents, for example, Setting view, Recognition language view, or Dictionary view, provide information depending on what they are attached to. Controls attached to an Engine show all engine-specific information. Controls attached to a Document show document-specific information. The displayed information can differ between the engine-attached and document-attached version of the same control.

The system-level information of IPRO is the session 0, stored as a template. Each document gets a session from the current document session and there is no inter-visibility among document sessions. The sessions requested from the various engines can show values that differ from the document session values.

The default interfaces of the framework are simple, and are mostly not used. For both types of control, the main type of interface is the event interface. The event interface has two types, both providing the same tasks.

- The disp interface usually denotes an ActiveX control. It is secure, as it is always called on a UI thread, the client-side does not have to be thread-safe. It has very slow execution.
- The custom interface provides very quick action, but as the call does not necessarily arrive on a UI thread. In these cases the event can get lost.

The disp interface is the default interface offered for objects.

All controls of the program implement a number of pre-defined interfaces, thus providing a unified user interface for the various controls.

The actual visual appearance of the visual controls takes its defaults from the operating system.

MediumWeightVisuals

Control name	Default interface	CLSID
CharacterBar (former CSC)	ICharacterBar	{13E35487-6ED8-478c-9396-27B7EFDB28DE}
ConverterView (former CMC)	IConverterView	{6BE0B22B-70F9-408F-97D5-5985B07EB855}
ConvertersView	IConvertersView	{4679CD03-CF13-4C5B-A3BF-B0AC134FF90D}
DetailView (former DMC)	IDetailView	{A99D49C8-AB2C-4BBD-828B-F5E7341EFE21}
DictionaryView	IDictionaryView	{BCE24B12-5F99-4e53-ACC5-C45F0C90FED7}
DocumentView	IDocumentView	{A45578C1-B538-4461-BB29-79B6386DA1E2}
LanguageView	ILanguageView	{BB012969-EB2D-4777-922E-27F1F5D93DC8}
SettingView	ISettingView	{AD3AFC3E-6BFE-41D4-A8A2-45E6AEA44500}
ThumbnailView (former TVC)	IThumbnailView	{9A9041B0-3938-467D-9A12-66FE06BD8D78}
WFInfoView	IWFInfoView	{65C23F0A-2346-42B4-8CBA-68C650EFDDCF}
WorkflowBar	IWorkflowBar	{26872619-DBE2-4abc-B982-5D072AED9677}

Visuals

The following Visuals controls are available and supported in v18 and point releases as well with a change of CLSID:

Control name	Default interface	CLSID
Font Matching Control (FMC)	IFmc	{CD9FDAAF-5C7C-40CC-80F2-1487FF7D9EBD}

Control name	Default interface	CLSID
Image View Control (IVC)	IIvc	{1E2420A3-7BA3-4E84-84CB-03AFBE689F96}
Scanner Parameter Control (SPC)	ISpc	{462EF4BE-787E-4ED1-BEE2-EF1CD37F4863}
Text Editor Control (TEC)	ITec	{219A1D06-3B8F-4EE1-BF15-C257A2CE623C}
WorkflowControl (WFC)	IWfc	{6C3DBC07-594F-4161-A9D4-D5F1400E42F7}
Workflow Viewer Control (WFVC)	IWfvc	{4FBD86A3-A56F-4302-88EC-3BC37BBCECA8}

The following Visuals controls became deprecated from v18.0:

Control name	Functions covered by
Character Set Control (CSC)	MediumWeight Visual Control Character Bar
Converter Manager Control (CMC)	MediumWeight Visual Control ConverterView
Document Manager Control (DMC)	MediumWeight Visual Control DetailView
Thumbnail View Control (TVC)	MediumWeight Visual Control ThumbnailView

Chapter 7

Use cases and related code samples

Code samples for language detection

If each pages contain only one language, and the language is known, the language can be set by the `kRecManageLanguages(sid, SET_LANG, language);` command.

If pages can contain multiple languages from a known set of languages, set the possible languages with the `kRecManageLanguages` command before preprocessing the image.

If each page contains only one language, and the language is unknown, use the automatic single language detection, described in Single Language Detection. The following limitations apply:

- Greek, Russian (Cyrillic), Thai, and Arabic languages and scripts are not recommended.
- Western (Latin) languages without dictionary are not supported.
- Accuracy of detection strongly depends on the image quality and other condition.

Note If the detection process cannot determine the page language, the language of the previous page is set, and LANGDET_INHERITED_WARN is returned by `kRecGetPageLanguages`.

Example: Single Language Detection

```
// Set the possible languages
rc = kRecManageLanguages(sid, SET_LANG, LANG_ALL_LATIN);
rc = kRecManageLanguages(sid, ADD_LANG, LANG_ALL_ASIAN); // CCJK
rc = kRecSetSingleLanguageDetection(sid, true); // Set Automatic Single Language
 Detection (ASLD) mode on
for (iPage = 0; iPage < nPages; iPage++) {
rc = kRecLoadImg(sid, hImageFile, &hPage, iPage);
rc = kRecPreprocessImg(sid, hPage); // Preprocess performs ASLD
rc = kRecGetPageLanguages(hPage, aDetectedLanguages); // Get the language(s) detected
 on the page
for (iLanguage=0; iLanguage < LANG_SIZE; iLanguage++) {
if (aDetectedLanguages[iLanguage] == LANG_ENABLED) {
rc = kRecGetLanguageInfo((LANGUAGES)iLanguage, &LanguageInfo);
printf(" %s detected on page\n", LanguageInfo.EnglishName);
}
}
rc = kRecRecognize(sid, hPage, NULL); // Recognize runs with the detected language
 only!
rc = RecInsertPage(sid, hDoc, hPage, -1); // Append the page
}
rc = RecSetOutputFormat(sid, L"Converters.Text.RTF2000");
rc = RecConvert2Doc(sid, hDoc, strConvOutFileName);
```

Example: Limit the number of possible languages

```
// Set precisely the possible languages
```

```
rc = kRecManageLanguages(sid, SET_LANG, LANG_ENG);
rc = kRecManageLanguages(sid, ADD_LANG, LANG_GER);
rc = kRecManageLanguages(sid, ADD_LANG, LANG_FRE);
rc = kRecManageLanguages(sid, ADD_LANG, LANG_ITA);
rc = kRecManageLanguages(sid, ADD_LANG, LANG_JPN);
rc = kRecManageLanguages(sid, ADD_LANG, LANG_KRN);
rc = kRecSetSingleLanguageDetection(sid, true); // Set Automatic Single Language
 Detection (ASLD) mode on
for (iPage = 0; iPage < nPages; iPage++) {
rc = kRecLoadImg(sid, hImageFile, &hPage, iPage);
rc = kRecPreprocessImg(sid, hPage); // Preprocess performs ASLD
rc = kRecRecognize(sid, hPage, NULL); // Recognize runs with the detected language
 only!
rc = RecInsertPage(sid, hDoc, hPage, -1); // Append the page
}
rc = RecSetOutputFormat(sid, L"Converters.Text.RTF2000");
rc = RecConvert2Doc(sid, hDoc, strConvOutFileName);
```

Code sample for handling large volume output

If several hundred pages need to be converted into a single PDF output, the straightforward workflow may need more system resources than available. To reduce the system resource needs, input data should be grouped into smaller chunks for processing, and append the processed chunks to the output. The input data can be any single or multi-page image file, or a set of files.

Output type	Output file type	Recommended processing size
Direct TXT output	DTXT_PDFIOT and DTXT_IOTPDF_MRC	5 to 10 pages
Layout retention output (Formatted output)	PDF, PDFEdited, PDFImageOnText, and PDFImageSubst	50 to 100 pages

Example: Appending technique for large volume output

```
RecSetOutputFormat(sid, L"Converters.Text.PDFImageOnText");
// set Converters.Text.PDFImageOnText.AppendFrom to the output file name
kRecSettingGetHandle(NULL, "Converters.Text.PDFImageOnText.AppendFrom", &hset, NULL);
kRecSettingSetUString(sid, hset, outpdf);

for (int i=0; i<pages; i++) {
 int ind = i % PACK;     // cache index
 if (ind == 0)
  RecCreateDoc(sid, L"", &hDoc, DOC_NORMAL);
 kRecLoadImg(sid, hFile, &hPages[ind], i);
 kRecPreprocessImg(sid, hPages[ind]);
 kRecRecognize(sid, hPages[ind], NULL);
 RecInsertPage(sid, hDoc, hPages[ind], -1);
 if (ind+1 == PACK || i+1 == pages) { // append cached pages
  RecConvert2Doc(sid, hDoc, outpdf);
  RecCloseDoc(sid, hDoc);
 }
}
```

Code sample for image loading

During the image loading, .NET specific problems can occur.

- The image is encapsulated in `System.Drawing.Bitmap` object in application memory. The bitmap data has to be transferred in byte[] array to `kRecLoadImgM`.
- `BitmapData.Stride` is a signed number:

  Positive value means TOP_DOWN, and negative value means BOTTOM_UP. The sign of `Stride` is mapped into `lineord` of `kRecSetImgFormat`.

  `BytesPerLine` is the absolute value of `Stride`.
- .NET and OmniPage SDK interprets differently the RGB order:
  - `Bitmap.PixelFormat Rgb` means BGR for `kRecLoadImgM`
  - `Bitmap.PixelFormat Argb` means BGRA for `kRecLoadImgM`

The following code sample can be used to load a bitmap image from the memory:

Example: Load bitmap image from memory

```
   //Image is received in argument (Bitmap bmp)
   Rectangle rect = new Rectangle(0, 0, bmp.Width, bmp.Height);
   BitmapData bmpData = bmp.LockBits(rect, ImageLockMode.ReadOnly, bmp.PixelFormat);
   IMG_LINEORDER imgLineOrder = bmpData.Stride < 0 ? IMG_LINEORDER.BOTTOM_UP :
IMG_LINEORDER.TOP_DOWN;
   IMG_INFO ImgInfo = new IMG_INFO();
   ImgInfo.Size.cx = bmp.Width;
   ImgInfo.Size.cy = bmp.Height;
   ImgInfo.DPI.cx = (Int32)bmp.HorizontalResolution;
   ImgInfo.DPI.cy = (Int32)bmp.VerticalResolution;
   ImgInfo.BytesPerLine = (uint)Math.Abs(bmpData.Stride);
   byte[] buf = new byte[Math.Abs(bmpData.Stride) * bmp.Height];
   Marshal.Copy(bmpData.Scan0, buf, 0, buf.Length);
   bmp.UnlockBits(bmpData);
   switch (bmp.PixelFormat){
       case PixelFormat.Format24bppRgb:
           ImgInfo.BitsPerPixel = 24;
           RecAPI.kRecSetImgFormat(0, imgLineOrder, IMG_PADDING.PAD_DWORDBOUNDARY,
IMG_RGBORDER.COLOR_BGR);
           break;
   }
   RecAPI.kRecLoadImgM(0, buf, ImgInfo, out hPage);
```

Code samples for image processing

Three contradicting requirement needs to be considered during image processing: quality, size, and speed. The settings can be selected according to the priorities. The code samples provide examples for the following priorities:

- Quality, size, speed
- Size, quality, speed
- Speed

Example: Embed the original image into PDF, optimized to quality

```
// load image as data stream
kRecLoadImgDataStreamF(sid, jpgfile, &hPage, 0);
// save image into FF_PDF format - preserving compression quality and size
kRecSaveImgF(sid, outpdf, FF_PDF, hPage, II_ORIGINAL, false);
```

Example: Embed the processed image into PDF, optimized to quality

```
// load image as data stream
kRecLoadImgDataStreamF(sid, jpgfile, &hPage, 0);
kRecDecompressImgDataStream(sid, hPage); // decompress DataStream only for
 preprocessing
kRecPreprocessImg(sid, hPage); // perform the necessary transformations
// save image into FF_PDF format - still the DataStream will be saved but with
 transformations!
kRecSaveImgF(sid, outpdf, FF_PDF, hPage, II_CURRENT, false);
```

Example: Create searchable PDF, optimized to quality

```
kRecSetDTXTFormat(sid, DTXT_IOTPDF);
// load image as data stream
kRecLoadImgDataStreamF(sid, jpgfile, &hPage, 0);
// decompress DataStream only for internal processing
kRecDecompressImgDataStream(sid, hPage);
kRecPreprocessImg(sid, hPage);
kRecRecognize(sid, hPage, NULL);
// create searchable PDF preserving compression quality, with readable orientation
 (II_CURRENT)
kRecConvert2DTXTEx(sid, &hPage, 1, II_CURRENT, outpdf); // use II_ORIGINAL for original
 orientation
```

Example: Optimize to size

Use one of the following formats:

- FF_JPG/JPG2K
- FF_PDF/PDF_MRC
- DTXT_IOTPDF/IOTPDF_MRC

```
kRecSetCompressionLevel(sid, 1); // Level 1
kRecSetCompressionTradeoff(sid, COMPRESSION_ADVANCED);
```

Note Compression Level 2 provides better quality with moderate file size.

Example: Optimize to speed

Use one of the following formats:

- FF_JPG/JPG2K
- FF_PDF/PDF_MRC
- DTXT_IOTPDF/IOTPDF_MRC

```
kRecSetCompressionTradeoff(sid, COMPRESSION_FAST);
```

If quality is more important than file size, set `kRecSetCompressionLevel(sid, 5); // Level 4-5`

If file size is more important than quality, set `kRecSetCompressionLevel(sid, 1); // Level 1-2`

For DTXT_IOTPDF_MRC, set `Kernel.OcrMgr.Images.KeepOcrImage = TRUE`

Note During MRC process the OCR results can be used and text detection can be omitted.

Chapter 8

Essay mode

Besides True Page and Flowing Page layouts where text precisely matches the raw picture, in essay mode size and positioning is not in the major focus. The separately formatted text blocks have common properties, like font and font size. It allows the end user to format the text easily by changing font and paragraph attributes of large blocks of text with the same style. The style updates affect all text, that belong to the same style, even if the text is in separate blocks. Moreover, removing headers and footers allows users to reformat the whole text more easily, and present it as an HTML page.

Essay mode supports the following functions:

- Attribute-based style categorization for text
- Building Table of Contents directly from heading levels in DOCX and RTF formats
- Header and footer can be removed to get a continuous text.
- Any of the above features can be used separately or in any combination.

Essay mode is intended for users, who want to process a whole document manually, after chosen individual export flags at save. The input should be large documents with continuous text and multiple pages. Certain sub-features, like Headers/Footers, Heading, and Style, can also be used on their own. This mode is based on the already existing Formatted Text mode with special default settings, that is already available in previous CSDK versions.

Styles in Essay mode

Styles are supported by the following converters:

- RTF: converters.text.rtf2000, converters.text.rtf2000Essay
- DOCX: converters.text.docx, converters.text.docxEssay
- HTML: converters.text.html50, converters.text.html50Essay

The RTF and DOCX converters produce similar result with styles, and have a full style support. HTML converters create both character and paragraph styles, but the HTML format uses only the paragraph styles.

CSDK creates the following style types from a single document, or from pages having similar styles:

Paragraph Style

Space before, and space after properties are exception from categorization, because these can create too many styles. These properties are coming from formatter, and are always unique.

Left-align and justified are commonly mixed up by formatter, hence these add a lower penalty at categorization.

Although many-to-one relationship exists with character styles, the attributes of the belonging character style are repeated in paragraph Style.

Character style

The most common textrun format within a paragraph is applied here. Character style is bound to a paragraph, not to a textrun.

Since character styles are bound across various categorization logic to paragraph styles, in the actual design character style can change only per paragraph units, despite that in RTF and DOCS format the character styles can vary per textruns. This concept differs from the existing practice of using character styles to express emphasis or intense reference, which is a common practice in legal documents. CSDK avoids this design, it differentiates character styles only per paragraph.

Although the attributes of character styles are included in paragraph styles, having character styles separate from paragraph is beneficial. In essays typically much less character styles exist than paragraph styles, especially when Headings processing is enabled. In this way users get rid of unique formatting. Practice tells that in essays the first one or two character styles are covering almost the whole document. The user can focus on dealing with only those few character styles, that belong to more paragraph styles.

Important The attributes of character styles override the paragraph styles. Paragraph styles contain the attributes of the belonging character style redundantly, as it is filled in by the converters with the same value.

It means that when you change any value in a commonly used character style, the effect is immediately visible. If you change the paragraph styles, only the paragraph related attributes take effect, because those are not overridden, for example, space before, and space after. The character related attributes of paragraphs takes effect only when you switch the character style on a text region.

Important The features called "linked paragraph" styles and "based-on" styles in Microsoft Word terminology are not supported.

Heading styles

Headings in a formatted text are extracted based on its outstanding font style or size, and placement above paragraph texts. Heading style is special because the text under this style is direct formatted, and does not follow the style consolidation logic. The heading levels are differentiated mostly by font size. Headings with smaller font size categorized as having higher heading levels, hence those are lower in the content hierarchy.

The following headings are removed:

- The first unique heading levels, like page title and author.
- Those headings, that have too many instances on a page, see Limitations.

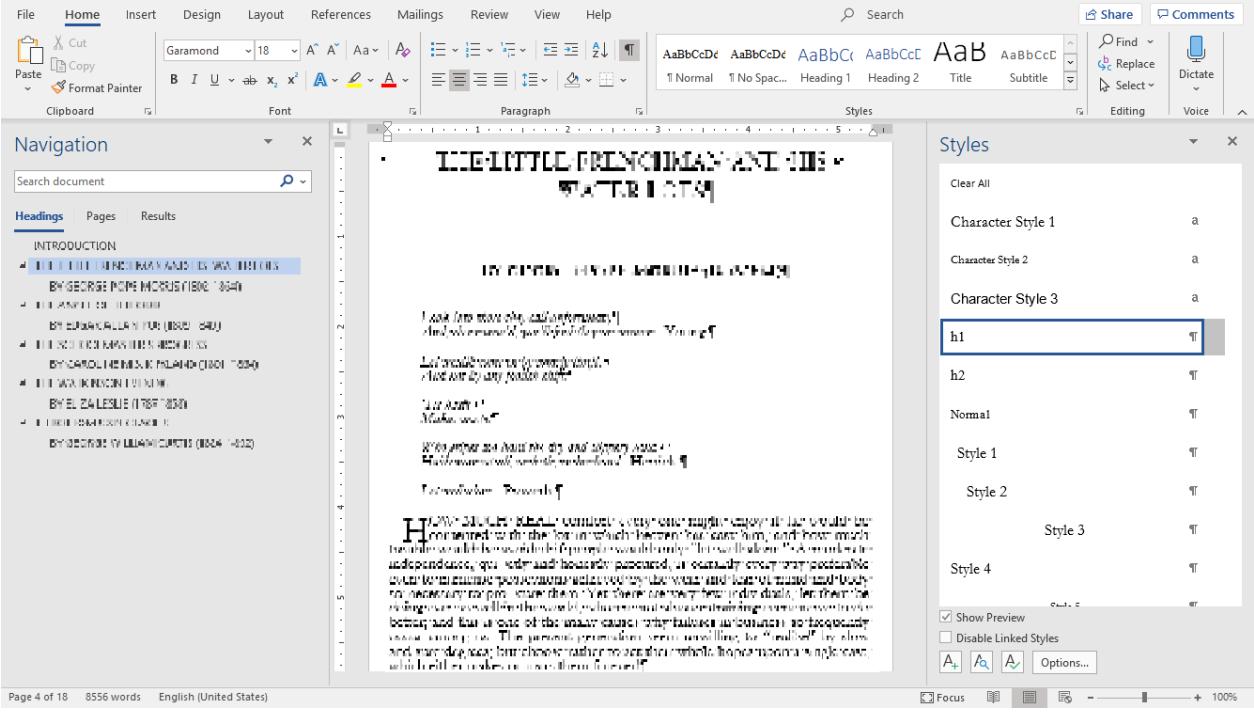
Headings are generated as styles, and the heading level paragraph property is extracted from these heading styles in case of DOCX and RTF conversions.

Headings are created as styles, named from h1 to h7, and the number postfix is taken as the level.

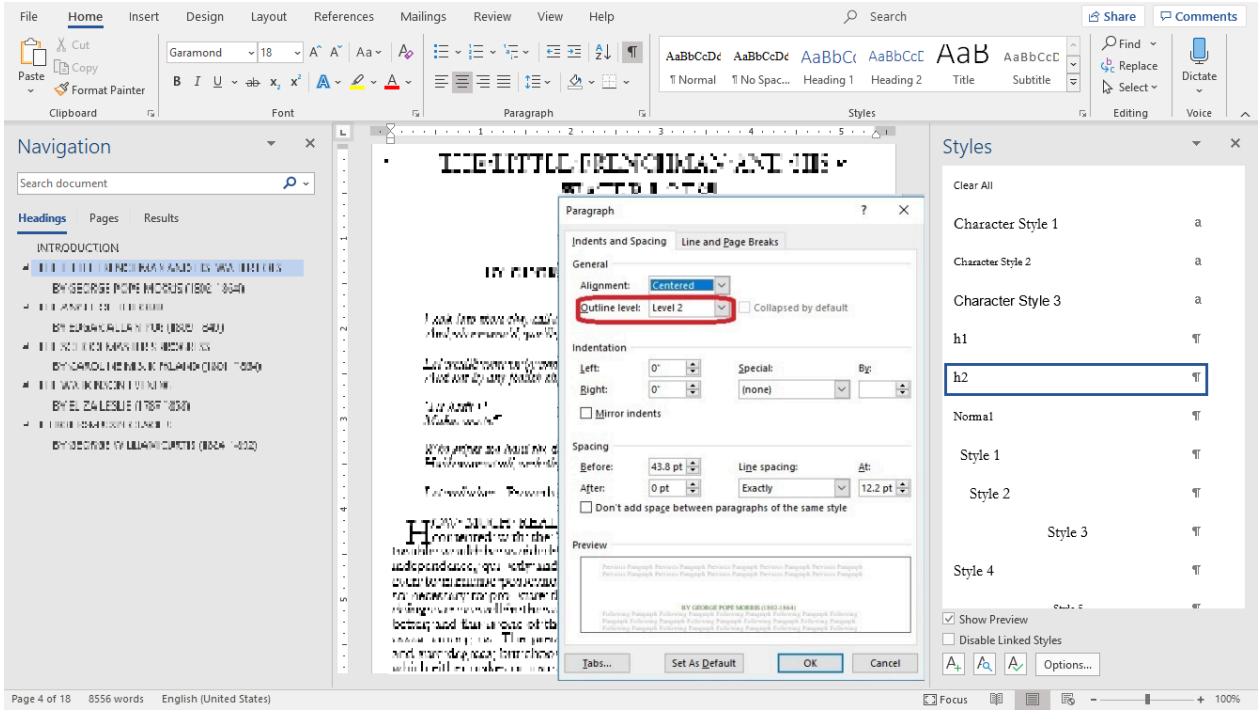
Only DOCX and RTF formats support heading levels, EPUB and HTML formats support headings without heading levels.

Use cases

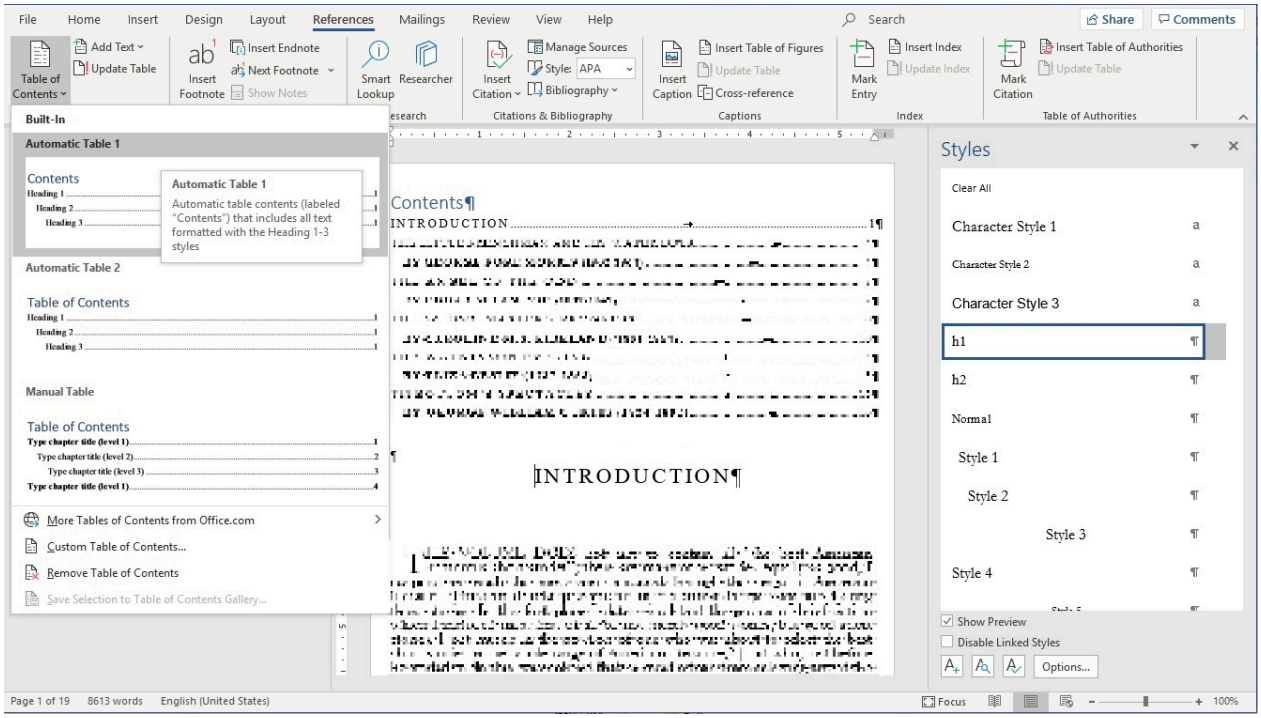
You can view Navigation pane by pressing CTRL+F. Clicking Headings brings up available headings, indented by their levels. This pane allows easy navigation in a large document by clicking on the heading level.



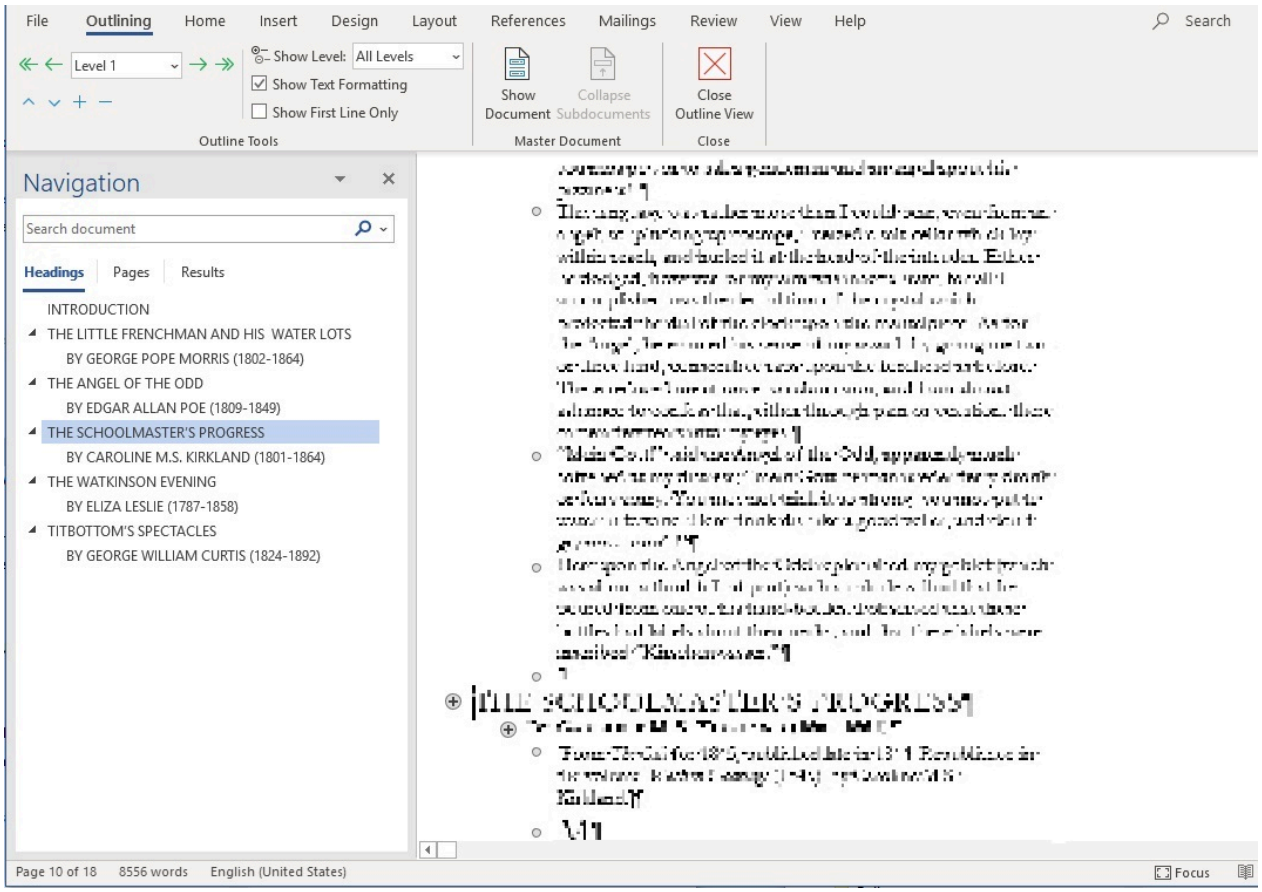
You can set the heading level through Paragraph dialog box, by either right clicking on text, or selecting the proper heading style. This heading level property is being set through style, not through direct format. As a consequence if you change the level to 3 or higher via Style paragraph property, all these levels change.



You can create TOC with Microsoft Word. Move the pointer where the TOC needs to be placed, then click in the menu **References** > **Table of contents** > **Built-in** and choose the preferred TOC type. Microsoft Word automatically generates the belonging metadata, like hidden bookmarks.



You can overview heading levels, and correct a single the heading level of a paragraph in Microsoft Word, by clicking is the menu **View** > **Outlining** . In this view you can reorder blocks of text vertically and horizontally by dragging the circle on the left side of the block. Blocks are derived from heading levels.



Limitations

In order to keep the overall number of headings manageable, certain limitations are introduced:

- Too many occurrences in a document:

  Heading level are excluded if headings at that level appear:

  - on more than 10 instances on at least one page, and
  - at least twice on the majority of the pages
- Short documents are limited by the number of heading levels:

  The maximum heading level is a 2-based logarithm of the number of pages + 1. Any further heading levels below these heading level limits are removed. The thresholds may change in future.

  Numbered headings avoid this limitation.

Character styles

A character style is determined by consolidating the formatting information of a whole paragraph.

Font names and scaling fields are ignored by consolidation differencing. Font sizes, pitch and family have the most influence on style determination.

At a paragraph-level consolidation, every character attribute are set to an average value that was dominantly set within a paragraph, regardless whether it has been used at differencing.

The following character attributes that differ in less than 50% of its paragraph text, are treated as direct formatting in Microsoft Word, applied over the style of the paragraph:

- Underline
- Strikethrough
- Superscript
- Bold
- Italic
- Color

Paragraph styles

The following paragraph properties are used for consolidation normally:

- Alignment (left, center, right, justify)
- First line indent
- Left indent
- Right indent
- Line spacing
- Drop cap
- Character style, see Character styles

However, there are paragraph properties that cannot be used for style consolidation in a normal way:

- Properties used for positioning are ignored:
  - Space before
  - Space after
- In special cases some of the normal properties are also ignored:
  - Depending on alignment

    If the alignment type is "left align", the "right indentation" property is ignored as it can have arbitrary value, and result in too many styles. For similar reasons, when alignment type is "right align", the "left indentation" property is ignored. The "justify" alignment is treated the same as the "left alignment".

The recognition is arbitrating between them, and it would result in too many styles. The more common one is used.

- Depending on First line indent (FLI)

  When FLI is close to 0, it is likely a continuation of the previous paragraph. This time, the first line indent is set through direct formatting. Creating new paragraph style for this case can result in duplicating the number of styles. When FLI is negative, called hanging indents, and the paragraph has numbering or bullets, then FLI is ignored.

- The following properties are fully ignored and set through direct formatting, as these can be too individual:
  - Tabs
  - Numbering and bullets

Style naming rules

CSDK creates styles using the following naming scheme:

- Paragraph style: "Style" + <space> + <number:0-9>
- Character style: "Character Style" + <space> + <number:0-9>
- Heading style, that is a special named paragraph style: h + <number: 1-7>, where number means the level.

The numbering postfix of the style names means:

- 0: direct formatting applies, because the occurrences are too rare to fit into categories 1 to 9.
- 1 to 9: sorted by the occurence of paragraph, from most common (1) to least used styles (9)

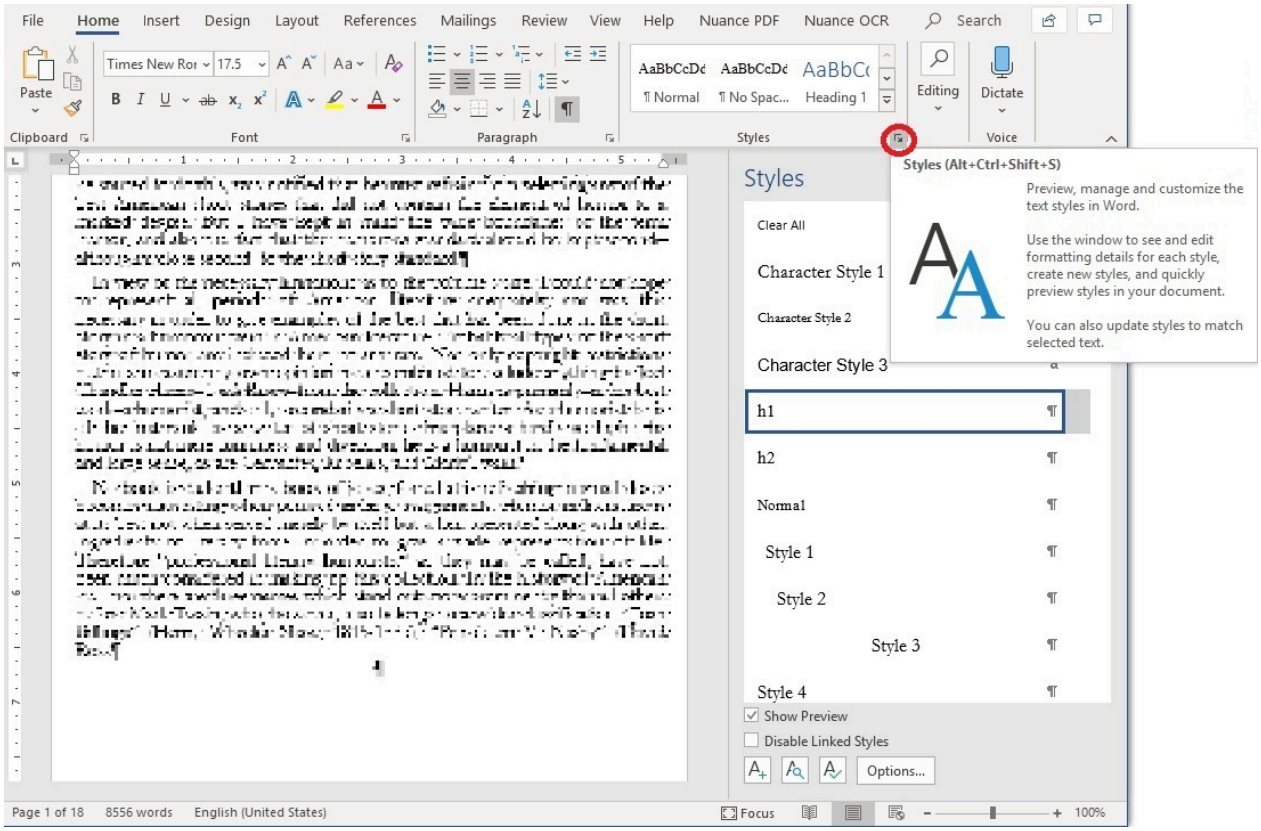
The maximum number of nine styles is based on an ergonomic decision, to avoid introducing too many styles.

You can view and modify a style by right click and select Modify....

To see which texts is affected by a given style in the list, you should right click and choose Select all. If you simply select the style with a left mouse click, it overrides the actual style at the cursor.

If you want to change the maximum number of character and paragraph styles, use the following parameters:

- `Formatter.df.style.maxCharStyles = 9`
- `Formatter.df.style.maxParaStyles = 9`



Troubleshooting

In general, the formatting is never perfect, and the errors are usually present even without styles. However, styles are able to remove noises, for example font size variation.

Perform the following steps in Microsoft Word to determine from where does a single attribute of a text coming.

1. Open the **Styles** ribbon.
2. Click the **Style inspector** ( ) icon.
3. Click **Reveal Formatting** ( ) icon.

   This reveals the whole precedence why a given attribute has appeared as the relevant value to be used by Microsoft Word. The **Style Inspector** pane usually explains the most trivial reasons.
4. Select the **Distinguish style source** check box in the **Reveal Formatting** pane when a more detailed decision of the Microsoft Word needs to be uncovered.

   You can determine where does a single attribute of a text coming from exactly.
5. For inspection purpose you can generate the same document without styles, by setting `Formatter.df.mode = 0` or as described in Use essay mode, and look at the **Reveal Formatting** pane.

Note Besides understanding where the complained attribute comes from, it is usually worth processing the document in different modes, for example Formatted Text (RFP) without Essay flags, or Flowing Page / TruePage, and finding which flag introduces the error. These errors usually come from OCR or formatting problems, or from the expected behavior of an export flag that the user can set.

Use essay mode

1. Enable the set of formatting features supported by Essay mode, before starting to insert pages into a document: `kRecSetIntSetting(sid, "Formatter.df.mode", DFM_Essay);`

   Specify this setting before a call to RecCreateDoc or RecOpenDoc.

   Note Make sure you always use the same flag with further RecOpenDoc calls for a previously saved OPD document.

2. Enable styles for the converter output by using any of the following converter names:
   - `RecSetOutputFormat(sid, "Converters.Text.DocXEssay");`
   - `RecSetOutputFormat(sid, "Converters.Text.Rtf2000Essay");`
   - `RecSetOutputFormat(sid, "Converters.Text.Html50Essay");`

   Without enabling the styles, only the Header/Footer feature can take effect.

3. Set the export flags you need to change from the default values.

   The default settings are:

   ```
   Converters.Text.DocXEssay.HeadersFooters = IgnoreHeadersFooters
   Converters.Text.DocXEssay.ParIndent = FALSE
   Converters.Text.DocXEssay.ParSpacing = FALSE
   Converters.Text.DocXEssay.Styles = TRUE
   ```

   The same flags apply for .rtf2000Essay and .html50Essay formats.

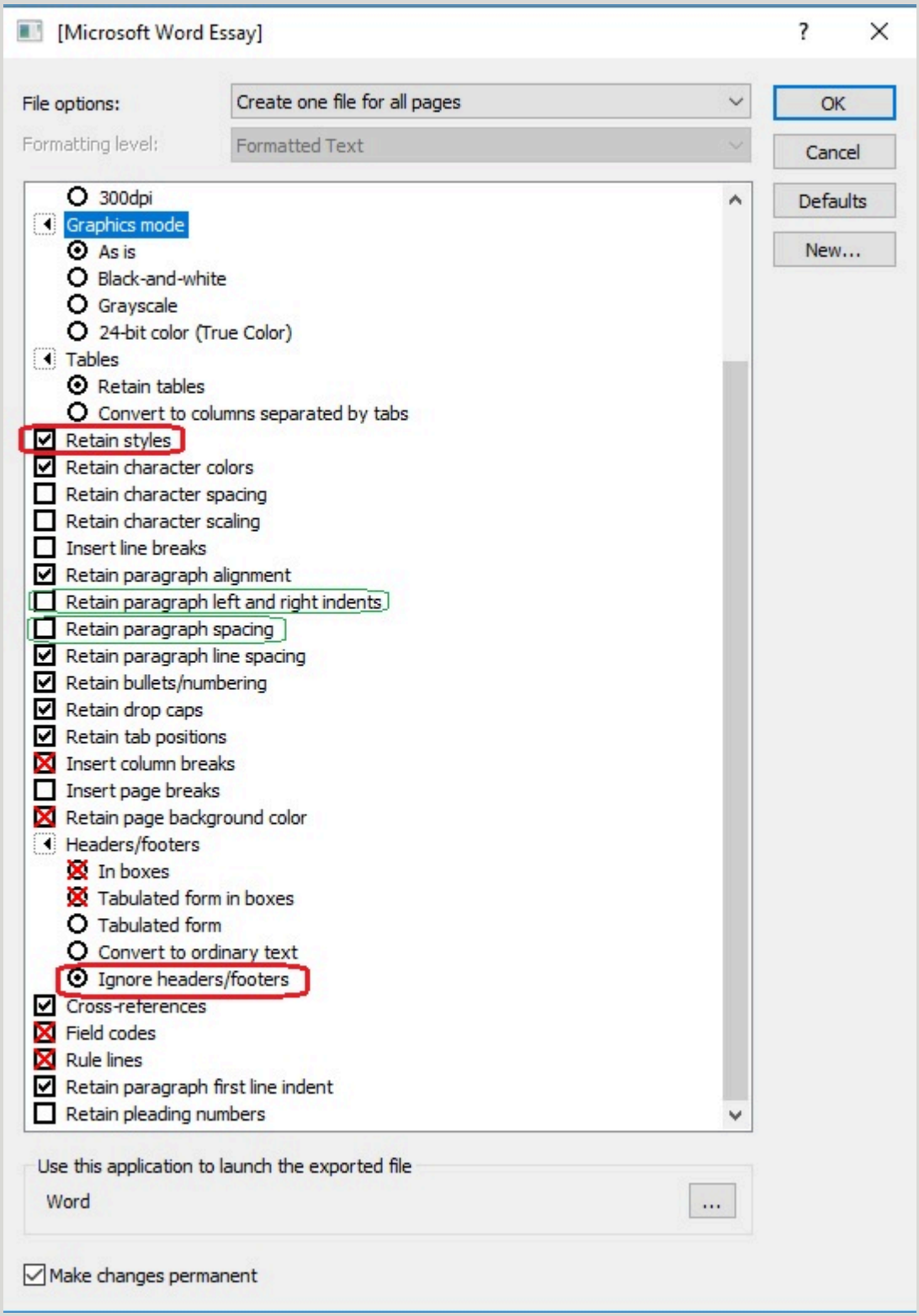
   Instead of DFM_Essay, you can also specify the set of flags separately, depending on which features you really want to have:

   ```
   DFM_StyleConsolidation   = 0x03
   DFM_HeaderFooter    = 0x04
   DFM_HeadingConsolidation  = 0x20
   ```

```
(other defined flags have no relation to Essay mode)
```

There are other predefined internal fixed flags that are not recommended to change: MaskOutputModes, OutputMode, ConsolidatePages.

Note The export flags can be found in the graphical interface of the IproPlus Test Application export options dialog with slightly different names.



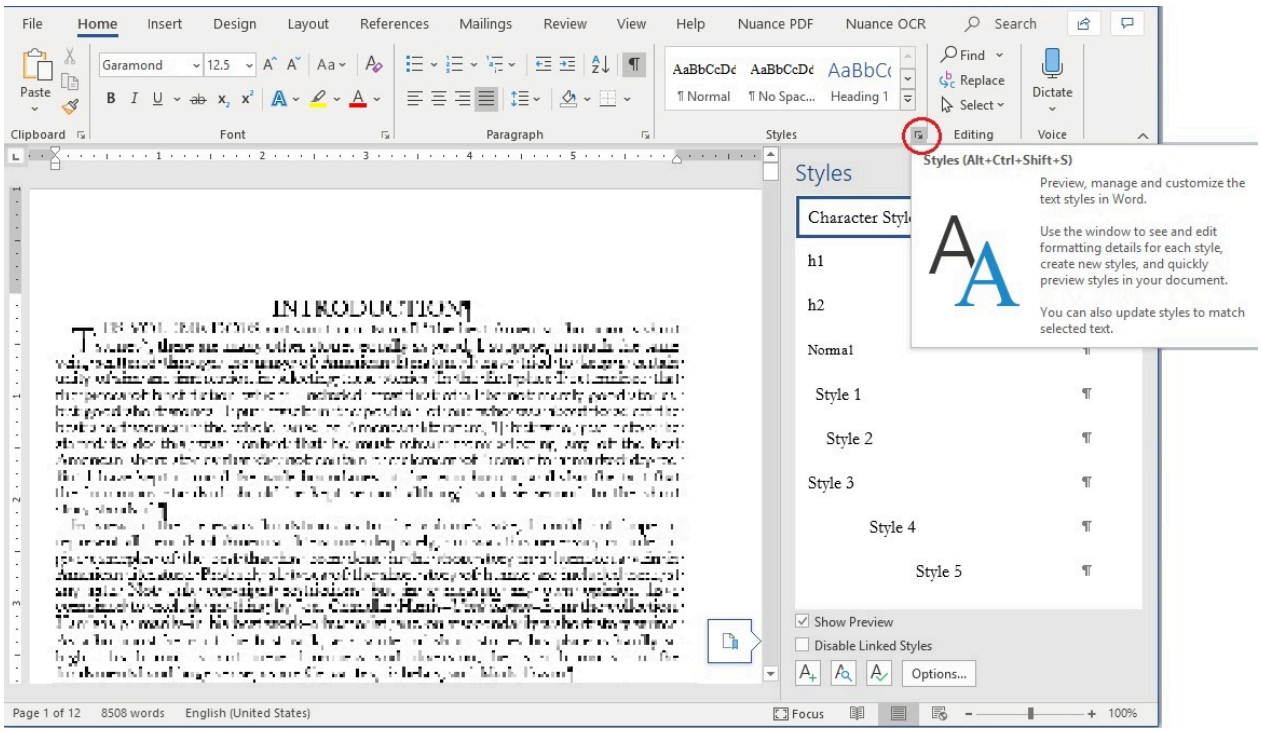
The DFM Essay flags can be specified individually in any arbitrary combination with a restriction: when DFM_HeadingConsolidation or DFM_StyleConsolidation is enabled, the Styles converter option has to be enabled. This flag is called **retainStyles** in the **Export options** dialog box in the IproPlus Test Application graphical user interface. Make sure that the desired Headers/footers (`<converter>.HeadersFooters`) converter option is chosen.

This 2-level enablement logic comes from the separation of recognition into OPD, and afterwards allowing to save the same loaded OPD content into multiple output formats with different converter flags, for example removing retainStyles flag after saving a document with Styles.

If understanding the separate features enablement is too complex for the first time, use DFM_Essay together with the converter names with Essay included. For details, see ConvSettings.sts and IproPlus Test Application export options.

4. Use the `RecConvert2Doc` command to convert the document.

The styles are created.



Appendix A

Abbreviations

CCJK

Simplified Chinese, Traditional Chinese, Japanese and Korean

DCA

Document Classifier Assistant

FLI

First line indent

Form Recognition

Formerly also called Form registration (up to version 18.5)

FTE

Form Template Editor

HWFP

Hardware Fingerprint (a hardware specific identifier)

IVC

Image Viewer Control (a visual control)

License Key

Formerly also known as serial number (up to version 18.5)

LFR

Logical Form Recognition™

OEM Code

The code used for OEM license activation making a link between the application and the license. Formerly also known as secret key or license key (up to version 18.5)

OPLA

OmniPage License Agent

OPLicMgr

OmniPage License Manager

Silent Mode

You can build it into your distribution installer.

SDK Terms

OmniPage Capture SDK

Also known as

- OmniPage Capture SDK
- OmniPage SDK
- CSDK

TEC

Text Editor Control (a visual control)

TOC

Table of content