PDFlib GmbH

TET PDF IFilter

Version 5.1

Windows 用ビジネス向け PDF 検索

PDFlib GmbH
Franziska-Bilek-Weg 9, 80339 München, Germany
www.pdflib.com
電話 *+49 • 89 • 452 33 84-0*
fax +49 • 89 • 452 33 84-99

疑問がおありの際は、*PDFlib* メーリングリストと、*groups.yahoo.com/neo/groups/pdflib/info* にあるアーカイブをチェックしてください。

ライセンスご希望の際の連絡先：*sales@pdflib.com*
商用 *PDFlib* ライセンス保持者向けサポート：*support@pdflib.com*（お使いのライセンス番号をお書きください）

この出版物およびここに含まれた情報はありのままに供給されるものであり、通知なく変更されることがあり、また、*PDFlib GmbH* による誓約として解釈されるべきものではありません。*PDFlib GmbH* はいかなる誤りや不正確に対しても責任や負担を全く負わず、この出版物に関するいかなる類の（明示的・暗示的または法定に関わらず）保証をも行わず、そして、いかなるそしてすべての売買可能性の保証と、特定の目的に対する適合性と、サードパーティの権利の侵害とを明白に否認します。

PDFlib と *PDFlib* ロゴは *PDFlib GmbH* の登録商標です。*PDFlib* ライセンス保持者は *PDFlib* の名称とロゴを彼らの製品の文書内で用いる権利を与えられます。ただし、これは必須ではありません。

Adobe・*Acrobat*・*PostScript*・*XMP* は *Adobe Systems Inc.* の商標です。*AIX*・*IBM*・*OS/390*・*WebSphere*・*iSeries*・*zSeries* は *International Business Machines Corporation* の商標です。*ActiveX*・*Microsoft*・*OpenType*・*Windows* は *Microsoft Corporation* の 商標です。*Apple*・*Macintosh*・*TrueType* は *Apple Computer, Inc.* の 商標です。*Unicode*・*Unicode* ロゴは *Unicode, Inc.* の 商標です。*Unix* は *The Open Group* の 商標です。*Java*・*Solaris* は *Sun Microsystems, Inc.* の 商標です。*HKS* は *the HKS brand association: Hostmann-Steinberg, K+E Printing Inks, Schmincke* の登録商標です。他の企業の製品とサービス名は他の商標やサービスマークである場合があります。

TET PDF IFilter は以下のサードパーティソフトウェアの変更された部分を含んでいます：
Zlib 圧縮ライブラリ、*Copyright © 1995-2012 Jean-loup Gailly and Mark Adler*
Eric Young の書いた *Cryptographic* ソフトウェア 、*Copyright © 1995-1998 Eric Young*（*eay@cryptsoft.com*）
Independent JPEG Group の *JPEG* ソフトウェア、*Copyright © 1991-1998, Thomas G. Lane*
Cryptographic ソフトウェア、*Copyright © 1998-2002 The OpenSSL Project*（*www.openssl.org)*）
Expat XML パーサ、*Copyright © 1998, 1999, 2000 Thai Open Source Software Center Ltd*
ICU International Components for Unicode、*Copyright © 1995-2012 International Business Machines Corporation and others*

TET PDF IFilter は *RSA Security, Inc.* の *MD5* メッセージダイジェストアルゴリズムを含んでいます。

目次

o TET PDF IFilter をインストール

TET PDF IFilter は、Windows 各種システム用のインストーラとして頒布されています。すべての TET PDF IFilter パッケージは、署名済みの IFilter DLL のほかに、サポートファイル群・文書・サンプル群を含んでいます。このインストーラを走らせるには管理者権限が必要です。このインストーラは TET PDF IFilter をインストールして登録します。これに加えて個別の検索環境（Windows Search・SharePoint など）で必要になる手順や、カスタム構成については、本マニュアル内で解説します。

32 ビット版と 64 ビット版　TET PDF IFilter には、32 ビットプラットフォーム用と 64 ビットプラットフォーム用があります。この 2 種類はインストーラが別々であり、同一システム上に共存してインストールする必要があればそれも可能です。64 ビット版は、64 ビット実行形式とともにのみ動作するネイティブ 64 ビット実装です。64 ビットインストーラは 32 ビットシステムへのインストールを拒みますが、32 ビット版は 32 ビットシステムと 64 ビットシステムの両方で動作します。

使用したい IFilter クライアントソフトウェアに一致する 32 ビット版か 64 ビット版かを適切に選んでインストールすることが不可欠です。現在のサーバに対してはこれはたいていは 64 ビット版となります。

新しいバージョンの TET PDF IFilter へアップデート　マシンに古いバージョンの TET PDF IFilter がすでにインストールされている場合には、新しいバージョンをインストールする前にその古いバージョンをアンインストールするべきです。インストレーションパッケージは必ず完全な製品を内容としており、既存のインストレーションに依存していることは絶対にありません。

TET PDF IFilter ライセンスキーを適用　TET PDF IFilter をサーバシステム上で業務目的に使用するには、有効なライセンスキーが必要です。TET PDF IFilter ライセンスを購入いただいたら、無制限に大きな文書の処理を許可するために、ご自分のライセンスキーを適用する必要があります。通常は、TET PDF IFilter をそのインストーラでインストールする際にそのライセンスキーを入力することになります。ただし、インストールの後でライセンスキーをレジストリ内で手動で適用することも可能です（6 ページの「手動インストール」を参照）。32 ビット版と 64 ビット版は同一のライセンスキーを受け入れます。

ライセンスキー 0（ゼロ）を使用すると、評価版をサーバシステム上にインストールすること、および無償デスクトップ版を非商用目的でインストールすることが可能です。

評価版の制限　TET PDF IFilter は、商用ライセンスがなくても完全機能の評価版として使用できます。非ライセンス版はすべての機能をサポートしていますが、ただし 10 ページかつ 1 MB サイズ以下の PDF 文書を処理することのみ可能になっています。TET PDF IFilter の評価版は、業務目的に使用してはなりません。TET PDF IFilter を業務目的に使用するには商用ライセンスが必要です。

無償デスクトップ版を非商用目的で使用　各種デスクトップシステム、すなわち Windows XP/Vista/7/8/10 用の TET PDF IFilter は、個人的利用のために、すなわち非商用目的であれば自由に使用できます。ただし、商用利用と見なされうる状況であれば必ず、このデスクトップ版を運用するにも商用ライセンスが必要です。Windows Server 用 TET PDF IFilter は必ず商用ライセンスが必要です。

対応する IFilter クライアント　TET PDF IFilter は、Microsoft の IFilter インタフェースを実装しています。さまざまなインデックス作成製品がこの IFilter インタフェースに対応しています。本文書ではそのような製品を IFilter クライアントと称します。TET PDF IFilter は下記の製品で試験済みですが、これ以外の、IFilter インタフェースに対応する Microsoft やサードパーティ製品でも動作する可能性があります：

- SharePoint 2013/2016・SharePoint Foundation 2013
- SharePoint Server 2010・SharePoint Foundation 2010
- SQL Server 2016・2014・2012
- Search Server 2010・Search Server Express 2010
- Exchange Server 2007・2010
- Windows Search：Windows Vista/7/8/10 に内蔵されています

TET PDF IFilter には、32 ビット版と 64 ビット版があります。この IFilter の 64 ビット版は、上記製品の 64 ビット版とともにのみ動作します。

XML 構成ファイルの登録とインストール後の手順　インストーラは、さまざまな IFilter クライアント（Windows Search や SharePoint）で使うための然るべき XML 構成ファイルを登録するためのオプションを提供します。いずれかの IFilter クライアントを選ぶと、その照応する TET PDF IFilter 用 XML 構成ファイルがレジストリに追加されます。Windows Search の場合には、TET PDF IFilter の定義済プロパティ群が登録されます（41 ページの 3.1.2「カスタムプロパティ」参照）。

IFilter クライアントによっては、TET PDF IFilter のインストールが済んだ後、さらに手順をふむ必要がある場合があります。この手順については、7 ページの 1 章「動作開始」のそれぞれの節で解説しています。

手動インストール　インストーラは、TET PDF IFilter を使用するために必要な手順をすべて適用しますが、場合によっては、特定の手順を手動で適用する必要が生じるかもしれません。その場合は下記の情報を参照してください。

ライセンスキーを手動で追加するには、下記のレジストリ値にそれを入力します：

```
HKEY_LOCAL_MACHINE\SOFTWARE\PDFlib\TET PDF IFilter5\license
```

TET PDF IFilter DLL をシステムに登録する（IFilter クライアントから確実に見つかるようにするために）には、下記のコマンド（インストレーションディレクトリが違う場合にはその部分を変更）をコンソールウィンドウで実行します：

```
regsvr32 "C:\Program Files\PDFlib\TET PDF IFilter 5.1 64-bit\bin\TETPDFIFilter.dll"
```

このコマンドは必ず、管理者権限を有するコマンドプロンプトから実行（後述）してください。

すでに TET PDF IFilter を使用している IFilter クライアントがある場合には、その DLL を再度登録する前に必ず、TET PDF IFilter にアクセスする関連サービスをすべて停止させてください。また、Windows イベントログも必ず閉じておいてください。

特権コマンドを実行　レジストリへの書き込みアクセス（*regsvr32* プログラムや *proptool* プログラムなどによる）には管理者権限が必要です。コマンドプロンプトを管理者権限で起動させる方法は右記のとおりです：スタートをクリック→検索ボックスに**「*cmd.exe*」**と打つ→すると**「*cmd.exe*」**項目が現れるので右クリック→**「管理者として実行」**を選択。すると、UAC プロンプトが表示されるので、確認の後、管理者権限を有するコマンドプロンプトが開きます。

1 動作開始

この章では、TET PDF IFilter が対応しているいくつかの検索製品（IFilter クライアント）を構成して使用するために必要な最初の手順を解説します。この解説は、まずは TET PDF IFilter が動作開始した状態まで導くことを目的としています。高度な構成事項については 21 ページの 2 章「メタデータプロパティをインデックス」で解説しています。

この章では、TET PDF IFilter の 32 ビット版か 64 ビット版が適切にシステムにインストール済みであることを前提とします。

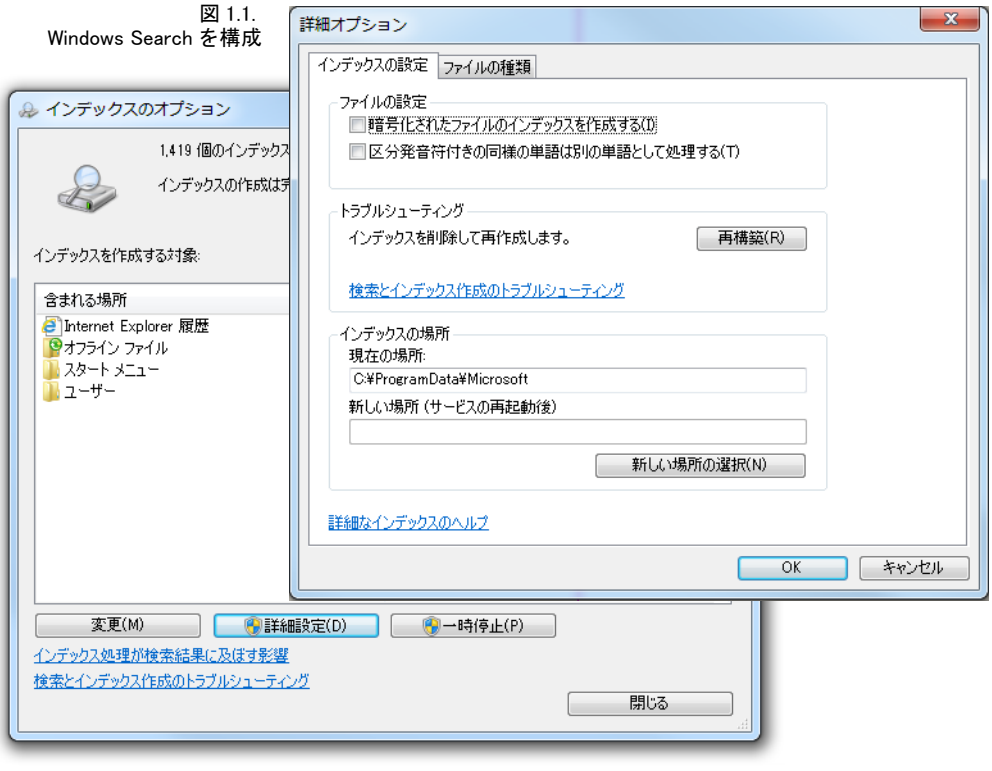
1.1 Windows Search

1.1.1 構成

システム要件　Windows Search は、メタデータのためのプロパティシステムを実装しており、TET PDF IFilter はこのプロパティシステムに対応しています。Windows Search は、Windows Vista/7/8/10 用と、Windows Server 2008/2012/2016 用があります。なお、Windows Server 2016 では Windows Search はデフォルトでは無効化されています。

セットアップと構成　デフォルトでは Windows Search は、ライブラリ（例：**「ドキュメント」**）とオフラインファイルの中の文書群をインデックスします。それ以外の場所（ネットワークドライブも含め）にある文書群をインデックスするよう Windows Search に指示することも可能であり、その方法は以下のとおりです：

図 1.1.
Windows Search を構成



- **「スタート」→「コントロールパネル」→「インデックスのオプション」**をクリック。あるいは Windows 10 では、エクスプローラーウィンドウの検索フィールドの中をクリックしてから（するとメニューバー内で**「検索」**アイテムが有効になります）、**「詳細オプション」→「インデックスが作成された場所の変更」**をクリックするという方法もあります。
- **「変更」**をクリック。**「選択された場所の変更」**セクション内で、インデックスしたいディレクトリ群を選び、OK をクリック。
- **「詳細設定」→「再構築」**をクリックすることによって、その文書群をただちにインデックスさせます。

Windows Search サービスを開始・停止　検索サービスを（より正確には、TET PDF IFilter を呼び出すインデックス作成プロセスを）手動で開始させたり停止させたりすることも可能であり、その方法は以下のとおりです：

- このサービスを開始させるには、管理者権限を有するコンソールウィンドウで下記を打ちます：

```
net start wsearch
```

このサービスを停止させるには下記をタイプします：

```
net stop wsearch
```

- このサービスをコントロールパネルで制御するには：
「スタート」→「コントロールパネル」→「管理ツール」→「サービス」をクリックし、利用可能なサービスの一覧の中で「*Windows Search*」を見つけます。このサービス名をダブルクリックすると、ダイアログが現れ、そこに**「開始」／「停止」**やその他のコントロールがあります。
- カタログを再構築するには：
「スタート」→「コントロールパネル」→「インデックスのオプション」→「詳細設定」→「再構築」をクリック。
すると、すべての文書が再インデックスされます。

なお、Windows Search は、状況によっては自動的にインデックス作成サービスを開始させます。

1.1.2 対話的検索

内容検索　Windows エクスプローラーウィンドウの右上端近くの検索ボックスの中に検索語群を入力することができ、あるいは Windows 7/8 では Windows キーと *F* をタイプすることによって専用の検索ウィンドウを開くこともできます。

シンプルに検索語群をタイプするだけではなく、さらに演算子を使って検索を絞り込むことも可能です。それには AQS（*Advanced Query Syntax* ＝高度クエリ構文）を使用します。AQS の全体的な説明は以下にあります：

msdn.microsoft.com/en-us/library/windows/desktop/bb266512%28v=vs.85%29.aspx

論理キーワード *AND*・*OR*・*NOT* を大文字で書いてはいけません。これらはローカライゼーションの影響を受けません。すなわち、Windows UI の言語にかかわらず、英語の形を使う必要があります。表 1.1 に、内容検索の例をいくつか示します。

Windows エクスプローラーにおけるプロパティ　テキスト検索だけでなく、メタデータプロパティを検索することもできます。エクスプローラーウィンドウでプロパティを表示

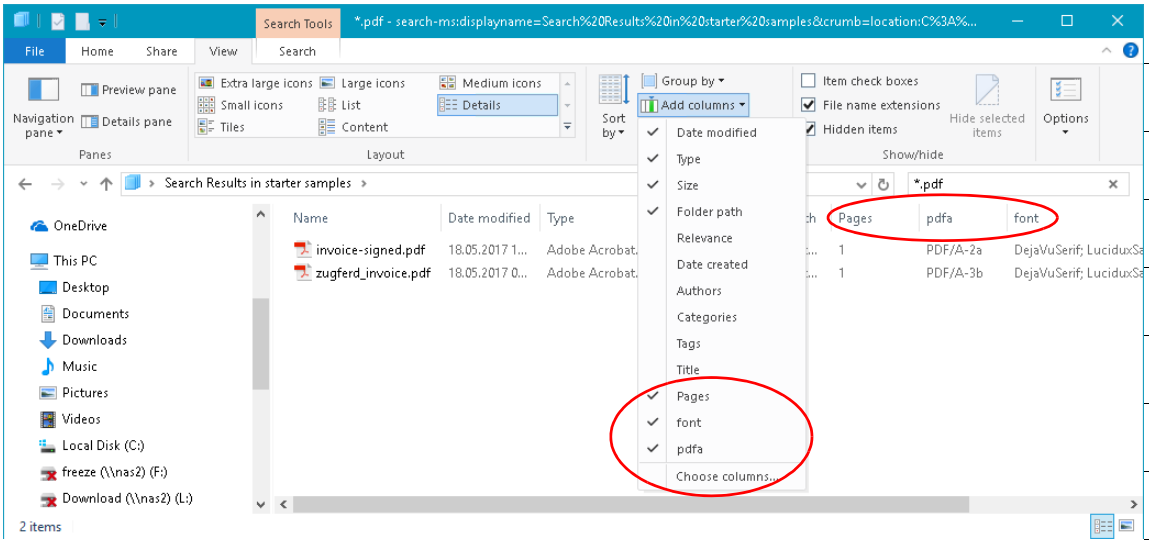
表 1.1 Windows Search を用いて内容を検索

クエリ	説明
Hol	文書の内容か任意のプロパティが、Hol で始まる語を含んでいる
"Sherlock Holmes"	文書内容か任意のプロパティが、完全フレーズ Sherlock Holmes を含んでいる
Holmes AND Watson Holmes + Watson Holmes Watson (Holmes Watson)	文書内容か任意のプロパティが、Holmes と Watson をともに含んでいる
Holmes OR Watson	文書内容か任意のプロパティが、Holmes か Watson のいずれかを含んでいる
Holmes NOT Mowgli Holmes -Mowgli	文書内容か任意のプロパティが、語 Holmes を含んでいるが、語 Mowgli を含んでいない
System.Search.Contents:nutshell	文書のテキスト内容が nutshell を含んでいる。プロパティは無視されます

するには、**「表示」→「列の追加」→「列の選択」**で、1 個または複数の追加のプロパティを選びます。そうしたうえで**「表示」→「詳細」**をクリックすることによって、詳細なファイル情報を表示できます。クエリが検索ボックスに入力された際に（* や *.pdf* でもよい）、プロパティ値が文書に対して得られた場合には、その値がこの新しい列に表示されます（参照）。新しいプロパティは、既存のエクスプローラーウィンドウでは使用されず、新規ウィンドウでのみ使用されます。

正準プロパティ名とローカライズされた表示名　正準名（*System.Author* 等）だけでなく、Windows UI の言語に応じて、ローカライズされた表示名またはラベル（*Authors* 等）をク

図 1.2
Windows エクスプローラーにおける検索結果と、定義済みプロパティによる追加列



エリ内で使用することもできます。クエリ内で認識されたプロパティ名は、検索ウィンドウ内で青色で表示されます。

Windows は、正準名プロパティ名（*System.Author* 等）だけでなく、ローカライズされた表示名（*Author* 等）も使用しています。表示名が空白キャラクタを含んでいる場合（*Date modified* 等）、そのプロパティを用いてクエリする際には、この空白を除去する（*datemodified*）必要があります。この表示名はエクスプローラーウィンドウ内で列見出しとして使用されます。

TET PDF IFilter が対応しているすべてのシェルプロパティの正準名と、ローカライズされた英語表示名を、85 ページの A.1「シェルプロパティセットコレクション」に挙げています。シェルプロパティの、これ以外のローカライズされた名前（ドイツ語等）を知るには、*proptool* ユーティリティを *--list* オプション付きで使用します：

```
proptool --list System.Author
```

ドイツ語システム上での結果出力は以下のようになります：

```
System.Author        Autoren              F29F85E0-4FF9-1068-AB91-08002B27B3D9/4
```

プロパティクエリ　プロパティクエリの例を表 1.2 に示します。メタデータクエリとカスタムプロパティに関する高度な内容については 41 ページの 3.1「Windows Search におけるメタデータプロパティ」を参照してください。プロパティは以下のグループに分類できます：

- シェルプロパティ群。Windows オペレーティングシステムによって定義されています。その正準（言語非依存の）Windows プロパティ名の一覧が下記にあります：
  msdn.microsoft.com/de-de/library/windows/desktop/dd561977%28v=vs.85%29.aspx

  いくつかの Windows シェルプロパティは、PDF 文書内で見つかった情報に基づいて TET PDF IFilter によって記入されます。たとえば *System.Document.PageCount* や *System.Keywords* などです。TET PDF IFilter によって出力されるすべてのシェルプロパティの一覧を表 A.1 に挙げます。この表には、正準名と、ローカライズされた英語表示名（ラベル）を挙げています。
- TET PDF IFilter はこれ以外にもサポートしている定義済みプロパティ群があります。これらは、プロパティセットにグループ分けされており、表 A.2 〜表 A.5 に挙げています。25 ページの 2.3「定義済みプロパティ」に、定義済みプロパティを構成する際に関する説明があります。
- カスタムプロパティ群：TET PDF IFilter 内で構成することができる、ユーザー定義のメタデータプロパティです。28 ページの 2.5「カスタムプロパティ」参照。

表 1.2 Windows Search を用いてプロパティを検索。正準名を用いた長い形式と、英語ラベルを用いた短い形式を示しています

クエリ	説明
Windows シェルプロパティを検索	
`System.Author:Doyle` `Authors:Doyle`	Doyle を含んでいる作成者
`System.Author:"Conan Doyle"` `Authors:"Conan Doyle"`	語群 Conan Doyle を含んでいる作成者
`System.Author:Doy` `Authors:Doy`	Doy で始まっている作成者

表 1.2 Windows Search を用いてプロパティを検索。正準名を用いた長い形式と、英語ラベルを用いた短い形式を示しています

クエリ	説明
System.DateModified:=2017-03-27 datemodified:=2017-03-27	更新日が 2017 年 3 月 27 日。この正準日付形式は Windows UI 言語にかかわらず使用できます
System.Document.DateCreated: = 2017-03-27 contentcreated: = 2017-03-27	作成日が 2017 年 3 月 27 日
System.Document.PageCount: >= 100 Pages: >= 100	文書に 100 ページ以上ある
TET PDF IFilter 内で定義済みのプロパティを検索	
PDFlib.TET.pdfa:=PDF/A-2b pdfa:=PDF/A-2b	文書が PDF/A-2b に準拠している
PDFlib.TET.pdfa:~<PDF/A pdfa:~<PDF/A	文書が PDF/A-1・PDF/A-2・PDF/A-3 のいずれかに準拠している
PDFlib.TET.producer:~=Microsoft producer:~=Microsoft	文書に Producer エントリがあり、その内容が Microsoft である
PDFlib.TET.fullpdfversion: <170 fullpdfversion: <170	文書が PDF 1.7 よりも古い PDF バージョンに準拠している
PDFlib.TET.font:=Calibri font:=Calibri	Calibri で始まる名前のフォントが適用されたテキストが文書内にある

1.1.3 プログラムで検索

対話的なクエリだけではなく、高度クエリ構文をプログラムで使用することも可能です。検索インデックスをデータベースライクなプログラミングインタフェースを通じて呈示する SQL 構文拡張を使用できます。

メタデータプロパティに対する SQL クエリ　SQL クエリは、定義済みのプロパティを検索することもできますし、カスタムプロパティを検索することもできます。いくつかの例を後述します。これらの例は、XML 構成ファイル内ですべての定義済みプロパティが有効化されていることと、*predefined_properties.propdesc* プロパティ記述が登録されていることを前提にしています。また、ADO（*ActiveX Data Objects*）と PowerShell スクリプトを使用して SQL ベースのクエリを発行しています。しかし、これ以外の任意の ADO または ADO.NET 環境においてもこれらの SQL ステートメントを使用可能です。Windows Search のための SQL 構文拡張の説明が下記にあります：

msdn.microsoft.com/en-us/library/bb231256(VS.85).aspx

プロパティ検索は 2 つの方向で行うことができます：

- 特定のプロパティ値をクエリ。例：作成者が *Doyle* なのはどの文書か？
- 1 個ないし複数のファイルの中における特定のプロパティの値をクエリ。例：この文書の作成者は誰か？

PowerShell で SQL クエリを送信　サンプル PowerShell クエリスクリプト群が TET PDF IFilter とともにインストールされます。PowerShell をよく御存じない場合の若干のヒント：

- 署名されていない PowerShell スクリプトを実行するには、管理者として以下のコマンドを 1 回実行する必要があります：

```
set-executionpolicy remotesigned
```

- スクリプトを実行するには、たとえば *query_text_in_pdf.ps1* を実行する場合、PowerShell ウィンドウ内で

```
& query_text_in_pdf.ps1
```

以下の PowerShell スクリプトは、すべての文書について PDF/A 準拠プロパティをリストします：

```
$objConnection = New-Object -comobject ADODB.Connection
$objRecordset = New-Object -comobject ADODB.Recordset
$objConnection.Open("Provider=Search.CollatorDSO;Extended
Properties='Application=Windows';")

$objRecordSet.Open(
"SELECT System.ItemPathDisplay, `"PDFlib.TET.pdfa`" FROM SYSTEMINDEX ", $objConnection)

While ($objRecordset.EOF -ne $True) {
    $private:item = $objRecordset.Fields.Item("System.ItemPathDisplay")
    Write-Output $item.Value
    $item = $objRecordset.Fields.Item("PDFlib.TET.pdfa")
    Write-Output $item.Value
    $objRecordset.MoveNext()
}
```

以下の PowerShell スクリプトは、PDF/A 準拠が *PDF/A-2* を含んでいるすべての文書をリストします：

```
$objConnection = New-Object -comobject ADODB.Connection
$objRecordset = New-Object -comobject ADODB.Recordset
$objConnection.Open("Provider=Search.CollatorDSO;Extended
Properties='Application=Windows';")

$objRecordSet.Open("SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE " +
        "`"PDFlib.TET.pdfa`" = SOME ARRAY ['PDF/A-2']", $objConnection)

While ($objRecordset.EOF -ne $True) {
    $private:item = $objRecordset.Fields.Item("System.ItemPathDisplay")
    Write-Output $item.Value
    $objRecordset.MoveNext()
}
```

VBScript で SQL クエリを送信　以下の VBScript コードは、すべての文書を、その文書を作成したアプリケーションの名前とともにリストします。残念ながら、VBScript クエリを用いる場合には、一部のシェルプロパティしか使用できません。それ以外のプロパティをクエリすることはできません：

```
On Error Resume Next

Set objConnection = CreateObject("ADODB.Connection")
Set objRecordSet = CreateObject("ADODB.Recordset")

objConnection.Open "Provider=Search.CollatorDSO;Extended
Properties='Application=Windows';"

objRecordSet.Open "SELECT System.ItemPathDisplay, System.ApplicationName FROM
SYSTEMINDEX", objConnection

objRecordSet.MoveFirst

Do Until objRecordset.EOF
    Wscript.Echo objRecordset.Fields.Item("System.ItemPathDisplay")
    Wscript.Echo objRecordset.Fields.Item("System.ApplicationName")
    Wscript.Echo ""
    objRecordset.MoveNext
Loop
```

SQL を用いて複雑なプロパティクエリ　以下のサンプル群は、関連する SQL ステートメントのみを内容としており、任意の SQL 環境で使用できます。これらのステートメントを PowerShell スクリプト内で使用するには、クォーテーションを正しく行う必要があります。たとえば *"PDFlib.TET.pdfa"* を *`"PDFlib.TET.pdfa`"* へ変えます。以下の多くの例は、ベクトルプロパティに対する配列クエリを使用しています（30 ページの「多値プロパティ」参照）。配列クエリのための構文に関する詳細が下記にあります：

msdn.microsoft.com/en-us/library/bb231264(VS.85).aspx

- 作成者が *Doyle* を含んでいるすべての文書をリスト：

```
SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE CONTAINS("System.Author",
'Doyle')
```

- 作成者が ***Rudy*** で始まるすべての文書をリスト：

```
SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE CONTAINS("System.Author",
'"Rudy*"')
```

- PDF/A-1a に準拠しているすべての文書をリスト：

```
SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE "PDFlib.TET.pdfa" = 'PDF/A-
1:2005'
```

- フォント ***Bembo*** と ***TimesNewRoman*** のうち少なくとも1つを含んでいるすべての文書をリスト：

```
SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE "PDFlib.TET.font" = SOME ARRAY
['Bembo', 'TimesNewRoman']
```

- ***Bembo*** と ***Bembo-Bold*** の両方のフォントを含んでいるすべての文書をリスト：

```
SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE "PDFlib.TET.font" = SOME ARRAY
['Bembo'] AND "PDFlib.TET.font" = SOME ARRAY ['Bembo-Bold']
```

- 少なくとも 1 つのテニス画像（Photoshop カテゴリ ***TEN*** ＝テニス）があるすべての文書をリスト：

```
SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE
"PDFlib.TET.images.photoshop.SupplementalCategories" = SOME ARRAY ['TEN']
```

- PDF バージョンが 1.6 よりも高いすべての文書をリスト（TET PDF IFilter は PDF バージョンを、そのバージョン番号を 10 倍にした文字列として返します。たとえば PDF 1.6 なら 16）：

```
SELECT System.ItemPathDisplay FROM SYSTEMINDEX WHERE "PDFlib.TET.pdfversion" > '16'
```

1.2 SharePoint

1.2.1 システム要件

TET PDF IFilter は、下記の SharePoint 構成とともに動作します：

- SharePoint Server 2016
- SharePoint Server 2013・SharePoint Foundation 2013：hotfix KB2883000 または 2014 年 7 月 8 日の SharePoint Server 2013 用累積的更新プログラムが必要です。
- SharePoint Server 2010、SharePoint Foundation 2010 を Search Server か Search Server Express とともに使用の場合、およびそれよりも古い SharePoint のバージョン群。

SharePoint に関する詳しい情報は下記にあります：
msdn.microsoft.com/en-us/sharepoint/default.aspx

1.2.2 SharePoint 2016・2013 に対するインストール

TET PDF IFilter を SharePoint 2016 または 2013 とともに使用するように構成するには以下の手順を踏みます：

- TET PDF IFilter を、インストーラを用いてインストール。
- 「*SharePoint* 管理シェル」ウィンドウを開き、下記のコマンド群を入力することによって TET PDF IFilter による PDF インデックス作成を有効にします：

```
$ssa = Get-SPEnterpriseSearchServiceApplication
Set-SPEnterpriseSearchFileFormatState -SearchApplication $ssa -Identity pdf ↵
     -UseIFilter $true -Enable $true
```

- 前記のコマンドが成功したかどうかを確認するには下記のコマンドを使用できます：

```
Get-SPEnterpriseSearchFileFormat -SearchApplication $ssa -Identity pdf
```

このコマンドの出力は下記のようになります（エントリ *UseIFilter* の値が *True* になっている必要があります）：

```
Identity   : pdf
Name       : PDF
MimeType   : application/pdf
Extension  : .pdf
BuiltIn    : True
Enabled    : True
UseIFilter : True
```

- ここで SharePoint 検索サービスを再起動します：

```
net stop OSearch16                  % SharePoint 2013の場合: net stop OSearch15
net stop SPSearchHostController
net start SPSearchHostController
```

SPSearchHostController サービスを起動すると、*OSearch15* サービスも暗黙的に起動されます。

1.2.3 SharePoint 2010 に対するインストール

インストールと再起動　TET PDF IFilter を SharePoint 2010 ともに使用するように構成するには以下の手順を踏みます：

- TET PDF IFilter を、インストーラを用いてインストール。
- TET PDF IFilterがインストールされている間SharePointが動作している場合、SharePoint クローラは新たにインストールされた PDF IFilter のことを、SharePoint 検索サービスが再起動されるまで知りません。これを再起動しないと、クローラから下記エラーメッセージを受けます：

```
The filtering process could not load the item.
This is possibly caused by an unrecognized item format or item corruption.
```

SharePoint 検索サービスを再起動するには下記コマンドを適用します：

```
net stop osearch14
net start osearch14
```

- SharePoint 管理ウェブページを開く：**「スタート」→「すべてのプログラム」→「*Microsoft Office Server*」→「*SharePoint 3.0* サーバーの全体管理」**。
- すると、ウェブページ「**共有サービス管理 : *SharedServices1***」が開きます。**「検索」**の下で**「検索の設定」**をクリック。**「ファイルの種類」**をクリック。
- 開いたページで**「新しいファイルの種類」**をクリックし、「***pdf***」ファイル拡張子を作成。
- 次節で述べるように PDF ファイルアイコンを構成することも可能です。

PDF ファイルアイコンを追加　SharePoint クエリの結果一覧の中で PDF 文書を視覚的に表現するために、PDF アイコンを構成することもできます。このアイコンは PDF 文書に対して表示されます。以下のように操作します：

- PDF アイコンを下記からダウンロードし、

  www.adobe.com/legal/permissions/icons-web-logos.html#pdficon

  pdficon_small.gif という名前で下記ディレクトリに保存：

  ```
  C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\TEMPLATE\IMAGES
  ```

- 下記ディレクトリにある *Docicon.xml* 構成ファイルをテキストエディタで開く：

  ```
  C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\TEMPLATE\XML
  ```

- *<ByExtension>* エレメントの子エレメントとして下記の行を追加：

  ```
  <Mapping Key="pdf" Value="pdficon_small.gif"/>
  ```

- IIS Web server を再起動。

1.2.4 シンプルなテキスト検索と高度なテキスト検索

SharePoint で検索クエリを構築するにはいくつかの方法があります：

- クエリキーワード
- SQL クエリ
- URL 内に符号化したクエリ

メタデータクエリについては 45 ページの 3.2「SharePoint におけるメタデータプロパティ」を参照してください。

1.3 Exchange Server

システム要件　TET PDF IFilter は Microsoft Exchange Server 2010 とともに動作します。もっと古いバージョンの Exchange Server とも動作する可能性がありますが、これらの組み合わせは試験されていません。

セットアップと構成　TET PDF IFilter を Exchange Server とともに使用するには以下の手順を踏みます：

- TET PDF IFilter を、インストーラを用いてインストール。
- 下記のインストール後手順を行います。

インストール後手順　TET PDF IFilter を Microsoft Exchange Server とともに使用するために登録する必要があります。そのために管理者権限で下記の PowerShell スクリプトを実行します：

```
register_in_exchange_2010.ps1
```

このスクリプトは、TET PDF IFilter インストレーションディレクトリのサブディレクトリ *IFilter clients\Exchange* にインストールされています。このスクリプトの実行が成功した後に、*Microsoft Exchange Search Indexer* サービスを再起動する必要があります。この作業は「サービス」コントロールパネルから行うこともできますし、あるいは PowerShell からコマンドラインで行うこともできます：

```
stop-service MSExchangeSearch -Force
start-service MSExchangeSearch
```

この登録スクリプトは、新しいバージョンの TET PDF IFilter をインストールするたびに実行する必要があります。なぜなら TET PDF IFilter のデフォルトインストレーションディレクトリは更新の際に変わるからです。

TET PDF IFilter が登録された後は、すべての新規メールの PDF 添付が Exchange によってインデックスされます。既存のメッセージ群の PDF 添付をインデックスするためには、すべてのメールボックスを再度インデックスする必要があります。MSDN ウェブサイトの下記の記事に、Exchange の全文テキストインデックスを再構築するための可能な手順が記述されています：
technet.microsoft.com/en-us/library/aa995966(v=EXCHG.80).aspx

1.4 SQL Server

システム要件　TET PDF IFilter は、以下のエディションの SQL Server とともに動作します：

- SQL Server 2012・2014・2016

SQL Server に関する詳しい情報は下記にあります：

msdn.microsoft.com/en-us/library/bb418498.aspx

SQL Server でのフルテキスト検索に関する詳しい情報は下記にあります：

msdn.microsoft.com/en-us/library/mt590198(v=sql.1).aspx

セットアップと構成　SQL Server 内におけるフィルタ群の使用を開発者が完全に制御できるようにするため、インストーラは、SQL Server のどのインスタンスにおいても、自動的に TET PDF IFilter を登録することはしません。ですので、手動で TET PDF IFilter を SQL Server のすべてのインスタンスに対して個別に登録する必要があります。

以下の手順は、SQL Server に、システムワイドにインストールされている IFilter 群にアクセスするよう指示します：

- TET PDF IFilter を、インストーラを用いてインストール。
- SQL Server Management Studio を走らせ、以下のステートメント群を実行することによって、システムワイドの文書フィルタ群を SQL Server のこのインスタンスで利用可能にします（詳しくは*msdn.microsoft.com/en-us/library/dd207002%28v=sql.120%29.aspx* を参照）：

```
exec sp_fulltext_service 'load_os_resources', 1;
GO
exec sp_fulltext_service 'update_languages'
GO
exec sp_fulltext_service 'restart_all_fdhosts'
GO
```

構成を試験　SQL Server の 1 つのインスタンスで TET PDF IFilter が利用可能になったことを確認するために、構成結果をチェックできます。下記ステートメントを用います：

```
SELECT document_type, path FROM sys.fulltext_document_types WHERE document_type = '.pdf'
```

下記のような結果出力行は、そのインスタンス向けに TET PDF IFilter が正しく構成されていることを示します（正確なパスは、お使いのインストレーションパスによって異なります）：

```
.pdf    C:\Program Files\PDFlib\TET PDF IFilter 5.1 64-bit\bin\TETPDFIFilter.dll
```

フルテキスト PDF インデックス作成用データベーステーブルを作成　TET PDF IFilter は SQL Server によって、型 *varbinary(max)* の列に格納された PDF 文書に対するフルテキストインデックスを生成するために使用されます。このままでは文書種別がわからないので、ファイル拡張子を、そのテーブル内の別の列、いわゆる**型列**に格納する必要があります。型列は、任意の文字ベースのデータ型にすることができます。*VARCHAR(4)* を使用し、ファイル拡張子 *pdf* を格納しましょう。

下記ステートメント群は、1 個のサンプル PDF 文書を *data* 列に、そのファイル名を *name* 列に、その型を *extension* 列に持つ *DocumentTable* を作成します：

```
CREATE DATABASE TestDatabase
GO
USE TestDatabase
GO
CREATE TABLE DocumentTable
(pk INT NOT NULL IDENTITY CONSTRAINT DocumentTablePK PRIMARY KEY,
data VARBINARY(MAX), name VARCHAR(100), extension VARCHAR(4))
GO
INSERT INTO DocumentTable(data, name, extension) SELECT *,'The_Hound_of_the_
Baskervilles.pdf','pdf' FROM OPENROWSET(BULK 'C:\PDF\The_Hound_of_the_Baskervilles.pdf',
SINGLE_BLOB) AS Document
GO
```

これで、フルテキストインデックスを生成できます：

```
sp_fulltext_database 'enable'
GO
CREATE FULLTEXT CATALOG TestCatalog AS DEFAULT
GO
CREATE FULLTEXT INDEX ON DocumentTable (data TYPE COLUMN extension)
KEY INDEX DocumentTablePK
GO
```

フルテキストインデックスを削除・再生成　フルテキストインデックスを削除するには下記ステートメント群を用います：

```
USE TestDatabase
GO
DROP FULLTEXT INDEX ON DocumentTable
GO
```

フルテキストインデックスを再生成：

```
USE TestDatabase
CREATE FULLTEXT INDEX ON DocumentTable (data TYPE COLUMN extension)
KEY INDEX DocumentTablePK
GO
```

シンプルなテキスト検索と高度なテキスト検索　フルテキストインデックス内の個々の語をクエリすることが可能です：

```
SELECT name FROM DocumentTable WHERE CONTAINS(*, 'Watson')
GO
```

複数の語で構成されたフレーズを検索するには、そのフレーズをダブルクォーテーションでくくります：

```
SELECT name FROM DocumentTable WHERE CONTAINS(*,'"Arthur Conan Doyle"')
GO
```

これらの手順を、与えられた PDF サンプル群で演示するサンプルスクリプトが一つ、TET PDF IFilter とともにインストールされます。Transact-SQL における *CONTAINS* 述語に関する詳しい情報は下記にあります：

msdn.microsoft.com/en-us/library/ms187787(SQL.100).aspx

メタデータクエリについては50ページの3.3「SQL Serverにおけるメタデータプロパティ」を参照してください。

2 メタデータプロパティをインデックス

TET PDF IFilter は、メインテキストだけでなく、さまざまなメタデータプロパティをインデクサへ与えることもできます。これによって、ページ上の特定のテキスト以外を検索したい場合であっても、強力な検索が可能になっています。

2.1 PDF 内のメタデータソース

多くの PDF 文書は、**「作成者」**や**「タイトル」**などの文書情報項目群を内容として持っています。文書情報項目群だけではなく、PDF 文書は XMP メタデータを内容として持っている場合もあります。TET PDF IFilter は、PDF 文書内のいくつかの種類のメタデータのインデックス作成に対応しています。

定義済みとカスタムの文書情報項目　文書情報項目群は、古くてシンプルな種類の PDF メタデータであるととらえることができます。これらは Acrobat で**「ファイル」→「プロパティ ...」**で表示できます。PDF 標準規格には下記の文書情報項目が定義されています：

```
Title Author Subject Keywords Creator Producer CreationDate ModDate Trapped
```

これらの標準項目群以外に、カスタムの項目群を文書情報項目の集合に追加することも可能です。これらは Acrobat で（Acrobat Reader では無理）**「ファイル」→「プロパティ ...」→「カスタム」**で表示・編集することができます。

定義済み文書情報項目群も、カスタム文書情報項目群も、いずれも TET PDF IFilter 内で pCOS パスを用いて指し示すことができます（後述）。

文書レベルと画像レベルの XMP プロパティ　XMP（*Extensible Metadata Platform*[1]）はメタデータのためのフレームワークです。XMP はたとえば、PDF/A 準拠のために必須であり、多くのアプリケーションがこれに対応しています。XMP メタデータは、多数のプロパティを内容とするスキーマ群の中に編成されています。プロパティは、1 個の名前空間接頭辞と、そのプロパティ名を用いて指し示されます。

XMP メタデータは通常、文書全体に関連づけられています。しかし、PDF では XMP を個別のページ・画像などのオブジェクトに関連づけることも可能になっています。実際には、この機能は主にラスタ画像について用いられています。たとえば、デジタル画像に、その撮影者・著作権表記・シーン詳細などの情報を帯びさせることができます。画像関連の XMP メタデータの重要なソースの 1 つは Adobe Photoshop です。Photoshop から PDF を生成した場合、あるいは Photoshop から生成された画像をいろいろな Adobe のワークフローで使った場合、その XMP メタデータが通常、その PDF 文書内に入れ込まれており、TET PDF IFilter でインデックスすることが可能です。

XMP 仕様は、すべての定義済みの XMP スキーマ群とプロパティ群の記述を含んでいます。XMP について詳しい情報は下記にあります：

www.adobe.com/devnet/xmp

文書レベルか画像レベルの XMP プロパティ群は（これ以外のいろいろなオブジェクトに関連づけられた XMP も）、2 段階で指し示すことができます：pCOS パスは、関連する PDF

1. www.adobe.com/products/xmp を参照。

オブジェクトを特定し、2 部分の XMP プロパティ名は、求める XMP スキーマとプロパティを指し示します。

拡張 pCOS パス　pCOS（*PDFlib Comprehensive Object Syntax*）インタフェースは、PDF 文書のページ寸法・メタデータ・インタラクティブ要素といった、ページ内容を記述しないすべてのセクションから任意の情報を取得するためのシンプルでエレガントな機構を提供します。pCOS インタフェースの利用例や、pCOS パス文法の説明は、別の文書として入手可能な pCOS パスリファレンスに含まれています。さらなる利用例が、*www.pdflib.com/pcos-cookbook/* にある pCOS Cookbook にあります。

pCOS は、TET PDF IFilter で、文書に関する情報を指し示すために用いることができます。pCOS パスは、その PDF 文書のしおり・フォント名・ページ・サイズといったいくつかの側面を表現し、また、文書情報項目や XMP メタデータストリームを指し示すこともできます（ただし XMP ストリーム内のプロパティを指し示すことはできません）。pCOS API 関数は TET PDF IFilter では利用できませんが、文書に関する情報をインデックスするために、いろいろな pCOS パスを XMP 構成ファイル内の表現として与えることが可能です。

表 2.1 に、よく用いられるいくつかの PDF オブジェクトに対する pCOS パスを挙げます。多くの pCOS オブジェクトは配列によって表現されており、これは配列添字を角カッコ内に必要とします。たとえば *pages[o]* は先頭ページを意味します（ページは 0 から始めてカウントされるのです）。TET 製品（TET PDF IFilter の基礎を成す）が対応しているすべての pCOS パスだけではなく、IFilter は、pCOS パスに対する右記の文法拡張に対応しています：「*」（アステリスク）ワイルドカードキャラクタを、配列または辞書の添字のかわりに用いることが可能です。これは、TET PDF IFilter が、その配列添字の可能な値すべてをなめて、それぞれのオブジェクト値をすべて、インデックス作成処理に含めることを意味します。1 個の pCOS パス内で、任意の数のワイルドカードが使えます。

表 2.1 いろいろな PDF オブジェクトに対する pCOS パス

pCOS パス	型	説明
length:pages	数値	その文書内のページ数
bookmarks[*]/Title	文字列	その文書内のすべてのしおり
pages[*]/annots[*]/Contents	文字列	全ページ上のすべての注釈の内容
/Info/Title	文字列	標準文書情報フィールド Title
/Info/ArticleNumber	文字列	カスタム文書情報フィールド ArticleNumber（文書情報項目は任意の名前を用いることができます）
/Root/Metadata	ストリーム	その文書のメタデータを持つ XMP ストリーム
images[*]/Metadata	ストリーム	その文書内の全画像に対する XMP メタデータストリーム群
fonts[*]/name	文字列	フォントの名前
length:fonts	文字列	文書内のフォント数
length:images	文字列	文書内の画像数
fonts[*]/embedded	論理値	フォント内の埋め込みステータス
pages[*]/width	数値	ページの表示領域の幅
pages[*]/annots[*]/A/URI	文字列	全ページ上の Web リンク群のターゲット URL
tagged	論理値	その文書が構造情報を持ったタグを含んでいるなら true

メタデータソースについての XML 構成　メタデータプロパティ群のソースは、XMP 構成ファイルの *Source* エレメント（*Property* エレメントの子）内で構成することも可能です。*pdfObject* 属性は、PDF オブジェクトに対する拡張 pCOS パスを内容とし、一方、*xmpName* 属性は、XMP プロパティのスキーマ接頭辞と名前を内容とします：

```
<Source pdfObject="/Info/ArticleNumber"/>
<Source xmpName="acme:number"/>
```

1 個のプロパティに対して、1 個ないし複数のソースを構成することが可能です。

2.2 メタデータの編成

メタデータは下記のように階層構造に編成されています：

- **プロパティ**は、メタデータのための基本的な構成単位です。Windows オペレーティングシステムと IFilter インタフェースの中のプロパティ群は、一意な数値識別子によって編成されています（後述）。
- **プロパティセット**は、プロパティ群のグループで構成され、そのプロパティ群は通常、何らかの論理的な関係を持っています。1 つの集合の中のすべてのプロパティは同一の GUID（後述）を共有しています。プロパティセット群は XML 構成ファイル内で指定することができます。
- **プロパティセットコレクション**は、プロパティセット群で構成され。TET PDF IFilter は、いくつかの定義済みのプロパティセットコレクションを実装しています。これらを用いると、複数のプロパティセットを一緒にまとめて有効にしたり無効にしたりすることができます。追加のプロパティセットコレクションを構成する必要はありません。

プロパティ識別と GUID　プロパティは、IFilter インタフェース内で、2 部分から成る識別子によって識別されます：

- 1 番目の部分は GUID（グローバル一意識別子）です。UUID（ユニバーサル固有識別子）もいいます。RFC 4122 に従った、大文字・小文字を区別しない 16 進表記による、一意な 128 ビットの識別子です。その各部分をダーシキャラクタ「-」で区切る必要があります。GUID を作成するにはさまざまなツールが利用可能です。たとえば下記のオンラインサービスなどです：

  www.uuidgenerator.net/

  GUID は右記のような形をしています：*7a737220-0cd0-11dd-bd75-0002a5d5c51b*.
- 2 番目の部分は、プロパティを、そのプロパティセット内部で一意に識別します。これは、**識別子**という、または略して *ID* という正の整数から成る場合があります。1 つのセットの中のプロパティ識別子群は値 2 で始まる必要がありますが、それ以外については任意です。プロパティ識別子には、すべての IFilter クライアントが対応しています。あるいは、この 2 番目の部分は平文名から成る場合もあります。ID のかわりに名前を用いることは非推奨であり、Windows Search などいくつかの IFilter クライアントはこれに対応していません。しかし、SharePoint などこれに対応している IFilter クライアントにおいては、これで構成がより便利になる可能性があります。GUID+ 名前方式を有効にすることについては、29 ページの「プロパティの GUID+ 名前処理のための XML 構成」を参照してください。

GUID+ID または GUID+ 名前の組み合わせは、メタデータプロパティクエリを検索製品内で構成するために必須です。メタデータプロパティのこの他の側面は 28 ページの 2.5「カスタムプロパティ」で解説します。

2.3 定義済みプロパティ

下記の定義済みプロパティセットコレクションが TET PDF IFilter に内蔵されています：

- **シェルプロパティ群**は、Windows に知られており、ユーザーフレンドリな名前を持っています。TET PDF IFilter は、PDF 文書に同等物を持つすべてのシェルプロパティに内容を与えます。シェルプロパティの値は、文書情報フィールドと文書 XMP メタデータから取得されます。*System.Author*・*System.Title*・*System.Document.DateCreated* などがよく利用されます。85 ページの A.1「シェルプロパティセットコレクション」に一覧があります。詳しい情報は下記にあります：

  msdn.microsoft.com/en-us/library/windows/desktop/dd561977(v=vs.85).aspx

- *PDF* プロパティ群は、PDF 文書に特有です。これらは pCOS パスから内容を与えられます。デフォルトで有効化されています。たとえば PDF バージョン番号、その文書が準拠している PDF/A 標準、しおり内容、ページサイズなどがあります。86 ページの A.2「PDF プロパティセットコレクション」に全一覧があります。
- **文書 *XMP*** プロパティ群は、XMP 仕様内の文書プロパティを多く網羅しており、デフォルトでは無効化されています。87 ページの A.3「文書 XMP メタデータプロパティセットコレクション」に全一覧があります。
- **画像 *XMP*** プロパティ群は、文書内の画像群に添付されている XMP プロパティを網羅しており、デフォルトでは無効化されています。89 ページの A.4「XMP 画像メタデータプロパティセットコレクション」に全一覧があります。
- **内部プロパティ群**は、業務用途ではなく開発・デバッグ支援として意図されたものです。デフォルトで有効化されており、ソフトウェアバージョン番号・インデックス作成時刻などがあります。90 ページの A.5「内部プロパティセットコレクション」に全一覧があります。

プロパティセットコレクションを有効にするための XML 構成　定義済みプロパティセットコレクションは、*Metadata/PropertySetCollection* エレメント内の対応する属性によって有効にすることができます：

```
<PropertySetCollection
        shell="true"
        pdf="true"
        documentXmp="true"
        imageXmp="true"
        internal="true"/>
```

デフォルトでは、*Shell*・*PDF*・*Internal* プロパティセットコレクションは有効になっており、*documentXMP*・*imageXMP* プロパティセットコレクションは無効になっています。カスタムプロパティは指定されると同時に有効になります。

あらゆる定義済みとカスタムのメタデータプロパティの処理を、*Filtering* エレメントの *metadataHandling* 属性で完全に無効にすることも可能です：

```
<Filtering metadataHandling="ignore">
```

注記 定義済みプロパティを有効化または無効化した場合には、インデックスを再作成する必要があります。

2.4 定義済みプロパティを用いた例

8 ページの 1.1.2「対話的検索」では、Windows Search におけるシェル・定義済みプロパティの使用例をいくつか挙げました。以下、定義済みプロパティに基づく、より高度なクエリを演示します。

一意 XMP 文書識別子　XMP は、さまざまな一意識別子を、XMP Media Management プロパティセット内の、以下のプロパティにおいて定義しています：

```
xmpMM:DocumentID
xmpMM:InstanceID
xmpMM:VersionID
xmpMM:OriginalDocumentID
```

Adobe InDesign などのアプリケーションはこれらの XMP プロパティを出力します。TET PDF IFilter は定義済みプロパティ *PDFlib.TET.xmpMM.DocumentID* などを用いてこれらに対応しています。これらの文書識別子は以下のように利用できます：

- *xmpMM:InstanceID* は、1 個の文書のそれぞれの更新バージョンについて一意なエントリとして作成されます。ファイル名にかかわらず、複製を特定するために利用できます。
- *xmpMM:OriginalDocumentID* は、1 個の文書が更新されても不変のままです。これを利用して、文書の更新版や流用文書を特定することが可能です。

これらのプロパティを使用するには、以下の XML 構成スニペットを用いて、XMP メタデータプロパティセットコレクション（プロパティの全一覧を見るには 89 ページの A.4「XMP 画像メタデータプロパティセットコレクション」を有効化する必要があります：

```
<n:PropertySetCollection documentXmp="true" imageXmp="true" internal="true" pdf="true"
        shell="true"/>
```

Photoshop XMP 画像メタデータ　プロフェッショナルな画像は、写真家やエージェンシーの名前や、その画像が撮影された町や国の名前に関するメタデータを含んでいることがあります。Adobe Photoshop などのアプリケーションはこのようなプロパティ群を、その画像に付属する XMP メタデータの中に記録します。この XMP は、PDF 内のその画像にも付属して格納されていることがあります。

これらのプロパティを使用するには、以下の XML 構成スニペットを用いて、XMP 画像メタデータプロパティセットコレクション（プロパティの全一覧を見るには 89 ページの A.4「XMP 画像メタデータプロパティセットコレクション」を参照）を有効化する必要があります：

```
<n:PropertySetCollection documentXmp="true" imageXmp="true" internal="true" pdf="true"
        shell="true"/>
```

これは画像関連の XMP メタデータを出力するよう TET PDF IFilter に指示します。インデックスを再作成した後、画像関連の XMP プロパティ群を検索において使用できるようになります。Windows Search における以下のクエリは、補足カテゴリ TEN（＝テニス）を有する画像を含んでいる文書の一覧を取得します。ただしその画像群が適切な XMP メタデータを保持していることが前提です：

```
PDFlib.TET.images.photoshop.SupplementalCategories:=TEN
```

完全な正準名のかわりに、もっと短いその表示名をこのクエリで使用することも可能です：

```
images.photoshop.SupplementalCategories:=TEN
```

2.5 カスタムプロパティ

カスタムメタデータプロパティは、企業・組織・業界などの個別の要請に応えるために定義済みプロパティのほかに追加するプロパティです。TET PDF IFilter を使用すると、カスタムプロパティを完全に制御できます。カスタムプロパティが TET PDF IFilter によって生成されて検索エンジンによってインデックスされるよう、構成ファイル内で指定することができます。たとえば、すべての文書の中で製品番号をメタデータプロパティとして割り当てておけば、文書を内容によってではなく製品番号によって検索することが可能になります。

カスタムデータプロパティを計画　カスタムプロパティを指定するためには、下記の諸側面を考慮する必要があります（GUID・識別子・フレンドリ名について詳しくは 24 ページの「プロパティ識別と GUID」を参照してください）：

- 1 個ないし複数のプロパティをプロパティセット内にグループ化することができます。各プロパティセットは、GUID という一意な 128 ビット識別子を必要とします。
- プロパティ**識別子**は、そのプロパティをそのプロパティセット内で識別する一意な整数です。セット内でプロパティ識別子は値 2 から始まります。IFilter クライアントによっては、この識別子をフレンドリ名へ置き換えることも可能です。定義済みのプロパティ群を、対応する GUID+ID 組み合わせを割り当てることによって上書きすることも可能です。
- プロパティに対する**フレンドリ名**は、識別子が利用可能な場合にはオプショナルであり、そうでない場合には必須です。これは任意の名前をとることができますが、構成ファイル内で一意な名前でなければなりません。SharePoint などいくつかの IFilter クライアントに対しては、これを識別子のかわりに使用することも可能ですが、それ以外の IFilter クライアント（Windows Search など）においてはフレンドリ名は動作しません。
- プロパティ**ソース**：プロパティは、21 ページの 2.1「PDF 内のメタデータソース」に従って、文書メタデータから、あるいは一般的 PDF 情報から、内容を与えられることができます。
- プロパティの**データ型**：*Int32*（32　ビット整数）・*Double*（倍精度の浮動小数点数）・*Boolean*（true/false）・*DateTime*（後述）・*String*。
- **優先権**規則：そのプロパティに対して複数のデータソースがあるときは、最初に得られる空でないデータソースが優先権を有する（すなわち、それより後のソース群は無視される）か、あるいはすべての空でないソースからのデータが収集されるかを、指定することが可能です。
- そのプロパティが**ベクトル**として出力されるかどうか、すなわち、単値ではなく複数の値がIFilterインタフェースへ配列構造で受け渡されるかどうかを指定します（30ページの「多値プロパティ」を参照）。
- プロパティがフルテキストの一部としてインデックスされる場合にそのプロパティ名の頭に付加される**接頭辞**（37 ページの 2.7「プロパティをテキストとしてインデックス」を参照）。

DateTime プロパティ型　時点を指定するには、プロパティに対して *DateTime* データ型を用いることができます。*DateTime*プロパティ群について生成される出力はつねに、IFilterインタフェースによって必要とされる形式になっていますが、入力は、そのプロパティのソースに応じてさまざまな形式に対応しています：

- データソースが　pCOS　パスの場合、たとえば標準文書情報フィールドや、*/Info/CustomDate* といったカスタム文書情報フィールドの場合には、その値は、ISO 32000-1 の 7.9.7 項で仕様定義されている標準 PDF 日付形式であると見なされます。PDF 日

付文字列の一般形式は *D:YYYYMMDDHHmmSSOHH'mm* です。なお、PDF 日付形式は（後述の XMP 形式とは異なり）、秒の小数には対応していません。いくつか例を挙げます：

D:201609231858	GMT
D:201609231058-08'00'	同じ時点を米国太平洋標準時で
D:201609231958+01'00'	同じ時点を中央欧州標準時で

- データソースが XMP プロパティの場合（例：*xmp:ModifyDate*）には、その値は、XMP 2005 リファレンスで基本値型 *Date* について仕様定義されている形式であると見なされます。これは、ISO 8601[1] で仕様定義されている形式と同等です。XMP 日付文字列の一般形式は *YYYY-MM-DDThh:mm:ss.sTZD* であり、ここで *TZD* はタイムゾーン指定子（*Z* か *+hh:mm* か *-hh:mm*）です。なお、いくつかの部分はオプショナルです：XMP 日付は精度に応じて 6 段階の粒度をサポートしており、一方、PDF 日付形式はただ 1 段階の粒度しかサポートしていません。いくつか例を挙げます：

2016-09-23T18:58:30Z	GMT
2016-09-23T13:58:30-05:00	同じ時点を米国東部標準時で
2016-09-23T19:58:30+01:00	同じ時点を中央欧州標準時で

TET PDF IFilter は、IFilter インタフェース仕様に定められているとおり、*DateTime* プロパティ群を UTC へ正規化します。この結果として、*DateTime* プロパティ群に対する検索は、その PDF 文書が作成された場所のタイムゾーンにかかわらず、つねにローカルタイムゾーンと関連づけて実行することが可能です。

カスタムプロパティのための XML 構成　1 個ないし複数のカスタムプロパティを、*PropertySet* エレメント内で指定することができます。そこでは各 *Property* エレメントが、そのセット内の各プロパティを記述します：

```
<PropertySet guid="E544AFE6-13E2-40F1-A702-DCEBE8FB7B03">
        <Property friendlyName="MailTo" identifier="2" type="String">
            <Source xmpName="acme:mailto"/>
        </Property>
</PropertySet>
```

複数の PDF ソースを、同一の Windows プロパティへマップすることが可能です。*Property* エレメントの存在は自動的に、その指定されたプロパティに対する処理を有効にします。ただし、あらゆる定義済みとカスタムのメタデータプロパティの処理を、*Filtering* エレメントの *metadataHandling* 属性で完全に無効にすることも可能です：

```
<Filtering metadataHandling="ignore">
```

プロパティの GUID+ 名前処理のための XML 構成　TET PDF IFilter は、プロパティの**識別子**が利用可能な場合には、GUID+ID を用いて IFilter インタフェース内でプロパティを識別します。*identifier* 属性を持たない、*friendlyName* 属性のみを持つカスタムプロパティについては、かわりに GUID+ 名前によって識別されます。定義済みプロパティについても GUID+ 名前処理を有効にする（そしてグローバルに GUID+ 名前処理を強制する）ためには、*Filtering* エレメントの *useIdentifier* 属性を用いることができます：

```
<Filtering useIdentifier="false">
```

1. www.w3.org/TR/NOTE-datetime を参照。

定義済みプロパティに対して用いられる名前は、表 2.2 で示すプロパティ接頭辞から頭の *TET_* と末尾のアンダースコア「_」を削ったものになります（例：*System_Author*・*pdfversion*・*dc_contributor*・*photoshop_DateCreated*）。

環境によっては、とりわけ SharePoint では、プロパティに対して GUID+ 名前処理を使うことは、GUID+ID よりも便利です。

特定のプロパティ値を抑制　場合によっては、特定のプロパティ値を抑制するほうが望ましいときもあります。たとえば、関心のないデフォルトを表している値はドロップして、関心のある値のみを IFilter クライアントへ通すなどです。これは *suppressValue* 属性を用いて実現できます。これを *Source* エレメントに対して与えると、そのソースによって返された特定の値が抑制されます。これを *Property* エレメントに対して与えると、そのプロパティに対して構成されたすべてのソースが処理され、その結果がその *suppressValue* 属性と比較されて、もし合致すればドロップされます。

多値プロパティ　メタデータプロパティは、1 個ないし複数の値を内容とすることができます。**単値プロパティは**、その文書を全体として記述する 1 個の単値から成ります。単値プロパティの例としては、作成日（プロパティソース：*xmp:CreateDate*・*/Info/CreationDate*）や一意な文書識別子（*dc:identifier*）があります。

多値プロパティは、文書あたり複数回出現する可能性があります。多値プロパティの例としては、文書作成者群のリスト、文書キーワード群のリストなどがあります。TET PDF IFilter を用いて多値プロパティを作成するにはいくつかの方法があります：

- そのプロパティのソースが XMP コンテナ型であり、それゆえ、1 度に複数の項目を保持しうる。例：*dc:creator* は XMP 内で型 *Seq* を持ちます。
- そのプロパティが、ワイルドカードを持つ多値な pCOS パスから内容を与えられる。ここでワイルドカードは、任意の数の個別の項目へ展開されえます。例：*bookmarks[*]/Title*。
- そのプロパティが、複数の *Source* エレメントを用いて定義され、かつ、そのプロパティの *precedence* 属性が値 *try-all* を持っている。例：*pdf:Keywords*・*/Info/Keywords*。これに対し、デフォルトの *precedence=first-wins* は、最初の空でないプロパティソースのみを処理します。

プロパティのベクトル処理　デフォルトでは、TET PDF IFilter はプロパティ定義内のすべての関連するソースを処理し（*precedence* 属性に従って）、そして利用可能なかぎりの空でないプロパティ値を出力します。いいかえれば、それぞれの値は IFilter クライアントに対して 1 個の値として返されます。複数のプロパティ値が不特定の順序で返されます。

あるいは、多値のプロパティを IFilter クライアントに対してベクトルとして与えることも可能です。これは、1 個ないし複数の値を保持できる 1 個の配列実体が出力されることを意味します。プロパティに対するベクトル処理の関連する側面がいくつかあります：

- SharePoint は、多値プロパティに、それがベクトル実体として処理される場合にのみ対応しています。
- IFilterクライアントによっては、たとえばWindows Searchなど、多値ベクトルプロパティ内の 1 個ないし複数の値を 1 回のクエリで検索できるベクトルクエリに対応しているものもあります。

2 つの別個の概念があることに留意してください：**多値**とは、そのプロパティのソースのありようを表していますが、**ベクトル処理**とは、プロパティ値群が IFilter クライアントへ転送される方式を表しています。ベクトル処理は、多値プロパティに対して、それが 1 個の値のみを内容とする場合であっても、適用することができます。

定義済みプロパティのうちのいくつかは多値です（付章 A「定義済みメタデータプロパティ」を参照）。

ベクトルプロパティのための XML 構成　カスタムプロパティに対するベクトル処理は、*Property* エレメントの *emitAsVector* 属性で有効にすることができます：

```
<Property friendlyName="MailTo" type="String" precedence="try-all" emitAsVector="true">
        <Source xmpName="acme:mailto"/>
        <Source xmpName="gov:mailto"/>
</Property>
```

2.6 カスタムプロパティを用いた例

以下の例において、いくつかのカスタムメタデータプロパティを演示します。TET PDF IFilter XML 構成スニペットだけでなく、照応する Windows Search 用 *.propdesc* スニペットも示します。*.propdesc* ファイルと、照応する *proptool* ユーティリティに関する詳細は、41 ページの 3.1.2「カスタムプロパティ」にあります。

カスタムメタデータのための共通手順　すべてのサンプルにおいて、以下のステップが必須です：

- TET PDF IFilter のための XML 構成ファイルに新規のプロパティ定義群を追加。いくつかの場合においては、この XML 構成ファイルの中で定義済みプロパティセットコレクションを有効化する必要もあります。XML 構成ファイル *starter_samples.xml* は、以下のすべての例のためのエントリを内容として持っていますので、これを出発点としてご利用いただけます。これを、67 ページの 5.1「構成ファイルの作業」で詳しく示すように、レジストリ内に構成する必要があります。この *starter_samples.xml* 構成ファイルは、Windows Search とともに使用するように作成されています。これを SharePoint または SQl Server とともに使用するには、若干の変更を加える必要があります（それぞれ 45 ページの表 3.2、50 ページの表 3.5 を参照）。
- 新規のプロパティをWindows Searchで使用するには、それを然るべきXMLプロパティ記述ファイルを用いて構成し、下記のようにそれを Windows に登録する必要があります（〈Register property descriptions in Windows〉, page 38 を参照）：

  ```
  proptool --register starter_samples.propdesc
  ```

  これで、Windows エクスプローラーを、この新規プロパティを表示するよう構成することができるようになります（10 ページの「プロパティクエリ」を参照）。プロパティ記述ファイル *starter_samples.propdesc* は、以下のすべての例のためのエントリを内容として持っていますので、これを出発点としてご利用いただけます。
- 新規プロパティを収集するためにインデックスを再作成します。新規の検索またはエクスプローラーウィンドウを開くことによって初めて、その詳細ダイアログ内で追加のプロパティ列を有効化できるようになります。

フォントまたは画像の数のための pCOS プロパティ　1 つの PDF 文書の中のフォントの総数は、pCOS パス *length:fonts* を用いてクエリできます。画像についても同様です。これらのプロパティを出力するには、*PropertySet* エレメント内で、以下の XML 構成スニペットを使用します：

```
<!-- PDFプロパティ： 文書内のフォントと画像の数 -->
<n:Property friendlyName="fontcount" type="Int32" identifier="2">
        <n:Source pdfObject="length:fonts" />
</n:Property>

<n:Property friendlyName="imagecount" type="Int32" identifier="3">
        <n:Source pdfObject="length:images" />
</n:Property>
```

この新規の *fontcount*・*imagecount* プロパティを Windows Search で使用するには、これを Windows プロパティ記述ファイル内で、以下の *.propdesc* スニペットを用いて構成・登録する必要があります：

```
<!-- 文書内のフォントの数 -->
<propertyDescription name="fontcount" formatID="{E544AFE6-13E2-40F1-A702-DCEBE8FB7B03}"
        propID="2">
   <typeInfo type="Int32" isInnate="true" isViewable="true"/>
   <labelInfo label="fontcount"/>
   <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>

<!-- 文書内の画像の数 -->
propertyDescription name="imagecount" formatID="{E544AFE6-13E2-40F1-A702-DCEBE8FB7B03}"
        propID="3">
   <typeInfo type="Int32" isInnate="true" isViewable="true"/>
   <labelInfo label="imagecount"/>
   <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>
```

これで、フォントと画像の数をエクスプローラーウィンドウ内の新規列に表示させることができるようになりましたので、下記のクエリを用いて、特定の数のフォントまたは画像を有する文書を検索できます：

```
imagecount: >10
```

PDF 規格のための pCOS プロパティ　定義済みプロパティ *PDFlib.TET.pdfa*・*PDFlib.TET.pdfx* 等は、文書の規格準拠状況を記述します。これらのプロパティは、照応する pCOS パスに基づいています。PDF 文書がいずれの種類の PDF/A 規格にも準拠していない場合には、その pCOS 疑似オブジェクト *pdfa* は *none* を返します。ただし、プロパティ *PDFlib.TET.pdfa* は値 *none* を抑制しています。以下の例では、多値のカスタムプロパティ *Standards* を定義します。このプロパティは、個別の定義済みプロパティ *PDFlib.TET.pdfa*・*PDFlib.TET.pdfx* 等と同様に動作するものとしますが、すべての規格準拠エントリをただ 1 個のベクトルプロパティ内に出力するものとします。これは、照応する pCOS パス *pdfa* 等をソースとして用いるものとし、ただし関心のない値 *none* をドロップするものとします。以下の XML 構成スニペットは、*PropertySet* エレメント内に入れることにより、*Standards* プロパティを定義します：

```
<n:Property friendlyName="Standards" emitAsVector="true" suppressValue="none"
        identifier="4" emitAsVector="true" precedence="try-all">
        <n:Source pdfObject="pdfa"/>
        <n:Source pdfObject="pdfe"/>
        <n:Source pdfObject="pdfx"/>
        <n:Source pdfObject="pdfua"/>
        <n:Source pdfObject="pdfvt"/>
</n:Property>
```

この新規の *Standards* プロパティを Windows Search で使用するには、これを Windows プロパティ記述ファイル内で、以下の *.propdesc* スニペットを用いて構成・登録する必要があります：

```
<propertyDescription name="Standards" formatID="{E544AFE6-13E2-40F1-A702-DCEBE8FB7B03}"
        propID="4">
           <typeInfo type="String" multipleValues="true" isInnate="true"
                isViewable="true"/>
           <labelInfo label="Standards"/>
```

```
        <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>
```

これで、エクスプローラーウィンドウ内に ***Standards*** 列を表示させることができるようになりました。この列には、すべてのあてはまる規格が、セミコロンで区切られて表示されます。例：

```
PDF-A-2a; PDF/UA-1
```

望ましくないエントリ ***none*** は抑制されます：文書が、認識されているいずれの規格にも準拠していない場合には、この ***Standards*** 列は空のままとなります。PDF/A 文書を検索するには、このプロパティ値が PDF/A を含んでいるかどうかをチェックする、下記のクエリを使用できます：

```
Standards:~<PDF/A
```

下記は、より限定して、規格の特定の種類 PDF/A-3b を検索します：

```
Standards:=PDF/A-3b
```

PDF 署名種別のための pCOS プロパティ　pCOS パス *signaturefields[o]/sigtype* は、最初の署名フィールドの種別をチェックします。これは、文字列 ***none*・*approval*・*certification*・*doctimestamp*** のうちいずれか 1 つを返します。これに照応するプロパティ定義においては、値 ***none*** は、あまり有用ではないので、抑制しましょう。以下の XML 構成スニペットは、***PropertySet*** エレメント内に入れることにより、TET PDF IFilter を、文書内のすべての電子署名に関する情報を出力するよう構成します。署名の数はあらかじめわかりませんので、アスタリスクキャラクタ「*」をワイルドカードとして与えます：

```
<n:Property friendlyName="Signature" suppressValue="none" identifier="5">
        <n:Source pdfObject="signaturefields[*]/sigtype"/>
</n:Property>
```

このプロパティは、文書内のすべての署名種別をレポートします。なお、同じ種別の署名が複数ある場合には、1 回のみレポートされます。なぜなら同じプロパティ値が複数回生じることはできないからです。この ***Signature*** プロパティを Windows Search で使用するには、これを Windows プロパティ記述ファイル内で、以下の ***.propdesc*** スニペットを用いて構成・登録する必要があります：

```
<propertyDescription name="Signature"
        formatID="{E544AFE6-13E2-4OF1-A7O2-DCEBE8FB7BO3}" propID="5">
           <typeInfo type="String" multipleValues="true" isInnate="true"
            isViewable="true"/>
           <labelInfo label="Signature"/>
           <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>
```

これで、署名ステータスをエクスプローラーウィンドウ内の新規列に表示させることができるようになりましたので、下記のクエリを用いて、このプロパティが空でない値を持っているかどうかをチェックすることによって、署名済み文書を検索することができます：

```
Signature: <>[]
```

この pCOS オブジェクトで使える値は *none*（署名されていないフィールドの場合）・*approval*・*certification*・*doctimestamp* です（pCOS パスリファレンス参照）。上記の式は、署名種別が *none* 以外であるかどうかをチェックするものです。

ユーザー定義文書情報フィールドのための pCOS プロパティ　タイトル・サブタイトル・作成者など、標準 PDF 文書情報項目の多くは、照応するシェルプロパティへ自動的にマップされます（85 ページの A.1「シェルプロパティセットコレクション」参照）。これらの標準項目のほかにも、インデックスされた PDF 文書の中にユーザー定義情報項目があれば、それらを出力することも可能です。以下の XML 構成スニペットは、*PropertySet* エレメント内に入れることにより、*InvoiceNumber* という情報項目から値を得る *invoicenumber* プロパティを定義します。

```
<n:Property friendlyName="invoicenumber" identifier="6">
|       <n:Source pdfObject="/Info/InvoiceNumber"/>
</n:Property>
```

このプロパティを Windows Search で使用するには、これを以下の *.propdesc* スニペットを用いて構成・登録する必要があります：

```
<propertyDescription name="invoicenumber"
        formatID="{E544AFE6-13E2-40F1-A702-DCEBE8FB7B03}" propID="6">
        <typeInfo type="String" isInnate="true" isViewable="true" isQueryable="true"/>
        <labelInfo label="invoicenumber"/>
        <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>
```

これで、インボイス番号をエクスプローラーウィンドウ内の新規列に表示させることができるようになりましたので、下記のクエリを用いてインボイス番号を検索できます：

```
invoicenumber: =R123456
```

ZUGFeRD インボイスを識別するための XMP プロパティ　ZUGFeRD 規格[1]は、カスタム XMP プロパティ群を有する PDF/A 文書を使用することによって、インボイスが ZUGFeRD 準拠であることを識別します。たとえば、以下の XMP 断片は基本的な ZUGFeRD インボイスを記述しています：

```
<rdf:Description
        rdf:about=""
        xmlns:zf="urn:ferd:pdfa:CrossIndustryDocument:invoice:1p0#"
        zf:ConformanceLevel="BASIC"
        zf:DocumentFileName="ZUGFeRD-invoice.xml"
        zf:DocumentType="INVOICE"
        zf:Version="1.0"/>
```

TET PDF IFilter は容易に、ZUGFeRD 準拠文書をクエリ内で、照応する ZUGFeRD XMP プロパティ群に基づいて識別できるよう構成できます。

以下の XML 断片は、必須の ZUGFeRD 名前空間 URI と接頭辞を、規格で定められているとおりに宣言します：

```
<n:PrefixDeclarations>
        <n:PrefixDeclaration
```

1. ZUGFeRD について詳しくは www.pdflib.com/knowledge-base/pdfa/zugferd-invoices/ を参照

```
            uri="urn:ferd:pdfa:CrossIndustryDocument:invoice:1p0#" prefix="zf"/>
</n:PrefixDeclarations>
```

以下の XML 構成スニペットは、***PropertySet*** エレメント内に入れることにより、TET PDF IFilter を、XMP プロパティ ***zf:DocumentType*** をカスタムプロパティ ***zugferd*** 内へ出力するよう構成します。ZUGFeRD 準拠文書であるためには、これが値 ***INVOICE*** を内容としている必要があります：

```
<n:Property friendlyName="Zugferd" identifier="7">
        <n:Source xmpName="zf:DocumentType"/>
</n:Property>
```

この新規の ***zugferd*** プロパティを Windows Search で使用するには、これを Windows プロパティ記述ファイル内で、以下の ***.propdesc*** スニペットを用いて構成・登録する必要があります：

```
<propertyDescription name="Zugferd"
        formatID="{E544AFE6-13E2-4OF1-A7O2-DCEBE8FB7BO3}" propID="7">
           <typeInfo type="String" isInnate="true" isViewable="true"/>
           <labelInfo label="Zugferd"/>
           <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>
```

これで、ZUGFeRD 準拠ステータスをエクスプローラーウィンドウ内の新規列に表示させることができるようになりましたので、下記のクエリを用いて ZUGFeRD 文書を検索できます：

```
Zugferd:=INVOICE
```

2.7 プロパティをテキストとしてインデックス

多くのテキスト検索エンジンは、何らかの方式でのプロパティクエリに対応しています。しかし、プロパティに対するクエリは、SQL Server のような、フルテキスト検索にのみ対応している検索製品では、可能でない場合もあります。あるいは別のシナリオとして、明示的にプロパティを検索することは、もしもその検索語のあらゆる出現を、それが文書内容の中に出現するか、それとも何らかのプロパティの中に出現するかにかかわらず検索したい場合には、望ましくないかもしれません。いずれの場合においても、TET PDF IFilter に対して、すべてのプロパティをフルテキストインデックスの中へ入れ込むよう指示することが可能です。プロパティを文書内容本体から区別するために、TET PDF IFilter は、文字列をプロパティ値の頭に付加することも可能です。これによって、検索エンジンがプロパティ検索に直接は対応していない場合においても、プロパティをプロパティであると見分けることが容易になります。付章 A「定義済みメタデータプロパティ」に挙げる定義済みプロパティに対しては、正準プロパティ名が接頭辞として用いられます。

プロパティをテキストとしてインデックスすれば、もしそれをしなかったらプロパティ検索に一切対応していない環境において、限定的なプロパティ検索が可能になる一方で、そのプロパティ検索は限定的なものであるという事実は知っておくべきです。たとえば、論理型などの表現がプロパティ値として得られません。

メタデータをテキストとしてインデックスするための XML 構成　メタデータプロパティをテキストとしてインデックスするには、*Filtering* エレメントの *metadataHandling* 属性を *propertyAndText*（プロパティをメインテキスト内へ透明な形で混ぜ込みたい場合）か *propertyAndPrefixedText*（プロパティを接頭辞で識別したい場合）に設定します：

```
<Filtering metadataHandling="PropertyAndPrefixedText">
```

文書をフィルタする際にカスタムプロパティの頭に付加されるオプショナルな接頭辞は、カスタムプロパティに対して *Property* エレメントの *textIndexPrefix* 属性で指定することができます：

```
<Property friendlyName="Title" identifier="7" textIndexPrefix="TITLE_">
...
</Property>
```

定義済みプロパティの頭に付加される接頭辞は、下記の方式に従って構築されます：

```
TET_<プロパティ名>_
```

ここでプロパティ名の中のピリオドキャラクタ「.」はアンダースコアキャラクタ「_」へ置き換えられます（表 2.2 の例を参照してください）。

表 2.2 メタデータプロパティをテキストとしてインデックスするための接頭辞

プロパティセット コレクション	サンプルプロパティ名	メタデータをテキストとして インデックスするための接頭辞
Shell	System.Author	TET_System_Author_
TET	PDFlib.TET.fullpdfversion	TET_fullpdfversion_
XMP	dc:contributor	TET_dc_contributor_
Image	photoshop:DateCreated	TET_photoshop_DateCreated_

シナリオ 1：メタデータプロパティをメインテキスト内へ透明な形で混ぜ込む　もしメタデータプロパティが、ターゲット文書（群）を特定する充分に弁別的なテキストを内容として持っているならば、プロパティをフルテキストインデックスへ入れ込んで、それを標準的なフルテキストクエリへ入れ込めば充分でしょう。たとえば、特定の記事番号をクエリする場合には、その番号がその文書のメインテキスト内に出現しようが、それともメタデータプロパティ内に出現しようが、探しているその記事番号について語っている文書を 1 個に絞り込めるかぎりは、どちらでもかまわないのです。いいかえれば、テキストがメインテキスト内に出現しようが、それとも何らかのメタデータプロパティ内に出現しようが、どちらでもかまわない場合には、単にプロパティをフルテキストとしてインデックスすることを有効にする必要があり、それを超える手順は必要ありません。

メタデータをメインテキスト内へ透明な形で混ぜ込むには、下記の XML 構成を用います：

```
<Filtering metadataHandling="propertyAndText">
```

シナリオ 2：メタデータをメインテキストから区別　別の場合においては、テキストがメイン文書内に出現しているのか、それとも何らかのメタデータプロパティ内に出現しているのかに、意味がある場合もあるでしょう。たとえば、作成者が *Doyle* である文書を検索するか、それともメインテキスト内に語 *Doyle* を含む文書を検索するかは、互いに大きな違いです。このシナリオにおいては、プロパティをフルテキストとしてインデックスすることを有効にするだけではなく、メイン文書内容中のテキストとメタデータプロパティ内のテキストとを区別できるよう、各プロパティに対して適切な接頭辞をも入れ込む必要があります。

定義済みプロパティ *System.Author* の値の頭には、接頭辞 *TET_System_Author_* が付加されます。たとえば、*System.Author=Doyle* に対するプロパティベースの検索を、*TET_System_Author_Doyle* に対するフルテキスト検索でエミュレートすることができます。*System.Author* は定義済みプロパティですので、これに対応する XML 構成はプロパティ独自の項目を一切必要とせず、単にプロパティを接頭辞付きテキストとしてインデックスすることを有効にする必要があります：

```
<Filtering metadataHandling="propertyAndPrefixedText">
```

記事番号 *XY123456* を持つ文書に対するプロパティベースの検索を、*ArticleNumber_XY123456* に対するフルテキスト検索でエミュレートするには、下記の XML 構成を用います：

```
<Filtering metadataHandling="propertyAndPrefixedText">

<PropertySet guid="404e8a40-2e85-11dd-97f6-0002a5d5c51b">
        <Property identifier="2" textIndexPrefix="ArticleNumber_">
                <Source pdfObject="/Info/ArticleNumber"/>
        </Property>
<PropertySet>
```

2.8 ページ内容を無視してプロパティのみを検索

場合によっては、ページ内容ではなくメタデータプロパティ群のみに基づいて検索を行うほうが望ましいこともあります。すなわち、インデックス作成を行う過程でページ内容を完全に無視して、メタデータプロパティ群にのみ依拠するということです。こうしたシナリオとして考えられるのは、下記のうちの 1 つないし複数です：

- ユーザーが発するクエリへの制御を強化：メタデータプロパティ群に基づいて正確な結果が得られるときに、長い結果リストを調べあげるのは無駄です。
- 検索したい文書群がだいたい同一の語群を内容としているが、ただしそれらが異なった組み合わせになっている。たとえば請求書などの取引文書。
- ページ内容本体があらかじめよくわかっているので、検索上実際の意味を持たない。しかし、特定の文書に対して集められたページ群が関心対象である。たとえば、可変の数の契約と紐付けられた保険取引では、契約の正確な文言ではなく、契約文書の数と種類が検索上意味を持つ。
- 検索したい文書群が検索可能なテキストを全く内容としていない。たとえばスキャンされた、OCR を全く実行されていない文書。
- インデックスに何ら意味のあるありようで貢献しない情報を内容とする文書群。たとえば数のみを内容とする長い財務文書や、テキストのない（ないし図中のキャプション内のテキストのみの）技術図面。
- 上述のいずれかの場合におけるパフォーマンス最適化：内容が検索の助けにならないとわかっているなら、文書をインデックスするのは無駄です。

ページ内容のインデックス作成を無効にするための XML 構成　ページ内容のインデックス作成を完全に無効にするには、*Filtering* エレメントの *indexPageContents* 属性を *false* に設定します：

```
<Filtering indexPageContents="false" metadataHandling="property">
```

3 各種 IFilter クライアントにおけるメタデータプロパティ

3.1 Windows Search におけるメタデータプロパティ

注記 最終的には SharePoint などの IFilter クライアントで作業を行いたい場合であっても、メタデータ処理を始めるにあたって Windows Search を使用することはできます。

3.1.1 定義済みプロパティ群

構成する Windows オペレーティングシステムは、メタデータプロパティ群を扱うための強力なプロパティシステムをサポートしています。このプロパティには、システムによって定義されているもの（シェルプロパティといいます）と、ユーザー定義（カスタム）プロパティとがあります。Windows Search は、この Windows プロパティシステムを使用して、システム（シェル）プロパティ群とユーザー定義メタデータプロパティ群の記述にアクセスします。

多くのプロパティが TET PDF IFilter にすでに内蔵されています（25 ページの 2.3「定義済みプロパティ」、付章 A「定義済みメタデータプロパティ」を参照）。これらのうちのいくつかは、Windows シェルプロパティへ与えられます。定義済みプロパティ群を用いて作業を行いたい場合であれば、それは比較的シンプルに構成できます：

- その定義済みプロパティ群の記述をWindowsが利用できるようにする必要があります。これは、TET PDF IFilterのインストール中にIFilterクライアントとして ***Windows Search*** を選択した場合には自動的に行われます。そうでない場合には、ファイル ***predefined_properties.propdesc*** を手作業で登録する必要があります（下記）。
- TET PDF IFilter 内のプロパティセットコレクションの多くはデフォルトで有効になっています。文書 XMP または画像 XMP プロパティ群で作業を行いたい場合にのみ、照応するコレクションを XML 構成ファイル内で有効化する必要があります（25 ページの「プロパティセットコレクションを有効にするための XML 構成」参照）。

Windows と TET PDF IFilter において構成したプロパティは、検索で使用できるようになります。

オプション ***--list _all*** は、システムに登録されているすべてのプロパティをダンプします。すべての定義済み非シェルプロパティの表示名は、その正準名の短縮形になります。たとえば ***PDFlib.TET.pdfa*** の表示名は ***pdfa*** です（UI 言語にかかわらず）。

3.1.2 カスタムプロパティ

Windows Search においてカスタムメタデータプロパティを用いて検索を行うには、それを Windows プロパティシステムと TET PDF IFilter の両方において構成する必要があります。なお、カスタムプロパティは、それが XML 構成ファイル内指定され次第暗黙的に有効化されます。

プロパティ記述ファイル Windows Search でカスタムメタデータプロパティを使うには、プロパティ群の名前・データ型・GUID などのプロパティ属性を指定するプロパティ記述ファイル（***.propdesc***）を用意する必要があります。このプロパティ記述は、TET PDF IFilter のための XML 構成ファイルの中の照応するプロパティ記述と整合している必要がありま

す。プロパティ記述は、下記で記述されている文法に従った XML ファイルとして指定される必要があります：

msdn.microsoft.com/en-us/library/bb773879(VS.85).aspx

この *.propdesc* 形式の XML スキーマ記述が *WDS_Property_Description_Schema.xsd* 内にあります。以下の断片はいくつかのプロパティ記述を演示します：

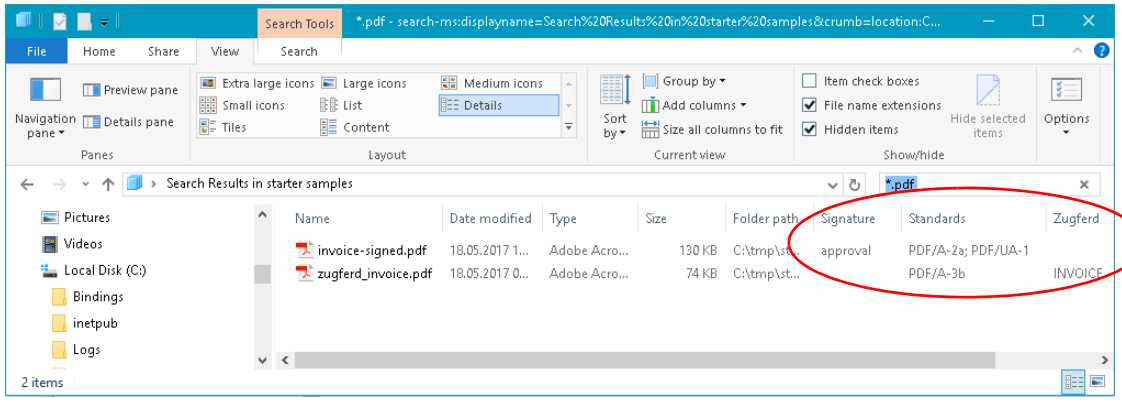
```
<propertyDescription name="fontcount" formatID="{5eac0060-1ba4-11dd-92c4-0002a5d5c51b}"
        propID="2">
  <typeInfo type="Int32" isInnate="true" isViewable="true"/>
  <labelInfo label="fontcount"/>
  <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>

<propertyDescription name="weblink"
  formatID="{5eac0060-1ba4-11dd-92c4-0002a5d5c51b}" propID="6">
  <typeInfo type="String" isInnate="true" multipleValues="true" isViewable="true"/>
  <labelInfo label="weblink"/>
  <searchInfo inInvertedIndex="true" isColumn="true"/>
</propertyDescription>
```

多値プロパティに対しては *multipleValues="true"* 属性が重要です（30 ページの「多値プロパティ」を参照）。なお、Windows ではプロパティ名は最長 64 キャラクタに制限されています。

注記 **TET PDF IFilter ディストリビューション内のファイル `starter_samples.propdesc` を出発点とするとよいでしょう。このファイルは、この章で以下示すすべての例のためのプロパティ記述を内容として持っています。**

プロパティ記述を Windows に登録　カスタムプロパティ記述については、TET PDF IFilter とともにインストールされているコマンドラインユーティリティ *proptool.exe*（このツールは Windows Vista 以上を必須とします）を用いて登録する必要があります。プロパティ記述ファイルの名前を与えると、それが Windows プロパティシステムで登録されます（パスが空白キャラクタを含む場合には、必ずパスをダブルクォーテーションでくくってください）：

図 3.1
Windows エクスプローラにカスタムプロパティの列を追加した検索結果



```
proptool --register acme.propdesc
```

これで、Windows エクスプローラを、この新規プロパティ群を表示するよう構成できるようになりました（10 ページの「プロパティクエリ」参照）。Windows プロパティシステムは、プロパティ記述ファイルの名前を、下記のレジストリキー群（および末尾要素の数を大きくしたもの）に格納しています：

```
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\PropertySystem\PropertySchema\0000
```

この *proptool* ユーティリティは、プロパティファイルがうまく登録できなかったとき、たとえばプロパティの重複を検出した場合などには、エラーメッセージと *HRESULT* エラー値を出力します。エラーが起きたとき、詳細を調べるには、アプリケーションイベントログをチェックします：

- **「スタート」→「設定」→「コントロール パネル」→「管理ツール」→「イベント ビューア」**
- 左のペーンで **「アプリケーション」** をクリック。
- プロパティ登録で問題があった場合には、ソース *Microsoft-Windows-propsys* の項目があります。その項目を内容としている行をダブルクリックして、エラーメッセージを調べます（例：*Omitted duplicate property*）。

あるいは、*proptool* が出力する *HRESULT* 値を用いて問題を分析することも可能です。*HRESULT* の値と説明の一覧は下記にあります：

msdn.microsoft.com/en-us/library/cc231198.aspx

XML 解析に関連したエラー番号のさらなる一覧が下記にあります：

https://msdn.microsoft.com/en-us/library/ms753129%28v=vs.85%29.aspx

Windows Search のためのプロパティ記述を登録する際には、以下の要件が満たされる必要があります：

- プロパティ記述はファイル名接尾辞 *.propdesc* を使用する必要があります。なぜなら Windows プロパティシステムは他の接尾辞を受け付けないからです。
- カスタムプロパティを検索で利用可能にするためには、プロパティ記述を登録して、検索サービスを停止させて再起動し、Windows Search オプションで再構築を強制する必要があります：

```
proptool --register "acme.propdesc"
net stop wsearch
net start wsearch
...カタログを再構築
   （8ページの「Windows Searchサービスを開始・停止」を参照）...
```

- 登録するプロパティは、既存の Windows プロパティ（シェルプロパティを含め）と GUID+ID または GUID+ 名前で衝突していてはいけません。Windows プロパティの一覧は下記にあります：

  msdn.microsoft.com/en-us/library/windows/desktop/dd561977(v=vs.85).aspx

- *proptool*ツールを走らせるには、*HKEY_LOCAL_MACHINE*レジストリハイブへ書き込める管理者権限が必要です（6 ページの「特権コマンドを実行」参照）。
- レジストリはプロパティ記述ファイルの名前を保持しているだけですので（その内容は保持していないので）、*propdesc* ファイルはその登録された場所にありつづけている

必要があります。そうでないとプロパティ記述が検索で利用可能でなくなってしまいます。

- *propdesc* ファイルは、すべてのユーザから読み込み可能である必要があります。
- 複数のプロパティ記述ファイルを登録することも可能です。ただし、2 つのプロパティ記述が同一のプロパティ（GUID+ID によって特定される）の記述を含んでいてはいけません。
- プロパティ記述ファイル群に変更を加え、1 つの *propdesc* ファイルを同じ名前で登録したり登録解除したりを繰り返した場合には（試験の際にはよくあることです）、再起動を行うことによって、Windows にすべてのプロパティ記述を正しく拾い上げさせる必要がある場合があります。

-unregister オプションを用いると、以前に登録されているプロパティ記述を除去することができます：

```
proptool -unregister "c:\property\acme.propdesc"
```

3.1.3 TET PDF IFilter におけるプロパティ

定義済み TET PDF IFilter プロパティ　すべての定義済みプロパティのためのプロパティ記述ファイル *predefined_properties.propdesc* が利用可能です。インストーラで ***Windows Search*** を IFilter クライアントとして選択した場合には、これはすでに Windows Search 用に登録されています。プロパティ定義は TET PDF IFilter に内蔵されていますが、クエリでそれを使うには、使いたいプロパティセットコレクションを XML 構成ファイル内で有効化する必要があります（25 ページの 2.3「定義済みプロパティ」参照）。

Windows Search のための XML 構成　表 3.1 に、Windows Search を使用する際の TET PDF IFilter のための XML 構成に関連する要件と推奨事項を挙げます。サンプル構成ファイル *config.xml* は、これらの要件を実装しており、出発点として利用できます。

表 3.1 Windows Search のための XML 構成

エレメント	属性	要件・推奨事項
Filtering	***useIdentifier***	true にする必要があります。なぜなら、Windows Search はプロパティ識別のためには GUID+ID 方式にのみ対応しており、GUID+ 名前には対応していないからです（29 ページの「プロパティの GUID+ 名前処理のための XML 構成」を参照）。
PropertySet-Collection	***shell***	Windows Search に知られているシェルプロパティ群に対応するには true にする必要があります（なお、true がもともとデフォルトです）。
Property	***identifier***	必須。なぜなら、Windows Search はプロパティ識別のためには GUID+ID 方式にのみ対応しており、GUID+ 名前には対応していないからです（29 ページの「プロパティの GUID+ 名前処理のための XML 構成」を参照）。

3.2 SharePoint におけるメタデータプロパティ

SharePoint におけるメタデータ処理は、Enterprise Search Administration 名前空間 *Microsoft.Office.Server.Search.Administration* を用いてプログラム的に制御できます。SharePoint におけるメタデータプロパティの処理については以下の概念が用いられます：

- **クロール対象プロパティ**は、TET PDF IFilter によって、PDF 文書をインデックス（クロール）する際に作成されます。クロール対象プロパティは複数の値を持つことができます。それは TET PDF IFilter 構成ファイルを通じて制御されます。SharePoint は、クロール対象プロパティを他のフィールドと同様に検索しますが、プロパティクエリのための高度な機能群は利用できません。
- **管理プロパティ**は、検索クエリと高度な検索で用いることができ、検索結果に表示されます。

「高度な検索」ページの管理プロパティに関する詳細は下記にあります：

msdn.microsoft.com/en-us/library/bb428648.aspx
msdn.microsoft.com/en-us/library/bb608302.aspx

SharePoint のための XML 構成 表 3.2 に、SharePoint を使用する際の TET PDF IFilter のための XML 構成に関連する要件と推奨事項を挙げます。サンプル構成ファイル *SharePoint config.xml* は、これらの要件を実装しており、出発点として利用できます。

表 3.2 SharePoint のための XML 構成

エレメント	属性	要件・推奨事項
Filtering	*useIdentifier*	定義済みプロパティを含めすべてのプロパティの GUID+ 名前処理を強制するには `false` に設定します。
Property	*emitAsVector*	複数の値を持ちうるプロパティ群に対しては `true` にする必要があります（30 ページの「多値プロパティ」を参照）。
Property	*friendlyName*	GUID+ 名前処理を使うと、リスト内でプロパティを見つけやすくなりますので、この属性の使用を推奨します（24 ページの「プロパティ識別と GUID」を参照）。

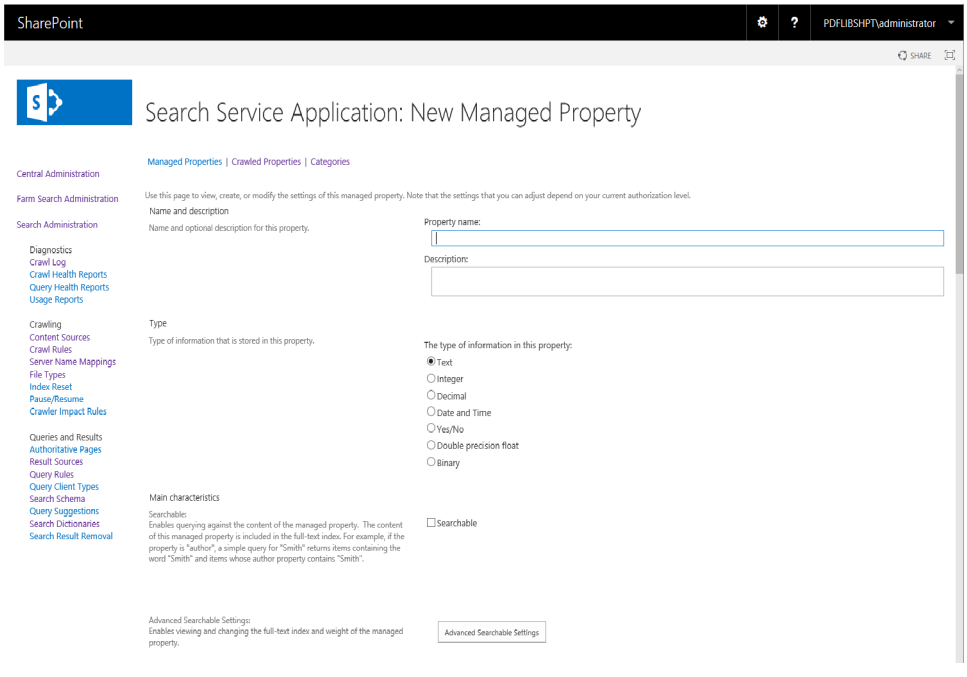
カスタムメタデータプロパティを構成 カスタムメタデータプロパティをインデックス作成のために用意するには下記のように操作します：

- フルクロールを走らせます。クロールの間、TET PDF IFilter は、XML 構成ファイル内で構成されている、かつ 1 個ないし複数の文書内で見つかったプロパティ群をレポートします。これは、SharePoint が新たに見つかったクロール対象プロパティ群をピックアップするために必要です。
  SharePoint は、すべての新規プロパティカテゴリに対して合成名（「*Category 1*」・「*Category 2*」など）を生成します。同じ GUID を持つすべてのクロールされたプロパティは同じ新規カテゴリ内へ収集されます。カテゴリ内の任意のクロールされたプロパティを見ることによって共通 GUID を知ることができます。SharePoint 管理ウェブサイト内の「*Application Management*」の下で「*Manage service applications*」→「*Search Service Application*」をクリックします。ページ「*Search Service Application: Search Administration*」の左側で「*Search Schema*」をクリックします。すると、ページ「*Search Service Application: Managed Properties*」が表示されます。このページの上端の「*Categories*」をクリックして、新規合成名を有するカテゴリをクリックします。任意

のクロールされたプロパティをクリックして、そのプロパティセット ID を見ることによって、その GUID を知ります。この GUID を、TET PDF IFilter のための XML 構成ファイルの中の ***PropertySet/@guid*** の中のものと見比べるか（カスタムプロパティを作成した場合）、あるいは付章 A「定義済みメタデータプロパティ」のプロパティセット GUID と見比べます（定義済みプロパティセットコレクション内のプロパティを使用の場合）。

- SharePoint　によって生成された合成カテゴリ名を、そのカテゴリを前の手順で知った GUID によって特定してから、ユーザーフレンドリな名前に変更：
  SharePoint：「***Category***」ページで、カテゴリ名に対するドロップダウンメニューをクリックして、「***Edit Category***」をクリック。「***Category name***」フィールドで名前を好きに編集して、OK をクリック。
- SharePoint：「***Search Service Application***」管理ページで、「***Managed Properties***」をクリック。
- 「***New Managed Property***」をクリック（図 3.2 参照）。
- プロパティの名前・説明・データ型を入力し、右下隅の、フィールド「***Mapping to crawled properties***」の右にあるボタン「***Add Mapping***」をクリック。
- ドロップダウンメニュー「***Filter on a category***」は、「***Crawled property selection***」と題されたリストボックスに表示されているクロール対象プロパティ群をフィルタします。このリストボックスは、プロパティと、そのカテゴリ名とプロパティ識別子またはフレンドリ名のリストを表示します。識別子かフレンドリ名かの選択は、TET PDF IFilter の XML 構成ファイル内のプロパティ定義に依存しますので、45 ページの「カスタムメタデータプロパティを構成」を参照してください。新規にクロールされたプロパティを選択し、その管理プロパティに名前を割り当てて、それを保存します。

図 3.2
SharePoint で管理プロパティを追加



管理プロパティへの検索のための SharePoint XML を作成　このステップでは、管理プロパティを「高度な検索」に追加するために必要な XML を作成します。次の項で、この XML を実際に適用する方法を説明します。以下、例とともにこの XML を説明します：

それぞれのプロパティについて、*PropertyDefs* エレメントの子として *PropertyDef* エレメントを作成する必要があります：

```
<PropertyDef Name="EbookDateOfBirth" DataType="datetime" DisplayName="Ebook Date of
Birth"/>
```

Name 属性はプロパティ名と対応している必要があり、*DataType* 属性はそのプロパティの型を表 3.3 に従って記述し、*DisplayName* はユーザインタフェースで表示される任意の名前を内容とします。

新規の管理プロパティを、PDF 文書群に対する検索で利用可能にするには、そのプロパティを *ResultType* エレメントに追加します：

```
<ResultType DisplayName="PDF Documents" Name="pdfdocuments">
        <Query>FileExtension='pdf'</Query>
        <PropertyRef Name="Author"/>
        <PropertyRef Name="Description"/>
        <PropertyRef Name="FileName"/>
        <PropertyRef Name="Size"/>
        <PropertyRef Name="Path"/>
        <PropertyRef Name="Created"/>
        <PropertyRef Name="Write"/>
        <PropertyRef Name="CreatedBy"/>
        <PropertyRef Name="ModifiedBy"/>
        <PropertyRef Name="EbookDateOfBirth"/>
</ResultType>
```

表 3.3 SharePoint のためのプロパティデータ型一覧

TET PDF IFilter におけるデータ型	SharePoint のためのデータ型
Int32	integer
Double	decimal
Boolean	boolean
DateTime	datetime
String	text

スタータセット内のすべてのプロパティのための完全な XML ファイル（*starter_advanced_search.xml*）が、TET PDF IFilter とともにインストールされています。これは、SharePoint 評価のための Microsoft の Virtual PC イメージで作成されています。この XML を評価・検証のために使うことができます。業務サイト用の XML は、そのサイトに存在する XML と、新たに構成されたプロパティ群に基づいて、注意深く構築しなければならないことに留意してください。

カスタムメタデータプロパティを検索　高度なメタデータ検索を実装するためには、TET PDF IFilter を、プロパティをインデックスするよう構成する必要があります（28 ページの 2.5「カスタムプロパティ」を参照）。これは TET PDF IFilter に対して、クロール対象プロパティを作成するよう指示します。その後、新規の管理プロパティを、下記のようにして**「高度な検索」**ページで利用可能にする必要があります：

- SharePoint　サイトにログオンし、**「高度な検索」**へ移動。今したいことは、この検索フォーム上で、ページ下端の*「Add property restrictions...」*の下に新しい管理プロパティを追加することです。
- ページの右上隅で、*「Site Actions」*をクリック。開いたドロップダウンメニューで、*「Edit Page」*をクリック。すると、ページの見た目が変わり、ページに変更を加えるためのコントロールが追加されます。
- *「Advanced Search Box」*と題されたボックスの右上隅で、*「edit」*をクリック。開いたドロップダウンメニューで、*「Modify Shared Web Part」*を選択。すると、*「Advanced Search Box」*が破線で囲まれ、その右に、これまた*「Advanced Search Box」*と題された新しいボックスが現れます。これは*「Search box」*・*「Scopes」*・*「Properties」*などの項目を含んでいます。
- *「Properties」*カテゴリを展開し、XML を内容とするフィールド*「Properties」*を探します。このフィールドの中をクリックすると、中に 3 個の点の入ったボタンとツールチップ*「Click to use builder」*が現れます。このボタンをクリックすると、XML の入ったテキスト入力ボックスが開きます。この XML を、下記の例に従って編集する必要があります。この XML を何らかの XML エディタへ複製し、編集した後に貼り付け戻すとよいでしょう。
- このテキスト編集ボックスを閉じた後、編集が行われた場所のボックスの下端の*「Ok」*をクリックし、ページの上端の*「Check In to Share Draft」*をクリックします。すると、**「高度な検索」**フォームの通常の見た目に戻りますので、*「Result type」*ドロップダウンメニューで*「All Results」*を選択すると、ページの下端の、*「Add property restrictions...」*の下の*「(Pick Property)」*メニューの中に新しいプロパティ名が現れます。変更されたフォームをすべてのユーザが利用可能にするためには、*「Check In to Share Draft」*をク

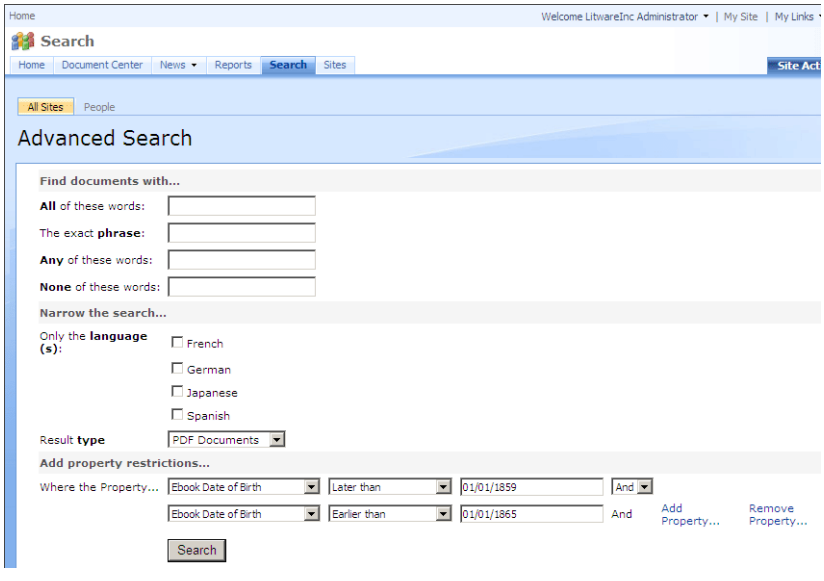


図 3.3.
SharePoint の「高度な検索」ページでカスタムプロパティ

リックするだけでなく、**「高度な検索」**フォームの上端の**「*Publish*」**をクリックする必要がある場合もあります。

これで、管理プロパティ内に特定の項目群を持つ文書群を検索できるようになりました。に、カスタムプロパティ**「*Ebook Date of Birth*」**を下部に持つ「高度な検索」ページを示します。

注記 個々の文書が検索結果のリスト内になぜか見当たらない場合、SharePoint はそれらを重複と見なして除去している可能性があります。重複除去を無効にするには、「Search Results」ページの「View duplicates」リンクをクリックします。

メタデータプロパティを検索　表 3.4 にプロパティクエリの例を挙げます。プロパティクエリは、下記のシンプルなスキームに従って構築されます：

<プロパティ名>:<値>

プロパティベースのクエリのための文法解説は下記にあります：

msdn.microsoft.com/en-us/library/office/ff394509%28v=office.14%29.aspx

表 3.4 SharePoint のためのメタデータクエリ例

検索語例	説明
author:Doyle	作成者が Doyle を含んでいる
author:Arthur author:Doyle	作成者が Arthur と Doyle を含んでいる
author:"Arthur Conan Doyle"	作成者が厳密なテキスト Arthur Conan Doyle を含んでいる

3.3 SQL Server におけるメタデータプロパティ

SQL Server は、メタデータプロパティのインデックス作成や検索に対応していません。ゆえにデフォルトでは、あらゆるプロパティは無視されます。メタデータクエリを実装するには、メタデータプロパティ群をテキストとしてインデックスすることを推奨します（37 ページの 2.7「プロパティをテキストとしてインデックス」を参照）。

SQL Server のための XML 構成　表 3.5 に、SQL Server を使用する際の TET PDF IFilter のための XML 構成に関連する要請・推奨事項を挙げます。サンプル構成ファイル *SQL Server config.xml* は、これらの要件を実装しており、出発点として利用できます。

表 3.5 SQL Server のための XML 構成

エレメント	属性	要請・推奨事項
Filtering	*metadataHandling*	プロパティをテキストとしてインデックスすることを有効にするには、この属性を propertyAndText か propertyAndPrefixedText に設定します（37 ページの 2.7「プロパティをテキストとしてインデックス」を参照）。
Property	*textIndexPrefix*	明示的にプロパティを検索したい場合は接頭辞を設定します。

メタデータプロパティを検索　SQL Server にはプロパティを検索するための専用機能は何もありませんので、フルテキストクエリの中でメタデータプロパティをクエリする必要があります。プロパティをテキストとしてインデックスすることを有効にした後は、作成者が *Arthur Conan Doyle* である文書群を下記のように検索できます：

```
SELECT name FROM DocumentTable WHERE CONTAINS(*,'"TET_System_Author_Arthur Conan Doyle"')
GO
```

作成者が *Arthur* で始まる文書群をクエリするには下記のステートメントを用います：

```
SELECT name FROM DocumentTable WHERE CONTAINS(*,'"System_Author_Arthur*"')
GO
```

4 高度な PDF インデクシング

4.1 さまざまな PDF バージョンと保護文書

使用可能な PDF 入力　TET PDF IFilter は、さまざまなソールからの数百万個の PDF 試験ファイルに対する試験を経ています。PDF 1.0 から PDF 1.7 拡張レベル 8 までを受け付けます。これは、Acrobat！～ DC と PDF 2.0（ISO 32000-2）にあたります。暗号化文書も受け付けます。TET PDF IFilter は、さまざまな種類の、正しく作られていない、または破損した PDF 文書の修復を試みます。

デフォルトでは TET PDF IFilter は、ページ内容・標準文書情報項目・注釈・しおり・フォームフィールドからテキストを抽出し、ファイル添付・PDF ポートフォリオを処理します。この動作は変更可能です。53 ページの 4.2「PDF 文書のさまざまな領域」を参照してください。

注記 TET PDF IFilter は動的 XFA フォームには対応していません。XFA は、PDF 規格 ISO 32000 には含まれていない、別個の形式です。XFA は小さな PDF ラッパの内部にパッケージされていますので、XFA フォームは PDF 文書と混同されることも多いですが、XFA は実際には完全に別個のファイル形式であり、専用のソフトウェアを必要とします。

保護された PDF 文書　TET PDF IFilter は、それが開くことのできる文書であるかぎり、あらゆる文書からのテキストとメタデータをインデックスします。これには、以下の種類の PDF 文書が含まれます。

- 暗号化されていない文書
- マスターパスワードを用いて暗号化されているが、ユーザーパスワードを必要としない文書。Acrobat のセキュリティ設定「**内容のコピー**：**許可／許可しない**」のステータスは、このグループの文書に影響を与えません。

一見、この 2 番目のカテゴリは、文書の作成者の、その文書の保護のための意図に違反しているように見えるかもしれません。しかし、そうではありません。なぜなら TET PDF IFilter は、実際にテキストを複製するための手段を一切提供しておらず、検索エンジンがその文書をインデックスするのを、ひいては検索においてその文書を特定するのを助けているだけだからです。その文書が検索で特定されて、Acrobat で開かれた後は、その文書に対して内容複製に関する制約が指定されているならば、その文書は依然それに従います。

暗号化 PDF 文書を開くことができなかった場合には、それはログ記録レベル 2 ではログ記録されます（81 ページの 6.4「デバッグ機能」参照）。このカテゴリには以下の場合が含まれます：

- ユーザーパスワードを必要とする暗号化文書。すなわち、Acrobat で、それに照応するパスワードを与えなければ開くことができない文書。
- 暗号化された PDF 添付が、それ以外は保護されていない文書の中にある場合。
- ユーザー指定された公開鍵証明書を用いて暗号化されている文書。

破損 PDF 文書　PDF 文書は、破損したデータ構造を含んでいる場合があります。その原因としては、PDF 生成ソフトウェアに不具合があるか、あるいは何らかのアクシデントで変更されてしまった場合です（ネットワーク転送が失敗した等により）。TET PDF IFilter は、破損 PDF 文書を自動的に検出し、そのような文書の修復を試みることによって、テキストとメタデータの抽出を成功させようとします。この修復モードはインデクシング処

理の一部として自動的に動作します。場合によっては、このモードだけでは充分ではなく、TET PDF IFilter がもっと徹底的な修復モードを用いてその文書を処理しなければならないこともあります。これは時間がもっとかかりますので、この強制修復モードは、自動修復モードでは処理が成功しなかった、深刻に破損した PDF に対してのみ行われます。

文書を開くことに成功したけれども、1 個ないし複数の破損ページが含まれている場合には、これらのページは無視され、その後のページ群に対して処理が続行されます。ページが無視されるたびに、アプリケーションイベントログにエントリが書き込まれます。

4.2 PDF 文書のさまざまな領域

PDF 文書は、そのページ内容のみならず、注釈やしおりといった多様な場所の中にもテキストを内容として持つことができます。また、Adobe の XMP 形式か固有の文書情報項目の形をとったメタデータを活用することもできます。PDF 文書内でテキストを内容として持ちうる場所を、PDF 文書の領域と呼びます。以下、PDF 文書のすべての領域を列挙して解説するとともに、それぞれのテキストを Acrobat で表示させる方法もあわせて示します。また、文書のすべての領域に対する TET PDF IFilter のデフォルト動作もこの列挙中に記していきます。簡単にいえば TET PDF IFilter は、関連するすべての場所にあるテキストをインデックスします。その結果として、ひと目見ただけではなぜヒットが生成されたのかわからないような文書についても検索ヒットを得る場合があります。通常、IFilter クライアントには検索語ハイライト機能がありませんので、結果文書の中で検索語がどこにあるか見つけだす方法を知っておくことが重要です。検索されたテキストがページ内容本体以外の場所にあるかもしれないことを念頭においたうえで、TET PDF IFilter が検索ヒットを報告した PDF 文書内で検索テキストがどこにあるか見つけにくいときには、以下の列挙を参照してください。

以下の説明に関する注記：

- Acrobat による「複数の PDF」の検索とは、右記の類の検索のことです：**「編集」→「高度な検索」→「詳細オプションを表示」**（あれば）をクリックし、**「検索する場所：」**プルダウンで PDF 文書群のフォルダを選択（図 4.1 参照）。
- 説明の中で、プロパティセットコレクション ***documentXMP・imageXMP・shell・pdf・internal*** について記している箇所がいくつかあります。これらは XML 構成ファイル内で有効にすることができます（25 ページの 2.3「定義済みプロパティ」を参照）。デフォルトでは、***shell・pdf・internal*** プロパティセットコレクションは有効にされており、***documentXMP・imageXMP*** プロパティセットコレクションは無効にされています。プロパティセットコレクションについて詳しくは 25 ページの 2.3「定義済みプロパティ」を参照してください。
- 記法 ***Filtering@indexNestedPdf*** は、XML 構成ファイル内の属性を意味します（69 ページの 5.2「XML エレメント・属性一覧」を参照）。

以下の項で、PDF 領域検索に関する情報を示します。さらに、これらの文書領域を Acrobat X/XI/DC を用いて検索する方法をまとめます。これは Acrobat 内で検索結果の場所を特定するために重要です。

ページ上のテキスト　ページ内容は、PDF 内においてテキストの主要なソースです。ページ上のテキストはフォントで視覚表現され、PDF 内で利用可能な多様なエンコーディング技法の一つを用いて符号化されています。

- Acrobat での表示方法：ページ内容はつねに表示されています。
- Acrobat X/XI/DC で一個の PDF を検索する方法：**「編集」→「検索」**または**「編集」→「高度な検索」**。Acrobat がグリフを Unicode 値へ正しくマップできない文書内のテキストを、TET PDF IFilter は処理できる可能性があります。このような状況では、TET をベースとした TET Plugin を利用することができます。TET Plugin は、その自前の検索ダイアログを**「*Plug-Ins*」→「*PDFlib TET Plugin...*」→「*TET Find*」**で提供します。ただしこれは、完全な検索機能を提供するようには意図されていません。
- Acrobat X/XI/DC で複数の PDF を検索する方法：**「編集」→「高度な検索」**で、**「検索する場所を指定してください。」**の中で**「以下の場所にあるすべての *PDF* 文書」**を選択し、PDF 文書群が入っているフォルダへブラウズ。

- TET PDF IFilter：ページ内容はデフォルトでインデックスされます。ただし特殊な場合においては、*Filtering@indexPageContents="false"* でページ内容のインデックス作成を無効にすることも可能です。

定義済み文書情報項目　標準文書情報項目はキー／値対です。

- Acrobat X/XI/DC での表示方法：**「ファイル」→「プロパティ ...」**
- Acrobat X/XI/DC で一個の PDF を検索する方法：なし
- Acrobat X/XI/DC で複数の PDF を検索する方法：**「編集」→「[高度な] 検索」**で、ダイアログの下端近くの**「高度な検索オプションを使用」**をクリック。**「検索する場所：」**プルダウンで PDF 文書群のフォルダを選択し、プルダウンメニュー**「その他の条件」**で**「作成日」「更新日」「作成者」「タイトル」「サブタイトル」「キーワード」**のいずれかを選択。
- TET PDF IFilter：定義済み文書情報項目群は、*PropertySetCollection/@shell="true"* であれば（これはデフォルトです）、インデックスされます。

カスタム文書情報項目　標準項目に加えて、カスタム文書情報項目を定義することもできます。

- Acrobat X/XI/DC での表示方法：**「ファイル」→「プロパティ ...」→「カスタム」**（無償の Adobe Reader では利用できません）
- Acrobat X/XI/DC での検索方法：なし
- TET PDF IFilter：カスタム文書情報項目群は、カスタムプロパティ群がこの文書情報項目に基づいて定義されていれば、インデックスされます。

文書レベルの XMP メタデータ　XMP メタデータは、拡張されたメタデータを内容とする XML ストリームから成ります。

- Acrobat X/XI/DC での表示方法：**「ファイル」→「プロパティ ...」→「その他のメタデータ ...」**（無償の Adobe Reader では利用できません）
- Acrobat X/XI で一個の PDF を検索する方法：なし
- Acrobat X/XI で複数の PDF を検索する方法：**「編集」→「検索」**または**「編集」→「高度な検索」**で、**「高度な検索オプションを使用」**をクリック。**「検索する場所：」**プルダウンで PDF 文書群のフォルダを選択し、プルダウンメニュー**「その他の条件」**で**「*XMP* メタデータ」**を選択（無償の Adobe Reader では利用できません）。
- TET PDF IFilter：文書 XMP は、*PropertySetCollection/@documentXMP="true"* であれば、あるいはカスタムプロパティ群が文書 XMP に基いて定義されていれば、インデックスされます。

画像レベルの XMP メタデータ　XMP メタデータは、画像・ページ・フォントといった文書構成要素にも付けることが可能です。しかし、XMP が通常見つかるのは画像レベルにおいてのみです（文書レベルに加えて）。

- Acrobat X での表示方法：**「ツール」→「コンテンツ」→「オブジェクトを編集」**→画像を選択→右クリック→**「メタデータを表示 ...」**（無償の Adobe Reader では利用できません）
- Acrobat XI/DC での表示方法：**「表示」→「表示切り替え」→「ナビゲーションパネル」→「コンテンツ」**。ツリー構造内でその画像を見つけ、それを右クリックして**「メタデータを表示 ...」**を選択（無償の Adobe Reader では利用できません）
- Acrobat X/XI/DC での検索方法：なし

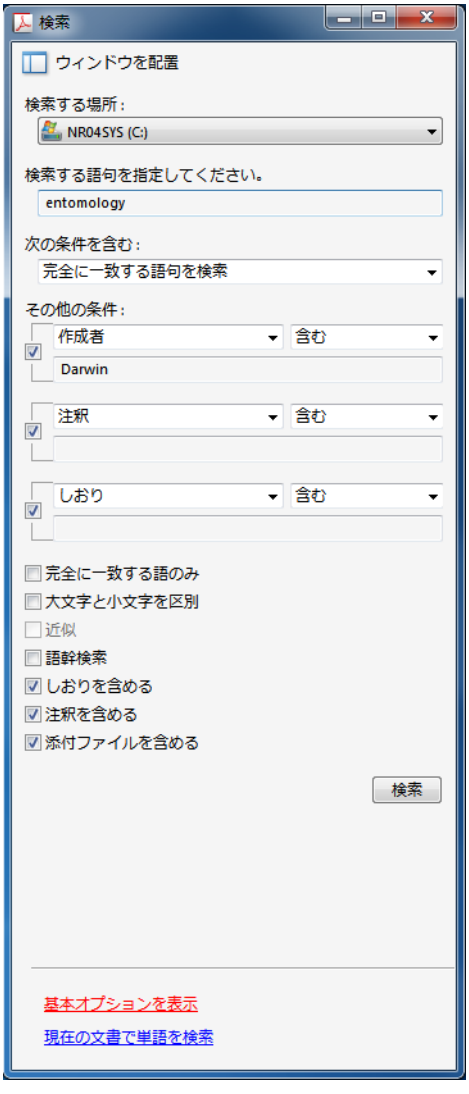


図 4.1
Acrobat の
高度な検索ダイアログ

- TET PDF IFilter：XMP 画像メタデータは、*PropertySetCollection/@imageXMP="true"* であれば、インデックスされます。

フォームフィールド内のテキスト　フォームフィールドはページにかぶさって表示されます。しかし、技術的にはこれはページ内容の一部分ではなく、別途のデータ構造によって表現されます。

- Acrobat X/XI での表示方法：**「ツール」→「フォーム」→「編集」**（無償の Adobe Reader では利用できません）
- Acrobat DC での表示方法：**「ツール」→「フォームを準備」**（無償の Adobe Reader では利用できません）
- Acrobat X/XI/DC での検索方法：Acrobat はフォームフィールドの可視内容を検索します

- TET PDF IFilter：フォームフィールド値は、*Filtering@indexFormFields="true"* であれば（これはデフォルトです）、インデックスされます。
  さらに、*Filtering@indexExtendedFormFields="true"* であれば、フォームフィールドの以下の部分もインデックスされます：デフォルト値・リスト項目・ツールチップ。
  もしメインページ内容（すなわちフォームフィールドキャプション群）が一定であるので必要でない場合は、ページ内容インデックス作成を *Filtering@indexPageContents="false"* で無効にすることも可能です。

注釈内のテキスト　フォームフィールドと同様、注釈（ノート・テキスト注釈等）は別個のデータ構造によって表現されます。注釈において関心対象となるテキスト内容は、その種類によって異なります。たとえば Web リンクの場合、そこで関心対象となる部分は URL でしょうし、それ以外の種類の注釈では、印字されるテキスト内容が意味を持つでしょう。

- Acrobat X/XI での表示方法：**「注釈」→「注釈のリスト」**
- Acrobat DC での表示方法：**「ツール」→「注釈」→「注釈のリスト」**
- Acrobat X/XI/DC で一個の PDF を検索する方法：**「編集」→「検索」**→ボックス**「注釈を含める」**をチェック、または注釈のリストツールバーで**「検索」**ボタンをクリック
- Acrobat X/XI/DC で複数の PDF を検索する方法：**「編集」→「高度な検索」**で、**「詳細オプションを表示」**をクリック。**「検索する場所：」**プルダウンで PDF 文書群のフォルダを選択し、プルダウンメニュー**「その他の条件」**で**「注釈」**を選択。
- TET PDF IFilter：注釈群は、*Filtering@indexAnnotations="true"* であれば（これはデフォルトです）、インデックスされます。

しおり内のテキスト　しおりは、直接にはページと結びついていませんが、特定のページへ飛ぶアクションを内容として持つ場合があります。しおりは、入れ子にして階層構造を形成させることもできます。

- Acrobat X/XI/DC での表示方法：**「表示」→「表示切り替え」→「ナビゲーションパネル」→「しおり」**
- Acrobat X/XI/DC で一個の PDF を検索する方法：**「編集」→「高度な検索」**で、ボックス**「しおりを含める」**をチェック
- Acrobat X/XI/DC で複数の PDF を検索する方法：**「編集」→「[高度な] 検索」**で、**「詳細オプションを表示」**をクリック。**「検索する場所：」**プルダウンで PDF 文書群のフォルダを選択し、プルダウンメニュー**「その他の条件」**で**「しおり」**を選択（無償の Adobe Reader では利用できません）
- TET PDF IFilter：しおり群は、*Filtering@indexBookmarks="true"* であれば（これはデフォルトです）、インデックスされます。

ファイル添付　PDF 文書はファイル添付を含むこともでき（文書レベルかページレベルで）、そのファイル添付自体が PDF 文書であることも可能です。

- Acrobat X/XI/DC での表示方法：**「表示」→「表示切り替え」→「ナビゲーションパネル」→「添付ファイル」**
- Acrobat X/XI/DC での検索方法：**「編集」→「高度な検索」**で、ボックス**「添付ファイルを含める」**をチェック（無償の Adobe Reader では利用できません）。ネストされた添付は再帰的に検索されません。
- TET PDF IFilter：PDF 添付群は、*Filtering@indexNestedPdf="true"* であれば（これはデフォルトです）、再帰的にインデックスされます。

PDF パッケージ・ポートフォリオ　PDF パッケージと PDF ポートフォリオは、ファイル添付に追加のプロパティ群を与えたものです。

- Acrobat X/XI/DC での表示方法：Acrobat は、パッケージ／ポートフォリオの表紙と中身の PDF 文書群を、PDF パッケージ専用のユーザインタフェース要素群を用いて表現します。
- Acrobat X/XI/DC で一個の PDF パッケージを検索する方法：**「編集」→「ポートフォリオ全体を検索」**
- Acrobat X/XI/DC で複数の PDF パッケージを検索する方法：なし
- TET PDF IFilter：パッケージ／ポートフォリオに含まれている PDF 文書は、*Filtering@indexNestedPdf="true"* であれば（これはデフォルトです）、再帰的にインデックスされます。

PDF の各種規格とその他各種 PDF プロパティ　この領域は明示的にテキストを含んではおらず、PDF/X・PDF/A ステータス、タグ付き PDF ステータスといった、PDF 文書のさまざまな固有プロパティを集めたコンテナとして用いられます。

- Acrobat X/XI/DC：**「表示」→「表示切り替え」→「ナビゲーションパネル」→「規格」**（規格準拠 PDF でのみ現れます）
- Acrobat X/XI/DC での検索方法：なし
- TET PDF IFilter：PDF プロパティ群は、*PropertySetCollection/@pdf="true"* であれば（これはデフォルトです）、インデックスされます。PDF/A など規格への準拠は、プロパティ *PDFlib.TET.pdfa* などを用いてクエリできます。

タグ付き PDF　TET は、レイアウトの構造とヒエラルキーを、タグ付き PDF 文書内に存在している構造ツリーを使用することなく、ページ構造から直接再構築します。

ページ内容のうち、その文書を理解するために必要ではなく、レイアウト目的や装飾のために生成されているものについては、タグ付き PDF 内でページ装飾としてマークされる場合があります。ページ装飾の最もよくある用途は、ページ番号や章見出しなどのランニングヘッダ・フッタです。その用法によって、ページ装飾としてマークされているページ内容を処理することが望ましいかどうかが異なります。

- Acrobat XI/DC での表示方法：**「表示」→「表示切り替え」→「ナビゲーションパネル」→「タグ」**。**「タグ」**のメニューで**「検索 ...」**をクリックして、**「ページ装飾」**を選択。ページ装飾としてマークされているテキスト・画像・ベクトルグラフィックがハイライトされます。
  あるいは、「ツール」→「アクセシビリティ」→「TouchUp 読み上げ順序」を開くという方法もあります。このツールは、ページ上のタグ付き内容を灰色の長方形でハイライトします。ハイライトされていない内容がページ装飾を示しています。
- Acrobat X/XI/DC で検索時にページ装飾を無視する方法：なし
- TET PDF IFilter でページ装飾を無視する方法：ページオプション *ignoreartifacts* を与えます。

レイヤー　レイヤー（技術的にはオプショナル内容として知られます）を利用すると、ページ内容の表示と非表示を切り替えることができます。その用法によって、隠しレイヤー上のページ内容を処理することが望ましいかどうかが異なります。

- Acrobat XI/DC での表示方法：**「表示」→「表示切り替え」→「ナビゲーションパネル」→「レイヤー」**。現在表示されているレイヤーは、その名前の頭に目の印があります。この印をクリックすると、レイヤーの表示と非表示を切り替えることができます。

- Acrobat X/XI/DC での検索方法：Acrobat は全レイヤーの内容を検索します。隠しレイヤー上で検索結果が見つかった場合、Acrobat はそのレイヤーを表示させるよう提案します。
- TET PDF IFilter での検索方法：ページオプション ***layers=all/visible/invisible*** を用いて、内容抽出を表示レイヤーか非表示レイヤーに限定することが可能です。あるいは全レイヤーの内容を処理させることもできますが、これはレイヤーどうしが重なりあっていない場合にのみ意味があるでしょう。デフォルトは ***visible*** です。

4.3 自動言語検知

正しい言語検知の重要性　内容インデックス作成のいくつかの側面は、インデックスされた内容の言語独自の後処理によって著しく向上します。細部はそれぞれの IFilter クライアントによって異なるものの、下記の側面は高度に言語依存です：

- 単語区切り機能という構成要素。テキスト内容はこれを用いて単語群に区切られます。
- 単語語幹化機能。インデックスされた単語の原形を抽出します。この方法で検索クエリは、インデックスされた文書の中でその検索語が多少変形（屈折）された形で用いられていたとしても、ヒットを生成します。
- シソーラスベースの検索
- ノイズ語リスト。検索上意味を持たないと見なされる、それゆえインデックスへ入れ込まれない、*the・a・and・at・on* といった語群を内容として持っています。

自動言語検知　上記のさまざまな機能は、テキストがどの自然言語で書かれているかがわかる場合にのみ実装できるものです。デフォルトでは、さまざまな IFilter クライアントは、あらゆる言語独自処理に際してシステムロケールを用います。その結果、日本語文書を英語システム上でインデックスすれば品質二流のインデックス内容が生成されるので、検索にあてはまる文書があっても見つけそこなうおそれがあります。自然言語を正しく割り当てることは、東アジア言語の内容を持つ文書について特に重要です。

インデックス作成処理における言語独自処理のあらゆる側面を向上させるために、TET PDF IFilter は自動的に、文書内容群と型 *string* のプロパティ群の中のテキストについて自然言語を特定します。TET PDF IFilter は、その言語を検出するために 2 種類の技法を運用しています：

- いくつかの場合においては、用字系（表記体系）を調べるだけで充分に言語を特定できます。たとえば日本語のひらがなは、日本語でテキストを書くためにしか使われません。
- 多くの言語は、他の言語と同じ表記体系を共有しています。たとえば西洋語の大半は、ラテン用字系で書かれます。このような場合には TET PDF IFilter は、統計分析を適用して正しい言語を特定します。

自動言語検知は各テキスト塊を個別に分析しますので、1 個の文書に複数言語がどのようにでも混在してありえます。各切片に対応する言語が下記のように割り当てられます：

- 自動用字系検知が、テキストを複数の、同一用字系から成る塊に分割します。
- その用字系が言語を一意に特定しない場合には、自動言語検知を用いてテキストが分析されます。この処理は、塊全体についてもっともありうる言語を割り当てますが、その塊を各言語ごとにもっと小さな塊へさらに分割することはしません。たとえば、英語・ドイツ語・フランス語の混在したラテン字テキストを持つ塊は、その塊全体に対してこれらの言語のうちの 1 つを割り当てられます。どれを割り当てられるかは言語の分布状況によって決まります。

手動言語割り当て　TET PDF IFilter における自動言語検知は、特殊な応用のためにカスタマイズすることも可能です。このカスタマイズ機能は LCID（ロケール識別子）を利用しています。LCID は自然言語を指定し、さらに、国別または地域別の言語変種どうし（英国英語と米国英語など）を区別することも可能です。表 4.1 に、よく使われる LCID 値をいくつか挙げます。LCID の全一覧は下記にあります：

msdn.microsoft.com/en-us/library/ms776294(VS.85).aspx

注記 TET PDF IFilter は、言語検知より後の言語独自処理は一切適用しません。その LCID 情報を使うかどうかは IFilter クライアントに委ねられます。いくつかの IFilter クライアント（SharePoint・SQL Server など）は、LCID を取り扱う洗練された機能を持っていますが、それ以外の IFilter クライアントのなかには、LCID 情報を全く無視するものもあります。

表 4.1 よくある LCID 値と、対応する第一・第二言語

LCID	第一言語	第二言語（国）
0x0000	中立ロケール言語	中立副言語
0x0401	アラビア語（ar）	サウジアラビア（SA）
0x0404	中国語（zh）	繁体字（Hant）
0x0407	ドイツ語（de）	ドイツ（DE）
0x0409	英語（en）	米国（US）
0x040c	フランス語（fr）	フランス（FR）
0x0410	イタリア語（it）	イタリア（IT）
0x0411	日本語（ja）	日本（JP）
0x0413	オランダ語（nl）	オランダ（NL）
0x0419	ロシア語（ru）	ロシア（RU）
0x0804	中国語（zh）	簡体字（Hans）
0x0c0a	スペイン語（es）	スペイン（ES）
0x0800	システムデフォルトロケール言語	
0x1000	不特定カスタムロケール言語	不特定カスタム副言語

LCID のための XML 構成　自動 LCID 検知を上書きまたは補足するための LCID は、XML 構成ファイルの *LocaleId* エレメント内で指定することができます：

```
<LocaleId default="1031" detection="auto" latin="1031" cyrillic="1026" chinese="4100"/>
```

detection 属性は値 *auto*・*disabled*・*script* をとりえます。他の属性は *default* 以外すべて、*detection=disabled* の場合には無視されます。デフォルトは *auto* です。*script* 設定は、用字系分析を有効にしますが統計分析を無効にします。

default 属性を使うと、*detection=disabled* の場合にすべてのテキストについて用いられるグローバルな LCID 設定を指定することが可能です。この属性がないときにはシステムロケールが用いられます。

detection 以外の属性ではすべて、10 進か 16 進文法の数値を指定できます。16 進値は先頭が *ox* でなければなりません。表 4.2 に、対応している用字系属性と、そのデフォルト値を挙げます。これ以外のあらゆる用字系のテキストに対する LCID は自動的に割り当てられます。

表 4.2 Attributes for specifying script-specific locale IDs (LCIDs)

属性	デフォルト値
default	0x0800 カレントシステムロケール（detection=disabled の場合にすべてのテキスト塊に対して用いられます）
latin	0x0409 英語（US）
cyrillic	0x0419 ロシア語（RU）
arabic	0x0401 アラビア語（SA）
chinese	0x0804 中国語（中華人民共和国）

4.4 Unicode 後処理

TET PDF IFilter のための PDF テキスト抽出エンジンを実装している基盤である TET カーネルは、Unicode 後処理のためのさまざまな制御手段を提供しています。これらについては、TET PDF IFilter パッケージに含まれている TET マニュアルで詳しく解説しています。以下、もっとも重要な機能にしぼって見ていきます。

さまざまな Unicode 後処理機能は、TET 文書オプション群によって制御されます。TET PDF IFilter では、これらのオプションは XML 構成ファイルの ***DocOptions*** エレメント内で提供することが可能です。例：

```
<Tet>
        <DocOptions>decompose={canonical=_all}</DocOptions>
        <PageOptions/>
        <TetOptions/>
</Tet>
```

Unicode 字形統合　字形統合は、1 個ないし複数の Unicode キャラクタを処理して、各キャラクタに対して特定の操作を適用します。以下の操作が利用可能です：

- キャラクタを温存。
- キャラクタを除去。
- 別の（固定の）キャラクタへ置き換え。

字形統合は連鎖しません：いずれかの字形統合の出力が、別に存在する字形統合によってさらに処理されることはありません。字形統合は、Unicode テキスト出力のみに影響を与え、***TET_char_info*** 構造の中で、あるいは TETML 内の ***<Glyph>*** エレメントの中でレポートされるグリフの集合には影響しません。たとえば、字形統合が特定の Unicode キャラクタ群を除去したとしても、その元のキャラクタ群を生成した照応するグリフ群はなおレポートされます。

下記のオプションリストは複数の字形統合を指定しています：

```
<Tet>
        <DocOptions>fold={ {[:blank:] U+0020 } {_dehyphenation remove} }</DocOptions>
</Tet>
```

字形統合のさまざまな例が TET マニュアルにあります。TET マニュアルは、TET PDF IFilter によって適用されるデフォルト字形統合も説明しています。

Unicode 分解　分解は、1 個のキャラクタを、等価な他の 1 個ないし複数のキャラクタへ置き換えます。[1] ある Unicode キャラクタが他のキャラクタまたはキャラクタ列に対して（互換または正準）等価であると呼ばれるのは、それらが実際には同じ意味を持ちながら、歴史的理由（たいていはレガシエンコーディングとのラウンドトリッピングがらみ）によって Unicode 内で別々に符号化されている場合です。分解は情報を破壊します。これが有用となるのは、元のキャラクタとその等価キャラクタとの違いを気にしない場合です。しかしその違いを気にする場合であれば、当該の分解を適用するべきではありません。

注記 多くの分解は実際には 1 個のキャラクタを複数部分へ分割せず、1 個のキャラクタを別の 1 個のキャラクタへ変換するのですが、用語「分解」はここでは Unicode 標準における定義に従って用います。

1. Unicode 分解に関する完全な説明は、www.unicode.org/versions/Unicode8.0.0/ch02.pdf（2.12 節）と www.unicode.org/versions/Unicode8.0.0/ch03.pdf（3.7 節）を参照してください。

正準分解　互いに正準等価であるキャラクタまたはキャラクタ列は、同一の抽象キャラクタを表現しており、したがって必ず同一の体裁と動作を持つべきものです。よくある例としては、合成済みキャラクタ（例：Ä U+00C4）と連結キャラクタ列（例：A ¨ U+0041 U+0308）が挙げられます：2 つの表現は互いに正準等価です。1 つの表現から別の表現へ替えても情報は失われません。正準分解は、1 つの表現を、正準表現と見なされる別の表現へ置き換えます。

Unicode コードチャート（*www.unicode.org/charts/* 参照）では、正準マッピングには記号 IDENTICAL TO ≡ U+2261 が付けられています（キャラクタテーブルにはなし）。分解名 **<*canonical*>** と暗黙的に見なされます。

下記の文書オプションは、すべての正準等価キャラクタをおのおの対応キャラクタへマップします：

```
<Tet>
        <DocOptions>decompose={canonical=_all}</DocOptions>
</Tet>
```

表 4.3 正準分解：decompose オプションのサブオプション（正準等価なキャラクタには、Unicode コードチャートで記号 IDENTICAL TO ≡ U+2261 が付けられています）

分解名	説明	分解前	分解後
canonical[1]	正準分解	À U+00C0	A ` U+0041 U+0300
		林 U+F9F4	林 U+6797
		Ω U+2126	Ω U+03A9
		ば U+3070	は ゙ U+306F U+3099
		אִ U+FB2F	א ◌ִ U+05D0 U+05B8

1. デフォルトではこの分解は、特定のキャラクタ群を温存するために、すべてのキャラクタに対しては適用されません。詳しくは TET マニュアルを参照してください。

互換分解　互換等価なキャラクタどうしは、同一の抽象キャラクタを表しますが、その体裁や動作は互いに異なっている可能性があります。例としては、アラビア文字キャラクタの単独形（س U+0633 等）と位置依存表示形（سـ ـسـ ـس U+FEB2 U+FEB4 U+FEB3 等）が挙げられます。互換等価キャラクタは組版上互いに異なるものです。この組版情報を除去すると、情報が失われる可能性がありますが、ある種の応用の場合（検索等）には処理を単純化できるでしょう。互換分解は組版情報を除去します。

Unicode コードチャートでは、互換マッピングには記号 ALMOST EQUAL TO ≈ U+2248 が付いており、その後に、分解名（「タグ」ともいいます）が **<*noBreak*>** のように山括弧にくくって書かれています。タグ名が書かれていないものについては、**<*compat*>** と見なされます。このタグ名は、***decompose*** オプションリストの中のオプション名として用いられています。いくつかの例に見られるように、分解の結果、1 個のキャラクタが複数キャラクタ列へ変換されることもあります。

注記 PDF 文書がすでにグリフを、分解されていない Unicode 値ではなく分解済みのシーケンスへマップしている場合があります。この場合には decompose オプションはその出力に影響を与えません。

分解のさまざまな例が TET マニュアルにあります。オプションリスト内の分解名（*font*・*circle*・*initial*・*vertical* など）とデフォルト分解の一覧も TET マニュアルに挙げています。

Unicode 正規化　Unicode 規格では、正準等価と互換等価の概念に基づき、4 種類の正規形を定義しています。[1] すべての正規形は、結合記号を特定の順序に置き、分解と合成を異なる方式で適用します：

- 正規形 C（NFC）は、正準分解の後に正準合成を適用します。たとえば、合成形 C は *Ä* を 1 個のキャラクタ Ä (U+00C4) として格納します。NFC は、Windows・Web 上・多くのデータベースにおいて Unicode テキストに対して望ましい形式です。
- 正規形 D（NFD）は、正準分解を適用します。たとえば、分解形 D は *Ä* を A ¨ (U+0041 U+0308) として、すなわちベースのキャラクタと、結合するダイアクリティカルキャラクタとのシーケンスとして格納します。
- 正規形 KC（NFKC）は、互換分解の後に正準合成を適用します。言い換えれば、いくつかのキャラクタが、互換な基本形へマップされます。たとえば合字 ﬁ (U+FB01) はシーケンス f i (U+0066 U+0069) へマップされます。
- 正規形 KD（NFKD）は、互換分解を適用します。これは正規形 KC と似ていますが、正準合成を適用しません。

正規形の選択は、用途の要件に依存します。

正規形は、Unicode 規格付録 #15「Unicode Normalization Forms」で定義されています（*www.unicode.org/versions/Unicode5.2.0/ch03.pdf#G21796*・*www.unicode.org/reports/tr15/* を参照）。

TET PDF IFilter は 4 種類の Unicode 正規形すべてに対応しています。Unicode 正規化は *normalize* 文書オプションで制御できます。例：

```
<Tet>
        <DocOptions>normalize=nfc</DocOptions>
</Tet>
```

TET PDF IFilter はデフォルトでは正規化を適用しません。*decompose* オプションと *normalize* オプションはかち合うおそれがあることから、*normalize* オプションを *none* 以外の値に設定するとデフォルト分解は無効化されます。

1. これらの正規形は、Unicode Standard Annex #15「Unicode Normalization Forms」で使用化されています（www.unicode.org/versions/Unicode8.0.0/ch03.pdf#G21796 と www.unicode.org/reports/tr15/ を参照）

4.5 カスタムグリフマッピングテーブル

TET PDF IFilter にはさまざまな回避策が実装されてはいるものの、ある種のいくつかのまれな場合においては、PDF 文書が Unicode マッピングのための決定的な情報を持っていなければ、その文書からテキストを正しく抽出することが不可能なこともあります。

TET PDF IFilter は、PDFlib TET 製品のマニュアルで解説されているさまざまなテーブル形式に対応しています。無償の PDFlib FontReporter や、Adobe Acrobat 用 PDFlib TET Plugin を使って、そのようなマッピングテーブルを試用することも可能です。Unicode マッピングテーブルは、構成ファイル内で正しい文書オプション群を用いて構成する必要があり、TET PDF IFilter インストレーションディレクトリの ***resource*** ディレクトリ内に置くことができます。

TET オプション群のための XML 構成　TETコアの操作を制御するためのオプションリスト群（）は、TET 製品のマニュアルに解説されているオプションリスト文法に従って構築する必要があり、TET PDF IFilter に対して、XML 構成ファイルの ***Tet*** エレメントの子エレメントである ***DocOptions*・*PageOptions*・*TetOptions*** エレメントの中で与えることができます：

```
<Tet>
        <DocOptions>glyphmapping {{fontname=T* glyphlist={tex}}}</DocOptions>
        <PageOptions/>
        <TetOptions>searchpath={C:/glyphlists}</TetOptions>
</Tet>
```

5 XML 構成ファイル

5.1 構成ファイルの作業

TET PDF IFilter の動作は XML 構成ファイルで制御できます。サンプル構成ファイル群が TET PDF IFilter とともにインストールされています。いくつかの定義済み構成ファイルが TET PDF IFilter とともにインストールされています：

- ファイル ***default_config.xml*** : TET PDF IFilter の内部的なデフォルト設定群を記述しています。カスタマイズした構成ファイルのための出発点として利用できるでしょう。
- ファイル ***starter_samples.xml*** : 32 ページの 2.6「カスタムプロパティを用いた例」で説明したプロパティサンプル群とともに使用できるプロパティ定義群を内容として持っています。
- さらに、Windows Search・SharePoint・SQL Server のいずれかとともに使用するための XML 構成ファイル群もあります。インストール過程の中で、ユーザーの要請により、これらの構成ファイルのうちの 1 つを登録できます。

XML 構成ファイルの場所を指定　構成ファイルは、その構成ファイルのフルパス名を持つ文字列値を内容とする下記のレジストリキー内で指定する必要があります：

```
HKEY_LOCAL_MACHINE\SOFTWARE\PDFlib\TET PDF IFilter5\configfile
```

注記 カスタマイズした XML 構成ファイルは、TET PDF IFilter のインストレーションディレクトリの外に置くことを推奨します。こうすれば、TET PDF IFilter. のアップデート版をインストールした後などにインストレーションディレクトリが変わっても構成は生き残ります。

このレジストリ項目が存在しない場合や、空文字列を内容としている場合には、デフォルト構成が用いられます。レジストリ項目で指定されている構成ファイルを開くことができない場合、または構成ファイルの XML 解析が失敗した場合には、イベントログにエラーメッセージが書き込まれ、インデックス作成は一切実行されません。

1 台のマシン上の TET PDF IFilter に対しては、ただ 1 つの構成ファイルだけが使えます。ただし、32 ビット版と 64 ビット版は別々の構成ファイルを同一マシン上に持つことも可能です。なぜなら、レジストリ項目はそれぞれ 32 ビットレジストリか 64 ビットレジストリの中で検索されるからです。

構成ファイルを変更したときは、その変更を有効にするには、インデックスを再構築する必要があります。

XML 名前空間とスキーマ記述　表 5.1 に、XML 構成ファイルで利用可能なエレメントと属性を挙げます。XML 構成言語のための XSD スキーマ記述は、TET PDF IFilter とともにインストールされており、また、スキーマファイル内で与えられている URI にもあります。このスキーマファイルを適切な XML エディタとともに使えば、生成した XML 構成ファイルが TET PDF IFilter が期待する文法を必ず遵守しているようにすることができます。

XML エレメント・属性のためのカスタムデータ型　値の説明が提供されている所を除いて、すべてのエレメントは空です。下記のカスタムデータ型が XML 構成ファイル内で用いられます：

- LCID：16 進か 10 進のロケール識別子。下記参照：

  msdn.microsoft.com/en-us/library/ms776294(VS.85).aspx

  値 0x0800 は、カレントシステムデフォルトロケールへ翻訳されます。
- GUID（グローバル一意識別子）：UUID（汎用一意識別子）ともいいます。RFC 4122 に従った、一意な 128 ビットの識別子の、大文字小文字を区別しない 16 進表記。その各部はダーシキャラクタ「-」で区切られる必要があります。GUID を作成するツールはさまざまなものが利用可能です。たとえば下記のオンラインサービスなどです：

  www.uuidgenerator.net/

  GUID のサンプルは次のようになります：*7a737220-0cd0-11dd-bd75-0002a5d5c51b*。
- pCOSパス：PDFオブジェクトを記述する拡張pCOSパス。pCOSリファレンスと、22ページの「拡張 pCOS パス」の項で説明している pCOS 拡張を参照してください。
- オプションリスト：***PDFlib TET* リファレンスマニュアル**で仕様定義されている文法に従ったオプションリストを内容とする文字列。
- 言語識別子：RFC 1766 に従った XMP 言語修飾子、またはその文書内のデフォルト言語を識別する ***x-default***。

5.2 XML エレメント・属性一覧

表 5.1 に、XML 構成ファイルのエレメントと属性の詳細を示します。XML 構成ファイルによって制御される効果に関するもっと詳しい情報は、このマニュアルのそれぞれの箇所にあります。個別の IFilter クライアントの場合の要請・推奨設定は、21 ページの 2 章「メタデータプロパティをインデックス」内のクライアントごとの節に挙げてあります。

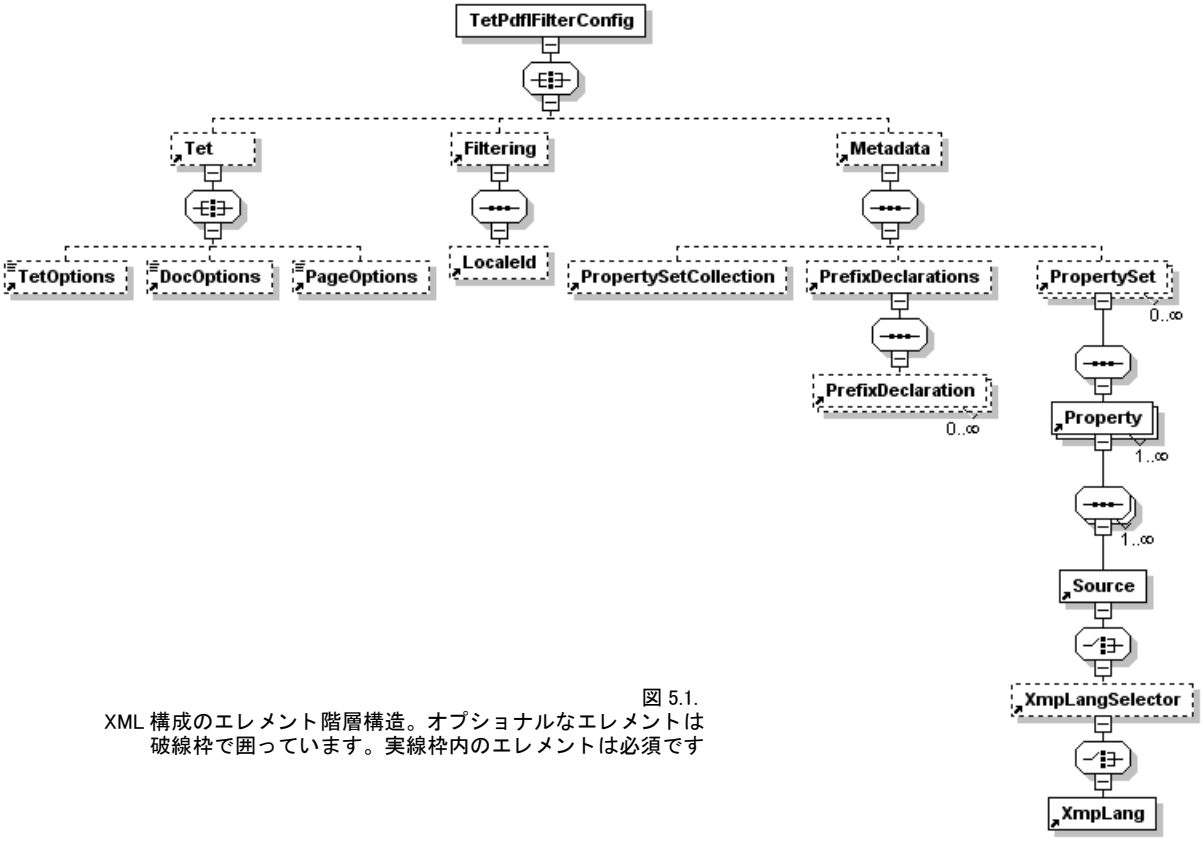


図 5.1.
XML 構成のエレメント階層構造。オプショナルなエレメントは破線枠で囲っています。実線枠内のエレメントは必須です

表 5.1 構成ファイル内の XML エレメント・属性一覧

エレメント	そのエレメントとその属性群の説明
DocOptions 親：Tet	（0 回または 1 回出現可能）その値は、TET カーネル内の TET_open_document( ) に対するオプションリストを内容とします。

表 5.1 構成ファイル内の XML エレメント・属性一覧

エレメント	そのエレメントとその属性群の説明
Filtering 親： TetPdfIFilterConfig	（0 回または 1 回出現可能）PDF フィルタリング処理の詳細を指定します。可能な属性： ***errorIndicator*** （文字列。オプショナル）TET 関数呼び出しが失敗した場合に IFilter クライアントへ与えられる文字列。問題に関する詳細を Windows イベントログ内で見いだせる可能性があります（81 ページの「アプリケーションイベントログ」を参照）。このエラー標識は、インデックス作成の問題を識別するために有用でしょう。これは、文書から取得された何らかの（部分的な）テキストに加えて出力されます。このエラー標識が実際のインデックス項目に混じってしまわないようにするために、一意な文字列を句読点なしで与えることを推奨します。例：TETPDFIFILTERERROR。デフォルト：エラー標識なし ***indexAnnotations*** （論理型。オプショナル。TET PDF IFilter 5.1）注釈群がインデックスされるかどうかを示します。デフォルト：true ***indexBookmarks*** （論理型。オプショナル。TET PDF IFilter 5.1）しおり群がインデックスされるかどうかを示します。デフォルト：true ***indexExtendedFormFields*** （論理型。オプショナル。TET PDF IFilter 5.1）フォームフィールドの以下の部分がインデックスされるかどうかを示します：フィールド名、デフォルト値、ツールチップ、メニューフィールドの項目リスト。デフォルト：false ***indexFormFields*** （論理型。オプショナル。TET PDF IFilter 5.1）フォームフィールド値群（すなわちフォームフィールド群の内容）がインデックスされるかどうかを示します。デフォルト：true ***indexNestedPdf*** （論理型。オプショナル）PDF ポートフォリオと添付が再帰的に処理されるかどうかを示します（53 ページの 4.2「PDF 文書のさまざまな領域」を参照）。デフォルト：true ***indexPageContents*** （論理型。オプショナル）PDF ページ群の内容がインデックスされるかどうかを示します。ページ内容インデックス作成を無効にすることは、検索がメタデータプロパティ群によってのみ推進されるシナリオにおいて有用でしょう。デフォルト：true ***metadataHandling*** （選択肢。オプショナル）メタデータ処理の種類を選択（37 ページの 2.7「プロパティをテキストとしてインデックス」を参照）。デフォルト：property ***ignore*** すべてのメタデータプロパティを落とします。これは、デバッグや、メタデータが必要でない状況においてパフォーマンス最適化のために有用でしょう。 ***property*** メタデータをプロパティとして処理。 ***propertyAndPrefixedText*** メタデータをプロパティとして処理することに加え、カスタムプロパティに対しては textIndexPrefix（あれば）で指定されている接頭辞を、定義済みプロパティに対しては 37 ページの表 2.2 に従った接頭辞を頭に付加し、結果をプレーンテキストとして処理。 ***propertyAndText*** メタデータをプロパティとして処理することに加え、メタデータをプレーンテキストとして処理。 ***useIdentifier*** （論理型。オプショナル）Property エレメントに identifier 属性と friendlyName 属性が両方ある場合に、プロパティを識別するために identifier 属性が用いられるかどうかを指定。デフォルト：true

表 5.1 構成ファイル内の XML エレメント・属性一覧

エレメント	そのエレメントとその属性群の説明
LocaleId 親：Filtering	（0 回または 1 回出現可能）ロケール ID 検知を構成（59 ページの 4.3「自動言語検知」を参照）。可能な属性： ***arabic*** （LCID。オプショナル）アラビア文字テキストに対する LCID。デフォルト：0x0401 アラビア語（SA） ***chinese*** （LCID。オプショナル）中国文字テキストに対する LCID。デフォルト：0x0804 中国語（中華人民共和国） ***cyrillic*** （LCID。オプショナル）キリル文字テキストに対する LCID。デフォルト：0x0419 ロシア語（RU） ***default*** （LCID。オプショナル）detection が無効の場合にすべてのテキスト塊に対して用いられるグローバル LCID。デフォルト：0x0800（システムロケール） ***detection*** （選択肢。オプショナル）自動 LCID 検知を制御。デフォルト：auto ***auto*** 用字系と統計的言語分析に基づいて LCID を決定。 ***disabled*** LCID 検知を無効にする。他のすべての属性は、default と useCatalogLang 以外、無視されます。 ***script*** （TET PDF IFilter 4.0）用字系に基づいて LCID を決定。 ***latin*** （LCID。オプショナル）ラテン字テキストに対する LCID。デフォルト：0x0409 英語（US） ***useCatalogLang*** （論理型。オプショナル。TET PDF IFilter 4.0）文書のカタログの中の Lang 項目が評価されるかどうかを指定。true の場合、TET PDF IFilter は PDF 文書カタログ内の Lang 項目をチェックします。存在する場合には、その Lang 項目は LCID へ変換されます。その変換がうまくいった場合には、この LCID は LocaleId/@default 属性の値を上書きします。この LCID がアラビア・中国・キリル・ラテン用字系のうちのいずれか 1 つに属する場合には、それは LocaleId エレメントの対応する属性の値を上書きします。デフォルト：true
Metadata 親：TetPdfIFilterConfig	（0 回または 1 回出現可能）メアデータプロパティ群を指定（28 ページの 2.5「カスタムプロパティ」を参照）。存在する場合には、このエレメントは Filtering と Tet の後に出現する必要があります。
PageOptions 親：Tet	（0 回または 1 回出現可能）TET 関数 TET_open_page( ) に対するオプションリスト。
PrefixDeclaration 親：PrefixDeclarations	（0 回または 1 回出現可能）Source/@xmpName 内で使える名前空間接頭辞を宣言。可能な属性： ***prefix*** （コロン「:」キャラクタを含まない文字列。必須）名前空間 URI の短縮形として用いられる接頭辞。 ***uri*** （URI。必須）名前空間 URI
PrefixDeclarations 親：Metadata	（0 回または 1 回出現可能）Source エレメントの xmpName 属性群の中の XMP プロパティ群に対する名前空間接頭辞群を宣言

表 5.1 構成ファイル内の XML エレメント・属性一覧

エレメント	そのエレメントとその属性群の説明
Property 親：PropertySet	（0 回または 1 回出現可能）インデックス作成のためのメタデータプロパティを指定（28 ページの 2.5「カスタムプロパティ」を参照）。 identifier と friendlyName のうち少なくとも 1 つが存在する必要があります。両方が与えられている場合には、Filtering/@useIdentifier=false でないかぎりは、identifier が IFilter インタフェース内で用いられます。 可能な属性： ***identifier*** （整数≧ 2。オプショナル）PropertySet 内でプロパティを一意に識別する数値。 ***emitAsVector*** （論理型。オプショナル）true の場合、プロパティ値は、値の数にかかわらず、1 個のベクトル実体として出力されます。 false の場合、プロパティは単値として出力されます。複数のソース項目が見つかったときは、複数の単値が出力されます。デフォルト：false ***friendlyName*** （文字列。オプショナル）PropertySet 内のプロパティを一意に識別する名前。これはプロパティを文書化するために、あるいは identifier のかわりとして使うことができます。TET PDF IFilter ではプロパティ名の長さに制限はありませんが、Windows では最長 64 キャラクタとなっています。 ***precedence*** （選択肢。オプショナル）複数の Source エレメントに対する優先権を指定（デフォルト：first-wins）： ***first-wins*** 最初の空でないソースが用いられます。 ***try-all*** すべての空でないソースがそのプロパティに寄与します。 ***suppressValue*** （文字列。オプショナル。TET PDF IFilter 5.1）指定されたソース（precedence="first-wins" の場合には 1 個のソース、あるいは precedence="try-all" の場合にはすべての空でないソース）が、この属性で与えられた値をもたらす場合に、その値を存在しないと見なす。type="DateTime" のプロパティに対しては、この値は PDF 文書で与える必要があります。例：suppressValue="D:20170116130311+01'00'" ***textIndexPrefix*** （文字列。オプショナル）Filtering/@metadataHandling が propertyAndPrefixedText である場合にプロパティ値の頭に付加される文字列。デフォルト：空 ***type*** （選択肢。オプショナル）そのメタデータプロパティの Windows データ型。可能な選択肢は Boolean・DateTime・Double・Int32・String です。デフォルト：String
PropertySet 親：Metadata	（0 回または任意回数出現可能）同一 GUID を持つプロパティ群のカスタムのセットのフィルタリングを指定（28 ページの 2.5「カスタムプロパティ」を参照）。 存在する場合には、このエレメントは PropertySetCollection と PrefixDeclarations の後に出現する必要があります。 可能な属性： ***guid*** （GUID。必須）そのプロパティセットに対する一意な 128 ビット識別子を 16 進表記で（67 ページの「XML エレメント・属性のためのカスタムデータ型」参照）。

表 5.1 構成ファイル内の XML エレメント・属性一覧

エレメント	そのエレメントとその属性群の説明
PropertySet-Collection 親：Metadata	（0 回または 1 回出現可能）定義済みプロパティセットコレクションのフィルタリングを指定（25 ページの 2.3「定義済みプロパティ」を参照）。プロパティのリストは付章 A「定義済みメタデータプロパティ」にあります。可能な属性： ***documentXmp***（論理型。必須）文書 XMP プロパティ群を出力。デフォルト：false ***imageXmp***（論理型。必須）画像 XMP プロパティ群を出力。デフォルト：false ***internal***（論理型。必須）TET PDF IFilter の内部プロパティ群を出力。デフォルト：true ***pdf***（論理型。必須）PDF 独自プロパティ群を出力。デフォルト：true ***shell***（論理型。必須）シェルプロパティ群を出力。デフォルト：true
Source 親：Property	（0 回または 1 回出現可能）メタデータプロパティに対する 1 個ないし複数のソースを指定。 Property/@precedence が値 first-wins を持っている場合には、エレメント群の順番は意味を持ちます。属性 pdfObject・xmpName のうちの少なくとも 1 つが与えられる必要があります。 可能な属性： ***pdfObject***（pCOS パス。オプショナル）そのプロパティを内容とする、論理値・数値・名前・文字列のいずれかの型の 1 個ないし複数の PDF オブジェクトへの拡張 pCOS パス。デフォルト：/Root/Metadata（すなわち文書レベル XMP） ***suppressValue***（文字列。オプショナル。TET PDF IFilter 5.1）指定されたソースが、この属性で与えられた値をもたらす場合に、その値を存在しないと見なす。 ***xmpName***（スキーマの接頭辞・コロン「:」・プロパティ名から成る文字列。オプショナル）完全修飾された XMP プロパティ名。接頭辞は、それが PrefixDeclaration エレメント内で宣言されていれば、名前空間 URI のかわりに使えます。この属性は、pdfobject が存在しない場合、または 1 個ないし複数の XMP ストリームを指し示している場合にのみ用いられます。デフォルト：空
Tet 親：TetPdfIFilterConfig	（0 回または 1 回出現可能）TET カーネルに対する処理オプション群を指定。オプションリスト文法と可能なオプションの説明は TET マニュアルを参照してください。いくつかのオプションは TET PDF IFilter によって上書きされます。
TetOptions 親：Tet	（0 回または 1 回出現可能）TET 関数 TET_set_option( ) に対するオプションリスト。
TetPdfIFilterConfig 親：（なし）	（ルートエレメントとしてちょうど 1 回出現する必要があります）XML 構成ファイルのルートエレメント。可能な属性： ***version***（文字列。オプショナル。TET PDF IFilter 4.0）この構成が想定して書かれている TET PDF IFilter のバージョンを指定。新しい構成は、適切なバージョン番号（TET PDF IFilter 5.1 なら 5.1）を持つこの属性を含む必要があります。
XmpLang 親：XmpLangSelector	（XmpLangSelector/@languages=subset の場合にちょうど 1 回出現する必要があります）XMP プロパティの言語を指定。可能な属性： ***lang***（言語識別子。必須）言語の名前。現在、x-default が唯一可能な値です。
XmpLangSelector 親：Xmp	（0 回または 1 回出現可能）インデックス作成のための XMP プロパティの言語変種を選択。これは型 Lang Alt の XMP ソースを持つプロパティに対してのみ意味を持ちます。可能な属性： ***languages***（選択肢）そのプロパティの言語独自インデックス作成を指定（デフォルト：all）： ***all*** そのプロパティのすべての利用可能な言語項億がインデックスされます。 ***subset*** XmpLang エレメントで指定されている言語群だけがインデックスされます。

5.3 サンプル構成ファイル

以下に、TET PDF IFilter のための完全な XML 構成ファイルを示します：

```
<?xml version="1.0" encoding="UTF-8"?>
<n:TetPdfIFilterConfig
    xmlns:n="http://www.pdflib.com/XML/TET_PDF_IFilter3/TET_PDF_IFilter_Config-3.0.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.pdflib.com/XML/TET_PDF_IFilter3/TET_PDF_IFilter_
Config-3.0.xsd http://www.pdflib.com/XML/TET_PDF_IFilter3/TET_PDF_IFilter_Config-3.0.xsd"
        version="5.1">

<n:Tet>
        <n:TetOptions></n:TetOptions>
        <n:DocOptions></n:DocOptions>
        <n:PageOptions></n:PageOptions>
</n:Tet>

<n:Filtering
        indexAnnotations="true"
        indexBookmarks="true"
        indexExtendedFormFields="false"
        indexFormFields="true"
        indexNestedPdf="true"
        metadataHandling="property"
        useIdentifier="true">

        <n:LocaleId
            detection="auto"
            useCatalogLang="true"
            default="0x0800"
            arabic="0x0401"
            chinese="0x0804"
            cyrillic="0x0419"
            latin="0x0409"/>
</n:Filtering>

<n:Metadata>
        <n:PropertySetCollection
            documentXmp="false"
            imageXmp="false"
            internal="true"
            pdf="true"
            shell="true"/>

        <n:PropertySet guid="E544AFE6-13E2-40F1-A702-DCEBE8FB7B03">
            <n:Property identifier="20">
                <n:Source pdfObject="/Info/Producer"/>
            </n:Property>
        </n:PropertySet>
</n:Metadata>

</n:TetPdfIFilterConfig>
```

6 トラブルシューティング

6.1 TET PDF IFilter が全く動かない

TET PDF IFilter が全く動作していないように見えるときは、下記の項目をチェックしてください。

TET PDF IFilter が正しく登録されているか？ コマンドラインツール *FiltReg.exe* を使って、TET PDF IFilter の正しい登録をチェックすることができます。このプログラムは、さまざまな IFilter に関連づいているすべてのファイル拡張子をリストします。具体的には、関連づけられた IFilter DLL のファイル拡張子と名前を印字します。このプログラムは、Microsoft Visual Studio とともにインストールされるほか、Windows 7 用 Windows SDK にも含まれています。詳しくは下記を参照してください：

msdn.microsoft.com/en-us/library/ms692537(VS.85).aspx

注記 64 ビットエディションの TET PDF IFilter をテストするには、64 ビット版の `FiltReg.exe` が必要です。64 ビット版を得るには、Windows SDK を 64 ビット機にインストールする必要があります。そうでない場合には 64 ビットツール群はインストールされません。

TET PDF IFilter が正しく登録されている場合、*FiltReg.exe* の出力は下記のような行を含んでいます：

```
Filters loaded by extension:
...
.pdf --> PDFlib TET PDF IFilter 64-bit (C:\Program Files\PDFlib\TET PDF IFilter 5.1 64-bit\bin\TETPDFIFilter.dll)
```

TET PDF IFilter が正しく登録されていない場合には、これを手動で登録する必要があります（6 ページの「手動インストール」を参照）。

Windows Search をインストールしてある場合には、正しい IFilter 登録を右記のようにテストできます：**「スタート」→「コントロールパネル」→「インデックスのオプション」→「詳細設定」→「ファイルの種類」**をクリック。すると、ファイルタイプと、関連付いたフィルタの長いリストが生成されます。このリストの中で、**「拡張子」**列の中の**「*pdf*」**へスクロールします。**「フィルターの説明」**列の中の対応する項目に**「*PDFlib TET PDF IFilter 32-bit*」**（64 ビット版なら ***64-bit***）と書いてあるはずです。

TET PDF IFilter と Adobe Acrobat TET PDF IFilter の後に Adobe Reader か Adobe Acrobat をインストールすると（あるいは Acrobat の自動修復モードを走らせると）、それらは TET PDF IFilter のレジストリ項目群を上書きします。この状況を直すには、TET PDF IFilter を修復モードで走らせるか、6 ページの「手動インストール」の項に従って TET PDF IFilter DLL を手動で登録します。

ライセンスキーは利用可能か？ TET PDF IFilter は、Windows XP/Vista/7/8/10 では商用ライセンスキーなしで使えますが、Windows Server ではライセンスキーが必要です。サーバシステムを使用していて PDF インデックス作成が動作していないように見える場合は、TET PDF IFilter のライセンスキーを見つけられないでいる可能性があります。この場合には、TET PDF IFilter は評価モードで動作しますので、小規模文書に制限されます。

この状況は、Windows イベントログをチェックすることで検知できます（81 ページの「アプリケーションイベントログ」を参照）。ライセンスキーの問題が起きた場合には、ソースが「*TET PDF IFilter*」で分類が「*TET Error*」の項目があります。そのエラーを内容としている行をダブルクリックしてエラーメッセージを調べてください。下記のテキストは、有効なライセンスキーが見つからなかったことを示しています：

```
TET API Error in TetIFilter::Init: open_document:
Invalid license key (error number 1986)
```

このメッセージを見つけたときは、レジストリにライセンスキーを入力する必要があります（6 ページの「手動インストール」を参照）。

6.2 TET PDF IFilter の動作上の問題

TET PDF IFilter が期待どおりに動作していないように見える場合には、以下に説明する分析手法が役立つ可能性があります。

ロックされている PDF 文書がインデックスされない　何らかのアプリケーションが PDF ファイルをロックしているとき、TET PDF IFilter はその文書をインデックスできません。特に、ファイルは Acrobat で開かれている間はロックされています。IFilter クライアントは、ロックされていた文書を後で再試行するかもしれませんが、ロックされている文書が解放されるまではインデックスは不完全になります。ですので、インデックス作成中は PDF 文書を Acrobat で表示することを避けることを推奨します。

XML 構成ファイルが読み込まれない　TET PDF IFilter のための XML 構成ファイルを使用するには、その構成ファイルを指し示す、然るべきレキストリ項目を作成する必要があります（67 ページの 5 章「XML 構成ファイル」参照）。XML 構成ファイルが読み込めなかった場合、または XML 解析が失敗した場合には、エラーメッセージがイベントログへ書き込まれ（81 ページの「アプリケーションイベントログ」参照）、インデックス作成は一切実行されません。

6.3 PDF 文書群が完全にはインデックスされない

SharePoint には、大容量文書のインデックス作成に影響を与えるさまざまな制約があります。これらの制約は Microsoft の文書ではよく説明されていないので、以下の諸注意は、Microsoft のサポート記事とブログに基づいて情報を収集しているものです。これらの諸注意は公認のものではありませんので、疑点があれば Microsoft に問い合わせてください。

SharePoint 2016・2013 の制約 SharePoint は、文書のインデックス作成にいくつかの制約を課します。SharePoint における固定の、または構成可能な制約に関して、下記にさらに情報があります：

https://technet.microsoft.com/en-us/library/cc262787%28v=office.15%29.aspx

下記の SharePoint の制約が TET PDF IFilter に課せられます：

- 最大ファイルサイズ（***MaxDownloadSize***）：クロールされインデックスされる文書の最大ファイルサイズを指定します。SharePoint の場合のデフォルト値は 64 MB です。
- 最大膨張係数（***MaxGrowFactor***）：インデックスされる 1 文書あたりのテキストの最大量を決定するために ***MaxDownloadSize*** 値に乗じられる係数です。この係数が必要な理由は、テキストがファイル内部で圧縮されている場合があるからです。PDF 文書ではたいていそうです（単位：なし、デフォルト：4）。
- 解析対象内容サイズ:1 文書あたり何個のキャラクタをインデックスできるかを指定します。SharePoint は 2 百万キャラクタという固定の上限を有します。この上限は変更できません。

SharePoint に対する最大ファイルサイズと膨張係数を変更 「*SharePoint* 管理シェル」で下記のコマンドを実行します（検索サービスが複数ある場合には、1 番目のコマンドに -*id <GUID of SSA>* を付加）：

```
$ssa = Get-SPEnterpriseSearchServiceApplication
$ssa.SetProperty("MaxDownloadSize", ...新しい値...)
```

同様のシーケンスを実行して ***MaxGrowFactor*** も設定できます。現在の値をチェックするには下記のようにします：

```
$ssa = Get-SPEnterpriseSearchServiceApplication
$ssa.GetProperty("MaxDownloadSize")
```

6.4 デバッグ機能

検索結果が自分の期待にそわず、インデックスされた文書群から抽出されたテキスト内容に問題があると推測している場合には、以下に説明するデバッグツールが有用である可能性があります。

アプリケーションイベントログ　TET PDF IFilter は、Windows イベントログの中に、さまざまなイベントについて項目を作成します。このアプリケーションイベントログは以下のようにしてチェックできます：

- Windows Vista/7：**「スタート」**をクリックし、**「検索の開始」**ボックス内に**「イベントビューア」**（または照応するローカライズされた用語。たとえばドイツ語版 Windows なら「*Ereignisanzeige*」）と打ち、**「イベント ビューア」**プログラムをクリック。「イベント ビューア」ウィンドウの中で**「*Windows* ログ」→「アプリケーション」**をクリック。Windows 8/10：**「イベント ビューアー」**（または照応するローカライズされた用語。たとえばドイツ語版 Windows なら「*Ereignisprotokolle*」）を開きます。「イベント ビューアー」ウィンドウ内で**「*Windows* ログ」→「*Application*」**をクリックします。
- TET PDF IFilter のイベント群は、ソース「***TET PDF IFilter***」としてリストされています。メッセージを内容とする行をクリックして、詳細メッセージを調べます。

アプリケーションイベントログ内の項目群は、レジストリ値

```
HKEY_LOCAL_MACHINE\SOFTWARE\PDFlib\TET PDF IFilter5\5.1\logging
```

を表 6.1 に従った DWORD 値に設定することによって、さまざまなクラスのイベントについて有効にすることができます。このログ記録レベルはデフォルトでは 1 に設定されています。

表 6.1 Windows イベントログのためのログ記録レベル一覧

レベル	要約	ログ記録されるイベント
0	致命的エラー	TET PDF IFilter の動作が不可能になる致命的な問題。たとえば XML 構成ファイル内のエラーや、ユーザーが与えた TET オプションリストが無効の場合や、ライセンスの問題など。文書を処理することは一切できません。
1 (default)	文書固有のエラー	レベル 0 の内容に加え、文書固有の致命的エラー。たとえばメモリ不足など。ある 1 つの文書が処理できません。
2	テキストを全く抽出できない	レベル 1 の内容に加え、テキストを全く抽出できない文書。たとえば PDF 文書ではない場合や、必須のパスワードが与えられていない場合や、完全に破損した PDF や、動的 XFA フォームなど。XMP 解析の問題もこのレベルでレポートされます。
3	完全なファイル一覧	レベル 2 の内容に加え、処理された各文書が、その結果にかかわらずレポートされます。これは、ある特定の文書が TET PDF IFilter へ実際に渡されたことを確認したいときに有用です。
4	処理の詳細	レベル 3 の内容に加え、XMP メタデータ解釈エラーと、文書処理に関するさらなる詳細。デバッグとサポートのために有用でしょう。

注記 Windows イベントログへの書き込みが失敗した場合（権限の問題などによって）には、いくらかの緊急ログ記録情報が *OutputDebugString()* 機構を通じてフォールバックとして出力されます。このフォールバックログ記録出力を表示するには、*Sysinternals* アプリケーション *Dbgview.exe* を用いて、「*Capture*」メニュー内「*Capture Global Win32*」でオプショ

ンをオンにします。特定の問題文書に対してさらに診断が必要な場合には、*filtdump.exe* を実行して、Windows イベントログ内に生成されるエントリ群を調べるという方法もあります。

インデックス処理時刻をチェック　TET PDF IFilter は、その処理日時を、正準名 *PDFlib.TET.indextime*（ユーザーインタフェース名 *indexname*）を持つプロパティへ格納します。これは、ある文書が TET PDF IFilter によって処理済みかどうかを、あるいはいつ処理されたかをチェックしたいときに有用でしょう。

たとえば、TET PDF IFilter が Windows Search とともに使用するよう構成されている場合であれば、このインデックス時刻プロパティをエクスプローラーか検索ウィンドウに表示させることもできます。そのためには、**「表示」→「列の追加」→「列の選択」**をクリックして、*indextime* を選択します。そのうえで**「表示」→「詳細」**をクリックすれば、詳細なファイル情報が表示されます。新しい列に、TET PDF IFilter によって処理された各 PDF ファイルに対するインデックス処理時刻が表示されています。この列の内容がないものは、その文書が TET PDF IFilter によって処理されていないことを示します。

問題を起こす文書を特定　使っている IFilter クライアントによって、イベントログのエントリは、問題のファイルの名前を含んでいることもあれば、いないこともあり、また、ファイル名は有用であることもあれば、そうでないこともあります。たとえば、SharePoint は文書を HTTP を通じてダウンロードして一時ローカルコピーを生成します。対照的に、Windows Search によって使用されるクローラは通常、インデックスされたファイルの名前を与えますが、圧縮された ZIP アーカイブを解凍する際など、特殊な場合においてはこれを行いません。ですのでイベントログは有用なファイル名をつねに内容として持っているわけではありません。問題の PDF 文書を特定する助けとして、イベントログのエントリはファイルサイズをバイト単位で表したものを内容として持っています。検索エンジン自体を使って問題の文書を特定することも可能です。

- Windows Search では、下記のクエリ表現が使えます（12345 はファイルサイズをバイト単位で表したものとして）：

```
size: = 12345
```

- SharePoint では、ファイルをフィルタしようとして失敗した試みを、SharePoint クロールログ内で特定することが可能です（Shared Services Administration：「*SharedServices*」→「*Search Settings*」→「*Crawl Log*」）。ここにリストされている PDF 文書に関するエラーは、Windows アプリケーションイベントログ内で TET PDF IFilter によって発せられているエラーに照応しています。このクロールログ内のファイルのファイルサイズを、イベントログ内の項目と比較すれば、問題を起こしている文書を特定することが可能です。

また、*Filtering/@errorIndicator* 構成属性を用いて、問題を起こした文書を特定する文字列をインデックス内へ書き出させることもできます（69 ページの 5.2「XML エレメント・属性一覧」を参照）。

文書からどのプロパティとテキストを出力したか？　ある特定の文書から TET PDF IFilter が抽出している正確なテキストを見るには、Windows SDK に入っているツール *FiltDump.exe* が使えます。64 ビット TET PDF IFilter DLL をテストするにはこのツールの 64 ビット版が必要です。詳しくは下記を参照してください：

msdn.microsoft.com/en-us/library/ms692535(VS.85).aspx

-o オプションを用いて、*FiltDump.exe* の出力を UTF-16 符号化されたファイルへリダイレクトさせることも可能です。こうすると、TET PDF IFilter によって出力されるテキストに対する正確な Unicode テキストと、検知されたロケール（LCID）を見ることが可能になります。サンプル呼び出し：

```
FiltDump.exe -o udhr_japanese.txt udhr_japanese.pdf
```

サンプル出力：

```
FILE: udhr_japanese.pdf
IFILTER: CLSID == {47A1AF35-C345-475D-AE68-EB07E948BD07}
IFILTER: Using IPersistStream
IFILTER: IFilter->Init returned IFILTER_FLAGS_OLE_PROPERTIES flag


CHUNK: ----------------------------------------------------------------
Attribute = {007867F0-C59B-43FC-AB1E-8EEE77057254}\4 (PDFlib.TET.indextime)
    idChunk = 3
    BreakType = 2 (Sentence)
    Flags (chunkstate) =  (Value)
    Locale = 1031 (0x407)
    IdChunkSource = 3
    cwcStartSource = 0
    cwcLenSource = 0

VALUE: ----------------------------------------------------------------
Type = 64 (0x40), VT_FILETIME
Value = "2010/06/10:08:28:04.587"

CHUNK: ----------------------------------------------------------------
    Attribute = {B725F130-47EF-101A-A5F1-02608C9EEBAC}\19 (System.Search.Contents)
    idChunk = 11
    BreakType = 2 (Sentence)
    Flags (chunkstate) =  (Text)
    Locale = 9 (0x9)
    IdChunkSource = 11
    cwcStartSource = 0
    cwcLenSource = 0

TEXT: ----------------------------------------------------------------
UDHR - Japanese

CHUNK: ----------------------------------------------------------------
    Attribute = {B725F130-47EF-101A-A5F1-02608C9EEBAC}\19 (System.Search.Contents)
    idChunk = 12
    BreakType = 2 (Sentence)
    Flags (chunkstate) =  (Text)
    Locale = 17 (0x11)
    IdChunkSource = 12
    cwcStartSource = 0
    cwcLenSource = 0

TEXT: ----------------------------------------------------------------
...この文書のテキスト内容...
```

TET カーネルログ記録　TET PDF IFilter によって駆動されている際の TET カーネルの挙動を解析するために、詳細な TET カーネルログ記録を有効にすることも可能です。TET ログ記録を有効にするには下記のようにします：

- XML 構成ファイル内で適切な TET オプション群を設定することによって（必ず XML 構成ファイルのファイル名を指定してください。67 ページの 5 章「XML 構成ファイル」を参照）：

```
<Tet>
        <TetOptions>logging={filename=C:\debug.log classes={pcos=2}}</TetOptions>
</Tet>
```

こうすると、TET 関数への内部呼び出しに関する詳細やエラーメッセージなどを含むログ・ファイルが生成されます。必ず、TET PDF IFilter を呼び出すサービスから書き込み可能なファイル名を用いてください。また、TET ログ記録は大量の出力を生成し、フィルタリング処理の速度を落とすことに留意しておいてください。

- PowerShell を用いて環境変数を設定することによって：

```
PS C:\> ${env:TETLOGGING} = "filename=tet.log classes={filesearch=3}"
```

A 定義済みメタデータプロパティ

定義済みプロパティ群は、TET PDF IFilter に内部的に知られています（25 ページの 2.3「定義済みプロパティ」参照）。各表の 1 列目に正準プロパティ名を挙げています。表 A.1 の 2 列目にはシェルプロパティ群の表示名（ユーザーインタフェース名）を挙げています。この表示名はエクスプローラウィンドウの詳細表示で列名として表示されます。他の定義済みプロパティ群の表示名は *PDFlib.TET* 接頭辞をなくすことによって得られます。たとえばプロパティ *PDFlib.TET.pdfa* の表示名は *pdfa* です。

A.1 シェルプロパティセットコレクション

シェルプロパティセットコレクションはすでに Windows に知られています。正準名と表示名（空白なしで）をクエリ内で使えます。ドイツ語など非英語の表示名を知る方法は 9 ページの「正準プロパティ名とローカライズされた表示名」を参照してください。

表 A.1 シェルプロパティセットコレクションの定義済みプロパティ。これらはすでに Windows に知られています

正準プロパティ名	英語の表示名（ラベル）	データ型	多値	プロパティセット GUID ／プロパティ ID	ソース：XMP プロパティまたは pCOS パス
System.Document.Contributor	*Contributors*	*String*	○	*F334115E-DA1B-4509-9B3D-119504DC7ABB/100*	*dc:contributor*
System.Document.DateCreated	*Content created*	*DateTime*	×	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/12*	*xmp:CreateDate, /Info/CreationDate*
System.Document.DateSaved[1]	*Date last saved*	*DateTime*	×	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/13*	*xmp:ModifyDate, /Info/ModDate*
System.Document.DocumentID	*Document ID*	*String*	×	*E08805C8-E395-40DF-80D2-54F0D6C43154/100*	*dc:identifier*
System.Document.PageCount	*Pages*	*Int32*	×	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/14*	*length:pages*
System.Document.Version	*Version number*	*String*	×	*D5CDD502-2E9C-101B-9397-08002B2CF9AE/29*	*xmpMM:VersionID*
System.Search.Contents	*n/a*	*String*	○	*B725F130-47EF-101A-A5F1-02608C9EEBAC/19*	PDF ページ群のテキスト内容
System.Title	*Title*	*String*	×	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/2*	*dc:title["x-default"], /Info/Title*
System.Subject	*Subject*	*String*	×	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/3*	*dc:description["x-default"], /Info/Subject*
System.Author	*Authors*	*String*	○	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/4*	*dc:creator, pdf:Author, xmp:Author, /Info/Author*
System.DateModified[1]	*Date modified*	*DateTime*	×	*B725F130-47EF-101A-A5F1-02608C9EEBAC/14*	*xmp:ModifyDate, /Info/ModDate*
System.Keywords	*Tags*	*String*	○	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/5*	*pdf:Keywords, /Info/Keywords*
System.MIMEType	*n/a*	*String*	×	*0B63E350-9CCC-11D0-BCDB-00805FCCCE04/5*	`application/pdf`（固定値）
System.ApplicationName	*Program name*	*String*	×	*F29F85E0-4FF9-1068-AB91-08002B27B3D9/18*	*xmp:CreatorTool, /Info/Creator*
System.Kind	*Kind*	*String*	×	*1E3EE840-BC2B-476C-8237-2ACD1A839B22/3*	`Document`（固定値）

1. Windows Search などいくつかの IFilter クライアントは、PDF ベースの値を、ファイルシステムプロパティでオーバライドします。

A.2 PDF プロパティセットコレクション

このコレクションのプロパティ群はデフォルトで出力されます。これらのプロパティを Windows Search で使うには、*proptool.exe* ユーティリティを用いてこれらを登録する必要があります（41 ページの 3.1「Windows Search におけるメタデータプロパティ」参照）。このコレクションのためのプロパティ記述群はファイル *predefined_properties.propdesc* 内にあります。

表 A.2 PDF プロパティセットコレクションの定義済みプロパティ

正準プロパティ名	データ型	多値	プロパティセット GUID ／プロパティ ID	ソース： XMP プロパティまたは pCOS パス
PDFlib.TET.pdfversion （PDF バージョンに 10 を乗じた内容。例：「17」）	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/2*	*pdfversion*
PDFlib.TET.fullpdfversion （PDF バージョンに 100 を乗じた内容。例：「173」）	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/12*	*fullpdfversion*
PDFlib.TET.pdfa[1]	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/3*	*pdfa*
PDFlib.TET.pdfe[1]	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/13*	*pdfe*
PDFlib.TET.pdfua[1]	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/14*	*pdfua*
PDFlib.TET.pdfvt[1]	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/15*	*pdfvt*
PDFlib.TET.pdfx[1]	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/4*	*pdfx*
PDFlib.TET.font	*String*	○	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/5*	*fonts[*]/name*
PDFlib.TET.producer	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/10*	*/Info/Producer*
PDFlib.TET.trapped	*String*	×	*E544AFE6-13E2-40F1-A702-DCEBE8FB7B02/11*	*/Info/Trapped*

1. この規格のどの部分にも文書が準拠していない場合に pCOS によって返される値 none は、抑制されます。

A.3 文書 XMP メタデータプロパティセットコレクション

このコレクションのプロパティ群はデフォルトでは出力されず、XML 構成ファイル内で *PropertySetCollection/@documentXmp="true"* を用いて有効化する必要があります。これらのプロパティを Windows Search で使うには、*proptool.exe* ユーティリティを用いてこれらを登録する必要があります（41 ページの 3.1「Windows Search におけるメタデータプロパティ」参照）。このコレクションのためのプロパティ記述群はファイル *predefined_properties.propdesc* 内にあります。

表 A.3 文書 XMP メタデータプロパティセットコレクションの定義済みプロパティ（XMP 仕様より）

正準プロパティ名	データ型	多値	プロパティセット GUID ／プロパティ ID	ソース： XMP プロパティまたは pCOS パス
Dublin Core				
PDFlib.TETPDFIFilter.dc.contributor	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/2*	*dc:contributor*
PDFlib.TETPDFIFilter.dc.coverage	*String*	×	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/3*	*dc:coverage*
PDFlib.TETPDFIFilter.dc.creator	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/4*	*dc:creator*
PDFlib.TETPDFIFilter.dc.date	*DateTime*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/5*	*dc:date*
PDFlib.TETPDFIFilter.dc.description	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/6*	*dc:description*
PDFlib.TETPDFIFilter.dc.format	*String*	×	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/7*	*dc:format*
PDFlib.TETPDFIFilter.dc.identifier	*String*	×	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/8*	*dc:identifier*
PDFlib.TETPDFIFilter.dc.language	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/9*	*dc:language*
PDFlib.TETPDFIFilter.dc.publisher	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/10*	*dc:publisher*
PDFlib.TETPDFIFilter.dc.relation	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/11*	*dc:relation*
PDFlib.TETPDFIFilter.dc.rights	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/12*	*dc:rights*
PDFlib.TETPDFIFilter.dc.source	*String*	×	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/13*	*dc:source*
PDFlib.TETPDFIFilter.dc.subject	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/14*	*dc:subject*
PDFlib.TETPDFIFilter.dc.title	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/15*	*dc:title*
PDFlib.TETPDFIFilter.dc.type	*String*	○	*D92BB3CA-CE2B-4B9B-972A-5BF54B468171/16*	*dc:type*
XMP Media Management				
PDFlib.TET.xmpMM.DocumentID	*String*	○	*743CD065-7F47-4b4d-ACF1-F0A09C92EAA9/2*	*xmpMM:DocumentID*
PDFlib.TET.xmpMM.InstanceID	*String*	×	*743CD065-7F47-4b4d-ACF1-F0A09C92EAA9/3*	*xmpMM:InstanceID*
PDFlib.TET.xmpMM.VersionID	*String*	×	*743CD065-7F47-4b4d-ACF1-F0A09C92EAA9/4*	*xmpMM:VersionID*
PDFlib.TET.xmpMM.OriginalDocumentID	*String*	×	*743CD065-7F47-4b4d-ACF1-F0A09C92EAA9/5*	*xmpMM:OriginalDocumentID*
XMP Basic				
PDFlib.TETPDFIFilter.xmp.Advisory	*String*	○	*C60E822A-074F-4BD5-9889-6EBD372F2000/2*	*xmp:Advisory*

表 A.3 文書 XMP メタデータプロパティセットコレクションの定義済みプロパティ（XMP 仕様より）

正準プロパティ名	データ型	多値	プロパティセット GUID ／プロパティ ID	ソース： XMP プロパティまたは pCOS パス
PDFlib.TETPDFIFilter.xmp.BaseURL	*String*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/3*	*xmp:BaseURL*
PDFlib.TETPDFIFilter.xmp.CreateDate	*DateTime*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/4*	*xmp:CreateDate*
PDFlib.TETPDFIFilter.xmp.CreatorTool	*String*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/5*	*xmp:CreatorTool*
PDFlib.TETPDFIFilter.xmp.Identifier	*String*	○	*C60E822A-074F-4BD5-9889-6EBD372F2000/6*	*xmp:Identifier*
PDFlib.TETPDFIFilter.xmp.Label	*String*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/7*	*xmp:Label*
PDFlib.TETPDFIFilter.xmp.MetadataDate	*DateTime*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/8*	*xmp:MetadataDate*
PDFlib.TETPDFIFilter.xmp.ModifyDate	*DateTime*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/9*	*xmp:ModifyDate*
PDFlib.TETPDFIFilter.xmp.Nickname	*String*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/10*	*xmp:Nickname*
PDFlib.TETPDFIFilter.xmp.Rating	*Int32*	×	*C60E822A-074F-4BD5-9889-6EBD372F2000/11*	*xmp:Rating*
XMP Rights Management				
PDFlib.TETPDFIFilter.xmpRights.Certificate	*String*	×	*0DE7A11C-E2C5-4EFA-8017-BECD888E7EC9/2*	*xmpRights:Certificate*
PDFlib.TETPDFIFilter.xmpRights.Marked	*Boolean*	×	*0DE7A11C-E2C5-4EFA-8017-BECD888E7EC9/3*	*xmpRights:Marked*
PDFlib.TETPDFIFilter.xmpRights.Owner	*String*	○	*0DE7A11C-E2C5-4EFA-8017-BECD888E7EC9/4*	*xmpRights:Owner*
PDFlib.TETPDFIFilter.xmpRights.UsageTerms	*String*	○	*0DE7A11C-E2C5-4EFA-8017-BECD888E7EC9/5*	*xmpRights:UsageTerms*
PDFlib.TETPDFIFilter.xmpRights.WebStatement	*String*	×	*0DE7A11C-E2C5-4EFA-8017-BECD888E7EC9/6*	*xmpRights:WebStatement*
XMP Basic Job Ticket				
PDFlib.TETPDFIFilter.xmpBJ.JobRef	*String*	○	*EBC983EF-C1CF-45C8-A29E-993543A0ECFB/2*	*xmpBJ:JobRef*
XMP Paged-Text				
PDFlib.TETPDFIFilter.xmpTPg.NPages	*Int32*	×	*7A9EB492-35AB-49FE-B364-A21FC9575C28/2*	*xmpTPg:NPages*
PDFlib.TETPDFIFilter.xmpTPg.PlateNames	*String*	○	*7A9EB492-35AB-49FE-B364-A21FC9575C28/3*	*xmpTPg:PlateNames*
Adobe PDF				
PDFlib.TETPDFIFilter.pdf.Keywords[1]	*String*	×	*17EB8447-FC9B-4D4D-81DF-31E9AA770CBF/2*	*pdf:Keywords*
PDFlib.TETPDFIFilter.pdf.PDFVersion[1]	*String*	×	*17EB8447-FC9B-4D4D-81DF-31E9AA770CBF/3*	*pdf:PDFVersion*
PDFlib.TETPDFIFilter.pdf.Producer[1]	*String*	×	*17EB8447-FC9B-4D4D-81DF-31E9AA770CBF/4*	*pdf:Producer*

1. これらの XMP プロパティはめったに使用されません。照応するプロパティ `System.Keywords`・`PDFlib.TET.fullpdfversion`・`PDFlib.TET.producer` をかわりに使用することが推奨されます。

A.4 XMP 画像メタデータプロパティセットコレクション

このコレクションのプロパティ群はデフォルトでは出力されず、XML 構成ファイル内で *PropertySetCollection/@imageXmp="true"* を用いて有効化する必要があります。これらのプロパティを Windows Search で使うには、*proptool.exe* ユーティリティを用いてこれらを登録する必要があります（41 ページの 3.1「Windows Search におけるメタデータプロパティ」参照）。このコレクションのためのプロパティ記述群はファイル *predefined_properties.propdesc* 内にあります。

表 A.4 XMP 画像メタデータプロパティセットコレクションの定義済みプロパティ（XMP 2005 仕様内の Photoshop スキーマより）

正準プロパティ名	データ型	多値	プロパティセット GUID ／プロパティ ID	ソース： XMP プロパティまたは pCOS パス
PDFlib.TETPDFIFilter.images.photoshop.AuthorsPosition	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/2*	*photoshop:AuthorsPosition*
PDFlib.TETPDFIFilter.images.photoshop.CaptionWriter	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/3*	*photoshop:CaptionWriter*
PDFlib.TETPDFIFilter.images.photoshop.Category	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/4*	*photoshop:Category*
PDFlib.TETPDFIFilter.images.photoshop.City	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/5*	*photoshop:City*
PDFlib.TETPDFIFilter.images.photoshop.Country	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/6*	*photoshop:Country*
PDFlib.TETPDFIFilter.images.photoshop.Credit	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/7*	*photoshop:Credit*
PDFlib.TETPDFIFilter.images.photoshop.DateCreated	*DateTime*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/8*	*photoshop:DateCreated*
PDFlib.TETPDFIFilter.images.photoshop.Headline	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/9*	*photoshop:Headline*
PDFlib.TETPDFIFilter.images.photoshop.Instructions	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/10*	*photoshop:Instructions*
PDFlib.TETPDFIFilter.images.photoshop.Source	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/11*	*photoshop:Source*
PDFlib.TETPDFIFilter.images.photoshop.State	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/12*	*photoshop:State*
PDFlib.TETPDFIFilter.images.photoshop.SupplementalCategories	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/13*	*photoshop:SupplementalCategories*
PDFlib.TETPDFIFilter.images.photoshop.TransmissionReference	*String*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/14*	*photoshop:TransmissionReference*
PDFlib.TETPDFIFilter.images.photoshop.Urgency	*Int32*	○	*C9F08C60-189D-11DD-8441-0002A5D5C51B/15*	*photoshop:Urgency*

A.5 内部プロパティセットコレクション

このコレクションのプロパティ群はデフォルトで出力されます。これらのプロパティを Windows Search で使うには、*proptool.exe* ユーティリティを用いてこれらを登録する必要があります（41 ページの 3.1「Windows Search におけるメタデータプロパティ」参照）。このコレクションのためのプロパティ記述群はファイル *predefined_properties.propdesc* 内にあります。

表 A.5 内部プロパティセットコレクションの定義済みプロパティ

正準プロパティ名	データ型	多値	プロパティセット GUID ／プロパティ ID	ソース： XMP プロパティまたは pCOS パス
PDFlib.TET.version	*String*	×	*007867F0-C59B-43FC-AB1E-8EEE77057254/2*	5.1 （パッチレベル番号が付く場合もあります）
PDFlib.TET.indextime	*DateTime*	×	*007867F0-C59B-43FC-AB1E-8EEE77057254/4*	インデックスランの日時
PDFlib.TET.eval	*Int32*	×	*007867F0-C59B-43FC-AB1E-8EEE77057254/5*	IFilter が評価モードで走っているなら例外番号

B 更新履歴

本マニュアルの更新履歴

日付	変更
2017 年 8 月 14 日	▸ *TET PDF IFilter 5.1*（*TET 5.1* をベースとする）のための更新
2016 年 7 月 28 日	▸ *SQL Server 2014・2016* への対応を記述
2016 年 4 月 21 日	▸ *TET PDF IFilter 5.0*（*TET 5.0* をベースとする）のための更新
2015 年 1 月 27 日	▸ *TET PDF IFilter 4.4*（*TET 4.4* をベースとする）のための更新
2014 年 5 月 26 日	▸ *TET PDF IFilter 4.3*（*TET 4.3* をベースとする）のための更新
2013 年 5 月 16 日	▸ *TET PDF IFilter 4.2*（*TET 4.2* をベースとする）のための更新
2012 年 10 月 22 日	▸ *Exchange Server 2010* に関する項を追加（*TET 4.1p9* をベースとする）
2012 年 2 月 13 日	▸ *TET PDF IFilter 4.1*（*TET 4.1* をベースとする）のための更新
2010 年 9 月 22 日	▸ *TET PDF IFilter 4.0p2*（*TET 4.0p2* をベースとする）のための更新
2010 年 7 月 22 日	▸ *TET PDF IFilter 4.0*（*TET 4.0* をベースとする）のための更新
2008 年 8 月 06 日	▸ *Search Server* のための更新
2008 年 6 月 16 日	▸ *TET PDF IFilter 3.0*（*TET 3.0pre2* をベースとする）
2008 年 6 月 6 日	▸ *TET PDF IFilter 3.0 beta3*（*TET 3.0pre2* をベースとする）
2008 年 5 月 09 日	▸ *TET PDF IFilter 3.0 beta2*（*TET 3.0pre1* をベースとする）のための初版

索引

PDFlib GmbH

PDFlib GmbH
Franziska-Bilek-Weg 9
80339 München, Germany
www.pdflib.com
電話 +49 • 89 • 452 33 84-0
fax +49 • 89 • 452 33 84-99

疑問がおありの際は、PDF メーリングリストと、
tech.groups.yahoo.com/group/pdflib のアーカイブをチェックしてください

ライセンスに関するお問い合わせ
sales@pdflib.com

サポート
support@pdflib.com （お使いのライセンス番号をお書きください）