Introduction

Appligent, Inc.
May 14, 2001

Day One

- **PDF Overview**
  - History
  - Properties
  - Imaging Model
  - Annotations & Bookmarks
  - PDF 1.4
- **Working with PDF**
  - PDF File Format Details
    - The Structure of a PDF File
    - Elements in the Body of a PDF File
    - Elements of a Page
  - Acrobat SDK
    - The AS Layer
    - The Cos Layer
    - The PD Layer

Working with PDF

Mark Gavin
Appligent, Inc.
May 14, 2001

Working with PDF

- **The Structure of a PDF File**
- **Elements in the Body of a PDF File**
- **The Elements of a Page**
- **The Acrobat SDK**
- **The AS Layer**
- **The Cos Layer**
- **The PD Layer**

Appligent

The Structure of a PDF File

The Structure of a PDF File

Where the rubber hits the road.

- **Four Basic Parts**
  - Header
  - Body
  - Cross Reference
  - Trailer
- **Incremental Save**
  - Body
  - Supplemental Cross Reference
  - Trailer

Appligent

PDF Header

- **%PDF**
- **PDF Version Number**
  - PDF1.0, PDF1.1, PDF1.2, PDF1.3, PDF1.4
- **Binary Data**
  - %„œ”
- **Example**
  - %PDF-1.3
  - %„œ”

PDF Body

- **A collection of indirect objects representing the contents of a document.**
- **The basic structure of an object is as follows:**
  - Object Number
  - Generation Number
  - obj
  - <<
  - -- Place stuff here --
  - >>
  - endobj

Appligent

PDF Body Examples

```
64 0 obj
<<
/Type /Font
/Subtype /Type1
/Name /F5
/Encoding 60 0 R
/BaseFont /Courier-Bold
>>
endobj
```

PDF Cross Reference

- **Permits random access to in-direct objects within the file, so that the entire file need not be read to locate any particular object.**

PDF Cross Reference Example

```
xref
0 8
0000000000 65535 f
0000000016 00000 n
0000000069 00000 n
0000000133 00000 n
0000000317 00000 n
0000000533 00000 n
0000000637 00000 n
0000003907 00000 n
```

PDF Trailer

- **Size**
  - Entries in Cross Reference Table
- **Info**
  - Object Number for Info Dictionary
- **Root**
  - Object Number for Root Dictionary
- **ID**
  - Two strings representing document hash values
  - One permanent, one changed with each update
- **Encrypt**
  - Object Number for the Encrypt Dictionary

Appligent

PDF Trailer Example

```
trailer
<<
/Size 8
/Info 3 0 R
/Root 1 0 R
/ID[<fe8cd3f12b75eb14b82b02f1b95b46ac>
<fe8cd3f12b75eb14b82b02f1b95b46ac>]
>>
startxref
3928
%%EOF
```

Linearized

- **Byte Serving**
- **Partial Cross Reference and Trailer are placed at the beginning of the file.**
- **Contains the essential information required to display the first page while the remainder of the file continues to download.**

Linearized Example

```
%PDF-1.3
%,„œ”
52 0 obj
<<
/Linearized 1
/O 56
/H [ 838 186 ]
/L 13971
/E 6268
/N 2
/T 12813
>>
endobj
```

Linearized Example Continued

```
52 16
0000000016 00000 n
0000000667 00000 n
0000000754 00000 n
0000000783 00000 n
0000001024 00000 n
0000001278 00000 n
0000001368 00000 n
0000001457 00000 n
0000001658 00000 n
0000003159 00000 n
0000003818 00000 n
0000005808 00000 n
0000005830 00000 n
0000005938 00000 n
0000000838 00000 n
0000001004 00000 n
```

What is a PDF File

A PDF file is really
a free format database file

- **Any individual object can be accessed at any time; independent of any other object.**
- **Objects can be used for general storage of just about any type of data; even data which can not be interpreted by Acrobat.**

Appligent

What is a PDF File

A Page Oriented Representation of Visual Data Intended to be Rendered by a Printer

- **Unless added suplementally; the PDF page has no knowledge of the data it represents**
  - Headers
  - Footers
  - Chapter Headings
  - Titles
  - Footnotes
  - Tables
  - Charts
- **Think of PDF as “Electronic Paper”**

Appligent

Elements in the Body of a PDF File

Objects and Dictionaries

- **Objects are composed of Dictionaries**
- **Dictionaries are composed of Name Value pairs**
- **A dictionary is delimited by the << and >> characters**

  <<
  Dictionary
  >>

- **Objects are distinguished by Type**

Appligent

Values

Types of Values

- **Name**
- **Integer**
- **Array**
- **Indirect Reference to an Object**
- **Stream**
- **Dictionary**

Dictionary Example

```
<<
/Type /Page
/MediaBox [ 0 0 612 792 ]
/Parent 2 0 R
/Contents 6 0 R
/Resources << /Font << /F15 5 0 R >>
/ProcSet [ /PDF /Text ] >>
/PieceInfo << /Illustrator8.0
<< /LastModified ()>> >>
>>
```

Document Structure

The Building Blocks of a Document

- **Catalog**
- **Pages Tree**
- **Outlines Tree**
- **Names Tree**
- **Forms Tree**

Catalog

- **A collection of references to where various objects can be found in the PDF document.**
  - The Root of the Pages tree
  - Bookmarks
  - Forms Data
  - Treads
- **Document Level Preferences**
  - Open Action
  - Page Layout
  - Page Mode

Catalog Example

```
327 0 obj
<<
/Type /Catalog
/Pages 325 0 R
/Outlines 332 0 R
/Threads 328 0 R
/Names 330 0 R
/OpenAction [ 331 0 R /XYZ null null null ]
/PageMode /UseOutlines
>>
```

Pages Tree

- **Ideally a binary tree structure**
- **References branches in the tree or leaf nodes**
- **Two basic types of objects:**
  - Pages - A branch to other branches or a leaf
  - Page - A leaf node

Pages Example

```
325 0 obj
<<
/Type /Pages
/Kids [ 331 0 R 1 0 R 4 0 R 129 0 R 132 0 R ]
/Count 5
>>
endobj
```

Page Example Continued

```
331 0 obj
<<
/Type /Page
/Parent 325 0 R
/Resources 354 0 R
/Contents 358 0 R
/CropBox [ 90 0 702 792 ]
/B [ 348 0 R 350 0 R ]
/MediaBox [ 0 0 792 792 ]
/Rotate 0
>>
```

Outlines Tree

Commonly Referred to as Bookmarks

- **A collection of user navigation aids allowing the user to quickly access defined locations**
- **A bookmark supports Destinations or Actions**
  - Destinations define a location within the document.
  - Actions can specify a location external to the document or a limitless host of other possabilities.

Appligent

Outlines Tree Example

```
14 0 obj
<<
/Title (6.0 Installing the SPDF Library)
/Dest (G290939)
/Parent 94 0 R
/Prev 13 0 R
/Next 17 0 R
/First 15 0 R
/Last 16 0 R
/Count 2
>>
endobj
```

Bookmark Example

```
333 0 obj
<<
/Title (1.0 New Features)
/Dest (G290899)
/Parent 332 0 R
/Next 343 0 R
/First 346 0 R
/Last 347 0 R
/Count 2
>>
endobj
```

Names Tree

Finding a location in the document;
even if it moves around

- **The Name Tree allows the programmer to locate an object using a name as opposed to an object number and generation number**
- **Useful if the document is altered significantly. For example:**
  - Pages Inserted
  - Pages Replaced
- **Dictionary type lookup to reference the current object number given the object name**

Appligent

Names Tree Example

```
330 0 obj
<<
/Dests 323 0 R
>>
endobj

323 0 obj
<<
/Kids [ 321 0 R 322 0 R ]
>>
```

Names Tree Example Continued

```
322 0 obj
<<
/Limits [ (G291008)(P.5)]
/Names [ (G291008)263 0 R (G291009)264 0 R
(G291010)265 0 R (G291011)266 0 R
(G291012)273 0 R (G291069)261 0 R
(G291072)260 0 R (G291083)267 0 R
(G291090)276 0 R (G291099)285 0 R
(G291104)289 0 R (G291145)268 0 R
(G291185)272 0 R (G30230)250 0 R
(G30232)251 0 R (G30234)270 0 R
(G30235)271 0 R (G30237)234 0 R
(G30239)235 0 R (L)306 0 R (P.1)
232 0 R (P.2)249 0 R (P.3)269 0 R
(P.4)281 0 R (P.5)305 0 R ]
>>
endobj
```

The Elements of a Page

The Elements of a Page

- **Parent Object**
- **Contents**
- **Resources**
- **Annotations and other stuff related to the page**

Contents

The stuff that gets drawn on the Page

- **A sequential stream of “Postscript” like commands which instruct a PDF viewing tool how to represent the “printed” page.**
- **Describes the appearance and placment of graphical elements on the page.**
- **Each page is represented by one or more content streams.**

Appligent

Contents Example

```
6 0 obj
<< /Filter [ /ASCII85Decode /FlateDecode ] /Length 7 0 R
>>
stream
8;X]TCN%t;')`”5&&`p3!Cmno;NJe+J^hfu%<.d
a`/!’gC*?P2’m6ScCC:_1V9YgqiY%OT$)QRgMa
...
@EH?e=Tma#$DO4l)e5=c””FaZ2#OB7C?pDo’
00@(f~>
endstream
endobj

7 0 obj
3175
endobj
```

Resources

Information required by the Page;
but, not necessarily stored with the Page.

- **Fonts**
- **Color Spaces**
- **Proc Sets**
- **Patterns**
- **XObjects**

Appligent

Resources Example

```
354 0 obj
<<
/ProcSet [ /PDF /Text ]
/Font << /F1 357 0 R /F2 356 0 R /F3 359 0 R >>
/ExtGState << /GS1 360 0 R >>
/ColorSpace << /Cs5 355 0 R >>
>>
endobj
```

The Acrobat SDK

The Acrobat SDK

- **Cos**
- **PD - Portable Document**
- **AS - Acrobat Support**
- **AV - Acrobat Viewer**

Cos

The Key to the Kingdom

- **Cos tools allow the programmer to directly manipulate the majority of informaiton saved in the PDF file.**
- **Cos is a double edged sword**
  - Gives the programmer incredible control.
  - Allows the programmer to create a PDF file which is unreadable.

Appligent

Portable Document

- **A set of high level functions used to create a manipulate documents and document objects.**
- **Primarily built using Cos**

Appligent

Acrobat Support

- **A collection of programming utilities which help in writting cross platform code.**
  - File System Management
  - File Management
  - Memory Management
- **This collection is not generally built using Cos.**

The AS Layer

The AS Layer

- **ASAtom**
- **ASFile**
- **ASFileSys**
- **ASStm**
- **Error Handling**
- **Fixed Point Math**
- **Memory Allocation**

Error Handling

Try, Catch, Throw

```
DURING

-- Write some code here

HANDLER
char       theMessage[256] ;
ASInt32    theError = ERRORCODE ;

fprintf( stderr, “# Error:%s\n”, theMessage ) ;
ASGetErrorString( theError, theMessage, sizeof(
theMessage ) ) ;

END_HANDLER
```

ASAtom

ASAtoms are hashed tokens that Acrobat uses in place of strings to optimize performance

- **ASAtomExistsForString**
  - Does the string already have an atom?
- **ASAtomFromString**
  - Converts a string to an atom.
- **ASAtomGetString**
  - Gets an atom's string.

ASAtom

```
if ( ASAtomExistsForString ( “Dest”, &theDestASAtom ) == false )
     theDestASAtom = ASAtomFromString( “Dest” ) ;

CosObj theDestCosObj = CosDictGet( theBookmarkCosObj, theDestASAtom ) ;
if ( CosObjEqual( theBookmarkCosObj, theNullCosObj ) == true )
     ASRaise( CosError( cosErrExpectedDict ) ) ;
```

ASFile & ASFileSys

A Wrapper for File System Services

- **There are three objects associated with file systems**
  - ASFile
  - ASFileSys
  - ASPathName
- **This collection of routines opens, closes, reads and writes files**
- **Note: File systems can be replaced**

Appligent

ASFileSys

- **ASGetDefaultFileSys**
- **ASPathFromPlatformPath**
- **ASFileSysCopyPath**
- **ASFileSysDIPathFromPath**
- **ASFileSysReleasePath**

ASPathFromPlatformPath

```
// Create the ASPathName
theASPathName = ASPathFromPlatformPath( inFileName ) ;
if ( theASPathName == (ASPathName)NULL )
    {
    fprintf( stderr, "Invalid path to PDF file - %s\n", inFileName ) ;
    ASRaise( ASFileError( fileErrOpenFailed ) ) ;
    }

-- Write some code here

// Release the ASPathName when done
ASFileSysReleasePath( NULL, theASPathName ) ;
theASPathName = ( ASPathName )NULL ;
```

ASFileSysDIPathFromPath

Device Independent Paths

```
theASFileSys = ASGetDefaultFileSys() ;

theDIPath = ASFileSysDIPathFromPath ( theASFileSys,
inASPathName, NULL ) ;
```

ASStm

Working With Streams

- **An ASStm is a stream a data which can be associated with three different sources.**
  - Memory
  - File
  - User supplied procedure
- **Stream Operations:**
  - Open
  - Close
  - Read
  - Write

Appligent

ASStm Example

Opening, Reading and Closing a Stream

```
ASStm theASStream = CosStreamOpenStm( theContentStreamCosObj,
cosOpenFiltered ) ;
while( true )
    {
    // read the next 1024 bytes
    ASInt32 theItemsRead = ASStmRead( theBufferPtr, sizeof( char ), 1024,
    theASStream ) ;
    fwrite( theBufferPtr, sizeof( char ), theItemsRead, theOutputFile ) ;
    if ( theItemsRead < 1024 )
        // we read less than 1024, so we must be done...
        break ;

    }   // end while


ASStmClose( theASStream ) ;
```

Memory Allocation

Wrapper functions for the standard C memory allocation routines

- **ASmalloc**
- **ASrealloc**
- **ASfree**

Memory Allocation Example

```
char * theBufferPtr = ( char * )ASmalloc( 1024 ) ;
if ( theBufferPtr == NULL )
    ASRaise( GenError( genErrNoMemory ) ) ;

-- Write some code here

ASfree( theBufferPtr ) ;
```

The Cos Layer

Cos Types

- **Object**
- **Dictionary**
- **Name**
- **Boolean**
- **Integer**
- **Fixed**
- **String**
- **Array**
- **Stream**
- **Document**
- **Null**

Cos Object

CosObj

- **Copy**
- **Destroy**
- **Enumerate**
- **Equal**
- **Get Document**
- **Get Generation Number**
- **Get ID**
- **Get Type**
- **Hash**
- **Is Indirect**

Appligent

Cos Document

CosDoc

- **Create**
- **Open**
- **Save**
- **Close**
- **Set Dirty**
- **Get Info Dictionary**
- **Get Root**

Cos Dictionary

CosDict

- **New**
  - CosNewDict
- **Known**
  - CosDictKnown
- **Get**
  - CosDictGet
- **Put**
  - CosDictPut
- **Remove**
  - CosDictRemove

Cos Dictionary Example

```
324 0 obj
<<
/CreationDate (D:19991128200419)
/Producer (Acrobat Distiller 4.0 for Macintosh)
/Author (mgavin)
/Title (StampPDF20Readme.fm)
/Creator (FrameMaker 5.5.6)
/ModDate (D:19991128200430-05’00’)
>>
endobj
```

Cos Name

CosName

New

CosNewName

Value

CosNameValue

Examples:

```
/Type /Font
/Subtype /Type1
/Encoding /MacRomanEncoding
/BaseFont /Courier-Bold
```

Cos Boolean

CosBoolean

New

CosNewBoolean

Value

CosBooleanValue

Examples:

```
/ImageMask true
/FitWindow true
/DGAP:FillWithBlankSpace true
/DGAP:AllowRedactionWithoutExemption
false
```

Cos Integer

CosInteger

New

CosNewInteger

Value

CosIntegerValue

Examples:

```
/Length 483
/MediaBox [ 0 0 612 792 ]
/Rotate 0
```

Note: If an Integer exceedes the exceeds the implementation limit for integers, it is converted to a real.

Cos Fixed

CosFixed

- **New**
  - CosNewFixed
- **Value**
  - CosFixedValue
- **Examples:**
  - 34.5, -3.62, 123.6, 4., -.002, 0.0
- **Note: A collection of Fixed Math functions can be found in the AS Layer of the SDK.**

Cos String

CosString

- **A sequence of literal characters enclosed in parentheses ( )**
- **New**
  - CosNewString
- **Get Value**
  - CosStringValue
- **Get and Set Hex Flag**
  The string can be written into the PDF file as Hex.
  - CosStringGetHexFlag
  - CosStringSetHexFlag

Appligent

Cos String Examples

```
/Title (3.0 Fixed in this Release)
/Dest (G290987)
/CreationDate (D:19991128200419)
/Producer (Acrobat Distiller 4.0 for Macintosh)
/Author (mgavin)
/Title (StampPDF20Readme.fm)
/Creator (FrameMaker 5.5.6)
/ModDate (D:19991128200430-05’00’)
```

Cos Array

CosArray

- **New**
- **Get**
- **Put**
- **Remove**
  - CosArrayRemove
  - CosArrayRemoveNth
- **Insert**
  - CosArrayInsert
- **Length**
  - CosArrayLength

Cos Array Example

```
/Kids [ 331 0 R 1 0 R 4 0 R 129 0 R 132 0 R ]
/CropBox [ 90 0 702 792 ]
/B [ 352 0 R ]
/MediaBox [ 0 0 792 792 ]
/ProcSet [ /PDF /Text ]
```

Cos Stream

CosStream

- **New**
  - CosNewStream
- **Get Dictionary**
  - CosStreamDict
- **Get Stream Length**
  - CosStreamLength
- **Open Stream**
  - CosStreamOpenStm
- **Get Stream Position**
  - CosStreamPos

Cos Stream Example One

```
201 0 obj
<< /Length 60 /Filter /FlateDecode >>
stream
Hâb``¯˜˜˜È”ßgÃòqÁŒù¸¸¸ï+Wñóókhhï/µKHHh·¬ÖÎ◊Øg  £`8 Ä    C
4
endstream
endobj
```

Cos Stream Example Two

```
56 0 obj
<< /Type /XObject /Subtype /Image /Mask 55 0 R /Width 357
/Height 4
/BitsPerComponent 8 /ColorSpace 148 0 R /Length 62
/Filter /FlateDecode >>
stream
Hâb`dbbf`dA ¨Hl6ÜQÄ 0, 7 `g á.  ¡@fa  ¯ u y
F ô  G    :Á
Û
endstream
endobj
```

Cos Null

CosNull

- **The Null Cos object is typically used to compare an existing Cos object will Null to determine if it is a valid Cos object.**
- **CosNewNull**

Working with Cos

CosString2CString

```
CosType       theCosType ;
char     *    theString ;
char     *    theStringPtr ;
ASInt32       theLength ;

DURING
theCosType = CosObjGetType( inCosObj ) ;
theString = CosStringValue( inCosObj, &theLength ) ;
if ( ! theString )
    ASRaise( CosError( cosErrExpectedString ) ) ;
if ( theLength <= 0 )
    E_RETURN( ( char * )NULL ) ;

theStringPtr = ( char * )ASmalloc( ( int )theLength + 1 ) ;
if ( ! theStringPtr )
    E_RETURN( ( char * )NULL ) ;

memcpy( theStringPtr, theString, ( int )theLength ) ;
theStringPtr[theLength] = '\0' ;
E_RETURN( theStringPtr ) ;

HANDLER
    ASRaise( ERRORCODE ) ;
END_HANDLER

return (char *)NULL ;
```

GetNamesDict

```
ASAtom   theNamesASAtom ;
CosObj   theNullCosObj = CosNewNull() ;
CosObj   theNamesCosDict = theNullCosObj ;

DURING
CosObj theCatalogCosObj = CosDocGetRoot( inCosDoc ) ;
if ( CosObjEqual ( theCatalogCosObj, theNullCosObj ) )
    ASRaise( CosError( cosErrExpectedDict ) ) ;

if ( ASAtomExistsForString ( “Names”, &theNamesASAtom ) == false )
    theNamesASAtom = ASAtomFromString( “Names” ) ;

if ( CosDictKnown ( theCatalogCosObj, theNamesASAtom ) == true )
    {
    theNamesCosDict = CosDictGet( theCatalogCosObj, theNamesASAtom ) ;
    E_RETURN ( theNamesCosDict ) ;
    }

HANDLER
    ASRaise( ERRORCODE ) ;
END_HANDLER

return theNamesCosDict ;
```

Adding an Integer to a New Cos Dictionary

```
CosDoc theCosDoc = PDDocGetCosDoc( inPDDoc ) ;

CosObj theAttributesDict = CosNewDict( theCosDoc, false, 5 ) ;

CosObj theLengthCosObj = CosNewInteger( theCosDoc, false, inLength ) ;

if ( ASAtomExistsForString ( “Length”, &theLengthASAtom ) == false )
    theLengthASAtom = ASAtomFromString( “Length” ) ;

CosDictPut( theAttributesDict, theLengthASAtom, theLengthCosObj ) ;
```

Working with CosArrays

```
ASInt32 theArrayLength = CosArrayLength( theNamesArray ) ;
if ( ASAtomExistsForString ( “D”, &theDASAtom ) == false )
    theDASAtom = ASAtomFromString( “D” ) ;

for ( index = 0; index < theArrayLength; index += 2 )
{
    CosObj theCosString = CosArrayGet( theNamesArray, index ) ;
    char * theString = CosString2CString( theCosString ) ;
    if ( strcmp( theString, inString ) != 0 )
        break ;

    CosObj theDestCosObj = CosArrayGet( theNamesArray, index+1 ) ;
    if ( CosDictKnown ( theDestCosObj, theDASAtom ) == true )
    {
        CosObj theDestArray = CosDictGet( theDestCosObj, theDASAtom ) ;
        if ( CosObjEqual ( theDestArray, theNullCosObj ) )
            ASRaise( CosError( cosErrExpectedArray ) ) ;

        outPageCosObj = CosArrayGet( theDestArray, 0 ) ;
        thePageNumber = PDPageNumFromCosObj( outPageCosObj ) ;
        break ;
    }   // end if
}   // end for
```

NameTree Sample

SNameTree.cpp

Page Contents Sample

SContent.cpp

The PD Layer

The PD Layer

- **PDBookmark**
- **PDAction**
- **PDAnnot**
- **PDDoc**
- **PDPage**

Acquire, Release & IsValid

- **If an item is Acquired you must remember to Release**
- **A data type used within the Acrobat SDK can not simply be tested with NULL. To check if an object is valid use the IsValid function.**

Appligent

PDBookmarks

- **Bookmarks or Outlines is a user navigation construct which contains the following:**
  - Title
  - Destination or Action
  - Link to Next Bookmark
  - Link to Previous Bookmark
  - Link to Parent Bookmark

Bookmark Sample

getbookmarks.cpp

PDActions

Getting Things Done

- **GoTo**
  - Go to a destination in the current document
- **GoToR**
  - (“Go-to remote”) Go to a destination in another document
- **Launch**
  - Launch an application, usually to open a file
- **Thread**
  - Begin reading an article thread
- **URI**
  - Resolve a uniform resource identifier

PDActions Continued

- **Sound**
  - (PDF 1.2) Play a sound
- **Movie**
  - (PDF 1.2) Play a movie
- **Hide**
  - (PDF 1.2) Set an annotation's Hidden flag
- **Named**
  - (PDF 1.2) Execute an action predefined by the viewer application

Appligent

PDActions Continued

- **SubmitForm**
  - (PDF 1.2) Send data to a uniform resource locator
- **ResetForm**
  - (PDF 1.2) Set fields to their default values
- **ImportData**
  - (PDF 1.2) Import field values from a file
- **JavaScript**
  - (PDF 1.3) Execute a JavaScript script

Appligent

Action Sample

APAction.cpp

PDDoc

PDDocOpen

```
PDDoc PDDocOpen (ASPathName fileName, ASFileSys fileSys,
PDAuthProc authProc, ASBool doRepair);

thePDDoc = PDDocOpen( theASPathName, NULL, NULL, false ) ;
if ( thePDDoc == (PDDoc)NULL )
{
fprintf( stderr, “Unable to open PDF file - %s\n”, inFileName ) ;
ASRaise( ASFileError( fileErrOpenFailed ) ) ;
}

-- Write some code here

PDDocClose( thePDDoc ) ;
```

PDPage

PDDocAcquirePage

```
for ( index = inFirstPage-1 ; index < inLastPage ; index++ )
{
    thePage = ::PDDocAcquirePage( thePDDoc, index ) ;

    for( ASInt32 streamIndex = 0 ;
        streamIndex < SContent::GetContentsStreamCount( thePage ) ;
        streamIndex++ )
        {
            CosObj theContentStreamCosObj = SContent::GetContentsStream(
            thePage, streamIndex ) ;
            ASStm theASStream = CosStreamOpenStm( theContentStreamCosObj,
            cosOpenFiltered ) ;

        }   // end for contents

    ::PDPageRelease( thePage ) ;

}   // end for pages
```

PDPage Example

contentdump.cpp

PDAnnot

APAnnot.cpp