```
/**************************************************************************
 *
 * File:            PDFVIEWERAPI_c.h
 *
 * Description:     The PDF Viewer API native C interface.
 *
 * Author:          Dr. Hans Bärfuss, PDF Tools AG
 *
 * Copyright:       Copyright (C) 2013 PDF Tools AG, Switzerland
 *                  All rights reserved.
 *
 **************************************************************************/

#ifndef _PDFVIEWERAPI_INCLUDED
#define _PDFVIEWERAPI_INCLUDED

#include "pdfcdecl.h"


#ifndef PDFVIEWERAPI
#   if defined(WIN32) && !defined(_ASSEMBLY)
#       ifdef PDFVIEWERAPI_EXPORTS
#           define PDFVIEWERAPI __declspec(dllexport)
#       else
#           define PDFVIEWERAPI __declspec(dllimport)
#       endif
#   else
#       define PDFVIEWERAPI
#   endif
#endif

#ifndef PDFVIEWERCALL
#   if defined(WIN32)
#       define PDFVIEWERCALL __stdcall
#   else
#       define PDFVIEWERCALL
#   endif
#endif

#ifdef _UNICODE
#define PdfViewerSetLicenseKey      PdfViewerSetLicenseKeyW
#define PdfViewerGetProductVersion  PdfViewerGetProductVersionW
#define PdfViewerCreateObject       PdfViewerCreateObjectW
#else
#define PdfViewerSetLicenseKey      PdfViewerSetLicenseKeyA
#define PdfViewerGetProductVersion  PdfViewerGetProductVersionA
#define PdfViewerCreateObject       PdfViewerCreateObjectA
#endif

#ifndef BOOL
#define BOOL int
#define TRUE 1
#define FALSE 0
#endif

#ifdef __cplusplus
extern "C" {
#endif

typedef struct CDocumentHandle CDocumentHandle;
typedef struct CPageHandle CPageHandle;
typedef struct CTextHandle CTextHandle;
typedef struct CAnnotationHandle CAnnotationHandle;


typedef enum TLayoutMode
{
    eLayoutModeNone = 0,
    eLayoutModeSinglePage = 1,
    eLayoutModeOneColumn = 2,
    eLayoutModeTwoColumnLeft = 3,
    eLayoutModeTwoColumnRight = 4,
```

```
74          eLayoutModeTwoPageLeft = 5,
75          eLayoutModeTwoPageRight = 6
76      } TLayoutMode;
77      typedef enum TDestination
78      {
79          eDestinationInvalid = 0,
80          eDestinationFit = 1,
81          eDestinationFitH = 2,
82          eDestinationFitV = 3,
83          eDestinationFitR = 4,
84          eDestinationFitB = 5,
85          eDestinationFitBH = 6,
86          eDestinationFitBV = 7,
87          eDestinationXYZ = 8
88      } TDestination;
89      typedef enum TViewerError
90      {
91          eLicenseError = 0,
92          ePasswordError = 1,
93          eFileNotFoundError = 2,
94          eUnknownError = 3,
95          eIllegalArgumentError = 4,
96          eOutOfMemoryError = 5,
97          eFileCorruptError = 6,
98          eUnsupportedFeatureError = 7,
99          eNoFileAccess = 8
100     } TViewerError;
101
102     typedef enum TAnnotationType
103     {
104         eAnntationUnknown = 0,
105         eAnnotationText = 1,
106         eAnnotationLink = 2,
107         eAnnotationFreeText = 3,
108         eAnnotationHighlight = 9,
109         eAnnotationInk = 15,
110         eAnnotationPopup = 16,
111         eAnnotationWidet = 20
112     } TAnnotationType;
113
114     typedef struct TPdfViewerPoint
115     {
116         double m_x;                 /* Horizontal coordinate */
117         double m_y;                 /* Vertical coordinate */
118     } TPdfViewerPoint;
119     typedef struct TPdfViewerOutlineItem
120     {
121         int m_iId;                  /* Unique outline identifier. */
122         int m_iLevel;               /* The outline level relative to the other outlines
            of the array. */
123         BOOL m_bDescendants;        /* Whether this outline has descendants. */
124         WCHAR* m_szTitle;     /* The title of the outline. */
125         int m_iPage;                /* Destination page. */
126         TPdfViewerPoint m_pt;       /* Destination coordinates on page (raw pdf
            coordinates). */
127         double m_dZoom;             /* Destination Zoom (0..100%). */
128         double m_dLeft;             /* left value of the destination */
129         double m_dTop;              /* top value of the destination */
130         double m_dRight;            /* right value of the destination */
131         double m_dBottom;           /* bottom value of the destination */
132         char* m_szDestType;   /* Destination type */
133     } TPdfViewerOutlineItem;
134
135     typedef struct TPdfTextFragment
136     {
137         WCHAR* m_szTextW;
138         int m_iTextLength;
139         double m_dX;
140         double m_dY;//from bottom of page
141         double m_dWidth;
142         double m_dHeight;
143         double* m_pdGlyphPosition;
144         int m_nGlyphPositionSize;
```

```
145 } TPdfTextFragment;
146
147 typedef struct TPdfRectangle
148 {
149     double m_x1;
150     double m_y1;
151     double m_x2;
152     double m_y2;
153 } TPdfRectangle;
154
155 typedef struct TPopupAnnotation
156 {
157     CAnnotationHandle*      m_pPopupAnnotationHandle; /* Annotation handle */
158     double                  m_aRect[4];               /* Rectangle of the popup */
159     BOOL                    m_bIsOpen;                /* is popup opened */
160     const char*             m_szSubtype;              /* Subtype - Must be 'Popup'
    for a popup annotation*/
161 } TPopupAnnotation;
162
163 typedef struct TPdfAnnotation
164 {
165     CAnnotationHandle*      m_pAnnotationHandle; /* Annotation handle*/
166     int                     m_iPage;             /* Page of the annotation */
167     const char*             m_szSubtype;         /* Annotation subtype */
168     int                     m_nColors;           /* No. of color components */
169     double*                 m_aColor;            /* Color of the annotation */
170     int                     m_iFlags;            /* Flags for the annotation*/
171     double                  m_aRect[4];          /* Rectangle of the annotation */
172     double*                 m_aQuadPoints;       /* Quadpoints for link and
    highlight */
173     int                     m_nQuadPoints;       /* Number of quad points*/
174     WCHAR*                  m_szContents;        /* Content of the annotation */
175     BOOL                    m_bIsLink;           /* Is link annotation*/
176     const char*             m_szActionType;      /* action type of destination */
177     BOOL                    m_bHasURI;           /* does destination have a URI */
178     WCHAR*                  m_szUri;             /* URI of the action (if m_bHasUri
    is true) */
179     int                     m_iDestType;         /* Type of destination */
180     BOOL                    m_aHasDestVal[5];    /* Helper array */
181     double                  m_pDestArray[5];     /* Destination of the link */
182     int                     m_iDestPage;         /* Page of destination */
183     BOOL                    m_bIsMarkup;         /* is it a markup annotation */
184     const char*             m_szTextLabel;       /* Label of the annotation (often
    the author) */
185     BOOL                    m_bHasPopup;         /* is there a popup annotation */
186     TPopupAnnotation*       m_pPopupAnnot;       /* Simplified popup annotation
    object */
187 } TPdfAnnotation;
188
189
190 /* Object creation / desctruction */
191 PDFVIEWERAPI CDocumentHandle*      PDFVIEWERCALL PdfViewerCreateObjectW(const WCHAR*
    szFileName, const void* pData, size_t nSize, const WCHAR* szPassword);
192 PDFVIEWERAPI CDocumentHandle*      PDFVIEWERCALL PdfViewerCreateObjectA(const char*
    szFileName, const void* pData, size_t nSize, const char* szPassword);
193 PDFVIEWERAPI void                  PDFVIEWERCALL
    PdfViewerDestroyObject(CDocumentHandle* handle);
194
195 /*initializing*/
196 PDFVIEWERAPI void                  PDFVIEWERCALL PdfViewerInitialize();
197 PDFVIEWERAPI void                  PDFVIEWERCALL PdfViewerUnInitialize();
198
199 /* Licensing */
200 PDFVIEWERAPI BOOL                  PDFVIEWERCALL PdfViewerSetLicenseKeyW(const
    WCHAR* szLicenseKey);
201 PDFVIEWERAPI BOOL                  PDFVIEWERCALL PdfViewerSetLicenseKeyA(const char*
    szLicenseKey);
202 PDFVIEWERAPI BOOL                  PDFVIEWERCALL PdfViewerGetLicenseIsValid();
203
204 PDFVIEWERAPI const char*           PDFVIEWERCALL PdfViewerGetProductVersionA();
205 PDFVIEWERAPI const WCHAR*          PDFVIEWERCALL PdfViewerGetProductVersionW();
206
207 /* Save document */
```

```
208 PDFVIEWERAPI BOOL                  PDFVIEWERCALL PdfViewerSaveAs(const long handle,
    const WCHAR* szPath);
209
210 /* Outlines */
211 PDFVIEWERAPI BOOL                  PDFVIEWERCALL
    PdfViewerGetOutlineItems(CDocumentHandle* pHandle, int iParentID,
    TPdfViewerOutlineItem** pItems, int* pCount);
212 PDFVIEWERAPI BOOL                  PDFVIEWERCALL
    PdfViewerDisposeOutlineItems(TPdfViewerOutlineItem* pItems, int nCount);
213
214 /* Document properties */
215 PDFVIEWERAPI int                   PDFVIEWERCALL
    PdfViewerGetPageCount(CDocumentHandle* pHandle);
216 PDFVIEWERAPI CPageHandle*          PDFVIEWERCALL
    PdfViewerGetPageRect(CDocumentHandle* pHandle, int iPage, double* width, double*
    height);
217
218 PDFVIEWERAPI int                   PDFVIEWERCALL
    PdfViewerGetRotation(CDocumentHandle* pHandle, int iPage);
219 PDFVIEWERAPI BOOL                  PDFVIEWERCALL PdfViewerDraw(CDocumentHandle*
    pHandle, int iPage,
220     int iWidth, int iHeight, void* pBuffer, int iRotation,
221     int iTargetX, int iTargetY, int iTargetWidth, int iTargetHeight,
222     double dSourceX, double dSourceY, double dSourceWidth, double dSourceHeight);
223 PDFVIEWERAPI TLayoutMode           PDFVIEWERCALL
    PdfViewerGetPageLayout(CDocumentHandle* pHandle);
224 PDFVIEWERAPI TDestination          PDFVIEWERCALL
    PdfViewerGetOpenActionDestination(CDocumentHandle* pHandle, int* iPageNo, double*
    dLeft, double* dTop, double* dRight, double* dBottom, double* dZoom);
225 PDFVIEWERAPI TDestination          PDFVIEWERCALL parseDestinationType(const char*
    szDestinationType);
226 PDFVIEWERAPI BOOL                  PDFVIEWERCALL
    PdfViewerExtractTextFragments(CDocumentHandle* pHandle, int iPage,
    TPdfTextFragment** pExtractedTexts, int* pCount);
227 PDFVIEWERAPI void                  PDFVIEWERCALL
    PdfViewerDisposeTextFragments(TPdfTextFragment* pExtractedTexts, int nCount);
228
229 /* Annotations */
230 PDFVIEWERAPI BOOL                  PDFVIEWERCALL
    PdfViewerGetAnnotationsOnPage(CDocumentHandle* pHandle, int iPage, TPdfAnnotation**
    pAnnotations, int* pCount);
231 PDFVIEWERAPI void                  PDFVIEWERCALL
    PdfViewerDisposeAnnotationItems(TPdfAnnotation* pAnnotations, int iCount);
232 PDFVIEWERAPI void                  PDFVIEWERCALL
    PdfViewerDisposeSingleAnnotation(TPdfAnnotation* pAnnotation);
233 PDFVIEWERAPI int                   PDFVIEWERCALL
    PdfViewerUpdateAnnotation(CDocumentHandle* pDocument, CAnnotationHandle* annot, int
    iPage, double rect[], const WCHAR* content, const WCHAR* label);
234 PDFVIEWERAPI TPdfAnnotation*       PDFVIEWERCALL
    PdfViewerCreateAnnotation(CDocumentHandle* pHandle, TAnnotationType eType, int
    iPage, double r[], int iLen);
235 PDFVIEWERAPI void                  PDFVIEWERCALL
    PdfViewerDeleteAnnotation(CAnnotationHandle* annot);
236
237 /* Error Handling */
238 PDFVIEWERAPI TViewerError          PDFVIEWERCALL PdfViewerGetLastError();
239 PDFVIEWERAPI size_t                PDFVIEWERCALL PdfViewerGetLastErrorMessageA(char*
    pBuffer, size_t nBufferSize);
240 PDFVIEWERAPI size_t                PDFVIEWERCALL
    PdfViewerGetLastErrorMessageW(WCHAR* pBuffer, size_t nBufferSize);
241
242 #ifdef __cplusplus
243 }
244 #endif
245
246 #endif /* _PDFVIEWERAPI_INCLUDED */
247
```