

# Radaee PDF Reader OCX Reference

## Deployment

You need copy 2 items to your product's dictionary.  
"RadaeePDF.dll" and "cmaps" dictionary.

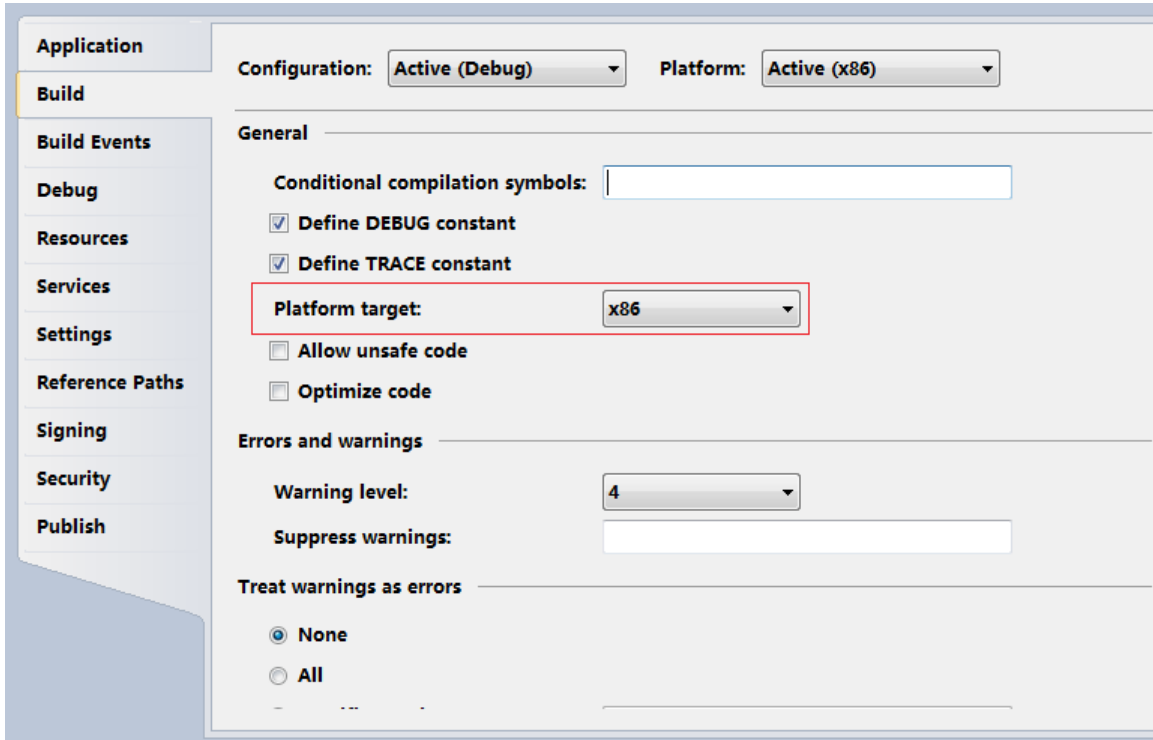
 <b>cmaps</b>	2010/8/25 9:01
 <b>RadaeePDF.dll</b>	2010/7/23 11:13

They must be in the same dictionary.  
Then you can run "regsvr32.exe RadaeePDF.dll" to finish the deployment.

## Development

In VC++ following codes are needed:  
`#import "your path\RadaeePDF.dll" raw_interfaces_only`

In .NET development x86 platform target is need. Just like below:



When developers build an application 7 interfaces may be called, 2 are COM objects, others are ActiveX Controls. List as below:

- PDFBookmark
- PDFDocument
- PDFThumbs
- PDFOutlines
- PDFViewer
- PDFBook
- PDFRoll

To an application, developer need create a PDFDocument object for opening PDF file, then attach the document object to the active control to view pdf file. For ActiveX Control PDFViewer, PDFRoll and PDFBook, only one of these control can attach the document in same time, others should detach the document object before a control attach the document.

## PDFBookmark

COM object for pdf outline item.

## Methods:

### **PDFBookmark get\_child()**

Get the first child item.

Return NULL means no children.

### **PDFBookmark get\_next()**

Get the next sibling item.

Return null means no more.

## Attributes:

### **string label;**

item's label.

### **int pageno();**

link page, 0 means no link.

### **DOUBLE top();**

y coord of link position(72 DPI).

## PDFDocument:

COM object for pdf document.

## Method:

### **bool open(string filename, string password);**

Open pdf file.

filename:pdf file pathname.

password:password.

return value is bool. If false you can get error code by attribute "error\_code"

### **void close();**

Close pdf file.

**PDFBookmark get\_first\_bookmark();**

Get the first bookmark(defined as outline in PDF reference) item of this document.

You need free the return value in C++ application by call Release() function.

If return null means no outlines in this document.

**HANDLE save\_as(string filename, file\_type type);**

Save pdf file as other type file.the function is asynchronous.

The event "OnSaving" notify saving events.

filename: destination file name.

type:

```
enum file_type
{
    [helpstring("txt file")] file_txt = 0,
};
```

return value:a handle of this session.

Use cancel\_session to cancel the session.

**HANDLE print(string title, int page\_start, int page\_end, DOUBLE page\_ratio, int copies, enum prn\_align align, HANDLE printer);**

Print document in asynchronous way.

title:print job title.

page\_start:start page to print.

page\_end:end page to print.

Page\_ratio:page scale for print.

copies:copies.

align:

```
enum prn_align
{
    [helpstring("align pages left")] align_left = 0,
    [helpstring("align pages center")] align_center = 1,
    [helpstring("align pages right")] align_right = 2,
};
```

printer:use "OpenPrinter" or "AddPrinter" to retrieve handle

return value: a handle of this session.

Use cancel\_session to cancel the session.

**HANDLE print2(string title, int page\_start, int page\_end, DOUBLE page\_ratio, int copies, enum**

**prn\_align align, string printer, HANDLE hDevMode )**

Print document in asynchronous way.

This method is same as print, except the paramaters.

printer:printer name.

hDevMode:handle of devmode. Int C# it retrived by PrinterSettings.GetHdevmode().

**HANDLE render\_page(HANDLE hdc, int page\_no, DOUBLE resx, DOUBLE resy, int offx, int offy, bool print\_mode)**

Render the page to given device.

hdc:GDI Context handle(HDC).

page\_no:page number.

resx: horizontal resolution.(use this paramater to scale)

resy:vertical resolution.(use this paramater to scale)

offx:offset x of view port.

offy:offset y of view port.

print\_mode:is the device printer?

return value: a handle of this session.

Use cancel\_session to cancel the session.

**void cancel\_session(HANDLE session);**

cancel the session setuped by save\_as/print/print2/render\_page.

You should dispose all resources when the event fired.

**Attributes:****int page\_count;**

Total page count in pdf file.

**pdf\_error error\_code;(read only)**

error code for open method.

Defined as below

enum pdf\_error

{

[helpstring("success")] err\_success = 0,

[helpstring("open file failed!")] err\_openfail = 1,

[helpstring("a password or key is wanted!")] err\_encrypted = 2,

};

**DOUBLE page\_width(int pageno);(read only)**

Page width of the given page.The value is in 72 DPI.

Pageno is from 1 to page count

**DOUBLE page\_height(int pageno);(read only)**

Page height of the given page.The value is in 72 DPI

Pageno is from 1 to page count

**string meta\_data(string name);(read only)**

get meta data in pdf file. predefine key name is:

#ver

Title

Producer

Creator

Author

Keywords

## Events:

**void OnPrinting(int page\_no, int page\_count)**

fired when printing called by method print or print2.

pageno:number of pages printed.

page\_count:total pages to print.

if pageno = -1 and page\_count = 0 means cancelled.

if pageno = -1 and page\_count != 0 means succeeded.

**void OnSaving(int page\_no, int page\_count)**

fired when saving called by method asve\_as.

pageno:number of pages saved.

page\_count:total pages to save. tt is equal to page\_count.

if pageno = -1 and page\_count = 0 means cancelled.

if pageno = -1 and page\_count != 0 means succeeded.

**void OnRenderEnd(int page\_no, int flag)**

fired when render page end or cancelled.

pageno:page to render.

flag:0 means cancelled, 1 means succeeded.

## PDFThumbs

Controller for displaying pdf thumb nails.



### Methods:

```
bool attach_doc(PDFDocument doc)
```

```
void detach_doc()
```

```
void goto_page(int page_no);
```

goto page, page\_no is from 1 to page\_count.

### Attributes:

```
COLOR BackColor;
```

Color of background.

**COLOR PageBackColor;**

Color of page's background.

## Events:

**void OnClickPage(int page\_no)**

fired when a page is clicked and selected.

# PDFBMTree

ActiveX Control of outline tree.

## Methods:

**bool attach\_doc(PDFDocument doc);**

**void detach\_doc();**

## Attributes:

**COLOR BackColor;**

## Events:

**OnItemSelected(string label, string page\_no, DOUBLE top)**

Fired when an item is selected.

# PDFViewer

ActiveX Control of viewer window.



## Methods:

```
bool attach_doc(PDFDocument doc);
void detach_doc();
void goto_first_page();
    goto the first page.
void goto_prev_page();
    goto previous page.
void goto_next_page();
    goto next page.
void goto_last_page();
    goto the last page.
void scroll_line_down();
    scroll line down, just link scrollbar.
void scroll_line_up();
    scroll line up, just link scrollbar.
void scroll_page_up();
    scroll page up, when pageup key is pressed.
void scroll_page_down();
    scroll page up, when pagedown key is pressed.
void mouse_wheel(int delta);
```

mouse wheel.

**HANDLE find(string sfind, int start\_page, int start\_obj, bool lookup, bool match\_case, bool match\_whole);**

Find the text.

sFind:text to search.

start\_page:start page to serach.

Start\_obj:start object to find.

lookup:search for up.

Return the session handle, use cancel\_find to cancel it.

**void cancel\_find(HANDLE session);**

cancel finding.

**void goto\_pos(int page\_no, DOUBLE top);**

goto page and position.

page\_no:page number to located.

top:y coord int page.

**bool save\_snapshot\_to\_file(string filename);**

save snapshot box as a bmp file.

**void save\_snapshot\_to\_clipboard();**

save snapshot box as iamge to clipboard.

**int point\_to\_page(int x, int y);**

get the page no from point.

The point is based on window's client area.

**bool save\_page(int LONG page, string bmp\_filename);**

save whole page as a bmp file. the page must be displayed in client area.

If you want save pages with no UI. Please call render\_page method in PDFDocument.

## Attributes:

**COLOR BackColor;**

Window's background.

**COLOR PageBackColor;**

Page's background.

**pdf\_layout layout;**

```
enum pdf_layout
{
    [helpstring("single page view")] layout_single = 0,
    [helpstring("continuous page view")] layout_continuous = 1,
    [helpstring("2 in 1 page view")] layout_folio = 2,
    [helpstring("2 in 1 continuous page view")] layout_foliocontinuous = 3,
};
```

**pdf\_fit\_mode fit\_mode;**

```
enum pdf_fit_mode
{
    [helpstring("no fit")] fit_none = 0,
    [helpstring("fit window width")] fit_width = 1,
    [helpstring("fit window height")] fit_height = 2,
    [helpstring("fit whole page")] fit_page = 3,
};
```

**DOUBLE ratio;****pdf\_tool tool**

```
enum pdf_tool
{
    [helpstring("drag page")] tool_drag = 0,
    [helpstring("text selection tool")] tool_sel = 1,
    [helpstring("snapshot to clipboard or image file")] tool_snapshot = 2,
};
```

**int current\_page**

current page number.you can set this value to goto the page.

**string selected\_string;(read only)**

get the text current selected.

## Events:

**void OnFinding(int page\_no, int page\_count);**  
fired while finding.

**void OnRatioChanged(DOUBLE ratio);**  
fired when the view ratio changed.

**void OnPageNoChanged(int cur\_page);**  
fired when page number changed.

**void OnSnapshot(int x, int y);**  
fired when left button is up in snapshot mode.

## PDFBook

ActiveX Control for another view style, just like below:



## Methods:

```
bool attach_doc(PDFDocument doc);  
void detach_doc();  
void roll_papers_forward( int papers );  
void roll_papers_backward( int papers );
```

## Attributes:

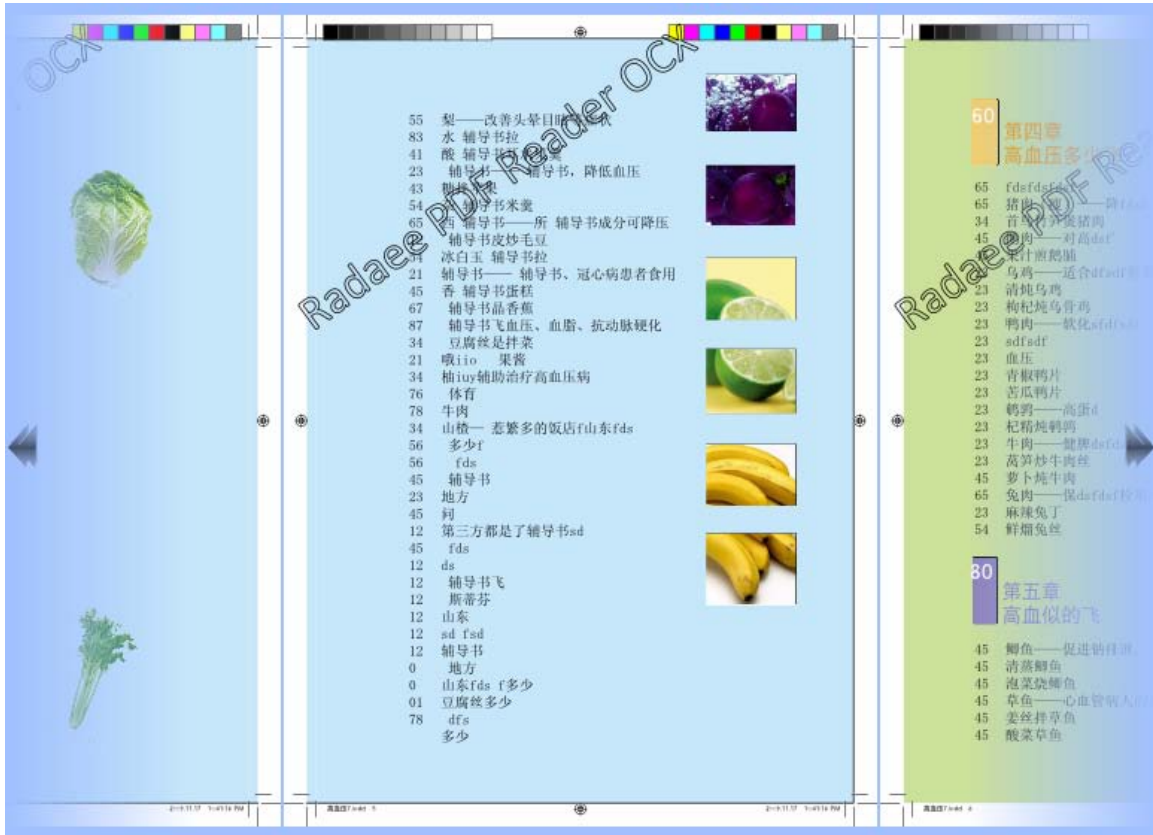
```
COLOR BackColor;  
    Window's background.  
COLOR PageBackColor;  
    Page's background.
```

## Events:

```
void OnPageNoChanged(int cur_page);  
    fired when page number changed.
```

# PDFRoll

ActiveX Control for another view style, just like below:



## Methods:

```
bool attach_doc(PDFDocument doc);
```

```
void detach_doc();
```

```
void roll_pages( int papers );
```

negative number roll pages backward, positive number roll pages forward.

## Attributes:

```
COLOR BackColor;
```

Window's background.

```
COLOR PageBackColor;
```

Page's background.

```
Int max_delta;
```

Max delta for rolling page.

```
Int cell_width;
```

Width for one page.

**Int edge\_width;**

edge width for transparent area on left and right side.

## **Events:**

**void OnPageNoChanged(int cur\_page);**

fired when page number changed.