Foxit PDF Conversion SDK for C API 2.1 Upgrade Warnings

This document contains a list of items that developers looking to upgrade to Foxit PDF Conversion SDK for C API 2.1 from previous version should pay attention to. Changes to code may be required in order for your projects to compile using the new version.

Upgrading to version 2.1 from version 2.0

If you are upgrading from version 2.0 to version 2.1, please take note of the changes that occurred between version 2.0 and version 2.1.

- In **FSPDF2OfficeSettingData** struct, the member "metrics_data_folder_path" has been deprecated. Instead, metrics data will be included in the dynamic library.

Upgrading to version 2.0 from version 1.5

There are two minor upgrade warnings in this release when moving from version 1.5 to version 2.0:

- In **FSPDF2WordSettingData** struct, add two new members as below:

  ```
  FS_BOOL enable_generate_headers_and_footers;
  FS_BOOL enable_generate_footnotes_and_endnotes;
  ```

- In **FSPDF2OfficeSettingData** struct, add two new members as below:

  ```
  FSPDF2PowerPointSettingData power_point_setting_data;
  FSPDF2ExcelSettingData excel_setting_data;
  ```

Upgrading to version 1.5 from version 1.4

There is only one minor upgrade warning in this release when moving from version 1.4 to version 1.5:

- In **FSPDF2OfficeSettingData** struct, add three new members as below:

  ```
  FS_RANGE_HANDLE page_range;
  FS_BOOL include_pdf_comments;
  FSPDF2WordSettingData word_setting_data;
  ```

Upgrading to version 1.4 from version 1.3

If you are upgrading from version 1.3 to version 1.4, please take note of the changes that occurred between version 1.3 and version 1.4.

Starting from version 1.4, the APIs of Foxit PDF Conversion SDK C language has been modified to support pure C language. The following are the changes to the pure C interface:

1. Only use C's standard libraries and headers, remove C++ Specific Features, Classes and Objects.

2. Change interface parameters from references to pointers.

3. Change the file extensions of simple demo from **.cpp** to **.c** to explicitly indicate that these files now exclusively contain C code.

4. Design the pure C interface using function pointers or similar constructs to pass callback functions, rather than relying on C++ Class with virtual functions.

The following are two examples of using the Foxit PDF Conversion SDK pure C interface:

- **Convert PDF to Word with Path**

```c
wchar_t src_pdf_path[MAX_FILE_PATH];
wchar_t saved_word_file_path[MAX_FILE_PATH];
FSPDF2OfficeSettingData setting_data;
const wchar_t* metrics_data_folder_path = L"../../../res/metrics_data";
setting_data.metrics_data_folder_path.str = (wchar_t*)metrics_data_folder_path;
setting_data.metrics_data_folder_path.len = wcslen(metrics_data_folder_path);
setting_data.enable_ml_recognition = FALSE;
swprintf(src_pdf_path, MAX_FILE_PATH - 1, L"word.pdf");
swprintf(saved_word_file_path, MAX_FILE_PATH - 1, L"pdf2word_result.docx");
FS_PROGRESSIVE_HANDLE progressive_handle = NULL;
FSErrorCode error_code = FSDK_PDF2Office_StartConvertToWord(src_pdf_path, NULL,
saved_word_file_path, setting_data, NULL, &progressive_handle);
```

- **Using FSConvertCallback**

```c
static FS_BOOL gNeedToPause(void* user_data) {
   return TRUE;
}

static void gProgressNotify(void* user_data, int converted_count, int total_count) {
}
FSConvertCallback* callback = (FSConvertCallback*)malloc(sizeof(FSConvertCallback));
callback->user_data = callback;
callback->NeedToPause = gNeedToPause;
callback->ProgressNotify = gProgressNotify;
```