Debenu Quick PDF Library Cross-Platform Beta
Beta 1 - Release Notes
9 December 2014

1. General

1.1 Introduction

Debenu Quick PDF Library Cross-Platform is a port of the popular Debenu Quick PDF Library product from Delphi to C++. This allows the library to run on new platforms. Currently the cross platform library runs on Windows (32/64-bit DLL and 32/64-bit ActiveX), Mac OS X (32/64-bit Dylib) and iOS (arm6, arm7, arm7s and arm64).

The original Debenu Quick PDF Library used GDI+ with an Agg-based renderer (DPLR) offered as a separate add-on. The new library incorporates the Agg-based renderer into the library.

The original library was available as a 32-bit Mac OS X Dylib with 16-bit characters for Unicode strings and a dependency on a separate Delphi runtime Dylib. The new library is available as both a 32-bit Dylib as well as a new 64-bit Dylib, uses 32-bit characters for Unicode strings and has no dependencies on 3rd party Dylibs (there is a dependency on the system provided libxml2.dylib).

1.2 Programming interfaces

The ActiveX edition exposes the API via a type library and COM class interface. This makes it very easy to use from many different programming systems with the system provided COM functionality and runtime method invocation. Separate import classes are provided for C++, C# and VB.NET to make it even easier to access the ActiveX edition.

For the DLL, Dylib and iOS editions the API is provided as a C-based flat API. This flat API is somewhat complicated to work with however C++, C# and Objective-C import classes are provided making interfacing with the API much easier. When using these import classes functions like **CreateLibrary** and **CreateBuffer** are not required as this functionality is taken care of by the import class.

All strings sent to and received from the library using the import classes, including strings containing null bytes, are automatically buffered as required and presented as native string types to the programmer (NSString, std::wstring, std:string, BSTR etc).

1.3 Flat API technical details

Most users will get along fine with the provided import classes, however If these classes are not sufficient it is possible to interface directly with the C-based flat API.

A call to the **CreateLibrary** function (exported as “DPLCreateLibrary” on Windows and “DPLCreateLibrary” on Mac/iOS) is required to create an internal instance of the library. This function will return an InstanceID value that should be passed as the first parameter to subsequent calls to other library functions.

With the flat API strings are sent to the library as character pointers. For strings that do not contain null bytes the library will look for a null terminator to determine the length of the string. Strings returned from the library are null terminated.
For strings sent to the library that contain null bytes the **CreateBuffer**, **AddToBuffer** and **ReleaseBuffer** functions can be used to send data to the library with an explicit length. A pointer to the buffer can then be used and the library will be able to determine the correct length of the string.

When binary data is returned from the library the **StringResultLength** (for Unicode strings) and **AnsiStringResultLength** (for 8-bit binary data) functions can be used to determine the length of the returned data from the most recently called library function.

2. iOS Static Library Edition

2.1 Files

The iOS Static Library edition is provided as a number of different binaries:

`DebenuPDFLibraryCPiOS1113-i386.a`	32-bit iOS simulator
`DebenuPDFLibraryCPiOS1113-x86_64.a`	64-bit iOS simulator (64-bit iPhone 5s and later)
`DebenuPDFLibraryCPiOS1113-arm6.a`	arm6 for 32-bit iOS devices (iPhone 3 and earlier)
`DebenuPDFLibraryCPiOS1113-arm7.a`	arm7 for 32-bit iOS devices (iPhone 4 and later)
`DebenuPDFLibraryCPiOS1113-arm7s.a`	arm7s for 32-bit iOS devices (iPhone 4s and later)
`DebenuPDFLibraryCPiOS1113-arm64.a`	arm64 for 64-bit iOS devices (iPhone 5s and later)

Different developers will require different combinations of these libraries during development depending on the particular devices they are targeting.

Various combinations of the supplied binaries can be combined into a single universal library using the **lipo** command. For example, the arm7 and arm64 device binaries with the 32-bit and 64-bit simulator binaries can combined with the following command:

```
$ lipo -create DebenuPDFLibraryCPiOS1113-i386.a DebenuPDFLibraryCPiOS1113-x86_64.a
DebenuPDFLibraryCPiOS1113-armv7.a DebenuPDFLibraryCPiOS1113-arm64.a -output
libDebenuPDFLibraryCPiOS1113.a
```

When the project is being built Xcode will extract the appropriate slice from the universal binary depending on the target.

2.2 Dependencies

All Xcode projects using the iOS Static Library must have the **libxml2.dylib** library added to the "Link binary with libraries" list in the target's "Build Phases" project settings.

2.3 Programming interfaces

The Debenu PDF Library API can be accessed from the static library in two different ways:

- An Objective-C class interface (.h file provided)
- A C-based flat API wrapped by a C++ interface class (.h and .cpp files provided)

2.4 Linker settings

When using the Objective-C class interface the **-lc++** option must be added to the "Other linker flags" option in the project settings.

2.5 Objective-C example

1. Create a new project in Xcode
2. Add the **libxml2.dylib** library to the "Link binary with libraries" list in the target's Build Phases project settings
3. Add the **DebenuPDFLibraryCPiOSObjC1113.h** header file to the project
4. Add the static library (the .a file created with lipo in section 2.1) to the project
5. Add the **-lc++** option to the “Other linker flags” option in the project settings
6. Add a button and link it to an IBAction called buttonClick, then use the following code:

```
#import "ViewController.h"
#import "DebenuPDFLibraryCPiOSObjC1113.h"

@interface ViewController ()

@end

@implementation ViewController

- (NSString*)getDocumentsFolder {
    NSArray *paths = NSSearchPathForDirectoriesInDomains(
    NSDocumentDirectory, NSUserDomainMask, YES);
    NSString *basePath = ([paths count] > 0) ? [paths objectAtIndex:0] : nil;
    return basePath;
}

- (IBAction)buttonClick {
    DebenuPDFLibraryCPiOSObjC1113 *DPL = [DebenuPDFLibraryCPiOSObjC1113 new];
    if ([DPL UnlockKey:@"...license key here..."] == 1)
    {
        NSLog(@"UnlockKey success: %@", [DPL LicenseInfo]);
        [DPL DrawText:100:700:@"Hello world"];
        NSString* outputFile = [[self getDocumentsFolder]
            stringByAppendingString:@"Hello.pdf"];
        if ([DPL SaveToFile:outputFile] == 1)
        {
            NSLog(@"File saved successfully");
        }
    }
}

@end
```

2.6 C++ example

1. Create a new project in Xcode
2. Add the **libxml2.dylib** library to the “Link binary with libraries” list in the target’s Build Phases project settings
3. Add the **DebenuPDFLibraryCPiOSCPP1113.h** and **DebenuPDFLibraryCPiOSCPP1113.cpp** class files to the project
4. Add the static library (.a file created with lipo) to the project
5. Add a button and link it to an IBAction called buttonClick

6. Change the extension of the ViewController.m file to .mm

```
#import "ViewController.h"
#include "DebenuPDFLibraryCPiOSCPP1113.h"

void runCPlusPlus()
{
    DebenuPDFLibraryCPiOSCPP1113 DPL;
    if (DPL.UnlockKey(L"...license key here...") == 1)
    {
        DPL.DrawText(100, 700, L"Hello world");
        DPL.SaveToFile(L"...folder here.../Hello.pdf");
    }
}

@interface ViewController ()

@end

@implementation ViewController

-(IBAction)buttonClick {
    runCPlusPlus();
}

@end
```

3. OS X Dylib Edition

3.1 Files

The Dylib edition is provided as two binaries:

`libDebenuPDFLibraryCPDylib1113.dylib`	32-bit OS X Dylib
`libDebenuPDFLibrary64CPDylib1113.dylib`	64-bit OS X Dylib

3.2 Programming interface

Two programming interfaces are provided for the Dylib edition, an Objective-C class:

`DebenuPDFLibraryCPDylibObjC.h`	Objective-C interface class header
`DebenuPDFLibraryCPDylibObjC.m`	Objective-C interface class implementation

`And a C++ class:`

`DebenuPDFLibraryCPDylibCPP.h`	C++ interface class header
`DebenuPDFLibraryCPDylibCPP.cpp`	C++ interface class implementation

Both the Objective-C and the C++ interface classes require a file name to be passed to the constructor/initialization function. This is the full path to either the 32-bit or the 64-bit Dylib file. It's possible to use an #ifdef command to select the appropriate file automatically.

3.3 Flat C API

If the Objective-C or C++ class interfaces are not sufficient the API can be accessed via the C-based flat API. A header file for this purpose is available from Debenu on request.

3.4 Objective-C example

1. Create a new project in Xcode
2. Add the **DebenuPDFLibraryCPDylibObjC1113.h** and **DebenuPDFLibraryCPDylibObjC1113.cpp** files to your project
3. Add a button and link it to an IBAction named buttonClick

```
#import "ViewController.h"
#import "DebenuPDFLibraryCPDylibObjC1113.h"

@implementation ViewController

- (IBAction)buttonClick:(id)sender
{

    NSString* dylibPath = @"...folder here...";

    // Select the appropriate 32-bit or 64-bit Dylib
#if __LP64__
    NSString* dylibFileName = [dylibPath
        stringByAppendingString:@"libDebenuPDFLibrary64CPDylib1113.dylib"];
#else
    NSString* dylibFileName = [dylibPath
        stringByAppendingString:@"libDebenuPDFLibraryCPDylib1113.dylib"];
#endif

    // Create an instance of the library
    DebenuPDFLibraryCPDylibObjC1113* DPL =
        [[DebenuPDFLibraryCPDylibObjC1113 alloc]
         initWithDylibFileName:dylibFileName];

    // Unlock the library, draw some text then extract the text
    if ([DPL UnlockKey:@"...license key here..."] == 1)
    {
        [DPL DrawText:100:700:@"Hello world"];
        NSLog(@"%@", [DPL GetPageText:0]);
    }
}

@end
```

4. ActiveX Edition

4.1 Files

The ActiveX edition of the library consists of the following two files:

`DebenuPDFLibraryCPAX1113.dll`	32-bit Windows ActiveX DLL
`DebenuPDFLibrary64CPAX1113.dll`	64-bit Windows ActiveX DLL

4.2 Registration

On Windows 7 and later the **regsvr32** command can be used to register both the 32-bit and 64-bit ActiveX libraries.

NB. These commands must be run from a console window with administrator privileges.

```
C:/> regsvr32 DebenuPDFLibraryCPAX1113.dll
C:/> regsvr32 DebenuPDFLibrary64CPAX1113.dll
```

4.3 Programming interfaces

The API is available via the following import classes:

- A C# import class
- A C++ import class

5. DLL Edition

5.1 Files

The DLL edition of the library consists of the following two files:

`DebenuPDFLibraryCPDLL1113.dll`	32-bit Windows DLL
`DebenuPDFLibrary64CPDLL1113.dll`	64-bit Windows DLL

5.2 Programming interfaces

The API is available via the following import classes:

- A C# import class
- A C++ import class