

# **Create PDF Forms in Android applications using Xamarin PDF library by Apitron**

**Written by Apitron Documentation Team**

## Introduction

We've recently published an article showing how to create Universal Window Store questionnaire application using Apitron PDF Kit for .NET ([link](#)). Now it's time to go further and demonstrate the same with Xamarin Android application.

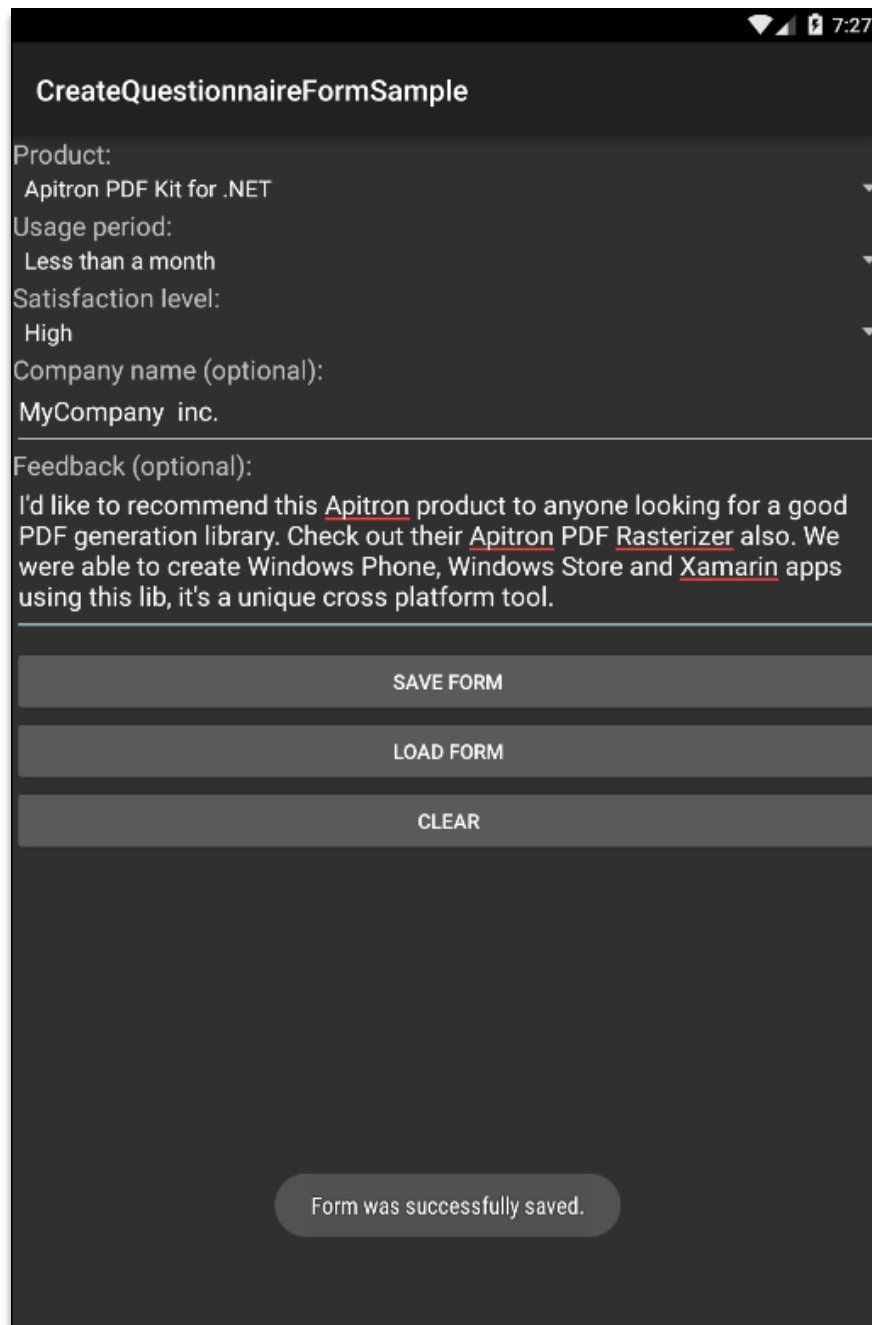
This application borrows most of its PDF generation code from the example mentioned above and, after small modifications related to file I/O, same code could be used to target all of these platforms.

The complete sample can be found in *Samples\Xamarin.Android* folder inside the download package available on our [website](#). It's called *CreateQuestionnaireFormSample*.

*Note: this application has been tested using Apitron PDF Kit for .NET version 1.0.6 and Nexus 7 device with Android 5.0.0 installed.*

## Solution overview

The demo Android application is very simple and is implemented using the single view, which holds all the controls, needed to capture user input. See its layout below:



**CreateQuestionnaireFormSample**

Product:  
Apitron PDF Kit for .NET

Usage period:  
Less than a month

Satisfaction level:  
High

Company name (optional):  
MyCompany inc.

Feedback (optional):  
I'd like to recommend this [Apitron](#) product to anyone looking for a good PDF generation library. Check out their [Apitron PDF Rasterizer](#) also. We were able to create Windows Phone, Windows Store and [Xamarin](#) apps using this lib, it's a unique cross platform tool.

SAVE FORM

LOAD FORM

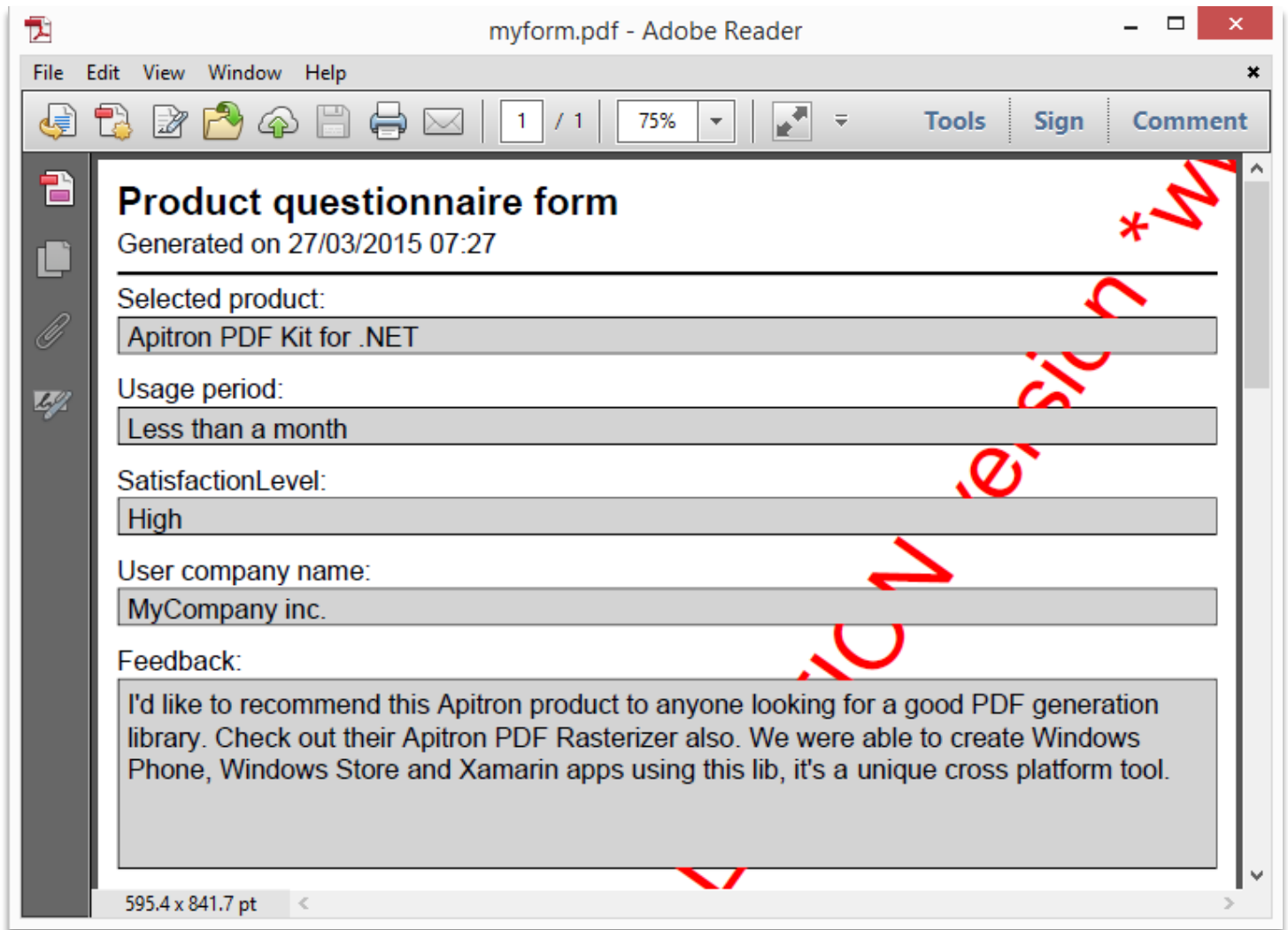
CLEAR

Form was successfully saved.

**Pic. 1 PDF form generation, Android app demo**

We used basic controls here but it's also possible to use `Xamarin.Forms`. When user clicks on "Save Form" the PDF form gets generated, "Load Form" loads it and "Clear" sets default value.

Resulting PDF form downloaded from the device is shown on the image below:



Pic. 2 Resulting PDF form generated by CreateQuestionnaireSampleForm Android application

Form fields are made read-only so they can't be changed in reader app after generation. These fields store the information entered by user and can be used to retrieve the information back when you need it. This is how "Load form" works. Resulting file is being written to directory returned by `GetExternalFilesDir()` call from the current context.

## The code

Xamarin Android UI specific code can be looked up in download package while here only PDF creation function is being demonstrated. It returns FlowDocument which can later be saved.

```
// Creates FlowDocument that represents PDF form.
private FlowDocument CreatePDFDocument()
{
    // create document with margin
    FlowDocument document = new FlowDocument(){Margin = new Apitron.PDF.Kit.Styles.Thickness(10)};

    // add form fields and set text values based on form data
    document.Fields.Add(new TextField("SelectedProduct", SelectedProduct){IsReadOnly = true});
    document.Fields.Add(new TextField("UsagePeriod", UsagePeriod) { IsReadOnly = true });
    document.Fields.Add(new TextField("SatisfactionLevel",SatisfactionLevel){IsReadOnly = true });
    document.Fields.Add(new TextField("UserCompanyName", UserCompanyName) { IsReadOnly = true });
    document.Fields.Add(new TextField("Feedback",Feedback){IsReadOnly = true,IsMultiline = true});

    // register styles defining controls appearance
    document.StyleManager.RegisterStyle(".formHeader",new Apitron.PDF.Kit.Styles.Style(){Display =
Display.Block, Font = new Font(StandardFonts.HelveticaBold, 20)});
    // set style for textblock and textbox content elements using type selectors
    document.StyleManager.RegisterStyle("TextBlock, TextBox", new Apitron.PDF.Kit.Styles.Style() {
Font = new Font(StandardFonts.Helvetica, 14)});
    // set style for textbox followed by a textblock using adjacent element selector
    document.StyleManager.RegisterStyle("TextBlock + TextBox", new Apitron.PDF.Kit.Styles.Style()
{ BorderColor=RgbColors.Black,Border=new Border(1),Height = 20,Background = RgbColors.LightGray});
    // set br style using type selector
    document.StyleManager.RegisterStyle("Br", new Apitron.PDF.Kit.Styles.Style() { Height = 10});

    // add form header
    document.Add(new TextBlock("Product questionnaire form"){Class = "formHeader"});
    document.Add(new TextBlock("Generated on " + DateTime.Now.ToString("dd/MM/yyyy HH:mm")));
    document.Add(new Hr(){Height = 2, Margin = new Thickness(0,5,0,5)});
    // add product info content
    document.Add(new TextBlock("Selected product:"));
    document.Add(new TextBox("SelectedProduct"));
    document.Add(new Br());
    // add usage info content
    document.Add(new TextBlock("Usage period:"));
    document.Add(new TextBox("UsagePeriod"));
    document.Add(new Br());
    // add satisfaction level content
    document.Add(new TextBlock("SatisfactionLevel:"));
    document.Add(new TextBox("SatisfactionLevel"));
    document.Add(new Br());
    // add company name content
    document.Add(new TextBlock("User company name:"));
    document.Add(new TextBox("UserCompanyName"));
    document.Add(new Br());
    // add feedback content
    document.Add(new TextBlock("Feedback:"));
    document.Add(new TextBox("Feedback"){Height = 100});
    return document;
}
```

## Conclusion

If you have to generate PDF on Xamarin Android or Xamarin iOS, Apitron PDF Kit library might be the right choice. In addition to these two, it's available for other platforms as well, so you'll be able share your PDF processing code and create cost-effective and easy to maintain solutions. You can download the evaluation version with lots of C# samples demonstrating the API from our website ([link](#)).

The library comes with a free book which is available for downloading from our [website](#). It will help you to understand PDF better and will provide necessary guidance in complex PDF processing scenarios. [Contact us](#) if you are interested or have any feedback.