PDF Security, part 2 – Sign PDF document and validate signatures

Written by Apitron Documentation Team

Introduction

Digital signatures are a powerful and reliable instrument for replacing the traditional signatures, and, unlike the traditional signatures, they also can be used to check whether the document was changed since signing. Using mechanisms defined in PDF standard, one may create signed PDF documents or check existing ones for changes.

Electronic signatures save time and trees, and also make the document's workflow very convenient and flexible. Since this process can be automated, it becomes possible to sign many required documents at once (e.g. in financial and government institutions) – making this essential and sometimes boring task feasible. Apitron PDF Kit .NET component can be used to create applications handling all aspects of digital signatures defined in PDF.

Signing PDF document

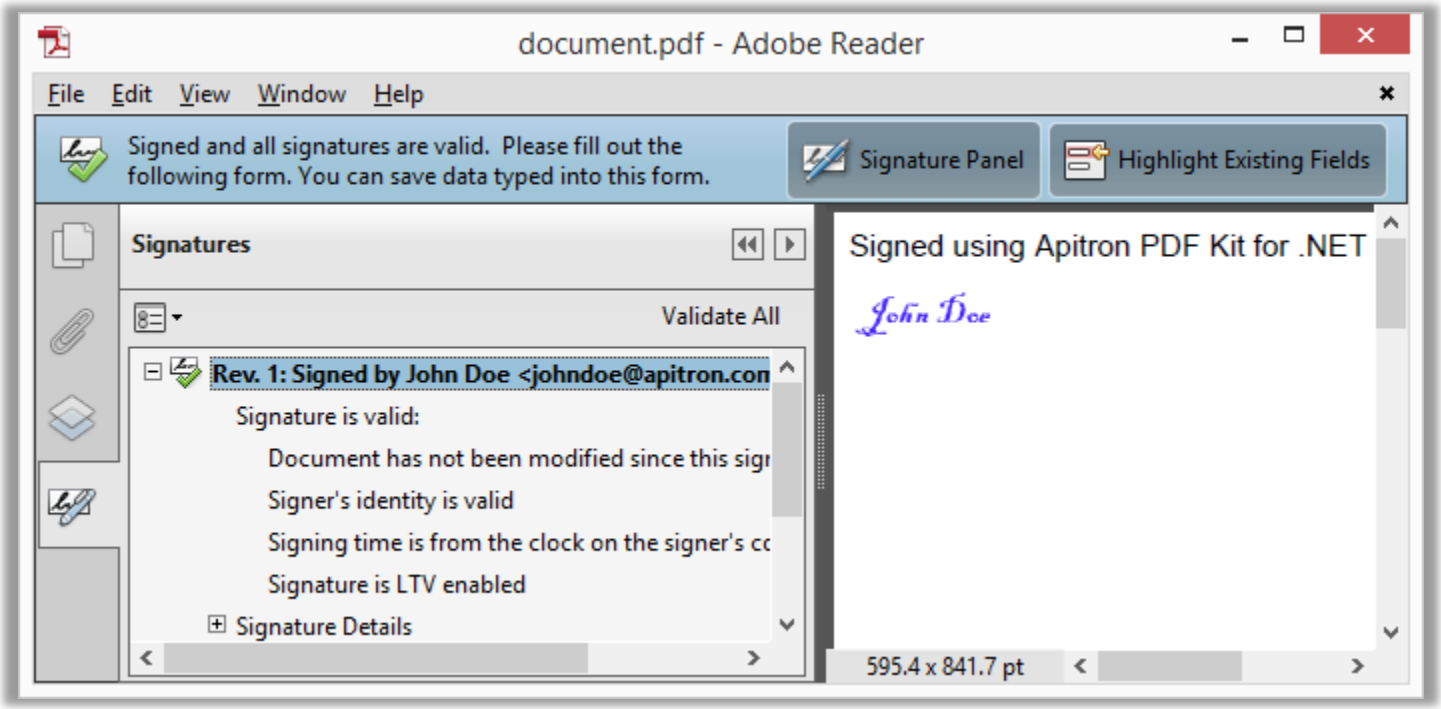
In this section we'll show how to sign a simple PDF document with a digital signature using fixed and flow layout APIs.

Fixed layout API

See the code sample below, it signs the PDF document using digital signature and adds signature view based on image.

```
public static void SignDocumentUsingFixedLayout()
{
    // open existing document for signing
    using (Stream inputStream=new FileStream("document.pdf",FileMode.Open, FileAccess.ReadWrite))
    {
        using (FixedDocument doc = new FixedDocument(inputStream))
        {
            // register signature image resource
            doc.ResourceManager.RegisterResource(
             new Image("signatureImage","signatureImage.png"));

            // create signature field and initialize it using a stored
            // password protected certificate
            SignatureField signatureField = new SignatureField("signature");
            using (Stream signatureDataStream = File.OpenRead("JohnDoe.pfx"))
            {
                signatureField.Signature = Signature.Create(new Pkcs12Store(signatureDataStream,
                 "password"));
            }

            // add signature field to a document
            doc.AcroForm.Fields.Add(signatureField);

            // create signature view using the image resource
            SignatureFieldView signatureView = new SignatureFieldView(signatureField,
             new Boundary(10, 750, 110, 800));
            signatureView.ViewSettings.Graphic = Graphic.Image;
            signatureView.ViewSettings.GraphicResourceID = "signatureImage";
            signatureView.ViewSettings.Description = Description.None;

            // add view to page annotations
            doc.Pages[0].Annotations.Add(signatureView);

            // save changes using incremental update
            doc.Save();
        }
    }
}
```

The resulting document is shown below:



Pic. 1 Signing PDF using digital signature, fixed layout API

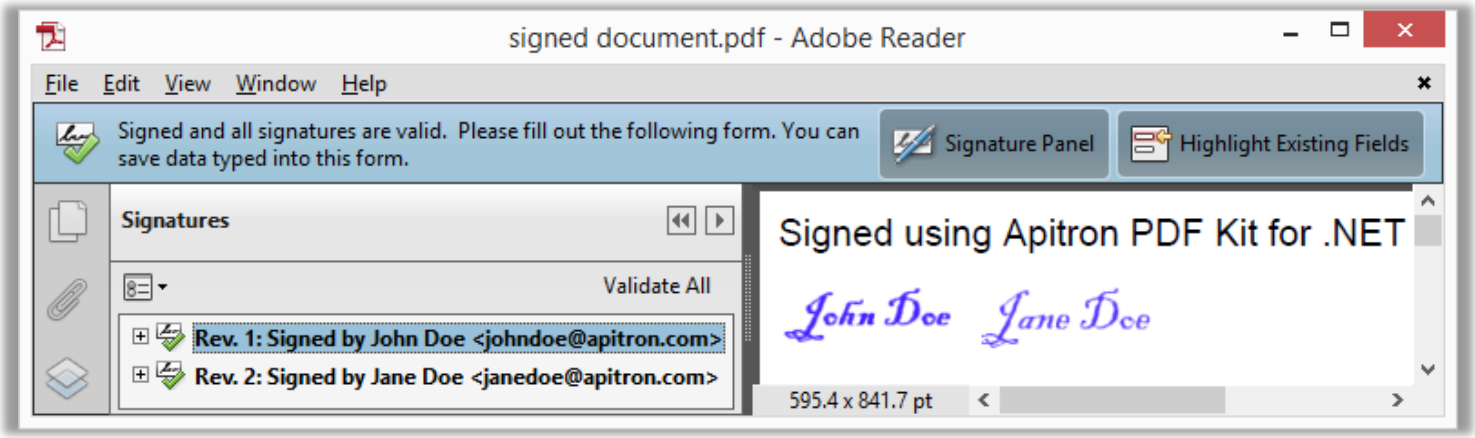
Note that added signature is LTV enabled, making the document prepared for long-term validation. LTV is an enhancement to standard revocation checking, and it allows the validity checking years after the certificate was created. Apitron PDF Kit captures the certificate's sign-time status and saves it inside the PDF document. This verification certificate remains in the document, so that its validity can be checked even at some later date, regardless of whether the certificate has expired or revoked, or if the issuing authority is no longer exists. Because the record is stored inside the signed document, it is also protected by the document's signature, reducing the chances for error or tampering.

Multiple signatures

If you were to sign the document using several signatures, just add additional signatures in the same way as you add one.

```
public void AddMultipleSignatures()
{
    // open existing document for update
    using (Stream inputStream = new FileStream("signed document.pdf",
        FileMode.Open,FileAccess.ReadWrite))
    {
        using (FixedDocument doc = new FixedDocument(inputStream))
        {
            // register signature image resource
            doc.ResourceManager.RegisterResource( new Image("signatureImage", "signatureImage.png"));
            doc.ResourceManager.RegisterResource( new Image("signatureImage2", "signatureImage2.png"));

            // create first signature field and initialize it using a stored certificate
            SignatureField signatureField = new SignatureField("signature");
            using (Stream signatureDataStream = File.OpenRead("JohnDoe.pfx"))
            {
                signatureField.Signature = Signature.Create(new Pkcs12Store(signatureDataStream,
                      "password"));
            }

            // create second signature field and initialize it using a stored certificate
            SignatureField signatureField2 = new SignatureField("signature2");
            using (Stream signatureDataStream = File.OpenRead("JaneDoe.pfx"))
            {
                signatureField2.Signature = Signature.Create(new Pkcs12Store(signatureDataStream,
                      "password"));
            }

            // add signature fields to the document
            doc.AcroForm.Fields.Add(signatureField);
            doc.AcroForm.Fields.Add(signatureField2);

            // create first signature view using the image resource
            SignatureFieldView signatureView = new SignatureFieldView(signatureField,
              new Boundary(10, 750, 110, 800));
            signatureView.ViewSettings.Graphic = Graphic.Image;
            signatureView.ViewSettings.GraphicResourceID = "signatureImage";
            signatureView.ViewSettings.Description = Description.None;

            // create second signature view using the image resource
            SignatureFieldView signatureView2 = new SignatureFieldView(signatureField2,
              new Boundary(120, 750, 220, 800));
            signatureView2.ViewSettings.Graphic = Graphic.Image;
            signatureView2.ViewSettings.GraphicResourceID = "signatureImage2";
            signatureView2.ViewSettings.Description = Description.None;

            // add views to page annotations collection
            doc.Pages[0].Annotations.Add(signatureView);
            doc.Pages[0].Annotations.Add(signatureView2);

            // save as incremental update
            doc.Save();
        }
    }
}
```

This code performs an incremental update on PDF file and signs it with several signatures. See the image below showing the result:



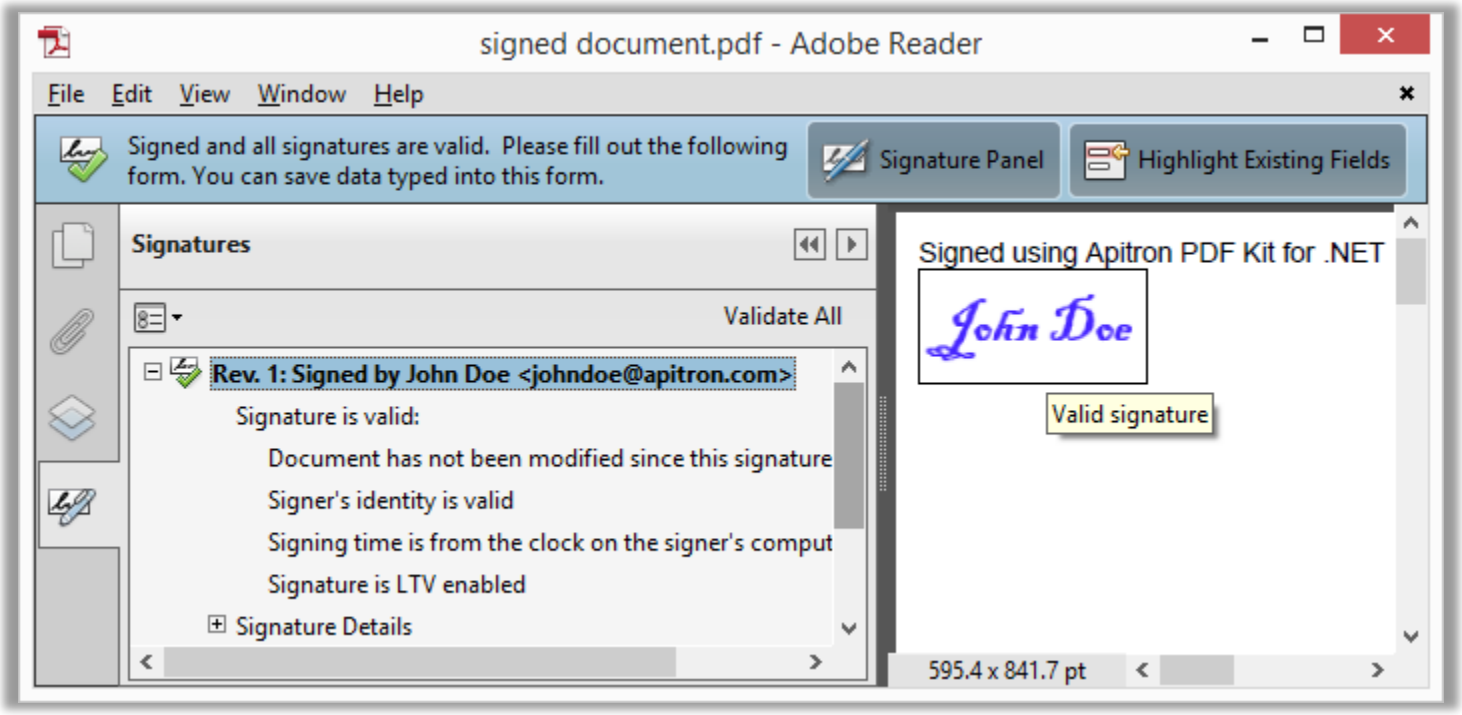
Pic. 2 Signing PDF with multiple signatures

Flow layout API

The code below shows how to sign a newly created PDF document using flow layout API. It looks similar to fixed layout sample except is uses `SignatureControl` content element which creates a placeholder for actual widget annotation (signature view).

```
public void SignDocumentUsingFlowLayout()
{
    //  create output document
    using (Stream outputStream = File.Create("signed document.pdf"))
    {
        ResourceManager resourceManager = new ResourceManager();

        // register signature image resource
        resourceManager.RegisterResource(
           new Apitron.PDF.Kit.FixedLayout.Resources.XObjects.Image("signatureImage",
              "signatureImage.png"));

        // create flow document
        FlowDocument doc = new FlowDocument(){Margin = new Thickness(10)};

        // create signature field and initialize it using a stored
        // password protected certificate
        SignatureField signatureField = new SignatureField("signature");
        using (Stream signatureDataStream = File.OpenRead("JohnDoe.pfx"))
        {
            signatureField.Signature = Signature.Create(new Pkcs12Store(signatureDataStream,
              "password"));
        }

        // create signature view using the image resource
        SignatureFieldView signatureView = new SignatureFieldView(signatureField,
              new Boundary(10, 750, 110, 800));
        signatureView.ViewSettings.Graphic = Graphic.Image;
        signatureView.ViewSettings.GraphicResourceID = "signatureImage";
        signatureView.ViewSettings.Description = Description.None;

        // set signature field's view
        signatureField.Views.Add(signatureView);

        // add signature field to a document
        doc.Fields.Add(signatureField);

        // add text and signature control representing the signature
        doc.Add(new TextBlock("Signed using Apitron PDF Kit for .NET"));
        doc.Add(new SignatureControl("signature"){Width = 100, Height = 50});

        // save document
        doc.Write(outputStream,resourceManager);
    }
}
```

Resulting document looks as follows:



Pic. 3 Signature added using flow layout API

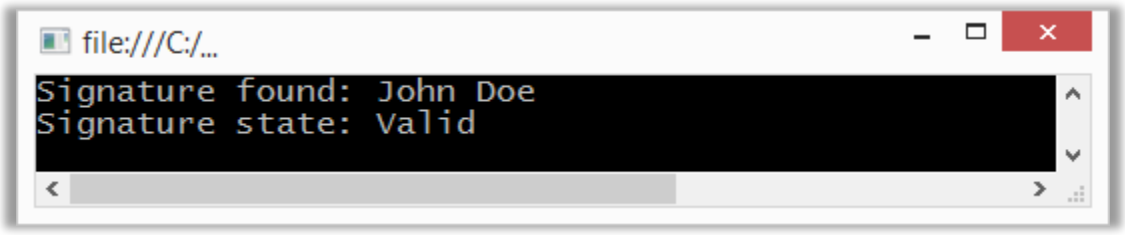
Validation of PDF signatures

What if you'd like to check whether the document's signature is valid, meaning the document wasn't changed since it was signed? It's possible to perform this check and validate the signature. In addition you could implement a certificate revocation check, but it's out of the scope of our article. We will simply validate the document's structural changes without checking the validity of the certificate itself.

See the code below:

```
/// <summary>
/// Validates the PDF signature.
/// </summary>
public void ValidateSignature()
{
    // open existing document and create output stream
    using (Stream inputStream = File.OpenRead("signed document.pdf"))
    {
        FixedDocument doc = new FixedDocument(inputStream);

        // check all signature fields
        foreach (SignatureField signatureField in doc.AcroForm.Fields.OfType<SignatureField>())
        {
            Console.WriteLine("Signature found: {0}", signatureField.Name);
            Console.WriteLine("Signature state: {0}", signatureField.IsValid ? "Valid" :
             "Invalid");
        }
    }
}
```

Resulting output is shown below:



Pic. 4 Validation of PDF signatures

Conclusion

In this article we demonstrated how to work with PDF signatures: sign PDF documents, validate existing signatures, and use multiple signatures at once. While the theory behind the electronic signatures remains fairly complex, and its implementation and testing may take significant amount of time, using tools like Apitron PDF Kit it's easy to get the job done without diving too much in mathematics and PDF specifics. Contact us if you have any questions regarding digital signatures or PDF in general and get an advice from PDF experts.