This is a special *iText* PDF

For this release, we have decided to create a PDF file that showcases a lot of what is possible with iText by Apryse.

This is no ordinary PDF, as it is PDF/A-4, PDF/UA-2 and WTPDF compliant, created with the help of pdfHTML.

You can read more about it below.

Table of Contents

Release iText Core 9.0

Release date: Nov 18th, 2024

For this Q4 release, we're pleased to announce a new major version of iText. iText Core version 9.0 introduces significant new features, support for new specifications, and revised APIs to delight developers.

We've added support for the ISO/TS 32003 and 32004 standards, enabling even more secure PDF documents. Also on the list is the finalized digital signature validation module, along with a new API to easily get layers used in a page, and improved PDF/UA signing.

That's by no means all though, as we're also releasing new versions of all add-ons across the iText Suite PDF SDK. Let's take a closer look at what iText 9 brings to the table.

⚠ Since iText 9.0 is a major release version, it naturally means a break in compatibility with iText 8. As the developers among you will appreciate, this is necessary to make quality-of-life improvements and reduce our technical debt.

However, thanks to the solid foundations laid with iText 7 and 8 there are few major API differences in iText 9. The iText Knowledge Base has extensive documentation on the breaking changes to ease the migration of existing projects from earlier versions.

New ISO Standards Support

First and foremost, iText Core version 9.0 incorporates support for the very latest ISO PDF document security standards. ISO/TS 32003 adds AES-GCM encryption to the PDF 2.0 specification, allowing documents to be protected with high-performance, yet extremely secure encryption.

ISO/TS 32004 introduces an integrity protection mechanism for encrypted PDFs, using a Message Authentication Code (MAC) to ensure authenticity. To fully understand what this means for securing PDF documents, we highly recommend reading two great articles on the PDF Association site: ISO 32004: an overview and the follow-up MACs vs. signatures in PDF which go into detail on this subject.

Digital Signature Validation Module

On the subject of PDF digital signatures, we're also proud to present the finalized version of

our dedicated validation module. This forms an integral piece of iText's enhanced digital signing capabilities introduced with iText 8.

The aim is to provide simpler, more extensive API methods to not just sign PDF documents, but also validate the digital signatures within them – whether iText created them or not. Since you can validate multiple document revisions as well as certificate chains, iText can now be your Swiss Army knife for digital signatures, as well as PDF creation and manipulation.

This release enables you to only validate a single signature in a document, as opposed to all signatures. In addition, the signature validator will now work for encrypted documents.

API Improvements

There are also major refinements to iText's API. These include streamlining PDF/A and PDF/UA creation and conformance to simplify the process. We've also developed a new API to identify the layers used in a page. This will help to find which Optional Content Groups (OCGs) belong to which page in a document.

Signing of PDF/UA documents has been improved. When creating a signature form field iText will now take into account if an alternative description is set in the accessibility properties of the signature appearance. Additionally, if you forget to set a font for the signature appearance this will now result in a conformance exception, rather than a property error.

Further improvements have been made to the rebuilding of invalid cross-reference (xref) tables in corrupt documents. When iText encounters and resolves such errors in non-strict mode, specific information on the cause will now be provided.

Adding to the recent addition of RSASSA-PSS encryption support for .NET, this release now allows it to be supported in FIPS mode.

Along with that is improved font selection and general handling, performance enhancements, and much more. iText has a reputation amongst Java and .NET developers for its speed and ease of use, and we're ensuring that remains the case in the future.

Make sure to check out the Breaking Changes if you're migrating from a previous version of iText.

Pull Requests

Once again, we'd like to thank Matthias Valvekens for another pull request submission. This relates to Unicode, and adds support for platform 0 encoding 3 in the Truetype and OpenType character map table. This is used in some fonts shipped with macOS, among other

places.

Bug Fixes and Miscellaneous

For content extraction, we fixed a bug in the `RegexBasedLocationExtractionStrategy` API (Java/.NET) . It now produces better results by default when processing multiple pages.

We fixed an edge-case bug in Certificate Revocation List validation. When a CRL response existed, but its issuer was neither trusted, nor self-signed, it could result in a stack overflow error.

In addition, a fix was made to SVG rendering to honor `dy` attributes in parent text attributes.

Other Stuff

As always, you can see the Changelog below to see the full rundown on what's new in Core, and details of other improvements and bugfixes for this release.

> If you use iText for digital signing, you may be interested in the Digital Signatures Hub which contains a ton of useful resources and examples. In particular, we have a new chapter to our Digital Signing with iText series. In Part V, we take you through the steps of signing PDFs with Java via a remote signing service offering CSC API access.

Don't forget that in addition to the resources on our Knowledge Base, on our GitHub you can find a ton of useful up-to-date samples in the following repos:

Java

- https://github.com/itext/itext-publications-examples-java
- https://github.com/itext/itext-publications-book-java
- https://github.com/itext/itext-publications-signing-examples-java
- https://github.com/itext/itext-publications-signatures-java
- https://github.com/itext/itext-publications-highlevel-java
- https://github.com/itext/itext-publications-jumpstart-java

.NET

- https://github.com/itext/itext-publications-samples-dotnet

If you want to create ZUGFeRD/Factur-X-e-invoices with iText Core, we have both Java and .NET code samples available targeting the current ZUGFeRD/Factur-X specification. They demonstrate how to embed the XML invoice data and add the metadata required for conformance.

Bear in mind that our `master` branch contains samples for the current stable release, while the default `develop` branch is for the bleeding edge commits towards the next release.

Also, don't forget to check out the release-related examples below, as well as the updated Core add-ons in the iText Suite we've released this time:

iText Suite 9.0 Releases

- Release pdfCalligraph 5.0
- Release pdfOCR 4.0
- Release pdfOptimizer 4.0
- Release pdfSweep 5.0
- Release pdfXFA 5.0

Release Related Examples

- Support for Message Authentication Code (MAC) integrity protection
- AES-GCM Encryption Support

Downloads

	GitHub	Maven	NuGet	Artifactory
iText Core – 9.0.0 (Java)	link	link	N/A	link
iText Core – 9.0.0 (.NET)	link	N/A	link	link

Changelog

New features

- Support ISO/TS 32004:2024 - Integrity protection in encrypted documents in PDF 2.0 (PDF MAC)
- Support ISO/TS 32003:2023 - AES-GCM in PDF 2.0
- Finalized validation module
- Develop API to get used layers in a page

Improvements

- Support RSASSA-PSS encryption in BC fips mode on .NET
- PDF/UA and signing
- SignatureValidator doesn't work with encrypted documents
- Support linearized PDF files in PdfRevisionReader
- When rebuilding xref, log more detailed info
- Support single signature validation

Bug fixes

- Improve API of RegexBasedLocationExtractionStrategy
- Valid CRL having verifiable issuer results in StackOverflow
- SVG dy attribute not honoured

Deprecation of Older iText Versions

We're taking this opportunity to announce End of Life dates for deprecated iText versions. EOL for iText 7.1 will be April 2025, and iText 7.2 will be October 2025. As for iText 8.0, this will reach EOL in October 2026.

EOL for these versions means they will transition to maintenance mode, and only receive security patches. However, just as with iText 5 we will continue to provide support for our customers. Even though iText 5 has been in maintenance mode for 2016, we regularly release new versions to address CVEs and other security-related bugfixes.

eBooks

- Blockchain for PDF Documents
- iText: Jump-Start Tutorial for .NET
- iText: Jump-Start Tutorial for Java
- iText: Converting HTML to PDF with pdfHTML
- iText: Building Blocks
- Best iText 7 Questions on StackOverflow

Installation Instructions

- Installing iText for Java
- Installing iText for .NET
- Installing iText Community for Java developers
- Installing iText Community for .NET developers

Examples (latest ones)

- PAdES Signing High Level API
- iText Core: Signature Appearance Improvements
- iText 8.0.2 Delivers PDF/A-4 Support
- High-level Annotation Flattening
- FIPS & SHA3 Examples for iText Core 8.0.0

FAQ (latest ones)

- I upgraded to iText version 7 or 8, how do I use my license?
- How to change a background Image into a watermark by altering the opacity?
- How to change the text color of an AcroForm field?
- Is it safe to remove XFA?
- How to fill XFA form using iText without breaking usage rights?

Features of this PDF

- Used pdfHTML to generate the PDF content (the HTML content is attached)
- It is both PDF/A-4 and PDF/UA-2 compliant (making it also a WTPDF)
- There is a MAC protected (AES 256 Encryted, SHA 256 Protected. Password is

'itext') version of this PDF attached (had to be separate, as PDF/A-4 does not allow it)

- Digitally signed using a Portuguese Identity Card (scroll below to check the code on how to validate it!)
- The main logo is generated using iText's custom SVG rendering engine
- Dynamically generated table of contents and bookmarks
- *Creatively* used Layers to toggle between Java and .NET code below
- Automatic Pagination by using our Events engine

To run the project (using .NET 8), just check the instructions on the attached file `README.md`.

Signed off by:

Digitally signed by ANDRÉ VENTURA LEMOS
Date: 2024.11.18 18:23:29 +01:00
Reason: Release notes for iText 9.0.0
Location: Ghent (Belgium)

Generated by: Content: Ian Morris Source code: Guust Ysebie

Addendum: Signature validation example java

```
package com.itextpdf.signatures;
import com.itextpdf.kernel.pdf.PdfDocument;
import com.itextpdf.kernel.pdf.PdfReader;
import com.itextpdf.signatures.validation.SignatureValidationProperties;
import com.itextpdf.signatures.validation.SignatureValidator;
import com.itextpdf.signatures.validation.ValidationOcspClient;
import com.itextpdf.signatures.validation.ValidatorChainBuilder;
import com.itextpdf.signatures.validation.context.CertificateSource;
import com.itextpdf.signatures.validation.context.CertificateSources;
import com.itextpdf.signatures.validation.context.TimeBasedContext;
import com.itextpdf.signatures.validation.context.TimeBasedContexts;
import com.itextpdf.signatures.validation.context.ValidatorContext;
import com.itextpdf.signatures.validation.context.ValidatorContexts;
import com.itextpdf.signatures.validation.report.ValidationReport;
import java.io.IOException;
import java.time.Duration;

public class ValidationExample {
        public static void main(String[] args) {
                String path = "<path/to/signed/pdf>";
                SignatureValidationProperties properties = GetSignatureValidationProperties();
        properties.addOcspClient(new ValidationOcspClient());
        properties.addCrlClient(new CrlClientOnline());
                IssuingCertificateRetriever certificateRetriever = new IssuingCertificateRetriever();
                ValidatorChainBuilder validatorChainBuilder = new ValidatorChainBuilder();
        validatorChainBuilder.withIssuingCertificateRetrieverFactory(
                () -> certificateRetriever).withSignatureValidationProperties(properties);
                ValidationReport report;
                try (PdfDocument document = new PdfDocument(new PdfReader(path))) {
                        SignatureValidator validator = validatorChainBuilder.buildSignatureValidator(document);
                        // Validate all signatures in the document.
            report = validator.validateSignatures();
        } catch (IOException e) {
                        throw new RuntimeException(e);
        }
                System.out.println(report);
    }

        private static SignatureValidationProperties GetSignatureValidationProperties() {
                SignatureValidationProperties properties = new SignatureValidationProperties();
        properties.setRevocationOnlineFetching(ValidatorContexts.all(), CertificateSources.all(), TimeBasedContexts
                .all(), SignatureValidationProperties.OnlineFetching.ALWAYS_FETCH);
        properties.setFreshness(ValidatorContexts.All(), CertificateSources.all(), TimeBasedContexts.of(
                                TimeBasedContext.HISTORICAL), Duration.ofDays(-5));
        properties.setContinueAfterFailure(
                                ValidatorContexts.of(ValidatorContext.OCSP_VALIDATOR, ValidatorContext.CRL_VALIDATOR),
                                CertificateSources.of(CertificateSource.CRL_ISSUER, CertificateSource.OCSP_ISSUER, CertificateSource
                        .CERT_ISSUER), false);
                return properties;
    }
}
```