September 2, 2025

# RELEASE NOTES

### This is a special iText PDF

As per usual, 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/UA-2 and PDF/A-4F compliant, created with the help of pdfHTML.

You can read more about it below.

## Table of Contents

## Release iText Core 9.3.0

> As always, we've made the Core release notes into a showcase PDF document. Not only does it conform to the latest PDF/UA-2 standard for accessible PDF documents, as well as the PDF/A-4F archiving standard, it's also digitally signed. You can find the source code and resources required to recreate the document yourself embedded. Have fun!

**Release date: Sep 2nd, 2025**

For this Q3 release of iText Core we've further enhanced iText's digital signature validation by adding support for the official EU eIDAS Trusted Lists. Not only that, there's also improvements to thread safeness and further work on the .NET MAUI support we introduced in version 9.2.0.

## EU Trusted Lists in Validation

Furthering our efforts to make developers' lives easier when dealing with PDF digital signatures, we're pleased to announce that iText now supports retrieval, validation, and usage of the EU's List of Trusted Lists (LOTL). This greatly simplifies the process of establishing a chain of trust for electronic signatures, and helps ensure that signatures validated with iText meet stringent European standards for trust and authenticity..

For those unaware, the LOTL is a central, signed XML file containing links to trusted lists from EU and EEA Member States. These Member State lists identify both trust service providers and the trust services they offer; e.g. digital signatures and seals. The LOTL is an official resource which aids achieving compliance with EU eIDAS regulations. Previously, you would need to manually provide the trusted certificates to use for validation with iText; which is still possible, but is less convenient. Now, iText will retrieve, parse, and validate the LOTL to provide the root trusted certificates for you.

For security reasons, rather than expect iText to repeatedly retrieve the trusted certificates to validate the LOTL from the Internet, we instead provide a specialized repository which has the required certificates pre-downloaded. This can be found on GitHub, or alternatively on Maven and NuGet.

See European Union List of Trusted Lists in Validation to find full details on this implementation and

its usage.

## Thread Safety Improvements

Internal updates have improved thread safety across key components, making iText Core more robust in multithreaded environments. This is especially beneficial for developers building scalable, concurrent applications.

## Faster XMP Metadata Parsing on .NET

.NET performance improvements include replacing `for` loops with `foreach` loops. The `iText.Kernel.XMP.Impl.ParseRdf` class is heavily used in XML-based metadata parsing, and the change significantly speeds up processing of large XMP collections.

## Improved .NET MAUI Compatibility

Building on the support for Native AOT compilation in the previous release, iText Core now offers better compatibility with .NET MAUI, supporting cross-platform mobile and desktop development and integration into modern .NET applications.

## Pull Requests

We'd like to thank SangeethaDivya for their contribution to remove duplicate constants on .NET, which we used as a basis for a Java reimplementation and ported it to .NET. Thanks also to craffael who fixed a typo in the PDFA-1 checking code which led to documents with no device-dependent color spaces failing the checks.

## Bug Fixes and Miscellaneous

A bug in PDF 2.0 structure destinations has been fixed, improving how tagged content is linked and navigated when converting from HTML. This is now more in line with the PDF 2.0 and PDF/UA-2

specifications and is particularly useful for accessibility and structured document workflows.

We've also fixed an issue related to color depth support in PDF image data streams, which would result in an `com.itextpdf.io.exceptions.IOException: The color depth 1 is not supported.` error.

We've also resolved a `StackOverflowException` resulting from invalid PDFs with cyclic references in the trailer dictionary, improving robustness and error handling.

## Other Stuff

> 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.

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. Read this article to learn more about ZUGFeRD/Factur-X, and using these code samples to create EN 16931-compatible e-invoices.

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 9.3 Releases

Release pdfCalligraph 5.0.3
Release pdfHTML 6.2.1
Release pdfOCR 4.1.0
Release pdfOptimizer 4.1.0
Release pdfSweep 5.0.3
Release pdfXFA 5.0.3

## Downloads

| | GitHub | Maven | NuGet | Artifactory |
|---|---|---|---|---|
| iText Core – 9.3.0 (Java) | link | link | N/A | link |
| iText Core – 9.3.0 (.NET) | link | N/A | link | link |

## Changelog

### New features

- DEVSIX-9161 – EU Trusted Lists in Validation.

### Improvements

- DEVSIX-9277 – Optimized .NET Metadata parsing.
- DEVSIX-9019 – Improved thread safety in the core library.
- DEVSIX-8596 – Advances compatibility with MAUI and .NET deployment models.

### Bug fixes

- DEVSIX-9203 – Stack Overflow fix when opening certain PDFs.
- DEVSIX-8614 – Fixed issues with unsupported 1-bit color depths.
- DEVSIX-8864 – PDF/UA-2 Destination Support.

## 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 since 2016, we regularly release new versions to address CVEs and other security-related bugfixes. See CVEs or the iText 5-specific CVEs page for more information.

## Contributors

We'd like to shout out the following contributors for this release:

- Core team
  - Eugene Bochilo
  - Dmitry Chubrick
  - Angelina Pavlovets
  - Alexandr Pliushchou
  - Vitali Prudnikovich
  - Dmitry Radchuk
  - Andrei Stryhelski
  - Nanou Persoons
  - Glenn Volckaert
  - Alexandr Fedorov
  - Guust Ysebie
  - Yulian Gaponenko
  - Raf Hens
- Product / Marketing
  - André Lemos
  - Ian Morris
- Infrastructure / Devops
  - Marco Andries
  - Yauheni Borbut
- Research
  - Michaël Demey
  - Vlad Lipskiy
- Input from other teams
  - Rainer Plöckl
  - Alison Anderson
- Input from outside
  - Michael Klink
  - Matthias Valvekens

## 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)

Refactored PDF Conformance Architecture
PAdES Signing High Level API
iText Core: Signature Appearance Improvements
iText 8.0.2 Delivers PDF/A-4 Support
High-level Annotation Flattening

## FAQ (latest ones)

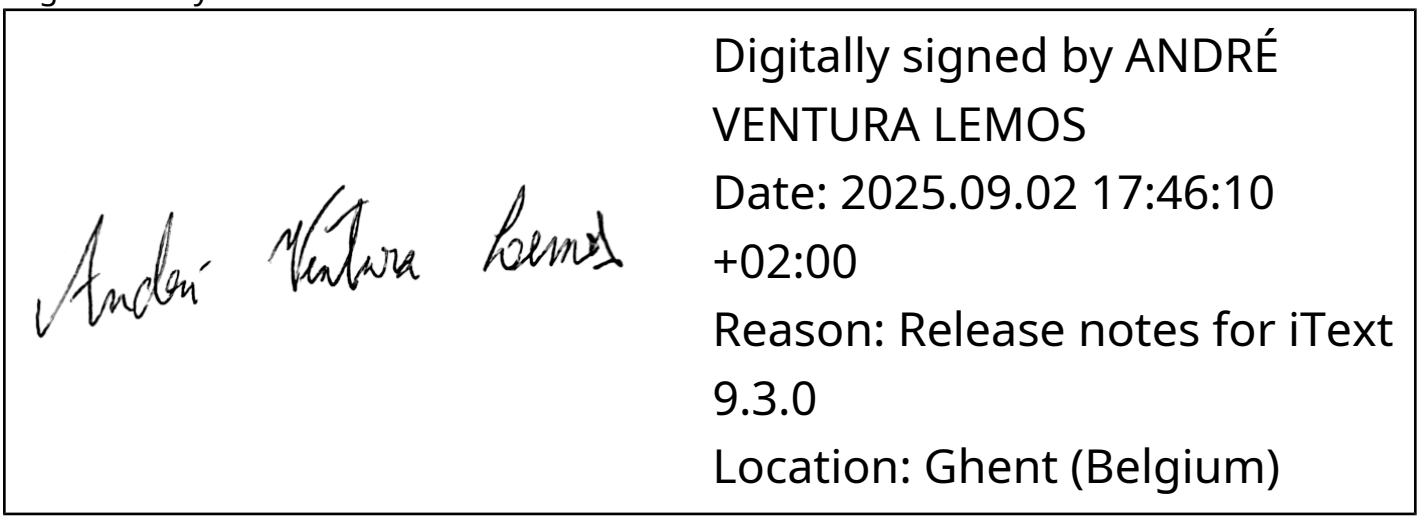
How can I find code examples for specific iText versions or releases?
How to read text from a specific position?
How does iText handle PDF encryption?
Resolving Microsoft.DotNet.PlatformAbstractions.dll Exception in Azure Functions
How to deploy an iText License for Azure PowerShell?

## Features of this PDF

- Used pdfHTML to generate the PDF content (the HTML content is attached)
- It is both PDF/A-4f 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
- Fonts were *subsetted* for a smaller file size

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

Signed off by:




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;
    }
}
```