

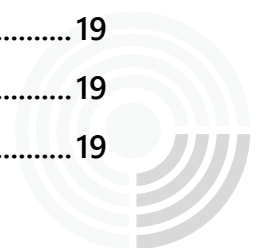


eSignAnyWhere 21.16

Release Features

Table of Contents

Envelope Expiration	4
What is it and what is it for?	4
How does it work?.....	4
Absolute date	4
Relative date	5
API configuration.....	5
Code Block for absolute expiration date (JSON).....	6
Code Block for relative expiration date (JSON)	6
Full Sample (JSON)	7
Customization of localizations.....	12
What is it and what is it for?	12
How to activate	12
How does it work?.....	12
In detail the following options are possible:	12
Summary of all possible configurations:	13
Signing certificate filter	14
What is it and what is it for?	14
How does it work?.....	14
Remove REST API v1 and v2	15
MetaData for templates	16
What is it and what is it for?	16
How to activate	16
How does it work?.....	16
Merging a draft's bulk recipient list with template.....	18
What is it and what is it for?	18
How does it work?.....	18
Stamp Imprint	19
What is it and what is it for?	19
How to activate	19
How does it work?.....	19



Configuration options	21
Colors	21
Fonts	21
Layouts	22
Layer	22
Row	22
Cells	23
UI clean up: Hide non-accessible features	30
What is it and what is it for?	30
How to activate	30
WscFeatureUpdateJob	30
User Session Permission Cache TTL	31
How does it work?	31
Password change	31
Admin Web Feature Set	31
Generic Signing Plugin Improvements	32
What is it and what is it for?	32
How to activate	32
New features/improvements:	32



Envelope Expiration

What is it and what is it for?

The feature implementation change of envelope expiry allows to specify absolute expiration timestamps (date and time), beside relative expiration date. For the relative expiration date, it allows specifying days, hours and minutes instead of just days. An expiration of less than one day is now supported. This enables senders of an envelope to set the exact expiration timestamp of an envelope, e.g. for offers valid just till an exact time like midnight.

How does it work?

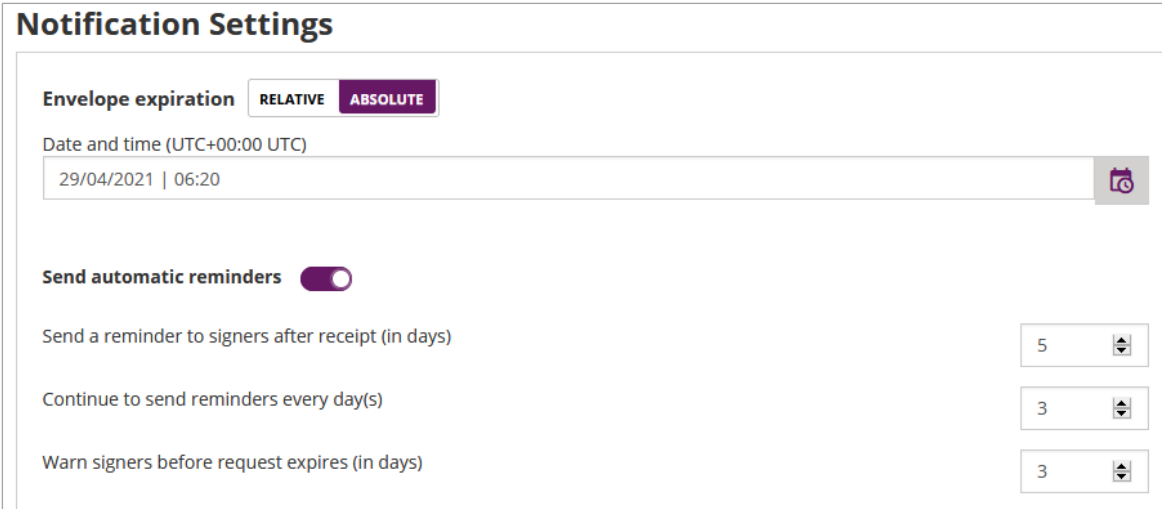
The following information relates to settings in the UI.

Before sending you have to choose between the following two options:

- Envelope expiry as relative time, provided in days:hours:minutes after sending the envelope
- Envelope expiry as absolute time, provided as timestamp and selected in the timezone in the account settings

On the last page (envelope summary) of creating and sending an envelope you can find the settings for the absolute and relative date. Please see the next two figures for detail information:

Absolute date



The screenshot shows the 'Notification Settings' interface. At the top, there are two tabs: 'RELATIVE' and 'ABSOLUTE', with 'ABSOLUTE' being the active tab. Below the tabs, there is a text input field for 'Date and time (UTC+00:00 UTC)' containing '29/04/2021 | 06:20' and a calendar icon on the right. Underneath, there is a toggle switch for 'Send automatic reminders' which is currently turned on. At the bottom, there are three dropdown menus: 'Send a reminder to signers after receipt (in days)' set to 5, 'Continue to send reminders every day(s)' set to 3, and 'Warn signers before request expires (in days)' set to 3.

Relative date

Notification Settings

Envelope expiration **RELATIVE** ABSOLUTE

Days: 28 Hours: 0 Minutes: 0 [RESET DATA](#)

Envelope expires on 29/04/2021 | 06:20 UTC

Send automatic reminders

Send a reminder to signers after receipt (in days) 5

Continue to send reminders every day(s) 3

Warn signers before request expires (in days) 3

For more information about how you can configure the absolute and relative date via API please see also the next section.

API configuration

Note: The following information is valid for envelopes as well as for drafts:

Settings for the relative time:

- Envelope expiry as relative time, provided in seconds (or milliseconds) after sending the envelope
- Envelope expiry as relative time, in days after sending the envelope

Settings for the absolute time:

- Envelope expiry as absolute time, as timestamp in GMT (with Z in the timestamp text) or optional with defined timezone

Please note the following: Defining no option will lead to usage of the default value of 28 days. Moreover, if you configure an absolute expiry time which is already in the past at time of sending the envelope, an error message will be returned.

For the draft, it is required to have both options - store absolute or relative time. For the envelopes, only an absolute timestamp is required to be set and considered while being in progress. Please also see the next REST API configuration.



Code Block for absolute expiration date (JSON)

```
// works with envelope/send
// works with draft/create
"SendEnvelopeDescription": {
  ...
  "ExpirationDate": "2021-03-22T08:34:50.775Z",
  ...
}

// works with draft/createFromTemplate
// works with envelope/sendFromTemplate
"EnvelopeOverrideOptions": {
  ...
  "ExpirationDate": "2021-03-22T08:34:50.775Z",
  ...
}

// works with draft/update
{
  "ExpirationDate": "2021-03-22T08:34:50.775Z"
}
```

Code Block for relative expiration date (JSON)

```
// works with envelope/send
// works with draft/create
"SendEnvelopeDescription": {
  ...
  "DaysUntilExpire": 1,
  // EXCLUSIVE OR!! (Only one is allowed - otherwise error)
  "ExpirationInSecondsAfterSending": 86400,
  ...
}

// works with draft/createFromTemplate
// works with envelope/sendFromTemplate
"EnvelopeOverrideOptions": {
  ...
  "DaysUntilExpire": 1,
  // EXCLUSIVE OR!! (Only one is allowed - otherwise error)
  "ExpirationInSecondsAfterSending": 86400,
  ...
}

// works with draft/update
{
  "ExpirationInSecondsAfterSending": 86400,
}
```



Full Sample (JSON)

Please note the following: An absolute expiration date is used in this example. Please also note the date format (YYYY-MM-DD[T]HH:mm:ssZ).

For example:

- 2022-12-24T18:21Z
- 2022-01-01T12:00:00+01:00

In the following full configuration a date was given with hours, minutes, seconds and milliseconds.

```
{
  "SspFileIds": [
    "af7475c2-1234-5678-b506-a511b5a56874"
  ],
  "SendEnvelopeDescription": {
    "Name": "Sample Contract",
    "EmailSubject": "Bitte unterschreiben Sie den Vertrag",
    "ExpirationDate": "2021-03-22T08:34:50.775Z",
    "Steps": [
      {
        "OrderIndex": 1,
        "Recipients": [
          {
            "Email": "placeholder1@placeholder1.com",
            "FirstName": "placeholder1",
            "LastName": "placeholder1",
            "LanguageCode": "en-US",
            "DisableEmail": false,
            "AllowDelegation": true,
            "SkipExternalDataValidation": false,
            "AuthenticationMethods": [
            ]
          }
        ]
      }
    ],
    "RecipientType": "Signer",
    "WorkstepConfiguration": {
      "WorkstepLabel": "RSGtestWorkstepForSample",
      "SmallTextZoomFactorPercent": 100,
      "FinishAction": {
        "ServerActions": [
        ],
        "ClientActions": [
        ]
      }
    }
  }
}
```

```

    },
    "TransactionCodeConfigurations":[
        //not required as we don't use those features in this envelope
    ],
    "SignatureConfigurations":[
        {
            "PdfSignatureProperties":{
                "PdfAConformant":false,
                "PAdESPart4Compliant":false,
                "IncludeSigningCertificateChain":false,
                "SigningCertificateRevocationInformationIncludeMode":"DoNotInclude"
            },
            "PdfSignatureCryptographicData":{
                "SignatureHashAlgorithm":"Sha256",
                "SigningCertificateDescriptor":{
                    "Identifier":"14527a6bcfa8b4d7d0183fca6b735b1c246d14ae",
                    "Type":"ShalThumbprint",
                    "Csp":"Default"
                }
            }
        },
        {
            "SpcId":"padesSigningId",
            "PdfSignatureProperties":{
                "PdfAConformant":false,
                "PAdESPart4Compliant":true,
                "IncludeSigningCertificateChain":false,
                "SigningCertificateRevocationInformationIncludeMode":"IncludeDss"
            },
            "PdfSignatureCryptographicData":{
                "SignatureHashAlgorithm":"Sha256",
                "SigningCertificateDescriptor":{
                    "Identifier":"14527a6bcfa8b4d7d0183fca6b735b1c246d14ae",
                    "Type":"ShalThumbprint",
                    "Csp":"Default"
                }
            }
        },
        {
            "SpcId":"timestampSigningId",
            "PdfSignatureProperties":{
                "PdfAConformant":false,
                "PAdESPart4Compliant":false,
                "IncludeSigningCertificateChain":false,
                "SigningCertificateRevocationInformationIncludeMode":"DoNotInclude",
                "SignatureTimestampData":{
                    "Uri":"https://timestamp.test.namirialtsp.com",
                    "Username":"xyzmo",
                    "Password":"xyzmo",
                    "SignatureHashAlgorithm":"Sha256"
                }
            }
        }
    ]
}

```



```

    }
  },
  "PdfSignatureCryptographicData":{
    "SignatureHashAlgorithm":"Sha256",
    "SigningCertificateDescriptor":{
      "Identifier":"14527a6bcfa8b4d7d0183fca6b735b1c246d14ae",
      "Type":"ShalThumbprint",
      "Csp":"Default"
    }
  }
},
"ViewerPreferences":{

},
"Policy":{
  "GeneralPolicies":{
    //we are using server defaults and organization defaults only.
  },
  "WorkstepTasks":{
    "PictureAnnotationMinResolution":0,
    "PictureAnnotationMaxResolution":0,
    "PictureAnnotationColorDepth":"Color16M",
    "SequenceMode":"NoSequenceEnforced",
    "PositionUnits":"PdfUnits",
    "ReferenceCorner":"Lower_Left",
    "Tasks":[
      {
        "PositionPage":7,
        "Position":{
          "PositionX":126.139999389648,
          "PositionY":614.588989257813
        },
        "Size":{
          "Height":40,
          "Width":150
        },
        "AdditionalParameters":[
          {
            "Key":"enabled",
            "Value":"1"
          },
          {
            "Key":"positioning",
            "Value":"onPage"
          },
          {
            "Key":"req",
            "Value":"1"
          }
        ]
      },

```

```

        {
            "Key": "fd",
            "Value": ""
        },
        {
            "Key": "fd_dateformat",
            "Value": "dd-MM-yyyy HH:mm:ss"
        },
        {
            "Key": "fd_timezone",
            "Value": "datetimeutc"
        }
    ],
    "AllowedSignatureTypes": [
        {
            "IsBio": false,
            "AllowSkipBiometricVerification": false,
            "AllowBiometricStoringOnly": false,
            "SignedResponseWithoutBioData": false,
            "Id": "8f41033a-11ac-48b0-a64b-72a32d20853c",
            "DiscriminatorType": "SigTypeBiometricSignature",
            "Preferred": false
        }
    ],
    "UseTimestamp": false,
    "IsRequired": true,
    "Id": "1#XyzmoDuplicateIdSeperator#sigField1",
    "DisplayName": "Unterschrift Antragsteller",
    "DocRefNumber": 1,
    "DiscriminatorType": "Signature"
}
]
}
},
"Navigation": {
    "HyperLinks": [
        ],
    "Links": [
        ],
    "LinkTargets": [
        ]
    }
},
"DocumentOptions": [
],
"UseDefaultAgreements": false

```

```
    }  
  ],  
  "AddFormFields":{  
  
  },  
  "MetaDataXml":"Free data section for integration"  
}  
}
```



Customization of localizations

What is it and what is it for?

Organization administrators can now customize localizations (especially text translations) per organization. It enables a higher level of customization and more adjustments of the SignAnyWhere Viewer (Signer Front-End). The front-end can now be adjusted to the company's common wording.

How to activate

Following feature flag is necessary for this feature: *UseCustomizationId*

Please note the following: This is a non-default feature of eSignAnyWhere. If you are interested in this optional feature please [contact us](#).

How does it work?

In detail the following options are possible:

- Customize localizations for a specific eSAW organization (only possible in combination with eSAW and Customization Service):
 - The WSC customization package contains a `Localizations.template.custom.json` containing all keys & english values of the XLF but in JSON format.
 - You can create your own `Localizations.*.custom.json` files (e.g. `Localizations.de.custom.json`), modify the elements which should be customized and add the file to the customization package.
 - For better maintenance all items which have not been changed should be removed from the JSON file.

Note: Not all sections of the XLF file are allowed to be configured that way (e.g. to prevent possible problems with changing audit trail texts)



- Customize localizations for the whole SSP instance (or in case Customization Service is not used):
 - Grab the Localizations.template.custom.json from the default customization service package (see Release Notes)
 - Modify the file like described above (content, filename)
 - Place the file in the i18n folder next to the XLF files
- Combination of these two possibilities
 - In case of a 'combined' customization (JSON files available in Customization Service and i18n folder) the two JSON files will be merged. In case items are included in both files, the items from the Customization Service will override the ones from the i18n folder.

Please note: It is not recommended anymore to change the XLF files (SignAnyWhere Viewer) directly!

Summary of all possible configurations:

- Customize WSC localizations using a local *.custom.json file in WSC's config/i18n directory
 - Allows to customize all entries
- Customize WSC localizations using the Customization Service
 - Allows to customize (override or add) the following sections only:
 - SignatureImageRendering
 - transactionCodeConfigurations



Signing certificate filter

What is it and what is it for?

The organization administrator can now define filters on intended-use of certificates, for envelopes containing local certificate (SmartCard) signing experience. With that filter configured, the list of selectable certificates can be restricted to the certificates relevant on a local market (e.g. if a local signature smartcard contains two or more certificates for different purposes, like signing and identification).

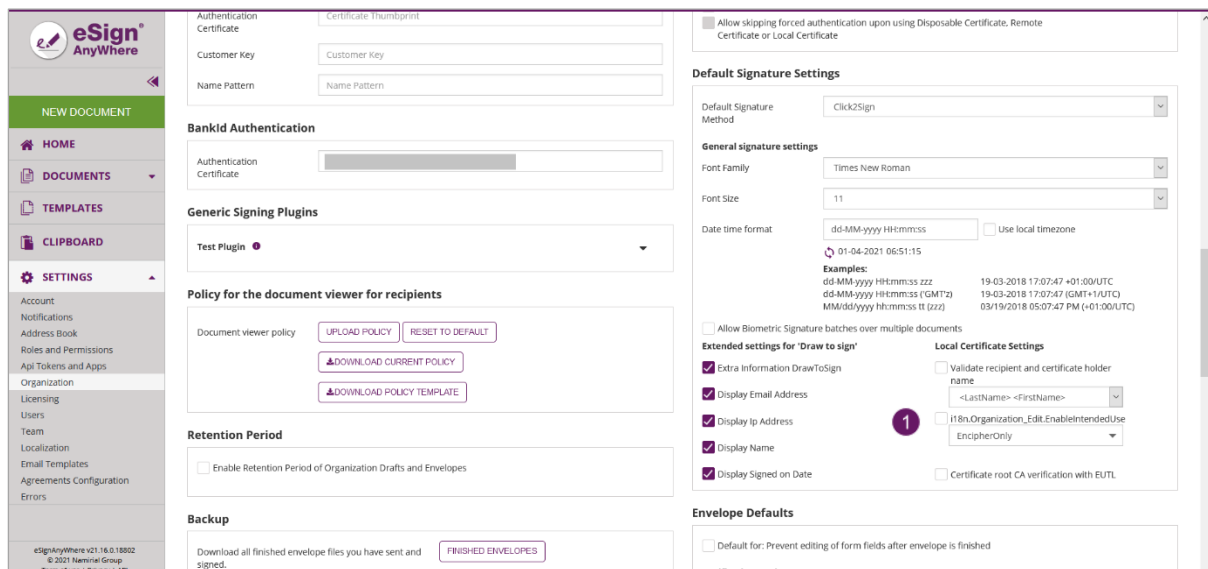
How does it work?

Please note: A local device driver is necessary for this feature.

PermittedKeyUsages

This allows to filter out certificates which are e.g. only meant for NonRepudiation or DigitalSignature etc.

You can find the signing certificate filter in the Organization settings in the section Default Signature Settings. The following figures show you where you can configure the filter in the UI:



1 Setting for the signing certificate filter



Remove REST API v1 and v2

With release 20.14 a year ago, the REST API Versions /v1 and /v2 have been marked as obsolete/deprecated and a migration guide has been published. With the 21.16 release, those versions have been removed:

- /api/v1 (also accessible via /api/v1.0)
- /api/v2 (also accessible via /api/v2.0)

The REST migration guide, which contains also some more information about the different API versions and in particular about the differences from version 1.0 or 2.0 to newer versions, is available here:

<https://www.esignanywhere.net/esignature-api/api-documentation/deprecated-api-versions/>

Please mind that the guide describes the migration to v4, but similar functionality will also be applicable for a migration to /v5. The swagger documentation, which is our REST API reference documentation, is available here:

<https://demo.esignanywhere.net/Api/swagger/ui/index>



MetaData for templates

What is it and what is it for?

When storing signed documents in a document management system (DMS), a tagging of the document(s) is common and mandatory to find the document again. While eSignAnyWhere already supported providing metadata in API integrations, older versions allowed the sender via WebUI to specify just one free-text metadata field with the recommendation to put an XML structure into it. Since 20.52, it is possible to integrate custom tagging implementations, which consider structures and allowed values predefined in a DMS. It allows organization admins to define a custom page, being presented to the sender before sending an envelope for signing. The UI of a metadata tagging form (or other before-send redirect page) can be aligned to the eSAW UI look and feel, or be aligned e.g. with your DMS. Consequently, the new change now allows defining metadata in templates. This can be used to set defaults, which are considered in a custom tagging page when a template was used to create the draft.

Beside DMS tagging, metadata can be used also to define other values necessary for post-processing by a callback handler. Any additional information/description can be added into the metadata section of an envelope.

How to activate

Following feature flag is necessary for this feature: *EnvelopeTemplates*

Following configuration in Settings-Organization is required: *Allow to set envelope meta data*

How does it work?

When creating or editing a template, the metadata text input field is displayed - similar to its presentation on a draft or envelope, when no before-send redirect (i.e. custom tagging page) is configured. You can find the section for the metadata on the "Create" page in the UI in the section 5 "Meta Data". Note: Metadata is saved in a template when save as template is used. The metadata will be copied to the draft if you create it from a template.

Please see the next figure for more information:



2. Documents

Drag & Drop files here

3. Recipients

Send finished documents to all signers and acknowledge recipients

4. Message

Subject: Please sign the enclosed envelope

Dear #RecipientFirstName# #RecipientLastName#

#PersonalMessage#

Please sign the envelope #EnvelopeName#

5. Meta Data

Enter Meta Data (optional)



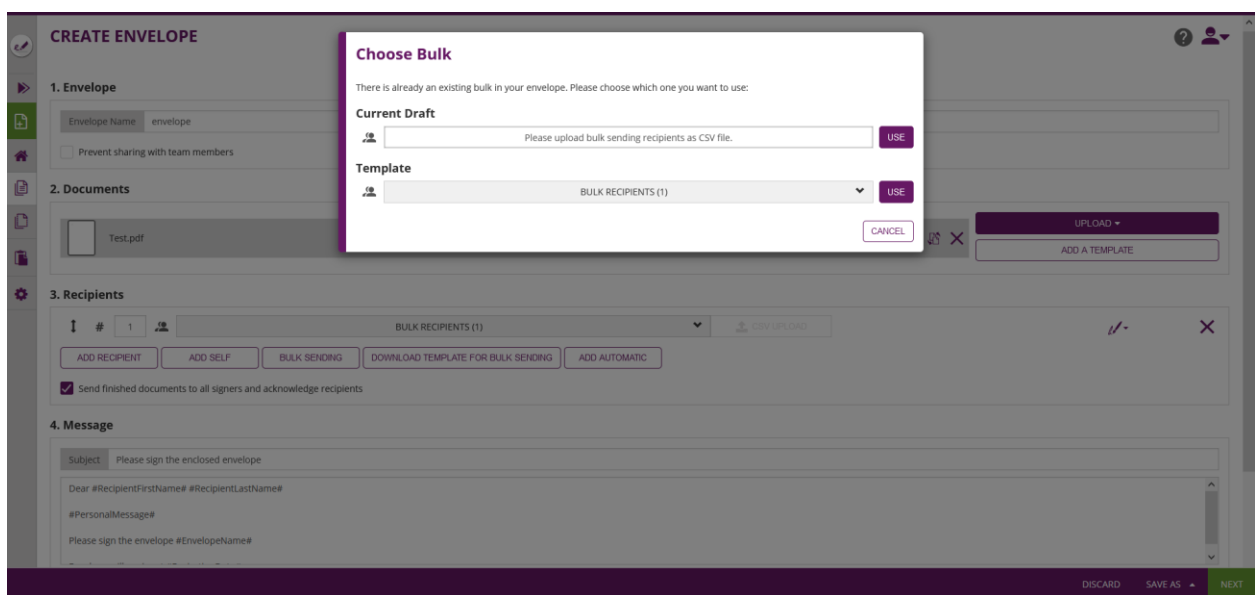
Merging a draft's bulk recipient list with template

What is it and what is it for?

A new dialog will let you choose between bulk recipients added in an envelope draft and bulk recipient list defined in a template. For example: If you add a template with a bulk to an envelope which already has a bulk included you will be asked if you want to continue with the bulk from the template or with the bulk from the envelope. This will help you with the process flow of bulk sending as you can now differentiate between the bulks of different envelopes.

How does it work?

After adding a template with a bulk to an envelope with a bulk you will see the following dialog:



There, just choose between the bulk of the template or the bulk in the current draft to continue.

For more information about the process please also see the following link (video):

<https://www.xyzmo.com/images/Resources/bulk-sending-with-eSignAnyWhere.mp4>



Stamp Imprint

What is it and what is it for?

With custom signature rendering layout configuration (stamp imprint configuration), an organization administrator can define how the stamp imprint on the signature image looks like (e.g. fonts, elements, layout etc). The new functionality allows to set organization wide background images (e.g. company logos) or define specific fonts for text added to the stamp imprint. While it has no impact on the legal levels of signatures (in EU, defined by eIDAS), a customer specific stamp imprint representation can create higher subjective trust and contract awareness of your customers.

How to activate

Following feature flag is necessary for this feature:

UseCustomStampImprintConfiguration

Please note the following: This is a non-default feature of eSignAnyWhere. If you are interested in this optional feature please [contact us](#).

How does it work?

If the feature is activated for your organization you can find the setting in the organization settings. Please see the next figure:



1 Custom Signature Rendering Configuration

In this section you can find the following settings:

- Upload your configuration
- Reset the signature rendering to default
- You can download the current configuration
- You can download the default template

If you download the default template (*SignatureRenderingLayouts.custom.xml*) you will see the following configuration:

```

<?xml version="1.0" encoding="utf-8"?>
<SignatureRenderingLayoutConfiguration>
  <Colors>
  </Colors>
  <Fonts>
  </Fonts>
  <Layouts>
  </Layouts>
</SignatureRenderingLayoutConfiguration>

```



Please note the following before you start to customize: The customized layouts, fonts and colors defined in the configuration file will override possibly existing elements of the default configuration. Also a combination of the two variants is possible. In that case the following hierarchy will be used:

1. Customized configuration from Customization Service
2. Customized local configuration
3. Default configuration

In the next section you can find possible configuration options.

Configuration options

Colors

Supported parameters:

- Parameter "a": Alpha
- Parameter "r": Red
- Parameter "g": Green
- Parameter "b": Blue

Fonts

Supported parameters:

- Parameter "family": Font family
- Parameter "size": Size (pt)
- Parameter "bold"
- Parameter "italic"
- Parameter "underline"
- Parameter "color": The id of the referenced color



Layouts

A layout consists of 1-n layers (e.g. to support background images). There are three different layer types supported:

- Layer
- ExternalImageLayer
- ImageLayer

Supported parameters:

- Parameter "backgroundColor": The id of the referenced color

Layer

A layer consists of 1-n rows. The rows of a layer are printed in the order defined in the configuration.

Supported parameters:

- Parameter "useTotalHeight": If set, the whole image height will be used for the imprint (75 % otherwise)
- Parameter "vAlign": Vertical alignment on signature imprint image
 - top, middle, bottom

Row

A row consists of 1-n cells. Supported parameters:

- Parameter "cellWidthMode"
 - "auto" or not defined: Width will be calculated automatically like in the old rendering
 - Aligned with neighbor rows with the same setting & number of cells
 - "fit-content": Width will be as required for the content (replaces "alignHeadings" from legacy rendering)



Cells

There are several different cell types supported:

- **Text cells**
 - TranslatedText: Translated text for captions (retrieved from WSC's translation file)
 - Additional parameter "key": References the corresponding translation unit in group "SignatureImageRendering"
 - Supports placeholders (see below)
 - TranslatedTextWithColon: Translated text for captions (retrieved from WSC's translation file) followed by a colon
 - Additional parameter "key": References the corresponding translation unit in group "SignatureImageRendering"
 - Supports placeholders (see below)
 - Text: Without translation, printed as-is
 - Supports placeholders (see below)

- **Data cells**
 - Signatory: The name of the signer or the subject of the signing certificate (e.g. local certificate signing)
 - Firstname: The first name of the signer as specified in the receiver information
 - Lastname: The last name of the signer as specified in the receiver information
 - FirstnameLastname: The first name & the last name (separated by a blank) of the signer as specified in the receiver information
 - LastnameFirstname: The last name & the first name (separated by a blank) of the signer as specified in the receiver information
 - CertificateCommonName: The subject of the signing certificate (e.g. local certificate signing)
 - Email: The email address of the signer
 - DateTime/DateTimeLocal: The signing date (local timezone)
 - DateTimeUtc: The signing date (UTC)
 - IpAddress: The ip address of the signer
 - PhoneNumber: The phone number of the signer
 - TransactionId: The transaction id (e.g. SMS-OTP signing)
 - TransactionToken: The transaction token (e.g. SMS-OTP signing)
 - Issuer: The issuer of the signing certificate (e.g. local certificate signing)



- PersonalNumber: The personal number of the signer (e.g. BankID)
- Metadata: Custom data
 - Uses XPath to select one element from
 - ◆ EnvelopeInformation
 - ◆ AdditionalClientWorkstepInformation
 - Additional parameter "xpath": To reference the desired node element
 - Additional parameter "attribute" (optional): To reference the desired node's attribute
 - Will print the value of the referenced node or attribute
- Custom
 - Can be used to add custom data to the signature image (currently only supported when signing with a Generic Signing Plugin)
 - Additional parameter "key": To reference the desired key/value pair
- **Additional cells**
 - Image
 - Content: Base64 encoded image
 - Additional parameter "height": To enable adjusting the size (in relation to the text size)
 - ◆ Supported units: pt
 - Image will not be scaled up (maximum size is the original image size)
 - Newline
 - Metadatalist
 - Uses XPath to select multiple nodes from
 - ◆ EnvelopeInformation
 - ◆ AdditionalClientWorkstepInformation
 - Iterates over the elements found

Supported parameters:

- Parameter "width"
 - Supported units: %
 - If set, the specified "cellWidthMode" of the row will be overruled
- Parameter "hAlign": Horizontal alignment
 - left, right, center
- Parameter "renderIfEmpty"
 - Cell will be skipped otherwise (like in old rendering)



- Allows more flexibility (e.g. table layout with two rows and two columns, but only the first row has a heading)

To allow even more flexibility, text cells (Text, TranslatedText, TranslatedTextWithColon) support the following placeholders which refer to the according data cell and will be replaced before rendering.

- ##Signatory##
- ##Firstname##
- ##Lastname##
- ##FirstnameLastname##
- ##LastnameFirstname##
- ##CertificateCommonName##
- ##Email##
- ##IpAddress##
- ##PhoneNumber##
- ##TransactionId##
- ##TransactionToken##
- ##Issuer##
- ##PersonalNumber##

Notes:

- In case a data value is not available (e.g. no phone number set) the placeholder will be simply removed (replaced by an empty string).
- In case an unknown placeholder (e.g. ##Phone##) is configured, this one will not be replaced or removed.
- Data cells which require additional parameters (e.g. DateTime, Metadata) are not supported.

ImageLayer

An image layer contains a single image only which can be positioned accordingly and might be used as a background image.

The image will not be scaled up (maximum size is the original image size)

Supported parameters:

- Content: Base64 encoded image
- Parameter "hAlign": Horizontal alignment on signature imprint image
 - left, center, right



- Parameter "height": Used to calculate the size on the signature imprint image
 - Can only be set in case "width" is not defined
 - Supported units: %
- Parameter "width": Used to calculate the size on the signature imprint image
 - Can only be set in case "height" is not defined
 - Supported units: %

ExternallImageLayer

An external image layer is used to define the layer where the additional (external) image for Click2Sign / Draw2Sign / Type2Sign should be rendered.

It must be defined in the layout for these three signature types.

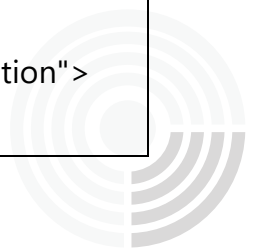
Please also see the next sample for more information about possible configurations:

```
<?xml version="1.0" encoding="utf-8"?>
<SignatureRenderingLayoutConfiguration>
  <Colors>
    <Color id="black" a="255" r="0" g="0" b="0" />
    <Color id="red" a="255" r="255" g="0" b="0" />
  </Colors>
  <Fonts>
    <Font id="default" family="Arial" size="10pt" bold="0" italic="0" underline="0"
color="black" />
    <Font id="bold" family="Arial" size="10pt" bold="1" italic="0" underline="0"
color="black" />
    <Font id="red" family="Arial" size="12pt" bold="1" italic="1" underline="1"
color="red"/>
  </Fonts>
  <Layouts>
    <Layout id="default">
      <ExternallImageLayer />
      <Layer>
        <Row>
          <TranslatedTextWithColon key="lbl_stamp_signatory" font="bold" />
          <Signatory font="default"/>
        </Row>
      </Layer>
    </Layout>
  </Layouts>
</SignatureRenderingLayoutConfiguration>
```

```

<Row>
  <TranslatedTextWithColon key="lbl_stamp_email" font="bold" />
  <Email font="default" />
</Row>
<Row>
  <TranslatedTextWithColon key="lbl_stamp_datetime" font="bold" />
  <DateTimeLocal format="dd-MM-yyyy HH:mm:ss (zzz)" font="default" />
</Row>
<Row>
  <TranslatedTextWithColon key="lbl_stamp_ipAddress" font="bold" />
  <IpAddress font="default"/>
</Row>
</Layer>
</Layout>
<Layout id="test" defaultFor="Click2Sign" backgroundColor="white">
  <ExternallImageLayer />
  <Layer useTotalHeight="1" vAlign="bottom">
    <Row cellWidthMode="auto">
      <TranslatedTextWithColon key="lbl_stamp_signatory" font="bold"
hAlign="right" />
      <Signatory font="default" />
    </Row>
    <Row cellWidthMode="auto">
      <TranslatedTextWithColon key="lbl_stamp_email" font="bold" hAlign="right" />
      <Email font="default" />
    </Row>
    <Row>
      <TranslatedTextWithColon key="lbl_stamp_datetime" font="bold"
hAlign="right" />
      <DateTime format="dd-MM-yyyy HH:mm:ss (zzz)" font="default" />
    </Row>
    <Row>
      <TranslatedTextWithColon key="lbl_stamp_ipAddress" font="bold"
hAlign="right" />
      <IpAddress font="default" renderIfEmpty="0"/>
    </Row>
    <MetadataList xPath="EnvelopeInformation/EnvelopeDocumentInformation">
      <Row>

```



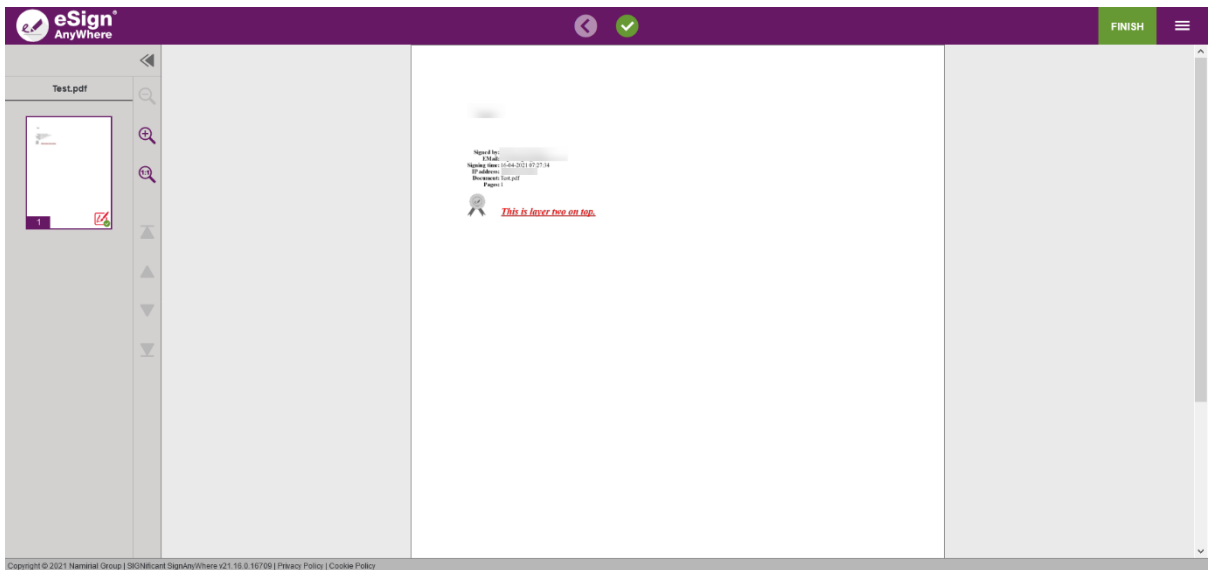
```

    <TranslatedTextWithColon key="lbl_stamp_documentName" font="bold"
hAlign="right" />
    <Metadata xPath="." attribute="name" font="default" />
</Row>
<Row>
    <TranslatedTextWithColon key="lbl_stamp_pageNumbers" font="bold"
hAlign="right" />
    <Metadata xPath="." attribute="numberOfPages" font="default" />
</Row>
</MetadataList>
<Newline/>
<Row>
    <!--Base64-->
    <Image height="100pt" hAlign="right"></Image>
</Row>
</Layer>
<Layer>
<Newline/>
<Row>
    <Text font="red" hAlign="center" width="100">This is layer two on top.</Text>
</Row>
</Layer>
</Layout>
</Layouts>
</SignatureRenderingLayoutConfiguration>

```

See also the result of the configuration above:





UI clean up: Hide non-accessible features

What is it and what is it for?

This feature clears up ambiguities in the UI. Only those features are displayed which are actually possible with the given settings. Non-accessible features will be hidden to optimize the UI experience.

How to activate

WscFeatureUpdateJob

Updates database table **Wcs Feature** periodically (configurable) with response from Wsc (GetServerInformation)

```
<configValues>
  ...
  <jobs>
    ...
    <WscFeatureUpdateJob>
      <runIntervalInMinutes>15</runIntervalInMinutes>
      <enabled>1</enabled>
    </WscFeatureUpdateJob>
    ...
  </jobs>
  ...
</configValues>
```

With the *Enabled* state and *LastUpdateTimeUtc*



User Session Permission Cache TTL

Permissions in the User session will be updated if any permissions have been changed.

```
<configValues>
...
<!-- user sessions permission cache time to live in minutes (-1 entries live forever, 0
reevaluated with every request, > 0 TTL in minutes) -->
  <userSessionPermissionCacheTimeToLiveInMinutes>15</userSessionPermissionCacheTimeToLive
InMinutes>
...
</configValues>
```

How does it work?

Password change

If the password gets changed in a user session, this and other sessions from this user will be ended.

Configured also with `<userSessionPermissionCacheTimeToLiveInMinutes>` node

Admin Web Feature Set

Admin Web FeatureSet will mark Features that are not usable because of one or more disabled WscFeature/s.



Generic Signing Plugin Improvements

What is it and what is it for?

The "Generic Signing Plugin" (GSP) allows implementation of custom 3rd party signature creation implementations (HSM based, web service based, etc). It is typically used to integrate external CAs into eSignAnyWhere. A GSP based implementation of a 3rd party CA is available for envelopes created via eSAW API or via eSAW WebUI. New features and improvements allow wider usage of the GSP.

How to activate

A GSP implementation has to be developed by our partner/customer. For development purposes, it can be tested on an environment where no Server Core license is installed. Before using in production environments, a GSP implementation must be code-signed by Namirial and installed on the server as defined in the "Generic Signing Plugin" How-To guide.

New features/improvements:

- Improved error handling
- Support for CMS (cryptographic message syntax) signatures (in combination with optional presence of certificate in prepare-step)
 - PrepareAsync has a new return type BasePrepareResult, that supports old PrepareResult and new CmsPrepareResult return types
 - PrepareAutomaticAsync has a new return type BaseAutomaticPrepareResult, that supports old AutomaticPrepareResult and new CmsAutomaticPrepareResult return types
 - CertificateCommonName and IssuerCommonName can be set in CmsPrepareResult and CmsAutomaticPrepareResult in order to add the correct name to the signature imprint image, when certificate is not available
- Allow passing of external (authentication) data in ExternalUri case via GSP callback receiver
 - Example added how the "state" URL parameter can be used for this use case
- Allow possibility to add custom values to signature imprint image via SignatureImageRenderingData in PrepareResult



For more information about the generic signing plugin in general please also see the following two links:

User Guide:

[https://www.esignanywhere.net/support/user-guide-esignanywhere-settings-customizing/#Generic Signing Plugin](https://www.esignanywhere.net/support/user-guide-esignanywhere-settings-customizing/#Generic_Signing_Plugin)

Signer Guide:

[https://www.esignanywhere.net/support/user-guide-signer-guide/#Generic Signing Plugin](https://www.esignanywhere.net/support/user-guide-signer-guide/#Generic_Signing_Plugin)

All information has been compiled to the best of our knowledge and belief. We exclude any liability due to incomplete, incorrect or outdated information. This document will be continuously revised and adapted to changes in legislation or case law, technology. We are pleased to receive any requests for clarification, updating and supplementation at any time via e-mail to contacts@namirial.com.

This document does not constitute legal advice. In particular, it cannot replace individual legal advice that takes into account the specifics of each case.

Version 1.0 | Author: Manuel Gierlinger | Last edited: 16.04.2021

