# PDF-AS 4.0 Documentation

EGIZ E-Government Innovationszentrum

## Package

# at.gv.egiz.pdfas.lib.api

# at.gv.egiz.pdfas.lib.api
# Class ByteArrayDataSink

```
java.lang.Object
   |
   +-at.gv.egiz.pdfas.lib.api.ByteArrayDataSink
```

**All Implemented Interfaces:**
DataSink

---

public class **ByteArrayDataSink**
extends Object
implements DataSink

A simple byte array data sink

---

## Field Summary

| | |
|---|---|
| protected | bos |

## Constructor Summary

| | |
|---|---|
| public | ByteArrayDataSink() |

## Method Summary

| | |
|---|---|
| OutputStream | createOutputStream() |
| byte[] | getData()<br>Returns the output data |

**Methods inherited from class java.lang.Object**

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

**Methods inherited from interface at.gv.egiz.pdfas.lib.api.DataSink**

createOutputStream

---

## Fields

### bos

```
protected java.io.ByteArrayOutputStream bos
```

## Constructors

(continued on next page)

(continued from last page)

### ByteArrayDataSink

```
public ByteArrayDataSink()
```

## Methods

### createOutputStream

```
public OutputStream createOutputStream()
```

---

### getData

```
public byte[] getData()
```

Returns the output data

**Returns:**
the output data

# at.gv.egiz.pdfas.lib.api
# Class ByteArrayDataSource

```
java.lang.Object
   |
   +-at.gv.egiz.pdfas.lib.api.ByteArrayDataSource
```

**All Implemented Interfaces:**
DataSource

---

public class **ByteArrayDataSource**
extends Object
implements DataSource

A simple byte array data source

---

## Constructor Summary

| | |
|---|---|
| public | ByteArrayDataSource(byte[] data) |

## Method Summary

| | |
|---|---|
| byte[] | getByteData() |
| String | getMIMEType() |

**Methods inherited from class** `java.lang.Object`

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

**Methods inherited from interface** `at.gv.egiz.pdfas.lib.api.DataSource`

`getByteData, getMIMEType`

---

## Constructors

### ByteArrayDataSource

```
public ByteArrayDataSource(byte[] data)
```

## Methods

### getMIMEType

```
public String getMIMEType()
```

## getByteData

```
public byte[] getByteData()
```

# at.gv.egiz.pdfas.lib.api
# Interface Configuration

public interface **Configuration**
extends

Configuration interface This interface is used to configure one PDF-AS run. It contains the configuration values from the configuration file. Use this interface to override properties during runtime.

## Method Summary

| | |
|---|---|
| `String` | `getValue(String key)`<br>Gets a specific Value |
| `boolean` | `hasValue(String key)`<br>Is the configuration key set |
| `void` | `setValue(String key, String value)`<br>Sets or overrides a configuration value |

## Methods

### getValue

```
public String getValue(String key)
```

Gets a specific Value

**Parameters:**
`key` - The configuration key

**Returns:**
The configured value

### hasValue

```
public boolean hasValue(String key)
```

Is the configuration key set

**Parameters:**
`key` - The configuration key

**Returns:**
true | false

### setValue

```
public void setValue(String key,
         String value)
```

Sets or overrides a configuration value

(continued from last page)

**Parameters:**

`key` - The configuration key
`value` - The configuration value

# at.gv.egiz.pdfas.lib.api
# Interface DataSink

**All Known Implementing Classes:**
ByteArrayDataSink

public interface **DataSink**
extends

Data Sink interface.

## Method Summary

| | |
|---|---|
| OutputStream | createOutputStream()<br>Creates an output stream to receive the data |

## Methods

### createOutputStream

```
public OutputStream createOutputStream()
```

Creates an output stream to receive the data

**Returns:**
an output stream for the data

# at.gv.egiz.pdfas.lib.api
# Interface DataSource

**All Known Implementing Classes:**
ByteArrayDataSource

public interface **DataSource**
extends

Data Source interface All data sources in PDF-AS implement this interface. Also custom data sources have to implement this interface to allow PDF-AS to use them.

## Method Summary

| | |
|---|---|
| `byte[]` | `getByteData()`<br>Gets the contained data |
| `String` | `getMIMEType()`<br>Gets the MIME Type of the contained data. |

## Methods

### getMIMEType

`public String getMIMEType()`

Gets the MIME Type of the contained data.

**Returns:**
MIME Type

### getByteData

`public byte[] getByteData()`

Gets the contained data

**Returns:**
the contained data

# at.gv.egiz.pdfas.lib.api
# Interface IConfigurationConstants

public interface **IConfigurationConstants**
extends

## Field Summary

| | |
|---|---|
| public static final | DEFAULT<br>Value: **default** |
| public static final | DEFAULT_SIGNATURE_PROFILE<br>Value: **sig_obj.type.default** |
| public static final | FALSE<br>Value: **false** |
| public static final | LEGACY_POSITIONING<br>Value: **.legacy.pos** |
| public static final | MAIN<br>Value: **main** |
| public static final | PLACEHOLDER_SEARCH_ENABLED<br>Value: **enable_placeholder_search** |
| public static final | POS<br>Value: **pos** |
| public static final | SEPERATOR<br>Value: **.** |
| public static final | SIG_OBJECT<br>Value: **sig_obj** |
| public static final | TABLE<br>Value: **table** |
| public static final | TRUE<br>Value: **true** |
| public static final | TYPE<br>Value: **type** |

(continued from last page)

# Fields

## TRUE

`public static final java.lang.String` **`TRUE`**

Constant value: **`true`**

## FALSE

`public static final java.lang.String` **`FALSE`**

Constant value: **`false`**

## SIG_OBJECT

`public static final java.lang.String` **`SIG_OBJECT`**

Constant value: **`sig_obj`**

## TYPE

`public static final java.lang.String` **`TYPE`**

Constant value: **`type`**

## TABLE

`public static final java.lang.String` **`TABLE`**

Constant value: **`table`**

## MAIN

`public static final java.lang.String` **`MAIN`**

Constant value: **`main`**

## POS

`public static final java.lang.String` **`POS`**

Constant value: **`pos`**

## DEFAULT

`public static final java.lang.String` **`DEFAULT`**

Constant value: **`default`**

(continued from last page)

## SEPERATOR

`public static final java.lang.String` **`SEPERATOR`**

Constant value: **`.`**

## LEGACY_POSITIONING

`public static final java.lang.String` **`LEGACY_POSITIONING`**

Constant value: **`.legacy.pos`**

## PLACEHOLDER_SEARCH_ENABLED

`public static final java.lang.String` **`PLACEHOLDER_SEARCH_ENABLED`**

Constant value: **`enable_placeholder_search`**

## DEFAULT_SIGNATURE_PROFILE

`public static final java.lang.String` **`DEFAULT_SIGNATURE_PROFILE`**

Constant value: **`sig_obj.type.default`**

# at.gv.egiz.pdfas.lib.api
# Interface PdfAs

public interface **PdfAs**
extends

## Method Summary

| | |
|---|---|
| SignResult | finishSign(StatusRequest statusRequest)<br>Finishes a signature process |
| Configuration | getConfiguration()<br>Gets a copy of the PDF-AS configuration, to allow the application to override configuration parameters at runtime. |
| StatusRequest | process(StatusRequest statusRequest)<br>Continues an ongoing signature process |
| SignResult | sign(SignParameter parameter)<br>Signs a PDF document using PDF-AS. |
| StatusRequest | startSign(SignParameter parameter)<br>Starts a signature process After the process has to be startet the status request has to be services by the user application |
| List | verify(VerifyParameter parameter)<br>Verifies a document with (potentially multiple) PDF-AS signatures. |

## Methods

### sign

```
public SignResult sign(SignParameter parameter)
  throws PdfAsException
```

Signs a PDF document using PDF-AS.

**Parameters:**
`parameter`

**Returns:**

### verify

```
public List verify(VerifyParameter parameter)
  throws PdfAsException
```

Verifies a document with (potentially multiple) PDF-AS signatures.

**Parameters:**
`parameter` - The verification parameter

(continued from last page)

**Returns:**
A list of verification Results

## getConfiguration

```
public Configuration getConfiguration()
```

Gets a copy of the PDF-AS configuration, to allow the application to override configuration parameters at runtime.

**Returns:**
A private copy of the pdf as configuration

## startSign

```
public StatusRequest startSign(SignParameter parameter)
  throws PdfAsException
```

Starts a signature process After the process has to be startet the status request has to be services by the user application

**Parameters:**
`parameter` - The sign parameter

**Returns:**
A status request

**Throws:**
`PdfAsException`

## process

```
public StatusRequest process(StatusRequest statusRequest)
  throws PdfAsException
```

Continues an ongoing signature process

**Parameters:**
`statusRequest` - The current status

**Returns:**
A status request

**Throws:**
`PdfAsException`

## finishSign

```
public SignResult finishSign(StatusRequest statusRequest)
  throws PdfAsException
```

Finishes a signature process

**Parameters:**
`statusRequest` - The current status

**Returns:**
A signature result

**Throws:**
`PdfAsException`

# at.gv.egiz.pdfas.lib.api
# Class PdfAsFactory

```
java.lang.Object
   |
   +-at.gv.egiz.pdfas.lib.api.PdfAsFactory
```

public class **PdfAsFactory**
extends Object

## Constructor Summary

| | |
|---|---|
| public | PdfAsFactory() |

## Method Summary

| | |
|---|---|
| static PdfAs | createPdfAs(File configuration)<br>Create a new instance of PDF-AS |
| static SignParameter | createSignParameter(Configuration configuration, DataSource dataSource)<br>Creates a sign parameter |
| static VerifyParameter | createVerifyParameter(Configuration configuration, DataSource dataSource)<br>Creates a verification parameter |
| static void | deployDefaultConfiguration(File targetDirectory)<br>Deploy default configuration to targetDirectory The targetDirectory will be deleted and |
| static void | dontConfigureLog4j() |
| static String | getSCMRevision()<br>Gets the PDF-AS SCM Revision |
| static String | getVersion()<br>Gets the PDF-AS Version |

**Methods inherited from class** `java.lang.Object`

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

## Constructors

### PdfAsFactory

`public` **`PdfAsFactory`**`()`

## Methods

(continued from last page)

## dontConfigureLog4j

```
public static void dontConfigureLog4j()
```

---

## createPdfAs

```
public static PdfAs createPdfAs(File configuration)
```

Create a new instance of PDF-AS

**Parameters:**
`configuration` - The PDF-AS configuration

**Returns:**

---

## createSignParameter

```
public static SignParameter createSignParameter(Configuration configuration,
        DataSource dataSource)
```

Creates a sign parameter

**Parameters:**
`configuration` - The configuration to be used
`dataSource` - The data source to be used

**Returns:**

---

## createVerifyParameter

```
public static VerifyParameter createVerifyParameter(Configuration configuration,
        DataSource dataSource)
```

Creates a verification parameter

**Parameters:**
`configuration` - The configuration to be used
`dataSource` - The data source to be used

**Returns:**

---

## deployDefaultConfiguration

```
public static void deployDefaultConfiguration(File targetDirectory)
  throws Exception
```

Deploy default configuration to targetDirectory The targetDirectory will be deleted and

**Parameters:**
`targetDirectory`

**Throws:**
`Exception`

## getSCMRevision

```
public static String getSCMRevision()
```

Gets the PDF-AS SCM Revision

**Returns:**

## getVersion

```
public static String getVersion()
```

Gets the PDF-AS Version

**Returns:**
PDF-AS Verison string

# at.gv.egiz.pdfas.lib.api
# Interface PdfAsParameter

**All Subinterfaces:**
VerifyParameter, SignParameter

public interface **PdfAsParameter**
extends

## Method Summary

| | |
|---|---|
| Configuration | getConfiguration()<br>Gets the configuration associated with the parameter |
| DataSource | getDataSource()<br>Gets the data source of the parameter |
| void | setConfiguration(Configuration configuration)<br>Sets the configuration associated with the parameter |
| void | setDataSource(DataSource dataSource)<br>Sets the data source of the parameter |

## Methods

### getConfiguration

`public Configuration getConfiguration()`

Gets the configuration associated with the parameter

**Returns:**

### setConfiguration

`public void setConfiguration(Configuration configuration)`

Sets the configuration associated with the parameter

**Parameters:**
`configuration`

### getDataSource

`public DataSource getDataSource()`

Gets the data source of the parameter

**Returns:**

## setDataSource

```
public void setDataSource(DataSource dataSource)
```

Sets the data source of the parameter

**Parameters:**

`dataSource`

# at.gv.egiz.pdfas.lib.api
# Interface SignaturePosition

public interface **SignaturePosition**
extends

## Method Summary

| | |
|---|---|
| `float` | `getHeight()`<br>Returns the height of the signature. |
| `int` | `getPage()`<br>Returns the page on which the signature was placed. |
| `float` | `getWidth()`<br>Returns the width of the signature. |
| `float` | `getX()`<br>Returns the x position. |
| `float` | `getY()`<br>Returns the y position. |

## Methods

### getPage

`public int getPage()`

Returns the page on which the signature was placed.

**Returns:**
Returns the page on which the signature was placed.

### getX

`public float getX()`

Returns the x position.

**Returns:**
Returns the x position.

### getY

`public float getY()`

Returns the y position.

**Returns:**
Returns the y position.

## getWidth

```
public float getWidth()
```

Returns the width of the signature.

**Returns:**
Returns the width of the signature.

## getHeight

```
public float getHeight()
```

Returns the height of the signature.

**Returns:**
Returns the height of the signature.

# at.gv.egiz.pdfas.lib.api
# Interface StatusRequest

public interface **StatusRequest**
extends

Status of a signture process

## Method Summary

| | |
|---|---|
| byte[] | getSignatureData()<br>Gets the data to be signed |
| int[] | getSignatureDataByteRange()<br>Gets the byte range of the data to be signed |
| boolean | isReady()<br>If true finishSign in PdfAs can be called to retrieve the signed pdf |
| boolean | needCertificate()<br>If true PDF-AS requires the signature certificate Retrieve the signing certificate and set it via setCertificate |
| boolean | needSignature()<br>If true PDF-AS requires a the CAdES signature use getSignatureData() and getSignatureDataByteRange() to retrieve the data to be signed and set the signature via setSigature |
| void | setCertificate(byte[] encodedCertificate)<br>Sets the signing certificate |
| void | setSigature(byte[] signatureValue)<br>Sets the signature |

## Methods

### needCertificate

`public boolean needCertificate()`

If true PDF-AS requires the signature certificate Retrieve the signing certificate and set it via setCertificate

**Returns:**

### needSignature

`public boolean needSignature()`

If true PDF-AS requires a the CAdES signature use getSignatureData() and getSignatureDataByteRange() to retrieve the data to be signed and set the signature via setSigature

**Returns:**

## isReady

```
public boolean isReady()
```

If true finishSign in PdfAs can be called to retrieve the signed pdf

**Returns:**

## getSignatureData

```
public byte[] getSignatureData()
```

Gets the data to be signed

**Returns:**

## getSignatureDataByteRange

```
public int[] getSignatureDataByteRange()
```

Gets the byte range of the data to be signed

**Returns:**

## setCertificate

```
public void setCertificate(byte[] encodedCertificate)
  throws java.security.cert.CertificateException
```

Sets the signing certificate

**Parameters:**
`encodedCertificate`

**Throws:**
`CertificateException`

## setSigature

```
public void setSigature(byte[] signatureValue)
```

Sets the signature

**Parameters:**
`signatureValue`

## Package

# at.gv.egiz.pdfas.lib.api.sign

# at.gv.egiz.pdfas.lib.api.sign
# Interface IPlainSigner

public interface **IPlainSigner**
extends

Signer interface PDF-AS uses an IPlainSigner instance to create the signature. Also custom IPlainSigner may be used to sign PDF-AS documents.

## Method Summary

| | |
|---|---|
| `X509Certificate` | `getCertificate()`<br>Gets the signing certificate |
| `String` | `getPDFFilter()`<br>Gets the PDF Filter for this signer |
| `String` | `getPDFSubFilter()`<br>Gets the PDF Subfilter for this signer |
| `byte[]` | `sign(byte[] input, int[] byteRange)`<br>Sign the document |

## Methods

### getCertificate

```
public X509Certificate getCertificate()
  throws PdfAsException
```

Gets the signing certificate

**Returns:**

**Throws:**
`PdfAsException`

### sign

```
public byte[] sign(byte[] input,
         int[] byteRange)
  throws PdfAsException
```

Sign the document

**Parameters:**
`input`
`byteRange`

**Returns:**

(continued from last page)

**Throws:**

`PdfAsException`

## getPDFSubFilter

`public String getPDFSubFilter()`

Gets the PDF Subfilter for this signer

**Returns:**

## getPDFFilter

`public String getPDFFilter()`

Gets the PDF Filter for this signer

**Returns:**

# at.gv.egiz.pdfas.lib.api.sign
# Interface SignParameter

**All Superinterfaces:**
PdfAsParameter

public interface **SignParameter**
extends PdfAsParameter

## Method Summary

| | |
|---|---|
| DataSink | getOutput()<br>Gets the data sink for the signature process |
| IPlainSigner | getPlainSigner()<br>Gets the signer to use. |
| String | getSignaturePosition()<br>Gets the signature position |
| String | getSignatureProfileId()<br>Gets the signature profile to use |
| void | setOutput(DataSink output)<br>Sets the data sink for the signature process |
| void | setPlainSigner(IPlainSigner signer)<br>Sets the signer to use |
| void | setSignaturePosition(String signaturePosition)<br>Sets the signature position |
| void | setSignatureProfileId(String signatureProfileId)<br>Sets the signature profile to use |

**Methods inherited from interface at.gv.egiz.pdfas.lib.api.PdfAsParameter**

getConfiguration, getDataSource, setConfiguration, setDataSource

## Methods

### getSignatureProfileId

`public String getSignatureProfileId()`

Gets the signature profile to use

**Returns:**

(continued from last page)

## setSignatureProfileId

`public void setSignatureProfileId(String signatureProfileId)`

Sets the signature profile to use

**Parameters:**
`signatureProfileId` - The signature profile

## getSignaturePosition

`public String getSignaturePosition()`

Gets the signature position

**Returns:**

## setSignaturePosition

`public void setSignaturePosition(String signaturePosition)`

Sets the signature position

**Parameters:**
`signaturePosition` - The signature position string

## setOutput

`public void setOutput(DataSink output)`

Sets the data sink for the signature process

**Parameters:**
`output`

## getOutput

`public DataSink getOutput()`

Gets the data sink for the signature process

**Returns:**

## setPlainSigner

`public void setPlainSigner(IPlainSigner signer)`

Sets the signer to use

**Parameters:**
`signer`

## getPlainSigner

`public IPlainSigner getPlainSigner()`

(continued from last page)

Gets the signer to use.

**Returns:**

# at.gv.egiz.pdfas.lib.api.sign
# Interface SignResult

public interface **SignResult**
extends

## Method Summary

| | |
|---|---|
| DataSink | getOutputDocument()<br>Returns the filled output data sink. |
| SignaturePosition | getSignaturePosition()<br>Returns the position where the signature is finally placed. |
| java.security.cert.X509Certificate | getSignerCertificate()<br>Returns the certificate of the signer. |

## Methods

### getOutputDocument

`public DataSink getOutputDocument()`

Returns the filled output data sink.

**Returns:**
Returns the filled output data sink.

### getSignerCertificate

`public java.security.cert.X509Certificate getSignerCertificate()`

Returns the certificate of the signer.

**Returns:**
Returns the certificate of the signer.

### getSignaturePosition

`public SignaturePosition getSignaturePosition()`

Returns the position where the signature is finally placed.

This information can be useful for post-processing the document.

Consult the PDF-AS documentation section Commandline for further information about positioning.

**Returns:**
Returns the position where the signature is finally placed. May return null if no position information is available.

## Package

# at.gv.egiz.pdfas.lib.api.verify

# at.gv.egiz.pdfas.lib.api.verify
# Interface SignatureCheck

public interface **SignatureCheck**
extends

## Method Summary

| | |
|---|---|
| int | getCode()<br>Returns the response code of the check. |
| String | getMessage()<br>Returns the textual response message of the check (corresponding to the code). |

## Methods

### getCode

`public int getCode()`

Returns the response code of the check.

**Returns:**
Returns the response code of the check.

### getMessage

`public String getMessage()`

Returns the textual response message of the check (corresponding to the code).

**Returns:**
Returns the textual response message of the check (corresponding to the code).

# at.gv.egiz.pdfas.lib.api.verify
# Interface VerifyParameter

**All Superinterfaces:**
PdfAsParameter

public interface **VerifyParameter**
extends PdfAsParameter

## Method Summary

| | |
|---|---|
| `Date` | `getVerificationTime()`<br>Gets the verification time |
| `int` | `getWhichSignature()`<br>Gets which signature should be verified This is a 0 based index of the signatures |
| `void` | `setVerificationTime(Date verificationTime)`<br>Sets the verification time. |
| `void` | `setWhichSignature(int which)`<br>Sets which signature should be verified This is a 0 based index of the signatures |

**Methods inherited from interface** `at.gv.egiz.pdfas.lib.api.PdfAsParameter`

`getConfiguration`, `getDataSource`, `setConfiguration`, `setDataSource`

## Methods

### getWhichSignature

`public int getWhichSignature()`

Gets which signature should be verified This is a 0 based index of the signatures

**Returns:**

### setWhichSignature

`public void setWhichSignature(int which)`

Sets which signature should be verified This is a 0 based index of the signatures

**Parameters:**
`which` - The index

### getVerificationTime

`public Date getVerificationTime()`

(continued from last page)

Gets the verification time

**Returns:**

---

## setVerificationTime

```
public void setVerificationTime(Date verificationTime)
```

Sets the verification time.

**Parameters:**

`verificationTime`

# at.gv.egiz.pdfas.lib.api.verify
# Interface VerifyResult

public interface **VerifyResult**
extends

## Method Summary

| | |
|---|---|
| SignatureCheck | getCertificateCheck()<br>Returns the result of the certificate check. |
| SignatureCheck | getManifestCheckCode()<br>Returns the result of the manifest check. |
| byte[] | getSignatureData()<br>Gets the signed data for the signature |
| X509Certificate | getSignerCertificate()<br>Gets the signer certificate |
| SignatureCheck | getValueCheckCode()<br>Returns the result of the value (and hash) check. |
| PdfAsException | getVerificationException()<br>Returns a verification exception if any. |
| boolean | isQualifiedCertificate()<br>Returns true, if the signer's certificate is a qualified certificate. |
| boolean | isVerificationDone()<br>Returns if the verification was possible or could not even be startet. |

## Methods

### isVerificationDone

`public boolean isVerificationDone()`

Returns if the verification was possible or could not even be startet. see `getVerificationException()` for details.

**Returns:**

### getVerificationException

`public PdfAsException getVerificationException()`

Returns a verification exception if any. Shows that the verification could not be started. See `isVerificationDone()`.

**Returns:**

## getCertificateCheck

`public SignatureCheck getCertificateCheck()`

Returns the result of the certificate check.

**Returns:**
Returns the result of the certificate check.

## getValueCheckCode

`public SignatureCheck getValueCheckCode()`

Returns the result of the value (and hash) check.

**Returns:**
Returns the result of the value (and hash) check.

## getManifestCheckCode

`public SignatureCheck getManifestCheckCode()`

Returns the result of the manifest check.

**Returns:**
Returns the result of the manifest check.

## isQualifiedCertificate

`public boolean isQualifiedCertificate()`

Returns true, if the signer's certificate is a qualified certificate.

**Returns:**
Returns true, if the signer's certificate is a qualified certificate.

## getSignerCertificate

`public X509Certificate getSignerCertificate()`

Gets the signer certificate

**Returns:**

## getSignatureData

`public byte[] getSignatureData()`

Gets the signed data for the signature

**Returns:**

# Index

## S



## T



## V