SetaPDF-Signer API

Manual and Reference

Version 1.5.1, 2010-03-25 16:58:13

Setasign - Jan Slabon
Major-Hirst-Straße 11
38442 Wolfsburg
Germany

http://www.setasign.de
support@setasign.de

Table of contents

SetaPDF-Signer API - Introduction

Digital signatures in PDF or any other document serves two purposes: The signature guarantees the document's integrity and it allows the recipient to identify the signer of a document.

The SetaPDF-Signer API allows PHP developers to digital sign PDF documents in pure PHP.

It's simple interface allows signing documents with just a few lines of php code.

To allow different signature modules in the future such modules are not part of the main API but are seperately written in own classes. Instances of this modules simply have to be passed to the sign()-method of the main API.

Currently there are only modules for OpenSSL available which leads into the fact that only certificate- and private key-formats could be used, which are supported by OpenSSL (see here).

Usage Demo

```
// require the SetaPDF_Signer class
require_once("Signer/SetaPDF_Signer.php");
// require the OpenSSL signing module
require_once("Signer/Module/OpenSsl.php");
// create an instance
$module = new SetaPDF_Signer_Module_OpenSsl();
// set the properties of the module
$module->setSignCert(file_get_contents('certificate.pem'));
$module->setPrivateKey(array(file_get_contents('private-key.pem'),
'password'));
// create a SetaPDF_Signer object
$signer =& SetaPDF_Signer::factory('pdfdocument.pdf');
// set some signature properties
$signer->setReason("Just for testing");
$signer->setLocation('www.setasign.de');
$signer->setContactInfo('+49 5361 3099400');
// sign the document and send it to the client
$res = $signer->sign($module, 'signed.pdf', 'I');
if (SetaPDF::isError($res)) // check for errors
    var_dump($res);
```

SetaPDF-Signer API - System Requirements

All SetaPDF APIs are written in pure PHP and does not need any other libraries installed except a PHP environment of a version later than 4.3 and an installed Zend Optimizer or installed Ioncube loader.

The SetaPDF-Signer API currently makes use of the OpenSSL Library. It is possible to use the OpenSSL Library through the command line or through the php build in OpenSSL functions. At least it requires OpenSSL to be compiled with PHP or the access to the openssl binary through a exec()-call.

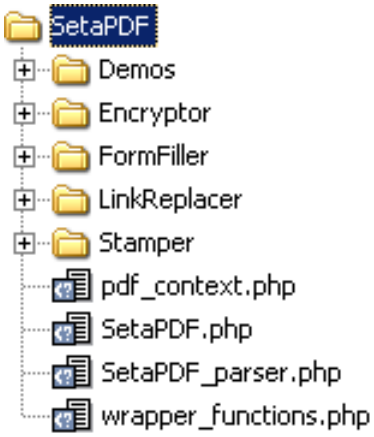
To deal with temporary files the SetaPDF-Signer API also needs write and read access to a directory for temporary files.

SetaPDF-Signer API - Installation

The SetaPDF API collection includes a directory structure which should be kept, because of the internal usage of pathes.

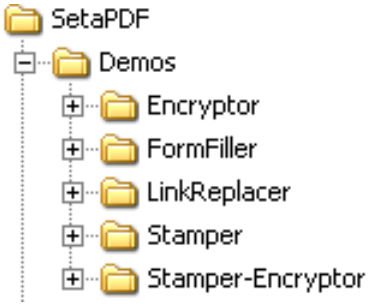
Files and directories

All packages includes a root directory called *SetaPDF*. In this directory you'll find the desired API directories. The directory structure for all current available SetaPDF APIs looks like this:



If you transfer the files via FTP
make sure you use binary mode.

For each API or API combination you'll find demo files in the /Demo directory in nearly the same structure:



To use one of the SetaPDF APIs in your applications you have to add the SetaPDF-directory to your include_path:

```
set_include_path(get_include_path() . PATH_SEPARATOR . 'PathTo/SetaPDF/');
```

Now you can simply include the SetaPDF-Signer API with the following lines:

```
require_once("Signer/SetaPDF_Signer.php");
```

```
// load the php build in OpenSSL module
require_once("Signer/Modul/OpenSsl.php");
// load the command line OpenSSL module
require_once("Signer/Modul/OpenSslCli.php");
```

SetaPDF-Signer API - Ioncube encoded package

If you own a package of the API, which is encoded with Ioncube you need a loader installed on your server. There are 2 ways to get ioncube encoded files to run:

1. Install the loader in your php.ini
2. Load the loader at runtime

For details how to install ioncube or simply to check if it is installed, just download the loaders from http://www.ioncube.com/loaders.php, extract its content to /SetaPDF/ioncube and open the file ioncube-loader-helper.php in the directory /SetaPDF/ioncube in your webbrowser and follow the instructions. For further instructions go to http://www.ioncube.com/

Licensing with Ioncube

Each ioncubed package needs a valid license to run. The provided licensefiles for the SetaPDF API are named: **.htSetaPDF-<API-NAME>.icl**

You don't have to rename that file, because the package search for exactly that named file in one of its upper directories. All APIs first searches for this file in the their initial directory. F.g. The SetaPDF-Encryptor API searches first in /SetaPDF/Encryptor/. If the licensefile is not found it goes one directory upwards: /SetaPDF/ and so on...

Please notice that the filename is prefixed with .ht. Some systems hide such prefixed files automatically.

SetaPDF-Signer API - Zend encoded package

If you own a package, which is encoded with the Zend Safguard Suite you have to install the Zend Optimizer. For more information please go to http://www.zend.com/products/zend_optimizer.

Licensing with Zend

Also the zend encoded packages need valid licenses to run. The provided licensefiles for the SetaPDF API are named:
.htSetaPDF-<API-NAME>.zl

Please notice that the filename is also prefixed with .ht. Some systems hide such prefixed files automatically.

For zend encoded packages the name of the license file can be changed and has no real meaning. You can load the licensefile dynamically on runtime in your php script (zend_loader_install_license()) or by default in your php.ini.

Zended packages are only available for development- and serverlicenses.

SetaPDF-Signer API - Constants / Configuration

The API needs some constants which are hard coded into the API.

Also you can define global variables which affects the behaviour of specific tasks.

Global Configuration Variables

$GLOBALS['SETAPDF_PARSE_INVALID_FILES'] (boolean)

(**DEPRECATED**) If this global variable is set the pdf parser tries to read/repair invalid PDF documents. This setting could affect the processtime on huge files very much.

This variable isn't used by the parser as of version 1.3 (all current version of the SetaPDF APIs)

$GLOBALS['SETAPDF_SEARCH_FOR_XREF_OFFSET'] (integer)

With this global variable you can adjust the offset position from which the pdf parser should search for the pointer to the xref table. If not defined the default value of 1500 is used.

The pdf specification says it has to be in the last 1024 bytes of a file. But sometimes there are errorious document in the wild that have some garbage at the end so we need the possibility to do a kind of finetuning for them.

Predefined Version Constants

The following constants defines the versions of specific files of the SetaPDF core:

SETAPDF_CORE_VERSION (string)1.3

Version of the abstract SetaPDF class. Defined in /SetaPDF/SetaPDF.php

SETAPDF_PARSER_VERSION (string)1.3

Version of the SetaPDF_Parser class. Defined in /SetaPDF/SetaPDF_parser.php

SETAPDF_PDF_CONTEXT_VERSION (string)1.3

Version of the pdf_context class. Defined in /SetaPDF/pdf_context.php

SETAPDF_WRAPPER_FUNCTIONS_VERSION (string)1.2.1

Version of the wrapper functions file. Defined in /SetaPDF/wrapper_functions.php

Predefined Constants

SETAPDF_SIG_NOT_CERTIFIED (integer)0

Don't certify the document.

SETAPDF_SIG_CERTIFIED_NO_CHANGES_ALLOWED (integer)1

No changes allowed

SETAPDF_SIG_CERTIFIED_FORM_FILLING (integer)2

Form fill-in only allowed

SETAPDF_SIG_CERTIFIED_FORM_FILLING_AND_ANNOTATIONS (integer)3

Form fill-in and commenting allowed

Predefined Constants for Errorhandling

Possible errorcodes for the SetaPDF main class starts at 1 and ends at 99.

E_SETAPDF_CANNOT_OPEN_FILE (integer)1

cannot open XXXX !

E_SETAPDF_UNABLE_TO_POINT_TO_XREF_TABLE (integer)2

Unable to find pointer to xref table

E_SETAPDF_UNABLE_TO_FIND_XREF (integer)3

Unable to find xref table - Maybe a Problem with 'auto_detect_line_endings'

E_SETAPDF_UNEXPECTED_HEADER_IN_XREF_TABLE (integer)4

Unexpected header in xref table

E_SETAPDF_UNEXPECTED_DATA_IN_XREF_TABLE (integer)5

Unexpected data in xref table

E_SETAPDF_FILE_IS_ENCRYPTED (integer)6

File is encrypted!

E_SETAPDF_WRONG_TYPE (integer)7

Wrong Type of Element

E_SETAPDF_UNABLE_TO_FIND_OBJECT (integer)8

Unable to find object at expected location

E_SETAPDF_ENC_UNSUPPORTED_FILTER (integer)9

E_SETAPDF_ENC_UNSUPPORTED_ALGO (integer)10

E_SETAPDF_ENC_UNSUPPORTED_REVISION (integer)11

E_SETAPDF_ENC_NO_RIGHTS_FOR_SPECIFIC_ACTION (integer)12

E_SETAPDF_ENC_WRONG_OWNER_PW (integer)13

E_SETAPDF_CANNOT_COPY_FILE (integer)14

Cannot copy file XXXX to YYYY

E_SETAPDF_HEADER_ALREADY_SEND (integer)15

Some data has already been output to browser, can't send PDF file

E_SETAPDF_UNABLE_TO_FIND_TRAILER (integer)16

Trailer keyword not found after xref table

E_SETAPDF_UNSUPPORTED_FILTER (integer)17

An unsupported compression filter is required.

E_SETAPDF_ZLIB_REQUIRED (integer)18

To handle /FlateDecode filter, php with zlib support is needed.

E_SETAPDF_DECOMPRESSION_ERROR (integer)19

Error while decompressing stream.

E_SETAPDF_UNABLE_TO_CREATE_CACHE_DIR (integer)20

Unable to create directories in cache directory.

API Related Predefined Constants for Errorhandling

Possible errorcodes for the SetaPDF-Signer API starts at 600 and ends at 699.

E_SETAPDF_SIG_UNKNOWN_PROPERTY (integer)600

You tried to set an unknown signature property in SetaPDF_Signer::setSignatureProperty().

E_SETAPDF_SIG_TMPDIR_DOES_NOT_EXISTS (integer)601

Fallback temporary directory _tmp/ does not exists. This error occurs if you didn't set an own temporary directory in the factory method and the fallback directory doesn't exists.

E_SETAPDF_SIG_DIR_DOES_NOT_EXISTS (integer)602

Will be thrown if a directory doesn't exists.

E_SETAPDF_SIG_FIELD_NAME_TO_SHORT (integer)607

The string length of the fieldname have to be greater than zero.

E_SETAPDF_SIG_WRONG_MODULE_CLASS (integer)608

Passed module isn't an instance/subclass of SetaPDF_Signer_Module.

E_SETAPDF_SIG_NEEDAPPEARANCES_FLAG_FOUND (integer)609

The document includes a NeedAppearances-Flag (see SetaPDF_Signer::setRemoveNeedAppearancesFlag()).

E_SETAPDF_SIG_SIGNATURE_LENGTH_TO_LARGE (integer)611

The signature length (?? bytes) doesn't fit into the reserved space (?? bytes) and "$this->allowSignatureContentLengthChange" is set to false. (see SetaPDF_Signer::setAllowSignatureContentLengthChange())

E_SETAPDF_SIG_PARSER_NO_KIDS (integer)612

Cannot find /Kids in current /Page-Dictionary

E_SETAPDF_SIG_ALREADY_CERTIFIED (integer)613

Document is already certified. Only approval signatures are possible

E_SETAPDF_SIG_PARSER_WRONG_PAGE_NO (integer)614

Wrong page number given

E_SETAPDF_SIG_NO_BOX_FOUND (integer)615

Page box not found

E_SETAPDF_SIG_MOD_OPENSSL_NOT_AVAILABLE (integer)610

This module requires php compiled with openssl.

E_SETAPDF_SIG_MOD_OPENSSL_ERROR (integer)603

An error from OpenSSL occurs. If you've set the track_errors directive to "On" the message will include a detailed error message.

E_SETAPDF_SIG_MOD_OPENSSL_BOUNDARY_ID_NOT_FOUND (integer)604

Will occur if the boundary id in a created smime message is not found.

E_SETAPDF_SIG_MOD_OPENSSL_SIG_EXTRACTION (integer)605

An error occurs while extracting the signature of the smime message.

E_SETAPDF_SIG_MOD_OPENSSLCLI_ERROR (integer)606

The commandline call of OpenSSL failed.

E_SETAPDF_SIG_CURL_ERROR (integer)616

curl_exec() returned false. The detailed error message is included in the error object.

E_SETAPDF_SIG_CURL_RES_ERROR (integer)617

The time stamp server responses with another status code than 200.

E_SETAPDF_SIG_TS_ERROR (integer)618

Time stamp server returned status other than 0 (zero). More details see: RFC3161 (2.4.2.)

Constans for Cache Mechanism

The following constants are used to control the behaviour of the caching mechanism of the pdf parser.

SETAPDF_P_CACHE_NO (integer)0x00

Don't read and write cache.

SETAPDF_P_CACHE_READ_XREF (integer)0x01

Try to read the cached xref table.

SETAPDF_P_CACHE_WRITE_XREF (integer)0x02

Write the xref table to cache.

SETAPDF_P_CACHE_XREF (integer)0x01 | 0x02

Try to read and write the xref table.

SETAPDF_P_CACHE_READ_OBJECTS (integer)0x04

Try to read cached objects.

SETAPDF_P_CACHE_WRITE_OBJECTS (integer)0x08

Write read objects to cache.

SETAPDF_P_CACHE_OBJECTS (integer)0x04 | 0x08

Try to read and write objects to cache.

SETAPDF_P_CACHE_ALL (integer)0x01 | 0x02 | 0x04 | 0x08

Read and write objects and xref-tables.

SetaPDF-Signer API - Caching

PDF parsing and handling can be an expensive task in view of needed cpu-power.

To avoid doing default tasks for a single document a few times the parser class offers a caching mechanism to reduce the overhead and avoid reparsing of PDF documents a few times.

The parser simply saves serialized data in the filesystem and load them back if needed. This data can be used with ANY SetaPDF API. So if for example the SetaPDF-Merger API creates the cache data, the SetaPDF-Stamper API can benefit from them.

As of this, the handling of the cache mechanism is done through static methods of the SetaPDF_Parser class. Calls to this methods will change static variables in their method contexts, so that changes doesn't depend on the object instance but applies to all instances of a parser object. (We used static variable because of compatibility to PHP4)

There are 2 parts that the parser can cache:

1. The Xref Table

This is a kind of table of contents of a PDF document. It includes information about all objects in a document and their byte-offset positions in the document. Often documents include several hundreds or thousands of entries in that table. Further more a PDF document can include more than one xref table, which relys on several updates of a document (incremental updates). But at least all tables have to be processed to get the final state of the document... By caching that data, the parser don't have to reparse the xref table out of the document.

2. Objects

Each entry in the above described xref table points to an object representing specific data, like Images, Fonts, Pages,... If the parser should read such an object it have to go to the desired byte-offset position in the document, known from the xref-table, and have to parse the object token-wise. This process needs several string comparsions and also runs recursive until the object is totally read.

The parser can cache the read objects and use the cached versions at the next situation when it is needed. No byte-position change or parsing of any string is done but simply unserializing the data from the cached data.

Usage

As already written the handling of the cache functionallity is done by static methods of the SetaPDF_Parser class.

You can use the static method right after including a desired API like the SetaPDF-Merger API:

```
require_once('Merger/SetaPDF_Merger.php');
// at this point you can access the SetaPDF_Parser class
```

First of all you have to tell the API where you would like to save the cached data. You have to use the SetaPDF_Parser::cacheDir()-method for this:

```
SetaPDF_Parser::cacheDir(realpath('../../path/for/cached/data/'));
```

Now you were able to activate the caching by calling the SetaPDF_Parser::cacheFlags()-method with special flags. The flags are predefined in Constants:

```
// Will read and write all data (xref table and objects) from/to cache.
SetaPDF_Parser::cacheFlags(SETAPDF_P_CACHE_ALL);
// Will just read and write the xref table from/to cache.
SetaPDF_Parser::cacheFlags(SETAPDF_P_CACHE_XREF);
// Will just read and write objects from/to cache.
SetaPDF_Parser::cacheFlags(SETAPDF_P_CACHE_OBJECTS);
```

After this the cache is active for all instances of any SetaPDF API.

Furthermore you can do some fintuning:

Build the cache slowly

If you want the cache to be build piecemeal you can use the SetaPDF_Parser::cacheNoOfObjectsPerInstance()-method to define a maximum of objects to cache in a single script instance. With this method you can avoid performance peaks because the cache writing process, for sure, also needs cpu time.

```
// cache a maximum of 100 objects per script instance
SetaPDF_Parser::cacheNoOfObjectsPerInstance(100);
```

How is a file identified and how you can control it

By default the cache mechanism uses the md5_file()-function to get an unique file identifier of the document. This file identifier is used as the directoryname in the cache output directory. To give you the possibility to use another method for the fileidentification you can define your own function/method, which will be called when a fileidentifier is needed, with the SetaPDF_Parser::cacheHashFunction()-method.

An Example: You already have your documents arranged in a database. This data have already unique ids related to the documents local path in your filesystem. As the ids are already known and are unique you should use the ids as a fileidentifier to avoid creating a hash with md5_file().

Furthermore it is easier for you to manage the cache data, as you can for example delete the cache data if the data in the database table were deleted or changed.

The passed argument is of the pseudo-type callback and will be used with call_user_func()-function.

```
function mapFilenameToId($filename) {
    // just pseudo code
    $db = YourDbClass::getInstance();
```

```
    $id = $db->getOne("SELECT id FROM documents WHERE filename =
".$db->quote($filename));
    return $id;
}
SetaPDF_Parser::cacheHashFunction('mapFilenameToId');
```

SetaPDF - Class

This class is the base class for nearly all SetaPDF APIs. It offers some public static helper methods.

Class Overview

SetaPDF

Child Classes

- SetaPDF_Signer

Methods

- SetaPDF::isError()

SetaPDF::isError()

Description

```
SetaPDF {
        boolean isError ( mixed $obj )
}
```

Determines if a variable is a SetaPDF_Error object.

Parameters

$obj

Variable to check

Return Values

True if *$obj* is a SetaPDF_Error object

SetaPDF_Error - Class

This class represents an error object thrown by a SetaPDF API. You can get more information about the error by checking the following properties $obj->message and $obj->code.

You can add your own error handling by defining your own class named SetaPDF_Error before you include any SetaPDF-File. The original class looks like this:

```
class SetaPDF_Error {
    var $message;
    var $code;

    function SetaPDF_Error($message = 'unknown error', $code = null,
                        $mode = null, $options = null, $userinfo = null) {
        $this->message = $message;
        $this->code = $code;
    }
}
```

SetaPDF_Parser - Class

The SetaPDF_Parser class is the base class for all indvidual SetaPDF parser classes. It is for example responsible for reading the xref table or objects of a document.

The SetaPDF_Parser class is an abstract class and *just* offers some static methods which let you control the cache functionality.

Class Overview

SetaPDF_Parser

Methods

SetaPDF_Parser::cacheDir()

Description

```
SetaPDF_Parser {
        mixed cacheDir ( [string $dir=null] )
}
```

Sets the directory for cache data.

This method should be called static.

Parameters

$dir

Path to the directory where to write the cache data. If *null* the directory will not be changed.

Return Values

The actual path.

SetaPDF_Parser::cacheFlags()

Description

```
SetaPDF_Parser {
        mixed cacheFlags ( [string $flags=null] )
}
```

Sets the flags how the parser should handle read and write processes of objects or xref-tables.

This method should be called static.

You can use this flags to do fine tuning of the caching mechanism. The flags can be combined using a bitwise AND (|) operation.

If any flag is set, except *SETAPDF_P_CACHE_NO,* a valid writeable path should be set with SetaPDF_Parser::cacheDir().

Parameters

$flags

The parameter defines the caching behaviour of the API. Available values are:

- *SETAPDF_P_CACHE_NO* - Don't read and write cache.
- *SETAPDF_P_CACHE_READ_XREF* - Try to read the cached xref table.
- *SETAPDF_P_CACHE_WRITE_XREF* - Write the xref table to cache.
- *SETAPDF_P_CACHE_XREF* - Try to read and write the xref table.
- *SETAPDF_P_CACHE_READ_OBJECTS* - Try to read cached objects.
- *SETAPDF_P_CACHE_WRITE_OBJECTS* - Write read objects to cache.
- *SETAPDF_P_CACHE_OBJECTS* - Try to read and write objects to cache.
- *SETAPDF_P_CACHE_ALL* - Read and write objects and xref-tables.

Return Values (see also Constants / Configurations)

The actual value.

SetaPDF_Parser::cacheMkdirMode()

Description

```
SetaPDF_Parser {
        mixed cacheMkdirMode ( [integer $mode=null] )
}
```

As the caching mechanism creates directories for each pdf document the API internally uses mkdir to create the directory. With this method you can define if and which parameter should be passed as the $mode parameter of the mkdir-function.

This method should be called static.

Parameters

$mode

The file mode.

The parameter consists of three octal number components specifying access restrictions for the owner, the user group in which the owner is in, and to everybody else in this order. More informations about the mode-parameter can be found here.

Return Values

The actual value.

SetaPDF_Parser::cacheNoOfObjectsPerInstance()

Description

```
SetaPDF_Parser {
        mixed cacheNoOfObjectsPerInstance ( [integer $no=null] )
}
```

For sure a caching process needs more process power as the cached data have to be written to the file system. Often a PDF document is build with more houndres or thousands of objects which can increase the process time to a bad value.

With this method you can define how many maximum objects should be cached per script instance. So you can chop the cache creation over several script executions.

This method should be called static.

If you set the $no-parameter, for example, to 100, the parser will cache 100 objects per script instance maximum, until all objects are cached.

By default the parser will cache ALL objects.

Parameters

$no

The maximum number of objects to cache per instance.

Return Values

The actual value.

SetaPDF_Parser::cacheHashFunction()

Description

```
SetaPDF_Parser {
        mixed cacheHashFunction ( [callback $hashFunction=null] )
}
```

To identify a pdf document the API uses the md5_file()-function by default.

If you want to create your own identification process or if you already know a hash or unique property of the document you can use this method to define an own function/method which will be called when the parser needs the hash.

This hash/value will be used as the directory name in the cache directory (see SetaPDF_Parser::cacheDir()).

The given value will be used as the function parameter of a call_user_func()-call.

This method should be called static.

Parameters

$hashFunction

The function to be called.
(See also informations about the callback type.)

Return Values

The actual value.

SetaPDF_Signer - Main class

This is the main class of the SetaPDF-Signer API.

Class Overview

File:

SetaPDF
└─ SetaPDF_Signer

Methods

Inherited Methods

Class: SetaPDF

- SetaPDF::isError()

SetaPDF_Signer::factory()

Description

```
SetaPDF_Signer extends SetaPDF {
        factory ( string $sourceFileName[, string $tmpDirectory=null] )
}
```

This method has to be called static and will return an instance of the SetaPDF_Signer class.

The SetaPDF_Signer class also can be initiated without the factory method in the normal PHP style:

$signer = new SetaPDF_Signer();

In this case you have to set the *$sourceFileName* with SetaPDF_Signer::setSourceFileName()-Method.

Parameters

$sourceFileName

A string that defines the path (relative or absolute) to the original document. Only local paths are allowed.

$tmpDirectory

A path for temporary files. If not or *null* is passed the default fallback directory SetaPDF/Signer/_tmp/ is used.

Return Values

A new instance of the SetaPDF_Signer class or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::setSourceFileName()

Description

```
SetaPDF_Signer extends SetaPDF {
        void setSourceFileName ( string $sourceFileName )
}
```

This method sets the documents filename, that should be signed.

Parameters

$sourceFileName

A string that defines the path (relative or absolute) to the original document. Only local paths are allowed.

SetaPDF_Signer::setAppearance()

Description

```
SetaPDF_Signer extends SetaPDF {
        void setAppearance ( string $fileName[, mixed $x|$position=0[, mixed
        $y|$translate=0[, mixed $w=null[, mixed $h=null[, integer $pageNo=1]]]]]
        )
}
```

With this method you can add a visible signature appearance to the document.

The size can be specified in different ways:

- explicit width (w) and height (h)
- one explicit dimension (w or h), the other being calculated automatically in order to keep the original proportions
- no explicit dimension, the size of the original document is taken

Parameters

$fileName

A string that defines the path (relative or absolute) to a PDF file that represents the appearance as the first page. Only local paths are allowed.

$x|$position

The position on the abscissa.

OR

A position in the following format:

- LT = left top
- LM = left middle
- LB = left bottom
- CT = center top
- CM = center middle
- CB = center bottom
- RT = right top
- RM = right middle
- RB = right bottom

$y|$translate

The position on the ordinate.

OR

An array with x- and/or y-values for a translation based on the position defined in the previous parameter:

```
array(
  'x' => 20,
  'y' => -10
)
```

$w

The width in points.

$h

The height in points.

$pageNo

The page on which the appearance should appear.

Version

since 1.4

SetaPDF_Signer::setTmpDirectory()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed setTmpDirectory ( [string $tmpDirectory=null] )
}
```

If you want to change the path for temporary files at runtime, you can use this method.

Parameters

$tmpDirectory

A path for temporary files. If not or *null* is passed the default fallback directory SetaPDF/Signer/_tmp/ is used.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::setFieldName()

Description

```
SetaPDF_Signer extends SetaPDF {
        void setFieldName ( string $fieldName )
}
```

With this method you can define the name of the signature field added to the PDF document.

If a form field with the same name already exists in the document you want to sign the name of the signature will be suffixed with a number. For example if a field named "Signature" already exists the signature fields name becomes: "Signature0" or if this also exists "Signature1",...

If you're using a development license the signature
fields name is fixed and cannot be changed!

Parameters

$fieldName

The signature field name.

The default value is: "Signature"

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::setSignatureContentMinLength()

Description

```
SetaPDF_Signer extends SetaPDF {
        void setSignatureContentMinLength ( integer $length )
}
```

As a signatures content various in length it is neccesary to reserve a defined amount of space in the document.

This space is defined with 3500 bytes by default.

If the API recognize that a signature is bigger than the reserved space the signing process will start again, but it automatically sets the signatureContentMinLength-value to the needed length before.

If you already know that your signature is very huge you can adjust the reserved space with this method.

You also can set this value to a very smal amount, to force the API to create the signature twice, so it'll fit exactly in the reserved space and as less as possible bytes were used.

Parameters

$length

The length of the reserved bytes in the document.

SetaPDF_Signer::setAllowSignatureContentLengthChange()

Description

```
SetaPDF_Signer extends SetaPDF {
        void setAllowSignatureContentLengthChange ( [boolean
        $allowSignatureContentLengthChange=true] )
}
```

This method can be used to stop the API from recreating signatures if it's content doesn't fit into the reserved space. (see SetaPDF_Signer::setSignatureContentMinLength())

Parameters

$allowSignatureContentLengthChange

A boolean value defining if the API should recreate the signature if the resulting signature content is to large as the reserved space (*true*) or not (*false*).

SetaPDF_Signer::setRemoveNeedAppearancesFlag()

Description

```
SetaPDF_Signer extends SetaPDF {
        void setRemoveNeedAppearancesFlag ( [boolean $status=true] )
}
```

PDF documents (with form fields) can include a flag which tells the viewer application to rebuild the appearance streams of a form field at any time. This issue will destroy any digital signature, so Acrobat or the Adobe Reader will simply ignore the digital signature. If the SetaPDF-Signer API should sign a document where this flag is set and it's value is true it'll return an SetaPDF_Error object, because it makes no sense to sign this document. If you want to force the API to sign the document you can define this behaviour with this method.

If you pass true (or nothing) to this method, the API will remove the flag.

You should check the resulting document if all needed
content is viewable after the signing process.

Parameters

$status

True: remove the flag. *False*: don't remove the flag.

SetaPDF_Signer::createNewTmpFileName()

Description

```
SetaPDF_Signer extends SetaPDF {
        string createNewTmpFileName ( void )
}
```

This method is a kind of helper method which is used to create unique filenames in the temporary directory.

You can use this method to create temporary files which will be included in the cleanTmpDirectory routine of the SetaPDF-Signer API.

Return value

An absoulte unique filename/path

SetaPDF_Signer::cleanTmpDirectory()

Description

```
SetaPDF_Signer extends SetaPDF {
       boolean cleanTmpDirectory ( void )
}
```

This methods deletes olded files in the temporary directory.

If the API causes an error it could be that temporary files will remain in the given temporary directory.

In the PHP 5 version this method is called automatically by the __destruct()-method.

In PHP 4 you can call it your own or register it as a shutdown function:

register_shutdown_function(array(&$signerInstance,'cleanTmpDirectory'));

The method makes use of the glob()-function. If this function is disabled cause of any security reason you can define your own function by setting it in the SetaPDF::$globFunctionName property. An example of a replacement function can be found in the user contributed notes on php.net

By default temporary files which are older than 60 seconds will be deleted. To adjust this value, change the SetaPDF::$tmpFilesLifetime property.

Return value

True if files were deleted. *False* if no file was deleted.

SetaPDF_Signer::setSignatureProperty()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed setSignatureProperty ( string $name[, mixed $value] )
}
```

Sets a value of a signature property.

Parameters

$name

The name of the signature property. Possible values are:

- name
- location
- reason
- contactInfo
- timeOfSigning

For a detailed description of each possible property check their synonym methods.

$value

The value of the given property.

If *null* is passed the property will not be written to the signature dictionary in the resulting PDF document. Except the "timeOfSigning"-property. If this property is set to *null* the time of signing will be the current local time on the server.

All values (except the "timeOfSigning"-value) are strings. The "timeOfSigning"-value is an integer value - a unix timestamp.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::getSignatureProperty()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed getSignatureProperty ( string $name )
}
```

Gets a value of a signature property.

Parameters

$name

The name of the signature property. Possible values are:

- name
- location
- reason
- contactInfo
- timeOfSigning

For a detailed description of each possible property check their synonym methods.

Return Values

The value or an SetaPDF_Error object if the property is unknown.

Version

since 1.4

SetaPDF_Signer::setName()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed setName ( string $name )
}
```

This method is a synonym method for:

$this->setSignatureProperty('name', $name);

With it you can set the name of the person or auhority signing the document. You should only set this value if it's not possible to extract the name of the signature.

Parameters

$name

The name of the person or authority signing the document.

If null is passed the property will not be written to the signature dictionary in the resulting PDF document.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::getName()

Description

```
SetaPDF_Signer extends SetaPDF {
       mixed getName ( void )
}
```

This method is a synonym method for:

$this->getSignatureProperty('name');

It will return the name of the person or auhority signing the document, if set.

Return Values

The name of the person or auhority signing the document or null.

Version

since 1.4

SetaPDF_Signer::setLocation()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed setLocation ( string $location )
}
```

This method is a synonym method for:

$this->setSignatureProperty('location', $location);

Parameters

$location

The CPU host name or physical location of signing.

If null is passed the property will not be written to the signature dictionary in the resulting PDF document.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::getLocation()

Description

```
SetaPDF_Signer extends SetaPDF {
       mixed getLocation ( void )
}
```

This method is a synonym method for:

$this->getSignatureProperty('location');

It will return the CPU host name or physical location of signing, if set.

Return Values

The CPU host name or physical location of signing or null.

Version

since 1.4

SetaPDF_Signer::setReason()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed setReason ( string $reason )
}
```

This method is a synonym method for:

$this->setSignatureProperty('reason', $reason);

Parameters

$reason

The reason for the signing, such as (I agree...).

If null is passed the property will not be written to the signature dictionary in the resulting PDF document.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::getReason()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed getReason ( void )
}
```

This method is a synonym method for:

$this->getSignatureProperty('reason');

It will return the reason for the signing, if set.

Return Values

The reason for the signing or null.

Version

since 1.4

SetaPDF_Signer::setContactInfo()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed setContactInfo ( string $contactInfo )
}
```

This method is a synonym method for:

$this->setSignatureProperty('contactInfo', $contactInfo);

Parameters

$contactInfo

Information provided by the signer to enable a recipient to contact the signer to verify the signature; for example, a phone number.

If null is passed the property will not be written to the signature dictionary in the resulting PDF document.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::getContactInfo()

Description

```
SetaPDF_Signer extends SetaPDF {
       mixed getContactInfo ( void )
}
```

This method is a synonym method for:

$this->getSignatureProperty('contactInfo');

It will return the contact information of the signer, if set.

Return Values

The contact information of the signer or null.

Version

since 1.4

SetaPDF_Signer::setTimeOfSigning()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed setTimeOfSigning ( integer $timeOfSigning )
}
```

This method is a synonym method for:

$this->setSignatureProperty('timeOfSigning', $timeOfSigning);

Parameters

$timeOfSigning

A unix timestamp representing the time of signing.

If null is passed the time of signing will be the local time of the server.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::getTimeOfSigning()

Description

```
SetaPDF_Signer extends SetaPDF {
       mixed getTimeOfSigning ( void )
}
```

This method is a synonym method for:

$this->getSignatureProperty('timeOfSigning');

Will return a unix timestamp representing the time of signing, if set.

Return Values

A unix timestamp representing the time of signing or null.

Version

since 1.4

SetaPDF_Signer::setCertificationLevel()

Description

```
SetaPDF_Signer extends SetaPDF {
        void setCertificationLevel ( integer $certificationLevel )
}
```

Sets/Activates the certification level. A certification level specifies what changes are allowed after the document is certified.

Parameters

$certificationLevel

With the certification level you can specify what changes are allowed.

Following values/constans are available:

- SETAPDF_SIG_CERTIFIED_NO_CHANGES_ALLOWED = 1
- SETAPDF_SIG_CERTIFIED_FORM_FILLING = 2
- SETAPDF_SIG_CERTIFIED_FORM_FILLING_AND_ANNOTATIONS = 3

To disable certification pass:

- SETAPDF_SIG_NOT_CERTIFIED = 0 (default)

Version

Available since version 1.3

SetaPDF_Signer::getPageBoxes()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed getPageBoxes ( integer $pageNo[, mixed $fileName=null] )
}
```

Gets the page boxes of a PDF document.

Parameters

$pageNo

The page number

$fileName

The path of the PDF file. If not set, the document which is set in the factory()-method or by setSourceFileName() is taken.

Return Values

An array of boxes, while the keys are the box-names (f.g. /MediaBox, /TrimBox,...).

Each box has following entries:

- x: the abscissa
- y: ordinate
- w: the width
- h: the height
- llx: lower left abscissa
- lly: lower left ordinate
- urx: upper right abscissa
- ury: upper right ordinate

If an error occurs this method will return a SetaPDF_Error-Object.

Version

since 1.4

SetaPDF_Signer::getOriginBox()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed getOriginBox ( integer $pageNo[, mixed $fileName=null] )
}
```

Gets the origin page box. This will be the /CropBox or the /MediaBox. The method will check the existence of the desired box for you.

Parameters

$pageNo

The page number

$fileName

The path of the PDF file. If not set, the document which is set in the factory()-method or by setSourceFileName() is taken.

Return Values

An array with the following entries:

- x: the abscissa
- y: ordinate
- w: the width
- h: the height
- llx: lower left abscissa
- lly: lower left ordinate
- urx: upper right abscissa
- ury: upper right ordinate

If an error occurs this method will return a SetaPDF_Error-Object.

Version

since 1.4

SetaPDF_Signer::getPageRotation()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed getPageRotation ( integer $pageNo[, mixed $fileName=null] )
}
```

Gets the number of degrees by which the page is rotated.

Parameters

$pageNo

The page number

$fileName

The path of the PDF file. If not set, the document which is set in the factory()-method or by setSourceFileName() is taken.

Return Values

An integer value of a factor 90 or false if no value is given.

If an error occurs this method will return a SetaPDF_Error-Object.

Version

since 1.4

SetaPDF_Signer::setTsModule()

Description

```
SetaPDF_Signer extends SetaPDF {
        boolean setTsModule ( SetaPDF_Signer_Module_Ts_Abstract $tsModule )
}
```

Adds a time stamp module to the signature process.

This method will trigger the onSet-method of the time stamp module.

Parameters

$tsModule

An instance of a time stamp module.

Return Values

True if the given variable is an instance of SetaPDF_Signer_Module_Ts_Abstract. *False* in the other case.

Version

as of version 1.5

SetaPDF_Signer::sign()

Description

```
SetaPDF_Signer extends SetaPDF {
        mixed sign ( SetaPDF_Signer_Module &$module, string $target[, string
        $dest='I'] )
}
```

This method starts the signing process.

Parameters

&$module

An instance of a signing module. Currently only modules for OpenSSL are available. See SetaPDF_Signer_Module_OpenSsl and SetaPDF_Signer_Module_OpenSslCli.

$target

The resulting filename or path to the local filesystem where the resulting document will be saved.

$dest

Defines how the resulting document is handled:

- "F" saves the file to the file system
- "D" the file will be send to the client with a download dialogue
- "I" the file will be displayed in the client's browser window.

Return Values

True is everything works as expected or an SetaPDF_Error object if an error occurs.

SetaPDF_Signer::unsetParser()

Description

```
SetaPDF_Signer extends SetaPDF {
        boolean unsetParser ( string $fileName )
}
```

The SetaPDF-Signer API caches parser objects, so if you want to sign one document multiple times the source document is parsed only once. If you want to sign multiple documents in one instance it's good to close the unneeded parser objects.

Parameters

$fileName

The filename of the sourcefile.

Return Values

True if a parser object existed and could be deleted for this filename. *False* if no parser object exists for this filename.

SetaPDF_Signer_Module - Abstract Module Class

The class SetaPDF_Signer_Module is the abstract class for all extended signature module classes. The abstract class doesn't offers any public functionalities.

Class Overview

SetaPDF_Signer_Module

Child Classes

- SetaPDF_Signer_Module_OpenSsl
- SetaPDF_Signer_Module_OpenSslCli

SetaPDF_Signer_Module_OpenSsl

This class represents a module which will create the signature with phps build in OpenSSL functions.

Class Overview

File:

SetaPDF_Signer_Module
└─ SetaPDF_Signer_Module_OpenSsl

Methods

- SetaPDF_Signer_Module_OpenSsl::setSignCert()
- SetaPDF_Signer_Module_OpenSsl::setPrivateKey()
- SetaPDF_Signer_Module_OpenSsl::setExtraCerts()

Inherited Methods

SetaPDF_Signer_Module_OpenSsl::setSignCert()

Description

```
SetaPDF_Signer_Module_OpenSsl extends SetaPDF_Signer_Module {
        void setSignCert ( mixed $signCert )
}
```

With this method you have to set the certificate with which you want to sign the document.

Parameters

$signCert

The signing certificate. This parameter can be passed like desribed here.

SetaPDF_Signer_Module_OpenSsl::setPrivateKey()

Description

```
SetaPDF_Signer_Module_OpenSsl extends SetaPDF_Signer_Module {
        void setPrivateKey ( mixed $privKey )
}
```

Define the private key.

Parameters

$privKey

The private key. This parameter can be passed like desribed here.

SetaPDF_Signer_Module_OpenSsl::setExtraCerts()

Description

```
SetaPDF_Signer_Module_OpenSsl extends SetaPDF_Signer_Module {
        void setExtraCerts ( string $extraCerts )
}
```

Let's you add extra certificates to include in the signature which can for example be used to help the recipient to verify the certificate that you used.

Parameters

$extraCerts

Specifies the name of a file containing the extra certificates.

SetaPDF_Signer_Module_OpenSslCli

This class represents a module which will create the signature with OpenSSL calls through the command line.

Class Overview

File:

SetaPDF_Signer_Module
└─ SetaPDF_Signer_Module_OpenSslCli

Methods

- SetaPDF_Signer_Module_OpenSslCli::setOpenSslPath()
- SetaPDF_Signer_Module_OpenSslCli::setSignCert()
- SetaPDF_Signer_Module_OpenSslCli::setPrivateKey()
- SetaPDF_Signer_Module_OpenSslCli::setPrivateKeyPassword()
- SetaPDF_Signer_Module_OpenSslCli::setExtraCerts()

Inherited Methods

SetaPDF_Signer_Module_OpenSslCli::setOpenSslPath()

Description

```
SetaPDF_Signer_Module_OpenSslCli extends SetaPDF_Signer_Module {
        void setOpenSslPath ( mixed $openSslPath )
}
```

Set the path to OpenSSL binaries.

The default value is: /usr/bin/

Parameters

$openSslPath

The path to OpenSSL binaries.

SetaPDF_Signer_Module_OpenSslCli::setSignCert()

Description

```
SetaPDF_Signer_Module_OpenSslCli extends SetaPDF_Signer_Module {
        void setSignCert ( string $signCert )
}
```

With this method you have to set the certificate with which you want to sign the document.

Parameters

$signCert

An absolute path to the signing certificate file.

SetaPDF_Signer_Module_OpenSslCli::setPrivateKey()

Description

```
SetaPDF_Signer_Module_OpenSslCli extends SetaPDF_Signer_Module {
        void setPrivateKey ( [string $privKey=null] )
}
```

Define a private key file.

Parameters

$privKey

The absolute path to the private key file.

It's not neccesarry to pass the private key apart if it is already included in the certificate, set with SetaPDF_Signer_Module_OpenSslCli::setSignCert().

To remove the private key property just omit this parameter or pass *null*.

SetaPDF_Signer_Module_OpenSslCli:: setPrivateKeyPassword()

Description

```
SetaPDF_Signer_Module_OpenSslCli extends SetaPDF_Signer_Module {
        void setPrivateKeyPassword ( string $password )
}
```

Set the password for the private key.

Parameters

$password

The password for the private key.

SetaPDF_Signer_Module_OpenSslCli::setExtraCerts()

Description

```
SetaPDF_Signer_Module_OpenSslCli extends SetaPDF_Signer_Module {
        void setExtraCerts ( string $extraCerts )
}
```

Let's you add extra certificates to include in the signature which can for example be used to help the recipient to verify the certificate that you used.

Parameters

$extraCerts

Specifies the name of a file containing the extra certificates.

SetaPDF_Signer_Module_Ts_Abstract

An abstract class for timestamp modules.

Class Overview

File:

SetaPDF_Signer_Module_Ts_Abstract

Child Classes

- SetaPDF_Signer_Module_Ts_Curl

Methods

- SetaPDF_Signer_Module_Ts_Abstract::setSignature()
- SetaPDF_Signer_Module_Ts_Abstract::getSignature()
- SetaPDF_Signer_Module_Ts_Abstract::getParsedSignature()
- SetaPDF_Signer_Module_Ts_Abstract::getHash()
- SetaPDF_Signer_Module_Ts_Abstract::setReqPolicy()
- SetaPDF_Signer_Module_Ts_Abstract::getReqPolicy()
- SetaPDF_Signer_Module_Ts_Abstract::setNonce()
- SetaPDF_Signer_Module_Ts_Abstract::getNonce()
- SetaPDF_Signer_Module_Ts_Abstract::setCertReq()
- SetaPDF_Signer_Module_Ts_Abstract::getCertReq()
- SetaPDF_Signer_Module_Ts_Abstract::_getTsq()
- SetaPDF_Signer_Module_Ts_Abstract::_getFinalSignature()
- SetaPDF_Signer_Module_Ts_Abstract::onSet()
- SetaPDF_Signer_Module_Ts_Abstract::createTimeStamp()

SetaPDF_Signer_Module_Ts_Abstract::setSignature()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        void setSignature ( string $signature )
}
```

Sets the original signature. (automatically called from SetaPDF_Signer class)

Parameters

$signature

The BER encoded signature.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::getSignature()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        string getSignature ( void )
}
```

Gets the original signature.

Return value

The BER encoded signature.

Version

as of version 1.5

...Signer_Module_Ts_Abstract::getParsedSignature()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        Asn1_Type getParsedSignature ( void )
}
```

Gets the signature in object form.

Return value

The signature in object form.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::getHash()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        string getHash ( void )
}
```

Gets the hash for the timestamp request.

Return value

The sha1-hash of the signed data.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::setReqPolicy()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        void setReqPolicy ( string $reqPolicy )
}
```

Sets the reqPolicy field in the timestamp request.

The reqPolicy field, if set, indicates the TSA policy under which the TimeStampToken SHOULD be provided.

Parameters

$reqPolicy

The OID of the policy.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::getReqPolicy()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        string getReqPolicy ( void )
}
```

Gets the reqPolicy (OID) value.

Return value

The OID if set or NULL.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::setNonce()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        void setNonce ( boolean $nonce )
}
```

Define if the nonce field should be set or not.

Parameters

$nonce

True or false

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::getNonce()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        boolean getNonce ( void )
}
```

Queries if nonce should be set.

Return value

True or false.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::setCertReq()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        void setCertReq ( boolean $certReq )
}
```

Set the certReq field.

Parameters

$certReq

True or false

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::getCertReq()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        string getCertReq ( void )
}
```

Get the value of the certReq field.

Return value

If the field is set to true \xFF.

If the field is not set \x00.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::_getTsq()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        string _getTsq ( void )
}
```

Creates the timestamp request/query.

Return value

The timestamp request in BER format.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::_getFinalSignature()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        Asn1_Type _getFinalSignature ( Asn1_Type $tsToken )
}
```

Attaches the timestamp token to the original signature and returns it.

Parameters

$tsToken

The timestamp token.

Return value

The signature including the timestamp token.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::onSet()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        Asn1_Type onSet ( SetaPDF_Signer $signer )
}
```

Will be called when ever the module is attached to the Signer class.

In this method you can adjust some settings of the signer class like the signature content length.

Parameters

$signer

A SetaPDF_Signer instance.

Return value

The signature including the timestamp token.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Abstract::createTimeStamp()

Description

```
SetaPDF_Signer_Module_Ts_Abstract {
        string abstract createTimeStamp ( void )
}
```

This method is an abstract method and have to be implemented later.

It should request a timestamp token and returns the final signature.

Return value

The signature including the timestamp token in BER format.

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Curl

A time stamp module, which uses curl for HTTP transport.

Class Overview

File:

SetaPDF_Signer_Module_Ts_Abstract
└─ SetaPDF_Signer_Module_Ts_Curl

Methods

- SetaPDF_Signer_Module_Ts_Curl::__construct()
- SetaPDF_Signer_Module_Ts_Curl::setCurlOpts()
- SetaPDF_Signer_Module_Ts_Curl::createTimeStamp()

Inherited Methods

Class: SetaPDF_Signer_Module_Ts_Abstract

- SetaPDF_Signer_Module_Ts_Abstract::setSignature()
- SetaPDF_Signer_Module_Ts_Abstract::getSignature()
- SetaPDF_Signer_Module_Ts_Abstract::getParsedSignature()
- SetaPDF_Signer_Module_Ts_Abstract::getHash()
- SetaPDF_Signer_Module_Ts_Abstract::setReqPolicy()
- SetaPDF_Signer_Module_Ts_Abstract::getReqPolicy()
- SetaPDF_Signer_Module_Ts_Abstract::setNonce()
- SetaPDF_Signer_Module_Ts_Abstract::getNonce()
- SetaPDF_Signer_Module_Ts_Abstract::setCertReq()
- SetaPDF_Signer_Module_Ts_Abstract::getCertReq()
- SetaPDF_Signer_Module_Ts_Abstract::_getTsq()
- SetaPDF_Signer_Module_Ts_Abstract::_getFinalSignature()
- SetaPDF_Signer_Module_Ts_Abstract::onSet()

SetaPDF_Signer_Module_Ts_Curl::__construct()

Description

```
SetaPDF_Signer_Module_Ts_Curl {
        void __construct ( string $url[, string $username=null[, string
        $password='']] )
}
```

The constructor

Parameters

$url

URL of the time stamp server.

$username

Username if server requires http-auth

$password

Password if server requires http-auth

Version

as of version 1.5

SetaPDF_Signer_Module_Ts_Curl::setCurlOpts()

Description

```
SetaPDF_Signer_Module_Ts_Curl {
        void setCurlOpts ( [boolean $verifyPeer=true[, boolean
        $verifyHost=true[, string $caInfo=null]]] )
}
```

Sets some curl options.

Parameters

$verifyPeer

Value for CURLOPT_SSL_VERIFYPEER.

$verifyHost

Value for CURLOPT_SSL_VERIFYHOST.

$caInfo

Value for CURLOPT_CAINFO.

Version

as of version 1.5