Payment Services

Payment Page

Specification

Version 5.1

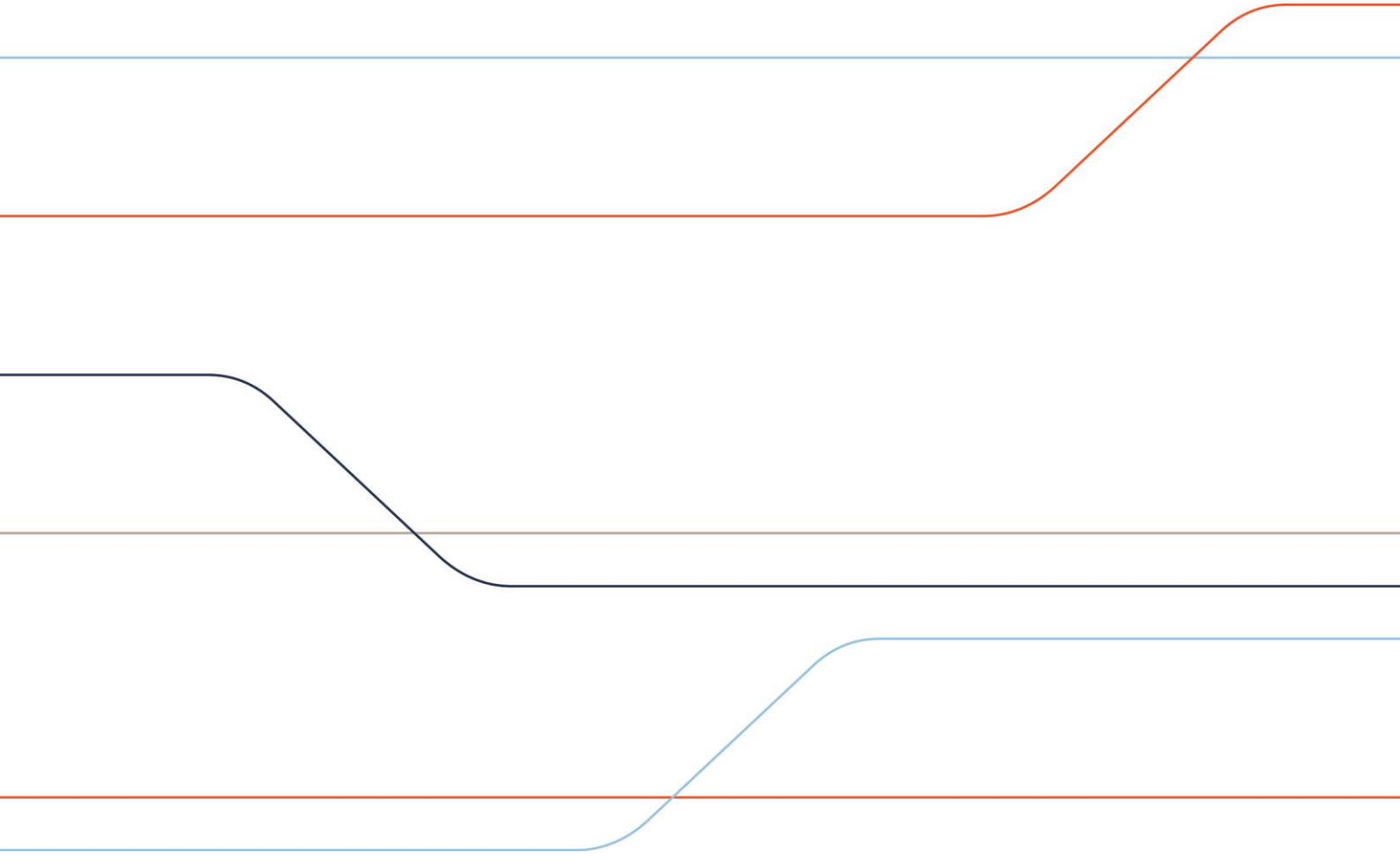


Table of contents

1 Introduction

The Saferpay Payment Page, in the following also called PP, is an online payment form provided and hosted by Saferpay. This document describes the integration of the PP in shop systems with the Client Library, in the following also called LIB and the Saferpay https Interface in the following also called HI.

1.1 Requirements

The use of the PP requires the fulfillment of the following conditions:

- A corresponding Saferpay eCommerce license and thus the existence of a valid identifier with Username and Password for the Saferpay system.
- At least one active Saferpay terminal, via which the payments can be processed, exists and the associated TERMINALID, respectively the concerned Saferpay ACCOUNTID, is available.
- The existence of an valid acceptance contract for credit cards or another payment means.
- In order to be able to use the HI, a HI configuration with the merchant data must be setup on Saferpay side. The keys for the signature of the (SSL secured) communication with saferpay are provided by this configuration. The setup is free of charge but must be individually requested for each Saferpay account. Please send an informale-mail requesting setup to onlinepayment@six-group.com if you have concluded your contract in Switzerland or to service.saferpay@six-payment-services.com if you have a contract for another country (D, NL, A, etc.).If you choose to use the Redirect Address please take care to transmit the complete web address of the SUCCESSLINK in your mail since the storage of this address on Saferpay side is then mandatory.

1.2 Presentation of the Payment Page

In addition to the full-page display of the Payment Page, Saferpay supports the embedding of the PP into an inline frame (iframe). Saferpay does not support opening the PP as a pop-up.

If you want to use the Payment Page inside an iFrame, you have to submit the parameter APPEARANCE with either the value “mobile” or “embedded” (see chapter 6.1)!

Although Saferpay supports the PP's display in an iframe, we recommend presenting it in full-page mode. In our view, this makes it a lot more user friendly for customers because the URL and SSL certificate are visible at all times. It makes clear to customers the provider they are using or the website on which they are currently located. It also leads to a reduction in the number of payment interruptions due to security questions on the part of the processor.

PayPal blocks payment queries from an iframe!

For this reason, the Saferpay Payment Page carries out a breakout from the iframe and then calls up PayPal full page. The call-up of the SUCCESS page thereby no longer occurs in iframe but as a whole page. Please take note of this when integrating PayPal.
It cannot be ruled out that more payment means providers and operators of 3-D Secure authentication pages follow PayPal. In the interests of security, ask your payment means processor whether the display of the PP in an iframe is permitted or if any integration into an iframe is prohibited on security grounds!

1.3 Data Security and PCI DSS

The credit card organizations have initiated the security program named PCI DSS (Payment Card Industry Data Security Standard) to prevent fraud and abuse of credit cards.

Please take care to respect the PCI DSS guidelines in the design of your payment processes and the usage of the Saferpay Authorization Interface. In combination with the optional Saferpay Secure Card Data service the payment process can be designed so safe that no credit card number is processed, stored or transferred via your (web) servers. The risk of abuse of the credit card data is thereby reduced and the expenses for the PCI DSS review of the merchant system are greatly reduced

If you have any questions regarding PCI DSS, please contact your acquirer or a qualified security provider (see https://www.pcisecuritystandards.org).

1.4 Supported Payment Means

The Saferpay Payment Page actually allows the processing of transactions for the following payment means:

- Visa (including the 3-D Secure security technology Verified by Visa)
- MasterCard (including the 3-D Secure security technology MasterCard Secure Code)
- Maestro international
- V PAY
- American Express
- Diners Club
- J.C.B.
- ELV electronic direct debit (Germany only)
- giropay
- iDEAL
- PayPal
- eps
- PostFinance Card und PostFinance E-Finance
- Przelewy Online
- MasterPass *

** Actually not a payment means, but an electronic wallet that contains the customer's payment data.*

1.5 Format Information

The following abbreviations for format information are used in this document:

a	Letters (a - z, A - Z)
n	numeric characters (0 - 9)
an	alphanumeric characters (a - z, A - Z, 0 - 9)
s	Special characters (:?,-(+'.)/ and space)
ans	alphanumeric and special characters

2 Saferpay Client Library

The Saferpay LIB is to be installed on the server that provides the application of the merchant.
After the installation Saferpay classes and methods are available on the server.

Root- respectively admistrator rights on the destination server are required to install the LIB and to generate a new configuration (generation of keys).

The LIB is available as .NET- or Java-version. The corresponding installation files can be downloaded in the download area of the Saferpay Backoffice via the following address:

https://www.saferpay.com/download/

If neither the .NET LIB nor the Java LIB can be used or if a local installation is not possible the sSferpay https Interface can be used as an alternative.

2.1 Requirements

2.1.1 .NET Client Library

The Saferpay .NET Client LIB is compiled with .Net Framework 2.0. So it is mandatory to have installed this version on the target server, too.

2.1.2 Java Client Library

On the target server a Sun Java Runtime Environment (JRE) version 1.3.1 or newer has to be installed. Other Java environments from IBM or OpenJDK are not compatible with Saferpay Java LIB.

2.2 Installation

2.2.1 .NET Client Library

Please start the downloaded file “saferpay_dotnet.exe” and follow the instructions of the setup-assistant.

2.2.2 Java Client Library

For the integration in java please unpack the downloaded .zip file “saferpayac_java_v1.0.14.zip” and copy the included “saferpay.jar” into the directory jre/lib/ext.

For the integration in other programming- or script-languages the “saferpay.jar” can be copied in any directory.

2.3 Proxy Server Configuration

In case communication in the network takes place via a proxy server, the relevant configuration data of the Saferpay LIBs are required.

2.3.1 .NET Client Library

To allow a proxy server to be used, a parameter needs to be added to the “config.xml” file. The file is located in the installation directory of the .NET client, for instance in `C:\Programme\Saferpay\Client\.`

Proxy server with user ID

To enable communication with individual login data via a proxy requires the following parameters to be added to the “config.xml” in any order:

```
PROXYPASSWORD="secret"
PROXYUSERNAME="MyProxyUser"
PROXYADDRESS="http://localhost:8080"
USEPROXY="True"
USEDEFAULTCREDENTIALS="False"
```

Proxy server without user ID

To enable communication without specifying a proxy user and proxy password, the following parameters need to be added to the “config.xml”:

```
PROXYADDRESS="http://localhost:8080"
USEPROXY="True"
USEDEFAULTCREDENTIALS="True"
```

Depending on the proxy configuration, the content of the “config.xml” will then be as follows:

```
<IDP MSGTYPE="SetupResponse" GXID="6216B171-B449-4D02-A114-D42AB58D42AE"
CUSTOMERID="99867" VERSION="47"
VTAUTOURL="https://www.saferpay.com/user/setup.asp"
VTURL="https://www.saferpay.com/vt2/Pay.aspx" VTKEYID="1-0"
CAPTUREURL="https://www.saferpay.com/scai2/index.aspx"
VTSCRIPTURL="http://www.saferpay.com/OpenSaferpayScript.asp"
USEDEFAULTCREDENTIALS="True" USEPROXY="True"
PROXYADDRESS="http://localhost:8080" />
```

2.3.2 Java Client Library

For the Java LIB, the configuration of a proxy server can be specified via a “settings.xml” file or via a command line call. If using the “settings.xml“, this must be created in the same directory in which “saferpay.jar“ is located, such as `jre/lib/ext`.

Proxy server with user ID

Example for “settings.xml”:

```
<IDP PROXYHOST="10.23.209.100" PROXYPORT="8080" PROXYUSERNAME="myUserId"
PROXYPASSWORD="myPassword" TRACEOPT="rawhttp" VERSION="1" USEPROXY="1" />
```

Example for command line call:

```
--proxyHost 10.23.209.100 --proxyPort 8080 --proxyUser myUserId
--proxyPassword myPassword
```

Proxy server without user ID

Example for “settings.xml”:

```
<IDP PROXYHOST="10.23.209.100" PROXYPORT="8080" TRACEOPT="rawhttp"
VERSION="1" USEPROXY="1" />
```

Example for command line call:

```
--proxyHost 10.23.209.100 --proxyPort 8080
```

2.4 Key Generation

Besides the SSL encrypted communication between the LIB and the Saferpay servers the data of a Saferpay account is also protected by a digital signature according to the PGP (Pretty Good Privacy) policy. For this purpose a key pair must be generated and stored on the merchant-server for every Saferpay Account. The generation of the keys requires a valid login and password for the Saferpay Backoffice. After successfully generating the keys the password can be changed via the Saferpay Backoffice since the generation of the key is only to be done once and does not need to be repeated. Generated key pairs remain valid and should therefore be kept safe with restricted access.

2.4.1 .NET Client Library

After installation of the .NET LIB a GUI (Graphical User Interface) is available for the key generation. The GUI can be found at:

The Saferpay Client Setup opens. Please follow the subsequent instructions.

2.4.2 Java Client Library

The key generation with the java LIB is done via the command line. Therefore please change to the directory with the Saferpay.jar and enter the following command:

```
java -jar Saferpay.jar -conf -p <target directory> -r
https://www.saferpay.com/user/setup.asp -u <YOUR-ACCOUNT> -w 8e7Yn5yk
```

The example uses the access data of the Saferpay Test account.

The command line help can be called with:

```
java -jar Saferpay.jar –h
```

3 Classes and Methods of the Client Library

This chapter describes the available classes and methods for the integration of the Saferpay Payment Page.

3.1 Summary

3.1.1 Creation of the payment link

The payment link is generated with the CreatePayInit() method. The generated MessageObject must be filled with the transaction parameters before the call of GetPostUrl().

1) Creation of a MessageFactory Object.
2) Accessing the corresponding configuration with Open().
3) Call of CreatePayInit(), in order to get an empty MessageObject.
4) Call of SetAttribute() with the MessageObject to set the parameters.
5) Call of GetPostUrl() to get a payment link for the PP.

3.1.2 Check of the authorization response

1) Creation of a MessageFactory Object.
2) Accessing the corresponding configuration with Open().
3) Call of VerifyPayConfirm(), in order to verify DATA and SIGNATURE.
4) Reading ID and TOKEN with GetAttribute()
5) Storing the values of ID and TOKEN

3.1.3 Settlement of a Reservation

6) Creation of a MessageFactory Object.
7) Accessing the corresponding Configuration with Open().
8) Call of CreateRequest("PayComplete"), in order to get a MessageObject.
9) Call of SetAttribute() with the MessageObject to set ID and TOKEN.
10) Call of Capture() with the MessageObject.

3.2 MessageFactory Class

```
Class MessageFactory
{
   void Open(String path);
   MessageObject CreatePayInit();
   MessageObject VerifyPayConfirm(String data, String signature);
   MessageObject CreateRequest(String msgtype);
};
```

3.3 MessageObject Class

```
Class MessageObject
{
    void SetAttribute(String name, String value);
    String GetAttribute(String name);
    String GetPostURL();
    String GetPostData();
    String GetPostSignature();
    void Capture();
};
```

3.4 Open() Method

The reference to the key pair of the merchant account is done by the call of Open(). In order to ensure that the other function calls of this MessageFactory do also refer to these keys Open() must be called before all other methods of the MessageFactory object.

3.5 CreatePayInit() Method

With CreatePayInit() a payment link can be generated. After setting the parameter values the link is generated by GetURL().

3.6 VerifyPayConfirm() Method

Verifies the digital signature of the confirmation message (MSGTYPE=PayConfirm) returned to the shop via SUCCESSLINK or NOTIFYURL in order to ensure that the response has not been manipulated.

3.7 CreateRequest() Method

Creates a new Request MessageObject of the specified message type (msgtype). For instance with CreateRequest("PayComplete") transactions with status Reservation can be settled or canceled. A reservation can also be settled with a partial amount, a transaction with status payment can be canceled and the Batch Close can be engaged.
CreateRequest("PayComplete") always needs ID and TOKEN for a settlement. For a settlement with reduced amount the additional parameter AMOUNT must be transmitted. For the cancel of a Reservation or a Payment as well as for the start of the Batch Close the additional parameter ACTION is required. Every call must contain the ACCOUNTID.

3.8 SetAttribute() Method

With SetAttribute() the needed parameters for the message are set. Please take care to respect the case sensitivity of the used parameter names.

3.9 GetAttribute() Method

GetAttribute() returns the value of a parameter of the message. If the parameter is not included in the message the call fails. Please take care to respect the case sensitivity of the used parameter names.

3.10 GetPostUrl() Method

The call of GetURL() returns the payment link of the message.

3.11 GetPostData() Method

The call of GetPostData() returns the DATA of the message.

3.12 GetPostSignature() Method

The call of GetPostSignature() returns SIGNATURE of the message.

3.13 Capture() Method

The call of Capture() transmits the message of the message type CreatePayComplete.

4 Saferpay https Interface

The Saferpay https Interface can be used as an alternative to the Saferpay Client Library.
This might be the case if for example the LIB cannot be installed or used on the destination system.

4.1 https Interface Addresses

Das Saferpay https Interface can be accessed via the following web addresses:

Generation of a payment link:
`https://www.saferpay.com/hosting/CreatePayInit.asp`

Verifying an authorization response:
`https://www.saferpay.com/hosting/VerifyPayConfirm.asp`

Settlement of a payment:
`https://www.saferpay.com/hosting/PayCompleteV2.asp`

For security reasons the parameters ACTION and AMOUNT are not available for the use of the HI PayComplete address.

Attention! Most frameworks verify the server certificate automatically. Nevertheless when using the Saferpay https interface, we recommend to make sure that your application verifies the www.saferpay.com server certificate to prevent man-in-the-middle attacks.

4.2 Transaction Processing Scheme

A transaction generally meets the following scheme:

1. Generation of the payment link (CreatePayInit).
2. The payment is independently processed by customer via Saferpay
3. In case of successful authorization the SUCCESSLINK is called with the authorization response.
4. The answer should be checked for plausibility/manopulation (VerifyPayConfirm).
5. Finally the Payment must be settled in order to get the money transfer initiated via the close batch. The settlement can be done or manually via the Saferpay Backoffice or automated via the PayComplete message.

4.3 Generation of the Payment URL via the CreatePayInit Address

The shop system transmits the Saferpay PayInit parameters via GET or POST to the HI. There a digitally signed payment link is generated by means of the signature of the concerned Saferpay Account and returned back to the shop system. The generated link can then be used for a payment for example by including it as link or button on the shop page.

Example call (GET):

```
https://www.saferpay.com/hosting/CreatePayInit.asp?AMOUNT=…
```

The result is returned as plain text without html tags:

```
https://www.saferpay.com/vt2/Pay.aspx?DATA=%3cIDP%20ACCOUNTID%3d%2299867%2
d94913159%22%20ALLOWCOLLECT%3d%22no%22%20AMOUNT%3d%22100%22%20BACKLINK%3d%
22%2e%22%20CURRENCY%3d%22DEM%22%20DELIVERY%3d%22no%22%20DESCRIPTION%3d%22T
estkauf%20Warenkorb%22%20EXPIRATION%3d%2220010408%2012%3a13%3a50%22%20FAIL
LINK%3d%22%2e%22%20KEYID%3d%220%2d37217%2dea645c3f3f0911d583d70050da413f31
%22%20MSGTYPE%3d%22PayInit%22%20SUCCESSLINK%3d%22%2e%22%20TOKEN%3d%22ea645
c5d3f0911d583d70050da413f31%22%2f%3e&SIGNATURE=2f1ec1fa51002817941c22e98b9
047422ba9ff8fce8b61dab8208a5aa8c82be7cda02ff8a66930481fc19b16d05e7bcedd2b0
e5be98fecad3d48bd43916a502f
```

In case of an error the identifier “ERROR” is returned together with an error description:

```
ERROR: Missing AMOUNT attribute
```

4.4 Checking the Authorization Response via the VerifyPayConfirm Address

After the successful payment the SUCCESSLINK is called and with it the parameters DATA und SIGNATURE are returned by GET to the shop system. The parameter DATA contains the PayConfirm message with the details of the authorization response and the parameter SIGNATURE the key used by Saferpay to sign DATA. In order to exclude a manipulation of the authorization response both parameters should be send to the VerifyPayConfirm address directly after reception. The HI will answer the request with OK or ERROR.

4.5 Example VerifyPayConfirm

The parameters DATA and SIGNATURE are send by GET or POST to the Saferpay Gateway:

Example call (GET):

```
https://www.saferpay.com/hosting/VerifyPayConfirm.asp?DATA=…
```

If the digital signature corresponds to the values of DATA the positive result is indicated by returning OK, the Saferpay ID and TOKEN:

```
OK:ID=56a77rg243asfhmkq3r&TOKEN=%3e235462FA23C4FE4AF65…
```

In case of an error the text "ERROR" is returned together with an error description:

```
ERROR: Possible manipulation
```

4.6 Settlement of a Payment via the CreatePayComplete Address

The settlement of an authorization is realized by sending the parameters ACCOUNTID and ID by GET or POST to the CreatePayComplete address of the HI. The Hi answers with OK or ERROR.

For security reasons settlements with reduced amount are not possible via the CreatePayComplete address of the HI. The parameters ACTION and AMOUNT are therefore ignored.

Example call (GET):

```
https://www.saferpay.com/hosting/PayCompleteV2.asp?ACCOUNTID=401860-
17795278&ID=5sfhmkq3rg54345abcd234&spPassword=8e7Yn5yk*
```

In case of successful processing OK is returned:

```
OK
```

In case of an error the text ERROR is returned together with an error description:

```
ERROR: Error description
```

** The submission of the parameter spPassword is specific to the use of the Saferpay test account. On live accounts this parameter must not be submitted.*

4.7 Call of the Payment Page via the Redirect Address

If for some reason the used system is not able to handle or receive the response data of the Saferpay HI (if for example the web server settings does not allow it or because a shop system based exclusively on JavaScript is used) the Redirect address of the HI can be used to open the Payment Page.

4.7.1 Disadvantages of Redirect

If Redirect is used the payment data has to be submitted via the web page allowing possible manipulation by experienced web users. The PayConfirm message could also be changed or simulated.

We strongly recommend to only use the Redirect address if the circumstances does not allow other proceeding. The CreatePayInit Address should always be preferred and used if possible!

4.7.2 https Interface Address

The Saferpay https Interface can be accessed via the following web address:

Call of the Payment Page via Redirect:

```
https://www.saferpay.com/hosting/Redirect.asp
```

4.7.3 Transaction processing with Redirect

The PayInit parameters needed for the payment are transmitted by GET or POST to the Saferpay HI. Thereupon the HI generates a payment link and opens the Payment Page directly via “redirect”.
Once the customer has successfully carried out his payment via the PP the authorization response (PayConfirm message) is transmitted to the Saferpay HI where it is checked for possible manipulation (VerifyPayConfirm) and then the Result is transmitted by GET with the SUCCESSLINK to the shop system. The Result of the check is indicated with the parameter RESULT. RESULT may have the value “0” for success or “1” in case of error or manipulation.

Example call (GET):

```
https://www.saferpay.com/hosting/Redirect.asp?AMOUNT=1095&…
```

The call will directly lead the customer, via his browser, to the PP in order to carry out his payment.

After the payment the digital signature will be checked by the HI and the result will be forwarded by GET with the SUCCESSLINK. If the check was successful the result with RESULT=0 is displayed and other PayConfirm parameters with additional authorization details are available.

```
http://www.shop.de/kasse_ok.php?RESULT=0&ID=J89HBV...
```

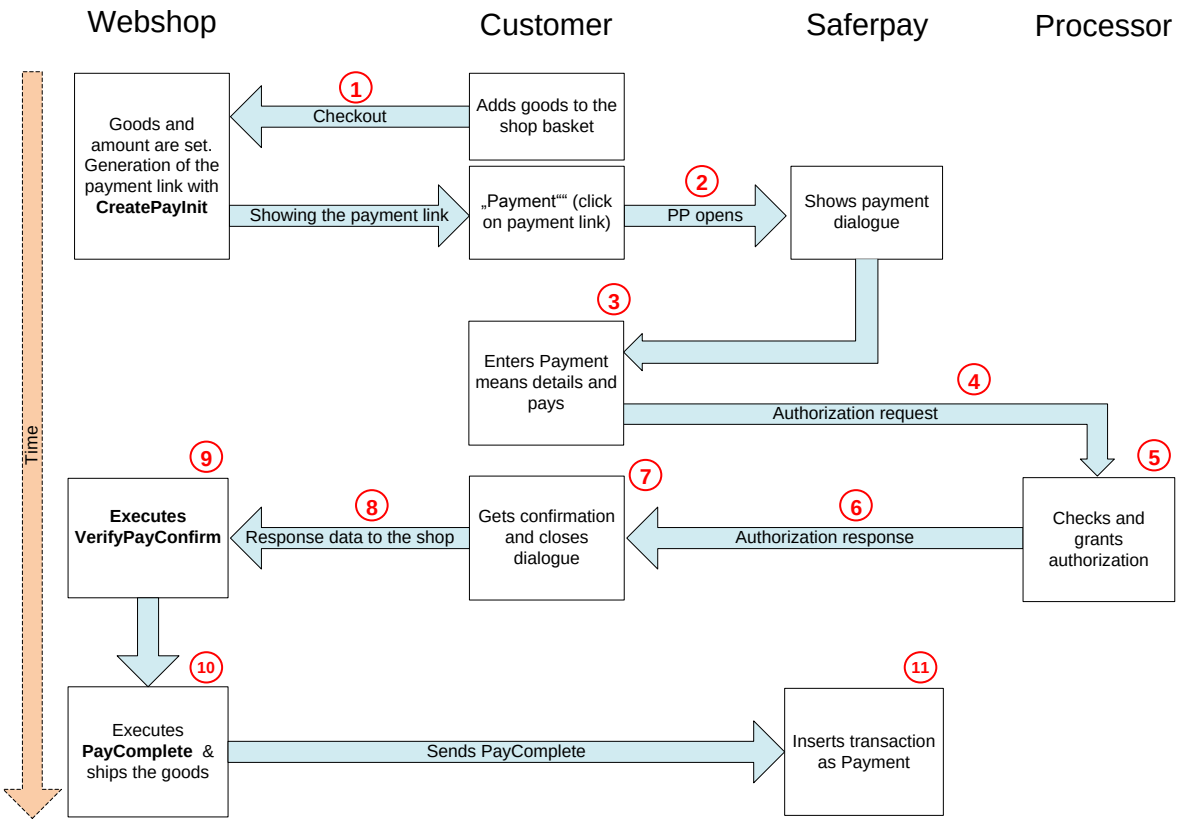
If the check of the digital signature fails just RESULT=1 is returned:

```
http://www.shop.de/kasse_ok.pl?Session=123&RESULT=1
```

5 Processing Steps

5.1 Overview

The following chart shows the process-flow of a successful online payment via the Saferpay Payment Page:



5.2 Process Description

1. Once the customer has filled his shop basket and the amount is set the shop creates the payment link (CreatePayInit). On the checkout page the webshop presents the payment link as „Payment“ button or something similiar.

2. The customer clicks on „Payment“ and the Payment Page does open.

3. The dialogue of the PP presets a payment means or the customer chooses a payment means for the transaction on the PP and then enters the needed payment credentials.

4. The online authorization of the payment means is processed.

5. The processor verifies and checks the authorization request and grants the authorization.

6. The processor returns the authorization response.

7. In the PP the customer is shown a confirmation of the payment and he is requested to finish the operation by clicking on “Close”.

8 The PP is closed and the customer is redirected to the shop via the SUCCESSLINK. The shop gets the response data of the Autorisation.

In rare cases it can happen that the SUCCESSLINK is called several times. This can be due to various reasons, which cannot always be prevented by technical means. To enable such instances to be recognised, we recommend the use of the NOTIFYURL or a check of the return values for unambiguity, for instance by using a shop session ID appended to the SUCCESSLINK via GET. That way you can make sure that the shop does not register purchase orders twice and that follow-on actions, such as the PayComplete call, are only effected once.

9 The merchant system checks the payment confirmation (VerifyPayConfirm) and stores it together with the transaction details.

10 The amount is settled (PayComplete) and the order can be processed.

The settlement of a Reservation is mandatory for the Batch Close. The Batch Close only processes transactions with the status "Payment" and forwards them to the Processor in order to engage the financial transfer. The money is subsequently credited as compound item to the merchant's business account. The merchant gets a settlement list from the payment means processor.

Depending on the business case the settlement can also be done at a later time, normally within 6 days since that is the normal lifetime of a reservation. Since this value might vary depending on processor and payment means please ask your processor for further information.

The Batch Close can be done manually or automatically.

11 Via the call of PayComplete the status of a transaction changes from "Reservation" to "Payment". The transaction status is shown in the Saferpay Backoffice Journal.

6 Parameters

6.1 PayInit Parameters

This table lists the available parameters for the CreatePayInit message.
If not specified as *Optional* the parameters are mandatory.

Parameter	Format	Description
ACCOUNTID	ns[..15]	The Saferpay account number of the merchant for this transaction. e.g. 401860-17795278 for the Saferpay Test Account.
AMOUNT	n[..8]	Authorization amount in minor currency unit e.g. 1230 in EUR means EUR 12,30.
CURRENCY	a[3]	ISO 4217 three-letter currency code e.g. CHF or EUR
DESCRIPTION	ans[..50]	Contains a description of the offer which is shown in the PP and after successful payment as well within the details of the transaction in the Saferpay Backoffice. It is also shown to the customer in the shopping cart when the MasterPass is used. It is not possible to customize the representation with control characters. It is therefore not recommended to use more than 50 digits even if technically possible. To ensure the correct processing of special characters the value must be transmitted html encoded (entities or unicode).
ORDERID	ans[..80]	*Optional, (mandatory for the payment mean giropay)* ORDERID should contain the shop reference number for the payment. In order to permit a later assignment the transmitted value should be unique. For PostFinance Alias Service a unique value is mandatory. Saferpay can process 80 characters. On Processor side generally far less characters can be processed. Long values are generally truncated. In practice a length of 12 characters has been proven to be a good value. In case of doubt please ask your processor how many characters he can process.
VTCONFIG	an[..20]	*Optional* VTCONFIG designates the configuration that is to be used for the Payment Page. In the Backoffice different configurations can be created. Some options as e.g. the insert of a logo can only be activated via the configuration. Value: name of the configuration in the Saferpay Backoffice.
SUCCESSLINK	ans[..1024]	URL to which the customer is to be forwarded to via browser redirect after the successful reservation. Saferpay appends the confirmation message (PayConfirm) by GET to this URL.
FAILLINK	ans[..1024]	URL to which the customer is to be forwarded to via browser redirect if the authorization attempt failed.
BACKLINK	ans[..1024]	*Optional* URL to which the customer is to be forwarded to via browser redirect if he aborts the transaction

Parameter	Format	Description
NOTIFYURL	ans[..1024]	Fully qualified URL which in case of successful authorization is called directly by the saferpay server transmitting the confirmation message (PayConfirm) by POST. Only standard ports (http port 80, https port 443) are allowed. Other ports will not work. We recommend to implement NOTIFYURL in order to ensure the reception of the confirmation message independently from possible errors or problems on customer or browser side. To facilitate the correlation between request and response it has proven to be useful to add a shop session ID as GET parameter to the NOTIFYURL. *** ATTENTION!** The use of NOTIFYURL is recommended under all circumstance to counter errors when the customer calls up SUCCESSLINKS. In this regard, please note that your system also normally receives the payment confirmation (PayConfirm) twice. The parameter ORDERID can be helpful in terms of identification here. In addition, the use of NOTIFYURL is mandatory for the payment means **Masterpass**. (See chapter 6.1.4).
AUTOCLOSE	n[..2]	*Optional* Specifies the period of time in seconds to close the PP automatically after the successful authorization; recommended value: “0” e.g. AUTOCLOSE=0
CCNAME	a[..3]	*Optional* Specifies whether the The card holder name input field of the PP is active or not. Values: “yes” or “no” Default value is “yes”
NOTIFYADDRESS	ans[..50]	*Optional* Email address of the merchant. In case of successful authorization Saferpay sends a confirmation email to this address.
USERNOTIFY	ans[..50]	*Optional* Email address of the customer. In case of successful authorization Saferpay sends a confirmation email to this address.
LANGID	a[2]	*Optional* Language code according to ISO 6391-1 Presets the language for the PP session. A list of the available language is in Chapter 6.1.1. Per default the Virtual Terminal uses the browsers language setting to determine PP language. If the browser language is not recognized German is taken as default language.
SHOWLANGUAGES	a[..3]	*Optional* Activates the language menu within the Saferpay Payment Page. Values: “yes” or “no”. Default value is “yes”

PAYMENTMETHODS	ns[..100]	*Optional* Use this parameter to restrict the payment means selectable in the Payment Page. Without PAYMENTMETHODS all active payment means of the terminal are shown. With PAYMENTMETHODS, a comma separated list of payment method IDs has to be transmitted.For example: "1,2,13" activates MasterCard, Visa and direct debit (ELV). You can find a list of all payment method IDs in Chapter 6.1.2 below.
DURATION	n[14]	*Optional* Specifies the duration of the payment link. After expiration of DURATION the payment link will be declined. *Format: YYYYMMDDhhmmss*.
PREAUTH	a[..3]	*Optional* *Is needed to identify a transaction as a preauthorization. Preauthorized transactions can be captured up to 30 days after the authorization with PayComplete.* *Values: "yes" or "no"* *If the parameter is not transferred, then a final authorization is carried out.* *** ATTENTION!** Preauthorization is only available for the payment means MasterCard and to the processors SIX and ConCardis!

Parameter	Format	Description
WALLETS	ns[..100]	*Optional* Triggers an immediate redirection to the login page of the customer's wallet, where the payment method for the automatic authorization request can be chosen. If PAYMENT METHODS are indicated at the same time, it is treated equally. Value: "MASTERPASS"
CARDREFID	ans[..40]	*Optional* Reference number for credit card number and expiry date or bank-account information (German direct debit). The value can be defined by the E-Commerce application or Saferpay (new). The use implies the service "Saferpay Secure Card Data" Values: unique reference value or "new" *
DELIVERY	a[..3]	*Optional* Specifies whether an address form will be displayed. Values: "yes", "no" Default value is "yes"
APPEARANCE	an[..7]	*Optional* Modifies PP appearance for target display. Values: "auto" (default), "mobile", "desktop", "embedded"
CSSURL	ans[..1024]	*Optional* Can only be used with APPEARANCE "mobile" or "embedded" Includes the fully qualified https URL to the CSS file for adjusting the PP to mobile view; see Section 7.
MANDATEID	an[..35]	*Optional* Mandate reference of the payment. Needed for German direct debits (ELV) only. The value has to be unique. As a default Saferpay transaction identifier is used.
RECURRING	a[..3]	*Optional* Flags the payment request as initial payment for possible recurring payments. Causes for ELV payments that a special mandate text for recurring payments is shown. Values: "yes" or "no" (default)
Display of the General Terms and Conditions**:		
TERMSCHECKBOXACTIVE	a[..8]	*Optional, required together with TERMSURL* Shows a checkbox for accepting the retailer's GTC. Values: "yes" or "no" (standard)
TERMSURL	ans[..1024]	*Optional, required together with TERMSCHECKBOXACTIVE* Contains the URL to the retailer's GTC.

Parameter	Format	Description
Address options:		
ADDRESS	an[..8]	*Optional* If this Attribute is set, an input form for an address appears during the PP session. Values: "DELIVERY", "CUSTOMER", "BILLING"
If the address information is already available it can be forwarded to the Payment Page with the following parameters in order to open the address form prefilled with the concerned data. Please note that the data has to be transmitted as html entity in order to ensure the correct processing of special characters.		
COMPANY	ans[..50]	*Optional* Company name
GENDER	a[1]	*Optional* gender values: “f” (female), “m” (male), “c” (company)
FIRSTNAME	ans[..50]	*Optional* firstname
LASTNAME	ans[..50]	*Optional* lastname
STREET	ans[..50]	*Optional* street
ZIP	an[..10]	*Optional* zip code
CITY	ans[..50]	*Optional* city
COUNTRY	a[2]	*Optional* 2 letter country code according to ISO 3166.
STATE	a[2]	*Optional* Identifier of a state of province. Corresponds to the last two characters according to ISO 3166-2.
EMAIL	ans[..50]	*Optional* e-mail address
PHONE	ns[..20]	*Optional* phone number

** In order to use CARDREFID=”new” a numeric start value must be set by Saferpay for the Account. This concerning please contact* integration.saferpay@six-payment-services.com*.*

*** For MasterPass transactions a note to accept the GTC is not shown. The parameters for the General Terms and Conditions are thus ignored when MasterPass is selected.*

6.1.1 Language codes for LANGID

Code	Name	Language
de	Deutsch	German
en	English	English
fr	Français	French
da	Dansk	Danish
cs	Čeština	Czech
es	Español	Spanish
hr	Hrvatski	Croatian
it	Italiano	Italian
hu	Magyar	Hungarian
nl	Nederlands	Dutch
no	Norsk	Norwegian
pl	Polski	Polish
pt	Português	Portuguese
ru	Русский	Russian
ro	Română	Romanian
sk	Slovenský	Slowak
sl	slovenščina	Slovenian
fi	Suomi	Finnish
sv	Svenska	Swedish
tr	Türkçe	Turkish
el	Ελληνικές	Greek
ja	日本語	Japanese
zh	中国語	Chinese

6.1.2 Payment Method IDs for PAYMENTMETHODS

Payment Method ID	Payment Method
1	MasterCard
2	Visa
3	American Express
4	Diners Club
5	JCB
6	Saferpay Testkarte
8	Bonus Card
9	PostFinance E-Finance
10	PostFinance Card
11	Maestro International
12	MyOne
13	Direct debit (ELV)
14	Invoice
15	Sofortüberweisung
16	PayPal
17	giropay
18	iDEAL
20	Homebanking AT (eps)
22	ePrzelewy

6.1.3 PayConfirm parameters of the authorization response

The PayConfirm message is returned with the authorization response. The following table lists the possible parameters:

Parameter	Format	Description
MSGTYPE	a[..30]	Always contains the value ”PayConfirm “.
VTVERIFY	ans[..40]	May contain additional information concerning the transaction processing. Default value: “(obsolete)”
KEYID	ans[..40]	Id of the key used to generate the signature
ID	an[28]	Unique saferpay transaction identifier.
TOKEN	ans[..40]	May contain additional information concerning the transaction processing. Default value: "(unused)"
ACCOUNTID	ns[..15]	The Saferpay account number of the merchant for this transaction. e.g. 401860-17795278 for the Saferpay Test Account.
AMOUNT	n[..8]	Authorization amount in minor currency unit e.g. 1230 in EUR means EUR 12.30.
CURRENCY	a[3]	ISO 4217 three-letter currency code e.g. CHF or EUR
DCCAMOUNT	n[..8]	*Optional (only payments with Dynamic Currency Conversion)* Amount of card currency in minor currency unit e.g. 1230 in EUR means EUR 12.30.
DCCCURRENCY	a[3]	*Optional (only payments with Dynamic Currency Conversion)* ISO 4217 three-letter code of card currency e.g. CHF or EUR
CARDREFID	ans[..40]	*Optional (only* if the parameter was submitted within the CreatePayInit message.) Contains the reference number for credit card number and expiry date or bank-account information (only german direct debit).
SCDRESULT	n[..4]	*Optional (only if the parameter CARDREFID was submitted within CreatePayInit Message)* contains the response code of the SCD registration: 0 request processed successfully. 7000 internal error (see DESCRIPTION). 7001 request could not be processed successfully 7002 Cardtype not available on this terminal 7003 Parameter with invalid content or format. 7004 CARDREFID not found (only with authorization). 7005 Missing parameter in registration request. 7006 CARDREFID already exists in database. 7007 No permission for SCD use.
PROVIDERID	n[..4]	Contains the Provider ID of the payment means processor
PROVIDERNAME	ans[..40]	Contains the name of the payment means processor
PAYMENTMETHOD	n[2]	Contains the payment method ID
ORDERID	an[..80]	*Optional* The reference number of the merchant submitted with CreatePayInit.

Parameter	Format	Description
IP	ns[..15]	*Optional** Contains the IP address of the customer. Only available if Riskmanagement is activated.
IPCOUNTRY	a[2]	*Optional** 2-letter ISO 3166 country code, e.g. CH, DE, AT… Contains the IP geo location country of the customer's IP address. If the country cannot be retrieved, IPCOUNTRY will be empty or contain "IX".
CCCOUNTRY	a[2]	*Optional** Country of origin of card according to ISO 3166. If an assignment is not possible, CCCOUNTRY will not be included in the response. Example: "DE"
MPI_LIABILITYSHIFT	a[..3]	*Optional*** Indicates whether technically formal liability shift is granted. Values: "yes" or "no" **Attention!** Not all processors can check the liability shift during the authorization and so exclude it already within the authorization response if appropriate. Therefore it is possible that even if MPI_LIABILITYSHIFT and ECI indicate an existing liability shift the processor might refuse it for contractual reasons. In case of this concerning questions please contact your processor for further information.
ECI	n[1]	*Optional*** Electronic Commerce Indicator Is needed for the flagging of 3-D Secure transactions („Verified by Visa", „MasterCard SecureCode"): 0 = SSL secure internet payment, no liability shift. 1 = SSL secure internet payment with 3DS and liability shift, customer is taking part in the process. 2 = SSL secure internet payment with 3DS and liability shift, customer is not taking part in the process.
XID	ans[28]	*Optional*** Extra identifier This base64 string is generated by the MPI and references to the instance within the 3-D Secure protocol.
CAVV	ans[28]	*Optional*** Cardholder Authentication Verification Value For a MasterCard the UCAF value is contained, for American Express the AEVV value is used. Saferpay, independently from the credit card type, uses the value CAVV.
IBAN	an[22]	Contains the IBAN of the request. e.g. "DE77970000010123456789".
MANDATEID	ans[..35]	Contains the mandate reference of an ELV payment.
CREDITORID	ans[..35]	Contains the creditor identifier of an ELV payment.

* Only available with Saferpay Risk Management.

*** Required condition is the participance in the 3D-Secure process. ("Verified by Visa","MasterCard SecureCode", "American Express SafeKey")*

ATTENTION! For protection against manipulation it is **strongly recommended** not to check the signature only but also the parameters **ACCOUNTID**, **ORDERID**, **AMOUNT** and **CURRENCY** that are returned with PayConfirm message. They have to contain the same values set to the PayInit message before!

6.1.4 Address parameter for the use of Masterpass

If you use the Masterpass Wallet, then the customer's address data will be transferred along with the rest of the data through the NOTIFYURL.

Parameter	Format	Description
GENDER	a[1]	Gender Values: "f", "m"
FIRSTNAME	ans[..50]	First name
LASTNAME	ans[..50]	Last name
DATEOFBIRTH	n[8]	Cardholder's date of birth
STREET	ans[..50]	Street
ZIP	an[..10]	Postal code
CITY	ans[..50]	City
COUNTRY	a[2]	Country identification according to ISO 3166.
STATE	a[2]	Identification of a state or a province. Corresponds to the last two characters according to ISO 3166-2.
EMAIL	ans[..50]	E-mail address
PHONE	ns[..20]	Phone number
ADDRESSADDITION	an[..50]	Additional address parameter
DELIVERY_GENDER	a[1]	Delivery address gender
DELIVERY_FIRSTNAME	ans[..50]	Delivery address first name
DELIVERY_LASTNAME	ans[..50]	Delivery address last name
DELIVERY_STREET	ans[..50]	Delivery address street
DELIVERY_ADDRESSADDITION	an[..50]	Supplement to delivery address
DELIVERY_ZIP	an[..10]	Delivery address postal code
DELIVERY_CITY	ans[..50]	Delivery address city
DELIVERY_COUNTRY	a[2]	Delivery address country identification according to ISO 3166.
DELIVERY_PHONE	ns[..50]	Delivery address phone number

6.2 VerifyPayConfirm request

The VerifyPayConfirm request to verify the signature of the authorization response contains following parameters:

Parameter	Format	Beschreibung
DATA	ans[..1024]	Unmodified XML message that was received withPayConfirm message.
SIGNATURE	an[28]	Contains the signature of DATA.
ACCOUNTID	ns[..15]	*Optional when using a LIB* The merchant's Saferpay account number of the transaction. For instance "401860-17795278" for Saferpay test account.

6.3 PayComplete request

This table lists the available parameters for the CreatePayComplete message.
If not specified as *Optional* the parameters are mandatory.

Parameter	Format	Description
ID	an[28]	Saferpay transaction identifier returned by the PayConfirm Message. Mandatory parameter excepted if ACTION=CloseBatch
AMOUNT	n [..8]	*Optional* Amount to settle in minor currency unit e.g. “1230” corresponds to 12.30 in EUR
ACCOUNTID	ns[..15]	The Saferpay account identifier of the merchant for this transaction. e.g. 401860-17795278 for the Saferpay Test Account.
ACTION		*Optional* Is used for special processing options. Possible values are: "Settlement", "CloseBatch", "Cancel" Settlement Instructs the Saferpay system to change the status of the transaction from “Reservation“ to “Payment”. Payments are forwarded by the next Batch Close to the concerned processor engaging the actual fund transfer. With the parameter AMOUNT an amount inferior than reserved can be settled. The difference between reserved and settled amount is canceled. The reserved amount is the maximum amount that can be settled. Batch Close Instructs the Saferpay system to engage the Batch Close for the specified ACCOUNTID. If no ACCOUNTID is specified the call fails. Cancel With that call a Reservation can be discarded or a Payment canceled as long as it has not been processed by the Batch Close. A discarded Reservation remains visible for 6 days in „Discarded Reservations” in the Backoffice after what it will be erased from the database. Whereas canceled Payments, flagt as “Cancellation Payment”, will remain visible in the backoffice. If the ACTION parameter is not submitted the default value ACTION=“Settlement“ is used.

6.4 PayComplete Response

The response to the PayComplete request contains the following parameters:

Parameter	Format	Description
MSGTYPE	a[..30]	Alway contains the value “PayConfirm“.
ID	an[28]	Saferpay transaction identifier.
RESULT	n[..4]	Contains the Result of the PayComplete request. 0 = request successfully processed. ≠0 = request not successfully processed.
MESSAGE	ans[..30]	Contains a textual response to the settlement request.
AUTHMESSAGE	ans[..30]	Can contain additional textual information about the request processing.

7 Adjusting using cascading style sheets

The Payment Page appearance can be completely changed to a mobile view using a CSS file. The CSS file has no influence on the structure of the Payment Page desktop view.This can be adjusted via the **VTCONFIG** parameter and a corresponding template under "Payment Page Configuration" to Saferpay Backoffice.

7.1 Styling options

There are two options available to adjust Payment Page with a style sheet.

7.1.1 Adjust Saferpay Mobile Payment Page

The first option uses the PayInit parameter APPEARANCE= "mobile" and the path to the CSS file is transferred to the PayInit parameter CSSURL.

Example:

```
APPEARANCE=”mobile”
CSSURL=”https://link.to.mydesign.css”
```

The mobile Payment Page is subsequently loaded in standard Saferpay layout and the designs required by Saferpay are overwritten with the CSS file.

7.1.2 Completely new design for Payment Page

As an alternative to the first option, the PayInit parameter APPEARANCE="embedded" can be used with the PayInit parameter CSSURL.

Example:

```
APPEARANCE=“embedded”
CSSURL=”https://link.to.mydesign.css”
```

The mobile Payment Page appears in the browser's default HTML view and can be adjusted from the ground up with the help of the CSS file. This normalises the HTML elements so that they are displayed in a uniform fashion in keeping with modern standards.

7.2 Size of the inline frame

The width and height of the iframe are passed on to the shop website via the HTML5 **postMessage** function. The details are in pixels.

JSON-example:

```
{
   "message":"css",
   "height":450,
   "width":650
}
```

Example of the message received (jQuery):

```
$(window).bind("message", function (e) {
        $("#iframe").css("height", e.originalEvent.data.height + "px");
});
```

As not every website transfers information with regard to its size, we would recommend defining a minimum height and width on the shop page.

7.3 Using CSS

The following information should be observed when adjusting Payment Page with the help of CSS.

7.3.1 Element name

The element name may be used in accordance with CSS specifications.

Example:

```
h1{
    text-decoration: underline;
  }
```

7.3.2 Class name

The CSS class names defined by Saferpay in Section 7.5 should be used.

Example:

```
.form-group
{
font-family: "Times New Roman", serif;
font-style: italic;
}
```

7.3.3 Element ID

The Element ID should not be used as the IDs can change without prior notice.

7.3.4 Element attributes

Element attributes should not be used as the attributes (name, value, data*, etc.) can change without prior notice.

7.3.5 CSS selectors

All CSS selectors are generally supported for CSS1, CSS2 and CSS3.

7.4 Important information for using CSS

- The style sheet referred to with the PayInit parameter **CSSURL** must be stored on a web server that supports https.
- Graphics must be loaded within the style sheet via https://, otherwise a warning appears in the browser. For example: *"[…] this page includes other resources which are not secure. […].*
- If the browser blocks the shopper's third party cookies, the Saferpay Payment Page indicates "*Cookies are disabled*".
- We would recommend displaying a progress bar while loading the iframe.
- We would also recommend breaking out of the frame to the success, abort or fail page when returning to the shop.

7.5 CSS class names

7.5.1 General classes

Html

- saferpay-paymentpage

Header

- img-logo

Footer

- btn-back
- btn-abort
- btn-next

- navitem
- navitem-back
- navitem-abort
- navitem-next

Display boxes

- box-content
- box-shop
- box-information
- box-success
- box-error

-

- box-information-required-fields

- img-shop

- text-information
- text-success
- text-error

- icon-information
- icon-success
- icon-error

Form

- form-group
- form-label
- form-input

- form-col-small
- form-col-large

- input-required
- input-large
- input-medium
- input-small

- icon-required

Form validation

- validation-summary-errors
- label-validation-error
- input-validation-error

Basic classes

- box
- btn
- img
- icon
- text

7.5.2 Specifically for payment selection

- page-paymentselection
- paymentgroup
  - paymentgroup-creditcard
  - paymentgroup-onlinebanking
  - paymentgroup-card
  - paymentgroup-onlinepaymentservice
  - paymentgroup-directdebit
  - paymentgroup-invoice
  - paymentgroup-wallet
- btn-select
- btn-creditcard-visa
- btn-creditcard-mastercard
- btn-creditcard-maestro
- btn-creditcard-amex
- btn-creditcard-jcb
- btn-creditcard-dinersclub
- btn-creditcard-saferpay
- btn-onlinebanking-sofort
- btn-onlinebanking-giropay
- btn-onlinebanking-ideal
- btn-onlinebanking-eps
- btn-onlinebanking-post
- btn-onlinebanking- eprzelewy
- btn-card-myone
- btn-card-bonuscard
- btn-card-postcard
- btn-card-lasercard
- btn-directdebit-zvt
- btn-directdebit-intercardelv
- btn-directdebit-billpay
- btn-invoice-billpay
- btn-wallet-masterpass

7.5.3 Specifically for card form

- page-card
  - page-creditcard-visa
  - page-creditcard-mastercard
  - page-creditcard-maestro
  - page-creditcard-amex
  - page-creditcard-jcb
  - page-creditcard-dinersclub
  - page-creditcard-saferpay
  - page-card-myone
  - page-card-bonuscard
  - page-card-postcard
  - page-card-lasercard
- text-hint
- icon-hint

7.5.4 Specifically for direct debit form

- page-directdebit
  - page-directdebit-billpay
  - page-directdebit-intercardelv
  - page-directdebit-zvt
  - -

7.5.5 Specifically for online banking

- page-onlinebanking
  - page-onlinebanking-ideal
  - page-onlinebanking-giropay

7.5.6 Specifically for billing form

- page-invoice
  - page-invoice-billpay

7.5.7 Specifically for address form (incl. Billpay address form)

- page-address
- form-col-small
- form-col-large

7.5.8 Specifically for GTC

- page-termsandconditions

7.5.9 Specifically for redirect pages (incl. 3D-Secure)

- page-redirect

7.5.10 Specifically for error pages

- page-error

7.5.11 Specifically for confirmation page

- page-confirmation

7.6 **CSS examples**

The following URLs will lead you to some CSS-examples, that you can use to test the SSaferpay Payment Page, but please note, that those are only examples. SIX Payment Services will not give any guarantee for a flawless run, when going live.
However, you are allowed to alter these examples for your own needs to use them in your productive environment.

- Mobile:

https://www.six-payment-services.com/dam/saferpay/testaccount/css/mobile1.css

-

- Embedded:

https://www.six-payment-services.com/dam/saferpay/testaccount/css/embedded1.css
https://www.six-payment-services.com/dam/saferpay/testaccount/css/embedded2.css

8 Saferpay Test Environment

For the integration phase and in order to be able to test Saferpay, we can offer you our External Test Environment (ETU).
In this environment, which is isolated from the operational environment, you can test Saferpay with simulations for all current payment means in your own test account.

All the details on our test environment can be found at the following address:

https://www.six-payment-services.com/en/site/saferpay-support/testaccount.html

9 Examples

9.1 Important Note

Please note that own values should always be transmitted html encoded (or as html entity or as Unicode) in order to ensure that all special characters are transmitted to Saferpay correctly.

9.2 C# mit der .NET LIB

Generating the payment link with CreatePayInit:

```
MessageFactory mf = new MessageFactory();
mf.Open(""); // Saferpay configuration path, e.g. "c:\\Programme\\Saferpay\\Client"
mo_payinit = mf.CreatePayInit();

string m_accountid = "401860-17795278";
string m_amount = "2095";
string m_currency = "EUR";
string m_description = "Test Einkauf";
string m_address = "no";
string m_orderid = "0815-4711";
string m_backlink = "http://www.myshop.com/back.aspx";
string m_faillink = "http://www.myshop.com/Fail.aspx";
string m_successlink = "http://www.myshop.com/Success.aspx";
string m_notifyurl = "http://www.myshop.com/notify.aspx";

mo_payinit.SetAttribute("ACCOUNTID", m_accountid);
mo_payinit.SetAttribute("AMOUNT", m_amount);
mo_payinit.SetAttribute("CURRENCY", m_currency);
mo_payinit.SetAttribute("DELIVERY", m_address);
mo_payinit.SetAttribute("ORDERID", Server.HtmlEncode(m_orderid));
mo_payinit.SetAttribute("DESCRIPTION", Server.HtmlEncode(m_description));
mo_payinit.SetAttribute("SUCCESSLINK", m_successlink);
mo_payinit.SetAttribute("BACKLINK", m_backlink);
mo_payinit.SetAttribute("FAILLINK", m_faillink);
mo_payinit.SetAttribute("NOTIFYURL", m_notifyurl);

string paymenturl = mo_payinit.GetPostUrl();
string data = mo_payinit.GetPostData();
string signature = mo_payinit.GetPostSignature();
```

Calling payment URL with a form:

```
<html>
<head><title>Zahlung mit der Saferpay Payment Page</title></head>
<body>
<h2> Beispiel - Saferpay Payment Page per POST aufrufen</h2>
 <form action="<%=paymenturl %>"method="POST">
   <input type="hidden" name="DATA" value="<%=data %>">
   <input type="hidden" name="SIGNATURE" value="<%=signature %>" >
   <input type="submit" value="Bezahlen">
 </form>
</body>
</html>
```

Verifying the PayConfirm message:

```
string data = Request.QueryString.Get("DATA");
string signature = Request.QueryString.Get("SIGNATURE");

MessageFactory mf = new MessageFactory();
mf.Open("");
mo_payconfirm = mf.VerifyPayConfirm(data, signature);

string id = mo_payconfirm.GetAttribute(ID);
```

Settlement of the payment CreatePayComplete:

```
MessageFactory mf = new MessageFactory();
mf.Open("");
MessageObject mo_paycomplete = mf.CreateRequest("PayComplete");

mo_paycomplete.SetAttribute("ID", id);
mo_paycomplete.SetAttribute("ACCOUNTID", m_accountid);

MessageObject captureresponse = mo_paycomplete.Capture();
```

PayComplete response:

```
int result = Convert.ToInt32(response.GetAttribute("RESULT"));
if (result == 0)
{
        String id = captureresponse.GetAttribute("ID");
        String msg = captureresponse.GetAttribute("MESSAGE");
        Console.WriteLine("Verbuchung erfolgreich!");
}
else
{
        Console.WriteLine("Verbuchung fehlgeschlagen!");
        return;
}
```

9.3 Java mit der Java LIB

Generating the payment link with CreatePayInit:

```
import Saferpay.*
import org.apache.commons.lang.*

MessageFactory mf = new MessageFactory();
mf.Open(""); // Saferpay configuration path, e.g. "c:\\Programme\\Saferpay\\Client"
MessageObject mo_payinit = mf.CreatePayInit();

String m_accountid = "401860-17795278";
String m_amount = "2095";
String m_currency = "EUR";
String m_description = "Test Einkauf";
string m_address = "no";
String m_orderid = "0815-4711";
String m_backlink = "http://www.myshop.com/back.jsp";
String m_faillink = "http://www.myshop.com/Fail.jsp";
String m_successlink = "http://www.myshop.com/Success.jsp";
String m_notifyurl = "http://www.myshop.com/notify.jsp";

mo_payinit.SetAttribute("ACCOUNTID", m_accountid);
mo_payinit.SetAttribute("AMOUNT", m_amount);
mo_payinit.SetAttribute("CURRENCY", m_currency);
mo_payinit.SetAttribute("DELIVERY", m_address);
mo_payinit.SetAttribute("ORDERID", StringEscapeUtils.escapeHtml(m_orderid));
mo_payinit.SetAttribute("DESCRIPTION", StringEscapeUtils.escapeHtml(m_description));
mo_payinit.SetAttribute("SUCCESSLINK", m_successlink);
mo_payinit.SetAttribute("BACKLINK", m_backlink);
mo_payinit.SetAttribute("FAILLINK", m_faillink);
mo_payinit.SetAttribute("NOTIFYURL", m_notifyurl);

String paymenturl = mo_payinit.GetPostUrl();
String data = mo_payinit.GetPostData();
String signature = mo_payinit.GetPostSignature();
```

Calling payment URL with a form:

```
<html>
<head><title>Purchase with Saferpay Payment Page</title></head>
<body>
<h2> Beispiel – Calling Saferpay Payment Page by POST</h2>
 <form action="<%=paymenturl %>"method="POST">
   <input type="hidden" name="DATA" value="<%=data %>">
   <input type="hidden" name="SIGNATURE" value="<%=signature %>" >
   <input type="submit" value="Purchase">
 </form>
</body>
</html>
```

Verifying the PayConfirm message:

```
import Saferpay.*

String data = request.getParameter("DATA");
String signature = request.getParameter("SIGNATURE");

MessageFactory mf = new MessageFactory();
mf.Open("");
MessageObject mo_payconfirm = mf.VerifyPayConfirm(data, signature);

String id = mo_payconfirm.GetAttribute(ID);
```

Settlement of the payment CreatePayComplete:

```
MessageFactory mf = new MessageFactory();
mf.Open("");
MessageObject mo_paycomplete = mf.CreatePayComplete(ID, "");

mo_paycomplete.SetAttribute("ID", id);
mo_paycomplete.SetAttribute("ACCOUNTID", m_accountid);

MessageObject captureresponse = mo_paycomplete.Capture();
```

PayComplete response

```
int result = response.GetAttribute("RESULT");
if (result == 0)
{
        String id = captureresponse.GetAttribute("ID");
        String msg = captureresponseresponse.GetAttribute("MESSAGE");
        System.out.println("Verbuchung erfolgreich!");
}
else
{
        System.out.println("Verbuchung fehlgeschlagen!");
        return;
}
```

9.4 Command Line Calls with the Java LIB

Generating the payment link with CreatePayInit:

```
java -jar Saferpay.jar -payinit -p C:\Programme\Saferpay\Client -a AMOUNT 1930 -a CURRENCY
EUR -a DELIVERY no -a ACCOUNTID 401860-17795278 -a DESCRIPTION "Test Einkauf" -a ORDERID
0815-4711 -a FAILLINK "http://www.testshop.de/fail.php" -a SUCCESSLINK
"http://www.testshop.de/success.php" -a BACKLINK "http://www.testshop.de/back.php" -a
NOTIFYURL "http://www.testshop.de/log.php"
```

Created payment link:

https://test.saferpay.com/vt2/Pay.aspx?DATA=%3CIDP+ALLOWCOLLECT%3D%22no%22+EXPIRATION%3D%2220110325+16%3A06%3A53%22+DESCRIPTION%3D%22Test+Einkauf%22+BACKLINK%3D%22http%3A%2F%2Fwww.testshop.de%2Fback.php%22+AMOUNT%3D%221930%22+DELIVERY%3D%22no%22+ACCOUNTID%3D%22401860-17795278%22+SUCCESSLINK%3D%22http%3A%2F%2Fwww.testshop.de%2Fsuccess.php%22+CURRENCY%3D%22EUR%22+ORDERID%3D%220815-4711%22+FAILLINK%3D%22http%3A%2F%2Fwww.testshop.de%2Ffail.php%22+MSGTYPE%3D%22PayInit%22+KEYID%3D%220-99867-959af13b7be94dd99cbd20ac7caa5888%22+NOTIFYURL%3D%22http%3A%2F%2Fwww.testshop.de%2Flog.php%22+TOKEN%3D%228ae94d50562c01e0d24085728e033992%22%2F%3E&SIGNATURE=38FE7F5483578D24EDC3F9A8D00CC1B87733B5CE1F984F82E2ED084C52B01B0E783C8007F022610D93E26E2AA254C02245A09A0F25A0C5A4961B9FDFB9FF8F50

Verifying the PayConfirm message:

Redirection to the shop after successful authorization via the SUCCESSLINK:

```
http://www.testshop.de/success.php?DATA=%3CIDP+MSGTYPE%3d%22PayConfirm%22+TOKEN%3d%22%28unuse
d%29%22+VTVERIFY%3d%22%28obsolete%29%22+KEYID%3d%221-
0%22+ID%3d%22brlb01AS3AphUA4fnAr0bQUS7thA%22+ACCOUNTID%3d%22401860-
17795278%22+PROVIDERID%3d%2290%22+PROVIDERNAME%3d%22Saferpay+Test+Card%22+ORDERID%3d%220815-
4711%22+AMOUNT%3d%221930%22+CURRENCY%3d%22EUR%22+IP%3d%22193.247.180.193%22+IPCOUNTRY%3d%22CH
%22+CCCOUNTRY%3d%22XX%22+MPI_LIABILITYSHIFT%3d%22yes%22+MPI_XID%3d%22NEpiXQIKWz8xBBszF38FVUUw
TAo%3d%22+ECI%3d%222%22+XID%3d%22NEpiXQIKWz8xBBszF38FVUUwTAo%3d%22+%2f%3E&SIGNATURE=b43f0ac94
ce260fa520010558dc552ef1c67a839538bd2346d6e593fa74b3e2db22e6de0f494a312bf3436af662219144dda2b
f472a5447da205009668a791fa
```

Received DATA:

```
<IDP MSGTYPE="PayConfirm" TOKEN="(unused)" VTVERIFY="(obsolete)" KEYID="1-0"
ID="brlb01AS3AphUA4fnAr0bQUS7thA" ACCOUNTID="401860-17795278" PROVIDERID="90"
PROVIDERNAME="Saferpay Test Card" ORDERID="0815-4711" AMOUNT="1930" CURRENCY="EUR"
IP="193.247.180.193" IPCOUNTRY="CH" CCCOUNTRY="XX" MPI_LIABILITYSHIFT="yes"
MPI_XID="NEpiXQIKWz8xBBszF38FVUUwTAo=" ECI="2" XID="NEpiXQIKWz8xBBszF38FVUUwTAo=" />
```

Received SIGNATURE:

```
b43f0ac94ce260fa520010558dc552ef1c67a839538bd2346d6e593fa74b3e2db22e6de0f494a312bf3436af66221
9144dda2bf472a5447da205009668a791fa
```

Executing VerifyPayConfirm:

```
java -jar Saferpay.jar -payconfirm -p C:\Programme\Saferpay\Client -d
%3CIDP+MSGTYPE%3d%22PayConfirm%22+TOKEN%3d%22%28unused%29%22+VTVERIFY%3d%22%28obsolete%29%22+
KEYID%3d%221-0%22+ID%3d%22brlb01AS3AphUA4fnAr0bQUS7thA%22+ACCOUNTID%3d%22401860-
17795278%22+PROVIDERID%3d%2290%22+PROVIDERNAME%3d%22Saferpay+Test+Card%22+ORDERID%3d%220815-
4711%22+AMOUNT%3d%221930%22+CURRENCY%3d%22EUR%22+IP%3d%22193.247.180.193%22+IPCOUNTRY%3d%22CH
%22+CCCOUNTRY%3d%22XX%22+MPI_LIABILITYSHIFT%3d%22yes%22+MPI_XID%3d%22NEpiXQIKWz8xBBszF38FVUUw
TAo%3d%22+ECI%3d%222%22+XID%3d%22NEpiXQIKWz8xBBszF38FVUUwTAo%3d%22+%2f%3E -s
b43f0ac94ce260fa520010558dc552ef1c67a839538bd2346d6e593fa74b3e2db22e6de0f494a312bf3436af66221
9144dda2bf472a5447da205009668a791fa
```

Settlement of the Payment with CreatePayComplete:

```
java -jar Saferpay.jar -capture -p C:\Programme\Saferpay\Client –i
brlb01AS3AphUA4fnAr0bQUS7thA –a ACCOUNTID 401860-17795278 –of capt.txt
```

9.5 https Interface

Generating the Payment Link with CreatePayInit:

```
https://test.saferpay.com/hosting/CreatePayInit.asp?ACCOUNTID=401860-
17795278&ORDERID=123456789-001&AMOUNT=1000&CURRENCY=EUR&DESCRIPTION=Testkauf&SUCCESSLINK="htt
p://www.myshop.com/Success.aspx"&FAILLINK="http://www.myshop.com/Fail.aspx"&BACKLINK="http://
www.myshop.com/back.aspx"&NOTIFYURL="http://www.myshop.com/notify.aspx"
```

Response returns the Payment Link:

```
https://test.saferpay.com/vt2/Pay.aspx?DATA=%3cIDP+MSGTYPE%3d%22PayInit%22+MSG_GUID%3d%22939a
4c930b5c482588d91f54f74ac110%22+CLIENTVERSION%3d%222.0%22+KEYID%3d%220-99867-7d5a273c0f5043e2
8811e764d6433086%22+TOKEN%3d%22bbf6577cd8e74d65a27f084c9cfe2592%22+ALLOWCOLLECT%3d%22no%22+DE
LIVERY%3d%22no%22+EXPIRATION%3d%2220110625+12%3a01%3a56%22+ACCOUNTID%3d%22401860-
17795278%22+AMOUNT%3d%221000%22+CURRENCY%3d%22EUR%22+DESCRIPTION%3d%22Testkauf%22+SUCCESSLINK
%3d%22http%3a%2f%2fwww.myshop.com%2fSuccess.aspx%22+BACKLINK%3d%22http%3a%2f%2fwww.myshop.com
%2fback.aspx%22+FAILLINK%3d%22http%3a%2f%2fwww.myshop.com%2fFail.aspx%22+ORDERID%3d%221234567
89-001%22+CCNAME%3d%22yes%22+NOTIFYURL%3d%22http%3a%2f%2fwww.myshop.com%2fnotify.aspx%22+%2f%
3e&SIGNATURE=428b356c87f7fdcf44417f670197c4a6395385e623e224653610b94db8acc1ead509b7b5e6dfc465
dcf987b3cf4b284fc799ee93ff9cb151c3bc9981e8320232
```

Verifying the PayConfirm Message:

Redirection to the shop after successful authorization via the SUCCESSLINK:

```
http://www.myshop.com/Success.aspx?DATA=%3CIDP+MSGTYPE%3d%22PayConfirm%22+TOKEN%3d%22%28unuse
d%29%22+VTVERIFY%3d%22%28obsolete%29%22+KEYID%3d%221-0%22+ID%3d%22A668MSAprOj4tAzv7G9lAQUfUr3
A%22+ACCOUNTID%3d%22401860-
17795278%22+PROVIDERID%3d%2290%22+PROVIDERNAME%3d%22Saferpay+Test+Card%22+ORDERID%3d%22123456
789-001%22+AMOUNT%3d%221000%22+CURRENCY%3d%22EUR%22+IP%3d%22193.247.180.193%22+IPCOUNTRY%3d%2
2CH%22+CCCOUNTRY%3d%22XX%22+MPI_LIABILITYSHIFT%3d%22yes%22+MPI_TX_CAVV%3d%22AAABBIIFmAAAAAAAA
AAAAAAAAAA%3d%22+MPI_XID%3d%22CxMTYwhoUXtCBAEndBULcRIQaAY%3d%22+ECI%3d%221%22+CAVV%3d%22AAABB
IIFmAAAAAAAAAAAAAAAAAA%3d%22+XID%3d%22CxMTYwhoUXtCBAEndBULcRIQaAY%3d%22+%2f%3E&SIGNATURE=7b2b
b163f4ef86d969d992b4e2d61ad48d3b9022e0ec68177e35fe53184e6b3399730d1a3641d2a984ce38699daad72ab
006d5d6a9565c5ae1cff8bdc8a1eb63
```

Received DATA:

```
<IDP MSGTYPE="PayConfirm" TOKEN="(unused)" VTVERIFY="(obsolete)" KEYID="1-0"
ID="A668MSAprOj4tAzv7G9lAQUfUr3A" ACCOUNTID="401860-17795278" PROVIDERID="90"
PROVIDERNAME="Saferpay Test Card" ORDERID="123456789-001" AMOUNT="1000" CURRENCY="EUR"
IP="193.247.180.193" IPCOUNTRY="CH" CCCOUNTRY="XX" MPI_LIABILITYSHIFT="yes"
MPI_TX_CAVV="AAABBIIFmAAAAAAAAAAAAAAAAAA=" MPI_XID="CxMTYwhoUXtCBAEndBULcRIQaAY=" ECI="1"
CAVV="AAABBIIFmAAAAAAAAAAAAAAAAAA=" XID="CxMTYwhoUXtCBAEndBULcRIQaAY=" />
```

Received SIGNATURE:

```
7b2bb163f4ef86d969d992b4e2d61ad48d3b9022e0ec68177e35fe53184e6b3399730d1a3641d2a984ce38699daad
72ab006d5d6a9565c5ae1cff8bdc8a1eb63
```

https call VerifyPayConfirm:

```
https://test.saferpay.com/hosting/verifypayconfirmVerifyPayConfirm.asp?spPassword=8e7Yn5yk&AC
COUNTID=401860-
17795278&DATA=%3CIDP+MSGTYPE%3d%22PayConfirm%22+TOKEN%3d%22%28unused%29%22+VTVERIFY%3d%22%28o
bsolete%29%22+KEYID%3d%221-0%22+ID%3d%22A668MSAprOj4tAzv7G9lAQUfUr3A%22+ACCOUNTID%3d%22401860
-
17795278%22+PROVIDERID%3d%2290%22+PROVIDERNAME%3d%22Saferpay+Test+Card%22+ORDERID%3d%22123456
789-001%22+AMOUNT%3d%221000%22+CURRENCY%3d%22EUR%22+IP%3d%22193.247.180.193%22+IPCOUNTRY%3d%2
2CH%22+CCCOUNTRY%3d%22XX%22+MPI_LIABILITYSHIFT%3d%22yes%22+MPI_TX_CAVV%3d%22AAABBIIFmAAAAAAAA
AAAAAAAAAA%3d%22+MPI_XID%3d%22CxMTYwhoUXtCBAEndBULcRIQaAY%3d%22+ECI%3d%221%22+CAVV%3d%22AAABB
IIFmAAAAAAAAAAAAAAAAAA%3d%22+XID%3d%22CxMTYwhoUXtCBAEndBULcRIQaAY%3d%22+%2f%3E&SIGNATURE=7b2b
b163f4ef86d969d992b4e2d61ad48d3b9022e0ec68177e35fe53184e6b3399730d1a3641d2a984ce38699daad72ab
006d5d6a9565c5ae1cff8bdc8a1eb63
```

Transaction ID returned within the response:

```
OK:ID=A668MSAprOj4tAzv7G9lAQUfUr3A&TOKEN=(unused)
```

Settlement of the Payment with CreatePayComplete:

```
https://test.saferpay.com/hosting/PpayCcompleteVv2.asp?spPassword=8e7Yn5yk&ACCOUNTID=401860-
17795278&ID=A668MSAprOj4tAzv7G9lAQUfUr3A
```

Response returns the result:

```
OK:<IDP RESULT="0" MESSAGE="request was processed successfully" ID="xxxxxxxxxx"
MSGTYPE="PayConfirm"/>
```

10 Error-Codes

The following table lists possible error codes which can be returned by the Saferpay Library.

Code	Name	Description
Common Errors		
0x80040201	Context Missing	No configuration context has been specified. Call the Open function before calling any other function.
0x80040202	File Not Found	A file is missing in the configuration. Check if the component was properly configured on the system.
0x80040203	File Access Error	A security violation occurred during file access. Check permissions on configuration directories.
0x80040204	Invalid File Format	File with no valid XML content.
0x80040206	Invalid Path	An invalid path or URL was specified.
0x80040207	Invalid Option	An invalid Option value was specified.
0x80040208	Request Failed	A synchonous server request failed.
0x80040209	Cryptographic Error	An invalid key or key-identifier was specified in the message.
0x8004020f	No Configuration	No configuration found in the path specified by Open.
0x80040214	Verify Failed	The verification of a signature failed because the signature was invalid.
HTTP Error		
0x80042019	Invalid URL	An invalid URL was specified in a synchronous call to the component.
0x80042021	DNS Error	An error occurred while looking up a hostname. Most likely the reason for this problem will be a DNS configuration problem or missing proxy settings.
0x80042xxx	HTTP Server Error	The HTTP Server returned a unsuccessful status code. The errorcode & 0xfff will be the status code returned by the server.
Socket Error		
0x80043xxx	Socket Base	Base for socket errors. The resulting error code is the base error number plus the socket error code.

11 Contact

11.1 Saferpay Integration Team

Do you have questions about this document or problems with the integration of Saferpay or do you need assistance? Then please contact our integration team:

Saferpay Switzerland
SIX Payment Services AG
Hardturmstrasse 201
8021 Zürich
+41 848 66 44 44
www.six-payment-services.com/saferpay
integration.saferpay@six-payment-services.com

Saferpay Europe
SIX Payment Services (Germany) GmbH
Langenhorner Chaussee 92-94
22415 Hamburg
+49 40 325 967- 280
www.six-payment-services.com/saferpay
integration.saferpay@six-payment-services.com

11.2 Saferpay Support Team

Do you have questions about error messages or do you encounter problems with your running system? Then please contact our support team:

Saferpay Switzerland
SIX Payment Services AG
Hardturmstrasse 201
8021 Zürich
+41 848 66 44 44
www.six-payment-services.com/saferpay
support.saferpay@six-payment-services.com

Saferpay Europe
SIX Payment Services (Germany) GmbH
Langenhorner Chaussee 92-94
22415 Hamburg
+49 40 325 967- 250
www.six-payment-services.com/saferpay
support.saferpay@six-payment-services.com

The Saferpay team wishes you every success with your Saferpay e-payment solution!