Wirecard CEE Integration Documentation

() -

Created: 2020-07-13 01:54

Storing Sensitive Payment Data in Wirecard Data Storage

Prerequisite

- Initialization of Wirecard data storage

Storing sensitive payment data

Before using the store operation please ensure that you include the returned JavaScript URL from the return parameter of the data storage initialization within your HTML page. We recommend to load the JavaScript at the bottom of the page to speed up the page loading time.

For example, include the returned JavaScript URL in PHP by the following code fragment, where `$javascriptURL` is the URL you have to include to your HTML page for using the JavaScript based storing functions and objects.

```
 <script src="<?php echo $javascriptURL; ?>"
type="text/javascript"></script>
```

Then you need to create an instance of the JavaScript class WirecardCEE_DataStorage, which is defined within the previously included URL.

```
var dataStorage = new WirecardCEE_DataStorage();
```

After this you create a new and empty JavaScript object which will be filled with the sensitive payment data:

```
var paymentInformation = {};
```

Depending on the payment method you add various properties to this JavaScript object, e.g. for Credit Card, Credit Card - Mail Order and Telephone Order, and Maestro SecureCode you add the following properties (the values of these properties are directly taken from your HTML form by accessing the values of the form elements via the DOM of the web browser):

```
paymentInformation.pan = document.getElementById('cc_pan').value;
paymentInformation.expirationMonth = document.getElementById(
'cc_expirationMonth').value;
paymentInformation.expirationYear = document.getElementById(
'cc_expirationYear').value;
paymentInformation.cardholdername = document.getElementById(
```

```
'cc_cardholdername').value;
paymentInformation.cardverifycode = document.getElementById(
'cc_cardverifycode').value;
paymentInformation.issueMonth = document.getElementById('cc_issueMonth').
value;
paymentInformation.issueYear = document.getElementById('cc_issueYear').value;
paymentInformation.issueNumber = document.getElementById('cc_issueNumber').
value;
```

After setting all properties to the paymentInformation object you only need to call a specific JavaScript function which is also available in the JavaScript file you included before:

Credit Card

```
dataStorage.storeCreditCardInformation(paymentInformation, callbackFunction);
```

Credit Card - Mail Order and Telephone Order

```
dataStorage.storeCreditCardMotoInformation(paymentInformation,
callbackFunction);
```

Maestro SecureCode

```
dataStorage.storeMaestroInformation(paymentInformation, callbackFunction);
```

The `callbackFunction` is a JavaScript function where you are able to handle the result of the storage operation.

Within this callback function you are able to use the following JavasScript functions:

Function	Description
getStatus()	Returns 0 for a successful storage operation otherwise 1.
getAnonymizedPaymentInformation()	Returns anonymized payment information as a simple JavaScript object.
getErrors()	Returns the errors as a simple JavaScript object.

A very simplified example of this callback function can be:

```
callbackFunction = function(aResponse) {
  // initiates the result string presented to the user
  var s = "Response of Wirecard data storage call:\n\n";
  // checks if response status is without errors
  if (aResponse.getStatus() == 0) {
    // saves all anonymized payment information to a JavaScript object
    var info = aResponse.getAnonymizedPaymentInformation();

    //Response for paymentTypes: CCARD, CCARD-MOTO, MAESTRO
    s + = "Response for paymentTypes: CCARD, CCARD-MOTO, MAESTRO\n";
    s += "anonymousPan: " + info.anonymousPan + "\n";
    s += "maskedPan: " + info.maskedPan + "\n";
```

```
    s += "financialInstitution: " + info.financialInstitution + "\n";
    s += "brand: " + info.brand + "\n";
    s += "cardholdername: " + info.cardholdername + "\n";
    s += "expiry: " + info.expiry + "\n";

    //Response for paymentType: SEPA-DD
    s + = "Response for paymentType: SEPA-DD\n";
    s += "bankBic: " + info.bankBic + "\n";
    s += "bankAccountIban: " + info.bankAccountIban + "\n";
    s += "accountOwner: " + info.accountOwner + "\n";

    //Response for paymentType: PBX
    s + = "Response for paymentType: PBX\n";
    s += "payerPayboxNumber: " + info.payerPayboxNumber + "\n";

    //Response for paymentType: GIROPAY
    s + = "Response for paymentType: GIROPAY\n";
    s += "accountOwner: " + info.accountOwner + "\n";
    s += "bankAccount: " + info.bankAccount + "\n";
    s += "bankNumber: " + info.bankNumber + "\n";

  } else {
    // collects all occurred errors and adds them to the result string
    var errors = aResponse.getErrors();
    for (e in errors) {
      s += "Error " + e + ": " + errors[e].message + " (Error Code: " +
errors[e].errorCode + ")\n";
    }
  }
  // presents result string to the user
  alert(s);
}
```

Availability of stored data

After the latest call of the store operation or a read operation the data storage for the consumer is valid for 30 minutes. If this time has been exceeded you have to initialize a new data storage for your consumer.

Storing payment specific data

To store the sensitive data of your consumer in the Wirecard data storage you may use the following payment-specific JavaScript functions, which you included as JavaScript file before.

```
dataStorage.storeCreditCardInformation(paymentInformation, callbackFunction);
```

```
dataStorage.storeCreditCardMotoInformation(paymentInformation,
callbackFunction);

dataStorage.storeMaestroInformation(paymentInformation, callbackFunction);

dataStorage.storeSepaDdInformation(paymentInformation, callbackFunction);

dataStorage.storePayboxInformation(paymentInformation, callbackFunction);

dataStorage.storeGiropayInformation(paymentInformation, callbackFunction);

===== Return values in case of an error =====

If the storing did not succeed you will get parameters describing the error
which you may handle in the callback JavaScript function:

^ Parameter                 ^ Data type                              ^ Short
description  ^
| errors                    | Numeric                                | Number
of errors occurred. |
| error.{n}.consumerMessage  | Alphanumeric with special characters. | Error
message in localized language for your consumer. |
| error.{n}.errorCode        | Numeric with a fixed length of 5.    | Numeric
error code which you should log for later use. |
| error.{n}.message          | Alphanumeric with special characters. | Error
message in English. |

For further details, see [[wcs:error_codes|Error Codes]].
===== Parameters for specific payment methods =====


==== Parameters for payment methods Credit Card, Credit Card - Mail Order
and Telephone Order, and MaestroSecureCode ====

=== Required request parameters ===

^ Parameter       ^ Data type                                    ^ Description
^
| pan             | Numeric with a variable length of 13 to 19. | Credit
card number. |
| expirationMonth  | Numeric with a fixed length of 2.           |
Expiration month of credit card. |
| expirationYear   | Numeric with a fixed length of 2.           |
Expiration year of credit card. |


=== Optional request parameters ===

^ Parameter       ^ Data type                                    ^
```

```
Description  ^
| cardholdername  | Alphanumeric with a variable length of up to 30. | Name
of owner of credit card. Optional/required depending on your acceptance
contract with credit card company. |
| cardverifycode  | Numeric with a variable length of 1 to 4.        |
Verification code of credit card (aka CVC, CVC2, CVV or CID). Optional/
required depending on your acceptance contract with credit card company. |
| issueMonth      | Numeric with a fixed length of 2.                |
Issuing month of credit card. |
| issueYear       | Numeric with a fixed length of 2.                |
Issuing year of credit card. |
| issueNumber     | Numeric with a variable length of 1 to 3.        |
Issuing number of credit card. |


Code example for setting the properties within the paymentInformation
JavaScript object:
<code php>
paymentInformation.pan = document.getElementById('cc_pan').value;
paymentInformation.expirationMonth = document.getElementById(
'cc_expirationMonth').value;
paymentInformation.expirationYear = document.getElementById(
'cc_expirationYear').value;
paymentInformation.cardholdername = document.getElementById(
'cc_cardholdername').value;
paymentInformation.cardverifycode = document.getElementById(
'cc_cardverifycode').value;
paymentInformation.issueMonth = document.getElementById('cc_issueMonth').
value;
paymentInformation.issueYear = document.getElementById('cc_issueYear').value;
paymentInformation.issueNumber = document.getElementById('cc_issueNumber').
value;
```

Response parameters

Parameter	Data type	Description
anonymousPan	Numeric with a fixed length of 4.	Anonymized credit card number containing only the rightmost 4 digits.
maskedPan	Numeric with special characters and a variable length of 13 to 19.	Masked credit card number: first 6 numbers followed by * and he last 4 numbers of the credit card.
financialInstitution	Enumeration	Financial institution
brand	Enumeration	Brand depending on payment method.
cardholdername	Alphanumeric with special characters.	Name of card holder.
expiry	Numeric with special characters.	Expiry date of credit card in format MM/YYYY.

Brand

The parameter brand may contain one of the following values:

Value of brand for Credit Cards
American Express
Diners Club
Discover
JCB
Mastercard
UATP
Visa
Maestro SecureCode
Mastercard SecureCode
Verified by Visa

Parameters for payment method SEPA Direct Debit

Required request parameters

Parameter	Data type	Description
accountOwner	Alphanumeric with special characters.	Name of owner of account.
bankAccountIban	Alphanumeric with a variable length of up to 34 characters.	IBAN of account. (International Bank Account Number)
bankBic	Alphanumeric with a variable length of up to 12 characters.	BIC of bank. (Bank Identifier Code) Required only for Hobex. For Wirecard Bank, bankBic is optional.

Optional request parameter

Parameter	Data type	Description
bankName	Alphanumeric with special characters.	Name of bank.

Code example for setting the properties within the paymentInformation JavaScript object:

```
paymentInformation.bankBic = document.getElementById('sepa-dd_bankBic').
value;
paymentInformation.bankAccountIban = document.getElementById(
'sepa-dd_bankAccountIban').value;
paymentInformation.accountOwner = document.getElementById(
'sepa-dd_accountOwner').value;
```

Response parameters

The response parameters are the same as the request parameters.

Parameters for payment method Paybox

Required request parameters

Parameter	Data type	Description
payerPayboxNumber	Numeric with minimum length of 8.	Number of Paybox account starting with 0.

Code example for setting the properties within the paymentInformation JavaScript object:

```
paymentInformation.payerPayboxNumber = document.getElementById(
'payerPayboxNumber').value;
```

Response parameters

The response parameters are the same as the request parameters.

Parameters for payment method giropay

Required request parameters

Parameter	Data type	Description
bankAccount	Numeric	Account number.
bankNumber	Numeric with a length of 8.	Bank number.

Optional request parameters

Parameter	Data type	Description
accountOwner	Alphanumeric with special characters.	Name of account owner.

Code example for setting the properties within the paymentInformation JavaScript object:

```
paymentInformation.accountOwner = document.getElementById(
'giropay_accountOwner').value;
paymentInformation.bankAccount = document.getElementById(
'giropay_bankAccount').value;
paymentInformation.bankNumber = document.getElementById('giropay_bankNumber').
value;
```

Response parameters

The response parameters are the same as the request parameters.

Response parameter

The response parameter is the same as the request parameter.

Next step

- Reading stored payment data from Wirecard data storage