**CHECKOUT APIs.**

Country

Denmark

Nertherland

Norway.

Austria.

Finland.

Germany.

Sweden.

Introduction

In order to integrate Klarna to your checkout you will need to use our API methods. To create an order you will need to use the function [reserve\_amount](https://developers.klarna.com/en/nl+php/kpm/checkout-api#reserve_amount).

Sandbox and live environment access information

To test Klarna's services, you need to [apply for API credentials](https://developers.klarna.com/en/nl+php/kpm/apply-for-test-account) for the test environment. The test environment is exactly the same as Klarna's live system, except no real invoices will be created and no credit checks are made.

To connect to the test environment, you need to change the host and port in the Klarna client libraries.

**Host**: payment.testdrive.klarna.com   
**Port**: 443

Digest

A digest is a cryptographic hash function that takes a block of data and returns a fixed-size bit string. The digest is used to verify the integrity of the call as well as to identify the caller, and sent in as a base64 encoded string.

Supported hashing algorithms are, in descending order of recommendation:

* SHA-512
* SHA-384
* SHA-256
* SHA-128
* MD5 (deprecated)

The use of MD5 is strongly discouraged, and has been deprecated.

Error handling

When an error occurs, Klarna will respond with an [error code and a message](https://developers.klarna.com/en/nl+php/kpm/error-codes). The error message language is determined by the language code in the API call. If no language code can be used, Klarna will answer with the language of the consumer.

<?xml version="1.0" encoding="ISO-8859-1"?>

<methodResponse>

<fault>

<value>

<struct>

<member>

<name>faultCode</name>

<value>

<int>8114</int>

</value>

</member>

<member>

<name>faultString</name>

<value>

<string>A problem ...</string>

</value>

</member>

</struct>

</value>

</fault>

</methodResponse>

Methods

get\_addresses

Get\_addresses can be used by Swedish private persons or organizations with one or more registered addresses. Klarna will return all registered addresses based on social security or organization number.

The get\_addresses function may be used as long as the following requirements are met:

* The function can only used by consumers with a Swedish social security number or organization
* Customer data is only retrieved for Klarna's payment methods in the checkout
* The function may not be used for registering customers
* The function may only be used after providing the consumer with Klarna’s terms & conditions
* The get\_addresses function and received data must disappear if the consumer chooses another payment method
* The consumer needs to actively press a button to collect the data, it may not be triggered by completion of a form field
* The button is not allowed to be called get address (hämta address). Approved names: Fetch (Hämta), Continue (Fortsätt), Search (Sök), Proceed (Vidare)

Digest

Consist of the values from eid, pno and secret (in this order) separated with a colon without the square brackets.

Example: base64encode(sha512("[eid]:[pno]:[shared\_secret]"))

Return value

An array of arrays containing the addresses (one array per address). The array contains strings.

Example when using Type 5 or 1  
0 = First name (not returned if company)  
1 = Last name or company name  
2 = Address  
3 = Postno  
4 = City  
5 = Country

Call structure

\* Required

| **Variable** | **Type** | **Description** |
| --- | --- | --- |
| proto\_vsn \* | string | Use 4.1. Automatically set in the library. |
| client\_vsn \* | string | xmlrpc:my-store-name:version-number. Automatically set in the library. |
| eid \* | string | A merchant ID which refers to your store in Klarna's database |
| pno \* | string | Consumer’s social security number. Use the following format:  (SE) yymmdd-nnnn, it can be sent with or without dash "-" or with or without the two first numbers in the year. |
| digest \* | string | Digest for authentication. Klarna’s libraries do this for you together with the shared secret. |
| pno\_encoding \* | int | 2: Sweden |
| type \* | int | How the name will be returned. We recommend type 5.  1: Gives all first names that the person has.  2: Gives only the last name and the consumer fills out the first name.  5: Gives the persons given name if available, otherwise it returns all names. Recommended |
| client\_ip \* | string | The IP address of the consumer who initiates the call |

Example

<?xml version="1.0" encoding="ISO-8859-1"?>

<methodCall>

<methodName>get\_addresses</methodName>

<params>

<param>

<value>

<!-- proto\_vsn -->

<string>4.1</string>

</value>

</param>

<param>

<value>

<!-- client\_vsn -->

<string>xmlrpc:my-store-name:1.2.3</string>

</value>

</param>

<param>

<value>

<!-- pno -->

<int>4103219202</int>

</value>

</param>

<param>

<value>

<!-- merchant id (eid) -->

<string>0</string>

</value>

</param>

<param>

<value>

<!-- shared\_secret -->

<string>replace\_with\_digest</string>

</value>

</param>

<param>

<value>

<!-- pno\_encoding -->

<int>2</int>

</value>

</param>

<param>

<value>

<!-- type -->

<int>5</int>

</value>

</param>

<param>

<value>

<!-- client\_ip -->

<string>customer\_ip\_address</string>

</value>

</param>

</params>

</methodCall>

has\_account

To verify whether the consumer has an active part payment plan. The has\_account function works only in the Nordic countries. If the function answers true, then you should put part payment as the default payment option.

Digest

Consist of the values from eid, pno and secret (in this order) separated with a colon without the square brackets.

Example: base64encode(sha512("[eid]:[pno]:[shared\_secret]"))

Return value

String - true or false

Call structure

\* Required

| **Variable** | **Type** | **Description** |
| --- | --- | --- |
| proto\_vsn \* | string | Use 4.1. Automatically set in the library. |
| client\_vsn \* | string | xmlrpc:my-store-name:version-number. Automatically set in the library. |
| eid \* | int | A merchant ID which refers to your store in Klarna's database |
| pno \* | string | Consumer’s social security number. Use the following format:    SE: YYMMDD-NNNN, it can be sent with or without dash "-" or with or without the two first numbers in the year.  FI: DDMMYY-NNNN, it can be sent with or without dash “-”.  NO: DDMMYYNNNNN  DK: DDMMYYNNNN |
| digest \* | string | Digest for authentication. Klarna’s libraries do this for you together with the shared secret. |
| pno\_encoding \* | int | Indicates the person's country of origin    2: Sweden  3: Norway  4: Finland  5: Denmark |

Example

<?xml version="1.0" encoding="ISO-8859-1"?>

<methodCall>

<methodName>has\_account</methodName>

<params>

<param>

<value>

<!-- proto\_vsn -->

<string>4.1</string>

</value>

</param>

<param>

<value>

<!-- client\_vsn -->

<string>xmlrpc:my-store-name:1.2.3</string>

</value>

</param>

<param>

<value>

<!-- merchant id (eid) -->

<int>0</int>

</value>

</param>

<param>

<value>

<!-- pno -->

<string>4103219202</string>

</value>

</param>

<param>

<value>

<!-- shared\_secret -->

<string>replace\_with\_digest</string>

</value>

</param>

<param>

<value>

<!-- pno\_encoding -->

<int>2</int>

</value>

</param>

</params>

</methodCall>

get\_pclasses

To obtain pclass values from Klarna’s system. Pclasses are necessary to calculate monthly costs on the product and checkout pages.

Pclasses are also used in the pclass variable when doing a reserve\_amount call in order to create part payment and mobile payment orders.

Please note: get\_pclasses function is to be used to obtain pclass values for stores one time only, unless you get a new type of part payment assigned to your store. You are not permitted to use get\_pclasses function for continuous calculation of monthly costs or with every purchase in the checkout.

Depending on the configuration, the libraries will store the pclasses locally in your system. Make sure your webserver user can access the file or database for both reading and writing when making a get\_pclasses call.

The get\_pclasses should not be confused with the library method "getPClasses" which gets locally stored pclasses. The libraries refer to the get\_pclasses call as "fetch\_pclasses".

Digest

Consist of the values from eid, currency and secret (in this order) separated with a colon without the square brackets.

Example: base64encode(sha512("[eid]:[currency]:[shared\_secret]"))

Return value

An array of arrays containing the pclass values

0 = Pclass id number  
1 = Description  
2 = Amount of months for part payment  
3 = Start fee  
4 = Invoice fee  
5 = Interest rate  
6 = Minimum purchase amount for pclass  
7 = Country  
8 = Type (This is used to determine what type the pclass id is and which calculation method is to be used, see below for the different options.)  
9 = Pclass expiry date in the format YYYY-MM-DD. This is used for Buy now, pay in X months.

|  |  |
| --- | --- |
| **Type no** | **Description** |
| 0 | Part payment - fixed |
| 1 | Part payment - flexible |
| 2 | Buy now, pay in X month e.g. christmas campaign) |
| 3 | Fixed price |
| 4 | Pay in X months |
| 5 | Klarna mobile |

Call structure

\* Required

| **Variable** | **Type** | **Description** |
| --- | --- | --- |
| proto\_vsn \* | string | Use 4.1. Automatically set in the library. |
| client\_vsn \* | string | xmlrpc:my-store-name:version-number. Automatically set in the library. |
| eid \* | int | A merchant ID which refers to your store in Klarna's database |
| currency \* | int | Currency code for which you want to retrieve pclasses  0: Swedish krona  1: Norwegian krona  2: Euro  3: Danish krona  Please note!  Currency must match the country variable e.g. SEK and Sweden. |
| digest \* | string | Digest for authentication. Klarna’s libraries do this for you together with the shared secret. |
| country\* | int | Country code for which you want to retrieve pclasses  15: Austria  59: Denmark  73: Finland  81: Germany  154: Netherlands  164: Norway  209: Sweden    Please note!  Currency must match the country variable e.g. SEK and Sweden. |
| language \* | int | Language code in which language you want to retrieve pclasses    27: Danish  28: German  28: Austria  37: Finnish  97: Norwegian  101: Dutch  138: Swedish |

Example

<?xml version="1.0" encoding="ISO-8859-1"?>

<methodCall>

<methodName>get\_pclasses</methodName>

<params>

<param>

<value>

<!-- proto\_vsn -->

<string>4.1</string>

</value>

</param>

<param>

<value>

<!-- client\_vsn -->

<string>xmlrpc:my-store-name:1.2.3</string>

</value>

</param>

<param>

<value>

<!-- merchant id (eid) -->

<int>0</int>

</value>

</param>

<param>

<value>

<!-- currency -->

<int>0</int>

</value>

</param>

<param>

<value>

<!-- shared\_secret -->

<string>replace\_with\_digest</string>

</value>

</param>

<param>

<value><!-- country -->

<int>209</int>

</value>

</param>

<param>

<value>

<!-- language -->

<int>138</int>

</value>

</param>

</params>

</methodCall>

reserve\_amount

To create a reservation (order) in Klarna’s system. The reservation is valid, by default, for 7 days.

Digest

Consist of the values from eid, pno, amount and secret (in this order) separated with a colon without the square brackets.

Example: base64encode(sha512("[eid]:[pno]:[amount]:[shared\_secret]"))

Return value

Array - [rno, invoiceStatus]

The value of "rno" is the reservation number for the purchase, at most 255 char string.  
The value of "invoiceStatus" shows if the reservation can be delivered immediately or requires manual approval by Klarna. If you receive “Pending”, you can use check\_order\_status to query Klarna for a new status.

1 = OK  
2 = Pending

Call structure

\* Required

† Required in Austria, Germany and the Netherlands

| **Variable** | **Type** | **Description** |
| --- | --- | --- |
| proto\_vsn \* | string | Use 4.1. Automatically set in the library. |
| client\_vsn \* | string | xmlrpc:my-store-name:version-number. Automatically set in the library. |
| pno \* | string | Consumer’s social security number or birth date. Use the following format:   SE: YYMMDD-NNNN, it can be sent with or without dash "-" or with or without the two first numbers in the year.  FI: DDMMYY-NNNN, it can be sent with or without dash “-”.  NO: DDMMYYNNNNN  DK: DDMMYYNNNN  DE: DDMMYYYY  AT: DDMMYYYY  NL: DDMMYYYY |
| gender † | int | Gender of consumer    0: Female  1: Male |
| amount \* | int | Amount to be reserved, including VAT, given in cents, e.g. 10000 = 100 EUR. |
| reference | string | The reference person for the purchase if it is a company purchase. You can also use reference variable to write a message or other important information to the consumer on the invoice. Requires a setting which can be made by Klarna integration sales. |
| reference\_code | string | The reference code for the sale. You can also use reference\_code variable to write a message or other important information to the consumer on the invoice. Requires a setting which can be made by Klarna integration sales. |
| orderid1 | string | Order ID #1 |
| orderid2 | string | Order ID #2 |
| delivery\_address \* | array | Delivery address. See [address structure](https://developers.klarna.com/en/nl+php/kpm/checkout-api#reserve_amount_address) below for more information.  Please observe that for Sweden the address must be exactly as you receive with get\_addresses function. |
| billing\_address \* | array | Billing address. See [address structure](https://developers.klarna.com/en/nl+php/kpm/checkout-api#reserve_amount_address) below for more information. |
| client\_ip \* | string | IP number of the consumer when the purchase is made online |
| flags \* | int | Flag which affects the invoiced purchase. Input 0 to set no flag.  2: If you set this flag, a test reservation is created despite your store working in live mode. It comes in handy if you wish to test something while avoiding any disturbance to your regular invoicing.    “Klarna Mobil” flags  512: To tell Klarna that it's a phone transaction  1024: To send the consumer a PIN code. |
| currency \* | int | Currency code to be used for the invoices    0: Swedish krona  1: Norwegian krona  2: Euro  3: Danish krona    Please note: Currency, country and language must be the same as where the consumer is registered. E.g. Swedish consumer, SEK, Sweden and Swedish. |
| country \* | int | Code for the country where your sales will be:    15: Austria  59: Denmark  73: Finland  81: Germany  154: Netherlands  164: Norway  209: Sweden    Please note: Currency, country and language must be the same as where the consumer is registered. E.g. Swedish consumer, SEK, Sweden and Swedish. |
| language \* | int | Language code for the language used on the invoice.    27: Danish  28: German  28: Austria  37: Finnish  97: Norwegian  101: Dutch  138: Swedish    Please note: Currency, country and language must be the same as where the consumer is registered. E.g. Swedish consumer, SEK, Sweden and Swedish. |
| eid \* | int | A merchant ID which refers to your store in Klarna's database |
| digest \* | string | Digest for authentication. Klarna’s libraries do this for you together with the shared secret. |
| pno\_encoding \* | int | Indicates the country where the consumer is registered:    2: Sweden  3: Norway  4: Finland  5: Denmark  6: Germany  7: Netherlands  8: Austria    Please note: Currency, country and language must be the same as where the consumer is registered. E.g. Swedish consumer, SEK, Sweden and Swedish. |
| pclass \* | int | Pclass variable designates code to be used at the purchase and is utilized when the sale is a part payment or invoice purchase. If the sale is an invoice (14 days) purchase, pclass variable shall be set to -1.    -1 = invoice (14 days)  xxx = Part payment or invoice \*    *Part payments are not available for organizations.*    \*There are two ways to retrieve pclasses.    1. Go to Klarna Online and press "view store" button in the menu on the left. In the store view, click "Click here to view campaigns".  2. Use the function get\_pclasses to save all pclass values to your database. |
| goods\_list \* | array | See [goods list structure](https://developers.klarna.com/en/nl+php/kpm/checkout-api#reserve_amount_goodslist) below for more information |
| comment | string | Not available currently |
| shipment\_info \* | array | See [shipment info structure](https://developers.klarna.com/en/nl+php/kpm/checkout-api#reserve_amount_shipmentinfo) below for more information |
| travel\_info | array | Extra data from merchants selling travel tickets, talk to your Integration Sales representative for more information about travel info structure for your integration. |
| income\_expense | array | Not used at this moment |
| bank\_info | array | Not used at this moment |
| session\_id | array | Not used at this moment |
| extra\_info | array | Talk to your integration sales representative if you are asked to send in extra information to Klarna. |

address structure

\* Required  
† Required in Germany and Netherlands  
‡ Required in Netherlands  
❧ Required if cellno is not used  
☙ Required if telno is not used

| **Variable** | **Type** | **Description** |
| --- | --- | --- |
| fname \* | string | Consumer's first name |
| lname \* | string | Consumer's last name |
| careof | string | C/O address |
| company | string | Company name |
| street \* | string | Street address |
| house\_number † | string | House number. Used in Germany and Netherlands. For all other countries you can send in an empty string. |
| house\_extension ‡ | string | House extension. Only used in Netherlands, if the customer has one. For all other countries you can send in an empty string. |
| zip \* | string | Zip Code |
| city \* | string | City |
| country \* | integer | Code for the country where the consumer lives:    15: Austria  59: Denmark  73: Finland  81: Germany  154: Netherlands  164: Norway  209: Sweden |
| telno ❧ | string | Telephone number |
| cellno ☙ | string | Cellphone number |
| email \* | string | E-mail address |

Goods list structure

\* Required

| **Variable** | **Type** | **Description** |
| --- | --- | --- |
| qty \* | integer | Quantity of the articles |
| artno \* | string | Article number |
| title \* | string | Article title |
| price \* | integer | Article price, excluding VAT, given in cents, e.g. 10000 = 100 EUR.   *We always recommend to send in the price incl VAT by using the flag 32.* |
| vat \* | double | VAT in percent |
| discount | double | Discount in percent |
| flags | integer | 8: Indicates that the item is a shipment fee 16: Indicates that the item is a handling fee  32: Send the prices including VAT |

shipment\_info structure

\* Required

| **Variable** | **Type** | **Description** |
| --- | --- | --- |
| delay\_adjust \* | int | The two values determine how long after invoice activation Klarna starts countdown to the expiration date. Both countdowns are by default zero days. The time of the countdowns can be negotiated with Klarna.    1: Normal shipment  2: Express shipment |

Example

<?xml version="1.0" encoding="ISO-8859-1"?>

<methodCall>

<methodName>reserve\_amount</methodName>

<params>

<param>

<value>

<!-- proto\_vsn -->

<string>4.1</string>

</value>

</param>

<param>

<value>

<!-- client\_vsn -->

<string>xmlrpc:my-store-name:1.2.3</string>

</value>

</param>

<param>

<value>

<!-- pno -->

<string>4103219202</string>

</value>

</param>

<param>

<value>

<!-- gender -->

<!-- If left empty, send as string -->

<int>0</int>

</value>

</param>

<param>

<value>

<!-- amount -->

<int>10000</int>

</value>

</param>

<param>

<value>

<!-- reference -->

<string>Testperson-se Approved</string>

</value>

</param>

<param>

<value>

<!-- reference\_code -->

<string>Department 160</string>

</value>

</param>

<param>

<value>

<!-- orderid1 -->

<string>12345</string>

</value>

</param>

<param>

<value>

<!-- orderid2 -->

<string>6789</string>

</value>

</param>

<param>

<value>

<!-- delivery\_address -->

<struct>

<member>

<name>fname</name>

<value>

<string>Testperson-se</string>

</value>

</member>

<member>

<name>lname</name>

<value>

<string>Approved</string>

</value>

</member>

<member>

<name>careof</name>

<value>

<string>C/O last name</string>

</value>

</member>

<member>

<name>company</name>

<value>

<string>Test company</string>

</value>

</member>

<member>

<name>street</name>

<value>

<string>Stårgatan 1</string>

</value>

</member>

<member>

<name>house\_number</name>

<value>

<string></string>

</value>

</member>

<member>

<name>house\_extension</name>

<value>

<string></string>

</value>

</member>

<member>

<name>zip</name>

<value>

<string>12345</string>

</value>

</member>

<member>

<name>city</name>

<value>

<string>Ankeborg</string>

</value>

</member>

<member>

<name>country</name>

<value>

<int>209</int>

</value>

</member>

<member>

<name>telno</name>

<value>

<string></string>

</value>

</member>

<member>

<name>cellno</name>

<value>

<string>0765260000</string>

</value>

</member>

<member>

<name>email</name>

<value>

<string>testperson-se@example.com</string>

</value>

</member>

</struct>

</value>

</param>

<param>

<value>

<!-- billing\_address -->

<struct>

<member>

<name>fname</name>

<value>

<string>Testperson-se</string>

</value>

</member>

<member>

<name>lname</name>

<value>

<string>Approved</string>

</value>

</member>

<member>

<name>careof</name>

<value>

<string>C/O last name</string>

</value>

</member>

<member>

<name>company</name>

<value>

<string>Test company</string>

</value>

</member>

<member>

<name>street</name>

<value>

<string>Stårgatan 1</string>

</value>

</member>

<member>

<name>house\_number</name>

<value>

<string></string>

</value>

</member>

<member>

<name>house\_extension</name>

<value>

<string></string>

</value>

</member>

<member>

<name>zip</name>

<value>

<string>12345</string>

</value>

</member>

<member>

<name>city</name>

<value>

<string>Ankeborg</string>

</value>

</member>

<member>

<name>country</name>

<value>

<int>209</int>

</value>

</member>

<member>

<name>telno</name>

<value>

<string></string>

</value>

</member>

<member>

<name>cellno</name>

<value>

<string>0765260000</string>

</value>

</member>

<member>

<name>email</name>

<value>

<string>testperson-se@example.com</string>

</value>

</member>

</struct>

</value>

</param>

<param>

<value>

<!-- client\_ip -->

<string>customer\_ip\_address</string>

</value>

</param>

<param>

<value>

<!-- flags -->

<int>0</int>

</value>

</param>

<param>

<value>

<!-- currency -->

<int>0</int>

</value>

</param>

<param>

<value>

<!-- country -->

<int>209</int>

</value>

</param>

<param>

<value>

<!-- language -->

<int>138</int>

</value>

</param>

<param>

<value>

<!-- eid -->

<int>0</int>

</value>

</param>

<param>

<value>

<!-- shared\_secret -->

<string>replace\_with\_digest</string>

</value>

</param>

<param>

<value>

<!-- pno\_encoding -->

<int>2</int>

</value>

</param>

<param>

<value>

<!-- pclass -->

<int>-1</int>

</value>

</param>

<param>

<value>

<array>

<data>

<value>

<!-- goods\_list -->

<struct>

<member>

<name>goods</name>

<value>

<struct>

<member>

<name>title</name>

<value>

<string>Matrox G200 MMS</string>

</value>

</member>

<member>

<name>price</name>

<value>

<int>29999</int>

</value>

</member>

<member>

<name>discount</name>

<value>

<int>0</int>

</value>

</member>

<member>

<name>flags</name>

<value>

<int>32</int>

</value>

</member>

<member>

<name>artno</name>

<value>

<string>MG200MMS</string>

</value>

</member>

<member>

<name>vat</name>

<value>

<int>25</int>

</value>

</member>

</struct>

</value>

</member>

<member>

<name>qty</name>

<value>

<int>1</int>

</value>

</member>

</struct>

</value>

</data>

</array>

</value>

</param>

<param>

<value>

<!-- comment -->

<string></string>

</value>

</param>

<param>

<value>

<!-- shipment\_info -->

<struct>

<member>

<name>delay\_adjust</name>

<value>

<int>1</int>

</value>

</member>

</struct>

</value>

</param>

<param>

<value>

<!-- traveL\_info -->

<struct></struct>

</value>

</param>

<param>

<value>

<!-- income\_expense -->

<struct></struct>

</value>

</param>

<param>

<value>

<!-- bank\_info -->

<struct></struct>

</value>

</param>

<param>

<value>

<!-- session\_id -->

<struct></struct>

</value>

</param>

<param>

<value>

<!-- extra\_info -->

<struct></struct>

</value>

</param>

</params>

</methodCall>