iDEAL Basic

ING Wholesale Banking

for iDEAL Basic

Version 2.3, April 2010

WHOLESALE BANKING

ING

WWW.INGWHOLESALEBANKING.COM

Contents

1 Introduction

This document is intended for:

- acceptors (online shop owners) who want to integrate iDEAL into their online shops and
- developers who are responsible for the actual integration of iDEAL Basic into an online shop.

1.1 Overview

This document describes the following steps required for the integration process:

- Chapter 1, Introduction
- Chapter 2, Overview, provides an introductory overview of the integration including an introduction to the two possible approaches: product list or hashcode.
- Chapter 3, Product list, describes the integration steps if the so-called product list approach is chosen.
- Chapter 4, Hashcode, describes the integration steps if the so-called hashcode approach is chosen.
- Chapter 5, Back to the online shop, describes the settings and actions needed for processing transactions in your online shop after the consumer has made the payment.
- Chapter 6, Request transaction status, describes how to request the status of transactions performed, including the so-called 'obligation of proactivity'. This obligation of productivity means that the online shop owner is responsible for obtaining the status of a transaction.
- Chapter 7, Testing, describes the mandatory tests that have to be carried out before the online shop goes online.

NB: For general information about iDEAL, please refer to the document 'iDEAL General information'. For the integration of iDEAL Advanced, please see the integration manuals for the development platforms supported (Java, PHP, .NET).

1.2 Obligations

You are urgently requested to read this whole document together with the introductory document 'iDEAL General information' before integrating iDEAL Basic into an online shop. We request your special attention for the following **responsibilities of the acceptor**:

- **Security**: Every iDEAL acceptor is personally responsible for the secure design of his or her own online shop. The software supplied by ING is integrated on the basis of all usual security best practices. Improper integration may, however, nevertheless lead to an insecure online shop. This integration manual contains a number of practical instructions on this subject. These passages are generally indicated by the padlock symbol (🔒).

- **Obligation of proactivity**: Every iDEAL acceptor must comply with the so-called ‘obligation of proactivity’. This obligation means that you yourself are responsible for obtaining the status of a transaction before you make a delivery. For more information about the ‘obligation of proactivity’, see Chapter 6.
- **Presentation**: You will find the requirements on the presentation of iDEAL on your website at http://huisstijl.idealdesk.com. You will also find iDEAL logos and banners there.
- **Testing**: After the integration of iDEAL into your online shop, you are obliged to carry out a number of tests, which are described in Chapter 7 of this document.

1.3 Further questions?

You can contact the iDEAL service desk if you have any questions or comments. Our service desk can be reached between 9.00 am and 5.00 pm at ideal@ing.nl or on 020 65 225 80.

NB: The iDEAL Dashboard also has a FAQ menu option. You will find the answers to frequently asked questions here.

1.4 Definitions

The iDEAL system is based on bilateral relationships within the so-called ‘4-party’ model. The 4 parties involved in the model are as follows:

- The acceptor: the owner of the online shop
- The acquirer: the acceptor’s bank (ING)
- The consumer: the customer who wants to buy a product from the acceptor’s online shop
- The issuer: the consumer’s bank relation.

2 Overview of iDEAL Basic

The integration of iDEAL Basic into your online shop consists of a number of steps. This chapter gives a brief overview of these steps. Each of the steps is addressed in detail in one of the following chapters.

Step 1: Choose approach

Before you start integrating iDEAL Basic into your online shop, you have to choose between two possible approaches: product list or hashcode. The most important distinction between the two is that the product list approach is only suitable for handling orders of one (1) *type* of product per transaction, regardless of the number of *items*. If you want to process several types of product per transaction (regardless of the number of items per product type), choose the hashcode approach.

For example: *An online shop sells a variety of products, types A to Z. A consumer wants to order 3 of product A and 2 of product X. With the product list approach, this means two transactions: the first to process 3 of type A, the second to process 2 of product X. With hashcoding, you can process the entire order in one total transaction.*

You will find more information on the differences in the chapters dedicated to the approaches: Chapter 3 for product lists and Chapter 4 for hashcoding.

Step 2: Implement transactions

After choosing the approach, you have to add the functions needed by your website to launch transactions.

The core here for both approaches is an HTML Form Post with which your transaction details are sent to the iDEAL Acquiring Platform. An overview of all the data sent, and their permitted values, can be found in Appendix A of this document.

The implementation of the HTML Form Post code is almost independent of the chosen approach. First of all, there is one extra mandatory item of data with the hashcode approach, i.e. 'hash', which has as its value the hashcode applicable to that transaction, as described in Chapter 4. In addition, it is of course possible with the hashcode approach to include products of several types in the Form Post.

Examples of HTML Form Posts for iDEAL can be found in Appendix B (product list) and Appendix C (hashcoding via PHP).

NB: All amounts (`amount, itemPrice<n>`) should be given in *whole* euro*cents*. Therefore, for a product costing 1 euro you should enter "100" as the price. Decimals are not accepted.

NB: For every *type* of product within a transaction, the following data should be included in the HTML Post Form command:

```
itemNumber<n>
itemDescription<n>
itemQuantity<n>
itemPrice<n>
```

Here, '<n>' denotes the unique serial number of the product type, for example `itemNumber1` for the article number of the first product, `itemNumber2` for the second product etc.

Step 3: Process transactions

Every transaction launched needs further handling in your online shop after being processed by iDEAL. The steps required for this are described in Chapters 5 and 6. These chapters provide, among other things, further information on the 'obligation of proactivity', your obligation to obtain the status of transactions yourself.

Step 4: Test the integration

As soon as you have finished integrating iDEAL Basic in your online shop, you have to test the integration in the iDEAL test environment. For further information on this topic, see Chapter 7.

Step 5: Activate iDEAL Basic

Once the tests have been successfully completed and the contract has been signed and returned, you can activate iDEAL in your online shop. You have to do this yourself, using the menu option *Activeren* (*Activate*) in the iDEAL Dashboard production environment (https://ideal.secure-ing.com).

3 Product lists

The simplest form of iDEAL Basic uses a so-called product list. The restriction here is that only one *type* of article can be sold per transaction in the online shop. It is, however, permitted to perform a transaction of several *items* of one and the same product type.

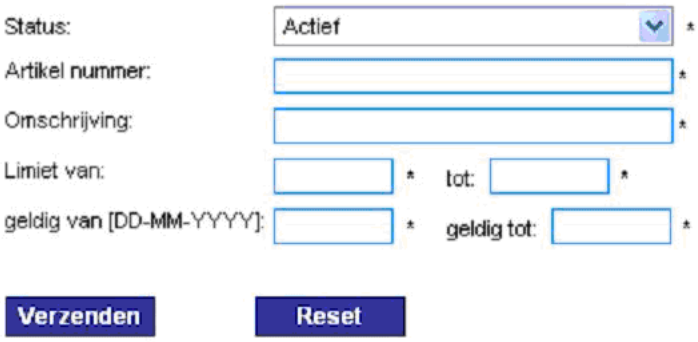
Example: *You can order 6 of article A in one transaction, but not 1 of article A and one of article B.*

If your online shop requires you to be able to bill several types of product at the same time, then use the hashcode approach as described in Chapter 4.

3.1 Define product list

To use a product list, you have to define it through the iDEAL Dashboard. That means that you have to enter all the products you want to sell in advance. For each transaction, the transaction details sent are compared with the product list submitted. Transactions involving products that are not on the list will result in error messages.

To define the product list, log into the iDEAL Dashboard (https://ideal.secure-ing.com) and choose the menu option *Produktlijst (Product list)*. The screen then looks as follows:



You have to enter a number of mandatory items of data via the screen for each product in your assortment:

- The article number. This is the identification of your article. You can determine this number yourself; however, it has to be unique for each product type.
- A description of the article, to be defined by the online shop owner.
- A minimum and maximum transaction amount. These values can be identical, which implies that the price cannot fluctuate. Permitted values are 0.10 to 99999999.

> **NB**: In contrast to most amounts in iDEAL, euros are used in this screen and not euro*cents*. Decimals are permitted.

NB: It is highly recommended to give realistic values here, particularly for the minimum price. Malicious persons who manage to gain access to your website may possibly misuse the minimum price you have entered. It is therefore recommended not to use a 'symbolic' minimum price, for example of 0.10 euros, for articles which are generally significantly more expensive.

- The period (date from, date until) in which the product will be sold.
- The status of the product, where the options are Actief (Active) or Inactief (Inactive).

NB: You have to define your product list separately for the iDEAL test and production environments.

3.2 Activate product list approach

Next, in order to activate the product list approach in your online shop, you have to carry out the following actions in the iDEAL Dashboard (https://ideal.secure-ing.com):

1. Log into the iDEAL Dashboard, and select the *Aanmeldproces (Registration process)* menu option. Next, select the *Configuratie (Configuration)* tab. If your iDEAL account is already active, choose the menu option *Profiel (Profile)* followed by the *Beveiliging (Security)* tab.
2. If a red cross appears beside the 'Certificaat upload geslaagd' ('Certificate upload successful') field, the product list approach has not yet been activated. Accept the generated key (it is changed every time the page is refreshed) and click on the Start Upload button.
3. A green check mark then appears at 'Certificaat upload geslaagd' ('Certificate upload successful'), which means that the upload has been successfully completed. Your webshop is now ready for the product list approach.

NB: You have to activate the product list approach separately for the iDEAL test and production environments.

3.3 Integrate the product list into your online shop

Once your product list has been defined through the iDEAL Dashboard, you can use it in your online shop. All you have to do is implement a limited quantity of code per transaction.

In view of the fact that you can only enter one (1) type of product per transaction with the product list approach, you will only use the following fields for the product:

```
itemNumber1
itemDescription1
itemQuantity1
itemPrice1                // Please note: price in WHOLE euroCENTS
```

Make sure that `itemNumber1` corresponds **exactly** with the article number you entered in your product list through the iDEAL Dashboard.

NB: You will find an example of HTML Form Post code for the product list approach in Appendix B.

4 Hashcoding

If your online shop has to be able to bill more than one type of product per transaction, you cannot use the product list approach (as described in the previous chapter), but will have to use the so-called hashcode approach. With hashcoding, an SHA1 hashcode is added to every transaction. The iDEAL Acquiring Platform uses this hashcode to validate the authenticity of the message.

4.1 Set the hashcode approach

To be able to use hashcoding in your online shop, you have to enter a so-called *hashkey* in advance in the iDEAL Dashboard (https://ideal.secure-ing.com). This is the key with which the hashcode is calculated per transaction.

To do this, log into the iDEAL Dashboard, and select the *Aanmeldproces (Registration process)* menu option. Next, select the *Configuratie (Configuration)* tab. If your iDEAL account is already active, choose the *Profiel (Profile)* menu option followed by the *Beveiliging (Security)* tab.

If a red cross appears beside the 'Certificaat upload geslaagd' ('Certificate upload successful') field, no hashkey has been defined. You then have two options for setting a key:

- Fill in a self-selected key then click on the Start Upload button.
- Accept the generated key (it is changed every time the page is refreshed) and click on the Start Upload button.

A green check mark then appears at 'Certificaat upload geslaagd' ('Certificate upload successful'), which means that your hashkey has been successfully saved.

Your hashkey is now entered in the 'Geheime sleutel' ('Secret key') field. You have to use this value in your code when creating the hashcode for a transaction. This is described in further detail in the following section.

NB: You have to define your hashkey separately for the iDEAL test and production environments.

NB: It is recommended to define different hash keys in the test environment and the production environment.

NB: Your hashkey is a secret key. Please ensure that unauthorized persons do not obtain access to it. In this context, also think about how to integrate the hashkey into your code.

4.2 Integrate hashcoding into your webshop

Once your hashkey has been defined through the iDEAL Dashboard, you can use hashcoding in your online shop. You need to create a hashcode for *every* transaction.

The hashcode is calculated by combining a number of fixed transaction data, including your own hashkey. An overview of the meaning of the different fields can be found in Appendix A.

The end result is the SHA1 hashcode of the transaction in question.

The step-by-step creation of the hashcode for a transaction looks as follows:

1. Combine your hashkey from the following transaction values (please note: it is essential for the variables to be in this order):

```
concatString =
   hashKey
   + merchantID
   + subID
   + amount                 // Please note: total amount in WHOLE euroCENTS
   + purchaseID
   + paymentType
   + validUntil
```

2. Add the following additional data to the step 1 results for the first type of product in the transaction (please note: it is essential for the variables to be in this order):

```
   + itemNumber1
   + itemDescription1
   + itemQuantity1
   + itemPrice1             // Please note: price in WHOLE euroCENTS
```

3. Repeat step 2 for every additional product type. If, for example, two more types of products are included in the transaction, you should combine the following values with the result of step 2:

```
   + itemNumber2
   + itemDescription2
   + itemQuantity2
   + itemPrice2             // Please note: price in WHOLE euroCENTS
   + itemNumber3
   + itemDescription3
   + itemQuantity3
   + itemPrice3             // Please note: price in WHOLE euroCENTS
```

4. The end result of step 3 then has to be stripped of the 'forbidden' characters `blank` (space), `\n` and `\t`. In Java, for example, the code may then look as follows:

```
concatString = value.replaceAll(" ", "");
```

```
concatString = value.replaceAll("\t", "");

concatString = value.replaceAll("\n", "");
```

In PHP, the code may look as follows:

```
$shastring = preg_replace(array("/[ \t\n]/"), array(''), $shastring);
```

5. In addition, `& > < and &quot` have to be replaced by `&, >, < and "` respectively. In Java, for example, the code may then look as follows:

```
concatString = value.replaceAll("&amp;", "&");

concatString = value.replaceAll("&gt;", ">");

concatString = value.replaceAll("&lt;", "<");

concatString = value.replaceAll("&quot;", "\"");
```

In PHP, this code may look as follows:

```
$shastring = preg_replace(

   array('/&amp;/i', '/&lt;/i', '/&gt;/i', '/&quot/i'),

   array(       '&',       '<',       '>',        '"'),

   $shastring};
```

6. The cleaned-up end result serves as the basis for calculating the SHA1 hashcode.
   In Java, for example, the code may then look as follows:

```
MessageDigest sha = MessageDigest.getInstance("SHA-1");

sha.update(concatString.getBytes());

byte[] hash = sha.digest();

BigInteger hashBI = new BigInteger(hash);
```

The same can be achieved in PHP with the following code:

```
$shasign = sha1($shastring);
```

7. The variables `BigInteger` (Java) and `$shasign` (PHP) then contain the hashcode that has to be added to the HTML Form Post as the value of the `hash` field. The relevant HTML code fragment may look as follows:

```
<input type="hidden" name="hash" value="$shasign">
```

NB: The final hashcode may also be a negative number, for example:

“-250994979195105729256580527516162610641984356011”.

NB: The hashcode may also be sent in hexadecimal format (without capital letters) in accordance with the standard SHA1 function in PHP, for example: "b6218aecdc3f2e9c1f2b9b8fd059a3f1633a4ebc".

NB: A detailed example of the code (in PHP) for the hashcode approach, including the associated HTML Form Post, is to be found in Appendix C.

5 Back to the online shop

Payments with iDEAL are dealt with through the iDEAL Acquiring Platform when iDEAL Basic is used. A consumer who clicks on the “Naar winkel” (“Back to shop”) button after a payment is automatically redirected to a previously determined static URL belonging to your online shop. The actual URL here is partially dependent on whether or not the transaction has gone through successfully. If something has gone wrong with the transaction, you have to display a mandatory error message in your webshop.

NB: Take into account that a consumer may not be automatically redirected to your online shop, for example because he or she closes the browser window early. See section 6.4 on this topic.

5.1 Set the redirection URLs

In order to automatically redirect the consumer to the right page in your online shop, you naturally have to set the page of your website that is to be used for this purpose. You therefore have to define the following static URLs in advance: [1]

- `UrlSuccess:` The consumer is automatically directed to this URL after a successful payment;
- `UrlCancel:` The consumer is automatically directed to this URL after the transaction has been cancelled;
- `UrlError:` The consumer is directed to this URL if an error has occurred.

You can define these URLs in two ways:

- Through the iDEAL Dashboard, *Aanmeldproces (Registration process)* menu option, *Configuratie (Configuration)* tab. If your iDEAL account is already active: *Profiel (Profile)* menu option, *Beveiliging (Security)* tab.
- In your online shop’s code, sent as part of the HTML Form Post. In PHP, this could, for example, look as follows:

```
// in the PHP part of the code
$baseurl = 'http://www.uwwebwinkel.nl';
$urlSuccess = "$baseurl/Success.html";
$urlCancel = "$baseurl/Cancel.html";
$urlError = "$baseurl/Error.html";
<…>
```

[1] The `URLService` no longer exists.

```
// in the HTML part of the code

<input type="hidden" name="urlSuccess" value="$urlSuccess">

<input type="hidden" name="urlCancel" value="$urlCancel">

<input type="hidden" name="urlError" value="$urlError">
```

NB: URLs in the code prevail over URLs entered in iDEAL.

5.2 Mandatory error messages

If an error occurs when the iDEAL Acquiring Platform is dealing with the transaction, the consumer (by selecting the “Naar winkel” (“Back to shop”) button) is automatically redirected to the previously defined error page of your webshop (set as `UrlError`).

On this page, you must display the following mandatory error message (in Dutch or English):

“Er is een fout opgetreden bij het verwerken van uw iDEAL transactie. Neem contact op met de webwinkel of probeer het later nogmaals.

Controleer of uw betaling is afgeschreven alvorens de betaling opnieuw uit te voeren.”

“An error occurred while processing your iDEAL transaction. Please contact the online shop or try again later.

Please check your account if the payment has been processed before you pay again with iDEAL.”

6 Request transaction status

Every iDEAL transaction passes through different statuses. It is the acceptor's responsibility to request the current status of a transaction. This is called the *obligation of proactivity*. You will find more information on the possible statuses in the document 'iDEAL General information'.

6.1 Requesting the status through iDEAL Dashboard

With iDEAL Basic, you can only request the status through the iDEAL Dashboard. You have to log in and select the *Betalingen (Payments)* menu option. Enter the desired search criteria and press the Zoeken (Search) button.

NB: If your online shop requires you to be able to get the status on an *automated* basis within the online shop itself, you should use iDEAL Advanced.

6.2 Automatic notification

iDEAL Basic offers two further options for receiving *notification* when an iDEAL transaction has been *completed*. This notification is optional and can be set through the iDEAL Dashboard.

To do this, log into the iDEAL Dashboard, select the *Aanmeldproces (Registration process)* menu option and then the *Configuratie (Configuration)* tab. If your iDEAL account is already active, you should choose the *Profiel (Profile)* menu option and the *Beveiliging (Security)* tab.

Now select the desired method (XML, e-mail, or none) in the 'Notificatie' ('Notification') field.

The two options are:

- E-mail: The acceptor receives an e-mail after every completed transaction. This e-mail is sent to the address entered as the commercial contact's e-mail address (to be found in the iDEAL Dashboard).
- XML: The acceptor receives an XML message after every completed transaction. This message is sent to an URL determined by the acceptor him- or herself. This URL has to be set in the 'Notificatie URL' ('Notification URL') field (also on the *Configuratie (Configuration)* tab or under *Beveiliging (Security)*.

NB: The authenticity of the information the acceptor receives by e-mail or XML cannot be guaranteed by ING. Only the status obtained through the iDEAL Dashboard is guaranteed. For this reason, you are recommended to check the status of your transactions regularly (for example, daily) through the iDEAL Dashboard.

6.3 Successful transactions

ING advise you only to make delivery if the transaction is successful. A successful transaction has the status “003 – Betaling bevestigd door bank” (003 – Payment confirmed by bank”) in the iDEAL Dashboard.

NB: If you use the reconciliation option, a successful transaction passes through two additional statuses after the status 003 (succes (success)): 007 (gereconcilieerd (reconciled)) and 009 (uitbetaald (paid)). These statuses also indicate a successful transaction. More information on reconciliation can be found in the document ‘iDEAL General information’ and through the iDEAL Dashboard.

6.4 Outstanding transactions

You must take into account that the consumer is *not* always automatically redirected to the online shop after a transaction, for example if the consumer closes the browser window early. In such cases, the transaction does not reach the final status, and the status remains at “001”. Because such transactions are not completed, you will not receive automatic notification.

Such transactions can reach the final status in two different ways:

- Log into the iDEAL Dashboard and select the menu option *Betalingen (Payments)*. Enter the status ‘001’ as the search criterion. Then select the ‘Details’ option for each transaction displayed and then the ‘Statusverzoek’ (‘Status request’) button in the details screen.
- iDEAL automatically processes all outstanding transactions during the following night, and the status is set to the final status. If this is done, you will automatically receive an e-mail about it to the commercial contact’s e-mail address.

7 Testing

This chapter describes all the mandatory tests that have to be carried out before an online shop with iDEAL integration can go online. In addition, the remaining functions of your online shop should of course also be tested. However, that goes beyond the scope of this document.

7.1 Mandatory tests

The mandatory tests are described functionally in the Chapter '*Registration process*' of the document 'iDEAL General information'. There are seven tests, which should all be carried out in the iDEAL **test** environment (https://ideal**test**.secure-ing.com).

The tests are run as follows:

1. The acceptor logs into the iDEAL test environment using the user name and password obtained in the registration process.
2. The acceptor sends 7 test orders to the URL of the iDEAL test environment (https://idealtest.secure-ing.com/ideal/mpiPayInitIng.do). The test environment returns the following **pre-programmed** results:

Order	**Expected result**
Transaction with `amount` = 100:	`Success`
Transaction with `amount` = 200:	`Cancelled`
Transaction with `amount` = 300:	`Expired`
Transaction with `amount` = 400:	`Open`
Transaction with `amount` = 500:	`Failure`
Transaction with `amount` = 700:	`SO1000 Failure in system`
Directory Request	`Issuer Simulator`

NB: If you are using the product list approach, you only have to create one test product of € 1.00. You then have to 'buy' 1, 2, 3, 4, 5 and seven items to carry out the first six mandatory tests.

3. The acceptor checks the results obtained. This can be done through the *Aanmeldproces (Registration process)* menu option, *Status* tab, and only in the **test** environment.

NB: Test results are sent automatically to iDEAL for verification several times a day. Positive verification of the test results is needed to activate iDEAL in the production environment. Activation must be carried out by the acceptor, via the iDEAL Dashboard. This can be done from the next day onwards.

APPENDIX A: Data catalogue

The core of a transaction with iDEAL Basic is an HTML Form Post to the iDEAL Acquiring Platform. An overview of all the data sent, and their permitted values, can be found in this Appendix.

Parameter	Format	Description
merchantID	PN..9	Your AcceptorID is provided in the registration process.
subID	N..max6	Provided in the registration process, value is normally "0" (zero)
purchaseID	AN..max16	The online shop's unique order number (determined by the acceptor)
amount	N..max12	Total amount of the transaction in **whole** euro**cents**
currency	AN3	Fixed value = EUR
language	AN..2	Fixed value = nl (at present only Dutch is supported)
description	AN..max32	Description of the order (determined by the acceptor)
itemNumber<n>	AN..max12	Article number. <n> is 1 for the first product, 2 for the second, etc. **N.B.** Note that for every product type the parameters itemNumber<n>, itemDescription<n>, itemQuantity<n> and itemPrice<n> are mandatory.
itemDescription<n>	AN..max32	Description of article <n>
itemQuantity<n>	N..4	Number of items of article <n> that the consumer wants to buy
itemPrice<n>	N..max12	Price of article <n> in **whole** euro**cents**
paymentType	AN..max10	Fixed value = ideal
validUntil	yyyy-MM-ddTHH:mm:ss.SSSZ	Time at which the transaction expires (maximum of 1 hour later). The consumer has time until then to pay with iDEAL. The Z stands for the time zone (CET).
hash	N..max50	SHA1 hashcode, used only with the hashcode approach (Chapter 4).
urlSuccess	AN..max512	Static URLs belonging to the webshop to which the consumer is automatically redirected after a transaction is carried out. See Chapter 5 for further information.
urlCancel	AN..max512	
urlError	AN..max512	
urlService	AN..max512	No longer exists

Format	Explanation
AN	Alphanumerical, free text
ANS	Strictly alphanumerical (letters and numbers only)
N	Numerical
PN	Numerical (padded), content is filled up to maximum length with preceding zeros
..23	Maximum number of positions for alphanumerical and numerical values

Appendix B: Example code for product list (HTML)

This appendix contains an example of the possible HTML Form Post code for initiating a transaction using the product list approach (see Chapter 3 for details).

The example relates to a fictional shop with `merchantID` 0050xxxx. The owner of this shop has defined a product list through the iDEAL Dashboard which includes (among others) a product with the article number 'DC554711'. That article number recurs in the code as the value of the field `itemNumber1`.

A description of the fields used can be found in Appendix A. For a further explanation of the URLs defined, see section 5.1.

```
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   </head>
   <body>
      <form method="post" action=https://idealtest.secure-
      ing.com/ideal/mpiPayInitIng.do id="form1" name="form1">
      <input type="hidden" name="merchantID" value="0050xxxxx">
      <input type="hidden" name="subID" value="0">
      <input type="hidden" name="amount" value="100" >
      <input type="hidden" name="purchaseID" value="0511181">
      <input type="hidden" name="language" value="nl">
      <input type="hidden" name="currency" value="EUR">
      <input type="hidden" name="description" value="Produktlijst">
      <input type="hidden" name="itemNumber1" value="DC554711">
      <input type="hidden" name="itemDescription1" value="Design Chair Billy">
      <input type="hidden" name="itemQuantity1" value="1">
      <input type="hidden" name="itemPrice1" value="25000">
      <input type="hidden" name="paymentType" value="ideal">
      <input type="hidden" name="validUntil" value="2009-12-01T12:00:00:000Z">
      <input type="hidden" name="urlCancel"
      value="http://www.uwwebwinkel.nl/Cancel.htm">
```

```
        <input type="hidden" name="urlSuccess"
        value="http://www.uwwebwinkel.nl/Success.htm">
        <input type="hidden" name="urlError" value="http://www.uwwebwinkel.nl/Error.htm">
        <input type="submit" name="submit2" value="Verstuur" id="submit2">
        </form>
    </body>
</html>
```

Appendix C: Example code for hashcoding (PHP)

This appendix contains an example of the possible code for initiating a transaction using the hashcode approach (see Chapter 4 for details).

The example relates to a fictional shop with `merchantID` 0050xxxx.

A description of the fields used can be found in Appendix A. For a further explanation of the URLs defined, see section 5.1.

```
<html>
  <head>
     <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  </head>
  <body>
     <?php
     $key = '41e3hHbYhmxxxxxx'; // your hashkey or secret key
     $merchantID = '0050xxxxx'; //your acceptorID
     $subID = '0';              //virtually always the value 0 (zero)
     $purchaseID = '10';        //free field - to be used for your back office
     $paymentType = 'ideal';    //always ideal

     # A piece of code for the automatic calculation of a Valid Until is given below
     # (time now plus 15 minutes)
     # NB: an explanation of php functions is provided at http://nl.php.net/
     # (http://nl.php.net/manual/nl/function.date.php)
     $validUntil = date('Y-m-d\TH:i:s.000\Z', time()+900);
     # $validUntil = "2009-01-01T12:34:56:000Z"; # example result date(..)
     # please note: maximum of 1 hour (3600 sec) later
     $itemNumber1 = '1';                    // article number
     $itemDescription1 = 'omschrijving';    // the description
     $itemQuantity1 = 1;                    // number of items
     $itemPrice1 = 100;                     // Unit price of article in whole eurocents
     $amount = $itemQuantity1 * $itemPrice1; // total amount of transaction
```

```
# Composition of the string to be hashed
$shastring = "$key$merchantID$subID$amount$purchaseID$paymentType$validUntil"
                . "$itemNumber1$itemDescription1$itemQuantity1$itemPrice1";
# Replacement of 'forbidden characters'
$shastring = preg_replace(
       array("/[ \t\n]/", '/&amp;/i', '/&lt;/i', '/&gt;/i', '/&quot/i'),
       array(         '',        '&',       '<',       '>',        '"'),
       $shastring);

# SHA1 calculation with the php formula sha1
$shasign = sha1($shastring);

# Other variables (not in hash)
$language = 'nl'; # preferably '' for constant characters, quicker to process.
$currency = 'EUR';
$description = 'Example hashcode';
$baseurl = 'http://www.uwwebwinkel.nl';
$urlSuccess = "$baseurl/Success.html";
$urlCancel = "$baseurl/Cancel.html";
$urlError = "$baseurl/Error.html";

# Build the actual web page:
# The 'Versturen' ('Send') button containing the variables to be sent follows
below.
echo <<<EOT
<form method="post" action="https://idealtest.secure-
ing.com/ideal/mpiPayInitIng.do" name="form1">
   <!-- Don't forget, after carrying out the tests, to change the url in the ACTION
   to the production environment -->
   <input type="hidden" name="merchantID" value="$merchantID">
   <!-- example with POST variables:
   <input type="hidden" name="merchantID" value="{$_POST['merchandID']}"> Please
   note: Always verify POST/GET or clean up before use. Extra html code can be
   injected here uncontrolled. (e.g. javascript does this auto-submit)-->
   <input type="hidden" name="subID" value="$subID">
```

```
        <input type="hidden" name="amount" value="$amount">
        <input type="hidden" name="purchaseID" value="$purchaseID">
        <input type="hidden" name="language" value="$language">
        <input type="hidden" name="currency" value="$currency">
        <input type="hidden" name="description" value="$description">
        <input type="hidden" name="hash" value="$shasign">
        <input type="hidden" name="paymentType" value="$paymentType">
        <input type="hidden" name="validUntil" value="$validUntil">
        <input type="hidden" name="itemNumber1" value="$itemNumber1">
        <input type="hidden" name="itemDescription1" value="$itemDescription1">
        <input type="hidden" name="itemQuantity1" value="$itemQuantity1">
        <input type="hidden" name="itemPrice1" value="$itemPrice1">
        <!-- and possibly more articles $itemXXX<n> - more variables (array)-->
        <input type="hidden" name="urlSuccess" value="$urlSuccess">
        <input type="hidden" name="urlCancel" value="$urlCancel">
        <input type="hidden" name="urlError" value="$urlError">
        <input type="submit" name="submit2" value="Verstuur">
    </form>
    EOT;
    ?>
  </body>
</html>
```