

Omnichcheckout Magento Module

[Compatibility and extendability](#)

[Installation](#)

[Package contents](#)

[The checkout page](#)

[1 - Shopping cart](#)

[2 - Shipping method](#)

[3 - Coupon code](#)

[4 - Iframe element](#)

[Settings](#)

[General settings](#)

[API settings](#)

[Callback settings](#)

[Htaccess protection](#)

[Callback registration](#)

[Events](#)

[Callback events](#)

[Unfreeze](#)

[Booked](#)

[Finalization](#)

[Automatic fraud control](#)

[Payment session events](#)

[Init payment session](#)

[Update payment session](#)

[Get payment session](#)

[Delete payment session](#)

[Callbacks](#)

[API connection](#)

[Payment session API](#)

[Payment handling API](#)

Introduction

This module replaces Magento's native checkout module (**Mage_Checkout**) by redirecting incoming calls to the module **Resursbank_Omnichcheckout**.

We are careful not to modify any of Magento's core functionality and make use of as many native classes and templates as possible.

The inline documentation explains the use of every function and attempts to outline their purpose as well as possible. If you should require additional information regarding any function or process, please contact our support staff.

Compatibility and extendability

The module is compatible with Magento 1.7.x - 1.9.x and supports all native Magento checkout features, except for gift options (messages, wrapping etc.).

The module contains a single rewrite (**`Mage_Adminhtml_Block_Sales_Order_View_Info`**), which helps us include payment information from Resursbank in the order view within the administration panel.

There are no known extension conflicts, and since we have been so vigilant in avoiding rewrites there are unlikely to be any direct conflicts at all.

The module takes full advantage of Magento's event system, extending it with custom events (explained in detail later in this document) to help developers integrate their modules and custom features with our checkout extension.

Approximately half of all information is submitted through an iframe which implements a custom event system to notify the client side of updates applied to its fields (explained in greater details later in this document).

Installation

To install the module please follow the step-by-step approach listed below.

- 1) Extract the compressed package containing the module.
- 2) Add the files from the extracted package to your Magento directory (following the paths included in the package, for example **app/code/community/Resursbank** should be moved to the directory **app/code/community/** on your installation). Below this set of instructions you will find a complete file tree explaining in detail what files should be placed where.
- 3) Once all files have been moved into your installation directory, proceed to the administration panel and navigate to **System -> Cache Management** in the top menu.
- 4) Select all available caches (by marking the checkbox available to the far right on each line in the list of cache alternatives).
- 5) Select the option “Refresh” in the **Actions** list, available in the top right-hand corner of the list itself (to the left of the **Submit** button).
- 6) Click on the **Submit** button to refresh the cache and allow the module to be installed.
- 7) To refresh administration permissions please log out from, and then back into, the administration panel.

Magento Admin Panel Global Record Search Logged in as admin | onsdag 18 maj 2016 | [Log Out](#)

Dashboard Sales Catalog Customers Promotions Newsletter CMS Reports **System** [Get help for this page](#)

Cache Storage Management

[Flush Magento Cache](#) [Flush Cache Storage](#)

Select All | Unselect All | Select Visible | Unselect Visible | 8 items selected Actions Refresh [Submit](#)

Cache Type	Description	Associated Tags	Status
<input checked="" type="checkbox"/> Configuration	System(config.xml, local.xml) and modules configuration files(config.xml).	CONFIG	ENABLED
<input checked="" type="checkbox"/> Layouts	Layout building instructions.	LAYOUT_GENERAL_CACHE_TAG	ENABLED
<input checked="" type="checkbox"/> Blocks HTML output	Page blocks HTML.	BLOCK_HTML	ENABLED
<input checked="" type="checkbox"/> Translations	Translation files.	TRANSLATE	ENABLED
<input checked="" type="checkbox"/> Collections Data	Collection data files.	COLLECTION_DATA	ENABLED
<input checked="" type="checkbox"/> EAV types and attributes	Entity types declaration cache.	EAV	ENABLED
<input checked="" type="checkbox"/> Web Services Configuration	Web Services definition files (api.xml).	CONFIG_API	ENABLED
<input checked="" type="checkbox"/> Web Services Configuration	Web Services definition files (api2.xml).	CONFIG_API2	ENABLED

Additional Cache Management

[Flush Catalog Images Cache](#) Pregenerated product images files.

[Flush Swatch Images Cache](#) Pregenerated configurable swatches image files.

[Flush JavaScript/CSS Cache](#) Themes JavaScript and CSS files combined to one file.

Package contents

The tree below will explain exactly what files should be placed where during installation. It also specifies what files you may or may not need depending on your installation.

As explained in the installation instructions above, the contents of the package follows Magento's directory structure to simplify locating files and ease installation.

[app/code/community/Resursbank](#)

Place this in [app/code/community/](#)

[app/design/adminhtml/default/default/layout/resursbank](#)

Place this in [app/design/adminhtml/default/default/layout/](#)

[app/design/adminhtml/default/default/template/resursbank](#)

Place this in [app/design/adminhtml/default/default/template/](#)

[app/design/frontend/base/default/layout/resursbank](#)

Place this in [app/design/frontend/base/default/layout/](#)

If you should experience any issues please move this directory to your active theme folder instead of the above suggested **base/default**.

[app/design/frontend/base/default/template/resursbank](#)

Place this in [app/design/frontend/base/default/template/](#)

If you should experience any issues please move this directory to your active theme folder instead of the above suggested **base/default**.

[app/etc/modules/Resursbank_Omnichcheckout.xml](#)

Place this in [app/etc/modules/](#)

[app/locale/en_US/Resursbank_Omnichcheckout.csv](#)

Place this in [app/locale/en_US/](#)

More languages will be added over time and you can choose yourself which to include.

en_US refers to [English](#), **sv_SE** to [Swedish](#) and so on.

[js/resursbank](#)

Place this in [js/](#)

[lib/Resursbank](#)

Place this in [lib/](#)

[skin/adminhtml/default/default/resursbank](#)

Place this in [skin/adminhtml/default/default/](#)

[skin/frontend/base/default/resursbank](#)

Place this in [skin/frontend/base/default/](#) if you are running Magento CE 1.8.x or older, otherwise ignore this directory.

If you should experience any issues please move this directory to your active theme folder instead of the above suggested **base/default**.

[skin/frontend/rwd/default/resursbank](#)



Place this in [skin/frontend/rwd/default/](#) if you are running Magento CE 1.9.x, otherwise ignore this directory.

If you should experience any issues please move this directory to your active theme folder instead of the above suggested **rwd/default**.

The checkout page

Contrary to Magento's native checkout module Omnich checkout embraces Ajax calls for almost everything. Applying changes to cart rows, coupon codes, changing shipping method and all information entered within the iframe element are all transmitted and stored on the quote object within the client's checkout session. The checkout page consists of four sections as illustrated by the image below.

SHOPPING CART

PRODUCT	PRICE	QTY	SUBTOTAL
 RETRO CHIC EYEGLASSES <i>SKU: ace002</i>	295,00 US\$	<input type="text" value="1"/> Edit	295,00 US\$
 JACKIE O ROUND SUNGLASSES <i>SKU: ace001</i>	225,00 US\$	<input type="text" value="1"/> Edit	225,00 US\$

[EMPTY CART](#) [UPDATE SHOPPING CART](#) -OR- [CONTINUE SHOPPING](#)

SHIPPING AND HANDLING

- Free Shipping**
- ☐ Free 0,00 US\$
- Flat Rate**
- ☐ Fixed 0,00 US\$

DISCOUNT CODES

Godkänn villkor

Vi jobbar med Resurs Bank för en säker och trygg köppplevelse. För att handla behöver du godkänna deras användarvillkor. Detta steg görs endast första gången du använder Resurs Banks Checkout. [Läs villkoren här.](#)

☒ Jag godkänner villkoren och accepterar användning av cookies

1. Ange personnummer

ååmmdd-xxxx

Hämta adress

Fyll i adressuppgifter manuellt

2. Välj betelsätt

- ☒ Resurskort
- ☐ Faktura - Faktura
- ☐ Kort - Visa/Mastercard
- ☐ Nytt kort
- ☐ Delbetalning
- ☐ Kort (test) - Visa/Mastercard

3. Slutför

☐ Spara mina uppgifter för framtida köp
När du sparar dina uppgifter blir det enklare att handla nästa gång du använder Resurs Banks Checkout. Informationen sparas i en cookie i din webbläsare. [Läs mer om hur vi skyddar dina personuppgifter.](#)

Summa: 562.9 kr

Genomför köp

Genom att klicka på "Genomför köp" accepterar jag

1 - Shopping cart

The shopping cart block is the original one from Magento's core without any modifications. Using JavaScript we observe changes made to an item row (eg. changing the quantity or removing the item) and relay this information to our controller through Ajax calls.

<app/code/community/Resursbank/Omnichcheckout/controllers/CartController.php>

`updateAction()`

Update item quantity.

`deleteAction()`

Delete item.

2 - Shipping method

The shipping method block is the original one from Magento's core without any modifications. Using JavaScript we observe changes applied (eg. switching between the various alternatives) and relay this information to our controller through an Ajax call.

<app/code/community/Resursbank/Omnichcheckout/controllers/IndexController.php>

`saveShippingMethodAction()`

Store selected shipping method on quote object.

3 - Coupon code

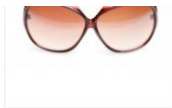
The coupon code block is the original one from Magento's core without any modifications. Using JavaScript we observe changes applied to the coupon code field and relay this information to our controller through an Ajax call.

<app/code/community/Resursbank/Omnichcheckout/controllers/CartController.php>

`couponAction()`

Apply coupon code.

The applied coupon code and its discount value will be displayed beneath the coupon code field.



[EMPTY CART](#)

[UPDATE SHOPPING CART](#) -OR- [CONTINUE SHOPPING](#)

SHIPPING AND HANDLING

Free Shipping

☐ Free 0,00 US\$

Flat Rate

☐ Fixed 0,00 US\$

DISCOUNT CODES

asd1234

Discount code (asd1234)

-40,00 US\$

Godkänn villkor

Vi jobbar med Resurs Bank för en säker och trygg köppplevelse. För att handla behöver du godkänna deras användarvillkor. Detta steg görs endast första gången du använder deras

4 - Iframe element

The iframe element is produced by Resursbank's API upon initializing a payment session. This is where the customers will enter their billing and shipping address, select their desired method of payment and complete the purchase.

All of this information is transmitted to Magento on-the-fly using **postMessage**:s in JavaScript, and then stored on the quote object through Ajax calls.

<app/code/community/Resursbank/Omnichcheckout/controllers/IndexController.php>

saveBillingAction()

Store billing address on quote object.

saveShippingAction()

Store shipping address on quote object.

savePaymentAction()

Store selected payment method on quote object (please note the function `_correctPaymentMethod()`). This function converts the payment method name submitted to the controller as the input values in the iframe are inconsistent (there is a select-box used for mobile view, and radio buttons used for desktop view)).

saveShippingMethodAction()

Store selected shipping method on quote object.

saveOrderAction()

Convert quote object to order object.

The module does not store SSN (social security numbers) in Magento.

When the customer clicks on the **complete purchase** button a **postMessage** is sent to Magento, which subsequently executes an Ajax request to convert the quote object into an order object, before sending a **postMessage** back to the iframe, allowing it to proceed to the payment gateway.

Settings

To reach the modules configuration first navigate to **System -> Configuration** in the top menu, and then proceed by clicking on the tab labeled **Omnichcheckout**, beneath the heading **Resursbank** at the very bottom of the left-hand menu.

The screenshot displays the Magento Admin Panel interface. At the top, the header includes the Magento logo, 'Admin Panel', a 'Global Record Search' bar, and user information: 'Logged in as admin | onsdag 18 maj 2016 | Log Out'. Below the header is a navigation bar with tabs: Dashboard, Sales, Catalog, Customers, Promotions, Newsletter, CMS, Reports, and System (highlighted in orange). A 'Get help for this page' link is also present.

On the left side, there is a sidebar menu. The 'Configuration' section is expanded, showing sub-menus for GENERAL, CATALOG, and CUSTOMERS. The 'GENERAL' sub-menu is further expanded, listing various configuration areas like General, Web, Design, Currency Setup, Store Email Addresses, Contacts, Reports, and Content Management.

The main content area is titled 'Omnichcheckout' and features a 'Save Config' button in the top right corner. It is divided into three sections:

- General Settings:** Contains a single setting 'Enabled' set to 'Yes' with a '[STORE VIEW]' label.
- API Settings:** Contains several settings: 'Debugger Enabled' (Yes), 'Username' (mageparts), 'Password' (masked with dots), 'Test Mode' (Yes), and 'Weight Unit' (kg). Each setting has a '[STORE VIEW]' label.
- Callback Settings:** Contains settings for 'Debugger Enabled' (No), 'Basic Auth Username', 'Basic Auth Password', and 'Callback Registration'. The 'Callback Registration' section includes an 'Update callbacks' button and two callback URLs: '[UNFREEZE]' and '[AUTOMATIC_FRAUD_CONTROL]', each with a '[STORE VIEW]' label.

Each of the three sections available here are described below.

General settings

General Settings

Enabled [STORE VIEW]

Enabled	Enable/disable the extension.
----------------	-------------------------------

API settings

API Settings

Debugger Enabled

[STORE VIEW]

Username

[STORE VIEW]

Password

[STORE VIEW]

Test Mode

[STORE VIEW]

Weight Unit

[STORE VIEW]

Debugger enabled	Whether or not to enable the debugger (this will store various calls and information in the file var/log/resurs_api.log to help you debug issues with the API connection).
Username	API username.
Password	API password.
Test mode	Whether or not we are in test mode.
Weight unit	Unit of weight for order items.

Callback settings

Callback Settings

Debugger Enabled	<input type="text" value="No"/>	[STORE VIEW]
Basic Auth Username	<input type="text"/>	[STORE VIEW]
Basic Auth Password	<input type="password"/>	[STORE VIEW]
Callback Registration	<div><div>Update callbacks</div><div><div>[UNFREEZE]</div><div>http://resursbank.dev/index.php/omnicheckout/callback/unfreeze/</div><div>-----</div><div>[AUTOMATIC_FRAUD_CONTROL]</div><div>http://resursbank.dev/index.php/omnicheckout/callback/automatic_fraud_control/</div><div>-----</div><div>[ANNULMENT]</div><div>http://resursbank.dev/index.php/omnicheckout/callback/annulment/</div><div>-----</div><div>[FINALIZATION]</div><div>http://resursbank.dev/index.php/omnicheckout/callback/finalization/</div><div>-----</div><div>[BOOKED]</div><div>http://resursbank.dev/index.php/omnicheckout/callback/booked/</div><div>-----</div><div>[TEST]</div><div>http://resursbank.dev/index.php/omnicheckout/callback/test/</div><div>-----</div></div></div> <td>[STORE VIEW]</td>	[STORE VIEW]

Debugger enabled	Whether or not to enable the debugger (this will store various calls and information in the file var/log/resurs_callback.log to help you debug issues relating to callbacks).
Basic auth username	Htaccess protection username.
Basic auth password	Htaccess protection password.

Htaccess protection

You can use a custom htpasswd file to protect the URL `[protocol]://[domain]/omnicheckout/callback` (eg. <http://mydomain.com/omnicheckout/callback>) with a username and password of your choice.

You can then relay this to Resursbank through the **basic auth username** and **password** settings described above.

These settings are both optional but can be used to provide an additional layer of protection by ensuring that only authorized calls can be made to the callback controller.

It should be noted that callbacks relating to orders are already secured using a token (SHA1 encoded string based on; a random string consisting of 64 characters, the quote id from the customer's checkout session and a timestamp).

Callback registration

Callback registration will occur automatically when you install the module for the first time. You can update your callbacks at any time by clicking the **update callbacks** button as illustrated by the image above.

The **update callbacks** button will initiate the callback registration process, which is the exact same process as the one executed when installing the module.

Callback registration generates callback URLs based on the URL of your website. If you should ever change your website's URL you must also update your callbacks, otherwise callbacks won't function.

This process cannot be invoked automatically as it can cause faulty callback registration in development, stage and testing environments since the URL may differ from the production environment.

Below the **update callbacks** button is a list of all the currently registered callback URLs. This list is fetched every time you visit this configuration section and is therefore always up to date with the information stored at Resursbank.

Events

We have included a variety of custom events which you can utilise using Magento's native event system.

Callback events

These events will be dispatched when callbacks are triggered by Resursbank.

Unfreeze

omniconcheckout_callback_unfreeze_before

Dispatched when a payment has been unfrozen at Resursbank, but the order object in Magento has not been handled yet.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omniconcheckout/controllers/CallbackController.php :: unfreezeAction\(\)](#) when the **UNFREEZE** callback is dispatched by Resursbank.

omniconcheckout_callback_unfreeze_after

Dispatched when a payment has been unfrozen at Resursbank and the order object in Magento has been handled.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omniconcheckout/controllers/CallbackController.php :: unfreezeAction\(\)](#) when the **UNFREEZE** callback is dispatched by Resursbank.

Booked

omniconcheckout_callback_booked_before

Dispatched when a payment has been booked at Resursbank, but the order object in Magento has not been handled yet.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omniconcheckout/controllers/CallbackController.php :: bookedAction\(\)](#) when the **BOOKED** callback is dispatched by Resursbank.

omniconcheckout_callback_booked_after

Dispatched when a payment has been booked at Resursbank and the order object in Magento has been handled.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omniconcheckout/controllers/CallbackController.php :: bookedAction\(\)](#) when the **BOOKED** callback is dispatched by Resursbank.

Finalization

omnicheckout_callback_finalization_before

Dispatched when a payment has been finalized at Resursbank, but the order object in Magento has not been handled yet.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omnichcheckout/controllers/CallbackController.php :: finalizationAction\(\)](#) when the **FINALIZATION** callback is dispatched by Resursbank.

omnicheckout_callback_finalization_after

Dispatched when a payment has been finalized at Resursbank and the order object in Magento has been handled.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omnichcheckout/controllers/CallbackController.php :: finalizationAction\(\)](#) when the **FINALIZATION** callback is dispatched by Resursbank.

Automatic fraud control

omnicheckout_callback_automatic_fraud_control_before

Dispatched when a payment has passed automatic fraud controls at Resursbank, but the order object in Magento has not been handled yet.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omnichcheckout/controllers/CallbackController.php :: automaticFraudControlAction\(\)](#) when the [AUTOMATIC_FRAUD_CONTROL](#) callback is dispatched by Resursbank.

omnicheckout_callback_automatic_fraud_control_after

Dispatched when a payment has passed automatic fraud controls at Resursbank and the order object in Magento has been handled.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omnichcheckout/controllers/CallbackController.php :: automaticFraudControlAction\(\)](#) when the [AUTOMATIC_FRAUD_CONTROL](#) callback is dispatched by Resursbank.

Annulment

omnicheckout_callback_annulment_before

Dispatched when a payment has been fully annulled by Resursbank, but the order object in Magento has not been handled yet.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omnicheckout/controllers/CallbackController.php :: annulmentAction\(\)](#) when the **ANNULMENT** callback is dispatched by Resursbank.

omnicheckout_callback_annulment_after

Dispatched when a payment has been fully annulled by Resursbank and the order object in Magento has been handled.

Parameters

- **order** ([Mage_Sales_Model_Order](#) instance of targetted order).
- **parameters** ([array](#) of all parameters included in the callback request from Resursbank).

Executed in

[app/code/community/Resursbank/Omnicheckout/controllers/CallbackController.php :: annulmentAction\(\)](#) when the **ANNULMENT** callback is dispatched by Resursbank.

Payment session events

These events will be dispatched while handling a payment session in Magento. Payment sessions are the means through which Magento and Resursbank communicates.

When a client enters the website and places an item into their shopping cart a **payment session** will be created at Resursbank (through a call made by Magento). The session will be given a unique ID, which is returned to Magento and kept within the customer's checkout session (**Mage_Checkout_Model_Session**).

When creating the payment session we also create a **token** which we store on the customer's quote object, and later on the resulting order object (assuming the purchase is completed).

A **token** is a SHA1 encoded string based on; a random string consisting of 64 characters, the current quote id (collected from the customer's checkout session) and a timestamp.

For the most part session handling (such as updating the items included in a session, deleting a session and retrieving information about completed sessions (meaning the order has been placed and the session has closed)) will utilise the **token** rather than the unique ID created by Resursbank when initializing the payment session.

It's worth noting both exist, but when communicating with the API we use the **token** to identify which order we mean to handle.

Init payment session

omnicheckout_api_init_session_before

Dispatched when a payment session is about to be initiated (eg. a client entered the website and placed an item into their shopping cart).

Parameters

- **data** ([array](#) of data which will be submitted to Resursbank).
- **quote** ([Mage_Sales_Model_Quote](#) instance referring to the quote being handled).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: initPaymentSession\(\)](#)

omnicheckout_api_init_session_after

Dispatched when a payment session has been initiated (eg. a client entered the website and placed an item into their shopping cart).

Parameters

- **data** ([array](#) of data which will be submitted to Resursbank).
- **response** ([JSON encoded stdClass](#) containing unique **payment session id** and the **iframe element** which will be rendered on the checkout page).
- **quote** ([Mage_Sales_Model_Quote](#) instance referring to the quote being handled).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: initPaymentSession\(\)](#)

Update payment session

omnicheckout_api_update_session_before

Dispatched when a payment session is about to be updated (eg. the client changed something which affects the total value and/or contents of their shopping cart).

Parameters

- **data** ([array](#) containing all items included in the quote, selected shipping method (if any) and applied coupon code (if any) formatted as expected by Resursbank).
- **quote** ([Mage_Sales_Model_Quote](#) instance referring to the quote being handled).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: updatePaymentSession\(\)](#)

omnicheckout_api_update_session_after

Dispatched when a payment session has been updated (eg. the client changed something which affects the total value and/or contents of their shopping cart).

Parameters

- **data** ([array](#) containing all items included in the quote, selected shipping method (if any) and applied coupon code (if any) formatted as expected by Resursbank).
- **response** ([string](#) returned from Resursbank (empty)).
- **quote** ([Mage_Sales_Model_Quote](#) instance referring to the quote being handled).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: updatePaymentSession\(\)](#)

Get payment session

omnicheckout_api_get_session_before

Dispatched when a completed payment session is about to be fetched from Resursbank.

Parameters

- **quote_token** ([string token](#) to identify quote/order object).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: getPayment\(\)](#)

omnicheckout_api_get_session_after

Dispatched when a completed payment session has been fetched from Resursbank.

Parameters

- **response** ([JSON encoded stdClass](#) containing payment information).
- **quote_token** ([string token](#) to identify quote/order object).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: getPayment\(\)](#)

Delete payment session

omnicheckout_api_delete_session_before

Dispatched when a completed payment session is about to be deleted.

Parameters

- **quote** ([Mage_Sales_Model_Quote](#) instance referring to the quote being handled).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: deletePaymentSession\(\)](#)

omnicheckout_api_delete_session_after

Dispatched when a payment session has been deleted.

Parameters

- **response** ([JSON encoded value](#)).
- **quote** ([Mage_Sales_Model_Quote](#) instance referring to the quote being handled).

Executed in [app/code/community/Resursbank/Omnicheckout/Model/Api.php :: getPayment\(\)](#)

Callbacks

Omnichcheckout will receive callbacks from Resursbank to update order information when certain events take place:

1. Unfreeze

A payment becomes unfrozen and is debitable (meaning that Resursbank approves the payment, and thus the order can be shipped by the merchant).

When this callback is invoked we automatically create an invoice and trigger the callback events `omnichcheckout_callback_unfreeze_before` and `omnichcheckout_callback_unfreeze_after`

2. Booked

A payment becomes booked.

When this callback is invoked we trigger the callback events `omnichcheckout_callback_booked_before` and `omnichcheckout_callback_booked_after`

3. Finalization

A payment becomes finalized.

When this callback is invoked we trigger the callback events `omnichcheckout_callback_finalization_before` and `omnichcheckout_callback_finalization_after`

4. Automatic fraud control

A payment passes automatic fraud checks performed by Resursbank.

When this callback is invoked we trigger the callback events `omnichcheckout_callback_automatic_fraud_control_before` and `omnichcheckout_callback_automatic_fraud_control_after`

5. Annulment

A payment becomes completely annulled.

When this callback is invoked we trigger the callback events `omnicheckout_callback_annulment_before` and `omnicheckout_callback_annulment_after`

For a more indepth description of all events triggered through callbacks, please visit the [callback events](#) chapter.

All callback URLs will point to the controller <app/code/community/Resursbank/Omnicheckout/controllers/CallbackController.php>

Callback URLs are registered (collected and sent to Resursbank) automatically during installation of the module, and can later be manually updated from the modules configuration section ([Callback registration](#)).

Please keep in mind that if you ever change the URL of your website you must update your callback URLs or callbacks won't work.

For more information regarding callbacks please visit the [callback settings](#) chapter.

API connection

There are two separate APIs provided by Resursbank.

Payment session API

The first is made specifically for the Omnichcheckout solution (keep in mind this is a module for this solution, not the other way around) and enables you to relay and retrieve payment information to Resursbank.

You relay cart content, selected shipping method, applied coupon code and other properties which affects the quote value to the API, and you can later retrieve information about payments which have been completed (for example to display payment information on the order view in the administration panel).

All functionality used when communicating with this API can be found in the file <app/code/community/Resursbank/Omnichcheckout/Model/Api.php>

Several events are triggered when various actions are executed against the API, you can find a complete list in the [payment session events](#) chapter.

Payment handling API

The second API is used to handle payments directly through Magento's administration panel. This means you will never need to access Resursbank's administration panel directly to handle payments.

All functionality relating to this API implementation can be found in the file <app/code/community/Resursbank/Omnichcheckout/Model/Order/Api.php>

The implementation is currently not completed and this chapter will be updated with more information as the API model is developed.