

Beecomm API

version 2.1.2

Table of Content

Environment.....3

Auth.....4

Access List.....5

Order Centre6

Environment

Base Url

<https://api.beecommcloud.com/v1/>

Token lifetime is 24 hours.

Fields marked with **(M)** are mandatory

All date fields should be in format 'YYYY-MM-DD HH:MM:ss'

Auth

To start communication with our API you'll need to acquire an `access_token` using `client_id` and `client_secret` (provided by Beecomm).

Url: `{{API_BASE_URL}}/auth/token`

Body format: `application/x-www-form-urlencoded`

Method: POST

Request:

```
let options = {
  headers: {
    'Content-Type': 'application/x-www-form-urlencoded'
  },
  form: {
    client_id: string (provided_by_Beecomm), (M)
    client_secret: string (provided_by_Beecomm) (M)
  }
}
```

Result:

```
{
  requestId: string,
  result: boolean,
  message: string,
  access_token: _token_
}
```

STATUS	MEANING	RESULT
200	Authenticated successfully	TRUE
401	Unauthorized	FALSE

Access List

Returns a list of user's access objects with restaurantId and branchId, you should use the identifiers you obtained from this end-point for any restaurantId and branchId reference in this document.

Url: `{{API_BASE_URL}}/ext/getAccessList`

Headers: { Authorization: `_token_` }

Method: GET

Result:

```
{
  result: true,
  request_id: string,
  access_list: [
    {
      restaurantId: string,
      restaurantName: string,
      branches: [
        {
          id: string,
          name: string
        }
      ]
    }
  ]
}
```

STATUS	MEANING
...	See Common Error Codes for more

Order Centre

Base Url
{{API Base Url}}/order-center

Table of Content

Push Order7

Get Menu Revision.....10

Get Menu11

Delivery Order Model14

Menu Model.....17

Common Error Codes.....19

Push Order

Send delivery or takeaway order to a specific branch.

Url: {{order_center_base_url}}/pushOrder

Headers: { Authorization: Bearer + _token_ , Content-Type: application/json}

Method: POST

Request:

```
{
  restaurantId: string,
  branchId: string,
  menuRevision: string,
  orderInfo: {
    orderType: number,
    comments: string,
    discountAmount: number,
    internalOrderId: string,
    dinnersCount: number,
    purchaseTime: date,
    orderTotal: number,
    dinners:[{
      firstName: string,
      lastName: string,
      phoneNumber: string,
      emailAddress: string,
      items: [{
        dishId: string,
        itemName: string,
        isCombo: boolean,
        quantity: number,
        dishPrice: number,
        totalPrice: number,
        preparationComments: string,
        remarks: [{
          remarkId: string,
          remarkDescription: string,
          operator: string
        }],
      },
      toppings: [{
        toppingId: string,
        toppingName: string,
        additionalCost: number,
        quantity: number,
        operator: string
      }],
    },
    subItems: [{
      dishId: string,
      itemName: string,
      quantity: number,
      additionalCost: number,
      totalPrice: number,
      preparationComments: string,
      remarks: [{
        remarkId: string,
        remarkDescription: string,
        operator: string
      }],
      toppings: [{
        toppingId: string,
        toppingName: string,
        additionalCost: number,
        quantity: number
      }
    ]
  }
}
```

-continue-

```

payments: [{
  charged: boolean,
  paymentType: number,
  paymentSum: number,
  tip: number,
  cardInfo: {
    approvalNumber: string,
    cardNumber: string,
    cardExpirationDate: string,
    cardHolderName: string,
    cvv: number
  }
}],
deliveryInfo: {
  firstName: string,
  lastName: string,
  cityName: string,
  streetName: string,
  homeNumber: string,
  addressGeoLocation_lat: string,
  addressGeoLocation_lng: string,
  formattedAddress: string,
  eta: number,
  disposableCutlery: boolean,
  entrance:string,
  entranceCode: string,
  doorName: string,
  companyName: string,
  deliveryLocationType: string,
  leaveOutside: boolean,
  residenceType: string,
  apartmentNumber: string,
  floor: string,
  deliveryCost: number,
  phoneNumber: string,
  cellular: string,
  remarks: string,
  disposableCutlery: boolean
}
}
}

```

Result:

```

{
  request_id: string,
  result: boolean,
  status: number,
  orderNumber: number,
  message: string
}

```


STATUS	MESSAGE	RESULT
1	Success	TRUE
2	Item {{id}} ({{item_name}}) has unsupported remark id {{id}} ({{remark_description}})	FALSE
2	Item {{id}} ({{item_name}}) has unsupported topping id {{id}} ({{topping_name}})	FALSE
2	Item {{id}} ({{item_name}}) has unsupported subItem id {{id}} ({{subItem_name}})	FALSE
2	Item {{id}} ({{item_name}}) does not exist in delivery menu	FALSE
3	This branch does not have delivery menu.	FALSE
14	Failed to parse date string	FALSE
22	You are not using the latest menu revision. You should /getMenu	FALSE
23	Incompatible discountAmount value	
24	Insufficient card information	
25	Payment clearance error IMPORTANT! order accepted by the requested branch but will not be prepared for delivery until payment will be settled fully.	TRUE
26	A payment was flagged as externally charged but one of the following fields is missing: approvalNumber, cardNumber and cardExpirationDate in case of a credit card.	FALSE
27	Payments total does not cover order balance	FALSE
28 1001	Please check your order, missing mandatory field(s). A description of the missing field will be included in the response.	FALSE
29	The target POS is closed. Please check open hours.	
30	Order total is different from the sum of the items	FALSE
31	We couldn't acquire proper address from supplied delivery info	
32	Cash payment is not allowed when leaveOutside is set true	
33	You cannot send more than one order with identical internalOrderId in the same day.	
...	See Common Error Codes for more	

Get Menu Revision

Get identifier of the current delivery menu.

if you hold a menu revision different from the response go to [/getMenu](#).

Url: `{{order_center_base_url}}/getMenuRevision`

Headers: { Authorization: `_token_` , Content-Type: `application/json`}

Method: POST

Request:

```
{
  restaurantId: string, (M)
  branchId: string, (M)
}
```

Result:

```
{
  request_id: string,
  result: boolean,
  status: number,
  menuRevision: string,
  lastUpdate: string,
  message: string
}
```

STATUS	MESSAGE	RESULT
1	Success	TRUE
20	Requested branch does not exists.	FALSE
21	This branch does not have any delivery menu.	FALSE
...	See Common Error Codes for more	

Get Menu

Get the latest delivery menu of a specific branch.

Url: {{order_center_base_url}}/getMenu

Headers: { Authorization: _token_ , Content-Type: application/json}

Method: POST

Request:

```
{
  restaurantId: string, (M)
  branchId: string, (M)
}
```

Result:

```
{
  requestId: string,
  result: boolean,
  status: number,
  message: string,
  lastUpdate: date,
  menuRevision: string,
  deliveryMenu: {
    categories: [{
      name:string,
      subCategories: [{
        name: string,
        dishes: [{
          dishId: string,
          price: number,
          name: string,
          description: string,
          barcode: string,
          prepareTime: number,
          tags: string[],
          saleUnit: string,
          unitMeasure: string,
          unitQuantity: number,
          imagePath: string,
          isCombo: boolean,
          toppingGroups: [{
            name: string,
            minQuantity: number,
            maxQuantity: number,
            toppings:[{
              toppingId: string,
              name: string,
              automaticSelection: boolean,
              additionalCost: number
            }]
          }]
        }]
      }]
    }
  ]
}
```

```
-continue-  
  
    remarkGroups: [{  
        name: string,  
        minQuantity: number,  
        maxQuantity: number,  
        remarks:[{  
            remarkId: string,  
            remarkDescription: string  
        }]  
    }],  
    comboSubs: [{  
        name: string,  
        minQuantity: number,  
        maxQuantity: number,  
        defaultDishId: string,  
        dishes: rootLevelDishModel[]  
    }]  
}  
}
```

NOTE: If a topping flagged with `automaticSelection=true` then you should pre select it for the customer automatically.

BEECOMM API

STATUS	MESSAGE	RESULT
1	Success	TRUE
20	Requested branch does not exists.	FALSE
21	This branch does not have any delivery menu.	FALSE
...	See Common Error Codes for more	

Delivery Order Model

FIELD	MEANING	MANDATORY
Root		
restaurantId	Restaurant identifier from /AccessList	Y
branchId	Branch identifier from /AccessList	Y
menuRevision	Menu revision identifier from /getMenu	Y
orderInfo		Y
orderType	2=takeaway 3=delivery	Y
comments	Order comment	N
discountAmount	Order discount.	Y
internalOrderId	Outer company order identifier	Y
dinnersCount	Number of dinners	Y
purchaseTime	Timestamp of order placement (YYYY-MM-DD HH:MM:ss)	Y
orderTotal	Items total + delivery cost - any discount)without tip)	
dinners		Y
firstName	First name of the dinner	Y
lastName	Last name of the dinner	Y
phoneNumber	Phone number of the dinner	N
emailAddress	Email address of the dinner	N
Items		N
dishId	Dish identifier from /getMenu	Y
itemName	Name of the item from /getMenu	Y
isCombo	If item is a combo meal. subItems will be processed only if this field is set to true .	Y
quantity	Quantity	Y
dishPrice	Net price of the dish - excluding any extras	Y
totalPrice	"Line total price" including item, item's toppings, sub items and any extra charge it has, multiplied by the quantity	Y
preparationComments	Comments for this Item	N
toppings		N

FIELD	MEANING	MANDATORY
toppingId	Topping identifier from /getMenu You can suffix this field with: _WO for with out _OS for On the side _L for a lot _S for some (little of..)	Y
toppingName	Topping name from /getMenu	Y
additionalCost	Additional cost for this topping	Y
quantity	Topping's quantity	Y
operator	Will be printed with the topping, options are: 'some', 'lot', 'with', 'without', 'on-the-side'	N
remarks		N
remarkId	Remark identifier from /getMenu	Y
remarkDescription	Remark description from /getMenu	Y
operator	Will be printed with the remark, options are: 'some', 'lot', 'with', 'without',	N
subItems		N
dishId	Dish identifier from /getMenu	Y
itemName	Name of the sub item from /getMenu	Y
quantity	Sub item quantity	Y
additionalCost	Additional cost on top of the root item price	Y
totalPrice	Sum of subItem and all additional cost from subItem toppings multiplied by subItem quantity	Y
remarks	See root item remark model	N
toppings	See root item topping model	N
preparationComments	Comments for this subItem	N
payments		Y
paymentType	options are: 1=Credit Card, 2=Cash, 12=10Bis, 27=Goodi, 10=Cibus	Y
Charged	Flag the payment as externally charged. If true, then the following fields are mandatory - <i>"cardInfo.approvalNumber"</i> , <i>"cardInfo.cardNumber"</i> and <i>"cardInfo.cardExpirationDate"</i> in case of a credit card.	N
paymentSum	Total amount for this payment(without tip)	Y
paymentConfirmation	If this payment has confirmation number from any authority, place it here.	N
tip	Per payment amount of tip (NOT percent)	N

FIELD	MEANING	MANDATORY
cardInfo		N
approvalNumber	If payment charged externally this field is mandatory	N
cardNumber	number on the card, if payment charged externally please supply Masked value.	Y
cardExpirationDate	Expiration date of the card, format: 'mmyy'	Y
cardHolderName	Card holder name	N
cvv	3-4 digits from the back of the card	Y
deliveryInfo		Y
firstName	First name of the client	Y
lastName	Last name of the client	Y
cityName	name of the city	Y
streetName	Name of the street	Y
homeNumber	Home number	Y
entrance	Residence entrance	N
entranceCode	Entrance code	N
floor	Floor number	N
apartmentNumber	Apartment number	N
doorName	Name on the door	N
companyName	Company name if the client is a company	N
cellular	Cellular phone number	N
phoneNumber	Main phone number for that order	Y
disposableCutlery	Include disposable cutlery (True/False)	N
leaveOutside	Leave order outside the home and let the client know it's there. With this option selected Cash payments will not be accepted	N
deliveryCost	Delivery cost	Y
formattedAddress	streetName + homeNumber+ "," + cityName.	N/?
eta	Minutes to supply time from purchase time ("mm")	N
remarks	Remarks for the delivery	N
addressGeoLocation_lat	Address location latitude	N
addressGeoLocation_lng	Address location longitude	N
deliveryLocationType	Type of delivery domain eg. 'street', 'army base' etc..	N
residenceType	Type of residence eg. 'building', 'private house' etc..	N

Menu Model

FIELD	MEANING
Root	
menuRevision	Revision id of the delivery menu
lastUpdated	Last modification date
deliveryMenu	
categories	
Name	Name of the category
subCategories	
Name	Name of the sub category
dishes	
dishId	identifier of the dish
price	Net price of the dish
name	Name of the dish
description	Description of the dish
barcode	Barcode of the dish
prepareTime	Time it takes to prepare this dish
tags	List of strings attached to the dish.
saleUnit	The unit used to sale this item, can be: 'unit', 'weight'
unitMeasure	The type used to measure this item, can be: 'unit', 'kg', 'gr', 'ltr'
unitQuantity	the quantity used for a single item
imagePath	Url for the item image
isCombo	Determine if the item is a combo meal or not, if true comboSubs will be populated
toppingGroups	
name	Name of the topping group
minQuantity	Minimum quantity from this topping; 0=no minimum
maxQuantity	Maximum quantity from this topping; 0=no maximum
toppings	
toppingId	Identifier of the topping
name	Name of the topping

BEECOMM API

FIELD	MEANING
additionalCost	Extra charge this topping carry
automaticSelection	Is default topping and can be pre-selected for the client
remarkGroups	
name	name of the remark group
minQuantity	Minimum quantity from this remark; 0=no minimum
maxQuantity	Maximum quantity from this remark; 0=no maximum
remarks	
remarkId	Identifier of the remark
remarkDescription	Name of the remark
comboSubs	
name	Name of the combo sub category
minQuantity	Minimum quantity from this combo sub category; 0=no minimum
maxQuantity	Maximum quantity from this combo sub category; 0=no maximum
defaultDishId	Identifier of default dish for this combo sub category if exist.
builtIn	True means that all ComboSub dishes are builtIn the root dish
dishes	

<<rootLevelDishModel>>

Common Error Codes

STATUS	MESSAGE	RESULT
0 or 1000	not implemented code: {{error_code}}	FALSE
1001	Missing mandatory field(s)	FALSE
1002	User is not authorised to access this resource with explicit deny	FALSE
1003	Unsupported action	FALSE
1005	Service is not configured for requested branch	FALSE
1006	internal communication error	FALSE
1007	Communication with the requested client is currently closed	FALSE