Описание протокола взаимодействия с APIv2 мини-доставки

ВНИМАНИЕ Вторая версия API поддержана в DeliveryHatch не ниже **v3.0.5.959**
Общий смысл запросов и ответов остался неизменным, однако их содержимое переработано с целью обеспечить
более качественную работу внешних систем за счет снижения влияния человеческого фактора.

APIv2 мини-доставки позволяет:

- Получить актуальное состояние меню из ресторана втч с учетом текущих ограничений блюд и стоп-листа на кассе.
- Предварительно проверить сформированную во внешней системе корзину гостя на предмет ошибок на стороне rk7.
- Передать заказ гостя на кассу rk7 посредством мини-доставки.
  Поддержана передача оплаты и персональной скидки для гостя.
- Проверить текущее состояние ранее сформированного заказа.

Общие положения

API доставки реализовано в рамках HTTP протокола.

Каждый запрос к API должен иметь заголовок Token, значение которого выдается владельцем доставки. Токен передается без префикса:

*Token: *token**

Базовый URL для запросов: **http://IP:PORT/api/v2/ext/** Этот URL можно использовать для проверки работоспособности.

Если набрать в любом браузере: http://192.168.1.10:11011/api/v2/ext/ - при работоспособном настроенном API мы должны получить ответ

```
result  false
msg "\"Token\" header expected."
```

В данном примере 11011 - порт DeliveryHatch, 192.168.1.10 - IP компьютера с установленным сервером мини-доставки. Аналогично стоит проверять доступность сервиса извне после проброса порта.

Важно Не передавайте «нейтральные» значения для необязательных полей в запросах, если нет вещественных данных для этих полей. Необязательные поля могут быть безболезненно исключены из запроса.

Список доступных функций

/ping /getmenu /validate /postorder /expressorders

Проверка соединения

Для проверки работоспособности цепочки связей ПО, необходимого для формирования заказа, используется запрос **GET /ping**

Запрос не ожидает никаких входных параметров.

Ответ на запрос содержит JSON объект, представленный полем **result: boolean** - индикатор успешного выполнения запроса, полем **msg: string**, в котором будет передан текст ошибки в случае, если **result = false**, иначе вложенным объектом data, содержащим поля **DHVersion** - Текущая версия модуля мини-доставки, **CSVersion** - текущая версия кассового сервера r-keeper и поле **CSNetName** - сетевое имя кассового сервера, с которым работает модуль мини-доставки.

Начиная с DHVersion: «3.0.6.1222» в ответе будет передаваться объект «params», содержащий локальные для DeliveryHatch данные для интеграции с сервисом dadata, а так же режим работы ресторана (/domains/delivery/const/const.py)

Пример успешного ответа:

```
{
  "result": true,
  "data": {
    "DHVersion": "3.0.6.1222",
    "CSVersion": "7.6.0.81",
    "CSNetName": "CONF_MIDSERVER",
    "params": {
      "dadata_apikey": "52b77b6f5ce32e0ab2af8d30eafdb2cc18961cef",
      "dadata_basecity": "Новосибирск",
      "dlvfrom": "9,00",
      "dlvto": "20,00",
      "dlvperiod": "15"
    }
  }
}
```

params.dadata_apikey - Ключ для доступа к сервису dadata
params.dadata_basecity - Базовый город для подстановки адресов
params.dlvfrom - Доставка начинает работать «С» ЧЧ,ММ
params.dlvto - Доставка работает «ДО» ЧЧ,ММ
params.dlvperiod - Шаг времени в минутах, на которое можно оформить заказ

ВНИМАНИЕ!!! Если в ответ на данный запрос вы получили **result = false**, то все дальнейшие

попытки воспользоваться API доставки бессмысленны в виду наличия проблем на стороне ресторана.
Однако и положительный ответ на данный запрос не гарантирует отсутствия ошибок в дальнейшем, т.к. проверяет только отсутствие проблем на физическом уровне. Логические ошибки должны быть так же обработаны внешней системой.

Получение меню

Для получения списка блюд, доступных к заказу необходимо отправить **GET запрос /getmenu**

В ответ вернется JSON объект, содержащий поля result:boolean, msg:string или data:array

result - указывает на существование ошибки при обработке запроса. (result = false - запрос не выполнен)
msg - текстовое сообщение. В случае ошибки (result = false) будет передан текст ошибки
data - один, или несколько JSON объектов, описывающих блюда, доступные к продаже

Пример объекта, описывающего блюдо:

```
{
  "code": 55,
  "name": "Булочка с корицей",
  "categpath": "Кухня\\Выпечка",
  "categcode" : 88,
  "recipe": "Сдобная булочка в посыпке из сахарной пудры и корицы.",
  "price": 6500,
  "ishit":0,
  "rests": 5,
  "modifiers": [
    {
      "schemeid": 1001058,
      "groupid": 1001060,
      "groupname": "Тестовая группа 2",
      "uplimit": 2,
      "downlimit": 1,
      "freecount": 0,
      "changesprice": true,
      "items": [
        {
          "ident": 1001063,
          "code": 31,
          "name": "Модификатор 2 1",
          "limit": 1,
          "price": 1000
        },
        {
          "ident": 1001064,
          "code": 39,
          "name": "Модификатор 2 2",
```

```
          "limit": 1,
          "price": 0
        }
      ]
    }
  ]
}
```

code - Код блюда в справочнике системы. Должен быть передан при создании заказа.
name - Название блюда для отображения.
categpath - Расположение блюда в дереве меню в справочниках r-keeper. Служит для разделения блюд по категориям
categcode - Код категории меню из r_keeper, к которому пренадлежит блюдо.
recipe - Значение поля «Рецепт» для блюда в справочниках rk-keeper. Может содержать данные БЖУ, рецепт или простое описание блюда, которое будет доступно в интерфейсе доставки.
price - Стоимость одной порции блюда в копейках.
ishit - Флаг - является ли блюдо популярным. (Для отображения в отдельной категории - «ХИТ»)
rests - Информация об остатках блюда в ресторане.
Если **rests > 0**, то добавить в заказ можно не более **rests** порций блюда.
Если **rests = -1**, то ограничений на количество доступных порций со стороны ресторана не передано.
Если блюдо по какой либо причине не может быть продано на кассе ресторана, то оно попросту не попадет в ответ на запрос.

modifiers - Список групп модификаторов, доступных для блюда:
modifiers[n]:
schemeid - Идентификатор схемы модификаторов
groupid - Идентификатор группы модификаторов
groupname - Название группы модификаторов
uplimit - Максимальное кол-во модификаторов, которое можно выбрать из этой группы (-1 - нет ограничений)
downlimit - Минимальное кол-во модификаторов, которое можно выбрать из этой группы (-1 - нет ограничений)
changesprice - Изменяет ли модификатор цену. Если false, то стоимость модификатора не нужно прибавлять к стоимости блюда
freecount - Кол-во бесплатных. Если кол-во модификаторов ⇐ freecount, то их стоимость 0, далее по price.
items - Список модификаторов в группе.
items[n]:
ident - Идентификатор модификатора
code - Код модификатора
name - Название модификатора
limit - Максимальное кол-во данного модификатора для одного блюда
price - стоимость одной порции модификатора

Валидация корзины

Для проверки возможности добавления в заказ содержимого корзины гостя предусмотрен запрос **POST /validate**

В теле POST запроса ожидается JSON объект, частично описывающий заказ гостя:

```
{
  "table": 1,
  "type": 1,
  "discount": 3,
  "content": [
    {
      "code": 61,
      "quantity": 1,
      "modifiers": [
        {
          "code": 31,
          "count": 1
        }
      ]
    }
  ]
}
```

(так отмечены НЕ обязательные поля**)**

(table) - (int) Код стола из r_keeper. Если передан, то будет использован при создании заказа на кассе. Если не передан, то будет использован код стола из конфигурации мини-доставки.
(type) - (int) Тип заказа. 0 - Самовывоз, 1 - доставка.
(discount) - (int) Код скидки из справочников rk7, с учетом которой будет рассчитана сумма заказа.
content - (array) Массив блюд, составляющих корзину гостя. (Не может быть пустым)
content[n].code - (int) Код блюда из справочников r-keeper, полученный в запросе /getmenu
content[n].quantity - (int) Количество порций блюда.
(content[n].modifiers) - (array) Список модификаторов, добавляемых к блюду.
modifiers[n].code - (int) - Код модификатора
modifiers[n].count - (int) - Кол-во модификаторов на порцию блюда

Результат выполнения запроса передастся в поле JSON объекта result, и в случае result = false будет передано поле msg, содержащее текст ошибки.
В случае result = true в поле **order** будет передана информация о заказе, каким его видит r-keeper:

```
"order": {
    "amount": 20700,
    "discountsum": 2300,
    "items": [
      {
        "code": 61,
```

```
        "name": "Кебаб из курицы",
        "price": 22000,
        "quantity": 1,
        "discounted": 1300,
        "total": 20700,
        "modifiers": [
          {
            "code": 31,
            "name": "Модификатор 2 1",
            "count": 1
          }
        ]
      }
    ],
    "discounts": [
      {
        "code": 3,
        "name": "11",
        "total": 2300
      }
    ]
  }
```

amount - Сумма заказа, рассчитанная r-keeper (с учетом возможной скидки)
discountsum - Общая сумма скидки по всему заказу

items - Список блюд, которые будут добавлены в заказ.
items[n].discounted - Общая сумма скидки по данному блюду (с учетом количества)
items[n].total - Окончательная сумма по блюду (с учетом количества и суммы)
items[n].modifiers - Информация о модификаторах, привязанных к блюду

discounts - Список скидок, которые будут добавлены в заказ.
discounts[n].code - Код скидки, примененной к заказу
discounts[n].name - Имя скидки
discounts[n].total - Общая сумма скидки

На основе ответа на данный запрос внешняя система должна скорректировать корзину гостя, пересчитать суммы по блюдам с учетом скидок, дабы показать актуальную информацию об окончательной сумме заказа.
Не стоит пренебрегать этим запросом если Вы не планируете передавать скидку гостя в заказ, ведь в ресторане может быть настроена автоматическая скидка.

Создание заказа

Для создания заказа необходимо отправить POST запрос **/postorder**, в теле которого должен содержаться JSON объект следующего содержания:

```
{
```

```
  "guest": {
    "sname": "Иванов",
    "fname": "Иван",
    "mname": "Иванович",
    "cardcode":"1000",
    "meta": {},
    "phone": "+7 (963) 949-99-99 ",
    "address": "Ул.Федосеева",
    "longitude":85.131548,
    "latitude":55.164833
  },
  "order":{
    "table" : 2,
    "type": 1,
    "comment": "Две персоны",
    "status_callback": "https://yourурl.com/yourmethod",
    "discount": 3,
    "paid":6000,
    "deliverat": "2020-04-08 22:30:00",
    "content": [
      {
        "name": "Кебаб из курицы",
        "code": 55,
        "quantity": 1,
        "modifiers":[
          {
            "code": 31,
            "name": "Модификатор 2 1",
            "count": 1
          }
        ]
      }
    ]
  }
}
```

(Так отмечены необязательные поля**)**

guest.[s,(f,m)]name - (str) Фамилия, Имя и Отчество гостя соответственно.
guest.phone - (str) Номер телефона гостя. **Все спецсимволы будут удалены**
guest.address - (str) Адрес доставки (необязателен при order.type = 0)

(guest.longitude), (guest.latitude) - (float) - Долгота и широта адреса доставки. *не передавайте **null**
введено с версии 3.0.5.983
(guest.meta) - (Object) Метаданные о пользователе. JSON объект любого содержания.
Не используется в логике ПО, но возвращается в orderstate, expressorders и order.status_callback
введено с версии 3.0.5.1053
(guest.cardcode) - (str) Код карты системы лояльности (например ПДС) (**НЕ РЕКОМЕНДУЕТСЯ К ИСПОЛЬЗОВАНИЮ**)
введено с версии 3.0.5.1053

order.table - (int) Код стола из r_keeper. Если передан, то будет использован при создании заказа на кассе. Иначе будет использован тот, что указан в конфигурации мини-доставки.
order.type - (int) Тип заказа. 0 - Самовывоз, 1 - Доставка.
(order.comment) - (str) Комментарий к заказу
(order.discount) - (int) Код скидки из справочника r-keeper, которая должна быть добавлена в заказ.
(order.paid) - (int) Сумма произведенной гостем оплаты в копейках. Если оплаты нет, то не передавать данное поле
order.deliverat - (str) Время, к которому гость будет ждать доставку в формате **yyyy-mm-dd hh:mm:ss**
Запас по времени должен обсуждаться с конкретным заведением.

(order.status_callback) - (url string) URL адрес, на который будут отправлены уведомления об изменении статуса заказа. **ВНИМАНИЕ. Внутри доставки адрес приводится к нижнему регистру (HttP://Ya.Ru/Status → http://ya.ru/status)**
введено с версии 3.0.5.1053
order.content - (array) Список блюд, заказанных гостем. **order.content[n].name, code** - (str, int) Название и код блюда, которые были получены в запросе /getmenu
order.content[n].quantity - (int) Целочисленное количество порций блюда.
order.content[n].modifiers - (array) Целочисленное количество порций блюда.
order.content[n].modifiers.name, code - (str, int) Название и код модификатора.
order.content[n].modifiers.count - (int) Кол-во модификатора на порцию блюда.

В ответ на запрос вернется JSON объект, содержащий поле result:boolean
Если result = true, то заказ создан и так же передано поле order_id:int - идентификатор нового заказа
Если result = false, то при создании заказа возникла ошибка, текст которой будет передан в поле msg:str

Получение статуса заказа

После успешного сохранения заказа можно получить его статус:

GET /orderstate?order_id=**[order_id]**

*order_id - ИД созданного заказа, который вернулся в ответ на **/postorder**

В ответ на запрос вернется JSON объект, где
result:boolean - результат выполнения запроса
если result == False, то будет передан msg - текстовое описание ошибки, иначе будет передано поле order, аналогичное таковому из запроса /validate, но дополнительно содержащее следующие поля:

id - Идентификатор заказа на стороне мини-доставки (равен order_id из запроса)
status - Текуйщий статус заказа. Расшифровку статусов см ниже.
paid - Сумма оплат в заказе

А так же поле guestmeta, содержащее метаданные о госте, переданные при создании заказа.

Пример ответа:


```
{
  "result": true,
  "guestmeta": {
  },
  "order": {
    "id": 5,
    "status": 10,
    "amount": 46000,
    "discountsum": 0,
    "paid": 46000,
    "items": [
      {
        "code": 61,
        "name": "Кебаб из курицы",
        "price": 22000,
        "quantity": 2,
        "discounted": -2000,
        "total": 46000,
        "modifiers": [
          {
            "code": 31,
            "name": "Модификатор 2 1",
            "count": 1
          }
        ]
      }
    ],
    "discounts": [
    ]
  }
}
```


Расшифровка статусов заказов:

- 0 - «Ожидает» - Такой статус имеют заказы только что созданные через API
  Они еще не обработаны оператором в ресторане.
  Заказы в статусе 0 не возвращают содержимое (массив items)
- 1X - «В работе» - Заказ обработан в заведении и передан на кухню.
- 2X - «В пути» - Заказ отправлен с курьером
- 3X - «В архиве» - Курьер вернулся в ресторан, заказ полностью завершен
- 4X - «Отменен» - Работники ресторана отменили заказ по той или иной причине

Получение списка активных заказов

Введено в версии 3.0.5.983 Позволяет получить список заказов мини-доставки.

GET /expressorders

Входящих параметров не ожидается

В ответ вернется JSON объект, содержащий информацию о текущих принятых (принятых оператором на кассе) заказах доставки

```
{
  "result": true,
  "data": [
    {
      "id": 139,
      "guestmeta": {
        "EXTID": 122
      },
      "comment": "БАНК. - ",
      "dlvdate": "2020-05-04 10:00:00",
      "status": 10,
      "sum": 1000,
      "waiter": "Петров Петр",
      "address": "ул.Пушкина 7, кв 112",
      "longitude": null,
      "latitude": null,
      "phone": "71234567890",
      "fullname": "Иванов Иван"
    }
  ]
}
```

result - Результат выполнения запроса. Если **false**, то так же будет передано поле **msg:string** содержащее текстовое описание ошибки.

data - список открытых в данный момент заказов.
Далее для data[n]:
id - Идентификатор заказа в системе
guestmeta - Метаданные о госте, переданные при создании заказа.
comment - Комментарий к заказу
dlvdate - Дата и время, на которое назначена доставка
status - Текущий статус заказа в системе (см Получение статуса заказа → Расшифровка статусов заказов)
sum - Сумма заказа в копейках
waiter - Имя пользователя в системе r-keeper, на которого назначен заказ
address - Адрес доставки
longitude, latitude - (float) Долгота и широта адреса доставки (может быть null)
phone - Телефон гостя
fullname - ФИО гостя

From:
https://wiki.carbis.ru/ - **База знаний ГК Карбис**

Permanent link:
https://wiki.carbis.ru/external/%D0%B4%D0%BE%D1%81%D1%82%D0%B0%D0%B2%D0%BA%D0%B0/api/extv2?rev=1643185150

Last update: **2022/01/26 08:19**