

Плагин для Mindbox

????????? ????????	Mindbox
???????????????	?????????????? ??????? ? Metabot
???????	????????? ?????? ?????????????? (ira.petrova@metabot.org) ????????? ?????? ?????????? (artem@metabot.org)
????? ???????????	22 ??????? 2022
????????????? ????? ??????????????	26 ??????? 2022

Описание

Mindbox — это платформа автоматизации маркетинга и клиентских данных. Этот плагин к платформе Metabot позволяет интегрировать ваш чат-бот, разрабатываемый на Metabot, с платформой Mindbox. Подробнее с Mindbox можно ознакомиться на их официальном сайте: <https://mindbox.ru>.

Плагин позволяет автоматически добавлять клиентские данные, собранные чат-ботом в ходе диалога с клиентом, в Mindbox. С помощью плагина вы сможете организовать передачу данных в единый профиль клиента в Mindbox прямо из диалога в окне чата на сайте, в мессенджере или социальной сети.

Подключение

Для интеграции Mindbox с вашим чат-ботом вам необходимо сделать несколько вещей:

1. На стороне Mindbox настройте все нужные точки интеграции и операции.
Подробнее смотрите в [документации Mindbox](#).
2. [Зарегистрируйтесь](#) на платформе Metabot, подтвердите почту, [авторизуйтесь](#) и создайте чат-бот.
3. Используйте готовый общий плагин или создайте плагин для своего бизнеса и скопируйте в него наш исходный код. Инструкции для обоих вариантов будут указаны ниже.

4. Создайте в чат-боте системный атрибут **mindbox.secretKey.<Системное имя точки интеграции>** и сохраните в него секретный ключ (токен) авторизации API запросов к Mindbox.
 - Если у вас планируется несколько точек интеграции в чат-боте, то нужно задать свой ключ для каждой из них. Например, так:
 - Mindbox.SecretKey.MyBusiness.Chatbot - атрибута с ключом для точки MyBusiness.Chatbot
 - Mindbox.SecretKey.MyBusiness.Chatbot.Shop - атрибута с ключом для точки MyBusiness.Chatbot.Shop
 - Внимание! Называйте атрибуты в точности и с учетом регистра именно так, как они указаны в Mindbox.
5. Реализуйте диалоговый сценарий (скрипт) с опросом данных пользователя (например, имя, фамилия, email адрес, телефон и прочее).
6. В конце диалога, обязательно (!) запросите согласие пользователя на обработку персональных данных и пришлите ссылку на положение о конфиденциальности компании. Если подписываете на маркетинговую рассылку, то дополнительно запросите согласие. Храните согласие в атрибуте лида вашего чат-бота на случай, если это понадобится юридической службе вашей компании.
7. Выберите универсальный или упрощенный метод для генерации запроса к Mindbox (смотрите детали ниже) и скопируйте код примера к себе в скрипт.
8. Кастомизируйте код интеграции под свои задачи. Если у вас возникнут затруднения, обратитесь за помощью в [Телеграм-чат](#) сообщества или [поддержку Metabot](#).

Вызов в диалоге

Вы можете использовать одну из двух функций на ваш выбор. Первая - универсальная - принимает тело JSON запроса в качестве одного из параметров, вторая - параметризованная - принимает поля в качестве параметров и формирует тело запроса JSON с помощью первой функции.

Способ 1. Универсальная (JSON) функция

Первый способ использования плагина в диалоге — с помощью универсальной функции **callMindboxEndpoint()**. Вы сами формируете тело запроса в формате JSON и передаете его в функцию.

Пример использования универсального (JSON) метода:

```
let email = lead.getAttr("email")
let phone = lead.getAttr("phone")
let lastName = lead.getAttr("lastName")
let firstName = lead.getAttr("firstName")
```

```

let endpointId = "<Идентификатор точки интеграции>"
let operation = "<Системное имя операции>"

let jsonBody = { "customer": {
  "email": email,
  "mobilePhone": phone,
  "lastName": lastName,
  "firstName": firstName,
  "customFields": {
    "AdCommunicationAgreement": true,
    "PersonalDataAgreement": true
  },
  "subscriptions": [
    {
      "brand": "<Системное имя бренда подписки клиента>",
      "pointOfContact": "<Системное имя канала подписки: Email, SMS, Viber, Webpush,
Mobilepush>",
      "topic": "<Внешний идентификатор тематики подписки>"
    }
  ]
},
"pointOfContact": "<Внешний идентификатор точки контакта>"
}

// Подключаем сниппет кода из плагина Mindbox
snippet('Common.Mindbox.Operations')

// Вызываем точку интеграции Mindbox и передаем нужный нам JSON
callMindboxEndpoint(endpointId, operation, jsonBody)

```

Функция требует передачи трех переменных **в строгом порядке**:

endpointId	Уникальный идентификатор интеграции. Не забудьте, что каждому endpointId соответствует свой secretKey, который нужно сохранить в системную атрибуту бота. Интеграции настраивается в системе Mindbox.
operation	Название операции в Mindbox. Каждому типу действия в Mindbox соответствует своя операция. Список операций настраивается в системе Mindbox.

jsonBody	Тело запроса в формате JSON. Формат тела запроса, различается в зависимости от типа операции.
----------	--

Способ 2. Альтернативная (параметризованная) функция

Второй способ использования плагина — с помощью альтернативной функции **callMindboxEndpointAlt()**, в которую вы передаете параметры запроса, а функция внутри себя формирует тело запроса в формате JSON и затем вызывает описанную выше универсальную функцию.

Пример использования альтернативного (параметризованного) метода:

```
let email = lead.getAttr("email")
let phone = lead.getAttr("phone")
let lastName = lead.getAttr("lastName")
let firstName = lead.getAttr("firstName")
let endpointId = "<Идентификатор точки интеграции>"
let operation = "<Системное имя операции>"

// Подключаем snippet кода из плагина Mindbox
snippet('Common.Mindbox.Operations')

// Вызываем точку интеграции Mindbox и передаем нужные нам параметры
callMindboxEndpointAlt(endpointId, operation,
                       email, phone, lastName, firstName,
                       subscriptionTopic, pointOfContact)
```

Функция требует передачи нескольких переменных **в строгом порядке**. Добавляйте и удаляйте параметры по вкусу ;) но при этом не забудьте поменять в плагине код формирования JSON.

endpointId	Уникальный идентификатор интеграции. Не забудьте, что каждому endpointId соответствует свой secretKey, который нужно сохранить в системную атрибуту бота. Интеграции настраивается в системе Mindbox.
operation	Название операции в Mindbox. Каждому типу действия в Mindbox соответствует своя операция. Список операций настраивается в системе Mindbox.

email	Email пользователя
phone	Мобильный телефон
lastName	Фамилия
firstName	Имя
subscriptionTopic	Внешний идентификатор тематики подписки. Настраивается в Mindbox.
pointOfContact	Внешний идентификатор точки контакта. Настраивается в Mindbox.

Варианты подключения плагина

Вы можете использовать один из двух вариантов: воспользоваться общим плагином без модификаций кода или же скопировать код плагина к себе и модифицировать код.

Вы можете использовать общий плагин и вызвать скрипт "Вызов операции" в точности следуя примерам выше. В этом случае, для использования общего плагина используйте вызов сниппета из коллекции общих плагинов **Common**:

```
snippet(' Common. Mindbox. Operations' );
```

Либо, вы можете создать свой плагин в вашем бизнесе, скопировать наш и изменив код под себя, для этого:

1. Создайте плагин в вашем бизнесе и назовите его **Mindbox**.
2. Создайте в плагине скрипт и назовите его **Operations**.
3. Скопируйте код, размещенный ниже, в код нового скрипта.

В этом случае, в примерах, указанных выше, замените вызов общего сниппета на ваш собственный:

```
snippet(' Business. Mindbox. Operations' );
```

В обоих случаях, перед вызовом сниппета требуется объявить все необходимые переменные и передать им соответствующие значения.

При успешном запросе во вкладке "Клиенты" в Mindbox будет создан клиент с переданными из чат-бота данными о нем:

<input type="checkbox"/>	Клиент	Телефон	Email	ID
<input type="checkbox"/>	Петрова Ирина	—	irenpdm@gmail.com	776

Так же, во вкладке "Действия" вы сможете найти историю выполненной операции:

<input type="checkbox"/>	Действие	Клиент	Дата	Бренд	Точка контакта
<input type="checkbox"/>	Подписка клиента в операции 'Подписка через чат-бот на сайте My Business' Подписка на рассылку	Петрова Ирина	23.11.2022, 18:20:07 23.11.2022, 18:20:08	My Business	Web > ... > Сайт My Business

Обработка ошибок

В случае успешного выполнения оба метода возвращают **true**. В случае ошибки оба метода возвращают **false**, а в атрибутах лида и бота вы сможете найти информацию о последней ошибке. Название атрибуты и описание указано в таблице ниже.

Хранилище	Название атрибута	Описание
lead	plugin.mindbox.lastError.code	В случае ошибки, будет содержать код ошибки плагина. Смотрите таблицу ошибок ниже.
bot		
lead	plugin.mindbox.lastError.message	В случае ошибки, будет содержать сообщение об ошибке плагина. Смотрите таблицу ошибок ниже.
bot		

Список ошибок

В случае ошибки плагин может вернуть одну из следующих ошибок, а также вернет сообщение об ошибке. В таблице ниже представлены все возможные ошибки плагина и рекомендации, что вы можете сделать в каждом конкретном случае.

Код ошибки	Сообщение об ошибке	Рекомендации
------------	---------------------	--------------

1	Ошибка вызова Mindbox API ({код ошибки}).	<p>Плагин вернет код ошибки Mindbox API. Поскольку чат-бот не запоминает детали вызова API потому что вызовы асинхронны (async), при получении данной ошибки детали смотрите на стороне Mindbox.</p> <p>Все возможные коды ошибок Mindbox API смотрите в документации к Mindbox: https://developers.mindbox.ru/docs/error_processing</p> <p>Подробнее про вызов операции через Mindbox API, который применяется в этом плагине, смотрите по ссылке: https://developers.mindbox.ru/docs/v3</p>
2	Не задан ключ интеграции {имя атрибута} в атрибутах бота.	Проверьте не ошиблись ли вы в регистре символов или названии.
3	Не заданы обязательные параметры параметризованного метода.	Проверьте не пытаетесь ли вы передать пустые данные в Mindbox в одном из полей (смотрите поля в атрибутах лида). А если параметр допустимо передавать пустым, удалите соответствующую валидацию в коде в <code>callMindboxEndpointAlt()</code> .

Исходный код плагина

Скопируйте указанный ниже код в скрипт Operations вашего плагина Mindbox и измените как вам требуется.

```
/**
 * Универсальная функция для регистрации операции в Mindbox через точку интеграции.
 * Используется явное указание тела запроса в формате JSON.
 * @param {string} endpointID - Идентификатор точки интеграции в Mindbox
 * @param {string} operation - Системное имя операции
 * @param {object} jsonBody - Тело запроса (берется из настроек операции в Mindbox)
 * @returns {bool} - Результат выполнения функции: успешно (true), проблемы (false).
 */
function callMindboxEndpoint(endpointId, operation, jsonBody) {
```

```

// Считываем ключ авторизации из атрибуты бота
let secretKey = bot.getAttr('mindbox.secretKey.' + endpointId)
// Если ключ не настроен
if (!secretKey) {
    outputError(endpointId, operation, '2', 'Не задан ключ интеграции plugin.Mindbox.secretKey
в атрибутах чат-бота.')
    return false
}

// Задаем заголовок запроса
api.setHeaders({'authorization':'Mindbox secretKey="' + secretKey + '"})
// Задаем URL запроса (используем асинхронный 'async' метод)
let url = "https://api.mindbox.ru/v3/operations/async?endpointId=" + endpointId +
"&operation=" + operation

// Выполняем запрос с помощью метода POST и запоминаем результат
let jsonResponse = api.postJson(url, jsonBody)
let jsonResponseCode = api.getLastResponseCode()

// Возникла ошибка
if (jsonResponseCode != 200) {
    outputError(endpointId, operation, '1', 'Проблема вызова Mindbox API. Детали смотрите в
Mindbox.')
    return false // Выполнено с проблемами
}

return true // Все ок
}

/**
 * Альтернативная (параметризованная) функция для регистрации операции в Mindbox, который
подготавливает тело запроса.
 * Адаптируйте эту функцию под свой проект или создайте копию.
 * @param {string} endpointID - Идентификатор точки интеграции в Mindbox
 * @param {string} operation - Системное имя операции
 * @param {string} email - Email
 * @param {string} phone - Мобильный телефон
 * @param {string} lastName - Фамилия
 * @param {string} firstName - Имя
 * @param {string} subscriptionTopic - Внешний идентификатор тематики подписки

```

```
* @param {string} pointOfContact - Внешний идентификатор точки контакта
* @returns {bool} - Результат выполнения функции: успешно (true), проблемы (false).
*/
function callMindboxEndpointAlt(endpointId, operation, email, phone, lastName, firstName,
subscriptionTopic, pointOfContact) {
    // Если передали пустые значения
    if (isStrEmpty(endpointId) ||
        isStrEmpty(operation) ||
        isStrEmpty(email) ||
        isStrEmpty(phone) ||
        isStrEmpty(lastName) ||
        isStrEmpty(firstName) ||
        isStrEmpty(subscriptionTopic) ||
        isStrEmpty(pointOfContact))
    {
        outputError('З', 'Не заданы обязательные параметры параметризованного метода.')
        return false
    }

    // Формируем тело запроса в JSON (в вашем проекте запрос может отличаться)
    let jsonBody = {
        "customer": {
            "email": email,
            "mobilePhone": phone,
            "lastName": lastName,
            "firstName": firstName,
            "customFields": {
                "AdCommunicationAgreement": true, // Согласие на рассылку
                "PersonalDataAgreement": true // Согласие на обработку персональных данных
            },
            "subscriptions": [
                {
                    "brand": "<Системное имя бренда подписки клиента>",
                    "pointOfContact": "<Системное имя канала подписки: Email, SMS, Viber, Webpush,
Mobilepush>",
                    "topic": subscriptionTopic
                }
            ]
        },
        "pointOfContact": pointOfContact
    }
```

```

}

// Вызываем универсальную функцию и возвращаем результат
return callMindboxEndpoint(endpointId, operation, jsonBody)
}

/**
 * Вспомогательная функция, которая сохраняет сведения об ошибке в атрибутах лида и атрибутах
бота.
 * @param {string} endpointId - Точка интеграции
 * @param {string} operation - Операция
 * @param {string} code - Код ошибки
 * @param {string} message - Сообщение об ошибке
 */
function outputError(endpointId, operation, code, message) {
  // Добавляем в конце сообщения доп. инфу.
  message = message + ' (точка=' + endpointId + ', операция=' + operation + ')'
  lead.setAttr("plugin.Mindbox.lastError.Code", code)
  lead.setAttr("plugin.Mindbox.lastError.Message", message)
  bot.setAttr("plugin.Mindbox.lastError.Code", code)
  bot.setAttr("plugin.Mindbox.lastError.Message", message)
}

/**
 * Вспомогательная функцию, которая проверяет является ли строка пустой.
 * @param {string} code - Код ошибки
 * @returns {bool} - если строка пустая (true), иначе (false).
 */
function isStrEmpty(str) {
  return (typeof str === 'string' && str.trim().length === 0) ? true : false
}

```

Версия #2

Юрий Гарашко создал 14 June 2023 13:51:25

Ирина Петрова обновил 5 February 2024 14:06:47