Плагины BETA (14.03.2022)

Данный модуль позволяет реализовать интеграцию практически с любой внешней системой, которая имеет API, либо может присылать данные по средствам вебхука.

Доступ к функционалу ограничен и выдаётся индивидуально!

Создание/Редактирование плагина

Название

Будет отображаться в качестве названия кнопки.

Активность

Будет ли плагин доступен пользователю.

Глобальный плагин

Данный вид плагин доступен во всех частях системы и отображается в главном меню (слева). Глобальный плагин не предназначен для работы с данными/состоянием заявки.

Доступ отображению

Доступ к плагину можно ограничить по департаменту заявки, а также по группе пользователя.

Настройки кнопки

Внешний вид кнопки. Кнопка отображается в левой панели внутри заявки над функциональными кнопками. Очередность кнопок регулируется из списка плагинов по средствам перетаскивания.

Настройка диалогового окна

Ширина и высота открывающегося окна, в котором будет отображаться плагин. Отображение плагина необязательно.

Доступы для авторизации

Плагин может содержать множество учётных данных для авторизации во внешнее API. На данный момент доступно 2 метода авторизации: базовая и по API ключу (bearer).

Настройки авторизации, по-мимо доступов во внешнее API, содержит поле «Идентификатор», которое необходимо использовать для совершения запросов во внешнее API.

Пример:

```
HDE.request({
      auth: ‘hde’ // Идентификатор авторизации
      url: ‘ссылка внешнего API’,
      method: ‘POST/GET/PUT/PATCH/DELETE/‘,
      contentType: ‘application/json’,
      data: ‘данные для запросов POST/PUT/PATCH’
}).then(({data}) => {
            console.log(data) // Ответ от внешнего API
})
```

Входящие вебхуки

Плагин может содержать множество входящих ссылок для получения данных из внешних систем. Входящая ссылка содержит уникальный ключ плагина, идентификатор вебхука, а также уникальное поле вебхука, которое будет являться уникальным ключом для хранения полученных данных.

Пример:

Идентификатор вебхука: test-webhook

Уникальное поле вебхука: email

Ссылка вебхука: https://demo.helpdeskeddy.com/api/v2/plugins/sNgeLTBvJHDPcTRcTJFg/test-webhook

Входящие данных присланные в формате JSON на ссылку вебхука методом POST:

```
{
	id: 1,
	name: ‘Helpdeskeddy’,
	email: ‘sample@hde.com'
}
```

```
HDE.webhook({
	endpoint: ‘test-webhook’,
	value: ‘sample@hde.com’
}).then(({data}) => {
	console.log(data)  // Если данные с указанным значением найдены, то они будут возвращены,          в противном случае - false
})
```

Переменные

Доступно 2 вида переменных: серверные и клиентские.

Пример использования клиентских:

```
HDE.on(‘ready’, () => {
	console.log(HDE.vars.CLIENT_VAR)
	const apiURL = ‘{{API_URL}}’
})
```

Пример использования серверных:

```
HDE.on(‘ready’, () => {
    HDE.request({
      data: {‘API_KEY’: ‘{{API_KEY}}’}
      headers: {'X-API-KEY’: ‘{{API_KEY}}’}
    })
})
```

Содержание плагина HTML/JS

Данное поле может содержать любой HTML код, который будет вставлен в тег <body> страницы плагина. Javascript код необходимо писать между тегов <script></script>

Доступные методы HDE

Функционал находится на стадии бета, доступные методы могут быть изменены.

Все ниже описанные методы доступны по обращению к глобальной переменной: HDE.метод()

```
Описание методов для заявок:
HDE.on('ready', () => {
      HDE.watch('plugin', (to, from) => {}) // слежение за изменением состояние. В текущий реализации доступно только изменения «plugin».

      HDE.getState() // позволяет получить состояние заявки. Включает ticketData, ticketValues. Узнать их содержимое можно через console.log(HDE.getState())

      HDE.emit('setPlugin', state.plugin) // позволяет устанавливать состояние заявки. к примеру, если плагин предполагает отображение информации пользователю в диалоговом окне, то необходимо установить значение состояния «plugin.showButton = true», в противном случае, кнопка не будет отображаться

//Доступные методы

HDE.emit('setTicketValue', {field, value})
HDE.emit('setTicketValues', [{field, value}])
HDE.emit('setTicketData', {field, value})
HDE.emit('setTicketCustomFieldValue', {customFieldId, value})

})
```

Описание общих методов:

HDE.request({ // совершение запросов во внешнее API
 url: '', // ссылка во внешние API
 auth: '', // идентификатор авторизации
 method: '', // метод GET/POST/PUT/PATCH/DELETE
 contentType: '', // тип данных application/json, application/x-www-form-urlencoded
 data: {} // данные
 headers: {} // заголовки
}).then(({ data }) => {
 console.log(data)
})

HDE.webhook({ // получение данных присланных из внешнего источника
 endpoint: '', // идентификатор вебхука
 value: '' // значение по которому будет осуществлён поиск по присланным данным
})

HDE.ajax({ // совершение любых ajax запросов

 url: '', // ссылка ajax запроса
 method: '', // метод GET/POST/PUT/PATCH/DELETE
 responseType: '', // тип данных ответа
 data: {} // данные
 params: {} // GET параметры
}).then(({ data }) => {console.log(data)})

HDE.saveVar([// Сохранение переменное плагина
 {
 varName: "REFRESH_TOKEN", // Название переменное
 varValue: data.refresh_token, // Значение переменное
 varType: "server", // Тип переменное server/client
 readOnly: 1 // Только для чтения 1/0
 },
 {
 varName: "DOMAIN",
 varValue: data.domain,
 varType: "client",
 readOnly: 1

 ]);