

АО ИНТЕГРАМ

Веб-сервис Интеграм

Документация, содержащая описание
функциональных характеристик ПО

Москва, 2025

Оглавление

Функциональные характеристики веб-сервиса	3
Ядро	3
Базовый пользовательский интерфейс.....	3
Терминология	3
Виртуальная база данных	4
Предустановленная конфигурация базы	4
Общие правила	4
Список команд API	5
Примеры обращений к API	7

Функциональные характеристики веб-сервиса

Ядро

Веб-сервис Интеграм предоставляет пользователю функциональную возможность создания и управления структурами данных и самими данными, а также создания произвольных рабочих мест для отображения и обработки данных и структур.

Набор основных функциональных характеристик программы включает методы (команды API) для выполнения следующих функций:

- DDL (Data definition language)
- DML (Data modification language)
- Запрос метаданных (описания данных)
- Построение запроса к данным
- Запросы, связанные с авторизацией пользователя

Таким образом, Веб-сервис Интеграм представляет собой ядро для управления данными – бэкенд, с использованием которого строятся пользовательские приложения – фронтенд и кастомный бэкенд. Общение с ядром Интеграм происходит посредством API.

Особенность сервиса в том, что для программирования бэкенда не требуется править или дорабатывать его ядро.

Базовый пользовательский интерфейс

Веб-сервис Интеграм изначально снабжен базовым пользовательским интерфейсом, реализованными на HTML, CSS и Javascript для работы с его ядром. Базовый интерфейс состоит из пяти универсальных форм и специализированного меню для работы с файловой системой:

1. Форма редактирования терминов и структуры данных — меню Структура
2. Список терминов (таблиц) системы — меню Таблицы
3. Форма для работы с таблицей
4. Форма редактирования записи таблицы
5. Форма отчета
6. Администратор директорий, где можно загружать шаблоны страниц

Базовый интерфейс Интеграма позволяет пользователю делать все те же действия, которые доступны по API. Есть некоторые команды, которые доступны только по API.

Терминология

В конструкторе есть понятие *термин* — это любой термин, которым оперирует разработчик или пользователь. Программист может называть термины *классами* или *объектами* в зависимости от контекста.

Примеры терминов: Пользователь, Клиент, Адрес, Статус. Пользователь работает с

таблицами данных, поэтому он чаще использует слово *таблица* для определения объекта действия, например: таблица Пользователь или таблица Клиент.

Термины могут быть подчинены друг другу и связаны с другими терминами, например: у Клиента есть Адрес и Телефон. В этом случае таблица клиентов будет выглядеть так:

Клиент ***	Адрес ***	Телефон ***
ПАО Эллипс	Центральная, 7	8(555)6543210
АО Ромашка	Летняя, 21	8(555)0123456

При этом пользователь назовет все эти термины *колонками* таблицы Клиент, в то время как программист скажет, что Адрес и Телефон — это *реквизиты* термина Клиент.

Виртуальная база данных

Пользователь работает с виртуальной базой данных, состоящей из виртуальных таблиц, в то время как на самом деле данные всей его базы данных хранятся в одной этой физической таблице.

Правила работы по API, изложенные ниже, описывают обращение к виртуальной базе данных пользователя. Исключение составляет создание самой виртуальной базы, которое также делается по API.

Предустановленная конфигурация базы

После установки инстанса вес-сервиса, пользователь получает доступ к собственной базе, в которой уже существуют такие термины как Пользователь, Роль, Запрос, имеющие свой набор реквизитов. Интеграм знает, как обращаться с этими терминами — Роль пользователя определяет уровни доступа ко всем объектам базы, Запрос выбирает данные из базы и так далее. Предустановленные термины необходимы в любом приложении, а далее пользователь создает необходимые ему термины самостоятельно. Пользователь, созданный при регистрации вместе с базой, имеет имя, совпадающее с именем базы, и права администратора в системе.

Общие правила

Обращение по API выполняется по адресу сервиса с указанием вашей базы, действия и ID объекта, над которым производится действие, если это применимо. Также необходимо указать параметр JSON, чтобы сервис понял, что это вызов API, и вернул результат в JSON:

`integram.io/{база}/{действие}/{id}?JSON=1`

Обязательный параметр действие может иметь два префикса для команд на изменение данных:

`_d_*` — изменение структуры данных (DDL, команды Редактора структуры);

`_m_*` — изменение самих данных (DML, команды базового интерфейса);

Некоторые команды (например, xcsrf, metadata, exit) используются только по API, поэтому для них нет необходимости указывать параметр JSON, например, как для метаданных термина Роль:

<https://integram.io/apix/metadata/42>

Далее, для унификации, все эти вызовы будут приведены в формате CURL:

```
curl --location 'integram.io/apix/metadata/42'
```

Ответ придет в формате JSON с описанием структуры таблицы Роль, где reqs — это описание её колонок:

Список команд API

В таблице приведен список команд, допустимые методы, применимость параметра ID и описание других параметров.

Команда	Метод	ID объекта	Описание	Параметры
auth	post		Авторизация пользователя, возвращает токены XSRF и авторизации	login={пользователь}, pwd={пароль}
_d_alias	post	id	Сохранить псевдоним для ссылочного реквизита	val={значение}
_d_attrs	post	id	Сохранить атрибуты реквизита (значение по умолчанию)	val={значение}
_d_del	post	id	Удалить тип	
_d_del_req	post	id	Удалить реквизит	
_d_multi	post	id	Включить/отключить множественный выбор для ссылочного рквизита	
_d_new	post	-	Создать новый тип (таблицу)	val={значение}, unique=1 (1 — первая колонка уникальна, пусто — нет), t={id базового типа}
_d_null	post	id	Установить/Снять флаг "Обязательный" у реквизита	

Команда	Метод	ID объекта	Описание	Параметры
_d_ref	post	id	Создать ссылку на тип	
_d_req	post	id	Добавить реквизит	t={id типа реквизита}
_d_save	post	id	Сохранить имя типа, его базовый тип и признак уникальности	val={значение}, unique=1 (передавать если установлен), t={id базового типа}
_d_up	post	id	Переместить реквизит на 1 позицию выше среди равных	
_m_del	post	id	Удаление объекта	
_m_move	post	id	Перемещение подчиненного объекта под другого родителя	up={id целевого родителя}
_m_new	post	id типа	Создание объекта заданного типа	Объект: t{id типа}={значение}, Для подчиненного типа: up={id родителя}, Необязательно: Реквизиты: t{id типа}={значение}
_m_save	post	id	Сохранить значение и реквизиты объекта, если они заданы	Объект: t{id типа}={значение} Реквизиты: t{id типа}={значение}
_m_set	post	id	Изменить / создать / удалить реквизит объекта	Реквизиты: t{id типа}={значение}
_m_up	post	id	Переместить объект на 1 позицию выше среди равных	
backup	post	id	Создать архив базы (в разработке)	
dict	get	-	Список независимых типов объектов	

Команда	Метод	ID объекта	Описание	Параметры
dir_admin	get, post	-	Список файлов и директорий, действия с ними	
edit_obj	get	id	Данные для формы редактирования объекта	
edit_types	get	-	Данные для Редактора типов	
metadata	get	id	Получение метаданных указанного объекта (таблицы)	
object	get, post	id типа	Данные для формы списка экземпляров типа объекта, команда на удаление	_m_del_select — если этот параметр передан, то будут удалены все объекты, попадающие под текущий фильтр
report	get, post	id отчета	Данные выполненного запроса (отчета), параметры, команда на выполнение изменений	_m_confirmed — если этот параметр установлен, то выполнятся изменения запроса
xsrif	get	-	Получение информации о сессии пользователя	

Примеры обращений к API

Описание команд и примеры использования API Интеграм находится по адресу:

<https://integram.io/api.html>