

# **API Документация mws**

Губанов Антон Дмитриевич

Exported on 02/10/2025

## Table of Contents

1. Пример обращения в API для листинга моделей .....	4
Эндпоинт .....	4
Тип запроса.....	4
Пример запроса.....	4
Ответ API.....	4
2. Пример обращения в Chat Completion (чат-комплишн).....	5
Для чего используется .....	5
Эндпоинт .....	5
Тип запроса.....	5
Пример тела запроса .....	5
Ответ API.....	5
Описание доступных параметров: .....	6
3. Пример обращения в Completion .....	7
Для чего используется .....	7
Эндпоинт .....	7
Тип запроса.....	7
Пример тела запроса .....	7
Ответ API.....	7
Описание доступных параметров: .....	8
4. Пример обращения в Embeddings.....	9
Для чего используется .....	9
Эндпоинт .....	9
Тип запроса.....	9
Пример тела запроса .....	9
Описание доступных параметров: .....	9
5. Доступные модели .....	10
6. Пример промпта .....	11

Что такое промт.....	11
Структура промта .....	11
Пример промпта:.....	12
Полный запрос с промптом.....	12

# 1. Пример обращения в API для листинга моделей

## Эндпоинт

```
GET /v1/models
```

## Тип запроса

```
GET
```

## Пример запроса

```
curl -X GET https://api.gpt.mws.ru/v1/models \
-H "Authorization: Bearer YOUR_API_KEY"
```

## Ответ API

```
{
  "data": [
    {
      "id": "mws-gpt-alpha",
      "object": "model",
      "created": 1677610602,
      "owned_by": "openai"
    },
    {
      "id": "kodify-2.0",
      "object": "model",
      "created": 1677610602,
      "owned_by": "openai"
    },
    {
      "id": "cotype-preview-32k",
      "object": "model",
      "created": 1677610602,
      "owned_by": "openai"
    }
  ]
}
```

## 2. Пример обращения в Chat Completion (чат-комплишн)

### Для чего используется

Основные функции использование chat completions:

- Диалоговое взаимодействие
- Учет контекста
- Поддержка многозадачности

### Эндпоинт

POST /v1/chat/completions

### Тип запроса

POST

### Пример тела запроса

```
curl -X POST https://api.gpt.mws.ru/v1/chat/completions \
-H "Authorization: Bearer YOUR_API_KEY" \
-d '{"model": "mts-anya",
"messages": [
{"role": "system", "content": "Ты помощник" },
{"role": "user", "content": "Привет, как дела?" }
],
"temperature": 0.6
}'
```

### Ответ API

```
{"id": "chatcmpl-f6d4b004e9ba4e95b2cede3ca82040b4",
"created": 1739133562,
"model": "mts_anya",
"object": "chat.completion",
"system_fingerprint": null,
"choices": [
```

```
{"finish_reason":"stop","index":0,"message":  
{"content":"Дела отлично, я готов помочь тебе в любых вопросах или задачах. Как я могу тебе помочь?","role":"assistant","tool_calls":null,"function_call":null}},  
"usage":  
{"completion_tokens":28,"prompt_tokens":51,"total_tokens":79,"completion_tokens_details":  
null,"prompt_tokens_details":null},"service_tier":null,"prompt_logprobs":null}
```

## Описание доступных параметров:

- `model` - модель обработки текста
- `messages` - массив сообщений, где каждое сообщение содержит `role` (system, user, assistant) и `content`
- `temperature` - параметр случайности вывода (0.0 - детерминированный ответ, 1.0 - более случайный)
- `max_tokens` - максимальное количество возвращаемых токенов
- `n` - целое число - количество вариантов ответов, которые вы хотите получить
- `presence_penalty` - значение от -2.0 до 2.0 - штраф за повторение токенов, которые уже присутствуют в истории чата.
- `frequency_penalty` - значение от -2.0 до 2.0 - штраф за использование часто встречающихся токенов

## 3. Пример обращения в Completion

### Для чего используется

Основные функции использования completions:

- Генерация текста
- Завершение предложений

### Эндпоинт

POST /v1/completions

### Тип запроса

POST

### Пример тела запроса

```
curl -X POST https://api.gpt.mws.ru/v1/completions -H "Authorization: Bearer YOUR_API_KEY" \
-d '{
  "model": "mts-any",
  "prompt": "Что такое искусственный интеллект?",
  "temperature": 0.6,
  "max_tokens": 150,
  "top_p": 1,
  "frequency_penalty": 0,
  "presence_penalty": 0,
  "stop": ["\n"]
}'
```

### Ответ API

```
{
  "id": "cmpl-d7f6336cd211498192b3cd78c7f7d87e",
  "object": "text_completion",
  "created": 1739134432,
  "model": "mts-any",
  "choices": [
    {
      "stop_reason": null,
      "prompt_logprobs": null,
      "finish_reason": "length",
      "index": 0,
      "text": "ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ – в широком понимании – это программное обеспечение, которое может выполнять определенные задачи так же эффективно, как и человек. Это также могут"
```

быть роботы, которые могут решать сложные задачи, а также могут быть приложения, которые могут выполнять определенные задачи. ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ – это это способность компьютеров, программ и систем решать проблемы и принимать решения, которые требуют человеческого интеллекта и интуиции. В широком смысле, ИИ, "logprobs":null}], "usage": {"completion\_tokens":150, "prompt\_tokens":10, "total\_tokens":160, "completion\_tokens\_details": {"":null, "prompt\_tokens\_details":null}, "system\_fingerprint":null}

## Описание доступных параметров:

- `model` - выбор модели
- `prompt` - текст запроса
- `max_tokens` - максимальное количество возвращаемых токенов
- `temperature` - параметр случайности вывода (0.0 - детерминированный ответ, 1.0 - более случайный)
- `top_p` - альтернативный способ контроля случайности
- `presence_penalty` - значение от -2.0 до 2.0 - штраф за повторение токенов, которые уже присутствуют в истории чата.
- `frequency_penalty` - значение от -2.0 до 2.0 - штраф за использование часто встречающихся токенов
- `stop` - условие остановки генерации текста



## 4. Пример обращения в Embeddings

### Для чего используется

Основные функции использования embeddings:

- Классификация текста
- Семантическое представление текста
- Поиск и сравнение текста

### Эндпоинт

POST /v1/embeddings

### Тип запроса

POST

### Пример тела запроса

```
curl -X POST https://api.gpt.mws.ru/v1/embeddings -H "Authorization: Bearer YOUR_API_KEY" -d '{ "model": "bge-m3", "input": "как у тебя дела?"}'
```

### Описание доступных параметров:

- `model` - выбор модели эмбединга
- `input` - текст, который нужно преобразовать в эмбединг

## 5. Доступные модели

Модель
mws-gpt-alpha
kodify-2.0
cotype-preview-32k
bge-m3

### Токены:

- 2-3 русских слова  $\approx$  1 токен
- 3-4 английских слова  $\approx$  1 токен

## 6. Пример промпта

### Что такое промт

Промт – это входные данные для генеративной модели, предназначенные для управления результатами ее работы. Промт может состоять из текста, картинки, звука или любой другой информации.

Шаблон промта – это незаконченный промт с одной или несколькими переменными, которые заменяются на какую-то информацию для создания отдельного экземпляра промта.

Например, нужно решить задачу бинарной классификации твитов. Исходный шаблон промта будет выглядеть так:

Классифицируй твит как «положительный» или «отрицательный»: {TWEET}.

Если в шаблон подставить твит из датасета, получится экземпляр промта, который можно отправить генеративной модели для предсказания результата.

Качество выходных данных модели зависит от того, насколько хорошо вы сформулируете запрос и сколько информации предоставите. Промт может содержать основные элементы: инструкция или вопрос, а также дополнительные: контекст, входные данные и примеры. Используя эти элементы, вы можете более эффективно управлять моделью и улучшать качество результатов.

Начнем с простого примера промта:

#### Промт

Что такое замок?

#### Ответ

Замок – это сложное и многофункциональное сооружение, которое обычно служит для защиты территории или имущества. Замки могут быть разных типов: средневековые, современные, военные, дворцовые и другие. Они могут включать в себя различные элементы, такие как стены, башни, ворота, рвы и мосты.

В прошлом замки были важными стратегическими объектами, которые защищали от врагов и позволяли контролировать территорию. Сегодня замки часто являются историческими памятниками и туристическими достопримечательностями.

### Структура промта

Обращаясь к модели через API, вы можете структурировать промт с помощью ролей. API поддерживает следующие роли: **system**, **user** и **assistant**. В промт-режиме в консоли эти роли соответствуют полям **Инструкции**, **Запрос** и **Ответ**. Структурировать промт не обязательно.

Для простоты все дальнейшие примеры, если не указано иное, будут использовать только сообщение с ролью **user**.

Языковая модель отвечает последовательностью токенов, которые имеют смысл в контексте «замок». Однако результат может быть неожиданным или далеким от задачи, которую вы хотите выполнить. Например, мы хотели узнать про дверной замок. Этот простой пример подчеркивает необходимость предоставления большего контекста или инструкций о том, чего именно вы хотите добиться с помощью модели.

Попробуем дополнить промт:

### Промт

Что такое замок? (речь идет о дверных замках)

### Ответ

Замок – это механическое, электронное или комбинированное устройство фиксации двери в закрытом состоянии. Оно предотвращает несанкционированный доступ в помещение и обеспечивает безопасность.

Дверные замки различаются по конструкции, типу установки, уровню секретности и другим характеристикам. Они могут быть врезными, накладными, навесными и др.

Замки используются для защиты жилых домов, квартир, офисов, складов и других помещений.

## Пример промпта:

```
{
  "model": "gpt-4",
  "messages": [
    { "role": "system", "content": "Ты эксперт по искусственному интеллекту." },
    { "role": "user", "content": "Как работает машинное обучение?" }
  ]
}
```

## Полный запрос с промптом

```
curl -X POST https://api.gpt.mws.ru/v1/chat/completions \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -H "Content-Type: application/json" \
  -d '{
    "model": "gpt-4",
    "messages": [
      { "role": "system", "content": "Ты эксперт по AI." },
      { "role": "user", "content": "Как работает машинное обучение?" }
    ],
    "temperature": 0.7
  }'
```