

# API

## *Application Programming Interface*

Интерфейс программирования приложений

Интерфейс прикладного программирования

# Что такое API ?

API (интерфейс программирования приложений) — набор готовых классов, процедур, функций, структур и констант, предоставляемых сервисом (библиотекой) или операционной системой для использования во внешних программных продуктах.

Используется в качестве инструментального средства при написании всевозможных приложений.

# Как использовать API ?

Практика разработки API позволила веб-сообществам создать открытую архитектуру для совместного использования контента и данных. Тем самым содержимое, которое создается в одном месте, может динамически обновляться и размещаться во многих местах в вебе.

При использовании в контексте веб-разработки, API определяется как протокол передачи сообщений запроса (HTTP), наряду с определением структуры ответных сообщений, которые как правило, выдаются в формате XML или JSON.

# Поиск информации через API

Прежде всего находим нужный API:

<http://www.programmableweb.com/apis/directory>

The screenshot shows the ProgrammableWeb website interface. At the top, there is a navigation bar with the ProgrammableWeb logo, menu items for API NEWS and API DIRECTORY, and a search bar containing the text "Search over 14,630 APIs and much more". Below the navigation bar, there are several category tabs: API UNIVERSITY, RESEARCH, SPORTS, SECURITY, TRAVEL, and DESIGN. A prominent blue banner for "The Future of API Management" by Cloud Elements is displayed, featuring a rocket icon and a "Download White Paper" button. The main content area is titled "Browse the world's largest API repository" and includes a search bar with "Search Over 14,630 APIs" and a "SEARCH APIS" button. Below the search bar, there are filter options: "Filter APIs" with two dropdown menus, "By Category" and "By Protocols/Formats", and a checkbox for "Include Deprecated APIs". The bottom of the page shows the start of a table with columns for API Name, Description, Category, and Updated.

Google maps наиболее популярный.

# API: категории, протоколы, форматы

Search Over 14,712 APIs SEARCH APIS

Filter APIs

Linked Data × Web Site Management × HTTP × XML ×  Include Deprecated APIs

API Name	Description	Category	Updated
<a href="#">Ontotext S4 RDF graph database (DBaaS)</a>	S4 provides a fully-managed RDF graph database-as-a-service (DBaaS) based on the Onotext GraphDB database. Access to the database is via a simple RESTful API (OpenRDF API) and...	<a href="#">RDF</a>	10.17.2015
<a href="#">Website Worth</a>	Use the Website Worth API to check website and domain worth, find information like page rank, estimated visitors count, or search engine visibility. Website Worth uses HTTP methods with responses in...	<a href="#">Web Site Management</a>	08.18.2015
<a href="#">Whois Lookup</a>	Whoxy is a WHOIS search engine. The Whoxy API is a hosted web service that queries WHOIS registries for WHOIS registrars and parses returned data into well-	<a href="#">Web Site Management</a>	07.29.2015

# API: протоколы, форматы

Apache AVRO

Apache Thrift

Atom

Binary

CSV

FTP

GData

GeoJSON

HAL

HTTP

JSON

JSON-LD

JSONP

KML

MQTT

RDF

REST

RSS

REST

RSS

SMPP

SMTP

SOAP

Text

Unspecified

WSDL

XML

XMPP

YAML



1

API

географические карты

# API карты Google

<http://www.jose-gonzalez.org/using-google-maps-api-r/#.Ux2LEflgy3M>

Два API, представляющие интерес:

- Географическое положение
- Статические карты

Что дают эти API?

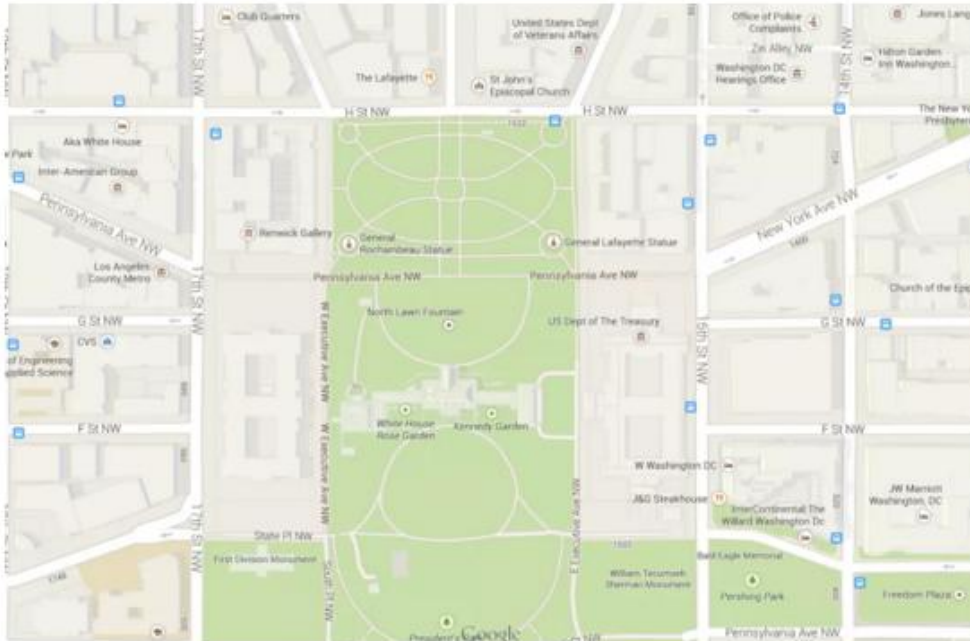
<https://developers.google.com/maps/documentation/geocoding/>  
<https://developers.google.com/maps/documentation/staticmaps/>



# API карты Google в RStudio

<http://www.jose-gonzalez.org/using-google-maps-api-r/#.Ux2LEflgy3M>

## Using Google maps API and R



This post shows how to use [Google Maps'](#) API with R making some tweaks to [this function](#). Combine the first part with `sapply` or [Plyr](#) and it becomes a very powerful tool in just a few lines of code. You can find a gist in RMarkdown with the code [here](#) or click below to continue reading.

```
1 ##### This script uses RCurl and RJSONIO to download data from Google's API:  
2 ##### Latitude, longitude, location type (see explanation at the end), formatted address
```

# Географическое положение

<https://developers.google.com/maps/documentati>

The screenshot shows the Google Developers website for the Google Maps API Geocoding service. The page has a blue header with the Google Developers logo, a search bar containing "Google Maps API геокод...", and a "ВОЙТИ" button. Below the header, there is a breadcrumb trail: "Продукты > Google Maps API > веб-сервисы > Google Maps API геокодирования". The main heading is "Google Maps API геокодирования" with a "РУКОВОДСТВА" link. A sidebar on the left contains links for "Руководство разработчика", "Получить ключ", "Ограничения использования", "Google Maps Web Services" (with sub-links for "обзор" and "Клиентская библиотека"), and "Другие API" (with sub-links for "Направления API", "API матрицы расстояний", "Elevation API", "Geolocation API", and "Места API веб-сервиса"). The main content area has the heading "Google Maps API геокодирования" and a "Содержание" section with a dropdown arrow. The content list includes "Что такое геокодирование?", "Прежде чем вы начнете", "Google Maps API геокодирования Формат запроса", and "Геокодирование (широта / долгота Поиск)". A blue callout box at the bottom contains a star icon and the text: "Эта услуга также доступна как часть API Google Maps JavaScript, или клиентские библиотеки Java и Python."

Необходимые параметры: `address [place]` и `sensor [=false]`

# Пример запроса геокоординат

## Запрос:

- `https://maps.googleapis.com/maps/api/geocode/json?`
- `address=Kremlin,Moscow`
- `sensor=false`

## Параметры разделяются '&'

`https://maps.googleapis.com/maps/api/geocode/json?address=Kremlin,Moscow&sensor=false`

Какое из полей JSON нас интересует?

# Результат запроса:

<https://maps.googleapis.com/maps/api/geocode/json?address=Kremlin,Moscow&sensor=false>

```
{
  "results" : [
    {
      "address_components" : [
        {
          "long_name" : "Московский Кремль",
          "short_name" : "Московский Кремль",
          "types" : [ "point_of_interest", "establishment" ]
        },
        {
          "long_name" : "Центральный административный округ",
          "short_name" : "Центральный административный округ",
          "types" : [ "sublocality_level_1", "sublocality", "political" ]
        },
        {
          "long_name" : "Москва",
          "short_name" : "Москва",
          "types" : [ "locality", "political" ]
        },
        {
          "long_name" : "город Москва",
          "short_name" : "г. Москва",
          "types" : [ "administrative_area_level_2", "political" ]
        },
        {
          "long_name" : "город Москва",
          "short_name" : "г. Москва",
          "types" : [ "administrative_area_level_1", "political" ]
        }
      ],
      "formatted_address" : "Московский Кремль, Москва, Россия, 103073",
      "geometry" : {
        "location" : {
          "lat" : 55.7520233,
          "lng" : 37.6174994
        },
        "location_type" : "APPROXIMATE",
        "viewport" : {
          "northeast" : {
            "lat" : 55.75337228029149,
            "lng" : 37.6188483802915
          },
          "southwest" : {
            "lat" : 55.75067431970849,
            "lng" : 37.6161504197085
          }
        }
      },
      "place_id" : "ChIJc-UVs1BKtUYRaC6bPVq_hqg",
      "types" : [ "museum", "point_of_interest", "establishment" ]
    },
    {
      "long_name" : "Россия",
      "short_name" : "RU",
      "types" : [ "country", "political" ]
    },
    {
      "long_name" : "103073",
      "short_name" : "103073",
      "types" : [ "postal_code" ]
    }
  ],
  "status" : "OK"
}
```

# Результат запроса:

<https://maps.googleapis.com/maps/api/geocode/json?address=Kremlin,Moscow&sensor=false>

"results" [

Какое из полей JSON нас интересует?

```
{
  "address_components": [
    {
      "long_name": "Московский Кремль",
      "short_name": "Московский Кремль",
      "types": [ "point_of_interest", "establishment" ]
    },
    {
      "long_name": "Центральный административный округ",
      "short_name": "Центральный административный округ",
      "types": [ "sublocality_level_1", "sublocality", "political" ]
    },
    {
      "long_name": "Москва",
      "short_name": "Москва",
      "types": [ "locality", "political" ]
    },
    {
      "long_name": "город Москва",
      "short_name": "г. Москва",
      "types": [ "administrative_area_level_2", "political" ]
    },
    {
      "long_name": "город Москва",
      "short_name": "г. Москва",
      "types": [ "administrative_area_level_1", "political" ]
    }
  ],
  "geometry": {
    "location": {
      "lat": 55.7520233,
      "lng": 37.6174994
    }
  },
  "formatted_address": "Московский Кремль, Москва, Россия, 103073",
  "place_id": "ChIJc-UVs1BKtUYRaC6bPVq_hqg",
  "types": [ "museum", "point_of_interest", "establishment" ]
},
{
  "status": "OK"
}
```

```
{
  "long_name": "Россия",
  "short_name": "RU",
  "types": [ "country", "political" ]
},
{
  "long_name": "103073",
  "short_name": "103073",
  "types": [ "postal_code" ]
}
],
"formatted_address": "Московский Кремль, Москва, Россия, 103073",
"geometry": {
  "location": {
    "lat": 55.7520233,
    "lng": 37.6174994
  }
},
"place_id": "ChIJc-UVs1BKtUYRaC6bPVq_hqg",
"types": [ "museum", "point_of_interest", "establishment" ]
},
{
  "status": "OK"
}
```

results[[1]]\$geometry\$location["lat"]  
results[[1]]\$geometry\$location["lng"]

# Запрос геокоординат в RStudio

## Напишем функцию `getUrl()`:

```
getUrl <- function(address,sensor = "false") {  
  root <- "http://maps.google.com/maps/api/geocode/json?"  
  u <- paste0(root,"address=", address, "&sensor=false")  
  return(URLEncode(u))  
}  
getUrl("Kremlin, Moscow")
```

## Используем функцию `getUrl()`:

```
require(RJSONIO)  
target <- getUrl("Kremlin, Moscow")  
dat <- fromJSON(target)  
latitude <- dat$results[[1]]$geometry$location["lat"]  
longitude <- dat$results[[1]]$geometry$location["lng"]
```

# Получение карты

<https://developers.google.com/maps/documentation/staticmaps/>

## Формирование URL:

base="http://maps.googleapis.com/maps/api/staticmap?"

center= "latitude (55.75), longitude (37.62)"

zoom (1 - 18)

maptype="hybrid"

suffix = "&size=800x800&sensor=false&format=png"

# Формирование URL карты в RStudio

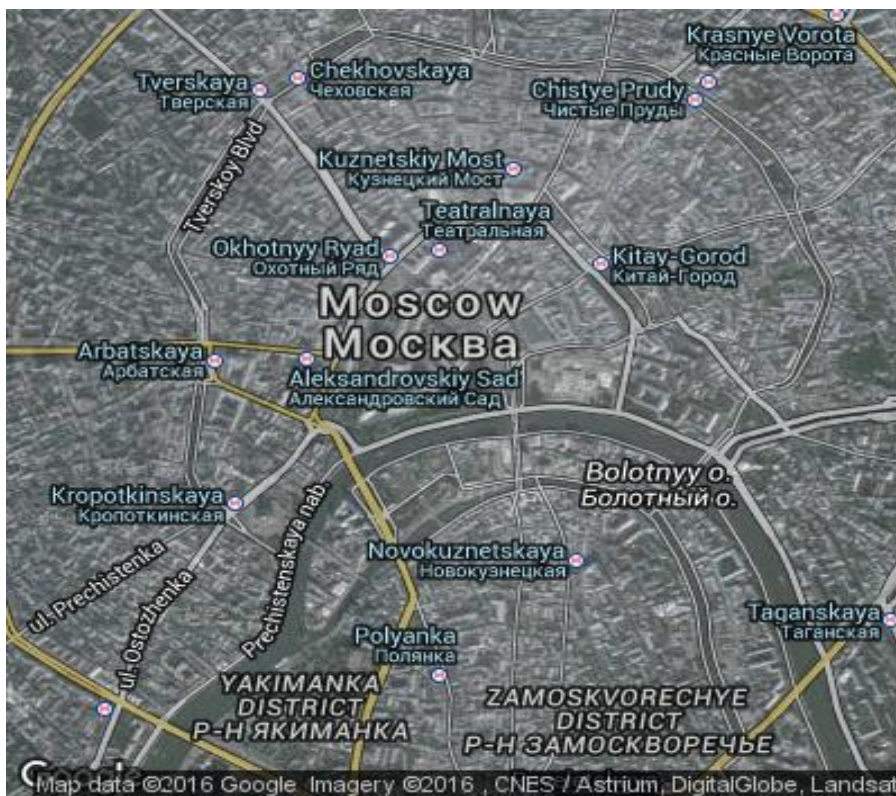
```
base="http://maps.googleapis.com/maps/api/staticmap?center="
latitude=55.75
longitude=37.62
zoom=13
maptype="hybrid"
suffix ="&size=400x400&sensor=false&format=png"

target <- paste0(base,latitude,",",longitude,
                 "&zoom=",zoom,"&maptype=",maptype,suffix)
```



# Загрузка карты lat=55,75 lon=37,62

`download.file(target, "test13.png", mode = "wb")`



**zoom=13**

# Загрузка карты lat=55,75 lon=37,62

`download.file(target, "test15.png", mode = "wb")`



**zoom=15**

# Использование карт с API Yandex

Yandex карты имеют очень похожий синтаксис с картами Google.

Чтобы построить сценарии с API для карт Yandex, нужно использовать информацию из:

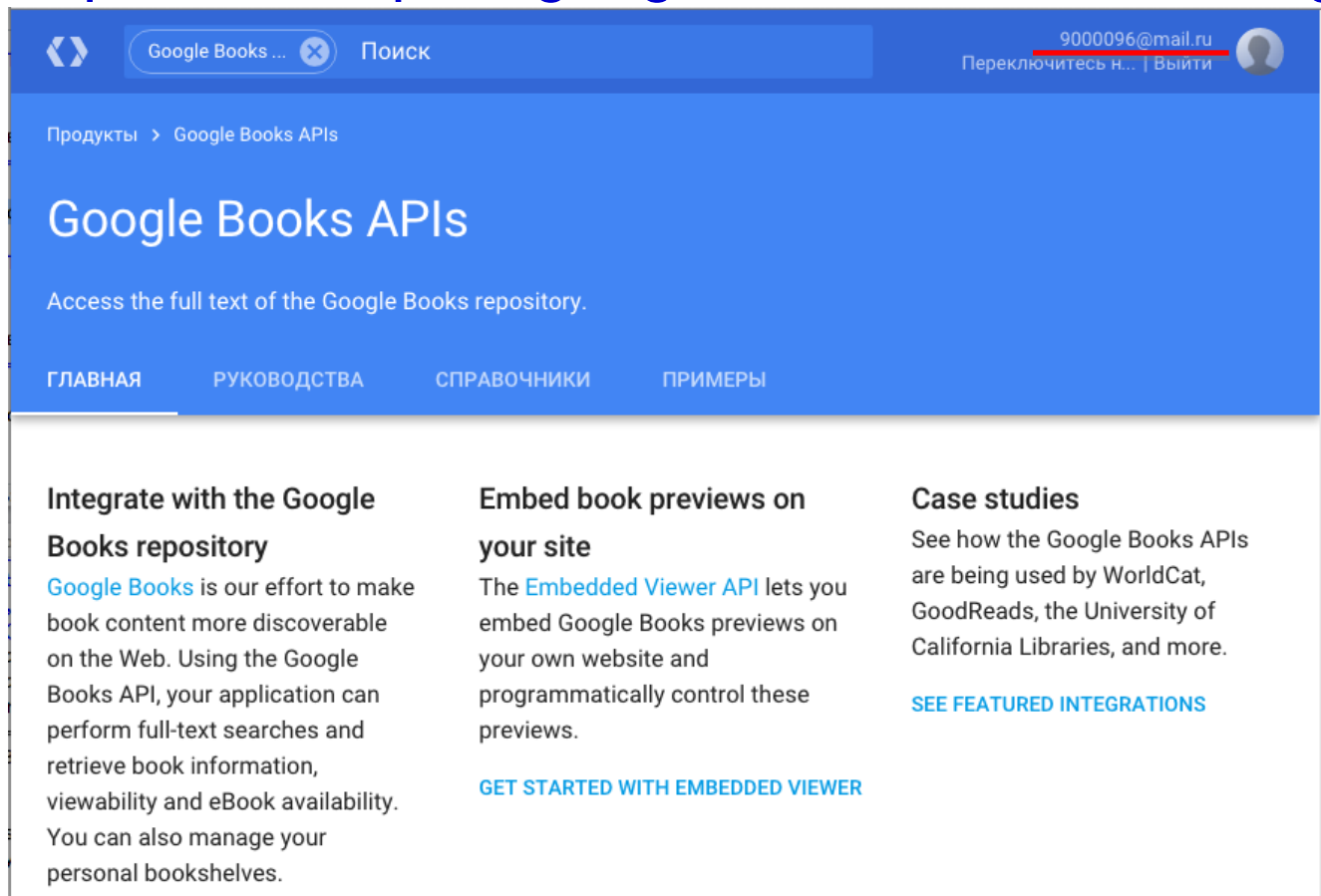
[http://api.yandex.com/maps/doc/staticapi/1.x/dg/concepts/input\\_params.xml](http://api.yandex.com/maps/doc/staticapi/1.x/dg/concepts/input_params.xml)



# 2 API Google книги

# Использование API Google книги

<https://developers.google.com/books/docs/v1/getting>



The screenshot shows the Google Books APIs developer page. At the top, there is a search bar with the text "Google Books ..." and a search icon. To the right of the search bar, there is a user profile section with the email address "9000096@mail.ru" and the text "Переключитесь н... | Выйти". Below the search bar, there is a navigation menu with the following items: "Продукты > Google Books APIs", "Google Books APIs", "Access the full text of the Google Books repository.", "ГЛАВНАЯ", "РУКОВОДСТВА", "СПРАВОЧНИКИ", and "ПРИМЕРЫ". The main content area is divided into three columns. The first column is titled "Integrate with the Google Books repository" and contains the text: "Google Books is our effort to make book content more discoverable on the Web. Using the Google Books API, your application can perform full-text searches and retrieve book information, viewability and eBook availability. You can also manage your personal bookshelves." The second column is titled "Embed book previews on your site" and contains the text: "The Embedded Viewer API lets you embed Google Books previews on your own website and programmatically control these previews." Below this text is a link: "GET STARTED WITH EMBEDDED VIEWER". The third column is titled "Case studies" and contains the text: "See how the Google Books APIs are being used by WorldCat, GoodReads, the University of California Libraries, and more." Below this text is a link: "SEE FEATURED INTEGRATIONS".

# Пример использования API Google книги

<https://www.googleapis.com/books/v1/volumes?q=harry+potter&callback=handleResponse>

```
"title": "Гарри Поттер и узник Азкабана",  
"authors": [  
  "J. K. Rowling"  
],  
"publisher": "Rosman",  
"publishedDate": "2002",  
"description": "Открой эту книгу - и ты окунешься в мир чародейства и волшебства, где оживают мифические чудовища, где школьный учитель - оборотень и где даже время можно повернуть вспять",  
  
"title": "Гарри Поттер и Философский Камень",  
"subtitle": "",  
"authors": [  
  "Джоан Роулинг"  
],  
"publisher": "Lires",  
"publishedDate": "2015-05-10",  
"description": "Одиннадцатилетний мальчик-сирота Гарри Поттер живет в семье своей тетки и даже не подозревает, что он - настоящий волшебник. Но однажды прилетает сова с письмом для него, и жизнь Гарри Поттера изменяется навсегда. Он узнает, что зачислен в Школу чародейства и волшебства, выясняет правду о загадочной смерти своих родителей, а в результате ему удастся раскрыть секрет философского камня."
```

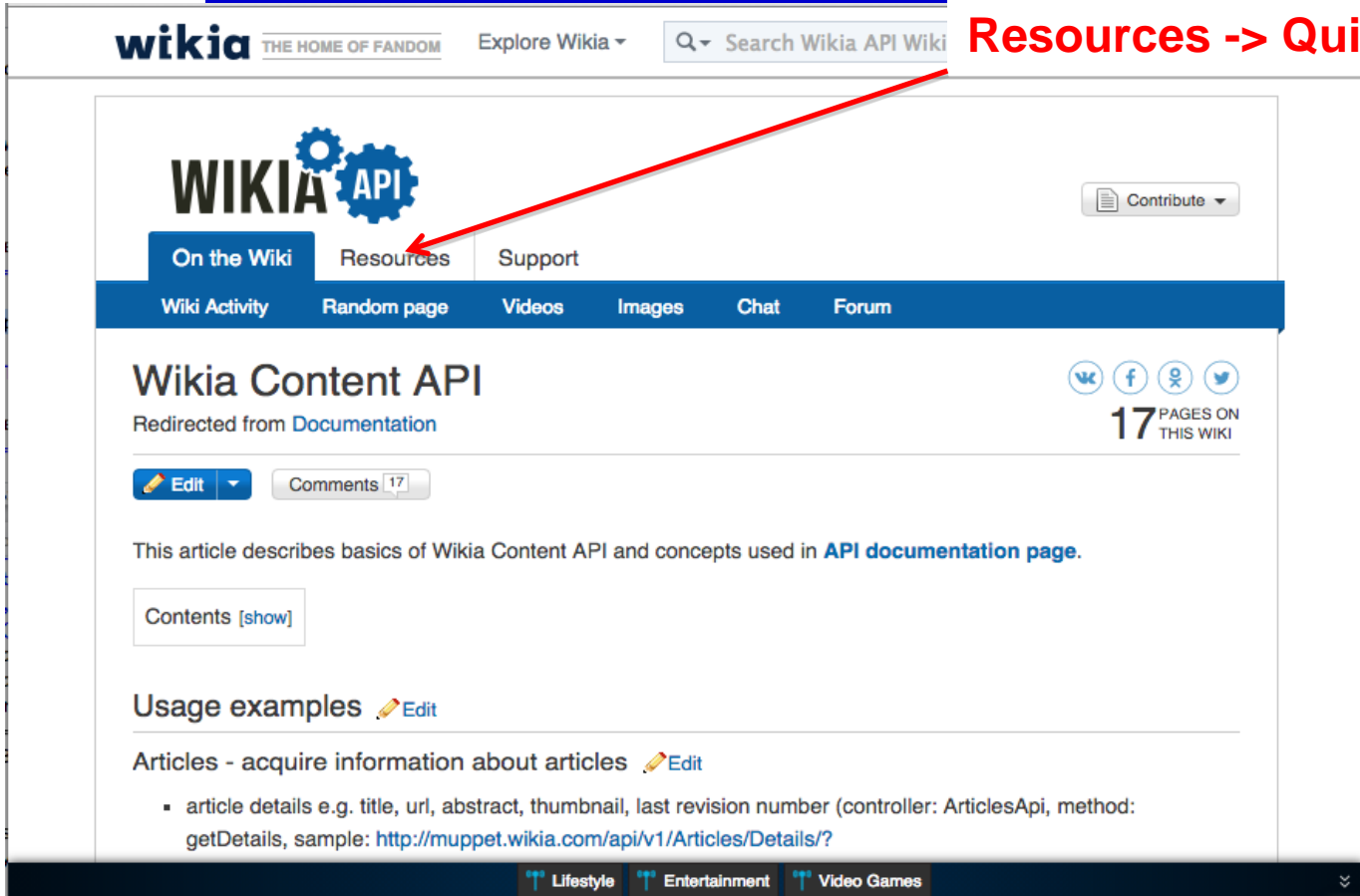


# 3 API Wiki

# Использование API Wiki

<http://api.wikia.com/wiki/Documentation>

Resources -> Quick Start



The screenshot shows the Wikia API Documentation page. At the top, the Wikia logo is followed by the tagline 'THE HOME OF FANDOM' and a search bar. The main navigation bar includes 'On the Wiki', 'Resources', and 'Support'. Below this, a secondary navigation bar contains 'Wiki Activity', 'Random page', 'Videos', 'Images', 'Chat', and 'Forum'. The page title is 'Wikia Content API', with a note that it is 'Redirected from Documentation'. It indicates there are 17 pages on this wiki. The page content includes an 'Edit' button, a 'Comments' count of 17, and a description: 'This article describes basics of Wikia Content API and concepts used in API documentation page.' There is a 'Contents [show]' button and a section for 'Usage examples' with an 'Edit' button. Below that, it lists 'Articles - acquire information about articles' with an 'Edit' button and a list item: 'article details e.g. title, url, abstract, thumbnail, last revision number (controller: ArticlesApi, method: getDetails, sample: <http://muppet.wikia.com/api/v1/Articles/Details/>)'. At the bottom, there are category tabs for 'Lifestyle', 'Entertainment', and 'Video Games'.



# Использование API Wiki

The screenshot shows the Wikia API Wiki homepage. At the top, the Wikia logo is followed by the tagline "THE HOME OF FANDOM". To the right, there is a search bar with the text "Search Wikia API Wiki..." and a "Sign In" button. Below the header, the main content area features the "WIKIA API" logo and a "Contribute" button. A navigation bar includes "On the Wiki", "Resources", and "Support". Below this, a secondary navigation bar lists "Wiki Activity", "Random page", "Videos", "Images", "Chat", and "Forum". The "Quick Start" section contains a three-step process: 1. Read our API docs (with a link to Terms of Use), 2. Explore API (with links to interactive documentation and API usage examples), and 3. Build, test, and go! (with instructions to share with the community). To the right of the quick start steps, there are social media icons for Wikia, Facebook, and Twitter, along with a counter showing "17 PAGES ON THIS WIKI". At the bottom, there is a "What is Wikia?" section and a footer with links for "Lifestyle", "Entertainment", and "Video Games".

wikia THE HOME OF FANDOM Explore Wikia Search Wikia API Wiki... Sign In

WIKIA API Contribute

On the Wiki Resources Support

Wiki Activity Random page Videos Images Chat Forum

Quick Start

17 PAGES ON THIS WIKI

View source Comments 9

1 Read our API docs  
Read our [Terms of Use](#).

2 Explore API  
Use our [interactive documentation](#) and read [API usage examples](#)

3 Build, test, and go!  
Build and test (over and over again). Then [share with the community](#).

What is Wikia?

Lifestyle Entertainment Video Games

# Пример использования API Wiki

## \* action=query \*

Query API module allows applications to get needed pieces of data from the MediaWiki databases, and is loosely based on the old query.php interface.  
All data modifications will first have to use query to acquire a token to prevent abuse from malicious sites

This module requires read rights

### Parameters:

- titles - A list of titles to work on
- pageids - A list of page IDs to work on  
Maximum number of values 50 (500 for bots)
- revids - A list of revision IDs to work on  
Maximum number of values 50 (500 for bots)
- prop - Which properties to get for the titles/revisions/pageids. Module help is available below  
Values (separate with '|'): info, revisions, links, iwlinks, langlinks, images, imageinfo, stashimageinfo, templates, categories, extlinks, categoryinfo, duplicatefiles, pageprops, wklasteditors, globalusage, infobox
- list - Which lists to get. Module help is available below  
Values (separate with '|'): allimages, allpages, alllinks, allcategories, allusers, backlinks, blocks, categorymembers, deletedrevs, embeddedin, filearchive, imageusage, iwbacklinks, langbacklinks, logevents, recentchanges, search, tags, usercontribs, watchlist, watchlistraw, exturlusage, users, random, protectedtitles, querypage, wkdomains, wkpppages, wkvoteart, wkaccessart, wkeditpage, wkedituser, wkmstvisit, checkuser, checkuserlog, abuselog, abusefilters, categoryintersection, unconvertedinfoboxes, allinfoboxes
- meta - Which metadata to get about the site. Module help is available below  
Values (separate with '|'): siteinfo, userinfo, allmessages
- generator - Use the output of a list as the input for other prop/list/meta items  
NOTE: generator parameter names must be prefixed with a 'g', see examples  
One value: links, images, templates, categories, duplicatefiles, allimages, allpages, alllinks, allcategories, backlinks, categorymembers, embeddedin, imageusage, iwbacklinks, langbacklinks, recentchanges, search, watchlist, watchlistraw, exturlusage, random, protectedtitles, querypage, categoryintersection
- redirects - Automatically resolve redirects
- converttitles - Convert titles to other variants if necessary. Only works if the wiki's content language supports variant conversion.  
Languages that support variant conversion include gan, iu, kk, ku, shi, sr, tg, zh
- indexpageids - Include an additional pageids section listing all returned page IDs
- export - Export the current revisions of all given or generated pages
- exportnowrap - Return the export XML without wrapping it in an XML result (same format as Special:Export). Can only be used with export
- iwurl - Whether to get the full URL if the title is an interwiki link

### Examples:

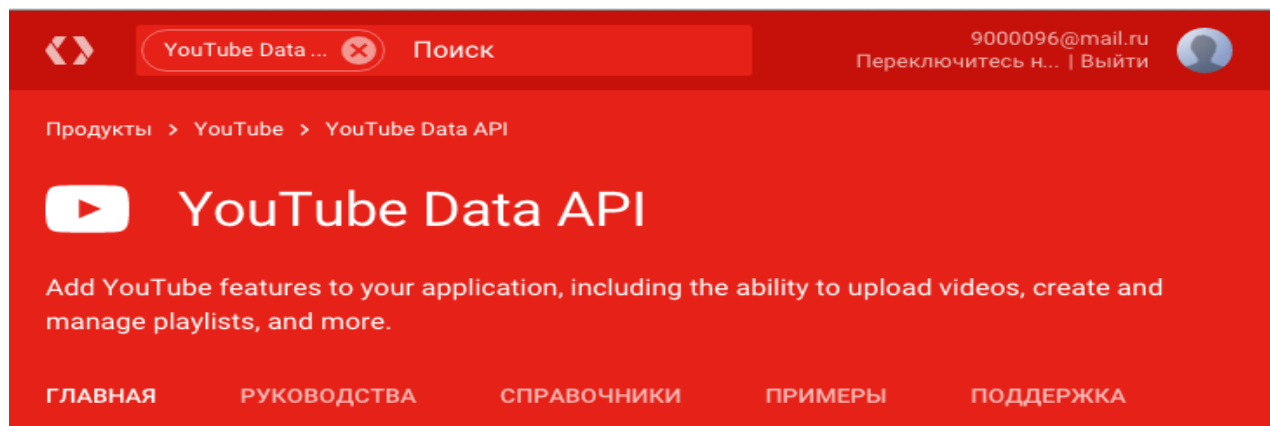
[api.php?action=query&prop=revisions&meta=siteinfo&titles=Main%20Page&rvprop=user|comment](#)  
[api.php?action=query&generator=allpages&gabbr=API/&prop=revisions](#)



# 4 API YouTube

# Использование API YouTube

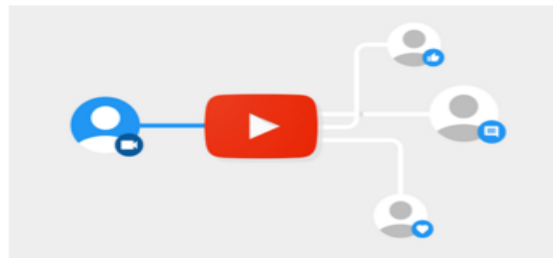
<https://developers.google.com/youtube/v3/>




## Add YouTube functionality to your app

### Add YouTube functionality to your site

With the YouTube Data API, you can add a variety of YouTube features to your application. Use the API to upload videos, manage playlists and subscriptions, update channel settings, and more.





# 5 API

## Требующие установки

# API требующие установки

Гнилые помидоры

<http://developer.rottentomatoes.com/docs>

Переводчик (более 50 языков)

<http://blogs.msdn.com/b/translation/p/gettingstarted1.aspx>

Погода

<http://www.worldweatheronline.com/free-weather-feed.aspx>

<https://developer.forecast.io/>

Музыка

<http://www.last.fm/api>

<http://rcrastinate.blogspot.co.uk/2013/03/peace-through-music-country-clustering.html>

Фото/Видео

<https://www.flickr.com/services/api/>

# API Rotten Tomatoes

<http://developer.rottentomatoes.com/docs>

The screenshot shows the Rotten Tomatoes API documentation page. The header is green with the Rotten Tomatoes logo (by Flixster) on the left, a search bar, and navigation links for Home, Documentation, Brand Guidelines, and Forum. A 'Sign In' link is in the top right. A red 'MASHERY MADE' badge is also present. The main content area is white with a light blue border. On the right side of the main content, there is a 'JSON API =>' link. The page is divided into two columns. The left column contains the main content, and the right column contains a 'Docs Navigation' sidebar with a list of links.

**API Overview**

The Rotten Tomatoes API is RESTful web service that was designed to be easy to explore and use. The base URI to access all resources is <http://api.rottentomatoes.com/api/public/v1.0>. It is our hope that through the base URI, a developer getting started with our API will be able to reach and manipulate our APIs without reading through multiple pages of documentation. This is accomplished by linking related resources and providing instructions on how to use each representation (link templates) in the response itself.

**For a dynamic view of our documentation, try our IO Docs!** IO Docs allows you to make live calls to our APIs while getting a stronger understanding for all of the resources and parameters involved.

**To access our Rotten Tomatoes design assets:**

- Fill out this [form](#) to gain access
- Enter your submitted email address into the following URL to access the page with the assets: [http://www.rottentomatoes.com/help\\_desk/assets?email=\[your\\_email\\_address\]](http://www.rottentomatoes.com/help_desk/assets?email=[your_email_address])

**A Quick Dive into the API**

Let's take a quick exploratory dive into the API and see this thing in action. To start out on our journey, simply register for a Rotten Tomatoes API key, select your favorite browser and navigate to the "homepage" of our API with your API key appended at the end. For example: [http://api.rottentomatoes.com/api/public/v1.0.json?apikey=\[your\\_api\\_key\]](http://api.rottentomatoes.com/api/public/v1.0.json?apikey=[your_api_key]) Replace [your\_api\_key] with your own API key. Notice the extension '.json' at the end of the URL. We currently support responses in JSON, with future plans to support other data formats. In the future, for example, if we supported xml, you would be able to hit [http://api.rottentomatoes.com/api/public/v1.0.xml?apikey=\[your\\_api\\_key\]](http://api.rottentomatoes.com/api/public/v1.0.xml?apikey=[your_api_key]).

You should get a response like below:

**JSON API =>**

**Docs Navigation**

- API Overview
- JSON API
- API Homepage - v1.0.json
- Movies Search
- Lists Directory
- Movie Lists Directory
- Box Office Movies
- In Theaters Movies
- Opening Movies
- Upcoming Movies
- DVD Lists Directory
- Top Rentals
- Current Release DVDs
- New Release DVDs
- Upcoming DVDs
- Movie Info
- Movie Cast
- Movie Reviews
- Movie Similar
- Movie Alias
- Example Code

Assets and Brand Guidelines