

UNIVERSITY OF
Southampton

Documenting REST

COMP3220 Web Infrastructure

Dr Nicholas Gibbins – nmg@ecs.soton.ac.uk

Documentation

What are the key aspects of a RESTful interface?

How should we document each of these?

What does a developer need to know to use our service?

Identification

URIs

PayPal Developer | Docs | APIs | Support | Search

← Back to APIs

REST API Reference

- Billing Agreements API
- Billing Plans API
- Identity API
- Invoicing API
- Limits Resolutions API
- Payment Experience API
- Payments API**
 - Payments
 - Create a payment
 - Execute approved PayPal payment
 - Show payment details
 - Update payment**
 - List payments
 - Sale transactions
 - Refunds
 - Authorizations
 - Captures

Update payment

PATCH `/v1/payments/payment/{payment_id}`

Partially updates a payment, by ID. You cannot update a payment after the payment is executed.

Parameters

Pass the `payment_id` in the URI.

payment_id *path* string

The ID of the payment to update.

dev.twitter.com

POST statuses/update — Twitter Develo...

Payments - PayPal Developer | Developer Resou...

Resource URL

`https://api.twitter.com/1.1/statuses/update.json`

Resource Information

Response formats	JSON
Requires authentication?	Yes (user context only)
Rate limited?	Yes

developer.wordpress.com

Documentation → REST API Resources → posts

GET `/sites/$site/posts/$post_ID`

Get a single post (by ID).

Resource Information

Method	GET
URL	https://public-api.wordpress.com/rest/v1.1/sites/\$site/posts/\$post_ID
Requires authentication?	No

Method Parameters

Parameter	Type	Description
<code>\$site</code>	(int string)	Site ID or domain
<code>\$post_ID</code>	(int)	The post ID

On this page:

- Resource URL
- Method Parameters
- Query Parameters
- Response Parameters
- Resource Errors
- Example

Authentication

Users

Sites

Posts

Get a list of matching posts.

Get a single post (by ID).

Edit a post.

Get a single post (by slug).

Create a post.

description

Def

text of your status update, locally up to 140 characters. encode as necessary. t.co wrapping may affect character counts. There are

URI Parameters

The image displays two overlapping browser windows illustrating URI parameters in REST APIs.

Top Window: PayPal Developer API - Update payment

Method: **PATCH** /v1/payments/payment/{**payment_id**}

Partially updates a payment, by ID. You cannot update a payment after the payment is executed.

Parameters

Pass the `payment_id` in the URI.

payment_id	<i>path</i> string
-------------------	--------------------

The ID of the payment to update.

SDK samples: [JAVA](#)

Bottom Window: WordPress REST API - posts

Method: **GET** /sites/{**\$site**}/posts/{**\$post_ID**}

Get a single post (by ID).

Resource Information

Method	GET
URL	https://public-api.wordpress.com/rest/v1.1/sites/{site}/posts/{post_ID}
Requires authentication?	No

Method Parameters

Parameter	Type	Description
\$site	(int string)	Site ID or domain
\$post_ID	(int)	The post ID

On this page:

- Resource URL
- Method Parameters
- Query Parameters
- Response Parameters
- Resource Errors
- Example

Authentication

- Users**
- Sites**
- Posts**

Get a list of matching posts.

Get a single post (by ID).

- Edit a post.
- Get a single post (by slug).
- Create a post.

Interaction

Methods

developer.paypal.com

REST API Reference

Update payment

PATCH /v1/payments/payment/{payment_id}

Partially updates a payment, by ID. You cannot update a payment after the payment is executed.

Parameters

Pass the `payment_id` in the URI.

`payment_id` *path* string

The ID of the payment to update.

dev.twitter.com

REST APIs / Reference Documentation / POST statuses/update

POST statuses/update

Updates the authenticating user's current status, also known as Tweeting.

For each update attempt, the update text is compared with the authenticating user's recent Tweets. Any attempt that would result in duplication will be blocked, resulting in a 403 error. A user cannot submit the same status twice in a row.

While not rate limited by the API, a user is limited in the number of Tweets they can create at a time. If the number of updates posted by the user reaches the current allowed limit this method will return an HTTP 403 error.

About Geo

- Any geo-tagging parameters in the update will be ignored if `geo_enabled` for the user is false (this is the default setting for all users, unless the user has enabled geolocation in their settings)
- The number of digits after the decimal separator passed to `lat` (up to 8) is tracked so that when the `lat` is returned in a status object it will have the same number of digits after the decimal separator.
- Use a decimal point as the separator (and not a decimal comma) for the latitude and the longitude - usage of a decimal comma will cause the geo-tagged portion of the status update to be dropped.
- For JSON, the response mostly uses conventions described in GeoJSON. However, the `geo` object coordinates that Twitter renders are **reversed** from the GeoJSON specification. GeoJSON specifies a longitude then a latitude, whereas Twitter represents it as a latitude then a longitude: `"geo": { "type": "Point", "coordinates": [37.78217, -122.40062] }`

developer.wordpress.com

Documentation → REST API Resources → posts

GET /sites/\$site/posts/\$post_ID

Get a single post (by ID).

Resource Information

Method	GET
URL	https://public-api.wordpress.com/rest/v1.1/sites/\$site/posts/\$post_ID
Requires authentication?	No

Method Parameters

Parameter	Type	Description
<code>\$site</code>	(int string)	Site ID or domain
<code>\$post_ID</code>	(int)	The post ID

On this page:

- Resource URL
- Method Parameters
- Query Parameters
- Response Parameters
- Resource Errors
- Example

Authentication

- Users
- Sites
- Posts**

Get a list of matching posts.

Get a single post (by ID).

Edit a post.

Get a single post (by slug).

Create a post.

Status Codes

Response

Returns the HTTP status of 204 if the call is successful.

Sample Response

```
{
  "id": "PAY-5YK922393D847794YKER71",
  "intent": "authorize",
  "create_time": "2015-04-24T14:26",
  "payer": {
    "payer_info": {
      "payer_id": "A123456789",
      "email": "anna@paypal.com",
      "name": "Anna",
      "phone": "805-521-1234",
      "address": {
        "line1": "123 Main St",
        "line2": "Apt 456",
        "city": "San Jose",
        "state": "CA",
        "zip": "95131"
      },
      "country_code": "DE",
      "currency": "EUR",
      "account_type": "VERIFIED"
    }
  }
}
```

Reference Documentation / POST statuses/update

POST statuses/update

Updates the authenticating user's current status, also known as Tweeting.

For each update attempt, the update text is compared with the authenticating user's recent Tweets. Any attempt that would result in duplication will be blocked, resulting in a 403 error. A user cannot submit the same status twice in a row.

While not rate limited by the API, a user is limited in the number of Tweets they can create at a time. If the number of updates posted by the user reaches the current allowed limit this method will return an HTTP 403 error.

About Geo

metadata	(array)	protected meta keys are available for authenticated requests with access. meta keys can be made available with rest_api_allowed_public_metadata filter.
meta	(object)	API result meta data
capabilities	(object)	List of post-specific permissions for tweet_publish_post, edit_post, delete_post
revisions	(array)	List of post revision IDs. Only available retrieved with context=edit.
other_urls	(object)	List of URLs for this post. Permalink and suggestions.

Resource Errors

These are the possible errors returned by this endpoint.

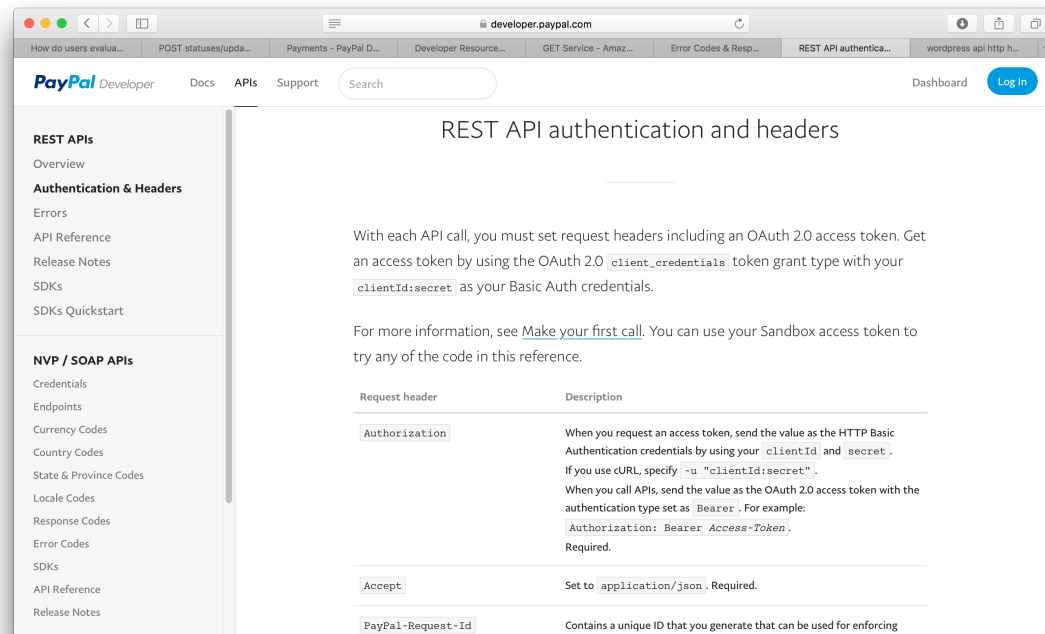
HTTP Code	Error Identifier	Error Message
400	invalid_field	Invalid API FIELD
403	unauthorized	User cannot access this private blog.
403	unauthorized	User cannot access this restricted blog
400	invalid_context	Invalid API CONTEXT
404	unknown_post	Unknown post

Headers

Mostly for Authorisation

- OAuth 2.0, etc

Consider how the various `Accept-*` headers might be used.



The screenshot shows the PayPal Developer REST API authentication and headers page. The page title is "REST API authentication and headers". The main content area contains the following text:

With each API call, you must set request headers including an OAuth 2.0 access token. Get an access token by using the OAuth 2.0 `client_credentials` token grant type with your `clientId:secret` as your Basic Auth credentials.

For more information, see [Make your first call](#). You can use your Sandbox access token to try any of the code in this reference.

Request header	Description
<code>Authorization</code>	When you request an access token, send the value as the HTTP Basic Authentication credentials by using your <code>clientId</code> and <code>secret</code> . If you use cURL, specify <code>-u "clientId:secret"</code> . When you call APIs, send the value as the OAuth 2.0 access token with the authentication type set as <code>Bearer</code> . For example: <code>Authorization: Bearer Access-Token</code> . Required.
<code>Accept</code>	Set to <code>application/json</code> . Required.
<code>PayPal-Request-Id</code>	Contains a unique ID that you generate that can be used for enforcing

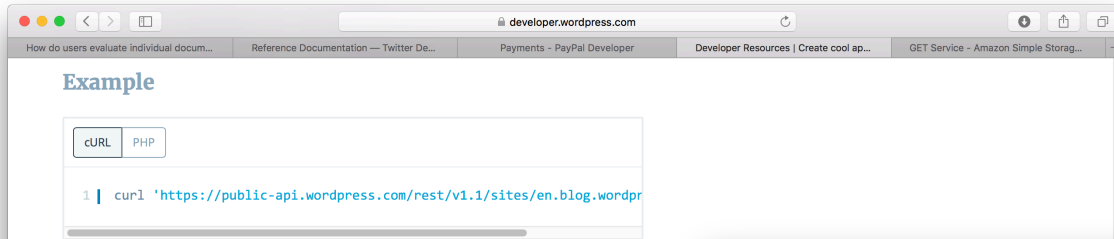
Representation

Representation

Response Parameters

Parameter	Type	Description
ID	(int)	The post ID.
site_ID	(int)	The site ID.
author	(object)	The author of the post.
date	(iso 8601 datetime)	The post's creation time.
modified	(iso 8601 datetime)	The post's most recent update time.
title	(html)	context dependent.
URL	(url)	The full permalink URL to the post.
short_URL	(url)	The wp.me short URL.
content	(html)	context dependent.
excerpt	(html)	context dependent.
slug	(string)	The name (slug) for the post, used in URLs.
guid	(string)	The GUID for the post.
		<p>publish: The post is published.</p> <p>draft: The post is saved as a draft.</p> <p>pending:</p>

Examples

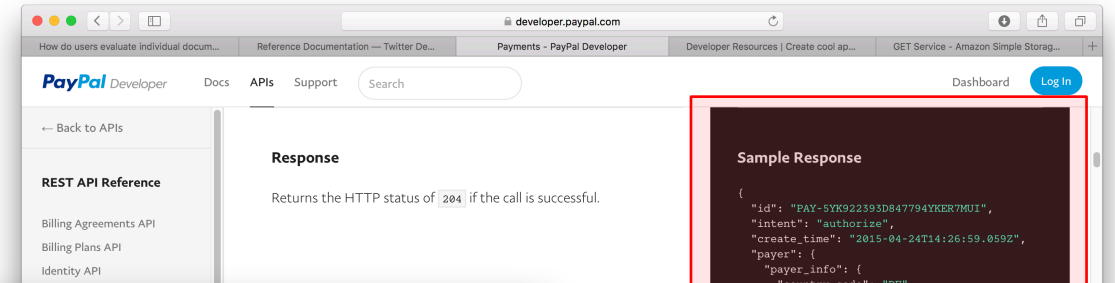


Response

```

1 {
2   "ID": 7,
3   "site_ID": 3584907,
4   "author": {
5     "ID": 5,
6     "login": "matt",
7     "email": false,
8     "name": "Matt",
9     "first_name": "Matt",
10    "last_name": "Mullenweg",
11    "nice_name": "matt",
12    "URL": "http://matt.wordpress.com",
13    "avatar_URL": "https://1.gravatar.com/avatar/767fc9c11",
14    "profile_URL": "http://en.gravatar.com/matt",
15    "site_ID": 492382
16  },
17  "date": "2005-09-26T21:43:58+00:00",
18  "modified": "2005-09-26T21:43:58+00:00",
19  "title": "So should we be, like, blogging and stuff too?",
20  "URL": "http://en.blog.wordpress.com/2005/09/26/blogg-21",
21  "short_URL": "http://wp.me/pf285-7",
22  "content": "cp>It is absolutely is criminal we don't have an of",
23  "excerpt": "cp>It is absolutely is criminal we don't have an of",
24  "slug": "blogging-and-stuff",

```

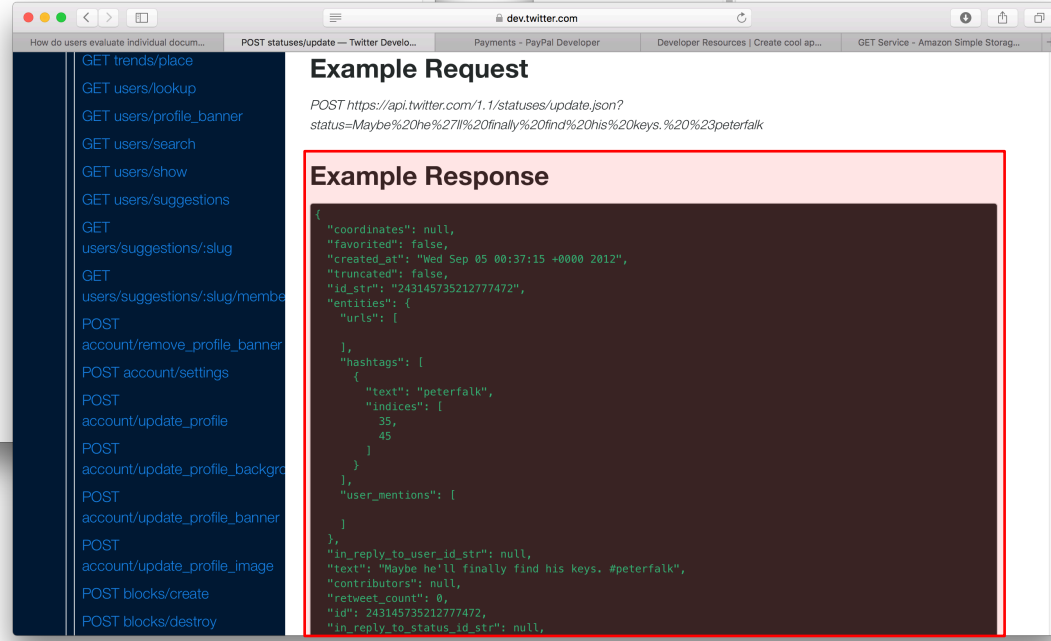


Sample Response

```

{
  "id": "PAY-5YX922393D847794YKER7MUI",
  "intent": "authorize",
  "create_time": "2015-04-24T14:26:59.059Z",
  "payer": {
    "payer_info": {
      "country_code": "DE",
      "email": "a@paypal.com",
      "first_name": "Gruneberg",
      "last_name": "Anna",
      "payer_id": "8J4WY56VUX06",
      "phone": "605-521-1234"
    },
    "payment_method": "paypal",
    "status": "VERIFIED"
  },
  "state": "approved",
  "transactions": [
    {
      "amount": {
        "total": "18.37",
        "currency": "EUR",
        "details": {
          "subtotal": "13.37",
          "shipping": "5.00"
        }
      },
      "description": "Uber",
      "item_list": {
        "items": [

```



Example Request

POST https://api.twitter.com/1.1/statuses/update.json?
status=Maybe%20he%27ll%20finally%20find%20his%20keys.%20%23peterfalk

Example Response

```

{
  "coordinates": null,
  "favorited": false,
  "created_at": "Wed Sep 05 00:37:15 +0000 2012",
  "truncated": false,
  "id_str": "24314573521277472",
  "entities": {
    "urls": [
      ],
    "hashtags": [
      {
        "text": "peterfalk",
        "indices": [
          35,
          45
        ]
      }
    ],
    "user_mentions": [
      ]
    },
  "in_reply_to_user_id_str": null,
  "text": "Maybe he'll finally find his keys. #peterfalk",
  "retweet_count": 0,
  "id": "24314573521277472",
  "in_reply_to_status_id_str": null,

```

HATEOAS

```
    },
    "description": "Uber",
    "item_list": {
      "items": [
        {
          "currency": "EUR",
          "name": "iPad",
          "price": "13.37",
          "quantity": "1"
        }
      ],
      "shipping_address": {
        "recipient_name": "Gruneberg, Anna",
        "line1": "Kathwarinenhof 1",
        "city": "Flensburg",
        "postal_code": "24939",
        "country_code": "DE"
      }
    },
    "payee": {
      "email": "paypal-de@paypal.com"
    }
  },
  "links": [
    {
      "href":
        "https://api.paypal.com/v1/payments/payment/PA
        Y-5YK922393D847794YKER7MUI",
      "method": "GET",
      "rel": "self"
    }
  ]
}
```

Listings

ayPal provides various payment-related operations through the `/payment` resource and related sub-resources. Use `payment` for direct credit card payments and PayPal account payments. You can also use sub-resources to get payment-related details.

Create a payment

POST `/v1/payments/payment`

Depending on the `payment_method` and the `funding_instrument`, you can use the payment resource for [direct credit card payments](#), [stored credit card payments](#), or [PayPal account payments](#).

Execute approved PayPal payment

POST `/v1/payments/payment_id/execute`

Executes a PayPal payment that payer has approved. Optionally, you can update transaction information when you execute the payment.

Show payment details

GET `/v1/payments/payment/payment_id`

Shows details for a payment, by ID, that is yet completed. For example, a payment that was created, approved, or failed.

List payments

GET `/v1/payments/payment`

Lists payments that were created by the `create payment` call and are in any state. The list shows the payments that are made to the merchant who makes the call.

Create a post

`/new` and `manage` posts including reblogs and likes.

Resource	Description
GET <code>/sites/\$site/posts/</code>	Get a list of matching posts.
GET <code>/sites/\$site/posts/\$post_ID</code>	Get a single post (by ID).
POST <code>/sites/\$site/posts/\$post_ID</code>	Edit a post.
GET <code>/sites/\$site/posts/slug:\$post_slug</code>	Get a single post (by slug).
POST <code>/sites/\$site/posts/new</code>	Create a post.
POST <code>/sites/\$site/posts/\$post_ID/delete</code>	Delete a post. Note: If the trash is enabled, this request will send the post to the trash. A second request will permanently delete the post.
POST <code>/sites/\$site/posts/\$post_ID/restore</code>	Restore a post or page from the trash to its previous status.
POST <code>/sites/\$site/posts/\$post_ID/likes/</code>	Get a list of posts across all the user's sites.
GET <code>/me/posts</code>	Get a list of the likes for a post.

- GET `statuses/retweeters/ids`
- GET `statuses/retweets/id`
- GET `statuses/retweets_of_me`
- GET `statuses/show/id`
- GET `statuses/user_timeline`
- GET `trends/available`
- GET `trends/closest`
- GET `trends/place`
- GET `users/lookup`
- GET `users/profile_banner`
- GET `users/search`
- GET `users/show`
- GET `users/suggestions`
- GET `users/suggestions/slug`
- GET `users/suggestions/slug/members`

POST

- POST `account/remove_profile_banner`
- POST `account/settings`
- POST `account/update_profile`
- POST `account/update_profile_background_image`
- POST `account/update_profile_banner`
- POST `account/update_profile_image`
- POST `blocks/create`
- POST `blocks/destroy`
- POST `collections/create`
- POST `collections/destroy`
- POST `collections/entries/add`
- POST `collections/entries/curate`
- POST `collections/entries/move`

OpenAPI

OpenAPI

Originated with Swagger tool for designing RESTful APIs

Represents API descriptions in JSON or YAML (Yet Another Markup Language)

- We'll concentrate on the YAML serialization

OpenAPI metadata

OpenAPI description starts with:

- Version number of OpenAPI in use
- Simple metadata about the service in the `info:` block

```
openapi: 3.0.0
info:
  version: 1.0.0
  title: Orinoco API
  description: The API for the Orinoco online bookseller
```

Servers

API endpoints are defined relative to a base URI

- Defined in OpenAPI using the servers: block

servers:

- url: `https://orinoco.com`
description: Live server
- _ url: `https://test.orinoco.com`
description: Test server (uses dummy data)

Components

components: block used to define repeatedly-used information

- Most often used to define format of message bodies

```
components:  
  schemas:  
    order:  
      type: object  
    properties:  
      items:  
        type: array  
        item:  
          type: string  
      status:  
        type: string
```

Paths

Lists available paths on the server

- e.g. <https://orinoco.com/order/1234>

For each path, lists:

- The methods which can be used on that path
- The content of any request body which should accompany the method (for PUT, POST)
- The responses which may be received from the method (including response bodies)

Path Example

```
paths:  
  /order{order_id}:  
    get:  
      description: Obtain information about an order  
      parameters:  
        - name: order_id  
          in: path  
          required: true  
          schema:  
            type: string
```

Path Example

```
paths:
  /order{order_id}:
    get:
      ...
      responses:
        '200':
          description: Successfully returned an order
          content:
            application/xml:
              schema:
                $ref: '#/components/schemas/order'
```

Summary

Documentation should cover all the bases of the web architecture

- Identification – parameterised URIs
- Interaction – HTTP methods, status codes and headers
- Representation – formats for request and response, with examples

Listings of all of the above

RESTful API Examples

Twitter

<https://developer.twitter.com/en/docs/api-reference-index>

Paypal

<https://developer.paypal.com/docs/api/payments/>

Imgur

<https://apidocs.imgur.com>

Wordpress

<https://developer.wordpress.org/rest-api/>

Tools and Further Reading

Swagger API development tool

<https://swagger.io/>

Overview of OpenAPI

<https://swagger.io/docs/specification/about/>

OpenAPI Specification

<https://github.com/OAI/OpenAPI-Specification>

Next Lecture: Trailblazers