About the API v1.0

Whos On. Location

Using our JSON based REST API you can easily break out of the boundaries of your WhosOnLocation and build integrations that can leverage your visitor and staff data, notifications, users, locations and departments. Run it on your servers the way you want it.

In the near future we will be adding more endpoints to the Web API, giving developers access to yet more WhosOnLocation data.

WhosOnLocation provides a REST API for managing data. The API currently supports JSON or XML data and is chosen via the Accept and Content-Type headers.

Contents

API Basics

This article covers the following WhosOnLocation REST API basics:

Authentication

You must be a verified WhosOnLocation user to make API requests. This is an SSL-only API and is authenticated using basic authentication. You must be a verified user to make API requests. The authenticate header takes the API key as the username and the password is not used.

API keys are managed in the WhosOnLocation Account interface. Users with the User Role of ‘Account Owner’ or ‘IT Support’ can access the API by navigating to:

Tools > Account > *API*.

The page lets you view, add, or delete keys. More than one key can be active at the same time. Deleting a key deactivates it permanently. When creating a key for the first time the User will need to accept the API Terms of Use.

Requests

WhosOnLocation provides a RESTful API for managing data. The API currently supports JSON or XML data and is chosen via the Accept and Content-Type headers

GET

Retrieve information. GET requests must be safe and idempotent, meaning regardless of how many times it repeats with the same parameters, the results are the same. They can have side effects, but the user doesn't expect them, so they cannot be critical to the operation of the system. Requests can also be partial or conditional.

Retrieve a Staff with an ID of 1

```
GET /staff/1
```

POST

Request that the resource at the URI do something with the provided entity. Often POST is used to create a new entity, but it can also be used to update an entity. See our API specification for the required data to create the resource.

Create a new staff member:

```
POST /staff
```

PUT

Update an entity at a URI. PUT can update an existing one. A PUT request is idempotent.

Modify the staff profile with an ID of 1:

```
PUT /staff/1
```

Note: PUT replaces an existing entity. If only a subset of data elements are provided, the rest will be replaced with empty or null.

DELETE

Request that a resource be removed; however, the resource does not have to be removed immediately. It could be an asynchronous or long-running request.

Delete a staff member with an ID of 1:

```
DELETE /staff/1
```

HTTP Status Codes

Status codes indicate the result of the HTTP request.

- **1XX** - informational
- **2XX** - success
- **3XX** - redirection
- **4XX** - client error
- **5XX** - server error

Common status codes returned by WhosOnLocation are

- **200 – The request has succeeded (with output)**
- **204 – The request has succeeded (without output)**
- **400 – Bad request may be malformed or missing data. See return message.**
- **403 – Forbidden. API key may be restricted to other endpoints.**
- **404 – Resource not found**
- **405 – Method not allowed (e.g. DELETE on a non-deletable resource)**
- **429 – Rate limited due to excessive requests, please retry after short pause**
- **500 – Server error. We were unable to process the request (and our development have been notified)**

Media types

The **Accept** and **Content-Type** HTTP headers can be used to describe the content being sent or requested within an HTTP request. The client may set **Accept** to **application/json** if it is requesting a response in JSON. Conversely, when sending data, setting the **Content-Type** to **application/xml** tells the client that the data being sent in the request is XML.

Rate limits

This API is rate limited. You can only query the API Service up to 100 times per minute per API Key. We reserve the right to adjust the rate limit for given endpoints to provide a high quality service for all clients.

If the rate limit is exceeded, WhosOnLocation will respond with a **HTTP 429 Too Many Requests** response code and a body that details the reason for the rate limiter kicking in. The response will also have a Retry-After header that tells you how many seconds to wait before retrying.

Legal notices

General Restrictions and Responsibilities

Your use and access to the API is expressly conditioned on your compliance with the policies, restrictions, and other provisions related to the API set forth in our API Terms of Use and the other documentation we provide you. You must also comply with the restrictions set forth in the WhosOnLocation Terms of Service and the WhosOnLocation Privacy Policy, in all uses of the API. If WhosOnLocation believes that you have or attempted to violate any term, condition, or the spirit of these policies or agreements, your right to access and use the API may be temporarily or permanently revoked.

Change policy

WhosOnLocation may modify the attributes and resources available to the API and our policies related to access and use of the API from time to time without advance notice. WhosOnLocation will use commercially reasonable efforts to notify you of any modifications to the API or policies through notifications or posts on the WhosOnLocation interface and/or @WolOps twitter account. WhosOnLocation also tracks deprecation of attributes of the API on its Changelog found in the Product Updates section of the Helpdesk.

Modification of the API may have an adverse effect on WhosOnLocation applications, including but not limited to changing the manner in which WhosOnLocation applications communicate with the API and display or transmit your data. WhosOnLocation will not be liable to you or any third party for such modifications or any adverse effects resulting from such modifications.

Common tasks performed with the WOL API

Function	Description	Function	
Staff Management	Add Staff	Creates a new staff profile	POST /staff
	Update Staff	Updates a staff profile	PUT /staff/{id}
	Delete Staff	Deletes a single Staff profile based on the ID supplied	DELETE /staff/{id}
Location Management	Add Location	Creates a new Location	POST /location
	Update Location	Updates a Location	PUT /location/{id}
	Delete Location	Deletes a single Location based on the ID supplied	DELETE /location/{id}
Department Management	Add Department	Creates a new Department	POST /department
	Update Department	Updates a Department	PUT /department/{id}
	Delete Department	Deletes a single Department based on the ID supplied	DELETE /department/{id}
Visitor Person	List of Visitors	The Visitor endpoint returns a list of visitors	GET /visitor/person
	Add new Visitor	Creates a new Visitor Person	POST /visitor/person
	Update a Visitor Record	Updates a visitor person.	PUT /visitor/person/{id}
	Delete a Visitor Record	Deletes a single Visitor Person based on the ID supplied	DELETE /visitor/person/{id}
Notifications	View a list of notifications sent	The Notifications Endpoint returns a list of messages sent to staff and visitors.	GET /notification