# Fareclock API Documentation

# Fareclock API Documentation

**Base URL:** `https://api.fareclock.com`

---

## Table of Contents



---

## Getting Started

### API Endpoint

```text
https://api.fareclock.com
```

### Authentication

All requests require an API key header:

```http
Authorization: Token YOUR_API_KEY
```

You can create API keys in Fareclock under Settings → Integrations.

### Response Format

JSON is the default response format. You can explicitly request JSON or XML:

```http
Accept: application/json
Accept: application/xml
```

---

## Concepts

### Authentication

The Fareclock API uses HTTP access authentication to authenticate each request. Multiple API keys can be created per account to support application isolation. The API keys are set up and managed in the Fareclock Admin Console application.

```http
Authorization: Token MY_API_KEY
```

To create an API key, sign in to https://www.fareclock.com/login. Then in the top menu bar, go to Settings → Integrations. On the screen below the menu bar, you can add an API key, or edit any existing ones. You can also set specific permissions for an API key.

### Versioning

The current version of the API is 0.2. If unspecified, the default version of the API will always be the latest version.

The version may be specified in the HTTP header:

```http
X-Api-Version: 0.2
```

If we make changes to the API in the future which are incompatible with the current version, then we will fork a new additional version.

## Content Format

The API currently supports JSON and XML formats. Format defaults to JSON, but can be specified in one of three ways, in the following priority order:

1. File-type media extension indicator
2. URL query string
3. HTTP Accept header

**Examples:**

```http
/punches.json
/punches.xml
?format=json
?format=xml
Accept: application/json
Accept: application/xml
```

## Request Parameters

Filter parameters in GET requests are sent using query string parameters.

Date parameters must be in ISO 8601 datetime format. If just the date part is specified (e.g. `2019-12-31`), then if the parameter is for the start of a query range, the beginning of the day will be used (`00:00`); and if the parameter is for the end of a query range, then end of the day will be used (`23:59:59`). If time is specified (e.g. `2019-12-31T23:59:00Z`), then that exact time will be used. If timezone is not specified, the organization unit timezone will be used, or otherwise the organization timezone.

Some parameters support multiple values. To query for multiple values for a specific field:

```text
field1[]=value1&field1[]=value2&field2=value2&...
```

Object data in POST/PUT requests are sent in the request body in JSON or XML format.

## HTTP Responses

HTTP response status codes reflect the outcome of the request:

| Code | Meaning |
|---|---|
| 200 | OK — Success |
| 400 | Bad Request — Likely invalid format or parameters |
| 401 | Unauthorized — Authentication failed |
| 403 | Forbidden — Permission refused |
| 404 | Not Found — Invalid URL |
| 429 | Too Many Requests — Throttle threshold exceeded |
| 500 | Internal Server Error |

Additional information may be included in the response body, such as resource data and error descriptions.

## API Errors

Errors may occur within API semantics. These errors are returned in the HTTP response using the selected content format.

JSON error example:

```json
{"detail": "Invalid token"}
```

XML error example:

```xml
<root><detail>Invalid token</detail></root>
```

## Throttling

The API currently limits the rate of requests per account:

- **Burst rate:** 50 / minute
- **Sustained rate:** 1,000 / hour

Once you reach one of these thresholds, you will receive an HTTP 429 response. You should not continue to query the API when you exceed this threshold, as it may delay the end of the limit period.

## Paging

Some API List methods support paging in order to support larger data sets.

| Parameter | Description |
| --- | --- |
| `cursor` | Inclusive value where to begin paged results, i.e. after where last page left off. Defaults to beginning or end of list, depending on direction. |
| `direction` | `1` to list values after cursor, `0` to list values before cursor. Defaults to `1`. |
| `limit` | Number of results per page. Usually defaults to 25, maximum 200. |

If the result set was paged, the response will contain a `cursor` value indicating where to begin the next page. The cursor value should be treated as an opaque string and not modified. Each subsequent page request should contain the exact identical filter query parameters as the first page.

## Async Invocation

Standard synchronous invocation of List methods may time out for larger data sets. Those collections which support asynchronous invocation are noted in each endpoint below.

To invoke the asynchronous List method, add the following URL suffix after the base method:

```http
POST {base_url}/async?{query_parameters}
```

This POST call should return HTTP status 202 with a response body:

```json
{ "success": true, "reportKey": "{REPORT_KEY}" }
```

Using the `REPORT_KEY`, poll for the results at:

```http
GET {base_url}/async/{REPORT_KEY}/poll
```

- If poll returns HTTP 202, the server is still processing. Try again later.
- If poll returns HTTP 200, the report is complete and the response body contains the result.

You should not poll more frequently than once every 10 seconds. Each poll call counts toward throttle.

## Field Selection

Some of the newer API methods support the ability to select which fields should be included. This can save memory, bandwidth, and latency. Fields can be selected via the `fields` query string parameter:

```text
?fields[]=firstName&fields[]=lastName&fields[]=duration
```

## Supplemental Data

Some of the newer API methods include a separate `supplementalData` section in the response containing related data. For example, instead of repeating `employeeFirstName` and `employeeLastName` on every punch, only the employee ID is included in each result, and the supplementalData section contains a dictionary of employee data keyed by ID.

By default, all supplemental data is included. To exclude it entirely:

```text
?supplementalData=no
```

To specify which supplemental data kinds to include:

```text
?supplementalDataFields[]=employee&supplementalDataFields[]=department
```

## Collection Capabilities

| Collection | Async Invocation | Paging | Field Selection | Supplemental Data | Notes |
|---|---|---|---|---|---|
| Employees | NO | YES | NO | NO | Paging supported. |
| Jobs | NO | YES | NO | YES | Paging results are ordered by job name. |
| Punches | YES | YES | NO | YES | Paging condition: orderBy=modified and direction=1 (ascending). |
| Time Cards | YES | NO | NO | YES | Paging is not supported. |
| Time Off Entries | NO | YES | YES | YES | Paging condition: orderBy=modified and direction=1 (ascending). |
| Delete History | NO | YES | NO | NO | Paging is ordered by deletion timestamp. |

---

# Endpoints

## Employees

### Features supported

- Support asynchronous invocation: NO
- Supports paging: YES
- Field selection: NO
- Supplemental data: NO

### `GET /employees` — List all Employees

### Query Parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| `employee` | string | No | Case-insensitive full text search of employee name |
| `orgUnit` | integer | No | Organization unit ID |
| `department` | integer | No | Department ID |
| `payClass` | integer | No | Pay class ID |
| `labels` | integer | No | User label ID |
| `active` | string | No | `active` or `inactive` |
| `orderBy` | string | No | `name` (default) or `modified` |
| `from` | date | No | Start of modification date range. Only when `orderBy=modified`. |
| `to` | date | No | End of modification date range. Only when `orderBy=modified`. |

### Response

```json
{
  "results": [
    {
      "id": 300,
      "active": true,
      "firstName": "John",
      "lastName": "Doe",
      "orgUnit": 100,
      "orgUnitName": "Organization Unit #1",
      "department": 200,
      "departmentName": "Department #1",
      "timezone": "America/New_York",
      "payrollId": null,
      "pin": "123456",
      "allowAllOrgUnits": true,
      "phones": [],
      "lastPunch": "2014-04-03T17:28:24.326Z",
      "created": "2013-12-06T20:38:21.347Z",
      "modified": "2014-04-03T17:28:24.326Z"
    }
  ]
}
```

## `POST /employees` — Create an Employee

### Request Body

```json
{
  "active": true,
  "firstName": "John",
  "lastName": "Doe",
  "orgUnit": 100,
  "department": 200,
  "payrollId": null,
  "neverHadFace": true,
  "pin": "123456",
  "allowAllOrgUnits": true,
  "allowOrgUnits": null,
  "phones": [],
  "birthDate": null,
  "hireDate": null,
  "email": null,
  "notes": null
}
```

## `GET /employees/{id}` — Retrieve an Employee

| Parameter | Type | Required | Description |
|---|---|---|---|
| `id` | integer | Yes | Numeric ID of the employee |

## `PUT /employees/{id}` — Update an Employee

The API does not allow updating an employee from inactive to active state.

### Request Body

```json
{
  "id": 300,
  "active": true,
  "firstName": "John",
  "lastName": "Doe",
  "orgUnit": 100,
  "department": 200,
  "payrollId": null,
  "pin": "123456",
  "allowAllOrgUnits": true,
  "allowOrgUnits": null,
  "phones": [],
  "notes": null
}
```

---

## Jobs

### Features supported

- Support asynchronous invocation: NO
- Supports paging: YES, with results ordered by job name
- Field selection: NO
- Supplemental data: YES

### `GET /jobs` — List all Jobs

#### Query Parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| `search` | string | No | Case-insensitive full text search of job name |
| `orgUnit` | integer | No | Organization unit ID |
| `department` | integer | No | Department ID |
| `labels` | integer | No | Work label ID |
| `active` | string | No | `active` or `inactive` |

### Response

```json
{
  "results": [
    {
      "id": 1,
      "name": "Job #1",
      "active": true,
      "orgUnits": null,
      "department": null,
      "departmentName": null,
      "payrollId": "job-1-payroll-id",
      "created": "2014-01-01T18:53:46.016Z",
      "modified": "2014-01-03T12:44:17.935Z"
    }
  ]
}
```

## `POST /jobs` — Create a Job

### Request Body

```json
{
  "name": "Job #1",
  "active": true,
  "orgUnits": null,
  "department": null,
  "payrollId": "job-1-payroll-id"
}
```

### `GET /jobs/{id}` — Retrieve a Job

| Parameter | Type | Required | Description |
|---|---|---|---|
| `id` | integer | Yes | Numeric ID of the job |

### `PUT /jobs/{id}` — Update a Job

Same request body as Create a Job.

---

## Punches

### Features supported

- Support asynchronous invocation: YES
- Supports paging: YES, with certain conditions -- orderBy must be modified, and page direction must be 1 (ascending order)
- Field selection: NO
- Supplemental data: YES

### `GET /punches` — List all Punches

#### Query Parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| `from` | date | Yes | Start of date range. Punch pairs are dated by the IN punch. |
| `to` | date | Yes | End of date range. |
| `orderBy` | string | No | `punch` (default) or `modified`. Use `modified` for periodic sync of edits. |
| `orgUnit` | integer | No | Organization unit ID of employee. |
| `department` | integer | No | Department ID. |
| `employee` | integer | No | Employee ID. If specified, orgUnit and department are ignored. |
| `clockOrgUnit` | integer | No | Organization unit ID of clock. |
| `labels` | integer | No | User or work label ID. |
| `job` | integer | No | Job ID. |
| `status` | string | No | `all` (default), `approved`, `flagged`, `exception`, `override`. |

### Response

```json
{
  "results": [
    {
      "id": 400,
      "employee": 300,
      "employeeFirstName": "John",
      "employeeLastName": "Doe",
      "employeePayrollId": "employee-payroll-id",
      "orgUnit": 100,
      "orgUnitName": "Org Unit #1",
      "inDt": "2014-03-07T14:30:00Z",
      "outDt": "2014-03-07T15:30:00Z",
      "job": 500,
      "jobName": "Job #1",
      "status": "approved",
      "approved": true,
      "flagged": false,
      "notes": "manager notes about the punch",
      "created": "2014-03-10T01:45:19.369Z",
      "modified": "2014-04-03T16:54:05.246Z"
    }
  ]
}
```

### `GET /punches/{id}` — Retrieve a Punch

| Parameter | Type | Required | Description |
|---|---|---|---|
| `id` | integer | Yes | Numeric ID of the punch |

---

## Time Cards

### Features supported

- Support asynchronous invocation: YES

- Supports paging: NO
- Field selection: NO
- Supplemental data: YES

### `GET /timecards` — List all Time Cards

#### Query Parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| `from` | date | Yes | Start of date range for shift cards. |
| `to` | date | Yes | End of date range for shift cards. |
| `orgUnit` | integer | No | Organization unit ID of employee. |
| `department` | integer | No | Department ID. |
| `payClass` | integer | No | Pay class ID. |
| `labels` | integer | No | User or work label ID. |
| `employee` | integer | No | Employee ID. If specified, orgUnit and department are ignored. |

#### Response

```json
{
  "results": []
}
```

---

## Time Off Entries

### Features supported

- Support asynchronous invocation: NO

- Supports paging: YES, with certain conditions -- orderBy must be modified, and page direction must be 1 (ascending order)
- Field selection: YES
- Supplemental data: YES

### `GET /duration-entries` — List all Time Off Entries

#### Query Parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| `from` | date | Yes | Start of date range for the date of time off entry. |
| `to` | date | Yes | Inclusive end of date range. |
| `orgUnit` | integer | No | Organization unit ID of employee. |
| `department` | integer | No | Department ID. |
| `payClass` | integer | No | Pay class ID. |
| `labels` | integer | No | User label ID. |
| `employee` | integer | No | Employee ID. If specified, orgUnit and department are ignored. |
| `orderBy` | string | No | `date` (default) or `modified`. Paging requires `orderBy=modified`. |
| `supplementalData` | string | No | `no` to exclude all supplemental data. |
| `supplementalDataFields[]` | string | No | Repeat for each kind to include: `employee`, `department`, `orgUnit`, `payClass`, `label`, `timeOffCode`. |
| `fields[]` | string | No | Repeat for each field to include in results. |

### Response

json


```
{
  "results": [
    {
      "id": 4538783999459328,
      "type": "PAID_TIME_OFF",
      "date": "2020-06-02",
      "duration": 480.0,
      "employee": 6122080743456768,
      "orgUnit": 6403555720167424,
      "department": 4996180836614144,
      "payClass": 5277655813324800,
      "labels": ["user:5559130790035456"],
      "timeOffCode": 6685030696878080,
      "created": "2020-06-02T23:11:08.690835Z",
      "modified": "2020-06-02T23:11:08.696838Z"
    }
  ],
  "supplementalData": {
    "timeOffCode": {
      "6685030696878080": { "name": "Vacation" }
    },
    "employee": {
      "6122080743456768": {
        "firstName": "Jane",
        "lastName": "Smith",
        "payrollId": "EMP-001"
      }
    }
  }
}
```

# Delete History

## Features supported

- Support asynchronous invocation: NO
- Supports paging: YES, ordered by deletion timestamp
- Field selection: NO
- Supplemental data: NO

## `GET /delete-history` — List all Delete History Items

### Query Parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| `from` | date | No | Start of date range for deletion timestamp. |
| `to` | date | No | End of date range for deletion timestamp. |
| `kind` | string | No | Object kind. Valid values: `durationEntry`, `employee`, `punch`, `punchCode`. |
| `deletedId` | integer | No | ID of the deleted object. |

### Response

```json
{
  "results": [
    {
      "id": 400,
      "timestamp": "2020-05-07T14:30:00Z",
      "kind": "employee",
      "deletedId": 500,
      "user": 100
    }
  ]
}
```