

# Package ‘imfapi’

May 8, 2026

**Title** Econdataverse 'IMF Data API' Client

**Version** 0.1.2

**Description** Provides user-friendly functions for programmatic access to macroeconomic data from the International Monetary Fund's 'SDMX 3.0 IMF Data API' <<https://data.imf.org/en/Resource-Pages/IMF-API>>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**Depends** R (>= 4.1.0)

**Imports** cli, dplyr (>= 1.1.2), purrr (>= 1.0.0), tibble (>= 3.2.1),  
httr2 (>= 0.6.0), jsonlite (>= 1.8.0)

**Suggests** curl, knitr, stringr, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**URL** <https://teal-insights.github.io/r-imfapi/>,  
<https://github.com/Teal-Insights/r-imfapi>

**BugReports** <https://github.com/Teal-Insights/r-imfapi/issues>

**NeedsCompilation** no

**Author** Teal Insights [cre, cph],  
Christopher C. Smith [aut],  
Christoph Scheuch [ctb]

**Maintainer** Teal Insights <lte@tealinsights.com>

**Repository** CRAN

**Date/Publication** 2025-11-19 20:20:02 UTC

## Contents

|                                 |   |
|---------------------------------|---|
| imf_get . . . . .               | 2 |
| imf_get_codelists . . . . .     | 3 |
| imf_get_dataflows . . . . .     | 4 |
| imf_get_datastructure . . . . . | 5 |

---

|         |                                          |
|---------|------------------------------------------|
| imf_get | <i>Retrieve data from an IMF dataset</i> |
|---------|------------------------------------------|

---

### Description

Fetches observations for a given `dataflow_id` and `resource_id` from the IMF SDMX 3.0 Data API. The request key is constructed from the dataset's datastructure (DSD) using the positional order of dimensions. Time filtering is applied via query parameters.

### Usage

```
imf_get(
  dataflow_id,
  dimensions = list(),
  start_period = NULL,
  end_period = NULL,
  progress = FALSE,
  max_tries = 10L,
  cache = TRUE
)
```

### Arguments

|                           |                                                                                                                                                                                                                                                                 |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>dataflow_id</code>  | Character scalar. The dataflow to query (e.g., "GFS").                                                                                                                                                                                                          |
| <code>dimensions</code>   | Named list mapping dimension IDs to character vectors of codes to include. Omitted dimensions are wildcarded in the key. Each dimension position in the DSD corresponds to one dot-separated slot in the key; multiple codes per slot are joined by '+'.        |
| <code>start_period</code> | Optional character. Lower bound for time filtering (e.g., "2000", "2000-Q1", "2000-01").                                                                                                                                                                        |
| <code>end_period</code>   | Optional character. Upper bound for time filtering, same format as <code>start_period</code> . The request always uses the SDMX 3.0 dataflow context under the hood and sets <code>dimensionAtObservation = "TIME_PERIOD"</code> to request a time-series view. |
| <code>progress</code>     | Logical; whether to show request progress.                                                                                                                                                                                                                      |
| <code>max_tries</code>    | Integer; maximum retry attempts for HTTP requests.                                                                                                                                                                                                              |
| <code>cache</code>        | Logical; whether to enable caching for HTTP requests.                                                                                                                                                                                                           |

### Details

By default, the request targets the all agencies scope for the data path, assuming dataflow IDs are globally unique in practice. The response layout uses a time-series context, and client code will shape the parsed payload into a tidy tibble.

The request key is built by ordering dimensions by their DSD position and filling each position with either a '+'-joined set of selected codes or a blank for wildcard. Time filtering is applied via `start_period` and `end_period` query parameters rather than encoding time into the key.

**Value**

A tibble with one row per observation, including dimension columns, time period, value column(s), and any requested attributes. Exact column names follow the dataset's DSD and may vary by dataflow\_id.

**Examples**

```
if (curl::has_internet()) {
  imf_get(
    dataflow_id = "FM", # Fiscal Monitor
    dimensions = list(COUNTRY = c("USA", "CAN"))
  )
}
```

---

|                   |                                                                   |
|-------------------|-------------------------------------------------------------------|
| imf_get_codelists | <i>Retrieve codes for one or more dimensions as a tidy tibble</i> |
|-------------------|-------------------------------------------------------------------|

---

**Description**

Returns a tibble mapping dimensions to their codes and labels by fetching the corresponding codelists. By convention, codelist IDs are assumed to be CL\_{dimension\_id} for first-pass coverage.

**Usage**

```
imf_get_codelists(
  dimension_ids,
  dataflow_id,
  progress = FALSE,
  max_tries = 10L,
  cache = TRUE
)
```

**Arguments**

|               |                                                                                                                                   |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------|
| dimension_ids | Character vector of dimension IDs (e.g., "COUNTRY").                                                                              |
| dataflow_id   | Character scalar. The dataflow whose datastructure is used to resolve each dimension's codelist via its concept scheme reference. |
| progress      | Logical; whether to show progress.                                                                                                |
| max_tries     | Integer; maximum retry attempts.                                                                                                  |
| cache         | Logical; whether to cache requests.                                                                                               |

**Value**

```
tibble::tibble( dimension_id = character(), code = character(), name = character(), description =
character(), codelist_id = character(), codelist_agency = character(), codelist_version = character()
)
```

## Examples

```
if (curl::has_internet()) {  
  imf_get_codelists(  
    c("FREQUENCY", "TIME_PERIOD"),  
    dataflow_id = "FM" # Fiscal Monitor  
  )  
}
```

---

imf\_get\_dataflows      *Get dataflow definitions for aLL available IMF datasets*

---

## Description

Retrieves and returns all available dataflow definitions from the SDMX dataflow endpoint.

## Usage

```
imf_get_dataflows(progress = FALSE, max_tries = 10L, cache = TRUE)
```

## Arguments

|           |                                        |
|-----------|----------------------------------------|
| progress  | Logical; whether to show progress.     |
| max_tries | Integer; maximum retry attempts.       |
| cache     | Logical; whether to cache the request. |

## Value

tibble::tibble( id = character(), # e.g., "MFS\_IR", "SPE", etc. name = character(), # English name  
description = character(), # English description version = character(), # e.g., "8.0.1" structure =  
character(), # DSD reference last\_updated = character() # from annotations )

## Examples

```
if (curl::has_internet()) {  
  imf_get_dataflows()  
}
```

---

imf\_get\_datastructure *Retrieve the datastructure definition for an IMF dataflow.*

---

### Description

Retrieve the datastructure definition for an IMF dataflow.

### Usage

```
imf_get_datastructure(  
  dataflow_id,  
  progress = FALSE,  
  max_tries = 10L,  
  cache = TRUE,  
  include_time = FALSE,  
  include_measures = FALSE  
)
```

### Arguments

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| dataflow_id      | The ID of the dataflow to retrieve the datastructure for. |
| progress         | Logical; whether to show progress.                        |
| max_tries        | Integer; maximum retry attempts.                          |
| cache            | Logical; whether to cache the request.                    |
| include_time     | Logical; whether to include time dimensions.              |
| include_measures | Logical; whether to include measure dimensions.           |

### Value

tibble::tibble( dimension\_id = character(), type = character(), position = integer() )

### Examples

```
if (curl::has_internet()) {  
  imf_get_datastructure("PSBS")  
}
```

# Index

`imf_get`, [2](#)  
`imf_get_codelists`, [3](#)  
`imf_get_dataflows`, [4](#)  
`imf_get_datastructure`, [5](#)