istacpy

Instituto Canario de Estadistica (ISTAC)

May 06, 2021

CONTENTS

istacpy is a Python package for obtaining open data from Instituto Canario de Estadística (ISTAC). It provides a wrapper to the open API catalog.

CHAPTER

ONE

INSTALLATION

```
pip install istacpy
```

CHAPTER

TWO

USAGE

The package is divided into several modules depending on the resource you want to retrieve:

2.1 Indicators

2.1.1 istacpy.indicators.lite

This is a lite version of the rest of the indicators API. It's a kind of wrapper to facilitate the access to indicators data. It's not as powerful as the underneath functions but it hides a lot of the business logic of the API, so it's quite suitable to quickly retrieve information.

An indicator is defined by three dimensions: **geographical**, **time** and **measure**. Through this lite submodule the idea is to reduce these dimensions setting a custom user value for each one:

Import the module

First of all, you have to import the right submodule:

```
>>> from istacpy.indicators.lite import indicators
```

Look for subjects

Indicators are grouped in subjects. For this purpose the function `istacpy.indicators.lite.indicators.get_subjects()` is provided. Access easily to these subjects:

```
>>> indicators.get_subjects()
(('011', 'Territorio y usos del suelo'),
 ('012', 'Medio ambiente'),
 ('021', 'Población'),
 ('022', 'Movimiento natural'),
 ('023', 'Movimientos migratorios'),
 ...
```

Each tuple represents: **subject code** and **subject title**. Subjects are **sorted** by its code.

Look for indicators

For this purpose the function `istacpy.indicators.lite.indicators.get_indicators()` is provided. Look for indicators using a *subject code*:

```
>>> indicators.get_indicators(subject_code='022')
(('DEFUNCIONES', 'Defunciones'),
 ('DEFUNCIONES_HOMBRES', 'Defunciones. Hombres'),
 ('DEFUNCIONES_MUJERES', 'Defunciones. Mujeres'),
 ('MATRIMONIOS_SEXO_DIFERENTE',
  'Matrimonios. Entre cónyuges de diferente sexo'),
  ...
```

Each tuple represents: **indicator code** and **indicator title**. Indicators are **sorted** by its code. No need to paginate results since all them are retrieved at once.

You can even search for indicators within a certain subject and using a query string. For instance, let's say you want to find out data about births within the subject with code `022`:

```
>>> indicators.get_indicators('nacimiento', subject_code='022')
(('NACIMIENTOS', 'Nacimientos'),
 ('NACIMIENTOS_HOMBRES', 'Nacimientos. Hombres'),
 ('NACIMIENTOS_MUJERES', 'Nacimientos. Mujeres'),
 ('TASA_FECUNDIDAD', 'Tasa de fecundidad'),
 ('TASA_FECUNDIDAD_10A14', 'Tasa de fecundidad. De 10 a 14 años'),
 ('TASA_FECUNDIDAD_15A19', 'Tasa de fecundidad. De 15 a 19 años'))
```

You are not restricted to look for indicators always within a subject. For instance, suppose you need to retrieve data about employment:

```
>>> indicators.get_indicators('empleo')
(('ACCIDENTES_TRABAJO_BAJA', 'Accidentes de trabajo con baja'),
 ('ACCIDENTES_TRABAJO_BAJA_JORNADAS',
  'Accidentes de trabajo con baja. Jornadas no trabajadas'),
 ('AFILIACIONES', 'Afiliaciones a la Seguridad Social'),
 ('AFILIACIONES_AGRICULTURA',
  'Afiliaciones a la Seguridad Social. Agricultura'),
 ('AFILIACIONES_ALOJAMIENTO',
  'Afiliaciones a la Seguridad Social. Servicios de alojamiento'),
  ...
```

Internationalization

As seen in previous examples, information is retrieved in Spanish. But it's also possible to set English as the default language of this lite indicators module:

```
>>> from istacpy.indicators.lite import i18n
>>> i18n.set_english()
```

Now you can search indicators using english terms and the returned values will (mostly) be also in english. In the next example indicators about *employment* will be looked for:

```
>>> indicators.get_indicators('employment')
(('AFILIACIONES_ASALARIADOS',
  'Affiliations to Social Security. Wage employment'),
 ('AFILIACIONES_ASALARIADOS_HOMBRES',
```

(continues on next page)

(continued from previous page)

```
  'Affiliations to Social Security. Wage employment. Men'),
 ('AFILIACIONES_ASALARIADOS_MUJERES',
  'Affiliations to Social Security. Wage employment. Women'),
 ('AFILIACIONES_AUTONOMOS',
  'Affiliations to Social Security. Self-employment'),
 ('AFILIACIONES_AUTONOMOS_HOMBRES',
  'Affiliations to Social Security. Self-employment. Men'),
  ...
```

It's possible to go back and enable again spanish language using `i18n.set_spanish()`.

Note: Not all data from API is fully translated to English. So please be patient if it's the case for you.

Indicator

Once you have identified which indicator you want to work with, it's time to get it from the API. To accomplish it, you will have to use the *indicator code*. Let's say you are interested in *population*, and more precisely, in the `POBLACION` indicator:

```
>>> indicator = indicators.get_indicator('POBLACION')

>>> indicator
POBLACION (Population)
```

For this purpose the class `istacpy.indicators.lite.indicators.Indicator` is provided. You can get more information about this indicator using the next method:

```
>>> indicator.info()
· Class: istacpy.indicators.lite.indicators.Indicator
· Indicator code: POBLACION
· Title: Population
· Subject: Not available
· Description: Number of persons according to official population figures, referred␣
↪to 1 January of each year, drawn from the Municipal Population Register
· Geographical granularities: {'COUNTIES': 'C', 'ISLANDS': 'I', 'REGIONS': 'R',
↪'MUNICIPALITIES': 'M'}
· Time granularities: {'YEARLY': 'Y'}
· Measures: {'ABSOLUTE': 'A', 'ANNUAL_PERCENTAGE_RATE': 'N', 'INTERPERIOD_PERCENTAGE_
↪RATE': 'I', 'ANNUAL_PUNTUAL_RATE': 'M', 'INTERPERIOD_PUNTUAL_RATE': 'J'}
· Available years: 2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,
↪2013,2014,2015,2016,2017,2018,2019
```

As you can see, every field contains helpful information:

- *Class*: shows the full qualified name of the indicator.
- *Indicator code*: shows the indicator code and can be also independently accessed through the `code` attribute.
- *Title*: shows the indicator title (internationalized if proceed) and can be also independently accessed through the `title` attribute.
- *Subject*: shows the subject where this indicator is included (internationalized if proceed) and can be also independently accessed through the `subject` attribute.
- *Description*: shows the indicator description (internationalized if proceed) and can be also independently accessed through the `description` attribute.

- *Geographical granularities*: shows the available geographical granularities for this indicator and can be also independently accessed through the `geographical_granularities` attribute. It's a **dict** where *keys* are **granularity codes** and *values* are **granularity ids** (they will be use later).
- *Time granularities*: shows the available time granularities for this indicator and can be also independently accessed through the `time_granularities` attribute. It's a **dict** where *keys* are **granularity codes** and *values* are **granularity ids** (they will be used later).
- *Measures*: shows the available measures for this indicator and can be also independently accessed through the `measures` attribute. It's a **dict** where *keys* are **measure codes** and *values* are **measure ids** (they will be used later).
- *Available years*: shows the available years (as time dimension) for this indicator and can be also independently accessed through the `available_years` attribute. It's a list containing the available years for the different combinations of granularities and measures.

Note: It's possible that some available year has no data for a certain combination of granularities and measures, since `available_years` is just a summary of all possible time slots.

Since this object internally uses an *istacpy.indicators.indicators* method to retrieve data from API, you can always access this information through `indicator.api_response`.

Indicator Data

Once you have inspected our indicator, you are ready to get some data from it. For this purpose the method `istacpy.indicators.lite.indicators.Indicator.get_data()` is provided. Suppose you need to know the evolution of population on every Canary island. Query this through the next sentence:

```
>>> data = indicator.get_data(geo='I')

>>> data
POBLACION (Población)
<2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,2015,2016,
↪2017,2018,2019>
{'Lanzarote': (96310, 103044, 109942, 114715, 116782, 123039, 127457, 132366, 139506,␣
↪141938, 141437, 142517, 142132, 141953, 141940, 143209, 145084, 147023, 149183,␣
↪152289), 'Fuerteventura': (60124, 66025, 69762, 74983, 79986, 86642, 89680, 94386,␣
↪100929, 103167, 103492, 104072, 106456, 109174, 106930, 107367, 107521, 110299,␣
↪113275, 116886), 'Gran Canaria': (741161, 755489, 771333, 789908, 790360, 802247,␣
↪807049, 815379, 829597, 838397, 845676, 850391, 852225, 852723, 851157, 847830,␣
↪845195, 843158, 846717, 851231), 'Tenerife': (709365, 744076, 778071, 799889,␣
↪812839, 838877, 852945, 865070, 886033, 899833, 906854, 908555, 898680, 897582,␣
↪889936, 888184, 891111, 894636, 904713, 917841), 'La Gomera': (18300, 18990, 19098,␣
↪19580, 21220, 21746, 21952, 22259, 22622, 22769, 22776, 23076, 22350, 21153, 20721,␣
↪20783, 20940, 20976, 21136, 21503), 'La Palma': (82483, 84319, 85547, 85631, 84282,␣
↪85252, 86062, 85933, 86528, 86996, 87324, 87163, 85468, 85115, 83456, 82346, 81486,␣
↪81350, 81863, 82671), 'El Hierro': (8533, 9423, 10002, 10162, 10071, 10477, 10688,␣
↪10558, 10753, 10892, 10960, 10995, 11033, 10979, 10675, 10587, 10587, 10679, 10798,␣
↪10968)}
```

For this purpose the class `istacpy.indicators.lite.indicators.IndicatorData` is provided. Get more information about this data using the next method:

```
>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
```

(continues on next page)

(continued from previous page)

```
· Indicator code: POBLACION
· Title: Población
· Geographical granularity: ISLANDS
· Time granularity: YEARLY
· Measure: ABSOLUTE
· Index: 2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,
↪2015,2016,2017,2018,2019
· Columns: Lanzarote,Fuerteventura,Gran Canaria,Tenerife,La Gomera,La Palma,El Hierro
· Shape: (20, 7)
· Num. observations: 140
```

As you can see, every field contains helpful information:

- *Class*: shows the full qualified name of the indicator data.
- *Indicator code*: shows the indicator code and can be also independently accessed through the `code` attribute.
- *Title*: shows the indicator title (internationalized if proceed) and can be also independently accessed through the `title` attribute.
- *Geographical granularity*: shows the pinned geographical granularity for this dataset and can be also independently accessed through the `geographical_granularity` attribute.
- *Time granularity*: shows the pinned time granularity for this dataset and can be also independently accessed through the `time_granularity` attribute.
- *Measure*: shows the pinned measure for this dataset and can be also independently accessed through the `measure` attribute.
- *Index*: shows the index of the dataset as a tuple of **sorted** years. It can be also independently accessed through the `index` attribute.
- *Columns*: shows the columns of the dataset as a tuple of geographical locations. It can be also independently accessed through the `columns` attribute.
- *Shape*: shows a tuple with index size by columns size and can be also independently accessed through the `shape` attribute.
- *Num. observations*: shows the total number of observations within the dataset and can be also independently accessed through the `num_observations` attribute.

Although data itself is not shown on `info()` calling, it's always available through `.data` attribute:

```
>>> data.data
{'Lanzarote': (96310,
  103044,
  109942,
  114715,
  116782,
  123039,
  127457,
  132366,
  139506,
  141938,
  141437,
  142517,
  142132,
  141953,
  141940,
  143209,
```

(continues on next page)

(continued from previous page)

```
  145084,
  147023,
  149183,
  152289),
 'Fuerteventura': (60124,
  66025,
  69762,
  ...
```

Note: Each indicator is linked to an indicator data. So, you can access extra information through this attribute `data.indicator`.

Since this object internally uses an *istacpy.indicators.indicators* method to retrieve data from API, you can always access this information through `data.api_response`.

Convert to dataframe

In case you are working with Pandas it's super easy to convert indicator data to dataframe:

```
>>> data.as_dataframe()
      Lanzarote  Fuerteventura  Gran Canaria  Tenerife  La Gomera  La Palma  El Hierro
2000      96310          60124        741161    709365      18300     82483       8533
2001     103044          66025        755489    744076      18990     84319       9423
2002     109942          69762        771333    778071      19098     85547      10002
2003     114715          74983        789908    799889      19580     85631      10162
2004     116782          79986        790360    812839      21220     84282      10071
2005     123039          86642        802247    838877      21746     85252      10477
2006     127457          89680        807049    852945      21952     86062      10688
2007     132366          94386        815379    865070      22259     85933      10558
2008     139506         100929        829597    886033      22622     86528      10753
2009     141938         103167        838397    899833      22769     86996      10892
2010     141437         103492        845676    906854      22776     87324      10960
2011     142517         104072        850391    908555      23076     87163      10995
2012     142132         106456        852225    898680      22350     85468      11033
2013     141953         109174        852723    897582      21153     85115      10979
2014     141940         106930        851157    889936      20721     83456      10675
2015     143209         107367        847830    888184      20783     82346      10587
2016     145084         107521        845195    891111      20940     81486      10587
2017     147023         110299        843158    894636      20976     81350      10679
2018     149183         113275        846717    904713      21136     81863      10798
2019     152289         116886        851231    917841      21503     82671      10968
```

Important: `pip install pandas` is a required dependency in case you want to use `.as_dataframe()` method.

For this purpose the method `istacpy.indicators.lite.indicators.Indicator.as_dataframe()` is provided.

Convert to list

It's also possible to flatten data and get a list of values as follows:

```
>>> data.as_list()
[96310,
 103044,
 109942,
 114715,
 116782,
 123039,
 127457,
 ...
 10979,
 10675,
 10587,
 10587,
 10679,
 10798,
 10968]
```

For this purpose the method `istacpy.indicators.lite.indicators.Indicator.as_list()` is provided.

Default values

If no arguments are given, you will get data with default granularities and measures. More precisely, returned data will use the following **specifications by default**:

- Geographical granularity: will be set by default as the granularity with the largest available grain.
- Time granularity: will be set by default as the granularity with the largest available grain.
- Measure: will be set by default as the absolute measure.

For example, you could get data from *population* indicator as follows (**using default values**):

```
>>> data = indicator.get_data()

>>> data
POBLACION (Población)
<2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,2015,2016,
↪2017,2018,2019>
{'Canarias': (1716276, 1781366, 1843755, 1894868, 1915540, 1968280, 1995833, 2025951,␣
↪2075968, 2103992, 2118519, 2126769, 2118344, 2118679, 2104815, 2100306, 2101924,␣
↪2108121, 2127685, 2153389)}
```

Check the used values for granularities and measure as follows:

```
>>> data.geographical_granularity
'REGIONS'

>>> data.time_granularity
'YEARLY'

>>> data.measure
'ABSOLUTE'
```

Query format

One of the most important features of this module is to allow queries in an easy and powerful way. Let's see distinct use cases to demonstrate its capabitilies.

A population indicator will be used within the next examples:

```
>>> indicator = indicators.get_indicator('POBLACION')

>>> indicator.info()
· Class: istacpy.indicators.lite.indicators.Indicator
· Indicator code: POBLACION
· Title: Population
· Subject: Not available
· Description: Number of persons according to official population figures, referred␣
↪to 1 January of each year, drawn from the Municipal Population Register
· Geographical granularities: {'REGIONS': 'R', 'ISLANDS': 'I', 'COUNTIES': 'C',
↪'MUNICIPALITIES': 'M'}
· Time granularities: {'YEARLY': 'Y'}
· Measures: {'ABSOLUTE': 'A', 'ANNUAL_PERCENTAGE_RATE': 'N', 'INTERPERIOD_PERCENTAGE_
↪RATE': 'I', 'ANNUAL_PUNTUAL_RATE': 'M', 'INTERPERIOD_PUNTUAL_RATE': 'J'}
· Available years: 2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,
↪2013,2014,2015,2016,2017,2018,2019
```

Evolution of population in counties of Tenerife

```
>>> data = indicator.get_data(geo='C|Tenerife')
```

```
>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
· Indicator code: POBLACION
· Title: Population
· Geographical granularity: COUNTIES
· Time granularity: YEARLY
· Measure: ABSOLUTE
· Index: 2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,
↪2015,2016,2017,2018,2019
· Columns: Tenerife - Área Metropolitana,Tenerife Norte - Acentejo,Tenerife Norte -␣
↪Daute,Tenerife Norte - Icod,Tenerife Norte - Valle de La Orotava,Tenerife Sur -␣
↪Abona,Tenerife Sur - Suroeste,Tenerife Sur - Valle de Güímar
· Shape: (20, 8)
· Num. observations: 160
```

Here we used the `geo` argument (for *geographical granularity*), indicating `C` for `COUNTIES` and filtering by *Tenerife*. As we described in *Default values*, measure is set to *absolute* when no value is provided.

Evolution of population in municipalities of Lanzarote and Fuerteventura

```
>>> data = indicator.get_data(geo='M|Lanzarote,Fuerteventura')
```

```
>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
· Indicator code: POBLACION
· Title: Population
· Geographical granularity: MUNICIPALITIES
· Time granularity: YEARLY
· Measure: ABSOLUTE
```

(continues on next page)

(continued from previous page)

```
· Index: 2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,
↪2015,2016,2017,2018,2019
· Columns: Arrecife,Haría,San Bartolomé,Teguise,Tías,Tinajo,Yaiza,Antigua,Betancuria,
↪La Oliva,Pájara,Puerto del Rosario,Tuineje
· Shape: (20, 13)
· Num. observations: 260
```

Here we used the `geo` argument (for *geographical granularity*), indicating `M` for `MUNICIPALITIES` and filtering by *Lanzarote* and *Gran Canaria*. As we described in *Default values*, measure is set to *absolute* when no value is provided.

Comparation of population in counties between 2009 and 2019

```
>>> data = indicator.get_data(geo='C', time='Y|2009:2019')
```

```
>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
· Indicator code: POBLACION
· Title: Population
· Geographical granularity: COUNTIES
· Time granularity: YEARLY
· Measure: ABSOLUTE
· Index: 2009,2010,2011,2012,2013,2014,2015,2016,2017,2018,2019
· Columns: Lanzarote - Este,Lanzarote - Norte,Lanzarote - Suroeste,Fuerteventura -␣
↪Centro,Fuerteventura - Norte,Fuerteventura - Sur,Gran Canaria - Área Metropolitana,
↪Gran Canaria Norte - Centro Norte,Gran Canaria Norte - Noroeste,Gran Canaria Norte -
↪ Oeste,Gran Canaria Sur - Sur,Gran Canaria Sur - Sureste,Tenerife - Área␣
↪Metropolitana,Tenerife Norte - Acentejo,Tenerife Norte - Daute,Tenerife Norte -␣
↪Icod,Tenerife Norte - Valle de La Orotava,Tenerife Sur - Abona,Tenerife Sur -␣
↪Suroeste,Tenerife Sur - Valle de Güímar,La Gomera - Norte,La Gomera - Sur,La Palma -
↪ Capitalina,La Palma - Noreste,La Palma - Noroeste,La Palma - Valle de Aridane,El␣
↪Hierro - El Hierro
· Shape: (11, 27)
· Num. observations: 297
```

Here we used the `geo` argument (for *geographical granularity*), indicating `C` for `COUNTIES` and `time` argument (for *time granularity*), indicating `Y` for `YEARLY` and filtering with a range between two years. Range can be either specified with `:` or `-`. As we described in *Default values*, measure is set to *absolute* when no value is provided.

Comparation of population in islands between first and last available years

```
>>> data = indicator.get_data(geo='I', time='Y|F,L')
```

```
>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
· Indicator code: POBLACION
· Title: Population
· Geographical granularity: ISLANDS
· Time granularity: YEARLY
· Measure: ABSOLUTE
· Index: 2000,2019
· Columns: Lanzarote,Fuerteventura,Gran Canaria,Tenerife,La Gomera,La Palma,El Hierro
· Shape: (2, 7)
· Num. observations: 14
```

Here we used the `geo` argument (for *geographical granularity*), indicating `I` for `ISLANDS` and `time` argument (for *time granularity*) filtering by first available year (denoted by `F`) and last available year (denoted by `L`). Note that time

filter uses a comma indicating separate values. As we described in *Default values*, measure is set to *absolute* when no value is provided.

Comparation of population in municipalities of La Gomera between 2005-2010 and 2015-last available year

```
>>> data = indicator.get_data(geo='M|La Gomera', time='Y|2005:2010,2015:L')
```

```
>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
· Indicator code: POBLACION
· Title: Population
· Geographical granularity: MUNICIPALITIES
· Time granularity: YEARLY
· Measure: ABSOLUTE
· Index: 2005,2006,2007,2008,2009,2010,2015,2016,2017,2018,2019
· Columns: Agulo,Alajeró,Hermigua,San Sebastián de La Gomera,Valle Gran Rey,
↪Vallehermoso
· Shape: (11, 6)
· Num. observations: 66
```

Here we used the `geo` argument (for *geographical granularity*), indicating `M` for `MUNICIPALITIES` and filtering by *La Gomera* and `time` argument (for *time granularity*) indicating `Y` for `YEARLY` and filtering by the required ranges. Note that `L` stands for *last available year* and can be used in every expression. As we described in *Default values*, measure is set to *absolute* when no value is provided.

Evolution of population in counties of Gran Canaria in annual percentage rate

```
>>> data = indicator.get_data(geo='C|Gran Canaria', measure='N')
```

```
>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
· Indicator code: POBLACION
· Title: Population
· Geographical granularity: COUNTIES
· Time granularity: YEARLY
· Measure: ANNUAL_PERCENTAGE_RATE
· Index: 2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,
↪2015,2016,2017,2018,2019
· Columns: Gran Canaria - Área Metropolitana,Gran Canaria Norte - Centro Norte,Gran␣
↪Canaria Norte - Noroeste,Gran Canaria Norte - Oeste,Gran Canaria Sur - Sur,Gran␣
↪Canaria Sur - Sureste
· Shape: (20, 6)
· Num. observations: 120
```

Here we used the `geo` argument (for *geographical granularity*), indicating `C` for `COUNTIES` and filtering by *Gran Canaria* and `measure` argument (for *measure representation*) indicating `N` for `ANNUAL_PERCENTAGE_RATE`. As we described in *Default values*, time is set to *largest grane* when no value is provided.

Evolution of vehicles in circulation in every Canary island with monthly values during the last available year

First of all you have to choose the right indicator:

```
>>> indicator = indicators.get_indicator('PARQUE_VEHICULOS')

>>> indicator.info()
· Class: istacpy.indicators.lite.indicators.Indicator
· Indicator code: PARQUE_VEHICULOS
· Title: National feet of vehicles
```

(continues on next page)

(continued from previous page)

```
· Subject: Not available
· Description: Number of vehicles in circulation according to the fleet prepared by␣
↪the Spanish Traffic Authority. The reference date is the last day of each month
· Geographical granularities: {'REGIONS': 'R', 'ISLANDS': 'I', 'MUNICIPALITIES': 'M'}
· Time granularities: {'YEARLY': 'Y', 'MONTHLY': 'M'}
· Measures: {'ABSOLUTE': 'A', 'ANNUAL_PERCENTAGE_RATE': 'N', 'INTERPERIOD_PERCENTAGE_
↪RATE': 'I', 'ANNUAL_PUNTUAL_RATE': 'M', 'INTERPERIOD_PUNTUAL_RATE': 'J'}
· Available years: 2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,2015,2016,2017
```

Now you can make the right query:

```
>>> data = indicator.get_data(geo='I', time='M|L')

>>> data.info()
· Class: istacpy.indicators.lite.indicators.IndicatorData
· Indicator code: PARQUE_VEHICULOS
· Title: National feet of vehicles
· Geographical granularity: ISLANDS
· Time granularity: MONTHLY
· Measure: ABSOLUTE
· Index: Jan 2017,Feb 2017,Mar 2017,Apr 2017,May 2017,Jun 2017,Jul 2017,Aug 2017,Sep␣
↪2017,Oct 2017,Nov 2017,Dec 2017
· Columns: Lanzarote,Fuerteventura,Gran Canaria,Tenerife,La Gomera,La Palma,El Hierro
· Shape: (12, 7)
· Num. observations: 84

>>> data
PARQUE_VEHICULOS (National feet of vehicles)
<Jan 2017,Feb 2017,Mar 2017,Apr 2017,May 2017,Jun 2017,Jul 2017,Aug 2017,Sep 2017,Oct␣
↪2017,Nov 2017,Dec 2017>
{'Lanzarote': (119405, 119707, 120112, 120313, 120913, 121656, 122850, 123042, 123401,
↪ 123774, 124536, 125137), 'Fuerteventura': (81898, 82203, 82561, 82748, 83077,␣
↪83444, 83851, 84170, 84499, 84949, 85208, 85485), 'Gran Canaria': (607174, 608634,␣
↪610313, 610878, 612377, 614437, 617703, 619479, 621736, 625164, 626967, 628259),
↪'Tenerife': (687245, 689104, 691208, 693475, 695655, 697987, 700952, 702968, 704847,
↪ 706923, 709383, 710869), 'La Gomera': (14627, 14666, 14684, 14715, 14742, 14771,␣
↪14798, 14808, 14838, 14893, 14956, 14984), 'La Palma': (67895, 68034, 68205, 68362,␣
↪68532, 68737, 68871, 69037, 69200, 69443, 69679, 69652), 'El Hierro': (8180, 8203,␣
↪8218, 8250, 8262, 8286, 8325, 8345, 8378, 8402, 8442, 8451)}
```

Here we used the `geo` argument (for *geographical granularity*), indicating `I` for `ISLANDS` and `time` argument (for *time granularity*), indicating `M` for `MONTHLY` and filtering by `L` (for the *last available year*). As we described in *Default values*, measure is set to *absolute* when no value is provided.

Automatic conversion of data

API essentially returns string data. This lite module converts values to its proper numeric representation (`int` or `float`). Besides, it handles possible `NaN` values (*not a number*) when conversion is not possible.

This feature can be illustrated throught a *sales of cement* indicator. Let's retrieve some data:

```
>>> indicator = indicators.get_indicator('CEMENTO_VENTAS')

>>> indicator
CEMENTO_VENTAS (Wholesale of cement)
```

(continues on next page)

(continued from previous page)

```
>>> data = indicator.get_data()
```

Let's take a look of the API response:

```
>>> data.api_response['observation']
['562199.2',
 '587198.8',
 '527980.0',
 '508627.2',
 '498687.5',
 '451703.5',
 '440478.3',
 '517181.9',
 '678307.4',
 '819994.0',
 '.']
```

You can see that values are *strings* and a *dot* is also in the list. These values are correctly handled by the lite module, converting them to numeric types and identifying `NaN` observations:

```
>>> data
CEMENTO_VENTAS (Wholesale of cement)
<2009,2010,2011,2012,2013,2014,2015,2016,2017,2018,2019>
{'Canarias': ('NaN', 819994.0, 678307.4, 517181.9, 440478.3, 451703.5, 498687.5,␣
↪508627.2, 527980.0, 587198.8, 562199.2)}
```

Automatic conversion from multipying units

There are some indicators whose response from API is based on **multiplying units**, that is, returned values must be multiplied by this factor to get the real ones.

This module converts automatically these values to stop worrying about factors.

For example, if you are working with *unemployed population*, this indicator has a *thousands* multiplying unit. Let's see how this automatic conversion works:

```
>>> indicator = indicators.get_indicator('POBLACION_PARADA')

>>> indicator.info()
· Class: istacpy.indicators.lite.indicators.Indicator
· Indicator code: POBLACION_PARADA
· Title: Unemployed population
· Subject: Not available
· Description: Persons aged 16 years and more without work, available to start work␣
↪and had actively sought work
· Geographical granularities: {'REGIONS': 'R', 'ISLANDS': 'I', 'COUNTIES': 'C'}
· Time granularities: {'QUARTERLY': 'Q'}
· Measures: {'ABSOLUTE': 'A', 'ANNUAL_PERCENTAGE_RATE': 'N', 'INTERPERIOD_PERCENTAGE_
↪RATE': 'I', 'ANNUAL_PUNTUAL_RATE': 'M', 'INTERPERIOD_PUNTUAL_RATE': 'J'}
· Available years: 2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,
↪2015,2016,2017,2018,2019,2020

>>> data = indicator.get_data()
```

Let's compare the last three values of `data` against the last three raw values from API response:

```
>>> data.api_response['observation'][-3:][::-1]
['7.62', '9.66', '9.87']

>>> values = data.as_list()
>>> values[:3]
[7620.0, 9660.0, 9870.0]
```

You can check that values are correctly converted.

Getting raw codes

By default, the `.get_data()` method provides titles for *index* and *columns* (when possible). This behaviour can be changed if you specify = as a prefix on the query string (for each dimension).

Query in a "normal" way:

```
>>> indicator = indicators.get_indicator('POBLACION')

>>> data = indicator.get_data(geo='I')

>>> data.columns
('Lanzarote',
 'Fuerteventura',
 'Gran Canaria',
 'Tenerife',
 'La Gomera',
 'La Palma',
 'El Hierro')
```

Query indicating to return raw codes:

```
>>> indicator = indicators.get_indicator('POBLACION')

>>> data = indicator.get_data(geo='=I')

>>> data.columns
('ES708', 'ES704', 'ES705', 'ES709', 'ES706', 'ES707', 'ES703')
```

Note: This can be used in time dimension as well.

API Reference

Functions

Indicator

IndicatorData

2.1.2 istacpy.indicators.geographic

istacpy.indicators.geographic.**get_indicators_geographic_granularities**()
Get geographic granularities

This function returns a list of geographic granularities treated in the ISTAC-indicators database. For example provincial, insular or municipal granularity.

Examples

```
>>> get_indicators_geographic_granularities()
```

istacpy.indicators.geographic.**get_indicators_geographical_values**(*geographicalgranularitycode*, *subjectcode=''*, *systemcode=''*)

Get geographical values

This function returns values of a geographical granularity that in turn are part of a specific theme or system of indicators.

Parameters

- **geographicalgranularitycode** (*string*) – geographical granularity code
- **subjectcode** (*string*) – subject code
- **systemcode** (*string*) – system code

Examples

```
>>> get_indicators_geographical_values("REGIONS")
>>> get_indicators_geographical_values(
... "REGIONS",
... subjectcode="051",
... systemcode="C00067A"
)
```

istacpy.indicators.geographic.**get_indicators_subjects**()

Get subjects

This function returns all subjects which the ISTAC classifies its statistical operations.

Examples

```
>>> get_indicators_subjects()
```

istacpy.indicators.geographic.**get_indicators_time_granularities**()

Get time granularities

This function returns a list of temporary granularity treated in the ISTAC data bank-indicators ordered from highest to lowest granularity. For example annual, quarterly or monthly granularity.

Examples

```
>>> get_indicators_time_granularities()
```

2.1.3 istacpy.indicators.indicators

Example 1: Get a list of all available indicators:

```
from istacpy.indicators import indicators
indicators.get_indicators()
```

Example 2: Get a list of geographic granularities treated in the ISTAC-indicators database. For example provincial, insular or municipal granularity:

```
from istacpy.indicators import geographic
geographic.get_indicators_geographic_granularities()
```

istacpy.indicators.indicators.**get_indicators**(*q=''*, *order=''*, *limit=25*, *offset=0*, *fields=''*, *representation=''*)

Get indicators

This function returns a list of indicators published in the ISTAC-indicators database. An indicator is a measure used to know the intensity of a phenomenon in spacetime. This measure can refer to different spatial or temporal granularities.

Parameters

- **q** (*string*) – Metadata query on which the searches can be built using `id`, `subjectCode` or `geographicValue`.
- **order** (*string*) – Order. Possible values are: `update` and `id`. Order criteria are `ASC` and `DESC`.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **fields** (*string*) – Use of the answer by adding new fields. Possible values are: `+metadata`, `+data` and `+observationsMetadata`.
- **representation** (*string*) – Allows filtering the observations by their value. Its use only makes sense when `+data` and/or `+observationsMetadata` has been included.

Examples

```
>>> get_indicators(
...     q='id IN ("AFILIACIONES", "EMPLEO_REGISTRADO_AGRICULTURA")',
...     order="id ASC",
...     fields="+data",
...     representation="GEOGRAPHICAL[35003|35005], MEASURE[ABSOLUTE]"
... )
```

istacpy.indicators.indicators.**get_indicators_code**(*indicatorcode*)

Get indicators code

This function returns the metadata that describe the characteristics of a specific indicator, allowing the compression of the measured fact; also through the data request the complete data (for all spacetime) of the indicator is provided.

Parameters **indicatorcode** (*string*) – an indicator code

Examples

```
>>> get_indicators_code("AFILIACIONES")
>>> get_indicators_code("PARO_REGISTRADO")
```

istacpy.indicators.indicators.**get_indicators_code_data**(*indicatorcode*, *representation=''*, *granularity=''*, *fields=''*)

Get indicators code data

This function returns complete data (for all spacetime) of the indicator. On the other hand, metadata describing the characteristics of a specific indicator are offered through the metadata request, allowing the compression of the measured fact.

Parameters

- **indicatorcode** (*string*) – an indicator code
- **representation** (*string*) – Allows filtering the observations by their value.
- **granularity** (*string*) – Allows to filter the observations through the granularities of the same.
- **fields** (*string*) – Allows you to customize the response by excluding fields. The possible values are: `-observationsMetadata`.

Examples

```
>>> get_indicators_code_data("AFILIACIONES")
```

2.1.4 istacpy.indicators.systems

istacpy.indicators.systems.**get_indicators_systems**(*limit=25*, *offset=0*)

Get indicators systems

This function returns a list of indicator systems published in the ISTAC-indicators database. The indicators are simple or compound statistics, however a single indicator can rarely provide useful information about complex phenomena such as the economic situation, living conditions, schooling or others. Indicator systems are generally designed to generate more and more accurate information about the conditions of a phenomenon; and for this they are organized in dimensions or areas of analysis, under which the indicators are integrated.

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.

Examples

```
>>> get_indicators_systems()
```

istacpy.indicators.systems.**get_indicators_systems_code**(*indicatorsystemcode*)

Get indicators system code

This function returns metadata of a system of indicators published in the ISTAC-indicators database. The indicators are simple or compound statistics, however a single indicator can rarely provide useful information about complex phenomena such as the economic situation, living conditions, schooling or others.

Parameters **indicatorsystemcode** (*string*) – an indicator system code

Examples

```
>>> get_indicators_systems_code("C00075H")
```

istacpy.indicators.systems.**get_indicators_systems_code_instances**(*indicatorsystemcode*, *q=''*, *order=''*, *limit=25*, *offset=0*, *fields=''*, *representation=''*, *granularity=''*)

Get indicators system code instances

This function returns instances of indicators associated with a specific indicator system. An instance of an indicator is nothing more than a spatio-temporal query of an indicator when it is incorporated into a specific indicator system.

Parameters

- **indicatorsystemcode** (*string*) – with an indicator system code
- **q** (*string*) – Query of metadata on which the searches can be built are: `id` and `geographicalValue`.
- **order** (*string*) – Order. Possible values are: `update` and `id` and order criteria are `ASC` and `DESC`.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **fields** (*string*) – Use of the answer by adding new fields. Possible values are: `+metadata`, `+data` and `+observationsMetadata`.
- **representation** (*string*) – Allows filtering the observations by their value. Its use only makes sense when `+data` and/or `+observationsMetadata` has been included.
- **granularity** (*string*) – Allows to filter the observations through the granularities of the same. Its use only makes sense when `+data` and/or `+observationsMetadata` has been included.

Examples

```
>>> get_indicators_systems_code_instances("C00075H")
>>> get_indicators_systems_code_instances(
...     "C00075H",
...     q='id EQ "INDICADORES_MUNICIPALES"'
... )
```

istacpy.indicators.systems.**get_indicators_systems_code_instances_code**(*indicatorsystemcode*, *indicatorinstancecode*)

Get indicators system code instances code

This function returns metadata of an indicator set associated with a specific indicator system. An instance of an indicator is nothing more than a spatio-temporal query of an indicator when it is incorporated into a specific indicator system.

Parameters

- **indicatorsystemcode** (*string*) –
- **indicatorinstancecode** (*string*) –

Examples

```
>>> get_indicators_systems_code_instances_code(
...     "C00075H",
...     "21af0477-d63b-493b-ad02-4ab181547223"
... )
```

istacpy.indicators.systems.**get_indicators_systems_code_instances_code_data**(*indicatorsystemcode*, *indicatorinstancecode*, *representation=''*, *granularity=''*, *fields=''*)

Get indicators system code instances code data

This function returns data of an indicator unit associated with a specific indicator system. An instance of an indicator is nothing more than a spatio-temporal query of an indicator when it is incorporated into a specific indicator system.

Parameters

- **indicatorsystemcode** (*string*) – Indicator system code
- **indicatorinstancecode** (*string*) – Indicator instance code
- **representation** (*string*) – Allows filtering the observations by their value.
- **granularity** (*string*) – Allows to filter the observations through the granularities of the same.
- **fields** (*string*) – Allows you to customize the response by excluding fields. The possible values are: `-observationsMetadata`.

Examples

```
>>> get_indicators_systems_code_instances_code_data(
...     "C00075H",
...     "21af0477-d63b-493b-ad02-4ab181547223"
... )
```

2.2 Statistical resources

2.2.1 Dataframe handling

Apart from all the functions below, there are **two special calls** we can use in order to get a pandas DataFrame instead of the usual json API response:

- *istacpy.statisticalresources.queries.get_statisticalresources_queries_agency_resource(*
- *istacpy.statisticalresources.cubes.get_statisticalresources_datasets_agency_resource_v*

Important: `pip install pandas` is required for this feature to work.

Guess you are looking for data about population under 18 in Canary Islands regarding its employment status. Once you get the *resource identifier* you could make this query:

```
>>> from istacpy.statisticalresources import queries

>>> response = queries.get_statisticalresources_queries_agency_resource(
...     agencyid='ISTAC',
...     resourceid='C00086B_000006',
...     as_dataframe=True
... )

>>> type(response)
istacpy.services.ResolvedAPIResponse
```

Note: Adding the parameter `as_dataframe=True` makes you get a `ResolvedAPIResponse` object.

This object provides two attributes: `dataframe` and `codelists`:

```
>>> response.dataframe
                MEDIDAS TIME_PERIOD SEXO      SITUACION_ECONOMICA_ICC VALORACION␣
↪OBSERVACIONES
0                POBLACION     2018-Q4   _T       ACTUAL_BUSQUEDA_EMPLEO         _T  ␣
↪    1791633
1                POBLACION     2018-Q4   _T       ACTUAL_BUSQUEDA_EMPLEO      IGUAL  ␣
↪     782350
2                POBLACION     2018-Q4   _T       ACTUAL_BUSQUEDA_EMPLEO      MEJOR  ␣
↪     272479
3                POBLACION     2018-Q4   _T       ACTUAL_BUSQUEDA_EMPLEO       PEOR  ␣
↪     736804
4                POBLACION     2018-Q4   _T  EXPECTATIVA_BUSQUEDA_EMPLEO         _T  ␣
↪    1791633
..                     ...         ...  ...                          ...        ...  ␣
↪        ...
283  POBLACION__PORCENTAJE     2020-Q4    M       ACTUAL_BUSQUEDA_EMPLEO       PEOR  ␣
↪      87.10
284  POBLACION__PORCENTAJE     2020-Q4    M  EXPECTATIVA_BUSQUEDA_EMPLEO         _T  ␣
↪     100.00
285  POBLACION__PORCENTAJE     2020-Q4    M  EXPECTATIVA_BUSQUEDA_EMPLEO      IGUAL  ␣
↪      19.25
286  POBLACION__PORCENTAJE     2020-Q4    M  EXPECTATIVA_BUSQUEDA_EMPLEO      MEJOR  ␣
↪      19.35
287  POBLACION__PORCENTAJE     2020-Q4    M  EXPECTATIVA_BUSQUEDA_EMPLEO       PEOR  ␣
↪      61.40

[288 rows x 6 columns]

>>> response.codelists
{'MEDIDAS': {'POBLACION': 'Población',
'POBLACION__PORCENTAJE': 'Población. Porcentaje'},
'TIME_PERIOD': {'2020-Q4': '2020 Cuarto trimestre',
'2020-Q2': '2020 Segundo trimestre',
'2019-Q4': '2019 Cuarto trimestre',
'2019-Q2': '2019 Segundo trimestre',
'2019-Q1': '2019 Primer trimestre',
'2018-Q4': '2018 Cuarto trimestre'},
'SEXO': {'_T': 'Ambos sexos', 'M': 'Hombres', 'F': 'Mujeres'},
'SITUACION_ECONOMICA_ICC': {'ACTUAL_BUSQUEDA_EMPLEO': 'Situación actual para␣
↪encontrar o mejorar un puesto de trabajo',
'EXPECTATIVA_BUSQUEDA_EMPLEO': 'Expectativa futura para encontrar o mejorar un puesto␣
↪de trabajo'},
'VALORACION': {'_T': 'Total',
'MEJOR': 'Mejor',
'IGUAL': 'Igual',
'PEOR': 'Peor'}}
```

Important: Codelists are mappings between codes and text descriptions. The returned `response.codelists` always contains **spanish texts** because they are most cases.

Recoding of downloaded dataset is as easy as:

```
>>> response.dataframe.replace(response.codelists)
                MEDIDAS            TIME_PERIOD         SEXO                          ␣
↪ SITUACION_ECONOMICA_ICC VALORACION OBSERVACIONES
```

(continues on next page)

(continued from previous page)

```
0               Población  2018 Cuarto trimestre  Ambos sexos  Situación actual para␣
↪encontrar o mejorar un p...      Total       1791633
1               Población  2018 Cuarto trimestre  Ambos sexos  Situación actual para␣
↪encontrar o mejorar un p...      Igual        782350
2               Población  2018 Cuarto trimestre  Ambos sexos  Situación actual para␣
↪encontrar o mejorar un p...      Mejor        272479
3               Población  2018 Cuarto trimestre  Ambos sexos  Situación actual para␣
↪encontrar o mejorar un p...       Peor        736804
4               Población  2018 Cuarto trimestre  Ambos sexos  Expectativa futura␣
↪para encontrar o mejorar un...      Total       1791633
..                    ...                   ...          ...                      ␣
↪                        ...        ...           ...
283  Población. Porcentaje  2020 Cuarto trimestre      Hombres  Situación actual para␣
↪encontrar o mejorar un p...       Peor         87.10
284  Población. Porcentaje  2020 Cuarto trimestre      Hombres  Expectativa futura␣
↪para encontrar o mejorar un...      Total        100.00
285  Población. Porcentaje  2020 Cuarto trimestre      Hombres  Expectativa futura␣
↪para encontrar o mejorar un...      Igual         19.25
286  Población. Porcentaje  2020 Cuarto trimestre      Hombres  Expectativa futura␣
↪para encontrar o mejorar un...      Mejor         19.35
287  Población. Porcentaje  2020 Cuarto trimestre      Hombres  Expectativa futura␣
↪para encontrar o mejorar un...       Peor         61.40

[288 rows x 6 columns]
```

Tip: This process can also be made for statistical datasets.

2.2.2 istacpy.statisticalresources.queries

istacpy.statisticalresources.queries.**get_statisticalresources_queries**(*lang='es'*, *limit=25*, *offset=0*, *orderby=''*, *query=''*)

Get queries

This function allows consulting all existing statistical queries.

Parameters

- **lang** (*string*) – Language in which you want to get the answer.
- **limit** (*int*) – Results limit. By default `limit=25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset=0`.
- **orderby** (*string*) – Order. Possible values are `ID ASC` or `ID DESC`,
- **query** (*string*) – Metadata query on which the searches can be built.

Examples

```
>>> get_statisticalresources_queries()
```

istacpy.statisticalresources.queries.**get_statisticalresources_queries_agency**(*agencyid*, *lang='es'*, *limit=25*, *offset=0*, *orderby=''*, *query=''*)

Get queries (agencyID)

This function allows to consult all the queries maintained by a certain organization.

Parameters

- **agencyid** (*string*) – Identifier of the maintainer organization of the resource. A possible value is `ISTAC`.
- **lang** (*string*) – Language in which you want to get the answer.
- **limit** (*int*) – Results limit. By default `limit=25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset=0`.
- **orderby** (*string*) – Order. Possible values are `ID ASC` or `ID DESC`,
- **query** (*string*) – Metadata query on which the searches can be built.

Examples

```
>>> get_statisticalresources_queries_agency(agencyid="ISTAC")
```

istacpy.statisticalresources.queries.**get_statisticalresources_queries_agency_resource**(*agency*, *resourceid*, *fields=*, *lang=*, *as_dat*

Get queries (agencyID / resourceID)

This function allows to obtain final data of a statistical query with a certain identifier and that also maintains a certain organization.

Parameters

- **agencyid** (*string*) – Identifier of the maintainer organization of the resource. A possible value is `ISTAC`.
- **resourceid** (*string*) – Resource identifier. A possible value is `C00010A_000002`.
- **fields** (*string*) – Allows you to customize the response by excluding fields from it. The possible values are `-metadata` and `-data`.
- **lang** (*string*) – Language in which you want to get the answer.

- **as_dataframe** (*bool*) – If True, this function returns a namedtuple with: - dataframe: pandas dataframe built from API response. - codelists: mapping between codes and representations for each column.

Examples

```
>>> get_statisticalresources_queries_agency_resource(
...     agencyid="ISTAC",
...     resourceid="C00017A_000001"
... )
```

2.2.3 istacpy.statisticalresources.cubes

istacpy.statisticalresources.cubes.**get_statisticalresources_datasets**(*lang='es'*, *limit=25*, *offset=0*, *orderby=''*, *query=''*)

Get datasets

This function allows consulting all existing statistical data cubes.

Parameters

- **lang** (*string*) – Language in which you want to get the answer.
- **limit** (*int*) – Results limit. By default `limit=25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset=0`.
- **orderby** (*string*) – Order. Possible values are `ID ASC` or `ID DESC`,
- **query** (*string*) – Metadata query on which the searches can be built.

Examples

```
>>> get_statisticalresources_datasets()
```

istacpy.statisticalresources.cubes.**get_statisticalresources_datasets_agency**(*agencyid*, *lang='es'*, *limit=25*, *offset=0*, *orderby=''*, *query=''*)

Get datasets (agencyID)

This function allows to consult all the data sets maintained by a certain organization.

Parameters

- **agencyid** (*string*) – Identifier of the maintainer organization of the resource. A possible value is `ISTAC`.

- **lang** (*string*) – Language in which you want to get the answer.
- **limit** (*int*) – Results limit. By default `limit=25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset=0`.
- **orderby** (*string*) – Order. Possible values are `ID ASC` or `ID DESC`,
- **query** (*string*) – Metadata query on which the searches can be built.

Examples

```
>>> get_statisticalresources_datasets_agency(agencyid="ISTAC")
```

istacpy.statisticalresources.cubes.**get_statisticalresources_datasets_agency_resource**(*agencyi*
re-
sour-
ceid,
lang='e
limit=2
off-
set=0,
or-
derby='
query='

Get datasets (agencyID / resourceID)

This function allows to obtain all the versions of a statistical cube with a certain identifier and that also maintains a certain organization.

Parameters

- **agencyid** (*string*) – Identifier of the maintainer organization of the resource. A possible value is `ISTAC`.
- **resourceid** (*string*) – Resource identifier. A possible value is `C00010A_000002`.
- **lang** (*string*) – Language in which you want to get the answer.
- **limit** (*int*) – Results limit. By default `limit=25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset=0`.
- **orderby** (*string*) – Order. Possible values are `ID ASC` or `ID DESC`,
- **query** (*string*) – Metadata query on which the searches can be built.

Examples

```
>>> get_statisticalresources_datasets_agency_resource(
...     agencyid="ISTAC",
...     resourceid="C00010A_000002"
... )
```

istacpy.statisticalresources.cubes.**get_statisticalresources_datasets_agency_resource_versio**

Get datasets (agencyID / resourceID / version)

This function allows to obtain all the versions of a statistical cube with a certain identifier and that also maintains a certain organization.

Parameters

- **agencyid** (*string*) – Identifier of the maintainer organization of the resource. A possible value is ISTAC.
- **resourceid** (*string*) – Resource identifier. A possible value is C00010A_000002.
- **version** (*string*) – Resource version. A possible value is 001.000.
- **dim** (*string*) – Allows filtering the data obtained in the response. A example is TIME_PERIOD:2009|2010.
- **fields** (*string*) – Allows you to customize the response by excluding fields from it. The possible values are -metadata and -data.
- **lang** (*string*) – Language in which you want to get the answer.
- **as_dataframe** (*bool*) – If True, this function returns a namedtuple with: - dataframe: pandas dataframe built from API response. - codelists: mapping between codes and representations for each column.

Examples

```
>>> get_statisticalresources_datasets_agency_resource_version(
...     agencyid="ISTAC",
...     resourceid="C00010A_000002",
...     version="001.000"
... )
```

2.3 Structural resources

Example 1: Get a list of classifications:

```
from istacpy.structuralresources import classifications
classifications.get_structuralresources_codelists()
```

Example 2: Get a list of geographic coordinate from Icod de los Vinos:

```
from istacpy.structuralresources import variables
variables.get_structuralresources_geoinfo('VR_TERRITORIO', 'MUN_ICOD_VINOS')
```

2.3.1 istacpy.structuralresources.category

istacpy.structuralresources.category.**get_structuralresources_categorisations**(*limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get categorisations

This function returns the content from `/v1.0/categorisations`

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_categorisations()
>>> get_structuralresources_categorisations(
...     query="ID EQ 2090",
...     orderby="ID ASC"
... )
```

istacpy.structuralresources.category.**get_structuralresources_categorisations_agency**(*agencyid*, *limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get categorisations agency

This function returns the content from `/v1.0/categorisations/{agencyID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_categorisations_agency("ISTAC")
```

istacpy.structuralresources.category.**get_structuralresources_categorisations_agency_resourc**

Get categorisations agency resource

This function returns the content from `/v1.0/categorisations/{agencyID}/{resourceID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_categorisations_agency_resource("ISTAC", "cat2")
```

istacpy.structuralresources.category.**get_structuralresources_categorisations_agency_resourc**

Get categorisations agency resource version

This function returns the content from `/v1.0/categorisations/{agencyID}/{resourceID}/{version}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **version** (*string*) – Specific version of the resource.

Examples

```
>>> get_structuralresources_categorisations_agency_resource_version(
...     "ISTAC",
...     "cat2",
...     "01.000"
... )
```

istacpy.structuralresources.category.**get_structuralresources_category_schemes**(*limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get category schemes

This function returns the content from `/v1.0/categoryschemes`

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_category_schemes()
>>> get_structuralresources_category_schemes(
...     query="ID EQ 2090",
...     orderby="ID ASC"
... )
```

istacpy.structuralresources.category.**get_structuralresources_category_schemes_agency**(*agencyi limit=2 offset=0, query=' orderby='*

Get category schemes agency

This function returns the content from `/v1.0/categoryschemes/{agencyID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_category_schemes_agency(
...     "ISTAC",
...     query="ID EQ 2090",
...     orderby="ID ASC"
...)
```

istacpy.structuralresources.category.**get_structuralresources_category_schemes_agency_resou**

Get category schemes agency resource

This function returns the content from `/v1.0/categoryschemes/{agencyID}/{resourceID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_category_schemes_agency_resource(
...     "ISTAC",
...     "TEMAS_CANARIAS"
... )
```

istacpy.structuralresources.category.**get_structuralresources_category_schemes_agency_resou**

Get category schemes agency resource version

This function returns the content from `/v1.0/categoryschemes/{agencyID}/{resourceID}/{version}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.

- **version** (*string*) – Specific version of the resource.

Examples

```
>>> get_structuralresources_category_schemes_agency_resource_version(
...     "ISTAC",
...     "TEMAS_CANARIAS",
...     "01.000"
... )
```

istacpy.structuralresources.category.**get_structuralresources_category_schemes_agency_resou**

Get category schemes agency resource version categories

This function returns the content from `/v1.0/categoryschemes/{agencyID}/{resourceID}/{version}/categories`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **version** (*string*) – Specific version of the resource.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_category_schemes_agency_resource_version_categories(
...     "ISTAC",
...     "TEMAS_CANARIAS",
...     "01.000"
... )
```

istacpy.structuralresources.category.**get_structuralresources_category_schemes_agency_resou**

Get category schemes agency resource version categories (id)

This function returns the content from `/v1.0/categoryschemes/{agencyID}/{resourceID}/{version}/categories/{categoryID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **version** (*string*) – Specific version of the resource.
- **categoryid** (*string*) – category identifier

Examples

```
>>> get_structuralresources_category_schemes_agency_resource_version_categories_
↪id(
...     "ISTAC",
...     "TEMAS_CANARIAS",
...     "01.000",
...     "060"
... )
>>> get_structuralresources_category_schemes_agency_resource_version_categories_
↪id(
...     "ISTAC",
...     "TEMAS_CANARIAS",
...     "01.000",
...     "060.060_010.060_010_010"
... )
```

2.3.2 istacpy.structuralresources.classifications

istacpy.structuralresources.classifications.**get_structuralresources_codelist_families**(*limit=*, *offset=0*, *orderby=*, *query=*

Get codelist families

This function returns the list of families of classifications

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.

- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **orderby** (*string*) – Field by which to sort the results.
- **query** (*string*) – Query to filter the results.

Examples

```
>>> get_structuralresources_codelist_families()
```

istacpy.structuralresources.classifications.**get_structuralresources_codelist_families_id**(*id*

Get codelist families

This function allows to obtain a family of classifications in particular.

Parameters **id** (*string*) – codelist family identificator

Examples

```
>>> get_structuralresources_codelist_families_id('CODELIST_ID')
```

istacpy.structuralresources.classifications.**get_structuralresources_codelists**(*limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get codelists

This function allows to obtain the list of classifications.

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_codelists()
```

istacpy.structuralresources.classifications.**get_structuralresources_codelists_agency**(*agencyi*
limit=2
offset=0,
query='
orderby='

Get codelists agency

This function allows obtaining the list of all the classifications maintained by a certain organization.

Parameters

- **agencyid** (*string*) – Agency identificator.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_codelists_agency("ISTAC")
>>> get_structuralresources_codelists_agency("ESTAT")
```

istacpy.structuralresources.classifications.**get_structuralresources_codelists_agency_resour**

Get codelists agency resource

This function allows to obtain all the versions of a classification with a certain identifier and that is also kept by a certain organization.

Parameters

- **agencyid** (*string*) – Agency identificator.
- **resourceid** (*string*) – Resource identificator.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_codelists_agency_resource("ISTAC", "CL_AREA_ES")
```

istacpy.structuralresources.classifications.**get_structuralresources_codelists_agency_resour**

Get codelists agency resource version

This function allows you to consult a particular version of a classification.

Parameters

- **agencyid** (*string*) – Agency identificator.
- **resourceid** (*string*) – Resource identificator.
- **version** (*string*) – Specific resource version.

Examples

```
>>> get_structuralresources_codelists_agency_resource_version(
...     "ISTAC",
...     "CL_AREA_ES",
...     "01.000"
... )
```

istacpy.structuralresources.classifications.**get_structuralresources_codelists_agency_resou**

Get codelists agency resource version codes

This function allows to consult the codes of a version of a classification. Note that if wildcards are used as `~all` or one of the `limit`, `offset`, `query` or `orderBy` parameters, the list will be automatically paginated.

Parameters

- **agencyid** (*string*) – Agency identificator.
- **resourceid** (*string*) – Resource identificator.
- **version** (*string*) – Specific resource version.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.
- **openness** (*string*) – Opening established for viewing.
- **order** (*string*) – Order established for visualization.
- **fields** (*string*) – Additional fields that you want to show in the answer.

Examples

```
>>> get_structuralresources_codelists_agency_resource_version_codes(
...     "ISTAC",
...     "CL_AREA_ES",
...     "01.000"
... )
```

istacpy.structuralresources.classifications.**get_structuralresources_codelists_agency_resou**

Get codelists agency resource version codes (codeID)

This function allows to consult a specific code of a version of a classification.

Parameters

- **agencyid** (*string*) – Agency identificator.
- **resourceid** (*string*) – Resource identificator.
- **version** (*string*) – Specific resource version.
- **codeid** (*string*) – Code identificator.

Examples

```
>>> get_structuralresources_codelists_agency_resource_version_codes_codeid(
    "ISTAC", "CL_AREA_ES", "01.000", "ES706A01")
```

2.3.3 istacpy.structuralresources.concepts

istacpy.structuralresources.concepts.**get_structuralresources_concept_schemes**(*limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get concept schemes

This function returns the content from `/v1.0/conceptschemes`

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_concept_schemes()
>>> get_structuralresources_concept_schemes(
...     query="ID EQ 2090",
...     orderby="ID ASC"
... )
```

istacpy.structuralresources.concepts.**get_structuralresources_concept_schemes_agency**(*agencyid*, *limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get concept schemes agency

This function returns the content from `/v1.0/conceptschemes/{agencyID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_concept_schemes_agency("ISTAC")
>>> get_structuralresources_concept_schemes_agency(
...     "ESTAT",
...     query="ID EQ 2090",
...     orderby="ID ASC"
... )
```

istacpy.structuralresources.concepts.**get_structuralresources_concept_schemes_agency_resourc**

Get concept schemes agency resource

This function returns the content from `/v1.0/conceptschemes/{agencyID}/{resourceID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.

- **resourceid** (*string*) – Resource identifier.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_concept_schemes_agency_resource(
...     agencyid="ISTAC",
...     resourceid="CSM_C00010A_SIE"
... )
```

istacpy.structuralresources.concepts.**get_structuralresources_concept_schemes_agency_resourc**

Get concept schemes agency resource version

This function returns the content from `/v1.0/conceptschemes/{agencyID}/{resourceID}/{version}`

Parameters

- **agencyid** (*string*) –
- **resourceid** (*string*) –
- **version** (*string*) –

Examples

```
>>> get_structuralresources_concept_schemes_agency_resource_version(
...     gencyid="ISTAC",
...     resourceid="CSM_C00010A_SIE",
...     version="01.000"
... )
```

istacpy.structuralresources.concepts.**get_structuralresources_concept_schemes_agency_resourc**

Get concept schemes agency resource version concepts

This function returns the content from `/v1.0/conceptschemes/{agencyID}/{resourceID}/{version}/concepts`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **version** (*string*) – Specific version of the resource.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.
- **fields** (*string*) – Additional fields that you want to show in the answer.

Examples

```
>>> get_structuralresources_concept_schemes_agency_resource_version_concepts(
...     agencyid="ISTAC",
...     resourceid="CSM_C00010A_SIE",
...     version="01.000"
... )
```

istacpy.structuralresources.concepts.**get_structuralresources_concept_schemes_agency_resourc**

Get concept schemes agency resource version concepts (id)

This function returns the content from `/v1.0/conceptschemes/{agencyID}/{resourceID}/{version}/concepts/{conceptID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **version** (*string*) – Specific version of the resource.
- **conceptid** (*string*) – Concept identifier.

Examples

```
>>> get_structuralresources_concept_schemes_agency_resource_version_concepts_id(
...     agencyid="ISTAC",
...     resourceid="CSM_C00010A_SIE",
...     version="01.000",
...     conceptID="ELECTORES"
... )
```

istacpy.structuralresources.concepts.**get_structuralresources_concept_types**()

Get concept types

This function returns the content from `/v1.0/conceptTypes`

Examples

```
>>> get_structuralresources_concept_types()
```

2.3.4 istacpy.structuralresources.datastructures

istacpy.structuralresources.datastructures.**get_structuralresources_content_constraints**(*limit=*, *offset=0*, *query=*, *orderby=*)

Get content constraints

This function returns the content from `/v1.0/contentConstraints`

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_content_constraints()
>>> get_structuralresources_content_constraints(
...     query="ID EQ 2090",
...     orderby="ID ASC"
... )
```

istacpy.structuralresources.datastructures.**get_structuralresources_content_constraints_agen**

Get content constraints agency

This function returns the content from `/v1.0/contentConstraints/{agencyID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_content_constraints_agency("ISTAC")
```

istacpy.structuralresources.datastructures.**get_structuralresources_content_constraints_agen**

Get content constraints agency resource

This function returns the content from `/v1.0/contentConstraints/{agencyID}/{resourceID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_content_constraints_agency_resource(
...     "ISTAC",
...     "CSM_C00010A_SIE"
... )
```

istacpy.structuralresources.datastructures.**get_structuralresources_content_constraints_agen**

Get content constraints agency resource version

This function returns the content from `/v1.0/contentConstraints/{agencyID}/{resourceID}/{version}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **version** (*string*) – Specific version of the resource.

Examples

```
>>> get_structuralresources_content_constraints_agency_resource_version(
...     "ISTAC",
...     "CSM_C00010A_SIE",
...     "01.000"
... )
```

istacpy.structuralresources.datastructures.**get_structuralresources_content_constraints_agen**

Get content constraints agency resource version regions

This function returns the content from `/v1.0/contentConstraints/{agencyID}/{resourceID}/{version}/regions/` {regionCode}

Parameters

- **regioncode** (*string*) –
- **agencyid** (*string*) –
- **resourceid** (*string*) –
- **version** (*string*) –

Examples

```
>>> get_structuralresources_content_constraints_agency_resource_version_regions(
...     "0001",
...     "ISTAC",
...     "CSM_C00010A_SIE",
...     "01.000"
... )
```

istacpy.structuralresources.datastructures.**get_structuralresources_data_structures**(*limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get data structures

This function returns the content from `/v1.0/datastructures`

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_data_structures()
>>> get_structuralresources_data_structures(
...     query="ID EQ 2090",
...     orderby="ID ASC"
... )
```

istacpy.structuralresources.datastructures.**get_structuralresources_data_structures_agency**(

Get data structures agency

This function returns the content from `/v1.0/datastructures/{agencyID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.

- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_data_structures_agency("ISTAC")
```

istacpy.structuralresources.datastructures.**get_structuralresources_data_structures_agency_**

Get data structures agency resource

This function returns the content from `/v1.0/datastructures/{agencyID}/{resourceID}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_data_structures_agency_resource(
...     "ISTAC",
...     "DSD_C00010A_00001"
... )
```

istacpy.structuralresources.datastructures.**get_structuralresources_data_structures_agency_**

Get data structures agency resource version

This function returns the content from `/v1.0/datastructures/{agencyID}/{resourceID}/{version}`

Parameters

- **agencyid** (*string*) – Identifier of the agency that publishes.
- **resourceid** (*string*) – Resource identifier.

- **version** (*string*) – Specific version of the resource.

Examples

```
>>> get_structuralresources_data_structures_agency_resource_version(
...     "ISTAC",
...     "DSD_C00010A_00001",
...     "01.001"
... )
```

2.3.5 istacpy.structuralresources.variables

istacpy.structuralresources.variables.**get_structuralresources_geoinfo**(*variableid*, *resourceid*, *fields=''*, *limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get geoinfo

This function returns data from `/v1.0/variables/{variableID}/variableelements/{resourceID}/geoinfo`

Parameters

- **variableid** (*string*) – Variable identificator.
- **resourceid** (*string*) – Resource identificator.
- **fields** (*string*) – Additional fields that you want to show in the answer.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_geoinfo("VR_TERRITORIO", "MUN_ICOD_VINOS")
```

istacpy.structuralresources.variables.**get_structuralresources_variable_families**(*limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get variable families

This function returns data from `/v1.0/variablefamilies`

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_variable_families()
```

istacpy.structuralresources.variables.**get_structuralresources_variable_families_id**(*id*)

Get variable families (id)

This function returns data from `/v1.0/variablefamilies/{id}`

Parameters **id** (*string*) – Variable family identificator.

Examples

```
>>> get_structuralresources_variable_families_id("VRF_DEMOGRAFICAS")
```

istacpy.structuralresources.variables.**get_structuralresources_variable_families_id_variable**

Get variable families (id) variables

This function returns data from `/v1.0/variablefamilies/{id}/variables`

Parameters

- **id** (*string*) – Variable family identificator.
- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_variable_families_id_variables("VRF_DEMOGRAFICAS")
```

istacpy.structuralresources.variables.**get_structuralresources_variableelements**(*variableid*, *limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get variableelements

This function returns data from `/v1.0/variables/{variableID}/variableelements`

Parameters

- **variableid** (*string*) –
- **limit** (*int*) –
- **offset** (*int*) – `offset = 0`.
- **query** (*string*) –
- **orderby** (*string*) –

Examples

```
>>> get_structuralresources_variableelements("VR_SEXO")
```

istacpy.structuralresources.variables.**get_structuralresources_variableelements_resource**(*variableid*, *resourceid*)

Get variableelements resource

This function returns data from `/v1.0/variables/{variableID}/variableelements/{resourceID}`

Parameters

- **variableid** (*string*) – Resource identificator.
- **resourceid** (*string*) – Variable identificator.

Examples

```
>>> get_structuralresources_variableelements_resource("VR_SEXO", "FEMALE")
```

istacpy.structuralresources.variables.**get_structuralresources_variables**(*limit=25*, *offset=0*, *query=''*, *orderby=''*)

Get variables

This function returns data from `/v1.0/variables`

Parameters

- **limit** (*int*) – Results limit. By default `limit = 25`.
- **offset** (*int*) – Displacement. Result from which it is returned. By default `offset = 0`.
- **query** (*string*) – Query to filter the results.
- **orderby** (*string*) – Field by which to sort the results.

Examples

```
>>> get_structuralresources_variables()
```

istacpy.structuralresources.variables.**get_structuralresources_variables_id**(*id*)

Get variables (id)

This function returns data from `/v1.0/variables/{id}`

Parameters **id** (*string*) – Variable identifier.

Examples

```
>>> get_structuralresources_variables_id("VR_SEXO")
```

2.4 Error handling

2.4.1 Custom exceptions

There are a bunch of custom exceptions defined in the package:

exception istacpy.exceptions.**GranularityNotAvailableError**(*value=''*, *message='given granularity{}is not available for this indicator'*)

exception istacpy.exceptions.**IndicatorNotFoundError**(*value=''*, *message='given indicator{}not found on API'*)

exception istacpy.exceptions.**IslandNotFoundError**(*value=''*, *message='given island{}not found'*)

exception istacpy.exceptions.**IstacPyError**(*value=''*, *message=''*)

exception istacpy.exceptions.**MeasureNotAvailableError**(*value=''*, *message='given measure{}is not available for this indicator'*)

exception istacpy.exceptions.**QueryMalformedError**(*value=''*, *message='given query{}is malformed or not properly written'*)

On the shoulder of gigants

Special mention to API requests handling. Behind the scenes, this package uses well-known requests package to retrieve data from API. Different errors can happen and they are raised to be captured for the user.

It's important to say that these exceptions will include a custom field called `requested_url` that let's the user to handle the API url.

2.4.2 Debug mode

Debug mode can be enabled using the proper function:

```
from istacpy import services
services.set_debug()
```

Amongh other side effects, when debug mode is enabled, API urls are displayed when a query is performed:

```
>>> indicators.get_subjects()
https://datos.canarias.es/api/estadisticas/indicators/v1.0/subjects
(('011', 'Territorio y usos del suelo'),
 ('012', 'Medio ambiente'),
 ('021', 'Población'),
 ('022', 'Movimiento natural'),
 ('023', 'Movimientos migratorios'),
 ('031', 'Calidad de vida'),
 ('033', 'Educación'),
 ('036', 'Justicia y seguridad'),
 ('041', 'Cuentas económicas'),
 ('042', 'Precios, consumo e inversión'),
 ('043', 'Empresas y centros de trabajo'),
 ('051', 'Empleo'),
 ('061', 'Agricultura, ganadería, pesca y caza'),
 ('071', 'Industria, energía y agua'),
 ('072', 'Construcción y vivienda'),
 ('080', 'SECTOR SERVICIOS'),
 ('081', 'Comercio'),
 ('082', 'Hostelería y turismo'),
 ('083', 'Transporte y comunicaciones'),
 ('084', 'Servicios financieros, monetarios y seguros'),
 ('091', 'Administración pública'))
```

Debug can be disabled as well. For that end, you can use the function `services.set_nodebug()`.

PYTHON MODULE INDEX

i

INDEX

G

M

Q