
ClinVar This! Documentation

Release 0.14.6

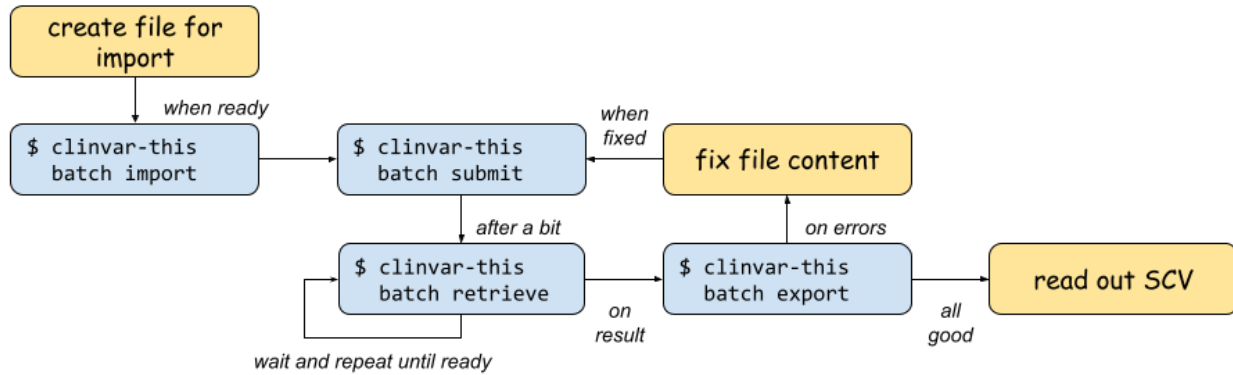
Manuel Holtgrewe

Jan 08, 2024

CONTENTS

1	Installation	3
2	Getting Started	5
3	API vs. CLI	9
4	Library Usage	11
5	Command Line Usage	13
6	File Formats	17
7	Limitations	23
8	Common Errors	25
9	Contributing	27
10	Credits	31
11	Changelog	33
12	Command Line Interface	41
13	clinvar_api	49
14	clinvar_this	107
15	Indices and tables	109
	Python Module Index	111
	Index	113

clinvar-this is a Python-based software for the easy submission of variants to ClinVar.



When reading this online, you can use the navigation bar to the left to find your way around. Otherwise, the following are good entry points:

- *Getting Started*
- *API vs. CLI*
- *Command Line Usage*

INSTALLATION

1.1 Stable Release (Pip)

To install ClinVar This!, run this command in your terminal:

```
$ pip install clinvar-this
```

This is the preferred method to install ClinVar This!, as it will always install the most recent stable release.

If you don't have `pip` installed, this [Python installation guide](#) can guide you through the process.

1.2 Stable Release (Conda)

As a prerequisite, you have to install conda and setup the bioconda channel [as documented on the Bioconda website](#).

Then, you can create a new environment with `clinvar-this`.

```
$ conda create -y -n clinvar-this python=3.11 clinvar-this
```

Or install the package in your current environment:

```
$ conda install -y clinvar-this
```

1.3 From Sources

The sources for ClinVar This! can be downloaded from the [Github repo](#).

Clone the public repository:

```
$ git clone https://github.com/bihealth/clinvar-this.git
```


GETTING STARTED

This section describes how to get started with `clinvar-this`. You will need some experience with VCF files, ClinVar, and the Linux/Mac command line.

2.1 Obtain ClinVar API Key

First of all, you need to register your organisation with NCBI, request a service account, and obtain an API key. Skip any step if you have already completed it.

1. Register your organisation with NCBI as they document
2. Send an email to clinvar@ncbi.nlm.nih.gov to request a service account for your organisation.
3. Once you have a service account, create an API key as outlined [at the top of the NCBI ClinVar API documentation](#).

2.2 Install `clinvar-this`

You can either install the PyPi package `clinvar-this`:

```
# pip install clinvar-this
```

Or you install via `conda/bioconda`:

```
# conda install -c clinvar-this
```

Check that your installation worked:

```
# clinvar-this --help
Usage: clinvar-this [OPTIONS] COMMAND [ARGS]...

Main entry point for CLI via click.

Options:
--verbose / --no-verbose
--profile TEXT           The profile to use
--help                  Show this message and exit.

Commands:
batch  Sub comment category ``batch ...``
config Sub command category ``varfish-this config ...``
```

2.3 Configure your API Token

```
# clinvar-this config set auth_token YOUR_AUTH_TOKEN_HERE
```

Check that this worked:

```
# clinvar-this config dump
# path: /home/holtgrem_c/.config/clinvar-this/config.toml
[default]
auth_token = "YOUR_AUTH_TOKEN_HERE"
```

2.4 Prepare a clinvar-this TSV file

You will need the following header in the first line. Mandatory columns have been marked with a (*). See [File Formats](#) for further format details and instructions for structural variants.

- **ASSEMBLY *** - the assembly used, e.g., GRCh37, hg19, GRCh38, hg38
- **CHROM *** - the chromosomal position without chr prefix, e.g., 1
- **POS *** - the 1-based position of the first base in REF column
- **REF *** - the reference allele of your variant
- **ALT *** - the alternative allele of your variant
- **CONDITION *** - ids for the carrier's condition, currently OMIM, ORPHA, MONDO ids are supported, give these as OMIM|ORPHA|MONDO:123456, e.g. OMIM:123456 is a valid id, ORPHA12345 is not. Can be left empty or use not provided if no condition should be published.
- **MOI *** - mode of inheritance, e.g., Autosomal dominant inheritance or Autosomal recessive inheritance
- **CLIN_SIG *** - clinical significance, e.g. Pathogenic, or Likely benign
- **CLIN_EVAL** - optional, date of late clinical evaluation, e.g. 2022-12-02, leave empty to fill with the date of today
- **CLIN_COMMENT** - optional, a comment on the clinical significance, e.g., ACMG Class IV; PS3, PM2_sup, PP4
- **KEY** - optional, a local key to identify the variant/condition pair. Filled automatically with a UUID if missing, recommended to leave empty.
- **HPO** - List of HPO terms separated by comma or semicolon, any space will be stripped. E.g., HP:0004322; HP:0001263.
- **PMID** - List of literature references, give these as multiple numbers separated by either , or ;. These will be listed as literature references for the variant interpretation.
- **ACCESSION** - Give an SCV-identifier, if this variant has already been previously uploaded by this institution and should be updated.

The following shows an example.

ASSEMBLY	CHROM	POS	REF	ALT	CONDITION	MOI	CLIN_SIG	HPO
GRCh37	19	48183936		C	CA	OMIM:619325	Autosomal	
→dominant		inheritance		Likely	pathogenic	HP:0004322;HP:0001263		

Note that you must use TAB characters (`\t`) for separating the file.

2.5 Import the TSV file into clinvar-this

Use the `batch import` command to import the TSV file into the local `clinvar-this` storage.

```
# clinvar-this batch import --name=BATCHNAME DATA_FILE.tsv
```

If you do not specify the `--name` parameter then `clinvar-this` will generate one based on the current time. This will create a new batch storage folder below `~/.local/share/clinvar-this/default` with the batch name and place a file `payload.$timestamp.json` there. This corresponds to the data that will be uploaded into ClinVar.

You can now import another TSV file or change your TSV file and re-import it to apply the changes.

2.6 Submit via ClinVar API

Use `batch submit BATCHNAME` to submit the data to the ClinVar API.

```
# clinvar-this batch submit BATCHNAME
```

This will create a new file `submission-response.$timestamp.json` in the batch storage folder. This file stores the identifier of the ClinVar submission. This information is subsequently used in `batch retrieve`.

2.7 Retrieve ClinVar API Submission Result

You can now use the following command to query the ClinVar API for the status of your submission.

```
# clinvar batch retrieve BATCHNAME
```

It will get the submission ID from the latest `submission-response.*.json` file (using lexicographic file name comparison) and query the ClinVar API. The API response will be written to `retrieve-response.$timestamp.json`. In the case that the API has processed your submission, `clinvar-this` will create a new `payload.$timestamp.json` file to reflect the change. You will probably have to wait a few or many minutes until the processing finishes. This will store any error message or ClinVar SCV.

2.8 Obtain SCV or Error Message

You could now look at the `payload.$timestamp.json` file to see the full server response. It is more convenient, however, to export the results to a TSV file again which will display the SCV identifiers and any error message:

```
# clinvar-this batch export BATCHNAME DATA_FILE.reply.tsv
```

The [ClinVar API documentation](#) says that variants submitted via the API do not have to pass manual curation. That is, the server will perform a number of checks. If your variants pass all checks then you will directly obtain an SCV and the variants will become publically available on the next Sunday.

2.9 Rinse and Repeat

In the case of a partial success, update the exported TSV file and submit it again until you are happy.

API VS. CLI

The `clinvar-this` software provides two entry points: API and CLI. In case you are not certain what you want/need, this section may help you.

3.1 Application Programmable Interface (API)

A Python module `clinvar_api` that you can use for making calls to the ClinVar submission API. **If you want to integrate ClinVar API submission into your Python software, this is for you.**

The module provides a “Pythonic” API based on `pydantic` with `snake_case` syntax that has full Python type annotations. Of course, you could just roll your own JSON based submission based on `requests` or `httpx` but using the module has some advantages:

- `clinvar_api` is fully typed so you can work with Python data types and all advantages that come this this (linting, editor completion, ...)
- the module authors monitor the ClinVar API docs and are on the relevant mailing lists and will adjust the library in case of API changes
- we provide full JSON schema validation of the submitted and received messages so you don’t have to.

Further, `clinvar_api`’s got what plants crave, it’s got electrolytes.

3.2 Command Line Interface (CLI) Application

Having an API library sounds great but you just want a standalone tool for submission? Then, the CLI `clinvar-this` is for you.

This software package allows you to create ClinVar submissions (and update or delete them). `clinvar-this` has a local repository of ClinVar submissions.

1. Create a submission in the local repository by importing one of the supported file formats (e.g., a very simple TSV/spreadsheet table). This will run some local sanity checks on your data so you can catch errors early on. `clinvar-this` uses the `clinvar_api` module so pitfalls such as JSON schema problems are circumvented.
2. Post the submission to the ClinVar API. The ClinVar will now process your submission which can take one or two hours.
3. Query the current processing status via the API.
4. Once complete, retrieve the ClinVar SCV identifiers (e.g., for your publication). On errors, retrieve the error messages, export back into a TSV/spreadsheet file for correction, and go to step 1.

Even if you plan to integrate ClinVar directly via the API, the `varfish-cli` package might be of interest to you to see how the general mechanics work of ClinVar API submission work. And if you don't work in Python, you can hopefully learn how to add ClinVar API submission support to your software stack.

LIBRARY USAGE

You will be most interested in using the API client library from the `clinvar_api` Python package:

```
import clinvar_api
```


COMMAND LINE USAGE

This section provides more “reference manual” style documentation when compared with *Getting Started*.

After installation, you can run `clinvar-this`:

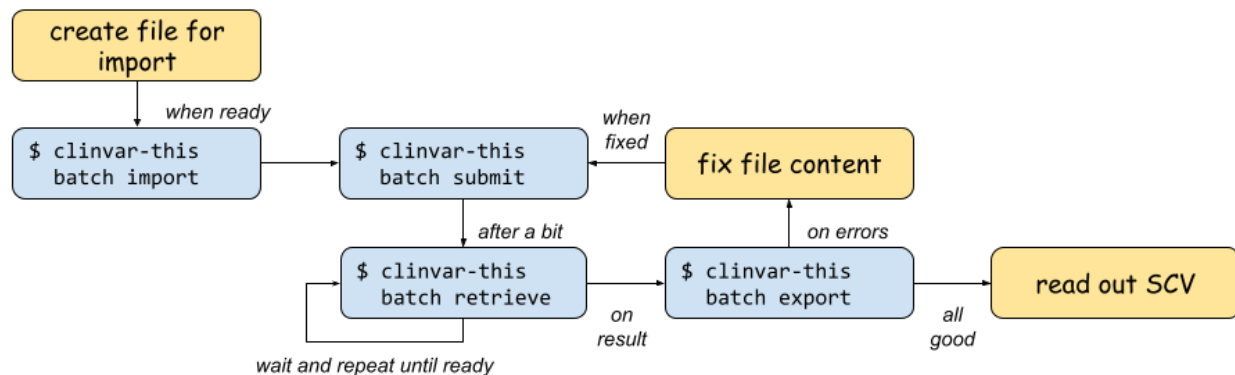
```
$ clinvar-this --help

usage: clinvar-this [-h] [--verbose]

options:
  -h, --help  show this help message and exit
  --verbose   Enable more verbose output
```

5.1 Workflow

The following figure shows the overall workflow when using `clinvar-this`.



- You start out by creating a file for import.
- You then import the data into a local repository batch with `clinvar-this batch import`.
- You can then post the data to the ClinVar API with `clinvar-this batch submit`.
- The server will process your data in the background. You can query the current result with `clinvar-this batch retrieve`. If this does not return yet, try again.
- Otherwise, you can export the current state of the batch with `clinvar-this batch export` to a TSV file.
- When there are errors, fix the variants to be submitted and re-submit with `clinvar-this batch submit`.
- If everything runs to your liking, read out your ClinVar SCV identifier.

Note that the NCBI ClinVar server process runs the checks in several steps. If an earlier step fails, you will not see the results of later checks. Also, when processing runs longer this means that more steps succeeded so waiting longer is a good thing.

5.2 ClinVar Terminology

ClinVar is an immensely useful and large resource. The most useful upstream documentation by NCBI ClinVar includes:

- [ClinVar Identifiers Documentation](#)
- [ClinVar FAQ](#)
- [ClinVar Submission FAQ](#)

A key concept in ClinVar is that it is **variant-centric**. That is, if you observe one variant in multiple samples (for the same condition! which may be an OMIM identifier or “not reported”!) it is one ClinVar record and ClinVar consider this record from your organisation as one submission. For your submission, you obtain a so-called SCV identifier and this is what you need for publications etc. ClinVar has these nice star ratings and these do not incorporate information on how many samples you report but only whether or not you apply formal assertion criteria such as the ACMG criteria or a local formal list of criteria. By yourself, you can only generate one star by providing an assertion criteria.

ClinVar then aggregates all submissions by all organisations for a given variant in a given condition (as explained above, yes the exclamation marks were intentional) into a reference record with an RCV record. Aggregation is based on conflicting interpretations and whether the submitters applied formal criteria (cf. [nudging](#)). In the case of multiple submitters providing conflict-free interpretations and assertion criteria, two stars may be gained. Submitters that don’t provide assertion criteria are overruled by those who do.

ClinVar will further aggregate all reference records into a variant record with a VCV identifier. For example, for BRCA variants all of these different tumor-related disorders will be thrown together.

Note that nothing in the above talked about multiple variants. You can *submit* them together via the *NCBI submission API* but each of these variants will be one **NCBI ClinVar submission** from your organisation. The software package *clinvar-this* (completely independent of NCBI and developed on the other side of the Atlantic) calls a list of variants to be submitted (as one submission each) a **batch**.

All of this hopefully leaves you less confused as before.

5.3 Configuration

The configuration will be stored in `~/.config/clinvar-this/config.toml` in [TOML format](#). The file can have multiple sections, each one configuring a **profile**. You should probably configure a **default** profile. You can set values using `clinvar-this config set NAME VALUE` and read values with `clinvar-this config get NAME`. A minimal configuration file looks as follows:

```
[default]
auth_token = "01234567890abcdefghijklmnopqrstuvwxyz0987654321"
```

Before you can use *clinvar-this* for the first time, you have to configure the API token to use with the ClinVar submission API.

```
$ clinvar-this set auth_token YOURTOKENHERE
```

Note that configuration values will be shown in full when using `varfish-cli config get/set`. Subsequently when using the tool for API submission, it will only show the first 5 characters of the secret key. This allows to determine

whether the right key is used but the value is safe enough to go to local log files etc. However, you should still ensure to take appropriate care when exposing these 5 first characters as applicable.

5.4 Local Repository

clinvar-this creates a local repository of data in `~/.local/share/clinvar-this/$profile` where `$profile` is the name of the profile that you use. Below this path, you will find one directory for each submission that you manage. Each such submission directory contains the following files:

payload.\$timestamp.json

The payload (to be) sent to ClinVar API server at the given timestamp. The lexicographically largest file is the latest one.

On each import, a new payload file will be created. If a previous one exists, the latest one will be merged with the new to-be-imported data. Also, when the NCBI server returns SCV identifiers on success or failures, this information will be stored in a new payload file.

You can safely manipulate these JSON files but that will require some knowledge about the ClinVar API format. However, it really is not hard and with some ClinVar/bioinformatics experience, you will be able to figure it out.

submission-response.\$timestamp.json

The response returned by the ClinVar API server returned at the given timestamp on submission. There is no direct correlation between the payload and submission response files at the moment, but you probably can figure it out based on the timestamp.

retrieve-response.\$timestamp.json

The response returned by the ClinVar API server when calling `batch retrieve BATCH`.

You can specify submission names when creating them (which is recommended). Otherwise, a name will be created for you based on the current date and time.

5.5 File Formats

See the dedicated section *File Formats*.

5.6 Submission Types

The following is written with the native TSV file format in mind. This translates to the other known file formats in the case that the *File Formats* describes the relevant columns/information.

5.6.1 Novel Submissions

If your sample sheet does not have a `clinvar_accession` column or it is empty for your variant, the variant will be submitted as novel. ClinVar will check whether your organisation has submitted this variant before for the same condition (OMIM code or “not provided”) and report back errors if one such record exists.

clinvar-this will write the SCV from the clinvar processing results to its local repository. On re-submission of the batch after processing and result retrieval, the variant will be submitted as an update.

5.6.2 Submission Updates

If you provided a `clinvar_accession` then `clinvar-this` will submit an update. Such a variant must already exist from your organisation for the given condition (again, OMIM code or “not provided”).

5.6.3 Record Removals

Removals have to take another path. You have to create a removal TSV file as documented in [File Formats](#) (you only have to provide the SCV identifier to delete for and a free-text comment), import it into a new `clinvar-this` batch and submit it.

Note that ClinVar refers to this as “deletion” but we refer as this to “removal” to have one less term collision to the meaning of “sequence deletion”.

FILE FORMATS

This section documents the supported file formats that can be imported by `batch import`. In the case of the non-native TSV format, this section documents how the file format maps to the information handled and stored by `clinvar-this`.

Overall, the aim of `clinvar-this` is to support you in submitting data easily with restrictions (see [Limitations](#)). If you need the full functionality of the NCBI ClinVar API then please consider using the `clinvar_api` Python module.

6.1 Sequence Variant TSV (Native)

The following headers are required. Clinvar-this will recognize the TSV file format based on these headers.

- **ASSEMBLY** - the assembly used, e.g., GRCh37, hg19, GRCh38, hg38
- **CHROM** - the chromosomal position without `chr` prefix, e.g., 1
- **POS** - the 1-based position of the first base in REF column
- **REF** - the reference allele of your variant
- **ALT** - the alternative allele of your variant
- **CONDITION** - the OMIM id of the carrier's condition (not the OMIM gene ID), e.g., 619325, alternatively also MONDO, ORPHA and HPO-Terms are supported, if multiple conditions are given, a multiple condition qualifier (Co-occurring, Uncertain, Novel disease) should also be given as an additional term. Leave empty or use `not provided` if you have no OMIM ID.
- **MOI** - mode of inheritance, e.g., Autosomal dominant inheritance or Autosomal recessive inheritance
- **CLIN_SIG** - clinical significance, e.g. Pathogenic, or Likely benign

The following headers are optional:

- **clinvar_accession** - ClinVar SCV accession if any exists yet. When this is set then this variant will be updated in the batch rather than added as a novel variant.
- **CLIN_EVAL** - date of last clinical evaluation, e.g. 2022-12-02, leave empty to fill with the date of today
- **CLIN_COMMENT** - a comment on the clinical significance, e.g., ACMG Class IV; PS3, PM2_sup, PP4
- **KEY** - a local key to identify the variant/condition pair. Filled automatically with a UUID if missing, recommended to leave empty.
- **HPO** - List of HPO terms separated by comma or semicolon, any space will be stripped. E.g., HP:0004322; HP:0001263.
- **PMID** - List of Pubmed IDs separated by a comma or semicolon, any space will be stripped. E.g., 31859447,29474920.

- **ACCESSION** - Existing clinvar SCV for this variant. This should only be set if the submitters organization has already uploaded the variant for the same condition before.
- **\$remove_from_batch** - you can use this for removing a previously added variant from the given batch; one of true and false, defaults to false.

Any further header will be imported into the local repository into an **extra_data** field. Note that the error returned by ClinVar for your variant will be written to a **error_msg** field.

The following shows an example.

ASSEMBLY	CHROM	POS	REF	ALT	CONDITION	MOI	CLIN_SIG	HPO
GRCh37	19	48183936		C	CA	OMIM:619325	Autosomal	
→dominant		inheritance		Likely	pathogenic	HP:0004322;HP:0001263		

Note that you cannot submission TSV imports with batches that contain removals already.

6.2 Structural Variant TSV (Native)

The following headers are required. Clinvar-this will recognize the TSV file format based on these headers.

- **ASSEMBLY** - the assembly used, e.g., GRCh37, hg19, GRCh38, hg38
- **CHROM** - the chromosomal position without chr prefix, e.g., 1
- **START** - the 1-based position
- **STOP** - the 1-based end position
- **SV_TYPE** - the type of the structural variant; one of Insertion, Deletion, Duplication, Tandem duplication, copy number loss, copy number gain, Inversion, Translocation, Complex. Note that ClinVar does not allow you to specify the second end of a non-linear event (e.g., a fusion with another chromosome). We suggest that you submit a second SV entry with the coordinate and link the two events in **CLIN_COMMENT**.
- **CONDITION** - the OMIM id of the carrier's condition (not the OMIM gene ID), e.g., OMIM:619325, alternatively also MONDO, ORPHA and HPO-Terms are supported, if multiple conditions are given, a multiple condition qualifier (Co-occurring, Uncertain, Novel disease) should also be given as an additional term.
- **MOI** - mode of inheritance, e.g., Autosomal dominant inheritance or Autosomal recessive inheritance
- **CLIN_SIG** - clinical significance, e.g. Pathogenic, or Likely benign

Note that instead of **START** and **STOP** you can also provide the following headers. You have to provide all of them and cannot provide them together with **START** and **STOP**

- **START:OUTER** - imprecise start location, outer boundary
- **START:INNER** - imprecise start location, inner boundary
- **STOP:INNER** - imprecise start location, inner boundary
- **STOP:OUTER** - imprecise stop location, outer boundary

The following headers are optional:

- **ACCESSION** - ClinVar SCV accession if any exists yet. When this is set then this variant will be updated in the batch rather than added as a novel variant.
- **CLIN_EVAL** - date of late clinical evaluation, e.g. 2022-12-02, leave empty to fill with the date of today

- CLIN_COMMENT - a comment on the clinical significance, e.g., ACMG Class IV; PS3, PM2_sup, PP4
- KEY - a local key to identify the variant/condition pair. Filled automatically with a UUID if missing, recommended to leave empty.
- HPO - List of HPO terms separated by comma or semicolon, any space will be stripped. E.g., HP:0004322; HP:0001263.
- PMID - List of Pubmed IDs separated by a comma or semicolon, any space will be stripped. E.g., 31859447,29474920.
- \$remove_from_batch - you can use this for removing a previously added variant from the given batch; one of true and false, defaults to false.

Any further header will be imported into the local repository into an `extra_data` field. Note that the error returned by ClinVar for your variant will be written to a `error_msg` field.

The following shows an example.

ASSEMBLY	CHROM	START	STOP	SV_TYPE	CONDITION	MOI	CLIN_SIG	HPO
GRCh38	chr1	844347	4398122	Deletion	not provided		Autosomal dominant	
→inheritance HP:0001263								

Note that you cannot submission TSV imports with batches that contain removals already.

6.3 Removal TSV

The following headers are required. Clinvar-this will recognize the TSV file format based on these headers.

- SCV - the ClinVar accession to be deleted
- REASON - a free text comment to give to ClinVar as a reason

You can optionally provide the following header:

- \$remove_from_batch - you can use this for removing a previously added variant from the given deletion batch; one of true and false, defaults to false.

The following shows an example:

SCV	REASON
SCV00042	Uploaded with hg38 coordinates but annotated as hg19; replaced by SCV00043.

Note that you cannot submission TSV imports with batches that contain removals already.

6.4 Phenopackets

Notes:

- This has not been implemented yet.

Note that only Phenopackets version 2 is supported. Phenopackets are interpreted as follows:

- When Family or Cohort are used then all contained Phenopacket records will be interpreted.
- Variants will be read from Phenopacket.diagnosis.genomic_interpretations and below.

- Each Diagnosis must be labeled with the corresponding disease (corresponds to OMIM in TSV). The following IDs are allowed for ClinVar: OMIM, MedGen, Orphanet, MeSH, HP, MONDO. When no disease is given, not provided will be used.
- `Diagnosis.genomic_interpretations` will be scanned for variants. When `interpretation_status` is `UNKNOWN_STATUS` or `REJECTED` then this `GenomicInterpretation` will be ignored. `GenomicInterpretation` records providing no `variant_interpretation` are ignored.
- `VariantInterpretation.acmg_pathogenicity_classification` will be mapped to the clinical significance (`CLIN_SIG` in TSV).
- `VariantInterpretation.variation_descriptor` will be used to describe the variant.
- See the section [Variant Call Files \(VCF\)](#) on the interpretation of `VariantDescription.vcf_record` (as it relates to the variant). As ClinVar API does not support allelic state yet, decode `allelic_state` to the mode of inheritance.

The following decoding `allelic_state` to mode of inheritance (MOI in TSV) is performed.

- `GENO:0000603` (heteroplasmic), `GENO:0000602` (homoplasmic) are mapped to `Mitochondrial inheritance`.
- `GENO:0000136` (homozygous), `GENO:0000402` (compound heterozygous) are mapped to `Autosomal recessive inheritance` unless the variant is on the X chromosome in which case `X-linked recessive inheritance` is used.
- `GENO:0000458` (simple heterozygous) is mapped to `Autosomal dominant inheritance` unless the variant is on the X chromosome in which case `X-linked dominant inheritance` is used.
- `GENO:0000604` (hemizygous X-linked) is mapped to `X-linked recessive inheritance`.
- `GENO:0000605` (hemizygous Y-linked) is mapped to `Y-linked inheritance`.
- In all other cases, `not provided` will be used.
- Note that you will need to use compound heterozygous even if you are matching the second hit to express recessive inheritance.

You currently cannot use phenopackets to update batches. You will need to export to TSV and re-import from there.

6.5 Variant Call Files (VCF)

Notes:

- This has not been implemented yet.
- The VCF file must contain headers for the chromosomes and the genome release is derived from the chromosome lengths.
- VCF files may only contain the one sample that is to be submitted.
- Small variants will be decoded directly from `CHROM`, `POS`, `REF`, `ALT`.
- Structural variants will be decoded as follows.
 - `REF` will be ignored
 - `ALT` should show one of the VCF alternative allele descriptions. We interpret the following ``, `<DUP>`, `<DUP:TANDEM>`, `<INV>`, `<INS>` and VCF encoded break-ends. If the `ALT` value matches a prefix in the list above (e.g., `<INS>` is a prefix for `<INS:ME>`) then this prefix will be used. All invalid variant specifications will be ignored.

- INFO/END must be the end position of the variant, for break-ends the target chromosome/pos is parsed from ALT.
- We will map break-ends and <INS> to Complex and the other types to the corresponding equivalents in ClinVar terminology.
- **You provide the following INFO fields (use URL encoding) for the mandatory information that you are used to from VCF.**
 - OMIM - the OMIM ID of the carrier, can be empty or “not provided”
 - HPO - corresponds to HPO in TSV
 - KEY - corresponds to KEY in TSV
 - CLIN_EVAL - corresponds to CLIN_EVAL in TSV
 - CLIN_COMMENT - corresponds to CLIN_COMMENT in TSV
 - clinvar_accession - corresponds to clinvar_accession in TSV

See the examples directory for example VCF files that also show you working VCF header sections for the INFO values used above.

You currently cannot use VCF to update batches (of course you can provide clinvar accessions to trigger ClinVar record updates). You will need to export to TSV and re-import from there.

6.6 ClinVar Excel Templates

Notes:

- This has not been implemented yet.

You already have a process for filling out these ClinVar Excel tables? You have one filled out already and not submitted before discovering clinvar-this? This is for you.

Only the “Variant” tab is used.

You have to use `SubmissionTemplate.xlsx`. The following columns are interpreted by clinvar-this.

- `Local ID / A` maps to `KEY` from the TSV format.
- For small variants, you can specify the coordinates based on transcripts or genomic description, so either will translate to (CHROM, POS, REF, and ALT; you will have to specify the release on the command line on import):
 - `Reference sequence / D` and `HGVS / E` are translated into chromosomal coordinates using the [Variant-Validator API](#), OR:
 - `Chromosome, Start, Stop, Reference allele, Alternate allele` in F–J.
- For structural variants, you have to provide:
 - `Chromosome, Start, Stop`, in F–H.
 - Alternatively to `Start/Stop`, you can provide `Outer start ... Outer stop` (L–O).
 - Provide the variant type in `Variant Type / K`.
- `Condition ID type / AB` and `Condition ID value / AC` map to OMIM in TSV.
- `Clinical significance / AH` maps to `CLIN_SIG` in TSV.
- `Date last evaluated / AJ` maps to `CLIN_EVAL` in TSV.
- `ClinVarAccession / CK` maps to `clinvar_accession` in TSV.

- Mode of inheritance / AK maps to MOI in TSV.

You currently cannot use ClinVar Excel to update batches (of course you can provide clinvar accessions to trigger ClinVar record updates). You will need to export to TSV and re-import from there.

LIMITATIONS

7.1 Design Decisions

7.1.1 clinvar_api

The `clinvar_api` module provides a full implementation of the NCBI ClinVar API as documented. In the case of changes to the upstream NCBI API, we aim to adjust the code appropriately. Please let us know about any issues in the [GitHub issue tracker](#). Note that the ClinVar API has some limitations when compared with uploading the spreadsheet templates via the ClinVar submission website.

At the time of writing (January 2023), the following were unsupported.

- age of onset
- observedIn
 - citations
 - tissue
 - variant allele count
 - zygosity

The ClinVar staff has been made aware of these limitations and intends to fill the gaps at some point.

7.1.2 clinvar-this

The `clinvar-this` software provides a **useful and not-so-minimal, yet limited, subset** for creating ClinVar submissions. The aim is to allow for the easy automated submission of variants with associated diseases and phenotypes to ClinVar in environments with medium to high turnover of cases such as diagnostics labs. Non-aims include providing a graphical interface and providing full, fine-grained access to all ClinVar submission API features.

The authors make the following assumptions:

- all variants have GRCh37 or GRCh38 coordinates
- users have experience in the Linux command line and bioinformatics data formats
- users have pathogenicity assessments and OMIM/HPO terms stored in machine-readable format already and do not need another database of their variants
- users are interested in getting their variant assessment information into ClinVar and then get on with their (work) life
- the ClinVar API is useful but not flawless; in the case of problems, users are willing to look into details (and report them back to the authors)

- if users want custom features, they will either implement them `clinvar_api` in their own software or work with the authors on improving `clinvar-this`

The `clinvar-this` package makes the following opinionated decisions:

- data is internally stored slightly augmented ClinVar API JSON data (full introspection) in internal repository
- all sent and received data is stored as in internal repository for debugging introspection when necessary
- common formats such as Phenopackets can be imported, but only certain subsets are interpreted with focus on usability in `clinvar-this`

7.2 Testing and Dry-run Inconsistencies

When enabling `--use-testing`, an alternative API endpoint provided by ClinVar will be used. This is different than when using `--dry-run` which uses the main endpoint with a dry-run parameter. This endpoint may use a different schema than the official endpoint (e.g., this has happened in November 2022). ClinVar has previously notified their submitters via email without official news posts.

7.3 Lack of Proper Versioning in NCBI ClinVar API

The ClinVar API from NCBI is not properly versioned. The same is true for their documentation and the JSON schemas that you may find on there.

7.4 Inconsistent Schema use by NCBI ClinVar API

You may see warnings about the results from the ClinVar API not fulfilling their provided schemas. This sometimes happens and you can ignore them. Apparently, the NCBI ClinVar server has no tight coupling to the JSON schemas and these schemas are mostly for informative purposes.

COMMON ERRORS

This section lists some errors that the ClinVar API commonly returns for data.

8.1 Reference Mismatch

Message

The submitted reference allele (<BASES>) does not agree with the actual reference sequence (<BASES>)

Explanation

There is a problem with the variant that you provided.

Resolution

Fix the variant description to match the reference bases in the reference.

8.2 Record is not Novel

Message

This record is submitted as novel but it should be submitted as an update, including the SCV accession, because your organization previously submitted <SCV> for the same variant and condition.

Explanation

Your organisation has already submitted a record with this “name” and condition (OMIM code or “not provided”). ClinVar generates a variant name from your genomic coordinates. Each organisation can only have one submission for the combination of the condition and variant.

Resolution

You can either submit a revision of your interpretation, (or, e.g., extend the “observed in” information), or leave it as is. Revisions are coded by providing the `ACCESSION` header in the TSV file.

8.3 Submission Names Cannot be Changed

Message

This update changes the description of the variant for <SCV>, which is generally not allowed on a ClinVar record. Please check the description of the variant and correct if necessary. If you intend to change the description of the variant, please submit as a new record and delete this record. Contact clinvar@ncbi.nlm.nih.gov if you have questions.

Explanation

Most likely, you try to update the coordinates of a variant with an existing SCV. ClinVar does not allow this. Rather, you should remove the old variant and create a new submission.

Resolution

Remove the old variant and add a new variant instead. Note that ClinVar will store your “local variant ID”. If you resubmit a new record, make sure that this is changed or cleared such that ClinVar does not link your new request to your old submission and thinks you want to change your variant coordinates.

8.4 Multiple Conditions have been submitted without explanation

Message

You provided multiple diseases as the condition for the classification. If they represent related diseases along a spectrum, provide `uncertain` for `multipleConditionExplanation`. If they represent diseases that occur together in an individual with the variant (this case is rare), provide `co-occurring` for `multipleConditionExplanation`.”

Explanation

Multiple Condition IDs have been submitted for single variant. Check if this has been intentional. If multiple conditions are to be submitted, a reason needs to be included with the submission.

Resolution

Explicitly add either `Uncertain`, `Co-occurring` or `Novel` disease to the list of `CONDITIONS`.

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

9.1 Types of Contributions

9.1.1 Report Bugs

Report bugs at <https://github.com/bihealth/clinvar-this/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

9.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

9.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

9.1.4 Write Documentation

clinvar-this could always use more documentation, whether as part of the official clinvar-this docs, in docstrings, or even on the web in blog posts, articles, and such.

9.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/bihealth/clinvar-this/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

9.2 Get Started!

Ready to contribute? Here's how to set up *clinvar-this* for local development.

1. Fork the *clinvar-this* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:bihealth/clinvar-this.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv clinvar-this
$ cd clinvar-this/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 clinvar-this tests
$ python setup.py test or pytest
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

9.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring.
3. The pull request should work for Python 3.8 to 3.11.

CREDITS

10.1 Development Lead

- Manuel Holtgrewe <manuel.holtgrewe@bih-charite.de>

10.2 Contributors

None yet. Why not be the first?

CHANGELOG

11.1 0.14.6 (2023-12-15)

11.2 Bug Fixes

- include JSON schemas in package (#184) (c3dc65a)

11.3 0.14.5 (2023-12-13)

11.4 Bug Fixes

- fixing MANIFEST.in for so py.typed are included (#182) (3015b32)

11.5 0.14.4 (2023-12-13)

11.6 Bug Fixes

- adding missing py.typed files for mypy (#180) (3b5945a)

11.7 0.14.3 (2023-12-12)

11.8 Bug Fixes

- fixing counter types (#178) (0b23c6b)

11.9 0.14.2 (2023-12-12)

11.10 Bug Fixes

- resolve dictionary dump enum issue (#176) (a6a6c20)

11.11 0.14.1 (2023-12-11)

11.12 Bug Fixes

- serialization of counter dict (#174) (06caa81)

11.12.1 0.14.0 (2023-12-11)

11.13 Features

- adding async client code via switch to httpx (#167) (#170) (7b76770)
- switch from attrs to pydantic (#166) (#168) (d84db77)
- updating to latest upstream JSON schemas (#173) (f5da1fc)
- write out RCV/VCV version in extract_vars and phenotype_link (#159) (#172) (534cba1)

11.14 Bug Fixes

- adding support for “no classification from unflagged records” review status (#171) (f144e74)

11.15 0.13.1 (2023-12-04)

11.16 Bug Fixes

- pinning python to 3.11 in release for setuptools (#164) (afb6035)

11.16.1 0.13.0 (2023-12-04)

11.17 Features

- adapt to ClinVar public XML update (#162) (8d90a36)

11.17.1 0.12.0 (2023-10-18)

11.18 Features

- include VCV and title in “clinvar-data extract-vars” (#152) (c3e1bf2)

11.18.1 0.11.0 (2023-10-06)

11.19 Features

- Adding Clinvar Accession to TSV input (#146) (1ef8b8f)
- Implement Pubmed IDs in tsv import format (#143) (9473380)
- Support multiple condition types and multiple conditions (#147) (f972356)

11.20 Bug Fixes

- mypy linting of tests (#144) (ee1fc5d)

11.21 Documentation

- Update docs on file formats and fixing errors (#150) (0b9efc6)

11.22 0.10.2 (2023-09-11)

11.23 Bug Fixes

- typo accession => accession (#135) (9ac72a2)

11.24 0.10.1 (2023-09-11)

11.25 Bug Fixes

- properly write out variants to JSONL (#133) (17b7e5c)

11.25.1 0.10.0 (2023-09-08)

11.26 Features

- write out HGNC identifier with variants (#130) (#131) (ea33100)

11.26.1 0.9.0 (2023-09-08)

11.27 Features

- write assessment to extract variants (#128) (dd72c90)

11.28 Bug Fixes

- normalizing molecular consequence (#126) (c2a72cf)

11.28.1 0.8.0 (2023-09-08)

11.29 Features

- variant extraction (#124) (c7bb7c9)

11.29.1 0.7.0 (2023-09-07)

11.30 Features

- writing out clinical significance in gene-to-phenotype link (#121) (1c65fc6)

11.30.1 0.6.0 (2023-09-07)

11.31 Features

- write out organisation and local key to phenotype gene link (#119) (f87facc)

11.31.1 0.5.0 (2023-09-07)

11.32 Features

- allow read/write .gz everywhere (#116) (d3b658e)

11.33 0.4.1 (2023-09-06)

11.34 Bug Fixes

- adding clinvar_data to built package (#114) (4a78256)

11.34.1 0.4.0 (2023-09-06)

11.35 Features

- add command to extract gene-to-phenotype links (#104) (#106) (b3b323a)
- adding `--version` option (#110) (7f3b091)
- adding command to generate per-gene impact report (#102) (#103) (c86a5c7)
- adding missing clinvar measure set (#109) (f0f64e0)
- adding support for parsing ClinVar XML (#99) (#100) (c673f7d)
- command to extract variants per ACMG class and freq. (#107) (#108) (c623f6d)

11.36 Bug Fixes

- making conversion more robust, indicate errors (#105) (8333aea)

11.36.1 0.3.0 (2023-09-04)

11.37 Features

- removing versioneer in favour of version from release-please (#95) (5b922bb)

11.38 Bug Fixes

- using varfish-bot token for release-please (#97) (35fad52)

11.39 0.2.1 (2023-03-17)

11.40 Bug Fixes

- fixing strucvar export (#84) (1050a67)
- sequence variant export (#82) (0fe3fb9)

11.40.1 0.2.0 (2023-03-15)

11.41 Features

- add support for SV TSV files (#75) (#78) (d4ffce)

11.42 Documentation

- add badges to readme (#65) (207038c)
- adding file format documentation (#71) (3a04ffd)
- extending docs; getting started from README (#44) (#68) (77892f0)
- updating docs introduction section (#67) (d6c865b)

11.42.1 0.1.0 (2022-12-02)

11.43 Features

- add basic config management in CLI (#32) (#33) (c903546)
- add functions for managing batch data (#34, #37) (#35) (0c7e0f9)
- add sphinx-based documentation (#30) (#31) (d10adc5)
- add tests for submission messages (#19) (#22) (b168d47)
- add unit tests for submission messages (#19) (#20) (1c4e11a)
- adding mypy type checking (#11) (#12) (700994a)
- adjust to ClinVar API change (Nov 2022) (#47) (#48) (0e4fb50)
- allow annotation with HPO terms in TSV format (#50) (#56) (0b0da41)
- allow import of extra columns (#53) (#54) (616bfe7)
- completing enums (#23) (#24) (9198983)
- implement api models (#4) (#5) (3690c36)

- implement attrs-based message models (#1) (#2) (f253c24)
- implement enums for submission messages (#14) (#16) (201cbe4)
- implement internal models for submissions (#9) (#26) (be04e40)
- implement minimal TSV format (#17) (#18) (960827d)
- implement models for extra files (#6) (#13) (852e4c6)
- implement models for submission message (#8) (#15) (90afd10)
- implement more columns in TSV (#39) (#55) (6184d76)
- implementing REST clinvar_api.client module (#28) (#29) (829d907)
- store errors from “batch retrieve” (#59) (#60) (67b8b0c)
- write data into profile sub directory (#52) (e699736)

11.44 Bug Fixes

- interpret TSV OMIM “not provided” (#57) (#58) (b655097)

11.45 Documentation

- add getting started tutorial to README (#45) (#61) (3de084b)

COMMAND LINE INTERFACE

12.1 clinvar-this

Main entry point for CLI via click.

```
clinvar-this [OPTIONS] COMMAND [ARGS]...
```

Options

--version

Show the version and exit.

--verbose, --no-verbose

--profile <profile>

The profile to use

--verify-ssl, --no-verify-ssl

Whether to enable SSL verification

12.1.1 batch

Sub comment category batch ...

```
clinvar-this batch [OPTIONS] COMMAND [ARGS]...
```

export

Export batch data to a given file

```
clinvar-this batch export [OPTIONS] NAME PATH
```

Options

--force, --no-force

Overwrite existing files

--struc-var, --no-struc-var

Export structural variants rather than sequence variants

Arguments

NAME

Required argument

PATH

Required argument

import

Import data for a new or existing batch

```
clinvar-this batch import [OPTIONS] PATH
```

Options

-m, --metadata <metadata>

Provide meta data settings as KEY=VALUE settings

--name <name>

Name of the batch to create or add to

Arguments

PATH

Required argument

list

List existing batches

```
clinvar-this batch list [OPTIONS]
```

retrieve

Submit the given batch to ClinVar

```
clinvar-this batch retrieve [OPTIONS] NAME
```

Options

--use-testing, --no-testing

Whether to use the testing API

Arguments

NAME

Required argument

submit

Submit the given batch to ClinVar

```
clinvar-this batch submit [OPTIONS] NAME
```

Options

--use-testing, --no-testing

Whether to use the testing API

--dry-run, --no-dry-run

Whether to use the ClinVar dry-run

Arguments

NAME

Required argument

update-metadata

Update batch metadata without importing files

```
clinvar-this batch update-metadata [OPTIONS] NAME [METADATA]...
```

Arguments

NAME

Required argument

METADATA

Optional argument(s)

12.1.2 config

Sub command category varfish-this config ...

```
clinvar-this config [OPTIONS] COMMAND [ARGS]...
```

dump

Sub command varfish-this config dump

Print the configuration file to stdout.

```
clinvar-this config dump [OPTIONS]
```

get

Sub command varfish-this config get NAME

Show the configuration variable with the given NAME. This will interpret the current --profile setting.

```
clinvar-this config get [OPTIONS] NAME
```

Options

--profile <profile>

The profile to get the value from

Arguments

NAME

Required argument

set

Sub command varfish-this config set NAME VALUE

Set the configuration variable with the given NAME to the given VALUE. This will interpret the current --profile setting.

```
clinvar-this config set [OPTIONS] NAME VALUE
```


Arguments

NAME

Required argument

VALUE

Required argument

12.1.3 data

Sub command category “data”.

```
clinvar-this data [OPTIONS] COMMAND [ARGS]...
```

acmg-class-by-freq

Create links between gene and phenotype.

```
clinvar-this data acmg-class-by-freq [OPTIONS] INPUT_FILE OUTPUT_FILE
```

Options

--thresholds <thresholds>

Whether to filter to rows with HPO terms (default: true)

Arguments

INPUT_FILE

Required argument

OUTPUT_FILE

Required argument

extract-vars

Write out variants from RCV records.

```
clinvar-this data extract-vars [OPTIONS] PATH_INPUT PATH_OUTPUT_DIR
```

Options

--gzip-output, --no-gzip-output

Whether to gzip output (default: true)

Arguments

PATH_INPUT

Required argument

PATH_OUTPUT_DIR

Required argument

gene-phenotype-links

Create links between gene and phenotype.

```
clinvar-this data gene-phenotype-links [OPTIONS] INPUT_FILE OUTPUT_FILE
```

Options

--needs-hpo-terms, --no-needs-hpo-terms

Whether to filter to rows with HPO terms (default: true)

Arguments

INPUT_FILE

Required argument

OUTPUT_FILE

Required argument

gene-variant-report

Create a gene variant summary report.

```
clinvar-this data gene-variant-report [OPTIONS] INPUT_FILE OUTPUT_FILE
```

Arguments

INPUT_FILE

Required argument

OUTPUT_FILE

Required argument

xml-to-jsonl

Convert XML to JSONL

```
clinvar-this data xml-to-jsonl [OPTIONS] INPUT_FILE OUTPUT_FILE
```

Options

--max-records <max_records>

Maximum number of records to convert

Arguments

INPUT_FILE

Required argument

OUTPUT_FILE

Required argument

CLINVAR_API

13.1 clinvar_api package

13.1.1 Subpackages

clinvar_api.models package

Submodules

clinvar_api.models.query_response module

Data structures for internal representation of query response.

```
class clinvar_api.models.query_response.SubmissionStatus(*, actions:
                                                         List[SubmissionStatusActions])
```

Bases: BaseModel

Internal submission status.

actions: List[SubmissionStatusActions]

The list of actions (one element only by the docs).

classmethod from_msg(other: SubmissionStatus)

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'actions':
FieldInfo(annotation=List[SubmissionStatusActions], required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.models.query_response.SubmissionStatusActions(*, id: str, responses:
                                                                List[SubmissionStatusResponse],
                                                                status: str, target_db: str,
                                                                updated: datetime)
```

Bases: BaseModel

classmethod from_msg(other: SubmissionStatusActions)

id: str

Identifier of the submission

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: ClassVar[dict[str, FieldInfo]] = {'id': FieldInfo(annotation=str, required=True), 'responses': FieldInfo(annotation=List[SubmissionStatusResponse], required=True), 'status': FieldInfo(annotation=str, required=True), 'target_db': FieldInfo(annotation=str, required=True), 'updated': FieldInfo(annotation=datetime, required=True)}

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

responses: List[SubmissionStatusResponse]

Entries in `actions[*].responses`, only one entry per the docs.

status: str

Status of the submission, one of “submitted”, “processing”, “processed”, “error”

target_db: str

Target database, usually “clinvar”

updated: datetime

Last updated time

class `clinvar_api.models.query_response.SubmissionStatusFile(*, url: str)`

Bases: `BaseModel`

classmethod `from_msg(other: SubmissionStatusFile)`

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: ClassVar[dict[str, FieldInfo]] = {'url': FieldInfo(annotation=str, required=True)}

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

url: str

class `clinvar_api.models.query_response.SubmissionStatusObject(*, accession: str | None, content: SubmissionStatusObjectContent, target_db: str)`

Bases: `BaseModel`

accession: str | None

Optional object accession.

content: SubmissionStatusObjectContent

Object content.

```

classmethod from_msg(other: SubmissionStatusObject)

model_config: ClassVar[ConfigDict] = {'frozen': True}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'accession':
FieldInfo(annotation=Union[str, NoneType], required=True), 'content':
FieldInfo(annotation=SubmissionStatusObjectContent, required=True), 'target_db':
FieldInfo(annotation=str, required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

target_db: str
    Target database, usually “clinvar” per the docs.

class clinvar_api.models.query_response.SubmissionStatusObjectContent(*, clin-
var_processing_status:
str,
clinvar_release_status:
str)

Bases: BaseModel

clinvar_processing_status: str
    Processing status

clinvar_release_status: str
    Release status

classmethod from_msg(other: SubmissionStatusObjectContent)

model_config: ClassVar[ConfigDict] = {'frozen': True}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'clinvar_processing_status':
FieldInfo(annotation=str, required=True), 'clinvar_release_status':
FieldInfo(annotation=str, required=True)}

    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

class clinvar_api.models.query_response.SubmissionStatusResponse(*, status: str, files:
List[SubmissionStatusFile],
message: SubmissionStatusRe-
sponseMessage | None, objects:
List[SubmissionStatusObject])

Bases: BaseModel

files: List[SubmissionStatusFile]
    Files in the response.

classmethod from_msg(other: SubmissionStatusResponse)

```

message: [SubmissionStatusResponseMessage](#) | None

Message

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'files': FieldInfo(annotation=List[SubmissionStatusFile], required=True), 'message': FieldInfo(annotation=Union[SubmissionStatusResponseMessage, NoneType], required=True), 'objects': FieldInfo(annotation=List[SubmissionStatusObject], required=True), 'status': FieldInfo(annotation=str, required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

objects: `List[SubmissionStatusObject]`

Objects

status: `str`

Status, one of “processing”, “processed”, “error”,

class `clinvar_api.models.query_response.SubmissionStatusResponseMessage(*, error_code: ErrorCode | None, severity: str, text: str)`

Bases: `BaseModel`

error_code: [ErrorCode](#) | None

The error code.

classmethod `from_msg(other: SubmissionStatusResponseMessage)`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'error_code': FieldInfo(annotation=Union[ErrorCode, NoneType], required=True), 'severity': FieldInfo(annotation=str, required=True), 'text': FieldInfo(annotation=str, required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

severity: `str`

The message severity.

text: `str`

The message text.

clinvar_api.models.sub_payload module

Data structures for internal representation of submission payload.

```
class clinvar_api.models.sub_payload.SubmissionAssertionCriteria(*, db: CitationDb | None =
None, id: str | None = None,
url: str | None = None)
```

Bases: `BaseModel`

db: [CitationDb](#) | None

id: str | None

model_config: `ClassVar[ConfigDict]` = {'frozen': True}

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]]` = {'db':
`FieldInfo(annotation=Union[CitationDb, NoneType], required=False)`, 'id':
`FieldInfo(annotation=Union[str, NoneType], required=False)`, 'url':
`FieldInfo(annotation=Union[str, NoneType], required=False)`}

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

to_msg() → [SubmissionAssertionCriteria](#)

url: str | None

```
class clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates(*, accession: str | None =
None, alternate_allele: str
| None = None, assembly:
Assembly | None = None,
chromosome:
Chromosome | None =
None, inner_start: int |
None = None, inner_stop:
int | None = None,
outer_start: int | None =
None, outer_stop: int |
None = None,
reference_allele: str |
None = None, start: int |
None = None, stop: int |
None = None,
variant_length: int | None
= None)
```

Bases: `BaseModel`

accession: str | None

alternate_allele: str | None

assembly: [Assembly](#) | None

chromosome: [Chromosome](#) | None

inner_start: int | None

inner_stop: int | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

model_fields: ClassVar[dict[str, FieldInfo]] = {'accession':
FieldInfo(annotation=Union[str, NoneType], required=False), 'alternate_allele':
FieldInfo(annotation=Union[str, NoneType], required=False), 'assembly':
FieldInfo(annotation=Union[Assembly, NoneType], required=False), 'chromosome':
FieldInfo(annotation=Union[Chromosome, NoneType], required=False), 'inner_start':
FieldInfo(annotation=Union[int, NoneType], required=False), 'inner_stop':
FieldInfo(annotation=Union[int, NoneType], required=False), 'outer_start':
FieldInfo(annotation=Union[int, NoneType], required=False), 'outer_stop':
FieldInfo(annotation=Union[int, NoneType], required=False), 'reference_allele':
FieldInfo(annotation=Union[str, NoneType], required=False), 'start':
FieldInfo(annotation=Union[int, NoneType], required=False), 'stop':
FieldInfo(annotation=Union[int, NoneType], required=False), 'variant_length':
FieldInfo(annotation=Union[int, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

outer_start: int | None

outer_stop: int | None

reference_allele: str | None

start: int | None

stop: int | None

to_msg() → *SubmissionChromosomeCoordinates*

variant_length: int | None

```
class clinvar_api.models.sub_payload.SubmissionCitation(*, db: CitationDb | None = None, id: str |  
None = None, url: str | None = None)
```

Bases: BaseModel

db: [CitationDb](#) | None

id: str | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

model_fields: ClassVar[dict[str, FieldInfo]] = {'db':
FieldInfo(annotation=Union[CitationDb, NoneType], required=False), 'id':
FieldInfo(annotation=Union[str, NoneType], required=False), 'url':
FieldInfo(annotation=Union[str, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`to_msg()` → *SubmissionCitation*

url: `str | None`

```
class clinvar_api.models.sub_payload.SubmissionClinicalFeature(*,
    clinical_features_affected_status:
        ClinicalFeaturesAffectedStatus,
    db: ClinicalFeaturesDb | None =
        None, id: str | None = None,
    name: str | None = None)
```

Bases: `BaseModel`

clinical_features_affected_status: *ClinicalFeaturesAffectedStatus*

db: *ClinicalFeaturesDb* | `None`

id: `str | None`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] =
{'clinical_features_affected_status':
FieldInfo(annotation=ClinicalFeaturesAffectedStatus, required=True), 'db':
FieldInfo(annotation=Union[ClinicalFeaturesDb, NoneType], required=False), 'id':
FieldInfo(annotation=Union[str, NoneType], required=False), 'name':
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

name: `str | None`

`to_msg()` → *SubmissionClinicalFeature*

```
class clinvar_api.models.sub_payload.SubmissionClinicalSignificance(*, clinical_significance_description: ClinicalSignificanceDescription, citation: List[SubmissionCitation] | None = None, comment: str | None = None, custom_assertion_score: float | None = None, date_last_evaluated: str | None = None, explanation_of_drug_response: str | None = None, explanation_of_other_clinical_significance: str | None = None, mode_of_inheritance: ModeOfInheritance | None = None)
```

Bases: BaseModel

citation: List[SubmissionCitation] | None

Must have at least one entry.

clinical_significance_description: ClinicalSignificanceDescription

comment: str | None

custom_assertion_score: float | None

date_last_evaluated: str | None

explanation_of_drug_response: str | None

explanation_of_other_clinical_significance: str | None

mode_of_inheritance: ModeOfInheritance | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'citation': FieldInfo(annotation=Union[List[SubmissionCitation], NoneType], required=False), 'clinical_significance_description': FieldInfo(annotation=ClinicalSignificanceDescription, required=True), 'comment': FieldInfo(annotation=Union[str, NoneType], required=False), 'custom_assertion_score': FieldInfo(annotation=Union[float, NoneType], required=False), 'date_last_evaluated': FieldInfo(annotation=Union[str, NoneType], required=False), 'explanation_of_drug_response': FieldInfo(annotation=Union[str, NoneType], required=False), 'explanation_of_other_clinical_significance': FieldInfo(annotation=Union[str, NoneType], required=False), 'mode_of_inheritance': FieldInfo(annotation=Union[ModeOfInheritance, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

`to_msg()` → *SubmissionClinicalSignificance*

```
class clinvar_api.models.sub_payload.SubmissionClinvarDeletion(*, accession_set:
                                                                    List[SubmissionClinvarDeletionAccessionSet])
```

Bases: BaseModel

accession_set: List[*SubmissionClinvarDeletionAccessionSet*]

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'accession_set': FieldInfo(annotation=List[SubmissionClinvarDeletionAccessionSet], required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

`to_msg()` → *SubmissionClinvarDeletion*

```
class clinvar_api.models.sub_payload.SubmissionClinvarDeletionAccessionSet(*, accession: str,
                                                                              reason: str | None
                                                                              = None)
```

Bases: BaseModel

accession: str

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'accession': FieldInfo(annotation=str, required=True), 'reason': FieldInfo(annotation=Union[str, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

reason: str | None

`to_msg()` → *SubmissionClinvarDeletionAccessionSet*

```
class clinvar_api.models.sub_payload.SubmissionClinvarSubmission(*, clinical_significance: Sub-
missionClinicalSignificance,
condition_set:
SubmissionConditionSet,
observed_in:
List[SubmissionObservedIn],
record_status: RecordStatus,
clinvar_accession: str | None =
None,
compound_heterozygote_set:
SubmissionCompound-
HeterozygoteSet | None =
None, diplotype_set:
SubmissionDiplotypeSet | None
= None,
distinct_chromosomes_set:
SubmissionDistinctChromo-
somesSet | None = None,
haplotype_set:
SubmissionHaplotypeSet |
None = None,
haplotype_single_variant_set:
SubmissionHaplotypeSet |
None = None, local_id: str |
None = None, local_key: str |
None = None,
phase_unknown_set:
SubmissionPhaseUnknownSet |
None = None, variant_set:
SubmissionVariantSet | None =
None, extra_data: Dict[str,
Any] | None = None)
```

Bases: BaseModel

clinical_significance: *SubmissionClinicalSignificance*

clinvar_accession: str | None

compound_heterozygote_set: *SubmissionCompoundHeterozygoteSet* | None

condition_set: *SubmissionConditionSet*

diplotype_set: *SubmissionDiplotypeSet* | None

distinct_chromosomes_set: *SubmissionDistinctChromosomesSet* | None

extra_data: Dict[str, Any] | None

Additional information from import. Will not be used for conversion to message but can be converted back to external formats.

haplotype_set: *SubmissionHaplotypeSet* | None

Has at least two elements in *variants*

haplotype_single_variant_set: *SubmissionHaplotypeSet* | None

Has exactly one elements in *variants*

`local_id: str | None`

`local_key: str | None`

`model_config: ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

`model_fields: ClassVar[dict[str, FieldInfo]] = {'clinical_significance': FieldInfo(annotation=SubmissionClinicalSignificance, required=True), 'clinvar_accession': FieldInfo(annotation=Union[str, NoneType], required=False), 'compound_heterozygote_set': FieldInfo(annotation=Union[SubmissionCompoundHeterozygoteSet, NoneType], required=False), 'condition_set': FieldInfo(annotation=SubmissionConditionSet, required=True), 'diplotype_set': FieldInfo(annotation=Union[SubmissionDiplotypeSet, NoneType], required=False), 'distinct_chromosomes_set': FieldInfo(annotation=Union[SubmissionDistinctChromosomesSet, NoneType], required=False), 'extra_data': FieldInfo(annotation=Union[Dict[str, Any], NoneType], required=False), 'haplotype_set': FieldInfo(annotation=Union[SubmissionHaplotypeSet, NoneType], required=False), 'haplotype_single_variant_set': FieldInfo(annotation=Union[SubmissionHaplotypeSet, NoneType], required=False), 'local_id': FieldInfo(annotation=Union[str, NoneType], required=False), 'local_key': FieldInfo(annotation=Union[str, NoneType], required=False), 'observed_in': FieldInfo(annotation=List[SubmissionObservedIn], required=True), 'phase_unknown_set': FieldInfo(annotation=Union[SubmissionPhaseUnknownSet, NoneType], required=False), 'record_status': FieldInfo(annotation=RecordStatus, required=True), 'variant_set': FieldInfo(annotation=Union[SubmissionVariantSet, NoneType], required=False)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`observed_in: List[SubmissionObservedIn]`

`phase_unknown_set: SubmissionPhaseUnknownSet | None`

`record_status: RecordStatus`

`to_msg() → SubmissionClinvarSubmission`

`variant_set: SubmissionVariantSet | None`

```
class clinvar_api.models.sub_payload.SubmissionCompoundHeterozygoteSet(*, hgvs: str,
                                                                    variant_sets:
                                                                    List[SubmissionCompoundHeterozygoteS
```

Bases: `BaseModel`

`hgvs: str`

`model_config: ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

`model_fields: ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str, required=True), 'variant_sets': FieldInfo(annotation=List[SubmissionCompoundHeterozygoteSetVariantSet], required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`to_msg()` → `SubmissionCompoundHeterozygoteSet`

`variant_sets: List[SubmissionCompoundHeterozygoteSetVariantSet]`

```
class clinvar_api.models.sub_payload.SubmissionCompoundHeterozygoteSetVariantSet(*, variant_set: SubmissionVariantSet | None = None)
```

Bases: `BaseModel`

`model_config: ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

`model_fields: ClassVar[dict[str, FieldInfo]] = {'variant_set': FieldInfo(annotation=Union[SubmissionVariantSet, NoneType], required=False)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`to_msg()` → `SubmissionCompoundHeterozygoteSetVariantSet`

`variant_set: SubmissionVariantSet | None`

```
class clinvar_api.models.sub_payload.SubmissionCondition(*, db: ConditionDb | None = None, id: str | None = None, name: str | None = None)
```

Bases: `BaseModel`

`db: ConditionDb | None`

`id: str | None`

`model_config: ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

`model_fields: ClassVar[dict[str, FieldInfo]] = {'db': FieldInfo(annotation=Union[ConditionDb, NoneType], required=False), 'id': FieldInfo(annotation=Union[str, NoneType], required=False), 'name': FieldInfo(annotation=Union[str, NoneType], required=False)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

`name: str | None`

`to_msg()` → `SubmissionCondition`


```
class clinvar_api.models.sub_payload.SubmissionConditionSet(*, condition:
    List[SubmissionCondition] | None =
    None, drug_response:
    List[SubmissionDrugResponse] |
    None = None,
    multiple_condition_explanation:
    MultipleConditionExplanation | None
    = None)
```

Bases: BaseModel

condition: List[SubmissionCondition] | None

drug_response: List[SubmissionDrugResponse] | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'condition': FieldInfo(annotation=Union[List[SubmissionCondition], NoneType], required=False), 'drug_response': FieldInfo(annotation=Union[List[SubmissionDrugResponse], NoneType], required=False), 'multiple_condition_explanation': FieldInfo(annotation=Union[MultipleConditionExplanation, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

multiple_condition_explanation: MultipleConditionExplanation | None

to_msg() → SubmissionConditionSet

```
class clinvar_api.models.sub_payload.SubmissionContainer(*, assertion_criteria:
    SubmissionAssertionCriteria | None =
    None, behalf_org_id: int | None = None,
    clinvar_deletion:
    SubmissionClinvarDeletion | None =
    None, clinvar_submission:
    List[SubmissionClinvarSubmission] |
    None = None,
    clinvar_submission_release_status:
    ReleaseStatus | None = None,
    submission_name: str | None = None)
```

Bases: BaseModel

assertion_criteria: SubmissionAssertionCriteria | None

behalf_org_id: int | None

clinvar_deletion: SubmissionClinvarDeletion | None

clinvar_submission: List[SubmissionClinvarSubmission] | None

clinvar_submission_release_status: ReleaseStatus | None

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'assertion_criteria':  
FieldInfo(annotation=Union[SubmissionAssertionCriteria, NoneType], required=False),  
'behalf_org_id': FieldInfo(annotation=Union[int, NoneType], required=False),  
'clinvar_deletion': FieldInfo(annotation=Union[SubmissionClinvarDeletion,  
NoneType], required=False), 'clinvar_submission':  
FieldInfo(annotation=Union[List[SubmissionClinvarSubmission], NoneType],  
required=False), 'clinvar_submission_release_status':  
FieldInfo(annotation=Union[ReleaseStatus, NoneType], required=False),  
'submission_name': FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
submission_name: str | None
```

```
to_msg() → SubmissionContainer
```

```
class clinvar_api.models.sub_payload.SubmissionDiplotypeSet(*, haplotype_sets:  
List[SubmissionHaplotypeSets], hgvs:  
str, star_allele_name: str | None =  
None)
```

Bases: BaseModel

```
haplotype_sets: List[SubmissionHaplotypeSets]
```

```
hgvs: str
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'haplotype_sets':  
FieldInfo(annotation=List[SubmissionHaplotypeSets], required=True), 'hgvs':  
FieldInfo(annotation=str, required=True), 'star_allele_name':  
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
star_allele_name: str | None
```

```
to_msg() → SubmissionDiplotypeSet
```

```
class clinvar_api.models.sub_payload.SubmissionDistinctChromosomesSet(*, hgvs: str, variants:  
List[SubmissionVariant])
```

Bases: BaseModel

```
hgvs: str
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str,
required=True), 'variants': FieldInfo(annotation=List[SubmissionVariant],
required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

to_msg() → *SubmissionDistinctChromosomesSet*

variants: *List[SubmissionVariant]*

Has at least two elements

```
class clinvar_api.models.sub_payload.SubmissionDrugResponse(*, db: ConditionDb | None = None,
drug_name: str | None = None, id: str
| None = None, condition:
List[SubmissionCondition] | None =
None)
```

Bases: *BaseModel*

condition: *List[SubmissionCondition] | None*

db: *ConditionDb | None*

drug_name: *str | None*

id: *str | None*

model_config: *ClassVar[ConfigDict] = {'frozen': True}*

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'condition':
FieldInfo(annotation=Union[List[SubmissionCondition], NoneType], required=False),
'db': FieldInfo(annotation=Union[ConditionDb, NoneType], required=False),
'drug_name': FieldInfo(annotation=Union[str, NoneType], required=False), 'id':
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

to_msg() → *SubmissionDrugResponse*

```
class clinvar_api.models.sub_payload.SubmissionHaplotypeSet(*, hgvs: str, variants:
List[SubmissionVariant],
star_allele_name: str | None = None)
```

Bases: *BaseModel*

hgvs: *str*

model_config: *ClassVar[ConfigDict] = {'frozen': True}*

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str,
required=True), 'star_allele_name': FieldInfo(annotation=Union[str, NoneType],
required=False), 'variants': FieldInfo(annotation=List[SubmissionVariant],
required=True)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

star_allele_name: `str | None`

to_msg() → `SubmissionHaplotypeSet`

variants: `List[SubmissionVariant]`

```
class clinvar_api.models.sub_payload.SubmissionHaplotypeSets(*, haplotype_set:
    SubmissionHaplotypeSet | None =
    None, haplotype_single_variant_set:
    SubmissionHaplotypeSet | None =
    None)
```

Bases: `BaseModel`

haplotype_set: `SubmissionHaplotypeSet | None`

haplotype_single_variant_set: `SubmissionHaplotypeSet | None`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'haplotype_set': FieldInfo(annotation=Union[SubmissionHaplotypeSet, NoneType], required=False), 'haplotype_single_variant_set': FieldInfo(annotation=Union[SubmissionHaplotypeSet, NoneType], required=False)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

to_msg() → `SubmissionHaplotypeSets`

```
class clinvar_api.models.sub_payload.SubmissionObservedIn(*, affected_status: AffectedStatus,
    allele_origin: AlleleOrigin,
    collection_method: CollectionMethod,
    clinical_features:
    List[SubmissionClinicalFeature] | None
    = None, clinical_features_comment: str |
    None = None, number_of_individuals:
    int | None = None,
    struct_var_method_type:
    StructVarMethodType | None = None)
```

Bases: `BaseModel`

affected_status: `AffectedStatus`

allele_origin: `AlleleOrigin`

clinical_features: `List[SubmissionClinicalFeature] | None`

clinical_features_comment: `str | None`

collection_method: [CollectionMethod](#)

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'affected_status': FieldInfo(annotation=AffectedStatus, required=True), 'allele_origin': FieldInfo(annotation=AlleleOrigin, required=True), 'clinical_features': FieldInfo(annotation=Union[List[SubmissionClinicalFeature], NoneType], required=False), 'clinical_features_comment': FieldInfo(annotation=Union[str, NoneType], required=False), 'collection_method': FieldInfo(annotation=CollectionMethod, required=True), 'number_of_individuals': FieldInfo(annotation=Union[int, NoneType], required=False), 'struct_var_method_type': FieldInfo(annotation=Union[StructVarMethodType, NoneType], required=False)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

number_of_individuals: `int | None`

struct_var_method_type: [StructVarMethodType](#) | None

to_msg() → [SubmissionObservedIn](#)

class `clinvar_api.models.sub_payload.SubmissionPhaseUnknownSet(*, hgvs: str, variants: List[SubmissionVariant])`

Bases: `BaseModel`

hgvs: `str`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str, required=True), 'variants': FieldInfo(annotation=List[SubmissionVariant], required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

to_msg() → [SubmissionPhaseUnknownSet](#)

variants: `List[SubmissionVariant]`

class `clinvar_api.models.sub_payload.SubmissionVariant(*, chromosome_coordinates: SubmissionChromosomeCoordinates | None = None, copy_number: str | None = None, gene: List[SubmissionVariantGene] | None = None, hgvs: str | None = None, reference_copy_number: int | None = None, variant_type: VariantType | None = None)`

Bases: `BaseModel`

chromosome_coordinates: [SubmissionChromosomeCoordinates](#) | None

copy_number: str | None

gene: List[[SubmissionVariantGene](#)] | None

hgvs: str | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'chromosome_coordinates': FieldInfo(annotation=Union[SubmissionChromosomeCoordinates, NoneType], required=False), 'copy_number': FieldInfo(annotation=Union[str, NoneType], required=False), 'gene': FieldInfo(annotation=Union[List[SubmissionVariantGene], NoneType], required=False), 'hgvs': FieldInfo(annotation=Union[str, NoneType], required=False), 'reference_copy_number': FieldInfo(annotation=Union[int, NoneType], required=False), 'variant_type': FieldInfo(annotation=Union[VariantType, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

reference_copy_number: int | None

to_msg() → [SubmissionVariant](#)

variant_type: [VariantType](#) | None

class clinvar_api.models.sub_payload.[SubmissionVariantGene](#)(**id: int | None = None, symbol: str | None = None*)

Bases: BaseModel

id: int | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'id': FieldInfo(annotation=Union[int, NoneType], required=False), 'symbol': FieldInfo(annotation=Union[str, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

symbol: str | None

to_msg() → [SubmissionVariantGene](#)

class clinvar_api.models.sub_payload.[SubmissionVariantSet](#)(**variant: List[SubmissionVariant]*)

Bases: BaseModel

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'variant':
FieldInfo(annotation=List[SubmissionVariant], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
to_msg() → SubmissionVariantSet
```

```
variant: List[SubmissionVariant]
```

clinvar_api.models.sub_response module

Data structures for submission response.

```
class clinvar_api.models.sub_response.Created(*, id: str)
```

Bases: BaseModel

```
classmethod from_msg(other: Created)
```

```
id: str
```

The submission ID.

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'id': FieldInfo(annotation=str,
required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.models.sub_response.Error(*, message: str)
```

Bases: BaseModel

```
classmethod from_msg(other: Error)
```

```
message: str
```

The error response's message.

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'message':
FieldInfo(annotation=str, required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.models.sub_response.SummaryResponse(*, batch_processing_status:
    BatchProcessingStatus, batch_release_status:
    BatchReleaseStatus, submission_date: str,
    submission_name: str, total_count: int,
    total_errors: int, total_public: int,
    total_success: int, deletions:
    List[SummaryResponseDeletion] | None =
    None, submissions:
    List[SummaryResponseSubmission] | None =
    None, total_delete_count: int | None = None,
    total_deleted: int | None = None,
    total_delete_errors: int | None = None,
    total_delete_success: int | None = None)
```

Bases: BaseModel

Representation of server's response to a submission.

batch_processing_status: *BatchProcessingStatus*

batch_release_status: *BatchReleaseStatus*

deletions: *List[SummaryResponseDeletion]* | None

classmethod from_msg(other: *SummaryResponse*)

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

model_fields: ClassVar[dict[str, FieldInfo]] = {'batch_processing_status': FieldInfo(annotation=BatchProcessingStatus, required=True), 'batch_release_status': FieldInfo(annotation=BatchReleaseStatus, required=True), 'deletions': FieldInfo(annotation=Union[List[SummaryResponseDeletion], NoneType], required=False), 'submission_date': FieldInfo(annotation=str, required=True), 'submission_name': FieldInfo(annotation=str, required=True), 'submissions': FieldInfo(annotation=Union[List[SummaryResponseSubmission], NoneType], required=False), 'total_count': FieldInfo(annotation=int, required=True), 'total_delete_count': FieldInfo(annotation=Union[int, NoneType], required=False), 'total_delete_errors': FieldInfo(annotation=Union[int, NoneType], required=False), 'total_delete_success': FieldInfo(annotation=Union[int, NoneType], required=False), 'total_deleted': FieldInfo(annotation=Union[int, NoneType], required=False), 'total_errors': FieldInfo(annotation=int, required=True), 'total_public': FieldInfo(annotation=int, required=True), 'total_success': FieldInfo(annotation=int, required=True)}

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

submission_date: str

submission_name: str

submissions: *List[SummaryResponseSubmission]* | None

total_count: int


```

total_delete_count: int | None
total_delete_errors: int | None
total_delete_success: int | None
total_deleted: int | None
total_errors: int
total_public: int
total_success: int

```

```

class clinvar_api.models.sub_response.SummaryResponseDeletion(*, identifiers: SummaryRe-
                                                                    sponseDeletionIdentifier,
                                                                    processing_status: str, delete_date:
                                                                    str | None = None, delete_status:
                                                                    str | None = None, errors:
                                                                    List[SummaryResponseError] |
                                                                    None = None)

```

Bases: BaseModel

```
delete_date: str | None
```

```
delete_status: str | None
```

```
errors: List[SummaryResponseError] | None
```

```
classmethod from_msg(other: SummaryResponseDeletion)
```

```
identifiers: SummaryResponseDeletionIdentifier
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'delete_date':
FieldInfo(annotation=Union[str, NoneType], required=False), 'delete_status':
FieldInfo(annotation=Union[str, NoneType], required=False), 'errors':
FieldInfo(annotation=Union[List[SummaryResponseError], NoneType], required=False),
'identifiers': FieldInfo(annotation=SummaryResponseDeletionIdentifier,
required=True), 'processing_status': FieldInfo(annotation=str, required=True)}

```

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces `Model.__fields__` from Pydantic V1.

```
processing_status: str
```

```

class clinvar_api.models.sub_response.SummaryResponseDeletionIdentifier(*, clinvar_accession:
                                                                    str, clinvar_local_key:
                                                                    str | None = None)

```

Bases: BaseModel

```
clinvar_accession: str
```

```
clinvar_local_key: str | None
```

```
classmethod from_msg(other: SummaryResponseDeletionIdentifier)
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'clinvar_accession':
```

```
FieldInfo(annotation=str, required=True), 'clinvar_local_key':
```

```
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
class clinvar_api.models.sub_response.SummaryResponseError(*, input:
                                                             List[SummaryResponseErrorInput],
                                                             output:
                                                             SummaryResponseErrorOutput)
```

Bases: `BaseModel`

```
classmethod from_msg(other: SummaryResponseError)
```

```
input: List[SummaryResponseErrorInput]
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'input':
```

```
FieldInfo(annotation=List[SummaryResponseErrorInput], required=True), 'output':
```

```
FieldInfo(annotation=SummaryResponseErrorOutput, required=True)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
output: SummaryResponseErrorOutput
```

```
class clinvar_api.models.sub_response.SummaryResponseErrorInput(*, value: str | None = None,
                                                                  field: str | None = None)
```

Bases: `BaseModel`

```
field: str | None
```

```
classmethod from_msg(other: SummaryResponseErrorInput)
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'field':
```

```
FieldInfo(annotation=Union[str, NoneType], required=False), 'value':
```

```
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

value: `str | None`

```
class clinvar_api.models.sub_response.SummaryResponseErrorOutput(*, errors:
                                                                    List[SummaryResponseOutputError])
```

Bases: `BaseModel`

errors: `List[SummaryResponseOutputError]`

classmethod `from_msg(other: SummaryResponseOutput)`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'errors': FieldInfo(annotation=List[SummaryResponseOutputError], required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
class clinvar_api.models.sub_response.SummaryResponseOutputError(*, user_message: str)
```

Bases: `BaseModel`

classmethod `from_msg(other: SummaryResponseOutputError)`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'user_message': FieldInfo(annotation=str, required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

user_message: `str`

```
class clinvar_api.models.sub_response.SummaryResponseSubmission(*, identifiers: SummaryResponseSubmissionIdentifiers,
                                                                    processing_status: str,
                                                                    clinvar_accession_version: str |
                                                                    None = None, errors:
                                                                    List[SummaryResponseError] |
                                                                    None = None, release_date: str |
                                                                    None = None, release_status: str
                                                                    | None = None)
```

Bases: `BaseModel`

clinvar_accession_version: `str | None`

errors: `List[SummaryResponseError] | None`

classmethod `from_msg(other: SummaryResponseSubmission)`

identifiers: `SummaryResponseSubmissionIdentifiers`

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'clinvar_accession_version':
FieldInfo(annotation=Union[str, NoneType], required=False), 'errors':
FieldInfo(annotation=Union[List[SummaryResponseError], NoneType], required=False),
'identifiers': FieldInfo(annotation=SummaryResponseSubmissionIdentifiers,
required=True), 'processing_status': FieldInfo(annotation=str, required=True),
'release_date': FieldInfo(annotation=Union[str, NoneType], required=False),
'release_status': FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
processing_status: str
```

```
release_date: str | None
```

```
release_status: str | None
```

```
class clinvar_api.models.sub_response.SummaryResponseSubmissionIdentifiers(*,
                                                                           clinvar_local_key:
                                                                           str,
                                                                           clinvar_accession:
                                                                           str | None = None,
                                                                           local_id: str |
                                                                           None = None,
                                                                           local_key: str |
                                                                           None = None)
```

Bases: `BaseModel`

```
clinvar_accession: str | None
```

```
clinvar_local_key: str
```

```
classmethod from_msg(other: SummaryResponseSubmissionIdentifiers)
```

```
local_id: str | None
```

```
local_key: str | None
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'clinvar_accession':
FieldInfo(annotation=Union[str, NoneType], required=False), 'clinvar_local_key':
FieldInfo(annotation=str, required=True), 'local_id':
FieldInfo(annotation=Union[str, NoneType], required=False), 'local_key':
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

Module contents

Data structures for the internal “business logic” representation.

These data structure use the `snake_case` used throughout the program.

clinvar_api.msg package

Submodules

clinvar_api.msg.query_response module

Data structures for the status update query communication.

```
class clinvar_api.msg.query_response.ErrorCode(value, names=None, *, module=None,
                                             qualname=None, type=None, start=1,
                                             boundary=None)
```

Bases: Enum

ALL_FAILURE = '2'

PARTIAL_SUCCESS = '1'

SUCCESS = '0'

```
class clinvar_api.msg.query_response.ProcessingStatus(value, names=None, *, module=None,
                                                       qualname=None, type=None, start=1,
                                                       boundary=None)
```

Bases: Enum

ERROR = 'Error'

SUCCESS = 'Success'

```
class clinvar_api.msg.query_response.SubmissionStatus(*, actions: List[SubmissionStatusActions])
```

Bases: BaseModel

Representation of server’s response to a submission status query.

actions: List[SubmissionStatusActions]

The list of actions (one element only by the docs).

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'actions':

FieldInfo(annotation=List[SubmissionStatusActions], required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces `Model.__fields__` from Pydantic V1.

```
class clinvar_api.msg.query_response.SubmissionStatusActions(*, id: str, responses:
                                                             List[SubmissionStatusResponse],
                                                             status: str, targetDb: str, updated:
                                                             datetime)
```

Bases: BaseModel

Type for SubmissionStatus entry actions[*].

id: str

Identifier of the submission

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'id': FieldInfo(annotation=str, required=True), 'responses': FieldInfo(annotation=List[SubmissionStatusResponse], required=True), 'status': FieldInfo(annotation=str, required=True), 'targetDb': FieldInfo(annotation=str, required=True), 'updated': FieldInfo(annotation=datetime, required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

responses: List[SubmissionStatusResponse]

Entries in actions[*].responses, only one entry per the docs.

status: str

Status of the submission, one of “submitted”, “processing”, “processed”, “error”

targetDb: str

Target database, usually “clinvar”

updated: datetime

Last updated time

class clinvar_api.msg.query_response.SubmissionStatusFile(*, url: str)

Bases: BaseModel

Type for SubmissionStatus entry actions[*].response[*].files[*].

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'url': FieldInfo(annotation=str, required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

url: str

File URL

class clinvar_api.msg.query_response.SubmissionStatusObject(*, accession: str | None, content: SubmissionStatusObjectContent, targetDb: str)

Bases: BaseModel

Type for SubmissionStatusObject entry in actions[*].response[*].objects[*].

accession: `str | None`

Optional object accession.

content: `SubmissionStatusObjectContent`

Object content.

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'accession': FieldInfo(annotation=Union[str, NoneType], required=True), 'content': FieldInfo(annotation=SubmissionStatusObjectContent, required=True), 'targetDb': FieldInfo(annotation=str, required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

targetDb: `str`

Target database, usually “clinvar” per the docs.

```
class clinvar_api.msg.query_response.SubmissionStatusObjectContent(*, clinvarProcessingStatus:
                                                                    str, clinvarReleaseStatus:
                                                                    str)
```

Bases: `BaseModel`

type for `SubmissionStatusObjectContent` entry in `actions[*].response[*].objects[*].content`.

clinvarProcessingStatus: `str`

Processing status

clinvarReleaseStatus: `str`

Release status

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

model_fields: `ClassVar[dict[str, FieldInfo]] = {'clinvarProcessingStatus': FieldInfo(annotation=str, required=True), 'clinvarReleaseStatus': FieldInfo(annotation=str, required=True)}`

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
class clinvar_api.msg.query_response.SubmissionStatusResponse(*, status: str, files:
                                                                List[SubmissionStatusFile],
                                                                message:
                                                                SubmissionStatusResponseMessage
                                                                | None, objects:
                                                                List[SubmissionStatusObject])
```

Bases: `BaseModel`

Type for `SubmissionStatus` entry `actions[*].response[*]`.

```
files: List[SubmissionStatusFile]
    Files

message: SubmissionStatusResponseMessage | None
    Message

model_config: ClassVar[ConfigDict] = {'frozen': True}
    Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'files':
FieldInfo(annotation=List[SubmissionStatusFile], required=True), 'message':
FieldInfo(annotation=Union[SubmissionStatusResponseMessage, NoneType],
required=True), 'objects': FieldInfo(annotation=List[SubmissionStatusObject],
required=True), 'status': FieldInfo(annotation=str, required=True)}
    Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].
    This replaces Model.__fields__ from Pydantic V1.

objects: List[SubmissionStatusObject]
    Objects

status: str
    Status, one of “processing”, “processed”, “error”,

class clinvar_api.msg.query_response.SubmissionStatusResponseMessage(*, errorCode: ErrorCode |
                                                                    None, severity: str, text:
                                                                    str)

Bases: BaseModel

Type for SubmissionStatusResponseMessage entry in actions[*].response[*].message.

errorCode: ErrorCode | None
    The error code.

model_config: ClassVar[ConfigDict] = {'frozen': True}
    Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'errorCode':
FieldInfo(annotation=Union[ErrorCode, NoneType], required=True), 'severity':
FieldInfo(annotation=str, required=True), 'text': FieldInfo(annotation=str,
required=True)}
    Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].
    This replaces Model.__fields__ from Pydantic V1.

severity: str
    The message severity.

text: str
    The message text.
```


clinvar_api.msg.sub_payload module

Data structures for encoding the submission payload.

```
class clinvar_api.msg.sub_payload.AffectedStatus(value, names=None, *, module=None,
                                                qualname=None, type=None, start=1,
                                                boundary=None)
```

Bases: Enum

NO = 'no'

NOT_APPLICABLE = 'not applicable'

NOT_PROVIDED = 'not provided'

UNKNOWN = 'unknown'

YES = 'yes'

```
class clinvar_api.msg.sub_payload.AlleleOrigin(value, names=None, *, module=None,
                                                qualname=None, type=None, start=1,
                                                boundary=None)
```

Bases: Enum

BIPARENTL = 'biparental'

DE_NOVO = 'de novo'

GERMLINE = 'germline'

INHERITED = 'inherited'

MATERNAL = 'maternal'

NOT_APPLICABLE = 'not applicable'

PATERNAL = 'paternal'

SOMATIC = 'somatic'

UNKNOWN = 'unknown'

```
class clinvar_api.msg.sub_payload.Assembly(value, names=None, *, module=None, qualname=None,
                                            type=None, start=1, boundary=None)
```

Bases: Enum

GRCH37 = 'GRCh37'

GRCH38 = 'GRCh38'

HG18 = 'hg18'

HG19 = 'hg19'

HG38 = 'hg38'

NCBI36 = 'NCBI36'

```
class clinvar_api.msg.sub_payload.Chromosome(value, names=None, *, module=None, qualname=None,
                                              type=None, start=1, boundary=None)
```

Bases: Enum

CHR1 = '1'

CHR10 = '10'

CHR11 = '11'

CHR12 = '12'

CHR13 = '13'

CHR14 = '14'

CHR15 = '15'

CHR16 = '16'

CHR17 = '17'

CHR18 = '18'

CHR19 = '19'

CHR2 = '2'

CHR20 = '20'

CHR21 = '21'

CHR22 = '22'

CHR3 = '3'

CHR4 = '4'

CHR5 = '5'

CHR6 = '6'

CHR7 = '7'

CHR8 = '8'

CHR9 = '9'

CHMT = 'MT'

CHRX = 'X'

CHRY = 'Y'

```
class clinvar_api.msg.sub_payload.CitationDb(value, names=None, *, module=None, qualname=None,
                                              type=None, start=1, boundary=None)
```

Bases: Enum

Allowed values for a citation's *db* field.

The values of the enumeration map to the values used by the ClinVar submission API.

```
BOOKSHELF = 'BookShelf'
```

```
DOI = 'DOI'
```

```
PMC = 'pmc'
```

```
PUBMED = 'PubMed'
```

```
class clinvar_api.msg.sub_payload.ClinicalFeaturesAffectedStatus(value, names=None, *,
                                                                module=None,
                                                                qualname=None, type=None,
                                                                start=1, boundary=None)
```

Bases: Enum

```
ABSENT = 'absent'
```

```
NOT_TESTED = 'not tested'
```

```
PRESENT = 'present'
```

```
class clinvar_api.msg.sub_payload.ClinicalFeaturesDb(value, names=None, *, module=None,
                                                    qualname=None, type=None, start=1,
                                                    boundary=None)
```

Bases: Enum

```
HP = 'HP'
```

```
class clinvar_api.msg.sub_payload.ClinicalSignificanceDescription(value, names=None, *,
                                                                module=None,
                                                                qualname=None, type=None,
                                                                start=1, boundary=None)
```

Bases: Enum

Allowed values for the clinicalSignificanceDescription.

The values of the enumeration map to the values used by the ClinVar submission API.

```
AFFECTED = 'affects'
```

```
ASSOCIATION = 'association'
```

```
BENIGN = 'Benign'
```

```
CONFERS_SENSITIVITY = 'confers sensitivity'
```

```
DRUG_RESPONSE = 'drug response'
```

```
ESTABLISHED_RISK_ALLELE = 'Established risk allele'
```

```
LIKELY_BENIGN = 'Likely benign'
```

```
LIKELY_PATHOGENIC = 'Likely pathogenic'
```

```
LIKELY_PATHOGENIC_LOW_PENETRANCE = 'Likely pathogenic, low penetrance'
```

```
LIKELY_RISK_ALLELE = 'Likely risk allele'
```

```
NOT_PROVIDED = 'not provided'
```

```
OTHER = 'other'
```

PATHOGENIC = 'Pathogenic'

PATHOGENIC_LOW_PENETRANCE = 'Pathogenic, low penetrance'

PROTECTIVE = 'protective'

UNCERTAIN_RISK_ALLELE = 'Uncertain risk allele'

UNCERTAIN_SIGNIFICANCE = 'Uncertain significance'

```
class clinvar_api.msg.sub_payload.CollectionMethod(value, names=None, *, module=None,
                                                  qualname=None, type=None, start=1,
                                                  boundary=None)
```

Bases: Enum

CASE_CONTROL = 'case-control'

CLINICAL_TESTING = 'clinical testing'

CURATION = 'curation'

IN_VITRO = 'in vitro'

IN_VIVO = 'in vivo'

LITERATURE_ONLY = 'literature only'

NOT_PROVIDED = 'not provided'

PHENOTYPING_ONLY = 'phenotyping only'

PROVIDER_INTERPRETATION = 'provider interpretation'

REFERENCE_POPULATION = 'reference population'

RESEARCH = 'research'

```
class clinvar_api.msg.sub_payload.ConditionDb(value, names=None, *, module=None, qualname=None,
                                              type=None, start=1, boundary=None)
```

Bases: Enum

HP = 'HP'

MEDGEN = 'MedGen'

MESH = 'MeSH'

MONDO = 'MONDO'

OMIM = 'OMIM'

ORPHANET = 'Orphanet'

```
class clinvar_api.msg.sub_payload.ModeOfInheritance(value, names=None, *, module=None,
                                                    qualname=None, type=None, start=1,
                                                    boundary=None)
```

Bases: Enum

AUTOSOMAL_DOMINANT_INHERITANCE = 'Autosomal dominant inheritance'

```
AUTOSOMAL_DOMINANT_INHERITANCE_WITH_MATERNAL_IMPRINTING = 'Autosomal dominant inheritance with maternal imprinting'
```

```
AUTOSOMAL_DOMINANT_INHERITANCE_WITH_PATERNAL_IMPRINTING = 'Autosomal dominant inheritance with paternal imprinting'
```

```
AUTOSOMAL_RECESSIVE_INHERITANCE = 'Autosomal recessive inheritance'
```

```
AUTOSOMAL_UNKNOWN = 'Autosomal unknown'
```

```
CODOMINANT = 'Codominant'
```

```
DEMIDOMINANT_INHERITANCE = 'Semidominant inheritance'
```

```
GENETIC_ANTICIPATION = 'Genetic anticipation'
```

```
MITOCHONDRIAL_INHERITANCE = 'Mitochondrial inheritance'
```

```
MULTIFACTORIAL_INHERITANCE = 'Multifactorial inheritance'
```

```
OLIGOGENIC_INHERITANCE = 'Oligogenic inheritance'
```

```
OTHER = 'Other'
```

```
SEX_LIMITED_AUTOSOMAL_DOMINANT = 'Sex-limited autosomal dominant'
```

```
SOMATIC_MUTATION = 'Somatic mutation'
```

```
SPORADIC = 'Sporadic'
```

```
UNKNOWN_MECHANISM = 'Unknown mechanism'
```

```
X_LINKED_DOMINANT_INHERITANCE = 'X-linked dominant inheritance'
```

```
X_LINKED_INHERITANCE = 'X-linked inheritance'
```

```
X_LINKED_RECESSIVE_INHERITANCE = 'X-linked recessive inheritance'
```

```
Y_LINKED_INHERITANCE = 'Y-linked inheritance'
```

```
class clinvar_api.msg.sub_payload.MultipleConditionExplanation(value, names=None, *,
                                                                module=None, qualname=None,
                                                                type=None, start=1,
                                                                boundary=None)
```

```
Bases: Enum
```

```
CO_OCCURRING = 'Co-occurring'
```

```
NOVEL_DISEASE = 'Novel disease'
```

```
UNCERTAIN = 'Uncertain'
```

```
class clinvar_api.msg.sub_payload.RecordStatus(value, names=None, *, module=None,
                                                qualname=None, type=None, start=1,
                                                boundary=None)
```

```
Bases: Enum
```

```
NOVEL = 'novel'
```

```
UPDATE = 'update'
```

```
class clinvar_api.msg.sub_payload.ReleaseStatus(value, names=None, *, module=None,
                                              qualname=None, type=None, start=1,
                                              boundary=None)
```

Bases: Enum

```
HOLD_UNTIL_PUBLISHED = 'hold until published'
```

```
PUBLIC = 'public'
```

```
class clinvar_api.msg.sub_payload.StructVarMethodType(value, names=None, *, module=None,
                                                    qualname=None, type=None, start=1,
                                                    boundary=None)
```

Bases: Enum

```
CURATED_PCR = 'Curated,PCR'
```

```
OLIGO_ARRAY = 'Oligo array'
```

```
ONE_END_ANCHORED_ASSEMBLY = 'One end anchored assembly'
```

```
OPTICAL_MAPPING = 'Optical mapping'
```

```
PAIRED_END_MAPPING = 'Paired-end mapping'
```

```
READ_DEPTH = 'Read depth'
```

```
SEQUENCE_ALIGNMENT = 'Sequence alignment'
```

```
SNP_ARRAY = 'SNP array'
```

```
class clinvar_api.msg.sub_payload.SubmissionAssertionCriteria(*, db: CitationDb | None = None,
                                                            id: str | None = None, url: str |
                                                            None = None)
```

Bases: BaseModel

```
db: CitationDb | None
```

```
id: str | None
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'db':
FieldInfo(annotation=Union[CitationDb, NoneType], required=False), 'id':
FieldInfo(annotation=Union[str, NoneType], required=False), 'url':
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces `Model.__fields__` from Pydantic V1.

```
url: str | None
```

```
class clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates(*, alternateAllele: str | None
                                                                = None, accession: str | None
                                                                = None, assembly: Assembly |
                                                                None = None, chromosome:
                                                                Chromosome | None = None,
                                                                innerStart: int | None = None,
                                                                innerStop: int | None = None,
                                                                outerStart: int | None = None,
                                                                outerStop: int | None = None,
                                                                referenceAllele: str | None =
                                                                None, start: int | None =
                                                                None, stop: int | None = None,
                                                                variantLength: int | None =
                                                                None)
```

Bases: BaseModel

accession: str | None

alternateAllele: str | None

assembly: Assembly | None

chromosome: Chromosome | None

innerStart: int | None

innerStop: int | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'accession':
FieldInfo(annotation=Union[str, NoneType], required=False), 'alternateAllele':
FieldInfo(annotation=Union[str, NoneType], required=False), 'assembly':
FieldInfo(annotation=Union[Assembly, NoneType], required=False), 'chromosome':
FieldInfo(annotation=Union[Chromosome, NoneType], required=False), 'innerStart':
FieldInfo(annotation=Union[int, NoneType], required=False), 'innerStop':
FieldInfo(annotation=Union[int, NoneType], required=False), 'outerStart':
FieldInfo(annotation=Union[int, NoneType], required=False), 'outerStop':
FieldInfo(annotation=Union[int, NoneType], required=False), 'referenceAllele':
FieldInfo(annotation=Union[str, NoneType], required=False), 'start':
FieldInfo(annotation=Union[int, NoneType], required=False), 'stop':
FieldInfo(annotation=Union[int, NoneType], required=False), 'variantLength':
FieldInfo(annotation=Union[int, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

outerStart: int | None

outerStop: int | None

referenceAllele: str | None

start: int | None

stop: int | None

variantLength: int | None

```
class clinvar_api.msg.sub_payload.SubmissionCitation(*, db: CitationDb | None = None, id: str | None = None, url: str | None = None)
```

Bases: BaseModel

db: [CitationDb](#) | None

id: str | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'db':

FieldInfo(annotation=Union[CitationDb, NoneType], required=False), 'id':

FieldInfo(annotation=Union[str, NoneType], required=False), 'url':

FieldInfo(annotation=Union[str, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

url: str | None

```
class clinvar_api.msg.sub_payload.SubmissionClinicalFeature(*, clinicalFeaturesAffectedStatus: ClinicalFeaturesAffectedStatus, db: ClinicalFeaturesDb | None = None, id: str | None = None, name: str | None = None)
```

Bases: BaseModel

clinicalFeaturesAffectedStatus: [ClinicalFeaturesAffectedStatus](#)

db: [ClinicalFeaturesDb](#) | None

id: str | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'clinicalFeaturesAffectedStatus':

FieldInfo(annotation=ClinicalFeaturesAffectedStatus, required=True), 'db':

FieldInfo(annotation=Union[ClinicalFeaturesDb, NoneType], required=False), 'id':

FieldInfo(annotation=Union[str, NoneType], required=False), 'name':

FieldInfo(annotation=Union[str, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

name: str | None


```
class clinvar_api.msg.sub_payload.SubmissionClinicalSignificance(*, clinicalSignificanceDescription:
    ClinicalSignificanceDescription, citation:
    List[SubmissionCitation] |
    None = None, comment: str |
    None = None,
    customAssertionScore: float |
    None = None,
    dateLastEvaluated: str | None
    = None,
    explanationOfDrugResponse:
    str | None = None,
    explanationOfOtherClinicalSignificance: str | None =
    None, modeOfInheritance:
    ModeOfInheritance | None =
    None)
```

Bases: BaseModel

citation: List[SubmissionCitation] | None

clinicalSignificanceDescription: ClinicalSignificanceDescription

comment: str | None

customAssertionScore: float | None

dateLastEvaluated: str | None

explanationOfDrugResponse: str | None

explanationOfOtherClinicalSignificance: str | None

modeOfInheritance: ModeOfInheritance | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'citation': FieldInfo(annotation=Union[List[SubmissionCitation], NoneType], required=False), 'clinicalSignificanceDescription': FieldInfo(annotation=ClinicalSignificanceDescription, required=True), 'comment': FieldInfo(annotation=Union[str, NoneType], required=False), 'customAssertionScore': FieldInfo(annotation=Union[float, NoneType], required=False), 'dateLastEvaluated': FieldInfo(annotation=Union[str, NoneType], required=False), 'explanationOfDrugResponse': FieldInfo(annotation=Union[str, NoneType], required=False), 'explanationOfOtherClinicalSignificance': FieldInfo(annotation=Union[str, NoneType], required=False), 'modeOfInheritance': FieldInfo(annotation=Union[ModeOfInheritance, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.msg.sub_payload.SubmissionClinvarDeletion(*, accessionSet:
                                                         List[SubmissionClinvarDeletionAccessionSet])
```

Bases: BaseModel

accessionSet: List[SubmissionClinvarDeletionAccessionSet]

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'accessionSet':
FieldInfo(annotation=List[SubmissionClinvarDeletionAccessionSet], required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.msg.sub_payload.SubmissionClinvarDeletionAccessionSet(*, accession: str,
                                                                           reason: str | None =
                                                                           None)
```

Bases: BaseModel

accession: str

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'accession':
FieldInfo(annotation=str, required=True), 'reason': FieldInfo(annotation=Union[str,
NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

reason: str | None

```

class clinvar_api.msg.sub_payload.SubmissionClinvarSubmission(*, clinicalSignificance:
    SubmissionClinicalSignificance,
    conditionSet:
    SubmissionConditionSet,
    observedIn:
    List[SubmissionObservedIn],
    recordStatus: RecordStatus,
    clinvarAccession: str | None =
    None, compoundHeterozygoteSet:
    SubmissionCompoundHeterozy-
    goteSet | None = None,
    diplotypeSet:
    SubmissionDiplotypeSet | None =
    None, distinctChromosomesSet:
    SubmissionDistinctChromosomes-
    Set | None = None, haplotypeSet:
    SubmissionHaplotypeSet | None =
    None, haplotypeSingleVariantSet:
    SubmissionHaplotypeSet | None =
    None, localID: str | None = None,
    localKey: str | None = None,
    phaseUnknownSet:
    SubmissionPhaseUnknownSet |
    None = None, variantSet:
    SubmissionVariantSet | None =
    None)

```

Bases: BaseModel

clinicalSignificance: *SubmissionClinicalSignificance*

clinvarAccession: str | None

compoundHeterozygoteSet: *SubmissionCompoundHeterozygoteSet* | None

conditionSet: *SubmissionConditionSet*

diplotypeSet: *SubmissionDiplotypeSet* | None

distinctChromosomesSet: *SubmissionDistinctChromosomesSet* | None

haplotypeSet: *SubmissionHaplotypeSet* | None

Has at least two elements in *variants*

haplotypeSingleVariantSet: *SubmissionHaplotypeSet* | None

Has exactly one elements in *variants*

localID: str | None

localKey: str | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [Config-Dict][pydantic.config.ConfigDict].

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'clinicalSignificance':
FieldInfo(annotation=SubmissionClinicalSignificance, required=True),
'clinvarAccession': FieldInfo(annotation=Union[str, NoneType], required=False),
'compoundHeterozygoteSet':
FieldInfo(annotation=Union[SubmissionCompoundHeterozygoteSet, NoneType],
required=False), 'conditionSet': FieldInfo(annotation=SubmissionConditionSet,
required=True), 'diplotypeSet': FieldInfo(annotation=Union[SubmissionDiplotypeSet,
NoneType], required=False), 'distinctChromosomesSet':
FieldInfo(annotation=Union[SubmissionDistinctChromosomesSet, NoneType],
required=False), 'haplotypeSet': FieldInfo(annotation=Union[SubmissionHaplotypeSet,
NoneType], required=False), 'haplotypeSingleVariantSet':
FieldInfo(annotation=Union[SubmissionHaplotypeSet, NoneType], required=False),
'localID': FieldInfo(annotation=Union[str, NoneType], required=False), 'localKey':
FieldInfo(annotation=Union[str, NoneType], required=False), 'observedIn':
FieldInfo(annotation=List[SubmissionObservedIn], required=True), 'phaseUnknownSet':
FieldInfo(annotation=Union[SubmissionPhaseUnknownSet, NoneType], required=False),
'recordStatus': FieldInfo(annotation=RecordStatus, required=True), 'variantSet':
FieldInfo(annotation=Union[SubmissionVariantSet, NoneType], required=False)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

observedIn: `List[SubmissionObservedIn]`

phaseUnknownSet: `SubmissionPhaseUnknownSet | None`

recordStatus: `RecordStatus`

variantSet: `SubmissionVariantSet | None`

```

class clinvar_api.msg.sub_payload.SubmissionCompoundHeterozygoteSet(*, hgvs: str, variantSets:
List[SubmissionCompoundHeterozygoteSetVariantSet]

```

Bases: `BaseModel`

hgvs: `str`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str,
required=True), 'variantSets':
FieldInfo(annotation=List[SubmissionCompoundHeterozygoteSetVariantSet],
required=True)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

variantSets: `List[SubmissionCompoundHeterozygoteSetVariantSet]`

```

class clinvar_api.msg.sub_payload.SubmissionCompoundHeterozygoteSetVariantSet(*, variantSet:
Submission-
VariantSet |
None = None)

```

Bases: `BaseModel`

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'variantSet':
```

```
FieldInfo(annotation=Union[SubmissionVariantSet, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
variantSet: SubmissionVariantSet | None
```

```
class clinvar_api.msg.sub_payload.SubmissionCondition(*, db: ConditionDb | None = None, id: str |
                                                    None = None, name: str | None = None)
```

Bases: `BaseModel`

```
db: ConditionDb | None
```

```
id: str | None
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'db':
```

```
FieldInfo(annotation=Union[ConditionDb, NoneType], required=False), 'id':
```

```
FieldInfo(annotation=Union[str, NoneType], required=False), 'name':
```

```
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

```
name: str | None
```

```
class clinvar_api.msg.sub_payload.SubmissionConditionSet(*, condition: List[SubmissionCondition] |
                                                         None = None, drugResponse:
                                                         List[SubmissionDrugResponse] | None =
                                                         None, multipleConditionExplanation:
                                                         MultipleConditionExplanation | None =
                                                         None)
```

Bases: `BaseModel`

```
condition: List[SubmissionCondition] | None
```

```
drugResponse: List[SubmissionDrugResponse] | None
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'condition':
```

```
FieldInfo(annotation=Union[List[SubmissionCondition], NoneType], required=False),
```

```
'drugResponse': FieldInfo(annotation=Union[List[SubmissionDrugResponse], NoneType],
required=False), 'multipleConditionExplanation':
```

```
FieldInfo(annotation=Union[MultipleConditionExplanation, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

multipleConditionExplanation: `MultipleConditionExplanation` | `None`

```
class clinvar_api.msg.sub_payload.SubmissionContainer(*, assertionCriteria:
    SubmissionAssertionCriteria | None = None,
    behalfOrgID: int | None = None,
    clinvarDeletion: SubmissionClinvarDeletion |
    None = None, clinvarSubmission:
    List[SubmissionClinvarSubmission] | None =
    None, clinvarSubmissionReleaseStatus:
    ReleaseStatus | None = None,
    submissionName: str | None = None)
```

Bases: `BaseModel`

Representation of the container for a submission.

assertionCriteria: `SubmissionAssertionCriteria` | `None`

behalfOrgID: `int` | `None`

clinvarDeletion: `SubmissionClinvarDeletion` | `None`

clinvarSubmission: `List[SubmissionClinvarSubmission]` | `None`

clinvarSubmissionReleaseStatus: `ReleaseStatus` | `None`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'assertionCriteria':
FieldInfo(annotation=Union[SubmissionAssertionCriteria, NoneType], required=False),
'behalfOrgID': FieldInfo(annotation=Union[int, NoneType], required=False),
'clinvarDeletion': FieldInfo(annotation=Union[SubmissionClinvarDeletion, NoneType],
required=False), 'clinvarSubmission':
FieldInfo(annotation=Union[List[SubmissionClinvarSubmission], NoneType],
required=False), 'clinvarSubmissionReleaseStatus':
FieldInfo(annotation=Union[ReleaseStatus, NoneType], required=False),
'submissionName': FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

submissionName: `str` | `None`

```
class clinvar_api.msg.sub_payload.SubmissionDiplotypeSet(*, haplotypeSets:
    List[SubmissionHaplotypeSets], hgvs:
    str, starAlleleName: str | None = None)
```

Bases: `BaseModel`

haplotypeSets: `List[SubmissionHaplotypeSets]`

hgvs: str

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'haplotypeSets':

FieldInfo(annotation=List[SubmissionHaplotypeSets], required=True), 'hgvs':

FieldInfo(annotation=str, required=True), 'starAlleleName':

FieldInfo(annotation=Union[str, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

starAlleleName: str | None

```
class clinvar_api.msg.sub_payload.SubmissionDistinctChromosomesSet(*, hgvs: str, variants:
                                                                    List[SubmissionVariant])
```

Bases: BaseModel

hgvs: str

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str,

required=True), 'variants': FieldInfo(annotation=List[SubmissionVariant], required=True)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

variants: List[SubmissionVariant]

Hast at least two elements

```
class clinvar_api.msg.sub_payload.SubmissionDrugResponse(*, db: ConditionDb | None = None,
                                                         drugName: str | None = None, id: str |
                                                         None = None, condition:
                                                         List[SubmissionCondition] | None =
                                                         None)
```

Bases: BaseModel

condition: List[SubmissionCondition] | None

db: ConditionDb | None

drugName: str | None

id: str | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'condition':  
FieldInfo(annotation=Union[List[SubmissionCondition], NoneType], required=False),  
'db': FieldInfo(annotation=Union[ConditionDb, NoneType], required=False),  
'drugName': FieldInfo(annotation=Union[str, NoneType], required=False), 'id':  
FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.msg.sub_payload.SubmissionHaplotypeSet(*, hgvs: str, variants:  
List[SubmissionVariant], starAlleleName:  
str | None = None)
```

Bases: BaseModel

hgvs: str

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str,  
required=True), 'starAlleleName': FieldInfo(annotation=Union[str, NoneType],  
required=False), 'variants': FieldInfo(annotation=List[SubmissionVariant],  
required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

starAlleleName: str | None

variants: List[SubmissionVariant]

```
class clinvar_api.msg.sub_payload.SubmissionHaplotypeSets(*, haplotypeSet:  
SubmissionHaplotypeSet | None = None,  
haplotypeSingleVariantSet:  
SubmissionHaplotypeSet | None =  
None)
```

Bases: BaseModel

haplotypeSet: SubmissionHaplotypeSet | None

haplotypeSingleVariantSet: SubmissionHaplotypeSet | None

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'haplotypeSet':  
FieldInfo(annotation=Union[SubmissionHaplotypeSet, NoneType], required=False),  
'haplotypeSingleVariantSet': FieldInfo(annotation=Union[SubmissionHaplotypeSet,  
NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.


```

class clinvar_api.msg.sub_payload.SubmissionObservedIn(*, affectedStatus: AffectedStatus,
                                                       alleleOrigin: AlleleOrigin,
                                                       collectionMethod: CollectionMethod,
                                                       clinicalFeatures:
                                                           List[SubmissionClinicalFeature] | None =
                                                           None, clinicalFeaturesComment: str | None =
                                                           None, numberOfIndividuals: int | None =
                                                           None, structVarMethodType:
                                                           StructVarMethodType | None = None)

Bases: BaseModel

affectedStatus: AffectedStatus

alleleOrigin: AlleleOrigin

clinicalFeatures: List[SubmissionClinicalFeature] | None

clinicalFeaturesComment: str | None

collectionMethod: CollectionMethod

model_config: ClassVar[ConfigDict] = {'frozen': True}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'affectedStatus':
FieldInfo(annotation=AffectedStatus, required=True), 'alleleOrigin':
FieldInfo(annotation=AlleleOrigin, required=True), 'clinicalFeatures':
FieldInfo(annotation=Union[List[SubmissionClinicalFeature], NoneType],
required=False), 'clinicalFeaturesComment': FieldInfo(annotation=Union[str,
NoneType], required=False), 'collectionMethod':
FieldInfo(annotation=CollectionMethod, required=True), 'numberOfIndividuals':
FieldInfo(annotation=Union[int, NoneType], required=False), 'structVarMethodType':
FieldInfo(annotation=Union[StructVarMethodType, NoneType], required=False)}
    Metadata about the fields defined on the model, mapping of field names to [Field-
    Info][pydantic.fields.FieldInfo].

    This replaces Model.__fields__ from Pydantic V1.

numberOfIndividuals: int | None

structVarMethodType: StructVarMethodType | None

class clinvar_api.msg.sub_payload.SubmissionPhaseUnknownSet(*, hgvs: str, variants:
                                                           List[SubmissionVariant])

Bases: BaseModel

hgvs: str

model_config: ClassVar[ConfigDict] = {'frozen': True}
    Configuration for the model, should be a dictionary conforming to [Config-
    Dict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'hgvs': FieldInfo(annotation=str,
required=True), 'variants': FieldInfo(annotation=List[SubmissionVariant],
required=True)}

```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

variants: `List[SubmissionVariant]`

```
class clinvar_api.msg.sub_payload.SubmissionVariant(*, chromosomeCoordinates:
    SubmissionChromosomeCoordinates | None =
    None, copyNumber: str | None = None, gene:
    List[SubmissionVariantGene] | None = None,
    hgvs: str | None = None, referenceCopyNumber:
    int | None = None, variantType: VariantType |
    None = None)
```

Bases: `BaseModel`

chromosomeCoordinates: `SubmissionChromosomeCoordinates | None`

copyNumber: `str | None`

gene: `List[SubmissionVariantGene] | None`

hgvs: `str | None`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'chromosomeCoordinates':
FieldInfo(annotation=Union[SubmissionChromosomeCoordinates, NoneType],
required=False), 'copyNumber': FieldInfo(annotation=Union[str, NoneType],
required=False), 'gene': FieldInfo(annotation=Union[List[SubmissionVariantGene],
NoneType], required=False), 'hgvs': FieldInfo(annotation=Union[str, NoneType],
required=False), 'referenceCopyNumber': FieldInfo(annotation=Union[int, NoneType],
required=False), 'variantType': FieldInfo(annotation=Union[VariantType, NoneType],
required=False)}
```

Metadata about the fields defined on the model, mapping of field names to `[FieldInfo][pydantic.fields.FieldInfo]`.

This replaces `Model.__fields__` from Pydantic V1.

referenceCopyNumber: `int | None`

variantType: `VariantType | None`

```
class clinvar_api.msg.sub_payload.SubmissionVariantGene(*, id: int | None = None, symbol: str | None
    = None)
```

Bases: `BaseModel`

id: `int | None`

model_config: `ClassVar[ConfigDict] = {'frozen': True}`

Configuration for the model, should be a dictionary conforming to `[ConfigDict][pydantic.config.ConfigDict]`.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'id':
FieldInfo(annotation=Union[int, NoneType], required=False), 'symbol':
FieldInfo(annotation=Union[str, NoneType], required=False)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

symbol: str | None

```

```

class clinvar_api.msg.sub_payload.SubmissionVariantSet(*, variant: List[SubmissionVariant])

```

Bases: BaseModel

```

model_config: ClassVar[ConfigDict] = {'frozen': True}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'variant':
FieldInfo(annotation=List[SubmissionVariant], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

variant: List[SubmissionVariant]

```

```

class clinvar_api.msg.sub_payload.VariantType(value, names=None, *, module=None, qualname=None,
type=None, start=1, boundary=None)

```

Bases: Enum

```

COMPLEX = 'Complex'

```

```

COPY_NUMBER_GAIN = 'copy number gain'

```

```

COPY_NUMBER_LOSS = 'copy number loss'

```

```

DELETION = 'Deletion'

```

```

DUPLICATION = 'Duplication'

```

```

INSERTION = 'Insertion'

```

```

INVERSION = 'Inversion'

```

```

TANDEM_DUPLICATION = 'Tandem duplication'

```

```

TRANSLOCATION = 'Translocation'

```

clinvar_api.msg.sub_response module

Data structures for parsing submission response.

```

class clinvar_api.msg.sub_response.BatchProcessingStatus(value, names=None, *, module=None,
qualname=None, type=None, start=1,
boundary=None)

```

Bases: Enum

```
ERROR = 'Error'
```

```
IN_PROCESSING = 'In processing'
```

```
PARTIAL_SUCCESS = 'Partial success'
```

```
SUCCESS = 'Success'
```

```
class clinvar_api.msg.sub_response.BatchReleaseStatus(value, names=None, *, module=None,
                                                    qualname=None, type=None, start=1,
                                                    boundary=None)
```

Bases: Enum

```
NOT_RELEASED = 'Not released'
```

```
PARTIAL_RELEASED = 'Partial released'
```

```
RELEASED = 'Released'
```

```
class clinvar_api.msg.sub_response.Created(*, id: str)
```

Bases: BaseModel

Representation of successful creation.

```
id: str
```

The submission ID.

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'id': FieldInfo(annotation=str,
required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.msg.sub_response.Error(*, message: str)
```

Bases: BaseModel

Representation of server's response in case of failure.

```
message: str
```

The error response's message.

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'message':
FieldInfo(annotation=str, required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.msg.sub_response.SummaryResponse(*, batchProcessingStatus: BatchProcessingStatus,
                                                    batchReleaseStatus: BatchReleaseStatus,
                                                    submissionDate: str, submissionName: str,
                                                    totalCount: int, totalErrors: int, totalPublic: int,
                                                    totalSuccess: int, deletions:
                                                    List[SummaryResponseDeletion] | None = None,
                                                    submissions: List[SummaryResponseSubmission]
                                                    | None = None, totalDeleteCount: int | None =
                                                    None, totalDeleted: int | None = None,
                                                    totalDeleteErrors: int | None = None,
                                                    totalDeleteSuccess: int | None = None)
```

Bases: BaseModel

Repretation of server's response to a submission.

batchProcessingStatus: [BatchProcessingStatus](#)

batchReleaseStatus: [BatchReleaseStatus](#)

deletions: [List\[SummaryResponseDeletion\]](#) | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [\[ConfigDict\]\[pydantic.config.ConfigDict\]](#).

model_fields: ClassVar[dict[str, FieldInfo]] = {'batchProcessingStatus': FieldInfo(annotation=BatchProcessingStatus, required=True), 'batchReleaseStatus': FieldInfo(annotation=BatchReleaseStatus, required=True), 'deletions': FieldInfo(annotation=Union[List[SummaryResponseDeletion], NoneType], required=False), 'submissionDate': FieldInfo(annotation=str, required=True), 'submissionName': FieldInfo(annotation=str, required=True), 'submissions': FieldInfo(annotation=Union[List[SummaryResponseSubmission], NoneType], required=False), 'totalCount': FieldInfo(annotation=int, required=True), 'totalDeleteCount': FieldInfo(annotation=Union[int, NoneType], required=False), 'totalDeleteErrors': FieldInfo(annotation=Union[int, NoneType], required=False), 'totalDeleteSuccess': FieldInfo(annotation=Union[int, NoneType], required=False), 'totalDeleted': FieldInfo(annotation=Union[int, NoneType], required=False), 'totalErrors': FieldInfo(annotation=int, required=True), 'totalPublic': FieldInfo(annotation=int, required=True), 'totalSuccess': FieldInfo(annotation=int, required=True)}

Metadata about the fields defined on the model, mapping of field names to [\[FieldInfo\]\[pydantic.fields.FieldInfo\]](#).

This replaces `Model.__fields__` from Pydantic V1.

submissionDate: str

submissionName: str

submissions: [List\[SummaryResponseSubmission\]](#) | None

totalCount: int

totalDeleteCount: int | None

totalDeleteErrors: int | None

```
totalDeleteSuccess: int | None
totalDeleted: int | None
totalErrors: int
totalPublic: int
totalSuccess: int
```

```
class clinvar_api.msg.sub_response.SummaryResponseDeletion(*, identifiers:
    SummaryResponseDeletionIdentifier,
    processingStatus: str, deleteDate: str |
    None = None, deleteStatus: str | None
    = None, errors:
    List[SummaryResponseError] | None =
    None)
```

Bases: BaseModel

```
deleteDate: str | None
```

```
deleteStatus: str | None
```

```
errors: List[SummaryResponseError] | None
```

```
identifiers: SummaryResponseDeletionIdentifier
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'deleteDate':
FieldInfo(annotation=Union[str, NoneType], required=False), 'deleteStatus':
FieldInfo(annotation=Union[str, NoneType], required=False), 'errors':
FieldInfo(annotation=Union[List[SummaryResponseError], NoneType], required=False),
'identifiers': FieldInfo(annotation=SummaryResponseDeletionIdentifier,
required=True), 'processingStatus': FieldInfo(annotation=str, required=True)}
```

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

```
processingStatus: str
```

```
class clinvar_api.msg.sub_response.SummaryResponseDeletionIdentifier(*, clinvarAccession: str,
    clinvarLocalKey: str |
    None = None)
```

Bases: BaseModel

```
clinvarAccession: str
```

```
clinvarLocalKey: str | None
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'clinvarAccession':
FieldInfo(annotation=str, required=True), 'clinvarLocalKey':
FieldInfo(annotation=Union[str, NoneType], required=False)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

class clinvar_api.msg.sub_response.SummaryResponseError(*, input:
List[SummaryResponseErrorInput],
output: SummaryResponseErrorOutput)

```

Bases: BaseModel

```

input: List[SummaryResponseErrorInput]

```

```

model_config: ClassVar[ConfigDict] = {'frozen': True}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'input':
FieldInfo(annotation=List[SummaryResponseErrorInput], required=True), 'output':
FieldInfo(annotation=SummaryResponseErrorOutput, required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

output: SummaryResponseErrorOutput

```

```

class clinvar_api.msg.sub_response.SummaryResponseErrorInput(*, value: str | None = None, field: str
| None = None)

```

Bases: BaseModel

```

field: str | None

```

```

model_config: ClassVar[ConfigDict] = {'frozen': True}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'field':
FieldInfo(annotation=Union[str, NoneType], required=False), 'value':
FieldInfo(annotation=Union[str, NoneType], required=False)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```

value: str | None

```

```

class clinvar_api.msg.sub_response.SummaryResponseErrorOutput(*, errors:
List[SummaryResponseErrorOutputError])

```

Bases: BaseModel

```

errors: List[SummaryResponseErrorOutputError]

```

```

model_config: ClassVar[ConfigDict] = {'frozen': True}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'errors':  
FieldInfo(annotation=List[SummaryResponseErrorOutputError], required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
class clinvar_api.msg.sub_response.SummaryResponseErrorOutputError(*, userMessage: str)
```

Bases: BaseModel

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'userMessage':  
FieldInfo(annotation=str, required=True)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
userMessage: str
```

```
class clinvar_api.msg.sub_response.SummaryResponseSubmission(*, identifiers: SummaryResponseSubmissionIdentifiers,  
processingStatus: str,  
clinvarAccessionVersion: str | None  
= None, errors:  
List[SummaryResponseError] |  
None = None, releaseDate: str |  
None = None, releaseStatus: str |  
None = None)
```

Bases: BaseModel

```
clinvarAccessionVersion: str | None
```

```
errors: List[SummaryResponseError] | None
```

```
identifiers: SummaryResponseSubmissionIdentifiers
```

```
model_config: ClassVar[ConfigDict] = {'frozen': True}
```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```
model_fields: ClassVar[dict[str, FieldInfo]] = {'clinvarAccessionVersion':  
FieldInfo(annotation=Union[str, NoneType], required=False), 'errors':  
FieldInfo(annotation=Union[List[SummaryResponseError], NoneType], required=False),  
'identifiers': FieldInfo(annotation=SummaryResponseSubmissionIdentifiers,  
required=True), 'processingStatus': FieldInfo(annotation=str, required=True),  
'releaseDate': FieldInfo(annotation=Union[str, NoneType], required=False),  
'releaseStatus': FieldInfo(annotation=Union[str, NoneType], required=False)}
```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

```
processingStatus: str
```


releaseDate: str | None

releaseStatus: str | None

```
class clinvar_api.msg.sub_response.SummaryResponseSubmissionIdentifiers(*, clinvarLocalKey:
                                                                    str, clinvarAccession:
                                                                    str | None = None,
                                                                    localID: str | None =
                                                                    None, localKey: str |
                                                                    None = None)
```

Bases: BaseModel

clinvarAccession: str | None

clinvarLocalKey: str

localID: str | None

localKey: str | None

model_config: ClassVar[ConfigDict] = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

model_fields: ClassVar[dict[str, FieldInfo]] = {'clinvarAccession': FieldInfo(annotation=Union[str, NoneType], required=False), 'clinvarLocalKey': FieldInfo(annotation=str, required=True), 'localID': FieldInfo(annotation=Union[str, NoneType], required=False), 'localKey': FieldInfo(annotation=Union[str, NoneType], required=False)}

Metadata about the fields defined on the model, mapping of field names to [FieldInfo][pydantic.fields.FieldInfo].

This replaces *Model.__fields__* from Pydantic V1.

Module contents

Data structures used for communicating with the server endpoints.

These data structures use the same case (camelCaseWithIDCase) as the ClinVar API JSON.

13.1.2 Submodules

13.1.3 clinvar_api.client module

REST API client code for communicating with server endpoints.

```
class clinvar_api.client.AsyncClient(config: Config)
```

Bases: object

NCBI ClinVar REST API client (async).

async retrieve_status(submission_id: str) → RetrieveStatusResult

Retrieve submission status from API.

Parameters

submission_id – The identifier of the submission as returned earlier from API.

Returns

The information about the created submission.

Raises

exceptions.QueryFailed – on problems with the communication to the server.

async submit_data(*payload: SubmissionContainer*) → *Created*

Submit new data to ClinVar API.

Parameters

payload – The submission data.

Returns

The information about the created submission.

Raises

exceptions.SubmissionFailed – on problems with the submission.

class clinvar_api.client.**Client**(*config: Config*)

Bases: object

NCBI ClinVar REST API client (sync).

retrieve_status(*submission_id: str*) → *RetrieveStatusResult*

Retrieve submission status from API.

Parameters

submission_id – The identifier of the submission as returned earlier from API.

Returns

The information about the created submission.

Raises

exceptions.QueryFailed – on problems with the communication to the server.

submit_data(*payload: SubmissionContainer*) → *Created*

Submit new data to ClinVar API.

Parameters

payload – The submission data.

Returns

The information about the created submission.

Raises

exceptions.SubmissionFailed – on problems with the submission.

class clinvar_api.client.**Config**(**, auth_token: SecretStr, use_testing: bool = False, use_dryrun: bool = False, presubmission_validation: bool = True, verify_ssl: bool = True*)

Bases: BaseModel

Configuration for the Client class.

auth_token: **SecretStr**

Token to use for authentication.

model_config: **ClassVar[ConfigDict]** = {'frozen': True}

Configuration for the model, should be a dictionary conforming to [ConfigDict][pydantic.config.ConfigDict].

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'auth_token':
FieldInfo(annotation=SecretStr, required=True), 'presubmission_validation':
FieldInfo(annotation=bool, required=False, default=True), 'use_dryrun':
FieldInfo(annotation=bool, required=False, default=False), 'use_testing':
FieldInfo(annotation=bool, required=False, default=False), 'verify_ssl':
FieldInfo(annotation=bool, required=False, default=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

presubmission_validation: bool

Whether to validate submission payload before posting.

use_dryrun: bool

Whether to use dry running.

use_testing: bool

Whether to use the test endpoint.

verify_ssl: bool

Whether or not to verify SSL on submission.

```

clinvar_api.client.ENDPOINT_URL_PROD =
'https://submit.ncbi.nlm.nih.gov/api/v1/submissions/'

```

URL of the server endpoint (non-test/production).

```

clinvar_api.client.ENDPOINT_URL_TEST =
'https://submit.ncbi.nlm.nih.gov/apitest/v1/submissions/'

```

URL of the test endpoint.

```

class clinvar_api.client.RetrieveStatusResult(*, status: SubmissionStatus, summaries: Dict[str,
SummaryResponse])

```

Bases: BaseModel

Result type for retrieve_status function.

```

model_config: ClassVar[ConfigDict] = {'frozen': True}

```

Configuration for the model, should be a dictionary conforming to *[ConfigDict][pydantic.config.ConfigDict]*.

```

model_fields: ClassVar[dict[str, FieldInfo]] = {'status':
FieldInfo(annotation=SubmissionStatus, required=True), 'summaries':
FieldInfo(annotation=Dict[str, SummaryResponse], required=True)}

```

Metadata about the fields defined on the model, mapping of field names to *[FieldInfo][pydantic.fields.FieldInfo]*.

This replaces *Model.__fields__* from Pydantic V1.

status: *SubmissionStatus*

The submission status.

summaries: Dict[str, *SummaryResponse*]

A dict mapping file URLs to the parsed Sum.

```

clinvar_api.client.SUFFIX_DRYRUN = '?dry-run=true'

```

URL suffix for enabling dry-run.

`async clinvar_api.client.async_retrieve_status(submission_id: str, config: Config) → RetrieveStatusResult`

Retrieve submission status from API.

Parameters

- **submission_id** – The identifier of the submission as returned earlier from API.
- **config** – The configuration to use.

Returns

The information about the created submission.

Raises

exceptions.QueryFailed – on problems with the communication to the server.

`async clinvar_api.client.async_submit_data(submission_container: SubmissionContainer, config: Config) → Created`

Submit new data to ClinVar API via async API (async).

Parameters

- **submission_container** – The submission data.
- **config** – The configuration to use.

Returns

The information about the created submission.

Raises

exceptions.SubmissionFailed – on problems with the submission.

`clinvar_api.client.retrieve_status(submission_id: str, config: Config) → RetrieveStatusResult`

Retrieve submission status from API.

Parameters

- **submission_id** – The identifier of the submission as returned earlier from API.
- **config** – The configuration to use.

Returns

The information about the created submission.

Raises

exceptions.QueryFailed – on problems with the communication to the server.

`clinvar_api.client.submit_data(submission_container: SubmissionContainer, config: Config) → Created`

Submit new data to ClinVar API (sync).

Parameters

- **submission_container** – The submission data.
- **config** – The configuration to use.

Returns

The information about the created submission.

Raises

exceptions.SubmissionFailed – on problems with the submission.

13.1.4 clinvar_api.common module

`clinvar_api.common.clean_for_json(value: bool | int | float | List[Any] | None | Dict[str, Any]) → bool | int | float | List[Any] | None | Dict[str, Any]`

Clean the given value for JSON submission.

13.1.5 clinvar_api.constants module

13.1.6 clinvar_api.exceptions module

Module with the exception classes.

exception `clinvar_api.exceptions.ClinvarApiException`

Bases: `Exception`

Base exception for `clinvar_api`.

exception `clinvar_api.exceptions.QueryFailed`

Bases: `ClinvarApiException`

Raised when the status query failed.

exception `clinvar_api.exceptions.SubmissionFailed`

Bases: `ClinvarApiException`

Raised when there was a problem with submitting to ClinVar.

13.1.7 Module contents

CLINVAR_THIS

14.1 clinvar_this package

14.1.1 Submodules

14.1.2 clinvar_this.cli module

Console script for ClinVar This!

14.1.3 clinvar_this.exceptions module

Exceptions used in clinvar_this module.

exception clinvar_this.exceptions.ArgumentsError

Bases: *ClinvarThisException*

Raised on problems with program arguments.

exception clinvar_this.exceptions.ClinvarThisException

Bases: Exception

Base exception class.

exception clinvar_this.exceptions.ConfigException

Bases: *ClinvarThisException*

Raised on configuration problems with clinvar-this.

exception clinvar_this.exceptions.ConfigFileMissingException

Bases: *ClinvarThisException*

Raised if the configuration file is missing.

exception clinvar_this.exceptions.IOException

Bases: *ClinvarThisException*

Raised on problems with I/O in clinvar-this.

exception clinvar_this.exceptions.InvalidFormat

Bases: *ClinvarThisException*

Raised on problems file contents in clinvar-this.

14.1.4 clinvar_this.tsv module

14.1.5 Module contents

Top-level package for ClinVar This!.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

C

- `clinvar_api`, 105
- `clinvar_api.client`, 101
- `clinvar_api.common`, 105
- `clinvar_api.constants`, 105
- `clinvar_api.exceptions`, 105
- `clinvar_api.models`, 73
- `clinvar_api.models.query_response`, 49
- `clinvar_api.models.sub_payload`, 53
- `clinvar_api.models.sub_response`, 67
- `clinvar_api.msg`, 101
- `clinvar_api.msg.query_response`, 73
- `clinvar_api.msg.sub_payload`, 77
- `clinvar_api.msg.sub_response`, 95
- `clinvar_this`, 108
- `clinvar_this.cli`, 107
- `clinvar_this.exceptions`, 107

Symbols

--dry-run
 clinvar-this-batch-submit command line
 option, 43

--force
 clinvar-this-batch-export command line
 option, 42

--gzip-output
 clinvar-this-data-extract-vars command
 line option, 45

--max-records
 clinvar-this-data-xml-to-jsonl command
 line option, 47

--metadata
 clinvar-this-batch-import command line
 option, 42

--name
 clinvar-this-batch-import command line
 option, 42

--needs-hpo-terms
 clinvar-this-data-gene-phenotype-links
 command line option, 46

--no-dry-run
 clinvar-this-batch-submit command line
 option, 43

--no-force
 clinvar-this-batch-export command line
 option, 42

--no-gzip-output
 clinvar-this-data-extract-vars command
 line option, 45

--no-needs-hpo-terms
 clinvar-this-data-gene-phenotype-links
 command line option, 46

--no-struct-var
 clinvar-this-batch-export command line
 option, 42

--no-testing
 clinvar-this-batch-retrieve command
 line option, 43
 clinvar-this-batch-submit command line
 option, 43

--no-verbose
 clinvar-this command line option, 41

--no-verify-ssl
 clinvar-this command line option, 41

--profile
 clinvar-this command line option, 41
 clinvar-this-config-get command line
 option, 44

--struct-var
 clinvar-this-batch-export command line
 option, 42

--thresholds
 clinvar-this-data-acmg-class-by-freq
 command line option, 45

--use-testing
 clinvar-this-batch-retrieve command
 line option, 43
 clinvar-this-batch-submit command line
 option, 43

--verbose
 clinvar-this command line option, 41

--verify-ssl
 clinvar-this command line option, 41

--version
 clinvar-this command line option, 41

-m
 clinvar-this-batch-import command line
 option, 42

A

ABSENT (*clinvar_api.msg.sub_payload.ClinicalFeaturesAffectedStatus*
 attribute), 79

accession (*clinvar_api.models.query_response.SubmissionStatusObject*
 attribute), 50

accession (*clinvar_api.models.sub_payload.SubmissionChromosomeCoord*
 attribute), 53

accession (*clinvar_api.models.sub_payload.SubmissionClinvarDeletionA*
 attribute), 57

accession (*clinvar_api.msg.query_response.SubmissionStatusObject*
 attribute), 74

accession (*clinvar_api.msg.sub_payload.SubmissionChromosomeCoordin*
 attribute), 83

`accession`(`clinvar_api.msg.sub_payload.SubmissionClinvarDeletion` attribute), 86
`accession_set`(`clinvar_api.models.sub_payload.SubmissionClinvarDeletion` attribute), 57
`accessionSet`(`clinvar_api.msg.sub_payload.SubmissionClinvarDeletion` attribute), 86
`actions`(`clinvar_api.models.query_response.SubmissionStatus` attribute), 49
`actions`(`clinvar_api.msg.query_response.SubmissionStatus` attribute), 73
`AFFECTED`(`clinvar_api.msg.sub_payload.ClinicalSignificanceDescription` attribute), 79
`affected_status`(`clinvar_api.models.sub_payload.SubmissionObservedIn` attribute), 64
`AffectedStatus`(class in `clinvar_api.msg.sub_payload`), 77
`affectedStatus`(`clinvar_api.msg.sub_payload.SubmissionObservedIn` attribute), 93
`ALL_FAILURE`(`clinvar_api.msg.query_response.ErrorCode` attribute), 73
`allele_origin`(`clinvar_api.models.sub_payload.SubmissionObservedIn` attribute), 64
`AlleleOrigin`(class in `clinvar_api.msg.sub_payload`), 77
`alleleOrigin`(`clinvar_api.msg.sub_payload.SubmissionObservedIn` attribute), 93
`alternate_allele`(`clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates` attribute), 53
`alternateAllele`(`clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates` attribute), 83
`ArgumentsError`, 107
`Assembly`(class in `clinvar_api.msg.sub_payload`), 77
`assembly`(`clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates` attribute), 53
`assembly`(`clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates` attribute), 83
`assertion_criteria`(`clinvar_api.models.sub_payload.SubmissionContainer` attribute), 61
`assertionCriteria`(`clinvar_api.msg.sub_payload.SubmissionContainer` attribute), 90
`ASSOCIATION`(`clinvar_api.msg.sub_payload.ClinicalSignificanceDescription` attribute), 79
`async_retrieve_status`() (in module `clinvar_api.client`), 103
`async_submit_data`() (in module `clinvar_api.client`), 104
`AsyncClient`(class in `clinvar_api.client`), 101
`auth_token`(`clinvar_api.client.Config` attribute), 102
`AUTOSOMAL_DOMINANT`(`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 80
`AUTOSOMAL_DOMINANT_INHERITANCE`(`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 80
`AUTOSOMAL_DOMINANT_INHERITANCE_WITH_MATERNAL_IMPRINTING`(`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 81
`AUTOSOMAL_DOMINANT_INHERITANCE_WITH_PATERNAL_IMPRINTING`(`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 81
`AUTOSOMAL_RECESSIVE_INHERITANCE`(`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 81
`AUTOSOMAL_UNKNOWN`(`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 81
B
`batch_processing_status`(`clinvar_api.models.sub_response.SummaryResponse` attribute), 68
`batch_release_status`(`clinvar_api.models.sub_response.SummaryResponse` attribute), 68
`BatchProcessingStatus`(class in `clinvar_api.msg.sub_response`), 95
`batchProcessingStatus`(`clinvar_api.msg.sub_response.SummaryResponse` attribute), 97
`BatchReleaseStatus`(class in `clinvar_api.msg.sub_response`), 96
`batchReleaseStatus`(`clinvar_api.msg.sub_response.SummaryResponse` attribute), 97
`behalf_org_id`(`clinvar_api.models.sub_payload.SubmissionContainer` attribute), 61
`behalfOrgID`(`clinvar_api.msg.sub_payload.SubmissionContainer` attribute), 90
`BENIGN`(`clinvar_api.msg.sub_payload.ClinicalSignificanceDescription` attribute), 79
`BIPARENTL`(`clinvar_api.msg.sub_payload.AlleleOrigin` attribute), 77
`BOOKSHELF`(`clinvar_api.msg.sub_payload.CitationDb` attribute), 78
C
`CASE_CONTROL`(`clinvar_api.msg.sub_payload.CollectionMethod` attribute), 80
`CHR1`(`clinvar_api.msg.sub_payload.Chromosome` attribute), 78
`CHR10`(`clinvar_api.msg.sub_payload.Chromosome` attribute), 78
`CHR11`(`clinvar_api.msg.sub_payload.Chromosome` attribute), 78

CHR12 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR13 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR14 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR15 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR16 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR17 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR18 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR19 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR2 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR20 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR21 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR22 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR3 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR4 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR5 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR6 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR7 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR8 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHR9 (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHRMT (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 Chromosome (class in *clinvar_api.msg.sub_payload*), 77
 chromosome (*clinvar_api.models.sub_payload.SubmissionChromosomeCaplinmsgsub_payload.SubmissionClinvarSubmission attribute*), 53
 chromosome (*clinvar_api.msg.sub_payload.SubmissionChromosomeClinicalSignificanceDescription (class in clinvar_api.msg.sub_payload)*), 83
 chromosome_coordinates (clin-
var_api.models.sub_payload.SubmissionVariant attribute), 65
 chromosomeCoordinates (clin-
var_api.msg.sub_payload.SubmissionVariant attribute), 94
 CHRX (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 CHRY (*clinvar_api.msg.sub_payload.Chromosome attribute*), 78
 citation (*clinvar_api.models.sub_payload.SubmissionClinicalSignificance attribute*), 56
 citation (*clinvar_api.msg.sub_payload.SubmissionClinicalSignificance attribute*), 85
 CitationDb (class in *clinvar_api.msg.sub_payload*), 78
 clean_for_json() (in module *clinvar_api.common*), 105
 Client (class in *clinvar_api.client*), 102
 clinical_features (clin-
var_api.models.sub_payload.SubmissionObservedIn attribute), 64
 clinical_features_affected_status (clin-
var_api.models.sub_payload.SubmissionClinicalFeature attribute), 55
 clinical_features_comment (clin-
var_api.models.sub_payload.SubmissionObservedIn attribute), 64
 clinical_significance (clin-
var_api.models.sub_payload.SubmissionClinvarSubmission attribute), 58
 clinical_significance_description (clin-
var_api.models.sub_payload.SubmissionClinicalSignificance attribute), 56
 CLINICAL_TESTING (clin-
var_api.msg.sub_payload.CollectionMethod attribute), 80
 clinicalFeatures (clin-
var_api.msg.sub_payload.SubmissionObservedIn attribute), 93
 ClinicalFeaturesAffectedStatus (class in *clinvar_api.msg.sub_payload*), 79
 clinicalFeaturesAffectedStatus (clin-
var_api.msg.sub_payload.SubmissionClinicalFeature attribute), 84
 clinicalFeaturesComment (clin-
var_api.msg.sub_payload.SubmissionObservedIn attribute), 93
 ClinicalFeaturesDb (class in *clinvar_api.msg.sub_payload*), 79
 clinicalSignificance (clin-
var_api.msg.sub_payload.SubmissionClinvarSubmission attribute), 87
 clinicalSignificanceDescription (class in *clinvar_api.msg.sub_payload*), 79
 clinicalSignificanceDescription (clin-
var_api.msg.sub_payload.SubmissionClinicalSignificance attribute), 85
 clinvar_accession (clin-
var_api.models.sub_payload.SubmissionClinvarSubmission attribute), 58
 clinvar_accession (clin-
var_api.models.sub_response.SummaryResponseDeletionIdentification attribute), 69

clinvar_accession	(clin-	module, 108
var_api.models.sub_response.SummaryResponseSubmissionIdentifier		module, 107
attribute), 72		
clinvar_accession_version	(clin-	clinvar_this.exceptions
var_api.models.sub_response.SummaryResponseSubmissionIdentifier		module, 107
attribute), 71		clinvar-this command line option
clinvar_api		--no-verbose, 41
module, 105		--no-verify-ssl, 41
clinvar_api.client		--profile, 41
module, 101		--verbose, 41
clinvar_api.common		--verify-ssl, 41
module, 105		--version, 41
clinvar_api.constants		clinvar-this-batch-export command line
module, 105		option
clinvar_api.exceptions		--force, 42
module, 105		--no-force, 42
clinvar_api.models		--no-struct-var, 42
module, 73		--struct-var, 42
clinvar_api.models.query_response		NAME, 42
module, 49		PATH, 42
clinvar_api.models.sub_payload		clinvar-this-batch-import command line
module, 53		option
clinvar_api.models.sub_response		--metadata, 42
module, 67		--name, 42
clinvar_api.msg		-m, 42
module, 101		PATH, 42
clinvar_api.msg.query_response		clinvar-this-batch-retrieve command line
module, 73		option
clinvar_api.msg.sub_payload		--no-testing, 43
module, 77		--use-testing, 43
clinvar_api.msg.sub_response		NAME, 43
module, 95		clinvar-this-batch-submit command line
clinvar_deletion	(clin-	option
var_api.models.sub_payload.SubmissionContainer		--dry-run, 43
attribute), 61		--no-dry-run, 43
clinvar_local_key	(clin-	--no-testing, 43
var_api.models.sub_response.SummaryResponseDeletionIdentifier		--use-testing, 43
attribute), 69		NAME, 43
clinvar_local_key	(clin-	clinvar-this-batch-update-metadata command
var_api.models.sub_response.SummaryResponseSubmissionIdentifier		line option
attribute), 72		METADATA, 44
clinvar_processing_status	(clin-	NAME, 44
var_api.models.query_response.SubmissionStatusIdentifier		clinvar-this-config-get command line option
attribute), 51		--profile, 44
clinvar_release_status	(clin-	NAME, 44
var_api.models.query_response.SubmissionStatusIdentifier		clinvar-this-config-set command line option
attribute), 51		NAME, 45
clinvar_submission	(clin-	VALUE, 45
var_api.models.sub_payload.SubmissionContainer		clinvar-this-data-acmg-class-by-freq
attribute), 61		command line option
clinvar_submission_release_status	(clin-	--thresholds, 45
var_api.models.sub_payload.SubmissionContainer		INPUT_FILE, 45
attribute), 61		OUTPUT_FILE, 45
clinvar_this		

clinvar-this-data-extract-vars command line option
 --gzip-output, 45
 --no-gzip-output, 45
 PATH_INPUT, 46
 PATH_OUTPUT_DIR, 46
 clinvar-this-data-gene-phenotype-links command line option
 --needs-hpo-terms, 46
 --no-needs-hpo-terms, 46
 INPUT_FILE, 46
 OUTPUT_FILE, 46
 clinvar-this-data-gene-variant-report command line option
 INPUT_FILE, 46
 OUTPUT_FILE, 46
 clinvar-this-data-xml-to-jsonl command line option
 --max-records, 47
 INPUT_FILE, 47
 OUTPUT_FILE, 47
 clinvarAccession (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission attribute), 87
 clinvarAccession (clinvar_api.msg.sub_response.SummaryResponseDeletionIdentifier attribute), 98
 clinvarAccession (clinvar_api.msg.sub_response.SummaryResponseSubmissionIdentifier attribute), 101
 clinvarAccessionVersion (clinvar_api.msg.sub_response.SummaryResponseSubmission attribute), 100
 ClinvarApiException, 105
 clinvarDeletion (clinvar_api.msg.sub_payload.SubmissionContainer attribute), 90
 clinvarLocalKey (clinvar_api.msg.sub_response.SummaryResponseDeletionIdentifier attribute), 98
 clinvarLocalKey (clinvar_api.msg.sub_response.SummaryResponseSubmissionIdentifier attribute), 101
 clinvarProcessingStatus (clinvar_api.msg.query_response.SubmissionStatusObject attribute), 75
 clinvarReleaseStatus (clinvar_api.msg.query_response.SubmissionStatusObject attribute), 75
 clinvarSubmission (clinvar_api.msg.sub_payload.SubmissionContainer attribute), 90
 clinvarSubmissionReleaseStatus (clinvar_api.msg.sub_payload.SubmissionContainer attribute), 90
 attribute), 90
 ClinvarThisException, 107
 CO_OCCURRING (clinvar_api.msg.sub_payload.MultipleConditionExplanation attribute), 81
 CODOMINANT (clinvar_api.msg.sub_payload.ModeOfInheritance attribute), 81
 collection_method (clinvar_api.models.sub_payload.SubmissionObservedIn attribute), 64
 CollectionMethod (class in clinvar_api.msg.sub_payload), 80
 collectionMethod (clinvar_api.msg.sub_payload.SubmissionObservedIn attribute), 93
 comment (clinvar_api.models.sub_payload.SubmissionClinicalSignificance attribute), 56
 comment (clinvar_api.msg.sub_payload.SubmissionClinicalSignificance attribute), 85
 COMPLEX (clinvar_api.msg.sub_payload.VariantType attribute), 95
 compound_heterozygote_set (clinvar_api.models.sub_payload.SubmissionClinvarSubmission attribute), 58
 compoundHeterozygoteSet (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission attribute), 87
 condition (clinvar_api.models.sub_payload.SubmissionConditionSet attribute), 61
 condition (clinvar_api.models.sub_payload.SubmissionDrugResponse attribute), 63
 condition (clinvar_api.msg.sub_payload.SubmissionConditionSet attribute), 89
 condition (clinvar_api.msg.sub_payload.SubmissionDrugResponse attribute), 91
 condition_set (clinvar_api.models.sub_payload.SubmissionClinvarSubmission attribute), 58
 ConditionDb (class in clinvar_api.msg.sub_payload), 80
 conditionOffset (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission attribute), 87
 CONFERS_SENSITIVITY (clinvar_api.msg.sub_payload.ClinicalSignificanceDescription attribute), 79
 Config (class in clinvar_api.client), 102
 ConfigException, 107
 ConfigFileMissingException, 107
 content (clinvar_api.models.query_response.SubmissionStatusObject attribute), 50
 content (clinvar_api.msg.query_response.SubmissionStatusObject attribute), 75
 copy_number (clinvar_api.models.sub_payload.SubmissionVariant attribute), 66
 COPY_NUMBER_GAIN (clinvar_api.msg.sub_payload.VariantType attribute), 90

tribute), 95
COPY_NUMBER_LOSS (clinvar_api.msg.sub_payload.VariantType attribute), 95
copyNumber (clinvar_api.msg.sub_payload.SubmissionVariantType attribute), 94
Created (class in clinvar_api.models.sub_response), 67
Created (class in clinvar_api.msg.sub_response), 96
CURATED_PCR (clinvar_api.msg.sub_payload.StructVariantMethod attribute), 82
CURATION (clinvar_api.msg.sub_payload.CollectionMethod attribute), 80
custom_assertion_score (clinvar_api.models.sub_payload.SubmissionClinicalSignificance attribute), 56
customAssertionScore (clinvar_api.msg.sub_payload.SubmissionClinicalSignificance attribute), 85
D
date_last_evaluated (clinvar_api.models.sub_payload.SubmissionClinicalSignificance attribute), 56
dateLastEvaluated (clinvar_api.msg.sub_payload.SubmissionClinicalSignificance attribute), 85
db (clinvar_api.models.sub_payload.SubmissionAssertionCriteria attribute), 53
db (clinvar_api.models.sub_payload.SubmissionCitation attribute), 54
db (clinvar_api.models.sub_payload.SubmissionClinicalFeature attribute), 55
db (clinvar_api.models.sub_payload.SubmissionCondition attribute), 60
db (clinvar_api.models.sub_payload.SubmissionDrugResponse attribute), 63
db (clinvar_api.msg.sub_payload.SubmissionAssertionCriteria attribute), 82
db (clinvar_api.msg.sub_payload.SubmissionCitation attribute), 84
db (clinvar_api.msg.sub_payload.SubmissionClinicalFeature attribute), 84
db (clinvar_api.msg.sub_payload.SubmissionCondition attribute), 89
db (clinvar_api.msg.sub_payload.SubmissionDrugResponse attribute), 91
DE_NOVO (clinvar_api.msg.sub_payload.AlleleOrigin attribute), 77
delete_date (clinvar_api.models.sub_response.SummaryResponseDeletion attribute), 69
delete_status (clinvar_api.models.sub_response.SummaryResponseDeletion attribute), 69
deleteDate (clinvar_api.msg.sub_response.SummaryResponseDeletion attribute), 98
deleteStatus (clinvar_api.msg.sub_response.SummaryResponseDeletion attribute), 98
DELETION (clinvar_api.msg.sub_payload.VariantType attribute), 95
deletions (clinvar_api.models.sub_response.SummaryResponse attribute), 68
deletions (clinvar_api.msg.sub_response.SummaryResponse attribute), 97
DEMONSTRATED_DOMINANT_INHERITANCE (clinvar_api.msg.sub_payload.ModeOfInheritance attribute), 81
diplotype_set (clinvar_api.models.sub_payload.SubmissionClinvarSubmission attribute), 58
diplotypeSet (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission attribute), 87
distinct_chromosomes_set (clinvar_api.models.sub_payload.SubmissionClinvarSubmission attribute), 58
distinctChromosomesSet (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission attribute), 87
DOI (clinvar_api.msg.sub_payload.CitationDb attribute), 79
drug_name (clinvar_api.models.sub_payload.SubmissionDrugResponse attribute), 63
drug_response (clinvar_api.models.sub_payload.SubmissionConditionSet attribute), 61
DRUG_RESPONSE (clinvar_api.msg.sub_payload.ClinicalSignificanceDescription attribute), 79
drugName (clinvar_api.msg.sub_payload.SubmissionDrugResponse attribute), 91
drugResponse (clinvar_api.msg.sub_payload.SubmissionConditionSet attribute), 89
DUPLICATION (clinvar_api.msg.sub_payload.VariantType attribute), 95
E
ENDPOINT_URL_PROD (in module clinvar_api.client), 103
ENDPOINT_URL_TEST (in module clinvar_api.client), 103
Error (class in clinvar_api.models.sub_response), 67
Error (class in clinvar_api.msg.sub_response), 96
ERROR (clinvar_api.msg.query_response.ProcessingStatus attribute), 73
ERROR (clinvar_api.msg.sub_response.BatchProcessingStatus attribute), 95
error_code (clinvar_api.models.query_response.SubmissionStatusResponse attribute), 52
ErrorCode (class in clinvar_api.msg.query_response), 96
errorCode (clinvar_api.msg.query_response.SubmissionStatusResponse attribute), 76
errors (clinvar_api.models.sub_response.SummaryResponseDeletion attribute), 69

[errors \(clinvar_api.models.sub_response.SummaryResponseError class method\), 71](#)
[errors \(clinvar_api.models.sub_response.SummaryResponseError class method\), 71](#)
[errors \(clinvar_api.msg.sub_response.SummaryResponseDeletion class method\), 98](#)
[errors \(clinvar_api.msg.sub_response.SummaryResponseError class method\), 99](#)
[errors \(clinvar_api.msg.sub_response.SummaryResponseSubmission class method\), 100](#)
[ESTABLISHED_RISK_ALLELE \(clinvar_api.msg.sub_payload.ClinicalSignificanceDescription class method\), 79](#)
[explanation_of_drug_response \(clinvar_api.models.sub_payload.SubmissionClinicalSignificance class method\), 56](#)
[explanation_of_other_clinical_significance \(clinvar_api.models.sub_payload.SubmissionClinicalSignificance class method\), 56](#)
[explanationOfDrugResponse \(clinvar_api.msg.sub_payload.SubmissionClinicalSignificance class method\), 85](#)
[explanationOfOtherClinicalSignificance \(clinvar_api.msg.sub_payload.SubmissionClinicalSignificance class method\), 85](#)
[extra_data \(clinvar_api.models.sub_payload.SubmissionClinvarSubmission class method\), 58](#)
F
[field \(clinvar_api.models.sub_response.SummaryResponseAllele class method\), 70](#)
[field \(clinvar_api.msg.sub_response.SummaryResponseError class method\), 99](#)
[files \(clinvar_api.models.query_response.SubmissionStatusResponse class method\), 51](#)
[files \(clinvar_api.msg.query_response.SubmissionStatusResponse class method\), 75](#)
[from_msg\(\) \(clinvar_api.models.query_response.SubmissionStatus class method\), 49](#)
[from_msg\(\) \(clinvar_api.models.query_response.SubmissionStatus class method\), 49](#)
[from_msg\(\) \(clinvar_api.models.query_response.SubmissionStatus class method\), 50](#)
[from_msg\(\) \(clinvar_api.models.query_response.SubmissionStatus class method\), 50](#)
[from_msg\(\) \(clinvar_api.models.query_response.SubmissionStatus class method\), 51](#)
[from_msg\(\) \(clinvar_api.models.query_response.SubmissionStatus class method\), 51](#)
[from_msg\(\) \(clinvar_api.models.query_response.SubmissionStatus class method\), 52](#)
[from_msg\(\) \(clinvar_api.models.sub_response.Created class method\), 67](#)
[from_msg\(\) \(clinvar_api.models.sub_response.Error class method\), 67](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponse class method\), 68](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseDeletion class method\), 69](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseError class method\), 69](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseError class method\), 70](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseError class method\), 70](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseSubmission class method\), 71](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseSubmission class method\), 71](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseSubmission class method\), 71](#)
[from_msg\(\) \(clinvar_api.models.sub_response.SummaryResponseSubmission class method\), 72](#)
G
[gene \(clinvar_api.models.sub_payload.SubmissionVariant class method\), 66](#)
[gene \(clinvar_api.msg.sub_payload.SubmissionVariant class method\), 94](#)
[GENETIC_ANTICIPATION \(clinvar_api.msg.sub_payload.ModeOfInheritance class method\), 81](#)
GERMLINE
[field \(clinvar_api.msg.sub_payload.Assembly class method\), 77](#)
GRCH37
[files \(clinvar_api.msg.sub_payload.Assembly class method\), 77](#)
GRCH38
[files \(clinvar_api.msg.sub_payload.Assembly class method\), 77](#)
H
[haplotype_set \(clinvar_api.models.sub_payload.SubmissionClinvarSubmission class method\), 58](#)
[haplotype_set \(clinvar_api.models.sub_payload.SubmissionHaplotypeSet class method\), 64](#)
[haplotype_sets \(clinvar_api.models.sub_payload.SubmissionDiplotypeSet class method\), 62](#)
[haplotype_single_variant_set \(clinvar_api.models.sub_payload.SubmissionClinvarSubmission class method\), 58](#)
[haplotype_single_variant_set \(clinvar_api.models.sub_payload.SubmissionHaplotypeSets class method\), 87](#)
[haplotypeSet \(clinvar_api.msg.sub_payload.SubmissionClinvarSubmission class method\), 92](#)

haplotypeSets (clinvar_api.msg.sub_payload.SubmissionCompoundHaplotypeSet (attribute), 90	id (clinvar_api.models.sub_payload.SubmissionClinicalFeature (attribute), 55
haplotypeSingleVariantSet (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission (attribute), 87	id (clinvar_api.models.sub_payload.SubmissionCondition (attribute), 60
haplotypeSingleVariantSet (clinvar_api.msg.sub_payload.SubmissionHaplotypeSet (attribute), 92	id (clinvar_api.models.sub_payload.SubmissionDrugResponse (attribute), 63
HG18 (clinvar_api.msg.sub_payload.Assembly (attribute), 77	id (clinvar_api.models.sub_payload.SubmissionVariantGene (attribute), 66
HG19 (clinvar_api.msg.sub_payload.Assembly (attribute), 77	id (clinvar_api.models.sub_response.Created (attribute), 67
HG38 (clinvar_api.msg.sub_payload.Assembly (attribute), 77	id (clinvar_api.msg.query_response.SubmissionStatusActions (attribute), 74
hgvs (clinvar_api.models.sub_payload.SubmissionCompoundHaplotypeSet (attribute), 59	id (clinvar_api.msg.sub_payload.SubmissionAssertionCriteria (attribute), 82
hgvs (clinvar_api.models.sub_payload.SubmissionDiploTypeSet (attribute), 62	id (clinvar_api.msg.sub_payload.SubmissionCitation (attribute), 84
hgvs (clinvar_api.models.sub_payload.SubmissionDistinctChromosomesSet (attribute), 62	id (clinvar_api.msg.sub_payload.SubmissionClinicalFeature (attribute), 84
hgvs (clinvar_api.models.sub_payload.SubmissionHaplotypeSet (attribute), 63	id (clinvar_api.msg.sub_payload.SubmissionCondition (attribute), 89
hgvs (clinvar_api.models.sub_payload.SubmissionPhaseUnknownSet (attribute), 65	id (clinvar_api.msg.sub_payload.SubmissionDrugResponse (attribute), 91
hgvs (clinvar_api.models.sub_payload.SubmissionVariant (attribute), 66	id (clinvar_api.msg.sub_payload.SubmissionVariantGene (attribute), 94
hgvs (clinvar_api.msg.sub_payload.SubmissionCompoundHeterozygousSet (attribute), 88	id (clinvar_api.msg.sub_response.Created (attribute), 96
hgvs (clinvar_api.msg.sub_payload.SubmissionDiploTypeSet (attribute), 90	identifiers (clinvar_api.models.sub_response.SummaryResponseDeletion (attribute), 69
hgvs (clinvar_api.msg.sub_payload.SubmissionDistinctChromosomesSet (attribute), 91	identifiers (clinvar_api.models.sub_response.SummaryResponseSubmission (attribute), 71
hgvs (clinvar_api.msg.sub_payload.SubmissionHaplotypeSet (attribute), 92	identifiers (clinvar_api.msg.sub_response.SummaryResponseDeletion (attribute), 98
hgvs (clinvar_api.msg.sub_payload.SubmissionPhaseUnknownSet (attribute), 93	identifiers (clinvar_api.msg.sub_response.SummaryResponseSubmission (attribute), 100
hgvs (clinvar_api.msg.sub_payload.SubmissionVariant (attribute), 94	IN_PROCESSING (clinvar_api.msg.sub_response.BatchProcessingStatus (attribute), 96
HOLD_UNTIL_PUBLISHED (clinvar_api.msg.sub_payload.ReleaseStatus (attribute), 82	IN_VITRO (clinvar_api.msg.sub_payload.CollectionMethod (attribute), 80
HP (clinvar_api.msg.sub_payload.ClinicalFeaturesDb (attribute), 79	IN_VIVO (clinvar_api.msg.sub_payload.CollectionMethod (attribute), 80
HP (clinvar_api.msg.sub_payload.ConditionDb (attribute), 80	INHERITED (clinvar_api.msg.sub_payload.AlleleOrigin (attribute), 77
	inner_start (clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates (attribute), 54
id (clinvar_api.models.query_response.SubmissionStatusActions (attribute), 49	inner_stop (clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates (attribute), 54
id (clinvar_api.models.sub_payload.SubmissionAssertionCriteria (attribute), 53	innerStart (clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates (attribute), 83
id (clinvar_api.models.sub_payload.SubmissionCitation (attribute), 54	innerStop (clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates (attribute), 83
	input (clinvar_api.models.sub_response.SummaryResponseError (attribute), 70
	input (clinvar_api.msg.sub_response.SummaryResponseError (attribute), 99
	INPUT_FILE

clinvar-this-data-acmg-class-by-freq
 command line option, 45
 clinvar-this-data-gene-phenotype-links
 command line option, 46
 clinvar-this-data-gene-variant-report
 command line option, 46
 clinvar-this-data-xml-to-jsonl command
 line option, 47
 INSERTION (clinvar_api.msg.sub_payload.VariantType
 attribute), 95
 InvalidFormat, 107
 INVERSION (clinvar_api.msg.sub_payload.VariantType
 attribute), 95
 IOException, 107
L
 LIKELY_BENIGN (clinvar_api.msg.sub_payload.ClinicalSignificanceDescription
 attribute), 79
 LIKELY_PATHOGENIC (clin-
 var_api.msg.sub_payload.ClinicalSignificanceDescription
 attribute), 79
 LIKELY_PATHOGENIC_LOW_PENETRANCE (clin-
 var_api.msg.sub_payload.ClinicalSignificanceDescription
 attribute), 79
 LIKELY_RISK_ALLELE (clin-
 var_api.msg.sub_payload.ClinicalSignificanceDescription
 attribute), 79
 LITERATURE_ONLY (clin-
 var_api.msg.sub_payload.CollectionMethod
 attribute), 80
 local_id (clinvar_api.models.sub_payload.SubmissionClinicalSubmission
 attribute), 58
 local_id (clinvar_api.models.sub_response.SummaryResponseSubmissionIdentifier
 attribute), 72
 local_key (clinvar_api.models.sub_payload.SubmissionClinicalSubmission
 attribute), 59
 local_key (clinvar_api.models.sub_response.SummaryResponseSubmissionIdentifier
 attribute), 72
 localID (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission
 attribute), 87
 localID (clinvar_api.msg.sub_response.SummaryResponseSubmissionIdentifier
 attribute), 101
 localKey (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission
 attribute), 87
 localKey (clinvar_api.msg.sub_response.SummaryResponseSubmissionIdentifier
 attribute), 101
M
 MATERNAL (clinvar_api.msg.sub_payload.AlleleOrigin
 attribute), 77
 MEDGEN (clinvar_api.msg.sub_payload.ConditionDb at-
 tribute), 80
 MESH (clinvar_api.msg.sub_payload.ConditionDb at-
 tribute), 80
 message (clinvar_api.models.query_response.SubmissionStatusResponse
 attribute), 51
 message (clinvar_api.models.sub_response.Error at-
 tribute), 67
 message (clinvar_api.msg.query_response.SubmissionStatusResponse
 attribute), 76
 message (clinvar_api.msg.sub_response.Error attribute),
 96
 METADATA
 clinvar-this-batch-update-metadata
 command line option, 44
 MITOCHONDRIAL_INHERITANCE (clin-
 var_api.msg.sub_payload.ModeOfInheritance
 attribute), 81
 mode_of_inheritance (clin-
 var_api.models.sub_payload.SubmissionClinicalSignificance
 attribute), 56
 model_config (clinvar_api.client.Config attribute), 102
 model_config (clinvar_api.client.RetrieveStatusResult
 attribute), 103
 model_config (clinvar_api.models.query_response.SubmissionStatus
 attribute), 49
 model_config (clinvar_api.models.query_response.SubmissionStatusAction
 attribute), 50
 model_config (clinvar_api.models.query_response.SubmissionStatusFile
 attribute), 50
 model_config (clinvar_api.models.query_response.SubmissionStatusObject
 attribute), 51
 model_config (clinvar_api.models.query_response.SubmissionStatusObject
 attribute), 51
 model_config (clinvar_api.models.query_response.SubmissionStatusResponse
 attribute), 52
 model_config (clinvar_api.models.query_response.SubmissionStatusResponse
 attribute), 52
 model_config (clinvar_api.models.sub_payload.SubmissionAssertionCriteria
 attribute), 53
 model_config (clinvar_api.models.sub_payload.SubmissionChromosomeC
 attribute), 54
 model_config (clinvar_api.models.sub_payload.SubmissionCitation
 attribute), 54
 model_config (clinvar_api.models.sub_payload.SubmissionClinicalFeature
 attribute), 55
 model_config (clinvar_api.models.sub_payload.SubmissionClinicalSignif
 attribute), 56
 model_config (clinvar_api.models.sub_payload.SubmissionClinvarDeletio
 attribute), 57
 model_config (clinvar_api.models.sub_payload.SubmissionClinvarDeletio
 attribute), 57
 model_config (clinvar_api.models.sub_payload.SubmissionClinvarSubmi
 attribute), 59
 model_config (clinvar_api.models.sub_payload.SubmissionCompoundHe
 attribute), 59
 model_config (clinvar_api.models.sub_payload.SubmissionCompoundHe
 attribute), 60

122 Index

`model_config` (clinvar_api.msg.sub_response.Created attribute), 59
`model_config` (clinvar_api.msg.sub_response.Error attribute), 96
`model_config` (clinvar_api.msg.sub_response.SummaryResponse attribute), 60
`model_config` (clinvar_api.msg.sub_response.SummaryResponseDeletion attribute), 60
`model_config` (clinvar_api.msg.sub_response.SummaryResponseDeletionFilter attribute), 98
`model_config` (clinvar_api.msg.sub_response.SummaryResponseError attribute), 62
`model_config` (clinvar_api.msg.sub_response.SummaryResponseErrorAttribute attribute), 99
`model_config` (clinvar_api.msg.sub_response.SummaryResponseErrorAttributeError attribute), 62
`model_config` (clinvar_api.msg.sub_response.SummaryResponseSubmission attribute), 63
`model_config` (clinvar_api.msg.sub_response.SummaryResponseSubmissionIdentifiers attribute), 101
`model_fields` (clinvar_api.client.Config attribute), 102
`model_fields` (clinvar_api.client.RetrieveStatusResult attribute), 103
`model_fields` (clinvar_api.models.query_response.SubmissionStatus attribute), 49
`model_fields` (clinvar_api.models.query_response.SubmissionStatusAttribute attribute), 50
`model_fields` (clinvar_api.models.query_response.SubmissionStatusFields attribute), 50
`model_fields` (clinvar_api.models.query_response.SubmissionStatusFieldsCreated attribute), 51
`model_fields` (clinvar_api.models.query_response.SubmissionStatusFieldsClient attribute), 51
`model_fields` (clinvar_api.models.query_response.SubmissionStatusFieldsResponse attribute), 52
`model_fields` (clinvar_api.models.query_response.SubmissionStatusFieldsResponseMessage attribute), 52
`model_fields` (clinvar_api.models.sub_payload.SubmissionCompoundHeader attribute), 53
`model_fields` (clinvar_api.models.sub_payload.SubmissionCompoundHeaderAttribute attribute), 54
`model_fields` (clinvar_api.models.sub_payload.SubmissionCondition attribute), 54
`model_fields` (clinvar_api.models.sub_payload.SubmissionConditionSet attribute), 55
`model_fields` (clinvar_api.models.sub_payload.SubmissionContainer attribute), 56
`model_fields` (clinvar_api.models.sub_payload.SubmissionDistinctChromosome attribute), 57
`model_fields` (clinvar_api.models.sub_payload.SubmissionDrugResponse attribute), 57
`model_fields` (clinvar_api.models.sub_payload.SubmissionHaplotypeSet attribute), 72
`model_fields` (clinvar_api.models.sub_payload.SubmissionHaplotypeSetAttribute attribute), 72
`model_fields` (clinvar_api.models.sub_payload.SubmissionObservedIn attribute), 72
`model_fields` (clinvar_api.models.sub_payload.SubmissionPhaseUnknown attribute), 72
`model_fields` (clinvar_api.models.sub_payload.SubmissionVariant attribute), 66
`model_fields` (clinvar_api.models.sub_payload.SubmissionVariantGene attribute), 66
`model_fields` (clinvar_api.models.sub_payload.SubmissionVariantSet attribute), 66
`model_fields` (clinvar_api.models.sub_response.Created attribute), 67
`model_fields` (clinvar_api.models.sub_response.Error attribute), 67
`model_fields` (clinvar_api.models.sub_response.SummaryResponse attribute), 68
`model_fields` (clinvar_api.models.sub_response.SummaryResponseDeletion attribute), 69
`model_fields` (clinvar_api.models.sub_response.SummaryResponseDeletionFilter attribute), 70
`model_fields` (clinvar_api.models.sub_response.SummaryResponseError attribute), 70
`model_fields` (clinvar_api.models.sub_response.SummaryResponseErrorAttribute attribute), 70
`model_fields` (clinvar_api.models.sub_response.SummaryResponseErrorAttributeError attribute), 71
`model_fields` (clinvar_api.models.sub_response.SummaryResponseSubmission attribute), 71
`model_fields` (clinvar_api.models.sub_response.SummaryResponseSubmissionIdentifiers attribute), 72
`model_fields` (clinvar_api.models.sub_response.SummaryResponseSubmissionIdentifiersAttribute attribute), 72
`model_fields` (clinvar_api.msg.query_response.SubmissionStatus attribute), 72

`multiple_condition_explanation` (clinvar_api.models.sub_payload.SubmissionConditionSet attribute), 61

`MultipleConditionExplanation` (class in clinvar_api.msg.sub_payload), 81

`multipleConditionExplanation` (clinvar_api.msg.sub_payload.SubmissionConditionSet attribute), 90

`NOVEL_DISEASE` (clinvar_api.msg.sub_payload.MultipleConditionExplanation attribute), 81

`number_of_individuals` (clinvar_api.models.sub_payload.SubmissionObservedIn attribute), 65

`numberOfIndividuals` (clinvar_api.msg.sub_payload.SubmissionObservedIn attribute), 93

N

NAME

`clinvar-this-batch-export` command line option, 42

`clinvar-this-batch-retrieve` command line option, 43

`clinvar-this-batch-submit` command line option, 43

`clinvar-this-batch-update-metadata` command line option, 44

`clinvar-this-config-get` command line option, 44

`clinvar-this-config-set` command line option, 45

`name` (clinvar_api.models.sub_payload.SubmissionClinicalFeature attribute), 55

`name` (clinvar_api.models.sub_payload.SubmissionConditionSet attribute), 60

`name` (clinvar_api.msg.sub_payload.SubmissionClinicalFeature attribute), 84

`name` (clinvar_api.msg.sub_payload.SubmissionConditionSet attribute), 89

`NCBI36` (clinvar_api.msg.sub_payload.Assembly attribute), 77

`NO` (clinvar_api.msg.sub_payload.AffectedStatus attribute), 77

`NOT_APPLICABLE` (clinvar_api.msg.sub_payload.AffectedStatus attribute), 77

`NOT_APPLICABLE` (clinvar_api.msg.sub_payload.AlleleOrigin attribute), 77

`NOT_PROVIDED` (clinvar_api.msg.sub_payload.AffectedStatus attribute), 77

`NOT_PROVIDED` (clinvar_api.msg.sub_payload.ClinicalSignificanceDescription attribute), 79

`NOT_PROVIDED` (clinvar_api.msg.sub_payload.CollectionMethod attribute), 80

`NOT_RELEASED` (clinvar_api.msg.sub_response.BatchReleaseStatus attribute), 96

`NOT_TESTED` (clinvar_api.msg.sub_payload.ClinicalFeature attribute), 79

`NOVEL` (clinvar_api.msg.sub_payload.RecordStatus attribute), 81

O

`objects` (clinvar_api.models.query_response.SubmissionStatusResponse attribute), 52

`objects` (clinvar_api.msg.query_response.SubmissionStatusResponse attribute), 76

`observed_in` (clinvar_api.models.sub_payload.SubmissionClinvarSubmission attribute), 59

`observedIn` (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission attribute), 88

`OLIGO_ARRAY` (clinvar_api.msg.sub_payload.StructVarMethodType attribute), 82

`OLIGOGENIC_INHERITANCE` (clinvar_api.msg.sub_payload.ModeOfInheritance attribute), 81

`OMIM` (clinvar_api.msg.sub_payload.ConditionDb attribute), 80

`ONE_END_ANCHORED_ASSEMBLY` (clinvar_api.msg.sub_payload.StructVarMethodType attribute), 82

`OPTICAL_MAPPING` (clinvar_api.msg.sub_payload.StructVarMethodType attribute), 82

`ORPHANET` (clinvar_api.msg.sub_payload.ConditionDb attribute), 80

`OTHER` (clinvar_api.msg.sub_payload.ClinicalSignificanceDescription attribute), 79

`OTHER` (clinvar_api.msg.sub_payload.ModeOfInheritance attribute), 81

`outer_start` (clinvar_api.models.sub_payload.SubmissionChromosomeCoordinate attribute), 54

`outer_stop` (clinvar_api.models.sub_payload.SubmissionChromosomeCoordinate attribute), 54

`outerStart` (clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinate attribute), 83

`outerStop` (clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinate attribute), 83

`output` (clinvar_api.models.sub_response.SummaryResponseError attribute), 70

`output` (clinvar_api.msg.sub_response.SummaryResponseError attribute), 99

`OUTPUT_FILES`

`clinvar-this-data-acmg-class-by-freq` command line option, 45

`clinvar-this-data-gene-phenotype-links` command line option, 46

- clinvar-this-data-gene-variant-report
command line option, 46
- clinvar-this-data-xml-to-jsonl command
line option, 47
- ## P
- PAIRED_END_MAPPING (clin-
var_api.msg.sub_payload.StructVarMethodType
attribute), 82
- PARTIAL_RELEASED (clin-
var_api.msg.sub_response.BatchReleaseStatus
attribute), 96
- PARTIAL_SUCCESS (clin-
var_api.msg.query_response.ErrorCode
attribute), 73
- PARTIAL_SUCCESS (clin-
var_api.msg.sub_response.BatchProcessingStatus
attribute), 96
- PATERNAL (clinvar_api.msg.sub_payload.AlleleOrigin
attribute), 77
- PATH
- clinvar-this-batch-export command line
option, 42
- clinvar-this-batch-import command line
option, 42
- PATH_INPUT
- clinvar-this-data-extract-vars command
line option, 46
- PATH_OUTPUT_DIR
- clinvar-this-data-extract-vars command
line option, 46
- PATHOGENIC (clinvar_api.msg.sub_payload.ClinicalSignificanceDescription
attribute), 79
- PATHOGENIC_LOW_PENETRANCE (clin-
var_api.msg.sub_payload.ClinicalSignificanceDescription
attribute), 80
- phase_unknown_set (clin-
var_api.models.sub_payload.SubmissionClinvarSubmission
attribute), 59
- phaseUnknownSet (clin-
var_api.msg.sub_payload.SubmissionClinvarSubmission
attribute), 88
- PHENOTYPING_ONLY (clin-
var_api.msg.sub_payload.CollectionMethod
attribute), 80
- PMC (clinvar_api.msg.sub_payload.CitationDb attribute),
79
- PRESENT (clinvar_api.msg.sub_payload.ClinicalFeaturesAffectedStatus
attribute), 79
- presubmission_validation (clin-
var_api.client.Config attribute), 103
- processing_status (clin-
var_api.models.sub_response.SummaryResponseSubmission
attribute), 69
- processing_status (clin-
var_api.models.sub_response.SummaryResponseSubmission
attribute), 72
- ProcessingStatus (class in clin-
var_api.msg.query_response), 73
- processingStatus (clin-
var_api.msg.sub_response.SummaryResponseDeletion
attribute), 98
- processingStatus (clin-
var_api.msg.sub_response.SummaryResponseSubmission
attribute), 100
- PROTECTIVE (clinvar_api.msg.sub_payload.ClinicalSignificanceDescription
attribute), 80
- PROVIDER_INTERPRETATION (clin-
var_api.msg.sub_payload.CollectionMethod
attribute), 80
- PUBLIC (clinvar_api.msg.sub_payload.ReleaseStatus at-
tribute), 82
- PUBMED (clinvar_api.msg.sub_payload.CitationDb
attribute), 79
- ## Q
- QueryFailed, 105
- ## R
- READ_DEPTH (clinvar_api.msg.sub_payload.StructVarMethodType
attribute), 82
- reason (clinvar_api.models.sub_payload.SubmissionClinvarDeletionAccession
attribute), 57
- reason (clinvar_api.msg.sub_payload.SubmissionClinvarDeletionAccession
attribute), 86
- record_status (clinvar_api.models.sub_payload.SubmissionClinvarSubmission
attribute), 59
- RecordStatus (class in clinvar_api.msg.sub_payload),
81
- recordStatus (clinvar_api.msg.sub_payload.SubmissionClinvarSubmission
attribute), 88
- reference_allele (clin-
var_api.models.sub_payload.SubmissionChromosomeCoordinate
attribute), 54
- reference_copy_number (clin-
var_api.models.sub_payload.SubmissionVariant
attribute), 66
- REFERENCE_POPULATION (clin-
var_api.msg.sub_payload.CollectionMethod
attribute), 80
- referenceAllele (clin-
var_api.msg.sub_payload.SubmissionChromosomeCoordinates
attribute), 83
- referenceCopyNumber (clin-
var_api.msg.sub_payload.SubmissionVariant
attribute), 94
- release_date (clinvar_api.models.sub_response.SummaryResponseSubmission
attribute), 72

release_status (clinvar_api.models.sub_response.SummaryResponseSubmission attribute), 72
 starAlleleName (clinvar_api.msg.sub_payload.SubmissionHaplotypeSet attribute), 92
 RELEASED (clinvar_api.msg.sub_response.BatchReleaseStatus attribute), 96
 start (clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates attribute), 54
 releaseDate (clinvar_api.msg.sub_response.SummaryResponseSubmission attribute), 100
 starAlleleName (clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates attribute), 83
 ReleaseStatus (class in clinvar_api.msg.sub_payload), 82
 status (clinvar_api.client.RetrieveStatusResult attribute), 103
 releaseStatus (clinvar_api.msg.sub_response.SummaryResponseSubmission attribute), 101
 status (clinvar_api.models.query_response.SubmissionStatusActions attribute), 50
 RESEARCH (clinvar_api.msg.sub_payload.CollectionMethodStatus attribute), 80
 status (clinvar_api.models.query_response.SubmissionStatusResponse attribute), 52
 responses (clinvar_api.models.query_response.SubmissionStatusActions attribute), 50
 status (clinvar_api.msg.query_response.SubmissionStatusActions attribute), 74
 responses (clinvar_api.msg.query_response.SubmissionStatusResponse attribute), 74
 status (clinvar_api.msg.query_response.SubmissionStatusResponse attribute), 76
 retrieve_status() (clinvar_api.client.AsyncClient method), 101
 stop (clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates attribute), 54
 retrieve_status() (clinvar_api.client.Client method), 102
 stop (clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates attribute), 83
 retrieve_status() (in module clinvar_api.client), 104
 struct_var_method_type (class in clinvar_api.models.sub_payload.SubmissionObservedIn attribute), 65
 StructVarMethodType (class in clinvar_api.msg.sub_payload), 82
 S
 structVarMethodType (class in clinvar_api.msg.sub_payload.SubmissionObservedIn attribute), 93
 SEQUENCE_ALIGNMENT (clinvar_api.msg.sub_payload.StructVarMethodType attribute), 82
 severity (clinvar_api.models.query_response.SubmissionStatusResponse attribute), 52
 submission_date (clinvar_api.models.sub_response.SummaryResponse attribute), 68
 severity (clinvar_api.msg.query_response.SubmissionStatusResponse attribute), 76
 submission_name (class in clinvar_api.models.sub_payload.SubmissionContainer attribute), 62
 SEX_LIMITED_AUTOSOMAL_DOMINANT (clinvar_api.msg.sub_payload.ModeOfInheritance attribute), 81
 submission_name (class in clinvar_api.models.sub_response.SummaryResponse attribute), 68
 SNP_ARRAY (clinvar_api.msg.sub_payload.StructVarMethodType attribute), 82
 SubmissionAssertionCriteria (class in clinvar_api.models.sub_payload), 53
 SOMATIC (clinvar_api.msg.sub_payload.AlleleOrigin attribute), 77
 SubmissionAssertionCriteria (class in clinvar_api.msg.sub_payload), 82
 SOMATIC_MUTATION (clinvar_api.msg.sub_payload.ModeOfInheritance attribute), 81
 SubmissionChromosomeCoordinates (class in clinvar_api.models.sub_payload), 53
 SPORADIC (clinvar_api.msg.sub_payload.ModeOfInheritance attribute), 81
 SubmissionChromosomeCoordinates (class in clinvar_api.msg.sub_payload), 82
 star_allele_name (clinvar_api.models.sub_payload.SubmissionDiplotypeSet attribute), 62
 SubmissionCitation (class in clinvar_api.models.sub_payload), 54
 star_allele_name (clinvar_api.models.sub_payload.SubmissionHaplotypeSet attribute), 64
 SubmissionCitation (class in clinvar_api.msg.sub_payload), 84
 starAlleleName (clinvar_api.models.sub_payload.SubmissionDiplotypeSet attribute), 91
 SubmissionClinicalFeature (class in clinvar_api.models.sub_payload), 55
 SubmissionClinicalFeature (class in clinvar_api.msg.sub_payload), 84

[SubmissionClinicalSignificance \(class in clin-var_api.models.sub_payload\)](#), 55
[SubmissionClinicalSignificance \(class in clin-var_api.msg.sub_payload\)](#), 84
[SubmissionClinvarDeletion \(class in clin-var_api.models.sub_payload\)](#), 57
[SubmissionClinvarDeletion \(class in clin-var_api.msg.sub_payload\)](#), 85
[SubmissionClinvarDeletionAccessionSet \(class in clinvar_api.models.sub_payload\)](#), 57
[SubmissionClinvarDeletionAccessionSet \(class in clinvar_api.msg.sub_payload\)](#), 86
[SubmissionClinvarSubmission \(class in clin-var_api.models.sub_payload\)](#), 57
[SubmissionClinvarSubmission \(class in clin-var_api.msg.sub_payload\)](#), 86
[SubmissionCompoundHeterozygoteSet \(class in clin-var_api.models.sub_payload\)](#), 59
[SubmissionCompoundHeterozygoteSet \(class in clin-var_api.msg.sub_payload\)](#), 88
[SubmissionCompoundHeterozygoteSetVariantSet \(class in clinvar_api.models.sub_payload\)](#), 60
[SubmissionCompoundHeterozygoteSetVariantSet \(class in clinvar_api.msg.sub_payload\)](#), 88
[SubmissionCondition \(class in clin-var_api.models.sub_payload\)](#), 60
[SubmissionCondition \(class in clin-var_api.msg.sub_payload\)](#), 89
[SubmissionConditionSet \(class in clin-var_api.models.sub_payload\)](#), 60
[SubmissionConditionSet \(class in clin-var_api.msg.sub_payload\)](#), 89
[SubmissionContainer \(class in clin-var_api.models.sub_payload\)](#), 61
[SubmissionContainer \(class in clin-var_api.msg.sub_payload\)](#), 90
[submissionDate \(clin-var_api.msg.sub_response.SummaryResponse attribute\)](#), 97
[SubmissionDiplotypeSet \(class in clin-var_api.models.sub_payload\)](#), 62
[SubmissionDiplotypeSet \(class in clin-var_api.msg.sub_payload\)](#), 90
[SubmissionDistinctChromosomesSet \(class in clin-var_api.models.sub_payload\)](#), 62
[SubmissionDistinctChromosomesSet \(class in clin-var_api.msg.sub_payload\)](#), 91
[SubmissionDrugResponse \(class in clin-var_api.models.sub_payload\)](#), 63
[SubmissionDrugResponse \(class in clin-var_api.msg.sub_payload\)](#), 91
[SubmissionFailed](#), 105
[SubmissionHaplotypeSet \(class in clin-var_api.models.sub_payload\)](#), 63
[SubmissionHaplotypeSet \(class in clin-var_api.msg.sub_payload\)](#), 92
[SubmissionHaplotypeSets \(class in clin-var_api.models.sub_payload\)](#), 64
[SubmissionHaplotypeSets \(class in clin-var_api.msg.sub_payload\)](#), 92
[submissionName \(clin-var_api.msg.sub_payload.SubmissionContainer attribute\)](#), 90
[submissionName \(clin-var_api.msg.sub_response.SummaryResponse attribute\)](#), 97
[SubmissionObservedIn \(class in clin-var_api.models.sub_payload\)](#), 64
[SubmissionObservedIn \(class in clin-var_api.msg.sub_payload\)](#), 92
[SubmissionPhaseUnknownSet \(class in clin-var_api.models.sub_payload\)](#), 65
[SubmissionPhaseUnknownSet \(class in clin-var_api.msg.sub_payload\)](#), 93
[submissions \(clinvar_api.models.sub_response.SummaryResponse attribute\)](#), 68
[submissions \(clinvar_api.msg.sub_response.SummaryResponse attribute\)](#), 97
[SubmissionStatus \(class in clin-var_api.models.query_response\)](#), 49
[SubmissionStatus \(class in clin-var_api.msg.query_response\)](#), 73
[SubmissionStatusActions \(class in clin-var_api.models.query_response\)](#), 49
[SubmissionStatusActions \(class in clin-var_api.msg.query_response\)](#), 73
[SubmissionStatusFile \(class in clin-var_api.models.query_response\)](#), 50
[SubmissionStatusFile \(class in clin-var_api.msg.query_response\)](#), 74
[SubmissionStatusObject \(class in clin-var_api.models.query_response\)](#), 50
[SubmissionStatusObject \(class in clin-var_api.msg.query_response\)](#), 74
[SubmissionStatusObjectContent \(class in clin-var_api.models.query_response\)](#), 51
[SubmissionStatusObjectContent \(class in clin-var_api.msg.query_response\)](#), 75
[SubmissionStatusResponse \(class in clin-var_api.models.query_response\)](#), 51
[SubmissionStatusResponse \(class in clin-var_api.msg.query_response\)](#), 75
[SubmissionStatusResponseMessage \(class in clin-var_api.models.query_response\)](#), 52
[SubmissionStatusResponseMessage \(class in clin-var_api.msg.query_response\)](#), 76
[SubmissionVariant \(class in clin-var_api.models.sub_payload\)](#), 65

SubmissionVariant (class in clinvar_api.msg.sub_payload), 94
 SubmissionVariantGene (class in clinvar_api.models.sub_payload), 66
 SubmissionVariantGene (class in clinvar_api.msg.sub_payload), 94
 SubmissionVariantSet (class in clinvar_api.models.sub_payload), 66
 SubmissionVariantSet (class in clinvar_api.msg.sub_payload), 95
 submit_data() (clinvar_api.client.AsyncClient method), 102
 submit_data() (clinvar_api.client.Client method), 102
 submit_data() (in module clinvar_api.client), 104
 SUCCESS (clinvar_api.msg.query_response.ErrorCode attribute), 73
 SUCCESS (clinvar_api.msg.query_response.ProcessingStatus attribute), 73
 SUCCESS (clinvar_api.msg.sub_response.BatchProcessingStatus attribute), 96
 SUFFIX_DRYRUN (in module clinvar_api.client), 103
 summaries (clinvar_api.client.RetrieveStatusResult attribute), 103
 SummaryResponse (class in clinvar_api.models.sub_response), 67
 SummaryResponse (class in clinvar_api.msg.sub_response), 96
 SummaryResponseDeletion (class in clinvar_api.models.sub_response), 69
 SummaryResponseDeletion (class in clinvar_api.msg.sub_response), 98
 SummaryResponseDeletionIdentifier (class in clinvar_api.models.sub_response), 69
 SummaryResponseDeletionIdentifier (class in clinvar_api.msg.sub_response), 98
 SummaryResponseError (class in clinvar_api.models.sub_response), 70
 SummaryResponseError (class in clinvar_api.msg.sub_response), 99
 SummaryResponseErrorInput (class in clinvar_api.models.sub_response), 70
 SummaryResponseErrorInput (class in clinvar_api.msg.sub_response), 99
 SummaryResponseErrorOutput (class in clinvar_api.models.sub_response), 71
 SummaryResponseErrorOutput (class in clinvar_api.msg.sub_response), 99
 SummaryResponseErrorOutputError (class in clinvar_api.models.sub_response), 71
 SummaryResponseErrorOutputError (class in clinvar_api.msg.sub_response), 100
 SummaryResponseSubmission (class in clinvar_api.models.sub_response), 71
 SummaryResponseSubmission (class in clinvar_api.msg.sub_response), 100
 SummaryResponseSubmissionIdentifiers (class in clinvar_api.models.sub_response), 72
 SummaryResponseSubmissionIdentifiers (class in clinvar_api.msg.sub_response), 101
 symbol (clinvar_api.models.sub_payload.SubmissionVariantGene attribute), 66
 symbol (clinvar_api.msg.sub_payload.SubmissionVariantGene attribute), 95

T

TANDEM_DUPLICATION (clinvar_api.msg.sub_payload.VariantType attribute), 95
 target_db (clinvar_api.models.query_response.SubmissionStatusActions attribute), 50
 target_db (clinvar_api.models.query_response.SubmissionStatusObject attribute), 51
 targetDb (clinvar_api.msg.query_response.SubmissionStatusActions attribute), 74
 targetDb (clinvar_api.msg.query_response.SubmissionStatusObject attribute), 75
 text (clinvar_api.models.query_response.SubmissionStatusResponseMessage attribute), 52
 text (clinvar_api.msg.query_response.SubmissionStatusResponseMessage attribute), 76
 to_msg() (clinvar_api.models.sub_payload.SubmissionAssertionCriteria method), 53
 to_msg() (clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates method), 54
 to_msg() (clinvar_api.models.sub_payload.SubmissionCitation method), 55
 to_msg() (clinvar_api.models.sub_payload.SubmissionClinicalFeature method), 55
 to_msg() (clinvar_api.models.sub_payload.SubmissionClinicalSignificance method), 56
 to_msg() (clinvar_api.models.sub_payload.SubmissionClinvarDeletion method), 57
 to_msg() (clinvar_api.models.sub_payload.SubmissionClinvarDeletionAction method), 57
 to_msg() (clinvar_api.models.sub_payload.SubmissionClinvarSubmission method), 59
 to_msg() (clinvar_api.models.sub_payload.SubmissionCompoundHeterozygosity method), 60
 to_msg() (clinvar_api.models.sub_payload.SubmissionCompoundHeterozygosity method), 60
 to_msg() (clinvar_api.models.sub_payload.SubmissionCondition method), 60
 to_msg() (clinvar_api.models.sub_payload.SubmissionConditionSet method), 61
 to_msg() (clinvar_api.models.sub_payload.SubmissionContainer method), 62
 to_msg() (clinvar_api.models.sub_payload.SubmissionDiplotypeSet method), 62

`to_msg()` (`clinvar_api.models.sub_payload.SubmissionDistributionSuccess` (set attribute), 63)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionDrugInteraction` (set attribute), 63)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionHaplotypeSet` (set attribute), 64)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionHaplotypeSet` (set attribute), 64)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionObservation` (set attribute), 65)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionPhaseUnknownSet` (set attribute), 65)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionVariant` (set attribute), 66)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionVariant` (set attribute), 66)
`to_msg()` (`clinvar_api.models.sub_payload.SubmissionVariant` (set attribute), 67)
`total_count` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 68)
`total_delete_count` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 68)
`total_delete_errors` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 69)
`total_delete_success` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 69)
`total_deleted` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 69)
`total_errors` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 69)
`total_public` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 69)
`total_success` (`clinvar_api.models.sub_response.SummaryResponse` (set attribute), 69)
`totalCount` (`clinvar_api.msg.sub_response.SummaryResponse` (set attribute), 97)
`totalDeleteCount` (`clinvar_api.msg.sub_response.SummaryResponse` (set attribute), 97)
`totalDeleted` (`clinvar_api.msg.sub_response.SummaryResponse` (set attribute), 98)
`totalDeleteErrors` (`clinvar_api.msg.sub_response.SummaryResponse` (set attribute), 97)
`totalDeleteSuccess` (`clinvar_api.msg.sub_response.SummaryResponse` (set attribute), 97)
`totalErrors` (`clinvar_api.msg.sub_response.SummaryResponse` (set attribute), 98)
`totalPublic` (`clinvar_api.msg.sub_response.SummaryResponse` (set attribute), 98)

U
`uncertain_risk_allele` (`clinvar_api.msg.sub_payload.ClinicalSignificanceDescription` (set attribute), 80)
`uncertain_significance` (`clinvar_api.msg.sub_payload.ClinicalSignificanceDescription` (set attribute), 80)
`unknown` (`clinvar_api.msg.sub_payload.AffectedStatus` (set attribute), 77)
`unknown_allele` (`clinvar_api.msg.sub_payload.AlleleOrigin` (set attribute), 77)
`unknown_mechanism` (`clinvar_api.msg.sub_payload.ModeOfInheritance` (set attribute), 81)
`update` (`clinvar_api.msg.sub_payload.RecordStatus` (set attribute), 81)
`updated` (`clinvar_api.models.query_response.SubmissionStatusActions` (set attribute), 50)
`updated` (`clinvar_api.msg.query_response.SubmissionStatusActions` (set attribute), 74)
`url` (`clinvar_api.models.query_response.SubmissionStatusFile` (set attribute), 50)

V
`value` (`clinvar_api.models.sub_payload.SubmissionAssertionCriteria` (set attribute), 53)
`value` (`clinvar_api.models.sub_payload.SubmissionCitation` (set attribute), 55)
`value` (`clinvar_api.msg.query_response.SubmissionStatusFile` (set attribute), 74)
`value` (`clinvar_api.msg.sub_payload.SubmissionAssertionCriteria` (set attribute), 82)
`value` (`clinvar_api.msg.sub_payload.SubmissionCitation` (set attribute), 84)
`use_dryrun` (`clinvar_api.client.Config` (set attribute), 103)
`use_testing` (`clinvar_api.client.Config` (set attribute), 103)
`user_message` (`clinvar_api.models.sub_response.SummaryResponseError` (set attribute), 71)
`userMessage` (`clinvar_api.msg.sub_response.SummaryResponseErrorOutput` (set attribute), 100)

V
`value` (`clinvar-this-config-set` command line option, 45)
`value` (`clinvar_api.models.sub_response.SummaryResponseErrorInput` (set attribute), 70)
`value` (`clinvar_api.msg.sub_response.SummaryResponseErrorInput` (set attribute), 99)

`variant` (`clinvar_api.models.sub_payload.SubmissionVariantSet` `var_api.msg.sub_payload.ModeOfInheritance` attribute), 67
`variant` (`clinvar_api.msg.sub_payload.SubmissionVariantSet` `(clinvar_api.msg.sub_payload.AffectedStatus` attribute), 77
`variant_length` (`clinvar_api.models.sub_payload.SubmissionChromosomeCoordinates` attribute), 54
`variant_set` (`clinvar_api.models.sub_payload.SubmissionClinvarSubmission` attribute), 59
`variant_set` (`clinvar_api.models.sub_payload.SubmissionCompoundHeterozygoteSetVariantSet` attribute), 60
`variant_sets` (`clinvar_api.models.sub_payload.SubmissionCompoundHeterozygoteSet` attribute), 60
`variant_type` (`clinvar_api.models.sub_payload.SubmissionVariant` attribute), 66
`variantLength` (`clinvar_api.msg.sub_payload.SubmissionChromosomeCoordinates` attribute), 84
`variants` (`clinvar_api.models.sub_payload.SubmissionDistinctChromosomesSet` attribute), 63
`variants` (`clinvar_api.models.sub_payload.SubmissionHaplotypeSet` attribute), 64
`variants` (`clinvar_api.models.sub_payload.SubmissionPhaseUnknownSet` attribute), 65
`variants` (`clinvar_api.msg.sub_payload.SubmissionDistinctChromosomesSet` attribute), 91
`variants` (`clinvar_api.msg.sub_payload.SubmissionHaplotypeSet` attribute), 92
`variants` (`clinvar_api.msg.sub_payload.SubmissionPhaseUnknownSet` attribute), 94
`variantSet` (`clinvar_api.msg.sub_payload.SubmissionClinvarSubmission` attribute), 88
`variantSet` (`clinvar_api.msg.sub_payload.SubmissionCompoundHeterozygoteSetVariantSet` attribute), 89
`variantSets` (`clinvar_api.msg.sub_payload.SubmissionCompoundHeterozygoteSet` attribute), 88
`VariantType` (class in `clinvar_api.msg.sub_payload`), 95
`variantType` (`clinvar_api.msg.sub_payload.SubmissionVariant` attribute), 94
`verify_ssl` (`clinvar_api.client.Config` attribute), 103

X

`X_LINKED_DOMINANT_INHERITANCE` (`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 81
`X_LINKED_INHERITANCE` (`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 81
`X_LINKED_RECESSIVE_INHERITANCE` (`clinvar_api.msg.sub_payload.ModeOfInheritance` attribute), 81

Y

`Y_LINKED_INHERITANCE` (`clin-`