

# Contents

1	<b>1 Guide for trident v1.1.6.0</b>	<b>1</b>
2	1.1 Poseidon package repositories . . . . .	1
3	1.2 Analysing your own dataset outside of the main repository . . . . .	2
4	1.3 Package creation and manipulation commands . . . . .	2
5	1.3.1 Init command . . . . .	2
6	1.3.2 Fetch command . . . . .	4
7	1.3.3 Forge command . . . . .	5
8	1.3.4 Genoconvert command . . . . .	10
9	1.3.5 Update command . . . . .	11
10	1.4 Inspection commands . . . . .	13
11	1.4.1 List command . . . . .	13
12	1.4.2 Summarise command . . . . .	14
13	1.4.3 Survey command . . . . .	15
14	1.4.4 Validate command . . . . .	15

## 1 Guide for trident v1.1.6.0

### 1.1 Poseidon package repositories

Trident generally requires Poseidon “packages” to work with (since version 0.28.0 it also supports direct interaction with “unpackaged” genotype data – see `-p` below). Most trident subcommands therefore have a central parameter, called `--baseDir` or simply `-d` to specify one or more base directories to look for packages. For example, if all Poseidon packages live inside a repository at `/path/to/poseidon/packages` you would simply say `trident <subcommand> -d /path/to/poseidon/dirs/` and trident would automatically search all subdirectories inside of the repository for valid poseidon packages (as identified by valid `POSEIDON.yml` files).

You can arrange a poseidon repository in a hierarchical way. For example:

```
/path/to/poseidon/packages
  /modern
    /2019_poseidon_package1
    /2019_poseidon_package2
  /ancient
    /...
    /...
  /Reference_Genomes
    /...
    /...
  /Archaic_Humans
    /...
    /...
```

You can use this structure to select only the level of packages you’re interested in, and you can make use of the fact that `-d` can be given multiple times.

Let’s use the `list` command to list all packages in the `modern` and `Reference_Genomes`:

```

41 trident list -d /path/to/poseidon/packages/modern \
42   -d /path/to/poseidon/packages/ReferenceGenomes --packages

```

## 43 1.2 Analysing your own dataset outside of the main repository

44 Being able to specify one or multiple repositories is often not enough, as you may have your own data to  
 45 co-analyse with the main repository. This is easy to do, as you simply need to provide your own genotype data  
 46 as yet another poseidon package to be added to your `trident list` command. For example, let's say you have  
 47 genotype data in EIGENSTRAT format (`trident` supports EIGENSTRAT and PLINK as formats.):

```

48 ~/my_project/my_project.geno
49 ~/my_project/my_project.snp
50 ~/my_project/my_project.ind

```

51 then you can make that to a skeleton Poseidon package with the `init` command. You can also do it manually by  
 52 simply adding a POSEIDON.yml file, with for example the following content:

```

53 poseidonVersion: 2.5.0
54 title: My_awesome_project
55 description: Unpublished genetic data from my awesome project
56 contributor:
57   - name: Stephan Schiffels
58     email: schiffels@institute.org
59 packageVersion: 0.1.0
60 lastModified: 2020-10-07
61 genotypeData:
62   format: EIGENSTRAT
63   genoFile: my_project.geno
64   snpFile: my_project.snp
65   indFile: my_project.ind
66   jannoFile: my_project.janno
67   bibFile: sources.bib

```

68 Two remarks: 1) all file paths are considered *relative* to the directory in which POSEIDON.yml resides. Here I  
 69 assume that you put this file into the same directory as the three genotype files. 2) Besides the genotype data  
 70 files there are two (technically optional) files referenced by this example POSEIDON.yml file: `sources.bib` and  
 71 `my_project.janno`. Of course you can add them manually - `init` automatically creates empty dummy versions.

72 Once you have set up your own “Poseidon” package (which is really only a skeleton so far), you can add it to  
 73 your `trident` analysis, by simply adding your project directory to the command using `-d`:

```

74 trident list -d /path/to/poseidon/packages/modern \
75   -d /path/to/poseidon/packages/ReferenceGenomes
76   -d ~/my_project --packages

```

## 77 1.3 Package creation and manipulation commands

### 78 1.3.1 Init command

79 `init` creates a new, valid poseidon package from genotype data files. It adds a valid POSEIDON.yml file, a dummy  
 80 .janno file for context information and an empty .bib file for literature references.

81 [Click here for command line details](#)

```
82 Usage: trident init ((-p|--genoOne ARG) | --inFormat ARG --genoFile ARG
83                   --snpFile ARG --indFile ARG) [--snpSet ARG]
84                   (-o|--outPackagePath ARG) [-n|--outPackageName ARG]
85                   [--minimal]
```

86 Create a new Poseidon package from genotype data

87  
88 Available options:

```
89 -h,--help           Show this help text
90 -p,--genoOne ARG    one of the input genotype data files. Expects .bed or
91                    .bim or .fam for PLINK and .geno or .snp or .ind for
92                    EIGENSTRAT. The other files must be in the same
93                    directory and must have the same base name
94 --inFormat ARG      the format of the input genotype data: EIGENSTRAT or
95                    PLINK (only necessary for data input with --genoFile
96                    + --snpFile + --indFile)
97 --genoFile ARG      the input geno file path
98 --snpFile ARG       the input snp file path
99 --indFile ARG       the input ind file path
100 --snpSet ARG        the snpSet of the new package: 1240K, HumanOrigins or
101                    Other. Default: Other
102 -o,--outPackagePath ARG the output package directory path
103 -n,--outPackageName ARG the output package name - this is optional: If no
104                    name is provided, then the package name defaults to
105                    the basename of the (mandatory) --outPackagePath
106                    argument
107 --minimal           should only a minimal output package be created?
```

108 The command

```
109 trident init \
110   --inFormat EIGENSTRAT/PLINK \
111   --genoFile path/to/geno_file \
112   --snpFile path/to/snp_file \
113   --indFile path/to/ind_file \
114   --snpSet 1240K|HumanOrigins|Other \
115   -o path/to/new_package_name
```

116 requires the format (--inFormat) of your input data (either EIGENSTRAT or PLINK), the paths to the respective  
117 files (--genoFile, --snpFile, --indFile), and optionally the “shape” of these files (--snpSet), so if they cover  
118 the 1240K, the HumanOrigins or an Other SNP set. A simpler interface added in trident 0.29.0 is available with  
119 -p (+ --snpSet).

	EIGENSTRAT	PLINK
genoFile	.geno	.bed
snpFile	.snp	.bim

	EIGENSTRAT	PLINK
indFile	.ind	.fam

The output package of `init` is created as a new directory `-o`, which should not already exist, and gets the package `title` corresponding to the basename of `-o`. You can also set the title explicitly with `-n`. The `--minimal` flag causes `init` to create a minimal package with a very basic `POSEIDON.yml` and no `.bib` and `.janno` files.

### 1.3.2 Fetch command

`fetch` allows to download poseidon packages from a remote poseidon server.

Click here for command line details

```
Usage: trident fetch (-d|--baseDir DIR)
        (--downloadAll |
        (--fetchFile ARG | (-f|--fetchString ARG)))
        [--remoteURL ARG] [-u|--upgrade]
```

Download data from a remote Poseidon repository

Available options:

<code>-h,--help</code>	Show this help text
<code>-d,--baseDir DIR</code>	a base directory to search for Poseidon Packages (could be a Poseidon repository)
<code>--downloadAll</code>	download all packages the server is offering
<code>--fetchFile ARG</code>	A file with a list of packages. Works just as <code>-f</code> , but multiple values can also be separated by newline, not just by comma. <code>-f</code> and <code>--fetchFile</code> can be combined.
<code>-f,--fetchString ARG</code>	List of packages to be downloaded from the remote server. Package names should be wrapped in asterisks: <code>*package_title*</code> . You can combine multiple values with comma, so for example: <code>"*package_1*, *package_2*, *package_3*"</code> . <code>fetchString</code> uses the same parser as <code>forgeString</code> , but does not allow excludes. If groups or individuals are specified, then packages which include these groups or individuals are included in the download.
<code>--remoteURL ARG</code>	URL of the remote Poseidon server (default: "https://c107-224.cloud.gwdg.de")
<code>-u,--upgrade</code>	overwrite outdated local package versions

It works with

```
trident fetch -d ... -d ... \
    -f "*package_title_1*,*package_title_2*,*package_title_3*,group_name,<Individual1>" \
    --fetchFile path/to/forgeFile
```

and the entities you want to download must be listed either in one or more simple strings with comma-separated values, which can be passed via one or multiple options `-f/--fetchString`, or in one or more text files

(**--fetchFile**). Entities are then combined from these sources. Entities are specified using a special syntax: Package titles are wrapped in asterisks: *package\_title* (see also the documentation of **forge** below), group names are spelled as is, and individual names are wrapped in angular brackets, like **<Individual1>**. Fetch will figure out which packages need to be downloaded to include all specified entities. **--downloadAll**, which can be given instead of **-f** and **--fetchFile**, causes fetch to download all packages from the server. The downloaded packages are added in the first (!) **-d** directory (which gets created if it doesn't exist), but downloads are only performed if the respective packages are not already present in an up-to-date version in any of the **-d** dirs.

Note that **trident fetch** makes most sense in combination with **trident list --remote**: First one can inspect what is available on the server, then one can create a custom fetch command.

**fetch** also has the optional arguments **--remote https://...** to name an alternative poseidon server. The default points to the [DAG server](#).

To overwrite outdated package versions with **fetch**, the **-u/--upgrade** flag has to be set. Note that many file systems do not offer a way to recover overwritten files. So be careful with this switch.

### 1.3.3 Forge command

**forge** creates new poseidon packages by extracting and merging packages, populations and individuals from your poseidon repositories.

Click here for command line details

```
Usage: trident forge ((-d|--baseDir DIR) |
                    ((-p|--genoOne ARG) | --inFormat ARG --genoFile ARG
                     --snpFile ARG --indFile ARG) [--snpSet ARG])
                    [--forgeFile ARG | (-f|--forgeString ARG)]
                    [--selectSnps ARG] [--intersect] [--outFormat ARG]
                    [--minimal] [--onlyGeno] (-o|--outPackagePath ARG)
                    [-n|--outPackageName ARG] [--no-extract]
```

Select packages, groups or individuals and create a new Poseidon package from them

Available options:

<b>-h,--help</b>	Show this help text
<b>-d,--baseDir DIR</b>	a base directory to search for Poseidon Packages (could be a Poseidon repository)
<b>-p,--genoOne ARG</b>	one of the input genotype data files. Expects .bed or .bim or .fam for PLINK and .geno or .snp or .ind for EIGENSTRAT. The other files must be in the same directory and must have the same base name
<b>--inFormat ARG</b>	the format of the input genotype data: EIGENSTRAT or PLINK (only necessary for data input with --genoFile + --snpFile + --indFile)
<b>--genoFile ARG</b>	the input geno file path
<b>--snpFile ARG</b>	the input snp file path
<b>--indFile ARG</b>	the input ind file path
<b>--snpSet ARG</b>	the snpSet of the new package: 1240K, HumanOrigins or

```

200 Other. Default: Other
201 --forgeFile ARG A file with a list of packages, groups or individual
202 samples. Works just as -f, but multiple values can
203 also be separated by newline, not just by comma.
204 Empty lines are ignored and comments start with "#",
205 so everything after "#" is ignored in one line.
206 Multiple instances of -f and --forgeFile can be
207 given. They will be evaluated according to their
208 input order on the command line.
209 -f,--forgeString ARG List of packages, groups or individual samples to be
210 combined in the output package. Packages follow the
211 syntax *package_title*, populations/groups are simply
212 group_id and individuals <individual_id>. You can
213 combine multiple values with comma, so for example:
214 "*package_1*, <individual_1>, <individual_2>,
215 group_1". Duplicates are treated as one entry.
216 Negative selection is possible by prepending "-" to
217 the entity you want to exclude (e.g. "*package_1*,
218 -<individual_1>, -group_1"). forge will apply
219 excludes and includes in order. If the first entity
220 is negative, then forge will assume you want to merge
221 all individuals in the packages found in the baseDirs
222 (except the ones explicitly excluded) before the
223 exclude entities are applied. An empty forgeString
224 (and no --forgeFile) will therefore merge all
225 available individuals.
226 --selectSnps ARG To extract specific SNPs during this forge operation,
227 provide a Snp file. Can be either Eigenstrat (file
228 ending must be '.snp') or Plink (file ending must be
229 '.bim'). When this option is set, the output package
230 will have exactly the SNPs listed in this file. Any
231 SNP not listed in the file will be excluded. If
232 option '--intersect' is also set, only the SNPs
233 overlapping between the SNP file and the forged
234 packages are output.
235 --intersect Whether to output the intersection of the genotype
236 files to be forged. The default (if this option is
237 not set) is to output the union of all SNPs, with
238 genotypes defined as missing in those packages which
239 do not have a SNP that is present in another package.
240 With this option set, the forged dataset will
241 typically have fewer SNPs, but less missingness.
242 --outFormat ARG the format of the output genotype data: EIGENSTRAT or
243 PLINK. Default: PLINK
244 --minimal should only a minimal output package be created?

```

245    --onlyGeno                    should only the resulting genotype data be returned?  
 246                                   This means the output will not be a Poseidon package  
 247    -o,--outPackagePath ARG    the output package directory path  
 248    -n,--outPackageName ARG    the output package name - this is optional: If no  
 249                                   name is provided, then the package name defaults to  
 250                                   the basename of the (mandatory) --outPackagePath  
 251                                   argument  
 252    --no-extract                Skip the selection step in forge. This will result in  
 253                                   outputting all individuals in the relevant packages,  
 254                                   and hence a superset of the requested  
 255                                   individuals/groups. It may result in better  
 256                                   performance in cases where one wants to forge entire  
 257                                   packages or almost entire packages. Note that this  
 258                                   will also ignore any ordering in the output  
 259                                   groups/individuals. With this option active,  
 260                                   individuals from the relevant packages will just be  
 261                                   written in the order that they appear in the original  
 262                                   packages.

263    forge can be used with

```
264 trident forge -d ... -d ... \
265    -f "*package_name*, group_id, <individual_id>" \
266    --forgeFile path/to/forgeFile \
267    -o path/to/new_package_name
```

268    where the entities (packages, groups/populations, individuals/samples) you want in the output package can be  
 269    denoted either as one or more simple strings with comma-separated values via one or more (-f/--forgeString)  
 270    options, or in one or more text files (--forgeFile). Because the order in which inclusions and exclusions  
 271    are given, the order strictly follows the order as these strings are given via options -f/--forgeString and  
 272    --forgeFile.

273    Including one or multiple Poseidon packages with -d is not the only way to include data for a forge operation.  
 274    It is also possible to include unpackaged genotype data directly with -p (+ --snpSet) or --inFormat +  
 275    --genoFile + --snpFile + --indFile (+ --snpSet). This makes the following example possible, where we  
 276    merge data from one Poseidon package and two genotype datasets to get a new EIGENSTRAT dataset.

```
277 trident forge \
278    -d 2017_GonzalesFortesCurrentBiology \
279    -p 2018_VeeramahPNAS/2018_VeeramahPNAS.fam \
280    --inFormat PLINK \
281    --genoFile 2017_HaberAJHG/2017_HaberAJHG.bed \
282    --snpFile 2017_HaberAJHG/2017_HaberAJHG.bim \
283    --indFile 2017_HaberAJHG/2017_HaberAJHG.fam \
284    -f "<STR241.SG>,<ERS1790729.SG>,Iberia_HG.SG" \
285    -o testpackage \
286    --outFormat EIGENSTRAT \
287    --onlyGeno
```

288 **1.3.3.1 The forge selection language** Entities in the `--forgeString` or the `--forgeFile` have to be  
289 marked in a certain way:

- 290 • Each package is surrounded by `*`, so if you want all individuals of `2019_Jeong_InnerEurasia` in the  
291 output package you would add `*2019_Jeong_InnerEurasia*` to the list.
- 292 • Groups/populations are not specially marked. So to get all individuals of the group `Swiss_Roman_period`,  
293 you would simply add `Swiss_Roman_period`.
- 294 • Individuals/samples are surrounded by `<` and `>`, so `ALA026` becomes `<ALA026>`.

295 Do not forget to wrap the `forgeString` in quotes.

296 You can use both `-f/--forgeString` and `--forgeFile` and even combine multiple of each. They are evaluated  
297 in order.

298 In the file each line is treated as a separate `forgeString`, empty lines are ignored and `#`s start comments. So this  
299 is a valid `forgeFile`:

```
300 # Packages
301 *package1*, *package2*
302
303 # Groups and individuals from other packages beyond package1 and package2
304 group1, <individual1>, group2, <individual2>, <individual3>
305
306 # group2 has two outlier individuals that should be ignored
307 -<bad_individual1> # This one has very low coverage
308 -<bad_individual2> # This one is from a different time period
```

309 By prepending `-` to the bad individuals, we can exclude them from the forged package. `forge` fig-  
310 ures out the final list of samples to include by executing all `forge`-entities in order. So an entity list  
311 `*PackageA*, -<Individual1>, GroupA` may result in a different outcome than `*PackageA*, GroupA, -<Individual1>`,  
312 depending on whether `<Individual1>` belongs to `GroupA` or not. If the `forge` entity list starts with a negative  
313 entity, or if the entity list is empty, `forge` will implicitly assume you want to include all individuals in all  
314 packages found in the `baseDirs` (except the ones explicitly excluded, of course). An empty `forgeString` will  
315 therefore merge all available individuals.

316 **1.3.3.2 Other options** Just as for `init` the output package of `forge` is created as a new directory `-o`. The  
317 title can also be explicitly defined with `-n`.

318 `--minimal` allows for the creation of a minimal output package without `.bib` and `.janno`. This might be  
319 especially useful for data analysis pipelines, where only the genotype data is required. Even more basic output  
320 comes with `--onlyGeno`, which means that only the genotype data is returned without any Poseidon package.

321 `forge` has a an optional flag `--intersect`, that defines, if the genotype data from different packages should  
322 be merged with an **union** or an **intersect** operation. The default (if this option is not set) is to output the  
323 union of all SNPs, with genotypes defined as missing in samples from packages which do not have a SNP that is  
324 present in another package. With this option set, on the other hand, the forged dataset will typically have fewer  
325 SNPs, but less missingness.

326 `--intersect` also influences the automatic determination of the `snpSet` field in the `POSEIDON.yml` file for the  
327 resulting package. If the `snpSets` of all input packages are identical, then the resulting package will just inherit  
328 this configuration. Otherwise `forge` applies the following pairwise merging logic:



Input snpSet A	Input snpSet B	--intersect	Ouput snpSet
Other	*	*	Other
1240K	HumanOrigins	True	HumanOrigins
1240K	HumanOrigins	False	1240K

329 `--selectSnps` allows to provide `forge` with a SNP file in EIGENSTRAT (`.snp`) or PLINK (`.bim`) format to  
330 create a package with a specific selection. When this option is set, the output package will have exactly the  
331 SNPs listed in this file. Any SNP not listed in the file will be excluded. If `--intersect` is also set, only the  
332 SNPs overlapping between the SNP file and the forged packages are output.

333 Merging genotype data across different data sources and file formats is tricky. `forge` is more verbose about  
334 potential issues, if the `--logMode` flag is set to `VerboseLog`.

335 **1.3.3.3 Treatment of the .janno file while merging** `forge` merges and subsets .janno files along with  
336 the genotype data. If a package lacks a .janno file, then a basic one will be created internally based on the  
337 information in the genotype data, and used for the output. Missing columns across packages will be filled with  
338 `n/a`.

339 For merging two .janno files **A** and **B** the following rules apply regarding undefined, arbitrary additional columns:

- 340 • If **A** has an additional column which is not in **B** then empty cells in the rows imported from **B** are filled  
341 with `n/a`.
- 342 • If **A** and **B** share additional columns with identical column name, then they are treated as semantically  
343 identical units and merged accordingly.
- 344 • In the resulting .janno file, all additional columns from both **A** and **B** are sorted alphabetically and  
345 appended after the normal, specified variables.

346 The following example illustrates the described behaviour:

347 **A.janno**

Poseidon_ID	Group_Name	Genetic_Sex	AdditionalColumn1	AdditionalColumn2
XXX011	POP1	M	A	D
XXX012	POP2	F	B	E
XXX013	POP1	M	C	F

348 **B.janno**

Poseidon_ID	Group_Name	Genetic_Sex	AdditionalColumn3	AdditionalColumn2
YYY022	POP5	F	G	J
YYY023	POP5	F	H	K
YYY024	POP5	M	I	L

349 **A.janno + B.janno**

Poseidon_ID	Group_Name	Genetic_Sex	AdditionalColumn1	AdditionalColumn2	AdditionalColumn3
XXX011	POP1	M	A	D	n/a
XXX012	POP2	F	B	E	n/a
XXX013	POP1	M	C	F	n/a
YYY022	POP5	F	n/a	J	G
YYY023	POP5	F	n/a	K	H
YYY024	POP5	M	n/a	L	I

### 1.3.4 Genoconvert command

**genoconvert** converts the genotype data in a Poseidon package to a different file format. The respective entries in the POSEIDON.yml file are changed accordingly.

[Click here for command line details](#)

```
Usage: trident genoconvert ((-d|--baseDir DIR) |
    ((-p|--genoOne ARG) | --inFormat ARG --genoFile ARG
    --snpFile ARG --indFile ARG) [--snpSet ARG])
    --outFormat ARG [--onlyGeno]
    [-o|--outPackagePath ARG] [--removeOld]
```

Convert the genotype data in a Poseidon package to a different file format

Available options:

<b>-h,--help</b>	Show this help text
<b>-d,--baseDir DIR</b>	a base directory to search for Poseidon Packages (could be a Poseidon repository)
<b>-p,--genoOne ARG</b>	one of the input genotype data files. Expects .bed or .bim or .fam for PLINK and .geno or .snp or .ind for EIGENSTRAT. The other files must be in the same directory and must have the same base name
<b>--inFormat ARG</b>	the format of the input genotype data: EIGENSTRAT or PLINK (only necessary for data input with --genoFile + --snpFile + --indFile)
<b>--genoFile ARG</b>	the input geno file path
<b>--snpFile ARG</b>	the input snp file path
<b>--indFile ARG</b>	the input ind file path
<b>--snpSet ARG</b>	the snpSet of the new package: 1240K, HumanOrigins or Other. Default: Other
<b>--outFormat ARG</b>	the format of the output genotype data: EIGENSTRAT or PLINK.
<b>--onlyGeno</b>	should only the resulting genotype data be returned? This means the output will not be a Poseidon package
<b>-o,--outPackagePath ARG</b>	the output package directory path - this is optional: If no path is provided, then the output is written to the directories where the input genotype data file (.bed/.geno) is stored

385     --removeOld                   Remove the old genotype files when creating the new  
386                                   ones

387 With the default setting

388 `trident genoconvert -d ... -d ... --outFormat EIGENSTRAT|PLINK`

389 all packages in `-d` will be converted to the desired `--outFormat` (either EIGENSTRAT or PLINK), if the data is  
390 not already in this format. This includes updating the respective POSEIDON.yml files.

391 The “old” data is not deleted, but kept around. That means conversion can result in a package with both PLINK  
392 and EIGENSTRAT data, but only one is linked in the POSEIDON.yml file, and that is what will be used by  
393 trident. To delete the old data in the conversion you can add the `--removeOld` flag.

394 Instead of `-d` to change Poseidon packages, the `-p` (+ `--snpSet`) or `--inFormat` + `--genoFile` + `--snpFile`  
395 + `--indFile` (+ `--snpSet`) allow to directly convert genotype data that is not wrapped in a Poseidon package  
396 and store it to a directory given in `-o`. See this example:

397 `trident genoconvert \  
398     -p 2018_Mittnik_Baltic/Mittnik_Baltic.bed \  
399     --outFormat EIGENSTRAT  
400     -o my_directory`

### 401 1.3.5 Update command

402 `update` automatically harmonizes POSEIDON.yml files of one or multiple packages if the packages were changed.  
403 This is not an automatic update from one Poseidon version to the next!

404 Click here for command line details

405 Usage: `trident update (-d|--baseDir DIR) [--poseidonVersion ARG]  
406                               [--ignorePoseidonVersion] [--versionComponent ARG]  
407                               [--noChecksumUpdate] [--newContributors ARG]  
408                               [--logText ARG] [--force]`

409 Update POSEIDON.yml files automatically

411 Available options:

412     413     -d,--baseDir DIR           a base directory to search for Poseidon Packages  
414                               (could be a Poseidon repository)  
415     --poseidonVersion ARG      Poseidon version the packages should be updated to:  
416                               e.g. "2.5.3" (default: Nothing)  
417     --ignorePoseidonVersion    Read packages even if their poseidonVersion is not  
418                               compatible with the trident version. The assumption  
419                               is, that the package is already structurally adjusted  
420                               to the trident version and only the version number is  
421                               lagging behind.  
422     --versionComponent ARG     Part of the package version number in the  
423                               POSEIDON.yml file that should be updated: Major,  
424                               Minor or Patch (see <https://semver.org>)  
425                               (default: Patch)

```

426 --noChecksumUpdate      Should update of checksums in the POSEIDON.yml file
427                          be skipped
428 --ignoreGeno            ignore SNP and GenoFile
429 --newContributors ARG   Contributors to add to the POSEIDON.yml file in the
430                          form "[Firstname Lastname](Email address);..."
431 --logText ARG           Log text for this version jump in the CHANGELOG file
432                          (default: "not specified")
433 --force                 Normally the POSEIDON.yml files are only changed if
434                          the poseidonVersion is adjusted or any of the
435                          checksums change. With --force a package version
436                          update can be triggered even if this is not the case.

```

437 It can be called with a lot of optional arguments

```

438 trident update -d ... -d ... \
439   --poseidonVersion "X.X.X" \
440   --versionComponent Major/Minor/Patch \
441   --noChecksumUpdate
442   --ignoreGeno
443   --newContributors "[Firstname Lastname](Email address);..."
444   --logText "short description of the update"
445   --force

```

446 By default `update` will not edit a package's POSEIDON.yml file, even when arguments like `--versionComponent`,  
447 `--newContributors` or `--logText` are explicitly set. This default exists to run the function on a large set of  
448 packages where only few of them were edited and need an active update. A package will only be modified by  
449 `update` if either

- 450 • any of the files with checksums (e.g. the genotype data) in it were modified,
- 451 • the `--poseidonVersion` argument differs from the `poseidonVersion` in the package's POSEIDON.yml  
452 file
- 453 • or the `--force` flag was set in `update`.

454 If any of these applies to a package in the search directory (`--baseDir/-d`), it will be updated. This includes  
455 the following steps:

- 456 • If `--poseidonVersion` is different from the `poseidonVersion` field in the package, then that will be  
457 updated.
- 458 • The `packageVersion` will be incremented. If `--versionComponent` is not set, then it falls back to `Patch`,  
459 so a change in the last position of the three digit version number. `Minor` increments the middle, and `Major`  
460 the first position (see [semantic versioning](#)).
- 461 • The `lastModified` field will be updated to the current day (based on your computer's system time).
- 462 • The contributors in `--newContributors` will be added to the `contributor` field if they're not there already.
- 463 • If any checksums changed, then they will be updated. If certain checksums are not set yet, then they will  
464 be added. The checksum update can be skipped with `--noChecksumUpdate` or partially skipped for the  
465 genotype data with `--ignoreGeno`.
- 466 • The CHANGELOG.md file will be updated with a new row for the new version and the text in `--logText`  
467 (default: "not specified"), which will be appended as the first line of the file. If no CHANGELOG.md file  
468 exists, then it will be created and referenced in the POSEIDON.yml file.

469 :heavy\_exclamation\_mark: As `update` reads and rewrites POSEIDON.yml files, it may change their inner order,  
 470 layout or even content (e.g. if they have fields which are not in the [Poseidon package definition](#)). Create a backup  
 471 of the POSEIDON.yml file before running `update` if you are uncertain.

## 472 1.4 Inspection commands

### 473 1.4.1 List command

474 `list` lists packages, groups and individuals of the datasets you use, or of the packages available on the server.

475 [Click here for command line details](#)

```
476 Usage: trident list ((-d|--baseDir DIR) | --remote [--remoteURL ARG])
477                (--packages | --groups | --individuals
478                [-j|--jannoColumn JANNO_HEADER]) [--raw]
```

479 List packages, groups or individuals from local or remote Poseidon  
 480 repositories

481 Available options:

```
483 -h,--help          Show this help text
484 -d,--baseDir DIR   a base directory to search for Poseidon Packages
485                   (could be a Poseidon repository)
486 --remote           list packages from a remote server instead the local
487                   file system
488 --remoteURL ARG    URL of the remote Poseidon server
489                   (default: "https://c107-224.cloud.gwdg.de")
490 --packages         list all packages
491 --groups           list all groups, ignoring any group names after the
492                   first as specified in the Janno-file
493 --individuals      list individuals
494 -j,--jannoColumn JANNO_HEADER
495                   list additional fields from the janno files, using
496                   the Janno column heading name, such as Country, Site,
497                   Date_C14_Uncal_BP, Endogenous, ...
498 --raw              output table as tsv without header. Useful for piping
499                   into grep or awk
500 --ignoreGeno       ignore SNP and GenoFile
```

501 To list packages from your local repositories, as seen above you can run

```
502 trident list -d ... -d ... --packages
```

503 This will yield a table like this

```
504 .------.------.------.
505 |           Title           |    Date    | Nr Individuals |
506 :=====:=====:=====:
507 | 2015_1000Genomes_1240K_haploid_pulldown | 2020-08-10 | 2535          |
508 | 2016_Mallick_SGDP1240K_diploid_pulldown | 2020-08-10 | 280           |
509 | 2018_BostonDatashare_modern_published   | 2020-08-10 | 2772          |
```

```

510 | ... | ... |
511 |-----|-----|-----|

```

so a nicely formatted table of all packages, their last update and the number of individuals in it.

To view packages on the remote server, instead of using directories to specify the locations of repositories on your system, you can use `--remote` to show packages on the remote server. For example

```
trident list --packages --remote
```

will result in a view of all published packages in our public online repository.

You can also list groups, as defined in the third column of EIGENSTRAT `.ind` files (or the first column of a PLINK `.fam` file), and individuals:

```
trident list -d ... -d ... --groups
```

```
trident list -d ... -d ... --individuals
```

The `--individuals` flag also provides a way to immediately access information from the `.janno` files on the command line. This works with the `-j/--jannoColumn` option. For example adding `--jannoColumn Country` `--jannoColumn Date_C14_Uncal_BP` to the commands above will add the `Country` and the `Date_C14_Uncal_BP` columns to the respective output tables.

Note that if you want a less fancy table, for example because you want to load this into Excel, or pipe into another command that cannot deal with the neat table layout, you can use the `--raw` option to output that table as a simple tab-delimited stream.

## 1.4.2 Summarise command

`summarise` prints some general summary statistics for a given poseidon dataset taken from the `.janno` files.

[Click here for command line details](#)

```
Usage: trident summarise (-d|--baseDir DIR) [--raw]
```

```
Get an overview over the content of one or multiple Poseidon packages
```

Available options:

```

535  -h,--help           Show this help text
536  -d,--baseDir DIR    a base directory to search for Poseidon Packages
537                      (could be a Poseidon repository)
538  --raw               output table as tsv without header. Useful for piping
539                      into grep or awk

```

You can run it with

```
trident summarise -d ... -d ...
```

which will show you context information like – among others – the number of individuals in the dataset, their sex distribution, the mean age of the samples (for ancient data) or the mean coverage on the 1240K SNP array in a table. `summarise` depends on complete `.janno` files and will silently ignore missing information for some statistics.

You can use the `--raw` option to output the summary table in a simple, tab-delimited layout.

### 547 1.4.3 Survey command

548 **survey** tries to indicate package completeness (mostly focused on .janno files) for poseidon datasets.

549 [Click here for command line details](#)

550 Usage: trident survey (-d|--baseDir DIR) [--raw]

551 Survey the degree of context information completeness for Poseidon packages

552 Available options:

553 -h,--help	Show this help text
554 -d,--baseDir DIR	a base directory to search for Poseidon Packages
555	(could be a Poseidon repository)
556 --raw	output table as tsv without header. Useful for piping
557	into grep or awk

558 Running

559 trident survey -d ... -d ...

560 will yield a table with one row for each package. See trident survey -h for a legend which cell of this table means what.

561 Again you can use the --raw option to output the survey table in a tab-delimited format.

### 564 1.4.4 Validate command

565 **validate** checks poseidon datasets for structural correctness.

566 [Click here for command line details](#)

567 Usage: trident validate (-d|--baseDir DIR) [--verbose]

568 Check one or multiple Poseidon packages for structural correctness

569 Available options:

570 -h,--help	Show this help text
571 -d,--baseDir DIR	a base directory to search for Poseidon Packages
572	(could be a Poseidon repository)
573 --ignoreGeno	ignore SNP and GenoFile
574 --noExitCode	do not produce an explicit exit code

575 You can run it with

576 trident validate -d ... -d ...

577 and it will either report a success (**Validation passed**) or failure with specific error messages to simplify fixing the issues.

578 **validate** tries to ensure that each package in the dataset adheres to the [schema definition](#). Here is a list of what is checked:

- 582 • Presence of the necessary files
- 583 • Full structural correctness of .bib and .janno file
- 584 • Superficial correctness of genotype data files. A full check would be too computationally expensive

585     • Correspondence of BibTeX keys in .bib and .janno  
586     • Correspondence of individual and group IDs in .janno and genotype data files

587 In fact much of this validation already runs as part of the general package reading pipeline invoked for many  
588 trident subcommands (e.g. **forge**). **validate** is meant to be more thorough, though, and will explicitly fail if  
589 even a single package is broken.

590 Remember to run it with **--logMode VerboseLog** to get more information if the output is not sufficient to debug  
591 an issue.