

# Administration Backend

image-1600352934446.png

## Command line Interface (BASH)

You need a PATH to `<path-to-nora>/src/node`

Call

`nora`

any get help

```
NORA -  
backend
```

```
usage: nora [--parameter]  
[value]  
--project (-p)  
[name]
```

```
all preceding commands refer to this  
project
```

```
MANAGEMENT
```

```
--sql (-q) [sqlstatement] [--  
csv]
```

executes SQL statment and return result as json (or as csv if option is given)

--add (-a) [file]

...

adds list of files to project

option: --addtag tag1,tag2,... (tags the files)

--del (-d) [file]

...

deletes list files

--del\_pat (-dp) [patient\_id] [patients\_id#studies\_id]

...

deletes list of patients/studies

--out (-o)

[filepattern]

computes absolute outputfilepath from filepattern (see description of filepattern below)

--pathout (-pa)

[filepattern]

computes absolute outfilepath from filepattern

--select (-s) [--json (-j)] [--byfilename]

[filepattern]

returns files matching filepattern, if --json is given, files are return as json filepattern

if --byfilename (together with --json) is given, files are restructered by filename

--addtag tag1,tag2,... -s

[filepattern]

add tag to file

--rmtag tag1,tag2,... -s

[filepattern]

removes tags fromfile

--launch (-l)

[json]

launches batch from json

description

--createproject (-c) [name] [module]

creates a project with [name] from [module] definition

--admin [parameter1] [parameter2] ...

calls the admin tools (see nora --admin for more help)

## PROCESSING

--launch json

launches a NORA job given by json

options: --throwerror

--handle\_job\_result (emits DONE signal, when ready)

--launchfile jsonfile

same as --launch but json given by fileref

--launch\_autoexec psid

launches the autoexecution pipeline of the project for the patient specified by psid

--import srcpath

imports dicoms located in sourcepath to given project

options: --autoexec\_queue (if given autoexecuter is initiated, use DEFAULT to select default

autoexecution queue given in config)

--handle\_job\_result (emits DONE signal, when ready)

--extrafiles\_filter: if there are non-dicom files in your dicomfolder and you

want to add them to your patient folder, then, give here a comma-separated list of accepted extensions (e.g. nii,hdr,img)

--patients\_id PIZ (overwrite patients\_id of imported data with PIZ )

--studies\_id SID (overwrite studies\_id of imported data with SID )

--pattern REGEX (use REGEX applied on srcpath of dicoms to overwrite

patients\_id/studies\_id, example:

--pattern "(?<studies\_id>[\w\.-]+)\V(?<patients\_id>\w+)\$"

## OTHER

--exportmeta [filepattern]

exports meta data as csv. Meta content from files matching [filepattern] are exported

options: --keys a comma separated list of key-sequences, which are exported [optional]

a key sequence is given by a dot-separated list, where the first key refers to the name of the json-file (as multiple jsons might be contained in the query). A wildcard \* is possible select multiple keys at once

examples:

nora -p XYZ --exportmeta '\*' 'nodeinfo.json' // gathers also patient info

```
nora -p XYZ --exportmeta '*' 'META/radio*.json' --keys '*.mask.volume,*.mask.diameter'
```

## MATLAB Interface

To setup NORA for MATLAB you change to the folder

```
<path-to-nora>/src/matlab
```

at the MATLAB prompt and start `DPX_startup`. This sets up all necessary paths and the database connection. To test the DB connection, make a call to `DPX_Project`. This should list all created projects (initially only the default project). There are a variety of management commands to manage users/project/data etc. Here a list of important ones:

- General
  - `DPX_startup` - sets up NORA
  - `DPX_Project` - choose a project
  - `DPX_getCurrentProject` - get the current project information
  - `DPX_SQL_createUser` - create a user
  - `DPX_SQL_createProject` - create a project
- File selection
  - `DPX_getOutputLocation` - compute the path to certain output
  - `DPX_selectFiles` - select files based on pattern
  - `DPX_getFileInfo` - get additional file information (no database involved)
  - `DPX_SQL_getFileInfo` - get information from database for this file
- Processing
  - `DPX_demon` - start the processing daemon
  - `DPX_debug_job` - execute a debug job at the MATLAB command prompt

`DPX_SQL_clearGridCmds` - clear all jobs from the pipeline

## The Database

There is one general database scheme containing information about users, projects, jobs, settings etc.

You can see below the table schemes.

- **Projects/User management**

- projects
- users
- rights - who can read which project in which role

- **Batch/Jobs**

- gridjobs - contains submitted and running jobs (mirror of qstat or queue)
- Commands - contains jobs submitted from web interface, but not yet submitted to resource management system.

- **Miscellaneous**

- json - various types of jsons (saved batches, shares, settings, states etc.)
- notifications
- error\_js

image1600353314610.png

image1600353338398.png

---

Revision #3

Created 17 September 2020 13:20:14 by reiser tm

Updated 28 September 2020 09:29:16 by reiser tm