BIDS Curation with `fw-heudiconv`

Comprehensive guide to curating your imaging data on Flywheel

Tinashe Michael Tapera

August 2019

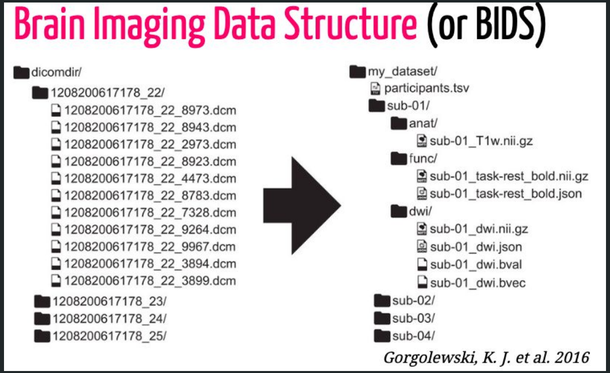
BBL
Brain Behavior Lab

To do:

1. Introduction to `fw-heudiconv`
2. How does it work?
3. Hands-on walkthrough
4. Workshop (optional)

Introduction to `fw-heudiconv`

1. Flywheel & BIDS
2. `heudiconv + Flywheel =` **`fw-heudiconv`**


Brain Imaging Data Structure (or BIDS)
dicomdir/
1208200617178_22/
1208200617178_22_8973.dcm
1208200617178_22_8943.dcm
1208200617178_22_2973.dcm
1208200617178_22_8923.dcm
1208200617178_22_4473.dcm
1208200617178_22_8783.dcm
1208200617178_22_7328.dcm
1208200617178_22_9264.dcm
1208200617178_22_9967.dcm
1208200617178_22_3894.dcm
1208200617178_22_3899.dcm
1208200617178_23/
1208200617178_24/
1208200617178_25/
my_dataset/
participants.tsv
sub-01/
anat/
sub-01_T1w.nii.gz
func/
sub-01_task-rest_bold.nii.gz
sub-01_task-rest_bold.json
dwi/
sub-01_dwi.nii.gz
sub-01_dwi.json
sub-01_dwi.bval
sub-01_dwi.bvec
sub-02/
sub-03/
sub-04/
Gorgolewski, K. J. et al. 2016

Flywheel & BIDS

- **Flywheel**: *Next generation informatics platform for biomedical research and collaboration*
- Eliminates challenges of traditional neuroscience research:
  - Boring
  - Difficult
  - Error-prone
  - Not directly related to neuroscience (e.g. submitting compute jobs, compiling software)
- **BIDS**: *A simple and intuitive way to organize and describe your neuroimaging and behavioral data*
- Enhances traditional neuroscience research:
  - Easy collaboration between labs and databases
  - Easy consumption by conversion & analysis software applications

Flywheel & BIDS: “Traditional” Research Model

1.	Identify researcher	Trainees have different skill levels
2.	Download scans from scanner	Conversion software varies, and choices matter
3.	Organize your data	Directory structures vary across labs
4.	Choose analysis software	Multiple dependencies of software versions
5.	Write custom processing pipeline code	Code poorly documented, no version control.
6.	Conduct quality control	Quality control often idiosyncratic
7.	Pass off analyses to next stakeholder	Documentation usually poor, does not keep pace with versions.

Flywheel & BIDS: Reproducible Neuroscience Model

1. Data pulled straight from scanner (*Flywheel reaper*)
2. Data curation with BIDS that describes data and acquisition parameters (**`fw-heudiconv`**)
3. High-performing, benchmarked pre-processing pipelines that auto-configure to data acquisition (*BIDS-apps**)
4. Automated, reproducible quality control
5. Guaranteed reproducibility and "glass box" code with wrapped dependencies (*Docker containers*)
6. Ease of sharing across labs and projects (*Flywheel gears*)
7. Unlimited scalability (*Google cloud platform*)

* Demos in the next meeting

So what does data curated in BIDS look like...?

Flywheel & BIDS: BIDS-ified Data

Key features:

- _{key}-{value}_ pairs separated by underscores
- Entity hierarchy: Project > Subject > Session > Acquisition > File
- JSON sidecars for parseable metadata

Using BIDS, the data describes itself!

sub-{value}, **ses-**{value}, **acq-**{value}, **ce-**{value}, **rec-**{value}, **dir-**{value}, **run-**{index}, **mod-**{value}...

`fw-heudiconv` helps us accomplish this on Flywheel!

```
.bidsignore
dataset_description.json
sub-106246/
  ses-1/
      anat/
          sub-106246_ses-1_T1w.json
          sub-106246_ses-1_T1w.nii.gz
      fmap/
          sub-106246_ses-1_magnitude1.json
          sub-106246_ses-1_magnitude1.nii.gz
          sub-106246_ses-1_magnitude2.json
          sub-106246_ses-1_magnitude2.nii.gz
      func/
          sub-106246_ses-1_task-frac2back.json
          sub-106246_ses-1_task-frac2back.nii.gz
          sub-106246_ses-1_task-idemo.json
          sub-106246_ses-1_task-idemo.nii.gz
```

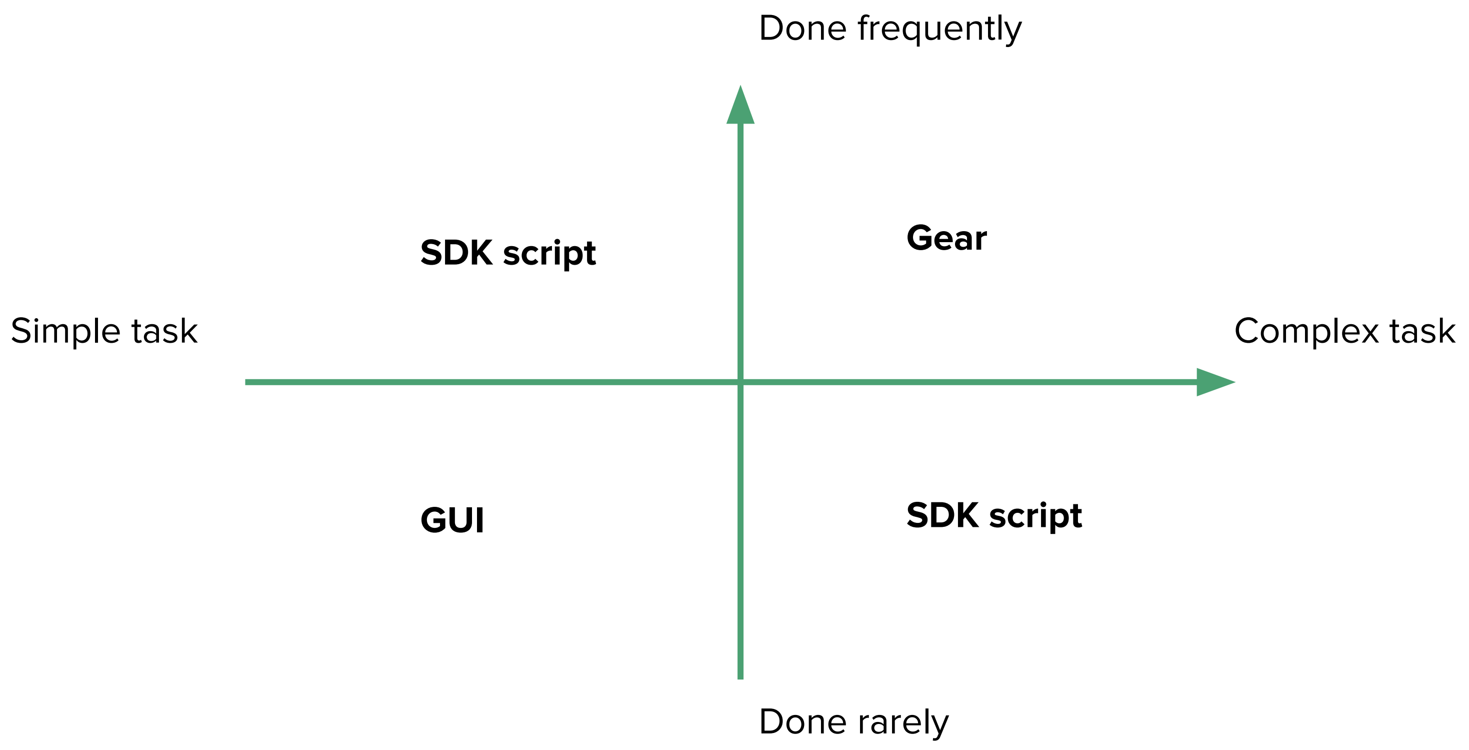
How does it work?

1. Flywheel SDK and Gears
2. Writing heuristics:
   a. `create_key()`
   b. `infotodict()`
   c. `MetadataExtras`
   d. `IntendedFor`
   e. `ReplaceSubject()` & `ReplaceSession()`

SDK & Gears

- **Software Development Kit**: *a set of tools, libraries, relevant documentation, code samples, processes, and guides that allow developers to create software applications on some platform*
    - Available in Python, Matlab, R
- **Gears**: *ready-to-use pre-processing or analysis pipeline applications; "an algorithm that has been packaged in a way that allows it to be run and managed within the Flywheel Platform"*
    - Containerised (through Docker), so almost anything is possible!
    - Metadata is preserved
    - Code is visible
    - Version controlled

SDK & Gears


Done frequently
SDK script
Gear
Simple task
Complex task
GUI
SDK script
Done rarely

So when are we going to see some code already?

HeuDiConv + FLYWHEEL

"a flexible DICOM converter for organizing brain imaging data into structured BIDS directory layouts"

=

BBL
Brain Behavior Lab

fw-heudiconv

Heuristics with `fw-heudiconv`

- **`fw-heudiconv`** is written in Python, and builds upon **`heudiconv`** principles (i.e. heuristics)
- **Heuristic**: *a discrete set of rules that differentiates imaging files by their DICOM header information*
- Like the original **`heudiconv`**, **`fw-heudiconv`** looks through your scans on Flywheel and renames them based on the ruleset you specify using boolean logic in your heuristic file.

Let's look at some Python functions that make up a heuristic file...

Basic Functions: `create_key()`

This function defines template keys for the different scan types and the naming convention they will take. **`fw-heudiconv`** uses string formatting to insert the values.

```
t1w = create_key('sub-{subject}/{session}/anat/sub-{subject}_{session}_T1w')
```

You can create as many keys as you want for your scans (the more strict/granular the better)

```
nback_HiConHiLoWMgated_run1 = create_key(
    'sub-{subject}/{session}/func/sub-{subject}_{session}_'
    'task-nback_acq-HiConHiLoWMgated_run-01_bold')
```

Basic Functions: **`infotodict()`**

This function loops over the scans and their metadata, and assigns each scan a template key in a variable. Using a python dictionary like this:

```
info = {t1w:[], nback_HiConHiLoWMgated_run1:[]}
```

`fw-heudiconv` will add a scan to the correct list if it meets some logical criteria:

```
if "anat_t1w" in protocol:

    info[t1w].append(s.series_id)

elif "HiConHiLoWMgated" in s.protocol_name and "M" in s.image_type:

    info[nback_HiConHiLoWMgated_run1].append(s.series_id)
```

Extensions: **`MetadataExtras()`**

`fw-heudiconv` functionality is easily extensible for Flywheel specific scenarios.

`MetadataExtras()` is a variable that allows users to hard code metadata to key templates:

```
MetadataExtras = {

    nback_HiConHiLoWMgated_run1: { "TaskName": "n-back" }

}
```

These metadata show up in the JSON sidecar file on export, and can overwrite pre-existing Flywheel data or create new fields

Extensions: `IntendedFor()`

Using fieldmaps? BIDS-enabled applications such as ***fMRIPrep*** need to be told what scans these fieldmaps are correcting for. This is easy to do with is a variable that specifies these in **`fw-heudiconv.`**

```
IntendedFor = {

    fmap: [

        '{session}/func/sub-{subject}_{session}_task-nback_acq-HiConHiLoWMgated_run-01_bold'

    ]

}
```

`fw-heudiconv` will check to make sure that each of these files exists post-hoc

Extensions: **`Replace*()`**

What if you have subject or session labels that you want to correct in BIDS? **`fw-heudiconv`** has a solution for this too. **`fw-heudiconv`** has dedicated functionality for parsing and replacing subject labels or session labels, so you can write a custom function in the heuristic that tells **`fw-heudiconv`** how to handle each case:

```
def ReplaceSubject(subj_label):

    return str(int(subj_label))

def ReplaceSession(sess_label):

    return str(int(sess_label + 1)
```

Walkthrough

1. Querying a project
2. Curating your data
3. Exporting your data

Walkthrough: Querying

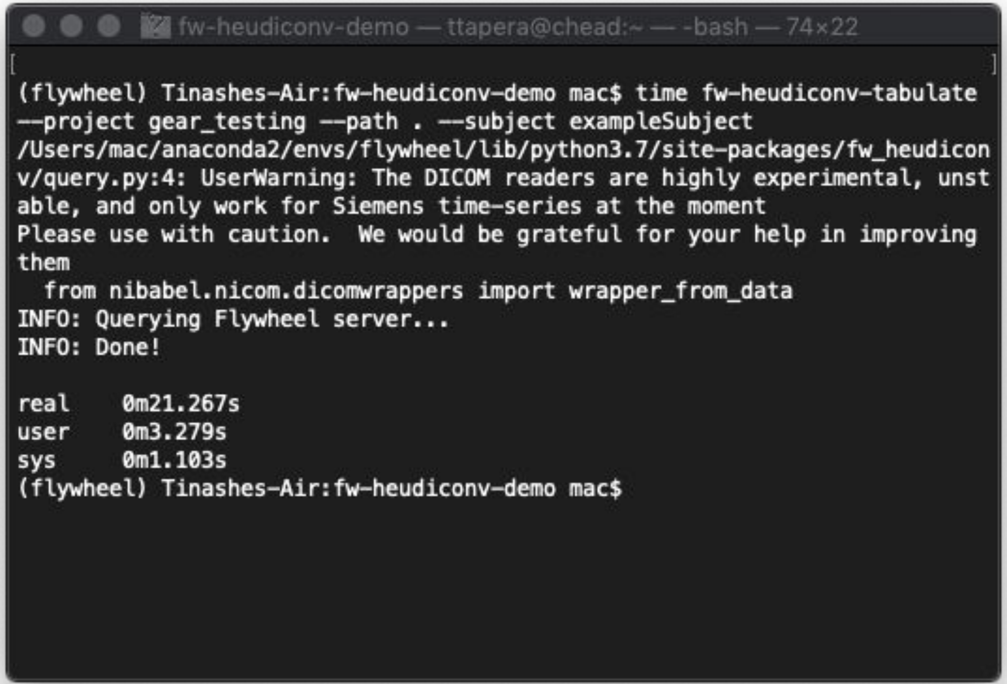
First step to curating your data: *understanding your data!*

Use the tabulation tool to get a tabular view of the unique dicom sequences in your dataset:

```
$ fw-heudiconv-tabulate --project gear_testing --path <where to download the table>
--subject <optional list of subjects> --session <optional list of sessions> --dry_run <to
print and not download>
```

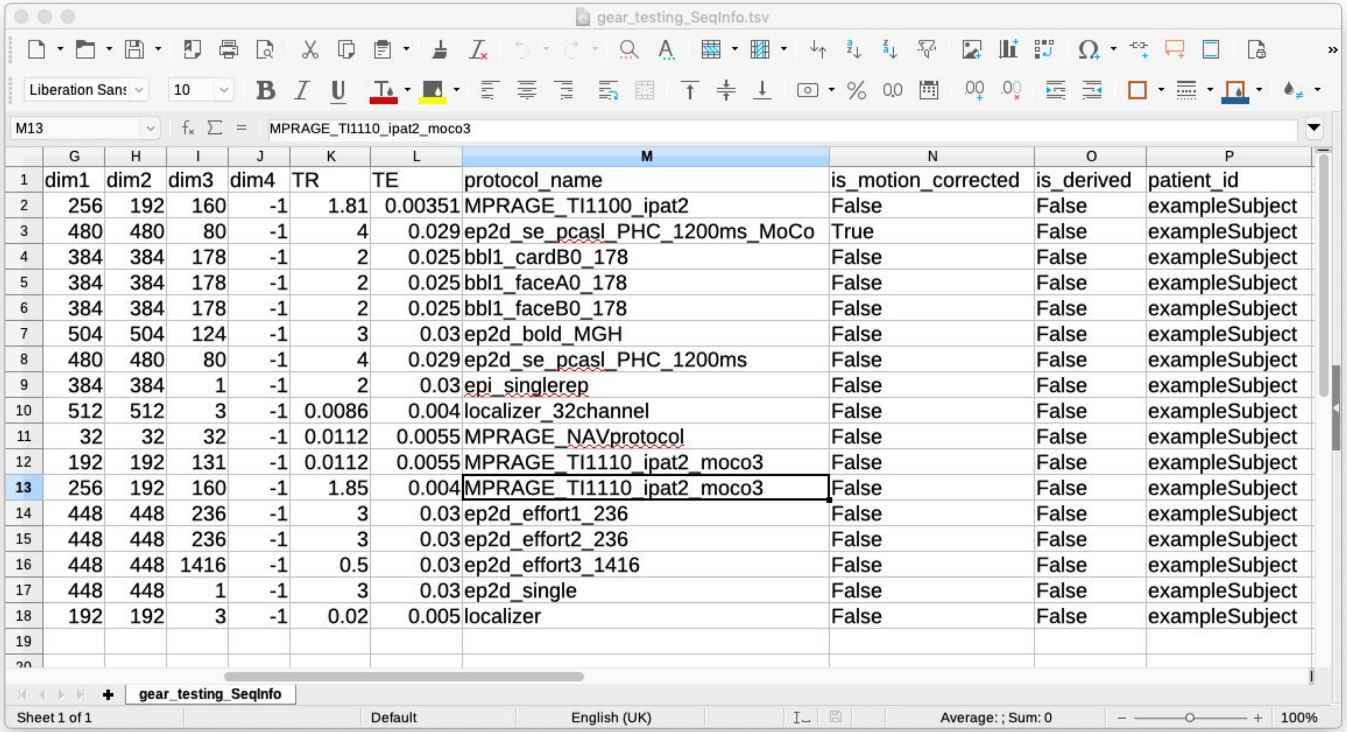
Unique fields include protocol names, sequence names, TR, TE, motion correction, and derived images.

Walkthrough: Querying



```
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ time fw-heudiconv-tabulate
--project gear_testing --path . --subject exampleSubject
/Users/mac/anaconda2/envs/flywheel/lib/python3.7/site-packages/fw_heudicon
v/query.py:4: UserWarning: The DICOM readers are highly experimental, unst
able, and only work for Siemens time-series at the moment
Please use with caution.  We would be grateful for your help in improving
them
  from nibabel.nicom.dicomwrappers import wrapper_from_data
INFO: Querying Flywheel server...
INFO: Done!

real    0m21.267s
user    0m3.279s
sys     0m1.103s
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$
```

Walkthrough: Querying



gear_testing_SeqInfo.tsv

Liberation Sans 10

M13 MPRAGE_TI1110_ipat2_moco3

	G	H	I	J	K	L	M	N	O	P
1	dim1	dim2	dim3	dim4	TR	TE	protocol_name	is_motion_corrected	is_derived	patient_id
2	256	192	160	-1	1.81	0.00351	MPRAGE_TI1100_ipat2	False	False	exampleSubject
3	480	480	80	-1	4	0.029	ep2d_se_pcasl_PHC_1200ms_MoCo	True	False	exampleSubject
4	384	384	178	-1	2	0.025	bbl1_cardB0_178	False	False	exampleSubject
5	384	384	178	-1	2	0.025	bbl1_faceA0_178	False	False	exampleSubject
6	384	384	178	-1	2	0.025	bbl1_faceB0_178	False	False	exampleSubject
7	504	504	124	-1	3	0.03	ep2d_bold_MGH	False	False	exampleSubject
8	480	480	80	-1	4	0.029	ep2d_se_pcasl_PHC_1200ms	False	False	exampleSubject
9	384	384	1	-1	2	0.03	epi_singlerep	False	False	exampleSubject
10	512	512	3	-1	0.0086	0.004	localizer_32channel	False	False	exampleSubject
11	32	32	32	-1	0.0112	0.0055	MPRAGE_NAVprotocol	False	False	exampleSubject
12	192	192	131	-1	0.0112	0.0055	MPRAGE_TI1110_ipat2_moco3	False	False	exampleSubject
13	256	192	160	-1	1.85	0.004	MPRAGE_TI1110_ipat2_moco3	False	False	exampleSubject
14	448	448	236	-1	3	0.03	ep2d_effort1_236	False	False	exampleSubject
15	448	448	236	-1	3	0.03	ep2d_effort2_236	False	False	exampleSubject
16	448	448	1416	-1	0.5	0.03	ep2d_effort3_1416	False	False	exampleSubject
17	448	448	1	-1	3	0.03	ep2d_single	False	False	exampleSubject
18	192	192	3	-1	0.02	0.005	localizer	False	False	exampleSubject
19										

gear_testing_SeqInfo

Sheet 1 of 1 Default English (UK) Average: ; Sum: 0 100%

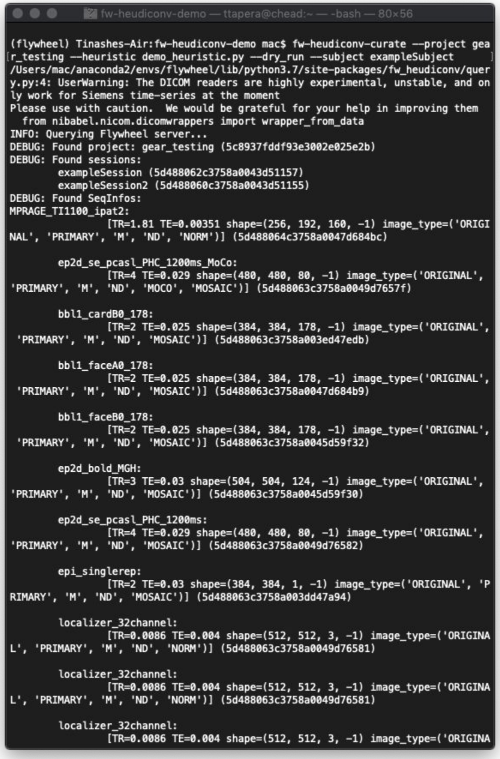
Walkthrough: Curating

Next we begin curating the data into BIDS.

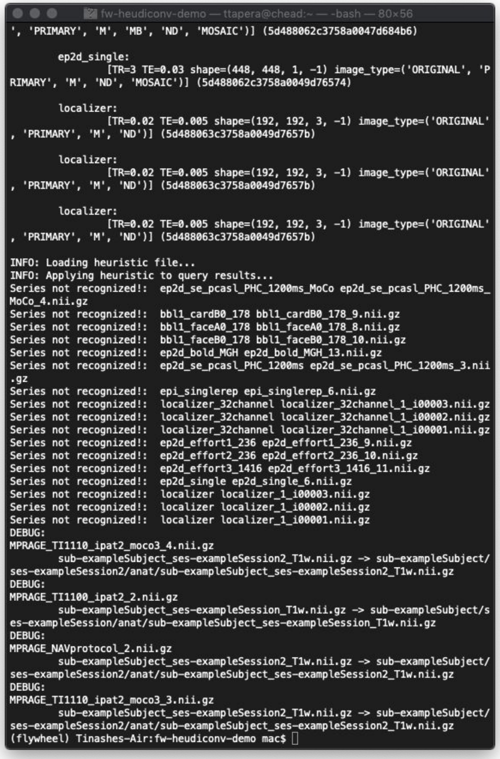
1. Add the basic heuristic functions to a new python file.
2. Use the curation tool in *dry_run* mode to test out and see what changes would be applied if you used the heuristic:

```
$ fw-heudiconv-curate --project gear_testing --heuristic <path to heuristic file> --subject <optional list of subjects> --session <optional list of sessions> --dry_run <to print and not apply changes>
```

Walkthrough: Curating



```
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ fw-heudiconv-curate --project gea
r_testing --heuristic demo_heuristic.py --dry_run --subject exampleSubject
/Users/mac/anaconda2/envs/flywheel/lib/python3.7/site-packages/fw_heudiconv/quer
y.py:4: UserWarning: The DICOM readers are highly experimental, unstable, and on
ly work for Siemens time-series at the moment
Please use with caution.  We would be grateful for your help in improving them
  from nibabel.nicom.dicomwrappers import wrapper_from_data
INFO: Querying Flywheel server...
DEBUG: Found project: gear_testing (5c8937fddf93e3002e025e2b)
DEBUG: Found sessions:
        exampleSession (5d488062c3758a0043d51157)
        exampleSession2 (5d488060c3758a0043d51155)
DEBUG: Found SeqInfos:
MPRAGE_TI1100_ipat2:
                [TR=1.81 TE=0.00351 shape=(256, 192, 160, -1) image_type=('ORIGI
NAL', 'PRIMARY', 'M', 'ND', 'NORM')] (5d488064c3758a0047d684bc)

        ep2d_se_pcasl_PHC_1200ms_MoCo:
                [TR=4 TE=0.029 shape=(480, 480, 80, -1) image_type=('ORIGINAL',
'PRIMARY', 'M', 'ND', 'MOCO', 'MOSAIC')] (5d488063c3758a0049d7657f)

        bbl1_cardB0_178:
                [TR=2 TE=0.025 shape=(384, 384, 178, -1) image_type=('ORIGINAL',
 'PRIMARY', 'M', 'ND', 'MOSAIC')] (5d488063c3758a003ed47edb)

        bbl1_faceA0_178:
                [TR=2 TE=0.025 shape=(384, 384, 178, -1) image_type=('ORIGINAL',
 'PRIMARY', 'M', 'ND', 'MOSAIC')] (5d488063c3758a0047d684b9)

        bbl1_faceB0_178:
                [TR=2 TE=0.025 shape=(384, 384, 178, -1) image_type=('ORIGINAL',
 'PRIMARY', 'M', 'ND', 'MOSAIC')] (5d488063c3758a0045d59f32)

        ep2d_bold_MGH:
                [TR=3 TE=0.03 shape=(504, 504, 124, -1) image_type=('ORIGINAL',
'PRIMARY', 'M', 'ND', 'MOSAIC')] (5d488063c3758a0045d59f30)

        ep2d_se_pcasl_PHC_1200ms:
                [TR=4 TE=0.029 shape=(480, 480, 80, -1) image_type=('ORIGINAL',
'PRIMARY', 'M', 'ND', 'MOSAIC')] (5d488063c3758a0049d76582)

        epi_singlerep:
                [TR=2 TE=0.03 shape=(384, 384, 1, -1) image_type=('ORIGINAL', 'P
RIMARY', 'M', 'ND', 'MOSAIC')] (5d488063c3758a003dd47a94)

        localizer_32channel:
                [TR=0.0086 TE=0.004 shape=(512, 512, 3, -1) image_type=('ORIGINA
L', 'PRIMARY', 'M', 'ND', 'NORM')] (5d488063c3758a0049d76581)

        localizer_32channel:
                [TR=0.0086 TE=0.004 shape=(512, 512, 3, -1) image_type=('ORIGINA
L', 'PRIMARY', 'M', 'ND', 'NORM')] (5d488063c3758a0049d76581)

        localizer_32channel:
                [TR=0.0086 TE=0.004 shape=(512, 512, 3, -1) image_type=('ORIGINA
```

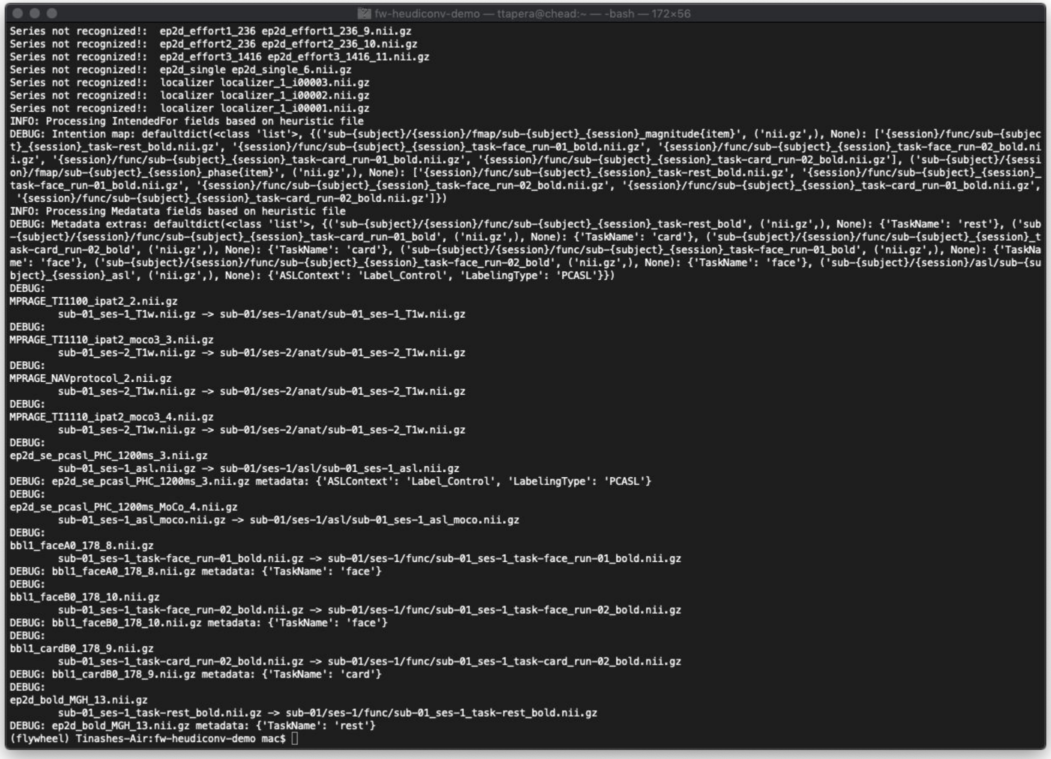


```
', 'PRIMARY', 'M', 'MB', 'ND', 'MOSAIC')] (5d488062c3758a0047d684b6)

        ep2d_single:
                [TR=3 TE=0.03 shape=(448, 448, 1, -1) image_type=('ORIGINAL', 'P
RIMARY', 'M', 'ND', 'MOSAIC')] (5d488062c3758a0049d76574)

        localizer:
                [TR=0.02 TE=0.005 shape=(192, 192, 3, -1) image_type=('ORIGINAL'
, 'PRIMARY', 'M', 'ND')] (5d488063c3758a0049d7657b)

        localizer:
                [TR=0.02 TE=0.005 shape=(192, 192, 3, -1) image_type=('ORIGINAL'
, 'PRIMARY', 'M', 'ND')] (5d488063c3758a0049d7657b)

        localizer:
                [TR=0.02 TE=0.005 shape=(192, 192, 3, -1) image_type=('ORIGINAL'
, 'PRIMARY', 'M', 'ND')] (5d488063c3758a0049d7657b)

INFO: Loading heuristic file...
INFO: Applying heuristic to query results...
Series not recognized!:  ep2d_se_pcasl_PHC_1200ms_MoCo ep2d_se_pcasl_PHC_1200ms_
MoCo_4.nii.gz
Series not recognized!:  bbl1_cardB0_178 bbl1_cardB0_178_9.nii.gz
Series not recognized!:  bbl1_faceA0_178 bbl1_faceA0_178_8.nii.gz
Series not recognized!:  bbl1_faceB0_178 bbl1_faceB0_178_10.nii.gz
Series not recognized!:  ep2d_bold_MGH ep2d_bold_MGH_13.nii.gz
Series not recognized!:  ep2d_se_pcasl_PHC_1200ms ep2d_se_pcasl_PHC_1200ms_3.nii
.gz
Series not recognized!:  epi_singlerep epi_singlerep_6.nii.gz
Series not recognized!:  localizer_32channel localizer_32channel_1_i00003.nii.gz
Series not recognized!:  localizer_32channel localizer_32channel_1_i00002.nii.gz
Series not recognized!:  localizer_32channel localizer_32channel_1_i00001.nii.gz
Series not recognized!:  ep2d_effort1_236 ep2d_effort1_236_9.nii.gz
Series not recognized!:  ep2d_effort2_236 ep2d_effort2_236_10.nii.gz
Series not recognized!:  ep2d_effort3_1416 ep2d_effort3_1416_11.nii.gz
Series not recognized!:  ep2d_single ep2d_single_6.nii.gz
Series not recognized!:  localizer localizer_1_i00003.nii.gz
Series not recognized!:  localizer localizer_1_i00002.nii.gz
Series not recognized!:  localizer localizer_1_i00001.nii.gz
DEBUG:
MPRAGE_TI1110_ipat2_moco3_4.nii.gz
        sub-exampleSubject_ses-exampleSession2_T1w.nii.gz -> sub-exampleSubject/
ses-exampleSession2/anat/sub-exampleSubject_ses-exampleSession2_T1w.nii.gz
DEBUG:
MPRAGE_TI1100_ipat2_2.nii.gz
        sub-exampleSubject_ses-exampleSession_T1w.nii.gz -> sub-exampleSubject/s
es-exampleSession/anat/sub-exampleSubject_ses-exampleSession_T1w.nii.gz
DEBUG:
MPRAGE_NAVprotocol_2.nii.gz
        sub-exampleSubject_ses-exampleSession2_T1w.nii.gz -> sub-exampleSubject/
ses-exampleSession2/anat/sub-exampleSubject_ses-exampleSession2_T1w.nii.gz
DEBUG:
MPRAGE_TI1110_ipat2_moco3_3.nii.gz
        sub-exampleSubject_ses-exampleSession2_T1w.nii.gz -> sub-exampleSubject/
ses-exampleSession2/anat/sub-exampleSubject_ses-exampleSession2_T1w.nii.gz
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ 
```

Walkthrough: Curating

3. Add more keys to capture the sequences not recognised by your logic
4. Rinse and repeat
5. Add hardcoded metadata, fieldmap intentions, and subject/session label replacements as needed

Walkthrough: Curating



fw-heudiconv-demo — ttapera@chead:~ — -bash — 172×56

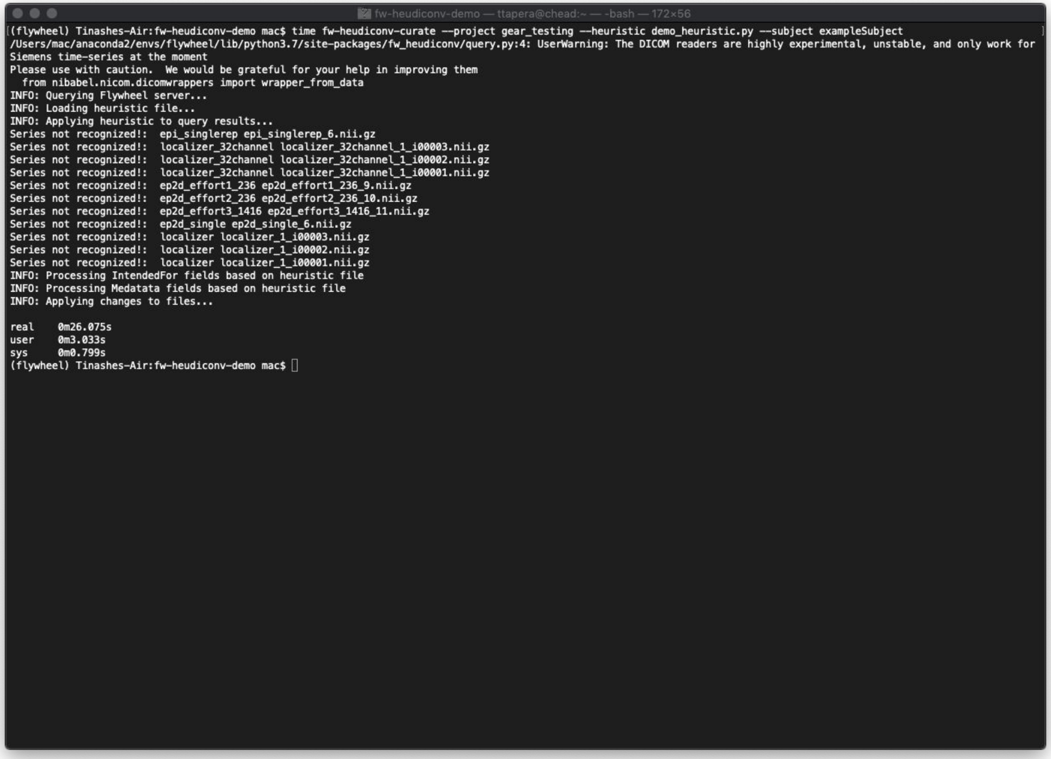
```
Series not recognized!:  ep2d_effort1_236 ep2d_effort1_236_9.nii.gz
Series not recognized!:  ep2d_effort2_236 ep2d_effort2_236_10.nii.gz
Series not recognized!:  ep2d_effort3_1416 ep2d_effort3_1416_11.nii.gz
Series not recognized!:  ep2d_single ep2d_single_6.nii.gz
Series not recognized!:  localizer localizer_1_i00003.nii.gz
Series not recognized!:  localizer localizer_1_i00002.nii.gz
Series not recognized!:  localizer localizer_1_i00001.nii.gz
INFO: Processing IntendedFor fields based on heuristic file
DEBUG: Intention map: defaultdict(<class 'list'>, {('sub-{subject}/{session}/fmap/sub-{subject}_{session}_magnitude{item}', ('nii.gz',), None): ['{session}/func/sub-{subjec
t}_{session}_task-rest_bold.nii.gz', '{session}/func/sub-{subject}_{session}_task-face_run-01_bold.nii.gz', '{session}/func/sub-{subject}_{session}_task-face_run-02_bold.ni
i.gz', '{session}/func/sub-{subject}_{session}_task-card_run-01_bold.nii.gz', '{session}/func/sub-{subject}_{session}_task-card_run-02_bold.nii.gz'], ('sub-{subject}/{sessi
on}/fmap/sub-{subject}_{session}_phase{item}', ('nii.gz',), None): ['{session}/func/sub-{subject}_{session}_task-rest_bold.nii.gz', '{session}/func/sub-{subject}_{session}_
task-face_run-01_bold.nii.gz', '{session}/func/sub-{subject}_{session}_task-face_run-02_bold.nii.gz', '{session}/func/sub-{subject}_{session}_task-card_run-01_bold.nii.gz',
 '{session}/func/sub-{subject}_{session}_task-card_run-02_bold.nii.gz']})
INFO: Processing Medatata fields based on heuristic file
DEBUG: Metadata extras: defaultdict(<class 'list'>, {('sub-{subject}/{session}/func/sub-{subject}_{session}_task-rest_bold', ('nii.gz',), None): {'TaskName': 'rest'}, ('sub
-{subject}/{session}/func/sub-{subject}_{session}_task-card_run-01_bold', ('nii.gz',), None): {'TaskName': 'card'}, ('sub-{subject}/{session}/func/sub-{subject}_{session}_t
ask-card_run-02_bold', ('nii.gz',), None): {'TaskName': 'card'}, ('sub-{subject}/{session}/func/sub-{subject}_{session}_task-face_run-01_bold', ('nii.gz',), None): {'TaskNa
me': 'face'}, ('sub-{subject}/{session}/func/sub-{subject}_{session}_task-face_run-02_bold', ('nii.gz',), None): {'TaskName': 'face'}, ('sub-{subject}/{session}/asl/sub-{su
bject}_{session}_asl', ('nii.gz',), None): {'ASLContext': 'Label_Control', 'LabelingType': 'PCASL'}})
DEBUG:
MPRAGE_TI1100_ipat2_2.nii.gz
        sub-01_ses-1_T1w.nii.gz -> sub-01/ses-1/anat/sub-01_ses-1_T1w.nii.gz
DEBUG:
MPRAGE_TI1110_ipat2_moco3_3.nii.gz
        sub-01_ses-2_T1w.nii.gz -> sub-01/ses-2/anat/sub-01_ses-2_T1w.nii.gz
DEBUG:
MPRAGE_NAVprotocol_2.nii.gz
        sub-01_ses-2_T1w.nii.gz -> sub-01/ses-2/anat/sub-01_ses-2_T1w.nii.gz
DEBUG:
MPRAGE_TI1110_ipat2_moco3_4.nii.gz
        sub-01_ses-2_T1w.nii.gz -> sub-01/ses-2/anat/sub-01_ses-2_T1w.nii.gz
DEBUG:
ep2d_se_pcasl_PHC_1200ms_3.nii.gz
        sub-01_ses-1_asl.nii.gz -> sub-01/ses-1/asl/sub-01_ses-1_asl.nii.gz
DEBUG: ep2d_se_pcasl_PHC_1200ms_3.nii.gz metadata: {'ASLContext': 'Label_Control', 'LabelingType': 'PCASL'}
DEBUG:
ep2d_se_pcasl_PHC_1200ms_MoCo_4.nii.gz
        sub-01_ses-1_asl_moco.nii.gz -> sub-01/ses-1/asl/sub-01_ses-1_asl_moco.nii.gz
DEBUG:
bbl1_faceA0_178_8.nii.gz
        sub-01_ses-1_task-face_run-01_bold.nii.gz -> sub-01/ses-1/func/sub-01_ses-1_task-face_run-01_bold.nii.gz
DEBUG: bbl1_faceA0_178_8.nii.gz metadata: {'TaskName': 'face'}
DEBUG:
bbl1_faceB0_178_10.nii.gz
        sub-01_ses-1_task-face_run-02_bold.nii.gz -> sub-01/ses-1/func/sub-01_ses-1_task-face_run-02_bold.nii.gz
DEBUG: bbl1_faceB0_178_10.nii.gz metadata: {'TaskName': 'face'}
DEBUG:
bbl1_cardB0_178_9.nii.gz
        sub-01_ses-1_task-card_run-02_bold.nii.gz -> sub-01/ses-1/func/sub-01_ses-1_task-card_run-02_bold.nii.gz
DEBUG: bbl1_cardB0_178_9.nii.gz metadata: {'TaskName': 'card'}
DEBUG:
ep2d_bold_MGH_13.nii.gz
        sub-01_ses-1_task-rest_bold.nii.gz -> sub-01/ses-1/func/sub-01_ses-1_task-rest_bold.nii.gz
DEBUG: ep2d_bold_MGH_13.nii.gz metadata: {'TaskName': 'rest'}
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ 
```

Walkthrough: Curating

6. Finally, when satisfied, remove the *dry_run* flag, and apply your changes!

Don’t be concerned if you mess up; **`fw-heudiconv`** only changes BIDS metadata and does not manipulate the underlying containers. Simply amend your heuristic and re-curate.

Walkthrough: Curating



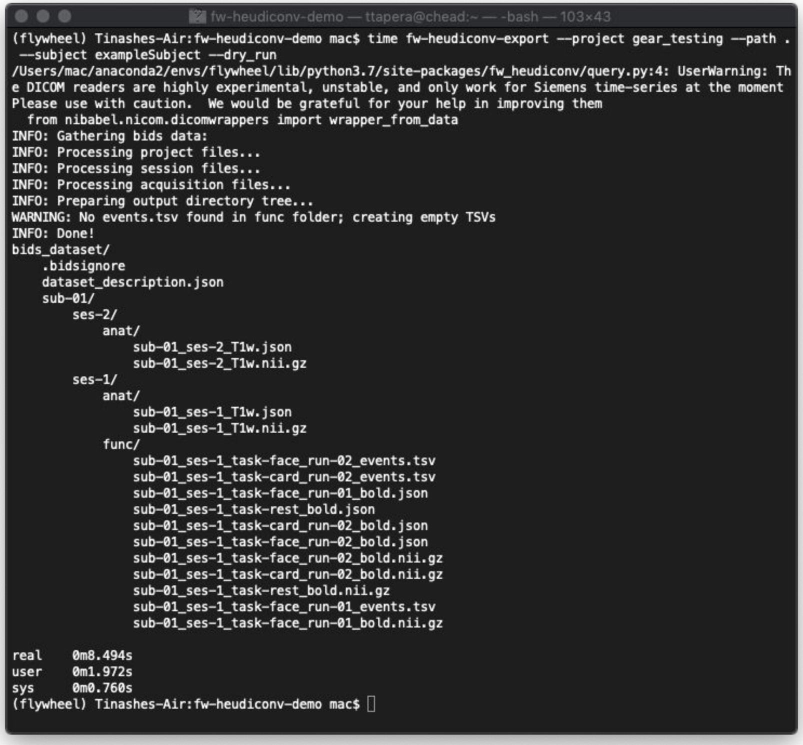
```
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ time fw-heudiconv-curate --project gear_testing --heuristic demo_heuristic.py --subject exampleSubject
/Users/mac/anaconda2/envs/flywheel/lib/python3.7/site-packages/fw_heudiconv/query.py:4: UserWarning: The DICOM readers are highly experimental, unstable, and only work for
Siemens time-series at the moment
Please use with caution.  We would be grateful for your help in improving them
  from nibabel.nicom.dicomwrappers import wrapper_from_data
INFO: Querying Flywheel server...
INFO: Loading heuristic file...
INFO: Applying heuristic to query results...
Series not recognized!:  epi_singlerep epi_singlerep_6.nii.gz
Series not recognized!:  localizer_32channel localizer_32channel_1_i00003.nii.gz
Series not recognized!:  localizer_32channel localizer_32channel_1_i00002.nii.gz
Series not recognized!:  localizer_32channel localizer_32channel_1_i00001.nii.gz
Series not recognized!:  ep2d_effort1_236 ep2d_effort1_236_9.nii.gz
Series not recognized!:  ep2d_effort2_236 ep2d_effort2_236_10.nii.gz
Series not recognized!:  ep2d_effort3_1416 ep2d_effort3_1416_11.nii.gz
Series not recognized!:  ep2d_single ep2d_single_6.nii.gz
Series not recognized!:  localizer localizer_1_i00003.nii.gz
Series not recognized!:  localizer localizer_1_i00002.nii.gz
Series not recognized!:  localizer localizer_1_i00001.nii.gz
INFO: Processing IntendedFor fields based on heuristic file
INFO: Processing Medatata fields based on heuristic file
INFO: Applying changes to files...

real    0m26.075s
user    0m3.033s
sys     0m0.799s
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ 
```

Walkthrough: Exporting

Optionally, use the export tool to view what your BIDS data looks like in a tree structure

```
$ fw-heudiconv-export --project gear_testing --path <path to where you want to download>
--subject <optional list of subjects> --session <optional list of sessions> --dry_run <to
print only a directory tree and not download>
```

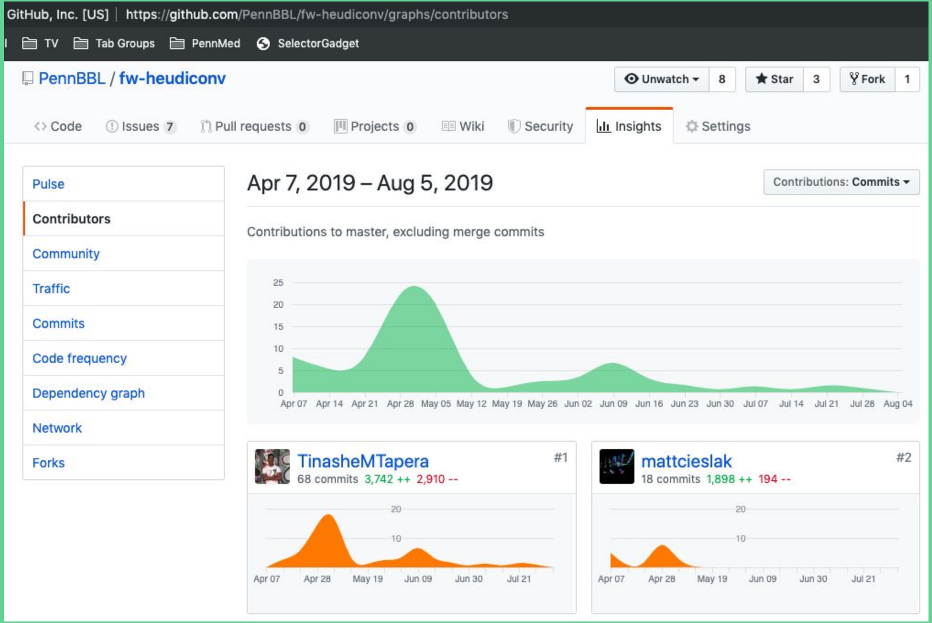
Walkthrough: Exporting



```
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ time fw-heudiconv-export --project gear_testing --path .
 --subject exampleSubject --dry_run
/Users/mac/anaconda2/envs/flywheel/lib/python3.7/site-packages/fw_heudiconv/query.py:4: UserWarning: Th
e DICOM readers are highly experimental, unstable, and only work for Siemens time-series at the moment
Please use with caution.  We would be grateful for your help in improving them
  from nibabel.nicom.dicomwrappers import wrapper_from_data
INFO: Gathering bids data:
INFO: Processing project files...
INFO: Processing session files...
INFO: Processing acquisition files...
INFO: Preparing output directory tree...
WARNING: No events.tsv found in func folder; creating empty TSVs
INFO: Done!
bids_dataset/
    .bidsignore
    dataset_description.json
    sub-01/
        ses-2/
            anat/
                sub-01_ses-2_T1w.json
                sub-01_ses-2_T1w.nii.gz
        ses-1/
            anat/
                sub-01_ses-1_T1w.json
                sub-01_ses-1_T1w.nii.gz
            func/
                sub-01_ses-1_task-face_run-02_events.tsv
                sub-01_ses-1_task-card_run-02_events.tsv
                sub-01_ses-1_task-face_run-01_bold.json
                sub-01_ses-1_task-rest_bold.json
                sub-01_ses-1_task-card_run-02_bold.json
                sub-01_ses-1_task-face_run-02_bold.json
                sub-01_ses-1_task-face_run-02_bold.nii.gz
                sub-01_ses-1_task-card_run-02_bold.nii.gz
                sub-01_ses-1_task-rest_bold.nii.gz
                sub-01_ses-1_task-face_run-01_events.tsv
                sub-01_ses-1_task-face_run-01_bold.nii.gz

real    0m8.494s
user    0m1.972s
sys     0m0.760s
(flywheel) Tinashes-Air:fw-heudiconv-demo mac$ 
```

Overview

1. Flywheel + BIDS = Awesome reproducible neuroscience!
2. Use **`fw-heudiconv`** to get your data BIDS ready
3. Community engagement is needed; submit **issues**, **bugs**, and **questions** to **Github**; **pull requests** welcome!
4. Show off your heuristics tips and tricks in our Repo too!



Apr 7, 2019 – Aug 5, 2019

Contributions to master, excluding merge commits

TinasheMTapera #1 — 68 commits 3,742 ++ 2,910 --

mattcieslak #2 — 18 commits 1,898 ++ 194 --

github.com/PennBBL/fw-heudiconv/

Interactive workshop

Have a project on Flywheel you want to curate?
Stick around and we'll help you get started!

Thanks to...

Ted Satterthwaite
Matt Cieslak
Azeez Adebimpe
Anna Xu
Diego Davila
Brain Behavior Lab
Desmond Oathes and Lab members
Matt Flounders
Joe Deluisi
UPenn Flywheel users
Flywheel support team

BBL
Brain Behavior Lab