

dmrr_prep

July 13, 2018

1 Preparing metadata for submission to the DMRR

```
In [1]: study_names = ['Healthy Controls']
        study_id = 'EXR-MTEWA1HealthyControls'
        directory = 'healthyCtrl/'
        is_time_series = False
        '''
        study_names = ['Feeding Study-30 min', 'Feeding Study-1 hr']
        study_id = 'EXR-MTEWA1HealthyControls'
        directory = 'feedingStudy/'
        is_time_series = True
        '''
```

```
Out[1]: "\nstudy_names = ['Feeding Study-30 min', 'Feeding Study-1 hr']\nstudy_id = 'EXR-MTEWA1HealthyControls'\n"
```

1.1 Load the samples spreadsheet

```
In [2]: import pandas as pd
        import numpy as np
        samples_df = pd.read_csv('sample_sheet.csv')
        for column in ['Gender', 'Race', 'Source']: # strip whitespace for consistency
            samples_df[column] = samples_df[column].str.strip()
            if column == 'Gender': # capitalize first letter
                samples_df[column] = samples_df[column].str.capitalize()
        # only use healthy control study and samples that passed quality control
        samples_df = samples_df.loc[(samples_df['Study'].isin(study_names)) & (samples_df['MIS'] == 'N')]
        # keep only the necessary columns
        samples_df = samples_df[['Participant.ID', 'Sample.ID', 'MT.Unique.ID', 'Age', 'Gender', 'Race', 'Source']]
        samples_df = samples_df.set_index(['MT.Unique.ID']).sort_values(by='Participant.ID')
        if is_time_series:
            samples_df['timepoint'] = pd.Series([sample_id.split('-')[1] for sample_id in samples_df['Sample.ID']]
            samples_df['timestep'] = pd.Series([study.split('-')[1] for study in samples_df['Study']]
        samples_df[:5]
```

```
Out[2]:
```

	Participant.ID	Sample.ID	Age	Gender	Race	Source	\
MT.Unique.ID							
1	70014	70014	23	Male	Asian	Plasma	

107	70014	70014	23	Male	Asian	Serum
2	70016	70016	39	Male	White	Plasma
108	70016	70016	39	Male	White	Serum
3	70028	70028	33	Female	White	Plasma

Study	
MT.Unique.ID	
1	Healthy Controls
107	Healthy Controls
2	Healthy Controls
108	Healthy Controls
3	Healthy Controls

1.2 Load the participant info

```
In [3]: # remove duplicate participants -- all have plasma but some additionally have serum
participants = samples_df.loc[(samples_df['Source'] == 'Plasma')]
if is_time_series:
    participants = participants.loc[(participants['timepoint'] == 'T0')]
participants = participants[['Participant.ID', 'Age', 'Race', 'Gender']].set_index('Participant.ID')
participants[:5]
```

```
Out [3]:
```

Participant.ID	Age	Race	Gender
70014	23	Asian	Male
70016	39	White	Male
70028	33	White	Female
70029	27	Black or African American	Female
70038	22	White	Female

1.3 Use correct ontology terms

```
In [4]: import pprint
race_ontology = {'Asian': 'Asian', 'asian': 'Asian',
                 'Black or African American': 'African American',
                 'mixed/asian & white': 'Multiracial',
                 'mixed/Asian &Black': 'Multiracial',
                 'mixed/black, white, asian': 'Multiracial',
                 'Native Hawaiian or other Pacific Islander': 'Native Hawaiian or Other Pacific Islander',
                 'Pacific Islander': 'Native Hawaiian or Other Pacific Islander',
                 'White': 'White', 'white': 'White',
                 '#MISSING#': '#MISSING#'}
for part_id in participants.index:
    race = participants.at[part_id, 'Race']
    participants.loc[part_id, 'Race'] = race_ontology[race] if race in race_ontology else race

pprint.pprint({'Age': set(participants['Age']), 'Race': set(participants['Race']), 'Gender': set(participants['Gender'])})
```

```
{'Age': {'20',
        '21',
        '22',
        '23',
        '24',
        '25',
        '26',
        '27',
        '28',
        '29',
        '30',
        '31',
        '32',
        '33',
        '34',
        '35',
        '36',
        '37',
        '38',
        '39',
        '40',
        '41'},
'Gender': {'Female', 'Male'},
'Race': {'African American',
        'Asian',
        'Multiracial',
        'Native Hawaiian or Other Pacific Islander',
        'White'}}
```

1.4 Load the donors template

```
In [5]: import pandas as pd
donors = pd.read_csv('templates/Donors.template.tsv', sep='\t')
donors = donors.set_index('#property')
donors.drop(['- Ethnic Group', '-- Current Health Status', '-- Medical History', '-- S
            '-- Family History', '-- Treatment History', '-- Family History', '-- Dev
            '-- Post-mortem Interval', '- Notes', '* Family Members', '*- Family Memb
            '*-- dbName', '*-- URL', '- Health Status', '*-- Notes'], inplace=True)
donors.head()
```

```
Out [5]:
```

	value	domain \
#property		
Donor	NaN	autoID(EXR, uniqAlphaNum, DO)
- Status	NaN	enum(Add, Modify, Hold, Cancel, Suppress, Rele...
- Sex	NaN	biportalTerms((SNOMEDCT,http://purl.bioontolo...
- Racial Category	NaN	biportalTerm(http://data.bioontology.org/sear...
- Donor Type	NaN	enum(Experimental, Control, Healthy Subject)

	default	required	\
#property			
Donor	NaN	NaN	
- Status	Add	True	
- Sex	NaN	True	
- Racial Category	NaN	NaN	
- Donor Type	NaN	True	

	description
#property	
Donor	Document Describing Information About the Dono...
- Status	Status of the document
- Sex	Gender of sample donor (Example: Male, Female,...
- Racial Category	The racial category of the donor
- Donor Type	Sample type (experimental sample or control sa...

1.5 Fill in the donors dataframe

```
In [6]: i = 1
        for part_id in participants.index:
            print(part_id)
            participant_column = 'value' + part_id
            donor_id = study_id + str(i) + '-D0'
            participants.loc[part_id, 'donor.id'] = donor_id
            donors.insert(i, participant_column, donors['value'])
            donors.loc['Donor', participant_column] = donor_id # for matching biosamples to d
            donors.loc['- Status', participant_column] = 'Add'
            donors.loc['- Sex', participant_column] = participants.at[part_id, 'Gender']
            donors.loc['- Racial Category', participant_column] = participants.at[part_id, 'Ra
            donors.loc['- Donor Type', participant_column] = 'Healthy Subject' if not is_time_
            donors.loc['- Age', participant_column] = str(participants.at[part_id, 'Age']) + '
            donors.loc['* Custom Metadata', participant_column] = 1
            donors.loc['*- Property Name', participant_column] = 'Participant.ID'
            donors.loc['*-- Value', participant_column] = part_id
            i += 1
        donors = donors.drop('value', axis=1)
        donors.iloc[:5,:3]
```

70014
70016
70028
70029
70038
70057
70067
70082
70092

70103
70105
70114
70125
70182
70283
70287
70348
70362
70406
70416
70435
70467
70484
70498
70510
70539
70548
70549
70557
70581
70583
70695
70716
70831
70956
70977
71004
71039
71080
71121
71134
71138
71154
71229
71244
71263
71305
71311
71316
71337
71346
71358
71367
71387
71398
71431
71438

71443
71467
71471
71473
71495
71521
71582
71591
71690
71796
71816
71818
71843
71856
71875
71906
71911
71916
71919
71927
71940
71973
71983
72049
72062
72078
72108
72111
72114
72119
72133
72138
72148
72165
72221*
72253*
72254*
72255*
72316
72339*
72349*
72376
72397

```
Out [6]:                                     value70014  \
#property
Donor                                     EXR-MTEWA1HealthyControls1-D0
```

```

- Status                                Add
- Sex                                  Male
- Racial Category                      Asian
- Donor Type                          Healthy Subject

                                value70016 \
#property
Donor                                EXR-MTEWA1HealthyControls2-D0
- Status                                Add
- Sex                                  Male
- Racial Category                      White
- Donor Type                          Healthy Subject

                                value70028
#property
Donor                                EXR-MTEWA1HealthyControls3-D0
- Status                                Add
- Sex                                  Female
- Racial Category                      White
- Donor Type                          Healthy Subject

```

In [7]: `participants.iloc[:5, :5]` *# the participants dataframe now has the donor ids*

```

Out[7]:
      Age  Race  Gender  donor.id
Participant.ID
70014    23   Asian   Male  EXR-MTEWA1HealthyControls1-D0
70016    39   White   Male  EXR-MTEWA1HealthyControls2-D0
70028    33   White  Female  EXR-MTEWA1HealthyControls3-D0
70029    27 African American  Female  EXR-MTEWA1HealthyControls4-D0
70038    22   White  Female  EXR-MTEWA1HealthyControls5-D0

```

1.6 Load the biosamples template

```

In [8]: biosamples = pd.read_csv('templates/Biosamples.template.tsv', sep='\t')
        biosamples = biosamples.set_index('#property')
        biosamples = biosamples.drop(['-- Age at Sampling', '-- Notes', '- Description', '--- :
            '--- Collection Details',
            '---- Sample Collection Method', '---- Geographic Location',
            '---- Collection Date', '---- Time of Collection',
            '---- Collection Tube Type', '----- Other Collection Tube Type',
            '---- Holding Time', '---- Holding Temperature',
            '---- Preservatives Used', '---- Freezing Method',
            '---- Number of Times Freeze Thawed',
            '---- Contamination Removal Method', '--- Notes',
            '-- Cell Culture Supernatant', '--- Source', '---- Type',
            '---- Cell Line', '---- Start Date', '---- Harvest Date', '--- Tissue',
            '---- Date Obtained', '---- Tissue Type', '--- Notes',
            '-- Starting Amount', '-- Replicate Information',

```

```

'--- Biological Replicate Number', '--- Technical Replicate Number', '-- Provided
'* Pooled Biosamples', '*- Pooled Biosample', '*-- DocURL', '* Aliases', '*- A
'*-- Date Submitted to External Database', '*-- Notes'])

biosamples = biosamples.T
biosamples.insert(18, '*-- DocURL', [np.nan, 'URL', np.nan, np.nan, 'Relative ID (accession)'])
if is_time_series:
    biosamples.insert(23, '*- Property Name2', biosamples['*- Property Name'])
    biosamples.insert(24, '*-- Value2', biosamples['*-- Value'])
    biosamples.insert(25, '*- Property Name3', biosamples['*- Property Name'])
    biosamples.insert(26, '*-- Value3', biosamples['*-- Value'])
biosamples = biosamples.T
biosamples.iloc[:5, :5]

```

```

Out[8]:
      value
#property
Biosample  NaN          autoID(EXR, uniqAlphaNum, BS)  NaN
- Status   NaN  enum(Add, Modify, Hold, Cancel, Suppress, Rele...  Add
- Name     NaN                                string  NaN
- Donor ID  NaN          regexp(EXR-[a-zA-Z0-9]{6,}-D0)  NaN
-- DocURL   NaN                                url      NaN

      required
#property
Biosample      NaN  Document Describing Information About the Bios...
- Status        TRUE              Status of the document
- Name          TRUE              Name of the sample
- Donor ID      TRUE              ID of related donor document
-- DocURL       NaN  Relative ID (accession) of Donor ID doc, provi...

```

1.7 Fill in the biosamples dataframe

```

In [9]: i = 1
for mt_unique_id in samples_df.index:
    donor_id = participants.loc[samples_df.loc[mt_unique_id, 'Participant.ID'], 'donor_id']
    sample_column = 'value' + str(mt_unique_id)
    biosamples.insert(i, sample_column, biosamples['value'])
    biosamples.loc['Biosample', sample_column] = study_id + str(i) + '-BS'
    biosamples.loc['- Status', sample_column] = 'Add'
    biosamples.loc['- Name', sample_column] = 'MT.Unique.ID_' + str(mt_unique_id)
    biosamples.loc['- Donor ID', sample_column] = donor_id
    biosamples.loc['-- DocURL', sample_column] = 'coll/Donors/doc/' + donor_id
    biosamples.loc['--- Scientific Name', sample_column] = 'Homo sapiens'
    biosamples.loc['--- Common Name', sample_column] = 'Human'
    biosamples.loc['--- Taxon ID', sample_column] = 9606
    biosamples.loc['-- Disease Type', sample_column] = 'Healthy Subject'
    biosamples.loc['-- Anatomical Location', sample_column] = 'Plasma cell'
    biosamples.loc['--- Biofluid Name', sample_column] = samples_df.loc[mt_unique_id, 'Biofluid Name']

```



```

biosamples.loc['-- exRNA Source', sample_column] = ' total cell-free biofluid RNA'
biosamples.loc['-- Fractionation', sample_column] = 'Yes'
biosamples.loc['* Related Experiments', sample_column] = 1
biosamples.loc['*- Related Experiment', sample_column] = study_id + '1-EX'
biosamples.loc['*-- DocURL', sample_column] = 'coll/Experiments/doc/' + study_id +
biosamples.loc['* Custom Metadata', sample_column] = 2 if is_time_series else 1
biosamples.loc['*- Property Name', sample_column] = 'Participant.ID'
biosamples.loc['*-- Value', sample_column] = samples_df.loc[mt_unique_id, 'Participant.ID']
if is_time_series:
    biosamples.loc['*- Property Name2', sample_column] = 'timepoint'
    biosamples.loc['*-- Value2', sample_column] = samples_df.loc[mt_unique_id, 'timepoint']
    biosamples.loc['*- Property Name3', sample_column] = 'timestep'
    biosamples.loc['*-- Value3', sample_column] = samples_df.loc[mt_unique_id, 'timestep']
i += 1

biosamples = biosamples.drop('value', axis=1)
biosamples.iloc[:5, :5]

```

```

Out[9]:
value1 \
#property
Biosample          EXR-MTEWA1HealthyControls1-BS
- Status              Add
- Name                MT.Unique.ID_1
- Donor ID            EXR-MTEWA1HealthyControls1-DO
-- DocURL    coll/Donors/doc/EXR-MTEWA1HealthyControls1-DO

value107 \
#property
Biosample          EXR-MTEWA1HealthyControls2-BS
- Status              Add
- Name                MT.Unique.ID_107
- Donor ID            EXR-MTEWA1HealthyControls1-DO
-- DocURL    coll/Donors/doc/EXR-MTEWA1HealthyControls1-DO

value2 \
#property
Biosample          EXR-MTEWA1HealthyControls3-BS
- Status              Add
- Name                MT.Unique.ID_2
- Donor ID            EXR-MTEWA1HealthyControls2-DO
-- DocURL    coll/Donors/doc/EXR-MTEWA1HealthyControls2-DO

value108 \
#property
Biosample          EXR-MTEWA1HealthyControls4-BS
- Status              Add
- Name                MT.Unique.ID_108
- Donor ID            EXR-MTEWA1HealthyControls2-DO

```

```

-- DocURL      coll/Donors/doc/EXR-MTEWA1HealthyControls2-D0

                                                    value3

#property
Biosample      EXR-MTEWA1HealthyControls5-BS
- Status      Add
- Name      MT.Unique.ID_3
- Donor ID      EXR-MTEWA1HealthyControls3-D0
-- DocURL      coll/Donors/doc/EXR-MTEWA1HealthyControls3-D0

```

1.8 Strip numbers from the value columns and property indices

```

In [10]: for df in [donors, biosamples]:
          df.columns = [''.join(l for l in col if not l.isdigit() and l != '*') for col in df.columns]
          df.index = pd.Index([''.join(l for l in col if not l.isdigit()) for col in list(df.index)])

```

1.9 Load the manifest template file

```

In [11]: import json
          with open('templates/manifest_template.manifest.json', 'r') as file:
              manifest = json.load(file)
          manifest

```

```

Out[11]: {'studyName': '',
          'userLogin': '',
          'md5Checksum': '',
          'runMetadataFileName': '',
          'submissionMetadataFileName': '',
          'studyMetadataFileName': '',
          'experimentMetadataFileName': '',
          'biosampleMetadataFileName': '',
          'donorMetadataFileName': '',
          'manifest': [{'sampleName': '', 'dataFileName': ''}],
          'settings': {'analysisName': ''}}

```

1.10 Load the list of fastq filenames

```

In [12]: with open('HealthyControl_and_FeedingStudy_fastq_file.names.txt', 'r') as file:
          sample_filenames = {int(f.split('_')[0]): f.strip() for f in file} # { MT.Unique

```

1.11 Fill in the manifest

```

In [13]: import datetime
          import time

          manifest['settings']['analysisName'] = 'MTEWA1_Healthy_Controls_' + datetime.datetime.now().strftime('%Y%m%d_%H%M%S')

          manifest['studyName'] = "U01 Healthy Controls July 2018"
          manifest['userLogin'] = 'sovacool'

```

```

manifest['group'] = 'exrna-mtewa1'
manifest['db'] = 'hg19_exRNA'
manifest['runMetadataFileName'] = study_id + '-RU.metadata.tsv'
manifest['submissionMetadataFileName'] = study_id + '-SU.metadata.tsv'
manifest['studyMetadataFileName'] = study_id + '-ST.metadata.tsv'
manifest['experimentMetadataFileName'] = study_id + '-EX.metadata.tsv'
manifest['biosampleMetadataFileName'] = study_id + '-BS.metadata.tsv'
manifest['donorMetadataFileName'] = study_id + '-DO.metadata.tsv'

manifest['manifest'] = list()
for mt_unique_id in sorted(samples_df.index): # only include fastq filenames in this
    fastq_filename = sample_filenames[mt_unique_id]
    sample_name = 'MT.Unique.ID_' + str(mt_unique_id)
    manifest['manifest'].append({'sampleName': sample_name, 'dataFileName': fastq_filename})
len(manifest['manifest'])

```

Out[13]: 130

1.12 Save all the files

In [14]: `import json`

```

donors.to_csv(directory + manifest['donorMetadataFileName'], sep='\t', index=True)
biosamples.to_csv(directory + manifest['biosampleMetadataFileName'], sep='\t')

with open(directory + study_id + '.manifest.json', 'w') as file:
    json.dump(manifest, file, indent=4)

```

1.13 Validate that the metadata files listed in the manifest exist

In [15]: `import os`

```

for key in manifest:
    if 'FileName' in key:
        assert os.path.isfile(directory + manifest[key])

```

1.14 Don't forget to manually fill in experiment, run, study, and submission metadata files!