Using the API

Exercise 1 (/home/bd/bd-tutorial/UsingAPI/Conversions/):

convert.py:

```
#!/usr/bin/python -u
import browndog.bd as bd
import time
import os
import time
from os.path import isfile, join, splitext, basename

#Parameters to access BrownDog DTS
bds = 'https://bd-api.ncsa.illinois.edu'
token = ""                                      #TODO: BrownDog token
input_path = "files"                            #TODO: path to input folder
output_path = "output_" + str(int(time.time())) #TODO: path to output folder
output_format = "pdf"                           #Output format

#Create an output folder
os.makedirs(output_path)

#Create a list of files within this folder
files = [join(input_path, f) for f in os.listdir(input_path) if
isfile(join(input_path, f))]

for (input_file) in files:
  filename, file_extension = splitext(basename(input_file))
  input_format = file_extension[1:]

  print 'File: ' + basename(input_file) + ', ' + input_format + ' -> '+ output_format

  #TODO: get possible output formats for each file
  outputs = bd.outputs(bds, input_format, token)

  #TODO: if the output format is not supported, skip the conversion
  if not(output_format in outputs):
    print output_format, 'is not supported'
    continue

  #TODO: create the output file name with path
  output_file = join(output_path, filename +"." + output_format)

  #TODO: do the conversion
  bd.convert(bds, input_file, output_format, output_file, token, 60, True)
```

Exercise 2 (/home/bd/bd-tutorial/UsingAPI/Extractions/):

index.py:

```
#!/usr/bin/python -u
import browndog.bd as bd
import os
from os.path import isfile, join, splitext, basename

#Parameters to access BrownDog DTS
bds = 'https://bd-api.ncsa.illinois.edu'
token = ""            #TODO: BrownDog token
input_path = "files"  #TODO: path to input folder

#Helper function to search a dictionary by key
def findItem(obj, key):
  if key in obj: return obj[key]
  for k, v in obj.items():
    if isinstance(v, dict):
      item = findItem(v, key)
      if item is not None:
        return item

#Helper function to search for a key with given string
def search(values, searchFor):
  found = []
  for k in values.keys():
    if searchFor in values[k].lower():
      found.append(k)
  return found

#Create a list of files within this folder
files = [join(input_path, f) for f in os.listdir(input_path) if
isfile(join(input_path, f))]

text_store = {}

#Index the files
for (input_file) in files:
  filename, file_extension = splitext(basename(input_file))
  input_format = file_extension[1:]

  print 'File: ' + basename(input_file)

  #TODO: do the extraction
  metadata = bd.extract(bds, input_file, token, 120)

  #TODO: process the metadata
  for m in metadata['metadata.jsonld']:
    txt = findItem(m, 'ocr_text')
    if not (txt is None):
      text_store[basename(input_file)] = txt

#TODO: search the index for a keyword (e.g. 'information')
print text_store
print search(text_store, 'information')
```

Exercise 3 (/home/bd/bd-tutorial/UsingAPI/ConversionsAndExtractions/):

extract.py:

```
#!/usr/bin/python -u
import browndog.bd as bd
import json
import time
import os
from os.path import isfile, join, splitext, basename

#Parameters to access BD services
bds = 'https://bd-api.ncsa.illinois.edu'
token = ""                                          #TODO: BrownDog token
input_file = "files/person.pcd"                     #TODO: input file
output_path = "output_" + str(int(time.time()))     #TODO: path to output folder
output_format = "png"                               #Output format

filename, file_extension = splitext(basename(input_file))
input_format = file_extension[1:]
print 'File: ' + basename(input_file) + ', ' + input_format + ' -> '+ output_format

#Create an output folder
os.makedirs(output_path)

#Create the output file name with path
output_file = output_path + '/' + filename + "." + output_format

#TODO: do the conversion
output = bd.convert(bds, input_file, output_format, output_file, token, 60, True)

#TODO: extract metadata from the converted file
metadata = bd.extract(bds, output, token, 120)

#TODO: print the "tag" section for to if items like "faces" where detected
print json.dumps(metadata['tags'])
```