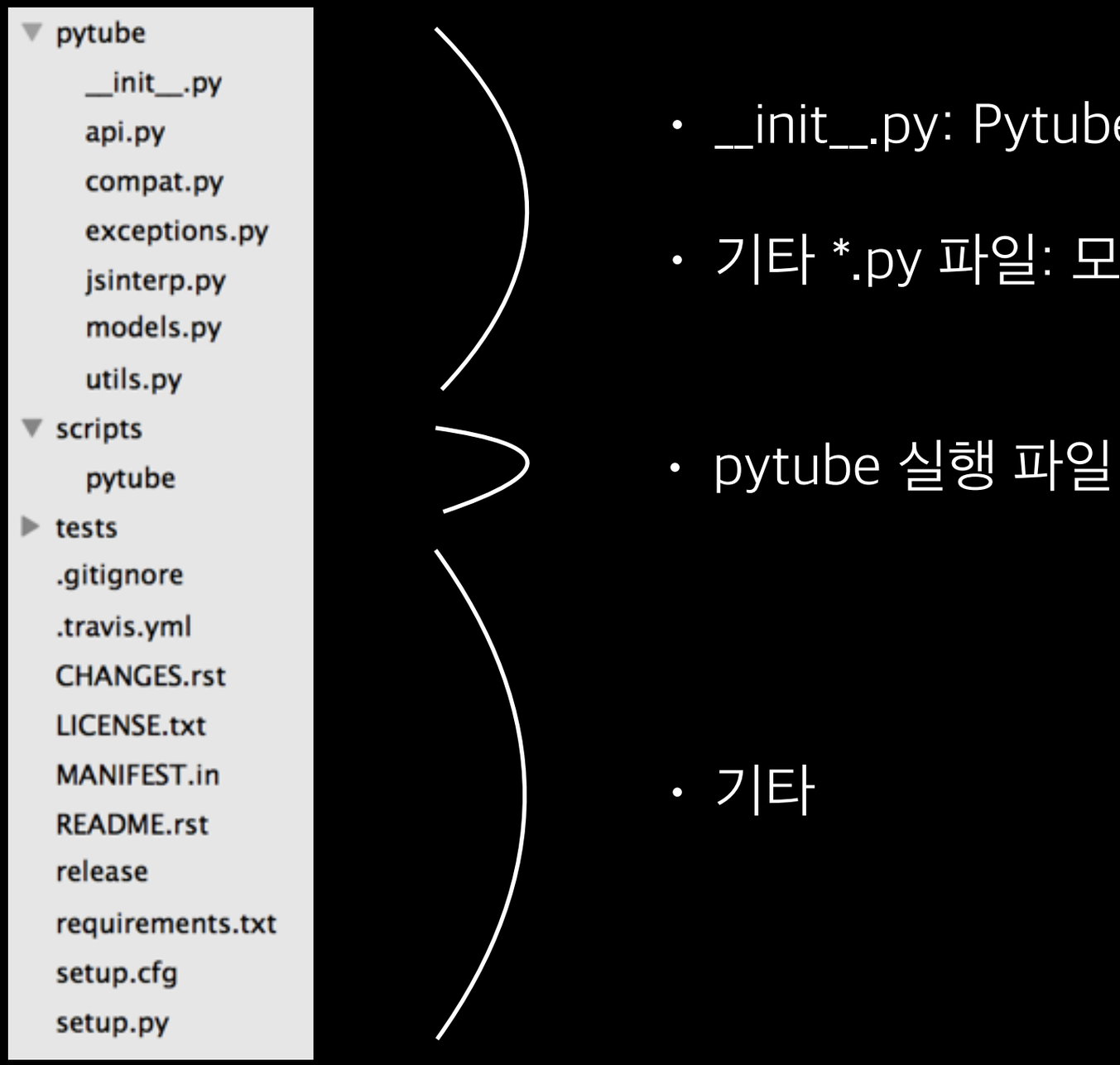
pytube?

무엇을 하는 패키지인가?

pytube 기본구조



- __init__.py: Pytube package
- 기타 *.py 파일: 모듈
- pytube 실행 파일
- 기타

pytube

- pytube
  - __init__.py
  - api.py
  - compat.py
  - exceptions.py
  - jsinterp.py
  - models.py
  - utils.py
- scripts
  - pytube
- tests
- .gitignore
- .travis.yml
- CHANGES.rst
- LICENSE.txt
- MANIFEST.in
- README.rst
- release
- requirements.txt
- setup.cfg
- setup.py

pytube

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from __future__ import print_function
import sys
import os
import argparse

from pytube import YouTube
from pytube.utils import print_status, FullPaths
from pytube.exceptions import PytubeError
from pprint import pprint


def main():
    parser = argparse.ArgumentParser(description='YouTube video downloader')
    parser.add_argument("url", help=(
        "The URL of the Video to be downloaded"))
    parser.add_argument("--extension", "-e", dest="ext", help=(
        "The requested format of the video"))
    parser.add_argument("--resolution", "-r", dest="res", help=(
        "The requested resolution"))
    parser.add_argument("--path", "-p", action=FullPaths, default=os.getcwd(),
                        dest="path", help=("The path to save the video to."))
    parser.add_argument("--filename", "-f", dest="filename", help=(
        "The filename, without extension, to save the video in."))

    args = parser.parse_args()

    try:
        yt = YouTube(args.url)
        videos = []
        for i, video in enumerate(yt.get_videos()):
            ext = video.extension
            res = video.resolution
            videos.append("{} {}".format(ext, res))
    except PytubeError:
        print("Incorrect video URL.")
        sys.exit(1)

    if args.filename:
        yt.set_filename(args.filename)

    if args.ext or args.res:
        if not all([args.ext, args.res]):
            print("Make sure you give either of the below specified "
                  "format/resolution combination.")
            pprint(videos)
            sys.exit(1)
```

pytube

- 파이썬 표준 라이브러리 및 pytube 모듈 import

- argparse 모듈로 parameter 파싱
- parser 객체 속성: url/확장자/해상도/경로/파일명
- 절대경로 미지정시, 현재 디렉토리 사용
- args에 파싱한 인자들을 할당

- try-except: 유효한 URL 입력시에만 Youtube 객체 생성
- yt: youtube 세션 객체
- enumerate(): 세션 내 모든 비디오를 불러 format후 “확장자 해상도”문자열로 videos 리스트에 저장

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from __future__ import print_function
import sys
import os
import argparse

from pytube import YouTube
from pytube.utils import print_status, FullPaths
from pytube.exceptions import PytubeError
from pprint import pprint


def main():
    parser = argparse.ArgumentParser(description='YouTube video downloader')
    parser.add_argument("url", help=(
        "The URL of the Video to be downloaded"))
    parser.add_argument("--extension", "-e", dest="ext", help=(
        "The requested format of the video"))
    parser.add_argument("--resolution", "-r", dest="res", help=(
        "The requested resolution"))
    parser.add_argument("--path", "-p", action=FullPaths, default=os.getcwd(),
                        dest="path", help=("The path to save the video to."))
    parser.add_argument("--filename", "-f", dest="filename", help=(
        "The filename, without extension, to save the video in."))

    args = parser.parse_args()

    try:
        yt = YouTube(args.url)
        videos = []
        for i, video in enumerate(yt.get_videos()):
            ext = video.extension
            res = video.resolution
            videos.append("{} {}".format(ext, res))
    except PytubeError:
        print("Incorrect video URL.")
        sys.exit(1)
```

```python
class YouTube(object):

    def __init__(self, url=None):

        self._filename = None
        self._video_url = None
        self._js_cache = None
        self._videos = []
        if url:
            self.from_url(url)
```

```python
def get_videos(self):
    """Gets all videos."""
    return self._videos
```

pytube

- 다음과 같은 parameter가 존재할 경우:,
  - 확장자 & 해상도: 해당 비디오 반환
  - 확장자 only: 필터 후 최대 해상도 동영상 반환
  - 해상도 only: 필터 후 최대 해상도 동영상 반환
  - default: 최대 해상도 동영상 반환

- try-except: 위 과정에서 선택된 동영상 다운로드
- download(): models.py내 video 클래스의 클래스 메소드

- 직접 pytube 파일 실행시에만 main() 실행

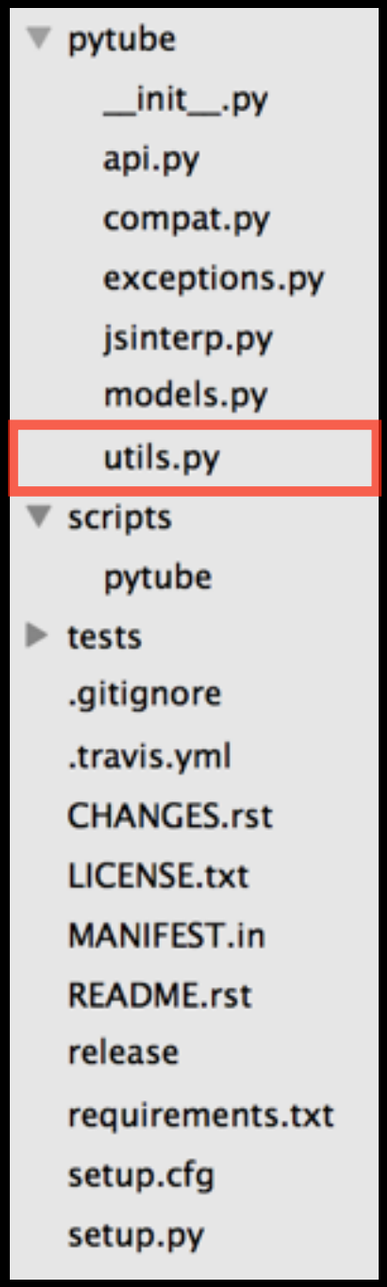
```
49
50  if args.ext and args.res:
51      # There's only ope video that matches both so get it
52      vid = yt.get(args.ext, args.res)
53      # Check if there's a video returned
54      if not vid:
55          print("There's no video with the specified format/resolution "
56                "combination.")
57          pprint(videos)
58          sys.exit(1)
59
60  elif args.ext:
61      # There are several videos with the same extension
62      videos = yt.filter(extension=args.ext)
63      # Check if we have a video
64      if not videos:
65          print("There are no videos in the specified format.")
66          sys.exit(1)
67      # Select the highest resolution one
68      vid = max(videos)
69  elif args.res:
70      # There might be several videos in the same resolution
71      videos = yt.filter(resolution=args.res)
72      # Check if we have a video
73      if not videos:
74          print("There are no videos in the specified in the specified "
75                "resolution.")
76          sys.exit(1)
77      # Select the highest resolution one
78      vid = max(videos)
79  else:
80      # If nothing is specified get the highest resolution one
81      vid = max(yt.get_videos())
82
83  try:
84      vid.download(path=args.path, on_progress=print_status)
85  except KeyboardInterrupt:
86      print("Download interrupted.")
87      sys.exit(1)
88
89  if __name__ == '__main__':
90      main()
91
```

pytube - 결론

pytube 실행파일은:

1. user가 입력한 인자들(비디오의 속성)을 파싱한 후,
2. youtube세션을 만들고,
3. user가 원하는 비디오의 속성과 일치하는 비디오를 검색해,
4. 해당 비디오가 있으면 다운로드!

utils.py




```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import argparse
import re

from os import path
from sys import stdout
from time import clock


class FullPaths(argparse.Action):
    """Expand user- and relative-paths"""
    def __call__(self, parser, namespace, values, option_string=None):
        setattr(namespace, self.dest, path.abspath(path.expanduser(values)))


def truncate(text, max_length=200):
    return text[:max_length].rsplit(' ', 0)[0]


def safe_filename(text, max_length=200):
    """Sanitizes filenames for many operating systems.

    :params text: The unsanitized pending filename.
    """

    # Tidy up ugly formatted filenames.
    text = text.replace('_', ' ')
    text = text.replace(':', ' -')

    # NTFS forbids filenames containing characters in range 0-31 (0x00-0x1F)
    ntfs = [chr(i) for i in range(0, 31)]

    # Removing these SHOULD make most filename safe for a wide range of
    # operating systems.
    paranoid = ['\"', '\#', '\$', '\%', '\'', '\*', '\,', '\.', '\/', '\:',
                '\;', '\<', '\>', '\?', '\\', '\^', '\|', '\~', '\\\\']

    blacklist = re.compile('|'.join(ntfs + paranoid), re.UNICODE)
    filename = blacklist.sub('', text)
    return truncate(filename)


def sizeof(bytes):
    """Takes the size of file or folder in bytes and returns size formatted in
    KB, MB, GB, TB or PB.

    :params bytes:
        Size of the file in bytes
    """
    alternative = [
```

utils.py

- __call__: instance로 함수를 호출해서 setattr()를 통해 경로 지정시 사용됨.

- 텍스트가 길면 오른쪽에서부터 잘라서 첫번째 반환

- filename 지정
- OS에 따라 글자 인식 가능여부가 달라지므로, 정리
- re.compile()로 정규화 패턴을 반환해 text로 변환
- truncate함수로 글자수를 잘라냄

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import argparse
import re

from os import path
from sys import stdout
from time import clock


class FullPaths(argparse.Action):
    """Expand user- and relative-paths"""
    def __call__(self, parser, namespace, values, option_string=None):
        setattr(namespace, self.dest, path.abspath(path.expanduser(values)))


def truncate(text, max_length=200):
    return text[:max_length].rsplit(' ', 0)[0]


def safe_filename(text, max_length=200):
    """Sanitizes filenames for many operating systems.

    :params text: The unsanitized pending filename.
    """

    # Tidy up ugly formatted filenames.
    text = text.replace('_', ' ')
    text = text.replace(':', ' -')

    # NTFS forbids filenames containing characters in range 0-31 (0x00-0x1F)
    ntfs = [chr(i) for i in range(0, 31)]

    # Removing these SHOULD make most filename safe for a wide range of
    # operating systems.
    paranoid = ['\"', '\#', '\$', '\%', '\'', '\*', '\,', '\.', '\/', '\:',
                '\;', '\<', '\>', '\?', '\\', '\^', '\|', '\~', '\\\\']

    blacklist = re.compile('|'.join(ntfs + paranoid), re.UNICODE)
    filename = blacklist.sub('', text)
    return truncate(filename)
```

utils.py

- 파일과 폴더 사이즈를 리턴
- bytes 인자를 받아 크기를 결정하고, 단복수 처리
- 중요X

```python
def sizeof(bytes):
    """Takes the size of file or folder in bytes and returns size formatted in
    KB, MB, GB, TB or PB.

    :params bytes:
        Size of the file in bytes
    """
    alternative = [
        (1024 ** 5, ' PB'),
        (1024 ** 4, ' TB'),
        (1024 ** 3, ' GB'),
        (1024 ** 2, ' MB'),
        (1024 ** 1, ' KB'),
        (1024 ** 0, (' byte', ' bytes')),
    ]

    for factor, suffix in alternative:
        if bytes >= factor:
            break
    amount = int(bytes / factor)
    if isinstance(suffix, tuple):
        singular, multiple = suffix
        if amount == 1:
            suffix = singular
        else:
            suffix = multiple
    return "%s%s" % (str(amount), suffix)
```

- video객체의 download() 함수에 인자로 넘어가 다운로드 % 남았는지 알려줌.
- 완료후 buffer를 비움

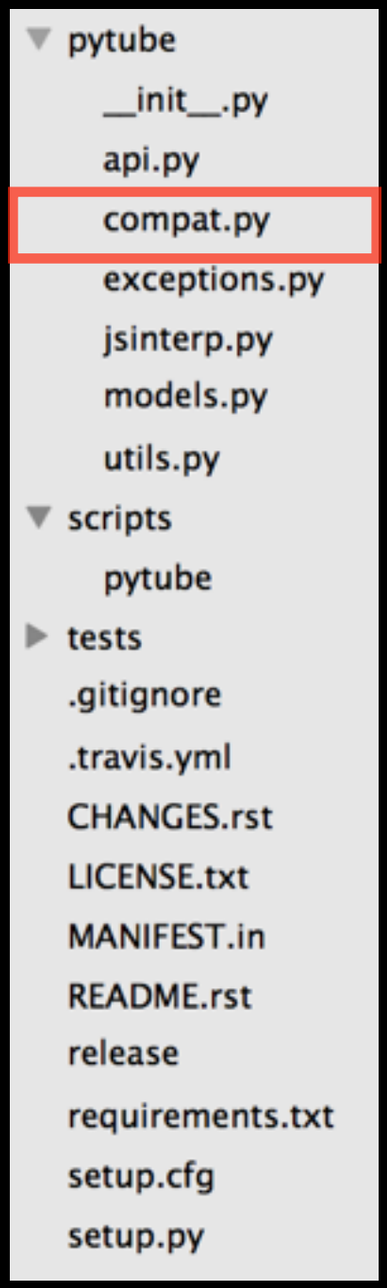
```python
def print_status(progress, file_size, start):
    """
    This function - when passed as `on_progress` to `Video.download` - prints
    out the current download progress.

    :params progress:
        The lenght of the currently downloaded bytes.
    :params file_size:
        The total size of the video.
    :params start:
        The time when started
    """

    percent_done = int(progress) * 100. / file_size
    done = int(50 * progress / int(file_size))
    dt = (clock() - start)
    if dt > 0:
        stdout.write("\r  [%s%s][%3.2f%%] %s at %s/s\r " %
                     ('=' * done, ' ' * (50 - done), percent_done,
                      sizeof(file_size), sizeof(progress // dt)))
    stdout.flush()
```

utils.py - 정리

utils.py 실행파일은:

1. Video(models.py) 또는 Youtube(api.py) 클래스 메소드가 실행될 때, 사용되는 모듈
2. 경로지정/파일명 체크/크기 확인 및 다운로드 상태 확인 등

compat.py




```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# flake8: noqa
import sys

PY2 = sys.version_info[0] == 2
PY3 = sys.version_info[0] == 3

if PY2:
    from urllib2 import urlopen
    from urlparse import urlparse, parse_qs, unquote
if PY3:
    from urllib.parse import urlparse, parse_qs, unquote
    from urllib.request import urlopen
```

- 호환성 확인: 파이썬 2와 3에 따라 모듈 경로가 다르므로 타입체크 후 사용