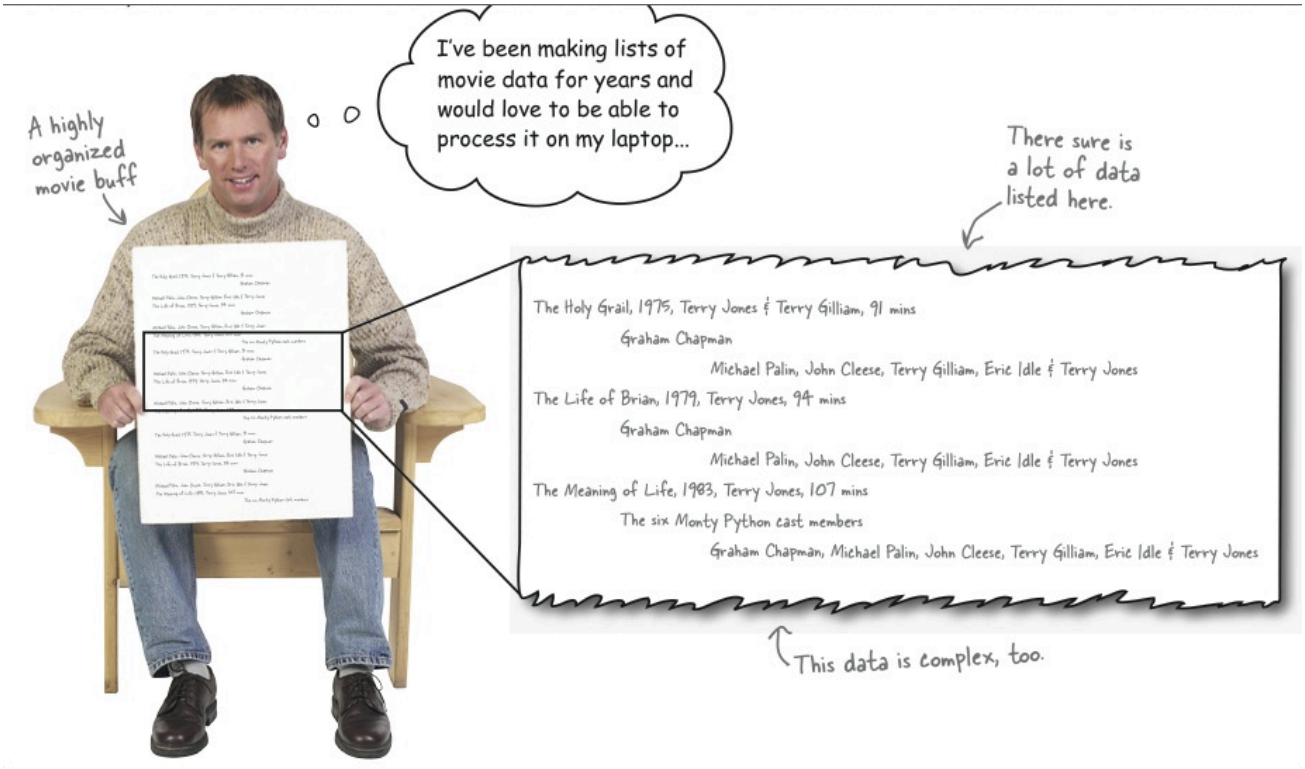
Ch2. 코드 공유하기: 함수들의 모듈

- summarized by 전승일(2014/9/20)
- source code: http://www.headfirstlabs.com/books/hfpython/
- support site: http://python.itcarlow.ie




```
movies = ["The Holy Grail", 1975, "Terry Jones & Terry Gilliam", 91,
                ["Graham Chapman",
                        ["Michael Palin", "John Cleese", "Terry Gilliam", "Eric
```

혼자 쓰기엔 아까운 코드

- 함수를 모듈로 바꾸고, 배포 유틸리티를 사용해서 모듈을 전체 파이썬 커뮤니티에 공유
- 모듈을 모아서 배포 단위로 만든 것이 패키지이나, 본 장에서는 1개의 모듈로 1개의 패키지를 구성하므로 서로 같은 의미로 혼용

함수를 모듈로 바꾸기

- **모듈**: 파이썬 코드가 들어 있는 텍스트 파일
  파일 이름이 **.py** 확장자로 끝나기만 하면 됨
  파일명이 모듈명이 되므로 파일명에 제한이 있음 (공백, hyphen포함 불가, 파이썬 내장함수명과 다른 파일명 사용)
- 1장에서 만든 print_lol 함수를 nester.py 파일로 저장

```
def print_lol(the_list):
    for each_item in the_list:
        if isinstance(each_item, list):
            print_lol(each_item)
        else:
            print(each_item)
```

온 세상에 모듈이...

- **PyPI** : Python Package Index, 인터넷에서 써드파티 파이썬 모듈을 제공하는 중앙 repository
- http://pypi.python.org

코드에 주석을 추가하세요

- 주석을 여러 라인에 작성하려면 **큰따움표 세 개** 혹은 **작은따옴표 세 개** 사용

```
"""This is the standard way to
   include a multiple-line comment in
   your code."""
```

- 파이썬 모듈이 컴퓨터 어디에 설치되어 있는지?

```
import sys; sys.path
```

- IDLE 실습: FILE-Open 메뉴를 선택하여 nester.py 파일을 편집창으로 불러들이고 F5 키로 모듈 실행
  movies list 만들고, print_lol(movies)로 확인

질문 (교재 74쪽 참조)

배포 준비

1. 모듈을 위한 빈 디렉토리 만들기

```
cd ~
mkdir nester
cd nester
```

2. 새로 만든 폴더에 setup.py 파일 생성 (배포 패키지에 대한 메타 데이터 포함)

```
from distutils.core import setup

setup(
        name         = 'nester',
        version      = '1.0.0',
        py_modules   = ['nester'],
        author       = 'hfpython',
        author_email = 'hfpython@navercorp.com',
        url          = 'http://www.naver.com',
        description  = 'Test - A simple printer of nested lists',
     )
```

배포 패키지 만들기

1. 배포 패키지 생성

```
$ python3 setup.py sdist
```

2. 배포 패키지를 local에 install 하기

```
$ sudo python3 setup.py install --record files.txt
```

배포 패키지 돌아보기

- 디렉토리 상태: 교재 78쪽 그림 참조

모듈을 사용하려면 import 하세요

```
    >>> import nester
```

연습문제

- 새로 만든 모듈을 import하고 cast라는 리스트를 정의하고, 모듈에서 정의한 함수를 사용하여 리스트의 내용을 출력하는 간단한 프로그램 작성

```
import nester

cast = ["Palin", "Cleese", "Idle", "Jones", "Gilliam", "Chapman"]

print_lol(cast)
```

- name 'print_lol' is not defined 에러 발생

파이썬 모듈은 네임스페이스를 만듭니다

- nester.print_lol 로 함수를 호출
- 모듈의 함수명을 __main__ namespace에 바로 불러 들이는 방법 ("연장통과 도구"로 비유설명)

```
from nester import print_lol

print_lol(cast)
```

PyPI 웹사이트에 등록하기

- pypi.python.org 에 접속해서 ID 만들기

코드를 PyPI에 등록하기

- 모듈 자체를 PyPI에 등록 하기 (모듈 별로 한번만 수행)

```
$ python3 setup.py register
```

- 모듈 업로드 하기

```
$ python3 setup.py sdist upload
```

- 모듈명 바꾸기

  1. setup.py 파일 안의 name, py_modules 인자를 변경하고,
  2. nester.py 파일 이름을 변경한 모듈 이름으로 바꾸고,
  3. python3 setup.py sdist 명령으로 다시 패키지를 만들면 됨

PyPI 커뮤니티에 오신 걸 환영합니다

- 현재 모듈 개수: 48997

질문(교재 85쪽 참조)

인자를 이용해서 행위 제어하기

- 들여쓰기로 출력하고 싶으면 새로운 함수를 만들어야 하나?
- 함수를 추가하지 말고 기존 함수에 새로운 인자를 추가하고, 기존 함수를 기존 방식대로 호출하는데는 API 변경 없이 구현

새로운 코드를 짜기 전에 내장 함수를 먼저 생각하세

요

- 일반적인 기능이라고 생각되는 요구 사항을 만났을 때는 내장함수가 있는지 먼저 확인
- range 내장합수

```
for num in range(4):
    print(num)
```

질문(교재 91쪽 참조)

- 내장함수는 **builtins** 라는 자체 namespace에 속함

```
for keyword in dir(__builtins__):
    print(keyword)
```

들여쓰기 인자 추가하여 print_lol 함수 수정

```
    def print_lol(the_list, level):
        for each_item in the_list:
            if isinstance(each_item, list):
                print_lol(each_item)
            else:
                for tab_stop in range(level):
                    print("\t", end('')
                print(each_item)

    nester.print_lol(movies, 0)
```

- TypeError: ... 에러 발생

파이썬은 최선을 다해 여러분의 코드를 실행하려 합니다

- 파이썬은 실행하기 전까지는 코드의 무결성을 완전히 확인하지 않음 (compile형 언어가 아님)

- 장점: 런타임에 함수를 동적으로 정의할 수 있음(유연성)
- 단점: 컴파일 언어에서 에러로 잡히는 것이 파이썬에서는 모르고 넘어가는 경우가 있으므로 주의 필요

코드 추적하기

- 에러의 의미 파악 -> 재귀 호출 시 인자 개수 오류 임

코드 오류 찾아내기

- print_lol(each_item)을 print_lol(each_item, level + 1)로 수정 필요

새 코드를 PyPI에 갱신하세요

- setup.py에서 version 수정 후

```
$ python3 setup.py sdist upload
```

여러분은 API를 바꿨습니다

- 기존에 print_lol(list) 형태로 호출하던 함수가 print_lol(list, level) 형태로 바뀜
- nester 모듈을 이용해 프로그램을 짜 놓은 사용자는 nester 모듈을 업그레이드하면 소스를 모두 수정해야 함 ㅠㅠ
- 기존에 짜여진 프로그램을 바꾸지 않는 하위 호환성이 있는 API를 제공해야 함

선택적 인자를 사용하세요

```
def print_lol(the_list, level)      # level은 필수인자임
    ...

def print_lol(the_list, level = 0)  # level은 선택적 인자이고 default 값이 0임
```

```
    ...

nester.print_lol(movies)로 호출하면, 내부적으로 nester.print_lol(movies, 0)로 호출 됨
```

여러분의 모듈은 두 API 모두 지원합니다

- 하위 호환성을 가지는 버전으로 변경 후 PyPI에 다시 업로드하여 배포
- 하지만, indent를 원하지 않는 기존 사용자는 이 버전으로 올리면 문제가 됨

API가 아직도 제대로 되지 않았어요

- 새로운 인자를 추가해서, 들여쓰기를 원하면 True로 지정하고, 그렇지 않으면 False로 지정하도록 하고 기본값은 False로 하면 됨

```
def print_lol(a_list, indent=False, level=0):
    for each_item in a_list:
        if isinstance(each_item, list):
            print_lol(each_item, indent, level+1)
        else:
            if indent:
                for l in range(level):
                    print("\t", end='')
            print(each_item)
```

모듈에 대한 평판이 다시 좋아졌어요

- 여러분의 파이썬 기술이 발전하고 있어요.
- test 후에 자신의 모듈을 지우는 방법 (불필요한 모듈이므로 삭제하는 것이 좋음)
  - http://pypi.python.org 에서 login 후
  - 우측 상단에서 나오는 자신의 모듈명 클릭
  - 업로드한 모듈 지우기: 개별 모듈 버전명을 체크한 후 "Remove" 클릭
  - "Remove this package completely" 클릭하여 package 자체를 삭제

Additional Information