Fastcampus

Computer Science Extension School

Python Basic_Day6

Review

Leap year

4로 나뉘어 떨어지면 윤년,
100으로 나뉘어 떨어지면 평년,
400으로 나뉘어 떨어질땐 윤년

Leap year(answer)

```
leap = False
def is_leap(y):
        if y % 4 == 0 and (y % 100 != 0 or y % 400 == 0)
                leap = True
        return leap

y = int(input("Is leap?? "))
print(is_leap(y))
```

variable outside function

```
a = "hello"
def glob_test(a):
        a += "world"
        return a

glob_test(a)
print(a)
```

```
a = "hello"
def glob_test(x):
        x += "world"
        return x

glob_test(a)
print(a)
```

variable outside function

```
def glob_test2(x):
        x += "world"
        return x

glob_test2("hello")
glob_test2(x)
```

So, how to globalize

(1) using return

```
a = "hello"
def glob_test(a):
        a += "world"
    return a

a = glob_test(a)
print(a)
```

So, how to globalize

(2) use global

```
a = "hello"
def glob_test(a):
        global a
        a += "world"
    return a

glob_test(a)
print(a)
```

`global 이라는 명령을 사용하여 전역변수로 사용하게 되면 함수는 독립성을 잃게 되어 함수가 외부변수에 의존적이게 됩니다.`

numguess with function

```
def guesser(guess):
        if guess == answer:
                print("Correct! The answer was ", str(answer))
                break
        else:
                print("That's not what I wanted!! Try again!!")
```

Recursive

```
times = int(input("How many times want to curse the beast??: "))
def recurse_beast(a):
        if a == 0:
                print("curse complete!")
        else:
                print("Fusion!!!(%d times left)" % a - 1)
                recurse_beast(a-1)

recurse_beast(times)
```

Shell Command

Shell

> 운영체제(OS)에서 사용자의 명령을 해석하여 대신 실행해주는 프로그램

- **Unix, Linux**

`bash(ubuntu, OSX default)`, sh, zsh, csh, ksh, ...

- **Windows**

explorer.exe(for GUI Windows)
cmd.exe(for CLI MS-DOS)

몇가지 간단한 Shell 명령어

`$` : Shell 명령어의 시작

```
$ ls
$ ls -al

$ cd Documents
$ mkdir css
$ cd css
$ mkdir python && cd python
```

몇가지 간단한 Shell 명령어

```
$ mkdir python - make directory python
$ cd python - change directory
$ cd .. - up to


$ touch hello.py - create hello.py
$ exit - terminate shell
$ mv hello.py /python
$ cp hello.py /python

$ python --version
$ python --help
```

Vim

> Vi IMproved의 약자로 vi 호환 텍스트 에디터

vim Basic

```
$ vim hello.py
```

Command

```
h,j,k,l - cursor
i - insert
v - visual
d - delete
y - yank
p - paste
u - undo
r - replace
$ - move end of line
^ - move start of line

:q - quit
:q! - quit(no warning)
:wq - write and quit

:{number} - move to {number}th line
```

git Basic


git

VCS (Version Control System)

== SCM (Source Code Management)

< SCM (Software Configuration Management: 형상관리)

chronicle of git

chronicle of git

- Linux Kernal을 만들기 위해 Subversion을 쓰다 화가 난 리누스 토발즈는 2주만에 git이라는 버전관리 시스템을 만듦
  git official repo

Characteristics of git

- 빠른속도, 단순한 구조
- 분산형 저장소 지원
- 비선형적 개발(수천개의 브랜치) 가능

Pros of git

- **중간-발표자료_최종_진짜최종_15-4(교수님이 맘에들어함)_언제까지??_이걸로갑시다.ppt**
- 소스코드 주고받기 없이 동시작업이 가능해져 생산성이 증가
- 수정내용은 **commit** 단위로 관리, 배포 뿐 아니라 원하는 시점으로 **Checkout** 가능
- 새로운 기능 추가는 **Branch**로 개발하여 편안한 실험이 가능하며, 성공적으로 개발이 완료되면 **Merge**하여 반영
- 인터넷이 연결되지 않아도 개발할 수 있음

Open-source project

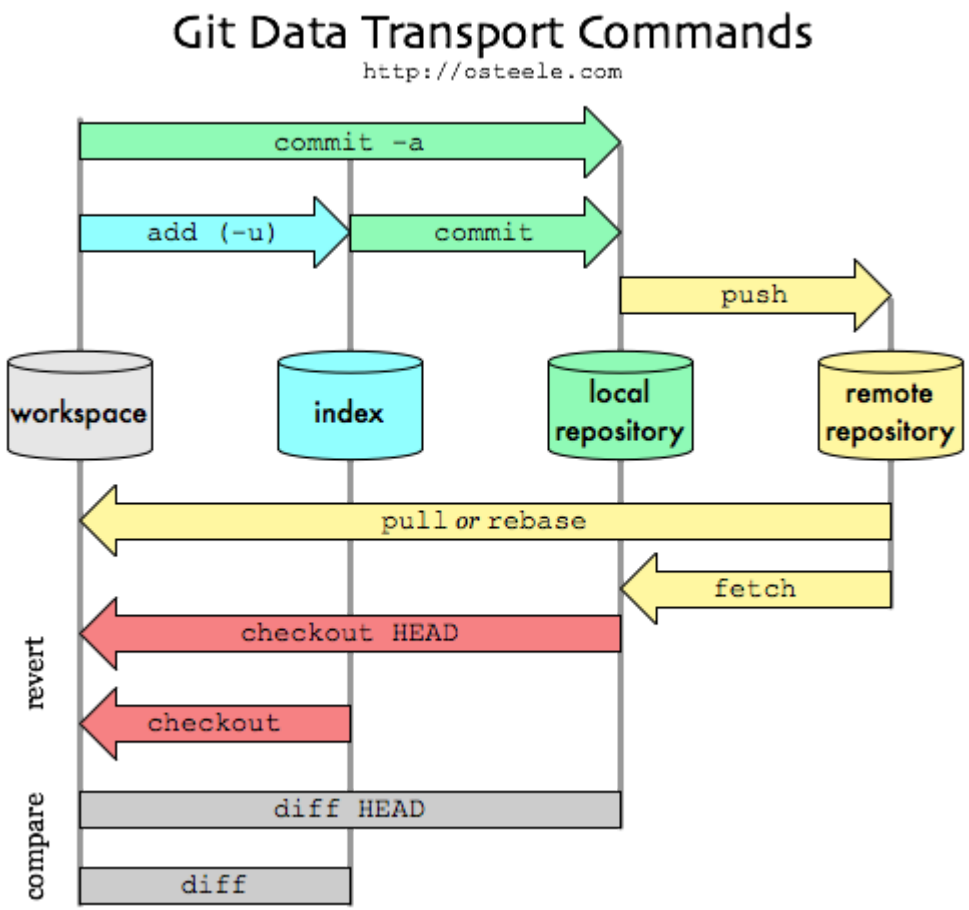
https://github.com/python/cpython
https://github.com/tensorflow/tensorflow

https://github.com/JuliaLang/julia
https://github.com/golang/go

git inside

- Blob: 모든 파일이 Blob이라는 단위로 구성
- Tree: Blob(tree)들을 모은 것
- Commit: 파일에 대한 정보들을 모은 것

git Process and Command



Useful manager for mac

http://brew.sh/index_ko.html

install git

https://git-scm.com/

```
// MacOS
$ brew install git
// Linux
$ sudo apt-get install git
```

- Windows: install git bash

`$ git --version` 으로 정상적으로 설치되었는지를 확인

git is not equal to github

sign up github

https://github.com/

important!!

- 가입할 `email`과 `username`은 멋지게
- private repo를 원한다면 $7/month

Set configuration

terminal

```
$ git config --global user.name "{{username}}"
$ git config --global user.email "{{github email address}}"
$ git config --list
```

My First Repo

Let's make your first repo with github

My First Repo

```
$ git init
$ git add .
$ git commit -m "some commit"
```

After create new repo through github,

```
$ git remote add origin
https://github.com/{{username}}/{{repo}}.git
$ git push origin master
```