

Week 01 • 소셜네트워크 데이터마이닝과 분석

Introduction to Social Computing

Joonhwan Lee
human-computer interaction + design lab.

오늘 다룰 내용

- 수업 소개
- 소셜 컴퓨팅(Social Computing)이란?
- 프로그래밍(programming)이란?
- 파이썬(Python)이란 무엇인가?
- 파이썬(Python)을 사용해 보자.

1. 수업 소개

담당 교수 및 TA

- ◆ 담당교수: 이준환
 - ◆ Email: joonhwan@snu.ac.kr
 - ◆ Office: 64동 405호
 - ◆ TA: 이규호 (art.and.play.7@gmail.com)
- ◆ 수업 홈페이지
 - ◆ <http://hcid.snu.ac.kr/courses/socialcomp2018/>

수업 개요

- ◆ 소셜 컴퓨팅(social computing)과 라지데이터 분석(large data analysis) 등이 커뮤니케이션 분야에서도 중요한 이슈로 부상함에 따라 컴퓨터공학을 전공하지 않은 연구자들도 소셜 네트워크 시스템의 기술적, 구조적 특성을 이해할 필요가 있다. 이 수업에서는 스크립팅 프로그래밍 언어인 파이썬(Python)을 사용하여 컴퓨터 프로그래밍의 기초를 학습하고, 웹 기반 기술(web technology), 데이터베이스 등의 관련 기술에 대한 학습을 통해 실제로 소셜 네트워크 데이터를 수집하고 분석하는 방법을 배운다.

강의 내용

- ◆ 파이썬을 이용한 기초 프로그래밍
- ◆ 웹 기반 기술(web technology)
- ◆ 소셜 데이터 마이닝(social data mining)을 통한 데이터 분석

수업진행 계획

- ❖ 강의 계획서 참고
 - ❖ <http://hcid.snu.ac.kr/courses/socialcomp2018/>

과제 및 평가

- ❖ 강의 계획서 참고
 - ❖ <http://hcid.snu.ac.kr/courses/socialcomp2018/>

2. 소셜컴퓨팅(Social Computing) 이란?

Social Computing

- ♦ Group of university researchers to make web science a field of study
 - Steve Lohr, New York Times, Nov. 2, 2006
 - http://www.nytimes.com/2006/11/02/technology/02compute.html?_r=3&adxnnl=1&oref=slogin&adxnnlx=1212113936-DbHS7WsdpYrJCC4d1pZXmw&

Social Computing

- ♦ “Web science, the researchers say, has **social and engineering dimensions**. It extends well beyond traditional computer science, they say, to include the **emerging research in social networks and the social sciences** that is being used to study **how people behave on the Web**. And Web science, they add, shifts the center of gravity in engineering research from how a single computer works to how huge decentralized Web systems work.”

Social Computing

- ♦ “Computer science is at a turning point, and it has to go **beyond algorithms and understand the social dynamics of issues** like trust, responsibility, empathy and privacy in this vast networked space.”
- Ben Shneiderman, a Professor at the University of Maryland

Social Computing

- ❖ Social computing is an area of computer science that is concerned with the **intersection of social behavior and computational systems.**
- ❖ Social computing is the collaborative and interactive aspect of online behavior.
 - Wikipedia, http://en.wikipedia.org/wiki/Social_computing

Why Social Computing?

- ◆ Facebook: 실질 사용자가 13억 명 돌파 (2014)
 - ◆ DAU(Daily Active User): 8억 4천명
- ◆ 2억 8천만명의 실질 트위터 사용자 (2014)
 - ◆ 하루에 5억 건의 트윗 메시지 생산
- ◆ 2011년 기준 스마트폰 보급률 53%, 2012년에는 80% 이상 예상
 - ◆ 정치, 사회, 문화 모든 면에서 social media 가 사람들의 일상 생활 속의 서비스로 자리매김

Why Social Computing?

- ◆ 인간의 사회적 행위를 지원
 - ◆ 사회적 커뮤니티를 위한 온라인 커뮤니케이션을 지원하는 웹 2.0 서비스와 도구
 - ◆ 블로그, 위키, 소셜 네트워크 서비스, 북마킹
- ◆ 사람들의 집단적 협력과 지능을 활용
 - ◆ 협력 필터링(Collaborative Filtering), 추천(Recommendation), 예측(Forecasting), 평판 (Reputation) 기술 등

소셜미디어 연구

facebook.

twitter

- ♦ 소셜 네트워크의 급속한 성장
→ 트위터, 페이스북 등의 소셜미디어 서비스 사용자 급증
 - ♦ 사적 정보의 교류
 - ♦ 뉴스와 같은 정보성 메시지의 소비, 공유
- ♦ 소셜미디어 사용 목적: 정보 공유와 사회적 친분 관계 유지
 - ♦ Java, Song, Finin, & Tseng, 2007; Naaman, Boase, & Lai, 2010; Zhao & Rosson, 2009
- ♦ 소셜미디어 사용자의 행동 패턴을 정량적으로 분석하는데에 주로 초점을 맞추고 있음.

관심 연구 주제

- ♦ 소셜미디어에서의 행동 분석
 - ♦ **why** - 왜 그런 행동을 하는지...?
 - ♦ follower-following study, homophily study, depression study, github study
 - ♦ **what** - 무엇을 이야기하고 공유하는지...?
 - ♦ text mining, opinion mining, word network analysis
 - ♦ **how** - 서로 어떻게 영향을 주고 받는지...?
 - ♦ social viewing and political judgements study

→ 이런 결과를 어떻게 활용할 수 있는지...?

연구의 방향

- ❖ Analyzing Connections
 - ❖ Social Network Analysis
 - ❖ Reachability
 - ❖ Distance & Number of Paths
 - ❖ Degree of Node
 - ❖ Centrality
 - ❖ Morphology Changes

연구의 방향

♦ Analyzing Activities

- ♦ 사람들의 행동에는 “어떤” 이유가 있을 것.
 - ♦ 예: 우울함의 전조가 보이는 사람들은 소셜미디어에서 어떻게 행동하는가?
 사람들은 왜 소셜미디어에서 친구관계를 맺는가?
- ♦ 행동의 결과를 종속변수로 사용
- ♦ 행동의 내용을 독립변수로 사용
- ♦ 다량의 소셜네트워크 데이터를 API를 통해 수집
- ♦ 사례: Depression Study of Facebook Users
 - ♦ 종속변수 - Depression Measure: CES-D
 - ♦ 독립변수 - Facebook activities
 - ♦ number of likes, number of comments, how often they login to facebook,
 how often they change their profile picture, how much comments they
 receive in a day...

연구의 방향

- ◆ Analyzing Activities

- ◆ Linear model analysis

$$Y_i = \beta_0 + \beta_1 \phi_1(X_{i1}) + \cdots + \beta_p \phi_p(X_{ip}) + \varepsilon_i \quad i = 1, \dots, n$$

- ◆ Y_i : survey results
 - ◆ X_{ij} : independent variables
(crawled data)

Residuals:					
Min	1Q	Median	3Q	Max	
-5.5031	-1.1005	0.0535	1.3613	3.8545	
Coefficients:					
	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	3.567127	0.073872	48.288	< 2e-16 ***	
likes_from_friends	-0.006566	0.006908	-0.950	0.34212	
likes_to_friends	0.003180	0.004377	0.727	0.46770	
comments_from_friends	0.014643	0.006913	2.118	0.03441 *	
comments_to_friends	0.008820	0.005714	1.544	0.12303	
photo_tags_from_friends	0.092660	0.090532	1.024	0.30633	
photo_tags_to_friends	-0.030312	0.041907	-0.723	0.46966	
status_tags_from_friends	0.566128	0.311284	1.819	0.06927 .	
status_tags_to_friends	0.936695	0.294069	3.185	0.00149 **	
msg_from_friends	0.004335	0.001307	3.315	0.00095 ***	
msg_to_friends	-0.004753	0.001354	-3.509	0.00047 ***	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1					
Residual standard error: 1.899 on 956 degrees of freedom					
Multiple R-squared: 0.05099, Adjusted R-squared: 0.04106					
F-statistic: 5.137 on 10 and 956 DF, p-value: 2.409e-07					

연구의 방향

- ♦ Analyzing Messages
 - ♦ 어떤 이야기를 소셜미디어를 통해 하는가?
 - ♦ Text mining
 - ♦ frequency
 - ♦ text categorization
 - ♦ topic extraction
 - ♦ Opinion mining (sentiment analysis)
 - ♦ computational study of opinions, sentiments
 - ♦ 예: 이베이/아마존 리뷰
 - ♦ Activity 연구와 관련, 예측모델을 만들 수 있으나 한글 분석 이슈가 있음
 - ♦ “Predicting Postpartum Changes in Emotion and Behavior via Social Media” (de Choudhury et al. 2013)



3. 프로그래밍(programming) 이란?

Ciao Espresso~



fully automatic • serves 2 shots • \$3,000 or more?

Ciao Espresso~



fully automatic • serves 2 shots • \$3,000 or more?

Ciao Espresso~



Ciao Espresso~

- ❖ Short Instruction
 - ❖ 그라인더에 간 커피를 portafilter에 넣고 꾹꾹 잘 누른 다음 커피머신에 돌려서 끼우고 버튼만 누르면 됨.

Ciao Espresso~

- ♦ portafilter에 뜨거운 물을 내려부어 따뜻하게 한다
- ♦ 커피를 portafilter에 담는다
- ♦ portafilter를 커피머신에 돌려 끼운다
- ♦ 에스프레소 잔을 올려놓고 버튼을 누른다

Ciao Espresso~

- ◆ 물이 없다면
 - ◆ 물탱크에 물을 담는다
- ◆ 물이 있다면
 - ◆ portafilter에 뜨거운 물을 내려부어 따뜻하게 한다
- ◆ 커피가 없다면
 - ◆ 커피를 그라인드로 간다
- ◆ 커피가 있다면
 - ◆ 커피를 portafilter에 담는다
- ◆ portafilter를 커피머신에 돌려 끼운다

Ciao Espresso~

- ◆ 한잔만 만들 경우
 - ◆ 에스프레소 잔을 올려놓고 버튼을 누른다
- ◆ 한잔 이상일 경우
 - ◆ 에스프레소 잔을 올려놓고 버튼을 누른다를 반복한다
- ◆ 아메리카노일 경우
 - ◆ 큰 컵에 뜨거운 물을 붓는다
 - ◆ 에스프레소 샷을 담는다
- ◆ 아이스 아메리카노일 경우
 - ◆ 큰 컵에 얼음을 넣는다
 - ◆ 찬 물을 붓는다
 - ◆ 에스프레소 샷을 담는다

프로그래밍 (Programming)

- ◆ 컴퓨터라는 지능을 가지고 있지 못한 기계에게 일의 순서를 알려주는 것
 - ◆ 모든 컴퓨터는 내부에 CPU를 가지고 있는데, 이 CPU는 다양한 명령어 집합(instruction sets)을 가지고 있다
 - ◆ 프로그래밍은 각각의 명령어 집합에 순차적으로 명령(instruction)을 내리는 과정
- ◆ 즉, 컴퓨터가 어떠한 일을 어떻게 처리해야 할지를 알려주는 방법

프로그래밍 언어 (Programming Language)

- ❖ A programming language is an **artificial language** designed **to communicate instructions to a machine**, particularly a computer. Programming languages can be used to create programs that control the behavior of a machine and/or to express algorithms precisely.
 - wikipedia (http://en.wikipedia.org/wiki/Programming_language)

프로그래밍 언어 (Programming Language)

- ♦ 어플리케이션을 개발하기 위해서는 프로그램 언어가 필요
 - ♦ 프로그래머 → 프로그래밍 언어 → 프로그램 작성
(programming) → 해석기 (compiler) → 실행
 - ♦ 프로그래밍 언어의 종류 (언어 표현의 난이도)
 - ♦ 저수준 언어 (Low-level Language)
 - ♦ 고수준 언어 (High-level Language)

프로그래밍 언어 (Programming Language)

♦ 저수준 언어 (Low-level Language)

- ♦ CPU가 직접 해독하고 실행이 가능하도록 비트 단위로 쓰여진 언어: 기계어 (Machine Language)
 - ♦ 컴퓨터 실행에는 효율적이지만, 사람이 이해할 수가 없어 작성하기에 불편
 - 유지/보수 거의 불가능
 - ♦ Machine Language (기계어) 와 Assembly Language 가 대표적

Machine Language

```
169 1 160 0 153 0 128 153 0 129 153 130 153 0 131 241 96
```

BASIC

```
5 FOR I=1 TO 1000: PRINT "A";: NEXT I
```

프로그래밍 언어 (Programming Language)

- ❖ 저수준 언어 (Low-level Language)
 - ❖ Assembly Language
 - ❖ 기계어 명령을 어느 정도 해독할 수 있도록 문자화 한 형태의 언어
 - ❖ CPU/OS 에 따라 명령어 구조가 다름
 - ❖ 어셈블리어도 기본적으로 기계어로 번역되는 과정을 거쳐야
 - ❖ 고수준 언어보다 빠른 처리 속도
→ 임베디드 프로그램 작성 시에 많이 사용

```
section .text
global _start
_start:
    mov edx, len
    mov ecx, msg
    mov ebx, 1
    mov eax, 4
    int 0x80

    mov eax, 1
    int 0x80

section .data
msg db 'Hello, world!', 0xa
len equ $ - msg
```

프로그래밍 언어 (Programming Language)

- ❖ 고수준 언어 (High-level Language)
 - ❖ 어셈블리어의 문제점
 - ❖ 사람이 해독하기 여전히 어렵다
 - ❖ 컴퓨터 시스템 (CPU) 마다 서로 다른 명령어를 사용해야 한다
 - ❖ 일상적으로 사용하는 언어와 비슷한 구조의 랭귀지 등장
 - ❖ 컴파일러 통해 기계어로 해석하기 때문에 랭귀지 레벨에서는 시스템에 따라 다른 명령어를 사용할 필요가 없다
 - ❖ Fortran, Cobol, C, Pascal, Basic, C++, Java, Processing, Objective-C, Python, Ruby 등등

프로그래밍 언어 (Programming Language)

#JAVA

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

#ANSI C

```
#include <stdio.h>  
  
/* Hello */  
int main(void)  
{  
    printf("Hello, World!");  
    return 0;  
}
```

#Ruby

```
puts "Hello, World!"
```

#Objective-C/Cocoa

```
#import <Foundation/Foundation.h>  
  
int main (int argc, const char * argv[ ]) {  
    NSLog(@"Hello, World!");  
    return 0;  
}
```

<http://www.scriptol.com/programming/hello-world.php>

컴파일러 (Compiler)

- ❖ 기계어 이외의 다른 언어로 작성된 프로그램 → 기계어로 번역되어야 실행가능
- ❖ 컴파일러
 - ❖ 고수준 언어로 작성된 소스코드 (Source Code)를 기계어로 번역한 후 오브젝트코드 (Object Code)로 저장하는 프로그램
 - ❖ Object Code: 기계어로 변환된 프로그램
 - ❖ .exe, .com, .app (mac os x) 과 같은 실행파일이 이에 해당
 - ❖ 이미 기계어로 변환된 상태라 실행 속도가 빠름
 - ❖ 기계어로 암호화 되어 소스코드를 볼 수 없다 → decompile 이 가능한 경우 도... (decompile: 원 소스 코드 상태로 되돌리는 것)
 - ❖ 오류나 구조 변경 시, 재 compile 과정을 거쳐야

인터프리터 (Interpreter)

- ♦ Compiler 와 같이 미리 기계어로 번역된 프로그램을 만드는 것이 아니라 소스코드를 그때 그때 기계어로 해석하여 실행
- ♦ BASIC
- ♦ Ruby
- ♦ Python
- ♦ JavaScript
- ♦ 컴파일된 어플리케이션 보다 실행속도 느림 → CPU 파워의 발전으로 어느 정도 해소
- ♦ 개발/업데이트 용이 → 프로토타이핑 툴에 많이 사용

프로그래밍 언어의 발전과정

- ♦ 초기의 프로그래밍 언어
 - ♦ 어셈블리 언어의 등장 → 기계어 대체 언어로 초기에 많이 사용
 - ♦ 기계어 보다 쉬우나 여전히 배우기는 어려운 언어
 - ♦ 반도체나 산업기계 제어 프로그램과 같은 영역에서만 사용
- ♦ 최초의 고수준 프로그래밍 언어
 - ♦ FORTRAN (FORmula TRANslator):
1954년 공학계산을 위한 프로그래밍 랭귀지로 개발 → 우주선 궤도 계산, 선박 설계 등 과학 계산에 (아직도) 많이 사용 됨.
 - ♦ COBOL: 데이터 처리 목적으로 개발
구조가 구어체 처럼 간단하고 읽기 쉬워 회계처리, 비즈니스 등에 주로 활용

프로그래밍 언어의 발전과정

- ♦ Procedural Programming Language (절차적 프로그래밍 언어)
 - ♦ 소프트웨어 개발이 활발해지면서 보다 체계적인 언어가 필요
 - ♦ 수행할 내용을 순서대로 체계적으로 작성해 주는 절차적 프로그래밍 언어가 등장
 - ♦ 이전의 코드는 필요에 따라 코드 내에서 이리 저리 이동하여 복잡도가 증가
 - ♦ 입력, 계산, 데이터 처리, 출력 등의 과정을 체계적으로 반복
 - ♦ Structured Program (구조적 프로그래밍) → if/else, for, while 등 제어문 사용
 - ♦ 프로그램 구조가 인덱스 페이지와 챕터처럼 구성
 - ♦ 대표적인 절차적 프로그래밍 언어: Pascal, C, BASIC, PL/1, Ada 등

프로그래밍 언어의 발전과정

- ❖ C Language
 - ❖ 1970년대 초반 AT&T Bell Lab
 - ❖ UNIX 운영체제의 프로그램을 개발하기 위해 개발
 - ❖ 현재 대부분의 어플리케이션을 개발하는데 사용
 - ❖ 객체지향형의 C++ 과 Objective-C 등 많은 프로그래밍 언어로 발전
 - ❖ 성능이 좋아 다양한 분야에 사용
 - ❖ 어플리케이션 개발, 반도체 제어, 산업기계 제어 등

프로그래밍 언어의 발전과정

- ❖ Object Oriented Programming Language (객체지향 프로그래밍 언어)
 - ❖ 절차적 프로그래밍 언어 → 컴퓨터가 수행할 작업 순서대로 프로그램 작성
 - ❖ OOP
 - ❖ 컴퓨터 프로그램: 명령어(함수)의 집단 → 객체(Object)의 모임
 - ❖ 각각의 객체는 데이터를 포함하고 있고, 그 데이터를 처리하기 위한 방법(method) 들로 구성되어 있다 → Class
 - ❖ 해당 객체의 데이터 처리는 객체 내에서만 이루어 진다 → 전체 프로그램에 영향 X
 - ❖ 프로그램을 객체별로 분리하여 취급하므로 재사용과 수정이 용이
 - ❖ Objective-C, SmallTalk, C++, Java, C# 등이 대표적 언어

프로그래밍 언어의 발전과정

- ❖ C++
 - ❖ 1980년대 AT&T Bell Lab
 - ❖ C 언어를 확장하여 클래스, 객체, 이벤트 등의 객체지향 개념을 적용
- ❖ Java
 - ❖ 1980년대 후반 Sun Microsystems
 - ❖ 웹어플리케이션, 일반 어플리케이션, 소형 가전기기의 임베디드 OS에 사용
 - ❖ 네트워크 플랫폼을 겨냥 → 서버 어플리케이션용 랭귀지로 발전
 - ❖ 컴파일러 → (중간단계의) Byte Code 생성 → JavaVM에서 어플리케이션 실행 (Platform independent)

왜 “프로그래밍”을 배워야 하는가?

“We believe every child should have the opportunity to learn computer science, from primary school up to and including further education. We teach elementary physics to every child, not primarily to train physicists but because each of them lives in a world governed by physical systems. In the same way, every child should learn some computer science from an early age because they live in a world in which computation is ubiquitous. A crucial minority will go on to become the engineers and entrepreneurs who drive the digital economy, so there is a complementary economic motivation for transforming the curriculum.”

The Guardian, March 31, 2012

(<http://www.guardian.co.uk/education/2012/mar/31/manifesto-teaching-ict-education-minister>)

왜 “프로그래밍”을 배워야 하는가?

- ♦ 컴퓨터에 의해 움직이는 네트워크 사회에 살고 있기 때문
 - ♦ 컴퓨터가 만들어 낸 다양한 컨셉들에 대해 이해할 필요
- ♦ 문제 해결을 위한 새로운 사고가 도래: computational thinking
 - ♦ understanding difference between human and artificial intelligence
 - ♦ 해결하지 못하던 수많은 문제를 해결
- ♦ 이로 인해 새로 등장한 개념들
 - ♦ algorithms, cryptography, machine intelligence, search, computational X 등

왜 “프로그래밍”을 배워야 하는가?

- ◆ 사회과학자의 입장

- ◆ 디지털미디어의 급속한 확산 → 전통적인 매스미디어 모델에서 벗어나 개인 미디어나 소셜 미디어를 통한 정보의 생성, 전달, 소비가 활발하게 이루어짐.
- ◆ 소수에 의해 정보가 생산되는 매스미디어에 비해 개인 미디어나 소셜 미디어를 통해서 생산되는 정보의 양은 엄청나며, 이를 분석하기 위해서는 기존 연구방법을 뛰어 넘는 새로운 데이터 중심의 연구(data driven research) 패러다임이 요구되고 있음.
- ◆ 소셜 미디어와 그 안에서 데이터가 생성되고 공유되는 방식을 이해하기 위해 기술적인 접근이 필요.
- ◆ 생산되는 엄청난 양의 데이터를 수집, 분석하기 위해 관련 프로그래밍 기술의 습득이 필수.

왜 “프로그래밍”을 배워야 하는가?

- ◆ 사회과학자의 입장

- ◆ 반면, 컴퓨터 공학 분야에서 이루어지는 소셜 미디어 연구는 사회적 함의를 가지는 질문 보다는 대부분 네트워크 자체의 형태적, 구조적 특성에 대한 묘사적 연구가 주류.
- ◆ 전체 구조에 대한 콘텍스트는 제공하나 그 안에서 전달되는 콘텐츠에 대해서는 의미있는 방법론을 제시하지 못하고 있음.
- ◆ 사회과학자들이 프로그래밍과 관련 기술의 습득을 통해 소셜 미디어에서의 빅 데이터 분석에 있어 기존 연구전통과 연결되는 새로운 방법론을 만들어 내야할 시점.

4. 파이썬(Python)이란 무엇인가?

파이썬(Python)이란 이름의 언어

- ◆ 1991년 프로그래머 Guido van Rossum 이 발표한 언어
- ◆ 텍스트 관련 처리 능력이 매우 뛰어남
- ◆ 문법이 간단하고 이해하기 쉬움
- ◆ OOP 지향 언어
 - ◆ 개발효율, 유지, 보수, 재이용성, 신뢰성을 높일 수 있음
- ◆ 인터프리터 언어
 - ◆ 실행할 때 소스코드를 기계어로 번역하기 때문에 편리하게 코딩 가능
- ◆ 오픈소스! 멀티플랫폼(mac, windows, linux 등)
- ◆ 과학 계산과 관련한 라이브러리가 많아 최근 데이터 사이언스에 많이 사용

Python 2 or Python 3?

- ◆ Python 2
 - ◆ 라이브러리의 호환성이 좋다.
 - ◆ 기존의 코드(샘플코드)가 대부분 Python 2로 되어 있어 학습에 편리
 - ◆ 한글 인코딩이 복잡
- ◆ Python 3
 - ◆ 라이브러리의 호환성이 상대적으로 떨어진다
(하지만 최근엔 대부분의 주요 라이브러리가 Python 3 지원)
 - ◆ 기존의 코드와 호환이 안되는 경우가 있다
 - ◆ `print("문자출력")` vs `print "문자출력"`
 - ◆ `for i in range(10):` vs `for i in xrange(10):`
 - ◆ 한글 인코딩 문제가 해결

Python 2 or Python 3?

- ◆ Python 2
 - ◆ 라이브러리의 호환성이 좋다.
 - ◆ 기존의 코드(샘플코드)가 대부분 Python 2로 되어 있어 학습에 편리
 - ◆ 한글 인코딩이 복잡
- ◆ Python 3 ← 수업에서는 Python 3을 사용할 계획
 - ◆ 라이브러리의 호환성이 상대적으로 떨어진다
(하지만 최근엔 대부분의 주요 라이브러리가 Python 3 지원)
 - ◆ 기존의 코드와 호환이 안되는 경우가 있다
 - ◆ `print("문자출력")` vs `print "문자출력"`
 - ◆ `for i in range(10):` vs `for i in xrange(10):`
 - ◆ 한글 인코딩 문제가 해결

Python의 설치

- ◆ 설치 방법은 매우 다양해서 인터넷에서 검색해서 자신에게 맞는 방법을 선택할 것
 - ◆ 표준 설치 방법: <http://kybin.github.io/translateDivElementToPython3korean/installing-python.html>

4. 파이썬(Python)을 사용해 보자

CMD & Terminal

- Windows 나 Mac OS X 에는 키보드로 명령을 입력하기 위한 command line tool 이 있음
 - Windows: 시작 → 실행 → cmd 입력
 - Mac: Applications → Utilities → Terminal 실행
 - Python은 이들 창을 통해 실행 됨

```
C:\WINDOWS\system32\cmd.exe
ruby:
  bin:      C:/RailsInstaller/Ruby1.9.2/bin/ruby.exe
  version:   ruby 1.9.3p125 (2012-02-16) [i386-mingw32]

rails:
  bin:      C:/RailsInstaller/Ruby1.9.2/bin/rails.bat
  version:   Rails 3.2.1

ssh:
  public_key_location: C:\Documents and Settings\joonhwan\.ssh/id_rsa.pub
  public_key_contents: ssh-rsa AAAAB3NzaC1yc2EAAQEA0dL6+jXJEyu2d3woE2Rtc
whfy5TeEoJnH4hErxFh738m0kBLRzA6wE0o1cDr99jA3Unt6vzjDkYm+wQa44QpwRBoi5Zhu,jjUVCr68
YEgU+v/YqxkmbGdLuF7/dcsJNzEfCKjWFw8+s1pnTYOU7WguIAkb7TDWUFjaJ0ysJb81bDj91foXC
5tpwInwgtUBGWILPX63gsPYpJnU4t1pSu/qPSSsJ5i+NYyNrpPQRDiTUofOAQYP+JKRwzenNz4hZRJC0
Ui5lqo1HmL2FIBoRcBhDKW+ILbZFBkBCeGAiEEPOKHYu5i81Yd5/q4ct8pez0TyjtMFb12/Z60KnZmEm
w== Joonhwan Lee <joonhwan@gmail.com>

C:WSites>
C:WSites>
C:WSites>
C:WSites>
```

명령프롬프트

```
var — bash — 100x30
bash
== LICENSE:
Copyright (c) 2009-2011 Wayne E. Seguin
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
monoair:var joonhwan$ rvm 1.9.3
monoair:var joonhwan$ rvm list
rvm rubies
  jruby-1.6.7.2 [ x86_64 ] ruby:
  ruby-1.9.2-p318 [ x86_64 ] bin:      C:/RailsInstaller/Ruby1.9.2/bin/ruby.exe
  ruby-1.9.2-p328 [ x86_64 ] version:   ruby 1.9.3p125 (2012-02-16) [i386-mingw32]
*=* ruby-1.9.3-p194 [ x86_64 ]
                               Window  rails:
                               # => - current      bin:      C:/RailsInstaller/Ruby1.9.2/bin/rails.bat
                               # *= - current && default  bin:      C:/RailsInstaller/Ruby1.9.2/bin/rails.bat
                               # * - default      version:   Rails 3.2.1
monoair:var joonhwan$ ssh:
```

Terminal

CMD & Terminal

♦ 간단한 명령어 리스트

	CMD	Terminal	example
디렉토리 이동	cd	cd	cd .. cd Documents cd ~/Desktop
디렉토리 내 파일 리스트 보기	dir	ls	-
디렉토리 생성/삭제	mkdir rmdir	mkdir rmdir	mkdir temp rmdir temp
디렉토리 이름 변경	rename	mv	rename temp1 temp2 mv temp1 temp2
파일 카피	copy	cp	copy temp1 temp2 cp temp1 temp2

Text Editors

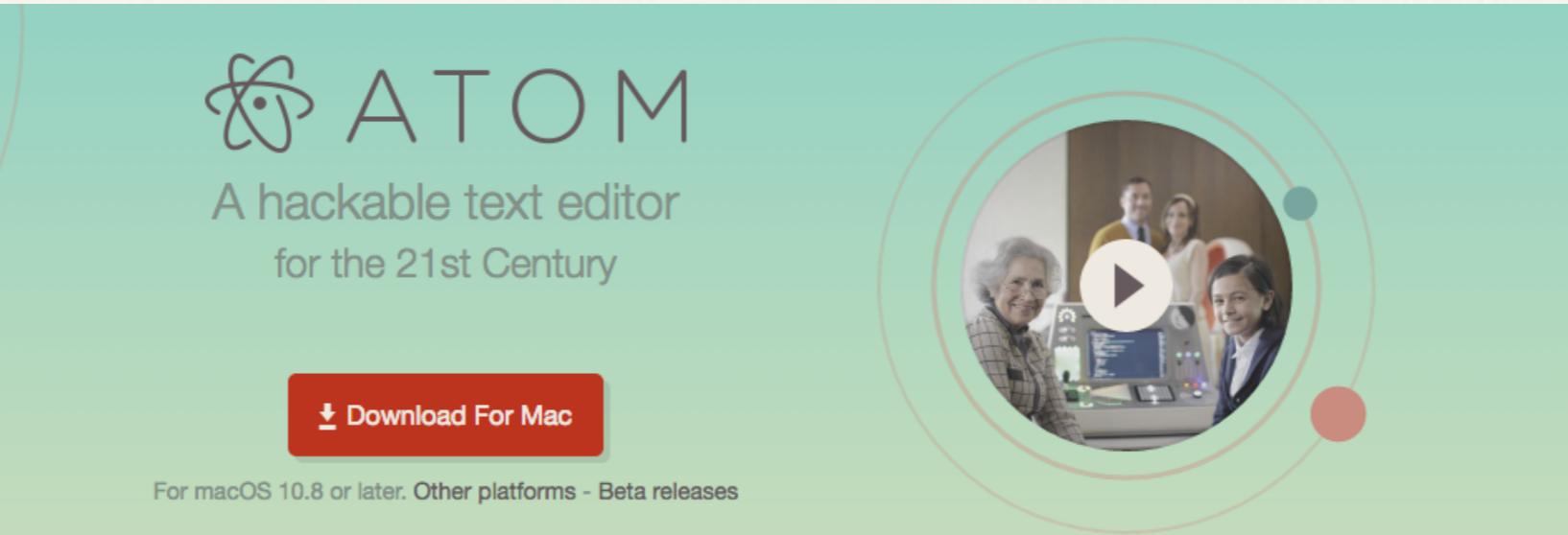
- ◆ 프로그램의 소스코드를 작성하기 위해 자신에 맞는 텍스트 에디터를 선택해야 함.
- ◆ 텍스트 에디터 = 일종의 워드프로세서
 - ◆ 워드프로세서처럼 텍스트의 속성이 저장되지 않는다.
(예: 볼드 x, 칼라 x, 밑줄 x, 글자크기 x)
- ◆ 그러나 프로그래밍 전용 텍스트 에디터는 키워드, 변수 등을 다른 색으로 표시하여 손쉽게 코드를 이해할 수 있음
 - ◆ 라인넘버 제공으로 특정 코드의 위치를 표시 → 에러 발생시 유용하게 쓰임
 - ◆ Mac: TextMate, Smultron 등
PC: SciTE, notepad++ 등



A screenshot of a Ruby code editor window titled "crawl_comments.rb". The code uses syntax highlighting to distinguish between different parts of the program. Line numbers are visible on the left side of the editor. The code itself is a script for crawling comments from a web page, using regular expressions and XPath queries to extract specific data fields like article titles and comment counts. The editor interface includes standard window controls (minimize, maximize, close) at the top and status information at the bottom (Line: 160, Column: 29, Ruby, Soft Tabs: 2, Symbol).

```
72 # - slice(.,.*\\/) : returns ",28218468,134057910"
73 # - slice(1..-1) : remove the first character -> returns
74 "# - chop : remove the last character -> returns "28218468,134057910"
75 #
76 def getArticleCommentID(doc)
77   ids = Array.new
78   doc.xpath('//dd[@class="action"]/a@onclick').each do |method_span|
79     ids << method_span.text.slice(.,.*\\/).slice(1..-1).chop
80   end
81   return ids
82 end
83
84 # get the number of a comment's child comment
85 def getChildCount(doc)
86   counts = Array.new
87   doc.xpath('//a[@class="reply_count"]').each do |method_span|
88     counts << method_span.content.gsub!("답글", "").strip
89   end
90   return counts
91 end
92
93 # clean up json file and parse
94 def parseJSON(content)
95   orig_title = Array.new
96   clen_title = Array.new
97   orig_comment = Array.new
98   clen_comment = Array.new
99
100  puts "Cleaning up JSON..."
101  @logfile << "Cleaning up JSON...\n"
102
103  # scan cleannning area
104  orig_title = content.scan(/articleSubject.+?articleUrl/)
105  orig_comment = content.scan(/commentContent.+?rheaAdminTool/)
106
107  # clean up title
108  orig_title.each do |method_span|
109    clen_title << method_span.gsub('":', ' "Q@T").gsub('","', ' "Q@T").gsub('","', ' "Q@T")
110  end
```

Atom



The screenshot shows the official Atom website. At the top left is the Atom logo (a stylized atom symbol) and the word "ATOM". Below it is the tagline "A hackable text editor for the 21st Century". A red button labeled "Download For Mac" is present. To the right is a video player showing three people (two men and one woman) smiling, with a play button icon in the center. Below the video is a screenshot of the Atom code editor interface, displaying a file named "atom.coffee" with some CoffeeScript code. The sidebar on the left shows a file tree with directories like "build", "docs", "dot-atom", etc., and files like ".gitignore" and ".gitattributes". The bottom right corner of the screenshot contains the "hci+d lab." logo.

For macOS 10.8 or later. Other platforms - Beta releases

hcidi lab.

atom.coffee

```
18 # Essential: Atom global for dealing with packages, themes, menus, and the window system.
19 #
20 # An instance of this class is always available as the `atom` global.
21 module.exports =
22 class Atom extends Model
23   @version: 1 # Increment this when the serialization format changes
24   #
25   # Load or create the Atom environment in the given mode.
26   #
27   # Returns an Atom instance, fully initialized.
28   @loadOrCreate: (mode) ->
29     startTime = Date.now()
30     atom = @deserialize(@loadState(mode)) ? new this({mode, @version})
31     atom.deserializeTimings.atom = Date.now() - startTime
32
33
```

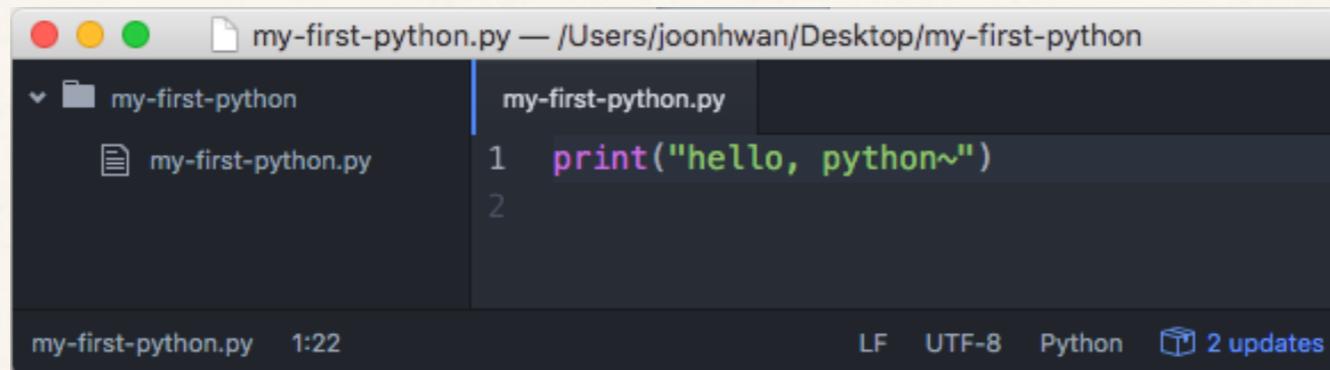
src/atom.coffee* 31,17

UTF-8 CoffeeScript master

<https://atom.io/>

My First Python Program!

- ❖ text editor 실행 후 다음과 같이 입력

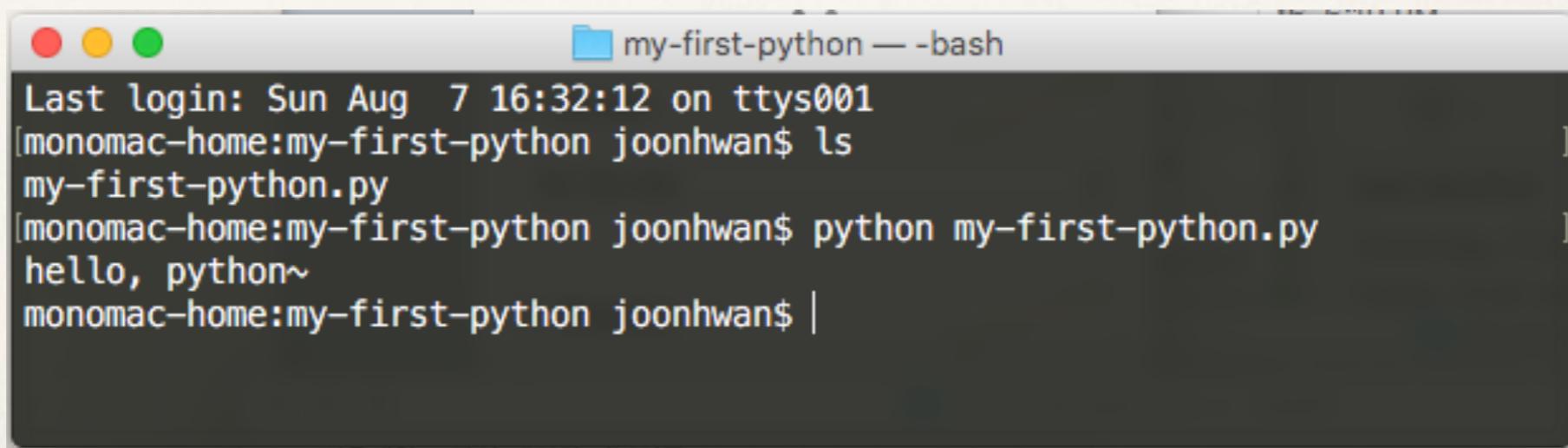


A screenshot of a Mac OS X TextEdit window. The title bar says "my-first-python.py — /Users/joonhwan/Desktop/my-first-python". The left pane shows a folder "my-first-python" containing a file "my-first-python.py". The right pane displays the code:

```
1 print("hello, python~")  
2
```

The status bar at the bottom shows "my-first-python.py 1:22" and "LF UTF-8 Python 2 updates".

- ❖ 해당 파일이 저장된 디렉토리로 이동 (cd <디렉토리>)
- ❖ python <파일명> 으로 python 프로그램 실행

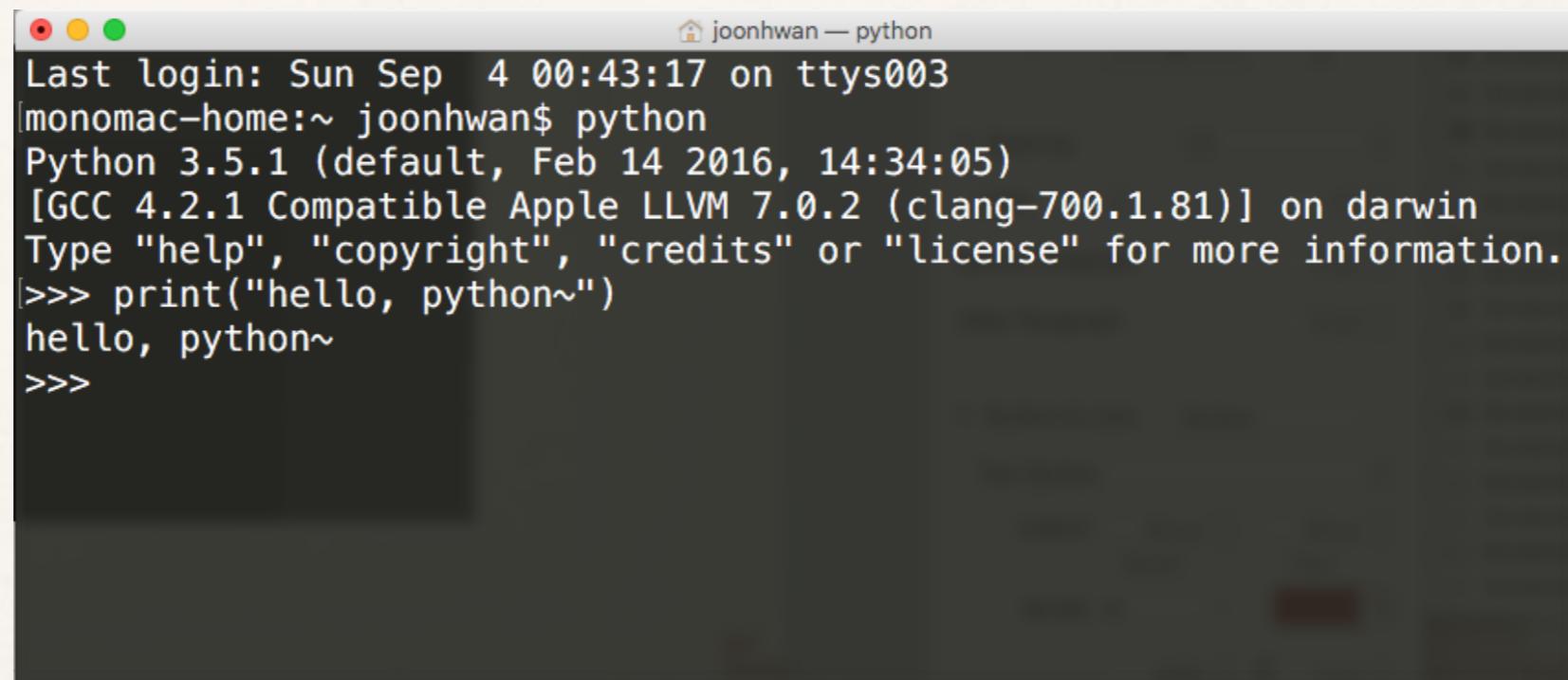


A screenshot of a Mac OS X terminal window. The title bar says "my-first-python — -bash". The terminal output is:

```
Last login: Sun Aug  7 16:32:12 on ttys001  
[monomac-home:my-first-python joonhwan$ ls  
my-first-python.py  
[monomac-home:my-first-python joonhwan$ python my-first-python.py  
hello, python~  
monomac-home:my-first-python joonhwan$ |
```

My First Python Program!

- ❖ 파이썬을 실행하는 또 다른 방법
 - ❖ interactive shell mode

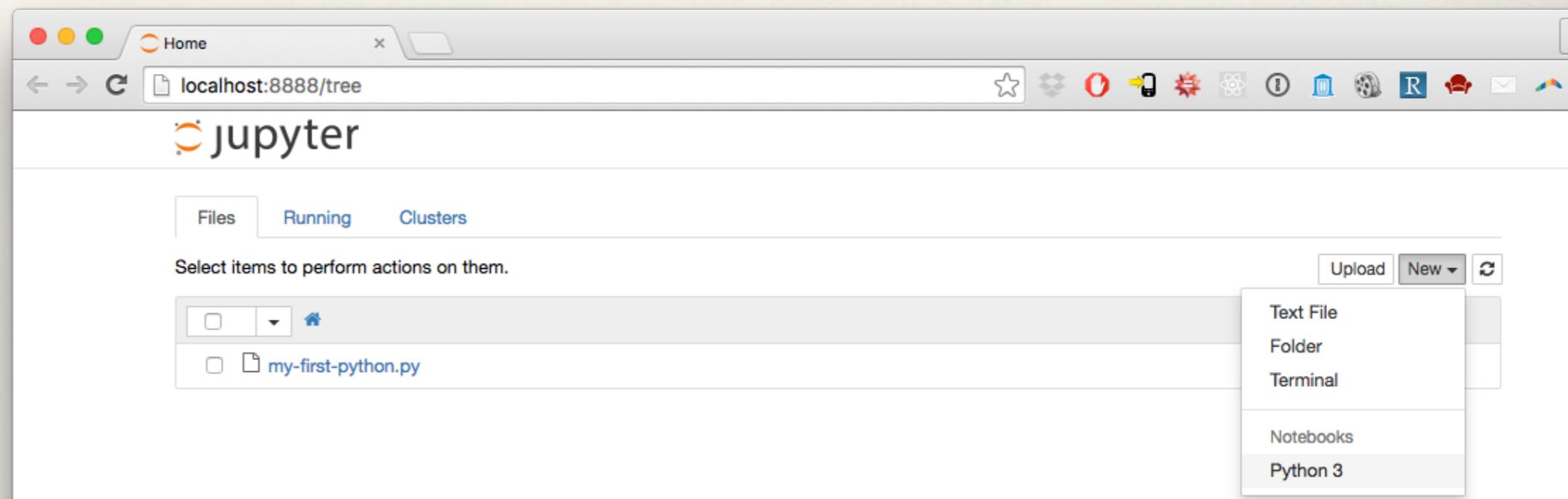


A screenshot of a Mac OS X terminal window titled "joonhwan — python". The window shows the following text:

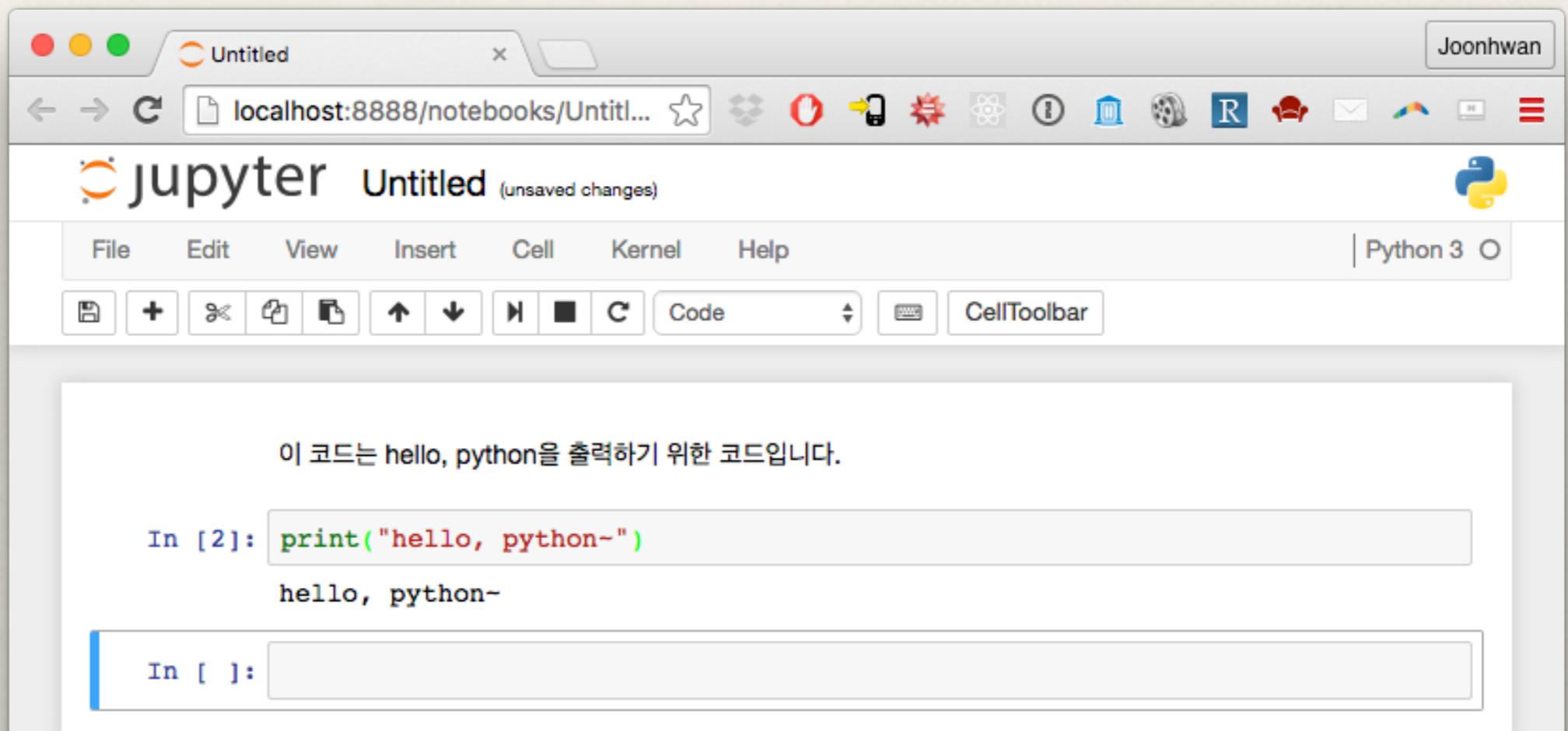
```
Last login: Sun Sep  4 00:43:17 on ttys003
monomac-home:~ joonhwan$ python
Python 3.5.1 (default, Feb 14 2016, 14:34:05)
[GCC 4.2.1 Compatible Apple LLVM 7.0.2 (clang-700.1.81)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("hello, python~")
hello, python~
>>>
```

My First Python Program!

- ❖ 파이썬을 실행하는 근사한 방법 - Jupyter Notebook
 - ❖ 설치: pip install jupyter
 - ❖ 실행: (터미널에서) jupyter notebook



Jupyter Notebook



유용한 자료

- ♦ 참고 교재
 - ♦ 점프 투 파이썬: <https://wikidocs.net/book/1>
 - ♦ CodeCademy: <https://www.codecademy.com/ko/tracks/python-ko>
 - ♦ Coursera: <https://www.coursera.org/learn/interactive-python-1>
 - ♦ Learn Python: <http://www.learnpython.org/>
- ♦ reference
 - ♦ <https://docs.python.org/3/>

Codecademy

The screenshot shows the Codecademy Python editor interface. The top navigation bar includes the Codecademy logo, a 'Python' dropdown, and user icons. The left sidebar displays a 'Conditions & Control Flow' track and a 'Go With the Flow' lesson. The main area shows a code editor with the file 'script.py' containing the following Python code:

```
1 def clinic():
2     print "You've just entered the clinic!"
3     print "Do you take the door on the left or the right?"
4     answer = raw_input("Type left or right and hit 'Enter'").lower()
5     if answer == "left" or answer == "l":
6         print "This is the Verbal Abuse Room, you heap of parrot droppings!"
7     elif answer == "right" or answer == "r":
8         print "Of course this is the Argument Room, I've told you that already!"
9     else:
10        print "You didn't pick left or right! Try again."
11    clinic()
12
13 clinic()
```

The editor has a dark theme with syntax highlighting. Below the code editor are 'Instructions' and 'Check out the code in the editor.' sections. At the bottom are navigation links for 'Q&A Forum' and 'Glossary', and buttons for 'Save & Submit Code' and 'Reset Code'.

<https://www.codecademy.com/learn/python>

<https://www.codecademy.com/ko/tracks/python-ko>

TODOs

- ◆ 파이썬 설치
 - ◆ 자신의 컴퓨터에 파이썬 설치
 - ◆ 강의실 컴퓨터는 실습이 들어가기 전에 설치를 해 놓을 예정
- ◆ 에디터 선택
 - ◆ 맘에 드는 에디터 선택
 - ◆ Command Shell 명령어에 익숙해지기
 - ◆ CMD, Terminal에서 쓰는 도스/유닉스 기본 명령어 찾아 공부하기

Assignment 1

- ❖ CodeCademy에서 Python 강좌 수강
 - ❖ 강좌 수강 증명 제출 (due: 9/30)
 - ❖ 과제 면제 요청 가능 (파이썬 유 경험자)

The screenshot shows the landing page for the Python course on Codecademy. At the top, there's a navigation bar with the Codecademy logo, an 'Upgrade to Pro' button, a 'Catalog' link, a notification bell icon, and a user profile icon. Below the navigation, there's a back-link 'Back to Courses' and the course title 'Python'. A brief description follows: 'Learn to program in Python, a powerful language used by sites like YouTube and Dropbox.' To the right is a large circular progress indicator with a red border. Inside the circle is a pink apple icon with a green stem and leaf, labeled 'Lessons' above a progress bar, and '28%' below it. Below the circle is a teal 'CONTINUE' button. At the bottom of the page, there's a call-to-action: 'Want more practice and review? Upgrade for the complete experience.', followed by statistics: '8 Projects', '9 Quizzes', and '1 Final Project', and a link 'Get Instant Access »'.

Questions?
