

Introduction

튜터 오시영



by ohahohah.dev

Intro | 강의를 어떻게 활용할까?



Photo by Leosprspctive on Unsplash

차근차근 업데이트

- 가장 중요한 핵심은 무엇일까?
- Ver 0.0.0.1 차근차근 업데이트 하기
- 구현하는 과정에서 새롭게 얻은 정보 사용해 버전 업
- 스스로 할 수 있는 힘 기르기



by ohahohah.dev

Intro | 강의를 어떻게 활용할까?

1. 개념 이해
2. 작은 예제 해보기
3. 프로젝트 만들기
4. 확장하기
5. 내가 만들고 싶은 프로젝트에 적용하기!



by ohahohah.dev

Intro | 프로그래밍 잘 배우기

프로그래밍 '잘하는 방법' 배우기



by ohahohah.dev

1. 레고 조립하기

큰 모듈을 작은 모듈로. 모듈을 조립하기

2. 에러 마음껏 내보기

에러를 고치는 것은 프로그래밍의 일부! 고치면 됩니다!

3. 질문하고 답변하기

바보 같은 질문은 없습니다. 나의 지식을 **우리** 지식으로!

4. 내 지식으로 만들기

내가 하고 싶은 프로젝트에 적용하기



1. 레고조립 처럼 프로그래밍 하기

- elements 요소가 모여서 화면이 되고
- 함수가 모여서 프로그램이 되고
- 내가 원하는 부분에 가져다 쓰기
- 입력값을 조금만 바꿔보기
- 만들고 싶은 걸 부분 부분으로 나누어보기



Photo by [Phil Hearing](#) on [Unsplash](#)



by ohahohah.dev

2. 에러 마음껏 내보기



Fritz Grobe and Stephen Voltz perform the "Diet Coke and Mentos Experiment"

2. 에러 마음껏 내보기

그러나 컴퓨터 프로그래밍을 배울 때 처음부터 잘하지는 못한다. 능숙한 프로그래머가 되려면 능숙하게 버그를 찾아내고 수정하는 법을 익혀야 한다.

프로그래밍을 하면서 던져야 할 질문은 이것이 맞느냐 틀리느냐가 아니라 **버그를 수정할 수 있느냐 없느냐**다. 이런 식의 지적 결과물을 바라보는 방식이 지식과 지식 습득을 대하는 좀 더 큰 문화에까지 보편화된다면 **우리 모두 '틀리는 것' 을 덜 두려워하게 될 것이다.**

2. 에러 마음껏 내보기

But when you learn to program a computer you almost never get it right the first time.

Learning to be a master programmer is learning to become highly skilled at isolating and correcting "bugs," the parts that keep the program from working. **The question to ask about the program is not whether it is right or wrong, but if it is fixable.** If this way of looking at intellectual products were generalized to how the larger culture thinks about knowledge and its acquisition, we all might be **less intimidated by our fears of "being wrong.**

2. 에러 마음껏 내보기

- 프로그램 만들기는 설계(Design) - 구현(Coding) - 테스팅(Testing) - 디버깅(버그 찾기, Debugging) 반복
- 디버깅은 프로그래밍의 일부
- 배워야하는 것! 에러를 발견하고 고치는 과정 Debugging
 - 같은 에러 10번 내기 → 해결방법 익히기!
 - 에러 해결? 코드 오타 고치기, 파일 구조 맞추기



3. 질문하고 답변하기

1. 문제를 적기

- A. 목적 : 하려고 했던 작업, 궁금한 것 (예. `print("hello")` 가 실행이 안되네요)
- B. 에러메시지 전체
- C. 에러나는 코드 (다른 사람에게 공유할 때는 전체 / Github 링크)

2. 구글에 검색하기

- 에러메시지를 그대로 복사해서 검색하기
- (What is) + 무엇 : http 403
- 기술명 + 키워드 : flask cors

3. 질문하기

4. 정리하기



by ohahohah.dev

3. 질문하고 답변하기

- 구글링 검색 팁

<https://news.hada.io/topic?id=4140>

- 빠르고 정확하게 답변을 받을 수 있는 질문하는 법

<http://bit.ly/how-to-ask-programming>

4. 내 지식으로 만들기

- 안다는 착각에서 벗어나자!
- 각 강의 끝나고 백지에 키워드 복습
- 다 알아야한다 🧙‍♀️ 뭘 모르는지 파악한다 👍
- 모르는 건 찾아서 붙여넣기 - 레고처럼!



Photo by [Phil Hearing](#) on [Unsplash](#)

강의 | 프로그래밍 잘 배우기



Fritz Grobe and Stephen Voltz perform the "Diet Coke and Mentos Experiment"

목표 | 클라우드에서 배포



Photo by Nick Seagrave on Unsplash

어떻게 만들어볼까?

1. 문제 정의하기
2. 문제를 해결할 수 있을 정도로 작게 나누기 (Decompositon)
3. '2'를 설계(Design) 하고
4. 구현 (coding) 하고
5. Test 하고
6. 선보이자!



by ohahohah.dev

Programming

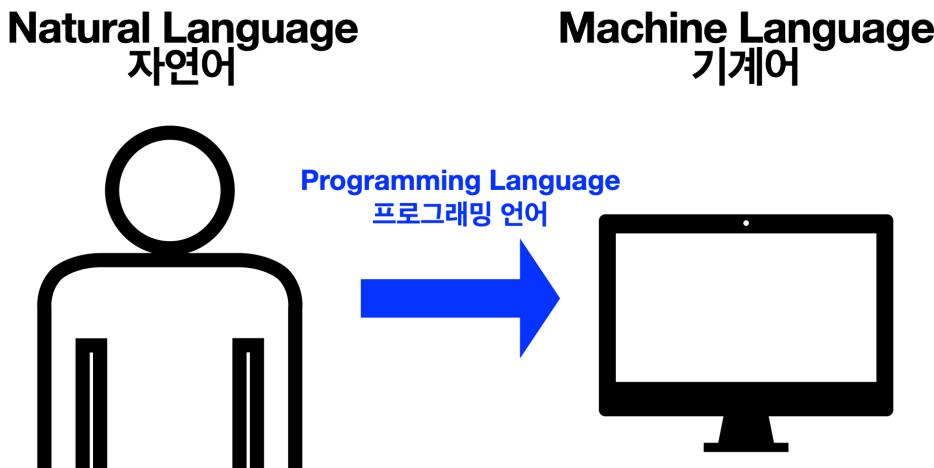
- Program + ing
- Program? 특정 목적을 수행하는 명령어의 집합
- 프로그램은 특정 목적을 가지고 있음. 문제를 해결하기 위해 만들어져 있음!



by ohahohah.dev

Programming

- '인간의 말(Natural Language)'을 기계가 알아들을 수 있게(Machine Language)
- 인간의 의도(문제)를 프로그래밍 언어(Programming Language)의 형식으로 나타내어
- 컴퓨터에게 일련의 명령을 내리는 것.



by ohahohah.dev

Programming

- 하나 이상의 추상 알고리즘을
특정한 프로그래밍 언어를 이용해 구체적인 컴퓨터 프로그램으로 구현하는 기술*
- 문제를 해결하기 위해 알고리즘과 프로그래밍 언어를 사용해 프로그램을 만드는 것

* 컴퓨터 프로그래밍. Wikipedia 한국. Retrieved 17 October 2020. https://ko.wikipedia.org/wiki/%EC%BB%B4%ED%93%A8%ED%84%B0_%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D