

3 Websites for Science Simulations for Teachers



Ansys SimAI: 빠른 시뮬레이션 예측
을 위한 Non-parametric AI 플랫폼
토크아이티(Talk IT)
611 回視聴・6日前に配信済み



CNG TV, CAE 트렌드와 AI 시뮬레이션
기술 전망(10.16) - 이종수/이승훈/...
캐드앤그래픽스
392 回視聴・6か月前



ANSYS Tutorial | Grid Independence Test In ANSYS Fluent Using...

Ansys-Tutor

6.8万回視聴・4年前

이수안컴퓨터연구소

@user-ss5no9xw6e · チャンネル登録者数 3.46万人 · 241 本の動画
이수안 교수와 함께 따라하며 배우는 컴퓨터 강의/실습 ...さらに表示
suanlab.com、他 3 件のリンク

登録済み

ホーム 動画 再生リスト コミュニティ 検索

新しい順

人気の動画

古い順

빅데이터 분석 제대로 끝내기

23. 통계 분석 기법

1:54:51

23 통계 분석 기법 - 빅데이터 분석 제대로 끝내기

227回視聴 · 1日前

빅데이터 분석 제대로 끝내기

22. 분석 모형 설계

58:26

22 분석 모형 설계 - 빅데이터 분석 제대로 끝내기

152回視聴 · 6日前

빅데이터 분석 제대로 끝내기

21. 추정과 검정

1:08:20

21 추정과 검정 - 빅데이터 분석 제대로 끝내기

209回視聴 · 13日前

빅데이터 분석 제대로 끝내기

20. 확률 분포

1:58:02

20 확률 분포 - 빅데이터 분석 제대로 끝내기

202回視聴 · 13日前

빅데이터 분석 제대로 끝내기

19. 표본 추출

1:54:51

빅데이터 분석 제대로 끝내기

18. 빅데이터 처리 기술

1:54:51

빅데이터 분석 제대로 끝내기

17. 데이터 변환

1:54:51

빅데이터 분석 제대로 끝내기

16. 데이터 축소

1:54:51

suanlab.com 홈페이지에 들어가서 특강에
게임 리소스 버튼이 있습니다

Not secure suanlab.com/lecture.html

App Bookmarks Path Course Computer Project Startup Investment Study Download News English Economy Favorite Other bookmarks

 suanlab

수연 영상 특강 강의 움짤 노트북 블로그

SPECIAL LECTURE

All 2018 2017

파이썬으로 슈팅게임 만들기

2018

파이썬과 pygame 라이브러리를 이용하여 쉽고 재미있는 슈팅게임을 만들 어보자.

• Technologies: Python, pygame
• Date: 2018년 10월 18일
• Location: 한림대학교

• 강의자료 • 게임 리소스

이수안

suanlab.com 홈페이지에 들어가서 특강에
게임 리소스 버튼이 있습니다

함께 따라하며 만드는 엑셀 VBA

엑셀로 네이버 다긁어오기

엑셀에서 네이버를 다긁어온다고? Naver

엑셀 VBA 스크래핑 NAV 34:18

엑셀 네이버 다긁어오기 Scraping Naver with Excel

Scraping Naver with Excel VBA

2万回視聴 · 5年前

쉽고 재미있게 만드는 파이썬 포토샵

파이썬으로 포토샵을 만든다고? 30분만 따라해보자!

파이썬 포토샵 만들기 Creating a PhotoShop with Python (한글자막)

32:01

파이썬 포토샵 만들기 Creating a PhotoShop with Python (한글자막)

7495回視聴 · 5年前

누구나 쉽게 따라하며 만드는 C언어 MIDI 피아노

C 언어로 전자악기 피아노를 만들 수 있다고??

Creating MIDI Piano Keyboard with C langauge

46:24

C 언어 MIDI 피아노 키보드 만들기 Creating a MIDI Piano Keyboard with C langauge (한글자막)

8266回視聴 · 5年前

쉽고 재미있게 만드는 파이썬 슈팅 게임

게임 한번 만들어보고 싶었지? 40분만 따라하면 게임 하나 만든다!

파이썬 비행기 슈팅 게임 만들기 Creating a Shooting Game with Python (한글자막)

41:00

파이썬 비행기 슈팅 게임 만들기 Creating a Shooting Game with Python (한글자막)

14万回視聴 · 5年前

누구나 쉽게 따라하는 컴퓨터 조립

너 컴퓨터 조립 할 줄 몰리? 헐!

자기 컴퓨터는 스스로 조립하자!

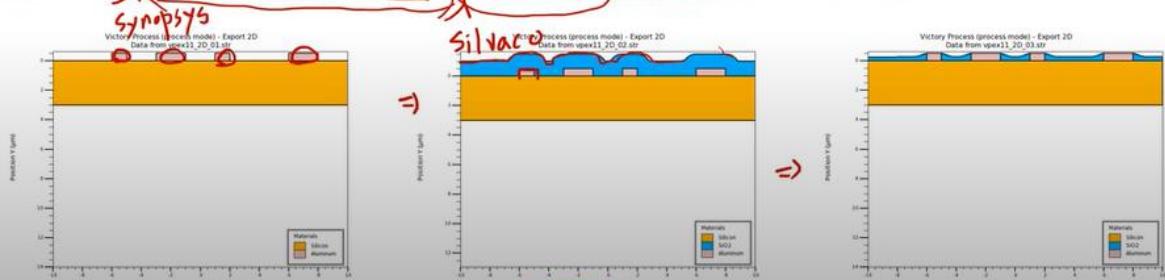
38:27

본체만 300만원짜리 조립 컴퓨터 만들기 (한글자막)

6212回視聴 · 5年前

TCAD 프로그램

- Device simulator: Pisces, Medici, Dessis, Sentaurus Device, Atlas(Pisces 상업 버전) 등
- Process simulator: Suprem3(1D), Suprem4(2D), **Suprem4 GS (2D+GaAs)**, Prophet, Sentaurus Process, Athena(Suprem4 상업 버전) 등



* Sivaco CAD Damp 8:29/35:35y Process



TCAD SUPREM 반도체공정 실습/시뮬레이션 - 시뮬레이션 개요



세미파이터

チャンネル登録者数 65人

チャンネル登録

19



共有

オフライン





Smart Design Lab @ KAIST

@smartdesignlabkaist8156 · チャンネル登録者数 1760人 · 76 本の動画

KAIST スマート 설계 연구실 (강남우 교수) Youtube 입니다. 강의자료와 실습코드는 [...さらに表示](#)

[チャンネル登録](#)

ホーム 動画 再生リスト コミュニティ 検索

연구실 소개

KAIST 스마트설계연구실 소개영상입니다.



KAIST 조천식모빌리티대학원 스마트설계연구실(강남우 교수) 소개 영상

Smart Design Lab @ KAIST · 2490 回視聴 · 2年前

KAIST 조천식모빌리티대학원 스마트설계연구실(강남우 교수)의 소개 영상입니다. www.smartdesignlab.org

4:51

연구 소개 ►すべて再生

KAIST 스마트설계연구실의 Deep Generative Design 연구소개 영상입니다.



⑧ Smart Design Lab | KAIST | Namwoo Kang | 1.6.2018 | 2019.1.8 | ⑨ Smart Design Lab | KAIST | Namwoo Kang | data_3000.p | Google Drive | ⑩ Smart Design Lab | KAIST | Namwoo Kang | +

딥러닝 실습 Ch3.2 Autoencoder & Anomaly Detection

SMART DESIGN LAB @ KAIST

HOME PEOPLE PUBLICATIONS RESEARCH TEACHING NEWS PHOTOS CONTACT

Lnb_GAN ([PDF](#))
Ch7. Mechanical Design + AI ([PDF](#))

실습 코드

실습 준비하기

- 1) 본 강습회는 실습을 위해 구글 Colab을 사용합니다. 원하시는 경우 개인컴퓨터에 Jupyter Notebook과 TensorFlow를 설치하셔도 됩니다. 아래 사용법을 참고해주세요
[TensorFlow 설치+Colab 사용방법+Python 기초문법](#)
- 2) 구조설계데이터인 Bracket data를 예시로 사용합니다. 아래 두가지 파일을 다운받아 주세요.
[data_3000](#), [data_1000](#)

Autoencoder (AE) & Anomaly Detection

- 1) Colab
--Bracket: [Autoencoder, Anomaly Detection](#)
 - 2) Jupyter Notebook
--Bracket: [Autoencoder, Anomaly Detection](#)
- #참고: 다양한 학습 방법 비교 (Bracket)
[ModelFit / GradientTape](#)

Variational AutoEncoder (VAE)

- 1) Colab
--Bracket: [VAE](#)
- 2) Jupyter Notebook
--Bracket: [VAE](#)

#참고: 다양한 학습 방법 비교 (Bracket)

[ModelFit / GradientTape](#)

6:30 / 35:08

^

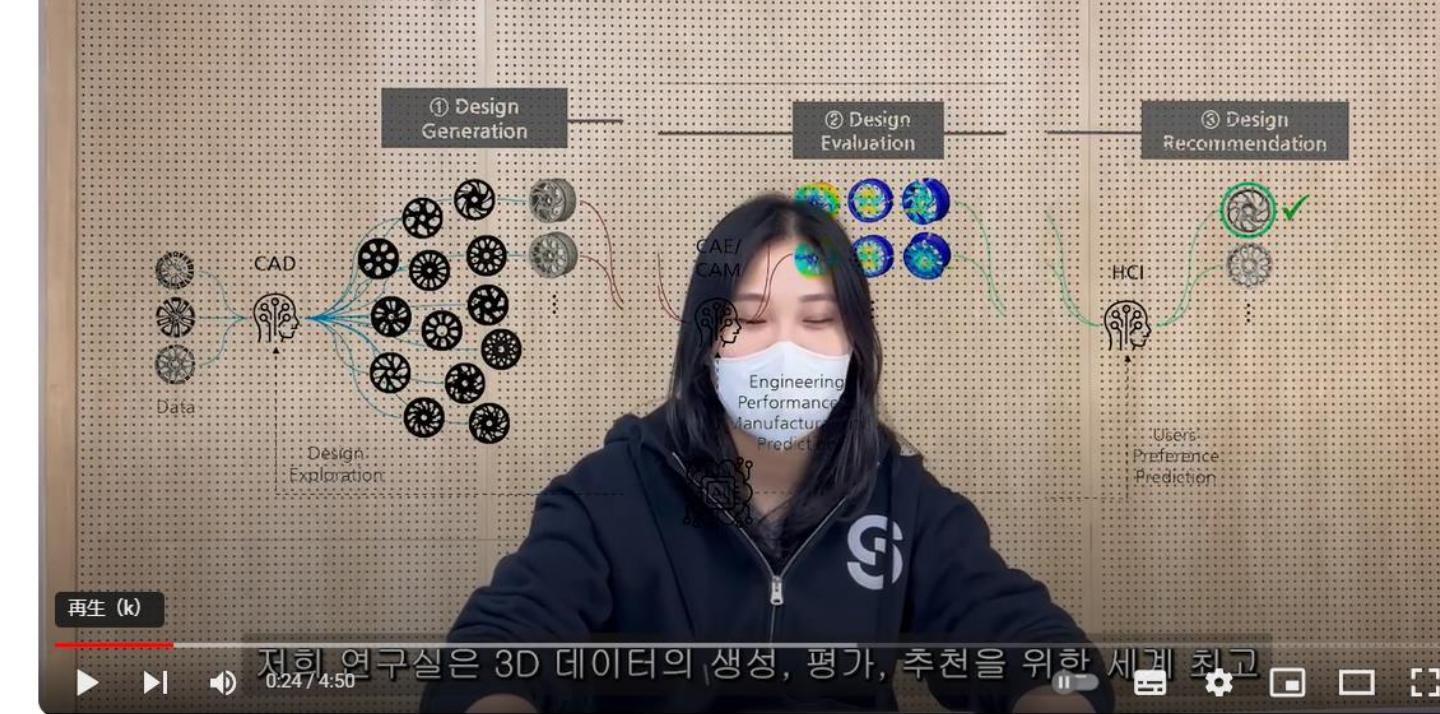
smartdesignlab.org

<http://www.smartdesignlab.org> ▾

[Smart Design Lab | KAIST | Namwoo Kang\(강남우 ...\)](#)

ウェブ WELCOME TO SMART DESIGN LAB Graduate School of Mobility | KAIST (2021-Present)\u000B Mechanical Systems Engineering | Sookmyung Women's University ...

タグ: KAIST Design United States Physics



KAIST 조천식모빌리티대학원 스마트설계연구실(강남우 교수) 소개 영상

 Smart Design Lab @ KAIST
チャンネル登録者数 1760人

チャンネル登録

19



共有

オフライン

0:26 / 4:50 수준의 generative design을 선도하고 있는 연구실인데요

첫번째는 3D generative design을 위해서

두 번째는 다양한 엔지니어링 설계 문제에 대해서 저희가 갖고 있는

AI 솔루션을 제공해 줄 예정입니다.

생성형 AI 기반 제품 설계 및 성능 예측 혁신



강남우 대표 (나니아랩스)

제조업 분야에서의 다양한 딥러닝 활용 방안에 대해 소개한다. 또한 생성형 AI를 활용한 3D 데이터 설계부터 성능 예측과 최적 설계까지 포괄적인 사례에 대해 공유하고자 한다.

▶ ▶ ◀ 0:00 / 34:07



[CAE 컨퍼런스 2023] 생성형 AI 기반 제품 설계 및 성능 예측 혁신



캐드앤큠그레피스
チャンネル登録者数 2740人

チャンネル登録

6



共有

オフライン

깜짝놀랄.. 딥러닝 강의 자료가 라이브로 올라가 있다.

감사

The screenshot shows two browser windows side-by-side. The left window is a Jupyter Notebook interface with several code cells. The right window is a Google Colab environment. Both windows display identical Python code for file upload and processing. The code includes imports for TensorFlow, Keras, and scikit-learn, along with logic to handle uploaded files and save them as zip archives.

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import Input, layers, Model
from tensorflow.keras.losses import mse, binary_crossentropy
from tensorflow.keras.utils import plot_model

print(tf.__version__)
2.1.0

데이터 불러오기

In [ ]:
from google.colab import files
uploaded = files.upload()

# 코드가 돌아가면 파일선택 버튼을 눌러서 upload할 파일 선택
for fn in uploaded.keys():
    print('User uploaded file "{}" with length {} bytes'.format(
        name=fn, length=len(uploaded[fn])))

In [ ]:
! mkdir data_3000
! unzip data_3000.zip -d ./data_3000

In [ ]:
# 파일선택 버튼을 찾아 복사해서 붙여넣어주세요
src = './data_3000/'

# 이미지 읽기 및 흑백
def img_read(src_file):
    img = cv.imread(src_file, cv.COLOR_BGR2GRAY)
    plt.imshow(img)
    plt.xticks([]) # x축 눈금
    plt.yticks([]) # y축 눈금
    plt.show()
    return img

# 이미지 읽기
def img_read(src_file):
    img = cv.imread(src_file, cv.COLOR_BGR2GRAY)
    return img

# src 경로에 있는 파일 명을 저장합니다.
files = os.listdir(src)

# 경로와 파일명을 원본으로 넣어 확인하고
# 데이터를 255로 나눠서 0-1사이로 정규화 하여 X 리스트에 넣습니다.

X,Y = [], []
for file in files:
```

```
+ Code + Text
[3] 2 import matplotlib.pyplot as plt
3 import os
4 import cv2 as cv
5 import scipy.io
6 from sklearn.model_selection import train_test_split
7
8 import tensorflow as tf
9 from tensorflow import keras
10 from tensorflow.keras import Input, layers, Model
11 from tensorflow.keras.losses import mse, binary_crossentropy
12 from tensorflow.keras.utils import plot_model
13
14 print(tf.__version__)

2.5.0

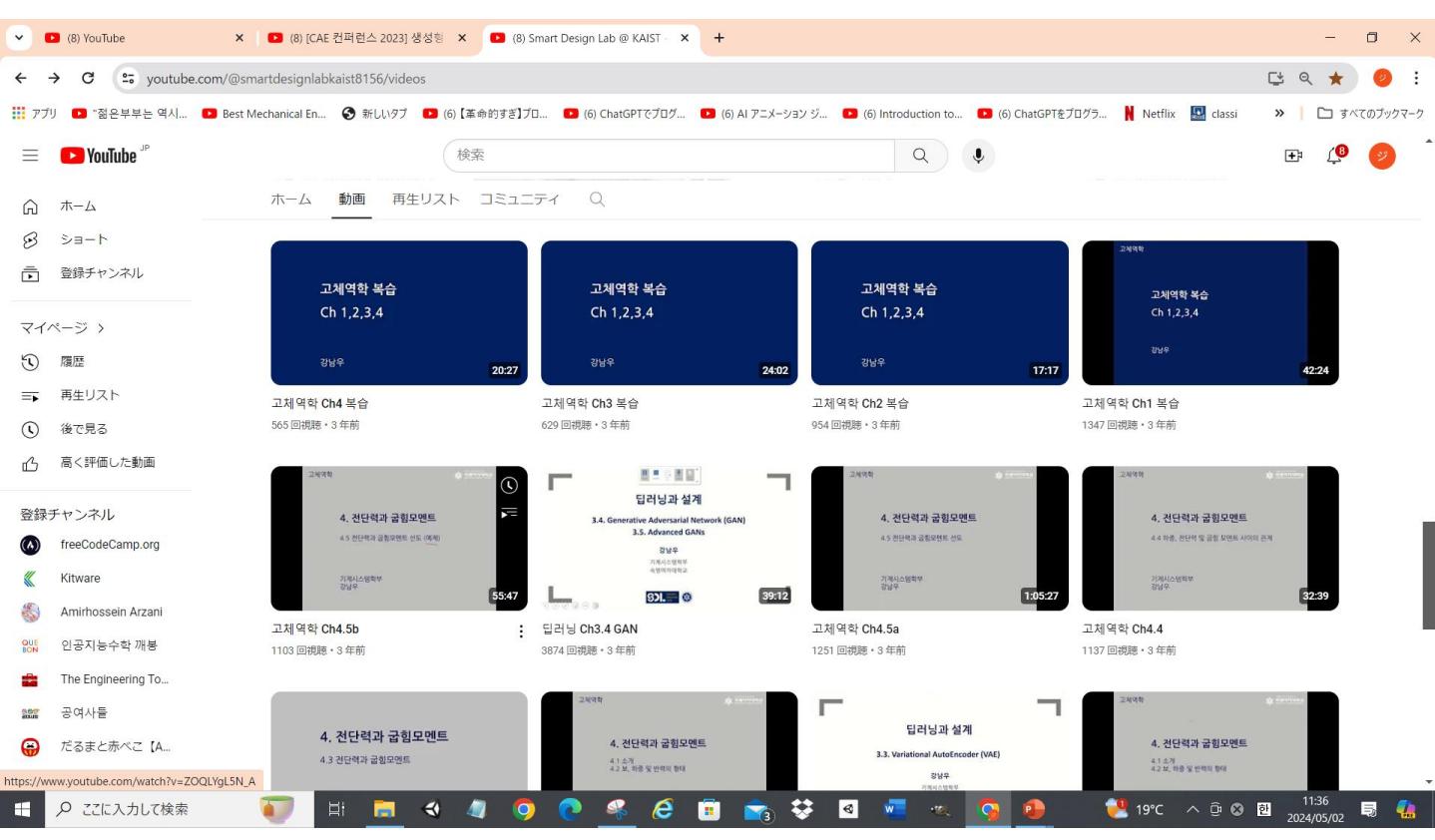
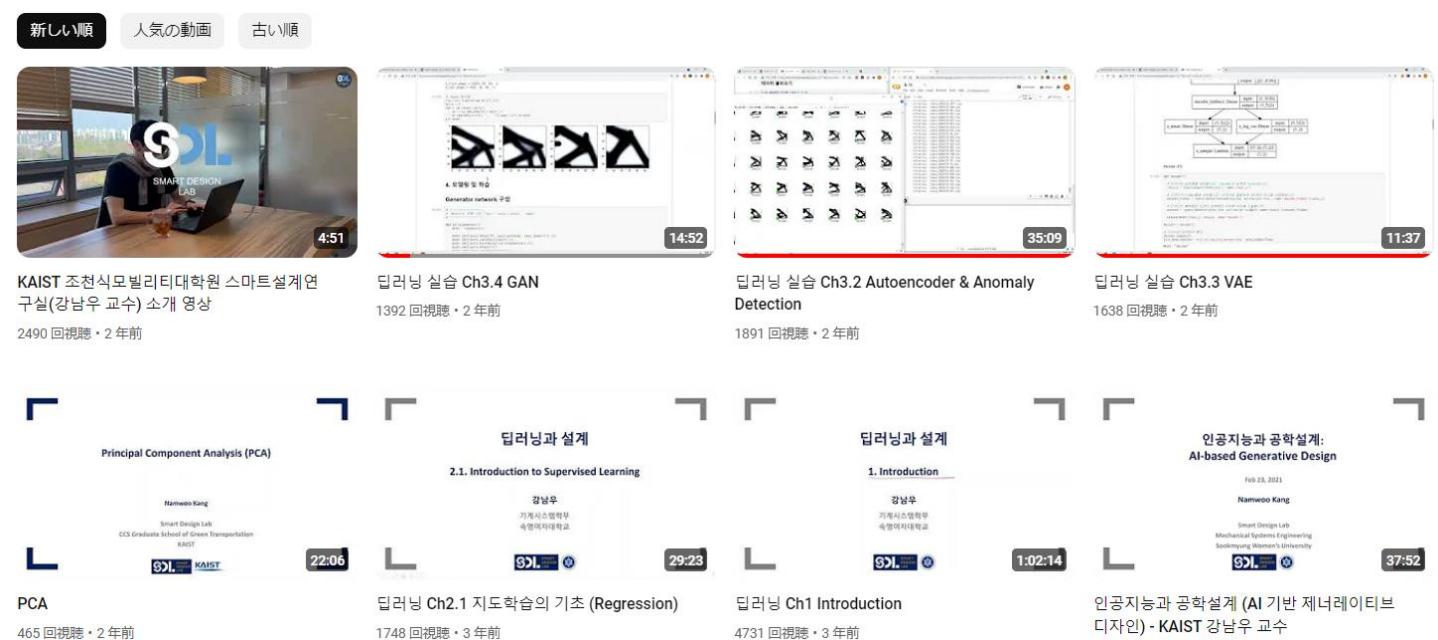
1 from google.colab import files
2
3 uploaded = files.upload()
4
5 # 코드가 돌아가면 파일선택 버튼을 눌러서 upload할 파일 선택
6 for fn in uploaded.keys():
7     print('User uploaded file "{}" with length {} bytes'.format(
8         name=fn, length=len(uploaded[fn])))

... 파일선택 data_3000.zip
• data_3000.zip(application/x-zip-compressed) - 3309996 bytes, last modified: 2021. 8. 12. - 37% done
Saving data_3000.zip to data_3000.zip

[ ] 1

Executing (3s) Cell > upload() > eval_js() > read_reply_from_input()
```

오전 10:21
2021-08-12



https://www.youtube.com/watch?v=ZOQLYgLSN_A

19°C 11:36 2024/05/02

<https://www.youtube.com/watch?v=eg-Ad1OsOJo>

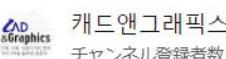
생성형 AI 기반 제품 설계 및 성능 예측 혁신



강남우 대표 (나니아랩스)

제조업 분야에서의 다양한 딥러닝 활용 방안에 대해 소개한다. 또한 생성형 AI를 활용한 3D 데이터 설계부터 성능 예측과 최적 설계까지 포괄적인 사례에 대해 공유하고자 한다.

A set of small, semi-transparent navigation icons located in the bottom right corner of the screen.



チャンネル登録

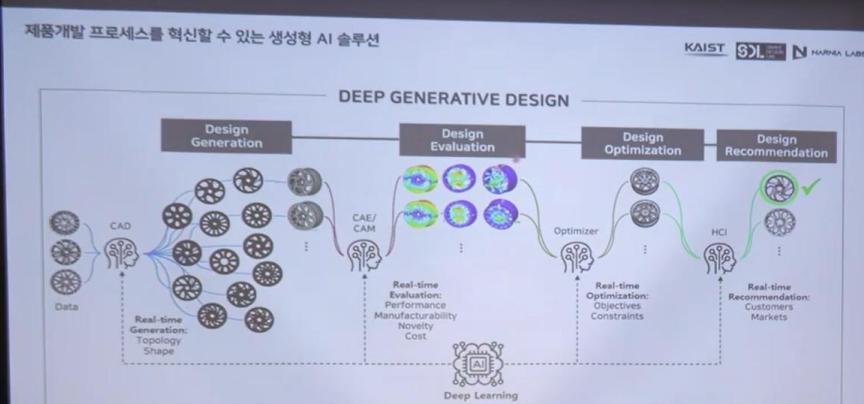
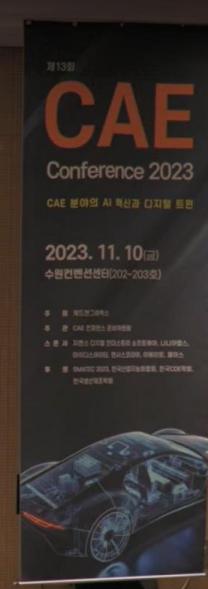
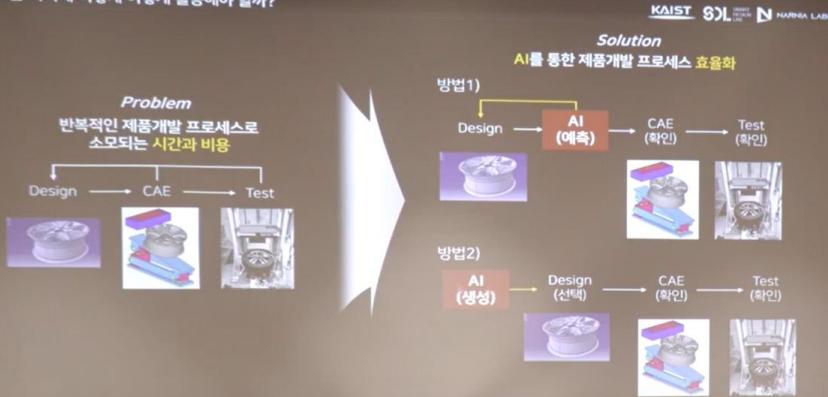
1

共有

↓ オフライン

20

[CAE 컨퍼런스 2023] 생성형 AI 기반 제품 설계 및 성능 예측 혁신



Generative AI-powered Product Design

나니아는 AI의 도움으로 복잡한 설계/디자인 문제를 해결할 수 있는 가상세계입니다.

**Products****Asian Engineer**

공학설계 분야를 위하여

AI를 통해 엔지니어링 시스템 설계를
생성, 탐색, 평가, 및 최적화하는 솔루션**Design Generation**과거 설계 데이터를 기반으로 실마리하고
공학적으로 타당한 설계안을 대량생성**Design Evaluation**생성된 설계안의 다양한 공학성능,
제조성, 특장성, 심미성 등을 실시간으로
예측하고 한인을 설명**Asian Designer**

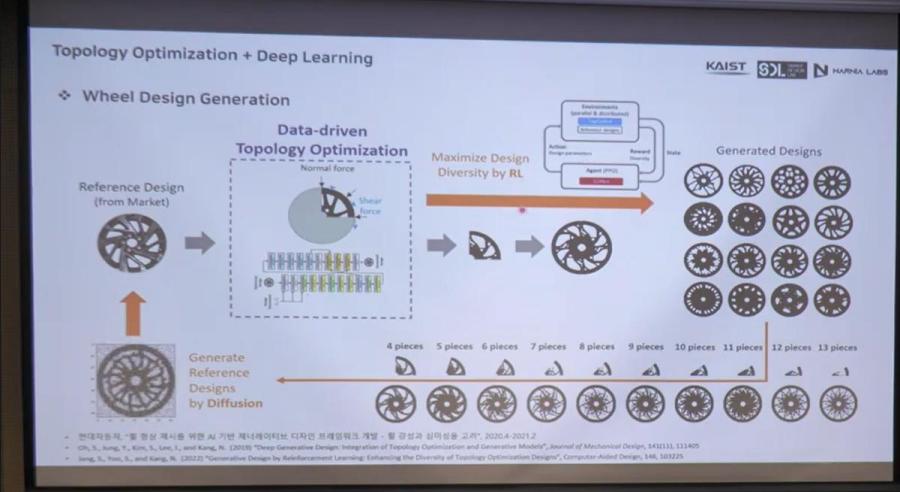
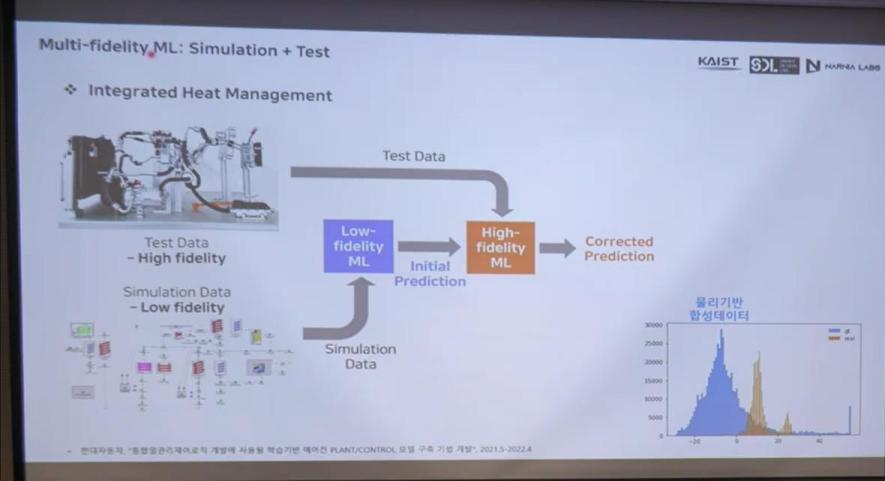
산업디자인 분야를 위하여

AI를 통해 새로운 디자인을 생성하고
고객의 선호도를 학습하여 추천하는 솔루션**Design Generation**디자이너는 원하는 스타일을 스케치로 입력하거나
프롬프트를 통해 키워드를 입력하여
자신의 의도가 반영된 사실적인 제품 디자인 생성**Design Evaluation / Recommendation**실시간으로 생성된 제품 디자인의
공학적 성능, 제조성, 고객 선호도를 실시간으로 예측smartdesignlab.org<http://www.smartdesignlab.org> ▾**Smart Design Lab | KAIST | Namwoo Kang(강남우 ...)**

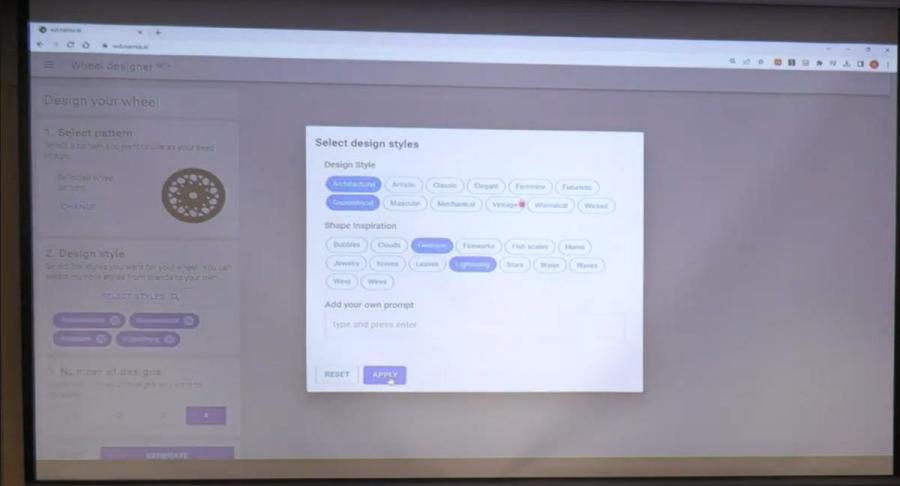
ウェブ WELCOME TO SMART DESIGN LAB Graduate School of Mobility | KAIST (2021-Present)\u0000BMechanical Systems Engineering | Sookmyung Women's University ...

タグ: KAIST Design United States Physics

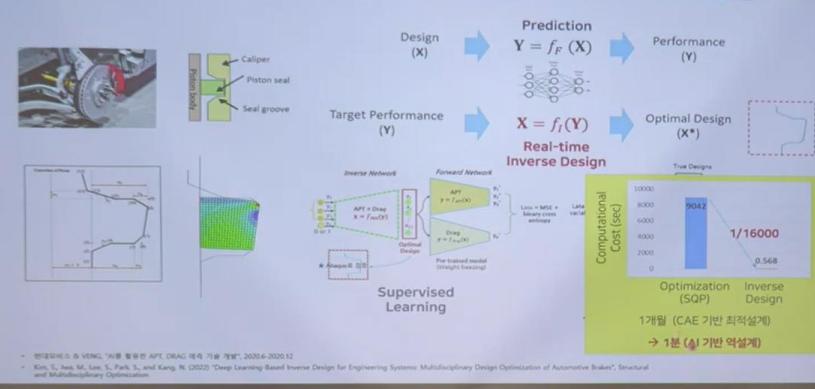
[COLLABORATORS]



[CAF 컨퍼런스 2023] 생성형 AI 기반 제품 설계 및 성능 예측 혁신



Parametric Inverse Design for Brake Systems



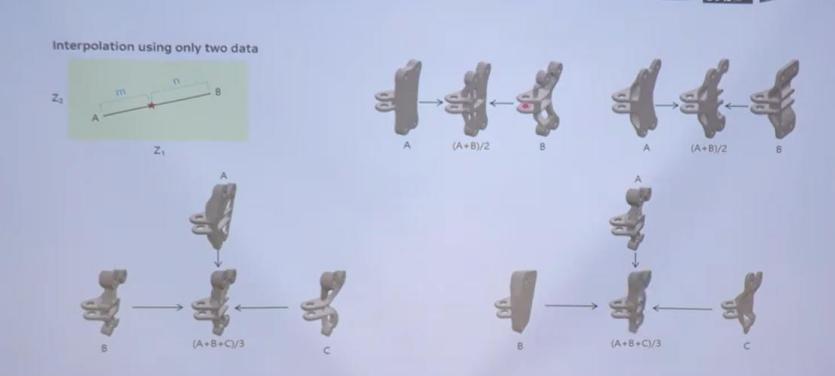
제13회
CAE
Conference 2023

CAE 분야의 시 혁신과 디자인 트렌드

2023. 11. 10(금)
수원컨vens센터(202-203호)

- 차량설계 분야
- CAE 분야는 혁신과 디자인 트렌드
- 차량 설계 분야에서 혁신과 디자인 트렌드
- CAE 분야는 혁신과 디자인 트렌드

3D Interpolation with Small Data Set

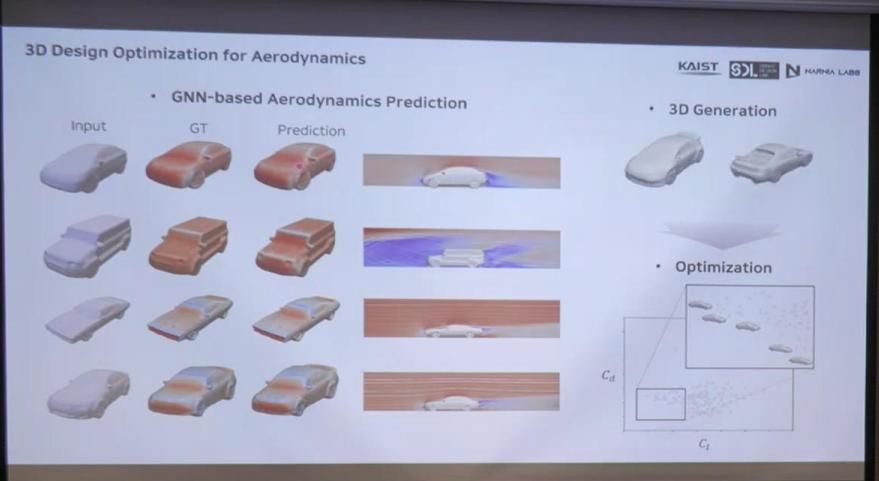
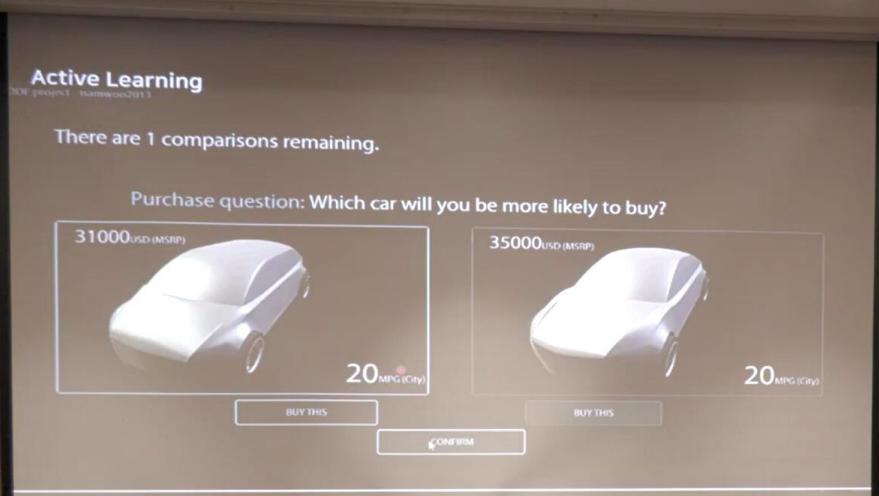
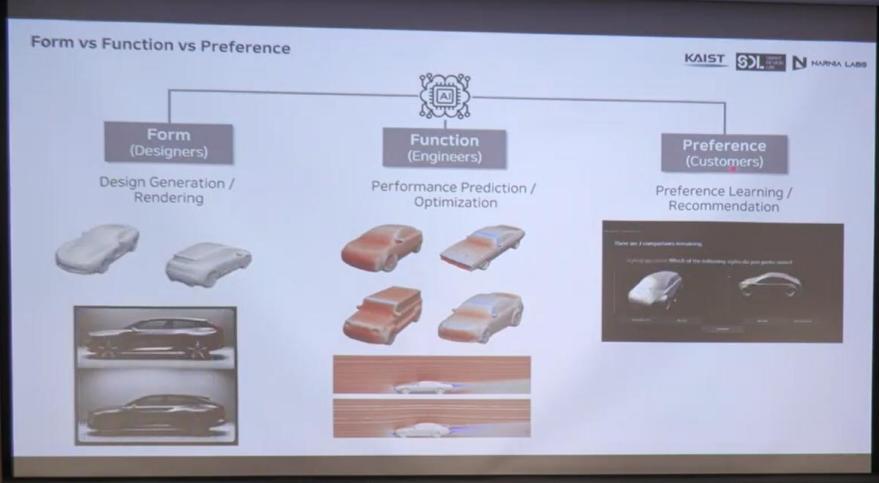


제13회
CAE
Conference 2023

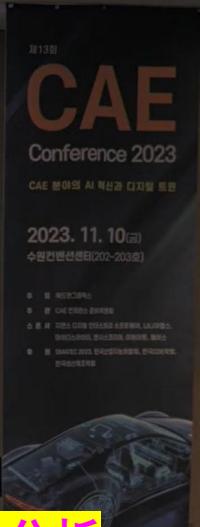
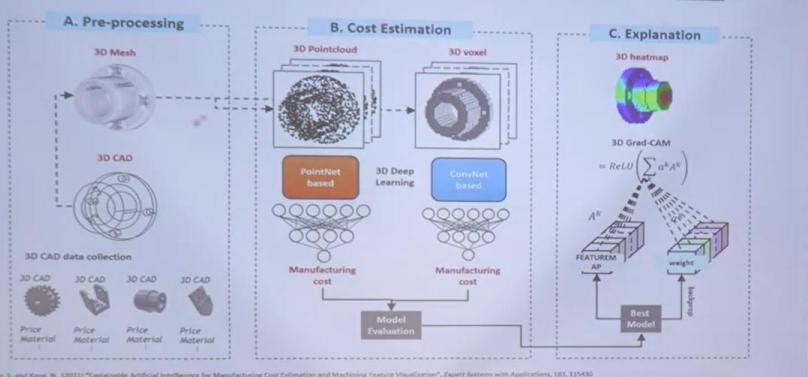
CAE 분야의 시 혁신과 디자인 트렌드

2023. 11. 10(금)
수원컨vens센터(202-203호)

- 차량설계 분야
- CAE 분야는 혁신과 디자인 트렌드
- 차량 설계 분야에서 혁신과 디자인 트렌드
- CAE 분야는 혁신과 디자인 트렌드

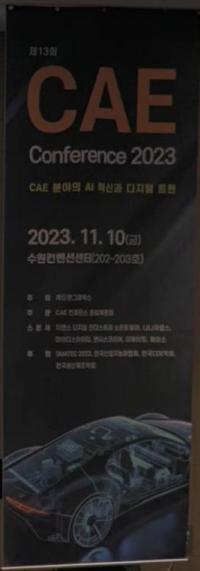
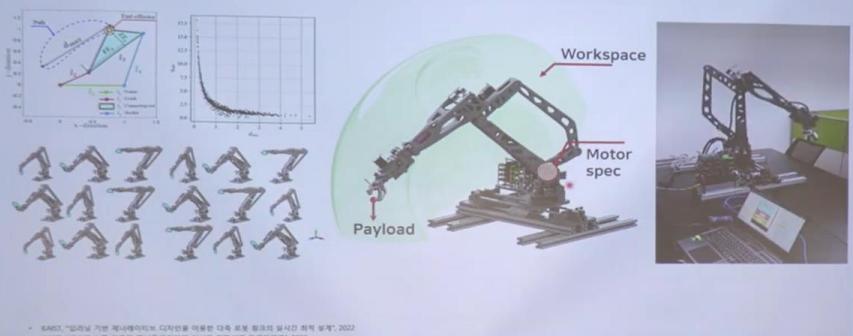


Manufacturing Cost Prediction



どの部分のためにcostが上がっているかを分析

Generative Robot Design for Smart Factory



どこにあるものをpick-upするか?
どのくらい思いものを動かす必要があるかのニーズを入れると、それに最適なメカニズムの逆設計、最適なリンク機構の逆設計が可能に。



TuxRiders

@TuxRiders · チャンネル登録者数 2860人・91 本の動画

TuxRiders is about helping you learn open-source scientific computing, which aims ...さらに表示

[tuxriders.com](#)、他 2 件のリンク

[登録済み](#)

ホーム 動画 再生リスト コミュニティ

新しい順 人気の動画 古い順

COMPILE & USE PARAVIEW PLUGINS



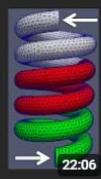
16:45

BUILD PARAVIEW FROM SOURCE CODES



15:52

PARALLEL 3D HEAT DIFFUSION FREEFEM+PETSc



22:06

MULTI-REGION MESH IN SALOME



13:05

Compiling and installing ParaView Plugins [On Demand 16]

354 回視聴・4か月前

Building ParaView from source codes [On Demand 15]

358 回視聴・5か月前

Parallel 3D transient heat diffusion using FreeFEM | Mini-Project 1

476 回視聴・7か月前

Generate multi-body multi-volume mesh geometries in SALOME with mesh...

1546 回視聴・10か月前

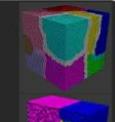
LATTICE BOLTZMANN MULTIPHASE POROUS



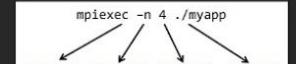
LATTICE BOLTZMANN USING



FREEFEM MESH



PARALLEL COMPUTING



ホーム 動画 再生リスト コミュニティ

High Performance Computing (Parallel Programming)



19:42

[Numerical Modeling 9] High-performance computing and parallel programming in...

532 回視聴・2年前

Python+SymPy To Solve Differential Equations



12:35

[Numerical Modeling 8] Use SymPy for solving ordinary differential equations in...

1854 回視聴・2年前

Quick Look At SymPy Symbolic Algebra in Python



15:36

[Numerical Modeling 7] A quick look at SymPy for symbolic computation in Python

309 回視聴・2年前

Essential Matplotlib Plotting in Python



14:25

[Numerical Modeling 6] Essential Matplotlib, data plotting library for Python

272 回視聴・2年前

SciPy In A Nutshell



15:37

[Numerical Modeling 5] SciPy in a nutshell, the library of scientific algorithms for Python

309 回視聴・2年前

A Short NumPy Tutorial



20:26

[Numerical Modeling 4] A short tutorial on NumPy, multi-dimensional arrays and...

319 回視聴・2年前

A Quick Overview Of Python



23:56

[Numerical Modeling 3] A quick overview of Python programming language

482 回視聴・2年前

Introducing Jupyter Notebooks



10:08

[Numerical Modeling 2] Introduction to Jupyter notebooks

576 回視聴・2年前

Introduction To Applied



Microsoft 365 Basic에는 이제 추가 비용 없이 파일과 사진에 대한 랜섬웨어 보호 기능이 포함됩니다. 자세한 정보 >

웹에서 Word, Excel, PowerPoint 등을 무료로 사용

ffice가 Microsoft 365로 변경됩니다

로그인

무료 등록

[Microsoft 365를 원하시나요? 플랜 및 가격 보기 >](#)

댓글에 온라인 오피스 링크를 접속하고

▶ ▶ | 🔍 1:04 / 1:23

A set of small, light-gray navigation icons located at the bottom of the screen. From left to right, they include: a back arrow, a forward arrow, a search icon (magnifying glass), a settings gear icon, a square icon, a rectangle icon, and a grid icon.

MS오피스365를 무료로 사용할 수 있다고?



B급지식

チャンネル登録者数 728人

チャンネル登録

 高評價

共有

↓ オフライン

...

【初心者向け】 売れる！Webコーピーライティング基礎講座 ~ホームページ・LPの構成を変えるだけで売上UP~



mikimiki web スクール
チャンネル登録者数 22.4万人

チャンネル登録

320



共有

オフライン

...

【徹底比較】WordPressとWixはどっちが使いやすい？



るうーのWeb酒場
チャンネル登録者数 775人

チャンネル登録

96



Wixの使い方 アカウント設定 ホームページの作り方の基本 初心者向け



ひでやのチャンネル【 Wix ...
チャンネル登録者数 3760人

チャンネル登録

227



共有

...

起業初心者のためのWIXホームページ作り方



有田絵梨
チャンネル登録者数 2250人

チャンネル登録

FreeCAD Arch - Creating 2D Plan - 02

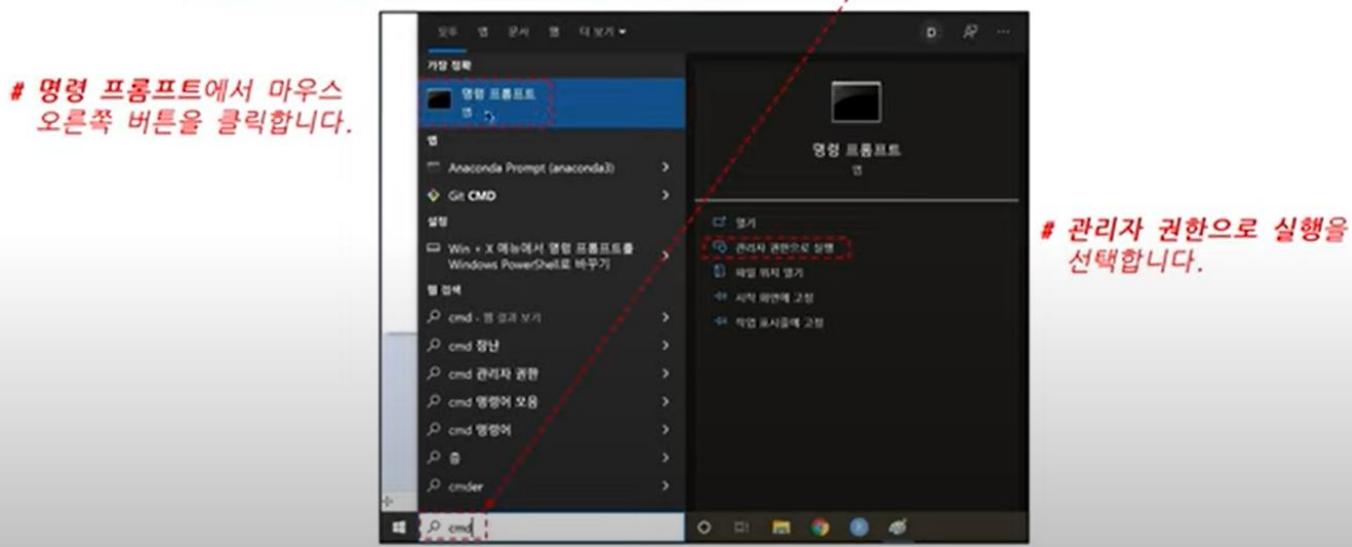


Regis Nde Tene
チャンネル登録者数 5150人

チャンネル登録

Windows에서 CMD 실행

- Windows 작업 표시줄에 있는 돋보기 메뉴에서 cmd를 입력하고 실행합니다.



Python pip 업데이트

- Python 라이브러리를 설치할 때 pip를 이용하면 좋습니다.
 - pip는 Python 라이브러리 설치 및 관리 시스템이며, 3.4 버전에서 추가되었습니다.
 - Windows의 CMD, MacOS의 Terminal을 열고 아래 명령어를 입력하여 pip를 실행합니다.
- 현재 설치된 Python 라이브러리 목록을 확인합니다.
 - # 원쪽 명령어를 실행했을 때 맨 아래에 노란색 경고를 확인했다면 pip를 업그레이드해야 합니다.
% pip list # [참고] Python을 설치할 때 환경변수에 추가하지 않았다면 맨 앞에 python -m을 추가해야 합니다.
- pip 버전을 업그레이드하고, 설치된 버전을 확인합니다.
 - % pip install --upgrade pip # [참고] 원쪽 명령어를 실행했을 때 CMD를 관리자 권한으로 실행하는 것이 아니라면 에러가 발생할 수 있으므로 마지막에 --user를 추가합니다.
% pip --version # pip 현재 버전을 확인합니다.

Python 라이브러리 설치(for Basic Course)

- Basic 과정에 필요한 Python 라이브러리를 설치합니다. # [참고] Python 라이브러리 탐색 위치
<https://pypi.org/>

```
% pip install jupyter # jupyter 라이브러리를 설치합니다. 의존성 라이브러리를 함께 설치합니다.  
[참고] JupyterLab을 실행하려면 jupyterlab 라이브러리를 추가 설치해야 합니다.
```

% pip install pandas # pytz, numpy, pandas 라이브러리를 설치합니다.

% pip install openpyxl # et-xmlfile, openpyxl 라이브러리를 설치합니다.

```
% pip install xlrd      # xlrd 라이브러리를 설치합니다. <table>
```

```
% pip install seaborn # pyparsing, pillow, kiwisolver, fonttools, cycler, contourpy, matplotlib, seaborn 라이브러리를 설치합니다.
```

```
% pip install chardet # chardet 라이브러리를 설치합니다.
```

```
% pip install joblib # joblib
```

% pip install joblib # joblib 라이브러리를 설치합니다.
'---.z' 파일
✓ UTF-8
pd.read_csv('---.csv')
encoding='cp949'

[참고] Python 라이브러리 설치 오류 해결

- 외부 라이브러리를 설치할 때 신뢰할 수 없다는 이유로 라이브러리를 설치하지 못하는 오류가 발생할 수 있습니다.

```
Could not fetch URL https://pypi.org/simple/pip/: There was a problem confirming the ssl certificate
: HTTPSConnectionPool(host='pypi.org', port=443): Max retries exceeded with url: /simple/pip/ <Cause
d by SSLError[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed <_ssl.c:852>
,--> - skipping
```

- 이와 같은 오류를 해결하기 위해 pip 명령어에 아래 코드를 추가합니다.

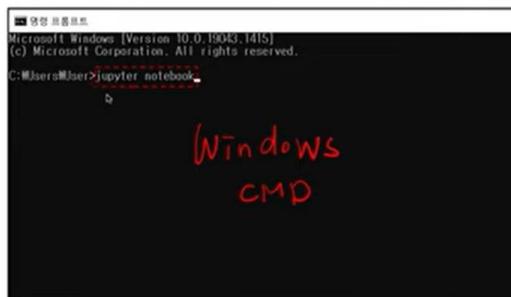
```
% pip install charset-normalizer --trusted-host pypi.org
```

[참고] 위 명령어로 해결되지 않으면 pypi.org 대신 pypi.python.org 또는 files.pythonhosted.org로 바꿔서 실행해 보시기 바랍니다.

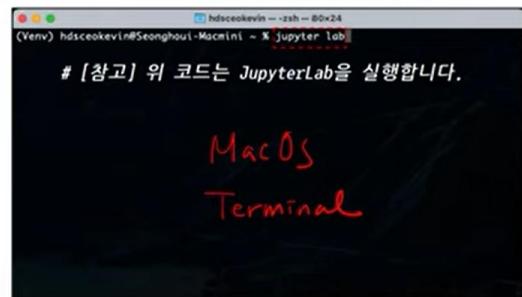
Jupyter Notebook 실행

- CMD(Windows) 또는 Terminal(MacOS)에서 아래 코드를 실행하면 기본 브라우저에서 새 탭이 열리고, Jupyter Notebook을 실행합니다.

[참고] Jupyter Notebook을 크롬에서 실행하는 것을 추천합니다.



[참고] Jupyter Notebook 실행 경로를 아래와 같이 추가할 수 있습니다.
% jupyter notebook --notebook-dir=경로명



[참고] 위 코드는 JupyterLab을 실행합니다.

Python 라이브러리 설치(for Advanced Course)

- Advanced 과정에 필요한 Python 라이브러리를 추가 설치합니다.

```
% pip install scikit_learn # threadpoolctl, scipy, scikit-learn 라이브러리를 설치합니다.
```

```
% pip install statsmodels # pasty, statsmodels 라이브러리를 설치합니다.
```

```
% pip install graphviz # graphviz 라이브러리를 설치합니다.
```

```
% pip install scikit_posthocs # scikit_posthocs 라이브러리를 설치합니다.
```

```
% pip install pingouin # littleutils, charset-normalizer, urllib3, tabulate, lazy_loader, certifi, requests, xarray, outdated, pandas_flavor, pingouin 라이브러리를 설치합니다.
```

[참고] 아나콘다 사용자는 터미널에서 아래 코드를 실행합니다.
% conda install -c conda-forge pingouin

[참고] Python 라이브러리 설치에 관한 튜토리얼은 아래 링크를 참고하세요.
<https://packaging.python.org/en/latest/tutorials/installing-packages/>

[참고] Python 라이브러리 업그레이드

- 현재 설치된 Python 라이브러리 중 업그레이드가 필요한 목록을 출력합니다.

```
% pip list --outdated # 라이브러리별로 현재 버전과 업그레이드 버전을 함께 출력합니다.
```

- 라이브러리를 하나씩 업그레이드합니다.

```
% pip install --upgrade 라이브러리 # 위 결과로 확인한 라이브러리를 하나씩 업그레이드합니다.
```

- [참고] 업그레이드가 필요한 라이브러리 목록을 txt 파일로 생성합니다.

```
% pip list --outdated --format=freeze > upgradeList.txt # 'upgradeList.txt' 파일을 열고  
# '='를 '>'으로 변경합니다.
```

- [참고] txt 파일을 읽고 여러 라이브러리를 한 번에 업그레이드합니다.

```
% pip install -r upgradeList.txt --upgrade
```

13

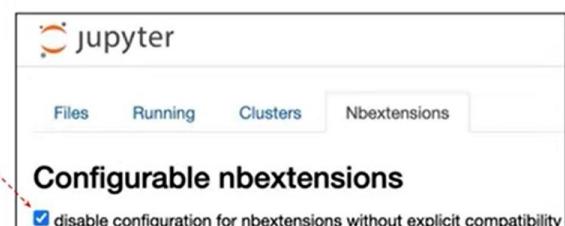
[참고] Jupyter Notebook 확장 프로그램 설치

- Jupyter Notebook에서 확장 프로그램을 사용하려면 아래 코드를 실행합니다.

```
{ % pip install jupyter_contrib_nbextensions # Jupyter Notebook 확장 라이브러리를 설치합니다.  
# [참고] JupyterLab은 설치할 필요 없습니다.  
% jupyter contrib nbextension install --user # Jupyter Notebook 확장 프로그램을 사용하려면  
# 원쪽 코드를 실행합니다.
```

- Jupyter Notebook을 열고 확장 프로그램을 사용할 수 있도록 설정합니다.

Nbextensions 탭을 열고 Configurable nbextensions 앞 체크를 해제합니다.
일부 유용한 도구를 추천합니다.
- Variable Inspector
- Table of Contents (2)
- ExecuteTime



14

The screenshot shows a Google Colab notebook titled "Python Basic Intro". The notebook has a single section titled "데이터 분석을 위한 파이썬 기초". Below this section, there is a URL: https://bit.ly/hds_python_basic_intro.

Colab 사용법

- 크롬을 열고 로그인한 상태에서 크롬 주소창에 위 링크를 붙여넣습니다.
- Colab 파일을 실행하기 전에 사본을 저장해놓으면 나중에 복습하기 좋습니다.
- Colab 파일 사본을 저장하려면 화면 상단에 있는 <Drive로 복사> 버튼을 클릭합니다.



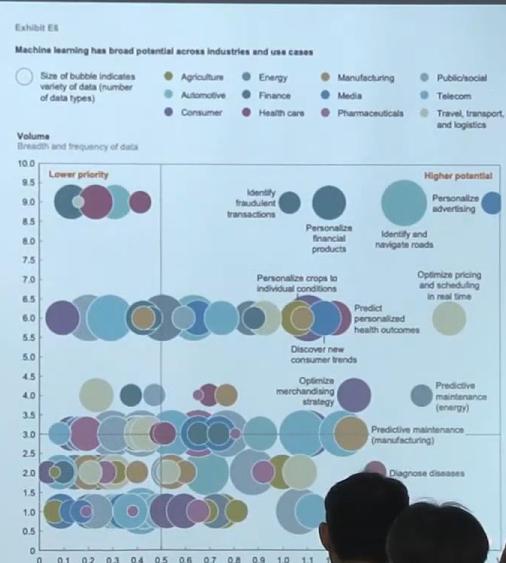
최초 머신러닝 팀 빌딩

직군	직업	최소인원
프러덕트	프러덕트 매니저	1
비즈니스	비즈니스 분석가	1
데이터 사이언스	데이터 사이언티스트	1
머신러닝	머신러닝 엔지니어	1



프로젝트 예산

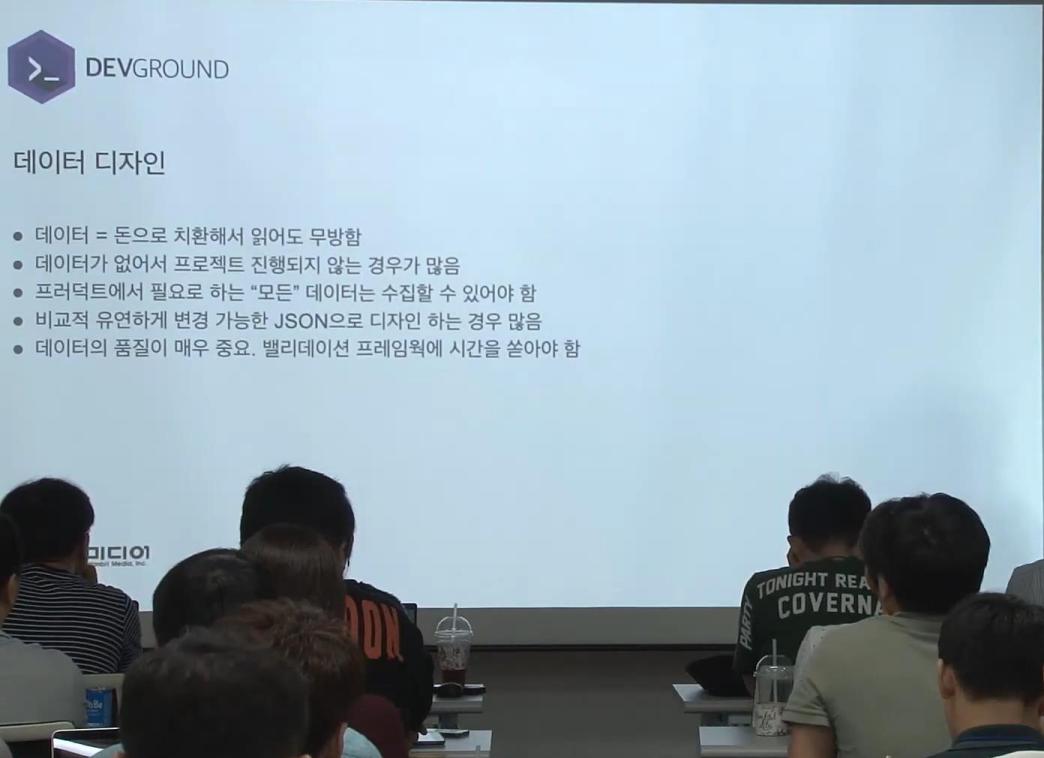
- 팀 인건비
 - 4명 월급 4000만원
- 인프라 비용
 - PC 1000만원
 - 머신러닝 인프라 3억
- 소프트웨어 구입 및 구독
 - 기타 개발도구 1000만원
- 예상 개발기간
1년





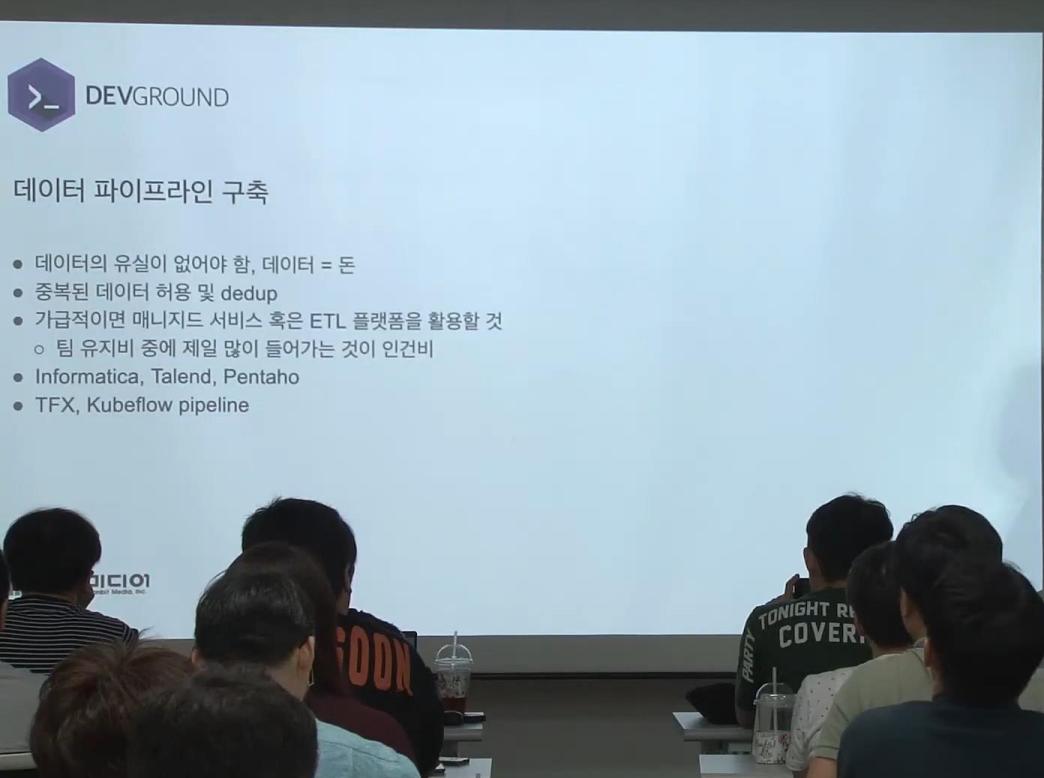
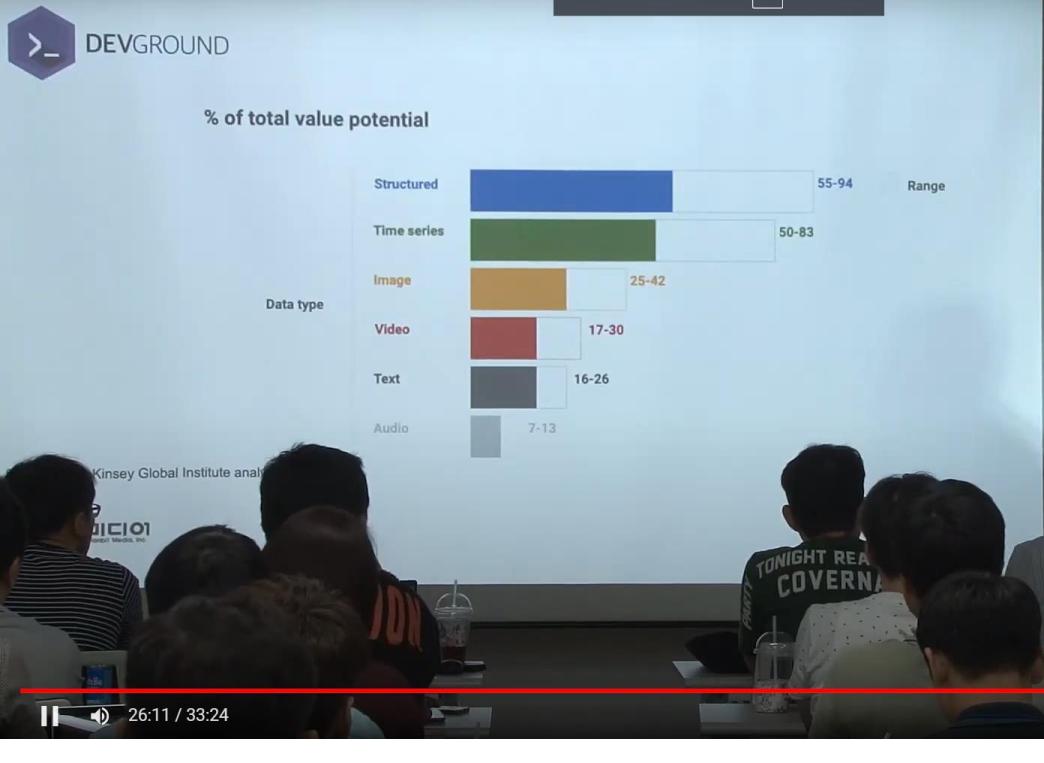
데이터 디자인

- 데이터 = 돈으로 치환해서 읽어도 무방함
- 데이터가 없어서 프로젝트 진행되지 않는 경우가 많음
- 프로젝트에서 필요로 하는 "모든" 데이터는 수집할 수 있어야 함
- 비교적 유연하게 변경 가능한 JSON으로 디자인 하는 경우 많음
- 데이터의 품질이 매우 중요. 밸리데이션 프레임워크에 시간을 쏟아야 함



[데브그라운드 2019] 백정상 구글코리아 - AI 프로젝트 간지나게 잘 진행하는 법

全画面表示を終了するには Esc を押します



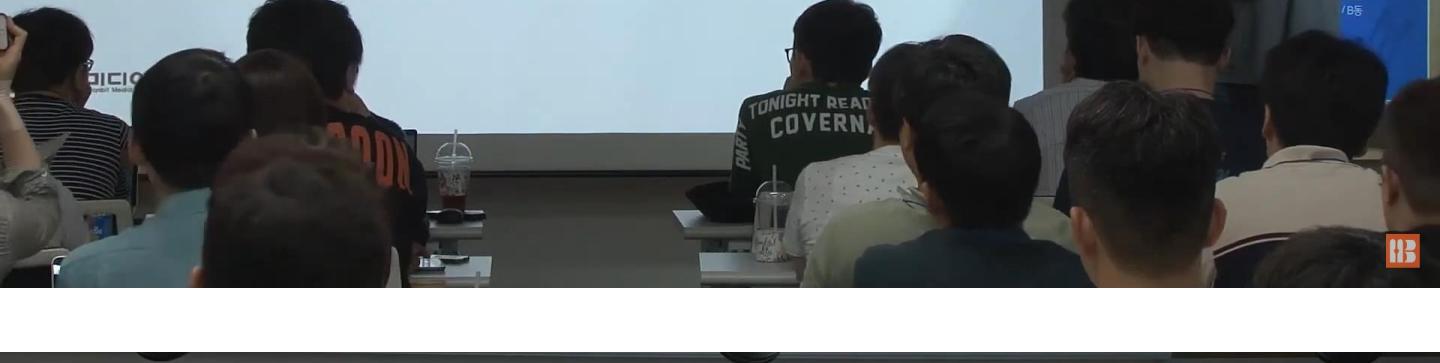
데이터 파이프라인 구축

- 데이터의 유실이 없어야 함, 데이터 = 돈
- 중복된 데이터 허용 및 dedup
- 가급적이면 매니지드 서비스 혹은 ETL 플랫폼을 활용할 것
 - 팀 유지비 중에 제일 많이 들어가는 것이 인건비
- Informatica, Talend, Pentaho
- TFX, Kubeflow pipeline



데이터 분석

- EDA
- 데이터 상관관계 분석
- 통계적 검증: 빈도검증, 타탕도 검증, z 스코어 검증, t 스코어 검증 etc.
- 고전적 머신러닝: 회귀모형, 클러스터링
- 데이터의 품질과 특징을 분석해야 한다
- 편견이나 확증편향이 생기면 안된다! 통계 검증을 통해 가설을 검증하는 것
- 데이터 분석만으로도 문제를 해결하는 경우가 많음. 이 경우에는 바로 비즈니스 임팩트를 만들고 다음 프로젝트를 간지나게 시작한다 이득!
- 통계나 머신러닝으로 threshold를 구할 수 있다면 룰 베이스 모델 구현



머신러닝 시작

- 분석 결과에 따른 최적의 모델 선택
- 논문 - 아카이브
- AI 툴셋 - 구글 클라우드 AI 플랫폼, AI Hub
- 모델 개발
 - 텐서플로, Pytorch
 - Colab, 워크스테이션, Cloud ML Engine etc
- 매니지드 서비스 활용: Automl Vision, Translation, Natural Language, Video Intelligence, Tables etc.



모델 학습 및 평가

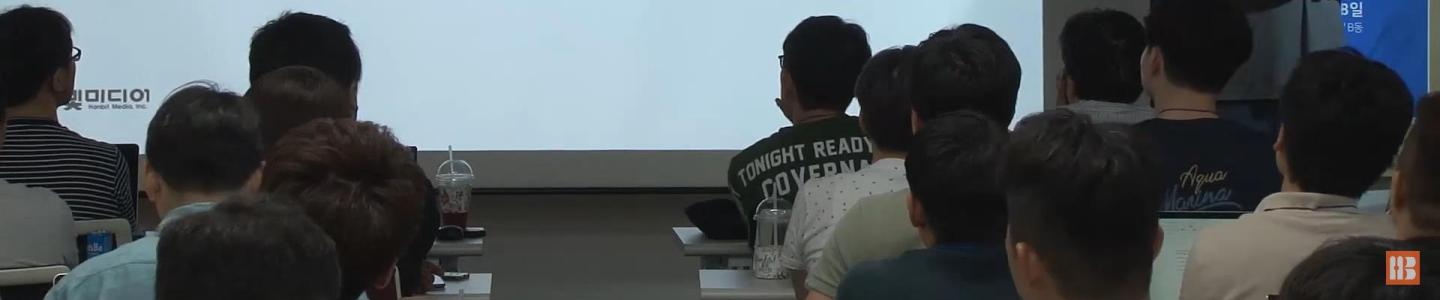
- 피쳐 셀렉션의 두가지 전략: 다 넣자 vs 상관관계에 따라 선택하자
- 절충안: 다 넣고 상관관계가 높은 피쳐에 웨잇을 더 가하자
- 피쳐 엔지니어링 & 셀렉션 작업 시작
 - Dataprep 툴을 쓰는 것이 편리
 - Numeric, Categorical
 - Standardization, Normalization
- 후보 모델간 평가 필수
 - 레이블 데이터가 필수
 - 평가 메트릭에 대한 이해 필수

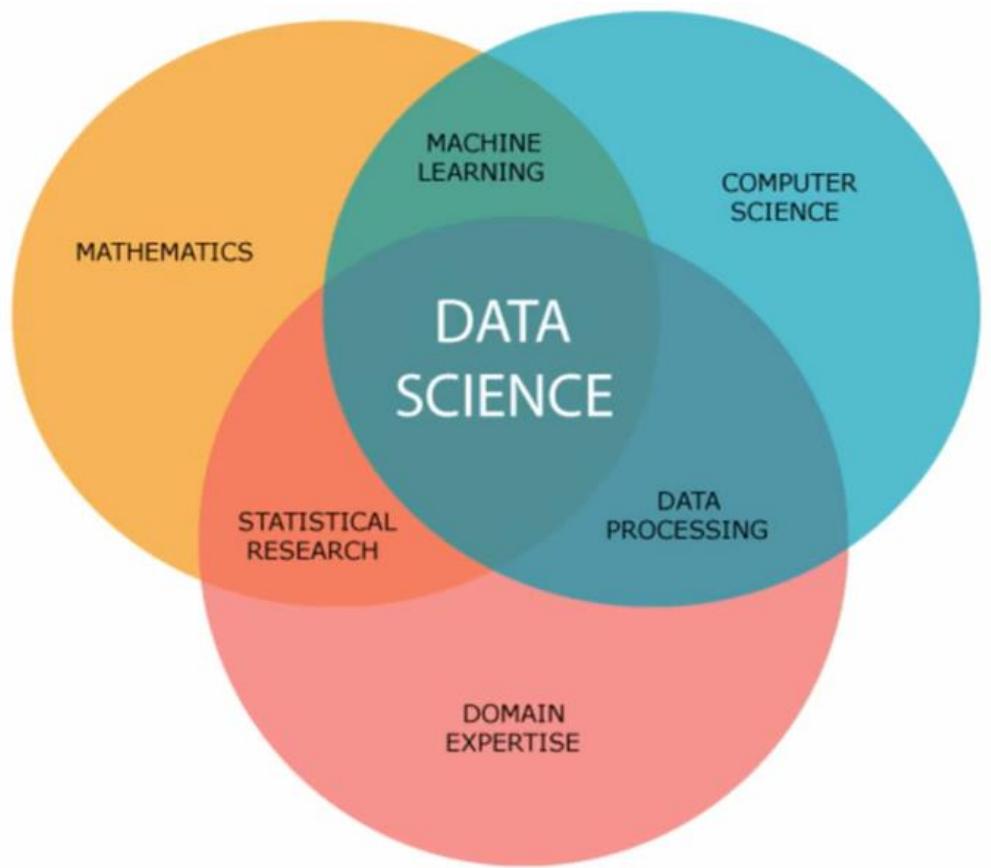




현실에서는

- AI 프로젝트는 매우 어렵다. 10개 중에 8개가 비즈니스 임팩트를 만드는 데 실패한다
- 80억 벌 수 있을만한 AI 비즈니스 케이스를 찾기가 매우 어렵다
- 좋은 인재는 1억을 줘도 채용하기 힘들다. 실리콘밸리는 기본 3억부터 시작
- 저거 다 1년만에 하기 어렵다
- 말은 연구 프로젝트 실제로는 스타트업. 한번의 실패 => 나락. 2번 기회를 주는 경우가 드물다
- 대표에게 지원 받으면 실패시 대표에게도 책임이 간다. 프로젝트 늘어지면 스트레스 심각함



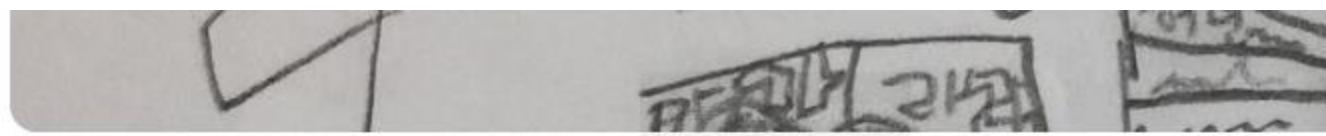


[이것이 데이터 분석이다 with 파이썬_00] 왜 Data Science 인가? 탐색, 시각화, 목적 정리



3.8万回 聽 4年前





오늘코드todaycode

@todaycode · チャンネル登録者数 2.5万人 · 339 本の動画

Microsoft MVP(Python Developer Technologies) >

github.com/corazzon、他 3 件のリンク

登録済み ▾

jupyter Pandas Expanding and Rolling (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

In [13]: # 이동평균
s.plot(style='k--')
r.mean().plot(style='k')

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x112253ac8>

A small orange dog icon is located in the bottom right corner of the Jupyter interface.

[16/29] Pandas 기초 - 시계열 데이터(Time Series Data) 분석을 위한 판다스 Expanding and Rolling 이해하기



오늘코드todaycode
チャンネル登録者数 2.5万人

チャンネル登録

107



共有

オフライン

...

chatgptを活用して工学用のプログラミングを実施するやり方を紹介している
youtube channelがあれば、有名な準で10個紹介して

ChatGPTを活用して工学用のプログラミングを学ぶためのYouTubeチャンネルをいくつかご紹介します。以下は、有名で参考になるチャンネルです：

1. 【革命的すぎ】 プログラミング学習をChatGPTを使って10倍効率化する方法¹:

1. チャンネル: だれでもエンジニア / 山浦清透
2. 内容: ChatGPTを活用してプログラミング学習を効率化する方法について解説しています。

2. ChatGPTをプログラミング学習に役立てる方法5選²:

1. チャンネル: たにぐちまことのともすたチャンネル
2. 内容: ChatGPTを使ってプログラムの意味を教えてもらったり、改善を依頼したりする方法を紹介しています。

3. ChatGPTでプログラミングを効率化しよう！驚異のプログラミング自動化！GPT-4も利用³:

1. チャンネル: 中村祐太のプログラミング入門ナビ
2. 内容: ChatGPTを使ってプログラミングを自動化する方法について解説しています。



中村祐太のプログラミング入門ナビ

@yuta-nakamura · チャンネル登録者数 3200人 · 71 本の動画

プログラミング学習者に役立つ情報を発信します。ITエンジニア / プログラミング講師。 >

[proglus.jp](#)、他 2 件のリンク

[チャンネル登録](#)

たにぐちまことのともすたチャンネル

@tomosta · チャンネル登録者数 4.9万人 · 1111 本の動画

学ぶ。をちゃんとコーポレートテーマに、プログラミング・クリエイティブを学びたい... >

[twitter.com/seltzer](#)、他 2 件のリンク



登録済み ▾

メンバーになる

中村祐太のプログラミング入門ナビ

@yuta-nakamura · チャンネル登録者数 3200人 · 71 本の動画

プログラミング学習者に役立つ情報を発信します。ITエンジニア / プログラミング講師。 >

[proglus.jp](#)、他 2 件のリンク

[チャンネル登録](#)

マークティングの定義

OOプロジェクトのアイディアを教えて

ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT Mar 3 Version

教えてと入力して様々な提案を

ChatGPTの使い方入門 - より良いアイディアを作成しよう！

中村祐太のプログラミング入門ナビ
チャンネル登録者数 3200人

登録済み

11



共有

オフ

ステップバイステップ

- 課題:「Pythonを使って、ウェブサイトからデータをスクレイピングし、そのデータを分析してグラフに表示する」
 - ステップ 1: ウェブスクレイピング
 - ステップ 2: データの整理
 - ステップ 3: データ分析
 - ステップ 4: グラフ作成

プロンプトのコツ

- 具体性
- 言語の指定
- ステップバイステップの指示

注意点

- 情報の正確性(間違った情報が含まれることも)
- セキュリティとプライバシー

コーディング支援専用のAIツール

GitHub Copilot &
Copilot Chat

VSCodeで始めるAIプログラミング支援

OpenAIが
800万ドル投資!!

Cursor

革新的AIコードエディタ

プログラミングゼミ・SiID

ChatGPT×プログラミングでできること8選【活用法】



セイト先生のWeb...
チャンネル登録者数 11.9...

メンバーになる

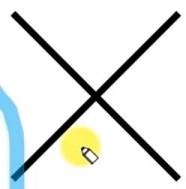
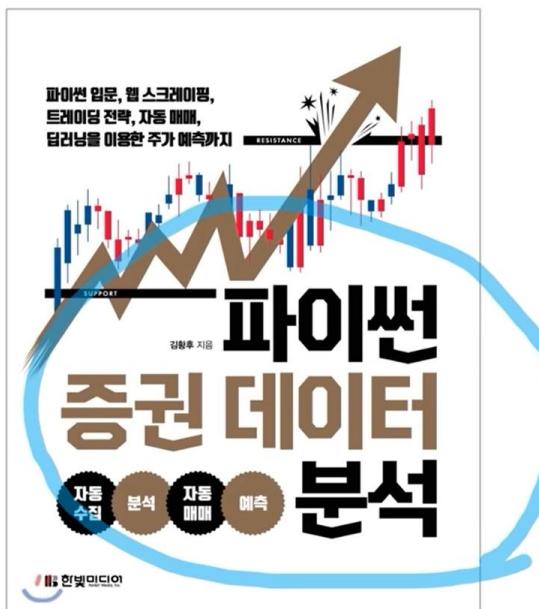
チャンネル登録

296



共有

オフ



크녕 제가 여러 책을 참고하다가 내용이 괜찮아서

▶ 구독

- 라이브 일정 • LIVE (토요일 저녁 8시~10시)
- 1주차 6/24(토) - 01장. 파이썬이란 무엇인가?
 - 2주차 7/1(토) - 02장. 파이썬 프로그래밍의 기초, 자료형 (1)
 - 3주차 7/8(토) - 02장. 파이썬 프로그래밍의 기초, 자료형 (2)
 - 4주차 7/15(토) - 03장. 프로그램의 구조를 쌓는다! 제어문
 - 5주차 7/22(토) - 04장. 파이썬의 입출력
 - 6주차 7/29(토) - 05장. 파이썬 날개 달기
- * 라이브 종료 후 풀버전은 멤버십으로 전환됩니다.
* 편집본은 조코딩 채널에 무료로 공개됩니다.
* 일정 변동이 생기면 유튜브 커뮤니티를 통해 사전 공지 드리겠습니다.



GitHub
Copilot

챗GPT로 누구나 쉽게 코딩..."SW 엔지니어 전성시대 끝났다"

입력 2023-06-14 10:51:45 수정 2023.06.14 18:06:55 실리콘밸리=정혜진 특파원

"코딩 몰라도 개발 문제없다"…MS, AI 기반 혁신 기술 공개

"2시간 걸리던 코딩 2분 만에"...AI 기반 개발 시대 열려



C Write the Python web crawling code.

Certainly! Here's an example of Python code using the 'requests' and 'BeautifulSoup' libraries for web crawling:

```
python
import requests
from bs4 import BeautifulSoup

# Define the URL you want to crawl
url = 'https://example.com'

# Send a GET request to the URL
response = requests.get(url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Parse the HTML content using BeautifulSoup
    soup = BeautifulSoup(response.content, 'html.parser')

    # Extract specific data from the parsed HTML
    # Here's an example of extracting all links from the page
    links = soup.find_all('a')

    # Print the extracted links
    for link in links:
        print(link['href'])
```

과거

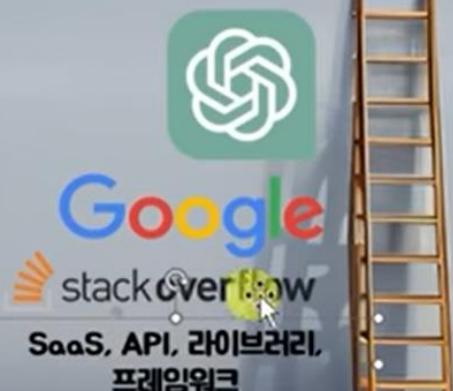
프로그램 개발

데이터베이스
운영체제
컴퓨터구조
알고리즘
자료구조
프로그래밍 언어

현재

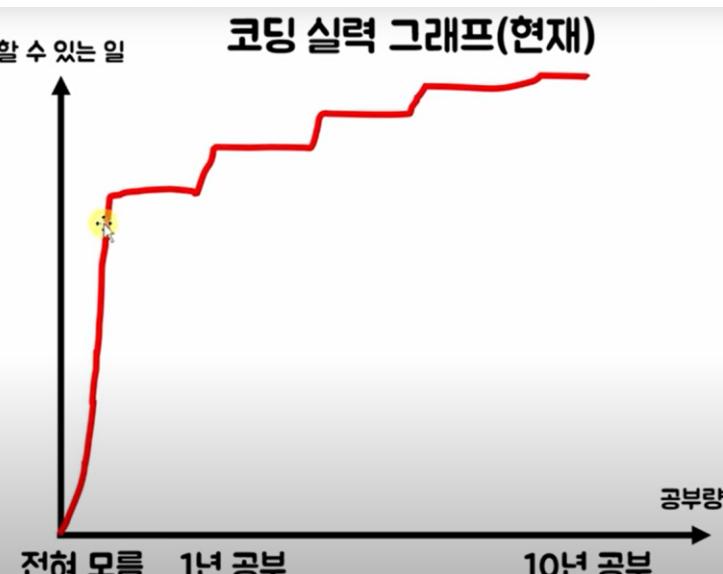
프로그램 개발

기초 지식



할 수 있는 일

코딩 실력 그래프(현재)



기초 지식?

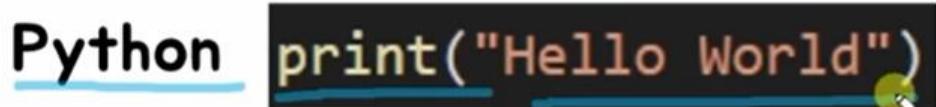
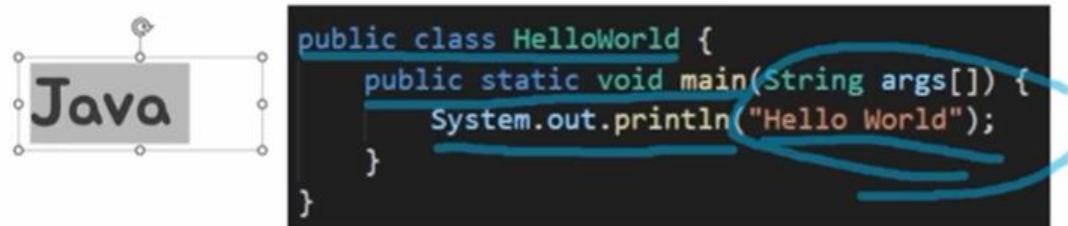
1. 프로그래밍 언어

(변수, 조건문, 반복문, 함수 등)

2. 도구 사용법

(챗GPT, API, 라이브러리, 프레임워크 등)

1. 쉬움

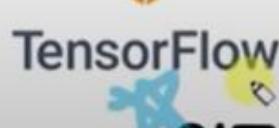


2. 써먹을 곳 많음



웹 개발

데이터분석



인공지능



사물 인터넷

3. 사람들 많이 씀

PYPL PopularitY of Programming Language

Worldwide, Jun 2023 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	27.66 %	+0.2 %
2		Java	16.16 %	-1.3 %
3		JavaScript	9.44 %	-0.2 %
4		C#	6.79 %	-0.5 %
5		C/C++	6.6 %	-0.1 %
6		PHP	5.04 %	-0.4 %
7		R	4.17 %	-0.3 %
8		TypeScript	3.11 %	+0.5 %
9		Swift	2.38 %	+0.2 %
10	↑↑↑↑	Rust	2.24 %	+0.8 %

The PYPL PopularitY of Programming Language Index is created by analyzing how often language **tutorial**s are searched on Google.

출처: <https://pypl.github.io/PYPL.html>

3. 사람들 많이 씀



출처: <https://survey.stackoverflow.co/2023/#technology>

3. 사람들 많이 씀



출처: <https://redmonk.com/sogrady/2023/05/16/language-rankings-1-23/>

파이썬 라이브 링크

Pythontutor.com/live.html#mode=edit

line that just executed
Next line to execute 3:11:48 / 4:46:23 • Immutable vs Mutable

최신 파이썬 코딩 무료 강의 - 5시간만 투자하면 개발자가 됩니다

조코딩 JoCoding チャンネル登録者数 59.6... メンバーになる 登録済み 2万 共有 オフライン

This mode is experimental. Use the [regular Python Tutor](#) to get live help.

Write code in Python 3.6 (drag lower right corner to resize code editor)

```
1 ##### Immutable #####
2 a = 1
3 def vartest(a):
4     a = a +1
5 vartest(a)
6 print(a)
7 #####
```

Print output (drag lower right corner to resize)

Frames Objects

Global frame

a → int 1

vartest → function vartest(a)

vartest

a → int 2

Return value → NoneType None

—列ずつ実行可能！

리스트를 쓰는 이유?

1 a
2 b
3 c

김민수
조코딩님 구름ide에서 아플 출시하신 영상 잘았구나
ios도 무드드려요ㅠㅠ

int 조수
공지상으로 10시로 되어있어요

Eric
@정서율 원래 8시부터 10까지입니다

박준하
매주 8시 부터 10시 교재이름 : 점프 투 파이썬 Q: 무
편집본 올라가나요? A: 네

Life is too short You need python
qkf

Life is too short You need python
방

int 조수
편집본도 올라가는데 워낙 시간이 길어서

흔자님
늦을뻔ㅋㅋ

int 조수
?

int 조수
[조코딩 라이브 시간 - 8시~10시]

최신 파이썬 코딩 무료 강의 - 5시간만 투자하면 개발자가 됩니다

03-1 If 문

돈이 있으면 택시를 타고, 돈이 없으면 걸어 간다

```
>>> money = True  
>>> if money:  
...     print("택시를 타고 가라")  
... else:  
...     print("걸어 가라")  
...  
택시를 타고 가라
```

```
>>> money = True  
>>> if money:
```

불 자료형

<, >, ==, !=, >=, <=,
and, or, not
&, |

in, not in

다중 조건 판단 elif

```
>>> pocket = ['paper', 'cellphone']  
>>> card = True  
>>> if 'money' in pocket:  
...     print("택시를 타고가라")  
... elif card: Else ifではないことに注意  
...     print("택시를 타고가라")  
... else:  
...     print("걸어가라")  
...  
택시를 타고가라
```

for와 함께 자주 사용하는 range함수

```
>>> sum = 0  
>>> for i in range(1, 11):  
...     sum = sum + i  
...  
>>> print(sum)  
55
```

Break

loopから脱出する。

Continue

loopの初めてのところに戻る。

최신 파이썬 코딩 무료 강의 - 5시간만 투자하면 개발자가 됩니다! hello.py - jocoding - Visual Studio Code

RUN ▶ 🔍 ⚙️ 🏃‍♂️ ⌂

VARIABLES
Locals
special variables
mark: 25
marks: [90, 25, 67, 45, 80]
number: 2
Globals

hello.py > ...

```
1 marks = [90, 25, 67, 45, 80]
2
3 number = 0
4 for mark in marks:
5     number = number +1
6     if mark < 60:
7         continue
8     print("%d번 학생 축하합니다. 합격입니다. " % number)
```

WATCH

CALL STACK PAUSED ON STEP
<module> hello.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

2: Python Debug Consc +

r 59252 -- c:\jocoding\hello.py "
one
two
three

C:\jocoding> cd c:\jocoding && cmd /C ""C:\Program Files (x86)\Python38-32\python.exe" c:\Users\{User}\Desktop\pythonFiles\lib\python\debugpy\launcher 59452 -- c:\jocoding\hello.py "
1번 학생은 합격입니다.
2번 학생은 불합격입니다.
3번 학생은 합격입니다.
4번 학생은 불합격입니다.

2:19:28 / 4:46:23 3-2번 복습문(for문) 4번 학생은 불합격입니다.

File Edit Selection View Go Run Terminal Help hello.py - jocoding - Visual Studio Code

hello.py X speed.py

hello.py > ...

```
1 sum = 0
2 for i in range(1, 11):
3     print(i)
4 print(sum)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

3: Python

C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
55

C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
1
2
3
4
5
6
7
8
9
10
0

C:\jocoding>[

1 이상 11 미만에 있는 숫자를 의미합니다.



File Edit Selection View Go Run Terminal Help hello.py - jocoding - Visual Studio Code

hello.py X speed.py Start Debugging F5

hello.py > ...

```
1 sum = 0
2 for i in
3     sum
4 print(su
```

Run Without Debugging Ctrl+F5

Stop Debugging Shift+F5
Restart Debugging Ctrl+Shift+F5

Open Configurations
Add Configuration...

Step Over F10
Step Into F11
Step Out Shift+F11
Continue F5

Toggle Breakpoint F9
New Breakpoint >
Enable All Breakpoints
Disable All Breakpoints
Remove All Breakpoints

Install Additional Debuggers...

3: Python

C:\jocoding>"C:/Program F
55

C:\jocoding>"C:/Program F
1
2
3
4
5
6
7
8

03-3 for 문

구구단

```
>>> for i in range(2,10):
...     for j in range(1, 10):
...         print(i*j, end=" ")
...     print('')
...
2 4 6 8 10 12 14 16 18
3 6 9 12 15 18 21 24 27
4 8 12 16 20 24 28 32 36
5 10 15 20 25 30 35 40 45
6 12 18 24 30 36 42 48 54
7 14 21 28 35 42 49 56 63
8 16 24 32 40 48 56 64 72
9 18 27 36 45 54 63 72 81
```

File Edit Selection View Go Run Terminal Help • hello.py - jocoding - Visual Studio Code

RUN ▶ 🔍 ⚙️ 🌐 1

▼ VARIABLES
 Locals
 > special variables
 a: 10
 i: 2
 j: 1
 > Globals

hello.py ● speed.py ⏪ ⏴ ⏵ ⏵ ⏵ ⏵ ⏵ ⏵

hello.py > ...

```
1 a = 10      ↗ 바깥쪽 For 문
2 for i in range(2,10):   ↗ 안쪽 For 문
3     for j in range(1, 10):
4         print(i * j, end=" ")
5     print(' ')
6
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python Debug Console

C:\jocoding> cd c:\jocoding && cmd /C ""C:\Program Files (x86)\Python38-32\python.exe vscode\extensions\ms-python.python-2020.6.91350\pythonFiles\lib\python\debugpy\launch\hello.py"
2 []

▼ CALL STACK PAUSED ON STEP <module> hello.py

최선 파일 코딩 부록 강의 5시간만 투자하면 개발자가 됩니다 • hello.py - jocoding - Visual Studio Code

RUN ▶ 🔍 ⚙️ 🌐 1

▼ VARIABLES
 Locals
 > special variables
 a: 10
 i: 2
 j: 7
 > Globals

hello.py ● speed.py ⏪ ⏴ ⏵ ⏵ ⏵ ⏵ ⏵ ⏵ ⏵

hello.py > ...

```
1 a = 10
2 for i in range(2,10):
3     for j in range(1, 10):
4         print(i * j, end=" ")
5     print(' ')
6
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python Debug Console

C:\jocoding> cd c:\jocoding && cmd /C ""C:\Program Files (x86)\Python38-32\python.exe vscode\extensions\ms-python.python-2020.6.91350\pythonFiles\lib\python\debugpy\launch\hello.py"
2 4 6 8 10 12 14 16 18

▼ CALL STACK PAUSED ON STEP <module> hello.py 2:27:05 / 4:46:23 • 3-2 반복문(for문) >

최선 파일 코딩 부록 강의 5시간만 투자하면 개발자가 됩니다 • hello.py - jocoding - Visual Studio Code

RUN ▶ 🔍 ⚙️ 🌐 1

▼ VARIABLES
 Locals
 > special variables
 a: 10
 i: 3
 j: 2
 > Globals

hello.py ● speed.py ⏪ ⏴ ⏵ ⏵ ⏵ ⏵ ⏵ ⏵ ⏵

hello.py > ...

```
1 a = 10
2 for i in range(2,10):
3     for j in range(1, 10):
4         print(i * j, end=" ")
5     print(' ')
6
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python Debug Console

C:\jocoding> cd c:\jocoding && cmd /C ""C:\Program Files (x86)\Python38-32\python.exe vscode\extensions\ms-python.python-2020.6.91350\pythonFiles\lib\python\debugpy\launch\hello.py"
2 4 6 8 10 12 14 16 18
3 6 []

▼ CALL STACK PAUSED ON STEP <module> hello.py 2:27:34 / 4:46:23 • 3-2 반복문(for문) >

```
result = [num * 3 for num in a if num % 2 == 0]
```

result = [] # リスト作成

for num in a: # aから取り出してfor反復してnumに入れる
if num%2 == 0: # if条件として、2で割って残りが0
 result.append(num*3) # すなわち偶数の場合、
numを3回 ? appendリストの最後に追加する。

```
result = [x * y for x in range(2, 10) for y in range(1, 10)]
```

result = []

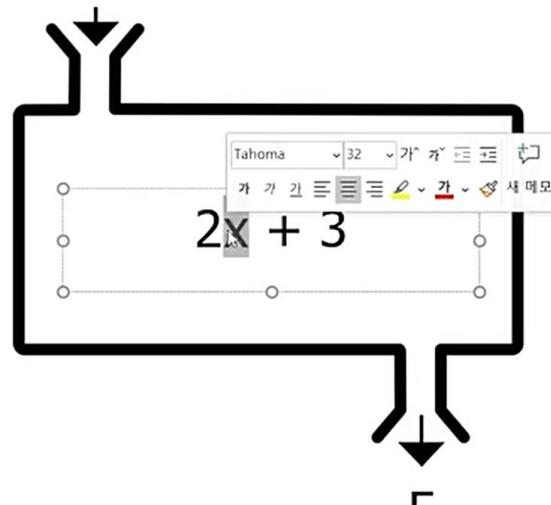
for x in range(2, 10):

for y in range(1, 10):

result.append(x * y)

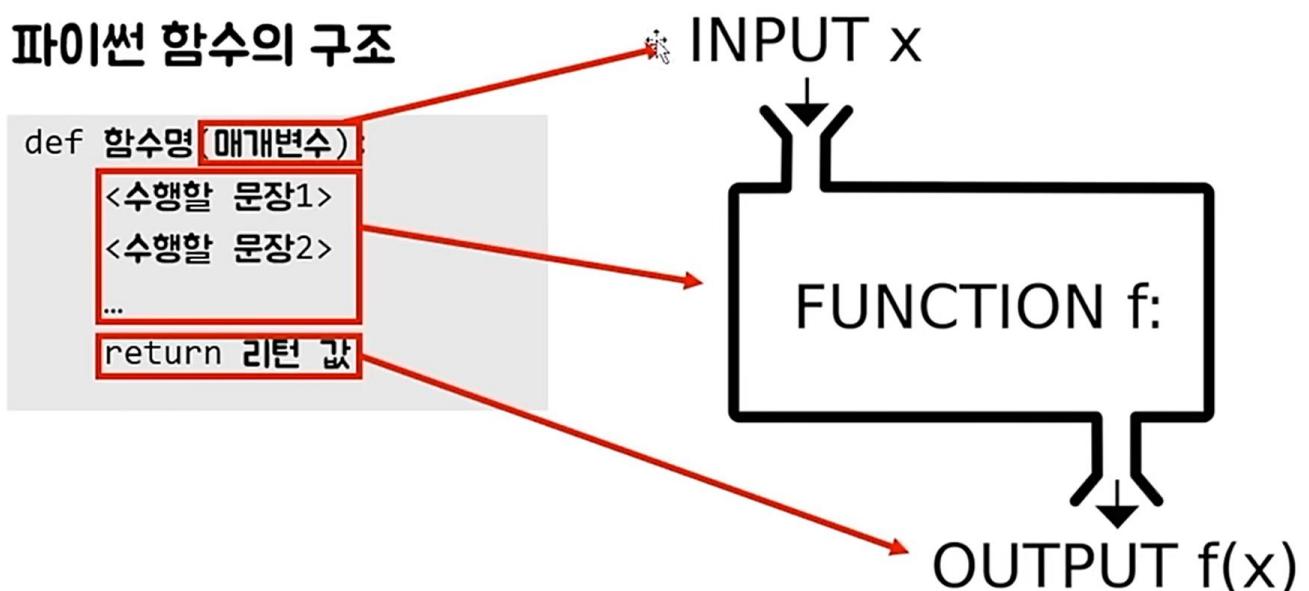
$$f(x) = 2x + 3$$

$x=1$



파이썬 함수의 구조

```
def 함수명(매개변수):
    <수행할 문장1>
    <수행할 문장2>
    ...
    return 리턴 값
```



입력, 출력 없을 수 있음

FUNCTION f:



関数はその作りによって、
入力あり、出力ありの関数と

入力なし、出力だけありの関数
入力はあるが、出力はない関数で
その性質が決まる。

```
⚡ hello.py > ...
```

```
1 myList = [1,2,3]
2 print(myList.pop())
3
```

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
3
```

POPはリストの最後の物を取り出す関数（出力あり）
()のため、inputはないが、出力はある関数。

->だから、print(mylist.pop())> 3

```
⚡ hello.py > [o] myList
```

3: int

```
1 myList = [1,2,3]
2 print(myList.append(4))
3
```

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
None
```

appendはリストの最後に入力物を追加する関数だが、
出力はない関数

(4)と入力はしたが、print(mylist.append(4)) しても出力はないので
なにも出力されずNONE

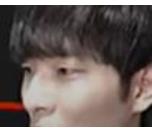
Print(Mylist) -> 1 , 2 , 3 , 4

File Edit Selection View Go Run Terminal Help • hello.py - jocoding - Visual Studio Code

hello.py ●

hello.py > sum_many > [e] c

```
1 def sum_many(a,b,c,d,e):  
2     sum = 0  
3     for i in args:  
4         sum = sum + i  
5     return sum  
6  
print(sum_many(1,2,3))  
Error: sum_many() takes 2 positional arguments but 3 were given
```



入力がいくら入力されるかが
決まらない場合、

*args

File Edit Selection View Go Run Terminal Help hello.py - jocoding - Visual Studio Code

hello.py ●

hello.py > sum_many > [e] args

```
1 def sum_many(*args):  
2     sum = 0  
3     for i in args:  
4         sum = sum + i  
5     return sum  
6 print(sum_many(1,2,3))  
7
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py  
1 2 3  
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py  
1 2 3  
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py  
Traceback (most recent call last):
```

Dictionaryの場合、 **kwargs

hello.py ●

hello.py > print_kwargs

```
1 def print_kwargs(**kwargs): (오타 수정) k는 key라서  
2     for k in kwargs.keys(): 값을 얻으려면 이렇게 써야함  
3         if(k == "name"): kwargs[k]  
4             print("당신의 이름은 :" + k)  
5     print(print_kwargs(name="int 조수", b="2"))  
6
```

hello.py > ...

```
1 def say_myself(name, old, man=True):
2     print("나의 이름은 %s 입니다." % name)
3     print("나이는 %d살입니다." % old)
4     if man:
5         print("남자입니다.")
6     else:
7         print("여자입니다.")
8 say_myself("라이유튜브", 20, False)
9
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
나의 이름은 라이유튜브 입니다.
나이는 20살입니다.
남자입니다.
```

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
나의 이름은 라이유튜브 입니다.
나이는 20살입니다.
```

変数の適用範囲にも注意

The screenshot shows a Visual Studio Code interface with a Python file named 'hello.py'. The code is as follows:

```
1 a = 1
2 def vartest(a):
3     2 a = a +1
4
5 vartest(a)
6 print(a)
7
```

The line '2 a = a +1' is highlighted with a yellow box and has a red number '2' next to it. The output window shows the result '1'. Below the code editor is a diagram titled 'Frames' and 'Objects' illustrating the variable scope.

Frames

- Global frame:
 - a → int 1
 - vartest → function vartest(a)
- vartest:
 - a → int 2
 - Return value → NoneType None

Objects

- int 1
- function vartest(a)
- int 2
- NoneType None

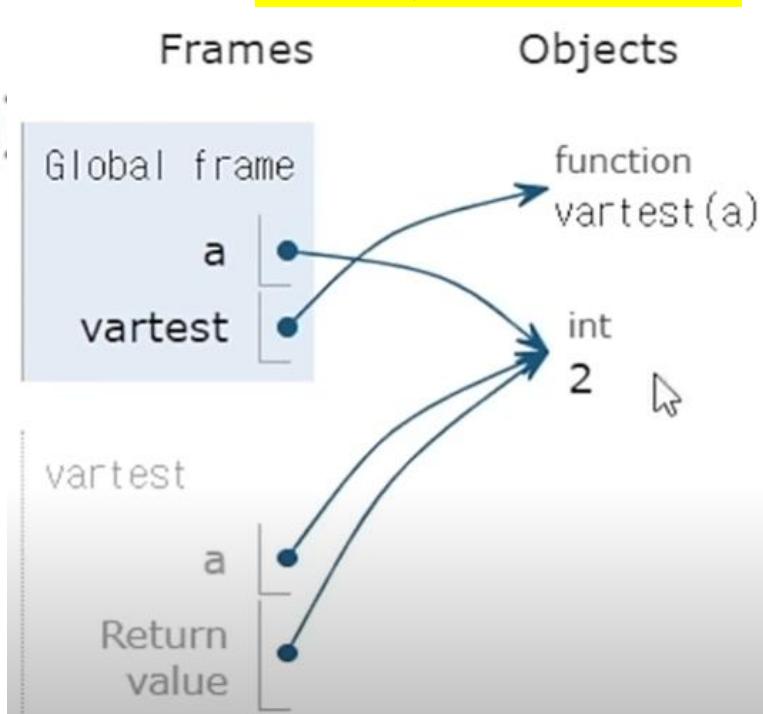
A young man is visible in the bottom right corner of the screen.

global変数

```
1 a = 1
2 def vartest(a):
3     2 a = a +1 local変数
4     return a 2
```

※参考
Global aと定義すると
どこからでもglobal変数に
なるので、必要に応じて定義

```
5 2 1
6 a = vartest(a)
7 print(a) 2
```



lambdaは特集で lambda input : outputを1列で関数の定義可能

hello.py > ...

```
1 # def add(a,b):  
2 #     return a+b  
3  
4 add = lambda a, b: a+b
```

The screenshot shows the Visual Studio Code interface with a dark theme. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a tab for 'hello.py - jocoding - Visual Studio Code'. On the left is a sidebar with icons for file operations like Open, Save, Find, and Refresh. The main editor area contains the following Python code:

```
File Edit Selection View Go Run Terminal Help hello.py - jocoding - Visual Studio Code  
hello.py < X  
hello.py > [o] myList  
1  
2 myList = [lambda a, b: a+b, lambda a, b: a*b]  
3  
4 print(myList[0](1,2))  
5
```

04-2 사용자 입력과 출력

input의 사용

```
>>> a = input()
Life is too short, you need python
>>> a
'Life is too short, you need python'
>>>
```

Input()はuserのinputを待つプログラミングの内蔵関数

```
hello.py > ...
1 number = input("숫자를 입력하세요: ")
2
3 print(a)
```

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
숫자를 입력하세요: 10
10

C:\jocoding>|
```

04-2 사용자 입력과 출력

print 문

```
>>> print("life" "is" "too short")
```

```
lifeistoo short
```

```
>>> print("life"+"is"+"too short")
```

```
lifeistoo short
```

```
>>> print("life", "is", "too short")
```

```
life is too short
```

```
>>> for i in range(10):
```

```
...     print(i, end=' ')
```

```
...
```

```
0 1 2 3 4 5 6 7 8 9
```

04-3 파일 읽고 쓰기

파일 생성하기

```
f = open("새파일.txt", 'w')
f.close()
```

파일열기모드	설명
r	읽기모드 - 파일을 읽기만 할 때 사용
w	쓰기모드 - 파일에 내용을 쓸 때 사용
a	추가모드 - 파일의 마지막에 새로운 내용을 추가 시킬 때 사용

The screenshot shows the Visual Studio Code interface. The title bar says "hello.py - jocoding - Visual Studio Code". The Explorer sidebar shows "hello.py" and "새파일.txt". The Editor pane contains the following Python code:

```
1 f = open("새파일.txt", 'w')
2 for i in range(1, 11):
3     data = "%d번째 줄입니다.\n" % i
4     f.write(data)
5 f.close()
```

The screenshot shows the Visual Studio Code interface with the " 새파일.txt" tab selected. The file contains the following text:

```
1 1번째 줄입니다.
2 2번째 줄입니다.
3 3번째 줄입니다.
4 4번째 줄입니다.
5 5번째 줄입니다.
```

The screenshot shows the Visual Studio Code interface. The title bar says "hello.py - jocoding - Visual Studio Code". The Explorer sidebar shows "hello.py" and "새파일.txt". The Editor pane contains the following Python code:

```
1 f = open("새파일.txt", 'w', encoding="UTF-8")
2 for i in range(1, 11):
3     data = "%d번째 줄입니다.\n" % i
4     f.write(data)
5 f.close()
```

The screenshot shows the Visual Studio Code interface with the " 새파일.txt" tab selected. The file contains the following text:

```
1 1번째 줄입니다.
2 2번째 줄입니다.
3 3번째 줄입니다.
4 4번째 줄입니다.
5 5번째 줄입니다.
6 6번째 줄입니다.
7 7번째 줄입니다.
8 8번째 줄입니다.
9 9번째 줄입니다.
10 10번째 줄입니다.
```

04-3 파일 읽고 쓰기

readline() 함수

```
f = open("C:/Python/새파일.txt", 'r')
line = f.readline()
print(line)
f.close()
```

```
f = open("C:/Python/새파일.txt", 'r')
while True:
    line = f.readline()
    if not line: break
    print(line)
f.close()
```


Immutable

(정수, 실수, 문자열, 튜플)

변하지 않는 자료형

```
a = 1
def vartest(a):
    a = a + 1

vartest(a)
print(a)
```

Mutable

(리스트, 딕셔너리, 집합)

변할 수 있는 자료형

```
b = [1, 2, 3]
def vartest2(b):
    b.append(4)
```

```
vartest2(b)
print(b)
```

▶▶▶

파이썬 라이브 텁크

[Pythontutor.com/live.html#mode=edit](https://pythontutor.com/live.html#mode=edit)

▶▶▶

최신 파이썬 코딩 무료 강의 - 5시간만 투자하면 개발자가 됩니다



조코딩 JoCoding
チャンネル登録者数 59.6...

メンバーになる

登録済み

2万

▼

共有

オフライン

...

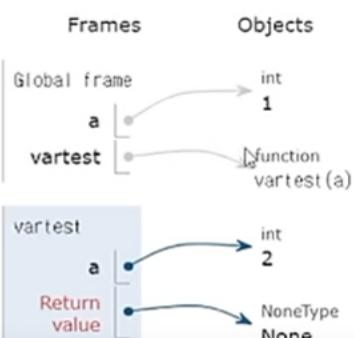
This mode is experimental. Use the [regular Python Tutor](#) to get live help.

一列ずつ実行可能 !

```
1 ##### Immutable #####
2 a = 1
3 def vartest(a):
4     a = a + 1
5 vartest(a)
6 print(a)
7 #####
```

(drag lower right corner to resize code editor)

Print output (drag lower right corner to resize)

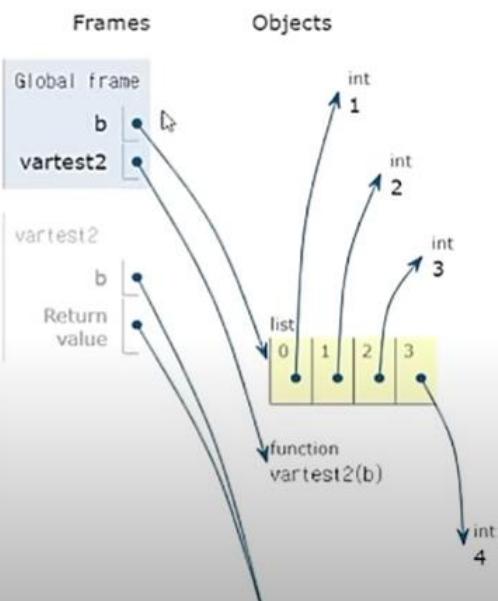


This mode is experimental. Use the [regular Python Tutor](#) to get live help and use more features.

```
1 b = [1, 2, 3]
2 def vartest2(b):
3     b.append(4)
4 vartest2(b)
5 print(b)
6
```

(drag lower right corner to resize code editor)

Print output (drag lower right corner to resize)



line that just executed

2:19:20 / 4:46:22 - Immutable vs Mutable

반복되는 변수 & 메서드(함수)를 미리 정해놓은 틀(설계도)

Class

05-1 파이썬 프로그래밍의 핵심, 클래스

클래스는 도대체 왜 필요한가?

```
result = 0
def add(num):
    global result
    result += num
    return result

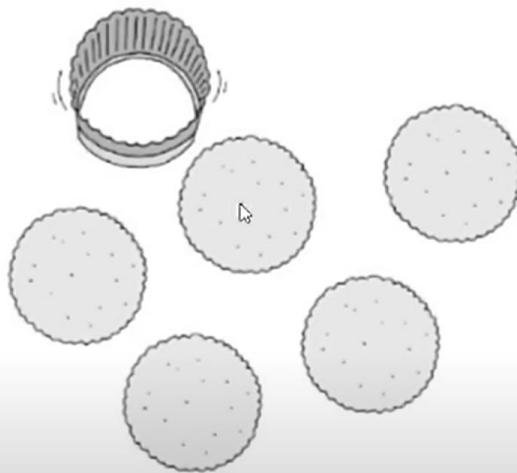
print(add(3))
print(add(4))
```

```
hello.py > ...
1  result1 = 0
2  result2 = 0
3
4  def add1(num):
5      global result1
6      result1 += num
7      return result1
8
9  def add2(num):
10     global result2
11     result2 += num
12     return result2
13
14  print(add1(3))
15  print(add1(4))
16  print(add2(3))
17  print(add2(7))
```

```
File Edit Selection View Go Run Terminal Help
hello.py > ...
1  class Calculator:
2      def __init__(self):
3          self.result = 0
4
5      def add(self, num):
6          self.result += num
7          return self.result
8
```

実行して計算機 1 を作成
実行して計算機 2 を作成
class 設計図を基にinstanceを作成可能

```
9  cal1 = Calculator()
10 cal2 = Calculator()
11
12 print(cal1.add(3))
13 print(cal1.add(4))
14 print(cal2.add(3))
15 print(cal2.add(7))
```



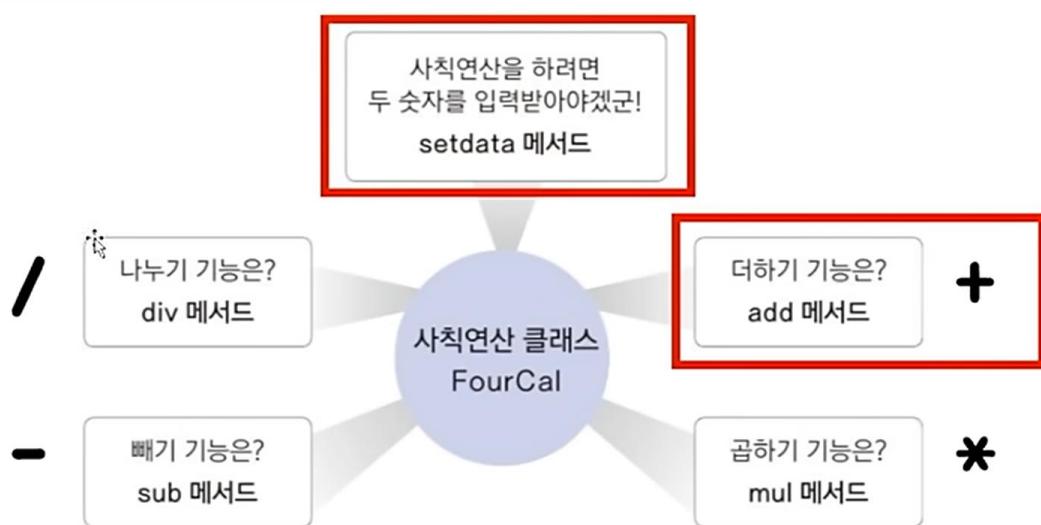
과자 틀(클래스)과 이 틀로 찍어 만든 과자(객체)

과자 틀(클래스), 과자(객체)

05-1 파이썬 프로그래밍의 핵심, 클래스

사칙연산 클래스 만들기(+,-,*,/)

어떤 클래스를 만들지 대충 그림을 그려 보자



File Edit Selection View Go Run Terminal Help hello.py - jocoding - Visual Studio Code

```
hello.py
1 class FourCal:
2     def setdata(self, first, second):
3         self.first = first
4         self.second = second
5
6     a = FourCal()
7     a.setdata(1,2)
8     print(a.first)
9     print(a.second)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python + ⌂ ↗

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
1
2
C:\jocoding>[]
```

File Edit Selection View Go Run Terminal Help hello.py - jocoding - Visual Studio Code

```
1 class FourCal:
2     def setdata(self, first, second):
3         self.first = first
4         self.second = second
5     def add(self):
6         result = self.first + self.second
7         return result
8
9 a = FourCal()
10 a.setdata(4, 2)
11 print(a.add())
12
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
```

File Edit Selection View Go Run Terminal Help hello.py - jocoding - Visual Studio Code

VARIABLES

- Locals
 - self: <__main__.FourCal object ->
 - > special variables
 - > function variables
 - first: 4
 - second: 2
- Globals

```
1 class FourCal:
2     def setdata(self, first, second):
3         self.first = first
4         self.second = second
5     def add(self):
6         result = self.first + self.second
7         return result
8
9 a = FourCal()
10 a.setdata(4, 2)
11 print(a.add())
12
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 3: Python Debug Consc

```
Microsoft Windows [Version 10.0.18362.959]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\jocoding> cmd /C ""C:\Program Files (x86)\Python38-32\python.exe" c:\Users\DongKeunJo\.vscode\ext\ms-python.python-2020.7.96456\pythonFiles\lib\python\debugpy\launcher 52543 -- c:\jocoding\hello.py"
```

File Edit Selection View Go Run Terminal Help ● hello.py - jocoding - Visual Studio Code

hello.py

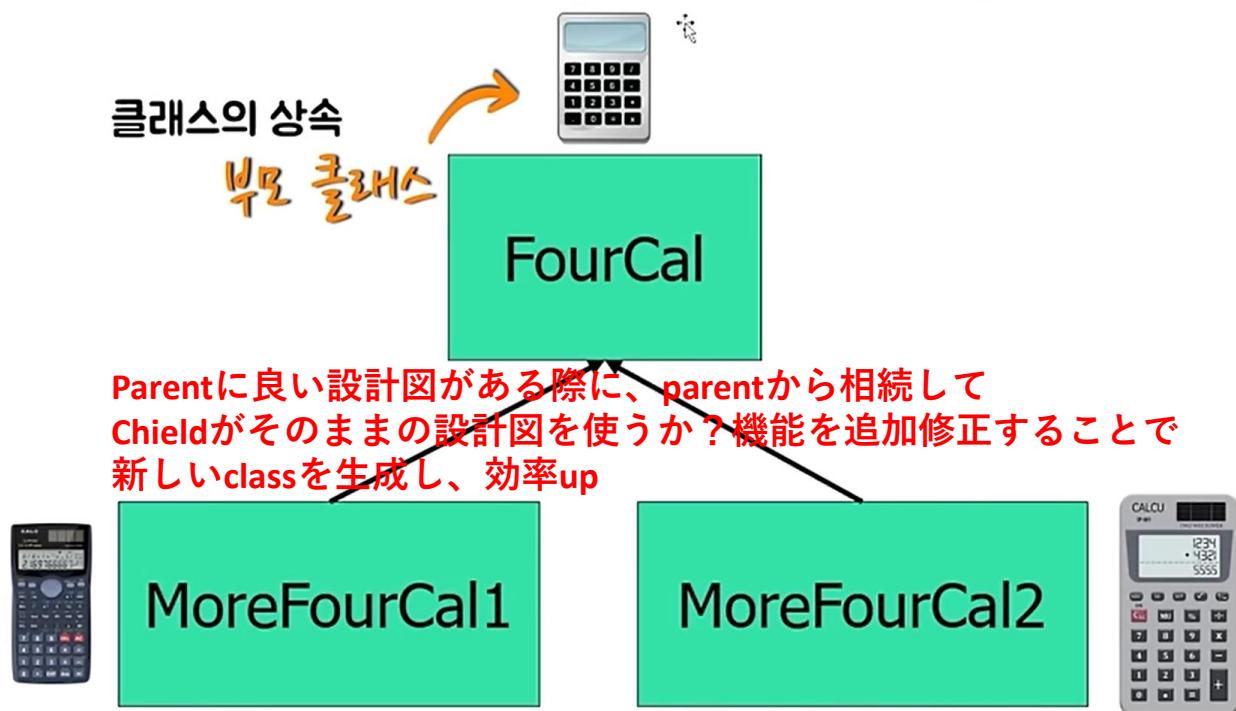
```
1 class FourCal:
2     def __init__(self, first, second):
3         self.first = first
4         self.second = second
5     def setdata(self, first, second):
6         self.first = first
7         self.second = second
8     def add(self):
9         result = self.first + self.second
10        return result
11
12 a = FourCal()
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL 3: Python Debug Consc

```
Microsoft Windows [Version 10.0.18362.959]
(c) 2019 Microsoft Corporation. All rights reserved.
```

初期化特別関数

05-1 파이썬 프로그래밍의 핵심, 클래스



```
File Edit Selection View Go Run Terminal Help
hello.py - jocoding - Visual Studio Code
hello.py MoreFourCal
1 class FourCal:
2     def __init__(self, first, second):
3         self.first = first
4         self.second = second
5     def setdata(self, first, second):
6         self.first = first
7         self.second = second
8     def add(self):
9         result = self.first + self.second
10        return result
11
12 class MoreFourCal(FourCal):
13     pass
14
15
```

```
11 class MoreFourCal(FourCal):
12     pass
13
14 a = MoreFourCal(4, 2)
15 print(a.add())
16
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py

6

C:\jocoding>[]

File Edit Selection View Go Run Terminal Help hello.py - Visual Studio Code

최신 파이썬 코딩 무료 강의 - 5시간만 투자하면 개발자가 됩니다

hello.py > ...

```
1 class FourCal:
2     def __init__(self, first, second):
3         self.first = first
4         self.second = second
5     def setdata(self, first, second):
6         self.first = first
7         self.second = second
8     def add(self):
9         result = self.first + self.second
10    return result
11 class MoreFourCal(FourCal):
12     def pow(self):
13         result = self.first ** self.second
14         return result
15
16 a = MoreFourCal(4, 2)
```

parentにはない...2じょう関数を
追加したり、
Parentの関数の内容を修正も可能

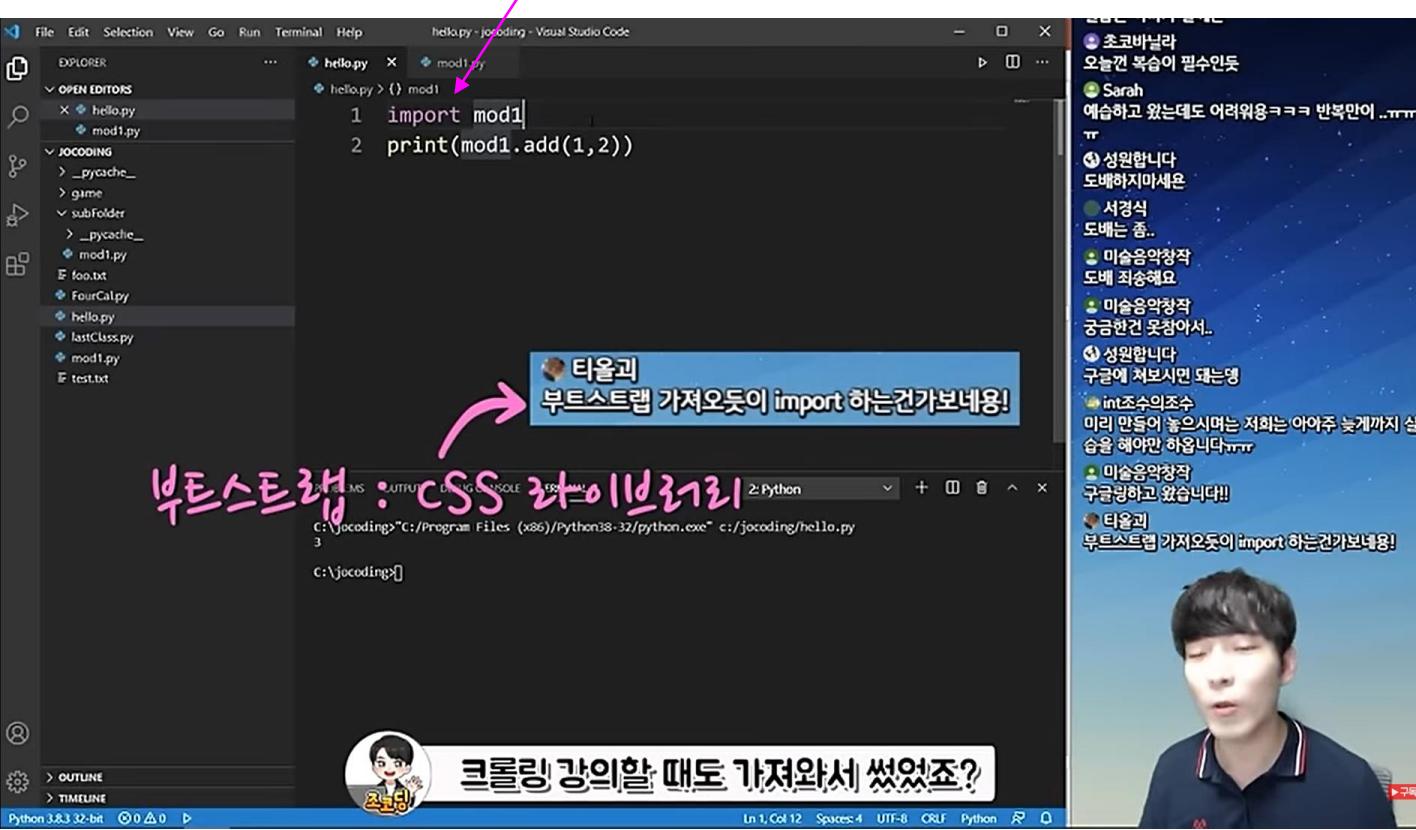
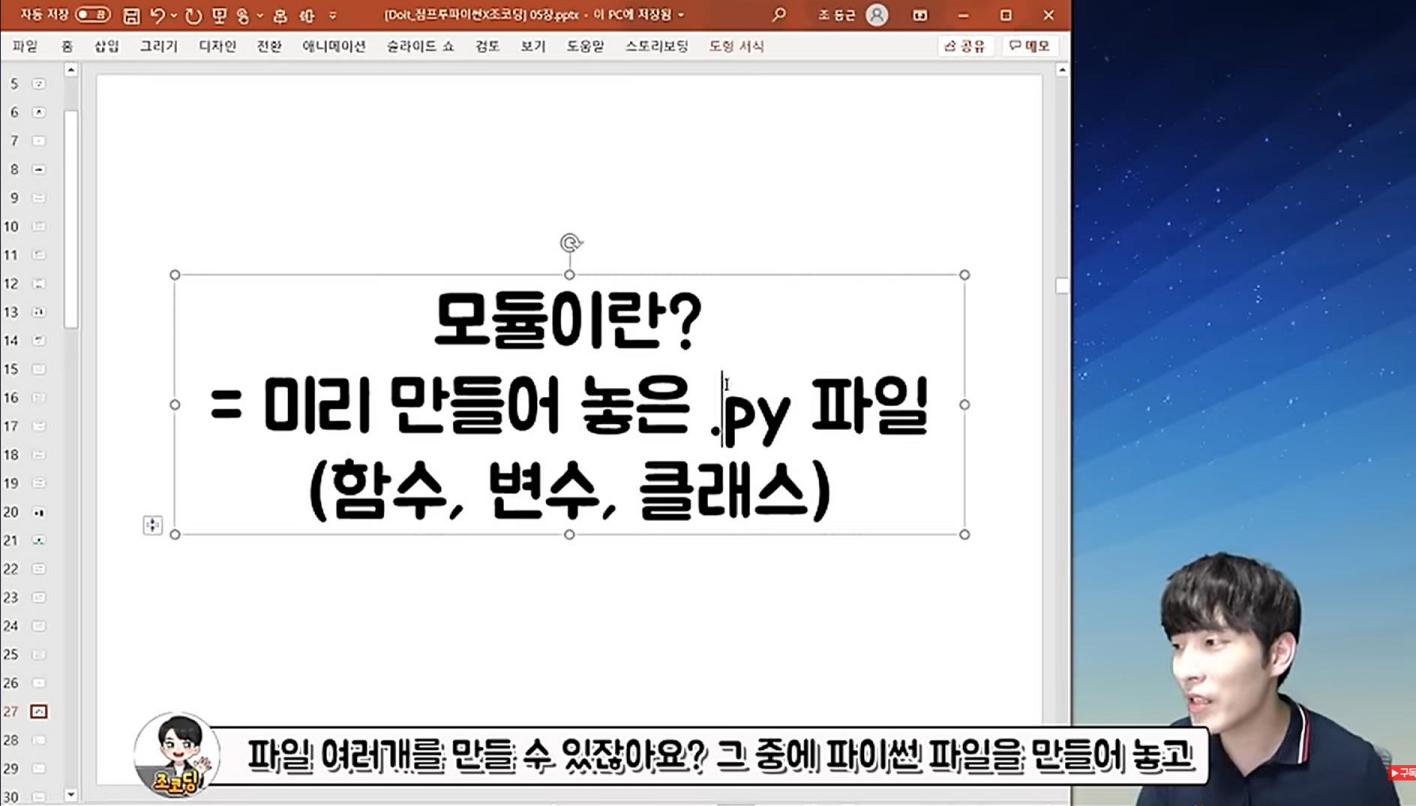
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python

hello.py > FourCal

```
5     def setdata(self, first, second):
6         self.first = first
7         self.second = second
8     def add(self):
9         result = self.first + self.second
10    return result
11     def mul(self):
12         result = self.first * self.second
13         return result
14     def sub(self):
15         result = self.first - self.second
16         return result
17     def div(self):
18         result = self.first / self.second
19         return result
```

hello.py > ...

```
13         return result
14     def sub(self):
15         result = self.first - self.second
16         return result
17     def div(self):
18         result = self.first / self.second
19         return result
20 class SafeFourCal(FourCal):
21     def div(self): Parentの関数の内容を修正も可能
22         if self.second == 0:
23             return 0
24         else:
25             return self.first / self.second
26
```

File Edit Selection View Go Run Terminal Help mod1.py - jocoding - Visual Studio Code

EXPLORER ... hello.py mod1.py

OPEN EDITORS 1 UNSAVED

hello.py mod1.py

JOCODING

pycache game subFolder _pycache_ mod1.py

mod1.py > add

```
1 #mod1.py
2 def add(a,b):
3     return a + b
4 def aasdfasdf(a,b):
5     return a + b
```

moduleに関数がたくさんある際に全部importして使う方法と、

최신파이썬 코딩 무료 강의 - 5시간만 투자하면 개발자가 됩니다! mod1.py - jocoding - Visual Studio Code

EXPLORER ... hello.py ... Esc を押します

OPEN EDITORS

hello.py mod1.py

JOCODING

pycache

mod1.py > ...

```
1 from mod1 import add
```

必要な関数だけをimportして軽くして使うことも可能。ここでは、add関数のみをImportして活用する事例

hello.py x mod1.py

hello.py > add mod1: mod1

```
1 from mod1 import add
2
3 print(add(1,2))
```

```
hello.py x mod1.py x
mod1.py > ...
1 #mod1.py
2 def add(a,b):
3     return a + b
4
5 def sub(a,b):
6     return a - b
7
8 # if __name__ == "__main__":
9 print(add(1,4))
10 print(add(4,2))
```

if __name__ == "__main__":
Moduleをimportするだけで
希望していない内容まで実行される

ことがあるので、
Moduleを作る際に、例外または条件
などの内容を定義しておくとよい

```
hello.py x mod1.py
hello.py > ...
1 import mod1
2
```

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
5
6
```

Underbar2つ init Underbar2つ

init

Underbar2つ name Underbar2つ 等は、特殊な内部関数？

```
File Edit Selection View Go Run Terminal Help • mod1.py - jocoding - Visual Studio Code
EXPLORER ... hello.py mod1.py •
OPEN EDITORS 1 UNSAVED
mod1.py > ...
1 #mod1.py
2 def add(a,b):
3     return a + b
4
5 def sub(a,b):
6     return a - b
7
8 if __name__ == "__main__":
9     print(add(1,4))
10    print(add(4,2))
```

すなわち、
Moduleをimportする側 main 側で
ないと、if以下を実行しないように
例外定義しておく

05-2 모듈



```
sys.path.append
```

```
import sys
print(sys.path)
[ '', 'C:\\Windows\\SYSTEM32\\python35.zip',
'c:\\Python35\\DLLs',
'c:\\Python35\\lib', 'c:\\Python35',
'c:\\Python35\\lib\\site-packages' ]
```

```
import sys
sys.path.append("C:\\jocoding\\subFolder")
import mod1
print(mod1.add(3,4))
```



주: 경로에 대해 가져온다.

2021.4.16.22

```
File "c:/jocoding/hello.py", line 1, in <module>
    import mod1
ModuleNotFoundError: No module named 'mod1'
```

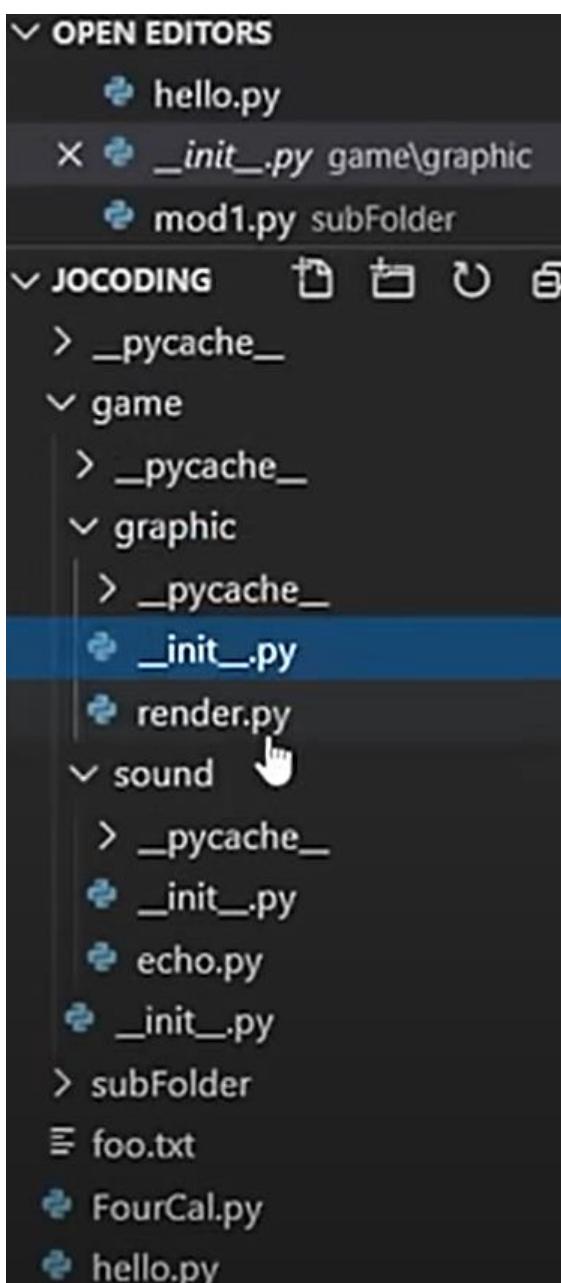
hello.py > ..

```
1 import sys
2 sys.path.append("C:\\jocoding\\subFolder")
3 import mod1
4 print(mod1.add(3,4))
5
```

패키지란?

= 모듈 여러 개 모아놓은 것

I



05-3 패키지

가상의 game 패키지 예

```
game/
    __init__.py
    sound/
        __init__.py
        echo.py
    graphic/
        __init__.py
        render.py
```

hello.py > ...

```
1 import game.sound.echo
2 game.sound.echo.echo_test()
```

05-4 예외처리

try .. else



```
try:  
    f = open('foo.txt', 'r')  
except FileNotFoundError as e:  
    print(str(e))  
else:  
    data = f.read()  
    print(data)  
    f.close()
```

try一旦実行

Except try失敗すると、

Else:try成功すると、

try:

#오류가 발생할 수 있는 구문

except Exception as e:

#오류 발생

else:

#오류 발생하지 않음

finally:

#무조건 마지막에 실행

hello.py > ...

```
1 f = open('foo.txt', 'w')  
2 try:  
3     # 무언가로 스해합니다  
4     dat class Exception(*args: object)  
5     pri Common base class for all non-exit exceptions.  
6 except Exception as e:  
7     print(e)  
8 finally:  
9     f.close()
```


hello.py x

hello.py

```
1 print(dir([1,2,3]))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

2: Python +

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
3
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__get__', '__getitem__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']
```

05-5 내장함수

filter => 함수를 통하여 참인 것만 둘려줌

```
def positive(x):
    return x > 0
I
a = list(filter(positive, [1, -3, 2, 0, -5, 6]))
print(a)

a = list(filter(lambda x: x > 0, [1, -3, 2, 0, -5, 6]))
print(a)
```

hello.py > ...

```
1 def positive(x):
2     return x > 0
3
4 a = list(filter(positive, [1, -3, 2, 0, -5, 6]))
5 print(a) Filter... filtering by what...positive and etc.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
C:\jocoding>"C:/Program Files (x86)/Python38-32/python.exe" c:/jocoding/hello.py
[1, 2, 6]
```

05-5 내장함수

map => 각 요소가 수행한 결과를 둘려줌

```
def two_times(x): return x*2
a = list(map(two_times, [1, 2, 3, 4]))
print(a)

a = list(map(lambda a: a*2, [1, 2, 3, 4]))
print(a)
```

05-6 외장함수

pickle

```
import pickle  
f = open("test.txt", 'wb')  
data = {1: 'python', 2: 'you need'}  
pickle.dump(data, f)  
f.close()
```

```
import pickle  
f = open("test.txt", 'rb')  
data = pickle.load(f)  
print(data)  
{2:'you need', 1:'python'}
```


The screenshot shows a Microsoft Edge browser window with a presentation slide titled "프론트엔드, 백엔드, 데이터, 임베디드, 보안" (Frontend, Backend, Data, Embedded, Security) in Korean. The slide includes icons for each category: a smartphone for Frontend, a server tower for Backend, a database for Data, the Windows logo for Embedded, and a padlock for Security. Below the categories, there are five sections with corresponding icons: "눈에 보이는 화면" (Visible Screen) with a laptop icon, "안보이는 정보 처리" (Hidden Information Processing) with a server icon, "데이터 처리" (Data Processing) with a database icon, "하드웨어와 밀접" (Close to Hardware) with a hard drive icon, and "보안 관리" (Security Management) with a lock icon. At the bottom of the slide, there is a large Korean title: "학습 시기면 머신러닝, 인공지능" (When learning, machine learning, AI). The browser's address bar shows the file path: "자료실_클개발_5주_완성_1주자.pptx". The status bar at the bottom indicates the video player is at 1:23 / 12:28, showing a preview of the slide content.

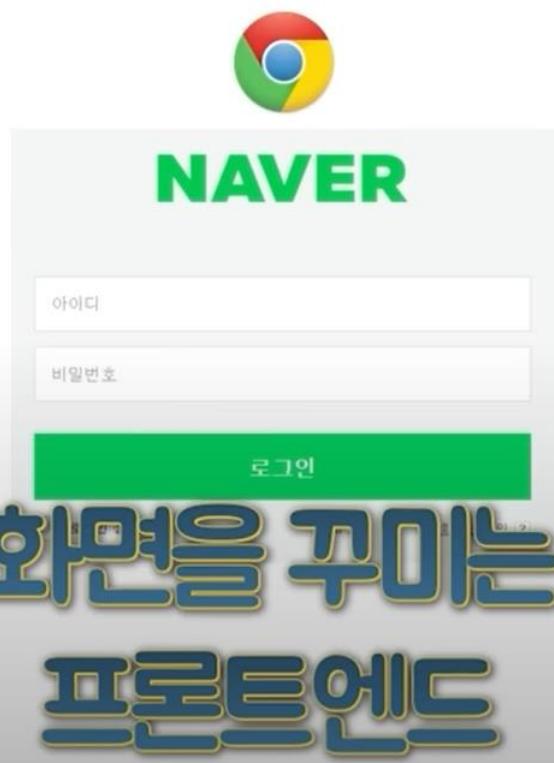
코딩 분야 소개와 가장 먼저 공부해야 할 것과 공부 순서



- 익숙함
 - 난이도 쉽다
 - 결과물 바로 확인
 - 자랑하기 좋음 (url, 모든 디바이스)
 - 이후 다른 개발 쉬움

프론트엔드

(클라이언트)



백엔드

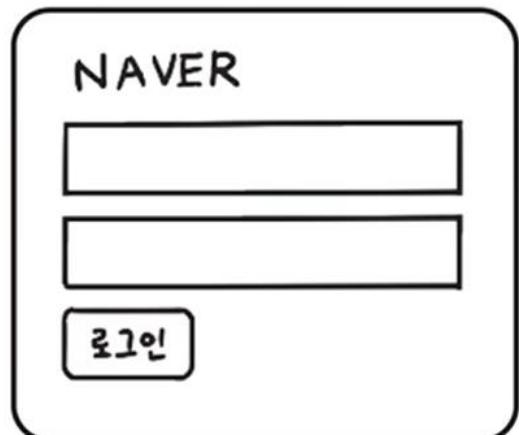
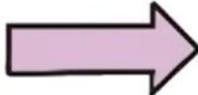
(서버)



NAVER



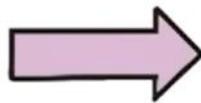
<제목>NAVER</제목>
<입력상/>
<입력상/>
<버튼>로그인</버튼/>



HTML



```
<제목>NAVER</제목>
<입력창/>
<입력창/>
<버튼>로그인<버튼/>
```

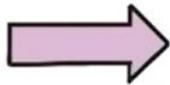


NAVER

CSS



제목 초록색
 배경 회색
 입력창 흰색
 버튼 초록색
 ...



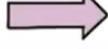
NAVER

NAVER

로그인



아이디 미입력 시 표시
 로그인 성공 시, 페이지 전환
 로그인 실패 시, 화면에 표시



NAVER

NAVER

로그인 상태유지

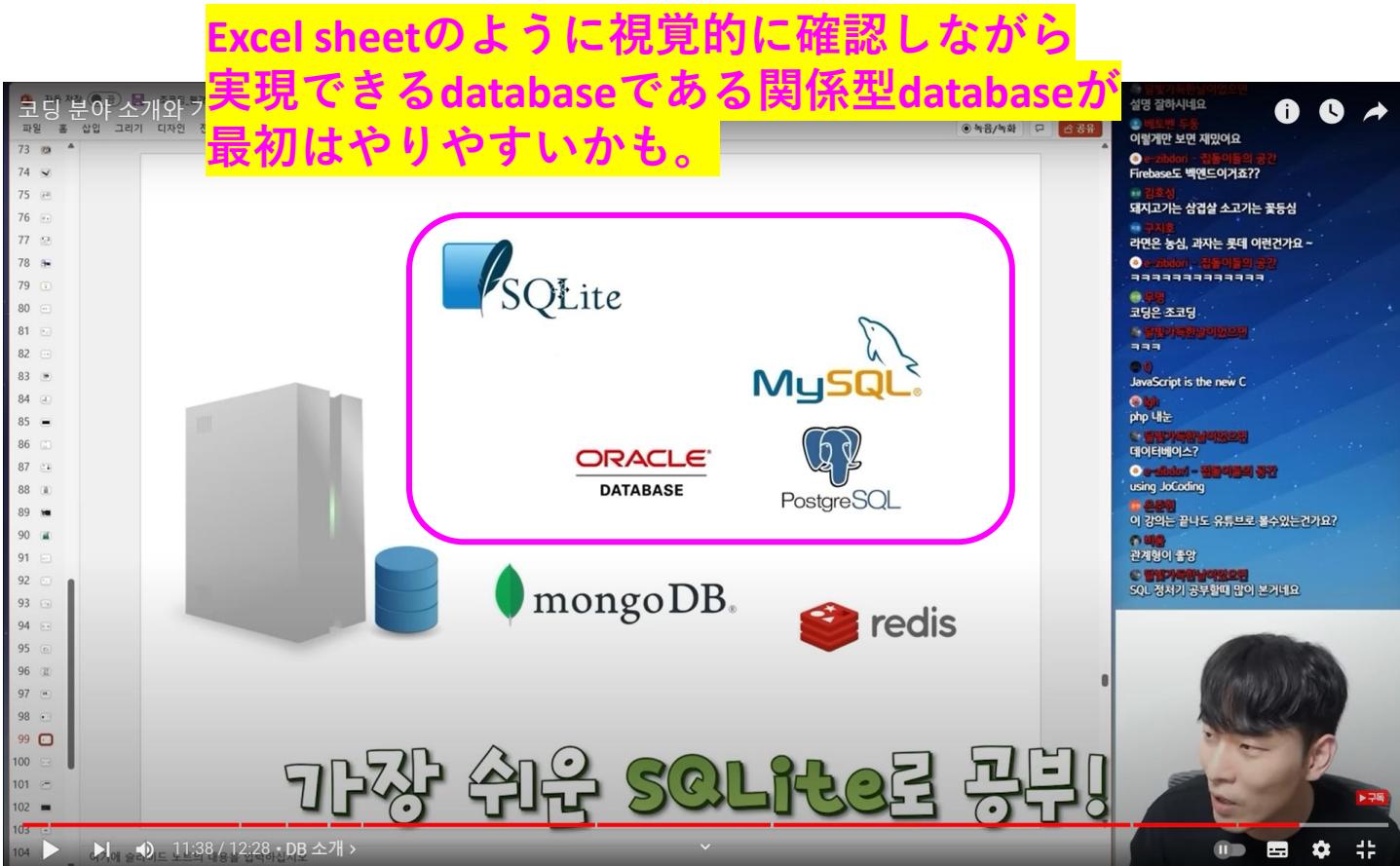
IP보안ON

자바스크립트를 추가하여 로그인 기능을 완성하기

로직이 있는 파파스크립트



Backendはframeworkを活用して処理か？」
各言語毎にframeworkがあるので、それでbackend処理
Pythonにはdjangoがあり、
Javascriptにはnodeというframeworkがある。



조코딩 웹 개발 1주차 출석 체크

출석체크

Name

유튜브 닉네임

Notes

공부 다짐, 하고 싶은 말

Submit

조코딩 웹 개발 5주 완성

1주차 출석체크

1주차 과제

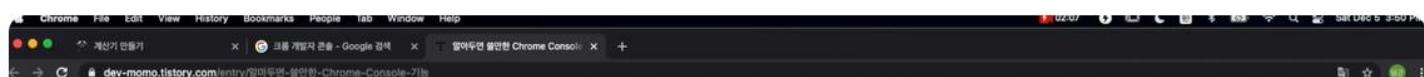
Add or import

Views Grid view Hide fields Filter Group Sort Color Share view

	A Name	Notes	
1	조코딩	화이팅!!!	
2	YJ Kim	열심히 배워보겠습니다!	
3	김재상	이번 웹페이지 강의를 통해 ...	
4	엄현성	열심히 배워서 꼭 웹 개발...	
5	whw가이	화이팅!	
6	황대표	가즈아~~~~~ 웹 마스터 ...	
7	리차드김	시작이 반이다! ㅋ	
8	김용근	아무것도 모르고 들왔어요....	
9	전진우	웹 업 모두 제대로 배우고 ...	
10	e-zibdori - 집들이들의 공간	저만의 완벽한 웹서비스 하...	
11	Bossmob	Nice!	
12	정스카이	백엔드 넘 궁금해요 감사하...	
13	강예인	앞으로 5주동안 꾸준히 참...	
14	매일매일꾸준히	열심히 하겠습니다!	
15	류영잔	코딩 공부 화이팅!!	
16	중3취미생활	열심히하고싶다	



개발자콘솔



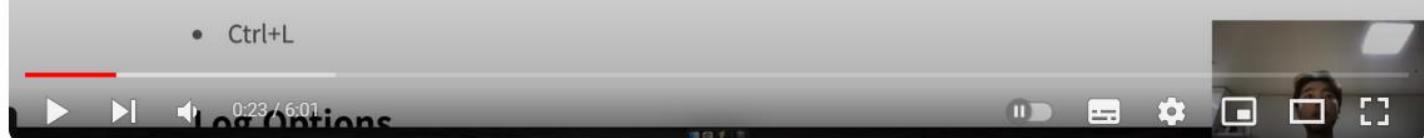
나는 개발할 때 Chrome 개발자도구의 console을 굉장히 자주 사용하는데(대부분의 개발자들이 그럴 것이라 생각한다), 오늘은 크롬 개발자도구의 편리한 기능과 잘 쓰진 않지만 알아두면 좋은 기능들을 한 번 정리해 보고자 한다.

Console Open

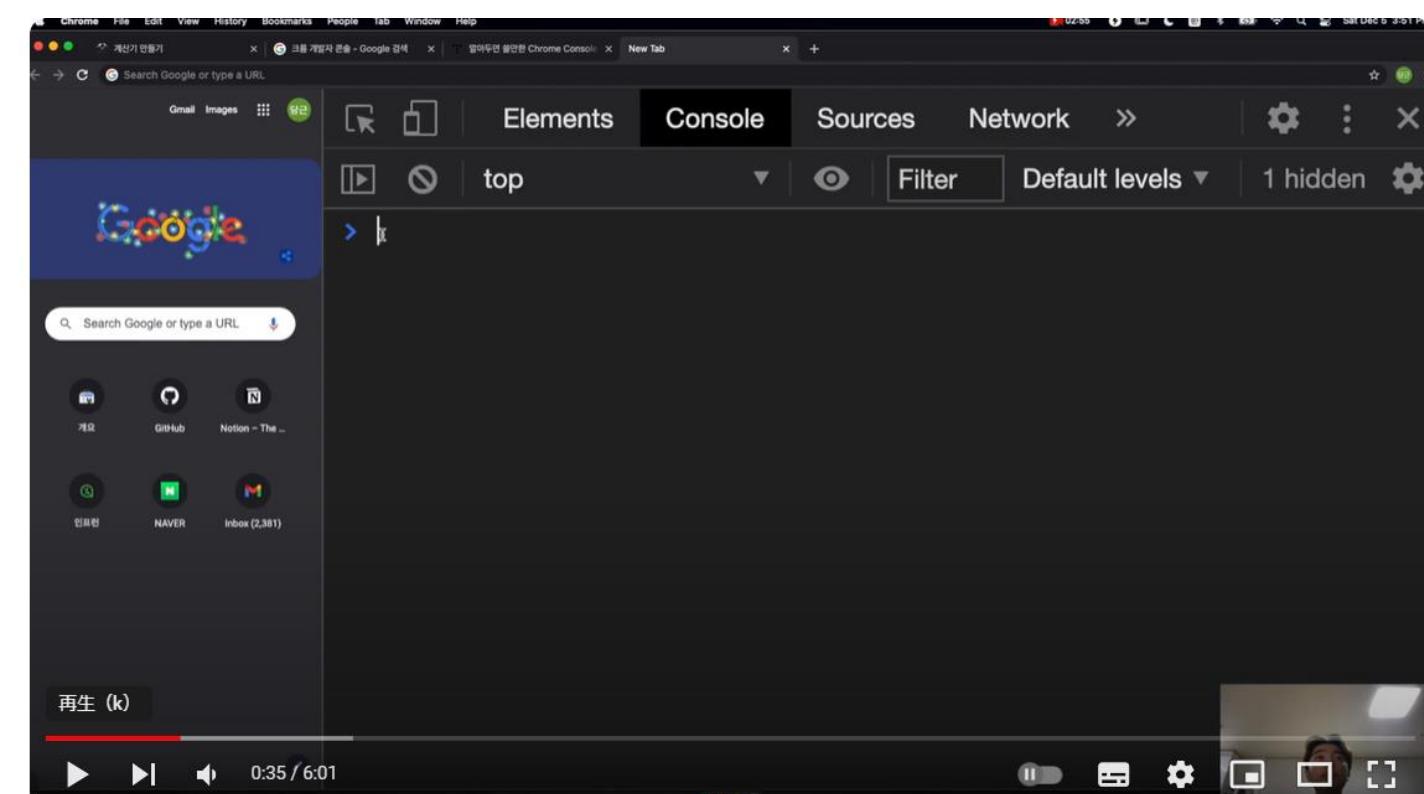
- Windows / Linux: Ctrl+Shift+J
- Mac: Cmd+Opt+J
- 개발자 도구가 열린 상태: ESC

Console Clear

- clear() 입력
- Ctrl+L



자바스크립트 함수 6분 만에 이해시켜드림 (안되면 200% 환불)



자바스크립트 함수 6분 만에 이해시켜드림 (안되면 200% 환불)



Google Chrome File Edit View History Bookmarks Profiles Tab Window Help

JavaScript란 무엇인가? - Pres... 2023년도 개발자 서비스 현... New Tab

canva.com/design/DAF-jM0dZzs/a4oa0GyyHnp477KD5ARw/jw/edit

File Resize & Magic Switch

Animate 5.0s Position

0 100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 1900

Design Elements Text Brand Uploads Draw Projects Apps

배열

하루 3분 개쉬운 자바스크립트 (강의)

하루 3분
개쉬운 JavaScript

2024년
JS

이메일 구독시
강의 구매시 사용 가능한
에피코딩 3,000 포인트를
지급해드립니다! 🎁

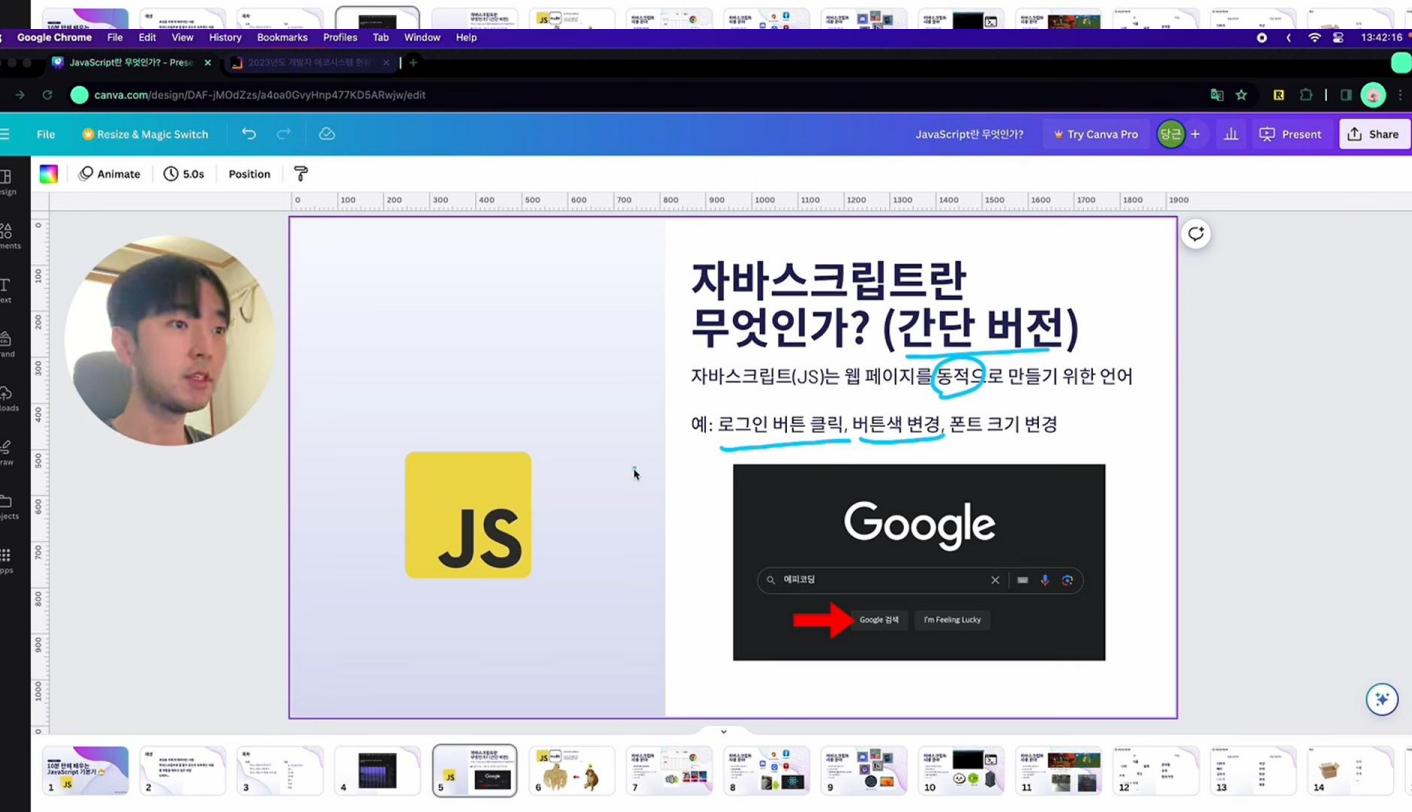
(링크는 댓글에)

Notes Duration Timer

Page 22 / 22 59%



백엔드 공부 첫 3개월은 이렇게 하세요 제발 (자바 공부 금지 ✗)



The screenshot shows a Canva editor interface with a purple-themed slide. At the top, there's a navigation bar with file options, a resize tool, and a timer set to 5.0s. The main content area has a white background with a purple border. On the left, there's a circular profile picture of a man. Below it is a list of five areas where JavaScript is used:

1. 웹 화면 개발(프론트엔드)
2. 모바일 앱 개발
3. 네이티브 앱 개발(Windows, macOS)
4. 서버 개발(백엔드)
5. 게임 개발

Number 2 is marked with a blue checkmark. To the right of the list is a central graphic featuring several logos arranged in overlapping circles: Microsoft Office, Facebook, Discord, Coinbase, Tesla, and React Native. Below the graphic are icons representing mobile devices (an iPhone and an Android phone) and the React Native logo.

A screenshot of a Canva presentation slide. The slide has a purple header bar with the title '자바스크립트 사용 분야' (JavaScript Usage Areas) in large white text. Below the title is a list of five items: 1. 웹 화면 개발(프론트엔드), 2. 모바일 앱 개발, 3. 네이티브 앱 개발(Windows, macOS), 4. 서버 개발(백엔드), and 5. 게임 개발. To the right of the list is a collage of images related to web development, including a screenshot of a Google Sheets document titled 'Business Report', several small photos of people, and a large logo for 'HTML5'. A circular inset on the left shows a young man's face. The slide is part of a larger presentation, as evidenced by the numbered thumbnails at the bottom.

Google Chrome File Edit View History Bookmarks Profiles Tab Window Help

canva.com/design/DAF-jM0dZzs/a4oa0GvyHnp477KD5ARwjjw/edit

JavaScript란 무엇인가? Try Canva Pro

Design Elements Text Brand Uploads Draw Projects Apps

Animate 5.0s Position

0 100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900

자바스크립트 사용 분야

- 웹 화면 개발(프론트엔드)
- 모바일 앱 개발
- 네이티브 앱 개발(Windows, macOS)
- 서버 개발(백엔드)**
- 게임 개발

Bun.js Node.js

Page 10 / 22 61%

Google Chrome File Edit View History Bookmarks Profiles Tab Window Help

canva.com/design/DAF-jM0dZzs/a4oa0GvyHnp477KD5ARwjjw/edit

JavaScript란 무엇인가? Try Canva Pro

Design Elements Text Brand Uploads Draw Projects Apps

Animate 5.0s Position

0 100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900

값, 타입 및 연산자

값 '에피'

타입

이름 '3월 22일' 날짜 문자열 숫자 주소 마우스 위치

서울특별시 은평구

Page 12 / 22 59%

Elements Console Sources > 27 1 Custom levels ▾

top Filter 21 Issues: 1 20 1 hidden

- > '에피코딩'
- < '에피코딩'
- > 35000
- < 35000
- > 3.1
- < 3.1
- > '나는 에피다'
- < '나는 에피다'
- > 35000 > 500
- < true

값, 타입 및 연산자

산술 연산자

더하기 빼기 곱하기 나누기

비교 연산자

이상 이하 미만 초과 동등

35000 > 500
true

21 Issues: ✎ 1 ✎ 20 | 1 hidden ⚙

```
> '에피코딩'  
< '에피코딩'  
> 35000  
< 35000  
> 3.1  
< 3.1  
> '나는 에피다'  
< '나는 에피다'  
> 35000 > 500  
< true          값, 타입 및 연산자  
> 10 <= 50          산술 연산자          비교 연산자  
< true  
> 10 < 5  
< false  
> 10 == 5  
< false  
> 10 == 10  
< true  
>
```

산술 연산자

비교 연산자

더하기	이상	$>=$
빼기	이하	$<=$
곱하기	미만	$<$
나누기	초과	$>$
	동등	$==$

변수



가치 value

나이 : 500

이름 : 어피

가격

•

값만 있으면, 그것이 나이인지
이름인지 뭔지 모른다.

따라서 나이라는 박스에 넣어두고
500는 돈이 아니라 나이라는 것을
알 수 있게 된다.
이 박스가 값을 넣어두는 변수라는 것

```
< > Elements Console Sources  
[ ] ( ) top ▾ | Filter  
21 Issues: ✘ 1 ⚠ 20 | 🔍  
  
> 500  
< 500  
> let 나이 = 500  
< undefined  
> 나이  
< 500  
> 나이 > 100  
< true  
> |
```

The image shows a digital interface, likely from a software like Canva or a similar design tool. On the left, there's a vertical sidebar with icons for Design, Elements, Text, Brand, Uploads, Draw, Projects, and Apps. The main area features a circular portrait of a young man in the top-left corner. To the right of the portrait is a box containing the Korean word "변수" (variable) with blue arrows pointing to it. At the bottom of the screen is a horizontal timeline with numerical markers from 0 to 700. Below the timeline are several small preview cards for different projects or assets, numbered 1 through 7. Project 1 is titled "2020년 첫 번째 웹툰" (2020 First Webtoon) and "JavaScript 기반 웹툰". Project 2 is titled "제작자에게 드리는 편지" (Letter to the Producer). Project 3 is titled "제작자에게 드리는 편지" (Letter to the Producer). Project 4 is titled "제작자에게 드리는 편지" (Letter to the Producer). Project 5 is titled "제작자에게 드리는 편지" (Letter to the Producer). Project 6 is titled "제작자에게 드리는 편지" (Letter to the Producer). Project 7 is titled "제작자에게 드리는 편지" (Letter to the Producer).

변수의 또 다른 특징은
나이를 넣어 두었다가
꺼내어서

다음에는 이름을 넣어 둘 수도 있다.

```
[Elements] [Console] [Sources] >>
[ ] [ ] top ▾ [ ] Filter

21 Issues: 1 20 | 2 hidden [gear]

> let 박스 = 500
< undefined
> 박스
< 500
> 박스 = '에피'
< '에피'
> 박스
< '에피'
> 박스
< '에피'
> 박스 = 10 < 50
< true
> 박스
< true
> |
```

변수 하나에

값 여러개 담기

뮤탈들

뮤탈 1
뮤탈 2
뮤탈 3
뮤탈 4



Elements Console Sources Network > 1 Custom levels

44 Issues: 1 43

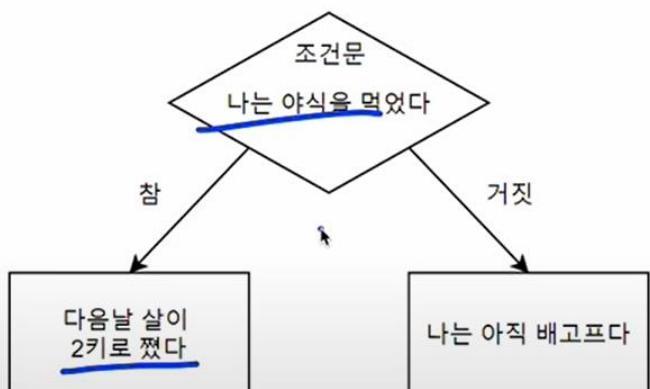
```
> const 뮤탈들 = ['뮤탈 1', '뮤탈 2', '뮤탈 3', '뮤탈 4']
< undefined
> 뮤탈들
< ▼ (4) ['뮤탈 1', '뮤탈 2', '뮤탈 3', '뮤탈 4'] i
  0: "뮤탈 1"
  1: "뮤탈 2"
  2: "뮤탈 3"
  3: "뮤탈 4"
  length: 4
  ► [[Prototype]]: Array(0)
> 뮤탈들[2]
< '뮤탈 3'
>
```

Elements Console Sources Network > 1 Custom levels

44 Issues: 1 43

```
> const 뮤탈들 = ['뮤탈 1', '뮤탈 2', '뮤탈 3', '뮤탈 4']
< undefined
  0 | 2 | 3
> 뮤탈들
< ▼ (4) ['뮤탈 1', '뮤탈 2', '뮤탈 3', '뮤탈 4'] i
  0: "뮤탈 1"
  1: "뮤탈 2"
  2: "뮤탈 3"
  3: "뮤탈 4"
  length: 4
  ► [[Prototype]]: Array(0)
> 뮤탈들[2]
< '뮤탈 3'
>
```

조건문



조건문

이 사람이 롤 스킨을 구입 했는가?

조건문

혈중 알콜 농도가 0.03% 이상인가?

이 사람이 입력한 이메일 주소가 올바른 형식인가?

정규식 .com → 조건문

A screenshot of the Chrome DevTools console tab. The interface includes tabs for Elements, Console, and Styles (partially visible). Below the tabs are buttons for play/pause, stop, and filter, along with a status bar indicating 44 issues, 1 error, and 43 warnings. The main area displays a piece of JavaScript code:

```
> if (나이 == 500) {  
    // 코드가 실행~  
}
```

The code is written in Korean, where "나이" means "age". The condition "나이 == 500" is likely a placeholder or a mistake, as ages do not typically reach 500. The code is intended to execute the block following the if statement.

canva.com/design/DAF-jModZzs/a4oa0GvyHnp477KD5ARw/jw/edit

File Resize & Magic Switch Position

Design Elements Text Brand Uploads Draw Projects Apps

반복문

2단
 $2 \times 1 = 2$
 $2 \times 2 = 4$
 $2 \times 3 = 6$
 $2 \times 4 = 8$
 $2 \times 5 = 10$
 $2 \times 6 = 12$
 $2 \times 7 = 14$
 $2 \times 8 = 16$
 $2 \times 9 = 18$

클릭 한번에 모든 유닛을 어떻게 움직일까?



| ~ //

Elements Console Sources > 23 1 Custom levels ▾

44 Issues: 1 43

```
> for (let i = 1; i <= 11; i++) {
  console.log(i + '번 뮤탈 이동')
}

1번 뮤탈 이동 VM488:2
2번 뮤탈 이동 VM488:2
3번 뮤탈 이동 VM488:2
4번 뮤탈 이동 VM488:2
5번 뮤탈 이동 VM488:2
6번 뮤탈 이동 VM488:2
7번 뮤탈 이동 VM488:2
8번 뮤탈 이동 VM488:2
9번 뮤탈 이동 VM488:2
10번 뮤탈 이동 VM488:2
11번 뮤탈 이동 VM488:2

< undefined
```

Elements Console Sources Network > 1 Custom levels ▾

44 Issues: 1 43

```
> for (let i = 1; i <= 9; i++) {
  console.log('2 * ' + i)
}

2 * 1 VM558:2
2 * 2 VM558:2
2 * 3 VM558:2
2 * 4 VM558:2
2 * 5 VM558:2
2 * 6 VM558:2
2 * 7 VM558:2
2 * 8 VM558:2
2 * 9 VM558:2

< undefined
> console.log('2 * 1')
2 * 1 VM600:1

< undefined
> console.log('2 * 2')
2 * 2 VM636:1

< undefined
>
```

canva.com/design/DAF-jMODZzs/a40a0Gvyhnp477KD5ARwjq/edit

File Resize & Magic Switch 5.0s Position

Elements Text Brand Uploads Draw Projects Apps

함수

함수 $f(x)$ 는 입력 x 를 출력 y 로 映射합니다.

$y = f(x)$

어렵다 😞

canva.com/design/DAF-jMODZzs/a40a0Gvyhnp477KD5ARwjq/edit

함수

회원가입 / 로그인
결제승인
매수체결
파일삭제

함수는 행위다

호출

Elements Console Sources VM717:2

Custom levels

44 Issues: 1 43 | 4 hidden

```
> function 로그인(이메일) {
    console.log(이메일 + '로 로그인 했습니다')
}
< undefined
> // 호출
< undefined
> 로그인('effy@naver.com')
effy@naver.com로 로그인 했습니다
< undefined
> |
```

Elements Console Sources Network VM851:2

Custom levels

44 Issues: 1 43 |

```
> function 파일삭제(파일명) {
    console.log(파일명 + '이 삭제됐습니다')
}
< undefined
> // 선언
// 호출
< undefined
> 파일삭제('photo.png')
photo.png이 삭제됐습니다
< undefined
> |
```

객체



객체는 행위와 데이터를 묶어둔 것

야스오 OWNER 행위
행동 데이터

```
Elements Console Sources Network > 1 Custom levels ▾
[▶ ⚡ top ▾ | Filter | 44 Issues: 1 43 | ⚡ ]
> const 야스오 = {
  Q: () => console.log('찌르기'),
  W: () => console.log('장막'),
  HP: 500,
  MovementSpeed: 350,
}
< undefined
> 야스오.Q()
찌르기 VM990:2
< undefined
> 야스오.W()
장막 VM990:3
< undefined
> 야스오.HP
< 500
> 야스오.Mov
< undefined
> 야스오.MovementSpeed
< 350
> |
```

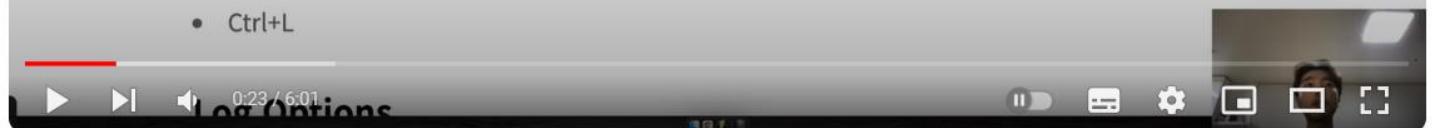

나는 개발할 때 Chrome 개발자 도구의 console을 굉장히 자주 사용하는데(대부분의 개발자들이 그럴 것이라 생각한다), 오늘은 크롬 개발자 도구의 편리한 기능과 잘 쓰진 않지만 알아두면 좋은 기능들을 한 번 정리해 보고자 한다.

Console Open

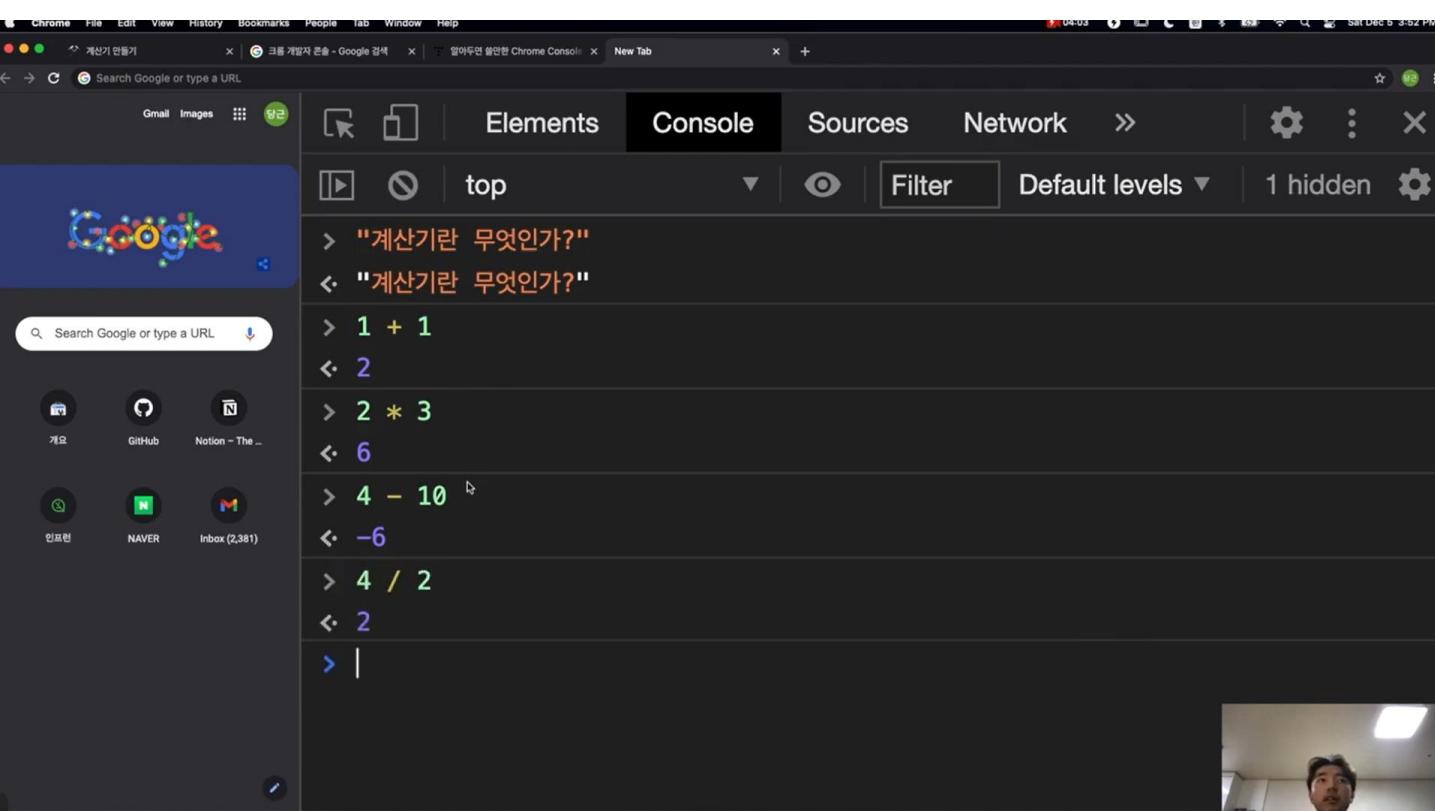
- Windows / Linux: Ctrl+Shift+J
- Mac: Cmd+Opt+J
- 개발자 도구가 열린 상태: ESC

Console Clear

- clear() 입력
- Ctrl+L



자바스크립트 함수 6분 만에 이해시켜드림 (안되면 200% 환불)



The screenshot shows a browser's developer tools console tab open. The console displays the following Korean code:

```
> function 더하기(a, b) {  
    a + b  
}  
< undefined  
> 더하기(10, 5)  
< undefined
```

Yellow annotations are overlaid on the code:

- The first two lines are highlighted with a yellow box, containing the text: "関数なので、アクションされたが、pc内部的に足すアクションはされたが、".
- The last two lines are highlighted with another yellow box, containing the text: "画面上に見せるアクション命令がないので、見た目的に確認できないだけ。".

Below this, there is a larger section of code:

```
> function 더하기(a, b) {  
    a + b  
}  
< undefined  
> 더하기(10, 5)  
< undefined  
> console.log("나는 사람이다")  
나는 사람이다  
< undefined  
> console.log(10000)  
10000  
< undefined
```

A yellow box highlights the line "나는 사람이다" with the text: "画面上に見せるアクション命令として、".

At the bottom of the screenshot, the browser's main window is visible, showing the Google homepage. The developer tools interface is overlaid on the top of the browser window.

