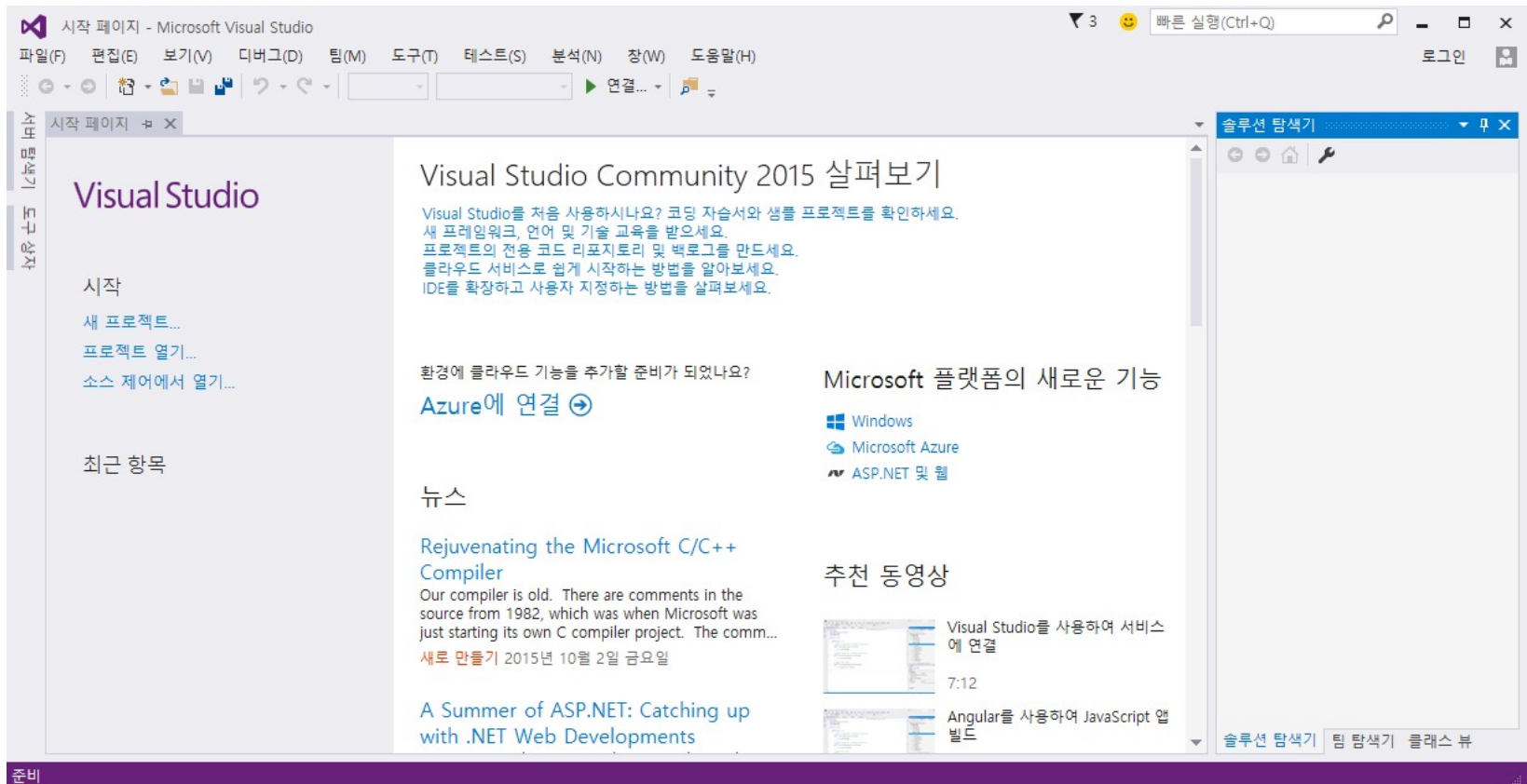


컴퓨터비전

AI프로그래밍을 위한 개발환경

SW프로그래밍을 위한 개발환경

- C언어, 고급 C언어, 자료구조, 알고리즘 수업시간에 사용한 통합개발환경(IDE)
 - 비주얼스튜디오(Visual Studio) ← 대표적인 예시



AI프로그래밍을 위한 개발환경

- 클라우드 기반 주피터 노트북
 - 퍼블릭 클라우드 정의와 예
 - 특정 기업이나 사용자를 위한 서비스가 아닌 인터넷에 접속 가능한 모든 사용자를 위한 클라우드 서비스 모델. 클라우드 서비스 제공자(CSP)가 하드웨어, 소프트웨어를 관리함. 데이터나 기능, 서버 같은 자원은 각 서비스에서 사용자 별로 권한 관리가 되거나 격리 되어, 서비스 사용자 간에는 전혀 간섭이 없다는 장점이 있음
 - (무료용) 구글이 제공하는 코랩(Colab)과 캐글(Kaggle)이 존재하며 수업용으로 활용하기에 제공되는 리소스가 적절함. 그러나 현업 연구/개발용으로는 부족하며 대용량 데이터 기반 모델 학습에 어려움이 많음
 - (유료용) 대표적인 CSP로는 구글, 아마존, 마이크로소프트 등 이 존재하며 서버의 운영을 전혀 신경쓰지 않고 연구/개발에만 집중할 수 있다는 장점이 있으나 비용이 비쌈

AI프로그래밍을 위한 개발환경

- 클라우드 기반 주피터 노트북
 - 프라이빗 클라우드 정의와 예
 - 제한된 네트워크 상에서 특정 기업이나 특정 사용자만을 대상으로 하는 클라우드 서비스 모델. 서비스의 자원과 데이터는 기업 내부에 저장되며, 기업이 자원의 제어권을 갖음. 보안성이 매우 뛰어나며, 개별 고객의 상황에 맞게 클라우드 기능을 커스터마이징 할 수 있다는 장점이 있음.
 - (내돈내산) 현업의 인공지능 분야 연구/개발팀은 연구원이 직접 서버를 구축하여 사용하거나, IT 부서 내 서버 운영팀이 서버를 구축하여 제공해줌. 장시간 컴퓨팅 리소스를 사용하여 연구개발을 진행할 경우 퍼블릭 클라우드를 사용하는 것보다 비용적으로 저렴함. 더불어 학습 데이터 및 알고리즘에 대한 유출의 위험을 줄일 수 있음
 - 인공지능 분야 연구/개발자를 꿈꾸는 학생은 개발환경 구축을 직접 해보길 권장함

퍼블릭 클라우드 개발환경 : 구글 코랩

- 구글 코랩(Colab)
 - 클라우드 기반 주피터 노트북 개발 환경 제공
 - gmail 계정만 있으면 바로 사용 가능 (약! 3분)
 - 리눅스OS + 도커 + 주피터노트북(GPU포함) + 구글드라이브 로 구성
 - 모든 코랩은 세션이 끝나면 작업하던 데이터가 삭제되므로, 반드시 구글드라이브 개인 계정을 통해 중간 백업 필수
 - 퍼블릭 클라우드 이므로, 보안 이슈가 있는 데이터는 활용하기 어려움
- 구글 코랩 (Colab) 종류
 - Colab: 무료, Colab Pro: 매달 9.99\$, Colab Pro+: 매달 49.99\$
 - Colab: GPU K80, T4, P100, 최대 12시간 지원, RAM 16GB
 - Colab Pro: GPU K80, T4, P100, 24시간 지원, RAM 32GB
 - Colab Pro+: GPU K80, T4, P100, 24시간 지원, RAM 52GB, 백그라운드 프로세스 지원

퍼블릭 클라우드 개발환경 : 캐글

- 캐글 (Kaggle)
 - 데이터 사이언스 경진대회 플랫폼
 - 대회를 위해 클라우드 기반 주피터 노트북 개발 환경 지원
 - 계정만 있으면 사용 가능하나, 1인 1계정만 지원 (핸드폰 인증 필수)
 - 리눅스OS + 도커 + 주피터노트북(GPU포함) 로 구성
 - 퍼블릭 클라우드 이므로, 보안 이슈가 있는 데이터는 활용하기 어려움
- 캐글 (Kaggle)
 - 캐글: 무료, 캐글+구글클라우드 (Google Cloud Services)
 - 캐글: 주당 GPU 40시간 지원, RAM 12GB
 - 캐글+구글클라우드: 사용한 만큼 비용 지급

프라이빗 클라우드 개발환경 : 내돈내산

- 프라이빗 클라우드 개발환경 구축 시 고려할 점: 서버 스펙
 - 머신러닝/딥러닝 모델 학습에 필요한 GPU 서버의 경우, 관련 연구의 논문을 살펴보면 모델 학습에 필요한 서버 스펙을 참고 할 수 있음
 - GPU 모델 정보 및 GPU 카드 장수
 - 머신러닝/딥러닝 모델 추론에 필요한 GPU 서버의 경우, 서비스 하고자 하는 응용의 알고리즘 처리 속도에 의존적임
 - GPU 처리 속도와 처리해야 하는 쿼리 수 등을 고려하여 설계
- 임베디드 개발환경 (참고*)
 - 머신러닝/딥러닝 모델 추론에 필요한 NPU(Neural Processing Unit) 보드를 사용하며, GPU (General Processing Unit)처럼 일반적인 연산 처리를 담당하는 코어를 제작하여 하드웨어로 제작한 것이 아닌, 인공지능 연산을 수행하기 위해 필요한 Layer를 직접 물리적으로 제작한 것임
 - 대표적으로 Nvidia Jetson 보드가 있음



프라이빗 클라우드 개발환경 : 내돈내산

- 형태로 구분하는 서버의 종류
 - 랙마운트형 서버: 서버 랙 안에 들어가는 서버, 데이터 센터 및 별도의 서버룸에 입고되는 서버는 랙마운트형이 일반적임 (서버당 GPU 8장 이상)
 - 타워형 서버: 사내 서버룸에 설치되거나 사무실에 설치되는 서버로 일반 PC와 비슷하게 생겼음. 주로 워크스테이션이라고 부름 (PC당 GPU 4장 이하)



랙마운트형 서버



타워형 서버

프라이빗 클라우드 개발환경 : 내돈내산

- 클라우드 서버 개발환경 구축
 - 리눅스OS (베이스) + Ubuntu 도커(GPU포함) + 아나콘다(파이썬+개발 툴 포함) + 주피터노트북/VSCODE (강력추천*) (수업 내 실습 예정)
- 윈도우OS (베이스) + Ubuntu 도커(GPU포함) + 아나콘다(파이썬+개발 툴 포함) + 주피터노트북/VSCODE
 - 최근 마이크로소프트 사의 지원으로 윈도우 OS 위에서 도커 환경 구축을 위한 기술지원을 본격적으로 시작하였음. 그러나 여전히 이슈가 많아 개발 환경 구축이 용이하지 않다는 단점이 있음
 - Window 10 Pro Edition 이상
 - Hyper-V 기능의 지원으로 도커 사용 가능
 - WSL2 기반 도커 사용 가능
 - Window 10 Home Edition
 - Hyper-V 기반 도커 사용 불가능
 - WSL2 기반 도커 사용 가능

프라이빗 클라우드 개발환경 : 내돈내산

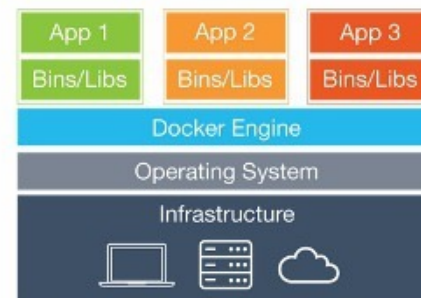
- 클라우드 서버 개발환경 구축
 - 왜 리눅스 OS인가?
 - 대다수 오픈소스 소프트웨어(OSS)들이 리눅스 환경에서 개발되었고, 리눅스 환경의 연구개발을 적극 지원함
 - 적극 지원하지 않는 윈도우 환경의 경우, OSS 혹은 MS사가 지원해주길 기다리거나 본인이 얼리어답터가 되어 여러 삽질을 통해 이슈 적극 해결
 - 왜 파이썬 언어인가?
 - 언어가 상대적으로 쉽고 간소하여 생산성이 높음
 - 유명 머신러닝/딥러닝 라이브러리와 프레임워크(scikit learn, tensor flow, pytorch)가 파이썬 기반으로 작성되어 있고, 대기업 중심(구글, 페이스북)의 적극적인 유지보수지원을 통해 지속적인 발전 중
 - 머신러닝/딥러닝의 붐이 시작되던 초반에는 C언어, Matlab, Lua 등의 라이브러리와 프레임워크도 많이 사용했으나 현재는 춘추전국시대를 넘어 안정기에 돌입

프라이빗 클라우드 개발환경 : 내돈내산

- 클라우드 서버 개발환경 구축
 - 도커 (Docker) 란?
 - 리눅스 컨테이너를 기반으로 특정한 서비스를 패키징하고 배포하는데 유용한 오픈소스 프로그램임.
 - 컨테이너란? 가상머신과 달리 꼭 필요한 것만 담겨서 구동되며, 컨테이너에 필요한 커널은 호스트의 커널과 공유해서 사용하고, 컨테이너 안에는 애플리케이션을 구동하는데 필요한 라이브러리 및 실행 파일만 존재하기 때문에 컨테이너를 이미지로 만들 경우 용량이 줄어듬



Virtual Machines



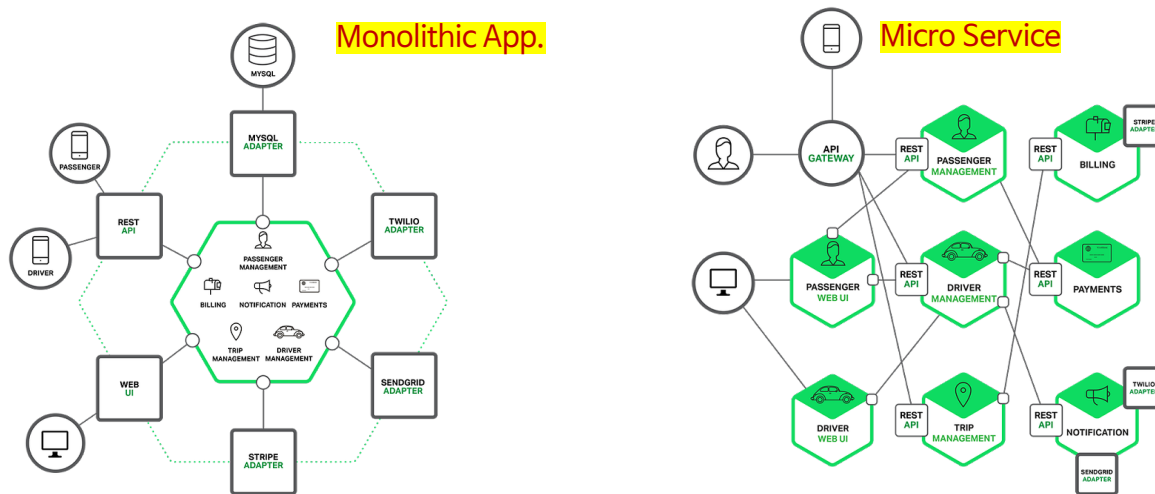
Containers

프라이빗 클라우드 개발환경 : 내돈내산

- 클라우드 서버 개발환경 구축

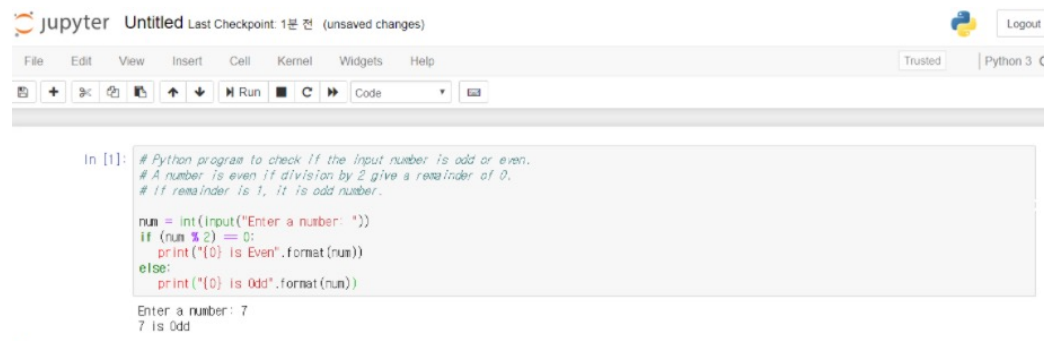
- 도커의 장점은?

- **어플리케이션의 독립성.** 독립된 공간을 보장받을 수 있어 호스트 OS 혹은 다른 컨테이너와 충돌 발생 염려가 없음
 - 컨테이너 내부에서 개발을 진행 후 배포가 필요한 경우 도커 이미지를 패키지로 만들어서 운영서버에 전달 하면 됨. **배포 매우 쉬워 짐**
 - 모놀리식(Monolithic) 어플리케이션 방식에서 마이크로(Micro)서비스 구조로 변화가 쉬움. **컨테이너 하나당 하나의 기능만을 제공하는 모듈로** 만들어서 부하가 많은 모듈은 여러 개로 나누어 관리가 가능



프라이빗 클라우드 개발환경 : 내돈내산

- 클라우드 서버 개발환경 구축
 - 주피터 노트북(Jupyter Notebook) 이란?
 - 웹브라우저상에서 파이썬 코드를 단계적으로 쉽게 실행하고, 시각적으로 빠르게 확인해볼 수 있도록 도와주는 프로그램
 - 탐색적 데이터 분석에 적합하여 많은 데이터 분석가가 주로 사용
 - 장점은 셀 단위로 코드를 작성하여 실행이 가능하기 때문에 큰 파일도 셀 단위로 나누어 실행하면서 인터랙티브한 동작이 가능함
 - 예를 들어, 데이터 분석을 위한 파이썬 파일 작성 후 실행하였을 때, 차트, 표 등의 결과 값 출력도 바로 직관적으로 볼 수 있음
 - Github에 주피터 노트북의 결과 출력 방식 그대로 업로드 할 수 있다는 장점도 가지고 있음



The screenshot shows the Jupyter Notebook web interface. At the top, it says 'jupyter Untitled Last Checkpoint: 1분 전 (unsaved changes)' and has a 'Logout' button. Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. Under the menu bar is a toolbar with icons for file operations, cell navigation, and execution. The main area contains a code cell with the following Python code:

```
In [1]: # Python program to check if the input number is odd or even.
# A number is even if division by 2 give a remainder of 0.
# If remainder is 1, it is odd number.

num = int(input("Enter a number: "))
if (num % 2) == 0:
    print("{0} is Even".format(num))
else:
    print("{0} is Odd".format(num))
```

Below the code cell, the output is displayed:

```
Enter a number: 7
7 is Odd
```