

# Open-source software (OSS) 소개

---

세종대학교 이은상

# Open-Source Software (OSS)

- 오픈소스 개념

- 소스 코드에 대한 접근, 자유로운 재배포, 파생 저작물의 작성, 제한없는 사용 등을 허용하는 라이선스와 함께 배포되는 SW
- 누구나 소스 코드를 검사, 수정 및 개선할 수 있는 소스 코드가 포함된 SW
- Open-source software (OSS)는 간략하게 오픈 소스라고도 불림
- 오픈소스 핵심 정의 요소 중 하나는 라이선스이며, 오픈소스는 이들 라이선스의 중요 요구사항들을 만족시켜야함

# Open-Source Software (OSS)

- 독점 SW
  - Closed source SW 혹은 독점 SW가 있음. 이는 소스코드에 대한 독점적인 소유권을 유지함. 즉, 그들만이 오류 수정 및 새로운 기능 추가를 위해 코드를 수정할 수 있음
    - ex) Adobe Photoshop, Norton AntiVirus
    - ex) Windows

# Open-Source Software (OSS)

- 독점 SW와의 차이점1: 신뢰성
  - 독점 SW는 단일 조직 또는 개발자를 통해 코드 업데이트함
  - 오픈소스는 더 넓은 커뮤니티에서 유지관리됨
  - 따라서 오픈소스에서 새 변경사항이 세밀하게 테스트되고 신뢰성이 더 높은 경우가 많음

# Open-Source Software (OSS)

- 독점 SW와의 차이점2: 보안
  - 모든 소스코드는 사이버 공격에 취약한 보안 결함이 있을 수 있음. 오픈소스는 수정이 더 빠름. 커뮤니티 구성원이 보안 취약점을 보고하면 오픈소스 프로젝트에서 하루, 이틀 내에 코드 업데이트 릴리즈함
  - 반면, 독점 SW는 다음 이유로 업데이트 주기가 김
    - 공급업체 작업자가 더 적을 수 있음
    - 공급업체가 보안 결함보다 재정적 고려사항을 우선시할 수 있음
    - 공급업체가 여러 변경 사항을 번들로 묶어 한 번에 릴리스하는 것을 선호하여 보안 업데이트 릴리스가 지연될 수 있음

# Open-Source Software (OSS)

- 독점 SW와의 차이점3: 라이선스
  - 회사는 보통 독점 라이선스에 따라 SW를 판매함
  - 누구도 허가없이 독점 SW를 보거나 편집하거나 수정할 수 없음.  
혹은 개인 사용은 허용하지만 재판매는 허용하지 않을 수 있음.

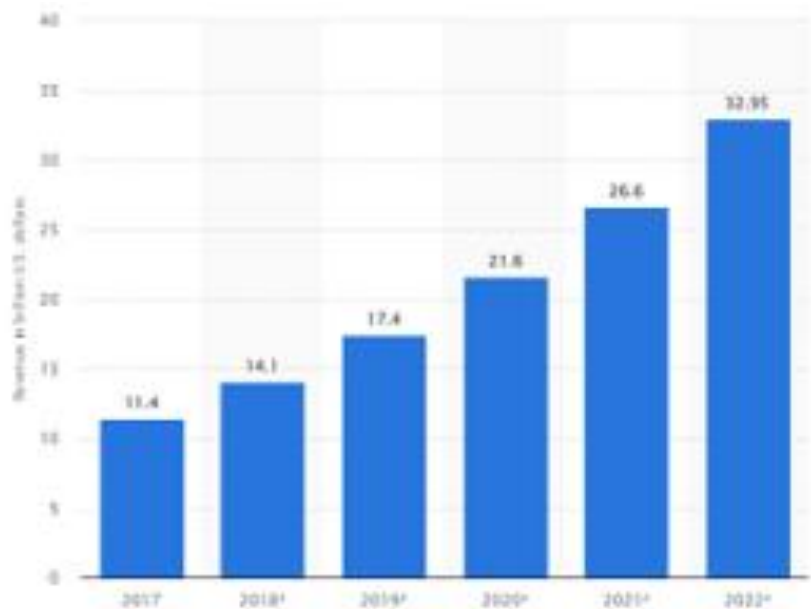
# 오픈소스 현황

- 오픈소스를 만들어도 독점 라이선스로 팔아 수익을 남길 수 없음
- 과연 오픈소스가 많이 활용될까?



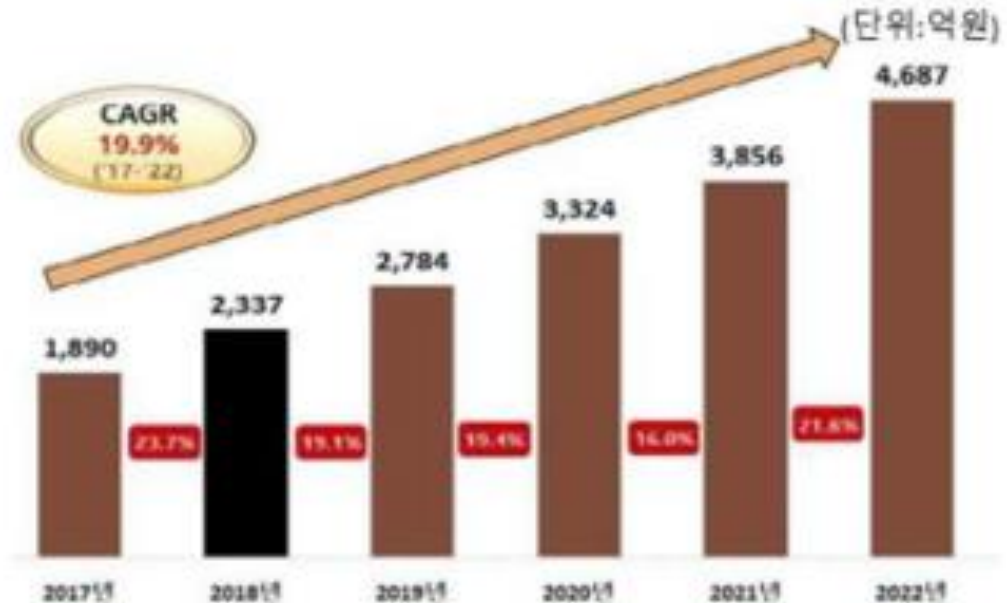
# 오픈소스 현황

- <https://www.comworld.co.kr/news/articleView.html?idxno=50514>



해외 오픈소스 SW 시장규모

- 가파른 성장세를 보이고 있음. 해외 시장의 경우 2022년 약 320억달러
- 국내 시장의 경우 2022년 약 4687억원 규모

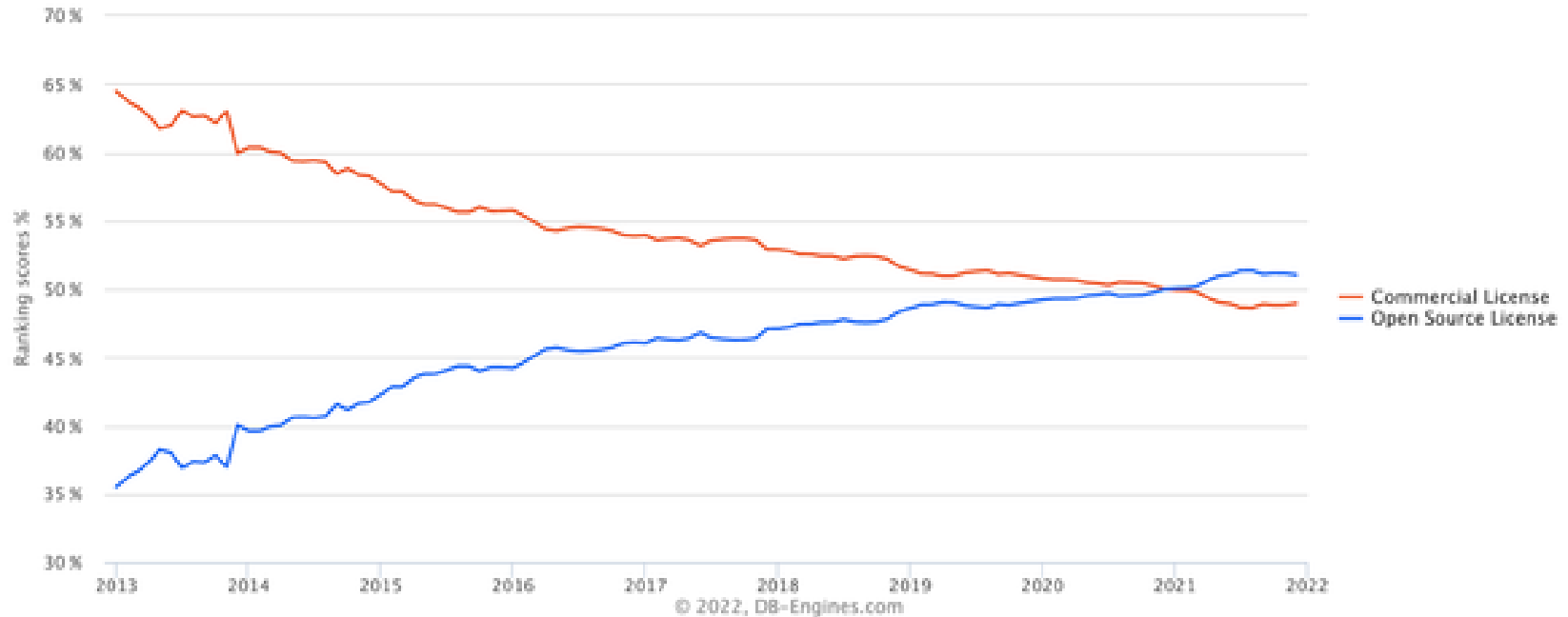


국내 오픈소스 SW 시장규모



# 오픈소스 현황

Popularity trend



- 2021년 기점으로 오픈소스의 영향력이 더 커짐. 상업 라이선스를 앞서고 있음
- 오픈소스는 현재 전 세계의 주류이며 산업의 패러다임을 바꿔 놓고 있음. 시대 흐름에 맞게 오픈소스 활용 필요

# 오픈소스 성립 조건

- 오픈소스의 배포 조건은 다음 기준을 준수해야함
- 1. 무료 재배포 (Free redistribution)
  - 오픈소스 SW를 누구나 자유롭게 받아서 사용하고 재배포할 수 있음
  - 라이선스는 SW를 판매하거나 제공하는 당사자를 제한해서는 안 됨
  - 라이선스는 해당 판매에 대한 로열티나 기타 수수료를 요구해서는 안 됨
- 2. 소스 코드 (Source code)
  - 오픈소스의 소스코드를 누구나 볼 수 있도록 공개해야함
  - 프로그램은 소스코드를 포함해야 하며 컴파일된 형식 뿐만 아니라 소스 코드로 배포할 수 있어야함

# 오픈소스 성립 조건

- 3. 파생 작품 (Derived Works)
  - 라이선스는 수정 및 파생 된 저작물을 허용해야함
  - 원본 소프트웨어의 라이선스와 동일한 조건에 따라 배포될 수 있어야함
- 4. 저자 소스 코드의 무결성(Integrity of the author's source code)
  - 오픈소스 소스코드의 무결성을 보장해야함
  - 다른 개발자가 소스코드를 수정하거나 개선할 때, 그 수정이나 개선이 원래의 소스코드와 다르게 나타나지 않도록 보장해야함
  - 수정된 소스코드는 반드시 원작자가 명시한 라이선스와 조건을 따라야함

# 오픈소스 성립 조건

- 5. 개인 또는 집단에 대한 차별 금지 (No discrimination against persons or groups)
  - 어떤 개인이나 집단을 차별해서는 안 됨
- 6. 노력 분야에 대한 차별 금지 (No discrimination against fields of endeavor)
  - 특정 분야에서 프로그램을 사용하는 사람을 제한해서는 안 됨

# 오픈소스 성립 조건

- 7. 라이선스 배포(Distribution of license)
  - 프로그램에 첨부된 권한은 해당 당사자가 추가 라이선스를 실행할 필요 없이 프로그램을 재배포하는 모든 사람에게 적용되어야 함
  - 오픈소스 SW를 사용하거나 배포할 때는 반드시 해당 소프트웨어의 라이선스를 함께 제공해야함
- 8. 라이선스는 제품에 한정되지 않아야 함 (License Must Not Be Specific to a Product)
  - 라이선스는 특정 제품에 한정되지 않아야 함
  - 프로그램에 첨부된 권리는 프로그램이 특정 소프트웨어 배포의 일부 인지에 따라 달라져서는 안 됨

# 오픈소스 성립 조건

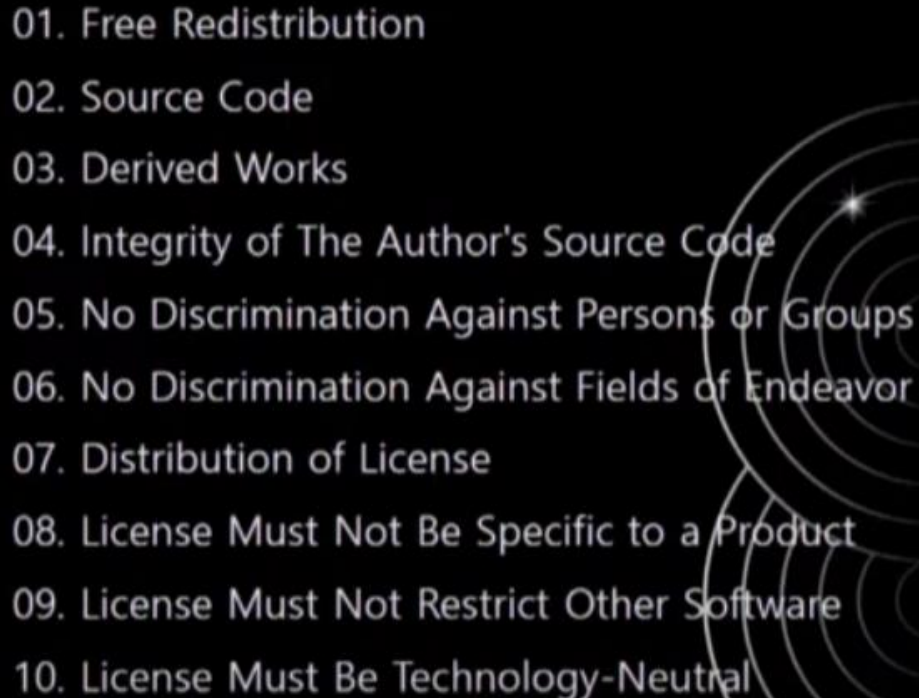
- 9. 라이선스는 다른 소프트웨어를 제한해서는 안 됨 (License Must Not Restrict Other Software)
  - 라이선스는 라이선스된 소프트웨어와 함께 배포되는 다른 소프트웨어에 제한을 두어서는 안 됨
  - 예를 들어, 라이선스는 동일한 매체에 배포된 다른 모든 프로그램이 오픈소스 소프트웨어야한다고 주장해서는 안 됨.
- 10. 라이선스는 기술 중립적이어야 함 (License Must Be Technology-Neutral)
  - 개별 기술 또는 특정 인터페이스 스타일에 대한 라이선스 조항은 제공하지 않음

# 오픈소스 성립 조건 필요성

- 모든 공개되어있는 소스가 다 유용한 소스 코드는 아님. 특히 산업계에서.
- 어떤 오픈소스를 제품(ex 갤럭시S23)에 넣었을 때 나중에 오픈소스 코드가 없어지거나 하면 난감해짐
- 없어진 오픈소스를 다시 만들어 오픈소스로 유지관리를 하든지 혹은 내부 코드로 자체적으로 개발해야함
- 제품 당 평균 수백개 정도의 오픈소스가 들어있음을 감안하면 이러한 불안한 오픈소스를 사용하는 것은 매우 위험한 소프트웨어 개발임

# 오픈소스 성립 조건 필요성

- 따라서 10개 기준이 만족하는 것만 사용하려고 함
- 가령 회사에서 코드 공개할 때 특정 경쟁사가 안 쓰도록 막고 싶어도 비차별성이 붙는 순간 사람들이 이 코드에 참여하지 않고 결국에는 사장됨 (5,6번 내용)
- 따라서 모든 오픈소스가 10개 기준을 따름

- 
01. Free Redistribution
  02. Source Code
  03. Derived Works
  04. Integrity of The Author's Source Code
  05. No Discrimination Against Persons or Groups
  06. No Discrimination Against Fields of Endeavor
  07. Distribution of License
  08. License Must Not Be Specific to a Product
  09. License Must Not Restrict Other Software
  10. License Must Be Technology-Neutral



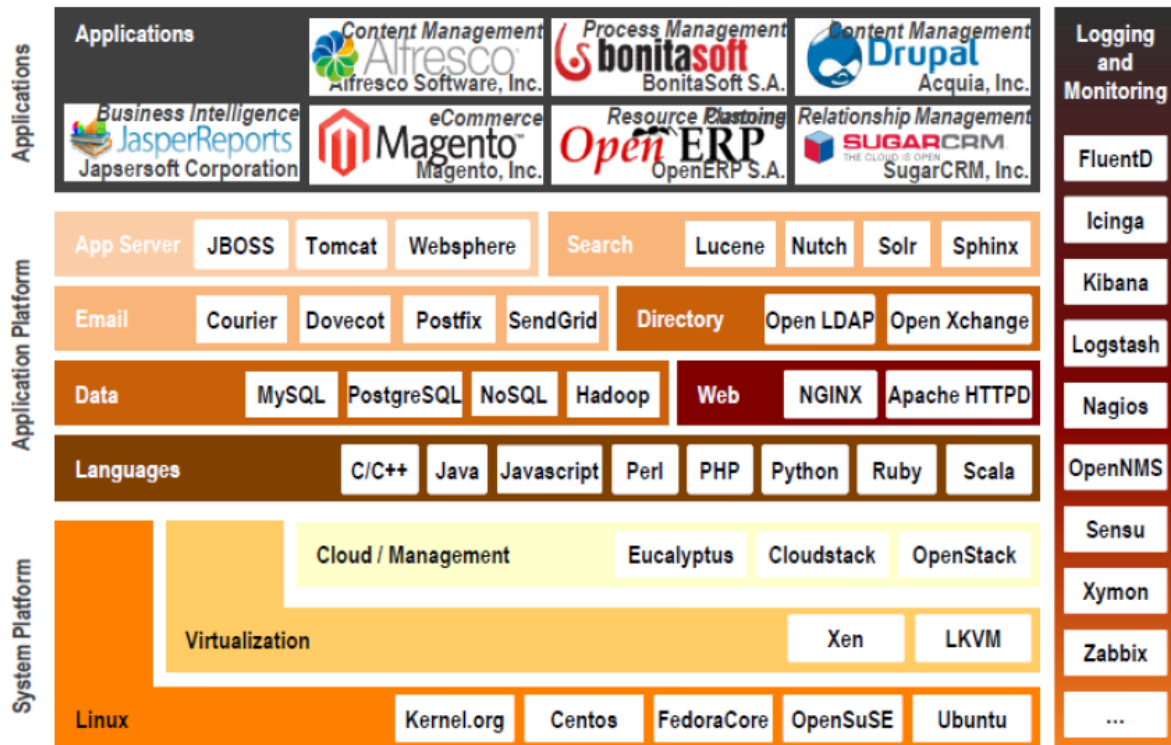
# 오픈소스 활용

- 오픈소스는 소스코드, 라이브러리, 유틸리티, 툴, 완제품 수준에 이르기까지 다양한 방법으로 활용될 수 있음

## 활용 단위



## 분류 체계



# 오픈소스 관련 단체

- FSF
  - Free software foundation
  - 리처드 스톨만에 의해 1985년 만들어진 비영리 단체
  - GNU 프로젝트를 운영하고 free SW를 배포 및 관리함
- OSI
  - Open source initiative
  - 오픈소스 정의 및 관련 표준을 관리함

# 오픈소스 관련 단체

- SFLC
  - Software freedom law center
  - 자유 소프트웨어/오픈소스 소프트웨어의 비영리 개발자를 위해 법률 상담 법률 대리 및 관련 서비스
- GPL Violation
  - 해럴드 벨테에 의해 2004년 만들어진 비영리 재단
  - GPL 라이선스 기반의 저작권 보호 및 소송 지원 단체
- 기타
  - Linux Foundation, FOSS, OIN 등

# 오픈소스 역사

- 초기

- 컴퓨터 초기에는 SW는 독점적이지 않았음
- 70, 80년 초반대부터 비로소 사람들이 자신 소프트웨어를 감추고 다른 사람이 볼 수 없게 함. 그 당시 컴퓨터를 사용하면 독점적인 운영체제를 사야했고 그것을 변경할 수 없었음

- GNU 프로젝트

- 리처드 스톨만은 소프트웨어 상업화에 반대하고 SW 개발 초기의 상호협력적인 문화로 돌아갈 것을 주장하며 1984년 GNU 프로젝트를 주도함
- 이듬해인 1985년 FSF 재단을 조직함.
- 다음과 같은 GNU 선언문을 제정하기도 함. 이를 지원하기 위해 copyright에 대응하는 copyleft 운동도 주장함
  - SW는 공유되어야하며 프로그래머는 SW로 돈을 벌어서는 안 된다.

# 오픈소스 역사

- 이름 변경
  - 원래 자유소프트웨어라는 이름에서 오픈소스 소프트웨어로 용어가 변경됨
  - 자유란 용어가 일반인이 무료로 인식하고, GPL 조항 엄격성 때문에 SW 개발이 용이하지 않다는 점을 탈피하기 위해서임
- OSI 결성
  - 1998년 오픈소스 SW를 인증하는 OSI가 에릭 레이몬드 등에 의해 결성되면서 오픈소스 SW 운동은 궤도에 오르게 됨

# GNU 프로젝트

- GNU 프로젝트
  - 리처드 스톨만이 창설한 공개 소프트웨어 프로젝트
  - GNU's Not Unix의 약자. 원래의 문장 안에 자신이 들어있는 재귀 약자 (그누로 발음)
  - GNU는 SW를 돈 주고 구입하지 말고 누구나 자유롭게 실행, 복사, 수정, 배포할 수 있게 하도록 주장함. 이러한 정신에 입각해서 만든 SW에 주어진 라이선스가 GPL임

# GNU 프로젝트

- GNU 프로젝트

- 1991년 리누스 토르발스는 유닉스 호환의 리눅스 커널을 작성하여 GPL 라이선스 아래 배포했고 다른 여러 프로그래머들에 의해 리눅스는 더 발전함. 1992년 리눅스는 GNU 시스템과 통합되었고, 이로써 완전한 공개 운영 체제가 탄생됨
- 리눅스는 수많은 사용자가 있고 수많은 프로그래머가 인터넷을 통하여 공동으로 개발하고 있는 가장 대표적인 오픈소스 운영체제임

# 오픈소스 Motivation

- 오픈소스는 기껏 만들어도 팔 수 없는데 왜 만들까?
- 자원봉사?





# 오픈소스 Motivation

- 대가

- 처음에 리눅스(1991년)는 선의로 만들어짐. 보수가 없어도 좋고 주말에 SW 개발에 참여하고 이 결과물을 다른 기업 등이 이득을 보으면 좋겠다는 마음으로 시작함
- 그러나 최근에는 오픈소스 개발이 실제 상업적인 이익이 되며 대부분 상업적인 이유로 오픈소스에 참여하게 됨
- 수많은 개발자가 개인의 이익을 위해 오픈소스에 기여하며 회사는 이런 개발자들에게 대가를 지불함

# 오픈소스 Motivation

- 호환

- 단순히 좋은 오픈소스 찾아 잘 쓰기만 하고 참여는 안 한다?
- 어떤 오픈소스를 제품에 사용한 이후에 업데이트되면서 기능이 많이 달라지면서 제품에 호환이 안 되는 경우 발생
- 그 때마다 안 맞는 부분을 다 조정하는 것은 불가능. 수백개의 오픈소스를 쓰기 때문
- 따라서 기업에서 자신이 가져다 쓴 부분에 대해 다음 오픈소스 버전에 적극적으로 기여함. 기여해놓으면 그 기여부분이 반영된 건 그대로 가져다 쓸 수 있음

# 오픈소스 Motivation

- 취업
  - 개발자에게 중요한 것은 실력
  - 실력을 정량화하기 어려움. 면접을 통해 실력이 있는지 알기 어려움
  - 회사에서의 작업물은 외부에 보여주기도 어렵고 말로 설명 쉽지 않음
  - 오픈소스 참여는 취업에 도움됨. 특히, 경력이 없는 신입개발자도 오픈소스 참여 이력을 보여주면 차별화를 줄 수 있음



“Talk is cheap. Show me the code.”

**Linus Torvalds**

# 오픈소스 Motivation

- 리뷰
  - 보통 오픈소스 committer들이 코드 수정을 그냥 받아주지 않고 유지관리를 함
  - 코드 리뷰를 많이 하고 이상한 것이 있으면 질문도 많이 함
  - 새로운 지식 학습 가능하며, 무료로 자신의 코드를 리뷰 받을 수 있음
  - 좋은 코드 작성법
    - 요구 사항 잘 이해
    - 코드 작성
    - 리뷰 받음
    - 리뷰 반영

# 오픈소스 성공 사례

- 리눅스, Fedora, Ubuntu, CentOS, Apache 웹서버, Firefox 웹브라우저, 안드로이드, MySQL, ...



# 오픈소스 성공 사례

- Stable diffusion
  - 텍스트를 이미지로 만들어주는 오픈소스
  - 예시)
    - a fantasy castle, landscape art, Ghibli, disney



# 오픈소스 성공 사례

- Stable diffusion
  - DALL-E (by OpenAI) 혹은 Midjourney 등의 유료 제품도 있음
  - 그런데 stable diffusion이 오픈소스로 나오면서 많은 사람들이 기여하며 성능이 매우 좋아짐

# 오픈소스 성공 사례

- Tensorflow, Pytorch
  - 신경망 학습에 사용되는 대표적인 오픈소스 라이브러리
- Flutter
  - 플러터(Flutter)는 단일 코드베이스에서 모바일, 웹, 데스크톱, 임베디드 기기를 위한 네이티브 컴파일 애플리케이션을 개발하기 위한 구글의 UI 킷



# 소프트웨어 유형

- Freeware
  - 프리웨어 SW는 무료로 사용할 수 있고 배포할 수 있는 SW
  - 사용자는 인터넷에서 프리웨어를 다운로드하여 사용할 수 있으나, 소스 코드를 공개하지 않음
  - ex) Adobe PDF, 구글톡
- Shareware
  - 셰어웨어 SW는 평가판으로 사용자에게 무료로 배포되는 SW
  - SW 사용에 시간 제한이 있음 (예: 30일 또는 2개월 무료)
  - 제한 시간이 지나면 비활성화되고 제한 시간 이후에 사용하려면 SW 비용을 지불해야함
  - ex) Adobe acrobat 8 professional, PHP Debugger 2.1.3.3

# 소프트웨어 유형

- Proprietary software
  - 독점 SW는 개인 또는 회사가 소유한 SW
  - 배포 및 사용에 제한이 있고, 폐쇄 소스(closed-source) 또는 상용 소프트웨어 (commercial software)라고도 함

	오픈소스 공개유무	비용	기능 or 사용기한 제한
오픈소스	O	X	X
Freeware	X	X	X
Shareware	X	무료 → 유료	O
Proprietary	X	O	

# 소프트웨어 지식재산권

- 현재 SW는 다음과 같이 저작권, 특허권, 상표권 영업비밀 등의 지식재산권에 의해 보호받고 있음
  - 저작권
    - 창작물에 대하여 창작자(저작자)가 취득하는 권리로서 창작의 결과물을 보호하며, 창작과 동시에 권리가 발생함
    - 따라서 어떤 프로그래머가 특정 SW를 개발하면 컴퓨터 프로그램 저작권이 자동 발생하며, 그 권리는 프로그래머 또는 그가 속한 회사에 부여됨
    - 저작권이 있는 저작물의 경우 누구도 저작권자의 허락 없이는 해당 저작물을 쓸 수 없음
- Copy left:** 저작권 없음이 아니라 저작권을 주장하지 않겠다는 의미 (GNU 프로젝트의 기본 이념)

# 소프트웨어 지식재산권

- 특허권

- 발명에 관하여 발생하는 독점적/배타적 지배권으로 법에 정해진 절차에 의해 출원을 해야하며, 심사를 통해 부여되는 권리임
- 특허기술을 사용하기 위해서는 반드시 특허권자의 허락을 얻어야함
- 특허 받은 방식을 구현하는 SW라면 프로그래밍 언어나 소스 코드와 상관없이 특허권자의 명시적인 허락을 받아야함

# 소프트웨어 지식재산권

- 상표권

- 상표권자가 지정상품에 관하여 그 등록상표를 사용할 독점적인 권리로서 일정한 절차에 따라 등록하여야 효력이 발생함
- 이러한 상표를 사용하기 위해서는 반드시 상표권자의 허락을 얻어야 하며 허락받지 않고 상표를 사용할 경우 처벌을 받게 됨
- 상표권을 취득한 SW의 경우 상표를 사용하려면 상표권자의 명시적인 허락을 받아야함

- 영업비밀

- 공개되지 않은 SW의 경우 영업비밀로서 보호를 받을 수 있으며, 공개된 SW라 하더라도 아이디어에 대한 부분은 영업비밀로 보호를 받을 수 있는 가능성이 있음
- 단, 영업비밀로서의 SW보호는 널리 공개되어 유통되는 경우에는 보호되고 어렵고, 이를 알지 못하고 사용한 제3자에게 법적으로 문제를 삼을 수 없음

# 상용 소프트웨어 대체

주요 오픈 소스 소프트웨어

소프트웨어	창시자	출시연도	설명
Linux	Linus Torvalds	1991	커뮤니티 주체로 개발한 컴퓨터 운영 체제. 혹은 커널을 뜻하기도 하며 리눅스는 자유 소프트웨어와 오픈 소스 개발의 가장 유명한 표본임
Android	Google	2008	휴대 전화를 비롯한 휴대용 장치를 위한 운영체제와 미들웨어, 사용자 인터페이스, 웹 브라우저 및 다양한 기능을 포함하는 소프트웨어 모음이자 모바일 운영 체제
MySQL	MySQL AB	1995	세계에서 가장 많이 쓰이는 오픈 소스의 관계형 데이터베이스 관리 시스템
Apache	Apache Software Foundation	1995	HTTP 웹 서버로서 BSD, 리눅스 등 유닉스 계열뿐 아니라 마이크로소프트 윈도우에서도 운용할 수 있음
WordPress	Matt Mullenweg	2003	세계 최대의 오픈 소스 저작물 관리 시스템으로 워드 프레스 기반 웹사이트는 전세계 웹사이트의 30%를 차지(2018년 기준)
Mozilla Firefox	Dave Hyatt & Blake Ross	2004	전 세계 웹 브라우저 시장점유율은 구글의 크롬(Chrome)과 애플의 사파리(Safari) 다음으로 약 12%를 차지함(2019년 기준)
GIMP	Spencer Kimball & Peter Mattis	1996	그림을 편집하는 데 쓰이는 무료 및 오픈 소스 소프트웨어. 이미지 편집, 다른 이미지 형식 간 변환 등 전문화된 작업에 사용됨.
OpenOffice	StarOffice	2002	다양한 운영 체제에서 사용할 수 있는 오피스 제품군. 마이크로소프트 오피스 97-2003 포맷을 비롯한 다양한 포맷을 지원
TensorFlow	Google	2017	다양한 작업에 대해 데이터 흐름 프로그래밍을 위한 오픈소스 인공지능 라이브러리
Bitcoin	Satoshi Nakamoto	2009	가장 유명한 오픈 소스 블록체인 프로젝트는 Bitcoin으로, 전 세계 어디서나 누구에게나 즉시 지불할 수 있다는 디지털 통화로 P2P 기술을 사용해 중앙 권한없이 작동함.

# 과제 1

- Revolution OS 영화(리눅스에 대한 다큐멘터리 영화) 감상문 쓰기
  - <https://www.youtube.com/watch?v=Eluzi70O-P4>
  - <https://www.youtube.com/watch?v=4ZHloJVhcRY> (한글자막)
  - 조교에게 메일로 제출(wndrnrdk@naver.com)