DEVOCEAN

C++ 개발자가 바라보는 Rust

SK텔레콤 양유석

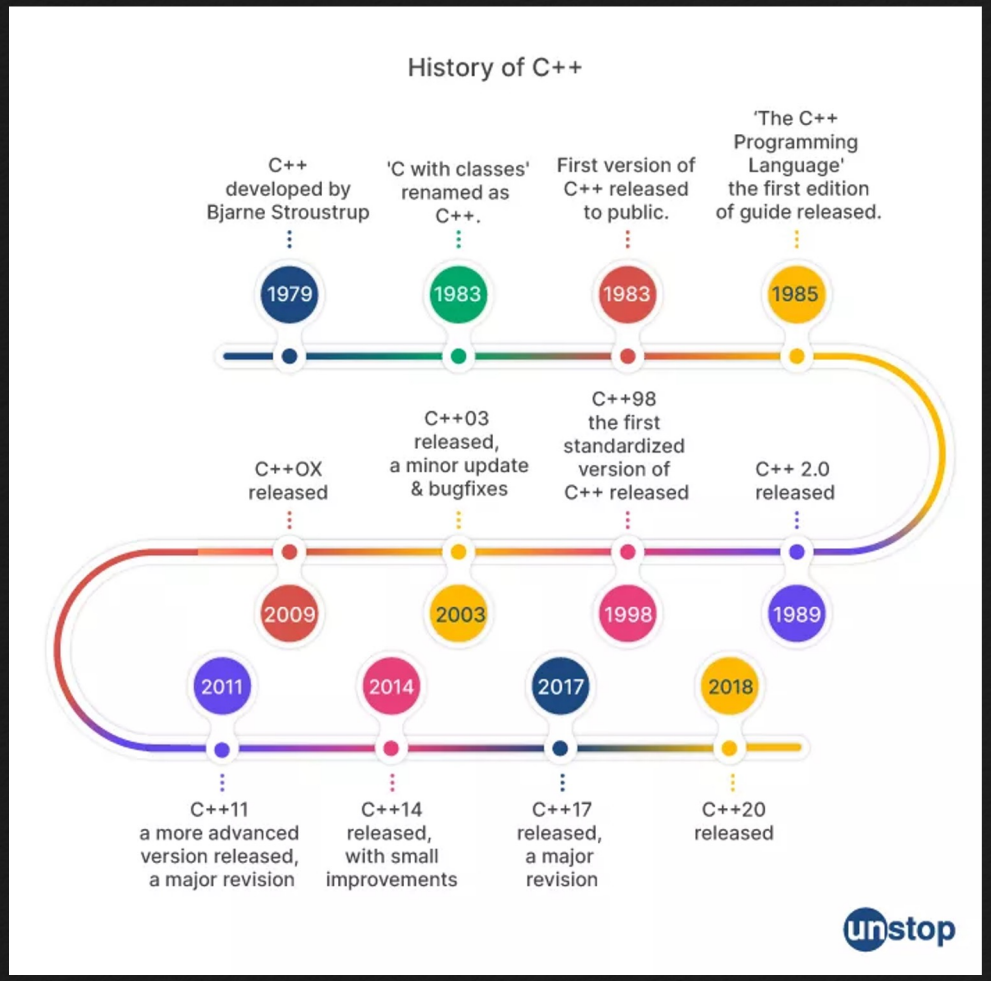
목차

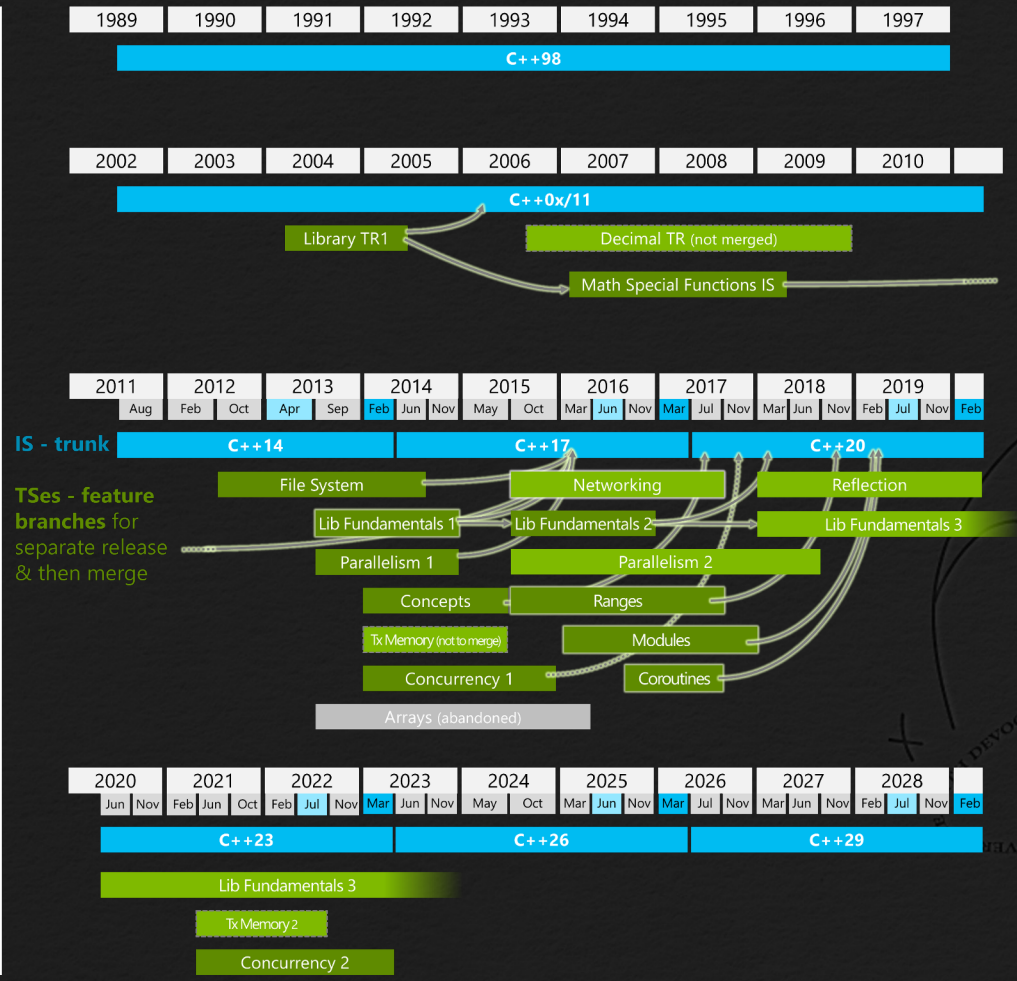
01. Rust Programming OpenLab

- 약 3개월의 여정
  - 4월 22일 ~ 7월 29일
- 스터디 선정 도서
  - The Rust Programming Language 2nd
- 결과물
  - 블로그
    - https://devocean.sk.com/blog/index.do?techType=OPENLAB
- 스터디를 통해 얻은 것
  - 지금부터 발표하겠습니다.
- 참고사항
  - Rust 와 C++ 를 비교한 내용은 인터넷에 많이 있습니다.
  - C++ 개발자로서 Rust 보단 C++의 중심에 맞춰 내용이 전개될 예정입니다.
  - Rust를 빙자한 C++ 이야기

02. History of C++ & Rust

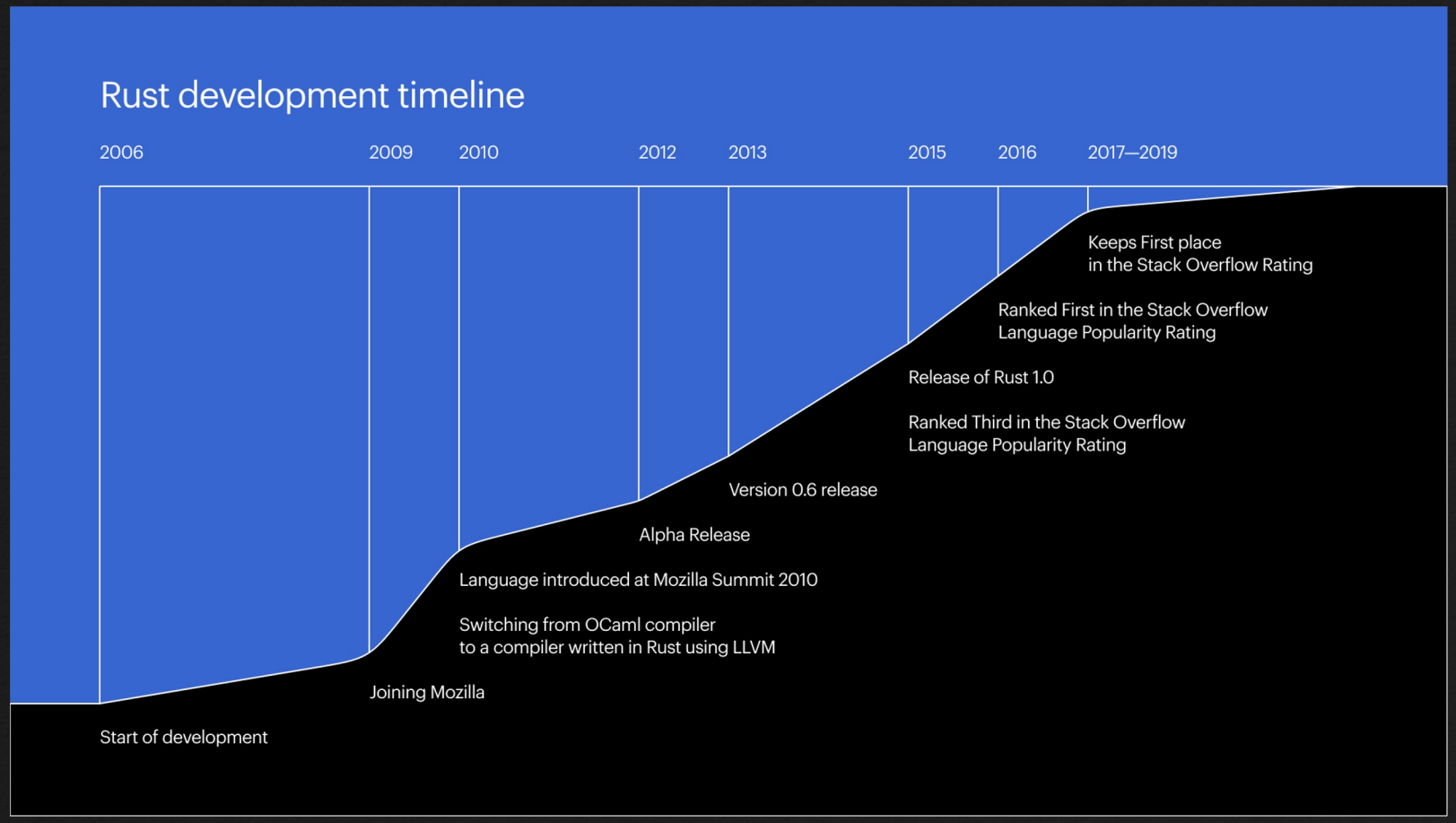
- C++
  - C++은 C++11 이전과 이후로 나뉜다. 3년 주기로 업데이트 되고 있지만 학교에선 여전히 C++11 이전 내용을

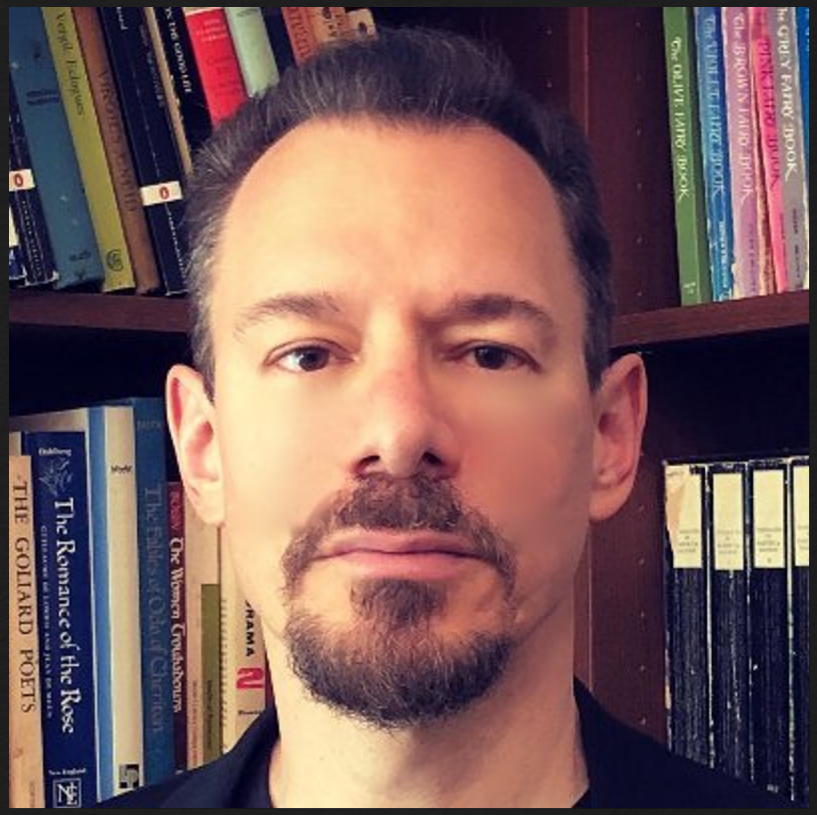




02. History of C++ & Rust

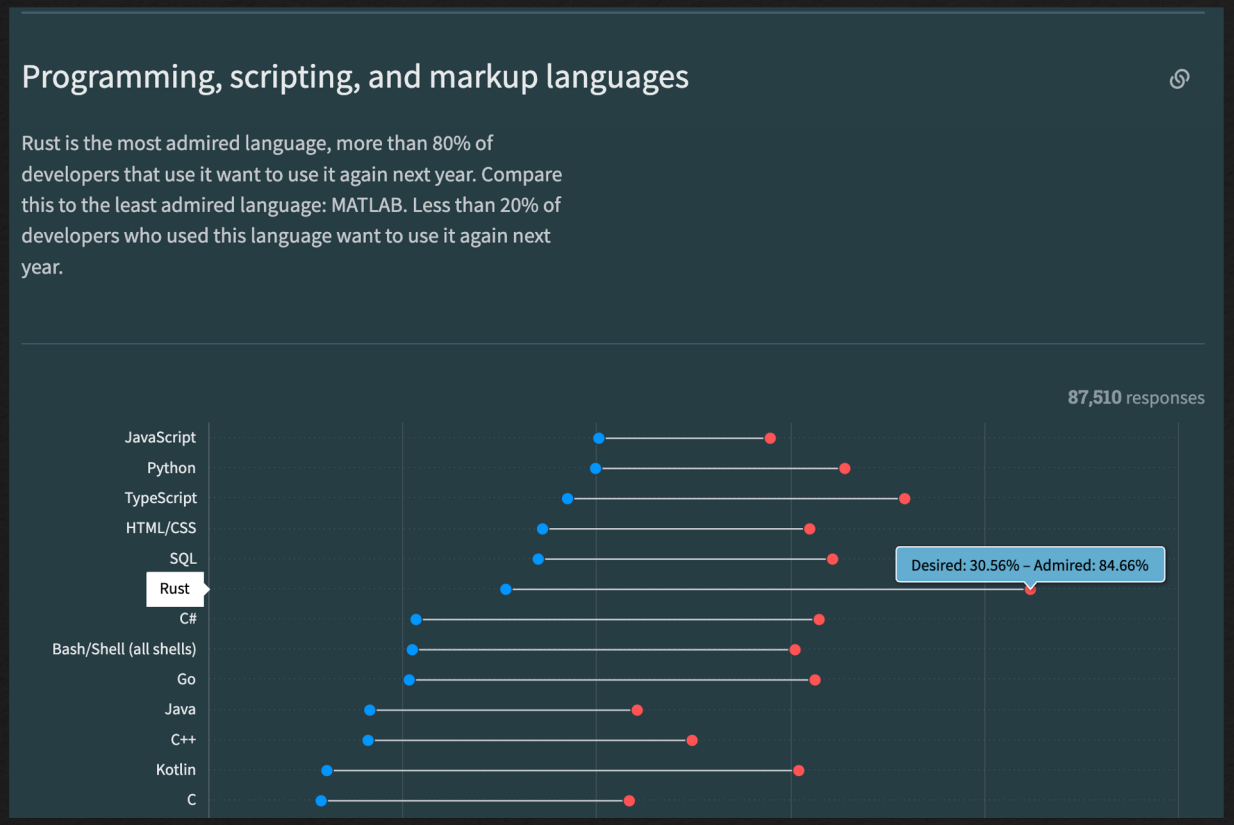
- Rust
  - Graydon Hoare의 취미로 시작,
  - 인터뷰에서 Hoare는 Rust의 타겟층이 "좌절감에 빠진 C++ 개발자"였다고 함

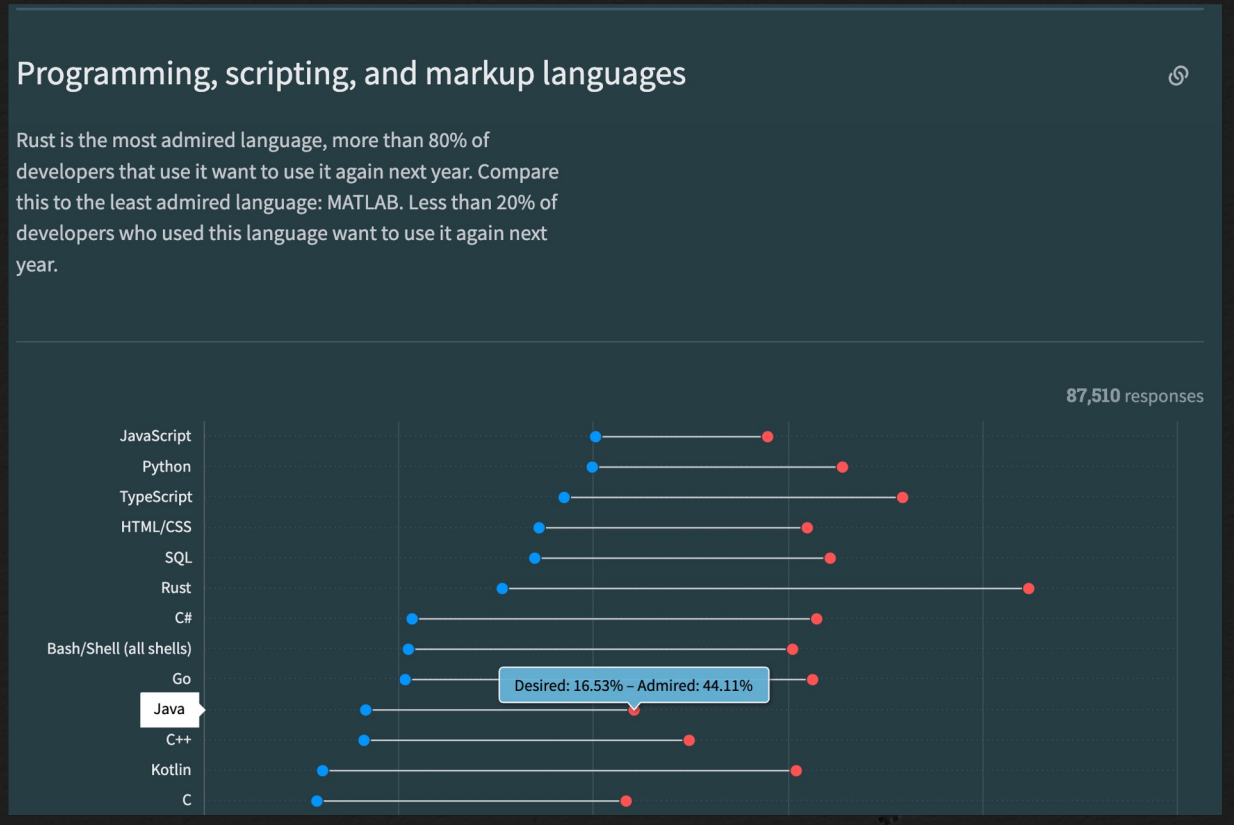




02. History of C++ & Rust

- 2024년 현재
  - “the most desired programming language” in Stack Overflow’s annual developer survey.
  - Admired : Rust 84.66% vs C++ 44.11%





02. History of C++ & Rust

- 2024년 현재
  - “the most desired program oper survey.
  - Admired : Rust 84.66% vs

02. History of C++ & Rust

- TIOBE Index for August 2024 : popularity of programming languages.

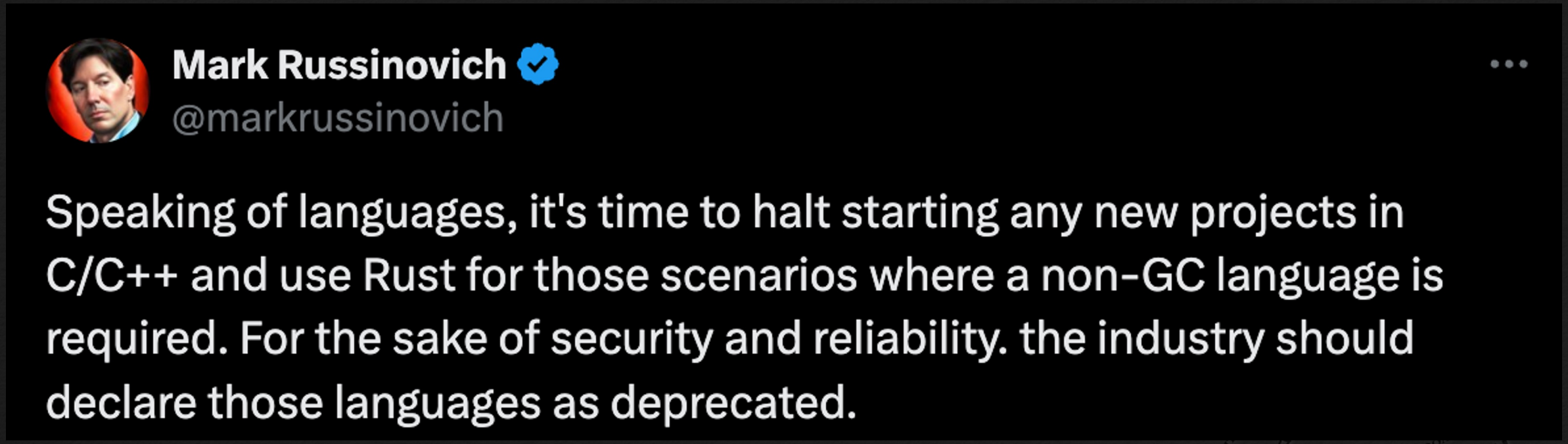
Aug 2024	Aug 2023	Change	Programming Language	Ratings	Change
1	1		Python	18.04%	+4.71%
2	3	↑	C++	10.04%	-0.59%
3	2	↓	C	9.17%	-2.24%
4	4		Java	9.16%	-1.16%
5	5		C#	6.39%	-0.65%
6	6		JavaScript	3.91%	+0.62%
7	8	↑	SQL	2.21%	+0.68%
8	7	↓	Visual Basic	2.18%	-0.45%
9	12	↑	Go	2.03%	+0.87%
10	14	⇈	Fortran	1.79%	+0.75%
14	19	⇈	Rust	1.28%	+0.39%

03. C++ is unsafe?

- 이제 C++ 와 Rust에 대해 이야기 해 봅시다.
    - 왜 Rust는 C/C++ 와 자꾸 비교 대상이 되는가

03. C++ is unsafe?

- 이제 C++ 와 Rust에 대해 이야기 해 봅시다.
    - 왜 Rust는 C/C++ 와 자꾸 비교 대상이 되는가
        - MS Azure CTO의 발언



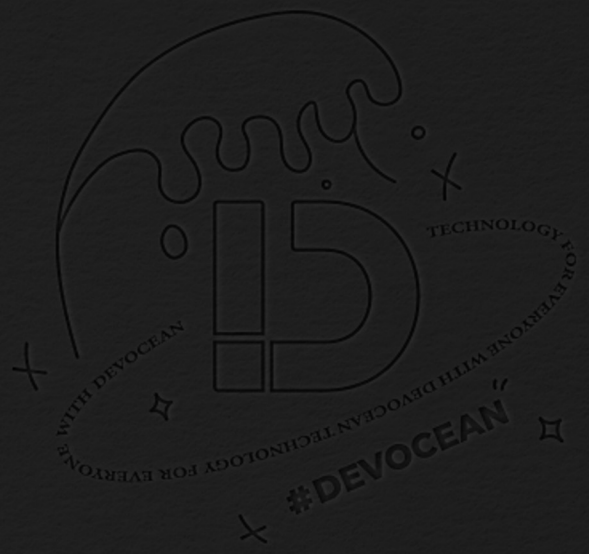
03. C++ is unsafe?

- 이제 C++ 와 Rust에 대해 이야기 해 봅시다.
  - 왜 Rust는 C/C++ 와 자꾸 비교 대상이 되는가
    - 백악관의 권고

InfoWorld

White House urges developers to dump C and C++

Biden administration calls for developers to embrace memory-safe programing languages and move away from those that cause buffer overflows and other memory access vulnerabilities.



03. C++ is unsafe?

- 이제 C++ 와 Rust에 대해 이야기 해 봅시다.
  - 왜 Rust는 C/C++ 와 자꾸 비교 대상이 되는가
    - 이젠 Swift 까지..
      - 애플의 언어 및 런타임 담당 이사인 테드 크레메넥은 6월 10일 애플 세계 개발자 회의의 기조 연설에서 스위프트가 C++를 대체할 수 있는 최고의 프로그래밍 언어라고 주장했습니다

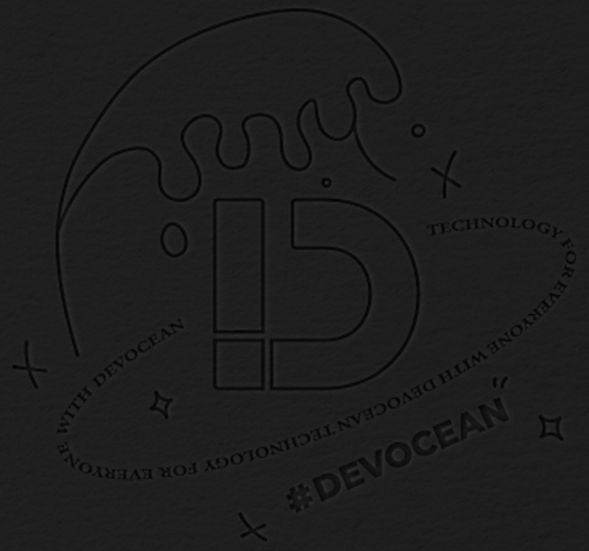
In a June 10 **keynote presentation** at Apple's World Wide Developers Conference, Ted Kremenek, Apple director of languages and runtimes, argued that Swift is the best programming language to replace **C++**.

03. C++ is unsafe?

- C++는 정말 안전하지 않은 언어인가요?
  - 언어적으로 충분히 안전하게 사용 할 수 있는 방법을 제공하고 있음
    - 표준 라이브러리 제공 (smart pointer, atomic, mutex, future/promise, concept …)
    - 다양한 정적 분석 도구 (vagrind, clang-tyde, coverity, cppcheck, ctest …)
    - 핵심 가이드라인 제공
    - 표준이 계속 업데이트 되고 있음
  - 다만, 컴파일 타임에 판단하지 않는다. (개발자가 의도한 것일 수도 있기 때문에)
    - Rust의 경우, 컴파일 타임에 에러가 발생 (의도한 경우엔 unsafe {} 처리를 해야 함)
- Rust는 안전성이 뛰어난 언어로 알려져 있는데, C++와의 비교가 아닌 닮은점을 통해 분석

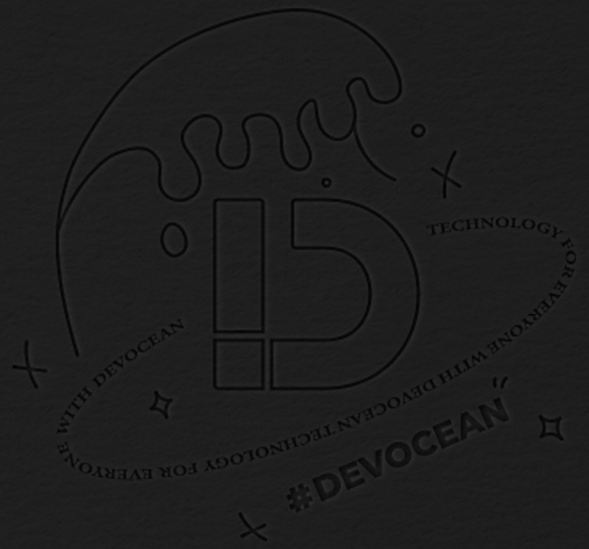
04. Similarities between C++ and Rust

- Rust의 특징
  - 메모리 안전성
    - 소유권 시스템 (Ownership System)
    - 빌림과 수명 (Borrowing and Lifetimes)
  - 안전한 동시성 (Safe Concurrency)
  - 철저한 컴파일 타임 검사
  - 강력한 패키지 관리자 (Cargo)
  - 제로 비용 추상화 (Zero-cost Abstractions)
  - 안전한 포인터 (Safe Pointers)
  - 매크로 시스템



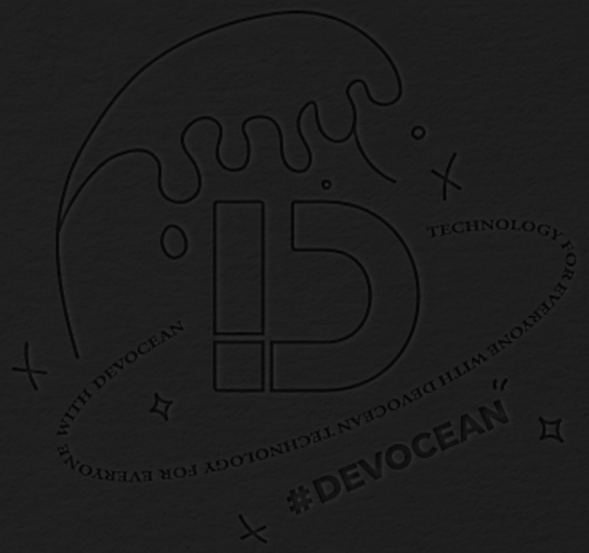
04. Similarities between C++ and Rust

- Rust 와 C++ 닮은점
  - 배우기 어렵다
    - 배우기 어려운 이유
      - 기본적인 CS 지식이 필요
  - OS를 만들 수 있는 언어
    - 메모리에 직접 접근이 가능
  - 컴파일 언어
    - 높은 성능
  - 멀티패러다임 언어
  - 메모리 관리에 대한 제어
    - 안전한 메모리 관리 방법 제공
    - 동시성 처리
  - 제로 비용 추상화
    - 템플릿과 제네릭



04. Similarities between C++ and Rust

- OS를 만들 수 있는 언어
  - 정확히는 OS Kernel을 만들 수 있는 언어
    - Assembly, C, C++, Rust
  - 하드웨어 제어를 위해서는 메모리에 직접 접근이 가능 해야함
    - 메모리 안전에 대한 위험성 증가



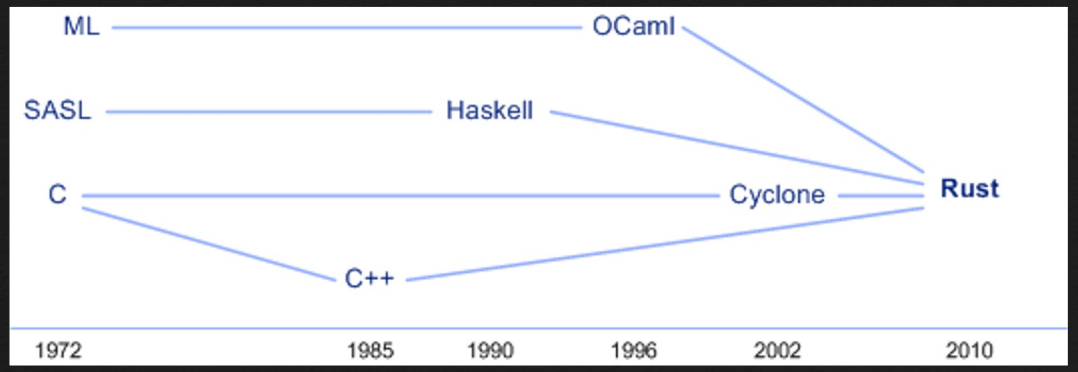
04. Similarities between C++ and Rust

- 컴파일 언어
  - GC 없음
  - 빠른 성능

04. Similarities between C++ and Rust

- 멀티패러다임 언어
  - C++
    - 객체지향에 더 가까운 편
    - C with classes
    - 함수형 패러다임 다수 지원
      - lambda(closures), optional, ranges, concepts, 등등..

04. Similarities between C++ and Rust

- 멀티패러다임 언어
  - Is Rust an object-oriented or a functional language?
    - 

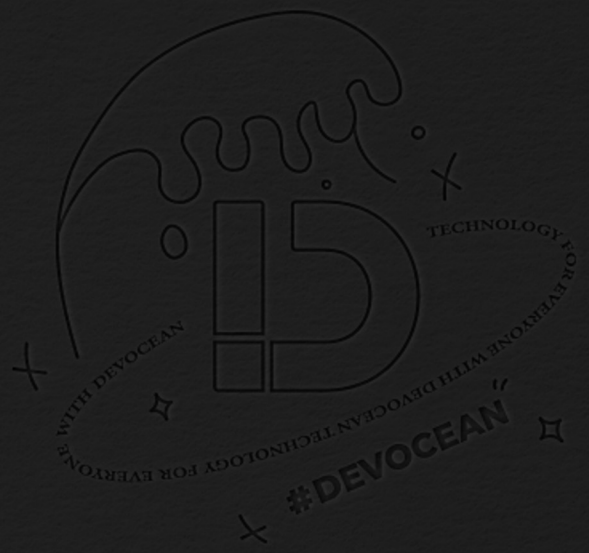
    - Rust is influenced by many other languages
    - FP : ML, SASL, Haskell, OCaml
  - 함수형 언어에 조금 더 가까운 편
    - 상속을 지원하지 않는다.
      - struct 와 trait 을 이용한 다형성 제공
    - 클로저, map, filter, fold 같은 monad 지원
    - 불변성, 패턴매칭 등등..

04. Similarities between C++ and Rust

- 메모리 관리에 대한 제어
  - 소유권 및 빌림 개념
    - 소유권 이전 기능 둘 다 제공
      - Rust는 기본 기능이 Move
      - C++의 경우 std::move를 이용 (컴파일러가 결정)
    - 기본적인 운영체제의 Memory model을 알아야 함
      - memory leak은 왜 발생하는가?
        - stack과 heap의 차이를 알아야 함
  - 동시성
    - 멀티스레딩
      - Race condition이 발생하는 이유
        - Rust의 불변성 컴파일 타임에 에러로 잡힘
        - C++는 런타임 에러 발생

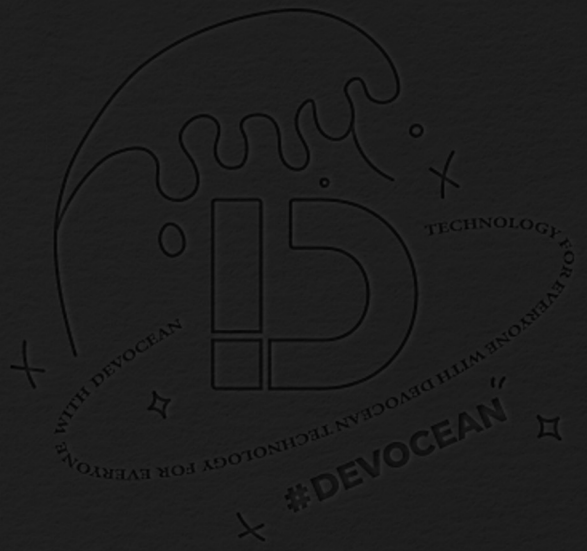
04. Similarities between C++ and Rust

- 제로 비용 추상화
  - 컴파일 타임에 최적화
  - C++
    - 템플릿 메타 프로그래밍, 메크로
    - concepts 를 통해 안전하게 사용하는 방법 지원 중
  - Rust
    - 제네릭과 트레이트, 메크로
    - 컴파일 타임에 에러를 잡아줌



05. Pros & Cons of Rust

- 개인적으로 느낀 Rust의 좋은점
  - 패키지 매니저 Cargo
  - 컴파일 타임에 에러를 잡아 준다.
  - 빠른 성능
  - 간결한 코드
  - FFI (Foreign Function Interface)



05. Pros & Cons of Rust

- 개인적으로 느낀 Rust의 안 좋은점
  - 빌드 시간이 오래 걸린다
    - 컴파일 타임에 모든 타입을 검사해야 하기 때문
  - 높은 진입 장벽
    - 트레이트, 패턴매칭, 라이프 타임, enum 등, 햇갈리고 가볍게 쓰기 어려운 개념을 잘 써야 빌드가 됨
  - unsafe
    - rust의 핵심 기능으로 소개하고 있지만, unsafe는 C++과 다르지 않음.
    - 하지만 실 은 기능

06. Conclusion

- C++, Rust 각자의 장점과 개성이 있는 언어
- Rust는 C++의 대안이 될 수 있는가?
    - Java - Kotlin
    - Objective C - Swift
    - C - C++
    - C++ - Rust
- Rust 와 C++의 미래?
    - AI 생태계
        - Python의 성능 이슈가 조금씩 부각되고 있음
        - C++, Rust가 대안이 될 수 있을 듯?
            - Cuda, TensorRT, llama.cpp 등등
    - WebAssembly
    - Embedded system
        - 전장 소프트웨어 등등

06. Conclusion


Q&A


TECHNOLOGY FOR EVERYONE WITH DEVOCEAN
#DEVOCEAN