시니어와 주니어의 협업 다리
온라인 및 오프라인 Pair Coding의 통찰

정동진 최재웅 / NAVER


정동진

- Solution, SI, SM, Naver Cafe
- 신기술 관심이 많습니다.
- 자동화 매우 좋아합니다.
- 성장에 관심이 많습니다.

최재웅

- Naver Cafe 안드로이드 개발자
- MBTI: ISTJ
- 설계에 대해 고민하고
  얘기하는 걸 좋아함
- 최신 기술 써보는거 좋아함

질문

https://bit.ly/4c2LAaO

slido : #3819 210

오늘 할 이야기

1. Pair Programming 소개
2. Pair Work Benchmarking
    a. 작업 환경 / 진행 방법
    b. 회고 내용 정리
3. 시니어 입장에서 느낀 점
4. 주니어 입장에서 느낀 점

Pair Programming 소개

- 두 명이 하나에 컴퓨터로 같이 작업
  - Navigator - Driver 방법
  - 주기적으로 역할을 변경 (5분/10분)

Pair Programming 소개

- 장점
  - 결함이 준다.
  - 통합 시간이 줄어든다.
- 단점
  - 개발에 시간이 늘어난다.

Navigator - Driver 방법

- Driver
  - 키보드로 직접 작업하는 사람
  - Driver는 자신이 coding하는 이야기 하면 됩니다.

Navigator - Driver 방법

- Navigator
  - 코드 분석
  - 아이디어 제시
  - 오류를 찾기
  - 생각하는 방향 전달
  - Driver에 의도 확인

주의 할 점

- 동등한 관계
  - 시니어/주니어, 작업자/비 작업자
  - Pair 하는 동안 자신에 코드인것 처럼
- 사전 준비(목표, 작업내용)
  - 간단한 리뷰 후 시작

주의 할 점

- 적극적인 자세

주의 할 점

- 적절한 휴식

Pair Work Benchmarking

Pair Programming 진행 방법

- 온라인 / 오프라인
- Navigator - Driver 방법
- 5분 단위 교대, 10분 단위 교대
- 온라인 : 허들, code with me, 원격제어
- 회고 : 좋았던 점 / 아쉬운 점

Pair coding Tools

1. 화면 공유 툴
    a. 공유를 하는 쪽에서만 작업이 가능하다
2. 온라인 협업 도구
    a. Coding 작업에 최적화 되어 있음
3. 원격제어 툴
    a. 속도가 빠름
    b. 전체 화면을 공유받기 때문에
    상대방에게 의도를 비교적 쉽게 이해

화면 공유 툴

- 허들(Slack), Google Meet
  - 여러 사람이 동시에 화면공유가 가능
  - 간단한 화면 주석 달기 기능
- Zoom
  - 한 번에 한 사람만 화면공유가 가능함
  - 화면 주석 달기 기능 우수

온라인 협업 툴

- code with me
  - pair coding mode에서는 스크롤에 문제가 있음
  - 게스트 쪽에서 자동완성 및 다이얼로그가 안 보임
- Duckly
  - Build 오류 내용이 안 보임
  - IDE color가 안 나옴
  - 처음 설정이 힘들었음

온라인 협업 툴

- 게스트쪽 문제점
  - 디자인 화면 볼 수 없었음
  - build 사용이 불가능
  - 팝업 메뉴가 안보인다

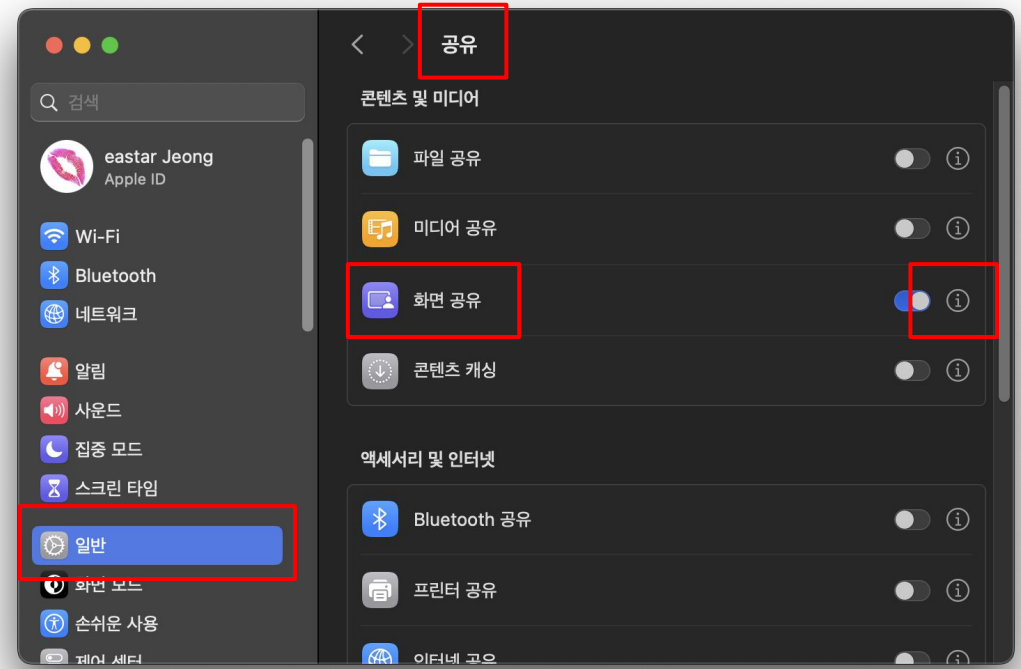
원격제어 툴

- Zoom 원격 제어
  - 대소문자 변환에 문제가 있었음
- Chrome 원격 데스크톱
  - 쉽고 간단한 설정, 브라우저에서 사용
  - 딜레이가 있음
  - 뒤로 가기 버튼이 눌리면 종료됨

OS 기본 제공 원격제어 툴

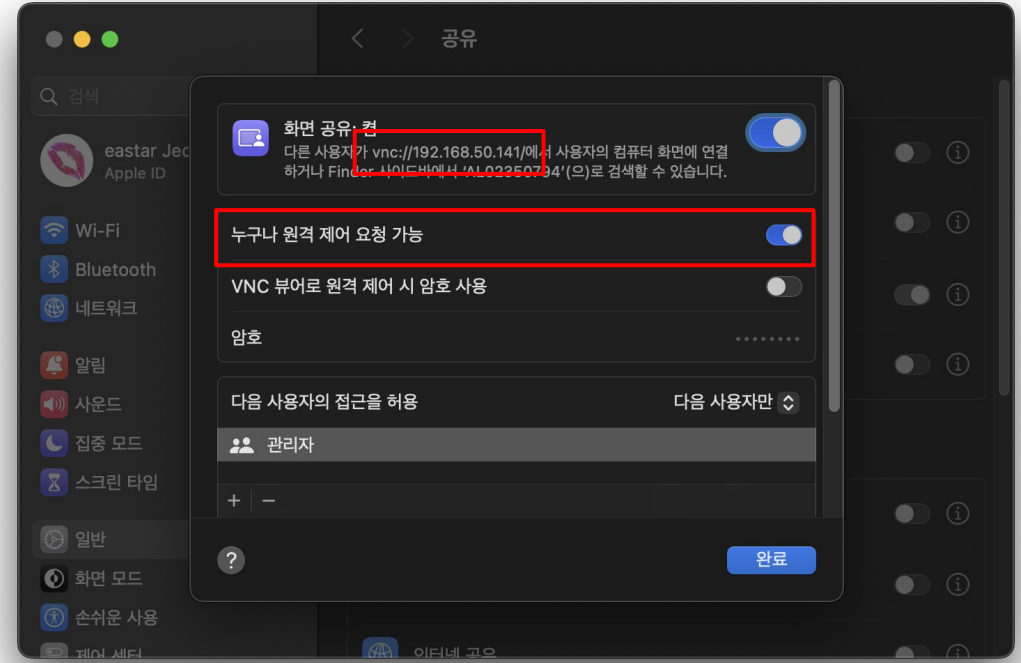
OS 기본 제공 원격제어 툴 - 설정

설정 > 일반 > 화면공유



OS 기본 제공 원격제어 툴 - 설정

누구나 원격제어 요청 가능



OS 기본 제공 원격제어 툴 - 설정

- 공유기를 사용하는 경우 포트 포워딩이 필요함
- 5900 포트 사용

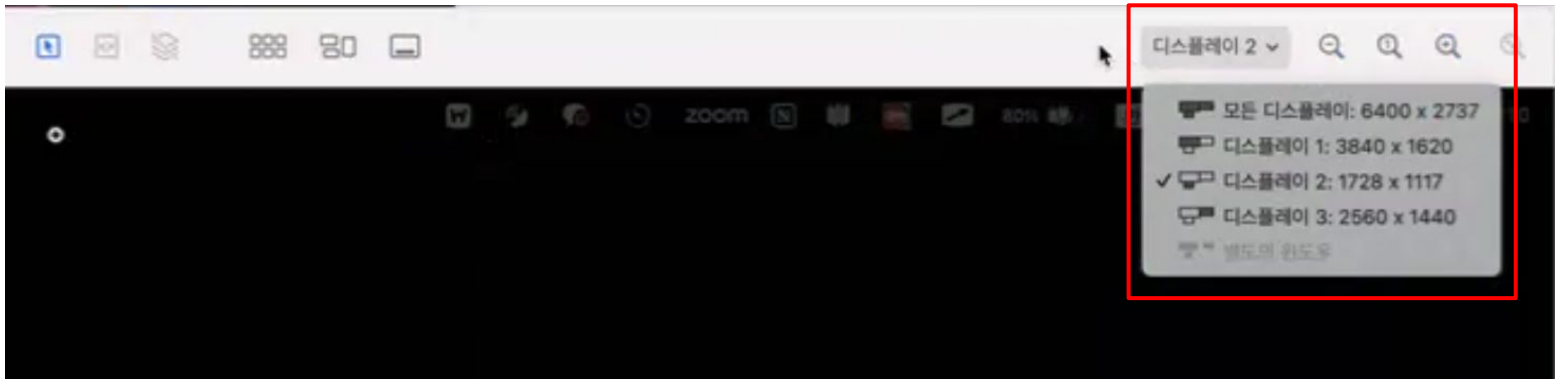
외부포트	내부 IP 주소	내부 포트	프로토콜
5900-5900	[redacted]	5900-5900	TCP

OS 기본 제공 원격제어 툴 - 접속

- 접속은 간단하게 URL로 가능
  - vnc://xxx.xxx.xxx.xxx
- 화면 공유 앱을 통해서도 접속 가능
  - Launchpad > 기타

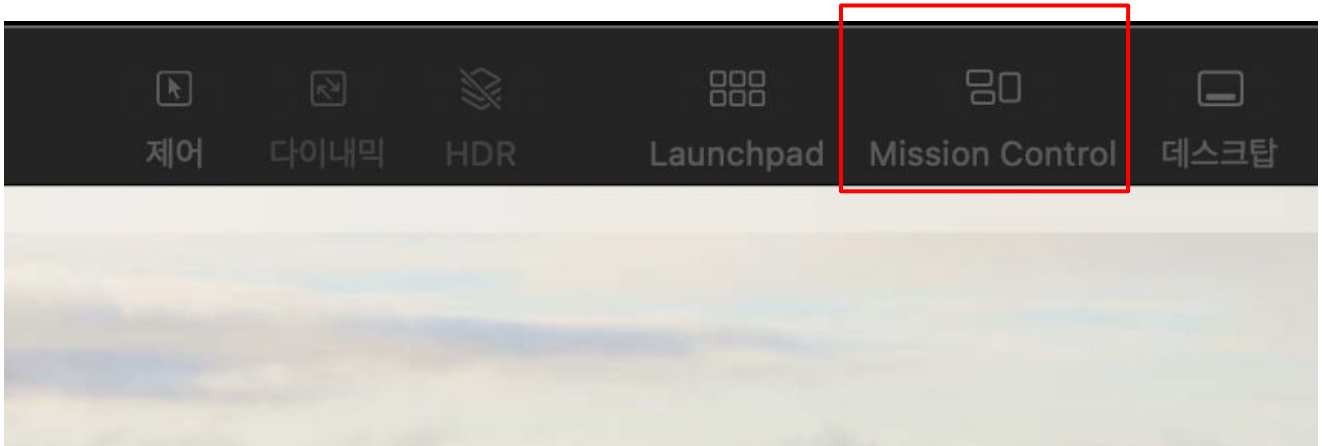
OS 기본 제공 원격제어 툴 - 사용

- 다중화면일 경우 원하는 Display 선택가능



OS 기본 제공 원격제어 툴 - 사용

- 원격 제어 기능
- Mission Control



Mac OS 기본 제공 원격제어 툴

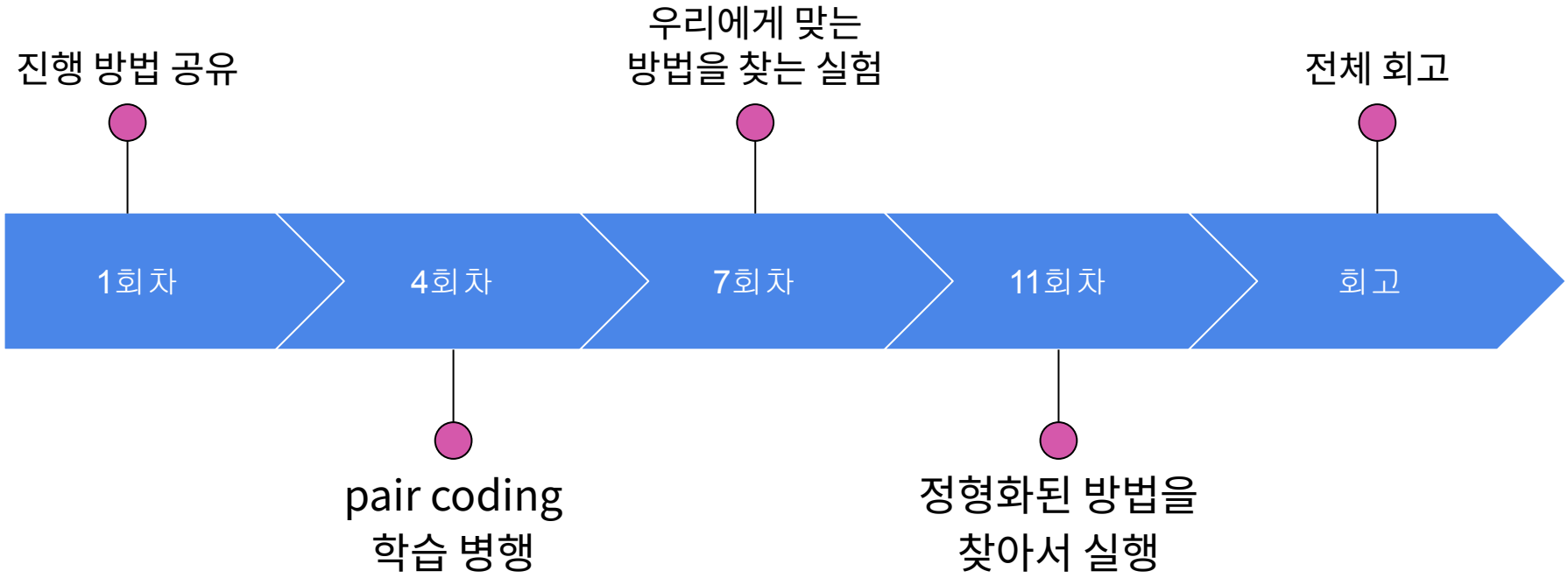
- 장점
  - OS에서 기본 제공하는 Tool이라서 별도에 설치가 필요 없음
  - ID를 몰라도 승인 과정을 거쳐 들어올 수 있음
  - 반응성이 좋음
- 단점
  - 최초 공유기 포트 포워딩 설정이 필요함

Offline

- 모니터를 두사람의 가운데 놓고 진행
- 자신에게 좀 더 익숙한 장비를 사용
  - 각자의 마우스
  - 각자의 키보드

어떤 작업을 해봤나?

진행 단계



History

- 1회차 : Pair Programming 계획 공유
  - Tools, 진행 방법
- 2회차 : 체험학습?
  - 5분 단위에 역할 교대
  - pair coding 툴을 사용했고, 익숙해지는 시간이 필요했음
  - Driver로 있을 때보다, Navigator 역할을 할 때 뭘 할지 익숙지 않음

History

- 3회차 : Offline Pair Programming
  - 양방향, 빠른 피드백
  - Online보다 많은 작업, 의사 표현
- 4회차 : 서로 잘 모르는 작업
  - 원격제어 생각보다 좋았음
  - Merge까지 작업이 쉽지 않음

History

- 5회차 : 서로 잘 아는 작업
  - 10분 단위에 역할 교대
  - 2시간 정도 까지 괜찮은 것 같음
- 6회차 : 단순하고 지루한 작업
  - 생각보다 덜 지루함
  - 한번 정도는 해보는것이 좋음

History

- 7회차 : 한쪽에 불편함을 주는 시도
  - Driver가 아무 말도 안 하는 경우
  - Navigator가 아무 말도 안 하는 경우
- 8회차 : 한쪽만 잘 아는 작업
  - 각자 하나씩 준비
  - 사전에 코드 리뷰 시간 후에 진행

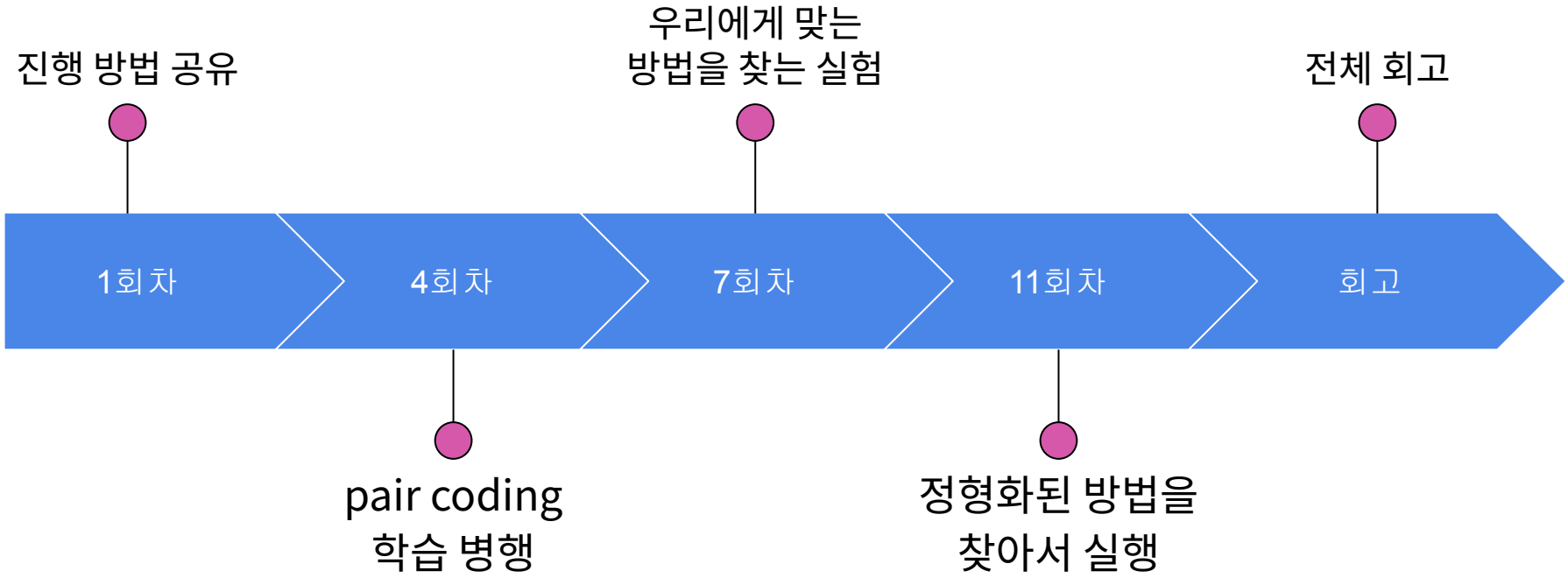
History

- 9회차 : 코드 사전 리뷰
  - 서로 아주 잘 알고 있는 코드로 만들기 위해
    별도 코드 리뷰 시간을 가짐
- 10회차 : 리팩토링
  - 소스 파일에 readme.md 파일을 만들어 정리
  - 모듈 분리, API 결과를 Flow로 전환, Compose 적용

History

- 11회차 : 신규 작업, 아키텍처 작업
  - 사전에 코드 리뷰 시간 후에 진행
- 12회차 : 회고
  - 전체 회고 및 총평

진행 단계

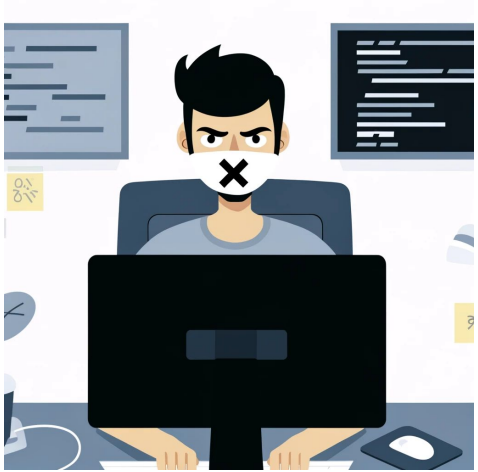


History

- 7회차 : 한쪽에 불편함을 주는 시도
  - Driver가 아무 말도 안 하는 경우
  - Navigator가 아무 말도 안 하는 경우
  - 각 1번씩 교대로 진행

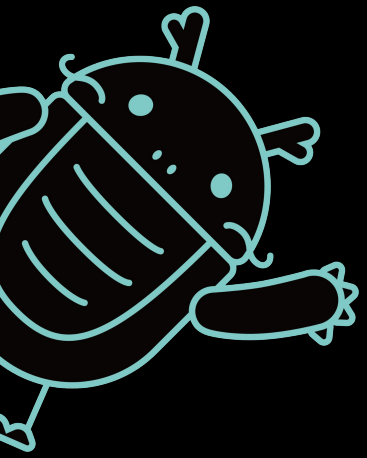
Driver가 수동적인 경우

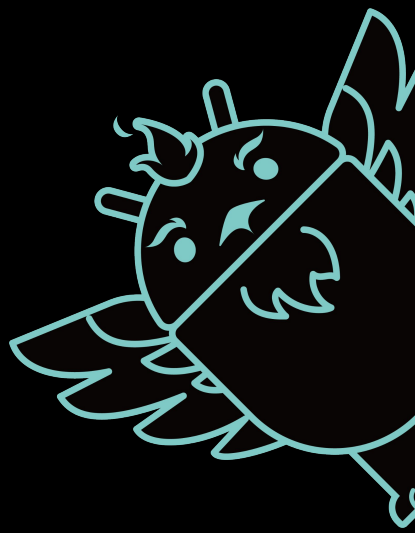
- navigator
  - 어떤 의도로 coding하는지 알 수 없어 답답함
  - 오더를 잘 이해하고 coding하는 건지 모름
- driver
  - 오더에 의한 coding



Navigator가 수동적인 경우

- driver
  - 혼자 coding하는 듯한 느낌
  - 익숙한 코드가 아닌 경우는 뭘 해야 할지 몰랐음
  - 혼자 말하는 느낌이 들어 좀 더 말을 덜 하게 됨
- navigator
  - coding 영상을 보는 느낌
  - 집중력이 떨어짐





소통

회고 내용 정리

효율적인 작업 방법인가?

- 혼자 할 때보다 시간이 줄어 들었나?
  - 30% 정도 줄었음
- 통합에 걸리는 시간은?
  - Review 시간이 대폭 감소 됨
- 버그 발생은 얼마나 줄까?
  - 30% 줄어듦

효율적인 작업 방법인가?

- 눈에 보이는 효과
  - 시간비용이 큰 작업에 경우
  - hotfix 대응
  - 장애 대응

효율적인 작업 방법인가?

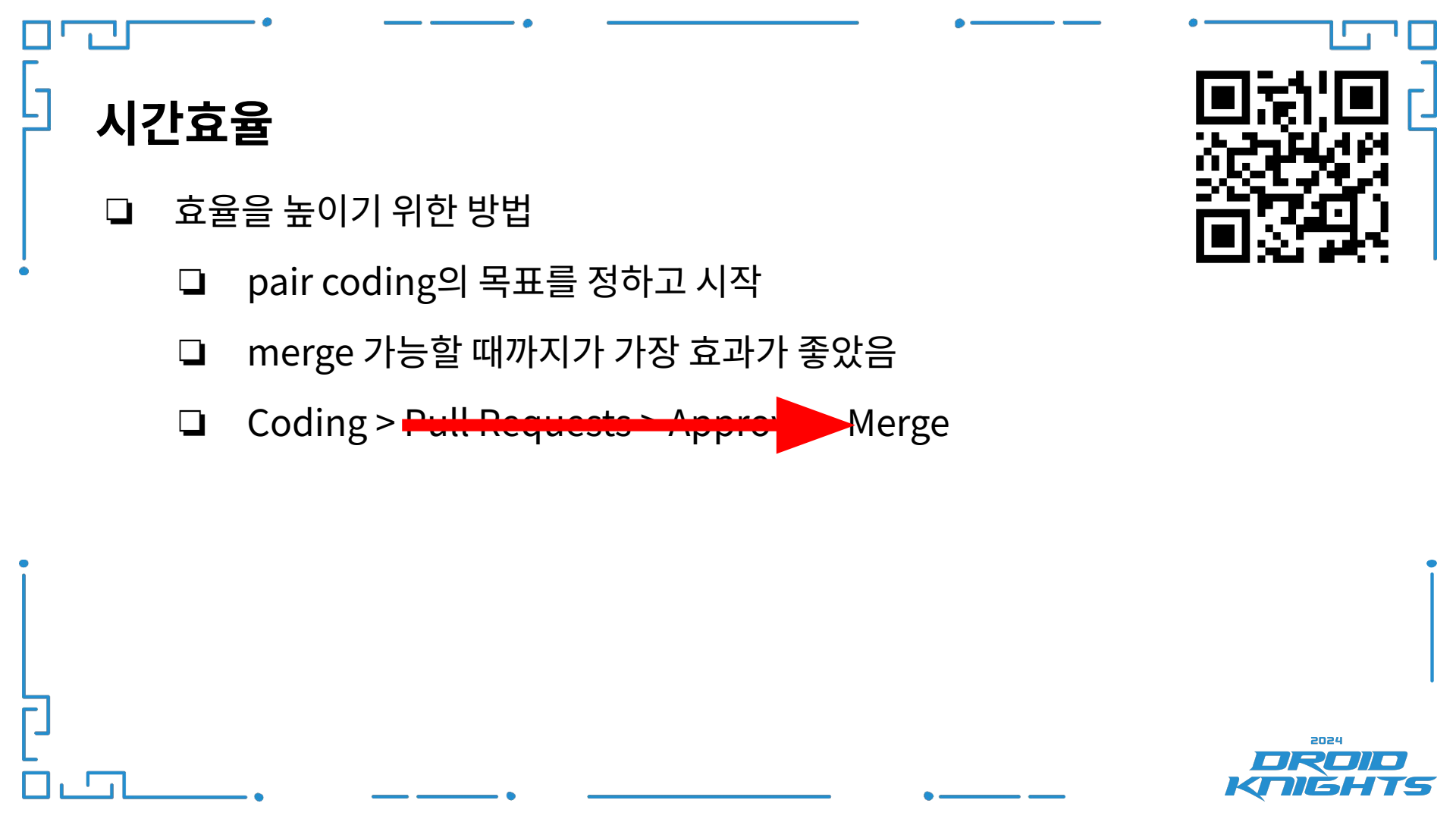
- 당장 보이지 않는 효과
  - pair coding으로 성장한다.
  - 코드 스타일이 비슷해진다.

시간효율

- 시간적 측면에서 본 pair coding
  - 장기적 관점에서 확실히 이득 단기적 관점에서는?
  - 같은 서비스를 오래 할수록 효과가 커진다.

시간효율

- 효율을 높이기 위한 방법
  - pair coding의 목표를 정하고 시작
  - merge 가능할 때까지가 가장 효과가 좋았음
  - Coding > ~~Pull Requests > Appro~~ > Merge



Pair Programming에 적합한 일은 뭘까요?

1. 일정 시간 이상에 리뷰가 필요한 작업
2. 도전적인 과제
    a. 리팩토링, 구조개선
3. 신규 작업
4. 2명이상 투입 과제
5. 겹치는 부분이 많은 과제

좀 더 어려운 경우가 있었나요?

- 단축키 커스텀을 많이 해 놓은 다른 사람 장비에서 할 때 힘들다

적절한 시간은?

- 10분 단위 교대 괜찮았음
- Build 할 때 5분이 넘어가는 경우도 있어서
  교대 간격이 좀더 길었으면 좋겠음
- 5분 단위로 교대 하는 것 자체는 괜찮았음
- 오랜 시간 같이 하는 것에 대한 의견
  - 1:30 coding, 30회고, 2시간 정도는 괜찮음

Pair Programming 전제

1. pair programming 시간이 늘어나는 일이다.
2. 시간을 단축하는 것을 목표로 하면 안 된다.
3. 단기적인 효과를 기대하고 할 수는 없다
4. 장기적으로 팀원에 역량을 높이기 위해서 진행하는 것이 올바르다

주니어의 입장

주니어 입장

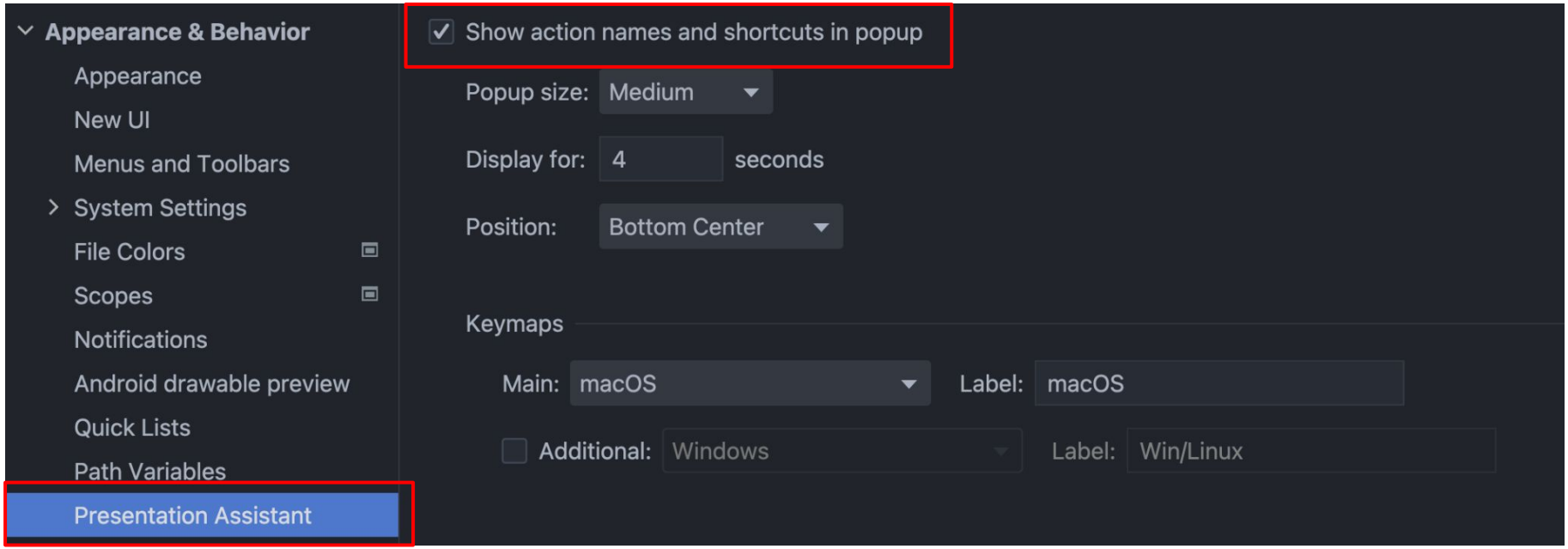
1. 시니어의 추론 과정을 지켜볼 수 있음.
2. 코드 스타일을 통일 할 수 있음.
3. 다양한 도구 노하우 습득.
4. 코드 리뷰에 적응할 수 있음.
5. 자신감이 오름.

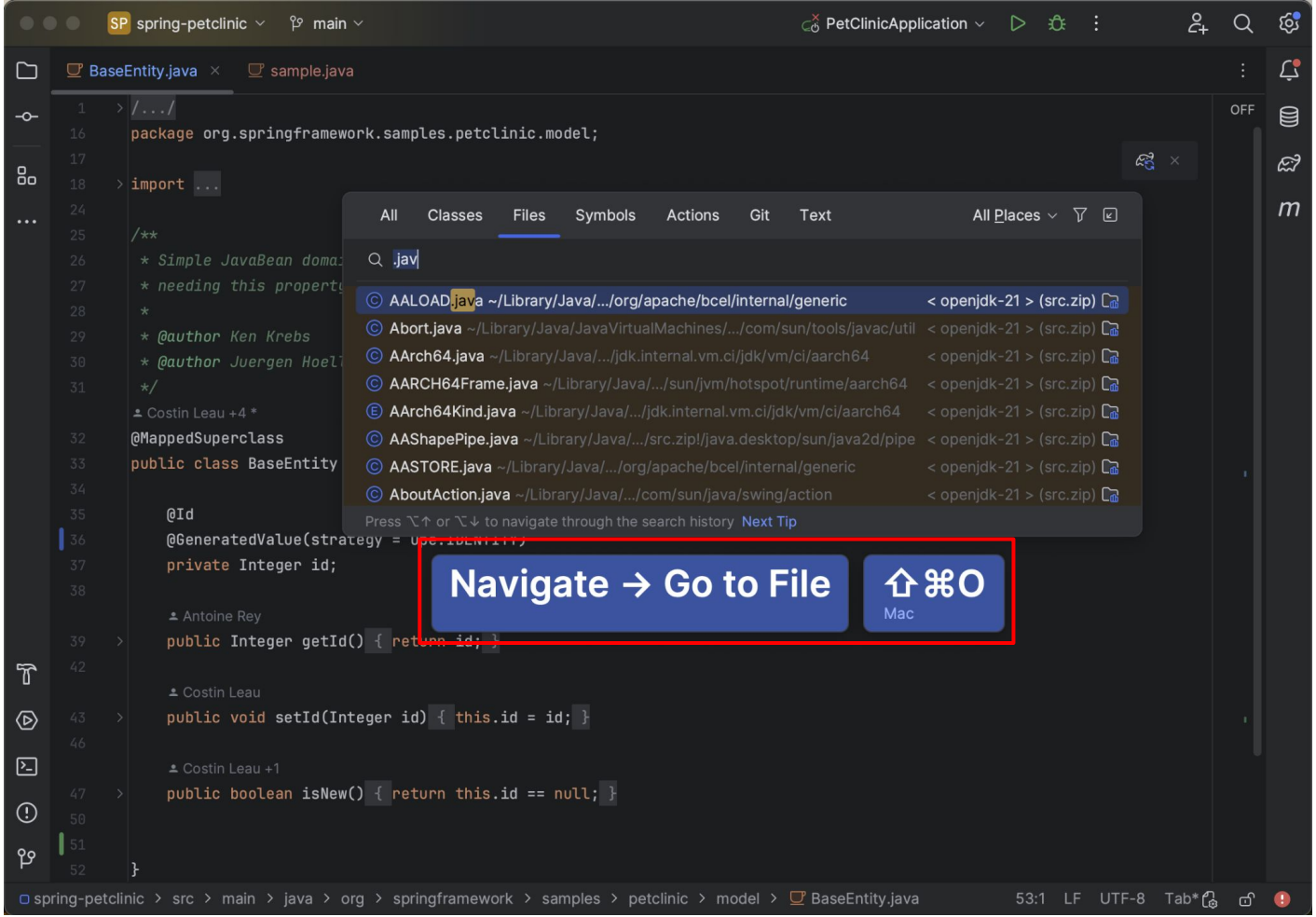
주니어 입장

1. 시니어의 추론 과정을 지켜볼 수 있음.
2. 코드 스타일을 통일 할 수 있음.
3. 다양한 도구 노하우 습득.
4. 코드 리뷰에 적응할 수 있음.
5. 자신감이 오름.

팁

1. 폰트 크기 조절
    a. Ctrl + Shift + , 폰트 줄이기
    b. Ctrl + Shift + . 폰트 키우기
2. 플러그인에서 IDE 기본 탑재
    a. Presentation Assistant
    b. https://www.jetbrains.com/help/idea/presentation-assistant.html





주니어의 입장

1. 시니어의 추론 과정을 지켜볼 수 있음.
2. 코드 스타일을 통일 할 수 있음.
3. 다양한 도구 노하우 습득.
4. 코드 리뷰에 적응할 수 있음.
5. 자신감이 오름.

시니어의 입장

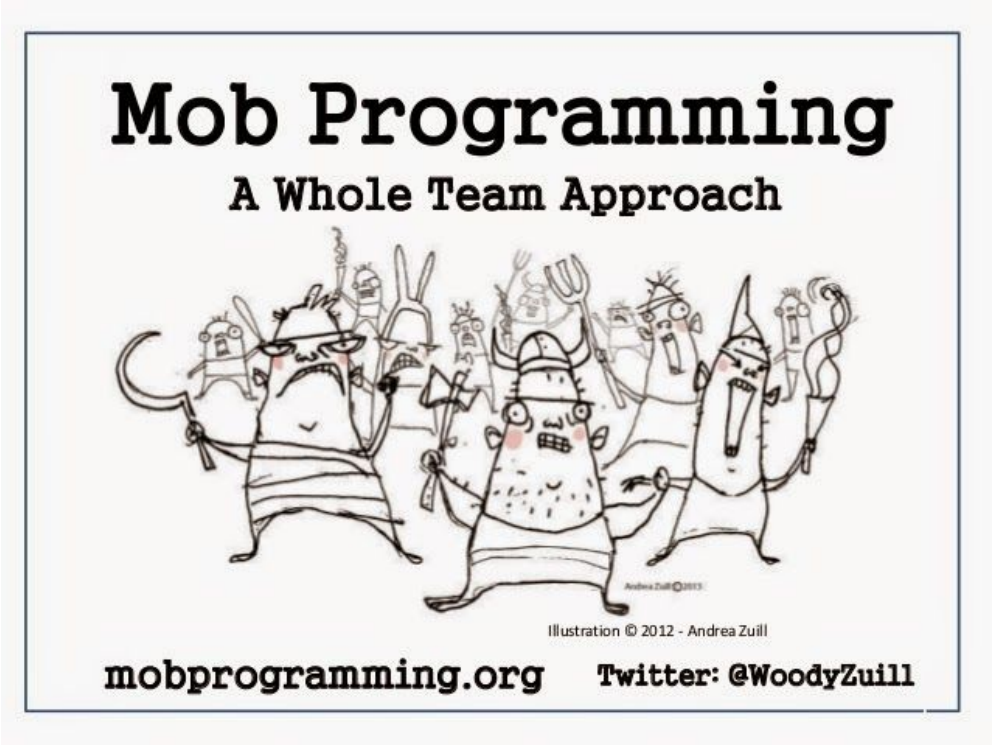
시니어 입장

- 시니어도 성장한다.
  - 시니어도 모를 수 있다.
- 암묵지 제거
  - 나에게는 당연하지만…

시니어 입장

- 소프트 스킬
  - 코드 설명, 더 좋은 방법에 대한 설명
  - 통찰력, 이해하기 쉽게 말하기
  - 성장시키기 기술

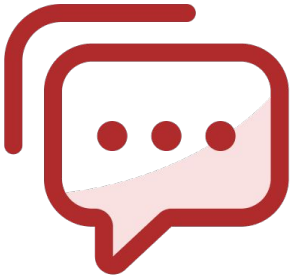
몹 프로그래밍


Mob Programming
A Whole Team Approach
Illustration © 2012 - Andrea Zuill
mobprogramming.org
Twitter: @WoodyZuill

참고자료

- Pair/Mob Programming
  - https://gmlwjd9405.github.io/2018/07/02/agile-pair-programming.html
  - https://www.podbbang.com/channels/14757/episodes/22408410
  - https://mobprogramming.org
- 협업툴
  - https://www.jetbrains.com/ko-kr/code-with-me
  - https://duckly.com
- Plugin
  - https://www.jetbrains.com/help/idea/presentation-assistant.html
- Timer
  - https://apps.apple.com/kr/app/be-focused-작업을-위한-포모도로-타이머/id973134470

slido



Audience Q&A Session

ⓘ Click **Present with Slido** or install our Chrome extension to show live Q&A while presenting.

2024 DROID KNIGHTS

부록 : 미처 못다 한 이야기

서로 잘 모르는 코드

1. 학습 및 시간 소요가 줄어드는지 잘 모르겠음.
2. Merge까지 가하기 쉽지 않음.
    a. Merge까지 할 때 시간 효율이 좋아지는데
    b. Merge까지 못 가면 효율이 떨어짐.

둘다 잘 아는 작업

1. 피드백에 빨랐음.
2. 헤매는 시간이 없었음.
3. 좀 더 많은 양을 한껏 같은 느낌이 듦
4. 실패 시에 다른 대안을 시작하는 게 빨랐음.
5. 알긴 하는데 한 번도 안 써본 것을 상대방이 잘 써서 관련 부분 학습효과가 높음
    a. 모르는 부분 hilt binds 학습이 용이했음
    추가적인 질의도 할 수 있었음
    b. EntryPoint 관련 학습

단순작업에 효과가 있나?

1. 생각보다 덜 지루함. 하기 싫은 일임에도 조금도 할만함.
2. 알 수 없는 오류를 빨리 찾음.
3. 혼자 할 때보다는 집중이 잘됨
4. 진행 속도는 혼자 하는 것보다 같거나 빠름
5. 처음 하는 작업이면 익숙해지는 데 도움이 됐지만 2번째는 별로임