

# The Pragmatic Programmer

<https://product.kyobobook.co.kr/detail/S000001033128>

자신의 기예에 관심을 가져라

자기 일에 대해 생각하라

카이젠 : 꾸준히 조금씩 개량한다는 일본어

나는 당신의 기대대로 살기 위해 이 세상에 있는 게 아니고, 당신도 내 기대대로 살기 위해 이 세상에 있는 게 아니다.

약점을 보이는 것에 대한 두려움이 가장 큰 약점이다.

어설픈 변명 말고 대안을 제시하라.

- 팀 내 신뢰, 책임지기 등

기술 부채

깨진 창문을 내버려 두지 말라

우선 망가트리지 말라.

시작 피로

변화의 촉매가 되라



큰 그림을 기억하라

상황 인식

우리는 종종 뭔가 나아지게 하려다가 괜찮은 것 마저 망친다

타협 과정에 사용자를 참여시켜라

멈춰야 할때를 알라

지식에 대한 투자가 언제나 최고의 이윤을 낸다.

포트폴리오 만들기

- 주기적인 투자, 다각화, 리스크 관리, 싸게 사서 비싸게 팔기, 검토 및 재조정,

목표

- 매년 새로운 언어를 최소 하나는 배워라
- 기술 서적을 한 달에 한 권씩 읽어라
- 기술 서적이 아닌 책도 읽어라
- 수업을 들어라
- 지역 사용자 단체나 모임에 참여하라
- 다른 환경에서 실험해 보라
- 요즘 흐름을 놓치지 말라

읽고 듣는 것을 비판적으로 분석하라

- 왜냐고 다섯번 묻기
- 누구에게 이익이 되나
- 어떤 맥락인가



- 언제 혹은 어디서 효과가 있을까
- 왜 이것이 문제인가

나는 무시당하느니 차라리 살짝이 훑어보는 시선이 낫다고 봐요

한국어든 영어든 하나의 프로그래밍 언어일 뿐이다.

청중을 알라

말하고 싶은게 무언지 알라

때를 골라라

스타일을 골라라

청중을 참여시켜라

경청하라

응답하라

코드와 문서를 함께 뒤라

좋은 설계는 나쁜 설계보다 바꾸기 쉽다

Don't Repeat Yourself

재사용하기 쉽게 만들어라



직교성

자족적인 컴포넌트

관련없는 것들 간에 서로 영향이 없도록 하라

생산성 향상

리스크 감소

코드의 결합도를 줄여라

전역 데이터를 피하라

유사한 함수를 피하라

가역성

당신이 가진 생각이 딱 하나밖에 없다면, 그것만큼 위험한 것은 없다.

유행을 좇지 마라.

사용자 인터페이스 < 권한 부여 < 비즈니스 로직 < 데이터 모델 < 데이터 베이스

예광탄 코드

- 사용자가 뭔가 작동하는 것을 일찍부터 보게 된다
- 개발자가 들어가서 일할 수 있는 구조를 얻는다
- 통합 작업을 수행할 기반이 생긴다
- 보여줄 것이 생긴다



- 진행 상황에 대해 더 정확하게 감을 잡을 수 있다

프로토타이핑으로 학습하라

프로토타이핑 대상

- 아키텍처, 기존 시스템에 추가할 새로운 기능, 외부 데이터의 구조 혹은 내용, 성능 문제 등

프로토타이핑

- 정확성, 완전성, 안정성, 스타일

언어의 한계가 곧 자기 세계의 한계다

추정으로 놀람을 피하라

점증적 개발

PERT

코드와 함께 일정도 반복하야 조정하라

지식을 일반 텍스트로 저장하라

유닉스 철학

자신만의 셸

- 색깔 조합, 별칭과 셸 함수, 명령어 자동 완성, 프롬프트 설정



진보라는 것은 변화와는 거리가 멀고 오히려 기억에 의존한다. 과거를 기억하지 못하는 사람은 과거를 반복할 운명이다.

버전 관리 시스템 (VCS)

브랜치

디버깅

가장 속이기 쉬운 사람은 자기 자신이다.

여러분은 완벽한 소프트웨어를 만들 수 없다.

상식과 정직만큼 사람을 놀라게 하는 건 없다.

계약에 의한 설계 (DBC)

테스트 주도 개발 (TCC)

자신이 시작한 것은 자신이 끝내라

헤드라이트(투사거리)를 앞서가다.

작은 단계들을 밟아라. 언제나.

결합도

우리가 어떤 것 하나만을 골라내려고 해도, 그것이 우주의 다른 모든 것과 얽혀있음을 깨닫게 된다.



열차 사고

데메테르 법칙

메서드 호출을 엮지 마라

전역적이어야 할 만큼 중요하다면 API로 감싸라

그냥 일어나는 일은 없다. 일어나도록 만들어진 것이다.

유한 상태 기계

감시자 패턴

게시-구독

반응형 프로그래밍과 스트림

자신이 하고 있는 걸 하나의 과정으로 서술할 수 없다면, 자기가 뭘 하고 있는지 모르는 것이다.

프로그래밍은 코드에 관한 것이지만, 프로그램은 데이터에 관한 것이다.

상속세

모든 물건은 제자리에 두고, 일은 모두 때를 정해서 하라.



정적 설정

서비스형 설정

동시성

병렬성

시간적 결합

동기화 막대

비-원자적 갱신

세마포어

슈퍼 비전 시스템 > 핫 코드 로딩

린다 > 자바스페이스, 티 스페이스

오직 인간만이 무언가를 직접 보고, 정확한 예측에 필요한 모든 정보를 획득하고, 심지어 순간적으로는 정확한 예측을 한 후에도, 그런데 그것이 아니라고 말할 수 있다.

우연에 맡기는 프로그래밍을 하지 말라

- 구현에서 생기는 우연, 유형 패턴, 상황에서 생기는 유형

추정을 테스트하라

: 코드 프로파일러 사용



성급한 최적화

소프트웨어 건축

리팩터링

- 중복, 직교적이지 않은 설계, 사용 사례, 성능, 테스트 통과, 더 이상 유효하지 않은 지식 등에 사용

일직 리팩터링하고, 자주 리팩터링하라.

테스트가 토드의 첫 번째 사용자다.

믿으라, 하지만 확인하라.

좋은 울타리가 좋은 이웃을 만든다.

기본 보안 원칙

- 공격 표면을 최소화하라. : 코드의 복잡성은 공격 매개체를 유발한다. 입력 데이터는 공격 매개체다. 인증없는 서비스는 공격 매개체다. 인증을 요구하는 서비스도 공격 매개체다. 출력 데이터는 공격 매개체다. 디버깅 정보는 공격 매개체다.
- 최소 권한 원칙
- 안전한 기본값
- 민감 정보를 암호화하라
- 보안 업데이트를 적용하라

컴퓨터 과학에는 어려운 문제가 딱 두 개 있다. 캐시무효화, 이름 짓기, 그리고 하나 차이 (off-by-one) 오류.

자신이 뭘 원하는지 정확히 아는 사람은 아무도 없다.



완성이라는 것은 더 이상 더할 것이 없을 때가 아니라, 더 이상 뺄 것이 없을 때 달성되는 것이다.

프로그래머는 사람들이 자신이 원하는 바를 깨다도록 돕는다.

요구 사항은 피드백을 반복하며 알게 된다.

사물자처럼 생각하기 위해 사용자와 함께 일하라.

관찰을 할 때, 행운은 준비된 사람에게 찾아온다.

구형 시스템 유지 보수

프로세스 회고와 개선

새로운 기술 탐험

학습 및 기술 갈고 닦기

실현하려면 계획하라

화물 송배

유행하는 것이 아니라 실제로 잘 맞는 것을 사용하라

생각 없이 행할 수 있는 중요한 작업의 수가 늘어남에 따라 문명은 발전한다.



길게 보면 우리가 우리의 삶을, 우리 자신을 조형한다. 이 과정은 우리가 죽을 때까지 끝나지 않는다. 우리가 내린 결정은 궁극적으로 우리 자신의 책임이다.

먼저, 해를 끼치지 말라.

쓰레기 같은 인간을 돕지 말라.

결국 당신의 삶이다. 삶을 사람들과 나누고, 삶을 축하하고, 삶을 만들어가라. 그리고 그걸 즐겨라.