CPython 파헤치기

CPython Internals

CPython Internals

CPython 파헤치기: 따라 하면서 이해하는 파이썬 내부의 동작 원리

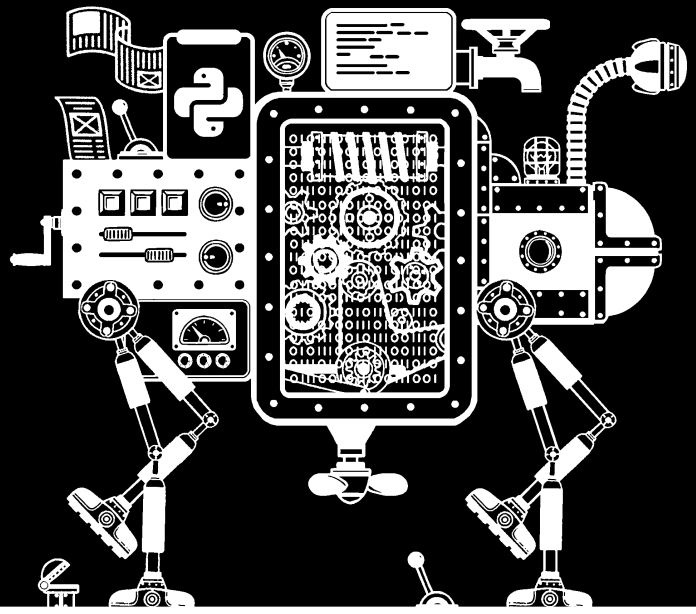
전자책 1쇄 발행 2022년 10월 14일 **지은이** 앤서니 쇼, realpython.com 튜토리얼 팀 **옮긴이** 김성현 **펴낸이** 한기성 **펴낸곳** (주)도서출판인사이트 **편집** 송우일, 정수진 **등록번호** 제2002-000049호 **등록일자** 2002년 2월 19일 **주소** 서울시 마포구 연남로5길 19-5 **전화** 02-322-5143 **팩스** 02-3143-5579 **블로그** https://blog.insightbook.co.kr **이메일** insight@insightbook.co.kr ISBN 978-89-6626-373-8

프로그래밍 인사이트

CPython 파헤치기

따라 하면서 이해하는 파이썬 내부의 동작 원리

앤서니 쇼, realpython.com 튜토리얼 팀 지음 | 김성현 옮김



인사이트

차례

추천의 글

프로그래밍 언어 구현체 개발은 자료 구조, 알고리즘, 운영 체제, 컴퓨터 구조 등의 지식이 총동원되는, 전산학의 꽃이라고 부르는 분야다. 화려한 명성만큼이나 많은 사람이 필요 이상으로 어렵게 생각하는 것도 사실이다. 그런 사람들에게 《CPython 파헤치기》는 사실상 파이썬 표준 구현체인 CPython 내부를 파헤쳐 보는 데 도움이 될 훌륭한 가이드북이다. 파이썬에 관심이 없더라도 프로그래밍 언어 구현이라는 주제에 관심이 있다면 그 자체만으로도 읽어 볼 만한 가치가 있는 책이기도 하다. 《CPython 파헤치기》를 읽으면서 CPython이 어떻게 설계, 개발되었는지 음미할 수 있으면 좋겠다. 옮긴이의 세세한 주석들이 독자들의 CPython 탐구 여행에 큰 도움이 될 것이다. 덧붙여 《CPython 파헤치기》 출간을 계기로 프로그래밍 언어 구현이라는 주제에 대해 뜨겁게 토론하는 사람들이 지금보다 더 많이 늘어났으면 하는 소망이 있다.

나동희, CPython 코어 개발자, LINE 소프트웨어 엔지니어

몇 년 전 파이썬 개발을 시작했을 때 이런 책이 있었더라면 좋았을 것 같다. 이 책을 읽고 나면 분명 실력이 향상될 뿐 아니라 세상을 개선하는 데 필요한 복잡한 문제들을 해결할 수 있을 것이다.

캐럴 윌링(Carol Willing), CPython 코어 개발자 겸 CPython 운영 위원회 멤버

이 책에서 가장 인상 깊었던 점은 CPython 코드 베이스를 변경하는 과정을 순서대로 따라가기 쉽게 안내한다는 점이다. 오라일리(O'Reilly) 출판사의 '미싱 매뉴얼(Missing Manuals)' 시리즈 책들 같다는 인상을 받았다.

C로 이루어진 파이썬의 기반에 대해 알아가는 건 매우 재미있었고 오래된 의문들도 해소되었다. 특히 CPython의 메모리 할당자 부분에서 많은 깨달음을 얻었다.

이 책은 파이썬에 대해 한층 높은 수준의 공부를 하려는 모든 사람들에게 훌륭한(그리고 독보적인) 참고 자료다.

댄 베이더(Dan Bader), 《슬기로운 파이썬 트릭》(인사이트, 2019) 지은이,
리얼 파이썬 편집장

추천사

공동체가 만들어 낸 언어가 전 세계 사용자에게 행복을 전합니다.

– 히도 판로섬(Guido van Rossum)[1], 네덜란드 국왕 탄생일 연설 중[2]

나는 지식과 아이디어를 다른 사람들과 공유할 수 있도록 학습을 돕고, 우리에게 창작할 역량을 주고, 우리를 움직일 수 있는 도구를 만드는 것을 좋아한다. 파이썬과 내가 만든 도구들이 기후 변화나 알츠하이머와 같은 현실 문제들을 해결하는 데 도움이 된다는 이야기를 들으면 감사함과 뿌듯함, 감동을 느낀다.

나는 40년 넘게 프로그래밍과 문제 해결을 사랑해 왔고 공부하며 많은 코드를 작성하고 아이디어를 공유하면서 시간을 보냈다. 긴 세월 동안 기술에 큰 변화가 일어나는 것을 보았는데 메인프레임에서 휴대 전화 서비스를 거쳐 웹과 경이로운 클라우드 컴퓨팅으로 발전했다. 파이썬을 비롯한 이 모든 기술에는 한 가지 공통점이 있다.

아무리 성공적인 혁신이라도 한때는 아이디어에 지나지 않았다. 히도 같은 창조자들은 앞으로 나아가기 위해 위험을 감수하고 새로운 땅에 발을 디뎠다. 헌신, 시행착오를 통한 학습 그리고 실패를 극복해 나가는 협력이 성공과 성장을 위한 단단한 기반을 구축했다.

이 책은 성공한 프로그래밍 언어인 파이썬을 탐색하는 여정으로 안내한다. 이 책은 파이썬이 내부적으로 어떻게 작동하는지에 대한 가이드 역할을 한다. 핵심 개발자들이 어떻게 언어를 만들었는지도 엿볼 수 있을 것이다.

파이썬의 강점은 언어의 가독성과 교육에 헌신하는 열린 커뮤니티다. 지은이도 이러한 장점을 알고 있기 때문에 CPython을 설명할 때 출처를 언급하고 언어의 구성 요소들을 여러분과 공유한다.

내가 이 책을 공유하고 싶은 이유는 무엇일까? 몇 년 전 파이썬 개발을 시작

1 (옮긴이) 국내에서는 주로 '귀도 반 로섬'으로 표기하지만 이 책에서는 네덜란드어의 실제 발음에 맞게 '히도 판로섬'으로 표기한다.

2 *http://neopythonic.blogspot.com/2016/04/*

했을 때 이런 책이 있었더라면 좋았을 것 같기도 하고, 더 중요하게는 우리가 파이썬 커뮤니티의 구성원으로서 직면하고 있는 현실의 복잡한 문제들을 해결하기 위해 전문적인 지식을 사용할 수 있는 특별한 기회가 있다는 점 때문이기도 하다.

이 책을 읽고 나면 분명 실력이 향상될 뿐 아니라 세상을 개선하는 데 필요한 복잡한 문제들을 해결하는 데 도움이 될 것이다.

이 책이 파이썬에 대해 더 알아 가도록 동기를 부여하고, 혁신적인 것을 만들 수 있도록 영감을 주고, 만들어 낸 것들을 세상과 공유할 수 있는 자신감을 줄 수 있기 바란다.

> 지금 하는 게 아예 안 하는 것보다 낫다.
>
> – 팀 피터스(Tim Peters), '파이썬의 선(Zen of Python)' 지은이

팀의 격언을 따라 지금 바로 CPython 내부로의 여정을 시작해 보자.

여러분을 환영한다.

캐럴 윌링, CPython 코어 개발자[3] 겸 CPython 운영 위원회 멤버

3 (옮긴이) CPython 코어 개발자는 CPython 소스 코드에 대한 풀 리퀘스트 병합을 할 수 있으며 파이썬 개발과 관련된 중요한 결정에 대해 투표 권한을 가진다.

옮긴이의 글

파이썬의 인기가 날로 높아지고 있다. 교육 분야뿐 아니라 기계 학습, 데이터 엔지니어링, 백엔드 개발, 심지어는 사물 인터넷(IoT) 기기에서도 파이썬이 사용되고 있다. 이렇게 많은 분야에서 많은 사람에게 파이썬이 사랑받는 이유는 무엇일까? 파이썬은 다른 언어 대비 뛰어난 성능을 제공하지도 않고, 문법적 편의 기능이 풍부한 편도 아니다. 하지만 파이썬은 단순하고 직관적이다. '파이썬의 선(Zen of Python)'을 따라 언어의 기본적인 문법부터 표준 라이브러리, 문서까지 모두 단순하고 직관적이다. 활발하고 개방적인 커뮤니티 문화도 파이썬이 사랑받는 이유 중 하나다.

단순하고 직관적인 설계, 개방적인 커뮤니티 문화는 파이썬 내부에도 어김없이 드러난다. 파이썬의 소스 코드는 잘 정리되어 있고 가독성이 뛰어나다. 개발 환경 설정과 런타임 자체를 설명한 문서도 풍부하다. 모든 수정과 스펙 제안은 커뮤니티를 통해 공개적으로 이루어진다.

이토록 친절한 파이썬이지만 그럼에도 파이썬은 혼자서 탐험하기에는 꽤 거대한 프로젝트다. 하지만 이 책이 파이썬의 내부라는 미지의 세계를 같이 탐험할 수 있는 길잡이가 되어 줄 것이다. C 언어를 알고 있거나 컴파일러 등에 대해 평소에 관심이 있었던 독자뿐 아니라 파이썬 내부에 대해 관심을 가지고 있었던 독자라면 누구든 이 책과 함께 파이썬 내부를 즐겁게 탐험할 수 있을 것이다.

개발 환경 설정부터 핵심 구성 요소들에 대한 안내, 소프트웨어 개발 실력을 한 단계 더 끌어올려 줄 테스트와 디버깅, 프로파일링 스킬에 이르기까지 지은이는 단순 해설이 아닌 누구나 쉽게 따라갈 수 있는 잘 짜인 구성을 통해 파이썬의 내부로 독자들을 안내한다.

파이썬을 사랑하는 사람 중 한 명으로서 국내에 처음으로 출간되는, 파이썬 내부를 다루는 책의 번역을 맡게 되어 매우 뜻깊게 생각한다. 이토록 매력적인 주제가 아니었다면 많은 시간을 할애하며 번역할 각오를 내지 못했을 것 같다.

소스 코드와 공식 문서를 최대한 참고하며 열심히 작업했지만 다양한 피드백과 도움이 있었기에 번역을 마칠 수 있었다.

역자로 추천해 주시고 베타 리딩에서 많은 피드백을 남겨 주신 김준기 님과 파이썬 코어 개발자로서 피드백뿐 아니라 다양한 질문에 답변을 주신 나동희 님에게 특별히 감사의 마음을 전한다. 또 이런 매력적인 주제를 다루는 책을 번역할 기회를 주신 인사이트와 꼼꼼하게 점검해 주신 편집 팀에도 감사의 마음을 전한다. 마지막으로 항상 곁에서 응원해 주시는 부모님에게도 감사를 전한다.

들어가는 글

빠르게 값을 찾을 수 있는 딕셔너리, 변수의 상태를 기억하며 값을 생성하는 제너레이터, 자동으로 관리되는 메모리 등 파이썬의 놀라운 기능들은 어떻게 구현되었을까?

그 답은 가장 널리 사용되는 파이썬 런타임인 CPython 소스 코드에서 찾을 수 있다. CPython은 사람이 읽을 수 있는 C와 파이썬 코드로 작성되어 있다.

CPython은 C 플랫폼과 운영 체제의 복잡성을 추상화한다. 추상화를 통해 CPython은 호환성 있는 간편한 스레딩 기능을 제공하고, C에서는 어려웠던 메모리 관리도 간단한 형태로 제공한다.

CPython은 개발자가 고성능의 확장 가능한 애플리케이션을 작성할 수 있는 플랫폼을 제공한다. 뛰어난 파이썬 개발자가 되려면 CPython 작동 방식을 이해해야 한다. 추상화는 완벽하지 않고 구멍이 있다.

CPython의 작동 방식을 알고 있으면 CPython을 최대한 활용하면서 애플리케이션을 최적화할 수 있다. 이 책은 CPython의 주요 개념과 다음 세부 내용을 설명한다.

- 소스 코드를 읽고 탐색하기
- CPython 소스 코드 컴파일하기
- 파이썬 문법을 수정하고 컴파일해서 자신만의 CPython 버전 만들기
- 리스트나 딕셔너리, 제너레이터 등이 내부에서 작동하는 방식 이해하기
- CPython의 메모리 관리 기능 이해하기
- 병렬성과 동시성을 통해 파이썬 코드 확장하기
- 코어 타입에 새로운 기능 추가하기
- 테스트 스위트 실행하기
- 파이썬 코드와 런타임 성능을 프로파일하고 벤치마크하기
- C 코드와 파이썬 코드를 전문가처럼 디버깅하기

- CPython 라이브러리의 구성 요소를 수정하거나 개선해 향후 버전의 CPython에 기여하기

시간을 들여 각 장의 데모와 대화형 요소를 체험해 보자. 더 나은 파이썬 프로그래머가 되기 위해 알아야 할 핵심 개념을 이해하면서 성취감을 느낄 수 있을 것이다.

이 책을 보는 방법

이 책은 실습을 통한 학습을 권장한다. 설명을 읽고 통합 개발 환경(integrated development environment, IDE)부터 구성한 후 코드를 다운로드해 예제를 작성해 보자.

코드 예제를 복사해서 붙여 넣는 것은 피하자. 예제 중에는 올바르게 실행될 때까지 여러 번 반복해서 작성해야 하는 예제도 있고, 의도적으로 버그를 포함한 예제도 있다.

실수했을 때 고치는 방법을 배우는 것도 학습 과정의 일부다. 더 나은 구현 방법을 발견했다면 바꿔 보고 어떤 효과가 있는지도 확인해 보자.

CPython의 내부를 완전히 이해하려면 충분한 연습이 필요하다. 연습을 즐겨 보자.

파이썬을 어느 정도 알아야 할까?

이 책의 대상 독자는 파이썬을 능숙하게 다루는 파이썬 개발자다. 코드 예시가 있지만 파이썬을 능숙하게 다룰 수 있을 정도의 테크닉이 전체적으로 필요하다.

C를 알아야 할까?

C에는 능숙하지 않아도 된다. C를 처음 접하는 경우에는 부록 '파이썬 프로그래머를 위한 C 안내서'를 참고하자.

다 읽기까지 얼마나 걸릴까?

급하게 읽기보다는 각 장을 시간을 들여 읽고, 장 뒤에 나오는 예제들을 따라 해 보면서 코드를 살펴보자. 다 읽고 난 후에도 이 책은 언제든 살펴볼 만한 훌륭한 레퍼런스 가이드가 될 것이다.

금세 뒤떨어진 내용이 되지 않을까?

파이썬은 서른 살이 넘었지만 CPython 코드 중에는 처음 작성된 이후로 변경되지 않은 코드도 있다. 또한 이 책에서 소개하는 많은 원리는 10년 이상 바뀌지 않았다.

이 책을 쓰면서 파이썬 창시자 히도 판로섬이 작성한 첫 버전 그대로 남아 있는 코드도 많이 발견했다.

물론 이 책에서 소개하는 개념 중 일부는 완전히 새롭거나 실험적이기도 하다. 이 책을 쓰면서 발견한 버그와 소스 코드의 문제들은 나중에 수정되거나 개선되기도 했다.[1] 이런 점이 바로 오픈 소스 프로젝트인 CPython의 멋진 부분이다.

이 책은 CPython의 현재 버전은 물론 향후 버전을 이해하는 데도 도움이 될 것이다. 변화는 지속적이고 전문성은 그 과정에서 키워 나갈 수 있다.

추가 학습 자료

*realpython.com/cpython-internals/resources/*에서 이 책을 위한 여러 추가 자료를 무료로 제공한다. 리얼 파이썬 팀이 관리하는 정오표도 해당 페이지에서 찾을 수 있다.

코드 샘플

예제와 샘플은 다음과 같이 `cpython-books-samples` 폴더에서 시작하는 경로와 파일명을 표시하고 해당 코드를 보여 준다.

1 *https://realpython.com/cpython-fixes*

cpython-book-samples▶01▶example.py

```
import this
```

*realpython.com/cpython-internals/resources/*에서 코드 샘플을 다운로드할 수 있다.

코드 라이선스

이 책의 파이썬 스크립트 예제는 크리에이티브 코먼스 퍼블릭 도메인(CC0) 라이선스[2]로 제공된다. 여러분의 프로그램에서 어떤 목적으로든 코드의 어떤 부분이라도 사용해도 괜찮다.

CPython은 파이썬 소프트웨어 재단(Python Software Foundation, PSF) 2.0 라이선스[3]로 제공된다. 이 책에서 사용된 CPython 토막 코드와 샘플은 PSF 2.0 라이선스를 따른다.

> ☑ 이 책의 코드는 윈도우 10과 macOS 10.14, 리눅스에서 파이썬 3.9를 사용하여 테스트했다.

서식 규칙

코드 블록은 예제 코드를 표시하는 데 사용된다.

```
# 다음은 파이썬 코드다.
print("Hello, World!")
```

운영 체제와 무관한 명령은 유닉스 스타일을 사용한다($ 기호는 명령의 일부가 아니다).

```
$ # 다음은 터미널 명령이다.
$ python hello-world.py
```

윈도우에만 해당하는 명령은 윈도우 명령줄 형식을 사용한다(> 기호는 명령의 일부가 아니다).

2 *https://creativecommons.org/publicdomain/zero/1.0/*
3 *https://github.com/python/cpython/blob/main/LICENSE*

```
> python hello-world.py
```

명령줄은 다음 형식을 사용한다.

- 고정폭 글꼴로 쓴 **괄호 없는 텍스트**는 표시된 대로 입력해야 한다.
- **<홑화살괄호 안의 텍스트>**는 값을 제공해야 하는 변수를 나타낸다. 예를 들어 `<filename>`은 특정 파일 이름으로 대체해야 한다.
- **[대괄호 안의 텍스트]**는 선택적 인자를 나타낸다.

강조된 텍스트는 새롭거나 중요한 용어를 뜻한다.

참고와 주의는 다음과 같이 표시한다.

참고는 이렇게 표시한다.

주의는 이렇게 표시한다.

CPython 소스 코드에 들어 있는 파일에 대한 모든 참조는 다음과 같이 표시한다.

```
path▶to▶file.py
```

바로 가기 또는 메뉴 명령은 다음과 같이 표시한다.

'File ⇨ Other ⇨ Option'

피드백과 정오표

이해가 되지 않는 부분을 발견하거나 코드나 본문에서 오류를 찾거나 내용이 부족하다고 느낄 수도 있을 것이다. 아이디어, 제안, 피드백 등은 언제나 환영이다. 리얼 파이썬 팀은 교육 자료를 개선하기 위해 항상 노력하고 있다. 이유가 무엇이든지 피드백이 있다면 다음 링크로 피드백을 보내 주기 바란다.

https://realpython.com/cpython-internals/feedback

1장

C P y t h o n I n t e r n a l s

CPython 소스 코드 받기

콘솔에서 `python`이라고 입력해 실행하는 파이썬이나 *python.org*에서 받아서 설치하는 파이썬은 CPython이다. CPython은 파이썬 구현 중 하나로 다양한 개발자가 개발하고 있다. 다른 유명한 파이썬 배포판으로는 파이파이(PyPy), 사이썬(Cython), 자이썬(Jython) 등이 있다.

CPython의 특별한 점은 다른 구현체들과도 공유하는 언어 사양과 런타임을 같이 포함한다는 것이다. CPython은 파이썬의 공식 구현체이자 레퍼런스 구현체다.

파이썬 언어 사양은 파이썬 언어를 설명하는 문서로 `assert`는 예약어이고 `[]`는 인덱싱과 슬라이싱, 빈 리스트 생성을 위해 사용된다는 것 등을 정의한다.

다음은 CPython 배포판에서 제공하는 기능 중 일부다. 배포판은 컴파일러뿐 아니라 다양한 것을 포함한다.

- `python`을 아무런 인자 없이 실행해서 대화형 프롬프트(REPL) 열기
- 표준 라이브러리에서 `json`, `csv`, `collections` 등의 내장 모듈 임포트하기
- `pip`를 사용하여 인터넷에서 패키지를 받아서 설치하기
- 내장된 `unittest` 모듈을 사용하여 애플리케이션 테스트하기

앞으로 CPython 배포판의 구성 요소들에 대해 알아볼 것이다.

- 언어 사양
- 컴파일러
- 표준 라이브러리 모듈
- 코어 타입
- 테스트 스위트

1.1 소스 코드에 포함된 것들

CPython 소스 배포판은 다양한 종류의 도구와 라이브러리, 구성 요소를 포함한다.

이 책은 CPython 3.9[1] 소스 코드를 다룬다.

git으로 최신 CPython 소스 코드를 다운로드하자.

```
$ git clone --branch 3.9 https://github.com/python/cpython
$ cd cpython
```

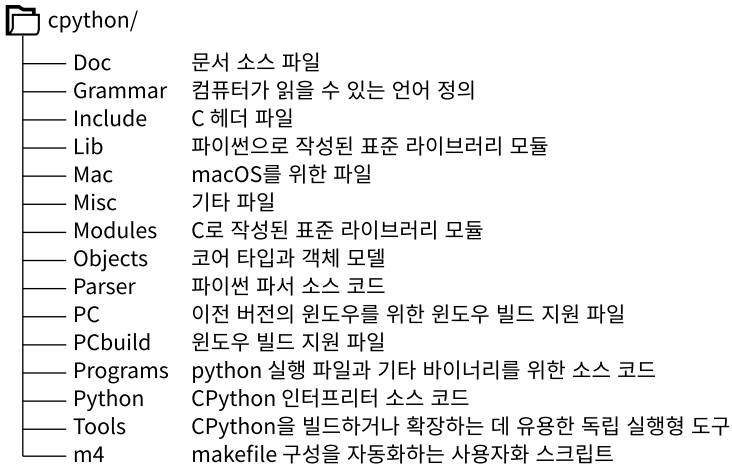
이 책의 모든 예제는 파이썬 3.9 기반이다.

3.9 브랜치를 선택하는 게 중요하다. 메인(main) 브랜치는 자주 변경된다. 이 책의 예제와 연습 문제 대부분은 메인 브랜치에서 작동하지 않을 것이다.

컴퓨터에 깃이 설치되어 있지 않다면 git-scm.com에서 깃을 받아서 설치하거나 깃허브에서 CPython 소스 코드 전체를 ZIP 파일[2]로 다운로드할 수 있다.
ZIP 파일로 소스 코드를 다운로드하면 히스토리, 태그, 브랜치 등은 포함되지 않는다.

다운로드한 cpython 디렉터리에서 다음과 같은 하위 디렉터리들을 확인할 수 있다.

1 *https://github.com/python/cpython/tree/3.9*
2 *https://github.com/python/cpython/archive/3.9.zip*



다음 장에서 개발 환경을 설정하자.

2장

C P y t h o n I n t e r n a l s

개발 환경 구성하기

앞으로 C 코드와 파이썬 코드를 모두 다뤄야 한다. 두 언어를 모두 지원하는 개발 환경을 구성하자.

CPython 소스 코드의 65%는 파이썬(테스트가 상당 부분을 차지한다), 24%는 C로 이루어져 있다. 나머지는 다른 언어가 섞여 있다.

2.1 편집기와 통합 개발 환경

사용할 개발 환경을 아직 정하지 않았다면 통합 개발 환경 또는 코드 편집기 중 하나를 골라야 한다.

- 통합 개발 환경은 특정 언어와 툴체인을 위한 도구다. 대부분의 통합 개발 환경은 통합된 테스트, 문법 검사, 버전 관리, 컴파일 기능을 포함한다.
- 코드 편집기는 언어에 상관없이 코드 파일을 수정할 수 있는 도구다. 대부분의 코드 편집기는 문법 강조 기능이 포함된 간단한 텍스트 편집기다.

완전한 기능을 갖춘 통합 개발 환경은 더 많은 하드웨어 자원을 사용한다. 램이 제한된 경우(8GB 미만) 코드 편집기를 권장한다.

통합 개발 환경은 시작 시간이 길다. 파일을 빠르게 수정하고 싶다면 코드 편집기가 좋은 선택이다.

많은 통합 개발 환경과 편집기가 있는데 무료인 것도 유료인 것도 있다. 다음은 CPython 개발에 적합한 일반적인 통합 개발 환경과 편집기 몇 가지다.

이름	구분	지원 환경
마이크로소프트 비주얼 스튜디오 코드	편집기	윈도우, macOS, 리눅스
아톰	편집기	윈도우, macOS, 리눅스
서브라임 텍스트	편집기	윈도우, macOS, 리눅스
Vim	편집기	윈도우, macOS, 리눅스
이맥스	편집기	윈도우, macOS, 리눅스
마이크로소프트 비주얼 스튜디오	C, 파이썬용 통합 개발 환경	윈도우
젯브레인스 파이참	파이썬용 통합 개발 환경	윈도우, macOS, 리눅스
젯브레인스 CLion	C용 통합 개발 환경	윈도우, macOS, 리눅스

맥용 비주얼 스튜디오가 있지만 맥용은 C 컴파일과 비주얼 스튜디오 파이썬 도구를 지원하지 않는다.

이어지는 절에서는 다음 통합 개발 환경과 편집기를 설정하는 법을 소개한다.

- 마이크로소프트 비주얼 스튜디오
- 마이크로소프트 비주얼 스튜디오 코드
- 젯브레인스 CLion
- Vim

선택한 도구에 대한 절로 건너뛰거나 전부 읽고 비교해 보자.

2.2 비주얼 스튜디오 구성하기

이 책을 쓰는 현재 비주얼 스튜디오 최신 버전인 비주얼 스튜디오 2019는 파이썬과 C 소스 코드를 기본적으로 지원한다. 비주얼 스튜디오 2017이 기존에 설치되어 있다면 그대로 사용할 수 있다. 이 책에 나온 예시와 연습 문제에 비주얼 스튜디오 사용을 권장한다.

☑ 이 책을 공부하거나 CPython을 컴파일하는 데 비주얼 스튜디오의 유료 기능이 필요하지는 않다. 무료 커뮤니티 버전으로도 충분하다.
하지만 프로파일 기반 최적화(profile-guided optimization, PGO) 빌드를 사용하려면 프로페셔널 에디션 또는 그 이상이 필요하다.

비주얼 스튜디오는 마이크로소프트 비주얼 스튜디오 웹 사이트에서 무료로 받을 수 있다.[1]

비주얼 스튜디오 인스톨러를 다운로드해 실행하면 설치할 구성 요소를 선택하라는 메시지가 표시된다. 이 책에는 다음 구성 요소가 필요하다.

- **파이썬 개발** 워크로드
- **파이썬 네이티브 개발 도구**
- 파이썬 3 64비트(3.7.8)

파이썬 3.7이 이미 설치되어 있다면 파이썬 3 옵션은 설치하지 않아도 된다. 디스크 공간을 절약하기 위해 다른 옵션을 추가로 제외할 수도 있다.

구성 요소를 선택하면 인스톨러가 모든 구성 요소를 다운로드해 설치한다. 설치에는 최대 1시간까지 소요될 수 있다.

설치를 완료하고 '시작'을 클릭해 비주얼 스튜디오를 시작하면 회원 가입 창이 표시되는데 마이크로소프트 계정이 있다면 로그인하거나 이 단계를 건너뛸 수 있다.

다음으로 초기 화면이 표시되면 '리포지토리 복제' 옵션을 선택해 비주얼 스튜디오에서 CPython 깃 저장소를 직접 클론할 수 있다.

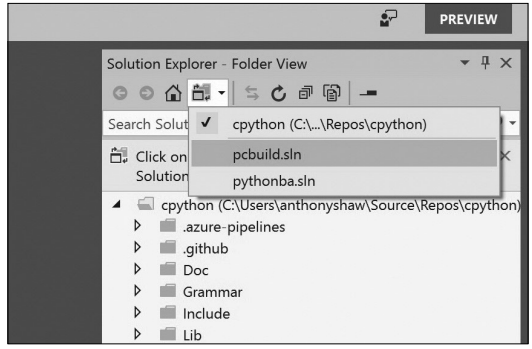
리포지토리 위치로 `https://github.com/python/cpython`을 입력한 후 저장할 경로를 선택하고 '복제'를 선택하자.

비주얼 스튜디오에 내장된 깃으로 깃허브에서 CPython 사본을 다운로드한다. 다운로드에는 최대 10분 정도가 걸린다.

1 *https://visualstudio.microsoft.com/vs/*

> 비주얼 스튜디오는 기본으로 메인 브랜치를 체크아웃한다. 컴파일 전에 팀 탐색기로 3.9 브랜치를 선택하자. 3.9 브랜치를 선택하는 게 중요하다. 메인 브랜치는 자주 변경된다. 이 책의 예제와 연습 문제 대부분은 메인 브랜치에서는 작동하지 않을 것이다.

프로젝트가 다운로드되면 '프로젝트 및 솔루션⇨pcbuild.sln'을 클릭하여 `PCBuild▶pcbuild.sln` 솔루션 파일을 사용하자.



이제 비주얼 스튜디오와 소스 코드가 준비되었다. 윈도우에서 CPython을 컴파일하는 방법은 다음 장에서 알아본다.

2.3 비주얼 스튜디오 코드 구성하기

마이크로소프트 비주얼 스튜디오 코드(이하 VS 코드)는 온라인 플러그인 마켓플레이스를 제공하는 확장 가능한 코드 편집기다.

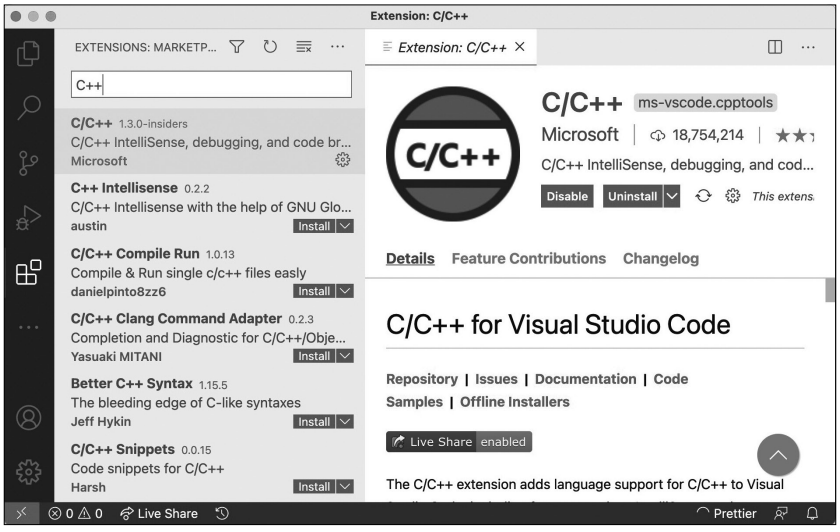
통합 깃 인터페이스를 제공하며 C와 파이썬 모두를 지원하기 때문에 CPython을 위한 좋은 선택이다.

2.3.1 설치

VS 코드는 *code.visualstudio.com*에서 다운로드 받을 수 있는 간단한 인스톨러를 사용해 설치할 수 있다.

VS 코드는 기본 상태에서도 코드 편집이 가능하지만 확장을 설치해 유용한 기능을 추가할 수 있다.

상단 메뉴의 'View⇨Extensions'를 클릭하면 Extensions 패널에 접근할 수 있다.



Extensions 패널에서 이름 또는 `ms-vscode.cpptools` 같은 고유 식별자로 확장을 검색할 수 있다. 비슷한 이름의 플러그인이 있을 경우에는 고유 식별자로 검색해 올바른 플러그인을 설치했는지 확인하자.

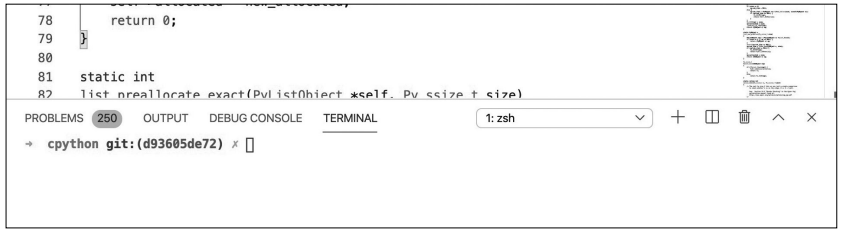
2.3.2 권장되는 확장

CPython 작업에 유용한 확장은 다음과 같다.

- C/C++(`ms-vscode.cpptools`): 인텔리센스, 디버깅, 코드 강조 등의 C/C++ 지원을 제공한다.
- Python(`ms-python.python`): 파이썬 코드 편집, 디버깅, 탐색 등의 파이썬 지원을 제공한다.
- reStructuredText(`lextudio.restructuredtext`): CPython 문서에 사용되는 reStructuredText에 대한 지원을 제공한다.
- Task Explorer(`spmeesseman.vscode-taskexplorer`): `make` 작업을 편리하게 실행할 수 있는 Task Explorer 패널을 Explorer 탭 안에 추가한다.

필요한 확장을 설치한 후 편집기를 새로 고침하자.[2]

앞으로 명령줄이 자주 필요하기 때문에 'Terminal⇨New Terminal'을 클릭해 VS 코드에 통합 터미널을 추가한다. 터미널은 다음과 같이 나타난다.



2.3.3 고급 코드 탐색 및 펼치기 사용

플러그인을 설치해 고급 코드 탐색을 실행할 수 있다.

예를 들어 C 파일에서 함수 호출을 우클릭하고 'Go to References'를 선택하면 해당 함수에 대한 다른 참조들이 보인다.

'Go to References'를 사용해 해당 함수를 사용하는 적절한 방식을 찾을 수도 있다.

C 매크로(macro)를 클릭하거나 그 위에 마우스를 놓으면 매크로가 컴파일된 형태로 펼쳐진다.

2 (옮긴이) VS 코드는 일렉트론(Electron)을 기반으로 만들어졌으며 웹 사이트를 새로 고침하는 것과 동일한 메커니즘을 사용한다.