

혼자 공부하며 함께 만드는

---

혼공 용어 노트

# 목차

## 가나다순

가비지 컬렉션 garbage collection	33	도메인 domain	23	문자열 셋 character set	29	생성 generation	16
강화 학습 reinforcement learning	15	돔 DOM	23	바이트 byte	28	서버 server	17
개발 development	12	동적 타입 dynamically typed	30	반응형 웹사이트 responsive website	19	서버 호스팅 server hosting	17
객체 object	30	디버그 debug	14	배포 deploy	14	세션 session	20
객체지향 프로그래밍 OOP	30	디스코드 discord	16	백엔드 backend	23	소스 코드 source code	12
고수준 언어 high-level language	13	라이브러리 library	34	변수 variable	33	소프트웨어 software	12
구문 오류 syntax error	32	램 RAM	27	병렬 처리 parallel processing	27	스레드 thread	26
규칙 기반 인공지능		런타임 오류 runtime error	32	병행 처리 concurrent processing	28	시스템 프로그램 system program	12
rule-based artificial intelligence	15	런타임 환경 runtime environment	33	보조 기억 장치 secondary memory	27	애플리케이션 application	13
기계어 machine language	13	리버트 revert	34	불리언 boolean	28	어셈블리어 assembly language	13
깃허브 코파일럿 github copilot	16	리셋 reset	34	브랜치 branch	34	언매니지드 언어 unmanaged language	31
깃 호스팅 서비스 git hosting service	34	리팩토링 refactoring	14	비지도 학습 unsupervised learning	15	예외 처리 exception handling	32
		매니지드 언어 managed language	31	비트 bit	28	오픈 소스 open source	25
네이티브 앱 native app	20	메모리 memory	27	비휘발성 메모리 non-volatile memory	27	온프레미스 on-premise	17
논리 오류 logical error	32	메모리 누수 memory leak	33	빅데이터 big data	15	운영체제 OS	25
		메소드 method	31	빌드 build	14	웹 뷰 web view	20
데이터 data	24	명령문 command statement	12			웹 브라우저 web browser	18
데이터 센터 DC/IDC	17	문서화 documentation	14	상속 inheritance	31	웹 서핑 web surfing	19
데이터베이스 database	24	문자 인코딩 character encoding	29	상수 constant	31	웹 앱 web app	20
데이터베이스 관리 시스템 DBMS	25						

웹사이트 web site	19	주 기억 장치 main memory	26
유니코드 unicode	29	주석 comment	12
유지 보수 maintenance	14	지도 학습 supervised learning	15
이해 understanding	16		
인공지능 AI	11	캐시 cache	20
인공 신경망 artificial neural network	15	커널 kernel	25
인터페이스 interface	25	커밋 commit	34
인터프리터 언어 interpreted language	29	컨텍스트 스위칭 context switching	28
입출력 장치 I/O	27	컴파일 compile	29
		컴파일 언어 compiled language	29
자료형 data type	29	컴파일 오류 compilation error	32
자바 가상 머신 JVM	32	컴파일러 compiler	29
자바 런타임 환경 JRE	33	코드 리뷰 code review	14
자바스크립트 javascript	19	코딩 coding	12
자연어 처리 natural language processing	16	코어 core	26
저수준 언어 low-level language	13	쿠키 cookie	19
적응형 웹사이트 adaptive website	19	크래커 cracker	11
절차지향 프로그래밍		크로스 플랫폼 cross platform	14
procedural programming	30	클라우드 컴퓨팅 서비스	18
정적 타입 statically typed	30	클라이언트 client	17

클래스 class	30	함수 function	31
클럭 clock	26	함수형 프로그래밍 functional programming	31
		해커 security hacker	11
타입스크립트 typescript	21	환경 변수 environment variable	33
토큰 token	20	휘발성 메모리 volatile memory	27
통합 개발 환경 IDE	13		
트랜스포머 transformer	16		
패키지 매니저 package manager	33		
프레임워크 framework	22		
프로그래밍 programming	12		
프로그래밍 언어 programming language	13		
프로그램 program	13		
프로세스 process	27		
프론트엔드 frontend	23		
프롬프트 prompt	16		
프롬프트 엔지니어링 prompt engineering	16		
하이브리드 앱 hybrid app	20		
하이퍼텍스트 hypertext	24		

ABC 순

adaptive website 적응형 웹사이트	19	child class 자식 클래스	20	cookie 쿠키	20	DOM Document Object Model 돔	23
AI Artificial Intelligence 인공지능	11	class 클래스	30	core 코어	26	domain 도메인	23
AJAX Asynchronous Javascript And XML	23	CLI Command Line Interface	25	CPU Central Processing Unit	26	dynamically typed 동적 타입	30
API Application Programming Interface	22	client 클라이언트	17	cracker 크래커	11	environment variable 환경 변수	33
application 애플리케이션	13	clock 클럭	26	cross platform 크로스 플랫폼	14	exception handling 예외 처리	32
artificial neural network 인공 신경망	15	cloud computing service		CSS Cascading Style Sheets	18	framework 프레임워크	22
assembly language 어셈블리어	13	클라우드 컴퓨팅 서비스	18	data 데이터	24	frontend 프론트엔드	21
		code review 코드 리뷰	14	data type 자료형	29	function 함수	31
backend 백엔드	21	coding 코딩	12	database 데이터베이스	25	functional programming 함수형 프로그래밍	31
big data 빅데이터	15	command statement 명령문	12	DBMS DataBase Management System		garbage collection 가비지 컬렉션	33
bit 비트	28	comment 주석	12	데이터베이스 관리 시스템	24	generation 생성	16
boolean 불리언	28	commit 커밋	34	DC Data Center / IDC Internet Data Center		github copilot 깃 허브 코파일럿	16
branch 브랜치	34	commit message 커밋 메시지	34	데이터 센터	17	git hosting service 깃 호스팅 서비스	34
build 빌드	12	compilation error 컴파일 오류	32	debug 디버그	14	GUI Graphical User Interface	25
byte 바이트	28	compile 컴파일	29	deploy 배포	14	high-level language 고수준 언어	13
		compiled language 컴파일 언어	29	development 개발	12	HTML HyperText Markup Language	18
cache 캐시	20	compiler 컴파일러	29	discord 디스코드	16		
CDN Content Delivery Network	20	concurrent processing 병행 처리	28	DNS Domain Name Server	23		
character encoding 문자 인코딩	29	constant 상수	31	documentation 문서화	14		
character set 문자열 셋	29	context switching 컨텍스트 스위칭	28				

HTTP HyperText Transfer Protocol Secure	24	JSON JavaScript Object Notation	22
HTTPS HyperText Transfer Protocol over Secure socket layer	24	JVM Java Virtual Machine 자바 가상 머신	32
hybrid app 하이브리드 앱	20	kernel 커널	25
hypertext 하이퍼텍스트	24	library 라이브러리	34
iaaS Infrastructure as a Service 아이아스	18	logical error 논리 오류	32
IDE Integrated Development Environment		low-level language 저수준 언어	13
통합 개발 환경	13	machine language 기계어	13
inheritance 상속	31	main memory 주 기억 장치	26
I/O 입출력 장치	27	maintenance 유지 보수	14
interface 인터페이스	25	managed language 매니지드 언어	31
interpreted language 인터프리터 언어	29	memory 메모리	27
IP Internet Protocol 인터넷 프로토콜	22	memory leak 메모리 누수	33
IP Address IP 주소	23	method 메소드	31
javascript 자바스크립트	19	native app 네이티브 앱	20
JDK Java Development Kit	34	natural language processing 자연어 처리	16
JRE Java runtime environment		non-volatile memory 비휘발성 메모리	27
자바 런타임 환경	33		

object 객체	30	RAM Random Access Memory 램	27
on-premise 온프레미스	17	refactoring 리팩토링	14
OOP Object Oriented Programming		reinforcement learning 강화 학습	15
객체지향 프로그래밍	30	reset 리셋	34
open source 오픈 소스	25	responsive website 반응형 웹사이트	19
OS Operating System 운영체제	25	revert 리버트	34
PaaS Platform as a Service 파스	18	runtime environment 런타임 환경	33
package manager 패키지 매니저	33	runtime error 런타임 오류	32
parallel processing 병렬 처리	27	rule-based artificial intelligence 규칙 기반 인공지능	15
parent class 부모 클래스	33	SaaS Software as a Service 싸스	18
procedural programming 절차지향 프로그래밍	30	SDK Software Development Kit	33
process 프로세스	27	secondary memory 보조 기억 장치	27
program 프로그램	13	security hacker 해커	14
programming 프로그래밍	12	server 서버	17
programming language 프로그래밍 언어	13	server hosting 서버 호스팅	17
progressive web apps 프로그레시브 웹 앱	21	session 세션	20
prompt 프롬프트	16	software 소프트웨어	12
prompt engineering 프롬프트 엔지니어링	16	source code 소스 코드	12

statically typed 정적 타입	30	web app 웹 앱	20
supervised learning 지도 학습	15	web browser 웹 브라우저	18
syntax error 구문 오류	32	web site 웹사이트	19
system program 시스템 프로그램	12	web surfing 웹 서핑	19
		web view 웹 뷰	21
thread 스레드	26	WWW World Wide Web	24
token 토큰	20		
transformer 트랜스포머	16	XML eXtensible Markup Language	22
typescript 타입스크립트	21	YAML YAML Ain't Markup Language	22
understanding 이해	16		
unicode 유니코드	29		
unmanaged language 언매니지드 언어	31		
unsupervised learning 비지도 학습	15		
URL Uniform Resource Locator	24		
variable 변수	31		
VCS Version Control System	34		
volatile memory 휘발성 메모리	27		

# 01 장

□ 크래커 **cracker** [01장 039쪽]

크래커를 먹을 때 잘게 잘게 부서진다는 추상적인 이미지에서 유래되어 컴퓨터 시스템을 뚫고 파괴하는 행위를 하는 사람. 블랙 해커라고도 한다.



블랙 해커가 시스템을 조각내는 모양이 크래커 과자가 부서지는 것과 비슷해서 크래커라고 부르기도 한다.

□ 해커 **security hacker** [01장 039쪽]

컴퓨터와 프로그래밍에 대한 전문 지식을 가진 사람을 뜻하는 단어. 화이트 해커라고도 한다.

□ 인공지능 **AI; Artificial Intelligence** [01장 041쪽]

인간의 사고와 학습 같은 지적 능력을 컴퓨터로 구현하는 기술. 인공지능에는 머신러닝과 딥러닝이 있다.

□ 명령문	<b>command statemen</b> [01장 049쪽]	컴퓨터에 각종 지시를 내리는 문구.
□ 소스 코드	<b>source code</b> [01장 049쪽]	개발자가 작업하는 작업물로, 주로 실행 프로그램을 만드는 과정을 입력하는 데 이용하는 텍스트 형태의 파일.
□ 주석	<b>comment</b> [01장 049쪽]	각 코드가 어떤 내용인지 개발자가 쉽게 알아볼 수 있도록 설명하는 글.
□ 코딩	<b>coding</b> [01장 049쪽]	프로그래밍 언어로 된 코드를 입력하는 작업 자체.
□ 프로그래밍	<b>programming</b> [01장 050쪽]	컴퓨터가 할 일의 절차와 알고리즘을 설계하는 것.
□ 개발	<b>development</b> [01장 048쪽]	소프트웨어를 설계, 구현, 운영, 관리하는 데 필요한 전반적인 기술 과정.
□ 소프트웨어	<b>software</b> [01장 050쪽]	프로그램과 라이브러리, 데이터 등으로 이뤄진 시스템.
□ 시스템 프로그램	<b>system program</b> [01장 051쪽]	운영체제의 일부로서 컴퓨터 이용 환경을 조성하는 소프트웨어.

□ 애플리케이션	<b>application</b> [01장 053쪽]	PC나 모바일 기기에서 사용자가 직접 사용하는 모든 프로그램.
□ 프로그램	<b>program</b> [01장 050쪽]	사용자의 명령에 따라 그 목적에 맞는 작업을 수행하는 일련의 명령어 모음.
□ 프로그래밍 언어	<b>programming language</b> [01장 059쪽]	컴퓨터와 소통하기 위해 만들어진 언어.
□ 고수준 언어	<b>high-level language</b> [01장 063쪽]	사람의 언어에 가까운 언어.
□ 기계어	<b>machine language</b> [01장 063쪽]	컴퓨터가 이해할 수 있는 숫자인 0과 1로만 구성된 언어.
□ 어셈블리어	<b>assembly language</b> [01장 063쪽]	기계어 바로 위 단계의 저수준 언어.
□ 저수준 언어	<b>low-level language</b> [01장 063쪽]	기계어에 가까운 언어.
□ 통합 개발 환경	<b>IDE; Integrated Development Environment</b> [01장 065쪽]	IDE는 개발에 관련된 다양한 기능들을 제공하는 개발용 프로그램.

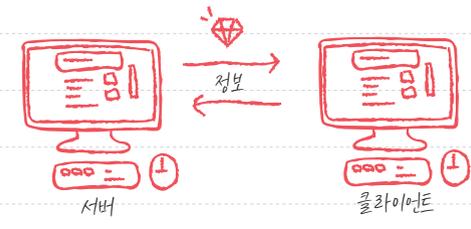
□ 디버그	debug	[01장 071쪽]
	오류 수정 프로그램과 그 작업 자체를 의미.	
	프로그램에서 문제를 찾아 제거하는 행동을 디버깅이라고 한다.	
□ 빌드	build	[01장 073쪽]
	프로그래밍한 소스 코드를 묶어 실행 가능한 파일을 만드는 것.	
□ 배포	deploy	[01장 076쪽]
	소프트웨어를 사용자들에게 전달하는 것.	
□ 유지 보수	maintenance	[01장 077쪽]
	소프트웨어 제품 출시 이후 계속되는 문제 해결 및 각종 업데이트 작업.	
	<i>유지 보수는 소프트웨어 개발에 중요!</i>	
□ 리팩토링	refactoring	[01장 079쪽]
	기능을 수정하지 않으면서 코드의 질을 높이는 것.	
□ 코드 리뷰	code review	[01장 079쪽]
	서로의 코드를 확인하고 피드백을 주고받는 과정.	
□ 문서화	documentation	[01장 081쪽]
	소프트웨어와 그 소스 코드를 쉽게 파악할 수 있는 문서.	

## 02장

□ 규칙 기반 인공지능	rule-based artificial intelligence	[02장 089쪽]
	미리 정해진 규칙이나 조건을 기반으로 작동하는 인공지능 시스템.	
□ 지도 학습	supervised learning	[02장 090쪽]
	컴퓨터에게 입력 데이터와 그에 해당하는 정답을 함께 제공하는 방식.	
□ 비지도 학습	unsupervised learning	[02장 091쪽]
	컴퓨터에게 레이블이 없는 데이터만 제공하고 스스로 제공된 데이터 내의 숨겨진 구조나 패턴을 찾아내는 방식.	
□ 강화 학습	reinforcement learning	[02장 091쪽]
	알고리즘이 특정 환경 내에서 시도와 오류를 통해 목표를 달성하는 방법을 학습.	
□ 빅데이터	big data	[02장 092쪽]
	방대한 양의 데이터.	
□ 인공 신경망	artificial neural network	[02장 094쪽]
	인간의 뇌를 모방한 컴퓨터 시스템.	

□ 이해	<b>understanding</b> [02장 102쪽]	컴퓨터가 인간의 언어를 인식하고 그 의미를 파악하는 과정.
□ 생성	<b>generation</b> [02장 102쪽]	컴퓨터가 인간이 이해할 수 있는 언어로 응답하거나 새로운 텍스트를 만들어내는 과정.
□ 자연어 처리	<b>natural language processing</b> [02장 101쪽]	컴퓨터가 인간의 언어를 이해하고 해석할 수 있도록 돕는 기술.
□ 트랜스포머	<b>transformer</b> [02장 104쪽]	데이터 처리를 비롯한 다양한 분야에서 사용되는 인공지능 모델.
□ 디스코드	<b>discord</b> [02장 117쪽]	텍스트와 음성 채팅뿐만 아니라 파일 공유도 가능한 채팅 메신저 프로그램.
□ 깃허브 코파일럿	<b>github copilot</b> [02장 122쪽]	깃허브에 게시된 수많은 코드 데이터를 학습하여 개발된 서비스.
□ 프롬프트	<b>prompt</b> [02장 124쪽]	사용자가 인공지능에게 말을 걸 때 쓰는 '시작 신호'.
□ 프롬프트 엔지니어링	<b>prompt engineering</b> [02장 124쪽]	인공지능이 역량을 발휘할 수 있도록 적절한 프롬프트를 작성하는 것.

# 03장

□ 서버	<b>server</b> [03장 130쪽]	정보나 서비스를 제공하는 컴퓨터.
		
□ 클라이언트	<b>client</b> [03장 131쪽]	서버가 제공하는 것을 받아 사용하는 컴퓨터.
□ 데이터 센터	<b>DC; Data Center / IDC; Internet Data Center</b> [03장 132쪽]	수많은 서버를 한 곳에서 안정적으로 관리하는 시설.
□ 서버 호스팅	<b>server hosting</b> [03장 133쪽]	서버용 컴퓨터를 임대해 주는 서비스.
□ 온프레미스	<b>on-premise</b> [03장 133쪽]	호스팅을 외부 데이터 센터에 두기에 민감한 정보를 다루는 회사들이 사내 전산실에 서버를 갖추고 관리하는 것.

□ 클라우드 컴퓨팅 서비스	cloud computing service 서버를 가상화하여 각종 편의 기능과 함께 필요한 만큼 사용할 수 있는 서비스.	[03장 134쪽]
□ IaaS	Infrastructure as a Service 클라우드를 이용해서 서버용 인프라, 즉 가상화된 서버 컴퓨터를 대여해 주는 서비스.	[03장 136쪽]
□ PaaS	Platform as a Service 애플리케이션을 개발하고 서비스하기 위해 필요한 서버, 운영체제, 개발 환경 등을 자동으로 설치하고 제공함으로써 사용자가 애플리케이션 개발에만 집중할 수 있도록 플랫폼을 제공하는 서비스.	[03장 137쪽]
□ SaaS	Software as a Service 소프트웨어가 이미 완성된 형태로 사용자에게 제공되는 클라우드 서비스.	[03장 138쪽]
□ 웹 브라우저	web browser 웹 서핑에 사용되는 소프트웨어.	[03장 143쪽]
□ HTML	HyperText Markup Language 웹 페이지에 요소들을 '가져다 놓는' 마크업 언어.	[03장 144쪽]
□ CSS	Cascading Style Sheets HTML로 올려놓은 요소들을 '꾸미는' 스타일 언어.	[03장 144쪽]

□ 자바스크립트	javascript 웹 페이지에 기능들을 부여해 '일을 시키는' 프로그래밍 언어.	[03장 144쪽]
□ 웹사이트	web site 단일 또는 다수의 웹 페이지로 구성.	[03장 145쪽]
□ 웹 서핑	web surfing 바다에서 파도를 타듯 이리저리 움직이며 웹 구석구석을 둘러본다는 뜻.	[03장 149쪽]
		
□ 반응형 웹사이트	responsive website 하나의 웹 페이지 화면 크기에 따라 내부 요소도 변경되는 웹사이트. 반응형 웹이라고도 한다.	[03장 153쪽]
□ 적응형 웹사이트	adaptive website 화면의 크기별로 웹 페이지를 따로따로 만든 웹사이트. 적응형 웹이라고도 한다.	[03장 154쪽]

□ 쿠키	<b>cookie</b> [03장 159쪽]	사용자의 브라우저에 저장되는 정보.
□ 세션	<b>session</b> [03장 160쪽]	서버가 사용자를 기억하고 있는 상태.
□ 토큰	<b>token</b> [03장 162쪽]	서버가 기억해 둘 필요 없이 사용자가 스스로를 증명할 수 있는 수단.
□ 캐시	<b>cache</b> [03장 164쪽]	다시 가져오지 않아도 되도록 데이터를 가까이 저장해 두는 기술. 메인 메모리 속 데이터에 보다 더 빠르게 접근할 수 있도록 CPU에 내장되는 형태.
□ CDN	<b>Content Delivery Network</b> [03장 165쪽]	각지에 캐시 서버를 두어 부하를 분산시키는 기술.
□ 네이티브 앱	<b>native app</b> [03장 171쪽]	동작할 운영체제에 특화된 방식으로 제작된 앱.
□ 크로스 플랫폼	<b>cross platform</b> [03장 173쪽]	하나의 소스 코드로 안드로이드와 iOS 양쪽에서 모두 동작하는 앱을 만들 수 있는 플랫폼.

□ 웹 앱	<b>web app</b> [03장 174쪽]	스마트폰의 브라우저에서 접속할 수 있는 모바일 웹사이트.
□ 하이브리드 앱	<b>hybrid app</b> [03장 176쪽]	네이티브 앱 안에 웹 뷰로 웹 앱을 실행하여 양쪽 방식의 장점을 취할 수 있는 앱.
□ 웹 뷰	<b>web view</b> [03장 177쪽]	앱 내에 웹 브라우저를 넣는 것.
□ 프로그레시브 웹 앱	<b>progressive web apps</b> [03장 178쪽]	진보된 브라우저 기능들을 활용하여 아이콘 추가 등 편의 기능을 더한 웹 앱.
□ 프론트엔드	<b>frontend</b> [03장 183쪽]	웹사이트의 구성 요소 중 클라이언트 컴퓨터의 브라우저에서 동작하는 부분
□ 타입스크립트	<b>typescript</b> [03장 185쪽]	자유분방한 자바스크립트에 타입을(Type) 부여해서 코딩할 때 오류의 소지를 줄이고 자동 완성 기능을 더해 생산성을 향상시킨 언어.
□ 백엔드	<b>backend</b> [03장 186쪽]	웹사이트나 모바일 앱 등 온라인 애플리케이션의 구성 요소 중 서버에서 돌아가는 프로그램.

□ 프레임워크	<b>framework</b> [03장 188쪽]
	프로그램의 기본 골격이 갖춰져 있어 개발자가 원하는 제품을 수월하게 만들 수 있도록 출시된 개발 도구.
	프레임워크 종류
	<ul style="list-style-type: none"> <li>• 이클립스 eclipse</li> <li>• 인텔리제이 IntelliJ IDEA</li> <li>• 안드로이드 스튜디오 Android Studio</li> <li>• 비주얼 스튜디오 Visual Studio</li> <li>• 엑스코드 Xcode</li> <li>• 스위프트 Swift</li> <li>• 파이참 pycharm</li> </ul>
□ API	<b>Application Programming Interface</b> [03장 190쪽]
	여러 소프트웨어의 특정 기능들을 요청하고 호출하기 위한 약속.
□ XML	<b>eXtensible Markup Language</b> [03장 193쪽]
	데이터를 저장하고 전달할 목적으로 구성된 표기 형식.
□ JSON	<b>JavaScript Object Notation</b> [03장 195쪽]
	데이터를 저장하고 전달할 목적으로 구성된 표기 형식.
□ YAML	<b>YAML Ain't Markup Language</b> [03장 196쪽]
	XML이나 JSON 파일이 프로그램 간 정보 전달에 목적이 있는 것과는 달리 주로 프로그램 설정 파일과 같이 개발자가 편리하게 읽고 작성하기 위한 용도로 사용.

□ AJAX	<b>Asynchronous Javascript And XML</b> [03장 197쪽]
	웹사이트에 필요한 정보를 언제나 서버로부터 받아오는 기법.
□ 돔	<b>DOM; Document Object Model</b> [03장 199쪽]
	HTML 요소들을 자바스크립트로 제어할 수 있도록 만드는 API, HTML 문서가 실체화된 API이다.
□ IP	<b>Internet Protocol</b> [03장 205쪽]
	데이터 통신 규약.
□ IP 주소	<b>Internet Protocol Address</b> [03장 205쪽]
	PC나 스마트폰 같은 기기의 네트워크 주소.
	<pre> graph LR     IP[IP] --- A[공인 IP]     IP --- B[사설 IP]     IP --- C[고정 IP]     IP --- D[유동 IP] </pre>
□ 도메인	<b>domain</b> [03장 208쪽]
	사람이 기억하기 쉽도록 문자로 만들어 특정 IP에 연결한 인터넷 주소.
□ DNS	<b>Domain Name Server</b> [03장 209쪽]
	IP 주소와 이에 해당하는 도메인의 IP 정보를 갱신하며 특정 도메인에 대한 요청이 들어오면 IP 주소를 찾아 알려주는 시스템.

□ URL	<b>Uniform Resource Locator</b> [03장 209쪽]
	네트워크 상에서 특정 자료가 어디에 있는가를 나타내는 주소.
□ WWW	<b>World Wide Web</b> [03장 210쪽]
	전 세계의 사람들이 정보를 공유하고 소통할 수 있는 인터넷 공간.
□ 하이퍼텍스트	<b>hypertext</b> [03장 211쪽]
	한 문서에서 다른 문서로 즉시 접근할 수 있도록 비선형적으로 연결된 텍스트.
	꼭 전체 글자를 순서대로 읽지 않아도 되는 문서를 의미한다.
□ HTTP	<b>HyperText Transfer Protocol Secure</b> [03장 211쪽]
	HTTP는 클라이언트의 요청과 서버의 응답으로 이뤄지는 통신 규약.
□ HTTPS	<b>HyperText Transfer Protocol over Secure socket layer</b> [03장 212쪽]
	HTTP에 보안 기능을 추가하여 보다 안전하게 만든 통신 규약.
□ 데이터	<b>data</b> [03장 223쪽]
	사실을 반영하며 정보로 가공되지 않은 자료로, 프로그램을 실행하는 데 필요한 기초 자료.
□ 데이터베이스	<b>database</b> [03장 224쪽]
	전산 상에 구축한 데이터 집합.

□ 데이터베이스	<b>DBMS; DataBase Management System</b> [03장 224쪽]
	데이터베이스를 구축하고 관리하는 프로그램.
□ 운영체제	<b>OS; Operating System</b> [03장 239쪽]
	사람이 컴퓨터를 쉽게 사용할 수 있도록 해 주는 소프트웨어.
□ 인터페이스	<b>interface</b> [03장 240쪽]
	두 개 이상의 시스템이나 장치가 상호 작용할 수 있는 접점(경계면).
	사람과 사물 또는 프로그램 사이에서 의사소통하도록 돕는 것은 사용자 인터페이스.
□ GUI	<b>Graphical User Interface</b> [03장 240쪽]
	그래픽 요소를 사용하여 직관적이고 편리하게 구성한 인터페이스.
□ CLI	<b>Command Line Interface</b> [03장 241쪽]
	명령어 줄을 입력하여 사용할 수 있는 인터페이스. 콘솔이라고도 한다.
□ 커널	<b>kernel</b> [03장 248쪽]
	운영체제의 핵심 기능을 담당하는 핵심 요소.
□ 오픈 소스	<b>open source</b> [03장 249쪽]
	누구나 소스 코드를 볼 수 있고 개발에 참여할 수 있는 소프트웨어.
	라이선스에 따라 유료 버전 파생이 가능.

# 04장

□ 주 기억 장치	<b>main memory</b> [04장 258쪽] 컴퓨터를 고를 때 사양에서 흔히 '램 용량'으로 표기되는 부품으로 컴퓨터의 메인 메모리.
□ CPU	<b>Central Processing Unit</b> [04장 258쪽] 컴퓨터를 통제하고 주어진 작업을 수행하는 장치.
□ 코어	<b>core</b> [04장 260쪽] CPU 중에서도 가장 핵심이 되는 부품. 코어의 개수에 따라 듀얼 코어(2개), 쿼드 코어(4개) 등으로 부른다.
□ 스레드	<b>thread</b> [04장 261쪽] 프로그램상에서 하나의 프로세스 안에서 돌아가는 하나 이상의 작업 단위. CPU상에서 스레드는 하나의 코어, 즉 하나의 로봇으로 두 대의 로봇이 일하는 듯한 효율을 낼 수 있도록 하는 기술.
□ 클럭	<b>clock</b> [04장 261쪽] 2.5GHz와 같이 기가헤르츠(GHz) 단위로 표시되는 정보로, 코어의 속도를 표현.

□ 메모리	<b>memory</b> [04장 262쪽] 컴퓨터의 작업에 사용되는 데이터를 일시적 또는 영구적으로 저장하는 장치.
□ 보조 기억 장치	<b>secondary memory</b> [04장 262쪽] 데이터를 보관하는 부품.
□ RAM	<b>Random Access Memory</b> [04장 263쪽] 메모리 어느 위치에 있는 데이터든지 같은 속도로 읽고 쓸 수 있다는 뜻. <i>RAM은 메인 메모리와 같은 뜻이다.</i>
□ 휘발성 메모리	<b>volatile memory</b> [04장 264쪽] 전기가 끊겼을 때 데이터가 날아가면 휘발성 메모리.
□ 비휘발성 메모리	<b>non-volatile memory</b> [04장 264쪽] 전기가 끊겨도 데이터가 유지되면 비휘발성 메모리.
□ 입출력 장치	<b>I/O</b> [04장 265쪽] 컴퓨터에 신호와 정보를 보내는 입력 장치와 컴퓨터의 연산을 결과로 내보내는 출력 장치.
□ 프로세스	<b>process</b> [04장 267쪽] 프로그램이 메모리에 올려져 CPU에 의해 실행되는 상태.
□ 병렬 처리	<b>parallel processing</b> [04장 267쪽] 실제로 여러 작업을 동시에 실행하는 방법.

□ 병행 처리	<b>concurrent processing</b> [04장 268쪽]	하나의 코어가 여러 프로세스를 돌아가면서 조금씩 처리하는 것.
□ 컨텍스트 스위칭	<b>context switching</b> [04장 269쪽]	하나의 프로세스가 CPU를 사용 중인 상태에서 또 다른 프로세스가 CPU를 사용할 수 있도록 하기 위해 이전 프로세스의 상태(문맥)를 보관하고 새로운 프로세스의 상태를 CPU에 적재하는 작업.
□ 비트	<b>bit</b> [04장 275쪽]	0과 1 두 값을 가지는 것으로 컴퓨터가 다루는 데이터의 최소 단위.
□ 바이트	<b>byte</b> [04장 275쪽]	비트가 여덟 개 모인 것.
□ 불리언	<b>boolean</b> [04장 276쪽]	true(참)와 false(거짓)을 갖는 값으로 보통 '예'를 나타내는 이진수 1과 '아니오'를 나타내는 이진수 0으로 표현.

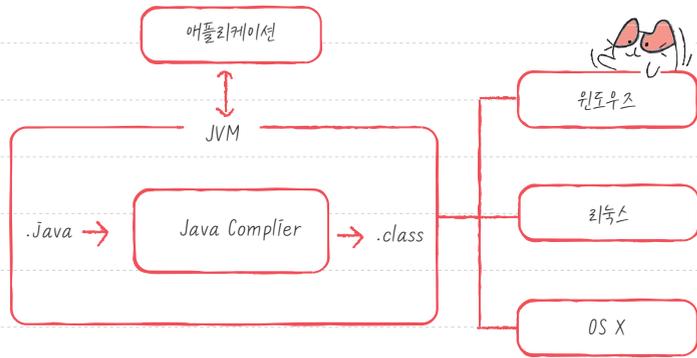
□ 자료형	<b>data type</b> [04장 278쪽]	프로그래밍 언어에서 여러 종류의 데이터를 저장하는 방식.
□ 유니코드	<b>unicode</b> [04장 281쪽]	전 세계에서 사용되는 대부분의 문자를 포함한 문자열 셋.
□ 문자열 셋	<b>character set</b> [04장 282쪽]	사용자가 입력한 문자나 기호들을 컴퓨터가 이용할 수 있는 숫자로 만든 것.
□ 문자 인코딩	<b>character encoding</b> [04장 282쪽]	사람이 사용하는 문자를 컴퓨터가 인지할 수 있는 숫자로 바꾸는 것.
□ 컴파일 언어	<b>compiled language</b> [04장 287쪽]	실행되기 전 다른 형식으로 번역되는 언어.
□ 인터프리터 언어	<b>interpreted language</b> [04장 287쪽]	작성된 코드 그대로 통역되어 실행되는 언어.
□ 컴파일	<b>compile</b> [04장 287쪽]	코드를 실행하기 전에 기계어나 다른 코드로 먼저 번역하는 것.
□ 컴파일러	<b>compiler</b> [04장 288쪽]	소스 코드를 다른 언어나 형태로 번역해 주는 프로그램.

□ 정적 타입	<b>statically typed</b> [04장 291쪽]	컴파일 언어처럼 자료형이 고정된 것.
□ 동적 타입	<b>dynamically typed</b> [04장 291쪽]	인터프리터 언어처럼 자료형이 고정되지 않은 것.
□ 절차지향 프로그래밍	<b>procedural programming</b> [04장 292쪽]	물이 위에서 아래로 흐르는 것처럼 소스 코드를 위에서부터 차례대로 읽고 실행하는 방법.
	절차 지향 편	
	① 콘을 만든다.    ② 아이스크림을 얹는다.    ③ 시럽을 뿌린다.    ④ 토핑을 얹는다.	
□ 객체지향 프로그래밍	<b>OOP; object oriented programming</b> [04장 293쪽]	프로그램을 객체란 단위로 나누어 프로그래밍하는 방식.
□ 객체	<b>object</b> [04장 293쪽]	구성 요소를 묶는 단위.
□ 클래스	<b>class</b> [04장 295쪽]	기능의 스펙을 정의하는 명세서.

□ 메소드	<b>method</b> [04장 296쪽]	특정 작업을 수행하기 위한 명령문 집합.
□ 상속	<b>inheritance</b> [04장 297쪽]	상위 클래스의 기능을 하위 클래스가 물려받아 기존 클래스에 기능을 추가하거나 재정의하는 것.
□ 함수	<b>function</b> [04장 300쪽]	프로그래밍 언어에서 기능을 표현한 것.
□ 변수	<b>variable</b> [04장 300쪽]	변수는 프로그램에서 개발자가 메인 메모리 공간에 올려놓은 값.
□ 상수	<b>constant</b> [04장 302쪽]	값이 한 번 대입되면 바꿀 수 없는 것.
□ 함수형 프로그래밍	<b>functional programming</b> [04장 300쪽]	함수형 기능들을 활용하여 변수의 사용을 최소화하는 프로그래밍 방식.
□ 매니지드 언어	<b>managed language</b> [04장 304쪽]	언어 자체에서 메모리를 관리해 주는 언어.
□ 언매니지드 언어	<b>unmanaged language</b> [04장 304쪽]	개발자가 직접 메모리를 관리해야 하는 언어.

□ 자바 가상 머신 **JVM; Java Virtual Machine** [04장 307쪽]

자바 운영체제 사이에 존재하는 가상머신으로, 어떤 운영체제에서든지 자바 파일을 실행할 수 있도록 도와주는 것.



□ 컴파일 오류 **compilation error** [04장 311쪽]

소스 코드를 컴파일하는 과정에서 생기는 오류.

□ 구문 오류 **syntax error** [04장 312쪽]

프로그래밍 언어의 사용법에 맞지 않는 코드로 인해 발생하는 오류.

□ 런타임 오류 **runtime error** [04장 313쪽]

프로그램 실행 중에 생기는 오류.

□ 논리 오류 **logical error** [04장 315쪽]

컴파일이나 실행 자체는 성공하지만 잘못된 결과를 반환하는 오류.

□ 예외 처리 **exception handling** [04장 317쪽]

오류 발생 시에도 프로세스를 지속시킬 수 있는 수단.

□ 메모리 누수 **memory leak** [04장 320쪽]

사용하지 않는 데이터가 비워지지 않고 메인 메모리 공간에 쌓여 있는 현상.

□ 가비지 컬렉션 **garbage collection** [04장 320쪽]

메인 메모리상에서 불필요한 데이터를 자동으로 치우는 기능.

□ 패키지 매니저 **package manager** [04장 327쪽]

라이브러리의 설치, 업데이트, 삭제 등을 관리하는 소프트웨어 도구.

□ 환경 변수 **environment variable** [04장 329쪽]

프로그램이 작동할 환경에 지정된 변경 가능한 값.

□ 런타임 환경 **runtime environment** [04장 333쪽]

프로그램이 동작할 수 있도록 해 주는 소프트웨어.

□ 자바 런타임 환경 **JRE; Java runtime environment** [04장 334쪽]

자바로 작성한 소스 코드를 컴파일했을 때 생성된 자바 바이트코드를 실행할 수 있도록 해 주는 소프트웨어.

자바 API와 JVM으로 구성된다.

□ SDK **Software Development Kit** [04장 335쪽]

특정 언어나 환경의 소프트웨어를 개발할 수 있도록 제공되는 도구.

□JDK	<b>Java Development Kit</b> [04장 335쪽]
	자바 코드를 컴파일하고 테스트하는 기능을 포함하는 등 자바 언어로 프로그램을 개발할 수 있는 환경까지 갖춰 주는 것.
□라이브러리	<b>library</b> [04장 337쪽]
	하나 이상의 프로그램에 활용될 수 있는 데이터와 명령어들의 집합. 소스 코드에 가져다 쓸 수 있다.
□VCS	<b>Version Control System</b> [04장 343쪽]
	프로젝트의 변경 내역을 관리하는 소프트웨어.
□커밋	<b>commit</b> [04장 347쪽]
	의미 있는 변경 작업들을 저장소에 기록하는 것. 커밋의 변경 내역을 알려주는 이력을 커밋 메시지라고 한다.
□리셋	<b>reset</b> [04장 348쪽]
	특정 시점의 커밋으로 되돌아가는 행위.
□리버트	<b>revert</b> [04장 348쪽]
	되돌리고자 하는 커밋을 거꾸로 실행해서 해당 부분만 복구하는 행위.
□브랜치	<b>branch</b> [04장 349쪽]
	커밋 사이를 가볍게 이동할 수 있는 포인터 같은 것.
□깃 호스팅 서비스	<b>git hosting service</b> [04장 351쪽]
	프로젝트를 관리하는 것에 공용 저장소를 제공하는 서비스.

## MEMO

