

혼자 공부하며 함께 만드는

혼공 용어 노트

목차

가나다순

가비지 컬렉션 garbage collection	29	런타임 오류 runtime error	28
개발 development	10	런타임 환경 runtime environment	29
객체 object	26	리버트 revert	30
객체지향 프로그래밍 OOP	26	리셋 reset	30
고수준 언어 high-level language	11	리팩토링 refactoring	12
구문 오류 syntax error	28	매니지드 언어 managed language	27
기계어 machine language	11	메모리 memory	23
깃 호스팅 서비스 git hosting service	30	메모리 누수 memory leak	29
네이티브 앱 native app	16	메소드 method	27
논리 오류 logical error	28	명령문 command statement	10
데이터 data	20	문서화 documentation	12
데이터 센터 DC/IDC	13	문자 인코딩 character encoding	25
데이터베이스 database	20	문자열 셋 character set	25
데이터베이스 관리 시스템 DBMS	21	바이트 byte	24
도메인 domain	19	반응형 웹사이트 responsive website	15
돔 DOM	19	배포 deploy	12
동적 타입 dynamically typed	26	백엔드 backend	17
디버그 debug	12	변수 variable	27
라이브러리 library	30	병렬 처리 parallel processing	23
램 RAM	23	병행 처리 concurrent processing	24

보조 기억 장치 secondary memory	23	온프레미스 on-premise	13
불리언 boolean	24	운영체제 OS	21
브랜치 branch	30	웹 뷰 web view	16
비트 bit	24	웹 브라우저 web browser	14
비휘발성 메모리 non-volatile memory	23	웹 서핑 web surfing	15
빌드 build	12	웹 앱 web app	16
상속 inheritance	27	웹사이트 web site	15
상수 constant	27	유니코드 unicode	25
서버 server	13	유지 보수 maintenance	12
서버 호스팅 server hosting	13	인공지능 AI	09
세션 session	16	인터페이스 interface	21
소스 코드 source code	10	인터프리터 언어 interpreted language	25
소프트웨어 software	10	입출력 장치 I/O	23
스레드 thread	22	자료형 data type	25
시스템 프로그램 system program	10	자바 가상 머신 JVM	28
애플리케이션 application	11	자바 런타임 환경 JRE	29
어셈블리어 assembly language	11	자바스크립트 javascript	15
언매니지드 언어 unmanaged language	27	저수준 언어 low-level language	11
예외 처리 exception handling	28	적응형 웹사이트 adaptive website	15
오픈 소스 open source	21		

절차지향 프로그래밍		클라이언트 client	13
procedural programming	26	클래스 class	26
정적 타입 statically typed	26	클럭 clock	22
주 기억 장치 main memory	22	타입스크립트 typescript	17
주석 comment	10	토큰 token	16
캐시 cache	16	통합 개발 환경 IDE	11
커널 kernel	21	패키지 매니저 package manager	29
커밋 commit	30	프레임워크 framework	18
컨텍스트 스위칭 context switching	24	프로그래밍 programming	10
컴파일 compile	25	프로그래밍 언어 programming language	11
컴파일 언어 compiled language	25	프로그램 program	11
컴파일 오류 compilation error	28	프로세스 process	23
컴파일러 compiler	25	프론트엔드 frontend	17
코드 리뷰 code review	12	하이브리드 앱 hybrid app	16
코딩 coding	10	하이퍼텍스트 hypertext	20
코어 core	22	함수 function	27
쿠키 cookie	15	함수형 프로그래밍 functional programming	27
크래커 cracker	09	해커 security hacker	09
크로스 플랫폼 cross platform	12	환경 변수 environment variable	29
클라우드 컴퓨팅 서비스	14	휘발성 메모리 volatile memory	23

ABC 순

adaptive website 적응형 웹사이트	15	clock 클럭	22
AI Artificial Intelligence 인공지능	09	cloud computing service	
AJAX Asynchronous Javascript And XML	19	클라우드 컴퓨팅 서비스	14
API Application Programming Interface	18	code review 코드 리뷰	12
application 애플리케이션	11	coding 코딩	10
assembly language 어셈블리어	11	command statement 명령문	10
backend 백엔드	17	comment 주석	10
bit 비트	24	commit 커밋	30
boolean 불리언	24	commit message 커밋 메시지	30
branch 브랜치	30	compilation error 컴파일 오류	28
build 빌드	12	compile 컴파일	25
byte 바이트	24	compiled language 컴파일 언어	25
cache 캐시	16	compiler 컴파일러	25
CDN Content Delivery Network	16	concurrent processing 병행 처리	24
character encoding 문자 인코딩	25	constant 상수	27
character set 문자열 셋	25	context switching 컨텍스트 스위칭	24
child class 자식 클래스	16	cookie 쿠키	16
class 클래스	26	core 코어	22
CLI Command Line Interface	21	CPU Central Processing Unit	22
client 클라이언트	13	cracker 크래커	09

cross platform 크로스 플랫폼	12	frontend 프론트엔드	17
CSS Cascading Style Sheets	14	function 함수	27
data 데이터	20	functional programming 함수형 프로그래밍	27
data type 자료형	25	garbage collection 가비지 컬렉션	29
database 데이터베이스	21	git hosting service 깃 호스팅 서비스	30
DBMS DataBase Management System		GUI Graphical User Interface	21
데이터베이스 관리 시스템	20	high-level language 고수준 언어	11
DC Data Center / IDC Internet Data Center		HTML HyperText Markup Language	14
데이터 센터	13	HTTP HyperText Transfer Protocol Secure	20
debug 디버그	12	HTTPS HyperText Transfer Protocol over Secure	
deploy 배포	12	socket layer	20
development 개발	10	hybrid app 하이브리드 앱	16
DNS Domain Name Server	19	hypertext 하이퍼텍스트	20
documentation 문서화	12	laaS Infrastructure as a Service 아이아스	14
DOM Document Object Model 돔	19	IDE Integrated Development Environment	
domain 도메인	19	통합 개발 환경	11
dynamically typed 동적 타입	26	inheritance 상속	27
environment variable 환경 변수	29	I/O 입출력 장치	23
exception handling 예외 처리	28	interface 인터페이스	21
framework 프레임워크	18	interpreted language 인터프리터 언어	25

IP Internet Protocol 인터넷 프로토콜	18	non-volatile memory 비휘발성 메모리	23
IP Address IP 주소	19	object 객체	26
javascript 자바스크립트	15	on-premise 온프레미스	13
JDK Java Development Kit	30	OOP Object Oriented Programming	
JRE Java runtime environment		객체지향 프로그래밍	26
자바 런타임 환경	29	open source 오픈 소스	21
JSON JavaScript Object Notation	18	OS Operating System 운영체제	21
JVM Java Virtual Machine 자바 가상 머신	28	PaaS Platform as a Service 파스	14
kernel 커널	21	package manager 패키지 매니저	29
library 라이브러리	30	parallel processing 병렬 처리	23
logical error 논리 오류	28	parent class 부모 클래스	29
low-level language 저수준 언어	11	procedural programming	
machine language 기계어	11	절차지향 프로그래밍	26
main memory 주 기억 장치	22	process 프로세스	23
maintenance 유지 보수	12	program 프로그램	11
managed language 매니지드 언어	27	programming 프로그래밍	10
memory 메모리	23	programming language 프로그래밍 언어	11
memory leak 메모리 누수	29	progressive web apps 프로그레시브 웹 앱	17
method 메소드	27	RAM Random Access Memory 램	23
native app 네이티브 앱	16	refactoring 리팩토링	12

reset 리셋	30	unicode 유니코드	25
responsive website 반응형 웹사이트	15	unmanaged language 언매니지드 언어	27
revert 리버트	30	URL Uniform Resource Locator	20
runtime environment 런타임 환경	29	variable 변수	27
runtime error 런타임 오류	28	VCS Version Control System	30
SaaS Software as a Service 싸스	14	volatile memory 휘발성 메모리	23
SDK Software Development Kit	29	web app 웹 앱	16
secondary memory 보조 기억 장치	23	web browser 웹 브라우저	14
security hacker 해커	12	web site 웹사이트	15
server 서버	13	web surfing 웹 서핑	15
server hosting 서버 호스팅	13	web view 웹 뷰	17
session 세션	16	WWW World Wide Web	20
software 소프트웨어	10	XML eXtensible Markup Language	18
source code 소스 코드	10	YAML YAML Ain't Markup Language	18
statically typed 정적 타입	26		
syntax error 구문 오류	28		
system program 시스템 프로그램	10		
thread 스레드	22		
token 토큰	16		
typescript 타입스크립트	17		

01 장

□ 크래커

cracker

[01장 037쪽]

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 037쪽]

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

□ 인공지능

AI; Artificial Intelligence

[01장 039쪽]

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

□ 명령문	command statemen [01장 043쪽]	컴퓨터에 각종 지시를 내리는 문구.
□ 소스 코드	source code [01장 043쪽]	개발자가 작업하는 작업물로, 주로 실행 프로그램을 만드는 과정을 입력하는 데 이용하는 텍스트 형태의 파일.
□ 주석	comment [01장 043쪽]	각 코드가 어떤 내용인지 개발자가 쉽게 알아볼 수 있도록 설명하는 글.
□ 코딩	coding [01장 043쪽]	프로그래밍 언어로 된 코드를 입력하는 작업 자체.
□ 프로그래밍	programming [01장 044쪽]	컴퓨터가 할 일의 절차와 알고리즘을 설계하는 것.
□ 개발	development [01장 046쪽]	소프트웨어를 설계, 구현, 운영, 관리하는 데 필요한 전반적인 기술 과정.
□ 소프트웨어	software [01장 048쪽]	프로그램과 라이브러리, 데이터 등으로 이뤄진 시스템.
□ 시스템 프로그램	system program [01장 049쪽]	운영체제의 일부로서 컴퓨터 이용 환경을 조성하는 소프트웨어.

□ 애플리케이션	application	[01장 049쪽]
	PC나 모바일 기기에서 사용자가 직접 사용하는 모든 프로그램.	
□ 프로그램	program	[01장 047쪽]
	사용자의 명령에 따라 그 목적에 맞는 작업을 수행하는 일련의 명령어 모음.	
□ 프로그래밍 언어	programming language	[01장 053쪽]
	컴퓨터와 소통하기 위해 만들어진 언어.	
□ 고수준 언어	high-level language	[01장 057쪽]
	사람의 언어에 가까운 언어.	
□ 기계어	machine language	[01장 057쪽]
	컴퓨터가 이해할 수 있는 숫자인 0과 1로만 구성된 언어.	
□ 어셈블리어	assembly language	[01장 057쪽]
	기계어 바로 위 단계의 저수준 언어.	
□ 저수준 언어	low-level language	[01장 057쪽]
	기계어에 가까운 언어.	
□ 통합 개발 환경	IDE; Integrated Development Environment	[01장 059쪽]
	IDE는 개발에 관련된 다양한 기능들을 제공하는 개발용 프로그램.	

□ 디버그	debug	[01장 065쪽]
	오류 수정 프로그램과 그 작업 자체를 의미.	
	프로그램에서 문제를 찾아 제거하는 행동을 디버깅이라고 한다.	
□ 빌드	build	[01장 068쪽]
	프로그래밍한 소스 코드를 묶어 실행 가능한 파일을 만드는 것.	
□ 배포	deploy	[01장 070쪽]
	소프트웨어를 사용자들에게 전달하는 것.	
□ 유지 보수	maintenance	[01장 071쪽]
유지 보수는 소프트웨어 개발에 중요!	소프트웨어 제품 출시 이후 계속되는 문제 해결 및 각종 업데이트 작업.	
□ 리팩토링	refactoring	[01장 073쪽]
	기능을 수정하지 않으면서 코드의 질을 높이는 것.	
□ 코드 리뷰	code review	[01장 073쪽]
	서로의 코드를 확인하고 피드백을 주고받는 과정.	
□ 문서화	documentation	[01장 075쪽]
	소프트웨어와 그 소스 코드를 쉽게 파악할 수 있는 문서.	

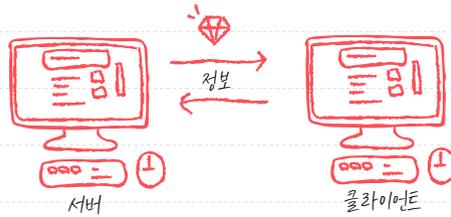
02장

□ 서버

server

[02장 082쪽]

정보나 서비스를 제공하는 컴퓨터.



□ 클라이언트

client

[02장 083쪽]

서버가 제공하는 것을 받아 사용하는 컴퓨터.

□ 데이터 센터

DC; Data Center / IDC; Internet Data Center

[02장 084쪽]

수많은 서버를 한 곳에서 안정적으로 관리하는 시설.

□ 서버 호스팅

server hosting

[02장 085쪽]

서버용 컴퓨터를 임대해 주는 서비스.

□ 온프레미스

on-premise

[02장 085쪽]

호스팅을 외부 데이터 센터에 두기에 민감한 정보를 다루는 회사들이 사내 전산실에 서버를 갖추고 관리하는 것.

□ 클라우드 컴퓨팅 서비스	cloud computing service [02장 086쪽] 서버를 가상화하여 각종 편의 기능과 함께 필요한 만큼 사용할 수 있는 서비스.
□ IaaS	Infrastructure as a Service [02장 088쪽] 클라우드를 이용해서 서버용 인프라, 즉 가상화된 서버 컴퓨터를 대여해주는 서비스.
□ PaaS	Platform as a Service [02장 089쪽] 애플리케이션을 개발하고 서비스하기 위해 필요한 서버, 운영체제, 개발 환경 등을 자동으로 설치하고 제공함으로써 사용자가 애플리케이션 개발에만 집중할 수 있도록 플랫폼을 제공하는 서비스.
□ SaaS	Software as a Service [02장 090쪽] 소프트웨어가 이미 완성된 형태로 사용자에게 제공되는 클라우드 서비스.
□ 웹 브라우저	web browser [02장 095쪽] 웹 서핑에 사용되는 소프트웨어.
□ HTML	HyperText Markup Language [02장 096쪽] 웹 페이지에 요소들을 '가져다 놓는' 마크업 언어.
□ CSS	Cascading Style Sheets [02장 096쪽] HTML로 올려놓은 요소들을 '꾸미는' 스타일 언어.

□ 자바스크립트

javascript

[02장 096쪽]

웹 페이지에 기능들을 부여해 '일을 시키는' 프로그래밍 언어.

□ 웹사이트

web site

[02장 097쪽]

단일 또는 다수의 웹 페이지로 구성.

□ 웹 서핑

web surfing

[02장 101쪽]

바다에서 파도를 타듯 이리저리 움직이며 웹 구석구석을 둘러본다는 뜻.



□ 반응형 웹사이트

responsive website

[02장 105쪽]

하나의 웹 페이지 화면 크기에 따라 내부 요소도 변경되는 웹사이트.

반응형 웹이라고도 한다.

□ 적응형 웹사이트

adaptive website

[02장 106쪽]

화면의 크기별로 웹 페이지를 따로따로 만든 웹사이트.

적응형 웹이라고도 한다.

□ 쿠키	cookie	[02장 111쪽]
	사용자의 브라우저에 저장되는 정보.	
□ 세션	session	[02장 112쪽]
	서버가 사용자를 기억하고 있는 상태.	
□ 토큰	token	[02장 115쪽]
	서버가 기억해 둘 필요 없이 사용자가 스스로를 증명할 수 있는 수단.	
□ 캐시	cache	[02장 116쪽]
	다시 가져오지 않아도 되도록 데이터를 가까이 저장해 두는 기술. 메인 메모리 속 데이터에 보다 더 빠르게 접근할 수 있도록 CPU에 내장되는 형태.	
□ CDN	Content Delivery Network	[02장 117쪽]
	각지에 캐시 서버를 두어 부하를 분산시키는 기술.	
□ 네이티브 앱	native app	[02장 123쪽]
	동작할 운영체제에 특화된 방식으로 제작된 앱.	
□ 크로스 플랫폼	cross platform	[02장 125쪽]
	하나의 소스 코드로 안드로이드와 iOS 양쪽에서 모두 동작하는 앱을 만들 수 있는 플랫폼.	

□ 웹 앱	web app [02장 126쪽]	스마트폰의 브라우저에서 접속할 수 있는 모바일 웹사이트.
□ 하이브리드 앱	hybrid app [02장 128쪽]	네이티브 앱 안에 웹 뷰로 웹 앱을 실행하여 양쪽 방식의 장점을 취할 수 있는 앱.
□ 웹 뷰	web view [02장 129쪽]	앱 내에 웹 브라우저를 넣는 것.
□ 프로그레시브 웹 앱	progressive web apps [02장 130쪽]	진보된 브라우저 기능들을 활용하여 아이콘 추가 등 편의 기능을 더한 웹 앱.
□ 프론트엔드	frontend [02장 135쪽]	웹사이트의 구성 요소 중 클라이언트 컴퓨터의 브라우저에서 동작하는 부분
□ 타입스크립트	typescript [02장 137쪽]	자유분방한 자바스크립트에 타입을(Type) 부여해서 코딩할 때 오류의 소지를 줄이고 자동 완성 기능을 더해 생산성을 향상시킨 언어.
□ 백엔드	backend [02장 138쪽]	웹사이트나 모바일 앱 등 온라인 애플리케이션의 구성 요소 중 서버에서 돌아가는 프로그램.

□ 프레임워크

framework

[02장 140쪽]

프로그램의 기본 골격이 갖춰져 있어 개발자가 원하는 제품을 수월하게 만들 수 있도록 출시된 개발 도구.

프레임워크 종류

- 이클립스 eclipse
- 인텔리제이 IntelliJ IDEA
- 안드로이드 스튜디오 Android Studio
- 비주얼 스튜디오 Visual Studio
- 엑스코드 Xcode
- 스위프트 Swift
- 파이참 pycharm

□ API

Application Programming Interface

[02장 142쪽]

여러 소프트웨어의 특정 기능들을 요청하고 호출하기 위한 약속.

□ XML

eXtensible Markup Language

[02장 145쪽]

데이터를 저장하고 전달할 목적으로 구성된 표기 형식.

□ JSON

JavaScript Object Notation

[02장 147쪽]

데이터를 저장하고 전달할 목적으로 구성된 표기 형식.

□ YAML

YAML Ain't Markup Language

[02장 148쪽]

XML이나 JSON 파일이 프로그램 간 정보 전달에 목적이 있는 것과는 달리 주로 프로그램 설정 파일과 같이 개발자가 편리하게 읽고 작성하기 위한 용도로 사용.

□ **AJAX**

Asynchronous Javascript And XML

[02장 149쪽]

웹사이트에 필요한 정보를 언제든 서버로부터 받아오는 기법.

□ **돔**

DOM; Document Object Model

[02장 151쪽]

HTML 요소들을 자바스크립트로 제어할 수 있도록 만드는 API,
HTML 문서가 실체화된 API이다.

□ **IP**

Internet Protocol

[02장 157쪽]

데이터 통신 규약.

□ **IP 주소**

Internet Protocol Address

[02장 157쪽]

PC나 스마트폰 같은 기기의 네트워크 주소.



□ **도메인**

domain

[02장 160쪽]

사람이 기억하기 쉽도록 문자로 만들어 특정 IP에 연결한 인터넷 주소.

□ **DNS**

Domain Name Server

[02장 161쪽]

IP 주소와 이에 해당하는 도메인의 IP 정보를 갱신하며 특정 도메인에 대한 요청이 들어오면 IP 주소를 찾아 알려주는 시스템.

□ URL	Uniform Resource Locator [02장 161쪽]
	네트워크 상에서 특정 자료가 어디에 있는가를 나타내는 주소.
□ WWW	World Wide Web [02장 162쪽]
	전 세계의 사람들이 정보를 공유하고 소통할 수 있는 인터넷 공간.
□ 하이퍼텍스트	hypertext [02장 163쪽]
	한 문서에서 다른 문서로 즉시 접근할 수 있도록 비선형적으로 연결된 텍스트.
	꼭 전체 글자를 순서대로 읽지 않아도 되는 문서를 의미한다.
□ HTTP	HyperText Transfer Protocol Secure [02장 163쪽]
	HTTP는 클라이언트의 요청과 서버의 응답으로 이뤄지는 통신 규약.
□ HTTPS	HyperText Transfer Protocol over Secure socket layer [02장 165쪽]
	HTTP에 보안 기능을 추가하여 보다 안전하게 만든 통신 규약.
□ 데이터	data [02장 175쪽]
	사실을 반영하며 정보로 가공되지 않은 자료로, 프로그램을 실행하는 데 필요한 기초 자료.
□ 데이터베이스	database [02장 176쪽]
	전산 상에 구축한 데이터 집합.

□ 데이터베이스 관리 시스템	DBMS; DataBase Management System 데이터베이스를 구축하고 관리하는 프로그램.	[02장 177쪽]
□ 운영체제	OS; Operating System 사람이 컴퓨터를 쉽게 사용할 수 있도록 해 주는 소프트웨어.	[02장 191쪽]
□ 인터페이스	interface 두 개 이상의 시스템이나 장치가 상호 작용할 수 있는 접점(경계면). 사람과 사물 또는 프로그램 사이에서 의사소통하도록 돕는 것은 사용자 인터페이스.	[02장 192쪽]
□ GUI	Graphical User Interface 그래픽 요소를 사용하여 직관적이고 편리하게 구성한 인터페이스.	[02장 192쪽]
□ CLI	Command Line Interface 명령어 줄을 입력하여 사용할 수 있는 인터페이스. 콘솔이라고도 한다.	[02장 193쪽]
□ 커널	kernel 운영체제의 핵심 기능을 담당하는 핵심 요소.	[02장 200쪽]
□ 오픈 소스	open source 누구나 소스 코드를 볼 수 있고 개발에 참여할 수 있는 소프트웨어. 라이선스에 따라 유료 버전 파생이 가능.	[02장 201쪽]

03장

□ 주 기억 장치	main memory [03장 210쪽]
	컴퓨터를 고를 때 사양에서 흔히 ‘램 용량’으로 표기되는 부품으로 컴퓨터의 메인 메모리.
□ CPU	Central Processing Unit [03장 210쪽]
	컴퓨터를 통제하고 주어진 작업을 수행하는 장치.
□ 코어	core [03장 212쪽]
	CPU 중에서도 가장 핵심이 되는 부품. 코어의 개수에 따라 듀얼 코어(2개), 쿼드 코어(4개) 등으로 부른다.
□ 스레드	thread [03장 213쪽]
	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	clock [03장 213쪽]
	2.5GHz와 같이 기가헤르츠(GHz) 단위로 표시되는 정보로, 코어의 속도를 표현.

□ 메모리	memory	[03장 214쪽]
	컴퓨터의 작업에 사용되는 데이터를 일시적 또는 영구적으로 저장하는 장치.	
□ 보조 기억 장치	secondary memory	[03장 214쪽]
	데이터를 보관하는 부품.	
□ RAM	Random Access Memory	[03장 215쪽]
RAM은 메인 메모리와 같은 뜻이다.	메모리 어느 위치에 있는 데이터든지 같은 속도로 읽고 쓸 수 있다는 뜻.	
□ 휘발성 메모리	volatile memory	[03장 216쪽]
	전기가 끊겼을 때 데이터가 날아가면 휘발성 메모리.	
□ 비휘발성 메모리	non-volatile memory	[03장 216쪽]
	전기가 끊겨도 데이터가 유지되면 비휘발성 메모리.	
□ 입출력 장치	I/O	[03장 217쪽]
	컴퓨터에 신호와 정보를 보내는 입력 장치와 컴퓨터의 연산을 결과로 내보내는 출력 장치.	
□ 프로세스	process	[03장 219쪽]
	프로그램이 메모리에 올려져 CPU에 의해 실행되는 상태.	
□ 병렬 처리	parallel processing	[03장 219쪽]
	실제로 여러 작업을 동시에 실행하는 방법.	

□ 병행 처리

concurrent processing

[03장 220쪽]

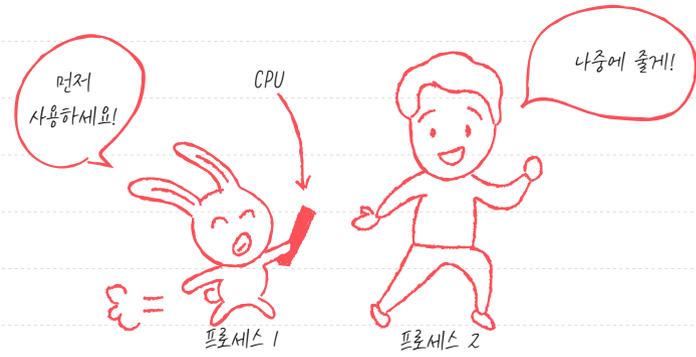
하나의 코어가 여러 프로세스를 돌아가면서 조금씩 처리하는 것.

□ 컨텍스트 스위칭

context switching

[03장 221쪽]

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

bit

[03장 227쪽]

0과 1 두 값을 가지는 것으로 컴퓨터가 다루는 데이터의 최소 단위.

□ 바이트

byte

[03장 229쪽]

비트가 여덟 개 모인 것.

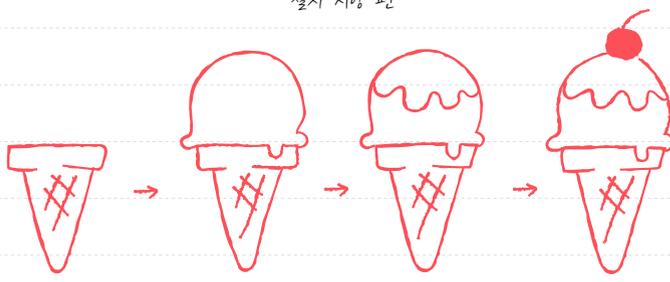
□ 불리언

boolean

[03장 228쪽]

true(참)와 false(거짓)을 갖는 값으로 보통 '예'를 나타내는 이진수 1과 '아니오'를 나타내는 이진수 0으로 표현.

□ 자료형	data type	[03장 230쪽]
	프로그래밍 언어에서 여러 종류의 데이터를 저장하는 방식.	
□ 유니코드	unicode	[03장 235쪽]
	전 세계에서 사용되는 대부분의 문자를 포함한 문자열 셋.	
□ 문자열 셋	character set	[03장 234쪽]
	사용자가 입력한 문자나 기호들을 컴퓨터가 이용할 수 있는 숫자로 만든 것.	
□ 문자 인코딩	character encoding	[03장 234쪽]
	사람이 사용하는 문자를 컴퓨터가 인지할 수 있는 숫자로 바꾸는 것.	
□ 컴파일 언어	compiled language	[03장 239쪽]
	실행되기 전 다른 형식으로 번역되는 언어.	
□ 인터프리터 언어	interpreted language	[03장 239쪽]
	작성된 코드 그대로 통역되어 실행되는 언어.	
□ 컴파일	compile	[03장 239쪽]
	코드를 실행하기 전에 기계어나 다른 코드로 먼저 번역하는 것.	
□ 컴파일러	compiler	[03장 240쪽]
	소스 코드를 다른 언어나 형태로 번역해 주는 프로그램.	

<p>□ 정적 타입</p>	<p>statically typed [03장 243쪽]</p> <p>컴파일 언어처럼 자료형이 고정된 것.</p>
<p>□ 동적 타입</p>	<p>dynamically typed [03장 243쪽]</p> <p>인터프리터 언어처럼 자료형이 고정되지 않은 것.</p>
<p>□ 절차지향 프로그래밍</p>	<p>procedural programming [03장 244쪽]</p> <p>물이 위에서 아래로 흐르는 것처럼 소스 코드를 위에서부터 차례대로 읽고 실행하는 방법.</p>
<p>절차 지향 편</p>  <p>① 콘을 만든다. ② 아이스크림을 얹는다. ③ 시럽을 뿌린다. ④ 토핑을 얹는다.</p>	
<p>□ 객체지향 프로그래밍</p>	<p>OOP; object oriented programming [03장 245쪽]</p> <p>프로그램을 객체란 단위로 나누어 프로그래밍하는 방식.</p>
<p>□ 객체</p>	<p>object [03장 245쪽]</p> <p>구성 요소를 묶는 단위.</p>
<p>□ 클래스</p>	<p>class [03장 247쪽]</p> <p>기능의 스펙을 정의하는 명세서.</p>

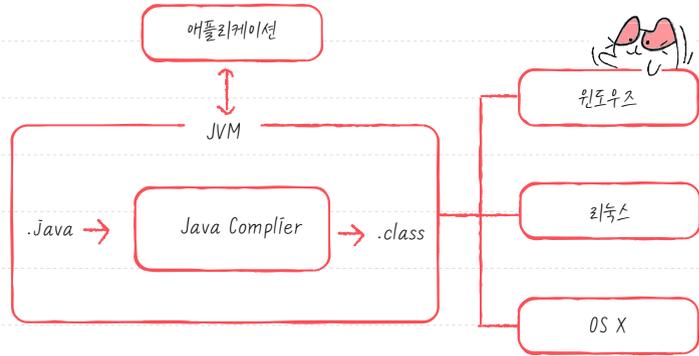
□ 메소드	method	[03장 248쪽]
	특정 작업을 수행하기 위한 명령문 집합.	
□ 상속	inheritance	[03장 251쪽]
	상위 클래스의 기능을 하위 클래스가 물려받아 기존 클래스에 기능을 추가하거나 재정의하는 것.	
□ 함수	function	[03장 255쪽]
	프로그래밍 언어에서 기능을 표현한 것.	
□ 변수	variable	[03장 252쪽]
	변수는 프로그램에서 개발자가 메인 메모리 공간에 올려놓은 값.	
□ 상수	constant	[03장 252쪽]
	값이 한 번 대입되면 바꿀 수 없는 것.	
□ 함수형 프로그래밍	functional programming	[03장 252쪽]
	함수형 기능들을 활용하여 변수의 사용을 최소화하는 프로그래밍 방식.	
□ 매니지드 언어	managed language	[03장 256쪽]
	언어 자체에서 메모리를 관리해 주는 언어.	
□ 언매니지드 언어	unmanaged language	[03장 256쪽]
	개발자가 직접 메모리를 관리해야 하는 언어.	

□ 자바 가상 머신

JVM; Java Virtual Machine

[03장 259쪽]

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



□ 컴파일 오류

compilation error

[03장 263쪽]

소스 코드를 컴파일하는 과정에서 생기는 오류.

□ 구문 오류

syntax error

[03장 264쪽]

프로그래밍 언어의 사용법에 맞지 않는 코드로 인해 발생하는 오류.

□ 런타임 오류

runtime error

[03장 265쪽]

프로그램 실행 중에 생기는 오류.

□ 논리 오류

logical error

[03장 267쪽]

컴파일이나 실행 자체는 성공하지만 잘못된 결과를 반환하는 오류.

□ 예외 처리

exception handling

[03장 269쪽]

오류 발생 시에도 프로세스를 지속시킬 수 있는 수단.

□ 메모리 누수	memory leak	[03장 272쪽]
	사용하지 않는 데이터가 비워지지 않고 메인 메모리 공간에 쌓여 있는 현상.	
□ 가비지 컬렉션	garbage collection	[03장 272쪽]
	메인 메모리상에서 불필요한 데이터를 자동으로 치우는 기능.	
□ 패키지 매니저	package manager	[03장 279쪽]
	라이브러리의 설치, 업데이트, 삭제 등을 관리하는 소프트웨어 도구.	
□ 환경 변수	environment variable	[03장 281쪽]
	프로그램이 작동할 환경에 지정된 변경 가능한 값.	
□ 런타임 환경	runtime environment	[03장 285쪽]
	프로그램이 동작할 수 있도록 해 주는 소프트웨어.	
□ 자바 런타임 환경	JRE; Java runtime environment	[03장 286쪽]
	자바로 작성한 소스 코드를 컴파일했을 때 생성된 자바 바이트코드를 실행할 수 있도록 해 주는 소프트웨어.	
	자바 API와 JVM으로 구성된다.	
□ SDK	Software Development Kit	[03장 287쪽]
	특정 언어나 환경의 소프트웨어를 개발할 수 있도록 제공되는 도구.	

□ JDK	Java Development Kit [03장 287쪽]
	자바 코드를 컴파일하고 테스트하는 기능을 포함하는 등 자바 언어로 프로그램을 개발할 수 있는 환경까지 갖춰 주는 것.
□ 라이브러리	library [03장 290쪽]
	하나 이상의 프로그램에 활용될 수 있는 데이터와 명령어들의 집합. 소스 코드에 가져다 쓸 수 있다.
□ VCS	Version Control System [03장 295쪽]
	프로젝트의 변경 내역을 관리하는 소프트웨어.
□ 커밋	commit [03장 299쪽]
	의미 있는 변경 작업들을 저장소에 기록하는 것. 커밋의 변경 내역을 알려주는 이력을 커밋 메시지라고 한다.
□ 리셋	reset [03장 300쪽]
	특정 시점의 커밋으로 되돌아가는 행위.
□ 리버트	revert [03장 300쪽]
	되돌리고자 하는 커밋을 거꾸로 실행해서 해당 부분만 복구하는 행위.
□ 브랜치	branch [03장 302쪽]
	커밋 사이를 가볍게 이동할 수 있는 포인터 같은 것.
□ 깃 호스팅 서비스	git hosting service [03장 304쪽]
	프로젝트를 관리하는 깃에 공용 저장소를 제공하는 서비스.

