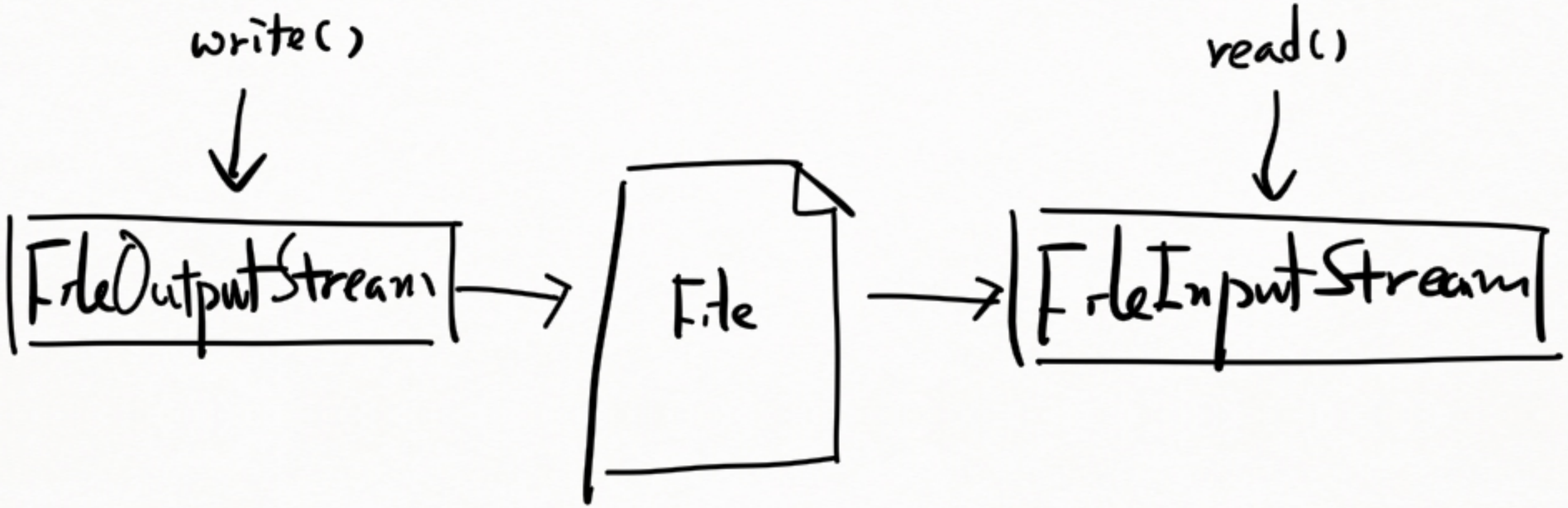
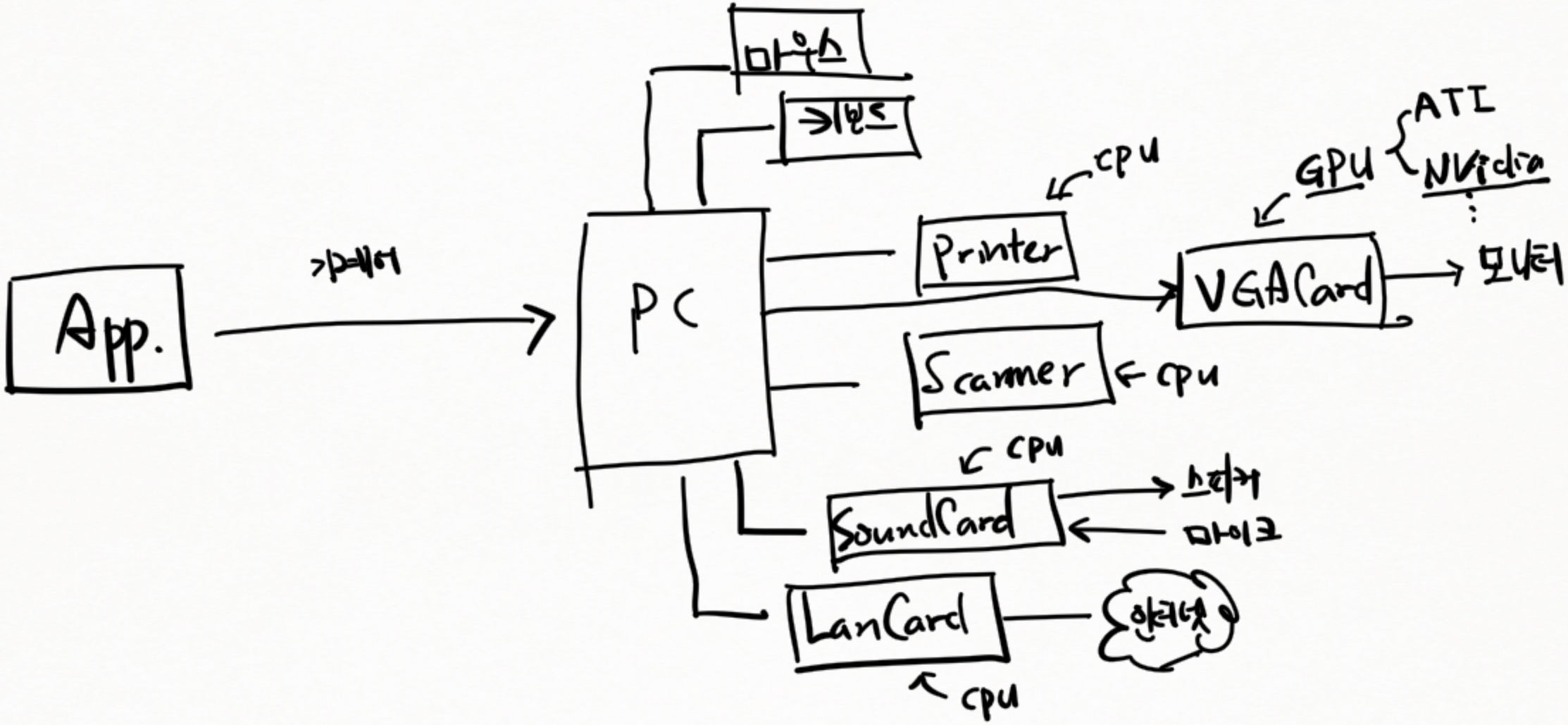
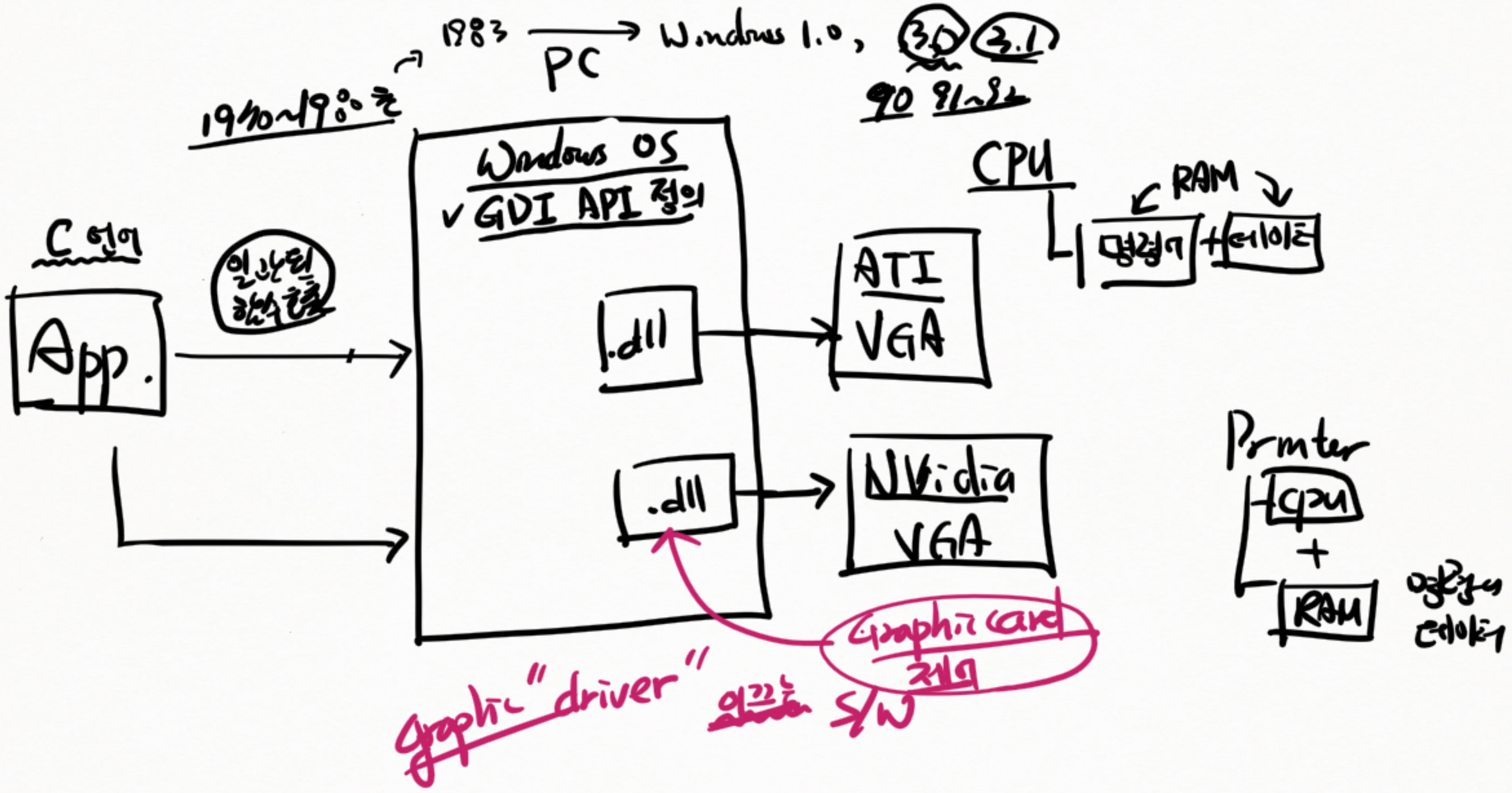
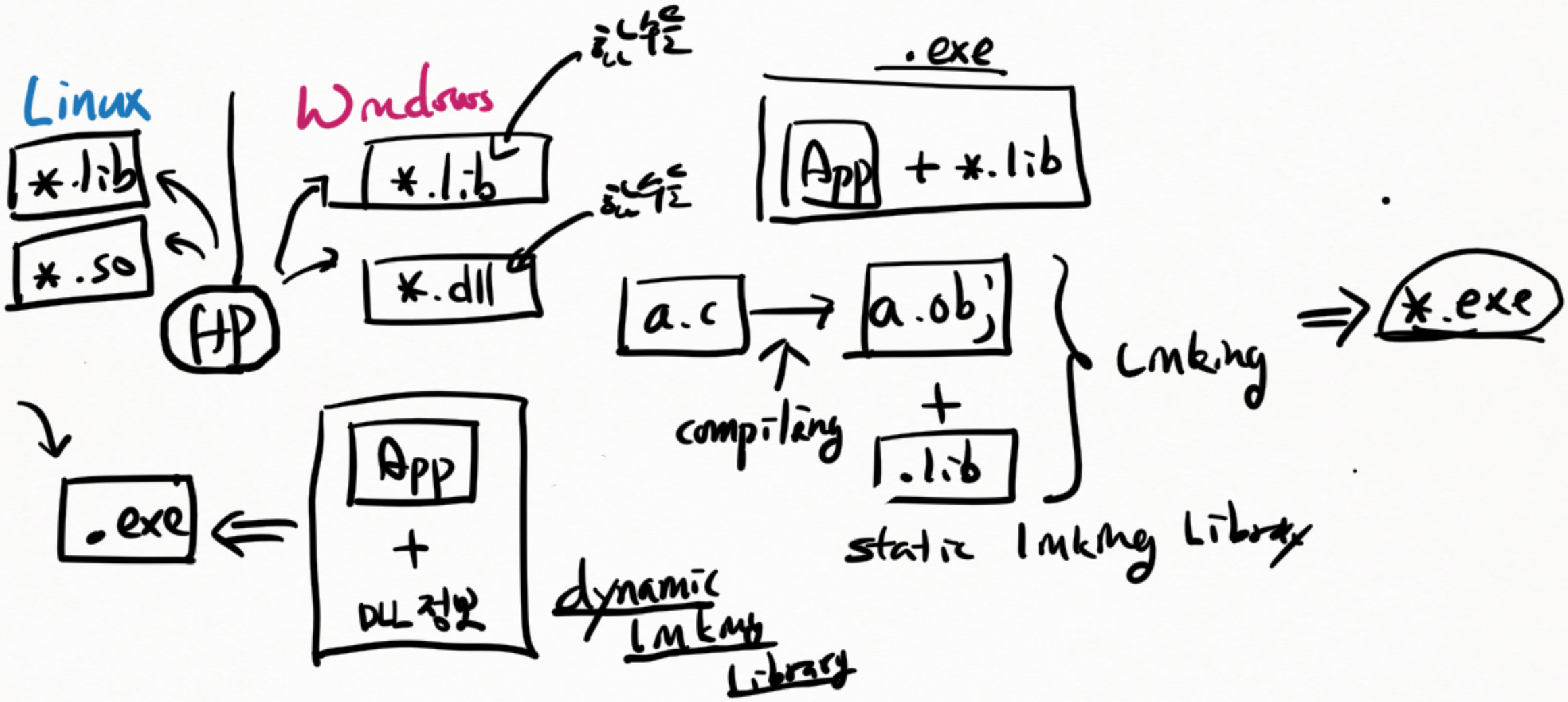
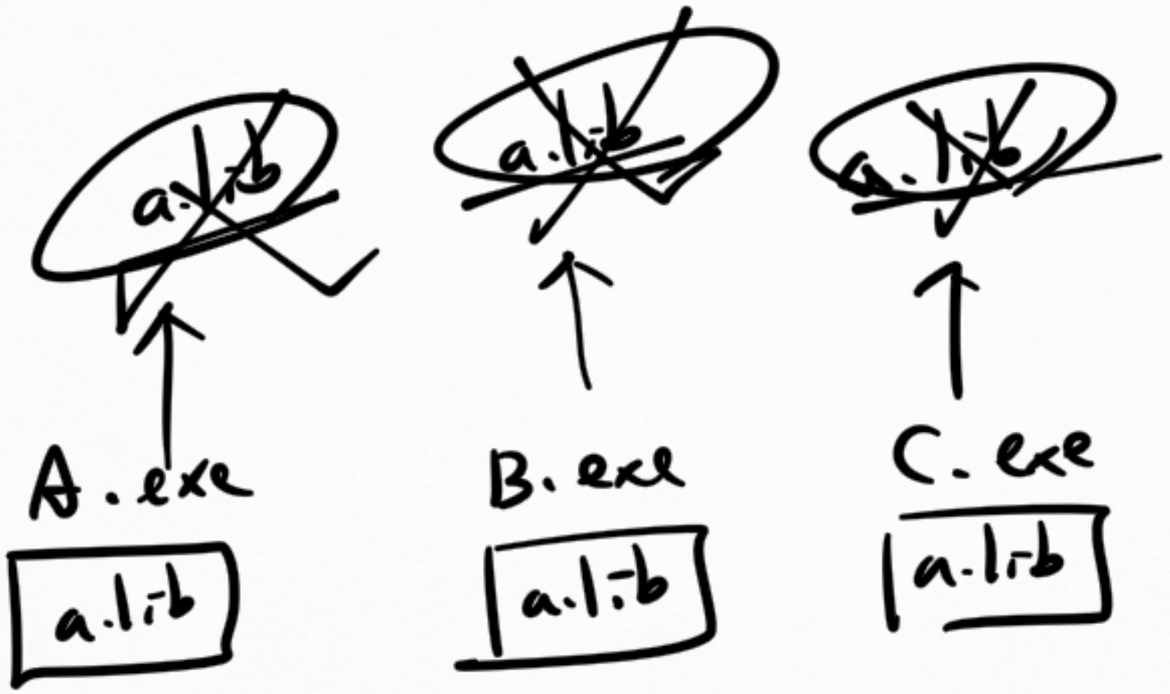
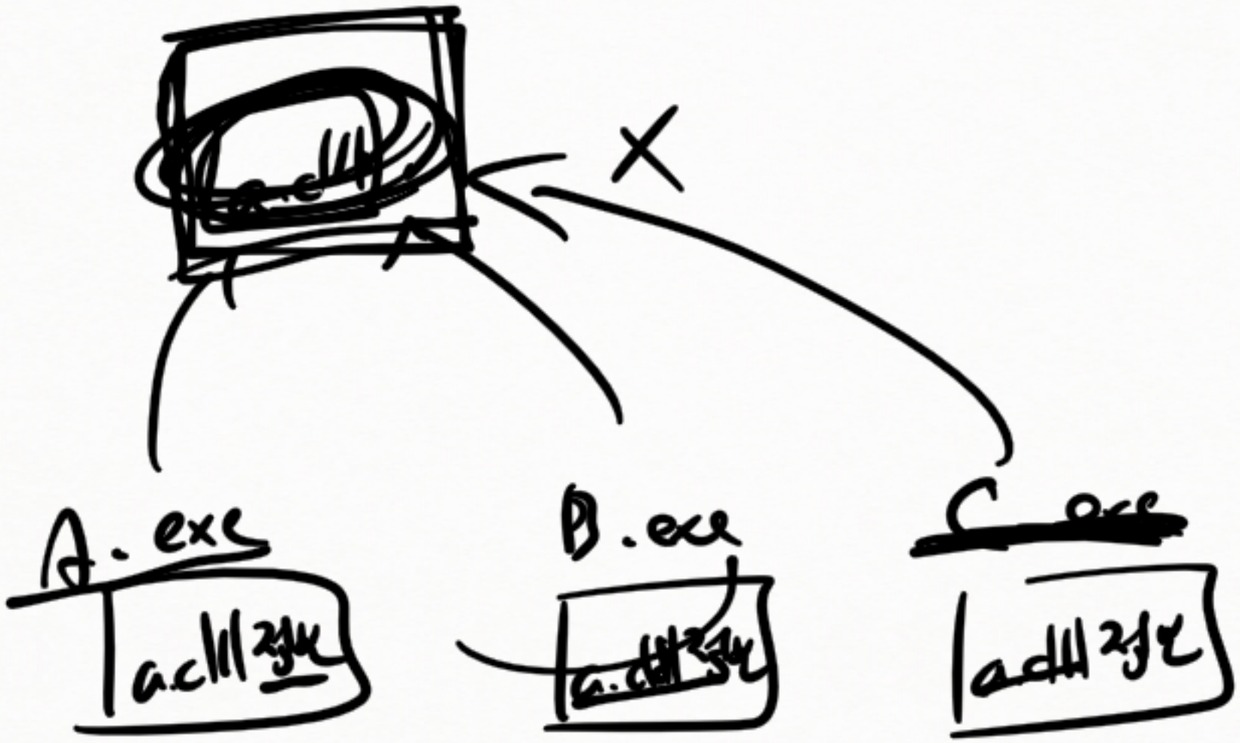

write()
FileOutputStream
File
read()
FileInputStream


App.
명령
PC
마우스
키보드
Printer
cpu
GPU
ATI
NVidia
VGACard
모니터
Scanner
cpu
cpu
SoundCard
스피커
마이크
LanCard
cpu
인터넷

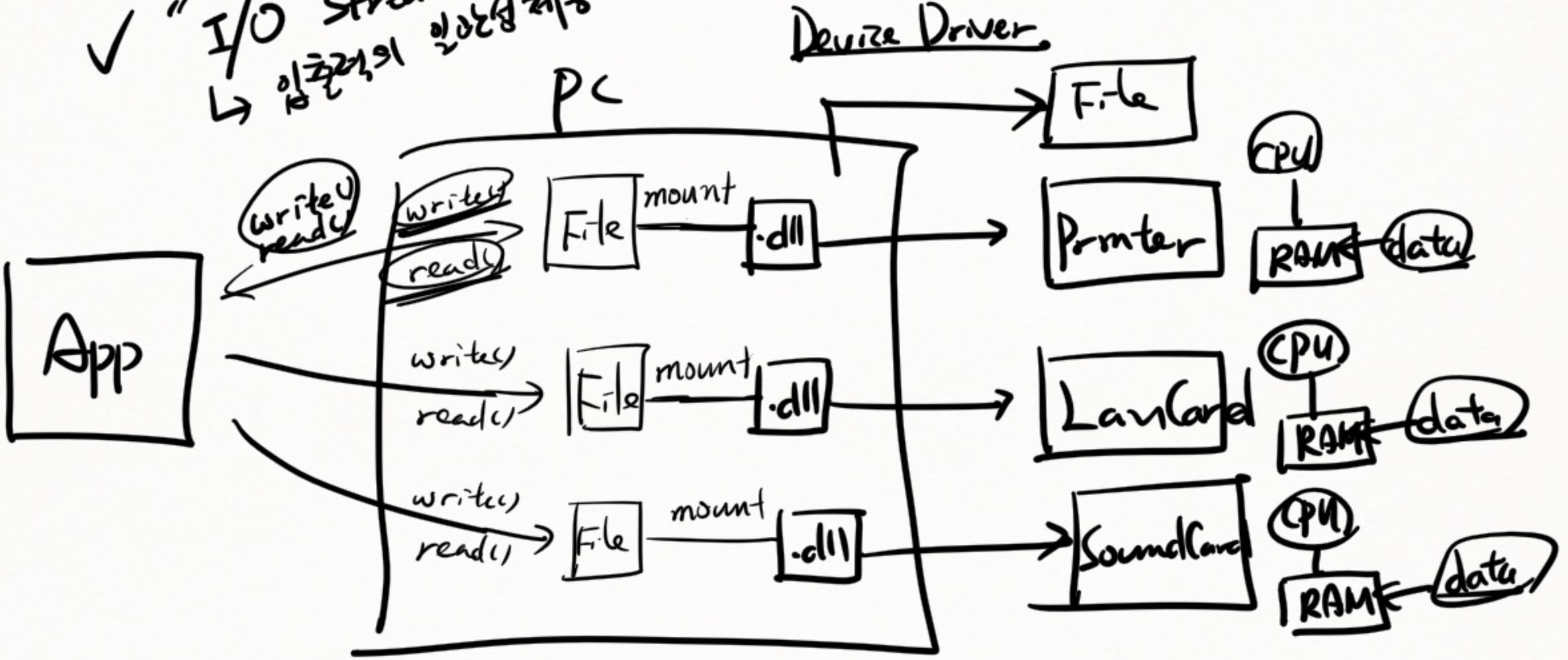

1983
PC
Windows 1.0,
3.0
3.1
90 초~중반
1970~1980년대
Windows OS
✓ GDI API 정의
C 언어
App.
일반적인 함수 호출
.dll
.dll
ATI VGA
NVidia VGA
CPU
RAM
명령어
데이터
Prnter
CPU
+
RAM
양방향 데이터
Graphic card 전용
Graphic "driver" 제공하는 S/W


Linux
Windows
*.lib
*.so
*.lib
*.dll
HP
.exe
App + *.lib
a.c
a.obj
compiling
+
.lib
Linking
*.exe
static linking Library
App
+
DLL 정보
.exe
dynamic linking Library


a.lib
a.lib
a.lib
A.exe
B.exe
C.exe
a.lib
a.lib
a.lib


X
A.exe
B.exe
C.exe

✓ "I/O Stream API"
↳ 입출력의 일관성 제공



Java I/O Stream API

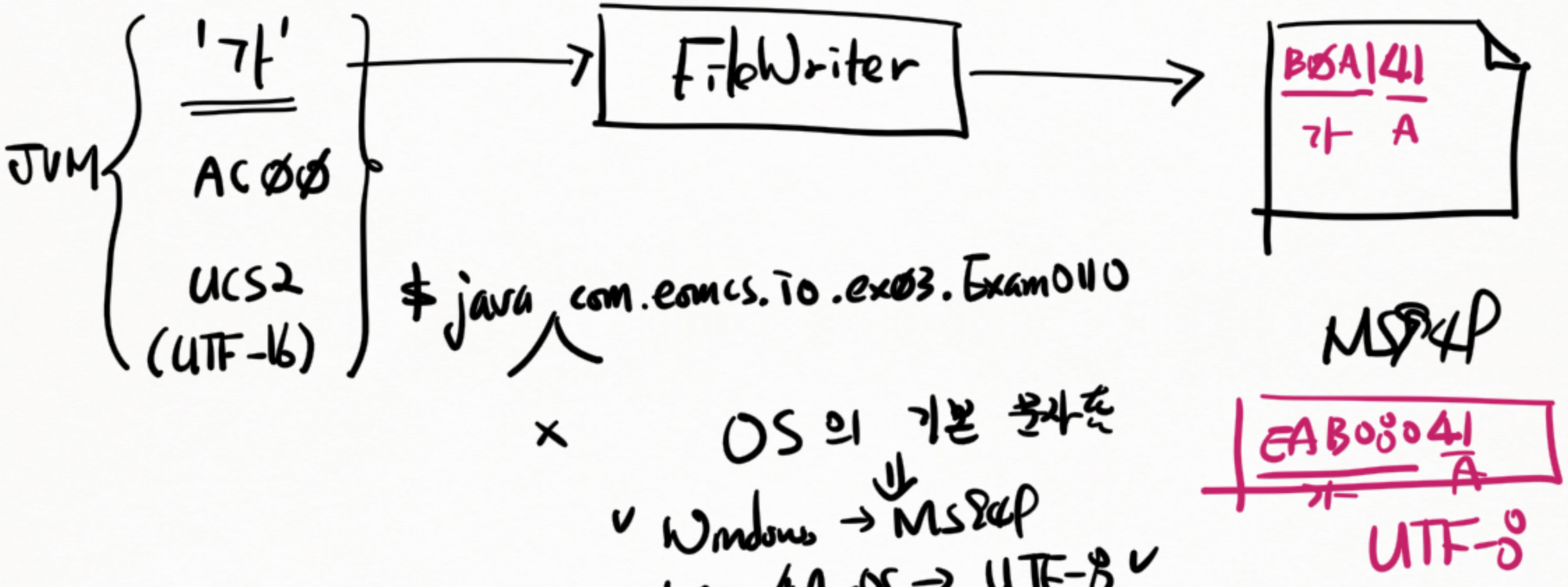
	byte stream	character stream
입력	InputStream	Reader
출력	OutputStream	Writer

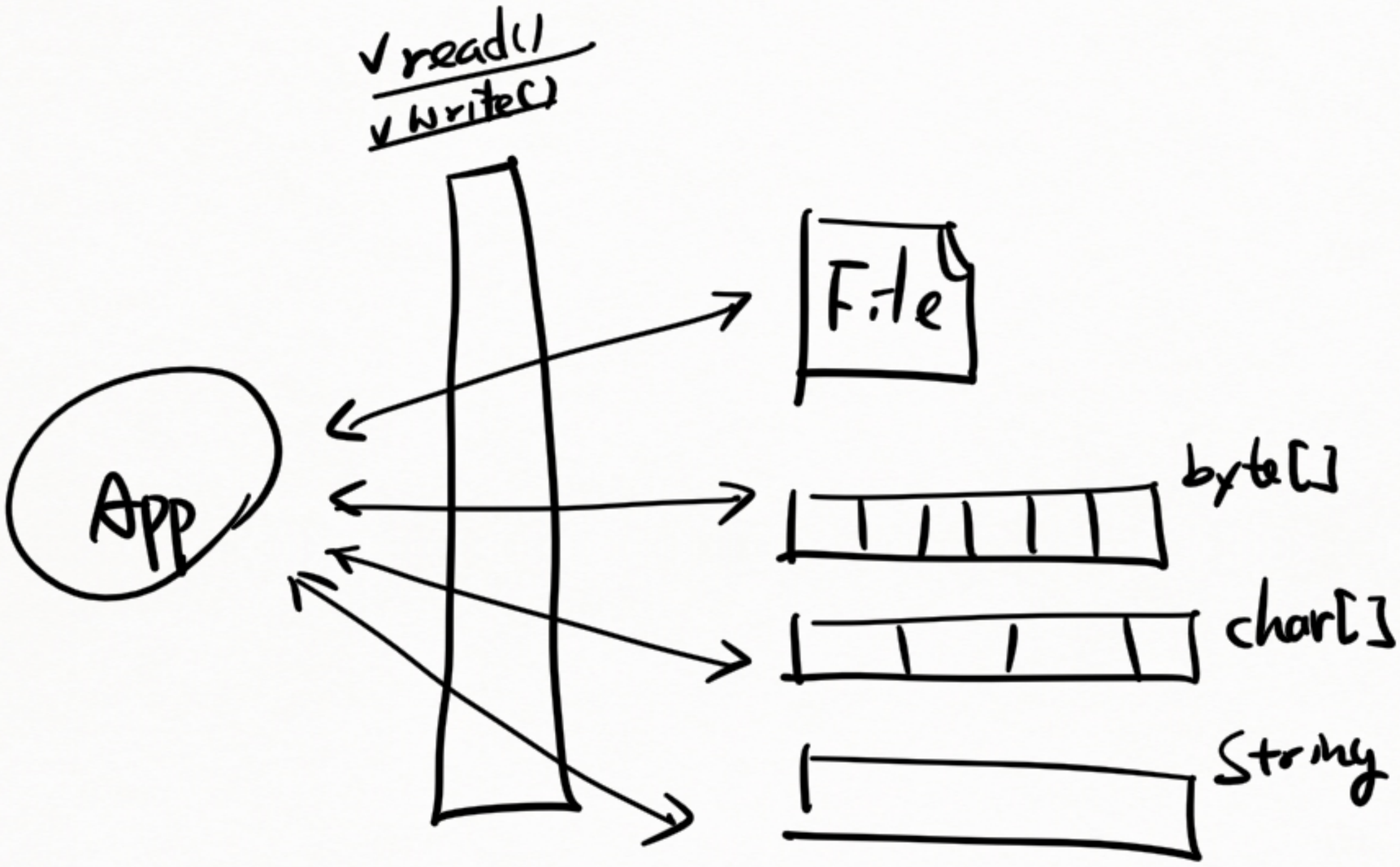
Data Sink Stream Class

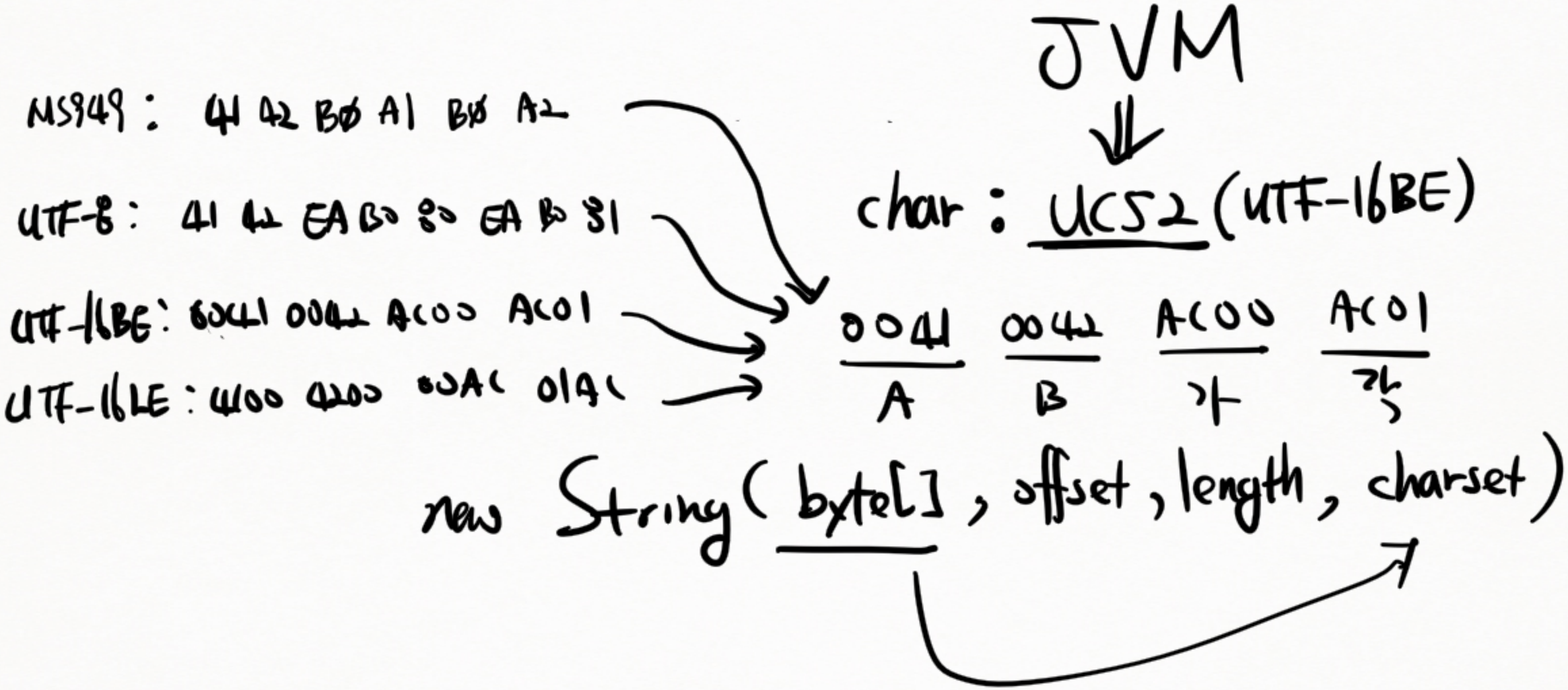
	Input	Output
File	FileInputStream FileReader	FileOutputStream FileWriter
Memory	ByteArrayInputStream CharArrayReader StringReader	ByteArrayOutputStream CharArrayWriter StringWriter
~~(다른 프로그램)~~ Process	PipedInputStream PipedReader	PippedOutputStream PippedWriter

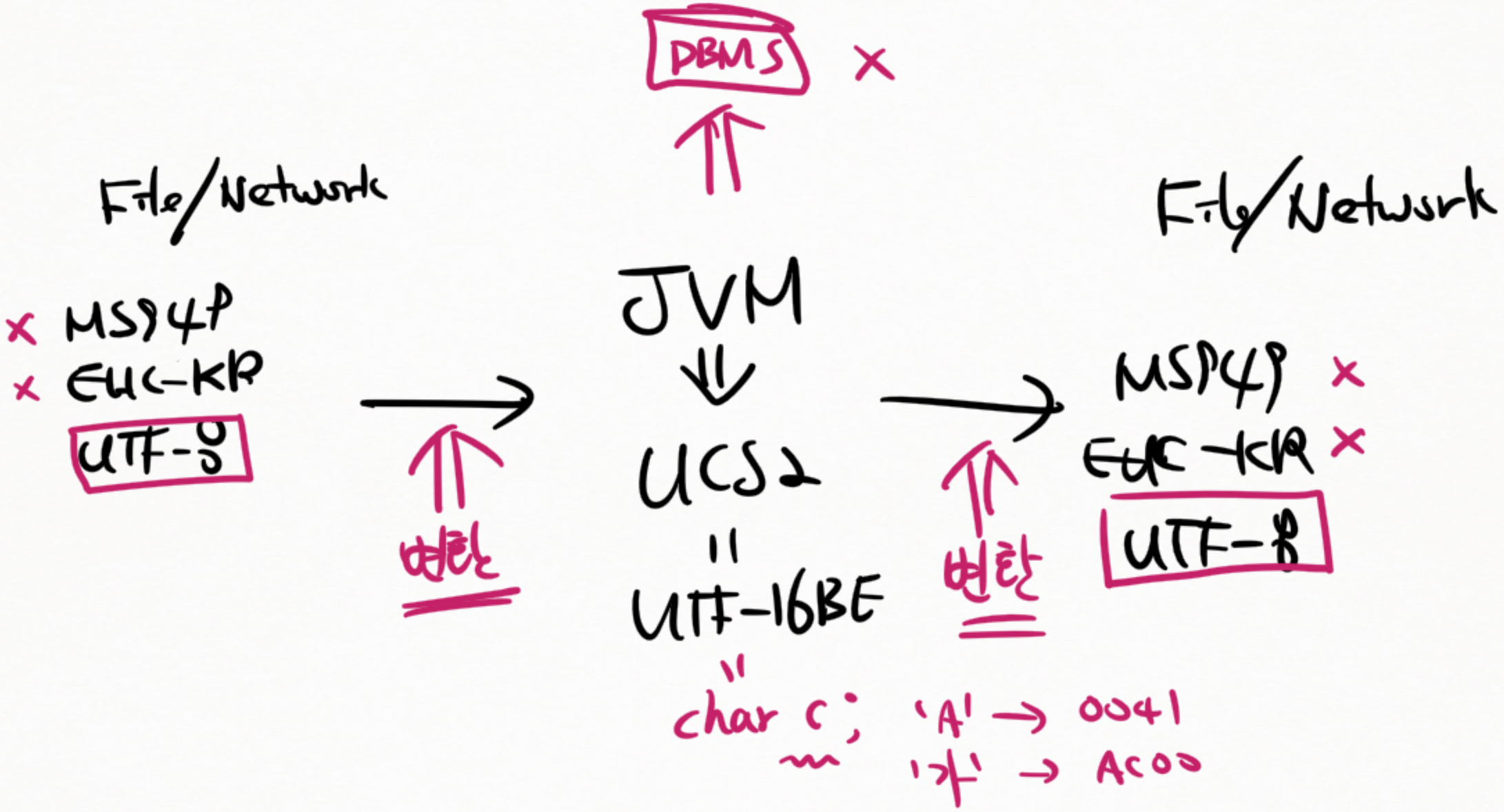
ex) .grf .jpg .ppt .avi .pdf

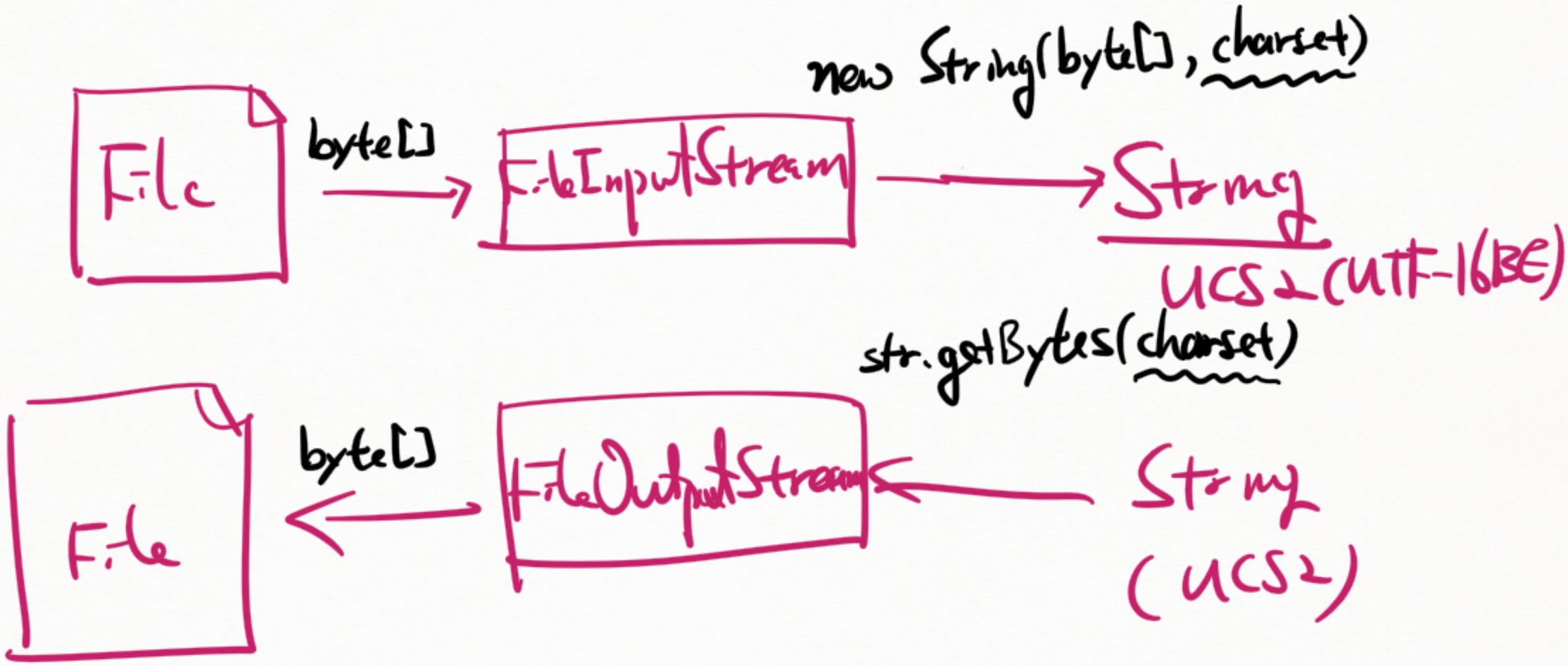
ex) .csv .html .java .js .css .txt

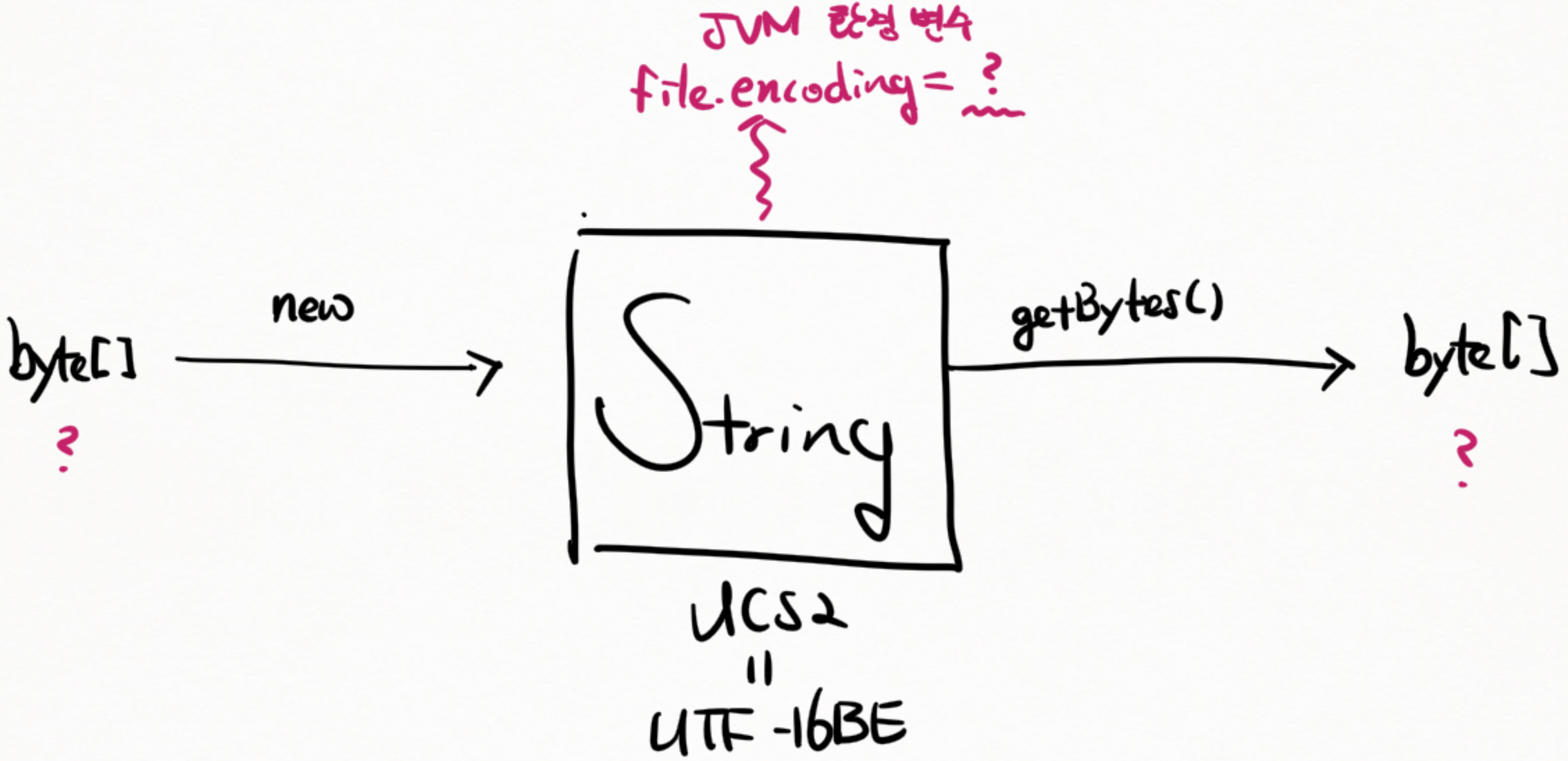

JVM
'가'
AC00
UCS2
(UTF-16)
FileWriter
B0A1 41
가 A
$ java com.eomcs.io.ex03.Exam0110
MS949
EAB0 0041
가 A
UTF-8
× OS의 기본 문자셋
Windows → MS949
Linux/MacOS → UTF-8


read()
write()
App
File
byte[]
char[]
String


JVM
MS949 : 41 42 B0 A1 B0 A2
UTF-8 : 41 42 EA B0 80 EA B0 81
UTF-16BE : 0041 0042 AC00 AC01
UTF-16LE : 4100 4200 00AC 01AC
char : UCS2 (UTF-16BE)
0041 0042 AC00 AC01
A B 가 각
new String(byte[], offset, length, charset)


DBMS
File/Network
MS949
EUC-KR
UTF-8
변환
JVM
UCS2
=
UTF-16BE
변환
MS949
EUC-KR
UTF-8
File/Network
char c;
'A' → 0041
'가' → AC00


new String(byte[], charset)
File
byte[]
FileInputStream
String
UCS2(UTF-16BE)
str.getBytes(charset)
File
byte[]
FileOutputStream
String
(UCS2)


JVM 환경 변수
file.encoding = ?
byte[]
?
new
String
getBytes()
byte[]
?
UCS2
=
UTF-16BE

E ⇒ 3017-3030 ⇒ 끝나 ?

2xxx 중식

34xx-30xx 한식

4444x-30xx-30xx 100만원

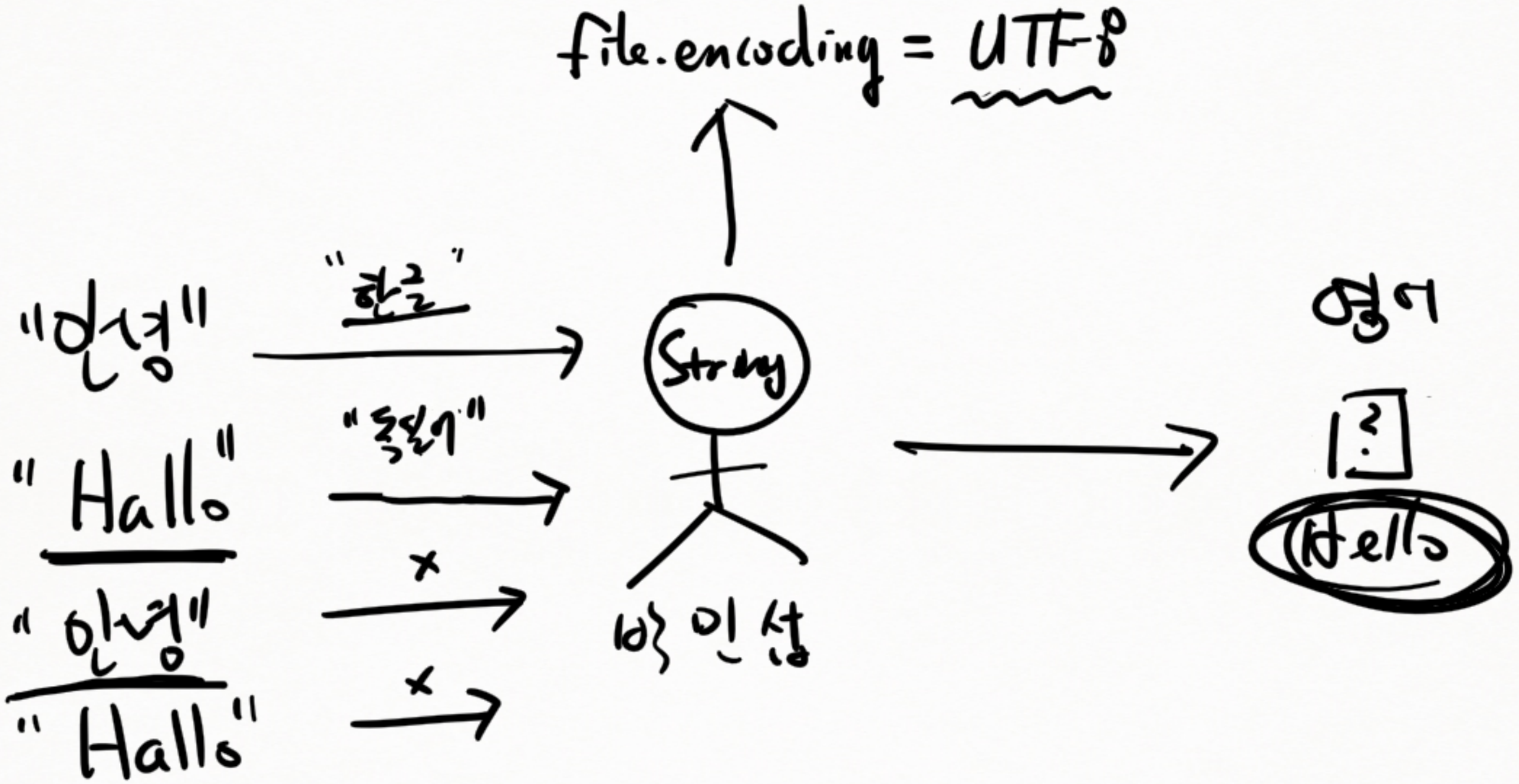
9999 +30xx -32xx-30xx 1개

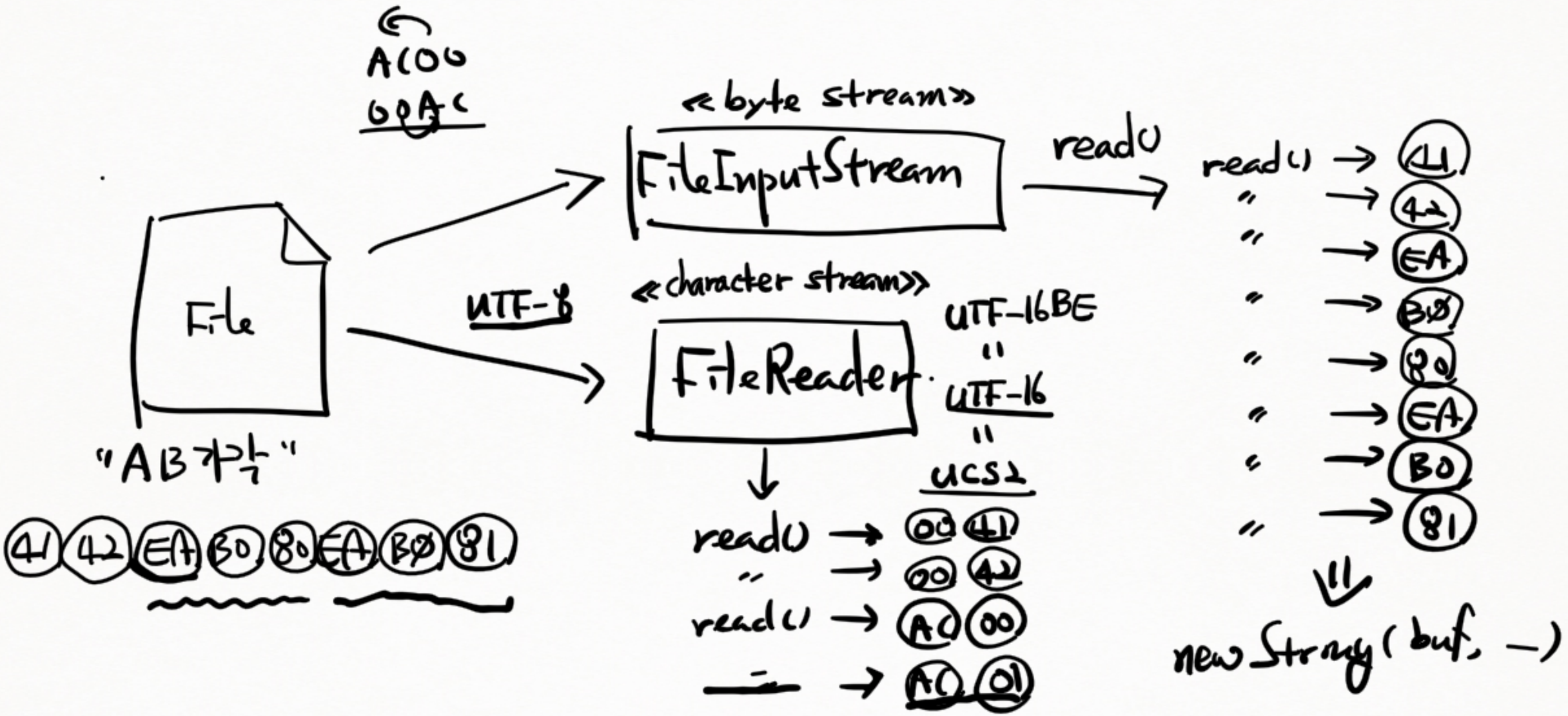
✓ A ⇒ 2234

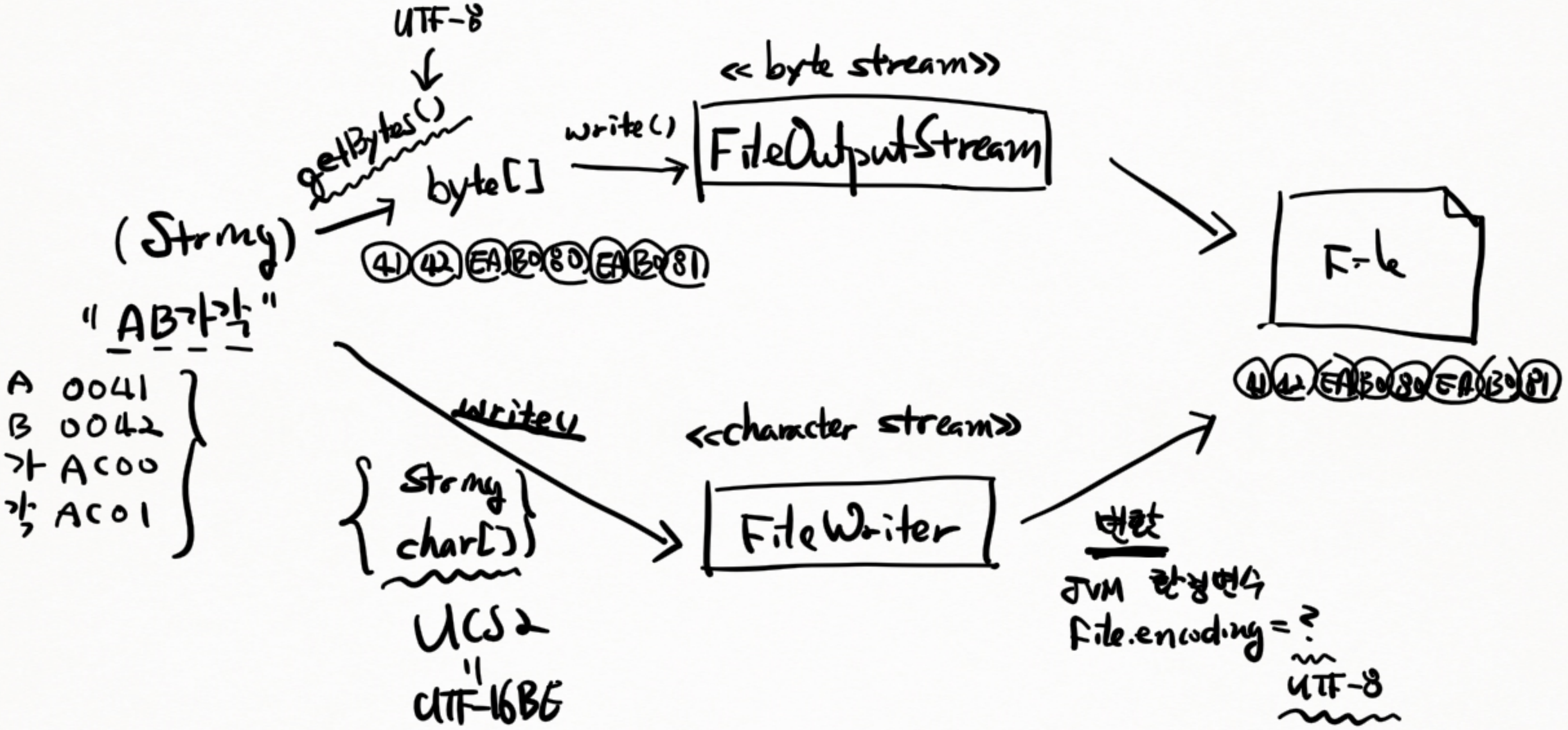
✓ B ⇒ 3478-3012

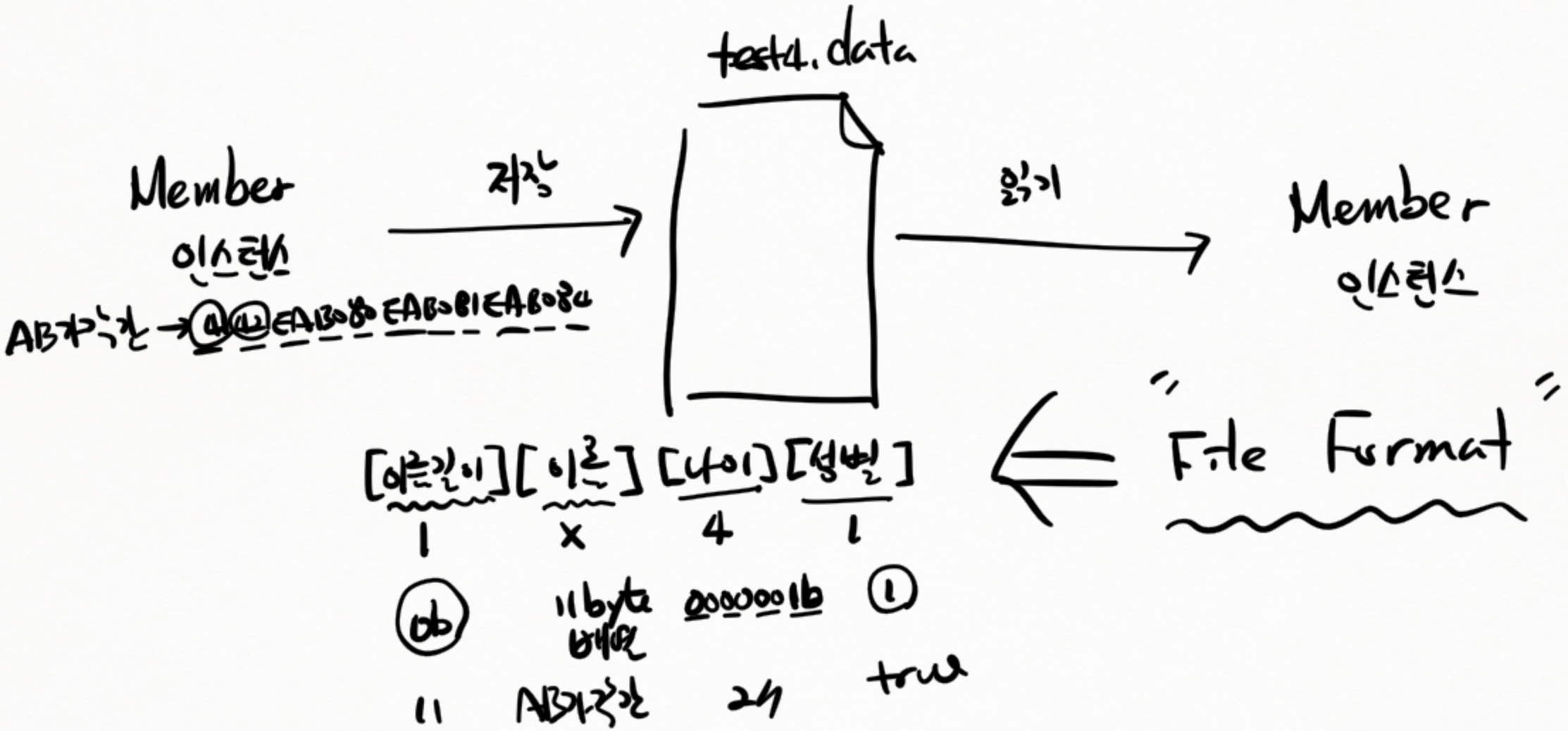
✓ C ⇒ 4444-3011-3033

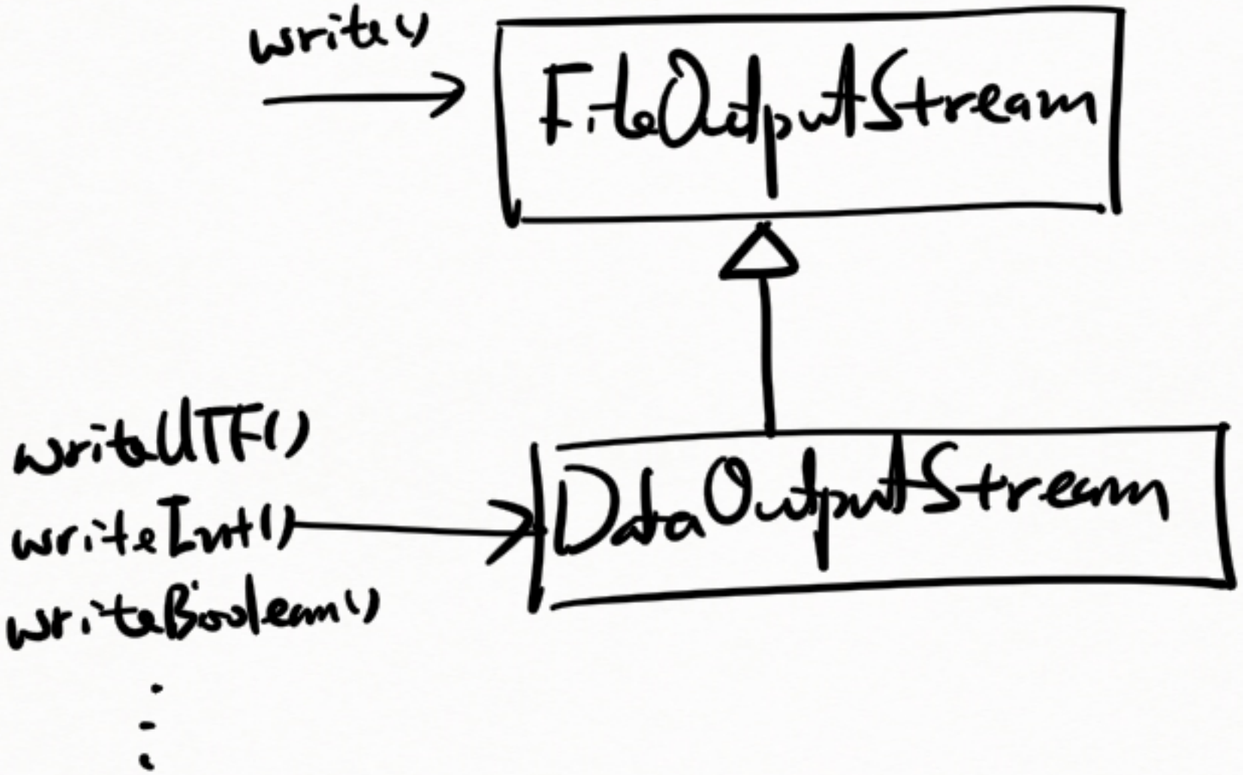
✓ D ⇒ 9999-3017-3211-3088


file.encoding = UTF-8
"안녕"
"한글"
"Hallo"
"동일"
"안녕"
x
"Hallo"
x
String
바이너리
출력
Hello


AC00
00AC
<<byte stream>>
FileInputStream
read()
read() → 41
" → 42
" → EA
" → B0
" → 80
" → EA
" → B0
" → 81
File
"AB가나"
41 42 EA B0 80 EA B0 81
UTF-8
<<character stream>>
FileReader
UTF-16BE
"
UTF-16
"
UCS2
read() → 00 41
" → 00 42
read() → AC 00
— → AC 01
new String(buf, —)


UTF-8
getBytes()
<< byte stream >>
write()
FileOutputStream
byte[]
(String)
"AB가각"
41 42 EA B0 80 EA B0 81
A 0041
B 0042
가 AC00
각 AC01
File
41 42 EA B0 80 EA B0 81
write()
<<character stream>>
String
char[]
FileWriter
UCS2
"
UTF-16BE
변환
JVM 환경변수
file.encoding = ?
UTF-8

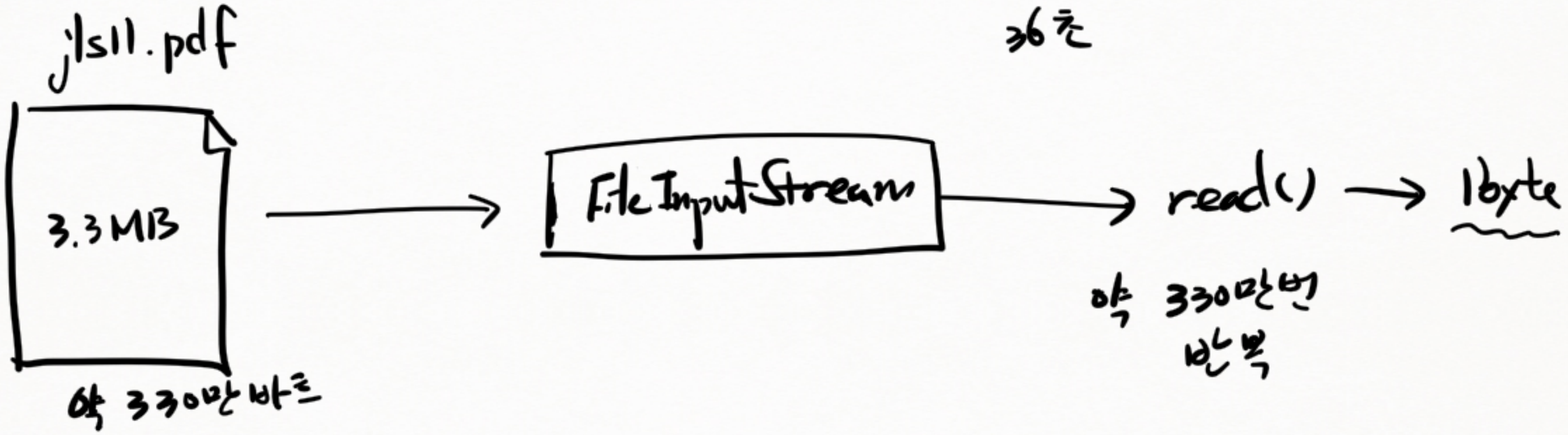

test4.data
Member
인스턴스
저장
읽기
Member
인스턴스
AB차장한
[이름길이] [이름] [나이] [성별]
1 x 4 1
0b
11byte
배열
00000016
1
true
11
AB차장한
24
File Format

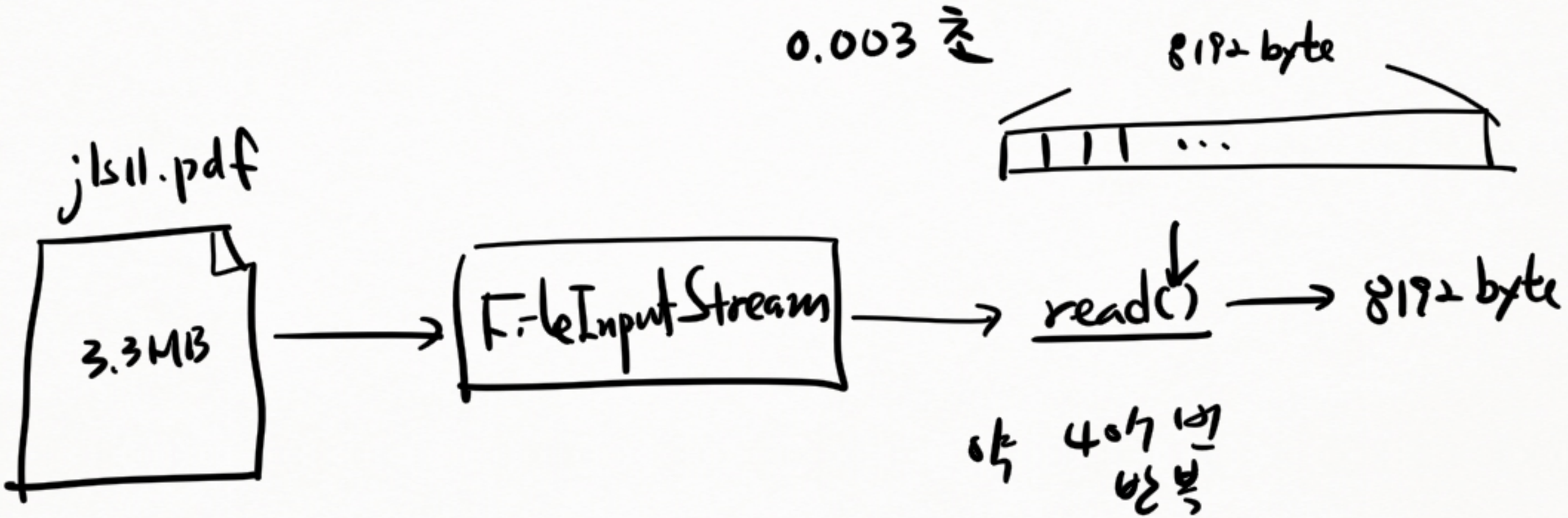

write()
FileOutputStream
writeUTF()
writeInt()
writeBoolean()
DataOutputStream

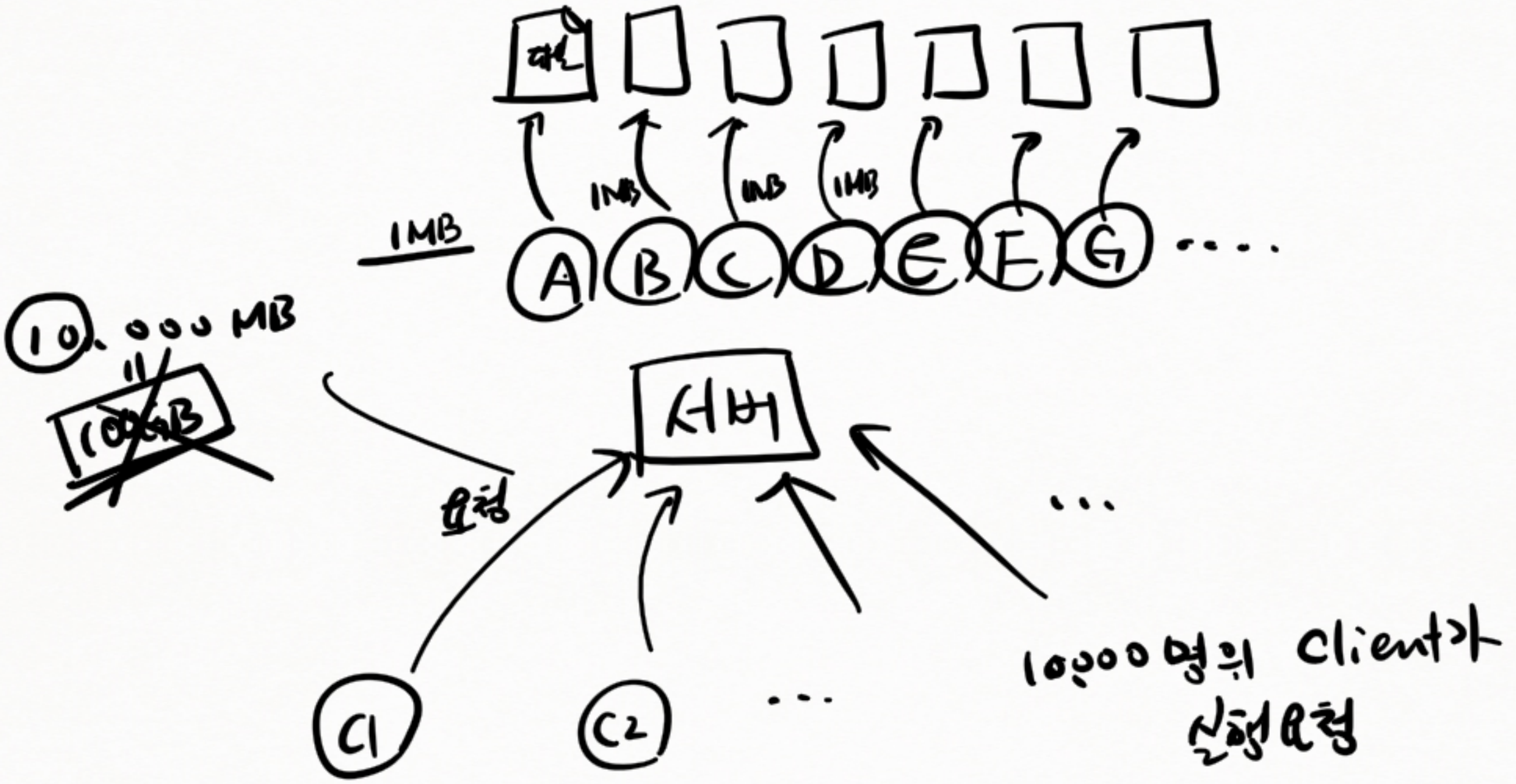
3.3초

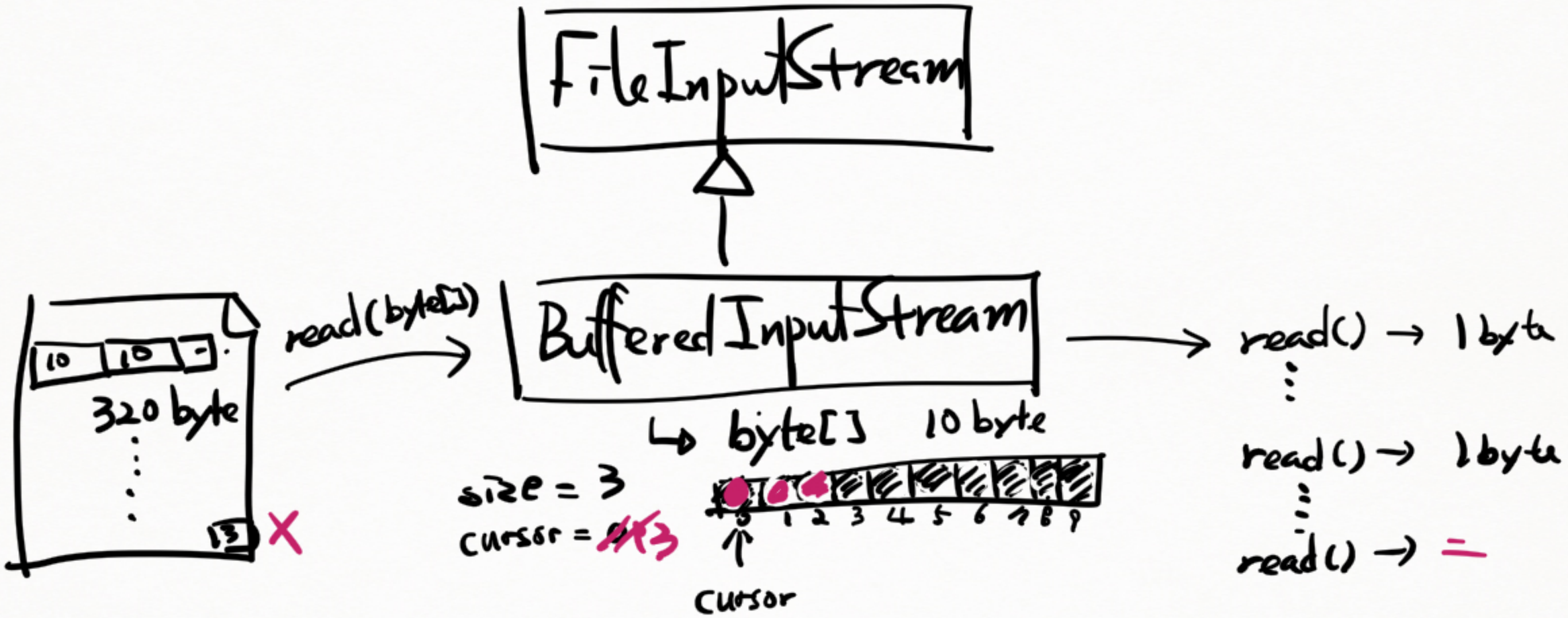
0.8초

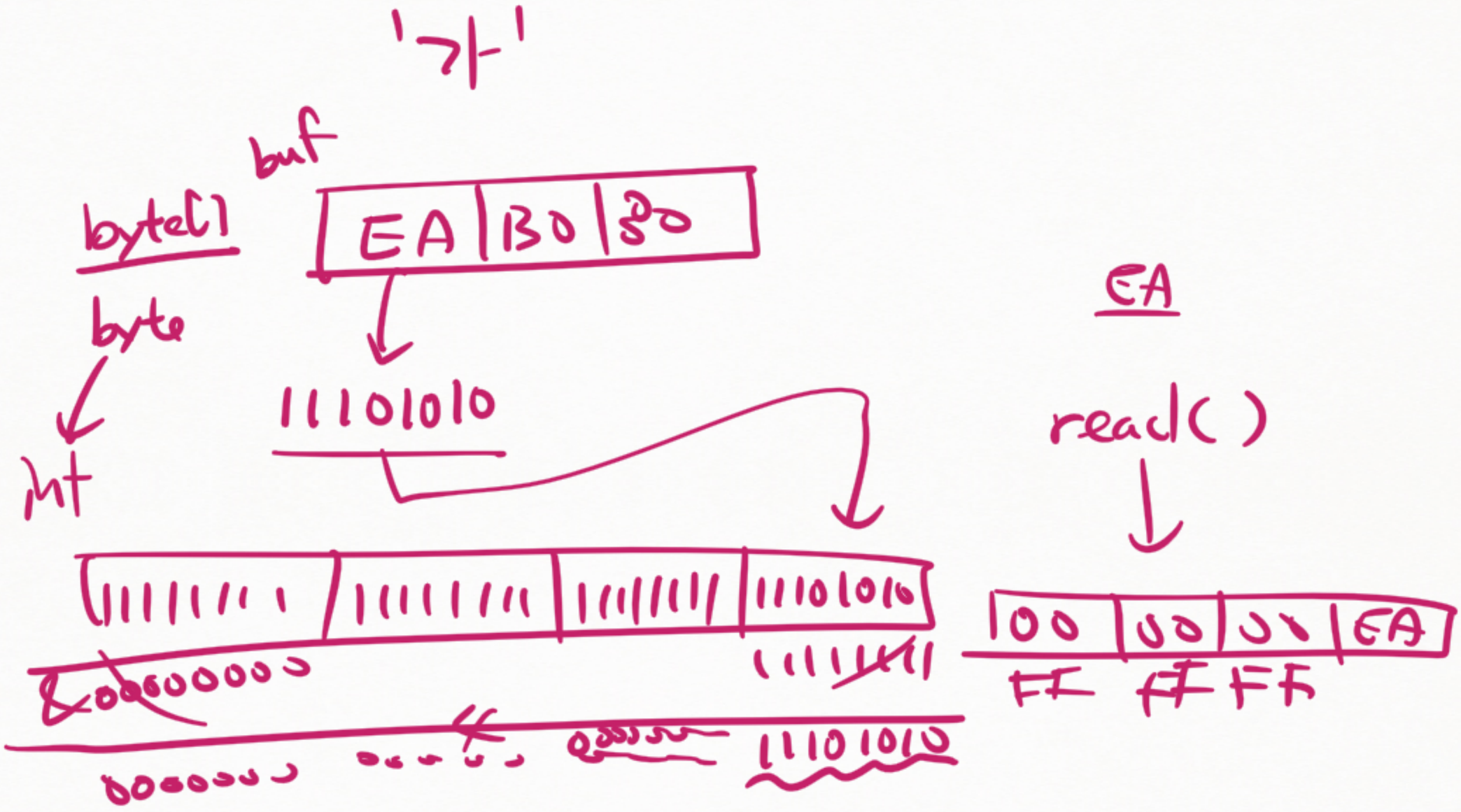
36초

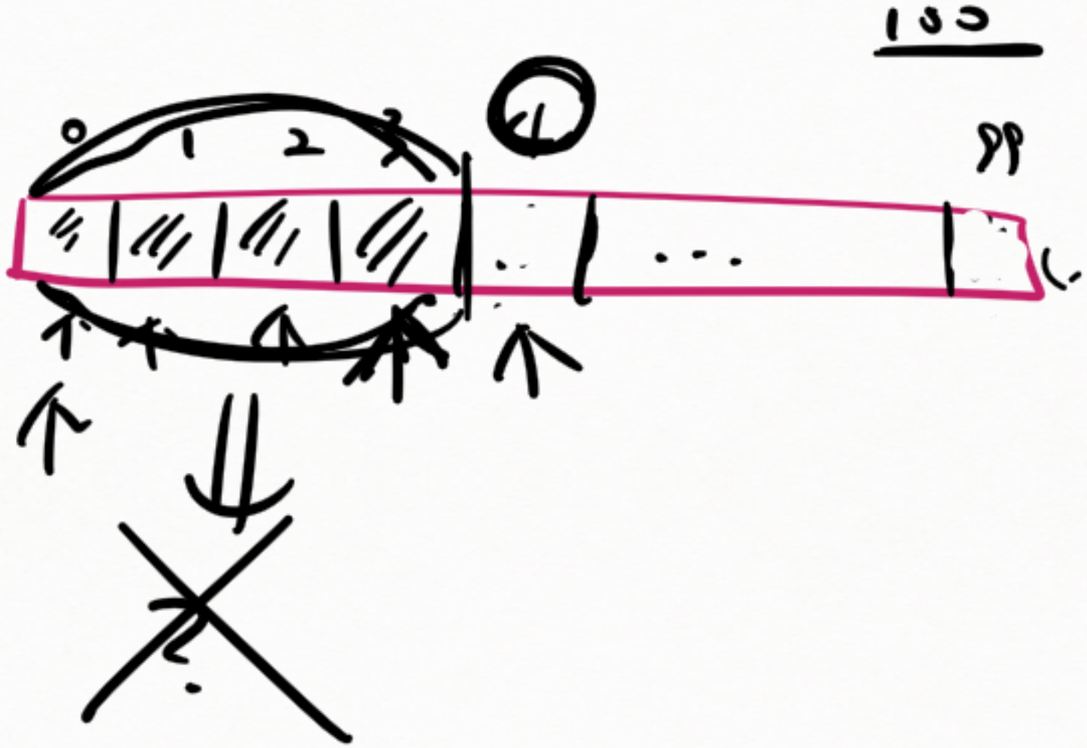

jlsll.pdf
3.3MB
약 330만 바이트
FileInputStream
read()
1byte
약 330만번
반복

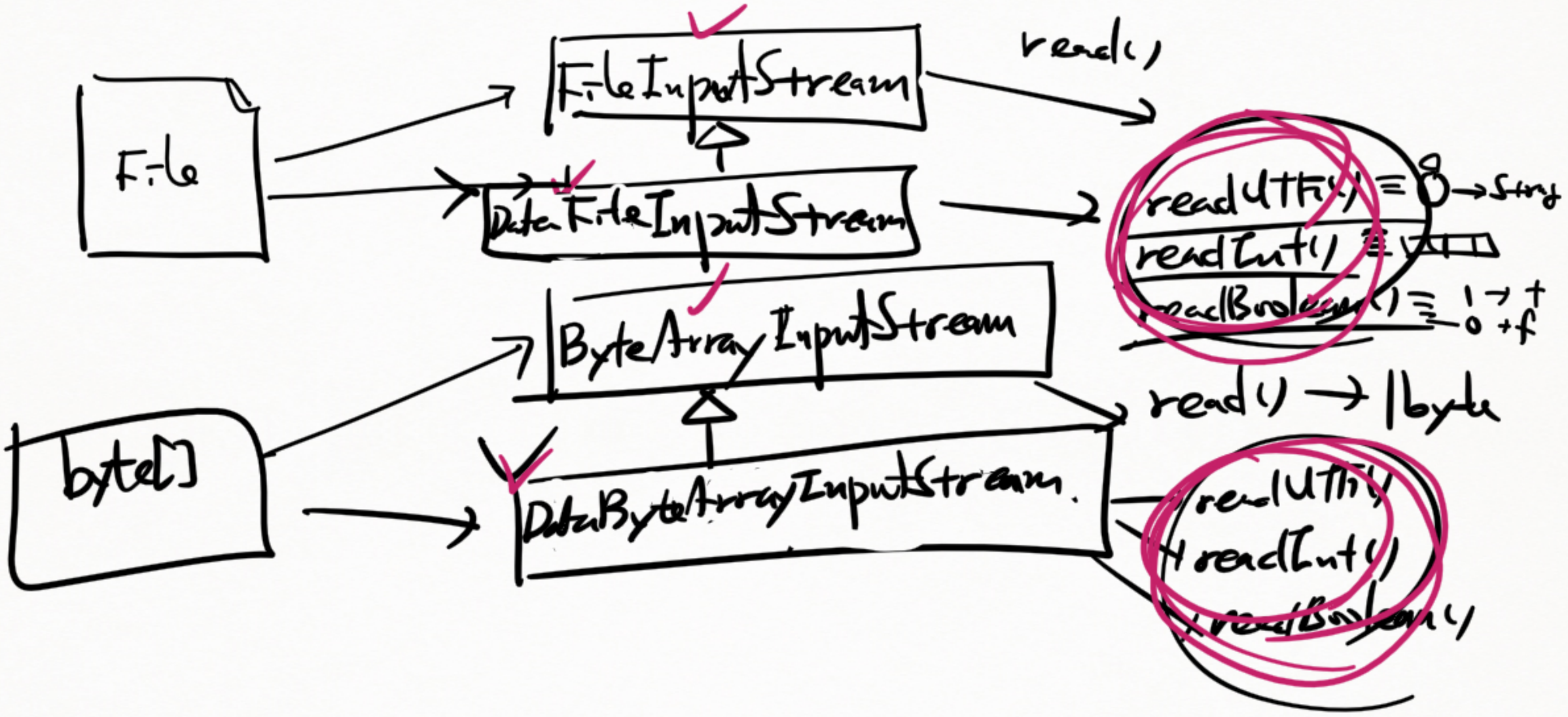

0.003 초
8192 byte
jls11.pdf
3.3MB
FileInputStream
read()
8192 byte
약 407번 반복

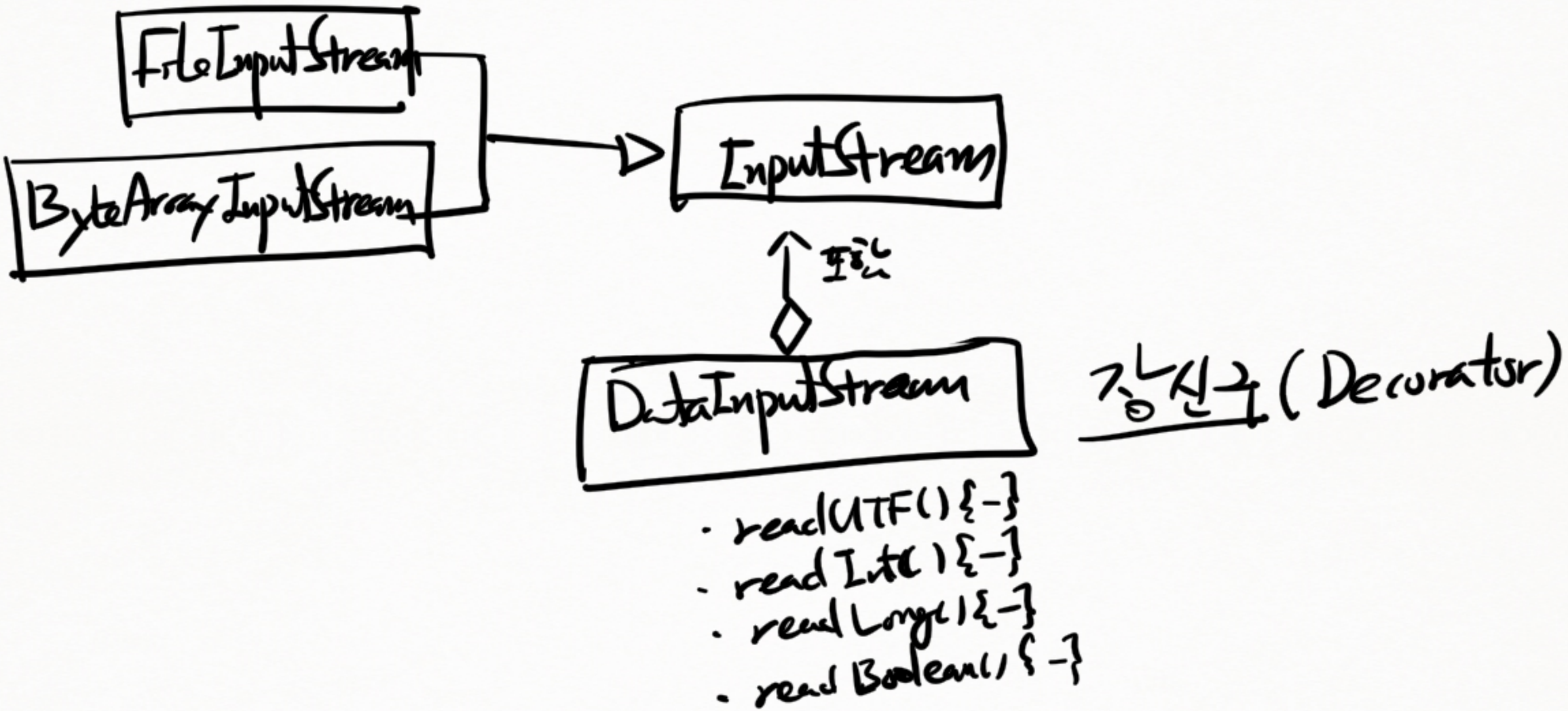

대기
1MB
1MB
1MB
1MB
A
B
C
D
E
F
G
....
10,000 MB
100MB
요청
서버
C1
C2
...
...
10,000명의 client가
실행요청

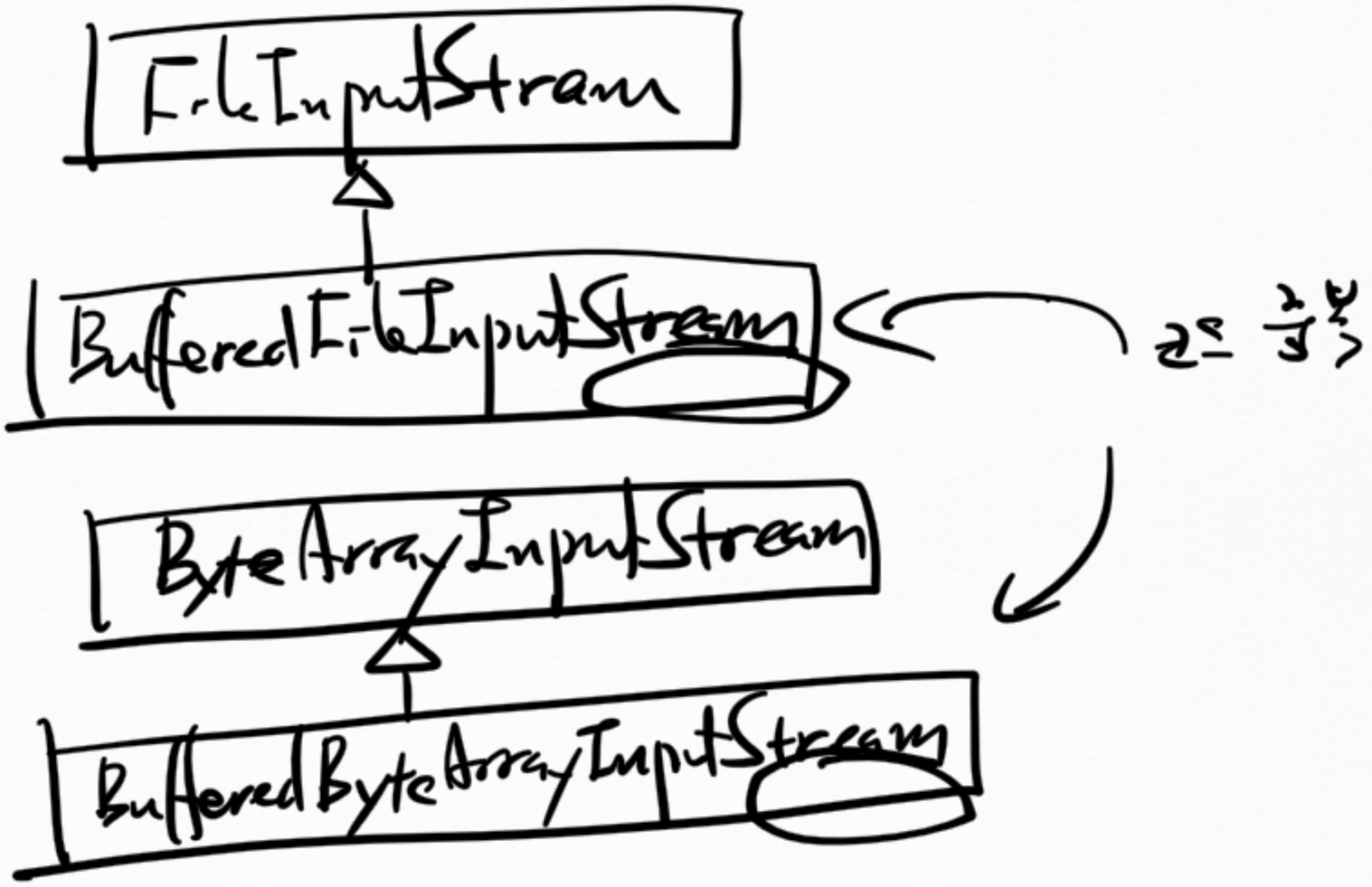

FileInputStream
BufferedInputStream
read(byte[])
10
10
-
320 byte
13
X
byte[] 10 byte
size = 3
cursor = 0 1 2 3
0 1 2 3 4 5 6 7 8 9
cursor
read() → 1 byte
read() → 1 byte
read() → -

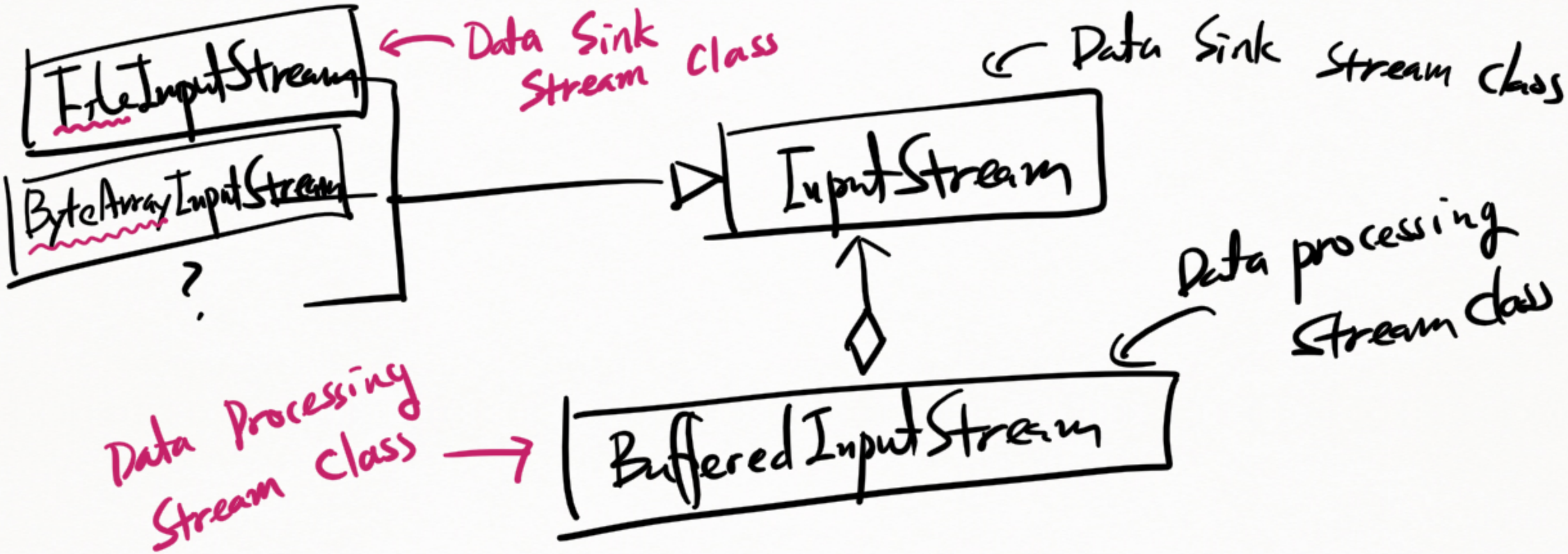

buf
byte[]
EA
B0
80
byte
11101010
int
EA
read()
00
00
01
EA
FF
FF
FF


0
1
2
3
4

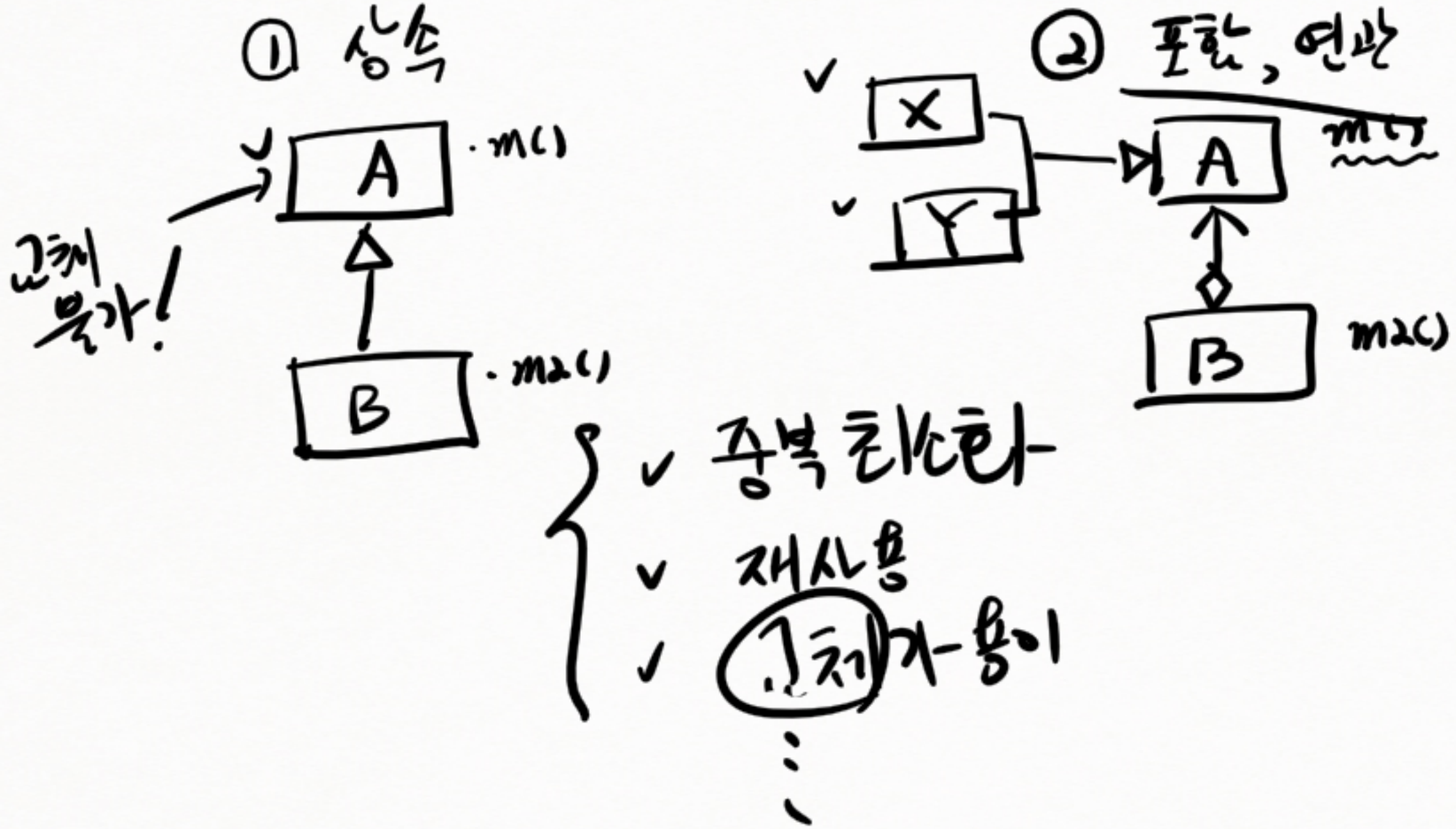

File
FileInputStream
read()
DataFileInputStream
readUTF() → String
readInt()
readBoolean() ≡ 1 → t
0 → f
ByteArrayInputStream
read() → byte
byte[]
DataByteArrayInputStream
readUTF()
readInt()
readBoolean()

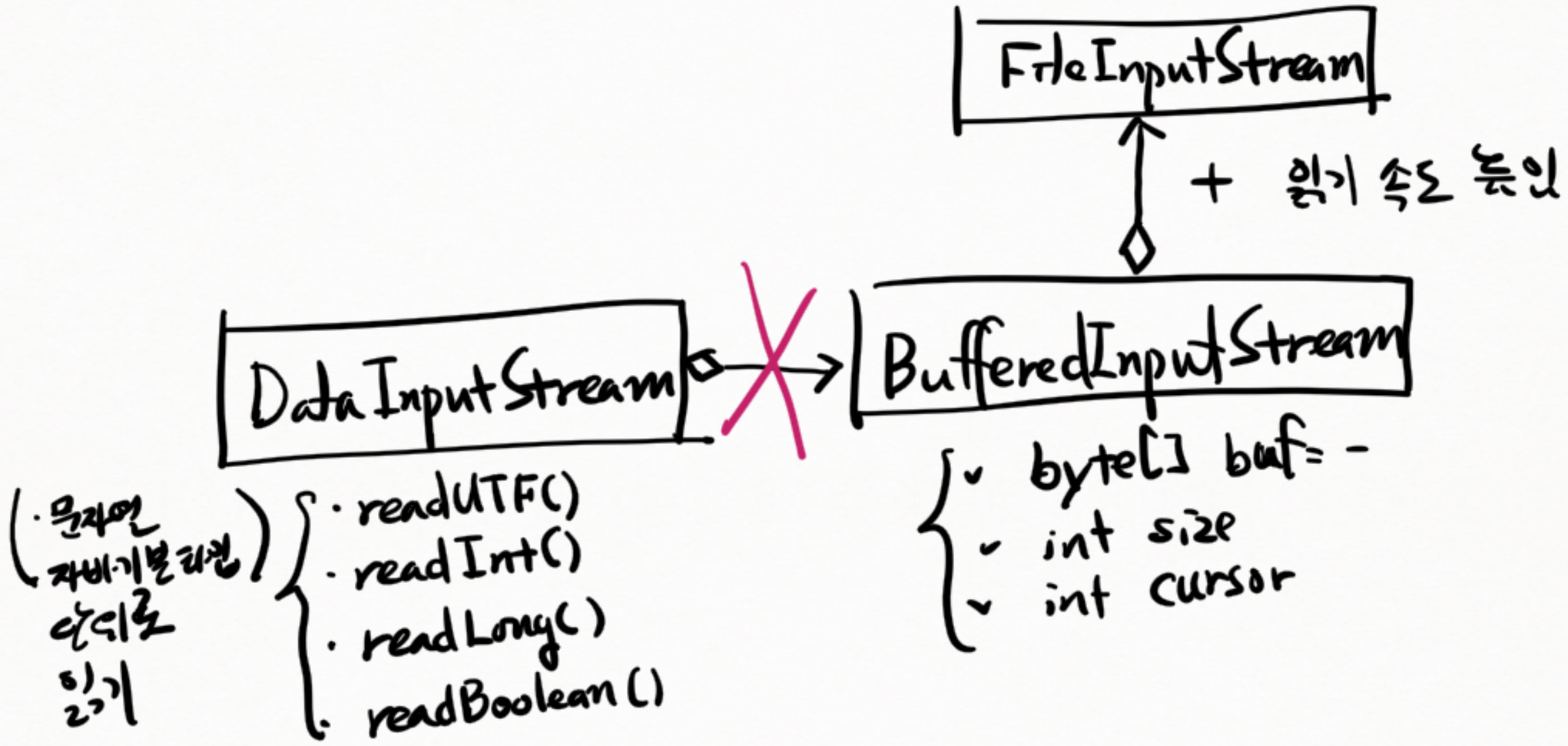

FileInputStream
ByteArrayInputStream
InputStream
포함
DataInputStream
기능 추가 (Decorator)
. readUTF() {-}
. readInt() {-}
. readLong() {-}
. readBoolean() {-}

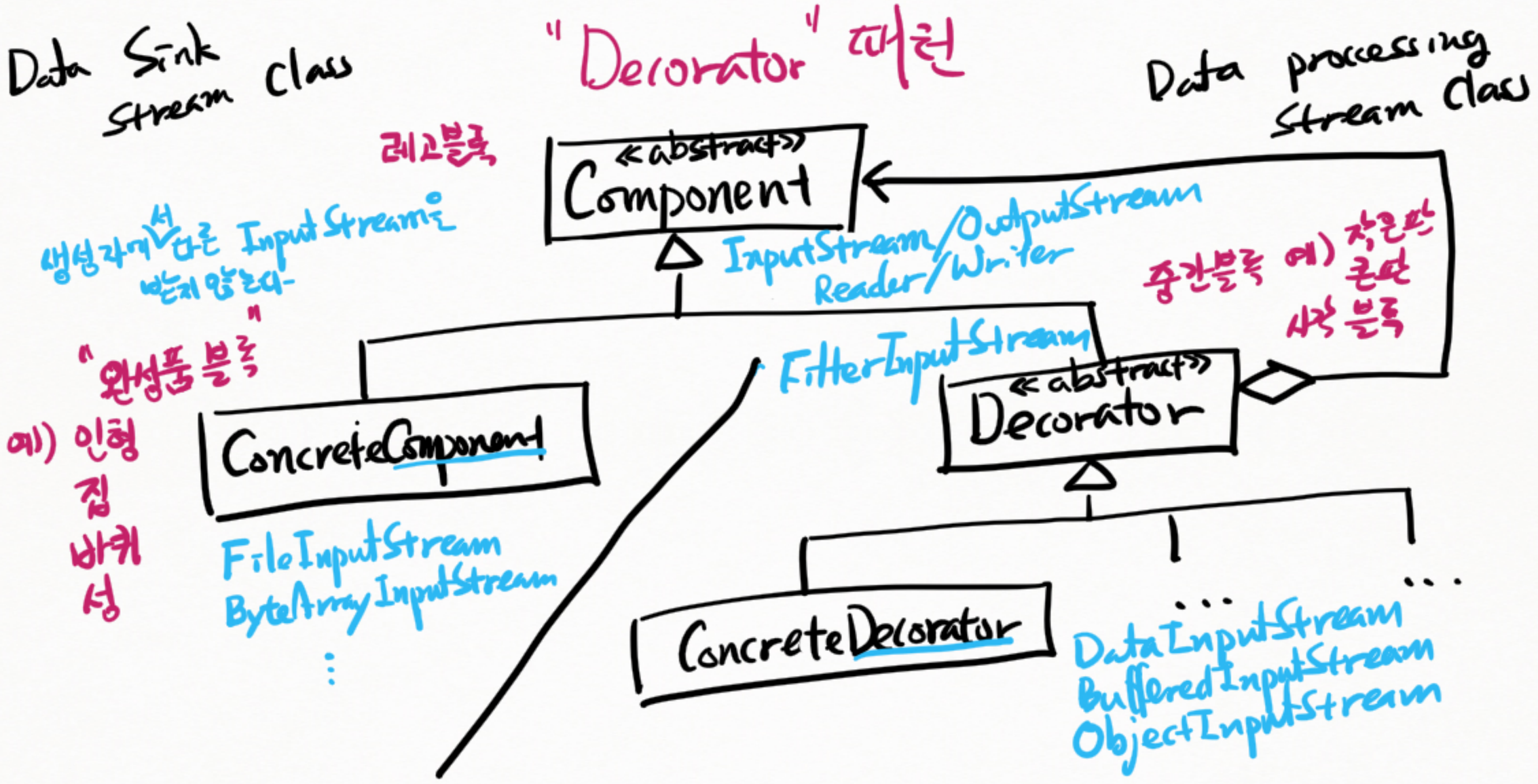

FileInputStream
BufferedFileInputStream
ByteArrayInputStream
BufferedByteArrayInputStream


FileInputStream
ByteArrayInputStream
?
Data Sink Stream Class
Data Sink Stream Class
InputStream
Data processing Stream class
Data Processing Stream Class
BufferedInputStream

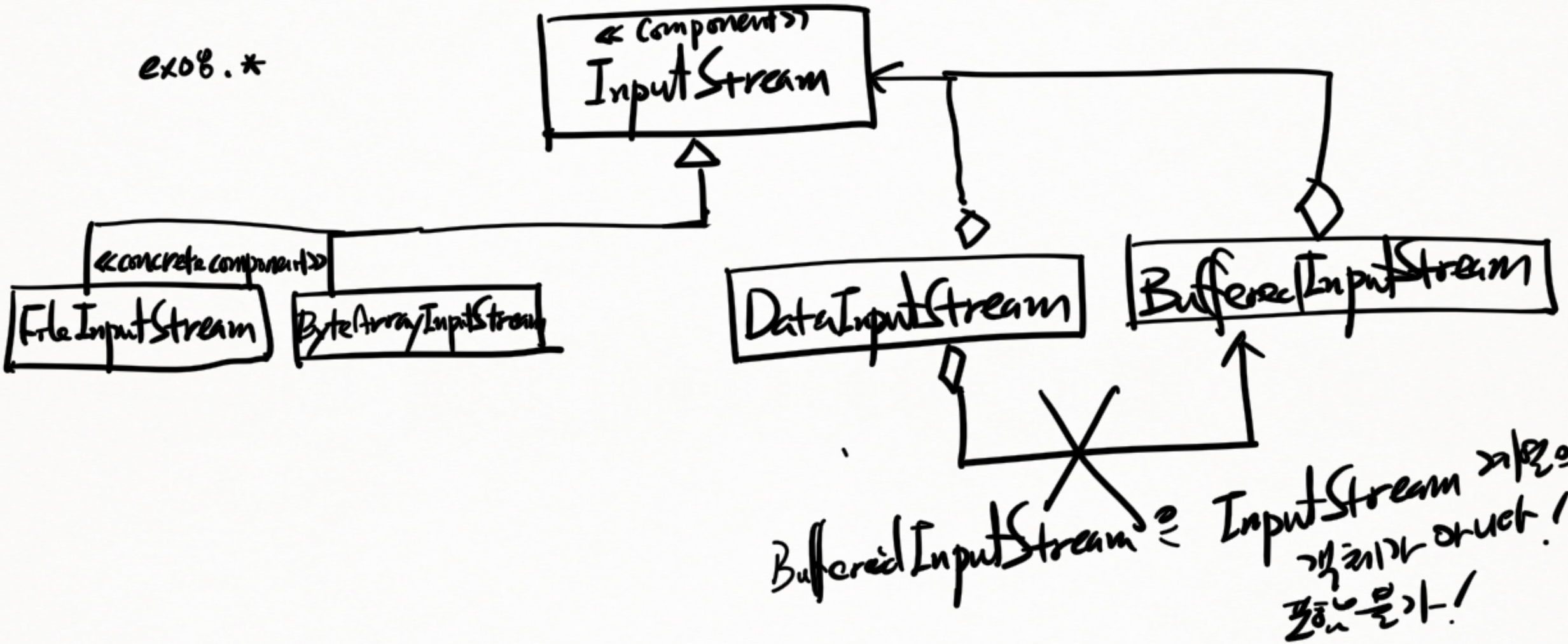
기능확장

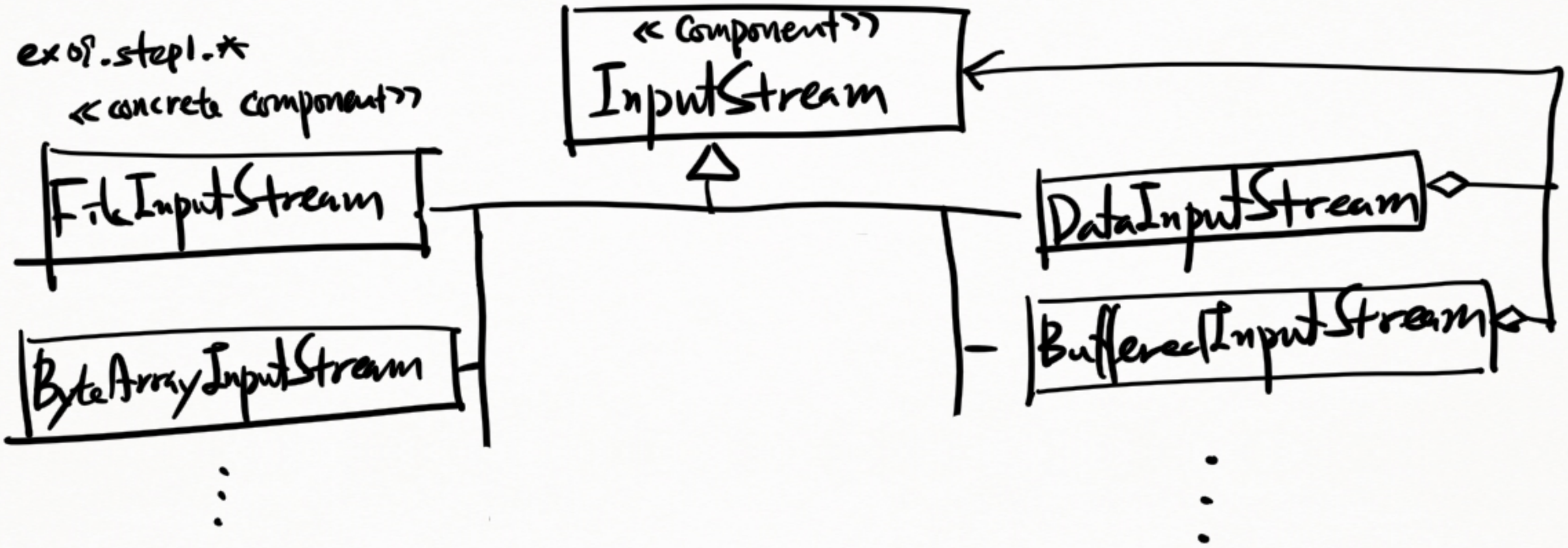


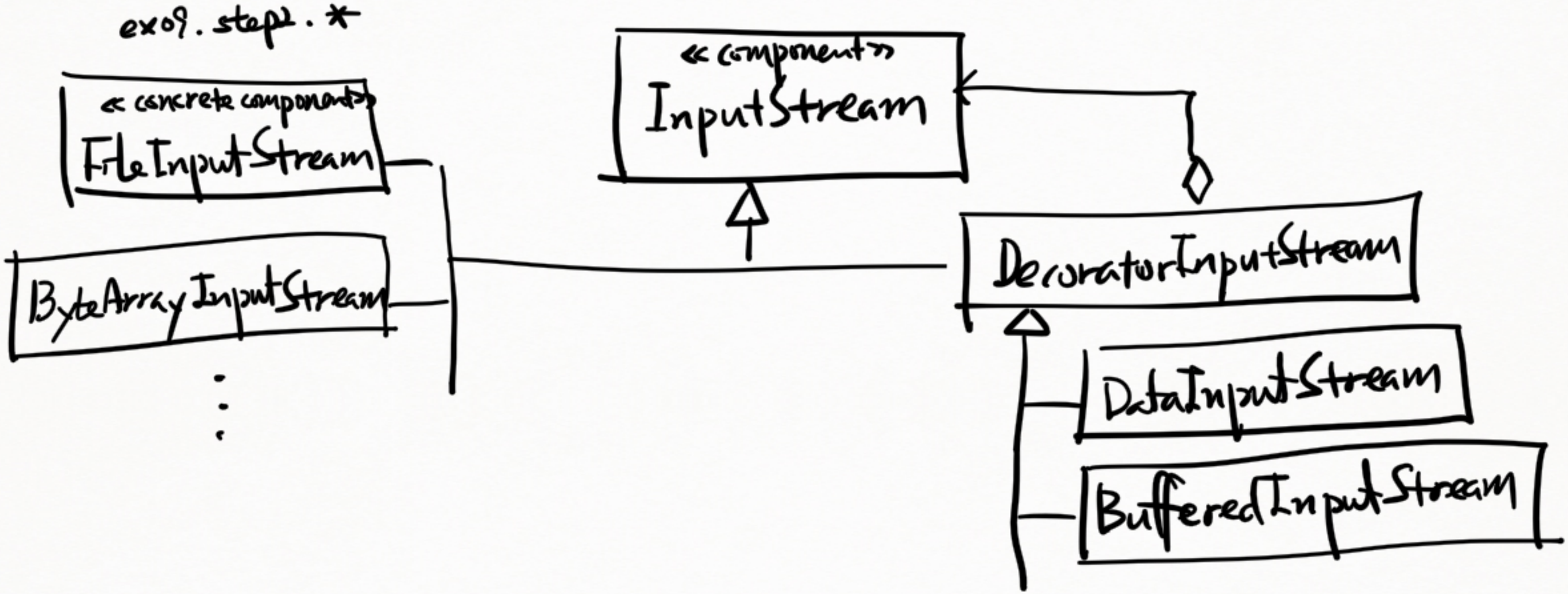

FileInputStream
+ 읽기 속도 높임
DataInputStream
BufferedInputStream
· 문자열
자바기본타입
단위로
읽기
· readUTF()
· readInt()
· readLong()
· readBoolean()
byte[] buf = -
int size
int cursor

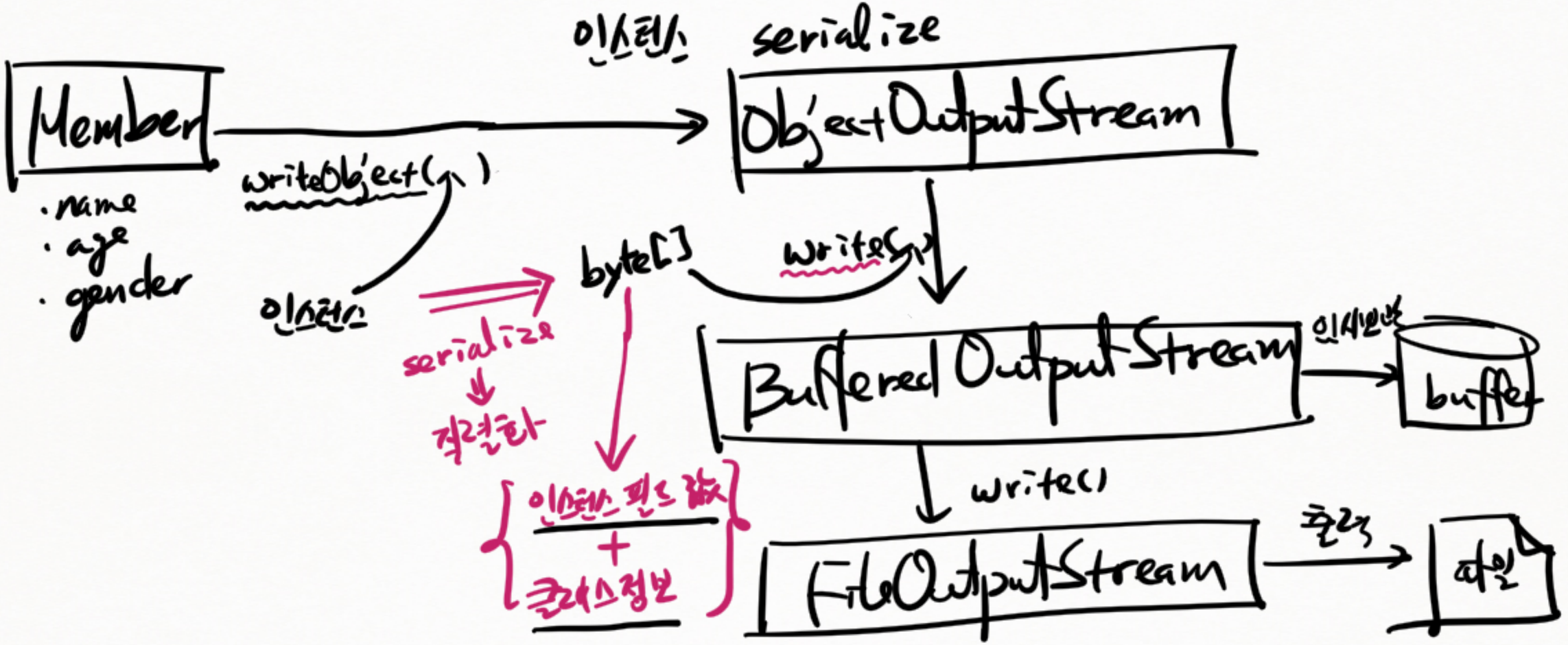

Data Sink Stream class
"Decorator" 패턴
Data processing Stream class
레고블록
생성자에서 다른 InputStream은 받지 않는다-
"완성품 블록"
예) 인형 집 바퀴 선
«abstract» Component
InputStream/OutputStream
Reader/Writer
중간블록 예) 장르별 콘텐 사용 블록
FitterInputStream
«abstract» Decorator
ConcreteComponent
FileInputStream
ByteArrayInputStream
ConcreteDecorator
DataInputStream
BufferedInputStream
ObjectInputStream

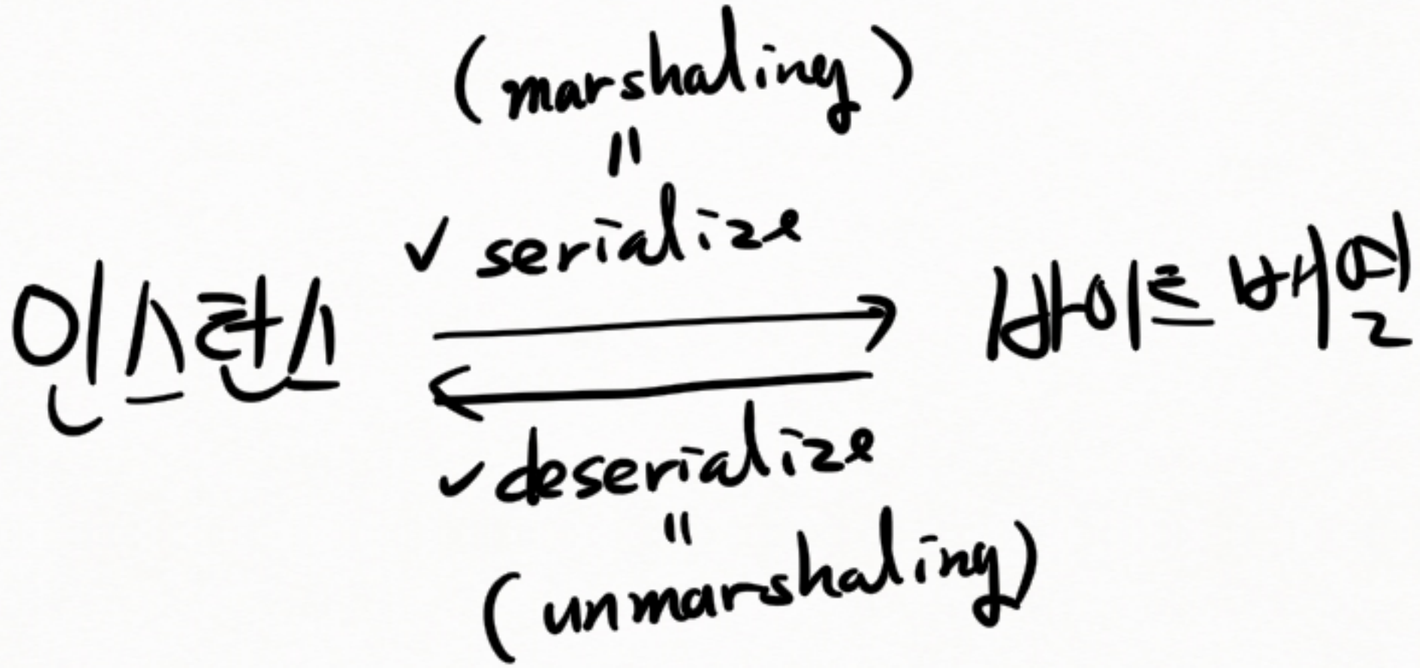
ex08.*

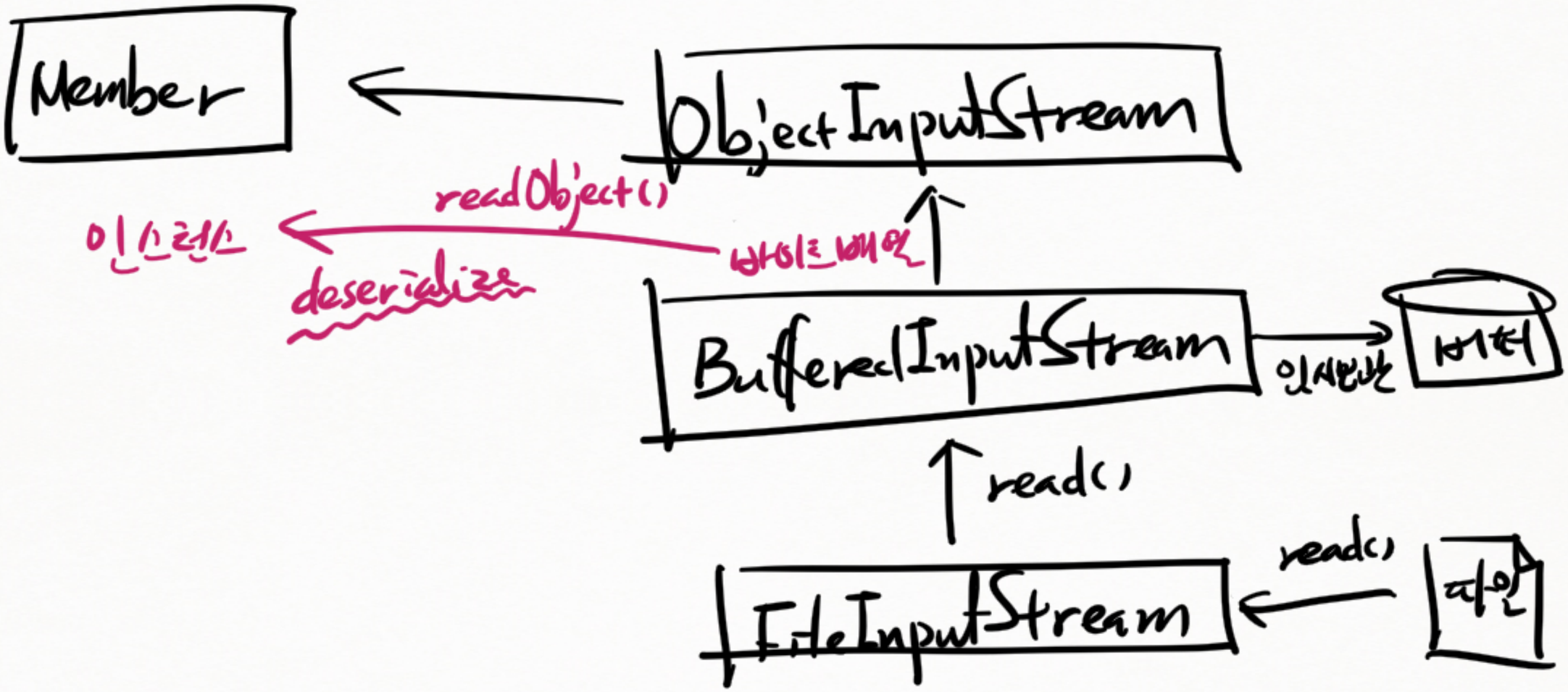

«Component»
InputStream
«concrete component»
FileInputStream
ByteArrayInputStream
DataInputStream
BufferedInputStream
BufferedInputStream ≒ InputStream 자료의
객체가 아니다!
필요할까!

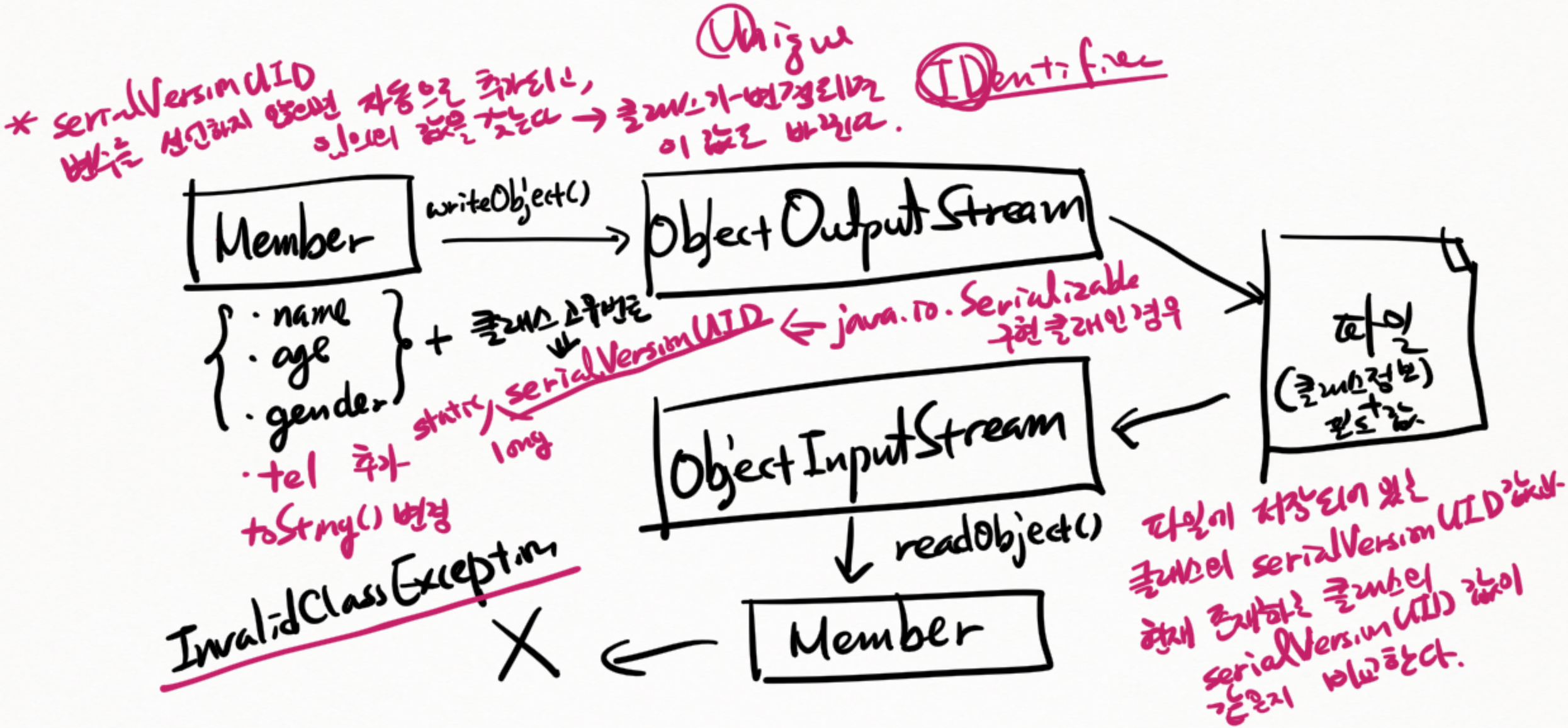

ex01.step1.*
«concrete component»
FileInputStream
ByteArrayInputStream
«Component»
InputStream
DataInputStream
BufferedInputStream

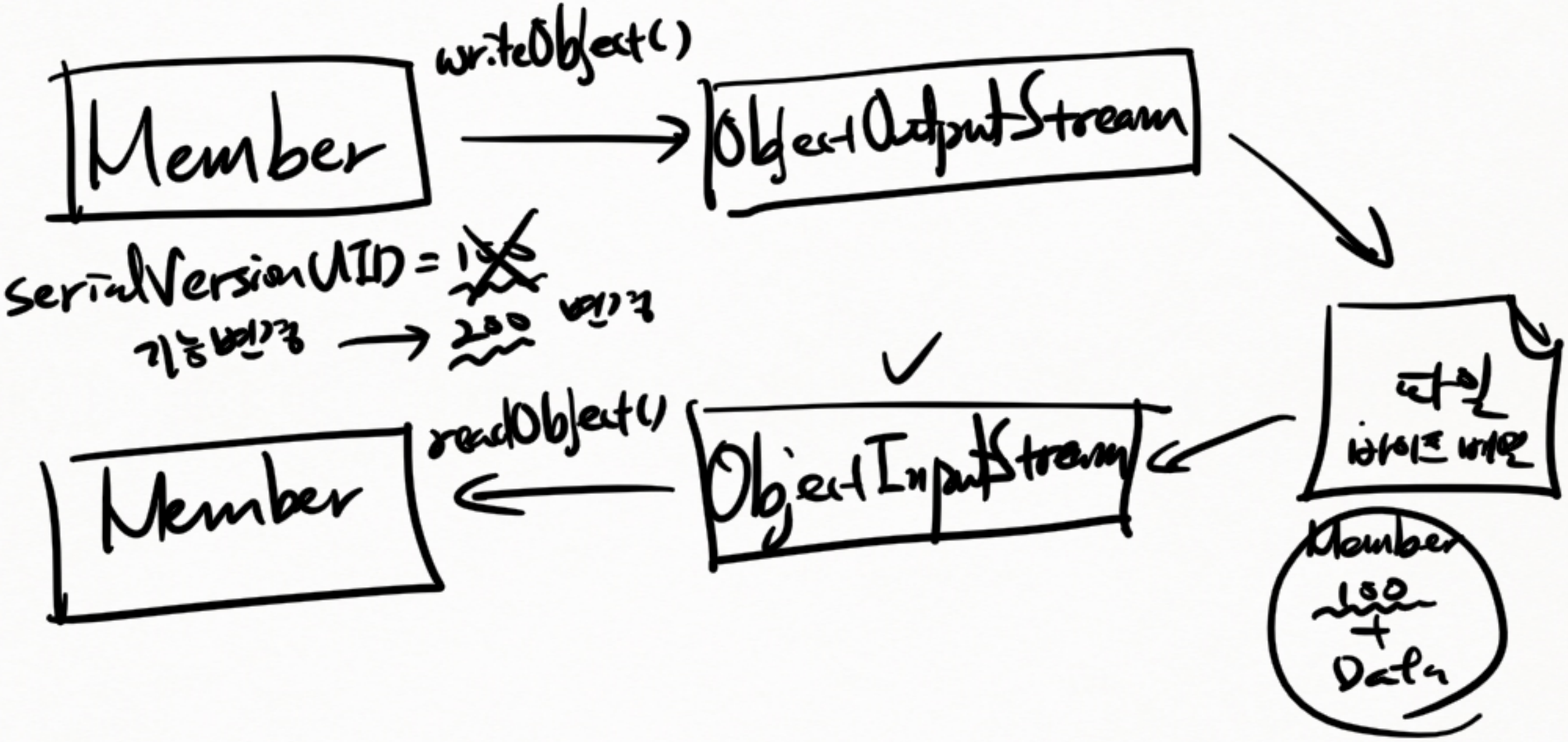

ex09.step2.*
«component»
InputStream
«concrete component»
FileInputStream
ByteArrayInputStream
DecoratorInputStream
DataInputStream
BufferedInputStream

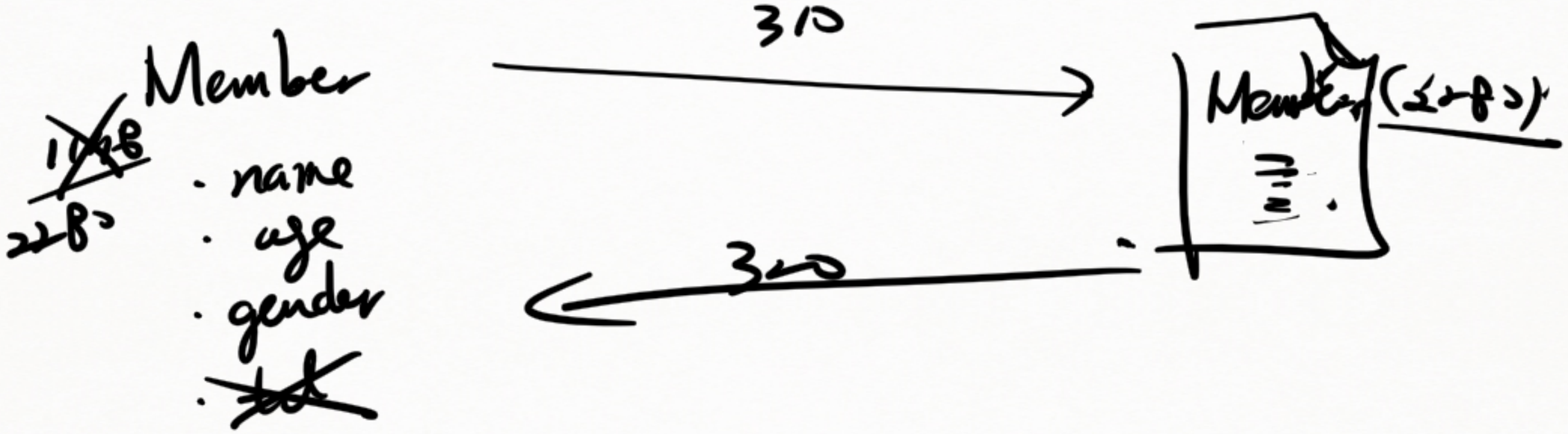

인스턴스
serialize
Member
.name
.age
.gender
writeObject(x)
인스턴스
ObjectOutputStream
byte[]
write(x)
serialize
직렬화
인스턴스 필드값
+
클래스정보
BufferedOutputStream
임시보관
buffer
write()
FileOutputStream
출력
파일

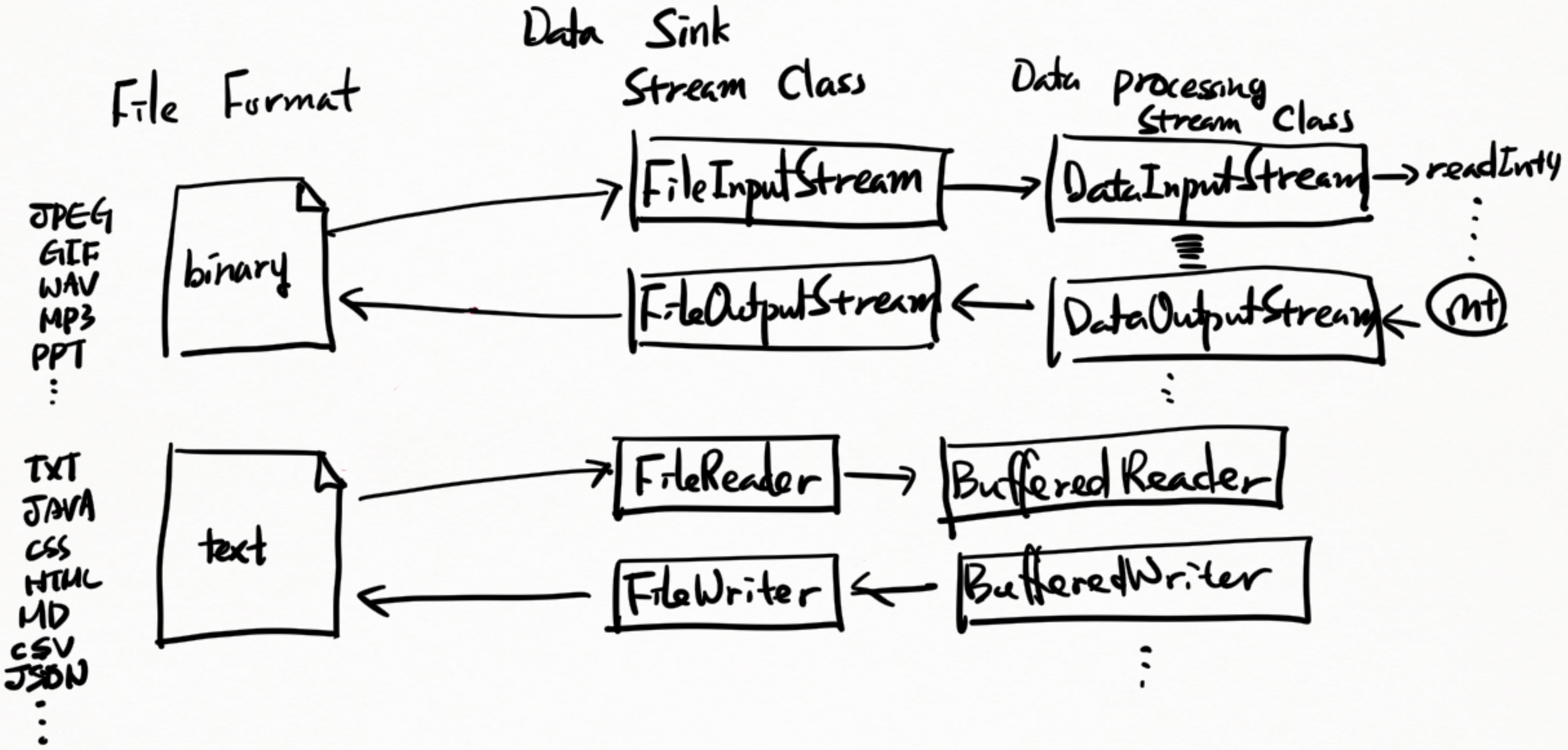

(marshaling)
=
✓ serialize
인스턴스
바이트 배열
✓ deserialize
=
(unmarshaling)


Member
ObjectInputStream
readObject()
인스턴스
바이트배열
deserialize
BufferedInputStream
입력받은
버퍼
read()
FileInputStream
read()
파일


* serialVersionUID
변수는 선언하지 않으면 자동으로 추가되는,
이노의 값을 찾는다 → 클래스가 변경되면
이 값도 바뀐다.
Object
IDentifier
Member
writeObject()
ObjectOutputStream
· name
· age
· gender
+ 클래스 수정정보
· tel 추가
toString() 변경
static
serialVersionUID
long
java.io.Serializable
구현클래스일경우
파일
(클래스정보
포함)
ObjectInputStream
readObject()
Member
InvalidClassException
X
파일에 저장되어 있는
클래스의 serialVersionUID값과
현재 존재하는 클래스의
serialVersionUID 값이
같은지 비교한다.


Member
writeObject()
ObjectOutputStream
serialVersionUID = 1
Member
readObject()
ObjectInputStream
Member
+
Data


Member
. name
. age
. gender
310
320


Data Sink
File Format
Stream Class
Data processing Stream Class
JPEG
GIF
WAV
MP3
PPT
binary
FileInputStream
DataInputStream
readInt4
FileOutputStream
DataOutputStream
int
TXT
JAVA
CSS
HTML
MD
CSV
JSON
text
FileReader
BufferedReader
FileWriter
BufferedWriter

- 30-a : FileInputStream / FileOutputStream
- 30-b : DataInputStream / DataOutputStream
- 30-c : BufferedInputStream / BufferedOutputStream
- 30-d : ObjectInputStream / ObjectOutputStream

} file format을 알아야 읽고 쓸 수 있다

← 다른 언어와 호환이 안된다

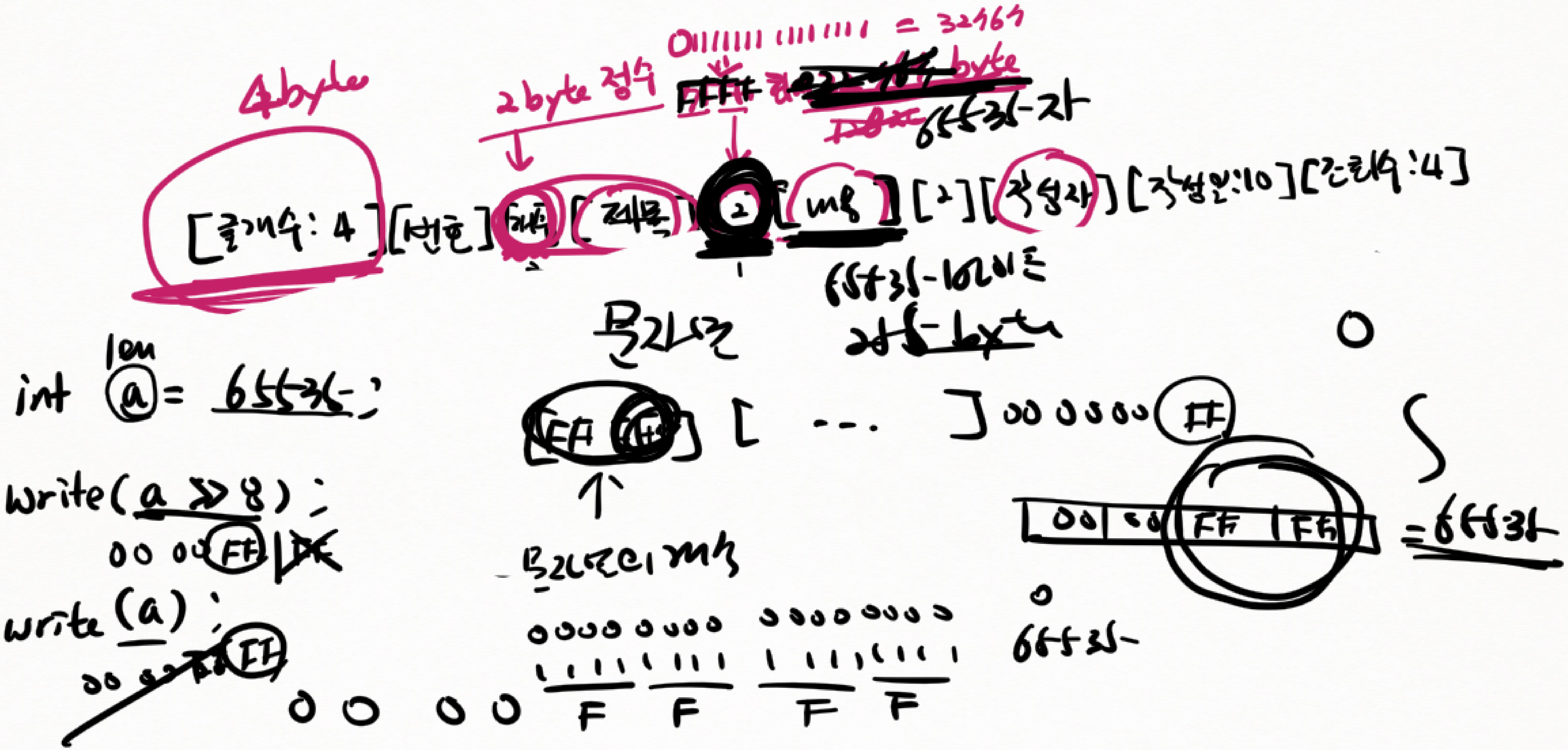
30-e : 리팩터링

31-a : FileReader / FileWriter text 데이터 포맷

31-b : BufferedReader / BufferedWriter

31-c : 리팩토링 I

31-d : 리팩토링 II

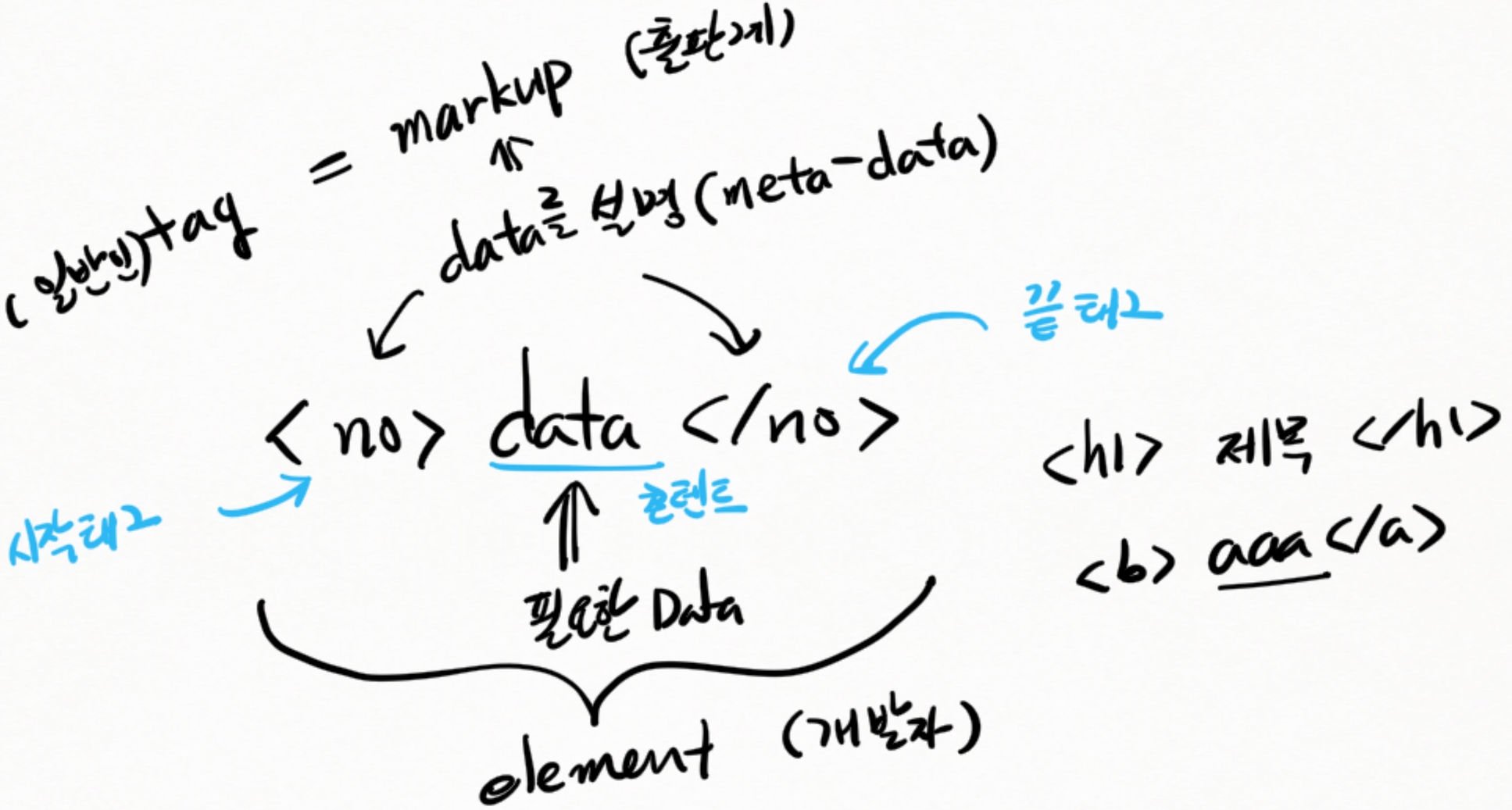


4byte
2byte 정수
65535-자
[글개수 : 4] [번호] [제목] [2] [msg] [2] [작성자] [작성일:10] [조회수:4]
65535-바이트
255-byte
문자열
문자열의 길이
int a = 65535;
write(a >> 8);
write(a);
00 00 FF
= 65535
65535

XML
HTML

(일반적) tag = markup (출판계)

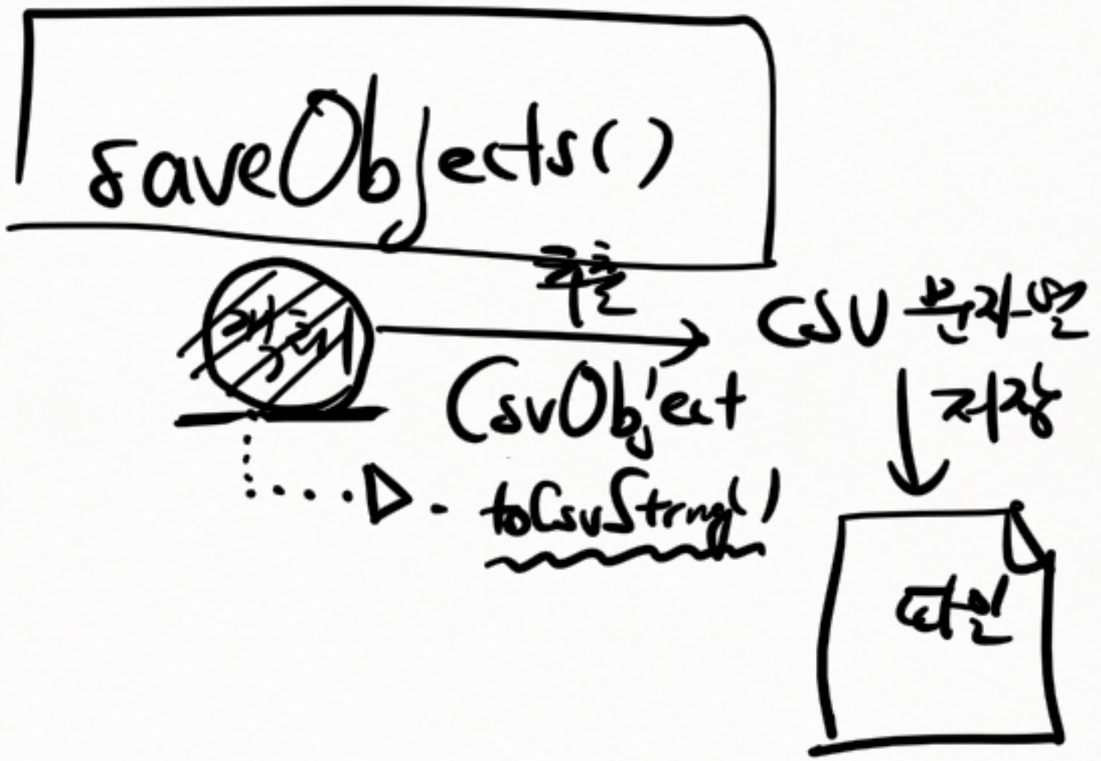
data를 설명 (meta-data)

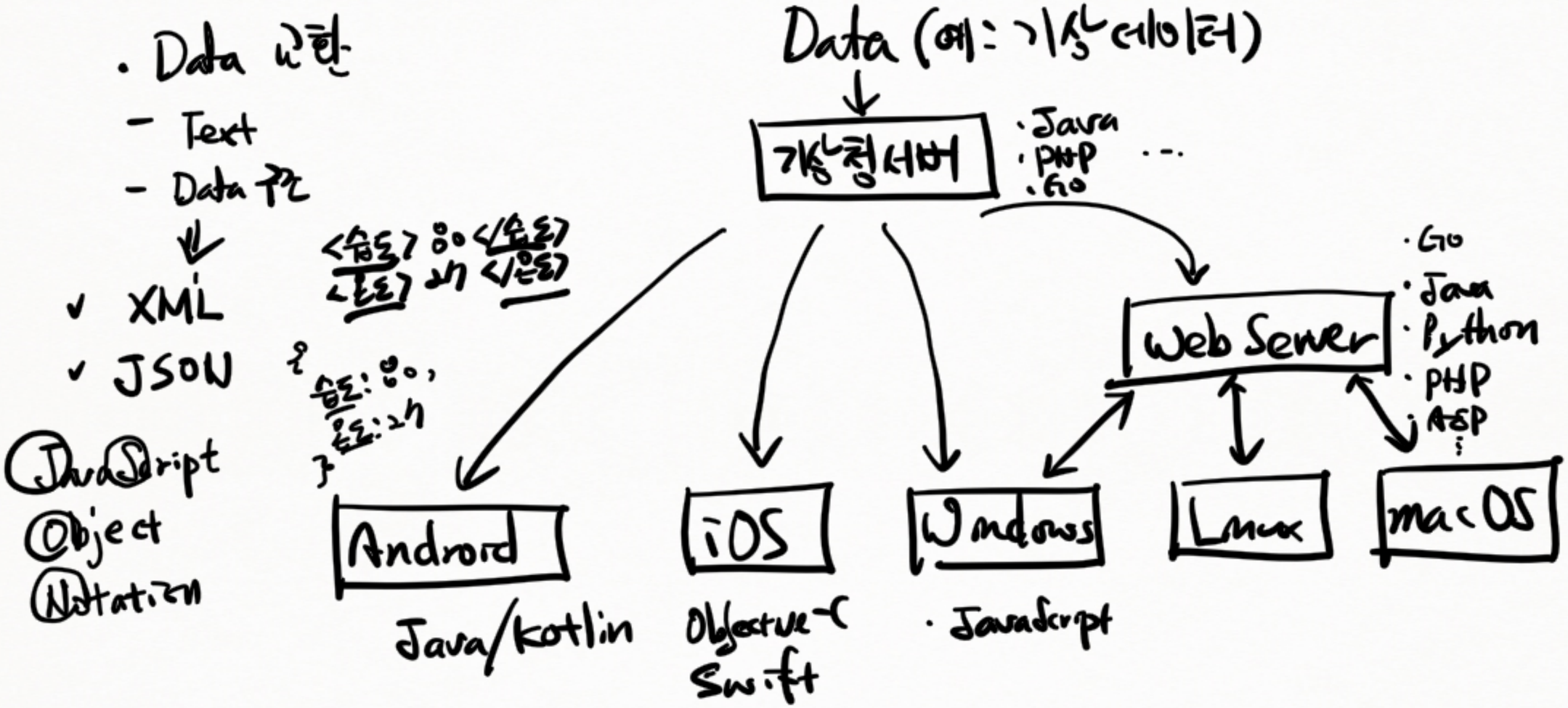


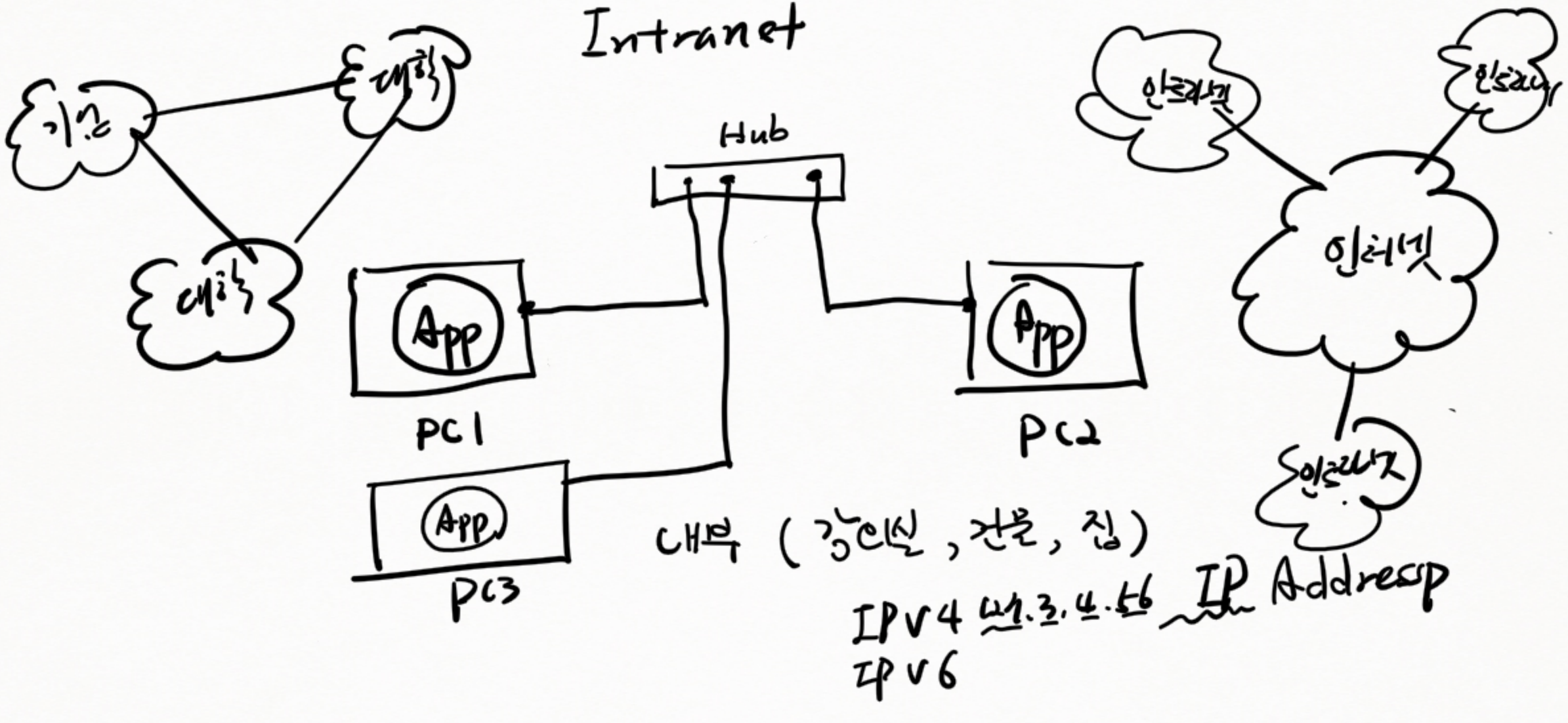
<h1> 제목 </h1>

<b> aaa

saveBoards()
saveMembers()
saveProjects()
saveTasks()




. Data 전달
- Text
- Data 구조
XML
JSON
JavaScript
Object
Notation
Data (예: 가상데이터)
가상환경서버
· Java
· PHP
· Go
Web Server
· Go
· Java
· Python
· PHP
· ASP
Android
iOS
Windows
Linux
macOS
Java/kotlin
Objective-C
Swift
· Javascript


Intranet
Hub
App
PC1
App
PC2
App
PC3
대역 (강남점 , 건물 , 집)
IPv4 111.3.4.56 IP Address
IPv6

0

약 42

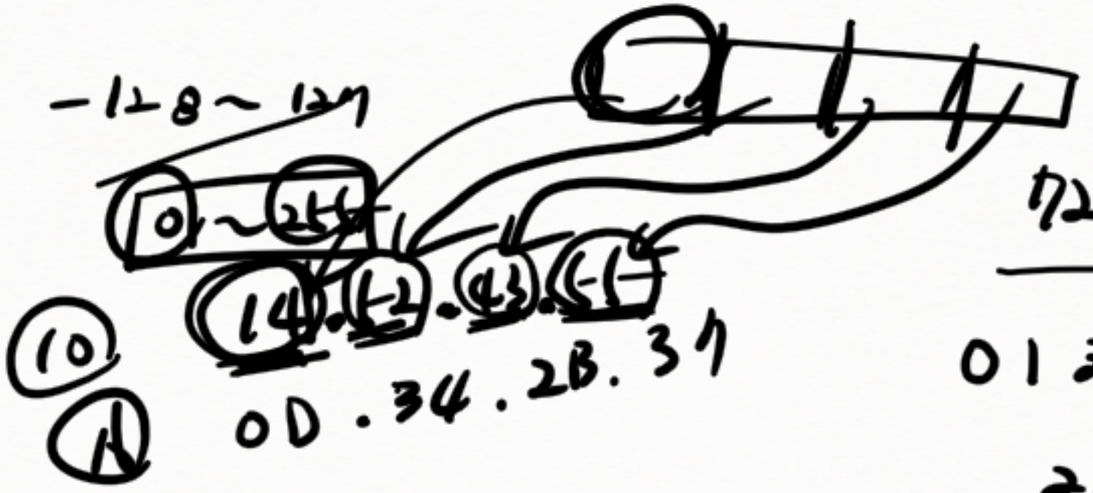
010-1234-4444

0.0.0.0 = 4byte

1byte 1byte 1byte 1byte

127.354.117

221.122.143



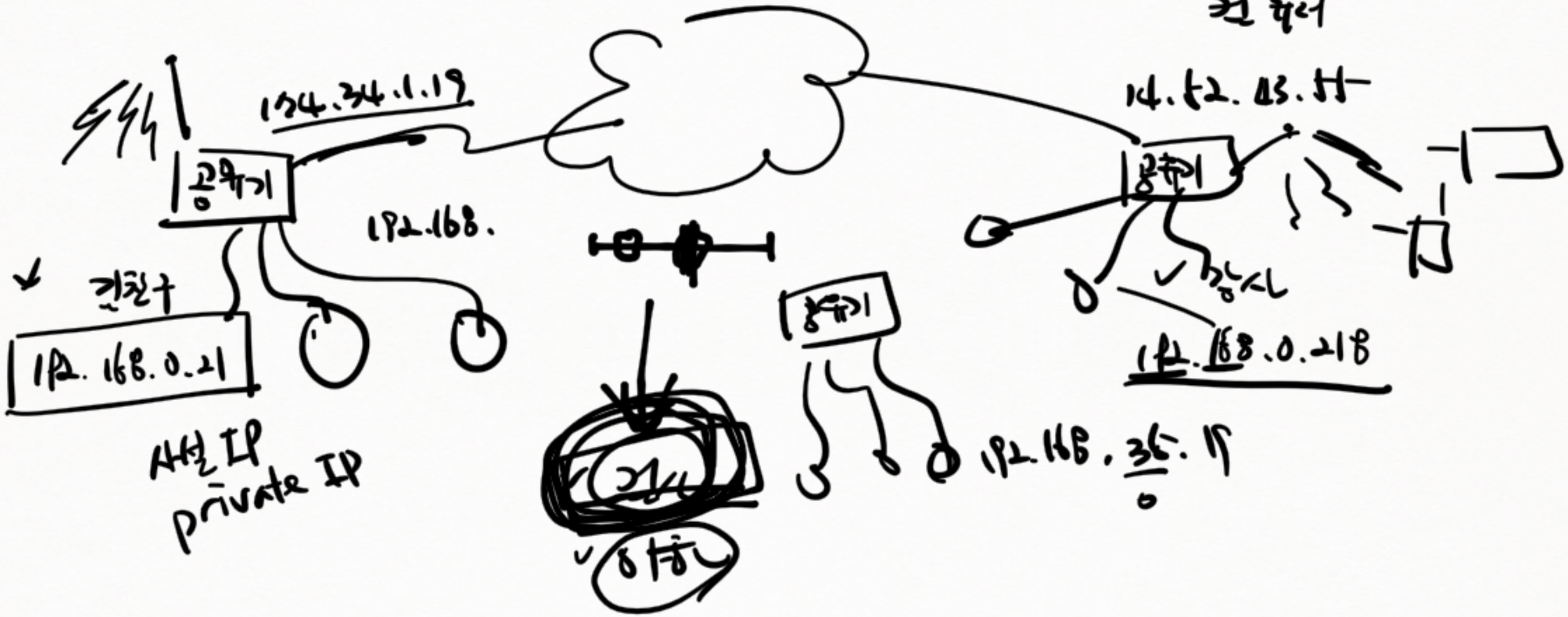
02001-122123x

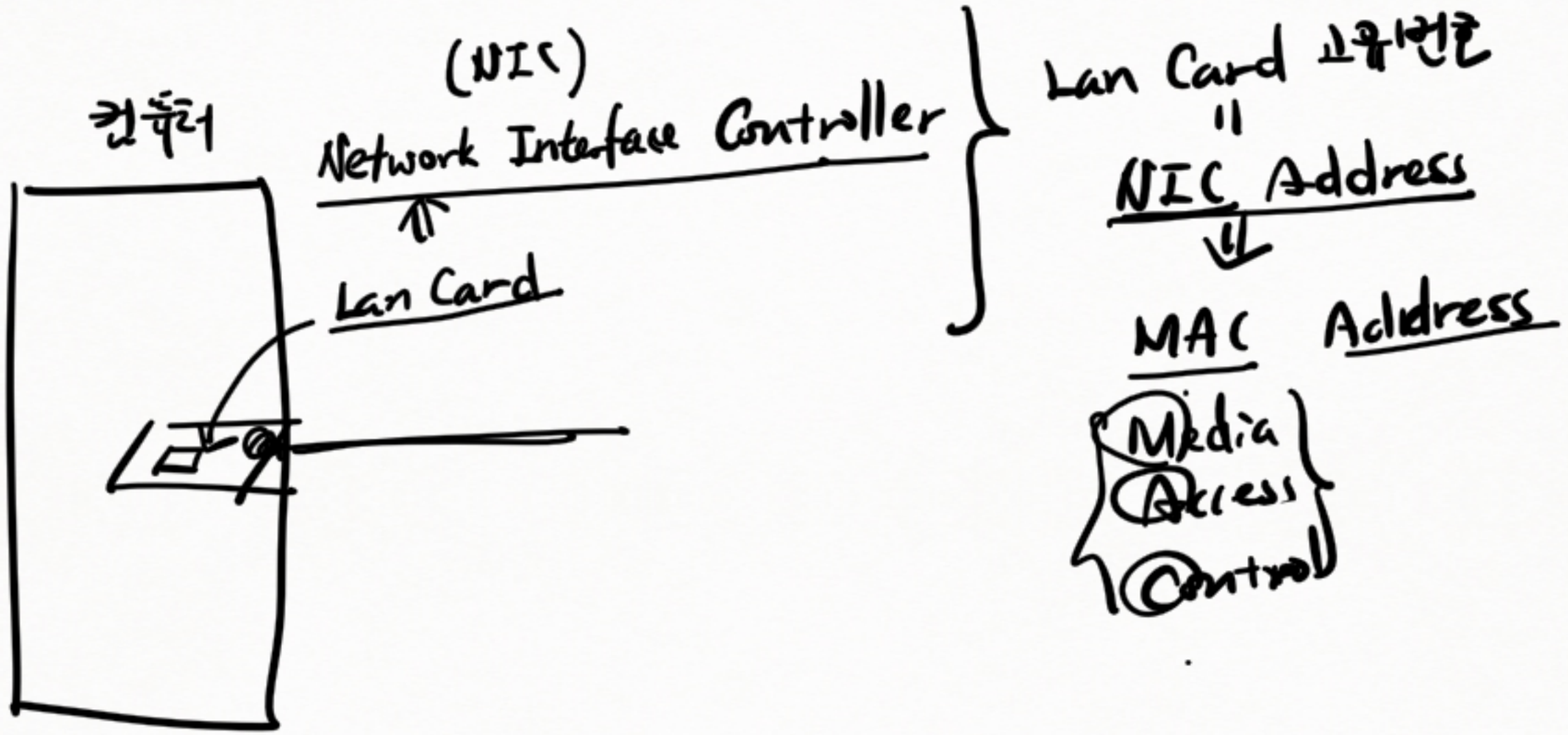
01234...

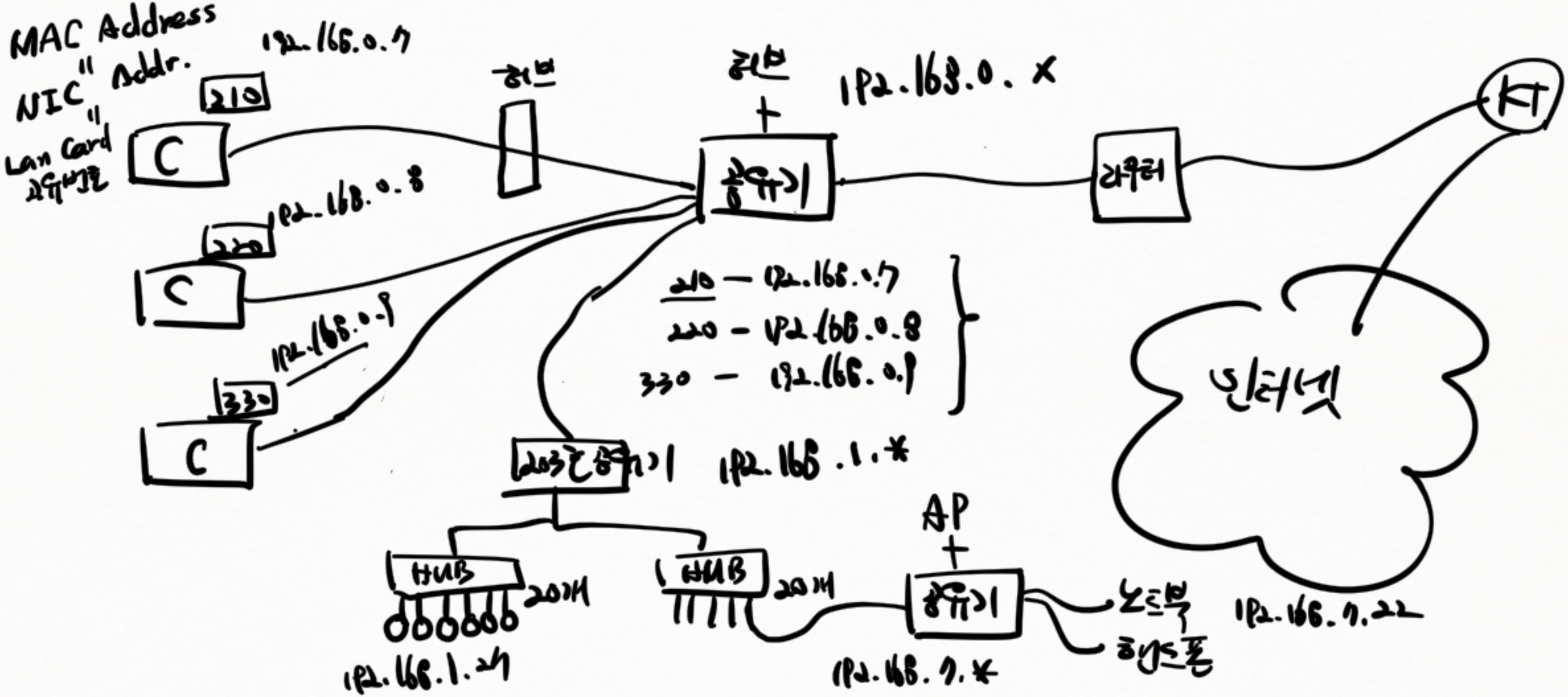
221.122.143 423

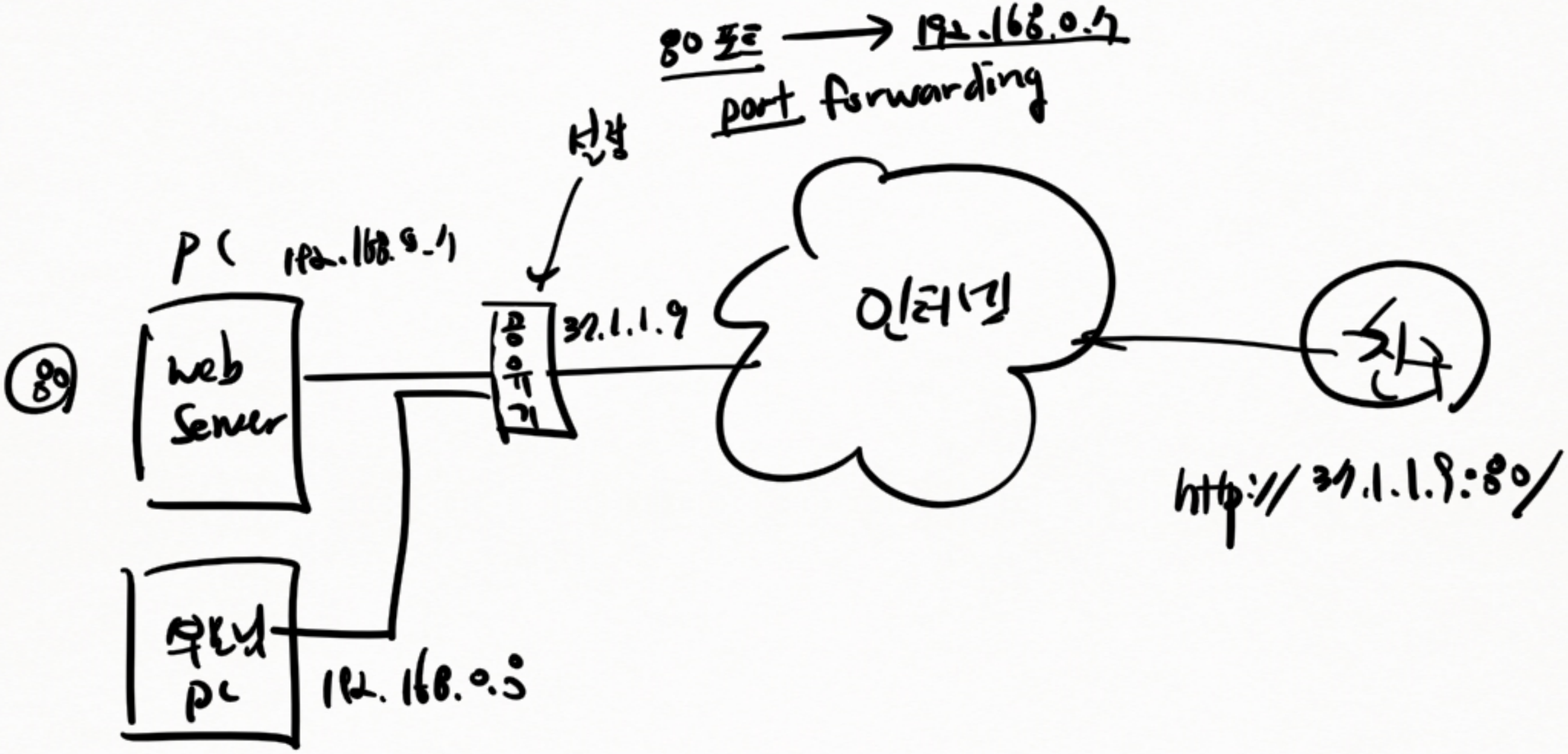
v4

IP Address 4Byte = 약 42억 대의 컴퓨터

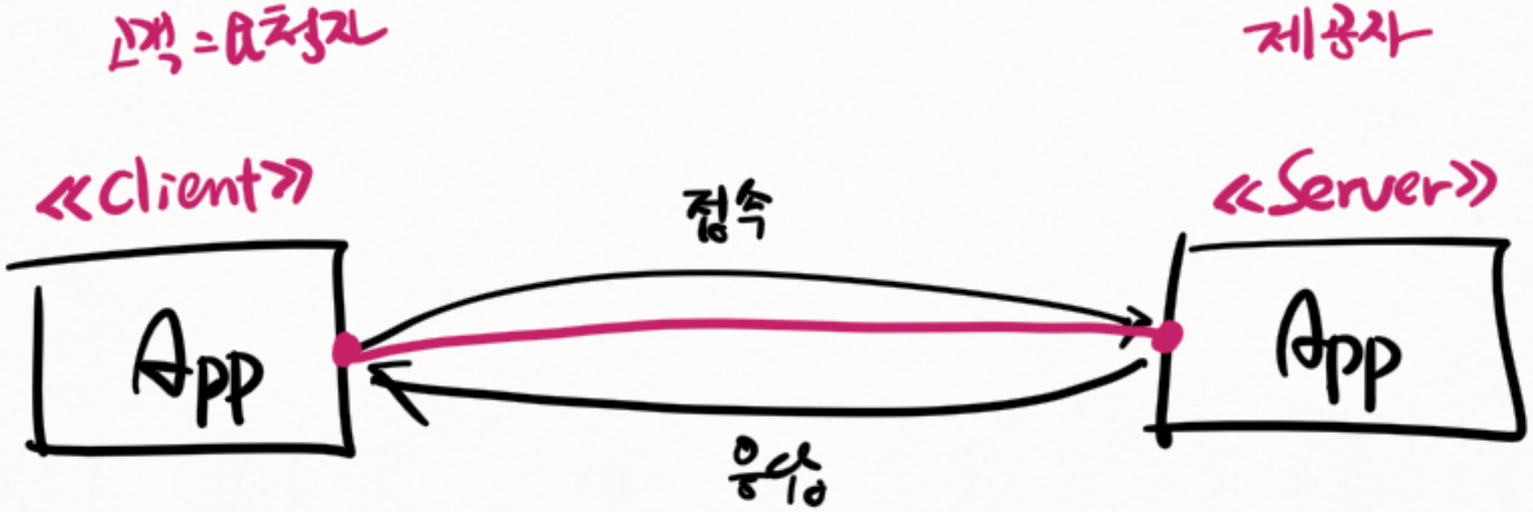


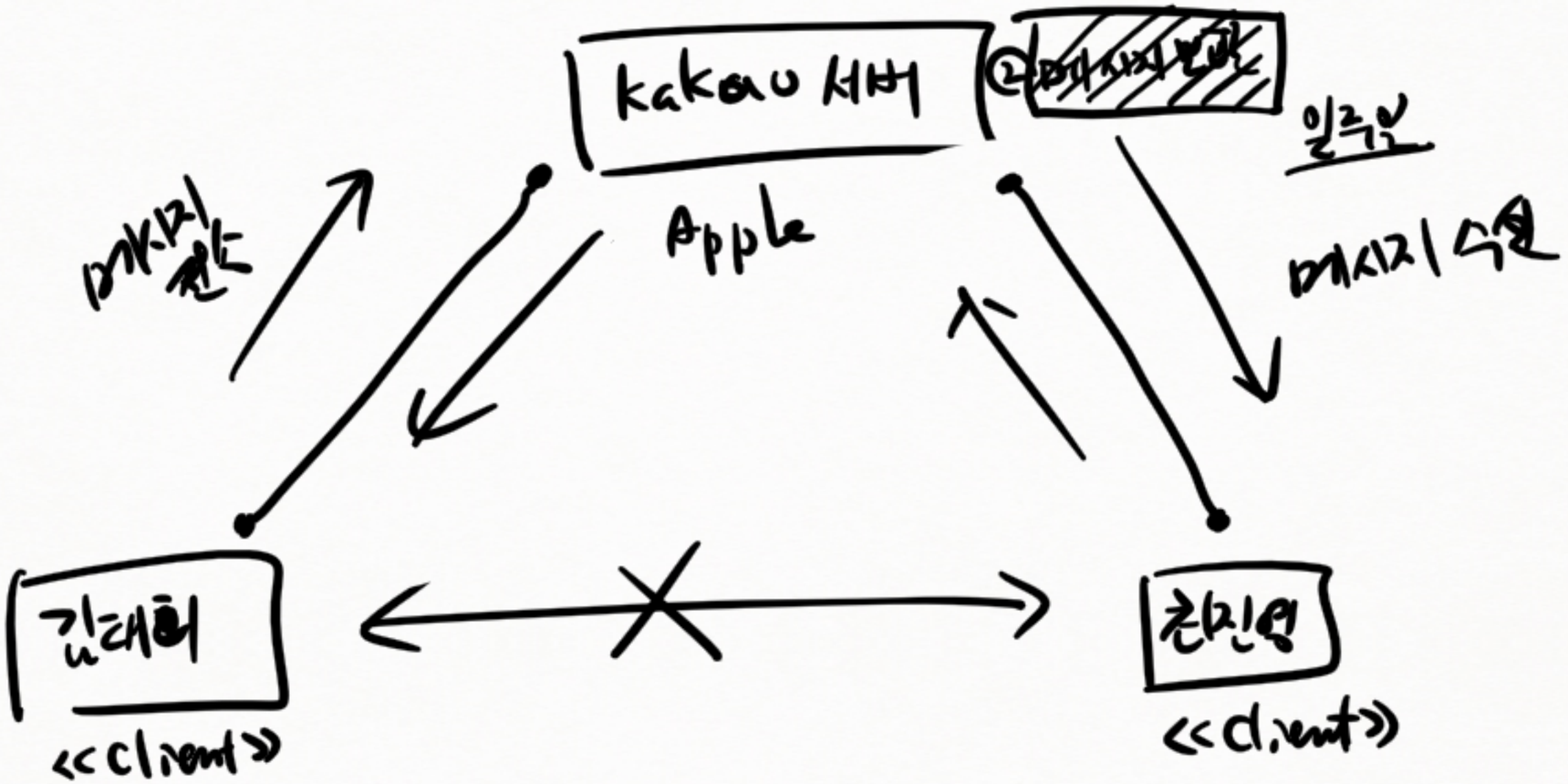

컴퓨터
(NIC)
Network Interface Controller
Lan Card
Lan Card 고유번호
=
NIC Address
MAC Address
Media
Access
Control

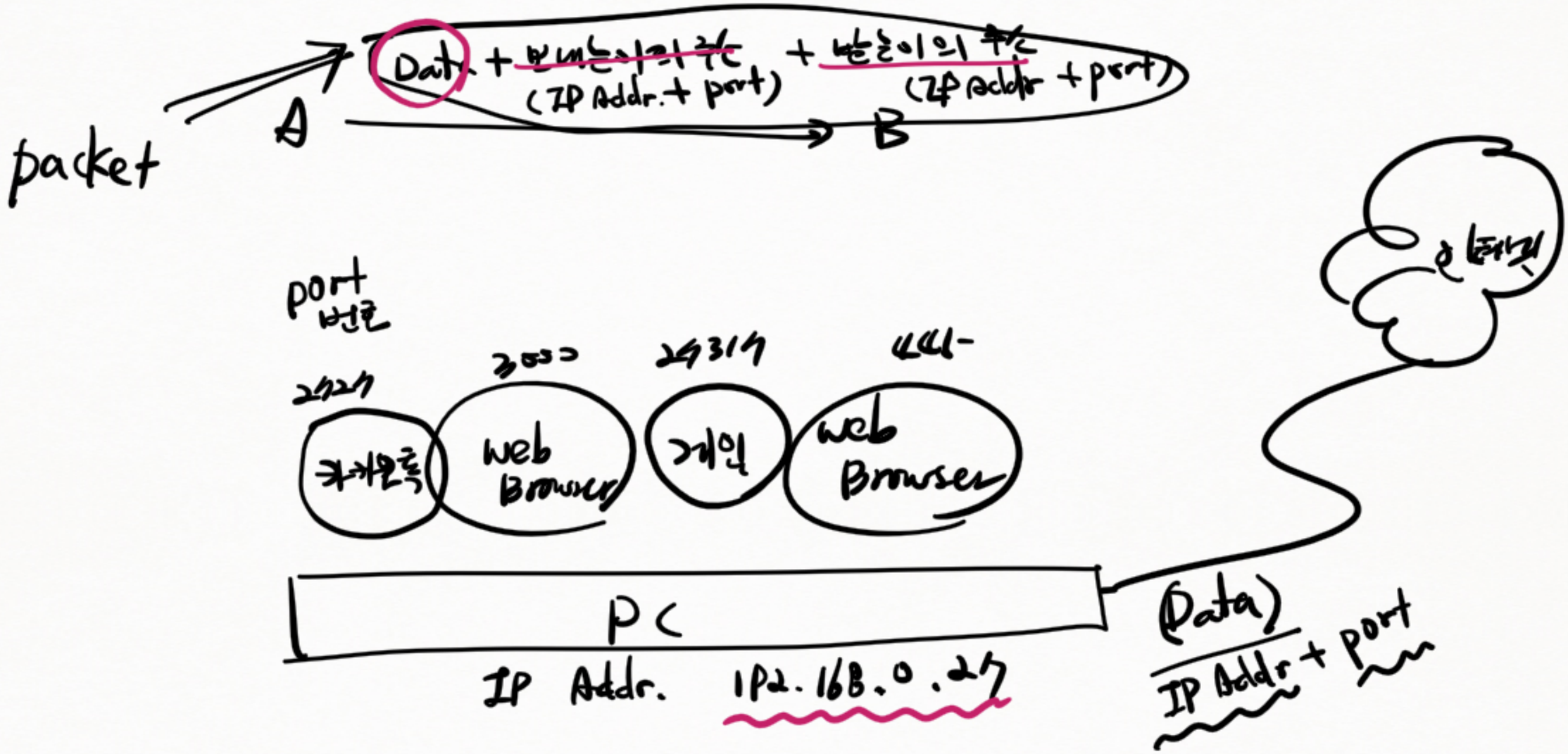

MAC Address
NIC "
Lan Card
랜카드
192.168.0.7
210
220
330
192.168.0.8
192.168.0.9
허브
허브
+
공유기
192.168.0.X
라우터
KT
210 - 192.168.0.7
220 - 192.168.0.8
330 - 192.168.0.9
무선공유기
192.168.1.*
HUB
20개
HUB
20개
AP
+
공유기
노트북
휴대폰
192.168.1.2개
192.168.1.*
192.168.1.22
인터넷

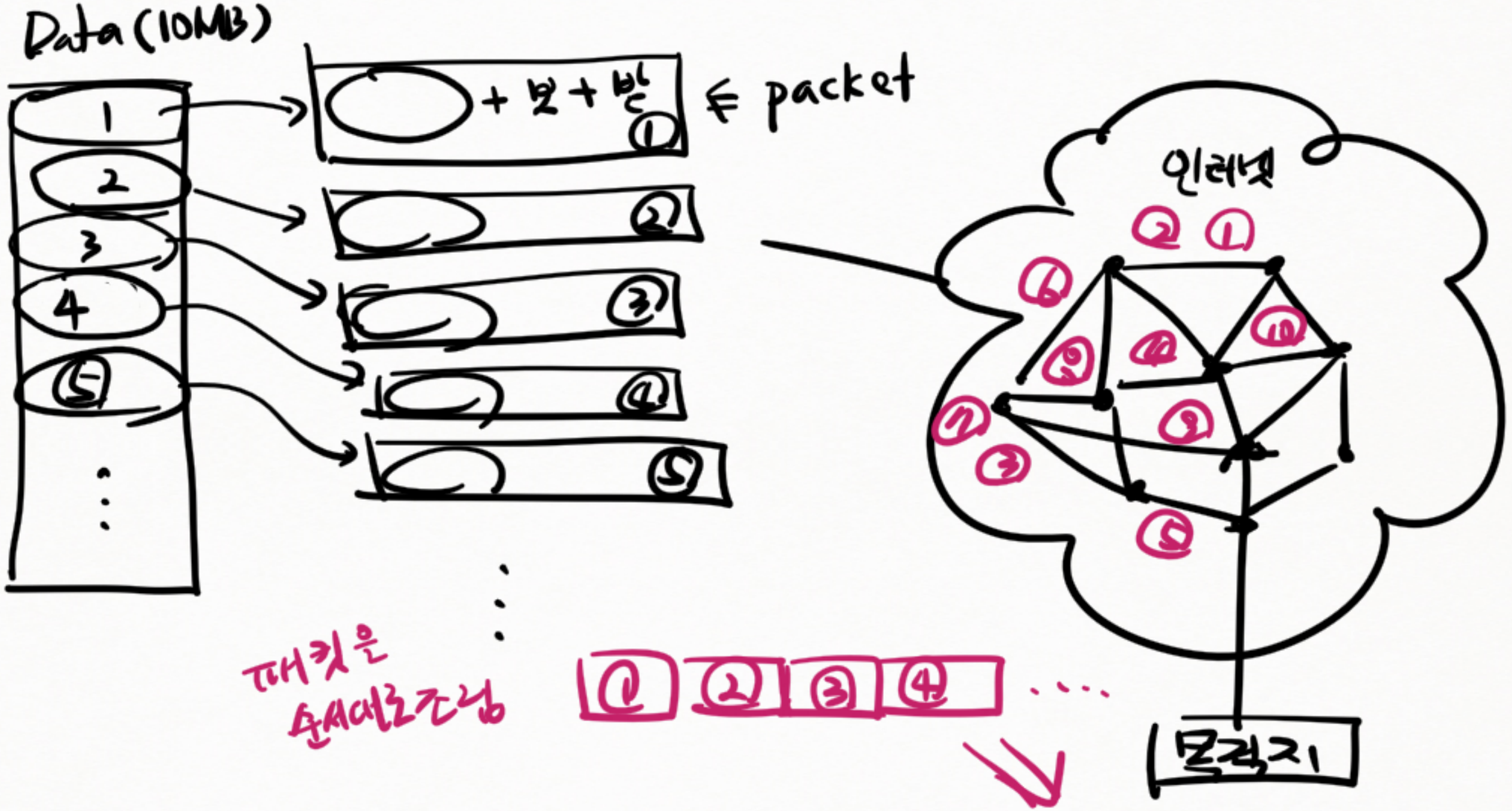

80 포트 → 192.168.0.7
port forwarding
선생
IP (192.168.0.1
80
Web Server
공유기
32.1.1.9
인터넷
친구
http://32.1.1.9:80/
무선 pc
192.168.0.3

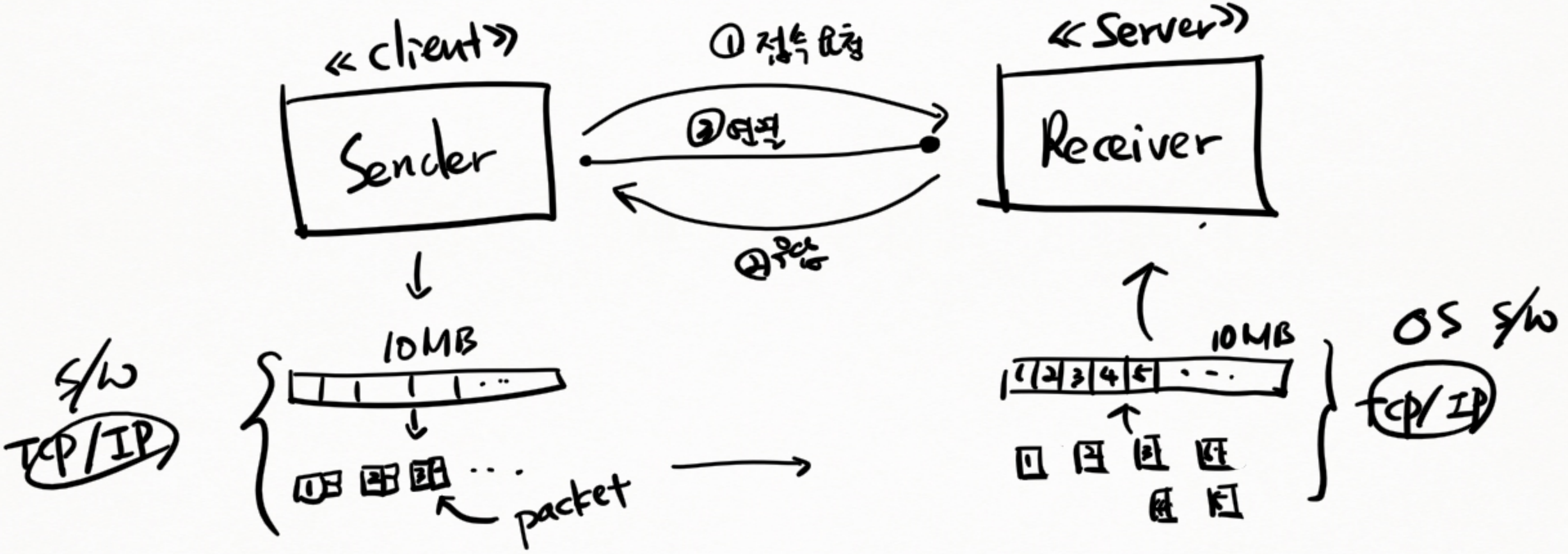
PC




kakao 서버
Apple
메시지 전달
읽음
메시지 수정
김대희
<<client>>
천민형
<<client>>


packet
A
Data + 보내는사람 주소 (IP Addr. + port) + 받을이의 주소 (IP Addr + port)
B
port 번호
2222
3000
2431
4444
web Browser
게임
web Browser
인터넷
PC
IP Addr. 192.168.0.27
(Data)
IP Addr + port


Data (10MB)
1
2
3
4
5
+ 보 + 받
packet
인터넷
패킷은 순서대로간다
목적지

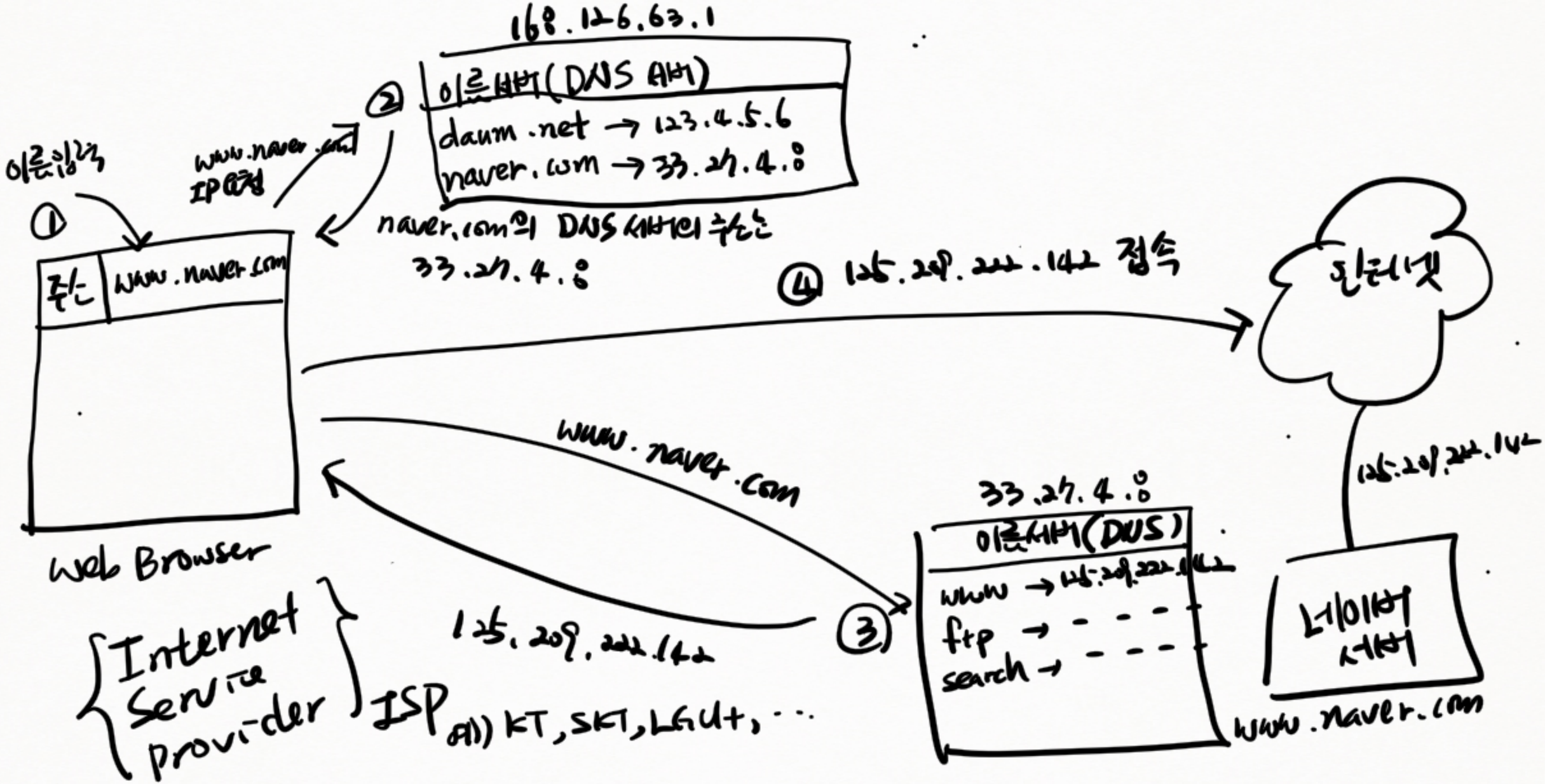

«client»
Sender
①
②
③
«Server»
Receiver
10MB
packet
S/W
TCP/IP
10MB
OS S/W
TCP/IP

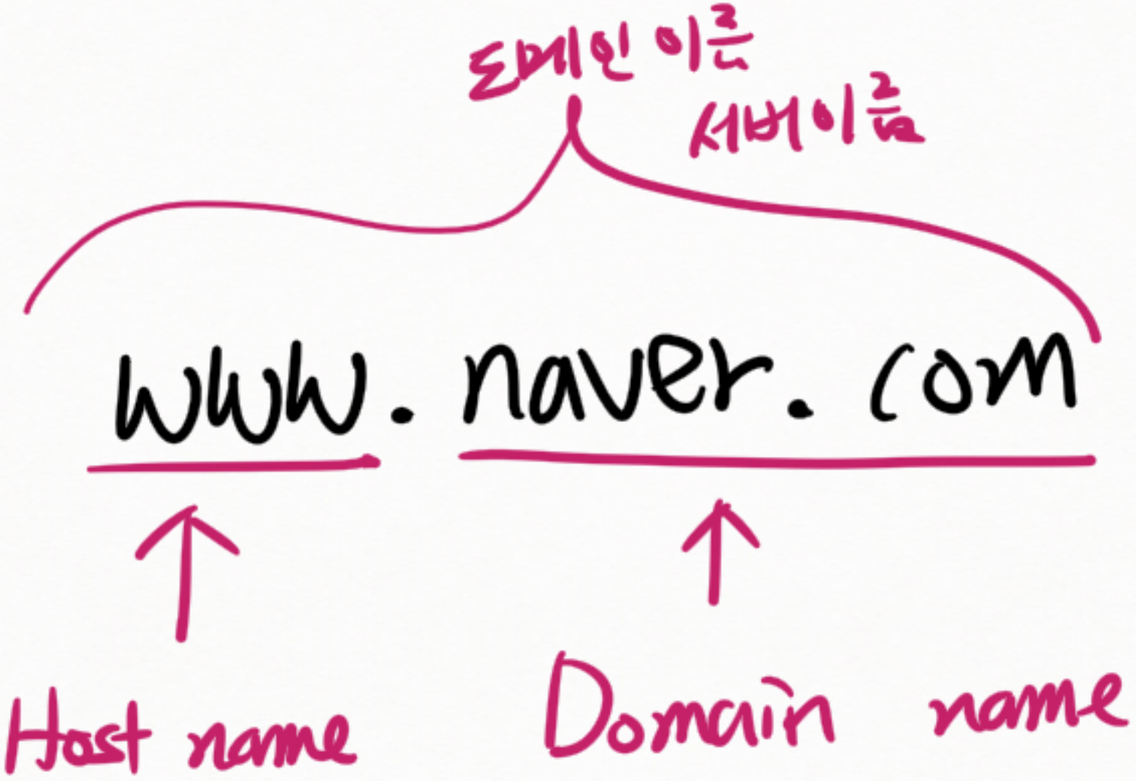
new Socket(IP Address , port)

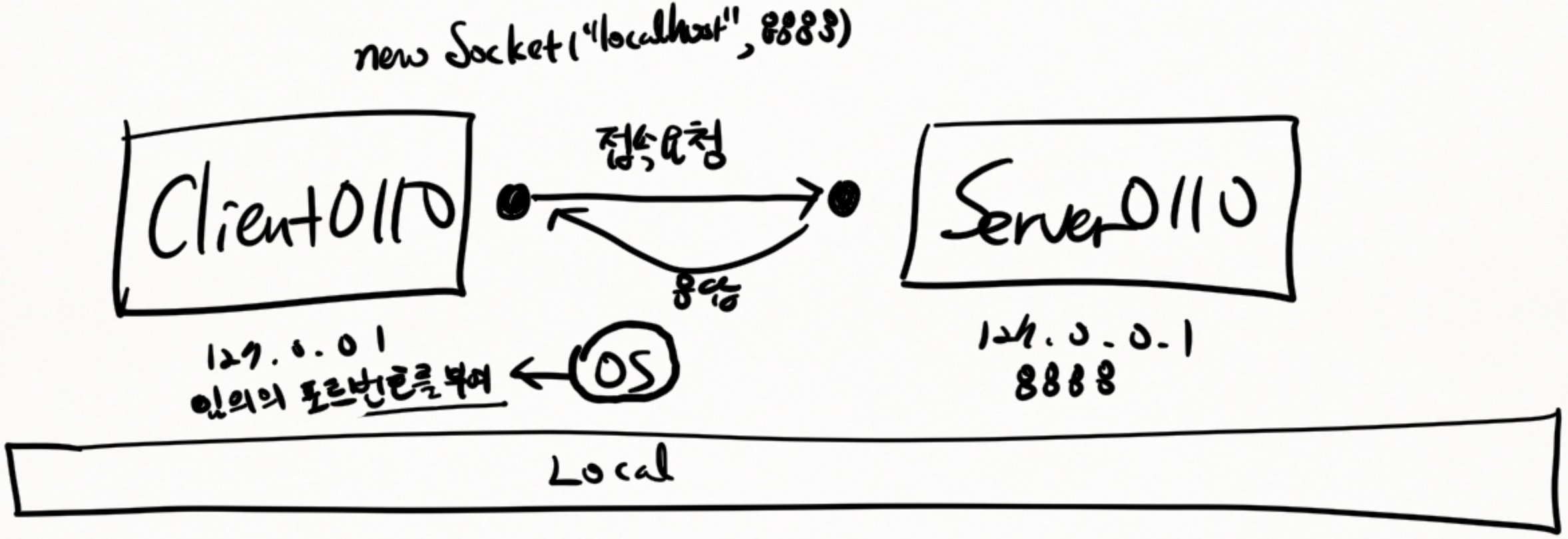
127.0.0.1 ⇒ 현재 컴퓨터

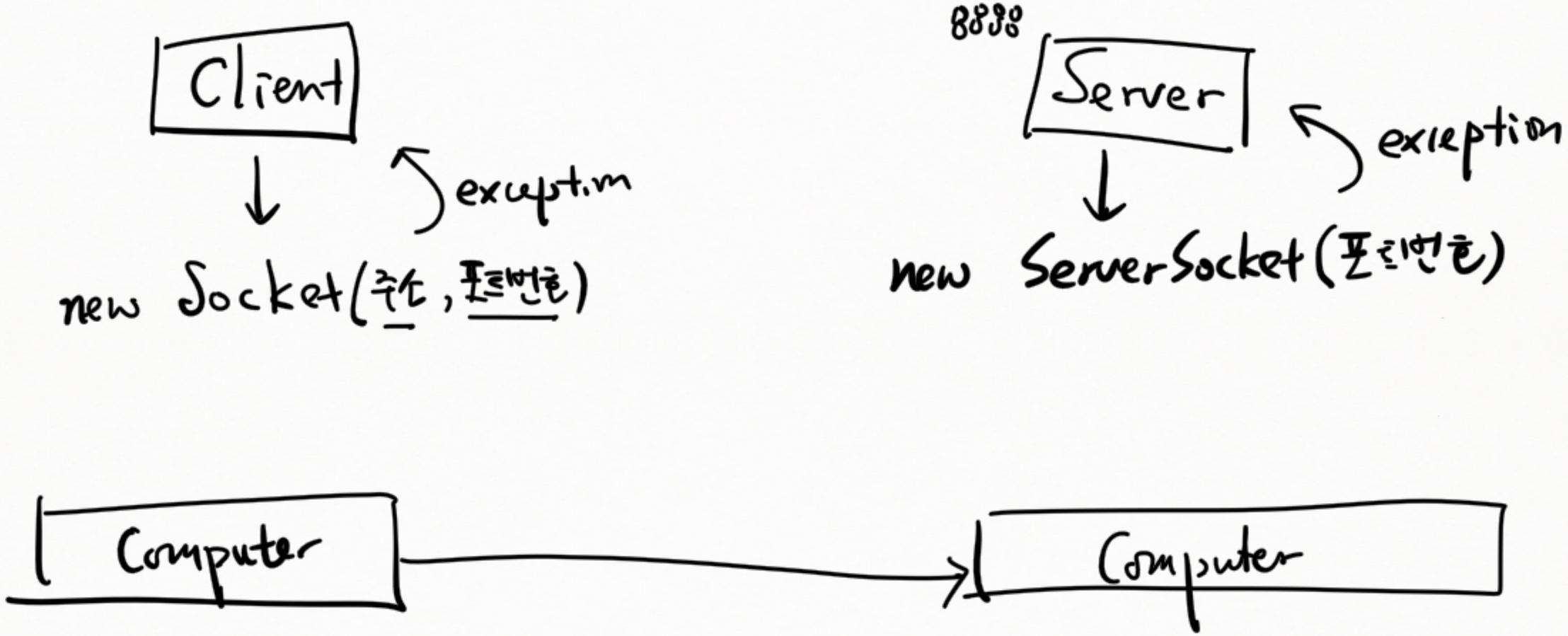
Domain Address

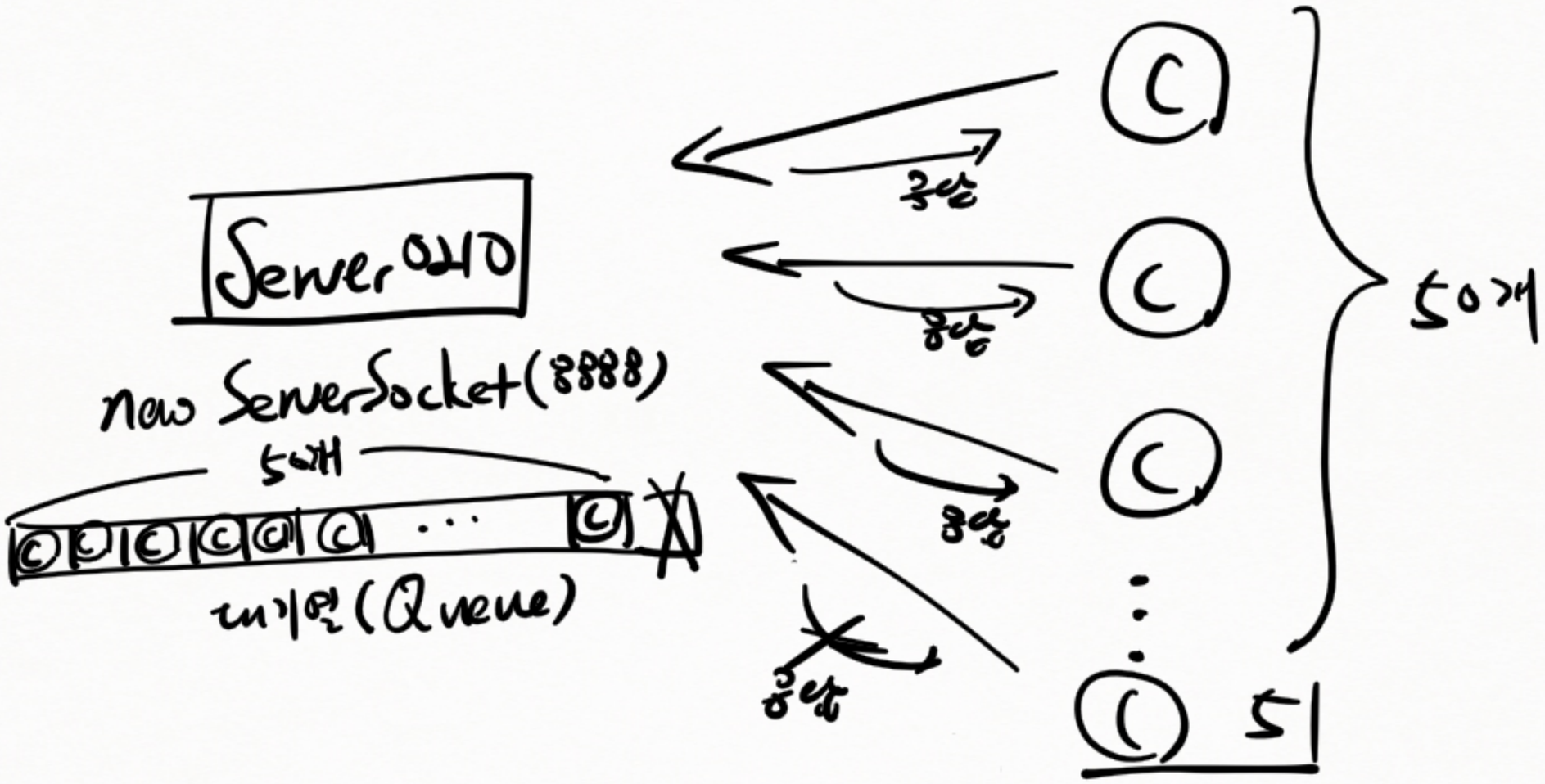
localhost ⇒ 현재 컴퓨터

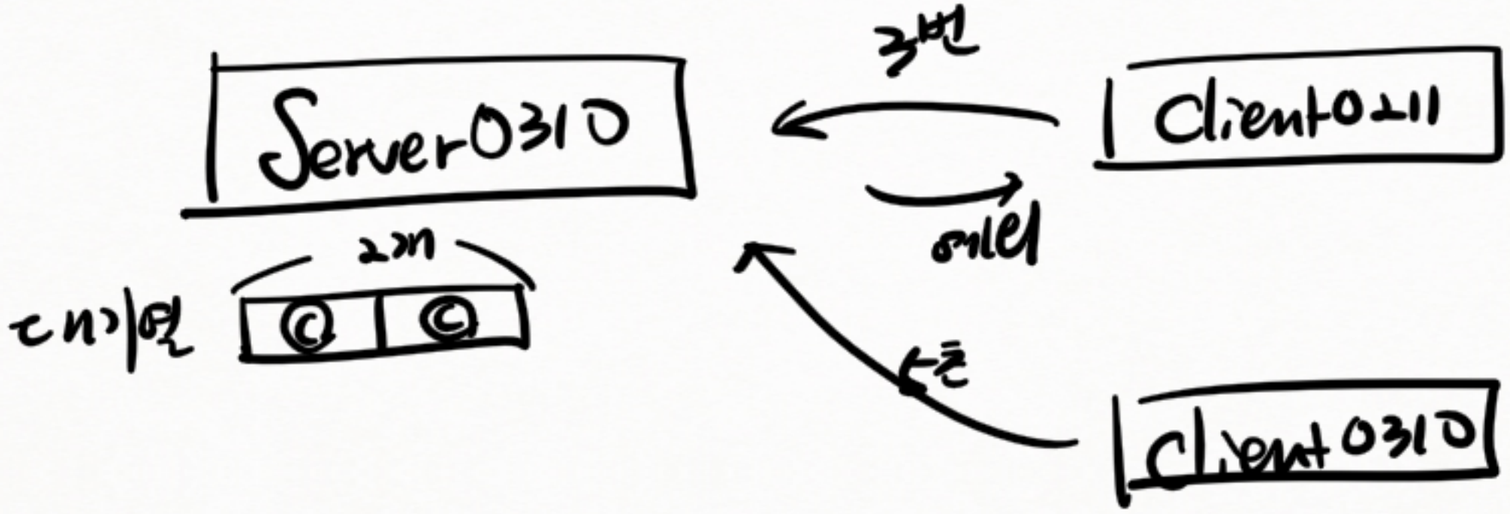

168.126.63.1
이름서버(DNS 서버)
daum.net → 123.4.5.6
naver.com → 33.27.4.0
②
이름입력
www.naver.com
IP요청
①
주소
www.naver.com
Web Browser
naver.com의 DNS서버의 주소는
33.27.4.0
④ 125.209.222.142 접속
인터넷
www.naver.com
125.209.222.142
③
33.27.4.0
이름서버(DNS)
www →
ftp →
search →
125.209.222.142
네이버 서버
www.naver.com
Internet Service provider
ISP
예) KT, SKT, LG U+, ...

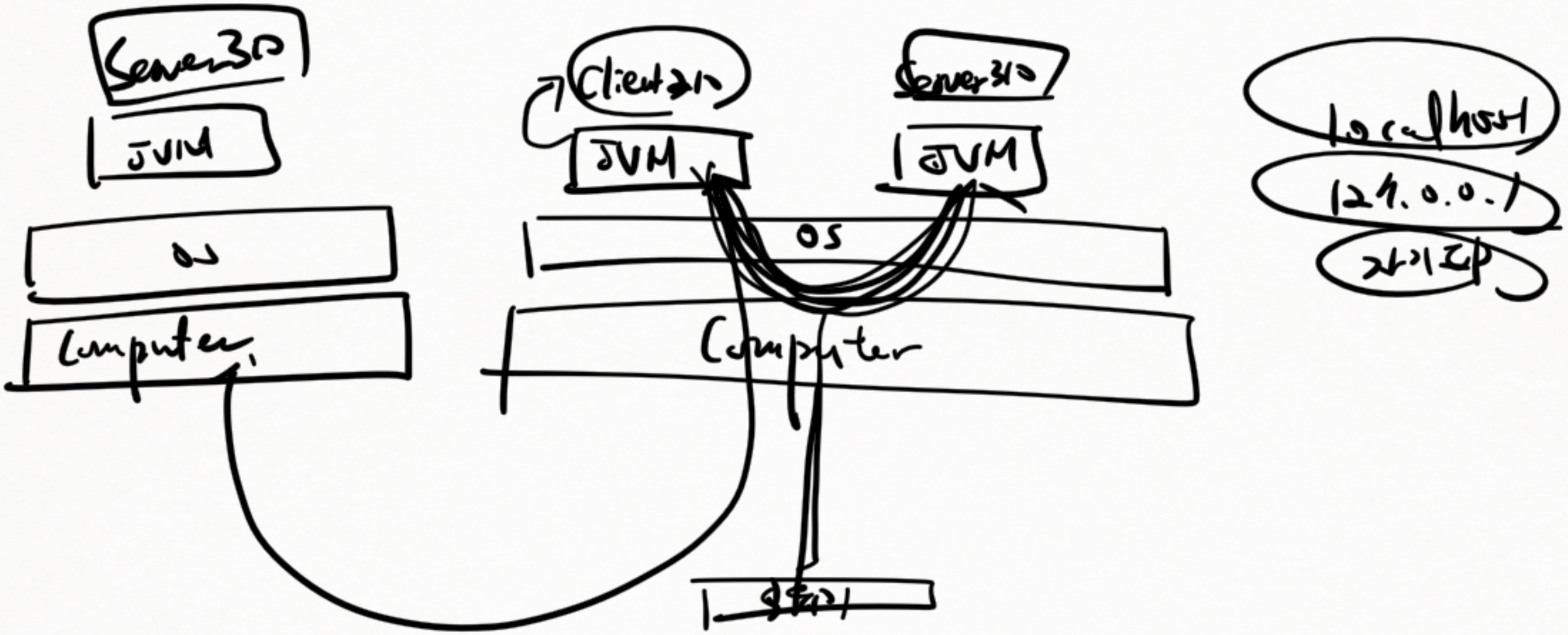

도메인 이름
서버이름
www.naver.com
Host name
Domain name

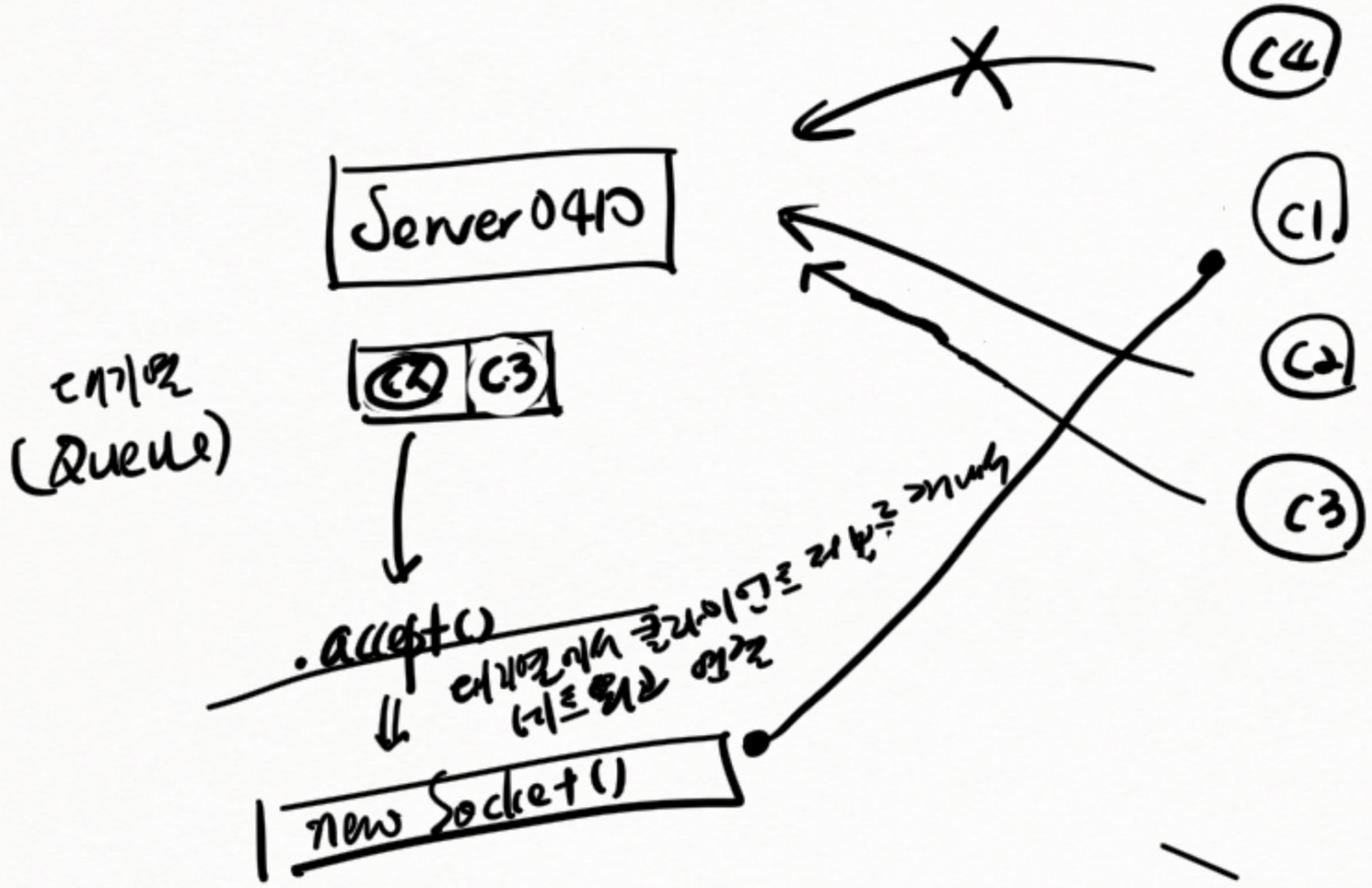

new Socket("localhost", 8888)
Client이미지
접속요청
응답
Server이미지
127.0.0.1
임의의 포트번호를 부여
OS
127.0.0.1
8888
Local

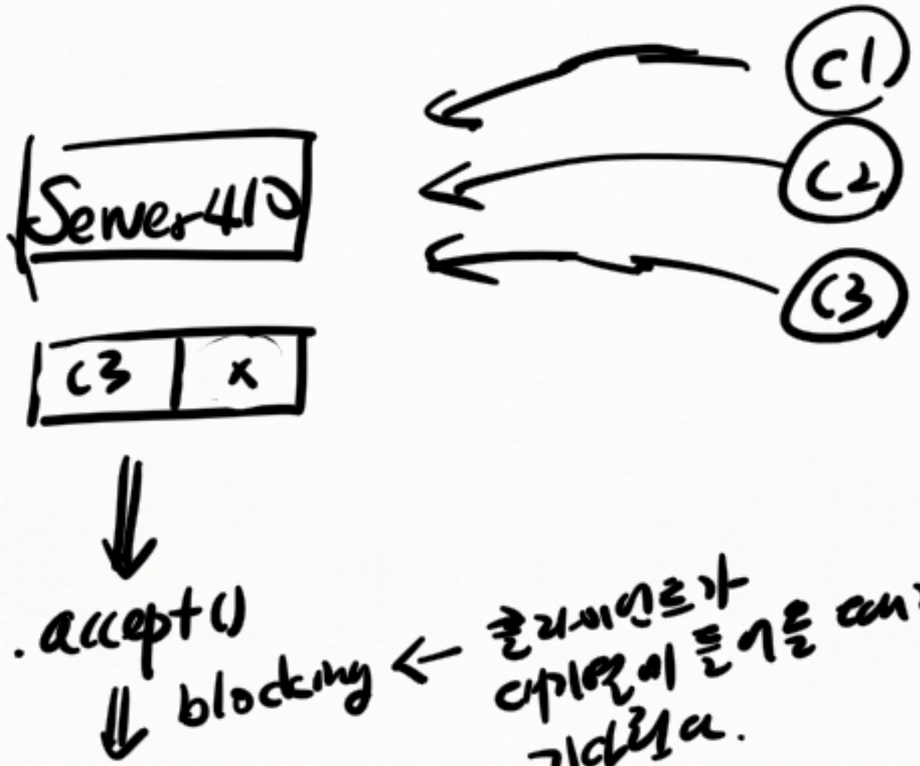

Client
exception
new Socket(주소, 포트번호)
8888
Server
exception
new ServerSocket(포트번호)
Computer
Computer

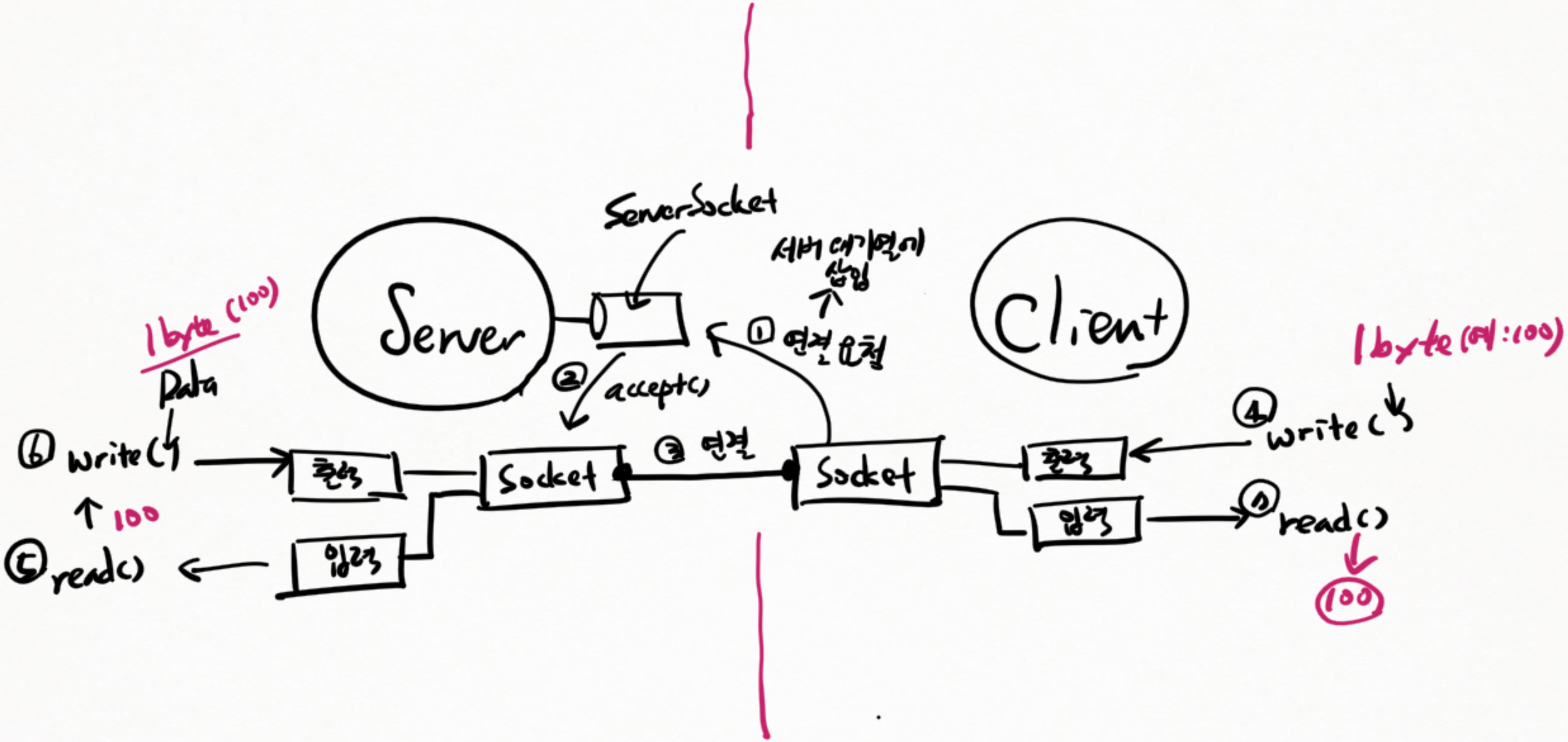

Server 입니다
new ServerSocket(8888)
5번째
대기열(Queue)
클라이언트

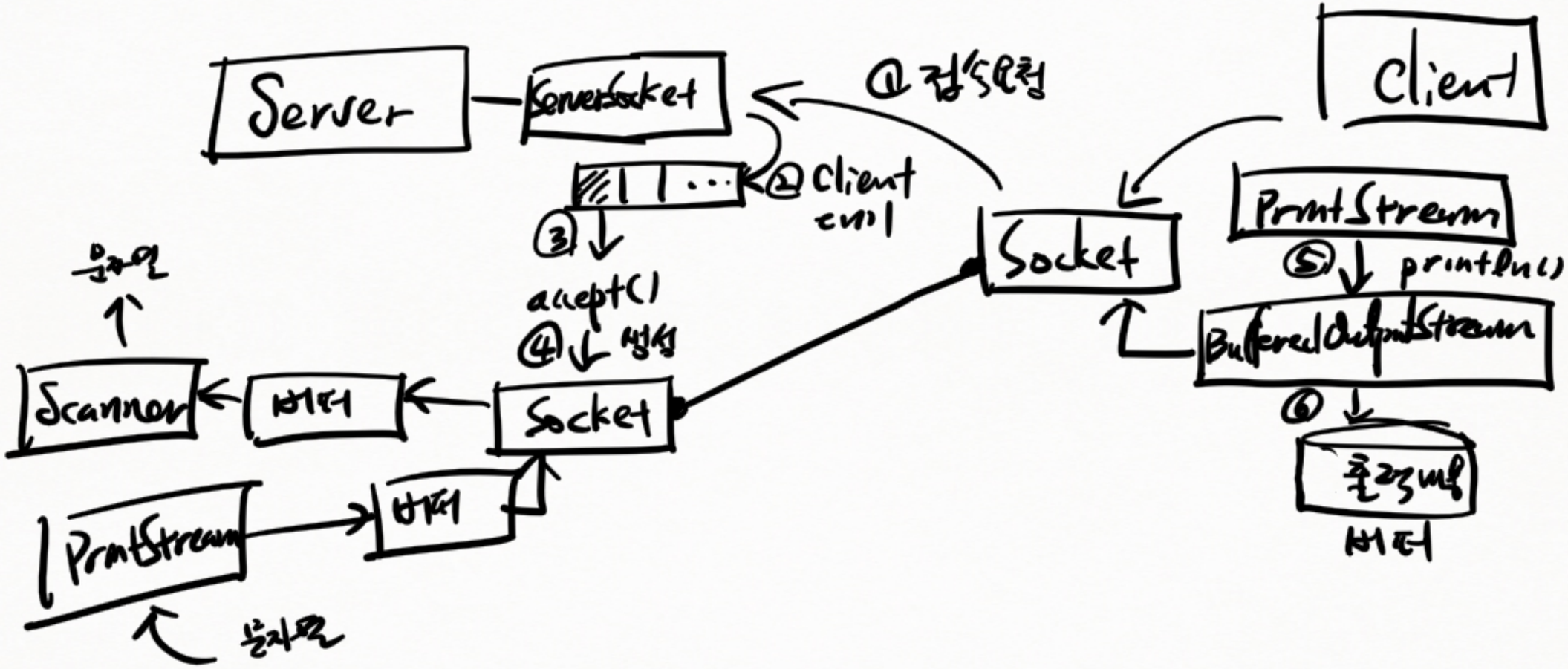

Server0310
3번
Client0211
메세지
2번
대기열
보냄
Client0310


Server3D
JVM
OS
Computer
Client3D
JVM
Server3D
JVM
OS
Computer
localhost
127.0.0.1
포트번호


C4
Server 대기
C1
C2
C3
대기열
(Queue)
C2
C3
.accept()
대기열에서 클라이언트를 꺼내옴
네트워크 연결
new Socket()


Server 4000
C1
C2
C3
C3
X
.accept()
blocking
클라이언트가
대기열에 들어올 때까지
기다림.


ServerSocket
서버 대기 (listen) 상태
Server
Client
① 연결 요청
② accept()
③ 연결
1byte (100) Data
1byte (예: 100)
⑥ write()
↑ 100
⑤ read()
④ write()
① read()
100
Socket
Socket
출력
입력
출력
입력


Server
ServerSocket
① 접속요청
Client
② Client 대기
③
accept()
④ 생성
Socket
Socket
PrintStream
⑤ println()
BufferedOutputStream
⑥
출력버퍼
버퍼
Scanner
버퍼
입력
PrintStream
버퍼
보내요