**0. 공통 명명 규칙**

\* 대소문자가 구분되며 길이에 제한이 없다.

\* 예약어를 사용해서는 안 된다.

\* 숫자로 시작해서는 안 된다.

\* 특수문자는 '\_' 와 '$'만을 허용한다.

\* 카멜 표기법(camelCase)를 사용한다.

\* 반의어는 반드시 대응하는 개념으로 사용해야 한다.

**1. 패키지(Package) 명명 규칙**

\* 패키지명은 표준 패턴을 따라야 한다.

ex) [com].[Company].[Project].[TopPackage].[LowerPackage]

\* 패키지명은 가급적 한 단어의 명사를 사용한다.

ex) 좋은 예 : com.nexon.sudden.member.object

ex) 나쁜 예 : sudden.memberObject

**2. 클래스(Class) 명명 규칙**

\* 클래스명에는 파스칼을 사용한다(첫 글자 대문자 + 카멜 표기법).

ex) public class HelloWorld {}

\* 인터페이스에는 특별한 접두사나 접미사를 사용하지 않고 파스칼을 사용한다.

ex) public interface Animal {}

\* 인터페이스를 구현한 클래스에는 특별한 접두사나 접미사를 사용하지 않고 파스칼을 사용한다.

ex) public class Tiger implements animal{}

\* 추상 클래스에는 특별한 접두사 접미사를 사용하지 않고 파스칼을 사용한다.

ex) public abstract class Animal {}

**3. 메소드(Method) 명명 규칙**

\* 메소드명에는 파스칼 표기법을 사용한다.

ex) public void SendMessage(String message) {}

\* 속성에 접근하는 메소드명의 접두사는 'get','set'을 사용한다.

ex) public void setDisplayName

ex) public void getDisplayName

\* 데이터를 조회하는 메소드명의 접두사는 find를 사용한다.

ex) publicc void findData(String data){}

\* 데이터를 입력하는 메소드명의 접두사는 input을 사용한다.

ex) public void inputData(HashMap data){}

\* 데이터를 변경하는 메소드명의 접두사는 modify를 사용한다.

ex) public void modifyData(HashMap data){}

\* 데이터를 삭제하는 메소드명의 접두사는 delete를 사용한다.

ex) public void deleteData(String data){}

\* 데이터를 초기화 하는 메소드명의 접두사는 initialize을 사용한다.

ex) public void initData(String data){}

\* 반환값의 타입이 boolean인 메소드는 접두사로 is를 사용한다.

ex) public void isData(String Data){}

\* 데이터를 불러오는 메소드명의 접두사는 load를 사용한다.

ex) public void loadData(){}

\* 데이터가 있는지 확인하는 메소드명의 접두사는 has를 사용한다.

ex) public void hasData(){}

\* 보다 지능적인 set이 요구될때 사용하는 메소드명의 접두사는 register를 사용한다.

ex) public void registerAccount(){}

\* 새로운 객체를 만든 뒤 해당 객체를 리턴해주는 메소드명의 접두사는 create를 사용한다.

ex) public void createAccount(){}

\* 해당 객체를 다른 형태의 객체로 변환해주는 메소드명의 접두사는 to를 사용한다.

ex) public void toString(){}

\* 해당 객체가 복수인지 단일인지 구분하는 메서드명의 접미사는 s를 사용한다.

ex) public void getMembers(){}

\* B를 기준으로 A를 하겠다는 메소드명의 전치사는 By를 사용한다.

ex) public void getUserByName(String name){}

\* 반환값의 타입이 boolean인 메소드는 접두사로 is를 사용한다.

ex) public void isData(String Data){}

\* 데이터를 불러오는 메소드명의 접두사는 load를 사용한다.

ex) public void loadData(){}

\* 데이터가 있는지 확인하는 메소드명의 접두사는 has를 사용한다.

ex) public void hasData(){}

\* 보다 지능적인 set이 요구될때 사용하는 메소드명의 접두사는 register를 사용한다.

ex) public void registerAccount(){}

\* 새로운 객체를 만든뒤 해당 객체를 리턴해주는 메소드명의 접두사는 create를 사용한다.

ex) public void createAccount(){}

\* 해당 객체를 다른 형태의 객체로 변환해주는 메소드명의 접두사는 to를 사용한다.

ex) public void toString(){}

\* 해당 객체가 복수인지 단일인지 구분하는 메서드명의 접미사는 s를 사용한다.

ex) public void getMembers(){}

\* B를 기준으로 A를 하겠다는 메소드명의 전치사는 By를 사용한다.

ex) public void getUserByName(String name){}

**4. 변수(Variable) 명명 규칙**

\* 변수와 메소드의 파라미터에는 카멜표기법을 사용한다.

\* 변수에 약어를 사용하지 않고 모든 의미를 충분히 담는다.

\* 한 글자로 된 이름을 사용하지 않는다.

\* 선언된 지점에서 초기화하며, 가능한 사용범위를 최소화 한다. 숫자 0 레퍼런스 null

\* 반복문에서 인덱스로 사용할 변수는 i,j,k 등으로 사용한다.

ex) for(int i = 0; i < 10; i++){}

\* 지역변수와 멤버변수(전역변수)는 변수명 앞에 밑줄(\_)을 사용하여 구별한다.

\* boolean타입의 변수는 접두사로 is를 사용한다 ex) isCheck

출처: 둔둔이 개발일기 <https://m.blog.naver.com/reona7140/221306141987>

**\* 사람인**

사업자등록번호: corp\_code

**\* 피플앤잡**

개인회원: personal-register

기업회원: company-register

회사명: company-name

회사 소개: company-intro

사업자등록번호: biz-number

업종: biz-code

**\*잡코리아**

사업자등록번호: Corp-RegNum

회사명: Corp-Name

개인: M

회사: Co

#패키지, 클래스는 풀네임을 사용하고, 변수명에는 융통성 있게 축약형 사용하면 어떨지

공통

회사 company COM

개인 personal PER

회원 member

인증 certificate CERT

추천 recommend REC

게시판 board

게시물 post(예약어라면 posting?)

내 글 myPost 내 리뷰 myReview

제안 suggest

정보 info

자기소개서/이력서/내 정보 수정 edit letter/resume/myInfo

제목 title

내용 content

고객센터 customer center

메시지 message MSG

인재 찾기 searchPerson

회사 찾기 searchCompany

학력 education EDU

경력 career

자격증 license LIC

매칭율(점수) total rate

별점 star rate

(자소서/이력서 항목: 자격증, 경력, 해외경험, 병역, …)

DB

입사일 hire date

선호 키워드 prefer

채팅방 message room

보낸 시간 send time

읽은 시간 read time

읽은 여부 message check

업종 company type

설립일 establish date EST

사업자등록번호 company code

사업자등록증 company license (개인의 자격증과 겹치지 않을지?)

대표자 boss

담당자 incharge INC

직위 position POS

소재지 location LOC

입학/입사 날짜 begin date

졸업/퇴사 날짜 end date

급여 pay

복지 welfare

워라밸 balance

승진 promotion

경영진 boss