표기법

카멜 표기법

- 각 단어의 첫문자를 대문자로 표기하고 붙여쓰되, 맨처음 문자는 소문자로 표기함
- 띄어쓰기 대신 대문자로 단어를 구분하는 표기 방식
- 예시

```
backgroundColor, typeName, iPhone
```

파스칼 표기법

- 첫 단어를 대문자로 시작하는 표기법
- 예시

```
BackgroundColor, TypeName, PowerPoint
```

스네이크 표기법

- 단어를 밑줄문자(_)로 구분하는 표기법
- 예시

```
background_color, type_name
```

헝가리안 표기법

- 접두어를 사용하는 표기법
- 형식은 카멜 표기법과 같지만, 맨 앞의 단어가 자료형을 나타내는 접두어이다.
- 예시

```
strName, bBusy, szName
```

- 접두어 예시
-

접두어	의미
b	불리언(boolean)
ch	문자(char)
f	float
dz	NULL로 끝나는 문자열 (string + zero)

네이밍 규칙

네이밍은 평균 3개의 단어로 조합

- 변수는 짧게 네이밍
- 함수와 클래스는 조금 더 길게 네이밍해도 무방
-

네이밍 종류	평균 단어 길이
class 네이밍	3.18 단어
함수 네이밍	3.36 단어
변수 네이밍	2.57 단어

네이밍의 단어 연결 구조

- [명사|동사|형용사] + 명사 + 명사 + ...
- 첫 [명사|동사|형용사] 에서는 목적에 맞게 선택
- 그 이후로부터는 명사 조합

네임스페이스는 회사 표준 패턴에 따라 사용

- <com/org>.<회사명>.<제품명/프로젝트명>.<최상위 모듈>.<하위 모듈>

```
org.apache.spark.spark-core.~~.~~
```

패키지

- 패키지 이름은 소문자로 생성
- 사업이름.업무대분류.중분류.기능구분

```
kts.spring.board.dao
kts.spring.board.controller
kts.spring.board.service
kts.spring.board.repository
```

클래스

- 파스칼 표기법 사용

```
public class HelloWorld{
}
```

메소드

- 대소문자를 혼용할 수 있지만 반드시 동사를 사용하며 소문자로 시작

```
getName();
computeTotalWidth();
```

변수, 파라미터

- 카멜 표기법 사용

```
int totalCount=0
String workerName="kts"

public void WorkerLog(String workerName){};
```

변수에 모든 의미를 충분히 담아야 한다.

- 가능하면 모든 축약어를 사용하지 않고, 의미를 바로 알 수 있어야 한다.

```
String imageName="~";  ==>  String imgName="~";
String buttonName="~"; ==>  String btnName="~";
```

반복문을 제외하고 길이가 1인 문자로된 이름은 사용 x

```
int a=0; ==> (x)
```

Boolean 변수에는 is 접두어 사용

- 때에 따라 has, can, should 사용

```
boolean isOk = false;
boolean canEvaluate();
boolean hasNext();
boolean shouldAbort();
```

상수(final 변수)를 표한하는 이름은 반드시 모두 대문자로 지정, '_'를 사용하여 단어들을 구분

```
final String DB_NAME = "~~"
```

boolean 타입의 변수의 이름은 부정적인 이름을 사용하지 않는다.

```
boolean isNotError; (x)
boolean isError;    (o)

boolean isNotFound; (x)
boolean isFound;    (o)
```

대응하는 단어가 있는 이름은 반드시 함께 사용

```
get/set, add/remove, create/destroy, start/stop, insert/delete,
increment/decrement, old/new, begin/end, first/last, up/down, min/max,
next/previous, old/new, open/close, show/hide
```

n 접두사는 객체의 개수를 나타내는 변수에 사용

```
nLines, nPoints
```

compute는 계산(시간이 소요되는)하는 메소드에 사용

```
computeAverage();
computeInverse();
computeSum();
```

find는 무엇을 찾는 메소드에 사용

- by를 통해 기준을 설정할 수 있다.

```
findWorker();
findWorkerById(String id);
```

No 접미사는 Entitiy 번호를 나타내는 변수에 사용

```
int workerNo;
int employNo;
```

일반적인 변수의 이름은 타입의 이름과 동일하게 지정

```
void setTopic(Topic topic)
void setWorker(Worker worker)
void connect(Database database)
```

호출하려는 객체의 이름을 통해 메소드를 간략화 한다.

```
worker.getWorkerId();  --> (x)
worker.getId();        --> (o)
```

get/set 용어는 반드시 직접 객체의 속성에 직접 접근하는 메소드에 사용

```
worker.getId();
list.getElement(int index);

worker.setId(String id);
list.setElement(int index, Element element)
```

참고 사이트

- https://freehoon.tistory.com/55
- https://wikidocs.net/16995
- https://brunch.co.kr/@goodvc78/12