







# Convention

## Git Commit

- 깃모지를 사용한다. 다음 링크를 참고하여 깃모지 역할을 적절히 사용한다.

<b>Gitmoji 사용하기</b> gitmoji란? Gitmoji = git + emoji 입니다. 글을 쓸 때 이모지를 이용하면, 나중에 글을 읽을때 명확합니다. 🍌 커밋할 때도 이모지를 이용한다면, 내용을 한 눈에 알아보기 더 쉽겠죠. 그래서 나온 것이 gitmoji 입니다.  <a href="https://treasurebear.tistory.com/70">https://treasurebear.tistory.com/70</a>	 Move website to Next.js (#368)  Move website to Next.js (#368)  Move website to Next.js (#368)  Update LICENSE  Migrate from Travis-CI to GitHub Actions (#609)
---	---

- Commit 내용은 영어와 특수문자로만 작성한다.
- 첫 단어는 동사로 한다.
- 깃모지에서 한 칸 띄우고 대문자부터 시작한다.

## FE

### Vue.js

- 컴포넌트 이름에 합성어 사용  
Root 컴포넌트인 App 컴포넌트를 제외하고는 컴포넌트의 이름은 항상 합성어를 사용한다. 표기법은 카멜 표기법을 따른다.
- 컴포넌트 데이터  
컴포넌트 데이터는 무조건 객체를 리턴하는 함수여야 한다.
- Props 정의  
Props 정의는 가능한 자세히 기술한다.
- V-for에 key 지정  
v-for 사용시 항상 key를 지정한다. Key를 지정하는 경우 불필요한 업데이트를 방지할 수 있다.
- v-if와 v-for를 동시에 사용하지 않는다.  
v-for가 우선순위가 더 높기 때문에 원하지 않는 방향으로 실행될 수 있다. V-if 사용 시 v-for 대신 filter를 사용한다.
- 컴포넌트 스타일 스코프

App 컴포넌트와 layout 컴포넌트를 제외한 다른 모든 컴포넌트의 스타일은 항상 scoped여야 한다. 또한 div, p와 같은 요소 기반 css보다 class 기반 전략을 취하도록 한다.

## JavaScript

Js는 하단 BE의 Java – F 들여쓰기부터 J 공백까지 공유한다.

# BE

## Spring

- 컨트롤러 클래스 내 메서드 명 작성 시 아래와 같은 접두사를 붙인다.
  - List : 목록 조회
  - Detail : 상세 조회
  - Add : 등록
  - Modify : 수정
  - Delete : 삭제
- 서비스

서비스 클래스 내 메서드 명을 작성할 때에 아래와 같은 접두사를 붙인다.

- Get : 조회 유형의 service 메서드
- Modify : 변경 유형의 service 메서드
- Delete : 삭제 유형의 service 메서드

## Structure

- 패키지는 목적 별로 묶어 생성한다.
- controller에서는 service 호출과 exception 처리만을 담당한다.  
비즈니스 로직은 service에서 처리한다.
- 메서드와 클래스는 하나의 목적만을 위해 생성한다  
한 개의 메서드는 한가지 기능만을 취한다.  
한 개의 클래스 내부에는 같은 목적만을 가진 코드가 존재하여야 한다.

- 메서드와 클래스는 가능한 작게 만든다.  
목적이 뚜렷한 작은 클래스 여러 개로 이루어진 시스템이 바람직하다.
- 도메인명의 Service 생성을 피한다.  
Ex) Order라는 도메인 내에 OrderService로 만드는 것을 피한다. 가능한 세분화 하여 Service를 만든다.

## File

- 파일 인코딩은 utf-8로 통일한다.
- 새줄 문자는 LF
- 파일의 마지막에는 새줄을 사용한다.

## Variable

- 영문/숫자/언더스코어만 허용한다.
- 한국어 발음대로 표기하지 않는다.
- 대문자로 표기할 약어를 명시한다.
- 패키지 이름은 소문자로 구성한다.
- 클래스/인터페이스 이름에 대문자 카멜표기법을 적용한다.
- 클래스 이름에는 명사를 사용한다.
- 인터페이스 이름은 명사/형용사를 사용한다.
- 테스트 클래스는 “Test”로 끝난다.
- 메서드 이름에 소문자 카멜표기법을 적용한다.
- 메서드 이름은 동사/전치사로 시작한다.
- 상수는 대문자와 언더스코어로 구성한다.
- 변수에 소문자 카멜표기법을 적용한다.
- 임시 변수 외에는 한 글자 이름 사용을 금한다.

## 선언

- 소스 파일당 한 개의 탭 레벨 클래스를 담는다.
- Static import에만 와일드 카드를 허용한다.
- 제한자 선언 순서는 다음과 같다

```
public protected private abstract static final transient volatile synchronized native strictfp
```

- Annotation 선언 후에는 새 줄을 사용한다.
- 하나의 선언문에는 하나의 변수만 정의한다.
- 배열에서 대괄호는 타입 뒤에 선언한다.
- “long”형 값의 마지막에 대문자 ‘L’을 붙인다.
- 특수 문자의 전용 선언 방식을 활용한다. (`\012(x) \n(o)`)

## 들여쓰기

- Tab 문자를 사용하여 들여 쓴다. 탭 대신 스페이스를 사용하지 않는다.
- 탭의 크기는 스페이스 4개와 같다.
- 클래스, 메서드, 제어문 등의 코드 블록이 생길 때마다 한 단계 더 들여 쓴다.

## 중괄호

- K&R 스타일로 중괄호를 선언한다. 줄의 마지막에서 시작 중괄호 ‘{’를 열고, 블록을 마친 후 새 줄 삽입 후 ‘}’를 닫는다.
- 닫는 중괄호와 같은 줄에 `else`, `catch`, `finally`, `while`을 선언한다.
- 빈 블록에 새 줄 없이 중괄호 닫기는 허용한다.
- 조건, 반복문은 한 줄이더라도 중괄호를 필수 사용한다.

## 줄 바꿈

- 최대 줄 너비는 120으로 한다.
- `Package`, `import` 선언 시 중간에 줄을 바꾸지 않는다.
- 줄 바꿈 후에도 이어지는 줄에서는 최초 시작한 줄보다 최소 한 단계 들여 쓴 후 시작한다.
- 가독성을 위해 줄을 바꾸는 위치는 다음 중의 하나로 한다.
  - `extends`선언 후
  - `implements`선언 후
  - `throws`선언 후
  - 시작 소괄호(`()`) 선언 후

- 콤마(,) 후
- .전
- 연산자 전
  - +, -, \*, /, %
  - ==, !=, >=, >, <=, <, &&, ||
  - &, |, ^, >>, >>, <<, ?
  - instanceof

## 빈줄

- Package 선언 후 빈 줄 삽입
- Import 선언 후 빈 줄 삽입
- Import 선언 시 다음과 같은 순서로 그룹을 묶어 선언한다. 각 그룹 사이에는 빈 줄을 삽입한다. 그룹 내에서는 알파벳 순으로 정렬한다.
  1. static imports
  2. java
  3. javax
  4. org
  5. net
  6. com.\*
  7. 위를 제외한 클래스
- 메서드 사이에 빈 줄을 삽입한다.

## 공백

- 공백으로 줄을 끝내지 않는다
- 닫는 대괄호 뒤에 ';'로 문장이 끝나지 않고 다른 선언이 올 경우 공백을 삽입한다.
- 중괄호 시작 전과 종료 후 else가 있는 경우에 공백을 삽입한다.
- 제어문 키워드와 여는 소괄호 사이에 공백을 삽입한다.
- 식별자와 여는 소괄호 사이에 공백을 삽입하지 않는다.
- 타입 캐스팅에 쓰이는 소괄호 내부 공백은 사용하지 않는다.

- 콤마/구분자 세미콜론의 뒤에만 공백을 삽입한다.
- 콜론의 앞 뒤에 공백을 삽입한다.
- 이항/삼항 연산자의 앞 뒤에 공백을 삽입한다.
- 단항 연산자와 연산 대상 사이에 공백을 삽입한다.
- 주석문 기호 전후에 공백을 삽입한다.