SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

코딩 컨벤션 적용

계절학기

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Contents

Ⅰ 과제 안내

Ⅱ 코딩 컨벤션이란?

Ⅲ 코딩 컨벤션이 필요한 이유

Ⅳ 필요요소와 예시

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

과제 안내

- 코딩 컨벤션을 정리한 후, 1학기 관통프로젝트에 적용해 봅니다

  리드미 파일에 아래 내용을 정리해서 제출합니다

  - 코딩 컨벤션 학습 내용
  - 코딩 컨벤션 적용 전/후 소스 추가

코딩 컨벤션이란?

- 읽고, 관리하기 쉬운 코드를 작성하기 위한 일종의 코딩 스타일 규약

코딩 컨벤션이 필요한 이유

- 다수의 개발자가 각자의 스타일로 코드를 작성한다고 하면,
  남의 코드를 이해하는데 시간이 오래 걸리고..
  남이 작성한 코드에 대해 잘못 이해할 수 있다
  이러한 사유로 사전에 코드의 스타일을 통일함으로써 협업에 용이하다

코딩 컨벤션의 필요 요소

① 파일의 구성

② Naming Convention

③ 들여쓰기

④ 주석

⑤ 선언

⑥ 명령문

⑦ 공백

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

파일의 구성

파일의 구성

- 자바 소스 파일은 여러 section을 식별하는 주석으로 구성

  2천 줄 이상의 긴 파일로 구성되는 것을 지양

  소스 파일의 순서는 ..

  ① 주석 (Beginning Comments)

  ② 패키지 및 import문

  ③ 클래스 및 인터페이스

파일의 구성


```
/*

 * Classname
 *
 * Version information
 *
 * Date
 *
 * Copyright notice
 */


package java.blah;

import java.blah.blahdy.BlahBlah;

/**
 *
        Class description goes here.
 *
 * @version
        1.82 18 Mar 1999  * @author
        Firstname Lastname  */
public class Blah extends SomeClass {

            /* A class implementation comment can go here. */
    /**
        classVar1 documentation comment */
    public static int classVar1;

    /**
```

Beginning Comments

Package&import

Class 및 Interface

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Naming Convention

패키지는 소문자로

- 패키지 이름은 소문자로 작성한다

  단어의 구분을 위해 '_' 및 대문자를 혼용하지 않는다

패키지는 소문자로

- 패키지 이름은 소문자로 작성한다

  단어의 구분을 위해 '_' 및 대문자를 혼용하지 않는다

나쁜 예

```
package com.navercorp.apiGateway

package com.navercorp.api_gateway
```

패키지는 소문자로

☑ 패키지 이름은 소문자로 작성한다

단어의 구분을 위해 '_' 및 대문자를 혼용하지 않는다

나쁜 예

```
package com.navercorp.apiGateway

package com.navercorp.api_gateway
```

좋은 예

```
package com.navercorp.apigateway
```

클래스는 Camel Case

- 클래스 이름은 Camel Case를 사용한다

  합쳐지는 단어의 첫 글자를 대문자로 표기하는 것을 의미

클래스는 Camel Case

☑ 클래스 이름은 Camel Case를 사용한다

합쳐지는 단어의 첫 글자를 대문자로 표기하는 것을 의미

나쁜 예

```
public class reservation

public class Accesstoken
```

클래스는 Camel Case

☑ 클래스 이름은 Camel Case를 사용한다

합쳐지는 단어의 첫 글자를 대문자로 표기하는 것을 의미

나쁜 예

```
public class reservation

public class Accesstoken
```

좋은 예

```
public class Reservation

public class AccessToken
```

메소드는 동사로 시작

- 메소드 명은 기본적으로 동사로 시작하며..

  다른 타입으로 변환하는 메소드는 전치사로 시작할 수 있다

메소드는 동사로 시작

- 메소드 명은 기본적으로 동사로 시작하며..
  다른 타입으로 변환하는 메소드는 전치사로 시작할 수 있다

좋은 예

동사사용 : renderHtml()

변환메소드의 전치사: toString()

상수는 대문자

- 상수 이름은 대문자로 작성하며, 합성어는 '_'를 사용하여 단어를 구분한다

상수는 대문자

- 상수 이름은 대문자로 작성하며, 합성어는 '_'를 사용하여 단어를 구분한다

좋은 예)

```
public final int UNLIMITED = -1;
public final String POSTAL_CODE_EXPRESSION = "POST";
```

가독성 있는 변수명

- 임시 변수 외에는 1글자 변수 명 사용 금지
- 이해하기 쉬운 변수 명으로 작성한다

가독성 있는 변수명

- 임시 변수 외에는 1글자 변수 명 사용 금지
- 이해하기 쉬운 변수 명으로 작성한다

나쁜 예

```
HtmlParser p = new HtmlParser();
```

검색 시에도 비효율적

가독성 있는 변수명

- 임시 변수 외에는 1글자 변수 명 사용 금지
- 이해하기 쉬운 변수 명으로 작성한다

나쁜 예

```
HtmlParser p = new HtmlParser();
```

좋은 예

```
HtmlParser parser = new HtmlParser();
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

선 언

클래스 등의 소스 구성요소를 선언할 때 고려해야 할 규칙

클래스 import 시에는..

- 클래스 import시에는 와일드카드(*)없이 필요한 클래스 명을 명시적으로 작성

클래스 import 시에는..

- 클래스 import시에는 와일드카드(*)없이 필요한 클래스 명을 명시적으로 작성

나쁜 예

```
import java.util.*;
```

성능에 영향을 줄 수 있다

클래스 import 시에는..

- 클래스 import시에는 와일드카드(*)없이 필요한 클래스 명을 명시적으로 작성

나쁜 예

```
import java.util.*;
```

좋은 예

```
import java.util.List;
import java.util.ArrayList;
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

하나의 선언문에는 하나의 변수

- 변수 선언문은 한 문장에서 하나의 변수만을 다룬다

  주석 사용을 위해 한 줄에 하나씩 선언하는 것을 권장한다

하나의 선언문에는 하나의 변수

- 변수 선언문은 한 문장에서 하나의 변수만을 다룬다

  주석 사용을 위해 한 줄에 하나씩 선언하는 것을 권장한다

나쁜 예

```
int base, weight;
```

하나의 선언문에는 하나의 변수

- 변수 선언문은 한 문장에서 하나의 변수만을 다룬다

  주석 사용을 위해 한 줄에 하나씩 선언하는 것을 권장한다

나쁜 예

```
int base, weight;
```

좋은 예

```
int level; // indentation level
int size;  // size of table
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

들 여 쓰 기

코드의 계층을 구분하기 위해 추가

들여쓰기

- 탭을 사용하여 들여쓴다 (스페이스를 사용하지 않음)

  1개의 탭 사이즈는 4개의 스페이스와 같도록 에디터에서 설정

  클래스, 메소드, 제어문 등의 block이 생길 때마다 1단계 들여쓰기

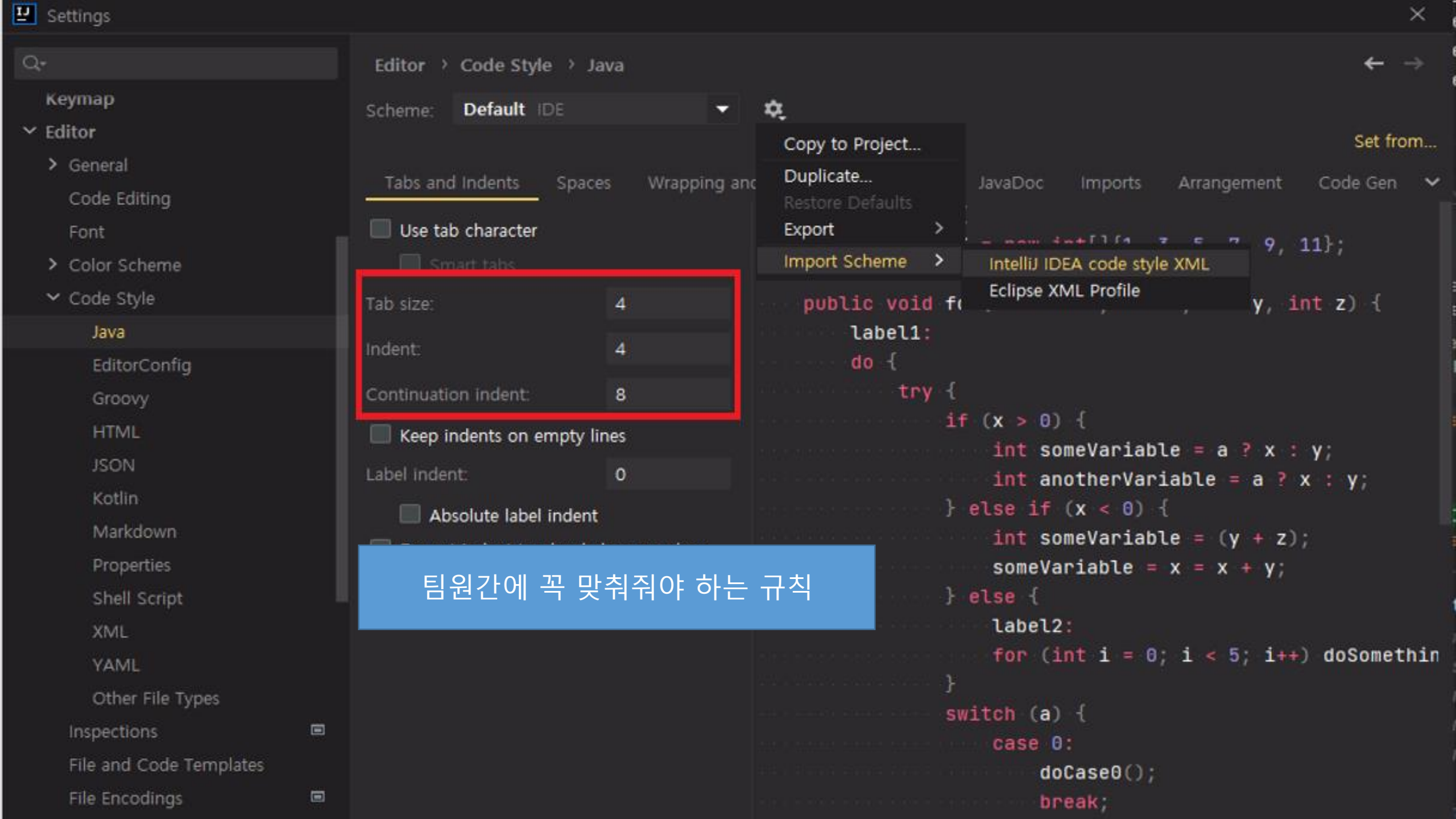
구글 코드 컨벤션, IDE 적용

- 구글에서 xml파일을 제공해주고 있어 IDE에 설정해두면 편하게 적용가능

  “구글 깃허브”에서 필요한 언어와 IDE환경에 맞는 파일을 다운로드

  (자바와 인텔리제이 조합 : intellij-java-google-style.xml)

  File -> Setting -> Editor -> Code Style에서 설정


Settings
Keymap
Editor
General
Code Editing
Font
Color Scheme
Code Style
Java
EditorConfig
Groovy
HTML
JSON
Kotlin
Markdown
Properties
Shell Script
XML
YAML
Other File Types
Inspections
File and Code Templates
File Encodings
Editor › Code Style › Java
Scheme: Default IDE
Copy to Project...
Duplicate...
Restore Defaults
Export
Import Scheme
IntelliJ IDEA code style XML
Eclipse XML Profile
Set from...
Tabs and Indents
Spaces
Wrapping and
JavaDoc
Imports
Arrangement
Code Gen
Use tab character
Smart tabs
Tab size: 4
Indent: 4
Continuation indent: 8
Keep indents on empty lines
Label indent: 0
Absolute label indent
팀원간에 꼭 맞춰줘야 하는 규칙
public void f
y, int z) {
label1:
do {
try {
if (x > 0) {
int someVariable = a ? x : y;
int anotherVariable = a ? x : y;
} else if (x < 0) {
int someVariable = (y + z);
someVariable = x = x + y;
} else {
label2:
for (int i = 0; i < 5; i++) doSomethin
}
switch (a) {
case 0:
doCase0();
break;

참고

- Kotlin Coding Convention
- Google Java Style Guide
- Google TypeScript Style Guide
- NHN 코딩컨벤션
- Java Code Convention - Oracle

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

SUMMARY

- 코딩 컨벤션에 100% 정답은 없습니다

  많은 컨벤션 가이드 中 ,

  무엇을 선택할지 보다는 일관된 기준으로 소스코드를 작성하는 것이 가장 중요

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

여러분의
폭풍 성장을 응원합니다!!