Web プログラミング
コーディング規約

麻生情報ビジネス専門学校
情報システム専攻科

目次

1. ファイルについて

- **ファイルの文字コード**

ファイルの文字コードはすべての種類において UTF-8 とする

- **ファイル名について**

命名規則の項を参照のこと

2. フォーマット

● インデント

・インデントは空白 4 文字分とする（tab 可）
・中かっこ開始（ { ）の次の行は必ず字下げする。
・中かっこの開始行と終了業のインデントは必ず同じ位置になるようにする

例）

【規約違反の例】

```
    if( parm == 1 ) {
    param1 = 10;
    param2 = 10;
    }
```

if 文の中は字下げすべきなのに下がっていないので NG

```
public class Main{
public void function(){
}
}
```

Main の次の行が字下げされていないので NG

【正しい例】

```
    if( parm == 1 ) {
        param1 = 10;
        param2 = 10;
    }
```

if 文の中が字下げされているので OK

```
public class Main{
    public void function(){
    }
}
```

Main の次の行が字下げされているので OK

例）

【規約違反の例】

```
    if( parm == 1 ) {
        param1 = 10;
        param2 = 10;
}
```

if 文の開始のカッコと終了のカッコのインデントが合っていないので NG

```
public class Main{
    public void function(){
}
}
```

メソッドの開始のカッコと終了のカッコのインデントが合っていないので NG

【正しい例】

```
    if( parm == 1 ) {
        param1 = 10;
        param2 = 10;
    }
```

if 文の開始のカッコと終了のカッコのインデントが合っているので OK

```
public class Main{
    public void function(){
    }
}
```

メソッドの開始のカッコと終了のカッコのインデントが合っているので OK

・if や for を記載するときは、処理が 1 行でも必ず中カッコをつける

● 中かっこ

・中かっこの開始({) の前には改行を入れない

例)

【規約違反の例】

```
if( parm == 1 )
{
    //do something
}
```

※中かっこの開始の前に改行があるので NG

【正しい例】

```
if( parm == 1 ) {
    //do something
}
```

※中かっこの開始の前に改行がないので OK

```
if( parm == 1 ) {
    //do something
} else if( param == 2 ){
```

※中かっこの終了の前に改行がないが else if なので OK

● 1メソッドの長さについて

1メソッドの長さは100行（コメント、空白行を除く）を目処とする。
100行を超える場合は、処理をメソッドに分けて行うことを検討すること。
※100行を数えるのが困難な場合は、画面をスクロールせずにメソッドの始まりと終わりが確認できる長さに収める

● 1命令の長さについて

1行の長さが画面に収まらないほど長い場合は、改行を入れて見やすくすること。

例）
【規約違反の例】

TaskBoImpl.java　JavaCProgramJudge.java

```
90          for( TestcaseTableEntity testcase : testCaseSet){
91              ////////////////////////////////////////
92              //シェルの実行（コンパイルと実行と品質解析）
93              execShell(testcase,dirName,fileName,resultDir,classDir);
94
95              ////////////////////////////////////////
96              //エラー情報をチェック
97              String erroInfo = getCompileErrorInfo(resultDir);
98              compileError = StringUtils.isNoneEmpty(erroInfo);
99
100             ////////////////////////////////////////
101             //正解かどうかのチェック
102             boolean result = checkAnswer(testcase,resultDir);
103
104             ////////////////////////////////////////
105             //結果をEntityを登録
106             resultEntity.addResultTestcaseTbl(getResultTestcaseTblEntity(testcase,result,compileError,resultDir,err
107         }
108
109         ////////////////////////////////////////
110         //ソースの品質情報をパース
111         resultEntity.addResultMetricsTbl(getResultMetricsTblEntity(resultDir,fileName));
112
113         ////////////////////////////////////////
114         //総合得点を計算
115         resultEntity.setTotalScore( getTotalScore(resultEntity) );
116
117         ////////////////////////////////////////
118         //課題提出フラグと提出日時をセット
119         json.allOK = isAllOK(resultEntity);
120         resultEntity.setHanded((json.allOK==true?1:0));
121         resultEntity.setHandedTimestamp((json.allOK==true?now:null));
122
123         ////////////////////////////////////////
124         //ソースファイルの内容を圧縮してセット
```

106行目が画面からはみ出ていて見辛い

【正しい例】

TaskBoImpl.java　*JavaCProgramJudge.java

```
//テストケース一つ一つに実行し、点数を集計
for( TestcaseTableEntity testcase : testCaseSet){
    ////////////////////////////////////////
    //シェルの実行（コンパイルと実行と品質解析）
    execShell(testcase,dirName,fileName,resultDir,classDir);

    ////////////////////////////////////////
    //エラー情報をチェック
    String erroInfo = getCompileErrorInfo(resultDir);
    compileError = StringUtils.isNoneEmpty(erroInfo);

    ////////////////////////////////////////
    //正解かどうかのチェック
    boolean result = checkAnswer(testcase,resultDir);

    ////////////////////////////////////////
    //結果をEntityを登録
    resultEntity.addResultTestcaseTbl(
            getResultTestcaseTblEntity(testcase,result,compileError,resultDir,erroInfo));
}

////////////////////////////////////////
//ソースの品質情報をパース
resultEntity.addResultMetricsTbl(getResultMetricsTblEntity(resultDir,fileName));

////////////////////////////////////////
//総合得点を計算
resultEntity.setTotalScore( getTotalScore(resultEntity) );

////////////////////////////////////////
//課題提出フラグと提出日時をセット
json.allOK = isAllOK(resultEntity);
resultEntity.setHanded((json.allOK==true?1:0));
resultEntity.setHandedTimestamp((json.allOK==true?now:null));

////////////////////////////////////////
```

改行を入れて見やすく修正した例

3. 命名規則

● ファイル名

・.java ファイルはクラス名と同じファイル名とする（大文字小文字も同じにする）
例）Main クラスは Main.java とする（main.java は NG）

・.jsp ファイルは全て小文字とし、区切りにはアンダーバーを使用する
例）会員の登録画面の JSP ファイルは entry_member.jsp とする）

・その他のファイル（xml など）は全て小文字とし、区切りにはアンダーバーを使用する

● クラス名

・先頭文字は大文字とし、文節は大文字で表す
例）会員登録のクラス
〇EntryMember
×entryMember
×Entry_Member（アンダーバーで区切らない）

・サーブレットは語尾に Servlet をつける
例）
〇LoginServlet
×Login

・Dao クラスの語尾には Dao をつける
例）
〇UserDao
×UserTbl

・Filter クラスの語尾には Filter をつける
例）
〇EncodeFilter
×Encode

・自作の例外クラスには Exception をつける
例）
〇AsoSystemException

×AsoSystemError

● メソッド名

・先頭文字は小文字にし、文節は大文字にする

・動詞＋名詞の順で名前を付ける

例）リストを取得するメソッドの場合

○getList

×GetList（先頭が小文字ではない）

×get_List（アンダーバーで区切らない）

×listGet（名詞＋動詞の順になっている）

・動詞はメソッドの動きが想像できるものにすること

以下の項目を参考にするとよい

メソッドの動き	メソッド名（XXX 部分は名詞が入る）	例
取得	getXXX	getList
設定	setXXX	setUserName
true か false を返す	isXXX	isStudent
検索	findXXX （getXXX でも可。プログラムの流れ上わかりやすい方を採用する）	findUser
作成	createXXX	createUserObject
削除	deleteXXX	deleteUserObject
更新	updateXXX	updateUser
変換	toXXX parse	toString
比較	equal match comp プログラムの流れによって使い分ける	
エラーチェックを行う	validateXXX	validateUserName

● 変数名

・先頭文字は小文字にし、文節は大文字にする
・文字数は 15 文字以内とする
（15 文字を超える場合は意味のある省略形をつける）
・インスタンスはクラス名の先頭小文字にしたものにする
（String のインスタンスは例外とする）
・意味のない変数名はつけない（for 文のループカウンタは例外）

例）太字が変数名
〇UserInfoBeans **userInfoBeans** = null;
×UserInfoBeans **user_Info_Beans** = null;（アンダーバーで区切らない）
×UserInfoBeans **param** = null;（クラス名と異なる変数名）
×UserInfoBeans **a** = null;（意味のない変数名なので NG）
〇String name = “Nishino”;　（クラス名とことなるが String クラスなので OK）
〇UserInfoBeans **userInfoBeans** = null;

● パッケージ名

・全て小文字とし、文節には記号などは使わない
・役割事（サーブレット、DAO、モデルなど）に適切なパッケージを作成する

例）
〇websample.servlet
×WebServlet.servlet（大文字が含まれているので NG）
×aaaa.bbb.ccc　（パッケージ名から役割が連想できないので NG）

● 定数名

マジックナンバーなどを避ける意味でも定数（enum 含む）を使用することを推奨する。
定数は以下にしたがって定義すること

・名前は全て大文字とし、区切り文字はアンダーバーとする
・宣言にはかならず final をつける
・public な定数については static をつける
・名前から何の定数かが想像できる名前を付ける

例）

○private final int MAX_INING = 9;

○public final static int MAX_PLAYER = 9;

×private int MAX_ININ G = 9;　（final がついていないので NG）

×public final int MAX_ININ G = 9;（public なのに static がついていないので NG）

×private final int max_ining = 9;（小文字なので NG）

×private final int M = 9;（名前から何の定数かがわからないので NG）

4. 禁止事項

- **大文字小文字で変数を区別しない**

同じスコープ内で変数のスペルが同じで大文字小文字だけで区別してはいけない

例)

```
public GatchaBeans getRandom(LoginInfoBeans loginBeans){
    int param = 0;
    int Param = 1;
}
```

param と Param で別の変数として扱われるが紛らわしいため NG

- **if , else , while , for , do while の中カッコは省略しない**

例)

```
if( loginBeans == null )
    return null;
```

return null は loginBeans が null の時だけ実行される。if 文の中カッコが焼灼されているので NG

5. 参考

・Java コーディング規約（Future Enterprise Coding Standards）
https://future-architect.github.io/coding-standards/documents/forJava/Java%E3%82%B3%E3%83%BC%E3%83%87%E3%82%A3%E3%83%B3%E3%82%B0%E8%A6%8F%E7%B4%84.html

・Google Java Style Guide(非公式和訳)
https://kazurof.github.io/GoogleJavaStyle-ja/

6. 更新履歴

修正日	修正箇所	修正内容	担当者
2018/10/30	-	新規作成	西野