Javaプログラミング講座

javaファイルの書式

・パッケージ宣言: package フォルダ.フォルダ…;

・インポート宣言: import フォルダ.フォルダ…;

・クラス(またはインターフェース, 列挙型)の定義

パッケージ: プログラムを分類して管理するための仕組み。

(実質的にはフォルダ(ファイルシステム)として管理)

パッケージ宣言: 定義したクラスが、どのパッケージに属するものなのかを宣言する。

インポート宣言: 異なるパッケージに属するクラスを使用したい場合に指定する。

Mainクラスの定義:

public class クラス名 {

public static void main(String[] args) {

// 処理

}

}

基本的にこれをいじっていく。

式(文)

・定義文(宣言, 初期化[定義式+代入式としても])

修飾子 型 変数名;

・代入式

変数名 = インスタンス/値;

・初期化式

修飾子 型 変数名 = インスタンス/値;

・処理式(命令, 実行, 演算)

(クラス.)関数(メソッド)名;

・ブロック文(複文): 複数の処理が一つの文として書ける。

{ //処理 }

・条件式[boolean](論理, 判定, 評価)

boolean型を返すメンバ

・if-else文

if (条件式) { //処理 }

else if (条件式) { //処理 }

…

else { //処理 }

・switch文

switch (変数\*) {

case 値:

//処理

break;

…

default: //処理

}

\*: 整数型, 列挙型, 文字列型

・while文

while (条件式) { //処理 }

・do-while文

do { //処理 } while (条件式);

・for文

for (定義式; 条件式; 代入式) { //処理 }

・for-each文

for (型 変数名 : 変数\*) { //処理 }

\*: Iterableを実装している型の変数

・try-catch文: 例外処理用

・try-catch-resouce文: I/O用

==インスタンス==

インスタンス: クラスがメモリ上に展開されたもの。

インスタンス化(クラス生成, 初期化):インスタンスを生成すること。下記の文でできる。

型 変数 = new クラス名(引数);

null: インスタンスがないことを表す値。

static: 別記する。

コンストラクター: クラスのインスタンス化が行われる際に呼ばれ、その際に必要な処理を記述するメソッド。

修飾子 class クラス名 {

修飾子 クラス名(引数) {

// 処理

}

}

==修飾子==

public: 全てのクラスから見える

private: 同一クラスの中からしか見えない(他のクラスから見られない)

protected: 同一クラス、およびその継承関係にあるクラス(子クラス)からしか見えない

→基本的にはprivateと言われているが、プロジェクト規模にもよるので一概には言えない。

詳細: http://java-code.jp/134

==Collection==

Collection: 要素のオブジェクトを管理(格納、取得)することを可能にするAPI

・List(リスト)

配列のように格納される要素が順番を持つ。

順番に並んでいるため、リストの要素はidex(インデックス)で参照したり、

追加したりすることができる。

Array(配列)との違い:

要素数が確定していなくてもできる。

・インデックスで管理

・要素は重複が可能

・Set(セット)

格納される要素に順番も重複もない。

・要素は順不同

・要素は重複不可

・Map(マップ)

・Key(キー)とValue(値)のペアで管理

・Keyは重複不可

・Queue(キュー)

・待ち行列として管理

・先頭または末尾の要素のみ追加、削除可能

Iterator(イテレーター, 反復子):

・Collectionの要素を1個ずつ処理するためのクラス

・Iterableインターフェースを実装していなければならない

・拡張for文はこれがあると使える

Generics(ジェネリクス):

・汎用的な型を使うための機能

・<任意の型> ←これのこと

Stream(ストリーム):

・配列とCollectionを元にして、そのオブジェクトを集合体として扱えるようにしたAPI

・ラムダ式(関数型インターフェース)を使って、楽に配列とCollectionを操作できる。

==明示的な書き方==

空白

改行

インデント

処理順

==オブジェクト指向==

・クラス

・メンバ

・フィールド

・メソッド

・オブジェクト

(・インスタンス: クラス, オブジェクト)

クラス関係:

・継承([extends])

class:

・クラスの定義の拡張。

・単一継承。

・親クラスのメソッド、フィールドを利用できる。

・親クラスとほぼ同一として扱われる。(キャスト)

・実装(abstract[extends], interface[implements])

abstract(抽象):

classとほぼ同様だが、抽象である。

・クラスのメソッド、フィールドの実装を定義しない。

・abstract(抽象)クラスは、インスタンス生成できず、

継承クラスで実装されなければならない。

interface(インターフェース):

・機能の宣言。

・クラスのメソッドの実装を定義しない。(フィールドはできる？)

・多重継承可能。

==拡張for構文==

for (型※1 変数※2 : 変数※3) {

//処理

}

※3: Iteratableを実装したクラス(Collection関係のクラス)

※2: 変数※3の内容が順次入る

※1: 変数※3の内部の型

==forEachメソッド==

Java8で追加された関数型を受けて追加されたもので、便利。

変数※1.forEach((型 変数) -> {

任意の関数(変数);

})

1. 変数の型の省略
2. 変数部分の()の省略 (※変数が一個の時)
3. 複文(ブロック文)表記の省略 (※処理が一行の時)
4. メソッド参照 (※変数と、任意の関数の引数が対応している時?)

詳細: Javaラムダ式メモ(Hishidama's Java8 Lambda Expression Memo) –

<http://www.ne.jp/asahi/hishidama/home/tech/java/lambda.html>

==static(静的)==

static: インスタンス生成しなくても、そのメソッドや変数にアクセスできるようになる。

しかし、全てのインスタンスに影響を及ぼす。

詳細: <https://java-reference.com/java_basic_static.html>

おすすめサイト: <https://manga.crocro.com/?cat=java&pg=index> (漫画でわかりやすくて触りには最高)