OPEN STREAM

214-07.ProjectCoin

Make IT your Success

目次

バイナリリテラル（2進数リテラル）
数値リテラルのアンダースコア区切り
Strings in switch
Type Inference（型推論）
Try with resources
マルチキャッチ、再スロー

この講義について

目的：

- Java7から追加された、使いやすくなった文法について学ぶ。

ゴール：

- Project Coinの利点を理解し、適切に運用することが出来る。

バイナリリテラル（2進数リテラル）

接頭辞に“0b”, “0B”をつける事で、2進数リテラルを扱える

例

```
byte b = 0b1010101; // 85
int i = 0b1001;  //9
```

数値リテラルのアンダースコア区切り

数値リテラルに”_(アンダースコア)”を入れられる

例

```
int i = 123_456_789;
long l = 0xB2_0C_F8_A1_05_12L;
```

※先頭や末尾は、数値リテラルではなく変数として扱われるのでエラーになる。

※小数点の前後や、接頭辞”0x” 、接尾辞”F”, “L” の前後もエラーとなる。

Strings in switch

switch文のcaseラベルに文字列を使える

例文

```
switch (str) {
    case "a":
        break;
    case "b":
        break;
    case "c":
        break;
    default:
        break;
}
```

Type Inference（型推論）

インスタンス生成時、型推論される

例文

```
List<String> list = new ArrayList<String>();
Map<String, List<String>> map = new HashMap<String, List<String>>();
```

```
List<String> list = new ArrayList<>();
Map<String, List<String>> map = new HashMap<>();
```

※“＜＞”をダイヤモンド演算子と呼ぶ
※定義とインスタンス生成を別々に書ける
※匿名クラスでは＜＞は使えない
※引数に＜＞は使えない

Try with resources

Try catch文のリソースの開放を自動化する

例文

```
try (Connection conn = ds.getConnection()) {
   // DB処理
} catch (SQLException e) {
   // 例外処理
}
```

インタフェース Pjava.lang.AutoCloseable は close()メソッドを持っている。
AutoCloseableを実装しているクラスはtry with resourceステートメントを使う事ができる。
tryブロックを実行し終えるとclose()メソッドがリソースを解放するために呼び出される。
※クローズされる順番は定義順の逆。
※finallyの前にクローズ処理が行われる。

マルチキャッチ、再スロー

複数の例外エラーをcatchする

また、catchした例外を再スローできる

例文

```
try {
        // 処理
} catch (NamingException | IOException e) {
        log(e);
        throw e;
}
```

マルチキャッチ文は、同じ例外処理を行なうバイトコードにコンパイルされる