Jクラスレポート
ユーザーガイド

Ver 8.2

2015 年 11 月 15 日

ネクストデザイン有限会社

内容

変更履歴

- 2015 年 11 月 15 日 有効ステップ数の定義を変更

旧：Javadoc 行数＋Java ステップ数 → 新：Java ステップ数

はじめに

本書では、J クラスレポートのインストール、操作、作成されるドキュメントの内容についてご説明します

1. 動作条件

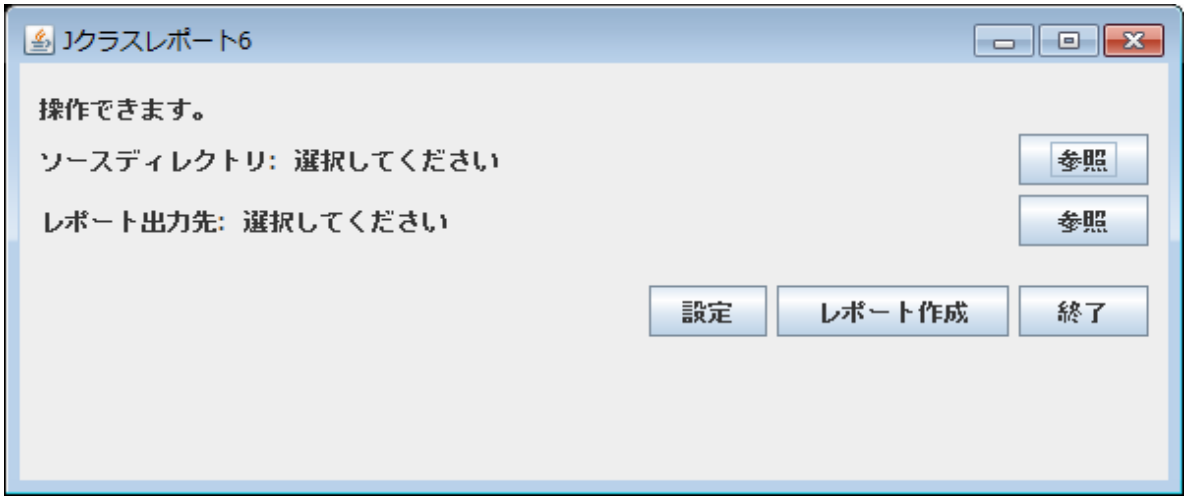
Java Version 6 以上が必要です。
出荷前の動作確認は次の通りです。
Windows 7 Professional SP1
Java SE 6
Microsoft Excel 2010（ドキュメントを開いて見る場合に必要です）

2. インストール

ダウンロードしたファイル（jp-co-nextdesign-jclassreport.zip）を適当な場所に解凍してください。

3. 起動

jp-co-nextdesign-jclassreport¥report.bat を起動（ダブルクリック）してください。
次の画面が表示されます。



4. ドキュメント作成

次の3つを行ってください。

4.1 ソースディレクトリの指定

上の参照ボタンを押してソースディレクトリを指定してください。
ソースディレクトリとは、パッケージディレクトリや Java ソースファイルを収容している一番上位のディレクトリです。

4.2 出力先ディレクトリの指定

このディレクトリ直下にすべてのドキュメントファイルが出力されます。

4.3 レポート作成を開始

レポート作成ボタンを押します。

5. 設定

Jクラスレポート画面の「設定」ボタンを押すと以下の画面が表示されます。

5.1 警告メソッド行数

プロジェクトレポートの「指標」シートの「最大ステップ数/クラス」がこの値を超えるとそのセルが黄色になります。

5.2 警告ネストレベル

プロジェクトレポートの「指標」シートの「最大ネストレベル/クラス」がこの値を超えるとそのセルが黄色になります。

5.3 Javaソースの文字コード

java.nio.charset.Charset で使用できる文字セット名を指定できます。UTF-8 と SJIS では動作確認済みです。

5.4 表紙のシステム名

プロジェクトレポートの「表紙」シートに出力されます。

5.5 表示の会社名

各ドキュメントの「表紙」シートに出力されます。

6. 作成されるドキュメント

6.1 ドキュメントの種類

作成されるドキュメントは次の 2 種類です。

(1) プロジェクトレポート

(2) クラスレポート

6.2 プロジェクトレポート

このドキュメントは、プロジェクトについて 1 つ作成されます。

ファイル名： ＠プロジェクトレポート.xlsx

6.3 クラスレポート

このドキュメントは、トップレベルの要素毎に 1 つ作成されます。

トップレベル要素とは、コンパイル単位（Java ソースファイル）の最上位に宣言された要素です。public またはデフォルト（修飾子なし）のクラス、アノテーション、列挙型です。内部クラス、無名クラス、別のクラス内に宣言された列挙型などについては作成されません。

ファイル名： クラスレポート_xxx_xxx_xxx.xlsx

ただし、xxx_xxx_xxx 部分はパッケージ、クラス名（アノテーション名、列挙型名）です。

7. プロジェクトレポートの項目説明

7.1 「指標」シート

7.1.1 パッケージ名

パッケージ名です。

7.1.2 サブパッケージ数

直下のサブパッケージの数です。

7.1.3 所属クラス数

直下のクラスの数です。

7.1.4 public

非 public の場合は「no」と出力されます。

7.1.5 abstract / interface

abstract の場合は「a」、interface の場合は「i」と出力されます。

7.1.6 タイプ名

トップレベルのクラス名、列挙型名、アノテーション名、内部クラス名、無名クラス(「無名 nn」)です。

nn は親クラス内での通し番号です。

7.1.7 総ステップ数

ソースファイルの行数です。空行を含みます。

テキストエディタで開いたとき単純な行数になります。

7.1.8 有効ステップ数

Java ステートメント行数です。

ステップ数はソースコードを抽象構文木構造化した状態でカウントしますので、テキストエディタで表示したときの行数とは異なる場合があります。

抽象構文木構造化については後述の例をご覧ください。

※もとのソースファイルの内容が整形・変更されることはありません。元のソースは何も変わりません。

7.1.9 Javadoc 行数

Javadoc の行数です。妥当ではない場所に書かれた Javadoc もカウントします。

7.1.10 他のコメント行数

ブロックコメント、ラインコメントの行数です。

7.1.11 属性数/クラス

インスタンス属性の数です。

7.1.12 メソッド数/クラス

コンストラクタを除くメソッドの数です。

7.1.13 コンストラクタ数

コンストラクタの数です。

7.1.14 内部クラス数/クラス

直下に含まれている内部クラスの数です。

7.1.15 無名クラス数/クラス

直下に含まれている無名クラスの数です。

7.1.16 列挙型数/クラス

親の直下に定義された列挙型の数です。トップレベルで定義された列挙型の数は含まれません。

7.1.17 初期化子数/クラス

直下に含まれているないぶくらす初期化子の数です。

7.1.18 最大ステップ数/メソッド

コンストラクタを含む最大のメソッドの行数です。

7.1.19 最大ネストレベル数/メソッド

メソッドのネストレベルは abstract の場合は0です。ボディが存在する場合の最小値は1です。

7.1.20 最大制御文数/メソッド

以下のステートメントの数です。

break

continue

do

for

if

switch

synchronized

while

7.2「依存」シート

Jクラスレポートでは、依存関係を導出手順は、以下の通りです。

(1) 属性型、new 式、スーパークラス、実装インタフェース、メソッド引数型、メソッド戻り値型の何れかとして使用した型の所属先を依存先としています。

(2) メソッド呼出しによる依存関係は抽出していません。

(3) import 文に指定されたパッケージを単純に依存先とはしていません。

8. クラスレポートの項目説明

8.1「参照・被参照」シート

8.1.1 参照関係を抽出しているもの

(1) スーパークラスの型

(2) 実装インタフェースの型

(3) インスタンス属性の型

(4) メソッドの戻り値型

(5) メソッドの引数型

(6) new 式の型

8.1.2 参照関係を抽出していないもの

(1) ローカル変数の型

(2) メソッド呼出し先の型

9. 構文木構造の例

弊社ホームページにも構文木構造にパースされた後の例があります。

(原コード)

```
if (boolValue) { System.out.println("boolValue is true."); }
```

(構文木構造にパース後)

```
if (boolValue) {
System.out.println("boolValue is true.");
}
```

(原コード)

```
@override
```

public String toString() {

（構文木構造にパース後）

@override public String toString() {

（原コード）

enum 名前 { 識別子 1, 識別子 2, 識別子 3, 識別子 4; }

（構文木構造にパース後）

enum 名前 {
識別子 1, 識別子 2, 識別子 3, 識別子 4;
}

10. アンインストール

解凍したファイルをファイルエクスプローラで削除します。

11. こんなとき

11.1 文字化けする

Jクラスレポートの読込み文字コードを対象 Java ソースの文字コードに合わせてください。
（操作） メイン画面→設定ボタン→プロパティ設定画面→Javaソースの文字コード

以上