3-9.パッケージ

提出する 評定を受ける

■ パッケージ

はじめに

パッケージとは、クラス（.class）ファイルの集まりを示すものです。
パッケージを使用することで複数のクラス（.class）ファイルを分類し、管理することができます。

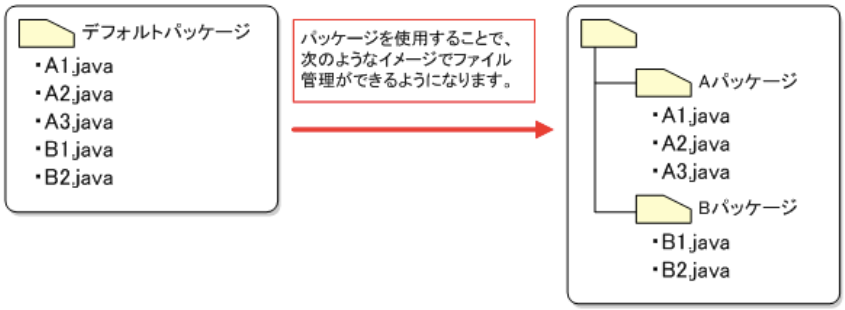
数多くクラスを作るとそのうちクラス名が重複することがあります。
その場合、 **クラス名は同名でもパッケージ名で区別することが可能** です。

またこれにより、クラス（.class）ファイルは「パッケージ名＋ファイル名」のように
名前空間（Namespace）が与えられて、名前の一意性も高まります。

では実際に見ていきましょう！

Step1: 概念を知る

【サンプルイメージ】



Javaはその仕様上、クラス（.class）ファイルを必ずどこかのパッケージに収める必要があります。
その為、パッケージを明示していない（いままでのような）クラス（.class）ファイルは
「デフォルトパッケージ」という形で取り扱っています。

上記と重複する内容もありますが、Javaにおいては以下の概念が存在します！

- 名前空間（Namespace）
- パッケージ（package）

> **名前空間（Namespace）**

「対象のものを指定した名称で確実に識別・特定するための概念」

コンピュータは人間には無いすごい能力持ってますが、 **使う側が適切に命令してあげないと上手く機能してくれません。**

例えば町中で、「鈴木さん！」と呼びかけたとしても、日本には沢山の鈴木さんがいます。
特定の鈴木さんを呼んだつもりでも、町中の沢山の鈴木さんが振り返ってしまいます。

?

でも、 **本来は一人だけを意図して呼んでいる（はず）** ですよね？
名前空間は、こういったことが起きないような役割を担っている訳です！

パッケージ（package）

「名前空間の中にあるJavaクラスをまとめる仕組み」

- 【名前空間＝パッケージ(package)】

鈴木 class を10から100まで扱わなきゃいけないとなっても、
パッケージ名（名前空間）がしっかりと区別できていれば問題ありません！

【パッケージ名（名前空間）】

- 一郎
- 二郎
- 三郎

アナタは（実際のJavaファイルで）以下のように指定して呼んであげれば、
意図した動作（振り向き笑）＋ 鈴木一郎さんだけにしか訊けないこと などを得られます！

```
package 鈴木;
```

ファイル的な階層だと下記の感じ。
（「 .（ピリオド or ドット）」で繋いでいきますね

```
// 鈴木.一郎（会社のことを色々教えてくれる処理
// 鈴木.二郎（スキルチェックとか見てくれる処理
// 鈴木.三郎（phpとかjavaとかレビューしまくっちゃう処理
```

という感じの例えをイメージしてもらった上で、以下の内容を見ていきましょう！

Step2: 使い方を知る

パッケージの指定の方法

パッケージに含めるファイルでソースの先頭部分に書きます。

```
package パッケージ名;
```

パッケージ内のクラスおよびインタフェースの呼び出し

```
パッケージ名.クラス名
パッケージ名.インタフェース名
```

補足

・インターフェース
簡潔に言えば、 **処理の実装されていないメソッドを束ねるクラスのようなもの** です。
「処理の実装されていない」というのは、 { }（ブロック） 内の処理が未実装 ということです。

【サンプル】

定義の方法は以下のようになります。

```
public interface インターフェース名 {
    // 処理の実装されていないメソッド
    public void sample();
}
```

※詳細は **インターフェース・抽象クラス・ポリモーフィズム** で解説します。

例

company/Worker.java

```
package company;

public class Worker {
    public String getName() {
        return "companyの従業員です。";
    }
}
```

Step3: インポート

インポートとは、他のパッケージに記入されているクラスや関数を実行するための方法です。
プログラミングしているとき、よく他のクラスを利用してクラスのインスタンスを生成します。

```
import [パッケージ名].[クラス名];
import [パッケージ名].*;
```

> **使い方**

```
// javaパッケージの中のlangパッケージの中のMathクラスをimport
import java.lang.Math;

public class StaticClass {
    public static void staticMethod(){

        int num1 = 10;
        int num2 = 20;

        // importしたMathクラスのmaxメソッドを使用
        int nmax = Math.max(num1, num2);
        System.out.println("nmax : " + nmax);
    }
}
```

【出力結果】

```
nmax : 20
```

Step4: パッケージの作成

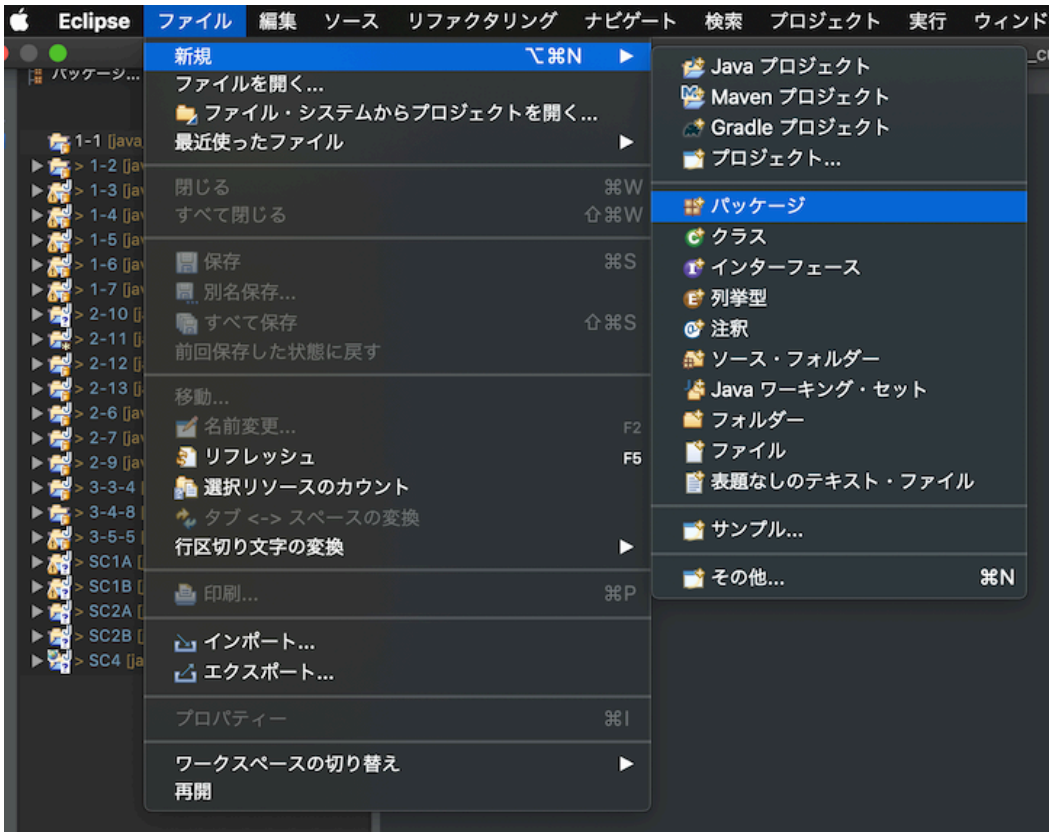
次は作り方になります。

まずは手順を提示していきますので、順に見ていきましょう！

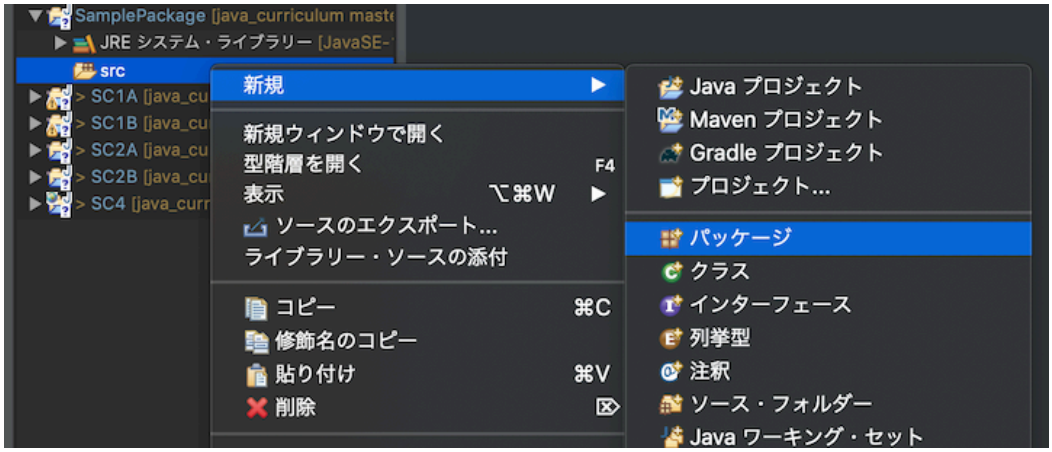
> **作成手順**

事前に適当なプロジェクトを作成しておきます。
今回はプロジェクト名を「 **SamplePackage** 」とします。

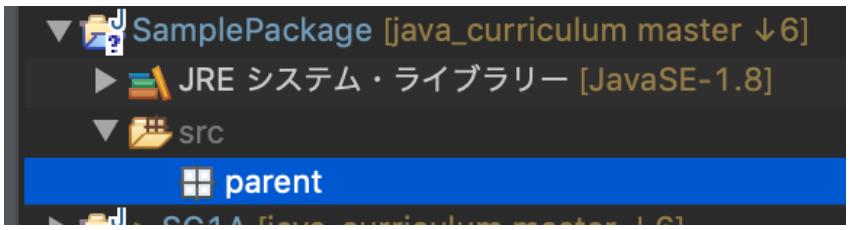
> 【1】プロジェクト直下の **src** よりメニューを開き、 [新規] → [パッケージ] を選択



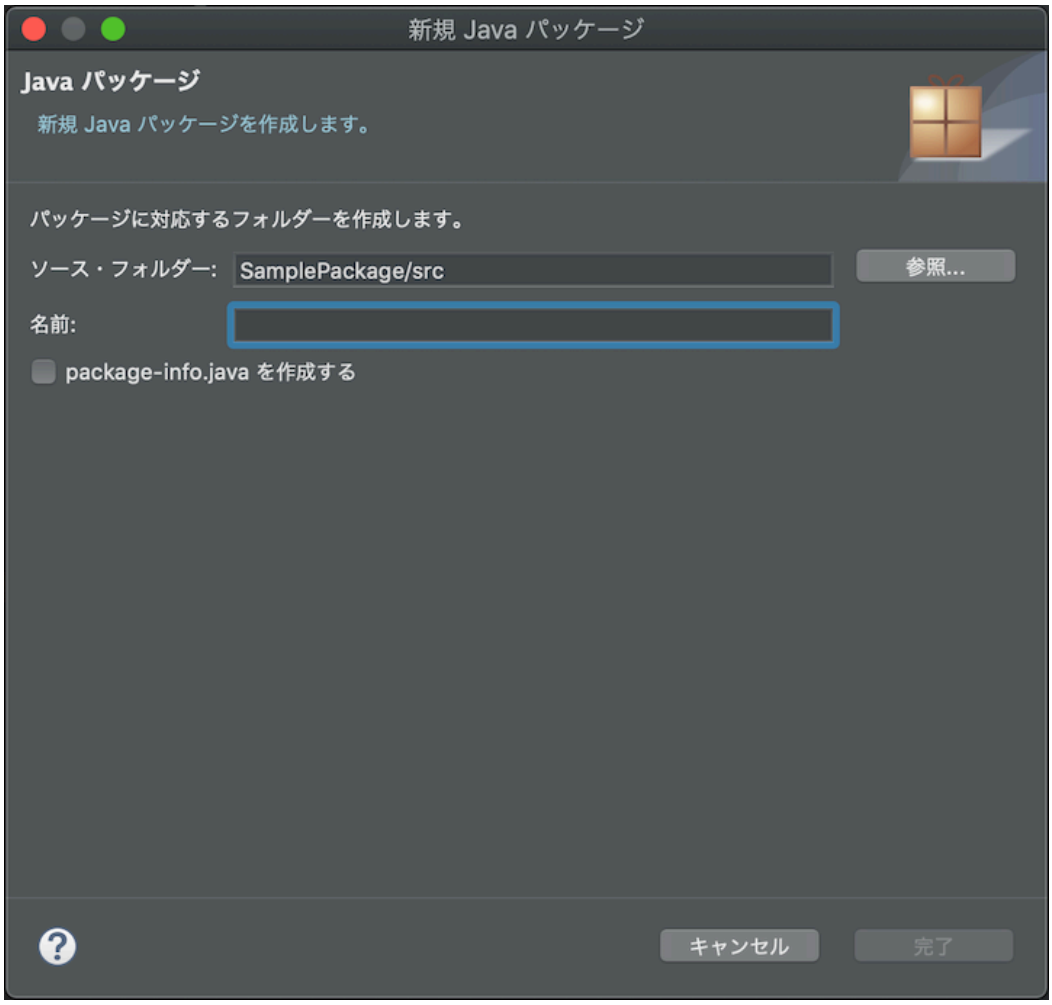
【1】Eclipse TOPメニューからでもパッケージの作成は可能です



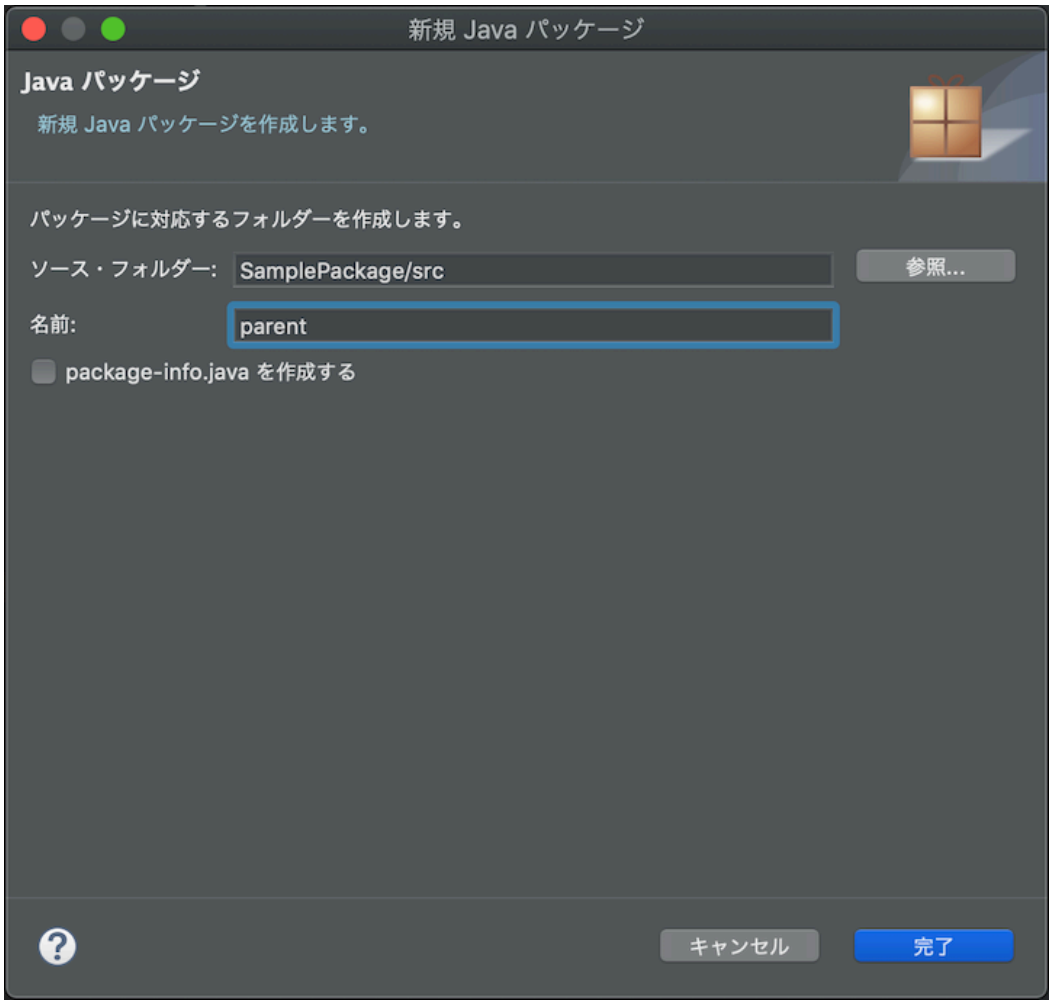
【2】パッケージ名を入力



【3】今回は「 **parent** 」 とします。



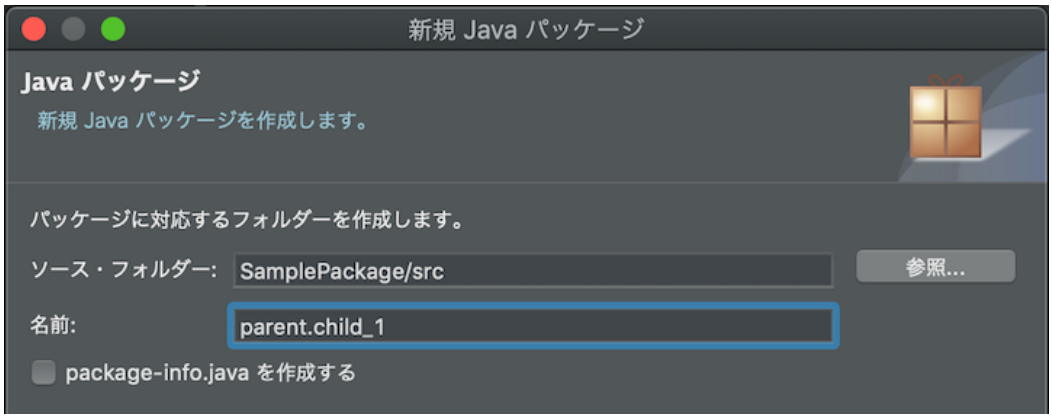
【4】作成後は、このように白い小包のようなアイコンが表示されます。



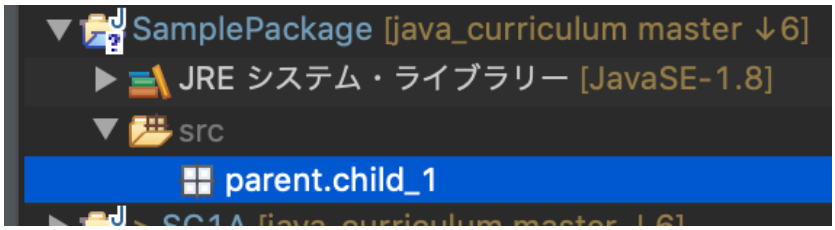
これが基本の作成方法になります。

続いて、パッケージの直下にパッケージを作成していきます。

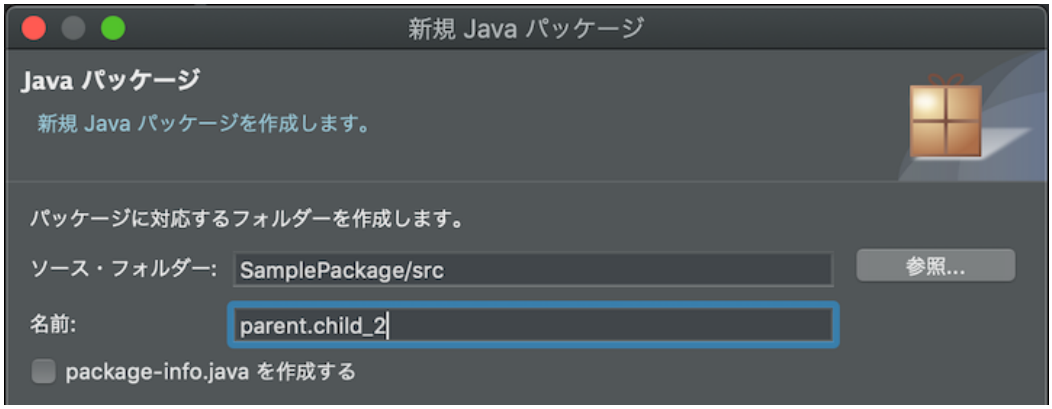
【5】「 **parent** 」 の子要素を作成していきます。パッケージ名は「 **child_1** 」とします。



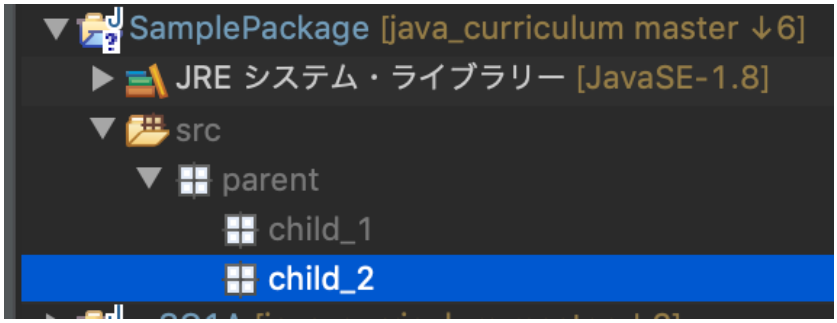
【6】「親.子」のようになりました。



【7】同様に、パッケージ名「 **child_2** 」を作成します。



【8】1行の「 .（ピリオド） 」で繋がれていたものが、階層構造になりました。



ポイント

・パッケージはカテゴリ
一つひとつ意味のある命名にすることはマストですが、大枠の考え方は「 **大分類 > 中分類 > 小分類** 」のような、
ディレクトリを作成するイメージです！

・階層構造
1つの親に対し、2つ以上の子パッケージが存在する場合は、Eclipseが見た目上の構造を階層構造へ切り替えて表示してくれます。

プロジェクトのファイル（javaファイルなど）は、
基本的にはパッケージ直下に作成して管理していきます。
グルーピングすることにより、 **作業効率とメンテナンス性の向上** を図ります！

今後プロジェクトの作成時にはパッケージを作成してもらう機会も出てくるので、
今のうちにしっかりと作成方法も身に着けましょう！

課題

インポートした3-9のフォルダの中にプロジェクトがありますので、
指示通りにコーディングして、ファイルを提出して下さい。

評定概要

学生から秘匿	No
参加者	75
提出	50
要評定	0