Java

オブジェクト指向とクラスの基礎

16時間目

プログラミング言語の種類

解説　プログラミング言語には、「プログラミングパラダイム」と呼ばれる種類があります。

1. 手続き型プログラミング
2. オブジェクト指向プログラミング

・・・などなど

補足　プログラミングパラダイムとは、プログラムを記述する際の考え方の種類です。
その他のプログラミングパラダイムでは、関数型プログラミング（例：Scala）も最近人気が高まっています。

手続き型プログラミングとは

1 手続き型プログラミング

言語	特徴
C言語、BASIC、perl など	１行目から最後の行まで、基本的に**個別に機能を作成し順番通り**に実行する

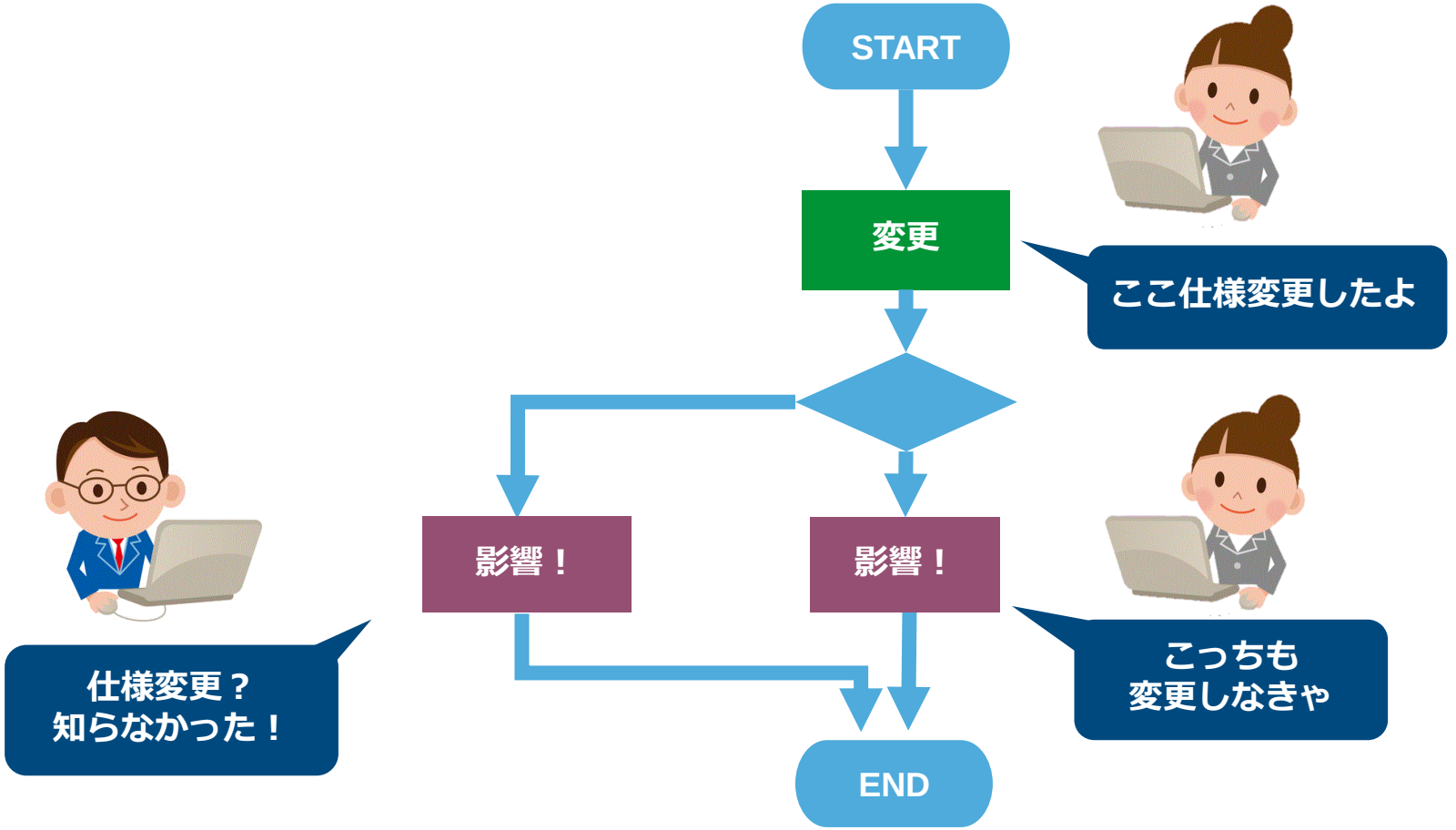
2 オブジェクト指向プログラミング

言語	特徴
Java、Ruby、PHP など	機能をオブジェクトと呼ばれる「データと動きのワンセット」単位で１つの部品として分解し、その**設計図を何度も流用**しながらも独立したパーツを作成・**組み合わせて**実行する。

手続き型プログラミング

解説

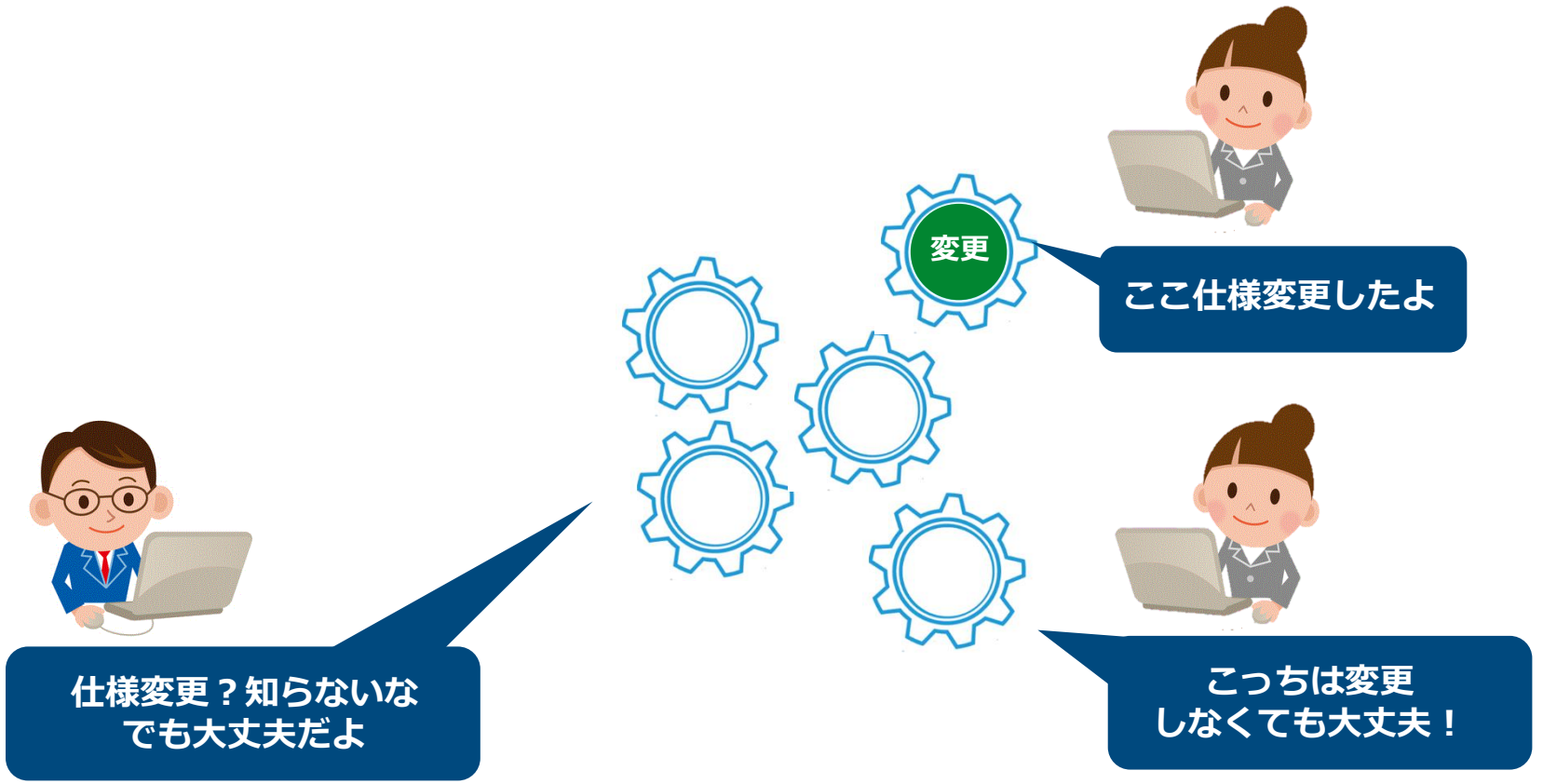
従来の手続き型では、各部品が完全には独立していないため、大勢で開発する場合に連絡ミスなどの混乱が生じたり、仕様変更への対応が広範囲になるなど大規模開発への対応が難しかった。



オブジェクト指向プログラミングのメリット

解説

「部品」に特化したオブジェクト指向を採用することによって、 全体（他の人の作業内容）までしっかり把握しなくても部品が作れたり、仕様変更の影響が出にくくなった。



クラスとメソッド

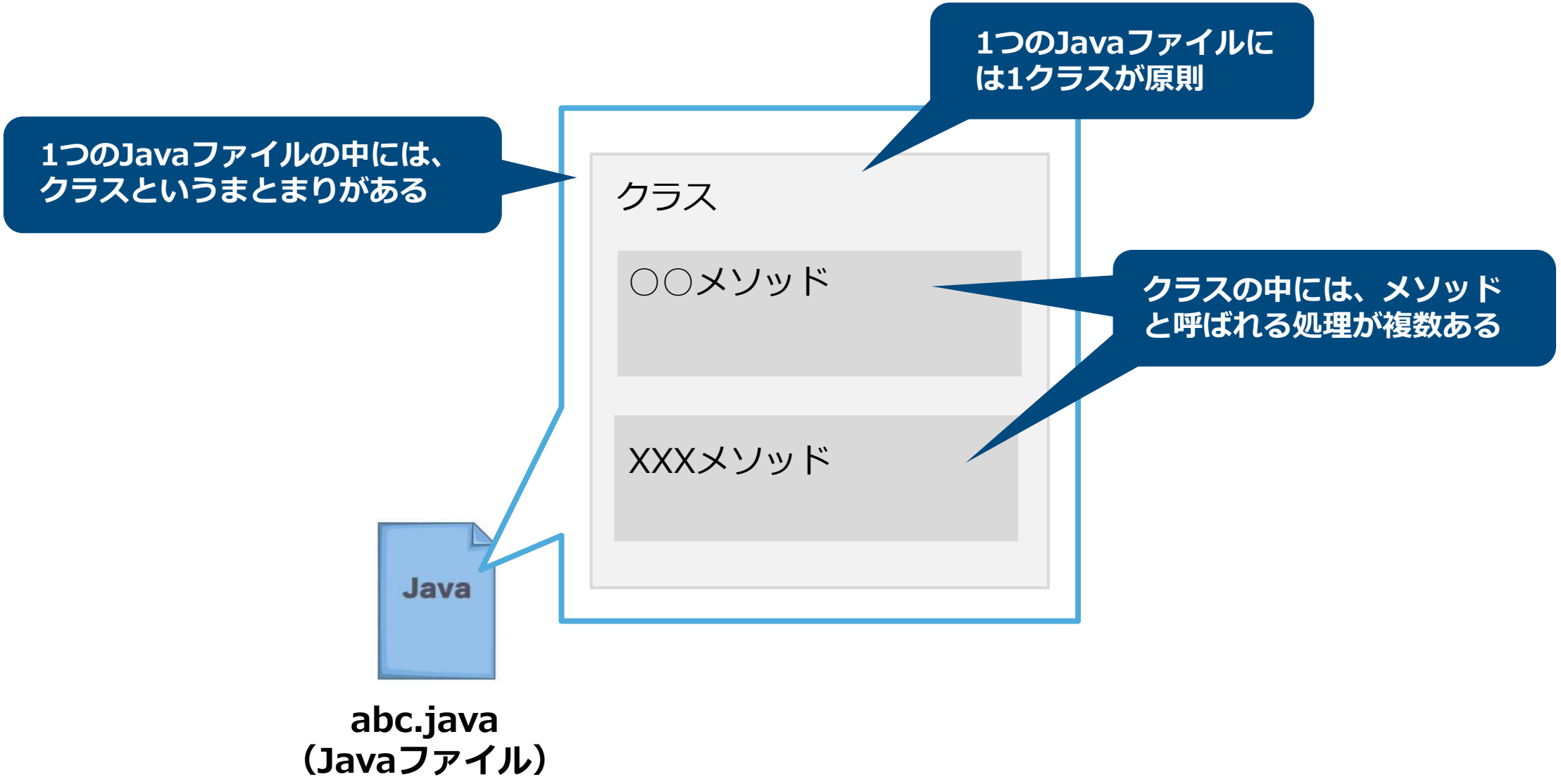
Javaプログラミングには、下記のルールがある

1. 1つのJavaファイルの中には、クラスという”まとまり”がある
2. 1つのJavaファイルには1クラスが原則
3. クラスの中には、メソッドと呼ばれる処理が複数ある

補足

厳密に言うと若干意味は異なりますが、現場では、Javaファイル=クラス=Javaファイル=クラスファイルという呼び方をすることがあります。

Javaファイルとクラスとメソッドのイメージ



クラスとメソッドの解説

Javaファイルの中には、必ずpublicというクラスが1つ存在する。publicクラスという文字は、eclipseを使用してJavaを記述する際に、デフォルト（初期状態）で必ず自動で出てくるモノ。

public クラスの後ろにつく○○○部分が（=自動的で表示される）、クラス名。

この場合『Humanクラス』と言う。

このクラス名と、ファイル名は同じにしなきゃダメ。

```
Public class Human {

    public static void main (String[] args)  {

    }
}
```

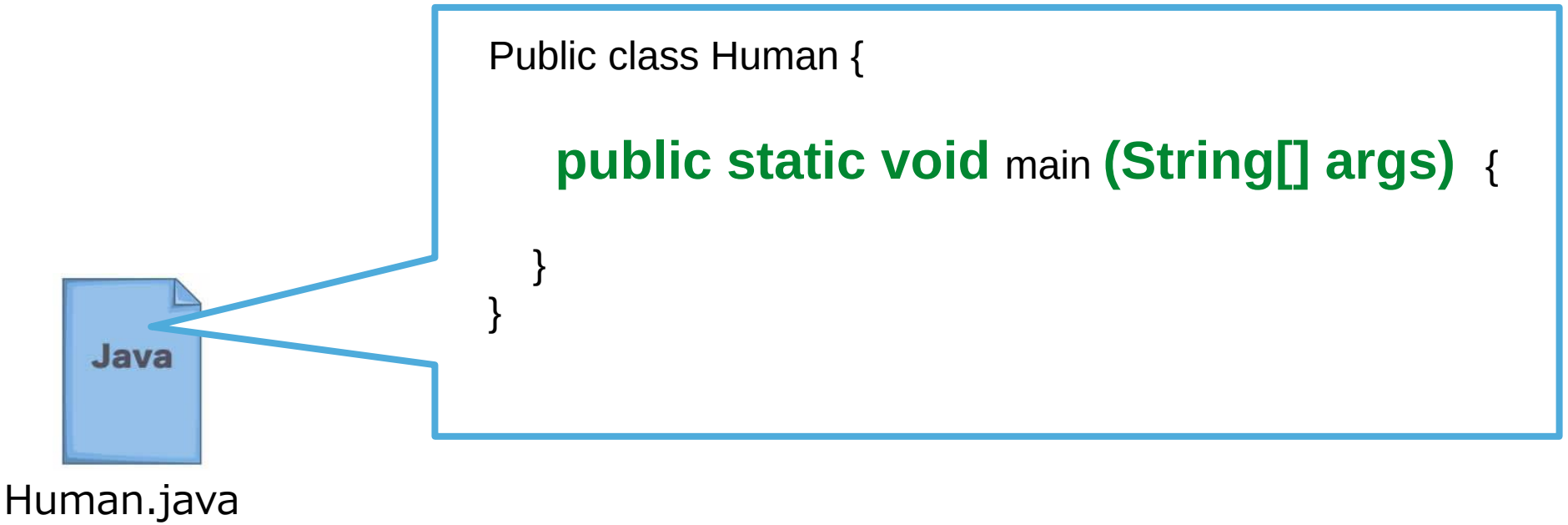
Human.java

緑の部分がメソッドとよばれる部分。メソッドとは、クラスを起動した時に実行したい処理の事。（任意の名前を付けられる）

メソッド名を「main」にすると、クラスを起動したときに、一番最初に処理されることになる。

static void、(String[]args)の意味は理解しないでOK

下記の**緑色**の部分は、現時点では**覚えなくてOK**
（無視して進めましょう）



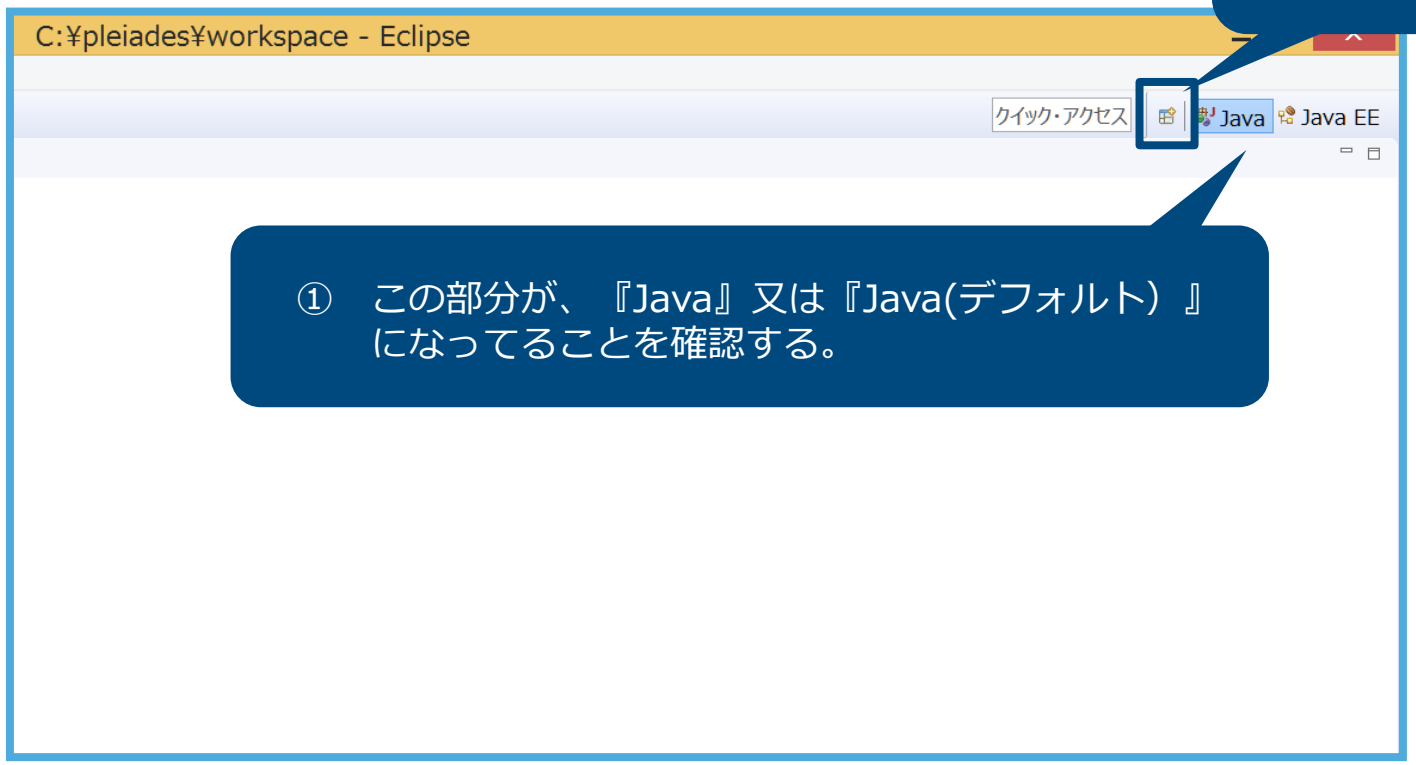
補足

staticはプログラミング用語で静的という意味、voidは処理を返す（returnを記述）してはダメという意味、String[]argsは○○○と言う意味ですが、プログラミングを始めて1～2年は理解する必要はありません。

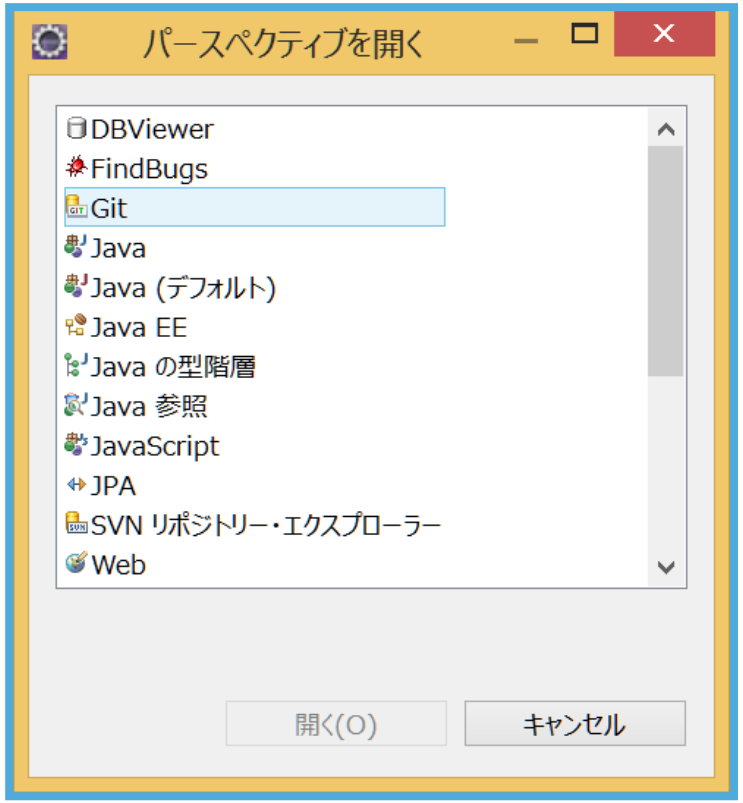
クラスやメソッドを意識してJavaを書いてみよう

Java、又は Java(デフォルト)を選択

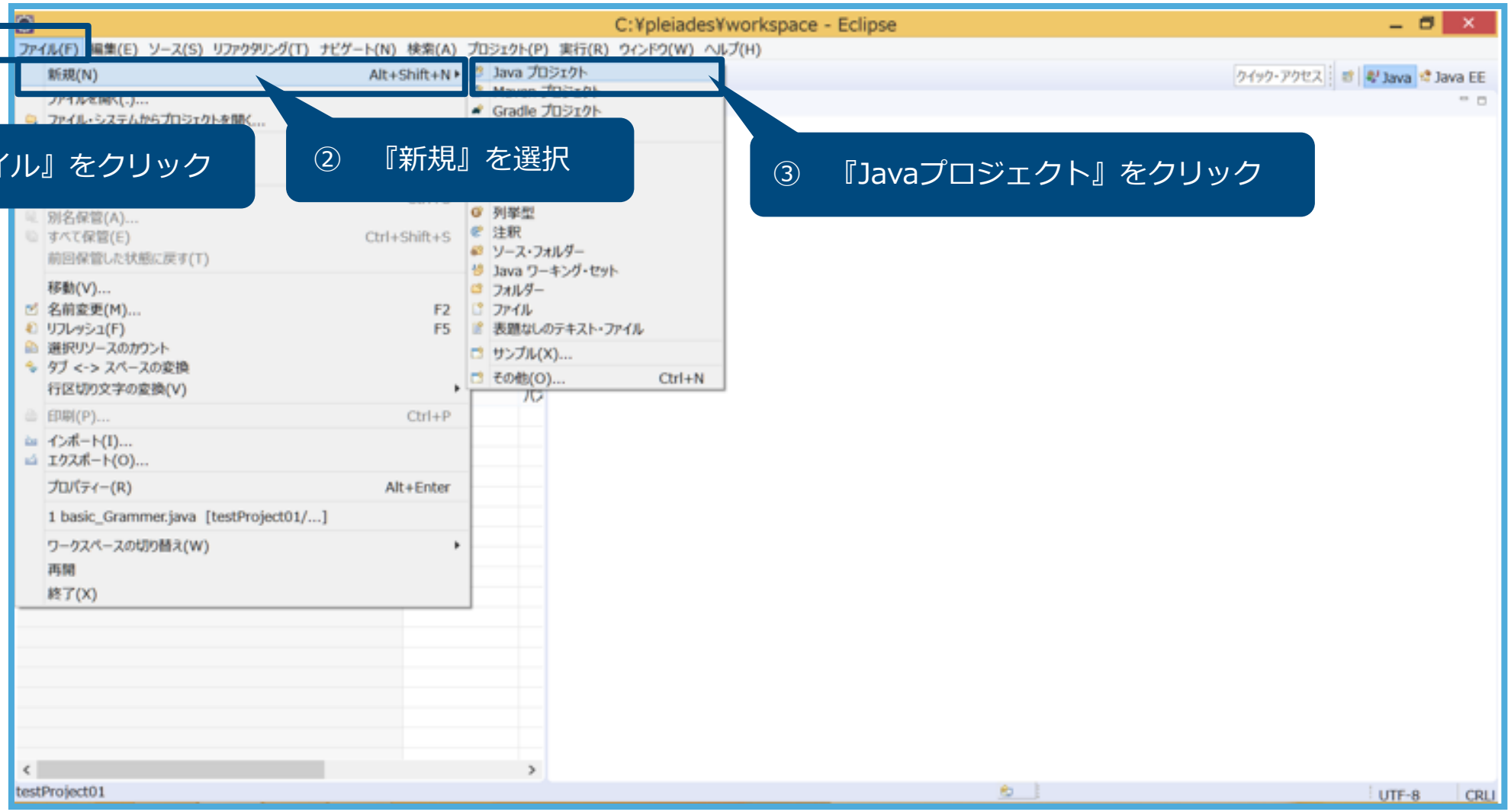
② なっていない場合は、ここのボタンをクリック。
すると、下のような画面が開くので、
『Java』又は『Java(デフォルト）』を選択する。



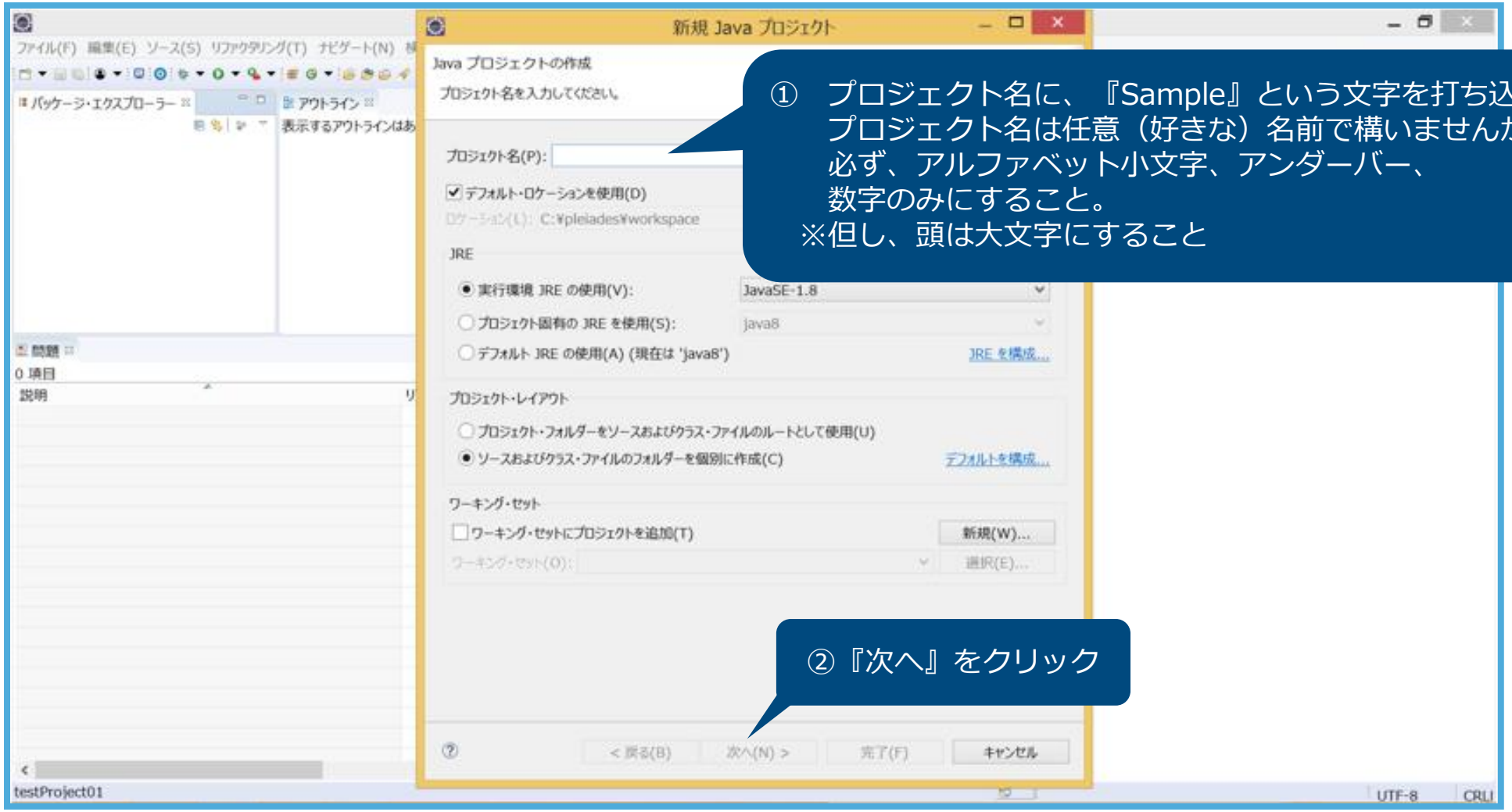
① この部分が、『Java』又は『Java(デフォルト）』
になってることを確認する。



新規でプロジェクトを作成



新規でプロジェクトを作成



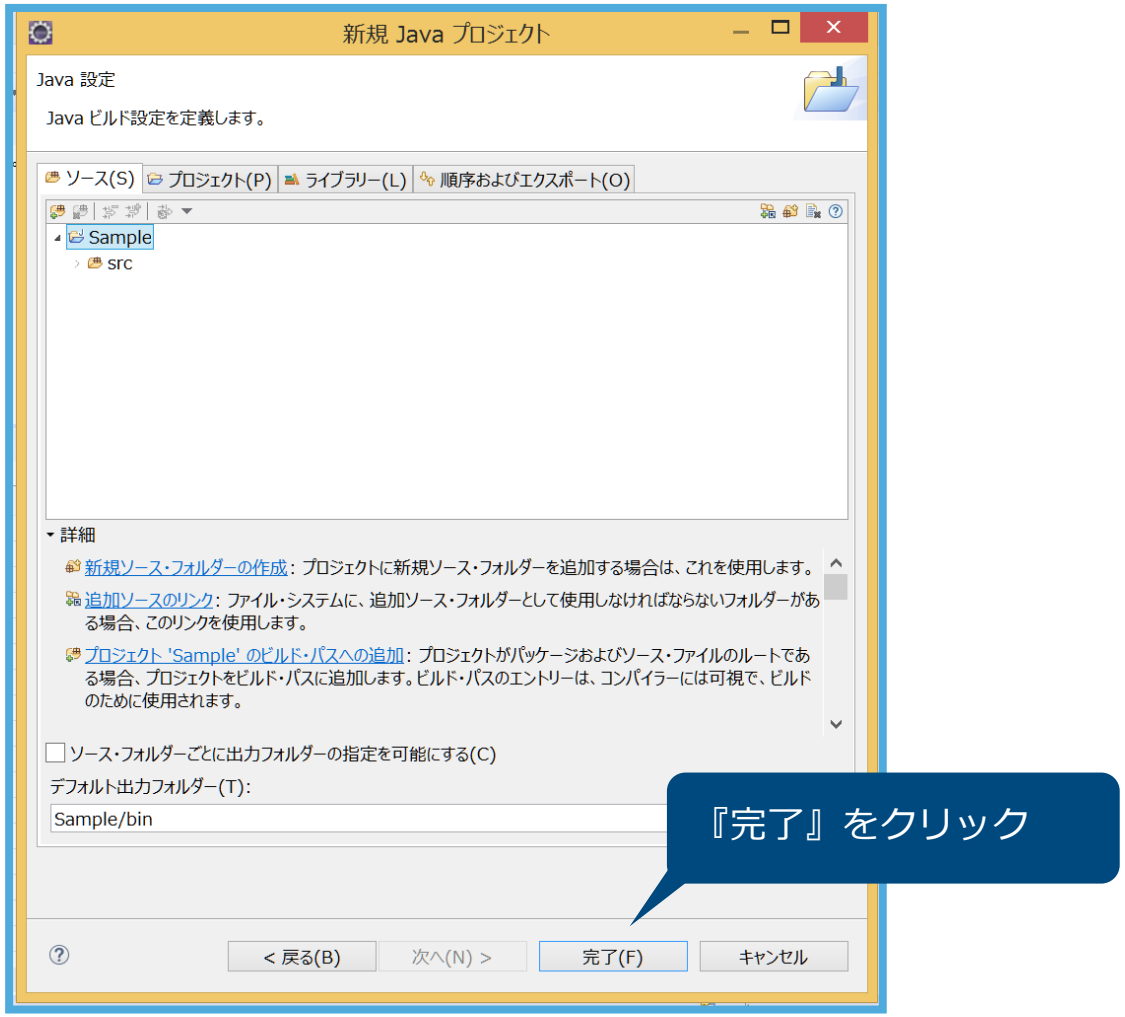
① プロジェクト名に、『Sample』という文字を打ち込む。
プロジェクト名は任意（好きな）名前で構いませんが、
必ず、アルファベット小文字、アンダーバー、
数字のみにすること。
※但し、頭は大文字にすること

②『次へ』をクリック

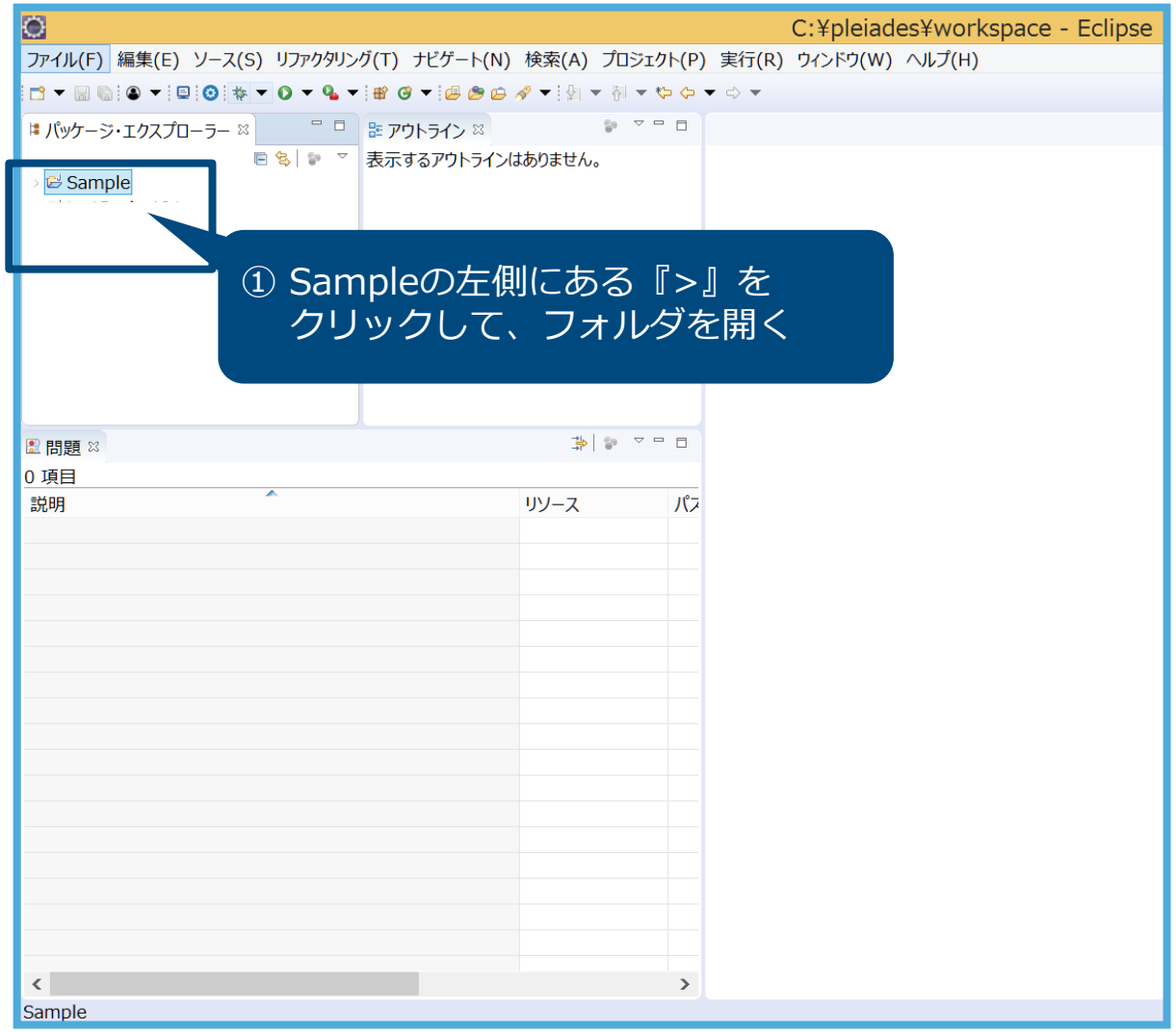
補足 後に解説しますが、プロジェクト名とは、Javaのプロジェクトと呼ばれる一番大きいフォルダ（＝ディレクトリ）のことを指します。

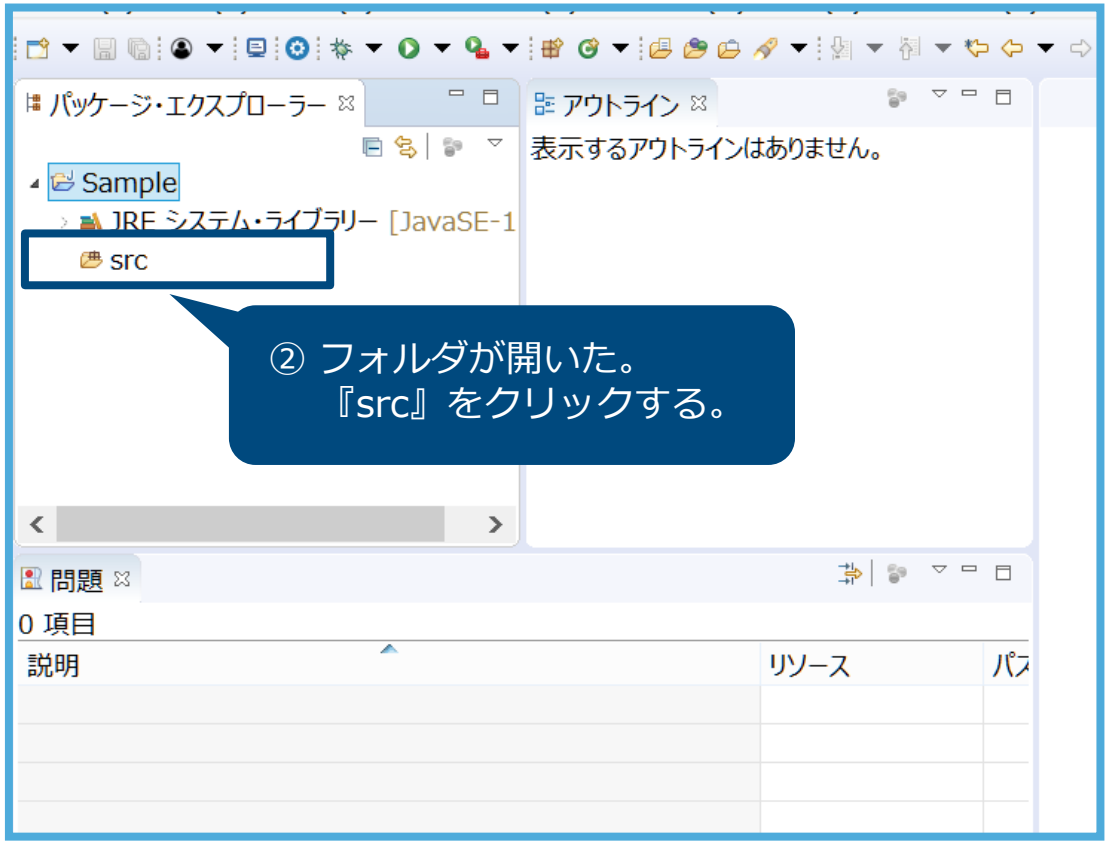
新規でプロジェクトを作成

SampleというJavaプロジェクト（フォルダ=ディレクトリ） の中に
「src」という名前のフォルダを作りますという確認ページ。



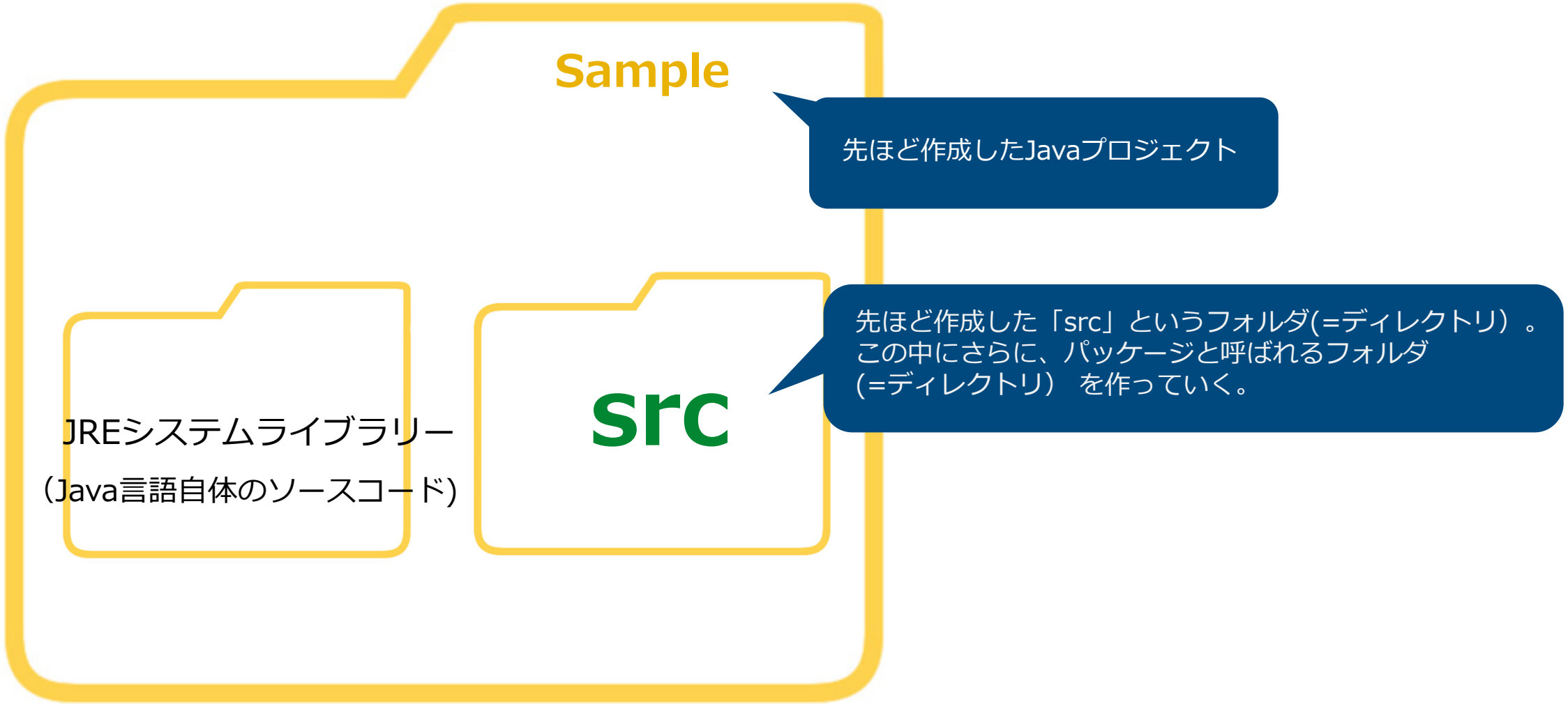
src（ソース）を選択





補足 「JREシステム・ライブラリー」とは、Javaの本体がソースコードが入っている場所です。（ここでは無視してください）

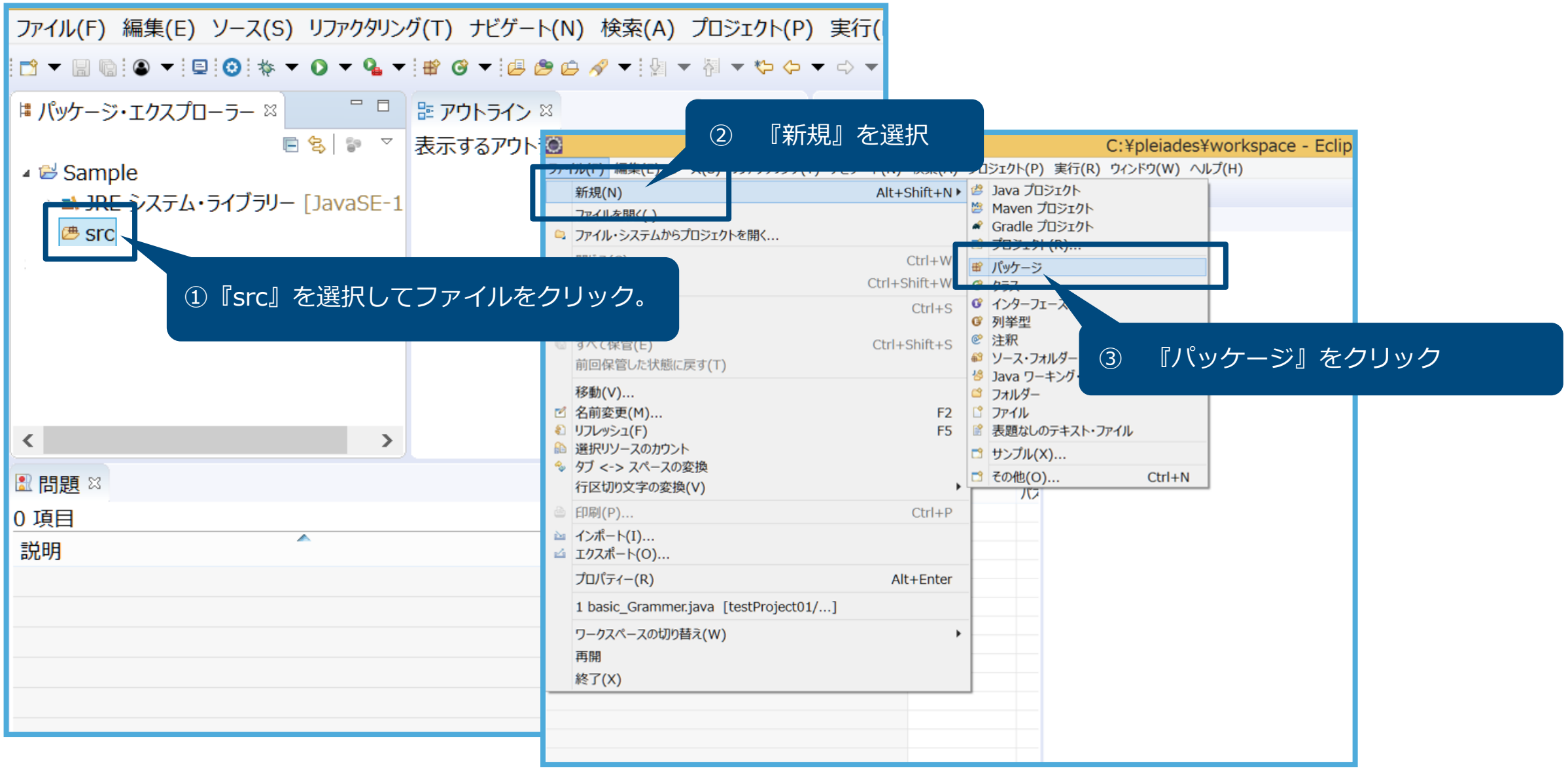
現時点での作成したフォルダ構成（イメージ図）



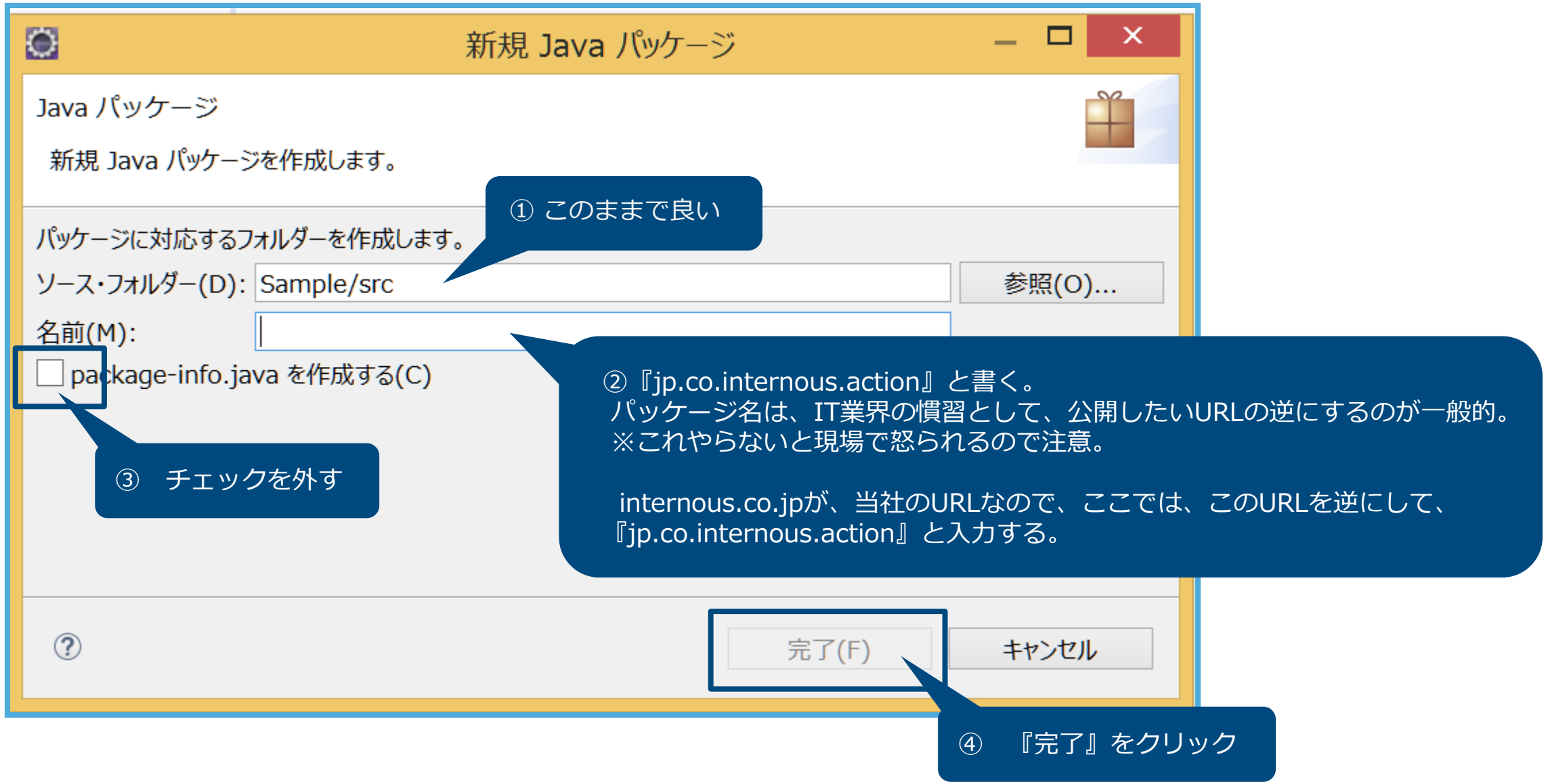
補足

src内に作成するパッケージには、そのパッケージが何のために作られたかをわかりやすくするための名前をつけていきます。パッケージ名前の付け方には、ルールがありますが、次のページ以降で解説します。

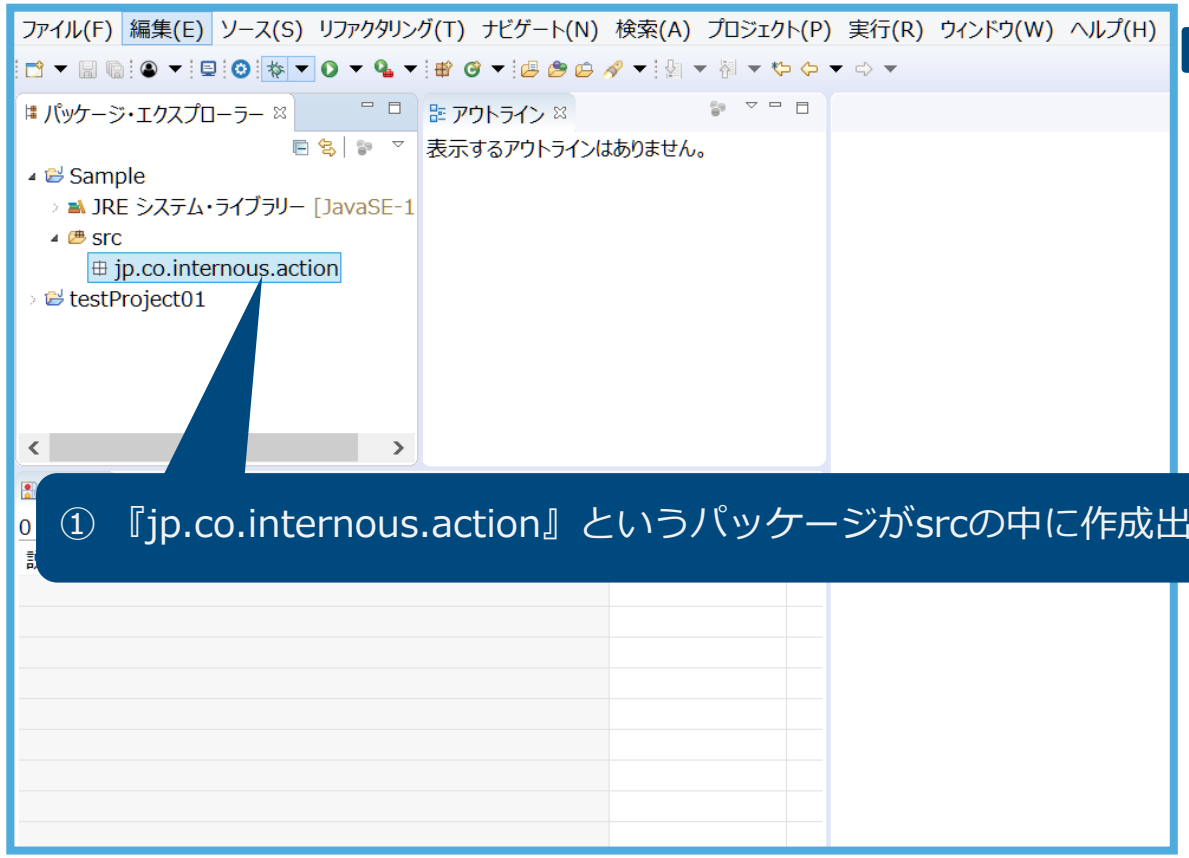
パッケージを作る

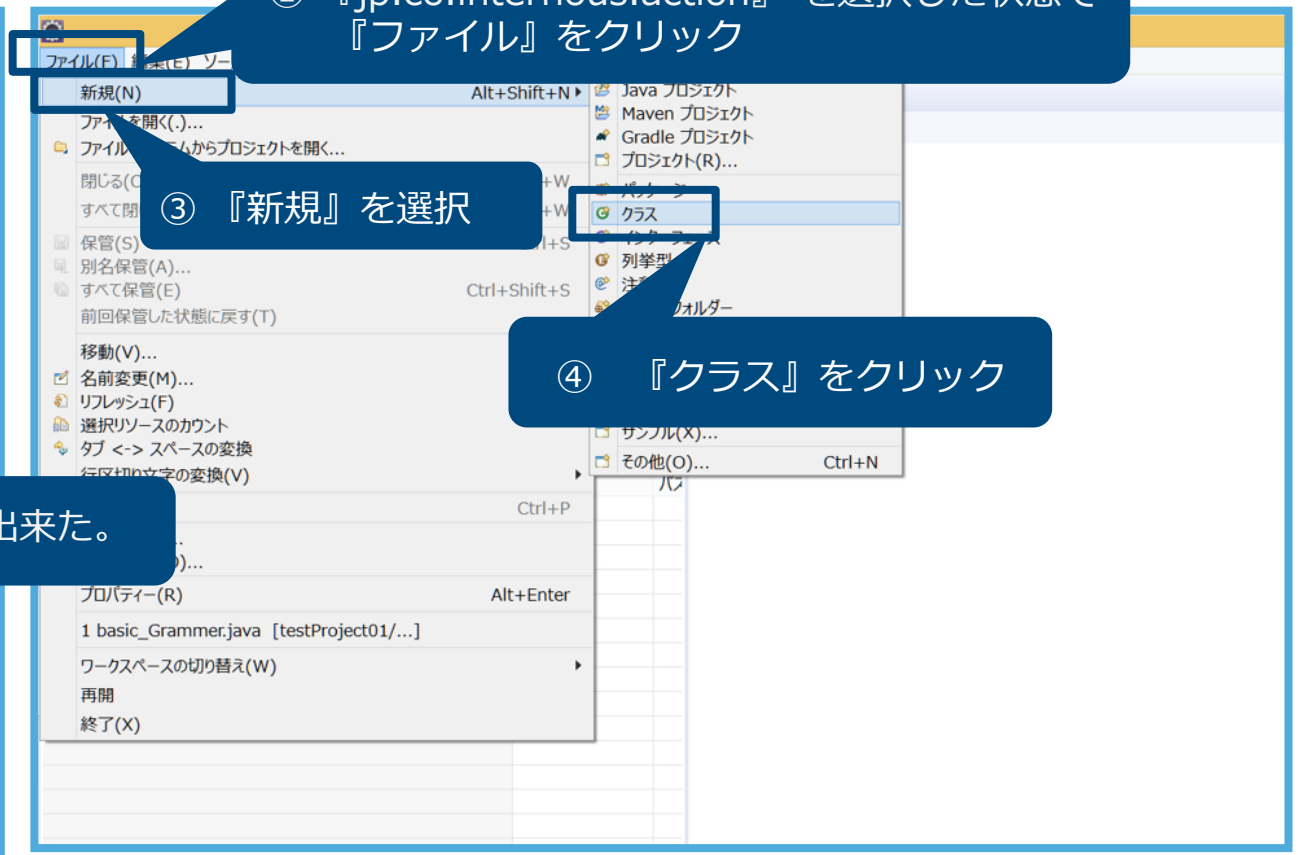

ファイル(F) 編集(E) ソース(S) リファクタリング(T) ナビゲート(N) 検索(A) プロジェクト(P) 実行(
パッケージ・エクスプローラー
アウトライン
表示するアウト
Sample
JRE システム・ライブラリー [JavaSE-1
src
①『src』を選択してファイルをクリック。
② 『新規』を選択
新規(N) Alt+Shift+N
ファイル・システムからプロジェクトを開く...
Ctrl+W
Ctrl+Shift+W
Ctrl+S
Ctrl+Shift+S
前回保管した状態に戻す(T)
移動(V)...
名前変更(M)... F2
リフレッシュ(F) F5
選択リソースのカウント
タブ <-> スペースの変換
行区切り文字の変換(V)
印刷(P)... Ctrl+P
インポート(I)...
エクスポート(O)...
プロパティー(R) Alt+Enter
1 basic_Grammer.java [testProject01/...]
ワークスペースの切り替え(W)
再開
終了(X)
C:¥pleiades¥workspace - Eclip
プロジェクト(P) 実行(R) ウィンドウ(W) ヘルプ(H)
Java プロジェクト
Maven プロジェクト
Gradle プロジェクト
パッケージ
インターフェース
列挙型
注釈
ソース・フォルダー
Java ワーキング
フォルダー
ファイル
表題なしのテキスト・ファイル
サンプル(X)...
その他(O)... Ctrl+N
③ 『パッケージ』をクリック
問題
0 項目
説明

パッケージを作る


新規 Java パッケージ
Java パッケージ
新規 Java パッケージを作成します。
パッケージに対応するフォルダーを作成します。
ソース・フォルダー(D): Sample/src
参照(O)...
名前(M):
package-info.java を作成する(C)
完了(F)
キャンセル
① このままで良い
② 『jp.co.internous.action』と書く。
パッケージ名は、IT業界の慣習として、公開したいURLの逆にするのが一般的。
※これやらないと現場で怒られるので注意。
internous.co.jpが、当社のURLなので、ここでは、このURLを逆にして、
『jp.co.internous.action』と入力する。
③ チェックを外す
④ 『完了』をクリック

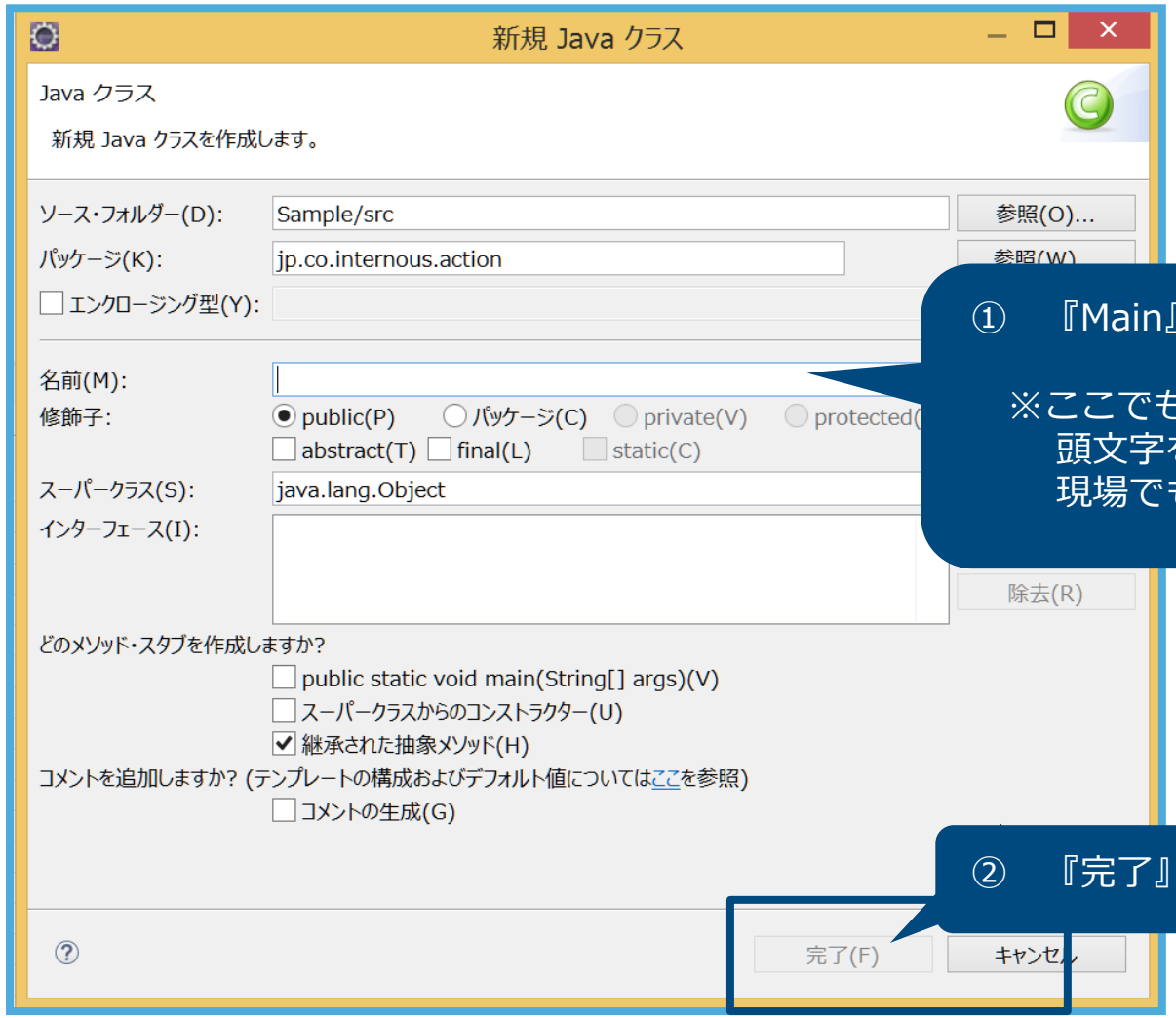
パッケージを作る





補足 『jp.co.internous.action』の左側に表示されている、漢字の田のようなアイコンは、現在パッケージの中身が空という意味。

パッケージを作る



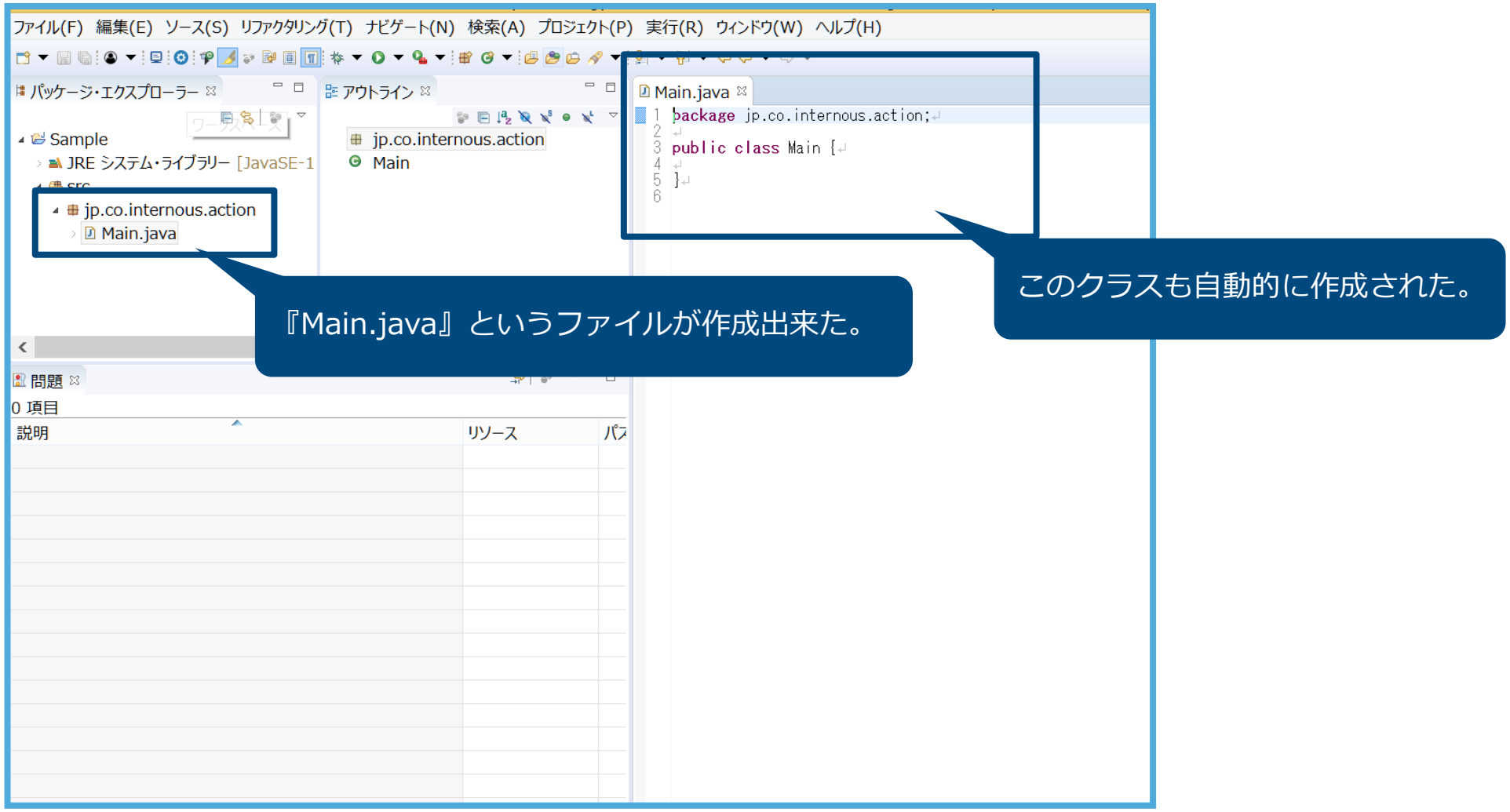
① 『Main』と入力。

※ここでも、頭文字は必ず大文字にすること。
頭文字を小文字にするとアラートがあがり、さらに現場でも怒られるので注意。

② 『完了』をクリック

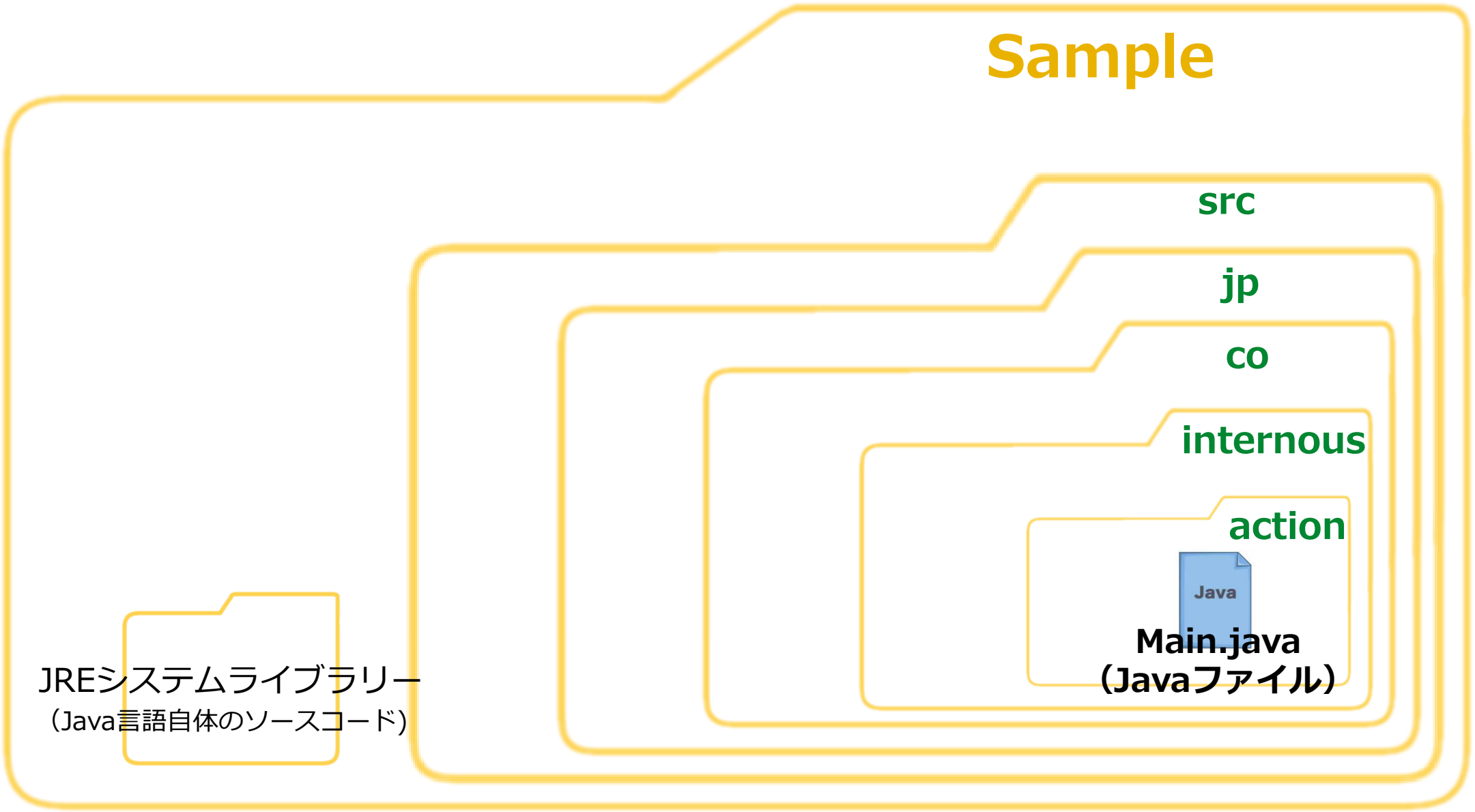
補足 ここで入力した名前『Main』は、クラス名と呼ばれるものです。クラス名は、Javaに最初から組み込まれている用語（=予約語）にしないようにすること。例えば、if、for、while、else、do、null、public、return、true、falseなどにしてはダメ。

パッケージを作る

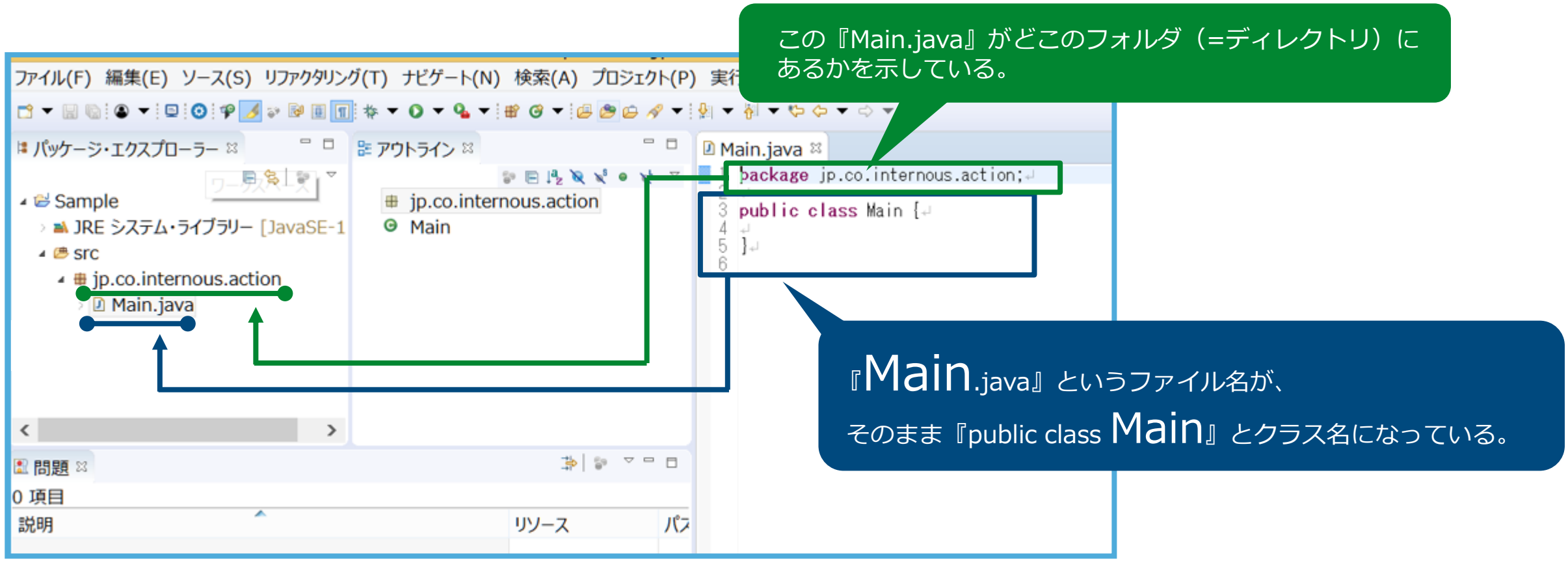


補足 このようにeclipseでJavaファイルを作成すると、自動的にクラス（public class Main）が作成されます。

現時点での作成したフォルダ構成（イメージ図）



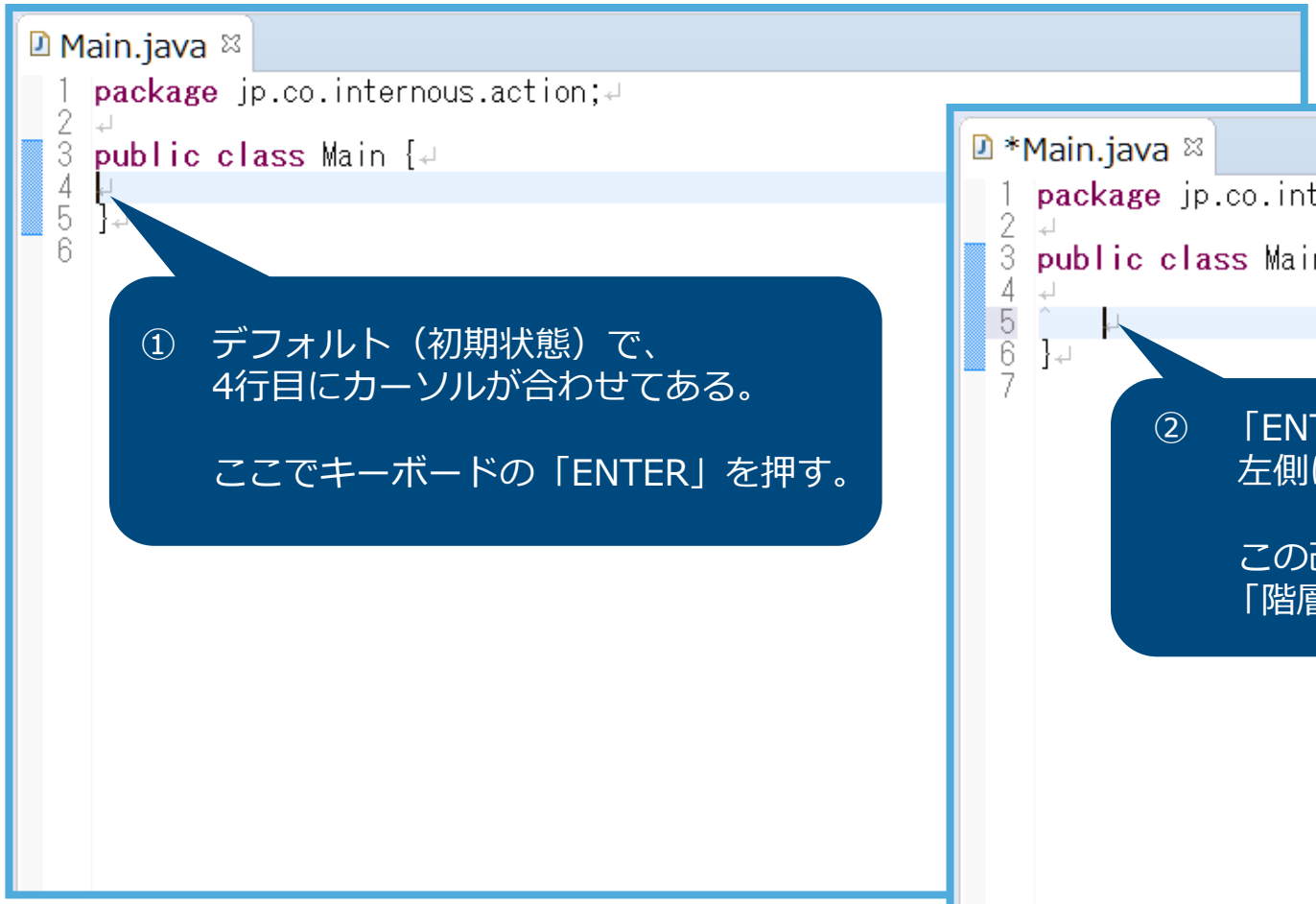
パッケージを作る

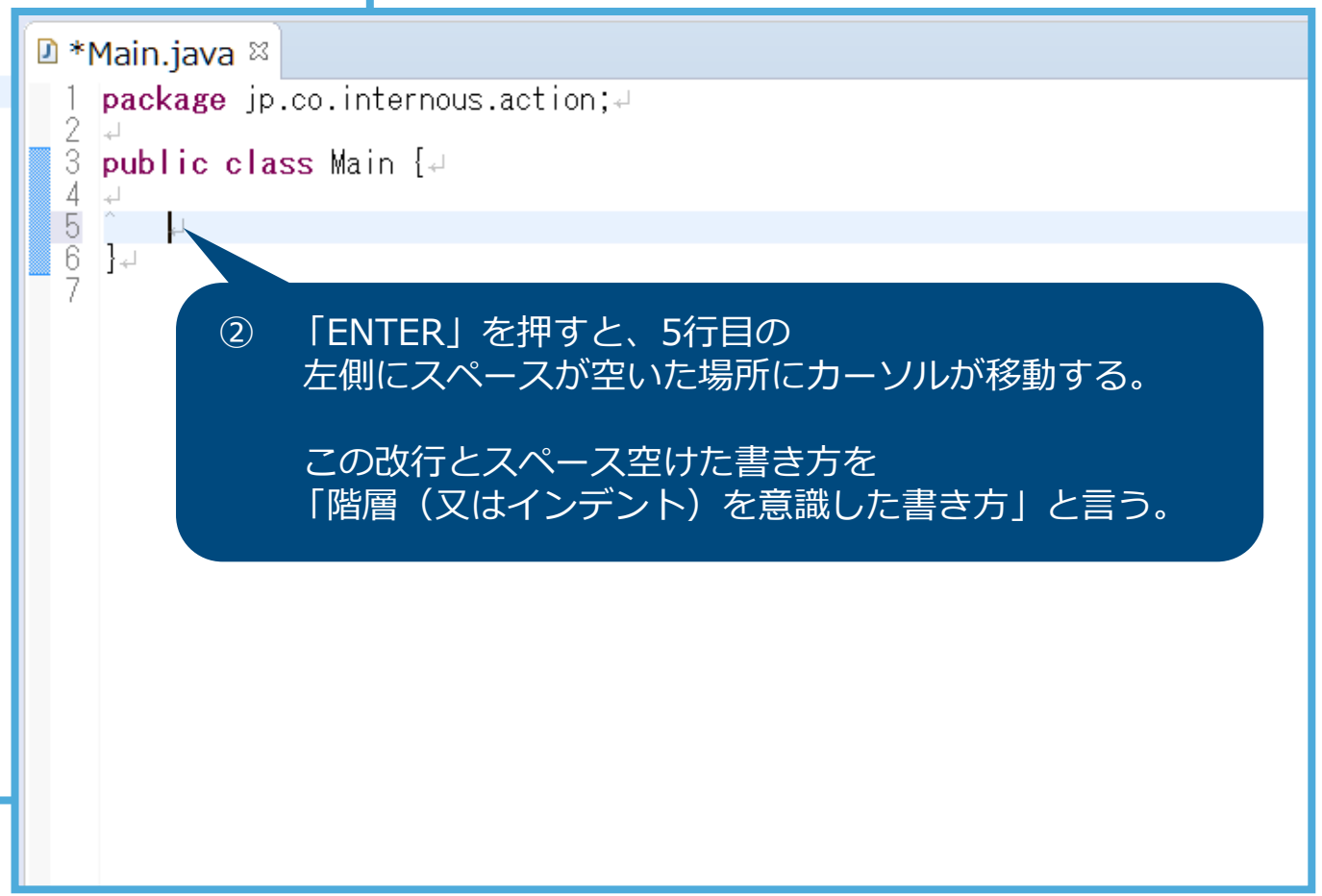

この『Main.java』がどこのフォルダ（=ディレクトリ）にあるかを示している。
ファイル(F) 編集(E) ソース(S) リファクタリング(T) ナビゲート(N) 検索(A) プロジェクト(P)
パッケージ・エクスプローラー
Sample
JRE システム・ライブラリー [JavaSE-1
src
jp.co.internous.action
Main.java
アウトライン
jp.co.internous.action
Main
Main.java
package jp.co.internous.action;
public class Main {
}
問題
0 項目
説明
リソース
『Main.java』というファイル名が、
そのまま『public class Main』とクラス名になっている。

可読性の高いコード（readableコード/リーダブルコード）

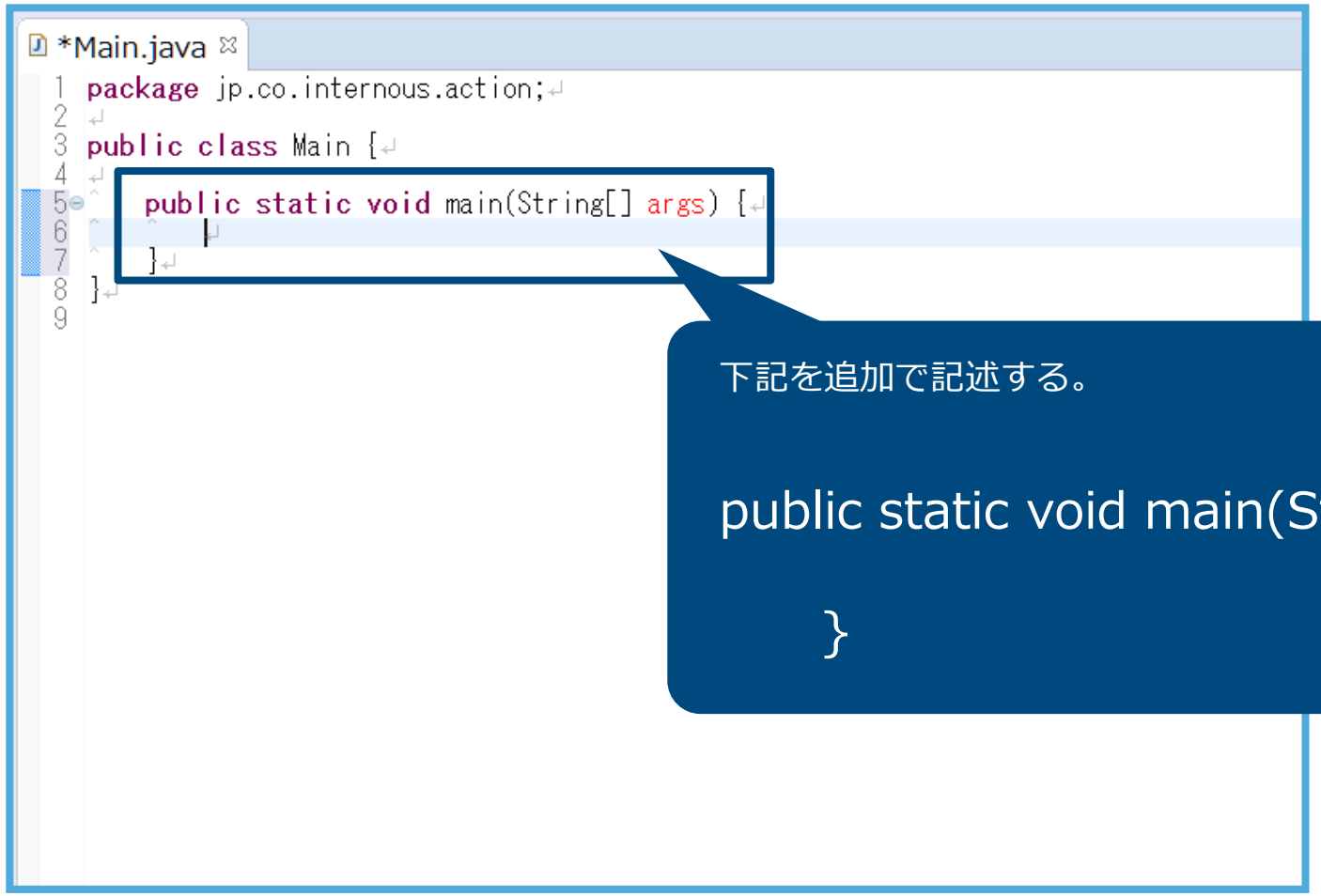
解説

チームで作業をするには、可読性（読みやすさ）の高いコードを書く必要があります。
可読性の高いコードの事を、readable（リーダブル）コードと言います。





public static void main(String[] args){} を追加で記述



下記を追加で記述する。

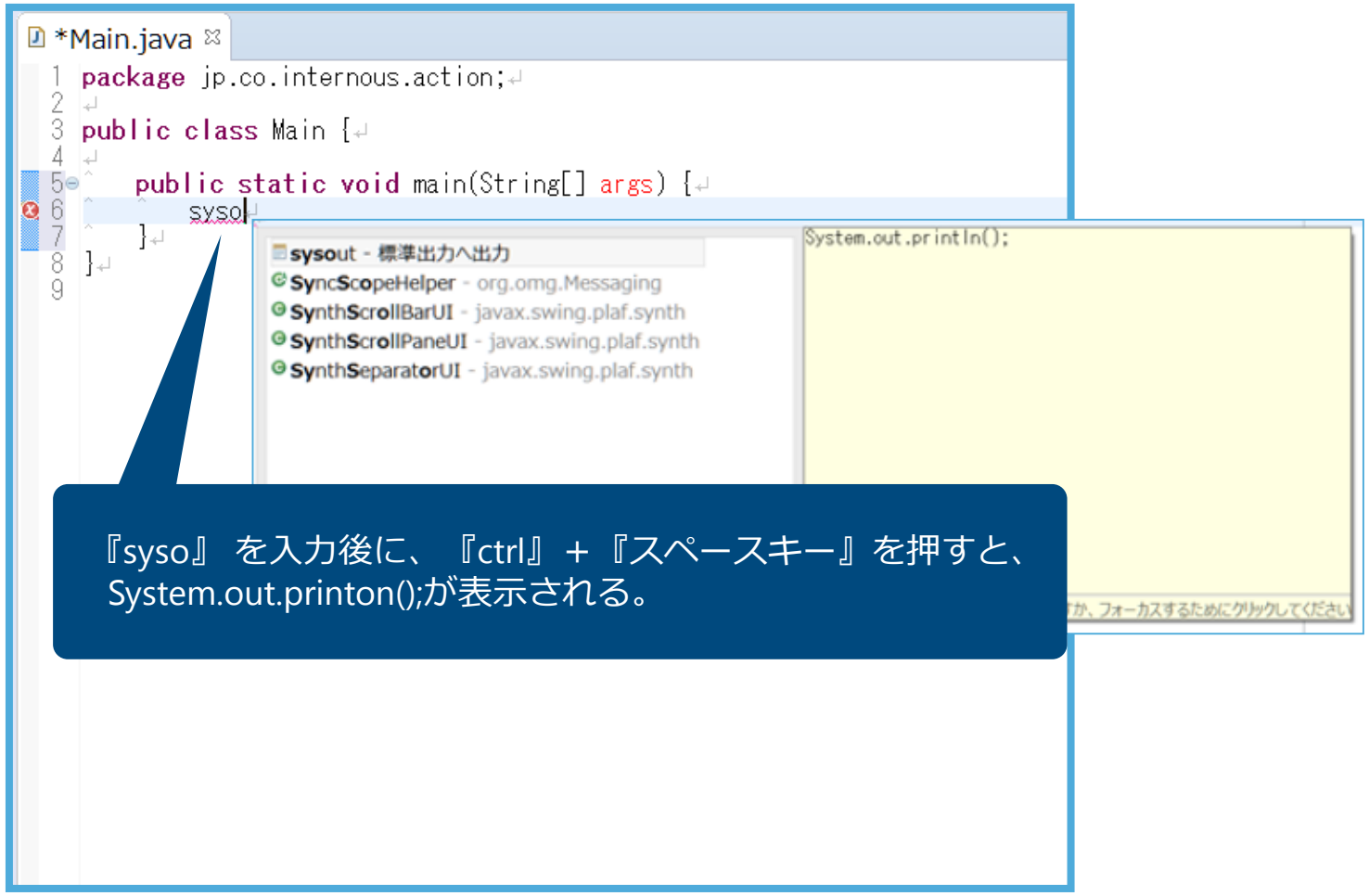
```
public static void main(String[] args){

    }
```

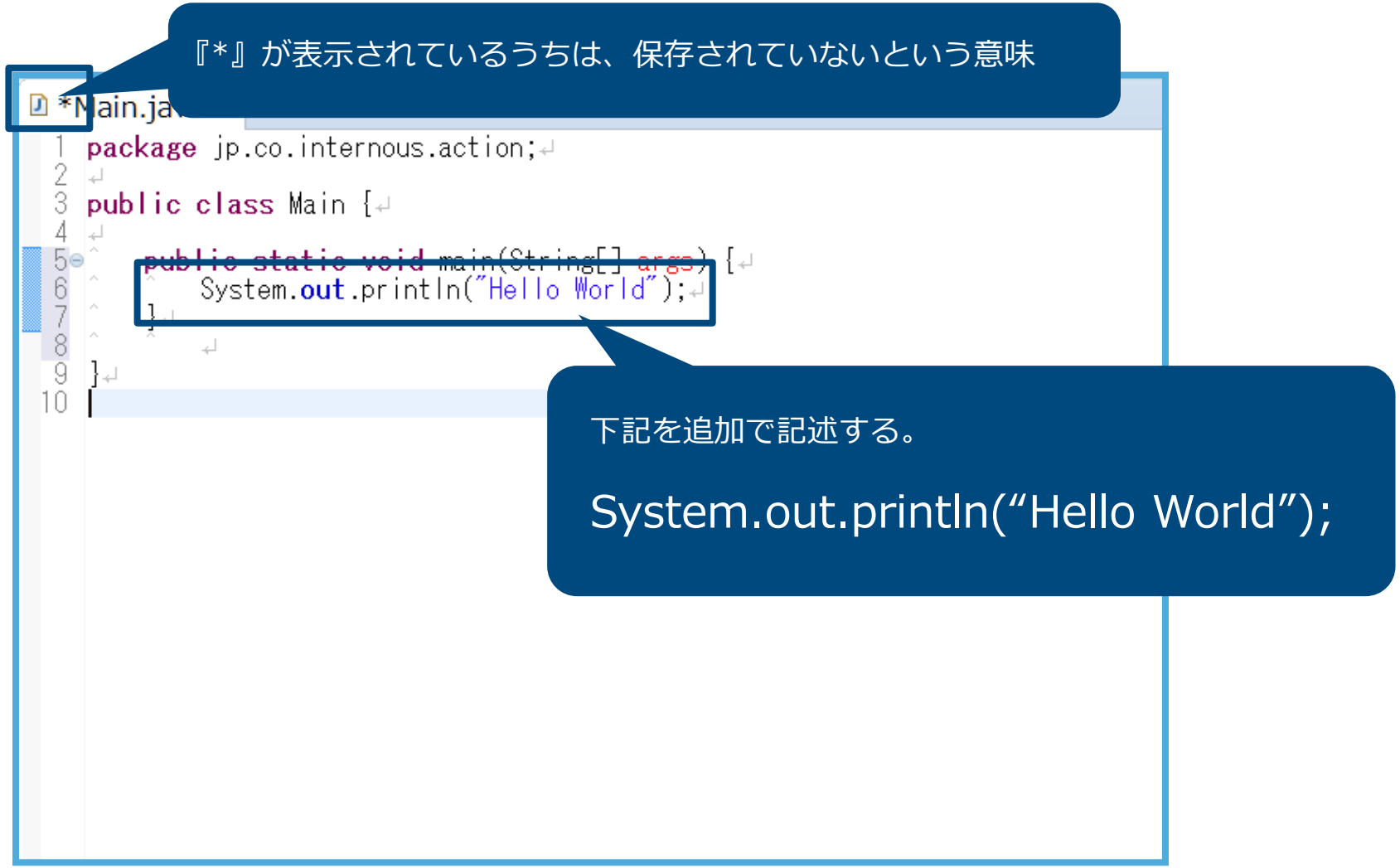
System.out.println();のショートカットキー

解説

『System.out.println();』を表示させる為には、下記のショートカットを活用すると便利です。
『syso』を入力後に『ctrl』＋『スペースキー』を押すと、下記の画面が表示されるので、ここで『ENTER』を押す。



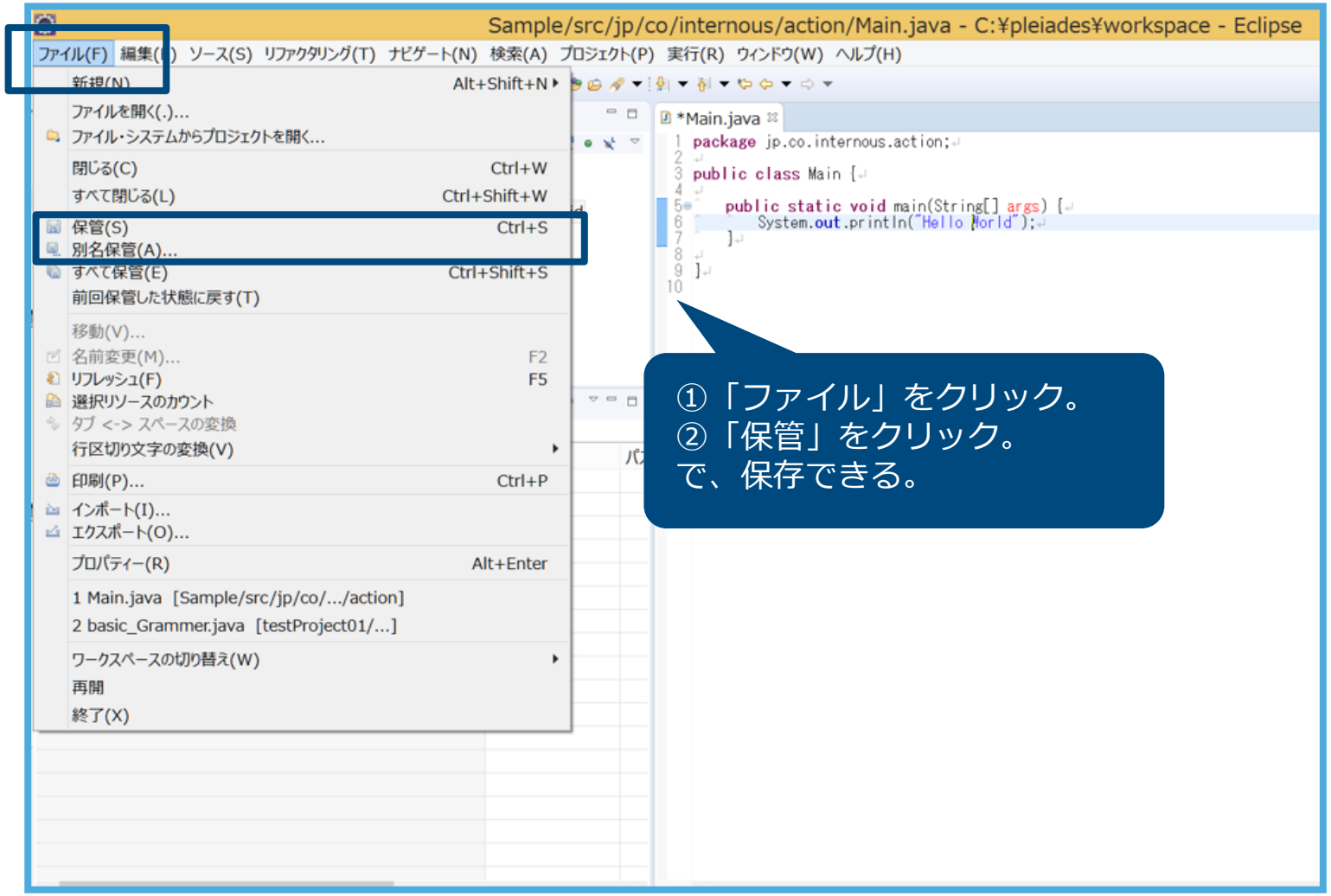
Hello Worldを記述



補足

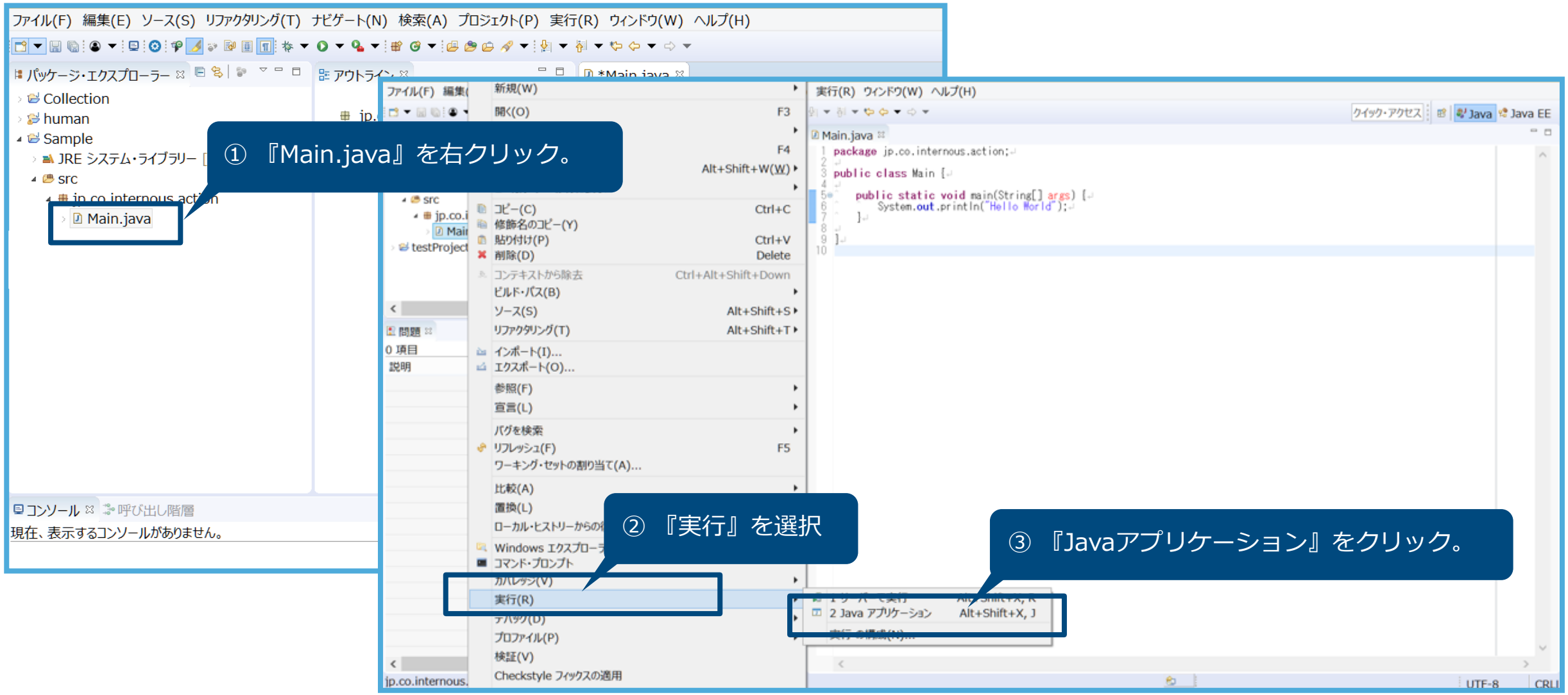
表示したい（=返したい）内容が、1文字の場合は、シングルクオテーション。
表示したい（=返したい）内容が、1文字以上(=文字列）の場合は、ダブルクオテーション。

保存

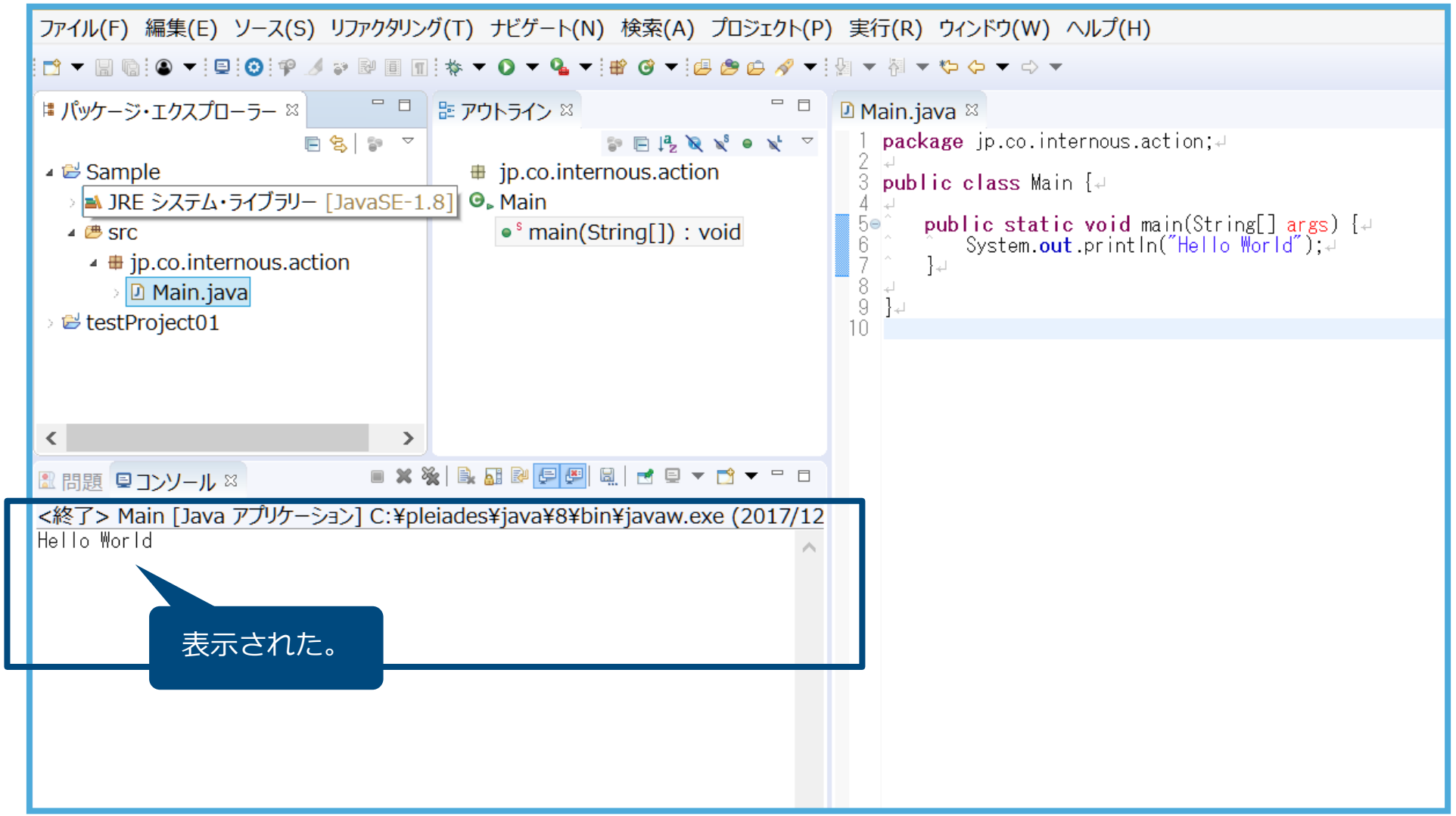


補足 上記の方法以外にも、『ctrl』＋『s』のショートカットで保存も可能。

実行する



実行結果


ファイル(F) 編集(E) ソース(S) リファクタリング(T) ナビゲート(N) 検索(A) プロジェクト(P) 実行(R) ウィンドウ(W) ヘルプ(H)
パッケージ・エクスプローラー
Sample
JRE システム・ライブラリー [JavaSE-1.8]
src
jp.co.internous.action
Main.java
testProject01
アウトライン
jp.co.internous.action
Main
main(String[]) : void
Main.java
1 package jp.co.internous.action;
2
3 public class Main {
4
5 public static void main(String[] args) {
6 System.out.println("Hello World");
7 }
8
9 }
10
問題
コンソール
<終了> Main [Java アプリケーション] C:¥pleiades¥java¥8¥bin¥javaw.exe (2017/12
Hello World
表示された。