

Javaプログラミング実習

03. Eclipseの基本

株式会社ジードライブ

今回学ぶこと

- Eclipseの基本的な用語や使い方
- Javaプログラムの作成から実行までの手順
- データの持ち帰り方

Eclipseについて

- 無償で使用可能な統合開発環境 (IDE)
- Javaだけでなく、PHP, C, C++ など、様々なプログラミング言語の開発に対応できる拡張性を持つ



Eclipseの便利な機能

- コード編集支援機能
 - コードの色付け
 - コード補完
 - 文法チェック, などなど...
- デバッグ支援機能
 - プログラムのステップ実行
 - プログラムで使用されている変数の閲覧/操作

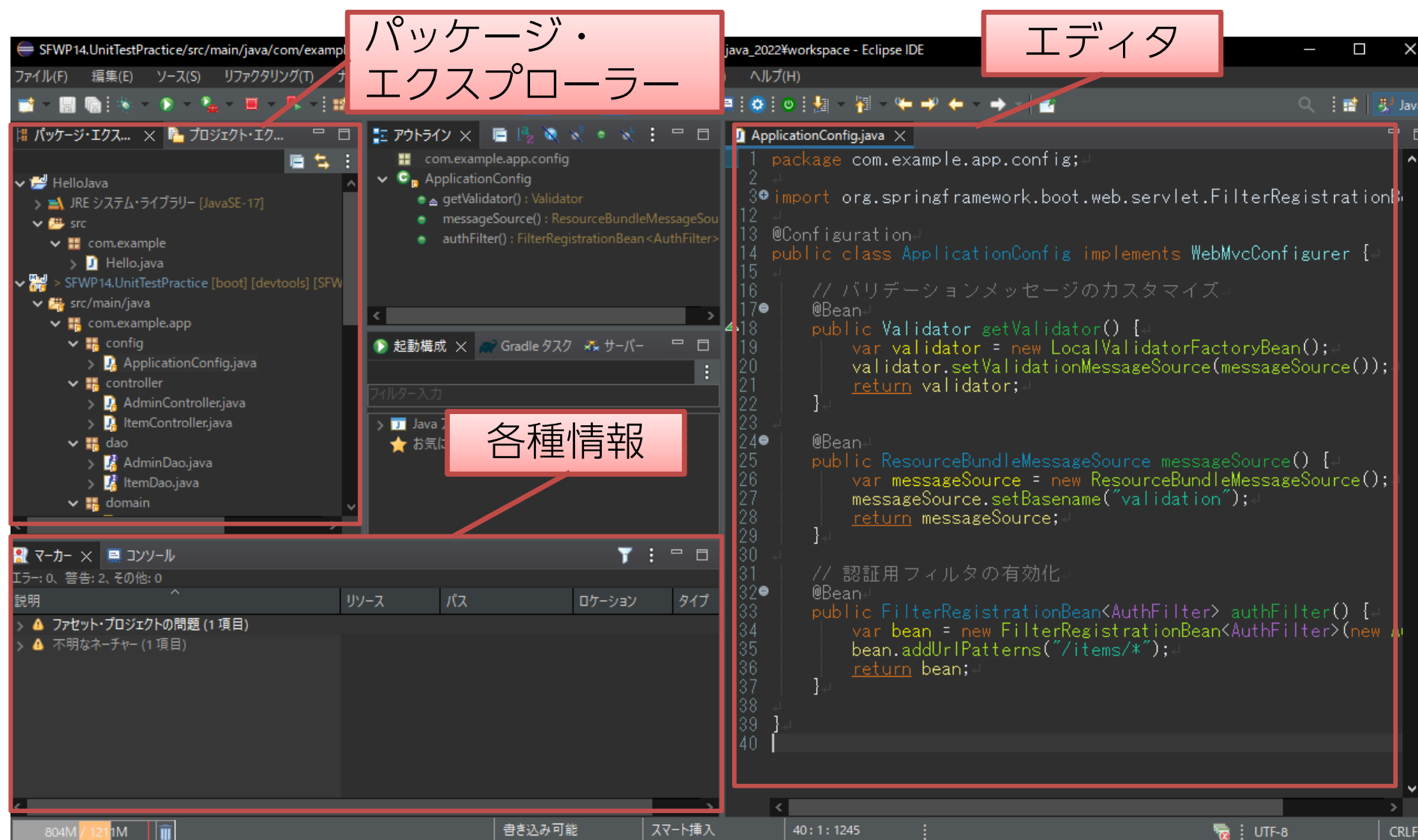
Eclipseの基本用語（１）

- ワークスペース
 - EclipseがJavaファイルや各種設定ファイルを保存するフォルダ
- ワークベンチ
 - Eclipseを起動すると現れる作業環境のこと
- ビュー
 - ワークベンチ内に配置される、ソースコードやファイルリストなどのパネル

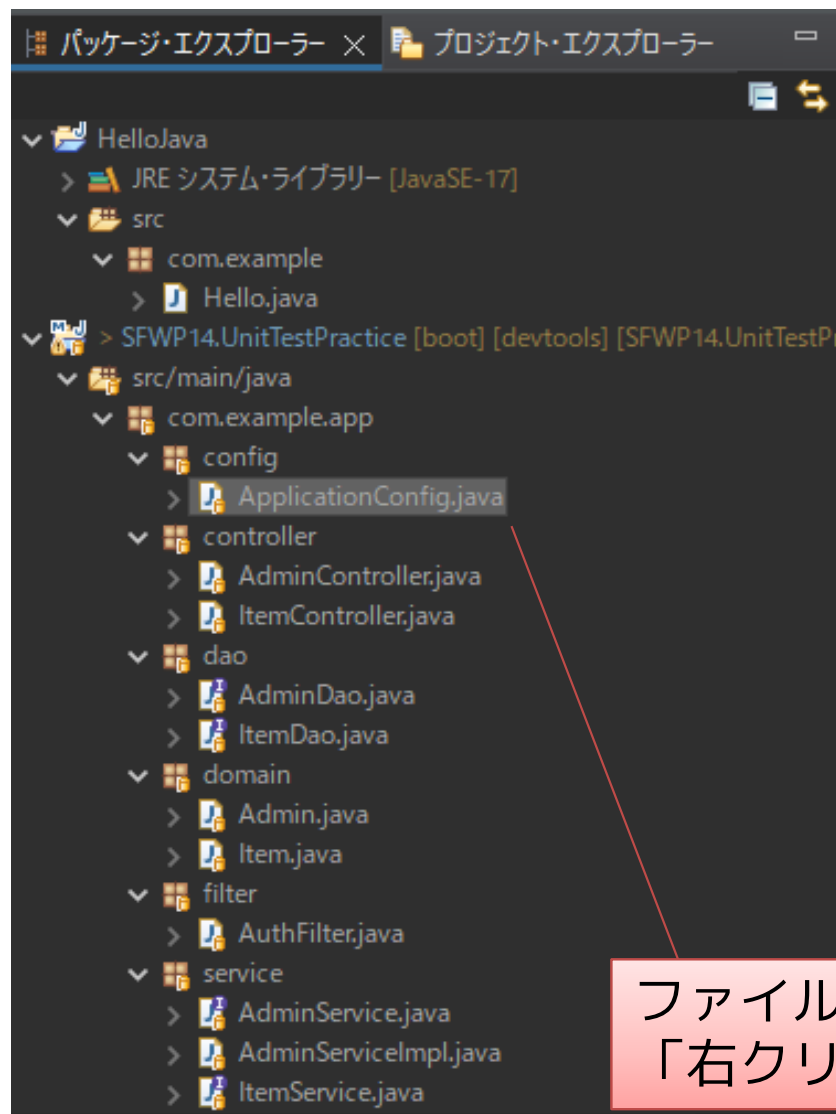
Eclipseの基本用語（2）

- パースペクティブ
 - ビューの組み合わせ／配置に名前を付けたもの
 - おもに Javaパースペクティブ や デバッグパースペクティブ を使う
- プロジェクト
 - 開発対象はプロジェクトという形で管理する
 - プロジェクトはワークスペース内に保存される

ワークベンチ内のおもなビュー



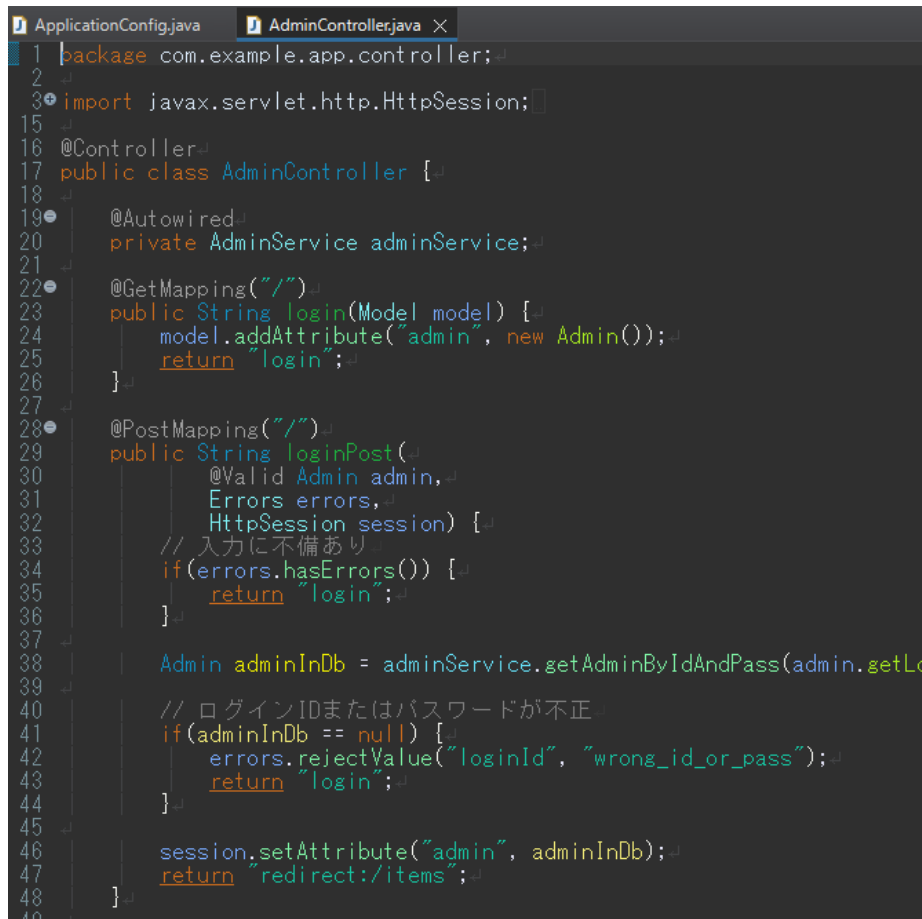
パッケージ・エクスプローラー



- プロジェクトごとにソースファイルをまとめてリスト表示する
 - プロジェクト・エクスプローラーも同様の機能を提供するが、表示の仕方や表示される項目が異なる
- 表示されているファイルをダブルクリックすると、エディタ内に内容が表示される
 - 外部のビューアーなどが起動する場合もある

ファイル名を変更する場合、
「右クリック」⇒「リファクタリング」⇒「名前変更」

エディタ



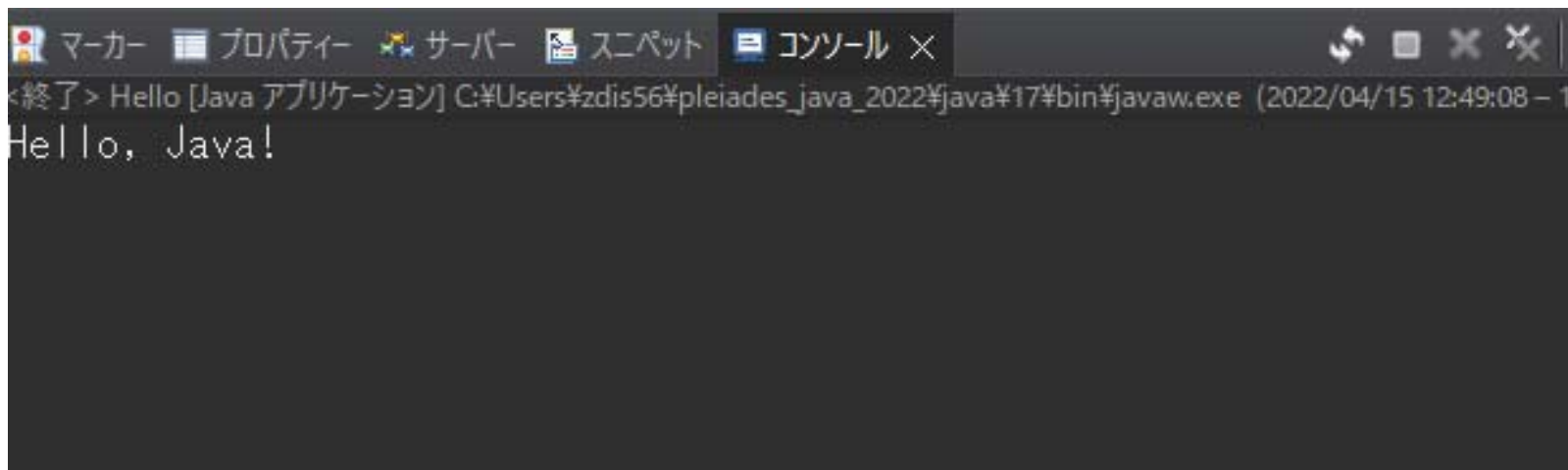
```
1 package com.example.app.controller;
2
3 import javax.servlet.http.HttpSession;
4
5
6
7
8
9
10
11
12
13
14
15 @Controller
16 public class AdminController {
17
18     @Autowired
19     private AdminService adminService;
20
21
22     @GetMapping("/")
23     public String login(Model model) {
24         model.addAttribute("admin", new Admin());
25         return "login";
26     }
27
28     @PostMapping("/")
29     public String loginPost(
30         @Valid Admin admin,
31         Errors errors,
32         HttpSession session) {
33         // 入力に不備あり
34         if(errors.hasErrors()) {
35             return "login";
36         }
37
38         Admin adminInDb = adminService.getAdminByIdAndPass(admin.getLoginId(), admin.getPassword());
39
40         // ログインIDまたはパスワードが不正
41         if(adminInDb == null) {
42             errors.rejectValue("loginId", "wrong_id_or_pass");
43             return "login";
44         }
45
46         session.setAttribute("admin", adminInDb);
47         return "redirect:/items";
48     }
49 }
```

- ソースファイルの内容を参照したり編集したりするためのウィンドウ
- 複数のファイルを同時に開き、タブで切り替えることができる
- コード補完など様々な補助機能を備える
 - 構文に従った色付け
 - 問題がある場所の指摘など

Ctrlを押しながら、+- で文字の大きさを変更できる

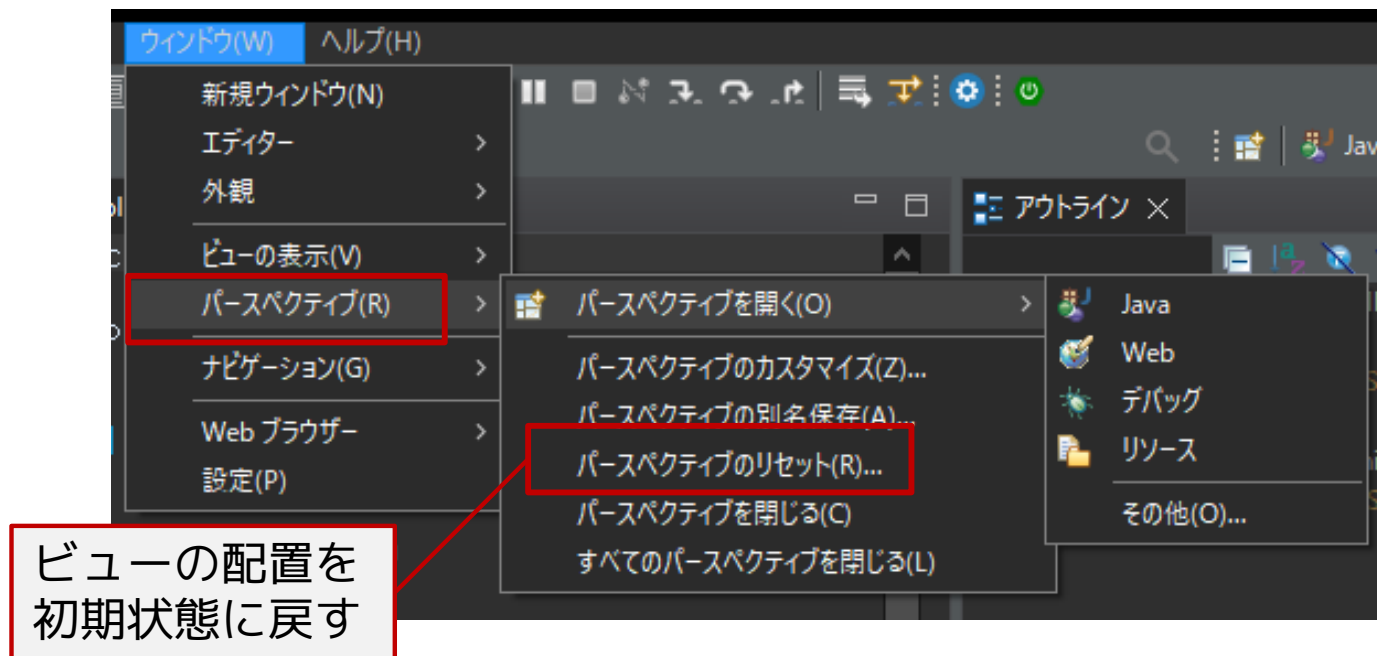
各種情報

- コンソールへの出力やエラーメッセージ、ファイル内の検索結果など各種情報を表示するウィンドウ
 - その時々で必要なタブが自動的にアクティブになる



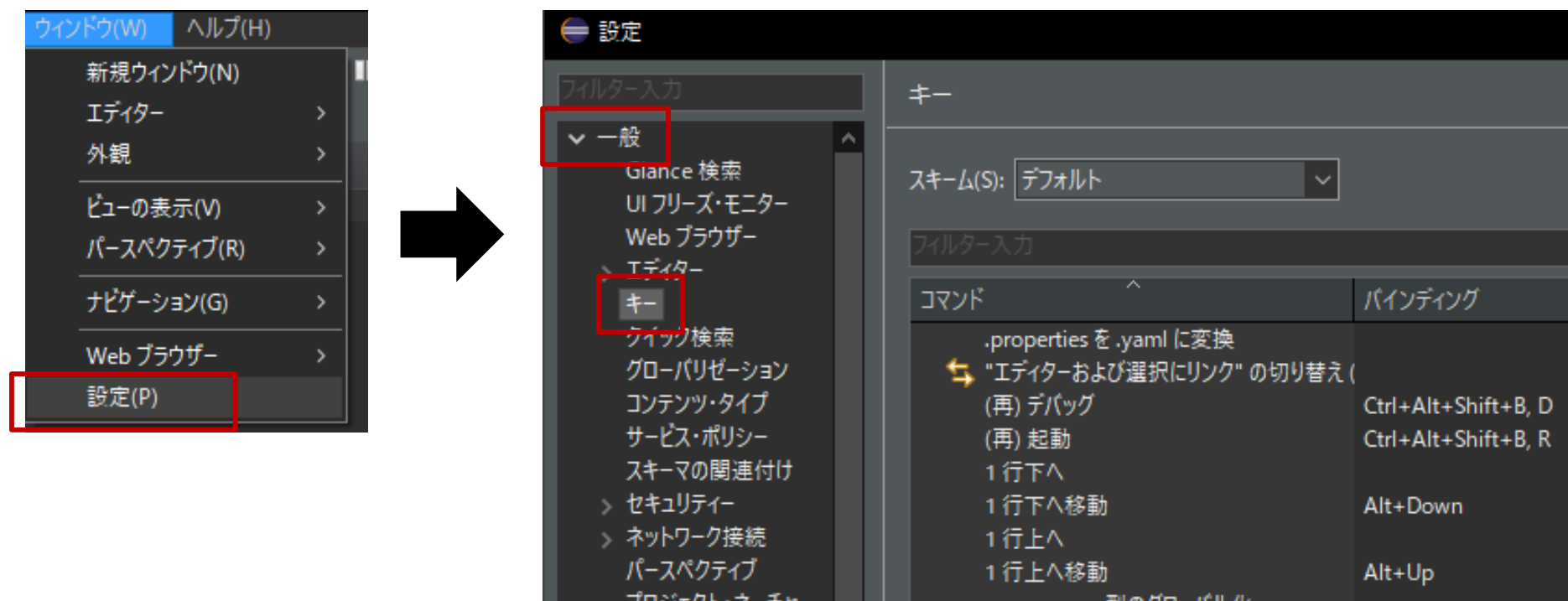
パースペクティブの切り替え

- ウィンドウメニューからパースペクティブ(ビューの組み合わせ)を切り替えることができる
 - 状況に応じてパースペクティブを変更すると作業しやすい
- ビューの配置を初期状態に戻したい場合は、パースペクティブをリセットする



設定の変更

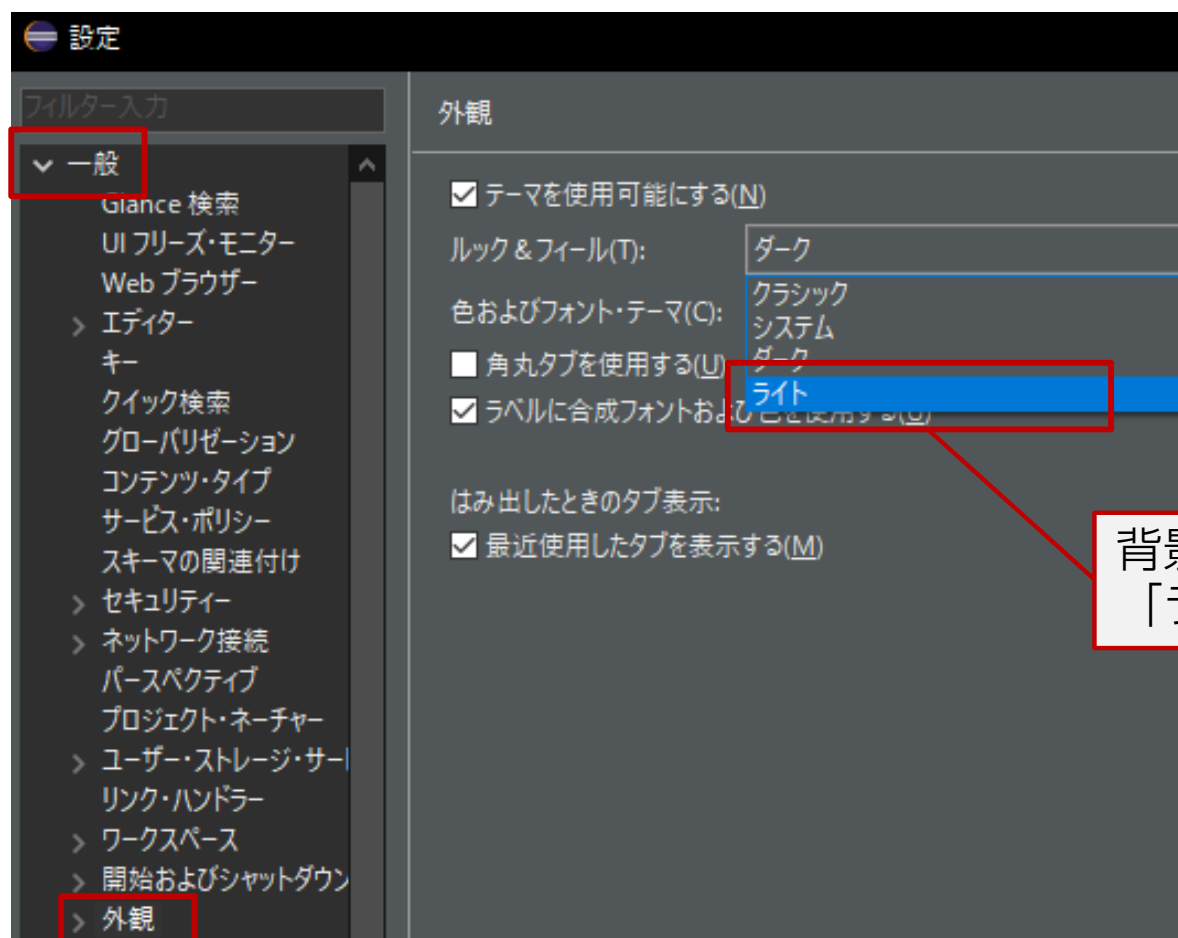
- ショートカットキーや画面の色などを変更する場合には、ウィンドウメニューから「設定」を開く



ショートカットは「一般」⇒「キー」で変更可能

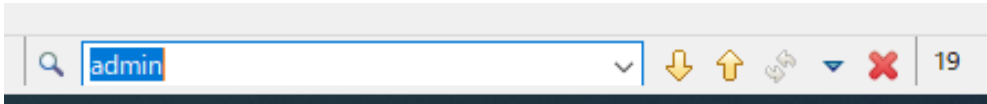
色の変更

- 画面の色は、「一般」⇒「外観」で変更することが可能



背景を白ベースにする場合は「ライト」を選択

ショートカットの一例

ショートカット	説明
Ctrl + Space	入力補助の表示。Escapeで非表示にできる
Shift + Enter	現在の行の下に空き行を作成し、カーソルを移動する
Alt + ↑ ↓	行の移動。行を複製する場合は、Alt + Ctrl + ↑ ↓
Ctrl + ← →	カーソルを単語ごと移動
Alt + Shift + A	矩形選択（上下左右にドラッグしてテキストを選択する）
Ctrl + Shift + F	フォーマット（ソースコードを整形する）
Ctrl + J	現在選択しているテキストをGlance検索する ⇒ 同じテキストがすべてハイライトされる ⇒ 画面下部にGlance検索パネルが表示される 

画面回転の解除

- Ctrl + Alt + ↑↓ で、画面が180度回転してしまう場合は、デスクトップ上で右クリックし、**その他のオプションを確認**を押下する
⇒ その後は、以下の図のようにホットキーを無効化する



Javaプログラムの 作成から実行まで

今回作成するプログラム

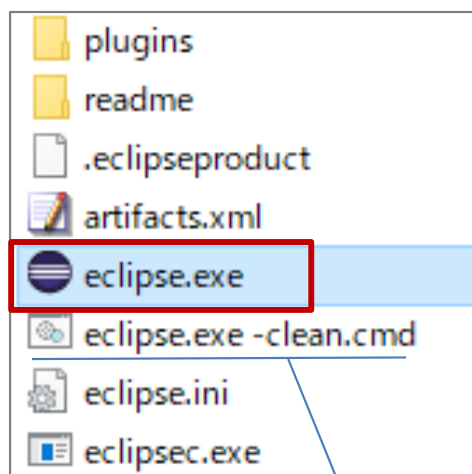
- コンソールウィンドウに「Hello, Java!」 と表示するプログラム

作業の流れ

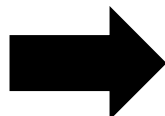
1. Eclipseの起動
2. ワークスペースの選択
3. 新規プロジェクトの作成
4. ソースファイルの作成
 - 今回は、コンソールウィンドウに「Hello, Java!」と表示するプログラムを作成
5. コンパイルと実行

1. Eclipseの起動

- `pleiades/eclipse/eclipse.exe`をダブルクリック



Eclipseが不調な場合は、
`clean.cmd`を実行してみる



初回起動は少し時間がかかる

2. ワークスペースの選択

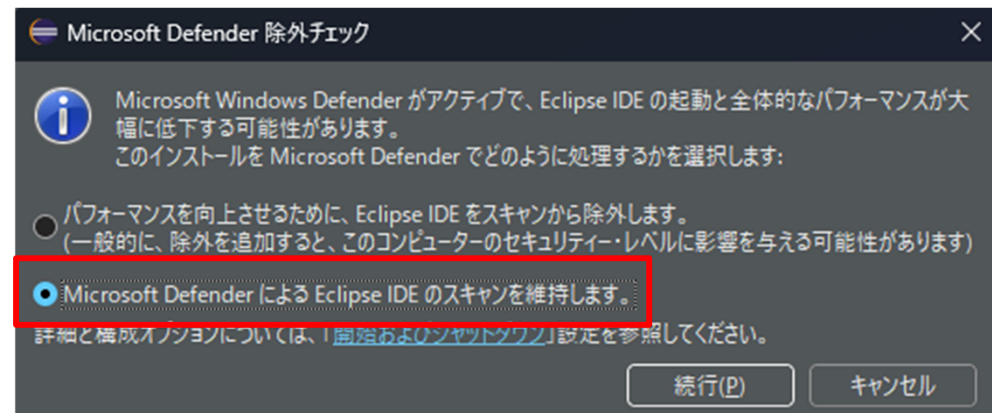
- ワークスペースを選択する
 - 特に変更せずに進めてよい
- ⇒ pleiadesフォルダ内に「workspace」フォルダが生成される



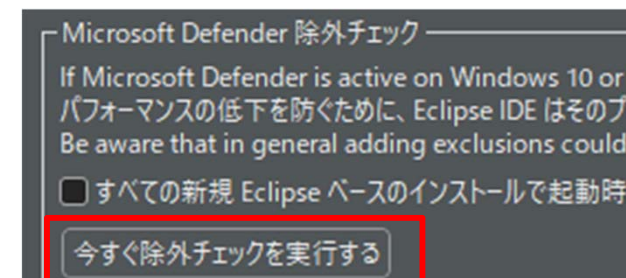
チェックを入れると、次回からこのウィンドウが表示されなくなる

Microsoft Defenderの設定

- 初回のみ表示されるので、スキヤンの維持にチェックを入れる

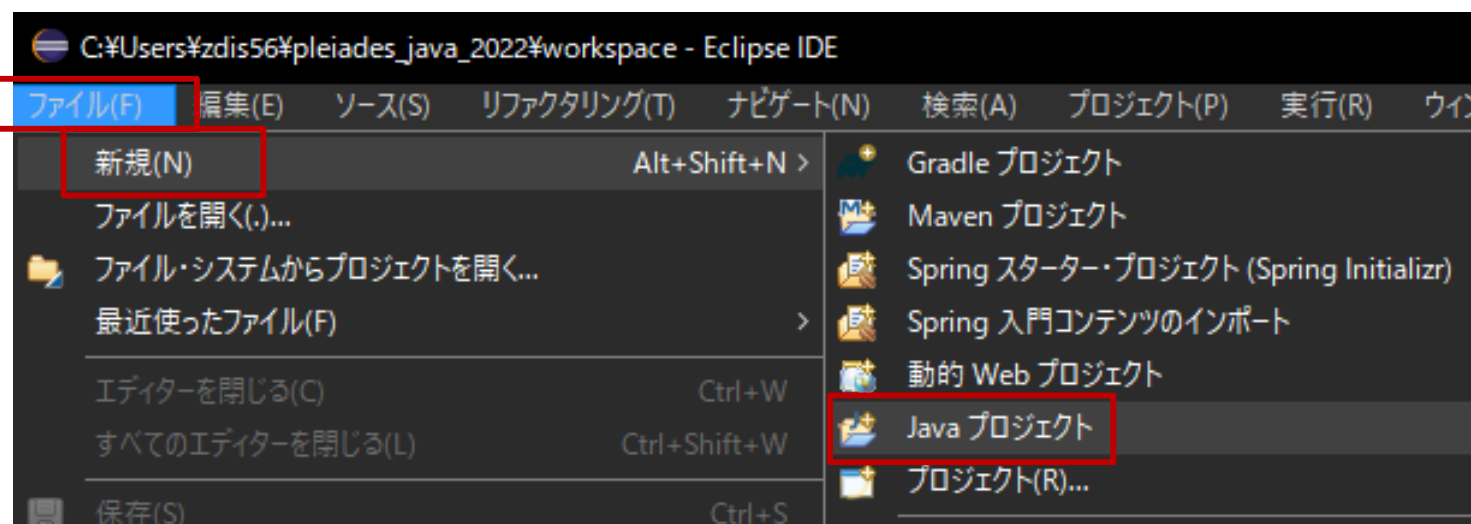


- Eclipseの起動が遅い場合は、
設定メニュー → 一般 → 開始およびシャットダウン
から再設定する（管理者権限が必要）



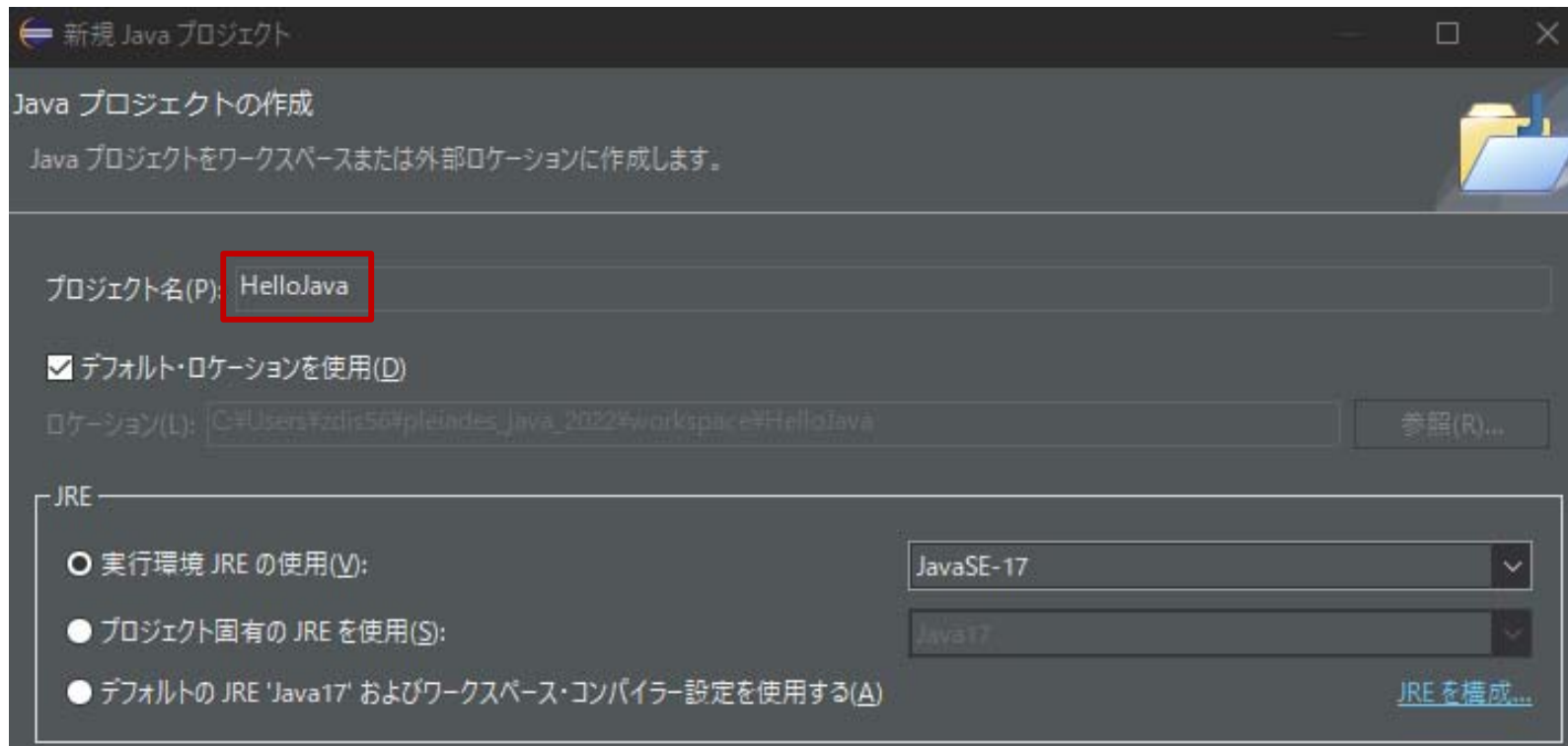
3. 新規プロジェクトの作成

- ファイルメニューから「新規」⇒「Javaプロジェクト」を選択



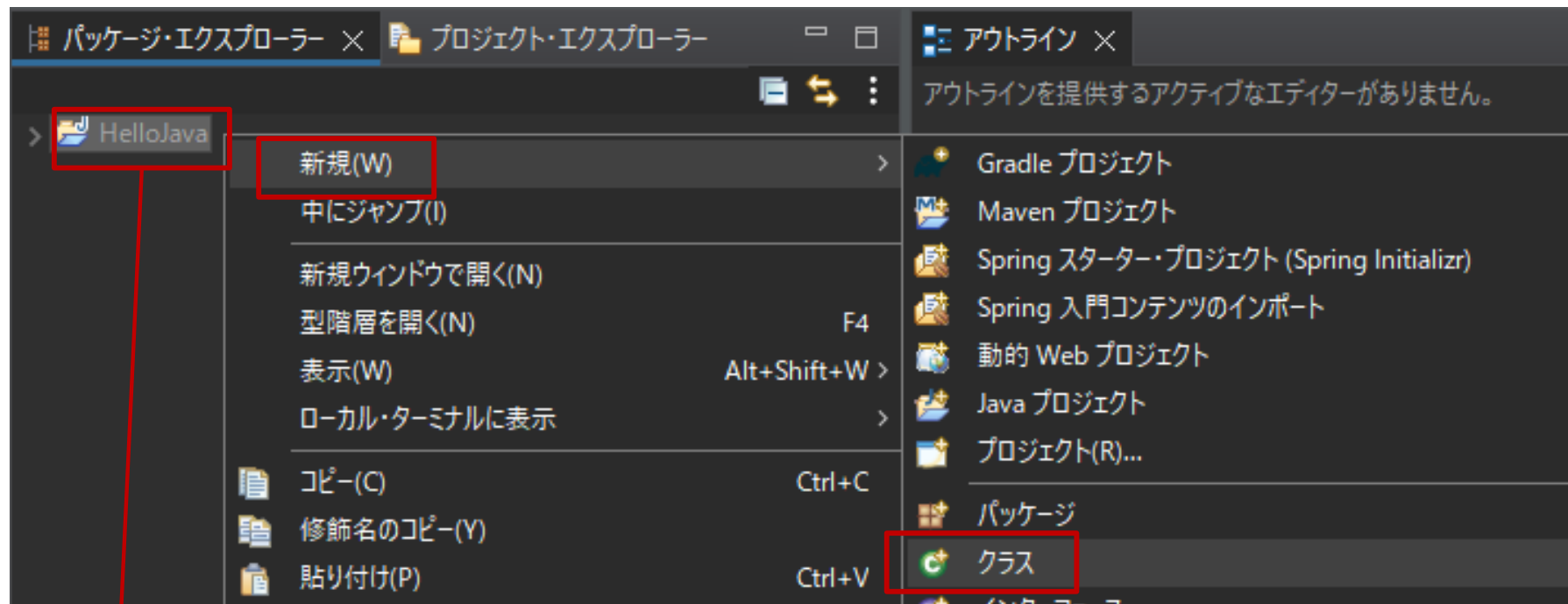
3. 新規プロジェクトの作成

- 任意のプロジェクト名を入力し、画面下部の「完了」をクリック
 - 以下の例では「HelloJava」というプロジェクト名を入力



4. ソースファイルの作成

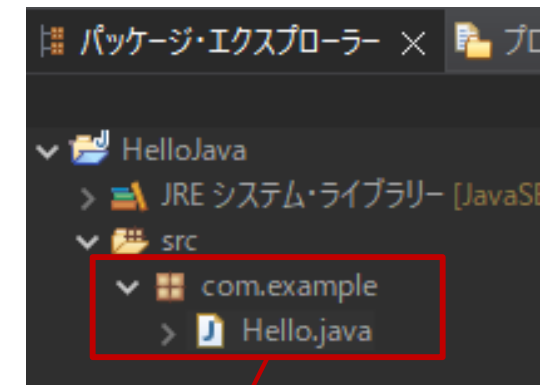
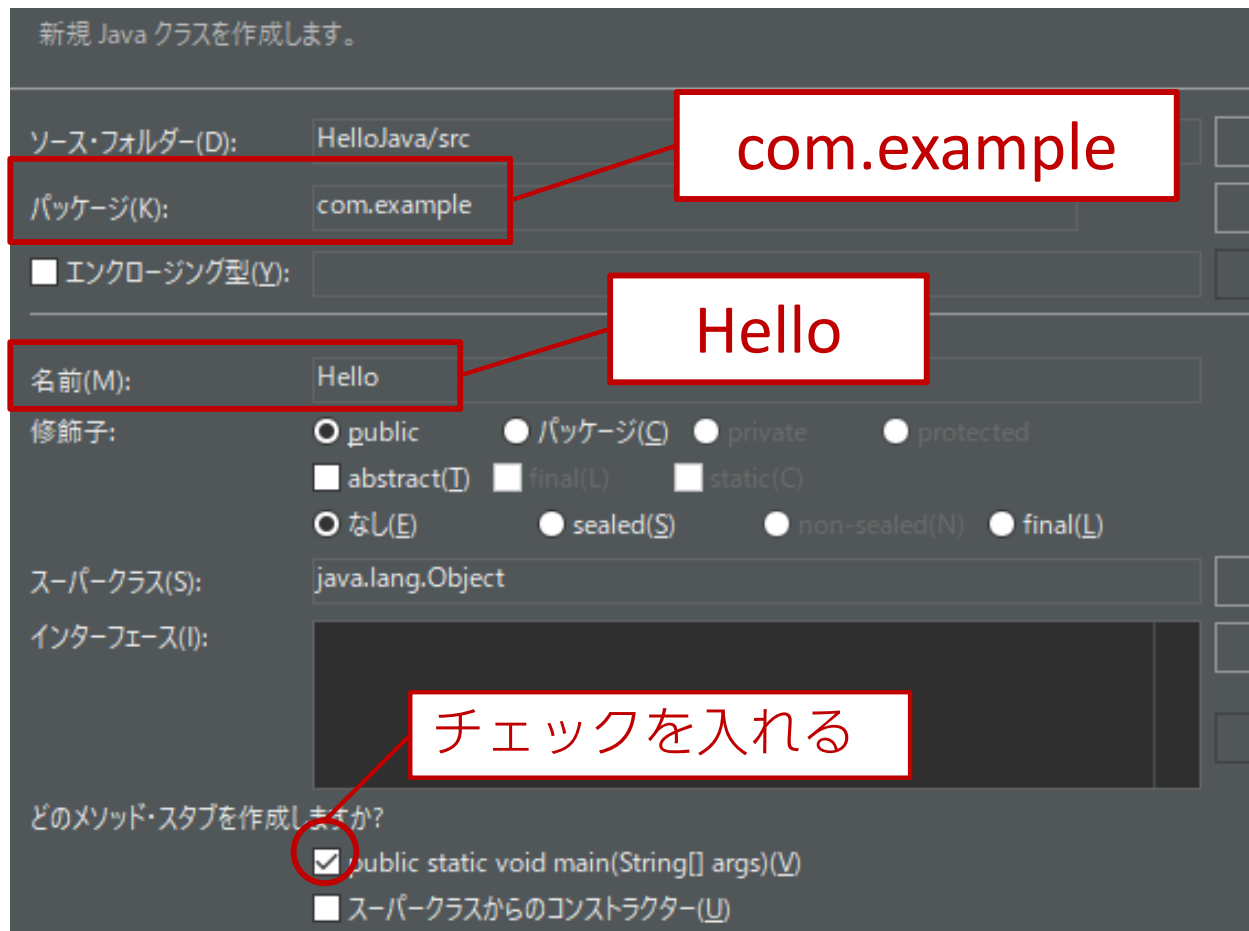
- プロジェクトフォルダ上で右クリックをし、「新規」⇒「クラス」を選択



右クリック

4. ソースファイルの作成

- 任意のパッケージ名とクラス名を入力し、画面下部の「完了」をクリック



ソースファイルが
作成される

4. ソースファイルの作成

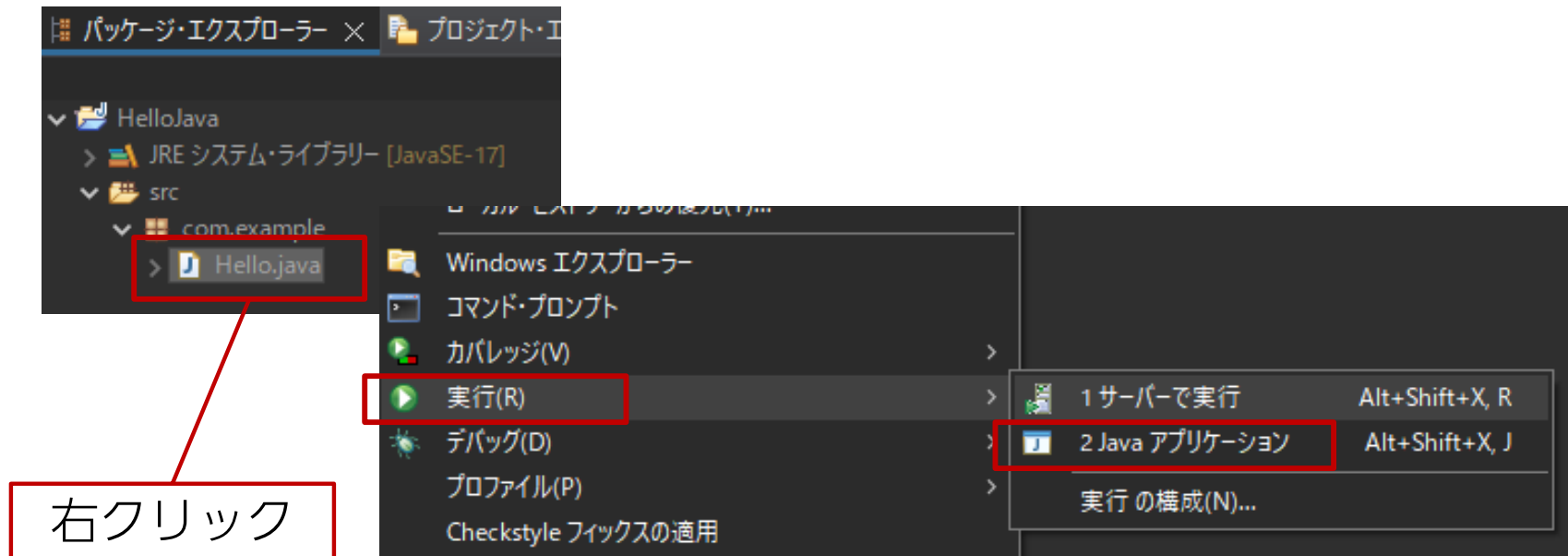
- 画面右側のエディタを使い、Hello.javaに以下のように記述する

```
package com.example;  
  
public class Hello {  
  
    public static void main(String[] args) {  
        System.out.println("Hello, Java!");  
    }  
}
```

赤字部分を追記する

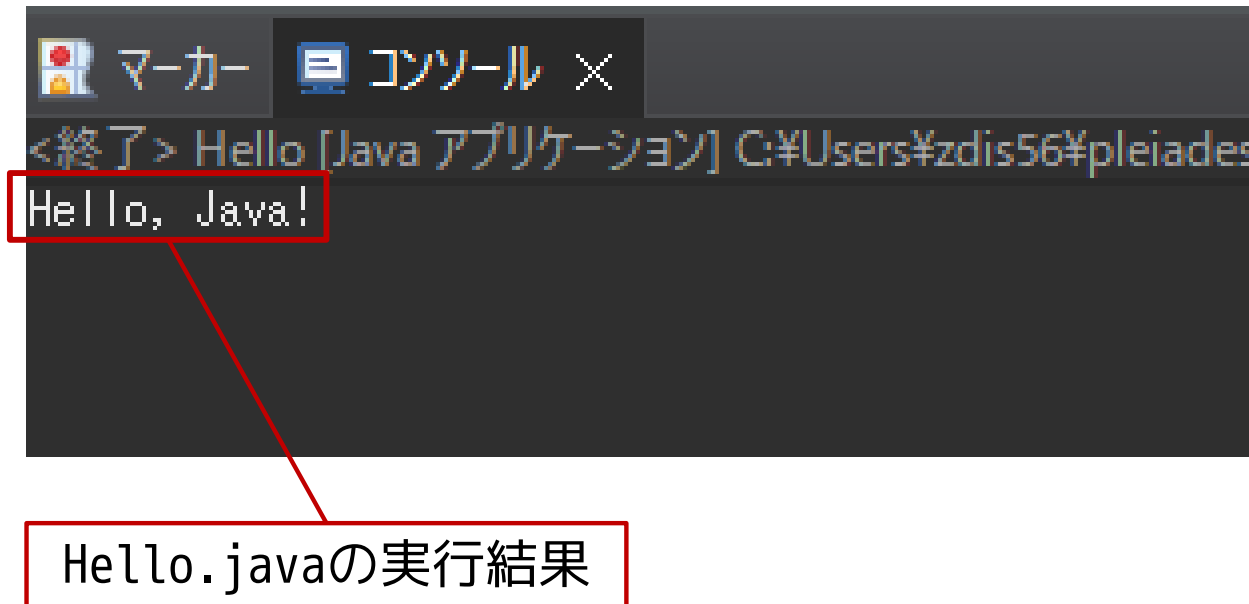
5. コンパイルと実行

- Hello.java上で右クリックし、
「実行」⇒「Javaアプリケーション」を選択
– Eclipseが自動的にコンパイルを行う



5. コンパイルと実行

- コンソールビューが開き、結果が表示される



補足：ファイル名の変更方法

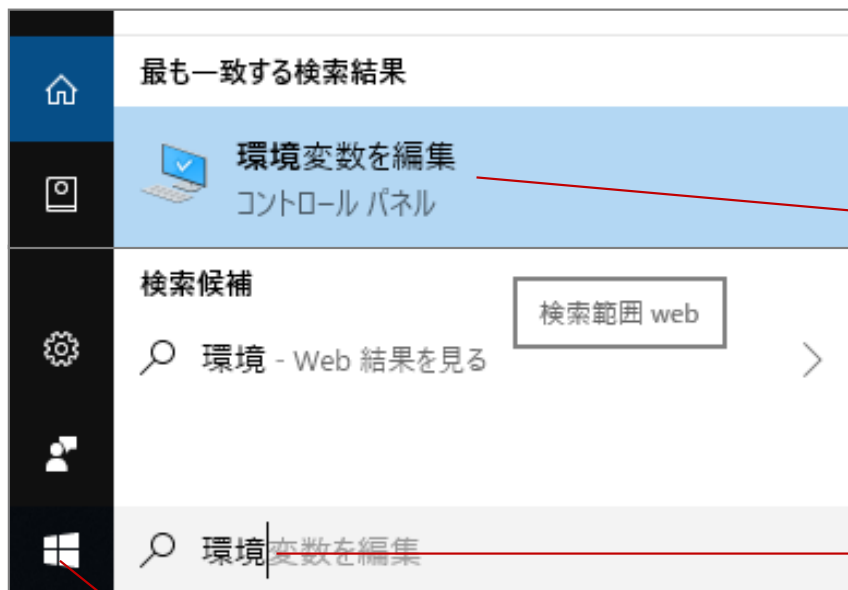
- ファイル名を変更する際は、ファイル名を右クリックし、リファクタリング → 名前変更
 - 新しい名前を入力し、完了ボタンを押す
 - 拡張子(.java) は入力しない

The screenshot shows the 'コンパイル単位名の変更' (Rename) dialog box in Eclipse. The title bar includes the Eclipse logo and the text 'コンパイル単位名の変更'. The dialog has a text field for '新しい名前(M):' containing 'Hello2'. Below this are four checkboxes: '参照の更新(R)' (checked), '類似の名前の変数とメソッドの更新(S)' (unchecked), 'コメントおよびストリング内のテキスト出現箇所の更新(T) (プレビューを強制)' (unchecked), and '非 Java テキスト・ファイル内の完全修飾名の更新(Q) (プレビューを強制)' (unchecked). To the right of the second checkbox is a '構成...' button. Below the checkboxes is a text field for 'ファイル名のパターン(P):' containing an asterisk (*). Below this field is a note: 'パターンはコンマで区切ります (* = 任意の文字列、? = 任意の文字)'. At the bottom are four buttons: '< 戻る(B)', '次へ(N) >', '完了(F)' (highlighted with a blue border), and 'キャンセル'.

参考：手動でのコンパイルと実行

準備：環境変数の設定

- コマンドプロンプトから、手動でコンパイルと実行を行う場合は、事前に環境変数の設定を行う必要がある



③ 環境変数を編集 をクリック

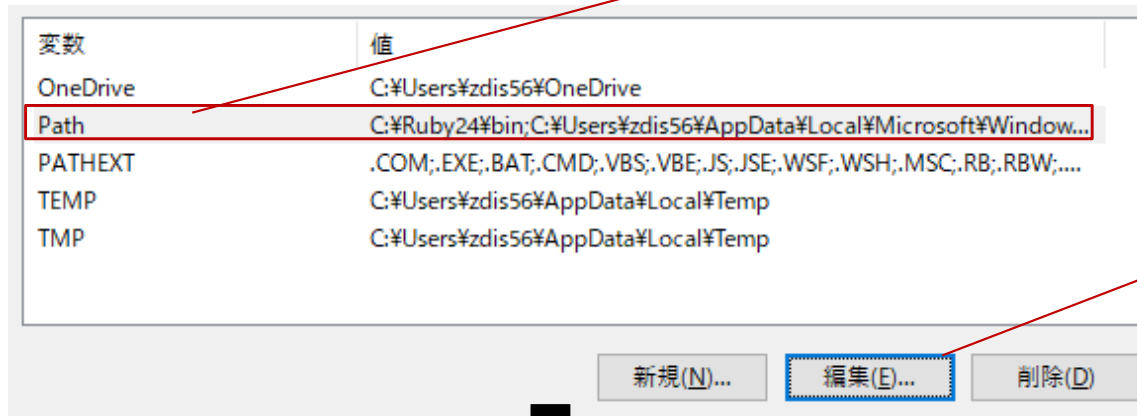
② 環境変数… と入力

① Windowsアイコンをクリック

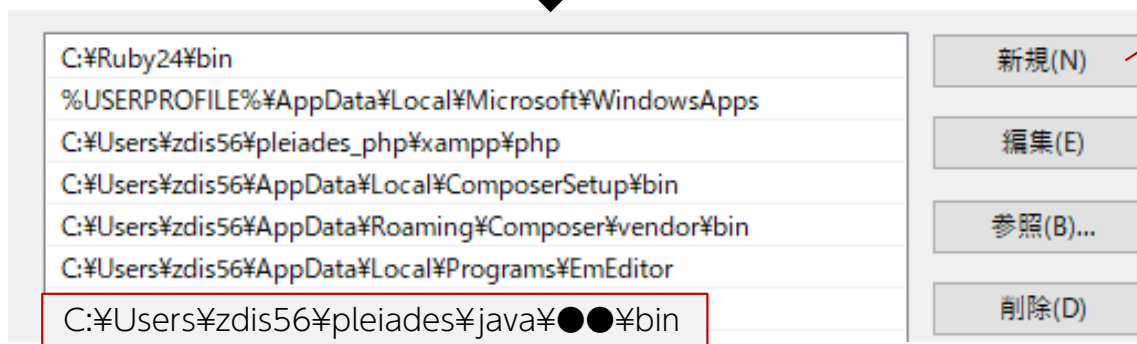
準備：環境変数の設定

- PathにインストールしたJDKのbinフォルダを指定する

① Pathをクリックして選択



② 編集 をクリック



③ 新規または参照で自身がインストールしたPleiadesのフォルダ内pleiades\java\●●\binを追加

17, 21などのバージョン

プログラムの記述

- デスクトップにHello.javaという名前のファイルを作成し、以下のように記述する

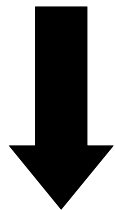
```
public class Hello {  
  
    public static void main(String[] args) {  
        System.out.println("こんにちは, Java!");  
    }  
  
}
```

コンパイル

- コンパイルは **javac** コマンドで行う

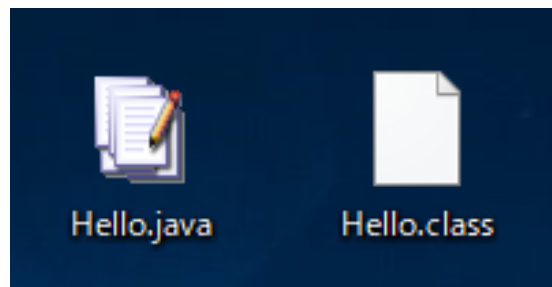
```
C:\Users\zdis56>cd Desktop  
C:\Users\zdis56\Desktop>javac Hello.java
```

① コマンドプロンプトを開き、cdコマンドでJavaのソースコードがあるディレクトリに移動



② **javac** **ファイル名.java** と入力し、Enter

.javaという拡張子を付ける



③ コンパイルが完了し、.class という拡張子のファイルが生成される

実行

- 実行は **java** コマンドで行う

```
C:\Users\zdis56\Desktop>javac Hello.java  
  
C:\Users\zdis56\Desktop>java Hello  
こんにちは, Java!
```

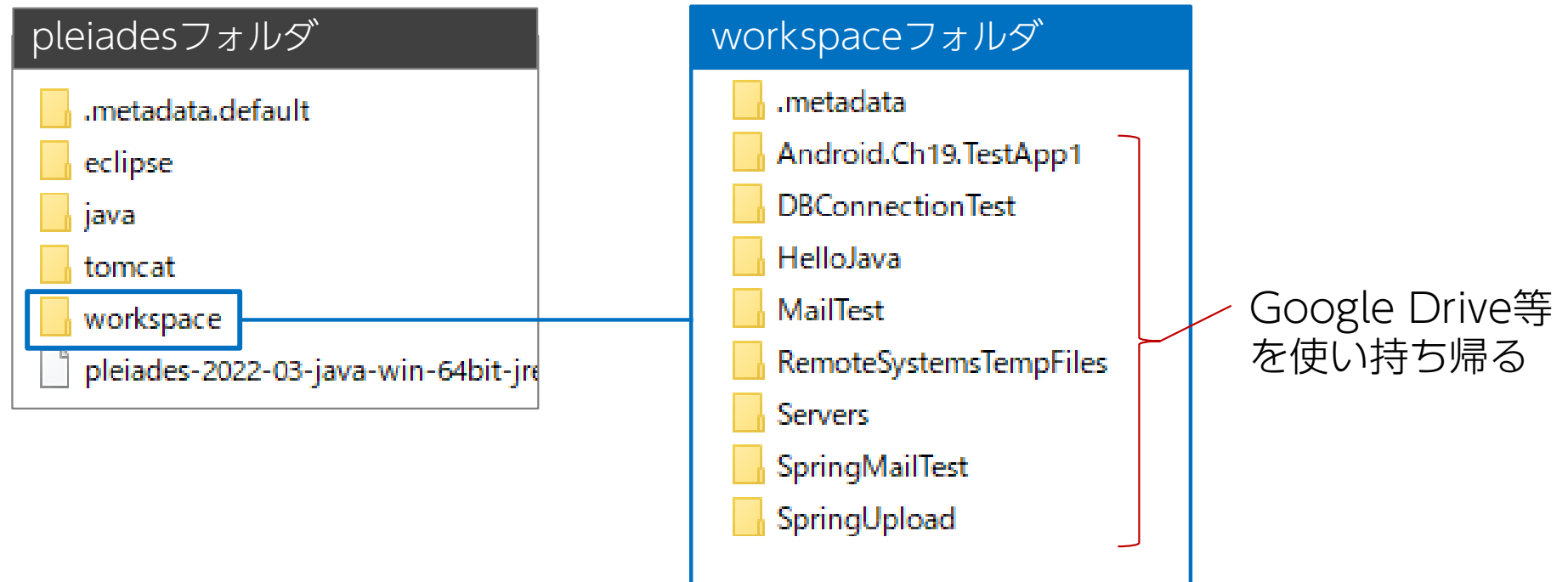
② 実行結果が表示される

① **java ファイル名**
と入力し、Enter
※ ファイル名に拡張子は付けない

データの持ち帰り方

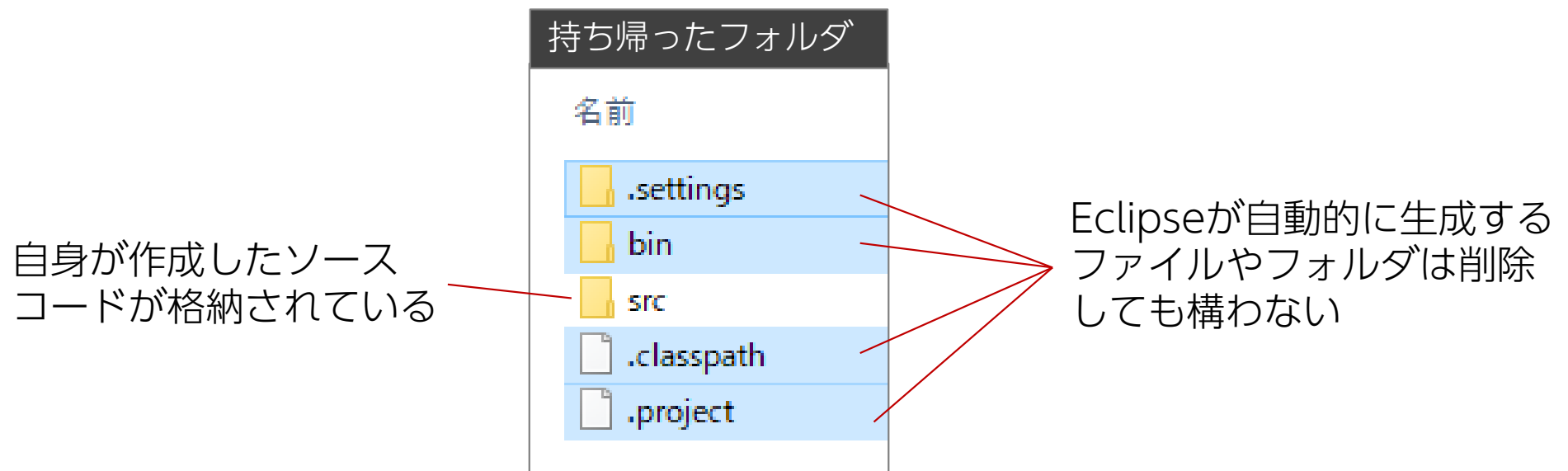
作業データの持ち帰り

- Eclipseで作成したプロジェクトは、workspaceフォルダに保存される



持ち帰ったデータのインポート

- 持ち帰ったプロジェクトフォルダは、どこに配置しても問題ない
 - workspaceフォルダに配置しなくてもよい



持ち帰ったデータのインポート

- ファイルメニューからインポートを選択
 - 一般 ⇒ フォルダーまたはアーカイブからプロジェクトをクリックし、持ち帰ったプロジェクトフォルダを選択する

