# ● 授業のポイント

① Composite パターンについて学習します。

## ● 学習項目

• Composite パターン

# ● J2Kad23D「Composite パターンへの道①(単品表示と一覧表示)」

# 1. Menultem (単品) と MenuList (一覧)

MenuItem と MenuList の表示を作成します。MenuItem は printItem メソッド、MenuList は printList メソッド で表示します。メソッドの仕様(名前)が異なるので、それぞれに対応した処理を作成する必要があります。

# ● J2Kad23C「Composite パターンへの道②(インターフェイスの共通化)」

## 1. print メソッド(インターフェイスの共通化)

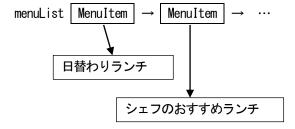
MenuItem の printItem メソッド、MenuList の printList メソッドの名前を print に変更します。print メソッドを持ったインターフェイス(課題では Component インターフェイス)を実装させれば、このインターフェイスを使って表示処理を共通化できます。

# ● J2Kad23B「Composite パターンへの道③ (参照の共通化)」

# 1. List<Component> (参照の共通化)

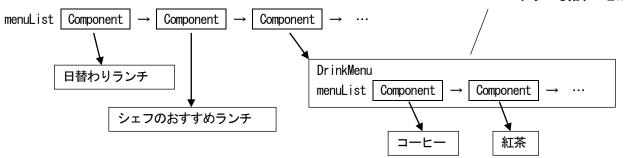
MenuList クラスの menuList に格納する参照を MenuItem から Component へ変更します。これにより menuList には MenuItem (単品) だけでなく MenuList (一覧) も登録することができるようになります。

#### Before : List<MenuItem>



#### After : List < Component >

MenuList クラスも指すことができる。



## 2. Composite パターン(容器と中身の同一視)

Composite パターンとは容器と中身を同一視して再帰的な構造を作るパターンです。例えば「フォルダ」の中に「ファイル」だけでなく「フォルダ」があったり、さらにその「フォルダ」の中にも「ファイル」や「フォルダ」があったりという構造です。課題では MenuList クラスが「容器」、MenuItem クラスが「中身」に該当します。

#### ● J2Kad23A「ファミレス ECC (完成版)」

## 1. AllMenu クラス(すべてのメニューの表示)

AllMenu クラスの作成はいたって簡単です。コンストラクタで MornigMenu と LunchMenu と DinnerMenu を add するだけです。 main メソッドでの表示も AllMenu クラスのインスタンスを生成するだけで OK です。 もし前回の J2Kad22X を作成していればコードを比較してみてください。

#### ● J2Kad23S「パッケージと Java ファイルの表示」

#### 1. フォルダとファイル

Composite の最も分かりやすい例がフォルダとファイルの関係です。ここでは J2Kad23 プロジェクトの src フォルダを例にフォルダとファイルの表示を作成します。Composite パターンが理解できたかどうかの確認用です。

## ● J2Kad23X「ガチャガチャマシーン」※次回、解答編の予定

### 1. State パターン

State パターンとは具体的な「もの」ではなく、「状態」をクラスとして表現するパターンです。同じ操作でも状態によって振る舞いが変化する場合に有効です。状態を個々のクラスで管理するため、コードがわかりやすくなります。