8章　ポリモリズムとパッケージ

目次

抽象クラス

Java言語では処理内容を記述しないメソッドや、それを持つクラスも定義できる。この処理内容を記述しないメソッドを抽象メソッド、抽象メソッドを持つクラスを抽象クラスと呼ぶ。

抽象クラスの構文
[修飾子] abstract class クラス名 {}

抽象メソッドの構文
[修飾子] abstract 戻り値の型 メソッド名(引数のメソッド);

インターフェース

公開するべき操作をまとめたクラスの仕様。実際の処理内容は記述せず、インスタンス化できない。宣言時はinterfaceキーワードを使用する。

抽象クラスの構文
[修飾子] interface インターフェース名 {}

インターフェースでの変数とメソッド

インターフェース内で変数を宣言すると、暗黙的にpublic static final修飾子が付与される。宣言時と同時に初期化しておく必要があり、初期化しないとコンパイルエラーになる。

インターフェースの実装クラス

インスタンス化できないインターフェースをインスタン化して使用するためには、全てのメソッドをオーバーライドしたクラスを定義する必要がある。これを実装クラスと呼ぶ。宣言にはimplementsキーワードを使用する。

抽象クラスの構文
[修飾子] class クラス名 {} implements インターフェース名{}

インターフェースの継承

インターフェースを継承したサブインターフェースを使用することができる。継承関係を持つためにサブインターフェースを宣言する際は、クラスと同様extendsキーワードを使用する。

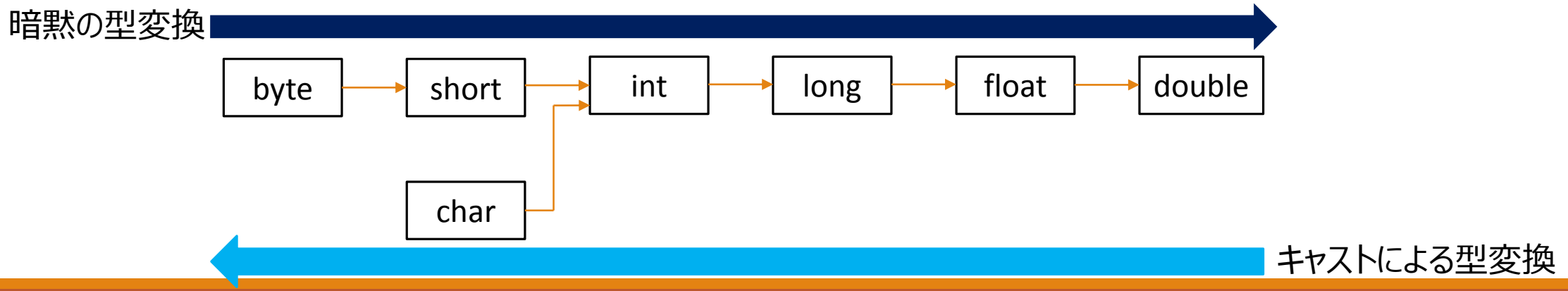
サブインターフェースを実現したクラスが抽象クラスでなく具象クラスの場合、スーパーインターフェース、サブインターフェースの全てのメソッドをオーバーライドする必要がある。

基本データ型の型変換

基本データ型で宣言した変数には宣言時の型で扱える範囲内のデータであれば、型変換の仕組みにそれを代入できる。異なる型の値の代入には2種類の型の変更ルールがある。

・暗黙の型変換

・キャストによる型変換



参照型の型変換

暗黙の型変換

サブクラスのオブジェクトをスーパークラスを型として宣言した変数で扱える。

実装クラスのオブジェクトをインターフェースを型とし宣言した変数で扱える。

キャストによる型変換

スーパークラスを型として宣言した変数で参照しているサブクラスの
オブジェクトサブクラスを型地して宣言した変数で扱うにはキャストを用いる。
インターフェース型として宣言した変数で参照している実装クラスの
オブジェクトを実装クラスを型として宣言した変数で扱うにはキャストを用いる。

ポリモフィズム

Java言語ではポリモフィズムを実現するために、抽象クラスや具象クラス、インターフェースや実装クラスを活用している。

同じ名前のメソッドの呼び出しであってもオブジェクトごとに異なる機能を提供する。

パッケージとは

どのような状況になってもクラスが衝突しないようJava言語はパッケージと言う仕組みを提供している。クラスをパッケージに含めることをパッケージ化という。

パッケージ化

```
package パッケージ名;
class x {･･･}
```

パッケージ化されたクラスのコンパイルと実行

パッケージ化されたクラスは、クラス名単体では扱うこつができない。「パッケージ名＋クラス名」で扱う必要がある。また、そのパッケージ名に対応したフォルダを用意し、そのフォルダの中にクラスのファイルを保存しておく必要がある。

import文

異なるパッケージに属するクラスをパッケージ化しなくてもクラス名だけで利用可能にする方法をインポートと呼ぶ。クラスをインポートするときはソースファイルの先頭でimport文を記述し、importキーワードに続き、パッケージ名も含めた完全な名前で利用するクラスを指定する。