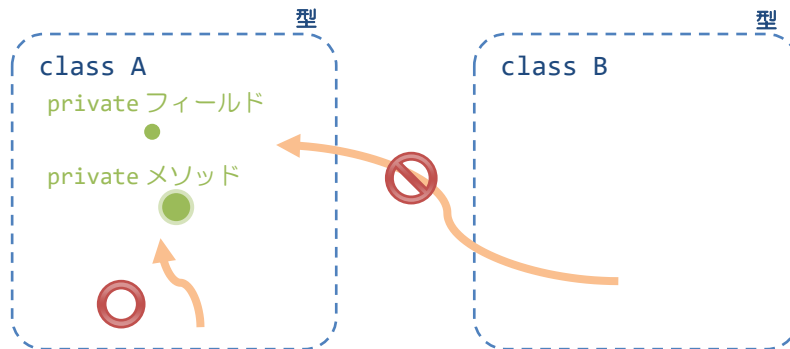


§ アクセス制限

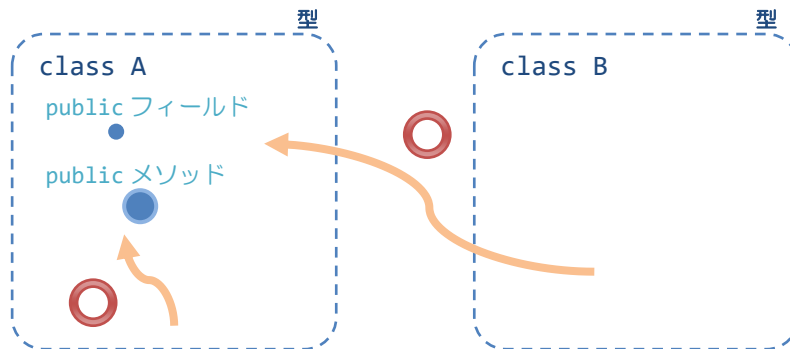
■private メンバー 同じクラスからのみアクセスできるメンバーです

■宣 言 メンバーの宣言に **private** 修飾子を付けます



■public メンバー どこからでもアクセスできるメンバーです

■宣 言 メンバーの宣言に **public** 修飾子を付けます



■その他の修飾子

どこからならアクセス可能か?		private	指定ない	protected	public
同じクラス		○	○	○	○
他のクラス	同じパッケージ内のサブクラス	×	○	○	○
	同じパッケージ内の一般クラス	×	○	○	○
	他のパッケージ内のサブクラス	×	×	○	○
	他のパッケージ内の一般クラス	×	×	×	○

※パッケージについては13章に説明があります

§ カプセル化

■ 抽象データ型

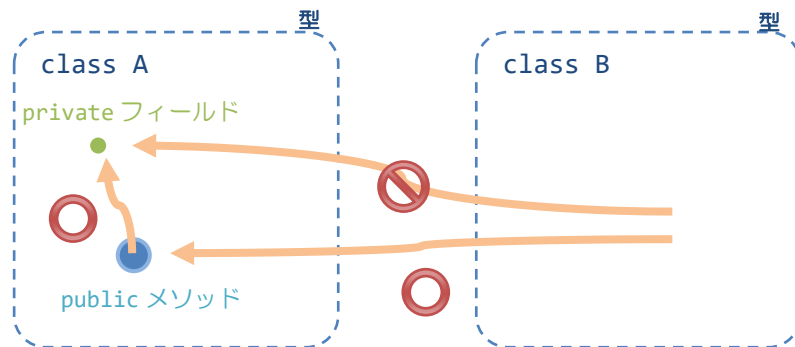
データとそれにアクセスする手続きを一つにまとめたデータ型です
抽象データ型を実現する方法の1つがクラスです

■ カプセル化

抽象データ型を用いることにより内部のデータへのアクセスを与えられた
手続きを用いてのみ可能にして、内部の細かなデータやその構成を
外部から隠蔽することです

クラスでは、フィールドに `private` 修飾子を、メソッドに `public` 修飾子をつけ、`private`
フィールドにアクセスする時には `public` メソッドを用いることでカプセル化を実現します

`public` メソッドに値のチェック機能を付けて `private` フィールドを保護するデータの保護
の他、データの保守や独立性に貢献します



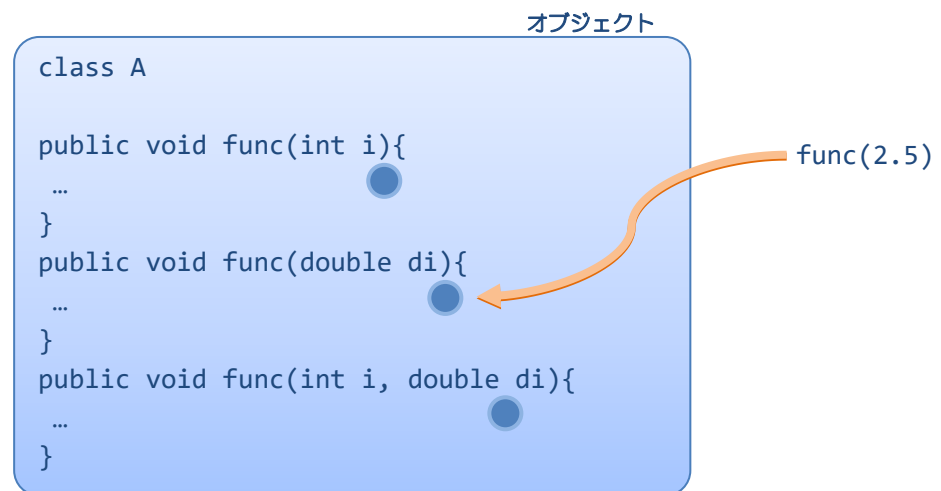
§ オーバーロード

■ オーバーロード

1つのメソッドに複数の機能を持たせることです

■ 宣言

同じ名前のメソッドを同じクラス内に定義します
各メソッドの引数の型・個数は異なります (戻り値は同じでもよいです)
※メソッドの呼出し時に、引数のパターンからどのメソッドが判別します



■ ポリモーフィズム (多態性/多様性)

一つの対象が状況に応じて別々の働きをすることです
ポリモーフィズムの実現の1つがメソッドのオーバーロードです