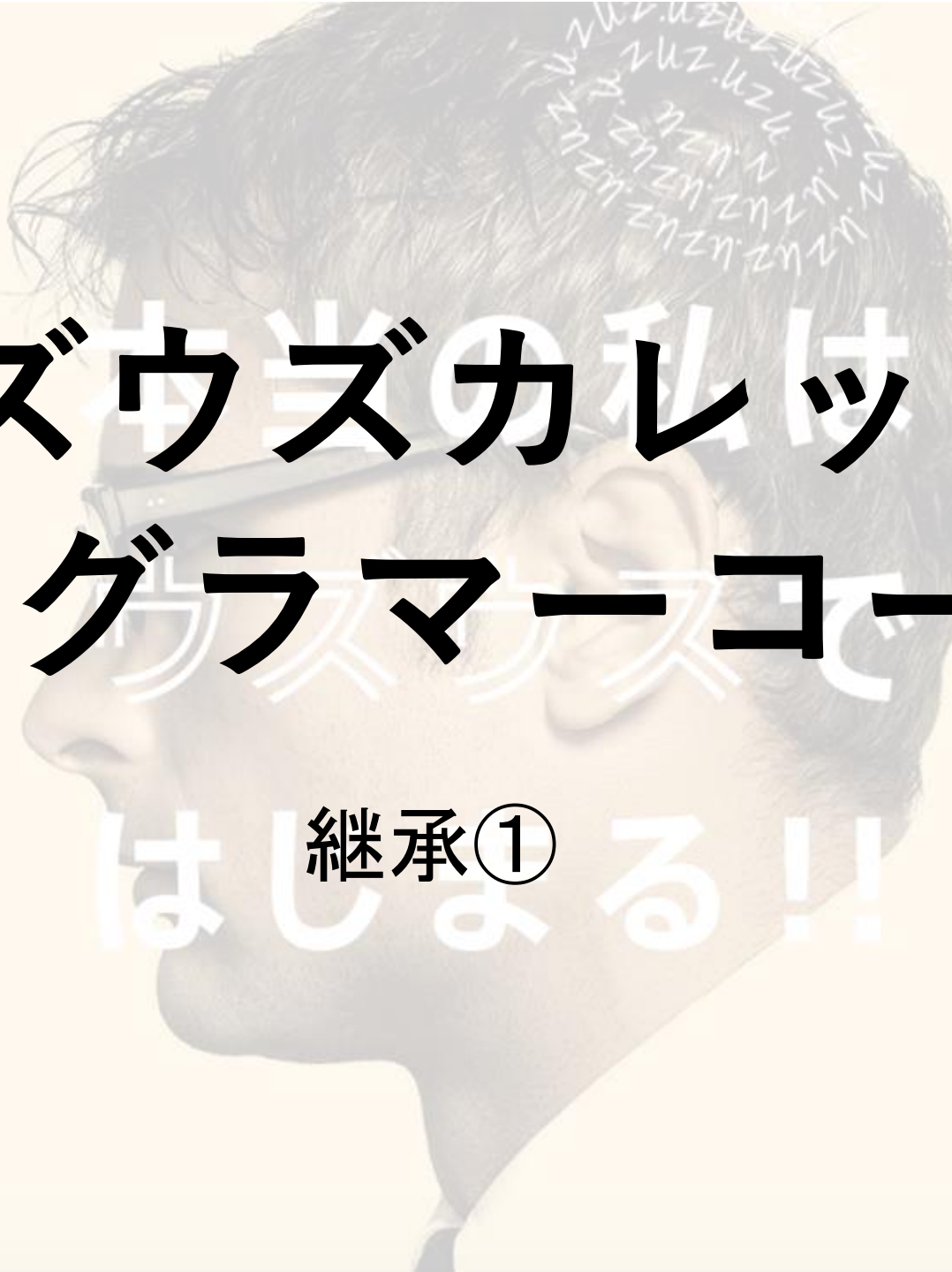
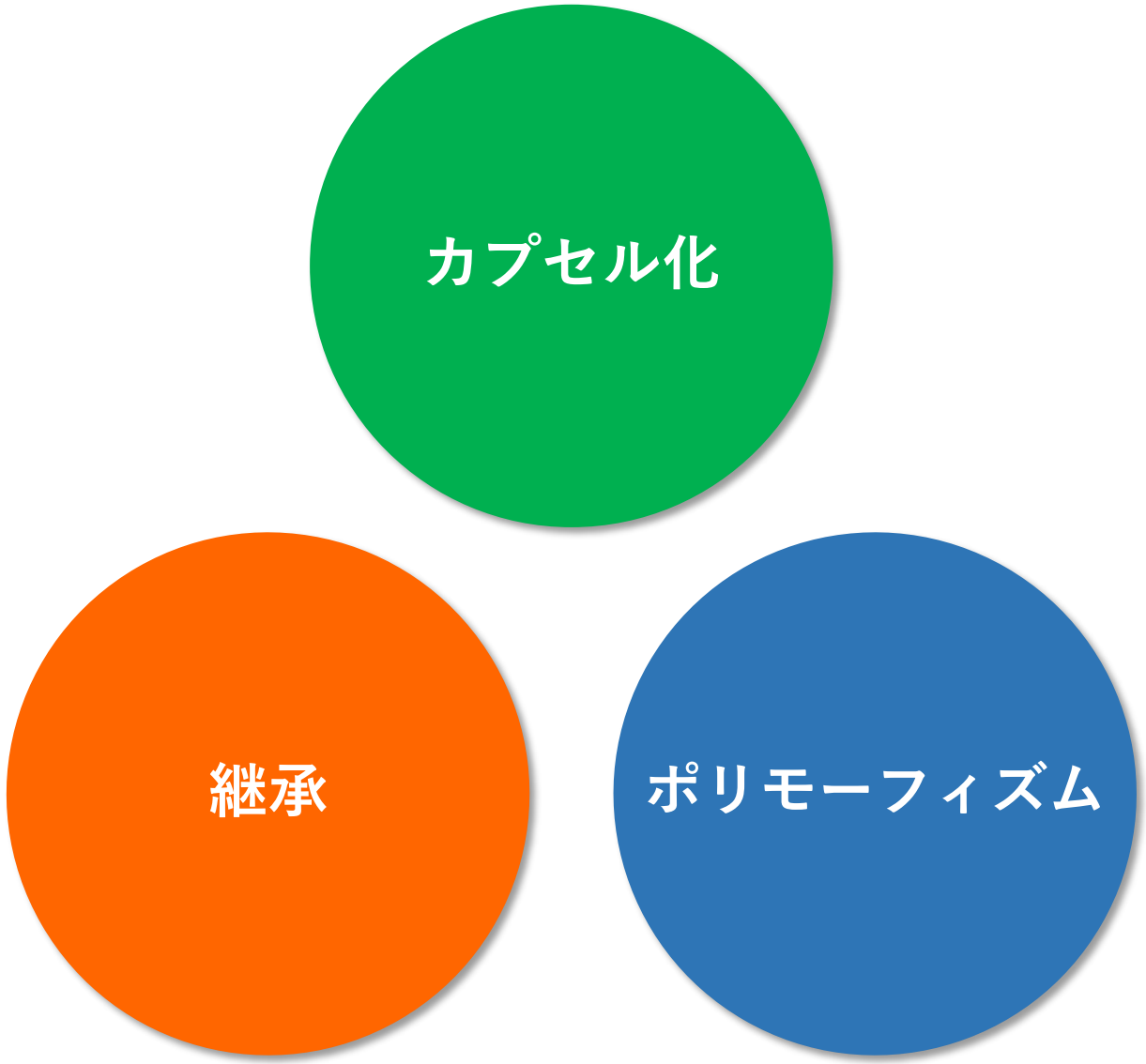


ウズウズカレッジ
プログラマーコース

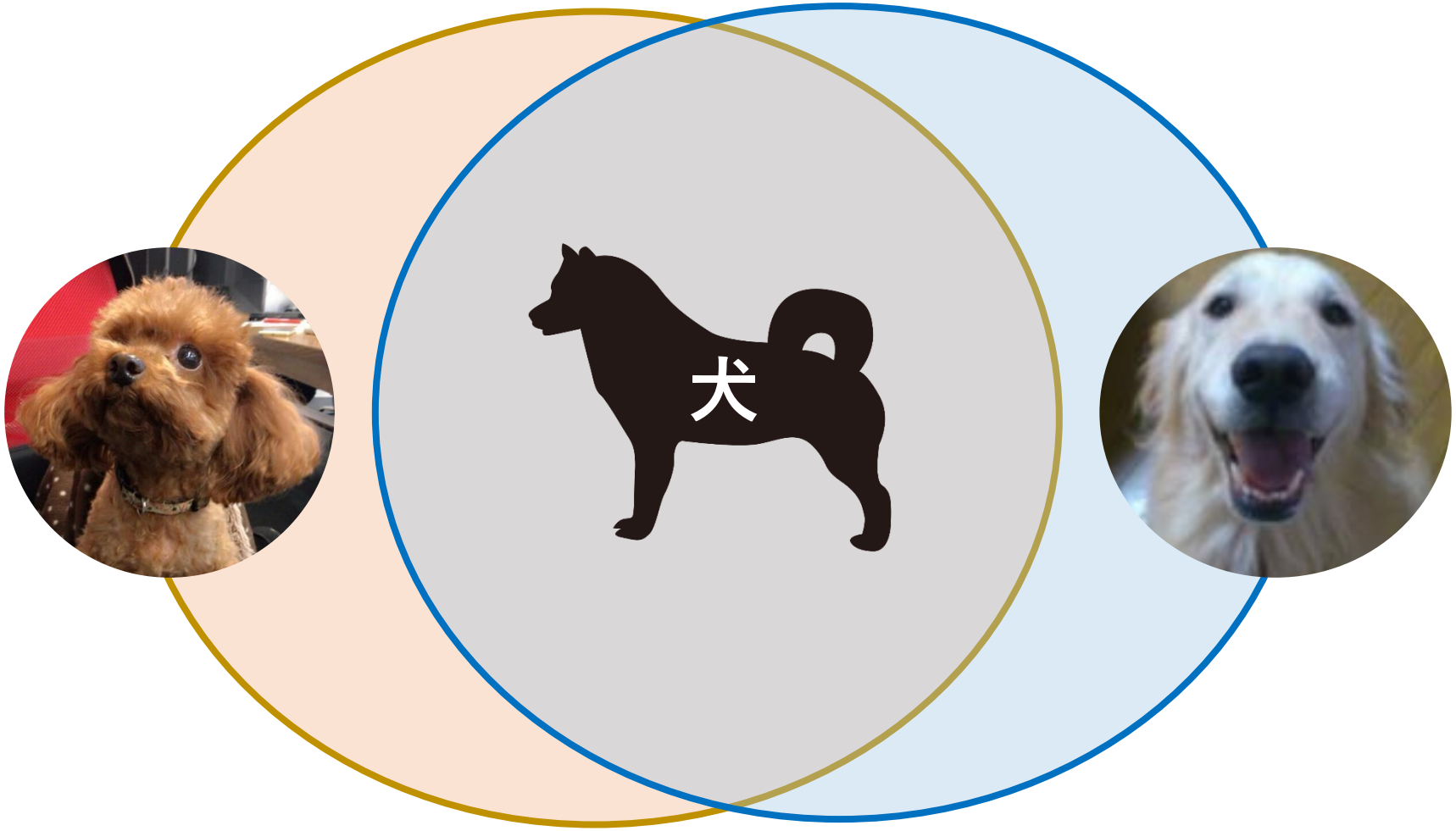
継承①

～オブジェクト指向の3大要素～

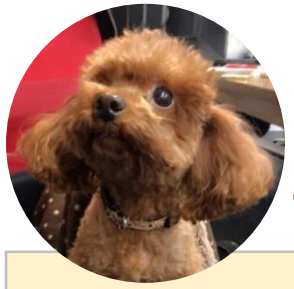


～共通のメンバを多く持つクラスの実装～

～共通のメンバを多く持つクラスの実装～



～共通のメンバを多く持つクラスの実装～



ToyPoodleクラス

[属性]
- **恒温動物**
- **四足**
- 小さい

[機能]
- **肺で呼吸する**
- **吠える**
- あざとい態度をとる

全く同じメンバ情報を書かなければならない

GoldenRetrieverクラス

[属性]
- **恒温動物**
- **四足**
- 大きい

[機能]
- **肺で呼吸する**
- **吠える**
- たくさん食べる

～共通のメンバを多く持つクラスの実装～

Dogクラス

[属性]
恒温動物
四足

[機能]
肺で呼吸する
吠える

```
class ToyPoodle extends Dog {
```

```
class GoldenRetriever extends Dog {
```

継承
犬クラスが持つ
メンバ情報を自動で反映

ToyPoodleクラス

[属性]
小さい

[機能]
あざとい態度をとる

GoldenRetrieverクラス

[属性]
大きい

[機能]
たくさん食べる

～継承を活用したシステムの構築～

システム全体で
同じようなソースコードが
存在しない！

[属性]
恒温動物

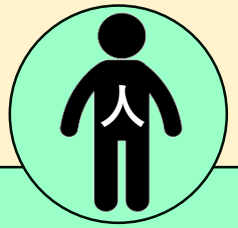
[機能]
肺で呼吸する

継承
哺乳類クラスで定義
されたメンバ情報を
自動で反映

[属性]
四足

[機能]
吠える

継承
犬クラスで定義された
メンバ情報（哺乳類含む）
を自動で反映



[属性]
二足

[機能]
手先を器用に使う

トイプードル

[属性]
小さい

[機能]
あざとい態度をとる

ゴールデンレトリーバー

[属性]
大きい

[機能]
たくさん食べる

～継承を活用したシステムの構築～

哺乳類

[属性]
恒温動物

[機能]
肺で呼吸する

継承
哺乳類クラスで定義されたメンバ情報を自動で反映

肺呼吸！
くるしい！

犬

[属性]
四足

[機能]
吠える

継承
犬クラスで定義されたメンバ情報（哺乳類含む）を自動で反映

肺呼吸！
くるしい！

トイプードル

[属性]
小さい

[機能]
あざとい態度をとる

人

[属性]
二足

[機能]
手先を器用に使う

肺呼吸！
くるしい！

ゴールデンレトリーバー

[属性]
大きい

[機能]
たくさん食べる

肺呼吸
くるしい！

～継承を活用したシステムの構築～

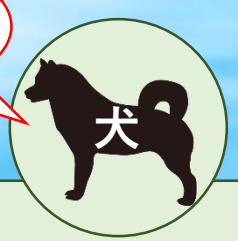
[属性]
恒温動物

[機能]
エラで呼吸する

どこを修正すれば
よいか一目瞭然
&
一カ所修正すれば
すべての継承先の
クラスに反映される

継承
哺乳類クラスで定義
されたメンバ情報を
自動で反映

エラ呼吸！
くるしくない！



[属性]
四足

[機能]
吠える

エラ呼吸！
くるしくない！

[属性]
二足

[機能]
手先を器用に使う

継承
犬クラスで定義された
メンバ情報（哺乳類含む）
を自動で反映

エラ呼吸！
くるしくない！

トイプードル

[属性]
小さい

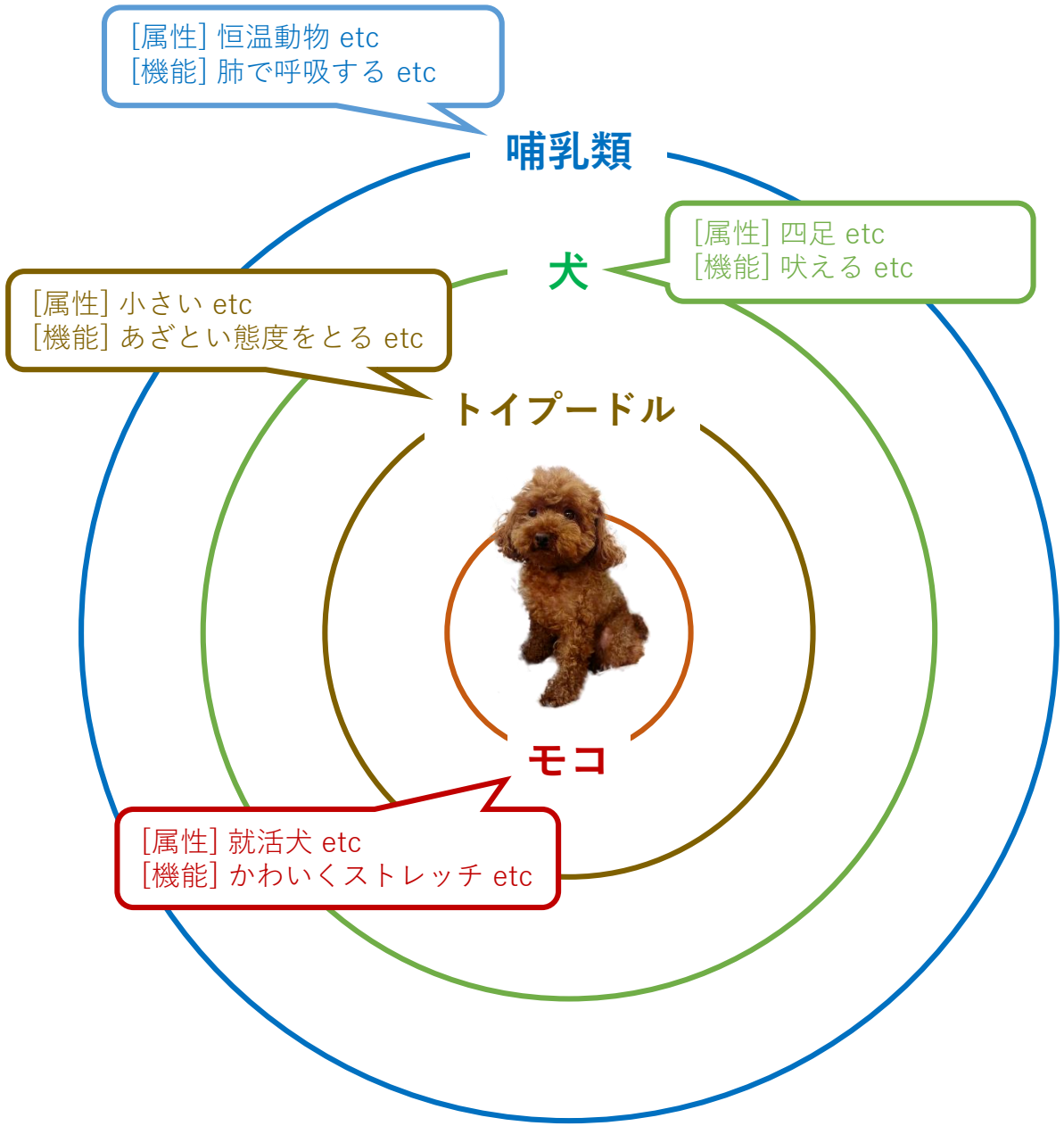
[機能]
あざとい態度をとる

ゴールデンレトリーバー

エラ呼吸！
くるしくない！

[属性]
大きい

[機能]
たくさん食べる



≪継承≫

□あるクラスのメンバを受け継いで別のクラスを作ることを**継承**と言います。この機能を利用することでシステムの機能拡張を**差分のみのプログラミング**で実現することが可能になります。

□継承元となるクラスを**スーパークラス**、継承先となるクラスを**サブクラス**と言います。

□継承の利点としていは以下のようなものが挙げられます。

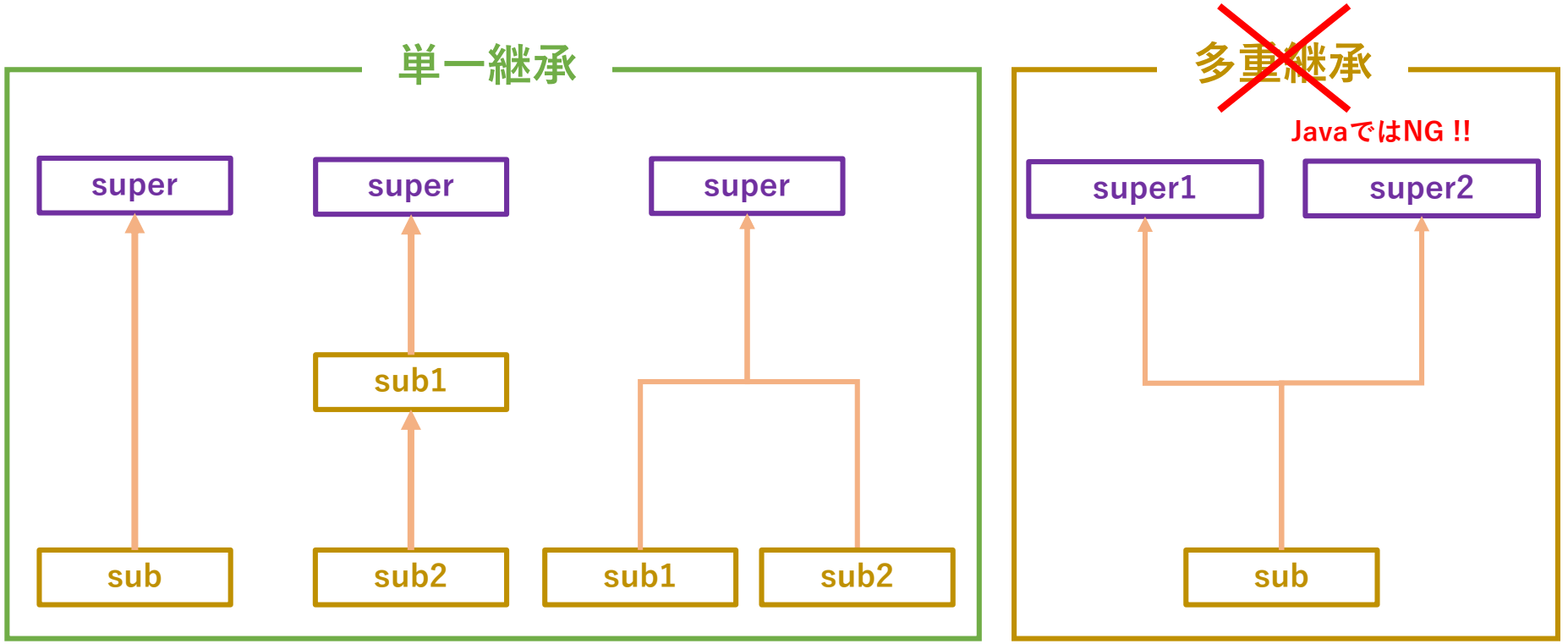
- **ソースコードの重複を防ぐ**
- **機能拡張が容易になる**
- **プログラム構造の理解が容易になる**
- **仕様変更への対応が容易になる**

□あるクラスを継承する場合、サブクラスのクラス定義の箇所で以下のように記述します。

class サブクラス名 extends スーパークラス名{　}

□Javaでは**単一継承のみ**が認められており、1つのクラスが複数のスーパークラスを持つことはできません。
あるスーパークラスを継承したクラスを更に継承するといった、**階層的な継承**は可能です。

～複数のスーパークラスを持つことはできない～



～継承関係にあるオブジェクトのイメージ～

```
class Sample2_04_1_Mammals {
    //---フィールド----------------------------------------
    String group = "哺乳類" ; //分類
    //---メソッド----------------------------------------
    //肺呼吸
    void breatheLungs(){
        System.out.println("ハァハァ！（肺で呼吸）") ;
    }
}
```

```
class Sample2_04_1_Dog extends Sample2_04_1_Mammals{
    //---フィールド----------------------------------------
    String type = "イヌ科" ; //分類
    //---メソッド----------------------------------------
    //吠える
    void bark(){
        System.out.println("ワンワン！") ;
    }
}
```

```
class Sample2_04_1_Moco extends Sample2_04_1_Dog {
    //---フィールド----------------------------------------
    String name = "MOCO" ; //名前
    //---メソッド----------------------------------------
    //かわいくストレッチ（UZUZの就活犬MOCOちゃんの特技）
    void cuteStretch(){
        System.out.println("かわいくストレッチ！") ;
    }
}
```