Java基礎演習
(オブジェクト指向編)

インスタンス化

インスタンス化

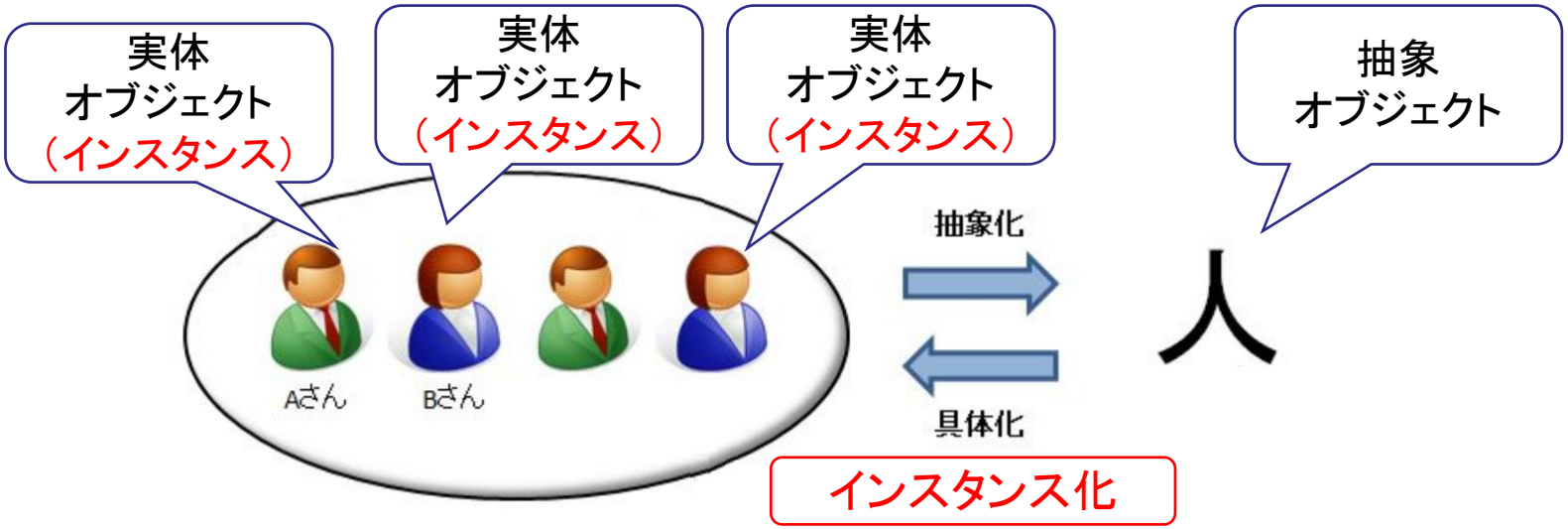
「インスタンス化」の概念について、「基礎構文編」でも少し触れました。

ここではあらためて「インスタンス化」の概念に触れながら、プログラミングの方法を学んでゆきましょう。

インスタンス化

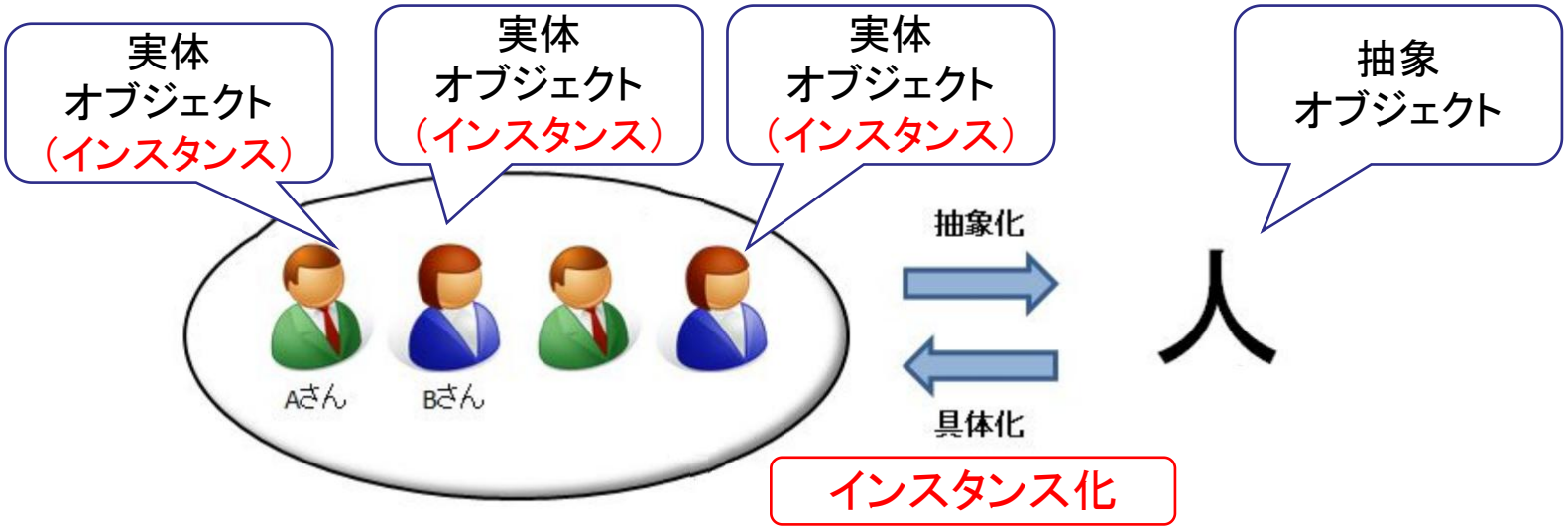
「オブジェクト指向」の世界では「実体オブジェクト」のことを、「インスタンス」といいます。

また、「抽象オブジェクト」が具体化されることを、「インスタンス化」といいます。



インスタンス化

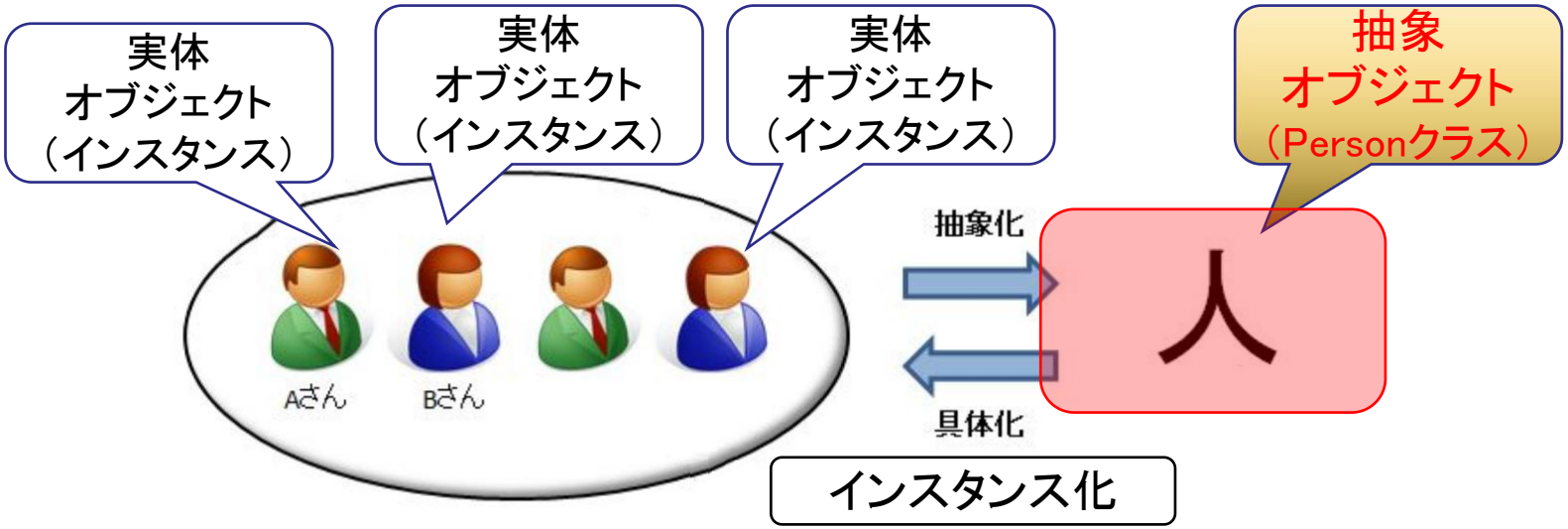
「インスタンス化」は何度もおこなうことができます。
「オブジェクト指向プログラミング」では、「人」という抽象オブジェクトを使って、
具体的な実体オブジェクト（インスタンス）を何人も増やすことができます。
これが、下図のAさんやBさんです。



インスタンス化

たとえば、「人」をPersonクラスとしましょう。

このPersonクラスをもとに、AさんやBさんをプログラミングすることができます。
次のページを参考にしながらPersonクラスを作成してみましょう。



インスタンス化

以下をプログラムしてみましょう。
■ファイル名：Person.java

```
public class Person {
    String name;
    String age;
}
```

インスタンス化

次に、コマンドプロンプトやターミナルでコンパイル（javac）してみましょう。
まだ実行する必要はありません。

```
C:¥Users¥internous> javac Person.java

C:¥Users¥internous> dir

Person.java
Person.class

C:¥Users¥internous>
```

ここでコンパイルします

javaファイルとclassファイルが
あることを確認します
（Macではlsと入力して下さい）

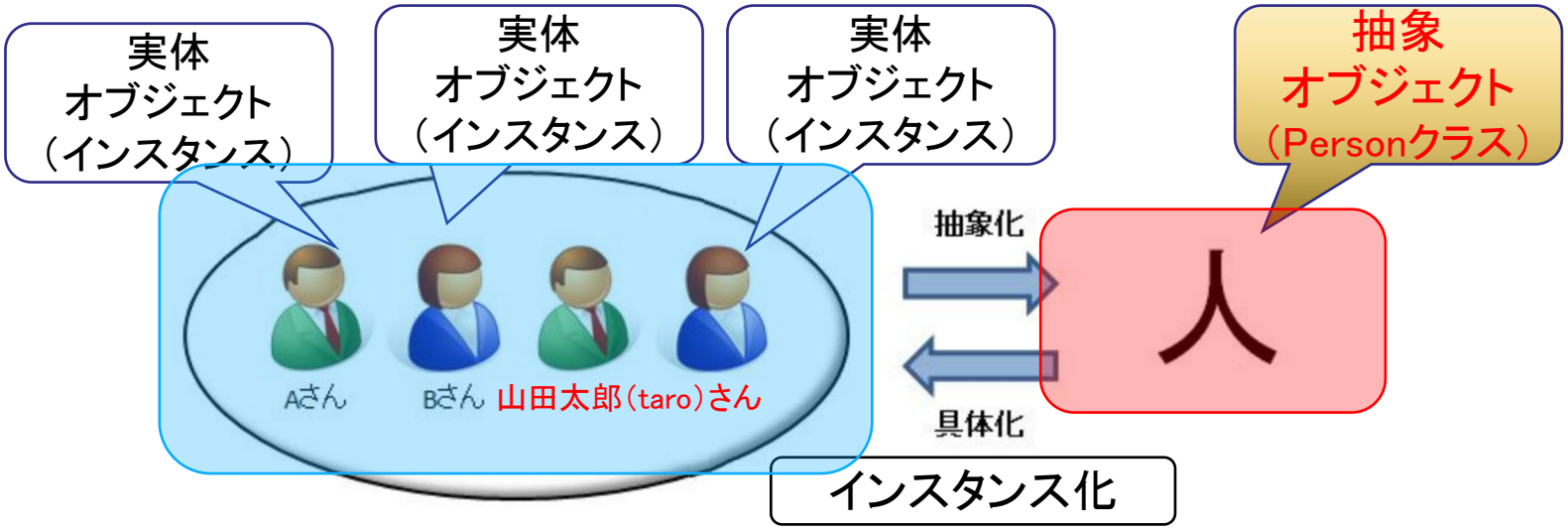
まだ実行する必要は
ありません

インスタンス化

ここでPersonクラス、つまり抽象オブジェクトとなるクラスを作成できました。

このPersonクラスをもとに、AさんやBさんをプログラミングすることができます。
今回は例として、山田太郎（taro）さんをプログラミングしてみます。

次のページを参考にしながらPersonクラスを作成してみましょう。



インスタンス化

以下をプログラムしてみましょう。
■ファイル名：Test.java

```
public class Test {
  public static void main(String[] args) {
    Person taro = new Person();
    taro.name = "山田太郎";
    taro.age = 20;
    System.out.println("名前:" + taro.name);
    System.out.println("年齢:" + taro.age);
  }
}
```

インスタンス化

次に、コマンドプロンプトやターミナルでコンパイル（javac）してみましょう。
コンパイルができたら実行してみましょう。

```
C:¥Users¥internous> javac Test.java

C:¥Users¥internous> dir

Person.java
Person.class
Test.java
Test.class

C:¥Users¥internous>java Test

名前：山田太郎
年齢：20
```

ここでコンパイルします

javaファイルとclassファイルが
あることを確認します
（Macではlsと入力して下さい）

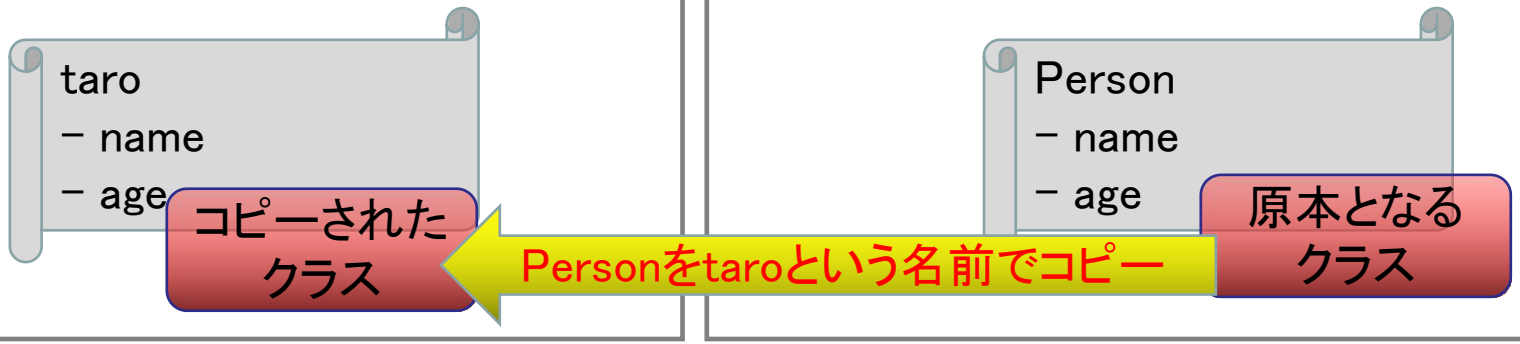
ここで実行してみましょう

インスタンス化

以下のTestクラスでは、Personクラスを原本として、これをtaroという名前でオブジェクトをコピーしています。

```
public class Test {
  public static void main(String[] args) {
    Person taro = new Person();
    taro.name = "山田太郎";
    taro.age = 20;
    System.out.println("名前:" + taro.name);
    System.out.println("年齢:" + taro.age);
  }
}
```

```
public class Person {
    String name;
    String age;
}
```

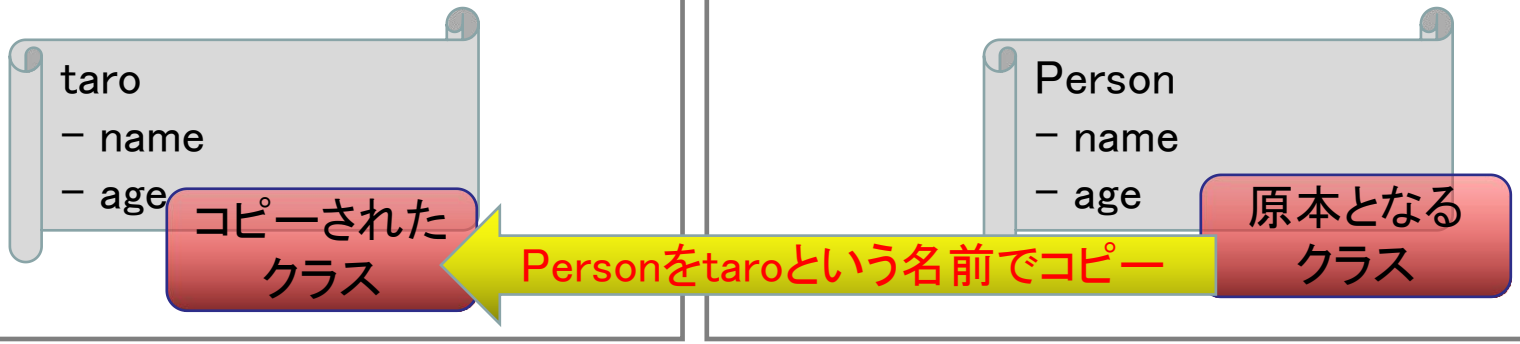


インスタンス化

このコピーは、以下の「Person taro = new Person()」という部分でおこなっています。

```
public class Test {
  public static void main(String[] args) {
    Person taro = new Person();
    taro.name = "山田太郎";
    taro.age = 20;
    System.out.println("名前:" + taro.name);
    System.out.println("年齢:" + taro.age);
  }
}
```

```
public class Person {
    String name;
    String age;
}
```

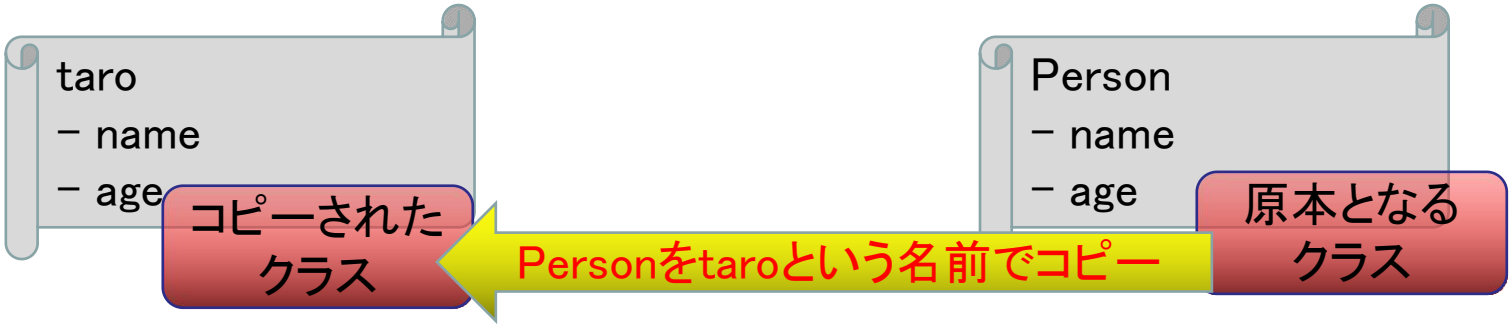


インスタンス化

以下のような記述をすると、クラスをコピーして使いまわせるようになります。

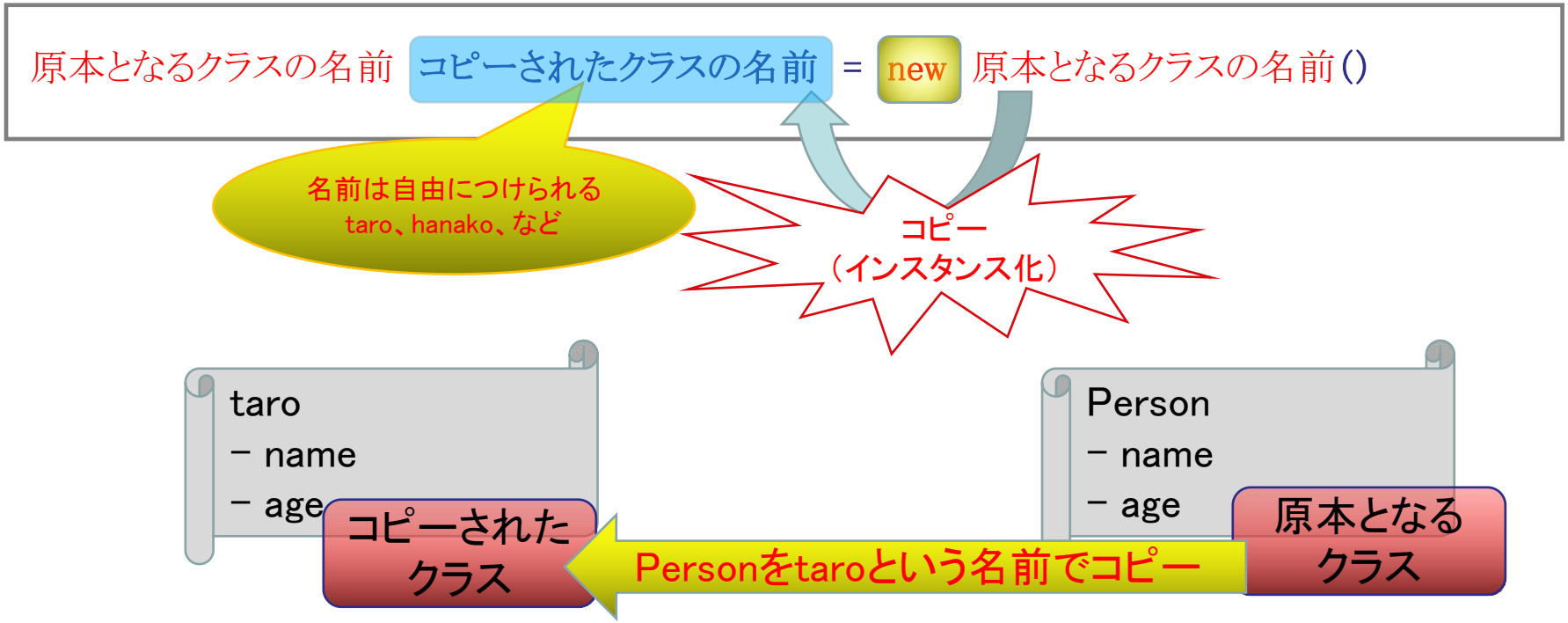
このようにクラスをコピーすることを「インスタンス化」といいます。

原本となるクラスの名前　コピーされたクラスの名前　＝　new　原本となるクラスの名前()



インスタンス化

「インスタンス化」は、”new”という言葉を使ってあたらしくクラスをコピーします。
また、コピーされたクラスの名前は自由につけることができます。

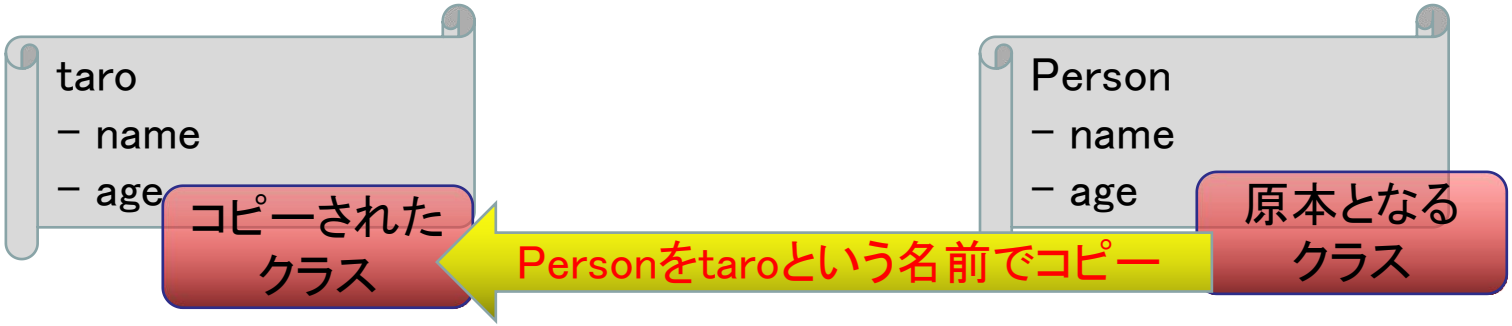


インスタンス化

また、こうしてコピーされたクラスは、「インスタンス」といいます。

原本となるクラスの名前 コピーされたクラスの名前 = new 原本となるクラスの名前()

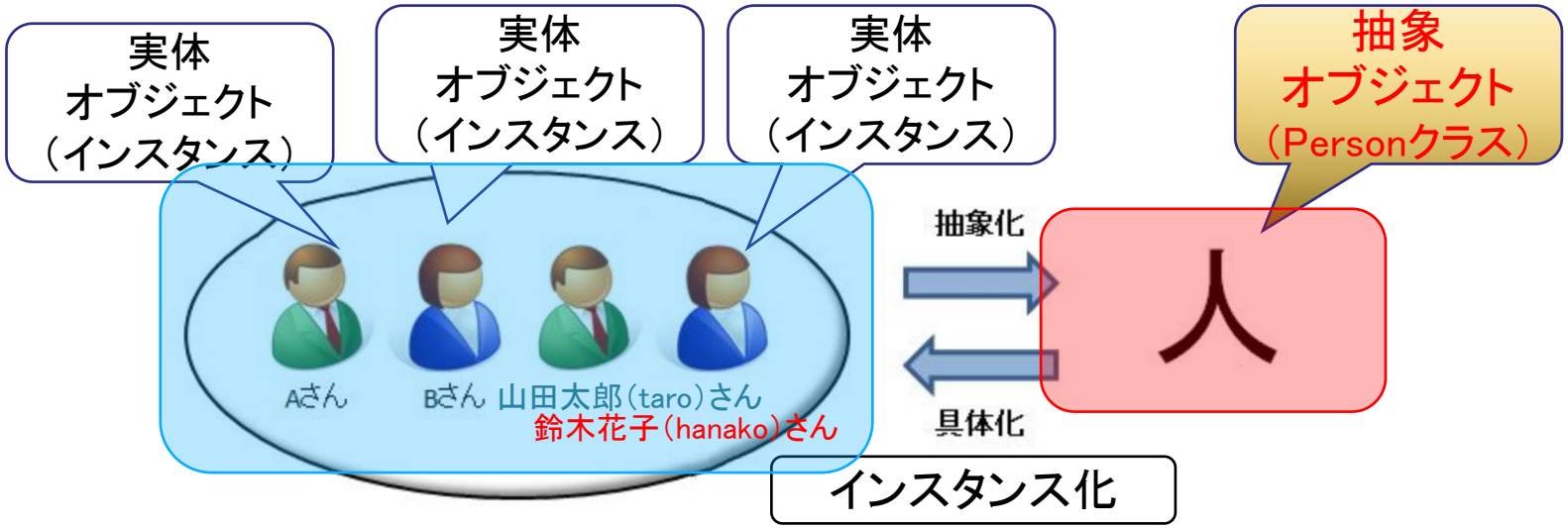
「インスタンス」



インスタンス化

この「インスタンス化」は、さまざまなクラスの中で何度も使うことができます。
今回は例として、Testクラスに鈴木花子（hanako）さんを追加してみます。
次のページを参考にしながらTestクラスを書き換えてみましょう。

原本となるクラスの名前　コピーされたクラスの名前　＝　new　原本となるクラスの名前()



インスタンス化

以下のようにプログラムを追加してみましょう。
■ファイル名：Test.java

```
public class Test {
  public static void main(String[] args) {
    Person taro = new Person();
    taro.name = "山田太郎";
    taro.age = 20;
    System.out.println("名前" + taro.name);
    System.out.println("年齢" + taro.age);

    Person hanako = new Person();
    hanako.name = "鈴木花子";
    hanako.age = 18;
    System.out.println("名前" + hanako.name);
    System.out.println("年齢" + hanako.age);
  }
}
```

インスタンス化

次に、コマンドプロンプトやターミナルでコンパイル（javac）してみましょう。
コンパイルができたら実行してみましょう。

```
C:¥Users¥internous> javac Test.java

C:¥Users¥internous> dir

Person.java
Person.class
Test.java
Test.class

C:¥Users¥internous>java Test

名前：山田太郎
年齢：20
名前：鈴木花子
年齢：18
```

ここでコンパイルします

javaファイルとclassファイルが
あることを確認します
（Macではlsと入力して下さい）

ここで実行してみましょう

インスタンス化

このように、何度もインスタンス化はプログラムすることができます。

```
public class Test {
  public static void main(String[] args) {
    Person taro = new Person();
    taro.name = "山田太郎";
    taro.age = 20;
    System.out.println("名前" + taro.name);
    System.out.println("年齢" + taro.age);

    Person hanako = new Person();
    hanako.name = "鈴木花子";
    hanako.age = 18;
    System.out.println("名前" + hanako.name);
    System.out.println("年齢" + hanako.age);
  }
}
```

```
public class Person {
    String name;
    String age;
```

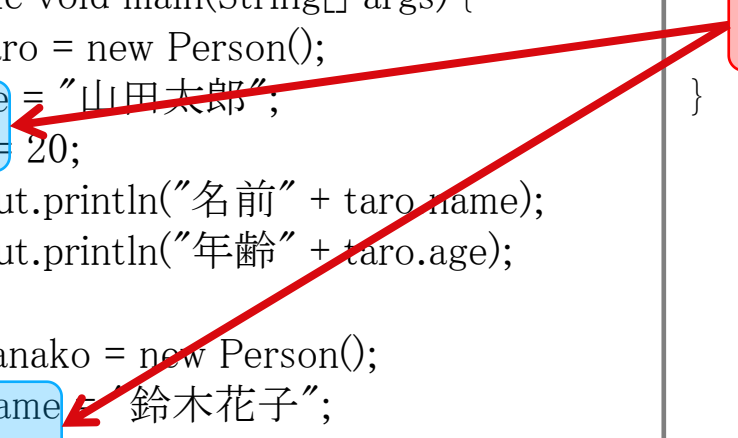
インスタンス化

また、各「インスタンス」に含まれる変数やメソッドは、”.”(ドット)をつけることで使用することができます(taro.nameやhanako.age、taro.run()やhanako.talk()など)。

```
public class Test {
  public static void main(String[] args) {
    Person taro = new Person();
    taro.name = "山田太郎";
    taro.age = 20;
    System.out.println("名前" + taro.name);
    System.out.println("年齢" + taro.age);

    Person hanako = new Person();
    hanako.name = "鈴木花子";
    hanako.age = 18;
    System.out.println("名前" + hanako.name);
    System.out.println("年齢" + hanako.age);
  }
}
```

```
public class Person {
    String name;
    String age;
}
```



インスタンス化

演習：

1．Testクラスのプログラムに佐藤一郎(ichiro)さんのインスタンスを作成しましょう。

2．1.で作成したインスタンスを使って、名前：佐藤一郎、年齢：23と表示できるようプログラミングしてみましょう。（完成したら、実行してみましょう。）

3．Testクラスのプログラムにあなたの名前（半角英小文字）でインスタンスを作成しましょう。

4．3.で作成したインスタンスを使って、あなたの名前と年齢を表示できるようプログラミングしてみましょう。（完成したら、実行してみましょう。）

次のページで答え合わせをしましょう。

インスタンス化

演習(解答)： 答え合わせをしてみましょう。

1. Testクラスのプログラムに佐藤一郎(ichiro)さんのインスタンスを作成しましょう。
   ⇒Person ichiro = new Person();
2. 1.で作成したインスタンスを使って、名前：佐藤一郎、年齢：23と表示できるようプログラミングしてみましょう。(完成したら、実行してみましょう。)
   ⇒ichiro.name="佐藤一郎";
   ichiro.age=23;
3. Testクラスのプログラムにあなたの名前(半角英小文字)でインスタンスを作成しましょう。
   ⇒(例)○○○○には、jiroやsaburoのようにして、あなたの名前をプログラムします。
   Person ○○○○ = new Person();
4. 3.で作成したインスタンスを使って、あなたの名前と年齢を表示できるようプログラミングしてみましょう。(完成したら、実行してみましょう。)
   ⇒(例)○○○○には、木村次郎、△△には、25のようにして、あなたの名前と年齢をプログラムします。
   ichiro.name="○○○○";
   ichiro.age=△△;

以上