Java基礎演習
(オブジェクト指向編)

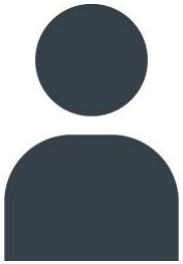
インスタンス化の応用

インスタンス化の応用

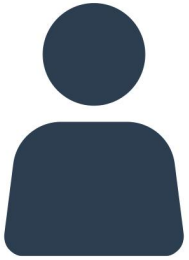
「オブジェクト指向プログラミング」では、様々なやり取りをおこなう仕組みがあります。

例えば、自分と友人のプログラムを書いたとしましょう。

2人は「オブジェクト指向プログラミング」では、「オブジェクト」だと言い換えることができます。



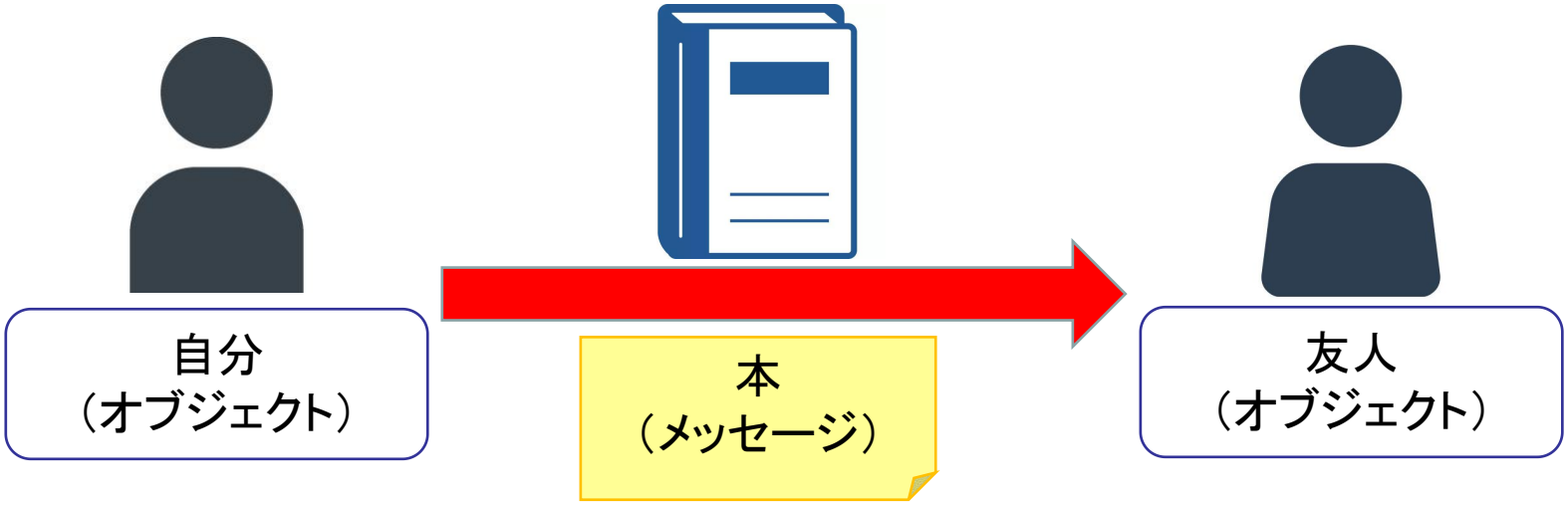
自分
（オブジェクト）



友人
（オブジェクト）

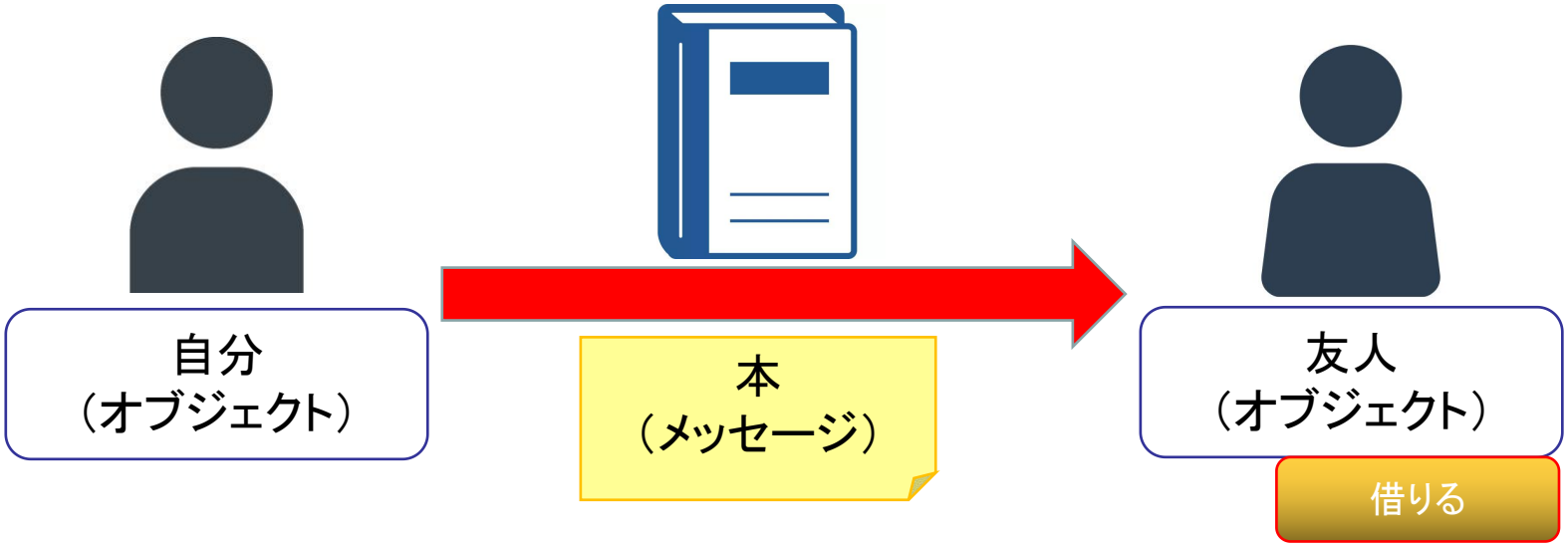
インスタンス化の応用

今回、自分は友人に本を貸したとします。
本は自分から友人に伝える「メッセージ」と言い換えることができます。



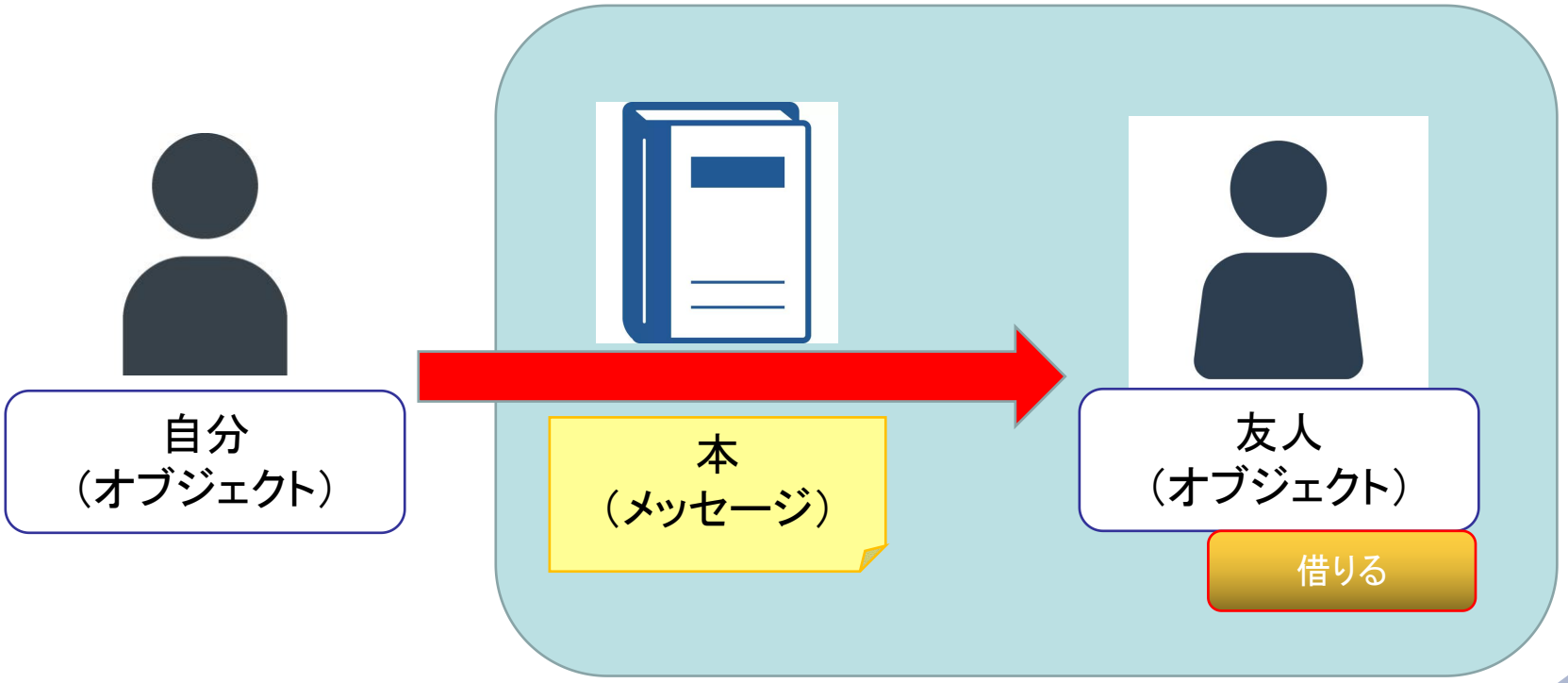
インスタンス化の応用

また、このとき本を実際に借りたのは友人です。
友人には「借りる」という機能を追加しておきます。



インスタンス化の応用

今回は、友人（オブジェクト）が本（メッセージ）を借りる為のプログラムを書いてみます。

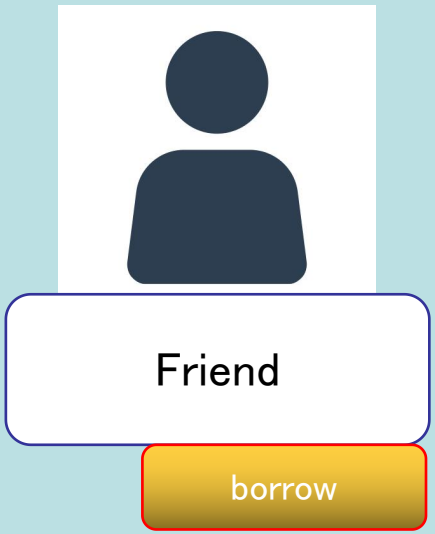


インスタンス化の応用

今回は、友人をFriendクラスとします。
また、借りるという機能をborrowとします。
以下のように機能（今回はborrow）はメソッドとしてプログラムしてゆきます。

```
public class Friend {
    public void borrow(){

    }
}
```



インスタンス化の応用

以下をプログラムしてみましょう。
■ファイル名：Friend.java

```
public class Friend {
    public void borrow() {
        System.out.println("本を借りた");
    }
}
```

インスタンス化の応用

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 Friend.java

C:¥Users¥internous> dir

Friend.java
Friend.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

インスタンス化の応用

つぎに、自分自身をMyselfクラスとします。以下のようにmainメソッドを使って、プログラムしてゆきます。次のページを参考に、友達(Friend)に本を貸す(borrow)プログラムを書いてみましょう。

Myself

```
public class Myself {
    public static void main(String[] args){

    }
}
```

インスタンス化の応用

以下をプログラムしてみましょう。
■ファイル名：Myself.java

```
public class Myself {
    public static void main(String[] args) {
        Friend friend = new Friend();
        System.out.println("---友達とのやりとり---");
        friend.borrow();
    }
}
```

インスタンス化の応用

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 Myself.java

C:¥Users¥internous> dir

Friend.java
Friend.class
Myself.java
Myself.class

C:¥Users¥internous>java Myself

---友達とのやりとり---
本を借りた
```

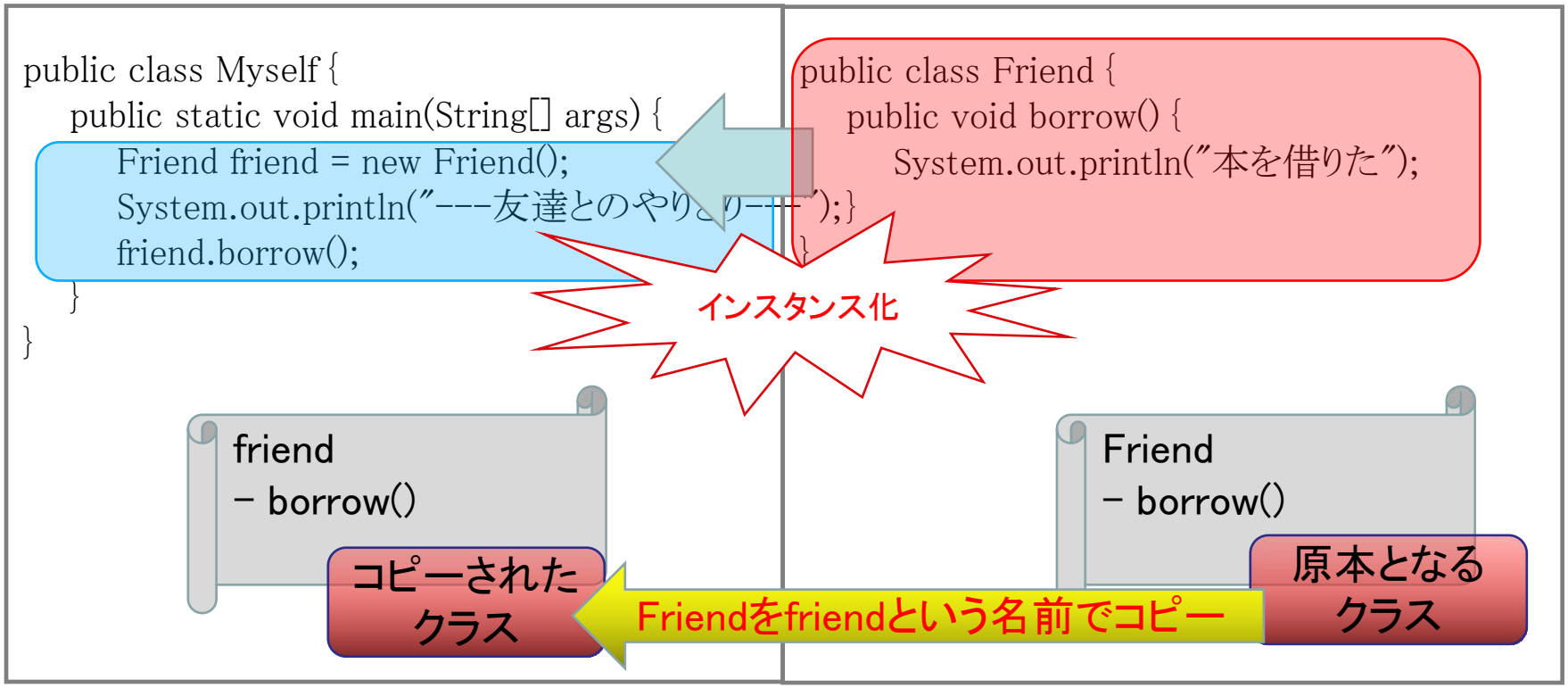
ここでコンパイルします

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

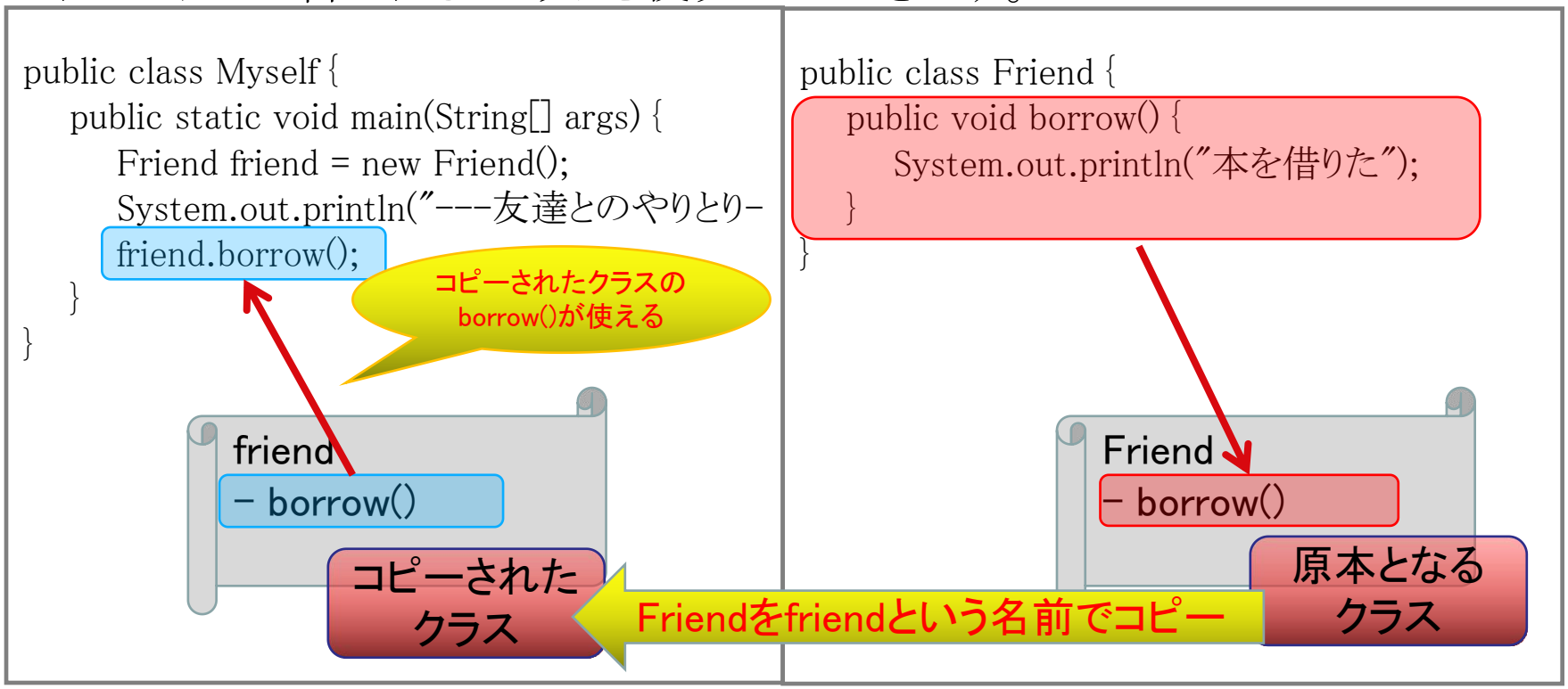
インスタンス化の応用

このようにインスタンス化をおこなうと、友人(friend)は本を借りることができました。



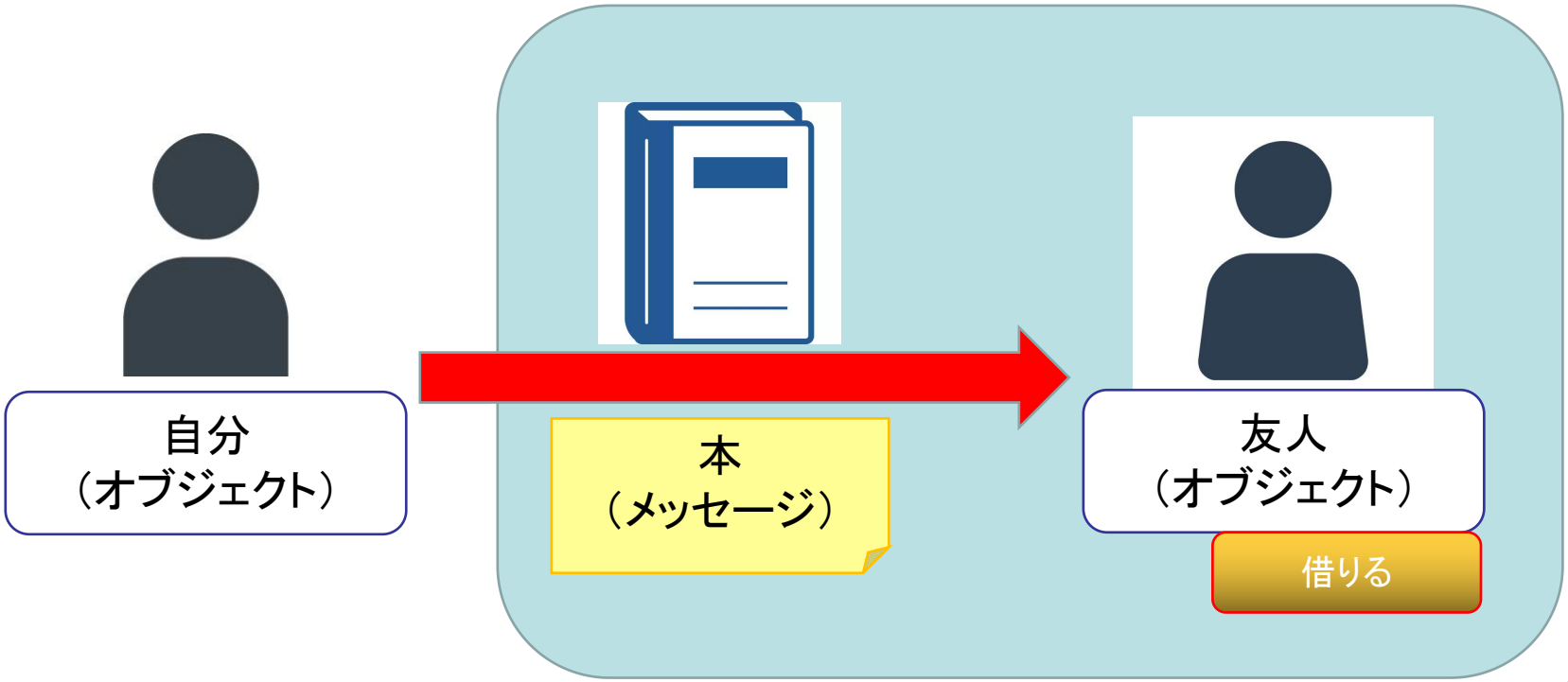
インスタンス化の応用

また、インスタンス(friend)は”.”(ドット)をつけてメソッド名を続けて書くと、コピーされたクラスに含まれるメソッドを使うことができます。



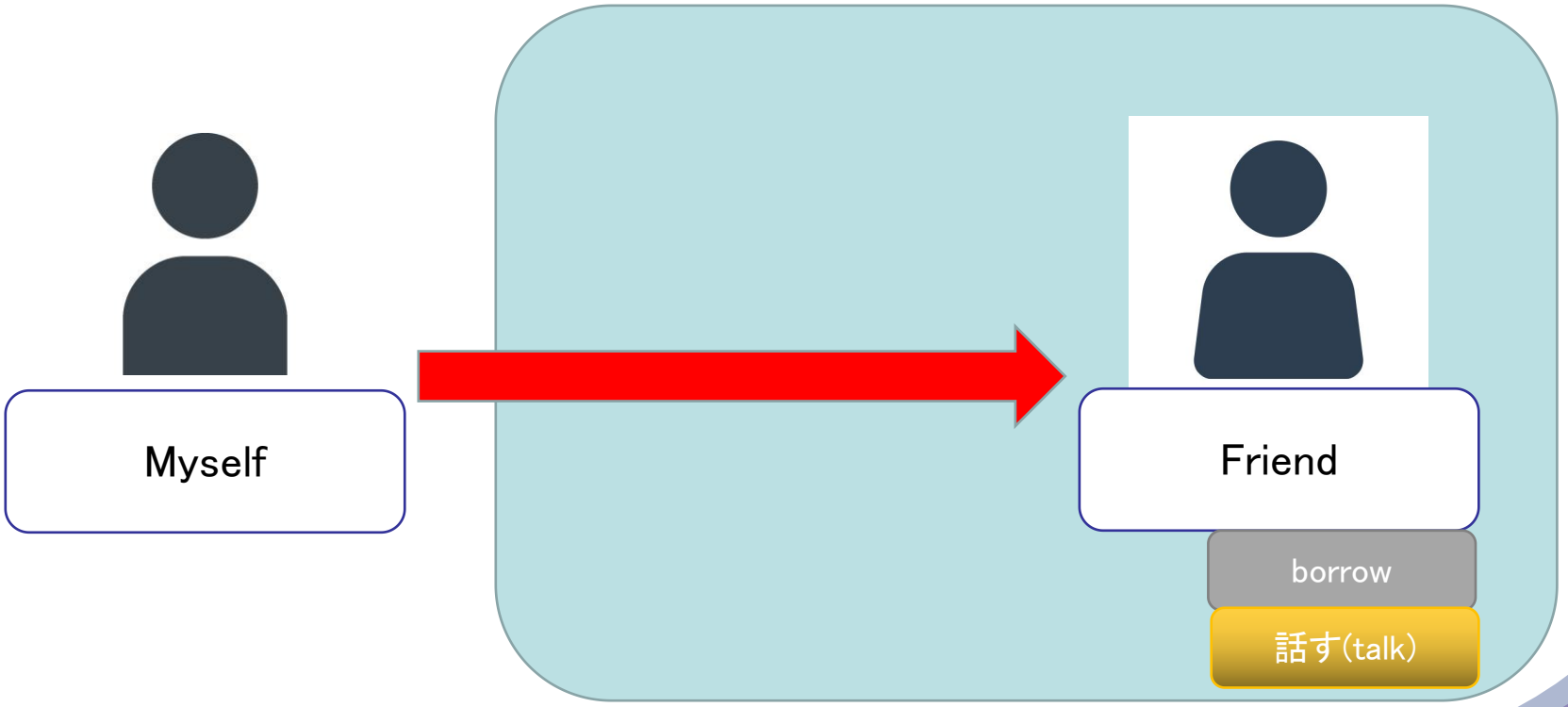
インスタンス化の応用

これで友人は本を借りることができました。
このようにインスタンス化すると、他のオブジェクトといろいろなやり取りができるようになります。



インスタンス化の応用

今度は友人と話すプログラムを追加してみます。
今回は例として、Friendクラスに話す（talk）機能を追加してみます。
次のページを参考にしながらFriendクラスを書き換えてみましょう。



インスタンス化の応用

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：Friend.java

```
public class Friend {
    public void borrow() {
        System.out.println("本を借りた");
    }
    public void talk() {
        System.out.println("話した");
    }
}
```

インスタンス化の応用

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 Friend.java

C:¥Users¥internous> dir

Friend.java
Friend.class

C:¥Users¥internous>
```

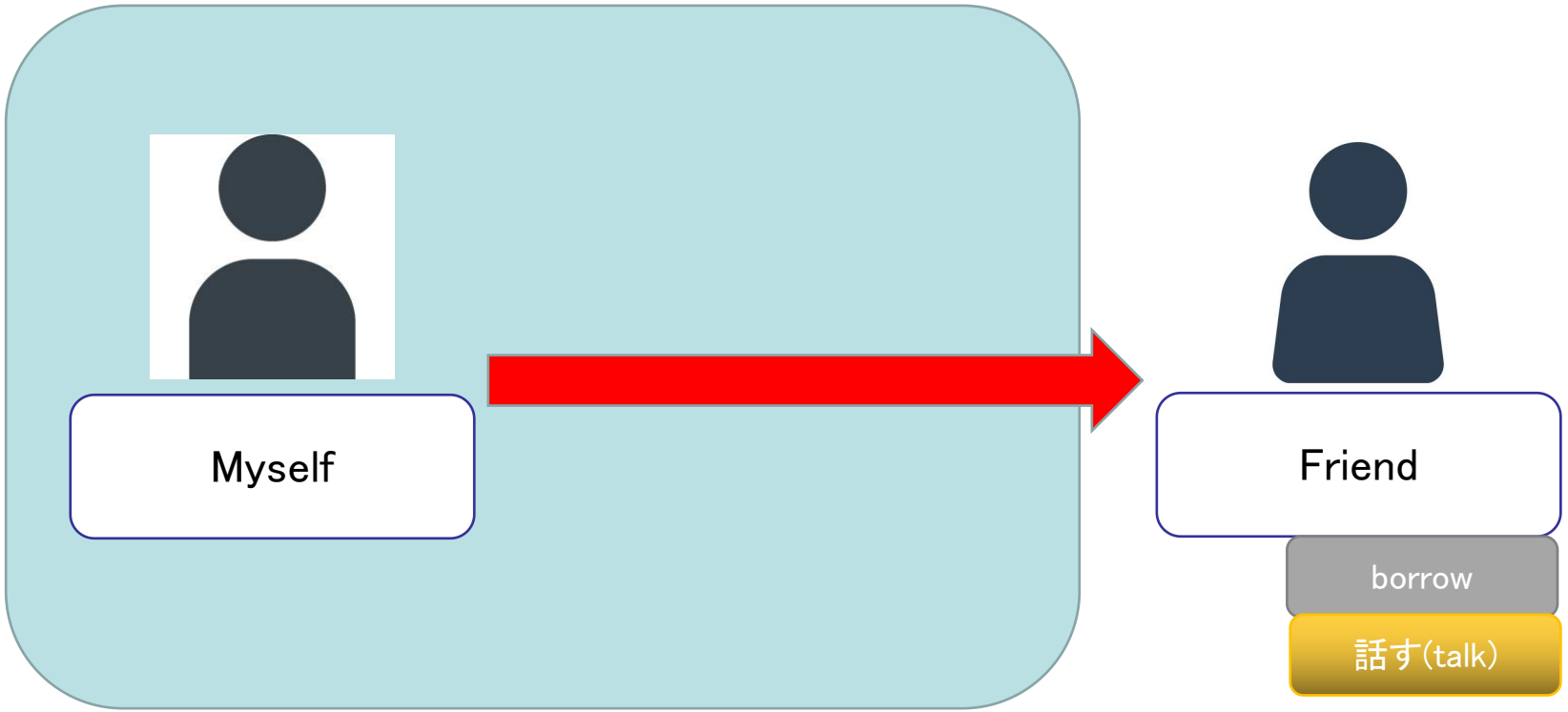
ここでコンパイルします

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

インスタンス化の応用

つぎに、Myselfクラスのmainメソッドを使って、話すプログラムを書いてゆきます。
次のページを参考に、友達(Friend)と話す(talk)プログラムを書いてみましょう。



インスタンス化の応用

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：Myself.java

```
public class Myself {
    public static void main(String[] args) {
        Friend friend = new Friend();
        System.out.println("---友達とのやりとり---");
        friend.borrow();
        friend.talk();
    }
}
```

インスタンス化の応用

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 Myself.java

C:¥Users¥internous> dir

Friend.java
Friend.class
Myself.java
Myself.class

C:¥Users¥internous>java Myself
---友達とのやりとり---
本を借りた
話した
```

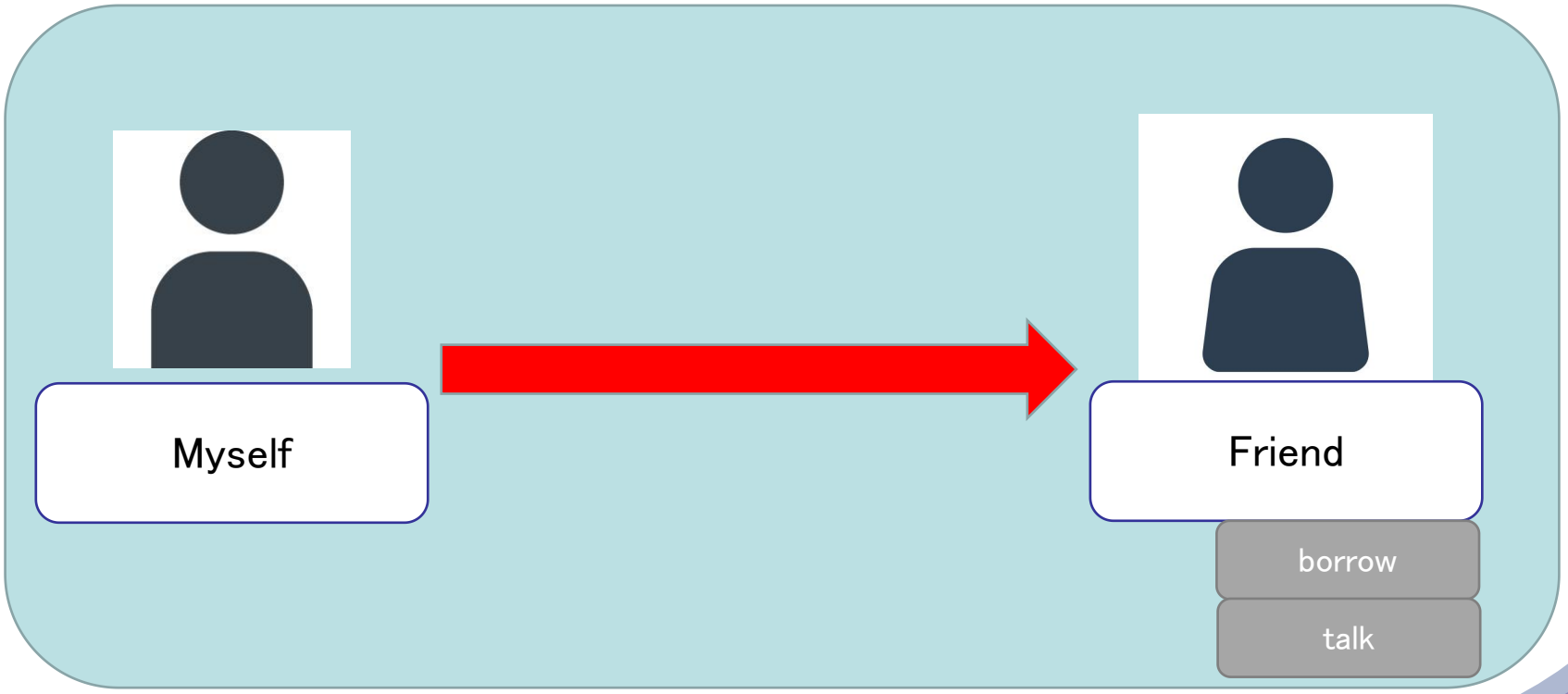
ここでコンパイルします

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

インスタンス化の応用

このように一度インスタンス化すると、シンプルなプログラミングが可能です。



インスタンス化の応用

演習：

1．Friendクラスに遊ぶ（play）という機能を追加してみましょう。また、「遊ぶ」と表示できるようプログラミングしてみましょう。

2．1．を使って、Myselfクラスから「遊ぶ」と表示できるようプログラミングし、実行してみましょう。

3．MyselfクラスでFriendクラスをインスタンス化してみましょう。
また、インスタンスの名前は、「taro」としてプログラミングしてみましょう。

4．Myselfクラスで「---太郎さんとのやりとり---」と表示できるようプログラミングしましょう。
また、3．のインスタンスを使って、「遊ぶ」と表示できるようプログラミングし、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。

インスタンス化の応用

演習（解答）：　答え合わせをしてみましょう。

1．Friendクラスに遊ぶ（play）という機能を追加してみましょう。また、「遊ぶ」と表示できるようプログラミングしてみましょう。

⇒public void play(){
　　System.out.println(″遊ぶ″);
　}

2．1．を使って、Myselfクラスから「遊ぶ」と表示できるようプログラミングしてみましょう。

⇒friend.play();

3．MyselfクラスでFriendクラスをインスタンス化してみましょう。
また、インスタンスの名前は、「taro」としてプログラミングしてみましょう。

⇒Friend taro = new Friend();

4．Myselfクラスで「---太郎さんとのやりとり---」と表示できるようプログラミングしましょう。
また、3．のインスタンスを使って、「遊ぶ」と表示できるようプログラミングしてみましょう。

⇒System.out.println(″---太郎さんとのやりとり---″);
　taro.play();

以上