

方法寫入外部物件

在 Java 中，方法參數的傳遞方式是按值傳遞，也就是說，方法內部參數的修改不會影響到原始變量的值。因此，在 Java 中無法直接通過方法回傳外部變量。

不過，可以通過一些技巧實現類似於按引用傳遞的效果。其中一種方法是將要回傳的變量封裝成一個對象，並將該對象作為方法的參數傳遞過去。這樣，方法內部對該對象的修改會影響到原始變量的值。

例如，假設有一個要回傳的變量 `int x`，可以將其封裝成一個對象 `IntHolder`，該對象包含一個整數屬性 `value`，並提供一個方法 `getValue()` 和 `setValue(int)` 分別用於取和設置屬性的值。然後，在調用方法時，將一個 `IntHolder` 對象作為參數傳遞過去。方法內部就可以通過 `IntHolder` 對象修改原始變量的值。

下面是一個示例代碼：

```
public class IntHolder {
    private int value;

    public IntHolder(int value) {
        this.value = value;
    }

    public int getValue() {
        return value;
    }

    public void setValue(int value) {
        this.value = value;
    }
}

public class Main {
    public static void main(String[] args) {
        IntHolder xHolder = new IntHolder(10);
        System.out.println("Before method call: " + xHolder.getValue());

        // 調用方法，將 xHolder 作為參數傳遞過去
        modifyIntHolder(xHolder);

        System.out.println("After method call: " + xHolder.getValue());
    }

    public static void modifyIntHolder(IntHolder holder) {
        // 修改 holder 的值，會影響到 xHolder
        holder.setValue(20);
    }
}
```

在上面的示例代碼中，`IntHolder` 對象 `xHolder` 的值被初始化為 10，然後被傳遞到 `modifyIntHolder` 方法。在方法內部，`holder` 對象的值被修改為 20。由於 `xHolder` 和 `holder` 引用的是同一個對象，因此 `xHolder` 的值也被修改為 20。最終輸出結果：

```
Before method call: 10
After method call: 20
```