Side Effects

Side Effects

- Side effect of a method: any externally observable data modification

```
public void transfer(double amount, BankAccount other)
{
   balance = balance - amount;
   other.balance = other.balance + amount;
      // Modifies explicit parameter
}
```

- Updating explicit parameter can be surprising to programmers; it is best to avoid it if possible

Side Effects

- Another example of a side effect is output

```
public void printBalance() // Not recommended
{
   System.out.println("The balance is now $" + balance);
}
```

Bad idea: message is in English, and relies on `System.out`

It is best to decouple input/output from the actual work of your classes

- You should minimize side effects that go beyond modification of the implicit parameter

Question?

- If a refers to a bank account, then the call `a.deposit(100)` modifies the bank account object. Is that a side effect?
- Answer: No–a side effect of a method is any change outside the implicit parameter

Another Question?

- Consider the `DataSet` class. Suppose we add a method

```
void read(Scanner in)
{
   while (in.hasNextDouble())
      add(in.nextDouble());
}
```

Does this method have a side effect?

- **Answer:** Yes–the method affects the state of the `Scanner` parameter

Common Error – Trying to Modify Primitive Type Parameter

-
```
void transfer(double amount, double otherBalance)
{
   balance = balance - amount;
   otherBalance = otherBalance + amount;
}
```

- Won't work
- Scenario:

```
double savingsBalance = 1000;
harrysChecking.transfer(500, savingsBalance);
System.out.println(savingsBalance);
```

- In Java, a method can never change parameters of primitive type

Modifying a Numeric Parameter Has No Effect on Caller

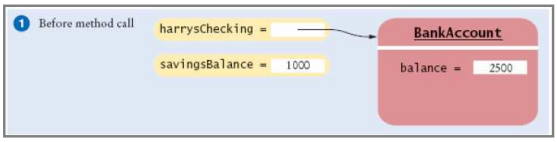

Figure 3(1):
Modifying a Numeric Parameter Has No Effect on Caller

Modifying a Numeric Parameter Has No Effect on Caller

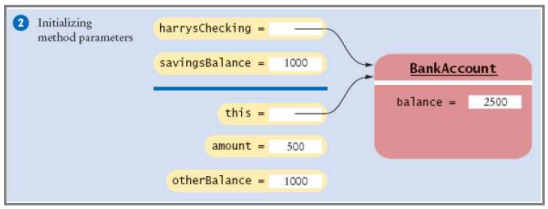

Figure 3(2):
Modifying a Numeric Parameter Has No Effect on Caller

Modifying a Numeric Parameter Has No Effect on Caller

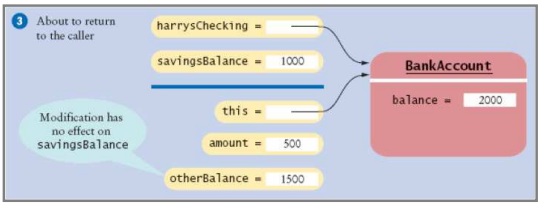

Figure 3(3):
Modifying a Numeric Parameter Has No Effect on Caller

Modifying a Numeric Parameter Has No Effect on Caller

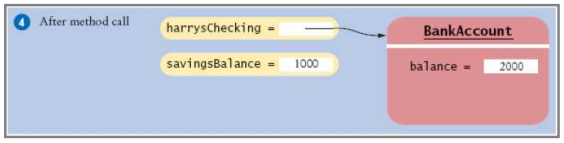

Figure 3(4):
Modifying a Numeric Parameter Has No Effect on Caller

Call By Value and Call By Reference

- **Call by value:** Method parameters are copied into the parameter variables when a method starts
- **Call by reference:** Methods can modify parameters
- Java has call by value

Call By Value and Call By Reference

- A method can change state of object reference parameters, but cannot replace an object reference with another

```
public class BankAccount
{
   public void transfer(double amount, BankAccount otherAccount)
   {
      balance = balance - amount;
      double newBalance = otherAccount.balance + amount;
      otherAccount = new BankAccount(newBalance); // Won't work
   }
}
```

Call By Value Example

```
harrysChecking.transfer(500, savingsAccount);
```

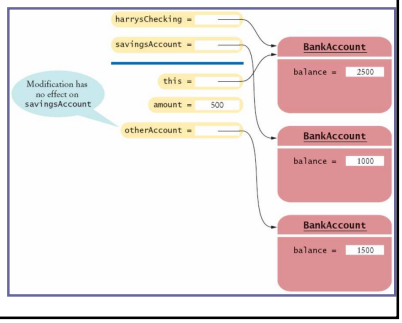

Figure 4:
Modifying an Object Reference Parameter Has No Effect on the Caller