CS 102
Object Oriented Programming

this

Reyyan Yeniterzi
reyyan.yeniterzi@ozyegin.edu.tr

October 3, 2016

this

- An important keyword used inside methods in order to refer to the current object.
- ***this*** is "**the calling object**"
  - *this* is the object which called the method
  - The object that occurs before the dot

this

- There are several use cases of ***this***

Bank Account

- Version 15

```java
public class Account {
    private int number;
    private double balance;
    private String currency;
    private double interestRate;

    public Account(int n, double b, String c) {
        number = n;
        if (b > 0)
            balance = b;
        else
            balance = 0;

        interestRate = 0;
        checkSetCurrency(c);
    }
    public Account(int n, String c) {
        number = n;
        balance = 0;
        interestRate = 0;

        checkSetCurrency(c);
    }
    public Account(int n) {
        number = n;
        balance = 0;
        currency = "TL";
        interestRate = 0;
    }
```

```java
public int getNumber()  {
    return number;
}
public double getBalance()  {
    return balance;
}
public String getCurrency() {
    return currency;
}
public double getInterestRate() {
    return interestRate;
}

public void setInterestRate(double i) {
    interestRate = i;
}
public void setCurrency(String c)   {
    if (currency.equals("TL") && c.equals("USD")) {
        balance = balance / 2.9;
    }
    if (currency.equals("USD") && c.equals("TL")) {
        balance = balance * 2.9;
    }
    if (c.equals("TL") || c.equals("USD")) {
        currency = c;
    }
}
```

this

- There are several use cases of ***this***
  1. It is used to access the instance variables without any confusion with parameter names.

this

- There are several use cases of ***this***
  1. It is used to access the instance variables without any confusion with parameter names.

```
public void setInterestRate(double i) {
    interestRate = i;
}
public void setCurrency(String c)   {
    if (currency.equals("TL") && c.equals("USD")) {
        balance = balance / 2.9;
    }
    if (currency.equals("USD") && c.equals("TL")) {
        balance = balance * 2.9;
    }
    if (c.equals("TL") || c.equals("USD")) {
        currency = c;
    }
}
```

this (first use case)


```
public void setInterestRate(double i) {
    interestRate = i;
}
public void setCurrency(String c)   {
    if (currency.equals("TL") && c.equals("USD")) {
        balance = balance / 2.9;
    }
    if (currency.equals("USD") && c.equals("TL")) {
        balance = balance * 2.9;
    }
    if (c.equals("TL") || c.equals("USD")) {
        currency = c;
    }
}
```

- One letter parameters are not very descriptive.
  - It is usually the convention in Java to use the instance variable name as parameters in the corresponding set methods.

this (first use case)

- It should be smt like:

```
public void setInterestRate(double interestRate) {
    interestRate = interestRate;
}
public void setCurrency(String currency)    {
    if (currency.equals("TL") && currency.equals("USD")) {
        balance = balance / 2.9;
    }
    if (currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        currency = currency;
    }
}
```

  - parameter names are same with class instance names
  - no compile error

this (first use case)


```java
public void setInterestRate(double interestRate) {
    interestRate = interestRate;
}
public void setCurrency(String currency)    {
    if (currency.equals("TL") && currency.equals("USD")) {
        balance = balance / 2.9;
    }
    if (currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        currency = currency;
    }
}
```

- What should be the output?

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

    Account account1 = new Account(1, 100, "TL");

    account1.setCurrency("USD");
    System.out.println(account1.getBalance());
    System.out.println(account1.getCurrency());

    account1.setInterestRate(0.02);
    System.out.println(account1.getInterestRate());
}
```

this (first use case)

Problems | Javadoc | Declaration | Console

<terminated> AccountTest (3) [Java Application] C:\Program Fil

```
100.0
TL
0.0
```

```
public void setInterestRate(double interestRate) {
    interestRate = interestRate;
}
public void setCurrency(String currency)    {
    if (currency.equals("TL") && currency.equals("USD")) {
        balance = balance / 2.9;
    }
    if (currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        currency = currency;
    }
}
```

- What should be the output?

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

    Account account1 = new Account(1, 100, "TL");

    account1.setCurrency("USD");
    System.out.println(account1.getBalance());
    System.out.println(account1.getCurrency());

    account1.setInterestRate(0.02);
    System.out.println(account1.getInterestRate());
}
```

this (first use case)


```
public void setInterestRate(double interestRate) {
    interestRate = interestRate;
}
publ                                    currency)     {
                                        & currency.equals("USD")) {
                                        ;
    if (currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        currency = currency;
    }
}
```

The assignment to variable interestRate has no effect

2 quick fixes available:

- Qualify left hand side
- Qualify right hand side

Press 'F2' for focus

- When class instance and parameter have the same names, the assignment operation has no effect

this (first use case)

- The parameters interestRate and currency **shadow** the class instances.

```
public void setInterestRate(double interestRate) {
    interestRate = interestRate;
}
```

- The interestRate assigns the parameter interestRate to itself, causing no change what so ever.

this (first use case)

- The parameters interestRate and currency **shadow** the class instances.

```
public void setInterestRate(double interestRate) {
    interestRate = interestRate;
}
```

- In such cases you can use the ***this*** keyword to reach the class instances.

```
public void setInterestRate(double interestRate) {
    this.interestRate = interestRate;
}
```

this (first use case)


```
public void setCurrency(String currency)     {
    if (currency.equals("TL") && currency.equals("USD")) {
        balance = balance / 2.9;
    }
    if (currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        currency = currency;
    }
}
```

```
public void setCurrency(String currency)     {
    if (this.currency.equals("TL") && currency.equals("USD")) {
        balance = balance / 2.9;
    }
    if (this.currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        this.currency = currency;
    }
}
```

this (first use case)


```java
public void setInterestRate(double interestRate) {
    this.interestRate = interestRate;
}
public void setCurrency(String currency)    {
    if (this.currency.equals("TL") && currency.equals("USD")) {
        balance = balance / 2.9;
    }
    if (this.currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        this.currency = currency;
    }
}
```

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

    Account account1 = new Account(1, 100, "TL");

    account1.setCurrency("USD");
    System.out.println(account1.getBalance());
    System.out.println(account1.getCurrency());

    account1.setInterestRate(0.02);
    System.out.println(account1.getInterestRate());
}
```

- What should be the output?

this (first use case)


```
Problems | @ Javadoc | Declaration | Console
<terminated> AccountTest (3) [Java Application] C:\Program Fil
34.48275862068966
USD
0.02
```

```java
public void setInterestRate(double interestRate) {
    this.interestRate = interestRate;
}
public void setCurrency(String currency)    {
    if (this.currency.equals("TL") && currency.equals("USD")) {
        balance = balance / 2.9;
    }
    if (this.currency.equals("USD") && currency.equals("TL")) {
        balance = balance * 2.9;
    }
    if (currency.equals("TL") || currency.equals("USD")) {
        this.currency = currency;
    }
}
```

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

    Account account1 = new Account(1, 100, "TL");

    account1.setCurrency("USD");
    System.out.println(account1.getBalance());
    System.out.println(account1.getCurrency());

    account1.setInterestRate(0.02);
    System.out.println(account1.getInterestRate());
}
```

- What should be the output?

this (first use case)

- It can be also used inside constructors to seperate class instances from parameters.

```
// Constructors
public Account(int n, double b, String c) {
    number = n;
    if (b > 0)
        balance = b;
    else
        balance = 0;

    interestRate = 0;
    checkSetCurrency(c);
}
public Account(int n, String c) {
    number = n;
    balance = 0;
    interestRate = 0;

    checkSetCurrency(c);
}
public Account(int n) {
    number = n;
    balance = 0;
    currency = "TL";
    interestRate = 0;
}
```

this (first use case)

- It can be also used inside constructors to seperate class instances from parameters.

```
// Constructors
public Account(int n, double b, String c) {
    number = n;
    if (b > 0)
        balance = b;
    else
        balance = 0;

    interestRate = 0;
    checkSetCurrency(c);
}
public Account(int n, String c) {
    number = n;
    balance = 0;
    interestRate = 0;

    checkSetCurrency(c);
}
public Account(int n) {
    number = n;
    balance = 0;
    currency = "TL";
    interestRate = 0;
}
```

First Constructor


```
public Account(int n, double b, String c) {
    number = n;
    if (b > 0)
        balance = b;
    else
        balance = 0;

    interestRate = 0;
    checkSetCurrency(c);
}
```

```
public Account(int number, double balance, String currency) {
    this.number = number;
    if (balance > 0)
        this.balance = balance;
    else
        this.balance = 0;

    this.interestRate = 0;
    this.checkSetCurrency(currency);
}
```

Second Constructor


```
public Account(int n, String c) {
    number = n;
    balance = 0;
    interestRate = 0;

    checkSetCurrency(c);
}
```

```
public Account(int number, String currency) {
    this.number = number;
    this.balance = 0;
    this.interestRate = 0;

    checkSetCurrency(currency);
}
```

Third Constructor


```
public Account(int n) {
    number = n;
    balance = 0;
    currency = "TL";
    interestRate = 0;
}
```

```
public Account(int number) {
    this.number = number;
    this.balance = 0;
    this.currency = "TL";
    this.interestRate = 0;
}
```

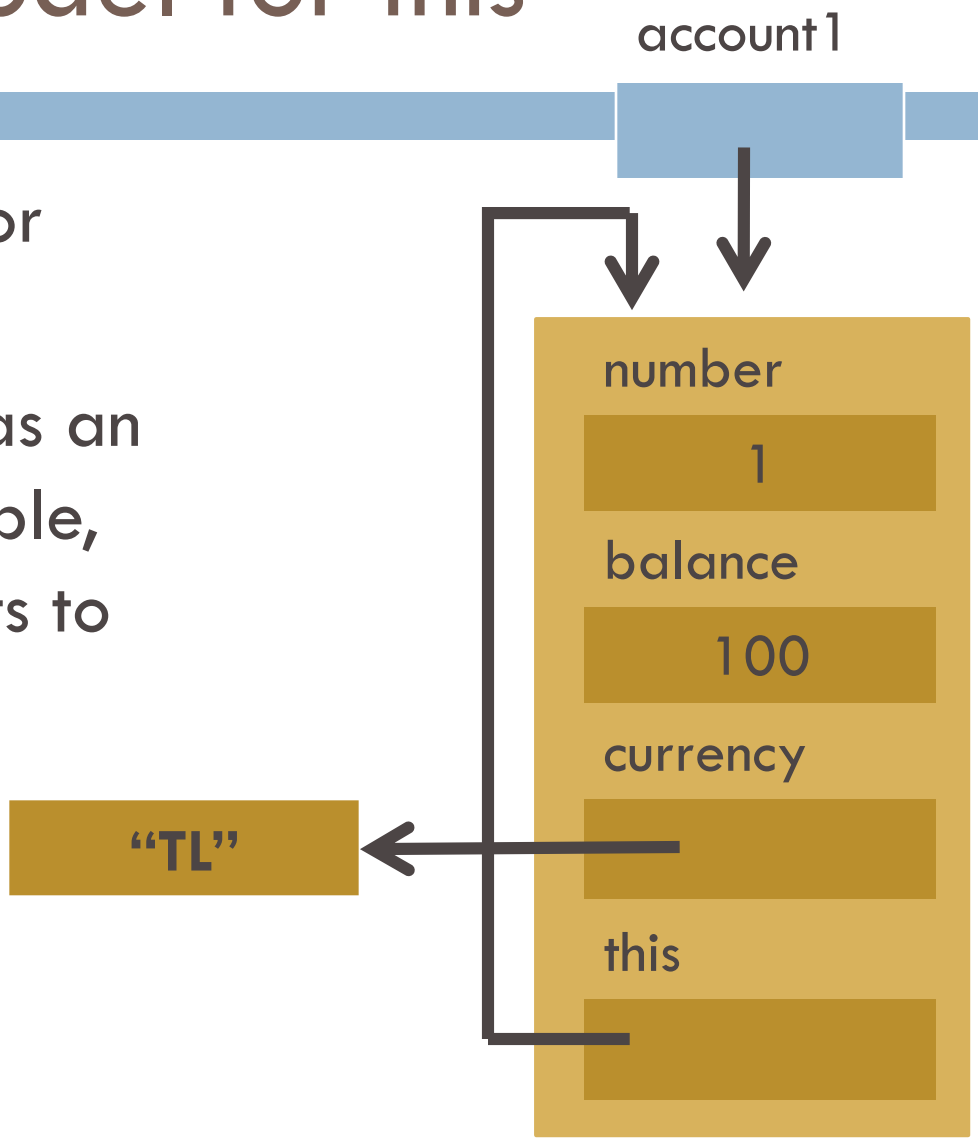
this (first use case)

- To be consistent, the same can be done in **get** methods.

```
public int getNumber()   {
    return this.number;
}
public double getBalance()   {
    return this.balance;
}
public String getCurrency() {
    return this.currency;
}
public double getInterestRate() {
    return this.interestRate;
}
```

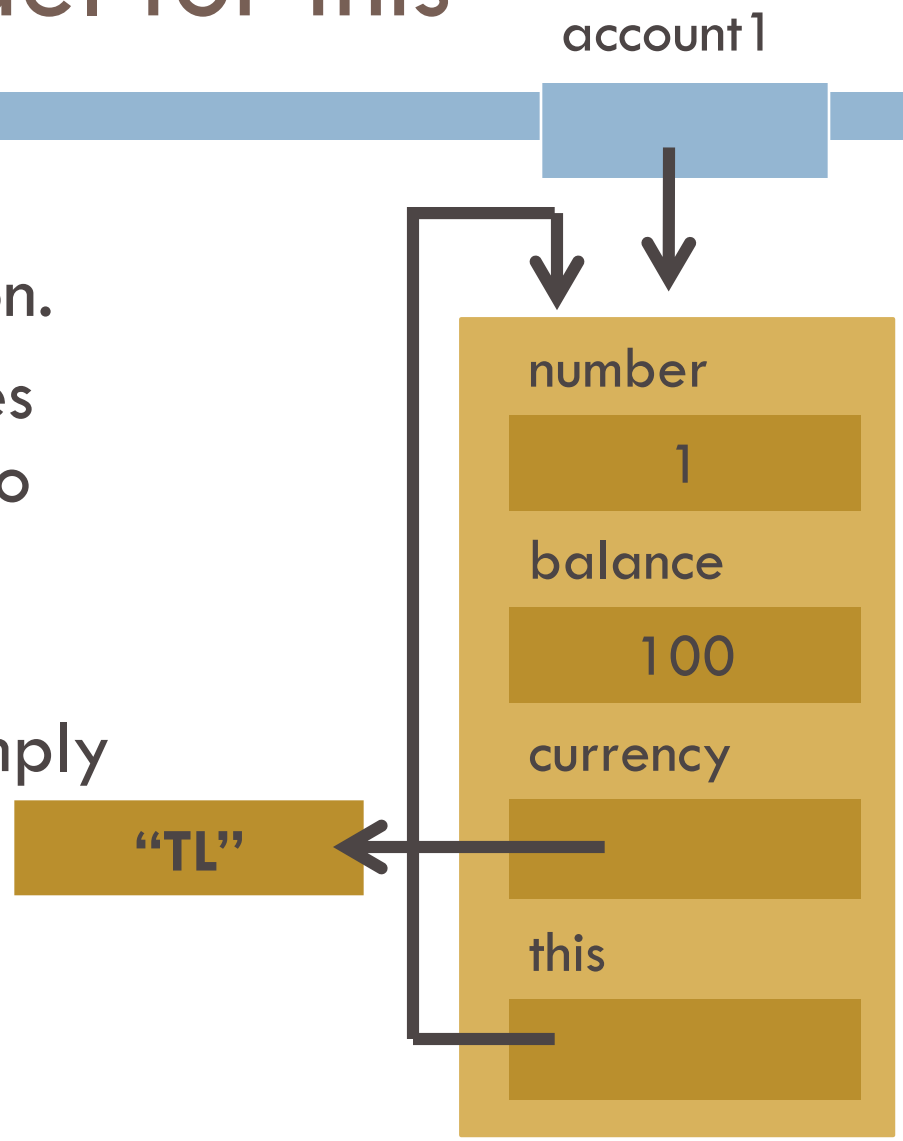
A conceptual model for this

- A conceptual model for understanding ***this***
- Each object in Java has an implicit instance variable, named **this**, that points to the object itself.



A conceptual model for this

- This is not what really happens during execution.
- In Java, each object does NOT keep a reference to itself – that would be a waste of memory space.
- However, take this as simply a conceptual model that helps us understand what ***this*** means.



this

- There are several use cases of ***this***
  1. It is used to access the instance variables without any confusion with parameter names.
  2. It can be used to invoke constructors of the same class.

this (second use case)

- When you have multiple constructors, ***this*** keyword can be used to call other constructors.

Three Constructors


```java
private int number;
private double balance;
private String currency;

// Constructors
public Account(int number, double balance, String currency) {
    this.number = number;
    if (balance > 0)
        this.balance = balance;
    else
        this.balance = 0;

    this.checkSetCurrency(currency);
}
public Account(int number, String currency) {
    this.number = number;
    this.balance = 0;

    checkSetCurrency(currency);
}
public Account(int number) {
    this.number = number;
    this.balance = 0;
    this.currency = "TL";
}
```

this (second use case)


```
private int number;
private double balance;
private String currency;

// Constructors
public Account(int number, double balance, String currency) {
    this.number = number;
    if (balance > 0)
        this.balance = balance;
    else
        this.balance = 0;

    this.checkSetCurrency(currency);
}
public Account(int number, String currency) {
    this(number, 0, currency);
}
public Account(int number) {
    this(number, 0, "TL");
}
```

this (second use case)


```
private int number;
private double balance;
private String currency;

// Constructors
public Account(int number, double balance, String currency) {
    this.number = number;
    if (balance > 0)
        this.balance = balance;
    else
        this.balance = 0;

    this.checkSetCurrency(currency);
}
public Account(int number, String currency) {
    this(number, 0, currency);
}
public Account(int number) {
    this(number, 0, "TL");
}
```

Using this for constructor call

- The constructor call should be the first statement in your constructor.
- Otherwise you will receive the following error!

```
public Account(int number, String currency) {
    int a;
    this(number, 0, currency);
}
```

Constructor call must be the first statement in a constructor

Press 'F2' for focus

```java
public Account(int number, double balance, String currency) {
    this.number = number;
    if (balance > 0)
        this.balance = balance;
    else
        this.balance = 0;

    this.checkSetCurrency(currency);
}
public Account(int number, String currency) {
    this(number, 0, currency);
}
public Account(int number) {
    Account(number, 0.0, "TL");
}
```

The method Account(int, double, String) is undefined for the type Account

1 quick fix available:

Create method 'Account(int, double, String)'

Press 'F2' for focus

- You cannot use the constructor of the class inside a constructor to call another constructor.
- The only way of calling another constructor within a constructor is through using ***this*** keyword.

Any Questions ?