Design Patterns

Chain of Responsibility Pattern*

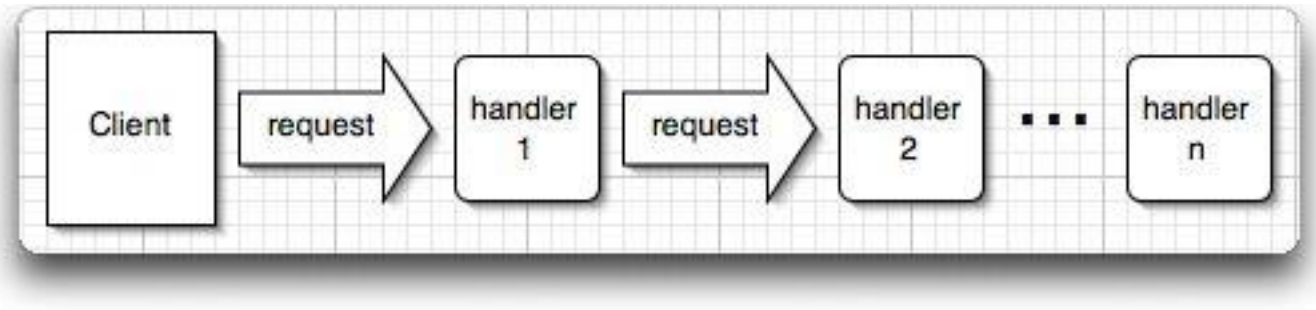
ebru@hacettepe.edu.tr
ebruakcapinarsezer@gmail.com
http://yunus.hacettepe.edu.tr/~ebru/
@ebru176

Kasım 2017

*modified from http://*www.cse.wustl.edu*

Chain of Responsibility Pattern

- Intent
  - Avoid coupling sender of request to its receiver by giving more than one object chance to handle request. Chain receiving objects and pass request along until an object handles it.



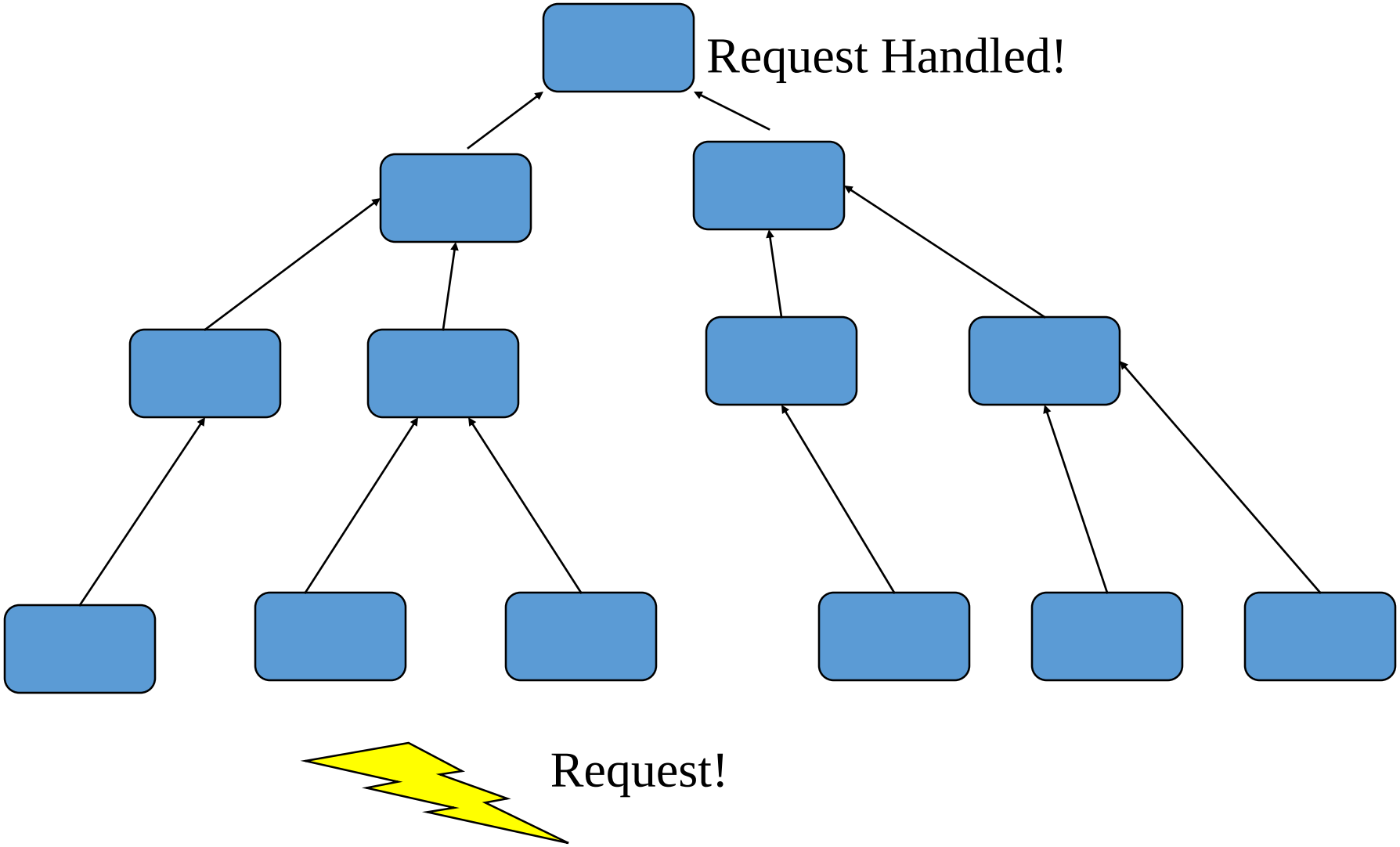
CoR is a Behavioral Pattern

Chain of Responsibility Pattern

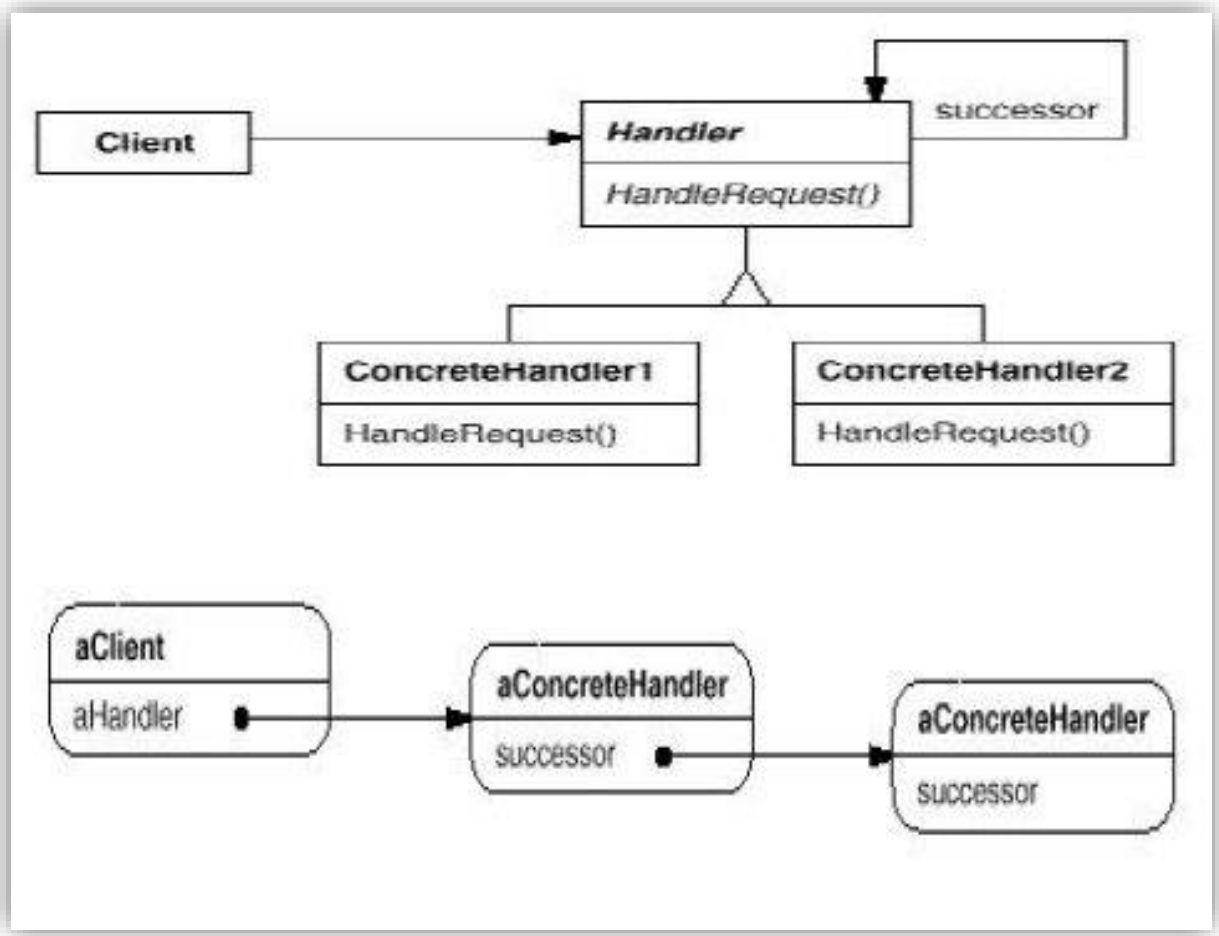
- Description
  - Decouple senders and receivers by giving multiple objects (in a set order) a chance to handle a request. Request passed until an object handles it.

  - 1st object in chain receives request – either handles it or forwards to next object in chain.
  - Object that makes request has no explicit knowledge of who will handle it – request has an *implicit receiver*.

Chain of Responsibility Pattern



CoR Structures



Chain of Responsibility Pattern

Participants:

- Handler – defines interface for handling requests. Can also implement successor link
- ConcreteHandler – handles requests it is responsible for; otherwise forwards requests to successor.
- Client – initiates request to a ConcreteHandler in the chain.

Chain of Responsibility Pattern

- Use Chain of Responsibility when:
  - More than 1 object may handle a request, and handle isn’t known beforehand.
  - Want to issue request to one of several objects without specifying receiver explicitly.
  - Set of objects that can handle a request should be specificed dynamically

Chain of Responsibility Pattern

CONSEQUENCES

- Benefits:
  - Decoupling of senders and receivers
  - Added flexibility
  - Sender doesn’t need to know specifically who the handlers are
- Potential Drawbacks:
  - Client can’t explicity specify who handles a request
  - No guarantee of request being handled (request falls off end of chain)

Sample:

Let’s think a coffee machine that we can buy coffee by throwing money into it. In order to get the coffee, we need to make a cash deposit of 1 tl worth. We need to create a chain so that we can recognize the metal moneys that are inserted into it. All metal money will circulate through this chain until finding the right money object.

```
package MetalParaAtma;

public abstract class MetalPara {
// Money can be dropped wtih the values of 5, 10, 50, 100 Kurus

	protected MetalPara para;

	public void setPara(MetalPara para) {
		this.para = para;
	}

	abstract public void checkPara(double atilanPara);

}
```

```java
package MetalParaAtma;


public class BirLira extends MetalPara {
private final double kabulDeger = 100;


public void checkPara(double atilanPara)
{
   if (atilanPara == kabulDeger)
    {
            System.out.println("Atilan para
            kabul");
   } else {
   if (para != null)
            para.checkPara(atilanPara);
   }
}
}
```

```java
package MetalParaAtma;

public class ElliKurus extends MetalPara {
private final double kabulDeger = 50;

public void checkPara(double atilanPara)
{
            if (atilanPara == kabulDeger)
             {
            System.out.println("Atilan para
kabul");
            }  else  {
            if (para != null)
            para.checkPara(atilanPara);
            }
}
}
```

```java
package MetalParaAtma;

public class OnKurus extends MetalPara {
private final double kabulDeger = 10;

public void checkPara(double atilanPara)
 {
   if (atilanPara == kabulDeger) {
   System.out.println("Atilan para kabul");
   } else {
   if (para != null)
   para.checkPara(atilanPara);
   }
}
}
```

```java
package MetalParaAtma;

public class BesKurus extends MetalPara {
private final double kabulDeger = 5;

public void checkPara(double atilanPara) {
   if (atilanPara == kabulDeger) {
   System.out.println("Atilan para kabul");
   } else {
   if (para != null)
   para.checkPara(atilanPara);
   } else {
   System.out.println("Para Uygun Değil!");
   }

}
}
```

```java
package MetalParaAtma;
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class KahveOtomati {
   public static void main(String[] args) {
            BirLira birLira = new BirLira();
            ElliKurus elliKurus = new ElliKurus();
            OnKurus onKurus = new OnKurus();
            BesKurus besKurus = new BesKurus();
            birLira.setPara(elliKurus);
            elliKurus.setPara(onKurus);
            onKurus.setPara(besKurus);
            while (true) {
                        System.out.println("Otomata para atınız!");
                        double d = 0;
                        try {
            d = Double.parseDouble(new BufferedReader(new InputStreamReader(System.in)).readLine());
                        } catch (Exception e) {
                                    e.printStackTrace();
                        }
                        birLira.checkPara(d);
            }
   }
}
```

Otomata para atınız!

100

Atilan para kabul

Otomata para atınız!

50

Atilan para kabul

Otomata para atınız!

10

Atilan para kabul

Otomata para atınız!

5

Atilan para kabul

Otomata para atınız!

500

Para Uygun Değil!

Otomata para atınız!

Sample

If you want to spend money from your company, you should ask your boss for permission or your patron's boss...

Administrator -> Manager -> Vice President ---> President

SatinAlmaGucu.java

```java
package com.comu.chainofresponsibility;

public abstract class SatinAlmaGucu {
    protected final double taban =500;
    protected SatinAlmaGucu patron;

    public void setPatron(SatinAlmaGucu patron){
        this.patron= patron;
    }
    abstract public void harcamaIstegi(HarcamaIstegi istek);
}
```

HarcamaIstegi.java

```java
package com.comu.chainofresponsibility;

public class HarcamaIstegi {
    private int sayi;
    private double tutar;
    private String niyet;
    public HarcamaIstegi(int sayi, double miktar, String niyet) {
        this.sayi = sayi;
        this.tutar = miktar;
        this.niyet = niyet;
    }
    public int getSayi() {
        return sayi;
    }
    public void setSayi(int sayi) {
        this.sayi = sayi;
    }
    public double getTutar() {
        return tutar;
    }
    public void setTutar(double tutar) {
        this.tutar = tutar;
    }
    public String getNiyet() {
        return niyet;
    }
    public void setNiyet(String niyet) {
        this.niyet = niyet;
    }

}
```

Yonetici.java

```java
package com.comu.chainofresponsibility;

public class Yonetici extends SatinAlmaGucu{
    private final double IZINVERILEN=10*taban;

    @Override
    public void harcamaIstegi(HarcamaIstegi istek) {
        if (istek.getTutar()<IZINVERILEN){
            System.out.println("Yönetici izin verecek:"+istek.getTutar());
        }else{
            if (patron!=null){
                patron.harcamaIstegi(istek);
            }
        }
    }
}
```

Mudur.java

```java
package com.comu.chainofresponsibility;

public class Mudur extends SatinAlmaGucu{
    private final double IZINVERILEN=20*taban;

    @Override
    public void harcamaIstegi(HarcamaIstegi istek) {
        if (istek.getTutar()<IZINVERILEN){
            System.out.println("Yönetici izin verecek:"+istek.getTutar());
        }else{
            if (patron!=null){
                patron.harcamaIstegi(istek);
            }
        }

    }
}
```

BaskanYardimcisi.java

```java
package com.comu.chainofresponsibility;

public class BaskanYardimcisi extends SatinAlmaGucu{
    private final double IZINVERILEN=40*taban;

    @Override
    public void harcamaIstegi(HarcamaIstegi istek) {
        if (istek.getTutar()<IZINVERILEN){
            System.out.println("Yönetici izin verecek:"+istek.getTutar());
        }else{
            if (patron!=null){
                patron.harcamaIstegi(istek);
            }
        }
    }
}
```

*Baskan.java

```java
package com.comu.chainofresponsibility;

public class Baskan extends SatinAlmaGucu{
    private final double IZINVERILEN=60*taban;

    @Override
    public void harcamaIstegi(HarcamaIstegi istek) {
        if (istek.getTutar()<IZINVERILEN){
            System.out.println("Yönetici izin verecek:"+istek.getTutar());
        }else{
            if (patron!=null){
                patron.harcamaIstegi(istek);
            }else{
                System.out.println("Parayı harcayamazsın");
            }
        }
    }
}
```

Uygulama.java

```java
package com.comu.chainofresponsibility;

import java.io.BufferedReader;

public class Uygulama {
    public static void main(String[] args) {
        Yonetici yonetici = new Yonetici();
        Mudur mudur = new Mudur();
        BaskanYardimcisi baskany = new BaskanYardimcisi();
        Baskan baskan = new Baskan();
        yonetici.setPatron(mudur);
        mudur.setPatron(baskany);
        baskany.setPatron(baskan);

        while (true) {
            System.out.println("Harcama yapacagınız miktarı giriniz");
            System.out.print(">");
            double d=0;
            try {
                d = Double.parseDouble(new BufferedReader(new InputStreamReader(System.in)).readLine());
            } catch (NumberFormatException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }
            yonetici.harcamaIstegi(new HarcamaIstegi(0, d, "Genel"));
        }
    }
```

Javadoc | Declaration | Console | Properties

Uygulama (3) [Java Application] C:\Program Files\Java\jdk1.6.0\bin\javaw.exe (04.El

```
Harcama yapacagınız miktarı giriniz
>2000
Yönetici izin verecek:2000.0
Harcama yapacagınız miktarı giriniz
>5500
Müdür izin verecek:5500.0
Harcama yapacagınız miktarı giriniz
>10500
Baskan Yardimcisi izin verecek:10500.0
Harcama yapacagınız miktarı giriniz
>20100
Baskan izin verecek:20100.0
Harcama yapacagınız miktarı giriniz
>
```