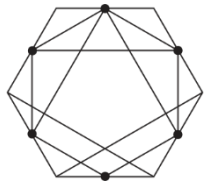
© Coding Akademie München GmbH

REFACTORING PATTERNS BASIC

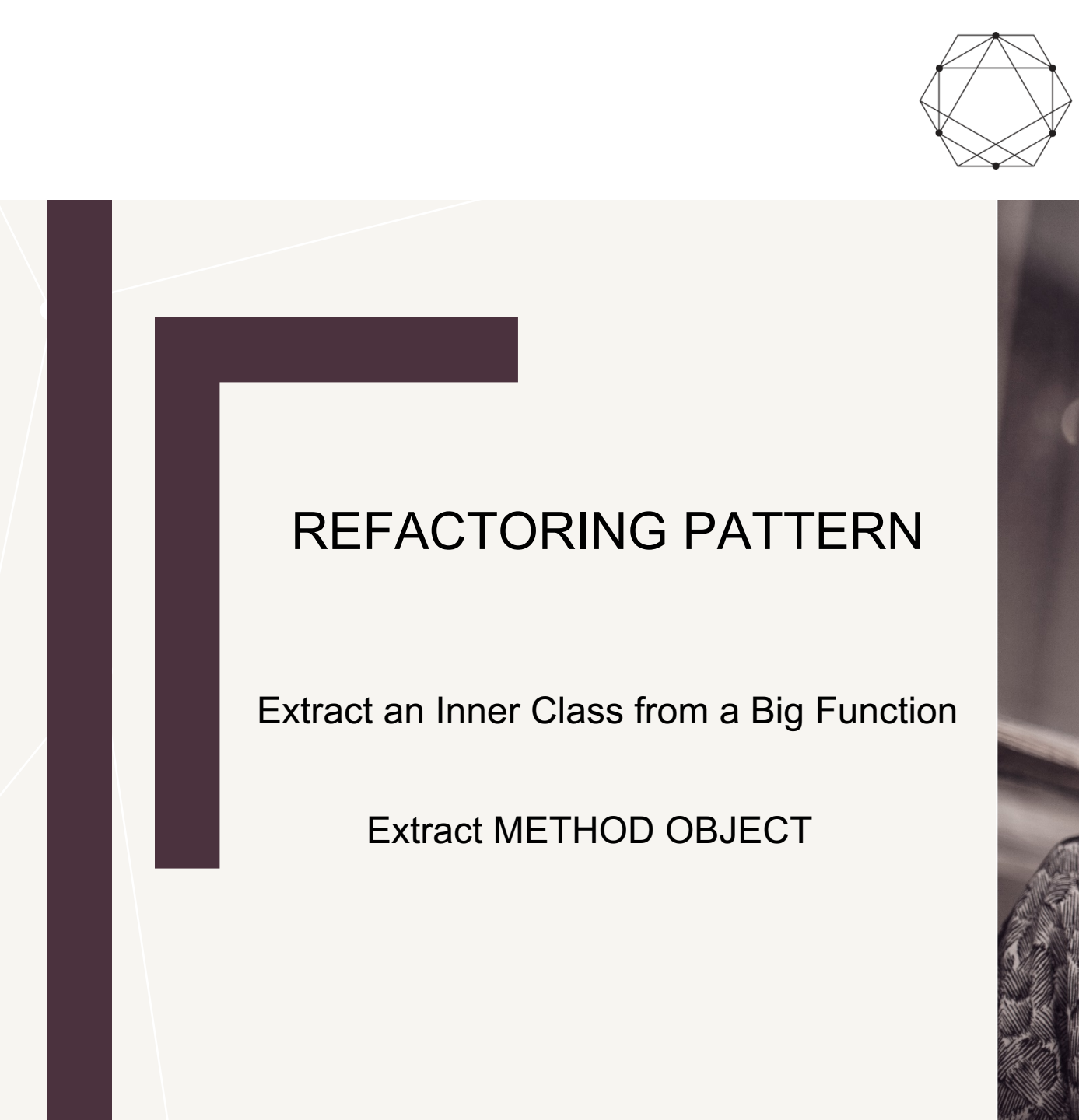
© Coding Akademie München GmbH



GEGEBEN IST FOLGENDER CODE

```
public class Demo {

      public String do(Data data, String name){

            int age;
            Address addres ;

            // Over 30 Lines
            …
}
```

© Coding Akademie München GmbH



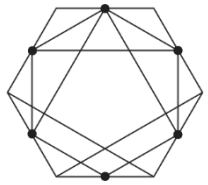
REFACTORING PATTERN

Extract an Inner Class from a Big Function

Extract METHOD OBJECT

EXTRACT METHOD OBJECT

- Create a private static Inner Class
- Parameters => Instance Variables
- Create a Constructor
- Create an invoke() Methode
- Local Variables => Instance Variables
- Call new InnerClass.invoke() Method



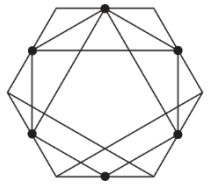
EXTRACT METHOD OBJECT

CREATE AN INNER CLASS

```
public class Demo {

String do(Data data, String name){
            int age;
            Address address ;
            // Over 30 Lines
            …
}
```

```
public class Demo {

private static class Do {


  }

}
```

© Coding Akademie München GmbH



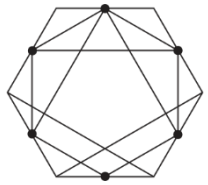
EXTRACT METHOD OBJECT

PARAMETER TO INSTANCE VAR

```
public class Demo {

String do(Data data, String name){
            int age;
            Address address ;
            // Over 30 Lines
            …
}
```

```
public class Demo {

private static class Do {
  private Data data;
  private String name

  }
}
```

© Coding Akademie München GmbH



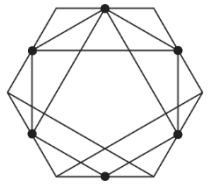
EXTRACT METHOD OBJECT

```
String do(Data data, String name){
            int age;
            Address address ;
            // Over 30 Lines
            …
}
```

CREATE A CONSTRUCTOR

```
private static class Do {
  private Data data;
  private String name

public Do(Data data, String name){
      this.data = data;
      this.name = name;
  }
}
```

© Coding Akademie München GmbH



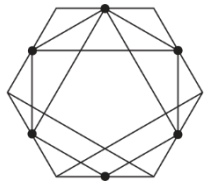
EXTRACT METHOD OBJECT

CREATE INVOKE METHOD

```
String do(Data data, String name){
            int age;
            Address address ;
            // Over 30 Lines
            …
}
```

```
private static class Do {
…

public String invoke(){
       int age;
       Address address ;
      // Over 30 Lines
         … }
}
```

© Coding Akademie München GmbH



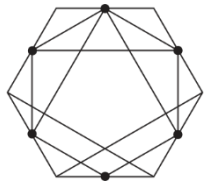
EXTRACT METHOD OBJECT

LOCAL VAR TO INSTANCE VAR

```
String do(Data data, String name){
            int age;
            Address address ;
            // Over 30 Lines
            …
}
```

```
private static class Do {
        …


public String invoke(){
        int age;
        Address address ;
       // Over 30 Lines
          … }
```

© Coding Akademie München GmbH



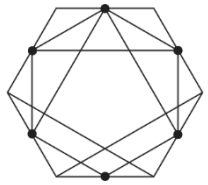
EXTRACT METHOD OBJECT

CONSTRUCTOR

```
String do(Data data, String name){
            int age;
            Address address ;
            // Over 30 Lines
            …
}
```

```
private static class Do {
…
public Do(Data data, String name){
      this.data = data;
      this.name = name;
      age = intValue;
      address = stringValue;
  }
}
```

© Coding Akademie München GmbH



EXTRACT METHOD OBJECT

```
public class Demo {

String do(Data data, String name){
            int age;
            Address address ;
            // Over 30 Lines
            …
}
```

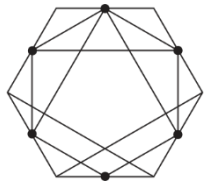
CALL INVOKE IN DEMO

```
public class Demo {

 new Do(data, “name“).invoke();

private static class Do {
 public Do(Data data, String name)
 public String invoke(){ .. }
}
```

© Coding Akademie München GmbH

REFACTORING PATTERN

REMOVE DUPLICATE



GEGEBEN IST FOLGENDER CODE

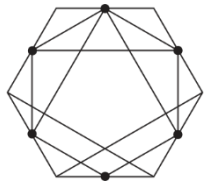
```
WikiPagePath pagePath = wikiPage.getPageCrawler().getFullPath(suiteSetup);
String pagePathName = PathParser.render(pagePath);
buffer.append("!include -setup . ").append(pagePathName).append("\n");
```

```
WikiPagePath tearDownPath = wikiPage.getPageCrawler().getFullPath(teardown);
String tearDownPathName = PathParser.render(tearDownPath);
buffer.append("!include -teardown . ").append(tearDownPathName).append("\n");
```

© Coding Akademie München GmbH

REMOVE DUPLICATE

- Extract Instance Variable from Method Call
- Parameterize Constants
- Extract a Method from Duplicate Code



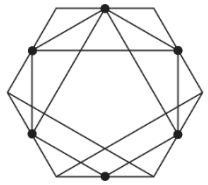
EXTRACT INSTANCE VARIABLES

```
private static class Do {
 ..
 private PageCrawler crawler;


}
```

- **Extract Instance Variable from a Method Call**

```
WikiPagePath tearDownPath = wikiPage.getPageCrawler().getFullPath(teardown);
String tearDownPathName = PathParser.render(tearDownPath);
buffer.append("!include -teardown . ").append(tearDownPathName).append("\n");
```

© Coding Akademie München GmbH



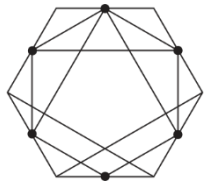
EXTRACT INSTANCE VARIABLES

```
private static class Do {
 ..
 private PageCrawler crawler;
 public Do(Data data, String name){
   crawler = wikiPage.getPageCrawler();
}
```

- **Initialize it in the Constructor**

```
WikiPagePath tearDownPath = wikiPage.getPageCrawler().getFullPath(teardown);
String tearDownPathName = PathParser.render(tearDownPath);
buffer.append("!include -teardown . ").append(tearDownPathName).append("\n");
```

© Coding Akademie München GmbH



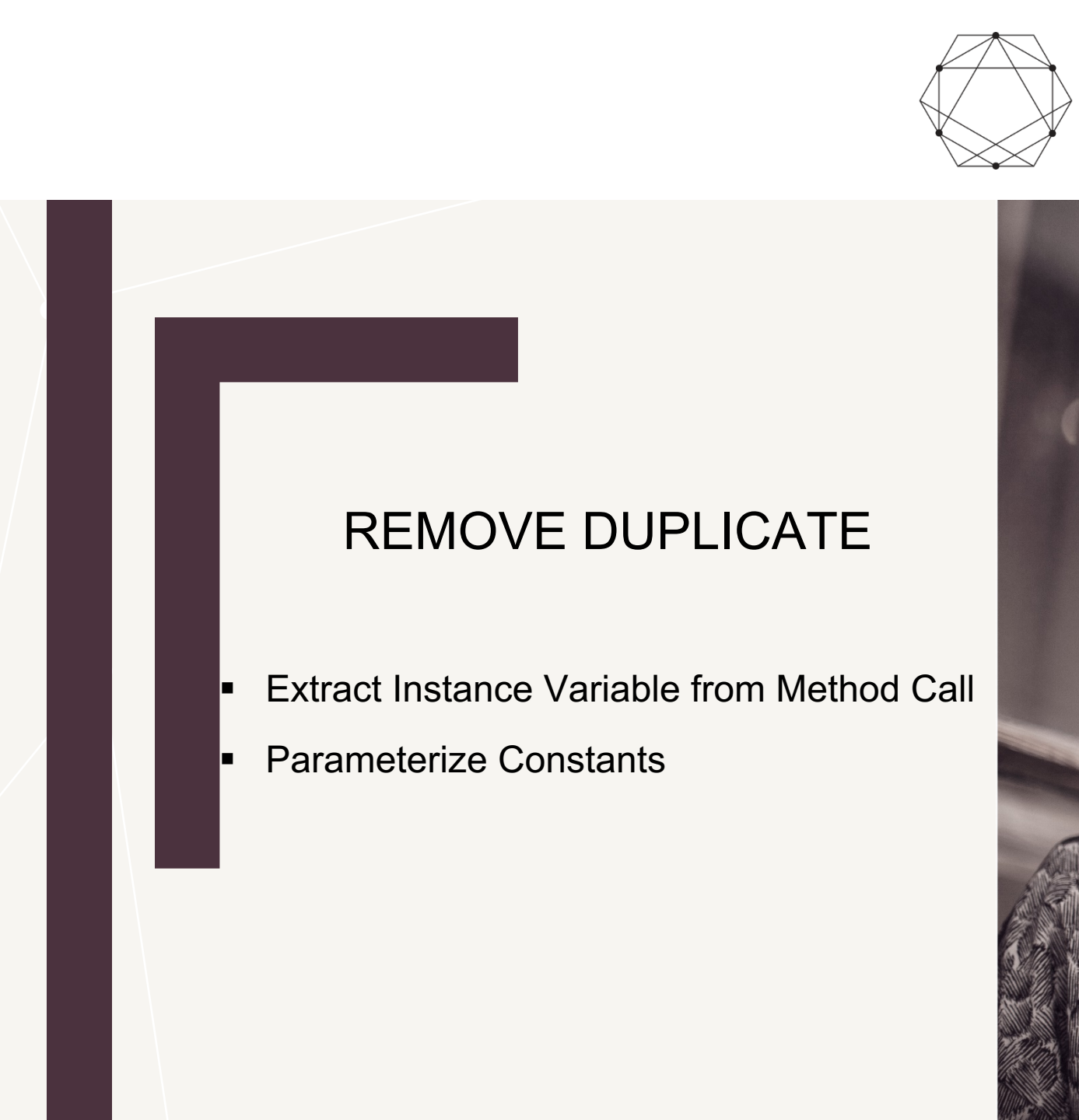
EXTRACT INSTANCE VARIABLES

```
private static class Do {
 ..
 private PageCrawler crawler;
 public Do(Data data, String name){
   crawler = wikiPage.getPageCrawler();
}
```

- **Replace the Method Call with the Instance Variable**

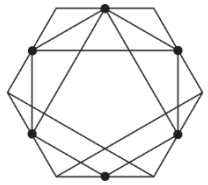
```
WikiPagePath tearDownPath = wikiPage.getPageCrawler().getFullPath(teardown);
String tearDownPathName = PathParser.render(tearDownPath);
buffer.append("!include -teardown . ").append(tearDownPathName).append("\n");
```

© Coding Akademie München GmbH



REMOVE DUPLICATE

- Extract Instance Variable from Method Call
- Parameterize Constants

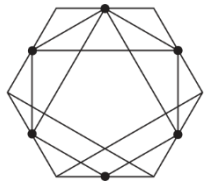


GEGEBEN IST FOLGENDER CODE

```
WikiPagePath pagePath = crawler.getFullPath(suiteSetup);
String pagePathName = PathParser.render(pagePath);
buffer.append("!include -setup . ").append(pagePathName).append("\n");
```

```
WikiPagePath tearDownPath = crawler.getFullPath(teardown);
String tearDownPathName = PathParser.render(tearDownPath);
buffer.append("!include -teardown . ").append(tearDownPathName).append("\n");
```

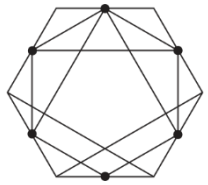
© Coding Akademie München GmbH



GEGEBEN IST FOLGENDER CODE

```
buffer.append("!include -setup . ").append(pagePathName).append("\n");

buffer.append("!include -teardown . ").append(tearDownPathName).append("\n");
```

© Coding Akademie München GmbH



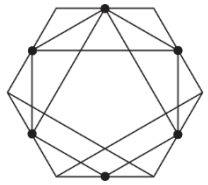
PARAMETERIZE CONSTANTS

EXTRAC A CONTAT TO A LOCAL VARIABLE

```
String  mode = "setup" ;

buffer.append("!include -setup . ").append(pagePathName).append("\n");
```

```
String  mode = "teardown" ;

buffer.append("!include -teardown . ").append(tearDownPathName).append("\n");
```

© Coding Akademie München GmbH



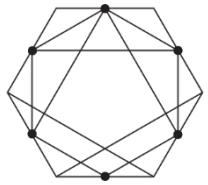
PARAMETERIZE CONSTANTS

REPLACE A CONTAT WITH THE LOCAL VARIABLE

```
String mode = "setup" ;

buffer.append("!include -setup ").append(mode).append(pageFileName).append("\n");
```

```
String mode = "teardown" ;

buffer.append("!include -teardown ").append(mode).append(pageFileName).append("\n");
```

© Coding Akademie München GmbH



VORBEREITUNG ABGESCHLOSSEN

```
WikiPagePath pagePath = crawler.getFullPath(suiteSetup);
String pagePathName = PathParser.render(pagePath);
String  mode = "setup" ;
buffer.append("!include -" + mode + ". ").append(pagePathName).append("\n");
```

```
WikiPagePath tearDownPath = crawler.getFullPath(teardown);
String tearDownPathName = PathParser.render(tearDownPath);
String  mode = "teardown" ;
buffer.append("!include -" + modes " + . ").append(tearDownPathName).append("\n");
```

© Coding Akademie München GmbH

REMOVE DUPLICATE

- Extract Instance Variable from Method Call
- Parameterize Constants
- Extract a Method from Duplicate Code

MORE REFACTORINGS

- Rename an Identifier
- Move a Method to another Class
- Extract a Class from an Inner Class

REFACTORING

LIVE DEMO

© Coding Akademie München GmbH