**PRAKTIKUM-7 PDPL**

***BEHAVIORAL PATTERN – CHAIN OF RESPONSIBILITY***

**Mata Kuliah : Pola Desain Perangkat Lunak**

**Semester : 6 (Enam)**

**Dosen : Tifanny Nabarian, S.Kom., M.T.I.**

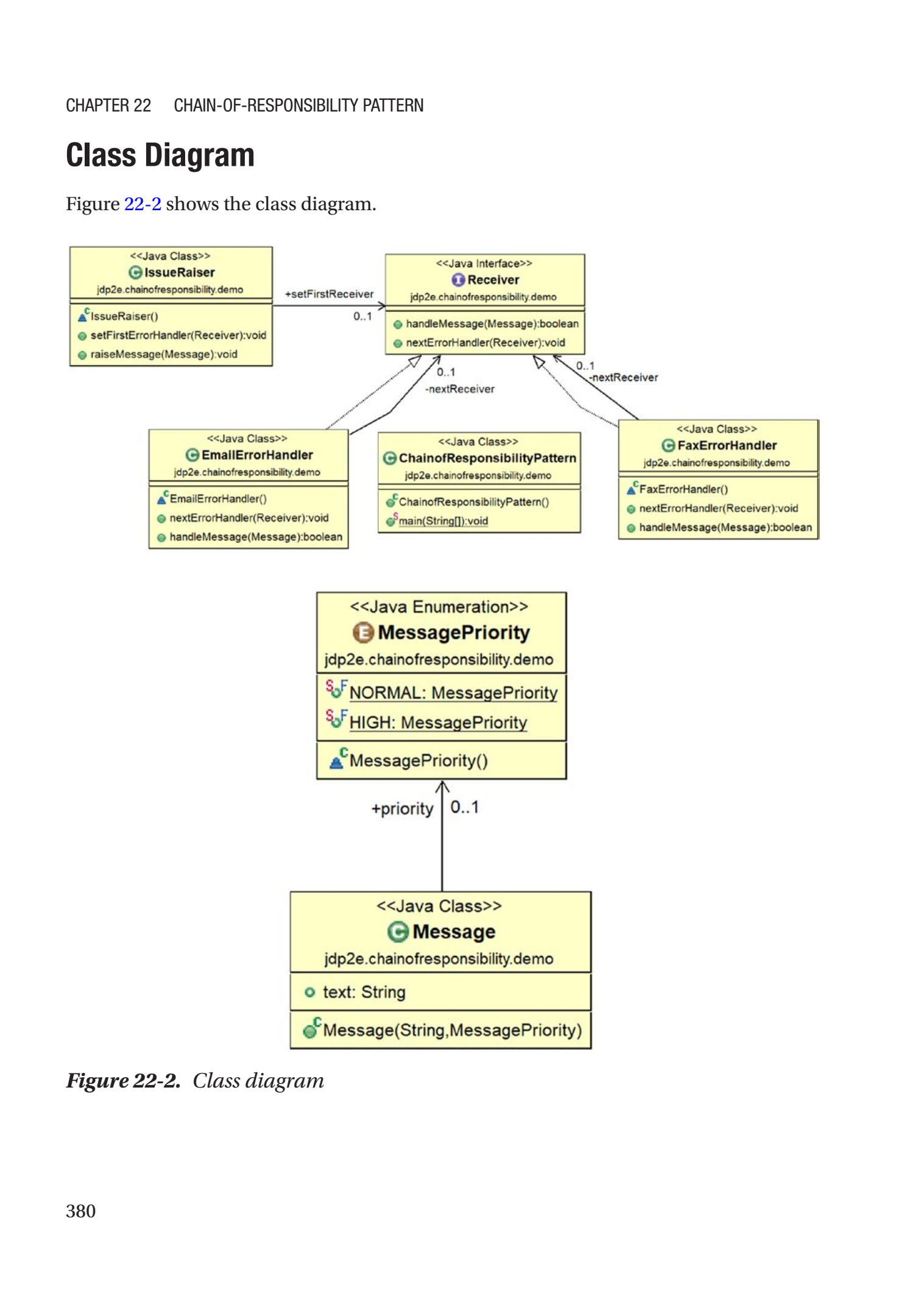
**Nama Mahasiswa : chairil hilman syah**

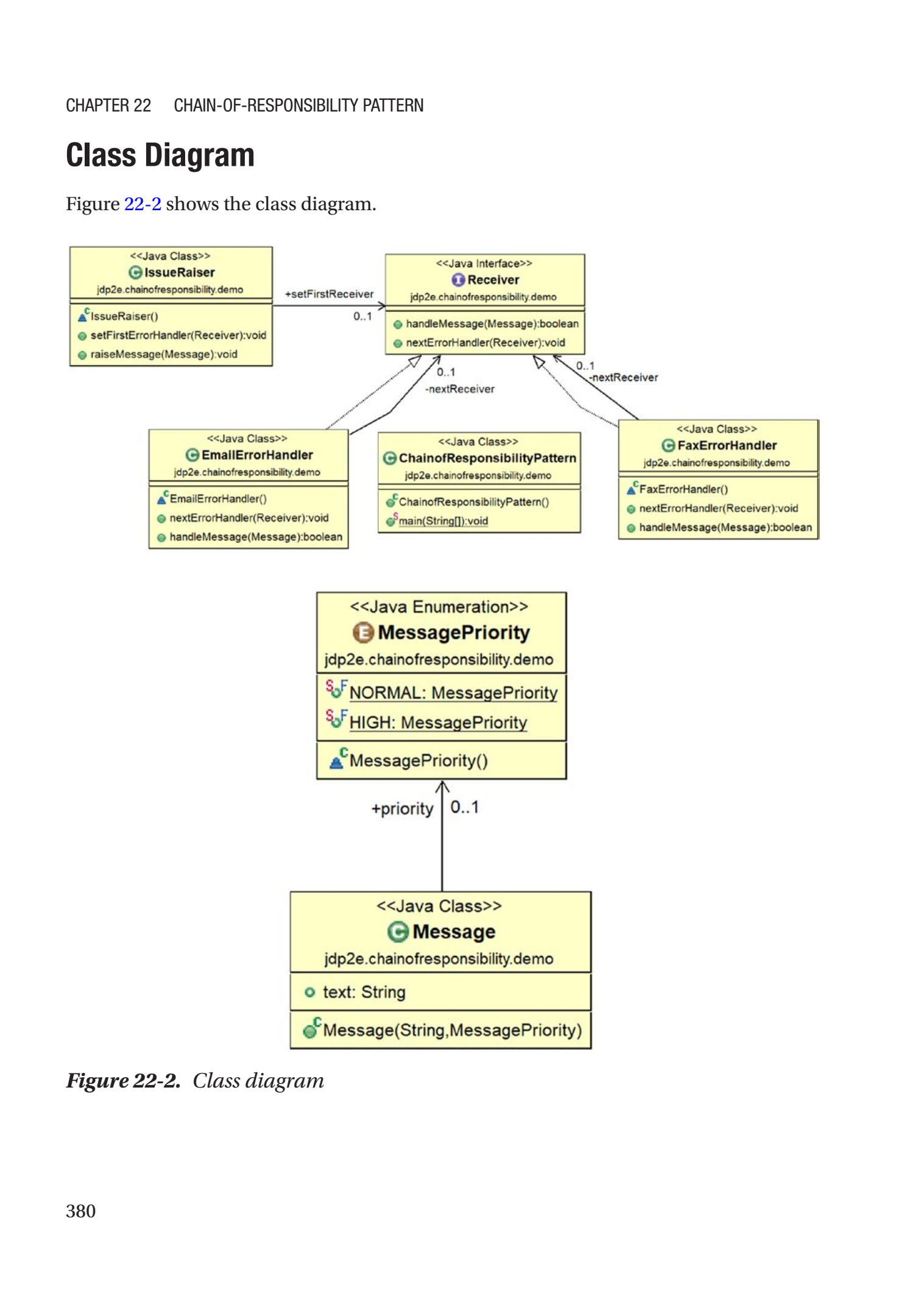
**NIM : 0110217054**

**Instruksi**

* Kerjakan tugas di bawah ini secara **individu.**
* Kerjakan secara **berurutan.**
* Modifikasi *source code*, dengan cara menambahkan **NIM** Anda pada setiap *caption* yang ditampilkan di *result* yang di*screenshot*.

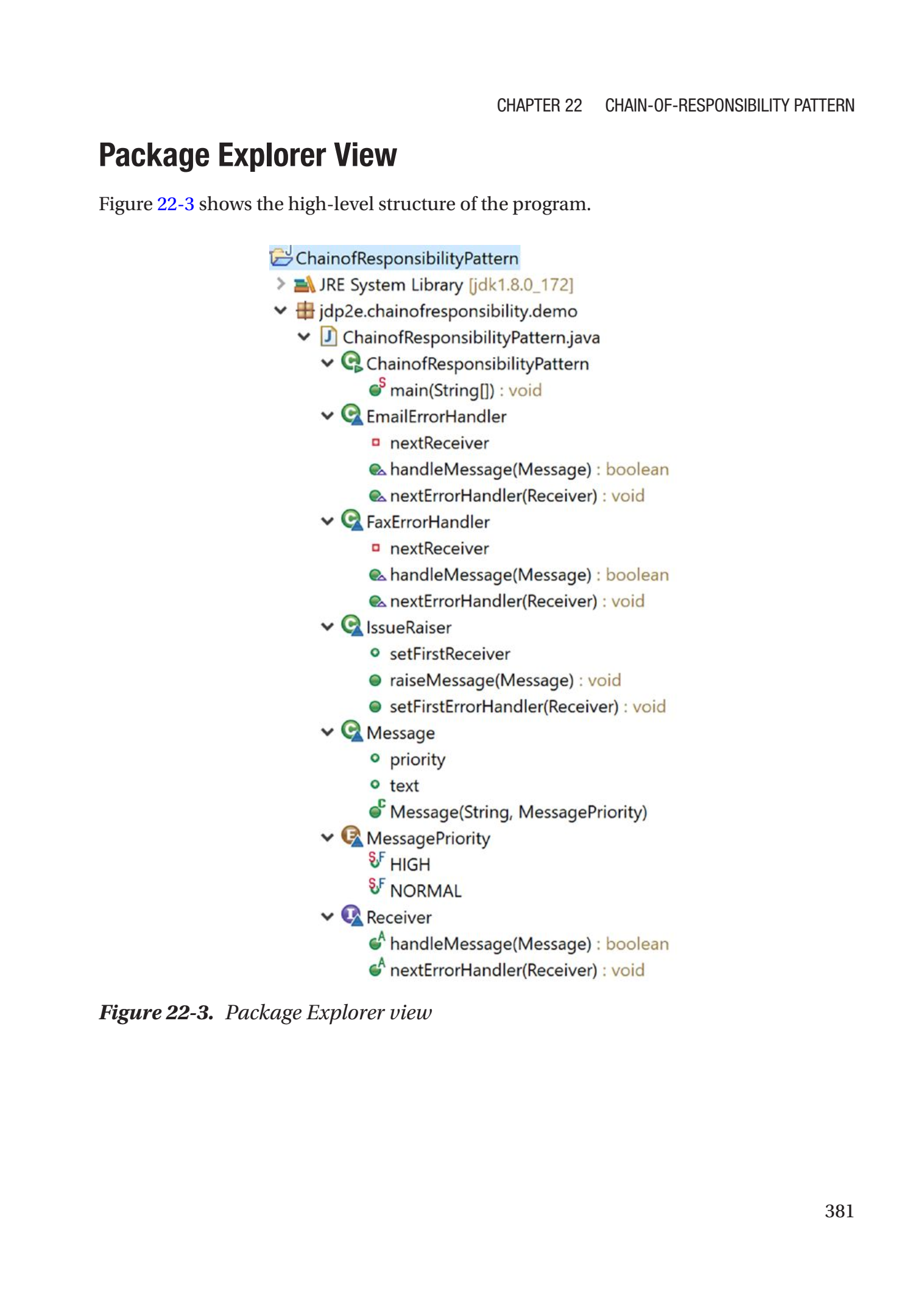
1. **Perhatikan ilustrasi berikut ini:**





Gambar 1. Ilustrasi Relasi

1. **Pahamilah struktur package yang akan dibuat.**



Gambar 2. Package Explorer

1. **Buatlah Kelas MessagePriority dan Message dengan source sebagai berikut:**

**MessagePriority.java**

|  |
| --- |
| **package** jdp2e.chainofresponsibility.demo;  **public** **enum** MessagePriority {  ***NORMAL***,  ***HIGH***  } |

**Message.java**

|  |
| --- |
| **package** jdp2e.chainofresponsibility.demo;  **public** **class** Message {  **public** String text;  **public** MessagePriority priority;  **public** Message(String msg, MessagePriority p)  {  text = msg;  **this**.priority = p;  }  } |

1. **Lanjutkan dengan membuat interface Receiver, kelas IssueRaiser, FaxErrorHandler dan EmailErrorHandler dengan sourcode sbb.,**

**Receiver.java**

|  |
| --- |
| **package** jdp2e.chainofresponsibility.demo;  **public** **interface** Receiver {  **boolean** handleMessage(Message message);  **void** nextErrorHandler(Receiver nextReceiver);  } |

**IssueRaiser.java**

|  |
| --- |
| **package** jdp2e.chainofresponsibility.demo;  **public** **class** IssueRaiser {    **public** Receiver setFirstReceiver;    **public** **void** setFirstErrorHandler(Receiver firstErrorHandler)  {  **this**.setFirstReceiver = firstErrorHandler;  }  **public** **void** raiseMessage(Message message)  {  **if** (setFirstReceiver != **null**)  setFirstReceiver.handleMessage(message);  }  } |

**FaxErrorHandler.java**

|  |
| --- |
| **package** jdp2e.chainofresponsibility.demo;  **public** **class** FaxErrorHandler **implements** Receiver{  **private** Receiver nextReceiver;  @Override  **public** **void** nextErrorHandler(Receiver nextReceiver)  {  **this**.nextReceiver = nextReceiver;  }    @Override  **public** **boolean** handleMessage(Message message)  {  **if** (message.text.contains("Fax"))  {    System.***out***.println(" FaxErrorHandler processed " +message.priority +" priority issue :"+ message.text);  **return** **true**;  }  **else**  {  **if** (nextReceiver != **null**)  nextReceiver.handleMessage(message);  }  **return** **false**;  }  } |

**EmailErrorHandler.java,**

|  |
| --- |
| **package** jdp2e.chainofresponsibility.demo;  **public** **class** EmailErrorHandler **implements** Receiver{  **private** Receiver nextReceiver;    @Override  **public** **void** nextErrorHandler(Receiver nextReceiver)  {  **this**.nextReceiver = nextReceiver;  }    @Override  **public** **boolean** handleMessage(Message message)  {  **if** (message.text.contains("Email"))  {  System.***out***.println(" EmailErrorHandler processed "+message.  priority+ " priority issue: "+message.text);  **return** **true**;  }  **else**  {  **if** (nextReceiver != **null**)  nextReceiver.handleMessage(message);  }  **return** **false**;  }  } |

1. **Lanjutkan dengan membuat kelas client -> ChainofResponsibilityPattern, dengan source sbb.:**

**ChainofResponsibilityPattern.java**

|  |
| --- |
| **package** jdp2e.chainofresponsibility.demo;  **public** **class** ChainofResponsibilityPattern {  **public** **static** **void** main(String[] args)  {    System.***out***.println("\n \*\*\*Chain of Responsibility Pattern Demo\*\*\*\n");    /\* Forming the chain as IssueRaiser->FaxErrorhandler->  EmailErrorHandler\*/  Receiver faxHandler, emailHandler;    //Objects of the chains  IssueRaiser issueRaiser = **new** IssueRaiser();  faxHandler = **new** FaxErrorHandler();  emailHandler = **new** EmailErrorHandler();    //Making the chain  //Starting point:IssueRaiser will raise issues and set the first  //handler  issueRaiser.setFirstErrorHandler(faxHandler);    //FaxErrorHandler will pass the error to EmailHandler if needed.  faxHandler.nextErrorHandler(emailHandler);    //EmailErrorHandler will be placed at the last position in the chain  emailHandler.nextErrorHandler(**null**);    Message m1 = **new** Message("Fax is going slow.",MessagePriority.***NORMAL***);  Message m2 = **new** Message("Emails are not reaching.", MessagePriority.***HIGH***);  Message m3 = **new** Message("In Email, CC field is disabled always.", MessagePriority.***NORMAL***);  Message m4 = **new** Message("Fax is not reaching destinations.", MessagePriority.***HIGH***);  issueRaiser.raiseMessage(m1);  issueRaiser.raiseMessage(m2);  issueRaiser.raiseMessage(m3);  issueRaiser.raiseMessage(m4);  }  } |

1. **Update source pada langkah 5, dengan menampilkan NIM Anda. Kemudian jalankan *main* program tersebut.**

|  |
| --- |
|  |

1. Berikan penjelasan terkait proses implementasi *Chain of Responsibility* pada source 3,4 dan 5.

|  |
| --- |
| pada Langkah ke 3, di tahap ini kita membuat sebuah class yang dmna nnti akan menjadi parameter prioritas massage apakah normal atau high.dan class untuk massage nya.  Pada Langkah ke 4. Di tahap ini kita membuat 4 class dimana terdiri dari , **Receiver.java,** class ini berfungsi sebagai class yang nnti akan menjadi receivernya.  **IssueRaiser.java,** class ini berfungsi sebagai pengatur request yang di kirim.  **EmailErrorHandler.java dan FaxErrorHandler.java,**  class ini berfungsi untuk mengecek error yang dikirimkan.  Pada Langkah ke 5. Tahap ini kita membuat program utama *Chain of Responsibility.* |

1. Bagaimana cara object issueRaiser mengetahui bahwa *request* yang dikirim diproses oleh *errorhandler* yang terakhir (***end of chain***)?

|  |
| --- |
| Menurut pendapat saya, issueRaiser dapat me:ngetahui request yang dikirm diproses karena terdapat source code berikut  **if** (setFirstReceiver != **null**)  setFirstReceiver.handleMessage(message);  jika dilihat dari source code, setfirstreceiver tidak sama dengan null maka setreceiver.handlemessage. nah apabila setreceiver sama dengan null maka handle akan mengetahui bahwa tidak ada lagi setreceiver. |

**\*\*\***