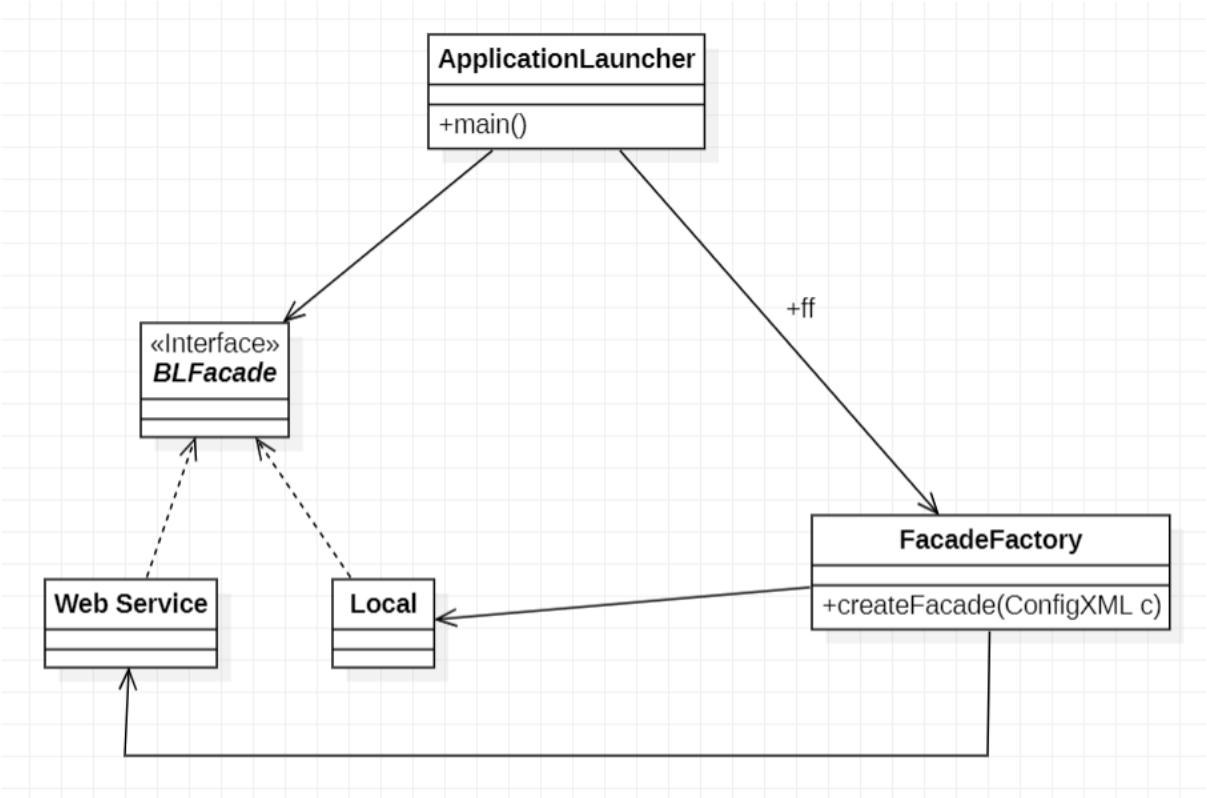
DISEINU PATROIEN PROIEKTUA

Damian Vela
Lander Soriano
Ander Berruezo

Github esteka: https://github.com/anderberru/Bets21

Factory Method Patroia



-Creator: FacadeFactory
-Product: BLFacade interfazea
-Concrete product: Web Service edo Local motako FacadeImplementation klaseak

FacadeFactory izeneko klase berri bat sortu dugu, facade locala edo web service motatakoa sortzeaz arduratzen dena. Klase honen metodo bakarra createFacade() da, ApplicationLauncher klaseko kode zati batetik mugitu duguna. Horrela, facade mota berri bat sortu nahi badugu edo sortzeko metodoan aldaketak egin nahi baditugu, bakarrik FacadeFactory klasea aldatu behar dugu, ez ApplicationLauncher klaseko main metodoa.

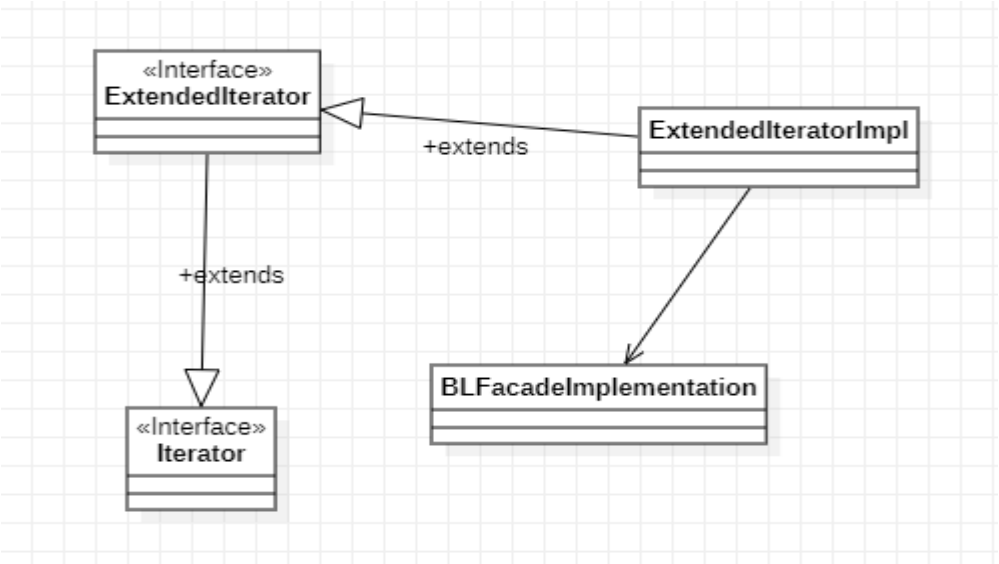
Hau da FacadeFactory klaseko createFacade() metodoa, BLFacade motako objetu bat itzultzen duena:

```
16    public BLFacade createFacade(ConfigXML c) throws MalformedURLException {
17
18        BLFacade appFacadeInterface;
19
20        if (c.isBusinessLogicLocal()) {
21
22            //In this option the DataAccess is created by FacadeImplementationWS
23            //appFacadeInterface=new BLFacadeImplementation();
24
25            //In this option, you can parameterize the DataAccess (e.g. a Mock DataAccess object)
26
27            DataAccess da= new DataAccess(c.getDataBaseOpenMode().equals("initialize"));
28
29            appFacadeInterface=new BLFacadeImplementation(da);
30
31        }
32
33        else { //If remote
34
35             String serviceName= "http://"+c.getBusinessLogicNode() +":"+ c.getBusinessLogicPort()+"/ws/"+c.getBusinessLogicName()+"?wsdl";
36
37            //URL url = new URL("http://localhost:9999/ws/ruralHouses?wsdl");
38            URL url = new URL(serviceName);
39
40            //1st argument refers to wsdl document above
41            //2nd argument is service name, refer to wsdl document above
42 //           QName qname = new QName("http://businessLogic/", "FacadeImplementationWSService");
43            QName qname = new QName("http://businessLogic/", "BLFacadeImplementationService");
44
45            Service service = Service.create(url, qname);
46
47             appFacadeInterface = service.getPort(BLFacade.class);
48        }
49
50        return appFacadeInterface;
```

Horrela geratu da ApplicationLauncher klasea, FacadeFactory objetu bat hasieratzen dugu eta bere metodoari deitzen diogu, appFacadeInterface aldagaian gordetzeko:

```
20    public static void main(String[] args) {
21
22        ConfigXML c=ConfigXML.getInstance();
23
24        FacadeFactory ff = new FacadeFactory(); // faktoria hasieratu
25
26        System.out.println(c.getLocale());
27
28        Locale.setDefault(new Locale(c.getLocale()));
29
30        System.out.println("Locale: "+Locale.getDefault());
31
32        MainGUI a=new MainGUI();
33        a.setVisible(true);
34
35
36        try {
37
38            BLFacade appFacadeInterface = ff.createFacade(c); // faktoria deitu
39
```

Iterator patroia



ExtendedIterator interfazea sortu dugu eta Iterator interfazetik hartu ditugu erabiliko ditugun metodoak. Ondoren, ExtendedIteratorImpl klasea sortu dugu metodoak inplementatzeko. BLFacadeImplementation klasean getEventes metodoa aldatu dugu.

```
@WebMethod
public ExtendedIterator<Event> getEvents(Date date) {
    dbManager.open(false);
    List<Event> events = dbManager.getEvents(date); // Assuming getEvents returns a List<Event>
    dbManager.close();
    return new ExtendedIteratorImpl(events);
}
```

Aldaketa horrek eragina izan du getEvents() erabiltzen zituzten beste klase askotan; horregatik, egokitu egin behar dira iterator erabil dezaten. Klase guztiek izan duten aldaketa hau izan da: objektu-mota Vector < Event> izatetik ExtendedIterator < Event> izatera aldatu da. Gero, gehienak for bat zuten, baina iterator ez dabil horrekin, while batekin aldatu egin dugu. Amaitzeko, zerrendaren tamaina 0 dela egiaztatu beharrean, isEmpty metodoa erabiltzen da.

```
    BLFacade facade = MainGUI.getBusinessLogic();

    ExtendedIterator<Event> events = facade.getEvents(firstDay);

    if (events.isEmpty())
        jLabelListOfEvents.setText(ResourceBundle.getBundle("Etiquetas").getString("NoEvents")
                + ": " + dateformat1.format(calendarAct.getTime()));
    else
        jLabelListOfEvents.setText(ResourceBundle.getBundle("Etiquetas").getString("Events") + ": "
                + dateformat1.format(calendarAct.getTime()));
    jComboBoxEvents.removeAllItems();
    System.out.println("Events " + events);

    while (events.hasNext()) {
        Event ev = events.next();
        modelEvents.addElement(ev);
    }
    jComboBoxEvents.repaint();


    if (events.isEmpty())
        jButtonRemove.setEnabled(false);
    else
        jButtonRemove.setEnabled(true);

} catch (Exception e1) {
```

Iterator ondo funtzionatzen duela egiaztatzeko, FindQuestionsGUI klasean metodo bat sortu dugu bi ibilbideak egin eta emaitzak inprimatu ditzan.

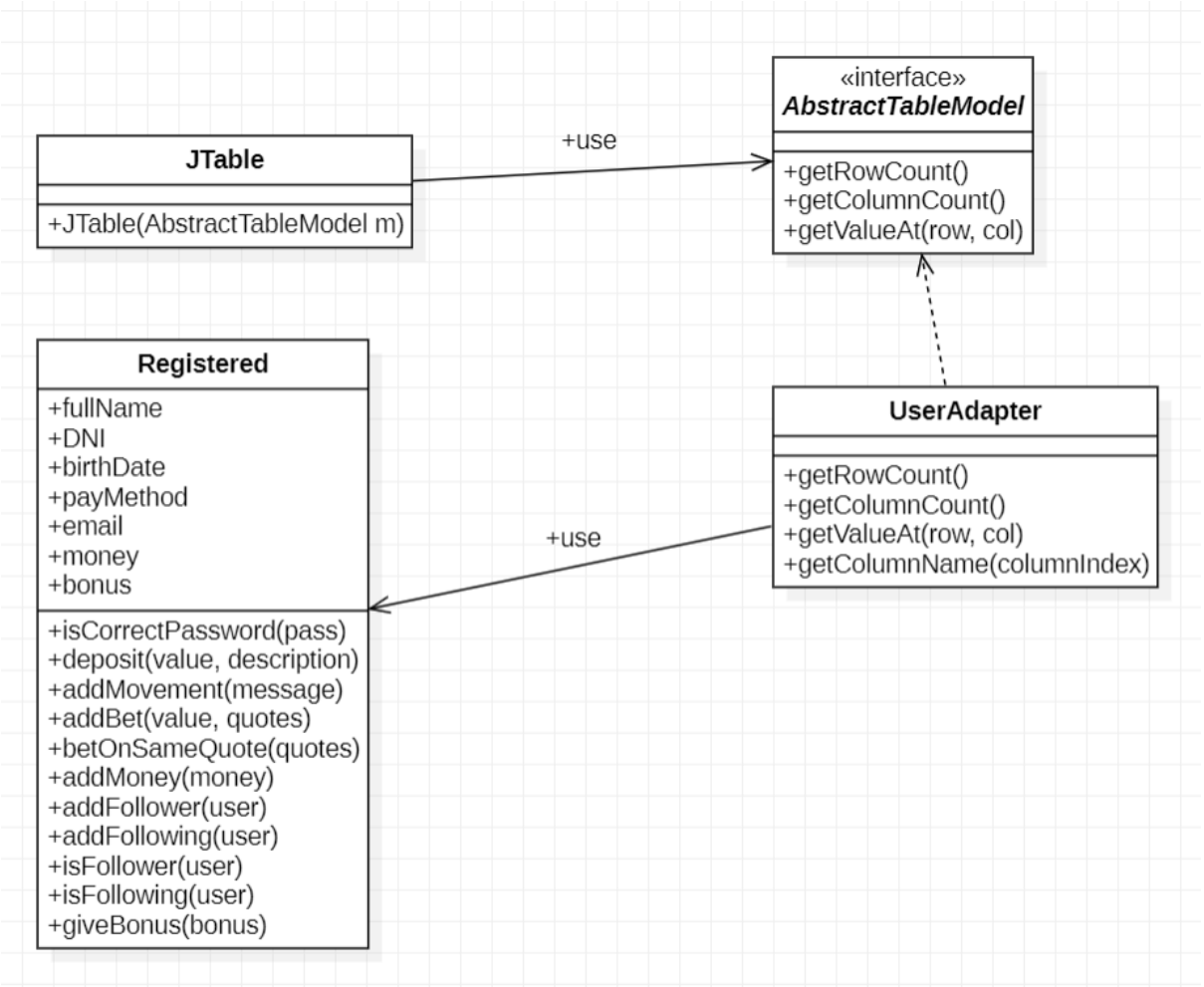
```
private void printDayEvents(ExtendedIterator<Event> i) {
    System.out.println("Atzetik aurrera inprimatzen");
    Event ev;
    i.goLast();
    while (i.hasPrevious()) {
        ev = i.previous();
        System.out.println(ev.toString());
    }
    // Nahiz eta suposatu hasierara ailegatu garela, eragiketa egiten dugu.
    System.out.println("Aurretik atzera inprimatzen");

    i.goFirst();
    while (i.hasNext()) {
        ev = i.next();
        System.out.println(ev.toString());
    }

}
```

```
Atzetik aurrera inprimatzen
9;Real Sociedad-Levante
8;Girona-Legan?s
7;Malaga-Valencia
6;Las Palmas-Sevilla
5;Espa?ol-Villareal
4;Alav?s-Deportivo
3;Getafe-Celta
2;Eibar-Barcelona
1;Atl?tico-Athletic
Aurretik atzera inprimatzen
1;Atl?tico-Athletic
2;Eibar-Barcelona
3;Getafe-Celta
4;Alav?s-Deportivo
5;Espa?ol-Villareal
6;Las Palmas-Sevilla
7;Malaga-Valencia
8;Girona-Legan?s
9;Real Sociedad-Levante
10;Betis-Real Madrid
```

Adapter Patroia



UserAdapter klasea sortu dugu, AbstractTableModel interfazea implementatzen duena, baina metodo berri batekin, getColumnName(), taulako zutabeen izenak lortzeko erabiltzen dena. Klase hau Registered motako erabiltzailea dauka atributu bezala, horrekin metodoak lan egiteko, Registered klasea aldatu behar izan gabe. GUI-an Jtable bat sortzean, UserAdapter motako objetu bat erabiltzen da taula betetzeko.

Hauek dira UserAdapter klaseko metodoak:

```
21●     @Override
22      public int getRowCount() {
23          return user.getBets().size();
24      }
```

Taulan errenkada bakoitzean apustu bat egon behar denez, errenkada kopurua lortzeko erabiltzailearen apustuen kopurua atera behar da.

```
26●     @Override
27      public int getColumnCount() {
28          return 4;
29      }
```

Lau zutabe behar ditugu, gertaera, galdera, data eta apustuaren balioa erakutsi nahi ditugulako.

```java
@Override
public Object getValueAt(int rowIndex, int columnIndex) {
    switch(columnIndex) {
    case 0:
        return user.getBets().get(rowIndex).getQuotes().get(0).getQuestion().getEvent().getDescription();
    case 1:
        return user.getBets().get(rowIndex).getQuotes().get(0).getQuestion().getQuestion();
    case 2:
        return user.getBets().get(rowIndex).getQuotes().get(0).getQuestion().getEvent().getEventDate();
    case 3:
        return user.getBets().get(rowIndex).getValue();
    }
    return null;
}
```

Metodo honetan, taularen errenkada eta zutabea zehaztuz elementu bat lortzen da. Zutabearen(columnIndex) arabera datu mota aukeratzen da, eta errenkada(rowIndex) zenbakiarekin apustua aukeratzen da.

```java
public String getColumnName(int columnIndex) {
    switch(columnIndex) {
    case 0:
        return "Event";
    case 1:
        return "Question";
    case 2:
        return "Event Date";
    case 3:
        return "Bet (€)";
    }
    return "";
}
```

Metodo hau zutabe bakoitzaren izena zehazteko balio du, zenbakiaren arabera zutabe baten izena lortzen da.

```java
user = facade.getUser(username);
adapter = new UserAdapter(user);
table = new JTable(adapter);
scrollPane.setViewportView(table);
```

Azkenik JTable motako taula TableGUI klasean sortzen dugu, eta logeatu den erabiltzailea sartzen dugu UserAdapter instantzian, gero taulan erabiltzeko.

Hemen ikus daiteke programaren exekuzioa, RegisteredGUI klasean botoi bat gehitu dugu(Table of “user”), taularen leihoa irekitzeko. Kasu honetan “a” erabiltzailearen apustu guztiak agertzen dira: