

DISEINU PATROIAK

Mikel Oscoz Rivas
Iñigo Gallegos Iglesias

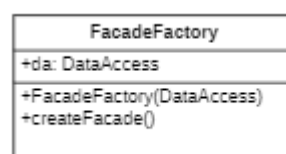
IN

a)UML diagrama hedatua egin dituzun aldaketak aurkeztuz.	3
1- FacadeFactory	3
2- Iterator	3
3- Adapter	4
b)Aldatu duzun kodea, lerro garrantzitsuenak azalduz	5
Factory Method Patroia	5
Iterator patroia	6
Adapter Patroia	10
c) Iterator eta Adapter patroientzako, exekuzioaren irudi bat	13
Iterator patroia exekuzioa:	13
Adapter patroia exekuzioa:	14

a)UML diagrama hedatua egin dituzun aldaketak aurkeztuz.

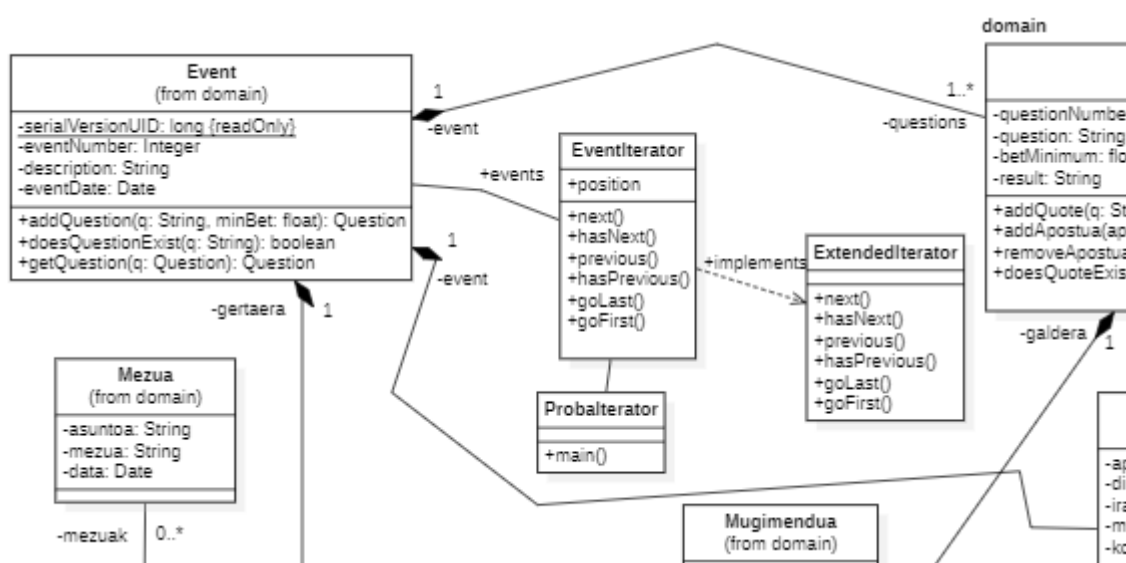
1- FacadeFactory

Lehenengo aldaketa FacadeFactory gehitzea izan da, hau gui paketea egin dugu, gero ApplicationLauncher-a erabiltzeko. Klase honek DataAccess atributua du, eraikitzailean eskatzen dena. Eta gero createFacade-rako erabiltzen du pasatako DataAccess-a.



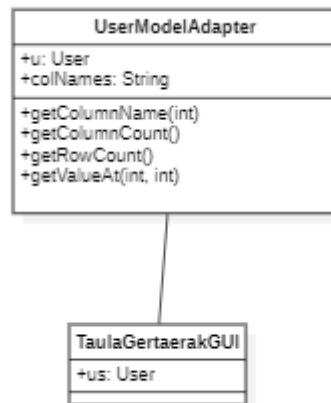
2- Iterator

Patroi honetarako, ExtendedIterator interfazea, EventIterator klasea eta Probalterator main-a sortu ditugu. ExtendedIterator-an soilik metodoen sinadurak daude. Eta EventIterator klasean implementatzen da. Probalterator, main bat da, eta hor dena probatzen da. Honetarako, aldaketa batzuk egin behar genituen BLFacade-n eta DataAccess-en getEvents berri bat egiten.



3- Adapter

Kasu honetan, gui paketea bi klase sortu genituen, UserModelAdapter eta TaulaGertaerakGUI. Lehenengoan AbstractTableModel-eko klaseko metodoetan override egiten da implementazioa gure kasura aldatzeko, horrela JTable bat erabiliz, gure kasu-ra egokitzen den taula bat eraiki dugu. TaulaGertaerakGUI, taula hau ikuskatzeko da, eta hemen ailegatzeko, botoi bat gehitu dugu UserGUI-an



b) Aldatu duzun kodea, lerro garrantzitsuenak azalduz

Factory Method Patroia

FacadeFactory klasea:

```
public class FacadeFactory {

    private DataAccess da;

    public FacadeFactory(DataAccess d) {
        this.da=d;
    }

    public BLFacade createFacade() {
        return new BLFacadeImplementation(this.da);
    }
}
```

Klase honek Facade instantzia berri bat sortzen du datu base batentzako.

ApplicationLauncher klasea:

```
try {

    BLFacade appFacadeInterface;
    UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsClassicLookAndFeel");
    UIManager.setLookAndFeel("com.sun.java.swing.plaf.motif.MotifLookAndFeel");
    UIManager.setLookAndFeel("javax.swing.plaf.metal.MetalLookAndFeel");

    if (c.isBusinessLogicLocal()) {

        //In this option the DataAccess is created by FacadeImplementationWS
        //appFacadeInterface=new BLFacadeImplementation();

        //In this option, you can parameterize the DataAccess (e.g. a Mock DataAccess object)

        DataAccess da= new DataAccess(c.getDataBaseOpenMode().equals("initialize"));
        FacadeFactory ff = new FacadeFactory(da);

        //appFacadeInterface=new BLFacadeImplementation(da);
        appFacadeInterface=ff.createFacade();

    }
}
```

Orain, objektua modu normalean sortzea ordeztu, ff faktoria erabiltzen da, parametro moduan DataAccess-a sartuz.

Iterator patroia

BLFacade klasea:

```
//-----ALDAKETA-----
@WebMethod public EventIterator getEventss(Date date);
//-----END_ALDAKETA-----
```

BLFacadeImplementation klasea:

```
//-----ALDAKETA-----
/**
 * This method invokes the data access to retrieve the events of a given date
 *
 * @param date in which events are retrieved
 * @return collection of events
 */
@WebMethod
public EventIterator getEventss(Date date) {
    dbManager.open(false);
    EventIterator events = dbManager.getEventss(date);
    dbManager.close();
    return events;
}
//-----END_ALDAKETA-----
```

DataAcces klasean:

```
//-----ALDAKETA-----
/**
 * This method retrieves from the database the events of a given date
 *
 * @param date in which events are retrieved
 * @return collection of events
 */
public EventIterator getEventss(Date date) {
    System.out.println(">> DataAccess: getEvents");
    Vector<Event> res = new Vector<Event>();
    TypedQuery<Event> query = db.createQuery("SELECT ev FROM Event ev WHERE ev.eventDate=?1", Event.class);
    query.setParameter(1, date);
    List<Event> events = query.getResultList();
    for (Event ev : events) {
        System.out.println(ev.toString());
        res.add(ev);
    }
    EventIterator evi = new EventIterator();
    for(Event ev : res) {
        evi.events.add(ev);
    }
    return evi;
}
//-----END_ALDAKETA-----
```

Aldaketa hauek egin ditugu, getEventss EventIterator moduko objektu bat bueltatzeko.

Extended Iterator interfazea:

```
package businessLogic;

public interface ExtendedIterator {

    //uneko elementua itzultzen du eta hurrengora pasatzen da
    public Object next();

    //true hurrengo elementua existitzen bada.
    public boolean hasNext();

    //uneko elementua itzultzen du eta aurrekora pasatzen da
    public Object previous();

    //true aurreko elementua existitzen bada.
    public boolean hasPrevious();

    //Lehendabiziko elementuan kokatzen da.
    public void goFirst();

    //Azkeneko elementuan kokatzen da.
    public void goLast();

}
```

Interfazea iterator metodoaren siadurarekin.

EventIterator klasea:

```
public class EventIterator implements ExtendedIterator {

    public Vector<Event> events=new Vector<Event>();
    public int position = 0;

    public Event next() {
        Event ev = events.elementAt(position);
        position = position + 1;
        return ev;
    }

    public boolean hasNext() {
        return position < events.size();
    }

    public Event previous() {
        Event ev = events.elementAt(position);
        position = position - 1;
        return ev;
    }

    public boolean hasPrevious() {
        return position >= 0;
    }

    public void goFirst() {
        position = 0;
    }

    public void goLast() {
        position = events.size() - 1;
    }

}
```

Iteratorak erabiliko duen metodoak implementatuta daude:

next(): hurrengo elementua hartzen du.

hasNext(): ikusten du hurrengo posizioan elementuak dauden ala ez.

previous(): aurreko elementura joaten da.

hasprevious(): ikusten du aurreko posizioan elementuak dauden ala ez.

goFirst(): Listaren lehengo posiziora joaten da.

goLast(): Listaren azkeneko posiziora joaten da.

Probalterator:

```
public static void main(String[] args) {
    boolean isLocal = true;

    Calendar today = Calendar.getInstance();

    int month = today.get(Calendar.MONTH);
    month += 1;
    int year = today.get(Calendar.YEAR);
    if (month == 12) {
        month = 0;
        year += 1;
    }

    // Facade objektua lortu lehendabiziko ariketa erabiliz
    BLFacade facadeInterface = new BLFacadeImplementation();
    Date d = new Date(year, month, 17);
    //EventIterator i = facadeInterface.getEvents(d);
    EventIterator i = facadeInterface.getEventss(d);
    i.events.add(new Event(3, "Real Sociedad", UtilDate.newDate(year, month, 17)));
    i.events.add(new Event(2, "Barcelona", UtilDate.newDate(year, month, 17)));
    i.events.add(new Event(1, "Atlético-Athletic", UtilDate.newDate(year, month, 17)));
    Event ev;

    i.goLast();
    while (i.hasPrevious()) {
        ev = i.previous();
        System.out.println("Primero "+ev.toString());
    }
    // Nahiz eta suposatu hasierara ailegatu garela, eragiketa egiten dugu.
    i.goFirst();
    while (i.hasNext()) {
        ev = i.next();
        System.out.println("Segundo "+ev.toString());
    }
}
```

Hemen soilik proba batzuk egiten dira, frogatzeko EventIterator ondo dabilela.

Adapter Patroia

UserModelAdapter klasea :

```
private User u;
private String[] colNames = new String[] { "Event", "Question", "Date", "Bet" };

public UserModelAdapter(User u) {
    // copy the HashMap data to a sequential data structure
    this.u = u;
}

@Override
public String getColumnName(int col) {
    return colNames[col];
}

@Override
public int getColumnCount() {
    return 4;
}

@Override
public int getRowCount() {
    return u.getMugimenduak().size();
}

@Override
public Object getValueAt(int rowIndex, int columnIndex) {
    if(u.getMugimenduak().get(rowIndex).getGertaera() != null) {
        switch (columnIndex) {
            case 0:
                return ((Object) u.getMugimenduak().get(rowIndex).getGertaera());
            case 1:
                return ((Object) u.getMugimenduak().get(rowIndex).getGalderaText());
            case 2:
                return ((Object) u.getMugimenduak().get(rowIndex).getGertaera().getEventDate());
            case 3:
                return ((Object) u.getMugimenduak().get(rowIndex).getDiruKop());
        }
    }
    return null;
}
```

Klase honek AbstractTableModel klasea “extends” egiten du, eta metodo batzuetan override egiten du. Hauek aldatu ditugu, gure kasura egokitzeko, kolumna kopuru zehatz bat jartzen eta errenkada bakoitzan nahi genuen informazioa.

TaulaGertaerakGUI klasea:

```
public class TaulaGertaerakGUI extends JFrame {

    private User us;

    /**
     * Create the frame.
     */
    public TaulaGertaerakGUI(User u) {
        UserModelAdapter uma =new UserModelAdapter(u);

        JFrame j=new JFrame();
        JTable table = new JTable(uma);
        j.add(new JScrollPane(table));
        table.removeAll();
        j.setTitle(u.getIzena()+"-ren apostuak");
        j.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        j.pack();
        j.setVisible(true);
    }
}
```

UserGUI klasea:

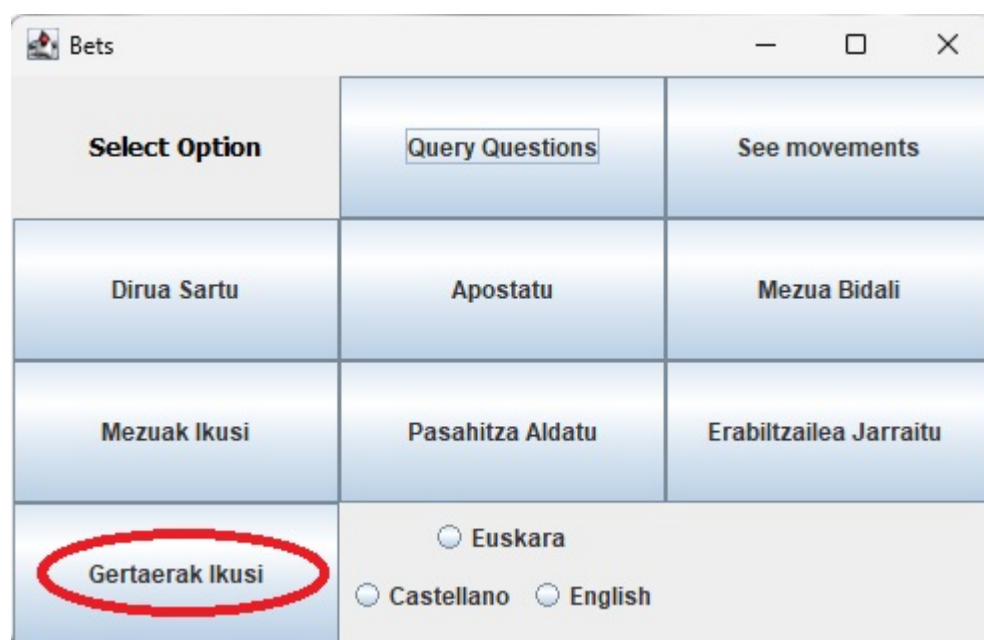
```
private JButton jButtonGertaerak=null;

jContentPane.add(getBoton11());

private JButton getBoton11() {
    if (jButtonGertaerak == null) {
        User u = user;
        jButtonGertaerak = new JButton();
        jButtonGertaerak.setText(ResourceBundle.getBundle("Etiquetas").getString("SeeEvents"));
        jButtonGertaerak.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent e) {
                JFrame a = new TaulaGertaerakGUI(u);
                a.setVisible(true);
            }
        });
    }
    return jButtonGertaerak;
}
```

Hemen, aurreko adapter-a erabiliz, taula interfaze batean erakuzten da.

Botoi berria:



The screenshot shows a window titled "Bets" with a standard Windows-style title bar (minimize, maximize, close buttons). The main content area is a grid of buttons and a language selection section. The grid has three columns and four rows. The first row contains a header "Select Option" and two buttons: "Query Questions" and "See movements". The second row contains three buttons: "Dirua Sartu", "Apostatu", and "Mezua Bidali". The third row contains three buttons: "Mezuak Ikusi", "Pasahitza Aldatu", and "Erabiltzailea Jarraitu". The fourth row contains a button "Gertaerak Ikusi" (circled in red) and a language selection section with three radio buttons: "Euskara", "Castellano", and "English".

Select Option	Query Questions	See movements
Dirua Sartu	Apostatu	Mezua Bidali
Mezuak Ikusi	Pasahitza Aldatu	Erabiltzailea Jarraitu
Gertaerak Ikusi	<input type="radio"/> Euskara <input type="radio"/> Castellano <input type="radio"/> English	

c) Iterator eta Adapter patroientzako, exekuzioaren irudi bat

Iterator patroia exekuzioa:

```

Creating BLFacadeImplementation instance
Read from config.xml:    businessLogicLocal=true        databaseLocal=true        dataBaseOpenMode=open
Creating DataAccess instance => isDatabaseLocal: true getDatabBaseOpenMode: open
Opening DataAccess instance => isDatabaseLocal: true getDatabBaseOpenMode: open
DataBase closed
Opening DataAccess instance => isDatabaseLocal: true getDatabBaseOpenMode: open
>> DataAccess: getEvents
DataBase closed
newDate: Sun Dec 17 00:00:00 CET 2023
newDate: Sun Dec 17 00:00:00 CET 2023
newDate: Sun Dec 17 00:00:00 CET 2023
Primero 1;Atlético-Athletic
Primero 2;Barcelona
Primero 3;Real Sociedad
Segundo 3;Real Sociedad
Segundo 2;Barcelona
Segundo 1;Atlético-Athletic

```

Adapter patroia exekuzioa:

izena-ren apostuak			
Event	Question	Date	Bet
1;Atlético-Athletic	Who will win the ...	Sat Jun 17 00:00:...	100.0
13;Getafe-Celta	Nork irabazi?	Thu Jun 01 00:00:...	1.215752192E9
12;Eibar-Barcelona	Nork irabazi?	Thu Jun 01 00:00:...	1.215752142E9
12;Eibar-Barcelona	Nork irabazi?	Thu Jun 01 00:00:...	1.215752092E9
14;Alavés-Deportivo	Nork irabazi?	Thu Jun 01 00:00:...	1.215752042E9
16;null			1.215752492E9
3;Getafe-Celta	Nork irabazi?	Sat Jun 17 00:00:...	1.215752642E9
98;Eibar-Barcelona	Nork irabazi?	Sun Jun 04 00:00:...	1.215752592E9
109;c-d	porque	Wed May 31 00:00:...	1.215752802E9
4;Alavés-Deportivo	Nork irabazi?	Sat Jun 17 00:00:...	1.215752752E9
5;Español-Villareal	Nork irabazi?	Sat Jun 17 00:00:...	1.215752652E9
7;Malaga-Valencia	Nork irabazi?	Sat Jun 17 00:00:...	1.215742208E9
8;Girona-Leganés	Nork irabazi?	Sat Jun 17 00:00:...	1.215742199E9
164;a-b	nork irabazi	Tue May 30 00:00:...	1.215741533E9
165;c-d	nork irabazi	Tue May 30 00:00:...	1.2157415E9