

PROYECTO CASA DE APUESTAS **PATRONES DE DISEÑO**

Carlos Vilarchao-Asier Alcibar-Abraham Casas

1. Patrón Factory Method

Código modificado:

```
public class BLFactory {
    private final ConfigXML c = ConfigXML.getInstance();
    public BLFacade getBusinessLogicFactory(int isLocal) throws
    MalformedURLException {
        // TODO Auto-generated method stub
        if(isLocal==1) {
            return new BLFacadeImplementation(new
    DataAccess(c.getDataBaseOpenMode().equals("initialize")));
        }
        else if(isLocal==0) {
            String serviceName= "http://"+c.getBusinessLogicNode() +":"+c.getBusinessLogicPort()+"/ws/"+c.getBusinessLogicName()+"?wsdl";

            //URL url = new
    URL("http://localhost:9999/ws/ruralHouses?wsdl");
            URL url = new URL(serviceName);

            //1st argument refers to wsdl document above
            //2nd argument is service name, refer to wsdl document above
            //
            QName qname = new QName("http://businessLogic/",
    "FacadeImplementationWSService");
            QName qname = new QName("http://businessLogic/",
    "BLFacadeImplementationService");

            Service service = Service.create(url, qname);

            return service.getPort(BLFacade.class);
        }
        else return null;
    }
}

}

-----
public static void main(String[] args) {

    ConfigXML c=ConfigXML.getInstance();
    BLFactory blf = new BLFactory();

    System.out.println(c.getLocale());

    Locale.setDefault(new Locale(c.getLocale()));
```

```

        System.out.println("Locale: "+Locale.getDefault());

        MainGUI a=new MainGUI();
        a.setVisible(false);

        MainUserGUI b = new MainUserGUI();
        b.setVisible(true);

        try {

            BLFacade appFacadeInterface;

            //
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsClassicLook
            AndFeel");
            //
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.motif.MotifLookAndFeel");

            UIManager.setLookAndFeel("javax.swing.plaf.metal.MetalLookAndFeel");
            appFacadeInterface =
            blf.getBusinessLogicFactory(c.isBusinessLogicLocal() ? 1 : 0);
            MainGUI.setBussinessLogic(appFacadeInterface);

        }catch (Exception e) {
            a.jLabelSelectOption.setText("Error: "+e.toString());
            a.jLabelSelectOption.setForeground(Color.RED);

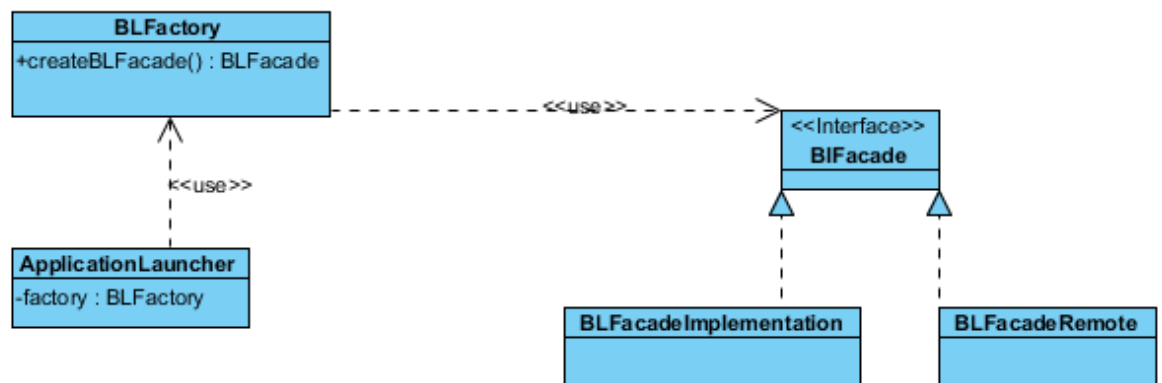
            System.out.println("Error in ApplicationLauncher:
            "+e.toString());
        }
        //a.pack();

    }

}

```

Básicamente lo que he hecho ha sido implementar una clase factoría que me genere las instancias de la jerarquía de clases en vez de hacerlo a pñón el método principal. La clase BLFcadeRemote no existe como tal, pero es para que se vea bien la estructura del patrón.



2. Patron Iterator

Código modificado:

```
@WebMethod public ExtendedIterator<Event> getEventsIterator(Date date);
```

Añadimos a la interfaz BLFacade el método que queremos implementar en BLFacadeImplementation, qué es un método que en vez de devolvemos un listado de eventos nos devuelve un Iterador de eventos. Sin embargo, este nuevo “Iterador extendido”, además de poder recorrer los elementos de la forma tradicional (secuencialmente hacia adelante), puede ir secuencialmente hacia atrás, o bien posicionarse en el primer o último elemento.

```
@WebMethod
    public ExtendedIterator<Event> getEventsIterator(Date date){

        dbManager.open(false);
        ArrayList<Event> events=dbManager.getEvents(date);
        dbManager.close();
        ExtendedIteratorEvents eie = new ExtendedIteratorEvents(events);
        return eie;

    }
```

Implementamos el método en BLFacadeImplementation, este método imita al método de getEvents normal pero lo que devuelve es un objeto ExtendedIteratorEvents. Ahora tenemos que crear la interfaz ExtendedIterator<Event> y la clase ExtendedIteratorEvents que la implementa

```
package businessLogic;
```

```
import java.util.Iterator;
```

```

public interface ExtendedIterator<Object> extends
Iterator<Object> {
    //return the actual element and go to the previous
    public Object previous();
    //true if there is a previous element
    public boolean hasPrevious();

    //It is placed in the first element
    public void goFirst();
    // It is placed in the last element
    public void goLast();
}

```

Esta interfaz extiende de iterador por lo que ya tiene implícitos los métodos de hasNext() y next(), pero como hemos dicho arriba queremos poder recorrer de más maneras las listas, queremos por ejemplo ir hacia atrás o ir al último de estas, para eso le añadimos las cabeceras de los métodos nuevos que harán lo anterior.

```

-----
package businessLogic;

import java.util.List;

import domain.Event;

public class ExtendedIteratorEvents implements
ExtendedIterator<Event> {

    List<Event> l;
    int posList = 0;

    public ExtendedIteratorEvents(List<Event> pList) {

        this.l = pList;

    }

    public boolean hasNext() {

        return posList<l.size();
    }

    public Event next() {

        Event event = l.get(posList);
        posList++;
        return event;
    }
}

```

```

    public Event previous() {

        posList--;
        Event event = l.get(posList);
        return event;
    }

    public boolean hasPrevious() {

        if(posList!=0) return true;
        else return false;
    }

    public void goFirst() {

        posList=0;

    }

    public void goLast() {

        posList=l.size();

    }

}

```

Esta clase es el objeto que devuelve el método nuevo que hemos implementado en BLFacadeImplementation, y la clase que implementa la nueva interfaz de nuestro nuevo tipo de iterador. Esta clase tiene dos atributos uno para la lista que le van a pasar y otro para guardar la posición de la lista, y acaba implementando todos los métodos de nuestra interfaz iterador nueva, incluidos los implícitos por el extends al iterador normal.

```

package businessLogic;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;

import com.sun.org.apache.xerces.internal.impl.xpath.regex.ParseException;

import configuration.ConfigXML;
import dataAccess.DataAccess;
import domain.Event;

public class Main {

```

```

public static void main(String[] args) {

    int eL = 1;
    BLFacade blFacade = new BLFactory().getBusinessLogicFactory(eL);
    SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
    Date date = null;
    DataAccess dbManager;
    ConfigXML c=ConfigXML.getInstance();

    if (c.getDataBaseOpenMode().equals("initialize")) {
        dbManager=new DataAccess(c.getDataBaseOpenMode().equals("initialize"));
        dbManager.initializeDB();
    }
    else dbManager=new DataAccess();
    dbManager.close();

    try {

        try {
            date = sdf.parse("17/12/2023");
        } catch (java.text.ParseException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
        ExtendedIterator<Event> eie = blFacade.getEventsIterator(date);
        dbManager.open(false);
        ArrayList<Event> events=dbManager.getList(date);
        dbManager.close();
        Event e;
        System.out.println("_____");
        System.out.println("RECORRIDO HACIA ATRÁS");
        eie.goLast(); // Hacia atrás

        while (eie.hasPrevious()) {
            e = eie.previous();
            System.out.println(e.toString());
        }

        System.out.println("_____");
        System.out.println("RECORRIDO HACIA ADELANTE");
        eie.goFirst(); // Hacia adelante

        while (eie.hasNext()) {
            e = eie.next();
            System.out.println(e.toString());
        }
    }
}

```

```

        System.out.println("_____");
        System.out.println("IR AL ultimo");
        eie.goLast(); // Posicion de la lista es la ultima
        e = events.get(events.size()-1);
        System.out.println(e.toString());

        System.out.println("_____");
        System.out.println("IR AL PRIMERO");
        eie.goFirst(); // Posicion de la lista es la primera
        e = events.get(0);
        System.out.println(e.toString());
    }
    catch(ParseException e1) {
        System.out.println("¿Que pasa guapo? ¿Tienes problemas con la fecha?" +
date);
    }

}
}

```

La clase main que usamos es muy parecida a la que nos daba de ejemplo en el enunciado, ya que hemos visto que es de la manera más efectiva de demostrar el funcionamiento de el patrón que hemos implementado, pero no es exactamente igual ya que hemos incluido también la visualización de el último evento y el primero, a la vez que tenido que actualizar el sistema de fechas y añadimos al dataAccess para alguno de los usos que queríamos mostrar.

Maximo cabe recalcar que a veces da problemas la base de datos ya que dice que se repiten las claves primarias para equipos, pero eso no es algo que nosotros hayamos modificado.

Muestra de ejecucion:



```
// TODO Auto-generated catch block
// TODO Auto-generated catch block

<terminated> Main [Java Application] C:\Users\Admin\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_14.0.2.v20200815-0932\jre\bin\javaw.exe (9 no
DataBase closed
Esto es el extended iterator
Opening DataAccess instance => isDatabaseLocal: true getDatabBaseOpenMode: initialize
>> DataAccess: getEvents
1;Atletico-Athletic
2;Eibar-Barcelona
3;Getafe-Celta
4;Alaves-Deportivo
5;Espanol-Villareal
6;Las Palmas-Sevilla
7;Malaga-Valencia
8;Girona-Leganes
9;Real Sociedad-Levante
10;Betis-Real Madrid
22;LA Lakers-Phoenix Suns
23;Atlanta Hawks-Houston Rockets
24;Miami Heat-Chicago Bulls
27;Djokovic-Federer
DataBase closed
Esto es el getEvents normal
Opening DataAccess instance => isDatabaseLocal: true getDatabBaseOpenMode: initialize
>> DataAccess: getEvents
1;Atletico-Athletic
2;Eibar-Barcelona
3;Getafe-Celta
4;Alaves-Deportivo
5;Espanol-Villareal
6;Las Palmas-Sevilla
7;Malaga-Valencia
8;Girona-Leganes
9;Real Sociedad-Levante

} catch (MalformedURLException e2) {
// TODO Auto-generated catch block
}

<terminated> Main [Java Application] C:\Users\Admin\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_14.0.2.v20200815-0932\jre\bin\javaw.exe (9 nov. 2023 17:05:15 - 17:05:17)

RECORRIDO HACIA ATRÁS
27;Djokovic-Federer
24;Miami Heat-Chicago Bulls
23;Atlanta Hawks-Houston Rockets
22;LA Lakers-Phoenix Suns
10;Betis-Real Madrid
9;Real Sociedad-Levante
8;Girona-Leganes
7;Malaga-Valencia
6;Las Palmas-Sevilla
5;Espanol-Villareal
4;Alaves-Deportivo
3;Getafe-Celta
2;Eibar-Barcelona
1;Atletico-Athletic

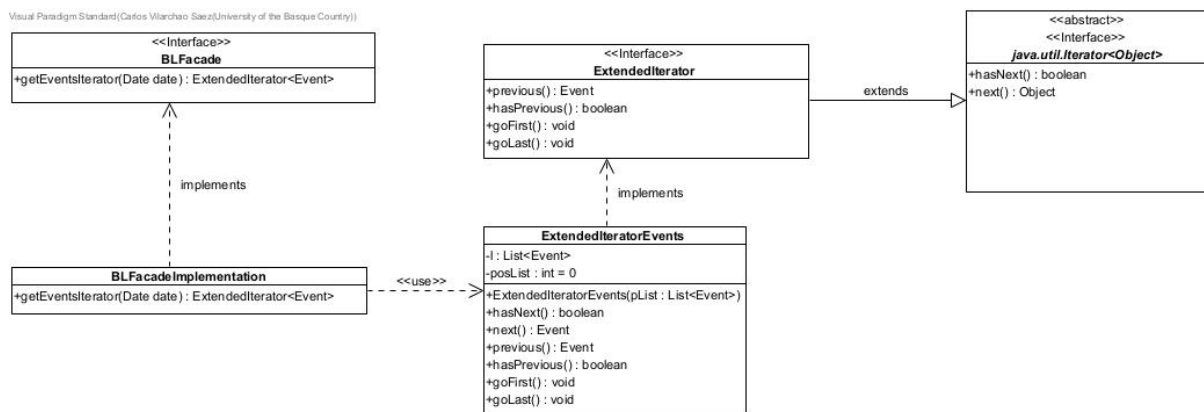
RECORRIDO HACIA ADELANTE
1;Atletico-Athletic
2;Eibar-Barcelona
3;Getafe-Celta
4;Alaves-Deportivo
5;Espanol-Villareal
6;Las Palmas-Sevilla

IR AL ultimo
27;Djokovic-Federer

IR AL PRIMERO
1;Atletico-Athletic
```

UML

Visual Paradigm Standard (Carlos Vilarcho Saez/University of the Basque Country)



3. Patrón Adapter

Creamos la clase WindowTable que muestra la información en una JTable de todas las apuestas realizadas por un usuario:

```
package domain;
```

```
import java.awt.BorderLayout;
import java.awt.Dimension;
```

```
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;
```

```
public class WindowTable extends JFrame{
    private Registered user;
    private JTable tabla;
    public WindowTable(Registered user){
        super("Apuestas realizadas por "+ user.getUsername()+":");
        this.setBounds(100, 100, 700, 200);
        this.user = user;
        UserAdapter adapt = new UserAdapter(user);
        tabla = new JTable(adapt);
        tabla.setPreferredScrollableViewportSize(new Dimension(500, 70));
        //Creamos un JScrollPane y le agregamos la JTable
        JScrollPane scrollPane = new JScrollPane(tabla);
        //Agregamos el JScrollPane al contenedor
        getContentPane().add(scrollPane, BorderLayout.CENTER);
    }
}
```

Creamos la clase adapter UserAdapter:

```
package domain;
```

```
import javax.swing.table.AbstractTableModel;
```

```
public class UserAdapter extends AbstractTableModel{
    //ERAIKITZAILEA
    private Registered u;
    private String[][] matriz;
    private String[] nombreColumnas = { "Evento", "Pregunta", "Fecha Evento", "Apuesta (€)"
};
```

```
    public UserAdapter(Registered user) {
        this.u=user;
        matriz = new String[getRowCount()][getColumnCount()];
        int i = 0;
```

```
        for(ApustuAnitza apusAn : user.getApustuAnitzak()) {
            for(Apustua apus: apusAn.getApustuak()) {
                matriz[i][0] = apus.getKuota().getQuestion().getEvent().getDescription();
                matriz[i][1] = apus.getKuota().getQuestion().getQuestion();
                matriz[i][2] = apusAn.getData().toString();
                matriz[i][3] = apusAn.getBalioa().toString();
                i++;
            }
        }
    }
```

```
    public int getRowCount() {
        int result = 0;
        for(ApustuAnitza apusAn : u.getApustuAnitzak()) {
            for(Apustua apus: apusAn.getApustuak()) {
                result++;
            }
        }
        return result;
    }
```

```
    public int getColumnCount() {
        return 4;
    }
```

```
    public Object getValueAt(int rowIndex, int columnIndex) {
        return matriz[rowIndex][columnIndex];
    }
```

```

@Override
public String getColumnName(int column) {
    return nombreColumnas[column];
}
}

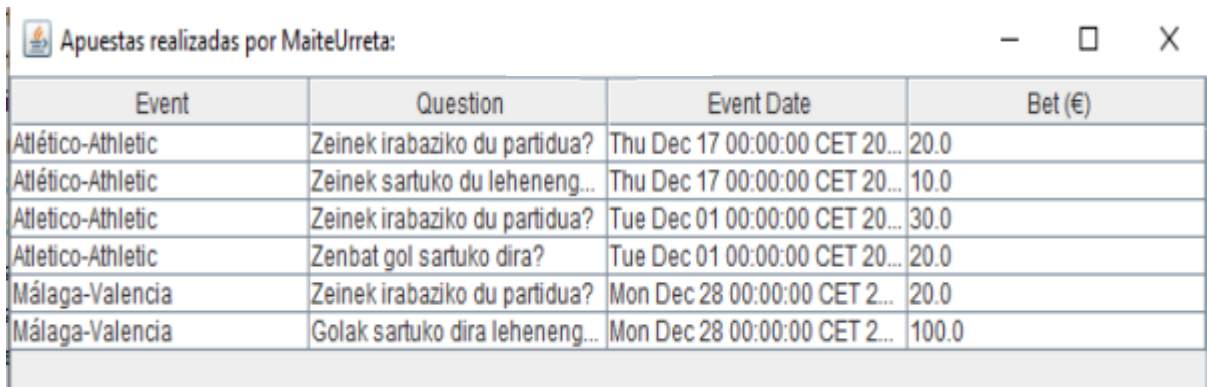
```

Los cambios en el main serán los siguientes, esto debería de generar la ventana. El nombre de usuario debería de ser de un usuario que exista:

```

try {
    Registered maiteUrreta= new Registered("MaiteUrreta", "123", 1);
    bIFacade= new BLFacadeImplementation(); //MainTable soilik
    Registered user = bIFacade.findUser(maiteUrreta);
    WindowTable vt = new WindowTable(user);
    vt.setVisible(true);
} catch (Exception e) {
    e.printStackTrace();
}

```



Event	Question	Event Date	Bet (€)
Atlético-Athletic	Zeinek irabaziko du partidua?	Thu Dec 17 00:00:00 CET 20...	20.0
Atlético-Athletic	Zeinek sartuko du leheneng...	Thu Dec 17 00:00:00 CET 20...	10.0
Atletico-Athletic	Zeinek irabaziko du partidua?	Tue Dec 01 00:00:00 CET 20...	30.0
Atletico-Athletic	Zenbat gol sartuko dira?	Tue Dec 01 00:00:00 CET 20...	20.0
Málaga-Valencia	Zeinek irabaziko du partidua?	Mon Dec 28 00:00:00 CET 2...	20.0
Málaga-Valencia	Golak sartuko dira leheneng...	Mon Dec 28 00:00:00 CET 2...	100.0

