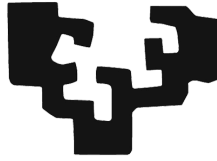


eman ta zabal zazu



Universidad
del País Vasco

Euskal Herriko
Unibertsitatea

IZENBURUA

Diseinu patroiak Rides proiektuan

INFORMATIKA INGENIARITZA GRADUA

Egileak:

KEPA GAZTAÑAGA

MIRIAM RODRIGUEZ

2024-ko azaroak 5a

gitHubeko Repositorio esteka:

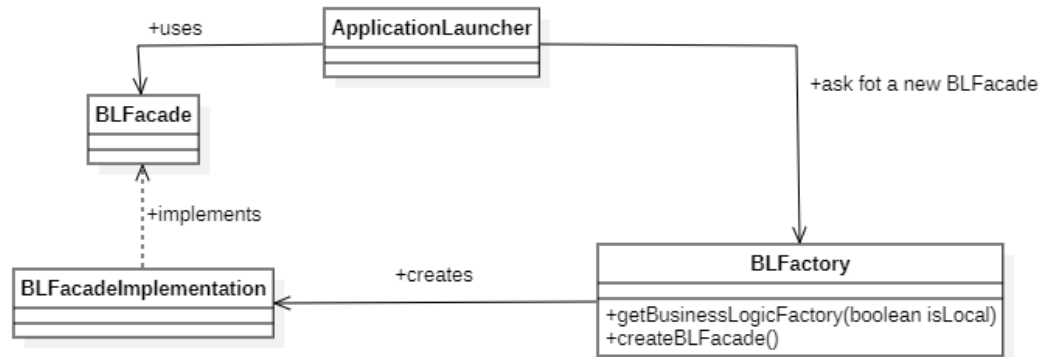
<https://github.com/Kepa8/Rides24-Miriam-Kepa-.git>

AURKIBIDEA

AURKIBIDEA	2
1. Factory Method Patroia	3
1.1. UML diagrama hedatua	3
1.2. Kode azpimagarria	3
2. Iterator patroia	5
2.1. UML diagrama hedatua	5
2.2. Kode azpimagarria	5
2.3. Exekuzioa	6
3. Adapter Patroia	7
3.1. UML diagrama hedatua	7
3.2. Kode azpimagarria	7
3.3. Exekuzioa	9

1. Factory Method Patroia

1.1. UML diagrama hedatua



1.2. Kode azpimagarria

- ApplicationLauncher klasean:

BLFactory klasean aipatu den bezala, negozio logika aukeratzeko duen kode zatia komentatu da, orain BLFactory klaseak egiten baitu.

```
public class ApplicationLauncher {

    public static void main(String[] args) {
        ConfigXML c = ConfigXML.getInstance();
        System.out.println(c.getLocale());
        Locale.setDefault(new Locale(c.getLocale()));
        System.out.println("Locale: " + Locale.getDefault());
        try {
            BLFactory factory = new BLFactory(c);
/*
UIManager.setLookAndFeel("javax.swing.plaf.metal.MetalLookAndFeel");
            if (c.isBusinessLogicLocal()) {
                DataAccess da = new DataAccess();
                appFacadeInterface = new
BLFacadeImplementation(da);

            }
            else { // If remote
                String serviceName = "http://" +
c.getBusinessLogicNode() + ":" + c.getBusinessLogicPort() + "/ws/"
+ c.getBusinessLogicName() +
"?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/", "BLFacadeImplementationService");

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

```

                                appFacadeInterface =
service.getPort(BLFacade.class);
                                }
*/
                                BLFacade appFacadeInterface = factory.createBLFacade();
                                MainGUI.setBusinessLogic(appFacadeInterface);
                                MainGUI a = new MainGUI();
                                a.setVisible(true);
                                } catch (Exception e) {
                                    // a.jLabelSelectOption.setText("Error: "+e.toString());
                                    // a.jLabelSelectOption.setForeground(Color.RED);
                                    System.out.println("Error in ApplicationLauncher: " +
e.toString());
                                }
                                // a.pack();
                                }
}

```

- BLFactory klasean:

ApplicationLauncher klasetik komentatu dugun kode zatia, hau da, aldagaiari zein negozio logika objektua (lokala edo urrunekoa) erabili behar den esleitzen dion zatia implementatu da sortu den BLFactory klasean. Modu honetan, BLFactory arduratzen da negozio logika aukeratzeaz eta ez ApplicationLauncher.

```

public class BLFactory {
    private ConfigXML config;
    private BLFacade businessLogicLocal;

    public BLFactory(ConfigXML config) {
        this.config = config;
    }
    public BLFacade createBLFacade() {
        if (config.isBusinessLogicLocal()) {
            DataAccess da = new DataAccess();
            da.initializeDB();
            return new BLFacadeImplementation(da);
        } else {
            String serviceName = "http://" + config.getBusinessLogicNode() + ":" +
config.getBusinessLogicPort() + "/ws/" + config.getBusinessLogicName() + "?wsdl";
            try {
                URL url = new URL(serviceName);
                QName qname = new QName("http://businessLogic/",
"BLFacadeImplementationService");
                Service service = Service.create(url, qname);
                return service.getPort(BLFacade.class);
            } catch (Exception e) {
                throw new RuntimeException("Error creating remote BLFacade: " +
e.getMessage(), e);
            }
        }
    }
    public BLFacade getBusinessLogicFactory(boolean isLocal) {

```

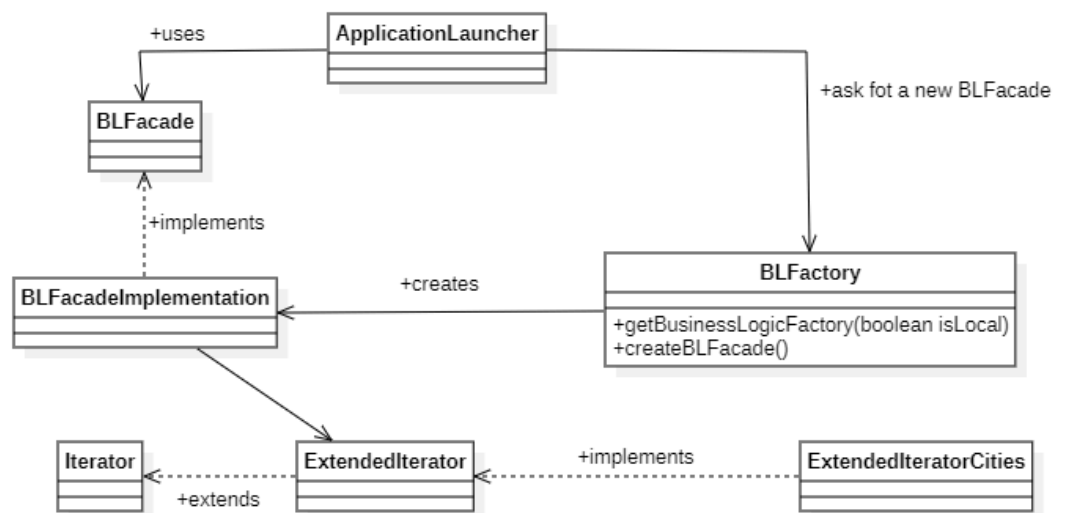
```

    if (isLocal) {
        if (businessLogicLocal == null) {
            businessLogicLocal = createBLFacade();
        }
        return businessLogicLocal;
    } else {
        return createBLFacade();
    }
}
}

```

2. Iterator patroia

2.1. UML diagrama hedatua



2.2. Kode azpimagarria

- ExtendedIterator interfazea:

Interfaze honek iterator interfazea zabaltzen du. Eta iteratorren hasNext() eta next() funtzioez gain ondorengo funtzio hauek implementatzen ditu: previous(), hasPrevious(), goFirst(), public void goFirst(), goLast().

```

public interface ExtendedIterator<Object> extends Iterator<Object>{
    public Object previous();
    public boolean hasPrevious();
    public void goFirst();
    public void goLast();
}

```

- Extended ExtendedIteratorCities klasea:

Klase honek ExtendedIterator interfazea implementatuko du, honela zerrenda batean, kasu honetan hirien zerrenda batean iteratzea ahalbidetuko degu.

```

public class ExtendedIteratorCities<T> implements ExtendedIterator<T> {

```

```

private List<T> list;
private int position;
public ExtendedIteratorCities(List<T> list) {
    this.list = list;
    this.position = 0;
}
@Override
public boolean hasNext() {
    return position < list.size();
}
@Override
public T next() {
    return list.get(position++);
}
@Override
public T previous() {
    return list.get(--position);
}
@Override
public boolean hasPrevious() {
    return position > 0;
}
@Override
public void goFirst() {
    position = 0;
}
@Override
public void goLast() {
    position = list.size() - 1;
}
}

```

2.3. Exekuzioa

The screenshot shows an IDE console window with the following output:

```

<terminated> Main [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (9 nov 2024, 11:46:24 - 11:46:25) [pid: 9360]
Read from config.xml:    businessLogicLocal=true    databaseLocal=true    dataBaseInitialized=true
nov 09, 2024 11:46:25 A. M. dataAccess.DataAccess <init>
INFO: Main file and temporary file deleted successfully.
nov 09, 2024 11:46:25 A. M. dataAccess.DataAccess initializeDB
INFO: Db initialized
Creating BLFacadeImplementation instance with DataAccess parameter
nov 09, 2024 11:46:25 A. M. dataAccess.DataAccess open
INFO: DataAccess opened => isDatabaseLocal: true
nov 09, 2024 11:46:25 A. M. dataAccess.DataAccess close
INFO: DataAccess closed

```

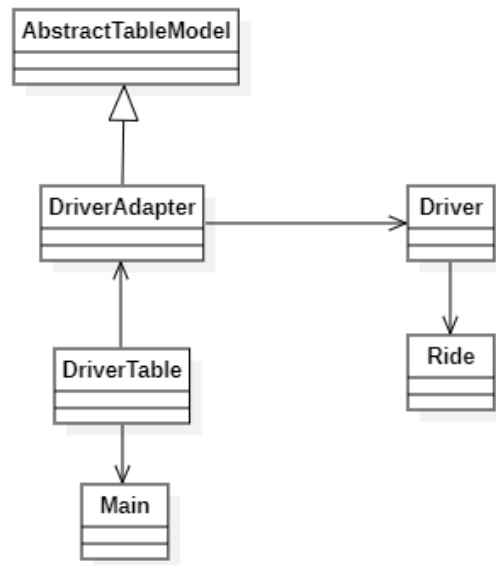
Below the log output, there are two tables:

FROM	LAST	TO	FIRST
Irun			
Donostia			
Barcelona			

FROM	FIRST	TO	LAST
Barcelona			
Donostia			
Irun			
Madrid			

3. Adapter Patroia

3.1. UML diagrama hedatua



3.2. Kode azpimagarria

- DriverAdapter klasea:

Klase honek Driver klaseko objektuak JTable batean erakusteko formatu egokian jartzeko balio du.

Horretarako `getRowCount()`, `getColumnCount()`, `getValueAt()` eta `getColumnName()` funtzioak implementatu dira.

```
public class DriverAdapter extends AbstractTableModel {
    private Driver driver;
    private List<Ride> rides;
    public DriverAdapter(Driver driver) {
        this.driver = driver;
        this.rides = driver.getCreatedRides();
    }
    @Override
    public int getRowCount() {
        return rides.size();
    }
    @Override
    public int getColumnCount() {
        return 5;
    }
    @Override
    public String getColumnName(int columnIndex) {
        switch (columnIndex) {
            case 0:
                return "Origen";
            case 1:
                return "Destino";
            case 2:
```

```

        return "Fecha";
    case 3:
        return "Plazas";
    case 4:
        return "Precio";
    default:
        return "";
    }
}
@Override
public Object getValueAt(int rowIndex, int columnIndex) {
    Ride ride = rides.get(rowIndex);
    switch (columnIndex) {
        case 0:
            return ride.getFrom();
        case 1:
            return ride.getTo();
        case 2:
            return ride.getDate();
        case 3:
            return ride.getnPlaces();
        case 4:
            return ride.getPrice();
        default:
            return null;
    }
}
}
}

```

- DriverTable klasea:

Leiho bat (JFrame) sortzen du Driver batekin, eta bere bidaiak JTable batean erakusten ditu.

```

public class DriverTable extends JFrame {
    private Driver driver;
    private JTable tabla;

    public DriverTable(Driver driver) {
        super(driver.getUsername() + "'s Rides");
        this.setBounds(100, 100, 700, 300);
        this.driver = driver;
        DriverAdapter adapt = new DriverAdapter(driver);
        tabla = new JTable(adapt);
        tabla.setPreferredScrollableViewportSize(new Dimension(500, 70));
        JTableHeader header = tabla.getTableHeader();
        header.setReorderingAllowed(false);
        JScrollPane scrollPane = new JScrollPane(tabla);
        getContentPane().add(scrollPane, BorderLayout.CENTER);
    }
}

```

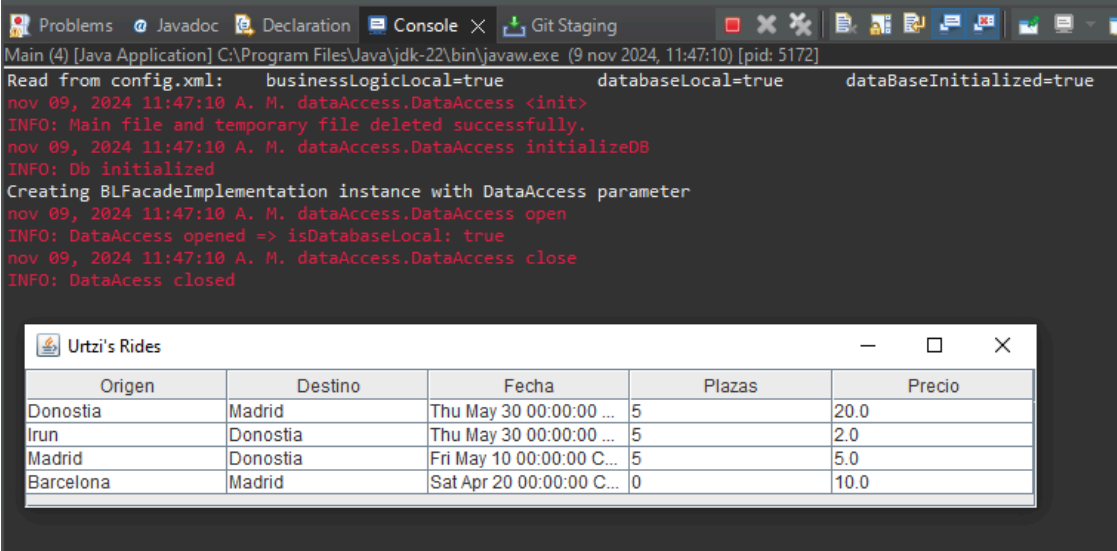
- Main klasea:


```

public class Main {
    public static void main(String[] args) {
        ConfigXML co = ConfigXML.getInstance();
        boolean isLocal = true;
        BLFacade blFacade = new
BLFactory(co).getBusinessLogicFactory(isLocal);
        Driver driver = blFacade.getDriver("Urtzi");
        DriverTable driverTable = new DriverTable(driver);
        driverTable.setVisible(true);
    }
}

```

3.3. Exekuzioa



The screenshot shows a Java IDE with the following console output:

```

Main (4) [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (9 nov 2024, 11:47:10) [pid: 5172]
Read from config.xml:  businessLogicLocal=true      databaseLocal=true      dataBaseInitialized=true
nov 09, 2024 11:47:10 A. M. dataAccess.DataAccess <init>
INFO: Main file and temporary file deleted successfully.
nov 09, 2024 11:47:10 A. M. dataAccess.DataAccess initializeDB
INFO: Db initialized
Creating BLFacadeImplementation instance with DataAccess parameter
nov 09, 2024 11:47:10 A. M. dataAccess.DataAccess open
INFO: DataAccess opened => isDatabaseLocal: true
nov 09, 2024 11:47:10 A. M. dataAccess.DataAccess close
INFO: DataAccess closed

```

Below the console, a window titled "Urtzi's Rides" displays a table with the following data:

Origen	Destino	Fecha	Plazas	Precio
Donostia	Madrid	Thu May 30 00:00:00 ...	5	20.0
Irun	Donostia	Thu May 30 00:00:00 ...	5	2.0
Madrid	Donostia	Fri May 10 00:00:00 C...	5	5.0
Barcelona	Madrid	Sat Apr 20 00:00:00 C...	0	10.0