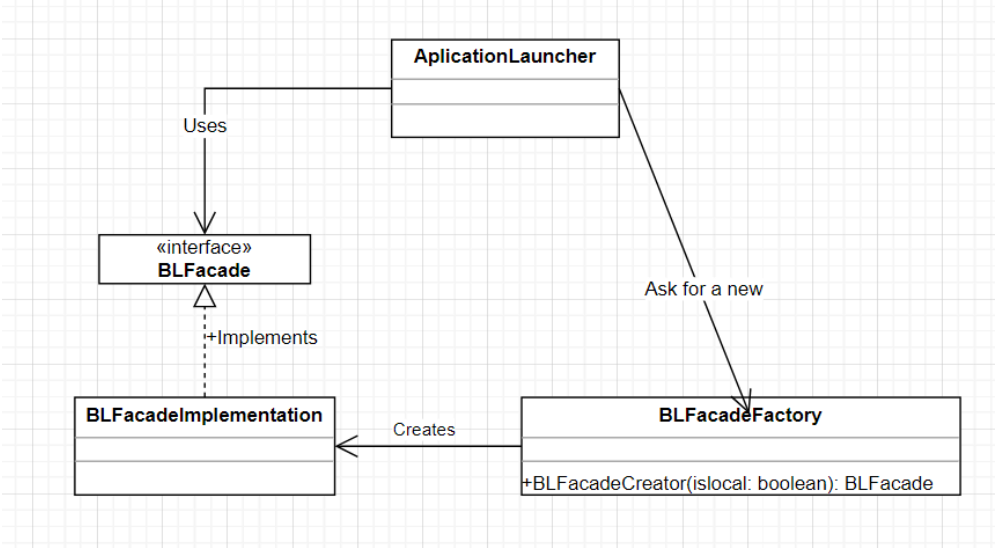
RIDES PROJECT PATTERNS EXTENSIONS

Egilea:Alaitz Shan Ye

1. Factory Patroia



- Aldaketak:

1. BLFacadeFactorysortu, zeinetan beste BLFacadeimplementation deitzen dioenean, BLFacade berri bat sortuko du, kasu honetan boolear bat pasatuz, lokala edo urrutikoa sortuko du.

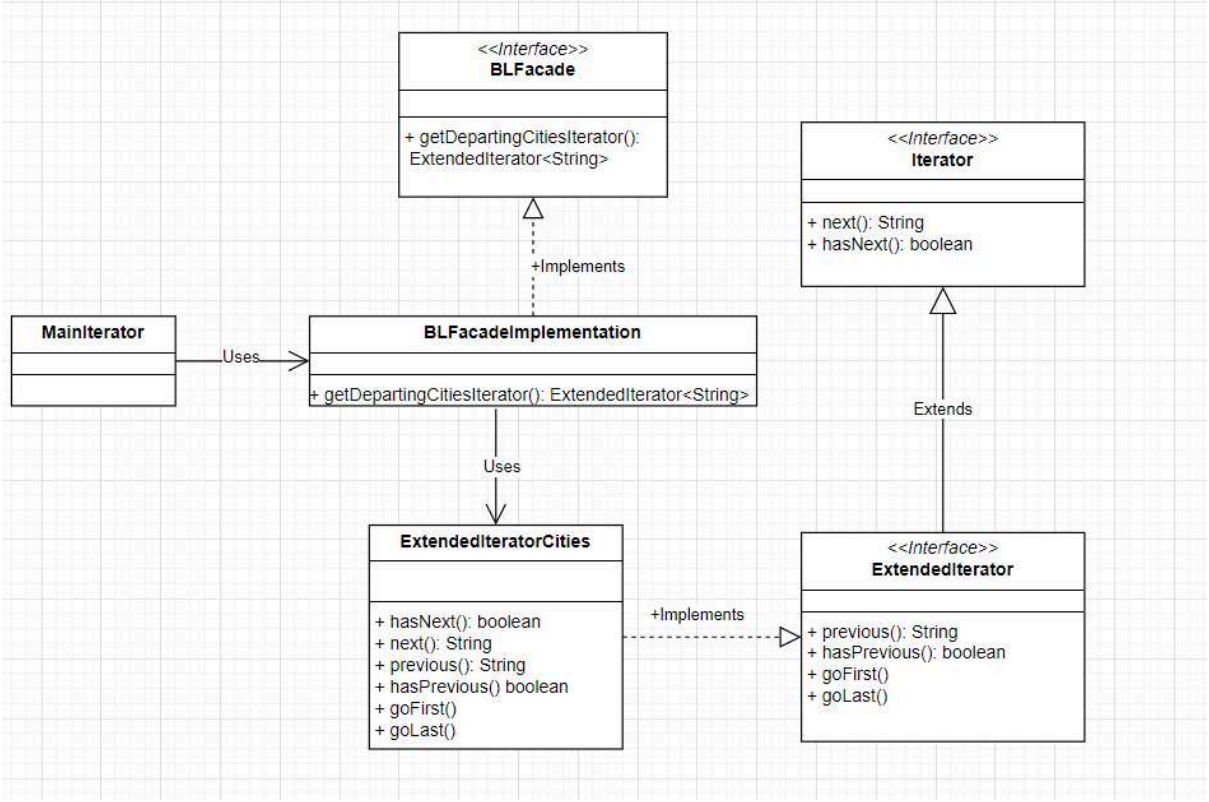
BLFacadeFactory klasean:

```
public BLFacade BLFacadeCreator(boolean isLocal) {

    ConfigXML c = ConfigXML.getInstance();
    try {
        BLFacade appFacadeInterface;
        if(isLocal) {
            DataAccess da = new DataAccess();
            appFacadeInterface = new BLFacadeImplementation(da);
        } else {
            String serviceName = "http://" + c.getBusinessLogicNode() + ":" + c.getBusinessLogicPort() + "/ws/"
                    + c.getBusinessLogicName() + "?wsdl";

            URL url = new URL(serviceName);

            QName qname = new QName("http://businessLogic/", "BLFacadeImplementationService");

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

            appFacadeInterface = service.getPort(BLFacade.class);
        }
        return appFacadeInterface;
    } catch(Exception e) {
        System.out.println("Error in ApplicationLauncher: " + e.toString());
        return null;
    }
}
```

Aplication Launcher klasean:

```
BLFacadeFactory factory = new BLFacadeFactory();
appFacadeInterface = factory.BLFacadeCreator(c.isBusinessLogicLocal());
```

2. Iterator Patroia



- Aldaketak:

1. ExtendedIterator interfazea sortu, Iterator inferfazea implementatzen duena.
2. ExtendedIteratorCities klasea sortu ExtendedIterator implementatzen duena.
3. BLFacade interfazean eta BLFacadeimplementation klasean getDepartingCitiesIterator() metodoa implementatu.

```java
public class ExtendedIteratorCities implements ExtendedIterator<String> {
    private List<String> citiesList;
    private int pos;

    public ExtendedIteratorCities(List<String> cities) {
        citiesList = cities;
        pos = 0;
    }

    @Override
    public boolean hasNext() {
        if(pos <= citiesList.size() - 1) return true;
        else return false;
    }

    @Override
    public String next() {
        String ret = citiesList.get(pos);
        this.pos++;
        return ret;
    }

    @Override
    public String previous() {
        String ret = citiesList.get(pos - 1);
        this.pos--;
        return ret;
    }

    @Override
    public boolean hasPrevious() {
        if(pos > 0) return true;
        else return false;
    }

    @Override
    public void goFirst() {
        this.pos = 0;
    }

    @Override
    public void goLast() {
        this.pos = citiesList.size();
    }
}
```

BLFacade interfazean:

```java
public ExtendedIterator<String> getDepartingCitiesIterator();
```

BLFacadeImplementation klasean:

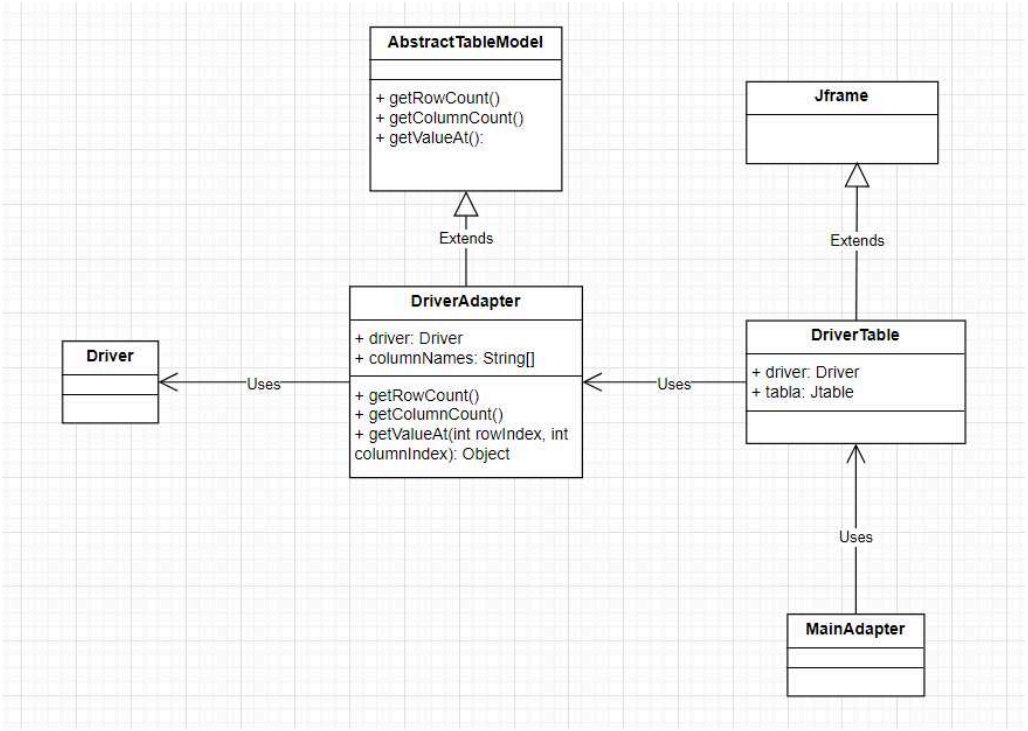
```java
@Override
public ExtendedIterator<String> getDepartingCitiesIterator() {
    ExtendedIterator<String> i = new ExtendedIteratorCities(this.getDepartCities());
    return i;
}
```

```java
public class MainIterator {

    public static void main(String[]    args)   {
        //      the BL  is  local
        boolean isLocal =   true;
        BLFacade    blFacade =  new BLFacadeFactory().BLFacadeCreator(isLocal);
        ExtendedIterator<String>    i = blFacade.getDepartingCitiesIterator();
        String c;
        System.out.println("_____");
        System.out.println("FROM    LAST    TO  FIRST");
        i.goLast(); //  Go  to  last    element
        while (i.hasPrevious()) {
            c = i.previous();
            System.out.println(c);
        }
        System.out.println();
        System.out.println("_____");
        System.out.println("FROM    FIRST   TO  LAST");
        i.goFirst();    //  Go  to  first   element
        while (i.hasNext()) {
            c = i.next();
            System.out.println(c);
        }
    }

}
```

Emaitza:

```
Console
<terminated> MainIterator [Java Application] C:\Users\Alaitz19\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin
Read from config.xml:    businessLogicLocal=true        databaseLocal=true      dataBaseInitialized=true
File deleted
DataAccess opened => isDatabaseLocal: true
Db initialized
DataAccess created => isDatabaseLocal: true isDatabaseInitialized: true
DataAcess closed
nov 09, 2024 12:23:45 P. M. businessLogic.BLFacadeImplementation <init>
INFORMACIÓN: Creating BLFacadeImplementation instance with DataAccess parameter
DataAccess opened => isDatabaseLocal: true
DataAcess closed
_____
FROM    LAST    TO      FIRST
Madrid
Irun
Donostia
Barcelona

_____
FROM    FIRST   TO      LAST
Barcelona
Donostia
Irun
Madrid
```

3. Adapter Patroia



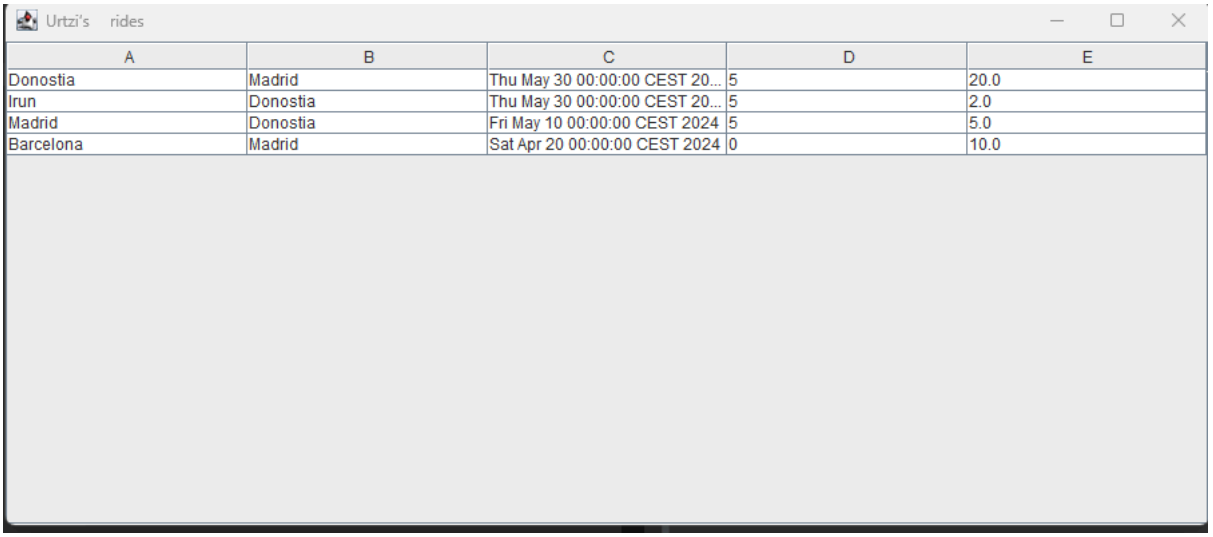
- Aldaketak:
    1. DriverTable klasea sortu, DriverAdapter erabiliko duena.
    2. DriverAdapter klasea sortu, AbstractTableModel klasea heredatzen duena
    3. MainAdapter klasea sortu proba egiteko.

```java
public class DriverAdapter extends AbstractTableModel{


    protected Driver driver;
     protected String[] columnNames =
                new String[] {"From", "To", "Date", "Places", "Price" };
     public DriverAdapter(Driver d) {
         super();
         this.driver=d;
     }
    @Override
    public int getRowCount() {
         return driver.getCreatedRides().size();
    }

    @Override
    public int getColumnCount() {
         return columnNames.length;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
         Iterator<Ride> iterator = driver.getCreatedRides().iterator();
          Ride currentRide = null;

          for(int i = 0; i <= rowIndex; i++) {
              if(iterator.hasNext()) {
                  currentRide = iterator.next();
              } else {
                  return null;
              }
          }
          if(currentRide == null) {
              return null;
          }
          switch(columnIndex) {
            case 0:
                return currentRide.getFrom();
            case 1:
                return currentRide.getTo();
            case 2:
                return currentRide.getDate();
            case 3:
                return currentRide.getnPlaces();
            case 4:
                return currentRide.getPrice();
            default:
                return null;
          }
    }
}
```

Kode honetan Driver (gidari) baten ibilaldiak (Ride objetuak) Java Swing-eko taula batean bistaratzea ahalbidetzen du. Taulak bost zutabe ditu: "From", "To", "Date", "Places", eta "Price", eta zutabe bakoitzean Ride objektuaren datu jakin bat erakusten da. getRowCount() metodoak errenkada kopurua (ibilaldi kopurua) itzultzen du, getColumnCount()-ek zutabe kopurua, eta getValueAt() metodoak errenkada eta zutabe jakin batean dagoen balioa itzultzen du.

```
package businessLogic;

import domain.Driver;

public class MainAdapter {
    public static void main(String[] args) {
        //the   BL  is  local
        boolean isLocal = true;
        BLFacade blFacade = new BLFacadeFactory().BLFacadeCreator(isLocal);
        Driver d = blFacade.getDriver("Urtzi");
        DriverTable dt = new DriverTable(d);
        dt.setVisible(true);
    }

}
```

Emaitza:



A	B	C	D	E
Donostia	Madrid	Thu May 30 00:00:00 CEST 20...	5	20.0
Irun	Donostia	Thu May 30 00:00:00 CEST 20...	5	2.0
Madrid	Donostia	Fri May 10 00:00:00 CEST 2024	5	5.0
Barcelona	Madrid	Sat Apr 20 00:00:00 CEST 2024	0	10.0

4. Github repoa

https://github.com/Alaitz19/Rides24Complete