STPT: XML Technologies Project

Alexandru Munteanu
Maria Vonica
Zouel Fikar Jahjah

January 27, 2021

Contents

Chapter 1

Overview of the project in regards to requirements

1.1 Implementation of the requirements

What follows is a short description of how and where the various requirements for the project have been implemented.

1.1.1 XML database

The database models a public transport system, representing three main components: Vehicles, transport stations, and timetables. The timetables data was colleted with the help of a Python script using the BeautifulSoup and requests libraries from the STPT raidfleet web service[1].

The XML database is split into three files: **vehicles.xml**, **timetables.xml**, respectively **statii-ratt.xml**. XSD schemas have been created for all three databases in files: **vehicles.xsd**, **timetables.xsd**, and **statii-ratt.xsd**.

1.1.2 Parsing

For the parsing of the documents we have used JAXB and created Java POJO's[2] with JAXB bindings. This offers support later to be able to use Apache Camel's **.type()** with **.bindingMode(RestBindingMode.xml)** in order to automatically marshall a request's XML body into a POJO. The classes are available in the **main.java.models** subpackage of our project.

1.1.3 XPath support

Support for parsing and XPath can be found in the **main.java.core** subpackage, the result of the JAXB parsing is marshalled into a DOM document, for which, later on **javax.xml.xpath.XPathFactory** is used in order to perform XPath queries.

[1] http://86.125.113.218:61978/html/timpi/ratt.php
[2] Plain Old Java Objects

1.1.4 Basic services

For the basic services, CRUD[3] operations we're implemented for all the three models on top of the parsing and XPath facilities.

1.1.5 REST API

The CRUD operations defined in the previous subsection have been exposed using Apache Camel's **.bean()** mechanism using **camel-rest** and **camel-netty** to be able to serve HTTP requests.

1.1.6 Web Service

The web service is also exposed through Apache Camel, implementing more useful computations baesd on the XPath support, for querying informations such as what vehicles will pass to a given station, which station is closest in terms of location, when the first/last vehicles depart from a given station.

1.1.7 Testing the REST API and web service

For testing the functionalities that the REST API and the web service expose through Camel, we have used Postman, which is a highly versatile platform for developing API's. It allows us to perform requests and test responses for both our various services.

1.1.8 Frontend

1.2 Division of the tasks among project members

The following table shows how we split the tasks in order to develop the project.

Task	Asignee
Project structuring, dependency management	Maria Vonica
Documentation using JavaDoc syntax	Maria Vonica
Data Gathering	Alexandru Munteanu
Frontend	Zouel Fikar Jahjah
Data cleaning, construction of the XML database	Alexandru Munteanu, Maria Vonica
Parsing using JAXB, creation of models	Alexandru Munteanu
XPath support	Maria Vonica, Alexandru Munteanu
CRUD operations	Alexandru Munteanu
REST application	Alexandru Munteanu
Web service	Maria Vonica, Zouel Fikar Jahjah
REST application testing	Maria Vonica, Zouel Fikar Jahjah
XSD schema definitions	Zouel Fikar Jahjah

[3]Create, Read, Upload, and Delete

Class Hierarchy

Classes

- java.lang.Object
  - core.Interactor (in 3.1, page 12)
    - core.StationsInteractor (in 3.2, page 14)
    - core.TimeTablesInteractor (in 3.3, page 19)
    - core.VehiclesInteractor (in 3.4, page 24)
  - models.Arrival (in 4.1, page 29)
  - models.Direction (in 4.2, page 31)
  - models.StationsWrapper (in 4.3, page 32)
  - models.Time (in 4.4, page 33)
  - models.TimeTable (in 4.5, page 34)
  - models.TimetablesWrapper (in 4.6, page 35)
  - models.TransportStation (in 4.7, page 36)
  - models.Vehicle (in 4.8, page 39)
  - models.VehiclesWrapper (in 4.9, page 40)
  - parsers.ParserUtils (in 2.1, page 8)
  - parsers.XPathUtils (in 2.2, page 10)

Chapter 2

Package parsers

parsers *Package Contents* *Page*

Classes

2.1 Class ParserUtils

parsers.ParserUtils

Class which implements basic parsing methods over an XML document.

2.1.1 Declaration

```
public class ParserUtils
 extends java.lang.Object
```

2.1.2 Field summary

parsers.ParserUtils.path$_{t}o_{d}oc$**path_to_doc**

2.1.3 Constructor summary

parsers.ParserUtils(java.lang.String)**ParserUtils(String)** Constructor of the ParserUtil class.

2.1.4 Method summary

parsers.ParserUtils.parseJAXB()**parseJAXB()** Method which parses an XML document by using JAXB.

parsers.ParserUtils.SaveDoc(org.w3c.dom.Document, java.lang.String)**SaveDoc(Document, String)** Method which, given a document and a location, saves the document at the specific location.

2.1.5 Fields

- parsers.ParserUtils.path$_to_doc$`public java.lang.String` **path_to_doc**

2.1.6 Constructors

- parsers.ParserUtils(java.lang.String)**ParserUtils**

```
public ParserUtils(java.lang.String path_to_doc)
```

  - **Description**
    Constructor of the ParserUtil class.
  - **Parameters**
    * `path_to_doc` – Location of the XML document to be used.

2.1.7 Methods

- parsers.ParserUtils.parseJAXB()**parseJAXB**

```
public org.w3c.dom.Document parseJAXB() throws javax.xml.bind.
    JAXBException, javax.xml.parsers.ParserConfigurationException
```

  - **Description**
    Method which parses an XML document by using JAXB. This is achieved by specifying which classes are to be taken into consideration for JAXB binding, then unmarshalling the XML document into the classes and returning the marshalled document back.
  - **Returns** – Marshalled XML document.
  - **Throws**
    * `javax.xml.bind.JAXBException` – @see JAXBException
    * `javax.xml.parsers.ParserConfigurationException` – @see ParserConfigurationException

- parsers.ParserUtils.SaveDoc(org.w3c.dom.Document, java.lang.String)**SaveDoc**

```
public void SaveDoc(org.w3c.dom.Document doc,java.lang.String
    location) throws javax.xml.transform.TransformerException
```

  - **Description**
    Method which, given a document and a location, saves the document at the specific location.
  - **Parameters**
    * `doc` – Document to be saved.
    * `location` – Location where the document will be saved.
  - **Throws**
    * `javax.xml.transform.TransformerException` – @see TransformerException

2.2 Class XPathUtils

parsers.XPathUtils

Class which implements the XPath operations needed for the application.

2.2.1 Declaration

```
public class XPathUtils
 extends java.lang.Object
```

2.2.2 Field summary

parsers.XPathUtils.doc**doc**

2.2.3 Constructor summary

parsers.XPathUtils(org.w3c.dom.Document)**XPathUtils(Document)** Constructor of the XPathUtils clas.
parsers.XPathUtils(javax.xml.bind.Marshaller, models.StationsWrapper)**XPathUtils(Marshaller, StationsWrapper)**

2.2.4 Method summary

parsers.XPathUtils.printNodes(org.w3c.dom.NodeList)**printNodes(NodeList)**
parsers.XPathUtils.QueryXPath(java.lang.String)**QueryXPath(String)** Method which, given a query in the form of a String object, generates a NodeList of responses using XPath.
parsers.XPathUtils.QueryXPathString(java.lang.String)**QueryXPathString(String)** Method which, given a query in the form of a String object, generates a ArrayList of responses using XPath.

2.2.5 Fields

- parsers.XPathUtils.doc`public org.w3c.dom.Document` **doc**

2.2.6 Constructors

- parsers.XPathUtils(org.w3c.dom.Document)**XPathUtils**

  ```
  public XPathUtils(org.w3c.dom.Document doc)
  ```

  - **Description**
    Constructor of the XPathUtils clas.
  - **Parameters**
    * `doc` – XML document, used for querying.

- parsers.XPathUtils(javax.xml.bind.Marshaller, models.StationsWrapper)**XPathUtils**

```
public XPathUtils(javax.xml.bind.Marshaller marshaller,models.
    StationsWrapper data)
```

2.2.7 Methods

- parsers.XPathUtils.printNodes(org.w3c.dom.NodeList)**printNodes**

```
public void printNodes(org.w3c.dom.NodeList node_list)
```

- parsers.XPathUtils.QueryXPath(java.lang.String)**QueryXPath**

```
public org.w3c.dom.NodeList QueryXPath(java.lang.String query)
    throws javax.xml.xpath.XPathExpressionException
```

  - **Description**
    Method which, given a query in the form of a String object, generates a NodeList of responses using XPath.
  - **Parameters**
    - `query` – Query which will be used for generating the ArrayList results.
  - **Returns** – NodeList Results of the given query.
  - **Throws**
    - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- parsers.XPathUtils.QueryXPathString(java.lang.String)**QueryXPathString**

```
public java.util.ArrayList QueryXPathString(java.lang.String
    query) throws javax.xml.xpath.XPathExpressionException
```

  - **Description**
    Method which, given a query in the form of a String object, generates a ArrayList of responses using XPath.
  - **Parameters**
    - `query` – Query which will be used for generating the ArrayList results.
  - **Returns** – ArrayList Results of the given query.
  - **Throws**
    - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

Chapter 3

Package core

core *Package Contents* *Page*

Classes

3.1 Class Interactor

core.Interactor

Class which represents the base for the interactors. Through this, one can access the document and pretty print methods.

3.1.1 Declaration

```
public class Interactor
 extends java.lang.Object
```

3.1.2 All known subclasses

TimeTablesInteractor (in 3.3, page 19), StationsInteractor (in 3.2, page 14), VehiclesInteractor (in 3.4, page 24)

3.1.3 Field summary

core.Interactor.document**document**
core.Interactor.putils**putils**
core.Interactor.xputils**xputils**

3.1.4 Constructor summary

core.Interactor(java.lang.String)**Interactor(String)** Constructor of the Interactor class.

3.1.5 Method summary

core.Interactor.getDocument()**getDocument()** Method which returns the parsed XML document.
core.Interactor.prettyPrintNode(org.w3c.dom.Node)**prettyPrintNode(Node)** Method to pretty print a Node element.
core.Interactor.prettyPrintNodeList(org.w3c.dom.NodeList)**prettyPrintNodeList(NodeList)** Method to pretty print the elements of a NodeList argument.
core.Interactor.SaveDocument(java.lang.String)**SaveDocument(String)** Method which saves the XML document.

3.1.6 Fields

- core.Interactor.document`protected org.w3c.dom.Document` **document**
- core.Interactor.xputils`protected parsers.XPathUtils` **xputils**
- core.Interactor.putils`protected parsers.ParserUtils` **putils**

3.1.7 Constructors

- core.Interactor(java.lang.String)**Interactor**

```
public Interactor(java.lang.String path_to_doc) throws javax.xml
    .bind.JAXBException, javax.xml.parsers.
    ParserConfigurationException
```

  - **Description**
    Constructor of the Interactor class.
  - **Parameters**
    * `path_to_doc` – Path to the XML document to be used.
  - **Throws**
    * `javax.xml.bind.JAXBException` – @see JAXBException
    * `javax.xml.parsers.ParserConfigurationException` – @see ParserConfigurationException

3.1.8 Methods

- core.Interactor.getDocument()**getDocument**

```
public org.w3c.dom.Document getDocument()
```

  - **Description**
    Method which returns the parsed XML document.
  - **Returns** – Return the parsed XML document.

- core.Interactor.prettyPrintNode(org.w3c.dom.Node)**prettyPrintNode**

  ```
  public void prettyPrintNode(org.w3c.dom.Node node)
  ```

  - **Description**
    Method to pretty print a Node element.
  - **Parameters**
    * `node` – Node element to be printed.

- core.Interactor.prettyPrintNodeList(org.w3c.dom.NodeList)**prettyPrintNodeList**

  ```
  public void prettyPrintNodeList(org.w3c.dom.NodeList nodeList)
  ```

  - **Description**
    Method to pretty print the elements of a NodeList argument.
  - **Parameters**
    * `nodeList` – A list of Node elements.

- core.Interactor.SaveDocument(java.lang.String)**SaveDocument**

  ```
  public void SaveDocument(java.lang.String location) throws javax
      .xml.transform.TransformerException
  ```

  - **Description**
    Method which saves the XML document.
  - **Parameters**
    * `location` – Location of the updated document.
  - **Throws**
    * `javax.xml.transform.TransformerException` – @see TransformerException

3.2 Class StationsInteractor

core.StationsInteractor

Class which holds the implementation for interacting with a transport-station object. A transport-station element of of the following structure in the XML: `2406 Tv9b Bv Sudului_2 Bulevardul Sudului / Hotel Lido (AEM) Sudului Sudului 45.737211 21.250093 0 dup script 11.12.16. http://maps.google.com/maps?q=Bulevardul%20Sudului%20/%20Hotel%20Lido@45.737211,21.250093 0` Using the StationsInteractor class we can operate on such elements by parsing the XML document and using the XPathUtils class to query, delete, edit and add.

3.2.1 Declaration

```
public class StationsInteractor
 extends core.Interactor
```

3.2.2 Constructor summary

core.StationsInteractor(java.lang.String)**StationsInteractor(String)** Constructor of the StationsInteractor class, which calls the parent class for creating the marshalled XML doc.

3.2.3 Method summary

core.StationsInteractor.createStation(java.lang.Integer, int, java.lang.String, int, java.lang.String, java.lang.String, java.lang.String, java.lang.String, double, double, java.lang.Boolean, java.lang.String, java.lang.String, java.lang.String, java.lang.String)**createStation(Integer, int, String, int, String, String, String, String, double, double, Boolean, String, String, String, String)** Method which is used for creating a new element of the type transport station.

core.StationsInteractor.createStation(models.TransportStation)**createStation(TransportStation)** Method for creating a new transport station which is used by JAXB binding.

core.StationsInteractor.deleteStation(java.lang.Integer)**deleteStation(Integer)** Method for deleting an element of type transport station based on a given id.

core.StationsInteractor.getAllStations()**getAllStations()** Method for querying for all available transport-stations, taken from the parent XML document.

core.StationsInteractor.getStation(java.lang.Integer)**getStation(Integer)** Method for finding a transport-station based on a given id.

core.StationsInteractor.replaceStation(java.lang.Integer, models.TransportStation)**replaceStation(Integer, TransportStation)** Method for replacing an element of type transport station with a new TransportStation, based on a given id.

3.2.4 Constructors

- core.StationsInteractor(java.lang.String)**StationsInteractor**

  ```
  public StationsInteractor(java.lang.String path_to_doc) throws
    javax.xml.parsers.ParserConfigurationException, javax.xml.
    bind.JAXBException
  ```

  - **Description**

    Constructor of the StationsInteractor class, which calls the parent class for creating the marshalled XML doc.

  - **Parameters**

    * `path_to_doc` – Path to the XML document which will be used by the interactor.

  - **Throws**

* `javax.xml.parsers.ParserConfigurationException` – @see ParserConfigurationException
* `javax.xml.bind.JAXBException` – @see JAXBException

3.2.5 Methods

- core.StationsInteractor.createStation(java.lang.Integer, int, java.lang.String, int, java.lang.String, java.lang.String, java.lang.String, java.lang.String, double, double, java.lang.Boolean, java.lang.String, java.lang.String, java.lang.String, java.lang.String)**createStation**

```
public org.w3c.dom.Node createStation(java.lang.Integer new_id,
    int lineID,java.lang.String lineName,int stationID,java.lang.
    String rawStationName,java.lang.String friendlyStationName,
    java.lang.String shortStationName,java.lang.String
    junctionName,double x,double y,java.lang.Boolean is_invalid,
    java.lang.String verif,java.lang.String verif_date,java.lang.
    String gmaps_links,java.lang.String info_comm) throws javax.
    xml.xpath.XPathExpressionException
```

  - **Description**

    Method which is used for creating a new element of the type transport station. This is achieved by using XPath for finding where to place the new transport station element, and creating it based on the passed parameters. After creation, we append the new Element to the parent.

  - **Parameters**

    * `new_id` – Integer: Id of the vehicle to be added. Example: 3306
    * `lineID` – int: Id of the line for the transport station. Example: 1266.
    * `lineName` – String: Name of the line. Example: Tv4.
    * `stationID` – int: id of the station.
    * `rawStationName` – String: Raw name for the station. Example: P-ta Crucii_2.
    * `friendlyStationName` – String: Friendlier version of the raw station name. Example: Piata Crucii (Torontalului)
    * `shortStationName` – String: Shorter version for the station name. Example: P-ta Crucii.
    * `junctionName` – String: Name of the junction. Example: P-ta Crucii.
    * `x` – double: Latitude of the station location.
    * `y` – double: Longitude of the station location.
    * `is_invalid` – Boolean: States whether the station is still in use.
    * `verif` – String: Method of the station is verified.
    * `verif_date` – String: Date of the last verification.
    * `gmaps_links` – String: Link for google maps location.
    * `info_comm` – String: More info.

  - **Returns** – Returns a Node object which represents the newly added transport station element.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.StationsInteractor.createStation(models.TransportStation)**createStation**

```
public org.w3c.dom.Node createStation(models.TransportStation t)
    throws javax.xml.xpath.XPathExpressionException
```

  - **Description**
    Method for creating a new transport station which is used by JAXB binding.
  - **Parameters**
    * `t` – TransportStation: TransportStation element representing the new element to be added.
  - **Returns** – Returns a Node element representing the newly added transport station element.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.StationsInteractor.deleteStation(java.lang.Integer)**deleteStation**

```
public org.w3c.dom.Document deleteStation(java.lang.Integer id)
   throws javax.xml.xpath.XPathExpressionException
```

  - **Description**
    Method for deleting an element of type transport station based on a given id. The querying to find the transport station whose specific id is the requested one is done by using the existent getVehicle(Integer id) method. If the transport station is found, a new transpor station is created with the new requirements and the parent will now replace the old transport station with the new one. If the requested transport station is found, it will be removed from its parent in the XML document.
  - **Parameters**
    * `id` – Integer: id for finding the requested transport station to be deleted.
  - **Returns** – Document: The XML document which has the requested transport station deleted.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionExceptiond

- core.StationsInteractor.getAllStations()**getAllStations**

  ```
  public org.w3c.dom.NodeList getAllStations() throws javax.xml.
      xpath.XPathExpressionException
  ```

  - **Description**
    Method for querying for all available transport-stations, taken from the parent XML document. The querying is done by passing the following xPath expression to the XPathUtils object: "/transport-stations-root/transport-stations/transport-station"
  - **Returns** – NodeList: A list of Nodes representing all the matched elements found by the query.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.StationsInteractor.getStation(java.lang.Integer)**getStation**

  ```
  public org.w3c.dom.Node getStation(java.lang.Integer station_id)
      throws javax.xml.xpath.XPathExpressionException
  ```

  - **Description**
    Method for finding a transport-station based on a given id. The querying is done by passing the searched id in the following xPath expression, and passing the expression to the XPathUtils class: "//transport-station[@id=%s]" The transport station whose id matches the required id will be returned.
  - **Parameters**
    * `station_id` – Integer: Searched transport station id.
  - **Returns** – Node: If the transport station with the requested id has been found, it will be returned.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.StationsInteractor.replaceStation(java.lang.Integer, models.TransportStation)**replaceStation**

  ```
  public org.w3c.dom.Document replaceStation(java.lang.Integer id,
      models.TransportStation t) throws javax.xml.xpath.
      XPathExpressionException
  ```

  - **Description**
    Method for replacing an element of type transport station with a new TransportStation, based on a given id. The querying to find the requested transport station

to be replaced will be done by using the existent getStation(Integer id) method. If the transport station is found, a new transport station is created with the new requirements and the parent will now replace the old transport station with the new one.

- **Parameters**
  - `id` – Integer: id for finding the requested transport station.
  - `t` – TransportStation: Replacement for the old transport station element.
- **Returns** – Document: The XML document which has the requested transport station replaced.
- **Throws**
  - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

3.2.6 Members inherited from class Interactor

`core.Interactor` (in 3.1, page 12)

- `protected` **document**
- `public Document` **getDocument**`()`
- `public void` **prettyPrintNode**`(org.w3c.dom.Node` **node**`)`
- `public void` **prettyPrintNodeList**`(org.w3c.dom.NodeList` **nodeList**`)`
- `protected` **putils**
- `public void` **SaveDocument**`(java.lang.String` **location**`) throws javax.xml.transform.TransformerException`
- `protected` **xputils**

3.3 Class TimeTablesInteractor

core.TimeTablesInteractor

Class which holds the implementation for interacting with a timetable object. A timetable element of of the following structure in the XML: `Gara de Nord 15:39` Using the TimeTablesInteractor class we can operate on such elements by parsing the XML document and using the XPathUtils class to query, delete, edit and add.

3.3.1 Declaration

```
public class TimeTablesInteractor
 extends core.Interactor
```

3.3.2 Constructor summary

core.TimeTablesInteractor(java.lang.String)**TimeTablesInteractor(String)** Constructor of the TimeTablesInteractor class, which calls the parent class for creating the marshalled XML doc.

3.3.3 Method summary

core.TimeTablesInteractor.createArrival(int, java.lang.String, models.Time)**createArrival(int, String, Time)** Method which is used for creating a new element of the type arrival.

core.TimeTablesInteractor.createDirection(java.lang.Integer, java.util.ArrayList)**createDirection(Integer, ArrayList)** Method which is used for creating a new element of the type direction.

core.TimeTablesInteractor.createTimeTable(int, java.util.ArrayList)**createTimeTable(int, ArrayList)** Method which is used for creating a new element of the type timetable.

core.TimeTablesInteractor.createTimeTable(models.TimeTable)**createTimeTable(TimeTable)** Method for creating a new timetable which is used by JAXB binding.

core.TimeTablesInteractor.deleteTimeTable(java.lang.Integer)**deleteTimeTable(Integer)** Method for deleting an element of type timetable based on a given id.

core.TimeTablesInteractor.getAllTimeTables()**getAllTimeTables()** Method for querying for all available timetables, taken from the parent XML document.

core.TimeTablesInteractor.getTimeTable(java.lang.Integer)**getTimeTable(Integer)** Method for finding a timetable based on a given id.

core.TimeTablesInteractor.replaceTimeTable(java.lang.Integer, models.TimeTable)**replaceTimeTable(Integer, TimeTable)** Method for replacing an element of type timetable with a new TimeTable, based on a given id.

3.3.4 Constructors

- core.TimeTablesInteractor(java.lang.String)**TimeTablesInteractor**

```
public TimeTablesInteractor(java.lang.String path_to_doc) throws
    javax.xml.parsers.ParserConfigurationException, javax.xml.
    bind.JAXBException
```

  - **Description**
    Constructor of the TimeTablesInteractor class, which calls the parent class for creating the marshalled XML doc.
  - **Parameters**
    * `path_to_doc` – Path to the XML document which will be used by the interactor.
  - **Throws**
    * `javax.xml.parsers.ParserConfigurationException` – @see ParserConfigurationException
    * `javax.xml.bind.JAXBException` – @see JAXBException

3.3.5 Methods

- core.TimeTablesInteractor.createArrival(int, java.lang.String, models.Time)**createArrival**

```
public org.w3c.dom.Node createArrival(int station_id,java.lang.
    String station_name,models.Time arrives_in)
```

  - **Description**
    Method which is used for creating a new element of the type arrival. This is achieved by creating a new element of type arrival and adding it to the timetable of the searched id.
  - **Parameters**
    * `station_id` – int: id of the station. Example: 4483.
    * `station_name` – String: Name of the station. Example: Gara de Nord.
    * `arrives_in` – Time: Time of arrival. Example: 16:05
  - **Returns** – Returns a Node object which represents the newly added arrival element.

- core.TimeTablesInteractor.createDirection(java.lang.Integer, java.util.ArrayList)**createDirection**

```
public org.w3c.dom.Node createDirection(java.lang.Integer way,
    java.util.ArrayList arrivals)
```

  - **Description**
    Method which is used for creating a new element of the type direction. This is achieved by creating a new element of type direction and adding it to the timetable of the searched id.
  - **Parameters**
    * `way` – Integer: 0 represents coming, 1 represents going.
    * `arrivals` – ArrayList of type Arrival: Elements of the type arrival.
  - **Returns** – Returns a Node object which represents the newly added direction element.

- core.TimeTablesInteractor.createTimeTable(int, java.util.ArrayList)**createTimeTable**

```
public org.w3c.dom.Node createTimeTable(int vehicle_id,java.util
    .ArrayList directions) throws javax.xml.xpath.
    XPathExpressionException
```

  - **Description**
    Method which is used for creating a new element of the type timetable. This is achieved by finding where to add the new timetable element in the XML document, using the following query in the XPathUtils object: `"//timetable[not(@vehicle_id = preceding-sibling::timetable/@id) and not(@vehicle_id =following-sibling::timetable/@vehicle_id)]"` We then create the vehicle id and the directions for that vehicle. We now need to only populate the directions with arrivals.

- **Parameters**
  - `vehicle_id` – Integer: id of the vehicle for which the timetable is created. Example: 1207.
  - `directions` – ArrayList of type Direction: Possible directions for the vehicle.
- **Returns** – Returns a Node object which represents the newly added timetable element.
- **Throws**
  - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.TimeTablesInteractor.createTimeTable(models.TimeTable)**createTimeTable**

```
public org.w3c.dom.Node createTimeTable(models.TimeTable t)
  throws javax.xml.xpath.XPathExpressionException
```

- **Description**
  Method for creating a new timetable which is used by JAXB binding.
- **Parameters**
  - `t` – TimeTable: TimeTable element representing the new element to be added.
- **Returns** – Returns a Node element representing the newly added timetable element.
- **Throws**
  - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.TimeTablesInteractor.deleteTimeTable(java.lang.Integer)**deleteTimeTable**

```
public org.w3c.dom.Document deleteTimeTable(java.lang.Integer id
  ) throws javax.xml.xpath.XPathExpressionException
```

- **Description**
  Method for deleting an element of type timetable based on a given id. The querying to find the timetable whose specific id is the requested one is done by using the existent getTimeTable(Integer id) method. If the timetable is found, a new timetable is created with the new requirements and the parent will now replace the old timetable with the new one. If the requested timetable is found, it will be removed from its parent in the XML document.
- **Parameters**
  - `id` – Integer: id for finding the requested timetable to be deleted.
- **Returns** – Document: The XML document which has the requested timetable deleted.
- **Throws**

    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionExceptiond

- core.TimeTablesInteractor.getAllTimeTables()**getAllTimeTables**

  ```
  public org.w3c.dom.NodeList getAllTimeTables() throws javax.xml.
      xpath.XPathExpressionException
  ```

  - **Description**

    Method for querying for all available timetables, taken from the parent XML document. The querying is done by passing the following xPath expression to the XPathUtils object: "/timetables-root/timetables/timetable"
  - **Returns** – A list of Nodes representing all the matched elements found by the query.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.TimeTablesInteractor.getTimeTable(java.lang.Integer)**getTimeTable**

  ```
  public org.w3c.dom.Node getTimeTable(java.lang.Integer
      vehicle_id) throws javax.xml.xpath.XPathExpressionException
  ```

  - **Description**

    Method for finding a timetable based on a given id. The querying is done by passing the searched id in the following xPath expression, and passing the expression to the XPathUtils class: "//timetable[@vehicle_id=%s]" The timetable whose id matches the required id will be returned.
  - **Parameters**
    * `vehicle_id` – Integer: Searched timetable id.
  - **Returns** – If the timetable with the requested id has been found, it will be returned.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.TimeTablesInteractor.replaceTimeTable(java.lang.Integer, models.TimeTable)**replaceTimeTable**

  ```
  public org.w3c.dom.Document replaceTimeTable(java.lang.Integer
      id, models.TimeTable t) throws javax.xml.xpath.
      XPathExpressionException
  ```

- **Description**
  Method for replacing an element of type timetable with a new TimeTable, based on a given id. The querying is done by searching for the timetable to be updated with the existing method getTimeTable(). If the timetable is found, we create a new timetable from t and we update the parent with the new node.
- **Parameters**
  - `id` – Integer: id for finding the requested timetable.
  - `t` – TimeTable: Replacement for the old timetable element.
- **Returns** – Document: The XML document which has the requested timetable replaced.
- **Throws**
  - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

3.3.6 Members inherited from class Interactor

`core.Interactor` (in 3.1, page 12)

- `protected` **document**
- `public Document` **getDocument**`()`
- `public void` **prettyPrintNode**`(org.w3c.dom.Node` **node**`)`
- `public void` **prettyPrintNodeList**`(org.w3c.dom.NodeList` **nodeList**`)`
- `protected` **putils**
- `public void` **SaveDocument**`(java.lang.String` **location**`) throws javax.xml.transform.TransformerException`
- `protected` **xputils**

3.4 Class VehiclesInteractor

core.VehiclesInteractor

Class which holds the implementation for interacting with a vehicle object. A vehicle element of of the following structure in the XML: `M42 Bus` Using the VehiclesInteractor class we can operate on such elements by parsing the XML document and using the XPathUtils class to query, delete, edit and add.

3.4.1 Declaration

```
public class VehiclesInteractor
 extends core.Interactor
```

3.4.2 Constructor summary

core.VehiclesInteractor(java.lang.String)**VehiclesInteractor(String)** Constructor of the VehiclesInteractor class, which calls the parent class for creating the marshalled XML document.

3.4.3 Method summary

core.VehiclesInteractor.createVehicle(java.lang.Integer, java.lang.String, java.lang.String)**createVehicle(Integer, String, String)** Method which is used for creating a new element of the type vehicle.
core.VehiclesInteractor.createVehicle(models.Vehicle)**createVehicle(Vehicle)** Method for creating a new vehicle which is used by JAXB binding.
core.VehiclesInteractor.deleteVehicle(java.lang.Integer)**deleteVehicle(Integer)** Method for deleting an element of type vehicle based on a given id.
core.VehiclesInteractor.getAllVehicles()**getAllVehicles()** Method for querying for all available vehicles, taken from the parent XML document.
core.VehiclesInteractor.getVehicle(java.lang.Integer)**getVehicle(Integer)** Method for finding a vehicle based on a given id.
core.VehiclesInteractor.replaceVehicle(java.lang.Integer, models.Vehicle)**replaceVehicle(Integer, Vehicle)** Method for replacing an element of type vehicle with a new Vehicle, based on a given id.

3.4.4 Constructors

- core.VehiclesInteractor(java.lang.String)**VehiclesInteractor**

```
public VehiclesInteractor(java.lang.String path_to_doc) throws
    javax.xml.bind.JAXBException, javax.xml.parsers.
    ParserConfigurationException
```

  - **Description**
    Constructor of the VehiclesInteractor class, which calls the parent class for creating the marshalled XML document.
  - **Parameters**
    * `path_to_doc` – Path to the XML document which will be used by the interactor.
  - **Throws**
    * `javax.xml.bind.JAXBException` – @see JAXBException
    * `javax.xml.parsers.ParserConfigurationException` – @see ParserConfigurationException

3.4.5 Methods

- core.VehiclesInteractor.createVehicle(java.lang.Integer, java.lang.String, java.lang.String)**createVehicle**

```
public org.w3c.dom.Node createVehicle(java.lang.Integer new_id,
    java.lang.String vehicleName,java.lang.String vehicleType)
    throws javax.xml.xpath.XPathExpressionException
```

- **Description**
  Method which is used for creating a new element of the type vehicle. This is achieved by using XPath for finding where to place the new vehicle element, and creating it based on the passed parameters. After creation, we append the new Element to the parent.
- **Parameters**
  - `new_id` – Integer: Id of the vehicle to be added. Example: 3306
  - `vehicleName` – String: Name of the vehicle to be added. Example: M42
  - `vehicleType` – String: Type of the vehicle to be added. Example: Bus
- **Returns** – Returns a Node object which represents the newly added vehicle element.
- **Throws**
  - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.VehiclesInteractor.createVehicle(models.Vehicle)**createVehicle**

```
public org.w3c.dom.Node createVehicle(models.Vehicle v) throws
    javax.xml.xpath.XPathExpressionException
```

  - **Description**
    Method for creating a new vehicle which is used by JAXB binding.
  - **Parameters**
    - `v` – Vehicle: Vehicle element representing the new element to be added.
  - **Returns** – Returns a Node element representing the newly added vehicle element.
  - **Throws**
    - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.VehiclesInteractor.deleteVehicle(java.lang.Integer)**deleteVehicle**

```
public org.w3c.dom.Document deleteVehicle(java.lang.Integer id)
    throws javax.xml.xpath.XPathExpressionException
```

  - **Description**
    Method for deleting an element of type vehicle based on a given id. The querying to find the vehicle whose specific id is the requested one is done by passing the following xPath expression to the XPathUtils object: "//vehicle[@id=%s]" If the requested vehicle is found, it will be removed from its parent in the XML document.
  - **Parameters**
    - `id` – Integer: id for finding the requested vehicle.
  - **Returns** – Document: The XML document which has the requested vehicle deleted.

  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.VehiclesInteractor.getAllVehicles()**getAllVehicles**

```
public org.w3c.dom.NodeList getAllVehicles() throws javax.xml.
    xpath.XPathExpressionException
```

  - **Description**

    Method for querying for all available vehicles, taken from the parent XML document. The querying is done by passing the following xPath expression to the XPathUtils object: "/vehicles-root/vehicles/vehicle"
  - **Returns** – NodeList: A list of Nodes representing all the matched elements found by the query.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.VehiclesInteractor.getVehicle(java.lang.Integer)**getVehicle**

```
public org.w3c.dom.Node getVehicle(java.lang.Integer vehicle_id)
    throws javax.xml.xpath.XPathExpressionException
```

  - **Description**

    Method for finding a vehicle based on a given id. The querying is done by passing the searched id in the following xPath expression, and passing the expression to the XPathUtils class: "//vehicle[@id=%s]" The vehicle whose id matches the required id will be returned.
  - **Parameters**
    * `vehicle_id` – Integer: Searched vehicle id.
  - **Returns** – Node: If the vehicle with the requested id has been found, it will be returned.
  - **Throws**
    * `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

- core.VehiclesInteractor.replaceVehicle(java.lang.Integer, models.Vehicle)**replaceVehicle**

```
public org.w3c.dom.Document replaceVehicle(java.lang.Integer id,
    models.Vehicle vehicle) throws javax.xml.xpath.
    XPathExpressionException
```

- **Description**

  Method for replacing an element of type vehicle with a new Vehicle, based on a given id. The querying to find the requested vehicle to be replaced will be done by using the existent getVehicle(Integer id) method. If the vehicle is found, a new vehicle is created with the new requirements and the parent will now replace the old vehicle with the new one.

- **Parameters**
  - `id` – Integer: id for finding the requested vehicle.
  - `vehicle` – Vehicle: Replacement for the old vehicle element.
- **Returns** – Document: The XML document which has the requested vehicle replaced.
- **Throws**
  - `javax.xml.xpath.XPathExpressionException` – @see XPathExpressionException

3.4.6 Members inherited from class Interactor

`core.Interactor` (in 3.1, page 12)

- `protected` **document**
- `public Document` **getDocument**`()`
- `public void` **prettyPrintNode**`(org.w3c.dom.Node` **node**`)`
- `public void` **prettyPrintNodeList**`(org.w3c.dom.NodeList` **nodeList**`)`
- `protected` **putils**
- `public void` **SaveDocument**`(java.lang.String` **location**`) throws javax.xml.transform.TransformerException`
- `protected` **xputils**

Chapter 4

Package models

models *Package Contents* *Page*

Classes

4.1 Class Arrival

models.Arrival

Class which holds the implementation for an XML element of the type arrival. An arrival element represents the time when a transport vehicle arrives at a specific transport station, and it is represented as follows in the XML document: `Regele Carol 15:43`

4.1.1 Declaration

```
public class Arrival
 extends java.lang.Object
```

4.1.2 Field summary

models.Arrival.station**station**
models.Arrival.time**time**

4.1.3 Constructor summary

models.Arrival()**Arrival()**
models.Arrival(models.TransportStation, models.Time)**Arrival(TransportStation, Time)** Constructor for the Arrival class.

4.1.4 Method summary

models.Arrival.toString()**toString()**

4.1.5 Fields

- models.Arrival.station`public TransportStation` **station**
- models.Arrival.time`public Time` **time**

4.1.6 Constructors

- models.Arrival()**Arrival**

  ```
  public Arrival()
  ```

- models.Arrival(models.TransportStation, models.Time)**Arrival**

  ```
  public Arrival(TransportStation station,Time t)
  ```

  - **Description**
    Constructor for the Arrival class.
  - **Parameters**
    - `station` – TransportStation: An object representing the transport-station where a vehicle arrives.
    - `t` – Time: Time of arrival for that vehicle.

4.1.7 Methods

- models.Arrival.toString()**toString**

  ```
  public java.lang.String toString()
  ```

4.2 Class Direction

models.Direction

Class which holds the implementation for an XML document of the type direction. A direction element represents the direction in which a vehicle goes. 1 represents going to, 0 represents coming from. A direction element is represented as follows in the XML document: ... ...

4.2.1 Declaration

```
public class Direction
 extends java.lang.Object
```

4.2.2 Field summary

models.Direction.arrivals**arrivals**
models.Direction.way**way**

4.2.3 Constructor summary

models.Direction()**Direction()**
models.Direction(int, java.util.ArrayList)**Direction(int, ArrayList)** Constructor for the Direction class.

4.2.4 Fields

- models.Direction.way`public int` **way**
- models.Direction.arrivals`public java.util.ArrayList` **arrivals**

4.2.5 Constructors

- models.Direction()**Direction**

  ```
  public Direction()
  ```

- models.Direction(int, java.util.ArrayList)**Direction**

  ```
  public Direction(int way,java.util.ArrayList arrivals)
  ```

  - **Description**
    Constructor for the Direction class.
  - **Parameters**
    - `way` – Integer: 0 represents coming, 1 represents going.
    - `arrivals` – ArrayList of type Arrival: Elements of the type arrival.

4.3 Class StationsWrapper

models.StationsWrapper

Class which holds the wrapper for objects of type TransportStation. The wrapper is represented by the following type of element in the XML document: ... ... * ... *

4.3.1 Declaration

```
public class StationsWrapper
 extends java.lang.Object
```

4.3.2 Field summary

models.StationsWrapper.transport$_s$*tations***transport_stations**

4.3.3 Constructor summary

models.StationsWrapper()**StationsWrapper()** Constructor of the StationsWrapper class.

4.3.4 Method summary

models.StationsWrapper.getArticles()**getArticles()** Method which returns all the objects of type transport-station which appear in the XML document.
models.StationsWrapper.setArticles(java.util.List)**setArticles(List)** Set the TransportStation list with a given one.

4.3.5 Fields

- models.StationsWrapper.transport$_s$*tations*`private java.util.List` **transport_stations**

4.3.6 Constructors

- models.StationsWrapper()**StationsWrapper**

  ```
  public StationsWrapper()
  ```

  – **Description**
  Constructor of the StationsWrapper class.

4.3.7 Methods

- models.StationsWrapper.getArticles()**getArticles**

  ```
  public java.util.List getArticles()
  ```

- **Description**
  Method which returns all the objects of type transport-station which appear in the XML document.
- **Returns** – A list composed of TransportStation objects.

- models.StationsWrapper.setArticles(java.util.List)**setArticles**

```
public void setArticles(java.util.List transport_stations)
```

  - **Description**
    Set the TransportStation list with a given one.
  - **Parameters**
    * `transport_stations` – A list of elements of the type TransportStation.

4.4 Class Time

models.Time

Class which holds the implementation for time handling. Time is used by the timetable element, through the arrival element.

4.4.1 Declaration

```
public class Time
 extends java.lang.Object
```

4.4.2 Field summary

models.Time.time**time**

4.4.3 Constructor summary

models.Time()**Time()**
models.Time(java.lang.String)**Time(String)**

4.4.4 Method summary

models.Time.toString()**toString()**

4.4.5 Fields

- models.Time.time`public java.lang.String` **time**

4.4.6 Constructors

- models.Time()**Time**

```
public Time()
```

- models.Time(java.lang.String)**Time**

```
public Time(java.lang.String time)
```

4.4.7 Methods

- models.Time.toString()**toString**

```
public java.lang.String toString()
```

4.5 Class TimeTable

models.TimeTable

Class which holds the implementation for an XML element of the type timetable. A timetable element is responsible of holding information about the directions and arrival for a specific vehicle. It is represented as follows in the XML document: ... ...

4.5.1 Declaration

```
public class TimeTable
 extends java.lang.Object
```

4.5.2 Field summary

models.TimeTable.direction**direction**
models.TimeTable.vehicleID**vehicleID**

4.5.3 Constructor summary

models.TimeTable()**TimeTable()**

4.5.4 Fields

- models.TimeTable.vehicleID`public int` **vehicleID**
- models.TimeTable.direction`public java.util.ArrayList` **direction**

4.5.5 Constructors

- models.TimeTable()**TimeTable**

```
public TimeTable()
```

4.6 Class TimetablesWrapper

models.TimetablesWrapper

Class which holds the wrapper for the Timetable object. The wrapper is represented by the following type of element in the XML document: ... ...

4.6.1 Declaration

```
public class TimetablesWrapper
 extends java.lang.Object
```

4.6.2 Field summary

models.TimetablesWrapper.timeTables**timeTables**

4.6.3 Constructor summary

models.TimetablesWrapper()**TimetablesWrapper()**

4.6.4 Method summary

models.TimetablesWrapper.getArticles()**getArticles()**
models.TimetablesWrapper.setArticles(java.util.List)**setArticles(List)**

4.6.5 Fields

- models.TimetablesWrapper.timeTables`private java.util.List` **timeTables**

4.6.6 Constructors

- models.TimetablesWrapper()**TimetablesWrapper**

```
public TimetablesWrapper()
```

4.6.7 Methods

- models.TimetablesWrapper.getArticles()**getArticles**

```
public java.util.List getArticles()
```

- models.TimetablesWrapper.setArticles(java.util.List)**setArticles**

```
public void setArticles(java.util.List timetables)
```

4.7 Class TransportStation

models.TransportStation

Class which holds the implementation for JAXB binding for a transport station XML element. A transport station element has the following structure in the XML document: `2406 Tv9b Bv Sudului_2 Bulevardul Sudului / Hotel Lido (AEM) Sudului Sudului 45.737211 21.250093 0 dup script 11.12.16. http://maps.google.com/maps?q=Bulevardul%20Sudului%20/%20Hotel%20Lido@45.737211,21.250093 0` Using the StationsInteractor class we can perform the following operations such as delete, add, edit and query.

4.7.1 Declaration

```
public class TransportStation
 extends java.lang.Object
```

4.7.2 Field summary

models.TransportStation.friendlyStationName**friendlyStationName**
models.TransportStation.gmaps$_{l}inks$**gmaps_links**
models.TransportStation.info$_{c}omments$**info_comments**
models.TransportStation.is$_{i}nvalid$**is_invalid**
models.TransportStation.junctionName**junctionName**
models.TransportStation.lat**lat**
models.TransportStation.lineID**lineID**
models.TransportStation.lineName**lineName**
models.TransportStation.longitude**longitude**
models.TransportStation.rawStationName**rawStationName**
models.TransportStation.shortStationName**shortStationName**
models.TransportStation.stationID**stationID**
models.TransportStation.verification$_{d}ate$**verification_date**
models.TransportStation.verified**verified**

4.7.3 Constructor summary

models.TransportStation()**TransportStation()**
models.TransportStation(int)**TransportStation(int)** Constructor for the TransportStation class, using only a station id for creation.
models.TransportStation(int, java.lang.String, int, java.lang.String, java.lang.String, java.lang.String, java.lang.String, double, double, java.lang.Boolean, java.lang.String, java.lang.String, java.lang.String, java.lang.String)**TransportStation(int, String, int, String, String,**

String, String, double, double, Boolean, String, String, String, String) Constructor for the TransportStation object.

4.7.4 Method summary

models.TransportStation.toString()**toString()** Override of string form for a TransportStation object.

4.7.5 Fields

- models.TransportStation.lineID`public int` **lineID**
- models.TransportStation.stationID`public int` **stationID**
- models.TransportStation.lineName`public java.lang.String` **lineName**
- models.TransportStation.rawStationName`public java.lang.String` **rawStationName**
- models.TransportStation.friendlyStationName`public java.lang.String` **friendlyStationName**
- models.TransportStation.shortStationName`public java.lang.String` **shortStationName**
- models.TransportStation.junctionName`public java.lang.String` **junctionName**
- models.TransportStation.lat`public double` **lat**
- models.TransportStation.longitude`public double` **longitude**
- models.TransportStation.is$_{i}nvalid$`public java.lang.Boolean` **is__invalid***models.TransportStation.ve*
- models.TransportStation.verification$_{d}ate$`public java.lang.String` **verification__date***models.Transp*

- models.TransportStation.info$_{c}omments$`public java.lang.String` **info__comments**

4.7.6 Constructors

- models.TransportStation()**TransportStation**

```
public TransportStation()
```

- models.TransportStation(int)**TransportStation**

```
public TransportStation(int station_id)
```

  - **Description**
    Constructor for the TransportStation class, using only a station id for creation.
  - **Parameters**

* `station_id` – Id of the station.

- models.TransportStation(int, java.lang.String, int, java.lang.String, java.lang.String, java.lang.String, java.lang.String, double, double, java.lang.Boolean, java.lang.String, java.lang.String, java.lang.String, java.lang.String)**TransportStation**

```
public TransportStation(int lineID, java.lang.String lineName, int
    stationID, java.lang.String rawStationName, java.lang.String
    friendlyStationName, java.lang.String shortStationName, java.
    lang.String junctionName, double lat, double longitude, java.
    lang.Boolean is_invalid, java.lang.String verified, java.lang.
    String verification_date, java.lang.String gmaps_links, java.
    lang.String info_comments)
```

  - **Description**
    Constructor for the TransportStation object. All the necessary parameters for constructing a transport station element are set here.
  - **Parameters**
    * `lineID` – int: Id of the line for the transport station. Example: 1266.
    * `lineName` – String: Name of the line. Example: Tv4.
    * `stationID` – int: id of the station.
    * `rawStationName` – String: Raw name for the station. Example: P-ta Crucii_2.
    * `friendlyStationName` – String: Friendlier version of the raw station name. Example: Piata Crucii (Torontalului)
    * `shortStationName` – String: Shorter version for the station name. Example: P-ta Crucii.
    * `junctionName` – String: Name of the junction. Example: P-ta Crucii.
    * `lat` – double: Latitude of the station location.
    * `longitude` – double: Longitude of the station location.
    * `is_invalid` – Boolean: States whether the station is still in use.
    * `verified` – String: Method of the station is verified.
    * `verification_date` – String: Date of the last verification.
    * `gmaps_links` – String: Link for google maps location.
    * `info_comments` – String: More info.

4.7.7 Methods

- models.TransportStation.toString()**toString**

```
public java.lang.String toString()
```

  - **Description**
    Override of string form for a TransportStation object.
  - **Returns** – Pretty printed format of a vehicle instance.

4.8 Class Vehicle

models.Vehicle

Class which holds the implementation for a vehicle object. A vehicle is represented as follows in the XML document: `M41 Bus` Operations of the type edit, add, delete and query involving the vehicle element will be done using the Vehicle Interactor class.

4.8.1 Declaration

```
public class Vehicle
 extends java.lang.Object
```

4.8.2 Field summary

models.Vehicle.vehicleID**vehicleID**
models.Vehicle.vehicleName**vehicleName**
models.Vehicle.vehicleType**vehicleType**

4.8.3 Constructor summary

models.Vehicle()**Vehicle()**
models.Vehicle(int, java.lang.String, java.lang.String)**Vehicle(int, String, String)**
Constructor for the Vehicle class.

4.8.4 Method summary

models.Vehicle.toString()**toString()** Override of string form for a vehicle object.

4.8.5 Fields

- models.Vehicle.vehicleID`public int` **vehicleID**
- models.Vehicle.vehicleName`public java.lang.String` **vehicleName**
- models.Vehicle.vehicleType`public java.lang.String` **vehicleType**

4.8.6 Constructors

- models.Vehicle()**Vehicle**

```
public Vehicle()
```

- models.Vehicle(int, java.lang.String, java.lang.String)**Vehicle**

```
public Vehicle(int id,java.lang.String name,java.lang.String
    type)
```

- **Description**
  Constructor for the Vehicle class.
- **Parameters**
  - `id` – Unique id for the vehicle object.
  - `name` – Name of the vehicle.
  - `type` – Type of the vehicle.

4.8.7 Methods

- models.Vehicle.toString()**toString**

```
public java.lang.String toString()
```

  - **Description**
    Override of string form for a vehicle object.
  - **Returns** – Pretty printed format of a vehicle instance.

4.9 Class VehiclesWrapper

models.VehiclesWrapper

Class which holds the wrapper for the Vehicle object. The wrapper is represented by the following type of element in the XML document: `... ... * ... *`

4.9.1 Declaration

```
public class VehiclesWrapper
 extends java.lang.Object
```

4.9.2 Field summary

models.VehiclesWrapper.vehicles**vehicles**

4.9.3 Constructor summary

models.VehiclesWrapper()**VehiclesWrapper()** Constructor of the VehicleWrapper class.

4.9.4 Method summary

models.VehiclesWrapper.getArticles()**getArticles()** Method which returns all the objects of type vehicle which appear in the XML document.
models.VehiclesWrapper.setArticles(java.util.List)**setArticles(List)** Set a list of vehicles.

4.9.5 Fields

- models.VehiclesWrapper.vehicles`private java.util.List` **vehicles**

4.9.6 Constructors

- models.VehiclesWrapper()**VehiclesWrapper**

```
public VehiclesWrapper()
```

  - **Description**
    Constructor of the VehicleWrapper class. Creates the vehicles list.

4.9.7 Methods

- models.VehiclesWrapper.getArticles()**getArticles**

```
public java.util.List getArticles()
```

  - **Description**
    Method which returns all the objects of type vehicle which appear in the XML document.
  - **Returns** – A list composed of Vehicle objects.

- models.VehiclesWrapper.setArticles(java.util.List)**setArticles**

```
public void setArticles(java.util.List vehicles)
```

  - **Description**
    Set a list of vehicles.
  - **Parameters**
    * `vehicles` – A list of elements of type Vehicle.