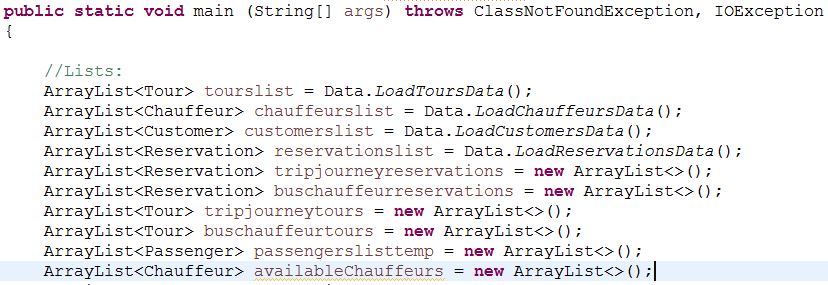
**Implementation**

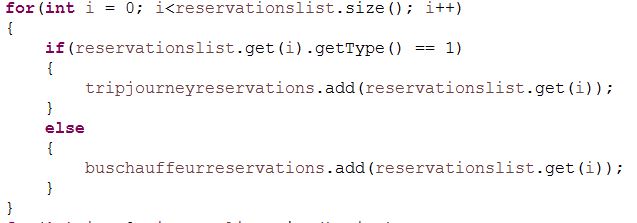
Our group thought of the best ways to display all necessary information from the system. There are lists containing tours, reservations, chauffeurs, passengers and customers. In order to edit or to add an item of a list we made a form-like design to be filled out really easily. The GUI is easy to understand and most important is user friendly.

**Code Snippet 1:**



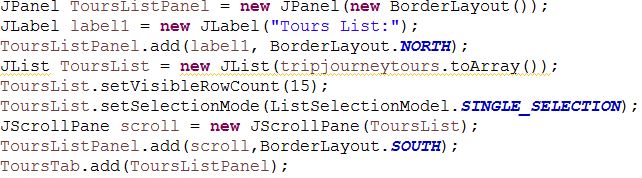
The GUI class starts out with these lines in which all the Arraylists that are necessary to the program are created and where all the saved data is loaded.

**Code Snippet 2:**



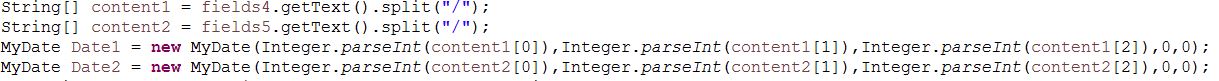
This snippet is responsible for separating the reservations by their type. Reservations for trips & journeys are placed in the “tripjourneyreservations” ArrayList and the reservations for Bus-and-Chauffeur are stored in the “buschauffeurreservations” ArrayList.

**Code Snippet 3:**



This code snippet is responsible for creating the Tours List JList that will contain the “tripjourneytours” ArrayList. This list will have a scroll bar if there are more than 15 elements inside it. All the other JLists created in this program are made in a similar way in order to maintain consistency.

**Code Snippet 4:**



This part of the code is responsible for reading the Departure Date and Arrival Date from the text fields of the GUI. The text fields contents are split at each “/” and creating two MyDate object (taking arguments the fields contents changed into integers).

**Code Snippet 5:**



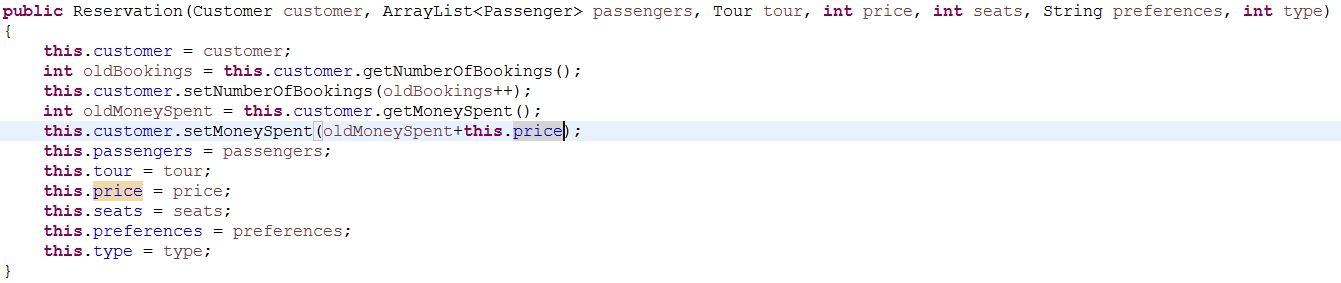
This code snippet is responsible for loading all the Tours from the “Tours.bin” file. The program will read an ArrayList of objects (of type Tour) and then return them (the returned object is used in the GUI class). All the other objects (Customer,Passenger,Chauffeur,Reservation) have a similar loading method.

**Code Snippet 6:**



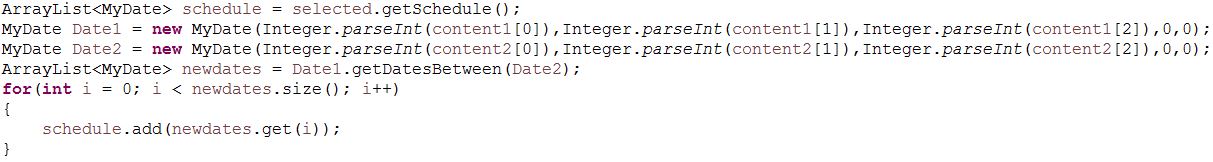
This code snippet is responsible for getting all the available chauffeurs and adding them to the “availableChauffeurs” Array List. It will compare each of the Chauffeur’s current schedule (an ArrayList of MyDate objects in which the chauffeur is busy) with the Tour’s dates (stored in “dates” ArrayList). If there are two equal MyDate objects in these two Array Lists the chauffeur will be unavailable.

**Code Snippet 7:**



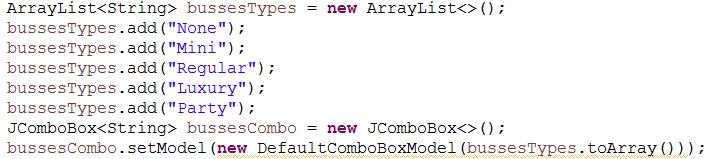
This is the constructor for the Reservation object. When creating a Reservation object, besides setting all instance variables, the customer that made the reservation will have the number of bookings increase by 1 and the money spent increase by the price of the Reservation made.

**Code Snippet 8:**



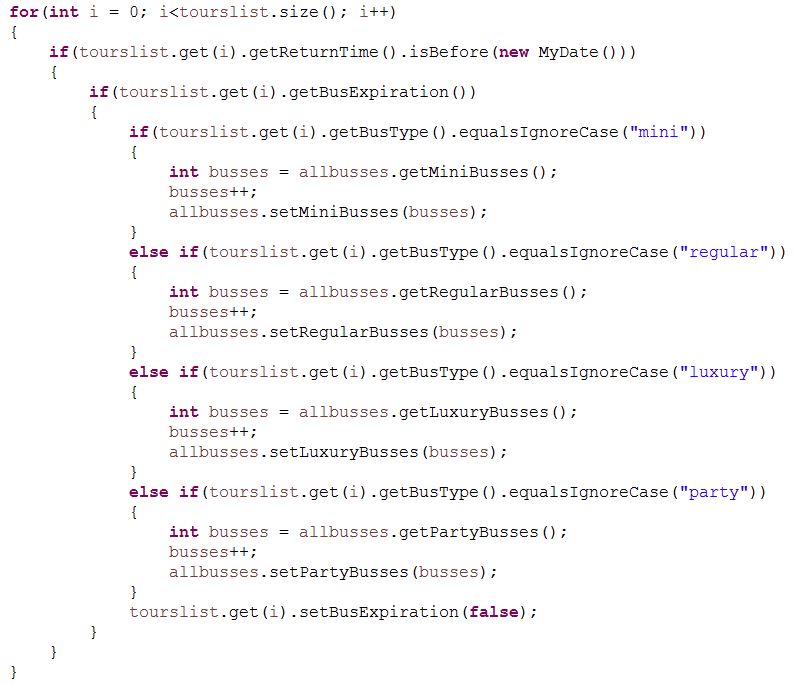
This snippet is part of setting a chauffeur to a tour. This will create two MyDate objects based on the date interval of the tour and then add all the dates in this interval to the chauffeur’s schedule.

**Code Snippet 9:**



This part of the code is responsible for creating a combo box filled with objects of type String. This combo box is used when creating a tour or making a bus-and-chauffeur reservation and lets the employee to select the bus.

**Code Snippet 10:**



This part of the code is responsible for going thourgh all the tours and checking if there are expired tours in order to set the bus used on that tour to be available. Besides checking if the tour is expired is also checking the Boolean variable inside each tour that is returned by the “getBusExpiration” method in order to avoid the duplication of the same bus.