# **A2 Code Complexities**

In this document I will highlight notable aspects of my program which may go unnoticed by the user.

Here is a summary of the good aspects of my system to give you a greater understanding of its capabilities:

* The interface is clean and user-friendly
* The close button on the menu strip of each form asks for confirmation
* An awareness of the wider system is shown on Crèche Menu Form and on the menu strip
* Parents
  + Validations
    - When key is pressed
      * Forename, surname and city textboxes don’t accept numbers
        + Error message informs user of this and disappears once a valid character is input
      * Phone number text box doesn’t accept letters
        + Error message informs user of this and disappears once a valid character is input
    - When button is clicked
      * Postcode length
      * Phone number length
      * Doesn’t allow a parent to be created or edited with the exact same details as an already existing parent
  + The user can choose to either save a parent and stay on Add Parent Form or save a parent and be brought to Add Child Form when adding a parent
    - If the user chooses to be brought to Add Child Form the parent, they just created will be selected as the child’s parent
  + When parent is deleted
    - All children of that parent are deleted
      * All future bookings of the children are deleted
      * Past bookings are not deleted so that records can be kept of attended bookings
      * When something is deleted, its deleted field is simply marked as true instead of erasing all trace of it
        + This is so that records can be kept of everything; Woodside Community Hub may want to find out statistics about the nature of parents, children or bookings being deleted
* Children
  + The same validations apply as with parents, excluding the phone number and postcode
  + The parents displayed in the parent name list box dynamically change as the user enters text into the text box
  + The date of birth calendar only allows the user to select dates of children that are between six months and four years old
* Bookings
  + Parents and children can be either created or selected when creating a booking
    - This is because the user might not think to create the child they want to book before loading Booking Select Child Form; this acts as a quick shortcut
    - If a new parent is created the existing child radio button is disabled as no children of that parent will exist
    - The child is automatically placed in the appropriate group based on their age when the user advances to Booking Select Dates Form
  + The user can only select weekdays to book
  + The start date calendar doesn’t allow the user to book in the past
    - The end date calendar doesn’t let the user select a date that is less than one month after the selected start date
  + The duration label dynamically changes every time the user selects a different start or end date
  + The number of days booked, and total cost labels dynamically change every time the user selects or deselects days to book, as well as when the start or end date is changed
  + Discounts are applied as outlined in the case study
    - If the booking start date is between three and six months after the current date, a 3% discount is applied to the total cost
    - If the booking start date is six months or more after the current date, a 5% discount is applied to the total cost
  + Bookings cannot be made that overlap with existing bookings of the current child
* Viewing data
  + Data from every table can be viewed via radio buttons
* Report
  + The average booking duration is calculated, a very complex calculation considering the duration is not stored as a field
  + The report displays data for bookings with start dates within the range selected by the user

## **Booking Select Child Form**

### Search Functionality (also found in Add Child Form and all edit and delete forms)

This is one of the most prominent complex features in my system, and it allows the user to scroll through a list of all (in this case) parents or simply input text into the text box and the contents of the list box will be streamlined based on which parents have that text in their name.

#### txtParentName text changed event

This adds every parent which contains the text found in the search text box to a list (updateParents) and goes through the list in a foreach loop, adding the full name of each parent to the list box, ready to be selected. It also adds a list of each parent’s details to parentValuesList so that the data of that parent can be accessed if it is selected in the list box.

#### txtParentName text changed event

This assigns the values of the selected parent to variables, so that they can be assigned to the selectedParent object once the select parent validation is acceptable. When a parent is selected, SelectExistingParent() is run, which instantly updates the select child panel, without requiring that the user clicks a button.

### Child Age Validation

The month calendar from which the user must select a new child’s date of birth when they are adding a new child only displays dates between six months and four years from the current date. This goes with the case study, which states that a child cannot be added unless they are within six months and four years old. However, the case study states that a child who is already on the system may stay on until they are six years old; in this case, the child is no longer able to be selected for booking. My system facilitates this aspect of the case study, allowing a child who is five years of age to be booked into the oldest age group, so long as the child was added to the database before they turned five (as the adding child functionality does not enable this). It must be noted that this crèche operates on a basis of the child’s age in years, not a based off whether the child was born before or after the 1st of July.

#### Add New Child Age Validation

#### Select Child of Ages 6 Months to 5 Years

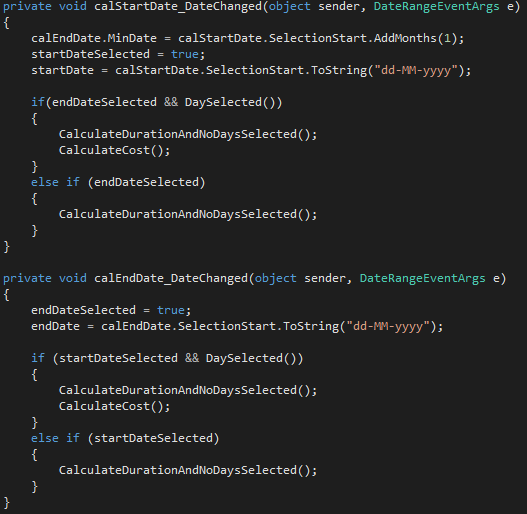
Here, the child is only added to the select child combo box if they are under five years old and over six months old.

## **Booking Select Dates Form**

### Calculate Booking Statistics Functionality

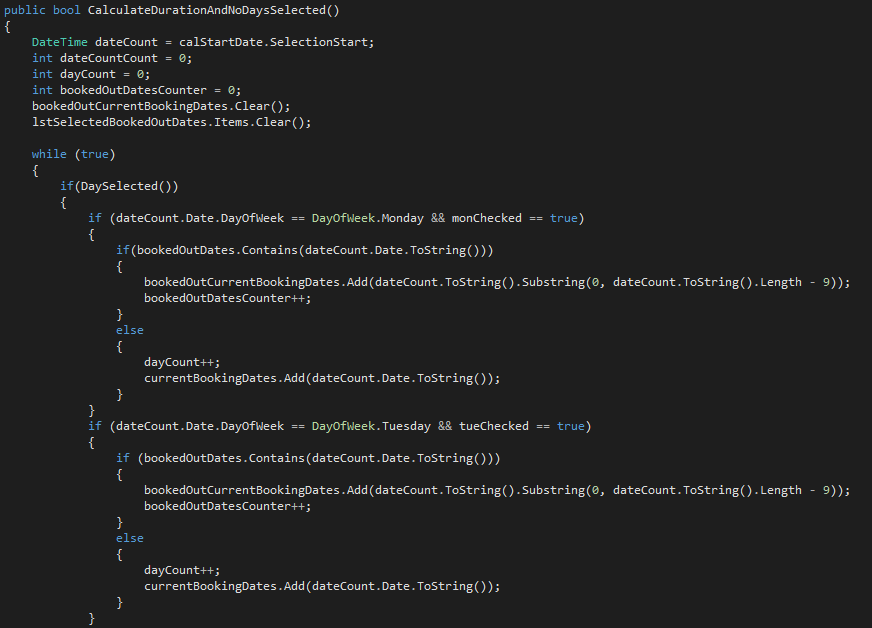
This functionality dynamically updates the booking duration, number of days selected, and cost of booking based on the start date, end date and days selected by the user. On each date change or check box check change, the necessary methods are run in order to calculate and display the data for the current booking. Discounts are also calculated, and the exact discount is displayed to the user.

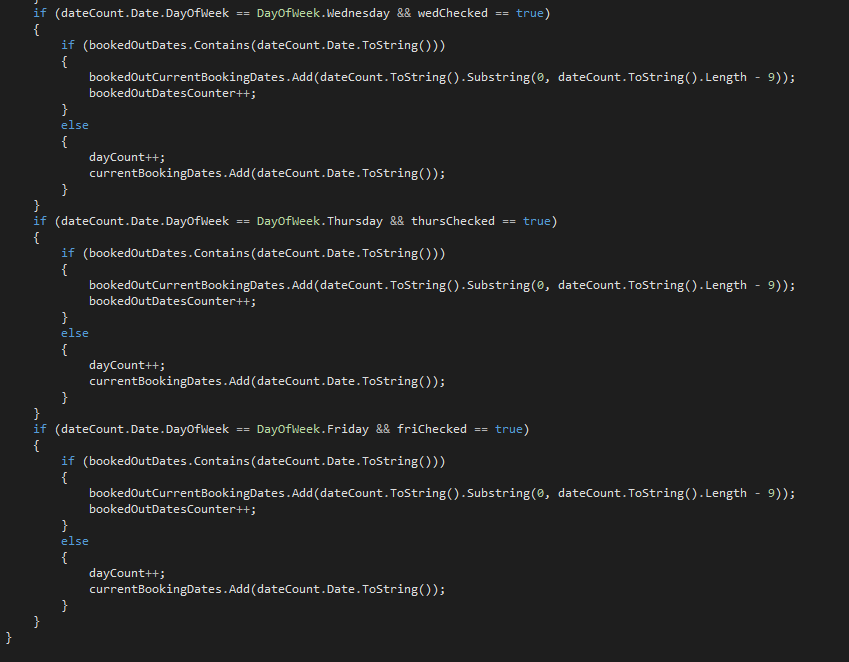
#### calStartDate and calEndDate date changed events



#### chkMonday check changed event (same for all other check boxes)

#### CalculateDurationAndNoDaysSelected() method





#### 

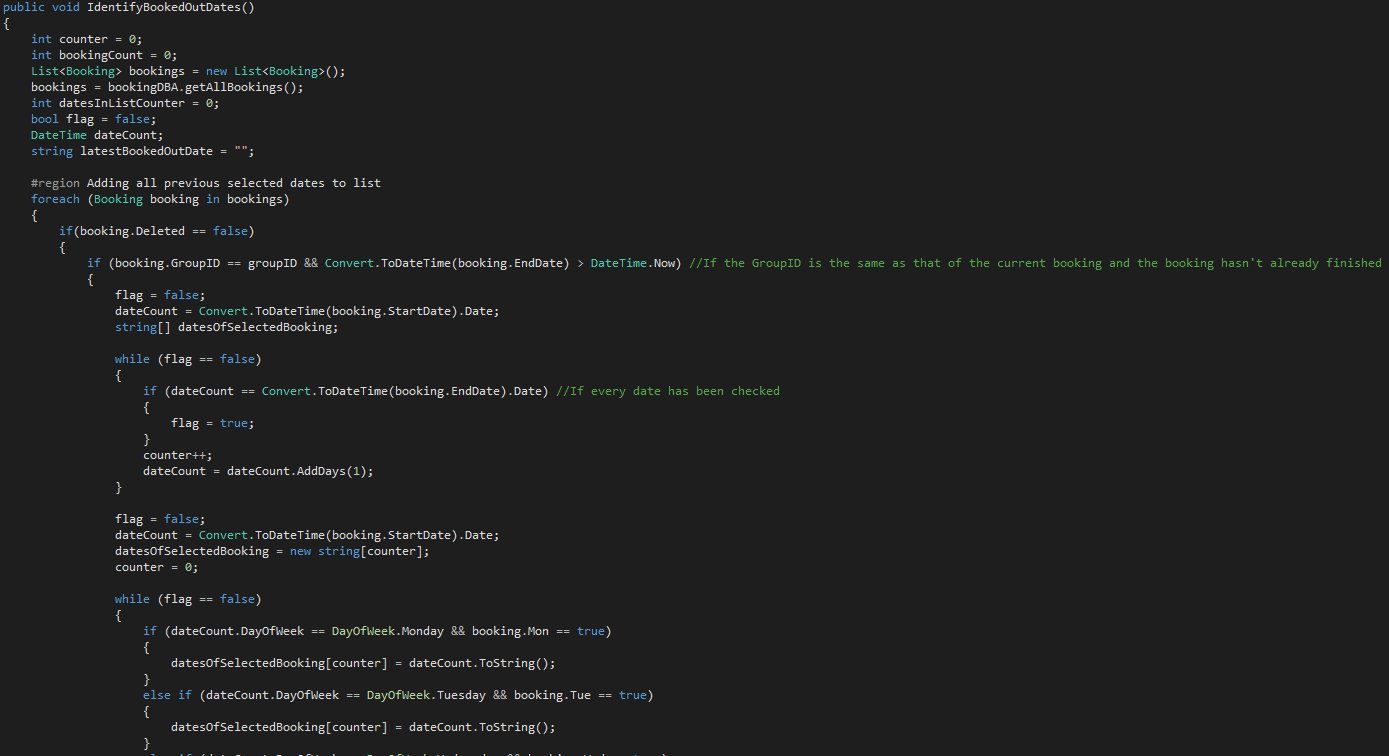
#### CalculateCost() method

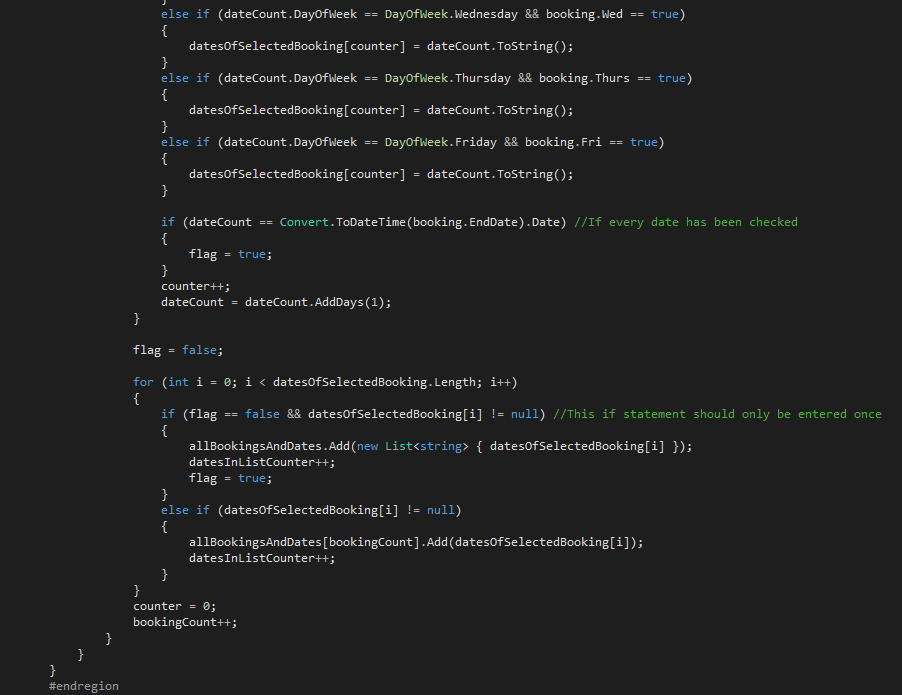
A 3% discount is applied if the booking is made between three and six months in advance, and a 5% discount is applied if the booking is made six months or more in advance.

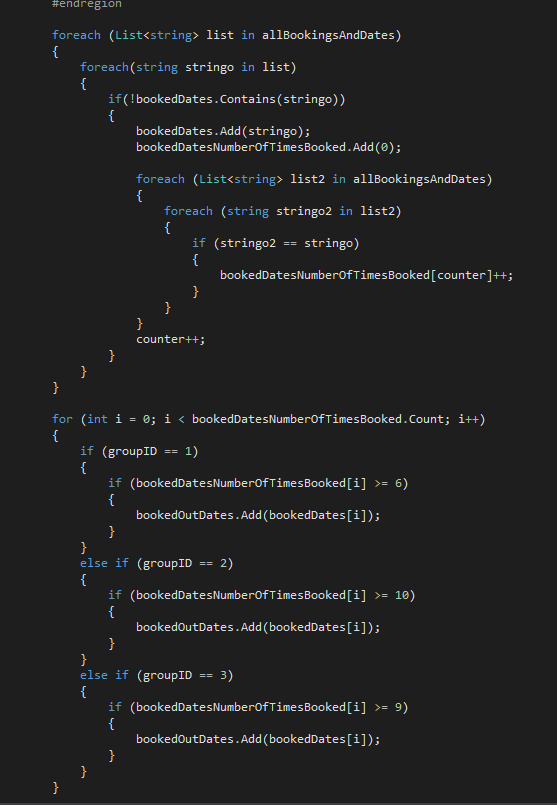
### Find and Display Booked Out Dates

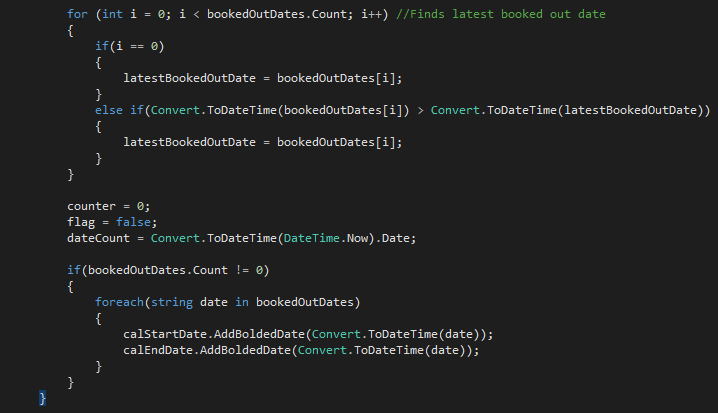
The user is informed of all dates which are booked out and will not be included in any bookings made as all booked out dates for the current booking are emboldened on both the start date and end date calendars. In addition, once the user has selected a start date, end date and at least one weekday, a list box is populated to display every date in the current booking which is booked out and therefore cannot be attended by the child. This is facilitated for in the cost; no days of the current booking are included in the cost so that the parent is not paying for days which they will not be able to book for their children. Additionally, the label displaying the number of days selected does not include the dates which are booked out. Therefore, this functionality is incredibly practical, allowing for a smooth and fair booking experience, regardless of the booked out days.

#### IdentifyBookedOutDates() method









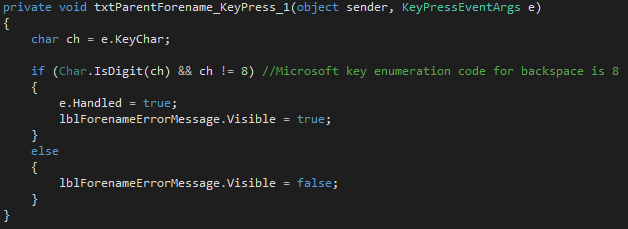
This highly complex method is run as soon as the form loads, and it adds every booked out date of past bookings of the selected group to a list, while emboldening each one on both calendars.

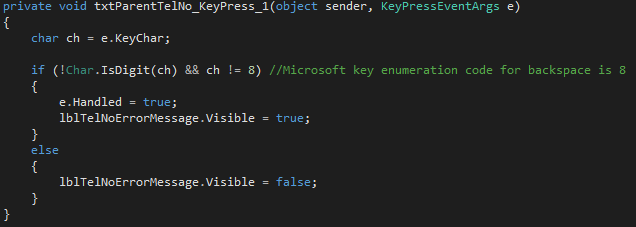
The code which adds each booked out date of the current booking to a calendar can be found in the CalculateDurationAndNoDays() method, shown in the previous complexity.

## **Add Parent Form**

### Validation on Key Press Functionality (seen in every form in which a parent or child can be added or edited)

An error message is displayed to the user if they attempt to input a numerical value in the forename, surname or city/town text boxes, and alternatively a non-numerical value in the phone number text box. This allows for a more streamlined user experience which guides the user as much as possible, not allowing them to make some simple mistakes.

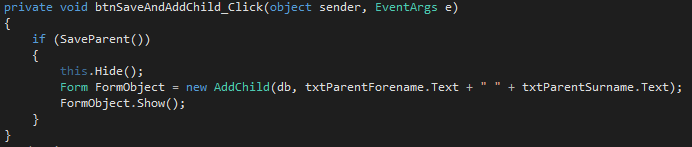




## **Add Child Form**

### Shortcut from Add Parent Form

If the user adds a valid parent in the add parent form by clicking ‘Save and add child’, the add child form will be loaded and a different constructor will be run based off the different parameters which are passed to the constructor when this button is clicked. This showcases a key principle of object orientation, overloading. The extra parameter that is passed when this button is clicked is the parent name, allowing for that parent to be automatically selected on the parent name list box.



## 

## **Edit and Delete Booking Form**

### Only Edit Future Bookings Functionality

While the user can view and delete any booking, only the bookings that are at least one day in the future are editable. When a booking that is not at least one day in the future is selected, the start and end date calendars as well as Monday to Friday check boxes are disabled, meaning the user can see their values but not change them. The edit button is also disabled, however view details and delete booking are not.

#### cmbBookingDates selected index changed event

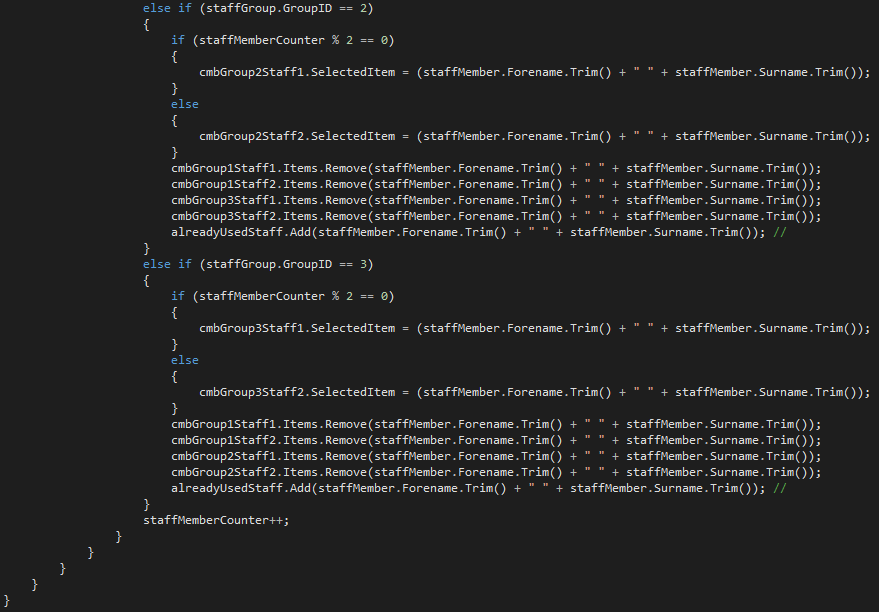
#### 

## **Assign Staff Form**

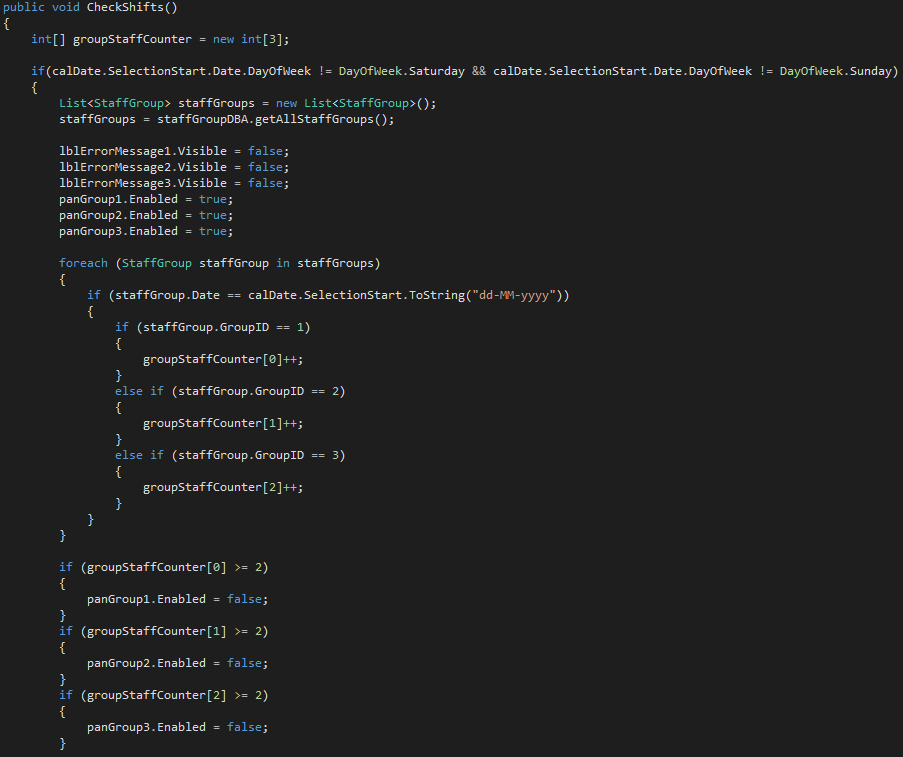
### Dynamically Changing Contents of Combo Boxes

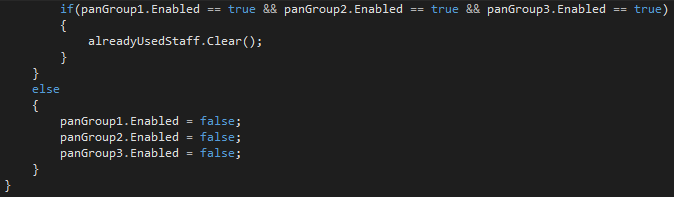
In this form, two staff members can be assigned to shifts for each group. If a staff member is selected in one combo box, it cannot be selected in any of the other combo boxes. If the user unselects this staff member by clicking the red x beside the combo box, the staff member is added back to the other combo boxes. If the user has already assigned the staff members to a group on a specific date, that staff member will not be available to be assigned to any other group on that date.

#### PopulateComboBoxes() method



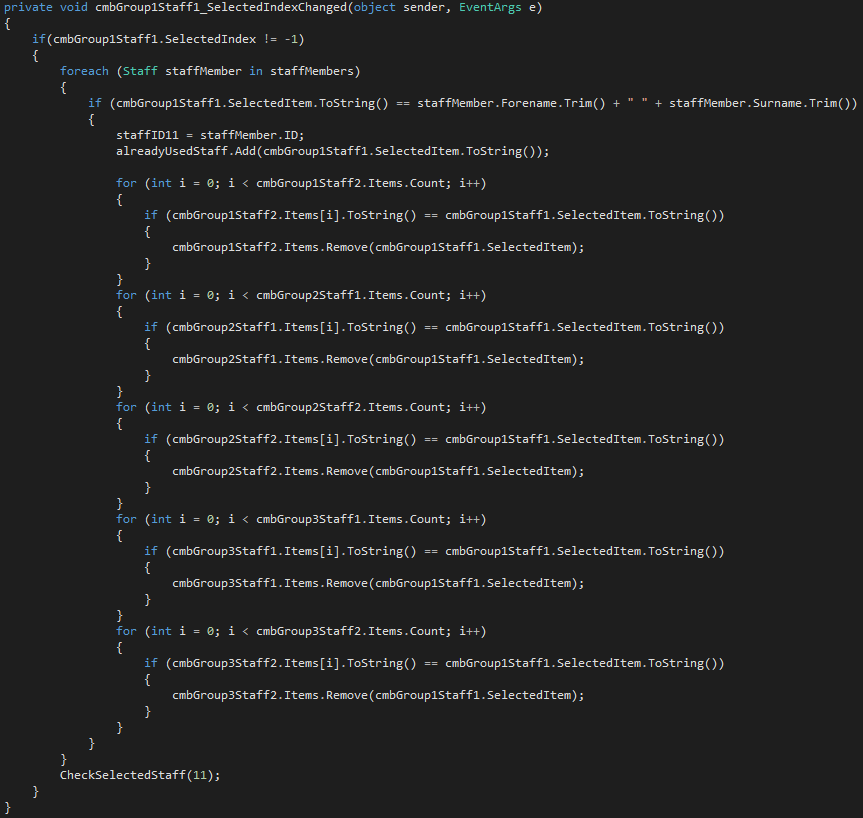
#### CheckShifts() method



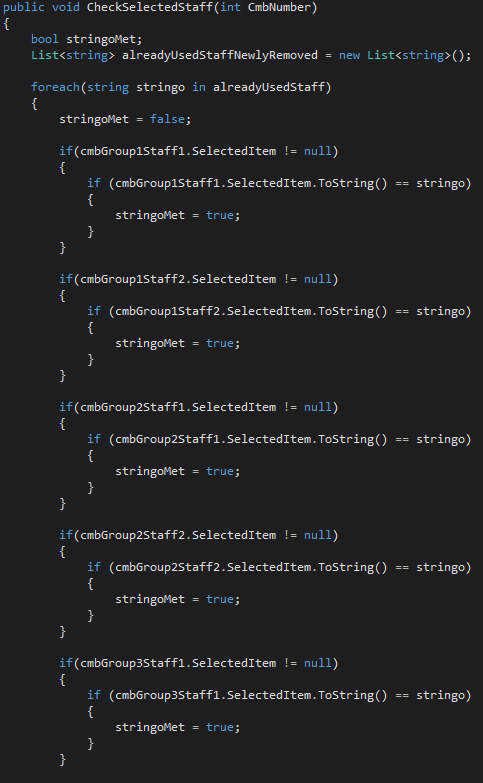


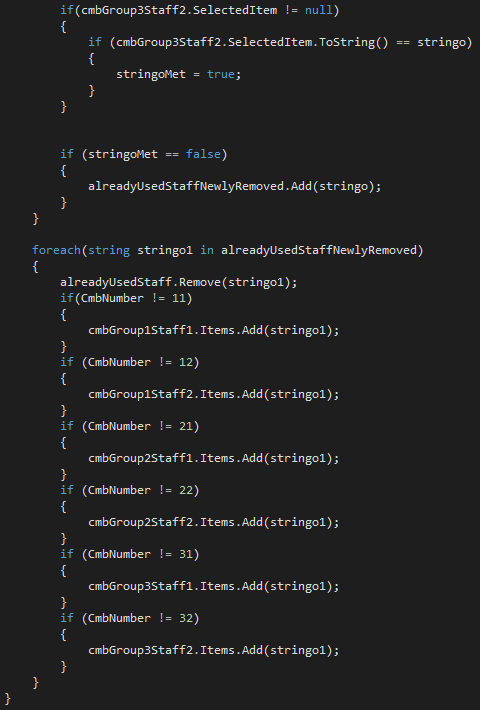
These two methods are run as soon as the form loads.

#### cmbGroup1Staff1 check changed event (same for all other combo boxes)



#### CheckSelectedStaff() method





The CheckSelectedStaff() method checks if the same staff members that were selected last time the method was called are still selected and updates the combo boxes to facilitate for the changes.