Student Planner A+

Software Architecture / SOEN 6471

Dr. Peter C Rigby

Winter 2013

Document Name : Patchsets for M4

Jamileh Mohammadkhani Ghiyasvand	Student ID : 5990351
Bahareh Mohammadpourbarghi	Student ID : 5894336
Shahla Noori	Student ID : 5972671
Robert Saliba	Student ID : 6412033
Harcharan Singh Pabla	Student ID : 6547702

Refactoring 1:

Patchset 1/8: SHA-1: 5c80f8a195899ae41dd5a80363a27df734272413

*** Extract Method checkingClassNames**

The method SaveClass is a kinda big method having almost 90 lines of code. To make this method cleaner and easier to read, understand and modify the related lines are extracted to a new method and are called in main method (SaveClass).
CheckingClassNames method is taking care of checking the Class Criteria and it returns a bool by validating the different conditions.

```
5c80f8a195899ae41dd5a80363a27df734272413
 SourceCode/Calendar/AddClassForm.cs | 13 ++++++++++---
 1 file changed, 10 insertions(+), 3 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index 6800f96..7fb687e 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -57,9 +57,8 @@ public AddClassForm()
       private bool saveClass() {

          //ensure at least one day is checked
-         if (txtClassName.Text.Equals("") || (chkClassMonday.Checked == false &&
chkClassTuesday.Checked == false) &&
-            chkClassWednesday.Checked == false && chkClassThursday.Checked == false &&
chkClassFriday.Checked == false ||
-            (chkClassFinished.Checked == true && cbFinalLetterGrade.Text.Equals(""))) {
+         if (checkingClassNames())
+         {
            Util.displayRequiredFieldsError(new string[] { "Class Name", "Days" });
            return false;
          }
@@ -187,5 +186,13 @@ public AddClassForm()
          methods = addGradeCategories.getMethods();
       }

+      #region Methods
+      private bool checkingClassNames()
+      {
+         return txtClassName.Text.Equals("") || (chkClassMonday.Checked == false &&
chkClassTuesday.Checked == false) &&
+                     chkClassWednesday.Checked == false && chkClassThursday.Checked == false &&
chkClassFriday.Checked == false ||
+                     (chkClassFinished.Checked == true && cbFinalLetterGrade.Text.Equals(""));
+      }
+      #endregion
    }
```

```
}
```

Patchset 2/8: `SHA-1: 1f1dc1ab146c01a195c5f15fc7c1327322ddbc36`

*** Extract Method raiseInvalidClassNameError**

```
raiseInvalidClassNameError method is added which displays a proper
message depending on the output value of checkingClassNames method. And
These lines are deleted from SaveClass methods.
```

```
1f1dc1ab146c01a195c5f15fc7c1327322ddbc36
 SourceCode/Calendar/AddClassForm.cs | 9 +++++++--
 1 file changed, 7 insertions(+), 2 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index 7fb687e..1328c87 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -59,8 +59,7 @@ public AddClassForm()
          //ensure at least one day is checked
          if (checkingClassNames())
          {
-             Util.displayRequiredFieldsError(new string[] { "Class Name", "Days" });
-             return false;
+             return raiseInvalidClassNameError();
          }

          //grade categories required if class is not finished
@@ -193,6 +192,12 @@ private bool checkingClassNames()
                      chkClassWednesday.Checked == false && chkClassThursday.Checked == false &&
chkClassFriday.Checked == false ||
                      (chkClassFinished.Checked == true && cbFinalLetterGrade.Text.Equals(""));
      }
+
+      private static bool raiseInvalidClassNameError()
+      {
+          Util.displayRequiredFieldsError(new string[] { "Class Name", "Days" });
+          return false;
+      }
      #endregion
   }
 }
```

Patchset 3/8: `SHA-1: 9e13d5832d86b72187a3814981e45044e898da99`

*** Extract Method raiseInvalidGradeCategory**

raiseInvalidGradeCategory method is added which displays a proper message and returns false, these lines are deleted from SaveClass methods.

```
9e13d5832d86b72187a3814981e45044e898da99
 SourceCode/Calendar/AddClassForm.cs | 12 ++++++++---
 1 file changed, 9 insertions(+), 3 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index 1328c87..f8aded5 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -63,9 +63,9 @@ public AddClassForm()
         }

         //grade categories required if class is not finished
-        if (chkClassFinished.Checked == false && categories.Count == 0) {
-            Util.displayRequiredFieldsError("Grade Categories");
-            return false;
+        if (chkClassFinished.Checked == false && categories.Count == 0)
+        {
+            return raiseInvalidGradeCategory();
         }

         //ensure start and end times are legal
@@ -198,6 +198,12 @@ private static bool raiseInvalidClassNameError()
         Util.displayRequiredFieldsError(new string[] { "Class Name", "Days" });
         return false;
     }
+
+    private static bool raiseInvalidGradeCategory()
+    {
+        Util.displayRequiredFieldsError("Grade Categories");
+        return false;
+    }
     #endregion
   }
 }
```

Patchset 4/8: `SHA-1: 2a34890b5875692950d01eeb1272631606f36d14`

*** Extract Method rasiInavlidTimeErorr**

rasiInavlidTimeErorr method is added which displays a proper message and returns false, these lines are deleted from SaveClass methods.

```
2a34890b5875692950d01eeb1272631606f36d14
 SourceCode/Calendar/AddClassForm.cs | 12 +++++++++---
 1 file changed, 9 insertions(+), 3 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index f8aded5..c2d78ae 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -69,9 +69,9 @@ public AddClassForm()
         }

         //ensure start and end times are legal
-        if (dtClassStartTime.Value.TimeOfDay > dtClassEndTime.Value.TimeOfDay) {
-            Util.displayError("Invalid Start and End Times", "Error");
-            return false;
+        if (dtClassStartTime.Value.TimeOfDay > dtClassEndTime.Value.TimeOfDay)
+        {
+            return rasiInavlidTimeErorr();
         }

         //set current grade to null unless the class is finished, upon which get the entered grade
@@ -204,6 +204,12 @@ private static bool raiseInvalidGradeCategory()
         Util.displayRequiredFieldsError("Grade Categories");
         return false;
       }
+
+      private static bool rasiInavlidTimeErorr()
+      {
+          Util.displayError("Invalid Start and End Times", "Error");
+          return false;
+      }
       #endregion
   }
 }
```

Patchset 5/8 : `SHA-1: 0ff7d4ca67b5b8da62f9b1c980e6c30346a7bb05`

*** Extract Method raiseInvalidLetterGradeError**

```
raiseInvalidLetterGradeError method is added which displays a proper
message and returns false, these lines are deleted from SaveClass
methods.
```

```
0ff7d4ca67b5b8da62f9b1c980e6c30346a7bb05
 SourceCode/Calendar/AddClassForm.cs | 15 ++++++++++----
 1 file changed, 11 insertions(+), 4 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index c2d78ae..b531c31 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -76,11 +76,12 @@ public AddClassForm()

          //set current grade to null unless the class is finished, upon which get the entered grade
          string currentGrade = "null";
-         if (chkClassFinished.Checked == true) {
+         if (chkClassFinished.Checked == true)
+         {
             //make sure user has selected a value
-            if (cbFinalLetterGrade.Equals("")) {
-               Util.displayError("Please select a valid letter grade for the class", "Invalid Letter Grade");
-               return false;
+            if (cbFinalLetterGrade.Equals(""))
+            {
+               return raiseInvalidLetterGradeError();
             }
             currentGrade = "'" + cbFinalLetterGrade.Text + "'";
          }
@@ -210,6 +211,12 @@ private static bool rasiInavlidTimeErorr()
          Util.displayError("Invalid Start and End Times", "Error");
          return false;
       }
+
+      private static bool raiseInvalidLetterGradeError()
+      {
+         Util.displayError("Please select a valid letter grade for the class", "Invalid Letter Grade");
+         return false;
+      }
       #endregion
    }
 }
```

Patchset 6/8: `SHA-1: 2151213f2ec2637502e5648026bfcbdbbe0d08af`

*** Extract Method insertIntoDB**

```
insertIntoDB method is added which handles all the operation regarding
inserting and committing the values into DB.
```

```
2151213f2ec2637502e5648026bfcbdbbe0d08af
 SourceCode/Calendar/AddClassForm.cs | 80 ++++++++++++++++++++++++----------
 1 file changed, 49 insertions(+), 31 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index b531c31..0c4dada 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -93,37 +93,7 @@ public AddClassForm()
         }

         //begin database transaction
-        Database.beginTransaction();
-        Database.modifyDatabase("INSERT INTO Class VALUES (null, " + Util.quote(txtClassName.Text) +
", '" + ctrCredits.Value
-            + "', '" + chkClassMonday.Checked + "', '" + chkClassTuesday.Checked + "', '" +
chkClassWednesday.Checked
-            + "', '" + chkClassThursday.Checked + "', '" + chkClassFriday.Checked + "'," + semesterIdValue
-            + ", TIME('" + dtClassStartTime.Value.TimeOfDay + "'), TIME('"
-            + dtClassEndTime.Value.TimeOfDay + "'), " + Util.quote(txtClassLocation.Text) + ", null, null," +
currentGrade + ");");
-
-        //get the id of the value just inserted
-        object classID = Database.getInsertedID();
-
-        //insert into database the class professor assignment
-        if (cbClassProfessor.SelectedIndex >= 0) {
-            Database.modifyDatabase("INSERT INTO ClassProfessor VALUES ('" +
profId[cbClassProfessor.SelectedIndex] + "', '" + classID + "');");
-        }
-
-        //insert grading scale
-        for (int i = 0; i < gradingScale.Length; i++) {
-            Database.modifyDatabase("INSERT INTO GradingScaleValue VALUES('" + gradeLetter[i] + "', '" +
classID + "', '" + gradingScale[i] + "');");
-        }
-
-        //add value for F
-        Database.modifyDatabase("INSERT INTO GradingScaleValue VALUES('F', '" + classID + "',
'0.00');");
-
-        //insert grade category
```

```
-          for (int i = 0; i < categories.Count; i++) {
-             Database.modifyDatabase("INSERT INTO GradeCategory VALUES('" + categories[i] + "', '" +
classID + "', '" +
-                percentages[i] + "', null, '" + methods[i] + "');");
-          }
-
-          //commit all inserts to database
-          Database.commit();
+           insertIntoDB(currentGrade, semesterIdValue);

           //clear all arrays after updating database
           categories.Clear();
@@ -187,6 +157,54 @@ public AddClassForm()
       }

       #region Methods
+       private void insertIntoDB(string currentGrade, string semesterIdValue)
+       {
+          //begin database transaction
+          Database.beginTransaction();
+
+          Database.modifyDatabase(
+             string.Format(@"INSERT INTO Class VALUES (null, {0}, '{1}', '{2}', '{3}', '{4}', '{5}', '{6}',{7},
TIME('{8}'), TIME('{9}'),
+                         {10}, null, null,{11});", Util.quote(txtClassName.Text), ctrCredits.Value,
chkClassMonday.Checked, chkClassTuesday.Checked,
+                         chkClassWednesday.Checked, chkClassThursday.Checked, chkClassFriday.Checked,
semesterIdValue, dtClassStartTime.Value.TimeOfDay,
+                         dtClassEndTime.Value.TimeOfDay, Util.quote(txtClassLocation.Text), currentGrade));
+
+          //get the id of the value just inserted
+          object classID = Database.getInsertedID();
+
+          //insert into database the class professor assignment
+          if (cbClassProfessor.SelectedIndex >= 0)
+          {
+             Database.modifyDatabase(
+                string.Format("INSERT INTO ClassProfessor VALUES ('{0}', '{1}');",
profId[cbClassProfessor.SelectedIndex], classID)
+                );
+          }
+
+          //insert grading scale
+          for (int i = 0; i < GradingScale.Length; i++)
+          {
+
+             Database.modifyDatabase(
+                string.Format("INSERT INTO GradingScaleValue VALUES('{0}', '{1}', '{2}');",
```

```
gradeLetter[i].ToString(), classID, GradingScale.code(i).ToInteger())
+                );
+          }
+
+          //add value for F
+          Database.modifyDatabase(
+             string.Format("INSERT INTO GradingScaleValue VALUES('F', '{0}', '0.00');", classID)
+             );
+
+          //insert grade category
+          for (int i = 0; i < categories.Count; i++)
+          {
+             Database.modifyDatabase(
+                string.Format("INSERT INTO GradeCategory VALUES('{0}', '{1}', '{2}', null, '{3}');",
categories[i], classID, methods[i], percentages[i])
+                );
+          }
+
+          //commit all inserts to database
+          Database.commit();
+       }
+
       private bool checkingClassNames()
       {
         return txtClassName.Text.Equals("") || (chkClassMonday.Checked == false &&
chkClassTuesday.Checked == false) &&
```

Patchset 7/8: `SHA-1: 59911d4e7c435f62a3ffd89c098b354a3e06022a`

*** Extract Method clearArrays**

```
clearArrays method is added which clears all the arrays uses in the
method.
Extracting the method SaveClass is done and the extra lines and spaces
are deleted.
```

```
59911d4e7c435f62a3ffd89c098b354a3e06022a
 SourceCode/Calendar/AddClassForm.cs | 25 +++++++++----------------
 1 file changed, 9 insertions(+), 16 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index 0c4dada..329d22e 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -56,25 +56,19 @@ public AddClassForm()
       //saves the class information, returning true if successful and false if there was an error
       private bool saveClass() {

-         //ensure at least one day is checked
          if (checkingClassNames())
          {
             return raiseInvalidClassNameError();
          }

-         //grade categories required if class is not finished
          if (chkClassFinished.Checked == false && categories.Count == 0)
          {
             return raiseInvalidGradeCategory();
          }
-
-         //ensure start and end times are legal
          if (dtClassStartTime.Value.TimeOfDay > dtClassEndTime.Value.TimeOfDay)
          {
             return rasiInavlidTimeErorr();
          }
-
-         //set current grade to null unless the class is finished, upon which get the entered grade
          string currentGrade = "null";
          if (chkClassFinished.Checked == true)
          {
@@ -85,21 +79,12 @@ public AddClassForm()
             }
             currentGrade = "'" + cbFinalLetterGrade.Text + "'";
          }
-
-         //check if a semester has been selected
          string semesterIdValue = "null";
```

```
          if (cbSemester.SelectedIndex >= 0) {
             semesterIdValue = "'" + semesterId[cbSemester.SelectedIndex] + "'";
          }
-
-         //begin database transaction
          insertIntoDB(currentGrade, semesterIdValue);
-
-         //clear all arrays after updating database
-         categories.Clear();
-         percentages.Clear();
-         methods.Clear();
-
+          clearArrays();
          return true;
       }

@@ -205,6 +190,14 @@ private void insertIntoDB(string currentGrade, string semesterIdValue)
          Database.commit();
       }

+      private void clearArrays()
+      {
+         //clear all arrays after updating database
+         categories.Clear();
+         percentages.Clear();
+         methods.Clear();
+      }
+
       private bool checkingClassNames()
       {
          return txtClassName.Text.Equals("") || (chkClassMonday.Checked == false &&
chkClassTuesday.Checked == false) &&
```

Patchset 8/8: `SHA-1: 03c506a635879e72b8ffed017822603f32fbb0cb`

*** Moved the method SaveClass to the Method Region**

```
Moved the method SaveClass to the Method Region and deleted the
redundant checking with True value.
```

```
03c506a635879e72b8ffed017822603f32fbb0cb
 SourceCode/Calendar/AddClassForm.cs | 74 ++++++++++++++++++++----------------
 1 file changed, 38 insertions(+), 36 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index 329d22e..c68df35 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -52,42 +52,7 @@ public AddClassForm()
             cbFinalLetterGrade.SelectedIndex = -1;
         }
       }
-
-      //saves the class information, returning true if successful and false if there was an error
-      private bool saveClass() {
-
-         if (checkingClassNames())
-         {
-            return raiseInvalidClassNameError();
-         }
-
-         if (chkClassFinished.Checked == false && categories.Count == 0)
-         {
-            return raiseInvalidGradeCategory();
-         }
-         if (dtClassStartTime.Value.TimeOfDay > dtClassEndTime.Value.TimeOfDay)
-         {
-            return rasiInavlidTimeErorr();
-         }
-         string currentGrade = "null";
-         if (chkClassFinished.Checked == true)
-         {
-            //make sure user has selected a value
-            if (cbFinalLetterGrade.Equals(""))
-            {
-               return raiseInvalidLetterGradeError();
-            }
-            currentGrade = "'" + cbFinalLetterGrade.Text + "'";
-         }
-         string semesterIdValue = "null";
-         if (cbSemester.SelectedIndex >= 0) {
-            semesterIdValue = "'" + semesterId[cbSemester.SelectedIndex] + "'";
```

```
-         }
-         insertIntoDB(currentGrade, semesterIdValue);
-         clearArrays();
-         return true;
-      }
-
+
       private void btnSaveClass_Click(object sender, EventArgs e) {
          //save class information and close form if successful
          if (saveClass() == true) {
@@ -142,6 +107,43 @@ public AddClassForm()
       }

       #region Methods
+      //saves the class information, returning true if successful and false if there was an error
+      private bool saveClass()
+      {
+
+         if (checkingClassNames())
+         {
+            return raiseInvalidClassNameError();
+         }
+
+         if (chkClassFinished.Checked == false && categories.Count == 0)
+         {
+            return raiseInvalidGradeCategory();
+         }
+         if (dtClassStartTime.Value.TimeOfDay > dtClassEndTime.Value.TimeOfDay)
+         {
+            return rasiInavlidTimeErorr();
+         }
+         string currentGrade = "null";
+         if (chkClassFinished.Checked)
+         {
+            //make sure user has selected a value
+            if (cbFinalLetterGrade.Equals(""))
+            {
+               return raiseInvalidLetterGradeError();
+            }
+            currentGrade = "'" + cbFinalLetterGrade.Text + "'";
+         }
+         string semesterIdValue = "null";
+         if (cbSemester.SelectedIndex >= 0)
+         {
+            semesterIdValue = "'" + semesterId[cbSemester.SelectedIndex] + "'";
+         }
+         insertIntoDB(currentGrade, semesterIdValue);
+         clearArrays();
```

```
+           return true;
+       }
+
        private void insertIntoDB(string currentGrade, string semesterIdValue)
        {
            //begin database transaction
```

Refactoring 2

Patchset 1/3: `SHA-1: cc62ecb60edd01db5ac12c7accf90b37a29297c6`

*** Create GradeLetter class**

```
A new class is created in order to keep all the Grade Letters in it and
by adding the proper functions and properties in this class, access is
given to the rest part of the application to use this class whenever
needed. The grade letters were used as an array of string in the system
and they were scattered in different places in the project. By having
this separate class we'll increase the reuse, modifiability and
maintenance.
```

```
cc62ecb60edd01db5ac12c7accf90b37a29297c6
 SourceCode/Calendar/GradeLetter.cs | 47 ++++++++++++++++++++++++++++++++++++++++
 1 file changed, 47 insertions(+)

diff --git a/SourceCode/Calendar/GradeLetter.cs b/SourceCode/Calendar/GradeLetter.cs
new file mode 100644
index 0000000..cf5f8c7
--- /dev/null
+++ b/SourceCode/Calendar/GradeLetter.cs
@@ -0,0 +1,47 @@
+using System;
+using System.Collections.Generic;
+using System.Linq;
+using System.Text;
+
+namespace Planner
+{
+   public class GradeLetter
+   {
+      private string _code;
+      public static GradeLetter A = new GradeLetter("A");
+      public static GradeLetter A_Minus = new GradeLetter("A-");
+
+      public static GradeLetter B_Plus = new GradeLetter("B+");
+      public static GradeLetter B = new GradeLetter("B");
+      public static GradeLetter B_Minus = new GradeLetter("B-");
+
+
+      public static GradeLetter C_Plus = new GradeLetter("C+");
+      public static GradeLetter C = new GradeLetter("C");
+      public static GradeLetter C_Minus = new GradeLetter("C-");
+
+      public static GradeLetter D_Plus = new GradeLetter("D+");
+      public static GradeLetter D = new GradeLetter("D");
```

```
+       public static GradeLetter D_Minus = new GradeLetter("D-");
+
+
+       private static GradeLetter[] _values = { A, A_Minus, B_Plus, B, B_Minus, C_Plus, C, C_Minus,
D_Plus, D, D_Minus };
+
+       private GradeLetter(string code)
+       {
+          _code = code;
+       }
+       public string getCode()
+       {
+          return _code;
+       }
+       public static GradeLetter code(int arg)
+       {
+          return _values[arg];
+       }
+       public override string ToString()
+       {
+          return getCode();
+       }
+   }
+}
```

Patchset 2/3: `SHA-1: ef8530ebea211207c6cc4933389e30275cea5a34`

*** Create GradingScale**

```
A new class is created in order to keep all the Grade Scale in it and by
adding the proper functions and properties in this class, access is
given to the rest part of the application to use this class whenever
needed. The grade scales were used as an array of string in the system
and they were scattered in different places in the project. By having
this separate class we'll increase the reuse, modifiability and
maintenance.
```

```
ef8530ebea211207c6cc4933389e30275cea5a34
 SourceCode/Calendar/GradingScale.cs | 53 +++++++++++++++++++++++++++++++++++++
 1 file changed, 53 insertions(+)

diff --git a/SourceCode/Calendar/GradingScale.cs b/SourceCode/Calendar/GradingScale.cs
new file mode 100644
index 0000000..d303430
--- /dev/null
+++ b/SourceCode/Calendar/GradingScale.cs
@@ -0,0 +1,53 @@
+using System;
+using System.Collections.Generic;
+using System.Linq;
+using System.Text;
+
+namespace Planner
+{
+   public class GradingScale
+   {
+      private int _code;
+
+      public static GradingScale ninetyThree = new GradingScale(93);
+      public static GradingScale ninety = new GradingScale(90);
+
+      public static GradingScale eightySeven = new GradingScale(87);
+      public static GradingScale eightyThree = new GradingScale(83);
+      public static GradingScale eighty = new GradingScale(80);
+
+      public static GradingScale seventySeven = new GradingScale(77);
+      public static GradingScale seventyThree = new GradingScale(73);
+      public static GradingScale seventy = new GradingScale(70);
+
+      public static GradingScale sixtySeven = new GradingScale(67);
+      public static GradingScale sixtyThree = new GradingScale(63);
+      public static GradingScale sixty = new GradingScale(60);
+
+      private static GradingScale[] _values = { ninetyThree, ninety, eightySeven, eightyThree, eighty,
seventySeven,
+                                    seventyThree, seventy, sixtySeven, sixtyThree, sixty };
```

```
+       public static int Length = _values.Length;
+
+       private GradingScale(int code)
+       {
+          _code = code;
+       }
+       public static void changeValues(decimal[] newValues)
+       {
+          _values = newValues.Select(a => new GradingScale(Convert.ToInt32(a))).ToArray();
+       }
+       public int getCode()
+       {
+          return _code;
+       }
+
+       public static GradingScale code(int arg)
+       {
+          return _values[arg];
+       }
+       public int ToInteger()
+       {
+          return getCode();
+       }
+   }
+}
```

Patchset 3/3: SHA-1: 4bf9aeb3e76cc5bc89ee3fc057aa91767f850e55

*** Deleting the ScaleLetter and GradeLetter Arrays**

By adding the two new classes "" and "" we can easily delete the ScaleLetter and GradeLetter Arrays and use these classes in order to get the proper values and to modify the code accordingly.

```
4bf9aeb3e76cc5bc89ee3fc057aa91767f850e55
 SourceCode/Calendar/AddClassForm.cs | 14 ++++++++------
 1 file changed, 8 insertions(+), 6 deletions(-)

diff --git a/SourceCode/Calendar/AddClassForm.cs b/SourceCode/Calendar/AddClassForm.cs
index c68df35..1958807 100644
--- a/SourceCode/Calendar/AddClassForm.cs
+++ b/SourceCode/Calendar/AddClassForm.cs
@@ -12,12 +12,9 @@ namespace Planner
 {
    public partial class AddClassForm : Form
    {
+       #region Fields
        //grading scale information
        AddGradingScaleForm addGradingScale = new AddGradingScaleForm();
-       private decimal[] gradingScale = {93, 90, 87, 83, 80, 77, 73, 70, 67, 63, 60} ;
-       private string[] gradeLetter = {"A", "A-", "B+", "B", "B-", "C+", "C", "C-", "D+", "D", "D-"};
-
-       //add grade category information
        AddGradeCategoriesForm addGradeCategories = new AddGradeCategoriesForm();
        private List<string> categories = new List<string>();
        private List<double> percentages = new List<double>();
@@ -26,7 +23,9 @@ public partial class AddClassForm : Form
        //list of professors id
        private List<int> profId = new List<int>();
        private List<int> semesterId = new List<int>();
+       #endregion

+       #region Constructor
        public AddClassForm()
        {
          InitializeComponent();
@@ -40,7 +39,9 @@ public AddClassForm()
          //dynamically add possible semesters
          Util.addSemesters(cbSemester, semesterId, false);
        }
+       #endregion

+       #region Events
        //event handler for whether the class is finished
        private void chkClassFinished_CheckedChanged(object sender, EventArgs e) {
          //if class is finished allow user to enter final grade
```

```
@@ -95,7 +96,7 @@ public AddClassForm()
       private void lnkAddGradingScale_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e) {
          //show the grading scale form and once it is closed, get the grading scale information
          addGradingScale.ShowDialog();
-         gradingScale = addGradingScale.getGradingScale();
+          GradingScale.changeValues(addGradingScale.getGradingScale());
       }

       private void lnkAddGradingCategories_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)
{
@@ -105,6 +106,7 @@ public AddClassForm()
          percentages = addGradeCategories.getPercentages();
          methods = addGradeCategories.getMethods();
       }
+       #endregion

       #region Methods
       //saves the class information, returning true if successful and false if there was an error
@@ -171,7 +173,7 @@ private void insertIntoDB(string currentGrade, string semesterIdValue)
          {

             Database.modifyDatabase(
-               string.Format("INSERT INTO GradingScaleValue VALUES('{0}', '{1}', '{2}');",
gradeLetter[i].ToString(), classID, GradingScale.code(i).ToInteger())
+                string.Format("INSERT INTO GradingScaleValue VALUES('{0}', '{1}', '{2}');",
GradeLetter.code(i).ToString(), classID, GradingScale.code(i).ToInteger())
               );
         }
```