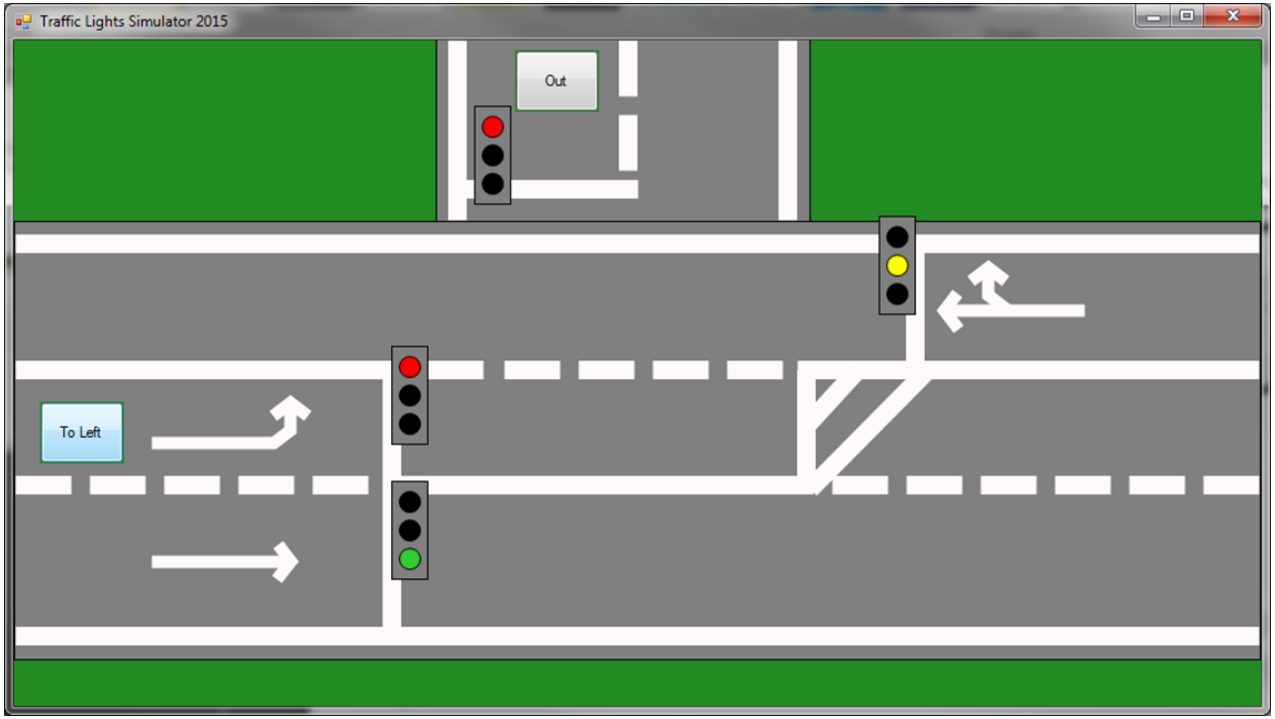
PROGRAMMIERUNG EINER AMPELSCHALTUNG IN VISUAL C#

Ziel:



ZIEL DER AMPELSCHALTUNG

Ziel:

Die Übungsaufgabe soll zur Einarbeitung in Visual C#, sowie das Entwicklertool Visual Studio und der Microsoft Entwicklerhilfe MSDN helfen.

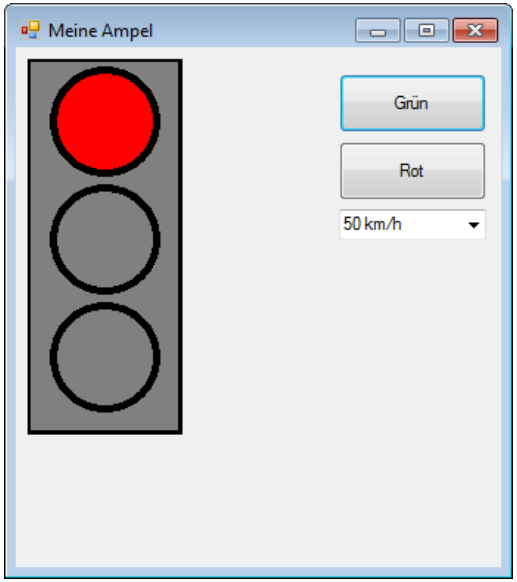
Die Ampelschaltung besteht aus mehreren eigenständigen Ampeln, die über Kontaktschleifen(Buttons) und eine zentrale Steuerung geschalten werden.

Durch die Entwicklung der Ampelschaltung (siehe Folie1) sollen Sie folgende Dinge erlernen:

- Erstellen von Grafiken und Benutzerkommunikation,
- Schreiben von Programmen und deren Strukturierung
- Testen und Fehler beheben

AUFGABE 1: ERSTE AMPEL DESIGNEN

1. Erstellen Sie eine Ampel in einem Form (Tipp: System.Drawing, OnPaint-Event)
2. Verwenden Sie zwei Buttons für eine kleine interne Schaltung:
   - Farben über eine Schaltung in rot und grün umschalten lassen nach Drücken von Buttons (Geschwindigkeit einstellbar)
     - Grün -> Rot; Gelbphase (nach StVO):
       - 3 s bei 50 km/h
       - 4 s bei 70 km/h
       - 5 s bei 100 km/h
     - Rot -> Grün; Rot-Gelb-Phase:
       - 1.5 Sekunden

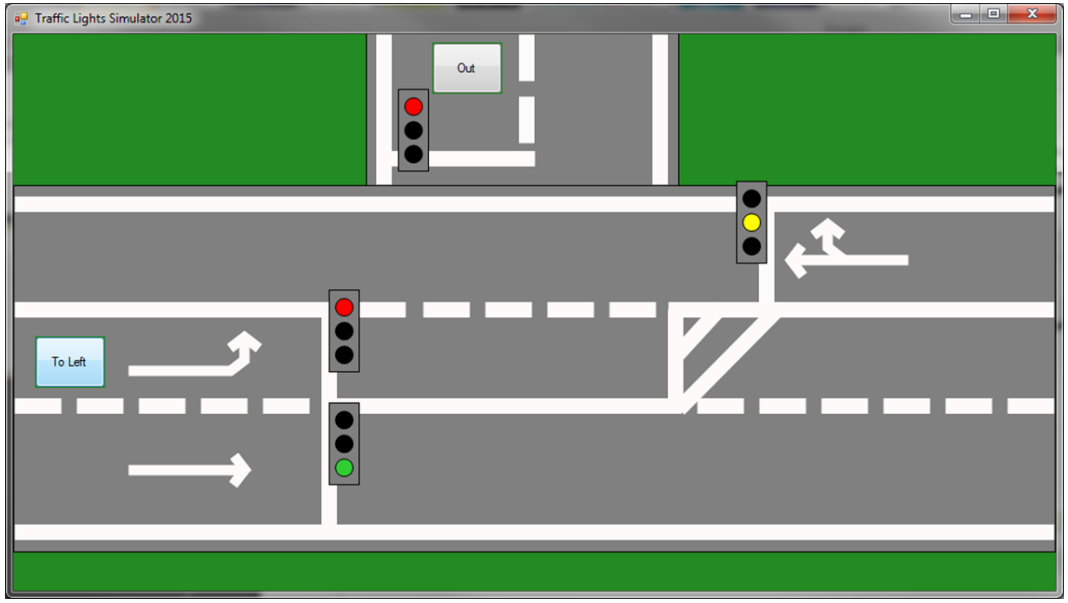


AUFGABE 2: AMPEL ALS BENUTZERSTEUERELEMENT

1. Wandeln Sie die Ampel in ein wiederverwendbares Benutzersteuerelement um (ohne die Buttons)
2. Lassen Sie mehrere Ampeln in einem Form unabhängig voneinander schalten
3. Achten Sie dabei darauf, die Zugriffe auf die Eigenschaften der Ampel richtig zu kapseln (siehe Kapselung, Properties, Zugriffsmodifizierer)
4. Achten Sie bei der Syntax auch jetzt schon darauf, sich an den EA-Standard zu halten (Q:\STT\Softwareentwicklung\Entwicklung mit C#\CodeDesign\EA Programming Guidelines for C# 2013-12-10.doc)

AUFGABE 3: GROßE SCHALTUNG

- Erstellen Sie ein Form mit einer T-Kreuzung mit Haupt- und Nebenstraße (s. Bild)
- Die Steuerung der Verschaltung der einzelnen Ampeln sollen Sie in einer separaten Klasse abbilden. Die Steuerung der Ampeln selbst soll im zuvor erstellten Benutzersteuerelement verbleiben.
- Eine genaue Beschreibung des Ablaufs finden Sie auf der nächsten Folie



ABLAUF

- Die Checkboxen (im Button-Design) simulieren Kontaktschleife und senden eine Anfrage auf Grün (Tipp: Events)
- Die Hauptstraße hat grundsätzlich grün, sofern es keine Anforderung von den Kontaktschleifen gibt.
- Kommt eine Anforderung über eine Kontaktschleife, so wird erst nach einer Wartezeit umgeschaltet. (Tipp: System.Timers) (Mindestgründauer der Hauptstraße)
- Haben beide Kontaktschleifen eine Anforderung gesendet, so werden auch beide Ampeln grün.
- Sendet eine Kontaktschleife ihre Anforderung erst, wenn die zur anderen Kontaktschleife gehörige Ampel schon grün ist, so wird die zugehörige Ampel in diesem Zyklus nicht mehr grün, sondern erst im nächsten.
- Die Hauptstraße (Linksabbieger) wird immer zuerst geräumt, erst dann die Nebenstraße (Tipp: Threads zur Ablaufsteuerung)
- Nur nötige Ampeln umschalten
- Neben-Ampeln schalten selbstständig auf Rot und Hauptampeln auf Grün

PROGRAMMOBERFLÄCHE

- Straße aus Rechtecken
- Fahrbahnmarkierungen aus Linien mit unterschiedlicher Dicke und verschiedenen Stilen
- Kontaktschleifen über Checkboxen im Button-Design

CHECKLISTE – SCHON BENUTZT?

- Hilfen für den Umgang mit Visual Studio
  - Nomenklatur
  - MSDN
  - Debugger (Breakpoint, Wert ändern & beobachten, Pausieren)
  - Dll erstellen
  - ErhardtAbt.dll einbinden, Trace verwenden

- Grafische Mittel
  - Button, CheckBox, RadioButton, ComboBox, Label, Textbox
  - Panel (mit DockStyles), GroupBox
  - UserControl

CHECKLISTE – SCHON BENUTZT?

- Programmbestandteile
  - Klassen/Interfaces
  - Enum
  - For, foreach (mit Liste), while
  - Thread, Timer, MessageBox
  - Try & catch
  - Private, public, protected
  - Eigenschaften, Members, Events
  - Eigenschaften von Controls anpassen (Schrift, Größe, Farbe, Enabled, Visible, ReadOnly)