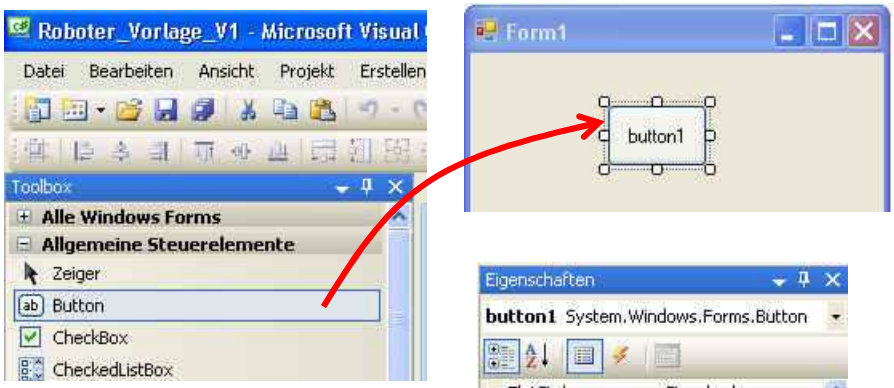
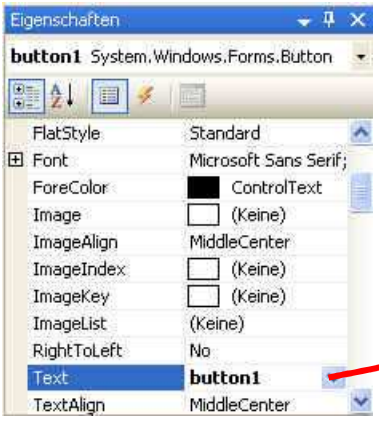
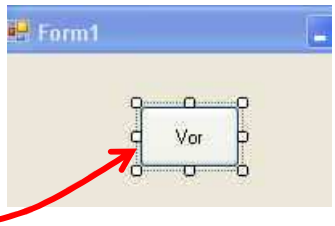
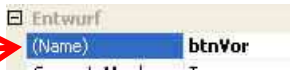
Im Folgenden wird gezeigt, wie der Roboter mit einfachen Steuerelementen über die PC-Maus gesteuert werden kann. Öffnen Sie dazu zunächst die Datei Form1.cs.
Holen Sie aus der Toolbox ein **Button-Steuerelement** auf die leere Form.



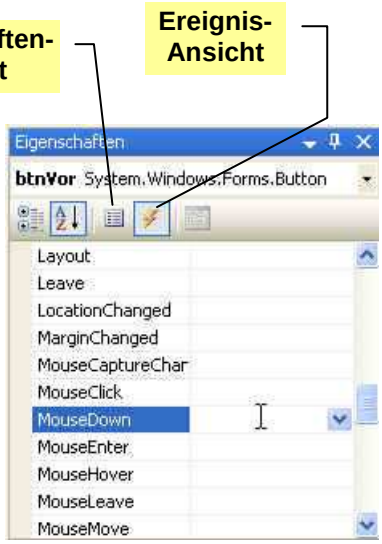
In den Eigenschaften des Button-Steuerelements können Sie die Text-Eigenschaft und den Namen des Steuerelements verändern.







Jetzt ist es an der Zeit, das Verhalten der Schaltflächen zu programmieren. In objektorientierten Sprachen bedeutet das, ein bestimmtes **Ereigniss** auszuwerten. Für das Button-Steuerelement gibt es sehr viele mögliche Ereignisse. Wir benötigen das **MouseDown-** und das **MouseUp-Ereignis**.



Klicken Sie die Schaltfläche *Vor* einmal an und aktivieren Sie die **Ereignisansicht** im **Eigenschaften-Fenster**.
Wählen Sie zunächst das MouseDown-Ereignis und Doppelklicken Sie in das zugehörige Eingabefeld.
Die Code-Ansicht mit dem Code-Grundgerüst wird geöffnet und der Rumpf der sogenannten **Ereignisprozedur** wird in den Code eingefügt. Der Name sollte nicht mehr geändert werden, da Visual Studio sonst durcheinader gerät.

```
private void btnVor_MouseDown(object sender, MouseEventArgs e)
{

}
```

Hier kann nun das Verhalten des Roboters programmiert werden, wenn die Vor-Taste nach unten gedrückt wird. In unserem Fall soll der Roboter mit Geschwindigkeitsstufe 8 vorwärts fahren. Hier nun die erste selbst geschriebene Codezeile:

```
private void btnVor_MouseDown(object sender, MouseEventArgs e)
{
    robo.motor(Mot.vor, 8);
}
```

Damit der Roboter beim Loslassen der Schaltfläche *Vor* stehenbleibt, muss der gleiche Vorgang für das **MouseUp-Ereignis** wiederholt werden. Mit dem Unterschied, dass der Roboter nun stehen bleiben soll. Erstellen Sie die btnVor_MauseUp-Prozedur!

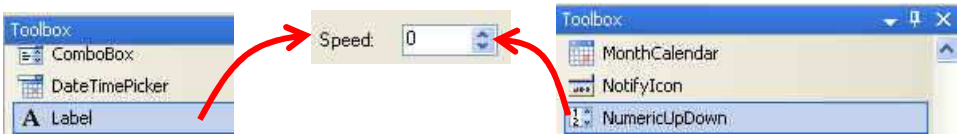
Fügen Sie weitere Steuer-Schaltflächen ein!
Erzeugen Sie die abgebildeten Schaltflächen mit den Namen:
btnVor
btnRechts
btnLinks
btnRueck

Anmerkung: Es ist äußerst hilfreich und macht einen Code besser lesbar, anstelle der von Visual Studio automatisch vergebenen Namen einen aussagekräftigen eigenen Namen zu wählen!

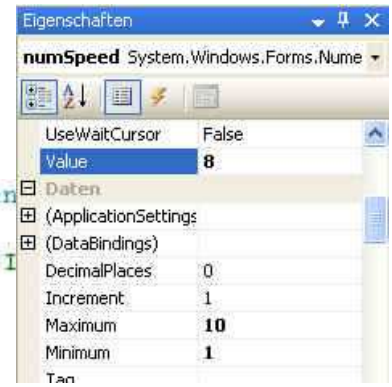
Programmieren Sie jeweils die Ereigniss-Prozeduren.
(Hinweis: Bei den MouseDown-Ereignissen muss der Motor immer stoppen. Damit man nicht 4 mal die gleiche Prozedur erzeugen muss, kann im Ereignis-Dialog dem MouseDown -Ereignis der 3 neuen Schaltflächen die bereits bestehende MouseDown-Prozedur der Vor-Schaltfläche zugewiesen werden!))

Testen Sie auch die anderen Bewegungsrichtungen und Geschwindigkeiten des Roboters.

Ergänzen wir nun das Programm mit einer komfortablen **Geschwindigkeitssteuerung**. Fügen Sie aus der Toolbox ein NumericUpDown- und ein Label-Steuerelement in die Form ein.



Machen Sie im **Eigenschafts-Dialog** des NumericUpDown-Steuerelements die dargestellten Änderungen.



Gehen Sie in die Code-Ansicht und ändern Sie die btnVor_MouseDown-Prozedur wie abgebildet ab.

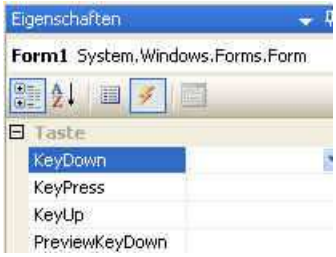
```
private void btnVor_MouseDown(object sender, MouseEven
{
    int speed;          // Variable vom Ganzzahl-Typ I
    speed = Convert.ToInt16(numSpeed.Value);

    robo.motor(Mot.vor, speed);
}
```

Erläuterung: wird zunächst eine Ganzzahl-Variable (**Integer**) angelegt. Ihr wird dann der Wert des NumericUpDown-Steuerelements (**Value**-Eigenschaft) zugewiesen. Dazu muss der Value-Wert aber noch in eine Integer-Variable umgewandelt werden. Die Methoden von **System.Convert.** können eine Vielzahl von Typkonvertierungen durchführen.

Ergänzen Sie jetzt auch die anderen 3 MouseDown-Prozeduren.

Als nächsten Schritt ergänzen wir eine Tastatursteuerung mit den Cursortasten. Dazu muss Form1 in der Formularansicht angeklickt werden, so dass die **Formularereignisse** im Eigenschaftenfenster bearbeitet werden können.
Fügen Sie dem Code das **Form1_KeyDown**-Ereignis hinzu:




```
private void Form1_KeyDown(object sender, KeyEventArgs e)
{
    int speed;          // Variable vom Ganzzahl-Typ Integer
    speed = Convert.ToInt16(numSpeed.Value);

    if (e.KeyValue == 38)
        robo.motor(Mot.vor, speed);
    else if (e.KeyValue == 39)
        robo.motor(Mot.rechts, speed);
    else if (e.KeyValue == 40)
        robo.motor(Mot.rueck, speed);
    else if (e.KeyValue == 37)
        robo.motor(Mot.links, speed);
}
```

Der Übergabeparameter **e** des KeyDown-Ereignisses enthält unter anderem auch die Tastaturnummer (**e.KeyValue**-Eigenschaft) der betätigten Taste. Mit der **if ... else if ...** -Konstruktion wird abhängig vom Tastaturcode ein jeweils anderer Roboter-Befehl ausgeführt.

Damit das Form auf Tastendrücke reagiert, muss allerdings noch die Eigenschaft **KeyPreview** des Formulars aktiviert werden.

Fügen Sie noch eine KeyUp-Ereignisprozedur hinzu!
Eventuell gibts mit den Cursor-Tasten Probleme. Versuchen Sie doch die Steuerung über die Tasten w,a,s,d zu realisieren. Überlegen Sie sich ein kurzes Programm, um die Tastaturcodes dieser Tasten herauszufinden!
Eine andere elegante Möglichkeit für die Auswertung der Variablen e.KeyValue ist die **switch... case...**- Konstruktion:

```
switch (e.KeyValue)
{
    case 38: robo.motor(Mot.vor, speed); break;
    case 39: robo.motor(Mot.rechts, speed); break;
    case 40: robo.motor(Mot.rueck, speed); break;
    case 37: robo.motor(Mot.links, speed); break;
    default: robo.motor(Mot.stopp, speed); break;
}
```

Machen Sie sich die Wirkungsweise dieser Konstruktion klar!