Einführung in das Programmieren mit Visual Studio 2013

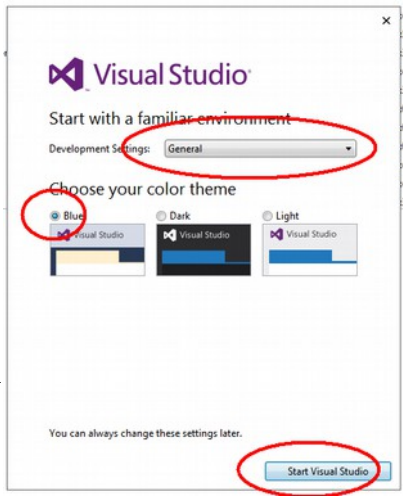
Um ein eigenes Programm auf einem PC zum Laufen zu bringen, sind drei grundlegende Schritte erforderlich.

1. Das Programm muss erdacht und strukturiert werden. Diese Vorbereitung findet zu Hause statt und liefert einen Entwurf auf Papier.
2. Eingabe des Quellcodes in einen Editor, Speicherung auf dem PC.
3. Übersetzung des Quellcodes in Binärcode für den aktuell verwendeten PC und Ausführung des Programms.

Um die Schritte 2 und 3 kümmert sich auf modernen Rechnern eine Entwicklungsumgebung, die sogenannte IDE (Integrated Development Enviroment). Diese gibt es von unterschiedlichen Herstellern mit unterschiedlichen Philosophien und in vielen Geschmacksrichtungen, von Open Source bis unbezahlbar. Wir verwenden in diesem Praktikum die IDE „Visual Studio 2013“ von Microsoft, das im Rahmen der DreamSpark-Verträge auch von Studierenden kostenlos herunter geladen und für private Zwecke genutzt werden kann.

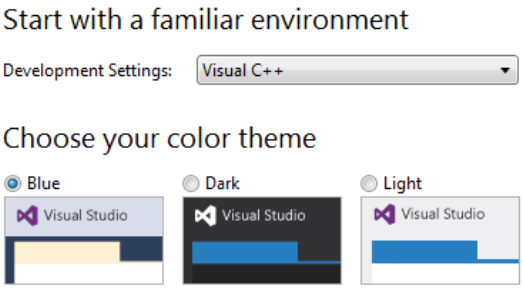
Gestartet wird diese IDE am einfachsten durch einen Doppelklick mit der linken Maustaste auf dieses Symbol auf dem Desktop (so nennt man die Ansicht, die Windows standardmässig anbietet)
Wundern Sie sich nicht, wenn es jetzt etwas langsamer zugeht, diese IDE ist groß, mächtig und umfangreich. Und sie ist in Englisch!

Wenn die IDE zum ersten Mal gestartet wird, müssen Sie sich entscheiden, welches Design sie verwenden möchten.



Stellen Sie zunächst die Development Settings auf „Visual C++“ ein, wählen Sie dann ein Design nach persönlichem Geschmack und klicken Sie dann auf „Start Visual Studio“.

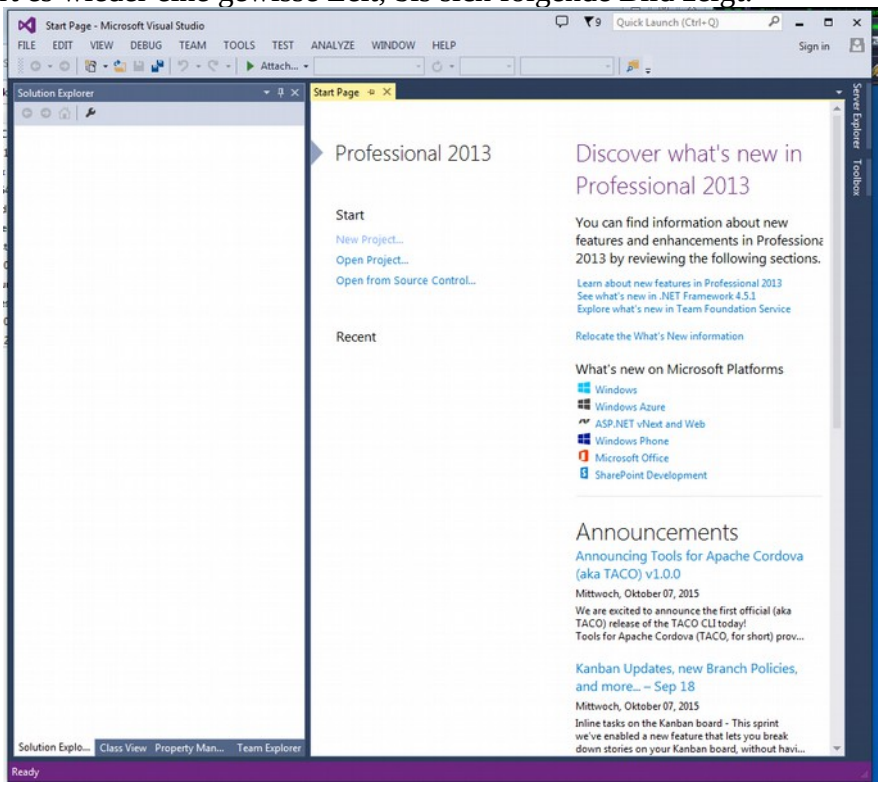
Ihre Einstellungen sollten eine gewisse Ähnlichkeit mit diesen hier haben:



Dabei ist die Wahl des „color scheme“ (Farbschema) eine reine Geschmacksfrage.

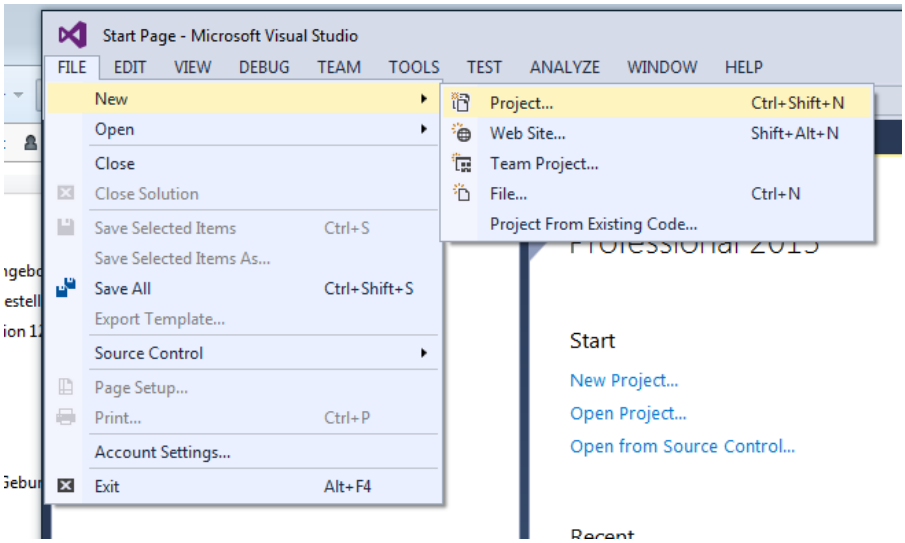
Diese Einstellungen merkt sich die IDE und sie werden später mit diesen Fragen nicht mehr belästigt. Beachten Sie aber: auch wenn da steht, dass diese Einstellungen später jederzeit geändert werden können, es ist sehr viel einfacher und schneller es hier einmal kurz richtig einzustellen, als später in vielen Menüs diverse Schrauben zu drehen.

Auch jetzt dauert es wieder eine gewisse Zeit, bis sich folgende Bild zeigt:



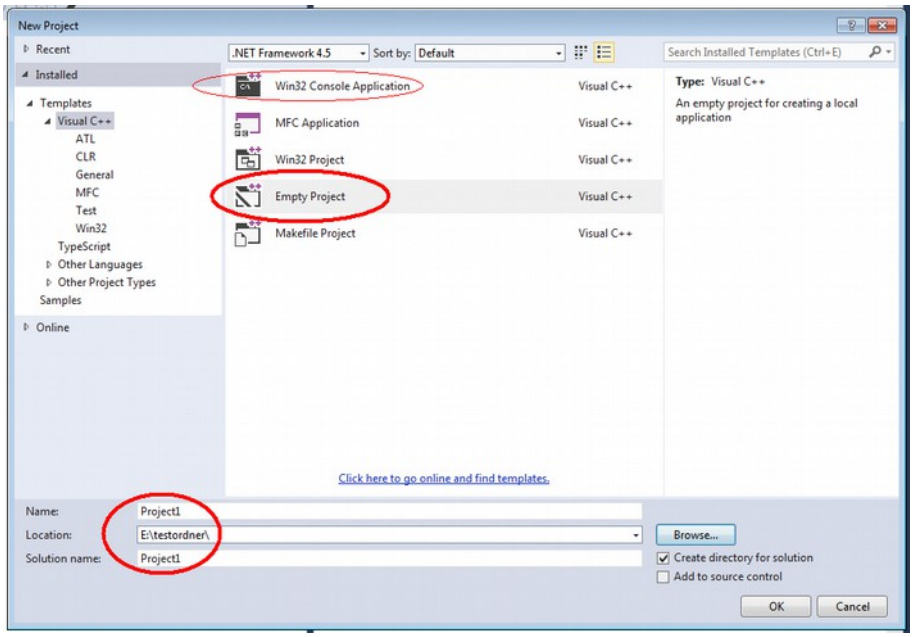
Das ist zunächst wenig hilfreich, die Meldungen in der rechten Spalte sind interessant, wenn Sie professionell Software entwickeln und werden nach aktuellem Anlass von Microsoft nachgeladen. Uns genügt es zu wissen, dass jetzt die eigentliche Arbeit beginnen kann.

Visual Studio ist dafür ausgelegt, auch große Programme zu entwickeln, auch Word und Excel werden damit erstellt. Es verfügt also über viele Fähigkeiten, die wir für unsere Zwecke nicht benötigen. Daher beschränke ich mich hier auch darauf, die Dinge zu zeigen und zu erläutern, die für das Praktikum erforderlich sind.



Jedes Programm, auch das allerkleinste, wird in Visual Studio als „Projekt“ (engl. Project) gesehen, wenn wir also ein (neues) Programm schreiben wollen, müssen wir als ersten Schritt immer ein

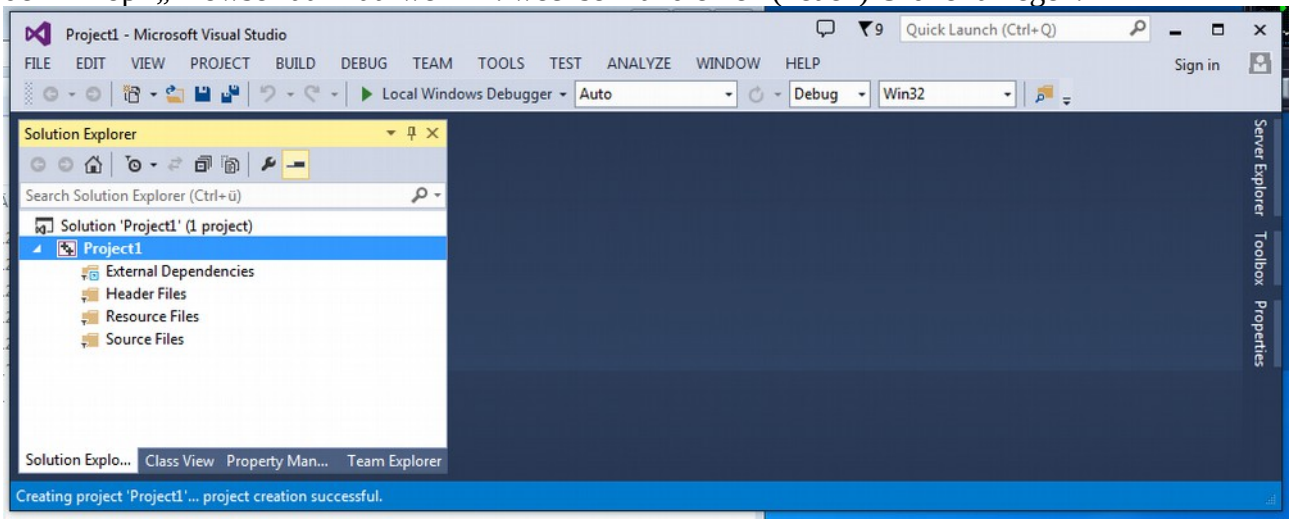
neues Projekt (new project) anlegen. Der dazu gehörige Menüpunkt versteckt sich unter „File“ (Datei). Prompt erhalten wir das nächste Menü:



Hier müssen wir jetzt wieder Einstellungen vornehmen, diese gelten für unser gesamte Projekt und müssen also für jedes neue Programm auch wieder neu vorgenommen werden.
Im oberen Bereich ist die Art des Projektes auszuwählen. Im Regelfall können wir mit einem „Empty Project“ (leeres Projekt) beginnen. (Manche Dozenten empfehlen auch einen „Win 32 Console Application“ (Windows Konsolenprogramm), damit gibt man aber einen Teil der Kontrolle an die IDE ab und baut sich eine potentielle Stolperfalle für später ein.)

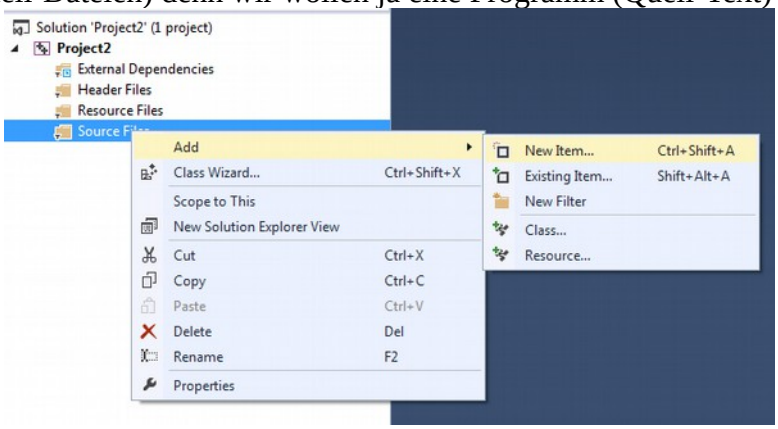
Im unteren Bereich bekommt unser Projekt einen Namen, die IDE schlägt hier meist etwas mehr oder weniger Sinnvolles vor. Das kann man übernehmen, es spricht aber einiges dafür, einen sprechenden Namen zu wählen.

In der mittleren Zeile wird festgelegt, wo unser Projekt abgespeichert wird (Location). Auf ihrem privaten Rechner können Sie die Voreinstellung einfach stehen lassen, auf den Rechnern des Fachbereichs Informatik erstellen Sie bitte einen eigenen Ordner auf Laufwerk D: Dazu einfach mit dem Knopf „Browse“ auf Laufwerk D: wechseln und einen (neuen) Ordner anlegen.

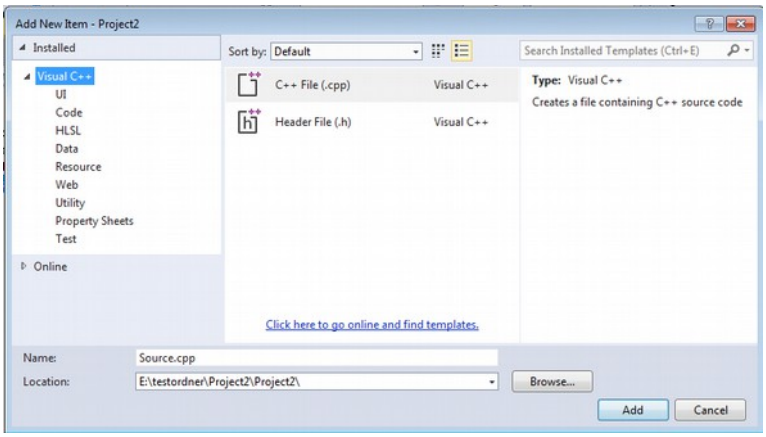


Unser neues Projekt wurde erfolgreich angelegt. Nur Inhalt hat es immer noch keinen! Den müssen wir leider von Hand selbst hinzufügen.[1]

Ein Projekt besteht mindestens aus einer Datei (File), in späteren Übungen auch gerne mal aus mehreren, die zusammen dann das Projekt bilden. Diese Datei müssen wir unserem Projekt jetzt einmal hinzufügen, das geht am besten durch einen Klick mit der **rechten** Maustaste auf den Ordner „Source Files“, (Quell-Dateien) denn wir wollen ja eine Programm (Quell-Text) eingeben:



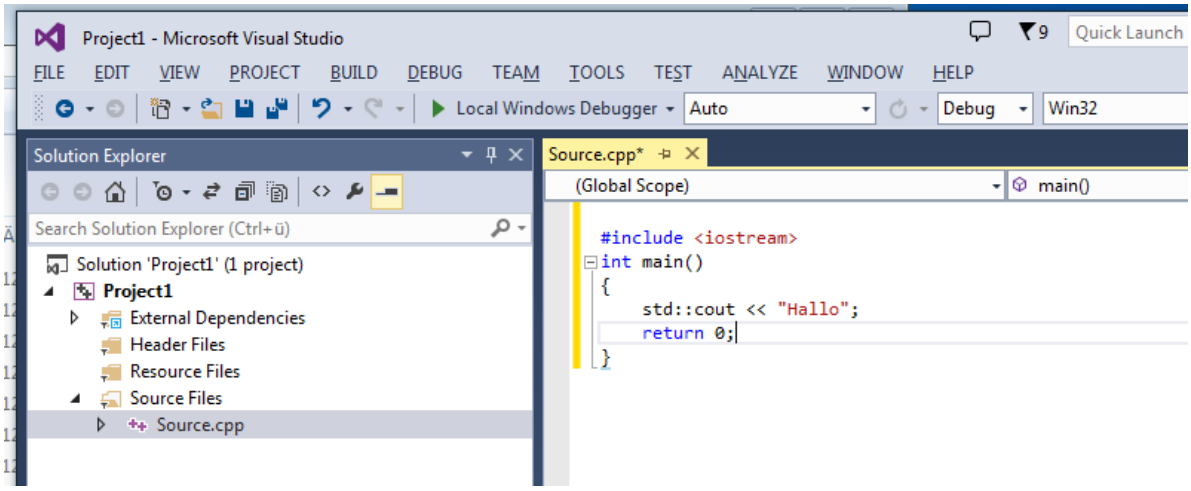
Natürlich muss man der IDE noch mitteilen, um welche Art Datei es sich handelt, wir kennen anfangs nur cpp—Dateien, also:



Im oberen rechten Bereich wählen wir „C++File(.cpp)“ aus, das ist ein Dateityp für allgemeine C++-Programme. Im unteren Bereich wird ein Name für die Datei vorgeschlagen, den kann man übernehmen, kann ihn aber auch sinnvoll ändern. Und unter „Location“ steht der Ordner, den wir vorher beim Anlegen des Projektes angegeben haben (er ist länger als dort angegeben, weil VS alle Dateien in Ordnern strukturiert).

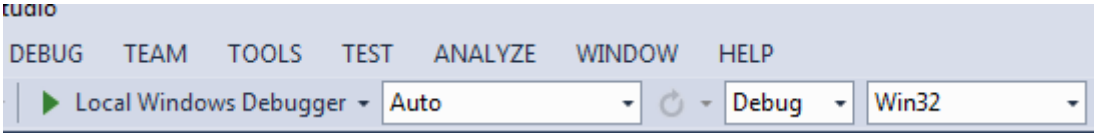
Haben wir alle Angaben gemacht, sieht das Ergebnis so aus:

1 Hier unterscheidet sich das weitere Vorgehen in Abhängigkeit des gewählten Projekt-Typs. Bei Win32 Console Application werden automatisch die notwendigen Dateien (4 Stück) angelegt und ins Projekt eingefügt

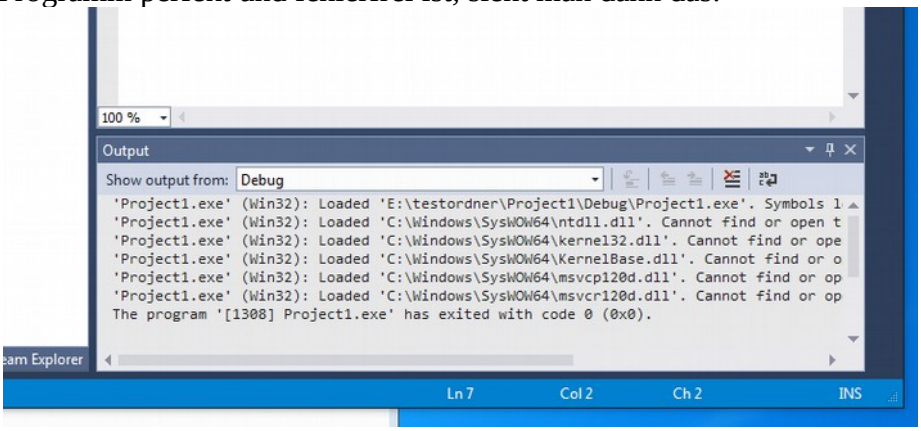


Rechts ist ein neues, leeres Fenster mit viel Platz zum Eingeben unseres Programmtetxtes. Als Beispiel hier schon mal das übliche Programm, welches Informatiker immer als erstes schreiben, „Hello World!“. Man sieht, die IDE hilft mit farbigen Hervorhebungen, bestimmte Wörter (Schlüsselworte) zu identifizieren, kann zwischen Text in Anführungszeichen und Programmanweisungen unterscheiden und einiges mehr. Beim Tippen bekommt man z.B. Vorschläge, wie der Text weitergehen könnte und kann den einfach aus einer Liste auswählen. Diese und weitere Annehmlichkeiten einer IDE werden Sie aber ganz schnell schätzen lernen und im Laufe der Übungen selbst herausfinden.

Damit ist der erste Teil unserer Übung, das Eingeben des Quelltextes abgeschlossen. Schritt 2 folgt, das Übersetzen dieses Textes in ein Binärprogramm und das Ausführen dieses Programms. Dabei laufen im Hintergrund der IDE viele Einzelschritte ab, deren Beschreibung hier aber zu weit führen

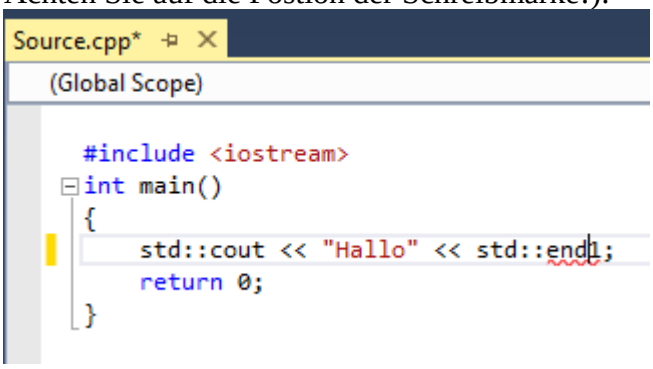


würde. Wir begnügen uns damit, auf einen Knopf zu drücken, der das alles für uns erledigt:Das kleine grüne Dreieck erledigt diesen Job zuverlässig, wenn in den Feldern daneben keiner Änderungen vorgenommen hat.
Und da unser Programm perfekt und fehlerfrei ist, sieht man dann das:

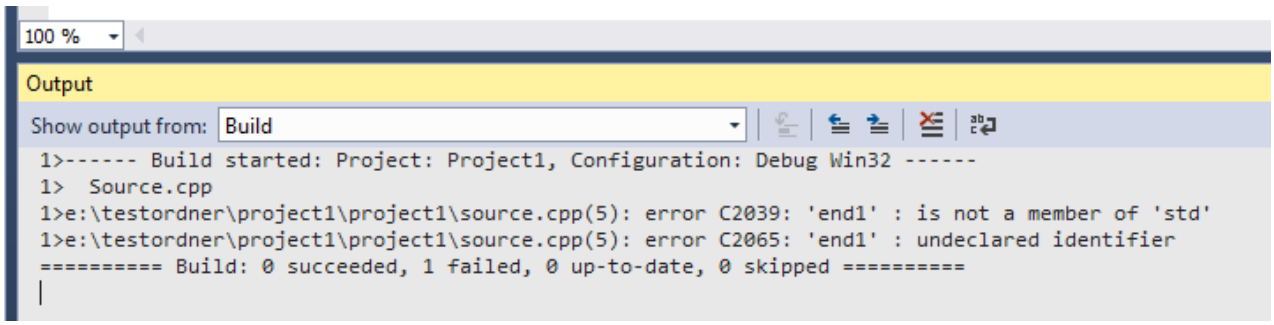


Wichtig ist die letzte Zeile: „The program ..was exited with code 0“, unser Programm wurde also einwandfrei (code 0) ausgeführt.
Leider sieht die Wirklichkeit anders aus, kein Programm ist einfach und kurz genug, um sofort

fehlerfrei zu sein. Wir fügen also in unser fehlerfreies Programm noch etwas ein (finden Sie die Stelle auf Anhieb? Achten Sie auf die Postion der Schreibmarke!):

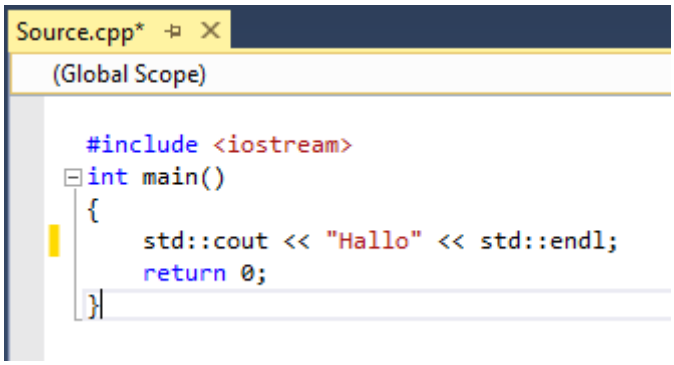


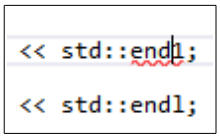
Jetzt erhalten wir andere Meldungen, nachdem wir den Befehl zum Übersetzen (Compile) gegeben haben:



Gleich zwei Fehlermeldungen produzieren unsere paar eingefügten Buchstaben. Es muss etwas mit dem „end1“ zu tun haben. Um den Fehler in längeren Programmen schneller zu finden wird vor der Fehlernummer in Klammern die Zeilennummer angezeigt, in der das Problem auftrat. Fangen Sie jetzt aber nicht an, die Zeilen im Eingabefenster zu zählen, das wird bei längeren Programmen zu einer abendfüllenden Aufgabe[2]. Klicken Sie einfach mit der linken Maustaste doppelt auf die Fehlermeldung, schon steht der Cursor (die Schreibmarke) in der richtigen Zeile. Das klapp bei Projekten mit mehreren Dateien sogar, wenn Fehler in unterschiedlichen Dateien stecken, der IDE sei Dank.

Den Fehler beseitigen müssen wir aber immer noch selbst, das kann die IDE nicht. Das sieht dann so aus:





2 Sie können auch gerne die Zeilennummern einschalten. Wählen Sie dazu im Menü TOOLS → Options → Text Editor → C/C++ → General „Line Numbers“

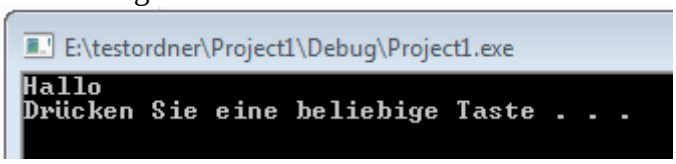
Sie sehen keinen Unterschied? Achten Sie auf das Detailsbild rechts! Die obere Zeile ist fehlerhaft, die untere Zeile richtig. Die IDE hat das Problem schon längst erkannt und deshalb eine rote Wellenlinie unter dem fehlerhaften Begriff gemalt, das sollte einen sofort stutzig machen. Und wenn man dann ganz genau hinsieht, kann man erkennen, dass die obere Zeile auf die Ziffer „1“ endet, die untere auf den Kleinbuchstaben „l“.

Wir wissen jetzt, wie wir ein Programm eingeben, übersetzen und ausführen. Dabei wird eingeben und ändern unsere längste Zeit beanspruchen und sich ständig wiederholen, bis alles funktioniert.

Stellt sich noch eine weitere Frage: wo sind denn die Ausgaben des Programms?
Wenn man ganz genau aufpasst, kann man ein kurzes Aufblitzen einer schwarzen Fläche irgendwo auf dem Bildschirm sehen, aber erkennen kann man darauf nichts. Wir müssen dafür sorgen, dass dieses Ausgabefenster solange offen bleibt, dass man alles drin lesen kann.
Das erreichen wir durch das Einfügen von Zeile 6:

```
1
2   #include <iostream>
3   int main()
4   {
5       std::cout << "Hallo" << std::endl;
6       system("Pause");
7       return 0;
8   }
```

Und können das Ergebnis endlich genießen:



Damit wissen Sie alles Notwendige, um selbst Programme zu erstellen und zu testen. Viel Erfolg!