



# Arduino Entwicklungsumgebung Sprint 4

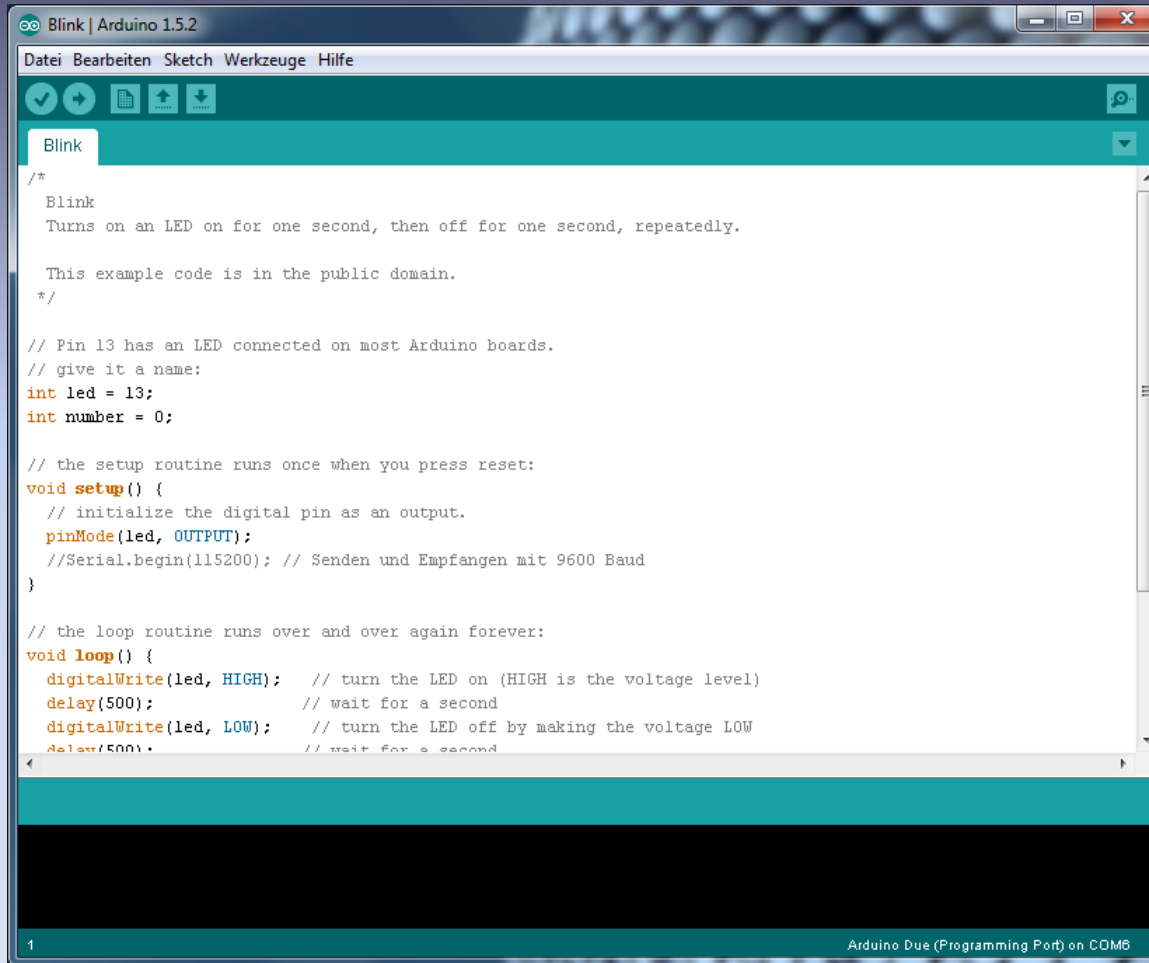
# Entwicklungsumgebung Arduino

Umgebung zur Entwicklung der Software

- Software Entwicklungsumgebung installieren und konfigurieren, damit Software für das Arduino Board erstellt und auf die Hardware geladen werden kann.

# Arduino Standard

Arduino 1.5.2, schlechte Entwicklungs- und Debugging-Funktionen



The screenshot shows the Arduino IDE 1.5.2 interface. The title bar reads "Blink | Arduino 1.5.2". The menu bar includes "Datei", "Bearbeiten", "Sketch", "Werkzeuge", and "Hilfe". The toolbar contains icons for opening files, saving, uploading, and downloading. The main editor area displays the following code:

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
  */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;
int number = 0;

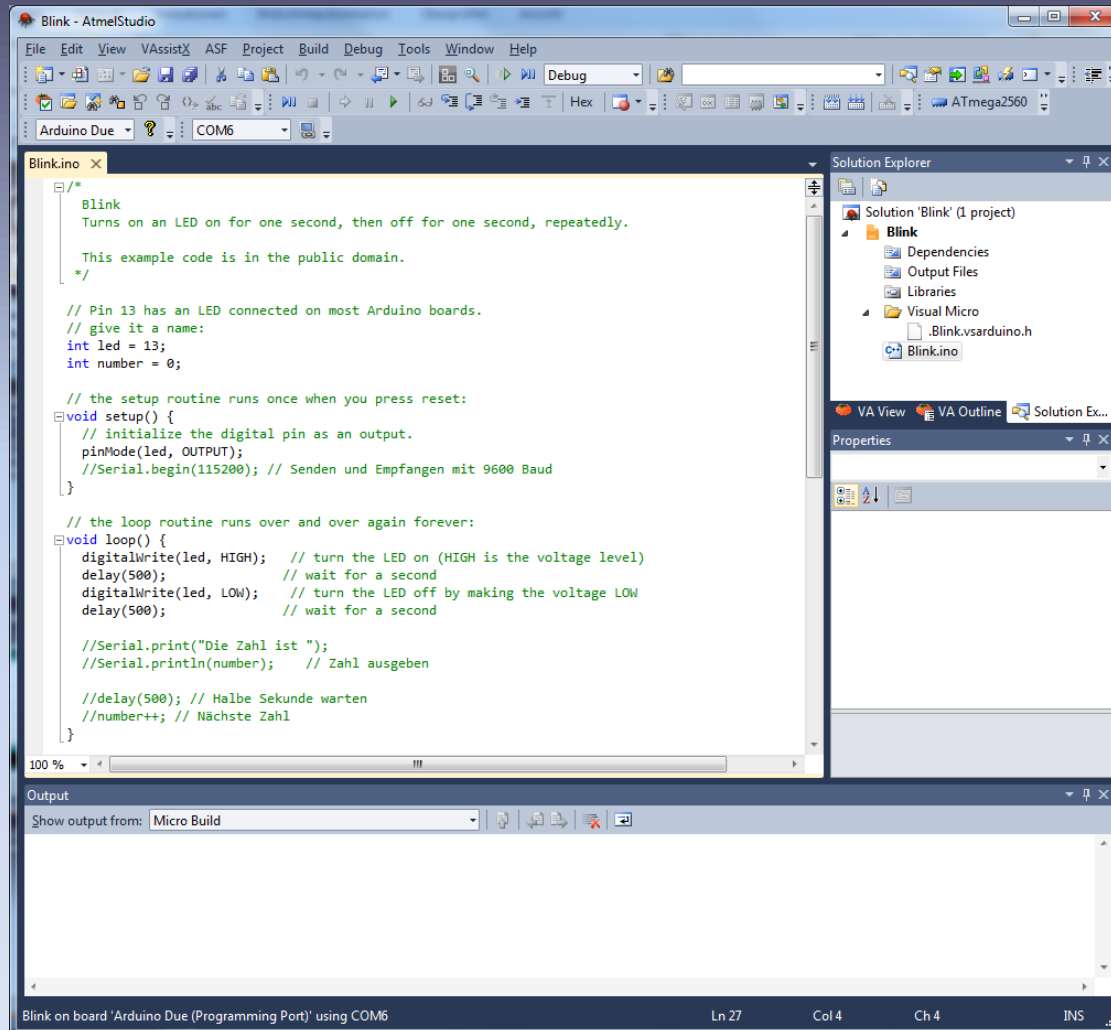
// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
  //Serial.begin(115200); // Senden und Empfangen mit 9600 Baud
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(500); // wait for a second
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
  delay(500); // wait for a second
}
```

The status bar at the bottom indicates "1" on the left and "Arduino Due (Programming Port) on COM8" on the right.

# Atmel Studio 6.1

Basiert auf Microsoft Visual Studio, Professionelle Entwicklungsumgebung



The screenshot displays the Atmel Studio 6.1 interface. The main window shows the source code for a project named 'Blinkino'. The code is written in C++ and is designed to blink an LED on an Arduino Due board. The code includes comments in German explaining the setup and loop functions. The setup function initializes pin 13 as an output and sets the serial port to 9600 baud. The loop function turns the LED on for one second, then off for one second, and increments a counter. The status bar at the bottom indicates the code is being compiled on an 'Arduino Due (Programming Port)' using COM6.

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;
int number = 0;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
  //Serial.begin(115200); // Senden und Empfangen mit 9600 Baud
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(500); // wait for a second
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
  delay(500); // wait for a second

  //Serial.print("Die Zahl ist ");
  //Serial.println(number); // Zahl ausgeben

  //delay(500); // Halbe Sekunde warten
  //number++; // Nächste Zahl
}

Blink on board 'Arduino Due (Programming Port)' using COM6
```

# Installation Atmel Studio 6.1

## Inkl. Visual Micro Plugin für Arduino

1. Arduino 1.5.2 installieren resp. kopieren (am besten nach c:\arduino-1.5.2\)
2. Atmel Studio 6.1 installieren (Freeware)
3. Visual Micro Plugin installieren (Freeware, ausser Debug-Funktion ist 30 Tage Trial)  
<http://www.visualmicro.com/page/Arduino-for-Atmel-Studio.aspx>
4. Visual Micro Patch installieren:  
<http://www.visualmicro.com/forums/YaBB.pl?num=1367685306>
5. Atmel Studio 6.1 starten und Pfad für Arduino Umgebung eingeben (c:\arduino-1.5.2\  
Hier hatte ich Probleme da das Plugin Mühe hatte mit dem deutschen Pfad c:\Programme\arduino-1.5.2\, daher nach c: \arduino-1.5.2\  
kopieren.
6. Arduino Due auswählen und COM X

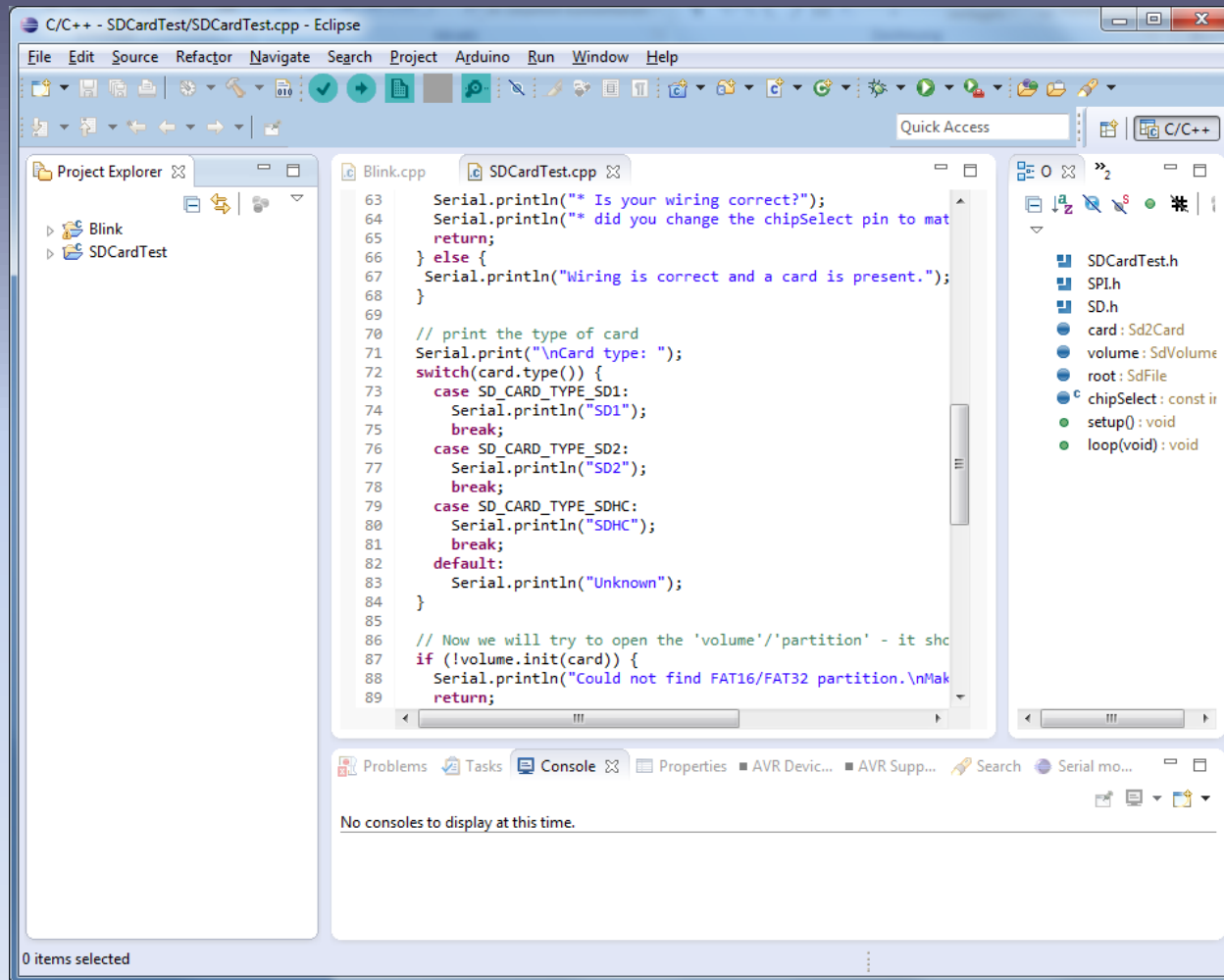
# Erkenntnisse zu Atmel Studio 6.1

Inkl. Visual Micro Plugin für Arduino

1. Installation geht einfach
2. Jedoch ist Visual Micro Plugin noch nicht 100% kompatibel mit Arduino 1.5.2 Version, da diese im Beta-Status ist
3. Probleme mit SD-Karten Integration
4. Alternative Eclipse C++ Juno mit Arduino Plugin

# Eclipse C++ Juno mit Arduino Plugin

Professionelle Open-Source Entwicklungsumgebung



# Installation Eclipse C++ Juno mit Arduino Plugin

## Teil 1/2

Dabei gilt es besonders folgendes zu beachten:

- Es muss zwingend Eclipse **Juno** verwendet werden
- Eclipse CDT muss installiert sein
- dann Anleitung (siehe nächste Seite oder Link unten) befolgen
  - Punkt 3 in der Anleitung: das Archiv ist ein mit gzip komprimiertes Tar-Archiv (\*.tar.gz), kann mit 7zip (in zwei Schritten) entpackt werden
  - Punkt 9 in der Anleitung “Group items by category” ist wichtig, sonst wird nichts angezeigt
- <http://trippylighting.com/teensy-arduino-ect/arduino-eclipse-plugin/arduino-eclipse-plugin-installation/>



# Installation Eclipse C++ Juno mit Arduino Plugin

## Teil 2/2

1. Download and install [Eclipse IDE for C/C++ Developers](#) (It needs to be Juno!)
2. Download and install Arduino 1.5.2 (beta).
  - Note: Avoid spaces in the Arduino.app file name. This is a confirmed bug that may be removed in future versions. While compiling code works fine, when uploading it to the Teensy, the teensy loader cannot be found.
3. Start Eclipse. The first time it will ask you to select a workspace. For an Arduino work environment I'd suggest you select the place where you have all your Arduino Projects/Sketches. You can always change to another workspace if you have several of such directories.
4. From the Eclipse menu bar Eclipse select "Help" → "Install new software".
5. Select "Add" to add the download site for the Arduino Eclipse plugin : <http://www.baeyens.it/eclipse/V2>
6. Eclipse will show which versions are available (Make sure to keep the "Group items by category" unchecked). Select the 2.1.0.4 version and follow the rest of the installation steps. The Plugin installs the usual Arduino button icons into the menu bar.
7. Now that the plugin is installed, from the Eclipse menu bar select "Eclipse" → "Preferences". In the preferences pane select "Arduino" and fill in the appropriate fields for "Arduino IDE path" and "Private library path". The example below shows where these directories located on my machine. Obviously this may be different on each users machine.
8. D.O.N.E. This concludes the installation. And now you can start with your [first project using the Arduino Eclipse Plugin](#)