# Arduino und ESP Display, drahtlos ohne Cloud

|  |  |
| --- | --- |
| Material | Kosten |
| ESP32 oder Arduino mit Bluetooth modul | ab ca. 5Eur |

Mikrokontroller sind leicht zu programmieren, aber sobald man eine Bildschirmanzeige benötigt oder Taster anschließen möchte, steigt der Aufwand bei der Verkabelung schnell an. Da kommt ein kabelloser Touchscreen, der sowohl Messwerte anzeigen kann als auch Dateneingaben erlaubt sehr gelegen. RemoteXY macht aus einem Handy oder Tablet Computer genau das. Es erlaubt eine sehr einfache Konfiguration des User Interface und bietet verschiedene Kommunikationsmöglichkeiten zu unterschiedlichen Mikrocontrollern. Am interessantesten für meine Anwendungen ist die direkte Kommunikation per Bluetooth. Anders als beispielsweise bei Blynk benötigt man dafür nicht zwingend eine Cloud Anbindung, die Daten können ohne Cloud oder Wifi direkt per Bluetooth zwischen dem Handy und dem ESP ausgetauscht werden. Alternativ ist eine Wifi-Kommunikation über das lokale Netz oder auch eine Cloud-Kommunikation möglich.

## Grundprinzip und Basiskonfiguration

Das Prinzip von RemoteXY ist schnell erklärt, über den Editor auf <https://remotexy.com/en/editor/> konfiguriert man zunächst auf welche Art kommuniziert werden soll, zur Auswahl stehen Bluetooth, Wifi, USB, Ethernet und Cloud-server. Anschließend wählt man einen der unterstützten Mikrokontroller (diverse Boards werden unterstützt, darunter Arduino, ESP8266 und ESP32). Dann folgt die Wahl des Kommunikationsmoduls, z.B. Integrated Bluetooth beim ESP32 oder ein externes HM-10 Modul für die BLE-Verbindung mit einem Arduino.

A screenshot of a computer

Description automatically generated

Figure 1 Auswahl von Mikrokontroller und Kommunikation (Im Beispiel: Bluetooth, ESP32 und Arduino IDE)

Ist die Hardware und die Kommunikation definiert, beginnt man das Display mit Schalter- und Anzeigeelementen zu gestalten. Für jedes Element kann ein Variabel-Namen definiert werden, den man später im Programmcode verwendet.

Über den Button „Get Source Code“ erhält man dann das Skelett des Programms, es enthält bereits die Anweisung zum Import aller relevanten Libraries, das Layout des Displays in Form eines Arrays und die Variablen in einer Variable-Struktur.

Auf dem Handy muss man die App RemoteXY installieren, die sowohl für Android als auch für iOS verfügbar ist. Startet man die App, kann über das Plus-Zeichen ein neues Gerät hinzugefügt werden, je nach Kommunikationsmethode muss man beispielsweise einen Access-Point oder eine IP-Adresse angeben. Im Falle einer Bluetooth-Verbindung muss der Mikrokontroller eingeschaltet werden, die App scannt dann die Bluetooth-Signale des Mikrocontrollers und kann diesen per Klick hinzufügen.

## Hello World

Das erste Programmbeispiel enthält nur einen Button und ein Textfeld auf der App und benötigt keine Verdrahtung des Microkontrollers. Sobald der Mikrocontroller eingeschaltet wird, soll ein Zähler jede Sekunde 1 hochzählen und den aktuellen Wert im Textfeld anzeigen. Wird der Button gedrückt, soll der Zähler zurückgesetzt werden.

Zunächst wird das User Interface über <https://remotexy.com/en/editor/> editiert. Aus den Controls & Indication Menüs auf der linken Seite zieht man die entsprechenden Elemente auf den Bildschirm in der Mitte (Figure 2 Bildschirmkonfiguration).

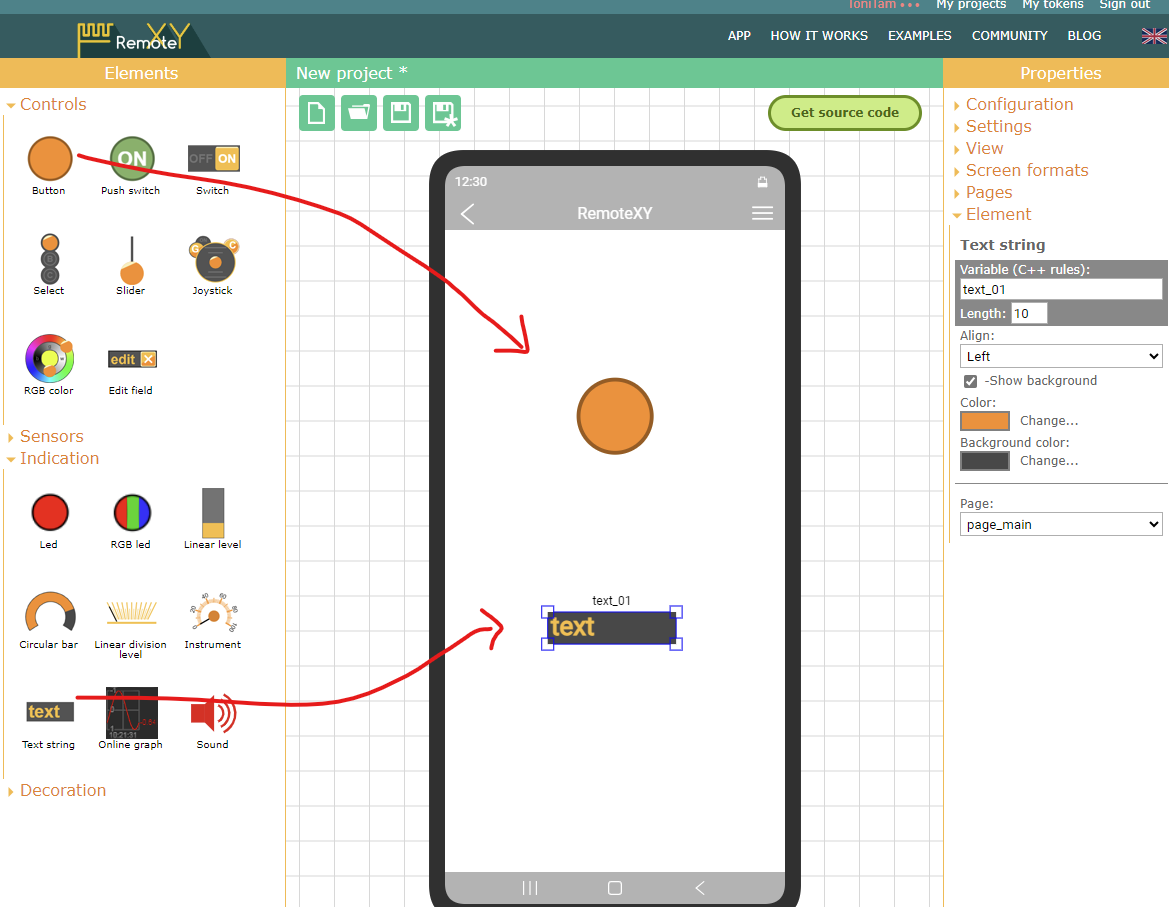


Figure 2 Bildschirmkonfiguration

Im nächsten Schritt kontrolliert man unter „Configuration“ auf der rechten Seite, ob der Microkontroller und das Kommunikationsmodul richtig ausgewählt wurden (Figure 3 Beispielkonfiguration). Für dieses Beispiel nutze ich einen ESP32 und nutze dessen eingebautes Bluetooth Kommunikationsmodul.

A screenshot of a cell phone

Description automatically generated

Figure 3 Beispielkonfiguration

Hat man das Projekt gespeichert, kann über den „Get source code“ Button der Programmcode heruntergeladen und in die Arduino IDE kopiert werden. Über den Library Manager der Arduino IDE muss dann noch die RemoteXY Library installiert werden.

Figure 4 zeigt einen Ausschnitt des Programmcodes, der von RemoteXY generiert wurde, die Zahlenfolge in dem Array definiert das Layout auf dem Display und die struct-variable RemoteXY enthält die Variablen für den Button und das Textfeld.

A screenshot of a computer program

Description automatically generated

Figure 4 RemoteXY Variabel

Im setup() Teil des Programmcodes wird RemoteXY initiiert (Figure 5 Zeile 66) und dann zu Beginn der Hauptschleife der aktuelle Status des Displays aktualisiert (Zeile 71). Damit die Verbindung zur App nicht abgebrochen wird, darf man im Code keine delay() Anweisung verwenden, stattdessen wird der Befehl RemoteXY\_delay() verwendet (Zeile 78). Der Abruf des Button-Wertes erfolgt über RemoteXY.button\_01 in Zeile 72. Der Variable-Name kann im RemoteXY Editor angepasst werden, wenn das entsprechende Element ausgewählt wird.

Das Textfeld auf dem Display hat den Variablenamen RemoteXY.text\_01, er ist vom Typ Char und muss zur Zuweisung des Integer Wertes mit der Funktion snprintf() (Zeile 74) umgewandelt werden.

A computer code on a white background

Description automatically generated

Figure 5 Hello World Code

Nach erfolgreicher Kompilierung des Programms kann der laufende ESP mit der App entdeckt werden. Klickt man in der App auf “+” und Bluetooth LE wird der ESP mit RemoteXY angezeigt. Dieser Name kann im Programmkopf mit dem Befehl #define REMOTEXY\_BLUETOOTH\_NAME "HelloWorld" z.B. auf HelloWorld angepasst werden. Hat alles funktioniert, sollte sich die Zahl im Textfeld im Sekundentakt erhöhen und wird beim Drücken des Buttons zurückgestellt.

Die in diesem Beispiel gezeigten Elemente würden bereits ausreichen, um das Display des Taupunkt-Lüfters (s.a. Make XY) zu ersetzen, so kann der Verkabelungsaufwand verringert werden und die Elektronik unter einer Abdeckung versteckt werden.

## Wasserwaage mit LED Anzeige und Cloud Verbindung

Im folgenden, etwas komplexeren, Beispiel, wird gezeigt, wie mit RemoteXY die Sensoren des Handy ausgelesen und die Daten über eine Cloud-Verbindung übertragen werden können. Um auch für dieses Beispiel komplett auf eine Verdrahtung verzichten zu können, soll die Kommunikation mit einem, in Wokwi simulierten, ESP erfolgen (s.a. Make-Artikel zu Wokwi).

Das Programm zeigt die Messungen des Gyroskope Sensors, den Roll- und Kippwinkel, an und erlaubt eine Nullsetzung zur Kalibrierung. Über die beiden Messwerte wird dann die Neigungsrichtung errechnet und diese auf einen an den ESP angeschlossenem Neopixel-Ring angezeigt. Zudem wird der Wert zusätzlich über ein Zeigerinstrument auf dem Display angezeigt.

Damit der RemoteXY Cloudserver genutzt werden kann, muss man ein Benutzerkonto anlegen und dann unter dem Menüpunkt MyTokens einen neuen Token erstellen. Zudem muss die Konfiguration für die Cloudverbindung entsprechend angepasst werden (Figure 6), in diesem Fall wählt man für den virtuellen ESP eine Wifi -Verbindung ohne Passwort und mit der Wokwi SSID „Wokwi-GUEST“. Im Feld Token im Abschnitt Cloud Server wählt man dann den Token, den man zuvor in RemoteXY angelegt hat.

A screenshot of a computer

Description automatically generated

Figure 6 Cloud Konfiguration und Settings

Figure 7 zeigt die Gestaltung des Displays für die Wasserwaage, zur Auswertung des Gyroskope Sensors wählt man den Sensor Orientation aus dem Sensor Menü. Soll das Symbol selbst später nicht auf dem Display erscheinen, wählt man die Option “Hide on screen” (unten rechts in der Abbildung)

A screenshot of a phone

Description automatically generated

Figure 7 Bildschirm Wasserwaage

Für das Instrument habe ich die Einstellungen gemäß Figure 8 gewählt, um die Richtung über die vollen 360° sauber anzeigen zu können.

A screenshot of a computer

Description automatically generated

Figure 8 Einstellung Instrument

Über “Get source Code” erhält man wieder das Programm, das so 1:1 in die Wokwi IDE kopiert werden kann.

Der Programmkopf muss noch etwas angepasst werden, damit die Wifi-Verbindung in Wokwi funktioniert, die Details können Figure 9 entnommen werden. Allerdings muss in Zeile 39 noch der selbst-generierte Token eingesetzt werden.

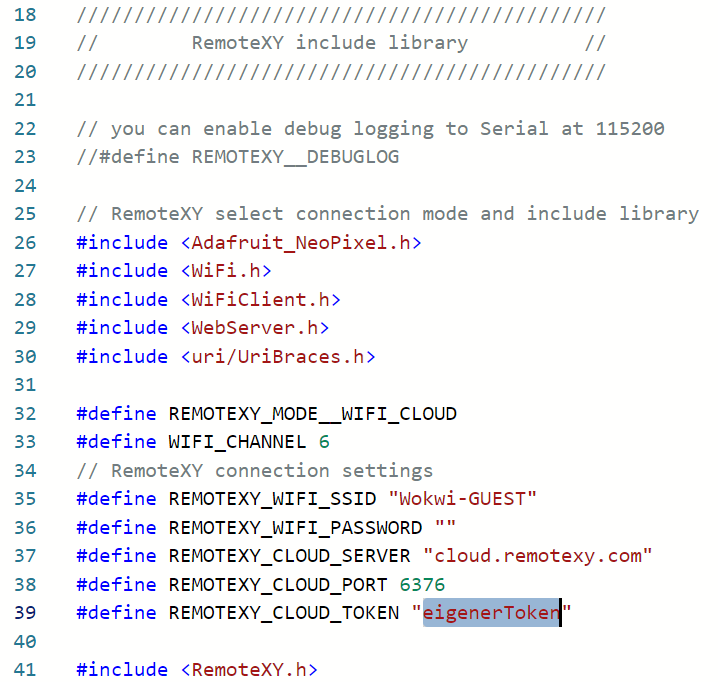


Figure 9 Woki Wasserwaage Programmkopf

Zudem habe ich im Tab Library Manager die beiden Libraries RemoteXY und Adarfruit NeoPixel hinzugefügt (Figure 10).

A screenshot of a computer

Description automatically generated

Figure 10 ESP Bibliotheken

Hat man diesen Teil erledigt, kann das Programm in Wokwi bereits kompiliert und gestartet werden. Am Wifi Symbol sieht man dann, ob der Wokwi ESP eine Internetverbindung hat, dies kann unter Umständen ein paar Sekunden dauern. Sobald die Verbindung steht, öffnet man die App auf dem Handy und wählt das QR-code Symbol (Figure 11). Mit dem QR-code Scanner kann der QR-Code unter dem Programmcode in RemoteXY eingescannt werden, dabei wird die Verbindung zu dem Cloudserver direkt mit dem richtigen Token aufgebaut.

A screenshot of a remote control

Description automatically generated

Figure 11 QR-code link

Ist alles richtig eingestellt, sieht man auf der App bereits das richtige Display, allerdings natürlich noch ohne Funktion.

Im Verdrahtungseditor von Woki habe ich einen Neopixel-Ring mit 16 LEDs ausgewählt und diesen wie in Figure 12 gezeigt verdrahtet. Der Dateneingang des Neopixel Ring ist mit PIN 25 des ESP verbunden.

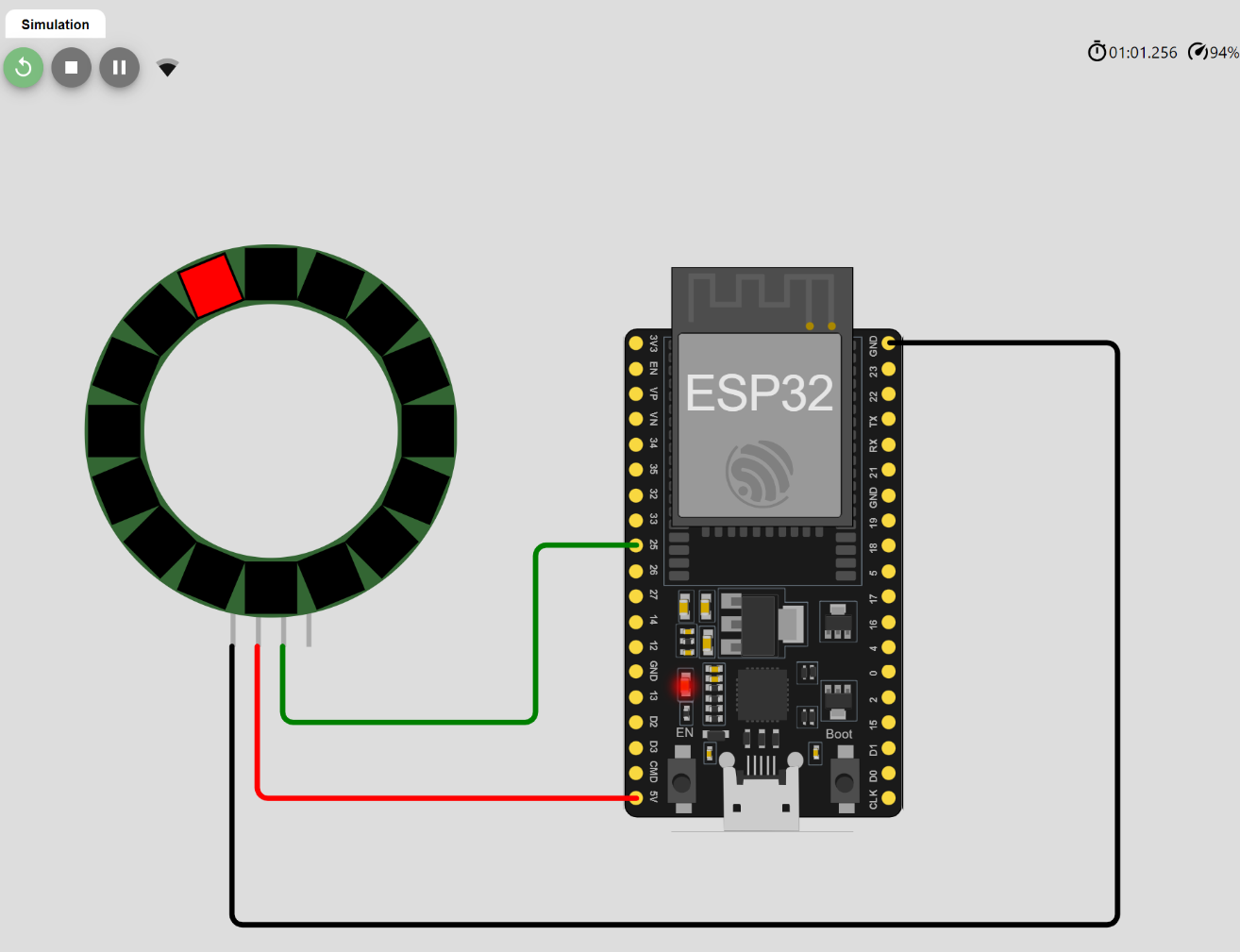


Figure 12 Neopixelring an ESP32

Figure 14 zeigt den Programmcode. In den Zeilen 106 und 107 wird der Messwert des Gyroskope- Sensors erfasst, sobald der Zero-Button gedrückt wird (RemoteXY.button\_zero == true). Diese Werte werden dann von den weiteren Messungen abgezogen (Zeilen 111, 112), damit die in der Ruhelage die seitliche Neigung (engl. roll) und der Sturz (engl. pitch) als 0 angezeigt werden.

In Zeile 115 wird dann der Kippwinkel (0-360°) über die Arctan() – Funktion der beiden Neigungswinkel berechnet (s.a. Figure 13) und dem Zeigerinstrument auf dem Display zugewiesen (RemoteXY.instrument\_winkel). In Zeile 122 wird der errechnete Winkelwert auf die 16 LEDs skaliert und schlussendlich in den Zeilen 123-130 die entsprechende LED angesteuert und auf Rot gesetzt.

A diagram of a triangle

Description automatically generated

Figure 13 Winkelberechnung

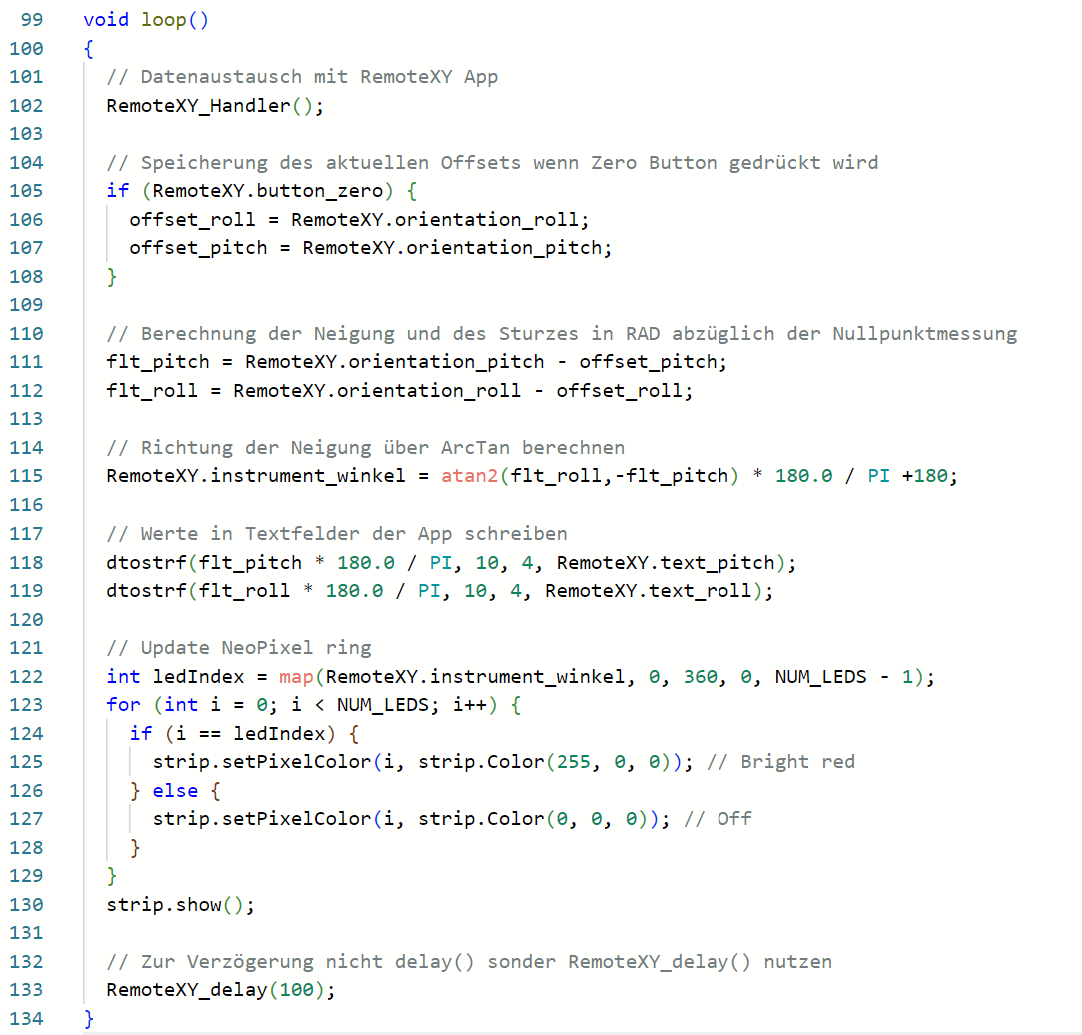


Figure 14 ESP Sketch

## Grenzen von RemoteXY

Mir persönlich gefällt an RemoteXY wie schnell man ein funktionales Display erstellen kann, da ich die Verwendung von OLEDs oder LCDs für kleine Projekte häufig zu mühsam finde. Bis zu 5 Elementen ist RemoteXY zudem kostenfrei, man benötigt nicht mal ein Nutzerkonto. Der Einfachheit fällt die Flexibilität zu Opfer, die Darstellung der Elemente kann nicht dynamisch gesteuert werden und vor allem der Darstellung eines Messwertgraphen sind enge Grenzen gesetzt, da die Zeitachse nicht skaliert werden kann. Für viele meiner Anwendungen ist das jedoch gar nicht nötig.

Schreibt uns doch bitte, wie ihr das seht, welches Projekt würdet ihr gerne mit einem kabellosen Display realisieren.