Connecting Everyday Things

Handbuch · Stand 16. Oktober 2026 · aktuelle Fassung online: connectingeverydaythings.com/handbuch.html

INHALT

1. Überblick
2. Schnellstart
3. Senden & Empfangen
    1. Handgesten
    2. Freihand-Zeichnung
    3. Körperpose
    4. Mimik-Spiegel
    5. Arduino am Empfänger
4. Things
    1. Einfach & Erweitert
    2. Vorbereitete Aufbauten
    3. Stationen
    4. Eingänge
    5. Verbindungen
    6. Kombinieren
    7. Abtasten & Quantisieren
    8. Testsignal
    9. Ziele
    10. Bühne & Klang
    11. Handalphabet
    12. Farbpunkte & Bewegung
5. Anleitungen
    1. Poti → LED (Firmata)
    2. Temperatur (eigener Sketch)
    3. Uno R4 WiFi über MQTT
    4. Scheinwerfer über DMX
    5. TouchOSC, Max, Pure Data
    6. MIDI und Musiksoftware
6. Monitor (für Lehrende)
7. Arbeiten ohne Internet
8. Fehlerbehebung
9. Downloads
10. Begriffe

Eine Plattform im Browser, die Menschen, Bewegungen und Alltagsdinge über das Netz verbindet: Handgesten steuern entfernte Figuren, Sensoren an einem Ort schalten Lampen, Motoren oder Klänge an einem anderen.

1 · Überblick

Adresse: connectingeverydaythings.com. Bei Störungen steht dieselbe Plattform unter hauchz.art bereit – alle Beteiligten müssen dann dieselbe Adresse verwenden.

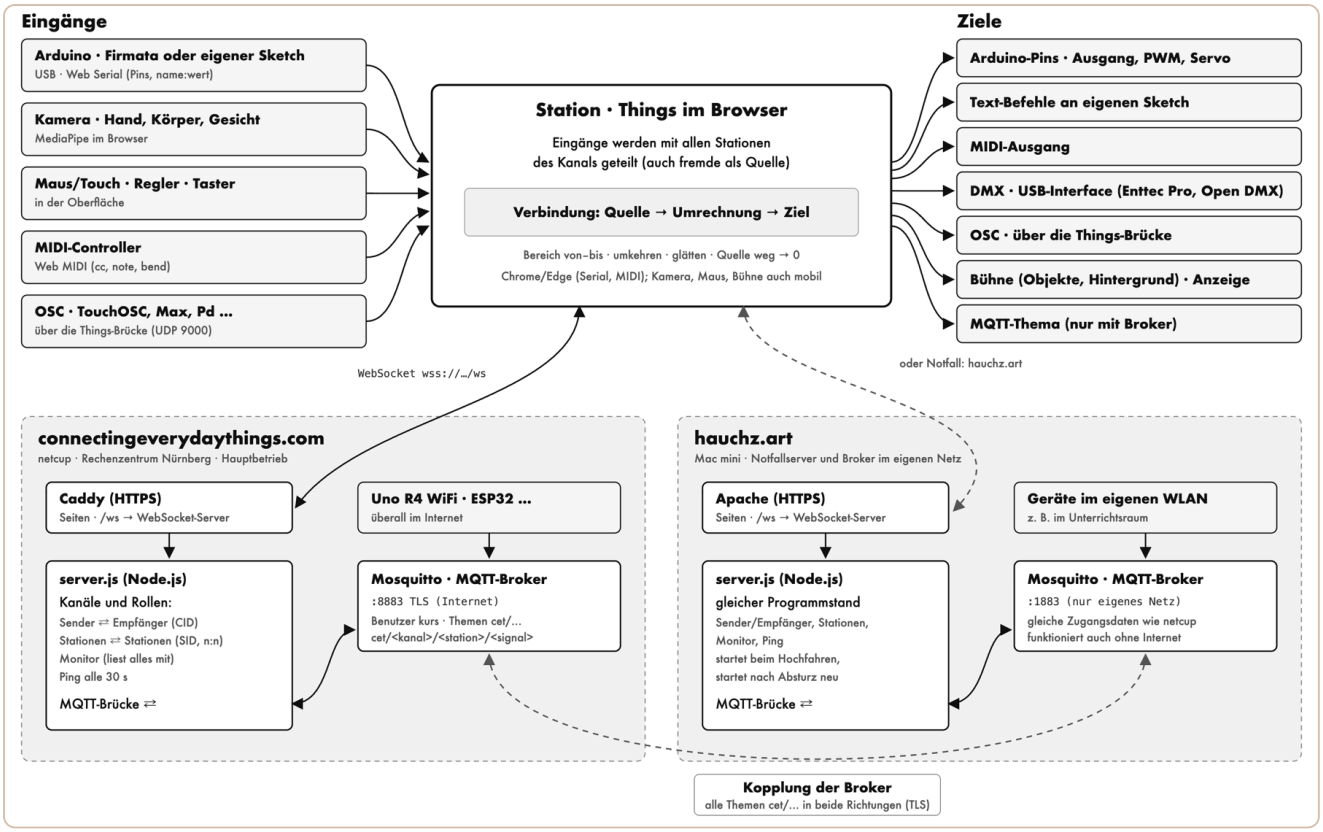
Auf der Startseite wählt man eine **Anwendung** und einen **Kanal** (eine Zahl). Alle, die denselben Kanal wählen, sind miteinander verbunden; verschiedene Kanäle stören sich nicht.

Ganz oben wählt man den **Umfang**; der Browser merkt ihn sich:

UMFANG	ANGEBOTEN WIRD
Jajaja	nur die Sockmonkey-Anwendungen: **Handgesten** und **Freihand-Zeichnung** (Maus/Touch)
Einfach	nur **Things** für Sensoren und Aktuatoren, mit den Grundfunktionen (siehe Einfach & Erweitert)
Erweitert	alle Anwendungen; Things mit allen Funktionen

> Für eine Gruppe lässt sich der Umfang im Link vorgeben, z. B. `connectingeverydaythings.com/?umfang=jajaja` oder `?umfang=einfach`.

ANWENDUNG	PRINZIP	WOFÜR
Handgesten, Freihand-Zeichnung, Körperpose, Mimik-Spiegel	**Senden** (Kamera bzw. Maus) → **Empfangen** (Darstellung, optional Arduino)	Fernsteuerung von Figuren und Objekten, gemeinsames Zeichnen
Things	gleichwertige **Stationen**, jede kann lesen und steuern	Sensoren, Aktoren, Licht, Klang, Bildschirm frei verknüpfen



Was mit was verbunden werden kann. Die Server vermitteln nur Werte; Geräte hängen an den Rechnern der Stationen oder sprechen MQTT.

Browser

- **Chrome oder Edge am Computer** können alles (Arduino per USB, MIDI, DMX).
- Firefox: alles außer USB-Geräten (Arduino, DMX).
- Safari, iPhone, iPad: Kamera, Maus/Touch, Anzeigen und Bühne – aber kein USB und kein MIDI.
- Die Kamera funktioniert nur über `https://`; der Browser fragt einmal nach der Erlaubnis.

2 · Schnellstart

Zwei Geräte, eine Bewegung

1. Gerät A: Startseite öffnen, **Handgesten**, Rolle **Senden**, Kanal z. B. `7`, **Verbinden**, Kamera erlauben.
2. Gerät B: gleiche Anwendung, Rolle **Empfangen**, Kanal `7`.
3. Hand vor die Kamera halten – der Cursor auf Gerät B folgt dem Zeigefinger.

Things in zwei Minuten (ohne Hardware)

1. Zwei Geräte öffnen **Things** mit demselben Kanal. Jede Station bekommt oben einen Namen.
2. Gerät A: **+ Regler**. Der Regler erscheint bei Gerät B unter „Andere Stationen".
3. Gerät B: **+ Verbindung** → Quelle „A · regler1" → Ziel „Bühne · Objekt 1 · Größe". Den Regler bewegen.

Things startet im Modus **Einfach** – das genügt für diese Schritte. Alles Weitere schaltet oben rechts **Erweitert** frei (siehe Einfach & Erweitert).

3 · Senden & Empfangen

In diesen Anwendungen gibt es zwei Rollen: Die **Senden**-Seite erfasst etwas (Kamera, Maus), die **Empfangen**-Seite zeigt es an und kann es an einen Arduino weitergeben. Mehrere Sender und Empfänger im selben Kanal sind möglich.

Auf beiden Seiten: **Video-Chat** öffnet eine Videokonferenz im Fenster. Auf der Senden-Seite schaltet `M` (oder ein Klick auf das Mikrofon) die **Tonübertragung** an den Empfänger ein und aus.

Handgesten

Die Kamera erkennt eine Hand. Übertragen werden die Position des Zeigefingers (X, Y), die Entfernung zur Kamera (Z) und die Zahl gestreckter Finger. Auf der Empfangen-Seite zeigt ein Cursor die Position, seine Größe die Entfernung. `D` blendet beim Senden die Finger-Erkennung ein.

Freihand-Zeichnung

Mit Maus oder Finger zeichnen; die Linien erscheinen bei allen Empfängern, jede Person in eigener Farbe. `C` oder **Löschen** leert die Fläche.

Körperpose

Die Kamera erfasst den ganzen Körper. Übertragen werden Position, Beugung, beide Arme, beide Beine und die Kopfhaltung. Die Empfangen-Seite zeigt eine Figur – **spiegelbildlich**, wie in einem Spiegel. Drei Figuren stehen zur Wahl:

TASTE	FIGUR
`1`	Hampelmann aus Karton – Glieder federn nach
`2`	Gliederpuppe (Zeichenpuppe aus Holz)
`3`	Figur mit T-Shirt
`F` / Button	nächste Figur

Mimik-Spiegel

Die Kamera erfasst das Gesicht: Kopfdrehung und -neigung, Augen, Brauen, Mund, Lächeln. Ein Avatar auf der Empfangen-Seite spiegelt die Mimik. `1` Mensch, `2` Biber, `3` Affe, `4` Fledermaus, `A` oder Button: nächster Avatar.

Arduino am Empfänger

Auf der Empfangen-Seite verbindet **Seriell verbinden** (Chrome/Edge) einen Arduino per USB. Er erhält die Werte als Textzeile mit 9600 Baud, Werte von 0 bis 180 (passend für Servos):

ANWENDUNG	ZEILE AN DEN ARDUINO
Handgesten	`x,y,z`
Körperpose	`position,beugung,armL,armR,beinL,beinR,kopfX,kopfY`
Mimik-Spiegel	`kopfX,kopfY` (Drehen, Nicken)

Ein passender Sketch für zwei Servos (Drehen an Pin 9, Neigen an Pin 10) ist unter Downloads zu finden.

4 · Things

Things verbindet beliebige Eingänge mit beliebigen Zielen – auch über Ortsgrenzen hinweg. Die Seite hat drei Bereiche: **Diese Station** (eigene Eingänge und Geräte), **Andere Stationen** und **Verbindungen**, darunter die **Bühne**.

Einfach & Erweitert

Oben rechts lässt sich der Umfang der Bedienung umschalten. Der Browser merkt sich die Wahl. Beim ersten Öffnen ist **Einfach** eingestellt.

	EINFACH	ERWEITERT – ZUSÄTZLICH
Eingänge	Regler, Taster, Testsignal, Abtasten & Quantisieren, Arduino mit Firmata, Hand-Kamera, Maus-/Touch-Fläche	eigener Sketch (Text-Protokoll), Körper, Gesicht, Farbpunkte & Bewegung, MIDI, OSC, Kombinieren
Verbindungen	Quelle → Ausgang von – bis → Ziel	Eingang von – bis, umkehren, glätten, Quelle weg → 0
Ziele	Pins (Ausgang, PWM, Servo), Bühne, Anzeige	Text-Befehl, MIDI, DMX, OSC, MQTT, Handalphabet

- Was gerade benutzt wird, bleibt auch im einfachen Modus sichtbar und läuft weiter – etwa eine verbundene MIDI-Schnittstelle, ein vorhandener Kombinieren-Eingang oder eine Verbindung mit „umkehren". Umschalten löscht nichts.
- Für den Unterricht kann der Modus im Link stehen: `things.html?channel=7&modus=einfach` bzw. `&modus=erweitert`.
- Andere Stationen sehen alle Eingänge, unabhängig davon, in welchem Modus sie selbst arbeiten.

Vorbereitete Aufbauten

Ein Link kann eine Station fertig einrichten – Name, Eingänge, Verbindungen und Pin-Modi. Danach nur noch den Arduino mit StandardFirmata verbinden. Bestehende Regler und Verbindungen des Kanals werden nach Rückfrage ersetzt.

LINK (KANAL FREI WÄHLBAR)	RICHTET EIN
`things.html?channel=7&aufbau=strich`	Station „Strich“: Analogeingang **A0** (z. B. Poti oder Grafitstrich) → **Abtasten & Quantisieren** (`quant1`), dazu ein **Testsignal**
`things.html?channel=7&aufbau=affe`	Station „Affe“: Pin **D9 als Servo**; Verbindung „Strich · quant1“ → Servo (0–1023 → 0–180°) und eine Anzeige. Fällt die Station „Strich“ weg, hält der Servo seine Stellung.

> Anschluss Servo: Signal (orange/gelb) an D9, Plus (rot) an 5 V, Minus (braun/schwarz) an GND. Größere Servos besser über ein eigenes Netzteil versorgen (GND verbinden). Anschluss Strich/Poti: Enden an 5 V und GND, Abgriff an A0.

Stationen

- Jeder Browser mit Things ist eine **Station**. Alle Stationen im selben Kanal sind gleichwertig: Jede teilt ihre Eingänge und kann Eingänge der anderen nutzen.
- **Name** oben eintragen, eindeutig wählen (z. B. „Werkstatt“, „Atelier“). Zwei Stationen mit gleichem Namen werden zusammengeworfen.
- Name, Regler, Verbindungen und Geräteeinstellungen merkt sich der Browser – beim nächsten Öffnen ist alles wieder da.
- MQTT-Geräte (z. B. ein Uno R4 WiFi) erscheinen ebenfalls als Stationen.

Eingänge

QUELLE	SO EINRICHTEN	EINGÄNGE HEISSEN
Arduino · Firmata	Auf dem Arduino „StandardFirmata" (Arduino-IDE → Beispiele → Firmata). Protokoll *Firmata*, **Verbinden**, Port wählen. Jeden Pin einstellen: Analog, Eingang, Eingang (Pullup).	`A0` …, `D2` … (Analog 0–1023, digital 0/1)
Arduino · eigener Sketch	Sketch sendet Zeilen wie `poti:512,temp=23.4`. Protokoll *Text*, Baudrate wie im Sketch.	die Namen aus dem Sketch
Kamera	„Kamera & Maus" → Hand, Körper oder Gesicht.	`hand_x`, `finger`, `griff`, `buchstabe`, `beugung_zeige` … · `arm_links`, `bein_rechts` … · `mund`, `laecheln`, `auge_links` …
Farbpunkte & Bewegung	„Kamera & Maus" → Farbpunkte & Bewegung. Ohne KI-Modell, z. B. für Sockmonkeys (mehr).	`drehen`, `nicken`, `neigen` (0–180), `bewegung`, `bewegung_dx` …
Maus/Touch	„Maus-/Touch-Fläche" ankreuzen.	`maus_x`, `maus_y` (0–100), `maus_druck`
Regler, Taster	**+ Regler**, **+ Taster** (Taster auf Wunsch „einrasten").	`regler1`, `taster1` …
Kombinieren	**+ Kombinieren**: aus mehreren Signalen wird ein neuer Eingang (mehr).	`kombi1` …
Testsignal	**+ Testsignal**: gleichmäßige Schwingung als Quelle für Versuche (mehr).	`test1` … (0–1023)
Abtasten & Quantisieren	**+ Abtasten & Quantisieren**: ein Signal seltener abtasten und/oder auf 2 bis 1024 Stufen vergröbern (mehr).	`quant1` …
MIDI-Controller	„MIDI aktivieren", Eingang wählen.	`cc7`, `note60`, `bend` (Kanal 2: `cc7_k2`)
OSC	Things-Brücke starten, „Mit Brücke verbinden" (siehe Anleitung).	`osc/1/fader1` …

Jeder Eingang zeigt seinen Wert und einen kurzen Verlauf. Bei Werten aus eigenen Sketchen errät Things den Bereich – in Verbindungen lieber selbst eintragen.

Verbindungen

+ Verbindung legt eine Zeile an: **Quelle** → **Umrechnung** → **Ziel**.

FELD	BEDEUTUNG
Quelle	ein eigener („Hier · …“) oder fremder Eingang („Station · …“)
Eingang von – bis	welcher Bereich der Quelle genutzt wird (z. B. Temperatur 20–30)
Ausgang von – bis	worauf er abgebildet wird (z. B. 0–255 für eine LED). Umgekehrte Werte (255–0) kehren die Richtung um.
umkehren	Eingang klein = Ausgang groß
glätten	weiche statt ruckartige Bewegung (Regler nach rechts = träger)
Quelle weg → 0	an: Verschwindet die Quelle (Station geschlossen, Gerät getrennt), geht der Ausgang auf 0. Aus: letzter Wert bleibt stehen.
an	Verbindung vorübergehend abschalten, ohne sie zu löschen

Eine rote Zeile unter der Verbindung erklärt, was noch fehlt (z. B. „Ziel-Pin ist nicht als Ausgang eingestellt“).

> Jede Verbindung führt **eine** Quelle zu **einem** Ziel. Zwei Verbindungen auf dasselbe Ziel werden nicht verrechnet – der Ausgang springt zwischen beiden hin und her. Dafür gibt es „Kombinieren“.

Kombinieren

+ Kombinieren (Modus Erweitert, unter „Eingänge dieser Station“) legt einen neuen Eingang `kombi1` an, der aus zwei bis sechs Quellen (A, B, …) berechnet wird – eigenen und solchen anderer Stationen. Er lässt sich wie jeder Eingang verbinden und ist auch für die anderen Stationen sichtbar.

VERKNÜPFUNG	ERGEBNIS	BEISPIEL
Mittelwert	0–100	zwei Personen regeln gemeinsam eine Lampe
Summe	0–100 (höchstens 100)	mehrere Taster füllen eine Anzeige
Maximum / Minimum	0–100	der lauteste / leiseste von mehreren Sensoren zählt
Produkt – A regelt B	0–100	Regler A als „Lautstärke“ für Sensor B; ist A auf 0, ist alles 0
Differenz A − B	−100 bis 100	Temperatur hier minus Temperatur dort
UND	0 oder 1	Licht an, wenn Bewegung **und** Taster an der anderen Station
ODER	0 oder 1	Alarm, wenn **eine** von mehreren Stationen auslöst

> Jede Quelle wird auf ihren eigenen Bereich bezogen: ein Poti auf 512 von 1023 zählt wie ein Regler auf 50 von 100 – nämlich halb. „Über halb“ bei UND/ODER heißt: über der Mitte dieses Bereichs, bei einem Taster also „gedrückt“. Fehlt eine Quelle (Station geschlossen), rechnen Mittelwert, Summe, Maximum, Minimum und ODER mit den übrigen weiter; UND und Produkt ergeben dann 0.

Abtasten & Quantisieren

+ Abtasten & Quantisieren (unter „Eingänge dieser Station“) legt einen neuen Eingang `quant1` an. Er übernimmt den Wert einer Quelle und vergröbert ihn auf zwei Arten, die sich getrennt einstellen lassen:

- **Stufen** (Werte-Auflösung): nur noch 2, 4, 8, 16, 32, 64, 256 oder 1024 Werte – das entspricht 1, 2, 3, 4, 5, 6, 8 oder 10 Bit –, gleichmäßig verteilt über den Bereich der Quelle, einschließlich Anfang und Ende. Zwischenwerte springen auf die nächstgelegene Stufe. Ein Arduino-Analogeingang liefert 1024 Stufen (10 Bit). „nicht quantisieren“ lässt die Werte unverändert.
- **Abtasten** (zeitliche Auflösung): nur 20, 10, 5, 2, 1 oder 0,5 Mal pro Sekunde wird ein neuer Wert übernommen; dazwischen bleibt der Wert stehen. „laufend“ übernimmt jeden Wert (etwa 50 Mal pro Sekunde).

Der Ausgang bleibt auf derselben Skala wie die Quelle (z. B. 0–1023) und ist ein eigener Eingang: Er wird an die anderen Stationen übertragen und lässt sich wie jeder Eingang verbinden. Als Quelle dient jeder Eingang dieser oder einer anderen Station.

Vergleichsbild: dünne Linie = Eingang (die digitalen Werte der Quelle in ihrer vollen Auflösung – keine stufenlose analoge Kurve), kräftige Stufenlinie = Ausgang, Punkte = Abtastzeitpunkte, waagerechte Linien = erlaubte Stufen. Rechts zeigen zwei Zeiger beide Werte auf derselben Skala. **Vollbild** zeigt den Vergleich groß.

ANZEIGE	BEDEUTUNG
Stufe, Code	Nummer der aktuellen Stufe und ihr Binärcode (bei 8 Stufen z. B. `101`)
Abweichung	Abstand zwischen Eingang und Ausgang. Durch die Stufen höchstens eine halbe Stufe; durch das Abtasten kommt hinzu, dass sich der Eingang seit dem letzten Abtastzeitpunkt verändert haben kann.
Abtastung	Werte pro Sekunde und Abstand zwischen zwei Abtastungen. Frequenzen oberhalb der halben Abtastrate können als niedrigere Frequenzen erscheinen (Aliasing); das ursprüngliche Signal ist aus den Abtastwerten dann nicht eindeutig rekonstruierbar.
Nötig	Bit pro Wert, die zur Unterscheidung der Stufen ausreichen würden, und daraus Bit pro Sekunde
Anwendungsnachricht	die Nachricht, die Things für diesen Wert überträgt, und ihre Größe in Byte. Gezählt wird nur diese Nachricht der Anwendung; Transport- und Netzwerkdaten (WebSocket-Rahmen, TCP/IP, WLAN) kommen hinzu.
Gemessen	tatsächlich gesendete Nachrichten pro Sekunde und Anwendungsdaten pro Sekunde. Things sendet nur, wenn sich ein Wert ändert – weniger Stufen und seltenere Abtastung bedeuten weniger Nachrichten. Zusätzlich geht alle 2 Sekunden eine Gesamtmeldung mit allen Werten hinaus, die hier nicht mitgezählt wird. Die gemessene Rate entspricht deshalb nicht einfach „Bit pro Wert × Abtastrate“.

> Der Wert der Quelle ist bereits digital (z. B. 10 Bit vom Arduino, abgefragt etwa 20- bis 50-mal pro Sekunde). Abtasten und Quantisieren vergröbern ihn zusätzlich. Die Glättung von Verbindungen wirkt erst danach.

Testsignal

+ Testsignal legt einen Eingang `test1` an, der sich von selbst gleichmäßig bewegt – als wiederholbare Quelle, etwa für Abtasten & Quantisieren, ohne dass jemand gleichmäßig an einem Regler drehen muss.

- **Form:** Sinus, Dreieck, Sägezahn oder Rechteck.
- **Frequenz:** 0,1 bis 2 Schwingungen pro Sekunde.
- Werte sind ganze Zahlen von 0 bis 1023 – wie bei einem 10-Bit-Analogeingang –, neu berechnet etwa 50-mal pro Sekunde. Wie jeder Eingang wird das Testsignal an die anderen Stationen übertragen und lässt sich verbinden.

> Mit dem Abtasten lässt sich so Aliasing zeigen – dafür die Form **Sinus** verwenden (Dreieck, Sägezahn und Rechteck enthalten zusätzliche höhere Frequenzen): Ein Sinus mit 1 Schwingung pro Sekunde, einmal pro Sekunde abgetastet, ergibt einen fast stehenden Ausgang; ein Sinus mit 1,2 Schwingungen pro Sekunde erscheint als langsame Schwingung mit 0,2 pro Sekunde.

Ziele

ZIEL	VORAUSSETZUNG	EINTRAG / WERTE
Arduino-Pin · Ausgang	Firmata, Pin auf *Ausgang*	an/aus (ab der Hälfte des Bereichs)
Arduino-Pin · PWM	Firmata, Pin auf *PWM* (Uno: 3, 5, 6, 9, 10, 11)	0–255, z. B. LED-Helligkeit
Arduino-Pin · Servo	Firmata, Pin auf *Servo*	0–180°
Text-Befehl	eigener Sketch (Protokoll Text)	Name, z. B. `led` → Sketch erhält `led:128`
MIDI	MIDI aktiviert, Ausgang gewählt	`cc74`, `note64`, `bend`, Kanal 2: `cc74/2`
DMX	USB-DMX-Interface verbunden	Kanal 1–512, Werte 0–255
OSC	Things-Brücke verbunden	Adresse, z. B. `/licht/rot`, Werte 0–1
MQTT-Thema	nur auf connectingeverydaythings.com	z. B. `lampe/hell` → `cet/<kanal>/lampe/hell`
Bühne	immer	Eigenschaft eines Objekts oder des Hintergrunds
Handalphabet	immer (Quelle: `buchstabe`)	Buchstabe, Text, übertragene Hand, Sprachausgabe
Anzeige	immer	Zeigerinstrument in der Verbindung

Bühne & Klang

Eine Bildfläche (16:9) zum Steuern über Verbindungen – auch als **Vollbild** für einen Beamer.

- **Hintergrund:** Grundfarbe wählen; steuerbar sind Helligkeit (0–100) und Farbton (0–360).
- **+ Objekt:** Kreis, Quadrat, Ring, Linse oder Dreieck mit Name und Grundfarbe; mit der Maus an die Grundposition ziehen.
- Steuerbar pro Objekt: X- und Y-Position (0–100), Größe, Helligkeit, Farbton, Drehung (0–360°), Deckkraft.
- Was nicht verbunden ist, behält seinen Grundwert.

Klang

Mit ♪ neben einem Objekt klingt es so, wie es auf der Bühne steht – jedes Objekt mit eigener Stimme:

AUF DER BÜHNE	KLANG
links → rechts (X)	Tonhöhe: tief → hoch (drei Oktaven, 110–880 Hz)
oben → unten (Y)	Klangfarbe: hell, obertonreich → dumpf, weich
groß → klein (Größe)	Lautstärke: nah, laut → weit, leise
Deckkraft 0	stumm

Verbindungen auf X, Y oder Größe eines Objekts steuern damit zugleich Bild und Klang – etwa „Werkstatt · drehen" → „Bühne · Affe · X-Position": Der Sockmonkey dreht den Kopf, das Objekt wandert, der Ton steigt. Auch Ziehen mit der Maus spielt.

> Browser erlauben Klang erst nach einer Bedienung: nach dem Neuladen einmal irgendwo auf die Seite klicken. Gehört wird an dem Gerät, auf dem die Bühne läuft – für einen Raum Lautsprecher anschließen.

Handalphabet (Fingeralphabet)

Mit der Kamera im Modus **Hand** erkennt Things Buchstaben des **DGS-Fingeralphabets**. Eine andere Station zeigt sie an, setzt sie zu Wörtern zusammen, zeichnet die übertragene Hand und kann sie vorlesen.

1. Station A: „Kamera & Maus" → **Hand**. Neue Eingänge: `buchstabe` (A = 1 … Z = 26, 0 = keiner) und `beugung_daumen` … `beugung_klein` (0 = gestreckt, 100 = gebeugt). Unter der Auswahl zeigt eine **Kontrollanzeige** live „Hand erkannt · Buchstabe: …".
2. Station B: **+ Verbindung**, Quelle „A · buchstabe", Ziel **Handalphabet**. Optional „Buchstaben sprechen" (deutsche Buchstabennamen: „be", „ell" …) und „Wörter sprechen" ankreuzen. Zum Ausprobieren an einem Gerät: Quelle „Hier · buchstabe".
3. Buchstaben jeweils etwa eine halbe Sekunde halten. Für **doppelte Buchstaben** (LL) die Hand kurz lösen. Eine Pause von gut 1,5 Sekunden ohne Buchstabe ergibt ein **Leerzeichen**.

Erkannt werden derzeit 13 Buchstaben: A, B, C, D, H, I, L, O, S, U, V, W, Y – Handflächen zur Kamera, Hand gut ausgeleuchtet. Buchstaben mit Bewegung (J, Z, Ä, Ö, Ü, SCH, ß) und solche, die sich nur in Feinheiten unterscheiden, werden nicht erkannt.

> Die Erkennung beruht auf festen Regeln, nicht auf geprüften Sprachdaten. Sie ist ein Experimentier- und Lehrwerkzeug, kein Ersatz für Gebärdensprache oder Dolmetschende. Die Handformen sollten mit DGS-Lehrmaterial bzw. mit gebärdensprachkompetenten Personen abgeglichen werden; im Fingeralphabet gibt es regionale Varianten. Die Regeln stehen offen in `things-alphabet.js`.

Die **Beugungswerte** eignen sich z. B. für eine Roboterhand: jede Beugung über eine Verbindung auf einen Servo (Ausgang 0–180) legen.

Farbpunkte & Bewegung (z. B. Sockmonkey)

Dieser Kameramodus verfolgt **Objekte** statt Menschen – etwa einen Sockmonkey, den man hier von Hand bewegt, damit der Affe am anderen Ort (mit Servos) es nachmacht. Er braucht kein KI-Modell und läuft direkt im Browser.

1. „Kamera & Maus" → **Farbpunkte & Bewegung**. Objekt gut ausleuchten, möglichst ruhiger Hintergrund.
2. **Wonach suchen:** *Augenweiß* findet die zwei größten hellen, weißen Flecken (z. B. die Augen). *Angeklickte Farbe*: zwei kleine grelle Marker (Aufkleber, Filzpunkte) anbringen und ins Kamerabild auf einen klicken. Gefundene Stellen erscheinen im Bild eingefärbt.
3. Objekt in Ruhehaltung bringen und **Mitte setzen** drücken – dann stehen Drehen, Nicken und Neigen auf 90.
4. Verbindungen anlegen, z. B. „Werkstatt · drehen" → Servo (0–180) am Affen der anderen Station.

EINGANG	BEDEUTUNG
`drehen`, `nicken`	seitliche bzw. senkrechte Lage der beiden Punkte gegenüber der Mitte (0–180, Ruhe = 90)
`neigen`	Kippen der Linie zwischen den Punkten (0–180, waagerecht = 90)
`punkt_abstand`	Abstand der Punkte – größer = näher an der Kamera
`punkt_da`	1 = beide Punkte gefunden, 0,5 = nur einer, 0 = keiner
`bewegung`	wie viel sich im Bild bewegt (0–100), unabhängig von Farben
`bewegung_x/_y`, `bewegung_dx/_dy`	wo und in welche Richtung (dx/dy: 50 = keine Richtung)

> Tipps: **Toleranz** erhöhen, wenn Punkte verloren gehen; senken, wenn andere Stellen mitgefärbt werden. Bei Augenweiß regelt **Helligkeit**, wie hell „weiß" sein muss. Grelle, im Bild sonst nicht vorkommende Farben (Neonpink, Neongrün) eignen sich als Marker am besten.

5 · Anleitungen

A · Poti an einem Ort, LED am anderen (Firmata)

Material: zwei Arduino Uno, ein Potentiometer, eine LED mit 220-Ω-Widerstand. Auf beide Arduinos **StandardFirmata** laden.

```
Poti:  linker Pin → 5V · Mitte → A0 · rechter Pin → GND
LED:   D9 → 220 Ω → LED langes Bein · kurzes Bein → GND
```

1. Station „Poti": Things, Kanal 5, Firmata, Verbinden. A0 steht automatisch auf *Analog*.
2. Station „Licht": Things, Kanal 5, Firmata, Verbinden. **D9** auf *PWM* stellen.
3. Bei „Licht": **+ Verbindung**, Quelle „Poti · A0", Ziel „D9 · PWM (0–255)".

> Ausprobieren: *umkehren*, *glätten*, Ausgang 0–1 mit D9 auf *Ausgang* (Schwelle), oder ein Servo an D10 als zweite Verbindung.

B · Temperatur mit eigenem Sketch

Für Sensoren mit Bibliothek (hier DHT22) sendet ein eigener Sketch Textzeilen. Beispiel-Sketch unter Downloads; Bibliothek „DHT sensor library" (Adafruit) installieren.

```
DHT22: VCC → 5V · DATA → D2 (10 kΩ nach 5V) · GND → GND
Sketch sendet alle 2 s:   temp:23.4,feuchte:45.2
Sketch versteht:          luefter:180   (PWM an D9)
```

1. Things: Protokoll *Text*, 9600 Baud, Verbinden – `temp` und `feuchte` erscheinen.
2. Verbindung „Hier · temp" → Ziel „Text-Befehl an Sketch", Name `luefter`, **Eingang 22–30** → Ausgang 0–255.
3. Oder bei einer anderen Station: „Klima · temp" → „Bühne · Hintergrund · Farbton", Eingang 18–30 → Ausgang 220–0 (kalt blau, warm rot).

C · Uno R4 WiFi mit Grove über MQTT

Mit WLAN-fähigen Boards nimmt ein Gerät **ohne Rechner und ohne USB** teil. Es erscheint in Things als Station.

GROVE-BUCHSE	MODUL	THEMA
A0	Drehpotentiometer	sendet `cet/5/werkbank/poti`
D4	Taster	sendet `cet/5/werkbank/taster`
D5	LED	empfängt `cet/5/werkbank/led` (0–255)
D6	Summer	empfängt `cet/5/werkbank/ton` (Hz)

1. Arduino-IDE: Board-Paket „Arduino UNO R4 Boards", Bibliothek „ArduinoMqttClient". Grove Base Shield auf 5 V.
2. Beispiel-Sketch öffnen und oben eintragen: WLAN, **MQTT-Passwort** (von der Kursleitung), Kanal, Stationsname.
3. `#define LOKAL 0` = über das Internet (verschlüsselt), `1` = über den Mac mini im eigenen Netz (siehe ohne Internet).
4. Hochladen. Im seriellen Monitor (115200 Baud) erscheint „MQTT verbunden"; in Things taucht „werkbank" auf.
5. LED steuern: in Things Verbindung mit Ziel **MQTT-Thema**, Thema `werkbank/led`.

Themen-Schema für eigene Geräte: `cet/<kanal>/<station>/<signal>`, Inhalt eine Zahl. Things-Stationen werden ebenfalls als Themen gespiegelt (z. B. `cet/5/Atelier/regler1`) und können direkt abonniert werden.

> Fehlercodes im seriellen Monitor: `−2` keine Verbindung (WLAN, Adresse, Port), `5` Benutzername oder Passwort falsch.

D · Scheinwerfer über DMX

1. USB-DMX-Interface anschließen, Scheinwerfer per DMX-Kabel verbinden, am Scheinwerfer die **Startadresse** einstellen (z. B. 1).
2. Things → „DMX (USB-Interface)" → Bauart wählen (*Enttec DMX USB Pro / kompatibel* oder *Open DMX*) → Verbinden.
3. Verbindung mit Ziel **DMX · Kanal**, Kanalnummer eintragen. Welcher Kanal was bewirkt (Dimmer, Rot, Grün, Blau …), steht in der Anleitung des Scheinwerfers.

> Reagiert nichts, die andere Bauart probieren. Viele Scheinwerfer brauchen den Dimmer-Kanal auf 255, damit die Farben sichtbar werden – dafür eine Verbindung von einem Regler auf diesen Kanal anlegen.

E · TouchOSC, Max/MSP, Pure Data (OSC)

Browser können kein OSC senden. Dafür gibt es die **Things-Brücke**, ein kleines Programm ohne Installation: Download und Anleitung.

1. Brücke per Doppelklick starten, Fenster offen lassen. Beim ersten Start fragen macOS/Windows nach (Anleitung auf der Download-Seite).
2. Things (Chrome, Edge oder Firefox) → „OSC (über die Things-Brücke)" → **Mit Brücke verbinden**.
3. **Empfangen:** In TouchOSC als Host die angezeigte Adresse, Port 9000 eintragen. Fader erscheinen als `osc/1/fader1`.
4. **Senden:** „Senden an" IP und Port des Programms eintragen; Verbindung mit Ziel **OSC** und Adresse z. B. `/licht/rot`.

```
Max/MSP:      [udpreceive 8000] → [route /licht/rot]
Pure Data:    [netreceive -u -b 8000] → [oscparse] → [route licht]
```

F · MIDI-Controller und Musiksoftware

1. Things → „MIDI aktivieren", Erlaubnis bestätigen. Am Controller drehen – `cc…` und `note…` erscheinen.
2. Musiksoftware am selben Mac: In der App **Audio-MIDI-Setup** → Fenster „MIDI-Studio" → **IAC-Treiber** → „Gerät ist bereit". „IAC-Bus 1" dann in Things als Ausgang wählen und in Ableton/Logic als Eingang aktivieren.
3. Verbindung z. B. „Werkstatt · A0" → Ziel **MIDI**, Befehl `cc74` → Filter im Synthesizer per MIDI-Learn zuweisen.

6 · Monitor (für Lehrende)

Der Monitor macht sichtbar, was auf dem Server passiert. Adresse: `…/monitor.html`, Zugang mit dem **Monitor-Schlüssel** (nur für Lehrende; er zeigt den Verkehr aller Kanäle).

- **Netz:** der Server in der Mitte, alle Verbindungen nach Kanälen gruppiert. Nachrichten fliegen als Punkte vom Absender über den Server zu den Empfängern; der Ping alle 30 Sekunden erscheint als Ring.
- **Terminal:** jede Anmeldung, jede Nachricht mit Inhalt, an wen weitergeleitet wurde – oder „niemand weitergeleitet", wenn im Kanal kein passender Empfänger ist.
- **Zeitlupe** (bis 1 Ereignis alle 2 s), **Pause** und **Filter** nach Ereignisart und Kanal.
- Alle Ereignisse stehen zusätzlich in der Browser-Konsole (`F12`).

Eine Station lässt sich auch ohne Browser im Terminal anmelden; ihre Nachrichten erscheinen dann im Monitor:

```
npx wscat -c "wss://connectingeverydaythings.com/ws?role=station&channel=5"
> {"t":"v","v":{"test":42}}
```

7 · Arbeiten ohne Internet

Im Raum mit dem **Mac mini** (hauchz.art) funktioniert vieles auch ohne Internet, solange alle Geräte im selben WLAN sind:

- Things und die übrigen Seiten über die **lokale Adresse des Mac mini** öffnen (nennt die Kursleitung, z. B. `http://192.168.…`).
- MQTT-Geräte mit `#define LOKAL 1`: Sie verbinden sich direkt mit dem Mac mini (Port 1883).
- Arduino, MIDI, DMX und OSC funktionieren, denn sie hängen an den Rechnern im Raum.
- Nicht möglich: Kamera-Modi (brauchen HTTPS) und Verbindungen zu anderen Orten.

Sobald das Internet wieder da ist, sind der Broker im Raum und der im Internet automatisch wieder gekoppelt.

8 · Fehlerbehebung

PROBLEM	LÖSUNG
„Verbinde …“ bleibt stehen	Internet prüfen; Seite neu laden. Alle müssen dieselbe Adresse nutzen (connectingeverydaythings.com *oder* hauchz.art).
„HTTPS erforderlich“ / keine Kamera	Adresse mit `https://` öffnen; Kamera-Erlaubnis im Browser erteilen (Schloss-Symbol in der Adresszeile).
Andere Station erscheint nicht	Gleicher Kanal? Beide in Things? Verschiedene Stationsnamen?
Kein „Verbinden“ für Arduino/DMX	Nur Chrome oder Edge am Computer. Seriellen Monitor der Arduino-IDE schließen (Port belegt).
Firmata: „Keine Antwort“	StandardFirmata erneut hochladen, 57600 Baud.
Eigener Sketch: Zeichensalat	Baudrate in Things = `Serial.begin()` im Sketch.
Ziel fehlt in der Auswahl	Voraussetzung erfüllen: Pin auf Ausgang/PWM/Servo, MIDI-Ausgang gewählt, DMX/Brücke verbunden. Die rote Zeile der Verbindung nennt den Grund.
LED leuchtet weiter, obwohl Quelle weg ist	„Quelle weg → 0“ einschalten.
Brücke wird nicht gefunden	Läuft das Fenster der Brücke? Safari verwenden geht nicht. Chrome fragt evtl. nach „Geräten im lokalen Netzwerk“ – zulassen.
OSC kommt nicht an	Gleiches WLAN, richtige IP und Port 9000; unter Windows Firewall-Frage mit „Zulassen“ beantworten.
MQTT-Gerät erscheint nicht	Serieller Monitor: Fehler `−2` = Netz/Adresse, `5` = Passwort. Kanal im Sketch = Kanal in Things?
MIDI nicht verfügbar	Safari kann kein MIDI. Gerät vor dem Aktivieren anschließen.
Handalphabet: kein Buchstabe	Kontrollanzeige an der gebärdenden Station prüfen („Hand erkannt · Buchstabe“). Handfläche zur Kamera, gutes Licht, Buchstaben ruhig halten. Erkannt werden nur A, B, C, D, H, I, L, O, S, U, V, W, Y.
Handalphabet: Buchstabe, aber keine Hand	Beide Seiten neu laden (Cmd/Strg + Shift + R). Die Hand erscheint nur, solange sie im Bild ist.
Verbindung ist weg	Verbindungen werden pro Kanal und Browser gespeichert – anderer Kanal, andere Adresse (hauchz.art) oder anderer Browser ⇒ neu anlegen.

9 · Downloads

Dieses Handbuch als PDF

Things-Brücke (OSC)
macOS und Windows

Sketch: Poti, Licht, Taster
Text-Protokoll, mit LED und Summer

zum Drucken und offline Lesen, mit Inhaltsverzeichnis und Links

Sketch: Temperatur DHT22

Text-Protokoll, mit Lüfter/LED

Sketch: Uno R4 WiFi + Grove

MQTT, Internet oder eigenes Netz

Sketch: Zwei Servos (Empfänger)

Drehen/Neigen für Senden & Empfangen

10 · Begriffe

Kanal	Zahl, die Teilnehmende zusammenführt; nur gleiche Kanäle sehen sich.
Station	ein Teilnehmer in Things – Browser oder MQTT-Gerät.
Firmata	Standard-Firmware für Arduino, mit der der Browser alle Pins direkt lesen und schalten kann.
PWM	schnelles Ein- und Ausschalten, das wie ein Zwischenwert wirkt (z. B. gedimmte LED).
MIDI	Protokoll für Musikinstrumente und Controller.
OSC	Open Sound Control – Netzwerkprotokoll für Medien- und Musikprogramme.
DMX	Protokoll für Bühnenlicht; 512 Kanäle mit Werten 0–255.
MQTT	Protokoll für vernetzte Geräte; Geräte veröffentlichen und abonnieren „Themen“ bei einem Broker.
Broker	Vermittlungsdienst für MQTT (hier auf dem Server und auf dem Mac mini).
WebSocket	dauerhafte Verbindung zwischen Browser und Server, über die alle Werte laufen.

Connecting Everyday Things · Jan M. Sieber · interactions.cc · Stand Oktober 2026