1. Day

Prepare the Raspberry Pi

To take the Raspberry Pi into operation, you need:

- USB keyboard and mouse
- HDMI-cable for the screen
- Network cable
- MicroSD-card with operating system
- Micro-USB mobile phone charger as a mains unit

Connect the mains unit last; the Raspberry Pi will switch on automatically. There is no dedicated on/off switch.

Brief notes on operating system installation

For all whose Raspberry Pis are not ready for operation yet with the current Raspbian version, find the system installation in 10 steps below:

1 Download NOOBS to the PC: **www.raspberrypi.org/downloads** and unpack the ZIP archive onto your hard disc.

2 If the SD card has already been used, reformat it on the computer with SD-Formatter: **www.sdcard.org/downloads/formatter_4**. Switch on **Format Size Adjustment**.

3 Copy the files and subdirectories onto the SD card.

4 Remove the SD card from the PC, insert it into the Raspberry Pi and boot it up. At the bottom, select **English** as the installation language. The English keyboard is now selected automatically.

5 Check the preselected Raspbian operating system.

6 After confirming a safety prompt that the memory card will be overwritten, installation will start. This will take a few minutes.

7 After completed installation, the Raspberry Pi will reboot and automatically start the configuration tool **raspi-config**.

8 Under **3 Enable Boot to Desktop/Scratch**, choose the option **Desktop Log in as user ‚pi' at the graphical desktop** .

9 Under **4 Internationalisation Options**, choose the option **Change Timezone** and set the time zone to **Europa/Berlin**.

10 Select **<Finish>** and reboot the Raspberry Pi.

The LED lights up

You do not need a program for the first experiment. The Raspberry Pi is only used to supply the LED with power. The experiment shows how LEDs are connected.

Components: 1 x plug board, 1 x LED red, 1 x 220-Ohm resistor, 2 x connection cables

2. Day

ScratchGPIO is an additional module for the particularly easy to learn programming language Scratch, which permits controlling the GPIO interface of the Raspberry Pi with Scratch.

Install ScratchGPIO

Scratch is already included in Raspbian; the additional module ScratchGPIO is installed subsequently with an LXTerminal window.

1 Start LXTerminal via the icon in the task bar.

2 `wget http://bit.ly/1wxrqdp -O isgh7.sh`

3 `sudo bash isgh7.sh`

4 Two new icons will appear on the Raspbian desktop.

5 For most experiments, choose **ScratchGPIO 7**. The version **ScratchGPIO 7plus** is needed for different additional PCBs.

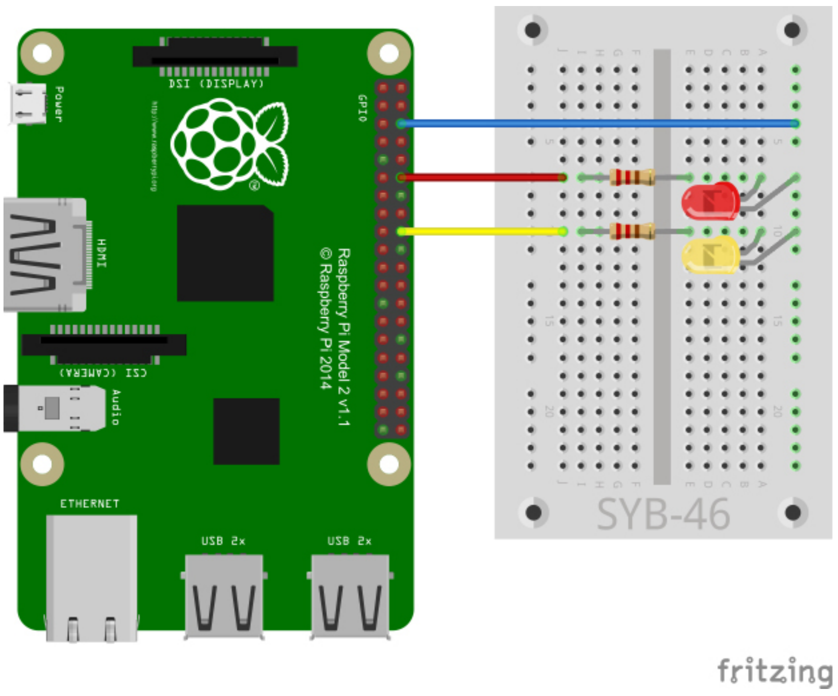
The pin assignment in ScratchGPIO

A GPIO-pin can be used as input or output. ScratchGPIO designates the GPIO-pins differently from any other programme and programming language on the Raspberry Pi. The figure in the lower right shows the pins defined as inputs (7, 8, 10, 19, 21-24 and 26) as well as the pins defined as outputs (11-13, 15, 16, 18). The pins numbered 29, 31-33, 35-38 and 40 can be used as inputs or outputs.

LEDs flash.

The experiment of the 2nd day makes two LEDs flash alternatingly. This is controlled via a programme in ScratchGPIO.

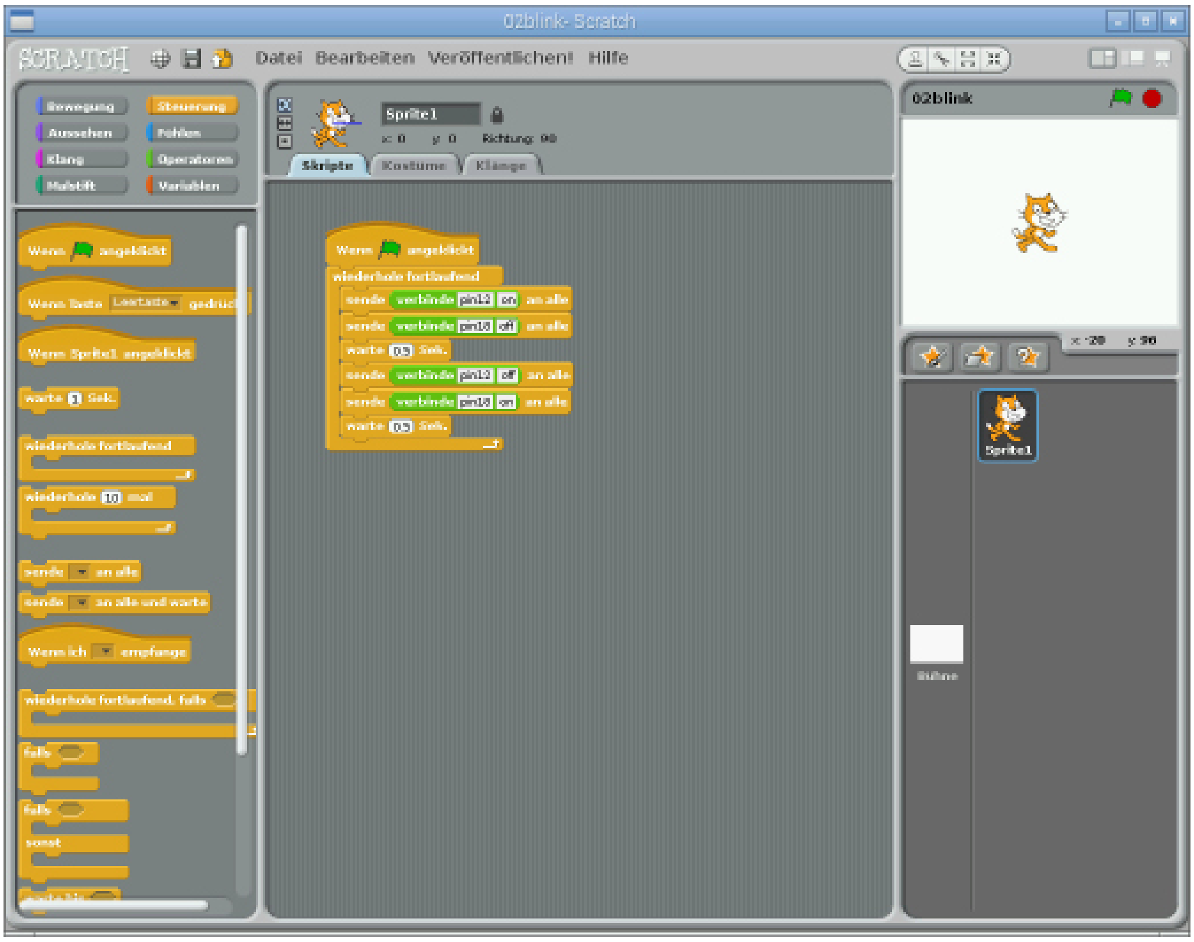
Components: 1 x plug board, 1 x LED red, 1 x LED yellow, 2 x 220-Ohm resistor, 3 x connection cables



Two LEDs flash at the Raspberry Pi.

The program

You do not need to type program code to program in Scratch. The blocks are simply assembled by drag and drop. The block pallet in the left part of the Scratch window contains the available blocks, sorted by subject.



```
Wenn [Flagge] angeklickt
wiederhole fortlaufend
    sende (verbinde [pin12] [on]) an alle
    sende (verbinde [pin18] [off]) an alle
    warte (0.5) Sek.
    sende (verbinde [pin12] [off]) an alle
    sende (verbinde [pin18] [on]) an alle
    warte (0.5) Sek.
```

This Scratch program `02blink` makes the LEDs flash alternatingly

The program starts when the user clicks the green vane at the upper right of the Scratch window.

A **repeated continuous**-loop ensures that the LEDs flash endlessly until the user clicks the red stop icon on the upper right in Scratch.

The GPIO commands are output via the Scratch block **send...to all**. In the text field, a **connect**-block from the green block pallet **operators** is used to connect the respective pin designation and the key word **on** or **off** into a message.

After the red LED at pin 12 has been switched on, and the yellow LED at pin 18 has been switched off, the program will wait for half a second. (Note: Like many other US programs, Scratch uses the period as decimal indicator instead of the comma, as it is customary in Germany.)

Then the red LED at pin 12 is switched off in the same manner, and the yellow LED at pin 18 is switched off.

3. Day

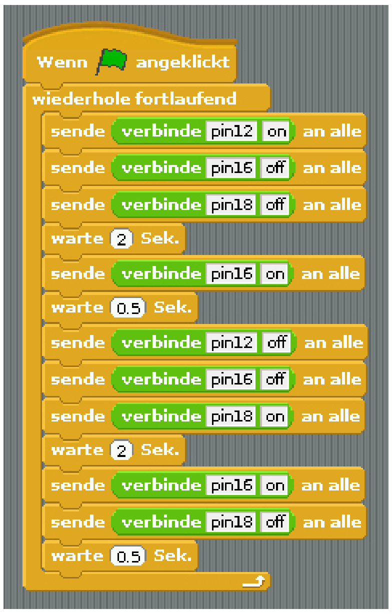
Traffic light

The experiment of the 3rd day switches a traffic light of three LEDs in a typical cycle of red to red and yellow to green and back to yellow and red.

Components: 1 x plug board, 1 x LED red, 1 x LED yellow, 1 x LED green, 3 x 220-Ohm resistor, 4 x connection cables

The program

The programme works similarly as yesterday's. Again, different combinations of LEDs are switched on and off in an endless loop. In the interim phases red-yellow and yellow, the traffic light will be lit for 0.5 seconds each; in the phases red and green, it will be lit for 2 seconds each. These times can also be set differently in the **wait for...sec**-blocks.



```
Wenn [flag] angeklickt
wiederhole fortlaufend
    sende verbinde pin12 on an alle
    sende verbinde pin16 off an alle
    sende verbinde pin18 off an alle
    warte 2 Sek.
    sende verbinde pin16 on an alle
    warte 0.5 Sek.
    sende verbinde pin12 off an alle
    sende verbinde pin16 off an alle
    sende verbinde pin18 on an alle
    warte 2 Sek.
    sende verbinde pin16 on an alle
    sende verbinde pin18 off an alle
    warte 0.5 Sek.
```

The program `03ampel` controls the traffic light

4. Day

RGB-LEDs

A normal LED is always lit in only one colour. The colour of the LEDs in the Advent calendar are evident even with the LED off. There also are LEDs that appear transparent and that will only show their colour when current is flowing. RGB-LEDs can be lit in different colours. Generally, three LEDs with different colours are installed in a transparent housing here. Each of these three LEDs has its own anode, through which it is connected to a GPIO pin. The cathode, which is connected to the ground line, is only present once. Therefore, an RGB-LED has four connection wires.

The connection wires of RGB-LEDs have different lengths to identify them clearly. In contrast to regular LEDs, the cathode is the longest wire here.

RGB-LEDs work like three individual LEDs and therefore also need three dropping resistors.

RGB-LED flashes in different colours

The experiment of the 4th day makes an RGB-LED flash randomly in different colours.

Components: 1 x plug board, 1 x RGB LED, 3 x 220-Ohm resistor, 4 x connection cables

The program

The RGB-LED is connected to three GPIO-Pins with subsequent numbers. This permits control via a random number between 11 and 13. At every loop passage, one of the three pins of the RGB-LED is switched on at random. For this, the program generates a random number in the range 11...13. Two nested **connect**-blocks generate one of these texts from this: **pin11on**, **pin12on**, **pin13on**. In Scratch, numbers and character chains can be linked into character chains as desired, while other programming languages often only permit this after prior conversion. Then a similar block will randomly switch off one of the three pins of the RGB-LED. Since in both cases the switching condition of the respective pin will not be checked first, any imaginable combination of colours switched on and off may result after a few loop passages.

How are random numbers generated?
Generally, it is assumed that nothing can happen at random in a program. – So how can a program be able to generate random numbers? Dividing a large prime number by any value, this will lead to figures that are

hardly foreseeable anymore from the xth decimal digit onwards. They change irregularly if the divisor is regularly increased as well. This result appears random, but can be reproduced by an identical program or by calling the program again at any time. However, if taking a number made up of some of these numbers and once again dividing it by a number that results from the current second of the time or the content of any memory digit of the computer, the result cannot be reproduced and is therefore called a random number.

5. Day

Switching the LED with a button

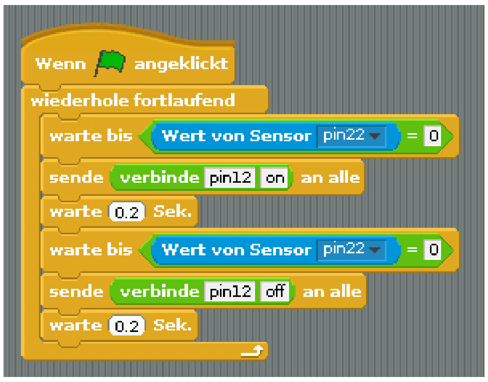
A button closes a connection between the two contact strips while the button is pushed. Releasing the button separates the connection. To switch on an LED every time the button is pushed, all you need is to close a circuit. A program is not necessary. The experiment of the 5th day switches the LED on with one push of a button and off again with the next.

Components: 1 x plug board, 1 x LED red, 1 x 220-Ohm resistor, 1 x button, 4 x connection cables

ScratchGPIO uses internal pull-up resistors in the Raspberry Pi, so that GPIO inputs clearly have the value 1 in the unconnected condition. Connecting this kind of GPIO input to GND will cause it to take the value 0. GPIO inputs can only take the values 1 or 0.

The program

In an endless loop, the program will wait until the user pushes the button. Then the LED is switched on. Then the program will again wait until the user pushes the button. Now the LED is switched off again and the endless loop starts over.



```
Wenn [flag] angeklickt
wiederhole fortlaufend
    warte bis < (Wert von Sensor [pin22]) = [0] >
    sende [verbinde pin12 on] an alle
    warte (0.2) Sek.
    warte bis < (Wert von Sensor [pin22]) = [0] >
    sende [verbinde pin12 off] an alle
    warte (0.2) Sek.
```

The program `05taster` switches the LED with a button

The Scratch block **wait until...** makes the program wait until a specific condition is met. Conditions are blocks with pointed ends, most of them found on the block pallet **operators**.

The current value of a GPIO-input is checked with the block **value of sensor...** of the block pallet **feel**. The list field contains all GPIO inputs for selection.

After the LED has been switched on or off, the program will wait for 0.2 seconds. Such "time-outs" are integrated whenever programmes communicate directly with hardware. Simply said, they prevent a programme from "overdoing it" and missing some hardware event.

6. Day

Controlling the LED with a sensor contact

Not only traffic lights, but also door openers, light switches and automatons are often controlled by sensor contacts that only need to be touched nowadays. Buttons that actually need to be pushed are growing increasingly rare. The experiment of the 6th day controls an LED via a simple sensor contact.

Components: 1 x plug board, 1 x LED red, 1 x 220-Ohm resistor, 1 x 20 MOhm resistor, 2 x wire bridges (sensor contact), 4 x connection cables

The two grey wire bridges are made of short pieces of blank circuit wire. The upper contact is plugged onto the ground line, the lower contact is connected to the GPIO-pin 22. This wire will be needed more often in the next few days to build connection bridges on the plug board.

This is how sensor contacts work:

The GPIO-pin switched as input is connected to +3.3 V via an extremely high-Ohmic resistant (20 MOhm), so that a weak signal that is still clearly defined as high is pending. A person who is not floating freely in the air is always grounded and supplies a low level through electrically conductive skin. When this person touches a sensor contact, the weak high signal is overlaid by the much higher low level of the finger tip, pulling the GPIO-Pin to low level.

The actual height of the resistance between hand and ground depends on many things, including shoes and floor. Barefoot in wet grass offers the best ground connection, but stone floors usually work well, too. Wood floors insulate more strongly, and plastic floorings often even are positively charged. For the circuit to work at all times, an additional ground contact is installed, similarly to sensor buttons at elevators and doors. When it and the actual sensor are touched at the same time, the ground connection is made in any case.

The program

For the sensor contacts to work, the internal pull-up resistors at the GPIO-pins must be switched off first. The ScratchGPIO is always on by default. This is done by a GPIO command **SetPinsNone** at the beginning of the program.

In an endless loop, an **if...else**-query checks if the value of the sensor **pin22** is equal to 0. Then the GPIO-pin is connected to the ground. The sensor has been touched.

In this case, the LED at the GPIO-pin 12 is switched on; otherwise - if the sensor is not touched - it is switched off.

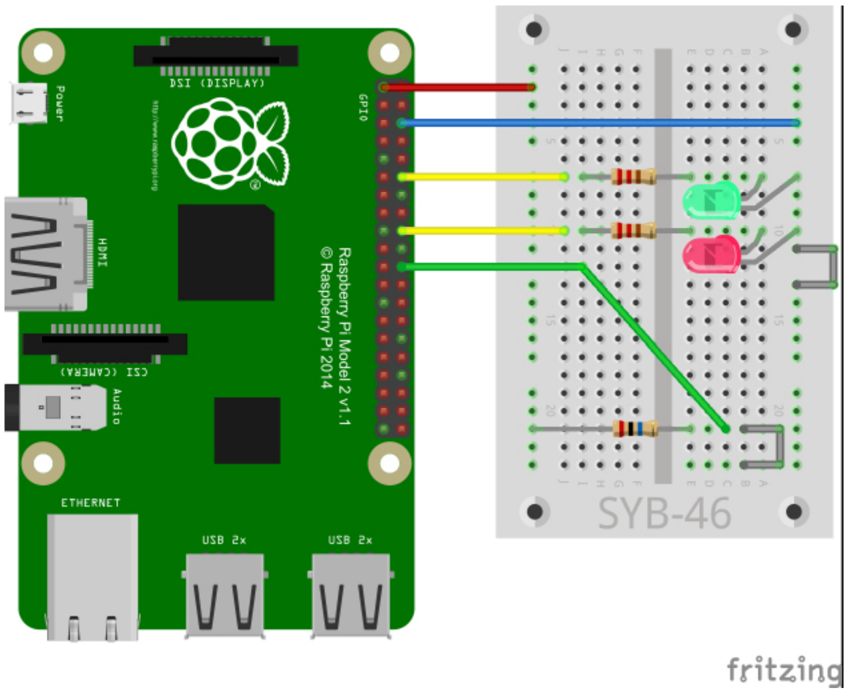
7. Day

Spoon as a sensor

A spoon or other metal object is well suitable as a sensor contact. The spoon is connected to a wire bridge on the plug board with an alligator clamp cable.

Touching the spoon will cause the green LED to light up in the experiment of the 7th day. While it is not touched, the red LED is lit.

Components: 1 x plug board, 1 x LED red, 1 x LED green, 1 x 220-Ohm resistor, 1 x 20 MOhm resistor, 2 x wire bridges (sensor contact), 5 x connection cables, 1 x alligator clamp cable



The alligator clamp cable is connected to the lower wire bridge. The ground contact at the upper wire bridge is only necessary if earthing is insufficient.

The program

The program is similar to yesterday's. When the sensor at the GPIO-pin 22 supplies the value 0, i.e. is connected to the ground, the green LED connected to the GPIO-pin 18 is switched on and the red LED connected to the GPIO-pin 12 off. Otherwise, the red LED is switched on and the green one off.

8. Day

Controlling the RGB LED via sensor contact

Three sensor contacts can control the three colours of an RGB-LED independently of each other. Touching two contacts at the same time will lead to a mixed colour. Touching all three contacts causes the RGB-LED to be lit white.

Components: 1 x plug board, 1 x RGB LED, 3 x 220-Ohm resistor, 3 x 20 MOhm resistor, 4 x wire bridges (sensor contact), 1 x wire bridge (connection wire), 8 x connection cables

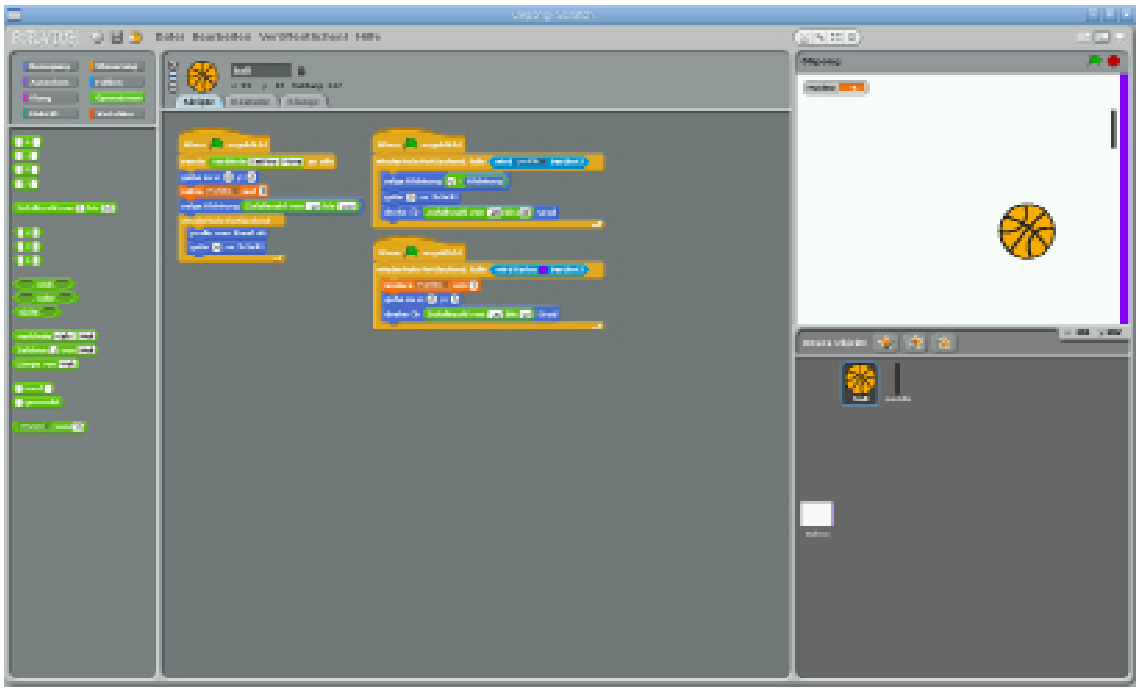
The program

The program will continually query the three sensor contacts at the GPIO-pins 11, 12 and 13 in an endless loop. If a sensor supplies value 0, the corresponding colour component of the RGB-LED is switched on; otherwise, it is off.

9. Day

Pong-game with two sensor contacts

Of course, Scratch can be used for more than programming LEDs. Originally, this programming language was meant to build simple games onscreen. Today's program controls a simple pong game in the retro design with two metal objects that are connected to wire bridges on the plug board via alligator clamp cables.

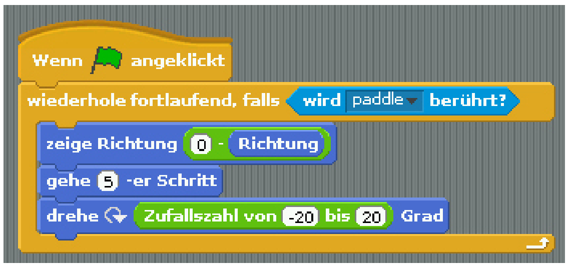

The game 09pong and the Scratch scripts for the ball

In this game, you try to beat a ball that is flying through the room back with a paddle. When the ball touches the coloured line, a point will be deducted from the payer, and the ball will restart in the middle. The paddle is moved up and down with two sensor contacts. The LEDs signal contact of the sensors.

The programme for the ball

The two objects in the game, the ball and the paddle, each have dedicated Scratch scripts, all of which start when clicking the green vane.

At the beginning of the game, the ball is set to the middle of the stage and the points counter is set to 0. The ball starts in a random direction between -20 and -160 degrees. Then it will fly in an endless loop and only change direction when it bounces off of the edge.



```
Wenn [flag] angeklickt
wiederhole fortlaufend, falls <wird (paddle) berührt?>
    zeige Richtung ((0) - (Richtung))
    gehe (5) -er Schritt
    drehe ↻ (Zufallszahl von (-20) bis (20)) Grad
```

This script controls the ball movement when the paddle is touched.

When the ball touches the paddle, the movement direction is inverted. The ball will fly on to the lower left at the same angle at which it came from the upper left, or vice versa - if it came from the lower left, it will continue on to the upper left. Then the ball will fly a small step so as not to touch the paddle again in any case. To make the movement direction a little less foreseeable, the flight direction is changed by a random value between **-20** and **20** degrees as compared to the previous direction.

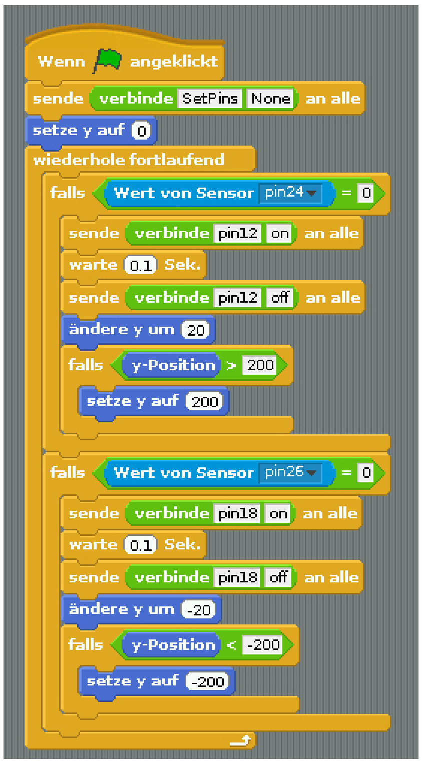
When the ball touches the purple bar, a point is deducted from the player. The variable **points** is increased by 1. Then the ball is put into the centre of the playing field again, to fly off again from there. The flight direction is turned by a random value between **-20** and **20** degrees to keep it from taking precisely the same trajectory again, while still roughly flying in the same direction that it did last.

The electronics for the paddle

Components: 1 x plug board, 1 x LED red, 1 x LED yellow, 2 x 220-Ohm resistor, 2 x 20 MOhm resistor, 2 x wire bridges (sensor contact), 6 x connection cables, 1 x alligator clamp cable

The program for the paddle

The paddle has a dedicated Scratch script that is started when clicking the green vane, and that queries and controls the electronics.



```
Wenn angeklickt
sende verbinde SetPins None an alle
setze y auf 0
wiederhole fortlaufend
    falls Wert von Sensor pin24 = 0
        sende verbinde pin12 on an alle
        warte 0.1 Sek.
        sende verbinde pin12 off an alle
        ändere y um 20
        falls y-Position > 200
            setze y auf 200
    falls Wert von Sensor pin26 = 0
        sende verbinde pin18 on an alle
        warte 0.1 Sek.
        sende verbinde pin18 off an alle
        ändere y um -20
        falls y-Position < -200
            setze y auf -200
```

This script controls the paddle.

The two sensors are queried at the GPIO-pins 24 and 26 in an endless loop. When a sensor is touched, the associated LED is switched on for 0.1 seconds. The movement of the paddle by 20 coordinate units up or down is much more important than the brief lighting up of the LED.

When the paddle has reached the upper edge, it should not move any further. For this, we use an **if**-query to check whether the y-position exceeds 200. If this is the case, the y-position is set to 200. According to the same principle, the position is set to -200 when the paddle has reached the bottom edge.

10. Day

Controlling a flashing light with putty

Putty conducts current about as well as the human skin does. It can easily be formed in any desired shape, and a putty contact is much easier to grasp than a simple piece of wire. The area on which the hand touches the contact is much larger. Thus, a "loose contact" is less likely to occur.

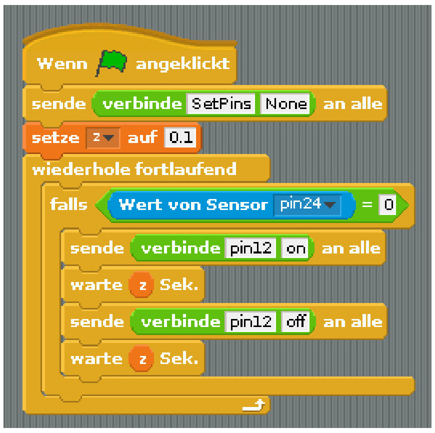
A putty sensor is made of a piece of putty, a wire and an alligator clamp cable.

Components: 1 x plug board, 1 x LED red, 1 x 220-Ohm resistor, 1 x 20 MOhm resistor, 2 x wire bridges (sensor contact), 4 x connection cables, 1 x alligator clamp cable, 1 x putty contact

The experiment of the 10th day no longer simply switches the LED on or off with the sensor contact but lets it flash at an adjustable speed while the putty sensor is being touched.

The program

The program uses a variable that is displayed on the stage with a slider. This way, you can change the flashing frequency while the program is running.



```
Wenn [flag] angeklickt
sende verbinde SetPins None an alle
setze z auf 0.1
wiederhole fortlaufend
    falls Wert von Sensor pin24 = 0
        sende verbinde pin12 on an alle
        warte z Sek.
        sende verbinde pin12 off an alle
        warte z Sek.
```

The program `10blinklicht` controls a flashing light with a putty sensor

When starting the program by clicking the green vane, the variable `z`, which controls the flashing speed, is set to `0.1`. An endless loop queries the sensor at the GPIO-pin 24. When this is touched, the LED will be lit for the time specified in variable `z` and will then be switched off again for the same period.

If the putty sensor is touched for longer than this, a flashing effect will appear. At the end of the loop, the LED will always be off. This way, it will stay off when the putty sensor is no longer touched.

11. Day

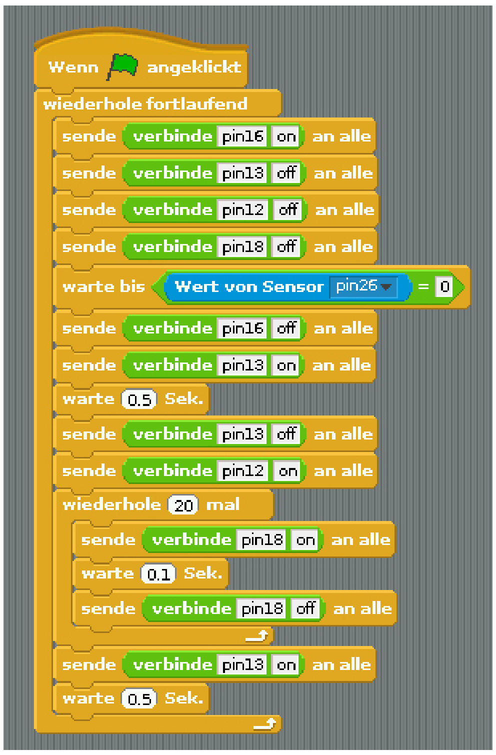
Pedestrian traffic light with flashing light

In some countries, the traffic lights at pedestrian crossings are not red and green, but a light flashes when the traffic light for cars shows red so that one can cross the street.

Components: 1 x plug board, 1 x LED red, 1 x LED yellow, 1 x LED green,1 x LED blue, 4 x 220-Ohm resistor, 1 x button, 1 x wire bridge, 6 x connection cables

The program

The program runs the traffic light cycle in an endless loop. After the traffic light shows green, the loop will wait for a button to be pushed. Then the traffic light cycle will continue until the traffic light shows red again. Then the blue LED will flash 20 times. There is no waiting time after this LED switches off. The LED is only off for a moment, while the script starts the next run of the loop. This way, a flickering light results, in contrast to the even flashing light of earlier programs.



```
Wenn [Flagge] angeklickt
wiederhole fortlaufend
    sende verbinde pin16 on an alle
    sende verbinde pin13 off an alle
    sende verbinde pin12 off an alle
    sende verbinde pin18 off an alle
    warte bis Wert von Sensor pin26 = 0
    sende verbinde pin16 off an alle
    sende verbinde pin13 on an alle
    warte 0.5 Sek.
    sende verbinde pin13 off an alle
    sende verbinde pin12 on an alle
    wiederhole 20 mal
        sende verbinde pin18 on an alle
        warte 0.1 Sek.
        sende verbinde pin18 off an alle
    sende verbinde pin13 on an alle
    warte 0.5 Sek.
```

The program `11fussgaengerblau` controls a pedestrian light with a flash

12. Day

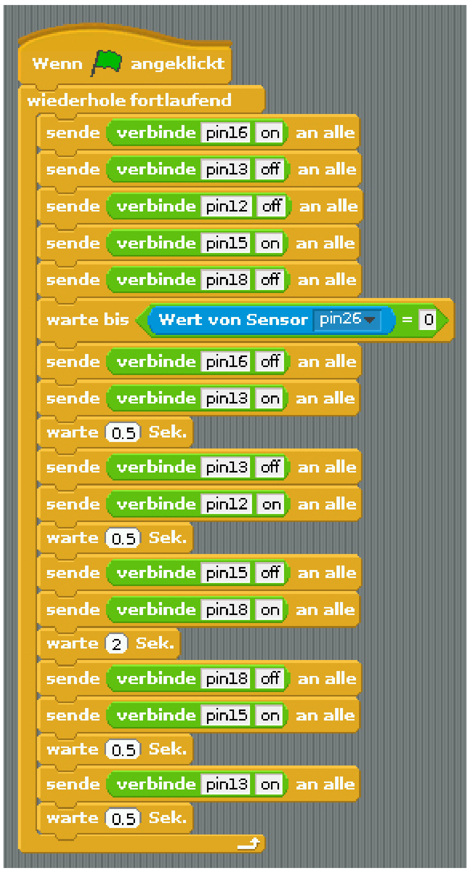
Pedestrian traffic light with red/green traffic light

The experiment of the 12th day is another version of the pedestrian traffic light, but the RGB-LED now shows red or green to the pedestrians.

Components: 1 x plug board, 1 x LED red, 1 x LED yellow, 1 x LED green,1 x RGB LED, 5 x 220-Ohm resistor, 1 x button, 1 x wire bridge, 7 x connection cables

The program

The program runs similarly to the previous version, but will initially additionally switch on the GPIO-pin to which the red colour of the RGB-LED is connected. After the traffic light has switched to red, the red colour of the RGB-LED is switched off for 2 seconds at the GPIO-pin 15, and the green colour of the RGB-LED at the GPIO-pin 18 is switched on.



```
Wenn [grüne Flagge] angeklickt
wiederhole fortlaufend
    sende verbinde pin16 on an alle
    sende verbinde pin13 off an alle
    sende verbinde pin12 off an alle
    sende verbinde pin15 on an alle
    sende verbinde pin18 off an alle
    warte bis Wert von Sensor pin26 = 0
    sende verbinde pin16 off an alle
    sende verbinde pin13 on an alle
    warte 0.5 Sek.
    sende verbinde pin13 off an alle
    sende verbinde pin12 on an alle
    warte 0.5 Sek.
    sende verbinde pin15 off an alle
    sende verbinde pin18 on an alle
    warte 2 Sek.
    sende verbinde pin18 off an alle
    sende verbinde pin15 on an alle
    warte 0.5 Sek.
    sende verbinde pin13 on an alle
    warte 0.5 Sek.
```

The program `12fussgaengerrot` controls a pedestrian traffic light with RGB-LED

13. Day

RGB flash effects

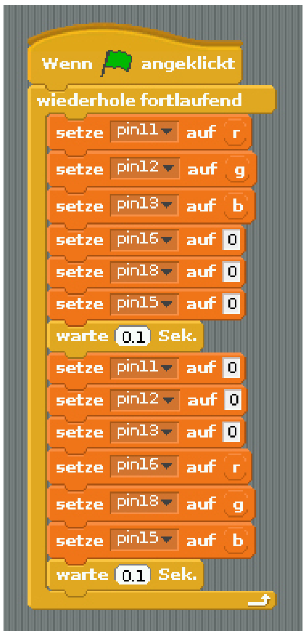
The experiment of the 13th day makes two RGB LEDs flash alternatingly. The colours can be switched on interactively on the screen.

Components: 1 x plug board, 2 x RGB LED, 6 x 220-Ohm resistor, 1 x wire bridge (connection wide), 7 x connection cables

The program

The program shows another method for setting output pins in ScratchGPIO. It uses three variables, one per colour. The variables `r`, `g` and `b` are displayed with controllers on the stage. Here, the user can switch the three colour components on and off interactively. Additionally, there are six variables for the pins used for the two RGB-LEDs. They are set to the corresponding values.

In an endless loop, the three GPIO-pins `11`, `12` and `13` of the first RGB-LED are set to the specified colour values, and the three GPIO-pins `16`, `18` and `15` of the second RGB-LED are switched off. After a brief waiting time, the first RGB-LED is switched off and the second one is given the set colour values. Alternatingly switching the LEDs will lead to an alternating flash.



```
Wenn [grüne Flagge] angeklickt
wiederhole fortlaufend
    setze pin11 auf r
    setze pin12 auf g
    setze pin13 auf b
    setze pin16 auf 0
    setze pin18 auf 0
    setze pin15 auf 0
    warte 0.1 Sek.
    setze pin11 auf 0
    setze pin12 auf 0
    setze pin13 auf 0
    setze pin16 auf r
    setze pin18 auf g
    setze pin15 auf b
    warte 0.1 Sek.
```

The program `13rgb2` makes two RGB-LEDs flash alternatingly.

14. Day

Crab crawling

The experiment of the 14th day is a quick game on the screen that is placed with a small game pad with two buttons. In this game, a crab runs on a round course and should veer off of the path as little as possible.

The crab crawls forward on its own. You can use the buttons to turn the crab to chance the direction in which it crawls.

Components: 1 x plug board, 2 x button, 1 x wire bridge (connection wire), 3 x connection cables

The program

When clicking the green vane, the crab is put in the starting position and the error counter is reset to 0. This starts an endless loop in which the crab moves a small step at each pass.

Then a check is performed whether the red or green claw of the crab touches the white background. In this case, the crab has veered off of its path and an error point will be assigned.

Two more queries check if one of the two buttons has been pushed. If this is the case, the crab is turned by 5 degrees for every push of the button in the respective direction.

15. Day

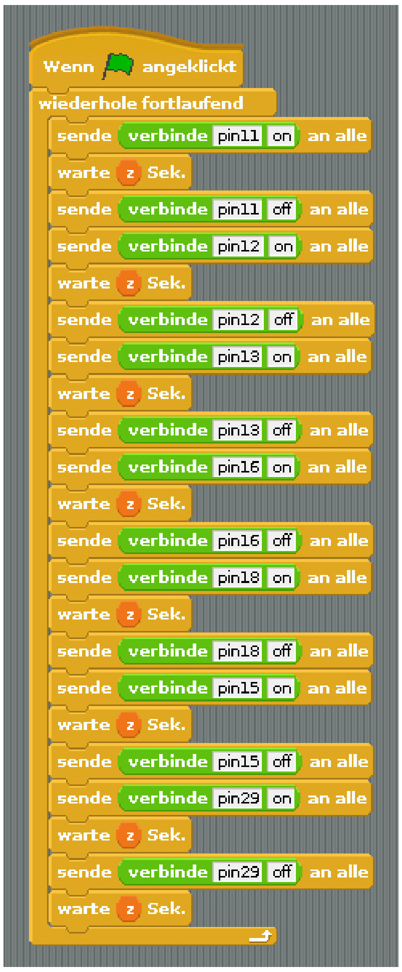
Running light

Running lights are popular effects, not only for advertisement and party rooms. The experiment of the 15th day makes seven LEDs light up as a running light. The three RGB-LEDs are connected with one colour only each.

Components: 1 x plug board, 1 x LED red, 1 x LED yellow, 1 x LED green, 1 x LED blue, 3 x RGB-LED, 7 x 220-Ohm resistor, 3 x wire bridge (connection wire), 8 x connection cables

The program

In an endless loop, the LEDs are switched on briefly in sequence. The time is saved in the variable `z` that can be set interactively via a controller on the screen.



```
Wenn angeklickt
wiederhole fortlaufend
    sende verbinde pin11 on an alle
    warte z Sek.
    sende verbinde pin11 off an alle
    sende verbinde pin12 on an alle
    warte z Sek.
    sende verbinde pin12 off an alle
    sende verbinde pin13 on an alle
    warte z Sek.
    sende verbinde pin13 off an alle
    sende verbinde pin16 on an alle
    warte z Sek.
    sende verbinde pin16 off an alle
    sende verbinde pin18 on an alle
    warte z Sek.
    sende verbinde pin18 off an alle
    sende verbinde pin15 on an alle
    warte z Sek.
    sende verbinde pin15 off an alle
    sende verbinde pin29 on an alle
    warte z Sek.
    sende verbinde pin29 off an alle
    warte z Sek.
```

The program `15lauflicht` makes seven LEDs flash alternatingly as a running light

16. Day

Controlling the RGB LED via sensor contact

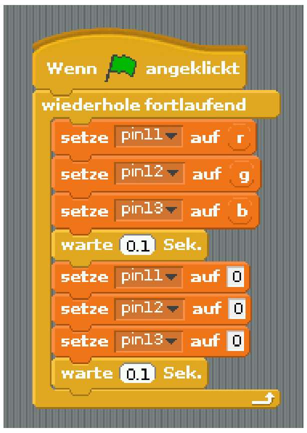
Three sensor contacts can control the three colours of an RGB-LED independently of each other. Touching two contacts at the same time will lead to a mixed colour. Touching all three contacts causes the RGB-LED to be lit white.

Components: 1 x plug board, 1 x RGB LED, 3 x 220-Ohm resistor, 3 x 20 MOhm resistor, 4 x wire bridges (sensor contact), 1 x wire bridge (connection wire), 8 x connection cables, 3 x alligator clamp cables, 3 x putty contact

Three putty contacts are connected to the three wire bridges via alligator clamp cables. The wire bridge at the very tip in the figure, the ground contact, is only used if necessary.

The program

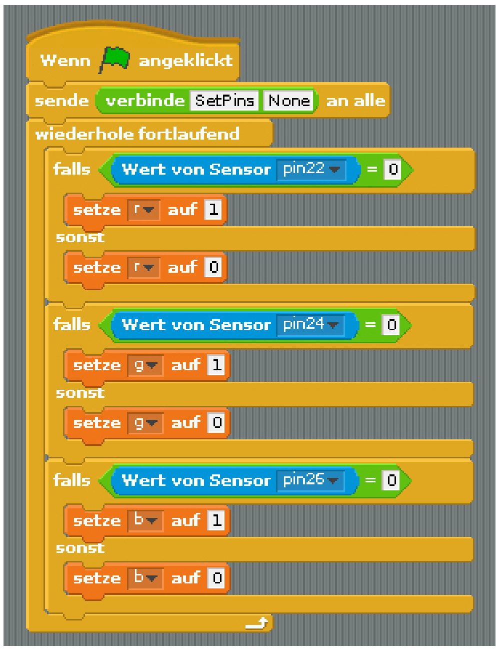
The program is based on an earlier program where the RGB values were saved in variables and written into the variables associated with the GPIO pins via **set...to**-blocks.



```
Wenn [Flagge] angeklickt
wiederhole fortlaufend
    setze pin11 auf r
    setze pin12 auf g
    setze pin13 auf b
    warte 0.1 Sek.
    setze pin11 auf 0
    setze pin12 auf 0
    setze pin13 auf 0
    warte 0.1 Sek.
```

The program `16rgb2-sensor` makes an RGB-LED flash.

A second script block that is also started when clicking the green vane will query the three putty sensors and set the variables `r`, `g` and `b` accordingly.



```
Wenn [Flagge] angeklickt
sende verbinde SetPins None an alle
wiederhole fortlaufend
    falls Wert von Sensor pin22 = 0
        setze r auf 1
    sonst
        setze r auf 0
    falls Wert von Sensor pin24 = 0
        setze g auf 1
    sonst
        setze g auf 0
    falls Wert von Sensor pin26 = 0
        setze b auf 1
    sonst
        setze b auf 0
```

The second script block queries the sensors.

17. Day

Car race

Two buttons are used to control a car in a simple race. While the car is on the road, the green LED on the gaming controller is lit; if the car veers off to the left or right, one of the two red ones is lit.

Components: 1 x plug board, 2 x LED red, 1 x LED green, 2 x 220-Ohm resistor, 2 x button, 2 x wire bridge (connection wire), 6 x connection cables

The program
Every time the car reaches the upper stage edge, a new route image will appear and the car will start at the lower stage edge again.

The car is controlled via two script blocks. The first block is made up of two nested endless loops. The outer loop places the car at the lower stage edge, waits until it has reached the top and then sends a message **background**, to which the background reacts and changes.

The inner loop moves the car while the edge is not touched, and checks whether a black wheel of the car touches the light-green meadow to the left of the street at every movement. If so, the left red LED lights up and the green one is switched off. If the meadow is not touched, another query will check whether the dark green meadow to the right of the street is touched. In this case, the right red LED is switched on and the green one off as well. If this is not the case either, the car is still on the road and the green LED remains on.

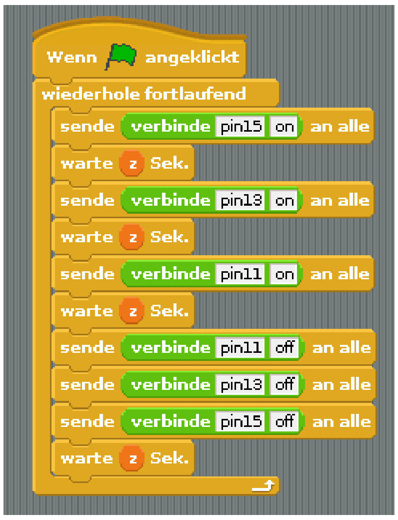
The stage has two more simple scripts. At the beginning, when clicking the green vane, the first background image is activated; every time the message **background** is received, the next background image is shown.

18. Day

Effect lights
GPIO-pins can also control more than one LED; however, each LED should still have its own dropping resistor. The experiment of the 18th day shows a light effect on six LEDs that only need three GPIO-pins, however.

Components: 1 x plug board, 2 x LED red, 2 x LED yellow, 1 x LED green,1 x LED blue, 6 x 220-Ohm resistor, 3 x wire bridge, 4 x connection cables

The program
The GPIO-pins are switched on in sequence in an endless loop. In contrast to the running light, they remain on and will only be switched off together at the end. The time between the switching processes is saved in the variable z that can be set interactively via a controller on the screen.



```
Wenn [flag] angeklickt
wiederhole fortlaufend
    sende verbinde pin15 on an alle
    warte z Sek.
    sende verbinde pin13 on an alle
    warte z Sek.
    sende verbinde pin11 on an alle
    warte z Sek.
    sende verbinde pin11 off an alle
    sende verbinde pin13 off an alle
    sende verbinde pin15 off an alle
    warte z Sek.
```

The program `18lichteffekt` controls the light effect at the three GPIO pins.

19. Day

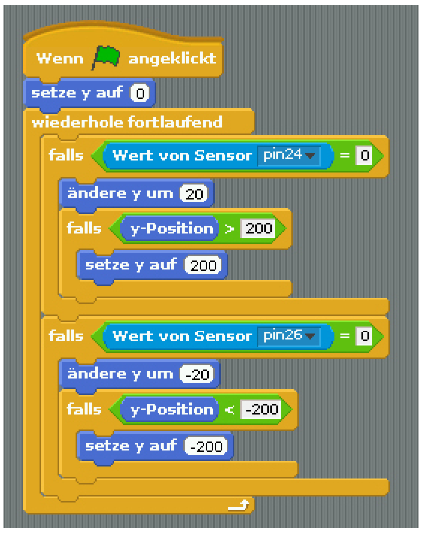
Tennis game
The experiment of the 19th day is a gaming classic of the early home computer era. Two players try to beat a ball across the playing field. When the opponent's base line is hit, a point is awarded. The program is based on the pong game from a few days ago.

Components: 1 x plug board, 4 x 20 MOhm resistor, 5 x wire bridges (sensor contact), 6 x connection cables, 4 x alligator clamp cable, 4 x putty contact

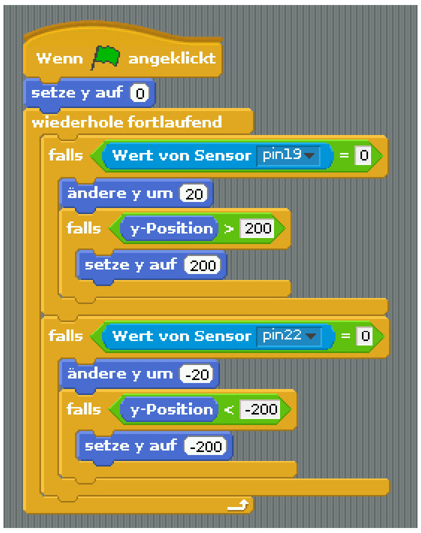
The program

The script for the ball makes it bounce off of the edges as in the pong game. If one of the two paddles is touched, the flight direction is changed a little at random.

Two variables save the points of the players. Each player is awarded a point when the ball touches the opponent's base line. The two paddles have program blocks that query two sensor contacts each in endless loops and move the paddle up or down when touched.



```
Wenn  angeklickt
setze y auf 0
wiederhole fortlaufend
    falls Wert von Sensor pin24 = 0
        ändere y um 20
        falls y-Position > 200
            setze y auf 200
    falls Wert von Sensor pin26 = 0
        ändere y um -20
        falls y-Position < -200
            setze y auf -200
```



```
Wenn  angeklickt
setze y auf 0
wiederhole fortlaufend
    falls Wert von Sensor pin19 = 0
        ändere y um 20
        falls y-Position > 200
            setze y auf 200
    falls Wert von Sensor pin22 = 0
        ändere y um -20
        falls y-Position < -200
            setze y auf -200
```

20. Day

Dice with LEDs

Everyone knows and owns the typical game dice with 1 - 6 pips. Electronically controlled dice that make the pipes light up at the push of a button is much cooler - but they shouldn't just light up as 1 - 6 LEDs in a row, but in the placement as on game dice. The rolled number is displayed until the next number is rolled.

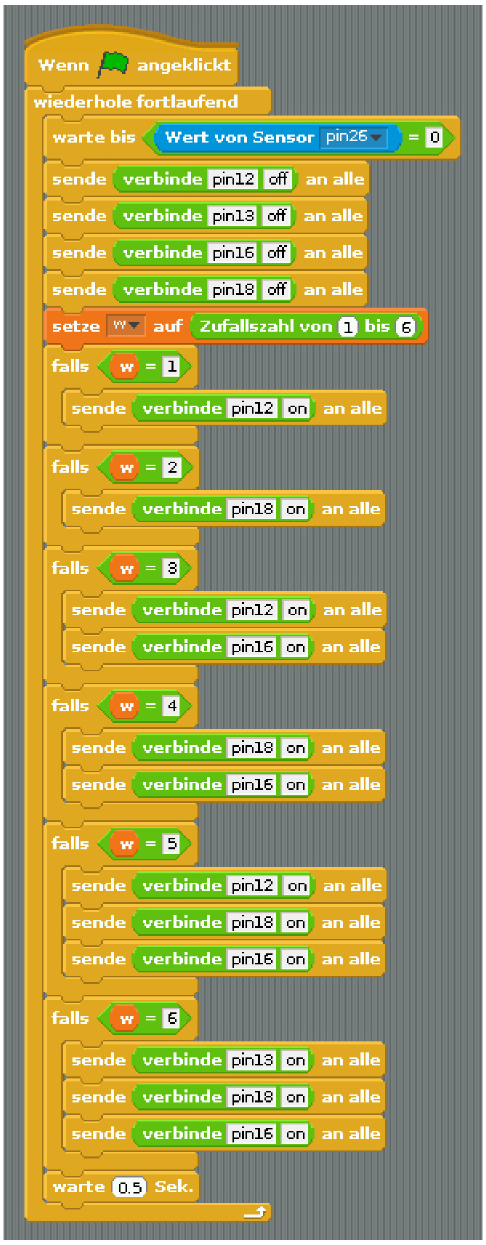
Components: 1 x plug board, 2 x LED red, 2 x LED yellow, 2 x LED green,1 x LED blue, 7 x 220-Ohm resistor, 1 x button, 1 x wire bridge, 6 x connection cables

The program

An endless loop first waits until the button at pin 26 is pushed. Now the four GPIO-pins used are switched off and the previously displayed rolled result is deleted.

Then a random number between 1 and 6 is generated and saved in the variable w. After the number has been rolled, six **if**-blocks follow for every possible value that can be rolled. Each of these blocks switches on the corresponding combination of LEDs when a specific number is rolled.

Independently of the rolled result, the programme will always wait for half a second after rolling, to keep from tripping two different dice actions by button bouncing.



```
Wenn [flag] angeklickt
wiederhole fortlaufend
    warte bis (Wert von Sensor pin26 = 0)
    sende verbinde pin12 off an alle
    sende verbinde pin13 off an alle
    sende verbinde pin16 off an alle
    sende verbinde pin18 off an alle
    setze w auf Zufallszahl von 1 bis 6
    falls (w = 1)
        sende verbinde pin12 on an alle
    falls (w = 2)
        sende verbinde pin18 on an alle
    falls (w = 3)
        sende verbinde pin12 on an alle
        sende verbinde pin16 on an alle
    falls (w = 4)
        sende verbinde pin18 on an alle
        sende verbinde pin16 on an alle
    falls (w = 5)
        sende verbinde pin12 on an alle
        sende verbinde pin18 on an alle
        sende verbinde pin16 on an alle
    falls (w = 6)
        sende verbinde pin13 on an alle
        sende verbinde pin18 on an alle
        sende verbinde pin16 on an alle
    warte 0.5 Sek.
```

The program `20wuerfel` controls the LED dice

21. Day

RGB light control panel

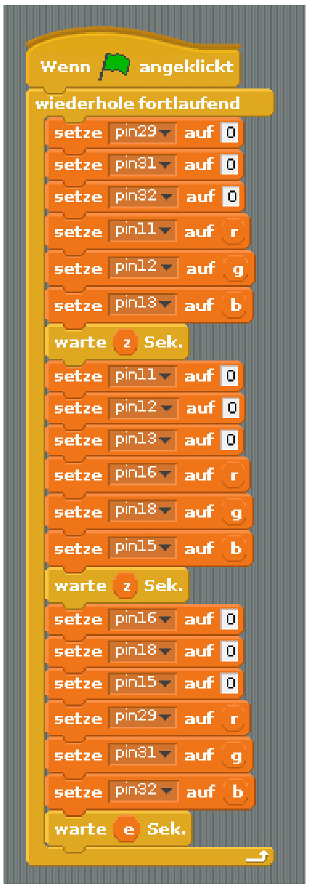
The experiment of the 21st day offers various light effects with three RGB-LEDs. The colours and effects can be switched on interactively on the screen.

Components: 1 x plug board, 3 x RGB LED, 9 x 220-Ohm resistor, 2 x wire bridge (connection wide), 10 x connection cables

The program

The program uses five variables that can be set via controllers on the stage. With the variables `r`, `g` and `b`, the user can interactively switch the three colour components on and off. The variable `z` specifies the waiting time between flashing of two subsequent LEDs. The variable `e` specifies the time at the end of a loop until the first LED lights up again.

In an endless loop, the three GPIO-pins of one RGB-LED are set to the specified colour values, and the three GPIO-pins of the previously lit RGB-LED are switched off.



```
Wenn [Flagge] angeklickt
wiederhole fortlaufend
    setze pin29 auf 0
    setze pin31 auf 0
    setze pin32 auf 0
    setze pin11 auf r
    setze pin12 auf g
    setze pin13 auf b
    warte z Sek.
    setze pin11 auf 0
    setze pin12 auf 0
    setze pin13 auf 0
    setze pin16 auf r
    setze pin18 auf g
    setze pin15 auf b
    warte z Sek.
    setze pin16 auf 0
    setze pin18 auf 0
    setze pin15 auf 0
    setze pin29 auf r
    setze pin31 auf g
    setze pin32 auf b
    warte e Sek.
```

The program `21rgb3` controls the light effect at the three RGB-LEDs

22. Day

Today, the advent calendar offers the manual on the Conrad experimenting package »Raspberry Pi - verstehen und anwenden« for download as an E-Book. This book contains various further experiments with the Raspberry Pi that are implemented with the pre-installed programming language Python.

Many of the components used are already included in the Advent calendar; others are found in electronic crafts boxes or can be purchased at **www.conrad.de**.

Dice with dice effect

An electronic dice that shows a number does not appear particularly realistic. The experiment of the 22nd day makes four numbers flash briefly before the dice shows a final result.

Components: 1 x plug board, 2 x LED red, 2 x LED yellow, 2 x LED green,1 x LED blue, 7 x 220-Ohm resistor, 1 x button, 1 x wire bridge, 6 x connection cables

The program
The program consists of two blocks. After clicking the green vane, an endless loop in the main program will wait for a button to be pushed. Then the message **roll** is sent five times, each time calling the program block to display a random rolling result. The time intervals between the rolled indications grow in length. This is what real dice look like when they come to a halt slowly.

23. Day

Christmas songs
The experiment of the 23rd day is a simple little piano that can be used to play five tones via putty contacts. The sixth contact, the ground line, is only used when necessary.

Components: 1 x plug board, 5 x 20 MOhm resistor, 6 x wire bridges (sensor contact), 7 x connection cables, 5 x alligator clamp cable, 5 x putty contact

Of course, the Raspberry Pi can play any sounds. To be able to play music sensibly with the five available putty contacts, we use the pentatonic, a scale that only has five tones - and the oldest historically documented scale, in use as early as in bone flutes from 3000 B.C.

Since the human brain is well able to remember five tones, the pentatonic scale with the notes C, D, E, G, A is often used in children's songs and advertising music. A few examples:

Backe, backe, Kuchen (G-G-A-A-G-E)
Laterne, Laterne, Sonne, Mond und Sterne (A-G-E, A-G-E, G-G-A-A-G-E)
Old Mac Donald had a farm, hea hea ho! (C-C-C-G-A-A-G, E-E-D-D-C)

The program
An endless loop will query all five sensor contacts in sequence. When a sensor is touched, the corresponding note is output. The Raspberry Pi can play music via an HDMI-monitor or an external speaker or headphones connected to the 3.5-mm jack plug socket.

If you don't hear anything ...
... call `sudo raspi-config` in an LXTerminal window and manually select the right audio output in **Advanced Options/Audio**.

24. Day

Christmas tree decoration

The experiment of the 24th day is a flashing Christmas tree decoration.

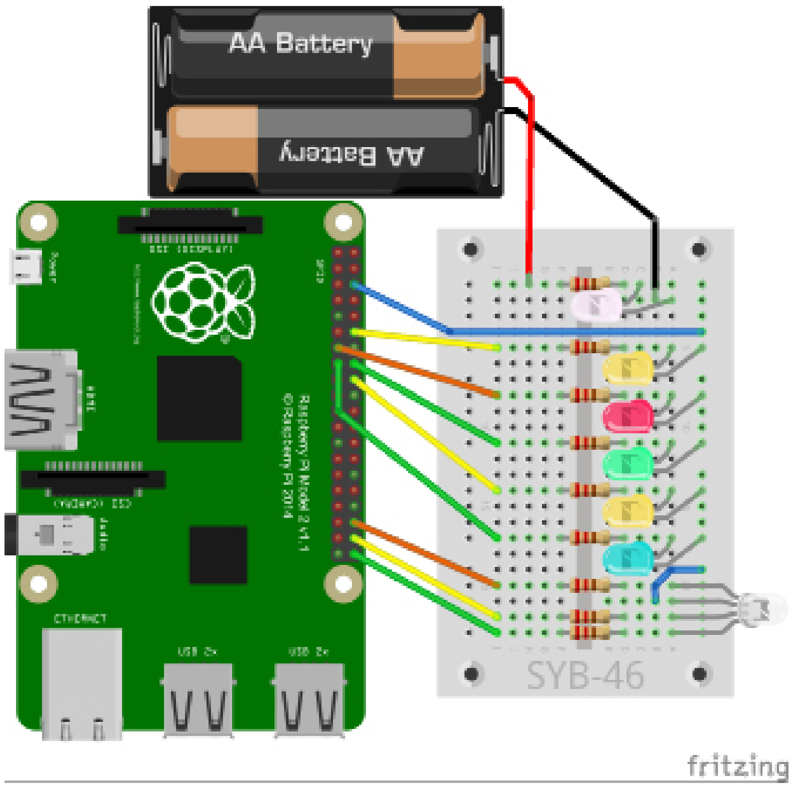
Components: 1 x plug board, 1 x LED red, 2 x LED yellow, 1 x LED green, 1 x LED blue, 1 x RGB-LED, 1 x flashing LED, 9 x 220-Ohm resistor, 1 x wire bridge (connection wire), 10 x connection cables

The program

The five LEDs and the RGB-LED flash entirely at random program-controlledly. The flashing LED from today's door of the Advent calendar will even flash without a program running. The program uses a list in which the GPIO-pin numbers of the LEDs used are saved. List variables can have any length in Scratch and may include numbers as well as character chains. After clicking the green vane, all list elements are deleted first. If the program has been run before, the list would otherwise already contain values. Then the pin numbers of the five LEDs and the three colours of the RGB-LED are saved in the list in sequence. Then an endless loop switches on a GPIO pin randomly selected from the list, and switches off another randomly selected GPIO pin. The program contains no time delay. Since all pins are selected at random, the LEDs will flash for different times. The RGB-LED lights up in colours that are mixed randomly.

Flashing LED without Raspberry Pi

With an additional power supply of 3 V (e.g. two 1.5-V-batteries), the flashing LED can flash even without the Raspberry Pi running.



Christmas tree decoration with flashing LEDs and external power supply

In this circuit, the flashing LED is connected to its own power supply, independently of the Raspberry Pi. This way, it can flash even without the Raspberry Pi running. The program remains unchanged since it does not influence the flashing LED.