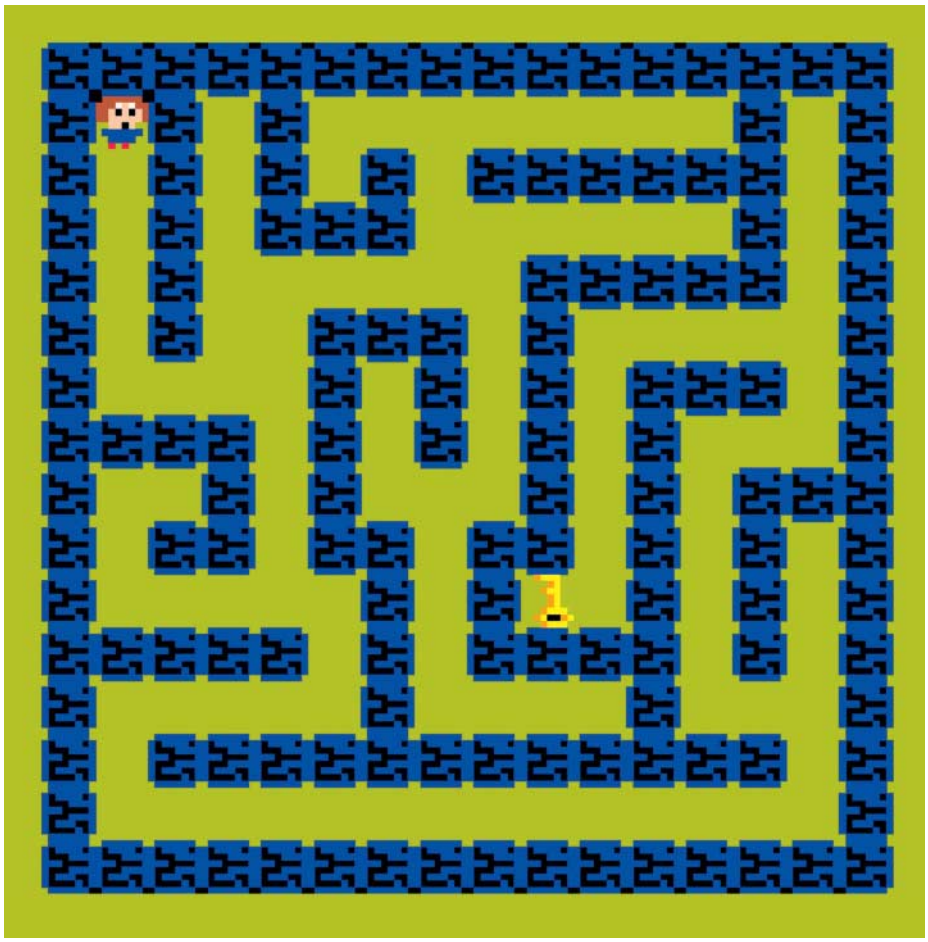


Pixelhelden

Ein Labyrinthspiel mit Pico-8 programmieren

Wer hat nicht schon mal davon geträumt, sein eigenes Spiel zu programmieren? Mit der virtuellen Spielekonsole Pico-8 gelingt das im Handumdrehen, selbst wenn man noch nie zuvor eine Zeile Code geschrieben hat.

Von Caroline Berger

Retro-Spiele fluten derzeit sämtliche App-Stores und beleben den Charme einer längst vergangenen Computerspielepoche. Genau das macht auch Pico-8 von Lexaloffle: Mit der virtuellen Retro-Spielekonsole für PCs und Raspi kann man hervorragend in Pixelwelten abtauchen und von der Community programmierte Spiele zocken. Aber noch viel spannender: Pico-8 hat die Skriptsprache Lua an Bord, mit der man eigene Spiele programmieren kann. Dazu bringt Pico-8 mehrere Editoren für Grafiken, Sounds, Musik, Karten und Programmcode mit. Im Beispiel entsteht ein einfaches Labyrinthspiel, in dem eine Figur einen Schlüssel finden muss. Hat sie ihn gefunden, endet das Spiel.

Laden Sie Pico-8 von der Herstellerseite herunter (ct.de/ymr3). Nach dem Start begrüßt Sie das Programm mit einer kurzen Animation, bevor es Sie in seine Kommandozeile schickt. Aus dieser heraus steuern Sie Pico-8 mit Texteingaben – alle wichtigen Befehle finden Sie in der Tabelle auf Seite 121. Nicht wundern, das Terminal kennt ausschließlich Großbuchstaben und wandelt Kleingeschriebenes von selbst um.

Tippen Sie als Erstes den Befehl `INSTALL_DEMOS` in die Konsole und laden Sie sämtliche Demo-Programme des Entwicklers herunter. Mit `CD DEMOS` wechseln Sie in den gerade neu erstellten Ordner und `LS` zeigt den Inhalt des Ordners an. Tippen Sie nun `LOAD HELLO` ein, um damit eines der gerade heruntergeladenen Programme zu laden. Mit `RUN` starten Sie das Programm. Sie sollten nun mit einer weiteren bunten Animation begrüßt werden. Beenden Sie das Programm mit Escape und löschen Sie mit `CLS` den Bildschirminhalt. Anschließend können Sie mit `LOAD` und `RUN` weitere Demoprogramme ausprobieren, etwa das 2D-Jump'n'Run JELPI.

Einen Charakter entwerfen

Nach den ersten Gehversuchen mit der Konsole können Sie bereits selbst kreativ werden. Tippen Sie `REBOOT` ein. Nachdem Pico-8 neu gestartet wurde, drücken Sie Escape, um den Code-Editor zu öffnen. Hier tippen Sie später den Programmcode ein. Hinter den Symbolen am Bildschirmrand rechts oben verstecken sich weitere Editoren: Öffnen Sie als Erstes den Sprite-Editor mit einem Klick auf das zweiten Symbol von links. Sprites sind in Pico-8 8 × 8 Pixel große Grafiken, die aus maximal 16 Farben bestehen können. Auf der rechten Seite wählen Sie die Farben aus und malen mit dem Stiftwerkzeug in den Editor am linken Bildrand. Im schwarzen Bereich unter dem Editor landen die fertigen Sprites. Experimentieren Sie mit dem Editor und zeichnen Sie eine Figur, die Sie später durch das Labyrinth steuern wollen. Um in den Code-Editor zurückzukehren, klickt man in der Menüleiste auf das Symbol mit den beiden Klammern.

Um ein Gefühl fürs Programmieren mit Pico-8 zu bekommen, starten Sie mit einem kleinen Experiment. Das Ziel: Per Code zeichnen Sie ein Objekt auf den Bildschirm. Auf Groß- und Kleinschreibung müssen Sie übrigens auch im Code-Editor nicht achten – auch er wandelt Geschriebenes in Großbuchstaben um.

Der Bildschirminhalt wird für die Programmierung von Pico-8 in ein Koordinatensystem mit X- und Y-Werten aufgeteilt. Der Ursprung befindet sich am Bildschirmrand links oben. Die Werte für X und Y zeigen an, wo sich ein Objekt befindet. Das Prinzip lässt sich mit dem nachfolgenden Code nachvollziehen:

```
X=64
Y=64
CLS()
CIRC(X,Y,7,8)
```

Nachdem Sie den Code eingetippt haben, drücken Sie Escape und führen ihn im Terminal mit `RUN` aus. `CLS()` löscht den aktuellen Bildschirminhalt (Clear Screen) und anschließend sollte das Programm mittig einen roten Kreis auf den Bildschirm zeichnen. Verändern Sie die Position des Kreises, indem Sie mit verschiedenen X- und Y-Werten experimentieren. Ändern Sie Farbe und Größe des Kreises, indem Sie die Werte in der Klammer hinter `CIRC` verändern. Das Kreis-Experiment dient als Basis für das Labyrinthspiel: Durch Erweiterungen und Anpassungen bauen Sie Ihre ersten Code-Zeilen nach und nach zum fertigen Spiel aus.

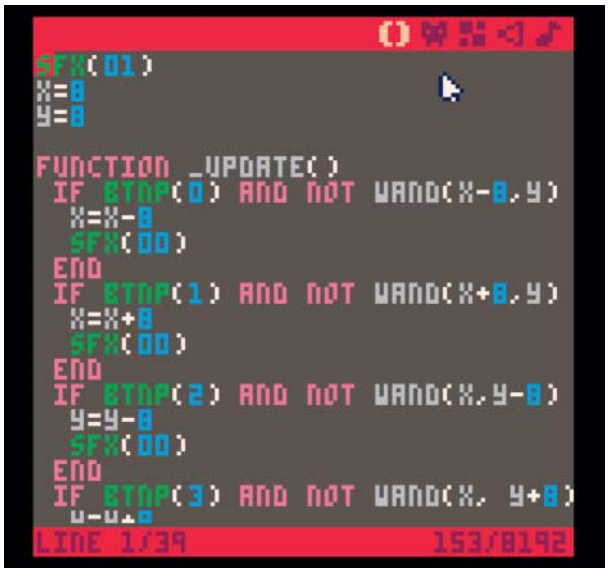
Damit sich Ihre Figur später durchs Labyrinth bewegen lässt, bringt Pico-8 zwei Funktionen mit: `_DRAW` und `_UPDATE`. 30 Mal pro Sekunde werden beide Funktionen automatisch aufgerufen und führen aus, was ihnen einprogrammiert wurde. Kurz gesagt ist `_DRAW` für alles zuständig, was auf dem Bildschirm zu sehen ist, während `_UPDATE` die Bewegungen verantwortet. Passen Sie das obere Programm folgendermaßen an:

```
X=0
Y=64
FUNCTION _UPDATE()
   X=X+1
END
FUNCTION _DRAW()
   CLS()
   CIRC(X,Y,7,8)
   PRINT("WERT VON X: " .. X)
   PRINT("WERT VON Y: " .. Y)
END
```

Jedes Mal, wenn `_UPDATE` aufgerufen wird, erhöht das Programm den Wert von `X` um 1. Und jedes Mal, wenn `_DRAW` aufgerufen wird, zeichnet das Programm einen neuen Kreis an der neu berechneten Stelle. Das geht so schnell vonstatten, dass der Kreis sich durch diesen Codeschnipsel in einer flüssigen Bewegung von links nach rechts über den Bildschirm bewegt. Das Programm beenden Sie wieder mit Escape.

Steuerung programmieren

Nach dem Prinzip der eben programmierten Bewegungsabläufe lässt sich später auch die Figur im Labyrinth bewegen. Für die Steuerung per Pfeiltasten nutzen Sie `IF/THEN`-, also `wenn/dann`-Abfragen. `IF` prüft, ob eine Bedingung wahr ist. Ein Beispiel: `IF X=10 THEN PRINT("X IST 10!") END`. Dies kann man fast direkt ins Deutsche übersetzen: „Wenn X=10 dann schreibe: X ist 10!“. Genau das tut diese Codezeile auch. Die geplante Spielsteuerung lässt sich so beschreiben „Wenn Pfeiltaste nach oben gedrückt, dann laufe nach oben!“ Ob ein Knopf gedrückt wurde, prüft `BTNP()` („button pressed“).



Im Editor programmieren Sie Ihr Spiel. Hinter den pixeligen Symbolen rechts oben verstecken sich die Editoren für Code, Sprites, Leveldesign, Sound und Musik.

Der c’t-Tipp für Kinder und Eltern

Spieleprogrammierung mit Pico-8

- Computer mit Windows, Linux oder macOS, alternativ Raspberry Pi mit Raspbian, Internetzugang, Pico-8
- grundlegende Computerkenntnisse, erste Englischkenntnisse von Vorteil
- Unser Labyrinthspiel programmiert man in rund einer Stunde nach.
- Kinder ab circa 12 Jahren legen alleine los, jüngere Kinder sind auf die Hilfe von Eltern oder älteren Geschwistern angewiesen.
- circa 15 Euro für die Software Pico-8 von Lexaloffle

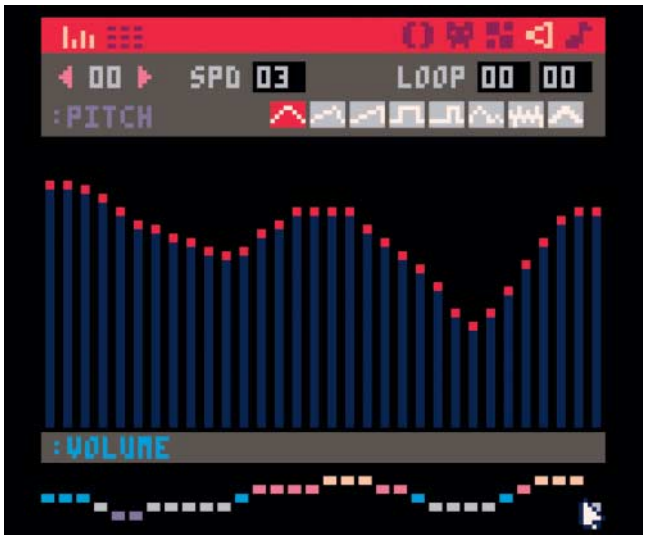
Ersetzen Sie die Update-Funktion in Ihrem Code durch den unten stehenden Schnipsel:

```
FUNCTION _UPDATE()
   IF BTNP(0) THEN
      X=X-8
   END
   IF BTNP(1) THEN
      X=X+8
   END
   IF BTNP(2) THEN
      Y=Y-8
   END
   IF BTNP(3) THEN
      Y=Y+8
   END
END
```

Die Zahlen `0` bis `3` innerhalb der Klammern von BTNP() stehen für die vier Pfeiltasten. Der Kreis lässt sich jetzt bequem mit den Tasten in alle vier Himmelsrichtungen über den Bildschirm scheuchen – die Steuerung ist damit bereits fertig.

Wechseln Sie anschließend zurück zum Sprite-Editor. Jedes Sprite besitzt einen individuellen dreistelligen Zahlencode, den Sie neben dem kleinen Vorschaubild finden – das gilt auch für Ihre Figur. Merken Sie sich die Zahl und ersetzen Sie nun in Ihrem Programm die komplette Zeile der `CIRC`-Funktion mit dem unten stehenden Code.

Die Heldin für das Computerspiel kreiert man mit wenigen Mausklicks im Sprite-Editor.



Mit dem Soundeditor von Pico-8 komponieren Sie kleine Soundeffekte.

```
SPR([…], X, Y)
```

Tragen Sie statt der eckigen Klammern den Zahlencode ihres Sprites ein. Nach diesem Prinzip fügen Sie später auch die Sprites für das Labyrinth und den Schlüssel per Zahlencode ein. Probieren Sie das Programm erneut aus – Sie sollten jetzt anstelle des Kreises Ihre Figur auf dem Bildschirm sehen.

Das Labyrinth bauen

Als Nächstes schwingen Sie die virtuelle Maurerkelle und bauen im Sprite-Editor Bausteine für das Labyrinth. Dafür brauchen Sie zwei neue Sprites: Eines dient als Baustein für Mauer oder Hecke, das andere wird das Zielobjekt, mit dessen Erreichen das Spiel endet. Das kann beispielsweise ein Schatz oder eine Tür sein – in unserem Beispiel haben wir uns für einen Schlüssel entschieden. Nachdem Sie die beiden Sprites zusammengeklickt haben, wechseln Sie durch Klick auf das mittlere Menüleisten-Symbol – das mit den verschieden großen Kästchen – in den Map-Editor.

In der unteren Hälfte des Map-Editors sehen Sie Ihre Sprites. Die obere Hälfte ist bisher noch leer. Dazwischen befindet sich ein Streifen mit Werkzeugen: Mit dem Stift zeichnen Sie ein ausgewähltes Sprite auf die Karte, mit dem Hand-Werkzeug verschieben Sie die Karte. Über die beiden Symbole in der linken oberen Ecke blenden Sie den Werkzeugkasten ein oder aus.

Wählen Sie Ihr Mauer-Sprite aus und zeichnen Sie damit Ihr Labyrinth in den Editor. Wichtig: Das Bauwerk darf eine Größe von 16 × 16 Sprites nicht überschreiten. Schließen Sie die Ränder mit durchgehenden Mauern, damit die Figur später nicht aus dem Spielfeld herauslaufen kann. Fügen Sie noch Ihr Zielsprite ein, also den Schatz oder Schlüssel. Das Sprite für die Spielfigur fügen Sie jetzt noch nicht ein. Wechseln Sie anschließend zurück zum Code-Editor und bauen Sie Ihr Labyrinth in den Code ein. Dafür ergänzen Sie in der `_DRAW`-Funktion eine Zeile unter `SPR` mit folgendem Inhalt: `MAP(0,0,0,0,16,16)`. Die Parameter in den Klammern bestimmen Position und Größe des Labyrinths. Löschen Sie außerdem die beiden darunter stehenden `PRINT`-Zeilen. Nun sollten Sie die Figur mit den Pfeiltasten durch das Labyrinth steuern können. Noch kann Ihre Figur durch Wände laufen – darum kümmern wir uns später. Speichern Sie zunächst Ihre Arbeit, indem Sie ins Terminal wechseln und `SAVE LABYRINTH` eintippen.

Soundeffekte basteln

Kein Computerspiel ohne Soundeffekte. Öffnen Sie den Soundeditor über das dreieckige Lautsprechersymbol in der Menüleiste. Mit Mausklicks in die zentrale schwarze Fläche setzen Sie Töne. Je höher Sie die Töne im Editor positionieren, desto höher klingen diese. In der Volume-Leiste am unteren Bildrand stellen Sie die Lautstärken für die einzelnen Töne auf gleiche Art und Weise ein: Je höher der Balken, desto lauter der Ton. Die Klangfarbe ändern Sie mit den Wellensymbolen am Bildrand rechts oben; hinter jeder Wellenform versteckt sich ein anderer Sound.

Über die beiden pinken Pfeile in der oberen grauen Leiste wechseln Sie zwischen verschiedenen Tonspuren. Die mit SPD („speed“) beschriftete Zahl direkt daneben zeigt die Schnelligkeit der Tonspur an. Ein Linksklick auf die Zahl erhöht sie um 1, ein Rechtsklick verringert sie um denselben Wert. Erstellen Sie nun zwei Tonspuren. Eine, die später zu Beginn des Spiels ertönen soll, und eine, die bei jedem Schritt Ihres Charakters erklingt. Hören Sie sich die Töne zwischendurch an, indem Sie die Leertaste drücken. Wenn Sie zufrieden sind, notieren Sie die Zahlenwerte Ihrer beiden Sounds, die zwischen den beiden pinken Pfeiltasten stehen. Die Sounds bauen Sie später anhand dieser Zahlenwerte in den Code ein.

Echte Wände

Jetzt werden die Wände abgedichtet. Dazu fehlt noch die Abfrage, ob der Charakter gegen eine Mauer läuft oder den Schlüssel erreicht hat. Beides erledigen Sie mit Bordmitteln von Pico-8. Zunächst ist das Erreichen des Ziels dran, im Beispiel des Schlüssels. Folgende Frage soll dafür in Code übersetzt werden: „Ist an dieser Stelle das Ziel?“ `MGET()` erleichtert dies: Die Funktion erhält die Position der

Figur und überprüft damit, ob diese gerade auf dem Zielobjekt steht. Der Zahlencode des Ziels ist bekannt, sodass man ihn mit dem Ergebnis, auf das `MGET()` kommt, vergleichen kann: Wenn `MGET(X,Y)` gleich dem Zahlencode des Ziels ist, hat die Figur ihr Ziel erreicht. Das Spiel soll damit enden, der Spieler zurück in die Konsole von Pico-8 geleitet werden.

Aufgrund des internen Aufbaus von Pico-8 müssen die X- und Y-Werte, die `MGET()` bekommt, jeweils durch acht geteilt werden, da ein Sprite immer acht Pixel groß ist. Damit der Code lesbar bleibt, erstellen Sie zwei neue Funktionen namens `ZIEL` und `WAND`. Ersetzen Sie `[ZIEL]` und `[WAND]` im Code durch die Zahlen der entsprechenden Sprites.

```
FUNCTION ZIEL(X,Y)
   IF MGET(X/8,Y/8) == [ZIEL] THEN
      STOP()
   END
END
FUNCTION WAND(X,Y)
   RETURN MGET(X/8,Y/8) == [WAND]
END
```

Wand gibt „wahr“ zurück, wenn sich ein Hindernis im Weg befindet. Ist der Weg frei, gibt die Funktion „falsch“ aus. Tippen Sie die beiden neuen Funktionen ans Ende des Codes. Damit die Figur nicht mehr durch Wände laufen kann, passen Sie die `UPDATE`-Funktion weiter an. Denn bisher prüfte der Code für die Bewegung ausschließlich, ob die Pfeiltaste gedrückt wurde. Ob eine Wand im Weg steht, wurde nicht abgefragt. Das ändern Sie, indem Sie die Bedingung „Wenn Pfeiltaste nach links gedrückt und keine Wand im Weg, gehe nach links“ in Code übersetzen: `IF BTNP(0) AND NOT WAND(X-8,Y) THEN`. Außerdem soll bei jedem Schritt des Charakters Ihr Schritt-Sound erklingen. Das erledigt `SFX()`. Passen Sie die `_UPDATE`-Funktion folgendermaßen an und setzen Sie die entsprechenden Zahlenwerte anstelle von `[Bewegungssound]` ein:

```
FUNCTION _UPDATE()
   IF BTNP(0) AND NOT WAND(X-8,Y) THEN
      X=X-8
      SFX([Bewegungssound])
   END
   IF BTNP(1) AND NOT WAND(X+8,Y) THEN
      X=X+8
      SFX([Bewegungssound])
   END
   IF BTNP(2) AND NOT WAND(X,Y-8) THEN
      Y=Y-8
      SFX([Bewegungssound])
   END
   IF BTNP(3) AND NOT WAND(X,Y+8) THEN
      Y=Y+8
      SFX([Bewegungssound])
   END
   ZIEL(X,Y)
END
```

Mit Pico-8 kann man zig Spiele der Community herunterladen und spielen.

Im letzten Schritt fügen Sie an allererster Stelle Ihres Codes die Zeilen an, in denen X- und Y-Werte zugewiesen werden, sodass beide nun den Wert 8 bekommen. Fügen Sie darüber einen Befehl zum Abspielen des Anfangssounds ein:

```
SFX([Anfangssound])
X=8
Y=8
```

Zeit für eine Pause

Das war es schon – mit rund 40 Zeilen Code haben Sie ein lauffähiges Pixel’n’Sprites-Spiel programmiert, das voll auf der Retro-Welle surft. Das Ergebnis ist natürlich noch ausbaufähig: So könnte man etwa ein schöneres Ende mit einem weiteren Soundeffekt oder einer kleinen Animation einbauen und fiese Hindernisse implementieren.

Inspiration für die Weiterentwicklung können Sie sich bei den zahlreichen Community-Projekten holen, die der Befehl `SPLORE` in die Kommandozeile auflistet. Besonders empfehlenswert ist die Kategorie „FEATURED“, zu der Sie mit der linken und rechten Pfeiltaste navigieren können. Zunächst müssen Sie einmal mit `UPDATE` die Spiele herunterladen, um danach mit den Pfeiltasten eines auszuwählen und mit Enter zu starten.

Darüber hinaus bietet das Pico-8-Wiki eine gute Dokumentation und im Pico-8-Forum finden Sie eine aktive Gemeinschaft, die Ihnen bei weiteren Programmierexperimenten helfen kann.

(mre@ct.de) c't

Pico-8, Foren, Lua, Beispielcode:
ct.de/ymr3

Die wichtigsten Terminal-Befehle

Befehl	Funktion
LS	zeigt aktuellen Ordnerinhalt an
CD […]	wechselt in anderen Ordner. Der Ordnername muss angegeben werden, alternativ wechselt man mit „..“ in den übergeordneten Ordner.
LOAD […]	lädt die Pico-8-Dateien, sodass man sie bearbeiten und ausführen kann
RUN	führt eine geladene Pico-8-Datei aus
SAVE […]	speichert die aktuelle Pico-8-Datei unter dem angegebenen Dateinamen
INSTALL_DEMOS	installiert vorgefertigte Demo-Cartridges
CLS	löscht alles, was auf dem Bildschirm zu sehen ist
REBOOT	startet Pico-8 neu
SHUTDOWN	beendet Pico-8