Manualul PIX37

Editor, apoi cod. Copie offline a cărții pe pix37.party/docs.

Editor

PIX37 DEV este editorul. Aici desenezi, scrii și asculți jocul, apoi exporți un cartuș de joc sau o versiune independentă. Poate reda și un cartuș de joc și poate deschide PIX37.PARTY pentru a descărca cartușe de joc și a utiliza catalogul. PIX37 DEV EDUCATIONAL este același editor și este gratuit. Nu are versiuni, nu exportă cartușe de joc și nu are acces la PIX37.PARTY. Consola PIX37 este playerul. Redă cartușe de joc `.p37` și deschizi PIX37.PARTY pentru a descărca cartușe de joc și a răsfoi catalogul.

Proiecte și cartușe de joc

Un fișier `.pix37` reprezintă proiectul: tab-uri sursă, foi de sprite, hărți, tabele de sunet și date. Un fișier `.p37` reprezintă cartușul de joc pe care îl rulează jucătorul: bytecode compactat.

Salvările și realizările se află în folderul autorului, lângă datele aplicației. Pe Windows, acesta arată ca `C:\\Users\\Ada\\AppData\\Roaming\\PIX37\\Mira\\Pong.p37s` pentru o salvare și `C:\\Users\\Ada\\AppData\\Roaming\\PIX37\\Mira\\Pong.ach` pentru realizări. Pe Linux, aceleași fișiere sunt `~/.local/share/PIX37/Mira/Pong.p37s` și `~/.local/share/PIX37/Mira/Pong.ach`. Ada este contul de pe computer. Mira este numele autorului, iar acest nume este folderul. Pong este slotul de salvare, care implicit este titlul jocului. `save` scrie fișierul `.p37s`. Prima realizare deblocată scrie fișierul `.ach`. Dacă fișierele sunt deja acolo, alergătorul le citește când pornește cartușul jocului.

Meniu Dezvoltatori

Tasta Escape deschide meniul Dev. În timp ce un joc rulează, tasta Escape revine la editor. Confirmați cu Enter sau cu butonul A (implicit V). Tasta Back (Înapoi) este butonul B (implicit C).

Fără niciun proiect încărcat, meniul este: CREARE PROIECT, ÎNCĂRCARE PROIECT, CONSTRUIRE, ÎNCĂRCARE cartuș de joc, CONECTARE LA PIX37.PARTY, SETĂRI.

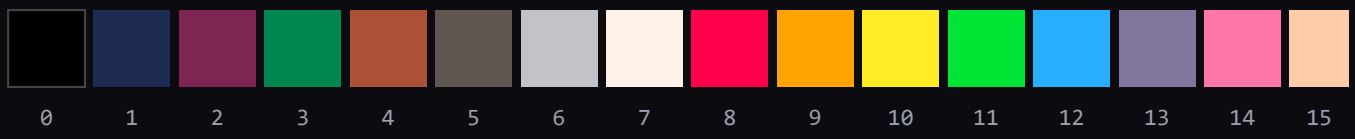
Cu un proiect încărcat, urmează: SALVEAZĂ PROIECTUL, ÎNCĂRCĂ PROIECTUL, CREARE PROIECT, RESPONSABILE, CONSTRUIE, ÎNCĂRCĂ cartuș de joc, CONECTEAZĂ-TE LA PIX37.PARTY, SETĂRI.

Ctrl+S salvează proiectul deschis. Ctrl+R îl rulează.

Creați un proiect

CREATE PROJECT solicită cartușul de joc înainte de a începe crearea proiectului.

- **VIDEO** — 1 BIT (2 culori), 2 BIT (4), 4 BIT (16), 8 BIT (256).
- **AUDIO** — adâncimea eșantionului sintetizatorului: 16, 8, 4, 2 sau 1 bit.
- **ECRAN** — presetările includ 64×64, 128×128, 160×90, 256×256 și 320×180. Dimensiunea PERSONALIZATĂ vă permite să alegeți orice dimensiune de la 1×1 la 512×512.
- **PALETE** — bănci de culori. Prima bancă începe cu setul de 16 culori de mai jos. LOAD PALETTE aplică un set încorporat: PICO-8 16, PICO-8 32, Game Boy, 1-bit, CGA, Sweetie-16 și o listă mai lungă preluată de la Lospec (Resurrect 64, Apollo, Endesga 32 și 64, CC-29, SLSO8, Pear36, Oil 6, Duel, AAP-64, Lost Century și altele). De asemenea, puteți încărca un fișier de paletă sau puteți încărca propriul fișier PNG și puteți extrage paleta din acea imagine.



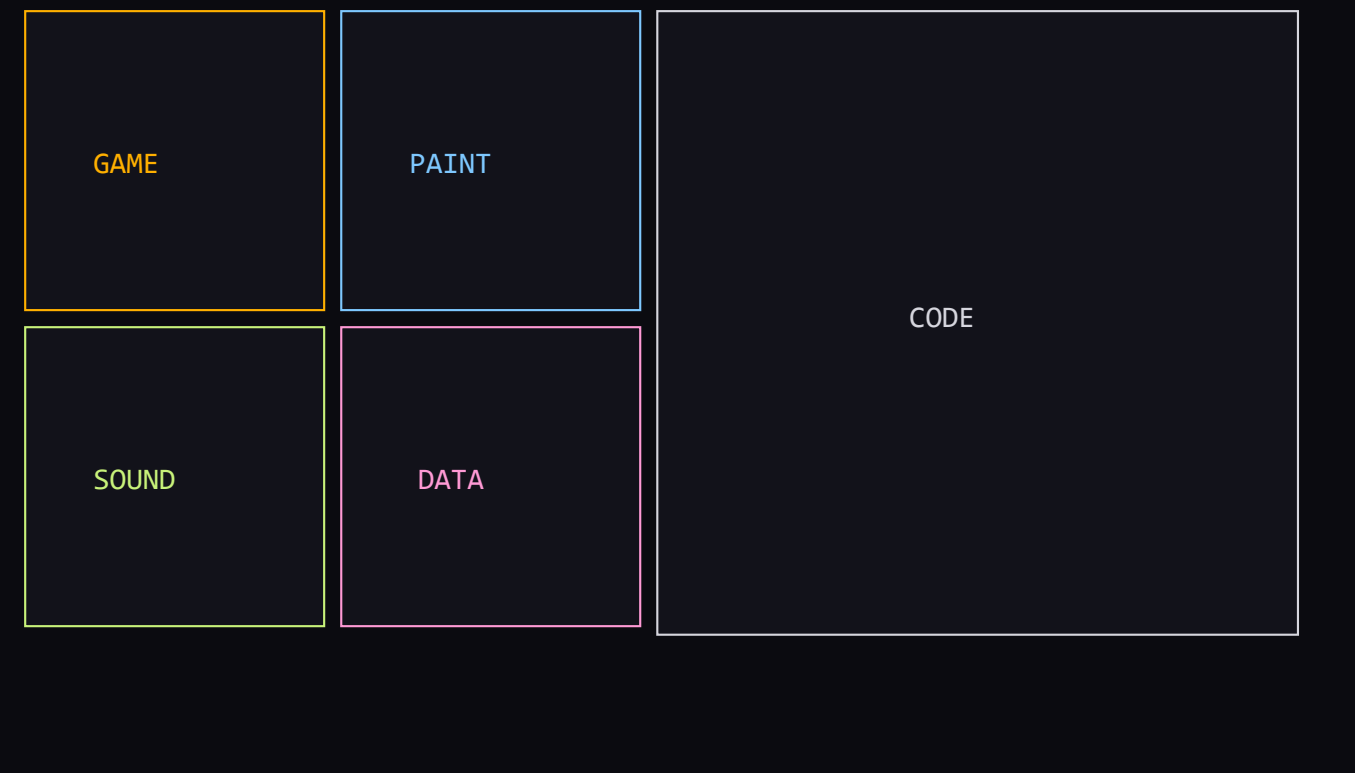
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

```
lette — black, navy, purple, green, brown, gray, silver, white, red, orange, yellow, lime, blue, indigo, p
```

Indexuri de culoare utilizate de `cls`, `pset`, `spr` și restul. Puteți repicta orice index în editorul de palete.

Cele cinci panouri

Editorul este GAME, PAINT, CODE, SOUND și DATA, în această ordine. Harta se află în PAINT. SFX-urile și melodiile se află în SOUND. Faceți clic pe un panou pentru a lucra acolo. Când CODE are focalizarea, SOUND și DATA ignoră tastele cu litere, astfel încât tastarea nu se face și cu pianul.



> Aspect complet, scris `1:1:4:1:1`. Fiecare număr reprezintă suprafața de rând ocupată de panoul respectiv: GAME, PAINT, CODE, SOUND, DATE. Un ecran mic se află centrat într-o celulă de 128×128. Un ecran mai mare de 128 px lărgește coloana GAME.

SETĂRI → GLOBAL → ARANJAMENT parcurge aranjamentele:

Cod	Ceea ce vezi
`1:1:4:1:1`	Totul. JOC și SUNET la un loc, VOPSEA și DATE la un loc, COD în dreapta.
`1:1:4:1:1 2X`	Aceeași aranjament. CODUL este desenat la o densitate dublă, deci încape de două ori mai mult text.
`1:0:1:0:1`	JOC, DATE, COD.
`1:0:1:0:1 2X`	Același aranjament, cu CODUL mai dens.
`0:1:1:1:1`	Fără previzualizare a jocului. VIZTEAZĂ, COD, SUNET, DATE.
`0:1:1:1:1 2X`	Același aranjament, cu CODUL mai dens.
`0:1:4:1:2`	Studio. COD lat, o bandă DATA mai înaltă, PAINT și SOUND.
`0:1:4:1:2 2X`	Același aranjament, cu CODUL mai dens.
`0:0:4:0:4`	Doar COD și DATE, ambele mari.
`0:0:4:0:4 2X`	Același aranjament, cu CODUL mai dens.
`0:1:4:1:0`	VOPSEA, SUNET și un COD amplu.
`0:1:4:1:0 2X`	Același aranjament, cu CODUL mai dens.
`0:4:0:0:0`	Doar VOPSEA, 256×256, un pixel din foaie la un pixel din pânză.

Un rând `2X` păstrează aceleași panouri. Panoul CODE este desenat la o rezoluție dublă și scalat înapoi în aceeași celulă, astfel încât textul, tabulatorii și marginea interioară au jumătate din înălțimea și sursa încape de două ori mai mult pe ecran.

JOC

Previzualizarea cartușului de joc care rulează. Faceți clic pe panoul JOC sau apăsați Ctrl+R pentru a rula previzualizarea respectivă. SETĂRI → PANEL JOC este locul unde este configurat panoul:

- **Previzualizare** — culoare inactivă, continuă sau reîncărcare automată la editare.
- **Joacă într-o fereastră nouă** — jocul se deschide lângă editor.
- **Afișare** — previzualizare ca RGB888 sau RGB565.
- **GAME 64** — un ecran de 64×64 este fie centrat 1:1 în celulă, fie scalat de 2× pentru a o umple.
- **Shadere** — CRT și filtre în joc, separate de filtrele din editor.
- **Culoare** — fundalul panoului.

VOPSEA

Foi de sprite și, pe partea cu harta a aceluiași panou, hărți tip dală. Foile pot fi mai mari decât ecranul. Dimensiunile de previzualizare a sprite-urilor sunt 8, 16, 32, 64 și 128 (128 doar când foaia are cel puțin 128 px). Derulați rotița mouse-ului peste previzualizarea sprite-ului pentru a o mări. Dimensiunile pensulei sunt 1, 2, 3 și 4.

Cheie	Instrument
B.	Creion
M.	Selectare dreptunghiulară
Î	Lasso
S.	Ştampila
G.	Găleată
Eu	Pipetă
V.	Mişcare
L / Shift+L	Linie / curbă
U / Shift+U	Dreptunghi gol / elipsă goală
F / Shift+F	Dreptunghi umplut / elipsă umplută
D.	Tremurând
[ ]	Pensulă mai mică / mai mare
Ctrl+Z / Ctrl+Y	Anulare / Refacere. Ctrl+Shift+Z reface și ea acțiunea.
Ctrl+C / Ctrl+X / Ctrl+V	Copiază, taie, lipește
Ctrl+A	Selectați pânza

Harta folosește aceeași idee, cu mai puține instrumente: B creion, M selecție dreptunghiulară, S ștampilă, V mișcare și Ctrl+Z, Ctrl+Y, Ctrl+C, Ctrl+V. Țineți apăsată tasta Alt pentru a mări textura hărții. Dimensiunea pensulei pe hartă rămâne la instrumentele de pictură.

Fiecare sprite are opt biți de steag. Paint îi setează pe sprite; `fget` și `fset` îi citesc în joc.

SETĂRI → EDITOR PAINT este locul unde este configurat panoul respectiv:

- **Paletă** — banca care este desenată, plus încărcare și salvare.
- **Previzualizare** — RGB888 sau RGB565.
- **Cursor de pensulă** — indicatorul în timp ce pictați.
- **Ajustare pixeli** — modul în care pensula se aliniază la pixeli.
- **Pensulă mare** — o pensulă mare se fixează din colțul stânga sus sau din întreaga sa amprentă.
- **Coajă de ceapă** — o fantomă a altor cadre și direcția în care este orientată acea fantomă.
- **Zoom al hărții** — pixeli reali sau zoom liber.
- **Culori** — culorile interfeței utilizator pentru vopsea.

COD

Filele sunt fișiere separate în cadrul proiectului. Faceți clic pe numele unei file pentru a o redenumi; Enter confirmă. Mai multe file care definesc fiecare `_init`, `_update` sau `_draw` sunt lipite împreună în ordinea tabulatorilor: prima filă rulează prima, ultima filă rulează ultima. Codul care nu se află în interiorul unui hook devine parte a `_init`. Un cartuș de joc fără hook-uri este tratat ca un singur `_draw`.

Cheie	Acțiune
Săgeți	Mută semnul circumflex. Shift extinde selecția.
Ctrl+Stânga / Dreapta	Prin cuvânt.
Acasă / Sfârșit, Ctrl+Acasă / Sfârșit	Începutul sau sfârșitul liniei sau al tabulatorului.
Tab	Introduceți două spații. În lista de completare, acceptați numele.
Introduceți	Linie nouă, indentată ca și cea anterioară. Acceptă o completare.
Ctrl+C / X / V	Copiază, taie, lipește.
Ctrl+Z / Ctrl+Y	Anulare / Refacere.
Ctrl+A	Selectați fila.
Ctrl+D	Duplicați linia sau selecția în jos.
Ctrl+/	Comentează sau anulează comentariile cu `--`.

Ctrl+F	Găsește. F3 sare la următoarea lovitură. Escape închide bara.
Ctrl+G	Mergeți la un număr de linie.
Evadare	Închide mai întâi meniul de căutare, meniul „mergi la” sau lista de sugestii. O a doua tastă Escape părăsește editorul și deschide meniul pentru dezvoltatori.

Apar indicii în timp ce tastați numele unei funcții. Enter sau Tab introduce numele și un `(` dacă COMPLETE (este activat în setări. SETĂRI → EDITOR DE COD:

- **Căutare** — Ctrl+F în fila activă sau în fiecare filă.
- **Text sus** — decupează glifele la 5 px sau afișează glifa completă.
- **Dimensiunea șirului de caractere** — dimensiunea fontului literalilor de șir.
- **Font pentru comentarii** — 3×5 sau fontul Fusion al limbii curente a interfeței utilizator.
- **Complet (** — dacă completarea tastează paranteza.
- **Indicii de cod** — activate sau dezactivate.
- **Limba sugestiilor** — limba meniului sau engleza.
- **Spațiu caret** — decalaj suplimentar pe liniile goale.
- **Jgheab** — numere de linie.
- **Culoare** — fundal, comentarii, tabulatori, numere, operatori, cuvinte cheie, șiruri de caractere, funcții ale motorului, funcțiile dvs., variabile, paranteze, semn circumflex, selecție, linie de eroare, avertisment.

SUNET

Panoul are două file. SFX este un model: o serie scurtă de note. SOUND este o melodie: plasează acele modele pe opt canale. TIC reprezintă numărul de tic-uri pe care le așteaptă un pas. Ceasul este de 120 de tic-uri pe secundă, iar un pas este o notă de șaisprezece secunde, deci 15 tic-uri reprezintă 120 de bătăi pe minut. Un TIC mai mare este un model mai lent. Două modele rămân în pas atunci când un TIC îl împarte pe celălalt: 15 și 30 o fac, 15 și 8 nu.

Pianul se aude când o notă este selectată cu mouse-ul. Clapele albe sunt ZXCVBNM (CDEFGAB). Clapele negre sunt SDGHJ (C# D# F# G# A#). Plasați cursorul pe orice valoare și tastați un număr sau derulați cu rotița. Numerele de înălțime din cod sunt de la 0 (C-0) la 63 (D#5).

Pe un model, LOOP și LEN sunt aceleași două numere. OUT peste 0 se repetă de la IN la OUT, iar titlul este LOOP. OUT la 0 și IN peste 0 este LEN: atâtea note se redau de la început, apoi modelul se termină, iar titlul este LEN. Ambele la 0 sunt redate la fiecare pas.

Cât de mult din sintetizator poți edita depinde de adâncimea de biți aleasă pentru cartușul jocului. SETĂRI → EDITOR SFX setează acea adâncime, dacă o notă avansează automat (dezactivat, lanț de note sau lanț de câmpuri), dacă Space repornește redarea sau comută între pornire și oprire și culorile editorului.

16 biți

Fiecare pas are o notă, un volum, o undă și un efect. Volumul este de la 0 la 7. 0 este silențios.

Valuri:

- **0 triunghi**
- **1 ferăstrău înclinat**
- **2 a văzut**
- **3 pătrate**
- **4 impulsuri** — un impuls îngust
- **5 orgă**
- **6 zgomote**
- **7 fazere**
- **8 PWM** — un pătrat a cărui lățime se modifică
- **9 impulsuri 25%**
- **10 sinusuri**
- **11 zgomot metalic**
- **12 FM clopot**
- **13 smulge**
- **14 superfierăstrău**
- **15 PWM (C64)**

Efecte pe o singură notă:

- **0 niciunul**
- **1 glisare** — înălțimea și volumul glisează la următoarea notă în acest pas
- **2 vibrato** — oscilații ale tonalității
- **3 drop** — sunetul se amuțește, lovitura obișnuită
- **4 apariții fade**
- **5 se estompează**
- **6 arpegii** — parcurge această notă și următoarele trei, una la fiecare 8 tic-uri. La TIC 8 sau mai mic, la fiecare 4 tic-uri.
- **7 arpegii rapide** — aceleași patru note, la fiecare 4 tichi sau la fiecare 2 când TIC este 8 sau mai mic.

Antetul modelului, deasupra notelor:

- **TIC** — tichi pe pas, de la 1 la 255. 15 înseamnă 120 de bătăi pe minut. Introduceți numărul sau treceți-l treptat la o unitate.
- **LOOP IN / OUT** — cu OUT mai mare decât 0, modelul se repetă în buclă de la IN la OUT. Lăsați OUT la 0 și titlul devine LEN: IN reprezintă numărul de note redate, iar 0 reprezintă întregul model.
- **ZGOMOT** — întregul model devine zgomot.
- **CRUSH** — o variantă vibrantă a fiecărui val.
- **DTUNE** — 0 oprit, 1 un chorus ușor, 2 un chorus puternic.
- **RVERB** — reverb: 0 dezactivat, 1 scurt, 2 lungi.

- **DAMPN** — un filtru trece-jos: 0 oprit, 1 ușor, 2 puternic.
- **LEVEL** — volumul întregului pattern, de la 0 la 7.
- **LEAGĂN** — amestecare. Pașii impari așteaptă puțin mai mult, pașii pare puțin mai puțin.
- **PAȘI** — câți pași folosește modelul: 32, 24, 16 sau 12.

8 biți

Un pas are o notă și o undă. Cele patru unde sunt pătrată, triunghi, saw și zgomot. Antetul are TIC, LOOP IN / OUT, FX și LEVEL. TIC este unul dintre 1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 30, 40, 60. FX este un efect pentru întregul pattern: 0 niciunul, 1 vibrato, 2 arpegiu, 3 fade out. LEVEL este de la 0 la 3.

4 biți

Un pas este doar o notă. Antetul are TIC, LOOP (activat sau dezactivat), WAVE (aceleași patru unde) și OCT (octava în care se află notele). TIC este una dintre cele 4, 6, 8, 10, 12, 15, 20, 30.

2 biți

Un pas este gol, o tobă, o lovitură sau o lovitură puternică. Antetul are TIC și BASE. TIC este 8, 15, 16 sau 30. BASE setează la ce înălțime se află toba și lovitura.

1 bit

Un pas este o notă. TIC este 8, 15, 16 sau 30, aceleași patru viteze ca și pe 2 biți.

Cântece

Fila SUNET este melodia deschisă. Opt coloane reprezintă canalele. O celulă reprezintă un ID de model. 255, desenat gol, înseamnă că acel canal este silențios pe rândul respectiv. Coloana X este o comandă: continuare, oprire, început buclă sau sfârșit buclă. Oprirea încheie melodia după rândul respectiv. Sfârșitul buclei revine la ultimul început de buclă.

Un rând se termină atunci când modelul din stânga cu LOOP OUT la 0. Modelele mai la dreapta nu lungesc rândul respectiv: unul mai lung este tăiat, unul mai scurt începe din nou de la primul său pas până când rândul se termină. Un model cu LOOP OUT peste 0 se repetă între LOOP IN și LOOP OUT și nu termină rândul. Dacă fiecare model de pe rând se repetă, rândul rămâne acolo.

Scurtături

Cheie	Acţiune
Clape de pian	Scrieți o notă despre pasul selectat cu mouse-ul.
Sus / Jos	Schimbați valoarea de sub cursor.

Spaţiu	Redă efectele sonore deschise sau melodia deschisă de pe primul rând. Pornește/oprește sau repornește din setările EDITORULUI DE EFECTE SONOR.
Ctrl+Z / Ctrl+Y	Anulare / Refacere.
Ctrl+C / Ctrl+X / Ctrl+V	Copiați, tăiați, lipiți selecția.

DATE

Tabele denumite. Fiecare tabel este format din rânduri și coloane. Din cod, `dget`, `dset`, `dstr`, `drows` și `dcols` utilizează aceste tabele. Numerele de rânduri și coloane din cod încep de la 0.

Tipuri de coloane, afișate ciclic în antet:

- **bool** — un bit.
- **2** — doi biți.
- **4** — patru biți.
- **8** — un octet.
- **16** — 16 biți, pentru ID-uri și numere mai mari.
- **spr** — un ID de sprite.
- **str** — text, desenat cu fontul sistemului.
- **str8** și **str10** — același text, desenat mai înalt.
- **f16** și **f32** — o mișcare în plutire cu jumătate de precizie și o mișcare în plutire completă.

Cu indicatorul plasat pe antetul unei coloane, Ctrl+derulare schimbă tipul coloanei respective. O derulare simplă se deplasează pe coloane sau pe rânduri, în funcție de opțiunea selectată de SETTINGS → DATE EDITOR. Ctrl schimbă direcția respectivă, motiv pentru care Ctrl+derulare pe un antet schimbă tipul atunci când rotița este setată pe coloane.

Tastele săgeată deplasează. Enter editează o celulă. SETĂRI → EDITOR DATE:

- **Roată** — derulează coloane sau derulează rânduri.
- **Culori** — grila, antetul și selecția.

Active

ASSETS importă și exportă piesa deschisă, nu întregul proiect. Sprite-urile folosesc sprite-ul deschis în PAINT, pe foaia deschisă. Harta este harta deschisă. SFX este modelul deschis. Muzica este melodia deschisă. Data este tabelul deschis. Codul este excepția: fiecare tab este scris și fiecare tab este citit înapoi.

Fel	Fişier	Ce se mișcă

Sprite-uri	`.png`	Sprite-ul deschis.
Hartă	`.p37map`	Harta deschisă.
Efecte sonore	`.p37sfx`	Modelul deschis.
Muzică	`.p37mus`	Melodia deschisă. Numerele de pattern rămân așa cum sunt. Exportați efectele sonore separat dacă aveți nevoie de acele sunete.
Date	`.p37dat`	Masa deschisă.
Cod	`.lua`	Fiecare filă.

Setări

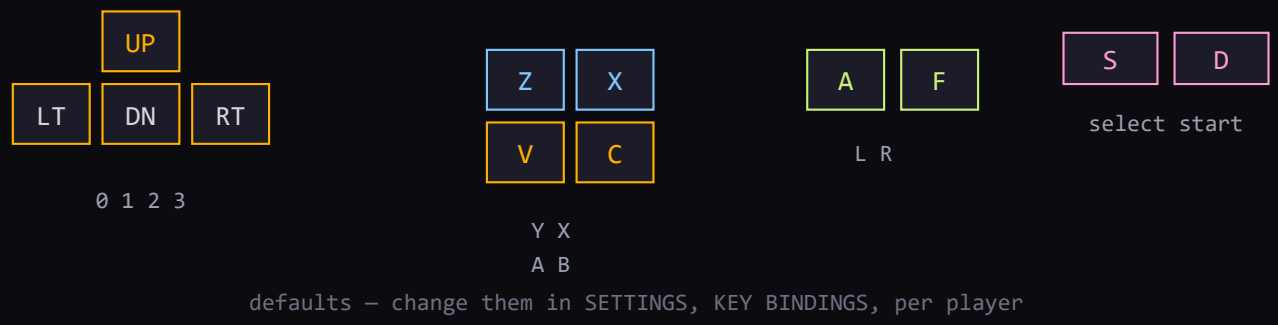
Lista rădăcinilor:

Pagină	Ce stochează
PROFILE	Salvați și încărcați fiecare setare. Creați profil nou, redenumiți, suprascrieți, ștergeți. Copiați sau lipiți profilul ca text.
Meniu	Fereastră, taste, culori meniu, fundal opțional.
PANEL DE JOC	Previzualizare și fereastra de redare.
EDITOR DE COD	Evidențiere și sugestii.
EDITOR DE VOCARE	Pensulă, paletă, vizualizare hartă.
EDITOR SFX	Sintetizator și introducere de note.
EDITOR DE DATE	Direcția și culorile roților.
LIMBĂ	Limbajul meniului și al indiciilor.
LICENȚE	Texte de licență, inclusiv fontul Fusion Pixel și SDL3.

GLOBAL:

- **ARANJARE** — aranjamentul panourilor, mai sus.
- **VOLUME** — nivel principal, de la 0 la 10.
- **LEGĂRI DE TASTE** — taste pentru fiecare dintre cei 8 jucători. Apăsați o tastă pentru a o captura. Instrucțiuni, ABXYLR, selectați.
- **Prezentare** — omite cadrele inactive sau desenează fiecare cadru.

- **Ecran complet** — fără margini.
- **Reîmprospătare** — FPS fereastră. Trepte stânga și dreapta 60, 90, 120 și max. Introduceți un număr de până la 1000. 0 înseamnă fără limită.
- **Vsync** — așteaptă afișarea între cadre.
- **Scală** — multiplicator de fereastră cu numere întregi.
- **Shadere** — CRT și filtre în editor, nu în joc.
- **Culoare** — textul meniului și evidențierea acestuia.
- **Fundal** — un fișier PNG opțional, decupat la 128, în spatele meniului. Ștergere îl elimină.
- **Cursorul editorului** — forma indicatorului.
- **Mouse** — indicator plutitor sau fixat pe pixeli.
- **Viteza interfeței utilizator** — cât de repede se repetă o direcție menținută apăsată într-o listă.
- **Explorer** — listele de fișiere încep de la rădăcina computerului sau din folderul fișierului exe.



Tastatură implicită. A este pentru confirmarea în meniuri. B este pentru înapoi. Într-un joc, acestea sunt butoanele 4 și 5. Selectare (S) este butonul 10, nu o comandă rapidă din meniu.

Buton	Rol	Cheie implicită
0	Stânga	Săgeată stânga
1	Corect	Săgeată dreapta
2	Sus	Săgeată în sus
3	Jos	Săgeată în jos
4	A, confirmați	V.
5	B, spate	C.
6	X	X
7	Y	Z
8	R	F
9	L.	O

10	Selecta	S.
11	Început	D.

Construi

Alege o platformă: Windows, Linux, Mac, Android, iOS sau Web.

- **Nume** — numele fișierului compilat.
- **Autor** — pe Windows și Linux. Acesta este numele folderului pentru salvări.
- **ID-ul aplicației Steam** — pe Windows și Linux. Gol înseamnă că nu există Steamworks. Un ID completat mapează `achieve(0)`…`achieve(15)` la realizările Steam ACH_0…ACH_15.
- **Id** — identificatorul pachetului acolo unde platforma solicită unul.
- **Ascunde codul** — pe suprapunerile de pe desktop și mobil. Încetinește `peek` al cartușului de joc. Nu este un blocaj.
- **Ecran complet** — playerul exportat pornește pe ecran complet.
- **Shadere** — până la trei straturi integrate în player.
- **Descărcare .p37** — Numai pentru web. Afișează o intrare de descărcare în meniul de pauză.
- **Pictogramă** — o dală dintr-o foaie de sprite: foaie, dimensiune, x, y. Sau o copertă PNG pe care o decupezi.
- **Export** — alegeți folderul. Meniul afișează CONSTRUCȚIE SALVATĂ sau CONSTRUCȚIE EȘUATĂ.

Creați o versiune cu internetul activat. Splash-ul poate întreba serverul, în mod anonim, dacă hash-ul acestei versiuni este cunoscut. Această verificare există din două motive. Mă protejează pe mine, autorul PIX37, de copii ale instrumentului care nu erau ale mele pentru a le oferi. De asemenea, protejează persoana care a creat jocul. Luarea jocului altcuiva și listarea lui ca fiind al tău devine o practică obișnuită pe piața telefoanelor, cum ar fi Play (Android) sau iOS. Jocul pornește totuși. Dacă mașina este offline, introducerea spune că jocul nu este conectat. Această linie este, de asemenea, un indiciu pentru jucător că este posibil ca copia să nu provină dintr-o sursă legală. Este o mică verificare, astfel încât, dacă jocul tău a fost furat și vândut fără consimțământul tău de către altcineva, poți face acest lucru într-un magazin precum Play (Android), Steam sau itch.io. Serverul stochează momentul în care hash-ul a fost înregistrat pentru prima dată, lângă hash, și nu îl modifică ulterior. Momentul anterior este versiunea originală. Un hash înregistrat după aceea este o copie ulterioară.

Vreau ca această verificare să protejeze oamenii care creează jocuri. Motorul de joc face deja o mare parte din muncă pentru ei, iar acesta este unul dintre aceste lucruri. Înțeleg că poate fi controversat. Dacă comunitatea decide că este mai bine să renunțe la acest micro-DRM, o voi face.

Un cartuș de joc vechi de la un editor anterior poate refuza să pornească după schimbarea limbii. Deschideți proiectul `.pix37` în acest editor și exportați un nou cartuș `.p37`.

Conectați-vă la PIX37.PARTY

CONECTAREA LA PIX37.PARTY se face în aplicație. Nu există o pagină de conectare în browser. Te conectezi cu un tichet de sesiune Steam sau cu itch.io, folosind tableta și mouse-ul. Serviciul afișează un nume de utilizator unic, format din 2 până la 16 caractere. Cheia de licență și ID-ul Steam nu sunt afișate altor jucători.

După termenii serviciului, centrul este Favorite, Catalog, Clasament, Jucători și profilul tău.

O licență de dezvoltator este editorul. Tu creezi jocul, îl încarci și lași stele. O licență de jucător este jucătorul. Inimioarele sunt de la oamenii care joacă pe un alergător.

- **Catalog** — răsfoiește jocuri. Filtrele acoperă genul (până la trei), dimensiunea ecranului, dimensiunea cartușului jocului și dacă încărcarea include un proiect `.pix37`.
- **Favorite** — jocuri pe care le-ai marcat.
- **Încărcare** — necesită o licență de dezvoltator. Trimiți un fișier `.p37` și poți atașa proiectul `.pix37`. Un fișier `.p37` este un fișier bytecode compactat. Este greu de citit și poate fi ofuscat intenționat. Un fișier `.pix37` este fișierul de proiect în care lucrezi efectiv: sursa, imaginile și sunetul, într-o formă pe care o altă persoană o poate deschide și din care poate învăța. Trimiterea acestuia permite altor creatori să studieze jocul în loc de un cartuș de joc comprimat. Titlul, autorul, genurile, coperta, pictograma și numele realizărilor merg odată cu acesta. Numărul de RAM trimis reprezintă vârful atins de cartușul de joc.
- **Comentarii și realizări** — pe pagina jocului. Deblocarea unei realizări în timp ce ești conectat este stocată atât în contul respectiv, cât și pe dispozitiv.
- **Clasament** — dezvoltatori, sortați după stele sau după inimioare. Fiecare rând arată ambele numărări. Deschideți un nume pentru a vedea jocurile autorului respectiv, ordonate după inimioare.
- **Jucători** — conturi ordonate după realizările deblocate.

Cod

Limbajul este un script de mici dimensiuni, în stil Lua. Această jumătate pornește de la un proiect gol, predă limbajul și apoi prezintă un scurt exemplu pentru fiecare funcție.

Un prim program

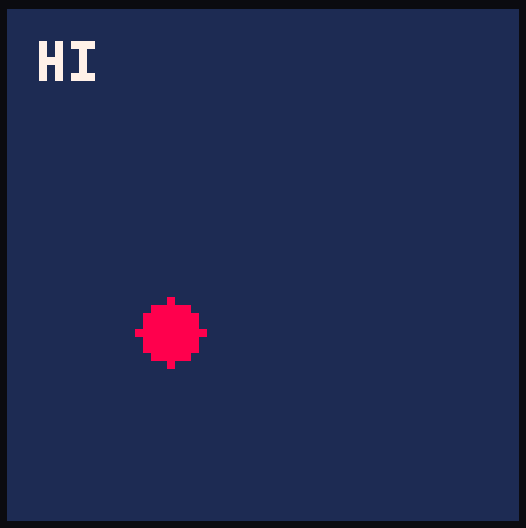
Creați un proiect la 64×64, culoare pe 4 biți. Deschideți CODE. Înlocuiți tabulatorul cu:

```
_init()
  x = 20
  y = 40
end

_update()
  if btn(0) then x -= 1 end
  if btn(1) then x += 1 end
  if btn(2) then y -= 1 end
  if btn(3) then y += 1 end
end

_draw()
  cls(1)
  print("HI", 4, 4, 7)
  circfill(x, y, 4, 8)
end
```

Ctrl+R îl rulează. Tastele săgeată mișcă cercul. Tasta Escape oprește jocul și revine la editor.



Aceleași trei apeluri: `cls(1)` pictează ecranul bleumarin, `print` desenează HI, iar `circfill` desenează cercul roz.

Ce se execută și când

- `_init` rulează o singură dată.
- `_update` funcționează la 60 Hz. Puneți mișcarea și butoanele aici.
- `_draw` pictează. Fără argument fps, pictează la viteza maximă permisă de computer și monitor. `_draw(30)` limitează pictarea la 30 fps și lasă logica la 60.
- `_set_fps(n)` modifică ulterior limita respectivă. 0 o șterge.

Numele nu sunt sensibile la majuscule/minuscule; editorul le stochează cu litere mici. `function` și `fun` sunt același cuvânt cheie.

Limbă

Noțiuni de bază despre programare

Un program este o listă de linii pe care le rulează cartușul jocului. Fiecare titlu arată mai întâi forma liniei, apoi un mic exemplu. O linie care începe cu `--` este un comentariu: o notă pentru tine. Jocul o sare peste.

Variabile

O variabilă este un nume care conține o valoare. Tu alegi numele. Liniile ulterioare îl pot citi și pot stoca o nouă valoare cu același nume.

Forma este `name = value`. `local name = value` păstrează acest nume în blocul curent, astfel încât restul cartușului de joc nu îl poate vedea. Fără `local`, numele este global și fiecare filă îl poate vedea.

```
hp = 3            -- global: the whole cart can read and change hp
local coins = 0   -- local: only this block can see coins
hp = hp + 1       -- read 3, add 1, store 4 back into hp
```

`nil` înseamnă că nu există nicio valoare. `true` și `false` sunt da și nu. Un nume pe care nu l-ați setat niciodată se citește ca `nil`.

```
bonus = nil            -- nothing stored yet
if bonus == nil then   -- condition: is bonus empty?
  bonus = 0            -- body: store a number
end
```

Tabele

Un tabel conține mai multe valori sub un singur nume. O listă folosește numere. Primul element este 1, nu 0. O înregistrare folosește nume. Un tabel poate conține un alt tabel.

Formă listă: `{ value, value }`. Formă înregistrare: `{ name = value, name = value }`. O listă de înregistrări este un tabel ale cărui elemente sunt tabele.

```
-- a list. [1] is the first item
scores = {10, 20, 30}
print(scores[1], 1, 1, 7)      -- print draws text. this draws 10

-- a record. player.x is the field named x
player = {x = 10, y = 20}
print(player.x, 1, 8, 7)       -- draws 10

-- a list of records. enemies[1] is the first record
enemies = { {x = 4, y = 8}, {x = 20, y = 12} }
print(enemies[1].x, 1, 16, 7)  -- draws 4
print(enemies[2].y, 1, 24, 7)  -- draws 12
```

Condiție și corp

O condiție este o întrebare cu un răspuns de tip da sau nu, cum ar fi `hp > 0` sau `btnd(4)`. Un corp este grupul de linii care rulează atunci când intri într-un bloc. `if` și fiecare buclă folosesc aceeași formă: o condiție, apoi un corp, apoi `end`. `repeat` se închide cu `until` în loc de `end`.

Dacă

`if` își execută corpul numai când condiția este adevărată. `elseif` încearcă o altă condiție. `else` rulează când toate condițiile erau false.

Formă: `if condition then`, apoi corpul, apoi `end`.

```
hp = 2
if hp > 0 then                 -- condition: is hp above zero?
  print("ok", 1, 1, 7)         -- body: print draws the word ok
elseif hp == 0 then            -- another condition
  print("empty", 1, 1, 7)
else
  print("no", 1, 1, 7)         -- body when every condition was false
end

if hp > 0 print(hp, 1, 32, 7)  -- one call, no then/end. draws 2
```

Bucle

O buclă repetă un corp. `while` verifică mai întâi condiția și continuă cât timp aceasta rămâne adevărată. `repeat` execută corpul, apoi verifică, astfel încât corpul să se execute întotdeauna cel puțin o dată. `for` numără. Ultimul număr este inclus. Fără un al treilea număr, pasul este 1, deci dacă începutul este deja trecut de sfârșit, corpul nu se execută. Numărătoarea inversă necesită un pas negativ.

```
n = 0
while n < 3 do         -- condition: keep going while n is under 3
  n += 1
  print(n)             -- 1, then 2, then 3: how many times the body has run
end

n = 0
repeat
  n += 1
  print(n)             -- body first, then the condition. prints 1, then 2, then 3
until n > 2

for i = 1, 3 do        -- i is 1, then 2, then 3
  print(i)
end

for i = 0, 4, 2 do     -- start, end, step of 2
  print(i)             -- 0, then 2, then 4
end

for i = 3, 1, -1 do    -- counting down needs the step
  print(i)             -- 3, then 2, then 1
end

for i = 1, 5 do
  if i == 3 then break end  -- leave the loop when i reaches 3
  print(i)                  -- 1, then 2
end
```

Mergând pe o masă

`all` vizitează fiecare element al unei liste, începând cu primul. `pairs` vizitează fiecare nume și valoarea stocată sub acel nume.

```
t = {1, 2, 3}                -- list: t[1] is 1, t[2] is 2, t[3] is 3
for v in all(t) do           -- v is 1, then 2, then 3
  print(v)                   -- print draws that number
end

player = {x = 10, y = 20}    -- record: a field x and a field y
for k, v in pairs(player) do
  print(k)                   -- k is the name, "x" then "y"
end
```

Funcții

O funcție este o rețetă denumită. Forma este `function name(arguments)`, apoi un corp, apoi `end`. O apelezi scriind numele și parantezele. `return` trimite înapoi o valoare și poate trimite mai multe. Argumentele sunt valori locale ale acelei funcții.

```
function greet(who)       -- who is a local that holds what you pass in
  return "hi "..who       -- .. glues two texts. return sends the result back
end

print(greet("ada"))       -- draws hi ada
```

```
function pair()
  return 1, 2             -- two values
end

a, b = pair()             -- a becomes 1, b becomes 2
```

`local function` este o funcție care poate vedea elementele locale din jurul ei. Aceasta este o închidere. Un tabel poate conține una, iar aceasta se apelează cu un punct.

```
function make()
  local n = 0
  local m = {}
  m.bump = function()     -- this function sees n
    n += 1
    return n
  end
  return m
end

counter = make()
print(counter.bump())     -- 1
print(counter.bump())     -- 2
```

Da, nu și alegerea

`not` inversează o condiție. `and` este adevărat numai când ambele părți sunt adevărate. `or` este adevărat când oricare dintre părți este adevărată. `a and b or c` alege `b` când `a` este adevărat, altfel `c`. `cond ? a : b` este aceeași idee într-un singur rând: `a` când condiția este adevărată, `b` când nu este.

```
hp = 2
hit = false
night = false

if hp > 0 and not hit then         -- both parts are true, so the body runs
  print("safe", 1, 1, 7)           -- draws safe
end

label = hp > 0 and "ok" or "out"   -- hp is above 0, so label is ok
print(label, 1, 8, 7)

shade = night ? 1 : 7              -- night is false, so shade is 7
print(shade, 1, 16, 7)
```

Matematică și text

`+ - * /` reprezintă adunarea, scăderea, înmulțirea și împărțirea. `//` împarte și păstrează numărul întreg. `%` este restul. `^` este puterea. `..` unește două texte. `#t` reprezintă numărul de elemente dintr-o listă. Comparațiile sunt `==` (aceeași), `~=` sau `!=` (diferiți) și `< > <= >=`. `x += 1` adună 1 la x. Aceeași formă funcționează și pentru `-= *= /= ^= %=`.

```
scores = {10, 20, 30}    -- the list named scores
tiles = 15 // 8          -- 1, the fraction is dropped
left = 15 % 8            -- 7, what is left over
title = "hi .."ada"      -- the text hi ada
n = #scores              -- how many items are in the list named scores
print(n, 1, 1, 7)        -- 3
print(title, 1, 8, 7)

score = 0
best = 10
if score ~= best then    -- condition: score is not the best
  score += 10            -- add 10 to score
end
print(score, 1, 16, 7)   -- 10
```

Unde locuiește un localnic

`local` durează până când `end` își închide blocul. `if` / `end` este propriul său bloc. Puteți menționa o comandă locală deasupra liniei care o setează, dar până când acea linie nu rulează, valoarea este `nil`.

```
if true then
  print(coins)       -- nil: the next line has not run yet
  local coins = 5
  print(coins)       -- 5
end
-- coins does not exist out here
```

Jumping

`::label::` marchează o linie. `goto label` sare acolo, dar numai în interiorul aceleiași funcții.

```
x = 0
::again::
x += 1                      -- add 1
if x < 3 then goto again end  -- condition: jump back while x is under 3
```

Memoria, pe scurt

`peek` și `poke` vizualizează pixelii activi printr-o singură adresă de 16 biți. Nu păstrează o a doua copie a cartușului jocului. Adresele `0x5F00` până la `0x5FFF` sunt registre: cameră, clip (doar citire), culoare de desen, model de umplere, cele trei remapări ale paletei și numerele de pagină. Octeții de pe ambele părți ale acelei găuri, 65280 dintre ei, reprezintă fereastra de pixeli.

Când ecranul încape în cei 65280 de octeți, 64×64 și 128×128 încape, iar 256×256 nu, ecranul începe de la adresa 0, rând cu rând. Ceea ce rămâne este foaia de sprite selectată de `0x5F41` (0 este foaia 0). Dacă acea foaie este mai mare decât spațiul rămas, `0x5F42` alege pagina, iar harta nu se află în fereastră. Dacă foaia încape, coada este formată din celulele hărții selectate de `0x5F43`, câte unul sau doi octeți fiecare, iar `0x5F44` parcurge paginile hărții atunci când aceasta nu încape în coadă.

Când ecranul nu se încadrează, întreaga fereastră este o pagină de 65280 pixeli. `poke(0x5F40, n)` întoarce pagina ecranului. `poke(0x5F41, n)` cu n diferit de 0 afișează foaia n în schimb, iar foaia 0 nu are vizualizare în acest mod. `poke(0x5F43, n)` cu n diferit de 0 afișează harta n, iar aceasta câștigă în fața foii. Harta 0 în acest mod este doar `mget` și `mset`.

Un `peek` care nu atinge un pixel sau un registru returnează 0. Un `poke` nu face nimic acolo. Indicatoarele Sprite, SFX și tabelele DATA nu se află în această fereastră. Folosiți `fget`, `sfxget` și `dget`.

Utilizare practică

Fiecare funcție este listată în partea de jos a acestei pagini, cu utilizări practice ale funcțiilor PIX37. Linkurile de sub fiecare exemplu oferă mai multe informații despre funcții.



Mișcare lină

Un cartuș de joc 128×128, văzut de sus. Sprite-ul 1 este deja desenat. Săgețile accelerează în orice direcție, iar frecarea te încetinește. Sprite-ul se înfășoară: o jumătate care pleacă de jos este desenată în partea de sus, și același lucru se întâmplă și cu celelalte margini.

`_init` `_update` `_draw` `btn` `mid` `cls` `spr` `palt`



Verificarea dacă următoarea mișcare este blocată

Un cartuș de joc de 64×64. Podeaua și peretele din dreapta sunt deja pe hartă, pictate cu sprite-ul 3, iar steagul 0 este setat pe acel sprite. Săgețile mișcă sprite-ul 1. Cuvântul din partea de sus spune „blocat” atunci când următorul pas s-ar suprapune pe o placă solidă și „liber” atunci când nu s-ar suprapune.

`flr` `fget` `mget` `btn` `cls` `map` `spr` `print` `_update` `_draw`



O ștergere cu flip

Apăsați A. Funcția „ștergere” este desenată de flip(), câte un rând negru pe rând, în loc să fie în interiorul funcției _draw(). _draw nu intervine în timpul rulării acesteia, deci fiecare rând rămâne pe ecran.

`rectfill` `flip` `btnd` `cls` `print` `_update` `_draw`



Un tabel de funcții (Clasă)

Sprite-ul 1 este deja desenat. make_monster funcționează ca o clasă. Returnează un tabel, iar update și draw sunt metode care păstrează valorile x și y ale monstrului respectiv. Numărul din partea de sus indică distanța parcursă.

`spr` `cls` `print` `flr` `_init` `_update` `_draw`



Patru moduri de a citi un buton

A este butonul 4. Țineți-l apăsat și încărcarea crește. Eliberați-l și încărcarea revine la 0. Apăsarea sus și jos se repetă cât timp este ținut apăsat, astfel încât rândul de meniu să se miște continuu. btnd este primul cadru al unei apăsări, pe care acest cartuș de joc îl folosește doar ca început al acelei apăsări.

`btn` `btnd` `btnp` `btnu` `cls` `print` `_update` `_draw`



Trage cu mouse-ul

Sprite-ul 1 este deja desenat în mijloc. Un punct alb este cursorul. Clic stânga declanșează un pixel spre el. Rotița schimbă viteza, imprimată în colț.

`scroll` `clickl` `mouse` `sqrt` `add` `all` `cls` `spr` `pset` `print` `_update` `_draw`



Tastați un nume și confirmați-l

Apăsați A pentru a deschide câmpul. Tasta sus și jos schimbă o literă care clipește. A o scrie, X șterge, Y este spațiu, iar B confirmă. Linia afișează apoi numele dvs.

`btnd` `keyon` `keystr` `flr` `time` `cls` `print` `_update` `_draw`



Cameră pe player

Un cartuș de joc de 128×128. Sprite-ul 1 începe în mijlocul unei podele de plăci. Săgețile se mișcă. Camera menține sprite-ul centrat. x și y în colț nu se derulează, deoarece camera() fără argumente readuce la vedere HUD-ul.

`btn` `screen` `camera` `cls` `map` `spr` `print` `flr` `_update` `_draw`



Câteva rânduri dintr-un roman vizual

Liniile sunt în DATA, în engleză și poloneză. A continuă sau alege să rămână atunci când o opțiune este pe ecran. B alege să iasă de pe o opțiune și, în caz contrar, schimbă limba. X schimbă limba în timp ce o opțiune este afișată. Dialogul folosește fontul de 8px, astfel încât literele poloneze sunt desenate.

`dstr` `dget` `btnd` `cls` `print8` `print` `_update` `_draw`



Pictează cu „peek and poke”

Un cartuș de joc 64×64, deci adresa 0 este pixelul din stânga sus, iar următorul rând începe de la 64. Cartușul de joc scrie pixeli, copiază un rând, șterge un rând și afișează valorile pe care le-a citit.

`cls` `poke` `peek` `poke2` `peek2` `poke4` `peek4` `memcpy` `memset` `print` `_draw`



O listă la care poți adăuga

A adună un număr, până la 8. B îl elimină pe ultimul. X îl elimină pe primul. Lista este afișată pe ecran.

`btnd` `count` `add` `deli` `cls` `ipairs` `print` `pack` `unpack` `next` `foreach` `select` `pcall` `assert` `del` `_update` `_draw`



Un nume destrămat

Cartușul de joc divide cărțile în ada, 12, albastru și imprimă numele, prima literă a acestuia, 12, și cuvântul albastru.

`cls` `split` `sub` `chr` `ord` `tonum` `tostr` `print` `_draw`



Cântă o notă, apoi schimb-o

Modelul 0 are o lungime de o notă, iar fiecare reluare are aceeași intensitate. A îl redă din nou când nota respectivă s-a terminat. B ridică înălțimea și redă nota respectivă.

`sfxset sfxhdr sfx btnd sfxstat sfxget cls print _init _update _draw`

 O melodie scrisă în cartușul jocului

Două modele se repetă în buclă pe rândul 0, câte unul pe fiecare dintre primele două canale. Rândul rămâne la 0. ch0 și ch1 sunt nota pe care se află fiecare canal. Editarea modelului, a parametrului LEN și a parametrului LOOP în cartușul jocului este o modalitate de a vedea ce fac aceste comenzi în panoul muzical.

`songget songset music musicstat cls print _init _draw`

 Trei monștri în DATA

Panoul DATA are deja o coloană cu nume și o coloană cu puncte de viață (hp), cu slime, liliac și ochi. A rănește slime-ul. Numerele din dreapta reprezintă punctele sale de viață.

`drows dstr dget add btnd dset dcols cls print all _init _update _draw`

 Rețineți scorul

A adaugă un punct și îl salvează. Ieși din joc și deschide din nou cartușul jocului: numărul este încă acolo. B îl uită.

`load btnd save delsave cls print _init _update _draw`

  Un al doilea capitol într-un alt fișier

Două cartușe de joc împart misiunea slotului de salvare. Partea 1: A marchează jocul terminat. Partea 2, deschisă prima, spune să se termine partea 1. După aceea, se deschide partea 2, iar B șterge salvarea, astfel încât partea 1 să poată fi jucată din nou.

`load saveslot btnd save delsave cls print _init _update _draw`

 O realizare

Sprite-ul 8 este deja desenat. Apăsați A de zece ori. Toastul denumește realizarea, iar sprite-ul 8 rămâne pentru că este deblocat. B șterge acea realizare din fișierul local și resetează numărătoarea.

`achdef btnd achieved achieve achclear cls print spr _init _update _draw`

 O cameră în care poți intra

Un cartuș de joc de 256×256. Un punct alb este cursorul. Meniul este gazdă, alăturare sau lista de servere. Gazda deschide o cameră numită walk și începe jocul. Join derulează o adresă (cifre, un punct, două puncte), iar B se conectează. Lista de servere se mișcă cu susul și în jos, iar A se alătură rândului. În joc, săgețile parcurg sprite-ul 1. O monedă este partajată: gazda o mută, iar ridicarea ei adaugă 1 la scorul acelui jucător și generează următorul. Fiecare sprite, inclusiv al tău, arată scorul respectiv lângă el. Colțul arată câți jucători sunt în joc, adresa LAN și adresa publică a gazdei. B pleacă: tu și ceilalți vă întoarceți la meniu, iar următoarea gazdă începe o cameră nouă la scorul 0.

`btn` `btnd` `mouse` `pset` `spr` `print` `flr` `rnd` `pairs` `nlisten` `nhost` `nannounce` `nlist` `njoin` `nleave` `nsend` `nkb` `nkbstr` `whoami` `nplayers` `nlan` `_mp` `_mp_join` `_mp_leave` `_mp_fail` `palt` `cls` `time` `_init` `_update` `_draw`



Dialog care așteaptă

A arată linia următoare. Funcția se întrerupe la fiecare cedare până când apăsați din nou A.

`yield` `cocreate` `btnd` `coresume` `cls` `print` `_init` `_update` `_draw`



Așteaptă o jumătate de secundă

Cuvântul se schimbă din ... în hi după aproximativ o jumătate de secundă. Ecranul continuă să se deseneze în timp ce corutina este pusă în pauză.

`yield` `cocreate` `costatus` `coresume` `cls` `print` `_init` `_update` `_draw`



Cercuri care sosesc unul câte unul

Cinci cercuri picurează de sus, la două cadre distanță, apoi cad.

`add` `yield` `cocreate` `coresume` `all` `cls` `circfill` `_init` `_update` `_draw`



Două cadre, de la ceas

Sprite-ul 1 și sprite-ul 2 sunt două cadre ale aceluiași personaj. Imaginea se schimbă de patru ori pe secundă.

`cls` `flr` `time` `spr` `_draw`

Fiecare funcție

Argumentele din paranteze pătrate sunt opționale. Indexurile tabelelor și șirurilor de caractere încep de la 1, cu excepția rândurilor și coloanelor DATA, care încep de la 0. Culorile sunt indexuri de paletă.

Index

`_desenează` `_init` `_mp`

_mp_fail
_mp_join
_mp_pleacă
_mp_update
_set_fps
_actualizare
abs
achclear
achdef
realiza
realizat
adaugă
toate
afirmă
atan2
bandă
bnot
bor
btn
btnd
btnp
btnu
bxor
aparat de fotografiat
plafon
chr
circ
circfill
clickl
clickm
clickr
clip
cls
cocreează
culoare
coresume
cos
costatus
numără
cursorul
dcols
del
delicatese
delsave
dget
drows
dset
dstr
fget
fillp
flip
flr
pentru fiecare
fset
info
ipairs
tastatura
keyon
keystat
keystr
linie
încărcătură
jurnal
lshr
harta
max
memcpy
memset
mget
mijlocul
min
mouse-ul
mset
muzica
musicstat
nannounce
ncall
în continuare
nhost
njoin
nkb
nkbstr
nlan
npleacă
nlist
nascult
notă
nplayers
nsend
nstun
ord
oval
ovalfill
pachet
perechi
prietene
palrgb
palt
pcall
privirea
privirea2
peek4
perf
pget
poke
poke2
poke4
imprima
print10
print8
pset
rect
rectfill
rnd
rotl
rotr
corect
rectfill
salva
slot de salvare
ecran
parcurge
selectați

setmap	sgn	sset
setpalette	shl	sspr
setspr	statistici	stat
setex	shr	sub
sectorn	păcatul	timp
sfx	cântec	tline
sfxget	set de cântece	tonum
sfxhdr	despicat	tostr
sfxset	spr	despacheteaza
sfxstat	sqrt	whoami
sget	srand	randament

Ciclu de viață

_init

`_init() ... end`

Rulează o singură dată când pornește cartușul jocului. Codul care se află în afara oricărui hook este, de asemenea, inclus în _init.

```
_init()
  x = 64
  y = 40
end
```

_update

`_update() ... end`

Logică de joc. Apelat la 60 Hz, chiar și atunci când _draw este limitat sau gazda este mai lentă. Citiți butoanele aici.

```
_update()
  if btn(0) then x -= 1 end
  if btn(1) then x += 1 end
end
```

_draw

```
_draw([fps]) ... end
```

Desenează un cadru. Fără argument sau cu valoarea 0, rulează ori de câte ori este prezent gazda. Se trece n (1..999) pentru a limita desenarea la n cadre pe secundă. _update rămâne la 60 Hz.

```
_draw(30)
  cls(1)
  spr(1, x, y)
end
```

_set_fps

```
_set_fps(n)
```

Modifică limita _draw în timp ce jocul rulează. 0 înseamnă nelimitat. n este fixat la 0..999.

```
if btn(4) then _set_fps(60) end
```

flip

```
flip()
```

Prezintă ecranul și trece la următorul cadru. Folosește-l doar dacă desenezi propria buclă de cadre. Nu aștepta după aceea.

```
cls(0)
print("tick", 4, 4, 7)
flip()
```

time

```
time()
```

Secunde de la _init.

```
print(time(), 4, 4, 7)
```

Intrare

btn

`btn([b], [p])`

1 cât timp butonul b este apăsat, altfel 0. p este jucătorul, 0..7. Fără argumente, returnează o mască de biți a jucătorului 0. Butoane: 0 stânga, 1 dreapta, 2 sus, 3 jos, 4 A, 5 B, 6 X, 7 Y, 8 R, 9 L, 10 selecție, 11 start.

```
if btn(4) then shoot() end
```

btnd

`btnd(b, [p])`

Adevărat doar în cadrul _update când butonul b este apăsat.

```
if btnd(5) then menu_open = not menu_open end
```

btnp

`btnp([b], [p])`

Adevărat la apăsare, apoi din nou cât timp este ținut apăsat: 30 de tic-uri de întârziere, apoi la fiecare 8 tic-uri. Fără buton, returnează o mască.

```
if btnp(2) then cursor -= 1 end
```

btnu

`btnu(b, [p])`

Adevărat doar pe cadrul în care butonul este eliberat.

```
if btnu(4) then charge = 0 end
```

mouse

`mouse()`

Returnează valorile x și y ale mouse-ului în pixelii jocului.

```
mx, my = mouse()
pset(mx, my, 7)
```

clickl

`clickl([p])`

Butonul stâng al mouse-ului. clickl() este ținut apăsat. clickl(1) este muchia de apăsare.

```
if clickl(1) then place(mouse()) end
```

clickr

`clickr([p])`

Butonul dreapta al mouse-ului. Aceeași muchie cu 1 argument ca și clicl.

```
if clickr() then cancel() end
```

clickm

`clickm([p])`

Butonul din mijloc sau un clic al rotiței de derulare.

```
if clickm(1) then drag = true end
```

scroll

`scroll()`

Mișcarea rotiței mouse-ului în acest cadru. Pozitiva și negativă depind de direcția rotiței.

```
cam_y -= scroll()
```

keyon

`keyon()`

Deschide un câmp text controlat de butoane. Tastele sus și jos schimbă o literă de la A la Z. A scrie litera respectivă. X șterge. Y este spațiu. B confirmă și închide câmpul. Tastatura nu tastează în el.

```
if btnd(4) then keyon() end
```

keyoff

`keyoff()`

Ascunde câmpul de text.

```
keyoff()
```

keystr

`keystr([s])`

Text în câmp, apoi 1 pe cadrul B confirmă, apoi litera de sus și de jos se schimbă. Litera respectivă devine goală odată ce câmpul se închide. keystr("") șterge textul.

```
name, done, letter = keystr()
if done == 1 then start(name) end
```

keystat

`keystat()`

1 cât timp câmpul de text este deschis, altfel 0.

```
if keystat() == 1 then return end
```

Desen

screen

`screen()`

Dimensiunea ecranului cartușului de joc curent, în pixeli. Returnează lățimea, apoi înălțimea. Aceasta este rezoluția jocului, nu fereastra editorului.

```
w, h = screen()
```

cls

`cls([col])`

Șterge ecranul. Culoarea implicită este 0.

```
cls(1)
```

color

`color(col)`

Setează culoarea de desenare utilizată atunci când un apel ulterior omite col. Fără argument, returnează culoarea curentă.

```
color(7)
print("hi")
```

camera

`camera([x, y])`

Compensează fiecare apel de desenare. camera() se resetează la 0, 0.

```
camera(cam_x, cam_y)
```

clip

`clip([x, y, w, h, clip])`

Limitează desenarea la un dreptunghi. Un al cincilea argument diferit de zero se intersectează cu clipul curent. clip() restaurează ecranul complet. Vizualizarea registrelor clipului este doar pentru citire.

```
clip(8, 8, 48, 48)
circfill(32, 32, 20, 8)
clip()
```

cursor

`cursor([x, y, col])`

Setează unde va fi imprimată următoarea imagine fără coordonate și, opțional, culoarea. cursor() returnează x, y.

```
cursor(4, 4, 7)
print("line 1")
print("line 2")
```

pset

`pset(x, y, [col])`

Setează un pixel pe ecran. Se numără în statistica px.

```
pset(x, y, 10)
```

pget

`pget(x, y)`

Culoarea pixelului ecranului la x, y.

```
if pget(x, y+8) == 3 then grounded = true end
```

print

`print(v, [x, y, col])`

Desenează text 3×5 (UTF-8, decupat din fontul Fusion). Fără x și y, se afișează la nivelul cursorului și se mută la linia următoare.

```
print("score "..score, 2, 2, 7)
```

log

`log(...)`

Adaugă argumentele, separate prin spații, la un buffer de text. pix37_run afișează acel buffer pe stdout. Nu desenează.

```
log("hp", hp)
```

print8

`print8(v, [x, y, col])`

La fel ca la imprimare, cu fontul Fusion de 8 px.

```
print8("pause", 16, 28, 7)
```

print10

`print10(v, [x, y, col])`

La fel ca la imprimare, cu fontul Fusion de 10 px.

```
print10("pix37", 8, 8, 10)
```

line

`line([x0, y0, x1, y1, col])`

Linie de la x0,y0 la x1,y1. Două argumente continuă de la capătul anterior. line() resetează acel lanț.

```
line(0, 32, 63, 32, 6)
```

circ

`circ(x, y, r, [col])`

Conturul cercului. Raza este în pixeli.

```
circ(32, 32, 12, 12)
```

circfill

`circfill(x, y, r, [col])`

Cerc umplut.

```
circfill(x, y, 3, 8)
```

oval

`oval(x0, y0, x1, y1, [col])`

Conturul elipsei în interiorul casetei de încadrare. Colțurile sunt inclusive.

```
oval(8, 20, 56, 44, 7)
```

ovalfill

`ovalfill(x0, y0, x1, y1, [col])`

Elipsă umplută într-o casetă de încadrare.

```
ovalfill(10, 40, 54, 60, 3)
```

rect

`rect(x0, y0, x1, y1, [col])`

Contur dreptunghiular. Colțurile sunt inclusive.

```
rect(0, 0, 63, 63, 7)
```

rectfill

`rectfill(x0, y0, x1, y1, [col])`

Dreptunghi umplut.

```
rectfill(0, 56, 63, 63, 3)
```

rrect

`rrect(x, y, w, h, [r, col])`

Contur dreptunghiular rotunjit. x și y reprezintă colțul din stânga sus. w și h reprezintă dimensiunea în pixeli; ambele trebuie să fie mai mari de 0. r este raza colțului și implicit este 0. Este fixată la jumătate din latura mai scurtă.

```
rrect(8, 8, 48, 16, 3, 7)
```

rrectfill

`rrectfill(x, y, w, h, [r, col])`

Dreptunghi rotunjit umplut. Aceleași argumente ca și rrect.

```
rrectfill(8, 40, 48, 16, 2, 1)
```

fillp

`fillp([p])`

Model de umplere pe 16 biți pentru formele umplute ulterior. 0 este solid. fillp() returnează modelul curent. Biții parcurg modelul 4×4.

```
fillp(0xA5A5)
rectfill(8, 8, 40, 40, 10)
fillp(0)
```

pal

`pal([c0, c1, [mode]])`

Remapează culoarea c0 la c1. Modul 0 este paleta de desenare, 1 paleta de afișare, 2 paleta de umplere a modelului. pal() se resetează.

```
pal(8, 10)
circfill(32, 32, 8)
```

palrgb

`palrgb(col, r, g, b)`

Setează o intrare din paletă la RGB, fiecare canal de la 0 la 255.

```
palrgb(8, 255, 80, 40)
```

palt

`palt([col, t])`

Transparență. palt(col) sau palt(col, true) face ca culoarea să fie transparentă. palt(col, false) o face solidă. palt() face ca fiecare culoare să fie opacă.

```
palt(0)
spr(1, x, y)
```

setpalette

`setpalette(n)`

Selectează banca de palete n. setpalette() returnează banca curentă.

```
setpalette(1)
```

tline

`tline(x0, y0, x1, y1, mx, my, [mdx, mdy])`

Desenează o linie pe ecran în timp ce eșantionează pixelii hărții de la pozițiile hărții mx, my, mdx, mdy și deplasează pas cu pas eșantionul hărții de-a lungul liniei.

```
tline(0, 40, 63, 20, 0, 0, 0.5, 0)
```

Sprite-uri și hărți

spr

`spr(n, x, y, [w, h, fx, fy, tex, rot])`

Desenează sprite-ul n la x, y. w și h sunt dimensiunea în pixeli, implicit 8. fx și fy se răstoarnă pe orizontală și verticală când sunt diferite de zero. tex este tabulatorul foii de sprite. rot este unghiul în rotații (0..1), aceeași direcție ca sin și cos. 0 lasă sprite-ul nerotit.

```
spr(1, x, y, 8, 8, 0, 0, 0, aim)
```

sspr

`sspr(sx, sy, sw, sh, dx, dy, [dw, dh, fx, fy, tex])`

Copiază un dreptunghi al foii de sprite pe ecran, scalat la dw cu dh. Valorile implicite păstrează dimensiunea sursei.

```
sspr(0, 0, 8, 8, 16, 16, 32, 32)
```

sget

`sget(x, y, [tex])`

Culoarea unui texel pe foaia de sprite.

```
c = sget(0, 0)
```

sset

`sset(x, y, [col, tex])`

Scrie un texel pe foaia de sprite.

```
sset(3, 3, 8)
```

fget

`fget(n, [f, tex])`

Indicatori de sprite. fget(n) returnează octetul de indicator. fget(n, f) este adevărat când bitul de indicator f (0..7) este setat.

```
if fget(tile, 0) then solid = true end
```

fset

`fset(n, [f], v, [tex])`

Setează flag-urile pentru sprite. fset(n, v) scrie întregul octet. fset(n, f, v) setează un bit.

```
fset(2, 0, true)
```

setspr

`setspr(n)`

Selectează foaia de sprite activă. setspr() returnează foaia curentă. Aceeași bancă ca settex.

```
setspr(1)
```

settex

`settex(n)`

Alias al lui setspr.

```
settex(0)
```

map

`map(cx, cy, sx, sy, [cw, ch, layer, tex])`

Desenează harta. Celula cx, cy a hărții este desenată la ecranul sx, sy. cw, ch cu valoarea 0 înseamnă întreaga hartă. Stratul este acceptat și ignorat. tex este foaia setului de tile-uri.

```
map(0, 0, 0, 0, 16, 16)
```

mget

`mget(cx, cy)`

ID-ul sprite-ului în celula hărții cx, cy.

```
if fget(mget(tx, ty), 0) then bounce() end
```

mset

`mset(cx, cy, n)`

Scrie sprite-ul n în celula de hartă cx, cy.

```
mset(3, 4, 0)
```

setmap

`setmap(n)`

Selectează harta n. setmap() returnează harta curentă. Harta 0 este o hartă reală.

```
setmap(1)
```

Memorie

peek

`peek(addr)`

Citește un octet din fereastra de 64 KB. Fereastra este ecranul live, apoi pixelii sprite, apoi mapează celulele când ecranul încape în 65280 octeți. Ecranele mai mari sunt paginate cu poke(0x5F40, page). Adresele care nu sunt mapate returnează 0. Se scrie și @addr.

```
px = peek(0)
```

poke

`poke(addr, ...)`

Scrie octeți la addr. Argumentele suplimentare scriu următorii octeți. Adresele nemapate sunt ignorate. De asemenea, este scris $addr, val. Punctajul pe ecran este considerat o scriere de pixel.

```
poke(0x5F00, 4) -- camera x
```

peek2

`peek2(addr)`

Valoare nesemnată pe 16 biți, little-endian, la adresa.

```
w = peek2(0x6000)
```

poke2

`poke2(addr, val)`

Scrie o valoare pe 16 biți fără semn, little-endian.

```
poke2(0x6000, 1000)
```

peek4

`peek4(addr)`

Valoare fără semn pe 32 de biți, little-endian.

```
n = peek4(0x6000)
```

poke4

`poke4(addr, val)`

Scrie o valoare pe 32 de biți fără semn, little-endian.

```
poke4(0x6000, 1)
```

memcpy

`memcpy(dst, src, len)`

Copiază len octeți în aceeași fereastră de 64 KB.

```
memcpy(0x6000, 0x6100, 64)
```

memset

`memset(dest, val, len)`

Completează octeții len începând de la dest cu val.

```
memset(0, 0, 64)
```

Matematică

abs

`abs(x)`

Valoare absolută.

```
d = abs(tx - x)
```

flr

`flr(x)`

Rotunjește spre minus infinit.

```
tx = flr(x / 8)
```

ceil

`ceil(x)`

Rotunjește spre plus infinit.

```
n = ceil(hp / 8)
```

max

`max(a, b)`

Cea mai mare dintre cele două valori.

```
x = max(0, x)
```

min

`min(a, b)`

Cea mai mică dintre cele două valori.

```
x = min(63, x)
```

mid

`mid(a, b, c)`

Mijlocul celor trei valori. Util ca clemă.

```
x = mid(0, x, 63)
```

sgn

`sgn(x)`

Semnul lui x: -1, 0 sau 1.

```
x += sgn(dir)
```

sqrt

`sqrt(x)`

Rădăcină pătrată. Un argument negativ nu este un număr.

```
dist = sqrt(dx*dx + dy*dy)
```

sin

`sin(x)`

Sinusul x este în rotații: 0, 0,25, 0,5, 0,75 fac o înconjurare a cercului o dată. Nu radiani.

```
y = 32 + sin(time()) * 8
```

cos

`cos(x)`

Cosinusul, la rândul său, are aceeași convenție ca și sinusul.

```
x = 32 + cos(time()) * 8
```

atan2

`atan2(dx, dy)`

Unghiul vectorului dx, dy, în viraje de la 0 la 1.

```
aim = atan2(mx - x, my - y)
```

rnd

```
rnd([x])
```

Număr aleator în [0, x). Implicit x este 1, deci rnd() este 0..1. Pentru un zar, flr(rnd(6))+1.

```
c = flr(rnd(16))
```

srand

```
srand(seed)
```

Semnează generatorul aleatoriu, astfel încât o secvență rnd ulterioară să poată fi repetată.

```
srand(1)
```

band

```
band(a, b)
```

ȘI la nivel de bit. Se scrie și a și b.

```
if band(flags, 1) ~= 0 then hit() end
```

bor

```
bor(a, b)
```

SAU la nivel de bit. Se scrie și a | b.

```
flags = bor(flags, 4)
```

bxor

```
bxor(a, b)
```

XOR la nivel de biți. Se scrie și a ^^ b.

```
n = bxor(n, 255)
```

bnot

`bnot(a)`

NU la nivel de bit. Se scrie și ~a.

```
mask = bnot(1)
```

shl

`shl(a, n)`

Shift la stânga. Se scrie și un << n.

shr

`shr(a, n)`

Deplasare aritmetică la dreapta. Se scrie și cu >> n.

```
n = shr(n, 1)
```

lshr

`lshr(a, n)`

Deplasare logică la dreapta. Se scrie și cu >>> n.

```
n = lshr(n, 4)
```

rotl

`rotl(a, n)`

Rotește biții la stânga. Se scrie și un <<> n.

```
n = rotl(n, 1)
```

rotr

`rotr(a, n)`

Rotește biții la dreapta. Se scrie și cu >><n.

Tabele și valori

add

`add(t, v, [i])`

Adaugă v la tabelul t. Dacă este dat i, inserează la acel index bazat pe 1.

```
add(bullets, {x=x, y=y})
```

del

`del(t, v)`

Elimină prima intrare egală cu v.

```
del(items, gem)
```

deli

`deli(t, [i])`

Elimină indexul i și returnează valoarea respectivă. Fără i, elimină ultima intrare.

```
local b = deli(bullets, i)
```

all

`all(t)`

Iterator peste partea de matrice, în ordine. Găurile opresc parcurgerea în modul în care se termină o matrice bazată pe 1.

```
for b in all(bullets) do
  b.x += 2
end
```

foreach

`foreach(t, fn)`

Apelează funcția fn(v) pentru fiecare element al tabloului.

```
foreach(enemies, draw_enemy)
```

pairs

`pairs(t)`

Iterator peste fiecare cheie și valoare, inclusiv câmpuri denumite.

```
for k, v in pairs(player) do
  print(k)
end
```

ipairs

`ipairs(t)`

Iterator peste partea de matrice cu indexul bazat pe 1.

```
for i, v in ipairs(deck) do
  spr(v, i*8, 8)
end
```

next

`next(t, [k])`

Returnează următoarea cheie și valoare după k. next(t) începe de la început. Folosită de perechi.

```
k, v = next(t)
```

count

`count(t, [v])`

Lungimea părții de matrice. Dacă este dat v, numără câte intrări sunt egale cu v.

```
if count(gems) == 0 then win() end
```

pack

`pack(...)`

Impachetează argumentele într-un tabel nou.

```
t = pack(1, 2, 3)
```

unpack

`unpack(t, [i, j])`

Returnează valorile de la t[i] la t[j] ca valori separate. Valorile implicite acoperă întreaga parte a matricei.

```
a, b = unpack(t)
```

select

`select(i, ...)`

select("#", ...) returnează numărul de argumente care urmează. select(i, ...) returnează argumentele de la i încolo.

```
n = select("#", 1, 2, 3)
```

assert

`assert(c, [msg])`

Se oprește cu o eroare când c este fals sau nil. msg înlocuiește textul implicit.

```
assert(hp > 0, "dead")
```

pcall

`pcall(f, ...)`

Apelează f. În caz de succes, returnează „true” și rezultatele. În caz de eroare, returnează „false” și textul erorii.

```
ok, err = pcall(load_level, 3)
```

Corzi

chr

`chr(n)`

Un caracter dintr-un cod de octeți de la 0 la 255.

```
s = chr(65) -- A
```

ord

`ord(s, [i, j])`

Coduri de octeți ale lui s de la indexul i la j. Indexurile sunt bazate pe 1. Implicit, se folosește primul caracter.

```
code = ord("a")
```

sub

`sub(s, i, [j])`

Subșir de la i la j, bazat pe 1. Omiteți j pentru a lua coada.

```
name = sub(keystr(), 1, 8)
```

split

`split(s, [sep, conv])`

Împarte s într-un tabel. sep are implicit virgulă. Un număr sep este lățimea unui bloc. conv convertește bucățile cu tonum când este adevărat.

```
words = split("red,blue", ",")
```

tonum

`tonum(s, [base])`

Analizează un număr. Baza poate fi 2..36. Returnează nil când textul nu este un număr.

```
n = tonum("255")
```

tostr

`tostr(v, [digits])`

Transformă o valoare în text. cifre setează zecimale pentru numere.

```
print(tostr(hp, 0), 4, 4, 7)
```

Audio

sfx

`sfx(n, [ch, offset, len])`

Redă modelul n și returnează canalul. Omite ch sau trece -1 și este ales primul canal liber. ch -2 oprește modelul n pe fiecare canal. n -1 oprește un canal sau fiecare canal când ch este omis. n -2 permite unui model în buclă să se termine în loc să se repete. sfx() fără argumente returnează -1.

```
ch = sfx(0)
```

sfxget

`sfxget(n, step)`

Returnează un tabel cu un pas al modelului: [1] înălțime, [2] undă, [3] volum, [4] efect. Înălțimea este 0..63 (C-0 până D#5). Undă este 0..15. Volumul este 0..7, iar 0 este silențios. Efectul este 0..7.

```
step = sfxget(0, 0)
pitch = step[1]
```

sfxset

`sfxset(n, step, pitch, wave, vol, fx)`

Scrie un pas în model. Toate cele șase argumente sunt obligatorii.

```
sfxset(0, 0, 24, 3, 5, 0)
```

sfxhdr

```
sfxhdr(n, [key, val])
```

Antet model. sfxhdr(n) returnează tabelul de antet. sfxhdr(n, key, val) setează un câmp: speed, loop_in, loop_out, level, noise, crush, dtune, rverb, dampn, swing, steps. Numerele de la 0 la 10 selectează aceleași câmpuri în această ordine. speed este TIC, tic-uri pe pas. Mai mare este mai lent. 15 este 120 de bătăi pe minut. 16 biți acceptă 1 la 255. Sintetizatoarele mai restrânse se fixează pe lista lor TIC.

```
sfxhdr(0, "speed", 15)
```

sfxstat

```
sfxstat([ch])
```

ID-ul modelului se redă pe canalul ch sau -1. Canalul implicit este 0.

```
if sfxstat(0) < 0 then sfx(1) end
```

note

```
note(pitch, [wave, vol, fx, ch, ticks])
```

Redă o notă fără model. înălțime 0..63, undă 0..15, volum 0..7, efect 0..7. Ticurile reprezintă durata de menținere, în aceleași ticuri de 120 Hz ca și TIC. Omiteți ticurile și durează 15, un pas implicit. Efectul rulează pe parcursul acestui timp, apoi canalul devine liniștit.

```
note(36, 3, 5, 5, 0)
```

music

```
music([n, fade, mask])
```

Începe melodia n de sus. music(-1) se oprește. Apelarea funcției music cu ID-ul care este deja redat o lasă să ruleze. music() returnează ID-ul melodiei redate sau -1. fade este în milisecunde: melodia intră sau dispare treptat când n este -1. Biții de mască setați sunt canalele utilizate pentru melodie. 0 înseamnă toate cele opt. Celelalte canale rămân libere pentru efecte speciale.

```
music(0, 1000, 7)
```

musicstat

`musicstat()`

Tabelul melodiei redate. song și row reprezintă ID-ul melodiei și rândul, sau -1 când nu se redă nimic. step și pos au opt numere. step[1] este canalul 0, step[8] este canalul 7. step este indexul notei, sau -1 când canalul respectiv este silențios. pos reprezintă distanța dintre nota respectivă și 0, de la 1. tick reprezintă numărul de ticuri redate deja pe acest rând. played reprezintă numărul de rânduri care au început de la music(), inclusiv cel redat.

```
st = musicstat()
row = st.row
note = st.step[1]
```

songget

`songget(song, [row, ch])`

ID-ul modelului este stocat într-o celulă a melodiei. ch este 0..7. 255 înseamnă că celula este goală. Acesta este apelul live; textul indiciului de cod arată încă o ordine a argumentelor mai veche.

```
id = songget(0, 0, 0)
```

songset

`songset(song, row, ch, pattern)`

Scrie un ID de model într-o celulă de melodie. Folosește 255 pentru a goli celula.

```
songset(0, 0, 0, 1)
```

stat

`stat(n)`

Numai audio. 16..23 este ID-ul efectelor sonore pe canalul n-16. 24..31 este modelul muzical pe canalul n-24. Orice alt n returnează 0. Timpul cadrelor și memoria RAM sunt perf și info, nu statistic.

```
id = stat(16)
```

Tabele de date

dget

`dget(t, row, col)`

Citește o celulă din tabelul de date t din panoul DATA. Rândul și coloana încep de la 0.

```
hp = dget(0, 0, 1)
```

dset

`dset(t, row, col, v)`

Scrie o celulă. Folosește un șir de caractere pentru o coloană de text.

```
dset(0, 0, 0, "slime")
```

dstr

`dstr(t, row, col)`

Celula ca text. Numerele sunt convertite.

```
name = dstr(0, row, 0)
```

drows

`drows(t)`

Câte rânduri are tabelul t.

```
for r = 0, drows(0) - 1 do
end
```

dcols

`dcols(t)`

Câte coloane are tabelul t.

```
cols = dcols(0)
```

Salvare și realizări

save

`save(k, v)`

Stochează v sub cheia k în fișierul de salvare. v poate fi un număr, un șir de caractere sau un tabel al acestora. Fișierul este comprimat.

```
save("hp", hp)
```

load

`load(k)`

Citește cheia k. Returnează nil când cheia lipsește.

```
hp = load("hp") or 3
```

delsave

`delsave(k)`

Șterge tasta k din salvare.

```
delsave("hp")
```

saveslot

`saveslot([name])`

Alege fișierul salvat din folderul autorului. Fără nume, slotul este titlul jocului. Returnează 1 când fișierul respectiv există deja.

```
if saveslot("slot1") == 1 then
  hp = load("hp") or 3
end
```

achieve

`achieve(i)`

Deblochează realizarea i, de la 0 la 15. Stocată local. O versiune Steam setează și ACH_i. O sesiune de catalog raportează realizarea în contul jucătorului.

```
if coins >= 100 then achieve(0) end
```

achieved

`achieved(i)`

1 când realizarea i este deja deblocată.

```
if achieved(0) == 1 then spr(8, 4, 4) end
```

achdef

`achdef(i, text, [spr, tex])`

Text de tip toast pentru realizarea i, afișat timp de 15 secunde din partea de jos a ecranului. spr este o dală opțională de 8×8, textul foii sale.

```
achdef(0, "rich", 8)
```

achclear

```
achclear(i)
```

Șterge realizarea i din fișierul local. Un ACH_i Steam care era deja setat rămâne setat.

```
if btnd(5) then achclear(0) end
```

Reţea

whoami

```
whoami()
```

ID-ul tău de jucător, 0..15, în timp ce o sesiune este în desfășurare. Offline este -1.

```
id = whoami()
```

nhost

```
nhost()
```

1 dacă această mașină este gazda, altfel 0.

```
if nhost() == 1 then spawn() end
```

nplayers

```
nplayers()
```

Câte ID-uri live există în 0..15.

```
print(nplayers(), 4, 4, 7)
```

nlisten

`nlisten()`

Pornește găzduirea UDP și returnează adresa LAN ca ip:port.

```
addr = nlisten()
```

njoin

`njoin(addr)`

Se alătură adresei ip sau ip:port.

```
njoin("10.0.0.8:7777")
```

nleave

`nleave()`

Părăsește sesiunea. Colegii văd _mp_leave cu valoarea „why” 0.

```
if btnd(11) then nleave() end
```

nsend

`nsend([p,] name, ...)`

Trimite un mesaj cu pierderi. Numele este evenimentul. Urmează până la patru valori suplimentare. p de la 0 la 15 șoapte către acel peer; omite p pentru a trimite tuturor.

```
nsend("pos", x, y)
```

ncall

`ncall([p,] name, ...)`

Aceeași formă ca și nsend, dar mesajul este reîncercat până la confirmare. Trei eșecuri la rând opresc sesiunea.

```
ncall("hit", target, dmg)
```

nlan

`nlan()`

IP-ul:portul LAN curent după nlisten.

```
print(nlan(), 2, 2, 7)
```

nstun

`nstun()`

IP-ul tău public:port de la STUN. Un oaspete din altă rețea poate încerca să se conecteze la ea.

```
pub = nstun()
```

setturn

`setturn(addr, [user, pass])`

Funcția TURN fallback atunci când o cale directă eșuează. setturn() cu argumente goale o dezactivează. Adresa este host:port.

```
setturn("turn.example:3478", user, pass)
```

nlist

`nlist(game)`

Tabel cu sesiunile deschise pentru numele jocului respectiv. Fiecare intrare are adresa, jucătorii și numele.

```
rooms = nlist("duel")
```

nannounce

`nannounce(game)`

Heartbeat, deci această gazdă apare în nlist pentru joc. Apelează-l cu un cronometru în timp ce găzduiește.

```
nannounce("duel")
```

nkb

`nkb([on])`

Câmp de adresă. nkb(1) îl deschide, nkb(0) îl ascunde, nkb() îți spune dacă este deschis. Tasta sus și jos schimbă un caracter, 0–9 sau un punct sau două puncte. A îl scrie. X șterge. Tastatura nu tastează în el.

```
nkb(1)
```

nkbstr

`nkbstr([s])`

Text în câmpul de adresă, apoi caracterele sus și jos se schimbă. nkbstr("") șterge textul.

```
addr, digit = nkbstr()
```

_mp

`_mp(from, name, ...)`

Tu scrii această funcție. Se execută atunci când sosește un mesaj. `from` este ID-ul expeditorului. `name` este evenimentul. Urmează până la patru valori.

```
function _mp(from, name, a, b)
  if name == "pos" then
    px[from] = a
    py[from] = b
  end
end
```

_mp_join

`_mp_join(p)`

Scrii această funcție. Un peer a primit id-ul p. Se declanșează și pentru tine; whoami() este acel id.

```
function _mp_join(p)
  add(log, p)
end
```

_mp_leave

`_mp_leave(p, why)`

Scrii această funcție. Peer p left. de ce 0 este o plecare normală. de ce 1 este o conexiune pierdută.

```
function _mp_leave(p, why)
  px[p] = nil
end
```

_mp_update

`_mp_update()`

Tu scrii această funcție. Se execută la fiecare 15 cadre de actualizare (_update frame), la aproximativ 4 Hz. Un loc bun pentru a trimite poziții prin nsend.

```
function _mp_update()
  nsend("pos", x, y)
end
```

_mp_fail

`_mp_fail()`

Scrii această funcție. Sesiunea este inactivă: gazda a dispărut sau apelul ncall a eșuat de trei ori.

```
function _mp_fail()
  online = false
end
```

Corutine

cocreate

`cocreate(f)`

Creează o coroutină din funcția f. Nu rulează până la coresume.

```
co = cocreate(cutscene)
```

coresume

`coresume(c, ...)`

Rulează corutina c până când returnează o valoare. În caz de succes: true și valorile generate. În caz de eroare: false și eroarea.

```
ok, a = coresume(co)
```

costatus

`costatus(c)`

„suspendat”, „alergând” sau „mort”.

```
if costatus(co) == "dead" then co = nil end
```

yield

`yield(...)`

Redă controlul celui care a apelat coresume. Valorile devin rezultatele acelei coresume.

```
yield(1)
```

Măsură

info

`info([x, y, col])`

Desenează HUD-ul live în _draw: fps (media de 1 secundă), upd, drw, px, max, ram, peak, ramc și net când o sesiune este activă. info() folosește cursorul de imprimare. info(col) setează doar culoarea. info(x, y) sau info(x, y, col) o plasează. Numerele folosesc B, k sau m pentru a se potrivi pe un ecran mic. Apelul în sine nu este numărat în px.

```
info(1, 1, 7)
```

showstats

`showstats([on])`

Același HUD ca și info, desenat automat la 0, 0 după _draw, ignorând camera și clipul. showstats(0) îl ascunde. showstats(1) sau showstats() îl afișează.

```
showstats(1)
```

perf

`perf(n)`

Un număr HUD, ca să-l poți desena singur. 0 fps, 1 actualizare, 2 desene, 3 px, 4 max (cel mai mare număr de cadre din secunda asta), 5 RAM, 6 RAM (RAM plus cartușul de joc încărcat), 7 RAM de vârf de la lansarea cartușului de joc.

```
print(perf(0), 2, 2, 7)
```

manualul PIX37. Kamil Stanisławski.