Manuál PIX37

Najprv editor, potom kód. Offline kópia knihy na pix37.party/docs.

Redaktor

PIX37 DEV je editor. Tu kreslíte, píšete a počúvate hru, potom exportujete hernú kazetu alebo samostatnú zostavu. Dokáže tiež prehrať hernú kazetu a otvoriť PIX37.PARTY na stiahnutie herných kaziet a používanie katalógu. PIX37 DEV EDUCATIONAL je rovnaký editor a je bezplatný. Nemá žiadne zostavy, nemá export herných kaziet a nemá prístup k PIX37.PARTY. Prehrávačom je PIX37 Console. Prehráva herné kazety `.p37` a otvoríte PIX37.PARTY na stiahnutie herných kaziet a prehliadanie katalógu.

Projekty a herné kazety

Súbor `.pix37` je projekt: zdrojové karty, hárky spritov, mapy, zvuk a dátové tabuľky. Súbor `.p37` je herná kazeta, ktorú hráč spúšťa: husto zabalený bajtkód.

Uložené súbory a úspechy sa nachádzajú v priečinku autora vedľa údajov aplikácie. Vo Windowse to vyzerá ako `C:\\Users\\Ada\\AppData\\Roaming\\PIX37\\Mira\\Pong.p37s` pre uložené súbory a `C:\\Users\\Ada\\AppData\\Roaming\\PIX37\\Mira\\Pong.ach` pre úspechy. V Linuxe sú rovnaké súbory `~/.local/share/PIX37/Mira/Pong.p37s` a `~/.local/share/PIX37/Mira/Pong.ach`. Ada je účet v počítači. Mira je meno autora a tento názov je priečinok. Pong je slot pre uloženie, ktorý je predvolene nastavený na názov hry. `save` zapíše `.p37s`. Prvý odomknutý úspech zapíše `.ach`. Ak sú súbory už nainštalované, bežec ich prečíta pri spustení hernej kazety.

Menu pre vývojárov

Stlačením klávesu Escape otvoríte ponuku vývojárov. Stlačením klávesu Escape počas behu hry sa vrátite do editora. Potvrďte stlačením klávesu Enter alebo tlačidla A (predvolene V). Späť je tlačidlo B (predvolene C).

Ak nie je načítaný žiadny projekt, ponuka je: VYTVORIŤ PROJEKT, NAČÍTAŤ PROJEKT, VYTVORIŤ, NAČÍTAŤ hernú kazetu, PRIPOJIŤ SA K PIX37.PARTY, NASTAVENIA.

Po načítaní projektu postupujte takto: ULOŽIŤ PROJEKT, NAČÍTAŤ PROJEKT, VYTVORIŤ PROJEKT, AKTÍVA, VYTVORIŤ, NAČÍTAŤ Hernú kazetu, PRIPOJIŤ SA K PIX37.PARTY, NASTAVENIA.

Ctrl+S uloží otvorený projekt. Ctrl+R ho spustí.

Vytvoriť projekt

CREATE PROJECT si pred začatím tvorby projektu vyžiada hernú kazetu.

- **VIDEO** — 1-BITOVÉ (2 farby), 2-BITOVÉ (4), 4-BITOVÉ (16), 8-BITOVÉ (256).
- **AUDIO** — hĺbka samplovania syntetizátora: 16, 8, 4, 2 alebo 1 bit.
- **OBRAZOVKA** – predvoľby zahŕňajú 64 × 64, 128 × 128, 160 × 90, 256 × 256 a 320 × 180. VLASTNÁ veľkosť vám umožňuje vybrať ľubovoľnú veľkosť od 1 × 1 do 512 × 512.
- **PALETY** — banky farieb. Prvá banka začína ako 16-farebná sada uvedená nižšie. NAČÍTAŤ PALETU použije vstavanú sadu: PICO-8 16, PICO-8 32, Game Boy, 1-bit, CGA, Sweetie-16 a dlhší zoznam prevzatý z Lospecu (Resurrect 64, Apollo, Endesga 32 a 64, CC-29, SLSO8, Pear36, Oil 6, Duel, AAP-64, Lost Century a ďalšie). Môžete tiež načítať súbor palety alebo načítať vlastný PNG a vybrať paletu z daného obrázka.



Farebné indexy používané `cls`, `pset`, `spr` a ostatnými. Ľubovoľný index môžete premaľovať v editore palety.

Päť panelov

Editor pozostáva z HRY, MAĽBY, KÓDU, ZVUKU a ÚDAJOV v tomto poradí. Mapa sa nachádza v programe MAĽBY. SFEXY a skladby sa nachádzajú v editore ZVUK. Kliknite na panel, ak chcete v ňom pracovať. Keď je fokus na KÓD, ZVUK a ÚDAJE ignorujú klávesy s písmenami, takže písanie zároveň nehrá na klavíri.



Úplné rozloženie, napísané `1:1:4:1:1`. Každé číslo predstavuje, akú časť riadku daný panel zaberá: HRA, MAĽBA, KÓD, ZVUK, DÁTA. Malá obrazovka je umiestnená v strede bunky s rozmermi 128 × 128 pixelov. Obrazovka väčšia ako 128 pixelov rozširuje stĺpec HRA.

NASTAVENIA → GLOBÁLNE → ROZLOŽENIE cyklicky mení usporiadania:

Kód	Čo vidíš
`1:1:4:1:1`	Všetko. HRA a ZVUK naukladané, MAĽBA a DÁTA naukladané, KÓD vpravo.
`1:1:4:1:1 2X`	Rovnaké usporiadanie. KÓD je vykreslený s dvojnásobnou hustotou, takže sa zmestí dvakrát toľko textu.
`1:0:1:0:1`	HRA, DÁTA, KÓD.
`1:0:1:0:1 2X`	Rovnaké usporiadanie s hustejším KÓDOM.
`0:1:1:1:1`	Žiadna ukážka hry. MAĽBA, KÓD, ZVUK, DÁTA.
`0:1:1:1:1 2X`	Rovnaké usporiadanie s hustejším KÓDOM.
`0:1:4:1:2`	Štúdio. Široký KÓD, vyšší DÁTOVÝ pás, MAĽBA a ZVUK.
`0:1:4:1:2 2X`	Rovnaké usporiadanie s hustejším KÓDOM.
`0:0:4:0:4`	Iba KÓD a ÚDAJE, obe veľké.
`0:0:4:0:4 2X`	Rovnaké usporiadanie s hustejším KÓDOM.
`0:1:4:1:0`	FARBENIE, ZVUK a široký KÓD.
`0:1:4:1:0 2X`	Rovnaké usporiadanie s hustejším KÓDOM.
`0:4:0:0:0`	Iba MAĽBA, 256 × 256, jeden pixel hárku na jeden pixel plátna.

Riadok `2X` zachováva rovnaké panely. Panel CODE sa vykresľuje s dvojnásobným rozlíšením a zmenšuje sa späť do rovnakej bunky, takže text, tabulátory a medzera sú polovične vysoké a na obrazovku sa zmestí dvakrát toľko zdrojového kódu.

HRA

Náhľad spustenej hernej kazety. Kliknite na panel HRA alebo stlačte Ctrl+R pre spustenie tohto náhľadu. Panel sa konfiguruje v časti NASTAVENIA → HERNÝ PANEL:

- **Náhľad** – farba v nečinnosti, nepretržitý alebo automatické opätovné načítanie pri úprave.
- **Hrať v novom okne** – hra sa otvorí vedľa editora.
- **Displej** – náhľad ako RGB888 alebo RGB565.
- **HRA 64** – obrazovka s rozmermi 64 × 64 je buď vycentrovaná v pomere 1:1 v bunke, alebo je zmenšená 2× tak, aby ju vyplnila.
- **Shadery** — CRT a filtre v hre, oddelené od filtrov v editore.
- **Farba** — pozadie panela.

FARBA

Hárky spritov a na strane mapy v tom istom paneli dlaždicové mapy. Hárky môžu byť väčšie ako obrazovka. Veľkosti náhľadov spritov sú 8, 16, 32, 64 a 128 (128 iba vtedy, keď má hárok aspoň 128 px). Posúvaním kolieska myši cez náhľad spritov ho priblížite. Veľkosti štetcov sú 1, 2, 3 a 4.

Kľúč	Nástroj
B.	Ceruzka
M	Výber obdĺžnika
Otázka	Laso
S	Pečiatka
G	Vedro
Ja	Kvapkadlo
V.	Presunúť
L / Shift+L	Čiara / krivka
U / Shift+U	Dutý obdĺžnik / dutá elipsa
F / Shift+F	Vyplnený obdĺžnik / vyplnená elipsa
D	Rozptyl
[ ]	Menší/väčší štetec
Ctrl+Z / Ctrl+Y	Späť / Znovu. Ctrl+Shift+Z tiež zopakuje.
Ctrl+C / Ctrl+X / Ctrl+V	Kopírovať, vystrihnúť, vložiť
Ctrl+A	Vyberte plátno

Mapa využíva rovnaký princíp s menším počtom nástrojov: B ceruzka, M výber obdĺžnika, S pečiatka, V pohyb a Ctrl+Z, Ctrl+Y, Ctrl+C, Ctrl+V. Podržaním klávesu Alt priblížite textúru mapy. Veľkosť štetca na mape zostáva zachovaná pri nástrojoch na kreslenie.

Každý sprite má osem príznakových bitov. Paint ich nastaví na sprite; `fget` a `fset` ich načítajú v hre.

NASTAVENIA → EDITOR KRÁSENIA je miesto, kde sa tento panel konfiguruje:

- **Paleta** — ktorá banka je nakreslená, plus načítanie a uloženie.
- **Náhľad** — RGB888 alebo RGB565.
- **Kurzor štetca** – ukazovateľ pri maľovaní.
- **Prichytenie pixelov** – spôsob prichytenia štetca k pixelom.
- **Veľký štetec** – veľký štetec sa odtrhne z ľavého horného rohu alebo z celej svojej plochy.
- **Cibuľová šupka** – duch iných rámov a smer, ktorým sa tento duch uberá.
- **Priblíženie mapy** – skutočné pixely alebo bezplatné priblíženie.
- **Farby** — farby používateľského rozhrania pre kreslenie.

KÓD

Karty sú samostatné súbory v rámci projektu. Kliknutím na názov karty ju premenujete; kláves Enter potvrdí. Niekoľko kariet, ktoré definujú `_init`, `_update` alebo `_draw`, sa spojí v poradí podľa kariet: prvá karta sa spustí ako prvá, posledná karta sa spustí ako posledná. Kód, ktorý nie je vo vnútri hooku, sa stáva súčasťou `_init`. Herná kazeta bez akýchkoľvek hookov sa považuje za jeden `_draw`.

Kľúč	Akcia
Šípky	Presunúť kurzor. Stlačením klávesu Shift rozšíriť výber.
Ctrl+Vľavo / Vpravo	Slovom.
Domov / Koniec, Ctrl+Domov / Koniec	Začiatok alebo koniec riadku, prípadne tabulátora.
Záložka	Vložte dve medzery. V zozname dokončovania prijmite meno.
Zadajte	Nový riadok, odsadený ako predchádzajúci. Prijíma doplnenie.
Ctrl+C / X / V	Kopírovať, vystrihnúť, vložiť.
Ctrl+Z / Ctrl+Y	Vrátiť späť / znova.
Ctrl+A	Vyberte kartu.
Ctrl+D	Duplikujte čiaru alebo výber smerom nadol.
Ctrl+/	Komentovať alebo odkomentovať môžete pomocou `--`.

Ctrl+F	Nájsť. F3 preskočí na ďalší zásah. Escape zatvorí lištu.
Ctrl+G	Prejsť na číslo riadku.
Útek	Najprv zatvorí vyhľadávanie, prechod na alebo zoznam rád. Druhý kláves Escape opustí editor a otvorí ponuku pre vývojárov.

Pri zadávaní názvu funkcie sa zobrazujú rady. Stlačením klávesu Enter alebo Tab sa vloží názov a `(`, ak je v nastaveniach zapnutá možnosť COMPLETE (). NASTAVENIA → EDITOR KÓDU:

- **Hľadať** – Ctrl+F na aktívnej karte alebo na každej karte.
- **Vysoký text** – orezanie glyfov na 5 px alebo zobrazenie celého glyfu.
- **Veľkosť reťazca** — veľkosť písma reťazcových literálov.
- **Písmo komentára** – 3×5 alebo písmo Fusion aktuálneho jazyka používateľského rozhrania.
- **Dokončiť (** — či dokončenie typuje zátvorku.
- **Tipy pre kód** – zapnuté alebo vypnuté.
- **Jazyk nápovedy** – jazyk ponuky alebo angličtina.
- **Medzera vsuvky** — dodatočný posun na prázdnych riadkoch.
- **Odkvap** – čísla riadkov.
- **Farba** — pozadie, komentáre, tabulátory, čísla, operátory, kľúčové slová, reťazce, funkcie nástroja, vaše funkcie, premenné, zátvorky, vsuvka, výber, chybový riadok, upozornenie.

ZVUK

Panel má dve karty. SFX je jeden vzor: krátky sled nôt. SOUND je skladba: umiestňuje tieto vzory na osem kanálov. TIC udáva, koľko tikov jeden krok čaká. Takt má frekvenciu 120 tikov za sekundu a jeden krok je šestnástina, takže 15 tikov je 120 úderov za minútu. Vyšší TIC je pomalší vzor. Dva vzory zostávajú v kroku, keď jeden TIC delí druhý: 15 a 30 áno, 15 a 8 nie.

Klavír hrá, keď je nota vybraná myšou. Biele klávesy sú ZXCVBNM (CDEFGAB). Čierne klávesy sú SDGHJ (C# D# F# G# A#). Podržte kurzor myši nad ľubovoľnou hodnotou a zadajte číslo alebo nad ňou podržte kurzor myši a rolujte kolieskom. Čísla výšok tónov v kóde sa pohybujú od 0 (C-0) do 63 (D#5).

V jednom patterne sú LOOP a LEN rovnaké dve čísla. OUT nad 0 sa slučí od IN do OUT a názov znie LOOP. OUT pri 0 a IN nad 0 je LEN: toľko nôt sa hrá od začiatku, potom sa pattern končí a názov znie LEN. Obe pri 0 sa hrajú každý krok.

Koľko syntetizátora môžete upravovať, závisí od bitovej hĺbky zvolenej pre hernú kazetu. NASTAVENIA → EDITOR SFX nastavuje túto hĺbku, či sa nota automaticky posúva vpred (vypnuté, reťazenie nôt alebo reťazenie polí), či Space reštartuje prehrávanie alebo prepína medzi spustením a zastavením a farby editora.

16-bitový

Každý krok má notu, hlasitosť, vlnu a efekt. Hlasitosť je od 0 do 7. 0 je ticho.

Vlny:

- **0 trojuholník**
- **1 naklonená píla**
- **2 píly**
- **3 štvorce**
- **4 pulzy** – úzky pulz
- **5 organov**
- **6 hluk**
- **7-fázorový**
- **8 PWM** — štvorec, ktorého šírka sa pohybuje
- **9 pulzov 25 %**
- **10 sínus**
- **11 kovový zvuk**
- **12 FM zvon**
- **13 trhať**
- **14 superpíl**
- **15 PWM (C64)**

Efekty na jednu notu:

- **0 žiadny**
- **1 snímok** — výška tónu a hlasitosť sa v tomto kroku kĺžu k ďalšej note
- **2 vibrato** — výškové kolísanie
- **3. pokles** — výška tónu klesá do ticha, obvyklý kop
- **4 zoslabenie**
- **5 zoslabovania**
- **6 arpeggio** — prechádza túto notu a ďalšie tri, jedno každých 8 tikov. Pri TIC 8 alebo menej, každé 4 tiky.
- **7 rýchle arpeggio** — rovnaké štyri noty, každé 4 tiky alebo každé 2, keď je TIC 8 alebo menej.

Hlavička vzoru, nad poznámkami:

- **TIC** – počet tikov na krok, od 1 do 255. 15 je 120 úderov za minútu. Zadajte číslo alebo ho zmeňte o jeden krok.
- **LOOP IN / OUT** — s OUT nad 0 sa vzor slučkuje od IN do OUT. Ak necháte OUT na 0, názov sa zmení na LEN: IN označuje počet prehraných nôt a 0 znamená celý vzor.
- **ŠUM** – celý vzor sa stáva šumom.
- **CRUSH** — bzučivý variant každej vlny.
- **DTUNE** — 0 vypnuté, 1 ľahký refrén, 2 silný refrén.
- **RVERB** — dozvuk: 0 vypnutý, 1 krátky, 2 dlhý.

- **DAMPN** — dolnopriepustný filter: 0 vypnutý, 1 slabý, 2 silný.
- **LEVEL** — hlasitosť celého vzoru, 0 až 7.
- **SWING** — miešanie. Nepárne kroky čakajú o niečo dlhšie, párne kroky o niečo kratšie.
- **KROKY** – počet krokov, ktoré vzor používa: 32, 24, 16 alebo 12.

8-bitový

Krok má notu a vlnu. Štyri vlny sú štvorcová, trojuholníková, pílová a šumová. Hlavička má TIC, LOOP IN / OUT, FX a LEVEL. TIC je jedna z možností 1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 30, 40, 60. FX je jeden efekt pre celý pattern: 0 žiadny, 1 vibrato, 2 arpeggio, 3 fade out. LEVEL je 0 až 3.

4-bitový

Krok je iba nota. Hlavička má TIC, LOOP (zapnuté alebo vypnuté), WAVE (tie isté štyri vlny) a OCT (v ktorej oktáve sa noty nachádzajú). TIC je jedna z možností 4, 6, 8, 10, 12, 15, 20, 30.

2-bitový

Krok je prázdny, bubon, hit alebo rozdrvenie. Hlavička má TIC a BASE. TIC je 8, 15, 16 alebo 30. BASE určuje, ako vysoko sa nachádza bubon a hit.

1-bit

Krok je nota. TIC je 8, 15, 16 alebo 30, rovnaké štyri rýchlosti ako 2-bit.

Piesne

Karta ZVUK predstavuje otvorenú skladbu. Osem stĺpcov predstavuje kanály. Bunka predstavuje ID vzoru. Číslo 255, nakreslené prázdne, znamená, že kanál je v danom riadku tichý. Stĺpec X predstavuje príkaz: pokračovať, zastaviť, začiatok slučky alebo koniec slučky. Zastavenie ukončí skladbu po riadku. Koniec slučky preskočí späť na posledný začiatok slučky.

Riadok končí, keď sa skončí ľavý vzor s LOOP OUT na 0. Vzory ďalej vpravo tento riadok nepredlžujú: dlhší vzor sa skráti, kratší vzor sa začína odznova od svojho prvého kroku, až kým sa riadok neskončí. Vzor s LOOP OUT nad 0 sa slučí medzi LOOP IN a LOOP OUT a riadok nekončí. Ak sa každý vzor v riadku slučí, riadok tam zostane.

Skratky

Kľúč	Akcia
Klávesy klavíra	Napíšte poznámku ku kroku vybranému myšou.
Hore / Dole	Zmeňte hodnotu pod kurzorom.

Priestor	Prehrajte otvorený SFX alebo otvorenú skladbu z prvého riadku. Spustenie/zastavenie alebo reštartovanie v nastaveniach SFX EDITORA.
Ctrl+Z / Ctrl+Y	Vrátiť späť / znova.
Ctrl+C / Ctrl+X / Ctrl+V	Skopírujte, vystrihnite, vložte výber.

ÚDAJE

Pomenované tabuľky. Každá tabuľka pozostáva z riadkov a stĺpcov. Z kódu tieto tabuľky používajú `dget`, `dset`, `dstr`, `drows` a `dcols`. Čísla riadkov a stĺpcov v kóde začínajú na 0.

Typy stĺpcov, cyklicky zobrazované v hlavičke:

- **bool** — jeden bit.
- **2** — dva bity.
- **4** — štyri bity.
- **8** — bajt.
- **16** — 16 bitov, pre ID a väčšie čísla.
- **spr** — ID spritu.
- **str** — text, vykreslený systémovým písmom.
- **str8** a **str10** — rovnaký text, nakreslený vyššie.
- **f16** a **f32** – polovičný plavák s plnou presnosťou a plný plavák.

Keď je kurzor na hlavičke stĺpca, stlačením klávesu Ctrl+posúvanie sa zmení typ daného stĺpca. Obyčajné posúvanie sa pohybuje medzi stĺpcami alebo riadkami nadol, podľa toho, čo vyberiete v časti NASTAVENIA → EDITOR ÚDAJOV. Stlačením klávesu Ctrl sa tento smer zmení, a preto sa stlačením klávesu Ctrl+posúvanie na hlavičke zmení typ, keď je koliesko nastavené na stĺpce.

Šípky slúžia na pohyb. Stlačením klávesu Enter sa upravuje bunka. NASTAVENIA → EDITOR ÚDAJOV:

- **Koliesko** – posúvanie stĺpcov alebo riadkov.
- **Farby** – mriežka, hlavička a výber.

Aktíva

Funkcia ASSETS importuje a exportuje otvorenú časť, nie celý projekt. Sprity používajú sprity otvorené v programe PAINT na otvorenom hárku. Mapa je otvorená mapa. Zvukové efekty sú otvorené vzory. Hudba je otvorená skladba. Dáta sú otvorené tabuľky. Výnimkou je kód: každý tabulátor sa zapíše a každý tabulátor sa spätne načíta.

Druh	Súbor	Čo sa hýbe
Škriatkovia	`.png`	Otvorený sprite.
Mapa	`.p37map`	Otvorená mapa.
Zvučné efekty	`.p37sfx`	Otvorený vzor.
Hudba	`.p37mus`	Otvorená skladba. Čísla patternov zostávajú nezmenené. Ak potrebujete tieto zvuky, exportujte SFX samostatne.
Dáta	`.p37dat`	Otvorený stôl.
Kód	`.lua`	Každá záložka.

Nastavenia

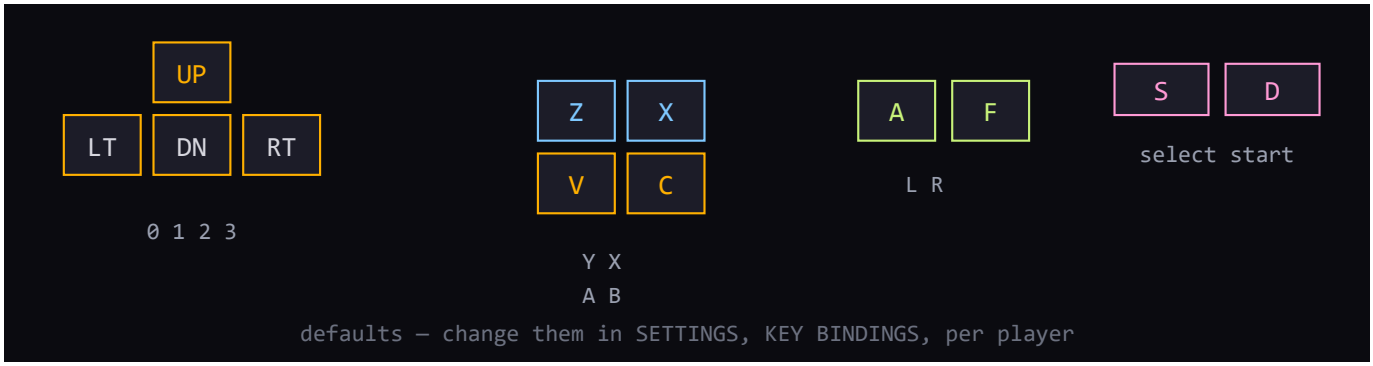
Koreňový zoznam:

Strana	Čo uchováva
PROFILY	Uložte a načítajte každé nastavenie. Nové, premenujte, prepíšte, vymažte. Skopírujte alebo vložte profil ako text.
Menu	Farby okien, klávesov, ponuky, voliteľné pozadie.
HERNÝ PANEL	Okno ukážky a prehrávania.
EDITOR KÓDU	Zvýraznenie a rady.
EDITOR FARBÍN	Štetec, paleta, zobrazenie mapy.
EDITOR ZNAČKOVÝCH EFEKTOV	Syntetizátor a zadávanie nôt.
EDITOR ÚDAJOV	Smer a farby kolies.
JAZYK	Jazyk ponuky a nápovedy.
LICENCIE	Texty licencií vrátane písma Fusion Pixel a SDL3.

GLOBÁLNE:

- **ROZLOŽENIE** — usporiadanie panelov, vyššie.
- **HLASITOSŤ** — hlavná úroveň, 0 až 10.
- **VÄZBY KLÁVESOV** — klávesy pre každého z 8 hráčov. Stlačením klávesu ho zachytíte. Pokyny, ABXYLR, výber.

- **Prezentovať** – preskočiť nečinné snímky alebo vykresliť každú snímku.
- **Celá obrazovka** – bez okrajov.
- **Obnoviť** – FPS okna. Krok vľavo a vpravo 60, 90, 120 a max. Enter zadá číslo do 1000. 0 znamená bez obmedzenia.
- **Vsync** — čaká na zobrazenie medzi snímkami.
- **Mierka** — celočíselný násobiteľ okna.
- **Shadery** — CRT a filtre v editore, nie v hre.
- **Farba** – text ponuky a zvýraznenie.
- **Pozadie** – voliteľný PNG súbor orezaný na 128 pixelov za ponukou. Kliknutím na tlačidlo Vymazať ho odstránite.
- **Kurzor editora** – tvar ukazovateľa.
- **Myš** – plávajúci ukazovateľ alebo prichytenie k pixelom.
- **Rýchlosť používateľského rozhrania** – ako rýchlo sa podržaný smer opakuje v zozname.
- **Prieskumník** – zoznamy súborov začínajú v koreňovom adresári počítača alebo v priečinku exe.



Predvolená klávesnica. A je potvrdenie v menu. B je späť. V hre sú to tlačidlá 4 a 5. Tlačidlo výberu (S) je tlačidlo 10, nie skratka menu.

Tlačidlo	Úloha	Predvolený kľúč
0	Vľavo	Šípka doľava
1	Vpravo	Šípka doprava
2	Hore	Šípka hore
3	Dole	Šípka nadol
4	A, potvrdiť	V.
5	B, späť	C
6	X	X
7	Y	Z

8	R	F
9	L	A
10	Vyberte	S
11	Štart	D

Zostaviť

Vyberte si platformu: Windows, Linux, Mac, Android, iOS alebo web.

- **Názov** — názov súboru so zostavením.
- **Autor** – v systémoch Windows a Linux. Toto je názov priečinka pre uložené súbory.
- **ID aplikácie Steam** – vo Windowse a Linuxe. Prázdne pole znamená, že Steamworks nie je k dispozícii. Vyplnené ID mapuje `achieve(0)`...`achieve(15)` na úspechy v službe Steam ACH_0...ACH_15.
- **Id** — identifikátor balíka, ak ho platforma požaduje.
- **Skryť kód** – na prekryvných vrstvách na počítači a mobile. Spomaľuje hernú kazetu `peek`. Nie je to zámok.
- **Celá obrazovka** – exportovaný prehrávač sa spustí v režime celej obrazovky.
- **Shadery** — až tri vrstvy zabudované do prehrávača.
- **Stiahnuť .p37** – Len pre web. Zobrazí položku na stiahnutie v ponuke pozastavenia.
- **Ikona** — dlaždica zo sprite hárka: hárok, veľkosť, x, y. Alebo PNG obal, ktorý orežete.
- **Exportovať** – vyberte priečinok. V ponuke sa zobrazí správa VYTVORENIE ULOŽENÉ alebo VYTVORENIE ZLYHALO.

Vytvorte zostavu s pripojeným internetom. Úvod sa môže anonymne opýtať servera, či je známy hash tejto zostavy. Táto kontrola je tam z dvoch dôvodov. Chráni mňa, autora PIX37, pred kópiami nástroja, ktoré neboli moje, aby som ich mohol rozdať. Chráni tiež osobu, ktorá hru vytvorila. Prevzatie hry niekoho iného a jej uvedenie ako vlastnej sa stáva bežným na trhu s telefónmi, ako je Play (Android) alebo iOS. Hra sa stále spustí. Ak je počítač offline, úvod hovorí, že hra nie je pripojená. Tento riadok je tiež náznakom pre hráča, že kópia nemusela pochádzať z legálneho zdroja. Je to malá kontrola, takže ak vaša hra bola ukradnutá a predaná bez vášho súhlasu niekým iným, môžete to preukázať v obchode, ako je Play (Android), Steam alebo itch.io. Server ukladá čas, kedy bol hash prvýkrát zaregistrovaný, vedľa hashu a neskôr ho nemení. Skorší čas je pôvodná zostava. Hash zaregistrovaný po tomto čase je neskoršia kópia.

Chcem, aby táto kontrola ochránila ľudí, ktorí tvoria hry. Engine už teraz robí veľa práce za nich a toto je jedna z tých vecí. Chápem, že to môže byť kontroverzné. Ak sa komunita rozhodne, že je lepšie zrušiť toto mikro-DRM, urobím to.

Stará herná kazeta z predchádzajúceho editora sa môže po zmene jazyka odmietnuť spustiť. Otvorte projekt `.pix37` v tomto editore a exportujte nový súbor `.p37`.

Pripojte sa k PIX37.PARTY

PRIPOJENIE K PIX37.PARTY je v aplikácii. V prehliadači nie je žiadna prihlasovacia stránka. Prihlasujete sa pomocou lístka na Steam alebo cez itch.io pomocou klávesnice a myši. Služba zobrazuje jedinečné zobrazované meno s dĺžkou 2 až 16 znakov. Licenčný kľúč a Steam ID sa ostatným hráčom nezobrazujú.

Po podmienkach služby sa centrum nachádza v častiach Obľúbené, Katalóg, Poradie, Hráči a váš profil.

Vývojárska licencia je editor. Vytvoríte hru, nahrajete ju a necháte hviezdičky. Hráčska licencia je hráč. Srdiečka sú od ľudí hrajúcich na bežeckej dráhe.

- **Katalóg** – prehliadanie hier. Filtre zahŕňajú žáner (maximálne tri), veľkosť obrazovky, veľkosť hernej kazety a to, či nahraná hra obsahuje projekt `.pix37`.
- **Obľúbené** – hry, ktoré ste označili.
- **Nahrať** — vyžaduje licenciu vývojára. Odošlete `.p37` a môžete priložiť projekt `.pix37`. `.p37` je zabalený bajtkód. Je ťažko čitateľný a môže byť zámerne znejasnený. `.pix37` je súbor projektu, s ktorým skutočne pracujete: zdrojový kód, obrázky a zvuk vo forme, ktorú môže otvoriť a učiť sa z nej iná osoba. Jeho odoslaním si môžu ostatní tvorcovia hru preštudovať namiesto komprimovanej hernej kazety. Súčasťou je názov, autor, žánre, obal, ikona a názvy úspechov. Odoslané číslo RAM predstavuje vrchol, ktorý herná kazeta dosiahla.
- **Komentáre a úspechy** – na stránke hry. Odomknutie úspechu, keď ste prihlásení, sa uloží na daný účet, ako aj na zariadenie.
- **Poradie** – vývojári zoradení podľa hviezdičiek alebo srdiečok. Každý riadok zobrazuje oba počty. Otvorte meno a zobrazte hry daného autora zoradené podľa srdiečok.
- **Hráči** — účty zoradené podľa odomknutých úspechov.

Kód

Jazyk je malý skript v štýle Lua. Táto polovica začína s prázdnym projektom, učí jazyk a potom ukazuje krátky príklad každej funkcie.

Prvý program

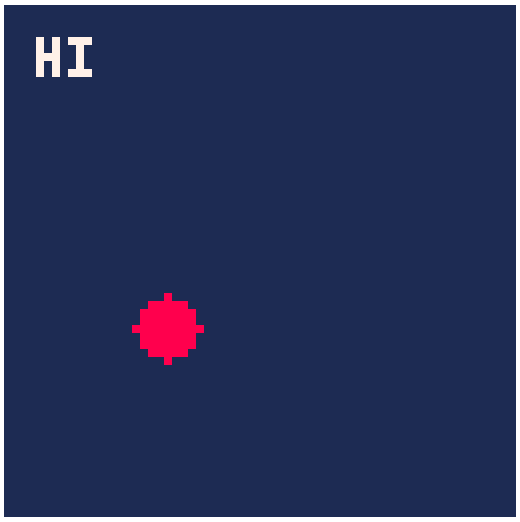
Vytvorte projekt s rozlíšením 64×64 a 4-bitovou farbou. Otvorte KÓD. Nahraďte tabulátor takto:

```
_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 ju spustí. Šípky posúvajú kruh. Escape hru zastaví a vráti do editora.



Tie isté tri volania: `cls(1)` vymaľuje námornícku obrazovku, `print` nakreslí HI a `circfill` nakreslí ružový kruh.

Čo beží a kedy

- `_init` sa spustí raz.
- `_update` beží na 60 Hz. Sem umiestnite pohyb a tlačidlá.
- `_draw` kreslí. Bez argumentu fps kreslí tak rýchlo, ako to počítač a monitor dokážu. `_draw(30)` obmedzuje kresbu na 30 fps a logiku ponecháva na 60.
- `_set_fps(n)` tento limit neskôr zmení. Hodnota 0 ho vynuluje.

Názvy nerozlišujú veľké a malé písmená; editor ich ukladá malými písmenami. `function` a `fun` sú to isté kľúčové slovo.

Jazyk

Základy programovania

Program je zoznam riadkov, ktoré herná kazeta spúšťa. Každý nadpis najprv zobrazuje tvar riadku a potom malý príklad. Riadok, ktorý začína `--`, je komentár: poznámka pre vás. Hra ho preskočí.

Premenné

Premenná je názov, ktorý obsahuje hodnotu. Názov si vyberiete. Neskoršie riadky ho môžu prečítať a pod rovnakým názvom môžu uložiť novú hodnotu.

Tvar je `name = value`. `local name = value` ponecháva tento názov v aktuálnom bloku, takže ho zvyšok hernej kazety nevidí. Bez `local` je názov globálny a každý panel ho vidí.

```
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` znamená, že neexistuje žiadna hodnota. `true` a `false` sú áno a nie. Názov, ktorý ste nikdy nenastavili, sa číta ako `nil`.

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

Tabuľky

Tabuľka obsahuje veľa hodnôt pod jedným názvom. Zoznam používa čísla. Prvá položka je 1, nie 0. Záznam používa názvy. Tabuľka môže obsahovať ďalšiu tabuľku.

Tvar zoznamu: `{ value, value }`. Tvar záznamu: `{ name = value, name = value }`. Zoznam záznamov je tabuľka, ktorej položky sú tabuľky.

```
-- 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
```

Stav a telo

Podmienka je otázka s odpoveďou áno alebo nie, napríklad `hp > 0` alebo `btnd(4)`. Telo je skupina riadkov, ktoré sa spúšťajú pri vstupe do bloku. `if` a každá slučka používa rovnaký tvar: podmienka, potom telo a potom `end`. `repeat` sa uzatvára s `until` namiesto `end`.

Ak

`if` spustí svoje telo iba vtedy, keď je podmienka pravdivá. `elseif` skúša inú podmienku. `else` spustí, keď sú všetky podmienky nepravdivé.

Tvar: `if condition then`, potom telo a nakoniec `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
```

Slučky

Slučka opakuje telo cyklu. `while` najprv skontroluje podmienku a pokračuje, kým je pravdivá. `repeat` spustí telo cyklu a potom skontroluje, takže telo cyklu sa vždy spustí aspoň raz. `for` počíta. Posledné číslo je zahrnuté. Bez tretieho čísla je krok 1, takže ak je začiatok už za koncom, cyklus sa nespustí. Odpočítavanie smerom nadol vyžaduje záporný krok.

```
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
```

Chôdza okolo stola

`all` navštívi každú položku zoznamu od prvej. `pairs` navštívi každý názov a hodnotu uloženú pod týmto názvom.

```
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
```

Funkcie

Funkcia je pomenovaný recept. Tvar je `function name(arguments)`, potom telo a nakoniec `end`. Zavoláte ju napísaním názvu a zátvoriek. `return` pošle späť hodnotu a môže ich poslať viac. Argumenty sú lokálne hodnoty danej funkcie.

```
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` je funkcia, ktorá vidí lokálne hodnoty okolo seba. Ide o uzáver. Tabuľka môže obsahovať jednu hodnotu a voláme ju s bodkou.

```
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
```

Áno, nie a výber

`not` zmení podmienku. `and` je pravdivý iba vtedy, keď sú pravdivé obe strany. `or` je pravdivý, keď je pravdivá ktorákoľvek strana. `a and b or c` vyberie `b`, keď je pravdivý `a`, inak `c`. `cond ? a : b` je tá istá myšlienka v jednom riadku: `a`, keď je podmienka pravdivá, `b`, keď nie je.

```
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)
```

Matematika a text

`+ - * /` je sčítanie, odčítanie, násobenie a delenie. `//` delí a ponecháva celé číslo. `%` je zvyšok. `^` je umocnenie. `..` spája dva texty. `#t` je počet položiek v zozname. Porovnania sú `==` (rovnaké), `~=` alebo `!=` (rôzne) a `< > <= >=`. `x += 1` pripočíta 1 k x. Rovnaký tvar funguje aj pre `-= *= /= ^= %=`.

```
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
```

Kde býva miestny obyvateľ

`local` trvá, kým `end` neuzavrie svoj blok. `if` / `end` je jeho vlastný blok. Môžete uviesť lokálnu premennú nad riadkom, ktorá ho nastavuje, ale kým tento riadok nezačne fungovať, hodnota je `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
```

Skákanie

`::label::` označuje riadok. `goto label` tam skočí, ale iba v rámci tej istej funkcie.

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

Pamäť, skrátka

`peek` a `poke` sa pozerajú na živé pixely prostredníctvom jednej 16-bitovej adresy. Neuchovávajú druhú kópiu hernej kazety. Adresy `0x5F00` až `0x5FFF` sú registre: kamera, klip (iba na čítanie), farba kreslenia, vzor výplne, tri premapovania paliet a čísla stránok. Bajty na oboch stranách tejto diery, z ktorých je 65280, tvoria okno pixelu.

Keď sa obrazovka zmestí do týchto 65280 bajtov, 64×64 a 128×128 áno a 256×256 nie, obrazovka začína na adrese 0, riadok po riadku. Zostane len hárok spritov vybraný pomocou `0x5F41` (0 je hárok 0). Ak je tento hárok väčší ako zostávajúce miesto, `0x5F42` vyberie stránku a mapa sa v okne nenachádza. Ak sa hárok zmestí, na konci sú bunky mapy z mapy vybranej pomocou `0x5F43`, jeden alebo dva bajty na každý, a `0x5F44` stránkuje mapu, keď sa na konci nezmestí.

Keď sa obrazovka nezmestí, celé okno je jedna stránka s 65280 pixelmi. `poke(0x5F40, n)` otáča stránku obrazovky. `poke(0x5F41, n)` s n iným ako 0 zobrazuje namiesto toho hárok n a hárok 0 v tomto režime nemá žiadne zobrazenie. `poke(0x5F43, n)` s n iným ako 0 zobrazuje mapu n, ktorá má prednosť pred hárkom. Mapa 0 v tomto režime zobrazuje iba `mget` a `mset`.

`peek`, ktorý sa nedostane na pixel alebo register, vráti 0. `poke` v tomto okne neurobí nič. Príznaky Sprite, SFX a tabuľky DATA sa v tomto okne nenachádzajú. Použite `fget`, `sfxget` a `dget`.

Praktické využitie

Každá funkcia je uvedená v dolnej časti tejto stránky s praktickým využitím funkcií PIX37. Odkazy pod každým príkladom otvoria viac informácií o funkciách.



Plynulý pohyb

Herná kazeta s rozmermi 128 × 128, pohľad zhora. Sprite 1 je už nakreslený. Šípky zrýchľujú v ľubovoľnom smere a trenie vás spomaľuje. Sprite sa zalamuje: polovica, ktorá opúšťa spodok, je nakreslená hore a to isté platí pre ostatné hrany.

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



Kontrola, či je ďalší pohyb zablokovaný

Herná kazeta s rozmermi 64×64. Podlaha a pravá stena sú už na mape, namaľované spritom 3 a na tomto sprite je nastavený príznak 0. Šípky posúvajú sprite 1. Slovo hore hovorí „blokované“, keď by ďalší krok prekrýval plnú dlaždicu, a „voľné“, keď by sa neprekrývala.

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



Utierka s otočením

Stlačte A. Vymazanie sa vykreslí pomocou funkcie flip(), jeden čierny riadok po druhom, namiesto toho, aby sa to stalo v rámci funkcie _draw(). Funkcia _draw počas jej behu neprekáža, takže každý riadok zostane na obrazovke.

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



Tabuľka funkcií (trieda)

Sprite 1 je už nakreslený. make_monster funguje ako trieda. Vracia tabuľku a update a draw sú metódy, ktoré uchovávajú vlastné x a y daného monštra. Číslo hore udáva, akú vzdialenosť prešlo.

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



Štyri spôsoby, ako prečítať tlačidlo

A je tlačidlo 4. Podržte ho a náboj stúpa. Uvoľnite ho a náboj sa vráti na 0. Počas podržania opakujte pohyb hore a dole, takže riadok ponuky sa neustále pohybuje. btnd je prvá snímka stlačenia, ktorú táto herná kazeta používa iba ako začiatok tohto podržania.

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



Strieľajte myšou

Sprite 1 je už nakreslený v strede. Biela bodka je kurzor. Kliknutie ľavým tlačidlom myši vystrelí pixel smerom k nej. Koliesko zmení rýchlosť, vytlačené v rohu.

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



Zadajte meno a potvrďte ho

Stlačením tlačidla A otvoríte pole. Stlačením tlačidla hore a dole meníte blikajúce písmeno. A ho napíše, X maže, Y je medzera a B potvrdí. Riadok potom prečíta vaše meno.

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



Kamera na prehrávači

Herná kazeta s rozmermi 128×128. Sprite 1 začína uprostred podlahy z dlaždíc. Šípky sa pohybujú. Kamera udržiava sprite vycentrovaný. Osi x a y v rohu sa neposúvajú, pretože camera() bez argumentov vráti pohľad späť pre HUD.

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



Pár riadkov vizuálneho románu

Riadky sú v jazyku DATA, v angličtine a poľštine. A pokračuje alebo vyberá možnosť zostane, keď je na obrazovke zobrazená možnosť. B vyberie možnosť odísť a inak zmení jazyk. X zmení jazyk, keď je zobrazená možnosť. Dialóg používa písmo veľkosti 8px, takže sa vykresľujú poľské písmená.

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



Maľovanie metódou nakukovania a pokukovania

Herná kazeta s rozmermi 64 × 64, takže adresa 0 je pixel vľavo hore a ďalší riadok začína na čísle 64. Herná kazeta zapisuje pixely, kopíruje riadok, vymaže riadok a vytlačí hodnoty, ktoré prečítala.

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



Zoznam, do ktorého môžete pridať

A sčíta číslo, až do 8. B odstráni posledné. X odstráni prvé. Zoznam sa vypíše na obrazovke.

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



Rozobraté meno

Herná kazeta rozdelí ada,12,blue a vypíše názov, jeho prvé písmeno, 12, a slovo blue.

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



Zahrajte notu a potom ju zmeňte

Vzor 0 má dĺžku jednej noty a každé opakovanie má rovnakú hlasitosť. A ho zahrá znova, keď daná nota doznie. B zvýši výšku tónu a zahrá danú notu.

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

 Pieseň napísaná do hernej kazety

Dva patterny sa slučkujú v riadku 0, jeden na každom z prvých dvoch kanálov. Riadok zostáva 0. ch0 a ch1 sú noty, na ktorých je každý kanál. Úprava patternu, LEN a LOOP v hernej kazete je spôsob, ako vidieť, čo tieto ovládacie prvky robia v hudobnom paneli.

```
songget songset music musicstat cls print _init _draw
```

 Tri príšery v DATA

Panel DATA už má stĺpec s názvom a stĺpec s výdržou, so slizom, netopierom a okom. A slizovi ubližuje. Čísla vpravo predstavujú jeho výdrž.

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

 Zapamätaj si skóre

A pridá bod a uloží ho. Ukončí hru a znova otvorí hernú kazetu: číslo je stále tam. B naň zabudne.

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

  Druhá kapitola v inom súbore

Úlohu v ukladacom priestore zdieľajú dve herné kazety. Časť 1: A označuje koniec hry. Časť 2, otvorená ako prvá, hovorí o dokončení časti 1. Potom sa otvorí časť 2 a B vymaže uloženie, aby sa časť 1 dala hrať znova.

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

 Jeden úspech

Sprite 8 je už nakreslený. Stlačte A desaťkrát. V oznámení sa zobrazí názov úspechu a sprite 8 zostane, pretože je odomknutý. B vymaže tento úspech v lokálnom súbore a vynuluje počítadlo.

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

 Izba, do ktorej môžete vojsť

Herná kazeta s rozmermi 256×256. Biela bodka predstavuje kurzor. Menu je hostiteľ, pripojiť sa alebo zoznam serverov. Hostiteľ otvorí miestnosť s názvom prechádzka a spustí hru. Pripojiť sa posúva adresu (číslice, bodka, dvojbodka) a B sa pripája. Zoznam serverov sa posúva hore a dole a A sa pripája k riadku. V hre šípky

prechádzajú po sprite 1. Jedna minca je zdieľaná: hostiteľ ju pohne a jej zdvihnutie pridá 1 k skóre daného hráča a spustí ďalšiu. Každý sprite, vrátane vašej, zobrazuje toto skóre vedľa seba. Roh zobrazuje, koľko hráčov je v miestnosti, LAN adresu a verejnú adresu hostiteľa. B odchádza: vy a ostatní sa vrátite do menu a ďalší hostiteľ začína novú miestnosť so skóre 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`



Dialóg, ktorý čaká

A zobrazí nasledujúci riadok. Funkcia sa pozastaví pri každom výnose, kým znova nestlačíte A.

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



Počkajte pol sekundy

Slovo sa zmení z ... na hi po približne pol sekunde. Obrazovka sa naďalej kreslí, aj keď je korutina pozastavená.

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



Kruhy, ktoré prichádzajú jeden po druhom

Zhora kvapká päť kruhov, dva rámy od seba vzdialené, a potom spadnú.

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



Dva rámy z hodín

Sprite 1 a sprite 2 sú dva snímky toho istého znaku. Obrázok sa strieda štyrikrát za sekundu.

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

Každá funkcia

Argumenty v hranatých zátvorkách sú voliteľné. Indexy tabuliek a reťazcov začínajú na 1, okrem riadkov a stĺpcov DATA, ktoré začínajú na 0. Farby sú indexy palet.

Index

`_nakresliť`
`_init`
`_mp`
`_mp_fail`
`_mp_join`
`_mp_leave`
`_mp_update`
`_set_fps`
`_aktualizovať`
`abs`
`achclear`
`achdef`
`dosiahnuť`
`dosiahnuté`
`pridať`

všetky
tvrdiť
atan2
kapela
bnot
bor
btn
btnd
btnp
btnu
bxor
fotoaparát
strop
chr
cirk
circfill
clickl
clickm
clickr
klip
cls
spoluvytvárať
farba
jadro
cos
costatus
počítať
kurzor
dcols
del
lahôdky
delsave
get
utápa
dset
dstr
fget
fillp
prevrátiť
flr
foreach
fset
info
ipairs
vypnutie kľúča
keyon
keystat
keystr
riadok
zaťaženie
log
lshr
mapa
max
memcpy
memset
mget
stred
min
myš
mset
hudba
musicstat
oznámiť
zavolať
ďalšie
nhost
njoin
nkb
nkbstr
nlan
nlist
nlist
počúvaj
poznámka
nplayers
nsend
nstun
obj
oválny
ovalfill
balenie
párov
kamarát
palrgb
palt
pcal
nahliadnuť
nahliadnuť 2
nahliadnuť 4
výkon
pget
tykať
poke2
poke4
vytlačiť
tlač 10
tlač8
pset
rect
rectfill
rnd
rotl
rotr
správne
rrectfill
uložiť
ukladací slot
obrazovke
rolovať
vyberte
setmap
setpalette
setspr
settex
setturn
sfx
sfxget
sfxhdr
sfxset
sfxstat
sget
sgn

shl	spr	čas
showstats	sqrt	tline
shr	srand	tonum
hriech	sset	tostr
songget	sspr	rozbaliť
súbor skladieb	stat	whoami
rozdeliť	sub	výnos

Životný cyklus

_init

`_init() ... end`

Spustí sa raz pri spustení hernej kazety. Kód, ktorý sa nachádza mimo akéhokoľvek hooku, je tiež zabalený do _init.

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

_update

`_update() ... end`

Herná logika. Volá sa na 60 Hz, aj keď je _draw obmedzený alebo je hostiteľ pomalší. Tlačidlá si prečítajte tu.

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

_draw

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

Nakreslí jeden snímok. Bez argumentu alebo s hodnotou 0 sa spustí tak často, ako hostiteľ prezentuje. Zadajte n (1..999) na obmedzenie kreslenia na n snímok za sekundu. _update zostáva na 60 Hz.

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

_set_fps

`_set_fps(n)`

Mení limit _draw počas behu hry. 0 znamená bez limitu. n je obmedzené na 0..999.

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

flip

`flip()`

Zobrazí obrazovku a prejde do ďalšieho snímku. Použite ho iba v prípade, že kreslíte vlastnú slučku snímok. Nečakajte po ňom.

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

time

`time()`

Sekundy od _init.

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

Vstup

btn

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

1, kým je stlačené tlačidlo b, inak 0. p je hráč, 0..7. Bez argumentov vracia bitovú masku hráča 0. Tlačidlá: 0 vľavo, 1 vpravo, 2 hore, 3 dole, 4 A, 5 B, 6 X, 7 Y, 8 R, 9 L, 10 vybrať, 11 začať.

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

btnd

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

Pravda iba v rámci _update, keď je stlačené tlačidlo b.

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

btnp

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

True pri stlačení, potom znova počas podržania: 30 tikov oneskorenia, potom každých 8 tikov. Bez stlačenia tlačidla vráti masku.

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

btnu

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

Pravda iba na ráme, kde sa tlačidlo uvoľní.

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

mouse

```
mouse()
```

Vráti súradnice myši x a y v pixeloch hry.

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

clickl

```
clickl([p])
```

Ľavé tlačidlo myši. clickl() je stlačené. clickl(1) je hrana stlačenia.

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

clickr

```
clickr([p])
```

Pravé tlačidlo myši. Rovnaká hrana s 1 argumentom ako clickl.

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

clickm

```
clickm([p])
```

Stredné tlačidlo alebo kliknutie rolovacieho kolieska.

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

scroll

`scroll()`

Pohyb kolieska myši v tomto ráme. Kladné a záporné hodnoty závisia od smeru otáčania kolieska.

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

keyon

`keyon()`

Otvorí textové pole ovládané tlačidlami. Stlačením hore a dole sa mení písmeno A–Z. A napíše dané písmeno. X ho vymaže. Y je medzera. B potvrdí a zatvorí pole. Klávesnica doň nepíše.

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

keyoff

`keyoff()`

Skryje textové pole.

```
keyoff()
```

keystr

`keystr([s])`

Text v poli, potom 1 v rámčeku B potvrdí, potom sa písmeno hore a dole mení. Toto písmeno je prázdne po zatvorení poľa. keystr("") vymaže text.

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

keystat

```
keystat()
```

1, kým je textové pole otvorené, inak 0.

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

Kreslenie

screen

```
screen()
```

Aktuálna veľkosť obrazovky hernej kazety v pixeloch. Vráti šírku a potom výšku. Toto je rozlíšenie hry, nie okno editora.

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

cls

```
cls([col])
```

Vymaže obrazovku. Predvolená farba je 0.

```
cls(1)
```

color

```
color(col)
```

Nastaví farbu vykresľovania, ktorá sa použije, keď neskoršie volanie vynechá farbu col. Bez argumentu vráti aktuálnu farbu.

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

camera

`camera([x, y])`

Offsety sa nastavujú pri každom volaní funkcie kreslenia. Funkcia camera() sa resetuje na 0, 0.

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

clip

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

Obmedzí kreslenie na obdĺžnik. Nenulový piaty argument sa pretína s aktuálnym klipom. clip() obnoví celú obrazovku. Registre Peek z klipu sú len na čítanie.

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

cursor

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

Nastaví, kam pôjde ďalší výpis bez súradníc a voliteľne aj farbu. cursor() vráti x, y.

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

pset

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

Nastaví jeden pixel obrazovky. Započítava sa do štatistiky px.

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

pget

```
pget(x, y)
```

Farba pixelu obrazovky v bodoch x, y.

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

print

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

Nakreslí text 3×5 (UTF-8, orezaný z písma Fusion). Bez x a y sa vytlačí v mieste kurzora a presunie sa na ďalší riadok.

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

log

```
log(...)
```

Pridá argumenty oddelené medzerami do textového bufferu. pix37_run vypíše tento buffer na stdout. Nevykresľuje.

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

print8

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

Rovnaké ako tlač, s fontom Fusion s veľkosťou 8 px.

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

print10

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

Rovnaké ako tlač, s fontom Fusion s veľkosťou 10 px.

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

line

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

Riadok od x0,y0 do x1,y1. Dva argumenty pokračujú od predchádzajúceho konca. line() resetuje tento reťazec.

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

circ

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

Obrys kruhu. Polomer je v pixeloch.

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

circfill

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

Vyplnený kruh.

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

oval

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

Obrys elipsy vo vnútri ohraničujúceho rámčeka. Rohy sú inkluzívne.

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

ovalfill

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

Vyplnená elipsa v ohraničujúcom rámčeku.

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

rect

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

Obrys obdĺžnika. Rohy sú inkluzívne.

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

rectfill

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

Vyplnený obdĺžnik.

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

rrect

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

Obrys zaobleného obdĺžnika. x a y sú ľavý horný roh. w a h sú veľkosti v pixeloch; obe musia byť väčšie ako 0. r je polomer rohu a predvolená hodnota je 0. Je obmedzený na polovicu kratšej strany.

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

rrectfill

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

Vyplnený zaoblený obdĺžnik. Rovnaké argumenty ako pri funkcii rrect.

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

fillp

```
fillp([p])
```

16-bitový vzor výplne pre neskoršie vyplnené tvary. 0 je plná. fillp() vráti aktuálny vzor. Bity prechádzajú vzorom 4×4.

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

pal

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

Premapuje farbu c0 na c1. Režim 0 je paleta kreslenia, 1 paleta zobrazenia, 2 paleta výplňových vzorov. Funkcia pal() vynuluje.

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

palrgb

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

Nastaví jednu položku palety na RGB, každý kanál 0..255.

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

palt

```
palt([col, t])
```

Priehľadnosť. Funkcia palt(col) alebo palt(col, true) urobí col priehľadným. Funkcia palt(col, false) ho urobí plným. Funkcia palt() urobí každú farbu nepriehľadnou.

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

setpalette

```
setpalette(n)
```

Vyberie banku palety č. setpalette() vráti aktuálnu banku.

```
setpalette(1)
```

tline

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

Nakreslí čiaru na obrazovku a zároveň vzorkuje pixely mapy z pozície mapy mx, my.mdx, mdy krok za krokom posúva vzorku mapy pozdĺž čiary.

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

Sprity a mapy

spr

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

Nakreslí sprite n v bodoch x a y. w a h sú veľkosti v pixeloch, predvolená hodnota je 8. fx a fy otočia horizontálne a vertikálne, ak sú nenulové. tex je karta hárka sprite. rot je uhol v otáčkach (0..1), rovnaký smer ako sin a cos. 0 ponechá sprite neotočený.

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

sspr

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

Skopíruje obdĺžnik zo sprite hárka na obrazovku, zmenšený na dw x dh. Predvolené nastavenia zachovajú veľkosť zdroja.

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

sget

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

Farba jedného texelu na hárku spritov.

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

sset

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

Zapíše jeden texel na hárok spritov.

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

fget

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

Príznaky Sprite. fget(n) vráti príznakový bajt. fget(n, f) je true, keď je nastavený príznakový bit f (0..7).

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

fset

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

Nastaví príznaky spritov. fset(n, v) zapíše celý bajt. fset(n, f, v) nastaví jeden bit.

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

setspr

```
setspr(n)
```

Vyberie aktívny hárok so spritemi. setspr() vráti aktuálny hárok. Rovnaká banka ako settex.

```
setspr(1)
```

settex

```
settex(n)
```

Alias pre setspr.

```
settex(0)
```

map

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

Nakreslí mapu. Bunka cx, cy mapy sa nakreslí na obrazovke sx, sy. cw, ch s hodnotou 0 znamená celú mapu. Vrstva sa akceptuje a ignoruje. tex je hárok s dlaždicami.

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

mget

`mget(cx, cy)`

ID spritu v bunke mapy cx, cy.

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

mset

`mset(cx, cy, n)`

Zapíše sprite n do bunky mapy cx, cy.

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

setmap

`setmap(n)`

Vyberie mapu n. setmap() vráti aktuálnu mapu. Mapa 0 je skutočná mapa.

```
setmap(1)
```

Pamäť

peek

`peek(addr)`

Prečíta jeden bajt z okna s veľkosťou 64 KB. Okno predstavuje živú obrazovku, potom pixely sprite a potom mapovanie buniek, keď sa obrazovka zmestí do 65280 bajtov. Väčšie obrazovky sa stránkujú pomocou poke(0x5F40, page). Adresy, ktoré nie sú namapované, vrátia 0. Tiež sa zapisuje @addr.

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

poke

`poke(addr, ...)`

Zapisuje bajty na adresu. Ďalšie argumenty zapisujú nasledujúce bajty. Nenamapované adresy sa ignorujú. Zapisuje sa aj $addr, val. Dotyk na obrazovke sa počíta ako zápis pixelu.

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

peek2

`peek2(addr)`

Neznamienková 16-bitová hodnota, little-endian, na adrese

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

poke2

`poke2(addr, val)`

Zapíše 16-bitovú hodnotu bez znamienka, little-endian.

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

peek4

`peek4(addr)`

Neznamienková 32-bitová hodnota, little-endian.

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

poke4

```
poke4(addr, val)
```

Zapíše 32-bitovú hodnotu bez znamienka, v poradí little endian.

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

memcpy

```
memcpy(dst, src, len)
```

Skopíruje len bajtov v rámci toho istého 64 KB okna.

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

memset

```
memset(dest, val, len)
```

Naplní len bajtov počnúc od cieľa hodnotou.

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

Matematika

abs

```
abs(x)
```

Absolútna hodnota.

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

flr

`flr(x)`

Zaokrúhľuje smerom k mínus nekonečnu.

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

ceil

`ceil(x)`

Zaokrúhľuje smerom k plus nekonečnu.

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

max

`max(a, b)`

Väčšia z dvoch hodnôt.

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

min

`min(a, b)`

Menšia z dvoch hodnôt.

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

mid

`mid(a, b, c)`

Stredná z troch hodnôt. Užitočná ako svorka.

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

sgn

`sgn(x)`

Znamienko x: -1, 0 alebo 1.

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

sqrt

`sqrt(x)`

Druhá odmocnina. Záporný argument nie je číslo.

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

sin

`sin(x)`

Sínus x je v striedavých bodoch: 0, 0,25, 0,5, 0,75 obíde kružnicu raz. Nie sú to radiány.

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

cos

`cos(x)`

Kosínus má zase rovnakú konvenciu ako sin.

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

atan2

`atan2(dx, dy)`

Uhol vektora dx, dy sa striedavo mení od 0 do 1.

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

rnd

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

Náhodné číslo v [0, x). Predvolená hodnota x je 1, takže rnd() je 0..1. Pre kocku, flr(rnd(6))+1.

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

srand

```
srand(seed)
```

Naprogramuje generátor náhodných čísel, aby sa neskoršia sekvencia RND mohla opakovať.

```
srand(1)
```

band

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

Bitový operátor AND. Tiež sa píše a & b.

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

bor

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

Bitový operátor ALEBO. Tiež sa píše a | b.

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

bxor

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

Bitový XOR. Tiež sa píše ako ^^ b.

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

bnot

`bnot(a)`

Bitové NIE. Tiež sa píše ~a.

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

shl

`shl(a, n)`

Posun doľava. Tiež sa píše ako << n.

shr

`shr(a, n)`

Aritmetický posun doprava. Tiež sa píše ako >> n.

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

lshr

`lshr(a, n)`

Logický posun doprava. Tiež sa píše ako >>> n.

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

rotl

`rotl(a, n)`

Otočiť bity doľava. Tiež sa píše <<> n.

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

rotr

```
rotr(a, n)
```

Otočte bity doprava. Tiež sa píše >>< n.

Tabuľky a hodnoty

add

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

Pridá v do tabuľky t. Ak je zadané i, vloží ho na index od 1.

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

del

```
del(t, v)
```

Odstráni prvú položku rovnú v.

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

deli

```
deli(t, [i])
```

Odstráni index i a vráti túto hodnotu. Ak i nie je zadané, odstráni posledný záznam.

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

all

```
all(t)
```

Iterátor nad časťou poľa, v poradí. Diery zastavujú prechádzku rovnako ako pole založené na 1.

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

foreach

```
foreach(t, fn)
```

Pre každý prvok poľa volá funkciu fn(v).

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

pairs

```
pairs(t)
```

Iterátor nad každým kľúčom a hodnotou vrátane pomenovaných polí.

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

ipairs

```
ipairs(t)
```

Iterátor nad časťou poľa s indexom založeným na 1.

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

next

`next(t, [k])`

Vráti nasledujúci kľúč a hodnotu po k. next(t) začína na začiatku. Používajú ho páry.

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

count

`count(t, [v])`

Dĺžka časti poľa. Ak je zadaná hodnota v, spočíta, koľko položiek sa rovná v.

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

pack

`pack(...)`

Zbalí argumenty do novej tabuľky.

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

unpack

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

Vráti t[i] až t[j] ako samostatné hodnoty. Predvolené hodnoty pokrývajú celú časť poľa.

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

select

`select(i, ...)`

select("#", ...) vráti počet nasledujúcich argumentov. select(i, ...) vráti argumenty od i ďalej.

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

assert

```
assert(c, [msg])
```

Zastaví sa s chybou, keď je c false alebo nil. msg nahradí predvolený text.

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

pcall

```
pcall(f, ...)
```

Volá funkciu f. V prípade úspechu vráti hodnotu true a výsledok. V prípade chyby vráti hodnotu false a text chyby.

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

Struny

chr

```
chr(n)
```

Jeden znak z bajtového kódu 0..255.

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

ord

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

Bajtové kódy s od indexu i do j. Indexy sú založené na 1. Predvolená hodnota je prvý znak.

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

sub

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

Podreťazec od i do j, založený na 1. Vynechajte j, ak chcete zostaviť chvost.

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

split

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

Rozdelí s do tabuľky. sep má predvolenú hodnotu čiarka. Číslo sep je šírka bloku. conv konvertuje časti s tonum, keď je pravda.

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

tonum

```
tonum(s, [base])
```

Analyzuje číslo. Základ môže byť 2 až 36. Vráti nil, ak text nie je číslo.

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

tostr

```
tostr(v, [digits])
```

Prevedie hodnotu na text. Funkcia digits nastaví desatinné miesta pre čísla.

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

Zvuk

sfx

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

Prehrá vzor n a vráti kanál. Ak vynecháte ch alebo zadáte -1, vyberie sa prvý voľný kanál. ch -2 zastaví vzor n na každom kanáli. n -1 zastaví kanál alebo každý kanál, keď je ch vynechaný. n -2 umožní dokončiť cyklický vzor namiesto jeho opakovania. sfx() bez argumentov vráti -1.

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

sfxget

```
sfxget(n, step)
```

Vráti tabuľku s jedným krokom patternu: [1] výška tónu, [2] vlna, [3] hlasitosť, [4] efekt. Výška tónu je 0..63 (C-0 až D#5). Vlna je 0..15. Hlasitosť je 0..7 a 0 je ticho. Efekt je 0..7.

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

sfxset

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

Zapíše jeden krok vzoru. Všetkých šesť argumentov je povinných.

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

sfxhdr

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

Hlavička vzoru. sfxhdr(n) vráti tabuľku hlavičiek. sfxhdr(n, key, val) nastaví jedno pole: speed, loop_in, loop_out, level, noise, crush, dtune, rverb, dampn, swing, steps. Čísla 0..10 vyberajú rovnaké polia v tomto poradí. speed je TIC, tiky na krok. Vyššia hodnota je pomalšia. 15 je 120 úderov za minútu. 16-bitový režim akceptuje 1..255. Užšie syntetizátory sa prichytia k svojmu zoznamu TIC.

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

sfxstat

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

ID vzoru prehrávaného na kanáli ch alebo -1. Predvolený kanál je 0.

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

note

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

Hrá jednu notu bez vzoru. Výška tónu 0..63, vlna 0..15, hlasitosť 0..7, efekt 0..7. Tiky udávajú dĺžku trvania, v rovnakých 120 Hz tikoch ako TIC. Ak vynecháte tiky, trvanie je 15, jeden predvolený krok. Efekt beží počas tohto času a potom sa kanál stíchne.

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

music

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

Spustí skladbu n od začiatku. music(-1) zastaví prehrávanie. Volanie funkcie music s ID, ktoré sa už prehráva, ju ponechá spustenú. music() vráti ID prehrávanej skladby alebo -1. Fade je v milisekundách: skladba sa postupne objavuje alebo stlmuje, keď je n -1. Nastavené bity masky predstavujú kanály použité pre skladbu. 0 znamená všetkých osem. Ostatné kanály zostávajú voľné pre zvukové efekty.

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

musicstat

`musicstat()`

Tabuľka prehrávanej skladby. song a row predstavujú ID skladby a riadok, alebo -1, keď sa nič neprehráva. step a pos majú osem čísel. step[1] je kanál 0, step[8] je kanál 7. step je index noty, alebo -1, keď je kanál nehybný. pos je, ako ďaleko sa nota posunula, od 0 do 1. tick je, koľko tikov sa tento riadok už prehral. played je, koľko riadkov sa začalo od vytvorenia music(), vrátane toho, ktorý sa práve prehráva.

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

songget

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

ID vzoru uložené v bunke skladby. ch je 0..7. 255 znamená, že bunka je prázdna. Toto je aktívne volanie; text nápovedy kódu stále zobrazuje staršie poradie argumentov.

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

songset

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

Zapíše ID vzoru do bunky skladby. Na vymazanie bunky použite hodnotu 255.

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

stat

`stat(n)`

Iba zvuk. 16..23 je ID sfx na kanáli n-16. 24..31 je hudobný vzor na kanáli n-24. Akékoľvek iné n vráti 0. Časovanie snímok a RAM sú výkon a informácie, nie štatistiky.

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

Tabuľky údajov

dget

`dget(t, row, col)`

Prečíta bunku dátovej tabuľky t z panela DATA. Riadok a stĺpec začínajú od 0.

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

dset

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

Zapíše bunku. Pre textový stĺpec použite reťazec.

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

dstr

`dstr(t, row, col)`

Bunka ako text. Čísla sú skonvertované.

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

drows

`drows(t)`

Koľko riadkov má tabuľka t.

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

dcols

```
dcols(t)
```

Koľko stĺpcov má tabuľka t.

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

Uloženie a úspechy

save

```
save(k, v)
```

Uloží v pod kľúč k v uloženom súbore. v môže byť číslo, reťazec alebo tabuľka týchto čísel. Súbor je komprimovaný.

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

load

```
load(k)
```

Prečíta kľúč k. Vráti nil, ak kľúč chýba.

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

delsave

```
delsave(k)
```

Odstráni kláves k z uloženej pamäte.

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

saveslot

```
saveslot([name])
```

Vyberie súbor na uloženie v priečinku autora. Ak nie je zadaný názov, slot je názov hry. Vráti 1, ak daný súbor už existuje.

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

achieve

```
achieve(i)
```

Odomkne úspech i, od 0 do 15. Uložené lokálne. Zostavenie na Steame tiež nastaví ACH_i. Relácia katalógu to nahlási na účet hráča.

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

achieved

```
achieved(i)
```

1, keď je úspech i už odomknutý.

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

achdef

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

Upozorňujúci text pre dosiahnutie úspechu i, zobrazený na 15 sekúnd od spodnej časti obrazovky. spr je voliteľná dlaždica 8×8, tex jej hárok.

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

achclear

```
achclear(i)
```

Vymaže úspech i v lokálnom súbore. Už nastavený úspech Steam ACH_i zostane nastavený.

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

Sieť

whoami

```
whoami()
```

Vaše ID hráča, 0..15, počas prebiehajúcej relácie. Offline je -1.

```
id = whoami()
```

nhost

```
nhost()
```

1, ak je tento počítač hostiteľom, inak 0.

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

nplayers

```
nplayers()
```

Koľko živých ID existuje v 0..15.

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

nlisten

`nlisten()`

Spustí hosťovanie UDP a vráti LAN adresu ako ip:port.

```
addr = nlisten()
```

njoin

`njoin(addr)`

Pripája sa k IP alebo IP:port.

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

nleave

`nleave()`

Opúšťa reláciu. Peerovia vidia _mp_leave s argumentom prečo 0.

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

nsend

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

Odošle stratovú správu. Názov je udalosť. Nasledujú až štyri ďalšie hodnoty. p od 0 do 15 šepká danému peerovi; vynechajte p pre odoslanie všetkým.

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

ncall

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

Rovnaký tvar ako nsend, ale správa sa opakuje, kým sa nepotvrdí. Tri zlyhania za sebou ukončia reláciu.

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

nlan

```
nlan()
```

Aktuálna IP adresa siete LAN po nlisten.

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

nstun

```
nstun()
```

Vaša verejná IP adresa:port zo STUN. Hosť z inej siete sa k nej môže pokúsiť pripojiť.

```
pub = nstun()
```

setturn

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

Záložná funkcia TURN sa použije, keď priama cesta zlyhá. Funkcia setturn() s prázdnymi argumentmi ju vypne. Adresa je host:port.

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

nlist

`nlist(game)`

Tabuľka otvorených relácií pre daný názov hry. Každý záznam má adresu, hráčov a meno.

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

nannounce

`nannounce(game)`

Prezenčný signál, takže tento hostiteľ sa zobrazí v zozname hostí pre hru. Zavolajte ho na časovač počas hosťovania.

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

nkb

`nkb([on])`

Pole adresy. nkb(1) ho otvorí, nkb(0) ho skryje, nkb() vám oznámi, či je otvorené. Stlačením klávesu hore a dole sa mení znak, 0–9 alebo bodka alebo dvojbodka. A ho zapíše. X ho vymaže. Klávesnica doň nepíše.

```
nkb(1)
```

nkbstr

`nkbstr([s])`

Text v poli adresy, potom znaky hore a dole sa menia. nkbstr("") vymaže text.

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

_mp

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

Túto funkciu napíšete. Spustí sa, keď príde správa. from je ID odosielateľa. name je udalosť. Nasledujú až štyri hodnoty.

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

_mp_join

`_mp_join(p)`

Napíšete túto funkciu. Peer dostal ID p. Spustí sa aj pre vás; whoami() je toto ID.

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

_mp_leave

`_mp_leave(p, why)`

Napíšete túto funkciu. Partner p odišiel. Prečo 0 je normálny odchod. Prečo 1 je stratené spojenie.

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

_mp_update

`_mp_update()`

Túto funkciu napíšete. Spustí sa každých 15 snímok _update, približne 4 Hz. Dobré miesto na odosielanie pozícií.

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

_mp_fail

`_mp_fail()`

Napíšete túto funkciu. Relácia je nefunkčná: hostiteľ zmizol alebo ncall trikrát zlyhal.

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

Korutiny

cocreate

`cocreate(f)`

Vytvorí korutinu z funkcie f. Nespustí sa, kým sa nevykoná coresume.

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

coresume

`coresume(c, ...)`

Spúšťa korutinu c, kým nevráti hodnotu alebo hodnotu. V prípade úspechu: true a vrátené hodnoty. V prípade chyby: false a chyba.

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

costatus

```
costatus(c)
```

„pozastavený“, „bežiaci“ alebo „mŕtvy“.

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

yield

```
yield(...)
```

Vracia kontrolu späť tomu, kto zavolal coresume. Hodnoty sa stávajú výsledkami tohto coresume.

```
yield(1)
```

Opatrenie

info

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

Vykreslí živý HUD v _draw: fps (priemerne 1 sekunda), upd, drw, px, max, ram, peak, ramc a net, keď je relácia spustená. info() používa kurzor tlače. info(col) nastavuje iba farbu. info(x, y) alebo info(x, y, col) ju umiestňuje. Čísla používajú B, k alebo m, aby sa zmestili na malú obrazovku. Samotné volanie sa nepočíta v px.

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

showstats

```
showstats([on])
```

Rovnaký HUD ako info, vykresľuje sa automaticky na 0, 0 po _draw, ignoruje kameru a klip. showstats(0) ho skryje. showstats(1) alebo showstats() ho zobrazí.

```
showstats(1)
```

perf

`perf(n)`

Jedno číslo HUD, aby si si ho mohol nakresliť sám. 0 fps, 1 upd, 2 draw, 3 px, 4 max (najťažší snímok v tejto sekunde), 5 ram, 6 ramc (ram plus zabalená herná kazeta), 7 peak ram od začiatku hernej kazety.

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

Návod na PIX37. Kamil Stanisławski.