laio_DOC

Benji_haOma/nA!

COLLABORATORS			
	TITLE : Iaio_DOC		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Benji_haOma/nA!	August 13, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Iaio_DOC

1.1 Iaio doc!

```
                    (nie)Peîna dokumentacja do programu "Iaio - The PackMaker"  ↩
                       v0.7ß
                                                  Benji_HaOma/nEGATIVE!


                  CO TO JEST?

                  OPIS KOMEND

                  REJESTRACJA

                  AUTOR

                  IAIO PRODS
```

1.2 Iaioprods

```
                     znane mi produkty napisane w Iaio'u


          pack:  instant #1/appendix

    crazy demo:  pamela/8bits
                 8bs-vzy-murderer/8bits
                 o.s.w./8bits
```

1.3 coto

```
                   Iaio  - The PackMaker to jak najbardziej multimedialny jëzyk  ↩
                      programowania
skryptowego,  z  poîoûonym  naciskiem  na  tworzenie  tzw.  packów scenowych
```

(patrz: Instant/Appendix), z tymûe Iaio zostaîo juû uûyte takûe do tworzenia crazy dem, oraz skryptów do tzw. bootowania systemu (w stylu Syspi*, oraz Examples/boot_amiga/).

Iaio odznacza sië pracâ z systemem, jednak pozostaje jeszcze wiele rzeczy do zrobienia.

Program ma status *SHAREWARE* (patrz
REJESTRACJA
.)

1.4 rejestracja

Program posiada status *SHAREWARE*, róûnice pomiëdzy wersjâ ↩
zarejestrowanâ
i nie sâ ûadne, poza imiennâ rejestracjâ oraz usuniëciem procedury 'jay'.

Aby jâ otrzymaê naleûy wysîaê na mój adres
e-mailowy lub domowy (+znaczek)
list zawierajâcy 3 przykîadowe skrypty Iaio'a wîasnego ↩
autorstwa lub
jakikolwiek program swojego autorstwa zarejestrowany na uûytkownika Benji_haOma/Negative!. UWAGA: zastrzegam sobie prawo doîâczenia skryptów do pakietu Iaio, oraz ewentualnych poprawek!

OSTRZEÛENIE!: próba uruchomienia zcrackowanej wersji Iaio'a, moûe spowodowaê reset komputer'a, oraz ewentualny pad walidacji dysku! Zostaîeô ostrzeûony ;)

1.5 autor

-iaio- by BENji_haOma/
nEGATIVE!

Pawel Nowak
ul.Szpitalna 7/13
31-024 Krakow
Poland

juen@tl.krakow.pl

1.6 spis

[KOMENDA] - nazwa komendy wraz z argumentami
[UÛYCIE] - przykîadowe uûycie komendy
[FUNKCJA] - opis komendy

Spis komend:

```
#(text)

-(proc):

jump

quit

debug

sumpalblack

lamelmb

lamermb

closescreen

fadetowhite

fadetoblack

loadpal

fadetopal

copypal

tofront

hidescreen

exec

checkaudio

print

onloaderror

onkey

digiplay

digistop

digifadedown

loadraw

inc

dec

case
```

```
nocase

loadbank

countpages

bintoascii

textpage

backpage

freegadget

addgadget

checkgadget

ongadget
                    v0.5:
checkfont
                    v0.6:
reset

freefree

freepublic

freechip
                    v0.7:    Uwaga, komendy od wersji 0.7+ majâ ↩
                       poprawionâ
             skîadnie:

             np.: delay[100]
                  =
                  set[0,100]
                  delay[!0]    (!nr_komorki)
                  =
                  set[0,100] : delay[!0]
                  =
                  delay[$64]   ($wartosc_hex)

               Stara skîadnia oczywiôcie dalej aktualna.
               Uwaga:  komendy dla, których sâ podawane
             argumenty w postaci =(text) muszâ znajdowaê sië
             w linii jako ostatnie!

             &file
             &digi

iaiochicken

delay
                            add
             sub
             setcol
```

```
                    chled
                    onlmb
                    onrmb

          text

          openscreen
                                        rectfill
                    setfont

          setpen

          set
                                        draw
                    move
                    writepixel
                    readpixel
                    getmouse
                    return
                    flood
                    deletefile
                    findcol
                    div
                    mul
                    digigetpattpos
                    digigetsongpos
                    putraw
                    neg
                    hipget
                    hipcloseport
                    hipopenport
                    do
                    loop

          freebank
                                        digiplug
```

Pozostajâ takûe funkcje takie jak (których z wiadomych dla mnie przyczyn nie opisuje):

```
clearpointer[]
cmp[]
high[]
low[]
same[]
initrnd
rnd[]
random[]
```

1.7 iaiochicken

```
KOMENDA: iaiochicken
UÛYCIE : #sprawdzam audio, czy nie zajete, jezeli nie:
         iaiochicken
         #...
FUNKCJA: iaio mówi :)
```

1.8 komentarz

```
KOMENDA: #(komentarz)
UÛYCIE : #tutaj nastëpuje odczyt artykuîów:
         ...
         #tu jest gîówna pëtla...
         ...
FUNKCJA: komentarz ;]
```

1.9 proc

```
KOMENDA: -(nazwa_procedury):(enter)
UÛYCIE : -moja_procedura:
UÛYCIE : -inna_procedura:
UÛYCIE : -loop:
         jump[loop]
FUNKCJA: etykieta procedury, która nastëpuje wîaônie w miejscu jej zapisania
```

1.10 jump

```
KOMENDA: jump[(procedura)]
UÛYCIE : -loop:
         ...
         jump[loop] ;zapëtlenie
FUNKCJA: skok do podanej procedury
```

1.11 quit

```
KOMENDA: quit
UÛYCIE : quit
FUNKCJA: zakoïczenie wykonywania programu
```

1.12 debug

```
KOMENDA: debug
UÛYCIE : debug
FUNKCJA: zwrócenie wartoôci rejestrów do okna (expert)
```

1.13 sumpalblack

```
KOMENDA: sumpalblack[(ekran)]
UÛYCIE : sumpalblack[0]
FUNKCJA: ustawia czarnâ paletë dla podanego ekranu (od 0 do 9)
```

1.14 lamelmb

```
KOMENDA: lamelmb
UÛYCIE : lamelmb
FUNKCJA: zawieszenie wykonywania programu do czasu naciôniëcia lewego
         przycisku myszy
```

1.15 lamermb

```
KOMENDA: lamermb
UÛYCIE : lamermb
FUNKCJA: zawieszenie wykonywania programu do czasu naciôniëcia prawego
         przycisku myszy
```

1.16 closescreen

```
KOMENDA: closescreen[(ekran)]
UÛYCIE : closescreen[0]
FUNKCJA: zamknij ekran (wczeôniej otwart przez openscreen), gdzie ekran
         to numer od 0 do 9 (max 10 ekranów na raz)
```

1.17 fadetowhite

```
KOMENDA: fadetowhite[(ekran),(delay)]
         fadetoblack[(ekran),(delay)]
UÛYCIE : fadetoblack[0,1]
FUNKCJA: fejduje palete ekranu (od 0 do 9), przy opóûnieniu podanym jako
         delay, ustawiony na 1 daje nam opóûnieje 1/50s, moûna ustawiê takûe 0
```

1.18 fadetoblack

```
KOMENDA: fadetowhite[(ekran),(delay)]
         fadetoblack[(ekran),(delay)]
UÛYCIE : fadetoblack[0,1]
FUNKCJA: fejduje palete ekranu (od 0 do 9), przy opóûnieniu podanym jako
         delay, ustawiony na 1 daje nam opóûnieje 1/50s, moûna ustawiê takûe 0
```

1.19 loadpal

```
KOMENDA: loadpal[(ekran)]=(plik_z_paleta);
UÛYCIE : loadpal[0]=progdir:data/pal/main.pal;
FUNKCJA: komenda odczytuje z pliku podanego na podany ekran (0-9) palete,
         format pliku z paleta to 32-u bitowa paleta, by uzyskaê takowâ
         z pliku ilbm iff, naleûy uûyê doîâczonego programu tools/savecmap,
         podajemy dla niego jako argument nazwe iffu, paleta jest zapisywana
         do ram:cmap.pal
```

1.20 fadetopal

```
KOMENDA: fadetopal[(ekran),(delay)]=(plik_z_paleta);
UÛYCIE : loadpal[0,1]=progdir:data/pal/main.pal;
FUNKCJA: zasada dziaîania ta sama co w przypadku fadetoblack i loadpal,
         z tâ ûe róûnicâ iû fejdowana jest paleta do podanej w argumentach
```

1.21 delay

```
KOMENDA: delay[(ile*1/50sekundy)]
UÛYCIE : delay[50]
FUNKCJA: odczekuje okreôlony czas, w przypadku podania 50 bëdzie to 1 sek.
```

1.22 copypal

```
KOMENDA: copypal[(ekran1),(ekran2)]
UÛYCIE : copypal[0,1]
FUNKCJA: kopiuje palete z ekranu, do ekranu, pamiëtaj ûe max. wartoôê ekranu
         to 9 (od 0)
```

1.23 tofront

```
KOMENDA: tofront[(ekran)]
UÛYCIE : tofront[0]
FUNKCJA: wystawienie podanego ekranu do przodu
```

1.24 hidescreen

```
KOMENDA: hidescreen[(ekran)]
UÛYCIE : hidescreen[0]
FUNKCJA: schowanie ekranu podanego pod wszystkie inne
```

1.25 exec

```
KOMENDA: exec=(program_wykonywalny);
UÛYCIE : exec=c:dir DF0: >CON:0/0/640/256/Zawartoôê DF0:;
UÛYCIE : exec=reset;
FUNKCJA: wykonuje podany program
```

1.26 checkaudio

```
KOMENDA: checkaudio[(procedura)]
UÛYCIE : checkaudio[audioerror]
         ...
         -audioerror:
         print=Nie mogë zarezerwowaê kanaîów audio!
         quit
FUNKCJA: testuje kanaîy audio, jeûeli sâ juû zarezerwowane skacze do podanej
         procedury
```

1.27 print

```
KOMENDA: print=(text)
UÛYCIE : print=Script by Lolek
FUNKCJA: wypisanie do wyjôcia textu (stdout, np. do cli)
```

1.28 onloaderror

```
KOMENDA: onloaderror[(procedura)]
UÛYCIE : onloaderror[load_error]
         ...
         -load_error:
         print=Nie mogë wczytaê danych!
         quit
FUNKCJA: ustala do jakiej procedury program ma przejôê podczas problemów
         z wczytywaniem jakichkolwiek plików
```

1.29 onkey

```
KOMENDA: onkey[(key),(proc)]
UÛYCIE : onkey[117,koniec]   ;117=ESC
FUNKCJA: sprawdza kod raw wciôniëtego przycisku, jeûeli zostaî podany
         naciôniëty przechodzi do podanej procedury
```

1.30 digiplay

```
KOMENDA: digiplay=(nazwa_modulu_digiboostera);
UÛYCIE : digiplay=progdir:data/music/voyager.digi;
FUNKCJA: wczytuje i zaczyna odtwarzaê podany moduî, który musi byê w formacie
         digiboostera =< 1.7
```

1.31 digistop

```
KOMENDA: digistop
UÛYCIE : digistop
FUNKCJA: zatrzymuje odtwarzanie moduîu digiboostera, oraz usuwa go z pamiëci
```

1.32 digifadedown

```
KOMENDA: digifadedown[(delay)]
UÛYCIE : digifadedown[1]
FUNKCJA: ôcisza aû do 0 gîoônoôê odtwarzania moduîu digiboostera, delay
         jest to wartoôê co jakâ bëdzie zmniejszana gîoônoôê (1=1/50 sek),
         podajâc tu jeden moduî zostanie caîkowicie wyciszony po:
         64*(1*1/50) sek ;]
```

1.33 text

```
KOMENDA: text[(ekran),(x),(y)]=(text)
UÛYCIE : text[0,10,10]=Super! Mój wîasny tekst ;]
FUNKCJA: wypisuje na podanym ekranie, o podanych wspóîrzëdnych podany text :]
```

1.34 setpen

```
KOMENDA:~setpen[(ekran),(kolor_tekstu),(kolor_podkladu),(spec)]
UÛYCIE : setpen[0,1,0,0]
FUNKCJA: ustala kolor pisanego textu i jego podkladu dla podanego ekranu,
         podajâc dla _spec_ wartoôê 0, tekst bëdzie pisany na podkîadzie,
         w przypadku 1, podkîad bëdzie koloru _kolor_podkladu_
```

1.35 openscreen

```
KOMENDA: openscreen[(ekran)]=(szerokoôê)/(wysokoôê)/(bitplanes)/(typ)/(fonts)
         (+size)/(spec);
UÛYCIE : openscreen[0]=640/256/8/h/thingpl.font/;
UÛYCIE : openscreen[0]=640/256/8/h/topaz.font+11/;
UÛYCIE : openscreen[0]=640/512/1/s//bakg;
FUNKCJA: otwiera (ekran) o szerokoôêi i wysokoôci podanej, o i liczbie
```

```
         podanych bitplaneów (1=2 kolory, 2=4 kolory, 3=8 kolorów, 4=16
         kol., 5=32 kol., 6=64 kol., 7=128 kol., 8=256 kolorów), w typ
         podajemy h dla ekranu hires, l dla ekranu lowres, lub s dla
         hires + lace, w fonts podajemy fonty (np. topaz.font), ew. nic
         nie podajemy (jak na przykîadzie), a takûe + i wielkoôê (domyôlnie
         8!), w spec moûna podaê argument bakg, który to otworzy nam ekran
         pod wszystkimi innymi, ew. teû nic nie podajemy.
```

1.36 loadraw

```
KOMENDA: loadraw[(ekran),(szer),(wys),(bitplanes),(x),(y)=(plik_raw);
UÛYCIE : loadraw[0,640,256,8,0,0]=progdir:data/gfx/panel640x256x8.raw;
FUNKCJA: wczytuje obrazek w formacie RAW (format ten moûna uzyskaê na
         takich programach jak AgaIFF, czy Personal Paint) o podanej
         szerokoôci, wysokoôci i iloôci bitplaneów, na podany ekran,
         o podanych wspóîrzëdnych
```

1.37 set

```
KOMENDA: set[(komorka),(wartoôê)]
UÛYCIE : set[0,10]  ;ustawi komorke 0 na 10
UÛYCIE : set[89,10] ;ustawi komorke 89-wiâtâ na numer 10
FUNKCJA: ustawia podanâ komórke na podanâ wartoôê, maxymalny numer komórki
         to 1000, kaûda komórka moûe przechowywaê maxymalnâ wartoôê 32bity
```

1.38 inc

```
KOMENDA: inc[(komorka),(o_ile)]
UÛYCIE : inc[0,10] ;zwiëkszy wartoôê komórki 0 o 10
FUNKCJA: zwiëksza wartoôê podanej komórki o podanâ wartoôê
```

1.39 dec

```
KOMENDA: dec[(komorka),(o_ile)]
UÛYCIE : dec[0,10] ;zmniejszy wartoôê komórki 0 o 10
FUNKCJA: zmniejsza wartoôê podanej komórki o podanâ wartoôê
```

1.40 case

```
KOMENDA: case[(komorka),(wartosc),(procedura)]
UÛYCIE : case[0,10,proc_kom0_ma_wartosc_10]
         ...
         -proc_kom0_ma_wartosc_10:
         ...
FUNKCJA: sprawdza czy podana komorka ma podana wartosc, jak tak to skacze
         do podanej procedury
```

1.41 nocase

```
KOMENDA: nocase[(komorka),(wartosc),(procedura)]
UÛYCIE : case[0,10,proc_kom0_nie_ma_wartosci_10]
         ...
         -proc_kom0_nie_ma_wartosci_10:
         ...
FUNKCJA: sprawdza czy podana komorka nie ma podanej wartosci, jezeli wlasnie
         tak jest to skacze do podanej procedury
```

1.42 loadbank

```
KOMENDA: loadbank[(numer_banku),(typ_pamieci)]=(plik);
UÛYCIE : loadbank[0,p]=progdir:data/text/introduction.txt;
FUNKCJA: wczytuje do banku podanego (od 0 do 9), podany plik, typ pamieci
         to p dla public c dla chip, oraz f dla fast, UWAGA: UÛYWAJ p !
```

1.43 freebank

```
KOMENDA: freebank[(bank)]
UÛYCIE : freebank[0]
FUNKCJA: usuwa z pamiëci podany bank (0-100)
```

1.44 countpages

```
KOMENDA: countpages[(bank),(enters),(komorka)]
UÛYCIE : countpages[0,18,30]
FUNKCJA: w enters podajemy ilosc enterów na strone, jako bank podajemy
         wczytany bank z !PLIKIEM TEXTOWYM! jako komorka podajemy numer
         komorki, w której chcemy otrzymaê wynik, którym jest juû caîkowita
         przeliczona iloôê stron (uûyteczne np. do licznika stron)
```

1.45 bintoascii

```
KOMENDA: bintoascii[(ekran),(komorka),(x),(y)]
UÛYCIE : bintoascii[0,30,20,10]
FUNKCJA: zapisuje w postaci tekstu wartosc z podanej komorki na x i y
         podanego ekranu
```

1.46 textpage

```
KOMENDA: textpage[(ekran),(komorka1),(bank),(x),(y),(linie),(kom2),(kom3)]
UÛYCIE : textpage[0,100,0,20,20,18,101,102]
FUNKCJA: funkcja wypisuje text, od aktualnego jego polozenia (po wczytaniu
         do banku textu, jestesmy zawsze na jego poczatku),
         (ekran) - na tym ekranie wypiszemy text,
         (komorka1) - tej komorki bedzie program uzywal, nie nalezy jej
         pozniej zmieniac,
         (bank) - numer banku z textem,
         (x),(y) - x i y ekranu no i dla textu,
         (linie) - ilosc linii textu jaka ma zapisac (ta sama wartosc
         podajemy jako argument (enters) dla countpages[]),
         (kom2) - tu podajemy nastepny numer komorki do uzywania dla funkcji,
         tez nie nalezy jej pozniej samemu zmieniac, jednak mozna ja
         sprawdzac: 1=tekst jeszcze nie doszedl do ostatniej strony,
         2=tekst juz doszedl do konca (eof),
         (kom3) - ostatnia juz komorka jaka tu ustawiamy, a nastepnie jej
         juz nie ruszamy, nalezy o tym pamietac, ze raz juz podane numery
         komorek podajemy juz za kazdym razem te same dla funkcji textpage
         jak i zarazem backpage!
```

1.47 backpage

```
KOMENDA: backpage[(komorka1),(bank),(kom2),(kom3),(linie)]
UÛYCIE : backpage[100,0,101,102,18]
FUNKCJA: cofa o caîâ strone text w banku (bank), uzywajac podanych trzech
         komorek (bardzo scisle zwiazanych z textpage[]), (linie) to ta
         sama iloôê linii podana w textpage[] takze jako (linie) (zobacz!)
```

1.48 freegadget

```
KOMENDA: freegadget[(ekran),(numer_gadgetu)]
UÛYCIE : freegadget[0,1]
FUNKCJA: usuwa gadget z listy gadgetow dla podanego ekranu
```

1.49 addgadget

```
KOMENDA: addgadget[(ekran),(numer_gadgetu),(x),(y),(szer),(wys)]
UÛYCIE : addgadget[0,1,20,20,50,10]
FUNKCJA: dodaje gadget do podanego ekranu, o podanym numerze, o wspolrzednych
         x i y, oraz podanej szerokosci i wysokosci
```

1.50 checkgadget

```
KOMENDA: checkgadget[(ekran)]
UÛYCIE : checkgadget[0]
FUNKCJA: sprawdza status gadgetow dla podanego ekranu, nastëpnie moûna
         uûyê komendy ongadget
```

1.51 ongadget

```
KOMENDA: ongadget[(gadget),(procedura)]
UÛYCIE : ...
         -proc_retry:
         ...
         checkgadget[0]
         ongadget[1,proc_abort]
         ongadget[2,proc_retry]
         ...
         -proc_abort:
         quit
FUNKCJA: sprawdza czy podany numer gadgetu zostal wcisniety, jak tak
         to skacze do podanej procedury, przed uzyciem komendy ongadget[],
         naleûy uûyê komendy checkgadget[], która pobiera stan gadgetów
         dla podanego ekranu!
```

1.52 checkfont

```
KOMENDA: checkfont[(nazwa_fontow)]
UÛYCIE : checkfont[thingpl.font]
         ...
FUNKCJA: sprawdza czy mozna otworzyc podane fonty, w wypadku nie udanej
         proby wyswietla do cli blad o otwarciu podanych liter i zakancza
         wykonywanie programu.
```

1.53 reset

```
KOMENDA: reset
UÛYCIE : ...
         reset
FUNKCJA: wykonuje najzwyklejszy (soft) reset.
         ( move.l 4.w,a6
           jsr    -726(a6) )
```

1.54 freefast

```
KOMENDA: freefast[(komorka)]
UÛYCIE : freefast[20]
         text[0,50,42]=Ilosc wolnej pamieci fast:
         bintoascii[0,20,50,50]
         ...
FUNKCJA: zwraca do podanej komorki wartosc wolnej pamieci fast.
```

1.55 freepublic

```
KOMENDA: freepublic[(komorka)]
UÛYCIE : freepublic[20]
         text[0,50,42]=Ilosc wolnej pamieci (total):
         bintoascii[0,20,50,50]
         ...
FUNKCJA: zwraca do podanej komorki wartosc calkowitej (chip+fast) wolnej
         pamieci.
```

1.56 freechip

```
KOMENDA: freechip[(komorka)]
UÛYCIE : freechip[20]
         text[0,50,42]=Ilosc wolnej pamieci chip:
         bintoascii[0,20,50,50]
         ...
FUNKCJA: zwraca do podanej komorki wartosc wolnej pamieci chip.
```