

# Start Up Guide

Software Development Tool



## Table of Contents

---

INTRODUCTION.....	7
MEMBERS ONLY WEB SITE.....	9
CONNECTION TO THE MEMBERS ONLY WEB SITE .....	9
SOFTWARE NECESSARY FOR CONNECTION .....	9
ID AND PASSWORD.....	9
BEING A MEMBER OF NET YAROZE.....	9
FURTHER INFORMATION- SUPPORT .....	9
CONTENTS OF THE PACKAGE.....	11
OPERATING ENVIRONMENT.....	13
HARDWARE.....	13
SOFTWARE .....	13
SET UP.....	15
HARDWARE CONNECTION .....	15
SOFTWARE SET UP.....	16
GETTING STARTED.....	22
QUICK START .....	22
QUICK DIRECTORY CONTENTS .....	26
SAMPLE PROGRAM.....	27
SAMPLE PROGRAM.....	30
SAMPLE PROGRAM MAKE FILE .....	30
SAMPLE PROGRAM SOURCE LIST .....	30
ADDITIONAL READING.....	41
C PROGRAMMING.....	41
GRAPHICS.....	41





## About NetYaroze

---

### What You Need to Know

In order to get started with Net Yaroze, you should have experience of C programming to a competent level and a knowledge of a 2D graphic creation/editing tool. In addition, at least a basic grasp of a 3D modelling package and a sound creation/editing tool would help you get the best out of your NetYaroze kit.

### The Net Yaroze Manual Set

There are three books in the set of NetYaroze manuals.

- Start Up Guide (this document)

An introductory booklet explaining the contents and requirements of the NetYaroze Starter Kit. It also gives step by step instructions on setting up the NetYaroze software on your PC and how to run Net Yaroze software on the system.

- User Guide

A reference manual providing details on making software for the NetYaroze system.

- Library Reference

A manual listing and describing the functions and structures in the NetYaroze libraries.

### Additional Reading

See Additional Reading at the back of this manual.



## Introduction

---

The Net Yaroze Starter Kit is an integrated environment for developing PlayStation software on your personal computer. Work developed on a PC can be played by downloading to a special PlayStation, available only to Members of Net Yaroze.

Members can then share their creations and knowledge with other Members via a unique Members only Web site server provided by Sony Computer Entertainment





## Members OnlyWeb Site

---

### Connection to the Members OnlyWeb Site

Members are provided with access to one of three NetYaroze servers, the URL being:

<a href="http://www.scei.co.jp/Net/">http://www.scei.co.jp/Net/</a>	for Members in Japan
<a href="http://www.scee.sony.co.uk/yaroze/">http://www.scee.sony.co.uk/yaroze/</a>	for Members in Europe
<a href="http://www.sony.com/Yaroze/">http://www.sony.com/Yaroze/</a>	for Members in North America

### Software Necessary for Connection

We recommend Netscape version 2.0 or later as the browser software necessary for connection to the Members only Web site.

### ID and Password

The ID and password for access to the Members only Web site will be provided with your Welcome Pack

### Being a Member of NetYaroze

A key part of Net Yaroze is the activity of its Members on their Web site. As well as sharing work that you have developed and enjoying the creations of other Members, the Members only Web site will offer a range of other activities.

### Further Information- Support

Sony Computer Entertainment also use the Members only Web site to provide additional technical information or data that is not included in the documentation Please ask for information that is not provided.



## Contents of the Package

---

The following items are included in the Net Yaroze package:

1. DTL-H3001 NetYaroze Member's PlayStation or  
DTLH-3002 NetYaroze Member's PlayStation  
(Power cable and AV (RFU) cable included)
2. DTL-H3010 Controllers x 2
3. DTL-H3020 Access card
4. DTL-S3035 NetYaroze boot disk
5. DTL-S3045 NetYaroze software development disk
6. DTL-H3050 Communications cable
7. DTL-D3065 Start Up Guide (this manual)
8. DTL-D3075 User Guide
9. DTL-D3095 Library Reference

### Notes

Please note that the model numbers of the package you receive may vary slightly depending on which of the three Net Yaroze Members' regions that you live in.

'Net Yaroze' is the registered trademark of the project. However, it is also referred to as 'Yaroze'.



## Operating Environment

---

You need the following hardware and software in order to use the NetYaroze development environment

### Hardware

A DOS/V compatible PC/AT with a 486DX2 66MHz CPU or faster

It should have one or more serial ports, and a modem (at least a 14,400bps modem) for connection to the Yaroze Web site

- **Hard Disk**

At least 10MB of free disk space in order to set up the basic development environment

- **Memory**

At least 4MB RAM

- **CD-ROM Drive**

- **Display**

For the PC

SVGA monitor

For the PlayStation

A standard TV monitor with a video input terminal  
(ideally with multi format capabilities)

- **Mouse**

### Software

MS-DOS Version 5.0 or later

Windows 3.1 or Windows 95



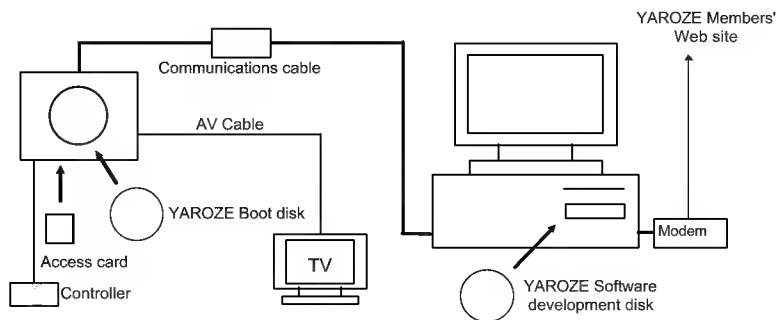
## Set Up

### Hardware Connection

Make sure that the PC and Net Yaroze PlayStation power switches are turned off. After connecting the serial port of the PC and the PlayStation with the communication cable, as shown in the diagram below, place the Access card in the PlayStation's Memory card slot 1.

Next, connect the PlayStation to the TV monitor with the supplied AV cable.

In order to access the Net Yaroze Members only Web site via the Internet, you need a separate modem and telephone line connection, a contract with an Internet service provider and an InternetWeb browser installed on your PC.



## Software Set Up

For the PlayStation

No special set up operation is necessary. The integrated environment is loaded automatically when you put the Net Yaroze boot disk in the Net Yaroze PlayStation, the access card in Memory card slot 1 and switch the power on. The following screen is displayed on the TV monitor.





For the PC

### Folder Copy

Place the enclosed Net Yaroze software development disc (hereafter 'CD-ROM') in the PC CD-ROM drive and copy the contents of each folder to the hard disk using a DOS XCOPY or Windows Program Manager/Explorer. (At the root of CD-ROM there are two folders called PSX and GNU. You can copy these folders anywhere on the hard disk.) In the example below, the PSX and GNU folders are copied onto the C drive root.

### Rewriting config.sys

Check the contents of config.sys in the root of c drive and confirm whether or not ansi.sys is included. If it is not included, add the following line and reboot the system.

```
devicehigh=C:\WINDOWS\ansi.sys
```

### Rewriting the Batch File

There is a file called djsetup.bat in the PSX folder which you have copied to the hard disk. Its contents are as follows.

```
@echo off
set DJGPP=c:/psx/djgpp.env
PATH %path%;c:\psx\bin;c:\gnu\bin
set TMP=c:\tmp
set DTLH3000=0x3f8,4,9600
```

If the psx and gnu folders are not on c drive root, amend this file so the contents correspond to the location and names (if you have changed these) of the copied folders. This procedure is described below.

## Note

Because the file is copied from the CD-ROM, the file is read only. Before editing change this in DOS as below.

```
C:\PSX> attrib -r djsetup.bat (where 'c:\PSX' is the path)
```

- The second line is the set up of the environment variable DJGPP. The compiler's environment setting file is specified. If appropriate, modify the c:/psx part so it refers to the correct path and name of the copied folder. Note that you must use a "/" (forward slash) to delimit the path for this environment variable only.
- The third line is the setting of the environment variable PATH. Rewrite the c:/psx and c:/gnu part to the name and path of the copied folder, if appropriate.
- The fourth line is the working folder. Create and allocate a folder if c:/tmp is not appropriate. For example, change the line to:

```
set tmp=c:\yaroze\wrk (where 'wrk' is your preferred working folder)
```

- The fifth line is the setting relating to communication with the NetYaroze PlayStation. Specify the port address, IRQ and speed (in this order) separated by the defaults, 0X3F8 and IRQ4, are the general address and IRQ of COM1.
- Specify 0X2F8 as the address and IRQ3 if you use COM2.
- Communication speeds of up to 115200bps are supported. Specify either 9600, 19200, 38400, 57600 or 115200 according to the processing capacity of your PC.
- (Please note that if you wish to use speeds other than 9600 baud, you should place a standard formatted PlayStation Memory card in your Net Yaroze PlayStation's Memory card Slot 2 and change the rate using SIOCONS- See chapter 17 of the Net Yaroze User Guide for details of SIOCONS)

### Execution of the Environment Setting Batch File

From the PSX folder (or your chosen folder name) in DOS (or the DOS box from Windows) execute the rewritten batch file `djsetup.bat`, as below.

```
C:\PSX>djsetup (where 'PSX' is your chosen folder name)
```

This completes the preparation of the development environment



## Getting Started

---

There are some sample and quick start programs which you can use to test your NetYaroze system set up and get started on compiling and running some programs.

Programmers with a lot of experience, may wish to go straight to the Sample Program section of this chapter, which skips a lot of explanation. The Quick Start section covers the same process, but gives a little more detail as it steps through NetYaroze software development.

### Quick Start

The 'quick' directory on the NetYaroze PC CD holds some quick start test programs. You probably copied it to your PC's hard disk together with all the NetYaroze software.

Using the quick start programs you can quickly check that everything is working on your NetYaroze system.

#### NetYaroze Program Development - Overview

Here are the basic steps that you need to take to develop software on the NetYaroze system. (See the NetYaroze User Guide, chapter 13 for more details on the development cycle.)

1. Create/edit text files which contain the C code.

Also known as the 'source code', these are usually named `name.c` (where 'name' is the selected file name).

(There are two source code files already in the quick directory: `tuto0.c` and `tuto1.c`.)

2. Compile and link the C files to create the executable file (the file that is the finished program/application).

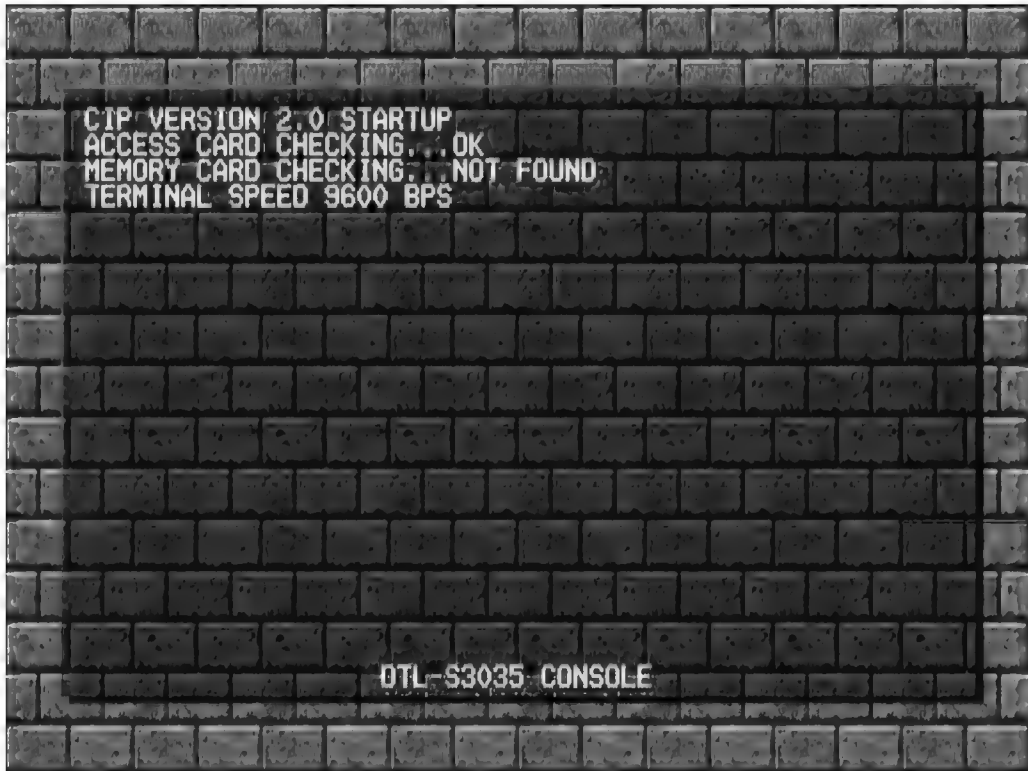
(In the test example, below, we use `make` file, called by the command 'make' to do this.)

3. Use the SIOCONS console tool, provided as part of your NetYaroze software, to establish contact between your NetYaroze PlayStation and PC.
4. Run the executable program.

5. (In this test example, there are two batch files - batch0 and batch1 - which you can use to run the executable files (or 'executables') of the test programs.)

#### Compile and Run the Quick Start Programs - Step by Step

1. Set up your NetYaroze PlayStation as described in the [Start Up Guide](#) link the Net Yaroze PlayStation to your TV and PC, and insert a Controller in the PlayStation's Controller port 1.
2. Turn on the PC, TV and NetYaroze PlayStation and wait for the ready-to-go screen to appear on the TV, as shown below.



This screen should appear within 30 seconds of turning on or resetting the NeYaroze PlayStation, after the Sony and PlayStation logo screens.

1. On the PC, bring up an MS-DOS box and, from the PSX folder (or your chosen NetYaroze folder name), invoke the NetYaroze environment setting batch file as follows:

```
c:\PSX>djsetup (where 'PSX' is your chosen NetYaroze folder name)
```

2. Compile the quick start programs. Use the 'make' utility to do this by typing 'make' to the DOS prompt to create the executable programs from the source files.

```
c:\psx\quick>make
```

3. Bring up the console tool, SIOCONS. (You must have set up the quick directory as a path for SIOCONS in your autoexec.bat file.)

```
c:\psx\quick>siocons
```

or, if you have already changed the baud rate, type

```
c:\psx\quick>sicons -B<baud rate> (where '<baud rate>' is the value of your chosen rate)
```

4. Now you are in SIOCONS, the console tool used to communicate with the NetYaroze PlayStation from the PC.

The PC monitor output should look similar to this:

```
c:\psx\quick>siocons -B115200
siocons -- PlayStation debug system console prog
for DTLH3000 1996/05/10 00:00:03
type F1 ----> display help
when hung up try type ESC
I/O addr = 0x03F8, IRQ=4 (vect=0x000C,8259=20)
BAUDRATE = 115200
```

5. Check that SIOCONS is working correctly by pressing return: if all is well, the SIOCONS prompt will appear, which is a double-right arrow:>>

6. Run the quick start batch files.

First, press [F3] on the PC keyboard. This brings up the `Auto[1] :` prompt. Then enter the name of the batch file; **batch0**.

For example (after pressing, [F3] on the PC keyboard):

```
Auto[1]:batch0
```

Batch0 downloads and runs tuto0.exe.

7. Immediately after starting the download, downloading messages appear on the PC monitor, similar to the following.

```
tuto0 [ .text] address:80100000-801001ef size:0001f0 0001f0: 1sec.  
tuto0 [ .rdata] address:801001f0-8010020f size:000020 000020: 1sec.  
tuto0 [ .data] address:80100210-8010035f size:000150 000150: 2sec.  
tuto0 [ .sdata] address:80100360-801003df size:000080 000080: 2sec.
```

8. The TV screen shows downloading messages, similar to those below:

```
BINARY DATA TRANSFER - BWR  
ADRS = 80100000  
SIZE = 000001F0 BYTES  
INTO BURST TRANSFER SUM:111F0/1F0 BYTES  
DONE.
```

#### Running and Terminating a Program

Tuto0 (called by batch0) prints 'Hello World!' to the SIOCONS console on the PC, and terminates automatically.

When you have run and terminated a program, the Net Yaroze PlayStation resets itself, going back to the ready-to-download screen (as shown in [2] of Compile and Run the Test Programs - Step by Step, above, and SIOCONS displays its prompt again (>>).

Once the PlayStation and PC are reset, you can run the second quick start program, tuto1, by following the steps described in [8] and [9] of Compile and Run the Test Programs - Step by Step above.



Tuto1 (called by batch1) prints 'Hello World!' to the TV screen. Terminate this program by pressing SELECT on the Controller.

#### Exiting SIOCONS

To quit SIOCONS and return to the MS-DOS prompt, on the PC keyboard, press [F10] and then [F2], or just press [Esc].

#### Using the Test Files

The source files, makefile and batch files are all simple text files which you can easily edit (Using the text editor in DOS, for example) to see their contents.

To edit or look at the tuto0.c source code, using the text editor in DOS, type:

```
c:\>psx\quick>textedit tuto0.c
```

Look at the test and other sample code to familiarise yourself with what the NetYaroze system can do, and how to achieve this by writing C programs.

#### Quick Directory Contents

Type	Name
C files for the sample programs	tuto0.c
	tuto1.c
	pad.c
Makefile	makefile
Batch files	batch0 (downloads and runs tuto0)
	batch1 (downloads and runs tuto1)
Subdirectories	
make	a help file on the program updating utility 'make' and some example makefiles
batch	a help file on batch (autoexecution) files and some example batch files
baudrate	a help file on changing the baud rate (rate of data downloading from the PC)

## Sample Program

In order to check your environment, try downloading the sample program provided. (The source code and executable file are in the `psx\sample\check` folder- where 'psx' is your chosen file name)

Download the sample code as follows:

1. Execute Make

Execute make at the DOS prompt to test the paths listed in `djsetup.bat`, and ensure that the program is compiled and linked correctly, as shown below (the text in bold indicates command line input).

```
C:\PSX\SAMPLE\CHECK>make
gcc -O1 -g -c main.c -o main.o
gcc -Xlinker -Ttext -Xlinker 80140000 -o main main.o
```

2. Run SIOCONS

When the program is compiled and linked, run the console monitor, SIOCONS as shown below.

```
C:\PSX\SAMPLE\CHECK>siocons
siocons -- PlayStation debug system console program
for DTLH3000 1996/05/10 00:00:00
type F1 ----> display help
when hung up try type Ctrl+C
I/O addr = 0x03F8, IRQ=4{vect=0x000C,8259=20}
BAUDRATE = 9600
```

3. Auto download

Hit the F3 function key to obtain the `Auto[1]` prompt. Input **auto** to indicate automatic download and press the enter key. (See below.)

```
Auto[1]: auto
main [ .text] address:80140000-80140b4f size:000b50 000b50: 1sec.
main [ .rdata] address:80140b50-80140c7f size:000130 000130: 1sec.
main [ .data] address:80140c80-801412ef size:000670 000670: 2sec.
main [ .sdata] address:801412f0-8014137f size:000090 000090: 2sec.
PC=80140990, GP=801492f0, SP=801ffff0

>>go
ResetGraph:jtb=80047dd0,env=80047e18
```

With the operation so far, the sample program window should be displayed on the TV monitor screen.

The Controller button operation is as follows.

Up key	Increases number of balls displayed
Down key	Decreases number of balls displayed
L1 button	Display pause
SELECT button	Terminate



## Sample Program

---

### Sample Program Make File

```
CFLAGS    = -O1 -g
LINKER    = -Xlinker -Ttext -Xlinker 80140000
RM        = del

PROG      = main
OBJS      = main.o

all: $(PROG)

$(PROG): $(OBJS)
        $(CC) $(LINKER) -o $@ $?

main.o: main.c

clean:
        $(RM) $(PROG)
        $(RM) $(OBJS)
```

### Sample Program Source List

```
/*
 *   Draw bouncing ball pattern on screen
 *   Increase/decrease ball pattern number by up/down key
 *   Read sound data from boot disk and play back as BGM
 *
 *   Copyright (C) 1996 Sony Computer Entertainment Inc.
 *   All Rights Reserved
 */

/*Preparation include header */
#include <libps.h>
```

```

#include "pad.h"

/*-----
        Macros
-----*/

/* Basic setup */
#define KANJI                /* Switch to display Kanji*/
#define OT_LENGTH            1    /* Ordering table number*/
#define MAXOBJ               1500 /* Sprites {balls} maximum number*/
#define MVOL                 127  /* Main volume level*/
#define SVOL                 127  /* SEQ volume level*/
#define DFILE                3    /* File number*/

/* Macros relating to data arrangement */
#define VH_ADDR              0x80090000
#define VB_ADDR              0x800a0000
#define SEQ_ADDR             0x80110000

/* Macros relating to display area */
#define FRAME_X              320   /* Display area size(horizontal) */
#define FRAME_Y              240   /* Display area size(vertical)*/
#define WALL_X               (FRAME_X-16) /* Ball pattern movement area size
                                         (horizontal) */
#define WALL_Y               (FRAME_Y-16) /* Ball pattern movement area size
                                         (vertical) */

/* Range check macro */
#define limitRange(x, l, h)    ((x)={{(x)<(l)?(l):(x)>(h)?(h):(x))}}

/*-----
        External variables
-----*/

/* Ordering table related variables */
GsOT            WorldOT[2];
GsOT_TAG        OTTags[2][1<<OT_LENGTH];

/* Primitive related variables */
PACKET GpuPacketArea[2][MAXOBJ*(20+4)];
GsSPRITE        sprt[MAXOBJ];

/* Ball pattern movement related variables */
typedef struct {

```

```

        u_short x, y;                /* Location */
        u_short dx, dy;              /* Speed */
    } POS;

/* Controller related variables */
    volatile u_char *bb0, *bb1;

/* File reading related variables */
typedef struct {
    char            *fname;
    void            *addr;
    CdlFILE finfo;
} FILE_INFO;

static FILE_INFO dfile[DFILE] = {
    {"\\DATA\\SOUND\\STD0.VH;1", (void *)VH_ADDR, 0},
    {"\\DATA\\SOUND\\STD0.VB;1", (void *)VB_ADDR, 0},
    {"\\DATA\\SOUND\\SAMPLE1.SEQ;1", (void *)SEQ_ADDR, 0},
};

/* Sound related variables */
short vab, seq;

/*-----
    Static functions template
-----*/
static void init_prim();                /* Ball pattern graphics related
initialisation */
static void init_point(POS *pos);       /* Ball pattern movement related
initialisation */
static long pad_read(long n);           /* Controller state analysis */
static u_long PadRead(long id);         /* Controller state reading */
static void datafile_search();          /* File retrieval on CD-ROM */
static void datafile_read();            /* File reading on CD-ROM */
static void init_sound();               /* Sound data on memory playback
preparation */
static void play_sound();               /* Sound playback start */
static void stop_sound();               /* Sound playback termination */

/*-----
    Main functions
-----*/

```

```

main()
{
    int      nobj = 1;          /* Number of ball patterns displayed
(from 1)*/
    GsOT      *ot;              /* Pointer to drawing OT */
    int      i, cnt, x, y;      /* Working variables*/
    int      activeBuff;
    GsSPRITE  *sp;
    POS      pos[MAXOBJ];
    POS      *pp;

    SetVideoMode( MODE_PAL );    /* PAL Mode */
/* SetVideoMode( MODE_NTSC );    /* NTSC Mode */

    GetPadBuf(&bb0, &bb1);      /* Get controller reception buffer */
    datafile_search();           /* Data file retrieval */
    datafile_read();             /* Data file reading */

    GsInitGraph(320, 240, 4, 0, 0); /* Drawing and display
environment setting */
    GsDefDispBuff(0, 0, 0, 240);    /* Same as above */
/*
    When drawing in {0,0}-{320,240}, display {0,240}-{320,480}{db[0]}
    When drawing in {0,240}-{320,480}, display {0, 0}-{320,240}{db[0]}
*/

    /* Ordering table information setting */
    for (i = 0; i < 2; i++) {
        WorldOT[i].length = OT_LENGTH;
        WorldOT[i].org = OTTags[i];
    }
    /* Font setting */
#ifdef KANJI                      /* In the case of Kanji display */
    KanjiFntOpen(160, 16, 256, 240, 704, 0, 768, 256, 0, 512);
#endif
    EntLoad(960, 256);
/* Load basic font pattern in frame buffer */
    EntOpen(16, 16, 256, 200, 0, 512);
/* Font display location setting */

    init_prim();                  /* Primitive buffer initial setting */

```



```

init_point(pos);          /* Ball pattern movement initial setting */
init_sound();             /* Sound initial setting */
play_sound();             /* Sound playback start */

/* Main loop */
while ((nobj = pad_read(nobj)) > 0) {
    /* Double buffer switch */
    activeBuff = GsGetActiveBuff();
    GsSetWorkBase({PACKET *}GpuPacketArea[activeBuff]);

    /* Ordering table clear */
    GsClearOt(0, 0, &WorldOT[activeBuff]);

    /* Ball pattern location update and registration to OT */
    sp = sprt;    pp = pos;
    for (i = 0; i < nobj; i++, sp++, pp++) {
        /* Horizontal coordinate value update */
        if ((x = (pp->x += pp->dx) % WALL_X*2) >= WALL_X)
            x = WALL_X*2 - x;
        /* Vertical coordinate value update */
        if ((y = (pp->y += pp->dy) % WALL_Y*2) >= WALL_Y)
            y = WALL_Y*2 - y;

        /* Set new coordinate value for sprite primitive */
        sp->x = x;    sp->y = y;
        /* Registration to sprite primitive ordering table */
        GsSortFastSprite(sp, &WorldOT[activeBuff], 0);
    }

    DrawSync(0);          /* Waiting for end of drawing */
    cnt = VSync(0);       /* Waiting for vertical synchronisation
                           interrupt */
    GsSwapDispBuff();     /* Double buffer switching */

    /* Registration to screen clear primitive ordering table */
    GsSortClear(60, 120, 120, &WorldOT[activeBuff]);

    /* Drawing of primitive registered in OT */
    GsDrawOt(&WorldOT[activeBuff]);

    /* Printing number of balls and elapsed time */

```

```

#ifdef KANJI
    KanjiFntPrint("Num  =%d\n", nobj);
    KanjiFntPrint("Time =%d\n", cnt);
    KanjiFntFlush(-1);
#endif

    FntPrint("sprite = %d\n", nobj);
    FntPrint("total time = %d\n\n", cnt);
    FntPrint("UP      : INCREASE\n");
    FntPrint("DOWN    : DECREASE\n");
    FntPrint("L1      : PAUSE\n");
    FntPrint("SELECT: END\n");
    FntFlush(-1);
}      /* Main loop terminal */

/* Execute this by pressing select button and verifying */
stop_sound();      /* Sound playback termination */
return(0);          /* Program termination */
}

/*-----
    Ball pattern initialisation
-----*/
#include "balltex.h"      /* Ball pattern texture pattern */

/* Ball pattern graphics related initialisation */
static void init_prim()
{
    GsSPRITE      *sp;
    u_short tpage;
    RECT          rect;
    int           i;

    rect.x = 640;  rect.y = 0;
    rect.w = 16/4; rect.h = 16;
    LoadImage(&rect, ball16x16);
    tpage = GetTPage(0, 0, 640, 0);

    for (i = 0; i < 32; i++) {
        rect.x = 0;    rect.y = 480+i;
        rect.w = 256;  rect.h = 1;
    }
}

```

```

        LoadImage(&rect, ballcolor[i]);
    }

    /* Sprite initialisation */
    for (sp = sprt, i = 0; i < MAXOBJ; i++, sp++) {
        sp->attribute = 0;
        sp->x = 0;
        sp->y = 0;
        sp->w = 16;

        sp->h = 16;
        sp->tpage = tpage;
        sp->u = 0;
        sp->v = 0;
        sp->cx = 0;
        sp->cy = 480+(i%32);
        sp->r = sp->g = sp->b = 0x80;
        sp->mx = 0;
        sp->my = 0;
        sp->scalex = ONE;
        sp->scaley = ONE;
        sp->rotate = 0;
    }
}

/* Ball pattern movement related initialisation */
static void init_point(POS *pos)
{
    int    i;
    for (i = 0; i < MAXOBJ; i++) {
        pos->x = rand();

        /* Start coordinate X */

        pos->y = rand();

        /* Start coordinate Y */

        pos->dx = (rand() % 4) + 1;
        /* Movement distance X 1<=x<=4 */

        pos->dy = (rand() % 4) + 1;
        /* Movement distance Y 1<=y<=4 */

        pos++;
    }
}

```

```

/*-----
    Reading and analysis of controller state
-----*/
/* Analysis of controller state */
/*    Return value
    -1 : At time of pressing select button and verifying
    1st argument + 4 : At time of pressing up button and verifying
    1st argument - 4 : At time of pressing down button and verifying
    Pause in function while pressing L1 key */
static long pad_read(long n)
{
    u_long padd = PadRead(1);          /* Controller reading */

    if (padd & PADLup)    n += 4; /* Left cross key up */
    if (padd & PADLdown)  n -= 4; /* Left cross key down */

    if (padd & PADL1)
        while (PadRead(1)&PADL1);      /* Pause */

    if (padd & PADselect)
        return(-1);                    /* Program termination */
    limitRange(n, 1, MAXOBJ-1);        /* n is given value 1<=n<=(MAXOBJ-1) */
    return(n);
}

/* Controller state reading */
static u_long PadRead(long id)
{
    return(~(*{bb0+3} | *(bb0+2) << 8 | *(bb1+3) << 16 | *(bb1+2) <<
24));
}

/*-----
    Reading the file on CD-ROM (DFILE)
-----*/
/* Retrieving file on CD-ROM */
static void datafile_search()
{
    int i, j;

```

```

        for (i = 0; i < DFILE; i++) {          /* Deal with DFILE file */
            for (j = 0; j < 10; j++) {          /* Return loop */
                if (CdSearchFile(&{dfile[i].finfo), dfile[i].fname) !=
0)
                    break;
                /* Retry loop interruption on normal termination */
                else
                    printf("%s not found.\n", dfile[i].fname);
            }
        }

    }

/* CD-ROM file reading */
static void datafile_read()
{
    int i, j;
    int cnt;

    for (i = 0; i < DFILE; i++) {          /* Deal with DFILE file */
        for (j = 0; j < 10; j++) {          /* Retry loop */

CdReadFile(dfile[i].fname,dfile[i].addr,dfile[i].finfo.size);

/* Normal processing can be executed by other side of read */
/* Here, remaining sector number is monitored until Read terminated */

            while ((cnt = CdReadSync(1, 0)) > 0 )
                VSync(0);                  /* Waiting for vertical
                                            synchronisation
                                            interrupt (for time
                                            adjustment */

            if (cnt == 0)
                break; /* Retry loop interruption on normal
                        termination*/
        }
    }

}

/*-----
Sound related

```

```

-----*/
/* Sound data on memory playback preparation */
static void init_sound()
{
    /* VAB opening and transmission to sound buffer */
    vab = SsVabTransfer( (u_char*)VH_ADDR, (u_char*)VB_ADDR, -1, 1 );
    if (vab < 0) {
        printf("SsVabTransfer failed (%d)\n", vab);
        return;
    }

    /* SEQ opening */
    seq = SsSeqOpen((u_long *)SEQ_ADDR, vab);
    if (seq < 0)
        printf("SsSeqOpen failed (%d)\n", seq);
}

/* Sound playback start */
static void play_sound()
{
    SsSetMVol(MVOL, MVOL);           /* Main volume setting*/
    SsSeqSetVol(seq, SVOL, SVOL);    /* Volume setting for each SEQ */
    SsSeqPlay(seq, SSPLAY_PLAY, SSPLAY_INFINITY); /* Playback switch ON */
}

/* Sound playback termination */
static void stop_sound()
{
    SsSeqStop(seq);                  /* Playback switch OFF */
    VSync(0);
    VSync(0);
    SsSeqClose(seq);                 /* SEQ close */
    SsVabClose(vab);                 /* VAB close */
}

/* Source code termination */

```



## Additional Reading

---

The following books may assist you to make NetLogo applications.

### C Programming

Title	Publisher	ISBN
The C Programming Language	Prentice Hall	ISBN 0-13-110362-8
Teach Yourself C	McGraw-Hill	ISBN 0-07-882011-1
Programming with GNU Software	O Reilly	ISBN 0-56592-112-7


### Graphics

Title	Publisher	ISBN
3D Computer Graphics	O Reilly	ISBN 0-201-63186-5
Encyclopedia of Graphic File Formats	Addison-Wesley	ISBN 1-56592-058-9



## Start Up Guide

### Software Development Tool

- This product is sold on a membership agreement basis to Members of NetYaroze, which is operated by Sony Computer Entertainment Inc.
- The  symbol, PlayStation' and 'NetYaroze' are trademarks of Sony Computer Entertainment Inc.
- Company and product names recorded in/on this product are generally trademarks of each company. Note that in/on this product the symbols '®' and 'TM' are not used explicitly.

Published February 1997

©1997 Sony Computer Entertainment Inc. All Rights Reserved.

Written and produced by :

Sony Computer Entertainment Inc.

Akasaka Oji Building

8-1-22 Akasaka, Minato-ku, Tokyo, Japan 107

Enquiries to: Network Business Project

E-mail: ny-info@scei.co.jp

TEL: +81 (0) 3-3475-1711

Sony Computer Entertainment Europe

Waverley House

7-12 Noel Street

London W1V 4HH, England

Inquiries to: TheYaroze Team

E-mail: yaroze-info@scee.sony.co.uk

TEL: +44 (0) 171 447 1616 / +44 (0) 7000 YAROZE

Sony Computer Entertainment America

919 E. Hillsdale Blvd., 2nd Floor

Foster City, CA 94404, USA

Inquiries to: TheYaroze Team

E-mail: yaroze@interactive.sony.com

TEL: +1-415-655-3600