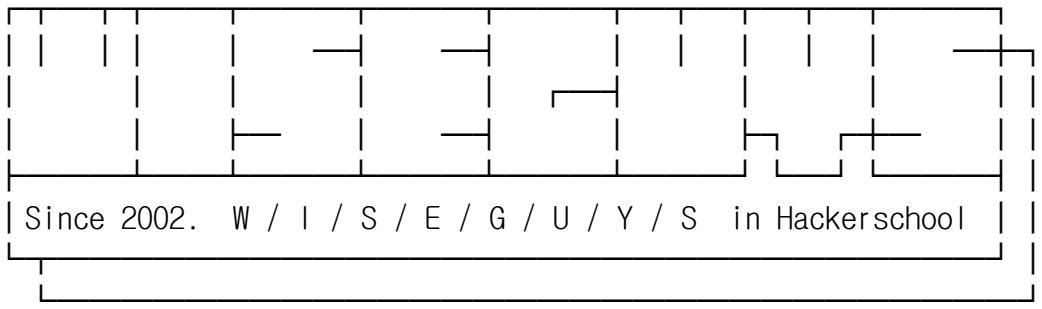


—[Document Information] ————————————————————————

:: Title :: NDS Homebrew Development

:: Date :: 2008. 4. 21.

:: Author :: zOnKT1g3r

:: Contact:: E-Mail(wantstar@hotmail.com)

——

—[Notice]————————————————————————————————————

본 문서의 잘못된 부분이나 지적이나 추가하고 싶은 내용은
저자에게 메일을 보내기 바랍니다.

——

—[Index] ————————————————————————————————

——

0x00.Introduction: What is Nintendo DS Homebrew?

닌텐도 DS 는(이하 NDS) 닌텐도사에서 나온 미니 포터블 게임기로서, PSP 와 견줄만한 인기로 비디오 게임 시장을 누리고있습니다. 그 인기를 이어 닌텐도 DS Lite 또한 출시되었습니다.NDS 는 게임 카트리지를 위쪽에 삽입하여 게임을 실행하는 방식으로, 게임 카트리지를 구입하지않는 이상 불법으로 유통되는 ROM 파일들을 사용할 수 없게되어있지만, 각종 mod chip 들로 인해 게임기의 무한한 가능성이 발견되었습니다. 갸중에는 R4DS, DSTT, G6DS 등등이 존재합니다. 이러한 mod chip 들은 microSD 메모리 칩을 내부에 삽입하여 mod chip 이 돌아가는 운영체제와 ROM 파일들을 담아둘 수 있는 메모리로 사용합니다. 불법으로 유통되는 ROM 파일들은 보통 nds 의 확장자를 가지고있습니다. (예. 1011 - Lost In Blue(U).nds) 이러한 nds 확장자를 가진 파일들을 microSD 메모리칩에 넣어두면, NDS Slot-1 카트리지 삽입구에 mod chip 을 삽입할 시, 메모리에 들어가있는 파일들을 실행할 수 있게됩니다. 본 문서에서는 microSD 메모리 칩에 들어가는 nds 파일을 개발 및 제작하는 방법을 설명합니다. NDS 프로그램 개발을 NDS Homebrew 라고 칭합니다. Homebrew 란 사전적인 의미로 '집에서 직접 만든 제품'을 뜻합니다. 허나 국내에는 NDS Homebrew 개발자가 많이 없기에 그에 따른 자료또한 많이 모자란 편입니다. 제가 NDS Homebrew 를 공부해나가는 과정을 문서화 하고싶었던 이유이기도합니다. 이 문서가 많은 양의 자료를 포함할 순 없을지라도, 조금이라마 국내 개발자들에게 도움이 되었으면 하는 바램에 이 문서를 작성했습니다. NDS Homebrew 제작시 이미 개발되어있는 개발툴과 라이브러리, devKitPro (http://www.devkitpro.org) 와 PA Library (http://forum.palib.info) 를 사용합니다. 또한 제작된 nds 파일을 시험하기 위해 DeSmuME (http://www.desmume.org) 에뮬레이터를 사용하게됩니다.

0x01. Then, What is devKitPro?

devKitPro 는 NDS 를 포함한 이외의 여러 게임 콘솔(PSP, Gamecube 등)의 프로그램 제작을 위해 개발된 툴체인인데요, 우리가 NDS Homebrew 를 위해 사용하게될 툴킷은 바로 devKitARM 입니다. NDS 콘솔의 CPU 가 ARM 인것이죠. 게임 콘솔에서 터치패드를 가능케하는 유일한 CPU 가 ARM 이라고 합니다. devKitARM 개발툴은 아래의 주소에서 내려받을 수 있습니다.

devkitPro:
http://sourceforge.net/project/showfiles.php?group_id=114505&package_id=124207

0x02. Then, What is PA Lib?

PA Lib 이란 Programmer's Arsenal Library 의 약자로서 NDS Homebrew 개발을 훨씬 수월하게 도와주는 라이브러리입니다. 본 문서에서 NDS Homebrew 개발을 할때 devKitARM 위에 PA Lib 을 이용하여 개발 환경을 구축하게됩니다.
아래의 주소에서 내려받으시면됩니다.

PA Lib:
http://sourceforge.net/project/showfiles.php?group_id=142901&package_id=168612

0x03. Setting up the development environment

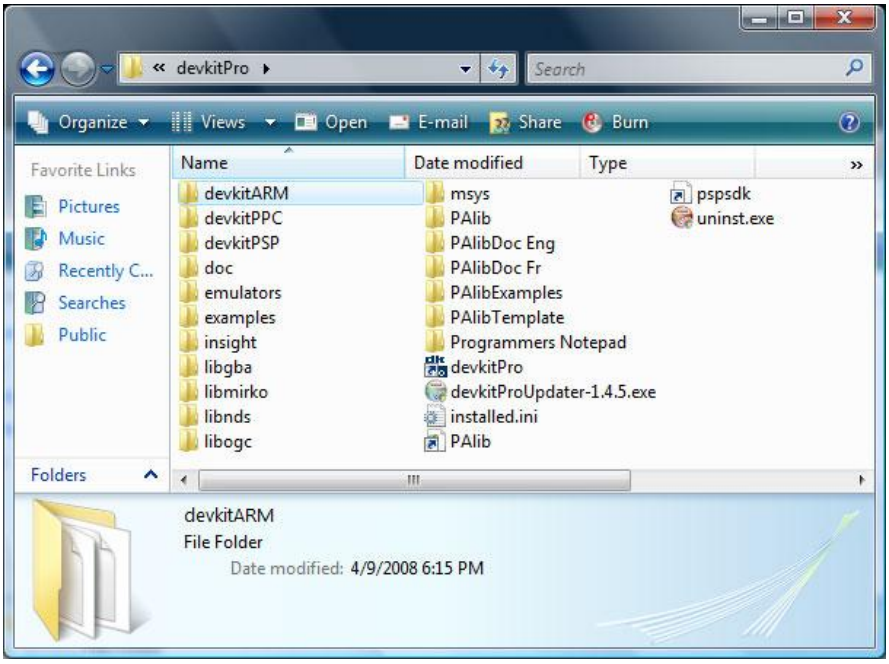
위의 주소에서 devkitARM_r20-win32.exe 파일을 받아 설치하면, 드라이브 최상위에 devkitPro 라는 디렉토리가 생깁니다. 그 후 PA Library 를 설치해야하는데, PA Library 는 .Net Framework 를 필요로합니다. 혹 없으신 분은, 아래의 주소에서 받아 설치하시면 됩니다.

.Net Framework:
http://www.microsoft.com/downloads/details.aspx?FamilyID=0856eacb-4362-4b0d-8edd-aab15c5e04f5&DisplayLang=en

위의 PA Library 주소에서 PAlib070717.exe 파일을 받아 설치하시면, 자동적으로 devkitPro 폴더에 PA Library 가 설치됩니다.

*주의하실점은 최신 PA Library 가 아직까지는 devkitARM Release 21 를 지원하지 않습니다. 그러므로, PA Library 웹사이트를 확인하시고, 아직까지 지원하지않는다면, devkitARM Release 20 를 받아 설치하셔야합니다.
개인적으로 저 같은 경우는 최신 devkitPro Updater 를 받아 설치한 후, devkitPro 디렉토리내의 devkitARM 디렉토리만 Release 20 로 바꿔주었습니다.

위 파일들을 모두 설치시, 아래 화면과 같이 PA Library가 devkitPro 디렉토리 아래에 설치됩니다.



후 Visual Studio의 PA Library 개발 환경을 만들기 위해서, 아래의 파일을 내려받아 설치한 후, 윈도우의 환경변수와 Visual Studio 의 설정을 바꾸어야합니다. 윈도우 시스템 변수 PATH 뒤에 다음 경로들을 추가해주어야합니다.

C:₩devkitPro₩devkitARM₩bin;
C:₩devkitPro₩devkitARM₩arm-eabi₩bin;
C:₩devkitPro₩devkitARM₩libexec₩gcc₩arm-eabi₩4.1.1;

또한 시스템 변수에 DKP_HOME란 이름의 변수를 만들어야합니다. 변수의 값은 devkitPro가 설치되어있는 디렉토리로 설정합니다.

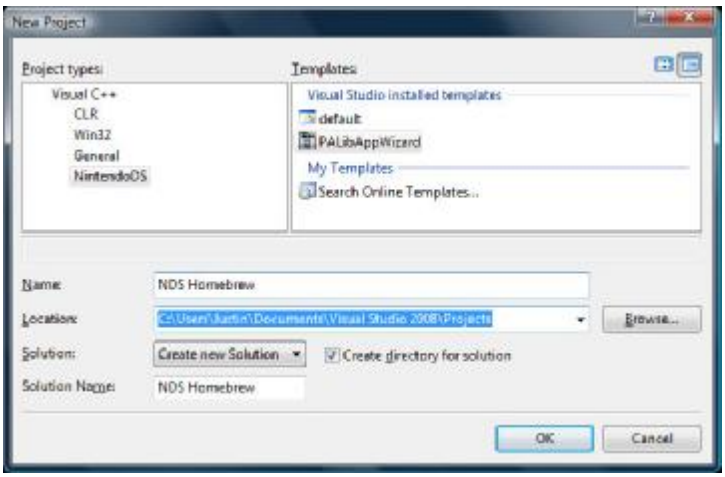
나머지 시스템 경로들은 인스톨러가 알아서 지정해둡니다. 다음으로, Visual Studio에서 PA Library 애플릿을 Application Wizard에 넣어두기 위해, 다음의 파일을 내려받아 설치합니다. 커맨트 프롬프트를 열어 해당 디렉토리로 이동한 후, 다음과 같이 명령어를 내려주어야합니다.

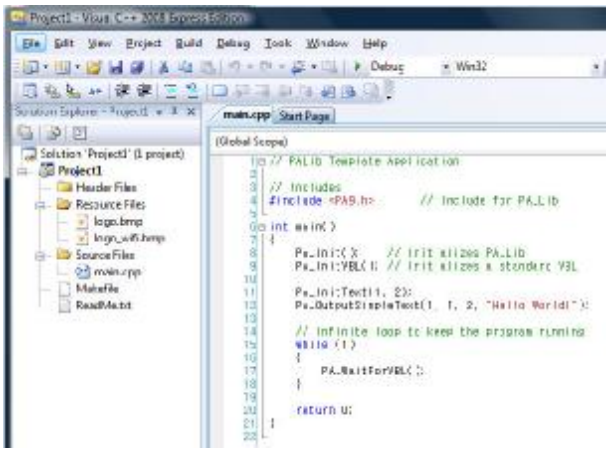
C:₩Users₩Justin₩Documents₩NDSL Homebrew>**wscript VC8_Express_Setup.js**

Visual Studio Pro 와 Visual Studio Express 의 구분, Visual Studio 2005 와 Visual Studio 2008 의 구분을 명확히 해야합니다. Visual Studio 2008 에 설치시, PALibAppWizard.vsz 안의 내용과, VC8 _Setup.js, VC8_Express_Setup.js 안의 "8.0" 그리고 "VC8"를 "9.0"과 "VC9"로 정정해줘야합니다.

윈도우 비스타 사용시, 수동으로 직접 파일을 복사해 해당 디렉토리에 넣어주어야합니다. 제 경우엔 C:\Program Files\Microsoft Visual Studio 9.0\VC\Express\VCProjects\NintendoDS 경로 안에 모든 파일을 넣어주고, PALibAppWizard.vsz 안의 내용을 수정했습니다.

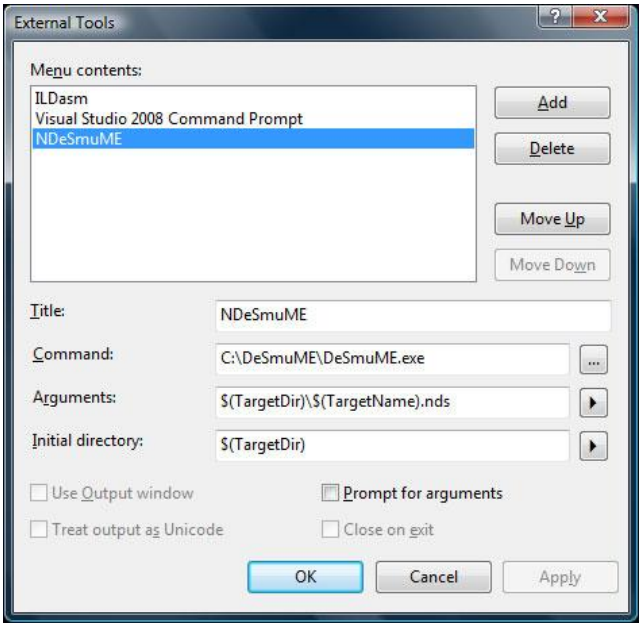
위의 설정이 끝나면 다음과 같은 화면이 우리를 반겨줍니다.





하지만 아직 바꿔야할 설정이 남아있습니다. 우리가 설치한 툴킷의 make 로 컴파일을 할때에 인자로 들어가는 경로가 너무 길거나 공백이 많을 경우 컴파일이 안되기에, Visual Studio 의 옵션의 General 탭 아래에서 프로젝트가 저장되는 경로를 짧게 바꿉니다. Tools->Options->Projects and Solutions->General 에서 Visual Studio projects location 을 C:₩VSProjects 와 같이 짧게 변경합니다. (프로젝트 이름 역시 공백을 포함하면 에러가 납니다).

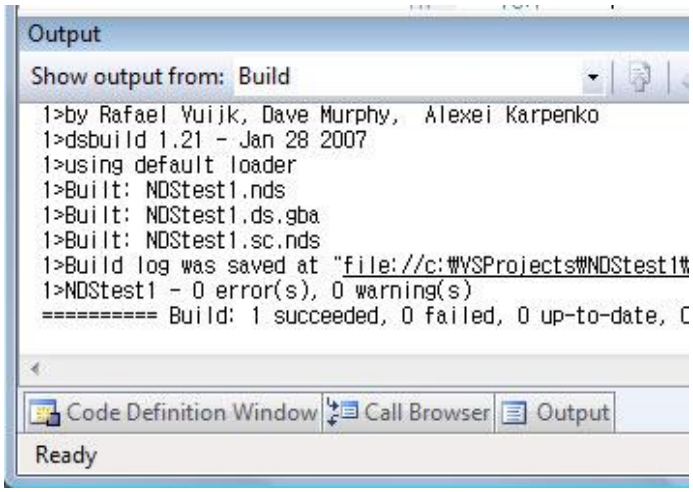
또한 컴파일시 바로 에뮬레이터와 연결해 실행하기 위해, Tools->External Tools을 열어 다음과 같이 설정해줍니다. 아래의 Command부분에는 에뮬레이터의 경로가 들어갑니다.

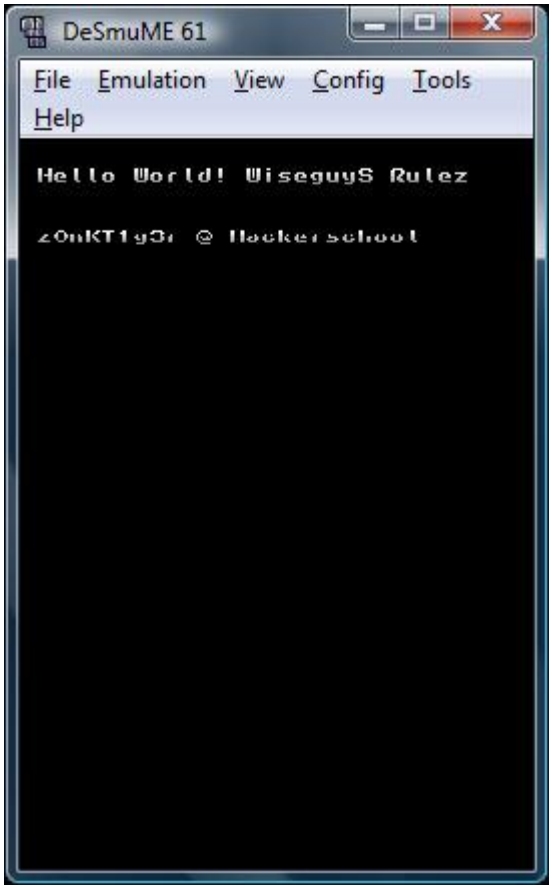


이로서 모든 설정이(삽질할 준비가) 완료되었습니다.

0x04. Hello NDS World?

길고 길었던 환경 설정이 제대로 된건지 테스트해보기 위해 컴파일을하여 빌드를한 후에 Tools 에 들어가 NDeSmuME 를 눌러 기본으로 템플렛에 들어가있던 코드를 실행해봅니다.





위의 화면을 보시면, 예제 코드(Hello World)가 성공적으로 실행되어 에뮬레이터에 로드된것을 확인하실 수 있습니다.

External Tools 를 미처 설정하지 못하셨다면, 에뮬레이터를 실행하여, 프로젝트 폴더에있는 nds 파일을 로드하셔서 실행시키면 됩니다.

0x05. Hello NDS World!

우선 예제 템플릿에 들어가있는 Hello World 의 소스코드를 분석해보겠습니다.

```
1: #include <PA9.h>

2: int main()
3: {
4:         PA_Init();
5:         PA_InitVBL();
6:        PA_InitText(1, 2);
7:        PA_OutputSimpleText(1, 1, 2, "Hello World! WiseguyS Rulez");
8:        PA_OutputSimpleText(1, 1, 6, "z0nKT1g3r @ Hackerschool");

9:        while (1)
11:       {
12:                PA_WaitForVBL();
13:       }

14:       return 0;
15: }
```

1 번 라인은 include 문인데요, 기본적인 PA Lib 를 포함시키는거구요, 4 번 라인은 PA_Init(); 함수 이름에서도 알 수 있다시피, PA Lib 를 시작하기위한 준비를 하는겁니다. 5 번 라인 PA_InitVBL(); 이 함수는 VBL 을 준비시키는건데요, VBL 의 자세한 자료는 찾지 못햇지만, VBL 비디오를 싱크로나이즈시켜주는 역할을 합니다. 12 번 라인의 PA_WaitForVBL 함수는 프로그램의 진행을 따라가지못하는 비디오 출력을 싱크시켜줍니다. 비디오 출력의 중요하고 또 필요한 함수입니다.

/* VBL 을 적용시킨 프로그램의 fps(Frame per second)가 60 을 웃돌때, 만약 12 번 라인의 PA_WaitForVBL 함수가 없었더라면, 해당 프로그램의 fps(Frame per second)는 아마 하늘을 찌르게 될겁니다. */

비디오 출력에 관한 기본적인 설정이되었다면, 그 위에 출력될 텍스트에 대한 설정을 해주어야합니다. 6 번 라인 PA_InitText(1,2);은 텍스트가 출력될 스페이스에 대한 초기화를 합니다. PA_InitText 함수의 첫번째 인자 1 은 스크린

넘버를 의미합니다. DS 스크린의 아랫화면은 0으로 지정되어있고, 윗화면은 1로 지정되어있습니다. 두번째 인자 2는 출력된 백그라운드의 넘버를 의미합니다.

PA Lib의 백그라운드는 0부터 3까지 총 4개가 존재하는데, 그 중 0이 최상위 백그라운드를 의미합니다. 만약 그림 A가 그림 B위에 출력되어야한다면 그림 A의 백그라운드를 그림 B의 백그라운드 위로 세팅해주어야합니다.

7번 라인 PA_OutputSimpleText(1, 1, 2, "Hello World! WiseguyS Rulez");는 함수 이름에서 알 수 있다시피, 텍스트를 드디어 화면에 출력해줍니다. 첫번째 인자 1은 스크린 넘버를 의미하구요, 두번째 인자와 세번째 인자는 각각 x, y 좌표를 의미합니다. 네번째 인자는 출력될 텍스트를 의미합니다.

06. Application of PA_lib

이 문서를 작성하고 있는 날 며칠뒤인 2008년 4월 16일이 여자친구와의 300일이라, 그때에 맞춰 이벤트성 게임 개발을 해주려고합니다. 이 문서의 작성을 시작한 이유 역시 300일을 기념하기위해 NDS를 즐기는 여자친구에게 선물을하기 위해서였습니다. 보통 여자친구와 심심할때마다 즉석복권을 사서 긁는데, 간단한 즉석복권 게임 프로그램을 만들어 서프라이즈를주고 싶었습니다. 그래서 아래는 즉석복권 게임을 만드는 과정을 설명하고자 합니다.

게임은 메인화면과 프로그램 설명 및 사용법, 그리고 랜덤 순서의 즉석복권입니다. 화면을 터치하는 곳에 있는 상자의 일부분이 지워지면서 그 밑에 있는 즉석복권의 내용이 나오게 됩니다. 이에 사용자의 입력 핸들링과 이미지, 그리고 사운드가 들어갑니다.

우선 메인화면에 배경 이미지를 넣기위해서는 PAGfx의 프로그램이 사용됩니다. 이 프로그램에 이미지를 로딩해서 저장하고 변환시켜야 NDS 개발에서 이미지를 사용할 수 있습니다. 이 프로그램은 devkitPro/PAlib/Tools/PAGfx 경로에 위치되어있습니다. 이 프로그램을 사용할때에, 프로그램에서 사용될 이미지(Sprite)와 배경 이미지들을 모두 프로그램에 로드시켜두고나서, 한꺼번에 변환해야합니다. 다만 이미지를 변환시킬때에, 이미지의 색상 설정이 높으면 높을수록 DS 메모리를 많이 차지하게된다는걸 명심해야합니다.

그래서 이번 프로그램에선 사용되는 이미지(Sprite)과 배경 이미지 모두 256 색상으로 설정을 하고 PAGfx 로 변환시켰습니다. PAGfx 로 이미지를 변환시키면 디렉토리내에 여러 파일들이 생성됩니다.

또한 PAGfx 이용한 변환시, PAGfx.log 를 확인하여 tile 의 갯수가 1024 가 넘을시, DS 메모리 제한을 넘기에 좀더 낮은 사양으로 맞추어 변환시켜 주어야합니다.

생성된 파일들을 프로젝트가 저장된 디렉토리안의 source 디렉토리안의 gfx 를 생성해 넣어주어야합니다. 그 후 코드안에 gfx 폴더 안의 헤더파일들을 추가시킨후(all_gfx.c, all_gfx.h), PA_EasyBgLoad(); 함수를 사용해 백그라운드를 출력시킵니다. (예: PA_EasyBgLoad(0, 0, bg_us);) 첫번째 인자는 스크린 넘버, 두번째 인자는 백그라운드 넘버, 그리고 세번째 인자는 이미지의 이름을 의미합니다.

다음은 그래서 로딩한 프로그램 메인 스크린샷입니다.

다음으로 게임 조작법을 설명하는 이미지가 출력되고, 그 다음으로 랜덤의 즉석복권이 출력됩니다.

그 후, 스타일러스의 입력을 받아 그 부분에만 즉석복권 긁고난 아랫부분의 이미지 픽셀 정보를 받아와 스크린에 출력합니다.

하지만 그 전 게임내의 배경 음악을 로딩하려합니다. PA Library 에서의 사운드는 두가지의 파일 형식을 지원합니다. raw 파일과 mod 파일입니다. Mod 파일은 midi 와 비슷한 악기 신호를 저장한 파일이고, raw 파일은 생 소리를 저장해둔 파일입니다. 전 여기서 raw 파일을 사용해 배경음악을 재생하려합니다.

가장 널리 알려진 Switch(http://www.nch.com.au/switch/)를 사용하여, mp3 파일을 8 signed bit, 11052 Mhz, Mono 의 설정으로 raw 을 생성한 후 data 폴더에 넣어두었습니다. 그 후 간단하게 코드 맨위에 #include “raw 파일이름.h” 를 넣어주면 음악파일 로딩이 완료됩니다. raw 파일이름.h 파일이 존재하지않아도 자동적으로 raw 파일이름.raw 를 로딩해줍니다. 그 후 사운드를 초기화해주고, PA_InitSound();, PA_PlaySimpleSound(0, raw 파일이름); 을 코드내에 삽입하면, 노래가 재생됩니다. Mod 파일 역시 같은식으로 노래를 재생하시면 됩니다.

다음으로 즉석 복권의 아랫이미지 파일을 랜덤으로 로딩시킵니다. 이 이미지는 사이즈 제한상 Sprite 대신, Background 로서 변환합니다. 그리고 그 위에 윗 커버 이미지를 덮어 출력합니다. 윗 이미지 또한 크기 제한에 의해 Background 에 포함됩니다. 하지만 필요한 부분을 제외하고는, PAGfx 에서 나머지 여백부분을 투명색으로 지정해줍니다. 복권 이미지는 백그라운드 2 에 출력하고, 윗 커버 이미지는 백그라운드 1 에 출력합니다.

이때 하위 백그라운드부터 출력해야, 출력하는데 있어서 이미지가 깨지지 않습니다. 덤으로 복권을 골라 출력하는 알고리즘을 중간에 삽입한 후,스타일러스가 입력되는 부분에 적당한 크기의 지역을 다시 복권 이미지에서 복사해, 백그라운드 0 에 출력합니다. 5x5 px^2를 권장합니다. 다음의 코드를 참고하세요.

```
while(1){
        PA_ResetBgSysScreen(0);
        PA_Init8bitBg(0,0);
        switch(i){
            case 0:
                PA_EasyBgLoad(0,2, scratch_1);
                break;
            case 1:
                중략
        }
        PA_EasyBgLoad(0,1, cover);
        while (1){
                if (Stylus.Held){
                        recoverSpot(Stylus.X, Stylus.Y);
                }
        }
}
```

위의 코드에서 recoverSpot 함수는 제가 임의로 5x5 px2 사이즈의 페인트 브러쉬를 구현하기 위해 만들어둔 임의 함수입니다.

또한 게임을 끝없이 loop 시킬때에, 항상 Background 시스템을 다시 초기화해주시거나, 이전에 출력되어있던 백그라운드를 제거한 후 다시 로드하셔야합니다.

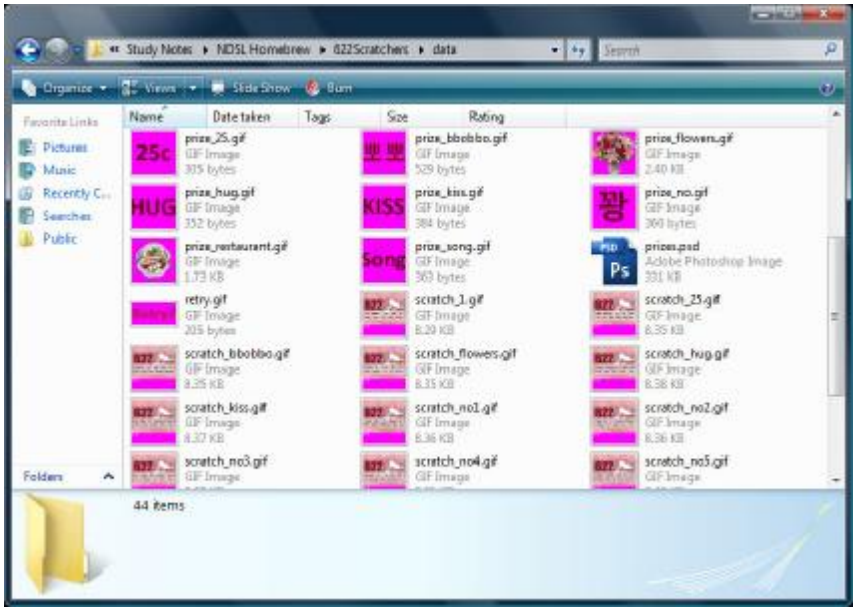
백그라운드 2 의 즉석복권이미지에서 픽셀 정보를 가져와 백그라운드 0 에 다시 칠하는 작업은 PA_EasyBgGetPixelCol(); 함수와 PA_Put8bitPixel()함수를 사용하였습니다.

예). PA_Put8bitPixel(0, x, y, findPal(PA_EasyBgGetPixelCol(0,2,x,y)));

위의 라인에서 findPal 함수는 제가 임의로 만들어둔 함수입니다. 팔레트에 색상에 관한 정보를 찾아주며, 찾는 색상이 없을시에는, 추가하도록 만들어두었습니다. PA_Put8bitPixel() 함수는 색상 정보를 직접받지 않고, 팔레트 번호를 받아 출력하기에, 이러한 구현이 필요하였습니다.

이로서 게임이 완성되었고, 저는 덕분에 여자친구한테 무한한 사랑을 받을 수 있었습니다.

다음은 게임 데이타들과 게임 실행중의 스크린샷입니다.



0x07. ETC

NDS Homebrew 를 배우면서 느낀게, 상업적인 게임과는 달리 Homebrew 에는 제한적인 요건이 너무 많다는 점이 상당히 안타까웠습니다. 상업적인 롬파일의 사이즈에는 제한이 없지만, NDS Homebrew 를 사용하면 4 메가바이트가 최고 롬 사이즈라고 하니, 상당한 제한이 아닐 수 없습니다. 또한 콘솔상의 메모리 역시 너무 적어, 방대한 프로젝트를 계획하기에는 상당한 무리가 따를것같습니다. 허나 메모리 조건하에, 알짜베기 프로그램을 만들어보는것도 재밌는 도전일것같습니다.

devkitPro/PAlibExamples 디렉토리안에 수많은 예제 프로그램들이 존재하니, 보고 응용하시면서 배우시면 학습에 더욱 많은 도움이 될거라 생각합니다.

여러 코딩 대회를 비롯해, NDS Homebrew가 점점 더 발전되고있습니다. NDS 게임기의 인기가 언제까지 지속될지는 의문이지만, 이 분야의 개발만큼은 충분히 재밌는 가능성이있음을 인지합니다. 최근에 확인된 바로는 NDS 콘솔에서 동작하는 여러 운영체제의 유틸리티(Win2DS, DSBrowser)는 물론, 콘솔 자체에 운영체제를 설치해 사용하는 개발자또한 나타나고 있는 추세입니다. 다음 세대 게임 콘솔이 등장하게되면 사라질 개발 환경이지만, 적어도 그때까진 여러 프로젝트를 만들어보는것도 충분히 흥미있는 생각이라고봅니다.

0x08. Reference

http://www.double.co.nz/nintendo_ds/index.html

http://www.ndstutor.org

http://palib.info

0x09. Appendix A : 622 Scratchers NDS file

다음의 주소에서 받으실 수 있습니다.

http://work.hackerschool.org/~wantstar/ndshb/622Scratchers.nds

http://work.hackerschool.org/~wantstar/ndshb/622Scratchers.sc.nds

http://work.hackerschool.org/~wantstar/ndshb/622Scratchers.ds.gba

0x0A. Appendix B : 622 Scratches Code

다음의 주소에서 받으실 수 있습니다만,코드가 많이 허접하고 드럽고 또 지저분하더라도 양해부탁드립니다.

http://work.hackerschool.org/~wantstar/ndshb/622Scratchers.zip

허접한 문서 읽어주셔서 대단히 감사드립니다. z0nKT1g3r