Bu yazının amacı GP2X için yazılım geliştirmek isteyen fakat nereden başlaması gerektiğini bilmeyen geliştiricilere yardımcı olmaktır. Ayrıntılı bilgilerden öte bu yazıda, gerekli bilgilere nereden ve nasıl ulaşabileceğiniz sorularına yanıt vermeye ve sizler için kapıyı aralamaya çalıştım. Sorularınızı canavar@fehmicans.net e-posta adresine gönderebilir ya da çok daha iyisi http://forum.gp2xtr.com/ adresindeki GP2X Türkiye Forumlarına yazabilirsiniz. Bunun yanında aradığınız cevap belki de http://wiki.gp2xtr.com adresinde çoktan yerini almıştır.

GP2X için kod yazmaya başlamadan önce bence ilk işimiz geliştirme makinası üzerinde kullanacağımız işletim sistemini seçmek olmalı. Windows mu kullanmalıyız yoksa bir Linux dağıtımı mı? Ya da MacOS mu? Bu sorunun cevabını vermek benim gibi Windows sistemler üzerinde yalnızca çile çekmiş; baskıya, dayatmalara karşı duran; kendine özgü olarak değiştirilebildiği, geliştirebildiği sistemleri tercih eden ve MacOS sistemleri zorunlu olmadıkça uzaktan seyretmekle yetinmiş biri için oldukça kolay. Ancak yine de Windows fanatikleri için mantıklı bir cevap bulmak gerekiyor. Şurası açık ki GP2X üzerinde Linux çekirdeği çalıştığı için elbette ki geliştirme yapılan sistemin de Linux tabanlı olması geliştirme ve test açısından daha verimli olacaktır. Ayrıca Linux için pek çok açık kaynak araç bulunması da biz geliştiricilere büyük kolaylık sağlıyor. Hatta benim en sevdiğim özellik bu araçlar üzerinde değişiklikler yapabiliyor olmak. Örneğin Code::Blocks editörü üzerinde ihtiyaç duyduğum değişiklikleri yaptıktan sonra "Code::Blocks Canavar Edition" adıyla kullanmak beni dünyanın en mutlu insanı yapıyor. Çünkü bahsettiğim editörün bu halini tüm evrende kullanan tek insan benim ve bu durum beni bütün insanlar arasında özel kılıyor. (Bkz: Züğürt tesellisi)

Linux kullanacağımıza biraz emrivaki de olsa karar verdiğimize göre şimdi sıra şu can alıcı soruya geliyor: Hangi Linux dağıtımını seçmeli? Elimizde pek çok seçenek olması seçim yapmayı oldukça zorlaştırıyor. Uzun zaman Fedora Core kullanmış ben için yanıt bu sefer aynı dağıtım olmayacak. Ben bu konuda tüm ihtiyaçlarımı fazlasıyla karşıladığı için, biraz da milliyetçilik yapıp Pardus dağıtımını seçtim.

Pardus'u http://pardus.org.tr'den indirip kuruyoruz. Bu noktada Pardus ile ilgili bir kaç yararlı bağlantı vermeden de geçmeyeyim.

1. http://gezegen.pardus.org.tr/
2. http://tr.pardus-wiki.org/
3. http://www.ozgurlukicin.com/
4. http://www.pardus-oyun.org/

Evet artık canavar gibi çalışan bir işletim sistemimiz var. Yeni sistemimizin orasıyla burasıyla oynamak elbette ki çok zevkli. Ama çok fazla zaman kaybetmeden yeni görevimiz olan GP2X için bir SDK(Software Development Kit. Ekşi sözlük'ten ssg der ki; bir yazilim icin yazilim geli$tirecek yazilimcilara verilen yazilimlara verilen isim) kurmaya başlamamız gerek. Önce http://wiki.gp2xtr.com/ adresine bir göz atalım. Burada Yazılım Geliştirme kısmında Linux bağlantısına tıklıyoruz. Açılan sayfada gelen öneriler arasından ben pek çok kütüphaneyi içinde barındıran Oopo's DevKit'i tercih ettim(Saygılar Oopo). http://archive.gp2x.de/cgi-bin/cfiles.cgi?0,0,0,0,14,1609 adresinden bu SDK'yı indiriyoruz. İndirdiğimiz dosyayı açtığımızda elimizde 2 klasör olacak. Önce toolchain klasörüne giriyoruz. Buradaki readme.txt dosyasında yazanları yapınca toolchain'imiz(Ekşi sözlükten detached der ki; yazilim dunyasinda belli bir urunu gelistirmek icin kullanilan araclar grubuna bu adi veririz. ornek olarak editor, compiler, linker ve hedef platforma yonelik temel kutuphaneler. [Canavar'dan not: Genelde toolchain ile beraber bir editor gelmez.]) kurulmuş olacak. Ancak bu işlemleri yapmadan önce

```
export LC_ALL=en_US
```

yazmak zorundayız(Niye? Konsolda dil problemini çözmek için...). Toolchain'i kurduktan sonra sıra kütüphaneleri kurmaya geliyor. lib klasörünün içinde bulunan readme.txt dosyasında anlatılanları yaptığımızda bu işlem de bitmiş olacak. Bu yazıyı yazarken o readme dosyalarını silmiş olduğum için ayrıntılı bir açıklama yapamıyorum. Ama giriş seviyesinde İngilizce'si olanlar için orada

anlatılanları yapmanın çok zor bir iş olacağını düşünmüyorum. Yalnızca SDK'yı /gp2xdev klasörüne kurmanızı tavsiye ederim.

SDK'mızı kurduktan sonra yanıt aramamız gereken bir soru ile daha karşılaşıyoruz: Hangi editörü kullanacağız? Bu soruya benim verdiğim cevap VI (http://www.vim.org/). Bu editörün sağladığı kolaylığın ne kadar büyük olduğunu bildiğim kadar acemilere verdiği ızdırabın büyüklüğünü de iyi biliyorum. Bu yüzden Code::Blocks(http://www.codeblocks.org) ya da CDT plugini (http://www.eclipse.org/cdt/) ile Eclipse(http://www.eclipse.org/) sizler için daha iyi bir çözüm olabilir.

SDK'mızı kurduktan sonra sıra artık bir test uygulaması yazmaya geliyor. Ancak bu uygulama her ne kadar basit olsa da SDL, SDL_image, libpng, libjpeg ve libz kütüphanelerinin kullanılabilirliğini de test etmeli. Bu noktada Oopo abimizden bir miktar daha yardım alıyoruz (http://www.oopo.net/consoledev/). Kendisi SDL, SDL_ttf ve SDL_mixer kütüphanelerinin portlarını test etmek için bir test uygulaması yazmış. Ben de bu test uygulamasından yola çıkarak libpng, libjpeg ve libz kütüphanelerini de test eden küçük bir uygulama yazdım. Aşağıda kodları yer alan uygulamayı örnek resim ve ses dosyaları ile birlikte http://fehmicans.net/joomla/source/sdl-tester.tar.gz adresinden indirebilirsiniz.

SDL kütüphanesinin İngilizce dokümantasyonuna http://www.libsdl.org/docs.php adresinden ulaşılabilir. Ya da "Ben İngilizce doküman istemem baba, Türkçe olanı yok mu bunun?" derseniz ragnor(http://ozanemirhan.blogspot.com/) kardeşimizin yazmış olduğu http://www.geocities.com/ragnor_whr/sdl_doc.txt bağlantısından indirebileceğiniz doküman isteğinizi fazlasıyla karşılayacaktır.

```
// tester.c -   Dan Peori <danpeori@oopo.net>
//              Fehmi Can SAGLAM <canavar@fehmicans.net>
// Copy all you want, please give me some credit.

#include <stdio.h>

#include <SDL.h>
#include <SDL_image.h>
#include <SDL_mixer.h>

SDL_Surface *screen = NULL; /*!< video surface */

int draw_bmp(void);
int draw_png(void);
int draw_jpg(void);
int play_music(void);

//////////////////
// MAIN PROGRAM //
//////////////////

int main(int argc, char **argv) {
  int result = 0;

  // Initialize the sdl library.
  result = SDL_Init(SDL_INIT_EVERYTHING);
  if (result < 0) { fprintf(stderr, "ERROR: Could not initialize the sdl library.\n"); return -1; }

  // Set the video mode.
  screen = SDL_SetVideoMode(320, 240, 16, SDL_SWSURFACE);
  if (screen == NULL) { fprintf(stderr, "ERROR: Could not set the video mode.\n"); return -1; }

  // Draw the sample bitmap.
  result = draw_bmp();
  if (result < 0) { fprintf(stderr, "ERROR: Could not draw the sample bitmap.\n"); return -1; }

  // Draw the sample png image.
```

```
    result = draw_png();
    if (result < 0) { fprintf(stderr, "ERROR: Could not draw the sample png image.\n"); return -1; }

    // Draw the sample jpeg image.
    result = draw_jpg();
    if (result < 0) { fprintf(stderr, "ERROR: Could not draw the sample png image.\n"); return -1; }

    // Play the sample music.
    result = play_music();
    if (result < 0) { fprintf(stderr, "ERROR: Could not play the sample music.\n"); return -1; }

    // Flip the video surface.
    result = SDL_Flip(screen);
    if (result < 0) { fprintf(stderr, "ERROR: Could not flip the video surface.\n"); return -1; }

    // Loop forever.
    for(;;) { }

    // End program.
    return 0;

}

//////////////////////
// SAMPLE FUNCTIONS //
//////////////////////

int draw_bmp(void) {
    SDL_Surface *bmp;

    // Load the image into a temporary surface.
    if ((bmp = SDL_LoadBMP("media/image.bmp")) == NULL) { return -1; }

    // Blit the temporary surface onto the video surface.
    if (SDL_BlitSurface(bmp, NULL, screen, NULL) < 0) { return -1; }

    // Destroy the temporary surface.
    SDL_FreeSurface(bmp);

    // End function.
    return 0;

}

int draw_png(void) {
    SDL_Surface *image = NULL;

    //Load the image using SDL_image·
    SDL_Surface *loadedImage = IMG_Load( "media/image.png" );

    //If the image loaded·
    if( loadedImage != NULL ) { //Create an optimized image
        image = SDL_DisplayFormat( loadedImage );
        //Free the old image·
        SDL_FreeSurface( loadedImage );
    }
    else { return -1; }

    // Blit the temporary surface onto the video surface.
    if (SDL_BlitSurface(image, NULL, screen, NULL) < 0) { return -1; }

    // Destroy the temporary surface.
    SDL_FreeSurface(image);
```

```
    // End function.
    return 0;

}

int draw_jpg(void) {
    SDL_Surface *image = NULL;

    //Load the image using SDL_image·
    SDL_Surface *loadedImage = IMG_Load( "media/image.jpg" );

    //If the image loaded·
    if( loadedImage != NULL ) { //Create an optimized image
        image = SDL_DisplayFormat( loadedImage );
        //Free the old image·
        SDL_FreeSurface( loadedImage );
    }
    else { return -1; }

    // Blit the temporary surface onto the video surface.
    if (SDL_BlitSurface(image, NULL, screen, NULL) < 0) { return -1; }

    // Destroy the temporary surface.
    SDL_FreeSurface(image);

    // End function.
    return 0;
}

int play_music(void) {
    Mix_Music *music;

    // Initialize the mixer library.
    if (Mix_OpenAudio(22050, AUDIO_S16, 2, 512) < 0) { return -1; }

    // Load the music.
    if ((music = Mix_LoadMUS("media/music.ogg")) == NULL) { return -1; }

    // Play the music.
    if (Mix_PlayMusic(music, 0) < 0) { return -1; }

    // Unload the music.
    // if (Mix_FreeMusic(music) < 0) { return -1; }

    // Shut down the mixer library.
    // if (Mix_CloseAudio() < 0) { return -1; }

    // End function.
    return 0;

}
```

Fehmi Can SAĞLAM (Canavar)
09/2007 İstanbul
web: http://fehmicans.net
e-posta: canavar@fehmicans.net