모바일 OS 크로스 플랫폼 지원을 위한 앱 게임 엔진 구축

김민호, 정경진, 안동언
전북대학교 대학원 전자정보공학부 컴퓨터공학
e-mail:free4u@mhsoft.co.kr, starfall.jung@gmail.com, duan@jbnu.ac.kr

Development of App Game Engine For Mobile OS Cross-Platform Support

MinHo Kim, KyoungJin Jung, DongUn An
Chonbuk National University Department of Computer Engineering

요 약

애플의 iOS가 처음 출시되고 앱 마켓이 공급자와 소비자 모두에게 각광을 받으며 성공하자 구글에서도 Android를 내세워 구글 마켓을 오픈하였고 결국 다양한 종류, 높은 품질의 애플리케이션들이 사용자들에게 공급되게 되었다. 언제 어디서나 스마트 폰을 통해 고품질의 다양한 앱을 이용할 수 있다는 점이 부각되어 각 마켓의 다운로드 횟수가 급증하는 등 폭발적인 앱 시장을 이룩하게 되었다. 하지만 개발자의 입장에서는 같은 솔루션을 개발할 때 모든 플랫폼을 지원하기 위해 각각의 OS에 맞게 따로 개발하기 때문에 오히려 전체 개발기간이 늦어지는 요소가 되었다. 이런 비효율적인 개발 기간을 단축하기 위해 상용엔진을 구매하여 적용하는 경우가 있지만 영세한 규모의 프로젝트 팀에서 적용하기에는 무리가 있으며 엔진 자체를 배우거나 개발하려는 앱을 위한 커스터마이징하는 기간 역시 짧지 않아 좋은 선택이 될 수 없다. 이에 본 논문에서는 모바일 OS 크로스 플랫폼 지원이 가능한 게임 엔진을 구축하기 위한 방법을 설명하고 이를 활용하여 개발된 프로젝트를 테스트하여 초당 54~61frame 이라는 그래픽 출력 속도를 보임으로써 해당 엔진의 활용 가능성을 증명하였다.

1. 서론

애플의 iOS 와 구글의 Android의 스마트폰의 발전에 따라 두 진영의 앱 마켓 시장도 급속도로 성장하여 왔고 앞으로도 발전 가능성이 무궁무진한 상황이다. 소비자는 다양하고 편리한 앱을 언제 어디서나 마켓으로부터 다운받아 사용해볼 수 있는 편리한 경험을 하고 있는 것이다. 하지만 개발자의 입장에서 보면 OS가 다르다는 점으로 인하여 하나의 OS에서 선 개발 후 동일한 기능을 하는 앱을 다른 OS에 맞게 포팅 작업을 해야 한다. 보통 선개발된 원본 소스 콘텐츠를 분석하고 타 OS 플랫폼에 맞게 재구성하는데 걸리는 시간과 인건비가 선 개발에 못지않기 때문에 상당한 부담이 된다. 이를 해결하기 위해 상용엔진을 구매하여 적용을 하기도 하지만 엔진 자체를 배워야 하는데 걸리는 시간이나 개발하려는 앱을 위해 엔진을 커스터마이징 하는 기간 역시 상당한 비용이 걸리므로 영세한 규모의 프로젝트 팀에서 적용하기에는 무리가 있다. 이에 본 논문에서는 모바일 OS 크로스 플랫폼 지원을 위한 게임 엔진을 구축하는 방법을 설명 한다. 논문의 구성은 2장에서 현재 크로스 플랫폼에 대한 기술동향을 포함한 관련연구를 서술하고 3장에서 크로스 플랫폼 지원을 위한 게임 엔진을 구축 하는 방법을 서술하며 4장에서 실험을 통해 엔진으로서 활용 가능성을 보인다.

2. 관련 연구

현재 모바일 OS 크로스 플랫폼 지원을 위한 연구가 활발히 진행 중이며 그중 대표적인 것이 가상 기계를 활용하는 방법과 HTML5로 대표되는 웹브라우저 기술이 있다.[1]-[7]

2.1. 가상기계

크로스 플랫폼 지원을 위해 기계의 설계를 통한 연구가 있다. 중간 언어를 사용하는 가상기계 기반의 플랫폼으로 여러 스마트 기기에 탑재되어 독립적으로 수행이 가능한 장점이 있다. 하지만 중간 언어를 번역하는 과정이 실시간으로 이루어지기 때문에 속도에 민감한 게임 앱의 경우 적용하기에 무리가 있다.[1][2]

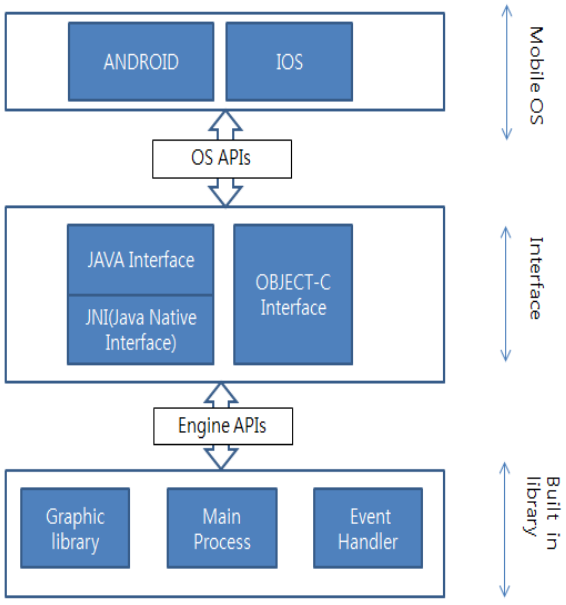
2.2. HTML5

표준 웹 기술을 활용해 네이티브 앱을 만드는 연구도 활발히 진행 중이다. 그중 HTML5는 차세대 앱 개발 언

어로서 각광을 받고 있다.[3] 브라우저 기반의 웹 앱은 손쉬운 개발과 모든 플랫폼에서의 실행을 가능하게 하기는 하지만, 동시에 보안, 실행 속도 등에서 여러 한계점을 갖고 있다. 또한 브라우저에 따라 지원을 하지 않는 것도 있기 때문에 이 역시 게임 앱에 적용할 수 없다.

3. OS 크로스 플랫폼

현재 스마트폰 OS는 크게 구글의 Android 와 애플의 iOS로 구분된다. Android는 JAVA를 기반으로 하는 프레임워크이며, iOS는 Object-C를 기반으로 하는 프레임워크이지만, 두 OS 모두 밑단의 프레임워크에 리눅스를 포함하고 있다. 이와 같은 특성을 활용하여 메인 기능을 C언어로 구현하고, 각 OS에 메인 기능을 연결하는 인터페이스를 설계해 주는 것으로 두 OS간 크로스 플랫폼 지원이 가능한 엔진을 설계한다. 언어 차원에서 Object-C는 C를 기본으로 지원하며, JAVA는 JNI(Java Native Interface)를 통해 간접적으로 C언어를 지원한다. 이러한 구조를 통해 게임의 메인 모듈을 포함한 핵심 기능들은 모두 C와 C++ 코드로 구성하고 해당 기능을 공통적인 인터페이스를 통해 각 OS와 통신하도록 함으로써 OS 크로스 플랫폼 지원이 가능해진다. 게임 엔진은 그래픽 라이브러리, 메인 프로세스, 이벤트 핸들러로 구성이 된다.



(그림 1) iOS와 Android 의 C 언어 지원

3.1. 그래픽 라이브러리

업계 표준인 OpenGL ES를 사용하여 그래픽 라이브러리를 설계한다. OpenGL ES 는 임베디드 단말을 위한 OpenGL의 서브셋으로 2차원 및 3차원 그래픽스 표준 API이다. 이를 활용하여 OS에 종속적이지 않은 그래픽 라이브러리를 구축할 수 있다. 그래픽 라이브러리는 화면 출력 및 텍스처 등록을 담당하는 라이브러리이다.

(그림 2) OpenGL ES Architecture

<표 1> Graphic library 함수목록

함수 명	기능
tm_loadTexture	텍스처를 등록한다.
tm_drawTexture	텍스처를 화면에 출력한다.
tm_mergeImage	텍스처 끼리 조합하여 새로운 이미지를 표현한다.
gr_drawRect	사각형을 그린다.
gr_drawLine	라인을 그린다.
gr_drawPoint	픽셀 하나를 그린다.
gr_drawPoints	픽셀로 이루어진 자유 곡선을 그린다.
gr_drawCircle	원을 그린다.
gr_readPixels	해당 영역의 픽셀들을 읽어온다.
tm_imgResize	텍스처의 크기를 재조정 한다.

3.2. 메인 프로세스

OS의 run roop에서 호출되어 매 프레임마다 실제 게임의 현재 진행 상태에 따라 해당하는 분기로 제어권을 넘기는 역할을 한다.

```
double bob_loop(void) {
    unsigned long testt = timer_milliseconds();
    if(_bFlagPauseResume == false) {
        return 1;
    }
    gr_set2D();
    double retInterval = 0;
   if(_step == STEP_LOADING) {
          ...중략
        if(_stepLoading == 2) {
           home_init();
           _step = STEP_HOME;
           /**/
           _pAniLoadingBar = ani_set("lo",5,0,0);
           /**/
           home_alloc();
        }
    }
    else if(_step == STEP_HOME) {
        ...중략
        uchar ret = home_proc();
        if(ret == PAGE_PLAY) {
        }
    }
    else if(_step == STEP_PLAY) {
        ...중략
        uchar nRet = play_proc();
        if(PAGE_NONE != nRet) {

        }
    }
    return retInterval;
}
```

(그림 3) Main Process 분기 코드

3.3. 이벤트 핸들러

스마트 기기들은 공통적으로 터치 이벤트, GPS, 기울기 센서를 통해 해당 값을 받을 수 있다. 이것을 이벤트 핸들러를 통해 게임의 메인 프로세스와 연동 될 수 있도록 만드는 역할을 한다.

<표 2> 스마트 기기의 외부 입력

함수 명	받게 되는 값
touchesBegan	화면 터치시 xy 좌표를 얻는다.
touchesMoved	화면을 드래그시 xy 좌표를 얻는다.
touchesEnded	화면에서 터치를 땐 후 xy 좌표를 얻는다.
accelerometer	디바이스의 기울기에 대한 xyz 값을 얻는다.
getGpsLocation	해당 위치의 gps 값을 얻는다.

4. 실험

구현된 게임 엔진을 사용하여 화면을 터치했을 때 캐릭터가 점프하는 간단한 게임 앱을 만들어 초당 Frame 수를 테스트 하였다. 실험환경은 Android 테스트를 위해 갤럭시 S-2와 iOS 테스트를 위해 아이폰 3GS를 각각 사용하였다.

그림 4 아이폰 테스트

그림 5 갤럭시 s-2 테스트

<표 3> 초당 프레임수

OS	1회	2회	3회
Android (갤럭시 s-2)	55 - 60	54 - 58	57 - 58
iOS (아이폰 3GS)	57 - 61	58 - 60	56 - 58

통상 적으로 사람의 눈은 초당 30프레임의 동영상에 대해 자연스럽다고 느끼며 60프레임에 가까운 동영상에 대해 매우 부드럽다고 느낀다. 이러한 사실로 볼 때 해당 실험의 결과 표 3에서 두 OS 모두 만족할만한 초당 Frame 수를 보여 주었으며 이것으로 엔진의 활용 가능성을 증명한 셈이 되었다.

5. 결과

본 논문은 모바일 OS 크로스 플랫폼 지원을 위한 앱 게임 엔진을 구축하여 각 OS 별 동일한 앱 게임을 개발함에 있어 그 기간을 획기적으로 단축시킬 수 있는 방법을 제시하였다. 또한 실험의 결과로 실제 활용 가능할 만큼의 성능을 보여 주었다. 이를 통해 OS 별로 포팅을 해야 하는 기존의 부담 없이 앱 게임을 개발할 수 있을 것으로 예상된다.

참고문헌

[1] 한성민, 손윤식, 이양선, “스마트 크로스 플랫폼을 위한 스마트 가상기계의 설계 및 구현” 멀티미디어학회 논문지 제16권 제2호, pp.191-197, 2013.1.

[2] 이양선, “임베디드 시스템을 위한 가상기계 기술”, 멀티미디어 학회지, 제6권, 제2호, pp.36-44, 2002.

[3] 정구민, 강동병, 이경수, “스마트폰 크로스 플랫폼 관련 기술 개발 동향” TTA Journal Vol.144, pp.059-063, 2012.

[4] Daniel Y. Na, “The What, Why, and How of Mobile Applications,” *Sigma,* Vol.11, Issue 1, pp.20-26, Oct.2011.

[5] Gavalas, D., Economou, D., “Development Platforms for Mobile Applications: Status and Trends,” Software, IEEE, Vol.38, Issue 1, pp.77-86, Jan. 2011.

[6] Feida Lin, Weiguo. Ye, “Operating System Battle in the Ecosystem of Smartphone Industry,” 2009 International Symposium on Information Engineering and Electronic Commerce, pp.617-621, 2009.

[7] 최재규, “하이브리드 모바일 앱 개발을 위한 폰갭 - 하이브리드 앱 전성시대 그리고 폰갭 플랫폼,” 마이크로소프트, pp.182-187, 2012.03