Journal of The Korea Society of Computer and Information
Vol. 19, No. 10, October 2014
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2014.19.10.091

Windows Phone Platform 기반의 모바일 게임 설계 및 구현

박 진 양 *

A Design and Implementation of Mobile Game Based on Windows Phone Platform

Jin-Yang Park*

요 약

본 논문에서는 Windows Phone 플랫폼 기반의 모바일 게임을 설계하고 구현한다. 이 게임은 비주얼 스튜디오와 윈도우 모바일 소프트웨어 개발 킷(SDK: Software Development Kit), 그리고 개발자 도구 킷(DTK Developer Tool Kit)으로 비주얼 C#언어를 사용하여 개발한다. 또한 게임 개발에 필요한 닷넷(.NET) 라이브러리의 집합인 XNA Framework를 사용한다. XNA Framework은 PC와 XBOX, ZUNE HD등 다중 플랫폼 환경을 지원하는 게임 개발 및 2D 및 3D 게임 개발에 필요한 기능들을 제공한다. XNA Framework를 사용하여 개발하기 때문에 상속받은 코드들을 자동으로 생성하고, 게임 개발 관련 리소스 및 게임 로직을 쉽게 구현할 수 있다. 이 게임의 특징은 슈팅(shooting), 이미지, 랭킹시스템 등의 3개 요소를 고려하여 개발한 것이다.

▸ Keywords : XNA, Windows Phone7 Framework, .NET Compact Framework, DirectX 라이브러리

Abstract

In this paper, we design and implement a mobile game based on windows phone platform. Visual Studio and the Windows Mobile software development kit(SDK) and developer tool kits (DTK) make it possible to create software for the Windows Mobile platform in Visual C#. Also The XNA Framework is the set of .NET libraries that developers will code against to build games. XNA Framework support PC and XBOX and a ZUNE HD multiplex platform environment and 2D and 3D games provides the necessary functions to a development. XNA Framework is to make it easier to develop games because the codes which are succeeded creates with automatic, implements a game development relation resource and the game logic. The key idea of our game is that we use shooting and ranking system to raise fun.

▸ Keywords : XNA, Windows Phone7 Framework, .NET Compact Framework, DirectX Library

•제1저자 : 박진양 •교신저자 : 박진양
•투고일 : 2014. 9. 11, 심사일 : 2014. 9. 17, 게재확정일 : 2014. 9. 24.
* 인하공업전문대학 컴퓨터정보과(Dept. of Computer Science, Inha Technical College)
※ 이 논문은 2013학년도 인하공업전문대학 교내연구비지원에 의하여 연구 되었음.

I. 서 론

Microsoft는 2010년 기존의 스마트폰 운영체제인 Windows Mobile과 전혀 다른 환경의 플랫폼을 제공하는 Windows Phone 7을 발표하였다. Windows Phone 7은 Siverlight와 XNA를 통해 새로운 개발 환경을 제공한다[1]. Silverlight와 XNA 개발 플랫폼은 .NET Compact Framework 기반으로 C# 언어를 사용하여 개발해야 한다. Silverlight는 Microsoft Expression Blend 4를 함께 사용할 경우 사용자 그래픽 인터페이스 작업을 보다 쉽고 편리하게 진행할 수 있기 때문에 Windows 응용프로그램 개발에 적합하다. XNA는 DirectX 라이브러리를 포함하여 게임과 관련된 다양한 멀티미디어 자원을 하나로 통합하여 PC와 XBOX360, Windows Phone 7에 이르는 멀티플랫폼 기반의 게임을 쉽게 개발할 수 있다[2].

Windows Phone 운영체제는 2010년 스페인 바르셀로나에서 열린 '모바일 월드 콩그레스(MWC) 2010' 에서 처음으로 공개되었다. 지금까지 많은 시간이 흘렀지만, 윈도우폰은 한자리 수 점유율에 머무르고 있다. 2013년 3분기 글로벌 스마트폰 운영체제의 점유율은 그림 1과 같다.

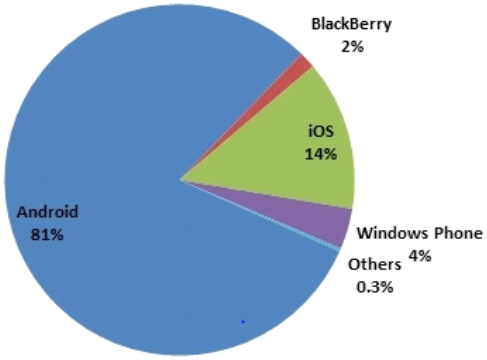

그림 1. 스마트폰 운영체제 점유율
Fig. 1. Smart Phone OS Share rate

그림 1에서 MS Windows Phone의 글로벌 시장 점유율은 4%로 165%로 성장하였다. 하지만 2015년에는 MS 윈도우 폰이 앞으로 구글 안드로이드 운영체제에 이어 시장 점유율 2위를 달성할 것이라는 업계 분석이 이어지고 있다. 시장조사업체 아이서플라이는 미국 현지시각으로 1월18일, MS 윈도우폰 운영체제가 2015년이 되면 모바일 운영체제 시장점유율 2위로 뛰어오를 것이라는 예측 자료를 내놨다[3]. 아이서플라이 자료를 보면, 2015년엔 구글 안드로이드 운영체제가 58.1% 시장을 차지해 1위를 기록하고, 윈도우폰 운영체제는 16.7%로 2위에 오른다. 3위는 16.6%를 차지할 것으로 예상되는 애플 iOS 운영체제다.

2015년 스마트 폰의 시장 점유율을 고려한다면 iOS 플랫폼 기반의 게임 개발 보다는 Windows Phone 기반의 게임 개발이 유리할 것으로 판단된다. 특히, Android 플랫폼과 iOS 플랫폼 기반에서 많은 게임들이 이미 개발된 상태이다. 따라서 iOS 플랫폼 기반의 게임 보다는 Windows Phone 기반의 게임 개발이 필요하다. 스마트 폰의 특성상 높은 이동성과 짧은 시간에 사용자에게 재미를 부여할 수 있다는 측면에서 캐주얼 아케이드 게임의 특성과 동일하다. 아케이드 게임의 쉬운 게임성과 간단한 규칙으로 흥미를 유발한다면 편하게 즐길 수 있는 게임을 개발할 수 있다. 또한 모바일 게임과 Kinect 센서[4]의 융합으로 XBox에서 사용할 수 있는 Windows Phone Platform 기반의 모바일 게임 개발은 더욱 증가할 것이다. 따라서 본 논문에서는 XNA Game Studio를 이용하여 Windows Phone 7 기반의 게임을 설계하고 구현한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구를 소개하고 3장에서는 Windows Phone 게임 설계 및 구현 방법에 대하여 설명한다. 4장에서는 구현 결과를 제시하고 5장에서 결론을 맺는다.

II. 관련 연구

1. Windows Phone SDK

Windows Phone 플랫폼 기반의 게임 개발자는 게임 개발 전에 Silverlight로 구현 할지, XNA로 구현 할지를 결정한다. 두 가지 플랫폼을 간단히 비교해보면 표 1과 같다.

표 1. Silverlight vs. XNA
Table 1. Silverlight vs. XNA

Silverlight	XNA
XAML 기반, event driven application Framework	고성능 게임 Framework
RIA 응용 프로그램의 빠른 생성	멀티 스크린 2D/3D 게임의 빠른 생성
Windows Phone Control	Modes, mesh, sprite, texture, effect, terrain, animation
디지털 미디어 플레이	-
HTML Web browser 컨트롤	-

Silverlight는 XAML 기반으로 event driven 방식의 응용프로그램에 적합하다. 반면에 XNA는 고성능 게임 개발에 적합하다. Silverlight 또는 XNA로 개발된 게임 및 앱은 에뮬레이터로 테스트 할 수 있다. 하지만 Windows Marketplace에 배포하기 위해서는 Windows Phone 장치에서 테스트하고 패키지와 검증(Verification)을 수행해야한다.

2. Windows Phone Emulator

Windows Phone 에뮬레이터는 가상 머신(VM: Virtual Machine)으로 동작하며 다음과 같은 기능을 제공한다.

- 다중 해상도 지원: 에뮬레이터 이미지는 Windows Phone 에서 지원되는 세 가지 해상도에 각각 사용할 수 있다.
- Windows Phone 에뮬레이터에 대한 시스템 요구 사항 Windows Phone 에뮬레이터는 Windows Hyper-V에서 실행되고 Hyper-V와 동일한 하드웨어, 소프트웨어 및 구성 요소 사항을 포함한다. 컴퓨터에서 Hyper-V를 지원하지 않는 경우 Windows Phone 용 앱을 개발하고 Windows Phone 장치에서 디버깅 및 테스트할 수 있다.
- 네트워킹 지원: Windows Phone OS 7.1 에뮬레이터에서는 개발 컴퓨터의 네트워크 연결을 사용한다. 그러나 Windows Phone 8 에뮬레이터 는 그 자체를 네트워크에서 별도의 장치로 구성한다.

3. Windows Marketplace / App Hub

개발자들이 개발한 Windows 기반의 앱은 Windows Marketplace에서 유료 또는 무료로 판매할 수 있다. Windows 기반 앱의 배포 과정은 그림 2와 같다.

그림 2. Windows 앱 배포 과정
Fig. 2. Windows App Deployment Process.

Windows Marketplace는 Window Phone이 국내 출시한 이후 사용자가 앱을 구매하거나 다운로드 받는 곳을 말한다[5]. Phone 에서는 Market place라고 불러지는 허브에서 다운로드를 받을 수 있고, PC 에서는 Zune 클라이언트 안에 Marketplace 의 App 카테고리가 있으며, Web 용은 디바이스가 출시 될 때 오픈할 예정이다. 반면에 앱 허브(App Hub)란 개발자가 마켓플레이스에 앱을 올리기 위해 앱 및 ISV 개발자 등록을 위한 포털을 말한다[6]. 따라서 앱 허브를 업데이트 하면서 변경된 사항 중에 하나가 일반 사용자들을 위한 배포뿐만 아니라 기업 사용자들이나 특정 사용자에게 베타 버전을 배포할 수 있다. 이 배포는 Private Distribution 이라고 하며, 폰 마켓플레이스에서 검색하지 않고 E-Mail 이나 Push Notification (Deep Link)을 통한 앱 다운로드가 가능하다.

Windows Marketplace는 베타 배포(Beta Distribution) 기능을 제공한다. 개발자들에게 먼저 인증된 앱을 100개 이상 그룹 단위로 베타 테스터에게 90일 동안 배포 할 때 사용하는 옵션이다. Windows Market Place에 업로드 전에 개발된 앱을 베타 테스팅 할 수 있도록 지원한다. 또한, 타겟 배포(Targeted Distribution) 기능을 제공한다. 이 기능은 Windows Market place에서 보이지 않게 앱을 배포할 수 있도록 지원한다. 사용자들은 Marketplace에서 검색 할 수 없지만 특정 사용자에게 이메일 또는 메시징을 통해 웹을 배포할 수 있도록 지원한다. 이 기능은 기업 사용자들을 위해 배포 한다면 사용하기에 편리하다.

4. Windows Phone 개발자 Feedback 시스템

Window Phone 망고 버전부터는 개발자들이 피드백을 받아 앱 관리를 하는 기능을 확장하였다. 따라서 App Hub를 사용하는데 있어서 좀 더 편리하게 account management tools을 사용하거나 streamlined application submission process를 돕거나 확장된 리포팅 기능을 다음과 같이 더 포함한다.

- 향상된 앱 관리: 'Lifecycle' 탭의 'Edit catalog details' 를 선택함으로써 개발자 앱을 다시 인증 받을 필요 없이 앱 메타데이트를 변경할 수 있다. 따라서 앱을 업데이트하거나 새로운 정보를 게재 할 때 시간을 단축 할 수 있다.
- 확장된 개발자 대쉬보드: 앱 퍼포먼스를 측정하거나 지불 상태, 배포 리포팅 및 순위 및 여러분들의 앱 리뷰, 언어 등 지원을 한 눈에 볼 수 있도록 제공한다.
- 업그레이드된 상세한 리포팅 기능: 'Summary' 에서 'Detail' 뷰를 봐서 상세한 내용을 파악할 수 있으며 Excel 호환 가능한 자료로 데이터를 분석할 수 있다.
- 신규 'Crash Count' 리포트: App Hub 는 충돌 카운트

데이터를 볼 수 있으며 더 빠르게 퍼블리셔가 디버깅할 수 있도록 도움을 준다. 만일 앱을 업로드할 때 실패한다면 관련된 상세한 메시지 문구를 전달한다.

- Streamlined application submission process: 개발자들이 'Browse' 버튼을 누르고, 앱을 등록하는 동안에 한 번에 파일을 올릴 수 있는 기능을 제공한다. 내부적으로 앱 허브가 자동적으로 image 파일 사이즈를 감지하고 개발자가 마지막으로 퍼블리싱하기 전에 수정 또는 제거 할 수 있는 기능을 제공한다.
- 신규 App 카테고리: education, kids & family and government & politics 와 같은 3개의 신규 앱 카테고리가 존재한다. 또한 개발자가 멀티 카테고리를 지정할 수 있는 기능을 제공한다.

5. Windows Phone Framework

Windows Phone7 Framework은 그림 3과 같이 Common Base Class Library, Silverlight Presentation and Media, XNA Framework, Sensors/Location 등으로 구성된다[7].

Common Base Class Library는 .NET Compact Framework에서 제공되는 기본 라이브러리로 IO, 네트워크, 스레드 및 기본 API를 포함된다. Silverlight Presentation and Media는 Silverlight 는 Windows Phone 7용 애플리케이션이나 게임 개발을 목적으로 하는 개발 환경이다. XNA Framework는 PC와 XBOX, ZUNE HD등 다중 플랫폼 환경을 지원하는 게임 개발 및 2D 및 3D 게임을 개발에 필요한 기능들을 제공한다. Sensors/Location은 다양한 하드웨어 기능들에 애플리케이션 및 게임 개발에 활용 할 수 있는 라이브러리를 제공한다.

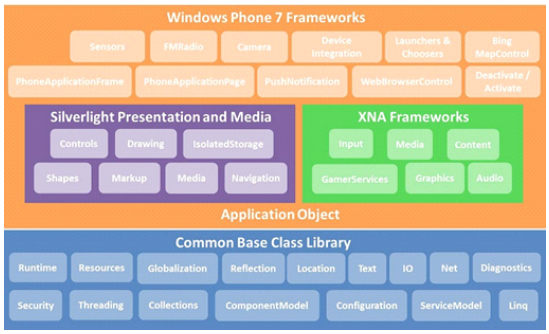

그림 3. Windows Phone7 프레임워크
Fig. 3. Windows Phone7 Framework

III. Windows Phone 게임 설계 및 구현

1. Windows Phone 게임 설계

본 논문에서는 Windows Phone Platform 기반의 모바일 게임으로 Rabbit-Run 게임을 설계하고 구현한다. Windows Phone Platform 기반의 모바일 게임 구조는 그림 4와 같다.

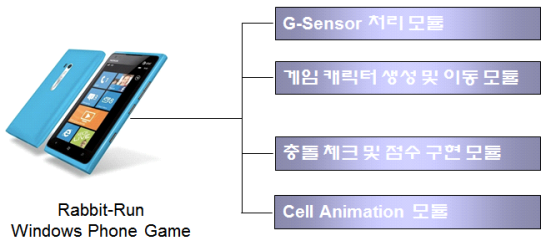

그림 4. Rabbit-Run 모바일 게임 구조
Fig. 4. Rabbit-Run Mobile Game Architecture

본 논문에서는 그림 4의 각 구성 요소에 대한 프로그래밍 모듈을 아래와 같이 구현한다.

- G-Sensor 처리 모듈 : G 센서의 X 축과 Y 축을 이용하여 플레이어의 상/하/좌/우 이동, 기울기에 따라 가속도 처리, 화면 터치 처리 모듈을 개발한다.
- 게임 캐릭터 생성 및 이동 모듈: 플레이어와 대상 객체에 각각의 레벨을 부여하고, 그 레벨에 따라 크기와 이미지를 다르게 표현하는 모듈을 개발한다.
- 충돌 체크 및 점수 구현 모듈: 플레이어와 대상 객체의 충돌 및 점수 처리 모듈을 개발한다.
- Cell Animation 모듈: Cell animation 및 배경 음악, 캐릭터 움직임에 따르는 음악 처리 모듈을 개발한다.

Rabbit-Run 게임에서 사용하는 캐릭터는 그림 5와 같다.

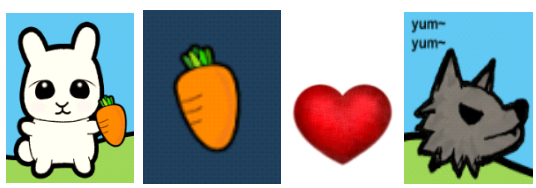

그림 5. 게임 캐릭터
Fig. 5 Game character

그림 5에서 토끼가 당근을 먹게 되면 점수가 1증가하고,

하트를 먹게 되면 목숨이 1증가 하며, 늑대를 먹게 되면 목숨이 1감소한다.

2. Windows Phone 게임 구현

Rabbit-Run 게임은 XNA Framework를 사용하여 개발하기 때문에 상속받은 코드들이 자동으로 생성되고, 게임 개발 관련 콘텐트 리소스의 등록 및 게임 로직을 해당 메소드에 구현한다. 구현된 메소드는 그림 6의 실행과정에 따라 수행된다[8][9].

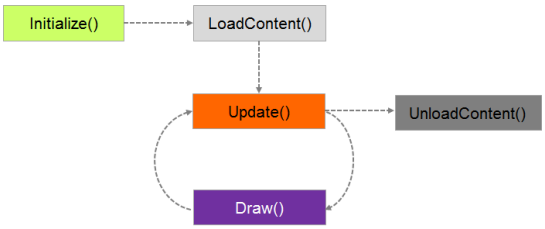

그림 6. 메소드 실행 과정
Fig. 6. Method Life Cycle

그림 6에서 Initialize() 메소드는 구현에 선언된 멤버변수를 초기화한다. LoadContent() 메소드는 1회 호출되며 게임에 필요한 리소스를 로드한다. UnloadContent() 메소드는 1회 호출되며 게임에 필요한 모든 리소스를 해제한다. Update()와 Draw() 메소드는 게임이 종료될 때까지 반복적으로 호출된다. Update() 메소드는 게임 로직에 관련된 기능을 구현한다. Draw() 메소드는 게임 배경 및 관련 캐릭터를 화면에 그리는 기능을 지원한다.

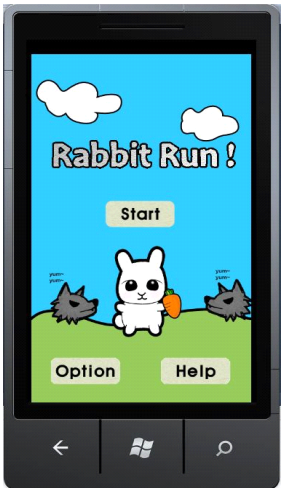

그림 7. Windows Phone 게임 시작 화면
Fig. 7. Windows Phone Mobile Game Start Screen

Rabbit-Run 게임의 초기화면은 그림 7과 같다.

초기화면에서는 게임 Option과 Help 기능을 구현한다.

그림 7에서 Option 버튼과 Help 버튼을 클릭하면 그림 8이 나타난다.

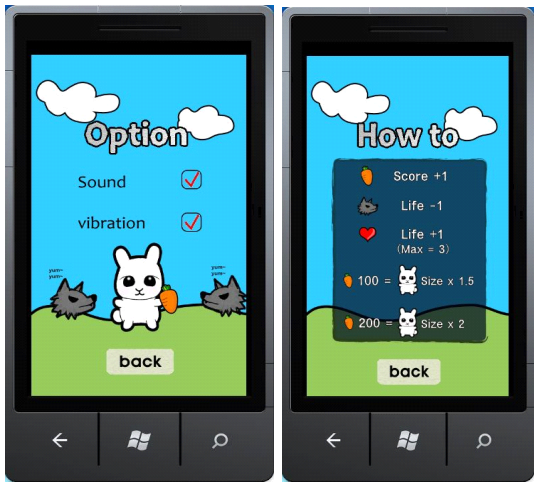

그림 8. 옵션 화면과 도움말 화면
Fig. 8. Option and Help Screen

그림 8 Option 기능에서는 소리(sound)와 진동(vibration)기능을 지정할 수 있다. Help 기능에서는 게임 실행에서 토끼가 당근을 취득하면 점수가 1증가하고, 늑대를 취득하면 Life 값이 1 감소하고, 하트를 취득하면 Life 값이 1 증가한다. 또한, 점수가 100 이상에서는 토끼 캐릭터의 크기가 1.5배 증가하고, 점수가 200 이상에서는 토끼 캐릭터의 크기가 2배 증가한다. 이러한 Rabbit-Run 게임 실행 화면은 그림 9와 같다.

그림 9에서 당근, 늑대, 하트가 화면 위에서 아래로 떨어진다. 토끼는 좌/우로 이동하면서 떨어지는 당근, 늑대, 하트를 취득한다. 이러한 토끼는 움직임에 따른 게임 로직을 구현한 소스는 그림 10과 같다.

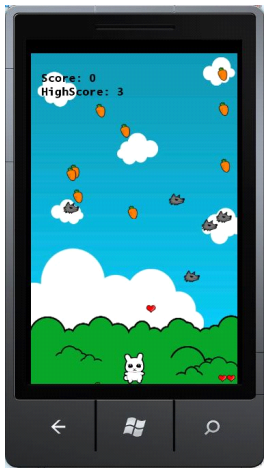

그림 9. Rabbit-Run 게임 실행 화면
Fig. 9. Rabbit-Run Game Execution Screen

그림 9에서 토끼 캐릭터는 좌/우로 이동하여 이러한 토끼의 움직임은 G 센서를 통하여 감지한다.

```
public void UpdateCarrots()
{
   foreach (GameObject carrot in carrots)
   {
      if (carrot.alive) {
         carrot.position += carrot.velocity;
         if (!viewportRect.Contains(new Microsoft.Xna.Framework.Point(
                              (int)carrot.position.X, (int)carrot.position.Y)))
            carrot.alive = false;
      } else {
         carrot.alive = true;
         carrot.position = new Vector2(MathHelper.Lerp(0,
               (float)viewportRect.Width - carrot.sprite.Width,
               (float)random.NextDouble()), viewportRect.Top);
         carrot.velocity =  new Vector2(0, MathHelper.Lerp(mincarrotVelocity,
                                 maxcarrotVelocity, (float)random.NextDouble()));
      }
      Rectangle carrotRect = new Rectangle((int)carrot.position.X,
                (int)carrot.position.Y,  carrot.sprite.Width, carrot.sprite.Height);
      if (gameScore < 100)
         carrotProcess((int)playerPosition.X, (int)playerPosition.Y,
                              cell.Width, cell.Height);
      else if (gameScore >= 100 && gameScore < 200)
         carrotProcess((int)playerPosition2.X, (int)playerPosition2.Y,
                              cell2.Width, cell2.Height);
      else if (gameScore >= 200)
         carrotProcess((int)playerPosition3.X, (int)playerPosition3.Y,
                              cell3.Width, cell3.Height);
   }
}

   public void carrotProcess(int playerPositionX, int playerPositionY,
                              int cellWidth, int cellHeight)
   {
      Rectangle playerRect = new Rectangle(playerPositionX, playerPositionY,
                                          cellWidth, cellHeight);
      if (carrotRect.Intersects(playerRect))
      {
         if (isSoundOn) eatCarrot.Play();
         carrot.alive = false;
         gameScore += 1;
         break;
      }
   }
```

그림 10. 당근 캐릭터 처리 코드
Fig. 10. Carrot character process source code

Rabbit-Run 게임의 랭킹 시스템 구현 그림 11과 같다.

```
private static void DoSave(string gameScore)
{
   IsolatedStorageFile file = IsolatedStorageFile.GetUserStoreForApplication();
   byte[] data = Encoding.UTF8.GetBytes(gameScore);
   using (IsolatedStorageFileStream stream = file.CreateFile("savedata.sav"))
   {
      stream.Write(data, 0, data.Length);
      Debug.WriteLine(gameScore);
   }
}

public String DoLoad()
{
   IsolatedStorageFile file = IsolatedStorageFile.GetUserStoreForApplication();
   if (file.FileExists("savedata.sav"))
   {
      using (IsolatedStorageFileStream stream
         = new IsolatedStorageFileStream(@"savedata.sav", FileMode.Open, file))
      {
         byte[] decoded = new byte[stream.Length];
         stream.Read(decoded, 0, (int)stream.Length);
         String readData = Encoding.UTF8.GetString(decoded, 0, (int)stream.Length);
         return readData;
      }
   }
   else
      return "0";
}
```

그림 11. 랭킹 시스템 소스 코드
Fig. 11. Ranking system source code

그림 11 랭킹시스템의 점수는 격리된 저장소 영역을 이용하여 Windows phone내에 저장한다. 그림 11의 DoSave() 메서드는 점수를 저장하고, DoLoad() 메서드는 점수를 로드한다.

그림 12. Rabbit-Run Game 게임 중단 및 종료 화면
Fig. 12. Rabbit-Run Game pause and Exit Screen

IV. 결 론

본 논문에서는 Windows Phone 7 기반의 모바일 게임

Rabbit-Run을 설계하고 구현한다. 이 게임은 XNA Framework를 사용하기 때문에 상속받은 코드들을 자동으로 생성할 수 있다. 다양한 캐릭터와 관련된 콘텐트 리소스 등록 및 게임 로직을 해당 메소드에 구현함으로써 쉽게 개발할 수 있다. 또한 C# 언어를 사용하여 구현하기 때문에 메모리 관리 및 생산성에 대한 부담을 줄일 수 있다. Rabbit-Run 게임은 Windows 앱 스토어인 Marketplace에 등록되어 있기 때문에 다운로드 할 수 있다.

참고문헌

[1] *http://blogs.msdn.com/b/xna/*

[2] *http://msdn.microsoft.com/en-us/centrum-xna.aspx*

[3] Tom Miller, *http://www.bloter.net/archives/92782*

[4] *http://www.microsoft.com/en-us/kinectforwindows/develop/sdk-eula-korean.aspx*

[5] *http://en.wikipedia.org/wiki/Windows_Marketplace*

[6] *http://dev.windows.com/ko-kr/develop*

[7] Tom Miller, Dean Johnson "XNA Game Studio 4.0 Programming: Developing for Windows Phone 7 and XBOX 360," Sams, 2010.

[8] S.H. Lee, D. H. Kim, E. D. Kim, J. C. Lim, B. K. Jung, "Windows Phone 7 Game Programming using XNA," BJPublic, 2011.

[9] W. J. Lee, J. Y. Kang, M. H. Park, J. H. Hong, J. W. Kim, M. H. Cho, "A Design and Implementation of Mobile Game Wings," Proceeding of Korea Society of Computer Information 2014 Winter Conference (ISSN 2005-1344), Vol. 23, No. 1, pp. 81-82, Jan. 2014.

저 자 소 개

박 진 양

1982: 단국대학교 전자공학과 학사.

1984: 단국대학교 전자공학과 공학석사.

1997: 단국대학교 전자공학과 공학박사.

현 재: 인하공업전문대학 컴퓨터정보과 교수

관심분야: 컴퓨터시스템, 디지털시스템, 지능형로봇, 모바일플랫폼

Email: jinyang@inhatc.ac.kr