게임 개발을 위한 사용자 인테페이스 개발 방법

김성수°, 김수균**, 안성옥**, 강신진***, 정진영****, 안우영****
°위메이드 엔터테인먼트
**배재대학교 게임공학과
***홍익대학교 게임학부
****대전보건대학교 바이오정보과
e-mail:{nicesk}@gmail.com, directx@hongik.ac.kr, jyjung@hit.ac.kr, wyahn@hit.ac.kr

Development of User Interface for Game

SungSu Kim° ,SooKyun Kim**, SungOhk An**, Shin-Jin Kang***, JinYoung Jung***
and WooYoung Ahn***
°Wemade Entertainment
**Dept. of Game Engineering, Paichai University
***School of Game, Hongik University
****Dept.of BioInformation, Daejeon Health Science College

● 요 약 ●

유니티3D는 게임 회사에서 가장 많이 사용하고 있는 게임 엔진 중의 하나이다. 다양한 장르의 게임을 쉽고 간편하게 개발 할 수 있고, 또한 개발비를 절약 할 수 있는 장점을 가지고 있다. 다양한 플랫폼 지원으로 인해 iOS(iPhone Operating System) 환경, 안드로이드(Android) 환경에 맞도록 최적화 하여 플랫폼 전환도 가능하다. 본 논문은 유니티3D에서 강점인 NGUI(Next-Gen UI)의 사용방법과 설계에 대한 방법을 통해서, 사용자 인터페이스 개발에 쉽게 접근 할 수 있는 방법에 대해 소개한다.

키워드: 유니티(Unity), 사용자 인터페이스(User Interface), 게임(Game)

I. 서 론

유니티3D에서 사용자 인터페이스를 개발하기 위해서는 아틀라스에 대한 개념을 이해해야 한다. NGUI 설계의 기본 구조는 아틀라스 간의 깊이 규칙이며, 아틀라스의 깊이를 명시하여 NGUI를 설계해야 한다. 아틀라스와, NGUI 설계에 대하여 논한다.

II. 본 론

2.1. 아틀라스

아틀라스는 게임 개발에 필요한 2차원 이미지들을 쉽고 편하게 제어 할 수 있도록 하나의 이미지 파일로 만들고, 그것을 텍스쳐로 변환하여 사용 할 수 있도록 그림 1.과 같이 만든 이미지들의 묶음이라고 정의 할 수 있다.[2] 드로우 콜을 최소화하기 위하여 이미지 리소스를 아틀라스로 만들고 아틀라스를 활용하여 한 장의 이미지를 드로우 콜 한다. 많은 UI 리소스를 한 장으로 생성하여 드로우 콜이 늘어나는 문제를 줄일 수 있다.

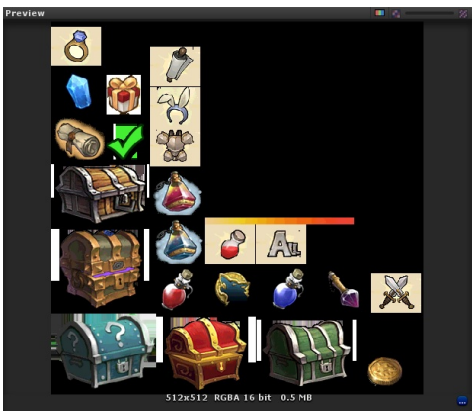

그림 1. NGUI 아틀라스 텍스쳐[1]
Fig. 1. NGUI atlas texture [1]

유니티3D는 기본 3차원 게임 엔진으로, 메트리얼을 이용하지 2차원 이미지를 바로 적용 하는 것은 적절하지 못하다. 자체 GUI 텍스쳐 속성을 제공하지만, 많은 드로우 콜을 발생 시키고, 이 과정에 불편한 부분이 많다.

NGUI를 사용하는 과정에는 유니티3D의 본질적 텍스쳐 사용법인 2차원 텍스쳐를 3차원 메트리얼에 적용하여 사용 한다.

2.2. NGUI UI 설계

NGUI UI 설계에서는 깊이(Depth)[3]와 3차원 Z값을 고려해서 설계해야 한다. 2차원 공간의 개념으로 하나의 UI 패널(Panel)[4]을 생성하고, 패널 안에 여러 가지 NGUI 요소들로 구성된다. 이러한 경우 깊이와 Z값을 고려하지 않을 경우 렌더링 순서에 의해서 화면에 출력되지 않는 현상이 발생할 수 있다.

표 1. 아틀라스 스프라이트 렌더링 순서
Table 1. Atlas sprite rendering order

패널	아틀라스	스프라이트	깊이	Z 값	출력 순서
패널 A	아틀라스 A	스프라이트A	1	0	1
		스프라이트B	0	0	2
패널 B	아틀라스 A	스프라이트A	0	-1	1
	아틀라스 B	스프라이트B	0	0	2
패널 C	아틀라스 A	스프라이트A	0	0	3
	아틀라스 A	스프라이트B	1	0	2
	아틀라스 B	스프라이트C	0	-1	1
패널 D	아틀라스 A	스프라이트A	0	0	1
	아틀라스 B	스프라이트B	0	1	출력 안 됨
	아틀라스 A	스프라이트C	0	2	1

설계에서의 아틀라스 기본 출력순서 규칙은 표 1.에 정리하였다. Z값 위치는 마이너스일수록 상위에 출력한다고 명시하겠다. 동일 아틀라스를 사용하는 스프라이트는 Z값에 상관없이 깊이 순으로 출력된다.

패널 안에 스프라이트A와 스프라이트B가 같은 아틀라스로 있을 경우 Z값에 상관없이 깊이 순으로 출력된다. Z값에 상관없다고 하여도, 다른 아틀라스와 겹치는 현상을 방지하기 위해서는 항상 Z값을 0으로 초기화 하는 것이 좋다. 반대로 서로 다른 아틀라스를 사용하는 스프라이트는 Z값을 기준으로 출력한다. 아틀라스A의 스프라이트와 아틀라스B의 스프라이트가 있을 경우 깊이와 상관없이 Z값이 마이너스 일 경우 상위에 출력한다.

동일한 아틀라스를 사용하는 스프라이트 1개의 Z값이 상위일 때 모든 스프라이트가 상위에 출력되는 문제가 가장 많이 발생하기 때문에 출력순서를 규칙에 맞게 설계해야 한다.

III. 결 론

앞서 설명한 개발 방법을 이용하여서 사용자 인터페이스에 적절하게 적용하면, 다양한 장르의 게임에 쉽고 간편하게 개발 할 수 있다. 뿐만 아니라, 디바이스(Device)의 성능을 최소화 하고 리소스가 출력되지 않는 상황을 예방 할 수 있다.

참고문헌

[1] 위메이드 엔터테인먼트 게임 NQ팀 달을삼킨늑대 http://weme.wemade.com/game/game_info.asp?GameCode=18

[2] 크레이그스티븐스, 사이먼퀵그(2011). 『Unity 3 Blueprint 한국어판』

[3] Creighton(2010). 『Unity 3D Game Development by Example Beginner's Guide』.

[4] 유니티3D 스크립트 레퍼런스 http://unity3d.com/support/documentation/ScriptReference/Camera.ScreenPointToRay.html