※ 본 아티클은 CMP MEDIA LLC와의 라이선스 계약에 의해 국문으로 제공됩니다

Gamasutra.com

벽돌 깨기 분석: 벽돌 깨기 스타일의 게임을 위한 시스템과 레벨 설계

Mark Nelson

2007년8월20일

2006년 겨울 동안 나는 뉴욕시에 있는 Large Animal Games에 계약되어 캐주얼 게임 시장을 목표로 한 벽돌 깨기 스타일의 게임인 *LEGO 브릭토피아(LEGO Bricktopia)*라는 게임 제작에 참여했다.

나는 벽돌 깨기 게임을 그리 많이 즐겨보지는 않았으므로 내가 처음에 해야 할 확실한 일은 바로 그 당시의 시장을 조사하는 동시에 역사적인 관점에서 해당 장르에 대한 조사를 병행하는 것이었다. 본 조사의 결과와 함께 내가 벽돌 깨기 스타일의 레벨을 설계하면서 얻은 경험에서 우러난 "경험적인 지식"이 본 아티클의 근간을 이루고 있다.

장르 개요, 짤막한 역사

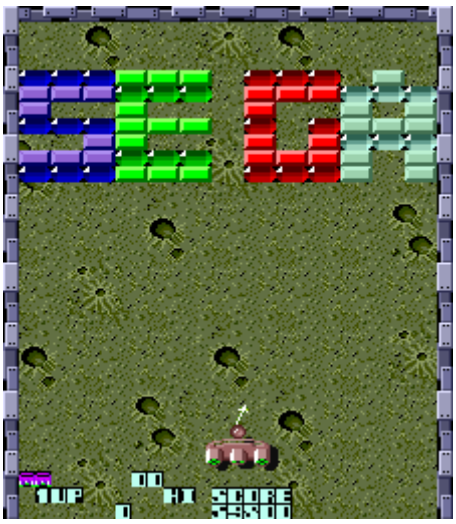
벽돌 깨기(Breakout), *아카노이드(Arkanoid)*와 같이 생각할 수 있는 모든 "패들, 공 vs. 벽돌" 형식의 비디오 게임의 근원은 1967년 초, Ralph Baer가 마그나복스 오디세이(Magnavox Odyssey) 게임 시스템과 패들 컨트롤러를 설계했을 때까지 거슬러 올라간다. 이 시스템에 포함되어 있던 "패들과 공" 게임 메카닉을 사용한 장래성이 있던 게임 중 하나가 *벽돌 깨기(Breakout)* 장르의 첫 번째 게임이 되었다.(대다수

의 사람들이 패들과 공 게임 메카닉이 1972년 Atari에서 발매된 Nolan Bushnell의 *퐁[Pong]*에서 유래되었다고 잘못 생각하고 있다.)

1년 후에 Nolan Bushnell과 Steve Bristow와 함께 Steve Jobs, Steve Wozniak(Apple로 유명한)이 *벽돌 깨기(Breakout)*(Atari, 1976)를 개발하며 패들과 공 게임 플레이를 한 단계 더 진화시켰다. 그리고 이 게임은 이후 벽돌 깨기 장르의 게임들에서 채용된 "패들과 공 vs. 벽돌"이라는 내용이 포함된 최초의 게임이었다.

1986년에 Sega사에서 *기가스(Gigas)*와 *기가스 마크 II(Gigas Mark II)*

를 출시하여 *벽돌 깨기* 스타일의 게임은 대단한 도약을 이루게 된다. 이 게임들은 화려한 그래픽 배경, 파워 업, 플레이어가 피해야만 하는 낙하하는 물체 등과 같은 혁신적인 내용을 담고 있었다.



같은 해에 Taito/Romstar는 *아카노이드 (Arkanoid)*를 발매하며 벽돌 깨기 시장에 뛰어들었다. 비록 아카노이드는 *기가스*보다 더 늦게 출시되었지만, 장르에서 가장 유명한 게임이 되었으며 현재 "패들과 공 **vs. 벽돌과 다른 물건**"으로 정의되는 게임 플레이를 가장 뚜렷하게 나타낸 게임으로 인식되고 있다.

벽돌 깨기 스타일 게임의 중요 구성 요소

모든 게임 장르에는 원조가 있는 법이고, 그 이후에 한 가지나 두 가지의 새로운 기능을 소개한 게임에는 복제품이라는 꼬리표가 필연적으로 따라다니기 마련이다. 그리고 벽돌 깨기 장르도 이런 점에서 전혀 차이가 없다. 원조 벽돌 깨기가 매우 제한적인 레벨 설계와 목표 물체의 다양성만을 가지고 있는 반면에 이후에 복제와 점진적인 개선 과정에서 출시된 게임들에서는 현재 벽돌 깨기 스타일 게임의 "중요한" 구성 요소로 간주되는 수많은 기능들이 소개되었다.

여러 번 맞춰야 부서지는 벽돌 – 벽돌은 벽돌 깨기 스타일 게임의 레벨에서 대부분의 공간을 차지하고 있다. 몇몇 게임에서는 벽돌의 내구성을 다양화해(한번 맞춰야 부서지는 벽돌과 여러 번 맞춰야 부서지는 벽돌) 플레이 화면을 좀 더 복잡하게 만들 때 활용하고 있다.

벽돌과 공의 크기 다양화 – 더 큰 벽돌과 공은 특정 목표를 맞추기가 더 쉽게 만들어주므로 게임을 더 짧고, 쉽게 만들어 준다. 반대로 더

작은 벽돌과 공은 (작아진 목표 크기와 가시성 때문에) 난이도를 상승시키는 요인이 된다.

벽돌 색깔 다양화 - 색깔은 순수하게 미학적인 측면과 상징적인 게임 레벨을 만들 때 사용된다.(본 아티클의 뒷부분에 나오는 **레벨 디자인 - 일반적인 구조적 원형** 참조) 또한 서로 다른 속성을 나타낼 때 다른 색깔을 사용하는 등의 특정 의미를 부여할 수도 있다.(파괴할 때 필요한 명중 수, 더 높은 점수 등)

부서지지 않는 벽돌 - 부서지지 않는 벽돌의 주요 목적은 레벨 목표에 대한 접근을 제한할 때 사용된다.(또는 공이 왔다갔다할 통로를 만들 때) 부서지지 않는 벽돌은 간혹 공을 더 빠른 속도로 튕겨낸다든지, 공이 특별한 스위치나 트리거에 맞으면 사라진다든지 하는 특수한 속성을 가지고 있을 때도 있다.

파워 업 - 수많은 종류의 파워 업이 있으며 그 모두를 이곳에 설명해 놓지는 않았다. 하지만 대부분의 게임에서 등장하는 것으로 보이는 근본적인 종류는 한 번 열거해보도록 하겠다. (a) 공을 패들에 들러붙게 만들기, (b) 공이 느려짐, (c) 공의 파괴력 증가, (d) 공의 크기가 커짐, (e) 추가적인 공이 화면에 나타남, (f) 패들이 더 길어짐 등이다. "공의 크기가 커짐"과 같은 몇몇 파워 업은 파워 업이 적용되었을 당시에 공이 어떤 위치에 있는 가에 따라 실제적으로는 파워 다운의 역할을 하게 된다.

대부분의 경우에 파워 업은 벽돌 더미에서 떨어지며(공에 맞았을 때) 효과를 나타내려면 패들로 건드려야 한다. 이로 인해 플레이어는 공을 쫓아가는가와 그 상태에서 필요한 파워 업을 획득하러 가는가 중에서 중대한 선택을 해야만 한다.

파워 다운 - 위와 동일하게 대부분의 게임에서 등장하는 근본적인 종

류만 나열하겠다. 파워 다운에는 (a) 패들이 느려짐, (b) 패들이 더 작아짐, (c) 공이 빨라짐, (d) 공이 작아짐 등이 있다. 파워 다운이 벽돌 더미에서 떨어질 때 보통은 플레이어가 그것이 안 좋은 것이라는 것을 인식하고 있으므로 실수로 획득하지 않도록 주의해야 한다.

공의 궤도 조작 - 벽돌 깨기 스타일 게임에서는 일반적으로 다음과 같은 몇 가지 공통된 궤도 조작 장치가 등장한다. (a) 트리거 이벤트(trigger event)에 따라 공을 잡아뒀다가 발사하는 메커니즘, (b) 공을 빨아들이고 미리 지정된 궤도를 따라 자동으로 공을 배출하는 구멍, (c) 예측하기 힘든 각도로 공을 반사시키는 쐐기꼴 모양, (d) 공의 이동 방향을 새로운 방향으로 강제적으로 변경시키는 통로.(또는 화살표)(http://www.e-giraffa.com의 Paper Ball이라는 논문 참조) 궤도 통제 장치는 궤도 통제 장치의 "반환 각도(return angle)"와 패들과의 거리(또는 화면의 가장 아래 부분)에 따라 게임을 극도로 어렵게 만들 수도 있다.

공의 속도 증가 장치 - 속도 증가 장치는 보통 속도를 증가시키며 공을 튕겨내는 전형적인 핀볼 스타일의 형태를 띠고 있다. 하지만 해당 영역을 공이 통과하면 공의 속도를 증가시키는 "속도 증가" 속성을 가진 플레이 영역으로 표현될 수도 있다.(플레이어의 타이밍 감각을 교란할 때 좋음)

움직이는 파워 업과 파워 다운 - 몇몇 게임들은 아이템을 획득하기 어렵도록 파워 업에 제한적인 움직임을 부여하기도 한다.(진동, 경로 기반, 논리 기반) 움직이는 파워 다운(플레이어를 쫓아가는)도 게임의 난이도를 높이는 아주 좋은 방법이다.

스위치(공 고정, 트리거 영역, 버튼) - 스위치는 종종 대량의 일반 벽돌이나 부서지지 않는 벽돌을 파괴해 레벨의 "보상 영역(reward area)"을 여는데 사용되고는 한다. 또 스위치는 추가적인 목표를 추가시키거

나 하는 등의 보스를 활성화시킬 때 사용될 수도 있다. 공 고정은 트리거로 사용될 수도 있고 궤도 조작 장치로 사용될 수도 있으며 공을 잡은 다음에 다른 스위치, 또는 폭탄(하단 참조)으로 발사할 수도 있다.

공 함정 – 공 함정은 벽돌 더미에 있는 하나나 그 이상의 공을 담고 있는 작은 구멍이다. 플레이어가 공 함정을 뚫을 수 있을 만큼 충분한 벽돌을 파괴하면 공 함정에 들어있던 공이 풀려난다. 공 함정은 또한 안에 들어있는 공을 풀려면 공으로 맞추거나 파손시켜야 하는 물체(나무 상자, 유리 상자 등)의 형태를 취하고 있을 수도 있다.

폭탄 – 폭탄은 자유롭게 이동할 수도 있지만 보통은 한자리에 고정되어 벽돌 더미에 둘러싸여 있다. 공으로 폭탄을 맞추면 폭발하며 작은 덩어리의 벽돌을 깎아내거나 거대한 연쇄반응을 유발시킨다.

방어막 – 방어막은 공이 화면의 바닥으로 떨어지는 것을 막아주는 특별한 파워 업이다. 방어막은 일정 시간 동안만 유지되거나 공에 한번 맞을 때마다 방어막의 한 층이 사라지게 된다.

보이지 않는 벽돌 – 이런 벽돌은 처음에 투명한 상태로 시작하며 그 중 하나를 공으로 맞추고 잠시 후에야 눈에 보이는 상태가 된다.

진행형 벽돌 – 공이 패들에 맞을 때마다 벽돌의 벽이 화면의 바닥을 향해 조금씩 내려온다. 플레이어는 벽돌 벽이 화면 끝까지 내려오기 전에 하단 부분을 파괴하는데 주력해야 한다.(또는 간단하게 최대한 많은 점수를 얻으려 노력하던지)

여러 개의 패들 – 이 기능은 플레이어에게 하나의 패들 위에 또 하나의 패들이 표시되는 식으로 두 개의 패들을 제공한다. 추가적으로 패들 사이의 간격을 조절할 수 있거나, 각각의 패들이 독자적인 크기를 가진다거나, 특정 패들을 사용하면 얻는 점수가 변경된다거나 하는 등

의 선택사항이 존재한다.

현대의 벽돌 깨기 스타일 게임의 새로운 기능

google에서 “breakout game”을 검색해보면 분명히 엄청난 양의 다운로드 가능한 데모 버전의 링크가 표시될 것이다. 이런 벽돌 깨기 복제품의 확산은 초보 프로그래머가 만든 조잡한 벽돌 깨기 시제품(prototype)들이 만연함에 따라 점점 더 증가 속도가 빨라지고 있다. 이 중 대다수의 게임들은 꽤 재미있지만 실제로 고전적인 *벽돌 깨기* 스타일 게임 메카닉의 신선함은 이미 사라진 지 오래이므로 현대의 벽돌 깨기 스타일 게임 제작에는 새로운 기능과 차별화가 점점 더 중요해지고 있다.

다음에 명시된 내용은 이 장르에서 좀 더 최근에 소개된 기능과 함께 실제로 해당 기능을 체험해볼 수 있도록 다운로드 가능한 몇 개의 게임을 소개하고 있다.

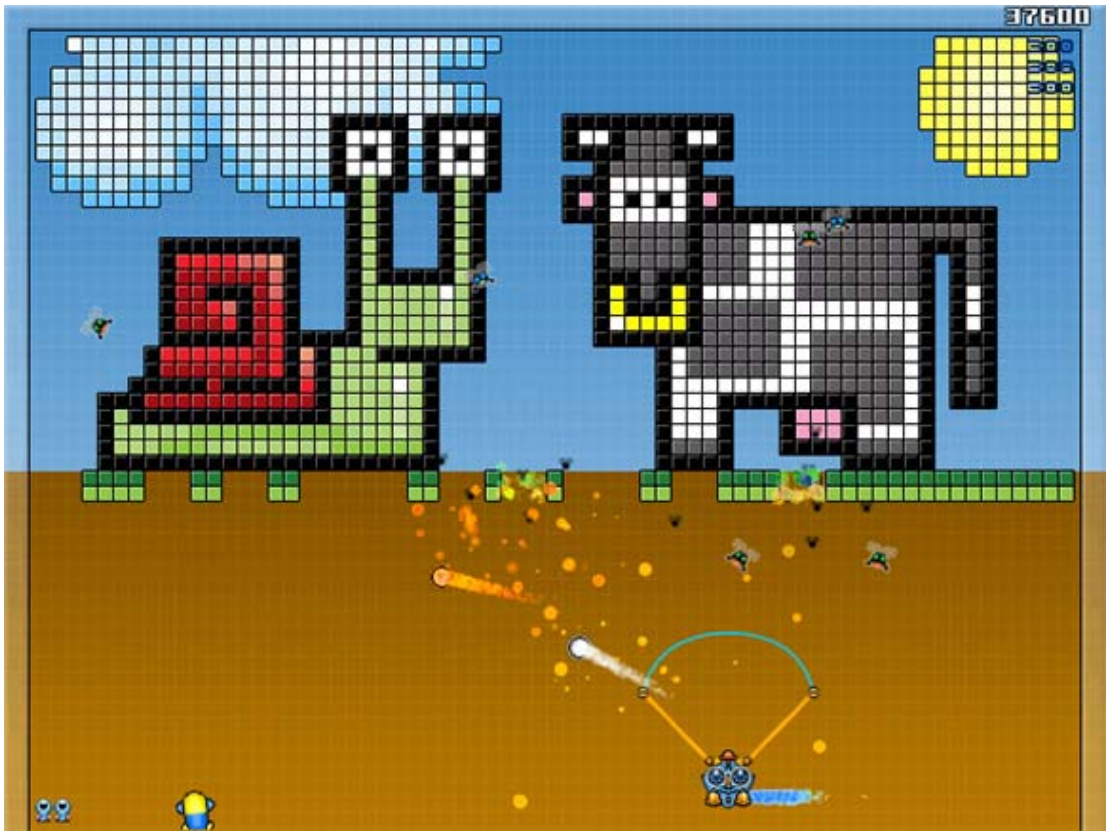
대부분의 게임 객체가 상당히 많이 움직임 – 오늘날의 벽돌 깨기 스타일 게임들은 간혹 플레이어에게 상당한 분량의 애니메이션을 보여줘 매우 역동적으로 보이도록 제작된다. 벽돌이 단순히 앞뒤로 움직이거나 미리 정의된 좌표를 따라 이동한다던 지부터 게임 플레이 환경과 플레이어에 반응해 상호 작용을 하며 움직이는 “똑똑한(smart)” 객체까지 말이다. 이런 움직임에 대한 훌륭한 예제로는 *리코쳇: 잃어버린 세계(Ricochet: Lost Worlds*, http://www.ricochetlostworlds.com/)을 들 수 있다.

고품질의 물리/역학 시뮬레이션 – 현대 CPU의 빠른 속도로 인해 개발자들은 이제 공뿐만이 아니라 게임에 등장하는 모든 객체에 고품질의

물리 시뮬레이션을 수행할 수 있게 되었다. 더 자세한 정보는 *브레이크 퀘스트(BreakQuest)*와 다이나모 역학 라이브러리(DynaMo motion library, http://home.iae.nl/users/starcat/dynamo)를 확인하라.

입자(Particle) 효과 - 그래픽 가속기의 발달과 강력한 CPU 덕에 입자 효과도 사용할 수 있게 되었다. 불꽃, 백열광, 입자 분수와 같은 효과들은 정적인 레벨에 생동감을 불어넣어 준다. 특히 "공의 궤적"은 공의 움직임을 더 쉽게 알아볼 수 있도록 도와준다. *LEGO 브릭토피아(LEGO Bricktopia)*는 공의 위치를 최대한 명확하게 표시하기 위해 배경 텍스처 섭동과 공의 궤적을 사용하고 있다. (http://www.largeanimal.com/games/deluxe/lego-bricktopia) 그리고 이런 기술을 활용하는 게임에는 *플라즈마 퐁(Plasma Pong*, http://www.plasmapong.com/)이 있다. 이 게임은 실제로 *벽돌 깨기* 스타일 게임은 아니지만(그냥 퐁이다...) 입자 효과와 함께 인상적인 유체 역학 시뮬레이션을 사용하고 있다.

움직이는 벽돌 외의 다른 물체 - 몇몇 게임(*브레이크 퀘스트*와 같은, http://www.nurium.com/)은 벽돌을 없애버리고 벽돌 대신에 매우 움직임이 많은 목표 객체를 사용하기도 한다. 그 결과 플레이의 다양성이 상당히 증대되었다. *마법의 공 3(Magic Ball 3*, http://www.alawar.com/games/magic-ball-3/)에서도 매우 다양한 목표 객체들이 등장하고 있다.



이야기 형식과 다중 레벨 목표 – 벽돌 깨기 스타일 게임의 동기 부여 방법은 화면에 "빠각! 펑! 뽀각! 화면의 상단에 벽돌로 구성된 벽이 등장했습니다."와 같은 글자가 표시되는 것에서 많은 발전을 해왔다. *펑키볼 어드벤처(Funkiball Adventure*, http://www.funkitron.com/games/funkiballadventure)를 살펴보라. 벽돌 깨기 레벨에서 도난 당한 미술품을 찾으러 다닐 날이 올 줄을 누가 상상이나 했을까?

유머 – *자르디나인 2(Jardinains 2*, http://www.jardinains.com/)와 같은 게임들은 벽돌 깨기 스타일 플레이에 캐릭터와 유머를 가미했다. 이 게임을 플레이 할 때에는 꼭 소리를 켜고 하라.

패들 튕기기 – *펑키볼 어드벤처(Funkiball Adventure)*는 패들을 "상승" 시켜 화면의 가장 아랫부분에서 벗어나 공을 좀 더 빨리 쳐내고 더 강하게 쳐낼 수 있도록 한 최초의 *벽돌 깨기* 스타일 게임이다. 매우 강

하게 쳐내서 공에 맞은 벽돌 주변에 피해를 전달하는 것을 제외하더라도 이렇게 조작을 간단하게 변경함으로써 공의 궤도 조절 능력이 매우 향상되었다.(http://www.funkitron.com/games/funkiballadventure)

패들 생성 - *LEGO 브릭토피아(LEGO Bricktopia)*는 내가 알고 있기로는 패들의 구조를 플레이어가 LEGO 블록을 쌓아 상당부분 변경할 수 있는 최초의 게임이다.(http://www.largeanimal.com/games/deluxe/lego-bricktopia)

고급 궤도 변경 장치 - *페이퍼 볼(Paper Ball)*은 레벨 전체에서 공이 완만한 코스를 따라 진행하도록 궤도를 변경하는 장치를 매우 광범위하게 사용한 게임이다.(http://www.e-giraffa.com/)

3D 그래픽 - 현대의 3D 그래픽 엔진을 사용한 벽돌 깨기 스타일 게임은 대부분이 공의 움직임은 2D적이지만 목표 객체를 여러 개의 평면을 쌓아놓은 것처럼 묘사하고 있다. 이로 인해 가장 밑에 있는 벽돌을 파괴하면 위에 쌓여있던 벽돌이 내려와 여러 번 맞춰야 파괴되는 벽돌의 변형판과 같은 역할을 하게 된다. *매직 볼 3(Magic Ball 3)*는 실제 물리학을 적용해 3D 객체가 쓰러지고 구르며 레벨에 있는 다른 객체에 영향을 끼치고, 그로 인해 예상할 수 없는 혼란이 발생하기도 한다.

시스템 설계, 플레이 영역

삼위일체 – 패들, 공, 벽돌

아주 당연한 말이지만 모든 벽돌 깨기 스타일 게임은 최소한 3 가지의 사항을 공통적으로 가지고 있다. 바로 패들, 공, 공으로 맞출 목표 객체가 그것이다. 새로운 벽돌 깨기 스타일 게임을 구상하고 있다면 어떤 것을 먼저 설계해야 할 것인지는 딱히 말해주기가 어렵다. 게임이 엄청나게

다양한 목표들에 기반하고 있다면 그 중에 가장 중심이 되는 목표가 어떤 종류의 패들과 공을 사용할 것인지를 결정하는데 도움을 줄 것이다. 그와는 반대로 게임이 “슈퍼 패들”이나 우아한 모양의 공을 가지고 진행한다면 플레이 영역에서 사용할 객체의 초기 속성값을 결정할 때 패들이나 공의 속성을 가지고 객체를 구상하면 될 것이다.

벽돌과 플레이 영역 크기

시제품(prototype) 레벨을 설계할 때 처음에 해야 할 작업 중에 하나는 바로 플레이 영역의 크기에 비례한 벽돌(혹은 다른 목표 객체)의 크기를 결정하는 것이다. 작은 크기의 벽돌로는 상징적인 레벨과 복잡한 플레이 영역을 설계하기가 쉬워진다.(하단의 레벨 원형 참조) 큰 벽돌을 사용하면 레벨 설계를 빠르게 진행할 수 있지만 레벨이 지루하게 보이지 않도록 하려면 다양한 종류의 벽돌을 사용해야 할 것이다.

플레이 영역과 벽돌의 크기를 결정했다면 그 다음에는 레벨에서 벽돌의 밀도를 결정해야 할 것이다. 레벨에서 몇 개의 벽돌이 표시되도록 할 것인가? 이것은 공/패들의 파괴력(하단의 공의 속도와 피해치 참조), 벽돌의 종류, 게임에 가장 잘 어울리는 파워 업의 종류를 결정하는데 도움을 줄 것이다.

빈 공간

벽돌 더미를 둘러싼 빈 공간(공백 부분과 플레이 영역의 배경)도 고려를 해야 한다. 빈 공간은 목표 객체의 구조를 두드러지게 보이도록 해주며 공이 얼마나 흥미롭게 움직일 지를 결정한다. 화면에 지나치게 많은 수의 벽돌을 배치하면 공의 움직임을 제한하게 될 것이며 레벨의 시작 부분을 지루하게 만들고 말 것이다. 혹시라도 레벨에 벽돌을 많이 등장시키려 한다면 빠르게 빈 공간을 만들 수 있는 파워 업, 스위치, 폭탄의 사용을 고려해보라.

배경

고전 벽돌 깨기의 레벨 구조에서는 간단한 검은색 배경이 게임에 훌륭한 대비를 제공해줬었지만 현대의 게임에서는 애니메이션이 추가된 배경이나 공이 지나가면 그에 반응을 하는 배경을 사용하고 있다.(예를 들어 공이 배경 이미지/텍스처를 비추는 것과 같이)

플레이 속도

게임이 “느리고 은밀한” 분위기인가, 아니면 “빠르고 격렬한” 분위기인가? 공의 기본 속도, 기본 수직 이동 속도, 패들의 충돌 속도 등은 게임의 성향을 결정하는 중요한 사항이다. 이를 먼저 고려한다면 속도를 조절함으로 인해 게임의 분위기에 맞도록 오디오/그래픽을 바꿔야 하는 일을 피할 수 있을 것이다. 게임의 속도를 단 한번의 플레이로 결정할 필요는 전혀 없으며 항상 목표 소비자층에 맞는 최상의 속도를 결정하기 위해 플레이 테스트를 하도록 하라.

패들

초기에 결정해야 할 중요한 사항 중 하나는 패들을 어떤 식으로 조작하고, 패들의 표면이 공을 어떤 식으로 튕겨내고, 플레이어에게 패들에 대한 통제권을 어느 정도까지 허용할 것인지를 결정하는 것이다. 개발 과정 동안 “공이 사라지면 플레이어가 자신들의 실력을 탓할 것인가, 패들을 탓할 것인가?”를 심사숙고 해보라.

패들의 속도와 크기

다음은 꼭 기억해야 할 중요한 두 가지 규칙이다. (1) 패들의 수평 이동 속도가 빨라질수록 플레이어가 공을 쫓아가는 것이 쉬워진다. (2) 패들의 표면 면적이 넓어질수록 플레이어가 공을 쳐내기가 쉬워진다.

이제 느리고 넓은 패들, 빠르고 좁은 패들의 두 가지 패들 원형을 살펴보자. 기능적으로는 거의 유사한 방어 범위를 가지고 있지만

그렇다고 두 가지가 서로 같은 종류의 플레이 경험을 제공하는 것은 아니다.

느리고 넓은 패들 – 느리고 넓은 패들은 플레이어가 공의 반사 궤도를 예상해야 하기는 하지만 패들의 면적이 넓어 패들과의 접촉 지점(공이 패들과 충돌하는 지점)은 그리 크게 신경을 쓰지 않아도 되므로 게임을 좀 더 가볍게 즐길 수 있다.

빠르고 좁은 패들 – 빠르고 좁은 패들도 플레이어가 공의 반사 궤도를 예상해야 하지만 접촉 지점이 작으므로 지속적으로 주의를 기울여야 한다. 이로 인해 플레이어가 튕겨서 돌아오는 공을 받아내려고 시도할 때의 긴장감이 매우 증가하게 된다.

정적인 레벨과 아주 소수의 파워 업만을 가지고 있는 "순수한" 게임이라면 약간 좁고 약간 느린 패들을 사용하는 것이 좋을 것이다. 움직이는 객체가 많고 레벨에 파워 업도 많이 등장하는 게임에서는 많은 물체들이 플레이어의 주의를 끌 수 있으므로 약간 더 빠르고 큰 패들을 사용하는 것이 좋다.

움직이는 벽돌은 공의 반사 각도를 판단하기가 매우 어렵게 만드니 게임에서 다수의 움직이는 벽돌을 사용한다면 패들의 기본 크기를 키우는 것이 좋을 것이다. 물론 게임이 진행될수록 이런 균형을 무너뜨릴 수 있는 방법을 준비해야 할 것이다.

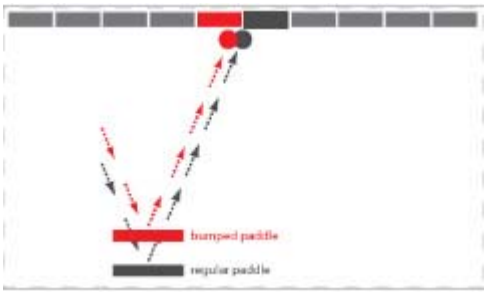
패들 튕기기

패들 튕기기는 새로 추가된 재미있는 형태의 상승 형태로써 *펑키볼 어드벤처(Funkball Adventure)*와 *LEGO*

*브릭토피아(LEGO Bricktopia)*에서 선보여졌다. 이 기능은 패들을 위쪽으로 약간 상승시켜 공을 좀 더 빨리 쳐낼 수 있도록 하는 기능이다. 이는 플레이어에게 더 넓은 사격 각도를 확보해 주며 공을 파워 업시켜 피해치와 속도를 증가시키기 쉽게 만들어 준다.

패들 튕기기를 사용하려 한다면 다음과 같은 5 가지를 먼저 고려해봐야 한다. (1) 패들을 얼마나 더 높게 상승시킬 것인가? (2) 패들이 얼마나 빠르게 상승할 것인가? (3) 패들이 얼마나 빨리 원위치로 돌아올 것인가? (4) 패들 튕기기로 인한 공의 속도나 영역 피해치 등의 증가량은 얼마나 증가할 것인가? (5) 패들 튕기기를 연속적으로 사용하면 공의 파괴력을 계속해서 증가시켜 줄 것인가가 바로 그것이다.

패들 튕기기에는 여러 이점이 있다. 가장 큰 이점은 위 그림에서 보여지는 것처럼 플레이어가 공을 통제할 수 있는 영역이 넓어진다는 것이다. 공이 반사되어 오는 각도는 한 번의 타격으로는 별 차이가 없지만, 수백 번 연속으로 반복했을 시에는 1 차원적인 패들보다 상대적으로 더 이점이 많아지게 된다. 패들 튕기기가 공의 속도까지 증가시키게 된다면 정적인 게임 레벨에서 더 큰 도움이 된다.(참고: 파워 업된 공이 상승되지 않은 패들에 맞을 경우, 공의 상태를 원래대로 복구하는 것이 좋다.)



패들 튕기기는 또한 플레이어에게 공이 돌아오기까지 심심하지 않도록 재미를 선사하기도 한다. 이상하게 들릴지는 모르겠지만, 아무 생각 없이 패들을 튕기는 것은 굉장히 재미있다! 특히 튕길 때 시각적인 효과나 음향 효과가 첨가된다면 재미도 배가 된다. 좋아하는 음악에 맞춰 발을 구르는 것과 비슷하다고 할 수 있을 것이다.

마지막으로 패들 튕기기는 게임에서 좀 더 공간감을 느낄 수 있도록 해준다. 플레이어를 화면의 맨 아래쪽에 가둬놓는 대신에 플레이 영역에서 작게나마 움직일 수 있는 공간을 주는 것이다.(참고: 패들 튕기기를 사용하지는 않지만 패들이 플레이 영역으로 올라갈 수 있도록 허용하는 게임의 예로는 Arkedo 사의 Nintendo DS 용 게임인 *너버스 브릭다운[Nervous Brickdown]*을 들 수 있겠다.)

패들의 형태

초창기의 벽돌 깨기 게임에서는 평평한 표면을 가진 사각형의 패들을 사용했었다. 그래서공의 궤도는 패들의 모서리 부분에 맞으면 좀 더 급격한 각도로 튕겨져 나가는 것과 같이 공이 패들의 어떤 부분에 맞았는가에 영향을 받았다.

현대의 벽돌 깨기 스타일 게임들에서는 반구형에서부터 2등분한 사다리꼴 모양(위와 아래는 평평하고 옆면은 쐐기꼴 모양)까지 좀 더 다양한 패들 모양을 사용하고 있다. 이런 패들들은 완전히 평평한 패들 표면보다 공을 어떤 방식으로 튕겨낼 지에 대해 더 많은 통제력을 제공해주고 있다. 2등분한 사다리꼴 모양의 패들과 패들 튕기기를 함께 사용하면 공 통제력을 놀랄만한 수준으로 끌어올릴 수 있다.(*LEGO 브릭토피아[LEGO Bricktopia]*)

패들의 마찰력

속도, 크기, 모양, 튕기기 외에도 패들 표면의 힘 전달력과 같은 속성도 고려해봐야 한다. 특히나 패들과 공이 충돌했을 때 패들의 수평 속도가 공에 얼마나 전해지는 가와 같은 마찰력을 말이다. 패들의 마찰력은 사용할 수도 있고 사용하지 않아도 상관없다. 개인적으로 나는 패들의 마찰력이 플레이어의 공에 대한 통제력을 더해주고 약간의 현실감을 부여해주기 때문에 마찰력을 사용하기를 선호한다. 패들의 마

찰력은 다음과 같은 3가지의 기본적인 항목으로 구분할 수 있다.

반사 패들 - 이것은 마찰력이 0이라는 것과 같다. 이런 패들은 마치 고정된 유리벽과 같이 공에 회전을 걸지 않으며 공의 에너지를 100% 반사한다. 물론 실생활에서는 공이 어떤 표면에 맞는다면 일정한 에너지를 빼앗기겠지만 벽돌 깨기에서는 이런 규칙을 "깨고" 계속 공이 빠르게 움직이도록 한다. 실제로 이런 패들 시뮬레이션은 가장 최소한의 CPU 사이클(cycle)만을 소모하기는 하지만 공의 모든 움직임을 꽤 정확하게 예측할 수 있으므로 플레이의 역동성이 떨어지게 된다.

간단한 마찰력을 가진 패들 - 이 패들의 표면은 수평 운동량의 일부를 공에 전달해 공의 궤도를 변경하게 된다. 예를 들어 패들이 공과 같은 방향으로 이동하고 있고 공이 가파른 각도로 패들의 표면에 접촉한다면 공의 수평 운동량이 줄어들어 좀더 수직에 가깝게 패들에서 튀어오르게 된다.

마찰력과 회전력을 사용하는 패들 - 다음 단계로는 간단한 마찰력을 사용할 뿐만이 아니라 공에 회전력을 추가할 수도 있는 패들이다. 이런 패들의 표면 같은 경우에는 공의 선형적인 속도뿐만이 아니라 각속도(Angular velocity)에도 영향을 미치게 된다. 이렇게 회전하며 튕겨나가는 공은 마치 회전량에 따라 다르게 튕겨나가는 회전하는 당구공과 같이 다른 표면과 충돌했을 때의 각도가 변화하게 된다.

또 다른 패들 개선 사항(슈퍼 패들)

대부분의 패들이 공의 궤도만을 변경하고 속도는 거의 변경하지 않는 반면에 튕겨나가는 공에 상당량의 에너지를 더해주는 패들을 가진 게임들도 존

재한다.(예를 들어 패들 튕기기와 같은 수단을 사용해) 그런 "슈퍼 패들"의 경우에는 마찰력이나 회전력을 사용해야 할 필요가 없다. 슈퍼 패들에 대한 예제로써는 *펑키볼 어드벤처(Funkiball Adventure)* 또는 *LEGO 브릭토피아(LEGO Bricktopia)*를 참조하라.

좀 이상할 지도 모르겠지만 나는 언젠가 마우스 오른쪽 버튼 클릭으로 사용할 수 있는 패들 중에서 플레이어가 패들을 즉각적으로 변경할 수 있는 *벽돌 깨기* 스타일 게임이 등장하기를 바라고 있다. FPS 게임들을 보면 여러 개의 총기를 가지고 다닐 수 있지 않는가? 그와 유사하게 나는 플레이어가 공을 숨기거나, 어두운 레벨에서는 스스로 빛을 발하는 공으로 변경하거나, 심지어 일시적으로 형태가 없는 공으로 변경할 수도 있다면 좋을 것 같다는 생각을 한다.(마우스 오른쪽 버튼 클릭으로 말이다.) 이로 인해 레벨 설계의 가능성이 더 넓어질 것이다!

공

공의 속도와 피해치

고전적인 벽돌 깨기 게임에서는 공의 속도가 고정되어 있고 한 번 충돌 시마다 하나의 벽돌을 파괴한다는 식으로 공의 속도와 피해치가 한정되어 있었다. 몇몇 게임들에서 파워 업을 사용해 공의 속도와 피해치를 변경하거나 플레이어가 더 높은 게임 레벨에 도달할수록 기본 속도를 높인다거나 하기는 했지만 기본적으로 공은 동일한 속도로 움직이고 동일한 양의 피해치를 입힌다는 이 규칙은 변함이 없었다. *펑키볼 어드벤처(Funkiball Adventure)*는 연속적으로 공을 맞추면 공의 속도와 파괴력이 증가시킬 수 있는 패들 튕기기를 소개하며 이 틀을 깨버린 최초의 게임이다.

이 게임에서 공의 속도와 파괴력이 높아지면 공이 깨지거나, 주변 벽돌을 완전히 파괴하는 영역 피해를 입히기도 한다. 간단한 파워 업 외에 이 기능의 추가를 고려해볼 만한 이유로는 다음의 몇 가지가 존재한다. 첫 번째로 이 기능은 파워 업을 얻었을 때 무작위로 속성을 변

경하는 것이 아니라 항상 공의 속도와 파괴력을 완전히 플레이어의 손으로 제어할 수 있도록 해준다. 둘째로 실력이 부족하거나 공을 자주 놓치거나 패들 튕기기를 사용하지 않는 플레이어는 공의 속도와 파괴력이 떨어지는 것과 같이 플레이어가 게임의 속도와 역동성을 자신이 스스로 조절할 수 있게 된다.

공의 움직임 시뮬레이션

모든 *벽돌 깨기* 스타일 게임의 중요한 요소는 바로 공의 움직임을 자연스럽게 시뮬레이션하는 것이다. 일반적으로 공은 어느 정도 예상 가능한 움직임을 가지고 시작하지만 파워 업(또는 패들 튕기기)를 통해 환상적인 움직임을 나타내게 된다. 이제 파워 업이나 다른 속성들이 적용되지 않은 일반적인 상태부터 시작해서 움직임 시뮬레이션에 관해 좀 더 자세히 살펴보도록 하자.

튕기는 공 – 이것은 아무런 장식도 하지 않은 *퐁(Pong)* 스타일의 공이다. 이것은 사방으로 튀어 다니며 물체와 충돌할 때마다 에너지를 얻거나 잃는다. 그리고 완벽한 무게 중심을 가지고 있으며 회전을 하지 않는다.

회전하며 튕기는 공 – 표면 마찰력을 사용하는 게임에서는 공이 회전을 할 수도 있다. 가장 간단한 구현 방법은 안정적이고 "완벽하게 중심이 잡힌" 완전한 구체를 사용하는 것이다. 회전은 패들(또는 다른 모든 게임 내의 물체 표면)에 맞고 공이 튕겨나올 때의 궤도를 수정하기 위해 사용하지만 실제로 그리 복잡한 공의 움직임을 만들어내지는 못한다.

복잡한 공 – 좀 더 복잡한 공(구체가 아닌 공, 반고체, 복합적인 물체)의 경우에는 무게 중심이 "얽히고 설켜 있으며" 어쩌면 공의 어떤 표면이 패들과 접촉했는가에 따라 달라지는 마찰력을 가지고 있을 수도 있다. 복잡한 공에는 다음과 같은 공이 포함된다. 달걀 모양의 공, 한

쪽 모서리에 돌이 박힌 사각형 고무 스펀지, 사슬로 연결된 두 개의 농구공.

내부를 들여다보기

고전적인 벽돌 깨기 스타일의 공 시뮬레이션은 상당히 쉽게 구현할 수 있다. 하지만 일단 회전이 추가되거나, 복잡한 공을 사용하거나, 완전한 물리 시뮬레이션을 적용하거나 한다면 구현의 복잡성이 급격히 증가하게 된다. 그래서 나는 Large Animal Games의 *LEGO 브릭토피아(LEGO Bricktopia)*에서 리드 프로그래머를 맡았던 Josh Welber에게 공 시뮬레이션 관리에 대한 몇 가지 조언을 부탁했다.

간단한 움직임 시뮬레이션 관리에서 공의 움직임이 너무 복잡해지는 명확한 지점이 존재합니까?

Josh Welber: 다음과 같은 고전적인 벽돌 깨기 메카닉을 사용하는 간단한 움직임 시뮬레이션에서는 공이 충돌 시에 완전 반사가 되게 하면 됩니다. (a) 모든 장애물이 사각형으로 구성됨. (b) 벽은 항상 한 자리에 고정되어 있음. (c) 공이 항상 동일한 속도로 움직임.

게임에 사각형이 아닌 모양의 장애물과 벽, 애니메이션이 있는 장애물을 추가하거나 공의 움직임을 좀 더 현실적으로 만들고 싶다거나, 어떤 변수를 추가하기 시작한다면 좀 더 일반적인 물리학 해법의 도입을 고려해보는 것이 좋습니다.(직접 제작하던지, 다양한 오픈 소스 패키지 중 하나를 사용하던 지와 같이, http://en.wikipedia.org/wiki/Physics_engine).

개발자들이 자신의 벽돌 깨기 게임에 완전한 물리 시스템을 사용하려 할 때 고려해야 할 사항이 있나요?

JW: 좀 더 일반화된 물리 시뮬레이션을 사용할 때 까다로운 부분은 시뮬레이션이 모든 것을 통제하려고 든다는 것입니다.

예를 들어 인공적으로 공의 속도나 게임 속의 다른 속성을 변경하려고 든다면 시뮬레이션이 오작동을 하는 경우가 있죠. 대부분의 견고한 시스템들은 융합/충돌(integration/collision) 순환의 특정 부분에서만 이런 행동을 허용할 것입니다.

하지만 몇몇 시스템들은 여러 단계에 걸쳐 충돌을 해소해 속도 벡터를 구하는 것에 의존하고 있으며 이런 경우에 융합 부분에서 물리 시뮬레이션을 "속이면" 충돌의 결과가 잘못될 수가 있습니다.

***LEGO 브릭토피아(LEGO Bricktopia)*에 빠르거나 느린 공 파워 업과 패들 튕기기 기능을 제작하면서 그런 점을 어떻게 해결하셨나요?**

JW: 공의 속도를 조절하면서도 사실적인 공의 움직임을 구현하기 위

해 저희는 공의 속도가 목표 속도와 다른 범위 안에 있는 경우에 약간의 힘을 가해 인공적으로 공의 속도를 높이거나 늦출 수 있는 시스템을 제작했습니다. 움직임을 느리게 만들어야 하는 경우에는 그저 "불릿 타임(bullet time" 편법을 사용해 시뮬레이션 시간을 조절했죠.(특정 상황에서는 전체 시뮬레이션을 느리게 만들기도 했습니다. 즉, 시뮬레이션에 실제로 프레임 간에 흐른 시간을 알려주는 대신에 그 절반의 시간만 흘렀다고 가정해 전체적으로 모든 것이 느려지게 하는 것이죠.)

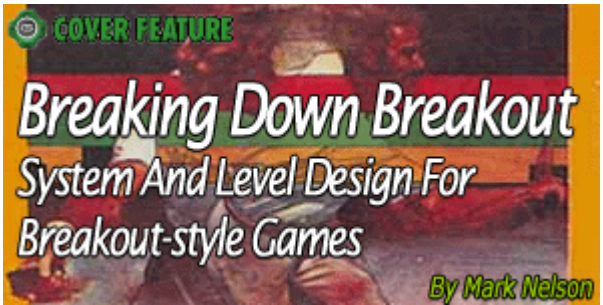
패들 튕기기도 마찬가지로 원하는 효과를 나타내기 위해 뭔가 인공적인 힘을 가할 필요가 있었죠. 실제적으로 보면 패들은 공에 비해 질량이 매우 큰 편이므로 현실에서 그런 것을 하면 "엄청나게 빠르게 튕겨"나가 버려서 공의 속도가 굉장히 빨라질 것입니다. 하지만 위에 말씀 드린 힘 보정 시스템 덕에 공의 속도가 너무 빠르다면 급격하게 느려지게 되어 있죠.

그래서 결과적으로 예측할 수 없는 공의 속도 변화가 일어나지 않는 만족스러운 패들 튕기기 기능이 구현되었습니다.

_LEGO 브릭토피아_의 공 시뮬레이션 중에서 당신이 가장 자랑스럽게 생각하는 것은 무엇입니까?

JW: 벽돌 쌓기/패들 메카닉을 구현한 것과 그것을 공과 잘 조화되도록 한 것입니다. 이 게임에서 플레이어는 자신의 패들에 LEGO 벽돌을 붙여서 패들의 모양을 변경하고 레벨을 완수하는 데 아주 다양한 전략을 사용할 수 있게 되었죠. 이 벽돌 쌓기 메카닉을 전체적인 물리 시뮬레이션과 융화시키는 것은 정말 힘들었지만 최종 게임에서 고생한 만큼 결과물이 잘 나왔습니다.

레벨 설계



일반적인 구조물 원형

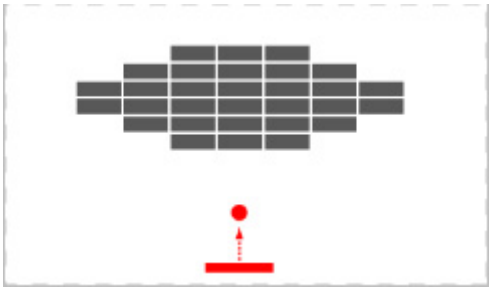
대부분의 게임들이 추가적인 객체 종류를 추가하거나 벽돌 깨기 게임의 간단한 벽돌 더미 이상의 레벨 설계를 보여주고 있기는 하지만 놀랄 정도로 많은 수의 원형들이 현재까지도 남아 플레이어에게 과거를 회상하도록 만들어 주고 있다.

첫 번째 레벨을 부분적으로라도 친숙한 모습으로 설계하면 캐주얼 플레이어들의 독특한 게임의 시스템에 대한 거부감을 완화시킬 수 있다. 특히 이전의 벽돌 깨기 스타일 게임들과 시스템이 근본적으로 다르다면 말이다. 하지만 하드코어 게이머를 대상으로 한 게임을 제작 중이라면 게임 시스템에 대한 거부감을 "완화"시킬 필요가 없을 수도 있다. 그들은 대부분이 벽돌 깨기 장르와 굉장히 친숙할 것이며 "전혀 딴판인" 게임을 찾으려 들 것이기 때문이다! 이런 경우라면 나는 첫 번째 레벨에서 강한 인상을 심어주기 위해 "독특한" 레벨을 설계하기를 권한다.(참고: 매우 추상적이고 별난 모양의 목표 객체를 사용하려 한다면 해당 객체의 속성을 플레이어가 확실히 알아볼 수 있도록 하라.)

일반적인 레벨 원형

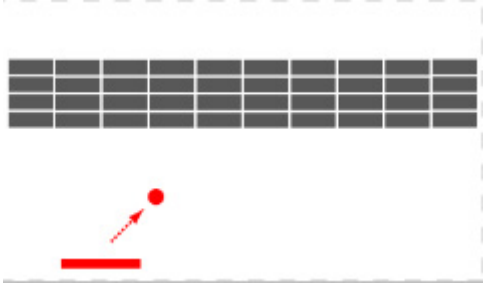
1980년대 초반의 벽돌 깨기 스타일 게임을 플레이 해본 경험이 있거나 스크린샷을 본적이 있다면 벽돌 깨기 레벨 설계가 더 이상 벽돌을 단순하게 4개에서 8개의 줄로 구성되지 않는 다는 것을 잘 알 것이다. 비록 그것이 벽돌 깨기 스타일 게임의 고전적인 벽돌 더미이기는 하지만 설계에 고려해 볼만한 다른 원형들도 존재한다.

중앙 집중(Central Mass)



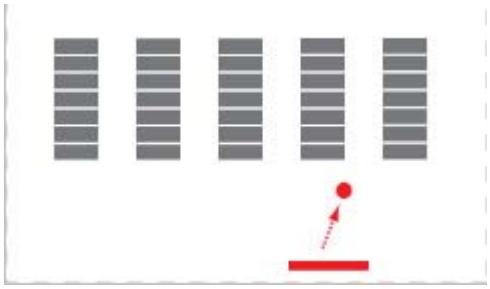
중앙 집중 형태에서는 벽돌 더미가 사방이 빈 공간의 중앙에 배치된다. 이 경우 벽돌 더미의 형태는 흔히 볼록 다각형의 형태를 띠며 간혹 오목 다각형의 형태를 취하기도 한다. 조언: 중앙 벽돌 더미의 내부에 고정적인 목표 객체를 배치하거나 벽돌 더미 위에 움직이는 목표를 배치하면 플레이어의 초기 선택이 매우 달라지게 된다. 벽돌 더미 주위를 회전하는 동적 목표를 배치하면 강제적인 시간 기반 구성 요소를 삽입할 수 있을 것이다.

벽(Wall)



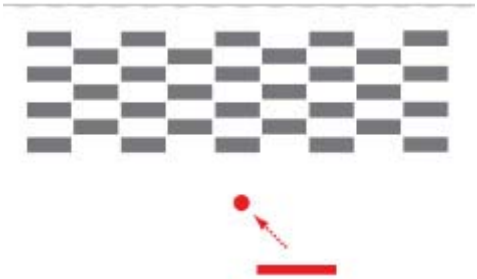
고전적인 벽돌로 구성된 벽으로써 흔히 벽돌 더미의 위와 아래에 빈 공간을 가지고 있다. 하지만 간혹 각 층간에 빈 공간이 있는 여러 층의 벽돌로 구성되기도 한다. 이 형태는 화면의 왼쪽 끝에서 오른쪽 끝 부분까지 늘어서 있거나 잘 튕겨낸 공이 밑바닥에서 화면의 맨 위까지 통과할 수 있도록 양 옆에 약간의 빈 공간을 가지고 있는 경우가 많다. 조언: 벽돌 더미의 중심 부분이나 맨 윗줄의 벽돌에 파워 업을 배치해 놓으면 플레이어에게 흥미로운 도전 목표를 부여할 수 있으며 이 형태가 지루해지는 것을 막을 수 있다. 아래쪽의 벽돌들을 오목한 모양으로 배치하면 처음에 공을 발사했을 때 움직임을 다양화시킬 수 있다.(하단의 동굴 모양 참조)

기둥(Columns)



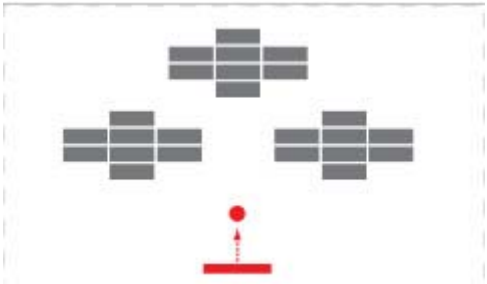
이런 형태의 레벨은 각 기둥 사이에 다양한 크기의 빈 공간이 있는 여러 개의 수직 기둥들로 구성된다. **조언:** 쐐기꼴 모양의 벽돌 더미를 혼용하거나 레벨의 천정(화면의 맨 위쪽) 부분에 목표 객체를 배치하면 플레이어의 초기 선택의 폭을 넓혀줄 수 있다.

그물(Mesh)



고정된 그물 모양은 보통 벽돌을 격자와 유사한 형태로 배열한다. 하지만 움직이는 경우에는 초기에는 마치 중앙 집중, 기둥, 벽과 같은 형태로 배열되지만 점차 다른 형태로 배치가 변형되며(몇몇 벽돌이 이동되어) 공을 그 사이에 가둘 수 있는 그물 형태가 된다. **조언:** 시스템이 다중 위치(multi-coordinate)나 논리 기반 움직임(logic-based motion)을 지원한다면 그물의 형태를 3~4가지 형태로 정렬해보라.

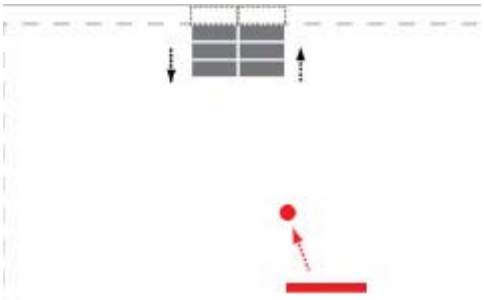
범퍼(Bumpers)



범퍼 레벨은 그물 형태와 유사하지만 벽돌 집단이 더 적고 집단 간의

빈 공간이 훨씬 더 크다. 또한 범퍼는 전체가 동일하게 움직이지 않고 각 집단이 독립적으로 움직인다는 것도 차이점이다. **조언:** 게임에 실제 범퍼(역주: 핀볼에서 공을 튕겨내는 용도로 사용하는 장치)를 삽입해도 좋다. 일례로 *LEGO 브릭토피아(LEGO Bricktopia)*에는 공으로 파괴할 수 없는 핀볼 스타일의 범퍼 객체가 포함되어 있다.

숨바꼭질(Peek-a-boo)

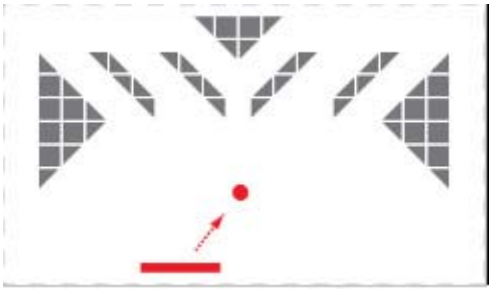


숨바꼭질 레벨에서는 다른 벽돌 더미의 뒤에 숨어있는 또 다른 벽돌 더미나 일시적으로(또는 계속적으로) 화면으로 들어왔다 나갔다 하는 벽돌 더미들이 포함되어 있다. **조언:** 게임이 다중 위치(multi-coordinate)나 논리 기반 움직임(logic-based motion)을 지원한다면 벽돌 더미가 화면으로 들어올 때의 각도를 변경해보라.

해괴한 형태(Oddball)

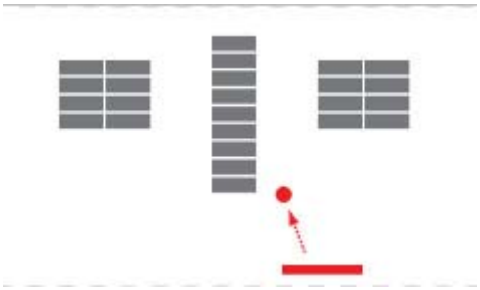
이 형태가 바로 “독특한” 레벨이다. 이 형태에는 특정 벽돌 깨기 스타일 게임의 설계/게임 시스템의 독특한 특성이나 기능을 표현할 수 있는 모든 레벨이 포함된다. 이 형태는 플레이어가 해당 게임에서 밖에 볼 수 없는 독특한 레벨을 선보여 일반적인 벽돌 깨기 스타일 게임 플레이에 익숙한 플레이어를 놀라게 하고 흥미를 유발시키는 것을 목적

으로 한다.



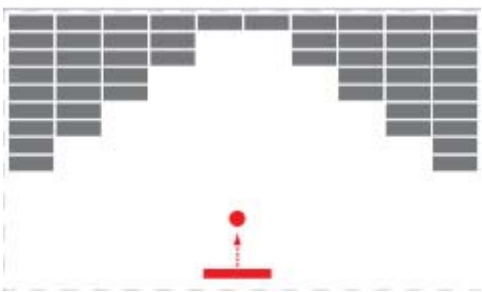
모서리(Angle)

실제로 경사진 벽돌이나 사각형의 벽돌을 계단 모양으로 배열한 사선 형태의 벽돌 더미이다. 이 형태는 사각형의 구조물로 들어찬 레벨들과 달리 신선한 느낌을 줄 수 있으며 2등분 사다리꼴 모양의 패들의 각진 모서리에 대한 명확한 사용법을 제시해 줄 수도 있는 레벨이다.



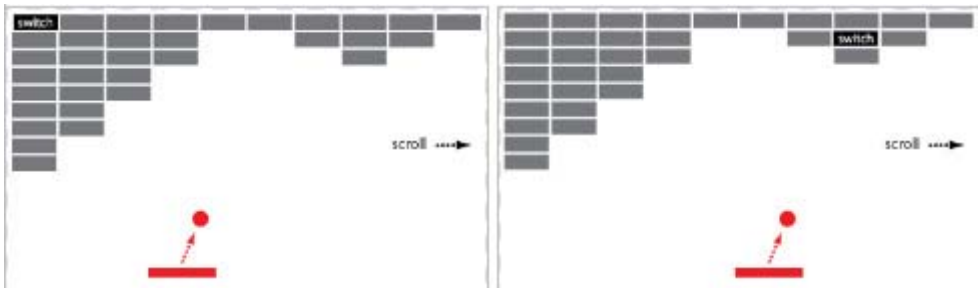
분할 화면(Split Screen)

분할 화면 레벨에서는 벽돌 더미가 명확하게 두 개의 구획으로 구분된다.(왼쪽/오른쪽, 위/아래, 안쪽/바깥쪽) 이런 벽돌 집단은 정적일 수도 있고 애니메이션될 수도 있으며 특정 시간 간격에 따라 일시적으로 분할될 수도 있다. 또한 각 구획을 구분하는 (파괴 가능하거나 파괴가 불가능한) 벽이 있을 수도 있다.



동굴(Cavern)

이것은 아래쪽 부분과 중앙의 대부분을 비워놓고 화면의 양 옆과 위쪽을 꽉 채우는 크기가 큰 오목한 벽돌 더미이다.



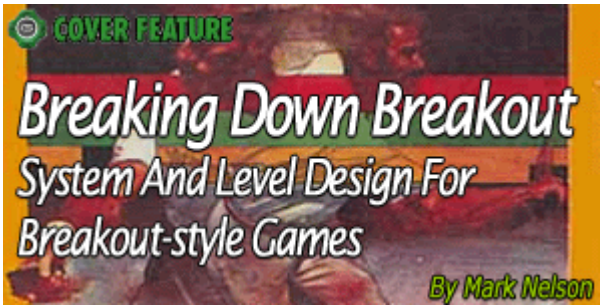
경기장(Arena)

화면이 다음 지역으로 스크롤되기 전까지 반드시 완수해야 하는 개별적인 목표(스위치를 맞추거나 공을 특정 공 함정에 빠뜨리거나)가 있는 화면 구획을 만들기 위해 벽돌 더미를 수직이나 수평으로 스크롤할 때 사용된다. 이런 형태의 레벨에서는 이전에 설명한 구성 요소를 혼용할 수 있지만 공이 화면의 경계에 걸려 질질 끌려 다닐 수 있고, 그로 인해 예상치 못했던 결과가 도출될 수 있으므로 화면을 스크롤하기 전에 화면에 있는 모든 벽돌이 확실히 파괴되도록 하라!

상징(Representational)

로봇, 피스톤, 난파된 공, 거북이 등의 시각적으로 플레이어가 이미 알고 있는 형태와 유사한 모양의 레벨이다. 이런 상징 레벨을 사용할 때에는 보이는 것만큼이나 플레이 하기도 재미있어야 한다는 것을 명심하라.

플레이 영역 움직임



위의 원형들이 고정적인 구조물에 효과적이기는 하지만 거기에 움직임을 추가하면 시간/타이밍이라는 완전히 새로운 요소를 추가하는 것이 된다. 움직이는 요소들 간의 복잡한 상호 작용은 플레이어에게 레벨이 발전한다는 느낌을 전해줄 수 있다. 또한 일정 개수의 정기적/무작위 이벤트를 첨가하면 플레이어가 레벨을 완수할 때 더 많은 전략이 요구되므로 반복 플레이 가치가 높아지게 된다.

얹혀있는 객체

얹혀있는 객체란 자기 자신이 실제로 움직이는 종류는 아니지만 다른 객체와 함께 이리저리 움직이도록 같은 객체 사슬에 연결된 객체를 말한다. 이 객체는 얹혀있는 객체에 작용하는 힘에 따라 딱딱하고 기계적인 움직임을 나타내거나 유연하고 자연스러운 움직임을 나타낼 수도 있다. 이것과 물리학이나 회전을 함께 사용하면 매우 흥미로운 레벨을 만들 수 있을 것이다.(*리코쳇 리차지드[Ricochet Recharged]*와 *브레이크 퀘스트[BreakQuest]* 참조)

회전

하나의 독특한 충돌 프로필을 가진 회전하는 객체를 추가하면 매우 재미있는 플레이 가능성을 가진 레벨을 구성할 수 있을 것이다.(선풍이날, 기어 등) 계층 구조와 함께 사용하면 회전하는 객체로 매우 다양한 종류의 움직이는 객체를 만들 수 있다.(참고: 회전 속도를 변화시키는 것을 잊지 말라!)

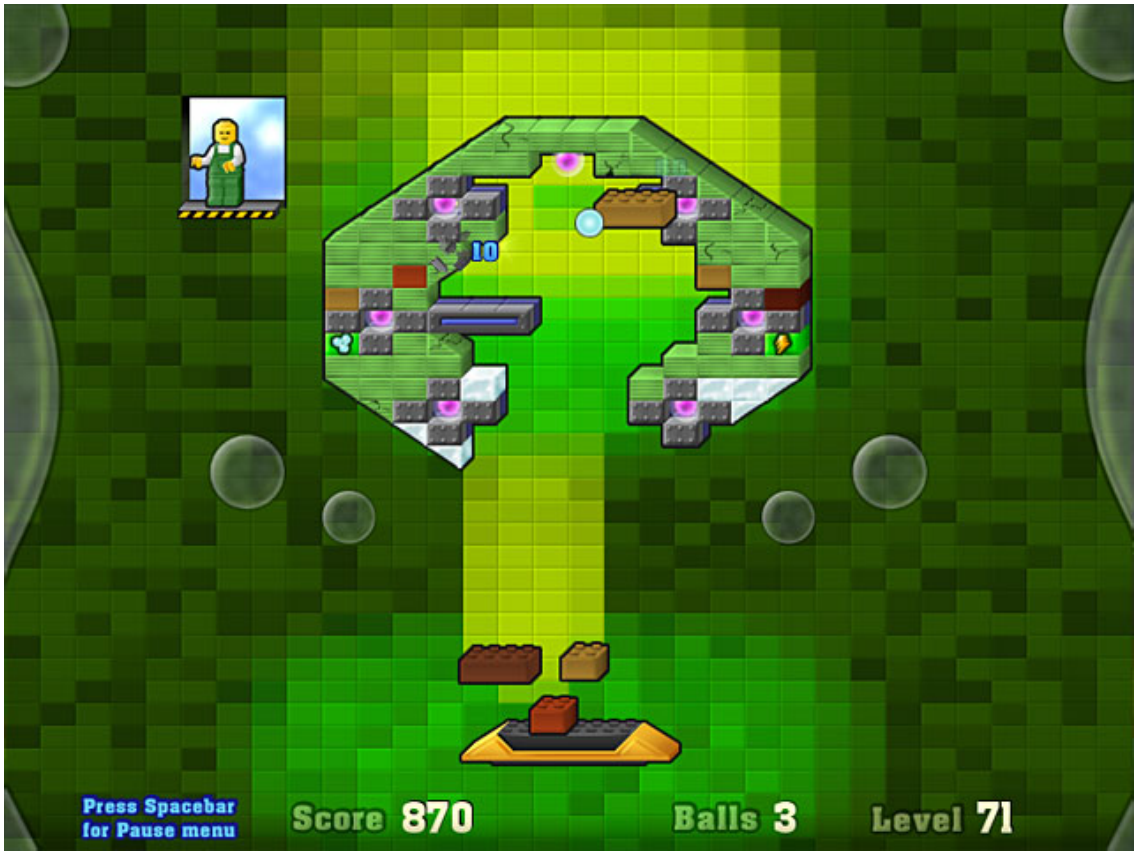
크기 변화

크기 변화는 플레이 영역의 깊이를 표현하거나 다중 충돌 목표물의 크기를 키우거나 줄일 때 유용하게 사용할 수 있다. 그물 형태의 레벨에서 다수의 크기 변화 객체를 사용한다고 생각해보라...

진동

진동은 벽돌이나 다른 레벨의 구성 요소가 간단하게 벡터(vector)를 따

라 앞뒤로 왔다 갔다 할 때 사용한다. 비록 이런 종류의 움직임은 항상 직선으로 움직이기는 하지만 다수의 진동 객체를 현명하게 사용하면 게임에 뭔가 더 복잡한 움직임을 가진 것처럼 “생각하게” 만들 수 있다.(*LEGO 브릭토피아[LEGO Bricktopia]* 참조) 제대로 된 진동 시스템에는 다음과 같은 속성들이 포함되어야 한다. 움직이기 전의 초기 지연 시간, 이동 속도, 이동 완료 후 잠시 멈춤, 부드러운 이동 시작과 종료, 완전히 멈추기 전까지 '몇 번' 움직임을 수행할 것인가



다중 좌표 움직임(Multi-Coordinate Motion)

레벨 객체가 여러 개의 좌표를 입력 받을 수 있도록 제작한다면 객체를 간단한 경로를 따라 이동시킬 수 있게 된다. 그리고 충분히 많은 양의 좌표, 잠시 멈춤 상태, 타이밍 조절, 트리거를 지원하기만 한다면 이 시스템을 활용해 극도로 복잡한 움직임도 생성할 수 있다.

경로 기반 움직임(Path-based Motion)

시스템이 좌표 목록을 사용한 움직임을 지원한다면 자연스럽게 다음

단계는 경로 기반 움직임이 되게 된다.(위에 설명한 다중 좌표 움직임보다 좀 더 간단한 시스템이므로) 위의 것과 다른 점은 레벨 디자이너가 (아마도 노드로 구성되는) 경로를 그릴 때 사용할 가시적인 도구를 제작하는 것이 좋다는 것이다. 가장 간단한 경로라면 직선 경로가 되겠지만 일단 시스템에서 경로 따라가기라는 개념을 사용하기 시작한다면 곡선을 구성하는 각 노드 사이를 보간(interpolate)해주어야 할 것이다.(예를 들어 B 스플라인과 같은 것을 사용해)

논리 기반 움직임(Path-based Motion)

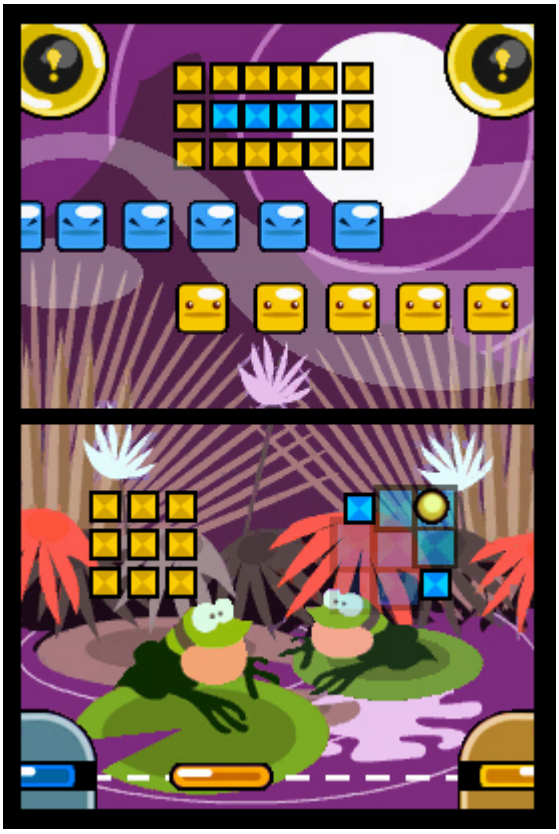
레벨 객체가 트리거 이벤트에 반응하게 만드는 것은 논리 기반 움직임으로 가는 좋은 방법이다. 하지만 플레이 영역의 객체가 지속적으로 변화하는 게임 플레이 환경에 반응해 이동하거나 움직임을 조절할 수 있도록 한다면 좀 더 많은 유연성을 가질 수 있게 된다. 그리고 이는 현재 패들이나 공의 위치와 같은 객체에 “피하기” 또는 “접근하기”와 같은 메소드(method)를 제공하는 것과 같은 간단한 방법으로 구현할 수 있다. 더 복잡한 것을 원한다면 다중 객체 무리 행동(multi-object swarming behaviors)이나 객체의 현재 상태를 인식이 용감함(내구력이 높을 경우 “접근”) 또는 겁먹은 듯한 행동을(내구력이 낮을 경우 “도망”) 시뮬레이션할 수도 있다.

플레이 영역 스크롤

이전 예제에서는 개별 객체에 대한 움직임 조작에 대해 논의를 했다. 플레이 영역을 스크롤한다는 것은 모든 객체를 한꺼번에 움직인다는 것이다. 이제 벽돌 깨기 스타일의 플레이 영역을 스크롤하는 것과 연관된 요소들을 살펴보도록 하자.

수직 스크롤 - 벽돌(또는 다른 레벨 객체)이 일정 시간 간격, 스위치와 공의 충돌, 게임 시스템 이벤트(맨 아랫줄의 벽돌들이 전부 사라졌다던 가와 같은) 등으로 인해 플레이어를 향해 아래로 내려온다. 수직 스크롤을 구현했을 경우에는 간혹 아직 사라지지 않은 벽돌이 패들에

닿거나 플레이 영역의 하단 부분에 벽돌이 심각하게 많아질 우려가 있다. 벽돌이 화면의 맨 아래 부분을 건드리는 것이 레벨의 종료 조건이 아닌 경우에 말이다. **조언:** 이런 경우를 해결하는 한 가지 방법으로는 스크롤을 시작하기 전에 맨 아랫줄의 벽돌이 모두 사라졌는지를 먼저 확인하는 것이 있다. 모든 벽돌을 레벨의 수평 단면을 나타내는 그룹으로 구분 짓고 스크롤을 시작하기 전에 알맞은 단면을 제거하라.



Arkedo사의 *너버스 브릭다운 (Nervous Brickdown)*은 "수직 스크롤 슈팅" 스타일의 게임 플레이를 가진 몇 개의 레벨을 가지고 있다. 이 게임에는 화면이 계속해서 스크롤되며 패들은 공을 튕겨내는 동시에 위에서 내려오는 벽돌도 피하도록 하고 있다.

수평 스크롤 – 벽돌(또는 다른 레벨 객체)이 일정 시간 간격, 스위치와 공의 충돌, 게임 시스템 이벤트(패들이 화면의 가장자리를 건드리거나) 등으로 인해 플레이 영역을 가로질러 왼쪽이나 오른쪽으로 이동한다. 수평 스크롤을 구현할 경에는 스크롤 과정에서 공이 벽돌 사이에 끼여서 움직이지 않는다거나 화면의 가장자리에 걸려서 질질 끌려갈 위험이 있다. **조언:** 모든 벽돌을 레벨의 수직 단면을 나타내는 그룹으로 구분 짓고 스크롤을 시작하기 전에 알맞은 단면을 제거하라. 또한 스크롤 도중에는 플레이 영역의 모든 움직임을 잠시 정지시켜서 스크롤이 끝날 때까지 공이 벽돌 사이에 끼지 않도록 하라. 마지막으로 레벨에 있는 벽돌 더미에 들어있는 중요한 파워 업이 있다면 스크롤되어 화면에서 사라지기 전에 플레이어에게 획득 기회를 주는 것을

고려해보도록 하라.

스크롤 트리거 - 위에서 말한 대로 레벨에서 몇 퍼센트의 벽돌이 제거되었는가, 스위치가 공과 충돌했는가, 일정한 시간이 지났는가와 같은 다양한 것들이 화면 스크롤을 유발할 수 있다. **조언:** 스크롤을 시작할 때에는 일관되고 명확한 규칙을 적용하라. 안 그러면 플레이어가 어째서 플레이 영역이 계속해서 변경되는지 이유를 알지 못해 혼란을 겪을 수도 있다.

스크롤 속도 - 벽돌 깨기 스타일 게임에서는 플레이어에게 레벨의 구성을 살펴볼 수 있는 시간을 주거나 스크롤된 화면에서 특수한 아이템을 얻을 수 있으리라는 기대감을 높이기 위해 상당히 느긋한 스크롤 속도를 사용하는 편이다. 스크롤을 사용하는 벽돌 깨기 스타일의 게임을 좀 더 자세히 살펴보고 싶다면 *펑키볼 어드벤처(Funkiball Adventure), LEGO 브릭토피아(LEGO Bricktopia), 너버스 브릭다운(Nervous Brickdown)*을 참조하라.

물리학 기반 레벨 설계

물리학 기반의 레벨은 벽돌 깨기 장르에 새로운 활력을 불어넣어 주었다. 그리고 이런 종류의 레벨은 Nurium Games사의 *브레이크 퀘스트(BreakQuest)*로 인해 유명세를 타게 되었다. 이런 레벨에서 벽돌(또는 다른 객체)은 공에 맞더라도 사라지지 않고 다른 객체를 건드려 복잡한 물리학 기반의 상호작용을 일으키게 된다. 그러므로 객체를 자주 맞춰서 간신히 통제할 수 있는 혼란을 유발하는 것이 레벨의 주요 초점이 된다. 그로 인해 브레이크 퀘스트의 레벨은 매우 역동적으로 보인다. 화면 내의 객체들 간에 거대한 상호작용이 일어나는 덕분에 레벨이 더 응집된 것처럼 느껴지며 간혹 꽤 우습게 보이기까지 한다. 경쾌하다, 유쾌하다, 엉클어지다. 이런 단어들을 벽돌 깨기 스타일

의 게임 플레이와 연관 지을 수 있으리라고 생각한 사람은 아마 거의 없을 것이다.

다음은 브레이크 퀘스트의 개발자인 Fèlix Casablancas와의 짧은 인터뷰이다.

어쩌다가 물리학 기반의 벽돌 깨기 게임을 만들 생각을 하셨습니까?

Fèlix Casablancas: 몇 가지의 게임에 대한 생각을 가지고 있었지만 최종적으로 '물리학을 사용하는 벽돌 깨기'를 만들어보자라고 생각했습니다. 뭔가 새롭지만 단 한 명이 제작할 수 있을만한 것을 만들고 싶었거든요. 현재 수없이 많은 벽돌 깨기 스타일의 게임이 존재하니 그런 게임이 하나 더 출시된다고 해도 그리 좋은 반응을 얻을 수는 없겠죠. 하지만 고전적인 게임 메카닉에 물리학을 추가함으로써 레벨 설계에 완전히 새로운 지평이 열렸습니다. 또 저는 공이 매우 사실적으로 움직이도록 만들고 싶었습니다.

어떤 점 때문에 전통적인 벽돌 기반의 레벨에 게임의 초점을 맞추지 않겠다고 결심하셨습니까?

FC: 앞서 말씀 드린 것과 같이 이미 수많은 벽돌 깨기 게임이 있었고 벽돌 기반의 레벨은 완전히 새로운 무엇인가를 만들기에는 유연성이 부족했기 때문이었습니다. 사실, 물리학을 사용하자라는 생각이 없었다

면 벽돌 깨기를 만들지는 않았을 겁니다. 벽돌 기반의 레벨에서 (예를 들어) 용수철로 연결된 벽돌이 움직인다는 것은 생각할 수조차 없는 일이죠. 하지만 물리 엔진을 사용하면 이런 것은 상대적으로 쉬운 일일 뿐입니다.

레벨에 배치할 특정 객체/도형이나 강조를 위해 사용한 특정 움직임을 선택하는데 있어서 어떤 특별한 이유가 있었나요?

FC: 모든 레벨에는 저마다의 이유가 있습니다. 사용한 도형에는 고전적인 사각형 벽돌부터 원형, 삼각형, 별 등과 같이 상당히 많은 종류의 도형을 사용했습니다. 일례로 공에 맞으면 불규칙한 모양의 작은 객체로 쪼개지는 벽돌이 있는 운석 레벨에서 벽돌을 보시면 외계인, 작은 괴물과 비슷하게 생겼죠. 움직임에 대해서는... 역시나 저는 다양함을 추구했습니다. 어떤 것은 선형적인 예상 가능한 방식으로 움직이고 어떤 것은 서로 충돌하거나 물리 엔진 또는 코딩된 알고리즘에 따라 조절되는 용수철이나 다른 통제 장치와 상호작용을 하죠. 중력의 영향을 받거나 받지 않는 경우도 있고 마찰력이 서로 다를 때도 있고, 회전 우력과 질량도 다르고, 공에 맞았을 때 그냥 회전만 하는 것도 있고 심지어는 다른 장소로 사라져버리기도 하죠.

레벨을 만들 때 객체 간의 상호작용을 어떻게 예측하셨습니까? 혹시 레벨 설계 과정이 그저 시행착오를 거치며 제작하는 작업이었나요?

FC: 어떤 레벨은 상호작용이 상당히 예측 가능하게 일어나지만 어떤 레벨들은 원래 설계에서 상당히 많은 부분을 수정해야 했습니다. 시행착오를 거치면서 수정하는 것도 중요하기는 하지만 항상 실험을 해보기 전에 기본적인 개념을 잡고 시작하는 것이 좋습니다.

레벨 설계를 통해 어떤 시각적인 효과를 얻으려고 하셨나요?

FC: 화려한 색깔과 약간의 픽셀 아트(Pixel art)에 약간의 현대적 디자인과 특수한 입자 효과를 사용한 고전적인 외형을 표현하려고 했습니다. 일종의 개선된 고전적 시각 효과라고 할 수 있겠죠.

게임 레벨이 상당부분 다른 것이 많은데 플레이어가 레벨의 목표를 이해하도록 하기 위해 어떤 조치를 취하셨습니까?

FC: 대부분의 레벨은 일관되게 플레이어의 눈에 보이는 객체가 부서지는 것인지 확인할 수 있도록 공을 튕겨내는 스타일입니다. 따로 설명이 필요한 몇몇 레벨에서는 작은 "조언" 창이 표시되어 해당 레벨의 목표를 명확하게 알려주도록 했습니다. 제 생각에는 플레이어가 처음에는 좀 충격을 받을 수도 있겠지만 게임을 진행하며 각각의 레벨에서 어떤 일을 해야 할지에 대한 감을 잡을 수 있으리라 생각합니다.

복잡한 물리학을 사용하는 벽돌 깨기 스타일의 게임을 제작하려는 개발자들에게 가장 중요한 조언은 어떤 것이라고 생각하십니까?

FC: 물리 엔진을 사용하기 전에(저와 같이) 사용할 엔진에 대해 조사를 먼저 해보기를 권합니다. 물리 시뮬레이터의 구현에는 몇 가지 방법이 있고 각각의 알고리즘은 서로 다른 상황에서 문제점을 안고 있죠. 그러니 게임 메카닉에 따라 어떤 방법이 좋을지를 먼저 선택하십시오. 또한 수학은 특정 상황에서 불안정해지기도 하므로 왜 엔진이 오작동을 일으키고 그에 적당한 차선책을 찾기 위해 엔진의 내부에서 자료를 어떤 방식으로 처리하는가를 이해해야 합니다.

_브레이크 케스트(BreakQuest)_와 Nurium Games의 차후 계획은 어떠십니까?

FC: _브레이크 케스트_의 차후 계획: 모바일(Mobile) 버전의 개발이 진행 중입니다. 몇 달쯤 후에는 제작이 완료되기를 희망하고 있습니다. 또 회사에서 Nintendo DS 버전을 개발하려고 하고 있기는 하지만 현 시점에서는 그저 기획단계일 뿐입니다. 또 대형 콘솔(역주: Xbox 360, PS3 등)로의 이식에 대한 이야기가 오가고는 있지만 현재는 논의일 뿐이죠.

Nurium Games의 차후 계획: 저는 이번에는 물리학을 사용하지 않는 퍼즐 게임을 개발하고 있습니다. 메카닉이 꽤 독창적인 편이니 성공을 거두기를 희망하고 있습니다. 하지만 현재 너무 많은 프로젝트에 관여 중이라 언제 완성할지는 모르겠군요!

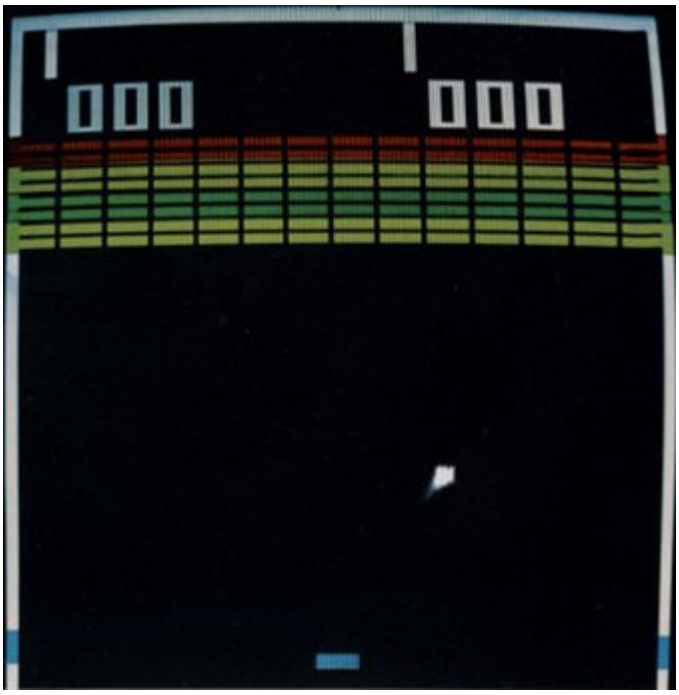
레벨 품질 점검표

레벨을 만들 때는 레벨의 품질에 악영향을 끼치는 결정을 하거나, 너무 틀에 박힌 레벨을 만들거나, 게임의 콘텐츠를 과도하게 사용하기 십상이다. 다음은 각 레벨을 차별화하면서도 레벨의 품질 향상에 도움을 줄 수 있는 점검표이다.

이 질문들이 모든 종류의 레벨에 적용 가능한 것은 아니지만 레벨 설계에서 자신이 놓친 품질 향상의 기회를 찾아낼 때에 유용하게 사용할 수 있을 것이다. 새 레벨을 설계할 때나 여전히 좀 더 다듬어야겠다고 생각한 레벨을 살펴볼 때에는 잠깐 시간을 들여 다음의 질문들을 살펴보기 바란다.

초기 인상 – 플레이어가 레벨을 불러와서 레벨을 처음 봤을 때 재미있겠다는 느낌을 받을 것인가? 그런 느낌이 처음에만 국한되는가, 아니면 시간이 지남에 따라 점점 강해지는가? 그 느낌이 시각적 설계에 기반한 것인가, 아니면 레벨을 만들 때 사용한 객체들에 의한 플레이 가능성에 의한 것인가?

초기 선택 – 단순히 공을 벽돌 더미로 날려보내는 것 외에 플레이어가 공을 가지고 처음에 선택할 수 있는 사항이 존재하는가? 공을 날려보낼 수 있는 재미있는 궤도가 존재하는가? 공으로 맞출 스위치가 있는가? 처음부터 연쇄 반응을 유발시킬 수 있는가? 레벨에서 플레이어가 선택할 수 있는 선택 사항에는 몇 가지가 있는가?



명확한 목표 – 레벨에서 단순히 모든 벽돌이 사라지면 종료되는가, “레벨 종료” 목표가 명확한가? 일련의 레벨 전체를 플레이 하며 달성해야 하는 하나의 큰 목표가 있는가? 그렇다면 그것이 명확하게 소개되었는가? 모든 게임 객체(벽돌, 파워 업, 파워 다운 등)는 그 사용법과 파괴력 등이 명확하게 소개되었는가? 중요한 레벨 목표들이 명확하게 정의되어 있는가?

피드백 – 플레이어가 자신이 플레이를 잘 했을 때 명확하게 그렇다라고 답할 수 있는가?(레벨을 끝마치기 위해 목표 객체의 감소에 의존하지 않는 레벨에서 중요함. 일례로 레벨을 끝마치기 위해 각 객체를 3번씩 맞추거나 여러 번 맞춰서 빨간색으로 만들어야 하는 다중 충돌 객체가 있는 레벨과 같은 경우) 플레이어가 공이 항상 논리적이거나 (최종적으로) 예측 가능한 방식으로 움직였다고 생각하는가?

공의 되돌아옴 – 공이 영구적으로 튕겨 다녀 플레이어가 지루해하지 않도록 상대적인 시간이 흐르면 플레이어에게 다시 되돌아오는가?(참고: 공이 플레이어가 닿지 않는 곳에서 계속해서 튕겨 다닌다면 그 동안 플레이어가 사용할 수 있는 두 번째 공을 풀어주도록 하라.)

보상 – 플레이어가 사용할 수 있는 다양한 보상과 파워 업이 충분히 존재하는가? 플레이어가 레벨과 좀 더 많은 상호작용을 하도록 미끼 역할을 하기 위해 좋은 보상을 벽돌 더미의 깊숙한 곳에 배치해놓았는

가? 플레이어가 여러 레벨에 걸쳐 효과를 발휘할 수 있는 아이템이나 파워 업을 획득할 수 있는가?

플레이 가능성 – 공, 패들, 목표 객체(벽돌, 파워 업 등) 간의 상호작용과 빈 공간이 플레이의 다양성과 게임의 만족도를 높일 수 있을 만큼 충분히 존재하는가? 레벨을 한 번 이상 플레이 했을 때 이전과 플레이 시의 체감 내용이 다른가? 무작위이거나 스크립트된 이벤트가 존재하는가? 레벨에 있는 객체들 간에 복잡한 알고리즘적 상호작용이 존재하는가?

대량 파괴/역동성 – 대량의 벽돌을 한 번에 파괴할 수 있는 기술이나 특수한 객체가 존재하는가? 레벨이 정적인가? 레벨에 있는 객체들 중에 공과 패들을 제외하고는 얼마나 많은 객체가 움직이는가? 공이 10초간 화면에서 사라져도 여전히 레벨이 재미있어 보이는가?

레벨의 길이 – 플레이어가 레벨을 완수하는데 얼마나 걸리는지를 시험해보았는가? 얼마나 오래 플레이 해야 너무 길다고 느껴지는지 알고 있는가? 얼마나 짧은 경우에 레벨이 대수롭지 않다고 느껴지는가?

레벨의 주제

고전적인 벽돌 깨기 레벨 구조도 여전히 플레이어에게 재미를 선사할 수 있기는 하지만 전체적인 주제를 레벨(또는 레벨 그룹)에 적용하면 더욱 재미있게 만들 수 있다.(그런 주제들 중 몇 가지는 위의 “레벨 품질 점검표”에서 이미 제시되었다.)

다중 경로 – 플레이어에게 중요한 목표 또는 레벨을 끝마칠 수 있는 목표에 대한 여러 가지 방법을 명확하게 제시해 몇 가지의 분명한 초

기 선택권을 부여하라. 예를 들어 다음과 같은 방법을 이용해 달성할 수 있는 목표를 말이다. (a) 몇 가지의 트릭 샷(Trick shot)을 사용, (b) 빠른 반사신경과 억지력을 활용한 전술 사용

파괴 – 플레이어에게 많은 개수의 벽돌, 공, 파워 업을 주어라. 이런 종류의 레벨에서는 공의 위치를 정확하게 판단하는 것보다는 통제할 수 있는 혼란을 최대한 오래 지속시키는 것이 더 중요하다.

사격 연습 – 이런 종류의 레벨은 상대적으로 공간이 비어있기 때문에 공을 다루는 기술이 어느 정도 있어야 끝마칠 수 있다. 공의 궤도를 통제하는 것과 관련한 레벨이기 때문에 시스템에서 얼마나 공을 정확하게 배치할 수 있는가가 바로 레벨의 난이도와 직결되므로 레벨의 난이도가 너무 상승하면 난이도를 조절할 시스템도 갖추고 있어야 한다.(시간 제한 파워 업 제공, 자동 조준 등) 이런 레벨은 너무 자주 등장하면 게임이 지루해질 수 있으므로 주의 깊게 활용해야 한다.

회피 – 이런 종류의 레벨은 패들로 낙하하는(공과 객체가 충돌하면 무작위, 또는 시간 간격을 두고 낙하하는) 파워 다운을 피하는 것과 관련된 레벨이다. 이런 레벨을 '사격 연습' 레벨의 바로 다음에 배치하면 게임의 국면을 전환하는데 좋다.

퍼즐 – 레벨을 스크롤하려면 어떻게 해야 할까? 어떻게 해야 보상을 얻을 수 있을까? 스위치들을 맞출 올바른 순서는 무엇일까? 이런 것들이 바로 플레이어들이 퍼즐 레벨에서 답을 찾아야 하는 질문들이다. 하지만 플레이어들이 이런 레벨에서 매우 일반적인 "대체 여기서 뭘 어쩌란 말인가?"라는 질문을 하면 어떻게 할 것인가? 그런 경우에는 문제/해법을 더 명확하게 만들 필요가 있다.

보스 – 보스 레벨은 일련의 레벨의 난이도를 가장 최종적으로 표현하는 레벨이다. 플레이어는 절박한 도전에 맞서 싸우던가, 이전 레벨에서 제시된 도전에 맞서 싸워야 한다. 또한 보스 레벨을 플레이어의 기술을 시험하거나 이전 레벨에서 얻은 파워 업을 소진시키는 방법으로 생각할 수도 있다. "보스"라는 것이 꼭 플레이어가 파괴해야 할 움직이는 거대한 게임 객체일 필요는 없다는 것을 참고하라. 벽돌 깨기의 관점에서 보면 보스 레벨은 공으로 맞출 때마다 다른 장소로 순간이동을 해버리는 하나의 벽돌이거나, 눈이 내려 플레이어가 패들을 한 자리에 너무 오래 머무르게 하면 얼어붙어 버리는 레벨일 수도 있다.

게임 토큰 우선순위

벽돌 깨기 스타일 게임에는 주어진 시간 동안 플레이어가 주의를 기울이게 되는 "판에 박힌" 요소들이 존재한다. 다음은 레벨에 존재하는 게임 토큰들의 우선순위를 정하는 하나의 지침이다. 이 목록에는 수동으로 배치된 요소와 프로그램적으로 생성되는 요소들이 모두 포함되어 있다.(참고: 숫자가 낮을수록 우선순위가 더 높다.)

우선순위 1 – 패들과 공, 공의 소리도 포함
우선순위 2 – 레벨의 목표 또는 전체적인 게임 토큰
우선순위 3 – 위협과 파워 다운
우선순위 4 – 파워 업과 특수 벽돌
우선순위 5 – 점수 획득 알림, "500점"과 같은 음성
우선순위 6 – 문자 알림 메시지, "최고입니다!"와 같은 메시지
우선순위 7 – 통상적인 벽돌

확실히 움직이고, 깜박이거나 빛이 나는 객체는 정적인 요소들보다 더 많이 플레이어의 주의를 끈다. 그리고 강한 음향 효과가 가미된 객체도 마찬가지이다. 낮은 우선순위를 가진 게임 토큰이 높은 우선순위를 가진 게임 토큰과 겹치는 특성을 가지지 않도록(또는 비슷하게 보이지 않도록) 특별히 주의하도록 하라.

레벨 편집 도구
게임 엔진

고전적인 *벽돌 깨기*는 매우 낮은 시스템을 요구했었다. 그러므로 고전적인 벽돌 깨기의 복제품을 제작하고 싶다면 어떤 것으로 개발해도 상관없다. 그 어떤 것을 사용해도 말이다.(http://excelgames.org/breakout.asp)

경쟁력이 있는 현대적인 게임을 제작하려 한다면 선택해서 사용할 수 있는 오픈 소스(Open source)와 상용 게임 엔진이 존재하니 C, C++, C#, Java 등과 자기 자신의 기술을 사용해 완전히 새로 만들 필요는 없다.(http://en.wikipedia.org/wiki/List_of_game_engines) 일반적인 상호작용 미디어 도구들 중에서 선택하라면 물론 Adobe Director와 Adobe Flash를 제일 처음 꼽을 수 있을 것이다. Adobe Flash의 경우에는 ActionScript 2.0+를 사용해 많은 일을 해야하며 게임 시스템에서

대량의 움직이는 객체와 효과를 동시에 사용할 경우에 성능 문제가 발생할 수 있다는 점을 주의하라.

레벨 편집기

대부분의 게임 엔진은 어떤 형태로든 레벨 생성 도구를 포함해서 출시되기 마련이다. 하지만 솔직하게 말해서 간단한 *벽돌 깨기* 스타일의 게임용 레벨을 만들 때는 그냥 아무 문서 편집기나 사용해도 제작할 수 있다. 그저 메모장에서 쉼표로 구분된 고정적인 레벨 원형들의 데이터 배열만 기술해도 되기 때문이다. 하지만 이런 종류의 편집기는 그리 상호작용적인 환경을 제공해주지 않으며 게임에 복잡한 기능(설정 가능한 속성, 움직임 좌표, 이름이 할당된 트리거, 이름이 할당된 그룹 등)이 추가되더라도 문법적이나 논리적인 오류를 표시해주지도 않는다.

Reflexive Entertainment사의 리코쳇 리차지드(Ricochet Recharged)는 매우 다양한 형태의 레벨을 가지고 있는 것으로 유명하다. *리코쳇*의

레벨 중에는 고급 편집기에서나 제공되는 기능들인 배경에 애니메이션이 표시되고 레벨에 있는 객체가 아주 많이 움직이는 레벨이 포함되어 있다. 이제 *리코쳇*의 레벨 편집기를 살펴보고 이 게임의 기획자이자 리드 프로그래머인 James Smith와 개발 과정에 대한 논의를 해보도록 하자.

서로 다른 지역에서 살고 있는 디자인 팀을 위한 레벨 편집기를 제작할 때 가장 중요하게 생각한 점은 무엇인가요? 플레이어가 사용할 수 있는 레벨 편집기도 원래부터 고려했던 사항이었습니까?

James Smith: 저희는 처음에 팀 내부의 레벨 디자이너가 사용할 목적으로 레벨 편집기를 제작했었습니다. 다양한 종류의 레벨을 만들 수 있도록 최대한의 유연성을 가지도록 제작했죠. 매우 쉽게 배울 수 있도록 설계된 것은 아니었습니다. 저희의 숙련된 레벨 디자이너가 효과적으로 사용할 수 있도록 설계된 것이니 당연한 것이겠죠. 새로운 사용자가 기능을 익히려면 시간이 좀 걸리겠지만 일단 사용 방법을 익히고 나면 단순화된 컨셉이나 사용자 인터페이스로 인해 불편을 겪는 일은 없습니다.

많은 사람들이 파일을 편집해야 하므로 파일 편집에서 병목 현상을 일으키지 않고 다양한 개발환경을 소화할 수 있도록 제작하는 것이 관건이었습니다. 그로 인해 게임의 모든 레벨에 대한 목록을 담고 있는 마스터 파일(master file)이 존재하지 않게 되었죠. 각각의 레벨 디자이너는 자신들의 레벨 집합을 서로 독립적으로 작업할 수 있었습니다. 또한 레벨 집합들의 병합과 분할(merge and split)을 쉽게 할 수 있도록 해주는 도구도 제작해야 했습니다. 이 프로그램은 그저 레벨 집합을 복사해서 다른 집합에 붙여 넣기만 하면 기본적으로는 작업이 끝나게 되죠.

리코쳇(Ricochet) **시리즈는 움직이는 객체가 다수 등장하는 것으로 유**

명합니다. 레벨 편집기에서 복잡한 움직임을 통제하는 것에 대한 생각이나 개선 방향을 설명해주시겠습니까?

JS: 리코쳇의 편집기에서는 타임라인(timeline)도 없고 키 프레임(key frame)도 없습니다. 움직임을 조절하는 부분은 간단한 부분들을 개선시킨 상당히 독특한 시스템에 기반해서 생성되죠.

게임의 편집기와 스크립트 엔진은 저희가 과거에 RPG와 RTS 게임용으로 제작했던 유사한 시스템의 영향을 크게 받았습니다. 이 시스템의 주요 개념은 작은 기본 움직임들을 조합해서 복잡한 움직임을 만들어내는 것입니다. 스크립트 명령어(플러그 인)는 모두 다양한 목적으로 사용할 수 있는 매우 간단한 행동으로 구성되어 있습니다. 특정 지점으로 이동, 특정 지점 주변을 회전하기, 다른 객체의 움직임을 흉내내기, 객체가 파괴될 때까지 대기, 메시지가 들어올 때까지 대기, 메시지를 다른 객체에 전달, 자폭, 경로를 따라 이동과 같은 것들로 스크립트 명령어가 구성되어 있습니다.

이런 명령어들은 동시에 수행되거나, 순차적으로 수행되거나, 무작위적인 순서로 수행되는 등의 방법으로 여러 개를 그룹으로 묶어 실행할 수 있습니다. 모든 행동 조합과 그룹들은 그 안에 다른 행동 조합이나 그룹을 포함할 수도 있죠. 구분을 분석하거나 따로 입력해야 하는 스크립트 언어는 사용하지 않습니다. 모든 것을 객체 조합을 통해 제작하죠. 플러그 인을 슬롯에 끼워 넣는 방식이고 이런 플러그 인들의 대다수는 내부에 다른 것을 끼워 넣을 수 있는 슬롯을 가지고 있습니다. 하나의 벽돌은 트리와 유사한 구조에 10개의 플러그 인을 연결할 수 있습니다.(루트에 하나나 두 개의 플러그 인을 연결하고 거기에 다른 플러그 인을 연결하는 방식)

복잡한 애니메이션은 보통 위치 표시기(position markers)라 불리는 숨겨진 벽돌을 포함한 몇 개의 벽돌이 서로 연동되도록 하여 제작합니다.

숨겨진 위치 표시기는 그 위치 주변을 돌아다닐 수 있도록 몇 개의 플러그 인을 연결해놓을 수 있죠. 그 다음에는 다른 벽돌에 위치 표시기의 움직임을 따라 하도록 만들거나 위치 표시기를 참조점으로 사용하는 플러그 인을 연결해 움직이는 참조점에 상대적인 움직임을 취하도록 합니다.

레벨 디자이너가 새로운 고수준 행동을 요청하면 저희는 항상 그 요청된 행동을 더 간단한 것들로 분해해서 구현하거나 다른 행동 조합을 재활용해서 그런 요청된 행동을 구현했습니다.

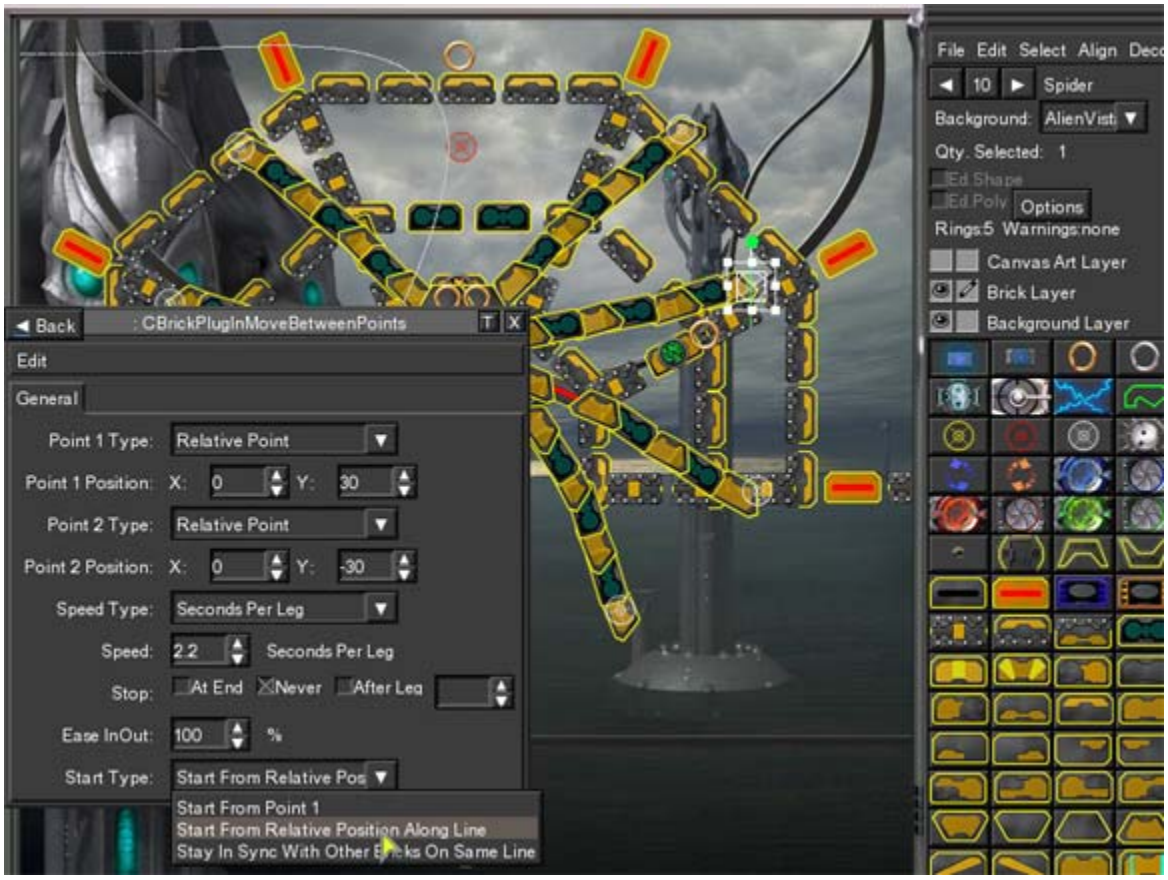
어째서 애니메이션이 표시되는 배경을 포함시켰고 이 기능을 레벨 디자이너가 어떻게 사용하도록 하셨나요?

JS: 각각의 배경은 (실제이건 가상이건) 특정 장소와 유사하게 보이도록 설계되었습니다. 그렇게 배경이 더 완전하고 신빙성이 있게 만들려면 애니메이션을 구현하는 것이 필수적인 요소 중에 하나였습니다. 헤엄치는 물고기, 떠다니는 플랑크톤, 떠오르는 물거품 같은 것도 없이 어떻게 바다를 표현하겠습니까?

각각의 리코쳇 게임에는 레벨 디자이너가 사용할 수 있는 서로 다른 배경 집합이 포함되어 있으며 각각의 배경 집합에는 고유의 사운드 트랙, 배경 그림, 배경 애니메이션, 벽돌 집합(어떤 사람은 타일 세트로도 부르더군요.)을 가지고 있습니다. 레벨 디자이너는 배경 집합을 선택한 후에 해당 배경 집합에서 사용할 수 있는 벽돌들을 사용해 레벨을 제작하면 됩니다. 그리고 수중 배경 집합에서는 헤엄치는 물고기와 같이 해당 배경 집합에서 볼 수 있는 애니메이션의 일부를 벽돌이 수행하도록 할 수도 있지만 레벨 디자이너들이 배경 집합의 일부로 포함되어 있는 배경 애니메이션을 수정할 수는 없습니다. 애니메이션을 설정하는 것은 레벨 디자이너의 몫이 아닌 아티스트(Artists)의 몫이며 이런 애니메이션은 항상 특정 배경 집합의 일부로 간주됩니다.

레벨 편집기와 게임의 설계가 동시에 개선되었습니까? 아니면 미리 작성해놓은 기능 목록에 따라 제작된 것입니까?

JS: 모든 것은 반복해서 개선되었습니다. 대개는 전체적인 설계 목표를 정해놓고 제작을 진행했지만 그 과정에서 예상치 못했던 많은 기능이 추가되었습니다. 그 중 많은 경우에서 새로운 기능이 필수적으로 게임에 추가되어야 했고, 결과적으로 편집기에도 적용되었습니다. 하지만 편집기에 추가된 새 기능으로 인해 새로운 디자인 아이디어가 떠오르기도 했습니다.



레벨 편집기 제작에서의 개발/교정 과정에서 중요했던 순간들을 좀 말씀해주시겠습니까?

JS: 저희의 편집기는 항상 매우 강력한 객체 속성 시스템을 기반으로 해왔습니다. 이 시스템은 편집기에서 동적으로 생성된 속성을 실시간

으로 편집하고 파일에 저장할 수 있도록 객체/클래스의 모든 멤버 변수를 열거할 수 있는 직접 제작한 “반사” 메커니즘을 사용할 수 있었습니다. 이 시스템의 사용으로 인해 편집기의 제작은 매우 간단했습니다.

두 번째로 *리코쳇* 게임을 제작하며 저희는 앞서 언급한 조합 기반의 스크립트 시스템 개발에서 이 속성 시스템의 덕을 톡톡히 보았습니다. 첫 번째 *리코쳇* 게임에서는 단지 매우 간단한 움직임만을 지원했고 트리거도 지원하지 않았었습니다. 두 번째 리코쳇 게임에 플러그 인 연결 구조를 추가하자 새로운 레벨 종류를 제작할 수 있는 가능성이 기하급수적으로 증가했습니다. 레벨에서 어떤 이벤트가 다른 이벤트를 발생시킬 수 있게 되자 레벨이 별안간 훨씬 더 역동적으로 변화했습니다.

그 다음으로 중요한 순간으로는 두 번째 리코쳇 게임이 출시된 지 몇 달 후부터 플레이어들이 레벨 편집기로 매우 뜻밖의 레벨들을 만들기 시작할 때를 꼽을 수 있습니다. 플레이어들은 게임에서 사용된 벽돌들이 더 재미있는 방식으로 작동하도록 몇몇 기능들을 저희가 의도하지 않았던 방식으로 사용했습니다. 그로 인해 갑자기 파워 업 그림, 폭발, 게임 오버 메시지, UI 구성 요소 등의 게임의 다른 부분의 이미지가 잘못 출력된 벽돌로 뒤 덮인 벽돌들이 화면을 가득 메우기 시작했죠. 처음에는 그런 것들을 보고 감탄했었지만 그 다음에는 실망을 했었고, 그 이후에는 레벨 디자이너들이 자신들의 레벨에 더 많은 색깔과 치장을 할 수 있는 방법을 필사적으로 찾고 있었다는 사실을 깨닫게 되었습니다.

그로 인해 저희는 *리코쳇 인피니티(Ricochet Infinity)에서* 레벨 디자이너들이 자신들의 레벨에 더 많은 색깔과 치장을 할 수 있는 새로운 기능을 추가했습니다. 이제 레벨 디자이너들은 자신들이 원하는 대로 화려하고, 우아하거나 하는 등의 객체를 더 자유롭게 만들 수 있게 되었

습니다.

레벨 편집기의 다양한 버전으로 인해 레벨 파일들간의 호환성이 문제가 된 적은 없었습니까?

JS: 편집기의 변경으로 인한 레벨의 비호환 문제는 발생한 적이 없었습니다. 리코쳇의 레벨은 XML과 유사하지만 더 간단한 구문 구조를 지닌 계층적 텍스트 파일을 기반으로 한 매우 유연한 파일 형식으로 저장됩니다. 이 형식에서는 객체의 각 속성을 저장하기 위해 "X Position = 40", "Brick Style = Underwater/Blue Stripe Power-Up"과 같은 이름과 값을 한 쌍으로 저장합니다. 이런 속성들은 각각 다른 속성에 포함되거나 하는 식으로 계층적으로 구성됩니다. 또 벽돌들은 각각 플러그 인(스크립트)을 포함하고, 레벨은 벽돌들을 포함하며, 레벨 집합은 레벨들을 포함하게 되는 것이죠.

이런 계층적 텍스트 기반 속성 시스템으로 인해 저희는 이전의 레벨 구조를 망가뜨리지 않으면서도 벽돌이나 레벨에 새로운 속성을 추가할 수 있었습니다. 간혹 가다가 오래된 속성 형식을 변환하기 위해 특수한 코드를 추가하기도 했지만 대부분의 경우에는 레벨 데이터에 이진 구조(Binary structure)를 사용하지 않았기 때문에 오래된 레벨도 최신 버전의 편집기에서 아무런 문제가 없었습니다.

나쁜 레벨들 추려내기

자신이 설계한 모든 레벨을 게임에 전부 적용할 수 있는 경우는 매우 드문 경우이다. 그리고 보통 자신이 제작한 레벨들 중에서 몇몇은 재미가 덜하거나 "조화롭지 못한" 경우가 태반일 것이다. 게임의 기능과 설계를 병행해서 개선하고 있는 경우라면 초기에 설계한 레벨이 개발 과정의 후반에 추가된

기능을 사용하지 않고 있을 가능성이 높으므로 이런 사태가 발생할 것이 거의 확실하다.

이럴 경우 한가지 해결책은 복잡한 게임 기능을 활용하지 않으므로 단순히 그런 레벨을 수정해서 새로운 기능을 “소개”하도록 한 다음에 게임의 초반부에 배치하는 것이다. 그래도 너무 간단해 보인다면 고정된 목표, 움직이는 객체, 명확한 스위치/트리거 등과 같은 새로운 중점 사항들을 소개해보도록 하라.

레벨의 사용을 포기하기 전에 한 번 잘 살펴보도록 하라. 간혹 초기에 제작한 레벨에서 재사용할 가치가 있는 부분이나 구조를 찾을 수 있을 때도 있다. 그리고 이런 작은 부분들을 조합해서 느낌이 약한 레벨의 설계를 보충하는데 사용할 수 있을 것이다. 그러므로 그런 레벨들 중에서 약간의 좋은 점을 발견한다면 통합용 레벨 파일에 차후 사용을 위해 복사한 후에 붙여놓아라.(레벨 편집기에서 복사/붙이기를 지원한다면...)

레벨의 일부분들을 모아서 완전히 새로운 레벨을 제작하기로 결정했다면 서로 공통되는 부분이 없는 구성 요소들 간의 조화를 맞춰야 한다는 것에 유의하라.

레벨 진행

완전한 레벨들의 모음을 제작했다면 다음으로 해야 할 일은 최상의 게임 플레이를 위해 그것들의 순서를 정리하는 것이다. 줄거리나 일관된 디자인 주제를 따라 레벨을 설계했다면 이미 암묵적인 레벨 순서가 정해져 있을 것이다. 레벨 각각이 서로 다른 게임 플레이를 가지도록 레벨을 설계했다면 적합한 순서를 찾기가 더 힘들 것이다. 다음은 레벨 순서를 정할 때의 몇 가지 조언이다. 이 중 몇몇은 전통적인 콘솔 시장에서는 말이 안될 지도 모르겠지만 일반적으로 다운로드할 수 있는

인터넷과 같은 공간에서는 상당히 실용적일 것이다.

크기 순서

작은 레벨이 처음에 나오고 더 큰 레벨이 나중에 나온다.(일례로 스크롤 가능한 플레이 영역을 가진 레벨이 등장하기 전에 화면 하나짜리 레벨이 등장하는 것과 같이) 캐주얼 플레이어들에게 이것은 게임이 더 시각적으로 이해하기 쉽고 접근하기 쉽도록 해준다. 단점으로는 자신이 제작한 크고, 제작하기 힘든 레벨들을 많은 수의 플레이어가 보지도 못할 가능성이 있다는 것이다!

파워 업을 등장시킬 때에는 초반과 후반 레벨의 크기 차이를 고려하고 해당 레벨의 크기에 어떤 파워 업이 가장 효과적일까를 고려하라.

플레이 시간 순서

이 순서는 더 큰 레벨은 보통 끝마치는데 더 오랜 시간이 소요되므로 크기 순서와 유사한 순서를 가지게 된다. 하지만 이 방식의 의도는 플레이어가 초반 레벨을 빠르게 끝마치며 성취감을 얻도록 하여 게임의 플레이 메카닉과 화면/음향에 익숙해지도록 하는데 있다. 레벨 별로 평균 플레이 시간을 결정할 때에는 많은 수의 플레이어들을 상대로 플레이 테스트를 해보는 것이 좋다는 것을 명심하라.

주관적인 난이도 순서

이 방법은 대부분의 게임들에서 사용하는 고전적인 레벨 구조이다. 간단한 레벨이 초반부에 등장하고 끝마치기 어렵고 복잡한 레벨은 나중에 등장하는 것이다.

주관적인 레벨의 난이도를 정확하게 결정하려면
레벨 별 플레이 시간, 레벨 별 소모 생명 수, 레벨 별 파워 업 사용 개수 등과 같은 데이터를 많은 수의 플레이 테스터에게서 수집해야 한다. 평균적으로 레벨은 플레이 시간 90초, 생명 소모 0, 파워 업 사용 0에

서부터 레벨이 조금씩 높아지며 최대 평균 플레이 시간 6분, 생명 소모 4, 획득한 파워 업 소모 80%까지 올라가게 된다.
 이렇게 수집된 정보는 다음과 같은 잠재적인 레벨 설계의 문제점을 보여준다. (a) 아무도 특정 파워 업을 획득하지 못함, (b) 하나의 맞추기가 불가능한 벽돌 때문에 완료에 추가적으로 5분이 더 소모됨

게임의 난이도가 지속적으로 가파르게 상승하지 않도록 주의하라. 플레이어가 더욱 어려운 도전 과제로 넘어가기 전에 객체/적/목표에 익숙해질 수 있도록 계단 형태의 난이도 상승률을 보이도록 하라.

시각적 영향력 순서

대다수의 벽돌 깨기 스타일의 레벨이 게임을 시작하자 마자 레벨 전체를 볼 수 있다는 것은 매우 흥미로운 사실이다. 즉, 레벨을 스크롤시켜서 화면에 새로운 도전 과제들을 표시하지 않는 이상에는 플레이어가 처음 몇 초 안에 레벨의 잠재적인 플레이 가능성에 대한 꽤 정확한 느낌을 받을 수 있다는 것이다. 이런 깨달음의 순간을 위해 게임을 플레이 하는 플레이어를 목표로 한다면 레벨의 시각적 영향력의 순서대로 레벨을 배열하면(난이도와 관계없이) 이런 플레이어들의 욕구를 만족시켜줄 수 있을 것이다. 매우 상징적인 레벨(공룡, 로봇, 얼굴과 같은)이 게임의 품질/복잡성을 대표하는 게임과 같은 경우에는 이런 구조가 최적일 것이다.

플레이어(또는 플레이 테스터) 선호도 순서

가장 재미있는 콘텐츠를 초반부에 배치하라. 이 순서가 전통적인 레벨 순서와 다르고 대다수의 플레이어들이 게임의 절반도 채 플레이 하지 않기는 하지만(심지어 돈을 내고 구입한 후에도) 체험판을 플레이 해본 플레이어를 돈을 지불하는 고객으로 전환시키는데 도움이 될 것이다. 하지만 이 방법에도 큰 단점이 존재한다. 이렇게 배열할 경우에는 상대적으로 끝이 흐지부지될 수도 있고 평판이 나쁜 리뷰도 산더미처럼 쌓일 우려가 있다. 이런 경우를 방지하려면 단순히 특정 기준을 만

족시키지 못하는 레벨들을 버려서 재미없는 레벨이 존재하지 않도록 만드는 것이다...

메뉴

플레이어가 특정 레벨에서의 통계와(난이도, 전체 성적, 평균 플레이 시간 등) 게임의 진행 상황을 알 수 있도록 레벨 액세스 시스템(Level Access System)을 제작하라. 각 레벨의 등급이 표시되는 메뉴와 같은 것을 말이다. 그러면 플레이어는 플레이 가능한 레벨을 자신이 고를 수 있으며 그로 인해 게임에서 최상의 콘텐츠를 모두 플레이 해볼 가망성이 높아진다. (나는 이런 방식의 시스템을 *둠[DOOM]*의 사용자 제작 레벨 모음집 CD-ROM에서 본 적이 있다.)

*브레이크 퀘스트(BreakQuest)*는 플레이어에게 각 레벨의 작은 미리 보기 이미지를 보여주는 메뉴를 가지고 있다. 레벨 전체의 플레이 가능성을 보여줄 수 있을 만큼 크지는 않지만 이런 메뉴 항목으로 인해 해당 레벨에 어떤 것들이 있는가에 대한 단서를 제공해 플레이어를 약간이나마 만족시켜줄 수 있다.(하지만 이 메뉴 시스템은 최소한 처음부터 모든 레벨에 접근할 수 있도록 허용하지는 않는다. 여전히 플레이어가 각 레벨을 순서대로 끝마쳐야 해당 레벨에 접근할 수 있도록 허용하기 때문이다.)

혼합 접근법

레벨의 순서를 정하는 것에 대한 좋은 혼합 접근법은 바로 다음과 같다. (a) 크기와 플레이 시간이 잘 조화되어 있는 주관적인 난이도 순서로 시작, (b) 매 5~10 레벨 마다 시각적 영향력을 가진 레벨(해괴한 형태나 보스)을 흩어서 배치, (c) 데모에서 사람들이 정식으로 넘어오는 비율을 높일 수 있도록 몇 개의 플레이 테스터 선호 레벨을 전반부에 배치하라. (플레이 테스터 선호 레벨이 민감한 레벨 진행과 조화를 이루지 못한다면 “보너스 레벨”로 부르도록 하라.) 마지막으로 플레이어가 이미 끝마친 레벨을 빠르게 다시 플레이 해볼 수 있도록 메뉴 시스템을 추가하는 것을 명심하라.

다수의 레벨 관리에 대한 조언

가장 간단한 벽돌 깨기 스타일의 레벨은 몇 분만 투자하면 만들 수가 있으며 약간 복잡한 벽돌 깨기 레벨은 일반적으로 제작에 하루 이상의 시간이 소요되지 않는다. 심지어 스위치와 파워 업 아이템이 무수히 많이 들어간 아주 복잡한 스크롤 가능한 레벨을 제작한다고 해도 몇 일만 투자하면 테스트까지 충분히 끝마칠 수가 있다. 따라서 한 명의 레벨 디자이너는 한두 달 이내에 100개 이상의 레벨을 제작할 수 있을 만큼의 작업을 할 수 있다.

혼자서 60여 개의 간단한 레벨로 구성된 자그마한 프로젝트를 진행하고 있다면 파일 이름을 사용해 콘텐츠를 관리할 수 있을 것이다. 하지만 (a) 여러 명의 디자이너와 함께 작업하는 경우, (b) 수백 개의 복잡한 레벨이 포함된 경우, (c) 오랜 기간에 걸쳐 제작을 하는 경우, (d) a, b, c의 모든 경우가 전부 포함된 경우 등에는 상세한 기록을 남겨 레벨을 관리하는 것이 좋다.

일관된 레벨 설계 고수하기

레벨의 진행 상태를 살펴보며 언제, 어디서, 어떤 이벤트나 기능을 사용할 것인지의 목록을 작성하라. 예를 들면, “벽돌 종류 6번은 전문가용 객체이다. 쉬움이나 보통의 레벨에서는 사용하지 말 것.”과 같이 말이다. 시간이 좀 있다면 레벨 편집기 자체에 이런 기능을 추가하여 편집기 상에서 쉬운 레벨 제작을 선택하면 쉬운 레벨에서 사용 가능한 객체만을 사용할 수 있도록 하는 기능이 있다면 좋을 것이다.

일관된 설명 고수하기

레벨 데이터에 “레벨 종류”라는 항목을 추가하라. 편집기의 드롭다운

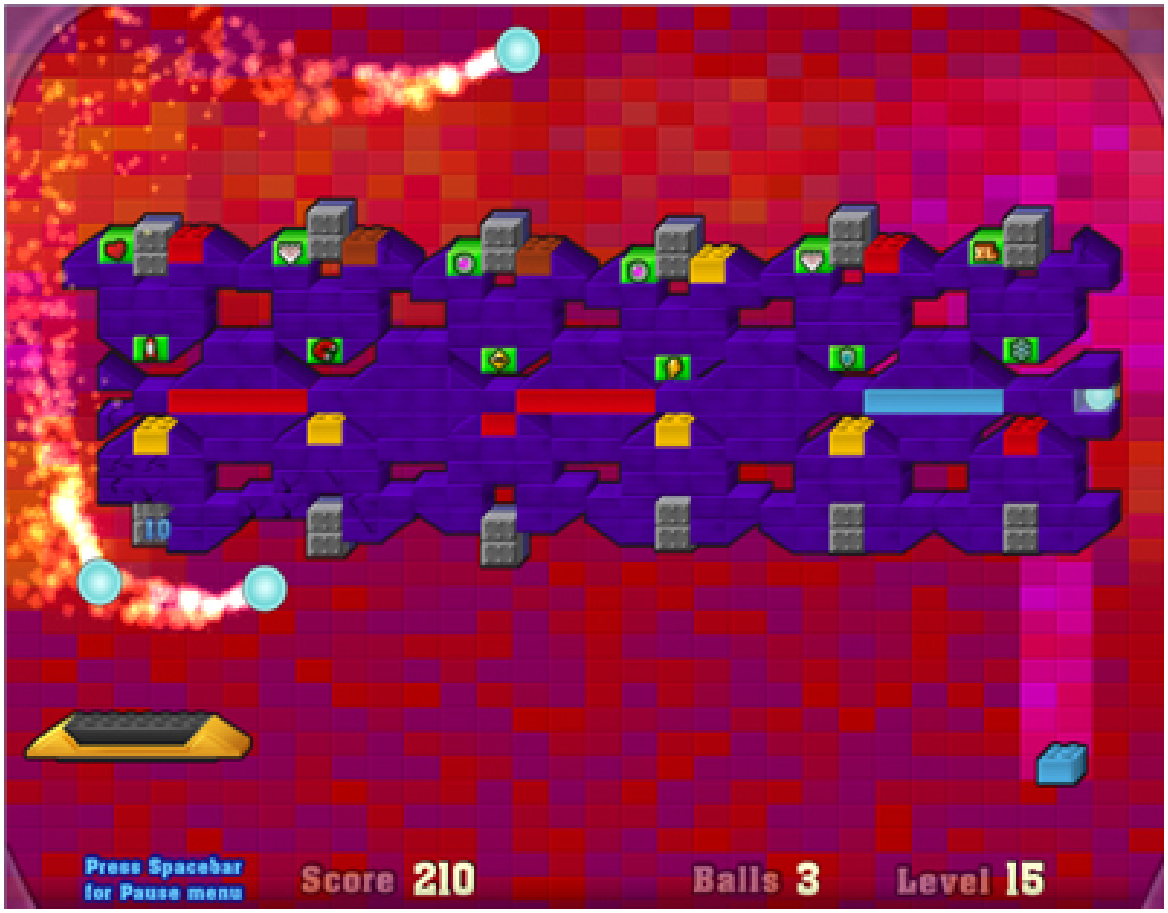
(Dropdown) 메뉴에 “퍼즐 레벨”, 또는 “보스 레벨”과 같은 미리 설정된 값을 추가해 놓아라. 파일들이 XML 형식으로 저장되어 있다면 이런 추가적인 값으로 인해 Windows 시작 메뉴에 있는 검색 도구에서 매우 손쉽게 특정 레벨을 찾을 수 있을 것이다. 특정 항목을 파일에 내장하고 종류에 따라 관리하는 것이 파일 이름에 “p“나 “b” 등의 접두사(예: 퍼즐은 “p레벨이름”, 보스는 “b레벨이름”)를 붙여서 관리하는 것보다 훨씬 안전한 방법이다.

게임 객체의 사용률 파악

게임 시스템이 원조 벽돌 깨기보다 조금이라도 더 복잡하다면 모든 레벨 파일에서 사용된 객체의 종류와 게임의 목표(예: 트리거 이동, 스크롤 등)를 쉽게 파악할 수 있어야 한다. 레벨을 통계에 기반해 정렬하고 그룹을 만들 수 있다면 대량의 레벨 관리에 있어 최대한의 유연성을 제공해줄 것이다.

각각의 레벨 파일을 불러와서 벽돌의 개수와 기타 게임 토큰들을 세고 나서 Excel에 결과를 수동으로 기록하는 것보다는 레벨 파일을 분석해서 그 안에 포함된 개체들의 상세한 통계를 생성하는 유틸리티를 제작하는 편이 좋다. 유틸리티에서 쉼표로 구분된 텍스트로 결과물을 출력해 스프레드시트로 옮길 수 있어야 한다. 그러면 스프레드시트의 정렬 도구를 사용해 상대적인 분류와 진행 상황을 빠르게 볼 수 있을 것이다.

또한 게임 객체 사용률 데이터와 플레이 테스터가 작성한 데이터를 서로 짝지어 보라.(**레벨 진행 – 주관적 난이도 순서** 참조) 이렇게 짝을 지어놓으면 레벨의 순서 결정과 미세 조정에서 매우 강력한 도구로 사용할 수 있을 것이다.



다양한 통계에 기반한 게임 조율

Nintendo DS(더 젊고/하드코어적 성향의 게이머)를 목표로 하는 벽돌 깨기 스타일의 게임들의 경우에는 캐주얼/다운로드 시장(좀 더 나이가 있고/간혹 여성 게이머도 있는)과는 상당한 차이점이 있다. 대부분의 벽돌 깨기를 플레이 하는 게이머들의 경우에는 그 두 가지 극단적인 성향을 가진 집단의 사이에 자리잡고 있지만 여기에 각각의 성향을 가진 게이머들을 위해 레벨을 조율하는 방법을 알려주겠다.

캐주얼 게이머 – 더 큰 벽돌과 한 번 공과 충돌하면 파괴되는 벽돌을 사용하고 벽돌 더미와 패들의 사이에 초기 간격을 넓혀라. 약간의 '자동 조준', 더 넓은 패들을 사용하고 초기의 공 속도를 약간 느리게 만들고 공의 "가시성을 높여주는 장치"를 사용하라.(더 큰 공, 차별화된 색깔, 잔상 등)

하드코어 게이머 – 다중 충돌 벽돌과 특이한 목표 객체의 사용을 주저하지 말고, 벽돌 더미와 패들 사이의 간격을 줄이고, 파괴 불가능한 벽돌과 스위치를 더 많이 사용하고, 패들을 더 작게 만들고, 공이 목표와 충돌할 때마다 속도를 높이고, 파워 다운 아이템을 활용하고, 파워 업이 없이는 끝마치기 힘들게 만들고, 파괴하고 피하는 형식의 플레이를 좀 더 많이 제공하라.

유념해야 할 마지막 사항들

성실하라 – 작업을 완료한 레벨을 주기적으로 다시 플레이 해보는 습관을 길러라. 게임 엔진의 아주 사소한 변경만으로도 매우 처참한 결과가 나타날 수 있다. 일반적으로 이런 일은 새로운 기능을 추가하려다가 자주 발생한다.

차별화하라 – 상업적인 제품을 개발하고 있다면 차별화되는 기능과 플레이 메카닉을 활용해 *알카노이드(Arkanoid)* 및 다른 경쟁자들과의 명확한 구분을 지어야 한다.

명확하게 하라 – 게임 디자인의 방향을 명확하게 잡고 일관되게 지켜서 플레이어에게 혼란을 주지 말아라.(게임 토큰 우선순위 참조) 초반부의 레벨들은 게임의 독특한 느낌을 전달해야 하기 때문에 특히 더 중요한 요소이다.

영리해져라 – 플레이어에게 게임의 특정 기능을 가르쳐준 후에 기존의 기능들을 조합해 새로운 플레이 방식을 창조하라. 이는 게임의 가능성을 더 증대시킬 뿐만 아니라 완전히 새로운 플레이 메카닉을 도입하지 않고도 다수의 재미있는 레벨을 제작할 수 있도록 해주기 때문에 아주 좋은 방법이다.

자비심을 가져라 – 캐주얼 플레이어를 겨냥해서 게임을 제작하고 있다

면 (특히 게임의 초반부에) 벽돌 더미를 패들에 너무 가깝게 배치하지 않도록 주의하라. 그리고 공은 최대 속도에서도 눈에 잘 띄어야 한다!

잔인해져라 – 재미없는 성공은 명예로운 실패보다 재미가 없다는 사실을 항상 명심하라.

정확해져라 – 하드코어 알카노이드 팬을 겨냥해서 게임을 제작하고 있다면 공의 움직임과 패들의 조작이 매우 정확해서 공을 정확한 위치로 보낼 수 있도록 하라.(공, 패들 참조)

존경을 표하라 – 한 번이나 두 번쯤 "고전적인" 게임 플레이를 제공한다고 그리 걱정하지 말아라. 장르를 개척한 선조들에 대한 경의를 표하는 것 외에도 이는 게임의 새로운 플레이 방식 변화를 더 강조하는 효과를 가지고 있다.(Killer List of Videogames를 참조하라.)

감사의 말

우선 이 아티클을 받아준 Gamasutra의 Simon Carless씨에게 감사를 표합니다. 제가 제작한 *LEGO 브릭토피아(LEGO Bricktopia)*의 레벨에 대한 소중한 피드백을 보내주신 Large Animal Games의 팀에게도 감사를 전합니다. 또한 물리 기반 레벨 설계에 대한 인터뷰에 응해주신 Nurium Games사의 Fèlix Casablancas씨와 공 시뮬레이션에 대한 많은 것을 알려주신 Large Animal Games사의 Josh Welber씨, 레벨 설계 도구에 대한 도움을 주신 Reflexive Entertainment사의 James Smith씨께도 감사의 인사를 전합니다. 마지막으로 본 문서의 검토에 도움을 주신 Dave Walls(Funkitron), Jorge Herdnadez, Todd Scott씨에게 감사드립니다.

참고 자료/추가적인 읽을 거리

Gamasutra article, "The Father of Home Video Games: Ralph Baer" – http://www.gamasutra.com/features/20060424/hawkins_01.shtml

K. Salen and E. Zimmerman. RULES OF PLAY. The MIT Press. Pages 319-323, 2003
David Sudnow. PILGRIM IN THE MICROWORLD. Warner Books. 1984.
Killer List of Videogames, Coin-Op Museum Search (Ball and Paddle Games) – http://www.klov.com/results.php?genre=Ball%20and%20Paddle
The Dot Eaters – http://www.thedoteaters.com/
Forbes article, "Bally Brings *Pong* to Casino Floor" – http://www.forbes.com/feeds/ap/2007/07/26/ap3958158.html

게임

벽돌 격파 디럭스(Block Breaker Deluxe)(휴대전화용) – http://www.gameloft.com/product.php?product=1&product_name=Block+Breaker+Deluxe
브레이크 퀘스트(BreakQuest) – http://nurium.com/
펑키볼 어드벤처(Funkiball Adventure) – http://www.funkitron.com/games/funkiballadventure
Excel 벽돌 깨기(Excel Breakout)(레벨 편집기 포함) – http://excelgames.org/breakout.asp
하이퍼 볼로이드 합본(Hyperballoid Complete) – http://www.alawar.com/games/hyperballoid/
자르디나인스 2(Jardinains 2) – http://www.jardinains.com/
LEGO 브릭토피아(LEGO Bricktopia) – http://www.largeanimal.com/games/deluxe/lego-bricktopia
매직 볼 3(Magic Ball 3) – http://www.alawar.com/games/magic-ball-3/
너버스 브릭다운(Nervous Brickdown)(DS Lite) – http://www.arkedo.com/nervous/
플라즈마 퐁(Plasma Pong) – (http://www.plasmapong.com/).
퐁(Pong)과 벽돌 깨기(Breakout)(오락실) – 거주 지역에 오락실이 없다

면 다음의 사이트에서 게임을 볼 수 있음.
http://www.videotopia.com/
리코쳇 리차지드(Ricochet Recharged) –
http://www.reflexive.com/RicochetLostWorldsRecharged.html

미디어
수퍼 브레이크아웃 동영상(Super Breakout Video) –
http://www.thedoteaters.com/videos/super_breakout.mpg