crimson alliance POSTMORTE

THE LEADING GAME INDUSTRY MAGAZINE VOL18 NO10

NOVEMBER 2011 INSIDE: THE GAME DEVELOPER TOP 50

UBM

gd

NOVEMBER 2011

GAME DEVELOPER MAGAZINE

50

GAME DEVELOPER MAGAZINE

CONTENTS.1111

VOLUME 18 NUMBER 10

DEPARTMENTS

POSTMORTEM

FEATURES

GAME PLAN // BRANDON SHEFFIELD

JOURNALISTIC RAGE

RUMINATIONS ON HOW FAR IS "TOO FAR" IN DEVELOPER INTERVIEWS

A FEW WEEKS AGO I PUBLISHED an interview on Gamasutra about id's RAGE. I spoke with CEO Todd Hollenshead and artist Andy Chang, and it created a bit of a stir. My line of questioning was perceived by some as abrasive, rude, or even hostile. Others, journalists and indie developers especially, thought I was simply asking tough questions and not letting up when I didn't hear satisfying answers. While the latter is closer to the truth, I had no real angle—we were just having a conversation.

ANGER MANAGEMENT

I played RAGE about two months before launch, in a hotel space in San Francisco, with decent screens and nice headphones. At the beginning of the game, you wake up in an "Ark," and stumble outside. You're almost killed by mutants, but are saved by someone in a nearby car. The next thing you're meant to do is get in the car with him. But game players tend to like to test the limits of systems, so I looked around to see what else I could do. There was another path leading the other direction—I figured I'd see what was up there. "Oh!" I heard behind me. It was Andy Chang. "What's wrong?" I asked. "Nothing," he chuckled. "You'll see." I walked up the path, and was killed instantly by a bullet from an invisible enemy. I got game over, and had to start anew, calibrating my controller all over again. This time I got in the car.

This would be the trend throughout my play experience. My character, instantly ready to kill anyone on command if someone suggested it, was given tasks by the fellow who saved him in the car. I would have to go up to a person who had a task for me, click on them, and they'd introduce themselves. They'd then wait around for me to click on them again to get a mission. I could just wander away if I wanted, and do something else. But there was nothing else to do! If I wanted to progress in any way, I had to just go right back and click on them again. Why give me the illusion of freedom if really all I can do to advance the story is go to the next node? Why give me options that don't actually exist?

I asked these questions of Chang and Hollenshead, because I couldn't figure out why they'd done it this way. This is not some amateur developer, this is id, so they had to have good reasons for their systems, and the makeup of the universe they'd created. The hostile tone people may have picked up on was likely a misinterpretation of my surprise at their responses. I was surprised that there wasn't a reason that all the different factions in the game use different accents. I was surprised there wasn't a better explanation for why the mutants were so artistic. I asked, in a part of the interview that was cut, why they didn't just include the character introductions with the mission briefings. Hollenshead told me this was because they couldn't know when the player would want to do the missions. Maybe they'd just want an intro. This makes sense in some games, but if there's nothing else to do...?

The oddest thing was the surprise felt by Hollenshead and Chang at my questions. How had nobody broached these subjects with them before? It felt as though the game had been developed in a bubble, where they were told everything they were doing was great, without question. I can understand that, it's id after all. But Hollenshead seemed to genuinely appreciate that I had taken a laser-focus to the game's systems, and the air in the room was contemplative, not hostile. We spoke for an hour, and smiled and shook hands at the end.

BLACKLISTED?

In my opinion, my interview with the RAGE folks was not spectacular. It was the bare minimum we should expect from journalists. If you're curious about this or that, or if something doesn't match what you saw, ask a question, no matter how "important" the interviewee may be. Sometimes the best answers can be gotten by playing devil's advocate. In my opinion, developers should be happy to have this sort of discussion. It allows you to explain your game's worldview and defend your gameplay choices, and your answers should tell you a lot about your own product.

Did I look like a jerk? Maybe a little. I would say a lot of the cutting done to the article makes my questions appear to come from nowhere, rather than being part of the hour-long conversation-space they occupied. But the interview addressed some rough spots that few had mentioned before, and which only surfaced once the game was reviewed. The evening the interview went live, I received an email from an anonymous "AAA creative director," saying that "on the basis of your hostile and clearly biased line of questioning I have instructed my PR manager to refuse any and all future requests from you and your outlet regarding our game. Having spoken to industry peers in similar leadership positions, I can assure you that I am not alone."

While I highly doubt the veracity of this email, which also CCed our sales department incidentally, it's interesting that something as simple as asking follow-up questions and not letting go of a topic would be viewed as biased and hostile. I have no bias against id. How could I? They're an amazing developer, and have some of the best talent in the industry. It's out of respect for id that I called them out on what I saw. I gave them an early chance to defend issues with the game that others were undoubtedly going to have upon release. If treating someone else's work the way you'd treat your own—that is to say with scrutiny and criticism—is disrespectful, then we clearly have different definitions of the word.

—Brandon Sheffield
twitter: @necrosofty

GAME DEVELOPER
MAGAZINE
WWW.GDMAG.COM

UBM LLC.
303 Second Street, Suite 900, South Tower
San Francisco, CA 94107
t: 415.947.6000 f: 415.947.6090

SUBSCRIPTION SERVICES

FOR INFORMATION, ORDER QUESTIONS, AND ADDRESS CHANGES
t: 800.250.2429 f: 847.763.9606
e: *gamedeveloper@halldata.com*

FOR DIGITAL SUBSCRIPTION INFORMATION
www.gdmag.com/digital

EDITORIAL

PUBLISHER
Simon Carless I *scarless@gdmag.com*

EDITOR-IN-CHIEF
Brandon Sheffield I *bsheffield@gdmag.com*

PRODUCTION EDITOR
Jade Kraus I *jkraus@gdmag.com*

ART DIRECTOR
Joseph Mitch I *jmitch@gdmag.com*

DESIGNER
Kurt Voklmann

CONTRIBUTING WRITERS
Tom Curtis
Jesse Harlin
Dave Voyles
Jeremy Putnam
Steve Theodore
Andy Firth
Damion Schubert
Matthew Wasteland

ADVISORY BOARD
Hal Barwood **Designer-at-Large**
Mick West **Independent**
Brad Bulkley **Microsoft**
Clinton Keith **Independent**
Brenda Brathwaite **Lolapps**
Bijan Forutanpour **Sony Online Entertainment**
Mark DeLoura **THQ**
Carey Chico **Independent**
Mike Acton **Insomniac**

ADVERTISING SALES

GLOBAL SALES DIRECTOR
Aaron Murawski **e:** *amurawski@think-services.com*
t: 415.947.6227

MEDIA ACCOUNT MANAGER
John Malik Watson **e:** *jmwatson@think-services.com*
t: 415.947.6224

GLOBAL ACCOUNT MANAGER, RECRUITMENT
Gina Gross **e:** *ggross@think-services.com*
t: 415.947.6241

GLOBAL ACCOUNT MANAGER, EDUCATION
Rafael Vallin **e:** *rvallin@think-services.com*
t: 415.947.6223

ADVERTISING PRODUCTION

PRODUCTION MANAGER
Pete C. Scibilia **e:** *peter.scibilia@ubm.com*
t: 516-562-5134

REPRINTS
WRIGHT'S MEDIA
Jason Pampell **e:** *jpampell@wrightsmedia.com*
t: 877-652-5295

AUDIENCE DEVELOPMENT
TYSON ASSOCIATES Elaine Tyson
e: *Elaine@Tysonassociates.com*
LIST RENTAL Merit Direct LLC
t: 914.368.1000

UBM
WWW.UBM.COM

around the Arab world there are more than **180 million** people under the age of 25.*

yet Arabic game development is still in its infancy.

your opportunity is right here, right now at twofour54° in the heart of Abu Dhabi.

The Arab world is one of the world's fastest growing media markets. With a young population of 180 million under the age of 25, more than 80% with mobile phones,* strong broadband take-up and new gaming innovations, it's a prime opportunity for Arabic gaming businesses.

We empower businesses across all media platforms from television, film, radio, gaming, digital, animation and publishing – with world-class training from **twofour54° tadreeb**, state-of-the-art production facilities with **twofour54° intaj** and venture funding and support for Arab creative entrepreneurs from **twofour54° ibtikar** – to seize every media opportunity the region has to offer.

It's all part of our vision at **twofour54°**, creating a centre of excellence for Arabic content creation in Abu Dhabi.

we are twofour54°. are you?
find us. join us. create with us.
+971 2 401 **2454** **twofour54**.com

follow us on:

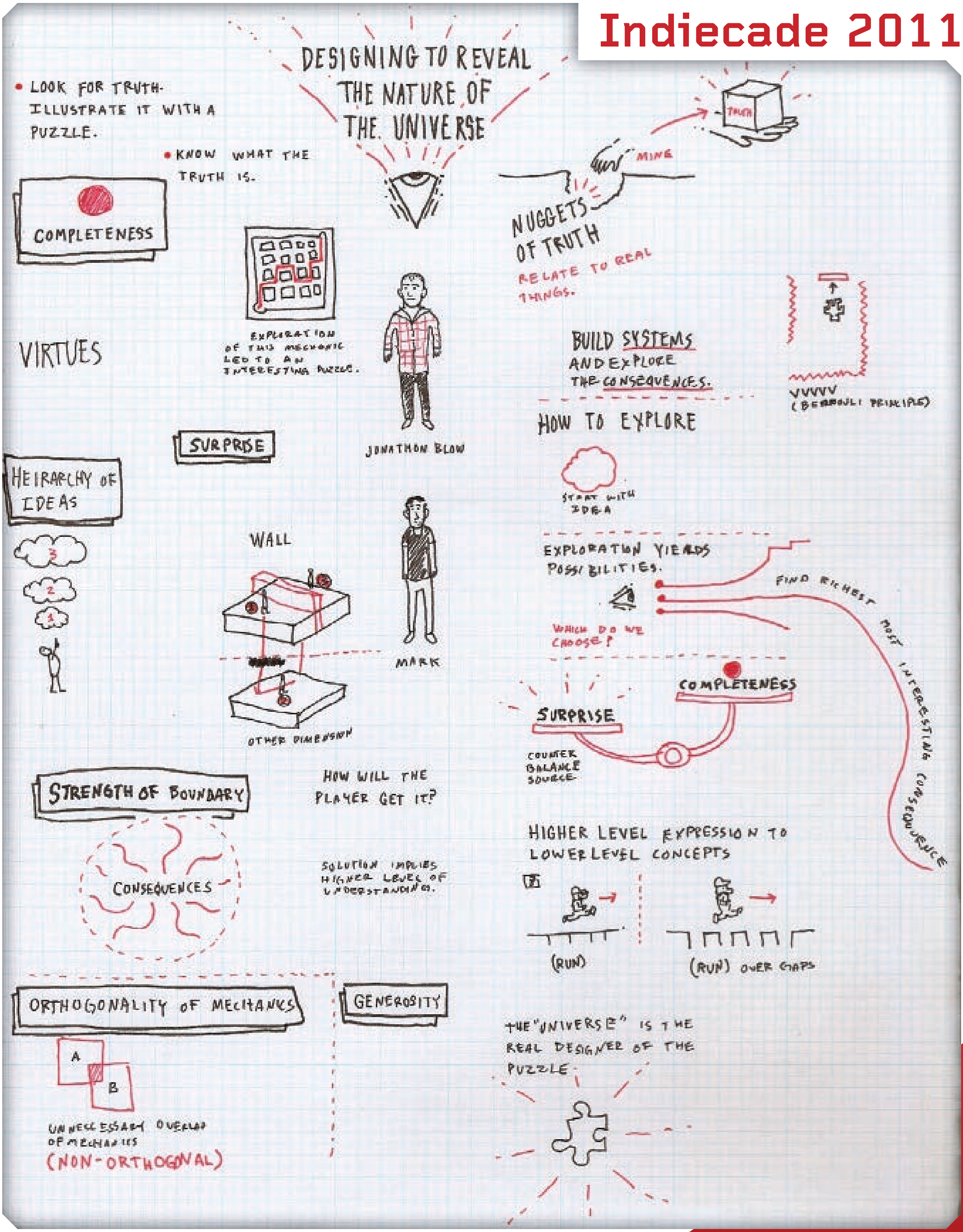
twofour54°
Abu Dhabi
content creation community

Sources: Arab Media Outlook 2010. Media on the Move 2009. A.T. Kearney. Introduction to Gaming. Michael Moore. Screen Digest. IDC.

twofour54° is an initiative of the Abu Dhabi Government.

This year's Indiecade conference has ended, so for those of you who couldn't take part, we present an illustrated version of Jonathan Blow (BRAID) and Mark ten Bosch's (MIEGAKURE) talk, "Designing to Reveal the Nature of the Universe," as illustrated by Gabriel Gaete.

Indiecade 2011


DESIGNING TO REVEAL THE NATURE OF THE UNIVERSE
• LOOK FOR TRUTH. ILLUSTRATE IT WITH A PUZZLE.
• KNOW WHAT THE TRUTH IS.
COMPLETENESS
VIRTUES
EXPLORATION OF THIS MECHANIC LED TO AN INTERESTING PUZZLE.
JONATHON BLOW
TRUTH
MINE
NUGGETS OF TRUTH
RELATE TO REAL THINGS.
BUILD SYSTEMS AND EXPLORE THE CONSEQUENCES.
VVVVV (BERNOULI PRINCIPLE)
HOW TO EXPLORE
START WITH IDEA
SURPRISE
HEIRARCHY OF IDEAS
3
2
1
WALL
MARK
OTHER DIMENSION
EXPLORATION YIELDS POSSIBILITIES.
WHICH DO WE CHOOSE?
FIND RICHEST MOST INTERESTING CONSEQUENCE
SURPRISE
COMPLETENESS
COUNTER BALANCE SOURCE
STRENGTH OF BOUNDARY
CONSEQUENCES
HOW WILL THE PLAYER GET IT?
SOLUTION IMPLIES HIGHER LEVEL OF UNDERSTANDING.
HIGHER LEVEL EXPRESSION TO LOWER LEVEL CONCEPTS
(RUN)
(RUN) OVER GAPS
ORTHOGONALITY OF MECHANICS
A
B
UNNECESSARY OVERLAP OF MECHANICS
(NON-ORTHOGONAL)
GENEROSITY
THE "UNIVERSE" IS THE REAL DESIGNER OF THE PUZZLE.

New retail Atari 2600 game found

Most astute game fans are aware of Atari 2600's E.T. THE EXTRA-TERRESTRIAL, of which thousands of cartridges are reportedly buried in the desert. But apparently there was another 2600 EXTRA TERRESTRIALS game developed for the system, that wasn't discovered until mere weeks before this issue went to print.

Skill Screen Games, a now-defunct three-man studio that was based in Ontario, and which only released one game, produced EXTRA TERRESTRIALS in 1984, shortly after the crash. The company only managed to sell a couple hundred of the game's shorter-than-usual cartridges to local retailers.

Somehow, the game escaped the attention of even the most hardcore collectors, and only surfaced when a family member of one of the developers donated the cartridge (without packaging materials, unfortunately) to the Personal Computer Museum, which describes the event this way.

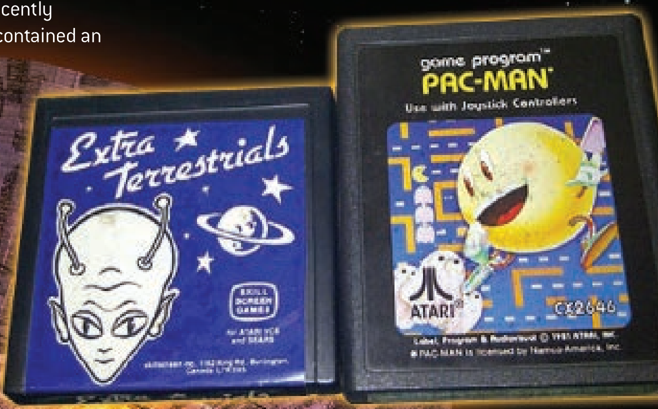
"Museum curator and founder Syd Bolton found himself in a state of disbelief when fellow volunteer George Yallop delivered a 'contribution' from someone he knew, who had recently visited the museum. The envelope contained an Atari 2600 cartridge called EXTRA TERRESTRIALS.

Searches of the web didn't reveal any information about the game. It was at this point that Syd realized he may have found a long lost game. This was an important discovery to the museum and the Atari community as well."

The game is a two-player-only combination of "FREEWAY, E.T., and PAC-MAN."

Bolton is currently looking for help to dump the ROM to make the new find available for anyone.

—Eric Caoili



Game Developers Choice Online Award winners announced

At this year's GDC Online Awards, which took place October 12, 2011, a jury of peers recognized the best games in the online space, across a host of categories. The biggest winners this year were RIFT and MINECRAFT, which took home awards in two categories each.

EVERQUEST was the second-ever title ushered into the hall of fame, largely due to its incredibly long and successful run as a live service. The game has been actively served and updated since 1999, and had its latest expansion in late 2010.

Finally, John Taylor and Kelton Flinn, creators of DUNGEONS (and ISLAND) OF KESMAI, were presented with the online game legend award. The duo are honored for creating one of the first pseudo-graphical MUDs.

We present here entire list of award-winners. GDC Online is run by the UBM TechWeb Game Network, as is this magazine.

BEST ONLINE VISUAL ARTS
DC UNIVERSE ONLINE (Sony Online Entertainment)

BEST SOCIAL NETWORK GAME
GARDENS OF TIME (Playdom)

BEST ONLINE GAME DESIGN
SPIRAL KNIGHTS (Three Rings Design/Sega)

ONLINE INNOVATION
SHADOW CITIES (Grey Area)

BEST ONLINE TECHNOLOGY
RIFT (Trion Worlds)

BEST COMMUNITY RELATIONS
MINECRAFT (Mojang)

BEST AUDIO FOR AN ONLINE GAME
CLONE WARS ADVENTURES (Sony Online Entertainment)

BEST NEW ONLINE GAME
RIFT (Trion Worlds)

BEST LIVE GAME
MINECRAFT (Mojang)

AUDIENCE AWARD
WIZARD101 (KingsIsle Entertainment)

ONLINE GAME LEGEND
John Taylor and Kelton Flinn

HALL OF FAME
EVERQUEST (Sony Online Entertainment)

GDC
12

THE GAME DEVELOPER 50

BRANDON SHEFFIELD
FRANK CIFALDI

SHADOW OF THE COLOSSUS

GAMES ARE A COLLABORATIVE ART, TO BE SURE, BUT THE INDIVIDUAL MEMBERS OF A TEAM CAN BE IMPORTANT TOO. FOR THE THIRD YEAR RUNNING WE'VE COMPILED A LIST OF 50(ISH) PERSONS WHO HAVE MADE A SIGNIFICANT IMPACT ON THE GAME INDUSTRY THIS YEAR, THROUGH GUIDING TEAMS, COMPLETING SOLO PROJECTS, OR SIMPLY SETTING A STANDARD FOR OTHERS TO FOLLOW.

We've done our best to credit the right folks for achievements in art, design, programming, business, and evangelism, but it should be noted that these are not ranked lists. And there are certainly more than 50 important individuals in this industry, but our pages would never fit them all.

Thanks to those mentioned (and apologies to those not) for continuing to shape our rapidly changing industry. There's no time to slow down now!

[ART]

Craig Adams

/// SUPERBROTHERS

It's not often that someone tries to create a new style of pixel art, but that's what Craig Adams has done through his ambiguously pluralized one-man company Superbrothers. He has chosen to call his style "rustic 21st century minimalism," and if you can get past that mouthful of a description, you'll find he's on to something.

In SWORD AND SWORCERY for iOS (cocreated by Capy Games), Adams demonstrates a style that is at once efficient in its use of space and incredibly emotionally evocative. His simplified pixel work is more representative than realistic, and so has struck a chord with both the traditional art and game worlds—an unusual feat.

Ren Yamazaki

/// GRASSHOPPER MANUFACTURE

Grasshopper has long been known for its unique visual styles, with titles like KILLER 7 pushing game visuals into the realm of the experimental. With the upcoming downloadable title BLACK KNIGHT SWORD, game and art directed by Ren Yamazaki, the company has found a curious puppet show and theatrical production hybrid art style, layering 2D images in three dimensions to create a new and instantly engaging visual theme. The game's characters are also different from the norm, with lumpy not-quite-human forms that lope along with their purposefully halting animations. Here's hoping for more big things from Yamazaki's altered brain.

BLACK KNIGHT SWORD

Jonathan Jacques-Belletête

/// EIDOS MONTREAL

DEUS EX: HUMAN REVOLUTION's bold art style combines cold cyberpunk sensibilities with, of all things, Renaissance-era artwork. The result, which art director Jacques-Belletête has dubbed "Cyber-Renaissance," is the poster child of proper art design: The game looks like nothing else out there, and its aesthetic actually manages to say something about body manipulation and the discovery of self, both of which are persistent themes in the game.

As Jacques-Belletête tells it, the team could have easily gone full *Blade Runner* and made a gorgeous game, but he didn't think that would say anything new.

Brian Min

/// DOUBLE FINE

Brian Min is lead sound designer for Double Fine, and makes our list for his unique take on audio for STACKING (XBLA/PSN). The game blends an excellent scene-setting score with "dialog" that's full of character, all without using traditional language. The game's denizens use unintelligible chatter to communicate, with a charming and somehow understandable result.

The game uses tropes and techniques from silent film (including text, of course) to communicate themes and ideas, which are at times complex. On top of the individual character "dialog," a dynamic walla system helped to fill out the backgrounds with vocal noise that made each place feel like a living world—but one which you are visiting as a foreigner. And in fact, isn't that what games are, in their essence?

Andy O'Neil, Marco Thrush

/// BLUEPOINT GAMES

Bluepoint's high definition recreations of the GOD OF WAR and METAL GEAR SOLID series—and more recently Sony's THE ICO AND SHADOW OF THE COLOSSUS COLLECTION—are beautiful examples of bringing a game into the modern era while lovingly preserving the source material. Textures have been redrawn, and certain liberties had to be taken to get the games running in widescreen, but all to great effect.

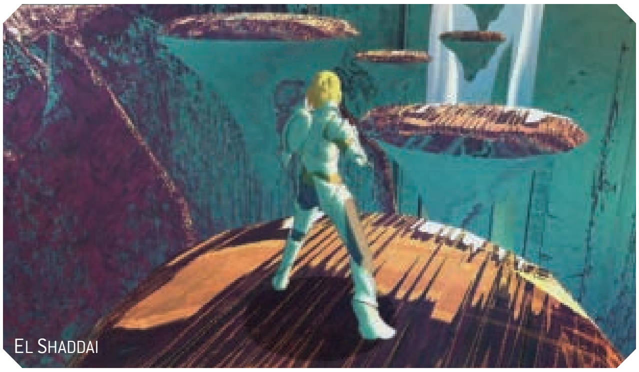
O'Neil

The games look and run just like your rose-tinted nostalgia remembers them, not how they actually were, and the tweaks to modernize them were done with a level of respect worthy of their source material. Bluepoint president Andy O'Neil and CTO Marco Thrush have built a company that perfects nostalgia.

Takeyasu Sawaki

/// IGNITION JAPAN

El Shaddai is one of those rare commercial games that pushes the concept of what we consider HD visuals. The game constantly shifts its form and challenges the player's visual perceptions in unexpected ways, but manages to retain a cohesive look, rather than becoming a scattered pastiche.


EL SHADDAI

In one section of the game, you'll find amorphous shifting colors with a cel-shading technique that brings to mind CG cutscenes of the 56-color era. In another, you'll find neon-on-black '80s-style futurism. In yet another, a two-dimensional platforming scene will call to mind moving Japanese Ukiyo-e. The game's art style, directed by game director Sawaki, boldly embraces the unreal, a rare and admirable quality.

Stuart Aitken

/// AXIS ANIMATION

Prior to February, DEAD ISLAND was not really on the radar. Polish developer Techland (CALL OF JUAREZ) was not a household name. Publisher Deep Silver had a small cult following with titles like CURSED MOUNTAIN and SACRED 2, but never had a real hit. DEAD ISLAND was shown at E3 2010, but was only mentioned as a footnote.

DEAD ISLAND

All of that changed in 2011 when a gripping three-minute CGI trailer debuted on YouTube, featuring a haunting piano theme and a tragic story of a vacationing family succumbing to a zombie attack. The trailer, directed by Stuart Aitken of Axis Animation, attracted over 7 million views, and the game shipped 1 million units in its debut week. We don't mean to undermine Techland's work, but a surprise hit of this magnitude almost certainly wouldn't have happened without such creative and artistic marketing.

Gustav Tilleby

/// EA DICE

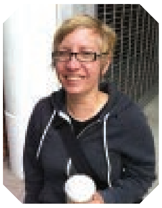
EA DICE's Frostbite 2 engine, which made its debut with BATTLEFIELD 3, has an impressive number of toys and tricks to make its lush environments among the most realistic in games. But deferred shading and real-time velocity lighting can only get you so far: It takes a real artist's touch to make a scene look great.

Art director Tilleby and his team have proven themselves masters at staging a scene and making sure the player is seeing exactly what she needs to see. It's an art form that's often overlooked because, like any good storytelling, it's best when you don't notice it.

Tasha Harris

/// PIXAR (EX-DOUBLE FINE)

We all let out a collective cheer when we heard that Double Fine was switching its business model from releasing one triple-A title every few years to pumping out several smaller, more experimental digital games. This is a studio full of creative thinkers and tinkerers, so freeing itself from long and arduous dev cycles is the best thing it could have done.



Harris, a Double Fine animator for five years, made her directorial debut with COSTUME QUEST, which brings her charming art style to life. Inhabiting her world brings us back to our childhoods in ways that most other media can only dream of. She's now off to work at Pixar, presumably because she hates being on "best of video game" lists.

Josh Randall

/// HARMONIX

How does one describe VIDRHYTHM, the iOS debut from Rock Band and Guitar Hero creator Harmonix? It's not quite a game—there are few rules to abide by, and no failure/reward system—but it's not just a tool either.

All we know is that it's a lot of fun to play with, and turns even the least astute amateur into a music video director. It's almost impossible to make a VIDRHYTHM video that isn't at the very least entertaining, an achievement only true masters of interactive rhythm could have pulled off. Creative director Randall and his team are those masters.

[DESIGN]

Michel Ancel

/// UBISOFT

Michel Ancel is a designer of unique vision, finding new ways to make action games meaningful with every project. Now, with RAYMAN ORIGINS, he has distilled what makes 2D platformers great, and added four players alongside inspired beautiful level design. RAYMAN ORIGINS is unfiltered fun, and feels humorous and accessible without sacrificing challenge or lacking precision.

This is the kind of project that rarely gets major financial backing, so one has to praise Ubisoft for indulging in this experiment. ORIGINS is also the proving ground for Ancel's design-oriented development toolset, which he hopes will be used for many future projects.

Kim Swift

/// AIRTIGHT GAMES

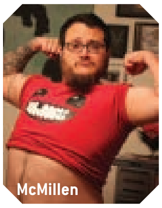
PORTAL and LEFT 4 DEAD designer Kim Swift is not afraid of stereotypes. Her new game, Airtight's upcoming Quantum Conundrum, sees the protagonist manipulating his environment by jumping in and out of different dimensions in a first-person perspective, to try to reach a series of exits and advance to the next room.

If it sounds like Portal, Swift doesn't disagree with you. As she tells it, first-person puzzle games are simply the kinds of games she wants to play, and so that's just what she's going to make. In Quantum Conundrum, Swift blends iterative design and experimental play with cinematic visual design to lead players to a goal, even if they don't realize it. It turns out that Portal was just the beginning of her evil scheme.

Edmund McMillen, Florian Himsl

/// TEAM MEAT, KOMIX GAMES

The Binding of Isaac is a Zelda-style roguelike shooter based on a biblical story—a curious combination to be sure, but one that has proved quite compelling to players. This sort of game genre mashup has become all the rage lately, and McMillen and Himsl's latest proves the concept. The key is to keep control tight, no matter what you do, something that the duo excels at remarkably.


McMillen

Additionally, McMillen may be the most outspoken developer on our list, having gone on public record about his grievances with the traditional publisher model and with distribution contracts. He has become something of a spokesperson for the indie designer.

Daisuke "Pixel" Amaya

/// STUDIO PIXEL

It's hard to believe, but we're about to hit the seventh anniversary of Pixel's 2004 retro-inspired indie platformer, Cave Story. Despite not having released a substantial game since (though he's working on an iPhone project), Cave Story still represents a lot of hope for game developers. Hope for a return to simpler times, for Cinderella stories, and for the ability of one person to affect many.

The game continues to be released in new iterations on new platforms. The latest, which Amaya is working on directly, is a 3DS reimagining called Cave Story 3D, which should be available at about the time you're reading this.

Katsura Hashino, Shigenori Soejima

/// ATLUS

Catherine

The difficult and possibly sexist storyline of love and infidelity told in Catherine might be polarizing, but the effectiveness in which it is told is worthy of praise. Vincent may be cheating on his girlfriend, but this doesn't happen in a cutscene: It's you, the player who gets him there. It's you who experiences his nightmares, who pushes him toward worse and worse decisions, and who makes the choices that ultimately affect his destiny.

Telling story through gameplay, regardless whether you agree with the story, should always be promoted for advancing our medium in its own way, and that's why game director Hashino and artist Soejima make our list.

Seth Sivak, Jesse Kurlancheek

/// ZYNGA BOSTON

Adventure World is Zynga's next step in moving the social game space toward more traditional mechanics. With an *Indiana Jones*-inspired theme and colorful maps, the game appeals more to the core gamer than many past efforts, and the puzzle-based design makes it even more of a "real game" than many other titles on the platform. Adventure World may not be the most core game on social networks, but Zynga is the industry leader. Designers Sivak and Kurlancheek at Zynga Boston's return to core game design should make the rest of the social space sit up and take notice.

Yoshinori Ono

/// CAPCOM

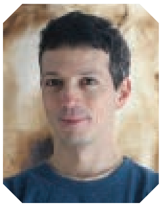
Yoshinori Ono is the curator of the Street Fighter brand for Capcom, continually breathing new life into the once-stagnant fighting genre, year after year. Now, as he works on Street Fighter X Tekken, he brings two fighting systems together in one universe, essentially making an amalgam of the two playstyles.

Tekken players can't play the game as though it were Tekken, and Street Fighter fans will find that their combo timing and movelists have changed. But in the fusing of these worlds, Ono and his team have found a play style that is at once both systems and neither, while remaining fun to play, and intuitive for both sides. This is no mean feat.

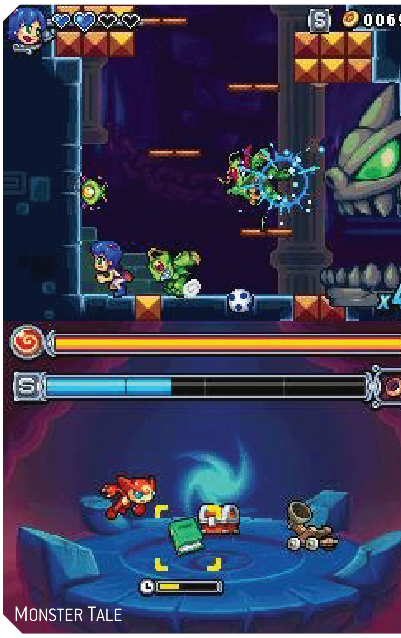
Eric Chahi

/// UBISOFT

Chahi is not a normal fellow. After designing Another World and Heart of Darkness, he went on a 10-year jaunt away from games, photographing volcanoes and painting, before deciding he had something to say in the digital space again. He wanted something to be proud of, he told us during the game's creation, and From Dust, which Chahi directed, certainly is something any designer could take pride in.



Its organic systems, self-perpetuating natural evolution, and simple input make emergent gameplay the only gameplay. Chahi proves that when your influences extend beyond games, you can create something significantly different that still appeals to a wide range of people.


Monster Tale

Peter Ong

/// DREAMRIFT

DreamRift's Monster Tale for the Nintendo DS is one of those extremely rare games where players care equally about what's happening on the top and bottom screens. On the top screen, you're playing a 2D action platformer, assisted by an evolving monster. On the bottom screen, when the monster's energy is depleted (or any time, really), you have him eat different food and interact with different objects in order to evolve along a branching skill tree.

This kind of balancing act is very difficult to achieve, and while there are stumbling blocks here and there, Ong's design finally realized a slice of the full potential of the Nintendo DS—and it only took until 2011!

Julian Gollop

/// UBISOFT

Gollop is well known for his commitment to turn-based strategy game design, as mastermind of the seminal X-COM series. It should come as little surprise that strategy is what puts him on our list, as his design in

2011's Ghost Recon: Shadow Wars was exactly what console strategy fans were looking for.

The game is an intelligent mix of shooting and close-range combat (though some may lament the somewhat decreased complexity) that is good enough to overcome a limp story and uninspiring visuals. This propelled the game to become the star of the 3DS' limited launch.

[PROGRAMMING]

John Carmack

/// ID SOFTWARE

What is left to be said about legendary programmer John Carmack? He's the gift that keeps on giving, someone whose dedication to his craft appears to be endless. His latest work, id Tech 5 (the engine used on Rage and the upcoming Doom 4), still shows that programming ingenuity will always beat out hardware. The system's ability to stream massive amounts of texture data in real time makes Rage look more detailed and fluid than just about any other game on the market.

id's acquisition by Zenimax let Carmack ditch some of his management duties and get back to programming, and the world is a better place because of it.

Andrey Iones, Greg Hermann

/// SABER INTERACTIVE, 343 INDUSTRIES

Fans of the original Halo: Combat Evolved are among the most rabid in games: Make one small tweak to the original's design, and they're going to notice. That's what makes Saber Interactive and 343 Industries' approach to HD remake Halo: Combat Evolved Anniversary unique, with tech led by Iones of Saber, and Hermann of 343.

Rather than try to recreate the experience, the game runs two rendering engines simultaneously. The original game is running at all times in the background, with few changes made, while a new rendering engine by Saber Interactive provides a facelift on top of it. That means none of the subtleties (or even bugs) of the original are left behind. It is literally the same game with a new coat of paint. (Just don't tell the super fans about the multiplayer maps.)

Greg Barwis

/// TRION WORLDS

Large-scale MMO launches are mainly a thing of the past, but not only did Trion Worlds launch Rift with over one million preregistered users, it did so without any major technical hurdles or a decline in user experience. Players were able to log on, pound the hell out of the game's servers, and play it just as intended.

The amount of scalability testing that must have gone into such a successful launch is certainly astounding, and was led by VP of service operations Greg Barwis and his team. This was impressive enough to industry peers that the game won Best Online Technology at this year's GDC Online Awards.

Ben Wyatt

/// ROCKSTEADY

Epic Games might promote Unreal Engine 3 as an a Swiss army knife engine capable of creating any kind of game, but for the most part the titles we've seen—including Rocksteady's previous title, Batman: Arkham Asylum—have largely focused on indoor environments and outdoor worlds of limited scope.

The scalability and size of Batman's virtual playground in Batman: Arkham City defy our expectations of what the engine is capable of, and surely have even Epic's gurus sitting up and paying attention. Batman's able to jet around the world, perch on a rooftop, then suddenly drop twenty stories down without texture pop or noticeable transitions, all while looking gorgeous and controlling precisely. Rocksteady technical director Ben Wyatt and his team have pushed the engine into bold new territory.

Batman: Arkham City

Markus Persson

/// MOJANG

Another list, another entry for Minecraft. But what can you do? Persson has done an excellent job scaling his game as more players have gotten involved, while also fixing bugs and responding to feedback. Mojang hasn't been resting on its laurels, and continues to push forward even with a small team, and it's Persson's solid systems that allow this to happen (though they did require a bit of a code rewrite at one point—let's ignore that). Persson is part of a new breed of "do it all" programmers that calls to mind the bedroom Amiga programmers of the '80s, in all the best ways.

Olga Sorkine

/// ETH ZURICH

While she doesn't work in games, Sorkine's research represents the direction that technology is moving, especially in the field of character animation. Sorkine is currently doing research at the Swiss Federal Institute of Technology Zurich, and was previously assistant professor at the NYU computer science department. She recently won the Significant New Researcher Award at SIGGRAPH based on her research on geometry processing, specifically differential coordinates and interactive mesh editing.

Most immediately relevant to games is her work on 3D model editing and creation using collections of sketched curves. While a short description doesn't do her research justice, much of it can be found online, or through past SIGGRAPH talks.

Dimitar Lazarov

/// TREYARCH

Imposing a mandatory 60 frames-per-second performance out of a game like Treyarch's Call of Duty: Black Ops will of course cause your graphics to take a hit, but thanks to Lazarov's clever techniques, you probably didn't notice.

Lazarov's talk at this year's SIGGRAPH on physically-based lighting for the game was insightful, open, and inspiring. His use of one primary source of light per object shows that even a triple-A studio like Treyarch can rely on trickery to stay ahead of the curve, and the game looks excellent as a result. After all, everything we do in games is a bit of digital trickery!

Makoto Anjo

/// CAVE

Anjo is one of the people responsible for the incredibly responsive and pixel-pushing ports of Cave's hardcore shooters (such as Espgaluda II and Dodonpachi) to the iPhone platform, which have won the company praise and favor from arcade players around the globe. He's also the leader of Android-no-kai, a huge Android development community in Japan, and one of the largest in the world.

He has created the hub of the Android development scene in Japan, and his group is part of the very quick uptick in Android game creation in the region. Not unexpectedly, Anjo and company are now taking on the not-insignificant challenge of porting those iPhone monsters to various Android devices.

Alexander Bruce

/// ANTICHAMBER (FORMERLY HAZARD: THE JOURNEY OF LIFE)

Alexander Bruce has been working on ANTICHAMBER by himself for more than two years now, but it is finally nearing release. Bruce has created a mind-bending physics-reliant first person shooter/puzzler which references PORTAL, HALF-LIFE, and expectation-defiers such as the 2D platformer I WANT TO BE THE GUY. The game continues to win independent game awards and honorable mentions, and the fact that Bruce coded (and created art for) it all himself, at quite a young age (24, as of this writing), is an even more impressive feat of coding.

Geremy Mustard

/// CHAIR ENTERTAINMENT

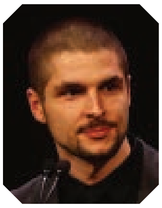
Chair appears to have taken it upon itself to prove what can be done with iPhone development, while simultaneously proving Unreal Engine 3's viability on the platform. With INFINITY BLADE 2, under the technical direction of Geremy Mustard, consumers' already-shattered expectations got ground into a fine mist, as the game looks near current gen console quality. This should come as no surprise, since the new iPhone 4S rivals the PlayStation Vita in graphical prowess—but Chair is constantly leading the push toward bigger and badder on the platform, and the technical prowess of Mustard and his team should be lauded.

[BUSINESS]

Dino Patti

/// PLAYDEAD

LIMBO took several years and a fair amount of money to make, especially for a small project. But the team pushed on, and wound up making a game that was not only critically acclaimed, but financially rewarded. The reason CEO Dino Patti is on this list is not because of the game, though. It's because he used the Playdead's new-found profits to buy the company away from its early investors, allowing the team to make its next game without concerns about measuring up to anyone's expectations. This is a solid move for a team with something to prove to the world—more businesspeople should think this way.



David Helgason

/// UNITY TECHNOLOGIES

Unity has been getting bigger and better, now even threatening traditional middleware with its powerful, easy-to-use tools and affordable price. But even as it grows, the company has remained focused on its vision to enable indies and small teams to do big things. CEO Helgason has kept the company on track, allowing the toolset to run rings around slower competition, and even pressuring giants like Unreal and Crytek to release indie versions of their engines. Unity is setting the standard for cross-platform

compatibility and flexible business models, and for this Helgason and co. should be praised.

Jason West, Vince Zampella

/// RESPAWN

We may never know what actually went down between Activision and Infinity Ward cofounders Jason West and Vince Zampella, but we do know this: The results of the May 7 trial may forever shape the future of creative ownership in games.

West and Zampella are accusing Activision of fraud, saying that the duo's removal from Infinity Ward was a breach in the contract between the companies. In addition to damages, the duo is seeking co-ownership of the Call of Duty brand and the rights to make new titles, which could make the future of video gaming's biggest franchise very interesting indeed.

Frank O'Connor

/// 343 INDUSTRIES

Transitioning the beloved Halo franchise from series creator Bungie to a new team (343 Industries) could have been a PR nightmare, but franchise director O'Connor, previously at Bungie, has done right by fans with several initiatives that seem to prove that Master Chief is in the right hands.

In addition to its main focus on Halo 4, the company is also co-developing Halo: Combat Evolved Anniversary, the HD remake of the original game that fans have been clamoring for. The company also holds fan events like this year's Halo Fest at PAX Prime, and a title update to the Bungie-developed Halo: Reach that addresses many of the multiplayer issues fans have been unhappy with. How do you deal with a rabid but fickle fanbase? You give them what they want, but also give them things they didn't even know they wanted yet.

Noah Heller, Chacko Sonny

/// BEACHHEAD

"Games as services" is a term we've been hearing a lot of for the past year or two. As packaged game development gets more expensive and as more people gravitate toward online play, companies are finding it necessary to expand the life of their products beyond the traditional shelf model.

Beachhead's Call of Duty Elite service, under head of development Chacko Sonny and product director Noah Heller, is a strong indicator of where things are going on console. It has daily activities to keep players engaged, original video content, and promises to deliver just about anything a Call of Duty fan might want. Whether the fans react positively is still up in the air, but these sorts of initiatives don't seem to be going away.

John Riccitiello

/// ELECTRONIC ARTS

John Riccitiello said earlier this year that the company would switch from "defense" to "offense," investing in social gaming companies, focusing on big IP and becoming a software platform rather than a traditional publisher.

EA today is a very different beast than it was even one year ago. Its Playfish-developed The Sims Social is poised to become a top Facebook app, its Origin platform is taking on Steam on its own turf, and its PopCap acquisition could make EA's digital initiative a $1 billion annual business. While the future is as uncertain as ever, EA has been able to move remarkably quickly in this rapidly-changing industry, and it's to Riccitiello's credit (in part) that the company has made such progress.

Kaz Hirai

/// SONY COMPUTER ENTERTAINMENT

Some looked at the unveiling of new PlayStation Vita features at Tokyo Game Show and were underwhelmed. We saw a different story—one where a somewhat humbler Sony realized the power of social media, and has gone to great lengths to integrate the social experience into its new handheld.

Sony isn't doing this with its own proprietary system, but by building in dedicated apps for platforms like Facebook and Twitter, allowing users to multitask and flip back and forth between game playing and networking. Humility is increasingly important in the video game perception space (in no small part due to social media!), so it's good to see Sony moving further in that direction under Hirai, as it recovers from missteps of the past.

Andy Rubin

/// GOOGLE

Android now controls a larger market share than any other smart phone platform, fragmented though that audience may be. Its continued growth is important for game developers as it expands to encompass tablets, set-top boxes, and more.

Rubin is SVP of mobile and head of the Android division at Google, and he has clearly laid out an aggressive timeline and plan for the platform, as its rapid explosion across non-Apple smart devices has been nothing short of astounding.

Tim Sweeney

/// EPIC

You can't really imagine the current generation of games without Unreal Engine 3. The toolset has become the new standard, even moreso than Criterion's RenderWare was back in the PS2 era. But what's been impressive this year has been on the smaller scale.

Technical director Sweeney and his team at Epic have pushed the engine down to smartphones, and now to browsers as well, in an attempt to truly capture the majority of the market. That's all well and good for Epic, but it also means that the browser space now has a more robust set of tools to work with, allowing bigger experiences to come to the most accessible game space in existence with even greater ease.

Marc Doyle

/// METACRITIC

Marc Doyle is the games editor at Metacritic, and he's responsible for selecting snippets of articles to display, as well as choosing which review sites to include and how each external ranking system is converted to Metacritic's scale.

No matter how you feel about Metacritic as an arbiter of quality, there's no question that review aggregation services have changed the way game businesses look at their titles. Hitting a certain Metacritic score "proves" a concept even if sales don't, or can even be a criteria for getting a bonus within a company. Doyle's work is subtle in its influence, but important nonetheless.

[EVANGELISM]

Jon-Paul Dyson

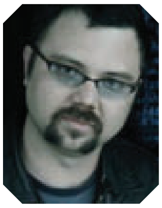
/// THE INTERNATIONAL CENTER FOR THE HISTORY OF ELECTRONIC GAMES

The International Center for the History of Electronic Games (a division of the Strong Museum of Play in Rochester, New York) is showing a lot of vision and dedication to preserving the history of our medium. Whereas most private collections and museums focus on amassing boxed product, the ICHEG goes beyond that to collect valuable paperwork, documentation, and ephemera.

Thanks to Dyson and his crew, priceless artifacts like Ralph Baer's handwritten notes, Will Wright's design documents, and the private collection of Sierra cofounders Ken and Roberta Williams will forever be accessible to historians. With a recent $500K grant and a 5,000 square foot play space, that collection is only the beginning.

Mike Acton

/// INSOMNIAC



Though he is also an advisor to this magazine, we simply must include Mike Acton here, for his site http://altdevblogaday.com. The site covers a wide variety of topics relevant to developers, from design postulates, to industry rants, to hardcore coding articles.

Acton is an incredibly well-connected developer, and the authors of these pieces come from a wide range of companies and backgrounds. This may be the future of connected game development media.

Antonin Scalia

/// U.S. SUPREME COURT

Supreme Court Justice Antonin Scalia wrote the majority opinion in Brown versus Entertainment Merchant's Association, which found that games are protected by the First Amendment. The ruling was also significant because it called out that research showing the negative effects of games was comparatively no better or worse than any other media.

Scalia has given the game industry valuable ammo against its detractors, and free speech protection is just another important step toward recognizing the art of games, as well as the more obvious entertainment level.

Danny Bilson

/// THQ

Bilson is one of the only CEOs in games who is still pushing for new original IP for core products. As Activision continues to (successfully) milk its established franchises, Bilson is trying to reinvent THQ as a hub for original core games. It's not a bad plan, considering core gamers are the folks that buy games even when nobody else does, and they'll support a company they approve of through thick and thin. Bilson and the THQ crew face an uphill battle, but original IP gives the game industry beacons of hope to point to, so it's worth the fight!

Keiji Inafune

/// INTERCEPT

Japanese game developers aren't often known for speaking out of turn, but Capcom veteran and MEGA MAN creator Keiji Inafune has been calling out what he feels are regressive business and design practices in the region.

Inafune quit Capcom after 23 years, and formed his own new studios, going on record saying that Japanese game development turns creators into salarymen, that men his age who now manage studios are holding creativity back, and how overbloated the staff sizes at major studios have become. We'll have to wait and see whether this will affect positive change, but revolutions have to start somewhere!

Brandon Boyer

/// IGF/VENUS PATROL

Brandon Boyer is the media king of the indie art scene, guiding the Independent Game Festival in 2012 (owned by UBM TechWeb Game Network, as is this magazine), as well as his own soon-to-launch consumer-facing site Venus Patrol (http://venuspatrol.com).

Boyer is a tastemaker in the indie game and art space, whose interests seem to mesh with a lot of other internet denizens, allowing his influence and praise to help launch careers. As Boyer starts to put together his newest media presence since his previous blog Offworld (using a Kickstarter that reached double its goal) we expect a reunification of the indie community around his banner.

Mike Morhaime

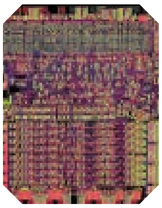
/// BLIZZARD

A little-known fact about Blizzard president and CEO Mike Morhaime is that he has used his success—both personally and professionally—to improve the lives of children. Through initiatives with the Make-A-Wish foundation, Blizzard has raised nearly $2 million over the past two years alone to help grant the wishes of children with life-threatening conditions, making Blizzard one of the organization's top contributors.

Morhaime doing this as a video game developer peripherally helps our industry as well, providing a counterpoint when mainstream media blames real-world violence on our art.

Greg James

/// THE VISUAL 6502 PROJECT (VISUAL6502.ORG)



The 6502 CPU powered the games many of us played in our youth. It ran home computers from Apple, Commodore, Acorn, and Atari. It was the main brains behind the Atari 2600 console, and was the core of the Nintendo Entertainment System. It is one of the most popular chips ever designed, and yet its schematics have been lost to time.

James has taken it upon himself to devise a method for preserving outdated computer chips on a microscopic level; stripping away the plastic, taking photographs, and re-creating every little trace in a virtual environment. It might not mean much to game development now, but his tireless work will ensure that we'll always be able to play these old games just as they were intended.

Mare Sheppard

/// METANET/THE DIFFERENCE ENGINE INITIATIVE

Diversity is a huge problem in our industry—one need only look at the comparative numbers of responses to our Salary Survey to see that. Women represent on average around 10 percent of game developers. And while a lot of us talk about the need for diversification, Sheppard's Difference Engine Initiative is actually doing something about it.

Rather than focus on advocating hiring of more women by video game companies, Metanet's Sheppard is attacking the problem at its core. She's making video game development interesting and feasible to a wider population by hosting free incubator programs in the Toronto area, where creative types can make games in six weeks without any programming knowledge. It's a small start, but we can only hope this is the beginning of a wider movement.

Jesse Schell

/// SCHELL GAMES

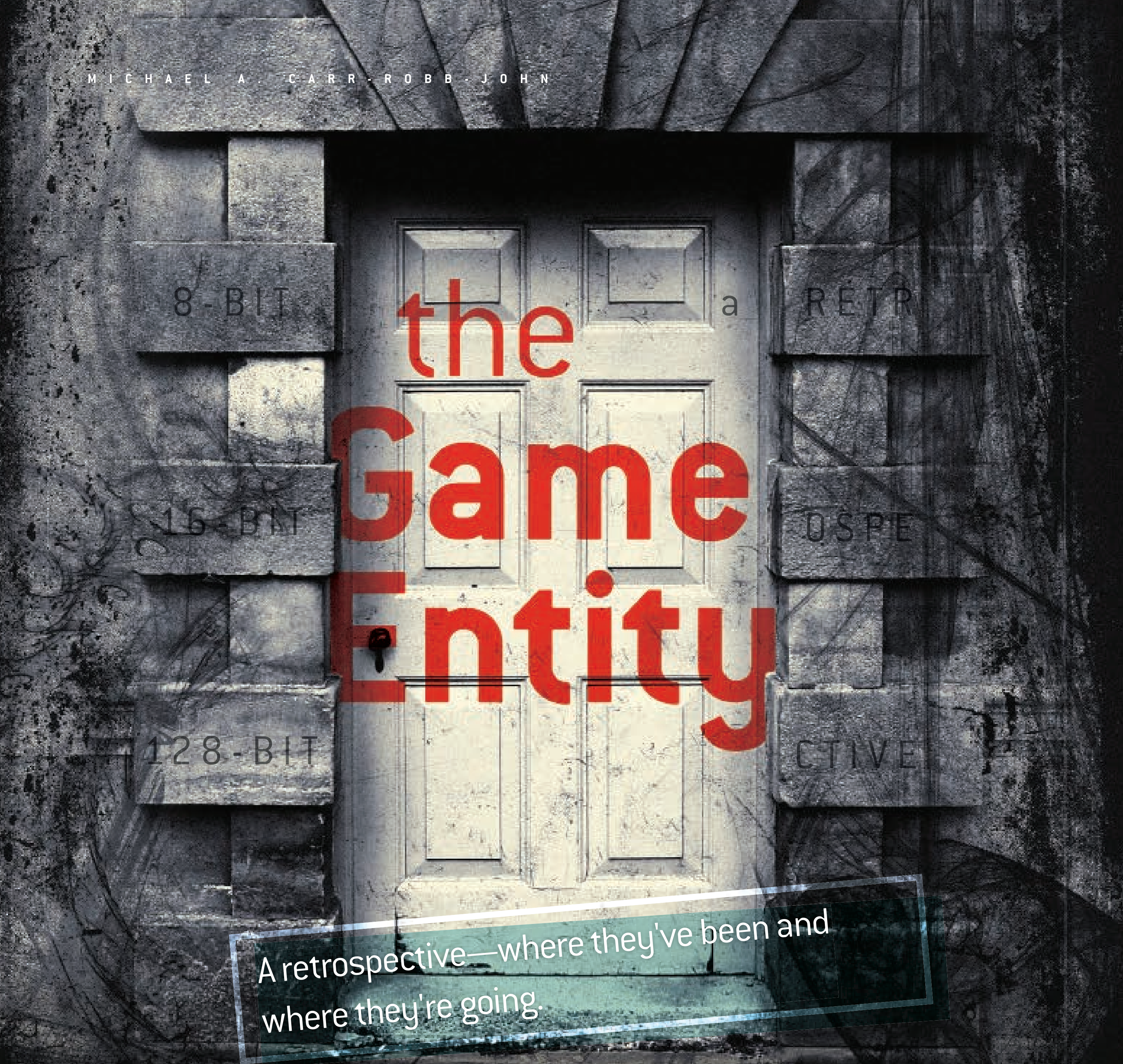
When it comes to advocating the positive powers of games Jesse Schell is a treasure. His closing keynote at the 2011 Games for Change event in New York was a beautiful reminder of what we should all be aiming for. It is through online games, he argues, that we can strip away concepts like race and gender and social status and be free to be our real selves.

Schell argues that games—even violent games—can bring about peace and resolve social problems, and he elucidates this in a way that everyone thinks, but can't quite express.

BRANDON SHEFFIELD and **FRANK CIFALDI** *are editor-in-chief of* Game Developer *and news editor of* Gamasutra *respectively .*

2003 Tony Hawk's Underground
2004 Tony Hawk's Underground 2
2005 Tony Hawk's American Wasteland
2005 Gun
2006 Tony Hawk's Project 8
2007 Tony Hawk's Proving Ground
2007 Guitar Hero III: Legends of Rock
2008 Guitar Hero: Aerosmith
2008 Guitar Hero World Tour
2009 Guitar Hero: Metallica
2009 Guitar Hero 5
2009 Band Hero
1996 Skeleton Warriors
1998 Apocalypse
2004 Tony Hawk's Underground 2
2005 Tony Hawk's American Wasteland
2005 Gun
2006 Tony Hawk's Project 8
2007 Guitar Hero III: Legends of Rock
2008 Guitar Hero: Aerosmith
2008 Guitar Hero World Tour
2009 Guitar Hero: Metallica
2009 Guitar Hero 5
2009 Band Hero
1996 Skeleton Warriors
1998 Apocalypse
1999 Tony Hawk's Pro Skater
2000 Spider-Man
2000 Tony Hawk's Pro Skater 2
2001 Tony Hawk's Pro Skater 3
2002 Tony Hawk's Pro Skater 4
2003 Tony Hawk's Underground
2004 Tony Hawk's Underground 2
2005 Tony Hawk's American Wasteland
2005 Gun
2009 Guitar Hero: Metallica
2009 Guitar Hero 5
2009 Band Hero
1996 Skeleton Warriors
1998 Apocalypse
1999 Tony Hawk's Pro Skater
2000 Spider-Man
2000 Tony Hawk's Pro Skater 2
2001 Tony Hawk's Pro Skater 3
2002 Tony Hawk's Pro Skater 4
2003 Tony Hawk's Underground
2004 Tony Hawk's Underground 2
2005 Tony Hawk's American Wasteland
2005 Gun
2008 Guitar Hero World Tour
2009 Guitar Hero: Metallica
2009 Guitar Hero 5
2009 Band Hero
1996 Skeleton Warriors
1998 Apocalypse
1999 Tony Hawk's Pro Skater
2000 Spider-Man
2000 Tony Hawk's Pro Skater 2
2001 Tony Hawk's Pro Skater 3
2002 Tony Hawk's Pro Skater 4
2003 Tony Hawk's Underground
2004 Tony Hawk's Underground 2
2005 Tony Hawk's American Wasteland
2005 Gun
2006 Tony Hawk's Project 8
2007 Tony Hawk's Proving Ground
2007 Guitar Hero III: Legends of Rock
2008 Guitar Hero: Aerosmith
2008 Guitar Hero World Tour
2009 Guitar Hero: Metallica
2009 Guitar Hero 5
2009 Band Hero
NEVERSOFT
NOW HIRING
WWW.NEVERSOFT.COM

MICHAEL A. CARR-ROBB-JOHN



the Game Entity

A retrospective—where they've been and where they're going.

I was recently reminded of a conversation I had some twenty-odd years ago. A fresh-faced and innocent programmer (yours truly) was discussing with his fellow engineers the future of CPU technology. At the time we were programming on 8-bit processors (6502 and Z80 CPUs). The 16-bit processor (68000) was in our sights and we joked about the day when we would enjoy—128-bit registers!

While we knocked around ideas of what we could do with this ocean of unlimited memory, our attention did for a time focus on what we could do with game entities. We thought there would be such large numbers of processors that we could run each individual game entity on a single CPU, and would no longer need to give up processing cycles to rendering and audio. We could dedicate all CPU time to game A.I. and game mechanics! This was in a time when 5–10% of the CPU for game A.I. was the most you could hope for. We figured a few things would be built into the hardware, such as path generation (such as A*), state machines, physics simulation, or random number generation.

Coming to the present day, 128-bit registers are here, but we rarely program in assembly anymore. We have our ocean of memory, but no matter how much you have it's never enough (I write this while looking at a 1024x1024 32-bit eyeball texture). Advancements in rendering and audio hardware have shifted a lot of work off the CPU, granting more time to actually process A.I. and game mechanics, but unfortunately the thousands of CPUs we hoped for really haven't materialized. Some attempts have been made at physics hardware, but that never gained enough critical mass to be worth supporting, and it certainly isn't being built into the console or mobile market.

This reminiscence got me thinking about our most fundamental game system—the game entity—and its supporting architecture. In the past twenty years we have seen some major jumps forward in almost all areas of game technology, but has the game entity kept up? The last major evolutionary step forward that comes to mind was components, which came along a little over ten years ago. What major improvements have happened since then? Have components really solved all the problems they set out to tackle? Is it possible that the game entity is perfect and there are no more major evolutions waiting to be made?

Inquiring minds want to know!

I set out to discover the current state of the game entity in the industry today, how it lives, and how we're using and abusing it. And very importantly, what can't it do? Maybe by the end of this article we'll have a clearer idea of where we are, what's missing, and a glimpse at what might be on the horizon.

THE LIFE CYCLE

You are born and then you die. The bit in between is called life! Pretty much sums up most things: bumble bees, dolphins, developers, dragons, and game entities. Putting components aside for a moment, these are the stages of life that game entities go through.

Construction. A blank slate, everything is cleared, default values set, pointers cleared to null.

I have encountered some architectures that have banned the use of constructors and destructors in game entities! Usually this is because of a misunderstanding of how C++ works and is usually backed up by the fact that some engineers just can't stop themselves from doing memory allocations and other nonsense in the constructor.

Init. Usually some kind of configuration data is passed into this stage (often loaded from the level), indicating how this entity should configure itself. This includes things like its position in the world, its physical properties, what visual assets to use, what audio assets to work with, how much health it has, and so forth.

Resolve. Once initialization of all entities has taken place, this "second" initialization stage allows us to setup inter-entity relationships, sometimes referred to as the "First Frame Process." Examples include a door's connection to the navigation system, an elevator to its trigger regions, Characters to possessions, and the like.

Update. This is the entry point through which all the cool things happen. Go crazy, have fun, make awesome things!

It's surprising how often I have encountered an architecture that supports the Resolve stage, but into which engineers have still implemented something like this at the top of the Update function:

```
if ( m_FirstFrame )
{
   M_FirstFrame = false;
   // Do something interesting that I
couldn´t do in the resolve stage.
}
```

The reason for doing this comes down to a simple logic problem. Entity A can't finish resolving itself until Entity B has resolved! There are several ways this could be solved—one method would be to introduce a processing priority so that Entity B is always processed before Entity A. Another method would be to split the Resolve stage into Pre-Resolve and Post-Resolve steps.

On more than forty projects, I have never seen either of these possible solutions implemented, and for two very good reasons. First, there are usually only one or two game entities that require this "fix" (or "hack," depending upon your perspective) and secondly even these specific fixes have scenarios where they would not be a good solution.

Draw. The draw stage is called prior to rendering of the entity, usually to allow the entity to configure the renderer with the latest information.

Deinit. It is at this point that any assets associated with this entity are released—and just to be clear this does not necessarily mean that the assets get removed from memory (usually they don't). Asset management is another interesting subject for another time.

Now we clean up and prepare for deconstruction.

Destruction. The deconstructor should really just be a checklist to make sure that before the entity is deleted nothing has been left behind—that is to say nothing was incorrectly deinited. Even the simplest of checks in debug mode, such as scanning the object's memory to make sure everything is zero, are invaluable at catching issues when something has been forgotten.

These are the seven stages of a game entity's life, though a few systems have additional pre- or post-functionality for the update and render stages. Generally everything here will exist in most systems being used today.

THE PROCESSING ARCHITECTURE

How we manage our entities can have a huge impact upon the CPU, and this is always an area that can be tailored to the specific game genre for optimal performance. Below I have detailed some of the common core processing systems I have used in the past.

Basic frame processing. Usually this is the first implementation of an entity system that we write when we are first learning how to make games, it's quick, it's dirty, it's simple, and you would be surprised by how many games published today use this as their processing architecture. Not that there is anything wrong with this. If you have less than a couple hundred objects and are not trying to push any limits then you don't need to do anything more complicated.

The game loop pretty much goes like this:

```
Update Entities
Render
...
```

The biggest problem with this is it locks the render and update stages into sync with each other. If you wanted to maintain 60 fps in your game this would mean that you would have to update every entity and render them all in 1/60th of a second. A couple hundred entities might work, but if you have several thousand, you have a problem, even if half of those objects are invisible (triggers, markers, barriers, and so forth).

Time splice processing. Time splicing has been used to great effect on a number of games. Essentially it spreads the processing over a number of frames. The game loop would look something like this:

```
Update 25% of entities
Render
...
```

This essentially means that the game update is running at 1/4 of the speed of rendering. Using 60 fps rendering, that would yield a 15 fps update. A variation to this system is to specify how much CPU time is available for processing each frame—the system would start processing entities until it ran out of time, then would pause until the next frame.

It's worth mentioning that this processing system and the following systems are not possible if you do not have a rendering system that is capable of running independently of the game entity update. By this I mean it must be capable of interpolating position, rotation, and animation between the entity updates.

Variable frame processing. Not every game entity needs to run at the same speed. For example,

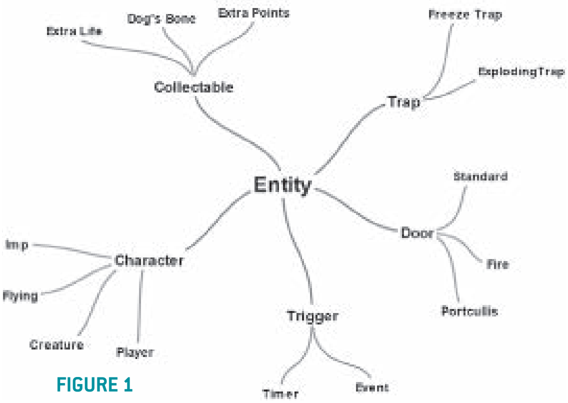


FIGURE 1

we might want to run the player at 60 fps, but do we really need that magical book that randomly flies up and down a corridor to also run at 60 fps? In all likelihood it could run at 10 fps with no issues. Variable frame processing allows you to indicate just how fast each game entity should update.

One major issue to watch out for when using this processing system is that it needs to auto-balance itself! Let's say we gave all game entities an update of 20 fps. It would be rather pointless if they all triggered at the same time, on the same frame. That really would make a lovely spike in the CPU graph.

Just to give you a sense of how powerful and useful this can be at spreading the entity processing load, the imps in Dungeon Keeper 2 were processed at 4 fps.

Level of detail processing. LOD processing allows the game entity to dynamically calculate its own fps based upon how far away it is from the focus of attention (usually the camera's position). A bird flying 500 meters away might only need to be updated at 1 fps, unless it happens to be within 50 meters of the camera, in which case it would ramp up as it got closer to a maximum of 20 fps, let's say.

Obviously this method doesn't work particularly well if all the game objects are clustered together in a small area.

Scene processing. Scene management isn't just for rendering—it can also be used quite dramatically to control CPU usage, especially when combined with variable frame processing. How you decided to manage your scene can also make a huge difference. You can use an octree, visual cells, rooms, zones, arenas, or myriad other options.

For all these different methods of processing game entities, I have never seen a priority system implemented. In fact, the closest I have seen to an implementation would simply be to switch two entities around in the update link list. Effective I suppose, but extremely clunky.

COMPONENTS

Just over twelve years ago I was putting together additional content for one of my previous games. The content included a new trap that you created like a trap, placed like a trap, and interacted with like a trap, but when it was sprung it would jump off the ground and move around behaving like a creature, before eventually turning into a trap again. It sounded great when the designer walked me through the idea, but the reality of actually implementing it hit as soon as I sat down to write it. The first issue was where to place it in the entity tree structure.

At the time the game used a traditional game entity structure similar to the one in Figure 1.

All the trap functionality of the entity obviously existed in the Trap branch, and all the creature functionality like navigation, piloting, and A.I. behaviors all existed off the Character branch.

Where you place it in the tree also has other knock-on issues to other areas of importance, such as the GUI and A.I. systems.

In the end since the object was for all intents and purposes a trap, I attached it coming off the Trap Entity. This meant I then had thousands of lines of code in the creature that I either had to relocate up the entity tree to the base entity, or use the infamous cut-and-paste (with all the issues associated with that).

This work wasn't eloquent, wasn't clever, was incredibly error prone, caused code bloat, and took triple the amount of time it should have. The only good to come out of it was a very cool trap for the player and a firm conviction on my part that there had to be a better way of doing things.

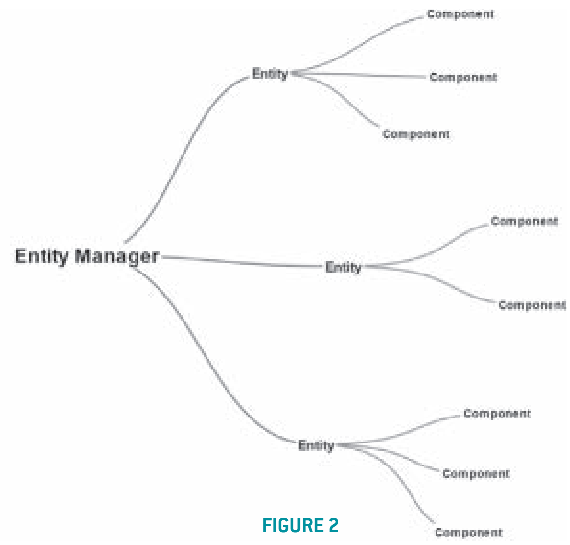


FIGURE 2

The Component Evolution. Components have a number of major features that traditional game entities don't.

- **Self-contained**. Functionality is isolated into manageable containers.
- **Reusable**. No more cut-and-paste. If you want an entity to have an ability, you just need to attach it, and depending upon your system, that can be as simple as a tick-box or an extra line in a configuration file.
- **Resources**. The entity only uses the resources (CPU and memory) it requires.

Data-driven components. The purest form of components is the data-driven version, it uses configuration data to assemble the entity at runtime, for example:

```
<Entity Type>
   <Name> Toggle Switch </Name>
   <Component List>
      <Component> Render Animation </
Component>
      <Component> Physics </
Component>
      <Component> Interactive </
Component>
      <Component> Two Stage Switch
Logic </Component>
   </Component List>
</Entity Type>
```

What is really interesting about this system is that because an entity is constructed at runtime, it's also possible to modify an entity while the game is running.

Under this model the game entity is essentially stripped of its functionality and data, making it for all intents and purposes a blank container. What is left of the game entity is pretty minimal:

```
Name (String)
Unique Id
World Transform
List of attached components.
```

Components written under this model must be bulletproof, and because of that you tend to get more stable and robust code being written (always a plus). The only negative is the slightly higher memory and CPU overhead of handling components.

Plug-in Components. Sometimes the overhead of working with a data-driven model is too high, especially if you are developing for a platform that has limitations (at least when compared to the Xbox 360, PS3, or PC) such as Nintendo's DS, Sony's PSP, even the Wii. The plug-in model offers a way to still have components and tap into their benefits while removing some of the overhead of the data-driven model.

With plug-ins you no longer create entities at runtime. Instead, they are usually declared in code and built at compile time, for example:

```
Class CToggleSwitch : public CEntity
{
public:
   EntityDeclare( CToggleSwitch,
CEntity, eEntityToggleSwitch );

   //- Constructor / Destructor -----
-------------------------------------
--------------------
          CToggleSwitch( void );
virtual  ~CToggleSwitch( void );

private:
   //- Plugins ----------------------
-------------------------------------
--------------------
   EntityUsePlugin( CToggleSwitch,
CPluginRender, m_pRender );
   EntityUsePlugin( CToggleSwitch,
CPluginPhysics, m_pPhysics );
   EntityUsePlugin( CToggleSwitch,
CPluginInteractive, m_pInteractive );
   EntityUsePlugin( CToggleSwtich,
CPluginTwoStageSwitchLogic, m_
pToggleSwitch );
};
```

Obviously this means that the plug-in model has an entity tree structure, but it is generally flat. There is nothing stopping you from creating a more complex structure, but personally I have always steered clear of that choice, mainly to make sure I never get into the same scenario I was in years ago.

When Components Go Bad. Granularity is the biggest pitfall when working with components. I've seen engineers take the entire contents of their game entity and just dump it into a single component. They often say that because the new component requires all the functionality, it made sense to include it all. One engineer even did it so he wouldn't have to worry about any intercomponent interactions.

Making components that are so large that they can't be reused by other entities is one extreme of the scale, and the opposite end is of course when components encapsulate such small functionality that the overhead of managing and processing them becomes greater than the cost of the features they provide.

A more insidious issue with components is circular dependencies. As a general rule data should only flow one way, from higher levels to lower levels. You do not, for example, want your physics component talking directly to your A.I. brain.

Component-to-Component Interactions.

The simplest method of interaction from one component to another is to retrieve a pointer to the required component and direct call functions. The problem here is you have just made a hard link between the calling component and the target component.

If you want to avoid this, there are two methods available. The first is by way of an event (or message), sent to the components to either perform a specific task or return some information. The components usually handle this event immediately, allowing access to component functionality without necessarily creating a hard link between components.

The second method is the mailbox, where a message is posted to the component and stored for when the component is ready to process it. This is usually during its next update phase, and gives the ability to completely isolate one component from everything else. The negative side is that it's a pain to debug, and you will start to get latency issues since it can take several frames to handle a message.

The Component Architecture. Under the traditional entity system, calling update on an entity meant everything relating to that entity was processed. One of the benefits of isolating functionality into discrete components is that we no longer need to update everything in one go. We could for example have our rendering components updated during the render phase, the physics components updated straight after the simulation step, and so forth.

I have actually never needed to do more than the basic components like rendering, physics, and audio. That doesn't mean it can't be exploited and used if the game would benefit from it.

GAME SYSTEMS

You might call them systems or managers—some call them modules, while others prefer singletons. Whatever your pattern of choice, at their core they are all the same; they wrap up a mechanism or section of functionality into a self contained handy package.

The Entity Manager. When you have a collection of game entities, you are guaranteed that somewhere there is an entity manager lurking in the background, creating them, maintaining them, handling their delayed deletion, keeping statistical data for profiling, occasionally prodding an entity when it misbehaves, shouting at the youngsters to keep up, and trying to keep the world in harmony.

From the manager's perspective, it believes that it has ownership over the entities. As far as it is concerned no other system touches them, except through the normal entity / component update. The Entity Manager sees the entity / component connections as demonstrated in Figure 2.

The Other Managers. What the entity manager doesn't know is that there are a large number of game systems that are not only talking directly to the entities, but also passing references, compiling them into lists, and making major changes to the entities itself, all outside the normal entity update. Some of these systems might include Player Manager, Squad Manager, GUI Manager, Cut-scene Manager, or Task Manager. In your average game there are probably more managers that touch entities than those that don't. So what are these managers doing?

Let's consider the Player Manager. It stores a list of game controllers connected to the hardware, and which controller is controlling which entity. The GUI system might ask the Player Manager which entity "player 1" is controlling so it can display the correct HUD information, or an enemy entity might ask which entity the player is so that it can focus its attention on that. Having the manager store a reference to the player-controlled entities means it can quickly provide the information upon request.

A Squad Manager might contain all the information about each squad in the world, which entities are in each squad and each squad's alliances, and so forth. It might also handle the higher level thinking required to achieve the objective or goal. Once it has planned the required steps, it instructs the members of the squad (the entities) to go and do what is necessary. The squad manager might also contain references to other important entities that it needs to track in order to carry out and execute its plan, for example kill this character, retrieve that object, move this object from here to here, activate that switch, or open this door.

The point I'm trying to make is that there are plenty of managers regularly interfering with game entities before, during, and after the update frame, usually without much constraint or control other than mechanisms and interfaces that might have been built into the entity.

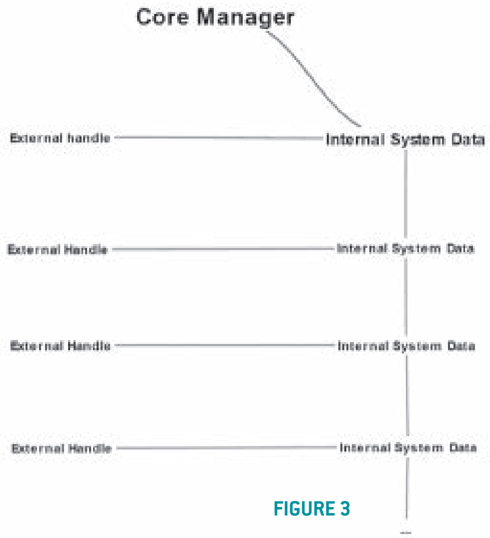


FIGURE 3

And managers are not the only ones that reference entities. Entities do it themselves as well. If we were to rebuild the diagram in Figure 2 with all the real connections that are going on we end up with something like Figure 3.

Connections. How we make these connections to an entity can be done in a variety of ways. The most basic one is a pointer to the entity or component. Obviously the biggest issue with this is if the entity itself is deleted, you suddenly have a dangling pointer with no way of validating before use. Although I don't recommend this, you would be surprised how many games have shipped using this method with very few issues (mainly because of some good interface design and a fair amount of liquid pixie dust, more commonly known as luck). Other methods that have been employed with success have been Weak Pointers or Smart Pointers. On one project I even got to use Unique IDs, which was great, but costly in terms of doing the lookup of Id to pointer. The only reason I say "great" is because it made a difference to me at 2 AM in the morning when looking up variables that read within a human context, i.e. entity 15 is targeting entity 89, instead of 0x5feabeda targeting entity 0xa3922ef1. I used GUIDS once, and I hope never to experience that ever again.

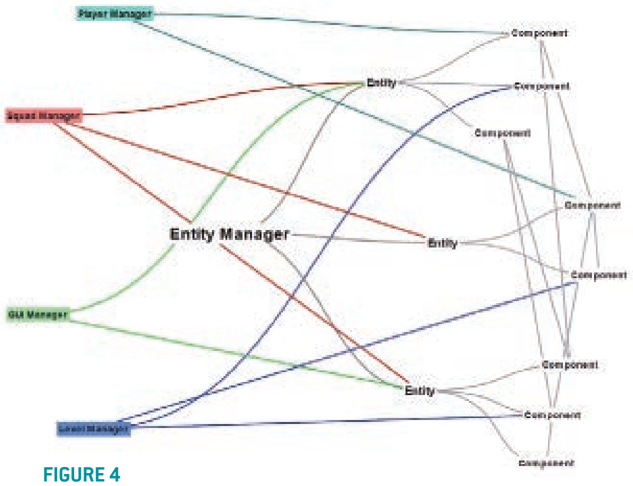


FIGURE 4

The Core Managers. Let's consider for a moment three of my favorite core managers, Rendering, Physics and Audio. These three have evolved from very early on to take advantage of multithreading. Part of that evolution has been out of necessity, to isolate the internal low level data and force any external interactions through an interface or handler.

You won't get the player entity tweaking the audio sample data as it's being played, and you don't have entities adjusting their vertex data either. This is not just because that data might be shared, but because this is a requirement when protecting data against multithreaded issues. See Figure 4.

FUTURE PONDERINGS

What conclusions are there in regard to our humble game entity and where it might be heading in the future?

Entities. The game entity is very much alive and kicking. It is an evolving and still-fundamental part of most games in development today.

As much as we all want to move away from the traditional hierarchical tree structure, it still has its uses, usually when developing on machines that don't have enough processing power or memory to allow the use of components. In some cases, the traditional structure is used because the game doesn't require anything more complex.

Components. Since components have come along, the game entity has lost its grip on the data and functionality it once commanded, and is now reduced to a container, or the glue that binds components together.

I remember a few years ago trying to sell the idea of components to someone with promises that level designers would be able to put game entities together without the need to take up valuable programming time. Unfortunately, my experience to date has not reflected this, and engineers are still building the entities for games, albeit under a system that is a lot more flexible. The only exception and pleasant surprise has been when prototyping with Unity3D I have witnessed level designers and even artists building entities.

Components evolved out of the issues we had with the bloated game entity, so it hardly seems surprising that the life cycle and processing architectures have not really changed that much from their ancestor. I suspect that as components continue to evolve their architecture will adapt to the component's requirements and not the entities they are attached to.

Some issues still haven't been resolved. Circular dependencies and processing priorities are just as much an issue under components as they were under entities. In the past we solved them on an individual basis rather than with a major change to the core system. This seems to have carried over to components as well.

Ideas have been knocked around suggesting that components could be placed into processing groups. There are some interesting possibilities here, not least of which is a potential solution to the inter-component dependency issue where component A must be updated before component B.

At various points during the game (level load, scripted event, et cetera) entities are created, components are attached and initialized, and then they are left to run within the world. At some point the entities are not needed anymore, and they are then deinitialized and deleted. This is common for so many projects you can almost see that we are still thinking in the traditional game entity way. What seems to be slipping people's minds is that data-driven components are dynamic. They can be disabled, enabled and deleted, created, or even detached from one entity and attached to another. The idea that a game entity could be radically morphed during gameplay from one thing into another is an area that I don't believe we have even begun to explore yet (designers take note).

MULTITHREADING

The potential gains with a multithreaded game entity or component system is obviously enormous, and just the thought of those extra cycles certainly has a fair few game engineers excited, me included. The hardware is going to keep getting better, so the sooner we start to take advantage of it, the better we will be placed in the future when more cores become available. My main question is why hasn't it happened yet?

Architecture. In order to become multithreaded we need to change more than just the entity and component system. We need to look at all the supporting game systems as well. Of particular importance is putting under scrutiny how the various systems interact and communicate with entities.

Unity3D makes all game system code components. Maybe this is the way we need to go? It certainly simplifies the problem down to a component-to-component issue.

Refactoring. I could see it being a bit of a nightmare to retrofit a current component system to be multithreaded. It would certainly be far easier and faster to implement from scratch.

Memory. No matter how much memory you have, it's never enough! Splitting a component update into multiple stages (read, execute, and write) creates additional requirements on the amount of memory entities consume.

Educate. When we moved from traditional game entities to component-based entities, a fundamental shift in thinking was required. I suspect the shift required when adapting to multithreaded components is going to be larger.

The shift from linear entity processing to multi-threaded processing is not going to be a quick overnight transition. I suspect that we will convert first the easiest components, then expand upon that slowly bringing the other components into the fold. Now that I have said that, someone will probably prove me wrong next week.

MICHAEL A. CARR-ROBB-JOHN *has been writing games for more than 20 years, and most recently worked at KMM Games in Brisbane.*

PHIL WATTENBARGER

crimson alliance

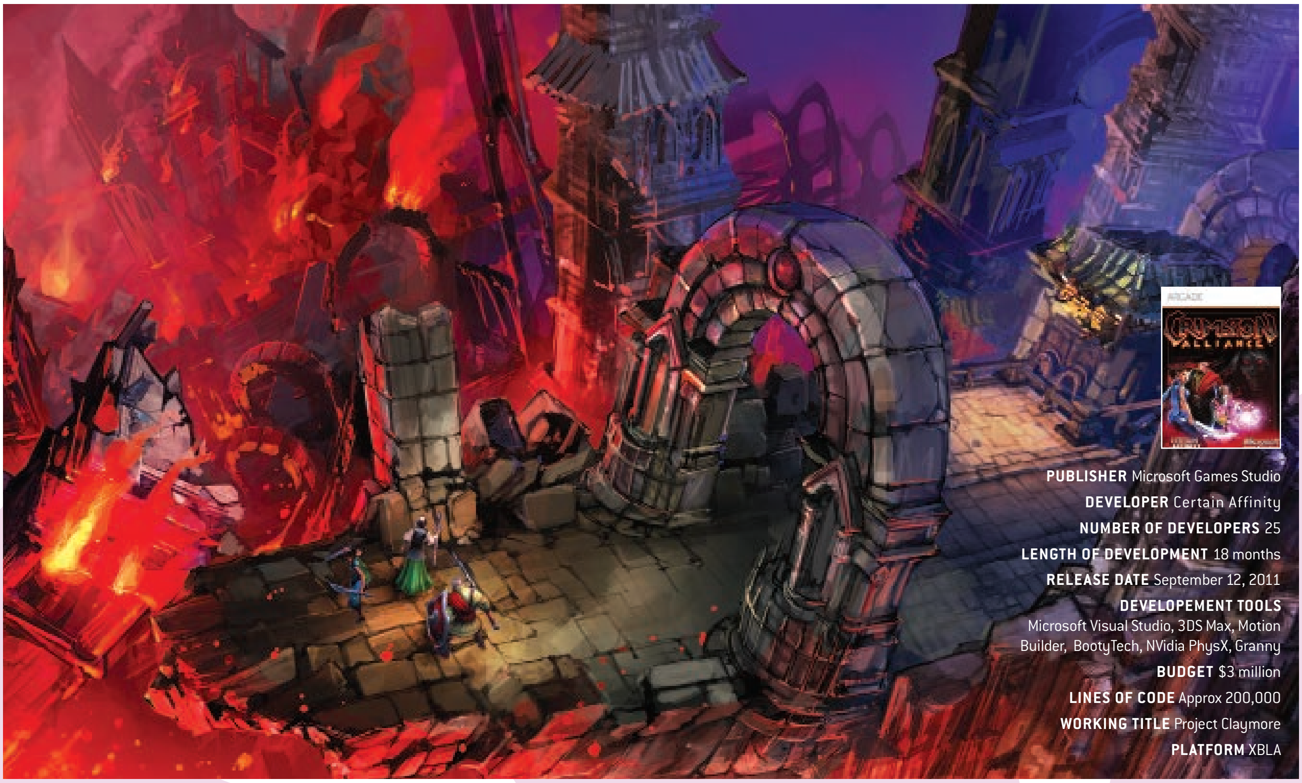
/// THIS YEAR MARKS CERTAIN AFFINITY'S FIFTH BIRTHDAY, AND IT WOULD TAKE A NOVELLA TO COVER ALL THAT HAS HAPPENED SINCE OUR INCEPTION. FROM HUMBLE BEGINNINGS IN A SMALL HOUSE IN CENTRAL AUSTIN, CERTAIN AFFINITY HAS SINCE WORKED ON HALO 2'S BLASTACULAR MAP PACK, AGE OF BOOTY, CALL OF DUTY: WORLD AT WAR, CALL OF DUTY: BLACK OPS, LEFT 4 DEAD, HALO WAYPOINT, HALO REACH'S DEFIANT MAP PACK, AND THE UPCOMING HALO: COMBAT EVOLVED ANNIVERSARY. ALONG THE WAY THE STUDIO HAS GROWN TO ALMOST 70 PROFESSIONALS WORKING ON TWO TO THREE PROJECTS AT ANY GIVEN POINT IN TIME.

AFTER CREATING AGE OF BOOTY, WE FELL IN LOVE WITH THE XBOX LIVE ARCADE SPACE, AND VOWED TO RETURN WITH ANOTHER ORIGINAL TITLE. WE CONTINUED DEVELOPING OUR PROPRIETARY ENGINE AND TOOLSET, LOVINGLY NICKNAMED BOOTYTECH, AND SET OUT TO BUILD A BADASS, FOUR-PLAYER COOPERATIVE ACTION RPG CALLED CRIMSON ALLIANCE.

BUILDING AN ACTION RPG THAT HARKENS TO THE ACTION GAMES AND RPGS OF OUR YOUTH WAS A LABOR OF LOVE. WE LEARNED A THING OR TWO FROM THE DEVELOPMENT OF AGE OF BOOTY, AND OUR WORK ON LEFT 4 DEAD, AND WERE SET TO KNOCK THIS PROJECT OUT OF THE PARK. BUT SOME LESSONS YOU GET TO LEARN OVER AND OVER, AND CRIMSON ALLIANCE WAS HAPPY TO RE-TEACH US ALL ABOUT THE CHALLENGES OF CREATING A NEW IP, AND THE PERILS OF BUILDING A TEAM, A GAME ENGINE, AND TOOLS WHILE ALSO TRYING TO SHIP A GAME. WE ALSO LEARNED A FEW NEW LESSONS ABOUT HOW CAREFUL YOU HAVE TO BE WHEN IT COMES TO SETTING PLAYER EXPECTATIONS, AND THE EXTRA SENSITIVITY SURROUNDING ANY SORT OF IN-GAME MICROTRANSACTIONS.

>>>

crimson alliance



PUBLISHER Microsoft Games Studio
DEVELOPER Certain Affinity
NUMBER OF DEVELOPERS 25
LENGTH OF DEVELOPMENT 18 months
RELEASE DATE September 12, 2011
DEVELOPEMENT TOOLS Microsoft Visual Studio, 3DS Max, Motion Builder, BootyTech, NVidia PhysX, Granny
BUDGET $3 million
LINES OF CODE Approx 200,000
WORKING TITLE Project Claymore
PLATFORM XBLA

WHAT WENT RIGHT

1/ focus on action first

We knew from the outset that the game needed to focus on action first. Particularly in the fantasy-RPG space, there is a rich heritage of games that have given the player incredible depth in inventory management, character creation, and dialog, but this often comes at the expense of combat action or multiplayer co-op accessibility. Because cooperative play and fast, visceral combat were central to our design, we often had to stray from the well-worn path to find different solutions. Letting one player hold up the action while they rummage around in their backpack or read through a dozen descriptions for different skills would have slowed down the pace of the cooperative game. So we elected to make design calls based around the goal of keeping the action going at all times, even when it meant sacrificing some features our audience expected from the genre.

Specifically, this focus on action led to key decisions around the item system, puzzle design, and NPC interaction that helped keep the game fast paced and accessible, not to mention within scope. We were often tempted to return to convention, but in the end, the emphasis on action kept us focused.

As examples, we were able to get our core auto-aiming system locked down fairly early, and made few changes to it over time. Player character movement speeds, the unique role of dash in the game, the camera position and location, and the general ranges at which combat could occur were critical pieces to the puzzle, and we did a good job of keeping focus on these until they felt just right, and then didn't tweak them much thereafter. Our systems designer, our gameplay programmer, and our animator all worked closely together tweaking speeds, number of frames per animation, and blending, until everyone enjoyed slaying monsters in a testbed.

Borrowing from much of our expertise building first-person shooters, we knew that tactical encounters were important. Armed with this experience, our design team prototyped encounters early that helped us understand where the fun came from in the tactical combat. This took many iterations, and at times lots of painful rework. Occasionally the team wondered if we were being too critical, but once it came time to enter production and truly build a collection of levels, we had a foundation of game systems and proven encounter templates to support that work.

In the end, knowing how much of the game was to be about action and how much about RPG helped us build a hybrid game that knew what it wanted to be, and could bring in the right elements of both genres without getting tangled up in complexity.

2/ it's not done until it's fun

Over the past few years, XBLA and PSN have blossomed into high quality platforms for delivering new IP. Games like Shadow Complex, Bastion, and Toy Soldiers have delivered near-retail-quality experiences at a fraction of the cost, though that cost has been steadily rising. It was clear to us from the start that Crimson Alliance could not be "just an Arcade title." Despite being our first RPG, it needed to breathe life into the action RPG genre. It's probably no surprise that in the end this commitment caused us to devote additional time and money to hit our quality bar.

The mantra of "It's not done until it's fun" served as a beacon during stormy times, helping guide us to the right decisions, even when they were challenging to make. For example, at one point players were given control of a hulking Ogre character capable of smashing everything in sight. Everyone loved the concept and could see the potential. However, despite investing a lot of time in the encounter, there were too many things that weren't coming together. Control wasn't up to the standard of the main characters, the co-op mechanics were iffy, the encounter lacked a proper conclusion, and the mechanism by which players assumed control of the Ogre was vague at best. After a long hard look, the feature was ultimately cut because it just didn't meet the quality bar we had established. It was a blow to the team, but looking through the lens of quality allowed people to ultimately come to terms with the decision.

Throughout development we cut, deleted, restarted, rebooted, and re-imagined tons of features in the name of quality. The only thing worse

than cutting something you've worked hard on is shipping something that isn't high quality. This sort of devotion to a standard is a hard thing for a team to always steer by, and it took some adjustment. By the end of the project, though, everyone had done an excellent job of putting themselves in the shoes of the player, and the game ended up stronger as a result.

3/ open collaboration

At Certain Affinity we all sit in one big open room—"the pit." We've always sought to break down hierarchies and maintain an open, collaborative atmosphere as another of our core company values; everyone from the president on down sit together in this space. Distinctions between teams are often based on which side of the room your desk is located.

The open floorplan is ideal for collaboration. To know about the game, you just look around. You can't help but overhear comments about a new system or feature you implemented. It requires more effort to remain isolated than it does to swivel your chair around and join the conversation.

We've learned over time that the only way to ensure a game is fun is to relentlessly playtest it; every damn day no matter how sick of a level you may be. We tried hard to maintain this practice on Crimson Alliance. The entire team engaged in playtests, and everyone was encouraged to share their thoughts in an open forum (even when the feedback was critical). We captured feedback, sent it out to the team as a whole, and then leads extracted actionable items. Particularly effective sessions had the designers and scripters in charge of the level hovering nearby, making observations and taking furious notes. It was absolutely critical that we encouraged everyone to share their ideas, but perhaps more importantly, everyone understood that sharing feedback, even from the very top, didn't necessarily mean that the feedback would become actionable. For us, collaboration meant that everyone had a voice, and that we tried to always give the greatest weight to whichever opinion had the greatest merit, not just whoever was the loudest or the most senior.

4/ extending timeline for polish

It's one thing to say that "quality is king." It's much harder to put your money where your mouth is. We hit an early vertical slice milestone and convinced ourselves we were ready to enter production. The schedule was partly to blame here. Within two months we knew that we weren't ready yet after all, so we added a full three months to the schedule. Then, later in development, we added several more months to the polish phase in order to give ourselves a few additional critical features and to fix a lot of critical bugs.

This is the kind of decision that companies like Blizzard or Valve can make, but for a small, self-funded studio like ours, it's one of the hardest calls to make. We believe now that this was the right choice, because the quality of the game is much higher as a result. But at the time there was constant pressure from both our schedule and wallet to call things "good enough."

In the end, we were able to resist the pressure long enough to reach our high quality bar in many areas, though not in everything. We feel great about the way the core game mechanics play out on a moment-to-moment basis, but as with every game, there are things we just weren't able to get to. We'd love to have improved matchmaking, given players a fourth class to play, and so on. There's no doubt that the RPG systems could have used a bit more time as well, though making them deeper always bumped up against the philosophical focus on action. As you might imagine, we're eager to dive into our next offering in the action-RPG space; we hope that the success of Crimson Alliance can buy us even more leeway to remain "quality focused" in the future.

5/ give 'em what they expect visually, and they'll love it. follow the 90/10 rule

We initially explored a variety of different art styles and sensibilities for the game, ranging from a more painterly style to some that felt more technical and realistic. Ultimately our palettes and slightly painted look ended up being a success. Particularly once we added a crack visual effects programmer to the team, our world began to come alive with an art style that felt familiar, and which could draw comparisons to retail games with far bigger budgets. Adding environmental particles, mists in the background, and over-the-top visual effects from fireballs to blood splatters ended up making the feel very engaging world in motion. The result is visuals that have drawn a lot of praise, and which gamers immediately understand. The dripping dungeons, dusty desert, creepy castles, and even the lovely lava from our upcoming Vengeance DLC Pack end up playing very much into the themes of traditional fantasy gaming.

By building a world that is familiar, rendered in a style that is evocative of other games in the space, we felt that players would feel an immediate connection to the game. They can imagine themselves and their friends battling through these spaces, because they've done it before. By relying heavily on particle effects, motion blur, and other advanced VFX tricks to help the world come alive, we were able to keep the game feeling "fast" and reinforce the action element. (Notice the exaggerated burning fuse on the barrels as an example of this.)

In our cutscenes we made a conscious effort to steer toward more mature themes with an art style that would be familiar to those steeped in the fantasy genre. You can see the inspiration of painters like Frazetta, who mixed violence and sensuality with palettes that were complementary to our game world. And there's little doubt that the decision to go mature in our cutscenes (though the game is Teen rated) has helped us more than it hurt. People love seeing the titillation of mild nudity in that opening cutscene, and the tapestry style situates the game fiction firmly in the realm of the

traditional Conan series and the like, which is exactly where we wanted to be.

The flip side to this is that a few folks have grumbled that the game feels like a fantasy cliché, in which a predictable trio of fantasy trope characters battle through well-worn spaces fighting enemies like skeletons, goblins, and zombies that they've seen in a hundred other titles. This is a fine line to walk. We sought to hew close to the 90/10 rule that suggests that a new IP needs to be 90% familiar, 10% fresh. Characters like our strange wights and levels such as "sea of sand" provided fresh content, while the rest of the game tended to walk more familiar paths. As Certain Affinity becomes more established, we expect we'll be able to begin pushing toward a slightly different mix, where a greater percentage of the game can provide unexpected delights, while still giving gamers what they want.

Still, we smile every time we read a review that compares the visuals in our game favorably to Diablo, knowing that we succeeded in giving the customers what they want and expect, even if there are other perils associated with this, as we'll discuss a little later.

WHAT WENT WRONG

1/ managing an open, collaborative environment takes more energy than managing tasks.

It wasn't long before we realized that not everyone is used to an open, collaborative environment; and it became even more obvious that managing this environment would be hard work. Getting feedback from anyone, at any point, can be overwhelming. Because it's so hard to take (and deliver) critical feedback well, this can be demoralizing for some people. We didn't realize until we were well into production that we needed to properly coach people in order for them to be comfortable in this environment.

One of the perils of an "all opinions are valid" culture is that people can quickly become confused amongst the swirling opinions and not know which feedback to act upon. Some people burned a lot of energy reacting to every bit of feedback. Others felt paralyzed by the feeling that nothing they did was right. At times, important critical feedback was lost in all of the noise.

The semi-flat hierarchy at Certain Affinity exacerbated this problem. Some people tended to assume that the hierarchy was really important in determining the weight of feedback from particular folks, so early on it would be common to hear, "You said you didn't like it, so I changed it," followed by, "Just because I said it doesn't mean that I was right!"

To make the process work smoothly, everything had to be run through the leads, but funneling all feedback through them was a challenge at first. However, with constant self-discipline, the team was able to more naturally digest and consider feedback, yet only act with the blessing of their lead.

2/ leads were in the trenches, wearing heavy hardhats.

As a studio we strive to have leads also be individual contributors, rather than just dedicated managers. This keeps us real, keeps us agile, and (we hope) keeps walls from going up between people "in the trenches" and project leadership. When a lead gets overloaded, though, it becomes a much bigger problem than just slipping tasks.

We used a gate process for deciding when a particular part of the game had hit our hard-to-define quality bar. Key stakeholders would evaluate the progress based on established criteria, and also the nebulous concept of "fun." The process worked well, once we got it in place and smoothed out the kinks. But it took time for people to recognize that the goal should always be to go into a gate review knowing that you'll pass it. The way to do this is to gather feedback from all key stakeholders in advance of any formal meeting, and make sure that their concerns have been addressed well in advance.

With so much feedback swirling in the pit, the leads were required to help their people interpret the storm of criticism and suggestions and help decide what was actionable and what it was alright to ignore (politely). When the leads didn't have sufficient bandwidth to manage and filter this feedback, the whole collaborative process broke apart. Individuals were left to sift through feedback on their own, and the gate process was held up by leads that were "too busy today, sorry." Islands of development started forming again—people stopped talking, stopped sharing, and stopped caring about anything other than their own work.

By the later stages of development, we had recognized the problem we had inadvertently created, and did a better job of lightening leads' workloads to help direct others' efforts. Now they had the time to mentor others and guide the less experienced team members. It was a simple fix, but the payoff was dramatic.

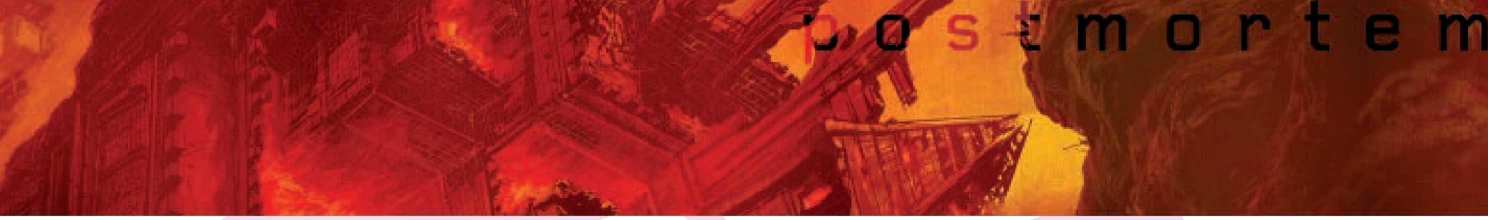
3/ building tools while you're building a game is like changing the tires on a moving car.

We developed our own engine, pipeline, and toolset for Crimson Alliance. Although its foundations were laid during our previous effort, Age of Booty, it was still nowhere near ready to create a third-person, isometric action RPG. The infancy of the toolset made early development difficult, and much of the burden was shouldered by overworked content creators.

With unfinished tools, it took far longer to prototype new game mechanics than it should have. We suffered from this in the early days particularly, when getting new ideas on the screen could take days or weeks instead of hours, as we would have preferred. Many behaviors required custom scripting, which was slow going. We ended up dedicating one highly technical designer to building "widget"-style tools for the others in order to speed up prototyping and iterating for the rest of the team. Had we used technology which was already mature, this redirected resource could have been devoted to building more content for the game or reaching a greater level of polish.

Ideally we would have had a dedicated engineer making improvements in tandem with early development work, learning how the tools are used and studying the habits of artists and designers right from the start. Instead, engineers were either trying to predict habits or were playing catch-up, and often ended up implementing features that were rarely used or failed to meet expectations.

Fortunately, one of the results of our open collaborative process was that content creators "collaborated" quite loudly and quite often when it came to pointing out shortcomings of the tools and pipeline. We dedicated an engineer to making efficiency improvements, and our tools and engine had improved dramatically by the time we hit our real production phase.



4/ creating new IP is a tremendous undertaking.

We began our journey in the realm of fantasy RPGs by working on a licensed product. That game was killed due to licensing issues, but we started a new project immediately with the same team, which was ready to take on the world. The pieces seemed in place, and we expected to hit the ground running, but we grossly underestimated the amount of energy that would be required to generate a new IP from the ground up. Beyond creating new fiction, new characters, and a whole new world that never existed before, there were big challenges related to communicating a fresh vision and getting buy-in from both project stakeholders and from the publisher. We were shorthanded throughout development, and so we ended up having to lean on our publishing partner for assistance with story creation and cinematics. While they were great to work with, the result was a lot of unexpected effort and cost, plus a lot of lost creative control.

Moreover, while we were glad we added the additional months at the end of the project for polish, we can't help but feel like our failure to properly account for the costs of building a new IP might be one of the big reasons that we needed extra time. During pre-production this problem needed focus, particularly from the leads team, but they were in the trenches with little time to communicate the vision. Settling on a story, themes, settings, and main characters that worked for everyone and didn't feel too terribly clichéd in a well-mined genre ended up consuming a great deal of energy. And while these were fun problems for the team to solve, it's a challenge that we'll approach with a great deal more planning in the future.

5/ we failed to properly set expectations amongst some reviewers and fans.

Crimson Alliance is neither Diablo nor Torchlight. As discussed above, we knew this from the onset, but we didn't do a very good job of letting the world know about this before release. Because the in-game art style ended up being very much like Diablo, comparisons were inevitably drawn to this beacon of the genre (which always flattered us). Our fixed-perspective isometric camera and the fantasy tropes that we indulged further reinforced this impression. In fact, the game is much more similar to a modern version of Gauntlet than it is to a classic loot-based dungeon crawler. Monsters don't explode like piñatas when killed, and the character advancement is streamlined for fast-action and couch play.

Unfortunately, Diablo and Torchlight have such a terrific following that many fans and reviewers felt betrayed when the RPG mechanics that they expected around loot and leveling turned out not to be a focus for Crimson Alliance. While many players and web sites have written that they love the game, others have panned us based on what the game is not. The amount of vengeful passion behind some of the more polarized scores and forum posts have led us to conclude that to many people, we've committed a nearly cardinal sin in the RPG world: We let them believe they were getting a "real" RPG, when in fact, they were getting an action game with RPG trappings.

Secondly, in concert with our publisher we experimented with a few new business ideas on Crimson Alliance. We allowed everyone to download a free version of the game that allowed for co-op online multiplayer (typically not seen in a traditional XBLA "demo") and let people play through a portion of the game as any of three characters. Then we sold an all-class pack for 1200 Microsoft Points ($15), or allowed customers to purchase a favorite single class at a lower entry point of 800 Microsoft Points ($10). The goal here was to try to ensure that price wasn't a barrier to entry for those recession-bound gamers who only wanted a single character class. Unfortunately we offered these additional choices without properly communicating to customers what the different options were. This led to a fair amount of online grief from those who felt tricked by the "free" version of the game. Our decisions in regard to communication on the dashboard weren't made capriciously, there are litanies of complex rules enforced by the platform and limitations in the Xbox Marketplace that led us to this structure, but the end result of this experiment ended up confusing and irritating a lot of customers.

Finally, we took the bold step of selling gold in the game. If a player wants to, they can buy a gold key worth 40,000 gold pieces for about a dollar from any vendor. We did this for two reasons. First, since all advancement is based on items, we wanted to make sure that when a friend comes over to play with a more advanced player, they could easily be "leveled up" to the proper point without having to play through all of the game's earlier levels. Second, we wanted to try an experiment to see if time-conscious gamers would be interested in speeding up their progress through the game. After players have completed the game, this option would give them a way to jumpstart subsequent characters. Because there is no PvP in the game we weren't particularly worried about game balance concerns. And of course, the best items in the game still have to be found by defeating challenges.

The sale of gold keys ended up being another highly polarizing element, in part because we didn't message it properly. Initially, hundreds of people decried us as money-grubbing bastards who were spoiling the purity of the game for the sake of an extra buck. In the West, it's an article of faith with hardcore gamers that "buying success" is sacrilegious. Many thought we'd designed the game to be unwinnable without spending additional money on gold packs. Of course, this wasn't the case, as everyone found out once they played it. The result was a clamor that died out fairly quickly, and many people quietly spent an extra dollar or two to help themselves, their kids, or their friends get up to speed quickly. There were others who just ignored the optional gold sale and played through the game as if it weren't there.

All of these failures in public perception could have been addressed through better upfront communication with our fans. Telling people clearly, "It's not Diablo, it's more like Gauntlet," would have reduced the number of people who felt betrayed by the lightweight RPG elements. More clearly labeling the base version of the game as a limited trial would have saved some users from a surprise when it turned out that not all content in the game was available for free. Talking more upfront about the in-game gold microtransaction model might have helped people understand that it wasn't an evil plot, but just another way to make the game friendlier to cooperative players and more accessible to those without a ton of time available.

While all the additional press we got from these three gaffes might have ultimately helped raise awareness of the product, we certainly learned a lesson about setting expectations more clearly.

A FUTURE ALLIANCE

We built a new game based on a new IP, using new tools and technology, with a team which had not worked together before. Although there are of course areas we'd love to improve, we're extremely proud of the results. We sincerely believe we ended up with an exceptionally fun, beautiful, polished game that breathes fresh life into the genre, and will have people playing together on couches and over the network for years to come.

While we woefully underestimated the difficulty of creating a new IP and prototyping with unproven, unfinished tools, we offset these challenges by hiring flexible people. We learned of the pitfalls associated with putting quality before any other virtues; our team often chafed under the culture of open criticism and collaboration, and our leads were often overwhelmed by interpreting and prioritizing so much feedback in addition to their regular duties. We learned a lot about the need to manage the expectations of our fans and critics. But by keeping quality as our guiding light, stubbornly insisting on a collaborative culture, and keeping a firm focus on the action core of the game, we feel like we've created a game many fans love, and a stronger company as a whole.

We may not have managed to put everything we'd hoped for into Crimson Alliance, but it's got a huge amount of the soul of Certain Affinity in it, and most of what's there is polished and high quality. We can't wait for our next great adventure.

PHIL WATTENBARGER *is director of product development for Certain Affinity.*

AUTODESK Maya 2012

REVIEW BY JEREMY PUTNAM

AUTODESK Maya 2012

111 McInnis Parkway, San Rafael, CA 94903
http://usa.autodesk.com

PRICE

- Starts at $3,495

SYSTEM REQUIREMENTS

For 32-bit Autodesk Maya 2012:

- Microsoft Windows 7 Professional, Vista Business (SP2), or XP Professional (SP3) operating systems
- Windows: Intel Pentium 4, AMD Athlon processor with SSE3 nstruction set support (or higher)
- 2 GB RAM
- 10 GB free hard drive space
- Qualified hardware-accelerated OpenGL graphics card
- Three-button mouse with mouse driver software
- DVD-ROM drive
- HDD: IDE, SATA, SATA 2, SAS, SCSI
- Microsoft Internet Explorer 7.0 or higher, Apple Safari, or Mozilla Firefox web browsers

For 64-bit Autodesk Maya 2012:

- Microsoft Windows 7 Professional, Vista Business x64 Edition (SP2), XP Professional x64 Edition (SP2), Apple Mac OS X 10.6.5, Red Hat Enterprise Linux 5.5 WS, or Fedora 14 operating systems
- Windows and Linux: Intel Pentium 4, AMD Athlon processor with SSE3 instruction set support (or higher)
- Macintosh computer: Macintosh computer with Intel-based 64-bit processor
- 4 GB RAM
- 10 GB free hard drive space
- Qualified hardware-accelerated OpenGL graphics card
- Three-button mouse with mouse driver software
- DVD-ROM drive
- HDD: IDE, SATA, SATA 2, SAS, SCSI
- Microsoft Internet Explorer 7.0 or higher, Apple Safari, or Mozilla Firefox web browsers

PROS

1. Improved animation tools, including viewport motion trails
2. Improved rendering tools, including Node-Based Rendering and Viewport 2.0
3. Improved save times and general performance fixes

CONS

1. Many bugs still lingering from previous releases
2. Premature release followed by several hotfixes
3. Many new features too specific in use, with questionable application

There's a joke going around various forums that summarizes my feelings about Autodesk's Maya fairly well: "Maya has the best animation tools. You just have to write them."

Autodesk's Maya is probably the most diverse toolbox product on the market. It caters to a huge range of industries, from games to film to commercials to medical animation. As such, many smaller and more specialized products are able to outperform Maya in specific tasks. Blender has a better UV unwrapper. RenderMan has a much more robust rendering engine. Modeling is better in ... almost anything, honestly. Due to Maya's breadth of purpose, Autodesk would be hard-pressed to counter these specialty products. Rather, it should continue to improve Maya's role as a generalist toolbox that offers greater stability, better performance, and wider functionality. Ideally, it would focus on these goals in that order.

When I looked into Maya 2012, my focus was more on increased functionality and improvements in what I consider the "essential elements," parts of the software that any Maya house will have to deal with. User Interface, integration with Python, and file handling are some examples. I also approached the product from the angle of my own profession: rigging and animation in the video game industry. Let's get into it.

WHAT'S NEW?

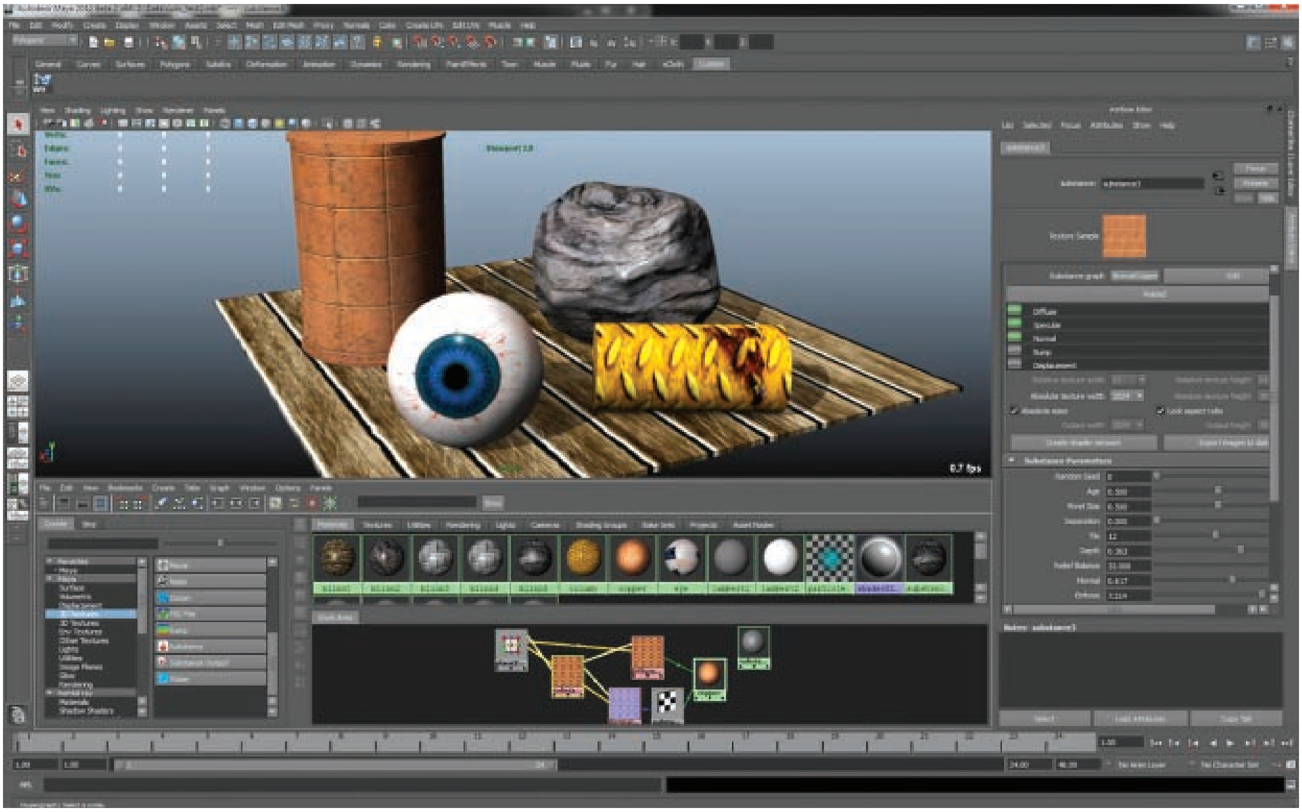
» Autodesk highlights Maya 2012's improved save times compared to Maya 2011. They were not kidding; in one instance, my test machine saved a large scene in four seconds in 2012 whereas it took a staggering thirteen in 2011. While these improvements are not as flashy as some of the tools we'll discuss below, they're still important. (If you think about all the files you open on a daily basis, this is really important!) Kudos to Autodesk for optimizing file load and letting users be more productive with their time.

It is interesting to see that Maya has added a node-based rendering feature. With this, Autodesk has opened up some nice compositing options without the need for rendering huge numbers of passes and transferring them to a program like Nuke. At this stage, however, these features seem more like stepping stones, or ways to quickly test lights or other small changes. Most productions will probably still need an additional compositing solution. This also means, however, that one could use these features as a partial solution, and that we may see future advances in this area, both from Autodesk and the community. I'm excited to see what sort of cross-scene compositing layer automation someone might come up with.

Perhaps more widely useful (at least to those with cinematic departments) are the Viewport 2.0 enhancements. Maya Hardware 2.0 offers a fairly advanced and comparatively fast solution to creating work-in-progress shots for review. With depth of field, motion blur, ambient occlusion, and more, Viewport 2.0 can now create a very advanced preview of your scene. But it does tend to chug along at low frame rates. It also has a tendency to bake in some of the effects, which can yield some odd results when the user changes the camera. I'd be inclined to use this for spot-checking different frames rather than watching at-frame-rate playback.

I'm very excited about Maya's new viewport editable motion trails. Although it's a feature I've seen in other plug-ins, having this in the base Maya install is a great addition. The controls are very straightforward, and everything live updates how you'd expect. Almost any animator should get great use out of this tool, both for editing motion and visualizing curves and timing. Going by the script editor output, it looks like it will be fairly easy to build tools around this new feature as well. Each point in the motion trail is defined as part of a sequential list, has the expected parameters, and is able to be queried. I look forward to seeing how the community helps this feature evolve further.

It feels as though HumanIK gets a major overhaul with every Maya release. With Maya 2012, much of the interface and workflow has been examined. The new Characterization and Character Controls windows allow for a very visual setup when assigning bones and editing animations. Overall, I think this is a strong set of tools. The features offered are very attractive. Still, I am wary. Different productions have widely divergent rig setups, and I'm concerned about some of the dependencies this system may have. For example, when I was attempting to apply FBIK and copy animation between two characters at work, I had several perplexing results that involved leg bones and other things flying off into space. However, this may be a conflict with our rigs or merely a learning curve for the tool. I was able to reach a working retarget, but it required a lot of manual finagling on my end.



Another question I had was looking at the Auxiliary Effectors for space switching IK controls; essentially, there's a one-button option for creating a temporary constraint when one needs to pin a hand or foot to something. While this is nice, such a constraint has been easy to create in previous releases. I wonder if this was added to avoid some edge cases where manual constraints may fail. While it's very important for many projects to have a robust retargeting rig system, I'd be wary of using FBIK for that purpose unless the pipeline was designed around this tool from the beginning.

IT BUGS ME!

» When thinking about the downsides of Maya 2012, the question that most frequently pops into mind is, "Why is this still not fixed?" Why do I still lose all custom weighting data if I accidentally flood to one hundred percent and then undo? Why do character sets still let you ruin animation data on referenced scenes by changing arbitrary nodes? Why are there still character sets at all? I'm not going to list every bug or oversight that I'm still waiting to see addressed, but suffice it to say that there are still plenty of improvements for Autodesk to make.

I recognize that it's unrealistic to expect a bug-free release, but it's unfortunate to see what is receiving attention while these bugs linger. While it's very nice to see a vehicle rig with nifty controls on the craft animation tools, how many projects need this type of vehicle simulation? Certainly there's a decent number of third-person shooters that might use this type of rig. By comparison, however, how many people would benefit from improvement to the weighting system? And how essential are each of these features? It's unfair to consider the vehicle rig a downside to the new release; rather, I am disappointed with what it states about Autodesk's priorities.

The Substance smart textures, similarly, are valuable directly in proportion to how customizable they are. The premades I've looked at seem fairly high quality and have a plethora of parameters, but I'm not too excited that Maya is helping me make parquet floors. Perhaps I'm not seeing the full value; I don't deal with procedural textures in my daily work. Surely there are users out there who have been waiting patiently for more robust texture presets in Maya. Still, these features seem too specific in application to warrant Autodesk's development effort.

The 2012 release continues a very concerning trend for Autodesk: a troubled initial release followed by many hotfixes. Maya 2011 had a total of three hotfixes, but Maya 2012 has already passed that with four updates in six months. The impression one gets is that Autodesk is overly focused on big releases each calendar year, rather than focusing on the overall quality of the product. This makes studios very wary: why be an early adopter if the product has a history of multiple major hotfixes before it becomes usable?

I wonder if the defaults that Maya comes with are the result of user studies. If so, I must have very different preferences than the average user. I've never worked with someone who prefers interactive creation for polygon objects, nor enjoys the "post" option for skin weighting. How is it intuitive to allow painting values to add up to numbers other than one (one hundred percent)? This is generally a one-time annoyance at initial set-up, but has been a long-standing question for me.

Overall, Maya 2012 has added some interesting functionality. The more tools in the toolbox, the more the development community can build upon Autodesk's progress and create solutions matched to their specific needs. The overall product, however, still feels too rushed. I will continue to use 2012 at home, but would not recommend upgrading from 2011 for professional work quite yet. The risks outweigh the added functionality.

JEREMY PUTNAM *is a technical animator at Riot Games. He works on rigs and tools for the popular online game* LEAGUE OF LEGENDS. *He holds a BFA from Ringling College of Art and Design in computer animation. The opinions expressed in this review are those of the author, and do not reflect those of Riot Games.*

gd

gd
BULLETSTORM POSTMORTEM
GAME DEVELOPER
gd
gd
TERMINATOR SALVATION ARCADE
gd
INSIDE: TOP 30 DEVELOPERS
LEADING GAME
gd
2011
SUPERMEATBOY
gd
GAME DEVELOPER MAGAZINE
gd
sid meier's civilization v
2011
gd
dancecentral

PROMOTING INDIES

LESSONS FROM OUR GRASSROOTS CAMPAIGN FOR XBOX LIVE INDIE GAMES

» Xbox Live Indie Games (XBLIG), have been considered the red-headed stepchildren of Microsoft's Internet-based platform ever since they were first revealed. Hidden beneath layers of menus and triple-A titles, XBLIG sales and promotions have never been outstanding for developers, unless they managed to earn a spot on the highly coveted Top Downloads section. In an attempt to overcome this challenge, in the winter of 2010 a group of XBLIG developers banded together to create the community's first grassroots marketing campaign: the Indie Games Winter Uprising. Sales were mixed, but along the way the marketers learned quite a bit about the community and increased awareness for the platform as a whole.

Recently, a new group of XBLIG developers banded together for a similar campaign, dubbed the Indie Games Summer Uprising (IGSU). This time our goals included promoting the platform while showcasing the diversity, talent, and potential of the development teams. Collectively, we've learned a lot from the promotion, and most importantly, we've come to understand that with the right people and an important message that a lot of people can relate to, any group of developers can see similar success.

THE FULL INDIE GAMES SUMMER UPRISING ROSTER:

- Battle High: San Bruno
- Chester
- Cute Things Dying Violently
- Doom & Destiny
- Speedrunner
- Take Arms
- T.E.C. 3001
- Train Frontier Express
- Raventhorne
- Redd: The Lost Temple

I'm grateful that the Xbox LIVE Indie Game community granted fellow coordinator Kris Steele and me the opportunity to build a team that would ultimately promote their studios and titles. Since I first became active in the community around March 2010, I knew that there was an abundance of talent available and that it was only a matter of time before gamers were made aware of not only the shining titles on the platform, but of the talented development teams as well.

In March of this year, conversation began on Microsoft's App Hub (XNA) forums to get the Uprising campaign going again, but nothing was moving. Around May, Kris, a member of the previous Uprising, resurrected the thread and began to light the fire once more. Together, we naturally fell into the role of organizers for the campaign and started to lay the groundwork. After three months we had selected ten titles from a pool of more than 70. The XBLIG developers would choose eight, while the gaming community as a whole would vote on the remaining two titles. These games would be released over a two-week stretch at the tail end of the summer.

We tried to make the selection process as democratic as possible. Working off the feedback from the App Hub, Kris and I managed to get Scott Nicols of GayGamer.net and Ryan Donnelly of VVGtV.com to serve as a panel of judges who would narrow down the selection of 70+ titles to 25. This proved to be more of a manageable number for the IGSU developers to vote on who should be in the top eight. Independently, we rated each title on a one-to-three scale, essentially based on their marketing material. We considered gameplay videos, screenshots, and how far along they were in the development process. Any titles which we each gave a "one" were automatically granted a spot in the top 25. This quickly filled up 13 slots. Titles which received a "two" could go either way, while those we graded a "three" would require a judge to really make an argument for why it should be included. Needless to say, it was the least entertaining element of the IGSU.

THINGS THAT WENT WELL

• Marketing and Buzz

One of our goals for the promotion was to generate as much support as we possibly could for the XBLIG community and XNA developers. I firmly believe we accomplished this, as we presented the community in a positive light by showcasing the quality and variety of games the XNA toolset was capable of, especially when used in the hands of capable developers. Some of the industry's largest blogs picked up on the Uprising, even at its inception, including Game Informer, Kotaku, Joystiq, and IGN.

The word spread like wildfire, even when we didn't intend for it to! The first pseudo press release I sent out was really more of a call out to developers, asking them to contact Kris and

GRASSROOTS
MARKETING

me to gauge interest for the promotion. Within a few hours a number of news outlets picked up on it, and helped us spread the word. I didn't feel comfortable sending the release to any sites, as I didn't believe they would feel it was newsworthy, but apparently they did, as did developers—within 24 hours we had received more than seventy titles for consideration for the IGSU.

In addition to handing out press releases, Kris and I appeared on podcasts together. One of the larger ones was Episode 4 of Joystiq's newly formatted "The Joystiq Show," which was quite an honor in itself, considering I've been a fan of the site for several years.

As the campaign continued, Microsoft also promoted the Uprising, beginning with Major Nelson's tweets about the release dates. Ultimately, though, I believe our biggest publicity boost came when actress Felicia Day tweeted a Kotaku article titled "These Eight Xbox Indie Games Will Rise This Summer" to her 1.8 million followers. That article alone received over 149,000 hits. After all the titles had been released, Microsoft revealed a dashboard promotion, which certainly helped with sales but ultimately showed a flaw in the system: It is very difficult to navigate to the XBLIG Marketplace. I'll cover this later in the article.

In addition to this, I attended GDC Europe and Gamescom in Cologne, Germany, and immediately followed it up with PAX Prime in Seattle. Although I was at these events to cover them for the site Armless Octopus (where I work as managing editor), I tried to plug and promote the IGSU as much as I could through networking and handing out business cards.

• Appearance and Public Image

I feel we excelled in presenting a well-organized and professional public image. Commonly, independent developers spend quite a bit of time creating their titles, but fall a bit short in the marketing department, and tend to send out unpolished press material. In fact, I was recently part of a panel at GDC Europe where we discussed the often-lackluster press releases that small studios send out.

One of the major factors contributing to keeping a clean image and maintaining synergy throughout the campaign was the excellent team supporting our efforts. You'd be amazed at how many people will volunteer to help a cause they believe in. Our first trailer was created by Christopher Brousseau of Game Production Studios, whose title Dragons vs. Spaceships was in the IGSU top 25. He did an excellent job of conveying the image we were looking for, as it instantly helped sell the campaign and was one of the first things that gamers saw in terms of gameplay for the first few weeks. Afterwards, Ryan Donnelly of VVGtV.com took charge with creating new trailers, and Zack Parish (Saturnine Games) continued to create new soundtracks for each one.

> "Word of mouth is the best form of marketing you can have—your consumers sell the product or idea for you."

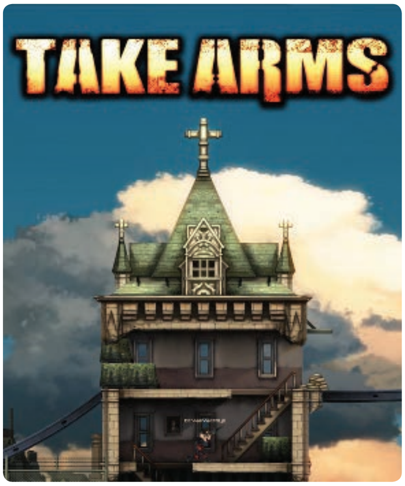
Almost immediately after the IGSU launched, Nathan Smith and Josh Addison (Blazing Forge Games, Redd: The Lost Temple) began to outline the prototypes for our web site, along with Kris, who handled a lot of the back-end .aspx work. From the moment I saw the prototype, I knew we had something special. In addition, Nathan drew excellent caricatures for each title as it was announced and placed them on the home page. Along the way, the three of them continued to maintain the site and add content. When all these elements came together at once, we effectively conveyed the image of an organized team, and maintained the same message throughout the duration of the campaign.

• Support for XBLIG

Word of mouth is the best form of marketing you can have—your consumers sell the product or idea for you. In our case, we had overwhelming support from our loyal fans and community.

Early on, Kris had the excellent idea of using a fan vote to not only involve the community, but also help spread the word that the Uprising was making another pass. Again, Josh and Nathan came to our aid. They had their coworkers create a poll in which fans could vote for the title they would most like to see included in the Uprising, after the first eight finalists were selected by the developers. We included the two titles with the highest number of votes in the final two top 10 IGSU slots. This helped us drive traffic to the Facebook page, which allowed us to speak directly to our audience, but also to the IGSU site.

If used correctly, Facebook can be an invaluable marketing tool, but I find that organizations far too frequently don't take a personal approach when interacting with their audience. The four of us logged on multiple times throughout the day to post updates, share links and news coverage on the IGSU, ask and answer



questions, and offer giveaways. This back-and-forth dialogue with our fan base allowed us to quickly balloon to 1,200 likes and over 4,000 votes for the fan-selected titles. Even after the titles were released, I would tweet or post a message stating that the first person to post a screen shot of themselves playing a selected IGSU title would win the abundance of swag I had acquired from my brief "press tour." Within minutes we would have multiple posts, each time from new users.

Even people that weren't involved in the XBLIG or XNA communities were talking about, tweeting, and covering the IGSU for the simple fact that it was a grassroots campaign designed to showcase the best work from people who are wildly passionate about what they do.

Finally, after the poll closed and the results were tallied, we announced the winners in a Twitter chat using the #IGSU hash tag. Looking back, I probably should have planned that a bit better. I called for a 9 p.m. EST meeting, but was in Germany at the time for GDC, making it 3 a.m. for me. Needless to say, it was well worth it, and a tremendous number of people showed up to participate and ask questions. I think these kinds of community activities best exemplify the strengths of being indie; a triple-A publisher would get swamped with responses from something like this. Fortunately our audience's manageable size allowed us to have a meaningful conversation with everyone involved.

• Learned a Bit About Marketing

Developers are a funny bunch. They can create massive overarching storylines, thought-provoking characters, beautiful environments, and intuitive gameplay mechanics, but most of them couldn't market their title if their life depended on it. That's where a publisher—or in our case, IGSU—comes into play. The Winter Uprising took a vastly different approach in that they approached developers independently and build a promotion around those titles, whereas we established a promotion and then gathered teams to be a part of it. Ultimately, both Uprisings proved to be a critical success for not only the developers, but the platform as a whole.

All too often, developers dismiss the importance of marketing and public relations. As much as we'd like to, we can't all achieve the runaway success that Notch has with his hit title Minecraft. We started marketing early, and kept the push consistent throughout the campaign by rolling out continual updates to keep the public informed. From the first week of the promotion, I had a clear picture in my mind of a press release schedule. Rather than swamp journalists with every new feature, screenshot, or update we had, we instead slowly released information, and included them in one press release every two to three weeks. This proved beneficial because nearly every press release generated a unique story, from the initialization of the IGSU, to the developer selections, to fan vote dates, to the release schedule.

Our intention was to keep consumers hungry for information, while feeding them morsels of news, trailers, and participation opportunities.

As the promotion began to take shape and appear on popular industry forums and blogs, I think the developers began to appreciate its value and contributed to the buzz. One critical part of marketing, which I know most didn't get right away, is accessibility—I had to hunt down most of the developers' contact information. If you are selling something, whether a product or idea, it is imperative that you offer numerous ways for people to get in touch with you, and make it clearly visible on everything you do, from a web site to an e-mail signature.

I've learned that above all else, communication is key, whether organizing teams of developers, or speaking with the community. Kris and I sent out dozens, if not hundreds of emails a day to the development teams, press, and fans, to make sure that we were all on the same page at all times. I've also learned that sometimes it's futile: We included "Please contact the respective developers for review codes," as the final line on every press release, but we were still asked for review codes.

THINGS THAT COULD HAVE GONE BETTER

• Scalability

As I mentioned earlier, Kris and I quickly became overwhelmed with the responses to the IGSU. This was a blessing in that we had a plethora of developers and titles to select from for our campaign, but a curse in that it was too much for two individuals to handle. I suppose this is where my lack of faith in people came into play as well. Trying to convey a unified message is difficult when you have numerous people responding to emails, but we probably should have taken the model presented by the Humble Bundle.

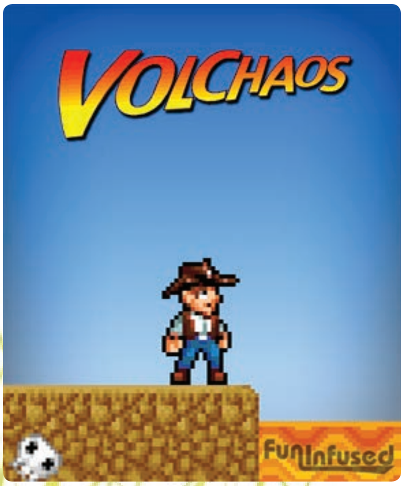
I saw a GDC panel with the guys who organized the Humble Indie Bundle afterward, where they described a similar situation. They chose to adopt an e-mail system that was composed of "tickets," which allowed multiple members of the campaign to respond to open queries at the same time, thereby seeing everyone else's responses. It looked like TweetDeck, except each tweet was a response to a question, of which dozens were open at one time. Perhaps if we do another Uprising we'll adopt that system. I'm certain that I failed to respond to at least a handful of messages simply because they got lost in the shuffle, and I felt horrible afterward.

My only concern with using something like that would be if someone spread incorrect information. A member of the press might run it and continue to spread the erroneous report. An easy way to resolve that situation would be for us to be proactive instead of reactive. That is, if we had a clearer, more fleshed-out plan at the very beginning, then we wouldn't have to worry about someone passing along bad information. I doubt that we could have anticipated every scenario, though, because as the project progressed people asked questions and brought up angles I had never considered before.

Scalability from the developer's side of things was an issue as well. Kris, Josh, and Nathan were constantly adding content and updating the web site on a daily basis as developers delivered new assets. Needless to say, this was time consuming and inefficient. If we had more time to organize, it would have been wise to allow developers to add their own content, using the template we set up. I commend the three of them for always managing to stay ahead of the curve and keeping the content up to date. The only thing I had to manage was updating the press coverage, so I had it easy.

• Set Clear Rules and Guidelines
As I previously stated, regardless of how clear you think you are on a point, someone will figure out another angle, or will fail to read what you wrote. Kris and I initially approached the IGSU as an organic process, in that we had a set of guidelines but wanted the campaign to be as democratic as possible. Therefore, we were constantly asking for developer feedback. My fear was that the developers would begin to think, "Who is this guy, thinking he can just step in and run things how he chooses?"

The App Hub served as our base for delivering and weighing ideas. It goes without saying that despite a lot of the positive feedback we were receiving, there was also negative. You'll never get a room of people to all agree on one thing and you can't please everyone, so just do your best to please those whose opinion you value most. One way to avoid this problem is to approach the developers with a clear set of rules and guidelines. At some point you have to say, "This train is leaving the station. You can either hop on, or you're being left behind."



I wouldn't expect anyone to get it right the first time, but the lessons learned from this experience alone would benefit us exponentially should we decide to do a marketing campaign of some sort again. Our team was also built organically, because it came together as we moved along, and people realized this was worth their time. Relying on the kindness of strangers worked out well for us this time, but I would suggest having members established before you start a campaign of this scale. We were always transparent about everything we were doing, and I firmly believe that is the best way to operate any business, but people began to recognize a conflict of interest as we moved ahead. I was writing for Armless Octopus, a blog that specifically covers XBLIG and XNA titles, so I stopped doing any work there while promoting the event. In addition, Kris was submitting VOLCHAOS as a candidate, as were Nathan and Josh with their title REDD: THE LOST TEMPLE, which would later be the last title released in the Uprising.

The Winter Uprising experienced the same situation; the organizers were also the ones developing the titles. I believe ours was done in a more democratic fashion, though, because we allowed for both developer and fan voting in two separate stages.

Finally, I can't stress enough the importance of having one concise mission statement overarching all that you do. This constantly reinforces what the campaign is about, not only to the developers on board, but also to fans and the press. People would often ask, "Why are you doing this?" and I would respond with the mission statement: "To promote Xbox LIVE Indie Games in the best light possible by showcasing the diversity, talent, and potential of the platform as a whole."

• Have Developers on Board, Then Do a Promotion
As I mentioned earlier, the Winter Uprising took the opposite approach when coordinating their campaign. They initially gathered a group of developers based on their titles, and then generated a campaign to market those titles. This has a number of advantages, such as having everyone on board from day one. The first half of our marketing push was spent promoting the campaign itself, first celebrating the fact that we had more than 70 entries, followed by us narrowing down the selection to 26 titles (whoops!), and finally the 10 IGSU finalists. Looking back, some may consider this a waste of valuable marketing time, but I believe it generated hype and buzz around the whole event. A few people disagreed with the eight entries that the developers voted into the IGSU, so we gave fans of XBLIG the opportunity to vote on any of the eligible initial entries to fill the final two spots. The fan vote had both a positive and negative effect on the two titles. The positive effect was the idea that the title was selected because people genuinely wanted to play it! However, this meant those titles lost out on two weeks of marketing.

Another contrasting feature from the Winter Uprising was the experience (or lack thereof) on the part of our developers. The IGSU only had a few developers who had released an XBLIG title before, or were previously involved in the industry, while nearly all the winter entries came from veterans. That's not to take anything from our developers, but a bit of experience can go a long way, especially when you factor in the quickly-approaching deadlines and narrow margin for error that we could accommodate.

• Developers Need To Do Some Marketing of Their Own
Not to say that any of the developers didn't market their titles—they all certainly did. The unforgiving time constraints I alluded to earlier weren't any help either, as most of their precious time was spent on development. Hopefully this campaign at least reinforced the value of marketing your brand, as well as having media assets readily available for the press. This includes at least one trailer, multiple screenshots containing critical gameplay elements, and perhaps a few concept images.

When I was seeking developers to build our initial candidate list, I noticed that many of them

CONCISE MISSION STATEMENT

did not prominently display any way to get in touch with them. In particular, the Dream.Build.Play entries that I was trying to scout from had this issue. I had to parse through all of the entries, check their YouTube links, and hope they had either a site or e-mail address listed there, though many did not. At that point I gave up and moved onto the next developer.

Marketing on the developer's end may have helped a bit, but it's difficult to gauge on the Xbox Indie Marketplace. I've noticed that a dashboard promotion drives a far greater number of sales than any marketing campaign could ever do. There are a number of factors involved here, but I believe the largest contributor is the number of steps required to find the titles, never mind purchase and rate them. Looking back, I should have put more of an effort into coordinating more of the development teams with podcasts, journalists, and fellow gaming hobbyists. In any grassroots campaign it's critical to win the hearts and minds of consumers, especially within a community as niche as Xbox LIVE Indie Games.

Small things go a long way, such as having a detailed e-mail signature. This should include your name, contact information, and which developer you work with. As the promotion went on, I began to learn everyone's names and which team they were working with, but at first many of their signatures did not include any of this information. That could have cost them some valuable networking and contacts early on when trying to contact industry professionals. You'd be surprised at how many click-throughs you receive to your site from your e-mail signature alone.

> "When I was seeking developers to build our initial candidate list, I noticed that many of them did not prominently display any way to get in touch with them."

• Time Constraints

When forced to work under tight time constraints, people often generate some of their best work, due to the pressure of the deadline. There are others who fold under that same pressure, however. Fortunately, we didn't have any of those individuals in the IGSU.

The IGSU was initially organized in early June, and the titles were to be released at the end of August. That only gave us 10 weeks to gather developers, create a web site, art, trailers, and music, as well as finish creating the titles, playtest and peer review them, all while marketing the IGSU. Needles to say, efficiency is key, and it was a summer of crunch. A few beach days were missed (I live on an island), but in the end it was well worth it.

There were a few reasons we chose to go with the final two weeks of August to release the titles, but Kris and I couldn't reveal the details to the developers until after the IGSU. You see, from the beginning we were in touch with Microsoft, which was in full support of the campaign the entire time. I'll touch on the issues with Microsoft, but its support was the driving force to why we did the promotion in August. We didn't want to compete with Microsoft's internal Summer of XBLA promotion, as it's promoted heavily and would obviously steal a bit of our thunder.

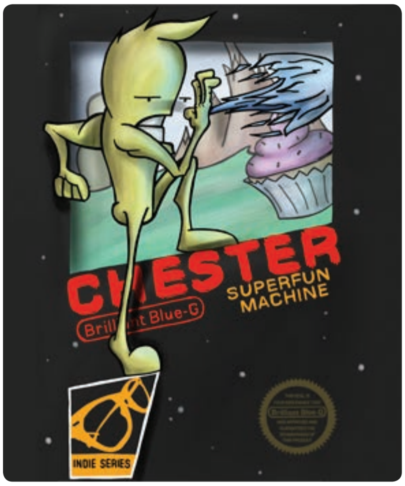
Microsoft's Dream.Build.Play promotion was occurring at the same time as well, in which their XNA team selects the standout titles from a pool of entries, very similar to our promotion. Their rules are slightly different, so many of their titles had already been released and were playable. The finalists were to be promoted during Seattle's PAX Prime event, with the winners got an opportunity to win an XBLA contract. Dream.Build.Play benefits the Uprising in that it already has a buzz surrounding the XBLIG marketplace and XNA tools, as well as promoting several IGSU entries. Some people saw the Uprising as a direct competitor to DBP, but that couldn't be further from the truth: Both parties were well acquainted with one another and were there to promote the same platform!

Because a dashboard promotion was never guaranteed, it was suggested that Kris and I not say anything to the developers until we knew it was certain, and we agreed. We were told that our best hope for a dashboard promo would be if we had the games released over a two-week period, instead of four as initially planned, and for it to be the final two weeks of August. Besides, that technically was still within the confines of summer. Good luck trying to explain to more than 70 developers why they need to have their titles completed in 10 weeks when you can't relay any of the backstory. They probably thought we were crazy. Kris and I wanted all 10 of the finalists to have their titles finished and in the peer review process two weeks before the IGSU was set to begin, so that made the time constraints even more pressing to the developers.

In the end, all but one of the titles hit their release date. TAKE ARMS was pulled shortly after it was released, though, as the developers found a latency issue when playing in a full room. It's difficult to test for issues such as this in peer review because the developer would need to round up seven other XNA developers each time they wanted to test a change in their network code. REDD: THE LOST TEMPLE just missed its release date due to a last-minute bug. It was corrected the next day, but XNA titles need to sit in peer review for at least 48 hours before they can be released, so it didn't appear for another two days. I think it's fair to say that the developers put their hearts into their work to hit that tight launch window.

•The Dashboard

The XBLA dashboard itself seems to be the inherent issue with XBLIGs. It's often confusing



and difficult to navigate, largely due to the constantly-changing panels, which reflect the day's promotions, games, and videos. There is still hope, however, as a there is a dashboard update slated this fall, titled Metro. Could this be the answer we've been looking for? Perhaps. But until then, here are the necessary steps to purchase an Xbox LIVE Indie title:

1. Log into an Xbox LIVE profile capable of purchasing games.
2. Scroll to the Game Marketplace.
3. Click Explore New Games.
4. Scroll up to Games & Demos.
5. Scroll right and select Indie Games.
6. Scroll up to Titles A–Z.
7. Scroll through T for Train Frontier Express.
8. Click Buy.

I believe you've already lost most gamers by the fifth step, as they crave instant gratification and are constantly bombarded with advertisements along the way. Let me make it clear that it is not the ads that are the issue, but the fact that so many steps are required to find a title.

OTHER NOTES

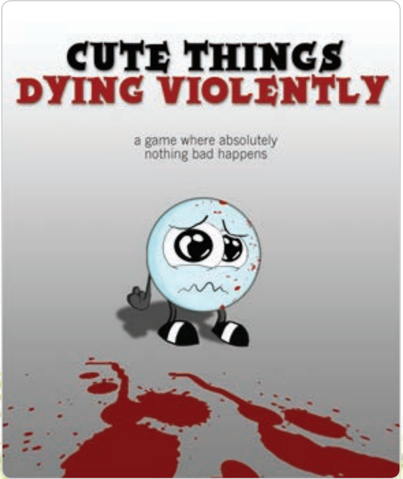
In closing, I want to clear some things in the air on Microsoft's behalf. Frequently, XBLIG developers feel as though Microsoft does not care about or support their cause, but this simply isn't the case. In working with numerous facets of the organization, I can tell you that they very much do care for the indies, but like most things in this industry, Xbox Live is driven with business in mind first. As part of our XNA Creator's membership, we each pay a $100 annual fee, and forfeit 30% of our profits on titles sold, which is what Apple charges for use of the App Store. In total, the amount of revenue that XBLIG developers generate for Microsoft is minuscule in comparison to what other services such as XBLA can potentially deliver. In addition to this, it costs publishers tens of thousands of dollars, generally much more, to have an XBLA title released on the platform. For these reasons alone it is not fair for XBLIG to receive equal promotion and support from Microsoft when other publishers are forking over valuable funds for things like a dashboard panel.

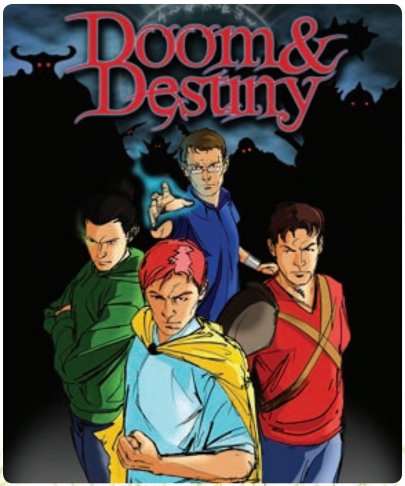
Perhaps at the risk of upsetting some of its large publishing partners, the Summer Uprising was still fortunate enough to receive not one, but two dashboard promotions from Microsoft—at zero cost to the developers. Picture the dash in terms of advertising revenue; there are 35 million active LIVE users according to the Major Nelson Podcast, Ep. 410, 9.11.11, and a large percentage of said users view promotions on the dashboard. Naturally, the further you move from the initial launch screen, the lower the value of the dashboard position. As I mentioned earlier, Microsoft was on board with this promotion from the very beginning, whether it was through its PR company, Edelman, its XNA team in Redwood, or even Major Nelson. In the end, Microsoft cannot give preference or support for only one XBLIG title, but it has the ability to support a campaign as a whole.

Ultimately, I'm proud of all that we've accomplished. We've not only exposed some of the weaknesses in the system (such as few problems with XBLIGs on the dashboard), but shown some strengths as well. In all honesty, I think you would be hard pressed to find another marketing campaign as successful as ours, and with out-of-pocket cost of basically zero dollars. Were we a financial success? Not by any means, but that was never the goal. We've raised awareness for the platform, continued to allow XBLIG to leave its mark on the map, and perhaps inspired others to make the changes that they would like to see as well. At the end of the day, I can say this is was an outstanding experience for all of those involved.

So what's the lesson to be learned from all of this? A bit of careful planning, some community support, and a strong network of developers can go a long way, and lead to a successful campaign that can prove beneficial for all parties involved.

DAVE VOYLES *is based out of Long Island, NY. He is currently working on a* MEGA MAN *2 remake using Unreal Development Kit, and has worked as a technical reviewer on two books covering the engine. He holds a BS in communications and is presently working on his MBA. He is also managing editor of Armless Octopus, a blog covering XBLIG and independent game development. For more information, visit his site at www.DavidVoyles.Wordpress.com.*


CUTE THINGS DYING VIOLENTLY
a game where absolutely nothing bad happens



STRONG NETWORK

INDIE GAMES SUMMER

GAMMA DRIVE ME CRAZY!

WHEN IT COMES TO COLOR, CONSISTENCY IS KEY

Do you ever feel like things don't add up? Do you feel as if you're trapped in a Thomas Pynchon story, where you constantly encounter hints of an elusive but all-powerful conspiracy behind lots of seemingly unconnected events? And no, this is not about your last Metacritic scores, or the way your profit sharing is calculated.

No matter how little you care about the computery side of game art, you can't sling pixels for a living without learning a few of the basics. You know, of course, that those pretty pixels on your screen are just numbers that represent how much red, green, or blue light is being emitted by your monitor. You can probably translate a string of numbers like 206,79,111 into a charming honeysuckle pink without thinking too hard about it. And of course you've learned from Photoshop how different computations done to those numbers can create different effects, so that "adding" the pixel values of two images is almost like projecting two slides onto the same screen at once, whereas multiplying the same two images is more like superimposing two transparencies.

But what if I told you all this was an illusion?

SLEIGHT OF HAND

» Here's an experiment that will surprise you. Make a Photoshop image, and then fill it with the default pattern that gives you alternating black and white pixels. At 100%, this will approximate a 50% gray tone. If you zoom out, though, the image will appear to darken to almost 25%. Even if you resize the image itself to a lower resolution, you will get the same result.

If you check the pixel values in the resized image, you'll see they're exactly what you expect: 128, 128, 128. That seems to make sense, since that's halfway between black (0, 0, 0) and white (255, 255, 255). There's a certain three-letter internet acronym that comes to mind here ... but to keep it family friendly we'll just ask, what happened?

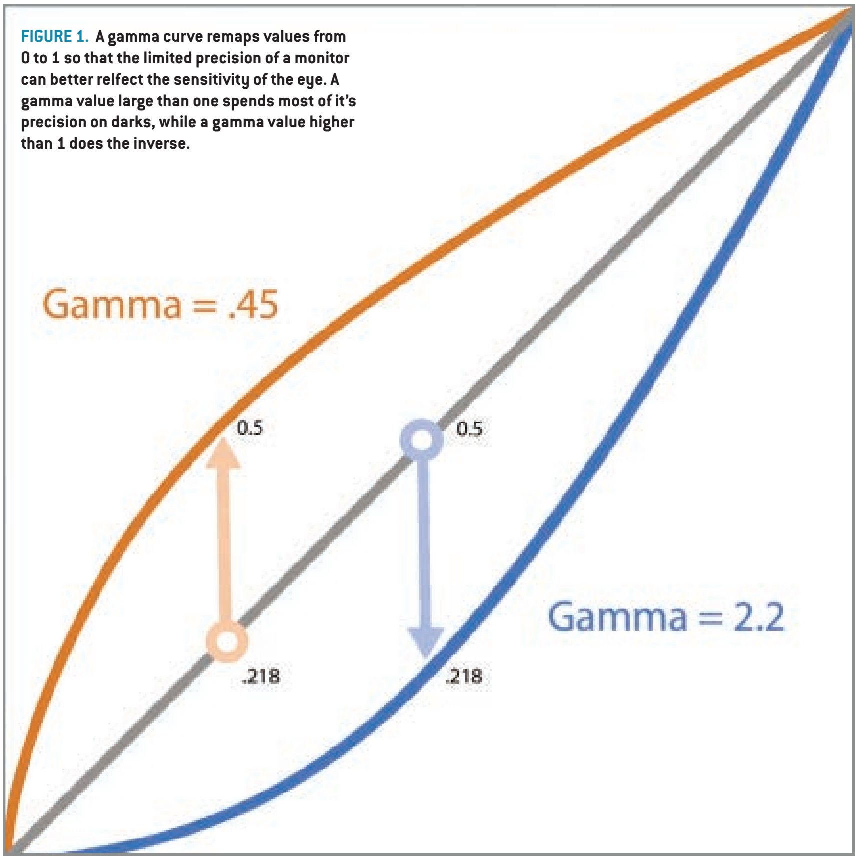


FIGURE 1. **A gamma curve remaps values from 0 to 1 so that the limited precision of a monitor can better relfect the sensitivity of the eye. A gamma value large than one spends most of it's precision on darks, while a gamma value higher than 1 does the inverse.**

Those numbers we know so well aren't behaving like we'd expect. The math works out fine—the 128 gray value is the obvious guess for halfway between black and white—but it looks wrong because our monitors don't show us a linear gradient from black to white by equal steps. They remap the values between black and white along a curve, known as the gamma curve (see Figure 1). That means the color which a printer's screen or a photographer's light meter calls "50% gray" is actually RGB 186, 186, 186 rather than 128, 128, 128, so when the pixels in the test image are averaged out to 128, they appear to turn darker.

This is where the conspiracy theories start to come in. Why do we bother with these "gamma curves?" Wouldn't it be easier to just let the pixels say what they mean?

While it seems like a pointless complication, there are very good reasons to use gamma. Our eyes aren't equally sensitive to all intensities of light. We do a very good job of distinguishing dark shades, but we're not nearly as good at picking out precise shades on the bright end. Gamma is an attempt to reflect that variable sensitivity in the limited range of values you can get with only 8 digital bits of precision. With only 255 discrete steps between black and white, TVs and computer monitors use gamma curves to show more detail in the lower registers where our eyes do well, at the expense of the brighter values. That's why 50% gray is located almost 75% of the way between 0 and 255; it's because the display hardware needs to conserve precious bits.

For most of us who've been working digitally all our professional lives, this magic goes on behind the scenes and rarely gets in the way of our work. Unless somebody has been fooling around with the color-correction settings on their monitor, or messed with the color profiles during installation, we're so used to the nonlinearity of the colors we use that it never bothers us. If you showed most of us a swatch of 128, 128, 128 gray, we'd automatically identify it as 50% gray. But anybody who has worked in print will usually identify the same color as closer to 75% gray; if you've ever looked at the Pantone color swatches in Photoshop and wondered why the selections seem so washed out, it's because the Pantone sets generally follow optical brightness, not our gamma corrected notion of brightness.

> “ The key to working with gamma is to get your colors to behave the way we all thought they did before the veil of secrecy surrounding all of this gamma stuff was lifted. You want to convert those dubious gamma-corrected color numbers into nice predictable linear numbers, where “twice as bright” translates into a number that's twice as big.

If your job never gets close to the mathematical roots of all this stuff, the knowledge of how it works is just a bit of boring but ultimately harmless trivia. We care more about what our images look like than what names to apply to the pixels in them. Unfortunately, gamma becomes anything but trivial if you're trying to use pixels for any purpose beyond just showing them on a monitor.

When you look at one of your files, it's important to remember you're seeing a remapped version of the image on your hard drive. The numbers on disk are actually those of an un-gammaed image—all that more-precision-in-the-darks stuff happens only when the image is displayed. Again, the conspiracy theorists are right: Things are not what they seem!

Usually, this difference between the visible image and the stored image doesn't matter. When you grab a picture with your digicam, the camera saves the color values using the inverse of a standard 2.2 gamma curve. That is, it artificially brightens the mid-tones to compensate for the way your monitor's gamma will artificially darken them at display time. As long as your monitor and software are all in sync, the difference between the stored image and the displayed one should be minimal.

However, things get trickier if you want to try some extra processing with that image. For example, imagine you're a TA who wants to add a feature to your shader that works like the Soft Light filter in Photoshop. It composites two images together, subtracting values below 50% gray and adding values above it (see Figure 2). It's easy to say, but it's a lot harder to implement when you don't know what the gamma curve for the image looks like. Shaders work in floating point numbers, not bytes, so your color will be converted to something like .5, .5, .5. The shader can easily check the color of a texel and see if it's above or below .5 and add or subtract it as needed. This is pretty simple, until you ask your shader what set of numbers in that file on disk it thinks look like .5. Is it 128, 128, 128, or 187, 187, 187, or something else? And what happens if the texture you're looking into is created as a render target in a different render pass? Did the other pass write out values in gamma space or not? Similar problems arise with any form of pixel math, whether it's image blending, alpha compositing, recoloring, or tinting.

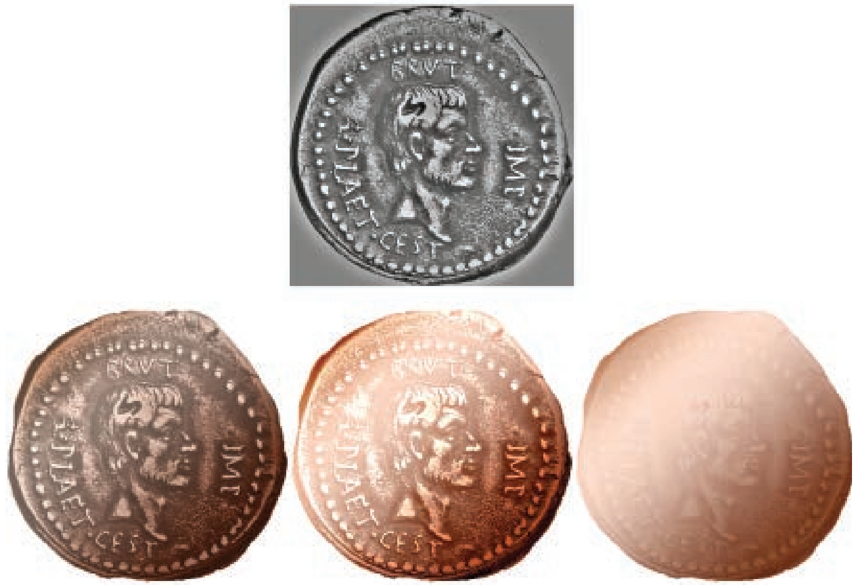


FIGURE 2. Gamma doesn't just affect textures or final images. Here the same image is applied as a MUL2X or "Soft Light" filter. The lower-left image applied the coin with 2.2 gamma and has almost no highlights. The lower right image applied the coin with a .45 gamma and has almost no darks. Shader artists and compositors need to make sure all their images are in the same linear space before doing any pixel math!

BEHIND THIS DOOR...

» Gamma also matters on the output side of the equation. Once you've done some color math, you need to remember to apply the right gamma on the way to the monitor, or else your rendered image will have messed-up midtones. Forgetting to apply gamma to the results of rendering calculations is a surprisingly common mistake that creates a distinctive “computery” look to renders that artists hate, but have a very hard time overcoming. Figure 3 shows a typical example of how lack of gamma on the output side softens the terminator of the diffuse lighting and flattens out the specular highlights.

Fortunately for us, correcting gamma problems isn't too difficult. Even more fortunately, most of the work involved will be done by graphics programmers, not us lowly artists. Still, it's useful to understand the basics so you can anticipate potential problems and advocate for a workflow that makes sense for your project.

The key to working with gamma is to get your colors to behave the way we all thought they did before the veil of secrecy surrounding all of this gamma stuff was lifted. You want to convert those dubious gamma-corrected color numbers into nice predictable linear numbers, where “twice as bright” translates into a number that's twice as big. That's how you can be sure that all the pixel math will work out correctly. Happily, gamma is a straightforward function: If you know the gamma used to create a particular image, you can always apply the inverse of that function to create a linear image. (It's handy to know that the inverse of the standard 2.2 gamma curve is a gamma of .455.) If you want to see this for yourself, you can apply a gamma curve to an image in Photoshop using the Exposure dialog (see Figure 4).

If you do try it for yourself, you'll probably notice that round-tripping an image from gamma 2.2 to .45 and back will cause banding. Gamma is really just a confusing special-purpose form of compression, allowing you to store an image that appeals to the eye in only 8 bits. To get an equivalent visual quality without gamma, you'd need somewhere between 12 and 16 bits per

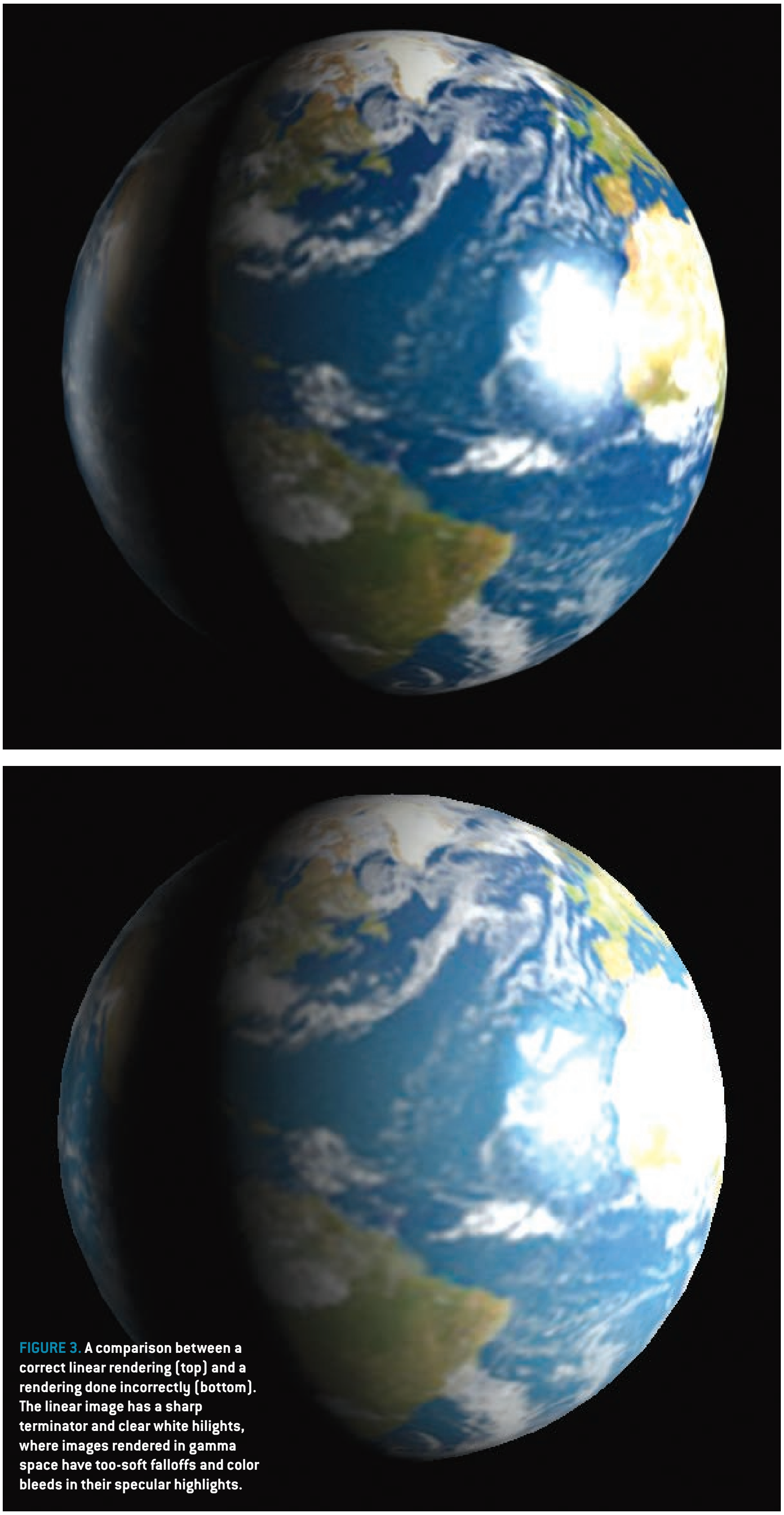


FIGURE 3. **A comparison between a correct linear rendering (top) and a rendering done incorrectly (bottom). The linear image has a sharp terminator and clear white hilights, where images rendered in gamma space have too-soft falloffs and color bleeds in their specular highlights.**

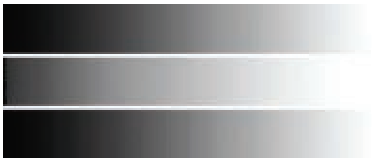


FIGURE 4. **You can use the Photoshop exposure control to see what a typical 2.2-gamma style ramp (top) would look like in linear space (middle). However doing the conversion in 8-bit colors is lossy and causes banding (bottom). If you need to manually linearize images, do it in 16- or 32-bit color.**

channel, depending on who you listen to and how sensitive your eyes are. So, if you manually linearize an image in Photoshop, keep an eye out for the loss of precision. That's why the hard work of creating a linear pipeline belongs to the graphics guys and gals, who'll have to actually write shaders that get everything into and out of linear space. Some fancy footwork will be required to balance the needs of color precision with the desire to save memory, but that's why those graphics engineers make the big bucks.

NO SMOKE, NO MIRRORS

» As an artist, the best contribution you can make toward keeping gamma issues from becoming a problem is simply to be aware of them. The critical thing is to make sure that the entire art team is working from the same playbook when it comes to color management. If your pipeline is expecting images to come in with a gamma of 2.2, you'd better make sure they're all in 2.2, or all those linearization calculations will be wrong. One of the most common sources of trouble is Photoshop's color profiles. If somebody on the team has tweaked their Photoshop for high-end photography or for working with a particular printer, they could be messing up files simply by opening them and resaving! Stick with default sRGB profiles, since OpenGL and DirectX both include easy methods for linearizing sRGB textures, unless your graphics team has given you a custom color profile, in which case make sure everybody uses that one. More than anything else, consistency is the key. Once you've got a standard and stick to it, you can forget all this gamma stuff and concentrate on your work.

In our post-"X-Files" universe, it often turns out that the murky secrets of the world just aren't that interesting. Unlike the Illuminati, the Knights Templar, or the Trilateral Commission, gamma isn't plotting world domination; it's just another one of those unsexy bits of computer arcana you need to know to survive as a game artist. Deal with it efficiently so you can get back to the good stuff.

STEVE THEODORE *has been pushing pixels for more than a dozen years. His credits include Mech Commander, Half-Life, Team Fortress, Counter-Strike, and Halo 3. He's been a modeler, animator, and technical artist, as well as a frequent speaker at industry conferences. He's currently the technical art director at Seattle's Undead Labs.*

THE INNER PRODUCT // ANDY FIRTH

LIGHTER THAN AIR

FLOATING POINT PERFORMANCE TIPS AND TRICKS

FLOATING POINT IS UBIQUITOUS IN PROGRAMMING THESE DAYS. It allows for relatively good freedom of representation when implementing many algorithms, and is both intuitive to set up and simple to work with. Hardware has also improved to where it is now actually faster to use floating point math as opposed to integer in many environments, which wasn't the case a decade ago.

This article will illustrate various tricks that can help push floating-point performance, usability, and relative accuracy even further, in the hopes that you might be able to avoid some of the pitfalls many experienced programmers, such as myself, have fallen into over the years.

We've all used them in our games and tools, so here is a question to ask yourself: What does the code below print out for the test variable?

```
    {
      float small_value = 1.0f;
      float smaller_value = 1 / 100000000.0f;
      float test = small_value + smaller_value;
      printf("{%g,%g} => {0x%08x,0x%08x}\n", small_value, test,
*(int*)&small_value, *(int*)&test);
    }
```

Almost all the programmers I spoke to assumed the value of the test to be 1.00000001, and of course mathematically it should be. However when represented as a single-precision floating point (32-bit) value, it is not. The printout would show:

```
    {1,1} => {0x3f800000, 0x3f800000}
```

The single precision floating point simply cannot represent the accuracy the result requires, and so the result is 1.0f.

If you are mathematically minded, this will likely poke at your OCD button and force you into using double precision. This is a perfectly tolerable solution in many situations where the program does not require the extra speed; however, double precision still suffers the same issues if you double the number of zeros in the denominator (1 / 10000000000000000.0f).

Math is important for those of us who work in games, but it's far less important than simulation determinism and getting things implemented expediently while executing within performance budgets. This forces us as programmers to make decisions that might otherwise be seen as somewhat mathematically incorrect, or even borderline wrong. The above is one such circumstance. In most games, we cannot afford to switch our math to double precision, due to the increased memory usage and hefty performance penalty. Therefore, our focus as game developers has turned in recent years to determining how we can best live with the quirks and limitations of single-precision math. I've been working with them so long that it is now natural to apply these limitations in all circumstances as par for the course.

Comparing floating point numbers using == is only sensible in a purely deterministic (read: constant) form. If any math is being used to generate the values to be compared, then great care has to be taken. For instance:

```
    {
            float a = 111.0001f;
            float b = 10000.01f;

            float c = a * b;
            float d = c / b;

            bool result = (a == d);
            printf("(0x%08x) %f %s %f (0x%08x) \n",
                        *(int*)&a,a,
                        result ? "==" : "!=",
                        d,*(int*)&d);
    }
```

One might expect the math above to show that a is equal to d given that the multiply and divide by b cancel each other out; however the output is:

```
    (0x42de000d) 111.000099 != 111.000107 (0x42de000e)
```

It's a sad fact that great care is rarely taken in practice, thus the general rule in most studios is simple: Don't compare floats using == on pain of death or public embarrassment. The usual method is to apply some form of epsilon to the comparison:

```
    {
      float a = 1.0f;
      float b = 1.0005f;
      static const float epsilon = 0.0001f;
      float temp = fabs(b - a);
      if(temp > epsilon)
      {
        // not the same
      }
    }
```

This is one method used to enforce determinism within games; apply a decent global epsilon and most math "behaves" (this doesn't mean it is accurate though). One complication, however, is that a good value for epsilon is not always obvious. Nor can it always be a constant, especially within helper classes such as vector math.

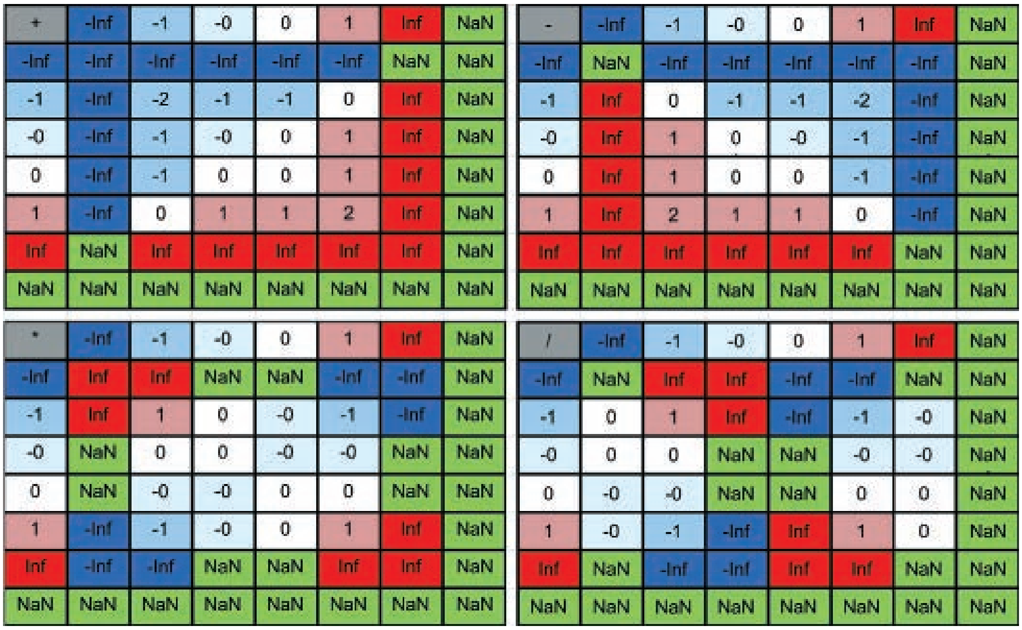
A good epsilon is usually dependent upon the data you're representing. If you are dealing in world coordinates, for instance where 1.0f = 1m, then:

```
    0.01   = 1cm
    0.001  = 1mm
    0.0001 = 100Δm
    0.00001 = 10Δm
```

Many of the games I've worked on use 1mm as their positional epsilon (0.001f), and 10μm for directional or rotational epsilon (0.00001f). This provides an effective range of approximately +/- 10000.001 (10km @ 1mm resolution) for positional epsilon, and +/- 1000.0001f for rotational epsilon.

The general rule of thumb I use is eight orders of magnitude between your highest and lowest accuracy requirements. If you extend beyond this you lose accuracy. If you feel you need a larger range but still want accuracy, consider using a reference frame; one value to represent low-accuracy high values (say kilometers) and another to represent high-accuracy low values (meters down to μm).

One area that seems to trip up a lot of people (myself included on many occasions) is Infinity and NaN rules, so to the right you'll find a handy table showing how various operations interact with and generate the so-called "special" values. NaN is a value representing an undefined or unrepresentable floating point value. One of its primary properties is that NaN != NaN. This can be relied upon as a basic check for NaN.



+	-Inf	-1	-0	0	1	Inf	NaN
-Inf	-Inf	-Inf	-Inf	-Inf	-Inf	NaN	NaN
-1	-Inf	-2	-1	-1	0	Inf	NaN
-0	-Inf	-1	-0	0	1	Inf	NaN
0	-Inf	-1	0	0	1	Inf	NaN
1	-Inf	0	1	1	2	Inf	NaN
Inf	NaN	Inf	Inf	Inf	Inf	Inf	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

-	-Inf	-1	-0	0	1	Inf	NaN
-Inf	NaN	-Inf	-Inf	-Inf	-Inf	-Inf	NaN
-1	Inf	0	-1	-1	-2	-Inf	NaN
-0	Inf	1	0	-0	-1	-Inf	NaN
0	Inf	1	0	0	-1	-Inf	NaN
1	Inf	2	1	1	0	-Inf	NaN
Inf	Inf	Inf	Inf	Inf	Inf	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

*	-Inf	-1	-0	0	1	Inf	NaN
-Inf	Inf	Inf	NaN	NaN	-Inf	-Inf	NaN
-1	Inf	1	0	-0	-1	-Inf	NaN
-0	NaN	0	0	-0	-0	NaN	NaN
0	NaN	-0	-0	0	0	NaN	NaN
1	-Inf	-1	-0	0	1	Inf	NaN
Inf	-Inf	-Inf	NaN	NaN	Inf	Inf	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

/	-Inf	-1	-0	0	1	Inf	NaN
-Inf	NaN	Inf	Inf	-Inf	-Inf	NaN	NaN
-1	0	1	Inf	-Inf	-1	-0	NaN
-0	0	0	NaN	NaN	-0	-0	NaN
0	-0	-0	NaN	NaN	0	0	NaN
1	-0	-1	-Inf	Inf	1	0	NaN
Inf	NaN	-Inf	-Inf	Inf	Inf	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Infinity and NaN interactions.

As the table shows, all operations involving NaN result in a NaN. This results in a phenomenon known as NaN propagation, where an operation inadvertently introduces a NaN into a data set. Any interaction with that data set will produce further NaNs until either the program detects a problem by explicit means, or an exception is thrown. Either way, it is often extremely difficult to figure out exactly what caused the NaN in the first place without significant defensive programming.

On the subject of error handling, consider the standard method of normalizing a 3D vector.

```
vector3 normalize_vector(const vector3 & vec)
{
  float length = vector_length(vec);
  vector3 result = vec;
  if(length > 0.0f)
  {
    float reciprocal = 1.0f / length;
    result.x *= reciprocal;
    result.y *= reciprocal;
    result.z *= reciprocal;
  }
  return result;
}
```

The usual problem with normalization is that vec may be of zero length (0,0,0), so the length may be zero. Dividing by zero generates infinity as the reciprocal, and multiplying the original vector by infinity under those circumstances generates (NaN, NaN, NaN) as the final result.

A branch is inserted to avoid the problem, which effectively removes the divide by zero. But it does so at great cost. On many platforms a branch results in an instruction cache flush, which is a very costly operation. Normalize is a very common operation in many game systems, which can result in significant performance issues. The problem is that there really isn't another way to avoid the divide-by-zero problems mathematically.

There is, however, a method of avoiding it if you rely upon floating point math and its representation limitation.

We've established that a large value remains the same when a small value is added; if those values are greater than eight decimal places removed, then the smaller value is ignored. We've also discussed that the effective range of floating point values for use in games is limited both for determinism and to avoid issues with the math itself. Combining the two provides a rather elegant method of avoiding the divide-by-zero issue under normalization and many other well-known situations. Consider:

```
const float very_small_value= 1.0e-037f;// ˜ FLT_MIN
vector3 normalize_vector_2(const vector3 & vec)
{
  float length= very_small_float + vector_length(vec);
  float reciprocal= 1.0f / length;
  vector3 result;
  result.x = vec.x * reciprocal;
  result.y = vec.y * reciprocal;
  result.z = vec.z * reciprocal;
  return result;
}
```

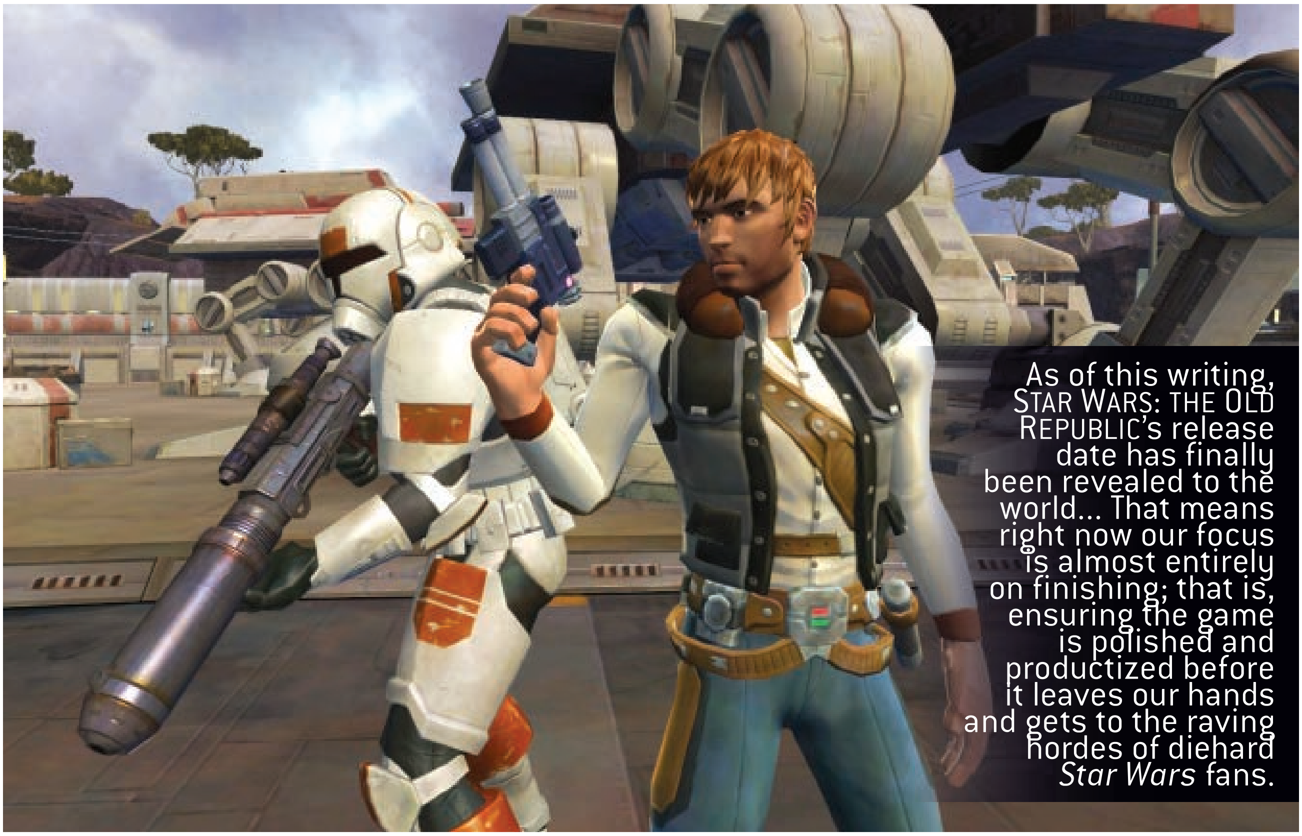
When vec is not (0,0,0), length will be within our accepted range (greater than 1.0e-29), thus the addition of FLT_MIN will have no effect on length. In the case when vec is (0,0,0) however, length will be equal to FLT_MIN and the resultant reciprocal will be very large. When this multiplies through, the result will be equal to (0,0,0) and the behavior of the original function is maintained without the branch overhead.

There is a plethora of information out there on floating point math, both practical and theoretical. The above is my attempt to represent those cases not covered or not well highlighted. Hopefully you can avoid the pitfalls I fell into.

ANDY FIRTH *is an engineer architect at Bungie in Seattle. He's been working on games for over 13 years in low level engine/rendering-related roles. Most recently he's been concentrating on infrastructure and workflow-related problems, especially where concurrency and multi-platform issues come into play.*

THE SENIOR DESIGNER MINDSET

MOVING UP AS A GAME DESIGNER



As of this writing, Star Wars: The Old Republic's release date has finally been revealed to the world... That means right now our focus is almost entirely on finishing; that is, ensuring the game is polished and productized before it leaves our hands and gets to the raving hordes of diehard *Star Wars* fans.

Every now and then, I am asked by a junior designer about how to move up the ranks in his organization and become a senior designer or a lead. There is no simple answer to this question as different designers at different organizations have wildly different day-to-day responsibilities and toolsets.

There are commonalities, though—things that are true no matter whether you're a world builder, a systems designer, or a game writer. They are core personality traits focused on leadership, teamwork, and quality. While they don't belong in game design theory books, they are crucial to the success of large software projects and are among the most important traits that I look for when interviewing potential design candidates.

BE A FINISHER

» When I give lectures to designers just starting out in the industry, they often want one piece of advice they can use in their career. They frequently seem disappointed when I respond with "finish and polish," but then again, they don't see all the design submissions I see—sloppy, unpolished, and speaking of huge ideas while demonstrating little follow-through. Ideas are a dime a dozen. Execution is what it's all about. A flawless 60-second demo is immensely more impressive to me than a 30-minute one with broken quests, untextured walls, and typos in the dialogue. Tragically, I see a lot more submissions like the latter.

As of this writing, Star Wars: The Old Republic's release date has finally been revealed to the world, generating a response of general jubilation from our fans. That means right now our focus is almost entirely on finishing; that is, ensuring the game is polished and productized before it leaves our hands and gets to the raving hordes of diehard *Star Wars* fans. It has to reach a quality level that we can be proud of. We've actually been in finishing mode for months, long before our release date was public. Part of what allows heavyweights like BioWare, Blizzard, and Valve to release such great games is their focus on ensuring they're proud to put their brand on their end products. Entry-level designers should strive to demonstrate that philosophy in their own work.

As a lead on a project, I'm insanely busy, so the designers I value the most are the ones who make problems disappear. If I give them a problem, they drive it into the ground. If someone files a bug, they not only fix the issue but also verify that the issue isn't systematic and fix other easily found variants of the issue. If they need art, they go and get it—through properly approved channels, of course. Designers who take on responsibility and then knock it out of the park will almost always be recognized and given more.

PLAN FOR ITERATION

» No design idea is perfect. Many ideas that are great on a whiteboard fall apart once they actually become tangible. Some ideas end up being awesome but inappropriate for the game, and sometimes, features fight each other. For instance, GEARS OF WAR was supposed to have more vehicle combat and squad control, but the team found that these features were actually impeding on the cover gameplay, which they quickly realized was their bread and butter. The game shipped with only light squad control and vehicle combat.

It's imperative that designers get their feature to iteration right away. The feature needs to be implemented and in the hands of people playing it as soon as possible. Get people to play it, watch them, take notes, and make changes. For the purpose of fact-finding, a pretty good prototype now is far better than a "perfect" implementation three months from now. Getting that pretty good implementation will help you understand whether the feature is on the right path, and may uncover (and force you to face) design roadblocks sooner than you otherwise would have.

Of course, this means that features need to be implemented in two phases: an initial phase, and then an iteration phase, where you apply what you've learned. Working with project managers to make this happen is both crucial and challenging, as the time you need can vary wildly. The feature may just need minor tweaks or require a lot of time in order to reach the perfect version that you originally spec-ed. You'll often figure out that the design needs to go in a different direction altogether, and sometimes, you'll discover that a beloved feature needs to die.

KILL YOUR OWN IDEAS

» A producer once told me that the mark of a senior designer to him was that designer's willingness to kill his own babies—to abandon his own ideas once the design, for whatever reason, begins to move in another direction. It is a sign of great maturity to declare that you were wrong and that something needs to be axed. Even more so if the resulting cut will create a void the designer then has to fill with his own sweat.

At BioWare, we consider humility one of our core design principles, and the result is an environment where ideas are fertile, and the bad ones are dispensed with in a very Darwinian way. Bad ideas can waste time, but they can also provide the team with valuable lessons about where it needs to go. Good ideas are even harder to kill when they're not needed, such as when they detract from some other aspect of the game, or play awesomely but require more content than you can possibly finish in the time you have.

KEEP YOUR PERSPECTIVE

» One of the more interesting discoveries of my career is how distant those lost features, and the fights over them, look in retrospect. I can remember arguments we got into in the early days of developing STAR WARS: THE OLD REPUBLIC. In many cases, the people pounding their fists on the conference table back then don't even remember what side they were arguing anymore. In some cases, the same design topics would reopen, and people who argued one stance or another would have switched sides, because of how the game had evolved.

With the benefit of hindsight the amount of energy and vim put into design discussions in the early days seems almost comical. Over the long course of the project, nearly all of these topics were reopened at one time or another, and in most

cases, the right decision won out in the long run based on how it actually played in the game.

VALUE DATA

» Some designers fear feedback. Designers that fear feedback are afraid that they're going to be discovered as frauds or be proven wrong, and this general tendency is the mark of a junior designer to me. They are ashamed that they might be called out on their work, challenged on their thinking, or asked to redo something.

Senior designers crave feedback. They understand that their idea isn't perfect, and that the only way for it to get better is to get hands on it. Good designers get on the forums and see what the fans are saying; they want to watch focus group players use the GUIs they've designed; and they ask for real metrics on how a particular feature is performing. Great designers have skin thick enough to keep on doing it, no matter how much their design is getting savaged.

TRUST YOUR SPIDEY SENSE

» It's sometimes hard for designers to have perspective about whether one of their designs is really good. However, most quality designers I've found have a really good instinct for when a design is bad. They have a hesitant, uncertain feeling when they encounter and play with a feature that isn't working out—a voice in the back of their minds that says, "Something's not quite right here!" In almost every instance of my career where I've ignored that feeling, I've come to regret it later.

Good designers don't let that feeling linger. Instead, they confront it. Find other people to bounce your concerns off of, and try to actually quantify them on a piece of paper. Make those concerns tangible and put them before the eyes of someone who can do something about it. Whoever is designing that system is, much like you, hopefully eager for feedback, provided it's presented constructively and actionably.

THE MINDSET OF DEVELOPMENT

» It's all good and well to be a wizard at UnrealEd, a savant with Excel, and a professor of game design theory, but actually putting a game on the shelves requires a lot more than the nuts and bolts of putting together gameplay. Senior designers have all the basic skills, and go beyond.

Only a top-notch product has a chance to compete in the crowded triple-A marketplace, and these designers would do anything to get their games to that level of polish. They also understand that fun is an elusive concept, and that the uncertain practice of iterating to find that fun is both necessary and limited by production schedules. Designers who can't manage this will always feel they can't move up that ladder—and to be honest, they probably shouldn't. But those who can master these real-world production challenges will likely find that the sky is the limit in terms of their own professional success. gd

*DAMION SCHUBERT is the lead systems designer of STAR WARS: THE OLD REPUBLIC at BioWare Austin. He has spent nearly a decade working on the design of games, with experience on MERIDIAN59 and SHADOWBANE as well as other virtual worlds. Damion also is responsible for Zen of Design, a blog devoted to game design issues. Email him at **dschubert@gdmag.com**.*

IGF China 2011 Announces Main Competition, Student Finalists

\\\ The Independent Games Festival China has announced the Main Competition and Student finalists for its third annual awards ceremony celebrating the most innovative indie and student games from throughout the Pan-Pacific area.

This year, the finalists offer an extremely broad range of game types and genres, from action brawlers like Pixel May Cry to mobile arcade titles like Super Sheep Tap, with developers hailing from throughout China and its surrounding regions.

Drawing from a prize pool totaling 45,000 RMB (roughly $7,000), IGF China's Main Competition will give away five distinguished awards: Excellence in Audio, Technology, and Visual Arts, as well as Best Mobile Game and Best Game. In addition to the prestige and prizes, winners will also receive two all-access passes for the upcoming GDC 2012 in San Francisco.

Alongside IGF China's Main Competition, the ceremony will also host the Student Competition, which honors six of the top regional student games, with teams hailing from DigiPen Singapore, the China Central Academy of Fine Arts, and others.

This part of the competition includes two awards—Best Student Game and Excellent Student Winners—and offers roughly 13,000 RMB (around $2,000) in cash prizes.

Winners in both competitions will be chosen by a panel of expert jurors, including Kevin Li (CEO, TipCat Interactive), Monte Singman (CEO, Radiance Digital Entertainment), Xubo Yang (director of digital art lab and assistant professor at Shanghai Jiaotong University's School of Software), Haipeng Yu (producer, Tencent Shanghai), and jury chairman Simon Carless, IGF Chairman Emeritus and EVP of the GDC shows, Gamasutra, and *Game Developer* magazine.

This year's IGF China will take place on November 12, 2011, alongside GDC China. Here are the finalists:

MAIN COMPETITION

- Billy Makin Kid!, by SLAB Games, Indonesia
- Clay's Reverie, by SuperGlueStudio, China
- FTL (Faster Than Light), by Matthew Davis & Justin Ma, China
- One Tap Hero, by Coconut Island Studio, China
- Pixel May Cry, by Feng Li, China
- Pocket Warriors, by WitOne Games, China
- Super Sheep Tap, by aBit Games, China
- The Line HD, by Ant Hive Games, China

STUDENT COMPETITION

- Nanobytes, by Singapore Polytechnic School of Design Splat Studios, Singapore
- Pixi, by DigiPen Institute of Technology, Singapore
- Robotany, by Singapore-MIT GAMBIT Game Lab, Singapore
- Shadow Fight, by China Central Academy of Fine Arts, China
- Terra: The Legend of the Geochine, by DigiPen Institute of Technology, Singapore
- Void, by DigiPen Institute of Technology, Singapore

In addition to the awards ceremony, GDC China will also host its own dedicated Independent Games Summit, which offers a host of lectures covering some of the most pertinent issues in independent development, featuring speakers such as thatgamecompany's Jenova Chen, Supergiant Games' Amir Rao, and more.

GDC China Reveals Mobile Talks On Doodle Jump, Gamevil, Fruit Ninja

\\\ GDC China has debuted new talks within its Mobile Games Summit, featuring an in-depth look at Lima Sky's Doodle Jump, a breakdown of Gamevil's strategy for global success, and a glimpse at Halfbrick's plans to develop the Fruit Ninja IP.

The event, which takes place November 12–14 at the Shanghai Convention Center in Shanghai, China, will once again serve as the premier game industry event in China, bringing together influential developers from around the world to share ideas, network, and inspire each other to further the game industry in this region.

This year, the show will feature two summits in addition to the main conference, covering independent games and mobile games.

The following are the latest lectures to be announced for GDC China's Mobile Games Summit:

Igor Pusenjak, creator of the iOS smash hit Doodle Jump, will offer an in-depth look at the game's development and success in "Doodle Jump—The Story Behind the Legend."

Puisenjak will explain how he and his team at Lima Sky have used frequent updates, direct player communication, and a number of social networking tools to keep this arcade platformer near the top of the App Store charts.

Next, Brian Oh of Korean mobile game publisher Gamevil (Air Penguin, Zenonia) will host a talk dubbed "Sharing with Developers the Vision and Ideas to Achieve a Successful Global Mobile Game," in which he will address the company's approach to global game distribution. Oh will detail how Gamevil works with its developers to share ideas and ensure that each product can succeed in Korea as well as other global markets.

Finally, Halfbrick CMO Phil Larsen will discuss the growth and evolution of Fruit Ninja in "The Rise and Rise of Fruit Ninja: Developing, Marketing and Supporting a Hit Mobile Game!" Reflecting on the hit title that put Halfbrick on the map, Larsen will examine the marketing and growth of the Fruit Ninja IP that allowed the studio to bring the title to new platforms and new global markets.

For more information on these or other sessions, please take a look at the official GDC China web site.

With registration for GDC China now open, interested parties can go to the event's official website (www.gdcchina.com) to start the registration process and gain access to the numerous talks, tutorials, and events the show will have to offer. Keep an eye out for more news as the show draws closer. GDC China is owned and operated by UBM TechWeb, as is *Game Developer*. gd

DIALOG DIARIES

MAKING THE MOST OF VOICE ACTORS' SCRIPTS

As the majority of our development tools undergo a continual process of refinement, one tool that we regularly use has actually remained fairly constant for thousands of years. Since the days of ancient Greece, dialogue has been communicated from author to performer via a script.

In game development, there is little standardization of script formats. Game scripts can be indistinguishable from cinematic screenplays, or as simple as lines scribbled on the back of an envelope. What is universal to all game scripts, though, is their audience. These scripts are tools used by three distinct roles within the voice production pipeline. To break down the ways in which these different roles make use of script, I spoke with voice directors Darragh O'Farrell (STAR WARS: THE OLD REPUBLIC) and Julian Kwasneski (BACK TO THE FUTURE: THE GAME).

THE ACTOR

» "As far as I'm concerned," says O'Farrell, "we're in the studio for one thing, and that's to get a performance out of the actor." For him, everything else is superfluous. The script, its formatting, and even the director are solely there to serve this main purpose. "Any other extraneous noise needs to go away."

When O'Farrell mentions noise, he's talking about visual noise on the script's page. The actors' script has a relatively small amount of information that needs to be communicated. "They want to see the setup for the scene; they want all of their lines in context; and they're going to want any direction notes that are applicable." Kwasneski agrees: "The talent script should be nice and legible, uncluttered, and easy on the eyes. I don't like to overload the actor with data, so their script only shows the feeding line—if one exists—and a bit of inflection notes like 'shouting to be heard over a battle' or 'whispering to avoid discovery.'"

For actors, the script's format is geared primarily toward fast, concise communication and familiarity. Questions from the actors in the booth cost time and money. As such, a cinematic script in the style of a movie's screenplay "is the visual language that they are used to dealing with day in and day out," says O'Farrell. "For me, there's nothing better than a film-style script." A screenplay-formatted script is best suited to in-ear conversations and cinematics.

Branching dialogue and AI barks, though, are harder to format as a traditional script and are often presented to actors as Excel documents with detailed line information. Regardless of the format of the actors' script, the goal always remains the same. "Where games are going with regard to interactive conversations, there's a lot of complexity in terms of how things are formatted," says O'Farrell. "It has to be more modular, but it has to retain the same appropriate information."

THE DIRECTOR

» The director is essentially a script's translator. It's the director's job to make sure that the writer's intent is translated into audio content that best serves the needs of the game's development team. The director also serves as the middleman between the actor and the voice editor. As such, the director's script needs to reflect the needs of these other two roles. "We have two script formats: One the actor sees and one the director sees," says Kwasneski. While the actors' script is uncluttered and concise, the director's script contains detailed notes for both actor and editor. "We like to have all feeding lines in place and will always read them to an actor in the correct inflection," continues Kwasneski. "If a line is to be whispered, we whisper it. If shouted, we shout. Another important factor is the characters' proximity to each other or the player. Are they across the room? Up on a ledge? Six feet away? It all makes a huge difference. The more an actor knows, the more realistic the performances will be."

Sometimes the information a director needs shouldn't be shared with an actor. "The director may have private notes," says O'Farrell, "particularly if you're doing pick-ups and redoing scenes. You may have private notes like 'The acting was weak in this middle section.'" This kind of comment can be unnerving for the talent who may get self-conscious about their performance, but can be important for a director to know exactly why a line is being rerecorded.

THE EDITOR

» A director's script also needs to contain an area for detailed notes on which take is best. These takes, called selects, are the blueprint for the voice editor's work. As Kwasneski explains, "The director script is laden with extra bits of information as well as space to notate takes and other editorial instructions," allowing the director to specify enumerated selects for the editor, or include detailed performance notes such as false starts or truncated performances.

"Nobody cares about the filenames but the editor," says O'Farrell, who recommends formatting the filename associated with each line in a lighter-colored font so as not to clutter page real estate while retaining what is critical information for the editor.

"Any system needs to have a certain amount of flexibility," sums up O'Farrell. "Each game is going to ultimately be different and require different layouts, different pieces of information, and serve different purposes." Remembering which role of voice production needs which information out of a script will keep your sessions moving smoothly.

JESSE HARLIN *has been composing music for games since 1999. He is currently the staff composer for LucasArts. You can email him at* ***jharlin@gdmag.com.***

Borne Again

KRISTEN BORNEMANN MOVES FROM ENGINEER TO PRODUCER, MS TO WB

Kristen Bornemann was an engineer at Microsoft, shipping Alan Wake and Kinectimals before moving to WB Games to become a producer. There, she specializes in interfacing between the core tech team and content—but in the evenings, she's a full-fledged indie developer, helping indies with their metrics, tech, and everything else.

Brandon Sheffield: What prompted the move from Microsoft to WB?

Kristen Bornemann: Originally at Microsoft I was writing test automation for Office. When I got hired on at Microsoft Studios, my role was more project management rather than the traditional SDET (Software Designer Engineer in Test) role at Microsoft. All the coding being done was for test tools, and we had a dev division that did most of that work for us. So I found myself spending most of my time managing a group of testers and interacting with the production team and internal/external dev studios. At some point, I realized that I really enjoyed managing a project, and I wanted to do more of that. I looked at what roles in the industry allowed for that opportunity, and the producer track just made sense. Going to WB (formerly Monolith) also allowed me the opportunity to be at a real game studio instead of a publishing house. The difference is remarkable, and I'm definitely glad I've had the opportunity to experience both (but that's another conversation). Also, I work directly with the core tech team (engine, graphics, and so forth), so I get the opportunity to spend all of my time with developers.

***BS:* What are the challenges involved in transitioning from engineering to production?**

KB: For me personally, the challenge has been switching from a role where I had direct impact on the game to a role where I'm facilitating those that have an impact. It can definitely be difficult to—excuse the football analogy—coach from the sidelines and not be on the field. Also, there's a lot of distrust of production in the engineering community. I like to say that I'm on a never-ending quest to prove to devs that producers can be valuable.

***BS:* Do you get to keep up your coding/engineering chops on the indie side?**

KB: Certainly! I wear all non-content-creation hats. We have dedicated programmers on the team, so I know to stay away from some of the more complicated features, but on my last project, I spent most of my time coding. Presently, I'm moving toward data analysis and putting in features to monitor and capture data. I see myself doing more of that in the future (instead of, say, implementing physics).

***BS:* You seem to be very interested in and dedicated to independent development. Why not do it full time?**

KB: I have no doubt that eventually I will be doing indie development full time. However, I'm in a great position now where I can test out a lot of ideas while not incurring the risk of full-time indie development. I want to ship a few more games where I can test some of my theories (in terms of monetization, marketing, and other aspects that are difficult for indie devs) and truly figure how to be successful without worrying about where next month's rent is going to come from.

***BS:* How do you balance work/life when you have a day job and night job that are in the same field?**

KB: It helps that my roles during the day and night are different, so I'm never doing the same thing at home as I was at work. The problems I am solving are also very different. I actually feel like I would be less balanced if it weren't for my indie work in the evening. Eventually the goal is to meld the day and night job, but in the meantime it's a fairly balanced setup.

who went where

Electronic Arts has appointed former Microsoft executive Rajat Taneja as its global chief technology officer, bringing with him 15 years of experience on products including Xbox LIVE and Microsoft Office.

Daniel Stahl, executive producer at Cryptic Studios and responsible for popular MMORPG Star Trek Online, has left the company to "pursue new challenges."

Mobile social gaming network OpenFeint co-founder and CEO Jason Citron has resigned from the company, with parent company Gree's Naoki Aoyagi stepping in to fill his role.

Social developer RockYou (Zoo World, Galactiv Allies) has announced that the company's senior vice president of games, Jonathan Knight, has left the company to join FarmVille developer Zynga.

new studios

Moscow-headquartered game publisher and developer Nival announced the founding of a new Kiev, Ukraine-based studio that will focus on the burgeoning mobile gaming market.

Mark Jacobs, formerly CEO at Warhammer Online developer Mythic Entertainment, has founded City State Entertainment, a new studio that will focus on developing casual games.

Video game talent agency Digital Development Management has announced a new publisher start-up, Red Stallion, which will focus on delivering high-quality games to the Middle East.

BioWare and parent company Electronic Arts today announced the opening of BioWare Ireland, a new state-of-the-art facility that will serve as a global customer service center for upcoming MMO Star Wars: The Old Republic.

David Darling, one of the cofounders of UK-based developer Codemasters, has founded a new studio called Kwalee that will focus on producing smartphones games and apps.

http://itbelongsinanancientruin.com

IT BELONGS IN AN ANCIENT RUIN

IT BELONGS IN AN ANCIENT RUIN OFFERS A TONGUE-IN-CHEEK SPIN ON INDIANA JONES, TASKING PLAYERS WITH STEALING ANCIENT ARTIFACTS FROM MUSEUMS TO RETURN THEM TO THEIR RIGHTFUL PLACE WITHIN SOME TREACHEROUS, UNDERGROUND CAVERNS. WE TALKED TO THE DIGIPEN STUDENTS BEHIND THIS STEALTH/ACTION PLATFORMER TO GET A BETTER LOOK AT THE CHALLENGES AND SUCCESSES THEY EXPERIENCED WHEN WORKING ON THE GAME.

***Tom Curtis:* How did you arrive at the overall concept for the game? Did the premise come first, or was it mostly gameplay driven?**

Grant Wynn, assistant producer: The overall concept for the game was fleshed out over time, but the original spark of an idea that snowballed into a game occurred when we were watching one of our friends play MINECRAFT. He accidentally placed a valuable block underground, due to some confusion over the controls, and I had to rub it in a little. I jokingly said, "What is this, reverse Indiana Jones? You're taking the valuables back into the ancient ruin?" Then came the fateful line that started it all from another friend of the group. Jared blurted out while using his best Indiana Jones impression, "It belongs in an ancient ruin!" One of my teammates turned and said, "Dude, game idea..." and so the game was born.

***TC:* What tools did you use for the project?**

Ansel Rognlie, technical director: We used OpenGL for graphics, FMOD for audio, SDL for keyboard and mouse events, as well as for wiring up OpenGL to Windows, XInput to support Xbox 360 controllers, and TinyXML for XML parsing.

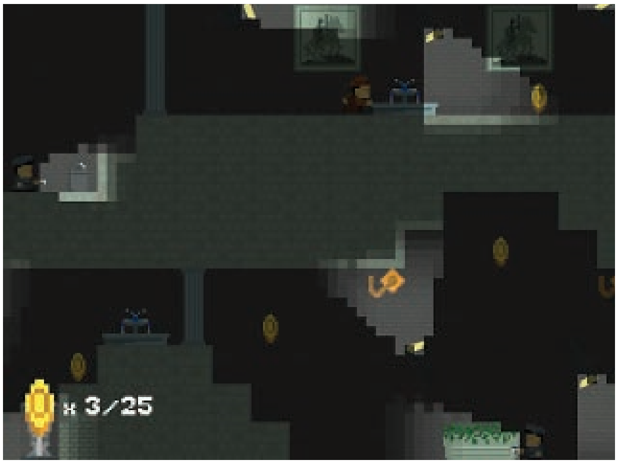
For the game itself, we had to write the entire engine ourselves in C++. This entailed setting up and managing all the graphics and sound resources, game objects, tile systems, physics, gameplay, UI, messaging—basically everything that wasn't provided by the low-level libraries.

David Evans used GameMaker for prototyping, and I built a level editor in C#. Our art was done in Photoshop and Paint.NET, and our sounds and music were predominantly composed using MilkyTracker.

***TC:* What would you say were the biggest challenges you faced during development?**

AR: One of the biggest issues that we encountered was when we realized that we needed to switch from a more physics-based engine to one that was tile-based. Our initial concept had involved traversing asteroid fields in space, and hence, a more object-focused level design with general rigid body physics for collision. When we fully decided to go with RUIN, it was obvious that tile-based collision and physics were what we needed. This required ripping out the entire physics and collision systems and reworking them in conjunction with a new tile graphics system toward the end of the fall semester. However, moving to the tile system also enabled us to incorporate a full lighting system rather than using statically placed shadow regions for hiding. Also, due to the late point at which we switched to a tile solution, I had to begin working on our level editor before the level formats or any C++ code for it had been written. To get this up and running quickly, I decided to use C#, and to only introduce a dependency on the level file format, rather than requiring the use of any engine code for the editor.

An ongoing problem we had throughout development was the clash of strong personalities. We all had strong ideas about how the parts we were developing should work, and the direction that the game should take. While some minority opinions were incorporated into the final game, on the whole, I would say that we didn't have a good process for soliciting group feedback, or reaching effective compromise. A few voices tended to dominate. Perhaps if we had had more time we could have been more receptive to trying out different approaches, but as it was, we were constantly scrambling just to refine what he had, trying to figure out how to incorporate our playtesting results.

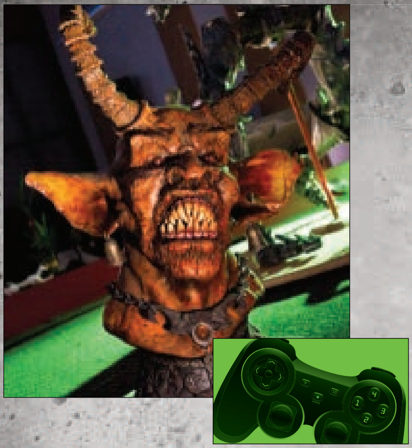


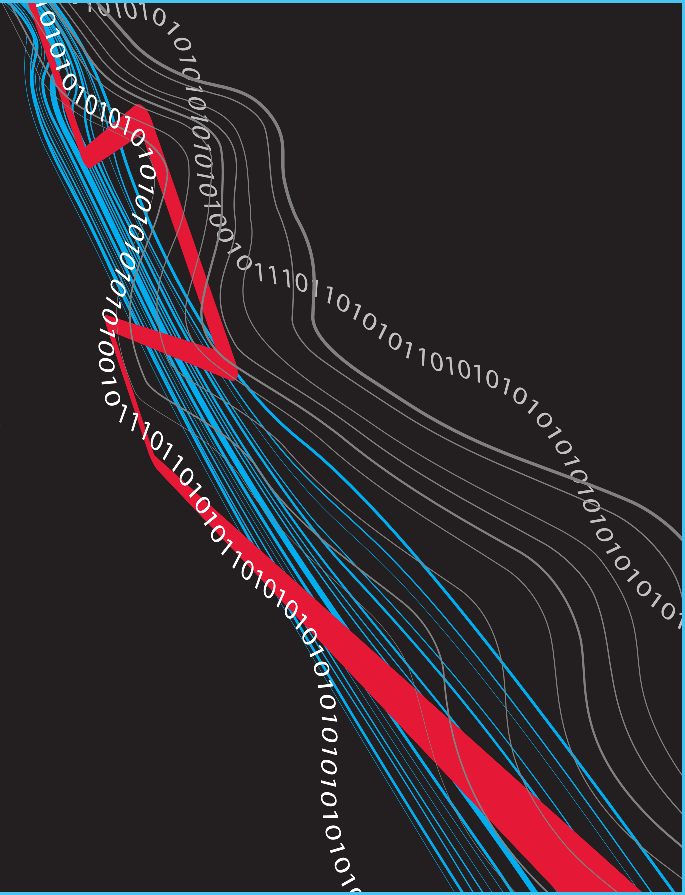
Nathan Carlson, producer: Time management with the other five classes during each semester was tough. When working on such a large-scale project during school, where everything else is about a one-week assignment, it's hard to measure how much progress needs to be made on the game each week. And it's not just scheduling the time to code for the game, you also have to make time for any necessary research to learn how to do whatever it is you're writing. The hardest thing to learn on this project was that letting just one week go by without working on it would effectively lose us two weeks of progress.

***TC:* If you could have done one thing differently during the course of this project, what would it have been?**

AR: One difficulty with which DigiPen game teams are often faced is scope. Our initial game idea had scoping issues that we were never able to condense down to a workable game idea within the timeframe. Even the first pitch for RUIN was very far-ranging in varying game play, level structure, puzzles, and so forth. We hear this over and over again from our instructors, but somehow we always wind up overestimating our output, or underestimating the amount of work required. I think the game would have reached a more polished state if we had started with a more narrowed concept, then built out. As it was, we were still able to bring in some of the initial concepts that we had thought we might have to cut, but I feel like we'd have been able to incorporate more had we started out with the same focus with which we finished.

© 2011 Full Sail, LLC





ADVERTISER INDEX

gd Game Developer (ISSN 1073-922X) is published monthly by UBM LLC, 303 Second Street, Suite 900 South, South Tower, San Francisco, CA 94107, (415) 947-6000. Please direct advertising and editorial inquiries to this address. Canadian Registered for GST as UBM LLC, GST No. R13288078, Customer No. 2116057, Agreement No. 40011901. **Subscription Rates:** Subscription rate for the U.S. is $49.95 for twelve issues. Countries outside the U.S. must be prepaid in U.S. funds drawn on a U.S. bank or via credit card. Canada/Mexico: $69.95; all other countries: $99.95 (issues shipped via air delivery). Periodical postage paid at San Francisco, CA and additional mailing offices. **Postmaster:** Send address changes to Game Developer, P.O. Box 1274, Skokie, IL 60076-8274. **Customer Service:** For subscription orders and changes of address, call toll-free in the U.S. (800) 250-2429 or fax (847) 647-5972. All other countries call (1) (847) 647-5928 or fax (1) (847) 647-5972. Send payments to *gd Game Developer*, P.O. Box 1274, Skokie, IL 60076-8274. Call toll-free in the U.S./Canada (800) 444-4881 or fax (785) 838-7566. All other countries call (1) (785) 841-1631 or fax (1) (785) 841-2624. Please remember to indicate *gd Game Developer* on any correspondence. All content, copyright *gd Game Developer* magazine/UBM LLC, unless otherwise indicated. Don't steal any of it.

DEVELOPER NEWS NETWORK!

KEEPING YOU UP TO DATE ON MIDDLEWARE AND MORE

This month, let's take a look at what's going on in the exciting world of news for video game developers!

LINEARSOFT DEBUTS CRATE CREATOR AND CRATE CREATOR PRO

» LinearSoft, LLC takes the pain out of creating great-looking game environments with its new Crate Creator and Crate Creator Pro products. Using these new tools, creative professionals in the game industry can now create crates at a pace never seen before.

The $7,995 Standard edition supports traditional wooden crates. The Pro edition (call for pricing) adds to the package with support for metal crates as well as blue and yellow "glowy bits" for crates intended for sci-fi universes. The Pro edition also allows crate creators to share and swap their crate creations online through the Crate Creator Crate Store.

"Crate Creator takes the drudge work out of making crates, so that developers can focus on what they do best: placing those crates in their levels," said Robert Ishikawa, president of LinearSoft.

Crate Creator Pro is part of a full suite of tools for authoring digital content from LinearSoft, which includes Barrel Creator and Barrel Creator Pro, Warehouse Designer 3, and Shipping Container Simulator EX.

GUY IN HIS BEDROOM SELECTS MICROSOFT'S NOTEPAD FOR TASK TRACKING

» Microsoft Corporation today announced that a guy in his bedroom has selected its powerful Notepad software application as his exclusive project management solution across multiple game titles currently in development.

"I performed a thorough survey of the competitive landscape, looking at all of the options available. What I found was that Notepad offers what I need: the right features at the right price," said the guy. "This multi-year, multi-project agreement with Microsoft now gets me access to the tools I need to closely track tasks and bugs, schedule production, and manage dependency chains. And I love that issues can be edited in real time!"

"Our Notepad software leads the industry in market share precisely because it has features that no other program out has yet duplicated," said Gordon Lang, senior vice president of the Notepad business unit at Microsoft. "Our commitment to continue pursuing this lucrative market is underscored by our massive R&D investment in improving Notepad as well as aggressive exclusivity agreements such as these. Other task and bug tracking software: you've been warned."

NEW TECHNOLOGY TO REVOLUTIONIZE GAMES

» Norbert McCree International is set to revolutionize the industry with the new InfiniPoly engine, which allows an infinite number of polygons, with arbitrarily infinite detail, to be rendered at a speed of infinite frame-rate. Developed single-handedly by Norbert McCree, a respected contributor to the xX_Ultra_Xx Gamedev forums and the alt.graphics.bryce Usenet group, InfiniPoly represents a giant leap forward in real-time rendering and visualization.

"I have been working on InfiniPoly technology for over 10 years, basically since I was seven years old," said McCree, "It's too complex to describe in a press release, but it's a completely new approach to graphics rendering that combines fractals, wavelets, and quantum entanglement—oh, and a dash of simple arithmetic, too. The holy grail of video game technology is here!"

Norbert McCree International is actively seeking investment in order to put the final touches on the engine and begin marketing and distribution. Please see the PayPal Donate button on our web page for more information.

MOCAP LIBRARY RELEASES DUDES STANDING AROUND, VOLUME 4

» The highly regarded Mocap Library series of library motion capture animation has expanded again with the release of Dudes Standing Around, Volume 4!

Volume 4 will find a home in any stealth-themed game as it includes oft-requested idle animations, such as dudes shifting their weight from their left foot to the right foot and back again, dudes staring straight ahead and then slowly looking to the left and to the right, and dudes who are rolling their right shoulder.

SURFACE OF THE SUN OFFERS INTERACTIVE MEDIA INCENTIVES

» Joining regions such as Quebec and Louisiana, the surface of the Sun today announced tax breaks and a development-cost refund program for interactive media developers in a bid to attract more high-tech jobs to this G-type main sequence star.

"Interactive media is one of the most exciting growth areas for us," said a representative of the Sun. "The new techniques developed for entertainment technology have wide-ranging applications, from education to public policy to health. Additionally, these high-paying jobs benefit the Sun's economy as a whole and will keep innovation close to home."

The surface of the Sun is a compelling environment in which to develop video games, he argued, with its temperature of around 10,000 degrees Fahrenheit and constantly changing "boiling" pattern as superheated gasses from the core rise through the photosphere and cool down. Occasional solar flares release unbelievable amounts of radiation across the entire electromagnetic spectrum.

"I hope that game developers will seriously consider the clear benefits of the Sun," concluded the representative.

MATTHEW WASTELAND *writes about games and game development at his blog, Magical Wasteland (www.magicalwasteland.com). Email him at* ***mwasteland@gdmag.com****.*

ADVERTISEMENT

UNREAL TECHNOLOGY NEWS

UT3 IN REAL-TIME FLASH

UNREAL ENGINE 3 SUPPORTS FLASH

On October 4 in Los Angeles, Epic Games Founder and Technical Director Tim Sweeney appeared onstage at Adobe MAX 2011 to demonstrate Unreal Engine 3 running in Flash.

We've been working closely with Adobe on this technology for quite some time. A few months ago when we decided we would do a demo for this event, we weren't sure what we were going to show.

The first content we decided to try in Flash was *Epic Citadel* and it ran amazingly well – better than we expected it would, considering how early on this was. But we began thinking that maybe a demo of content designed for mobile was setting expectations too low and we should aim higher.

So what did we do? We chose as our demo a fully playable level from *Unreal Tournament 3*, and it turned out to look even better than the version we shipped on Xbox 360 and PlayStation 3, with improvements like global illumination, better shadows, and god rays. We're not just talking about triple-A console quality on the Web, we're actually showing it onscreen, in a Web browser, playing inside Flash.

I can't blame you if you couldn't imagine a Facebook game at the level of *Unreal Tournament 3* before today. But know now that UE3 with Flash support is the technology that will enable experiences like this, and we're just getting started.

UE3 has earned recognition as the best game engine for PC, console and mobile platforms, and now we're adding the Web via Adobe Flash support. With many of the world's best developers using UE3's professional-strength tools, we're sure to see amazing uses for this down the road.

There's still some work to do before we can release this technology to developers, and we'll have more to talk about soon. We plan to continue working closely with Adobe, and the long-term goal is to be able to bring amazing *Samaritan*-like experiences, and beyond, to Web browsers through Flash.

INFINITY BLADE 2 ANNOUNCED

Onstage at Apple's "Let's Talk iPhone" event the very same day, Epic's president, Mike Capps, and Chair Entertainment Creative Director, Donald Mustard, revealed and demonstrated *Infinity Blade 2,* the full-blown sequel to the hugely successful iOS game of the year, *Infinity Blade.*

Infinity Blade 2 uses some cool new technical features that demonstrate what developers are able to do with UE3.

Let's start with two high-end graphics features that take advantage of the latest A5-equipped iOS devices (iPad 2 and the new iPhone 4S). Dynamic character shadows, which include self-shadowing, allow for greater realism in both combat and cinematic sequences, and provide greater range of visual contrast and color depth. Dynamic light shafts, also known as "god rays," enhance the visual appeal of outdoor areas, as well as allow for great realism and cinematic quality together with visual effects such as lens flares.

iOS 5 introduces Apple's new iCloud service, and *Infinity Blade 2* will take advantage of iCloud save games, which will automatically sync your save files between multiple iOS 5 devices. This allows players to start the game on one iOS 5 device and then continue on another from the point at which they left off.

Infinity Blade 2 features "massively social" gameplay challenge modes, driven through back-end server support, providing new challenges for players to complete in conjunction with each other.

Finally, Facebook integration is another great feature coming in *Infinity Blade 2*. This will allow players to post accomplishments, invite other players and friends to challenges, and utilize their Facebook network to enhance their gameplay experience.

Infinity Blade 2 is set to release on the App Store December 1, 2011. Be sure to visit http://www.infinityblade2.com for all the exciting details.

UDK BREAKS ONE MILLION, ADDS MAC SUPPORT

And, last but not least, I'm pleased to report the Unreal Development Kit has more than 1 million unique installs and now support Mac OS X. Yes, that's ONE MILLION unique installs. This isn't a download count nor does it count users who installed a new version of UDK over an old version, or reinstalls. This means there are more than one million different computers onto which UDK has been installed.

While UE3 source code licensees have had access to Mac support for several months, UDK developers can now also target Mac OS X, opening up new revenue opportunities.

We're very excited to see what developers do with these new features utilizing our toolset. The fact that any developer can now build games and apps with UDK and deploy them for Mac, iOS and PC with minimal effort is a thrilling prospect, indeed.

Between seeing UE3 in Flash, in iOS, and on Macs, we're looking forward to a very eventful 2012, to say the least.

WWW.UNREAL.COM

Canadian-born Mark Rein is vice president and co-founder of Epic Games based in Cary, NC. Epic's Unreal Engine 3 has won Game Developer magazine's Best Engine Front Line Award five times along with entry into the Hall of Fame. UE3 has won three consecutive Develop Industry Excellence Awards. Epic is the creator of the mega-hit "Unreal" series of games and the blockbuster "Gears of War" franchise. Follow @MarkRein on Twitter.

UPCOMING EPIC ATTENDED EVENTS

Montreal Int'l Game Summit
Montreal, Quebec
November 1-2, 2011

Please email **licensing@epicgames.com** for appointments

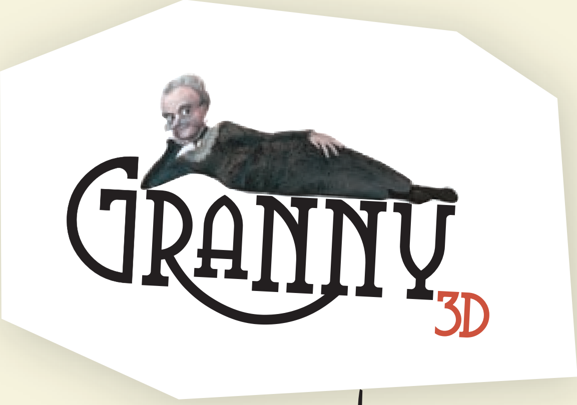
IF YOU WANT TO BE AN EVEN MORE

AWESOME

GAME DEVELOPER

THEN YOU NEED TO BE USING

Granny 3D



You get a run-time dynamic

3D ANIMATION SYSTEM with

AMAZING content exporters

EASY data manipulation

and a brand new blend graph editor.

USING OUR TOOLS NOT ONLY MAKES YOU MORE awesome, THEY MAKE YOU rad!

RAD© GAME TOOLS

www.radgametools.com
(425) 893-4300