DEVELOPER ROUNDTABLE: **TRIPLE-A, FREE-TO-PLAY**

FEBRUARY 2013
VOLUME 20 NUMBER 02

gd

GAME DEVELOPER MAGAZINE

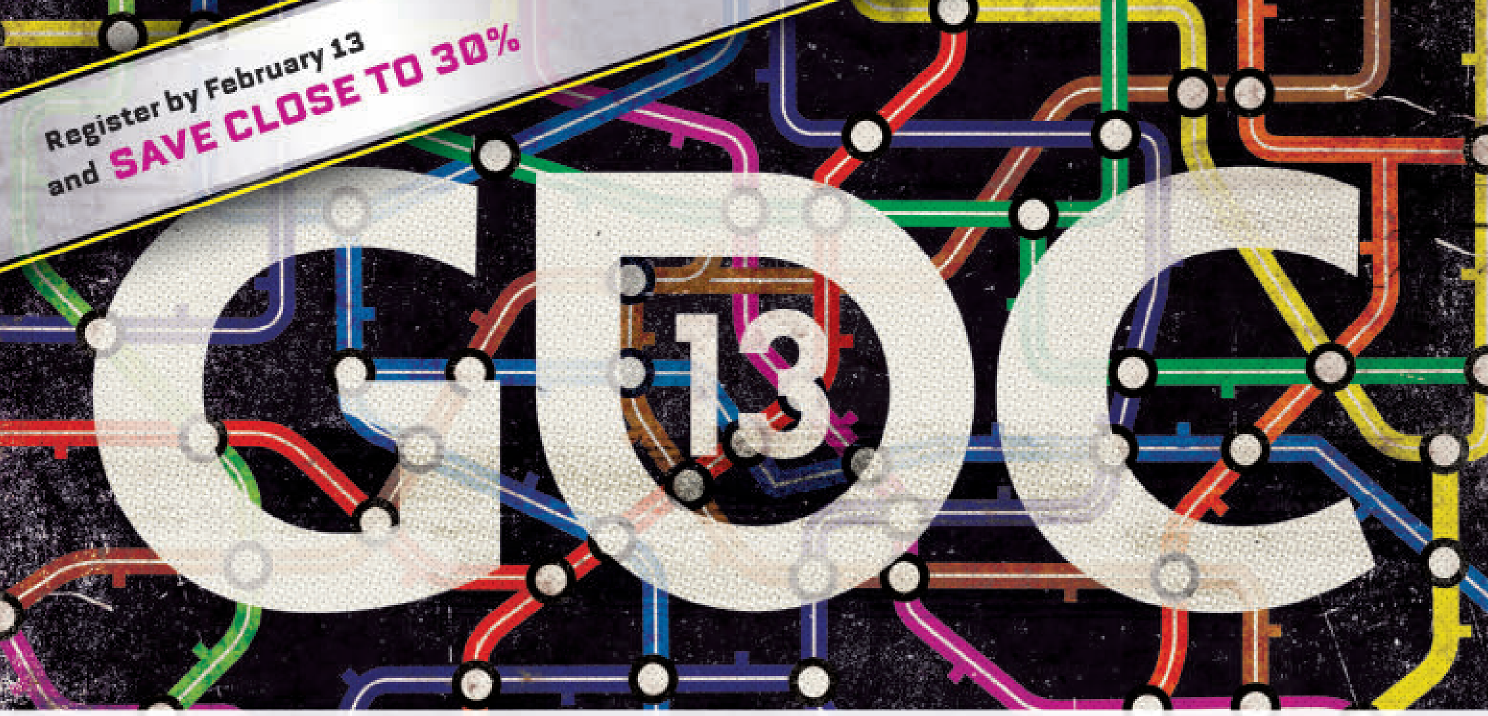
+
BUILD BETTER
TOUCHSCREEN CONTROLS

postmortem
MARK OF THE NINJA

THE LEADING GAME INDUSTRY MAGAZINE

UBM Tech

Register by February 13 and **SAVE CLOSE TO 30%**



GAME DEVELOPERS CONFERENCE® 2013

SAN FRANCISCO, CA | MARCH 25–29, 2013 | EXPO DATES: MARCH 27–29, 2013

JOIN US FOR FIVE DAYS OF SESSIONS, TUTORIALS, BOOTCAMPS, AND ROUNDTABLE DISCUSSIONS ON A COMPREHENSIVE SELECTION OF GAME DEVELOPMENT TOPICS TAUGHT BY LEADING INDUSTRY EXPERTS.

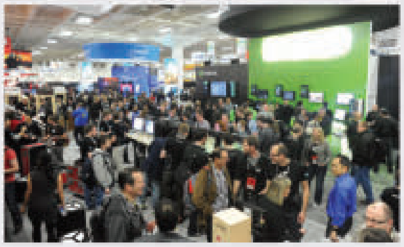
learn

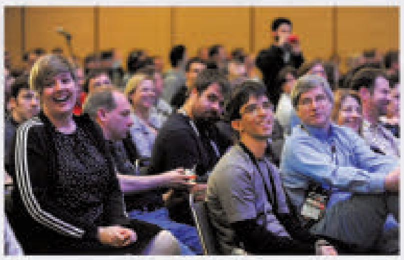
SUMMITS AND TUTORIALS
(MONDAY & TUESDAY)

- AI
- Free to Play Design & Business
- Game Narrative
- GDC Education
- Independent Games
- Localization
- QA
- Smartphone & Tablet Games

MAIN CONFERENCE SESSIONS
(WEDNESDAY–FRIDAY)

- Audio
- Business, Marketing & Management
- Game Design
- Production
- Programming
- Visual Arts
- Monetization (SPONSORED)
- Game Career Seminar (FRIDAY)





network

GDC Play
(TUESDAY–THURSDAY)

Business Center
(WEDNESDAY–FRIDAY)

Career Pavillion
(WEDNESDAY–FRIDAY)

inspire

INDEPENDENT GAMES FESTIVAL
(MONDAY–FRIDAY)

Game Developers Choice Awards
(WEDNESDAY)

Expo Floor
(WEDNESDAY–FRIDAY)

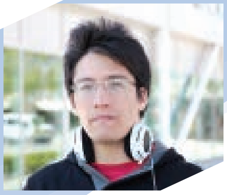
VISIT **GDCONF.COM** FOR MORE INFORMATION.

UBM Tech

toc 001
CONTENTS_February 2013
VOLUME 20 NUMBER 02

POWER WORD: GAME

CHANGING THE TERMS OF THE VIOLENT VIDEO GAME DEBATE



As our industry is dragged into yet another round of scapegoating, I am discovering that the conversation about violent video games is rigged against us, and that we collectively need to change the terms of the conversation before we sit down to talk with anyone.

THE QUESTION IS THE PROBLEM If someone asked you, "Do violent video games cause people to be more violent?", how would you answer? Well, science is a good first bet, but it's difficult to draw a scientifically valid chain of causality behind the act of playing a violent video game, an individual's corresponding physical response (increased heart rate, higher blood pressure, and so on) and psychological response, and then use the results to connect all those factors to something like an overall uptick in mass shootings. We simply don't understand enough about people's behavior (individually or in the aggregate) to answer this question definitively. So, instead, we rely on our intuitions and our past experience to guide us. Have you ever consumed a media work that made you feel something? Probably. Did those feelings incite you to do something bad to other people? Probably not—but perhaps you might imagine it could incite other people (children, mentally ill, and so on) to do so.

"GAMERS" ARE OTHER PEOPLE In general, we don't think of ourselves as "Book-Readers" or "Movie-Watchers" or "Music-Listeners." But playing games is marketed as an identity; if you play games, you are a Gamer. This is likely left over from the days before everyone carried around smartphones, but it persists because people still make plenty of money selling to Gamers. I'm not a businessperson, but I imagine that creating a dedicated audience that defines themselves primarily as "people who buy what you're selling" is pretty amazing (even if that means energy-drink vendors show up to professional conferences and sling tall-boys around).

But when you've defined your consumers as "different from everyone else because they consume your product," it's easier to blame them (and you) for things that go wrong, because you've conveniently defined them as "different." (This is one reason why we generally don't use the word "Gamer" in *Game Developer*, by the way; it is exclusive, not inclusive, and it paints a picture of a person that many people who play games simply cannot relate to in order to sell stuff to people, which does the medium as a whole a disservice.)

GAMES ARE DEFINED BY VIOLENCE What is a violent video game? The tautological answer is "a video game with violence in it." But ANGRY BIRDS is basically about avian suicide bombers, and no one calls it a violent video game, so the answer must be something else. Video games suffer from an unfortunate rhetorical shift because our genres typically describe what we do, and that kind of makes us look bad when our most popular genre has (first) "person shooter" right there like it's an aisle at Blockbuster. Sure, as a consumer, it makes sense to group games that are similar in action, just like how movies group by what viewers feel (movie genres are typically defined by emotions and setting themes; "science fiction comedy," for example). Unfortunately, that means we have a big shelf of games about shooting people in the face.

If we broke down the NPD Group's list of 2012's top-10 best-selling retail games in terms of violent games vs. non-violent games, then it doesn't look great; there are five games out of the top 10 that would be violent games. But if we described that top 10 in terms of movie genres, we'd have two war movies (CALL OF DUTY games), a historical action movie (ASSASSIN'S CREED 3), a sci-fi action movie (HALO 4), a post-apocalyptic sci-fi thriller (BORDERLANDS 2), and a kiddie superhero cartoon (LEGO BATMAN 2) up there with three sports documentaries (NBA 2K, FIFA SOCCER, MADDEN) and a dance movie (Just Dance 4).

We should be talking about controversial games, and discuss their messages and merits—including their questionable, gratuitous, or excessive uses of violence—but that shouldn't hold the medium hostage any more than *Django Unchained* should be able to hold film hostage. We should talk about how the companies that sell games which involve shooting people in countries that the U.S. is currently at war with to secure its access to oil are forging cross-promotions with the ones that sell guns and Hummers, but that shouldn't require industry reps across the spectrum of games to meet with a government task force.

GREAT (POWER, RESPONSIBILITY) So how do we recast the conversation about violent video games? We can refuse to participate in conversations that insist on pigeonholing the medium, but only if we're also advancing other conversations in its place; first, by ditching the word "Gamer" and all of its respective marketing connotations; second, by defining our games in terms of content and theme; third, by making games that use violence with purpose and calling out games and creators that don't. Game developers have the power to simulate experiences that no other medium can, and that power should not be dismissed with "Oh, they're just video games."

—Patrick Miller
@pattheflip

gd 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
gamedeveloper@halldata.com
www.gdmag.com/contactus

EDITORIAL

PUBLISHER
Simon Carless *scarless@gdmag.com*
Pearl Verzosa *pearl.verzosa@ubm.com*
EDITOR
Patrick Miller *pmiller@gdmag.com*
EDITOR EMERITUS
Brandon Sheffield *bsheffield@gdmag.com*
MANAGER, PRODUCTION
Dan Mallory *dmallory@gdmag.com*
ART DIRECTOR
Joseph Mitch - *jmitch@gdmag.com*
CONTRIBUTING WRITERS
Alexandra Hall, David Daw, Tim Rogers, Brad Bulkley, Will Luton, Jamie Cheng, Nels Anderson, Tatu Aalto, Jason McGuirk, Steve Theodore, Soren Johnson, Damian Kastbauer, Sana N. Choudary, Brandon Sheffield, Matthew Wasteland, Magnus Underland
ADVISORY BOARD
Mick West Independent
Brad Bulkley Microsoft
Clinton Keith Independent
Bijan Forutanpour Sony Online Entertainment
Mark DeLoura Independent
Carey Chico Independent
Mike Acton Insomniac
Brenda Romero Loot Drop

ADVERTISING SALES

VICE PRESIDENT, SALES
Aaron Murawski *aaron.murawski@ubm.com*
t: 415.947.6227
MEDIA ACCOUNT MANAGER
Jennifer Sulik *jennifer.sulik@ubm.com*
t: 415.947.6227
GLOBAL ACCOUNT MANAGER, RECRUITMENT
Gina Gross *gina.gross@ubm.com*
t: 415.947.6241
GLOBAL ACCOUNT MANAGER, EDUCATION
Rafael Vallin *rafael.vallin@ubm.com*
t: 415.947.6223

ADVERTISING PRODUCTION

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

REPRINTS

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

AUDIENCE DEVELOPMENT

AUDIENCE DEVELOPMENT MANAGER
Nancy Grant **e:** *nancy.grant@ubm.com*
LIST RENTAL
Peter Candito
Specialist Marketing Services
t: 631-787-3008 x 3020
petercan@SMS-Inc.com
ubm.sms-inc.com

UBM Tech
WWW.UBM.COM

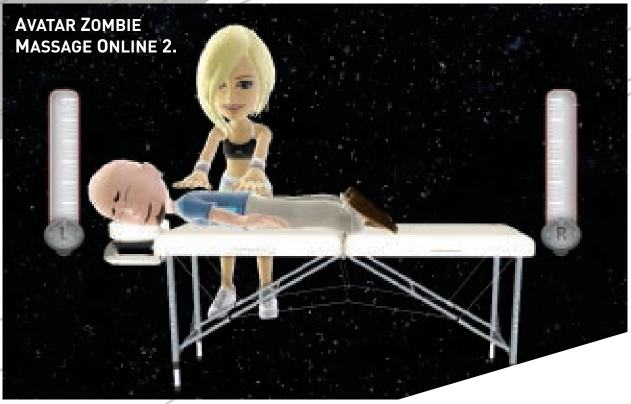
fmod® studio

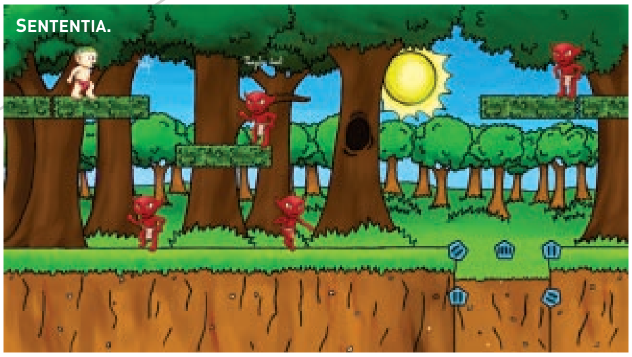
Dowload it today at www.fmod.com

STAGING AN UPRISING

RECAPPING XBOX LIVE INDIE GAMES UPRISING 2012

SINCE 2010, GROUPS OF DEVELOPERS HAVE BANDED TOGETHER TO PROMOTE ORIGINAL HIGH-QUALITY GAMES ON THE XBOX LIVE INDIE GAMES CHANNEL IN A GAME-RELEASE EVENT CALLED THE INDIE GAMES UPRISING. WITH THE CONTINUED HOPE OF RAISING AWARENESS FOR THE CHANNEL, 2011 SUMMER UPRISING COORDINATOR DAVE VOYLES AND I PLANNED A THIRD AND POSSIBLY FINAL UPRISING TO LAUNCH IN SEPTEMBER 2012. HERE'S WHAT WE LEARNED FROM IT.


AVATAR ZOMBIE MASSAGE ONLINE 2.


SENTENTIA.

SOWING THE SEEDS In early 2012 I began developing a small-scale video game called SENTENTIA. Toward the end of development, I noticed a lot of exciting games would be coming out around the same time as my own, so I started to gauge interest among the community for doing another Uprising promotion. To my surprise, it seemed like a lot of developers weren't interested due to poor sales of the Summer Uprising games, but I posted a thread on App Hub urging the community to put concerns of money aside. Dave Voyles soon dropped by the thread, and before long things were starting to happen!

The previous Uprising used a public submission process and had the community vote on the final titles, but Dave explained that he felt this was one of the biggest mistakes of the last promotion due to the entire process taking so long, so instead we decided to stick with just the developers who had finished games ready for an August/September release. In total, we ended up with GATEWAYS by Smudged Cat Games, CITY TUESDAY by Chris Zukowski, QRTH-PHYL by hermitgames, ENTROPY by Autotivity Entertainment, DIEHARD DUNGEON by tricktale, PIXEL by Ratchet Game Studios, XENOMINER by Gristmill Studios, SMOOTH OPERATORS by Andreas Heydeck, and my game, SENTENTIA. One game would be released each weekday from September 10-20.

WILL NOT BE TELEVISED Thanks to Ryan Donnelly over at VVGtv, we quickly had a completed trailer that showcased the games, and we sent out a press release to coincide with it. The response was insane; within the first 48 hours we were doing interviews with *Wired*, Ars Technica, and *Official Xbox Magazine*, and getting coverage on pretty much every major gaming site. It was hard for Dave and me to keep up with the emails we were getting, and the developers were keeping busy with interviews as well. In retrospect, I feel that we fired the gun too early in terms of marketing. We released the trailer a few weeks before the promotion started, and all of our major press coverage came out at this time. Could we have driven more traffic to the games if there were actually some available to purchase?

REVIEW CODE SETBACKS We wanted to distribute review codes to the press, but we needed to get those codes from App Hub, and for almost the entire promotion the code page was down. Microsoft eventually fixed this issue, but many of the developers involved were infuriated at the response time. (I personally found it a little insulting that some review sites refused to fork over a dollar apiece to buy the games.)

SUBPAR SALES Within the first month, our best-selling title, XENOMINER, sold around 4,000 copies. QRTH-PHYL and DIEHARD DUNGEON each sold around 2,000, and all of the other games sold fewer than 1,000 copies. In total, our sales were the lowest of the Uprising events, even though we had received more press coverage than any of the previous promotions—perhaps because other Uprising events were promoted on the Xbox Live Dashboard at time of launch.

WHAT PLAYERS WANT When you talk to players about Xbox Live Indie Games, they often say things like, "It's filled with a bunch of clones and garbage apps"—mostly because the most successful games on XBLIG seem to fall into this category. FORTRESSCRAFT, CASTLE MINER Z, and TOTAL MINER: FORGE have all sold over one million copies, and many developers figure that they should make similar games to succeed on the channel. In a thread on App Hub, JForce Games, the developer of AVATAR ZOMBIE MASSAGE ONLINE 2 famously defended his game by saying, "You can't get mad at a business for chasing easy money, you should really just be mad at the consumers. [It's] supply and demand, yo. Looking at the [sales data] is such a strong indication of the market's demand. We're just giving people what they want. We're making people happy."

If we want to see any type of change in the market, the developers in the community need to start making games that are more authentic and sincere. I'm not suggesting everyone should go out and make a game about their life, but rather something that shows the world who you are and brings something unique to the table. After all, that's what indie games are all about!

IT'S NOT ME, IT'S MICROSOFT The other issue I see is the inability of developers to accept that they can do better. Many developers have started to pack their bags and leave the channel because they feel they aren't making enough money, and they typically blame Microsoft, claiming that they are killing the channel with their lack of support and marketing. Certainly there are a number of things Microsoft could do to help improve visibility, but these types of comments typically come from developers who have published around two to four games and have actually done pretty well (as in, $20k-40k on their second or third title)—not a huge amount of money, but considering public sales data indicates the majority of XBLIG games make less than $800, it's certainly a respectable amount.

XBLIG has opened the doors for quite a few developers, and has allowed some of us to make decent money early on in our careers. In just a couple of years, Zeboyd Games made four games on the channel and then moved on to creating the newest Penny Arcade game, which is pretty crazy.

THE MORAL OF THE STORY If you're going into indie games solely focused on money, there's a good chance that you will not succeed. I think the success of games like MINECRAFT have created a false perception that independent development is a gold mine waiting to be drilled, but even Notch made quite a few games before MINECRAFT. This seems like a no-brainer, but success is truly a long and hard struggle.

LOVE OF THE GAME You could start developing clones or gag games and find yourself making some good money on XBLIG, but how long will that last? In 10 years how many people will look back and remember those games as truly great experiences? I urge developers to look farther ahead and realize what effects our games have on the overall channel's reputation. We don't need another Uprising of a small group of developers; we need an Uprising involving the community at large. Only then can we start to turn Xbox Live Indie Games into a place where original and creative games can truly prosper and grow.

—Michael Hicks

BEST SPEEDRUN *EVER*

TURNING POKÉMON YELLOW INTO A MIDI PLAYER

Tool-assisted speedruns can be a great way to kill an hour or two—who doesn't like watching people push classic games past their breaking point? However, Robert "Bortreb" McIntyre recently put together a POKÉMON YELLOW "speedrun" that allowed him to write a program in the game itself—specifically, one that played the *My Little Pony: Friendship is Magic* MIDI theme song. You can find a demonstration video and the in-depth explanation at http://aurellem.org/vba-clojure/html/total-control.html ; here's a brief excerpt from the write-up.

"The Game Boy is an 8-bit computer. That means that ultimately, anything that happens in POKÉMON is a result of the Game Boy's CPU reading a stream of 8-bit numbers and doing whatever those numbers mean. For example, in the Game Boy, the numbers [62 16 37 224 47 240 37 230 15 55] mean to check which buttons are currently pressed and copy that result into the "A" register. With enough numbers, you can spell out an interactive program that reads input from the buttons and allows you to write any program you want to the Game Boy. Once you have assembled such a program and forced the game to run it, you have won, since you can use that program to write any other program (like TETRIS or PAC-MAN) over POKÉMON YELLOW'S code. I call a program that allows you to write any other program a "bootstrapping program." So, the goal is to somehow get a bootstrapping program into POKÉMON YELLOW, and then force YELLOW to run that program instead of its own...[and] if we can get the right items in the right quantities, we can spell out a bootstrapping program."

—Patrick Miller

Perspective

http://www.seewithperspective.com/
https://www.digipen.edu/?id=1170&proj=25930

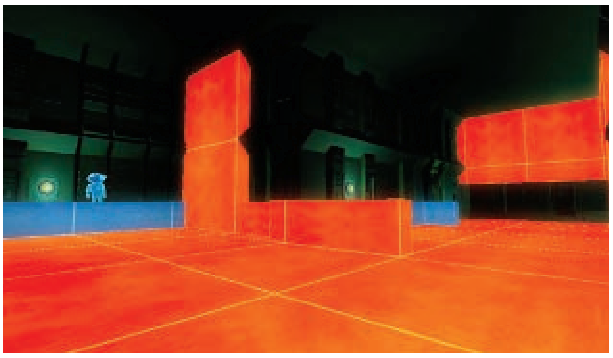
IN EARLY DECEMBER, A TEAM OF STUDENTS AT DIGIPEN RELEASED PERSPECTIVE, A MIND-WARPING TREAT OF A PUZZLER THAT CHALLENGES YOU TO SAFELY NAVIGATE A 2D PLATFORMING HERO THROUGH OFT-HAZARDOUS TERRAIN DERIVED FROM THE FIRST-PERSON VIEW OF A MOVEABLE CAMERA. (IT'S HARD TO SUM UP IN A SENTENCE.) YOU CAN DOWNLOAD IT NOW, OR TRY IT OUT AT THE INDEPENDENT GAMES FESTIVAL NEXT MONTH.

ALEXANDRA HALL: ***How'd you come up with that concept?***
POHUNG CHEN: Initially we wanted to make cameras and projectors that displayed 3D holograms. The idea was similar to PORTAL, and we weren't sure how we could create interesting puzzles with that particular mechanic. Eventually we described the basic idea using scribbles on the whiteboard. Our first levels were created using Google SketchUp. It gave us a visual tool to describe an initial proof of concept.

There are games with a similar premise, but PERSPECTIVE is more of a first-person puzzle game than platformers like Crush or FEZ. We ended up borrowing more heavily from first-person level design and game structure than we did from 2D platformers.

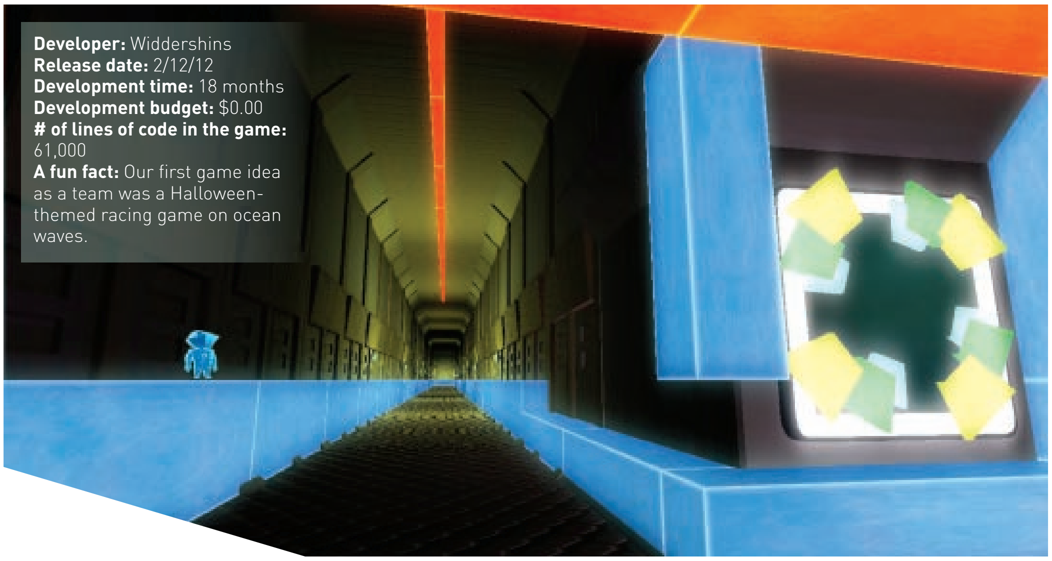
AH: ***Did you conceive of all the perspective-warping gimmicks at the outset?***
PC: We had most of the core pieces early on. It took a long time before we figured out how we can best teach each piece to our players. We spent a lot of time making the game more accessible to people who aren't spatial-contortion masochists. Players who seek really challenging puzzles didn't find PERSPECTIVE to be mind-bogglingly complex.

There were a few mechanics that we came up with or stumbled into later in the development process. One of the more controversial mechanics was moving objects; we didn't want to turn PERSPECTIVE into a reflex/dexterity-driven platformer. We wanted to expand upon the purely puzzle elements of the game by using moving objects, but it was always tempting to create levels that require dexterity and reflex.



AH: ***Did Perspective present similarly tricky programming challenges?***
PC: There were challenges across the board. Our first approach to generate the 2D world was geometric—it involved vertex projections and a lot of tricky triangle splitting. It was fairly slow and buggy, and had a lot of difficult-to-solve edge cases. There were ideas we wanted to try that would be impossible without better performance.

After a few months, we switched over to a raster, pixel-based approach. We now render the scene into a fixed-size render target with object IDs and read the buffer from the graphics card. This causes the graphics card to lock and



Developer: Widdershins
Release date: 2/12/12
Development time: 18 months
Development budget: $0.00
# of lines of code in the game: 61,000
A fun fact: Our first game idea as a team was a Halloween-themed racing game on ocean waves.

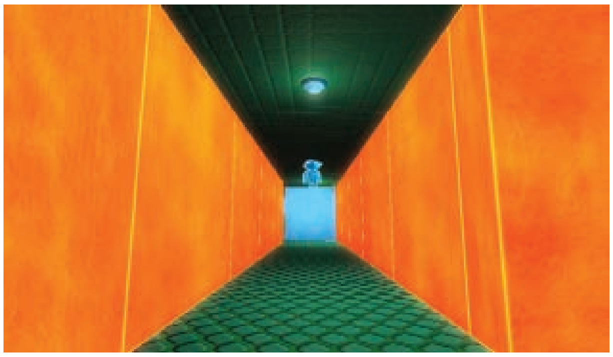
required us to render the scene twice, but it was a better solution than our initial geometric approach. Then came the challenge of writing a pixel-based character controller for arbitrary 2D terrain.

SEAN REILLY: One of the interesting problems with writing a player controller for a game like this is that you have no idea what type of terrain the player is going to encounter. You can't design your levels to avoid certain types of slopes or extremely narrow platforms because the user gets to manipulate the world, essentially creating their own levels. It was important during development to take into account all possible edge cases and to create a 2D player which could navigate any possible environment.

AH: ***What was your level-design process like?***

LOGAN FIETH: At first I took my usual approach of designing levels, which is to draw out ideas. For side-scrollers, it's best to draw a level layout from the side—the way you would expect. For first-person games, I usually draw from a top-down view. For PERSPECTIVE, I quickly learned that neither option was ideal, so I really had to draw from a multitude of angles. Even so, it wasn't easy to communicate ideas (even to myself) this way, so I got into the editor to build levels earlier than I usually do. I would still draw out levels beforehand, but it quickly became scribbles that were just shorthand for ideas I had.

I created levels and puzzles at a rapid pace throughout development. The key to this is most levels, ideas, or puzzles were going to be cut, so I couldn't waste time with polish until we got closer to launch. This results in levels being a learning process—



you build it quickly, playtest it, probably cut it, and bring the good parts into the next level you build.

AH: ***Any unexpected difficulties?***

PC: Lots. We didn't anticipate how difficult it would be to structure our game in such a way that made it accessible. Our first external playtest was a disaster. We went into a board-game shop in Redmond and no one got past the first area. Some people didn't even realize there was a 2D character on the wall and were just stumbling around in first-person mode.

Because of this, we decided to bring on Logan as a level designer. We needed someone to work full-time in figuring out the arc of the game and how we can better teach new players the mechanics of PERSPECTIVE. ep

ASOBO

STUDIO

Crafting the future of FUN

Programmers, Game Designers, Producers, 3D Artists or Animators,

Apply Now.

jobs@asobostudio.com

See website for details: www.asobostudio.com

Class Revolution

SETH SIVAK LEADS EX-ZYNGA BOSTON DEVS AT NEW STUDIO PROLETARIAT INC.

PATRICK MILLER: ***So what are you doing now, and how'd you get there?***
SETH SIVAK: Right now, I am the CEO of a small game studio named Proletariat Inc. in Boston. The team is made up of senior members and leads from Zynga Boston. We are trying to build high-quality multiplayer games on mobile and tablet, and we are close to closing a distribution deal for our first title.

I started as a gameplay engineer at a music social game company called Conduit Labs, where I did mostly game prototyping. In August 2010, Conduit Labs was acquired by Zynga, and I moved into product management because I was somewhat interested in analytics and it felt like a good challenge. When we started to work on INDIANA JONES ADVENTURE WORLD (IJAW), I went back to my gameplay engineering roots and ended up designing and prototyping most of the moment-to-moment gameplay and puzzles in the game. As we started to hire content designers to build levels and quests I shifted into the lead designer role. When the team transitioned away from IJAW I became the executive producer (what Zynga calls a project lead) for a new mobile title, and we were only weeks away from shipping that game when Zynga closed the Boston studio and laid off the entire staff.

PM: ***Why'd you choose the name "Proletariat"?***
SS: We are all part of the working class in the industry; we are all contributors to the construction of games. We plan to work really hard and have everyone on the team see the benefits from that hard work.

PM: Were there any particular lessons about the game industry you learned while working at Zynga?
SS: Zynga taught me a great many things about free-to-play games and about how to run a game like a service. The most important lesson I learned from Zynga is the value of analytics, and how they can be used to help designers better understand their game and their players. The cautionary tale from Zynga is that developers were encouraged to focus only on optimizing for the short term; that attitude is what continues to hurt Zynga's games. At Zynga, there was constant pressure to sacrifice long-term potential for short-term gains, and it is starting to catch up to them.

PM: ***So how are you planning on developing with long-term potential in mind?***
SS: In order to build long-running games, we hope to focus on quality, and developing economies and features that can last for the lifetime of a game. Most Zynga games are constantly inventing new economies and then inflating them quickly through sales, which eventually makes the games extremely complicated and gives diminishing returns. Our goal is to create free-to-play games that are fun to play for free, and have the "pinch" to pay only come after the player is invested and interested. We also want to focus on making sure that every time a player spends money in our game they feel good about it, which is a very hard thing to do. We hope to avoid the trap of overoptimizing in a single area by picking reasonable goals that we think will give us good returns without sacrificing other metrics too severely. It is all about balance.

PM: ***Did starting your own studio feel like a risk, or like the natural next step in your career?***
SS: We looked around at the types of games we were building for Zynga, and we knew that we could be making these games on our own. Doing this sort of thing will always be a risk, but we felt that the experience of the team and the knowledge that we have right now coming out of Zynga will give us an advantage. For me, I always knew I wanted to try running an independent studio at some point, and this felt like as good of a time as any.

PM: ***What's it like transitioning from game design and product management work to being the CEO?***
SS: There is plenty of overlap between being a designer or product manager and a CEO. The big leap is in responsibility and trust—as a designer you are generally responsible for an experience being fun and interesting, as a product manager you are responsible for a business outcome, as the CEO you are responsible for both and keeping the team happy. I prefer to lead by example, so I like to stay involved on the design and product side as much as I can, while balancing the needs of the company and the team. As the CEO you need to be willing to trust your team more and give up control knowing your team will make the right choices, and that can be difficult. gj

Who Went Where

↗ **KATE EDWARDS**, the founder of the International Game Developers Association's Localization SIG, has succeeded Gordon Bellamy as executive director of the IGDA. An experienced Microsoft veteran, Edwards most recently worked as a localization consultant on games including FALLOUT 3, MODERN WARFARE 3, and DANCE CENTRAL.

↗ Glasgow-based mobile publisher One Thumb Mobile has hired a number of veteran developers as part of its expansion. **PAUL SIMON** (RUNESCAPE, LORD OF THE RINGS ONLINE) has been appointed lead game designer, while **PVIKTOR SÁGHY** (Most Wanted Entertainment, Eidos Hungary) and **DAVE ALLSOP** (Blizzard, Wizards of the Coast) came onboard as art director and concept artist, respectively.

↗ Former Muve and Napster exec **JAMES MITCHELL** has been appointed chief_technology officer at Music Mastermind. The company's Zya software, currently in beta, aims to make it easier for players to create catchy music by manipulating famous hooks from hit songs.

New Studios

↘ **STAR FILLED STUDIOS**, founded in September by Tod Semple (formerly of LucasArts/PopCap) and Jeff Gates (Maxis/PopCap), has been acquired by Valve. There's not yet word on what this new Valve subsidiary, located in San Francisco, will be working on.

↘ Up-and-coming social game developer **KIXEYE** is continuing its expansion by opening a new studio in British Columbia, Canada. Former Zynga director of development Clayton Stark has been appointed general manager, and the studio is currently staffing up in prep for "new special projects."

↘ **NAUGHTY DOG, UBISOFT, AND SONY** vets have launched a new mobile games studio, called Mojaro. The studio's first release is iOS game KNIGHTSCAPE, which utilizes an in-house technology platform.

A Complete Monetization Platform for Online Games

100+ Payment Choices

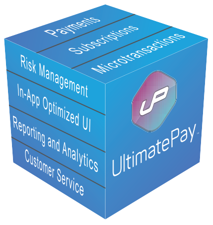
Localized Purchase Experience

Flexible Subscriptions

Hybrid Free to Play + Subscription Models

Reporting & Analytics

In-Game Optimized Purchase Experience



playspan®

Online Games Commerce

Learn more

- GDC 2013 - Stop by the Visa booth #824 in South Hall to learn more about our PlaySpan and V.me by Visa products
- Go to *www.playspan.com/gdc2013* for more product information and white paper downloads

Frank N. Magid Associates, Inc.

Come by Visa booth #824 on Wednesday at 3:30 to hear Magid Associates present their report on online game business models.

Or request a copy at *www.playspan.com/gdc2013* We will send it out after the event.

PlaySpan is now part of VISA

GDC PREVIEW 2013

The Game Developers Conference in San Francisco (March 25–29, 2013) reflects not only the current state of the industry, but also shows hints of where it's going. The main conference features, analysis, lessons, and thought leadership from the best of the industry—across consoles and PC, mobile and social, indie and triple-A—while the summits dive deep into specific areas of game development that might not otherwise get enough attention.

From audio to visuals, design to business, GDC's programming covers the full gamut of today's game industry. It's also host to prestigious events like the 15th annual Independent Games Festival, which has enjoyed another year of record-breaking entrant numbers, and the 13th annual Game Developers Choice Awards.

As a teaser before the show, we've highlighted a number of interesting talks from both the mainline GDC and the summits (see page 14), and more talks are added frequently at www.gdconf.com. See you at GDC!

10 QUESTIONS: AM I READY TO GO INDIE?

Don Daglow (Daglow Entertainment LLC)

➔ What really goes into being an indie developer? Is there a checklist that tells you what to do? When is an indie job just the same job in a new package? What do you give up and what do you gain? What are the perils that can turn that dream of building your own game into a long-term nightmare? Don Daglow has spent the last five years of his 40-year career working with small teams. In this session, he'll walk you through a checklist of questions that cover if, when, and how you want to set sail on your own ship.

A SCIENTIFIC ASSESSMENT OF THE VALIDITY AND VALUE OF METACRITIC

Adams Greenwood-Ericksen (Full Sail University)

➔ Over the last decade Metacritic has become an important but controversial fixture in the industry. In this session, a statistical assessment of the validity and financial impact of the influential metareview site will be presented. Metacritic's formula will be explained, and a method for modeling and predicting the weights Metacritic staff assign to various publications will be presented and discussed.

BRAVE NEW WORLD: NEW BUNGIE IP

Joe Staten (Bungie) and Christopher Barrett (Bungie)

➔Two decades of success in the gaming industry is no small feat, but after ten years of Halo, Bungie found themselves faced with a tremendous challenge: to build a whole new world, filled with even more amazing mysteries, places, creatures, and opportunities for player investment. For the first time ever, Bungie creative directors will discuss their world building techniques, from concept to production. Attendees will walk away with key insights into Bungie's battle-tested design process. They'll get a glimpse of the brave new world that has been built, a place where the next ten years of great Bungie adventures will unfold.

CASE STUDY OF A MICROSTUDIO STARTUP: THE NEXT THREE YEARS

Randy Smith (Tiger Style)

➔ At GDC 2010, game designer and non-businessperson Randy Smith related his story of starting indie studio Tiger Style, investing a breathtakingly minimum amount of money, time, and effort, but culminating in the hit iPhone game SPIDER: THE SECRET OF BRYCE MANOR. Three years later, Randy is returning to share what has

PHOTOS COURTESY OF GAME DEVELOPERS CONFERENCE

happened to Tiger Style since then. The business has grown more complex, and the team strives to keep it healthy while staying focused on development. Topics will include evolutions to staffing and team management, royalties and compensation, promotion, business type, finance, and all-revealing sales data, including data on its latest release, WAKING MARS.

CLASSIC POSTMORTEM: MYST

Robyn Miller

⊖ The best-selling PC game of the 1990s, MYST is also often attributed as the game that sold CD-ROM drives. Its majestic 3D world was too large for floppy disks, filled with puzzles and mysteries that unraveled in front of players' eyes in an engrossing first-person adventure. MYST'S immersive atmosphere and play even gave rise to the debate of games two decades ago. Since its release in 1993, it has been remade and ported to over 10 platforms, including most recently Nintendo 3DS and iOS. Robyn Miller, the original co-creator and sound composer, will discuss how he and his brother Rand created a game that remains relevant and commercially desirable over 20 years later.

CLASSIC POSTMORTEM: X-COM: UFO DEFENSE

Julian Gollop

⊖ Firaxis's and 2K Games's recent X-COM: ENEMY UNKNOWN is actually a remake of a series that began 20 years ago. It all started in 1994 with MicroProse's UFO: ENEMY UNKNOWN—titled X-COM: UFO DEFENSE in North America—a real-time base-management simulation with turn-based tactical combat and an engaging story of alien invasion. The marriage of its distinct Geoscape and Battlescape views represented the game's strategy and battle modes, respectively; and they provided what felt like two different and compelling games in one. In this postmortem, Julian Gollop will lay out the tactics he deployed in directing, codesigning, coprogramming, and even codrawing the first, and often highest-regarded, UFO/X-COM entry.

DESIGNING HUMOR IN BORDERLANDS 2

Anthony Burch (Gearbox Software)

⊖ The lead writer of BORDERLANDS 2 will explain how the team attempted to convey humor, not just through dialogue and art, but through game mechanics. He will discuss how a quest with no gameplay can actually be funnier than a quest with it, how every mechanic holds the potential for humor, and how even jokes need debugging. There will also be more use of the word "fart" than you are probably comfortable with.

DESIGNING WITHOUT A PITCH: FTL POSTMORTEM

Justin Ma (Subset Games), Matthew Davis (Subset Games)

⊖ The creation of FTL: FASTER THAN LIGHT began with the desire to experience what it would feel like to be the captain of a starship. Many games have focused on space battles, but few games have focused on what happens in the ship itself. The player experience was the primary goal; gameplay structures, mechanics, and genre were all secondary. By focusing on a high-level goal of experiencing a singular feeling, and developing it with minimal preconceptions about FTL'S gameplay, Subset Games was allowed to frequently alter or abandon aspects of its design, which eventually led to the game we know today. Matthew Davis and Justin Ma will share their creative process as they take you step by step, from concept to crowdfunding to release.

GAME WRITING FUNDAMENTALS IN A DAY

Evan Skolnick (LucasArts)

⊖ A perennial GDC favorite, this dynamic, engaging presentation on the fundamentals of fiction writing is designed for anyone interested in improving the narrative quality of their games. Hosted by Marvel

GDC 13

Comics writer/editor Evan Skolnick, the comprehensive, day-long tutorial offers writers and nonwriters an opportunity to learn (or relearn) the basics of good story structure, vibrant character development, snappy dialogue writing, and more. Prior attendees of this tutorial—a broad mix of writers, designers, artists, animators, engineers, and producers—have called it "amazing," "wonderfully paced," and "essential knowledge to further the medium."

HALO REBORN: A POSTMORTEM ON THE CREATION OF HALO 4

Josh Holmes (343 Industries)

343 Industries was formed with a monumental task: to build a new studio from scratch and take over a much-beloved universe from legendary developer Bungie. Its first internally developed title, HALO 4, was released on November 6, 2012, to widespread critical acclaim. In this session, franchise creative director Josh Holmes will discuss how the studio managed to overcome the overwhelming odds and many challenges involved with assembling a brand-new team to deliver a game worthy of the *Halo* franchise legacy.

INTERACTIVE FICTION: TRADITIONS VS. POTENTIAL

Mordechai Buckman (Independent)

The history of the adventure game is filled with many ambitious stories, but these ambitions are often hindered by the hodgepodge of gameplay styles that the medium is known for. The original text adventures showcased the flexibility of the text-parser interface. Many current adventure games do not use the text parser, and are still providing the same arbitrary activities while trying to tell more complex plots. There is a place for pixel hunts and inventory puzzles in stories, but in most narrative situations they are not appropriate. This is where the dynamic interface becomes useful, a tool for turning story scenarios into intuitive gameplay. This method is demonstrated in the context of several potential narrative scenarios, and contrasted with the conventional approach to adventure game design.

MAD AS HELL: HOTHEAD DEVELOPERS RANT BACK

Eric Zimmerman (Independent), Jason Della Rocca (Execution Labs), Kellee Santiago (Independent), Margaret Robertson (Hide&Seek), Anna Marsh (Lady Shotgun), Naomi Clark (Brooklyn Game Ensemble), Anna Anthropy (Auntie Pixelante)

Each year the rant session brings together a panel of game developers to b*#%h about whatever the hell they want. This year we'll blow the doors off the hinges with a panel of the angriest game developer hotheads we could find. Cutting through the clutter of polite industry chitchat, the rant session takes on the issues that matter to developers in a no-holds-barred format. Fasten your seat belts, and prepare for strong opinions from some of the game industry's most distinguished and dissatisfied game developers. Cohosted by Jason Della Roca and Eric Zimmerman, the rant session is about identifying solutions as well as problems. The audience will have a chance to respond to the rants and join in the discussion.

NEO-RETRO: SMOOTHING IT OUT WHILE KEEPING THE PIXELS

Paul Veer (Skullrobot)

Retro games are no longer just a thing of the past, as new games inspired by retro aesthetics and gameplay are coming out almost every year. Since we are no longer bound by restrictions, we can use this to our advantage and take these games a step further. Paul Veer, indie pixel artist and animator, shows how to take advantage of today's more advanced technology to modernize the graphics of your own neo-retro games in order to create an even stronger nostalgic reaction with your player.

RENDERING ASSASSIN'S CREED III

Jean-Francois St-Amour (Ubisoft Montreal)

This presentation will describe the rendering techniques used for the latest opus in the ASSASSIN'S CREED series. ASSASSIN'S CREED III takes place in an extremely vast and varied world, in both summer and winter, and this required work across a wide number of topics. This talk will cover the game's weather system,

lighting solution, ocean rendering, and material system. In addition, this presentation covers what the AC3 team believes are the biggest bang-for-the-buck improvements that can be made to current-generation titles being ported to DX11 PC hardware.

RETHINKING HOW WE BUILD GAMES AND WHY: THE PAPO & YO STORY

Vander Caballero (Minority)

→ If books and movies can be entertaining experiences that help us grow emotionally, better cope with the difficulties of life, and give us closure in the last pages or minutes—why do only 30% of gamers finish console games? The game industry is failing players by not giving them closure that provides growth or a meaningful takeaway. Based on Caballero's own allegorized, autobiographical story of growing up with his alcoholic father, PAPO & YO proposes a new way of bringing emotional journeys to players, and defies the norm of games that are only being used for escapism.

STILL KICKING: THE VIABILITY OF PAID APPS IN THE ERA OF F2P

Nathan Vella (Capy)

→ Much has been made of the rise of the free-to-play model. Pundits claim that paid apps are dead, and offer stats showing large portions of App Store revenue generated by IAPs. But is the end truly nigh? This session will detail the continued viability of developing paid apps for the App Store by leveraging experience from successes like SWORCERY, WORLD OF GOO, and more. Attendees will be confronted with key issues faced when tackling the F2P shift as small developers, discuss the rationale for choosing to develop a paid app, and learn best practices for success from launch and promotion through long tail.

THE MUSIC OF DEAR ESTHER: CREATING POWERFUL SCORES WITH LIMITED RESOURCES

Jessica Curry (thechineseroom)

→ The soundtrack for the 2012 cult indie hit DEAR ESTHER has been praised for its emotional power and innovative fusion of digital effects and live instrumentation. This talk will explain the process of evolving the soundtrack from the original 2007 mod, which was built using only digital samples, to the remake's lush orchestration. The talk will focus on how to create rich and deep soundtracks on a tight budget, and our process of design, placement, and optimization of the music as a device to focus the player's emotional journey. The success of DEAR ESTHER is frequently ascribed to the deep, holistic fusion of art, story, and music. The core of this talk is how this was achieved, and how studios can create powerful, integrated scores, even on a low budget.

THE PROTOTYPE THAT WAS BANNED FROM HALFBRICK

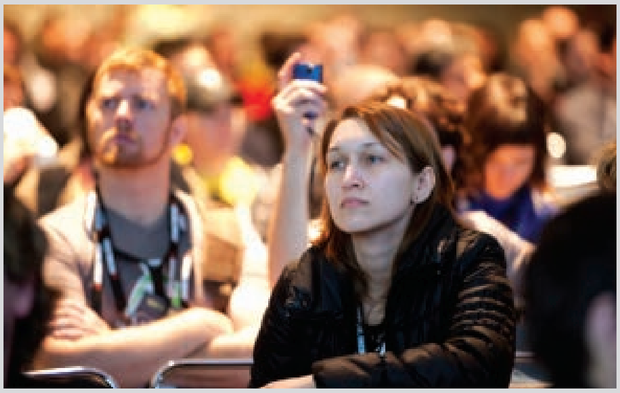
Luke Muscat (Halfbrick Studios)

→ TANK TACTICS was a simple strategy game that was played by 17 employees of Halfbrick Studios. Eight days later it was banned from the workplace. People got upset, rivalries formed, and two employees in particular still don't speak to each other to this day. So the question is, whose fault is all of this? Was it the players' fault for taking the game too seriously, or the designers' fault for creating a system that fosters such destructive social behavior? This talk will explore the story of TANK TACTICS, and examine how the mechanics of this game, and many others, influence player behaviors.

WHY VIRTUAL REALITY IS HARD (AND WHERE IT MIGHT BE GOING)

Michael Abrash (Valve Software)

→ Augmented and virtual reality have been staples of science fiction for decades, and have been touted as "just around the



corner" in the real world for much of that time. They're back again, and this time they have a good shot at breaking through into the consumer space, thanks to the convergence of a number of key technologies. Nonetheless, a great deal of R&D is still needed, because the human perceptual system is remarkably sensitive to certain types of discrepancies from the real world. This talk will look at one such discrepancy in some detail, review a laundry list of such issues, and discuss one possible roadmap to augmented VR's full potential.

WHY WON'T FARMVILLE GO AWAY?

Mike Perry (Zynga), Abhinav Agrawal (Zynga)

While FARMVILLE helped put social games on the map, public perceptions of the game have varied widely. Loved by players but often derided by vocal members of the gaming community, one thing is certain: FARMVILLE has lasted longer than almost all other social games to date. There is a good reason why FARMVILLE has withstood the test of time. Join Mike Perry, executive producer, and Abhinav Agrawal, director of product, from Zynga's FARMVILLE studio to hear how a combination of platform expertise, game design, and development processes have helped sustain FARMVILLE over time. Learn how these lessons can be applied not only to social games, but to many games at large.

GDC SUMMITS

GDC hosts a variety of summits, which delve into specialized and emerging markets in the game industry. With the introduction of the brand-new QA Summit and the Game Narrative Summit, which was previously a fixture at GDC Online, GDC will offer eight summits in all. Below, we've outlined each summit and given you a taste of the key talks featured within them. New talks will be added regularly in the weeks leading up to the show, so be sure to keep an eye on www.gdconf.com for more.

Independent Games Summit

The Independent Games Summit represents the voice of the independent game developer at GDC. It features lectures, postmortems, and rants from some of the most notable independent game creators, including many former and current Independent Games Festival finalists and winners. The Independent Games Summit seeks to highlight the brightest and the best of indie development, with discussions ranging from game design philosophy, distribution, business, marketing, and much more.

HOW WE CREATED MARK OF THE NINJA WITHOUT (TOTALLY) LOSING OUR MINDS

Jeff Agala (Klei Entertainment)

MARK OF THE NINJA was, without question, Klei Entertainment's most ambitious game. Art, technology, design, and audio domains were challenged in ways they had never been challenged before. Klei managed to build the game in 16 months, without significant overtime, to a level of quality it was immensely proud of. Creative director Jeff Agala will discuss how the team grew and arose to meet these challenges, delivered a great game, and did it without destroying themselves.

Localization Summit

Game localization is a vital function of the ever-expanding global game industry, as it's responsible for half of the industry's total revenue stream. Successful game publishers and developers realize that localized versions of their games can drive revenues and increase international appeal. The Localization Summit at GDC is supported and organized by the IGDA Game Localization SIG, and it is aimed at helping localization professionals as well as the entire community of game developers and publishers understand how to plan and execute game localization and culturalization as a part of the development cycle.

ENGLISH IS A LOCALIZED LANGUAGE: TRANSLATING DIABLO III

Andrew Vestal (Blizzard Entertainment)

Items in DIABLO III can have one of 400,000 possible names. How do you translate and test this huge pool of randomized content? By developing flexible systems up front that work for every language—including English. This presentation outlines the development pipelines that allow localization to support DIABLO III'S high level of replay value and randomization. Special focus is given to conversation implementation, and the randomly constructed item and monster names. Blizzard's internally developed localization tool will also be shown. Stop translating whatever the developers hand you, and start thinking of English as another localized language!

AI Summit

The AI Summit at GDC features panels and lectures from more than two dozen of the top game AI programmers in the industry. Organized as a collective effort by the AI Game Programmers Guild, this event promises to give you an inside look at key architectures and issues within successful commercial games, as well as let you eavesdrop on conversations, debates, and rants on how game AI can move forward. This summit is targeted toward the intermediate to advanced programmer who wants deeper insight into the world of game AI, but anyone who is interested in what AI can offer next-generation games will find invaluable insights and lessons from the speakers.

Some of the latest and greatest techniques will be on display in the AI disciplines of navigation, pathfinding, animation control, behavior selection, group tactics, and strategy. There will be sessions on AI tool development and the issue of crafting scalable AI architectures.

While the AI summit is targeted toward the intermediate-to-advanced programmer who wants deeper insight into the world of game AI, anyone who is interested in what AI can offer next-generation games will find invaluable insights and lessons from the speakers.

OFF THE BEATEN PATH: NON-TRADITIONAL USES OF AI

Stephane Bura, Jeff Orkin, Ian Horswill, and Leif Fogred

In the game industry, AI is typically thought of as a collection of simple tools used to make characters "do things." This lecture will show three different ways that people have leveraged more esoteric AI techniques in manners not traditionally seen in games. Jeff Orkin (MIT Media Lab) will show how he used data-capture of players from his project,

The Restaurant Game, and how data-capture can be used to generate not only actions, but procedural dialog as well. Ian Horswill and Leif Fogred (Northwestern) will show how constraint-based procedural level design for roguelikes can generate content, yet still satisfy designers' needs and desires. Lastly, Stephane Bura (Storybricks) will discuss using AI techniques to contextually parse the player's actions, to give him or her more control over NPCs.

Game Narrative Summit

The Game Narrative Summit is new to the GDC after seven years at GDC Online in Austin. It covers interactive narrative in all its forms, from triple-A blockbusters to indie games to transmedia projects. The event features an all-star lineup of speakers from every corner of the discipline, with session content ranging from the advanced and theoretical to hands-on workshops for writers, designers, and others seeking to hone their skills. We hope the Game Narrative Summit will attract attendees from all over the world with a passionate interest in the ongoing evolution of interactive storytelling as a driving force in the future of entertainment.

SEVEN (OR SO) TECHNIQUES FOR WRITING A MORAL GAME

Richard Rouse III (Microsoft Game Studios)

Writers have wanted a strong moral component to game stories for almost as long as digital games have existed. Yet it is easy to veer too far to one extreme and become tedious and pedantic, or lean too far in the other direction and have players miss your moral themes entirely. This talk follows up on the inspirational GDC talk, "Seven Ways a Video Game Can Be Moral," and lays out the nuts and bolts on how to write moral stories for games. The talk will pull examples from some of the best moral storytelling games, and look at the successes and failures of the speaker's own projects.

GDC Education Summit

At the 2013 GDC Education Summit attendees will explore experimental and inventive educational approaches that established game curriculum builders can bring back to their faculty and classrooms. This program is aimed toward educators from established game development programs or new game course creators who want to understand the challenges they'll face in the next few years. It will bring scholars together with experienced professionals willing to learn and share ideas and achievements. The summit is a great professional development opportunity that will explore how collaboration leads to success not only in the classroom but in all aspects of work and life.

PLAY, MAKE, APPRECIATE: A GAME EDUCATION MANIFESTO

John Sharp (Parsons The New School for Design), Colleen Macklin (Parsons The New School for Design)

Over the last 20 or so years, game educators have collectively developed a game design and development education model based on engineering and studio art traditions. This has served our students well enough, and has provided the game industry with the next generation of employees. But should we be satisfied with the status quo? Macklin and Sharp suggest that we are just getting started. They propose a broader consideration of games and play in higher education—the Play, Make, Appreciate Manifesto. Join them as they question game puritanism and the fetishizing of games at the expense of play, as they ponder how we can increase enrollment and strengthen our position within academic institutions, and as they suggest ways to bring more diversity to our student populations.

QA Summit

This summit will discuss new and/or current tools, processes, and organization methods being used in Quality Assurance (QA) today, and will show how QA is an integral part of development. What works for some might not work well for others. What used to work may not hold up with the growing culture of smartphone, social, and online gaming. Attendees will gain some new insight and knowledge that will help existing QA teams/managers as well as those new to QA. Every attendee will leave with his/her own list of guiding principles to build QA practices upon.

BIOWARE QA: PARTNERS IN DEVELOPMENT (EMBED MODEL)

Tulay Tetiker McNally (BioWare), Arone Le Bray (BioWare)

This presentation will explore the working relationship between BioWare QA and the BioWare development team. We will explore the challenges encountered while implementing the integrated QA model, and the pros and cons that have resulted from the embedded QA model. The session will conclude with a specific case study describing how the embedded model at BioWare applies to the narrative creation process (VO, writing, cinematics) on MASS EFFECT 3 and how QA was able to find issues early on. This made it easier to avoid costly changes in content at later stages of development, and ultimately provided the end user with a better experience.

Smartphone & Tablet Games Summit

The Smartphone & Tablet Games Summit at GDC 2013 brings together top game developers from around the world to share ideas, discuss best practices, and consider the future of gaming on mobile platforms, including iOS, Android, Windows 8, and more. This two-day program will focus on the nuts and bolts of great game design and successful business strategies specifically tailored to popular smartphones and tablets.

THE ITERATIVE TOUCH: CRAFTING THE CONTROLS FOR INFINITY BLADE: DUNGEONS

Nick Cooper (Epic Games)

Touchscreens have become increasingly common input devices for gaming over the past decade, especially with the ubiquity of smartphones and tablets. With INFINITY BLADE: DUNGEONS, Epic Games set out to develop a triple-A, action-oriented dungeon crawler for iOS. It created a game that appeals to both casual and hardcore gamers, by utilizing the strengths of the touchscreen to develop an intuitive and responsive touch-based input scheme.

Free to Play Design & Business Summit

Over the past few years, the free-to-play (F2P) model has revolutionized the game industry, from social games, to hardcore MMOs, to open web and mobile games in general. Developers and publishers are grappling with the implications of service-based games, metrics-driven business, A-B testing, viral user acquisition, whales versus minnows—a whole new way of doing business. Designers are grappling with the creative implications of a world in which gameplay and monetization are intimately intertwined, the social interactions between players, how the game grows in depth without losing its core audience, and how to keep the audience engaged in games with no ending.

F2P games already dominate the web and mobile ecosystems and are expected to become a major disruptive force on next-generation consoles. Are you facing this brave new world with anxiety and questions? Then come to the Free-to-Play Design & Business Summit, where leading professionals from the F2P world can offer stress relief and provide answers.

OPERATING WESTERN GAMES IN CHINA

Jim Feng (Tencent)

Jim Feng will share his experiences and insights from leading the operations of LEAGUE OF LEGENDS in China, and from working with a Western developer, Riot Games, to bring the game to market. Takeaways will include best practices for both the operation and development of titles for the Chinese market, and the process of working with Chinese game operators.

DESIGNER ROUNDTABLE TRIPLE-A, FREE-TO-PLAY

FIND OUT HOW FREE-TO-PLAY GAMES TRIBES: ASCEND, PLANETSIDE 2 , AND MECHWARRIOR ONLINE ARE MAKING INROADS INTO CORE PC AUDIENCES

By David Daw and Patrick Miller

Free-to-play is nothing new for many core PC game audiences, with Valve's TEAM FORTRESS 2, Wargaming.net's WORLD OF TANKS, and Riot Games's LEAGUE OF LEGENDS championing microtransaction-based models for a few years now. However, in the latter half of 2012 we saw three new entrants in the free-to-play market, each bringing back classic boxed PC franchises in a new free-to-play format: TRIBES: ASCEND (Hi-Rez Studios), MECHWARRIOR ONLINE (Piranha Games Inc.), and PLANETSIDE 2 (Sony Online Entertainment). *Game Developer* caught up with Todd Harris (Hi-Rez COO), Matt Higby (PLANETSIDE 2 creative director), and Bryan Ekman (MECHWARRIOR ONLINE creative director) to try to unpack each of their strategies for cracking the core PC game market.›››

MECHWARRIOR ONLINE.

TRIBES: ASCEND.

Game Developer ***Is there anything about your respective IPs that made them particularly conducive to a free-to-play (F2P) business model?***

Todd Harris Um, no. I would actually say it was probably not that expected that we would make it free-to-play with that particular franchise. I think it kind of had maybe the opposite reputation. We're the fourth TRIBES game for the PC, but it had been a while since there had been a TRIBES game. They were known for being quite hardcore and for having a pretty passionate group of veterans that still played the old game, and also known for being all about the multiplayer, not single player, so things like balance and any perception around pay-to-win would be a big deal. So I actually think TRIBES was not an IP that people would have expected to go F2P, and we saw that as a challenge.

Matt Higby One of the best things about it, I think, is that a lot of times you're making an F2P game, and a lot of the people coming in and getting enjoyment out of your game aren't really doing a lot for you unless they're buying stuff from the store. As a developer, you're not getting much out of the people that are playing for free, unless you can entice them into buying something. With PLANETSIDE, that's not true; everybody that jumps in and plays PLANETSIDE is actually providing content for all of the people that are playing PLANETSIDE with them.

Bryan Ekman The nature of the MECHWARRIOR IP allows us to attract a large player audience; who doesn't like giant robots? Once a player gets through the basics of learning how to pilot a walking tank, they will find a very deep and engaging experience that allows you to tinker and customize your BattleMech (avatar) endlessly. The nature of a BattleMech gives us plenty of opportunities to monetize non-pay-to-win (P2W) concepts, such as time-savers and cosmetics, that add real value to your battlefield persona.

***

Game Developer ***Was the game originally conceived as a free-to-play game, or was that model added during the development process?***

Todd Harris Step one was that we wanted to make a game that had the gameplay of Tribes—specifically, a fast-paced shooter with jetpacks. We liked that gameplay, so first we wanted to make that type of game. Second, we were fans of the game, so we looked into buying the IP and we ended up doing that so we could make a Tribes game. It was originally envisioned as a one-time purchase, but then throughout the development cycle we shifted toward making it free-to-play.

Matt Higby: When we first set out to make PLANETSIDE 2, we knew that F2P was on the table as a possibility, and as we built the game out more and more, we found all the ways that it fit, and I think one of the things that's fascinating about free-to-play is how well it fits a lot of different types of games. So as we were building the game out it became more and more clear that free-to-play was the best option for us.

Bryan Ekman When Piranha Games first started working with the MECHWARRIOR IP back in 2008, our intent was to make a traditional console product. In early 2011, we acquired the Xbox and PC licensing rights, and quickly decided to scrap the old brick-and-mortar design in favor of taking MECHWARRIOR in a new direction. And thus MECHWARRIOR ONLINE was born.

***

Game Developer ***How does your game convert a free-rider into a paying player?***

Todd Harris What someone gets value-wise is two categories of things. One of them is cosmetics. We have cosmetic skins, so you can make yourself look a little more badass, but it doesn't affect your actual game. We also have voice packs, which have been kind of nostalgic for the TRIBES players; the early games had these built-in voice quick-key commands that let you taunt or call out tactics, so we offer custom voice packs for those. Then the second thing is you get more variety faster by paying. You can unlock additional classes, so if you want to play a stealth infiltrator class, or a technician who deploys things, you can unlock new classes and new weapons for those classes much more quickly if you decide to pay.

Matt Higby With PLANETSIDE 2 one of our keystones is ensuring that we have fair competitive gameplay. So one of the things we decided is that anything that can affect gameplay in any way can always be unlocked through gameplay. You can go and purchase items, but you can unlock those items through gameplay too. I think if you're making a competitive free-to-play game, that's a must-have.

The main thing we're doing with monetization right now is convenience. People who might not have 40 or 50 hours to play games anymore (like you might have had if you were me when I was in college) can use microtransactions or purchase a membership to unlock items more quickly. Also, we have lots of cool cosmetic things. Since those don't actually affect gameplay, those are sold pretty much exclusively for Station Cash (currency purchased with in-game money --*ed*) in the game, and we find a lot of people actually have those so they can be a little more distinctive. For us, though, the most important way to turn a free player into a funded player or a paid player is to just have a fun game. We know that having a game that people can log into every day and have fun in every day—have an enjoyable experience—that's the thing that convinces them to spend money more than anything else—including even what we can offer them to spend money on. The players feel like the game is fun, and they want to be able to support the developers of the game, and we see that being a true thing within our community.

Dontnod Entertainment Brings Neo-Paris to Life with Unreal Engine 3

As next-generation hardware approaches, Capcom is launching a brand new franchise, having been captivated by French developer Dontnod Entertainment's *Remember Me*, a science fiction action adventure that introduces a sexy elite memory hunter named Nilin. Set in Neo-Paris circa 2084, the game introduces new gameplay elements that involve entering the minds of characters. Dontnod is using Unreal Engine 3 (UE3) technology to bring this experience to PC, Xbox 360 and PlayStation 3.

"Technology is what you make of it, and we are in a very technology-driven industry," said Jean-Max Moris, creative director, Dontnod Entertainment. "We have the luck of having extremely talented concept artists, art directors and 3D artists who have created this wonderful, contrasted, colorful world of Neo-Paris with the Unreal Engine. We've made this game look very distinct, proving that it's really about what you do with the technology. This industry is content-driven and Unreal allows us to focus on that."

Moris said choosing UE3 was a "no brainer." Dontnod Entertainment CEO Oskar Guilbert said from the get-go, "We're going to have Unreal. We're not going to re-invent the wheel. We're creating a new IP, a new studio, with a new team. If we had added new technology to that, I can assure you, we wouldn't be here today."

"During our evaluation period we were impressed with its features and middleware to support a number of specific needs," said Dontnod's technical director Jerome Banal. "As UE3 is highly popular among artists and other programmers, it is easy to find people in France who are familiar with the engine. This considerably reduced any training costs and meant we had a team ready to begin development from the get-go."

Dontnod has worked closely with Epic Games' engineering team at various points throughout the development process. The studio appreciated being able to work to its own schedule during the evaluation period while they consulted technical documentation and became comfortable with the engine's source code. Seeing the project's promise, Epic made sure that Dontnod's UE3 evaluation was extended for as long as they needed while building the game and polishing it to the extent required to secure publisher funding.

In addition, Dontnod has been able to rely on the Unreal Engine development community. "The programmers have utilized the Unreal Developer Network (UDN) and mailing list a lot," said Banal. "Some art departments have their lead or technical director summarize the content into a set of step-by-step instructions or checklists, as it can become very technical for pure artists, but the TDs are very happy that such information is readily available."

All of that communication has resulted in a game that really stands out from a visual perspective, while pushing gameplay into new directions with its marriage of action and the unique Memory Remix mechanic, which Moris describes as "a super dynamic puzzle that is tightly interwoven with the narrative and the concept." Entering a character's mind can have huge consequences on the game's story.

"We modified Unreal Kismet to transform it into a finite-state machine with a lot of features inspired by UnrealScript, and coupled this with a large use of prefabs," said Banal. "Together, it makes it easier for level designers to script the game and reuse various scripts, thereby improving productivity and reducing bugs and costs.

"We switched the main shading model to a physically based model similar to the one presented by Tri-Ace to make the rendering even more realistic. We have added support for fast pre-computed and dynamic perspective-correct reflections – Epic's *Samaritan* real-time UE3 demo inspired this technique which we presented at SIGGRAPH; our version has in turn inspired the UE4 team. These new reflective features make our wet world more believable. We also developed a memory-friendly rewindable particle system to make Memory Remix even more impressive."

Remember Me also makes extensive use of Unreal Matinee. Banal said the whole Memory Remix gameplay is very cinematic and relies a lot on Matinee, with a few of Dontnod's own internal enhancements. Multiple matinees are played at the same time, then selected and synchronized according to the player's inputs. The real-time preview feature allowed iterating quickly over the scenes and the camera work, which was essential to achieving a superior level of quality with a small team.

Like many other studios, Dontnod has used UE3 to develop *Remember Me* simultaneously across platforms. Banal said the engine enables the majority of development to be led on PC with an easy-to-use pipeline that facilitates quick testing on console. In particular, he said, UE3's multi-threaded cooking has been invaluable for cross-platform development.

The end result of all this effort is a new game that pushes the envelope of the action adventure genre. Aside from its gorgeous future Neo-Paris locale, entering the minds of characters adds a fresh twist to the genre. And the entire game came together, creatively, thanks to UE3 technology doing the heavy lifting, allowing Dontnod Entertainment to focus on raising the bar.

Thanks to Dontnod for speaking with freelance reporter John Gaudiosi for this story.

UPCOMING EPIC ATTENDED EVENTS

SXSW Interactive
Austin, TX
March 8-12, 2013

Game Developer Conference
San Francisco, CA
March 25-29, 2013

Please email licensing@epicgames.com for appointments

© 2013, Epic Games, Inc. Epic, Epic Games, Unreal, Unreal Developer Network, UDN, Unreal Engine, UE3, UE4, Unreal Kismet and Unreal Matinee are trademarks or registered trademarks of Epic Games, Inc. in the United States of America and elsewhere. All other trademarks are the property of their respective owners. All rights reserved.

Bryan Ekman First, we focus on getting the player engaged and teaching them the mechanics of what makes MECHWARRIOR ONLINE fun and refreshing. Then, after a player has learned the basics of piloting, tinkering in the MechLab, and customizing their BattleMech, they discover a cash store full of items filled with fun or advantageous items to purchase. That said, it's still possible to have a free and fun MECHWARRIOR ONLINE experience.

***

Game Developer ***What kind of in-game metrics and analytics tools do you use to measure your game's health?***

Todd Harris I think what's most interesting, from a developer standpoint versus a business view, is all of the game design metrics that we collect and that any F2P developer can collect. So yes, we look at retention rates, and monetization rates, and what's selling, but our designers have access to really, really detailed data on the strength of every weapon. They can look at, for instance, the kill-to-death ratio of the nine different classes in the game, and whether those ratios are actually bearing out in reality as we would expect from the game design. With a single day's worth of data, our design team can see enough statistically relevant data to see if any design changes are working as intended or not. So that's really what's most exciting to the game design team. Every match you play in TRIBES, that data, in terms of how many kills, what weapon you used, how effective those weapons were, how effective your team was—all of that is being captured in a persistent database, and our designers can use that data to improve the game.

Matt Higby We have very extensive monitoring and metrics tools in PLANETSIDE 2 for us to figure out stuff like how many people logged in today, logged in yesterday, and percentages of falloff of people. Also every kill that happens, every death that happens, we track and we can filter that through a variety of tools to figure out balance—figuring out which areas of the game people play and stick around with. So yeah, data gathering and metrics for a game like this, where we're planning on making changes for years to come, being able to track all our metrics and what people are doing and what we can do to make people keep playing is really, really important.

Bryan Ekman We designed our own proprietary telemetry system that logs pretty much every user action in the game, from where they click, to how well they do in each match, to how much they spend and when, to their average FPS. Our community of players also gives us regular feedback and has been a huge asset in the Closed and Open Beta phase.

***

Game Developer ***How do you decide what to charge for and how much to charge? Is there a coherent philosophy behind your monetization design?***

Todd Harris The philosophy that we started with in TRIBES was that we wanted it to be relatively less expensive in terms of time or money to unlock different classes, so that various roles on the battlefield would be filled up pretty quickly. Then, in terms of weapons, we wanted there to be more progression involved in terms of player time or money. So classes first, weapons second, just so there would be diversity on the battlefield. Beyond that, it's fairly metrics-driven, and we do a lot of experimentation in terms of price points.

Matt Higby I think it's a feel thing. I don't think there's really a formula that you can plug stuff into to figure it out exactly. It has to do with how many items you're going to allow people to unlock. What sort of progression is involved in unlocking items? What's the gameplay associated with unlocking? For us, as you're playing PLANETSIDE and getting kills and capturing bases and all that stuff, you're earning stuff that you need. You don't really need to go out of your way to do stuff that's not fun, or not part of the core game, to get the points that you use to unlock new items. So the very core of the entertainment experience of the game is also helping you progress your character and unlock new stuff. But we set kind of a wide range of prices from things that are like 50 cents to, I think our highest-priced items right now are bundles that give you multiple items for around 10 bucks. At the end of the day, with a free-to-play game, the best possible thing you can do is make people feel good about the purchases that they're making. Make them feel they got a good value for what they're spending, and that they're supporting a game they enjoy. If you can accomplish those two things I think you can be successful in the free-to-play space.

Bryan Ekman We're still working out how elastic our economy is, testing a variety of price points, value propositions, and rarity. We generally start with a theory on value, and the player confirms (through a purchase or not) the value of an item, and how much would they be willing to pay for it. Then we test the theory and analyze the results. Based on early Closed Beta data, we tuned our prices and content toward the results of these test. Now that we're in Open Beta and seeing true user-buying habits, we've tweaked a few of our original theories, most notably by adding both temporary and permanent buying options.

***

Game Developer ***Lots of rather vocal players toss "pay-to-win" accusations around. From a game-design perspective, how do you monetize your game effectively without being labeled as P2W?***

Todd Harris I think at the end of the day it's all a matter of degrees and perception, because in the West, the client downloadable games at least are rarely in-your-face pay-to-win, so it's a matter of degree and perception. And for us, because TRIBES is known as this high-skill-level game, there are really three rules that we had. First, you can acquire anything that affects the gameplay by playing the game, so it's really a time-for-money tradeoff. Second, we wanted anything that you could unlock to really be sidegrades—not necessarily better weapons, but just a different play style. Third, just the way the game is designed, whether it's free-to-play or not, it does really depend a lot on player skill. Very specifically, you're moving around like crazy, you're having to lead your opponent with most weapons over a very, very large battlefield. So a skilled player with just the free weapons can and will beat a new player that has all the items unlocked just because of the nature of the game.

Matt Higby Nobody wants to be a pay-to-win, but it's almost impossible to define what pay-to-win is, because it's a really personal decision that you're going to make. Some people are going to be real hardliners about it, and will call your game pay-to-win if it has anything that costs real money that will give you a boost to how much experience you earn, or unlock an item that does anything noncosmetic for you. Other people, I guess, are far more liberal in their definition of what they think pay-to-win is, to where if I have to pay money to buy a tank that does twice as much damage, then that is pay-to-win. The tank, to us, would be a really egregious example of P2W, and we would absolutely avoid doing anything like that. But because it's such a personal decision for the player, it's really hard to make those kinds of determinations.

We've done our best to make sure our business model is completely fair, and I think we have a really fair non-pay-to-win business model that still allows people to make shortcuts and unlock items in a fair way. But of course, since it's your own personal opinion,

PLANETSIDE 2.

there are still going to be people out there who say that we're a pay-to-win game. You can't please all the people all the time.

Bryan Ekman You have to be very creative and disciplined about adding perceived value, without adding too much of any power. We found a nice balance with our Hero Mech design; a unique BattleMech variant can be designed with special properties, which are not viewed as overpowered, and thus not P2W.

***

Game Developer *When are you "done" building an F2P game? Do you ever really get finished?*

Todd Harris That depends. The thing that's the biggest change is that in the packaged-game business, what people defined as "done" actually equals "start" for the free-to-play game business. You're really only starting once you have real gamers using your system. The analogy that I think is accurate that people talk about a lot is that packaged games are like movies and free-to-play games are more like a TV series. So when's a TV series done? Well, sometimes it's a commercial decision, sometimes it's a creative decision, sometimes it's a production-related decision.

Matt Higby For certain aspects of the game, I think you certainly do. We're always striving to achieve balance and create new exciting things for our players to enjoy. So nothing ever really finishes. One of the things I was telling people a lot just before the game shipped is that I have no idea when the game's going to be done. And that isn't really an important question to me because we're continually adding stuff, constantly thinking about things that can go in the game. So "finishing" the game isn't really a concern. We have years of tasks in the backlog right now for things that we plan on adding to the game, and that can be a bit intimidating at first when new people are coming from more traditional game development into making these longer-term games... It takes people a little while to get used to the fact that you don't just ship a project and then just move to the next product; you're continually working on building out and enhancing the game that you just made. Once you've gotten used to that, though, it's a great feeling because the sky's the limit. There's never a "We only have a few months to get this game finished" thing. It's a "Well, maybe next year we can really work on that cool thing that's going to take a lot longer than two months out" thing. There's never a closed door.

Bryan Ekman I'll let you know when it happens! (Perhaps never...) MECHWARRIOR ONLINE is a living, persistent game.

***

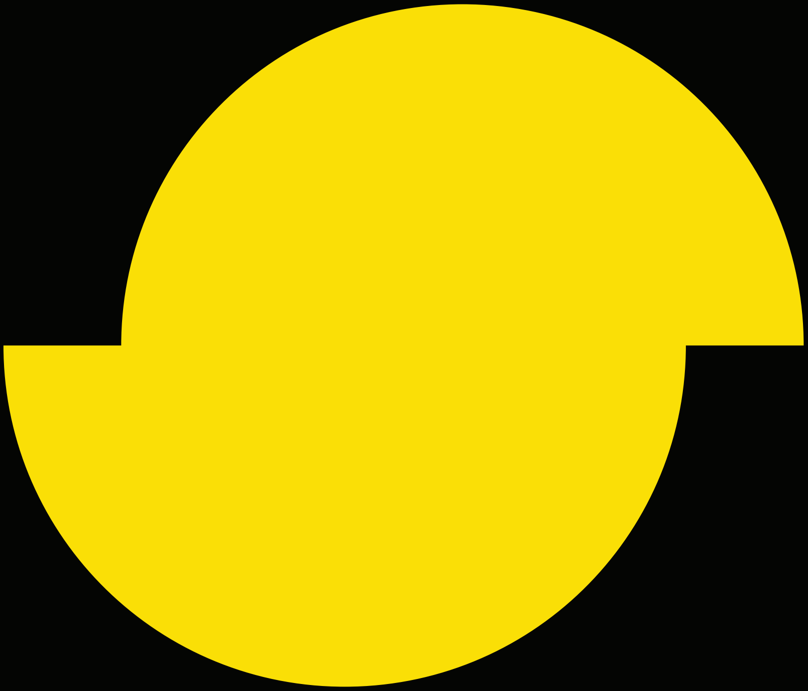
Game Developer *How does a free-to-play game's dev cycle compare with a more traditional boxed/one-time purchase dev cycle?*

Todd Harris I think that with free-to-play games you typically go out into the marketplace earlier. Once you feel like your most critical mechanics are there, then you're more likely to go out and have players earlier in the cycle, then you're adding scope, both features and content, for a much longer period of time. The most successful free-to-play games out there continue to get content updates four or five years after the players first started playing them.

Matt Higby One of the things that I love as a designer working on a free-to-play game is that my job every single day is to come into work and figure out ways to make the game awesome, and make the game more fun. Because at the end of the day, if people are logging in every day and enjoying it, then we have a chance to maybe get them to buy some cosmetics, or buy a membership because they love playing the game. That's a really great feeling. I don't really need to worry about making a demo that's going to trick people into buying my $60 game, I just have to worry about making sure that the game is really, really fun—and as a developer, that's really, really fun.

Bryan Ekman So far, the free-to-play dev cycle is completely different than the traditional boxed/console model. There's always more work to be done. As time goes on, further refinement of our processes will lead to segregation between live ops and feature development. Eventually a portion of the team will be shared to work on regional versions, and new game concepts.

Automagic 3D Optimization



MeshLOD. ProxyLOD. BoneLOD. MaterialLOD.
Simplygon is the leading tool-chain middleware for automatic optimization of 3D-game content and Level of Detail. By replacing tedious and time-consuming manual work, Simplygon offers the benefits of LODs but without the drawbacks of increased production time and development cost.

www.simplygon.com

SIMPLYGON™

Simplygon is used by industry-leading developers including

AniPark, Avalanche Studios, Cryptic, CCP Games, Doobic Game Studios, Epic Games, Funcom, Giant Interactive, IMC Games, Nexon, Pearl Abyss, Piranha Games, Quantic Dream, RealU, RedGate Games, Reloaded Studios, Joymax, Neowiz, XLGames, Wemade Entertainment.

INTEGRATED PARTNER

Donya Labs, Simplygon and the Simplygon logo are all trademarks or registered trademarks of Donya Labs AB. All rights reserved. All other trademarks are the property of their respective owners.
Copyright © 2013 Donya

LET'S TALK ABOUT TOUCHING!

HOW TO MAKE TOUCHSCREEN CONTROLS FEEL AS GOOD AS (IF NOT BETTER THAN) A GAMEPAD

By Tim Rogers

I have a lot of fond memories of pressing buttons. As a six-year-old, it felt like magic to turn a television on with a wireless remote control for the first time. If I am not mistaken, you are not six years old, so it's fine if you object to the following claim: Buttons are doomed; touchscreens are the new game controllers. I've been working on making mobile games as the founder of an independent studio called Action Button Entertainment. In order to make the best mobile games, I've been dissecting and researching every interesting game-control mechanic I can find, from PONG to ANGRY BIRDS. Here is what I've found. ›››

PUSHING BUTTONS I've always been a proponent of the mechanical particulars of a game's feel over any sort of gimmick related to its product construction. The triumph of SUPER MARIO BROS. was one of Game Design by the Milliseconds—of the developers pre-understanding the game as more than a series of short-, medium-, and long-term goals. SUPER MARIO BROS. is about the immediate-term goals, and the way that the player's microscopic actions feel in increments of five or six milliseconds.

SUPER MARIO BROS. felt like magic. The fine degree to which the minute variations of button-press length could affect Mario's jump heights and lengths checked every box in my child-brain's "best thing ever" wish list. More than 20 years after the Nintendo Entertainment System, we had the Nintendo Wii: Shake a little television-remote-like thing any which way to make a cartoon person hit a tennis ball. Nintendo was shifting the emphasis, aiming for the part of the brain that makes a six-year-old find a television remote control more magical than ASTEROIDS.

Games With Buttons are not superior by default; they are superior because a parade of geniuses like Shigeru Miyamoto laid the groundwork. Players needn't wage a culture war of casual versus hardcore, social versus everything else, mouse and keyboard versus controller; in a perfect world, action gamers would only speak scientifically of the milliseconds of a game. If a game's milliseconds unite its action and its player, then the game is real and true.

In order to understand the touchscreen-versus-button dichotomy, let's revisit the old mouse-and-keyboard-versus-controller debate: You can't click on a recent blog post about or review of HALO 4 without accidentally scrolling down to the comments and seeing someone groaning about how they'll never play a first-person shooter on a console because "mouse and keyboard is the only way to play an FPS." This opinion has raged since the moment HALO was announced as an Xbox exclusive.

I am convinced that we could get a room full of theoretical physicists to prove that Halo does a pretty darn good job with a controller, and that CALL OF DUTY'S by-the-millisecond design concessions for controller players (such as the smart snap-to auto-aiming) add up to a game that would be just as competitive with a controller as it would be with a mouse and keyboard—that is, if all the best first-person-shooter players weren't born and raised with a mouse and keyboard in their hands. As the mouse disseth the controller, so does the controller disseth the touch screen. (And we all diss motion controls, but that's a topic for another day.)

And maybe most touchscreen games deserve the disses.

I see a lot of games with virtual buttons on the screen. This is always a mistake. That's a sign of a subliminal conspiracy between game developers—everyone on the team silently agreeing to commit to an inferiority complex: "[Sigh.] It sure would be cool if we were making a game with buttons."

WHERE'S MY PERRY?

Know this: A friend was telling me just the other day that his four-year-old son just doesn't want to touch a game controller. This friend has a glorious collection of old and new game consoles in his many-televisioned house, and all his son wants to do is play WHERE'S MY PERRY? on the iPad. "Controllers just aren't real games to him," he told me.

I know "hardcore gamers" that will spend an hour of their lives trying to make a SKYRIM avatar that looks exactly like themselves, and then they'll say mobile games aren't "real" games because their fingers get in the way of seeing the screen. This is interesting. Personally, I prefer IMAX to 3D movies, because I like feeling like I'm inside the movie, rather than feeling like the movie is popping out at me.

Now, imagine the way a four-year-old child feels playing with a touchscreen: The child touches her fingers to the screen, and the simulated world reacts. The child can literally touch her favorite cartoon character, and watch that character move. How is that not superior to pressing a button over here and watching the character move inside that screen over there?

Modern touchscreen technology has closed the distance from which children will consider electronics magical. For a four-year-old—one who, in 10 years, will be a 14-year-old buying the games you're hopefully still making—your remote control simply won't cut it.

PATIENT ZERO: PONG Designing essential game mechanics for touchscreens requires an understanding that hardcore action has never, ever been "about" the control method. It's about the way the action interacts with the player's brain. The control method is only ever an instrument for fabricating that brain-screen coordination.

Let's consider PONG. It's a hyper-competitive, finely nuanced contest between two players. The control implements are nothing like modern video game controllers. Players twist a tiny, mosquito-bite-sensitive knob. Twist a tiny bit clockwise, and your paddle zips to the bottom of the screen. Twist a tiny bit counter-clockwise, and the paddle zips to the top of the screen.

Furthermore, the paddle is made up of eight segments that appear to be seamless: The part of the angle of the return depends on where the ball contacts the paddle. Throughout a PONG contest, the player must balance the urge to trick their opponent against the urge to fire a return which is not so tricky as to result in any tactical backfires if returned. I like to think that the essence of all hardcore action games is purely available in PONG.

What I most take away from PONG is the relationship between the paddle and the control knob: The delicate sensitivity creates a brain impression that the game is more than just something happening on the screen. It's an object with a physical presence on planet Earth.

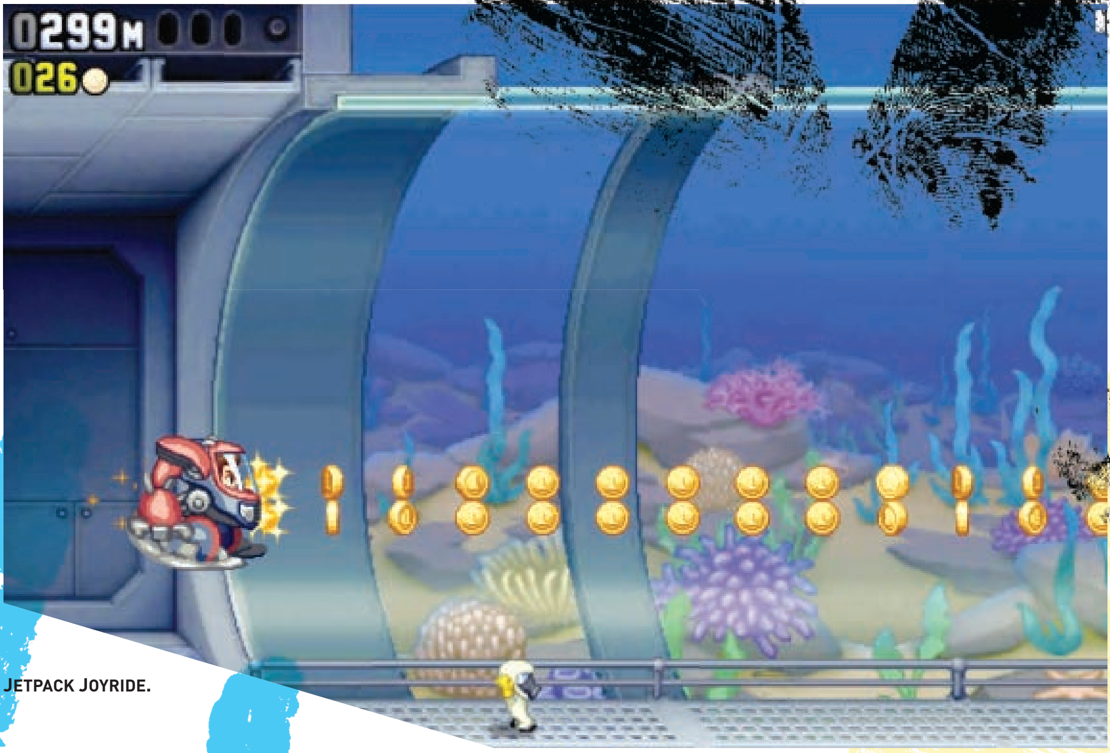
That brings us back to the image of a child, holding a screen, touching her favorite cartoon character, and watching that character react, like magic.

CASE STUDY #1: JETPACK JOYRIDE You might have played JETPACK JOYRIDE. It's a casual, free-to-play, monetized, mobile game—all those buzzwords—though it's also an action game, and its central action is tuned to a level of nuance on par with any individual gun in HALO.

In JETPACK JOYRIDE, the player touches the screen—anywhere—to fire the character's jetpack. The longer the finger contacts the screen, the longer the jetpack fires. A short burst and then release will move the player-character less than halfway up the screen. A longer hold will bump the character's head into the ceiling. To obtain skill at the game, the player must master tapping and releasing as obstacle-context requires, juggling the character eternally between positions that are neither precisely the top nor precisely the bottom of the screen.

JETPACK JOYRIDE is tenaciously feel-balanced. The speeds of by-jetpack ascent

infinity ward
IS HIRING AT GDC
WANT TO JOIN THE STUDIO BEHIND CONSECUTIVE BLOCKBUSTER HITS?
EMAIL US AT JOBS@INFINITYWARD.COM TO MEET UP AT THE CONFERENCE.
WWW.INFINITYWARD.COM/JOBS
CALL OF DUTY
CALL OF DUTY 2
CALL OF DUTY 4 MODERN WARFARE
CALL OF DUTY MODERN WARFARE 2
CALL OF DUTY MW3



JETPACK JOYRIDE.

and by-gravity descent are locked in a tastily lopsided yin-yang. JETPACK JOYRIDE is what happens when a game gets even one thing truly, actually, perfectly right: millions of downloads.

I first played JETPACK JOYRIDE as someone who had tremendously enjoyed CANABALT, the pioneer of the "endless runner" genre. CANABALT is another game that gets one thing truly, actually, perfectly right. CANABALT is about tap-hold-duration-sensitive jump heights with inevitable gravity. It's about the beauty of parabolas. (Action games are almost universally about parabolas.) As your character accelerates, his maximum and minimum jump distances grow; the shape of his minimum jump parabola becomes an oddly stretched caricature of its original self. The game becomes "unpredictable" until you master the feel of it.

CANABALT and JETPACK JOYRIDE are games that most players have experienced exclusively with a touchscreen, though they are also highly playable with a mouse—as their Flash and Facebook versions, respectively, indicate. Therefore, it's probably better to call them "one-button" games; they work well on a touchscreen, since a touchscreen is a big button, but touchscreens do not have to be one-button games.

In all sincerity, I say that CANABALT and JETPACK JOYRIDE, two games whose creators I presume love SUPER MARIO BROS., have actually demonstrated a masterful understanding of that game's playful friction. But is it possible to make a one-button SUPER MARIO BROS.? By god, no.

SUPER MARIO BROS. is not just about the fine manipulation of jump parabolas through button-press durations; it's also about direct manipulation of running speed. When I was a kid, you knew another kid knew what was up in SUPER MARIO BROS. if he was holding the controller perpendicular to his torso, buttons out, thumb keeping the B button depressed. Narrating the experience of parkouring through World 8-3 is no easy task: It'd be a Morse code of taps and releases of the A and B buttons, with B held down most of the time and only let go for 15 to 20 milliseconds at a time as obstacles or enemies demanded it. SUPER MARIO BROS.'S mechanics are deceptively rife with moving parts, and there's probably at least two whole textbooks to be written about the friction when Mario slides to a stop before changing direction.

If we remove the ability to change direction from SUPER MARIO BROS., and make it a two-button level-based runner, where one button is "continuous horizontal thrust/acceleration" and the other button is "jump," well, now we're getting somewhere. Touch your left thumb to the left side of the screen to make your character accelerate; release to decelerate. Touch your right thumb to the right side of the screen to jump. (I can't think of any mobile games that use this scheme, though if you've made one, please tell me—I'd love to play it.)

Now we just need to super-fine-tune the acceleration rate, the top speed, the deceleration rate, the jump-height acceleration, the jump-height cutoff, the speed of gravity, and craft interesting multiple-level paths and enemy behaviors. Do all that, and you've made something pretty SUPER-MARIO-ish without using more than two buttons or a touchscreen.

Of course, this two-button SUPER MARIO BROS. would also be about 27 Rubik's Cubes-worths more complicated than CANABALT and JETPACK JOYRIDE duct taped together. It'd be like CANABALT: STICK-SHIFT EDITION. Developing it into a successful product would be a nightmare (if not multiple nightmares).

CASE STUDY #2: FASTERBLASTER

During the prototype phase for my studio Action Button Entertainment's upcoming hyperkinetic touchscreen mindsport FASTERBLASTER, I hit a brick wall in my attempts to explain the way the player's avatar should rotate.

The avatar is, at the end of a movement, always pointing upward; the device is always in portrait mode; the camera is rotating along with the avatar.

Our second prototype made us seasick, because the camera rotated at the exact speed as the avatar, and the avatar's direction related one-to-one with the position of the player's finger on the bottom edge of the screen. To return to the PONG example, it would be kind of like tweaking a PONG knob one millimeter counterclockwise, and then finding yourself suddenly standing in the arcade parking lot.

Only enemy formations could lend context to FASTERBLASTER's grotesque spiral. I had this quirky idea that the player's finger-touchdown point should correspond directly to a clock-face position. Assuming the avatar begins pointing to 12 o'clock despite no touch input, touching the left edge of the screen would point the avatar to 6 o'clock. The right edge, also, would point the avatar to 6 o'clock. Now, the very center would point to 12 o'clock; left-center would point to 9 o'clock, and right-center would point to 3 o'clock.

In FASTERBLASTER, the player charges up grenades and then lobs them at enemies who are constantly advancing from all directions. The grenades' destination point is constantly fluctuating between the tip of the triangular avatar and the upper edge of the screen. Each up-down pump increases the charge level of the grenade, which increases the radius of its explosion if it contacts an enemy.

So it's important to be able to hold onto your bullet while it's charging and shift your aiming position easily. The bullets charge quickly, and the target cursor moves quickly. The enemies move quickly, and the travel time of the player's grenades is balanced to enforce maximum friction.

It's just—that sliding. Lord, it just about gave me an aneurysm.

Part of the problem was the size of the iPhone screen. The ratio of my thumb-tip-width to total width of the screen in portrait mode to circumference of the playing field (in iPhone screen heights) was hardly golden. An iPad Mini (the target platform) is just fine, because the travel speed of the average person's finger across that real estate yields a sensible enough rotation speed. On an iPhone, though, forget it.

My pride cried. I'd believed in the FASTERBLASTER pre-prototype controls, because they'd worked so well in ZIGGURAT.

Or had they?

CASE STUDY #3: ZIGGURAT I had coffee with a friend several months after we released our debut game ZIGGURAT. His website had spoken favorably of the game, though, as my friend admitted, he "didn't really get it." I asked him to open the game on his iPhone and show it to me. He complied. "Give me your best performance." There it was: He was just tapping his finger all over the screen. "I can't see the guys," he said. "My finger gets all over the screen."

Of course, he'd skipped through the (merciful, exceptionally brief) tutorial. So I showed him how to play: Slide your finger along the bottom of the screen. Touch the middle, and the character standing on top of the pyramid points straight up. Touch the right, and the character points down and to the right. Touch the left, and the character points down and to the left. This way, you can slide to aim your shot. (**See diagram on Page 28**)

Some reviewers had said the game was "Just like MISSILE COMMAND." These reviewers had probably never played MISSILE COMMAND. (Note: I love MISSILE COMMAND.) "The game would be better with buttons," another review had said. Not that I cared about those reviews; *Edge* gave us a 9 out of 10, so whatever. The whole point of ZIGGURAT had been to make a game that was for touchscreens.

In ZIGGURAT, you hold your finger on the screen to charge a bullet. The bigger the bullet, the straighter and faster it flies. Bullets grow and shrink as they charge. You'll have to deeply understand the feel of the charge timing to get the parabola you want. We designed eight unique enemy types (many of whom are constantly jumping up and down) to keep the parabolas maddeningly varied and thus interesting. And the enemies' heads are growing and shrinking at a rate that's just different enough from your gun-charge speed to defy muscle memory: The bigger the enemy head, the bigger the bullet, the bigger the explosion, the bigger the chain reaction, the longer you stay alive, the higher your score.

At the center of all this is the player's ability to always be in control of exactly where the gun is pointing. You're not just charging and releasing shots, you're also aiming the gun left and right, and every point your finger travels through correlates exactly to a position the gun can be pointing.

Moreover, the bottom of the screen is a negative space: the silhouette of a pyramid. Your finger doesn't get in the way of anything, and the "control pad" on the screen is both contextualized and practically invisible.

I feel pretty clever for thinking of all that: I wanted to do for a first-person shooter what CANABALT had done for SUPER MARIO BROS., except we had to stop at Japanese 1980s arcade games and STARCRAFT along the way.

Not everyone loved the controls. A couple of iTunes reviews were livid at the lack of "offset controls." They wanted an invisible virtual analog stick wherever they put down their finger. I feel like this would have ruined the game: They'd get what they want, and then, unable to aim in literally every angle, their scores would suck, and they wouldn't get it, and then those emails would start coming again.

CASE STUDY #4: MAC OS X Taking ZIGGURAT'S control style over to FASTERBLASTER was a nightmare. Luckily, I had a backup plan. I feverishly typed a long email to programmer Michael Kerwin, in which I explained controls that'd work "like an old-fashioned stereo knob except not really." He said he'd need a couple hours to think it all through.

A couple hours later, he came back with the suggestion that the controls I mentioned were "like the iPhone alarm-setting wheel." "Oh," I replied. "Yeah."

If you ever meet my parents, they can confirm this: I have, since childhood, had perhaps unnecessary amounts of fun with unlikely things. Recently, one of those things is the multiple desktop switching on the more recent Mac OS X versions. Sometimes, I'll put four fingers on the trackpad and record-scratch between two spreadsheets in time with the music I'm listening to.

The best thing is, of course, "natural" scrolling in OS X. I just spent 30 seconds scrolling up and down through this document. This scrolling possesses such charismatic friction in its coasts and turnarounds. It's as finely honed as SUPER MARIO BROS., but it's smooth in its nuanced complexity to allow for perfect natural usability.

Natural scrolling accelerates to a set point relative to the speed over distance of a two-finger swipe. To halt the scrolling immediately, just touch one finger to the trackpad. I have watched many Mac users scroll and halt, and they always use two fingers for the halt, even though one finger will do. This is important: "Two fingers" is attached inseparably from any action related to scrolling. This is as close to "proof" of an interface's intuitiveness as we can get.

The most important aspect of natural scrolling is the natural deceleration. Somebody working at Apple loved that action. OS X's two-finger trackpad scrolling is to a mouse scroll wheel as Netscape is to Google Chrome. So here's where we have to talk about "friction": Without some quirk to snag an action and pull it away from "perfect usability," an "interface" cannot become a "game." The soul of game design, after all, is in assigning rules to objects.

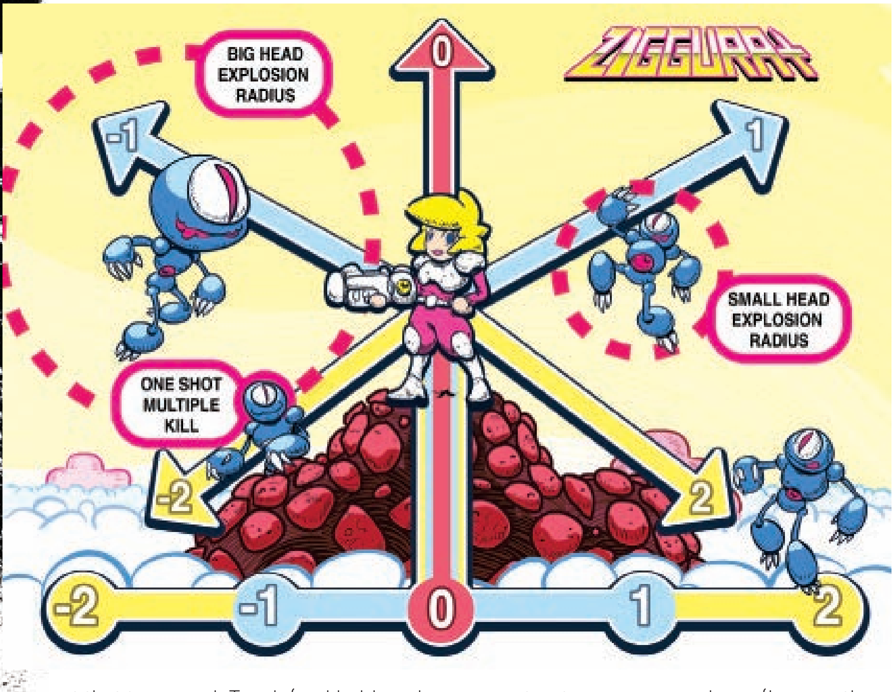
Natural scrolling in OS X halts immediately when I put a finger back on the trackpad. What if the finger on the trackpad were brakes? What if to halt the scroll, I had to input a reverse of the gesture—matching the speed over distance of my previous swipe, only in the opposite direction? Now we might have made a game: You have 10 seconds to scroll the scroll bar up to a touchdown zone; you have to land it precisely between two lines. Swipe up to accelerate. Swipe multiple times to accelerate higher and faster. Now keep your distance over time in mind when you start applying the braking swipes.

We've just started designing the Per-10-Seconds experience of a stupidly abstract train simulator or an Atari-2600-like curling game. It's probably not very fun. That's okay, because we're not actually going to make that game.

Instead, imagine you have your smartphone in landscape mode. Now imagine we're making SUPER MARIO BROS. Your character is facing to the right, and located just a bit left of the middle of the screen.

Slide your thumb up and down on the left side of the screen to move right and left. Double- and triple-swiping dials in multiple accelerations. You're controlling Mario with a scroll gesture: Swipe X distance in Y time and then release to accelerate Mario up to top speed for Z seconds, after which he starts to slow down again. Swipe X distance in Y time and then leave your finger down to keep Mario at that top speed. Touch (and hold, and release) anywhere on the right half of the screen to jump.

Now try prototyping this. So, uh, why do people use virtual buttons, again?



CASE STUDY #5: ANGRY BIRDS AND GASKETBALL You could play ANGRY BIRDS with a keyboard, just the same way we used to play SCORCHED EARTH on a PC. The up and down arrow keys pitch the angle of the bird, left and right pull back the slingshot, and the space bar fires. But by being a touchscreen game, ANGRY BIRDS offers children of all ages the joy of reaching out and literally touching their game character with their finger. That slingshot-stretching sound enhances the perception that our flesh is making something happen. It's barely tactile, but it's still tactile. The world inside the touchscreen is a little cartoon still life of a pinball machine; we wind it up and watch it go.

I'm not here to talk about ANGRY BIRDS' innovations or branding or level design (which all sort of suck, to be perfectly honest): Let's just say it feels like something. It's not something that we necessarily have to be touching, though it is popular as something that we touch.

It's safe to say that ANGRY BIRDS would not be popular without either its branding or its mechanical particulars. ANGRY BIRDS is as much a house of cards as any hundred-million-dollar property. My opinion, though, is that the controls and mechanics are as sound as the characters are ugly, and that without the surrounding game design the birds would be dead on a street corner somewhere. (I mean, they don't even have wings.)

I wouldn't be surprised if one ANGRY BIRDS-esque stuff-throwing "clone" is released on the app stores of the world every six hours, but only one stuff-throwing game stands out—and this is a perfect way to end what I'm trying to talk about in this article.

GASKETBALL for iPad is a game about throwing basketballs into basketball hoops. On the way to the hoop, your ball must bounce off of various objects or walls. You must sink each stage's basket challenge in one throw.

The speed of the balls' flight through the air is faster than four of the angriest birds taped together. The physics are deliciously swift. Yet the controls are not a keyboard-surrogate touch-slingshot. They put ANGRY BIRDS on Xbox 360, you know. You can play with just an analog stick. It's actually intuitive and highly playable. GASKETBALL would be neither intuitive nor playable at all with an analog stick: In order to shoot the ball, you trace a swooshing pattern, putting spin or speed or power or all three behind it simply with a flicking flourish of your finger. The range of expression contained in a single 0.15-second input is astounding.

Once you've beaten one "world" of stages—the game gives you limited balls; lose them all, and you start the set over—you unlock the ability to play a time-attack mode, and here's where my brain lights on fire. It's like ANGRY BIRDS, only you are angrybirding for your mortal life, at reckless speeds, applying full labyrinths of brainwork to every twitch of the finger.

This is what a touchscreen game is: a single simple joyful motion pregnant with consequence, birthed into a stage where things are happening. You must time your action to fit between, around, or through obstacles.

In short, touchscreen action games require incredible feel in simple actions on the tens-of-milliseconds scale, and superlative level design. Hey—that sounds like all other action games.

CALL OF ANGRY BIRDS GASKETBALL is as "real" a game as Call of DUTY (at the very least).

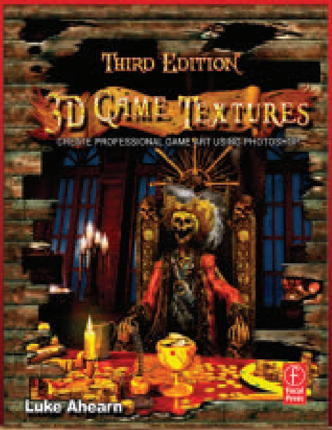
I first learned about GASKETBALL from Bennett "QWOP" Foddy, after showing him our tennis-like ball-and-paddle game TNNS (iOS/Android). In TNNS, aftertouch is everything; you tap the baseline to move your paddle, then slide or tap your finger along the baseline during the 0.xx-second freeze-frame impact after the ball hits the paddle to bend the ball's flight path. The bend is relative to the distance from the paddle (and ball) at which you released your finger, and the time it took to travel there. You can change the direction twice—tap twice on opposite sides of the baseline to do freaky right-angle bends. It's fun for multiple players (and it has a single-player mode, too!).

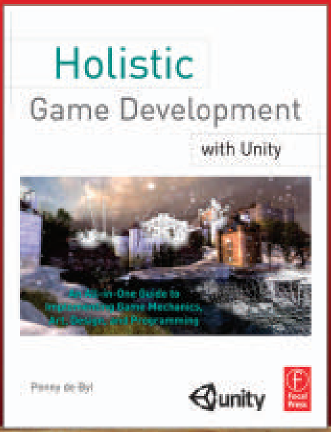
After showing him TNNS and getting his approval ("TNNS is great."-Bennett Foddy, Oxford University), I told him we had something more fun in mind for a golf-like game. I explained it, and then Foddy said I should try GASKETBALL. I saw one YouTube trailer, and almost chewed a row of teeth-holes in my bottom lip. Well, great minds think alike, et cetera. It just might be that some of the Great Touchscreen Games of the Future are here already, and more are arriving as we speak.

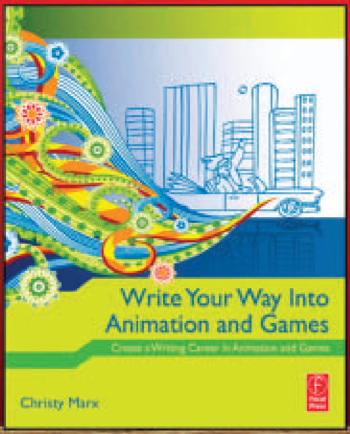
***

Tim Rogers is the founder of game development studio Action Button Entertainment. Follow him on Twitter via @number108.

Focal Press has new tools for your game design toolkit




Holistic
Game Development
with Unity
An All-in-One Guide to Implementing Game Mechanics, Art, Design, and Programming
Penny de Byl
unity



By Luke Ahearn
$59.95

Learn everything you need to create stunning, professional textures. Give your digital art depth and breadth.

By Penny de Byl
$44.95

An authoritative guide to creating games in Unity. Taking you through game design, programming, and art.

By Christy Marx
$39.95

Launch your career in writing for video games or animation. Let our award-winnin writers and game developers show you ho to generate ideas and create compelling storylines, concepts, and narratives.

Are you a game pro looking to make a few bucks while sharing your knowledge?
Learn more about freelance opportunities writing for our new Game focused Mastering-site.

Also visit focalpress.com/creativity.aspx, for a chance to win a Kindle Fire and a shelf's worth of books!

Focal Press

Creativity has an endless shelf life.
focalpress.com

POSTMORTEM_Jamie Cheng & Nels Anderson

mark of the ninja

pm 031

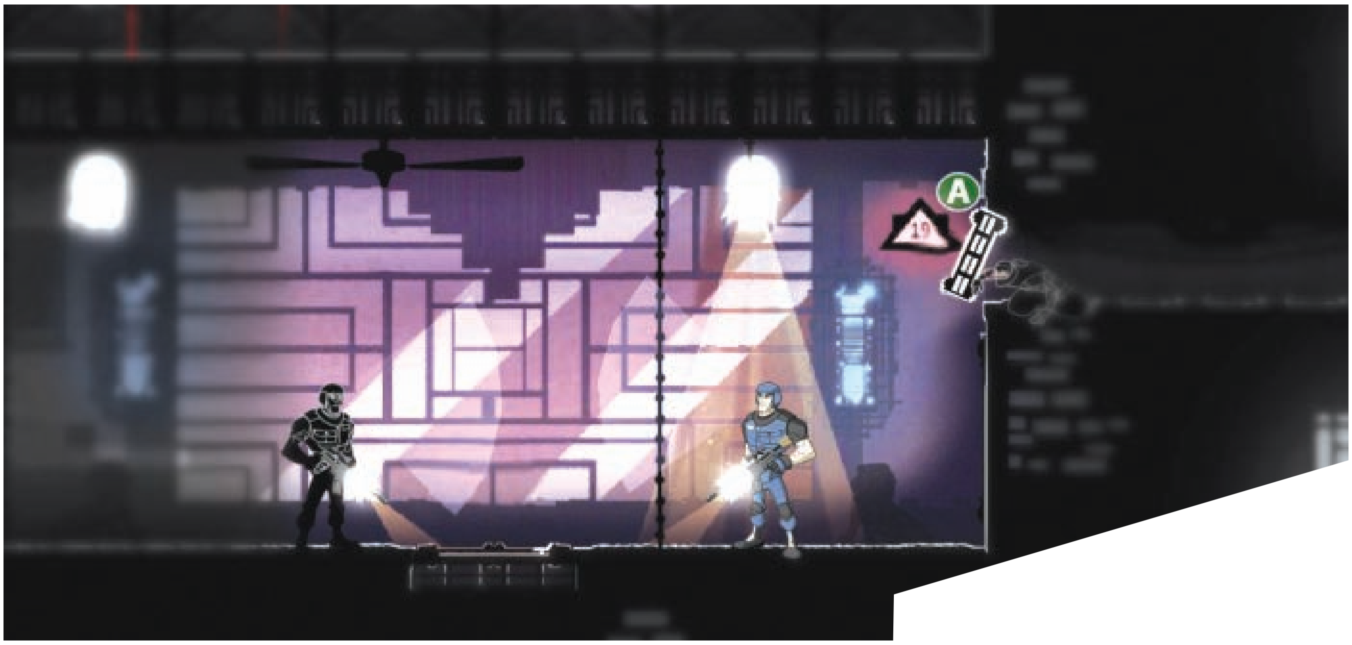
MORTEM_Jamie Cheng & Nels Anderson

MARK OF THE NINJA originated from a simple concept: We wanted to make a ninja game where the player actually acts and feels like one. The only way to do that properly was to make a stealth game. But stealth games have a reputation for being a notoriously difficult genre to work in, and we certainly discovered that said reputation was deserved. Furthermore, a 2D side-scrolling stealth game is something that had basically never been done before.

On just about every front—design, art, engineering, and sound—MARK OF THE NINJA pushed us beyond our experience. We had to find, often with no small effort, new solutions to the myriad challenges we faced. But by concentrating on the core experience we were seeking to deliver, and collaborating as a trusting, solution-focused team, we were able to produce a game that seems to have truly earned a place in the stealth canon.

MARK OF THE NINJA began as a two-minute concept video. Sixteen months later, we released the game on Xbox Live Arcade, with a Steam version following one month after that. In this postmortem, we will detail the largest things that made the game a success, as well as the aspects of the project that challenged us the most. ›››

POSTMORTEM_Jamie Cheng & Nels Anderson



WHAT WENT **RIGHT**

1. BUILDING ON SOLID TECH MARK OF THE NINJA had one huge design risk: We had no idea how stealth gameplay would actually work in 2D. Had we also started our technology from scratch, we could easily have spent over a year in pre-production. Thankfully, we had at our disposal the robust and proven pipeline of SHANK 2, which allowed us to very quickly begin prototyping our ideas (many of which failed). Having this mature pipeline ensured that the art team could work unfettered to create their amazing animation and background art without being bottlenecked by technology, or having to worry about fitting their work into memory.

In addition, we spent a large amount of time upfront designing our level-creation pipeline. In our previous projects (SHANK and SHANK 2), each level was meticulously hand-painted. We built those levels using the painful, classic waterfall process of designing the level upfront, then blocking it out, then painting the level, then hoping to never change the level again, because doing so meant a costly repainting of significant portions of artwork.

We knew that NINJA would live and die by being able to iterate on level design, and to facilitate that we designed a toolset that would enable designers to quickly change the level without huge knock-on effects to the rest of the team. We invested months into building a robust texture-tiling system for the environment art and better preview tools, because we knew that building good tools would have a multiplicative effect on our productivity.

2. PLAYTESTING EARLY AND OFTEN One of our key decisions on NINJA was to have early and constant playtesting. Playtests were held twice a week, with candidates combed from Craigslist. We needed to constantly and consistently test our design assumptions with fresh players.

Over the months, we fine-tuned the process of receiving and acting on feedback, focusing not on implementing player suggestions, but trying to get to the underlying motivations that our design caused. This was key to our success as a stealth game: When players complained that the combat was not satisfying, we would ask why they were motivated to fight. When players didn't understand our tutorial, we slowly tuned our controls and subtle cues, rather than simply flashing the instructions with even bigger text. This ran the gamut from changing the positions of light sources to emphasize the light/darkness visibility mechanic, to adding a pair of props the player had to strike simultaneously to progress, to ensure they understood multi-target aiming.

The playtests also helped us tune some of our core visual feedback, which became the solid foundation of our game. From designing subtle animation cues for what would happen when the player pressed a button, to producing the vast environment art that needed to look both dark and clearly readable at the same time, we needed to watch new players fumble through our game in order to see it from the perspective of someone who hadn't been playing NINJA every day.

Perhaps our only complaint is that we didn't start our playtests early enough. We did do some playtesting with other local devs, but the game felt too rough to get what we wanted out of public playtests until about eight months into development. For future projects, we're definitely going to figure out how to do playtesting even sooner, even if the builds are rather rough around the edges.

3. GREAT RELATIONSHIP WITH MICROSOFT It's hard to talk about publishers, because the bad news is immediately reported on, and the good news is largely dismissed as biased because the developer needs to preserve its business relationships. For our part, we've always worked incredibly hard to have a great, positive relationship with our partners, mostly because the alternative just isn't any fun. Who would want to spend over a year in an adversarial relationship?

Our approach was to once a week give our producer at Microsoft Studios (Torin Rettig) the information he needed in order to determine our major risks, and discuss them with him over Skype. This enabled both parties to turn what is traditionally a status update into a mutual problem-solving relationship, and indeed

Developer: Klei Entertainment
Publisher: Microsoft Studios
Number of developers: 7 for first half, ~16 for the second
Development time frame: 16 months
Release date: September 7, 2012 (XBLA); October 16, 2012 (Steam)
Development tools: Microsoft Visual Studio, Flash, Fmod, in-house tools including the absolutely vital "Shanker" (a.k.a. "Ninjer") level editor
Number of lines of dialogue: Approximately 8,000

more than a few times he was able to clear platform-specific roadblocks before we hit them. Microsoft was tremendously helpful, in everything from dealing with ratings boards worldwide to helping resolve certain TCR issues, which were all things a self-publishing indie would have to deal with totally on their own.

The hardest part of the relationship was when both parties were wondering whether this game could actually be made. Indeed, for about six months, both Klei and Microsoft asked the question "Can this actually be done?" However, once we were over that hump, and we decided that the answer was yes, this is something special (even though there are still at least a dozen things that we don't yet know how they will turn out), Microsoft put 100% trust in us.

Those who work with publishers probably understand that this is not exactly commonplace, and the mutual trust—that we could deliver and they would help—allowed us to focus on the task at hand. Crucially, the relationship turned what is usually a resource drain into a net positive, and I believe that is to the credit of both parties.

4. POLISH AND ITERATION TIME Our game was first slated to complete in mid-April, and we ended up in certification at the end of July, making the game about three or four months over our original schedule. Having lived through the hell of crunching to hit a date for SHANK, and missing some crucial polish time, we were adamant to not repeat our previous missteps and instead paid for the extension out of pocket.

The extra months allowed us to not only complete the game, but also to take a step back, and thoroughly playtest the complete experience from end to end. Being able to do this, and understand that at this stage in the process we didn't need to churn out assets but simply let the game simmer, afforded us the time and focus to make the small adjustments that took the game experience to the next level. We cut and edited cinematics, tweaked levels, adjusted controls, and even merged some items that we felt were too similar to each other.

The need for a holistic polish stage was a lesson we learned from our previous games, and we're super glad that we listened to ourselves, and took the time to continue to iterate, right up to the end.

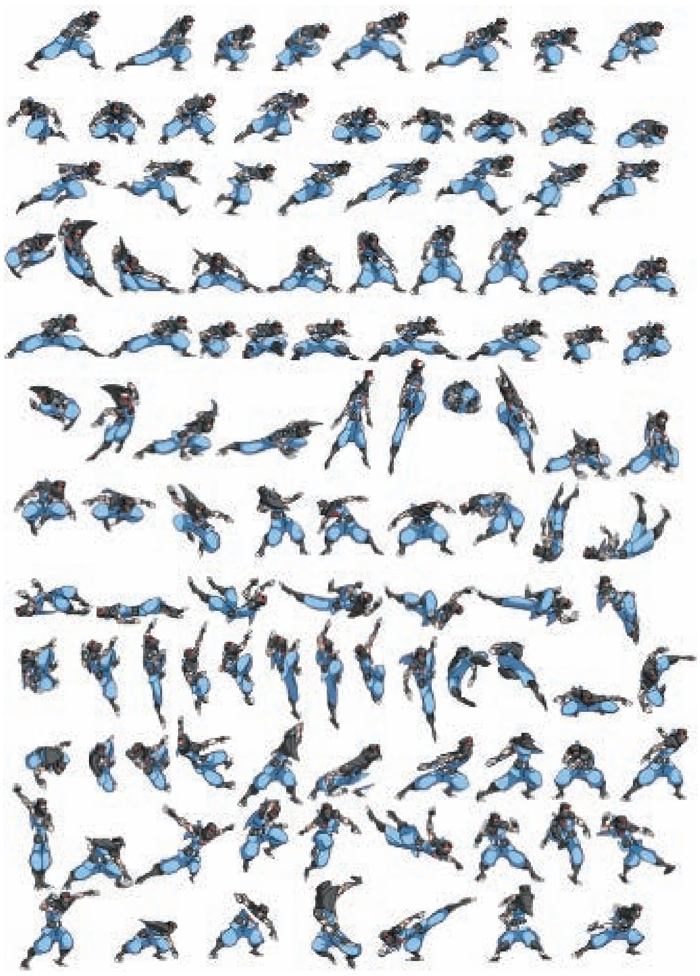
5. FOCUS ON CORE STEALTH PRINCIPLES, RATHER THAN GENRE CONVENTION Early on in the project, we decided that being stealthy had four core elements: Observe, Plan, Execute, React. Our design decisions continued to come back to this core, and allowed us to move away from simply copying genre tropes to creating new ways to move the genre forward.

As there are very few examples of 2D stealth games, and there were even fewer when we began NINJA, we had no templates to draw from. Instead, we ended up looking at the core experience that 3D stealth games provided, and really dug into how they are novel compared to other types of character-based action-adventure games. From this, we distilled what we believe makes stealth games interesting: player-centric systems, and intentional gameplay.

Stealth games thrive by placing the player in a world with a lot of interacting systems, and giving players the ability to approach situations as they see fit. They are fundamentally games about player choice. Encounters can be approached in a number of ways, all of which are valid. This was our gameplay target.

Once we had identified this, we were able to make design decisions that moved us closer to this goal, even if those decisions violated the conventions of the stealth genre. For example, most stealth games have being hidden as an analog state, where a light gem or some similar mechanism will convey a spectrum of concealment. In NINJA, stealth is totally binary. Either the player is hidden, or the player is illuminated, and that's it.

We made these kinds of design decisions throughout NINJA and it ultimately ended up creating an experience that feels like a stealth game, even though it differs in a lot of significant ways from 3D stealth games that came before it.



WHAT WENT **WRONG**

1. INITIAL LACK OF FOCUS Early on in development, we tried a lot of stuff. We had grand ideas of multistep elaborate cause-and-effect chains, where fire could propagate and enemies would react to every situation intelligently, and differently. We did this for a few months, and actually made significant progress in many of the systems.

But the problem was that none of it was any fun. About four months in, it was apparent that we still didn't have a clear idea of what the core game experience was going to be, and we needed to head into production if we were to complete the game on time. In the summer of 2011, the team regrouped, and we spent the next two months in an intensive deep dive to find the core. We did this by creating level after level to test what designs were actually creating interesting decisions for players, and soon determined the core mechanics that were essential to our experience.

As a result, we ended up wasting large amounts of art and animation as we drastically changed the direction of the game over a few short months. Although we eventually found the essence of the game, it caused many knock-on effects (further explained below), and we began to worry whether the game we envisioned could be made at all. We fully believe that the iterative process could have been significantly sped up and resulted in far less wasted effort.

2. STORY CAME LATE IN THE DEVELOPMENT CYCLE Our initial lack of focus, and the subsequent rush at the end of pre-production, had huge knock-on effects during production. It was only after the level iteration that we decided on shifting from light storytelling to a full-on story with plenty of cinematics, and we had not planned for the resources we needed to animate that story.

Worse, at this point we still had no script, and even once it was written it was still subject to change based on gameplay. So we scrambled to hire contractors and organize the script, storyboards, and scene—all of which became a daunting task. It's a huge testament to the art team that we were able to create our highest-quality animation in such a short timespan.

We're mostly happy with the story delivered (with one caveat, **see 5**), but there were definitely times where we would have liked to change something but knew we simply didn't have the flexibility in the schedule anymore to do so.

3. NAILING DOWN CERTAIN MECHANICS LATE Since we figured out NINJA'S core concept rather late in development, the advanced mechanics had to come even later still. We had written down what we felt the late-game mechanics would be, but we had no idea if they would work, and indeed many of them turned out to be duds.

For example, the last major ability the player receives is a short-range teleport. There were two other mechanics that we tried: an air-dash and a time-stop. These both required a significant investment in programming, art, design, and testing, and were testable only a few months before we shipped. Once we tested those abilities, we discovered they just weren't fun within the context of our game. Scrapping them after so much investment so late in the process was difficult, but we had to put the quality of the game first. Many games fall apart in their final act (probably for similar reasons) and we didn't want to make the same mistake.

The teleport ability that we shipped in NINJA was only implemented two or three months before we went into certification, and its implementation required designers to rework the level design to utilize it effectively.

4. DESIGN CRUNCH Largely due to the issues mentioned above, the design team ended up having to put in some major effort late in the project. Although we nailed the level design, many of the later design elements were still in the nascent stages of development. And even though we invested plenty of time in developing our tools, they were not initially meant for the large, sprawling levels that we were now designing, and the giant levels ended up taking minutes for an incremental change or up to an hour for full exports, causing considerable slowdown in design.

Beyond this, NINJA is a game that lives or dies by its polish and precision. Moving a single light a couple tiles in one direction can transform an encounter from dull to interesting. An encounter having four guards when space only really supports three can take the situation from challenging and interesting to basically impossible. But having never made a game like this before, a lot of these lessons had to be learned in the trenches, trying many different things until they gelled.

The design team had the daunting task of creating levels while some mechanics were still in flux and the story was rushing in. Although every department felt the effect of pre-production sneaking into production, it was the design team that took the brunt of it, and the last six months of development were especially hard.

5. STORY PLAYED TOO STRAIGHT Even though we determined that MARK OF THE NINJA would have a more traditional story structure, we still wanted the narrative to be ambitious and unique. However, we also felt that we didn't want to overwhelm players with a complicated, unfamiliar narrative right from the outset, especially considering the game has a lot of mechanical and systemic novelty players must also understand. So we opted to play the story very straight at the outset and let the complexity and nuance come in about halfway through the game.

What we didn't realize was that some people would see the seemingly simple setup—a rote revenge story—and assume that's all there was. By the time the complicated and nuanced aspect of the story began to unfold, we' had already lost a portion of the audience plot-wise and couldn't get them back.

Now we don't think this was a complete failure, as we have heard from a number of people that they were very pleasantly surprised by the narrative in NINJA and how it turned out to be far more than they expected. The ending especially was praised by a number of players and critics alike. But this certainly wasn't the proportion we intended, and it became something we will certainly be mindful of in any future projects that have a narrative component.

***

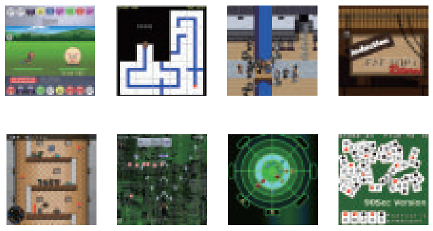
GO NINJA GO MARK OF THE NINJA was, without question, Klei's most ambitious project to date. It was also the game that proved, albeit not without some growing pains, that Klei can successfully work on more than one project at a time. NINJA provided an opportunity for a great deal of the team to grow and gain valuable experience, as the team was a combination of SHANK veterans and developers new to the studio.

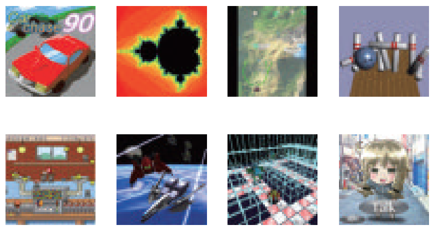
Our guiding principle is that we believe we can walk the walk: We can create a great game without sacrificing our lives and borrowing from the future, and we work together to find solutions to the challenges posed by creating new and different games. Despite, or possibly because of the challenges, the game is one we are all tremendously proud of, and it has been extremely well-received. A number of people we trust commented that MARK OF THE NINJA is Klei's breakout game, and given what we learned creating it, we certainly intend to do everything we can to make that true.

***

Jamie Cheng is the founder of Klei Entertainment, and Nels Anderson is MARK OF THE NINJA'S lead designer. Follow them on Twitter via @biiigfoot and @nelsormensch, respectively.

OPEN SOURCE / HTML5 / WEBGL / PHYSICS SIMULATION





enchant.js is a time-saving, and completely open source engine for creating games in HTML5. With the number of both official and user-created plugins ever increasing, enchant.js currently supports WebGL, drawing in both canvas and DOM, 2D and 3D physics, event-driven processing, and more.
Find out just how easy HTML5 game authoring can be at

enchantjs.com

WHAT *MAGIC: THE GATHERING* CAN TEACH GAME DEVS

COLLECTIBLE CARD GAMES ARE BLOWING UP ON MOBILE DEVICES. IT'S TIME TO EXAMINE THE ORIGINAL.

Editor's note: This article is an excerpt from an article originally posted on Gamasutra. You can find the full text at www.gamasutra.com/view/feature/183623/what_Magic_the_gathering_can_.php.

Magic: The Gathering has undergone a revival lately. The game's current card set, Return to Ravnica, is widely regarded as one of the strongest in its 19-year history, with retailers running low on supplies *worldwide.*

MTG alone invented and defined the (collectible card game), and its revival—coinciding with developers' race to build gacha-fusion card battlers in the mold of GREE and DeNA's Japanese hits such as RAGE OF BAHAMUT and DORILAND—has made it Hasbro's top IP, as well as the most popular CCG in the United States.

Magic's core concepts are pretty simple: Use land cards to generate mana, use mana to cast spells and summon creatures, then use those to attack and defeat the other player. The complexity, however, comes from the emergent strategy generated by both these base rules and the over 10 thousand unique cards that could potentially make up a deck today.

Magic is a treasure trove of learning, and as a relapsed *MTG* addict and a designer, I'm going to share with you the top five things we can all gain from its success.

LESSON 1: EMERGENT STRATEGY *Chess* is a classic of game design due to its emergent strategy. The base rules of the game are relatively simple and uninspiring by today's standards, yet the complexity that arises from the movement and counter-movement between two players is beyond what could be mastered in a lifetime.

The human mind can't comprehend the complexity of cause and effect in *chess*, so it goes about seeking patterns in order to model and understand it. When the mind uses these models to apply a strategy that generates a win condition, it provides a sense of satisfaction and exhilaration as a reward.

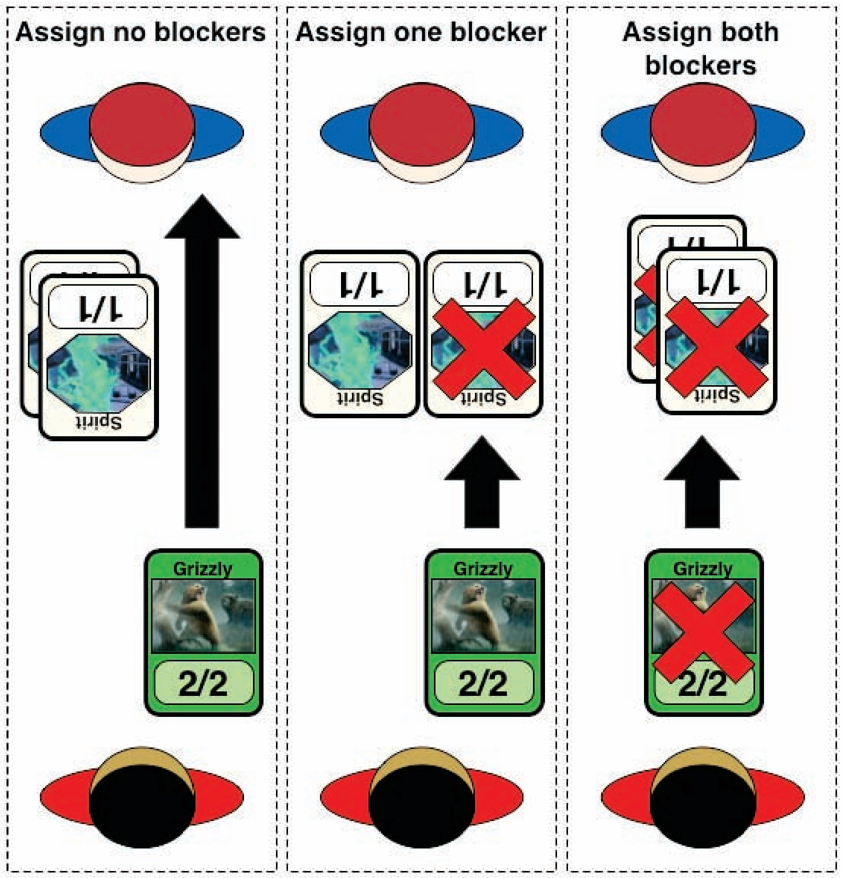
Designing for the sort of emergent strategy found in *chess* is elusive, if not impossible. You are far are more likely instead to discover it in an early form and then build upon it, as is the case with *Magic*. Indeed, *chess* itself has evolved to its current form over 1,500 years.

In *Magic*, players control creatures that have two stats: "power" and "toughness." When in play, their controllers may assign them to attack and, in response, defend against attacks. This simple rule provides a good deal of *MTG's* core strategy.

For example: It is player A's turn and they have a Grizzly Bears creature on the battlefield, while player B has control of two Spirit creatures.

Grizzly Bears has a power and a toughness both rated at 2 (depicted as 2/2), meaning it will deal 2 damage to a player or any defending creatures, yet will be killed when 2 damage is inflicted upon it. Meanwhile Spirits have power and toughness each of 1 (1/1).

Player A declares Grizzly Bears to attack player B. In response, player B three options: Do nothing and take the damage from the bear, assign one of the Spirits to defend, or assign both Spirits to defend. Below is a matrix of outcomes in each scenario:



A player's decision in this situation is likely affected by multiple other factors, including the other cards in play, their hand, their deck, and creature abilities. For example: The Spirits have the ability Flying, so Grizzly Bears (which do not have Flying) cannot block them, meaning they can attack unchecked for 2 damage next turn.

Also in consideration are the remaining mana and cards in each player's hands, due to the potential to play "tricks." For example: Player A has declared attack with Grizzly Bears and has in hand, unbeknownst to Player B, Giant Growth.

Giant Growth is an instant card that can be played after attackers and blockers are declared, bolstering a target creature's power and toughness by 3. With Giant Growth applied to Grizzly Bears, it can do 5 damage and dies after taking 5 damage. Below is a matrix of outcomes for this new scenario:

Player B, however, may have a Cancel card, which would counter Giant Growth and so be played accordingly. Possibly, both players expected to come up against each other's abilities, and built their decks around them with many spells or counter spells.

This second-guessing of a player's actions and card selection is known as the metagame—a big part of all tournament play. The range of abilities attached to creatures, spells, and lands gives any player thousands of options in any game. Each set of cards, of which there are four per year, usually provides one or more new ability to the game, which creates a constantly shifting landscape for players.

As in *chess*, building mental models and applying them for success triggers the brain to provide a sense of satisfaction. However, unlike *chess*, *MTG*'s emergent strategy is somewhat forced by the printing of these sets—building out a player's options and leaving it to the community to collectively determine the best ones.

Players then build, play, and refine decks over the months as sets are released. The strongest prevail, with supply-and-demand economics making many rare cards (known as "chase rares") valuable. When another player builds a stronger deck or a combination of cards that defeats the strongest decks, the economics shift.

Magic teaches us to design a game that is basic at its core but gives players a multitude of meaningful options in play, even if that is somewhat forced. This provides a game that has a learning curve—one that will keep players striving as they discover and apply strategy.

LESSON 2: COMPULSION THROUGH VARIABLE REINFORCEMENT Imagine it is your birthday, and all your friends each bring you a wrapped present—yet you can't open them until you call a coin toss right twice in a row. If you call one wrong, you get another go, and keep going until you win.

Now imagine the same proposition, but instead of presents being wrapped, they are open for you to see—things like socks, books, and DVDs; some things you want, some things you don't. Which is the most compelling?

In the second example, the coin toss feels like a chore, whereas knowing you're going to get something but not knowing what it is makes the coin toss more (if not very) exciting. This is called variable reinforcement, and leads to repetition of an action much more consistently than a fixed equivalent.

In *MTG*, each player draws a card from his or her deck each turn. It is possible whole games could be won or lost on a single draw. This gives playing the game an addicting quality in the short term, which marries with the strategy of deck-building, plus the game's goals (see **Lesson 3**) in the long term. It also provides the game with the somewhat-affectionate nickname "Cardboard Crack."

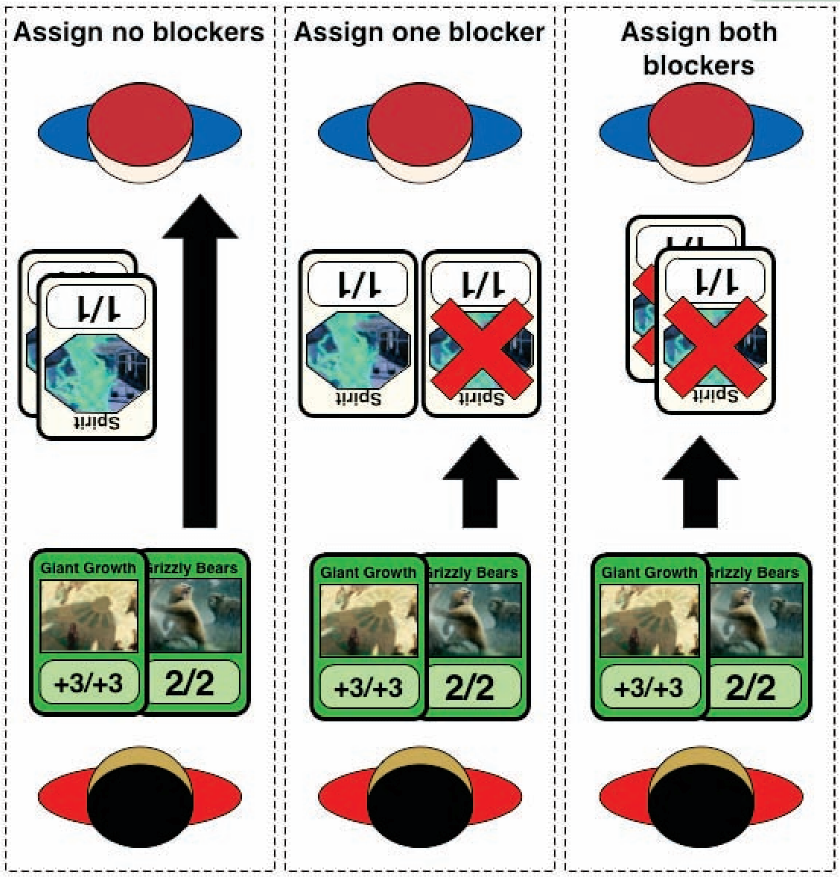
This same theory can also be applied to the addictive nature of sealed packs of 15 semirandom cards known as booster packs. Boosters feature a set ratio between common, uncommon, and rare (or mythical rare) cards, plus a land and tip card or token. With each pack you know how many cards you are getting when you buy it, but you don't know which cards you'll get.

The scarcity of each card is actually printed on it, and the ratio of cards in boosters breaks down as follows:

RARITY	MAKEUP
COMMON	71.4%
UNCOMMON	21.4%
RARE	6.3%
MYTHICAL RARE	0.9%

With 15 mythical rare cards in Return to Ravnica (the latest set), if you wanted a Jace, Architect of Thought—one of the strongest mythical rare cards in the game—you have a 0.06 percent (or six in 10,000) chance on a single card, one in 120 per booster, or less than one in three per booster box (a pack containing 36 booster packs).

These numbers, anecdotally, stack up very similarly to many gacha-fusion card battlers. Each purchase has the potential to deliver an "epic pull"—a card that is so



powerful it can stack games in favor of the player—yet the likelihood is against it happening, encouraging players to repeat the action.

Magic even has it own tournament format that utilizes the compulsion of opening boosters. In booster drafts, players bring three booster packs. Each player opens one at a time, taking a card and passing it on to the next player, who also takes a card and passes it on. This continues until the pack is depleted; then, the next one is opened, and so on. When all boosters are depleted, each player has a stack of cards with which to build a deck that is used in a tournament.

MTG is possibly one of the best examples of using variable reinforcement in both play and at retail. The probabilities of rarity for each purchase and the thrill of the draw each turn make it an incredibly addicting experience.

LESSON 3: RETENTION THROUGH GOALS

Variable reinforcement is not the only way *Magic* inspires lifelong dedication and spending from a great many players. It achieves these aims via a series of explicit and derived goals that satisfy and retain a number of different player types. These mechanics can be applied to offering possible appeasement through play to the four Bartle types: Explorers, Killers, Achievers, and Socializers.

Explorers: World, Strategy, and Theory Explorers like discovering and mapping worlds. While the cards of *Magic* tie in with a narrative based around the "Multiverse," and there are novels and other fiction available, the majority of Explorers in *MTG* enjoy organizing and sharing their discoveries of the game itself.

The web community on Wizards of the Coast's own site is host to a great deal of strategy and theory, with videos and articles on deck construction and gameplay techniques. The cards already provide Explorers with a lifetime of possible combinations and categorization, and the constant release of new sets expands this indefinitely.

Killers: Tournaments Killers like the buzz of triumph over an adversary. *MTG* as a zero-sum game provides this thrill over the kitchen table, yet the popularity of casual and official tournaments put on by DCI, *Magic*'s tournament-regulating body, provides much more opportunity for the aggressive Killer.

Wizards boosts the popularity of tournament play by offering cash prizes and prestige to winners of the official Pro Tours, who become celebrities of the game. This legitimizes the long-term goal of becoming a Pro Tour champion for any dedicated player, driving them to be continually loyal to the game.

Achievers: Planeswalker Points and Collecting Achievers like clear indications of their progress. Planeswalker Points are a system for players of DCI-sanctioned tournaments provided by Wizards of the Coast. The points tick up for doing ancillary things like joining guilds, but primarily come from playing in tournaments. DCI maintains league tables of local players, which encourages them to continuously engage in competitive play, stay in the *Magic* fold, and improve their decks through the purchasing of new cards.

Additionally, each set is accompanied by a player guide, which lists each card in print on a checklist. This targets a subsection of *MTG* players who perhaps aren't even players at all: collectors.

Collectors stay focused on the self-appointed goal of completeness, and the checklist is a measure of their progress. The human mind instinctively focuses on a scarce resource or deficiency (often money or companionship, but sometimes the blank tickbox), and then formulates ways to rectify this deficit.

Collecting is a strong goal set in lots of games, from alternate costume unlocks to gamifaction badges. It works either with finite (cards) or infinite (Planeswalker Points) resources.

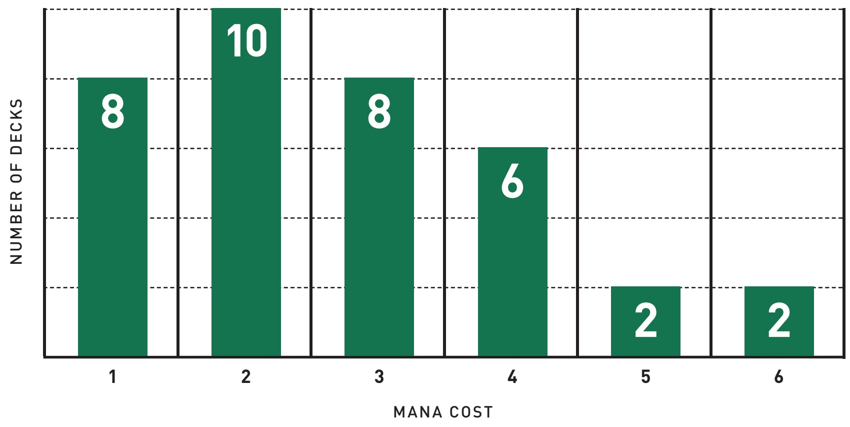
Socializers: Building a Network of Friends and Teams Socializers like interacting with people. While *Magic* dictates that the players need at least one other person to be able to play, committed players will seek out a large roster of potential opponents.

The community around *Magic* is absurdly strong at local, national, and international levels. Wizards of the Coast's activity starts through local organized play sessions like Friday Night *Magic*, which are run in conjunction with local retailers (often comic shops) and also through the Internet. There are other sites that unsurprisingly offer forums, chat, and articles, as with many other games, but as developer-manufacturer, Wizards of the Coast provides an unusually large amount of community reinforcement.

Furthermore, competition-level players (Killers) rely on teams of people filling various roles, including collectors (Achievers) and deck builders (Explorers), for their success. This further increases the social element, and fosters a community around a game, making it a hub for a player's life.

Players with multiple friends in a game are more likely to stick with it for longer than those who have no social connection, making

MANA CURVE



it an important decision for marketers and designers. If your game can connect and bring people together to have fun, it is likely to have a loyal and active fan base.

LESSON 4: BALANCING AND RESOURCE CONTROL Balancing and the control of limited resources are important skills for designers, with small tweaks having massive implications for F2P economies. In *Magic,* building decks teaches players these skills through the rapid iteration of moving cards into and out of the deck.

Mana, of which there are five types, is produced by color-specific lands at the rate of one per turn once in play. To put a land out the player may play it from their hand, again at a rate of once per turn. Spell cards, however, may be played from the hand without limit, but only if a player can provide their mana cost.

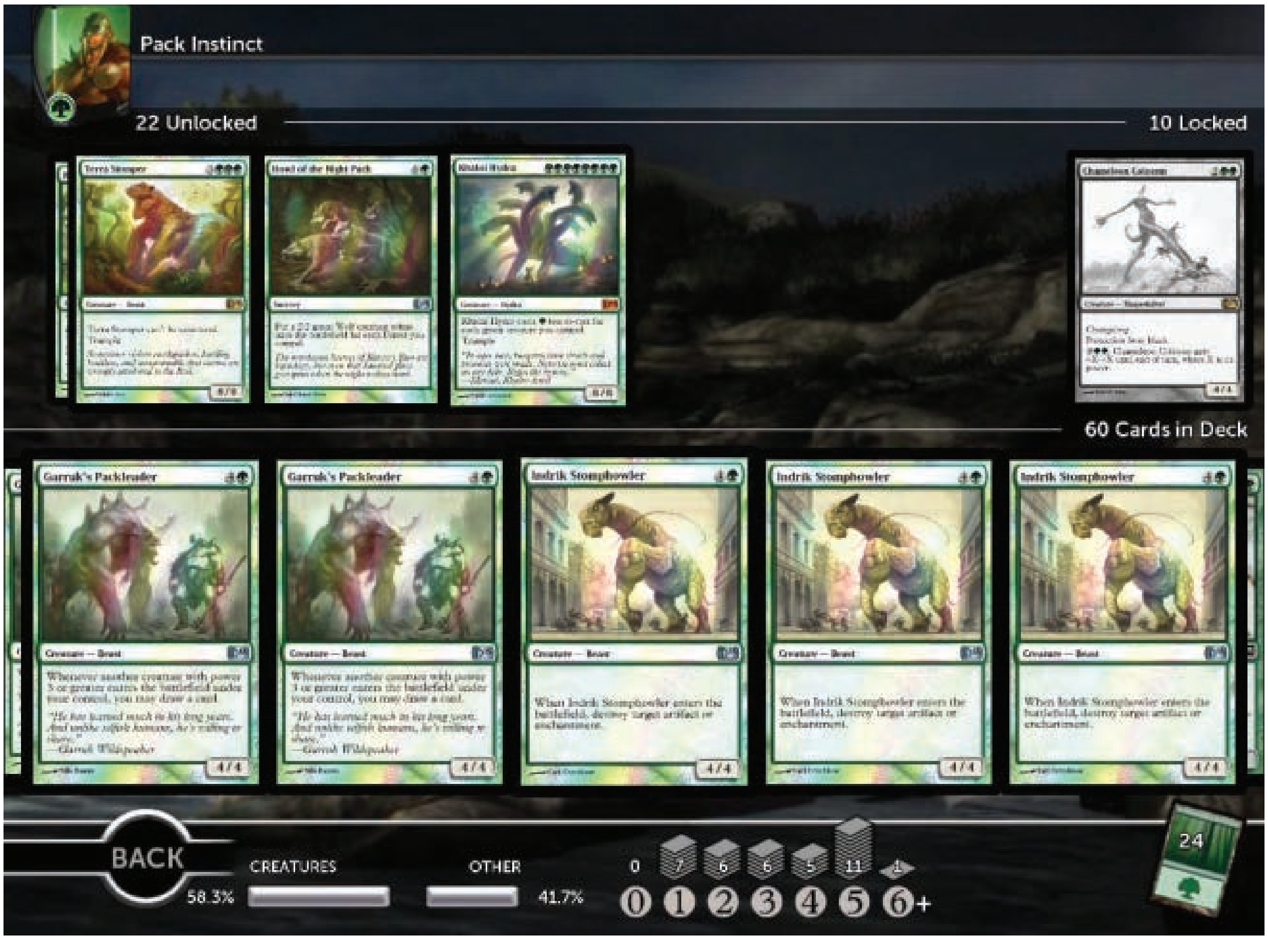
These restrictions, along with a starting hand of seven and drawing from the deck once per turn, form the basis of *Magic*'s balancing mechanics.

While a deck must consist of at least 60 cards, it may have as many or as few lands as the player chooses, which results in a surprising amount of strategy and consideration.

Too few lands and the player will have great hands full of spells but no land with which to bring them into play (this is known as being "mana screwed"), while too many lands means the player is able to play lots of spells but has less of a chance of drawing them (this is known as being "mana flooded").

This is further compounded by the effectiveness (or power) of a card being designed to correlate with its mana cost (or how many lands must be used to cast the spell). Low-cost cards can be played quickly in the early stages before the opponent is in a position to defend, while high-cost cards can be played later in the game to dramatic effect.

If the player plays only low-mana-cost cards in their deck, they will be in a position to play multiple cards in the early



stages of the game, but the single draw per turn will throttle their progress as they run out of cards in hand, picking up only low-cost, low-effectiveness cards that have little impact on the game's outcome.

Therefore players must consider their mana curve (how many of each card they have at each mana cost) for building an effective deck. If the deck has enough of the most effective cards at each mana cost for the first five turns, then it will generally play well, with powerful options always available.

Many sites and smartphone apps exist to help the deck builder analyze their mana base and curve, but balancing *Magic*'s complex systems with only statistical analysis is near-impossible. Small, overlooked elements and subtleties can have massive implications as they begin to interact.

As such, *Magic* players have devised a method known as "goldfishing," in which they play against an imaginary opponent who does not respond (e.g., a goldfish), counting how many turns it takes to win the game. From this, and understanding the underlying theories of mana screw, mana flooding, and mana ramps, players will set about refining by adding and removing cards, adjusting their land-spell ratio and mana ramp for the most effective play.

Goldfishing is very close to the analytics we know in modern video games: the recording of how a system, or more specifically a system under human control, performs. Physically swapping cards in and out takes seconds and drastically affects how a game plays out. This can help a designer plan around the cause and effect in a system too complex to comprehend.

Magic has a second lesson for us here: pinch points. A pinch point occurs when a resource is so scarce that a change in availability has huge knock-on effects to its market price. Chase rares, as described earlier, fuel the economy around *Magic*. These cards become valuable because of the supply being artificially low, while the demand grows thanks to their success in tournament-winning decks.

As booster packs are the only source of these cards, players and resellers open them in the hundreds or thousands, creating huge demand for the packs.

Limited resources define economies; if you are too generous with them, then making money is difficult in F2P. But scarcity's desire-generating effect isn't only applied to IAP economies—think equipment in an MMO and points in a shmup. Both act as major driving factors for return play.

LESSON 5: PRESENTATION You may have heard the adage that any good game is still fun with text and box graphics. That's true, but presentation is a key function of delivering an experience and can instill a sense of quality and value in players' minds; all great games look great.

Humans are drawn to other human forms, especially eyes. Wizards of the Coast uses key cards, specifically Planeswalkers, to showcase human characters that provide players with an emotional reaction. They become emblems across products, both physical and digital.

However, not just Planeswalkers but all cards have strong character imagery. For example, a modern printing of Mind Rot features a striking image of a character praying, with the top of his skull collapsed, while Switcheroo depicts a dragon facing off to a turtle. All cards tell a story of their function.

Mind Rot
2
Sorcery
Target player discards two cards.
With no memories left to compare to his present situation, Andalum was finally content.

Magic card artwork has its own fan base, and many collectors take more interest in the art than the game itself, with original artwork exchanging hands for vast sums.

Moreover, *Magic* is an information-rich game, and the border color and character artwork make cards easily distinguishable from each other, creating a mental association between the physical object and its function. Watching high-level players play is an incredible experience, especially considering the sheer number of cards they play against without checking card text.

A flash of the Mind Rot artwork informs an opponent that they must discard two cards, while the functional placement of a card's mana cost, type, and text are laid out to make interpreting them easy for less-familiar players.

Furthermore, supporting artwork of card borders and packaging drives the brand of *Magic* as a whole as well as the characteristics of each set. Clearly Wizards is carefully creating a sense of quality that pervades almost all of the modern *Magic* products: Shiny boosters, foil cards, and sturdy boxes make opening a product an exciting event over and above the random chance of rare cards.

The game would not enjoy its success today had the cards been crudely drawn and printed on flimsy paper. The world it establishes and the quality it presents, as with any game product, comes from the physical and the visual.

In video games we've long been good at visual fidelity, but now we need to pay more mind to how we create a sense of quality that makes making a purchase and playing easy and exciting for the player—especially when designing free-to-play games.

It is common for characters to be uninteresting or for in-app purchases and menus to be drab because designers focus instead on the quality of game mechanics, yet presentation can create a sense of quality, while character artwork conveys stories. Both increase players' emotional response to the game, which increases their likelihood to continue playing and spending.

GATHERING THE LESSON The success of Japanese gacha-fusion card battlers derives very clearly from the mechanics of *Magic* as the father of all CCGs. Yet the game can teach all of us lessons beyond the function and design of these games.

Playing it can give young designers the theory to create emergent strategy and the ability to solve problems and balance resources to create a fun, optimized experience that ramps correctly.

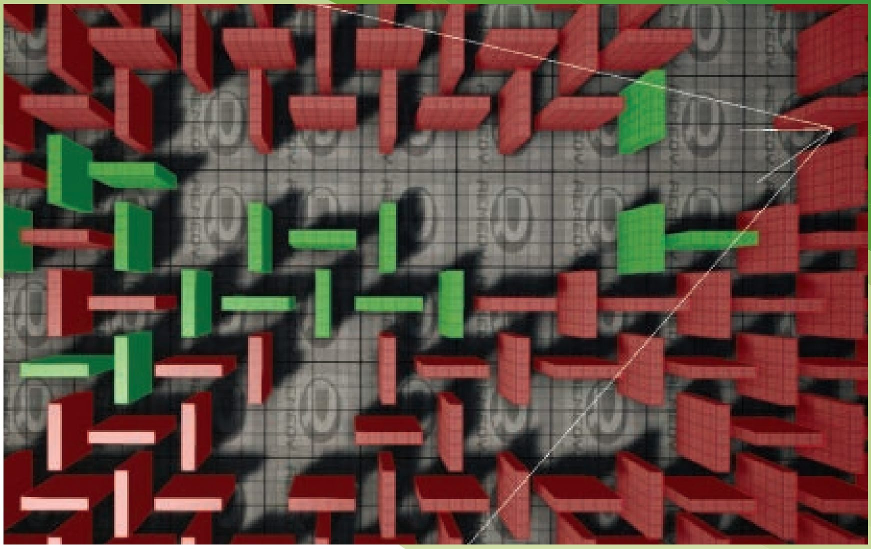
It also gives us clues for building variable reinforcement through random chance in both play and purchase, while using community, collections, and tournaments to satisfy Bartle types and keep players in the long term.

Finally, all of these can be bolstered by providing a sense of quality via the visual and physical representation of a product to create value in the player's mind.

Magic: The Gathering is a unique game that sits alongside *Dungeons & Dragons* and *Fighting Fantasy* in its impact to the world of games and potential lessons for players. I highly recommend that you go out and buy a couple of decks (any of the Duel Decks are great starting points) and set about understanding it—it will be the most enjoyable design class you'll ever take.

***

Will Luton is a consultant in design, retention, marketing and monetisation for mobile and free-to-play videogames. Contact him at @will_luton on twitter or go to will-luton.co.uk.



Top view of the test scene. Occluded objects are drawn red, visible ones green. The white triangle marks edges of camera frustum.

UMBRA 3.2s

The best middleware companies choose a narrow focus and master it on a level that would require massive amounts of work if developers were to attempt to solve the problem themselves. Umbra Software's developers chose the classic problem of determining object visibility (occlusion culling) and have been successfully cracking it since 2006.

While the previous versions of Umbra focused on GPU-accelerated visibility queries, Umbra 3 (3.2s being the latest release) has been revamped with support for a new CPU-based algorithm that can take advantage of occluder data that is generated automatically from the scene geometry. This is a welcome improvement, as the latency introduced by GPU queries has been a major problem for many applications. Umbra 3 still operates only on the scene geometry without requiring any artist annotation, which means content doesn't have to be modified manually in any way. Occluder data is automatically generated by Umbra.

Umbra 3.2s

Umbrasoftware.com
Usable for Wii U, PlayStation 3, Xbox 360, PC, iOS, and next-generation consoles

SYSTEM REQUIREMENTS

Available on request

PROS

1] Reduces draw calls for primary and shadow-caster passes
2] Automatic; no additional content work is needed
3] Clean API and robust runtime

CONS

1] Requires pipeline modifications
2] Performance with very large view distances needs improvements
3] Occluder simplification isn't optimal strategy for extremely dynamic scenes

UMBRA 3 BASICS In case you're reading this review and you're new to Umbra, I'll explain a few of Umbra 3's fundamental concepts, which I'll refer to throughout the article.

A **Tile** is an axis-aligned block of a scene that can be loaded and saved individually. Tiles can be split into multiple caching regions for preprocessing, so you can avoid recomputing large areas when you're only making local modifications to a scene.

A **Tome** is a read-only data block that Umbra uses to perform visibility queries. The Tome is created by combining a set of Tiles.

The **Optimizer** is a library for generating the occluder data in the Tiles (and, optionally, the cached combinations of the Tiles). The Optimizer is typically integrated into the game-world editor or content pipeline.

The **Runtime** is the part of Umbra 3 that performs the occlusion queries using the Tome, and it's integrated into the rendering application.

Lastly, the **Umbra Viewer** is a tool that ships with the Umbra SDK. The Viewer is used to verify that the Umbra content pipeline integration works properly by allowing the user to examine the geometry as it is passed to Umbra. The Viewer lets you quickly check to see if the culling is working properly without forcing you to go through a full integration first. It's also a powerful debugging tool, with a plethora of visualizations, statistics, and other useful information.

PIPELINE AND CACHING Portal-based occlusion culling typically requires the scene geometry to be processed offline. Fortunately, Umbra's developers have done a good job in providing a set of simple, powerful tools that produce quality results without much hassle.

The basic idea is that you feed all the static geometry of your scene into the Umbra Optimizer. The Optimizer runs geometry voxelization in the Tiles, which are also used as a processing cache. If there are any modifications to the scene geometry, you only need to recompute the areas

GUILD WARS 2

affected by the changes. When you're moving objects around or modifying the scene geometry, the Tiles will have to be recomputed, but if your change only affects a small chunk of the scene, the update process should only take a couple of seconds and can be easily done in the background. For us, this means our editor feeds modifications to Umbra Optimizer, which, after a short while, returns the processed data without the end-user ever noticing that something happened.

In the event of making a larger change, you can expect the full recomputation of the scene to take a few minutes on an average desktop setup. When we were working with large levels, we found that it made development much smoother if we had the team working on a distributed base set of cached data.

It's worth noting that the Optimizer API is very flexible in terms of splitting work and choosing how to store cached data, so it was fairly easy for us to make a few small, local changes to the content pipeline.

OCCLUDER SIMPLIFICATION The length of time it takes to finish the occluder generation process per scene depends mostly on the settings you feed to the Umbra Optimizer. Setting the size of the smallest recognized hole in the geometry controls the resolution of voxelization. By setting the smallest occluder size, you control the resolution of the portal graph, which in turn dictates not only the final data size, but also the level of detail in the occlusion and the occlusion query times.

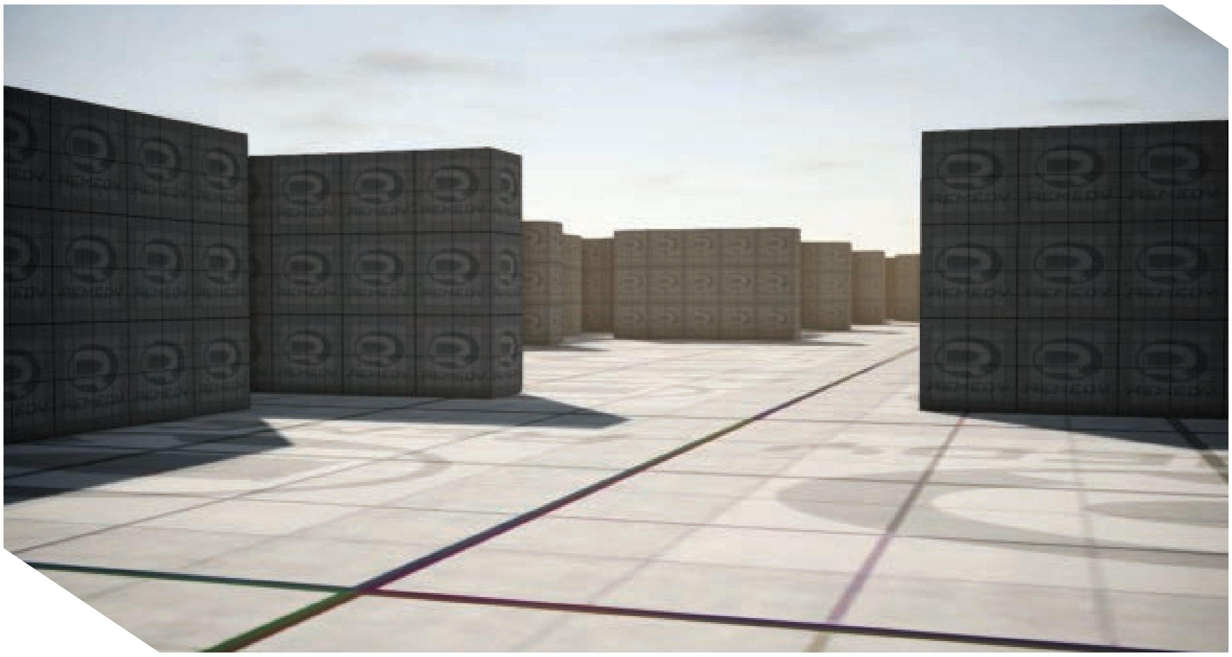
In order to find your optimal parameters, you'll need to experiment. Fortunately, the options are fairly intuitive for the user; you can easily tweak the parameters to optimize for situations where you need accurate occlusion information, or situations where you need to prioritize memory consumption. In our testing, we found that using an occluder size of many meters works nicely for large outdoor levels and still requires reasonable amounts of storage, but for indoor scenes, you generally want to capture an occluder size of less than a meter in order to make sure all the doors and windows are in their correct positions and to ensure that there is accurate occlusion information between rooms. Visibility results are always conservative, so having suboptimal parameters will result in bad occlusion, but will not create rendering artifacts.

Umbra 3 also provides the possibility to override parameters for specific areas if your game contains both indoor and outdoor environments. Unfortunately, Umbra 3 doesn't have any tools that can automatically determine a good baseline set of parameters for the area you're working on, which would have been nice; while we wouldn't expect it to cover all cases, having a reasonable base set with optional overrides would make it easier to use, especially before you get familiar with tweaking things.

STREAMING YOUR WORLD In modern games, you rarely want to have a single static world. For our project, we needed the ability to build a large base world, and then efficiently create local variations on top of that. This is where Umbra 3's Tile model came in handy; we can have multiple versions of a Tile for any given location, so we only have to load the ones needed at runtime. Note that combining an arbitrary set of neighboring Tiles does add some computation overhead to loading, but you can of course cache all permutations if necessary.

The Tiles that the Optimizer generates are then combined into a Tome, which is then used for occlusion queries. For our typical scenes, building a Tome from a couple thousand Tiles (with a final size of approximately a megabyte) takes less than a second without caching. For us, the typical case is to unload Tiles from one far end of the world, and then load them in from another. In such cases, a simple runtime cache is enough to provide good speedup without any preprocessing. You can handle all these computations in a background thread while using the previous Tome, and you can have multiple Tomes loaded at any given time and use them as you like. Implementing content streaming on top of this means making some modifications to the engine, and might cause delay for getting updated occlusion

A test scene rendered from the main camera.



results, but if you handle these computations in the background, you can keep your framerate smooth.

RUNTIME AND DYNAMIC OBJECTS The Umbra 3 runtime provides separate occlusion culling techniques for static and dynamic objects. The traversal of the generated portal graph while determining the visibility of static objects can be used to rasterize a conservative depth buffer, against which dynamic objects can quickly be tested for visibility. Both operations are performed purely on the CPU. These techniques cover most use cases, so it should be possible to find a good solution for most of the applications. The pre-processed approach is mainly useful for static data, but it can efficiently handle extremely complex setups as well.

Umbra 3's memory model lets you perform all visibility queries in multiple threads, making it easy to split work from render loop on multicore machines. Umbra 3 can also be used for culling shadow casters, with separate functionality for directional and sphere lights. The typical cost of camera queries in our outdoor scenes is around a couple of milliseconds, though if you use smaller viewing distances and less traversal in Portal hierarchy, you can cut the query time down further.

Overall, we hope to see Umbra 3's performance improve in this regard, particularly when it comes to working with large outdoor scenes, and we'd like to see visibility query times become less scene-dependent. Optimally, visibility queries should run roughly in constant time regardless of the parameters fed to Optimizer, making run-time performance less dependent on type of content. The performance impact of looking out from a window is high if your Optimizer parameters are not tuned to match density of detail in the scene.

All of Umbra 3's shadow-query methods support additional optimizations using the set of objects that are visible to the main camera. Because CPU-based algorithms do not introduce the latency seen with GPU queries, it is easy to fit the visibility queries for all the shadow-casting lights into the rendering engine. Achieved reduction of shadow casters with large cascaded directional lights is very good, and makes the content production workflow much more intuitive. Ever had a problem where the player is looking at a wall and there are hundreds of objects drawn into a shadow map? It shouldn't happen if you only draw casters that are shadowing visible geometry (in this case, the wall).

ADDITIONAL FEATURES Having a low-resolution, easily traversed representation of your scene's geometry gives you many possibilities beyond mere occlusion queries. Umbra 3 provides a basic API for querying line intersection and the shortest path between two points. This opens up many opportunities for game development, and doesn't require additional content or pipeline work. For example, you might not want to implement any complex AI pathfinding using Umbra, but for determining regions to stream in, it should be enough.

We have not really explored all the possibilities of using the data, but we're looking forward to digging into it in the near future. For example, we've discussed using Umbra 3 for audio occlusion. If the visible scene geometry is not enough to do this, it should be easy enough to create custom blockers using the same pipeline with slight modifications. Umbra 3 can also detect continuous areas, like separate rooms in a building, which can have many applications on the tool side as well.

CONCLUSION Umbra 3 has a simple and clean API like its predecessors, and does its job (determining object visibility) well. The steps involved for occluder simplification are well-defined, and the Optimizer produces high-quality portal information. There is no reason to have your level team working long nights doing something that can be automated with Umbra 3.

We have rarely experienced stability issues with any version of the Umbra runtime, and Umbra Software does offer the option to license the full source-code version, which makes it easy to debug if anything goes wrong. In our experience, Umbra Software's tech support has always been quick to respond, and we had no problem getting custom versions of Umbra 3 with new features or fixes we had requested. After evaluating Umbra 3, we're hard-pressed to find reasons why most game developers would write an occlusion library themselves.

***

Tatu Aalto is a senior graphics programmer at Remedy Entertainment. He is currently working on an unannounced triple-A game for the future generation of consoles.

Our universe is expanding GROW WITH US

» www.ccpgames.com/jobs

With offices in
Atlanta, Silicon Valley, Reykjavik,
Newcastle, and Shanghai

CCP

©2012 CCP hf.

BENCHMARKING WEB FRAM

TESTING WEB FRAMEWORK PERFORMANCE FOR FUN AND PROFIT

I AM CURRENTLY IN THE EARLY STAGES OF SETTING UP A MOBILE GAMES STARTUP CALLED JUICEBOX GAMES WITH A FEW CO-FOUNDERS. ALONG WITH SETTING UP A BUSINESS MODEL, PRODUCT ROADMAP, AND SPACE, THERE'S A VARIETY OF TECHNICAL DECISIONS THAT WE NEED TO MAKE. THESE CHOICES RANGE FROM PROCESS TOOLS (SUCH AS WHAT BUG-TRACKING DATABASE TO DEPLOY), TO COLLABORATION TOOLS (SUCH AS WHAT SOURCE-CONTROL REPOSITORY TO USE AND WHAT INSTANT MESSAGING SERVICE TO USE), TO GAME INFRASTRUCTURE CHOICES (SUCH AS A BACKEND FOR PLAYER-STATE STORAGE OR A MIDDLE TIER TO PROCESS WEB REQUESTS). SOME OF THESE DECISIONS ARE RELATIVELY EASY TO MAKE, AS THEY'RE INFORMED BY OUR PRIOR EXPERIENCE AND CONSENSUS, BUT WHEN IT CAME TO DECIDING THE FRAMEWORK WE WANT TO USE FOR OUR WEB-APPLICATION TIER, WE FELT THAT WE SHOULD GO THROUGH A MORE FORMAL DUE DILIGENCE AND SHOOTOUT PROCESS.

In our architecture—and, I suspect, that of many other online social games as well—the web tier is responsible for handling game-action requests from clients, fetching player state from persistent storage, executing game logic to perform those actions, and saving the updated state back to persistent storage. In deployments, I've seen the web tier comprise well over half of both operating costs and lines of code, so picking a framework is no light decision.

When picking a web tier for any application, there are a few criteria you might consider using to help you choose. The obvious ones here are:

- **Performance: How well equipped is the framework to handle your workload? Will it become an undue operating expense?**
- **Ergonomics: How easy is it to develop against? How easy is it to profile?**
- **Maturity: How stable and supported is the framework? Can we get all the modules we need to support our business?**
- **Popularity: How easy will this framework be to hire for?**

Since the latter three factors are somewhat subjective, and generally difficult to get a read on without significant exposure to the framework, we decided to focus on building a shootout that was exclusively focused on performance. In building out a benchmark for a few popular frameworks, we hoped to both exclude obviously bad choices, and to get a preliminary understanding of how well the solutions fit among some of the softer criteria.

HOW WE TESTED Having read a fair share of web benchmarks in the past, there were a few things I wanted to make sure we got right in this shootout. First, we wanted to make sure we tested with a realistic workload; many web benchmarks posted in the past focus on workloads that are either rather trivial, or overly CPU-intensive, and neither of those are actually likely to be encountered by many domains. I wanted to make sure we benched with a workload that demonstrated a plausible simulation of production workload. Second, we wanted to make sure the tests were run on a microcosm of what a real production deployment would look

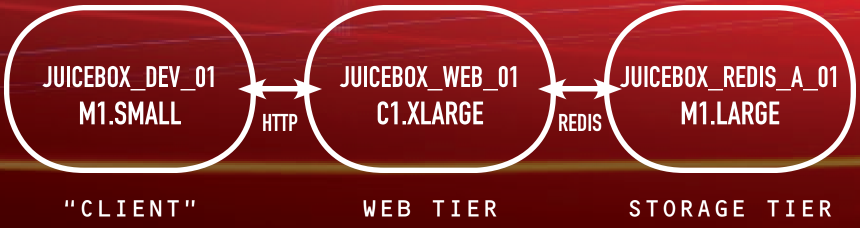


FIGURE 1: The three different dedicated AWS instances we used to generate a load for our web framework benchmarks.

like, and not just a single developer laptop where a mixed workload could cloud the results. Third, I knew that the results from other web benchmarks were often criticized because the engineers who ran them in the past didn't do enough to tune the frameworks for their workloads. Many of these solutions don't come pretuned out of the box, and it takes a fair amount of elbow grease to get them in reasonable shape. We endeavored to follow published best practices (where they truthfully yielded higher RPS) while tuning for our tests.

For this benchmark, we ran on three dedicated AWS instances (shown in **Figure 1**), and used Apache Bench (http://httpd.apache.org/docs/2.2/programs/ab.html) for generating the load. We experimented with increasing our m1.small to a higher CPU instance, but the m1.small with a single core was more than enough to generate sustained load (that is, enough to saturate the web box).

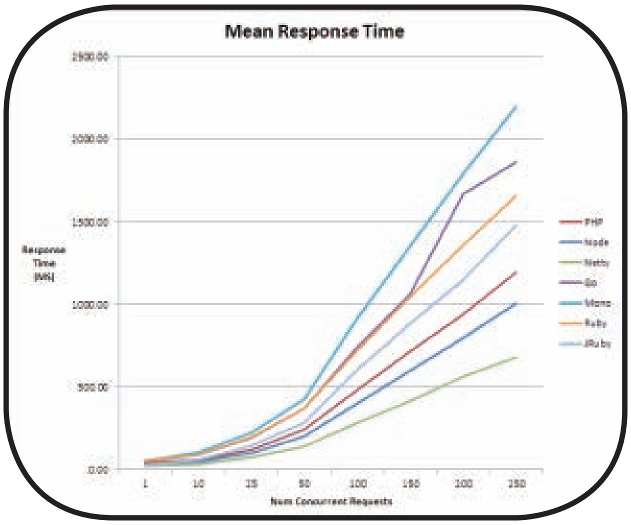
For this project, we tested a total of seven web frameworks, including a mix-and-match of older "blocking" frameworks, such as PHP and Ruby, and newer Async frameworks, such as Node, Go, and Netty.

- **PHP 5.3.3 running on Prefork Apache 2.2.15**
- **Node JS 0.8.14 with Cluster running with Nginx local proxy**
- **Netty 3.5 with Nginx local proxy**
- **Go 1.0.3**
- **Mono 3.0.1 with Mod_mono running on Prefork Apache 2.2.15**
- **Ruby 1.8.7 with Thin server 1.4.1 with Nginx local proxy**
- **JRuby 1.7.0 with Trinidad 1.4.4**

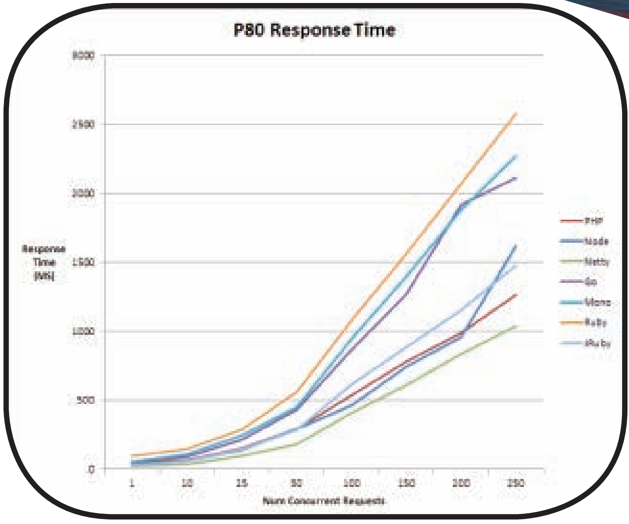
MODELING A GAME WORKLOAD There are many flavors of games out there—and many ways to write the backend of those games—so we chose to limit the benchmark to blob-based, sandbox games that are mostly single-player in focus. The workload for these games is generally characterized by the following processes:

1. **Fetching a document for a player from authoritative storage, such as memcache, couchebase, or in our case, Redis**
2. **Decompressing that document using some strategy (Gzip or LZO, for example)**
3. **Deserializing that document from its persistent representation—JSON decoding or using some other scheme**
4. **Doing some amount of game logic on that player state (this is usually mildly CPU intensive)**
5. **Reserializing player state into its persistent representation—JSON encoding or similar**
6. **Compressing the resulting document (again, with Gzip or LZO)**
7. **Sending that document back to persistent storage**

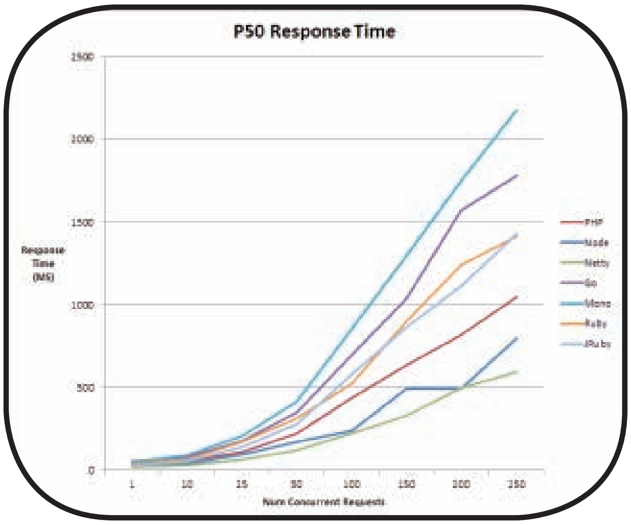
The entire operation is usually controlled by some distributed synchronization scheme, such as CAS or Surrogate memcache keys, and these can indeed cause additional load,



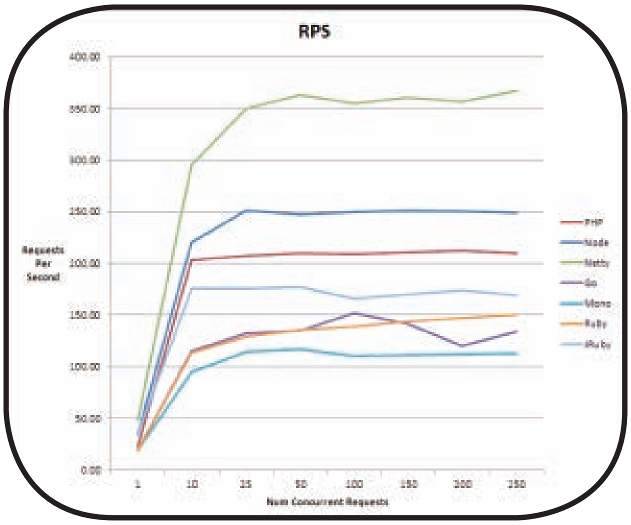
Workload 1: Mean Response Time (Smaller is Better)



Workload 1: 80th Percentile Response Time (Smaller is Better)



Workload 1: Median Response Time (Smaller is Better)



Workload 1: Requests per Second (Bigger is Better)

but this doesn't change based on the web framework chosen, so we'll leave it out of this discussion.

Players will post up transactions to their game state on a semiregular interval while logged into the app (say, every 10 to 20 seconds), and will do so by posting HTTP requests up to our web server. When profiled, this workload produces a walltime breakdown that looks like this:

20% Fetching records
20% Decompressing/deserializing documents
20% Game logic
20% Compressing/serializing documents
20% Saving records

This creates a workload that's roughly 40% I/O-bound, 60% CPU-bound, and on the whole is probably a bit more I/O rich than most other web applications. For this exercise, we'll define two representative game workloads and then benchmark our frameworks against them.

WORKLOAD 1: 40% I/O This workload represents a plausible social game workload for a player logged into our application. The pseudo code looks roughly like this:

- **Fetch a known gzipped JSON document from redis**
- **Unzip the document**
- **JSON decode the document**
- **Do 10k FLOPs as a rough simulation of game logic**
- **JSON encode the document**
- **Rezip the document**
- **Send it back to storage**

The document we'll use for the benchmark represents an elder player blob in our game. It's actually pulled from another popular social web title, which we obfuscated, though complexity and decode speed stays the same. (You can find the blob here: https://github.com/juiceboxgames/web-bench/blob/master/util/document.json) This player state is 100k uncompressed, 10k gzipped, and represents what I think is a pretty reasonable player state for a player.

WORKLOAD 2: 80% I/O Early on in the testing process with the first workload, we found that the zlib or JSON implementation present in each framework strongly dictated the performance we were getting. In other words, they became bottlenecks. To provide another lens for looking at these frameworks, we came up with another workload, which while not plausible for a live game server, gave us a better sense of how these frameworks dealt with an I/O-heavy workload without being clouded by specific library implementations (which can always be optimized).

The pseudo code for this workload looks something like the following:

- **Fetch a document from storage**
- **Do 10k FLOPs**
- **Save the document back to storage (retaining compression)**

Foundations of Digital Games

fdg 2013

www.fdg2013.org

Chania Crete Greece

14 - 17 May

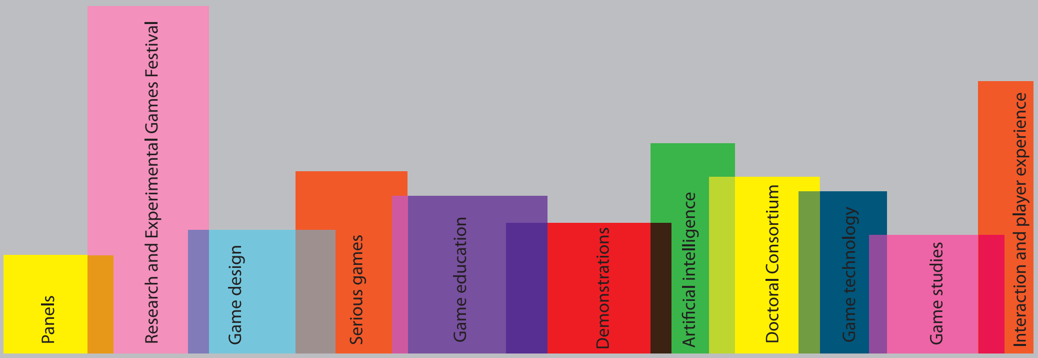
Take a step back from the trenches of development and explore topics that a looming crunch time often precludes. At the conference Foundations of Digital Games conference, for the 8th year in a row leading researchers and developers from around the world will be presenting their latest work.

Topics range from:
- adaptive games - procedural content generation - player psychology - game analytics
- player experience - game culture - games for change - game philosophy

Contribute to this forum focused at advancing the study of digital games both as cultural and technological artifact.

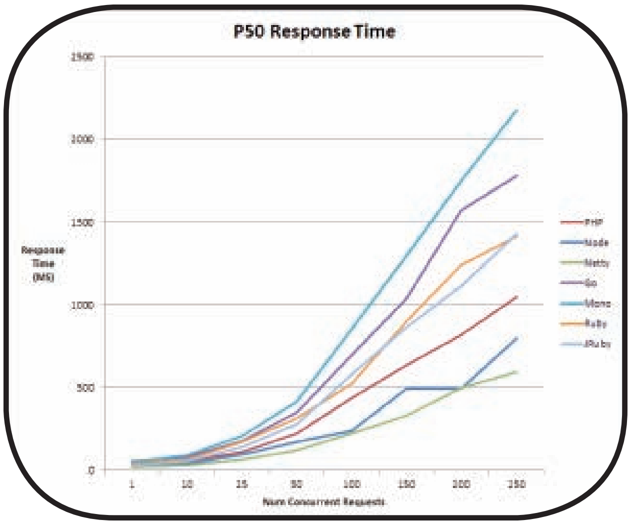
Join us in Chania, Greece for FDG 2013!

Tracks

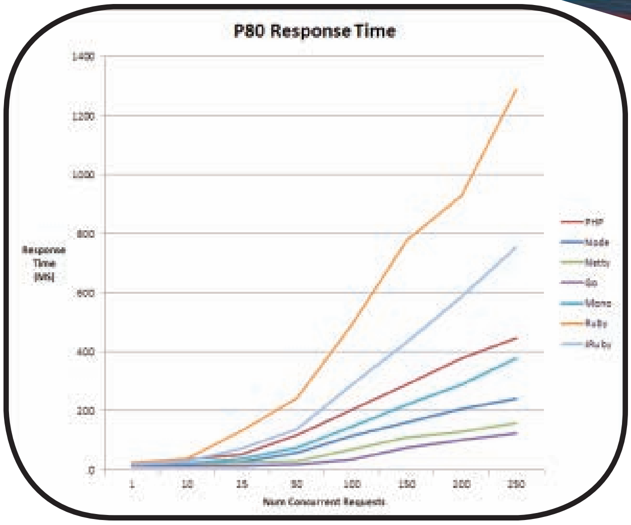


Microsoft Research   Microsoft Studios   unity

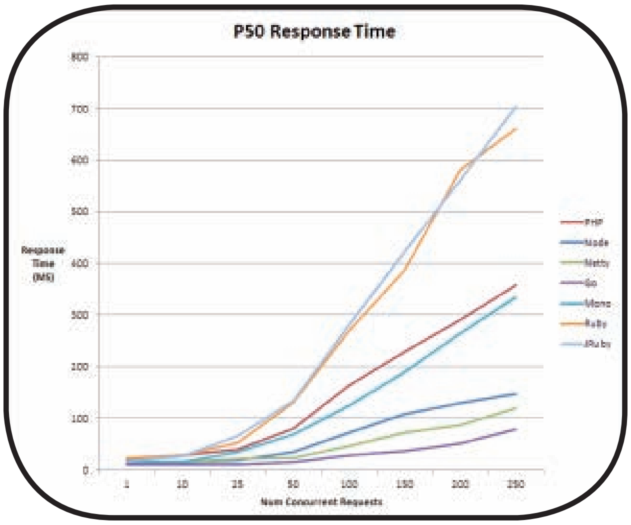
FDG 2013 is a conference of the Society for the Advancement of the Science of Digital Games



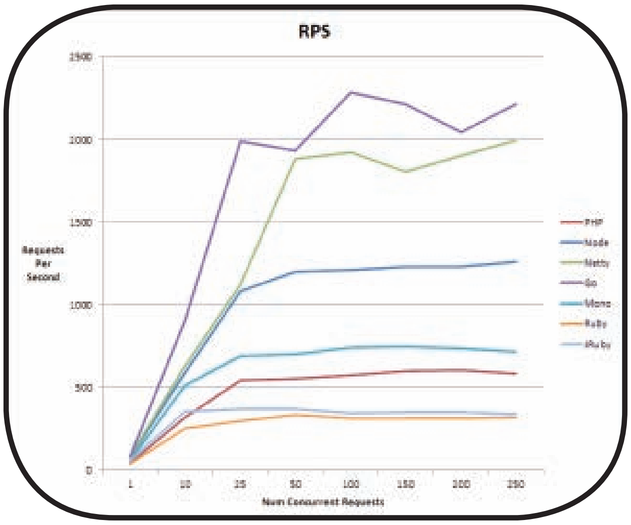
Workload 2: Mean Response Time (Smaller is Better)



Workload 2: 80th Percentile Response Time (Smaller is Better)



Workload 2: Median Response Time (Smaller is Better)



Workload 2: Requests per Second (Bigger is Better)

This produces a workload that's something closer to 80% I/O.

BENCHMARKING RESULTS Again, these were run on a c1.xlarge box (8 cores), and represent a reading from the first 10,000 requests (with concurrency noted).

WHAT WE LEARNED With the above testing data in hand, we were able to make a few conclusions about each different framework.

There's a marked difference in requests-per-second (RPS) and response time for new async frameworks and sync frameworks; this becomes especially obvious as the number of concurrent requests exceeds the number of physical cores on the box. Async frameworks achieve a peak RPS at above concurrent workers.

Workloads demonstrating a richer I/O workload (as a % of wall time) get larger benefits from using an async framework. Workloads with a pure CPU workload should demonstrate near-identical performance on both sync and async frameworks.

Go achieves very terrible scores on workload 1 and really excellent scores on workload 2. I think this is pointing to particularly crappy JSON and ZLib implementations present in Go.

Mono 3.0.1Ð's implementation of .NET 4.5 is super disappointing, and really not ready for prime time (async keyword isn't implemented yet).

Don't assume that system util libraries demonstrate good/best performance. The best JSON lib can be 10x faster than the worst.

There appears to be an emerging trend of decoupling a web framework from its server container (e.g., having a local nginx proxy). This introduces some new complexity in managing both processes and making sure that the guest web framework keeps its service and workers in a healthy state.

There are some clear winners and losers for ergonomics and popularity. Measured just by hours to develop, PHP, Ruby, and Node take the cake for speed of development—but again, this is very subjective, as it's dependent in part on my own prior experience.

Interestingly enough, if we had viewed the results only through the performance lens, we would choose either Netty or Go (with the assumption that we could close the gap on the ZLib and JSON encoding performance). While carrying out this exercise, however, we found that those two frameworks took us by far the longest to set up and configure, and they weren't particularly well-documented. Either of those may be a fine choice for certain applications, but for our purposes, we're willing to pay a small performance cost to improve development speed, and we feel like Node presents a good tradeoff here. ip

***

Jason McGuirk is the chief technical officer for Juicebox Games. Contact him at jason@juicehub.com or read more about Juicebox Games's development process at blog.juicebox.com.

GDC 13 NEXT

CO-LOCATED WITH ADC 13 APP DEVELOPERS CONFERENCE

LEARN. NETWORK. INSPIRE.

GAME DEVELOPERS CONFERENCE® NEXT

LOS ANGELES, CA
NOVEMBER 5–7, 2013
EXPO DATES: NOVEMBER 5–6

2013

GDCNEXT.com

UBM Tech

INFORMING, ENGAGING, AND EMPOWERING THE INDUSTRY

gamasutra.com

the art and business of making games

UBM Tech

THE GOAT IN THE PYTHON

TRENDS AND CHANGES IN APP SCRIPTING

Technology trends come and go all the time. You can't miss the darned things, because they are hyped up the wazoo: One year everything is "social!", the next it's "HTML5!" or "diegetic!" If you've been around computers for any length of time, it's easy to get jaded. That's why it's interesting when a trend sneaks up on you with less fanfare. Often, it's the trends that don't get a lot of hype that really stick. When a lot of people make the same decision on their own, for their own reasons, it's probably a pretty good idea, rather than just an empty buzzword. In the old phrase, it's an Idea Whose Time Has Come. For game teams, the Idea Whose Time Has Come lately is a pretty familiar one: scripting.

Nowadays, nobody is surprised to see the words "technical" and "artist" side-by-side on a business card, and if you describe yourself as a "scripter" nobody will assume you're just another barista with a messenger bag full of screenplays. What has slowly become apparent, especially in the last year or so, is that scripting isn't just a lightweight way to save artists a few button clicks or speed up tedious grunt work. From talking to databases, to parsing XML data files, to remotely controlling huge render farms, scripting is everywhere. It's the weld that holds the complex plumbing of our game pipelines together. So we're going to devote this month's column to a quick overview of the state of the art in art-tool scripting.

To get a sense of how deeply scripting has become embedded in the business of making games, consider this long but far from complete list of art packages that now include some kind of scripting:

APPLICATION	SCRIPTING LANGUAGE
Blender	Python
Cinema 4d	Python, Coffee
Houdini	Python
Flame	Lua
Illustrator	JavaScript
Katana	Python
Lightwave	Python, LScript
Mari	Python
Maya	Python, MEL
Modo	Python
MotionBuilder	Python
Nuke	Python
Photoshop	JavaScript
Project Messiah	MessiahScript
Rhino	Python
SoftImage|XSI	JScript, Python, VBScript
SketchUp	Ruby
Unity Editor	JavaScript, Boo, C#
ZBrush	ZScript

With a couple of exceptions (don't be embarrassed, Mudbox!) and a handful of programs that need plug-ins to support scripting (Silo, Hash Animation Master), pretty much every significant art tool released in the last five years has included some kind of scripting support. That's irrefutable proof of how important scripting has become.

Indeed, these days it is getting harder and harder to ship a serious art tool without a scripting language. Otherwise impressive tools such as 3DCoat and Mudbox have a hard time finding a home in the game business if they are perceived as isolated programs, rather than easily integrated components that can be plugged into the complex infrastructure of a modern studio. Without the ability to read custom file types, talk to databases, or integrate with source control, these tools may be popular aids for individual artists, but it's hard to see them becoming as indispensable as something like Max or Photoshop, which can be tightly woven into the life of the studio.

SNAKE CHARMERS It's not just art tools that bank on scripting as a way to become more tightly integrated into the production process. Source-control services like Perforce or Mercurial, for example, can usually be accessed using Python or JavaScript libraries. Asset management systems like Tactic (www.southpawtech.com) and Tank (www.shotgunsoftware.com/tank) offer Python hooks. Scripting provides these backend tools with a one-size-fits-all way to interface with a wide variety of regular art software: Any tool that supports Python can, for example, check the version history of an asset automatically, or auto-generate a notification in the asset database. Probably the most interesting example of the trend is a new project called Fabric Engine (http://fabricengine.com), which provides a set of high-performance graphics components designed to be linked together with scripting languages like Python and JavaScript. The goal is to allow users to create tools that have the performance of traditional heavyweight applications, but use the more flexible scripting languages to handle UI and communications with the rest of the pipeline.

Because interoperability is so vital, scripting languages themselves are becoming standardized. If you glance back at that list again, you'll probably notice another fact: The days of proprietary scripting languages like Lightwave's LScript or ZBrush's ZScript are fading fast. Standard languages—particularly Python, but to a lesser extent JavaScript as well—have largely supplanted the menagerie of proprietary scripting languages that sprang up after MEL pioneered the concept of scriptable art tools back in the 1990s. For current and aspiring tech artists, the subtext is also pretty clear: If you want to stay current, you probably need to learn Python. The choice of a language is as emotional for coders as the choice among Max, Maya, and XSI is for production artists. However, in this case the point is not that Python is the "best" language—merely that it is the most common, and therefore the one you're most likely to need for your career. It's certainly possible that Microsoft's push toward JavaScript for Windows 8 development, along with the appeal of web-based tools, may turn JavaScript into a serious force in pipeline development—but as of today, it's hard to see a future where most TAs won't need at least passing familiarity with Python.

SNAKES ON A PIPELINE Python has become so ubiquitous because it offers a huge library of features for common tasks. Python fans like to say that the language comes with "batteries included"; almost anything you need to do can be tackled with the standard toolset. If you want to download a file using FTP, say, or read XML files, there are readily available libraries to speed the task. Tackling the same tasks in a narrowly focused mini-language like ZScript or LScript, on the other hand, is reminiscent of the old saying, "It is not done well; but you are surprised to find it done at all." Python's rise to scripting dominance owes a great deal to its large and reliable collection of standard libraries for just about every imaginable computing task, from asynchronous programming to zip file management. While most of the same functions can be done in other languages using a variety of extension modules, no other language offers so much functionality for tool developers right out of the box.

Python is everywhere these days, but it's by no means perfect. It's fairly slow, even by the standards of scripting languages, and it demands a fairly significant chunk of memory. These flaws used to be a serious drawback; many programmers (particularly those who make their living squeezing every last millisecond out of aging console CPUs) look on it with a certain degree of suspicion. Lighter, simpler languages like Lua are more common for game scripting and AI work. For tool development, however (especially on big beefy modern computers!), Python's speed and memory

MAKE MORE ENEMIES

VFS student work by Benjamin Erdt

Game Design at VFS lets you make more enemies, better levels, and tighter industry connections.

In one intense year, you design and develop great games, present them to industry pros, and do it all in Vancouver, Canada, a world hub of game development.

VFS

Find out more.
VFS.COM/ENEMIES

THE ONLY ONE-YEAR PROGRAM IN PRINCETON REVIEW'S **2012** **TOP GAME DESIGN PROGRAMS**

problems matter less than the cleanliness of the language and the enormity of the built-in toolset.

The appeal of that huge set of ready-made functionality has encouraged many teams to extend their in-house tools with scripting. Adding a scriptable interface gives homebrew tools the same power that it grants to third-party applications: instant access to a huge range of useful features and easy integration into the pipeline universe. Of course, scripts aren't appropriate for everything—Python and JavaScript can do a lot, but compared to C++, Java, or C#, they are still pretty slow (as much as 100 times slower for some kinds of intensive calculations!). Custom tools written the old-fashioned way remain a vital part of most studio ecosystems: World editors, 3D previewers, and specialty tools that pack data in the correct formats for different output media, for example, will still be written in traditional compiled languages for the foreseeable future. However, it's increasingly easy to extend these tools with script interpreters. Python was designed for embedding in C and C++ programs, and there are flavors of Python that can easily be integrated with tools written in C# (IronPython), or Java (Jython).

This is not always a completely carefree process. It's still up to developers to actually expose the functionality of their programs to the script interpreter. The standard tool for this is an open-source project known as the *"Simplified Wrapper and Interface Generator,"* or SWIG (www.swig.org), which is widely used despite the fact that "simplified" is an optimistic description. IronPython and Jython can actually import the hosting application's code directly into their respective versions of Python, which simplifies the relationship between the host app and the scripts considerably. Unfortunately, these alternate versions are not completely compatible with default Python. The language remains the same, and individual scripts can be shared among all the different flavors of the language. However, many of those all-important library modules don't work identically across platforms. For example, IronPython only recently gained the ability to read and write zip files after a wait of several years. Jython's implementation of Python is still stuck on Python 2.5, while most of the Python world (including, notably, Maya) uses 2.6 or 2.7. These are real drawbacks—but the attraction of being able to share script code between in-house tools and outside apps like Maya or Houdini makes them seem much more manageable.

SHEDDING SCALES? Almost all of the pieces that make up the modern studio scripting environment have been around for quite some time, but in the last couple of years they've really reached a critical mass. There are now so many script-enabled applications—and so many script-capable tech artists and tool programmers—that the entire way game pipelines get made is metamorphosing.

Tech artists, riggers, and power-user artists are being tasked with ever larger and more sophisticated problems. Old-school scripting focused on automating boring tasks inside an art tool's UI instead of having a rote sequence of steps: "Apply UV maps to all these faces, scale the UVs from 0 to 1, and apply the concrete material" or "Find all the objects with 'temp' in their names and delete them." These days, tech artists have to deal with all sorts of technological arcana that used to be the exclusive province of "real" programmers. Sit in on the Tech Art roundtable at GDC and you'll see artists arguing about things like the right way to batch-process SQL queries or how to create a new wiki page using HTTP POST.

With all these powerful tools at their disposal, TAs are managing larger and larger chunks of the game pipeline. Until pretty recently, getting custom data out of an art tool and into the game usually meant hunting up an engineer to write a C++ plug-in. Since few of these engineers actually used the art software on a regular basis—and because the job was often a thankless chore passed on to the least-experienced coders—the plug-ins rarely matched the host packages in sophistication, reliability, or user-friendliness. Nowadays, though, it's increasingly common for a TA to handle writing data out, typically in a standard format like XML or JSON. This leaves the UI in the hands of folks who know the tool well, while freeing the engineers to concentrate on hardcore problems like performance and memory management. A host of data-wrangling tasks are also being taken over by TAs; things like collecting statistics about asset usage in the game, finding textures that are referenced in any materials, or doing search-and-replace changes to game levels are common examples of jobs that used to be the exclusive province of tool programmers but are increasingly falling into the province of "technical art."

Now, if all these programming terms and acronyms sound like gibberish to you, you might wonder if tech artists are being seduced by the Dark Side and morphing into engineers. For most of the last decade, the technical art community has consisted primarily of production artists who graduated into more technical roles, sometimes under the pressure of circumstances and sometimes following personal inclinations. In the last few years, however, the discipline has become deeper and more specialized—and harder to break into. It's fairly easy to learn MaxScript or Mel by cutting and pasting bits out of your script listener window. Learning how to design and maintain a complex tools ecosystem built on object-oriented design patterns, on the other hand, is a much more complicated undertaking. It demands more specialization and more study—and the stakes are also a lot higher. It's not surprising that many art and vocational schools are starting technical art curricula that train students to be scripters from the ground up.

This kind of specialization is a mixed blessing. The brave new world of scripting has a lot of exciting opportunities for the technically minded artist, but it's also a more demanding place. The TAs of the future will need to know enough about serious programming that they can be trusted to build critical infrastructure that keeps the studio running, which is far harder—and more stressful—than cobbling together UI scripts. As the scripting-based ecosystem around game systems gets more mature and more complex, it's inevitable that a caste of dedicated specialists, with specialized skills, will arise to tend it. At the same time, TAs need to stay true to their historical roots as artists. If the discipline loses touch with the needs of production artists, TAs really will be "just programmers," and their ability to make life better for their fellow artists will be seriously undermined.

It's always stressful to be a generalist in our specialist-obsessed business. Let's hope that all the cool new toys in the scripting toy box keep those TAs happy as they struggle to keep current with both their artistic and technical selves. pp

***

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.

ILLUSTRATION: JUAN RAMIREZ

ACADEMY *of* ART UNIVERSITY®

FOUNDED IN SAN FRANCISCO 1929 BY ARTISTS FOR ARTISTS

Gil Weinstock

Wala Alhadad

STUDY ONLINE OR IN SAN FRANCISCO

Acting*
Advertising
Animation & Visual Effects
Architecture
Art Education
Art History
Fashion
Fine Art
Game Design
Graphic Design
Illustration
Industrial Design
Interior Architecture & Design
Jewelry & Metal Arts*
Landscape Architecture
Motion Pictures & Television
Multimedia Communications
Music Production & Sound Design for Visual Media
Photography
Visual Development*
Web Design & New Media

ENROLL NOW

EARN

YOUR AA, BA, BFA, MA, MFA OR M.ARCH ACCREDITED DEGREE

ENGAGE

IN CONTINUING ART EDUCATION COURSES

EXPLORE

PRE-COLLEGE ART EXPERIENCE PROGRAMS

WWW.ACADEMYART.EDU

800.544.2787 (U.S. Only) or 415.274.2200

79 NEW MONTGOMERY ST, SAN FRANCISCO, CA 94105

Accredited member WASC, NASAD, CIDA (BFA-IAD, MFA-IAD), NAAB (M.ARCH)

**Acting, Jewelry & Metal Arts and Visual Development degree programs are not available online.*

Visit www.academyart.edu to learn about total costs, median student loan debt, potential occupations and other information.

GLaDOS.

SHOULD GAMES HAVE STORIES?

HOW TO MIX NARRATIVE AND PLAY

Stories and games have always had an uneasy marriage. From the beginning, designers have written stories into their games, giving the player a fixed beginning, a narrative path to follow, and a preset ending. At the same time, many players flocked to games because of their lack of narrative structure; a game experience is a chance to create a story, not to subject oneself to a designer's unpublished novel.

At the root of this problem is an almost theological dilemma: Can a game designer tell a story if the player's choices actually matter? If the most important element of a game is its interactivity, then every static plot point a designer crams into the experience takes away from the centrality of the player. Put another way, if a game has a spoiler, is it really still a game?

To be clear, with the exception of a few abstract games like TETRIS, almost all games benefit from story elements: an interesting setting, a distinctive tone, memorable characters, engaging dialogue, dramatic conflict, and so on. The best games have characters and settings that rival those of any other media—consider GLaDOS from PORTAL, or Rapture from BIOSHOCK.

However, the actual narrative of a game—meaning the series of events that determines the plot—is the hardest element to reconcile with the essential interactivity of games. For this reason, narrative cannot be handled as it is with books or movies, in which the story is the core element that everything else must support.

Consider how Sid Meier added story elements to PIRATES!, a game set in a period dripping with narrative possibilities. Instead of creating a single swashbuckling tale, with fixed plot points and a preset ending, he filled the game with the bits and pieces of a traditional pirate story. Depending on his choices, the player can rescue a long-lost sister, duel an evil Spaniard, survive a treasonous mutiny, discover buried treasure, escape from prison, and woo the governor's daughter. Upon retirement, the game displays the notable events of the pirate's life, chronicling the ebbs and flows of fortune. While the plot of a single playthrough would suffer in comparison to that of an authored work, the events have a special meaning for its intimate audience of one.

BASTION.

However, not every game is well-suited to become a dynamic story generator; some themes and mechanics are best handled against a mostly fixed backdrop. A hero needs an evil wizard to slay, a soldier needs an enemy to fight, and a plumber needs a princess to rescue. The solution is to use a light touch, to suggest rather than to dictate, to let go of the very idea of plot. Let the player explore the world and then assemble the final story in her own head.

"THE ROLLING STONES CONFIRMED THAT LYRICS ARE MOST EVOCATIVE WHEN JUST SHORT OF INDECIPHERABLE."
—PAUL EVANS, THE ROLLING STONE ALBUM GUIDE

Indeed, the role of narrative in games is more akin to the role of lyrics in music. A song's words give the piece its context, its mood, and its setting while still leaving a suggestive gap for the listener's imagination. Indeed, recordings often have lyrics that are inaudible, leaving the meaning intentionally obtuse. Would a writer ever do the same with the text of a novel? Further, people often enjoy songs in a foreign language—even if they don't understand the lyrics. How many readers pick up a book in a different tongue? The exact meaning of a lyric is not its primary role; great songs leave room (often a great deal of room) for the listener. So, too, must a game's narrative leave room for the player.

Consider LIMBO, the puzzle platformer noted for its atmosphere, with its monochromatic tone and minimalist audio. The game's story revolves around a very primal quest—a boy's search for his missing sister—and raises more questions than it answers. Why is the boy looking for her in a dark, mysterious forest? Why is he chased by a monstrous spider? Who are the kids trying to attack him? Although LIMBO is completely linear, the lack of a traditional narrative, with a plot, dialogue, and answers, means the story must be written by the player.

Another example is ATOM ZOMBIE SMASHER, the micro-RTS about a patchwork military trying to stop a zombie apocalypse in the fictional South American city of Nuevos Aires. The game is peppered with gonzo vignettes ("Esposito scores the winning goal. Minutes later, he's eaten alive."), showing how the citizens handle the onslaught. The epilogue is a masterpiece of

ILLUSTRATION: JUAN RAMIREZ

Malak.

bizarro narrative, with scenes of a cyborg El Presidente and AK-47 fruit trees backed by President Eisenhower's famous "military-industrial complex" speech.

Most importantly, ATOM ZOMBIE SMASHER creates an evocative world without a traditional, canned narrative; the vignettes, in fact, are delivered at random during the campaign, letting the player's imagination fill in the gaps. Brendon Chung, the game's designer, points out that "piecing information together is fun, and knowing the work trusts and respects you is satisfying." The effect is perhaps a bit too jarring for a mainstream audience, but the result is that Atom Zombie Smasher feels so much more open and alive than any predigested corridor shooter or bloated, dialogue-heavy RPG. A fixed plot is the enemy of player engagement.

THE QUESTION OF CHOICE One of the most tantalizing aspects of mixing video games and narrative is the possibility of interactive fiction, in which the player gets to make the big decisions in an otherwise traditional story. So far, this potential is unrealized, as the player's choices are usually limited to selecting among a few preset branches. Although there may be more than one ending, as long as the outcomes are finite, interactivity only promises a difference in degree, not in kind.

As the cost of production rises, developers cannot risk creating sections of a game without guaranteeing that the player will experience them. Thus, regardless of player choice, the interactive storyline must synchronize at key points. The plot of KNIGHTS OF THE OLD REPUBLIC exemplifies this problem. The player can pursue a good or evil path, but both paths lead to the same place; the villain Darth Malak must be defeated, either to stop him (the good path) or to usurp him (the evil path). Even with completely divergent ethical paths, no outcome is possible without Malak's death.

These static plotlines lead to a jarring disconnect for many players, who might spend tens of hours playing an RPG, but will leave no lasting memory of the story, because it has nothing to do with the player's own interests or choices. Ultimately, people write stories to share what it means to be human. What does that goal mean in the context of games? The core element of most stories is the choices made by the characters; the core of games is the choices made by the players. Thus, what makes games meaningful must be the choices made by the players themselves. Can a game ever tell a specific story and still preserve the importance of player choice?

The action-RPG BASTION successfully tackles this dilemma. The game tells the story of a mysterious "Calamity" that shattered the world into pieces. As the player progresses, he learns why the weapon that caused the disaster was created and what went wrong when it was triggered. At the game's conclusion, the player must choose between either reversing time to possibly prevent the Calamity, or to evacuate the survivors to a safer place and a new start.

What is most interesting about this decision is what happens next: Almost nothing happens. The game simply ends, with only a single image reflecting the player's choice. The designers do not pretend that they are giving the player actual agency with this decision. Instead, the choice becomes almost meditative, a simple reflection of the player's own nature. Would you undo your greatest mistake, or would you move forward as a new person?

"THE FUNCTION OF PRAYER IS NOT TO INFLUENCE GOD, BUT RATHER TO CHANGE THE NATURE OF THE ONE WHO PRAYS."
—SOREN KIERKEGAARD

In BASTION, the player learns about herself through the act of making a choice, not from seeing what some designer thinks should be the result. In THE WALKING DEAD, the designers emphasize player choice by providing feedback on how one's choices compare with those of other players. These results similarly illuminate the player's own personality by showing which of his decisions go with or go against society at large.

Games that focus purely on the designer's plot choices ignore that the most important part of a game is the player. Putting a story, regardless of its power or depth, inside a game is actually a crutch, an easy way out that stunts the advancement of our form. Games must leave room for the player, not just within the rules and the mechanics and the systems, but within the story as well. d

***

Soren Johnson was the co-designer of CIVILIZATION III and the lead designer of CIVILIZATION IV. He is a member of the GDC Advisory Board, and his thoughts on game design can be found at www.designer-notes.com.

ATOM ZOMBIE SMASHER (right), BIOSHOCK (left).

BECOME A LEADER IN **DIGITAL MEDIA**

With digital media in mind from conception to completion, the new CENTRE FOR DIGITAL MEDIA features student apartments, project rooms and classrooms all designed to inspire creativity and collaboration. Located in Vancouver, Canada the new CENTRE FOR DIGITAL MEDIA offers a full and part-time Master's program that focus on real-time, industry-facing collaborative projects.

Learn more about our MASTERS OF DIGITAL MEDIA PROGRAM at:
www.thecdm.ca/programs/mdm/gd

The future of work is at the new CENTRE FOR DIGITAL MEDIA.
www.thecdm.ca/gd

Application Deadline: Feb 15th, 2013

UBC SFU emily carr BCIT

SUPERCHARGE YOUR GAME DEVELOPMENT SKILLS

NVIDIA's GPU Technology Conference has expanded to include cutting edge mobile and game development sessions. Spend time with Valve, Gearbox, and NVIDIA's game development engineers in a friendly, fun and collaborative environment. Come learn first-hand about the benefits of a partnership with NVIDIA and walk away with the latest development techniques!

Register at **www.nvidia.com/gtc**. Use 10% discount promo code **GM10GDM**.

Thank you to our sponsors: Adobe hp DELL PNY Microsoft acer NVIDIA

WHATCHAMACALLIT

ACHIEVING TRANSPARENCY IN NAMING

The ability to see from one end of the pipeline to the next is often impossible on a large project. A simple sound file must weave its way through wormholes of coding, levels of abstraction, and a cornucopia of naming standards in order to eventually be played back by the game. When you're trying to figure out why something that *should* happen *isn't* happening, you might have to take a trip down your game's rabbit hole, and that can be tricky if you can't clearly see how the process works. So: What does it mean for a pipeline to be transparent, and why should we be striving for transparency at every step of the way?

I JUST MET YOU I often find myself dropped into a project long after its groundwork has been laid. For me, I get the chance to admire the divine interactions that have been established, and work to dissect the idiosyncrasies that define the creative process. More often than not, I'm looking for the truth within an implementation: something that immediately defines a working methodology. Sometimes the truth is easy to see. For example, take this case of a simple scripted event that plays a single sound file:

- **A lever model in-game named: lever_awesome_01**
- **lays an animation called: lever_awesome_01_on**
- **Which triggers a sound event called: playSound_lever_ awesome_01_on**
- **Ultimately playing a sound file called: lever_awesome_01_ on.wav**

In this example, it's easy to see what the implied relationship is from top to bottom, based on the naming standard that has been maintained throughout the pipeline. Being able to enable the in-game debug, find the model, view the animation, and know where to look for the sound event and/or eventual sound file makes life easier if anything ever goes astray.

If your colleagues *hadn't* adhered to the project's established naming convention, you can imagine the amount of detective work you'd have to go through to find out that the "lever_awesome" model is driven by an animation set called "naughty_lever," which triggers a misspelled sound event called "lever_of_aawsome," and the eventual "leever_001.wav" sound file. There are worse things in the world, but every little bit of sanity you can bring to the pipeline contributes to peace of mind at the end of production.

THIS IS CRAZY Of course, not every pipeline is built in a day, and very few of them are as simple as this example. Most pipelines take months and years to develop, involve multiple disciplines, descend deep into the mouth of madness, and are subject to million-mile-a-minute fat-fingered misnamings. But if you and your team can at least start off by keeping some semblance of consistency in mind when you establish a naming standard, your discipline will have a ripple effect through development, and people who must step into the system later and try to wrap their heads around it (people like me, for example) will appreciate it.

But this isn't just about naming standards. While some of you may be looking for a stone tablet that establishes, once and for all, the perfect way to name things, the truth is far more fluid than that. Just like any iterative process, you'll be implementing, living with, changing, and reimplementing naming standards for the rest of your career. The point is not that your naming standard must be perfect; it's that you have one to begin with,

ILLUSTRATION: JUAN RAMIREZ

and it's designed with the intent that others may inevitably hope to understand it. Whatever way you go about defining it, try to maintain consistency, and try to keep in mind human readability.

HERE'S MY NUMBER Of course, you won't always be able to maintain a consistent naming standard. There are always points in the pipeline that demand divergence, and recognizing those points is the first step toward understanding when you must abstract the naming consistency and carry on. Take, for example, a group of sound files that will be played back randomly:

- **physics_impact_wood_small_01**
- **physics_impact_wood_small_02**
- **physics_impact_wood_small_03**

These would often be grouped together and named so that the game engine can request the group name, which would then determine a random sound file. One way to keep your naming scheme consistent is to drop the numerical delineation and call the group "physics_impact_wood_small." While it might seem just as valid to drop other parts of the naming standard (by calling the group "impact_wood_small" or possibly abbreviating to "impact_wood_sml," it's important that when you make that choice, you understand the repercussions downstream within the pipeline.

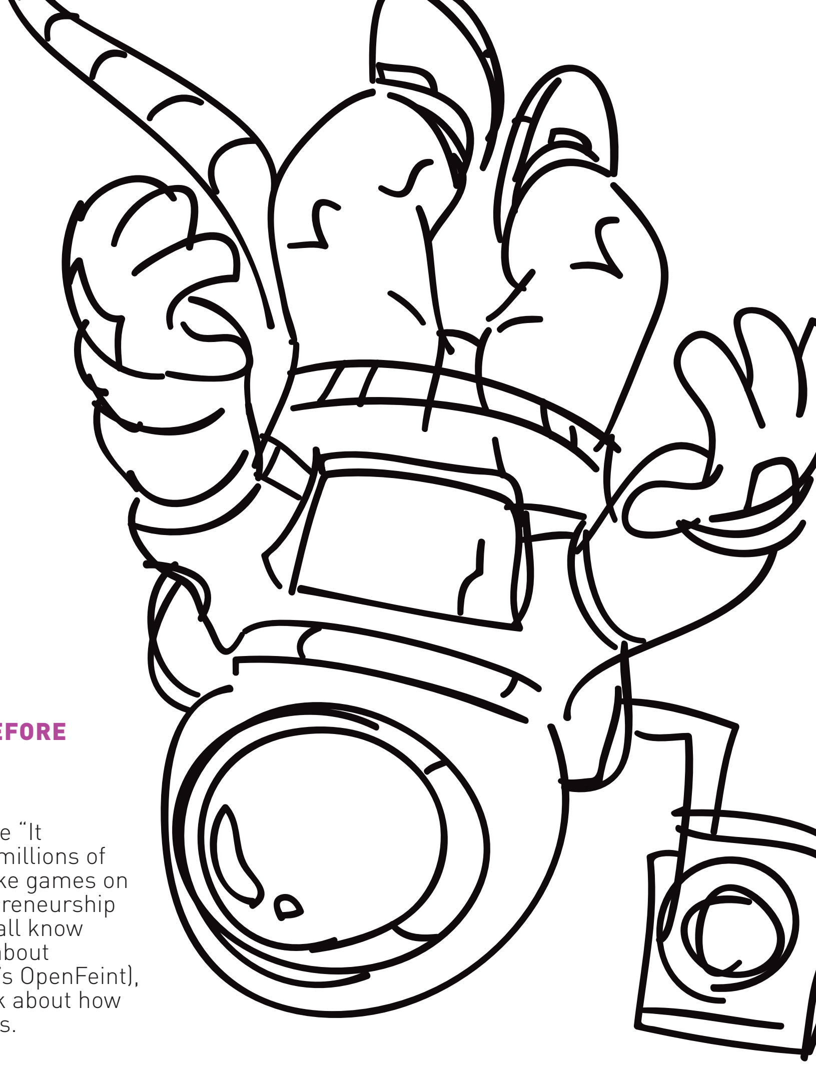
If you're searching for all "small" sound files and groups, will you think to look for both "small" and "sml?" From the in-game debug that shows the sounds and sound groups playing, will you be able to immediately identify the correct impact sound is playing based on the name printed on the screen? It's possible that the system you're working in allows for a great deal of visibility at every level of the pipeline. Regardless, striving for transparency from end-to-end allows for a greater understanding of what is happening at-a-glance; keep consistent, and you might be able to spend less time Sherlock Holmes-ing solutions and more time making your game sound good.

CALL ME, MAYBE? There's no way to anticipate every pitfall in the moment, but if you can keep in mind how an asset might be used when architecting your project's naming convention, you just might be able to keep your pipeline transparent (or at least somewhat coherent). Keep your pipeline transparent, and you'll be able to fix problems faster and stay focused on the jam at hand. af

"A stitch in time saves nine."–popular saying

***

Damian Kastbauer cultivates musical earworms (KHAAAAAN!) at LostChocolateLab.com and on Twitter @lostlab.



IT'S DANGEROUS TO GO ALONE; TAKE THIS

TWO QUESTIONS YOU NEED TO ASK YOURSELF BEFORE QUITTING YOUR JOB AND GOING IT ALONE

Game startups and independent dev studios have become the "It Thing" over the last several years. Whether you're drawn by millions of downloads and millions of dollars, or you simply want to make games on your own terms, the Hollywood-style glamorization of entrepreneurship has led thousands of hopefuls to launch game startups. We all know about the many resulting successes, whether we're talking about publishers (Neil Young's ngmoco), middleware (Jason Citron's OpenFeint), or developers (Anil Dharni's Funzio). But we don't always talk about how this trend has also led to lots of carcasses of dead companies.

Before you quit your job, tell your significant other that you're starting your own company (and probably give them a heart attack in the process), or even begin to write your first line of code, there are two fundamental questions you must ask yourself:

1. **Do you have the emotional strength to withstand the startup rollercoaster?**
2. **What are your core competencies and weaknesses? Is it the right combination to succeed?**

1. ARE YOU A BAD ENOUGH DEV TO HANDLE A STARTUP?

Startups are hard. The press, conference circuits, and reality TV shows may tell you otherwise. Don't believe them.

Before your game launches, you'll spend long hours coding, designing, and playing all games under the sun that are similar to yours (or similar to the way you want yours to perform in the market). If you were lucky and were able to bootstrap or raise a small amount of funding, each person in the startup is making some money, though perhaps much less than they would on the market. If you aren't that lucky, you are putting all that time in without cash flow, hoping against hope that once you launch, you make enough money to cover costs for servers, marketing, and other basic expenses.

If you have a live freemium game, you work even longer post-launch, because you'll need to make sure players can get to the game. You'll have to maximize your player distribution, often with low marketing funds, in a market where bigger, better-funded game companies are driving up customer acquisition costs (the install floor for Android has increased from 20 cents to 35 cents, with an average of $1.35 during the last 12 months, according to our internal statistics based on the companies we advise). You are trying to get press, but they aren't taking your calls and instead are covering a better-known competitor who just cloned your game and touted your uniqueness as their own. You have four weeks' worth of cash left, a fact that you are trying to hide from the employees, who are considering major life decisions that will impact their finances—decisions like getting married, having a baby, or buying a house. And maybe you have talked to your parents and friends about using their couch or office floor to sleep on—if you haven't already done that to save on rent money so you can plough it back into your startup.

As all of this is going on, you are trying to negotiate with expensive service providers who are telling you what you should do, even though it's fairly obvious they do not know what they are talking about. Investors keep taking up your meeting time to pick your brain on the game's ecosystem but won't give you money, saying you are too early, or that they don't invest in hits-based businesses yet, and telling you to come back once you have more traction—as if you would need them then.

Eventually your optimism will crack, and self-doubt will begin to bubble through. Do you have what it takes? Can you really make this happen? Though you may feel that doubt, you can't show it—not to your co-founder, or your team, or advisors, or investors, or anyone else. After all, your motivation and passion feeds these people. They take their cues from you about how motivated and passionate they feel about your startup. You've got to be your startup's biggest champion even when you don't feel like it, because you know perseverance pays off, and honestly, you don't have a choice.

You try a lot of things along the way that you have never done before. Some of these work, but many others don't, and all you can do is lick your wounds and summon the emotional strength to get up and try again. Startups are about learning, and now more than ever in your life, how fast you and your team learn makes the difference between dying, surviving, or thriving.

Everyone goes through this, though it's rare that anyone will talk about it. We are all working for that home run, but it takes lots of practice and lots of games before you get there. The question is, do you have the courage and the emotional strength to go through the bad to make it to the good? Can you persevere when you least want to?

If the answer is yes, then maybe you have a chance of hitting that home run. In the words of Jeff Arch, writer and director of *Sleepless in Seattle* [source: *The Success Principles* by Jack Canfield]:

"It's like the Spanish conquistador Hernando Cortez in 1519... after he landed in Mexico, he burned all of his ships. Well, I've rented new ships just for the sake of burning them. I took out loans on ships that weren't even mine. I'm throwing money, credibility—every single thing there is—into my new project. And it's either going to be a home run or a strikeout—not a single or a double... I think one of the secrets to my success is that I'm willing to be terrified, and I think a lot of people are not willing to be scared to death. And that's why they don't achieve the big dream."

Are you willing to be terrified? Can you burn all your ships and keep going?

2. DO YOU HAVE THE RIGHT STUFF? It's no surprise that those starting games startups are usually very optimistic people; it takes a certain amount of rose-colored tint in your glasses to be willing to take the personal, financial, and social risk inherent in beginning a startup. This optimism is an asset that gets you onto the startup path, but can also be a liability if it makes you oblivious to a clear understanding of your internal weaknesses and the limits of your competencies—and, moreover, whether those will be the right combination to succeed in the current game industry.

Over the last 12 months, I've met thousands of developers at the various YetiZen events who rush to show me their game. Most of these are amazing games that I absolutely love as a player and will play again. However, amid the optimistic fervor of their game vision coming to reality, most devs tend to overestimate:

- **The ability of a great game to help them get discovered and for their players to stay.**
- **The attention span a player will give their tutorial section, leading to UI confusion during gameplay.**
- **How long the player will be playing the game (and therefore when monetization needs to be plugged in).**

They also underestimate:

- **The threat of competitors' fast follows (companies cloning most of your game while improving on a few choice features), or that others may already be developing or launching the idea you have been working on in stealth for months.**
- **The cost-structure realities of the market around cost of distribution and servers.**
- **The importance of being clear on the future vision of the company—will you exit to another larger company, or keep making games in the future? You should know on day one and build in the processes and systems necessary so that when you are ready to sell you have that choice, and if you never want to sell you do not waste your time on entertaining selling offers.**

Even if you or your co-founders are the best game designers in the world, you can't assume that your great game will help you succeed. Spend some time to assess why others have succeeded and failed—and identify what you have in common with them. How can you either focus on using the strengths to succeed or minimize the weaknesses?

Running a startup can be one of the most intense experiences of your life—and one of the most rewarding. If you feel you have the startup bone in you, ask yourself those two questions. I look forward to hearing from you as you create the next big gaming business. Good luck! b

***

Sana N. Choudary is the CEO and founder of YetiZen, which includes the YetiZen Innovation Lab http://yetizen.com/innovation-lab/ (a game industry community space) and the game-focused YetiZen accelerator program http://yetizen.com/about-yetizen-accelerator/. You can follow her and YetiZen at http://yetizen.com/blog/ or ask her a question on Twitter at @SanaOnGames https://twitter.com/SanaOnGames.

ALWAYS ASK

Negotiating contracts is always a tricky situation, especially when you're a newer developer or are in a disadvantageous position (such as needing to sign something immediately to get your team fed, for example). But in my experience, it never hurts to ask for more than what's offered, and it's surprising how often people just don't do it.

I'll admit, my experience is somewhat limited. I've probably signed just over a dozen contracts in my career. But after getting burned just once by vague terms that kept me obligated to work once the money had dried up, I started to ask for more, every time, while being more careful about my terms and provisions. And every time I've gotten what I wanted.

I'LL GLADLY PAY YOU TUESDAY... As an independent game developer without significant savings, I live or die by the contracts I sign with publishers. The money that comes in from them pays my rent and my team's rent, so any extra bit helps. That's why it's a good idea to ask for more than you estimate a project will actually cost; who knows, you might really need that extra couple-month buffer to finish the project! Or maybe you finish it in your initial time frame, and have some wiggle room to prototype before your next pitch. People who are used to living lean (read: indies and mid-size developers in dire straits) tend to estimate very conservatively as well, so this can help make your budgets more realistic.

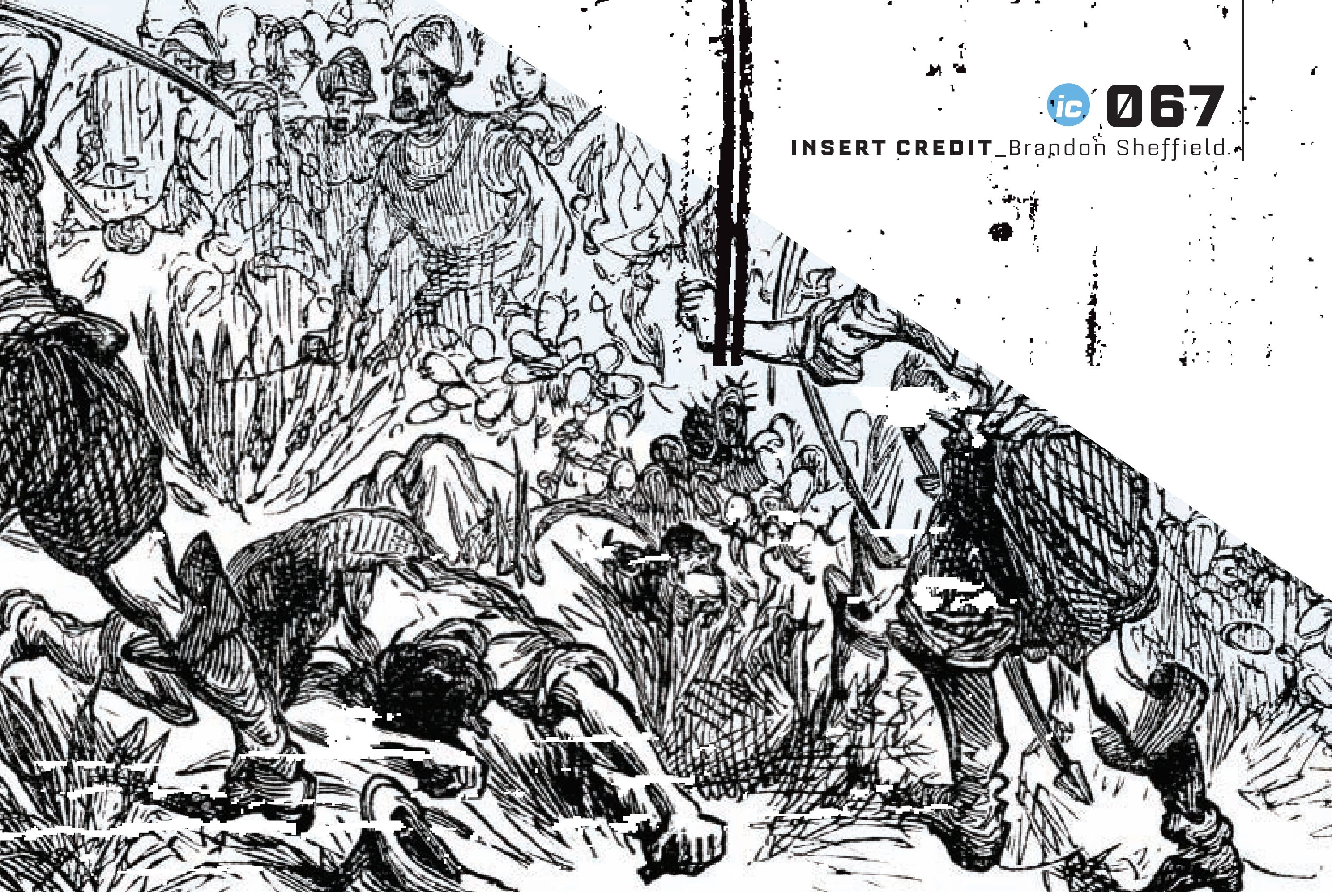
You'll understand that I have to use very vague terms here, but I know of two developers who both signed with a major platform publisher at the same time, for two games each. One dev asked for a very reasonable amount, and got it. The other dev asked for 20% more than the first dev, and one month less platform exclusivity—and they got that too. The first developer could've gotten a better deal, but they just didn't ask for anything more.

I heard from another platform publisher representative about how people complain of low rates in their publishing deals. My contact said to me, "Why don't they just ask? If they need another $50,000, that's not going to sour the deal." Sometimes people do ask, my contact said, and they quite often get what they asked for.

But the major point is that it didn't ruin the deal. In my experience, simply asking for more doesn't take a deal off the table, and if it does, that's probably not a publisher who really believes in your game in the first place. If asking for 5% more in revenue share is all it takes for them to think maybe this isn't such a good idea, maybe it isn't such a good idea for you to work with them, either.

So you're not just taking my word for it, I decided to ask fellow *Game Developer* columnist (and Spry Fox CEO) David Edery whether he's ever heard of or experienced a deal falling through simply because of asking for more. He said, "I've had that experience once. But it's worth noting that I was asking for a lot, as in, probably more than they had ever been asked for before. In general, it does not happen."

WHEN NEGOTIATING CONTRACTS, DON'T BE AFRAID TO ASK FOR MORE THAN THE BASELINE



He also agreed that a publisher who drops you simply because of an inquiry is probably not very interested in you. "Asking someone for another 5% should not cause them to walk away from the deal all by itself," he said. "After all, they can simply say 'no.' If they do walk away, what I would most likely assume is that: A) They have no respect for you, and/or B) They are hoping to intimidate you, and/or C) Like you said, they're not really interested in working with you anyway."

But of course, you can't be asking for things all the time. Best to do that all at once, and sign a contract that has everything you've asked for in one batch. "Asking for one thing, then another, then another, and dragging a negotiation out for months can definitely cause someone to walk away even if all your requests are reasonable," Edery added.

In my experience, it's best to find out what all the terms of the contract are up front, spend a day or two mulling over what works best for you, then make your requests all in one go. Knowing a bit about what the contract will look like in advance, simply by talking about things like amount of funding, how milestones are paid, revenue share percentages, marketing costs, how publisher risk is recouped, and exclusivity terms will help to make the process smoother once you're actually looking at a live contract document.

There certainly is a temptation to ask for more, after your demands seem to be easily met. If you asked for 5% more revenue share than is the norm, and the publisher immediately said yes, there's a tendency to feel like you should have asked for 7%, or 10% more. But pushing harder on the same issue makes you look a bit precious, and does not ingratiate you to your overlords. (If you try this consistently, please let me know your results.)

ON YOUR OWN As a development team, it feels like you've got more bargaining chips. As an independent contractor, though, it's often quite tough to know how much to charge. I've talked to a number of people who have gone independent after leaving a studio, or transitioned into the industry from another field, who simply have no idea what they should ask for. You don't want to go too low, because you need to eat. Obviously you want as much as you can get. But you don't know who else is bidding. What if someone else underbids you?

I've certainly had that happen to me, but I haven't regretted it missing out on those deals. Never sign a contract that's lower than what you really want. In the end, it's good to go into negotiations feeling like you're worth a lot of money. If you start sliding down the scale, people will realize you're worth whatever they feel like paying.

When I was doing more specific contract work, such as dialog writing, I've always asked for a bit more than they might expect. I use previous projects as a gauge, and try to slowly increase my rate; after all, with every project I gain more experience, so shouldn't I be worth more? And if ever the company asks me to do something that I feel is outside the scope of the existing contract, I ask for a new one that includes the new work, and additional compensation. The worst they can say is no, at which point I will not do the extra work. The old contract still holds, after all!

Now, it's probably not the best idea to try extensive negotiations with a salaried contract, unless you're a high level lead or exec who has a lot of bargaining power due to your level of experience. In a salaried situation, sometimes too much negotiation can make the potential employee seem like they're not a team player.

But overall, the feeling I've gotten is that asking for more doesn't hurt. It's never hurt me, and the developers I've talked to haven't been hurt by it either. So why not try to get a little more? See what you can get. They're the ones with the money; we're the ones with the content they need in order to make more money. When you think about it that way, we're really in a good position as developers. So go ahead: Ask for more!

***

Brandon Sheffield is director of Oakland, California–based Necrosoft Games, and editor emeritus of Game Developer magazine. He has worked on over a dozen titles, and is currently developing two small-team games for PlayStation Mobile. Follow him on Twitter via @necrosofty.

GDC Vault

UNLOCK A TROVE OF 8,000+ IN-DEPTH DESIGN, TECHNICAL AND INSPIRATIONAL TALKS AND SLIDES FROM GAME DEVELOPMENT INDUSTRY INFLUENCERS. ALL TAKEN FROM 20 YEARS OF THE WORLDWIDE GAME DEVELOPERS CONFERENCES.

STUDIO AND EDUCATION MEMBERSHIP PACKAGES AVAILABLE.

CONTACT GILLIAN CROWLEY AT
GILLIAN.CROWLEY@UBM.COM

GDCVAULT.COM

with Markus 'Notch' Persson
sson (Mojang)
A Prac l Guide to Building Browser Based
by Marc orain (Swrve)
GDC 2012
AI-driven Dynam Dialog through Fuzzy Patte
by Elan Ruskin (Va Corporation)
Programming
GDC 2012
Classic Gam
by Frederic

ATTENTION: GDC 2013 ALL-ACCESS PASS HOLDERS
YOUR MEMBERSHIP NOTIFICATION WILL BE SENT TO YOUR IN-BOX NO LATER THAN APRIL 5, 2013.

CHECK OUT THE ALL NEW GDC VAULT. NOW OFFERING:

- RECOMMENDED VIDEOS BY SESSION
- IMPROVED FILTERING BY CATEGORY AND EVENT
- CONTENT RELATED TAGS FOR EACH SESSION
- VIDEO CAROUSEL HIGHLIGHTS OF FEATURED SESSION
- UPGRADED VIDEO PLAYER EXPERIENCE

10%
DISCOUNT FOR ALL GAME DEVELOPER MAGAZINE READERS

SUBSCRIBE NOW ★
AT GDCVAULT.COM FOR YOUR DISCOUNT CODE.
OFFER EXPIRES APRIL 30, 2013.

UBM Tech

GROW IN THE GAME INDUSTRY

GAME CAREER GUIDE.COM

BROUGHT TO YOU BY GAMASUTRA

- Reference industry news and features
- Consult your digital counselor
- Play student games and join the forum

VISIT YOUR YEAR ROUND MENTOR AT GAMECAREERGUIDE.COM

GAME CAREER GUIDE

BROUGHT TO YOU BY gd

- Examine tutorials and exclusive features
- Check out the Annual Salary Survey
- Reference the premier Game School Directory

DOWNLOAD YOUR FREE DIGITAL COPY AT GAMECAREERGUIDE.COM

GAME CAREER SEMINAR

BROUGHT TO YOU BY GameDevelopers Conference

- Learn from the pros
- Attend deep-dive sessions with Q&A
- Connect with your game making peers

VISIT GDCONF.COM FOR INFO ABOUT THE NEXT SEMINAR AT GDC 2013

-- FOR PROFESSIONALS --

GAMASUTRA JOBS

- Search for active junior to senior level jobs from 100+ leading game companies
- Upload your resume and get in front of direct hiring managers
- Organize your career research with your personalized Job Seeker dashboard

VISIT GAMASUTRA.COM/JOBS TO ADVANCE YOUR GAME CAREER

UBM Tech

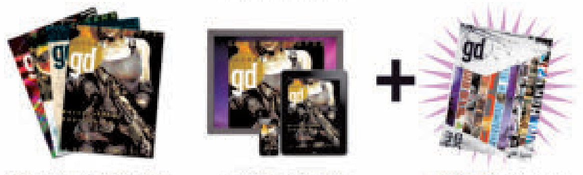
GAMEDEVELOPER MAGAZINE

THE BEST OF POSTMORTEMS, PRODUCT REVIEWS, AND STANDOUT COLUMNS!

GET THE PRINT+DIGITAL ACCESS BUNDLE FOR ONLY

$49.95 YEAR

INCLUDES:



PRINT SUBSCRIPTION

DIGITAL + GAME DEVELOPER APP

+ BONUS! BEST OF POSTMORTEMS PRINT ISSUE

PLUS GET DIGITAL ACCESS TO BACK ISSUES & EXCLUSIVE INTERACTIVE EXTRAS!

SUBSCRIBE TODAY! GDMAG.COM/SUBSCRIBE

ADVERTISER INDEX

COMPANY NAME	PAGE #
ACADEMY OF ART UNIVERSITY	057
ACTIVISION	025
ASOBO STUDIO	007
CCP GAMES	045
ENCHANT.JS INC	036
EPIC GAMES	019
FMOD BY FIRELIGHT TECHNOLOGIES	003
MASTERS OF DIGITAL MEDIA PROGRAM	061
MICROSOFT CORPORATION	C3
NVIDIA CORPORATION	061
PLAYSPAN INC	009
RAD GAME TOOLS	C4
SIMPLYGON	022
SOCIETY FOR THE ADVANCEMENT OF THE SCIENCE OF DIGITAL GAMES	049
TAYLOR & FRANCIS GROUP LLC	029
VANCOUVER FILM SCHOOL	055

IN ASSOCIATION WITH UBM TECH	PAGE #
GDC VAULT	068-069
GAMASUTRA.COM	052
GAME CAREER NETWORK	070
GAME DEVELOPER SUBSCRIPTIONS	071
GAME DEVELOPERS CONFERENCE 2013	C2
GDC NEXT	051

For more information visit www.jointhegamenetwork.com

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: $59.95; all other countries: $69.95 (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. Or else.

M. WASTELAND & M. UNDERLAND

COVER MY KICKSTARTER! KICKSTARTER PLEAS FOR MEDIA ATTENTION

ILLUSTRATION: JUAN RAMIREZ

THE COOL PR FIRM LETTER

Greetings [MAILMERGE:JOURNO_NAME],
How've you been? I wanted to let you in on a Kickstarter project that your readers simply must know about: KILL UNIT is back!

That's right—the arcade hit from 20 years ago is set for a dramatic revival as the game's creator, Robert Unit, returns to helm KILL UNIT REDUX!

"Remarkably, in the 20 years since the original KILL UNIT was a gigantic success, nobody else has made a game about shooting aliens," Unit said. "But, with the help of the fans, I aim to change that with KILL UNIT REDUX."

You'll want to mention our sweet Kickstarter rewards. Backers will get exclusive insight into the workings of Robert Unit's "game design mind" as he re-creates the original classic, as well as access to a backers-only forum where fans of the game can talk to each other about how excited they are!

The team is seeking to raise $5,000,000 over the next 10 days. We have a passel of hilarious video diaries (already shot and edited), endorsements from other famous game makers, and Promoted Tweets™ ready to fire off throughout the fundraising period—so get those reblogging buttons ready! Thanks in advance for your cooperation.

Have a wonderful week, [MAILMERGE:JOURNO_NAME].
Sincerely,
MegaSplash PR™

***

THE BIG DREAMER WITH A CHIP ON HIS SHOULDER

SUPPORT INDIES! I am an indie game developer and I have a Kickstarter and a Greenlight page up for my dream game. It's called MY DREAM GAME. I have written about some of my AMAZING ideas on my LiveJournal, but this has been IGNORED by the big sites.

The game is set on an uninhabited island with a mysterious past. The player must solve ingeniously designed puzzles while proceeding through a complex and nonclichéd storyline with over 30 characters and emotional depth. Jaw-dropping graphics and white-knuckled gameplay round out the package. It is somewhat influenced by FINAL FANTASY, but with other elements as well.

PLEASE SUPPORT INDIES by writing about this project. THE GAME WILL NOT BE MADE unless the word gets out and the project gets more fans. The time is now for gamers to rise up and DEMAND the games they want by backing this project!!!

***

THE SUBSTITUTE FOR VENTURE CAPITAL

Greetings,
Are you content to live in a world where you cannot experience video games with smell? I didn't think so. SensePlay LLC has just launched a Kickstarter for its highly disruptive product, the ScentPlay™, and now we need *your* help to make this revolutionary device a reality.

The heart of the ScentPlay™ experience is the ScentInjector. This accessory uses high-quality silicone clips to sit comfortably inside the user's nose for hours of olfactory gaming. And, for backers at the $375 level, we will create custom-fitted ScentInjector devices with a custom plastic mold from players' nostrils.

Working prototypes of the ScentPlay™ system already exist as plans on paper, and some of the industry's top aroma designers are already dreaming up use-case scenarios! We plan on shipping the first units in April.

Thank you,
The SensePlay LLC Team

***

THE WEIRD NEW-AGEY THING

Dear Sir or Madam:
What is reality?
That is the question that the cybernetic consciousness-expanding game, The Reality Project, will answer. With a fully featured MMO engine dubbed Universe 2.0, The Reality Project will take players on a path of holistic self-discovery. With your generous support, we will reach the $20,000 needed to make a digital awakening occur in the entire human race.

Key features include:
- Development of expanded cyber-consciousness
- Digital synthesis of all existing religious and spiritual traditions
- Customizable avatars

The amount of $20,000 will pay for artists, designers, programmers, musicians, and server-hosting costs to build and operate this groundbreaking MMO experience. Please join us on this journey into the infinite.
Sincerely,
The Reality Project

***

THE TREND-CHASER

Available for Comment: Kickstarter Expert Chris Clinger

In this era of Kickstarter projects by the dozen, Kickstarter Expert Chris Clinger is available for comment NOW on a wide range of issues regarding Kickstarter projects. Chris has advised dozens of Kickstarter projects and backers, both through blog comments and via tweet.

Please attribute quotes to "Chris Clinger, The Kickstarter Coach" or "Kickstarter expert, Chris Clinger, who can be found at www.kickstarterexpert-tips.biz". Chris Clinger's forthcoming book on successful Kickstarter projects, From Kick to Start, will be available once its Kickstarter funding goal is reached. ad

***

Matthew Wasteland writes about games and game development on his blog, Magical Wasteland (www.magicalwasteland.com). Email him at mwasteland@gdmag.com. Magnus Underland writes about games and other topics at www.above49.ca. Email him at magnus.underland@gmail.com.

Microsoft is changing the face of entertainment.

Come and learn how at GDC 2013.

- Microsoft Developer Day
  Tuesday, March 26
- Microsoft Lobby Bar & Product Showcase
  Monday, March 25 – Friday, March 29
- Microsoft Sessions
  Wednesday, March 27 – Friday, March 29
- Xbox Recruiting Booth
  Wednesday, March 27 – Friday, March 29

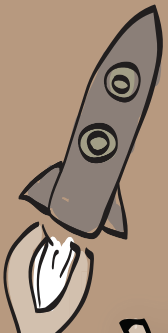
Windows Phone 8 Direct3D
Natural User Interface
Xbox LIVE Indie Games Visual Studio MSDN
Game Developer Network SmartGlass
IntelliSense Windows Phone PIX
ATG Visual C++ Xbox LIVE
Windows SDK
Xbox LIVE Marketplace XAudio2
Windows Phone SDK
Xbox 360
DirectX 11
Dream.Build.Play
Xbox LIVE Arcade
Channel 9 Kinect
Windows 8 Xbox 360 XDK
Windows Store

Microsoft

yes, it's NEW

THE NEWEST RAD TOOL IS

NOW SHIPPING

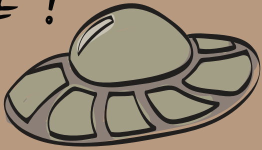


OUT OF THIS WORLD

TELEMETRY

42

TELEMETRY is a programming library and set of tools for instrumenting, profiling, tuning and visualizing application *PERFORMANCE*!



VISUALIZE real-time game performance,

see **WHEN** things happen — not merely **WHAT** happened!

PROBE the hierarchical display —

see thread interactions, context switches, and mutex locking!

THIS ISN'T JUST rocket science, THIS IS rad!

RAD© GAME TOOLS

www.radgametools.com/telemetry
(425) 893-4300