







## NetherRealm Brings the Fight to DC Super Heroes with Unreal Engine 3

NetherRealm Studios has been working with Unreal Engine 3 (UE3) technology since it was part of Midway Entertainment. The Chicago-based studio has focused on fighting games, establishing itself with the bestselling *Mortal Kombat* franchise.

After exploring a hit crossover a few years ago with *Mortal Kombat vs. DC Universe*, the Warner Bros. Interactive Entertainment-owned developer has returned to comic book lore with the original fighter, *Injustice: Gods Among Us*. The game, which features iconic super heroes including the Dark Knight, the Man of Steel, Wonder Woman and Catwoman, was designed using Unreal Engine technology.

"We're always making advancements," said Ed Boon, creative director, NetherRealm Studios. "This game has much more dynamic lights in terms of lighting the environments and the characters. It showcases a lot more detail than what we had going in the last *Mortal Kombat* game."

Boon said his team has made optimizations that particularly cater to the fighting genre, since games such as *Mortal Kombat* and *Injustice* feature large characters that rely

on heavy motion capture animation.

"We have a faster rendering engine, so we can put more on the screen running at 60 frames per second," he explained. "We have a great memory manager that lets us have multiple arenas for any fight, so you can knock someone from one arena into the next. *Injustice* is reaping the benefits from all of these steps forward that we took since *Mortal Kombat*."

In addition, Boon said that NetherRealm has crafted a scripting system for implementing animations and moves specifically for its fighting games. "This is proprietary technology that we've added to Epic's engine. It's a more code-based component than a visual interface, so it lets us iterate, speeding up animation, slowing down changing collisions and all that. And it allows us to do all of that with a very fast turnaround."

**"With *Injustice*, we can put characters in and start animating right off the bat."**

Boon said that the team's experience with Unreal Engine technology over all of these years has made it easier to add these types of modifications.

"UE3 gives us a great way to start off running," said Boon. "With *Injustice*, we can put characters in and start animating right off the bat. We're improving the systems with each iteration."

UE3 has allowed the developer to focus on innovating the fighter genre. While the last *Mortal Kombat* game was the best-selling in franchise history, the studio sought to make something completely original this time around.

"Our main goal was to do another fighting game that was completely different than *Mortal Kombat*," said Boon. "When we joined the Warner Bros. family, we looked at their wealth of intellectual properties and the DC heroes seemed like a great fit for it. We had actually worked with them before on *Mortal Kombat vs. DC Universe* and we wanted to do something that was focused on celebrating the whole exaggerated world of super heroes.

"So instead of doing things like swinging a sword, you swing a car. Instead of knocking somebody down, you throw them through a building. We really wanted to exaggerate all these events and create this hyper-realistic battle of the gods' world and that's what *Injustice* is."

*Injustice: Gods Among Us* is one of 2013's highly anticipated UE3-powered blockbuster releases.

Thanks to Ed Boon at NetherRealm for speaking with freelance reporter John Gaudiosi for this story.

### UPCOMING EPIC ATTENDED EVENTS

**D.I.C.E Summit**  
Las Vegas, NV  
February 5-8, 2013

**Cloud Gaming Europe**  
London, UK  
February 21-22, 2013



Please email [licensing@epicgames.com](mailto:licensing@epicgames.com) for appointments



GAME DEVELOPER MAGAZINE

Z V A G W R E D O T E A E O F

## CONTENTS.1212

VOLUME 19 NUMBER 12

### post mortem

**26 SOUND SHAPES**  
Making a game built around making music (and having fun doing it) is incredibly challenging, but Queasy Games's *SOUND SHAPES* made it look easy. Building *SOUND SHAPES* was far from easy, though—underneath its elegant design lies dozens of discarded prototypes and small-studio growing pains.  
*By Mathew Kumar*

### features

**6 YEAR IN REVIEW**  
It was nice knowing you, 2012! *Game Developer* rings out the year by asking a selection of accomplished developers about the new stuff they love, the old stuff they're glad to get rid of, and more.  
*By Staff*

**13 THE ART OF WAR**  
Military shooter creators routinely consult military experts, but designers of games featuring swordplay or other forms of melee weapon-based combat rarely go to the medieval weapons experts. Medieval martial arts specialist John Clements and animator Eben Bradstreet show you how understanding realistic sword combat can make your game better (and easier to animate).  
*By John Clements and Eben Bradstreet*

**21 BEYOND SHADOWS OF DOUBT**  
Shadows in 3D games don't always play nicely with other ways artists make their games look good, especially when you look at how the shadow falls on the model that's casting it. 3D experts Andrew Woo and Pierre Poulin explain four workarounds for common self-shadowing issues.  
*By Andrew Woo and Pierre Poulin*

### departments

- 2 GAMEPLAN** *By Patrick Miller* [EDITORIAL]  
2012 In Games
- 4 HEADS UP DISPLAY** *By Staff* [NEWS]  
Front Line Award finalists and a homemade 8-bit assembly compiler
- 33 TOOLBOX** *By Mike de la Flor* [REVIEW]  
Autodesk Mudbox 2013
- 36 THE INNER PRODUCT** *By Eddie Cameron* [PROGRAMMING]  
Recording Unity
- 41 PIXEL PUSHER** *By Steve Theodore* [ART]  
Cheek by Jowl
- 44 DESIGN OF THE TIMES** *By VandenBerghe* [DESIGN]  
The Four Fs of Game Design
- 47 AURAL FIXATION** *By Damian Kastbauer* [SOUND]  
Death of an Audio Engine
- 49 THE BUSINESS** *By Kim Pallister* [BUSINESS]  
The 2012 Changelog
- 50 INSERT CREDIT** *By Brandon Sheffield* [EDITORIAL]  
The Avant-Game
- 52 GDC NEWS** *By Staff* [NEWS]  
IGF 2013 hits record number of entries, and GDC 2013 opens registration
- 53 GOOD JOB** *By Alexandra Hall* [CAREER]  
Going indie in Mexico City, new studios, and who went where
- 54 EDUCATED PLAY** *By Alexandra Hall* [EDUCATION]  
NEVERMIND
- 56 ARRESTED DEVELOPMENT** *By M. Wasteland and M. Underland* [HUMOR]  
Stack Trace and the Death of A.A.A. Development, Part I





# 2012 IN GAMES

THE PAST YEAR OF VIDEO GAMES IN THREE STORIES

Due to the whims of the print-publishing biz, you'll be reading this with a holiday-appropriate seasonal drink in hand, safely insulated from December weather, but I'm writing this in early November. It's a sunny 85 degrees outside in San Francisco, the U.S. elections haven't happened yet, and people are just starting to tweet about *Wreck-It Ralph*—certainly not ideal conditions for writing a year-end wrap-up column.

So instead of casting about for wise words on the game industry's past year (Crowdfunding! Studio closures! Mobile games!), I'll leave the year-end stuff to our esteemed contributors and colleagues in our Year In Review feature, and instead list three moments that defined 2012 in games for me.

## EXECUTION AT E3

» Personally, I've never been particularly worried about the effects of violent video games on younger players, or people in general—in my opinion, anything over-the-top or tasteless in games tends to be present in far larger amounts in the rest of popular culture. But if there was ever a moment where I stopped to think about it, it would have to be during the Sony press conference at this year's E3, when a stadium full of grown men and women watched the virtual point-blank shotgun execution of a man pleading for his life during the *THE LAST OF US* demo and immediately erupted in applause.

On one hand, that moment was completely understandable; it was a pitch-perfect moment for a demo, and if I had been sitting in a friend's living room watching that same sequence, I probably would have reacted with enthusiasm. Hearing thousands of people cheering at that moment,

on the other hand, made me feel a little sick—and considering that demo capped off a long reel of ultraviolence, a little bit worried for the state of triple-A games.

In retrospect, I remember that moment less for my cynicism in the moment, and more for the chills that ran down my spine. Emotionally evocative moments in and around games are to be treasured, I think—even if that emotion is disgust.

## GAME OF PHONES

» I had the good luck to serve as a judge on the panel for Tokyo Game Show's 2012 Sense of Wonder Night, which is the show's accompanying independent game showcase event. *MEMORY OF A BROKEN DIMENSION* walked us through haunting vistas of an alien world; *GRANDMASTER* looked into the life of homeless people in Ukraine; I could go on and on, but I'll stick to talking about one of my favorite games there.

*TAISO*, by Japanese dev team ZacoZaco, is very simple. It's an iPhone game where you control a gymnast—"Taiso" means "gymnastics" in Japanese—and you launch that gymnast a certain distance by using your iPhone's accelerometer. Basically, it's an iPhone game about throwing your iPhone as high in the air as possible.

Naturally, *TAISO* is full of emotionally evocative moments; the thrill of throwing your iPhone in the air, the relief when you catch it intact, the loss and regret when you don't. The fact that three independent devs—two of which spent the development period going to girlie bars and all-you-can-eat barbecue joints, according to the presentation—can whip up a game in Unity that hits us in the feels harder than any major 2012 title gives me lots of hope for 2013.

## WHERE EVERYONE KNOWS YOUR NAME

» I attended GDC Online this year, just in time for us to say goodbye to Austin (see GDC News, page 52). On our second night there, I went to a "barcade" called Recess with Gamasutra editors Kris Graft and Frank Cifaldi. We tried to play *TEENAGE MUTANT NINJA TURTLES: THE ARCADE GAME*, but it just wasn't happening; the beer was good, but the broken controls kind of killed it, and arcade beat-em-ups aren't nearly as fun when they're on free-play mode (a lesson for free-to-play developers, perhaps!).

The next night, I joined a few dedicated arcade-goers in a nighttime jaunt to ArcadeUFO, Austin's go-to arcade with genuine Japanese-style head-to-head fighting game setups, POP'N MUSIC cabinets, and even a *PAC-MAN BATTLE ROYALE* four-player setup. The money these days might be all up in the massively multiplayer crowdfunded casualcore mobile/social/free-to-play markets, but there is simply nothing like the experience of walking into a room far from home and throwing down in some *MARVEL VS. CAPCOM* until you've made some new friends.

## ON TO THE NEXT ONE

» I have no doubt that 2013 will bring us a new truckload of buzzwords that promise even-more-photorealistic graphics tech, more DAUs/MAUs/ARPU's/ARPPUs, and consoles with bigger GBs than last year's—and it'll be our collective job to sort out the wheat from the chaff. But when I remember 2012, I'm not going to remember a buzzword; I'm going to remember the games I played, the people I played them with, and how they (the games *and* the people) made me feel.

—Patrick Miller  
@pattheflip

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

## SUBSCRIPTION SERVICES

FOR INFORMATION, ORDER QUESTIONS, AND ADDRESS CHANGES

t: 800.250.2429 f: 847.763.9606  
e: [gamedeveloper@halldata.com](mailto:gamedeveloper@halldata.com)  
[www.gdmag.com/contactus](http://www.gdmag.com/contactus)

## EDITORIAL

### PUBLISHER

Simon Carless e: [scarless@gdmag.com](mailto:scarless@gdmag.com)

### EDITOR

Patrick Miller e: [pmiller@gdmag.com](mailto:pmiller@gdmag.com)

### EDITOR EMERITUS

Brandon Sheffield e: [bsheffield@gdmag.com](mailto:bsheffield@gdmag.com)

### MANAGER, PRODUCTION

Dan Mallory e: [dmallory@gdmag.com](mailto:dmallory@gdmag.com)

### ART DIRECTOR

Joseph Mitch e: [jmitch@gdmag.com](mailto:jmitch@gdmag.com)

### CONTRIBUTING WRITERS

John Clements, Eben Bradstreet, Mathew Kumar, Mike De La Flor, Eddie Cameron, Steve Theodore, Jason Vandenberghe, Damian Kastbauer, Kim Pallister, Matthew Wasteland, Magnus Underland

## ADVISORY BOARD

Mick West Independent  
Brad Bulkley Microsoft  
Clinton Keith Independent  
Brenda Brathwaite Loot Drop  
Bijan Forutanpour Sony Online Entertainment  
Mark DeLoura THQ  
Carey Chico Globex Studios  
Mike Acton Insomniac

## ADVERTISING SALES

### VICE PRESIDENT, SALES

Aaron Murawski e: [amurawski@ubm.com](mailto:amurawski@ubm.com)  
t: 415.947.6227

### MEDIA ACCOUNT MANAGER

Jennifer Sulik e: [jennifersulik@ubm.com](mailto:jennifersulik@ubm.com)  
t: 415.947.6227

### GLOBAL ACCOUNT MANAGER, RECRUITMENT

Gina Gross e: [gina.gross@ubm.com](mailto:gina.gross@ubm.com)  
t: 415.947.6241

### GLOBAL ACCOUNT MANAGER, EDUCATION

Rafael Vallin e: [rafael.vallin@ubm.com](mailto:rafael.vallin@ubm.com)  
t: 415.947.6223

## ADVERTISING PRODUCTION

### PRODUCTION MANAGER

Pete C. Scibilia e: [peter.scibilia@ubm.com](mailto:peter.scibilia@ubm.com)  
t: 516-562-5134

## REPRINTS

### WRIGHT'S MEDIA

Jason Pampell e: [jpampell@wrightsmedia.com](mailto:jpampell@wrightsmedia.com)  
t: 877-652-5295

## AUDIENCE DEVELOPMENT

### AUDIENCE DEVELOPMENT MANAGER

Nancy Grant e: [nancy.grant@ubm.com](mailto:nancy.grant@ubm.com)

### LIST RENTAL

Peter Candito  
Specialist Marketing Services  
t: 631-787-3008 x 3020  
e: [petercan@SMS-Inc.com](mailto:petercan@SMS-Inc.com)  
[ubm.sms-inc.com](http://ubm.sms-inc.com)





# GROW IN THE GAME INDUSTRY



- Reference industry news and features
  - Consult your digital counselor
  - Play student games and join the forum
- VISIT YOUR YEAR ROUND MENTOR AT [GAMECAREERGUIDE.COM](http://GAMECAREERGUIDE.COM)



- 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](http://GAMECAREERGUIDE.COM)



- Learn from the pros
  - Attend deep-dive sessions with Q&A
  - Connect with your game making peers
- VISIT [GOCONF.COM](http://GOCONF.COM) FOR INFO ABOUT THE NEXT SEMINAR AT GDC 2013

-- FOR PROFESSIONALS --



- 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](http://GAMASUTRA.COM/JOBS) TO ADVANCE YOUR GAME CAREER





## 8-BIT WIZ

SOFTWARE ENGINEER ANDREW CROWELL ROLLS HIS OWN 8-BIT ASSEMBLY COMPILER



I'VE BEEN WORKING ON WIZ, A HIGH-LEVEL ASSEMBLY LANGUAGE COMPILER FOR MAKING HOMEBREW 8-BIT CONSOLE GAMES. I NAMED IT "WIZ" BECAUSE I FIGURE YOU HAVE TO BE SOME SORT OF WIZARD TO ACTUALLY FEEL DEDICATED ENOUGH TO WRITE CODE FOR OLD CONSOLES IN THIS DAY AND AGE. IN A WAY, IT FEELS LIKE YOU'RE CHANNELING SOME SORT OF ANCIENT ART OR SOMETHING.

I wrote the compiler from scratch using the D programming language, which is kind of obscure, but I wanted an excuse to play around and learn a new language. It has a few language/library features, like garbage collection, compile-time mixins, and the like, that made it sort of appealing.

So far, Wiz targets two platforms: the Game Boy's GBZ80 instruction set, and the 6502, which includes the NES, Apple, C64, Atari, Apple II, and others. Instead of using weird assembly mnemonics like the "lda player.x / clc / adc 5 / sta player.x" of most assemblers, you can simply write "a = [player.x] + 5; [player.x] = a" or as a convenience, "[player.x] = [player.x] + 5 via a".

At the moment, Wiz has "if" statements, loops, and function blocks that cut down the number of go-tos you need to write (and save you from the horrible bug of forgetting to return at the end of a subroutine). It also has features for in-lining function calls and unrolling loops, cutting down on the amount of visible repetition in your code when it's actually faster to mash everything.

Essentially, Wiz can do most things another assembler can do, but Wiz is written



GRAND THEFTENDO.

to be a little less daunting; you don't have to remember 5,000 mnemonics, and the syntax is slightly more forgiving and easier to grasp than most assemblers. You don't have

to worry about one command per line when it makes sense to have multiple bunched together. You don't have to do weird things like indent something eight spaces for it to be recognized as a directive. Wiz is intended to make things a little easier, while still giving you fairly explicit control over the memory layout of your program, which registers to use, and all that.

Also, for those wondering, Wiz is the fourth iteration of an assembly language compiler I've been trying to write. For most of them, I made another language called Nel, which was NES-only, and it had a really bizarre way of writing programs, like "a: get @player.x, add #5, put @player.x"—you'd be getting things into registers, and then putting them somewhere when you're done. I was able to make some neat things with that, but I decided that was a little too funky. So I took the good, threw out the crap, and am rewriting a lot of the language. This time I'm pretty happy, and I'm adding features as I go.

It's all still kind of getting there, but my next goals are to actually write the manual for it and make more demos, and possibly tutorials to show it off. It was written for





my personal use, but I imagine that anyone else who wants to write a homebrew could benefit. The only similar project that I'm aware of is NESHLA by Brian Provinciano, who was working on an NES game called GRAND THEFTENDO, the precursor to his recent indie game RETRO CITY RAMPAGE.

The platforms I'm primarily interested in myself are the NES and Game Boy. So far I've made a few demos, and it seems to hold up to the job. There are still bugs that I'm uncovering along the way, and it's a bit slow at times, but it's pretty cool to see things in action in an emulator. Because assembly is so low level, even with these tools that make it a bit more convenient, it's possible to run into corners. Since you have direct access to the hardware registers in your game and there isn't anything to supervise you when you screw something up badly and automatically crash out, you can make some pretty hilarious and frightening glitches. One Game Boy emulator in particular [BGB] has a few nice features that you can turn on to breakpoint the game, in case you try to do really bad things with uninitialized memory, or mess with video hardware while the screen is busy using it.

Eventually I'll get a homebrew dev cart and be able to test on the real hardware! In time, I suppose. I'm primarily doing this for a fairly short-winded Game Boy Color dungeon-crawler RPG that my friend and I are collaborating on. It'll be in a Dragon Warrior/Earthbound style, and will hopefully push the hardware graphics a little bit. In addition to my Wiz assembler, there are other tools that we need to build to actually embed tiles, sprites, and other assets into the game—and these need to have data formats that can work on the specific hardware they're targeting. They need to be fast and memory conscious eventually, but "good enough for now" will do in the short term. So far, I've been winging it for my RPG, but maybe later I'll make those more multipurpose and reusable too.

The source is here: <https://github.com/Bananattack/wiz>. There is no official release yet, but that's in the works, once I get it a bit more polished and documented.

—Andrew G. Crowell

////////// GAME DEVELOPER MAGAZINE IS PROUD TO ANNOUNCE THE FINALISTS FOR THE FRONT LINE AWARDS, OUR ANNUAL CELEBRATION OF THE BEST TOOLS FOR GAME DEVELOPMENT. ¶ NEW TO THIS YEAR'S FRONT LINE AWARDS IS THE INCLUSION OF A SECTION FOR THE BEST FREE TOOLS, NOT LIMITED TO ANY SPECIFIC CATEGORY. PUBLIC NOMINATIONS WERE OPEN FROM OCTOBER 4–15, 2012, AND ONLY FIVE FINALISTS IN EACH CATEGORY WERE CHOSEN. ¶ IN ADDITION, GAME DEVELOPER HAS INDUCTED UNITY 3.5 INTO THE FRONT LINE AWARDS HALL OF FAME FOR ITS POWER, FLEXIBILITY, RELATIVE EASE OF USE, AND SIGNIFICANT IMPACT ON THE GAME INDUSTRY OVER THE LAST YEAR. PREVIOUS HALL OF FAME WINNERS INCLUDE ADOBE FLASH, UNREAL ENGINE, AND XNA GAME STUDIO. (AS SUCH, UNITY WON'T BE REPRESENTED IN THE BEST ENGINE CATEGORY FOR THIS YEAR ONLY, AS WAS TRUE FOR PREVIOUS HALL OF FAME HONOREES.)



## ART

### 3ds Max 2013

(AUTODESK)

### Maya 2013 SP1

(AUTODESK)

### Modo 601

(LUXOLOGY)

### Photoshop CS6

(ADOBE)

### ZBrush 4R4

(PIXOLOGIC)

## AUDIO

### Miles Sound System 9

(RAD GAME TOOLS)

### Pro Tools 10

(AVID TECHNOLOGY)

### REAPER 4

(COCKOS INC.)

### Reason 6.5.1

(PROPELLERHEAD SOFTWARE)

### Wwise 2012.1

(AUDIOKINETIC)

## GAME ENGINE

### CryENGINE 3

(CRYTEK)

### GameMaker: Studio

(YOYO GAMES)

### Havok Vision Engine

(HAVOK)

### Torque 3D

(GARAGEGAMES)

### Unreal Engine 3

(EPIC GAMES)

## MIDDLEWARE

### Havok Physics

(HAVOK)

### SpeedTree for Games v6.2

(INTERACTIVE DATA VISUALIZATION, INC.)

### Tableau Desktop

(TABLEAU SOFTWARE)

### Umbra 3

(UMBRA SOFTWARE)

### XaitCONTROL 3.7

(XAITMENT)

## FREE TOOLS

### Blender

(BLENDER FOUNDATION - OPEN SOURCE)

### Box2D

(ERIN CATTO - OPEN SOURCE)

### Flixel

(ADAM SALTSMAN - OPEN SOURCE)

### GIMP 2.8

(THE GIMP TEAM - OPEN SOURCE)

### Ogre3D

(TORUS KNOT SOFTWARE - OPEN SOURCE)

## PROGRAMMING/ PRODUCTION

### Bugzilla

(MOZILLA FOUNDATION - OPEN SOURCE)

### CruiseControl

(CRUISECONTROL DEV TEAM - OPEN SOURCE)

### Jenkins

(OPEN SOURCE)

### JIRA

(ATLASSIAN)

### Perforce

(PERFORCE SOFTWARE)

SOFTWARE  
FOR THE  
GAMES  
INDUSTRY

# 2012 IN REVIEW

## DEVELOPERS WEIGH IN ON THE PAST YEAR

MARKUS

### PERSSON

(MOJANG)



**OUT WITH THE OLD:** Physical distribution. Finally consoles are starting to catch up to PC and mobile and offer most of their game content digitally. Soon we'll be able to fully cut out this unnecessary middleman between the game developer and game player. The only downside to this is that game libraries in basements will look a lot less impressive when they're just posters of screenshots of your game collections.

**IN WITH THE NEW:** Personalities. With the rise of crowdfunding and indie games, there's been a stronger focus on the people behind the games again. This is a wonderful thing for me as a gamer, as I get to know whose vision I'm playing through and maybe understand the creators through their work. As a game developer, it also helps to get new heroes to look up to, which is something I miss dearly from the good old days.

**SHOUT-OUTS:** I could go on forever on this; it's been a great year! Brian managed to release

RETRO CITY RAMPAGE, and it is glorious; Terry made a game even more frustrating than VVVVVV, and it is wonderful; Firaxis somehow managed to make a worthy remake of X-COM, and it's killing all my free time; and Valve managed to go another year without giving us Gordon.

**ITOLD YOU SO:** Microsoft's choice of setting up their own store was a very predictable one, and it's a sad one, for a wide range of reasons.

**ONE-SENTENCE REVIEW:** You know it's been a good year when you end up with a large stack of must-play games you haven't had time to play, and 2012 has been very good so far.

DAVID

### HELGASON

(UNITY)



**OUT WITH THE OLD:** Old things don't go away, but rather take a backseat to the new shiny things. It's not like the console industry died in 2012, and Zynga-style social games are still being played.

**IN WITH THE NEW:** What I had feared (and predicted) was that there would be a strong

consolidation on mobile, with a few leaders emerging to dominate the business. This didn't happen, and instead the biggest hits are still being created by new and/or small studios. Remember that even Rovio was a small company until after ANGRY BIRDS, and a tiny studio like Boss Alien created CSR RACING with Unity, which is probably going to make between \$50M and \$100M (!) in its first year. And there are many, many examples of this.

**SHOUT-OUTS:** Madfinger's DEAD TRIGGER: an insanely cool triple-A zombie-shooter on your iPad. InXile's WASTELAND 2: badass indie development by rock-star developers doing crowdsourcing and running an open-development style. Defiant's SKI SAFARI: infinitely addictive and polished game, which started as a Unity-based Flash game, and which I still play daily on my iPhone after nearly a year. ENDLESS SPACE: a visually impressive Unity-based 4X space-strategy game, created in a radically open-development style. Oh, and amazing UI work, too!

**ITOLD YOU SO:** How the mobile game industry "grew up." Last year, it became evident that small teams could have great successes in terms of ROI, and make a few million dollars with a great game. What became obvious this year

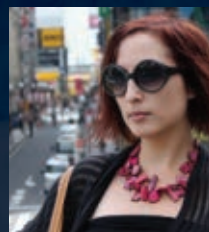
is that similar or only slightly larger teams now make mobile hits whose revenues come into the range of console games, but with massively better ROIs. That's a big deal and is leading all of the big studios to jump in with full force (using Unity, of course).

**ONE-SENTENCE REVIEW:** We finally stepped into a new era full force, where small studios are going to dominate, and where, soon, more players, games, and revenue will be on mobile devices than anywhere else.

MARE

### SHEPPARD

(METANET SOFTWARE)



**OUT WITH THE OLD:** Prefacing the names of apps, software, books, interior goods, and basically everything with "i," à la "iPad," in a now-awkward attempt to be clever. Oh wait, we haven't stopped doing that yet, have we? Sigh. Maybe next year.

**IN WITH THE NEW:** There have been a lot of efforts to encourage and welcome new creative people into the world of making games lately, which is great! It's something I personally

think is an important part of the vibrant future of this industry—I can't wait to play these newbies' second and third and fourth games. Everybody can make games, and it's important that interested people give it a try, but it's equally important that they stick with it and learn and evolve—as we each refine our craft, that's how our best ideas and concepts develop, so that potential is very exciting to me. It means better games for all of us!

**SHOUT-OUTS:** Queasy Games with SOUND SHAPES. It was an immensely challenging project for the team, and I was impressed at the way they persevered through the difficulties and focused on their intent to create something beautiful, and then I was doubly impressed at how well they pulled that off. SOUND SHAPES is gorgeous to look at and easy on the ears, but the best part is how it inspires players to become musicians themselves. Very cool.

**ITOLD YOU SO:** We at Metanet have been trying to finish a project that we've been working on for...geez, it feels like forever. It was supposed to be finished in September, but the deadline slipped again. I had a hunch. Seriously though, December. It's happening.

**ONE-SENTENCE REVIEW:** There were so many updates in the last year



# EW AR IN GAMES

From crowdfunding to studio shutdowns, 2012 has been a big year for the game industry. We've seen indie studios surge to the forefront on new and established platforms, new business models and dev tools lower the barrier to entry, and all kinds of major changes that will undoubtedly be felt for years to come. *Game Developer* polled accomplished industry veterans and indies alike for their respective takes on what we're glad to be done with, looking forward to working with, year-end shout-outs, and more.

to projects I'm eagerly anticipating, especially on the indie side!

CAREY

CHICO

(GDMAG ADVISORY BOARD)



#### OUT WITH THE OLD:

Gamification: I think we've discovered that this isn't really a game and this word is way too buzzy to convey the next generation of gaming experiences tied with social applications. Ultimately, gamification isn't creating a game—it's a way to fit a fun experience philosophy into a tiny crack of the social-application universe.

#### IN WITH THE NEW:

Windows 8! I think what they are doing is sublime. They are trying to get the world to transition out of desktop PCs and into something more intimate. I think the idea that the PC is dying is a misdirection that the mobile/tablet world uses as a rally call. In truth, the PC is transforming into a new experience in the same way the first Windows trumped DOS. I get it.

#### SHOUT-OUTS:

HAWKEN. I think Meteor Entertainment showed a guerilla product at the

right moment at the right time and are capitalizing on the fury surrounding their upcoming launch. They are an example of a dev start-up doing it right. They are also coming out at the height of the session-based gaming craze, most clearly driven by the success of *WORLD OF TANKS*.

#### ITOLD YOU SO:

I saw 38 Studios's closure coming a while ago. From when they first started working on *COPERNICUS*, to the ensuing survival strategies of buying Big Huge Games to generate early revenue, this company showed struggle from the beginning. Also, hardcore games on Facebook—I knew this was the next step in the evolution of social games.

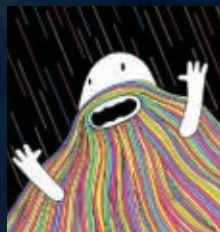
#### ONE-SENTENCE REVIEW:

Stop copying *ANGRY BIRDS*!

DAMIAN

KASTBAUER

(GDMAG AUDIO COLUMNIST)



#### OUT WITH THE OLD:

Proprietary audio engines and tools in game audio are dying off like the dinosaurs. The closing of Radical Entertainment saw the loss of its 10-years-in-

the-making audio toolset. When a studio closes and takes its technology to the grave, the community feels its loss. In the wake of this, the development of proprietary tools seems to have slowed in comparison to publicly available audio tools. Where once proprietary tools may have been the bleeding edge, the atrophy and evolution of game audio has superseded most general-use in-house tools in favor of audio middleware.

#### IN WITH THE NEW:

Middleware—everyone's using it! Powering up your development with existing libraries is nothing new, but the widespread adoption of middleware has lowered game development's barriers to entry. It's not just a trend among new developers either; the power of middleware in games has firmly taken root and enabled the creation of wildly diverse game types through a combination of accessibility and power.

#### SHOUT-OUTS:

Nu retro and sublime beatitude. The audio work in games like *FEZ* and *DUSTFORCE* tapped the nostalgia vein while simultaneously projecting an aural vision of the future. Meanwhile, the *BOTANICULA* soundtrack and audio aesthetic aligned perfectly with its gameplay's sense of childlike glee. The sound for *JOURNEY* also shone brightly for its understated

and minimal reflection of a truly inspiring experience.

#### ITOLD YOU SO:

*DYAD*, *SOUND SHAPES*, and *PIXELJUNK 4AM* subversively pushed the confines of interactive audio into real-time music creation. The winning combination of pure joy and the mastery of craft made these games the fruition of music games in the post-*ROCK BAND* landscape. Expressive musical instruments in their own right? You betcha!

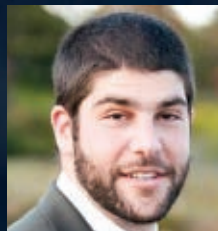
#### ONE-SENTENCE REVIEW:

Developers seem comfortable with the tools at their disposal and confident in their ideas of the kind of experiences they want to create.

DAVID

EDERY

(SPRY FOX)



#### OUT WITH THE OLD:

It seems that in the past year, a big number of independent game developers have switched focus from downloadable console games to iOS, Android, and Steam. The general perception seems to be that it's just too expensive, risky, and difficult to focus on consoles; that's a pretty

big shift in thinking from just two years ago for many people. These kinds of things don't make me happy or sad; they just are what they are. Platforms are cyclical things; at some point consoles will become attractive to indies again—that is, assuming consoles as we know them today actually survive the transition this industry is currently going through!

#### IN WITH THE NEW:

Kickstarter! While I don't think it will work for the majority of indies, I'm really excited to see that for at least a subsegment of highly accomplished long-time game developers, it has been a brilliant way to raise cash to develop games that publishers simply aren't interested in funding (or that they would fund, but demand a disproportionate return in exchange for their funding). Excited to see where this all leads.

#### SHOUT-OUTS:

I've put a disproportionate amount of my personal game-playing time into Robot Entertainment's games this year. *HERO ACADEMY* and *ORCS MUST DIE 2* have both kept me up for way more late nights than I'm comfortable quantifying! I'm impressed by how deftly the guys at Robot have tackled different platforms (PC, mobile) without much help and without simply copying a proven formula, unlike so many other developers.



**✿ ITOLD YOU SO:** Everything coming to mind is negative, and I'd rather not rub salt in anybody's wound. Hey, I knew MINECRAFT was going to kill it on XBLA. Of course, so did everybody else, so I can't exactly brag about this. :-)

**🕒 ONE-SENTENCE REVIEW:** Back to hunting for the next great game platform.

**ERIN ROBINSON**  
(IVY GAMES)



**🕒 OUT WITH THE OLD:** Although it hasn't been confirmed by Microsoft yet, I share the concern that Windows 8 will be a closed platform. There are fears that releasing a game on Windows will be like developing apps for an Apple device, vetting process and all. That's bad for indie games for obvious reasons, but it's been interesting to see the bigger names—Gabe Newell from Valve, Rob Pardo from Blizzard—weigh in against the closed-system approach as well. We've really taken it for granted that anyone can release a game on Windows, and I hope we don't lose that.

**🌱 IN WITH THE NEW:** I'd say this is the year that crowdsourcing really came into its own. This was certainly a banner year for Kickstarter

games; there's clearly a strong desire among certain fan bases to see more of the types of games they love. And that means more offbeat, creative games, which I'm always in favor of. But I suspect there might be a bumpy road ahead—not all of the funded projects will ship a game, and some have already failed to deliver their promised rewards. And I think people are starting to question why so many large companies are moving into a space that seemed to be built for smaller, artistic projects.

**🕒 SHOUT-OUTS:** DEAR ESTHER was great. I sat down to play half an hour at most, and instead spent the next three hours, uninterrupted, as I traversed the whole thing. It was a strong, quiet presence in this year's game releases, and no one quite seemed to know what to make of it. But I think it'll hold its own as an example of what the games medium can do for storytelling for many years to come. UNMANNED by Molleindustria was an unexpected and worthwhile game. I think it's better if I don't spoil it, but there's a lot of freedom in what you're able to try in the game, and I found myself playing through a couple times just to see how I could affect the main character's life. One last one: shout-out to the Northways for finishing INCREDIPEDE—it's a great, weird physics game that's totally worth checking out.

**✿ ITOLD YOU SO:** I think some of the recent self-examination going

on in game journalism is a [perhaps clumsy] step in the right direction. It's good to question the closeness of the relationship between PR and game journalism, and even certain journalists. But there was some finger-pointing, which probably did more harm than good. Vilifying certain people may make for a better story, but it also gives the false impression that the problem is solved, rather than looking at the larger picture that created it.

**🕒 ONE-SENTENCE REVIEW:** 2012 was stressful, but every year is—and there was not enough time to play too many games. :)

**MARK DELOURA**  
(GDMAG ADVISORY BOARD)



**🕒 OUT WITH THE OLD:** At the beginning of the year, social games were still The Big Idea. Now we talk about games that are social, no matter what platform they are on. I like that idea much better.

**🌱 IN WITH THE NEW:** I've really come to appreciate JavaScript. I've been a diehard C++ guy for years, but the productivity increase I've seen using JavaScript for prototyping and Node.js for back-end systems has been dramatic. Partly it is due to ease of use, partly due to the massive

community, and partly the number of established frameworks. Of course, JavaScript performance doesn't compare to native code, and debugging can be an adventure. But all in all, it is incredibly useful.

**🕒 SHOUT-OUTS:** Spry Fox continually impresses me with their creative, whimsical game designs.

**✿ ITOLD YOU SO:** Graphics fidelity continues rising in mobile games, leading to rising development costs. Add to that a very crowded marketplace and developers are starting to ask, "How do we improve discoverability? How can we make our game stand out?" Enter Publisher 2.0.

**🕒 ONE-SENTENCE REVIEW:** 2012 was tumultuous and full of opportunity.

**ROBERT BOYD**  
(ZEBOYD GAMES)



**🕒 OUT WITH THE OLD:** I can't think of any things that the industry has grown out of this year. However, the number-one thing I wish the industry would grow out of is manipulative freemium practices. The freemium model is not bad in and of itself, but the practice of designing games specifically to addict the weak-willed in an attempt

to encourage them to spend hundreds or even thousands of dollars needs to stop.

**🌱 IN WITH THE NEW:** One thing that I'm extremely excited about is the combination of traditional controls with new forms of control in the same game. We started to see this in the DS and the Wii, but I expect the use of traditional controls being supplemented by alternate controls to really come into its own with the Vita and the Wii U. We've already seen some great examples with some of the early Vita games—using gyro controls to fine-tune your aim and camera in various 3D action games, having the UI double as additional commands in SILENT HILL: BOOK OF MEMORIES, and fast menu controls in many games. You get the precision and speed of traditional controls combined with the intuitiveness and versatility of touch- and motion-based controls—the best of both setups.

**🕒 SHOUT-OUTS:** Keiichiro Toyama's work on GRAVITY RUSH. We already knew that he was a talented horror game director thanks to his work on SILENT HILL and the SIREN series, and with GRAVITY RUSH, he's cemented his status as one of the top video game creators of our time. GRAVITY RUSH is extremely creative in every aspect—from its vertically oriented floating cities, to its unique art palette and unorthodox soundtrack, to its exhilarating gravity-based combat and exploration. And it's a great showcase of how



to combine traditional and nontraditional controls to increase the player's immersion.

**✦ I TOLD YOU SO:** Gonna have to go with the obvious one—the fall of Zynga. It should have been clear to everyone that Zynga's business practices were not sustainable—they were engaging in the video game equivalent of strip-mining.

**🎧 ONE-SENTENCE REVIEW:** With more and more quality games coming out every year, visibility is going to become the number-one challenge for developers.

**KEITH FULLER**  
(FULLER GAME PRODUCTION)



**👁️ OUT WITH THE OLD:** I'm glad to see many first-time developers going straight into entrepreneurship rather than falling prey to the long-held belief that you have to "break into games" at a big studio. It's exciting to see so many people focusing on just shipping games that they want to make and worrying about the details of a career or a company afterward.

**🌱 IN WITH THE NEW:** In the past year I've appreciated the flight of so many triple-A developers into indie ventures, eschewing big bucks and

corporate benefits in order to fulfill their passions more precisely. What I hope to see next year is more of these individuals banding together either as slightly larger studios or consortiums of studios to enhance their discoverability in the market.

**🗨️ SHOUT-OUTS:** Brian Provinciano of VBlank, who singlehandedly and with great determination created RETRO CITY RAMPAGE on, like, a billion platforms. Aaron San Filippo and Greg Shives, former coworkers of mine who left the warm nest of triple-A to strike out as indie devs, creating their own companies simply to better fulfill their creative urges. Local Madison mobile studio PerBlue, who in the space of about four years went from a group of students with no game-dev knowledge to a flourishing organization.

**✦ I TOLD YOU SO:** I wish I'd been wrong, but the triple-A sector has continued to collapse under the weight of its own horribly broken system, consolidating into fewer larger studios and leaving a trail of laid-off employees in their previously occupied space.

**🎧 ONE-SENTENCE REVIEW:** With the proliferation of tablets and smartphones, the increased availability of powerful and easy-to-use tools, and the lowered barrier to entry across the channels, it's been great to see the growth of mobile games bringing to light not only more gamers than we were previously aware of, but more developers, too!

**DAVE MARK**  
(INTRINSIC ALGORITHM)



**👁️ OUT WITH THE OLD:** I think we are finally getting past the institutionalized mentality of "gamers [want]/[don't want] [whatever]." For too long, that led to devs *telling* us what we would like to play rather than letting us *choose* what we liked. Instead, lately there are enough definitions of "game" out there to make Ludwig Wittgenstein and Sid Meier toast each other.

**🌱 IN WITH THE NEW:** I see a lot of promise in the increasing use of automatically or procedurally generated content—whether it be for narrative, dialogue, animation, or level generation. Anything that can help us power through the bottleneck of content creation in games will be a major boon to the industry.

**🗨️ SHOUT-OUTS:** At first, indies were cute, a little awkward, and mostly obscure. Now, with the tools, platforms, visions, and the freedom to do something different, they've gotten to the point where they are commanding a lot of attention—and changing games in the process. Huge +1 to the indie space as a whole!

**✦ I TOLD YOU SO:** A year ago, I was telling people that Unity was

going to take over the dev world. While it hasn't quite done that yet, there is a significant level of legitimacy to it now [Rovio!]. This can only be a good thing for the indie space in particular because it lowers the technical and financial barriers for entry.

**🎧 ONE-SENTENCE REVIEW:** I like to think of this past year in game development as the one where we finally stopped caring about what Roger Ebert thought of us.

**ADAM SALTSMAN**  
(SEMI SECRET SOFTWARE)



**👁️ OUT WITH THE OLD:** Not to beat a dead horse or anything, but I do think it's worth noting that Zynga's approach appears to be proving to be not terribly sustainable. I prefer to interpret this as a whole new demographic of game players wising up and looking for something that gives their life more value. I think we [we being the rest of the game makers at large] have a huge responsibility to provide intelligent, diverse and approachable games for them.

**🌱 IN WITH THE NEW:** True self-distribution. Humble Bundle/Store is awesome, but a lot of games are going all the way. Obviously MINECRAFT is one example, but SPY PARTY

is doing a pretty good job even though it's not even an open beta yet. FTL ran a great beta. Introversion Software [PRISON ARCHITECT] built their own Kickstarter service. From a business perspective I think these are all really positive examples. It's not *just* MINECRAFT anymore. I love that. [Also, holy crap, did you see how many amazing small/indie games came out this year? The list is pretty staggering. How are we going to top that next year?]

**🗨️ SHOUT-OUTS:** I hope he reads this—I know he will get all embarrassed—but for me it's probably Zach Gage. That dude makes exactly the kinds of games I love to make, only he's really, really good at it. SPELLTOWER is great, GUTS OF GLORY is *fantastic*, and all the prototypes he's been building and not even releasing are all totally amazing too. Dude is brilliant. Second place to Nels Andersen and the MARK OF THE NINJA team at Klei—that game really, really pushes my buttons. ALL OF THEM.

**✦ I TOLD YOU SO:** I was going to say "Zynga" but ugh, that horse is so dead. This might be a better observation for last year [or maybe next year], but the ascendance of Capy Games as a kind of joyful, exploratory force is something I saw coming way before SWORD & SWORCERY happened, much less SUPER TIME FORCE or their assistance on SOUND SHAPES, or, or, or...

**🎧 ONE-SENTENCE REVIEW:** Sounds cheesy, but at least on the indie side I



think we're growing up and getting more open-minded (though we have a long way to go still), and the awesome results of this past year are the obvious consequence of that.

ANDY

## SCHATZ

(POCKETWATCH GAMES)



**OUT WITH THE OLD:** Good riddance to the App Store/Facebook gold rush. It seemed that for a couple years all anybody could talk about was the gold (GOLD!) to be found in IAP and spammy micro-trash. Now that approach is just another color in the game designer's palette. It's not gone but it's not the talk of the town anymore either.

**IN WITH THE NEW:** The Rise of the YouTube press! It seems that as the larger blogs have come to supplant much of the traditional gaming press, youtube channels from erudite upstarts have found a ton of popularity due to the personalities of their creators and the intimacy of their presentation.

**SHOUT-OUTS:** For a long time "retro" in the video game world referred to retro-console games: platformers, metroidvanias and the like. But this year saw some mega-popular retro COMPUTER games like FTL, LEGEND OF GRIMROCK, and XCOM. From someone who

never had a console as a child, may this trend of digging up old C64 and DOS relics continue!

**ITOLD YOU SO:** I wish I could say the Steam Box, but that still hasn't happened (what the hell, Valve!) I'd also like to shake my head at the downfall of a high-profile Kickstarter. While that hasn't happened yet...it will. Other than that I guess I haven't had much of a year for prognostication!

**ONE-SENTENCE REVIEW:** Late in the console cycle, the game industry had a distraction-free year with massive growth in crowdfunding; may we have many more like this!

CLINTON

## KEITH

(GDMAG ADVISORY BOARD)



**OUT WITH THE OLD:** We haven't grown out of the blatant sexism just yet, but we're far more conscious of it than we were a year ago due to some outspoken individuals.

**IN WITH THE NEW:** Kickstarter. I love seeing studios like Double Fine getting serious strings-free money directly from consumers before a game is made.

**SHOUT-OUTS:** I've been impressed with so many studios. I can't choose one.

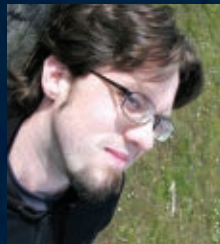
**ITOLD YOU SO:** Valve's employee handbook reveals that we can treat game developers like trusted creative professionals and build a culture that breeds success.

**ONE-SENTENCE REVIEW:** After 20 years of development, I've never seen a better time in which to make video games!

NELS

## ANDERSON

(KLEI ENTERTAINMENT)



**OUT WITH THE OLD:** It seems we've mostly gotten over motion controls, which I think isn't unwelcome. They're still very well suited to particular domains (Harmonix continues to absolutely nail it with DANCE CENTRAL), but in general, motion controls felt like a solution looking for a problem. It feels like we're past that wave of pseudo-Holodeck fetishism that motion controls were supposed to provide, which I think was a distraction from the real goal of finding ways to design games that are more intellectually and emotionally engaging.

**IN WITH THE NEW:** This is probably a little of a cheat, but next year is probably when we're going to see a lot of the big Kickstarter games come to fruition (or not). There have been a few already,

of course—most notably FTL and DIAMOND TRUST OF LONDON—but things like DOUBLE FINE ADVENTURE, REPUBLIQUE, WASTELAND 2, and others are probably going to start blooming next year. Hopefully they'll all be fantastic games (or at least most of them) and they'll demonstrate that the whole crowdfunding thing can be viable, and more importantly, a reliable way of making games.

**SHOUT-OUTS:**

Again, a little bit of a cheat, but I didn't play DARK SOULS until this year (ed. note: We'll allow it, thanks to the PC release), and wow, I cannot begin to discuss how transfixed I was (and am!) by the game. The design is a tempest of beautiful whirling contradictions. It's so tight and focused, but broad and allowing for a tremendous diversity of choice. It's restrained but has so much faith in the player's desire to richly explore the game's offerings. It's confident, both in its own design and in its audience. From Software did an absolutely breathtaking job. As a second shout-out, Telltale is absolutely killing it with THE WALKING DEAD. The writing is smart and mature (in the proper sense), the characters are robust and interesting, and it really commits to its notion of player choice.

**ITOLD YOU SO:** The shine coming off vapid social games. As hollow treadmills that offer vanishingly little while making increasing demands of the audience's time and/or wallet, I think the shallowness of many of those games is being increasingly

acknowledged, both by players and the industry.

**ONE-SENTENCE REVIEW:** 2012 was a year of innovative design and bold games that found success by refusing to cleave to tired and mundane tropes.

ANNA

## ANTHROPY

(AUNTIE PIXELANTE)



**OUT WITH THE OLD:** I'm really proud of how the industry has finally acknowledged and overcome its rampant sexism and hostility toward women. Wait, whoops—this is an answer from a year in the future. You'd all better make sure it comes to pass, or there's gonna be a killer paradox.

**IN WITH THE NEW:** I appreciate that Game Developer magazine has moved away from having to include a two-page spread of a white dude with a huge gun in every issue. I hope this trend continues into the future.

**SHOUT-OUTS:** SWORDFIGHT made me experience greater professional jealousy than any other game I've ever seen. That's the one you play by wearing Atari joysticks in strap-on harnesses, trying to press the other player's button with your shaft. Also, your hands are cuffed behind your back. I wish I'd made it.



**ONE-SENTENCE REVIEW:**  
Freaks, normals, amateurs, artists, dreamers, dropouts, queers, housewives, and people like you still have a lot of work ahead of us.

**NOEL LLOPIS**  
(GDMAG ADVISORY BOARD)



**OUT WITH THE OLD:**  
Good riddance to physical media! It had it coming for a while, and I expect

it'll disappear with the next generation of consoles. So liberating not to have any more game boxes, DVD cases, or books to lug around!

**IN WITH THE NEW:**  
Without a doubt, the "cloud." Seamlessly keeping data on the network and making it available at any time from any device is a complete game changer. Whether it's Steam games, iOS game saves, iTunes Match music, tool data, or even Kindle books, it's clear that the future lies in that direction.

**SHOUT-OUTS:**  
I was most inspired by *THE BINDING OF ISAAC* by Edmund McMillen and Florian Himsl. The game is

a fresh take on rogue-likes and has a sublime pacing and difficulty curve. It shows the amazing depth possible in an indie game created over a few months.

**ITOLD YOU SO:**  
The rise of free-to-play games on most platforms (and the rest are coming). Unfortunately, most free-to-play games are much worse than games with traditional payment models. Here's hoping that changes next year as we learn how to design better games that fit that approach.

**ONE-SENTENCE REVIEW:**  
A year of indie games of amazing quality and quantity, and the distribution channels are starting to get saturated.

## 2012 According to Twitter

Here's what our Twitter readership had to say about 2012:

"2012 is the year when free-to-play was no longer the hip thing to do...It became the \*only\* thing you could do." —@ShnSmit

"Creativity Wars: The Indies Strike Back." —@evans gr

"I think our 'friends' over at EA put it best: 'Consumers won't pay for crap.'" —@thesmall001

"It's been one hell of an up-and-down roller coaster ride!" —@Tephlon212

"Calm before the storm." —@originx1

"Another year of closing great studios." —@noahbradley

"The indies are rising." —@tgdfweb

"Independent breakthrough innovation." —@heatherross

"Indie dev became mainstream." —@grindheadgames

"Tumultuous." —@DaveVoyles

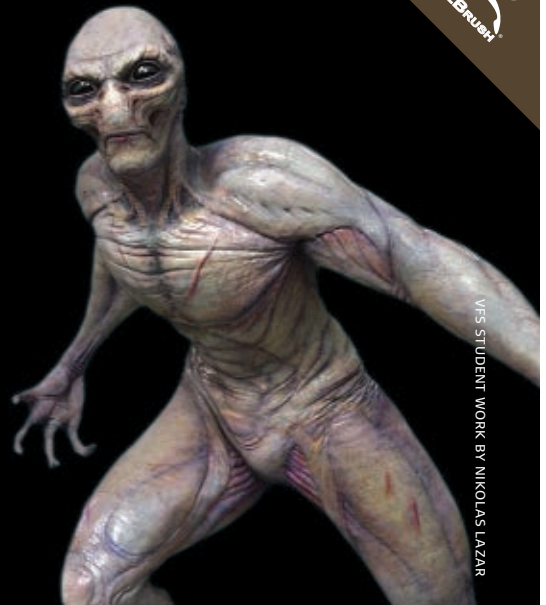
"The year of no fear." —@VitoGesualdi

# MAKE MORE ENEMIES

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.

The LA Times named VFS a top school most favored by game industry recruiters.



VFS STUDENT WORK BY NIKOLAS LAZAR

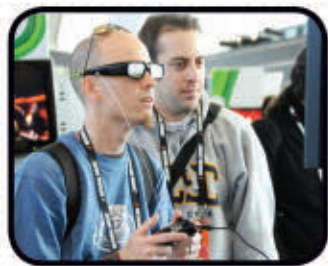
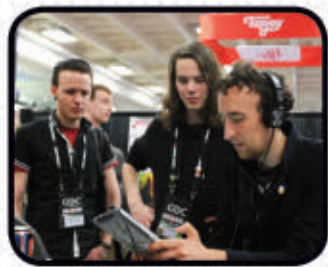
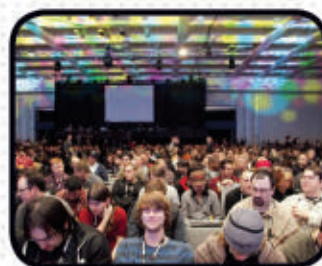
VFS

Find out more.  
[vfs.com/enemies](http://vfs.com/enemies)





# THE BEST ON-DEMAND CONTENT FROM THE GAME DEVELOPERS CONFERENCE SHOWS



# GDC Vault

Streaming video, audio, and  
PowerPoint presentations  
from GDC 2012, GDC Europe,  
GDC China, and GDC Online.

**EDUCATION**  
GROUP RATES AVAILABLE!

For more information visit: [WWW.GDCVAULT.COM](http://WWW.GDCVAULT.COM)



# art of *war*

how to animate realistic  
medieval combat—and  
why that will make your  
fantasy game better

>>>

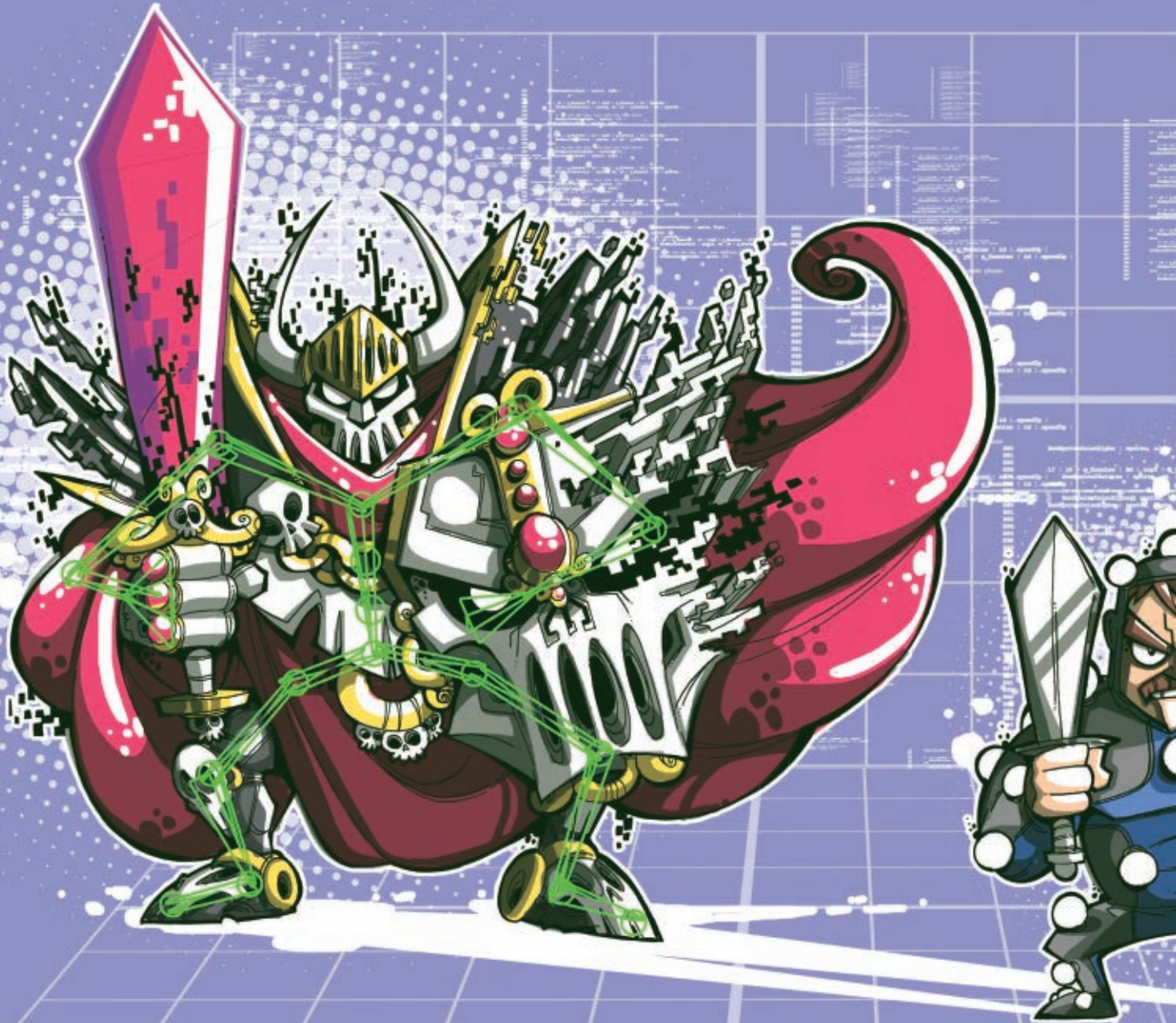
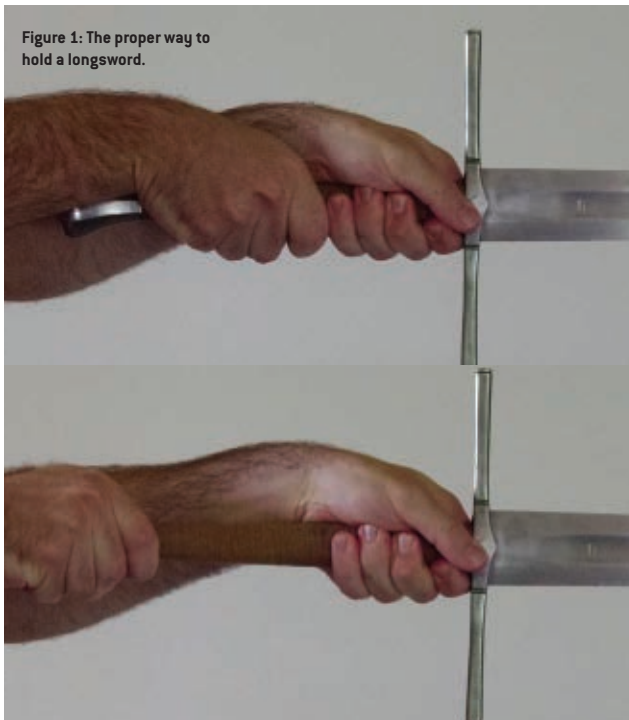




Figure 2: The four primary guards: (clockwise from top left) Phlug, Alber, Vom Tach, and Ochs.



Figure 1: The proper way to hold a longsword.



The first time I walked into John Clements's Iron Door Studio, I learned how to hold a longsword.

I thought it was obvious: Grip the handle with both hands. That's how I'd always imagined it was done. It's how they did it in the movies, after all. The handle is the comfortable bit between the pommel and the cross-guard. It's large enough to accommodate both hands. So I held it there, right?

**Wrong.** The right hand (or the leading hand) indeed goes just below the cross-guard. The *left* hand should grip the weapon by the pommel—that's the knob at the end of the handle—in most circumstances.

At first, I was a little dubious. "Hold it there? Really? I thought that part was just for smashing skulls. Or balance. Or decoration." My mind drifted to Orlando Bloom in an early scene in *Kingdom of Heaven*, where it's quite clear that he grips his weapon in the way that seems most harmonious: by the handle,

with both hands. Later in the film, he even helpfully confirms my bias by smacking someone in the noggin with his hand-free pommel.

The reality of the pommel is a little more complicated. It is used to knock sense into your enemies, it does affect the balance of the sword, and sometimes it's even pretty to look at. But, it's also a great place to hold the weapon. And it's a perfect example of how all my assumptions about the sword were challenged when I first set out to learn the reality of the weapon.

### grounding fantasy in reality

As it turns out, how to hold the longsword is also a great place to start talking about what that reality can mean for animators. If a video game character grips the pommel with its trailing hand, the resulting animation will have fewer





Figure 3: Example of turning 180 degrees between two guard positions.



problems with deformation around the wrist, less clipping between the sword's mesh and the character's mesh, and will display better biomechanics when cutting (which we'll talk about later).

The first two points are subtle improvements, and are best demonstrated by gripping the handle by both hands, then extending the sword forward, holding the weapon at eye-level. From this position, start turning, windmilling, and cutting with the weapon while keeping it in front of you. If you don't have a sword immediately available, you can do this with any wooden or plastic dowel. Just hold the dowel roughly three to five inches from the end in order to simulate the exposed pommel.

As you swing, pay attention to how often the pommel wants to intersect with your wrist, especially when you try to drop the blade to the lower right, and note that in

some positions, you can't continue an arc because your trailing wrist simply won't contort enough to facilitate the movement.

As I said, these are subtle points. The real magic happens when you now try the same thing, but grip



the pommel, instead of the handle, with your left hand. The first thing you'll notice is just how much more leverage you have. The sword is a lever, after all, and your leading wrist is the fulcrum, so it makes sense that the further back from the fulcrum you're able to grip, the more control you'll exert on the blade. You should also notice that this method is easier on your wrist (which will help with deformation), especially if you allow the pommel to slide and turn freely in your palm. Your wrist can now stay relatively straight through most swings, and there's no longer any danger of the pommel clipping through the wrist's mesh on your character models.

### bad sword, bad reference, bad animation

Inevitably, you'll want to capture some reference. Even if you're

using motion capture, it's always good practice to "feel" the movement yourself, or to whip out a camera and go through some of the motions.

The best way to not get good reference, no matter how you decide to hold the sword, is to use a crappy weapon. For most people, the easiest access they have to a generic sword is either through a catalogue, a renaissance faire, or even the odd novelty store. With the rare exception, the weapons you get from these sources are universally terrible for your animations—they're often much too heavy and poorly balanced. We have one of these weapons at my workplace, and even after two years of swinging steel as a martial art, I still can't do anything with it. That weight transcends physical reality, and every skilled attempt to mitigate it directly influences how our characters move. It's an awkward and sluggish prop that



Figure 4: A sequence of strikes starting in Vom Tag.

awkward and sluggish. Early on, we decided not to use it.

Instead of trying your luck with weapons that will harm your animations, it's almost always better to simply go to a hardware store and buy a wooden dowel,

or a length of weighted PVC pipe. If you're feeling crafty, you might even want to make your own wooden sword (historically called a "waster"). Wooden props like these will certainly be much lighter than steel, but they'll be easier

to swing in a good way (good for the director, the animator, and the actor). The alternative is a poor knockoff that will cheapen your results by virtue of your trying to use it, rather than just hanging it up and staring at it (as it was made for). Bad swords make your job more difficult than it has to be.

The weapons that John and I are using in these images are made by Albion Swords ([www.albion-swords.com](http://www.albion-swords.com)). There can be long waiting times for these weapons, so it may not be an ideal solution for gathering good reference quickly or cheaply.

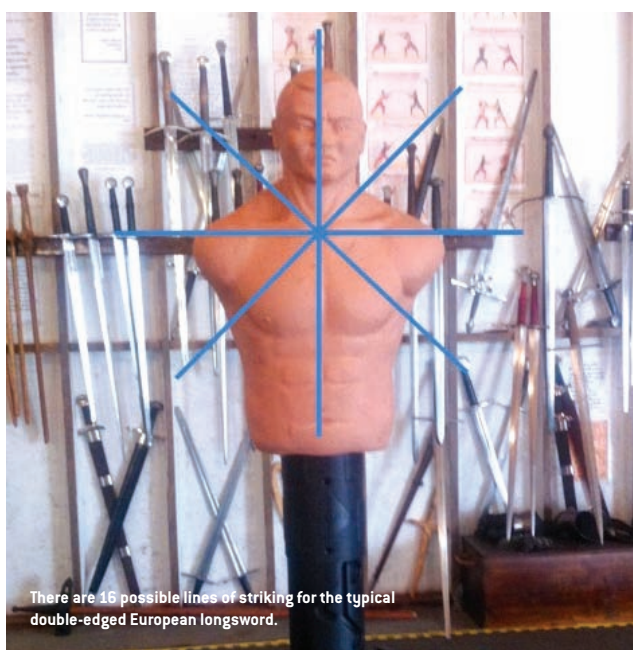
#### ||||| ALL ABOUT UNIVERSAL BIOMECHANICS

As I learned more about the art of fighting with the longsword, as Medieval and Renaissance Europeans understood it (and actually chronicled in dozens of study guides), I began to understand how intuitive the whole skill set actually was. There are a few basic guards, roughly nine vectors of attack, and a handful of

rules that guide your footwork. The more advanced techniques, while impressive to look at, are largely ancillary: In all the sparring matches I've seen, the flashier techniques are never used. In fact, the more I watched and the more I learned, the more it started to feel familiar.

That sense of familiarity didn't come from the movies or the stage—and certainly not from the highly sportified world of foil fencing. No, the moves I was seeing in sparring matches, which were reflected in the historical imagery plastered on the walls around me in John's studio, looked more like the close-quarters combat training we'd done while I served in the Army, or mixed martial arts matches on TV. It was savage and in-your-face. There was nothing at all pompous or chivalric about it. This stuff was real, and it was universal—because no matter what time or place we come from, we're all human and we're all governed by the same biomechanics.

Consider the weapon we've been talking about. If I had been handed this weapon three years ago and someone told me to swing



There are 16 possible lines of striking for the typical double-edged European longsword.





ago and someone told me to swing it, I would have done what pretty much anyone would do. Maybe swing it like a baseball bat, or because I used to cut wood when I was kid, I might swing it like an ax. If you were handed that weapon, what would you do? You might try to mimic what you'd seen Conan do, or imitate a samurai.

What you wouldn't do (at least, what most of us wouldn't do), is swing it like a golf club. But why not? Both are roughly the same length, both are used to hit things, and both do most of their business up to five inches from the end. Certainly, to swing a sword like a golf club and connect would be devastating to your target. So why don't we use it that way?

The answer is biomechanics and perception. Biomechanically, it's not efficient: The target of a golf club is at ground level, where the target of a sword is at eye level. Because the difference in targets is so drastic, we instinctively perceive that to swing a sword like a golf club is wrong.

Now let's use this thought process to delve a little deeper. Think about the target of your animations,

and where your character's enemies are located. Is swinging the weapon like a baseball bat the solution? Consider the game you might be working on: In an environment where your character is trying to kill people, the target of your swing is more likely to be the pitcher (in front of you), than the ball (next to you at the moment you swing). Is it still correct then, to stand like a batter, and swing as though you're trying to hit the ball? Probably not.

So now we can take a step back and ask ourselves: "What is the best practice?" Nothing beats calling in an expert, of course, but even if you don't have the time or resources for that, putting in a little effort in developing your fundamental understanding of human biomechanics with respect to weapons can help clean up your animations in a major way.

### Realism plays well with your IK rig

As I stated earlier, the basics of Renaissance fencing and martial arts (the discipline we call "MARE,"

or **Martial Arts of Renaissance Europe**) are pretty straightforward, and once you understand them, the rest becomes a matter of relentless conditioning and practice. For the purpose of the animator, however, the basics can be a great starting point that will give your warriors a unique visual silhouette, without requiring the animator herself to become a scion of martial prowess.

Once you're comfortable with holding the sword, the best place to start thinking about animating a swordsman is to understand the basic stances: what we call "guards." For the purpose of our animation, it's best to think of these stances as our idle positions. As you begin to employ these idles, you'll notice that no matter what sequence of cuts your characters perform or what direction they face, they will always end their movement in one of these poses. They work fluidly and efficiently with each other, and they emphasize control and tactical positioning. The four primary guards, as illustrated by John in **Figure 2**, are (from top to bottom) Phlug, Alber, Vom Tach, and Ochs.

What you'll also notice is that they play very well with your inverse kinematics rig. Like the algorithms that drive your rig, the weapon *leads* the motions, just like your IK target leads your animation. You'll also notice that the torso and arms seem to move almost independently of the legs. What's more, as you step through the footwork, you'll find that you can pivot your character around on a single foot, and stepping forward or backward is a simple matter of just mirroring your animation across your center plane.

Of course, these idles require the context of the basic cuts (called "Master Cuts") to be fully appreciated. Rather than go through the entire catalogue of idles, transitions, and cuts, I'd like to illustrate my point with a particular sequence.

In **Figure 3**, I start at the top in a fifth guard, called *Nebenhut*. Note how my leading leg is bent, and the trailing leg is straight. Also note how my feet are at a 45-degree angle to each other. Let's pretend that a new enemy has presented itself to my rear, and I want to turn 180 degrees





John Clements in action.

to meet the new threat. Rather than shuffle around and swing my blade awkwardly in an attempt to maintain Nebenhut, I opt instead to hold the sword steady.

As I begin to turn, my head and torso rotate first, followed by my trailing foot, which ends at 120 degrees (as relates to my left foot). On the third step to this sequence, I shift my weight onto my right leg, which is now my leading leg, and I deliberately bring my trailing foot to its new position. Notice that now our Nebenhut has transformed into Alber. To end the sequence, I lift my weapon out of Alber, and into Phlug.

Despite turning my entire body to face a new direction, my right wrist (the presumed target of our IK), never changed position. Also,

until I lifted the weapon at the end, the sword remained almost entirely motionless. Note that for my entire turn, the ball of one foot remained planted in a single position (please ignore the general position shift at step three; a wall was in the way, so I had to move back).


This theme of always returning to our basic guards continues to manifest even as we begin striking. In **Figure 4**, we see John start in a Vom Tag, and then step forward into a strike. What follows is a rapid rotation of the weapon to strike again from the opposite angle. He repeats this rapid back-and-forth several times, striking at a different angle on each pass.

Despite the change in vector for every strike, John always returns

to Vom Tag before striking again. He doesn't do this because he's necessarily trained himself to perform this specific transition for its own sake; he does it because it's the most biomechanically efficient way to pass from one strike to the next. This phenomenon is particularly useful for animators, because at any point between strikes we can end our sequence without popping into an idle pose, potentially jarring players out of their immersion.

|||||  
**start real, then  
exaggerate**

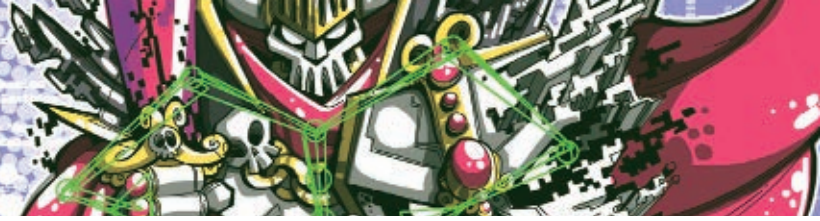
Exaggeration is, fundamentally, one of our jobs as animators; we

make our characters perform like an actor would perform on stage or on camera. Reality is always exaggerated or altered to fit the needs of a production. But whether you're talking about JADE EMPIRE or CALL OF DUTY, you should always start with a solid foundation in reality. For games in medieval or fantasy settings that include sword combat, taking inspiration from the right sources (like MARE) can set your combat animation apart and make the task of animating cleaner and easier. 

---

**EBEN BRADSTREET** is a visual effects artist and animator working at Xaviant LLC, in North Georgia. A U.S. Army veteran and martial arts enthusiast, he's also a student of John Clements and member of ARMA.

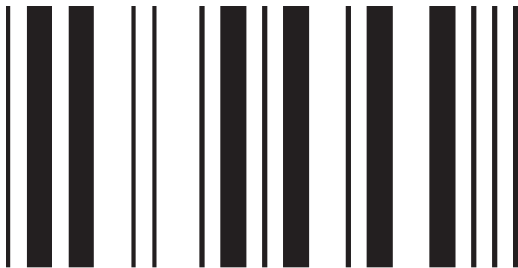




# 19

## give reality a chance

BY JOHN CLEMENTS



Imagine if game designers had never seen or heard of serious Asian martial arts, and never made any game with such influences. Then, one day, a budo master or kung fu expert steps up and says, “Hey, I think you could make some really interesting things using our unique craft as a resource. We move in really neat ways that you haven’t explored.” I like to think that game developers would quickly see that there was something significant and sophisticated there worth examining. They probably wouldn’t respond with conceited indifference—which I’ve seen firsthand when I bring up the historical medieval combatives I study, teach, and practice.

### people in the know

I’ve been studying Medieval and Renaissance close combat for over three decades. I make my living writing and researching on the subject and operate the world’s only private facility dedicated exclusively to the craft. I am no stunt fighter, costumed performer,

nor showman entertainer, but an accomplished martial artist who teaches an authentic combative discipline following genuine sources. Study of these historical fighting methods is my life’s passion and my career.

Now, I don’t think that everyone who makes a game in a medieval or fantasy setting needs to make a 100 percent accurate hand-to-hand combat simulator any more than I want to see the next CALL OF DUTY game stop working forever after your character dies for the first time. I do think, however, that the hard-working developers who make these games would have an easier time (and make even better games) if they drew from more realistic sources of inspiration when it comes to medieval combat.

The funny thing is, we already know this to be true. Just take a look at the original PRINCE OF PERSIA, where creator Jordan Mechner filmed his brother actually walking, jumping, and going through some rudimentary fencing motions as the basis for its rotoscoped animations and action. That game was an influential breakthrough, but later titles would essentially copy and

embellish upon those sequences, and the titles after that would copy the copies—and so on until the insightful grounding in realism of the original source was lost.

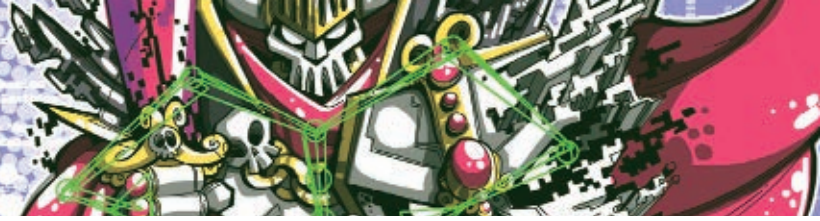
### drawing from the source

This process is common sense; if you are doing a modern special ops game, you consult with authorities of that profession. If you’re doing a boxing game, you consult with a professional boxer. If you’re doing an aircraft fighter game, you consult with

a fighter pilot. If you’re making a game about samurai, you certainly want to get the form and movements right by working with budo experts. When you copy the copies of copies, you get ever further from your original realistic base—which means you’re adopting the same embellishments and limitations that each successive generation of copies did without looking back at the real source material to see what your real design and animation options could be.

For example, a designer might see a fighting move in a movie and think, “This looks cool. I wonder





how I can devise a mechanic for players to do that?" But what of the possibility that what they witnessed is mere nonsense; an inferior action the game maker is just not qualified to evaluate? What if there are better alternatives? What if the "real thing"—a general principle of self-defense, or some element of employing a particular weapon, or a specific combination of techniques—is actually cooler? If the designer doesn't get the move from the right source, with a proper explanation of how it works and why, they will be missing out on how it fits in with the "game" of hand-to-hand combat, and won't be able to use that understanding as inspiration for how it could fit in with the game they're designing.

Yet this is more or less the general process I have seen for devising archaic close combat in games, and when I point this out, developers often feel insulted. Why? Aside from perhaps offending the creative sensibilities of designers, it's because I am suggesting that (*gasp!*) people who design games are not themselves also experts in the authentic sources of historical close combat. They have not trained long-term with accurate weapons in those methods, and they do not have extensive hands-on experience in striking realistic target materials with sharp weapons using genuine techniques, nor do they usually fit the profile of athletes conditioned to rigorous training in armed fighting skills. It seems like kind of a strange thing to be offended by. After all, most people haven't!

My job is to understand how such weapons handle, how they're maneuvered and manipulated, how they engage one another, what type of techniques and motions the human body is really capable of with said weapons, how physics affects melee combat, and how people respond (physiologically and psychologically) to violent actions. The specifics often include examples of how little-known gripping actions, armor, postures,

and different footwork are all interconnected. These elements are ones I can guarantee you have never seen in any movie, TV show, game, renaissance faire performance, or choreographed routine. It's this knowledge and these details which I hope to see developers take inspiration from while building their own games—instead of copies of copies.

### collecting core assumptions

When you develop a game combat system or a series of combat animations, you do so based upon a certain set of *core assumptions*: assumptions about how weapons and swords handle, about how armor functions, about what wounds could be causes, about how people respond emotionally to personal violence, how bodies and limbs react to injury, and about how real fighters learned martial skills. Naturally, if you're building a combat simulator off of a relatively shallow (or even erroneous) set of core assumptions, your game won't feel right. Even if your game intends to take a more stylized approach to its combat mechanics and animations, however, it's



worth investing the time to understand how the reality of combat translates into your game's core assumptions, so you at least understand *what* you're stylizing, and *why*. The martial knowledge and historical combat skills I've redeveloped permit developers to paint with a far richer palette of colors, if only the effort is made to pay some attention.

Without this realistic base, a video game animator ends up replicating the bad form capture of exaggerated stage

combat, accepting impressions from pretend bouts with poor weapon simulators, or copying the ritualized movements of some traditional fighting style. The consumers in turn get simplistic strikes and rigid blocks delivered from static, unwieldy postures combined with incessant spinning, whirling, leaping, and assorted useless (and even suicidal) actions that defy both common sense and basic human biomechanics. Meanwhile, a wealth of more dynamic and sophisticated movements, wardings, and alternative counterstriking actions from genuine fighting methods remain untapped for gamers. This is true even of light sabers—an imaginary weapon whose depiction virtually screams for a direct and brutal approach but instead is habitually stylized into an operatic wu-shu version of kendo.

I regularly see weapons, particularly swords, wielded by characters in video games in which the familiar figure animations and fighting motions are primitive and crude. For example, there are 16 possible lines of striking for the typical double-edged European longsword. Yet, players are repeatedly offered only the same standard three or four strokes taken right out of Japanese swordplay or borrowed from modern saber fencing and stage combat. It's little more than how children manipulate Nerf swords. All the diverse dynamic motions and distinct manners of adeptly manipulating a real weapon—with its wards, cuts, thrusts, slices, closures, and displacements—are entirely absent. Certainly, not every game with a sword needs to be a sword-combat simulator, but I am confident the software does not know how to allow players do them because developers themselves aren't aware that these things are interesting—or even possible—in the first place. With a little effort and attention, however, I think developers could use these realistic historical combat skills to paint with a richer palette of colors.

### devs and demonstrations: a deadly combination

Whenever I demonstrate for game developers, the initial reaction is often simply "Whoa!" They've never seen someone move the way I do or wield particular weapons as adeptly, and certainly not in person instead of on YouTube. And when I demonstrate that these movements are universal and apply to all weapons, whether it's a dagger or spear or a sword and shield, something seems to click. Developers tell me, "Wow, we won't have to use the same old things again, I didn't know that you could hold a weapon that way, I didn't know that you could strike with it that way, I didn't know it was possible for someone to step and pose in such a way, moving from one to another in that way."

Realism doesn't close doors. It opens them. Designers can see that with one kind of weapon, one certain type of move can come after another or that one move has a counter, or a certain position can be interfered with, stifled, or interrupted by another. Combat doesn't have to be the familiar "parry-riposte, parry-riposte, whackety-whack-whack, swirl-swirl" pattern.

Realism is not a dirty word for combat in fantasy games; it's a center point from where everything can and should begin. Realism doesn't lock you in or freeze you in place as a game designer. It's an empowering tool that lets you say, "Wow, I've got a really strong foundation to build on now." Only once your feet are grounded in the right place can your imagination really take off. 📖

**JOHN CLEMENTS** is a leading authority on historical fencing and one of the world's foremost instructors of Medieval and Renaissance fighting methods. He instructs both nationwide as well as internationally and (since 2005) from his one-of-a-kind private facility, Iron Door Studio, based outside Atlanta, Georgia.



# BEYOND SHADOWS OF DOUBT

## WORKAROUNDS FOR FOUR COMMON SELF-SHADOWING ISSUES

BY ANDREW WOO AND PIERRE POULIN

The shadows your in-game objects cast don't always play nicely with the other millions of hacks and tweaks game artists depend on to make a game look good, *especially* when you're looking at the shadows objects cast upon their own surfaces. Here are four common problems that crop up with self-shadowing in games, and strategies for eliminating and minimizing their impact as much as possible.

### AFRAID OF YOUR OWN SHADOW

¶ Digital shadow generation is an important visual effect in games. Shadows cause some of the highest-intensity contrasts, providing strong clues about the shapes, relative positions, and surface characteristics of the objects. They can also indicate the approximate location, intensity, shape, size, and distribution of the light source.

In a 1987 paper called *"Ten Unsolved Problems in Rendering"* [1] by Paul Heckbert, the fourth problem was the lack of an efficient and robust approach to deal with shadows. This barrier clearly persisted into the 1990s, where shadows in games were desired, but hardly realized because of insufficient performance. Instead, game developers would use certain tricks to give the impression of approximate shadows, which worked surprisingly well. Take, for example, Mario in Nintendo's SUPER MARIO 64: He casts a simple circular shadow on just the ground, without shadowing on other objects



# BEYOND SHADOWS

# BEYOND SHADOWS



and no self-shadowing taken into account (see **Figure 1**). It was not until the 2000s that realistic shadows became feasible for games, due to advances in both the actual algorithms and the graphics hardware.

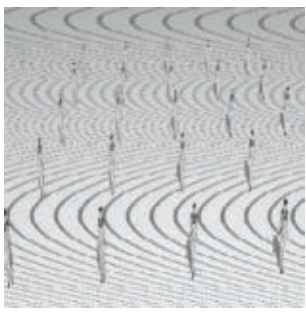
Most of the advances in shadow-related technology, technique, and literature, however, have been focused on resolving issues regarding (shadow) occlusion from other objects, which generally belong to the category of shadow algorithms that include shadow depth map, shadow volumes, and ray tracing [2]. Shadow occlusion from other objects is important for determining correct shadowing, but it is only one of two main factors; the other component is self-shadowing. While self-shadowing is usually accounted for in most shadow algorithms,

there are a few common issues with self-shadowing that can lead to some undesirable or missing visual artifacts.

In this article, we discuss unique self-shadowing issues to watch out for, and visually recognize the cause correctly so as to know which workarounds will succeed, as most issues have yet to find proper solutions. These issues include surface acne and level of detail, specular cutoff, the terminator problem, and bump mapping.

### SURFACE ACNE AND LEVEL OF DETAIL

¶ The surface acne problem appears in the form of moiré patterns or spurious black dots on surfaces (see **Figure 2** below for an example of moiré patterns that may appear). These errors in self-shadowing are usually either



**Figure 2:** "Surface acne" is a shadowing problem that can cause unintended moiré patterns to appear.

caused by a lack of numerical precision (common in ray tracing implementations) or insufficient resolution (common in shadow depth map implementations). You can usually work around surface acne issues with an offset or bias value within these shadow algorithms, but when they're

caused by differences in levels of detail (LOD), it can be a bit trickier.

To optimize your rendering performance, you can choose an optimal LOD representation from the camera's perspective, so that the polygonal representation only needs to be detailed enough from that camera. Shadows are computed as a render pass from the perspective of using the light source as the camera, so if there is a mismatch in LOD representations between the camera and light, you'll end up with bad self-shadowing. This situation is most common with shadow depth maps, but there can be scenarios when this would cause problems with other shadow algorithms. The workaround is to constrain the LOD to better match between the camera and light views, trading possible loss of performance for self-shadowing quality. One possible way to achieve



this would be to record the LOD per part during the camera rendering, and insist on the same LOD for shadow computations.

### SPECULAR CUTOFF

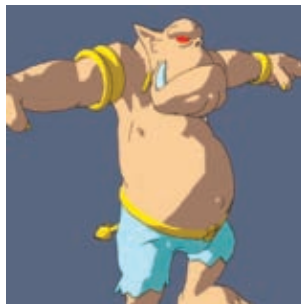
¶ The simplest example of self-shadowing is achieved with a dot product check,  $N * L$ , where  $N$  is the surface normal and  $L$  is the light direction, both with respect to the point to be shaded. This check is done in almost all rendering systems. This implies that no light directly reaches the portion of the surface that is facing away from the light without further computations. This also means that direct shading computation and shadowing from other occluding surfaces are only checked when  $N * L > 0$ . This is a concept similar to back-face culling from the view direction, except that it applies to the lighting direction in this case. While this check is physically correct, natural, and optimal, it does have a few consequences you should understand, such as specular highlight cutoff and the terminator problem.

It's easy to spot a specular highlight cutoff problem, which appears in the form of a shadow ending too abruptly [see **Figure 3**]. This issue crops up because the  $N * L$  evaluation also corresponds to the amount of diffuse reflection. Because the specular component is calculated independently of the diffuse evaluation, there can be surface points where  $N * L < 0$ , but the specular component is positive, indicating a specular contribution when the diffuse component has no contribution. In **Figure 3**, you can see that the self-shadowing check appears to have prematurely cut off the specular component.

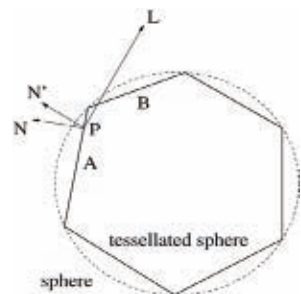
Most developers usually just ignore this problem, because it usually occurs only under unusual circumstances, such as when the light shines at grazing angles on the object, towards the camera. While it hasn't been solved as of this writing, you can reduce its visual impact by having a specular component with lower highlight size. As well, you can approach the problem from an algorithmic standpoint and multiple



**Figure 3:** Specular highlight cutoff issues cause the shadow on this sphere to end too abruptly.



**Figure 4:** Specular highlight cutoff can be used to creative effect—if you're going for a cartoony look, for example.



**Figure 5:** The terminator problem is caused by self-shadows falling upon low-poly meshes.

the specular term with a smoother decay function, using  $N * L$  as a factor for this function, which should reduce the effect of a sharp cut-off.

However, there are times where you want that sharp cutoff look—cartoon shading, for example. **Figure 4** shows a self-shadowing check that has been modified to diffuse shading only considering  $0 < N * L < 0.5$ , which causes the shadows to accentuate the cartoonish look.

### THE TERMINATOR PROBLEM

¶ The terminator problem has no connection to Arnold

Schwarzenegger whatsoever, but like some of the characters he plays, it can be hard to fight. The problem [illustrated in **Figure 5**] appears in the form of a jagged line dividing dark and lit regions, and it often happens in games when self-shadows are cast by low-poly meshes that are used to approximate smooth surfaces in order to improve performance.

In the 2D diagram in **Figure 5**, polygons  $A$  and  $B$  represent polygonal approximations to the smooth surface in dash. At point  $P$  on  $A$ , the vertex-interpolated normal  $N'$  is used to compute the illumination as opposed to the plane's normal  $N$ . Since  $N * L > 0$ , light contribution is present, and the shadow occlusion from other surfaces must be computed to determine whether  $P$  is shadowed. The shadow ray from point  $P$  intersects  $B$  and incorrectly concludes that  $P$  is in self-shadow. The result is the "staircasing" effect, as seen in the 3D diagram in **Figure 6**.

Obviously, you can work around this problem by increasing the level of tessellation, but odds are good that if you've run into this problem your poly budget is already stretched thin. You can also try to offset the shadow ray origin by a small distance along  $N'$  to avoid self-shadowing. Unfortunately, the correct offset value is difficult to figure out, and this offset typically assumes convex region behavior. (You could apply a similar approach in concave regions, this time to cast a correct self-shadow that would be absent otherwise, but you'll run into the same problem.) Furthermore, although this problem has often been described in the context of ray tracing, it is actually a problem in all shadow algorithms. The shadow depth map and ray tracing algorithms can provide the offset workaround, but unfortunately, a workaround has not been made available in the shadow volume approach.

In the case of the shadow depth map, the low mesh approximation can also result in errors near the polygonal edges. You can see in the 2D diagram within **Figure 5** that the

edge joining polygons  $A$  and  $B$  can vary significantly in terms of depth within a very small region, which means the shadow depth map pixel comparison can result in dotted-line artifacts such as ones seen in **Figure 6**.

### BUMP MAPPING

¶ Self-shadowing is often missing with bump mapping, where surface normals are perturbed to give the impression of a displaced, nonsmooth surface. Bump mapping is a useful representation to generate fine details without actually displacing the geometry [as in displacement mapping], which can easily increase the polygon count and therefore slow down performance. As a result, shadowing for bump-mapped surfaces makes those surfaces appear perfectly smooth, because shadow determination does not use the perturbed surface normal information at all.

Horizon mapping approximates the shadows cast by the bumps on the same surface [3] by interpreting the bump function as a 2D table of height values, so you can compute and store (for points on the surface) the angle between the horizon and the surface plane at eight or more azimuthal directions on the surface plane. During the rendering process, the horizon angle at the intersection point is interpolated from the light direction and the horizon map. If the horizon angle from the surface tangent exceeds the angle to the light, then this point lies in shadow. (GPU versions of horizon mapping have also been introduced [4] to achieve real-time performance.) For example, take a look at **Figure 7** and you can see that the left image indicates bump mapping with no self-shadows, and the middle and right images show self-shadowing with darker and lighter self-shadows, respectively.

Other self-shadowing algorithms for bump mapping exist [5]. However, all those algorithms only account for self-shadowing. Since the silhouette of a bump-mapped object is



Figure 6: Further terminator problems caused by shadow depth maps.



Figure 7: Left to right: Bump mapping with no self-shadows, darker self-shadows, and lighter self-shadows.

not displaced, shadowing from a bump-mapped object would not look correct—see **Figure 8** on the left-hand side, where the silhouette cast onto the floor doesn't reflect the object's shape. What's more, shadows of other objects cast upon bump-mapped objects do not look correct either; if you look at the left-hand diagram on **Figure 8** again, you can see where the shadow of a rectangle on the bump-mapped surface looks straight. You can resolve this issue with relief mapping (also known as steep parallax mapping, parallax occlusion mapping, or cone step mapping) [6][7][8] or view-dependent displacement mapping [9][10]—see the right-hand diagram in **Figure 8**—though the performance of such techniques will be slower than the

standard bump-mapping shadow computations.

### (SELF) SHADOWS OF THE DAMNED

Self-shadowing issues are an important research domain that has not received much attention in the past. We hope to see more straightforward solutions to these problems in the future. We are already seeing some progress, as nonoffset solutions have been attempted in the Thea Render and Blender to solve the terminator problem; and nonoffset solutions have been attempted for the shadow-depth map to avoid bad self-shadowing. For now, though, the ability to visually recognize its issues and choose an appropriate workaround remains key to generating realistic images.

Final note for the reader: It is not necessary to consider self-shadowing in all situations. For example, in serious games where technical illustration ideas are being presented for training purposes, self-shadowing can become a distraction to the user. Self-shadows may occlude important parts of certain objects, such as virtual equipment parts, that the user wants to see. In these situations, the only shadows appropriate would be one cast onto a floor (or wall) so that we have a reference to where the objects might be—as is the case with Nintendo's SUPER MARIO 64. 🎮

**ANDREW WOO** and **PIERRE POULIN** are the authors of *Shadow Algorithms Data Miner*, a digital shadow-generation resource. Andrew is the chief technology officer for NGRain, provider of interactive 3D simulation software and solutions. Pierre is a professor in the Computer Science and Operations Research department of the Université de Montréal, where he teaches and supervises research in computer graphics.

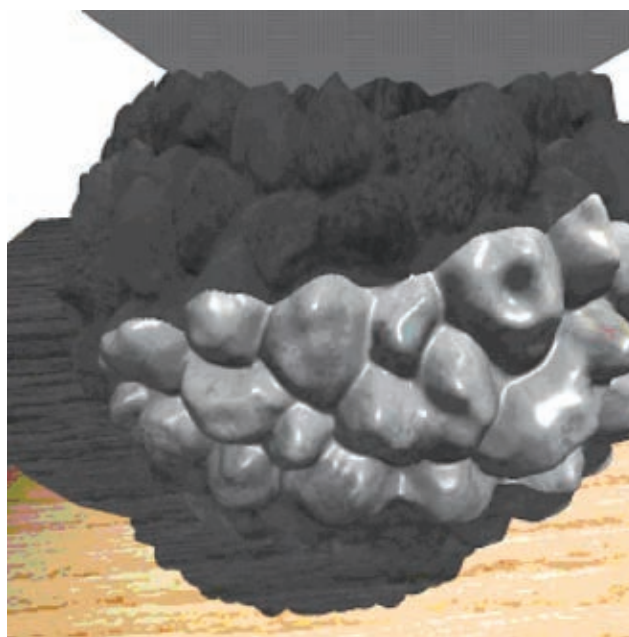
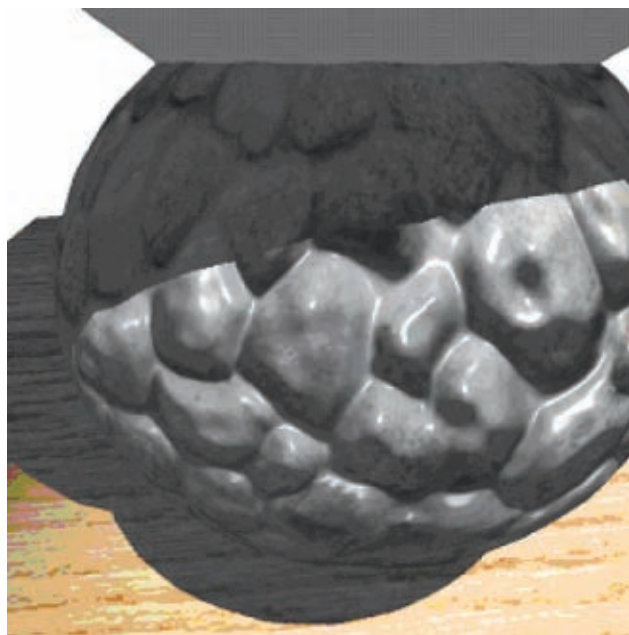


Figure 8: Top: Bump-mapped objects often cause self-shadowing issues, both with the shadows they cast and the shadows cast upon them. Bottom: View-dependent displacement or relief mapping can solve your self-shadowing problems here, but it is a computationally expensive process.

### REFERENCES

- [1] P. Heckbert. "Ten Unsolved Problems in Rendering." Workshop on Rendering Algorithms and Systems, at Graphics Interface Conference, 1987.
- [2] A. Woo, P. Poulin. "Shadow Algorithms Data Miner." A.K. Peters/CRC Press, June 2012.
- [3] N. Max. "Horizon Mapping: Shadows for Bump Mapped Surfaces." *The Visual Computer*, 4:2 (1988), 109–117.
- [4] P.-P. Sloan and M. Cohen. "Interactive Horizon Mapping." *Eurographics Workshop on Rendering*, pp. 281–286. London: Springer-Verlag, 2000.
- [5] T. Forsyth. "Self-Shadowing Bumpmap Using 3D Texture Hardware." *Journal of Graphics, GPU, and Game Tools* 7:4 (2003), 19–26.
- [6] A. Watt and F. Policarpo. "Relief Maps with Silhouettes." *Advanced Game Development with Programmable Graphics Hardware*. Wellesley, MA: A K Peters, 2005.
- [7] M. McGuire and M. McGuire. "Steep Parallax Mapping." *ACM Symposium on Interactive 3D Graphics and Games (Posters)*, 2005.
- [8] N. Tatarchuk. "Dynamic Parallax Occlusion Mapping with Approximate Soft Shadows." *Proceedings of ACM Symposium on Interactive 3D Graphics and Games*, pp. 63–69. New York: ACM, 2006.
- [9] L. Wang, X. Wang, X. Tong, S. Lin, S. Hu, B. Guo, and H. Shum. "View-Dependent Displacement Mapping." *ACM Transactions on Graphics (Proceedings of SIGGRAPH 03)* 22:3 (2003), 334–339.
- [10] X. Wang, X. Tong, S. Lin, S. Hu, B. Guo, and H. Shum. "Generalized Displacement Maps." In *Eurographics Symposium on Rendering*, pp. 227–233. Aire-la-Ville, Switzerland: Eurographics Association, 2004.





# Our universe is expanding **GROW WITH US**

» [www.ccpgames.com/jobs](http://www.ccpgames.com/jobs)

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











# 27

game data



Developer: Qeaszy Games  
Publisher: Sony Santa Monica  
Release date: August 7, 2012 (North America)  
Platforms: PS3, PS Vita  
Initial number of developers: 2  
Largest number of developers: 21 (15 full-time, 6 contributors)  
Length of development: 3-1/2 years (2 years prototyping, 1-1/2 years final product)  
Development tools: C++, GameMonkey, Dirac, FMOD, Box2D, Freetype2, libpng, zlib, SHA1, TinyXML

Jonathan Mak



# HOW SOUND SHAPES RESHA



**1** what · went · right

**LEARN WHAT YOU AREN'T MAKING**

Shortly after the release of 2007's *EVERYDAY SHOOTER*, Queasy Games founder Jon Mak (who had developed *EVERYDAY SHOOTER* solo) began collaborating with Shaw-Han Liem (also known as electronic artist I am Robot and Proud), based

*SHAPES*," Mak said. "We were just working on stuff."

Liem added his memories of the time: "Vaguely, the goal was to make a game where through playing it you were also creating music—whatever that was. A lot of the first stuff was: How can we do something similar to *EVERYDAY SHOOTER*, where you are literally just playing a game, but through your

we can, and use as many strategies as we could to get closer to that."

Mak explained that they shortly realized that the *EVERYDAY SHOOTER* setup of backing track plus sound effects wasn't powerful enough to let players make their own music, so they worked as a pair to go through dozens of prototypes over the following year.

"The reason that we kept on rejecting prototypes and trying new things was because each subsequent one would get us closer, but didn't have *that* feeling yet," said Liem. "We had a vague idea and a vague goal, but we knew what it *wasn't*. If we had a prototype and we played it, we could tell this was not what we wanted; it didn't have the feeling that we wanted from it. It narrowed down what we weren't going to do; it helped us decide what the game wasn't going to be."

Ultimately, the pair realized that—even having shed the *EVERYDAY SHOOTER* basis—they were being restricted by their own mindset.

"I don't like platformers, or level editors," Mak admitted, "but in the back of my mind they made sense. Making a level could be like writing a song, and platformers, like *SUPER MARIO BROS.*, are a game type that

everyone understands. It would be a safe environment for players to get into it."

With that in place, the game finally started to "click" for Liem and Mak, and thus began the first prototype that formed the true basis of *SOUND SHAPES*.

"What we were naturally trying to do was what it wasn't," said Mak. "That's a thing that we learned: We couldn't achieve our design goals with what we would do naturally."

**2** DEV TOOLS SUPPORTED ITERATION AND COLLABORATION

Even with the core concepts of *SOUND SHAPES* locked down, the game still continued to evolve across its entire development cycle, to the point where even the version that was shown at E3 in 2011 ended up vastly different from the final product. With new ideas to implement and new problems to be quashed, it was massively important that changes be quick to implement and work with, even if they were extensive. As a result, *SOUND SHAPES* was built from the start using scripting tool GameMonkey sitting atop a bespoke C++ engine.



on a shared interest in interactive music. Using the help of an Ontario government grant, together they began prototyping concepts without a final goal in mind.

"When we started, it wasn't like we were going to make *SOUND*

actions, you are composing the music? With the criteria that the system is deep enough that when you play the game, and when I play the game, it could create completely different music—or at least to map as many musical things to play as



# APED MUSIC GAMES—AND W

“The engine is set up for quick iteration,” Mak explained. “UI editing, debugging... We went through three quite different front ends, and you could whip them up really quickly. We had an in-game color editor and mixer, so you could change colors and sounds on the fly. And obviously, getting the in-game level editor working early was important, so anyone could jump in and make levels.”

Although Mak maintained that running the scripting tool causes your game to run “basically 10 times slower,” it was an important factor that allowed us to create the 300+ entities for our campaign levels without limiting our creativity—or that of our players. In a conventional run-and-jump platformer title without a level editor, devs can design entities to be used only in certain positions, so they won’t require code for every possible interaction. In SOUND SHAPES, however, a player can unlock an entity, place it on any surface, and use it to interact with any other entity and see what happens, with every possibility (within reason) having to be

accounted for. Should a tester find that an interaction between two entities wasn’t working properly, GameMonkey made it easy to quickly apply and test a fix.

The tools were critical for collaborating on SOUND SHAPES as well. A simple scripting engine allows nonengineers, like Liem, to create their own code; he was able to fully prototype “loop notes”—a key feature of SOUND SHAPES that was introduced only within the last year of development.

“The original SOUND SHAPES musical system was built around the idea that the music you made needed to sound like traditional songs,” said Liem. “When we started including external artists, we realized that if we were serious about making people excited about making music, it would be in the game’s interest to create ways to make their music something they’d recognize. When I came in with the loop-note concept to make that work, it wasn’t on a piece of paper—it was in the game, so I could show what was good about it. That was a big deal in being able to include more diverse minds.”



▼▼ We had a vague idea and a vague goal, but we knew what it wasn't. If we had a prototype and we played it, we could tell this was not what we wanted; it didn't have the feeling that we wanted from it. It narrowed down what we weren't going to do; it helped us decide what the game wasn't going to be. ▼▼



Photos by Mark Rabo

# HOW SOUND SHAPES RESHA

**3**  
**LIMITING PLAYER  
 POSSIBILITIES  
 CAN ALLOW MORE  
 CREATIVITY**

SOUND SHAPES may offer 300 individual entities—each with its own musical and play behavior—but the level-creation tool itself has been kept intentionally simple, with an admittedly large number of things the player cannot do, such as modifying behaviors, modifying individual colors, or having greater freedom in note placement.

These restrictions stemmed from our own challenges in designing levels for our main campaign. In a SOUND SHAPES level, all entities and notes on screen also create music. This vastly increases the complexity in making a level: You can make screens that play well but don't sound good, and vice versa. Balancing that is part of the fun of level creation, but if the editor offers you too many options, it becomes harder to find that balance. "A lot of creating the editor was deciding what not to put in," explained Liem. "Every time you add a new concept, you are creating a new possibility, but you are also adding more complexity."

The concept that informed the design of the creation tool was the same as that which informed SOUND SHAPES itself: Make the player feel like he or she is playing an instrument.

"When you play a note on an instrument, you can decide if it sounds good in that instant. If it sounds good after that last note, then you plan on what the next note will be," Liem said. "The process of creating the level should be like jamming on an instrument: immediate."

The intention, therefore, is to allow the player to play freely without fear of hitting "wrong" notes. "We could have had a polygon shape in the game, where you could modify every vertex," Mak said, "But you can make a lot of polygons with that that aren't fun at all. So we went with circle, square, and triangle—let's just have three. That turned out to be super powerful, and more than that, it's something that a player just gets."

**4**  
**BE PREPARED TO ASK  
 FOR HELP**

SOUND SHAPES began as a two-person project that swelled to, at its most,

21 people at one time including part-time contributors. But it's hard to limit the true number of people who had an important effect upon the final product, thanks to the influence and value of both our local Toronto independent game community and the wider global indie game community.

While we personally have been lucky enough to be based in a hub city with a (well-publicized) tight-knit game development community, Queasy Games also collaborated with developers such as designer Cory Schmitz (at the time, Seattle-based) and Pixeljam artist Rich Grillotti (at the time, Eugene, Oregon-based.) Distance is no impediment to finding like-minded individuals, but what was important for us was knowing when and where to rely on them.

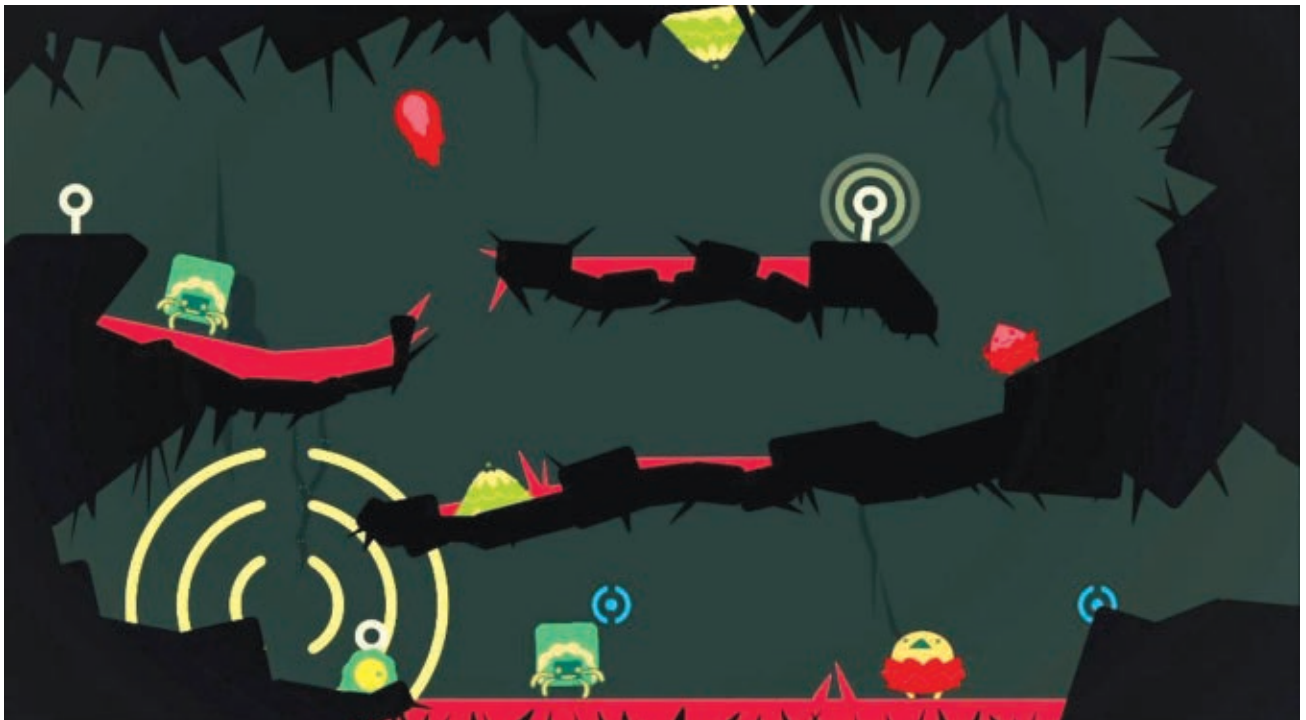
In the lead-up to E3 2011, Queasy Games was more than Mak and Liem alone, but it was still small, with just a few developers. And although Mak in particular has completed dozens of games since beginning his development career in the late 1990s, it was almost always solo, leaving the pair with no familiarity in running a studio while still "trying to figure out what the

hell this game was," in Mak's words. "It was really messy," he admitted.

For SOUND SHAPES, Mak and Liem began by drawing help from others when possible, such as from Metanet's Raigan Burns and Mare Sheppard, who helped flesh out the platforming mechanics through early level designs. An office was located in the same building as Capy (developer of MIGHT AND MAGIC: CLASH OF HEROES and the upcoming SUPER TIME FORCE), allowing Queasy Games to share resources—such as by hiring Capy's Dan Vader, who would become a vital part of the team—and expertise.

"Even people who didn't technically work on the game helped," Mak added, such as Jim and Em McGinley (of Bigpants Games and the Toronto Indie Game Jam) who used their knowledge of the local community to help find additions to the Queasy team.

"This happened to be local," Liem said, "But reaching out doesn't have to be to people in your city; there are like-minded people out there... If we had been in the mindset to be competitive with these people, rather than collaborative, things could have





# SHAPED MUSIC GAMES—AND W

gone really shitty, and for many companies that's the default stance."

Liem continued: "It's not a total lack of competition. It's similar to the music scene..."

Mak finished his sentence, "... you want to make some killer shit, but you want your buddy to make killer shit too."

## 1 what · went · wrong

### DON'T WORK FROM DEADLINE TO DEADLINE

With the PS Vita due in early 2012, the team was set on releasing the game within the launch window, and as a result we set ourselves harsh (and ultimately unrealistic) deadlines to work toward.

"Every day we'd come in and the mindset was, 'this had to be done yesterday,'" Mak said.

This pressure was doubled by our decision during development to release simultaneously on PS3, at a point when the PS3 version was lagging behind the PS Vita version. Getting the two versions in parity—leading to, at times, vastly different sets of bugs—became an intense drain, especially considering that we were also in the process of localization for four regions. At one point, we were under internal pressure to submit as many as eight different builds to our QA at Sony Santa Monica per day.

"That totally killed us," Mak said. "We even had to upgrade our bandwidth to 100mb/second. Maintaining two builds shouldn't be a big deal, but when you add localization and compress the time you have, it becomes a big deal."

With limited time and high stress leading to—unfortunately—dreaded crunch, the effect was highly detrimental to our ability to plan and keep to a schedule, only compounding the problem until the game was completed—in a timescale that, ultimately, could have been handled better (or at least, with a shorter crunch) had we planned to finish in July 2012 instead of *hoping* to finish six months sooner (if not earlier).

From a personal standpoint, Mak said, "When I compare my experience with EVERYDAY SHOOTER, where I had no deadline, I gave myself a deadline and my productivity went to bits. I'm not making decisions, or I'm not making good decisions that are bringing the game closer to finished. I do think for a certain type of person—not everyone—having milestones doesn't work. I naturally have a time I'm willing to work on a project; I'm like, 'Okay, this is enough; I can see this going down a black hole, so let's take what I have and ship it.'"

## 2 DEFINE YOUR SCOPE

Perhaps the most severe issue that we faced with deadlines was that we set deadlines before the game had fully taken shape. When we showed SOUND SHAPES at E3 2011, the basic design had been put in place, and it was expected to be completed as such. However, the design and scope continued to evolve across the entire following year of development.

"When we showed the game at E3, we expected to continue and complete what we showed," Liem said, "But there was so much momentum and excitement about the game that we kept thinking, 'Oh, well, we can add this, we can do this idea, we can include external artists and musicians while also thinking, 'This is supposed to be done.' We had never made a game of this scale to see what was coming."

This lack of experience running a studio and making a larger-scale game meant that when opportunities arose (such as working with musical artists Beck and Deadmau5), the SOUND SHAPES team would take advantage of them for the ultimate good of the game—but the deadlines wouldn't change to reflect SOUND SHAPES'S increased scope, which piled more pressure onto the devs.

"There were so many possibilities with SOUND SHAPES," Mak says, "And a lot of it is my fault for acting on it, but I always placed pressure to expand it. It was almost like building a music production studio. What if we add another sequencer? What if we get some



## SOUND SHAPES MIXTAPES

The SOUND SHAPES team continues to select the top levels from our community and highlight them on the "Greatest Hits" category (available on the SOUND SHAPES website [soundshapesgame.com](http://soundshapesgame.com) and in-game), but here's a selection of the levels that we found in the first month of release that let us know we hadn't completely failed in our aims.

### AND I TO FIRE

BY JORDANBUSTER

Jon Mak described this level as "blowing his mind," and there's good reason. We did our best to lock our entities and level tools down to stop the player from using them in ways we didn't expect, and within a week a player had already done things that we could never have expected, to thrilling ends.

### ARCADE FIREWORKS

BY DAFTBOMB

Daftbomb is one of the team's heroes in the community, and this level is the first

that caught our eye: a tough level that took our retro-game concepts to their logical extremes.

### INTERMISSION

BY ZOETROUPE

Most levels were initially like this: short and perhaps not too consistent in audio or visuals, but with strong themes and lots of promise.

### METAMORPHOSE

BY TABULATOR\_AT

This is another level that very clearly used an entity in a way that we didn't expect. We get a kick out of watching creators ask the player to abuse or treat an

entity in a completely different way than we used them in our campaign.

### QUICK, DINOSAURS!

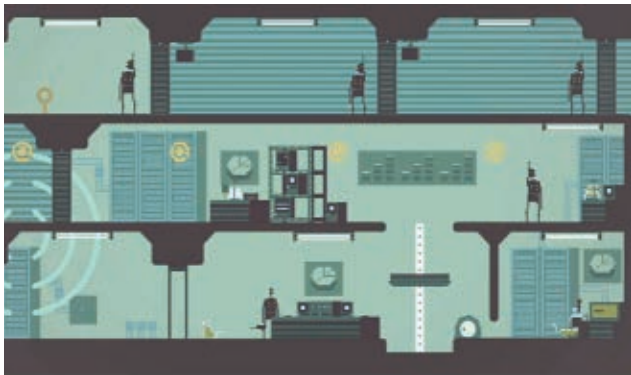
BY THEAVENGEDMARIN

Sound Shapes intentionally only has three "shapes" to draw in the world with and no ability to work with layers, but players are managing to draw more and more incredible art despite these limitations. This is one of the first levels we saw that did this—the artist created a world full of dinosaurs that was still fun to play.

# HOW SOUND SHAPES RESHA

distortion tools? It was just such a new frontier, and the concept sort of captures your imagination.”

Locking down scope is a double-edged sword. Had we ignored opportunities that arose within the last year of development, SOUND SHAPES would have been a lesser product. However, as time began to run out, we did end up having



to cut some in-progress content just to ensure we could polish and complete the most-finished parts of the game. If we had a clearly defined overall scope with some leeway (maintained through proper project management), we could have made our development smoother while still allowing for new worthwhile opportunities. Instead, we tried to do it all.

### 3 PLAN AHEAD FOR YOUR COMMUNITY

Since much of SOUND SHAPES’s design was in flux across the entire development cycle, it was an unfortunate reality that certain aspects of the game weren’t given as much attention as they deserved prerelease. Perhaps the most short-sighted decision in SOUND SHAPES’s development was to think of the game as a finished product we would deliver, rather than a service we would launch and build up by attracting and supporting an active and engaged community through our level-sharing tools.

Despite the fact that we had planned to create the game and polish the tools for the community to use, we didn’t properly plan how to structure or involve that community until midway through the final year

of development. As a result several ideas that we should have shipped with were simply not included—if they were even thought of in time.

“There’s a lot of stuff we should have shipped with,” Mak said, “such as the ability for us to select and feature levels on the website and in-game, Twitter integration... Ultimately, we didn’t have time.

Allowing players to record level audio for a sound preview, like on our campaign levels, would have been sweet, especially if you could do that on the website.”

While SOUND SHAPES has managed a respectable community thanks to the response to the level-creation tool and a highly functional community front end and website (over 10,000 levels were created in the first week, and the levels that continue to be created are regularly beyond what we thought were possible to create), there is a general feeling that with better planning we could have served our community better, such as in the case of the missing audio previews for community levels.

“Missing ideas like that hasn’t helped people create great levels [in] our community, even though they still do,” said Liem. “If you made a level and the audio preview sounded shitty, you’d think, ‘How do I fix that?’”

### 4 COMMUNICATE CLEARLY TO THE PRESS, DON’T LET IT DISTRACT

Games that feature interactive music aren’t always easily demonstrated to every possible player, which can

make it hard to clearly communicate exactly what the game is, how to play it, or why it works.

“We still have this problem,” said Liem. “It’s not an easy game to communicate. In game journalism, the medium you are communicating in is largely still text and images, and we haven’t come up with a great way of communicating the game in that way. There are these aspects that you can talk about, but you don’t really get it unless you see them all working together.”

Mak added, “I wonder if a lot of people are, like, ‘Oh, it’s just like a music toy,’ and don’t realize that there’s a real game there.”

This struggle to ensure the game was communicated clearly to the press also had its own cost to the project.

“In the middle of development we kept having to create press builds,” said Mak. “You’re essentially creating a finished version of the game on a regular basis, which is crazy. You’re trying to release a game, pushing stuff that isn’t final, but relatively bug free.”

“I think we handled it as well as we could,” Mak said, “in that the stuff we were polishing for press did end up in the game.”

In addition, Liem and Mak both made a point of being there if the game was being demonstrated, which was important, but costly. While the feedback was important (“We had been working on this game for three, four years and seeing people responding to it was really helpful,” Mak said), in each case, they could lose up to several days of development off-site.

However, Liem didn’t consider that to be entirely a mistake in terms of the game’s chances on release. “Because the game is so hard to describe, we were able to say what we thought it was directly to people rather than describe it to other people and have them do it,” Liem said. “I think part of the E3 2011 response that set off what the final game would become is because Jon himself went there, and through his enthusiasm for the game, he showed this kind of weird thing and

was able to crystallize what was cool about it. A demo of SOUND SHAPES is a performance; all game demos are performance, but demoing this game you actually have to do it, make it seem fun, make it seem interesting, and early on people on the team were the only people who were capable of doing that.”

It’s to the team’s pleasure that we have already seen—through YouTube and our community—that there are now hundreds (if not thousands) of people out there who are capable of doing that.

## SHAPING OUR SOUND

SOUND SHAPES was a truly unique project, and its union of music, play, and community engagement meant it had a range of particular needs and problems. Looking back, it becomes easy to say we should have done things this way rather than that way, but—perhaps due to our bad habit of letting scope expand beyond our control—it’s hard to say that we could have



done things differently. The team at Queasy Games is now trying to take what they’ve

learned and apply that to improving the community experience and maintaining on-time release of DLC (the first of which should have been released by the time you read this).

We’re most grateful that, despite our struggles, players have taken our work and run with it. They’ve created levels that are beyond anything we could have imagined during development—so if you get a chance to play SOUND SHAPES, it’s the player’s levels that we recommend you try! 🎮

**MATHEW KUMAR** is a freelance designer, developer journalist and general gadabout in Toronto, Canada who served as a designer and producer on SOUND SHAPES for the final year of production. He also worked on DYAD and THEY BLEED PIXELS and you should check those out too, ok?





AUTODESK

# MUDBOX 2013

BY MIKE DE LA FLOR

Autodesk Mudbox is one of the most widely used applications for high-resolution 3D sculpting and design, and its success is due in part to the feature sets Autodesk adds with each new release, such as the painting system in Mudbox 2009 and the Pose tools in Mudbox 2011. Autodesk's Mudbox 2013 release, by comparison, feels a little bit more incremental; it adds much-needed customizability and usability updates to its interface, improves the way Mudbox works with the rest of the Autodesk suite and Photoshop, and introduces an array of new tools to improve the sculpting and painting workflows, but it has little in the way of flagship features that immediately sold us on an update.

## VIEWCUBE AND THE ROLL-YOUR-OWN INTERFACE

» Mudbox's straightforward user-friendly interface was initially a key to its success, but as the product evolved into a robust application, the interface simply hasn't kept up. Aside from minor tweaks here and there (Mudbox

-----  
**BELOW: Mudbox 2013 can toggle on and off the display of hard-surface data such as edge creasing angles. Unfortunately, that's all it can do. Mudbox does not have any tools with which to assign hard-surface data to meshes.**

2012 introduced Maya-like marking menus, for example), the interface remained largely the same—until now.

For starters, you can now customize Mudbox's interface. You can resize, drag, hide, and redock the Layers, Object List, and Viewport Filters palettes; you can move and redock all the tool trays, and once you've made the changes you want, you can save the layout preset so it's easy to restore specific layouts for specific tasks (and even share layouts with other users). It's a

handy feature that really makes it easier to expose Mudbox's full set of tools and features.

Second, the popular ViewCube featured in other Autodesk applications (like Maya and 3ds Max) is now also available in Mudbox. The ViewCube makes it easier to navigate through 3D space, which is useful while sculpting and painting.

Third, Mudbox's dev team appears to have resolved a few basic usability issues that simply make the app easier to use. For

instance, you can now select, transform, and delete multiple scene objects at once, and you can delete unused materials to free up memory. You can also delete polygons from the base level mesh, though there is still no way to cap holes in a mesh. With previous versions of Mudbox, it could be a bit difficult to select the right object or sub-object, but Mudbox 2013 has updated the Select tool to make it easier to select polygon borders and UV shells.

These interface and usability improvements bring Mudbox generally up to par with what users expect from a modern 3D application, but it still has a few strange quirks. For example, objects in the Objects List Scene tree cannot be reordered, grouped, or hidden—but you'd think that users should be able to use that





Objects List window to organize objects. Also, since Mudbox has two distinct workflows, it'd be a big help to have custom tabs to store favorite tools, like Maya does.

### NEW CURVE TOOLS HELP WITH SCULPTING AND PAINTING

» One welcome addition to Mudbox 2013 is the revamped curve toolset. Mudbox has included a basic toolset for curves since its initial release, but the 2013 version extends its functionality rather significantly. Where the old curves implementation was simply a set of objects projected onto 2D screen space, the new curve toolset is comprised of three basic tools (Create Curve, Grab Curve, and Smooth Curve) that you can use to draw and edit freeform curves.

The most significant difference between the old and new curves is that the new curve tools produce curves that adhere to the surface of meshes, so you can use them as sculpting and painting guides—which comes in handy when you're trying to sculpt or paint complex or repetitive patterns. Note that while you can edit the new curves by smoothing

or dragging curve segments, you can't scale, move, or rotate them once they adhere to a mesh.

Mudbox 2013's new curve toolset doesn't replace the old one—they're both still in the application—but they are segregated from each other, which is a little strange. In my opinion, Mudbox's curve workflow would be more efficient if the Curve Tools tray included options for the original curve presets along with the

new Curve tools. After all, drawing a perfect square or circle is not easy with the new freeform curves.

### HARD-SURFACE AND DUPLICATION WORKFLOWS: BETTER, BUT NOT PERFECT

» The ability to duplicate and flip meshes is new to Mudbox 2013—and sorely needed—but I found the duplicate/flip workflow to be lacking. For instance, the Duplicate function has no offset or mirror options, so when you duplicate the mesh it occupies the same space as the original. In most other 3D applications, like Maya or 3ds Max, the duplicate function features the option to offset or mirror the copy across an axis in one step. Instead, you have to duplicate the mesh, and then use the Flip function to flip the mesh across your desired axis. Furthermore, you can only duplicate meshes, not polygons or groups of polygons. Ultimately, the Duplicate and Flip functions are nice, but they don't seem polished.

In hard-surface modeling, you use edge creasing and smoothing data to describe the topology of

This release introduces Duplicate and Flip. Regrettably, these new functions seem unfinished. For instance, duplicating and flipping a mesh requires two separate steps instead of one as in most 3D applications. Also new in Mudbox 2013 is the ability to customize the interface.

an object. For example, in Maya, the 90-degree edges of a cube are assigned a creasing angle to make the cube render properly. Since Mudbox was developed as an organic subdivision surfaces modeler, not a hard-surface modeler, previous versions could not display edge creasing, which created problems for any imported hard-surface model. With Mudbox 2013, you can display hard-surface data assigned by other applications, though you can't assign or edit that hard-surface data from within Mudbox, which would have been ideal.

### INTEROPERABILITY KEY TO IMPROVED PAINTING SYSTEM

» Mudbox's layer-based 2D/3D painting system is one of its best features; with its layer blending modes and opacity settings, and tool/brush selection, it's practically

#### AUTODESK Mudbox 2013

[www.autodesk.com](http://www.autodesk.com)

##### PRICE

> \$795, upgrade \$400 USD

##### SYSTEM REQUIREMENTS

> Windows XP/Vista/7 (32-bit and 64-bit) / Mac OS X 10.7.x / Red Hat Enterprise Linux 6.0 / Fedora 14 (64-bit only)

##### PROS

- 1 Customizable interface and better usability
- 2 Display hard-surface data
- 3 New Curve tools

##### CONS

- 1 Objects List cannot be customized
- 2 New Duplicate/Flip tools feel unfinished
- 3 No UV Tools



like working in Photoshop. It's not completely like working in Photoshop, though, so I'm glad to see that Mudbox 2013 added support for 16-bit Photoshop files, because that frees artists up to work with a more subtle range of colors for paint textures, and sharper details for displacement maps, bump maps, and normal maps.

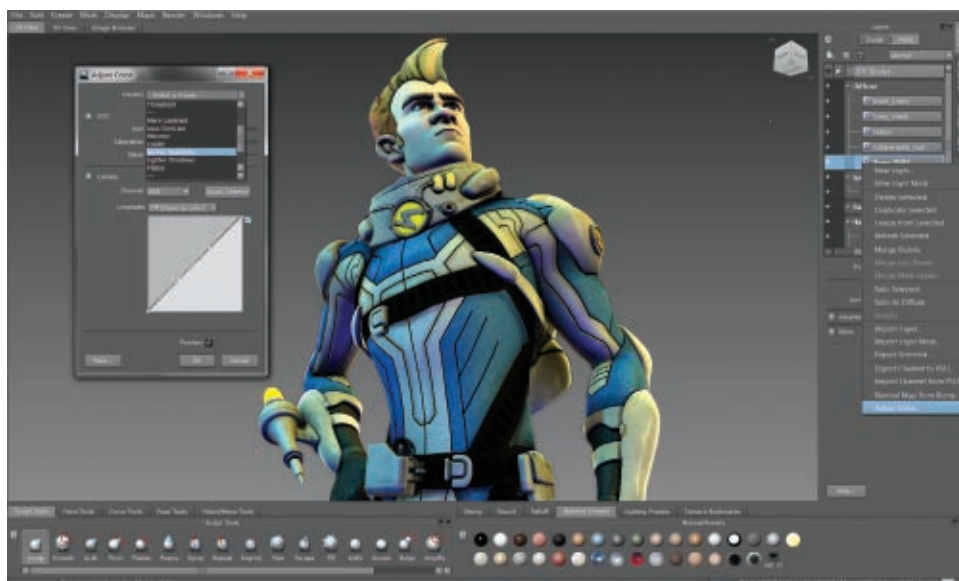
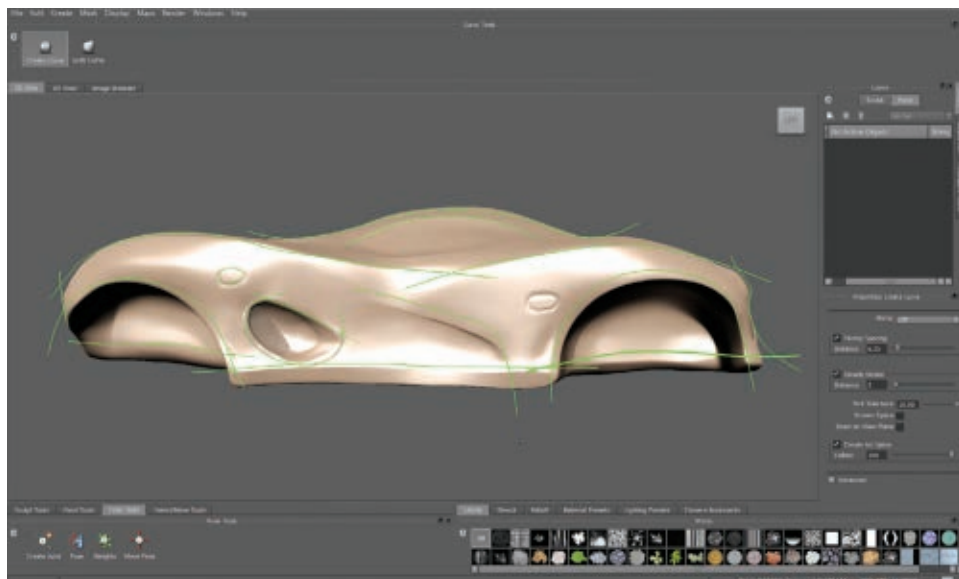
Mudbox 2013 also adds a new Tiling Plane feature to its painting workflow, so you can easily sculpt or paint a seamless repeating pattern, like the metal hull of spaceship. Though at first it takes a bit of practice, this new tool makes it easy to quickly cover a large surface with details. You can use it for sculpting, too, and extract the tiled sculpture data as different types of displacement maps.

The profile of a sculpting or painting tool is determined by its falloff curve, so the more control you have over the shape of the curve, the more subtle strokes you can make. Thankfully, you can edit and adjust your falloff curves in the newly expanded falloff curve editor by adding and deleting points and adjusting a point's control handles.

Better texture management  
Everyone wants high-resolution textures, but creating high-resolution textures takes precious GPU processing cycles. Autodesk first introduced the "Gigatexel Engine" in Mudbox 2012, which made the painting workflow operate more smoothly by automatically loading and unloading texture tiles depending on what the artist could see onscreen at any given moment. In Mudbox 2013, the Gigatexel Engine has been optimized to handle more data, and the artist has finer control over how much of the GPU is dedicated to the Gigatexel Engine.

Mudbox 2013 also includes a new Adjust Color layer option which allows users to make global color changes to a paint layer—kind of similar to the Hue/Saturation function in Photoshop—by using presets or tweaking HSV sliders and color channel falloff curves. The bottom line is that with Adjust Colors it is possible to significantly alter a texture's hue, contrast, and gamma all inside Mudbox.

There are a few more new texture-management tweaks worth



**TOP:** The new Curve tools allow artists to draw freeform curves on the surface of meshes for use as sculpting or painting guides. The new curves are different from the original curve presets. **BOTTOM:** One of the features that makes Mudbox popular is its layered, texture-based 3D painting system. In this release Mudbox ships with better Photoshop interoperability, support for 16-bit Photoshop files, and a new paint layer Adjust Color option similar to Photoshop's Hue/Saturation.

noting; you can now combine painted bump maps and sculpted details from normal maps into a single map, meaning you can combine both sculpting and painting to produce relief details and export the results to a game engine; and you can now apply PTex files as displacement maps through the Sculpt Using Map option. Odds are, neither of these features are going to justify the price of the upgrade, but they're a nice bonus.

#### SHOULD YOU UPGRADE?

» Mudbox 2013 has a better interface, optimized texture management, and a few much-needed updates to the sculpting and painting tools—but it's just not different enough from Mudbox 2012 to justify the upgrade price for most artists. The major exception to this is artists who are regularly working with the rest of the Autodesk suite (Maya and 3ds Max, in particular), since the improved interoperability

between Mudbox 2013 and other applications will save enough headaches to justify the upgrade cost. However, artists using Mudbox along with Cinema 4D, Houdini, modo, or other 3D applications can probably skip this one. [👉](#)

**MIKE DE LA FLOR** is a freelance medical illustrator, animator, instructor and writer. He's the author of *The Digital Biomedical Illustration Handbook* and other CG books.



# RECORDING UNITY

## BUILDING A REPLAY SYSTEM IN UNITY

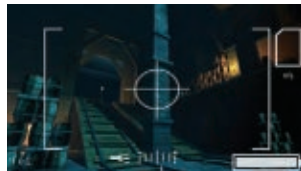
Replaying player actions in a game is a common feature used in match replays, cutscenes, “ghost” players in racing games, and a variety of puzzles (like those for Clank in RATCHET & CLANK: A CRACK IN TIME). Many engines support at least a limited ability to record action in-game, as opposed to recording a movie (as with Source’s Demo command). Unity, however, does not. One of my more recent projects was heavily reliant on character replays. I had spent too much time writing custom replay systems in the past to want to go down that road again, so I decided to roll my own generic, adaptable record/replay system for Unity. It’s called InputVCR, and you can check out the source on Github (<https://github.com/EddieCameron/InputVCR>), or see an example of an older version on the Unity Asset Store (<http://u3d.as/content/eddie-cameron/input-vcr/2Mx>). Give it a look while you’re reading this article and you can play along!

### PAST EXPERIENCES

» I first needed to record something in Unity while Robert Yang and I were making LO&VE (<http://grapefruitgames.com/2011/06/14/made-love-successfully/>) for a game jam at Babycastles, an indie-games space in New York City. LO&VE is an arcade-style game with novel controls, so we needed some sort of attract mode tutorial. I tried recording a movie with Fraps or something, but getting a sufficiently high-quality capture killed the frame rate while we were

recording, and we didn’t want to tack a 40MB video on to a 3MB web player game. In the end, I decided to just take down the positions of the two players in each frame, and snap them to the recorded positions during replays. This method was simple, and as long as you didn’t look too closely or have a really slow or fast frame rate, it was pretty robust—good enough for a game jam, anyway.

Robert and I also started work on MUCKRAKER (<http://grapefruitgames.com/2011/08/07/enforced-hiatus/>), a game about

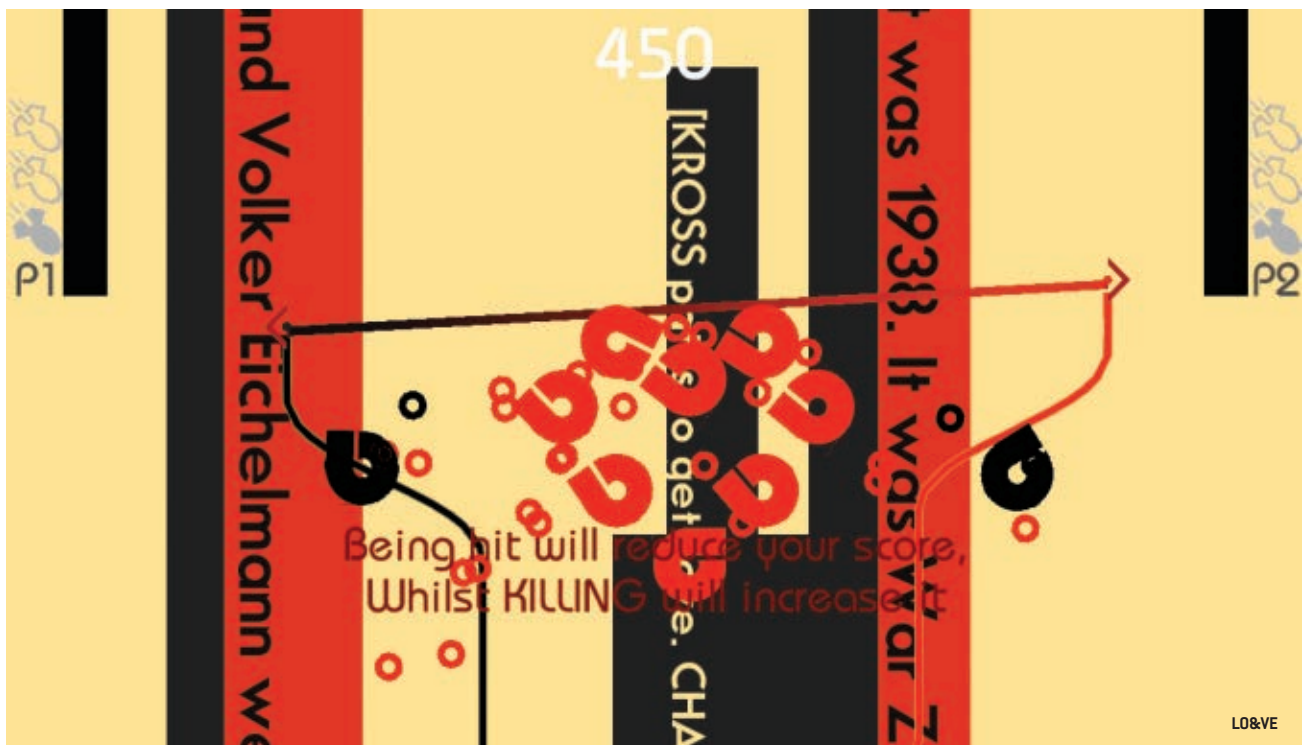


MUCKRAKER

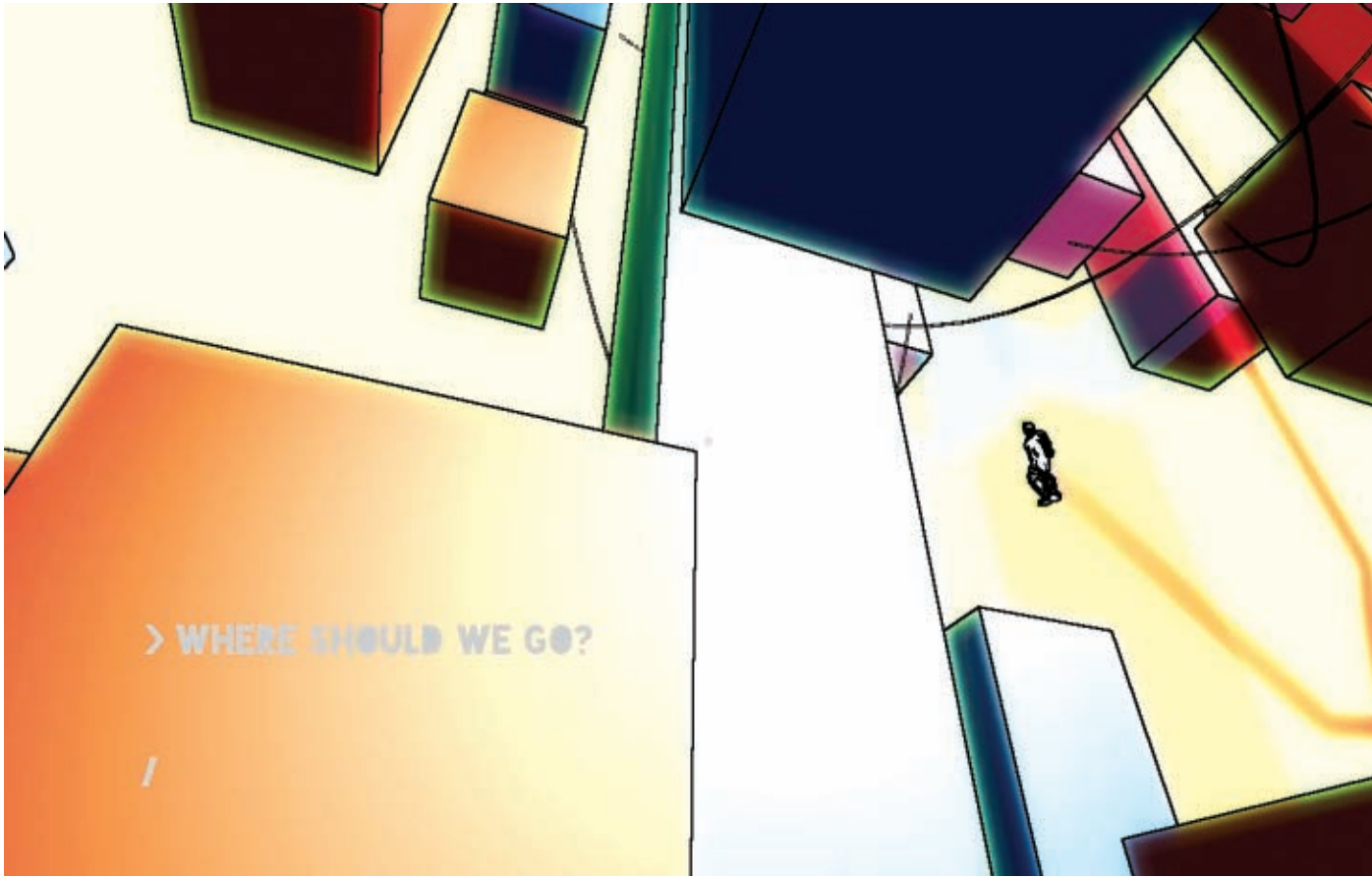
investigative journalism. It involved “filming” certain events, replaying them, and editing the replay. This time, I quickly rigged up a system where you would drop a “recordable” script on each game object you wanted to record.

When the player started filming, a master recorder would take down the position and rotation of each recordable object within the camera’s view. Our recordings were all short, and we never got too far into production, so this system worked well enough, but it was unwieldy, completely frame-rate reliant, and inflexible. I couldn’t have two different recordings at once, and couldn’t easily take down any other events.

By now, everything looked rather dire. The next time, I vowed to do things right.



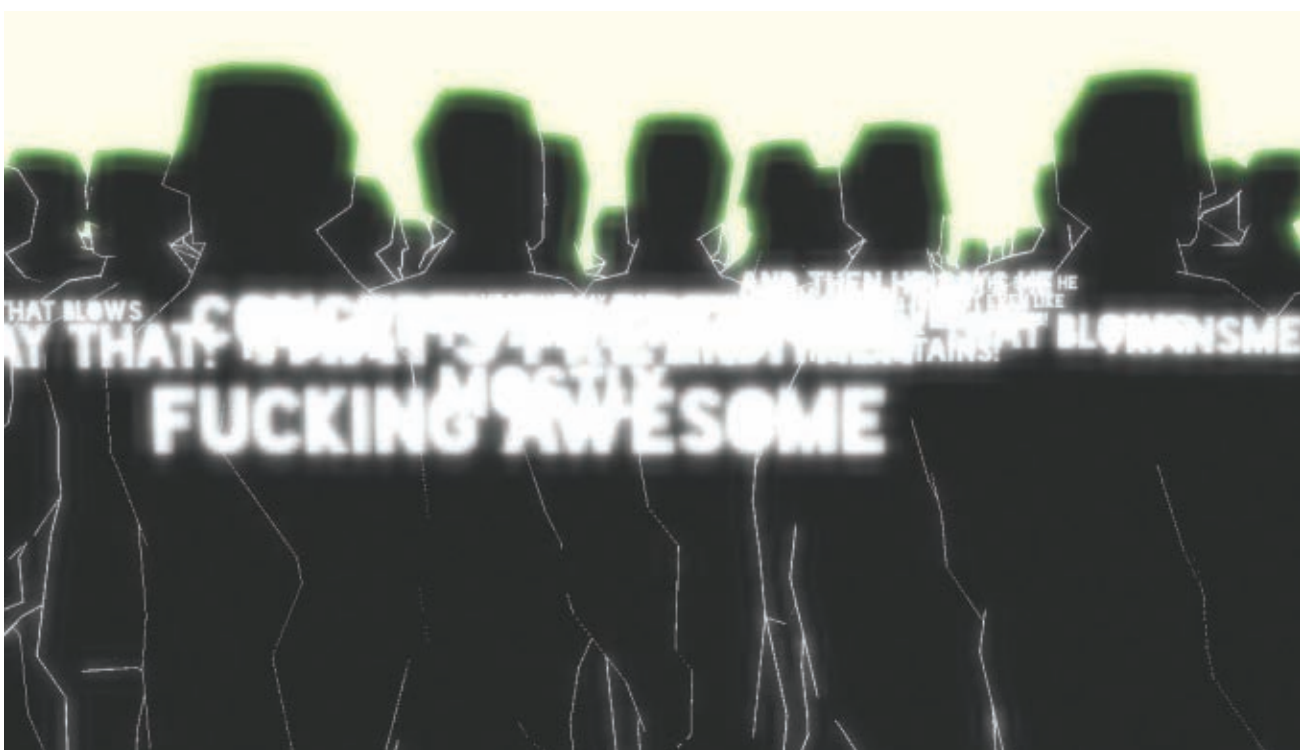




> WHERE SHOULD WE GO?

/

NO ARCHITECT



THAT BLOWS

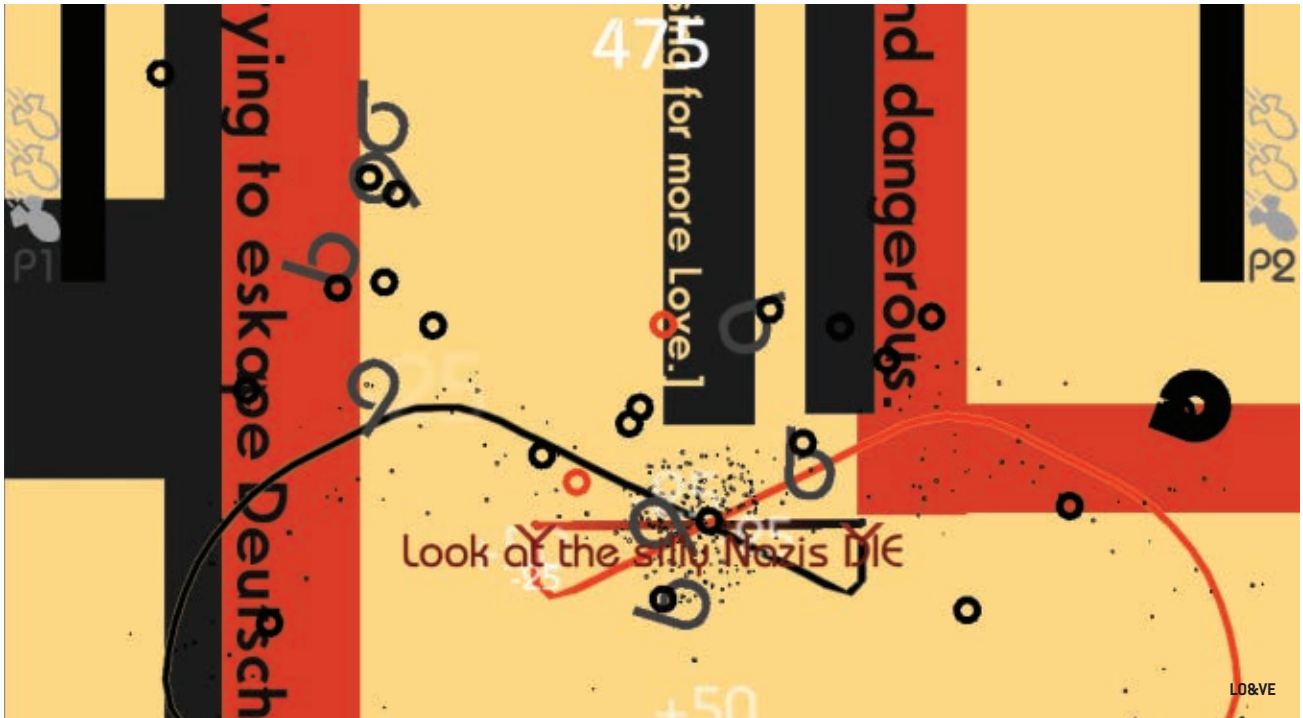
AY THAT?

FUCKING AWESOME

AND THEN HE SAYS 'SHE HE'

WHAT BLOWNSME

MOSTLY



**INTRODUCING INPUTVCR**

» More recently, I started work on NO ARCHITECT, a cooperative FPS platformer/short story/dull-party adventure. I wanted the co-op part to be available offline as well, with some way of including the feeling of having live players, so I decided to build a system that would let players share recorded play-throughs with each other. This way, people could still play with

me the flexibility to mess with the playback. If, say, a wall appeared in front of a recorded agent, that recorded agent would get stuck against it instead of clipping through.

I set up a basic script (InputVCR.cs—see Listing 1 for those of you playing along at home) where the user chooses which named buttons/axes to record from those already set up in Unity's input manager. For

doesn't have to know whether it's getting live input or not, and a VCR can record, pause, rewind, or even swap recordings (VHS tapes) with other VCRs, all without having to care about what uses its output.

This method is pretty simple, and except for replacing static calls to Input (like Input.GetMouseButtonDown()) with a reference to an instance of InputVCR, you don't have to change existing scripts.

I could make sure recording or playback was done during FixedUpdate (where Unity does physics calculations at regular intervals), but this was too restrictive. The default fixed timestep is 25fps, which is too slow for fine input. Instead, I added timecodes for each frame, so that playback could slow/speed up if the recording frame times didn't match the playback times. This helped somewhat, but if there was a spike during recording or playback, the character would still freak out.

More recently, I tried to improve InputVCR by putting playback on a separate thread with a set frame rate. Unfortunately, Unity doesn't play nice with other threads, so playback kept pausing. I got around the multithreading problem by just checking for input changes in the recording every frame, looking backward through the recording if necessary (see Listing 2 for an excerpt from the main playback loop.)

Even so, rounding errors and slight physics simulation differences mean that a recording based off input alone will always drift over time. For short recordings, or those less reliant on physics calculations, this might be

▼▼ In the end, I decided to just take down the positions of the two players in each frame, and snap them to the recorded positions during replays. This method was simple, and as long as you didn't look too closely or have a really slow or fast frame rate, it was pretty robust—good enough for a game jam, anyway. ▼▼

each other even if there weren't enough players online to keep a live multiplayer session going.

This time, I decided to try only recording inputs, and dealing with the expected inaccuracy inherent in this method at runtime. I was worried that syncing the position and rotation of an object every frame would take up a lot of bandwidth, which was precious in this web-based game. On the other hand, this recording method gives

each frame, the VCR records the status of these inputs, along with the position of the mouse. Now, you can give this recording to any InputVCR, and it will spit out the same inputs in the same frames.

Let's painfully stretch the VCR metaphor for a second: Say the user's input is the signal from an aerial. You can plug this input into the VCR so gameobjects (TVs) can get their inputs from it rather than the direct user feed. A gameobject

At this point, I had a pretty solid yet flexible model, but the playback quality still wasn't up to scratch; for two-minute recordings, the characters would slowly drift off course because the difference between the recording frame rate and the playback frame rate started to add up. Considering we wanted to use this for a game about making long jumps onto small platforms, we couldn't allow for a whole lot of wiggle room.



### LISTING 1: THE BASIC INPUTVCR SCRIPT

```
public class RecordingFrame
{
    public float recordTime;
    public List<InputInfo> inputs = new List<InputInfo>();
    public List<FrameProperty> syncedProperties = new
    List<FrameProperty>();
}

[System.Serializable]
public class InputInfo
// represents state of certain input in one frame.
// Has to be class for inspector to serialize
{
    public string inputName; // from InputManager
    public bool isAxis;

    [HideInInspector]
    public int mouseButtonNum = -1;
    // only positive if is mouse button

    [HideInInspector]
    public bool buttonState;

    [HideInInspector]
    public float axisValue; // not raw value
}

public struct FrameProperty
{
    public string name;
    public string property;
}
```

enough, but I really needed to keep things on course.

I also added the ability to record the position and rotation with any given frame, along with arbitrary information, such as button presses or enemies spawning. You can sync every second or so, and snap (or interpolate smoothly) back to the desired location if the playback goes off kilter. Unfortunately, I was playing recordings in a changing environment, and still needed the real-time character to override the recording if a platform disappeared from under their feet. But parsing all that extra information was becoming unpleasantly slow, too (it actually turned out to be because of a bug in Mono with `StringReader.ReadLine()`—you can find more about it here: <http://lists.ximian.com/pipermail/mono-bugs/2009-November/095088.html>). So instead I chose to record what platform the

character was meant to be on, and synced the location each time the playback landed on the “correct” platform. If a platform was missing during playback, the controller wouldn’t sync and would be allowed to fall, as expected. There were still some inconsistencies during frame-rate spikes, but at least they wouldn’t kill the character unintentionally.

### REINVENTING THE VCR

» Recording stuff is a pain! And, unfortunately, there is no all-knowing solution for every game. Syncing position/rotation of an object is guaranteed to be accurate, but the playback is stuck on rails and a changing world won’t affect it. Recording inputs allows for interactive playback, but in many cases won’t be accurate over time.

I found the best solution is to start with a recording of the input

### LISTING 2: AN EXCERPT FROM THE MAIN PLAYBACK LOOP

```
// Playback
void Update()
{
    // go through all changes in recorded input since last
    // frame
    var changedInputs = new Dictionary<string,
    InputInfo>();
    for( int frame = lastFrame + 1; frame <= currentFrame;
    frame++ )
    {
        foreach( InputInfo input in currentRecording.
        GetInputs ( frame ) )
        {
            // thisFrameInputs only updated once per game
            // frame, so all changes, no matter how brief,
            // will be marked
            // if button has changed
            if ( !thisFrameInputs.ContainsKey ( input.
            inputName ) || !thisFrameInputs[input.
            inputName].Equals( input ) )
            {
                if ( changedInputs.ContainsKey ( input.
                inputName ) )
                    changedInputs[input.inputName] = input;
                // changed input holds most recent (&
                // still different to last frame) value
            }
            else
                changedInputs.Add( input.inputName, input );
        }
    }

    // update input to be used this frame
    foreach( KeyValuePair<string, InputInfo> changedInput
    in changedInputs )
    {
        if ( thisFrameInputs.ContainsKey ( changedInput.
        Key ) )
            thisFrameInputs[changedInput.Key] = changedInput.
            Value;
        else
            thisFrameInputs.Add ( changedInput.Key,
            changedInput.Value );
    }

    playbackTime += Time.deltaTime;
}
```

only, and add syncing until the playback is accurate enough for you. If there are events that rely on the playback, you can record them separately from the input to ensure they still happen. `InputVCR` still does the heavy lifting when dealing with recordings, but I’ve had to accept that you’ll always need a decent amount of custom code to get the results you need. What is the main purpose of your playback? Timing accuracy? Scripting events? Interacting with a

changing world? Focus on and build this into the framework first, and the rest will follow.

Now that you’ve read the article and seen the source in Github, send me your feedback! I’m still improving the code as I use it in different projects, and if you have any ideas or improvements, I’m interested in folding them in. 🙌

---

**EDDIE CAMERON** is a Unity 3D specialist. Find him on Twitter at [@eddiecameron](https://twitter.com/eddiecameron) or email him at [eddiec@grapefruitgames.com](mailto:eddiec@grapefruitgames.com).

**INFORMING, ENGAGING, AND  
EMPOWERING THE INDUSTRY**



**gamasutra.com**

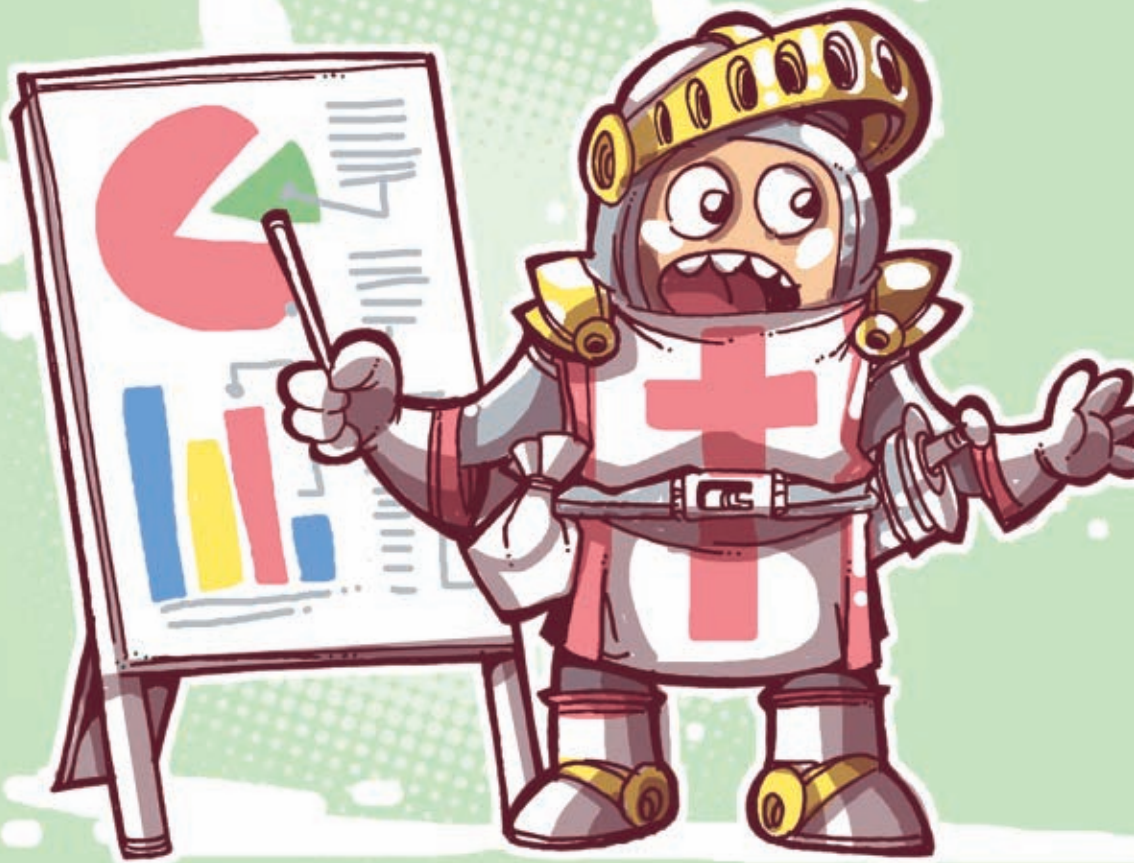
the art and business of making games





# CHEEK BY JOWL

ARTIST-DESIGNER DIPLOMACY THROUGH COLOCATION



When you see people who are basically in agreement—people whose differences are completely invisible to anybody outside their little private conflicts—utterly unable to communicate or cooperate effectively with each other, it makes you wonder about the future. And of course, it's even worse when you get caught in the crossfire between hostile tribes that don't (or won't) try to understand each other's point of view. Like, say, game designers and game artists.

Recently, we've started seeing more studios start structuring office layouts with interdepartmental groups in mind ("pods," or "strike teams," or what have you). Colocating art and design departments can certainly encourage the two to collaborate, but only if you do it right—and at the right time.

## ETERNAL CONFLICT

» Unless you're a one-man band, you work with others—which means you know that development can be pretty contentious. While there are plenty of legendary artist-programmer culture clashes in the game-industry canon, most of them boil down to a straightforward negotiation over resources: "How much memory can I have for that?", "Can't you add a slider that lets

me control this?", or "Hey, what happened to the Doritos?" Artist culture and programmer culture are so different that each side is almost obliged to defer to the authority of the other when in the other's own mysterious domain ("How the Other Half Lives," *Game Developer*, January 2005).

When artists and designers clash, however, things are more nuanced. Artists and designers share

responsibility for the emotional context and atmosphere of a game, which means it's a lot harder for either side to mark out exclusive ownership of its own turf. The design department might not care how much memory you have to spend on lightmaps or what kinds of shaders you like, but they are going to be very unhappy if your color choices and lighting style undercut the overall mood of the story. You, on the other hand,

might not care how many points of damage the X-X-down-back combo delivers to the Ice Ninjas—but you do know how painful it will be to squeeze that animation into seven measly frames to satisfy the Shoryuken.com crowd. Because art and design are so interdependent, each discipline can easily make life difficult for the other.

To complicate matters further, designers and artists are frequently in competition for



the player's attention. Designers have to be able to teach players difficult and complex skills. To do this, they need to present relevant information clearly and explicitly. Artists, on the other hand, prize nuance and mystery, happy accidents and visual resonances that don't necessarily lead up to anything more than a striking look or distinctive atmosphere. Obviously this leads to conflict: In my first game back in 1996, the art team created a lovingly detailed, environmentally appropriate set of camouflage patterns for our army of giant mechs—only to have the designers step in and turn them bright red and blue to aid in team identification.

ILLUSTRATION BY JUAN RAMIREZ

### SIDE BY SIDE

» The single best way to improve art-design collaboration is to make sure that artists and designers are in constant, daily contact—which means that instead of having them living and working in distant departmental silos and communicating via spreadsheets and design docs, they should sit and work side by side. After all, the Latin roots of “collaboration” translate literally to “working alongside.”

“Strike teams” or “pods” where members of different disciplines sit together to work on common projects have become increasingly fashionable over the last few years: Bungie's rolling desks and Valve's cabal system are the most famous,

but hardly the only examples of the trend. The MBA-speak moniker is “colocation” or “tight matrix management,” which sounds vaguely kinky and presumably involves shiny trench coats.

Call it what you will, it's obvious why closer cooperation between the disciplines (and especially between art and design) has become popular: It's far easier to work out a problem by turning your swivel chair around and asking the person behind you than by firing emails into the ether or trying to get two sets of leads to schedule a meeting (and let's not even talk about those Kafkaesque “let's schedule some time to sit down and talk about your meeting request” situations).

Better communication is not the only benefit of colocation, though. Interdisciplinary collaboration helps educate both artists and designers about the costs and benefits of different choices. When you're brainstorming with your departmental peers, it's all too easy to make a decision that creates a lot of work for those anonymous “resources” sitting down the hallway, or one that suddenly axes several months of completed work with nary a regret. Of course, even in the most cabalistic of environments, directions change and ideas don't work out: New work will still come up, and old work will sometimes get cut. But when artists are included in the decision-making, those



course corrections can be much less disruptive for everyone involved.

What's more, those disruptive changes might not happen quite so often when artists and designers are in constant contact. A diverse strike team includes more perspectives and different skill sets, helping to spot problems that might be missed by more homogeneous groups. Alone with a whiteboard, the design team can easily come up with bullet points describing a new creature type that seems to make perfect verbal sense—but which don't translate well into practical models or animations. If the design team had included the artists in charge of building said creature type early on in the discussion, they could potentially avoid that kind of blind spot. Conversely, the designer at the next desk might be able to warn you that those cool-looking shoulder pads aren't going to work now that all enemies are supposed to be able to climb ladders.

(As an aside: What is it with all the shoulder pads in this business? This is what happens when you build an industry on the backs of WARHAMMER 40K nerds.)

The final reason designers and artists should interact more is to improve iteration time. In an interactive medium like ours, nobody can predict with precision how a level will play or how well a character will mesh with the environment. Iterative development, where levels and characters evolve from concept to completion with many reality checks along the way, is a topic we've covered before ("Raw Crude," *Game Developer*, May 2008), but it's an evergreen issue in making games.

Say, for instance, there's a push to add a new move to the game. In old-school development, the move is designed in a whiteboard meeting and written up in a doc. From there, a list of tasks gets drawn up and dispatched to legions of idle animators and effects specialists who start beavering away—not just on the new animation, but on the scads of transitions, variants, hit reactions, effects, and so on that will make

up the finished movement and all its permutations. Of course, there's no guarantee that this feature is actually fun—it may be unbalancing, or too hard for players, or simply irrelevant to the rest of the game. All of those keyframes, pixels, and shaders that were created to support it have to be junked. Recriminations abound.

In a more iterative strike-team process, that move would get roughed in quickly. The artists and designers would work together to get enough of the feature set in place that the whole thing can be evaluated for its impact on the game before the whole cumbersome spreadsheet of glue animations and effects needs to be created. Feedback from both

attractive play space with lots of gameplay. Your average poly-cruncher, however, will quietly doze off once the conversation starts turning on the proper way to dampen feedbacks in the in-game economy. Conversely, a designer can provide invaluable feedback to an animator trying to hone a set of dodge animations, but isn't going to have much useful advice when it comes to cleaning up 16 hours of raw mocap.

Sometimes you just need to huddle with people who understand your daily problems. If your studio's implementation of a small-team setup makes that difficult, some aspects of your work will suffer. If artists can't get together to bitch about their tools, the tools

those concepts, there's something to be said for the old-fashioned departmental assembly-line model. Balancing workloads, sharing resources, and making sure that bugs get fixed are all easier to do when artists spend most of their time in the departmental embrace.

You also have to remember that not everybody is well suited to the give-and-take of working outside his or her disciplinary comfort zone. Even though colocating artists and designers can help fight interdepartmental friction, it's not all flowers and rainbows (unless you've got your Pyrovision goggles on). Constant discussion and negotiation can be stressful, particularly for artists who are more visual than verbal.

## ❗ You can't forget that some parts of game development aren't exercises in nimble, iterative exploration and rapid prototyping of new experiences. Sometimes it's just a goddamn slog. ❗

sides can improve the prototype version so that unneeded extras can be dropped—or new avenues opened up.

Of course, this kind of iterative approach could, theoretically, be done by a traditional departmental organization too—but it's harder. Developers of every discipline have a hard time working fast and loose. The urge to dot every "i" and cross every "t" is hard to resist—and it's much harder to resist when nobody is sitting right next to you, asking when you'll be ready for the next go-round. The fast progress and instant feedback that both artists and designers get from working directly with each other provide a good incentive to stick with the prototyping process and not start polishing prematurely.

### PERSONAL SPACE

➤ Cabals, pods, and strike teams are popular for good reason, but seating arrangements and luncheon pairings don't automatically turn every developer into an agile development powerhouse.

For one thing, there are some tasks or topics that need to be kept in the family. Designers and environment artists can collaborate very effectively to create a fun,

won't improve. If designers can't brainstorm without artists telling them their ideas will take too long to build, they may miss out on some brilliant innovations.

You can't forget that some parts of game development aren't exercises in nimble, iterative exploration and rapid prototyping of new experiences. Sometimes it's just a goddamn slog. You can prototype and cabal till the cows come home, but those 382 breakable objects on your task list are still waiting to be built, textured, and rigged up. When you've got a long list of deliverables and the clock is ticking, you might find that you'd appreciate a little more time with fellow artists who could take a few of those tasks off your plate. You might also find that your appetite for interacting with anybody of any discipline is taking second place to your desire to hit your milestones and get some sleep.

Games have lifecycles, and different kinds of work arrangements may suit the different epochs in the life of a product. Collaborative pods and cabals are unbeatable for experimentation, prototyping, and proof-of-concept work. Once your team has proven those concepts, and clearly sees how to build out

Many artists don't really want to participate in design discussions—they'd prefer to power through a stack of assets and then go home. Lots of designers, too, would rather put on their headphones and write scripts than debate the proper timing of a jump animation. Some people prefer to work heads-down, and that's just hard to do in a strictly pod-based small group environment. A smart studio won't try to shoehorn people into roles they just can't fill.

Organizational fads come and go; nothing ages faster than a management buzzword. Simply getting people to work together effectively is just pretty hard, as most veterans of our industry can attest. That said, there really is no better way to get people working together, especially when they're coming from very different perspectives, than to get them working, literally, together. 🎧

---

**STEVE THEODORE** has been pushing pixels for more than a dozen years. His credits include *MECH COMMANDER*, *HALF-LIFE*, *TEAM FORTRESS*, *COUNTER-STRIKE*, and *HALO 3*. He's been a modeler, animator, and technical artist, as well as a frequent speaker at industry conferences. He's currently the technical art director at Seattle's *Undead Labs*.



# THE 4 FS OF GAME DESIGN

## FAIL FASTER, AND FOLLOW THE FUN

In 2001, I attended a workshop at GDC taught by one Marc “MAHK” LeBlanc. Over the course of those two days, I would be exposed to a great deal of development wisdom (and if you haven’t attended one of Marc’s sessions, what the hell have you been doing? Go, people! Go!), but there were five words in particular that Marc shoved down our collective throats that have completely altered the course of my career.

### FAIL FASTER. FOLLOW THE FUN.

» Those words are a complete methodology for how to not be eaten by snakes, and perhaps succeed, when pushing into the wilderness of the New.

Marc repeated the phrase endlessly, until there was absolutely no chance that anyone in that room would ever forget them. Here, I’m going to continue the cycle of abuse, and do my damndest to inflict the same brain wound on as many readers

“4F” method does not necessarily apply to everything.

What we will be discussing is most useful when you are exploring the New. Though failing faster does have some application when making small iterations on a known design, you can actually succeed without doing it if you have a strong template in front of you (heresy!).

### HOWEVER.

» If you are making something new from scratch that you have

### FAIL EARLY, FAIL OFTEN

» We game developers hate this word, generally.

Oh, sure, if we are sitting around a table sipping the mind-altering beverage of our choice amongst our industry comrades, we will happily extol the virtues of rapid prototyping, praise Valve for its highly iterative approach, and shake our heads in horror at the tales of They Who Just Do Not Get It.

Go ahead, try it for yourself: Walk up to your scrum leader or manager

my day-to-day work. Our highly pressurized game development culture has precious little room for processes that make things that aren’t fun. Solutions, people! We need solutions! Winter is coming!

But... we do know about this one, don’t we? The first time a new design is put into play, there is a (scientifically verified) 99.97% chance that it will suck. Anyone building new features must account for this in their process. This doesn’t happen because we’re dumb. It happens because designing fun games is a hard thing to do.

### GAME DESIGN: NOT ROCKET SCIENCE

» I had an executive producer back in ye olden days who was fond of the phrase “It’s not rocket science, people!” When we would hit something seriously tricky, and were up against the wall, he would say this, and we would all laugh, reminded that what we were doing wasn’t so hard after all (I mean, rocket science, right?! Come on!).

I’ve decided since then that he was completely ass-over-teakettle wrong. Now, he might have been right way back in the 1990s, when video games were built out of sticks and glue, but in the current era that sentiment is so horribly outdated as to be dangerous.

I propose to you that the underlying technology that drives most interactive systems is at the very least as complicated as the systems that put rockets into space. On top of that, the goal of most of this technology is to elicit human emotion, a task that has consistently eluded casual reproduction for thousands of years (and still defies most of our attempts to codify it).

Designing a fun game is hard. Much harder, in fact, than rocket science. (When John Carmack needs a relaxing break from game development, you know what he



of this magazine as I can. It worked on me, so maybe it will work on you.

So, full credit goes to Marc LeBlanc. Also, fail faster, people. Fail faster! And, follow the goddamn fun.

### SCOPE: WHEN TO APPLY THE 4F METHOD

» Before the brain wounding, though, let’s talk about limits. This

no reasonable guide for, then your situation is somewhat more dire. In the end, all of the success you will have in “finding the functional New” will come from the moments when you first fail (and you will fail; in fact, that’s the next step), quickly recover, and then follow what shards of fun you may have managed to discover. Everything else will be a waste of time and resources.

or whatever and tell her that you have just failed. Big time. You built something that totally sucks.

Did she smile? Pat you on the back? Congratulate you?

If so, then awesome! You can stop reading now if you want. (In fact, you are probably on my team. Get back to work!)

Myself, I’ve had to fight mightily to make Marc’s words a reality in





ILLUSTRATION BY JUAN RAMIREZ

does? Yep, rocket science.) It's hard enough, in fact, that exactly zero game developers can get it right the first time, every time. None. So. Embrace failure. Because it is inevitable. So really, you might as well.

#### **BE WILLING TO FAIL PUBLICLY**

» We know that making games is hard, and we know that rapid iteration is the cure, so why don't

more teams embrace it? Simple: Very few people want to look dumb in front of their peers.

This is the heart of the matter. This, right here, is big enough to effectively be the entire problem. I'm not kidding. If you can get past this one, much of the rest of the 4F approach follows naturally.

When I started my current project, I printed a big sign that

said "FAIL FASTER" and hung it over my desk. I think my team initially thought I was literally insane. From the outside, it looked like the act of someone who wanted to end his career. "You're... not actually going to leave that there, are you?" someone asked me, I think a tad fearful that my madness might reflect poorly on him. And reasonably so! I was nervous when I put it up there!

[Now, of course, many months later, when I am frustrated that our current review wasn't amazing, they point at that sign. "Well, at least we failed fast!" they say, lobotomizing my concerns. I love them for this.]

Anyone with a resume to protect will naturally be suspicious of being asked to fail. If you want to try to implement a successful



iteration culture on your team, you ignore the natural terror of failing publicly at your peril.

## FAIL FAST II: FAIL FASTER

» The bright side of failure, when accompanied by fair, critical evaluation as to the reasons behind it, is that it is the most certain road toward success with the New. But how many iterations does it take? Yeah, we don't know. So, the solution is to make your failures as fast as possible.

Paper prototyping is a pure incarnation of the 4F approach. Even so, I've seen people polish the crap out of their paper prototypes before testing them, and then discovering that their laminated, be-graphicked cards are terribly unfun. "Don't polish a turd," Marc LeBlanc told us during his course. "It's a dirty job, and in the end all you have is a shiny turd." The trick to escaping this trap is to develop an instinct for the shortest-possible road to prove a design idea, and a willingness (nay, an urge!) to share your work well before it is "done."

Years ago, I was directing a team working on a shootery-thing, and we needed some new gameplay. (It was, in fact, my first real-world opportunity to try to put the Fail-Faster-Fun-Following system into play.)

I told my designers that we were going to do rapid level prototyping. I'm not sure what they thought I meant by "rapid," but when I explained that I expected them to build fully functional levels in one-day cycles, their faces turned gray. Yet we still met each day at 4 p.m., and played each other's work.

Day one was a disaster. Only two guys had levels (and I was one of those two), and none of what we had was fun. They were merely "professionally executed demonstrations of existing mechanics" that lacked love. But we were brainstorming; no criticisms or name-calling were allowed. The people who didn't bring levels to the meeting were not allowed to speak. If you wanted to share your opinion, you had to bring a level.

The next day, we had levels from everyone. Still nothing really

fun, but all the designers presented their work with something more resembling a competitive spirit. My guys were not dumb (as is the case with most game developers, in my experience), and had quickly picked up on the fact that the best designs would simply win the meetings.

So far, we had been failing plenty fast. I was proud! Now we had to find the fun.

## FOLLOWING THE FUN

» The "win the meeting" thing happened the next day. One of our guys got frustrated with the lack of awesome in the first set of daily levels, and designed something I can only call "Rocket-pocalypse." He had put the player up on a platform overlooking an endless enemy charge, and had assigned the action button to fire a metric ton of rockets straight down from the heavens on the field whenever the player so desired.

We played nothing else for the whole session. The team loved it, I loved it. It was really, really fun.

The fact that this mechanic had nothing to do with the actual game was irrelevant. The designer knew that he was making something for the trash bin, but it didn't matter. After all, it only took one day, we were doing research, and most importantly, it was super fun. After that, the "fun target" had been set. "Beat John's Rocket-pocalypse" was now everyone's goal. Whatever they made needed to be at least as fun as that.

This is what is meant by "following the fun." It means, in reviews and in your design planning, only allowing yourself to pursue existing implementations that rock.

Heh. There's a pretty good chance right now that you're thinking, "Well, yeah, we do that all the time. I've heard that before. What's the big deal?"

Over the years, I have worked with hundreds of devs. Again and again and again, I have encountered smart, creative, dedicated, talented people who are producing mediocre gameplay. Why?

Usually, it's because (for reasons of environment, training, experience, constraint, or whatever) they have accepted,

consciously or unconsciously, a fun-quality bar that is lower than what will make the audience want to keep playing. Often, ideas like, "Yeah, it works!" or "It's pretty fun!" are the guiding criteria for success, especially when time is burning.

I call this "foolish optimism." If you played a game, and your response was, "It's pretty fun!" would you recommend it to a friend with enough passion to convince her to buy it?

There's a better criteria, and it is what drives the idea of following the fun. Think about the game you are making. Maybe you're working on a feature, or a level. When you playtest it, do you often find yourself continuing to play long after you validated the feature? Just because it's fun? Do other people do this?

If not (and if we're talking about a key feature that you need to be fun for your game to work), then you probably haven't found the fun.

Revise, redesign, or cut that feature.

This is hard to do. In fact, if it isn't hard, then that's a sign that you're not moving your quality bar high enough. Following the fun requires a commitment to actual fun.

Of course, that doesn't mean that everyone's iterations need to be perfect. Learning to see through the imperfections into the underlying fun in a demonstration is one of the most important skills a game developer can develop. (So get started on that.)

## IN THE FIELD

» I used the 4F process during the production of RED STEEL 2, and the results were unanimously positive. We had the daunting problem of solving first-person melee combat (with guns) with the Wii MotionPlus, without any prior successful examples to work from. We went through whole new approaches to combat at a stunning rate, sometimes prototyping complete combat systems in a single week, learning what sucked, and moving on. Using this approach, we designed a one-of-a-kind first-person motion-control sword-and-gun combat system, and we did it in nine

months. IGN called the game "one of the top titles on the Wii." I credit the team's embracing of the Fail Faster philosophy with that success. (I wish it had sold more, but I suppose you can't have everything.)

In my current project, I have a relatively small team working



on some new stuff. I have applied, almost to the letter, the same process that I described to you above: 24-hour test cycles of whatever-in-the-hell-you-can-come-up-with-as-fast-as-possible, and instilling in the team the belief that I will never crucify them for failing, that the only sin is the sin of failing to share your failures quickly enough. They have embraced it, and the speed with which we have been making progress exploring the New in our chosen category is inspiring to behold.

## GAME JAM SESSIONS

» One last thing: If you've been involved in a game jam, and have made it this far in the article, you may have noticed that the Fail Faster approach could be described as a game jam with a producer.

Yes. Yes, it is. That is how I make games these days. And I hope you'll join me, because we need more and better New, now more than ever. ☺

**JASON VANDENBERGHE** is a creative director at Ubisoft, which he has to admit doesn't exactly suck. You can read his intermittent blog and various scribbles at [www.darklorde.com](http://www.darklorde.com). He can be reached by email at [jason.vandenberghe@ubisoft.com](mailto:jason.vandenberghe@ubisoft.com).

**MARC "MAHK" LEBLANC'S** work can be digested at [www.8kindsoffun.com](http://www.8kindsoffun.com). And, thanks again, Marc.





# DEATH OF AN AUDIO ENGINE

## REINVENTING THE WHEEL NO LONGER

The death of proprietary game audio engines may well be upon us.

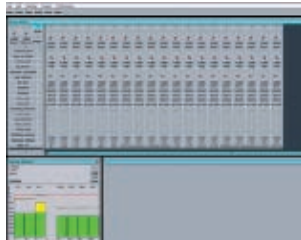
### SWEET DREAMS

» Building your own game audio engine and toolset from scratch was once the pinnacle of achievement; now it has fallen from grace. Games these days need deep feature sets in their audio engines, and it's too expensive to build those engines from the ground up each time—especially when most dev houses need those tools to be easy to use as well. Instead of building powerful, easy-to-use engines in-house, most of us are moving from developing engines and tools, to, well, developing games.

For me, witnessing the decommissioning of tools like LucasArts's iMUSE, Stormfront's adaptive music engine, Radical Entertainment's Audiobuilder, and countless others over the years has been a melancholy process. I am sad, because with their passing goes a knowledge of pipeline and process that could have informed the next generation. But I am also joyous, because the fallen soldiers that battled for new feature sets—like property randomization (volume, pitch, and so on), industry standard names and numbers for decibels or cents, or multiple ways to shuffle a playlist of sounds—have won the war. I am sad to see the loss of these forward-thinking workflows and unique implementations, but they go having left their respective marks on the game audio business.

### TRAVEL THE WORLD

» Audio middleware providers have established a powerful and comprehensive emerging standard in game audio. They did what no coalition could: They have finally given us a common vocabulary to use when we talk about game audio. It hasn't been



Radical Entertainment's Audiobuilder

ratified, nobody voted on it, but whether you like it or not, people are adopting it. And they have been listening and learning and growing to the point where their toy box of tricks has far exceeded the potential of any one studio. Still, the voices of dissent continue to complain that while the box is full of toys, nobody needs them all. That's missing the point—you don't need to use them all, but you can use any tool you like without having to build it first.

Everyone who chose to roll their own audio engine over the past generation is likely facing a pile of mounting problems, after years of heaping designer-requested features onto their antiquated framework. It's no small feat to architect an audio engine that is interdependent on other systems in a way that will endure multiple development cycles, not to mention potential console changes. Furthermore, creating and maintaining the tools necessary to communicate with that audio engine in a way that facilitates a creative process is more than most companies can bear. In a lot of ways, it feels like proprietary audio engine and tool developers are spread too thin. In an age when an audio programmer is seen as a unicorn in the game development landscape, you simply can't rely on this mythical creature to both create (or maintain) an audio engine and toolset any more than you can count on a double rainbow during a clear day.

When “getting the job done” means you have to circumnavigate a toolset's UI by hacking a text file together, something in your pipeline is lacking due to ability, resources, or aptitude. If you don't have the resources to make your tools usable, you didn't think hard enough about how to solve that problem. Accessibility has become the single biggest obstacle blocking creative nontechnical sound designers from actually feeling technically creative. It's not because they don't get it, it's because the tools are in the way.

### WHO AM I TO DISAGREE?

» Don't get me wrong, there are still times when you want to get close to the metal. There are times when building a lookup table in Excel and exporting it to a runtime-readable format is going to beat any labyrinthian spaghetti of nodes or hierarchy hands down. But that has nothing to do with the audio engine or toolset; it is purely an extension that leverages the core competency of what's already there. I think it's this extensibility that makes audio middleware desirable—it's a solid foundation to build a specialized pipeline upon. But look at the upcoming wave of consoles, and then back to your engine: Does your audio engine look ready to make the leap to the next generation of hardware and processing capabilities?

Maybe you tooled your pipeline to be able to live-connect to the game and push values while running to immediately hear changes. You might have even integrated a system for hot-swapping sound banks or loose loading sound files. I would guess that there are few tools locked behind closed doors that can provide the same degree of live-connect functionality we see in today's audio middleware, not to

mention audio debug and profiling. But there is a growing divide in game audio between those who can see what's going on under the hood in-game with sound, and those who can't see the radius through the pixels. Chances are good that if you can't edit volume, pitch, randomization, falloff distance curves, parameter controls, or any number of other values in real time with the game running, you probably can't connect a control surface. If this all sounds kind of sci-fi, well, the future is now.

### EVERYBODY'S LOOKING FOR SOMETHING

» Other creative industries have come to place far less emphasis on rebuilding the wheel. As industry standard toolsets become prevalent, their use as a solid foundation becomes a cornerstone of the development pipeline. Whether you're leveraging scripts and solvers in Maya, manipulating the laws of physics with Havok, or simply exporting spreadsheets formatted for runtime, we creatives all work smarter in the next generation when we can move away from re-solving already-solved problems in the current generation. We must use the technology to enable creativity across a wide array of educational backgrounds, free from the institutional muscle memory of brute-force integration (that is, integration that relies on knowledge of esoteric processes unique to a given workflow), because at the end of the day, it's the talent, not the tools, that makes a great game. 🎧

*“Some of them want to use you, some of them want to be used by you.”—Eurythmics*

DAMIAN KASTBAUER is a hot mess of game audio at *LostChocolateLab.com* and on Twitter @lostlab.



# GAME DEVELOPER MAGAZINE

the best of  
postmortems,  
product reviews,  
and standout  
columns

STUDENT DISCOUNTS  
NOW AVAILABLE.

**SUBSCRIBE TODAY!**  
GDMAG.COM/SUBSCRIBE

DOWNLOAD THE GAME  
DEVELOPER APP  
[bit.ly/gdmag\\_ios](http://bit.ly/gdmag_ios)



# gd



FOLLOW GAME DEVELOPER







# THE 2012 CHANGELOG

## WELL, WHAT HAVE WE LEARNED?

There's a great scene in the dark spy spoof *Burn After Reading*, where the CIA superior played by J.K. Simmons, exasperated at the comedy of errors that has come from his subordinates' handling of the events over the course of the film, seemingly reverts to his "CIA Management 101" basics, leans back and says, "So...what have we learned?" With no helpful reply from his subordinate, he continues, "I guess we learned not to do it again," and adds shortly thereafter, "...I'm fucked if I know what we did."

It's not hard to feel as confused as Simmons's character when looking at the dramatic rate of change in the game industry. Given that this article will be appearing in print at year end, it's useful to look back at the year, at what we were collectively anticipating were the directions for 2012, and ask ourselves what we got wrong and why, in order to see whether we can perhaps learn something about how to anticipate the future going forward.

Here IS a selection of topics that were making the news at the tail end of 2011 [Granted, I've cherry-picked some of the ones people called poorly—but that's the point, isn't it?].

### SOCIAL GAMES GOLD RUSH

» The excitement about the growth in the social games segment carried through 2011, and by the end of the year was certainly at a peak of sorts. It reached a crescendo with Zynga's IPO at the tail end of the year. There certainly was no shortage of bluster leading up to that, though the post-IPO stock performance tempered that somewhat. Fast-forward a year, and things look dramatically different. At the time of this writing, Zynga is bleeding executive talent and laying off staff, and the segment as a whole seems to be suffering from a glut of content, rising user acquisition costs, and a sense of consumer ennui regarding the content.

### FREE-TO-PLAY AS THE BUSINESS MODEL

» Somewhat related to the above, but also buoyed by trends in PC online and mobile, there was a strong sentiment that free-to-play would become the one business model to rule them all. Many iOS titles were transitioning to F2P, and even big players like Microsoft and Blizzard were rumored to be preparing big-budget F2P offerings.

The model certainly has taken off, but claims that it would decimate all other models were premature. Certainly some of the developer

conversations I've had seem to indicate that a traditional one-time purchase model or a hybrid approach may work better for certain types of content, certain audiences, or certain platforms.

### THE CROWD UNDERESTIMATES CROWDFUNDING

» Many had been talking about crowdfunding's potential for games in 2011 and earlier. I wrote a blog post in 2010 talking about game projects raising \$10K-\$20K on Kickstarter, and suggesting that they might climb slightly higher. Wired wrote a piece in early 2012 about some projects raising up to \$50K. Of course, a few weeks later Tim Schafer raised \$3.3M and reset everyone's understanding of what was possible. Today, developer conversations I'm having seem to carry the sentiment that three million seems like the ceiling for a Kickstarter project, but really, I think it's premature to say that the dust has settled on this one.



### THE HTML5 REVOLUTION

» Another topic that frequented headlines heading into 2012 was HTML5. Not only was it touted as technology viable for use in games within browsers on PCs, tablets, and phones, it was also pointed to as what would allow developers to get around the "walled gardens" of platforms where gatekeepers demanded a share of the revenue and restricted access based on content policies. Adding to the fervor were major commitments by companies like Adobe and Facebook. A year later, the technology still shows promise, but numerous companies have backed off their original stance, at least stepping back their timeline for deployment or abandoning the idea indefinitely.

### THE REVOLUTION SHALL BE STREAMED

» There were many instances of overemphasis on things fitting the mold of "the new thing will kill the

old thing," but cloud gaming in particular was being discussed as something that would potentially kill traditional consoles by delivering triple-A experiences to all manner of devices. Looking at the two players making the most headlines at that time, Gaikai was acquired by a console vendor, and OnLive, while still in operation, has laid off most of its staff and sold the rest of the company and IP for a relatively cheap sum. There's no question that cloud-based gaming will serve a role within the industry, but clearly the potential for it to kill off other platforms was exaggerated.

### SO...WHAT HAVE WE LEARNED?

» First, whenever someone claims, "Everything is going to be..." it isn't. As the industry grows more diverse, "business model convergence" is as much a fallacy as hardware convergence was. More platforms and more people playing more games should mean more opportunity to do things in ways that are tuned to the particular customer and their situation. Social games, free-to-play business models, and cloud-gaming delivery models will all take their place alongside what already exists.

Secondly, the speed of technology transition is generally slower than the hype would have you believe. The HTML5 example above is a great example of technology that will have its day *someday*, but not now. Getting products to a customer-ready state is far harder than showing something is technically feasible, even though the latter is what gets the headlines.

Finally, with regard to the crowdfunding point above (though this applies to some degree to points about social games and anything having to do with the scale of the Internet), it is important that we don't limit our thinking. The types of things that are possible at these scales are, well, not yet known. Limiting our thinking only to what has already been achieved is a sure way to be surpassed.

If anything, we've learned that a lot can change in a year, and surely will in the coming year. Stay nimble, diversify efforts where possible, and remember that all industry rules should be questioned. ☹

*KIM PALLISTER works at Intel doing game industry forecasting and requirements planning. When not prepping the world for super-cool hardware, he blogs at [www.kimpallister.com](http://www.kimpallister.com). His views in this column are his and do not reflect those of his employer.*



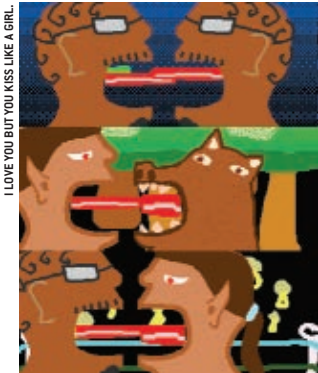
# THE AVANT-GAME

## HOW DO YOU RECAPTURE VINTAGE MAGIC ON MODERN TOUCH DEVICES?

I'm a proponent of diversity in the game industry—that's no secret to anyone who's read my prior columns. This time though, I'm talking about diversity in the games, not the developers. We've only begun to see games tackle interesting subjects, emotions, and genres; games like *FLOWER*, *JOURNEY*, *THE UNFINISHED SWAN*, and *DEAR ESTHER* are experimental, but within mainstream bounds. But what about games for the hardcore weirdos? Do we have something for the Ed Wood aficionados of the game world?

Beyond triple-A, beyond social, and even beyond the realm of the standard indie game, there lies a world of curious, confusing, and confounding computer entertainment—and though they don't often make much money, they show us how incredibly broad and full of potential games can be.

### THE GAMES



I LOVE YOU BUT YOU KISS LIKE A GIRL.

#### I LOVE YOU BUT YOU KISS LIKE A GIRL

[www.glorioustrainwrecks.com/node/1142](http://www.glorioustrainwrecks.com/node/1142)

» This is a jousting game for two players, featuring giant faces with tongues that you can extend and retract as the faces continually advance toward each other. You score hits with a long tongue and block hits with a short stubby tongue as you play a game of cat and mouse, raising and lowering your tongue-lance to knock the other player off the screen. It's an awkward experience for everyone, and thus feels very much like high school, but it is actually a proper competitive eSport wrapped in a ridiculous shell. Play it with a friend and you'll see that an odd premise plus a solid mechanic can yield very solid results.

#### MAGNETIC SHAVING DERBY

<http://nyarlulabs.tumblr.com/games>

» Use a magnet to clumsily guide a razor across your face as quickly

as possible—and avoid those sensitive eye, nose, and lip areas! If you're too slow, hair grows back and you begin anew. A bonus mode has your magnet guiding a space penguin to grab coins and avoid spiky meteors. Ultimately, this is an experiment in delayed control input—while you have direct control over the magnet, it has a delayed effect on the object you actually want to move. This give-and-take is the main mechanic of the game, and it works because you know what you're getting into (the lag is a feature, not a bug!) and fits the game's universe.

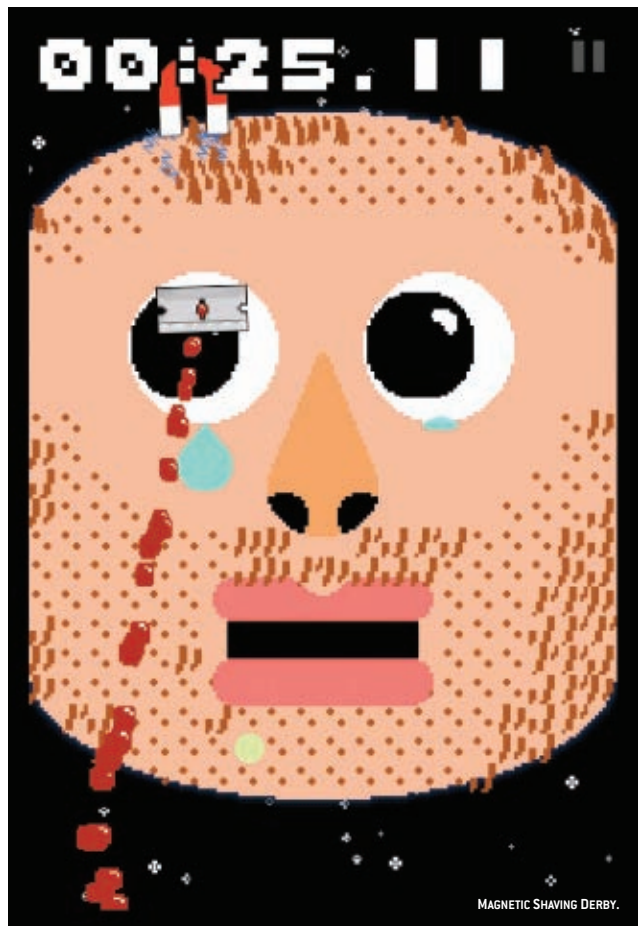


DYS4IA.

#### DYS4IA

[www.newgrounds.com/portal/view/591565](http://www.newgrounds.com/portal/view/591565)

» *DYS4IA* is essentially a series of WarioWare-style mini-games, most of which you can't lose, that tell the story of one person's biological and mental transition through the use of female hormones. What's most striking here is that easy metaphors can become more powerful through interactivity: breaking through a brick wall, for



MAGNETIC SHAVING DERBY.

instance, or having to stealthily hide from others to avoid scrutiny, or being unable to fit into the established shapes you're given. It's a lesson in how, with the right framing, designers can use simple, straightforward gameplay to speak to something much more subtle.

#### ENVIRO-BEAR 2000

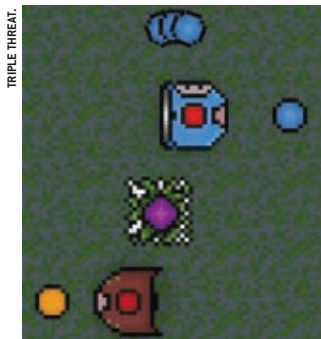
[www.enviro-bear.com](http://www.enviro-bear.com)

» You're a bear in a car, trying to get enough food within five minutes so you can hibernate for the winter. You have to run into animals, fish, and whatever else,





then grab and eat your spoils and get back to your cave. You can only use one hand to shift gears, steer, brake, accelerate, and eat, since your other arm is out the window—because you're a cool bear and don't want to look lame in front of the others. The "narrative" is bizarre and the controls are odder, but ultimately players are frustrated into a head-on collision with enjoyment. Using one finger to control several things at a time is an interesting exercise as a player, but even more valuable to think about as a designer.



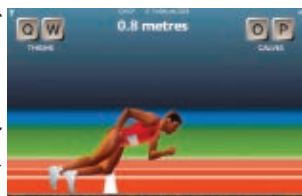
### TRIPLE THREAT

[www.ludumdare.com/compo/ludum-dare-24/?action=preview&uid=15833](http://www.ludumdare.com/compo/ludum-dare-24/?action=preview&uid=15833)

» Triple Threat is a two-player competitive tank-battle game in which you control the direction of your vehicle, but not the rate of fire. You have to learn when your tank will fire and try to damage your opponent while also capturing turret nodes to do additional damage. The game is more complex

than it should be, but that's part of the charm—removing a critical element (like manual fire) changes the dynamics completely.

QWOP, CLOP, GIRP (FROM TOP TO BOTTOM).



### QWOP/CLOP/GIRP

[www.foddy.net](http://www.foddy.net)

» Most of you have likely seen Bennett Foddy's input experiments, which have players perform inane tapping motions on their keyboards to complete a conceptually straightforward task, like running a race, moving a horse to a goal, or scaling a rock. In these games, the controls are fiddly enough that you can barely get anywhere—and that's the fun.

You know what you have to do, but making it happen is always just out of your reach. In QWOP, for example, you move your avatar's thighs and calves separately. The input method makes it hard to succeed, but due to the amusing physical animation, failure is entertaining and almost a reward in itself.

FROG FRACTIONS.



### FROG FRACTIONS

<http://twinbeardstudios.com/frog-fractions>

» Frog Fractions has only just come out as of this writing, but is already making waves for its irreverent mash-up of genres. This game begins as an homage to the edutainment games of the 1990s, but winds up referencing scrolling shooters, text adventures, typing tutors, PHOENIX WRIGHT: ACE ATTORNEY, and more, as every new screen seamlessly shifts genres. In a way it's like a long-form mini-game compilation—the theme and drive is always the same, but the input method and general design is constantly shifting underneath you.

GENM.



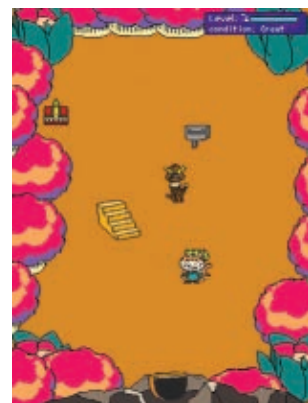
### QUIKDING

<http://quimdung.com>

» Last but not least is my favorite developer of game oddities: Quikding (formerly Quimdung). This mysterious collective of weirdos uses amateurish MS Paint sprites and backgrounds amid bizarre themes to a gloriously surreal effect. BONKEY TREK has you travel along a perilous road by smacking your bonkey (not

a typo) on the back of the head and feeding it so it doesn't die of starvation, all while answering curious questions along the way (again in order to not die). In CAVE RESCUE, you have to save 100 residents of a town who have been stuck in caves of various sorts, and take them to the government health center. Alternate input methods abound, bizarre stories leave you constantly questioning what you're doing, and glitches somehow don't feel out of place.

CAVE RESCUE.



### OUTSIDE THE BOX

» While the teachings of these oddities may not easily filter up into the world of monetized game development, they are certainly brain food for the designer, the businessperson, the musician, and the artist. There is so much more that games can do, and it seems that right now we're only investigating what they've already done. These games, and countless others like them, push just a little further than what we're used to, and there is definitely something to take away from them, regardless of what discipline we work in or how big our teams are. So why not try out some different ideas of your own in a prototype or a game jam? What you make may well inform your next "real" project. 🎮

**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.

## 2013 INDEPENDENT GAMES FESTIVAL REVEALS RECORD MAIN COMPETITION ENTRANTS

The organizers of the 15th annual Independent Games Festival—the longest-running and largest showcase for independent developers—are proud to announce that the event has once again seen record entry numbers for its latest Main Competition.

In total, the GDC 2013 co-located festival attracted 589 Main Competition entries from both already-renowned indie developers and first-time entrants, just topping the show's record-breaking 567 game entries in 2012.

Some of the hundreds of intriguing-looking titles entered in the IGF Main Competition this year include EightyEightGames's RPG matching game 10000000, Christine Love's visual novel ANALOGUE: A HATE STORY, and Blue Manchu's CCG/RPG hybrid CARD HUNTER.



FEZ developer Polytron Corporation won the 2012 Seamus McNally grand prize.

The entrants also feature titles such as Hitbox Team's action platformer DUSTFORCE, Santa Ragione and Bloody Monkey's unusual first-person puzzler MIRRORMOON, and much more.

With the event growing ever larger, IGF 2013 has expanded each of its Main Competition award categories to six finalists (except Nuovo, which has eight finalists). The Main Competition finalists will be announced in January 2013, and all will be available in playable form at a larger, expanded IGF pavilion on the GDC show floor.

In addition, all IGF 2013 Main Competition entrants are once again eligible for Microsoft Studios's second annual sponsored prize—a guaranteed first-party publishing deal (including funding if desired) to release the selected title on Live-enabled platforms, including the Xbox Live Arcade service, Windows Phone, and Windows. (Last year's winner was Capy's frantic retro platform shooter SUPER TIME FORCE.)

The festival's organizers have also provided an official JSON feed, which is updated every 30

minutes from live back-end data—teams can update info on their games and have the official entry page change, and third parties are welcome to use this feed to make their own custom IGF entry lists and pages.

Once again, winners will be honored on stage during the IGF Awards ceremony during the 2013 Game Developers Conference in San Francisco in March, showcased in the IGF Pavilion on the GDC Expo Floor from March 27-29. (GDC and the IGF are owned and operated by UBM TechWeb, as is *Game Developer*.)

## GDC 2013 OPENS REGISTRATION

Next March's Game Developers Conference 2013 is now starting to take shape, and those interested in attending can secure their passes, as online registration is now open for the major industry event.

You can register for the show by visiting the info page on the official GDC 2013 website, and discounted Early Bird pricing will remain in effect until February 13.

The GDC Summits, which take place the first two days of the five-day conference, are one- and two-day events that cover

relevant topics in emerging sectors of the game industry, with a focus on broadening the scope of games to incorporate new audiences, new platforms, and new gameplay ideas.

This year, the show will add three new summits to its robust lineup: The Game Narrative Summit, the QA Summit, and the Free-to-Play Design & Business Summit. (Submissions for the GDC 2013 Main Conference have already closed.)

The Game Narrative Summit comes to GDC in San Francisco after being held for seven years at GDC

Online in Austin, Texas. The two-day program covers interactive narrative in all its forms, from triple-A blockbusters to indie games to transmedia projects. The summit is looking to feature an all-star lineup of speakers from every corner of the discipline, with session content ranging from the advanced and theoretical for writers, designers, and others seeking to hone their skills.

The introduction of the QA Summit, meanwhile, marks the first time that the GDC will have content dedicated to

quality assurance, which is a critical component in game development. Since there is no standard methodology to QA, this summit will discuss new and/or current tools, processes, and organization methods being used in QA today.

Finally, the Free-to-Play Design & Business Summit (the successor to the Social & Online Games Summit) will be taking submissions covering a range of topics relating to free-to-play (F2P) titles—from postmortems on new and successful

F2P games, to the latest and greatest monetization techniques, to lessons learned in designing F2P titles from the ground up, to experiences on social features, among other pressing trends that F2P developers face.

GDC 2013 itself will take place March 25-29, 2013 at the Moscone Convention Center in San Francisco, California. More information is available via its official website. (GDC and the GDC summits are owned and operated by UBM TechWeb, as is *Game Developer*.)





## who went where

/// Laura Fryer, former director of Microsoft's Xbox Advanced Technology Group and a founding member of the Xbox project, is now general manager of the new Epic Games Seattle studio. With the new studio dedicated to engineering and supporting Epic's next-gen Unreal Engine 4, her prior experience in Xbox Developer Support makes her an obvious choice.

/// Daniel Erickson, the lead writer for BioWare's high-profile MMO STAR WARS: THE OLD REPUBLIC, recently confirmed that he's left the developer to assume the role of creative director at Bluepoint Games.

/// Laurence "Lo" Toney marks another high-profile departure from Zynga. Toney was the general manager of the company's highly successful ZYNGA POKER, and hasn't yet announced any further plans.

## new studios

/// People Can Fly vets Adrian Chmielarz, Andrzej Poznanski, and Michal Kosieradzki have formed a new studio in Poland called The Astronauts. The studio's working on an unannounced Unreal Engine game with "imaginative visuals and rich story-telling."

/// Jagex, the U.K.-based developer of RUNESCAPE, has founded a new California-based studio that aims to bring triple-A-style racing games to social media platforms. The former MIDNIGHT CLUB developers, with Arash Amini producing, recently finished the Facebook game CARNAGE RACING.

/// Rich Vogel, former executive producer of BioWare Austin's STAR WARS: THE OLD REPUBLIC, is heading up a new Bethesda studio in Austin, called Battlecry. The studio's initial project is still under wraps.

/// Stuart Black, best known for EA's late-PS2-era first-person shooter BLACK, has founded a new indie studio in London called Self. On his plate are an iOS title and a PC game built with the Unreal Engine.

/// 7 Wish Studios of Brighton, U.K. is officially open for business. Founded in July by Caspar Field, Tom Bennett, and Paul Brooke, the outfit is working on an unannounced original IP for Sony Computer Entertainment Europe.

## Doing Just Phyne

ARTURO NEREU ON GOING INDIE IN MEXICO CITY

WHEN MEXICO CITY'S SLANG STUDIO CLOSED, PROGRAMMER ARTURO NEREU FOUND HIMSELF WITHOUT A JOB. RATHER THAN SEEK OUT ANOTHER TRADITIONAL STUDIO POSITION, NEREU, WHO ALSO TEACHES GAME DESIGN AT TEC DE MONTERREY, DECIDED TO START AN INDIE STUDIO CALLED PHYNE GAMES. WE CHATTED WITH HIM ABOUT WORKING OUT OF A STARBUCKS AND MAKING THE LEAP TO INDIE DEV.

**Alexandra Hall:** *When Slang closed, was going indie the obvious path, or did the opportunity just sort of come together?*

**Arturo Nereu:** I felt very good, actually. When I finally made the decision of working as an indie developer I knew that I could have a lot more creative freedom, and for me that is priceless. Even before working at Slang Studio, I wanted to make my own studio to explore my ideas. I considered working for other studios, but in the end I wanted to take the risk with the Phyne Games team.

I love the freedom—the creative freedom I have on my games, but also the freedom I have to share what I'm doing with others. When working in a studio you have to sign an NDA, but as an independent developer I can be as open as I want.

**AH:** *I heard you were working out of a nearby Starbucks. How's that going?*

**AN:** To be honest, I don't like working at Starbucks—the place is just so uncomfortable to work in. But given that we don't have a formal physical place to work, we have to have some of our meetings there. What I do love about Starbucks is that you can have free focus groups; at just the next table there is a girl who you can ask about your game.

**AH:** *Does becoming an indie dev change how you relate with your game design students?*

**AN:** Yes, now I am more open with them about what I'm



developing. I can actually show them my work in progress so they can know that what I'm telling them is stuff I actually practice.

**AH:** *Given your teaching job, is it tough to find time for indie development?*

**AN:** Preparing the material for the class actually consumes a lot of development time, but I find that time useful because I'm always learning. Also, being in contact with so many creative minds is an amazing experience and I believe it helps me as a developer.

**AH:** *Given the major changes in the games business over the last few years, has your career played out differently than you expected when you started out?*

**AN:** Absolutely. Three years ago the only path I could think about was working in a professional game studio as a game programmer. But given the available opportunities, here I am, starting my own studio and working as programmer and game designer.


**AH:** *MICTLAN looks like a more personal project than you would've attempted at a larger studio.*

**AN:** Yes, especially given that we have a very unique art style and theme. Many more-corporate studios would find that very risky because it might scare gamers from other countries. We have demonstrated that such games can appeal to many players around the globe, but I still think that bigger studios are not willing to take those kinds of risks.

**AH:** *Why did you choose Windows Phone ??*

**AN:** We saw an opportunity on the Windows Phone 7 marketplace. The marketplace was (and still is) not as saturated as the App Store for example, so even when that's also a drawback it was a great platform to showcase our game and reach a lot of gamers. As a dev platform, WP7 is amazing; we used XNA and C#. We developed some Xbox 360 prototypes before MICTLAN and because both the Xbox and WP7 can deploy XNA, it was a natural step for us.

**AH:** *What's next for Phyne?*

**AN:** We are working on an iOS / Android version of MICTLAN right now, and are also prototyping a few ideas for our next project (that could be MICTLAN 2). We want to keep making games that tell our childhood stories. For now, we will be making mobile, tablet, and web games, but we want to explore other things soon. 

[WWW.NEVERMINDGAME.COM](http://WWW.NEVERMINDGAME.COM)

# NEVERMIND

WE'RE USED TO MAKING GAMES THAT HONE A PLAYER'S REFLEXES. WHAT ABOUT USING GAMES TO CONTROL BASIC BIOLOGICAL RESPONSES? THAT'S WHAT THE STUDENT DEVELOPERS BEHIND NEVERMIND SET OUT TO EXPLORE, BY DESIGNING A HORROR GAME THAT RESPONDS TO THE PLAYER'S HEART RATE. WE SPOKE WITH CREATIVE DIRECTOR ERIN REYNOLDS ABOUT WHAT MAKES NEVERMIND TICK.

**Alexandra Hall: What gave you the idea for a biofeedback game?**

**Erin Reynolds:** Biofeedback was a concept that I had explored alongside the team for *TRAINER*, a game I worked on earlier in my graduate school career. Ultimately, we didn't end up using biofeedback technology for that project, but I remained fascinated with the potential that biofeedback tech held for video games.

Fast-forward a few years. I knew that I wanted to create a game that would implicitly benefit the player in the real

**AH: How capable is the sensor hardware?**

**ER:** The tech works to the extent that the game does respond to the player's physiologic reactions as we designed it to. That said, would I want any important medical decisions to be made based on its readings? Probably not. However, for the purposes of the game, it definitely gets the job done—and, for the most part, is responsive to most users.

Heart rate variability ultimately tells us what the player's psychological arousal

the game can respond to the entire range of a player's possible emotional responses.

**AH: Is it tricky to temper basic fear responses?**

**ER:** It's actually deceptively tricky to master one's internal reaction to situations. I think many of us learn how to mold our external responses when faced with a variety of circumstances to mask internal turmoil. However, our ability to control our natural reflexes generally ends there and, as a result, we never actually


temper those biological and often automatic responses to fright and other emotions. That's ultimately the core of what *NEVERMIND* tries to achieve. We like to think of it as the P90X of stress management.

**AH: Are some players just naturally better at NEVERMIND?**

**ER:** Some players engage with *NEVERMIND* and have to stop halfway through because it is simply too intense for them. Another small section of players can play it start to finish and leave having felt nothing.

As far as players' experience with the biofeedback aspect of *NEVERMIND* goes, we found that it was actually fairly consistent. Players would notice some unexpected manifestations in the game and, after spending some more time with it, figure out that it was actually a response to their fear and stress levels. From there, they would try to control it by breathing deeply, taking a pause to get their bearings, closing their eyes, etc.

Some players reported that the game reacted as if they were stressed when they felt they were calm. In many cases, we feel that the player may have actually been physiologically responding but, again, since many of us are so desensitized to those subtle internal reactions, was simply not equipped to be aware of them.

Naturally, much more testing needs to be done to be able to say how effective *NEVERMIND* is on both a short- and long-term level—not to mention that there is always room for improvement and expansion. However, based on our observations and player responses, we feel that the current experience, if nothing else, at least gets players thinking about what's going on in the inside. In many ways, that in and of itself is a big win for all of us. 

**Developer:** Team Nevermind, The University of Southern California

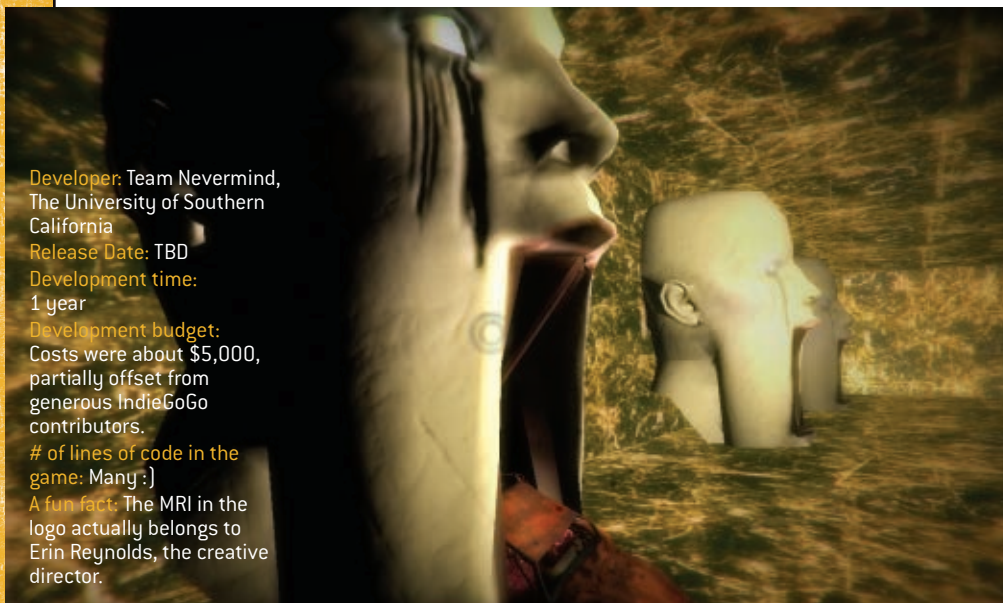
**Release Date:** TBD

**Development time:** 1 year

**Development budget:** Costs were about \$5,000, partially offset from generous IndieGoGo contributors.

**# of lines of code in the game:** Many :)

**A fun fact:** The MRI in the logo actually belongs to Erin Reynolds, the creative director.



world. Additionally, based on my personal interests, I also really wanted to use the opportunity to design a unique horror game. The combination of these goals created the perfect opportunity to revisit biofeedback. I [feel] that biofeedback is the natural next step in both the evolution of video game technology and our efforts as an industry to create a more intimate connection between player and media.

level is—that is, the intensity of their feelings. Since the player is in a dark, disturbing, and horrific environment while playing *NEVERMIND*, we interpret rises in their arousal levels as being signs of fear, stress, or anxiety. While there is certainly a diversity of game experiences that can come from reacting to changes in the player's psychological arousal levels alone, I think many of us are looking forward to a day when

learn how to address what might be going on beneath the surface.

In other words, it's the difference between looking like you're calm under pressure, and learning how to actually reach a calm state under pressure. It's like a muscle that hasn't been exercised and strengthened. As such, like strength-training a muscle, by practicing simply becoming aware of what's going on inside, it becomes easier to



# 3D SQUARE

Where ideas take shape.

3D Square is a competence center of Howest in the gaming and interactive 3D sector. 3D Square has two major goals: at the one hand supporting enterprises and knowledge institutions active in the gaming and interactive 3D sector; at the other hand guiding enterprises from other sectors in realizing their 3D projects.

A complete description of the activities of 3D Square can be found at [www.3DSquare.be](http://www.3DSquare.be)



Howest University College | Belgium | [www.howest.be](http://www.howest.be)

# INTERNATIONAL DIGITAL ARTS AND ENTERTAINMENT

MAJOR GAME DEVELOPMENT  
MAJOR 3D ARTS



## Become a technical 3D artist

UNIQUE IN EUROPE

Education: Bachelor's degree (3 years) Language: English

Location: Belgium-the centre of Europe

Curriculum: Industry-approved & award-winning

**Now accepting applications: limited entry**

Get your international career started and apply today.

For more information: [www.digitalartsandentertainment.com](http://www.digitalartsandentertainment.com)



## ADVERTISER INDEX

COMPANY NAME	PAGE	COMPANY NAME	PAGE
CCP GAMES	25	RAD GAME TOOLS	C4
EPIC GAMES	C2	VANCOUVER FILM SCHOOL	11
HOWEST UNIVERSITY	55		

*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.



# STACK TRACE AND THE DEATH OF A.A.A. DEVELOPMENT (PART 1)

## IS THE GAME-DEVELOPMENT DETECTIVE IN OVER HIS HEAD?



It was pouring rain as I stared out the window. It always was during this time of year—the Fall Release Deluge, they called it. I was about to take another sip of my LevelUppp GamerJuice on the rocks when the phone rang.

"Stack? It's First," said the gravelly voice on the line. Police Commissioner Depth First sounded tired. Of course he did. People come to me when they're out of options.

"First. Look, I'm not in the mood for pizza."

"Stack, I already apologized for that. I said I'd never order Papa John's for crunch again. Let it go, okay? This is serious. I mean real serious."

"Shoot."

"He's dead, Stack. Triple-A Development is dead."

There was a pause as long as a dropped ping.

"Development? You mean the Aaron Alexander Akbar Development? Are you sure?"

"I am. What's more, it's murder. Someone killed him."

Commissioner First was right: This was serious. Everyone in town knew who A.A.A. "Triple-A" Development was. He'd basically built the place—bought up all the best

property with the help of his old buddies from the military days. Sure, he had ups and downs over the years, but he was always a force to be reckoned with. He was a fixture. An icon.

And, to be honest, not the kind of man whose business I wanted to get mixed up in.

"So what do you want me to do about it?"

"Look, Stack. This is hot—too hot for us. There are so many players involved: the platform holders, core and casual audiences, retail chains, analysts, bloggers—you name it. Stack, this is like the Arcane Explosion of cases."

"The what?"

"AoE, Stack. There's a huge area of effect on this one and I can't get near it. That's why I called you—to see if you could do a little digging, talk to his former associates, find out what he was up to in the last few months. Are you in? I promise there's a reward in it for you that's not pizza."

\*\*\*

I knew who I had to talk to first. I figured I'd find her in the subway station in the morning or the afternoon, and I was right. I spotted Mobi LePhone, absorbed in the latest indie beats, as she was getting off the train the next day. She was

hard to miss, silhouetted against a shifting sequence of bright, flat colors.

"Hey, Mobi! Hey!" I said, waving at her. She finally noticed me and yanked out one of her earbuds.

"Oh, hi, Stack."

She continued dancing and had the earbud halfway back before I managed to say, "Did you hear? Triple-A is dead."

"Is he?" She stopped.

"Well, isn't that something? I'd heard things had been rough for him lately, so I can't say I'm surprised."

"Listen, Mobi... Do you know if Development had been hanging out with anyone new lately? Trying any risky new business model?"

"Well, sure he was—but he did that all the time, you see. You don't stay the kingpin of this town without dancing with every new devil that comes your way."

"Business models like free-to-play, for example? Microtransactions? Your kind of thing?"

"Stack, don't be ridiculous. You can't possibly think that I'm the one who did it, do you?"

"Just investigating every lead. I am a detective, you know."

"Not a very good one. Sure, ol' Dev was looking at free-to-play—who isn't these days? But it doesn't

take a venture-backed online metrics platform to figure out he was spending too much to chase down too few users." She shrugged. "I thought everyone knew that by now."

"Then I guess I'm the last to know."

"Besides, I couldn't have killed him, even if I wanted to. I don't have the memory or processing power to take on someone like Triple-A. You know as well as anyone that true gamers want rich, high-fidelity experiences that only fixed consoles and large television screens can provide. In fact, I believe there will always be an audience and a future for A.A.A. Development's games."

"Now you're just repeating his propaganda, Mobi. You can't expect me to believe you posed absolutely no danger to his empire of—"

"If you want, you can stop bothering me and follow up on a real lead," Mobi said. "There was a new guy in town that Development was real keen to work with."

My ears perked up. "Yeah? Who was that?"

"Honestly, Stack, you really are the last to know. Everyone's been talking about him—he shows up outta nowhere a while back and suddenly he's the talk of the town. Development was

falling all over himself to get in cahoots with this guy."

I was starting to get impatient. The sun had just begun to set, and there was a bottle of GamerJuice at the office with my name on it.

"Okay. And who was that, Mobi?"

"Well, maybe I don't remember."

"Come on. Don't do this to me."

"Sorry. If you want the hint, it's one coin. You can buy coins for 99 cents each. For two coins I'll give you an upgraded magnifying glass, which allows you to find clues more easily, and for five coins—"

"Yeah, just the hint, please. You can put it on my tab."

"Pleasure doing business with you. Ah, yes, it's coming back to me now... His name was... It was... Mr. Waggle." 📞

### TO BE CONTINUED...

**MATTHEW WASTELAND** writes about games and game development on his blog, *Magical Wasteland* ([www.magicalwasteland.com](http://www.magicalwasteland.com)). email him at [mwasteland@gdmag.com](mailto:mwasteland@gdmag.com).

**MAGNUS UNDERLAND** writes about games and other topics at [www.above49.ca](http://www.above49.ca). email him at [magnus.underland@gmail.com](mailto:magnus.underland@gmail.com).





LEARN.NETWORK.INSPIRE.

GAME DEVELOPERS CONFERENCE

SAN FRANCISCO, CA  
MARCH 25-29, 2013.  
EXPO DATES: MARCH 27-29

2013

[WWW.GDCONF.COM](http://WWW.GDCONF.COM)







THERE'S NOTHING LIKE BEING A

**RAD**

GAME DEVELOPER



.....



It feels like you have



when you use Bink, our amazing, super **FAST** video and audio codec.



You'll **LOVE** using Granny, our

run-time dynamic **3D ANIMATION SYSTEM.**

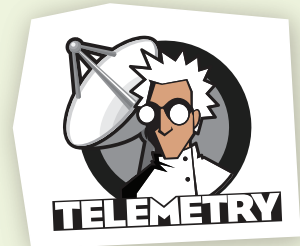


There's nothing like Miles, our multichannel sound system.



And our newest tool,

Telemetry, is a library for profiling, tuning and visualizing application **PERFORMANCE**



.....

So it's obvious, there's nothing like using **RAD** tools.

Well, nothing except having a killer

**MUSTACHE**

