

THE MONEY ISSUE 2013 **INSIDE:**



APRIL 2013
VOLUME 20 NUMBER 04

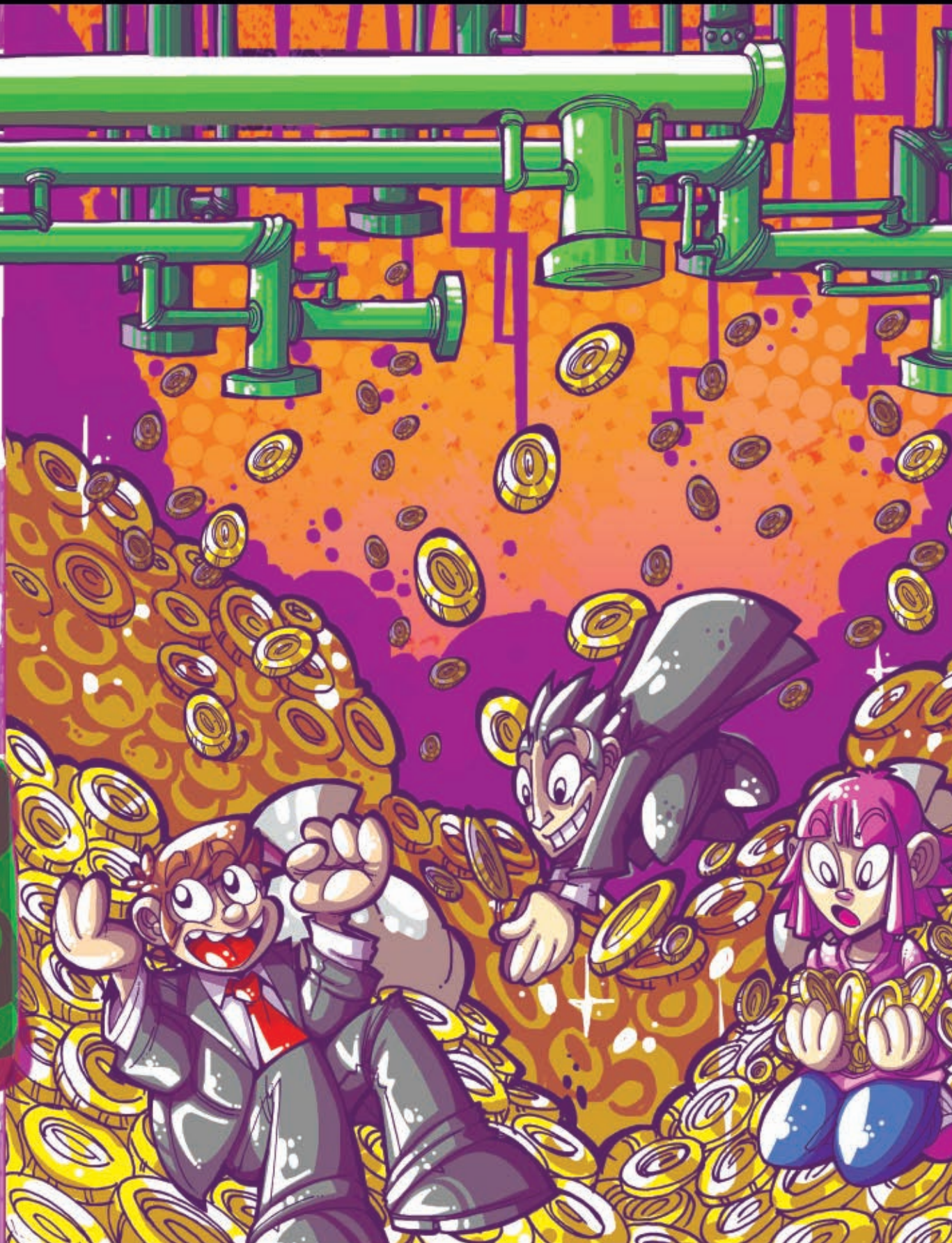
gd

GAME DEVELOPER MAGAZINE



XCOM: ENEMY UNKNOWN POSTMORTEM

12TH ANNUAL SALARY SURVEY



+ CROWDFUNDING: ONE YEAR LATER // // // // // // // // // //

MAKE GREAT GAMES

Developers, build HTML5 games for the regulated gaming industry using our:

Certified RNG & game engines

HTML5 Game Development Kit (GDK)

Session handling (resume play)

Cloud content delivery network

Multi-currency handling

Multilingual library & tokens

Territorial rights management

Single licensing and distribution agreement

Drive your success by promoting your games to players and operators on Odoobo Play.

Apply today.

THERE'S NOTHING
VIRTUAL ABOUT
THE OPPORTUNITY
IN REAL-MONEY
GAMBLING.

odobo
RAISE YOUR GAME

www.odobo.com/developers

EARN \$100+ PER PLAYER

Real-money online gambling is already a regulated \$35bn industry outside the US. The US market is quickly emerging. With OdoBo, developers can produce games for the leading licensed online casino operators worldwide. OdoBo looks after distribution, monetization and compliance.

As an OdoBo developer, you can earn substantial royalties from the play of your games offered to existing vast player bases. Earn an additional 30% of the lifetime value across all gaming activity when you drive new players via OdoBo Play.

The combination of immediate royalty income and affiliate commission earns OdoBo developers the highest ARPUs in the games industry.

Want to know more? Get 'The Real Money Gambling Opportunity', an exclusive report by H2 Gambling Capital. Visit www.odobo.com for more information or tumblr.odobo.com to download your free copy. Also available by emailing press@odobo.com





LET YOUR CAREER
**REACH THE
PEAK!**

JOIN PEAK GAMES !

SHAPE AND DRIVE THE FUTURE OF TECHNOLOGY
WHILE CONTINUOUSLY DEVELOPING YOURSELF

Talk to us at

hr@peakgames.net

peak
GAMES

www.peakgames.net



gd

GAME DEVELOPER MAGAZINE

003

CONTENTS_April 2013
VOLUME 20 NUMBER 04

Postmortem

036 XCOM: ENEMY UNKNOWN
Resurrecting a complicated turn-based strategy game that practically defined "old-school hard" seems like a recipe for commercial failure in today's market. In this month's postmortem, Firaxis Games's Garth DeAngelis explains how they defied the odds with XCOM: ENEMY UNKNOWN. *By Garth DeAngelis*

Features

015 SALARY SURVEY
We're back with another *Game Developer* Salary Survey! Find out how much the industry made in 2012. *By Patrick Miller*

22 CROWDFUNDING, ONE YEAR LATER
What does crowdfunding look like in 2013? *Game Developer* talked to Greg Rice (Double Fine), Chris Roberts (Roberts Space Industries), Brenda Romero (Loot Drop), and Jim Rossignol and James Carey (Big Robot) about the current crowdfunding climate. *By David Daw*

030 THE LANGUAGE OF MONETIZATION DESIGN
If you're going to try to design your game to make money, make sure you know what you're talking about with this guide to monetization design terms. *By Ramin Shokrizade*

033 MINI-MORTEM ROUNDUP
What's it like developing for <insert platform here>? The devs behind FASTER THAN LIGHT, DYAD, DRAGON FANTASY, and WAR COMMANDER share their dev experiences in this collection of short postmortems. *By Staff*

Departments

004	Game Plan	[Editorial]
008	Heads Up Display	[News]
010	Educated Play	[Education]
011	Good Job	[Career]
012	GDC News	[News]
049	Toolbox	[Review]
053	Inner Product	[Programming]
057	Pixel Pusher	[Art]
060	Design of the Times	[Design]
063	Aural Fixation	[Sound]
064	The Business	[Business]
089	Insert Credit	[Editorial]
096	Arrested Development	[Humor]

game developer magazine

game developer magazine

003

MONEY ISSUES

MONETIZATION IS OUR INDUSTRY'S NEXT BIG PROBLEM—AND OPPORTUNITY

"Free-to-play is killing video games!" If you've heard it once, you've heard it a thousand times. From a business perspective, free-to-play is a useful tool because it can offer smaller studios a shot at an extraordinarily wide audience and higher overall revenues than the pay-once model—which, in turn, means more stability and job security. But nobody likes playing—or making—a game that feels like it's powered by your wallet, either. Rather than wish f2p would go away, we'll just have to get really good at using it.

MORE (MONEY, PROBLEMS)

Judging from the comments from this year's Salary Survey, monetization is on everyone's minds these days. But while devs and consumers alike aren't shy about heaping disdain upon free-to-play games, it's worth pointing out that over the last three years of salary surveys, we've seen a gradual decline in layoff rates and an increase in average salaries. Certainly, not all of those gains are necessarily due to the rise of the f2p model, but if you think about some of the hidden costs incurred with the traditional pay-once development cycle, you might be a little bit less skeptical about f2p.

Traditional game development, as we think of it, is somewhere between the entertainment industry/Hollywood model, where you assemble a one-time team of people to produce one project, and the software development model, where you have a team of developers focused on building and improving a product for as long as that product is sold. If you're a developer making, say, Microsoft Word, you can be pretty sure that once you've shipped a version of Word, you'll still have plenty of work left to do with fixing remaining bugs, releasing new patches, and working on the next version of Word. If you're a game developer, though, at some point your game will be "done," and your studio might not have another project for you to work on. Game devs


end up with all the liability of a film worker, but without any of the unions or support structures that make that model sustainable.

On the other hand, many f2p games launch as early as they can put together a minimum viable product in order to start getting revenue coming in, and then gradually add new features and content after launch. As long as there is something to add to the game, there's a reason for the dev studio to keep people employed and working on the game. Logically, that means we should see fewer layoffs in f2p game dev (when the games are performing well, anyway). As my film editor buddy Brian put it, "Pay-once dev is like working on a blockbuster film, free-to-play is like working on a TV show."

FUN-TO-PAY? Free-to-play proponents like to mention that arcade games were the first example of monetization design. What many people seem to miss is that some of those games actually hit the Holy Grail of monetization design; they made paying fun. Play CONTRA on free-play mode and it gets dull fast because there's no cost to failing. Play it with a fixed amount of lives and continues and things get more interesting, but you end up playing through the same segments over and over. Play at 25 cents per continue, and you'll find yourself marshaling every last pixel in that health meter, asking yourself whether it's worth

another 25 cents to see the next level, and so on. The experience is actually enhanced by the presence of actual, real-world stakes [the quarters in your pocket].

Another unorthodox example of effective monetization design is the time-honored "money match," where two players bet on the outcome of a game. The fighting game community has taken these to rather ridiculous extremes (see the MARVEL VS. CAPCOM 2 \$50,000 money match between Toan and Fanatiq), but as an enthusiast myself, I love upping the stakes by putting a dollar or two on the line just to give each in-game moment a little bit more real-world weight. I lose more than I win, but the extra thrills make it worth it. And as f2p models continue to develop, I suspect we'll see more going on in f2p than just sticking a price tag on in-game content.

MAKE THE WORLD GO ROUND Nobody wants to play a game that makes you feel like a cash cow. But pay-once games are harder to sell than they were 10 years ago, and those business models also gave us wonderful workplace practices like "crunch time" and "laying everyone off after ship"—both of which make it harder to attract, cultivate, and retain talented developers. If we want to see the game industry become a place where developers can reasonably see themselves supporting their families, buying homes, and sticking around until retirement, we're going to have to solve The Money Issues in a way that makes everyone—devs, suits, and consumers—happy. 

—Patrick Miller
[@pattheflip](#)



GAME DEVELOPER
MAGAZINE
WWW.GDMAG.COM

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

SUBSCRIPTION SERVICES

FOR INFORMATION, ORDER QUESTIONS, AND ADDRESS CHANGES

t: 800.250.2429 f: 847.763.9606
gamedeveloper@halldata.com
www.gdmag.com/contactus

EDITORIAL

PUBLISHER

Simon Carless scarless@gdmag.com

EDITOR

Patrick Miller pmiller@gdmag.com

EDITOR EMERITUS

Brandon Sheffield bsheffield@gdmag.com

MANAGER, PRODUCTION

Dan Mallory dmallory@gdmag.com

ART DIRECTOR

Joseph Mitch jmitch@gdmag.com

CONTRIBUTING WRITERS

Alexandra Hall, David Daw, Ramin Shokrizade, Garth DeAngelis, Dave Mark, Chris Parnin, Steve Theodore, Damien Schubert, Damian Kastbauer, Kim Pallister, Matthew Wasteland, Magnus Underland

ADVISORY BOARD

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

ADVERTISING SALES

VICE PRESIDENT, SALES

Aaron Murawski aaron.murawski@ubm.com
t: 415.947.6227

MEDIA ACCOUNT MANAGER

Jennifer Sulik jennifer.sulik@ubm.com
t: 415.947.6227

GLOBAL ACCOUNT MANAGER, RECRUITMENT

Gina Gross gina.gross@ubm.com
t: 415.947.6241

GLOBAL ACCOUNT MANAGER, EDUCATION

Rafael Vallin rafael.vallin@ubm.com
t: 415.947.6223

ADVERTISING PRODUCTION

PRODUCTION MANAGER

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

REPRINTS

WRIGHT'S MEDIA

Jason Pampell jpampell@wrightsmedia.com
t: 877-652-5295

AUDIENCE DEVELOPMENT

AUDIENCE DEVELOPMENT MANAGER

Nancy Grant nancy.grant@ubm.com

LIST RENTAL

Peter Candito
Specialist Marketing Services
t: 631-787-3008 x 3020
petercan@SMS-Inc.com
ubm.sms-inc.com



UBM
Tech

WWW.UBM.COM

ANDROID USERS: The *Game Developer* Android app is now live on Google Play! Check it out here: bit.ly/11btQ7V (If you installed a previous build and are having problems, try deleting and reinstalling it.)

Track Your Game Development with

DevSuite

Integrated Quality Management,
Defect Tracking & Agile Development.



- Multisite Enabled
- Web, Windows and iPad Based Clients
- Small Teams Pay Maintenance Only!

ADVERTISEMENT

A **YOUNG** BUT **GROWING** VIDEO GAME INDUSTRY THAT MIXES CREATIVITY WITH TECHNICAL QUALITY



The Colombian video game and mobile application industry has been growing exponentially in the past few years, due to several factors: a high level of local talent, top-notch technology, and competitive prices amongst many others. In fact, Colombian developers have already made successful strides into markets such as the United States and Canada, proving to meet international quality standards.

The increasing presence of Colombian video game companies in some of the most important global events in the field, demonstrates the level of their services regarding digital content. In 2012 alone, 150 Colombian companies participated in events such as the Mobile World Congress in Barcelona, the Game Developers Conference in San Francisco and Germany, and the MIPCOM festival in France.

2013 will be no exception. Seven companies will participate in the next Game Connection event in San Francisco: Brainz, Efecto Estudio, NDiTeravision Games, C2 Studios, Colombia Games, Press Start, and Gara Entertainment with an additional 12 companies participating at the Game Developers Conference: Higuera Studios, NASKA Digital, Himedia, Wekantú Studios, Blazing Soft, Blokwise, The Ethereal Game Factory, Below the Game, Blurteam, 360 Digital, Flamin Lab, and Unitcom Digital.

"In these events, Colombian developers have had the opportunity to see first-hand industry trends. This has helped them focus their products and services towards the demands of the international market," explained Maria Claudia Lacouture, President of Proexport, the Colombian government entity in charge of promoting investment, tourism, and endorsing Colombia as a supplier of goods and services.

This has provided better exposure of the Colombian talent, which translates into business success stories.

For example, the Canadian company NDi Media (leader in North America in digital content development), decided to buy the Colombian company Teravision Games, which was a decision driven by the mix of creativity and technical quality in the Colombian video game services. "This is a winning combination between our experience and Colombian skills in creating fun and addictive games," said Neil Smolar, President of NDiMedia and current CEO of NDiTeravision.

Proximity, cultural similarity, technical quality, creativity, and cost-effectiveness are some of the reasons that have motivated deal closures between Colombian companies and international buyers. So don't miss out on having a closer look at this growing and thriving industry. Visit Colombia at the Game Connection event in San Francisco on the 4th floor (March 25-27, 2013) and at the Game Developers Conference (March 25-29, 2013).

If you want more information about the Colombian video game industry, please contact us at sanfrancisco@proexport.com.co or visit www.proexport.com.co/en/events/gdc2013.



www.proexport.com.co



Government of COLOMBIA

PROEXPORT COLOMBIA
TOURISM, FOREIGN INVESTMENT AND EXPORTS PROMOTION

IN THE VIDEO GAME
INDUSTRY
THERE ARE HARD MOVES



AND THERE ARE
SMART MOVES
THAT CAN LEAD YOU TO SUCCESS

START



LIKE CHOOSING A COUNTRY THAT HAS A GREAT
OFFER IN VIDEO GAME DEVELOPMENT

THE
ANSWER
IS



Visit our Colombian companies at Game Connection
America on the 4th floor and at Game Developers
Conference (GDC) in San Francisco and discover
their high technical level and competitive prices.

LEARN ABOUT OUR COMPLETE OFFER IN
www.proexport.com.co/en/events/gdc2013
FOR MORE INFORMATION PLEASE CONTACT US AT:
[SANFRANCISCO@PROEXPORT.COM.CO](mailto:sanfrancisco@proexport.com.co)

www.proexport.com.co



Government
of COLOMBIA

PROEXPORT
COLOMBIA
TOURISM, FOREIGN INVESTMENT AND EXPORTS PROMOTION

GO FORTH, WARRIOR

TEACHING THE FORTH PROGRAMMING LANGUAGE THROUGH A VIDEO GAME



To many in the industry, a game that can teach programming concepts is the Holy Grail of technological education—see CODE HERO'S rampant Kickstarter success, for example. Well, John Earnest developed a small game called FORTH WARRIOR (<https://github.com/JohnEarnest/Mako/tree/master/games/Warrior2>), in which you control a hero's actions using only Forth commands. We talked to John about the process of coding a game to teach coding.

PATRICK MILLER: *What's your day job?*

JOHN EARNEST: I'm a PhD student at Michigan Tech, specializing in programming language design and compilers. When I'm not tinkering with compilers or writing video games I enjoy making pizza from scratch and drawing comics. I aspire to own pet chickens.

PM: *Tell us a little bit about Forth. What's it used for? Is there any carryover to other modern languages?*

JE: Forth is what's called a stack-based language, or more formally a concatenative language. Code is written in postfix, sort of like old HP calculators, with operands flowing between procedures via an implicit stack rather than explicit variable names. This leads to code that is very compact, even if conventionally trained programmers feel everything is "backward" at first. Forth has the low-level character of assembly language, but offers metaprogramming facilities that allow one to radically extend the language to suit the task at hand.

The language was invented in the mid-'70s, and was always a little too strange to hit mainstream. It still maintains a cult following among embedded systems programmers when resources are tight. Forth compilers are an order of magnitude smaller and simpler than those for languages like C, so it's a great option for bootstrapping new computer architectures. The modern PDF file format is based on another stack-based language called PostScript, which bears many similarities to Forth.

PM: *Why'd you decide to build Forth Warrior? Was there anything about the language itself that makes it easier to teach its principles via video game?*

JE: There aren't many modern resources for learning about Forth. I'd talked to a few programmers who were curious about the language but didn't know what sort of project to build. I thought that designing a game that provided a simple, portable Forth dialect and a range of well-defined tasks would be a wonderful source of motivation.

The simplicity of Forth certainly made it easier to write a self-contained compiler and programming environment for the game. (Last I checked, the executable is around 41k!) The

dynamic, interactive nature of the language helps too; you can test your programs in tiny pieces as you go along.

PM: *What did you learn from the process of building Forth Warrior? Did you have any interesting failed iterations along the way?*

JE: Designing puzzles that are meant to be solved by a computer is an interesting task. You can't depend on programs making intuitive leaps, and many tasks that are easy for humans are very difficult for computers. Even a humble Sokoban-style block-pushing puzzle becomes computationally intractable rather quickly. I ended up making levels much like I would write a test fixture—each explores a few subtleties of how the game elements interact, gradually forcing the programmer to make fewer assumptions and perform more elaborate decision-making, which in turn provides opportunities to use more language features.

I wrote the first prototype of the game in an afternoon roughly a year ago—it's still available in my GitHub repository. It was much clunkier to work with, since it depended on all sorts of external compilers and tools. Inspired by success with a Logo IDE I wrote for use with a local after-school program, I decided to revisit FORTH WARRIOR and build something a little more polished and standalone. Adding support for external code editors was loudly demanded by my earliest testers, and in retrospect I think writing sufficiently elaborate programs to beat the game would be torture without it.

PM: *What's next? Any plans for similar projects?*

JE: Well, I've received more positive feedback for FORTH WARRIOR than any of my other projects, so it definitely seems like there's an audience for more games like this. Many players have expressed a desire for a head-to-head competitive mode in the game, a little more in the spirit of classics like ROBOWAR. Maybe an enhancement for FORTH WARRIOR, or maybe a sequel? I also have a few ideas for a Logo-based game that can teach basic geometry concepts alongside programming, aiming more at beginners. I can't say for certain what my next project will be, but I definitely want to continue designing games that educate as they entertain.

—Patrick Miller





INDEPENDENT AGENT

A VETERAN INDIE DEV LOOKS BACK—AND FORWARD

Twenty-year industry veteran Keith Nemitz, creator of DANGEROUS HIGH SCHOOL GIRLS IN TROUBLE! and IGF 2013 finalist 7 GRAND STEPS, has been indie for over a decade—way before it was hip. We checked in with him in the midst of developing his latest, and possibly last, self-funded game.

ALEXANDRA HALL: *What aspects of the biz have changed the most, and least?*

KEITH NEMITZ: What changed most was the market. Games invent markets, because all humans play games. It's taken developers 40 years to diversify their craft enough to reach everyone. Sparse thanks to the corporations. Nearly every new game market was created by indies. Indie games sold homebrew computers. Indie games fueled the adoption of PCs. Some of those indies became corporations (Roberta Williams), some work in corporations today (Bill Budge), some are still indies (Jeff Minter).

What hasn't changed much is the laser-focus attention on hit games. Most gamers think hit-driven is a fact of the industry, but in reality the system can take a few simple steps to soften the curve.

The most difficult challenge is getting sufficient attention to achieve profitability. We are now in an era of great games, but nobody realizes it because of all the crap. Everyone can point to a few greats: ANGRY BIRDS, PLANTS VS. ZOMBIES, XCOM, DEMON'S SOULS, PERSONA... But there are 10 other great games for every hit. I just enjoyed playing a fine point-and-click adventure, THE SILENT AGE. It's been out for a while, but I'd never heard of it. The developers even won an award! Will those creators of a fine game survive?

AH: *Do you feel optimistic?*

KN: I'm super optimistic about our industry's future! Games and audiences are diversifying exponentially. The market is expanding. And, something which has never happened before in an artistic medium, independents are determining the course of the industry as it grows/transmogrifies, not after a few corporations and brands have taken all the mindshare.

One caution, however. If indies continue to rally behind a few "heroes," they are hindering their own futures. We

need to spread the mindshare, to flatten the nasty curve endemic in hit-driven businesses. Every indie's website should link to a dozen or more of their favorite games, not to their friends' games or their heroes' games or games already well known outside indies, but to indie games each dev genuinely loves that deserve more mindshare.

I'm part of a community of developers who recently formed a philosophy about the indie game market. It is functionally infinite. We are not actually in competition with each other. That only happens in contests. I could put my older games on hundreds of game websites today, where the audiences have never heard of them. There are millions of gamers who've never heard of BRAID. The indie game market is essentially infinite.

AH: *Any advice for aspiring indies?*

KN: The chance to succeed is tiny; 3% of devs on iPhone are sustaining themselves, and it's much worse on XBLIG. On PC and PSN, devs fare better, but only the ones selected to be on Steam or PSN. The challenges are many:

- 1 Can you make a great game?
- 2 Can you drive traffic to your IP? (build a community)
- 3 Can you network in the game dev ecosphere? (indie and industry)
- 4 Can you run a business?

If you can't do all of those well, then make games for the fun of making games, instead of making them your livelihood. Personally, I'm not very good at #2 and only okay at #4. Because of it, after 10 years of making critically successful games, my company has yet to make a profit. If 7 GRAND STEPS doesn't sell twice as well as DANGEROUS HIGH SCHOOL GIRLS IN TROUBLE! (my best-selling game), then I will probably be out of business next year.



DANGEROUS HIGH SCHOOL GIRLS IN TROUBLE!

AH: *Do you miss big teams/budgets?*

KN: Honestly, I think I've reached my limit. To make higher-quality games, I'll need to work with partners instead of contractors. Partners have equal input and ownership. I dislike imagining and deciding everything myself. I remember a vivid example. One tiny suggestion, from a friend, tightened a disconnected narrative into the fantastic story of DANGEROUS HIGH SCHOOL GIRLS IN TROUBLE!: "What if the pony was in love with the mayor?"

I enjoy the freedom of working for myself, developing games, but I may not be sufficiently suited to the business of it.

AH: *So a small team would be your ideal?*

KN: For me, yes. I think there's diminishing returns as an indie team gets larger. Partnering is pretty optimal. Myself, I would like to work in a team of four to eight. It's unlikely I will be able to afford that many employees. Nor would I want to manage them! So I would work at a company to be part of a team again. This time, it would have to be an exceptional company that honored the indie spirit.

—Alexandra Hall

THE MINDFUL XP VOLUME

[HTTP://WWW.MINDFULXP.COM](http://www.mindfulxp.com)



While still in a rocky adolescence, video games are increasingly accepted as a serious art form that can stand alongside mainstays like literature and film. To that end, three students at Carnegie Mellon aimed to create a series of short-form games that would communicate “meaningful” ideas in ways unique to the form. The result, THE MINDFUL XP VOLUME, was a finalist in the 2013 IGF Student Showcase.

ALEXANDRA HALL: Define a “meaningful” game.

MINDFUL XP: It isn’t a particularly cut-and-dry thing, even from our perspective. For the purposes of our project we looked at meaning as something admittedly vague—an impactful experience that invites introspection, personal reflection, or gives insight. We’re not cultural arbiters or philosophers about the topic of meaning, but that definition was something for us to

aspire to if not achieve. Ideally if someone were to play a game of ours, if what we were trying to express was just enough to make a player go, “Yeah, there was something to that,” then that was something meaningful. For a meaningful game, we went a step further, and specified that the meaning must be arrived ultimately through their systems, rules, mechanics, interactions, or anything that makes them games, rendering them unrepresentable in other mediums in any deep way. While these concepts aren’t unique to games specifically, we feel that games in particular explore these aspects in unparalleled depth, especially when it relates [to] an audience. We simplified this down to, “If you can read the book or watch the movie version of our game and still reach the same conclusions, then we’re doing something wrong.” As to what makes a meaningful game... all we could do as creators was create games that spoke to ourselves, and hope that

the universality of what we were trying to say was enough for others to find meaning in it.

AH: Which mindful xp games best achieved your goals?

MXP: In MARCH we made a game that formed a powerful connection through metaphors created through spaces as well as the main mechanic of leading someone by the hand, to create introspective moments. In CONNECTIONS we focused on a system that conveyed the difficulty of maintaining relationships over time and distance. And in EMPTINESS, the gameplay was centered around a moral choice of taking advantage of the people around you or trying with more effort to reach an end, together (or not).

AH: What did you learn along the way?

MXP: We struggled with expressing grand messages. Making a game about the power of cooperation (as an example) is an abstract thing that is hard to

capture for one person much less a thousand. The messages we initially tried to convey were all in this vein, and we found that trying to capture every nuance of these broad messages often ended with us feeling like we had captured none of it. So in the second half of the semester we focused on messages that were informed through personal experiences, which meant we could refine the small details that really mattered. Surprisingly, the more intimate we made a message the more universally it spoke to players. Also, we found that having too many game systems diluted any overall message we were conveying. Narrative and aesthetics were important tools in making our message understandable. Each game ended up being a reaction to the previous one as we took lessons learned and applied them going forward.

AH: Did you worry about being branded as pretentious?

MXP: We did two pretty

Developer: mindful xp

Release date: Final game release May 10, 2012, volume release May 13, 2012

Development time: 15 weeks
Development budget: \$0 (just time, blood, sweat, and tears)

of lines of code in the game: 28,849, not including all prototypes + MARCH

A fun fact: Mike once took a class under Ian Bogost, Dan once took a class under Andy Nealen, and Felix once took a class under Paolo Pedercini.

Team members: Mike Lee, lead programmer Dan Lin, lead artist Felix Park, lead designer

damning things—we weren’t trying to make fun games, nor were we trying to create games that would appeal to a wider audience (however that can be defined). Since we weren’t trying to meet those expectations in the first place there really wasn’t any huge concern about being generalized as pretentious. What we were worried about though was the reaction of people who make games in this same space as well as their players (which of course includes ourselves). Our focus around researching what practices might lead to more meaningful games could be seen as an attempt to prefabricate meaning, or the very notion of being able to inherently make something meaningful, just like that, would be pretentious at worst and presumptuous at best. In the end our concerns might have been overly cautious (so far). Still, we’re very much aware of possible perception issues.

—Alexandra Hall

LEAVING UNITY

UNITY CO-FOUNDER NICHOLAS FRANCIS LEAVES TO GO MAKE GAMES



Over the last few years, Unity has been credited with radically changing the technical barrier to entry and enabling individual devs and small studios to jump in and just start making games. But what do you do once you're done changing the industry? We caught up with Unity co-founder Nicholas Francis, who recently announced he was leaving Unity to make games himself, about his role with Unity and his upcoming plans.

PATRICK MILLER: *What was your primary role with Unity?*

NICHOLAS FRANCIS: The main thing I accomplished at Unity was the Unity Editor. Keeping it clean. Working with the other devs and helping them realize ways of making Unity's features simpler and easier to use. I think one of the greatest things about Unity is that there's not this "Let's throw a feature together, expose every option and let people figure it out" mindset. Instead, we took the time to get the design

right, reiterate, fix things, clean them up, and then ship.

Quite often, when people come from other engines to Unity, they get this serene look on their face—like you just handed a glass of ice water to someone in Hell. If I had to sum up one "greatest thing" that I accomplished, it would be that look.

In the early days, we were three guys in a basement flat, so everybody did pretty much everything; I did a lot of the core graphics programming. As the company grew, I also did a ton of marketing work, website design, meeting with investors, explaining to people what we were trying to do. Classical founder-type stuff. As we grew, we got dedicated teams: GFX programming, platform teams, marketing, sales, legal, and so on. That's when I zeroed in on the core product design.

PM: *How would you describe the impact that Unity has had on the game dev industry? Where do you think it will go from here?*

NF: I remember back when we started Unity, the world was

quite a different place: We wanted to make a game, and pretty much the only option was to buy some triple-A engine for an obscene amount of money—of course they wouldn't tell you what that amount actually was—and you had to sign NDAs before you could even try it. Today, all that's changed: A game engine is just something you have, like Maya or Photoshop. This means that pretty much everyone can make games today. If you combine that with the rise of mobile, I think Unity has been one of the key factors in the re-emergence of the indie game dev. Now, it's not like I think all of this was our doing, but it certainly seems like we played a key part in that.

PM: *From all accounts, you're leaving Unity to make games yourself; do you have any specific plans you're willing to share? What kind of games do you want to make?*

NF: Let's just say that I've always been a cyberpunk fan.

PM: *Was this in the plans for some time? What prompted you*

to leave right now?

NF: It had been coming for quite a while—some sort of restlessness. Last autumn I took some time off to think on the meaning of life. That turned up absolutely nothing. But there were some games I had to make before I died, and since you never know when it's your time, I guess that means you should get started sooner rather than later. The plans solidified over Christmas, and then it was all about actually executing it—Unity's quite spread out, and there were some friends I wanted to tell in person, so it took a while.

PM: *So, now that you're leaving Unity, you're going to be working all in Unreal, right?*

NF: ...And, in related news, I have also just picked up masochism and taken delivery of my first ball gag...

Who Went Where

- **IAN LIVINGSTONE**, co-creator of FIGHTING FANTASY and long-time Eidos executive, has been appointed chairman at Playdemic, a social games developer with several Facebook hits to its credit.
- Pauline Jacquey has been appointed as managing director of **UBISOFT'S NEWCASTLE-BASED STUDIO**, Reflections. Jacquey has a long history at Ubi, having produced the console versions of GHOST RECON ADVANCED WARFIGHTER and mobile games in the PRINCE OF PERSIA and ASSASSIN'S CREED series.
- **RAHUL SOOD**, former general manager of interactive entertainment business at Microsoft, will now serve as adviser to the board at Razer, makers of hardcore-oriented PC peripherals.

New Studios

- Japanese **MOBILE GAMING GIANT GREE** has formed a new company with Yahoo! Japan, with the aim to focus on development of social games for smartphones. The joint venture, "GxYz, Inc.," will combine the user acquisition capabilities of Yahoo! Japan with Gree's development and operation skills.
- LEAGUE OF LEGENDS developer Riot Games has opened a second studio, this time in Sydney, Australia, as it looks to better support its **POPULAR FREE-TO-PLAY MOBA** in the Oceania region. Riot Games Sydney will focus on marketing, e-sports, and the community aspects of LEAGUE OF LEGENDS, and the PR teases "much more to share soon."
- In Los Angeles, **THE NEW XBOX ENTERTAINMENT STUDIOS** is headed by Nancy Tellem, Microsoft's president of entertainment and digital media. The studio's mandate has it creating "visionary original interactive content" for the company's consoles.
- **PROLETARIAT INC.**, a new game development studio formed by the original founding group behind Zynga Boston, is open for business with iOS game LETTER RUSH.

GDC STATE OF THE INDUSTRY RESEARCH REVEALS MAJOR TRENDS

INDIE DEVS INCREASING, CONSOLE DEVELOPMENT STABLE, NEW PLATFORMS HAVE DEVELOPERS' INTEREST

In order to paint a picture of the games business as it looked right before GDC 2013, the Game Developers Conference polled more than 2,500 North American game developers who attended the conference in 2012—or planned to attend GDC 2013—about their development practices, revealing several notable trends with regard to platforms, money, team sizes, and more. (GDC intends to field a similar survey each winter, in advance of the conference in San Francisco.)

THE RISE OF THE INDIE

A key finding of the GDC 2013 State of the Industry survey is that independent game development and smaller teams are on the rise like never before. In fact, 53% of the respondents identified themselves as “indie developers,” and of those, 51% have been indie developers for less than two years.

In addition, 46% of the survey's respondents work within companies of 10 people or less. Further proving the move to indie, only 24% of those surveyed worked with a publisher on their last game, while an even smaller 20% are doing so on their current projects.

SMARTPHONES, TABLETS, PC DOMINATE DEV PLATFORMS

Another major focus of the survey was the platform preferences and interests of



the development community, an important topic in a year when next-gen consoles are anticipated to come to market—alongside a slew of new types of consoles such as PC-based TV consoles and Android consoles.

The survey found that more of the respondents are developing for smartphones and tablets than for any other platform; 38% of the survey's developers released their last game for smartphones and tablets collectively, but 55% are making their current games there. Even more impressive, a whopping 58% plan to release their next games on these platforms.

PCs and Macs are the next-strongest platforms, with 34.6% of developers releasing their last games for PCs/Macs, 48% developing their current games for those platforms, and 49% planning their next games on PCs/Macs.

CONSOLE DEVELOPMENT STABLE AT A LOWER BASE

In terms of Sony, Microsoft, and Nintendo, the survey found Microsoft at the top, albeit from lower numbers, with 13.2% currently developing for the Xbox 360 and close to 14% planning their next game on the 360. For the PlayStation 3, 13% are releasing their current game for the console, and 12.4% their next game.

In terms of the Nintendo Wii U, only 4.6% of developers are currently making a Wii U game,

and just 6.4% of our surveyed developers are making their next game for the console. (Finally, an identical 11% of respondents are making their next game for upcoming Sony and Microsoft platforms.)

DEDICATED GAMING HANDHELDS SHOW MINIMAL SUPPORT

Sony's PlayStation Vita handheld showed a slight uptrend in support, but from extremely low percentages. While less than 2% of respondents made their last game for the console, 4.2% are making their current game for Vita, and just over 5% plan to release their next game there. And North American developers are unconvinced of the Nintendo 3DS's potential: 2% are currently making a 3DS game, and only 2.8% of developers plan to release their next game on the 3DS.

SMARTPHONES, TABLETS, PC/ANDROID "CONSOLES" TOP DEVELOPER INTEREST CHARTS

The survey also asked developers about their levels of interest in developing for the variety of platforms on the market or coming soon, and received a very different response from what they said about current and future projects.

Tablets and smartphones are still way out ahead in terms of interest, with 58% and 56% respectively interested in the platforms. PC-based

TV consoles, such as Valve's Steam Box, have a very high 45% level of interest for developers. Android home consoles, like the OUYA and GameStick, are also high up the interest curve at 37%.

Interestingly, when asked simply about expressions of interest, next-gen Microsoft and Sony consoles shoot up the relative graph ranking to 29% and 27% respectively. And Nintendo platforms continue to lag in developer interest, with a relatively small 13% and 5% interest respectively for the Wii U and 3DS.

SELF-FUNDING GOES BIG, CROWDFUNDING INTEREST BLOOMS

Finally, the GDC survey looked into how developers are funding their projects. The vast majority of games are being funded from the company's existing war chest (37%) or an individual's personal funds (35%).

Only 9% of our survey respondents were primarily funded by venture capital; 10% are still primarily publisher-funded, and 4% are actually primarily crowdfunded. With regard to crowdfunding, 8% of respondents have worked on a project that was crowdfunded, while a surprisingly sizable 44% plan to do so in the future. (GDC and the GDC summits are owned and operated by UBM Tech, as is *Game Developer*.)



MAKING
GAMES

IS NOT A

HOBBY

SO STOP TREATING IT THAT WAY.

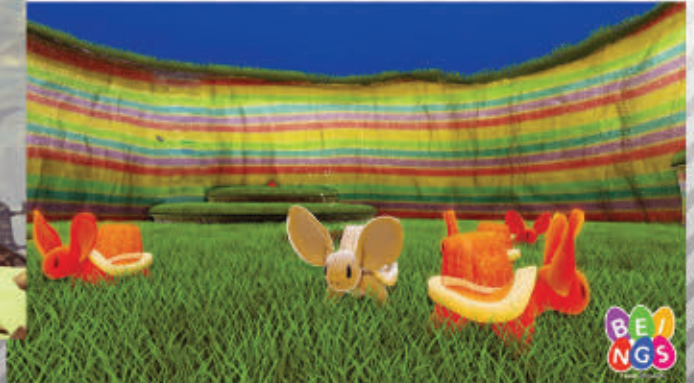
Playing games is easy – making games is hard. You want to be a game designer? Better get off the couch and sharpen your skills in level design, narrative, art, coding, and game mechanics until they could cut steel. Get serious about your career during the most intense year of your life.

Take 'HOBBY' out of your vocabulary. [VFS.EDU/MAKEGAMES](https://vfs.edu/makegames)

VFS

VANCOUVER FILM SCHOOL
GAME DESIGN

UNREAL ENGINE NEWS



The **Unreal Development Kit**, the free edition of Unreal Engine 3, continues to help aspiring game developers harness their creativity and jumpstart their careers.

Four finalist games in the latest Make Something Unreal competition have been developed around the theme 'Mendelian inheritance: human genomics and genetics,' which was set and is supported by the Wellcome Trust, a global charitable organization.

During the **Make Something Unreal Live** grand finale, teams will rapidly iterate on their games and present work daily to mentors, judges and the public at the **Gadget Show Live**, the premier consumer electronics show coming to Birmingham, UK, from April 2-7, 2013. The winning team will walk away with an **Unreal Engine 4** license for PC digital distribution.

MAKE
SOMETHING
UNREAL
LIVE
2013

See what small student teams can build with the Unreal Development Kit in six months!

Thanks to our **mentor studios** and **industry veterans** for their advisory contributions to Make Something Unreal Live 2013:

Climax Studios, Lucid Games, Ninja Theory, Splash Damage, Jon Hare, Miles Jacobson, Dave Jones, Peter Molyneux, Clive Robert, Ella Romanos, Anton Westburgh, Kostas Zarifis.

To follow the progress of MSUL 2013, visit facebook.com/MakeSomethingUnreal.



Come see Epic at upcoming industry events: **Gadget Show Live** (April 2-7, Birmingham, UK), **East Coast Game Conference** (April 24-25, Raleigh, NC), **GameHorizon** (May 8-9, Newcastle, UK)
Email licensing@epicgames.com for appointments and sign up for our newsletter at unrealengine.com.



I N D U S T R Y I N

F L U X

THE 12TH ANNUAL GD MAGAZINE SALARY SURVEY



That's the main takeaway from the 12th annual *Game Developer* magazine Salary Survey. On one hand, we saw the industry explode with creativity and new tech; on the other hand, seeing several high-profile studio closures left many worrying about the long-term outlook for their career. Overall, most game developers made more money and received better benefit coverage than last year (with roughly a 12–15% increase in medical, dental, and vision coverage across the board), but that didn't stop developers from expressing uncertainty about the industry's direction.

Every year, we ask thousands of *Game Developer* and *Gamasutra* readers to tell us what they made in the last

year, asking a slew of related questions along the way. For some numbers, the industry is looking up. We found that the average salary across the U.S. game industry is \$84,337, which is up approximately \$3,100 over last year's average. Layoffs are at 12%, down 1% from last year. 64% of developers made more money than last year, 29% made the same, and only 7% made less. When asked if they thought the game industry was a great industry to work in, 24% of developers strongly agreed, 45% agreed, 21% felt neutral, and only 7% disagreed and 3% strongly disagreed. Only 9% of developers reported being dissatisfied with their potential career path (down 2% from last year), compared to 22% who felt extremely satisfied, 41% who

felt satisfied, and 27% who felt somewhat satisfied.

However, the literal comments revealed a shared feeling that the industry was in flux; practically every comment we received spoke to the decline of triple-A and traditional console-development paths, the rise of mobile games as the new industry focus (and an associated unease with the prospects of getting noticed on overflowing app stores), distrust of a growing free-to-play bubble, and a mix of enthusiasm for indie developers' creativity, and worry about indie developers' earnings. When asked whether they thought there were more jobs in 2012 for game developers, whether the game industry was picking up, and whether there were more opportunities than ever before,

devs were much more negative than last year.

In other words, even though most numbers are going up, they might not tell the whole story. It seems like everyone knows that mobile and multiplatform is where the industry is headed, but that knowledge isn't particularly reassuring. Between the console developers worried about finding a place in the new job market, indies still waiting for their passion projects to pay off, current mobile devs underwhelmed by how their games have fared in the hyper-competitive app stores, and the possibility of another bubble forming (and bursting) on the horizon, a few extra bucks here and there isn't going to do much to assuage those fears. >>>

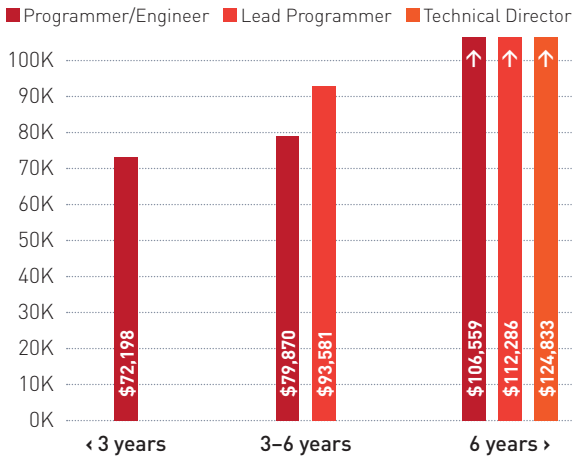
PROGRAMMERS

\$92,151
{avg. salary}

Programmers make the (game) world go 'round; once again, they're second only to the business and legal people in terms of overall compensation. This year's average programmer salary is down slightly (\$811) from last year's average. Most of those cuts, interestingly enough, came on the more-experienced side of the industry: Newer programmers reported salaries \$8,000 higher than last year's, while programmers with 3-6 years of experience made \$900 less, and those with over six years of experience made \$2,500 less. Furthermore, most of those cuts came from more senior positions; technical directors' salaries are down \$10,300 from last year, and lead programmers' salaries are down \$3,500 for 3-6 years of experience and down \$7,000 for over six years of experience. Contract programmers averaged \$62,500 this year.

Canadian programmers averaged \$70,712, which is down about \$4,300 from last year, and European programmers averaged \$43,914, which is down \$2,900.

Programmer salaries per years experience and position



ALL PROGRAMMERS AND ENGINEERS

Years experience in the industry % receiving additional income: 81%

Average additional income: \$15,797

Type of additional compensation received

Annual bonus	45%
Pension/Employer contribution to Retirement plan	41%
Profit sharing	10%
Project/title bonus	24%
Royalties	5%
Stock options/equity	33%
Percent receiving benefits:	97%

Gender stats

Gender	% Represented	Average Salary
Male	96%	\$91,969
Female	4%	\$96,136

Type of benefits received

Medical	95%
Dental	88%
Vision	86%

ARTISTS AND ANIMATORS

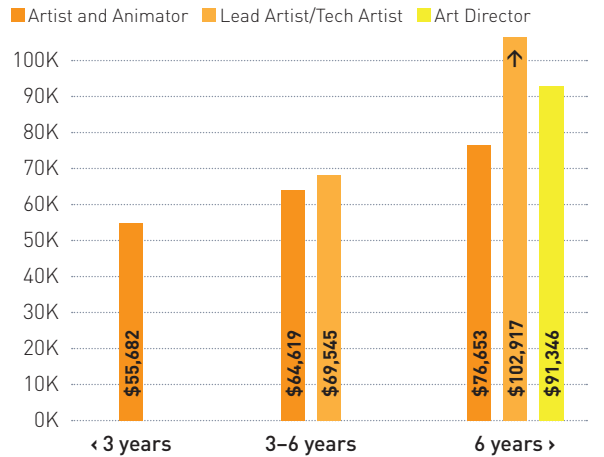
\$75,009
{avg. salary}

Artists and animators also saw a slight decline (\$770) from last year's survey average. Again, most of the losses were from more-experienced devs: Artists or animators with over six years of experience averaged \$16,500 less than last year, and art directors averaged \$30,000 less than last year. Interestingly enough, lead/tech artists with over six years of experience actually made \$10,000 more than last year, pushing their salary average above that of the more-senior art director title. Contract artists averaged \$64,741 this year.

Canada-based artists and animators averaged \$63,227 this year (down \$3,400 from last year), and Europe-based artists and animators averaged \$40,776, which is up \$5,000 from last year and puts them right back where they were in 2010.

We received 3% more responses from female artists and animators this year, bumping the gender balance up to 84% male, 16% female, and their average salary bumped up \$7,400 over last year's. Male artists' and animators' salaries fell \$1,300, but they still make about \$16,000 more than their female counterparts.

Artist and Animator salaries per years experience and position



ALL ARTISTS AND ANIMATORS

Years experience in the industry % receiving additional income: 84%

Average additional income: \$14,300

Type of additional compensation received

Annual bonus	40%
Pension/Employer contribution to Retirement plan	32%
Profit sharing	26%
Project/title bonus	30%
Royalties	26%
Stock options/equity	25%
Percent receiving benefits:	93%

Gender stats

Gender	% Represented	Average Salary
Male	84%	\$77,791
Female	16%	\$60,238

Type of benefits received

Medical	92%
Dental	88%
Vision	80%

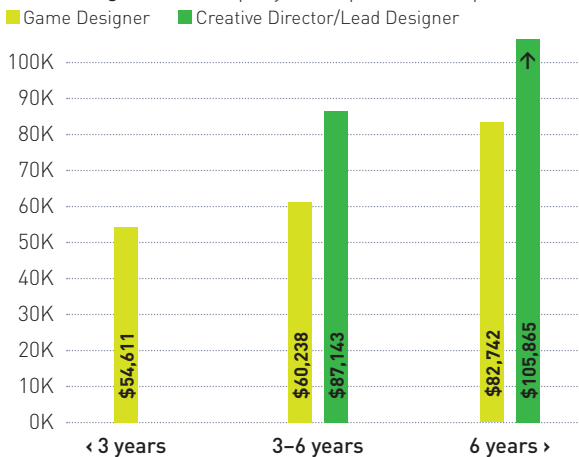
GAME DESIGNERS

\$75,065
{avg. salary}

Unfortunately, our writer respondent pool was so small this year that we weren't able to break out their results separately like we have in previous salary surveys. Entry-level game designers saw a \$6,300 increase over last year; game designers with 3-6 years of experience made \$1,500 less, and game designers with over six years of experience made \$5,000 more than last year, while creative directors and lead designers made \$16,600 more at the 3-6 year level and \$6,500 more at the over-six-year level. Considering how important design is for mobile and free-to-play games, we're not terribly surprised to see this increase. Contract designers averaged \$46,786. Canada-based game designers didn't fare quite so well, though; their \$56,576 average is down \$3,750 over last year's, while Europe-based designers' average of \$43,600 is up \$5,400 from last year's.

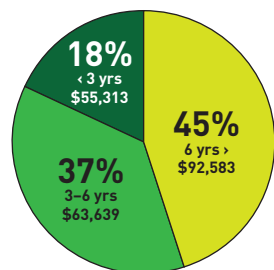
Gender balance remained the same (89% male) in the design field. Female designers' average salaries also stayed the same, while male designers averaged \$2,400 more over last year's salary.

Game Designer salaries per years experience and position



ALL GAME DESIGNERS

Years experience in the industry % receiving additional income: 79%



Average additional income: \$14,534

Type of additional compensation received

Annual bonus	36%
Pension/Employer contribution to Retirement plan	39%
Profit sharing	13%
Project/title bonus	24%
Royalties	5%
Stock options/equity	30%
Percent receiving benefits:	96%

Gender stats

Gender	% Represented	Average Salary
Male	89%	\$76,646
Female	11%	\$61,983

Type of benefits received

Medical	96%
Dental	93%
Vision	85%

PRODUCERS

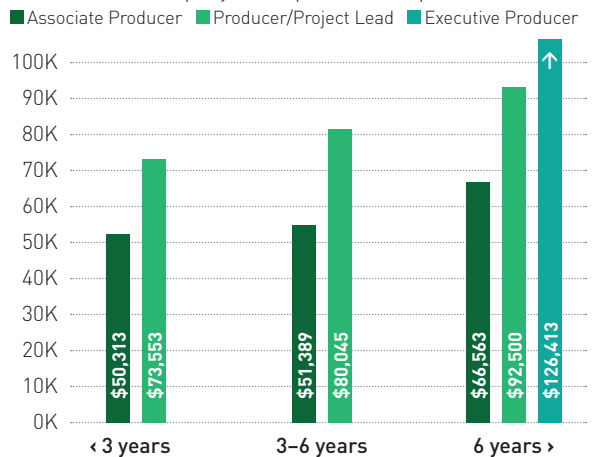
\$84,127
{avg. salary}

Producers averaged \$1,500 less than last year, with cuts felt pretty much across the board; entry-level associate producers made \$8,000 less, associate producers with 3-6 years of experience made \$4,500 less, and project leads and executive producers with over six years made \$6,000 less and \$13,000 less, respectively. Also, producers/project leads with 3-6 years of experience made \$12,000 more than last year. Contract producers averaged \$65,833.

Canada-based producers fared better, averaging \$76,875 (\$5,400 higher than last year's average), while Europe-based producers averaged \$54,167, which is \$2,200 less than last year's average.

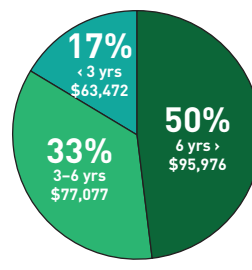
Historically, we've seen more women in production than we have in other development disciplines, and this year continues that trend with a 7% increase in responses from female producers, bringing the count to 77% male, 23% female. Overall, male producers saw a \$1,500 average salary decrease from last year, while female producers made an extra \$650.

Producer salaries per years experience and position



ALL PRODUCERS

Years experience in the industry % receiving additional income: 82%



Average additional income: \$16,454

Type of additional compensation received

Annual bonus	58%
Pension/Employer contribution to Retirement plan	40%
Profit sharing	9%
Project/title bonus	14%
Royalties	5%
Stock options/equity	35%
Percent receiving benefits:	97%

Gender stats

Gender	% Represented	Average Salary
Male	77%	\$85,591
Female	23%	\$78,989

Type of benefits received

Medical	94%
Dental	92%
Vision	85%

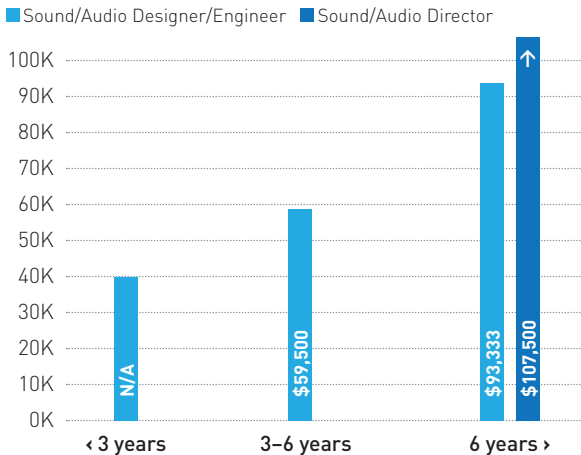
AUDIO PROFESSIONALS

\$81,543
{avg. salary}

Salaried audio professionals' average salary saw a slight decrease of about \$1,600 from last year. Note that our response rate for salaried audio workers is always fairly low compared to other disciplines; there simply aren't that many audio jobs out there, and most of the available audio gigs are contract rather than full-time permanent positions. However, we did get about 15% more respondents from salaried audio devs this year over last year, and last year we saw 30% more audio respondents than the year before, so it looks like the industry is gradually adding more full-time audio jobs. Audio contractors averaged \$110,500.

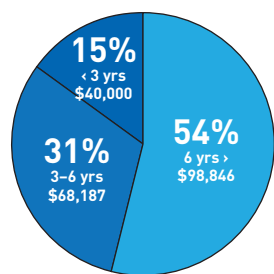
Most of this year's decrease came from devs with over six years of experience; sound/audio designers and engineers saw a \$16,000 cut, and sound/audio directors a \$3,000 cut. Sound/audio designers or engineers with 3-6 years of experience actually saw a gain of \$1,600 over last year's salary. Unfortunately, we didn't receive enough responses from entry-level audio professionals to break their salaries out.

Audio Professional salaries per years experience and position



ALL AUDIO DEVELOPERS

Years experience in the industry % receiving additional income: 78%



Average additional income: \$13,000

Type of additional compensation received

Annual bonus	44%
Pension/Employer contribution to Retirement plan	35%
Profit sharing	11%
Project/title bonus	26%
Royalties	11%
Stock options/equity	25%
Percent receiving benefits:	96%

Gender stats

Gender	% Represented	Average Salary
Male	96%	\$82,944
Female	4%	\$50,000

Type of benefits received

Medical	91%
Dental	87%
401K/Retirement	83%

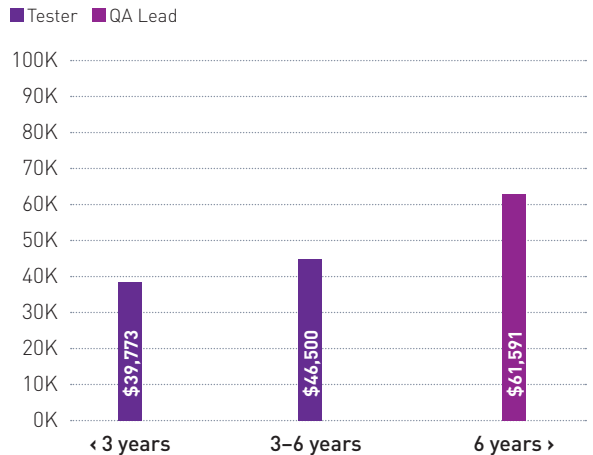
QA TESTERS

\$48,611
{avg. salary}

QA testers and leads continue to be the lowest-paid profession in game development. Overall, salaried QA testers saw a slight bump of \$700 this year, which puts them at almost twice the average QA contractor's income (\$27,237). Testers with less than three years' experience saw a \$2,200 increase over last year, testers with 3-6 years of experience saw a \$5,400 increase, and QA leads with over six years in the industry saw a \$6,400 increase. Also, while the percentage of QA devs that received additional income (besides salary) fell 2%, the devs that did receive additional income were 10% more likely to receive an annual bonus and 9% more likely to participate in a profit-sharing plan over last year. Contractors, meanwhile, only saw an average increase of \$200 over last year.

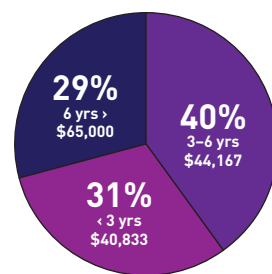
Canada-based testers averaged \$41,731, which is \$1,500 less than last year's average salary. European QA salaries also fell by \$1,200, bringing their average salary to \$31,346.

QA Tester salaries per years experience and position



ALL QA TESTERS

Years experience in the industry % receiving additional income: 75%



Average additional income: \$12,102

Type of additional compensation received

Annual bonus	59%
Pension/Employer contribution to Retirement plan	50%
Profit sharing	16%
Project/title bonus	16%
Royalties	5%
Stock options/equity	21%
Percent receiving benefits:	91%

Gender stats

Gender	% Represented	Average Salary
Male	93%	\$49,196
Female	7%	\$39,375

Type of benefits received

Medical	68%
Dental	70%
401K/Retirement	82%

BUSINESS AND LEGAL PEOPLE

\$103,934
{avg. salary}

The "business and legal people" category includes chief executives and executive managers, community managers, marketing, legal staff, human resources, IT, content acquisition and licensing, and general administration staff.

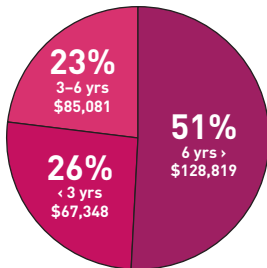
This aggregated group typically receives the highest average salary in the industry, and 2012 was no exception; its average is actually up about \$1,800 over last year. This group also receives the most non-salary income in the industry, though this year they saw a \$4,400 decrease from last year's non-salary income.

Entry-level businesspeople saw a decrease of \$4,500 from last year's average, while businesspeople with 3-6 years of experience made \$6,800 more than last year, and those with over six years of experience made \$5,000 more than last year.

The gender ratio in the business field remained at a steady 82% male, 18% female. Men's average salaries stayed about the same from last year, while women's average salaries increased by \$9,000. That said, the average businessman's salary is roughly \$26,000 higher than the average businesswoman's.

ALL BUSINESS AND LEGAL PEOPLES

Years experience in the industry



Gender stats

Gender	% Represented	Average Salary
Male	82%	\$108,571
Female	18%	\$82,292

% receiving additional income: 82%
Average additional income: \$20,473

Type of additional compensation received

Annual bonus	53%
Pension/Employer contribution to Retirement plan	30%
Profit sharing	22%
Project/title bonus	15%
Royalties	12%
Stock options/equity	33%
Percent receiving benefits:	90%

Type of benefits received

Medical	90%
Dental	82%
401K/Retirement	75%

{LAYOFF RATES}

Layoff rates continue to trend slightly downward; 12% of respondents were laid off at some time in 2012, which is 1% lower than 2011's rate and 2% lower than 2010's rate. Overall, we've seen a 7% decrease in layoffs over the last three years.

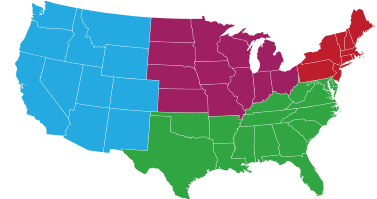
Of the devs who were laid off, 59% found new employment at a studio or publisher (up 1% from last year); 16% went into contracting or consulting (down 3%); 7% founded or co-founded a new company (down 3%); 14% went into indie game development (down 1%); 12% haven't found new work since (down 1%), and 11% simply reported "Other." (Note that for this survey question, multiple responses were allowed.) Once again, it appears that the industry is ever-so-slightly stabilizing year over year.

When it came to finding new jobs, the three most popular methods were by referral (32%), searching job postings (18%), and simply sending in a resume or CV (10%).

{AVERAGE SALARY} BY U.S. REGION

{across all levels of experience and disciplines}

West	\$90,822
Midwest	\$70,906
South	\$72,480
East	\$77,027



{TOP 10 STATES} WITH HIGHEST AVERAGE SALARIES

{across all levels of experience, excluding states with low sample size}

	AVERAGE SALARY	PERCENT WHO OWN HOMES	AVG. SALARY OF HOMEOWNERS
1 Washington	\$90,907	45%	\$106,951
2 California	\$87,561	27%	\$111,219
3 Oregon	\$87,500	45%	\$107,500
4 New Jersey	\$85,833	41%	\$99,500
5 North Carolina	\$82,000	62%	\$93,056
6 Virginia	\$81,250	44%	\$102,500
7 Maryland	\$78,636	33%	\$94,808
8 Nevada	\$78,145	48%	\$87,500
9 Colorado	\$78,125	45%	\$93,750
10 Michigan	\$76,136	41%	\$91,667

{AVERAGE SALARY} BY U.S. REGION BY DISCIPLINE

	EAST	MIDWEST	SOUTH	WEST
Programmer	\$83,375	\$75,577	\$78,777	\$101,168
Art and Animation	\$78,041	\$63,500	\$63,773	\$79,304
Game Design	\$71,354	\$60,000	\$68,654	\$78,869
Production	\$75,694	\$63,125	\$75,203	\$90,248
Audio	\$N/A	\$N/A	\$N/A	\$82,770
QA	\$N/A	\$N/A	\$N/A	\$54,464
Business	\$84,167	\$N/A	\$78,333	\$114,457

{AVERAGE SALARY} FOR HOMEOWNERS VS. NON-HOMEOWNERS BY U.S. REGION

	EAST	MIDWEST	SOUTH	WEST
Homeowners	\$88,049	\$85,611	\$86,167	\$106,243
Non-Homeowners	\$68,039	\$55,000	\$58,392	\$77,176

{AVERAGE SALARIES} IN THE U.S., CANADA, AND EUROPE

{across all levels of experience, by discipline, given in USD}

	U.S.	CANADA*	EUROPE**
Programmer	\$92,151	\$70,712	\$43,914
Art and Animation	\$75,009	\$63,227	\$40,776
Game Design	\$75,065	\$56,576	\$43,600
Production	\$84,127	\$76,875	\$54,167
Audio	\$81,543	\$77,143	\$41,071
QA	\$48,611	\$41,731	\$31,346
Business	\$103,934	\$78,750	\$72,652

*Most Canadian respondents were from British Columbia (20%), Quebec (44%), and Ontario (24%).

**Most European respondents were from the United Kingdom (18.2%), Germany (17.9%), France (11.5%), Poland (11.4%), and Sweden (7%).

{AVERAGE SALARY} BY EDUCATION LEVEL AND DISCIPLINE

(across all levels of experience)

	PROGRAMMING	ART	DESIGN	PRODUCTION	AUDIO	QA	BUSINESS
High school/GED	N/A	N/A	N/A	N/A	\$99,583	N/A	N/A
Some College	\$86,250	\$84,405	\$106,786	\$91,538	\$79,808	\$104,842	\$48,750
Associates Degree	N/A	\$77,344	N/A	N/A	\$65,682	\$89,375	N/A
Bachelors Degree	\$77,187	\$73,350	\$101,100	\$86,084	\$70,508	\$88,630	\$47,321
Some Graduate	N/A	\$60,357	N/A	\$81,346	\$77,833	\$94,817	N/A
Masters Degree	\$105,000	\$71,875	\$98,636	\$74,551	\$74,600	\$94,872	\$25,000

Everything after master's degree was omitted due to insufficient responses.



{METHODOLOGY}

Now in its 12th year, the *Game Developer Salary Survey* was conducted in February 2013 for the fiscal year January 1, 2012 through December 31, 2012 with the assistance of Audience Insights. Email invitations were sent to *Game Developer* subscribers, Game Developers Conference attendees, and Gamasutra.com members asking them to participate in the survey.

We gathered 4,042 responses from developers worldwide, but not all who participated in the survey provided enough compensation information to be included in the final report. We also excluded salaries of less than \$10,000 and the salaries of students and educators. The small number of reported salaries greater than \$202,500 were excluded to prevent their high numbers from unnaturally skewing the averages. We also excluded records that were missing key demographic and classification numbers.

The survey primarily includes U.S. compensation, but consolidated figures from Canada and Europe were included separately. The usable sample reflected among salaried employees in the U.S. was 1,520, for Canada 367, and for Europe 527; and 422 for indies and independent contractors who provided compensation information worldwide.

The sample represented in our salary survey can be projected to the U.S. game developer community with a margin of error of plus or minus 2.6% at a 95% confidence level. The margin of error for salaried employees in Canada is plus or minus 5%, and is 4.2% for Europe.

THE INDIE REPORT



This is the fourth year we've collected data for our indie report, where we survey individual independent developers, independent teams, and contractors for their perspective on the industry. Individual indie developers' average income of \$23,130 is \$420 lower than last year's average, while members of indie teams reported an average of only \$19,487, which is down \$20,000 from last year's average. (Note that last year's average was up \$26,780 from the year before that, so some rather drastic fluctuation in this number appears to be rather common.) When it comes to indie game sales revenue, the results are still rather spread out. Half of indie developers made less than \$500 from the sale of their games (which includes in-app purchases and DLC); 13% made between \$500 and \$3,000, 15% made between \$5,000 and \$30,000, and 5% made over \$200,000. Alternate sources of income (advertising, awards/grants, sponsorship opportunities) remain hard to obtain; 79% of indie devs didn't make any money from these methods at all. Of the devs that did, 25% made less than \$100, 28% made between \$100 and \$2,000, 22% made between \$2,000 and \$10,000, 5% made between \$10,000 and \$20,000, and 20% made over \$20,000.

JOB FUNCTIONS When it comes to indie job functions, we decided to change the

survey this year to reflect each developer's primary contribution. We know that being an indie dev requires wearing multiple hats, but we wanted to find out which disciplines indies specialized in. 40% of indie devs reported their primary role was programming, followed by 19% in design, 12% in art, 11% in QA, 8% in production, 8% as "other," and 2% in audio. Programming, design, and art are, understandably, the most popular primary disciplines (and perhaps the most crucial to the nuts and bolts of indie game creation); production, meanwhile, appears to be a role that indie teams simply can't afford to bring on specialists to handle quite yet, and audio development continues to be a rather niche role (most likely one that is contracted out with indies, just as it is with mainstream game development).

For contractors, the most popular discipline is QA (24%), followed by art (19%), programming (17%), audio and design (10% each), other assorted roles (8%), production, (7%), and writing (4%). There hasn't been much significant fluctuation in the respective proportions of contracted dev roles between this year's survey and last year's survey, so it seems as though dev studios aren't changing the way they handle which roles need salaried employees and which roles to contract out. survey, so it



seems as though dev studios aren't changing the way they handle which roles need salaried employees and which roles to contract out.

{CONTRACTORS} BY JOB FUNCTION

Art	19%
Audio	10%
Design	10%
Production	7%
Programming	17%
QA	24%
Writing/Scenario	4%
Other	8%

{INDIES} BY JOB FUNCTION

Art	12%
Audio	2%
Design	19%
Production	40%
Programming	8%
QA	12%
Other	8%

SALARY SURVEY COMMENTS

THE BAD

"IT WAS MORE ENJOYABLE WHEN IT WAS LESS MATURE."

QA is undervalued and not compensated fairly."

I'm seeing the failures [some spectacular] of more and more studios lately. New ones are sprouting up as well, but it doesn't feel like there are as many new ones as there are failing old ones. I worry about long-term sustainability for my career as I continue to get older [I'm 41 now]."

When I got my first industry job in 2005, it felt like there were all these Sure Bet Career jobs out there. Now, less than 10 years later, I can't think of a single job that will be safely guaranteed to be around for 5 years."

I hear dentistry is in demand."

We're a bit stuck in the mud. I don't see a whole lot of pure innovation (but I'm not sure that's really what people want anyway). I'd like to see some honest excitement in games again because I think we're getting a bit predictable."

It's difficult to get in to my line of work. All the jobs that exist are filled. Companies that don't have writers or editors can't be convinced that they need them. It's really a pain."

THE GOOD

"IT WAS REFRESHING TO SEE SMALLER, MORE UNIQUE GAMES GET RECOGNITION THIS YEAR."

Lots of turnover, but lots of new opportunities for smaller companies."

I absolutely love the industry I work in. I can't imagine any other career track. Quality of Life is

picking up, crunchmongering developers are dying off, and new business models are supporting innovation like never before."

The variety of opportunity [given the huge rise in casual and independent games] is greater than ever."

Not always the highest-paying option, but the game industry is the most rewarding career path I could imagine."

There's no better way to earn a living. While it has its ups and downs and unique challenges, I'm very happy to be working within it, and hope to do so for a very long time."

F2P

"F2P RULES."

Free-2-play. Do you speak it!?"

Monetization sucks."

This current influx of quick-cash-grab F2P and social games is strongly reminiscent of the early '80s pre-crash boom."

A bit sad that it's now focusing on monetization [more] than ever."

It's been a downward spiral. Soon, you will have to pay people to play your games. In fact, it's already happening!"

CONSOLES

"2012 WAS THE FIRST YEAR I NOTICED OUR COMPANY STRONGLY RECOGNIZE THE IMPORTANCE OF PERSONAL DEVICES AND HOW THEY CAN ENHANCE A CONSOLE EXPERIENCE."

I've worked for a major first-party developer for over 15 years and they've never acknowledged the existence of anything besides their

own platform. Now they're realizing that strong titles may need to include multiple devices, some of which may not be made by themselves."

I would feel like an outdated dinosaur developing for consoles... even unreleased hardware. Mobile is clearly king, and developers must react or continue to shut their doors."

The bloodletting in console dev is scary, especially since mobile/indie doesn't seem to have the \$\$ to pick up the talent."

MOBILE

"THE MOBILE AND CASUAL SPACE IS QUITE THE EXCITING AREA TO BE IN."

Mobile games suck."

The mobile space is very competitive, and not very profitable."

The obsession with mobile platforms taking over is just another trend. Of course mobile is and looks to remain a very viable platform for monetization; however, developers should stay more focused on pleasing customers than trying to figure out the next big profit wave to ride. That'll be the key to a respectful future."

Mobile/casual games are a scary potential direction. The games the casual market wants to play are not the games I got into the

industry to make. I would, ideally, never want to work on creating a low-budget, monetizing treadmill."

INDIES

"THERE ARE MORE OPPORTUNITIES FOR INDIE DEVELOPERS, BUT LESS FOR EVERYTHING ELSE."

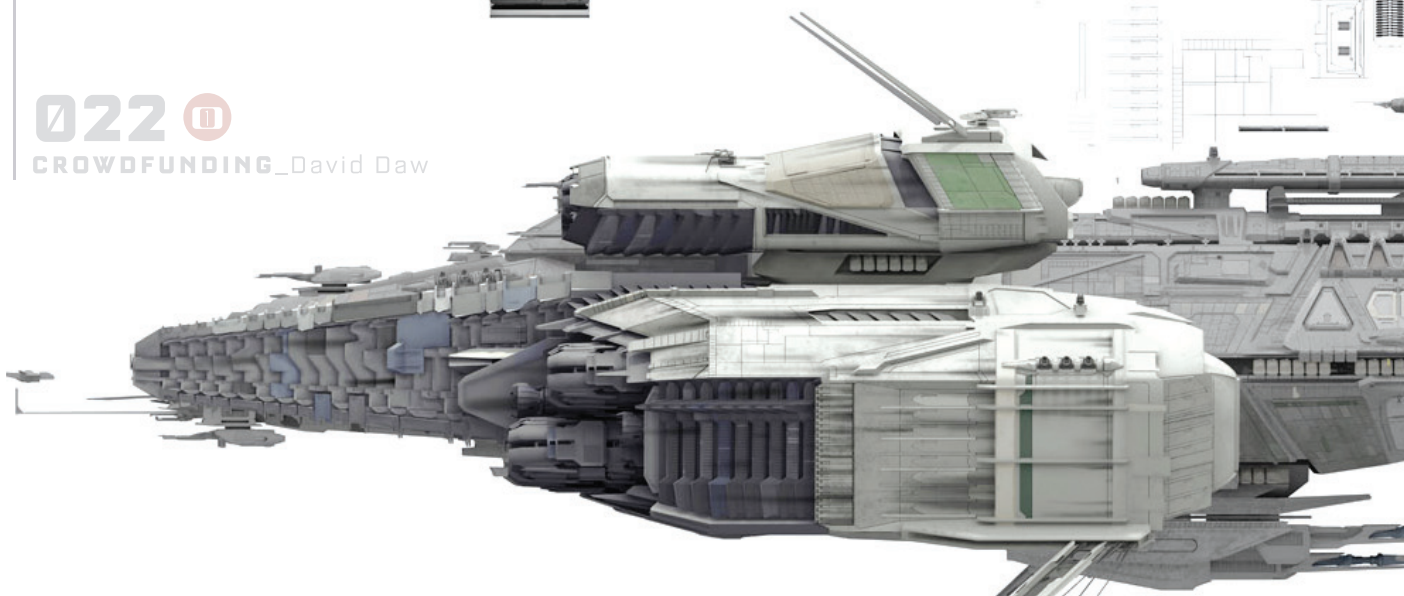
There are more ways than ever now for indie game developers to do well and publish their games."

In 2012 I felt like a drop in the app-store ocean, and that as an indie, I had neither the development resources nor the marketing budget to compete. That has since changed in 2013 as I'm now developing for the OUYA, and it feels great to be part of a small but growing community with prospects, and be involved at something from the ground floor."

It's a great time to be an employee in mobile/web, but it seems like financing is drying up for people who want to start their own companies. "Going indie" isn't really viable in expensive places like Silicon Valley, so the "day job" can feel like a prison at times."

There is an obvious and exciting increase in the number of opportunities for game developers on an individual and independent level. Anyone who wasn't working on a personal project in 2012 is falling behind."





CROWDFUNDING

ONE YEAR LATER

BY DAVID DAW

WHAT DOES THE CROWDFUNDING LANDSCAPE LOOK LIKE FOR GAME DEVELOPERS ONE YEAR AFTER **KICKSTARTER** EXPLODED ONTO THE SCENE? WE ASKED DEVS BEHIND SEVERAL MAJOR CROWDFUNDING **PUSHES FROM THE LAST YEAR**—CHRIS ROBERTS (*STAR CITIZEN*), BRENDA ROMERO (*SHAKER*), GREG RICE (*DOUBLE FINE ADVENTURE*), AND JIM ROSSIGNOL AND JAMES CAREY (*SIR, YOU ARE BEING HUNTED*)—FOR THEIR THOUGHTS AND **ADVICE ON WHAT WORKED** FOR THEM AND WHAT THEY'D DO DIFFERENTLY.



CHRIS ROBERTS, STAR CITIZEN
ROBERTS SPACE INDUSTRIES

DAVID DAW: *You funded STAR CITIZEN with a rather unorthodox combination of in-house crowdfunding, Kickstarter, and traditional investment. How'd you come up with that mix?*

CHRIS ROBERTS: Yeah, we're not typical. Everybody talks about our Kickstarter, but that was just one small part of what we did. In fact, most of our money's been raised outside of Kickstarter.

I've always felt like there was a pretty strong community that were fans of my previous games, and fans of space sims in general. I felt like if I looked at everything going on through Kickstarter, a lot of it was really good—but once you did the initial campaign, it was done and over, and I was fairly disappointed with what happened after that. I felt like no matter what, you've got to have a place where all the people that backed you are going to hang out, listen to what's happening with the game, and interact, so why wait to do that later on?

So we actually launched a teaser site for what we were doing a month before we started the crowdfunding campaign—the idea was to aggregate the really diehard fans. We'd gotten about 30,000 people to sign up and register when we launched the campaign, which gave us a bit of a leg up in terms of the initial awareness in crowdfunding.

I think it really just came out of the fact that you've got to have your own solution, even with Kickstarter, because a Kickstarter campaign ends at some point. So then you've got to have some way that you're interacting with your community, and everyone always has to have some kind of option for PayPal or whatever, so we figured, "If you have to build it anyway, let's just build it and do it upfront."

DAW: *Why did you decide to continue the campaign on Kickstarter?*

ROBERTS: Later on, we went to Kickstarter because the one thing we didn't count on is that it's not so easy to build a robust scalable site if there's a lot of demand on it. We spent most of the money on the prototype, so we didn't have a lot of money to spend on a really big,

ecommerce-style robust site; we built on top of WordPress with some plug-ins, and it kinda collapsed under the weight of the initial interest. Basically, we experienced the downside of building our own site, but at the end of the day, I still think it was a good decision because we're still raising money. We've collected over \$2 million since we closed the actual count for the crowdfunding campaign, so I think we're at 8 and a quarter million right now, which is pretty amazing.

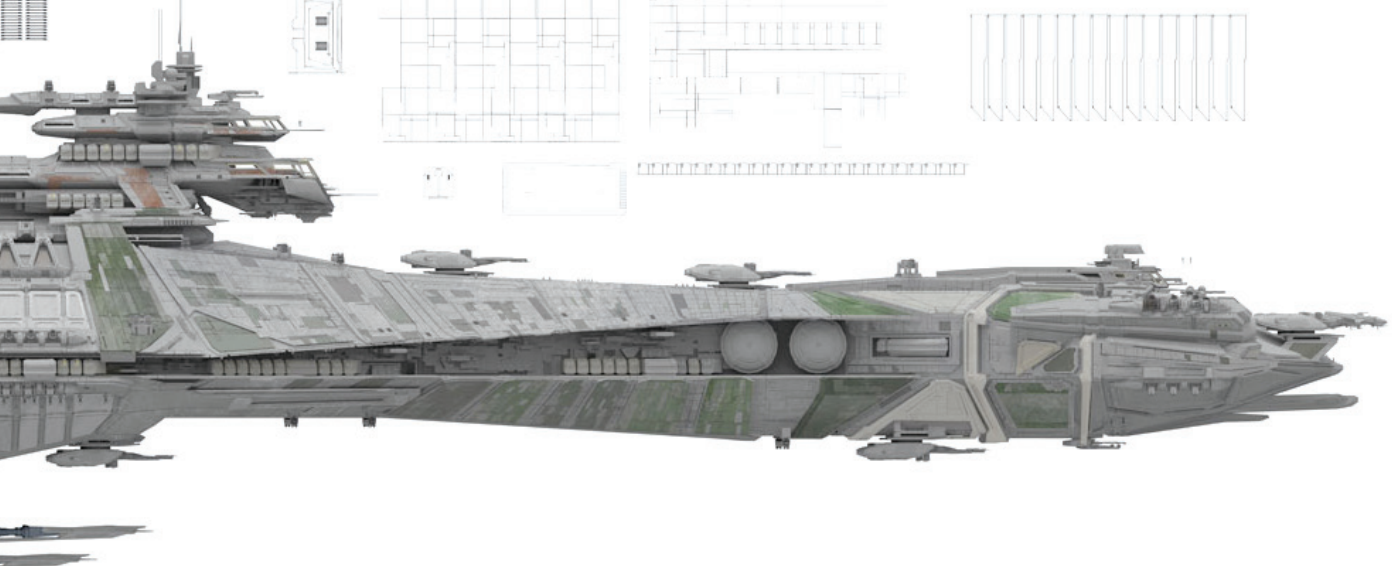
DAW: *How does pitching to a traditional investor differ from pitching to Kickstarter or to fans in general?*

ROBERTS: It's very different. Pitching on a crowdfunding campaign is really similar to pitching to your audience when it's close to done, saying "Here's what I'm going to build, this is what it is, this is why it's cool." It's like a really long lead time preorder. You read the E3 reveal, and do the exclusive interviews, and people go down to GameStop and reserve their copy. Crowdfunding, in a game sense, is like that, but you're a year or two years ahead of that schedule.

So you focus much more on the game. If I was pitching to a traditional financial investor, half of them don't even play the games, so it's a whole different thing. They just go "Oh yeah, video games, that makes some money." You're basically saying "Wargaming.net has *WORLD OF TANKS*, and they're doing X million dollars per month, and if we do really well, we can become like them and your investment will be worth a lot of money, so you should invest in us." It's a very different logic.

I actually much prefer pitching to the crowd because what you're pitching is what you're building. When you're pitching to investors, basically, they're assuming you'll make a decent game, but want to know the business reasons for it. It feels more pure and more connected to what you're doing in crowdfunding as opposed to traditional investment.

DAW: *STAR CITIZEN had a lot of really impressive visuals ready when the campaign launched. How do you decide what "done" is for your campaign?*



THE STATE OF GAME-DEVELOPMENT CROWDFUNDING IN 2013

FUNDING

How do you decide when you have enough content to get people excited?

ROBERTS: I think it sort of depends on how you approach it. I looked at everything and I said, "I think I can raise some money from a nostalgia standpoint," but I thought if we really wanted to hit it out of the park, we should treat it exactly the same way we would treat it if I was funded by EA or Microsoft: At some point, I would reveal it at E3, and then there would be a year of press hype, leading up to the release, and you're basically doing that to get people excited and hyped. So I sort of viewed going into the crowdfunding campaign with the same level of attention and respect that I would if I was doing it from the traditional funding route.

That's not normal for crowdfunding stuff. I spent a year on the prototype because I felt like I needed to do something that was pretty ambitious, and I needed to show everyone what it would look like and the feel of it. If you treat it more like you would if you were funded by a major publisher, my guess is you'll do better on the crowdfunding front, because there's something more tangible.

When you see somebody sitting in front of a camera saying, "Hey, I really want to revolutionize this kind of game and change this," there's only so much you can take. If you're a name, they'll give you some credit if they've seen what you've done before and they've liked it, but nothing speaks louder than some really great imagery on the scene.

I think you'll see that the sort of crowdfunded campaigns that are doing well, or will do well, are the ones that are able to show much more up front, show what you're going to get as someone who's backing it. I think you've gotta treat that really seriously. Last year, at the beginning of this whole craze, you could get away [with] nostalgia, but I think that's a lot harder to do nowadays.

DAW: *You said you spent about a year on the game before launching. How much time did you spend on the PR campaign for STAR CITIZEN?*

ROBERTS: I started lining up the press and the whole campaign about three months before I made the formal announcement,



STAR CITIZEN.



maybe four months. I've done this before, with EA, and Microsoft, and Origin before that. I have an advantage because there's a track record and I have my relationships, so I sort of knew the gig.

I did a press tour for a week before we did the announcement in Austin, where I went to Germany, I went to New York, I went to San Francisco, and I sat down with a bunch of key outlets and showed them my prototype and talked them through what I wanted to do because I just feel like that's really important to get people excited.

The other thing that was important to me in the year before the campaign was to research how the game's going to get built, and what the issues are—how long it would take to build a spaceship or a player with X polys? I wanted to get a good gauge. If I'm promising something to the crowd, I want to really make sure that I've done my homework, because when you're making promises you want to deliver on them.

It's great when everybody gives you all this money, and when you don't deliver on it, people will give you a little bit of leeway because they know making something good takes some time—but you can't screw up. I definitely think I've seen some crowdfunding where I look at the campaigns and I can tell that the people behind them are basically naive. They've got the best intentions, but they've never made anything that hard to make.

One of the biggest things of the preparation year was doing a lot of R&D on things that I'd need to know for the full production, and any issues we'd see, so once we raised the money we'd be in a production phase instead of an R&D phase. If you're really going to take it seriously and do it well it's a bunch of work to get it teed up and ready.

DAW: *How does having to answer to the crowd change development cycles in terms of deliverables and making sure you hit deadlines?*

ROBERTS: I definitely feel that with the crowd, I've got more pressure to deliver, and if I'm not going to deliver on a date, I need to give them a really good reason, and explain it to them, and be up-front about it. With a publisher, you sort of have this relationship that you kinda know. If they're into it, you know you'll be able to say, "I need a few more months," or, "I need some more money." Typically, unless the publisher decides to just write you off, you'll be able to do that to some degree.

I feel like you've got less give with the crowd. I don't necessarily have an issue with that. It's kind of invigorating, because it makes you focus. To give you an example, on the October 10th GDC Online reveal I did, if I didn't have that date I'd have probably spent another two to three weeks polishing the prototype and the demo, because that's just my nature. I want it to look as good as possible. But, I had that day so there was a certain point where I had to get it out there... That side is a little scary but it's definitely motivating.

I do think that what the crowd does add to the development process is that they help you focus on what's important in your game much earlier. Having 100,000-plus people who love this game so much they're giving you money before the game is made gives you a really great focus group in terms of their hotspots, and the top five things they want to do in your game. Too often in a typical development cycle, you go off and work for two, three, four years, and you don't really have that direct communication.



BRENDA ROMERO, SHAKER
LOOT DROP

DAVID DAW: *Do you think that the Kickstarter has shifted? Has the wave of big Kickstarter-funded games passed?*

BRENDA ROMERO: Kickstarter in and of itself has become a game. It's a spectator sport, and it's super fun to be involved in these projects. It's fun to watch them succeed, and it's fun, in a sadistic game, to watch them fail. Watching people succeed and watching people fail, for better or for worse, as humans, there's something to that.

I think people have a limited amount of funds to spend on Kickstarters, and I think the market is a lot more crowded than it used to be. I also think in the early days there was a lot of press coverage of "Here's some RPGs on Kickstarter you might like," and you're not seeing as much of that these days. So I think there is a bit of atrophy in the community and apathy in the community. There's not as much money because the money there was to go around has gone around. Kickstarter really is its own social network, and it's incredibly fun to see what's on there, but that wanes after a while.

DAW: *Many devs think we've settled on a Kickstarter format where a lot of work has already gone on, and the developer is asking for a little money to put them over the top. Is there a standardized format for successful Kickstarter games we're settling into, or do you think the model will get shaken up again?*

ROMERO: The Kickender, I guess you'd call it, is a possibility, but there's also the "We're done, help us get enough money through preorders to actually publish the thing." So the game is done, but you need the money for community management or what have you, so the whole thing is basically a preorder platform.

You said something about "successful Kickstarters," and it's funny to me, because I've never viewed the games that didn't get funded on Kickstarter as "failures." They're just not there yet, or there's some gem of fun there that was interesting enough to make me want to play the game. Kickstarter actually prevents a bigger failure—an actual failure would be to spend a goodly sum of cash to create a game that wasn't as fully realized as it could be.

So one can view Kickstarter as milestone zero, just like when you're passing through a publisher with a pitch. When I meet with a publisher and they say, "I like this element here, and I like this, but how about if we did something a bit different?" I don't call that a failure—that's game design, that's iteration.

That was my first response [to SHAKER].

"What's not right here? What do we need to do?" and that's when it became obvious that we were spending more time addressing the weakness of the pitch than we were building the world, so better to walk away and come back with something stronger.



DAW: *Do you feel like Kickstarter pushes success and failure in binary terms? That Kickstarter has less of a give and take than going to a traditional publisher, for instance?*

ROMERO: It's tough, because when I'm just making a game, for instance, way ahead of my concern for story is my concern for systems, and how the systems are playing and feeling. But with Kickstarter, you have to come out with it all. A year or so ago, people looked at Kickstarter as the silver bullet for game development, but Kickstarter shares some of the same hurdles we find in traditional game development. You have to please the board, whether that board is 9,000 individuals, or backers, or five people sitting around a table, or one person in charge of green-lighting your idea.

If the crowd doesn't like your pitch, so be it. It's better to know sooner rather than later. Naturally, it doesn't feel great, but it also doesn't feel horrible. There's this weird, wonderful day of acceptance. Once you fail, interestingly enough, you don't feel afraid of doing it again, because the world didn't change. Nobody showed up to take away my game developer card. It's a wonderfully humbling experience. Failure's okay. What matters is debugging after the failure.



GREG RICE, DOUBLE FINE ADVENTURE
DOUBLE FINE

DAVID DAW: *What have you learned about crowdfunding since the campaign closed?*

GREG RICE: We actually didn't run into too many surprises, because we planned out a lot of stuff, like reward structures, really thoroughly before we started... Early on I wasn't expecting as many support requests from people, say, wanting to change their emails, or change their shipping addresses, or dealing with lost packages, and things like that. But we did hire a community manager and a fulfillment agency to help us out with that kind of stuff, and because of that it ended up going pretty smoothly.

Ultimately, we've been pretty successful because we have a very talented, very senior team working on this project and they've made a lot of games together here at Double Fine before. They know what they're doing, and they've been able to make a special game. Then, with the 2 Player guys, we've been able to keep our backers informed of what's happening, and they've been able to feel like they're aware of what issues we're facing (if we are facing issues) and why they should be excited about the game.

DAW: *Does Kickstarter make you feel any more or less production pressure than normal?*



SHAKER concept art.

RICE: It has been a learning experience. It was difficult for us, because we didn't have too many touchstones to look at and base our project around. I started working on putting our Kickstarter together around November of 2011, and we launched that next February. So it definitely took a long time to track down answers to all the questions I had, and to make sure we'd thought through everything, from how our reward structure was conceived, to being able to actually get codes and deliver them to backers.

A lot of things that a publisher typically handles on a game, like testing, and distributing builds, and marketing, we'd done pieces of those on our PC games, and we had started to do our marketing before, but now we had to take all of those on. A lot of support issues come out of Kickstarter as well, like having 9,000 backers that we needed to answer to.

We definitely have settled into it, though, and now that we're deep into production on the game, things are feeling great. It's really freeing to not have to answer to those kind of strict



DOUBLE FINE ADVENTURE.

“

THINK THROUGH YOUR REWARD STRUCTURE. MAKE IT SOMETHING THAT'LL BE APPEALING TO FANS BUT ISN'T GOING TO WASTE YOUR BUDGET. MAKE SURE YOU HAVE A GOOD MARGIN OF HOW MUCH MONEY IS GOING INTO THE GAME VERSUS HOW MUCH IS GOING INTO THE KICKSTARTER. ONE OF THE THINGS THAT I WISH THAT WE HAD DONE DIFFERENTLY WAS THAT WE LOPPED A LOT OF THIS CONTENT OFF TO ONLY KICKSTARTER BACKERS.”

DOUBLE FINE ADVENTURE.



milestone schedules. On our end, we are still building schedules and working toward all those milestones and dates, but it's nice to be able to be flexible with them, make changes as we see fit, and make sure the team is working on what's going to be the most important thing at that moment, without having to answer to a milestone structure that had been set years in advance.

DAW: *Does crowdfunding change your relationship with the audience in any way? Does it change how you're developing the game?*

RICE: That was definitely one of our big worries from the start with Kickstarter. Once we had thousands of fans invested in the game who have already given us money, it was just going to be a matter of figuring out how to continue to keep them up-to-date, on how the projects were going, without severely impacting the project itself.

That was one of the reasons we got really excited when the 2 Player Productions guys came on board. We thought that would be a really nice opportunity to let them in on the process, and see how things are going, and also be able to take part in it a little bit, without really taking resources away from the game. They kinda can just stay in the background, and watch things, and piece that together into a story, and make it entertaining for people to figure

out where the project is and what's going on.

In that sense, it's been awesome because the backers have been so supportive and every time we come up against something that's a bit difficult, they're all just really trusting of us and supportive, and in some cases have even pitched in some ideas that have helped to move us along, so that's been rad.

DAW: *Has the success of your Kickstarter changed the way you pitch and green-light projects internally?*

RICE: I think it has let people get a glimpse of what Double Fine is like, and helped define our brand, but ultimately we've always wanted to be more involved with our community. The Kickstarter has kinda forced us to spend a lot more time on working with our community. It's also allowed us the freedom to be able to do that—since we don't have to worry about a publisher, we're able to speak about a game really early. It's been nice to be able to do that, and to see how excited fans get about that stuff. Once we saw how much they loved every bit of information they've been getting on games, we really wanted to start doing that more on our other projects as well. We've tried to put a lot more effort into being more vocal on our website, and blogs, and Twitter, and Tumblr, and trying to put more



KICKSTARTER TIPS

CHRIS ROBERTS

➤ Have a very clear pitch, where you can visually understand what the game is. Do that well, and you don't even need to be well-known. There were three kids in Montreal that did this thing called *CASTLE STORY*; they raised \$700,000 and they'd just graduated from university. I sort of liken it to the way you go into a movie theater and you see the trailers, and you sort of know if a movie's going to be funny, or full of action, or cool, from the trailers.

Also, you need to engage your community on a daily basis. Kickstarter campaigns get really exhausting because it's not an "I do it from 9 to 5, Monday to Friday" thing; you've got people all around the world in different time zones wanting to back something. Ask them questions and show them new stuff all the time. On top of all that, if you manage to listen to what the community says and come up with some stuff that folds in their input, that really pays off. Your community is really important. When they feel engaged with you they're your best marketing.

BRENDA ROMERO

➤ Go in with the minimum amount of money it will take you to do this. Don't ask for the funding necessary—ask for the smallest funding you think you can make the game with, because the days of "This is what it will cost to do this" are over. People are expecting developers to shoulder part of the burden. There's a term out there, "Minimum Viable Product." I believe with Kickstarter that it's really Minimum Viable Ask: "If we get this amount, we will not be doomed." You don't want to be one of the people who has a successful Kickstarter but never shows up with a finished product.

Before you launch your Kickstarter, share it with as many critical eyes as possible, particularly those in the development community. Link to the thing before it launches. And share it with people who give you negative feedback. Look for that stuff. Ask about what rewards they don't like, and what they think is missing. I can't tell you the amount of Kickstarter [notices] I get saying, "Just wanted to let you know our Kickstarter is live!" but I rarely get asked "What do you think about this Kickstarter we're going to launch?"

Some of the best feedback that I got on the *SHAKER* Kickstarter came after its launch from game developers who wished I had shown it to them before we launched, and that feedback addressed problems I very easily could have fixed. The wisdom of crowds will determine if you get funding. You have access to those crowds before you launch, so reach out to them.

GREG RICE

➤ One of the major things in conceiving everything around your Kickstarter page, including your videos and your messaging and your rewards, is to really just think like a fan. I was able to do that because I was a fan of *Double Fine* before I was here, so that was easy. When you're planning to pitch your game, you're trying to pitch a publisher on why this game's going to be successful, and why it's going to make them money. On Kickstarter you're not talking to somebody who cares about that. You need to think more about why the fans should be excited about this game, and find a hook in the story around your Kickstarter that makes it something that the fans should be excited about, and something that could only work on Kickstarter. They'll read through anything that you do that is super beneficial to you, but not for them.

Also, make sure to pore over every detail, and ask yourself all the questions you think a fan would ask if they were reading it, because it's going to happen. As soon as we launched, people wanted to know about platforms, and wanted to know if there was going to be DRM. Think through all of those details so you have an answer for them.

Be prepared for the amount of PR and support that'll be coming afterward. With so many eyeballs on the project, there's going to be a lot of questions, and those questions come, many times, in the form of direct messages. Ultimately, we felt like we needed to respond to all of them and we were getting dozens of those a day. As soon as you launch, it's going to be a straight month of PR, of trying to track down interviews, and trying to get people to talk about your project as much as possible, and trying to get people as excited as possible. So make sure that once it launches you're ready to fully support your Kickstarter for the entire month.

JAMES CAREY

➤ Plan, to some extent, the whole campaign. That doesn't mean you need to have every day of your whole month campaign worked out, but you should have some idea of what content you're going to be able to add to that to build the momentum of that drive.

JIM ROSSIGNOL

➤ Plan your updates. The other thing I would say is to expect loads of queries from people via Kickstarter. That's one thing a lot of people don't realize: Kickstarter's backer messenger will give you messages, and you'll get hundreds of them every day. Be prepared to answer every single one of them personally, because you need to do that.

things out there for our fans to enjoy. It seems like it's working.

DAW: *It seemed like that was especially noticeable during the recent *Amnesia Fortnight*. A lot of things that would normally be small notes to send to the team seemed to be put up on the public forums.*

RICE: We've always had internal forums for people to throw stuff back and forth on the project, and for *Amnesia Fortnight* we kinda just moved those to the external forums. People were posting music, and concepts, and everything to the forums as we were going, and fans were definitely eating it up. We were doing eight-hour livestreams, and you would see the same people just sitting there all day, every day, for those two weeks. So I think that there's a lot of people that love video games, and are interested in the industry, and want to find out more about what a job in games looks like. Unfortunately, there hasn't been a good thing for them to look at, and I feel like people are kinda eating this up because it's such an intimate, transparent look at a day in a video game studio and what that looks like.

DAW: *What are some Kickstarter problems that you guys maybe sidestepped that you'd warn other developers about?*

RICE: Think through your reward structure. Make it something that'll be appealing to fans but isn't going to waste your budget. Make sure you have a good margin of how much money is going into the game versus how much is going into the Kickstarter. One of the things that I wish that we had done differently was that we lopped a lot of this content off to only Kickstarter backers.

We have dozens of forum posts and hours and hours of documentary footage that's exclusive to backers, so they've been getting a good idea of what's been going on with the game, and what development is looking like. Outside of that though, a lot of people are kinda clueless about what's happened. We see a lot of articles about "What's going on with *DOUBLE FINE ADVENTURE*? It launched a year ago and we haven't heard anything about it," because we promised to keep everything exclusive to backers.

Now, as we're getting closer to our traditional marketing campaign for the game, we'll start doing things like teasers and trailers, and being at events, and things like that, so people will start seeing it. But it was a little strange, just because this period of the game came way earlier than people usually hear about a game. Usually you don't hear about a game until it's in alpha. We announced our game before it was even an idea, or even had a team behind it. So, figuring out how to navigate those waters and show a game to fans in early stages before they're used to seeing it has been difficult.



JIM ROSSIGNOL & JAMES CAREY
SIR, YOU ARE BEING HUNTED
BIG ROBOT

DAVID DAW: *Obviously your name's known around the gaming press, Jim, but I think the Kickstarter is probably the first time a lot of people knew you as a game designer. How do you go about building an audience with a Kickstarter campaign if you aren't a big name in that space?*

JIM ROSSIGNOL: We definitely didn't have to work as hard as some other people. The fact that both James and I have lots of contacts in the U.K. press helped. I had some profile from *Rock Paper Shotgun*. Although this is our first game with any significant profile, I think we as individuals probably had a bit more profile than some people new to Kickstarter. There are certainly people who have struggled while being really amazingly talented. I was talking to the guys from *Moonbot* who are doing *THE GOLEM*. *Moonbot* is a big animation studio—40 or 50 people in the animation world. They're renowned. They have a couple of guys that are famous animators, a famous children's illustrator, and in the game world nobody knows who they are. There's no excitement about *THE GOLEM* at all. The project looks incredible, the artwork's fantastic, and the idea is brilliant. It's an amazing proposition, and yet in the gaming world, these guys have no profile at all. They're only big in their own field.



I think just the fact that we have some traction in the game sphere did help us and I think it's helping others, if they come from the mod community, or if they have some kind of profile in indie games. There are other people that are trying to cash in on nostalgia and not getting anywhere.

JAMES CAREY: I think that's sort of shown a bit with the way The Frontier sort of ran the initial part of their campaign—they had people going "Where's all the content that all these other Kickstarters are putting up?" I think it was helpful to have a plan for all kinds of videos, and have a plan from the outset for what we wanted to do, and when we wanted to be able to show certain stuff, and we wouldn't run our Kickstarter until we had that. We were very clear that we wanted to have something to show people before we went on Kickstarter.

ROSSIGNOL: Even then, we felt a bit like we didn't have as much content as some of these more established guys that were able to reel off videos and reel off art. We don't have an in-house artist. We're just not big enough for that. So we weren't able to roll out some of the stuff that these established studios were able to. Those guys have the resources to convince people, and to promise more than we could promise when we're just a couple of guys, which I think makes a big difference to how the pitch is perceived and how convinced people are at a glance.

CAREY: We were determined to show a level of professional polish. We said we're going to try and not show any kind of broken bits or any kind of placeholder UI or broken geometry or things like that. So people seeing it actually get a real idea of what the game's going to be, and what they're going to get, what we're aiming to deliver.

ROSSIGNOL: You're starting to get a little bit of a backlash on Kickstarter now. Starting to get some high-profile names not getting the resources they expect and stuff like that. I think the way we approached it may end up being the way that Kickstarter has to work, even if you're someone with a name, which is that, "Look, we really are a long way into this. We have committed to it. Look at all the stuff we've done." I mean, we've put [in] a large amount of our own money and a huge amount of our time. It was six months before we went for a Kickstarter, so we were fully entrenched at that point, really going for it.

Perhaps that's not possible for larger studios with bigger overheads, but for us, it was everything. I mean we were already committed to it, so I think there's a difference between guys who are going to make it work somehow, and these larger studios that have failed to do anything by saying, "This project just isn't going to work unless we get Kickstarter money," which is perhaps a shame in some cases.

DAW: *How do you scope your game for Kickstarter to make sure your pitch seems exciting without overpromising on what you can deliver with the game?*

ROSSIGNOL: The biggest factor for us was that we basically put our own money into it, so we said, "We have this much, let's get as much done as we can on the basis of that budget and see where we are." I mean, we hadn't done a game like this before, though we've spent a few years exploring related technologies. This was the big one for us, our big, exploratory, experimental thing.

We just wanted to see what we could make with the money we've got, and that was the sort of deciding factor for when we would go for Kickstarter. I think we realized pretty early in that process that Kickstarter would be the best prospect. It was really coming into its own at the time, plus it gave us full ownership, so all the reasons people usually go for Kickstarter appealed to us as well. But I think the other thing we discussed quite early on was: Given that we had some money and time, the best way to make the pitch really strong was to work out what the game was going to be, and get as far toward that as we could.

So nothing that was promised was out of bounds for what we wanted to get done, no matter how much it would take to achieve it. I think we perhaps benefited a bit from how ambitious we were anyway—it was just quite an ambitious idea for a game, using some

technology that gets a lot of profile, but that you don't necessarily see being used the way that we're using it.

DAW: *How do stretch goals change your scope?*

CAREY: Stretch goals are really weird. You're constantly in development thinking about whether something is viable or not, and the kind of stuff that you need to do, or if something that you can do isn't working, or if it's working so well you want to have more things like it. So you're constantly testing what you think the balance of the game needs to be.

Stretch goals kind of fall into that for me. They're kind of stuff that you have on your "nice to have" list but that aren't really essential.

ROSSIGNOL: I had a conversation with Andy Schatz (MONACO) where he was saying, "What a weird thing to do to be promising things that should be optional." We knew what the core game was going to work, and we knew how it could be extended.

We're at an interesting stage now, where we've built on the prototype stage we had, and smoothed out the major issues, and now it works essentially as a complete game with just one NPC, just the hunters, the original concept art of the robotic gentlemen. That now works as a complete game. I think, perhaps, if we hadn't had the Kickstarter money, we'd have gotten to that stage anyway, but that might have been the game we put out, because it's playable just with this one NPC that we've given the behaviors that we want, who hunts you down and allows us to play at combat. All of that exciting stuff that we originally sat down and conceptualized and said, "This is how that should work," that's all in the game and it works.

But we can extend that, and we knew how we could extend it, and how NPC behavior could extend it, and the challenge now is to add all those things. None of the stretch goal things were unreasonable. It was all within the scope of a design doc that we could extend, because we knew how it would all work.

I think actually the most challenging thing for us, initially, was adding multiplayer to it. We didn't just want to do a single-player game, but we knew the core of the game had to be this single-player, being-hunted experience, and I think that putting multiplayer up was a gamble. But it's also the one that we feel the most pleased to have ticked off the list. Because of our existing interests, it felt like the biggest gamble, and people were saying to me and to James that multiplayer is a big task. At the same time, that was the thing that...

CAREY: It's the stretch goal, right? It's the thing to stretch toward.

ROSSIGNOL: It was the thing to stretch toward, but I also felt it was the most natural thing, because we were already clearly going to end up having a multiplayer game at some point. So, to kind of get on with it and make that, is good. I think the other thing is, and this is why Kickstarter is brilliant for a lot of people and not just us, is that a lot of game development is a kind of scientific exploration process of trying a lot of different things and seeing how they can be thrown together in this mix to make interesting experiences.

For us to be able to go on and do that and experiment with the multiplayer stuff in a game we've put so much time into already, that's great.

DAW: *Do you think Kickstarter can be widely used as a way to break into the game industry or do you think your case was relatively unique?*

CAREY: I think there's a lot of danger in assuming that you'll put this up, and the money will come, and you'll get going, especially if you're completely new to it. But I think that's going to be more damaging to the individual than it's going to be for Kickstarter itself; Kickstarter's going to be around for a while. If somebody says, "We're going to do this!" and then they can't do it, and then asks for something else in the future, the chances are that they aren't going to get it.

DAW: *How did you organize your PR for the Kickstarter campaign?*

CAREY: There's a touch of controversy there because there was the question of how fair it was for me to cover my own game on

my own independent website. That was something that we all discussed within Rock Paper Shotgun. We ended up feeling that honesty was the only way to go with this stuff.

If I was open and honest about everything, and I want to make games, and I am making games and enjoy doing so, to not discuss it on the website that I own just seems bizarrely self-defeating, and as dishonest as not talking about it. It felt to me like I'm the owner of Big Robot and the owner of Rock Paper Shotgun, and I've got to just be open on both of those, and the people who read RPS are going to want to know what I'm doing with Big Robot.


I did want to heavily disclaim that and say, "Of course I can't be objective about this because this is something I'm making myself. This is something that comes from what I'm interested in." But our audience seemed to totally accept that. If the policy is honesty, and you stick with that then it works just fine.

I suppose the consequence of all that is that everyone is reading everyone else. So when that went up, we were able to get a whole lot of other people talking about it. I think what's really interesting about that, though, and it's probably something that a lot of people around the gaming press don't really realize, is that a huge number of gamers, or a huge number of people who don't read the specialist press at all, have essentially no feel for games stuff. They buy essentially what they see advertised or what their friends recommend to them. So what was fascinating about coming to Kickstarter is that I think the majority of people who backed it knew nothing about my RPS stuff, had no idea who I was, and no idea who James was.

In fact, I spoke to Ken Levine a few weeks ago and I said, "Aside from what we're talking about now, I really want you to watch this video of this game called SIR, YOU ARE BEING HUNTED." And he said "Wow, I'd back that," but he hadn't connected it to me at all, he just saw it on Kickstarter and thought "Wow, that's something I want to happen." I think lots of people on Kickstarter have done that. I think one of the reasons that Kickstarter is so powerful is that its audience isn't that audience of game sites that all read each other, and all reblog each other's stuff, and all read the same press releases, and essentially have the same audience.

Being able to have people come in and be completely unfamiliar with us, who haven't seen or heard anything of the game before, seeing those people on Kickstarter excited about it, has been an awesome thing for us to go through. So, although we were able to leverage Rock Paper Shotgun and the connections within the gaming community, generally I think actually just going on Kickstarter and the leverage of having that huge Kickstarter audience browsing the site has been just as important if not more important.

CAREY: I think we hit the wave with Kickstarter as well, that we came on right at the crest of that interest in it, and perhaps a bit before a bit of the backlash. We got in just before Christmas, and that seemed to be a good time to hit Kickstarter.

ROSSIGNOL: There does seem to be some exhaustion with it now. People have backed a lot of stuff now and not gotten the games yet. 

David Daw (@davidhdaw) went to NYU and studied comic books and zombie movies. He's written about games, TV, and technology for publications like PC World, Io9, and Game Developer.



MONEY TALK

SPEAKING THE LANGUAGE OF MONETIZATION DESIGN

Automobiles and computers were so simplistic in their first 10 years that today we have a hard time looking back and appreciating just what a leap in technology they were at the time. Like all technology, they benefited from the iterative process, slowly adapting to changes in allied technologies, consumer demands, and infrastructure. Today both cars and computers have components in them that did not even have names 10 or 20 years ago. Before those components could be added to these products, they had to be thought about and given names so that they could then be optimized and adapted to various uses.

Today the technology of monetization design is literally in its first years of creation. The concepts can be quite complex, but in talking about them we use broad terms like free-to-play, microtransactions, and pay-to-win to describe concepts that would be difficult to explain in detail. The upside of this is that it gives others a general idea of what you mean without a long academic discussion. If a developer says, "I am going to replace the subscription on my game with microtransactions," we all nod and pretend to understand what the developer means. We can also pretend the developer understands what they're saying. The downside to this simplification is that often no one in the conversation understands what is going on at a level that can be applied to product.

In this article, I am going to attempt for the first time to put a majority of the monetization design language I have been developing over the last four years in one short article. All of these terms have been used in my other papers on gameful.org and here in *Game Developer* and *Gamasutra*, but never in one place. I cannot understate the importance of a unified language for this space. Its importance goes beyond intellectual discussion. The whole "Subscription vs. Microtransaction" paradigm is so limiting that it is crippling our industry. There are not two ways to monetize a game. There are millions, limited only by our imagination and our ability to articulate our ideas to our colleagues.

WHAT IS A "SUBSCRIPTION"? Generally, a subscription is defined as a recurring charge for service. In games, it usually means you pay one fee monthly and get the entire game 24 hours a day, seven days a week. For clarity's sake, I describe this as the Unlimited Use Subscription Model. The weaknesses of this model are profound, and go beyond the scope of this article. Think of it this way: Potentially any part of your game can be charged for in a recurring fashion, for any duration you set, and you can have as many modules of your game as you like charged for in a

recurring fashion. Instead of forcing your players into an all-or-nothing unlimited-use subscription model, you might find that a more flexible pricing system built around gating your content around smaller subscriptions ("microsubscriptions") will be an easier sell to a wider consumer audience.

Microsubscriptions can be used to gate content access, boosts, time, or any other feature you can think of. There are two primary advantages to offering multiple microsubscriptions in your product:



- ◆ This allows consumers to tailor your product to their needs. You cannot (and should not want to try to) predict how each consumer will use your game. Make it flexible enough to appeal to the widest possible audience.
- ◆ This allows you to continue charging for your game service, and thus maintains your revenue stream beyond the first month.

The end result is you capture the biggest possible slice of each consumer's available gaming budget.

WHAT IS A "MICROTRANSACTION"? A microtransaction is the purchase of one "piece" of your game for a set price. It is a one-time transaction that in some cases can be repeated. Used by itself, the term is so broad that it becomes a liability in any discussion, because what you are selling is much more important than the fact that

it was purchased piecemeal. So here I will discuss the what of nonsubscription content purchases and attempt to wean us off the term "microtransaction."

Virtual Goods Sales: Nexon began playing with microtransacted virtual goods sales when I was working for them as a designer in 2001. There are two dynamics that are important here. Do the items provide a competitive advantage? If so, I call them "Supremacy Goods" (see **Reference 1**). Supremacy goods can have profoundly negative effects on your monetization efficacy. Non-supremacy goods generally take the form of cosmetic and "flavor" items that do not provide a clear competitive advantage in a game. There are gray areas that may sell well, like items that are balanced but give more tactical options and thus provide more "flexibility" to the user.

Virtual goods can, in some cases, be sold by entities other than the developer. In these cases the process is often described as a real money transfer (RMT) sale. (For a full discussion of these dynamics, see **Reference 2**). I break down RMT into three categories: **RMT3** consists of industrial-scale gold farmers and sellers, which were organized almost 10 years ago by a company called IGE that caused considerable harm to our industry. **RMT1** consists of the now-standard set of developer-designed options that we usually now call "microtransactions." **RMT2** is what came before both of these, what I call "expert trades" between active participants in a virtual world. It was essentially wiped out by both **RMT3** and **RMT1**. Blizzard recently attempted to bring back **RMT2** and monetize it using a real money auction house in *DIABLO 3* without understanding the involved potential pitfalls (see **Reference 3**).

If you sell an item that can normally be earned in gameplay, then you are selling game objectives, which can break your game (see **Reference 4**). This will have negative effects on your revenue generation.

DIABLO 3.

Content Sales: If there is some part of your game that is inaccessible until you pay to unlock it, then I consider this a content sale. DLC sales are exactly the same thing, though usually that term is associated with single-player games, not online multiplayer games. Note that while players can compete against each other in single-player games via leaderboards (a very useful method of motivating your customers), this also reduces the Supremacy Goods effect of selling content that gives players a gameplay advantage.

Please note that for something to be a content sale and not a time gate (see below) it has to allow access to some part of your game that is normally completely inaccessible without payment.

Time Controls: Time in-game can be rationed via time gates. A time gate prevents one player from getting ahead of another player just because they have more time to play. Turn-based games like chess and checkers do this automatically. As soon as you remove all time gates, players will be motivated to win your game by not sleeping (or eating, or working, etc.). This has substantial negative effects on revenue generation, though a full discussion of why this happens is beyond the scope of this article.

Note that a time gate need not strictly use a time metric—it can also ration a player's actions. Zynga does this in all of their products via the energy resource, for example. This is a preferable approach to using time itself as the control, as players will get aggravated if real life draws their attention away from the game and they return to find their time has expired. The greatest weakness of the Zynga time gate is that it can be bypassed completely with money, thus breaking it.

A game without a time control is a bit like an "all you can eat" restaurant. This works for some restaurants because they can just keep putting out cheap content forever, but for game developers, very little in the way of attractive content is "cheap."

Once a player has had their fill of your content, they will move on, as there is no way you can create content as fast as your gamers can consume it. Uncontrolled time also acts as a Supremacy Good, with the added disadvantage that you are not being paid for it. This will make your game much less attractive to those players with huge money budgets and small time budgets (your ideal customer).

DEFINING PAY-TO-WIN When you sell game advantage via any of the above methods, you're intentionally breaking the game. (I would go so far as to say your product is no longer a game, but just an entertainment product.) Further, when you make the game highly competitive between players, you create what is in reality an ante game (see **Reference 5**). An ante game is one where you can win just by raising the ante to the point where your competitors cannot match you. Skill and effort become irrelevant. Most Facebook ante games have no cap at all. Some games, like EA's browser-based iteration of its COMMAND & CONQUER franchise, have such high caps that the game still becomes an ante game. This has negative effects on revenue generation as further detailed in my Supremacy Goods microeconomic model.

SUPERIOR MONETIZATION While your "hardcore" gamers (I define these as playing more than two hours per day

on average, and typically eight or more hours on at least one day per week) may be your most vocal, they are not necessarily your biggest spenders. People with a job, high income, and very little free time will spend a premium for quality entertainment on those times when they do have time to play. Creating a game that provides a quality experience for these high-budget but relatively casual players, but without resorting to selling Supremacy Goods, should be the ultimate goal of a well-crafted monetization design. Obviously, exceptions will occur in the case of niche products, but those niches can get crowded very quickly, leading to rapid loss of positive net income.

Just as there are almost infinite ways to craft said design, there are also just as many ways to really foul up your design. The key is to know your consumers, and how any change you make to your design will affect the relationship between you, them, and your product. 🎮

Ramin Shokrizade is a lifelong gamer, having started as a competitive chess player at age five, then moving to play-by-(e)mail, tabletop role playing, and computer games as they became available. He has focused on creating advanced monetization models for online games and explaining the fundamentals of applied virtual economics in the hope that his work will lead to healthier, more profitable, and more sustainable virtual economies and business models worldwide.

References

- 1 Next Generation Monetization: Supremacy Goods http://gamasutra.com/view/feature/177190/next_generation_monetization_.php
- 2 Real Money Transfer Classification <http://gameful.org/group/games-for-change/forum/topics/real-money-transfer-classification>
- 3 Smedley's Dream: Predictions of the Diablo 3 RMAH <http://gameful.org/group/games-for-change/forum/topics/smedley-s-dream-part-1-2-predictions-of-the-diablo-3-rmah>
- 4 Game Monetization Defined <http://gameful.org/group/games-for-change/forum/topics/game-monetization-defined>
- 5 How Pay-to-Win Works <http://gameful.org/group/games-for-change/forum/topics/how-pay-to-win-works>

Scotland. Famous for golf and innovative games development.

All the best players
come here.



We've got quite a reputation for invention, innovation and discovery. And it stretches way beyond Highland Games, the bicycle and golf. We were the first to award a degree in Computer Games Technology and our pioneering games work ranges from the creation of Grand Theft Auto to Bloons and Quarrel. The fact is, Scotland is one of Europe's top games development locations. We have a growing hive of creative and talented games developers and our universities are

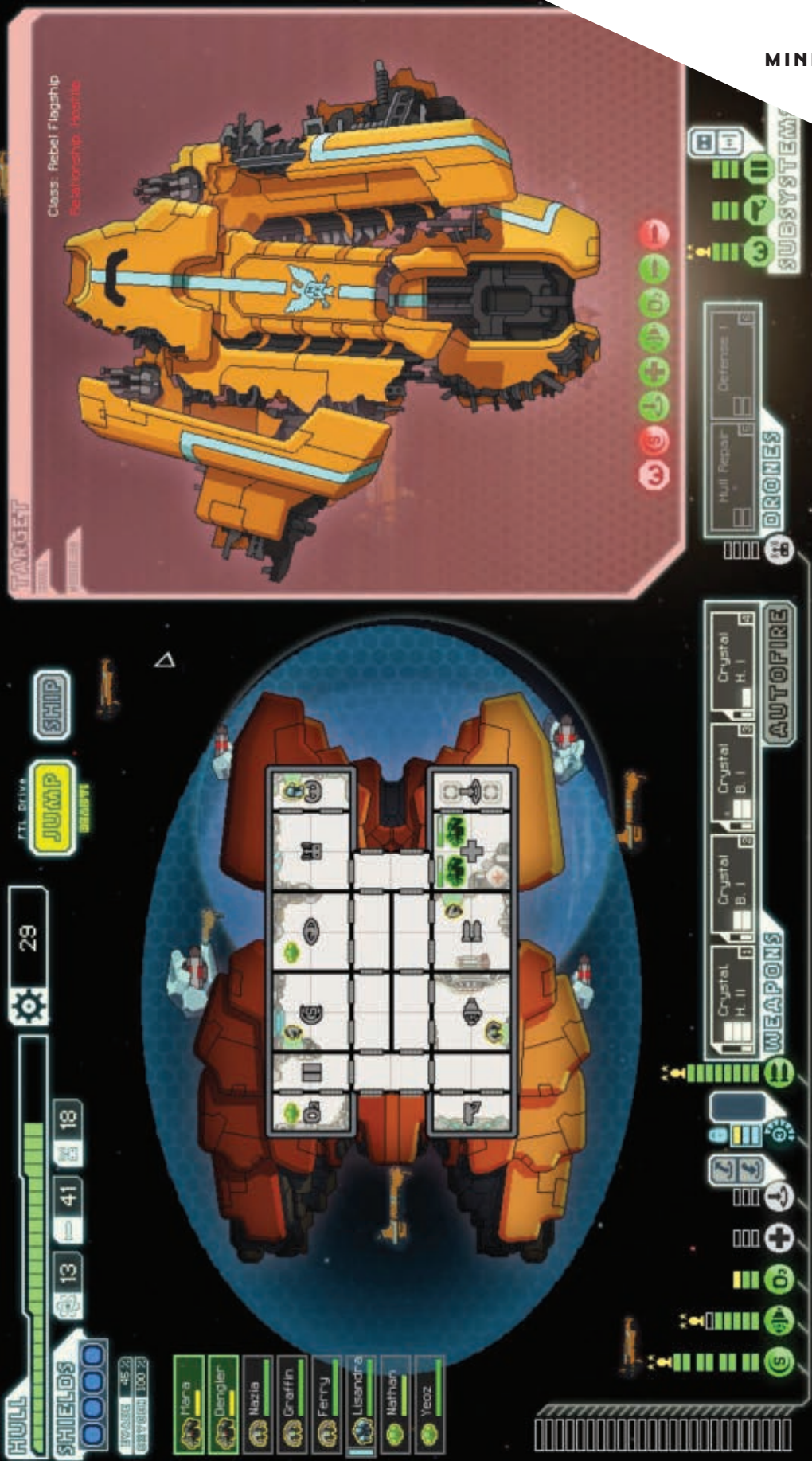
developing new and converging technologies across a range of platforms.

Above all, our people are dedicated, committed and passionate for success. And this passion, combined with our world-class academic institutions, outstanding research and superb facilities make Scotland financially irresistible. We can develop your products and help shape your business. And that's what makes Scotland such a popular place to live, work and play.

To see what we can do for your business, visit www.sdi.co.uk/games

SCOTLAND. SUCCESS LIKES IT HERE.

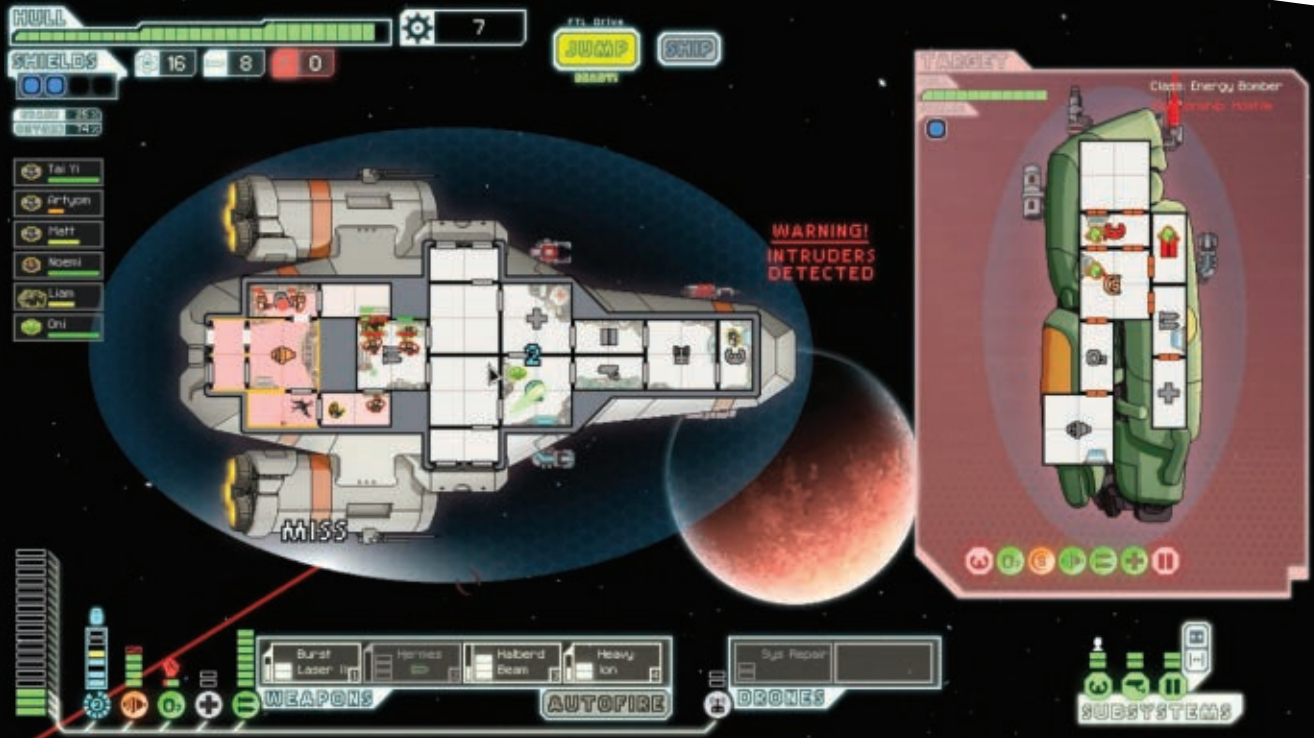




mm mini-mortem roundup postmortems from across the platform spectrum

If there is one thing we've learned over the last year at *Game Developer*, it's that dev studios need to stay current on every potential game platform out there, or risk missing opportunities to reach the widest-possible audience. That's why we've put together a collection of four shorter postmortems, each for a game developed for a different platform: Muteki's DRAGON FANTASY (mobile), Subset Games's FASTER THAN LIGHT (PC), KIXEYE's WAR COMMANDER (social), and RSLSB's DYAD (console). So whether you're a single-platform dev wondering if the grass really is greener, or you just want to learn more about what went right and wrong with a handful of standout games from last year, read on for the mini-mortems. >>>





ftl: faster than light pc

By Justin Ma and Matthew Davis

FASTER THAN LIGHT (FTL) was a hobby project that tumbled into success. Here are our highs and lows from the development process—and the Kickstarter campaign.

what went right

UNIQUE, NOVEL DESIGN FTL started as a pretty basic concept. We felt that space games often focused on the act of piloting a ship or managing a fleet, but never let you feel like the captain. We wanted to experience making high-level decisions about the ship's strategy as well as managing every action of the crew. In our search for an enjoyable experience we created a unique type of gameplay: part simulation/strategy, part "choose your own adventure," and part RPG.

We found that for many people, the strength of the basic gameplay (with the help of amazing music from Ben Prunty) was enough to overlook what they might have considered flaws in the game: the simplistic graphics, its repetitive nature, and the sometimes frustratingly brutal difficulty. It effectively satisfied the little fantasy that we've all had when watching great science fiction movies and TV shows.

Initially we thought that there would be no potential market for a game this unforgiving, and we were surprised by how much other people liked it. In the end, we believe that the desire to create a game that we ourselves wanted to play allowed us to come up with something that appealed to others. This sentiment has been stated by developers for ages, but we think it played an especially large role in FTL's success.

AMAZING TIMING (AND LUCK) In early 2011 we had both quit our jobs to spend a year making small game prototypes. After starting FTL, we used the IGF China 2011 submission deadline as a concrete milestone for our development; we decided that if we couldn't get a solid game prototype by that time, we'd move on to another idea. We then spent four months working on game mechanics, rather than a game, and we were frustrated and unsure what the game would be until something clicked during the last two weeks. We determined the game's structure and pacing almost overnight, and were able to submit our first playable prototype to IGF, where it was well received. Indie game competition deadlines continued to line up nicely as periodic development milestones, but that

was just the start of our fortuitous timing.

By early 2012, we were running out of funds and started to look into crowdfunding options. This was before Kickstarter had been used to raise millions of dollars for game projects, and at the time we thought it would be a good way to reach out to some people and raise a few thousand dollars. We had gained some publicity with honorable mentions in the 2012 IGF, which got us a public demo on OnLive's service during GDC, so we were already planning to line up our Kickstarter with those events—but our luckiest break actually came when Double Fine launched their Kickstarter campaign, which was two weeks before we were ready. Thanks to Double Fine, our Kickstarter got much more traffic.

So in just one week, the FTL Kickstarter benefited from the "Double Fine effect," two IGF honorable mentions, and a demo available on OnLive and the GDC show floor. This perfect storm of publicity undoubtedly led to our Kickstarter's overwhelming success.

KICKSTARTER: POSITIVES Money changes everything. The Kickstarter's success changed FTL from a hobby project to a business overnight. We became a "studio" with a decent number of fans keenly awaiting the release of our game.

This was one of the biggest moments of success in FTL's development, but also the primary source of stress and issues.

Our savings were all but used up one year into development. If we planned on releasing the game as a commercial product, we'd have to cover everything from food and rent to licenses and a lawyer, so we launched a Kickstarter campaign with the modest goal of \$10,000, expecting that we could barely reach that amount, or, if very lucky, perhaps achieve \$15K to \$18K. But due to the fortunate timing of events described above, we ended up with just over \$200,000 in funding.

The Kickstarter had a number of positive benefits on our development: First and foremost, we no longer had to worry about paying rent. We were also able to expand FTL's scope; Ben expanded our initially slim music plan into a full soundtrack, and we enlisted writer Tom Jubert to expand FTL's universe and lore, meaning more ships, aliens, and weapons.

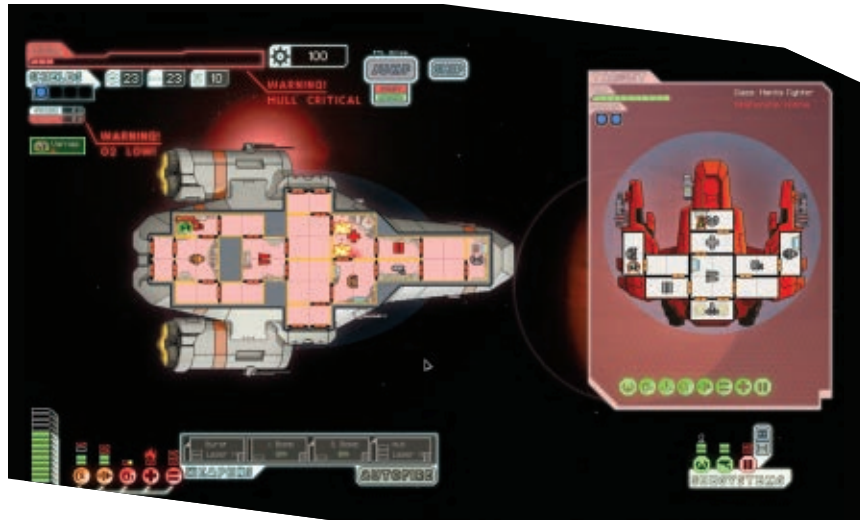
Our campaign also made us part of an expanding Kickstarter movement, and when we released, we became one of "the first" from that new wave of Kickstarter successes—both of which led to a lot of press attention. It even got the attention of Valve, allowing us to both distribute the game and host our beta on Steam. This private beta was possibly the most important part of the Kickstarter; we got almost 3,000 beta testers, and FTL became a far more polished and stable game for it.

what went wrong

KICKSTARTER: NEGATIVES With \$200,000 in funding and nearly 10,000 new fans, public expectations are bound to change. Many people felt that with the extra financing, the game should be bigger and better than previously planned, but we had set a release date that was only five months from the end of the Kickstarter. Nearly every way to expand the project (hiring more help, licensing better technology, and so on) also required more time to get it done, so we had a hard time balancing the scope and time expectations. The game was greatly expanded and released only two weeks later than expected, but it was far from a smooth experience.

The extra fans and publicity also meant a lot more public relations work, and we didn't have a PR firm or marketing manager. We set up a forum to start building a community space for our new fans, which involved additional technical challenges we've never experienced. In retrospect, we should have had someone manage public relations for us so that we could focus on development.

In addition to contracting out help and building a community, we were setting up a company, finding a lawyer, discussing contracts with distributors, and much more. Every day we had to learn how to do



things we've never had to do before. We were wholly unprepared for all of it, and relied on our extremely generous friends and family. We set out to make a game and didn't realize we had to learn how to make a business, too.

QUANTITY OF EVENTS/LIMITED DEVELOPMENT TIME Since FTL started out as a small experimental project, the original vision was quite limited. We thought we would have a dozen or so basic event types with a dozen different flavor texts for each type. We wanted to create something similar to a deck of event cards in a board game. Our (perhaps naive) plan was to simply add more text if we needed more variety. How hard could that be?

After the Kickstarter, we decided to expand the game universe by adding a number of alien races, which meant more locations and events. At that point, we had something in the realm of 10,000 words worth of events. By the end of the project, FTL had nearly 20,000 words. That's a lot of text, but we discovered that even this amount would not be enough. When you divide the events between the sectors, even 20,000 words start to spread pretty thin.

One of the most common issues reviewers and players have with FTL is repetition of events. Even with a writer working for six months prior to release, we couldn't create the variety we wanted. While text is perhaps easier to create and integrate into a game than unique animations and art, it's hard to pump out the sheer volume needed to keep it fresh for hundreds of replays. Perhaps our time would have been better spent finding ways to make common events more compelling rather than adding tons of unique events that lose their impact after the first encounter.

MULTI-OS LAUNCH Cross-platform development is generally a great plan; we

wanted to support alternative operating systems and expand our user base to reach as many interested players as possible. But attempting a simultaneous, three-platform release for your very first game project is not a great plan.

We thought we were prepared—FTL was planned from day one to be a cross-platform game, so all of the libraries and code would (mostly) easily transfer to other systems, and there was even a development version of FTL for Linux as early as five months into the 18 months of development. We believed, foolishly, that it would be an easy task to finish off the OSX and Linux builds once the Windows build was ready to go.

The crunch of release involved the typical 12- to 15-hour workdays to just finish the Windows version, much less the last-minute cross-platform work. Even once we thought we'd succeeded, four days before the launch we discovered the Linux build wouldn't run on a substantial number of the different Linux flavors. Fixing the problems involved intense all-nighters and frantic phone calls to Linux-porting experts. We ended up launching smoothly, but it was hard earned.

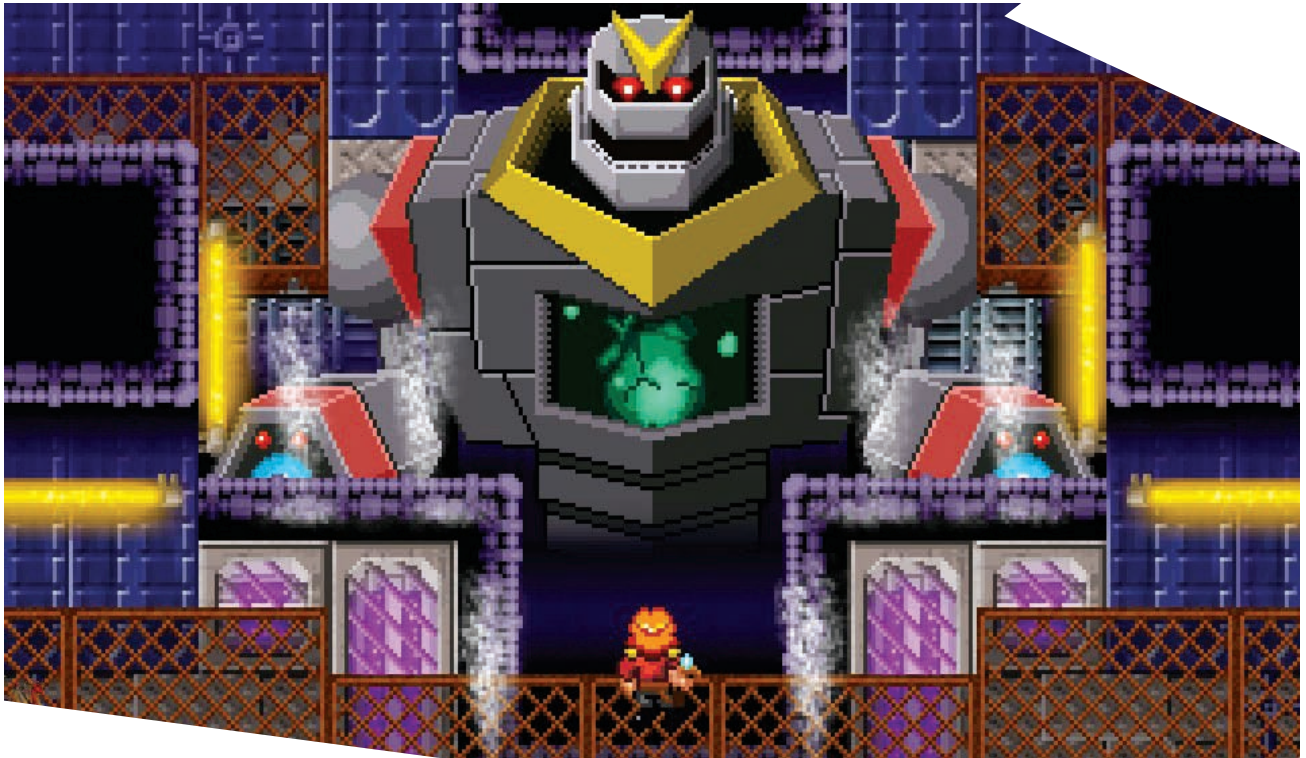
Supporting extra OSes causes problems during the immediate post-release support. We released two patches within a month of release to solve system-specific issues, and each one required making and testing four builds (Windows, OSX, and both 32-bit and 64-bit Linux) before we could release it. As new developers, our build pipeline wasn't ideal, and testing four builds with unique quirks with just two people doesn't speed up the process!

Simply delaying the Linux/OSX releases until two to four weeks after the Windows release would have made things more manageable. Stumbling into it clueless from day one was not ideal. **ttt**

Justin Ma and Matthew Davis are the founders of Subset Games.

dragon fantasy mobile

By Adam Rippon and Bryan Sawler



We started on DRAGON FANTASY on April 1, 2011 as a tribute to Adam's late father, Tom. Adam started making the game as a way to cope with the depression and stress in his life. While it probably wasn't particularly healthy to be as obsessed as he was with one project, he sure did get a lot of work done in a surprisingly short amount of time! The first chapter of DRAGON FANTASY launched on iOS on August 23, 2011.

what went right

REGULAR CONTENT UPDATES The game was a modest success, and we immediately set to work on adding more content to it, hoping that by continually adding new content we could keep sales consistent.

While we weren't hugely financially successful from all of our free content updates, the goodwill and reputation that it earned us was a huge benefit. We've made a lot of friends in the indie developer community, which has been a huge help. We learned a lot about how to market our game via shows and via the press. Also, we bumped into Sony several times during the development of the game, and I believe that it was our dedication and cult-favorite status that led them to decide to include DRAGON

FANTASY BOOK II in the Pub Fund. Had we put out chapter one and called it a day, I wouldn't be writing this article right now!

GREAT PRESS COVERAGE If there's one thing you absolutely need to have on your side, it's great reviews—and we got lots of 'em. We enjoy a 4.5 star rating on both iOS and Android, despite the perpetually entitled rage of the "OMG WHY ISN'T IT FREE" crowd. We got great coverage from RPGamer, whose editor-in-chief absolutely loves the game. Joystiq gave us some great shout-outs. And our crowning achievement was our interview with Kotaku Australia—Adam has a copy of it printed and hung up on his wall, and his mom even mailed a copy of it to his grandma. [It was that good.] Apparently it wasn't that common for Kotaku U.S. to run Kotaku Australia's articles, but they ran this one. Oh, and the sales bump from that beautiful article? Very, very nice. Great press goes a long way.

GOOD TECH HELPS DRAGON FANTASY may not look like it's a super high-end engine, what with all the ginormous pixels and whatnot, but you'd be surprised! We've always rolled our own engine and tools, and the work on DRAGON FANTASY was a serious boon to the production of our very powerful

and very easy-to-use UI system. While we didn't make a ton of money on the game itself, we did make a fair bit by using the tech we built for the game on other contract projects. We've done numerous paid projects for larger clients using our MuTech engine, even going so far as to use it in a political news app! And despite being reviewed by dozens of blogs, not a single one noticed that it wasn't a native iPhone app. We're pretty proud of that. So while we probably could have just done DRAGON FANTASY with some off-the-shelf engine, there are some serious benefits to building your own cross-platform, application-agnostic engine if you have the means.

what went wrong

WE LAUNCHED ON MOBILE With each update we realized that it was getting harder and harder to reach more users interested in an 8-bit RPG on mobile. That's not to say they didn't exist, but it's very hard to inform gamers with more hardcore tastes about mobile games. That, and you can only rely on mobile game sites to cover your updates so many times. We might have potentially had more financial success by doing paid updates, but it would diminish quickly without a larger base. Lesson

one: Finding an audience for a \$3 game on mobile is as hard as they say it is, even when your audience is more hardcore than the average mobile consumer.

And, speaking of launching: Our biggest problem, and one that still haunts us with DRAGON FANTASY 1 today, is the stigma of mobile games. Our game didn't feel like a crappy mobile RPG, but because it was made by a small team on mobile we had a tremendous amount of difficulty getting anyone to pay attention to us outside of mobile. Our Steam submission(s) took months before they were rejected, and one of the other, more indie-friendly stores outright told us they weren't interested in mobile ports. (I was incredibly punchy that day, let me tell you.) Lesson two: If you're making a core game, launch on other platforms first to avoid being called "a mobile port."

EASY PORTING LED TO UNDERTESTING

Since we loved working on DRAGON FANTASY and really didn't want to see it end in obscurity on mobile, we started porting it to other platforms. The tech we had built up let us port the game quickly from iOS to both Mac and Windows. Unfortunately, the ease with which we ported the game led to us being overly confident about the state of those ports.

Being primarily a Mac shop meant that the Mac port was pretty heavily tested and has enjoyed a fairly stable existence. On the PC side, our testing simply wasn't sufficient and... Well, it just wasn't a great product at launch. The Windows game needed a lot of things that weren't necessary on other platforms—things like an installer, runtimes, and checks to make sure appropriate versions of DirectX were installed. Add to that the different versions of Windows, ranging from XP ("just put things wherever, it's fine") up to Windows 7 ("sorry, you can't put those files there!") and we come to lesson three: Just because it works on everything else, doesn't mean you can cut corners on testing.




WE WERE TOO AUTHENTIC When we first started talking to the press about the game, we took pride in how authentic we kept everything. We stuck strictly to the range of colors the original NES was capable of putting out, and even limited our artwork to the number of colors-per-tile. The problem is, while as developers we appreciate that, and even a lot of the press we spoke to thought it was great, the masses didn't agree. We learned that what 8-bit games looked like, and what people remember 8-bit games looking like are two very different things. Some of our maps could stand their ground against the very best-looking NES titles just fine, but still we saw cries about how bad the game looked.

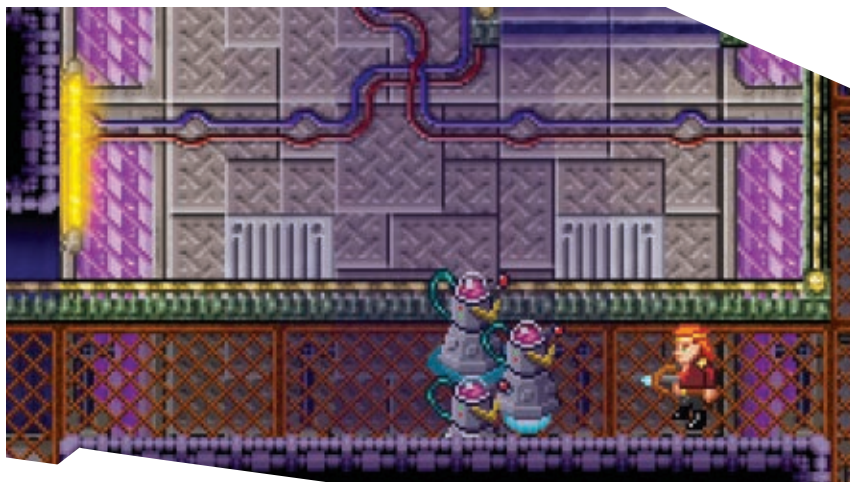
Internally, we brushed these off as people who just didn't "get it" or weren't there in the mid-1980s to play the games we were inspired by. Our choosing to ignore this feedback just meant that we had turned more potential gamers away before we ever had a shot. Looking back over the feedback we received is part of the reason

that when preparing our big "relaunch" of the game on PS3/PS Vita (and updates for existing platforms) we went back and updated all of the artwork to much better line up with what people want. There's nothing "not indie" about keeping your vision, but presenting it in a way to get the biggest possible audience!

FINAL WARRIOR QUEST If someone were to ask us three or more years ago what platform to develop their indie game for, it's almost guaranteed that I'd have suggested they target iOS. Now we have to face the irony that the platform largely lauded for giving small developers a chance with a cheap entry cost and removing the need for a publisher has so much competition that the best way to be really noticed is by getting a publisher behind your title who can devote a large amount of money to the launch. Finding an audience for your game on any platform is a challenge, but one as smothered by new releases as the iTunes App Store? Unless you're well-known or incredibly lucky, don't expect to get a lot of traction.

And so, coming out of DRAGON FANTASY we're not abandoning iOS as a platform altogether, but it's far removed from being the first platform we look to when planning our next games. The PC has made a huge comeback, and even the consoles are opening up a lot more to indies (we can speak firsthand about Sony, and from what we've heard Nintendo is a lot easier to work with as well these days). There are a lot of places to put your game. Pick one you can truly make it shine on and go for it. 

Adam Rippon is the creative director for Muteki, writer and lead programmer for Dragon Fantasy, and a lover of video games, art, and shanks of meat. Bryan Sawler is the president of Muteki, juggling the business, engineering, and platform-specific code with effortless style and grace.



war commander social

By David Scott

I've always been a huge fan of real-time strategy (RTS) games: It started with DUNE II, continued into TOTAL ANNIHILATION, and peaked (for me) with WARCRAFT III and COMMAND & CONQUER: GENERALS. I played those games to death, and I met several good friends through those games—including KIXEYE co-founder Paul Preece.

Inspired by the mods we played in WARCRAFT, in 2007 Paul and I had developed a few tower defense games in Flash that were successful enough for us to quit our jobs and make games full-time. Fast-forward three years, and we were working on a version of desktop TD for Facebook (DESKTOP DEFENDER) and I was looking for some browser-based RTS games to play in my spare time.

The games I found were simple HTML affairs, you had slots you placed buildings in (the location of buildings didn't matter) and attacks were instant; all you saw was a battle report informing you of how many troops you lost and resources you looted. For someone whose favorite part of a RTS session was the epic battles toward the end of the game, this was a huge letdown.

We wanted to merge Tower Defense and RTS into one game; a game where the placement of your buildings and defenses mattered and, more importantly, where you got to watch and take part in the attack in real time.

In early 2010 I started work on BACKYARD MONSTERS, an MMORTS that built core RTS mechanics around a friendly and playful art direction (monsters building bases, instead of tanks and guns, in your backyard). As BACKYARD MONSTERS launched to rave reviews and our confidence (and player base) grew, we updated the graphics to be more realistic and edgy.

Emboldened by BACKYARD MONSTERS'S success, I set to work on creating WAR COMMANDER. It was to be an MMORTS game with nothing held back; we would have tanks, guns, airplanes, helicopters, suicide bombers, the works! In short, it was to be the game we were too afraid to make only 12 months earlier—and since then, it has become the most popular MMORTS on Facebook, with the average player spending 8.6 hours in-game per week. But when you go from "We should totally remake BACKYARD MONSTERS with tanks and

guns," to a live game in six months with a four-person team, you end up cutting some corners with the intent of re-adding them after launch. We hadn't anticipated getting one million installs in our first three months, and we weren't ready for how quickly players would progress through the game, and so we had to spend the first few months after launch rapidly adding new content.

what went right

KEEPING THE TEAM LEAN AND KEEN

We launched WAR COMMANDER in six months with three people; that's pretty efficient! It took a year to get the headcount into double digits, and today we're still pretty lean compared to other game teams. We start all our games with just one or two people designing the game and making prototypes to prove out the mechanics, and slowly add tech and others to the team as the game gets closer to launch. Keeping the team small keeps communication overhead down to a minimum and, as a result, early progress is extremely fast.

Another thing we do right at KIXEYE is fully embrace the studio model. Each



team is fully independent with the executive producer acting as the CEO of their own little company. What works for one team may not work for another, so we don't enforce certain methodologies cross-team. We do have shared learnings, but it's more organic, with people chatting over lunch to their counterparts in other teams, and weekly presentations to break down past releases and discuss upcoming features.

MAKING TRACKS! WAR COMMANDER is two years old and still has weekly updates with new content and features. We now have five tracks in development: The first track is for large features that may take over a month to develop (the world map or customizable units, for example), the second track is for smaller features (anything that requires one or two weeks of dev time), and tracks 3, 4, and 5 are dedicated to improving infrastructure, fixing bugs and exploits, and improving operational efficiencies.

All five tracks work independently of each other, so if one is delayed it doesn't impact the progress on the others. We move developers between the tracks to keep things fresh for them—it's good to have everyone on the team working on something that is player-facing now and then.

THE VERTICAL SLICE With the exception of the world map (explained later in this article), we made the right calls on what to cut to get the game out on time and on quality. We had to sacrifice a few buildings to launch with air units, cull the number of infantry units to add a wider variety of weapons, and cut music to have the units speak when you give them orders, but it was all worth it.

Making a vertical slice was key to this success. It forces you to develop each part of the game a little, instead of getting bogged down in one area. As an indie game developer, I know it's very easy to narrow your focus down to a very tiny and insignificant area of a game and just keep iterating on it. If there's only one thing you take away from this article, make it this: Build a vertical slice of your game ASAP, and play through it end-to-end before developing out any one aspect.

what went wrong

SKIMPING ON THE METAGAME One of the corners we cut to get the game out was the world map that you play in. In the early days of WAR COMMANDER, you were presented with a list of targets (AI- and player-controlled bases). You could scout and attack anyone on the list, and killing them would remove them from your list and replace them with a higher-level target. The system worked well, but it was a poor substitute for a real world map, which is a game in itself. We didn't get around to adding a map until

April 2012, eight months after launch. As soon as we released it we saw a 25% jump in engagement, retention, and monetization—and realized how important the world map metagame layer was (and how stupid we had been in not delivering it sooner).

NOT GOING WIDE ENOUGH, SOON

ENOUGH For too long we went down the path of adding more buildings and units to the game to satisfy the needs of the late-game players. Instead we should have added features that create sandboxes they can play in. Instead of offering them new things to build, we should have given the players new ways to tweak existing content to make the game unique to each player and his or her army/base. This changes the game from "I have everything built and at max level" to "I have everything at max level, but maybe this isn't the best combination of things; I need to experiment!". This attitude has the benefit of creating a wider variety in base designs and attack strategies, making it more varied for the attackers and defenders.

POOR TIME ESTIMATES We had a solid RTS game; you had your base, your army, and a world to dominate, but there was one bit missing: real-time battles. For many technical reasons we didn't allow attacking between two players that were online at the same time. We fixed this in October 2012 with the launch of Live

Battles, so now players who were online at the same time could take part in the same battle and interact with them in real time. We knew this was not going to be something that would drive monetization in a big way, but we did it anyway because it was cool and we were not happy putting our name to an RTS that didn't have synchronous PvP battles.

We knew it would be hard to retrofit synchronous multiplayer into a live game while continuing to develop and release new features, but we had no idea how long it would actually take. We had to rewrite a very large portion of the game in a completely different language (a language none of the existing dev team was familiar with) so it could run on both the client and server. This took us a painful, drawn-out six months or so to develop, test, and release. If we had done it before we launched, we might have been able to save ourselves a few months of work, and if we had taken the time to sit down and correctly break out all the tasks and estimate them, instead of just diving into production, we could have better planned our time. The one silver lining to come out of it all was a new process on the team that has helped us hit every deadline since the launch of Live Battles with a pretty high level of accuracy and minimal crunch. 

David Scott is a co-founder and executive producer at KIXEYE.



dyad console

By Shawn McGrath

Developing Dyad was a long and difficult process. It all started when I was playing a lot of Kenta Cho games; I liked RROOTAGE and PARSEC 47, but TORUS TROOPER wasn't doing it for me. So Pekko Koskinen and I dissected TORUS TROOPER, along with some other racing games, discovered a few design issues most racing games had in common, and decided to make a game that fixed those issues. We wanted to make a game that encouraged you to think tactically, instead of forcing you to rely solely on reflexes and muscle memory as the game got faster, or complicate the controls unnecessarily.

We figured we'd solve these problems and make DYAD in a year. We were wrong. Pekko eventually left the project, and I continued on for over three years trying to solve seemingly unsolvable problems. I worked 10-16 hours a day, six or seven days a week, for three years straight. It was worth it.

what went right

I'M TOO PICKY Before making DYAD, I heard stories of Miyamoto's ability to discard months of work if it wasn't working out, and thought it'd be easy to do the

same. It isn't. I threw away 90-95% of all the work I did on DYAD. I spent nine months on a mechanic that followed and replaced zip lines called "gates." Gates contained more variation and interesting gameplay than the rest of the game combined, but it was completely unteachable. I couldn't even teach my wife how they worked after several hours of one-on-one tutoring (and she's very good at DYAD), so I scrapped them. I created over 200 other levels, and I threw all of it away except for the very best stuff. As it is, the game is longer than I'd like, but I can't cut any more of it.

DOING (ALMOST) EVERYTHING MYSELF

In 2010 I showed the game to several publishers, and three showed serious interest. We negotiated getting funding and staffing up. I called it all off and decided to pay for it myself by living as cheaply as possible and draining my life savings. That way, I could make the game I wanted with as much time as I needed.

I did all the programming, game design, and graphics essentially by myself, except for some help with the PS3 version and a part-time co-designer for the first year of development. By programming my own engine, I was able to get the load times down to <1 second, and I was able to use low-level graphic effects and

experiment with a wide variety of new graphics techniques. I used one monitor for Photoshop, one for code, and one for the game, which let me change the graphics and the code and see the changes instantly. Without this, I wouldn't have been able to come up with the visuals in DYAD, and without the live code update, I wouldn't have been able to test out nearly as many game design ideas.

I also did all my own promotion and advertising. This started when I built a large motorized chair ("THE MACHINE") that tilts and rotates to match the player's in-game actions, and went to PAX East in 2011 with the disassembled chair packed into my Chevy Impala. Jason DeGroot and I spent two days assembling it before I showed DYAD for the first time to a large audience with THE MACHINE. I received a lot of positive press from PAX East, so I decided at that point to do all the marketing myself. Even though it was very time-consuming I wouldn't have done it any other way.

3. MUSIC COLLABORATION WITH DAVID KANAGA

David Kanaga really pushed what's possible with DYAD'S reactive music system. We worked for four months on the first track: He was in Oakland, I was in Toronto. He'd send me stems of the song

he made for the level, complete with chord changes and interaction event sounds, and he'd explain to me how he wanted it mixed and how different game elements should change the mix. I'd send him a video of me playing the game with his system. We did this back and forth for four months until he was able to fly out to Toronto where we worked on the first 15 levels or so. He went back to Oakland, this time with a computer capable of playing the game, and sketched out the final 12 levels, then came back to Toronto to finish everything.

Most people don't believe me when I tell them the music was added last in the game. The game was essentially done with no music, and David was able to create unique music rules for each level to match the game rules, and write 27 unique songs!

what went wrong

SPEED MAKES THINGS COMPLICATED

DYAD is one of the fastest, most complicated games I know; players must track the location and state of their avatar, their immune status/combo status/polarity, and the type/state/location of each enemy. It's almost impossible to process it all without a million tiny elements designed to help you, and if one of these elements isn't perfect the game completely breaks. I'll explain a few of those elements.

Representing depth on a 2D screen is hard, especially when moving quickly. There are two primary elements in the tunnel that make it easier to discern depth: Enemies will leave a faint highlight on the tube, and most enemies also have a trail drawn in front of them.

The player's "space squid" avatar is visually designed to be entirely functional. The bright main color and black/white circle in the center is a purposefully vague representation of its hitbox. Its physical size constantly shrinks and grows. When grazing, the player is about 10 times smaller relative to the enemy in the center of the graze circle than they are relative to normal enemies. When lancing, the player is about 100 times bigger. The player can also be two different sizes depending on polarity.

All of this information is hidden; the player's center is just a vague indicator of position. The trails behind the player exaggerate lateral motion in order to make it easier to see where the player is and where the player is going. DYAD is far too fast for the player to even look at their avatar, so I designed the trails to make it easy to perceive the player's position and motion from peripheral vision only.

The tube design was another important area in maximizing information processing. The tube acts as a reference point for all objects in the game, and needs to feel like a fluid space while

looking pretty. Most levels use a grid pattern to make it easy to discern distance and enemy patterns, and to see what players have lined up.

In DYAD, most levels have a double tube to enhance the perception of lateral speed in order to match the hyper-exaggerated depth speed—if you focus carefully on the player vs. enemy speed, you'll notice the game isn't nearly as fast as it feels. The outer tube is offset such that the inner and outer tubes line up at the bottom where the player is. This increases the visual noise, and draws the eye to the area directly in front of the player. Blending is used to "white out" the area in front of the player to make it easy to see what's going on.

There are many more visual design techniques in play to make information processing as efficient as possible. I didn't expect there to be so many restrictions and would have loved more freedom in the visual design space.

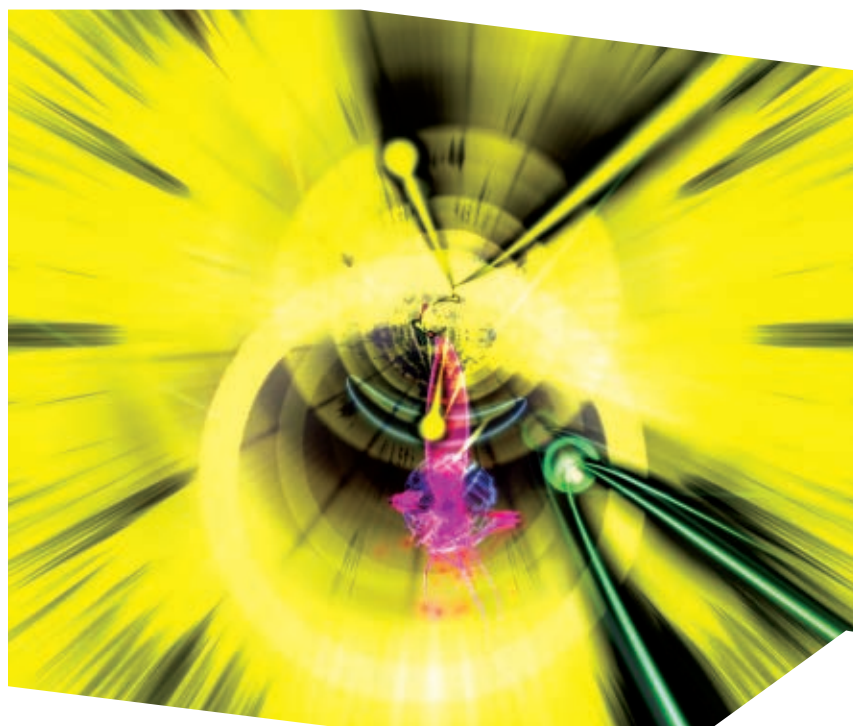
HARD-TO-TEACH MECHANICS

It took me a while to realize that DYAD'S mechanics are fucking weird. I showed the game for the first time at a Scott Pilgrim launch event expecting everyone to be able to pick up and play it; they couldn't. Then I added a reasonable tutorial, and showed it again at an Ontario College of Art and Design event. Still unplayable. It took over a year of near-constant playtesting before anyone could play it without assistance, and at least another year before it had a reasonable learning curve.

The entire game is a tutorial. I had to split the mechanics up into small pieces and teach them individually. The mechanics are boring in isolation, so I had to come up with a bunch of unique goals and modes to keep the game fresh and interesting, with varying subsets of the mechanics available to the player at any time. In the end, this made the game a million times better than it would have been otherwise, but it was extremely irritating to see people completely unable to grasp what I thought were simple concepts. The inexplicable nature of the game led me to a nervous breakdown in early 2011 before I revamped the entire structure of the game.

COMMUNICATING WHAT DYAD IS I really can't describe what DYAD is, which hurt sales more than anything else. I made DYAD to be as "pure" of a game experience as possible, without relying on tropes from other mediums. Communicating the things that DYAD does to your brain while playing is impossible without playing it. I got a lot of inspiration from Vertov's film *Man With a Movie Camera*, which as a movie meant to do only things that are unique to film. I think I did that with DYAD, which made it hard to talk about (and therefore very uninteresting to most people). I wish I could have come up with a way to talk about the game without compromising its game-ness. 

Shawn McGrath is the creator of DYAD.



XCOM: ENEMY UNKNOWN

042 mg

POSTMORTEM_042_2013



pm 043

POSTMORTEM_April 2013

BY GARTH DE ANGELIS

There may have been wounds, but somehow, the XCOM: ENEMY UNKNOWN development team evaded permanent death. ¶ In 1994, Microprose released a special PC game called UFO: ENEMY UNKNOWN. The turn-based strategy title accumulated a devoted fanbase for its unique take on high-level management against an alien invasion blended with boots-on-the-ground, intimate combat controlling individual soldiers. Fast-forward almost two decades, and Firaxis Games has released XCOM: ENEMY UNKNOWN, a reimagining of Julian Gollop's original design. ¶ The road to completing XCOM was an arduous one. We made many of the same mistakes that other devs have made (and documented in previous *Game Developer* postmortems): feature creep, communication shortfalls, not enough time, not enough people... If you're a game developer, you know the story. But through it all, the entire team remained resolute, and in October 2012, against all odds, the development team managed to save Earth ship XCOM on PC, Xbox 360, and PlayStation 3.

WHAT WENT RIGHT

1 UPHELD THE SPIRIT OF THE ORIGINAL Lead designer Jake Solomon planted a small seed within the creative walls of Firaxis Games in 2004. As the self-proclaimed "biggest fan of the original X-COM" and as a designer/programmer working directly with Sid Meier, it made sense to spark discussion regarding resuscitating one of the greatest strategy titles of all time. After shipping CIVILIZATION REVOLUTION, the stars began to align, and the rebuilding of XCOM became a reality. Before any code was written or any art completed, Jake defined core pillars that would act as the foundation for





pillars that would act as the foundation for designing the rest of the game. All facets of game development had clearly evolved since 1994, from game and narrative design to user interface, but Jake remained adamant that certain high-level elements from the original remain holy. These inspirations included maintaining a turn-based combat system, which seemed risky in the modern age of frenetic first-person shooters and real-time action games. The game would also preserve the symbiotic relationship between the micro combat layer and a macro, grand strategic mode, where the player runs and builds their own secret headquarters to counter the simulated alien invasion. Fans of the original game understood the appeal of these interdependent systems, but to those unfamiliar with the XCOM franchise, this was a foreign game structure, and that equaled potential big risk.

Other smaller pillars included: updating systems such as the fog of war and how visibility would be communicated to the player in a 3D environment; fully destructible environments, a satisfying staple from UFO: ENEMY UNKNOWN, but a potential challenge in a game of this scope; reintroducing players to permanent death, and the fact that when it comes to XCOM soldiers, there are no such things as extra lives; and recreating the tense atmosphere from the original. It was critical to place the player in a world they recognized—in settings they may recall from their own neighborhood or city—and then introduce a menagerie of new and classic aliens into these usually safe environments. This led to an unnerving despair that XCOM fans are all too familiar with. Collectively, these pillars provided the foundation for a challenging experience that presented true consequences, just like UFO: ENEMY UNKNOWN.

2 BUILDING THE VISION EARLY Firaxis underwent a pitch process that previously had not been attempted with past projects, but then again, we were pitching what was essentially a new IP as a big-budget endeavor, so we needed to make a major splash. Getting the green light to remake an antiquated turn-based strategy game would be a challenge, and simply verbalizing what made XCOM so special and ripe for a rebirth wouldn't be

enough. The team needed something that would make everyone understand why we were so passionate about undertaking this project—something beyond a static presentation.

Jake began working with project art director Greg Foertsch and a small band of artists to create a gameplay previsualization. Over the course of multiple months, Greg and his team ate, breathed, and slept UFO: ENEMY UNKNOWN, ultimately planning a storyboarded sequence that would illustrate how Firaxis's take on the XCOM universe would not only look, but also how it would play. On Fridays, Jake would sit in a room with the team while they immersed themselves in the original game, becoming familiar with its nuances and big concepts alike. This collaboration led to a compelling previsualization that not only got the development team on the same page, but also communicated to nondevelopers the potential for a classic turn-based experience to be reborn as something cutting-edge, distinctive, and thrilling in the modern age of video games.

3 OVERCOMING THE “ACCESSIBLE” STIGMA Even before beginning work on XCOM, we heard it all before: Games had become too easy. The development (or marketing) buzzword “accessible” translated to “dumbing down,” the idea that developers would take an otherwise deep, rich, and satisfying game and distill its intricacies to its barest form so the entirety of the world could understand, buy, and play said game. It sounds hyperbolic, but I've seen games with easy modes that literally played themselves, making failure impossible, so this stigma against accessibility wasn't without merit! Making a game “for the masses” could be the ultimate transgression, especially for a complex game with a hardcore past, and we anticipated that XCOM fans would be skeptical that our work would hold up to those who fell in love with the original.

While UFO: ENEMY UNKNOWN may have been magnificent, it was also a unique beast when it came to beginning a new game. We often joked that the diehards who mastered the game independently belonged in an elite club, because by today's standards the learning curve was like climbing Mt. Everest.

As soon as you fire up the original, you're placed in a Geoscape with the Earth silently looming, and various options to explore within your base—including reading (unexplained) financial reports, approving manufacturing requests (without any context as to what those would mean later on), and examining a blueprint (which hinted at the possibility for base expansion), for example—the player is given no direction.

Even going on your first combat mission can be a bit of a mystery (and when you do first step off the Skyrainger, the game will kill off a few of your soldiers before you even see your first alien—welcome to XCOM!). While many fans on the team found this learning curve to be a part of the game's charm and wore it as a badge of honor, we ultimately knew that, in 2012, we needed to enable gamers to experience the truly fun elements without overly testing their patience. But neither could we bear to dumb XCOM down.

We were on a mission to flip the perception on streamlining, to remove the stigma that accessibility equaled a dirty word. We wanted anyone to be able to give XCOM a whirl without expecting them to become fluent in the game's many systems on their own accord. At the same time, we needed to preserve all of the richness, depth, and challenge ingrained in the core pillars. If someone wanted to walk away from the experience due to the game's challenge, we were okay with that; but we didn't want to alienate anyone simply due to a lack of information. To accomplish this, we built an optional, integrated tutorial that peeled off the components of XCOM one layer at a time. It was important to keep this hour-and-a-half experience optional, as experienced players could save Earth again without the tutorial force-fed to them (and we also knew some players, even in 2012, would want that old-school badge of honor by skipping the tutorial altogether, which is somewhat appropriate for certain types of X-COM fans).

The introduction to the game wasn't the only area we redesigned. Jake and the design team refined low-level mechanics from the original, such as removing Time Units and capping the squad loadout at six. Both of these changes were the result of internal playtesting over the course of many months, with the development team finding a combat "sweet spot" with respect to approximate time spent on a map and number of decisions made per turn (we found, depending on map size, battles should average 20 minutes, not to exceed 50 minutes on the absolute longest missions). Six units also made every decision vitally important, promoting group tactics with no moves feeling like unnecessary filler.

This "new era of accessible" mindset also helped the design and user interface teams build a platform-agnostic experience. This is an element that could have gone horribly wrong (and did have its inherent challenges, detailed later), but the team did an admirable job of crafting a historically PC experience for consoles as well. We knew games like XCOM weren't traditionally available on Xbox 360 and PlayStation 3, but we're extremely happy we could provide the same experience (without compromising features or "dumbing down" the console versions) across all platforms.

4 DEDICATED TEAM EFFORT The developers at Firaxis are extremely professional, with each discipline playing the hero role at some point and overcoming monumental obstacles throughout development. From the audio group to the animation and narrative team, they were continually course-adjusting due to dependencies, yet still producing incredible content to polish the game.

On the engineering front, months of changing design had to be technically supported in many complex situations. Systems were built, iterated upon, and some were even discarded after determining a new direction was needed. For example, over the course of a few milestones in midproduction, design asked for sightlines to be drawn from every game unit, soldier, and alien alike, so it was clear what each unit could see. Our graphics engineering team and artists diligently worked to make this system digestible, but unfortunately, it was tough for the player to determine what was going on amidst the plethora of multicolored lines. After months of trying to get sightlines to work, we eventually realized that the strongest solution was to remove them.

There were plenty of other challenging systems to decipher: the building visibility system underwent various ceiling, wall, and



floor rule changes; destruction fidelity fluctuated through a shaky toughness system; and the fog of war was a full 3D cloud early in production, which proved to be a nightmare for both graphics engineering and performance. Additionally, each gameplay layer (combat and strategy) received drastic overhauls after months of playtesting. In all of these cases, initial engineering efforts had to ultimately be thrown out. To the team's credit, they understood the nature of iterative design and admirably continued to put in the time needed to make the game a better experience.

In addition, the engineers banded together in a Herculean effort to fix thousands of bugs in postproduction. XCOM is a large, system-driven game with many procedural elements. This meant that many bugs were not only difficult to reproduce, but challenging to even find! Together, engineering raised the bar of the final player experience by squashing these bugs feverishly. Obviously, we couldn't find and fix every bug, but we're proud of the effort given in the race to the finish line.

The art and content teams also worked minor miracles. A primary example was the game's levels. We all love maps and levels, and want more of them; but they are a nexus of many different disciplines somehow crafting the same sculpture all together, and this requires tight coordination and lots of time. Firaxis had never created a level-driven game before (with a strategy system still on top of it, no less), so we had to learn how to build a pipeline that would let us efficiently design and build level assets. This specifically required an inordinate amount of collaboration between level design and level art, weeks of gameplay testing and feedback per map, and an extreme amount of content creation (we needed to have approximately enough maps for two full playthroughs). In the end, our modestly sized level team ended up exceeding the original goal of 70 unique maps.

Beyond levels, there was still an entire headquarters to build on the strategy layer, with dozens of expandable rooms that could be hand-placed by the player. After making various isometric

prototypes, we realized the base wasn't nearly as gripping as we'd like; something was missing. Lead technical and HQ artist Dave Black pitched the "ant farm," a diorama-style side view that instantly connected with the entire team. This was an entirely new process as well, but Dave and the art team concurrently exceeded expectations on headquarters while finalizing all of the combat maps.

5 2K/FIRAXIS PARTNERSHIP We've heard countless horror stories about publisher-developer relations, with publishers stifling creativity, dictating direction, or creating impossible deadlines—but our partnership with 2K Games was not one of those horror stories. While there was give-and-take from both sides (as in any relationship), we were overwhelmingly happy with 2K Games's support—especially considering no major publishers have funded a large-scale, multiplatform, turn-based strategy game in recent memory.

2K believed in our vision and greenlit the project, something we're not so sure would have happened elsewhere. The 2K Product Development group believed in the potential for a reimagined XCOM and also understood that taking risk was necessary. We were ecstatic to learn we would be given this opportunity.

Furthermore, 2K trusted in us as a studio to own the creative direction of the title. While they provided in-depth milestone feedback, every item was up for discussion, and they ultimately trusted in our design vision. 2K also provided us with additional resources to build an integrated tutorial, something that became critical late and ballooned beyond our initial resource estimations. This type of support proved invaluable to finish the game.

Also, 2K's public relations team was instrumental in raising the awareness for XCOM. They took the time to understand the vision and value of the project, and allowed the team leads to directly and candidly communicate that vision to the player base. PR worked diligently to uncover many valuable opportunities for the game, including a cover reveal with Game Informer magazine and various





demo presentations to targeted press. These presentations planted the seed in our most passionate advocates—the press—to pass along what they liked (or disliked) about the game’s potential. There was also a strong working relationship between PR and the development team, leading to joint initiatives like the “Jake Solomon Undercover” video (<http://bit.ly/T1TPK8>) and exciting panel discussions like PAX’s “1000 Stupid Ideas on the Road to Glory.”

WHAT WENT WRONG

1 DESIGN CONTINUED INTO POST-PRODUCTION XCOM required constant design iteration, with some features being implemented beyond Alpha. It may sound cliché, but Firaxis has always lived by the mantra “Find the Fun,” and the company takes that very seriously. Sometimes, fun can be a challenge to find, especially in a product that is unlike any other we’ve built before. XCOM boasts two interdependent systems that could almost be standalone games, and discovering that special synergy between the two was the key to unlocking the magic within the XCOM universe.

Trying to focus concurrently on both gameplay layers was challenging. We spent various milestones on certain features that didn’t progress as we’d hoped. By midproduction, the strategic layer was a turn-based card system for various months, and it stagnated while the team focused on improving combat. Ultimately, the strategy layer was molded into the version we’re satisfied with, but it was neglected for too long and required a late HALF-LIFE-inspired Cabal process to get there. We (myself, Solomon, and other members of the dev team as necessary) would meet every morning, every day, until each component of the strategy layer had a concrete game plan and a clear implementation schedule.

Additionally, the tutorial and narrative, critical components of the game, couldn’t be pushed to final until the design was locked. And since the design tentpoles ran late, the narrative team (including animators, writers, and audio) came under immense pressure to finalize high-quality cinematics in an extremely short timeframe.

The extra design time helped make the game as good as it could possibly be from a gameplay perspective, but it’s worth asking whether we could have made tough calls on certain systems earlier in the schedule. This is one of game development’s largest challenges: Holding a game’s design to immovable deadlines can be stifling to the iterative and tricky-to-quantify creative process. Shipping an unpolished combat game with a completely disconnected strategy layer would have spelled disaster for the future of XCOM, so we kept the process malleable much later into the schedule, allowing the team to find the answers through discovery and experimentation.

Practices like the design cabal helped the team focus on areas of the game that weren’t fun, but in a perfect world, we would have locked down as many high-risk systems as possible as preproduction wrapped up. We did ultimately cut content, but the bulk of our wishlist shipped in the final product, which was great for the game but taxing on the team.

2 ADDITIONAL HELP MEANT ADDITIONAL WORK By Alpha, only a few systems needed to be implemented, but a new challenge was looming around the corner: the bug database. Before Alpha, the team had a good sense of the state of the game and which systems were most playable, but it was difficult to quantify the true workload until QA began fully testing the game for a few weeks. The reported bug count rapidly multiplied like termites silently infesting the framework of an otherwise beautiful house. Initially,

XCOM: ENEMY UNKNOWN

we weren't quite sure what to expect, but as the picture became clearer, we knew we were in for an inordinate effort to keep pace with the influx of bugs. There were concerns about the amount of work needed to fix the game relative to the engineers on the team. We had a ship date to hit and we wanted to get our dedicated engineers help.

But the mythical man-month is a very true concept. While our publisher was generous with additional resources to assist toward the end of production, we found that a flood of external helpers had undesired consequences. Knock-on bugs due to unfamiliarity with the codebase, content that needed to be fixed by internal artists, and communications inherent in outsourcing relationships all led to an extreme amount of overhead that ultimately fell onto the laps of internal team members who were already responsible for an aggressive workload.

Outsourcing challenges also hit the content-creation team during production. Communicating with an external cinematic team overseas led to a staggered communication channel. Since there were dozens of unanticipated clerical issues just to get their tools up to speed with ours (no fault of theirs), it was extremely challenging to troubleshoot any setbacks. Also, providing creative feedback to most external partners often led to significant delays due to the remote feedback loop and misinterpretations of feedback via email.

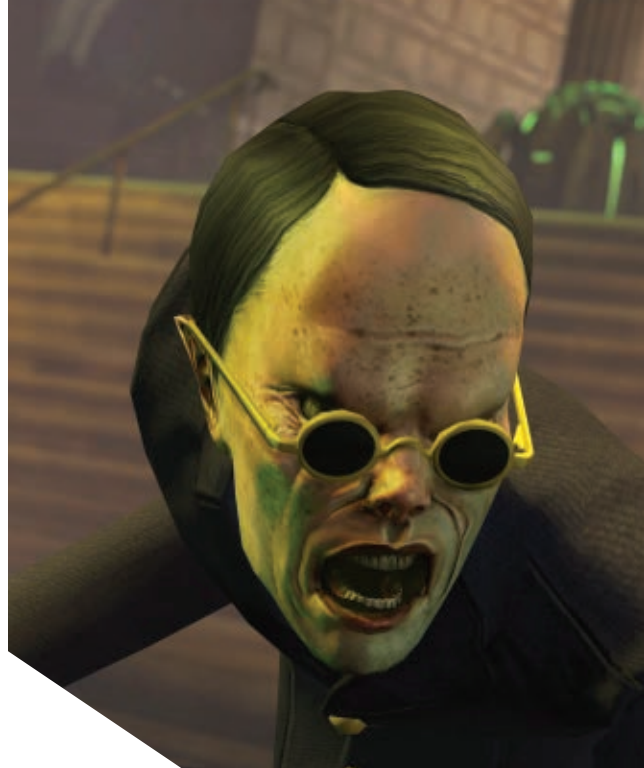
3 LACK OF COMMUNICATION Once we were late into production, the leadership team wanted to maximize each developer's working hours by being judicious regarding meeting requests, even amongst ourselves. Process-driven meetings were reduced along with costly, 20-plus-person large-scale meetings. We still maintained informal but intimate one-on-one reviews with each discipline's lead, which was intended to be more focused and fruitful per developer. While the leadership team and some team members appreciated this, others were understandably yearning for additional official communication channels. Also, team members wanted quicker information on the high-level changes to the design of the game, but with our lead designer doubling as a gameplay engineer, he would often be tied up with coding. Finally, cross-discipline groups (like level design and level art, and feature-specific teams) surely could have benefited from a more formalized stand-up process, which we implemented toward the end of production.

Moving forward, the leadership team knows it needs to strike an appropriate balance between optimal information flow and excessive meeting time, hedging toward more opportunities for formal communication.

4 NEW, MULTIPLATFORM CHALLENGES Not only was the game structure of XCOM unlike anything the studio had built before, this was also the first time we've had to concurrently develop versions for three different platforms. It turned out managing all three was a massive amount of work.

Design-wise, the team knew there would be feature parity between PC and consoles; the only difference would be the control scheme. While the design and UI team did an admirable job on this front, there were continuous challenges throughout development to accommodate multiplatform user interface design, specifically tied to this genre. The team had to ensure all tactical commands were accessible via gamepad, and this involved quite a bit more than accommodating a point-and-shoot mechanic. The movement system, mapping a system to support dozens of contextual abilities, and crafting a uniform Shot HUD were just a few areas that took time to master across the board. While this specific instance arguably didn't go "wrong," it is a small example of the multiplatform challenges faced daily.


The system-specific optimizations needed for each platform were significantly more difficult, particularly for the consoles. Understanding the console constraints for items like number of maps, audio files, texture budget, and animation sizes was a continual process between engineering and the specific disciplines. There



were also severe, system-specific bugs, technical requirements, and crashes that ate up much of our senior engineers' time.

Our systems engineering team was a very talented duo, but they didn't have a dedicated platform engineer, which meant that they had to partner on all of these complex issues across the board. While they worked together effectively, they simply had too much work on their plates: universal systemic issues, art optimization requests, and other general and technical requirement bugs, just to name a few major workloads. Our lead engineer assisted on the most difficult issues when he was free from putting out other fires, and another internal systems engineer joined the cause late in the project to own the Xbox 360 technical certification requirements, but these were solutions that emerged late in development.

5 EXTENDED CRUNCH We're not proud about the fact that we had to crunch to finish XCOM. We have a dedicated and passionate team, and all team members put in serious extra hours at some point for the good of the project. For many, the malleable structure of the game led to frustration as we were knee-deep in the trenches. Certain dependencies were continually pushed (especially impacting audio, effects, cinematics, and user interface) and the lack of testing on late gameplay systems led to a heavy bug load for the engineers. On the art side, the content creators had production crunches to finish all maps. As said before, this was the first time we created a game of this structure, and the first time we had to iterate so much on the process itself. While we improved certain inefficiencies throughout production, we simply couldn't accurately predict how much time we'd need to make the game the way we wanted to make it.

DEVELOPER FROM THE DEEP In the end, we avoided permadeath. And after all of the extra hours, the thousands of bugs fixed, the hundreds of level playtests dissecting every piece of cover, the dozens (hundreds?) of gameplay prototypes and healthy debate that accompanied each new system, through every team meal, and in the wake of every hopeful or concerned hallway discussion, in the end, the XCOM development team emerged victorious. We shipped the project within weeks of our original target release date, earned a near-90 Metacritic from video game journalists, garnered hundreds of game accolades, and won 13 Overall Game of the Year awards. Most importantly, a wildly creative and cross-discipline team banded together to contribute to the unlikely revitalization of a classic game, capturing the magic of X-COM for a new generation of gamers and hardcore fans of the original alike. 

Garth DeAngelis was the lead producer and a level designer on XCOM: ENEMY UNKNOWN.

RAIN{indie}

When the Unity engine came on the scene, it opened up the world of game development to a significantly wider audience. Unity allowed people to sidestep the knowledge, time, and frustration of the complicated process of creating their own rendering engine, lighting effects, physics modeling, and more. Instead, they could get straight to the process of creating worlds, levels, and ultimately games (which is, in and of itself, a complicated process). However, as tickled as people were to be able to dive into Unity and “make things,” one question kept coming up: How do I make AI?

Much like the other systems listed above, creating even simple game AI often takes a lot of investment in infrastructure. Even armed with Mexican food metaphors [see my AI Primer in the August 2012 issue of *Game Developer*], creating AI architectures is not an easy task. Unity users would simply be better off if there was a tool that allowed them to bypass the messy work of creating the underlying infrastructure and get right to the task of creating the actual behaviors. After all, isn’t that what Unity is all about? Well, that’s what Rival Theory set out to do with RAIN{indie}.

AI FOR FREE I was originally introduced to Rival Theory a year ago when they showed me a brief demo of their initial product, RAIN{one}. Its successor, RAIN{indie}, has subtle differences, with a tightened feature set that focuses more on what the smaller developer needs the most access to. More importantly, while the original was a purchased product, RAIN{indie} is available as a free download. That’s a huge bonus to the small developer who may not have the budget for tools.

At the time of this writing, the tutorials and documentation of RAIN{indie} were still evolving, so I was given a tour of the new product’s features by Rival Theory founder and CEO, William Klein. I had already downloaded the product from their site, installed it, and taken a walk through their demo applications. These showed, in varying degrees of fidelity, the end result of what their behavioral engine could do. William’s explanation showed me how the tools are used to achieve those end results. I actually felt a little guilty for taking up his time—once I found out how simple many of the processes were, I realized I probably could have done it myself in short order.

RAIN{indie} is actually a number of different AI

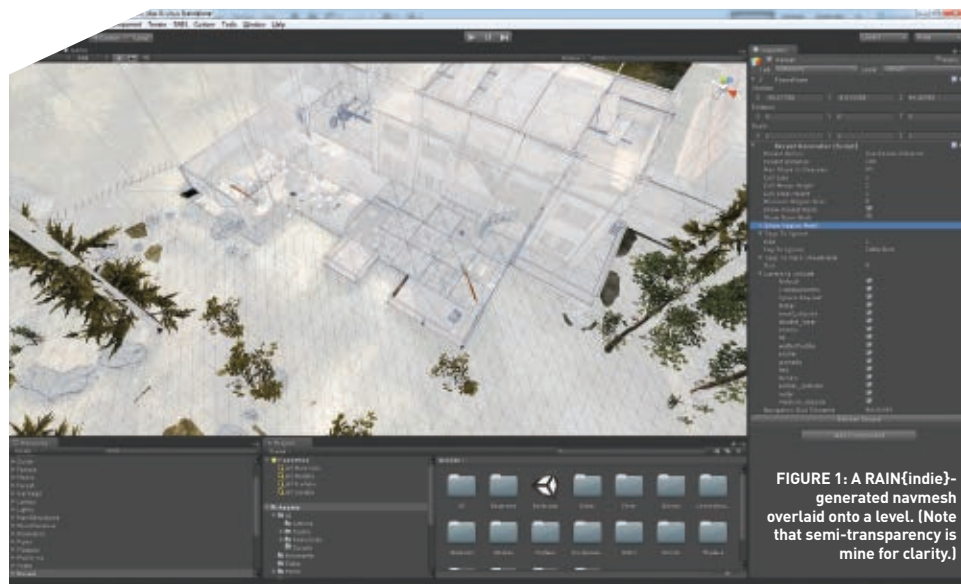


FIGURE 1: A RAIN{indie}-generated navmesh overlaid onto a level. (Note that semi-transparency is mine for clarity.)

products in one. Included in the package are support for creating pathfinding assets, the actual pathfinding code, behavior trees, and a sensory system. The systems are fully separate and can be included individually as desired. Adding them to your project is similar to adding any other sort of Unity add-in. More importantly, once added to your projects, the modules themselves are accessible. This means that you can use them “as is” or in some modified fashion—even combining them with Unity’s default tech or that of another add-in. This flexibility, in and of

itself, is something that should allow users to feel comfortable about incorporating RAIN{indie} into their projects without the chains, cages, or soul-selling that often comes with middleware packages.

PATHFINDING WITH RAIN{indie} The pathfinding components in RAIN{indie} include a voxel-based navmesh generator. Auto-generating a navmesh is as simple as adding a RAIN Recast object to your Unity scene, setting parameters for options such as the desired cell size and maximum traversable angle (e.g., 45°),

and clicking Refresh Recast. Depending on your region size, you have a generated navmesh in seconds or minutes. Oddly, the generated navmeshes are grid-based rather than the typical “odd-shaped polygon” ones that many of us are accustomed to. At a cell size of 1, this makes for many cells—even in large open areas. The result looks more like an odd hybrid of a very dense nav graph rather than a true nav mesh. You can reduce the number of cells by specifying a larger size, but as with any resolution change, the fidelity of the auto-created walkable

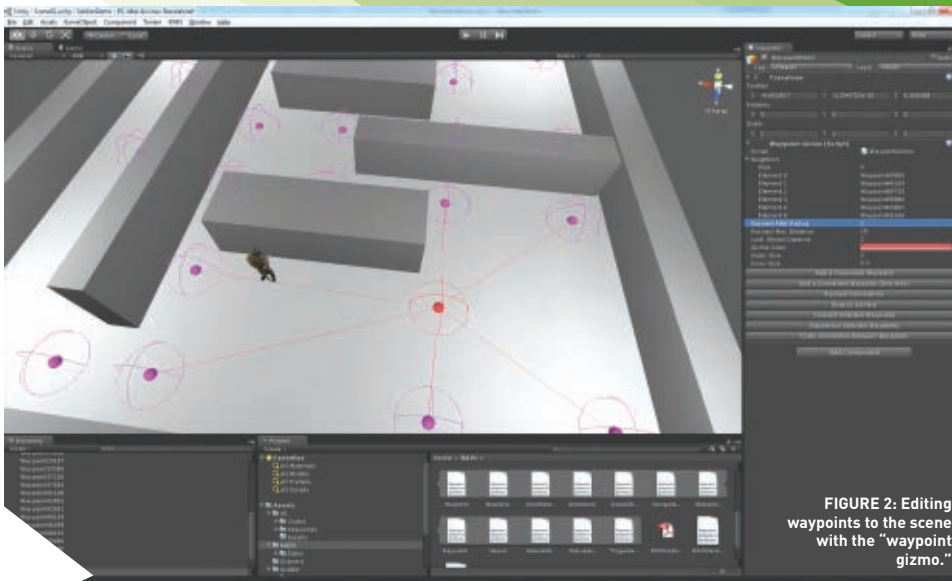


FIGURE 2: Editing waypoints to the scene with the “waypoint gizmo.”

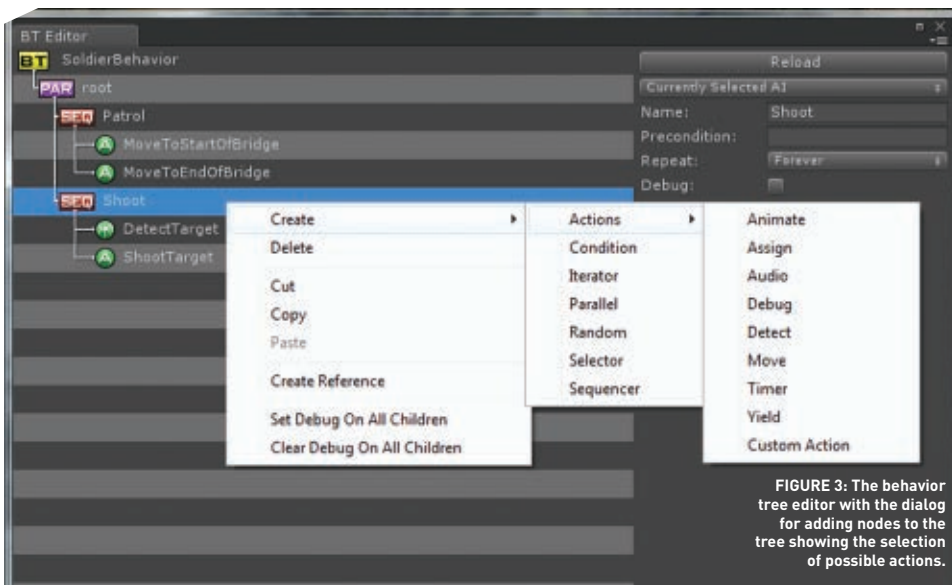


FIGURE 3: The behavior tree editor with the dialog for adding nodes to the tree showing the selection of possible actions.

RAIN{indie}

Rival Theory rivaltheory.com/rainindie

PRICE:

Free

SYSTEM REQUIREMENTS:

Any computer capable of running Unity (Mac/PC)

PROS:

- 1 Components are usable and editable directly in Unity
- 2 Graphical behavior tree structure editor
- 3 Hellooo? It's FREE!

CONS:

- 1 Still only a framework—not a magic bullet
- 2 Documentation and tutorials still “in process”
- 3 Nonstandard, grid-based navmeshes

areas suffers. Regardless, the solution is certainly workable (see **Figure 1**).

On top of your navmesh, you can add waypoints to your environment, connect them to each other, and hook them up as paths to assign to your agents. Using the “waypoint gizmo,” navigation nodes can be dragged and dropped into the environment and automatically connected to each other based on raycast-based line-of-sight checks (see **Figure 2**).

In addition to pathfinding, RAIN{indie} has built-in collision avoidance and steering behaviors. It uses a form of “look ahead” steering that

responds to terrain, static, and dynamic obstacles. I played with some of the samples in the steering demo (really just capsules gliding through an environment), and they performed respectably even when there was more than one dynamic obstacle. There was one agent in the demo who would get a little hung up on a combination of terrain and an obstacle, but managed to muddle through. To his credit, he was using steering only where a navmesh would have helped out the situation.

While not necessarily as robust in complex situations as more-involved, custom solutions, the steering in

RAIN{indie} performed well enough to create intelligent avoidance in simple dynamic environments.

SENSORY SYSTEM AND BEHAVIOR TREE EDITOR

RAIN{indie} also includes easy-to-attach sensors that help streamline the setup of the agent's detection of objects in the world. In essence, these amount to colliders that are looking for intersections with specified objects or types of objects. You can actually specify sensors for not only vision, but also for touch, sound, smell, and yes... taste. Really, there isn't much difference between them—a collider of a specified size and shape, and some tags to define what it is attempting to sense. Theoretically, you could define the sense as anything you wanted: ghosts, tachyon fields, teenage angst, whatever. It is amusing, however, to hook up a “taste sensor” to your agent—if only conceptually.

While all of the above features are nice, the behavior tree component in RAIN{indie} is something that really caught my eye. For those that don't know, behavior trees are becoming increasingly popular as the go-to architecture for crafting AI. They are both easy to understand and powerful—so much so that many of the triple-A games today are using some form of behavior tree architecture. Thankfully, because of how they are constructed, they also lend themselves to being constructed and manipulated with visual design tools such as the one included with RAIN{indie}.

Again, as with the other components in RAIN{indie}, attaching a behavior tree to a character only involves a few mouse clicks. Once that “mind” is in place, adding, moving, and editing nodes of the tree are fairly straightforward. Not only can you insert typical node types such as sequencers and selectors right from the tree interface, you can also assign animations, sound events, and more (see **Figure 3**).

Editing the entire tree graphically is as easy as



FIGURE 4: A larger behavior tree expanded in the editor. Note that some of the parameters are editable from the properties screen.

dragging and dropping. This is important, of course, because, as with any AI development, constructing behavior trees is often a very iterative process. To not have to worry about xml braces, tags, indenting, etc. is very relieving. Also the graphical tree structure is easy to read and helps you visualize the overall structure of your AI. Key parameters for the selected node are exposed right in the tree editor so that browsing the tree is simple and intuitive (see **Figure 4**).

Naturally the nodes themselves are not the end of the journey. If you want, RAIN will do a lot of the heavy lifting for you; the editor will create the scripts for you in JavaScript, C#, or Boo. However, depending on what your behaviors do or what decision logic you need to leverage, you may still have to write some code on your own. This is a very key point, though. The fact that the BT editor is writing code for you means that you can edit

that code. Many middleware solutions are closed, black box systems; you do it their way whether you want to or not. With RAIN{indie}, you can lean on the system to do most of it on its own, or you can utilize only the framework and write the bulk of the code on your own. This is a huge boon that allows the product to scale gracefully from the casual dabbler to the more advanced user.

MAKING IT RAIN All told, RAIN{indie} brings a lot to the table. I feel somewhat remiss as a reviewer since there is no way that I have completely kicked all the proverbial tires on the product. That means there might be more for me to discover—both positive and negative, of course. As they flesh out the documentation and tutorial videos, getting to know the ins and outs of the different features will certainly be easier.

Rival Theory seems to have accomplished what it set out to do, however: Make creating AI for Unity characters simple yet powerful. Another thing it certainly did right is the price. Regardless of any of the features, benefits, and caveats that RAIN{indie} provides, there is really no risk in giving it a test run. †

Dave is the president and lead designer of Intrinsic Algorithm, an independent game development studio and AI consulting company in Omaha, Nebraska. He is the author of the book Behavioral Mathematics for Game AI and is a contributor to the AI Game Programming Wisdom and Game Programming Gems book series from Charles River Media. Dave is also a founding member of the AI Game Programmers Guild, has spoken at numerous conferences, and was a co-advisor for the previous AI Summits at GDC.



When you're working at a job you love, creating games that will be played by people around the world, Monday doesn't seem half bad. In fact, you might even learn to love it. At Bally Technologies, we're busy creating the most cutting-edge games, systems, and mobile apps on the planet, and we've got a track record that'd make your parents proud, right after it makes their jaws drop.

And we haven't even touched on all the great benefits, travel opportunities, unrivaled job security, and generous pay. If you've ever had a bad case of the Mondays, the cure is waiting for you at booth #2334. Drop by, and find out how you can make a living doing what you love.

We want you. Mobile & Interactive Content Developers • Flash/Flex & 3D/Unity Programmers • C++/Linux Programmers • PHP & Scala Web Developers • Graphic Artists & Digital Illustrators • Motion Graphics Artists & 3D Animators • Mathematicians • And More!



BallyTech.com/Careers
jobs@ballytech.com





entertainment[®]
software
association



WHAT'S JUNE 11-13, 2013
LOS ANGELES, CALIFORNIA
NEXT NOW
E3Expo.com

BE A PART OF WHAT'S NEXT IN
THE VIDEO GAME INDUSTRY.

Register at E3Expo.com

Produced By

IDG
WORLD EXPO

E3 is a trade event and only qualified industry professionals may attend.
No one under 17 will be admitted, including infants. Visit www.E3Expo.com for registration guidelines.
© 2013 Entertainment Software Association

International
Media Sponsor

MCV

PROGRAMMER, INTERRUPTED

STRATEGIES FOR AVOIDING INTERRUPTED CODING SESSIONS

I'm writing this article in a dull state: low sleep, busy, disorientated, and interrupted. I try all the remedies: using the Pomodoro Technique, working in coffee shops, wearing headphones, and avoiding work until being distraction-free in the late night. But it is only so long before interruption finds a way to pierce my protective bubble. Like you, I am "programmer, interrupted." Unfortunately, our understanding of interruption and remedies for restoring focus are not too far from homeopathic cures and bloodletting leeches. But what is the evidence, and what can we do about it?

THE COST OF INTERRUPTION Every few months I see another programmer asked to not use headphones during work hours or interrupted by meetings too frequently to do any work, who has little defense against these demands. I also fear our declining ability to handle these mental workloads and interruptions as we age.

Researchers who have studied the costs of interruptions in office environments estimate that interrupted tasks take twice as long and contain twice as many errors as uninterrupted tasks. They also found that workers have to work in a fragmented state, because 57% of tasks are interrupted (see **References** for citations).

For programmers, there is less evidence of the effects and prevalence of interruptions; typically, the number that gets tossed around for getting back into the "zone" is at least 15 minutes after an interruption. Interviews with programmers produce a similar number. Nevertheless, numerous figures in software development have weighed in: Y Combinator founder Paul Graham stresses the differences between a maker's schedule and a manager's schedule, and 37signals founder Jason Fried says the office is where we go to get interrupted. >>>

STUDYING PROGRAMMER

INTERRUPTION Based on an analysis of 10,000 programming sessions recorded from 86 programmers using Eclipse and Visual Studio, and a survey of 414 programmers, we found:

- ◆ A programmer takes 10-15 minutes to start editing code after resuming work from an interruption.
- ◆ When interrupted during an edit of a method, a programmer resumed work in less than a minute only 10% of the time.
- ◆ A programmer is likely to get just one uninterrupted two-hour session in a day.

We also looked at some of the ways programmers coped with interruption:

- ◆ Most sessions programmers navigated to several locations to rebuild context before resuming an edit.
- ◆ Programmers insert intentional compile errors to force a “roadblock” reminder.
- ◆ A source diff is seen as a last-resort way to recover state, since it can be cumbersome to review.

THE WORST TIME TO INTERRUPT A PROGRAMMER

Research shows that the worst time to interrupt anyone is when they have the highest memory load. Using neural correlates for memory load (by measuring pupil diameter, for example), studies have shown that interruptions during peak loads cause the biggest disruption (see **Figure 1**).

In our study, we looked at subvocal utterances during a programming task to find different levels of memory load during programming tasks (see **Figure 2**). When people perform complex tasks, subvocal utterances (electrical signals set to the tongue, lips, or vocal cords) can be detected. This phenomenon has long intrigued researchers, some likening subvocal signals to the conduits of our thoughts. Recently, researchers have even been able to decode these signals into words.

If an interrupted person is allowed to suspend their working state or reach a “good breakpoint,” then the impact of the interruption can be reduced. However, programmers often need at least seven minutes before they transition from a high memory state to a low memory state. An experiment evaluating which state a programmer less desired an interruption in found these states to be especially problematic:

- ◆ During an edit, especially with concurrent edits in multiple locations.

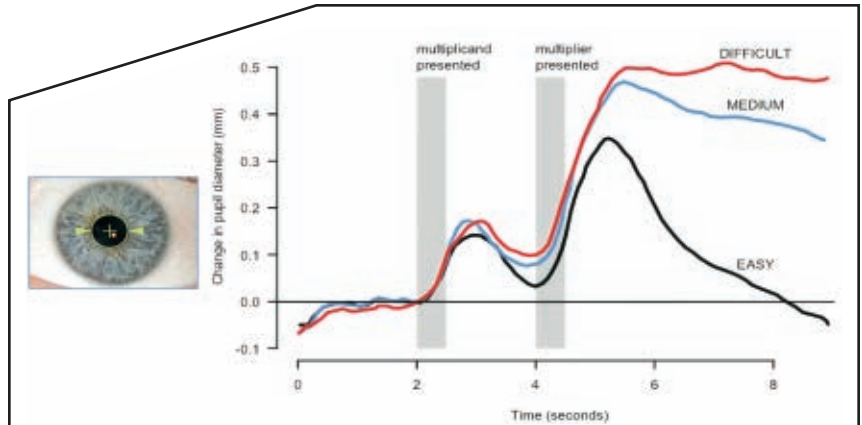


FIGURE 1: Tracking the change in pupil diameter over time for individuals given tasks of varying difficulty.

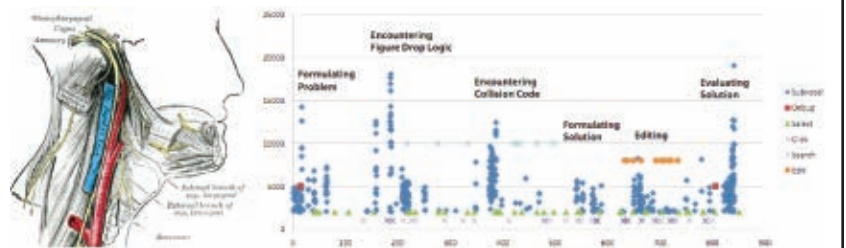


FIGURE 2: Electromyogram (EMG) signals correlated with a 13-minute programming task for modifying a Tetris game.

- ◆ Navigation and search activities.
- ◆ Comprehending data flow and control flow in code.
- ◆ IDE window is out of focus.

STRUCTURING YOUR ENVIRONMENT TO SUPPORT YOUR MEMORY

Ultimately, we cannot eliminate interruptions. (In some cases, interruption may even be beneficial; 40% of interrupted tasks are not resumed, and possibly because we realize that the task is not as important, or because the interruption gives us a chance to reflect on the problem.) But we can find ways to reduce the impact on the memory failures that often result from interruption. In this next section, I’ll introduce some types of memory that get disrupted or heavily burdened during programming, and discuss some conceptual aids that can support them.

PROSPECTIVE MEMORY

Prospective memory holds reminders to perform future actions in specific circumstances—for example, reminding you to buy milk on the way home from work.

Various studies have described how developers have tried to cope with prospective memory failures. For example, developers often leave TODO comments in the code they are working on. A drawback of this mechanism is that there is no

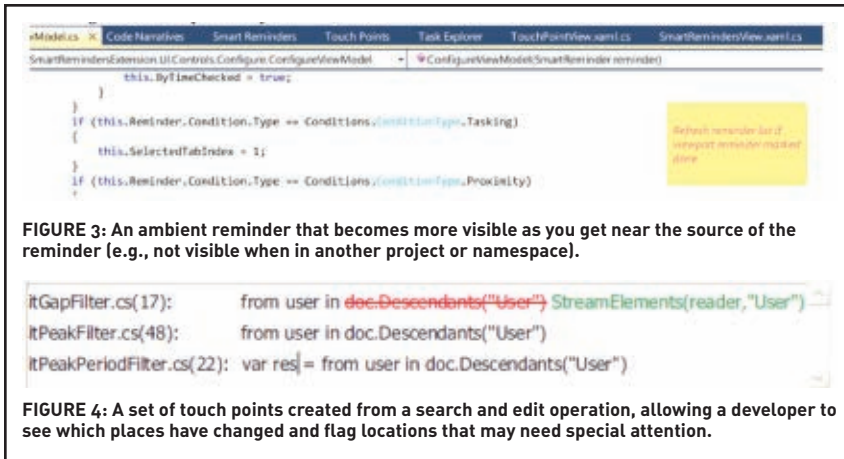
impetus for viewing these reminders. Instead, to force a prospective prompt, developers may intentionally leave a compile error to ensure they remember to perform a task. However, introducing compile errors creates a different problem, because they inhibit the ability to switch to another task on the same codebase. Finally, developers also do what other office workers do: leave sticky notes and emails to themselves.

A “smart reminder” is a reminder that can be triggered based on specific conditions, such as a teammate checking in code, or spatial proximity to a reminder (see **Figure 3**). It’s basically the programming equivalent of a Post-It note.

ATTENTIVE MEMORY

Attentive memory holds conscious memories that can be freely attended to.

This can come up in programming when a developer has to make similar changes across a codebase—for example, if a developer needs to refactor code in order to move a component from one location to another, or update the code to use a new version of an API, then that developer needs to systematically and carefully edit all those locations affected by the desired change. Unfortunately, even a simple change can lead to many complications,



ASSOCIATIVE MEMORY

Associative memory holds a set of nonconscious links between manifestations of co-occurring stimuli.

Developers commonly experience disorientation when navigating to unfamiliar code. The disorientation stems from associative memory failures that arise when a developer must recall information about the places of code they are viewing or what to view next. Researchers believe the lack of rich and stable environmental cues in interface elements, such as document tabs, prevent devs from recalling associative memories.

The presence of multiple modalities in a stimulus increases the ability to form an associative memory. In this sense, a modality refers to a distinct type of perception that is processed by distinct regions of the brain, such as auditory or visual pathways. Examples of different modalities and corresponding stimuli include: visual (error underline bars, highlighting code), lexical (name of file), spatial (position of scroll bar or tab), operational (edit/search/debug step action on file), and structural (logical position of file in hierarchy).

When multiple modalities are present in the same stimulus, more pathways are

requiring the developer to track the status of many locations in the code. Even worse, after the interruption, the tracked statuses in attentive memory quickly evaporate and the numerous visited and edited locations confound retrieval.

Touch points (see **Figure 4**) allow a programmer to track status across many locations in code. Studies examining refactoring practices have found several deficiencies in tool support, and one of those is the lack of ability to track many

locations in code. As a workaround, developers abandon refactoring tools and instead rely on compile errors that were introduced when refactoring. Unfortunately, using compile errors to track changes is not a general solution and can still lead to errors. Touch points are inspired from how developers use compile errors. They can be recovered automatically by deriving all points of code recently visited, edited, and searched for.

GAMASUTRA JOBS

YOUR GAME INDUSTRY CAREER RESOURCE

- Find the most sought after positions with top companies around the world
- Put your resume in front of industry leading employers via the Resume Database
- Source talent on Gamasutra – the game industry's thought leader



gamasutra.com/jobs

A timeline of programming activities can help you or your teammates remember how you did a task and the resources you used.



Customizing Visual Studio Search Results
Display of search results in Visual Studio has been remarkably primitive despite numerous improvements in other areas. Eclipse, in contrast...
blog.ninlabs.com

Experimental > PasteUrls > ProvenanceViewModel.cs

```
// Get search results window 1
IVSUIShell shell = (IVSUIShell)this.GetService(typeof(SVSUIShell));
Guid windowGuid = new Guid(EnvOte.Constants.vswindowFindResults1);
IVSWindowFrame windowFrame = null;
shell.FindToolWindow((uint)_VSFINDTOOLWIN.FTW_FFIndFirst, ref windowGuid
```

[asp.net - MEF can load an assembly but doesn't load any parts - Stack Overflow](#)

[html - insert vertical divider line between two nested divs, not full height - Stack Overflow](#)

[xaml - How to change the color of the a WPF <Separator />? - Stack Overflow](#)

 [BubbleService.cs](#) yesterday

```
return bubbles.OrderByDescending( b => b.Start ).ToList();
```

activated, thus increasing the chance of forming an associative memory. In contrast, a monotonous stimulus with a single modality is less likely to form an associative memory.

An associative link helps a programmer by situating information of multiple modalities with a program element; observations of developers suggest that they frequently rely on associations with environmental cues, such as tabs and scrollbars, for maintaining context during navigation. However, these cues are often insufficient: The act of navigation often disturbs the state of the cues, and the paucity of interface elements such as tabs, which often contain only a file name, starves associability. By improving navigating document tabs, which in default configurations are especially spartan, often showing just the name of the document, we could see increased recall from associative memory.



Two tabs adorned with cues of different modalities: such as error lines (visual) and edit icons (operational).

EPISODIC MEMORY

Episodic memory is the recollection of past events. Software developers continually encounter new learning experiences about their craft. Retaining and making use of such acquired knowledge requires that developers are able to recollect those experiences from their episodic memory.

When recalling from episodic memory, developers commonly experience failures that limit their ability to recall essential details or recollect the key events. For example, a developer may forget the changes they performed for a programming task, or forget details such as a blog post that was used for implementing part of the task.

A code narrative is an episodic memory aid that helps a developer recall contextual details and the history of programming activity. Different types of narratives can be supported; for example, a review mode for

high-level recall of events and a share mode for publishing a coding task for others.

For more on code narratives, check out this blog (<http://codenarratives.tumblr.com/>), which is shared and published semiautomatically via a code narrative. (Note that as of this writing, the blog hasn't been updated recently.)

CONCEPTUAL MEMORY

Conceptual memory is a continuum between perceptions and abstractions. How does the brain remember objects such as a hammer and concepts such as "tool"? Well, it first learns basic features of encountered stimuli, such as the wood grains and metal curves of a hammer, and then organizes those features into progressively higher levels of abstraction.

Developers are expected to maintain expertise in their craft throughout their careers. Unfortunately, the path to becoming an expert is not easily walked: For a novice, evidence suggests this can be a 10-year journey. Furthermore, for experts trying to become experts in new domains, like the desktop developer becoming a web developer, there are many concepts that must be put aside and new ones learned.

Studies examining the difference between an expert and novice find that performance differences arise from differences in brain activity. Not only do experts require less brain activity than novices, they also use different parts of their brains: Experts use conceptual memory whereas novices use attentive

References

For a full list of citations and references, read the original post here: <http://blog.ninlabs.com/2013/01/programmer-interrupted/>

A diary study of task switching and interruptions (Czerwinski) <http://dx.doi.org/10.1145/985692.985715>


No task left behind?: examining the nature of fragmented work (Mark) <http://dl.acm.org/citation.cfm?doid=1054972.1055017>

Resumption strategies for interrupted programming tasks (Parnin, Rugaber) <http://www.cc.gatech.edu/~vector/papers/sqj.pdf>

Subvocalization — Toward Hearing the Inner Thoughts of Developers (Parnin) <http://www.cc.gatech.edu/~vector/papers/emg.pdf>

Task-evoked pupillary response to mental workload in HCI (Iqbal) <http://dx.doi.org/10.1145/985921.986094>

memory. That is, experts are able to exploit abstractions in conceptual memory, whereas novices must hold primitive representations in attentive memory.

Sketchlet (alpha) is a software tool designed to help a programmer form and prime concepts by supporting abstraction and reviewing concepts that need to be refreshed. You can try it for yourself at sketchlet.sourceforge.net. 

Chris Parnin is a PhD candidate at Georgia Tech studying software engineering from empirical, HCI, and cognitive neuroscience perspectives. He has also worked at various organizations including Microsoft Research and GTRI. Interested in participating in an experiment or have any ideas? Email him at chris.parnin@gatech.edu.



By expanding the programming environment to tablets, sketchlets on a "Code Pad" can provide extra mental space to build and remember concepts about code.

THE DOWNWARD SLOPE

WHY ARTISTS' SALARIES ARE DECLINING, AND WHAT WE SHOULD DO ABOUT IT

The annual Game Developer Salary Survey always makes interesting reading. As of last year, the average annual salary for game artists was about \$75,000. That's a pretty respectable average—it compares favorably with the average salary for teachers in the U.S. (around \$55,000). It's also notably higher than the average given by the Bureau of Labor Statistics for "multimedia animators"—which includes film and games but also web, advertising, and other kinds of computer animations, and is quoted as \$58,000. So, it's hardly a terrible number.

But averages are always suspect. There's no such thing as an "average" artist—these numbers include both successful veterans with fat profit-sharing checks, and

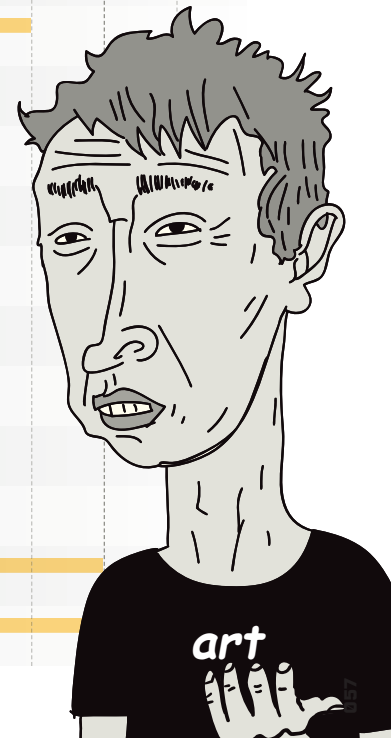
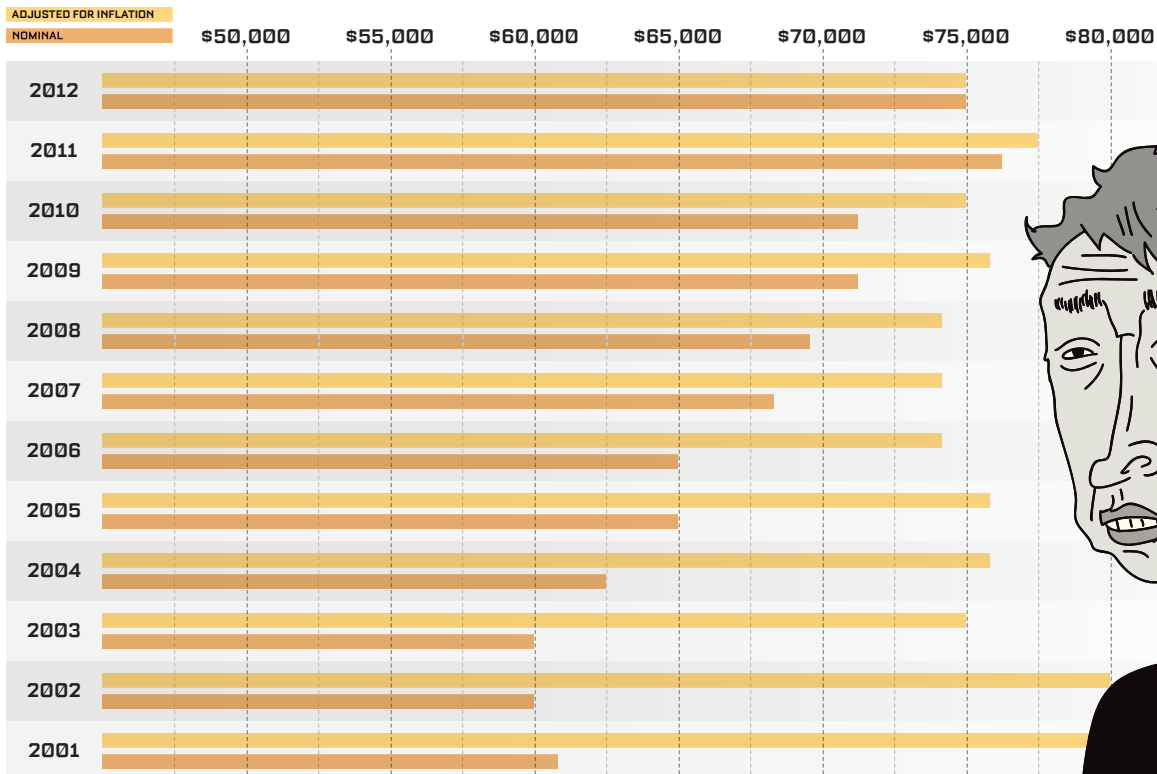
struggling indies living off of Kickstarter and Visa cards. However, there is one thing you really can see by looking at those averages: Overall, our wage picture is not improving. Back in 2001, the average artist salary was close to \$61,000. In the last decade the average has increased by 16% or so, and when you adjust for inflation, the comparison isn't so rosy: \$61,000 in 2001 dollars is around \$79,000 today. In other words, the average salary has actually *slipped* by almost 10% in real terms from where it was a decade ago.

It's tempting to blame the decline on the recession, but the trend is actually consistent across the whole decade. The biggest dip doesn't appear in 2009 or 2010 (as you might expect if the recession were to blame); it's actually in 2003. Moreover, that trend line

goes back even further. Most veterans of the premillennial days (your humble columnist included) can tell you that the market for CG artists was a *lot* hotter back in the closing years of the 20th century. I started in games in 1995, and my starting salary back in those distant days works out to just over \$77,000 in today's dollars. The *Game Developer* survey puts today's starting salary's average at around \$45,000. That's a decline of over 45% in real terms over the last couple of decades.

OUCH. Now, you might think that cushy starting salary of mine was a recognition of my sheer artistic genius... or perhaps I might like you to think that. The truth, however, is that the going rate in those heady days was a lot heftier than it is today. For

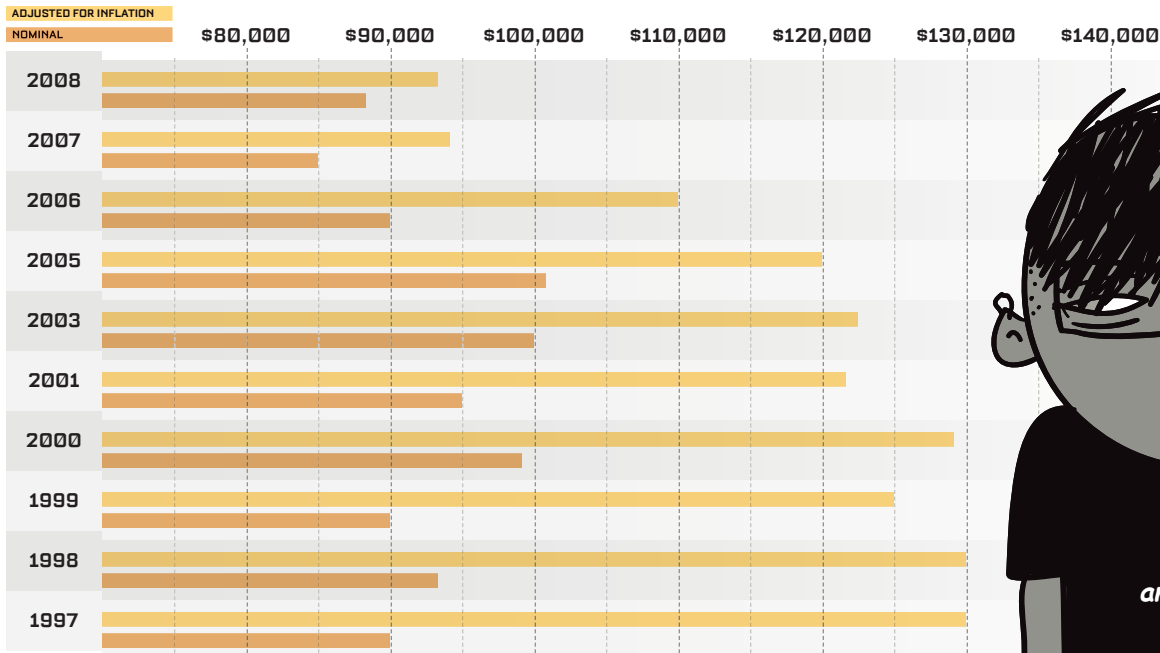
Artist salaries have been increasing. However, adjusted for inflation, they're actually falling.



comparison's sake, the Animators Guild (the Hollywood union that covers CG artists) started doing their own salary surveys in 1997, and they report an average weekly rate that works out to a whopping \$96,000 for that year, equivalent to \$137,000 today—just about twice today's average for games. (Of course, before you book your tickets to L.A., remember that film work comes with a high cost of living, union dues, and frequent periods without work.) The really interesting (or rather, really depressing) point here is that the latest number in their data, for 2008, works out to about \$93,000 in today's dollars. Sure, it's still higher than our average, but it's a drop of more than one-third in a decade. Double ouch.

As an individual artist, when you're trying to make sense of the business you're in and build a career, it's hard to get a good view of the macro trends that are shaping your economic life. We see the details: We remember our struggles and our triumphs, the successful gambles and the jobs that looked like sure things but turned out to be disasters. We remember companies that paid stupid money and the shady chiselers who bought Porsches while laying off our friends. What we don't always see is the way that larger shifts in the business are driving our futures. Averages, surveys, and the like help us make sense of all those hidden forces so we can plan more effectively for

Hollywood CG animator salaries, as reported by the Animator's Guild, have fallen by nearly a third over the last 15 years when you take inflation into account.



ourselves and our careers. In this case, the averages tell us a pretty scary tale.

The blog entry on the Animator's Guild web site that posted the Hollywood CG salary numbers ended with this laconic observation from Steve Hulett, the business representative of the animator's union:

"Soak them in and draw your conclusions. [The one I draw is: The laws of supply and demand have weight and meaning.]"

THIS IS, SAD TO SAY, A PROFOUND MEDITATION ON OUR SITUATION.

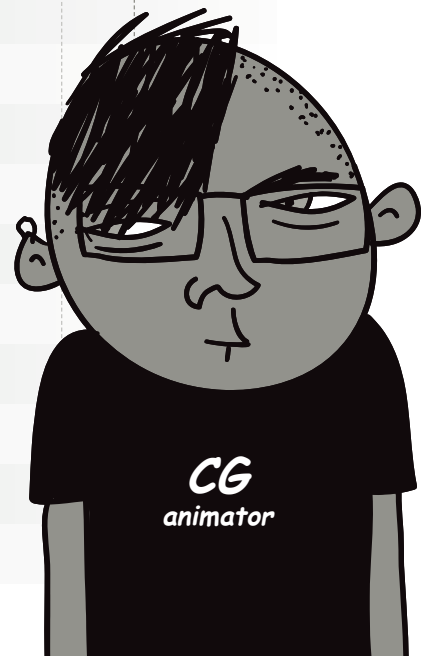
The culture and outlook of the game business in general, and of game artists in particular, formed in a unique historical moment. In the 1990s, the demand for people who could work with the clunky software and arcane processes of 3D modeling and animation vastly outstripped the supply. Computer graphics had only just emerged from academic research into the entertainment realm; although cult favorite *TRON* debuted in 1982, computer-generated imagery really captured the public imagination with *Terminator 2* (1992), *Jurassic Park* (1993), and of course *Toy Story* (1995).

Back then, there were few schools that taught the tools of the trade—and even when the schools could find the money for exotic workstations and pricey software, there were few experienced users to teach the courses. I recall, with particular secondhand embarrassment, an "Advanced

Computer Graphics Workshop" at the Rhode Island School of Design circa 1992, which consisted mostly of watching the professor rifling nervously through a thick Wavefront manual. Outside the schools, hobbyists and enthusiasts were hard-pressed to find tools that would let them learn on their own: a fully tricked-out seat of Alias PowerAnimator (the predecessor to Maya) ran in the same price range as a brand-new Cadillac, and it required a Unix workstation that cost about the same as a new Mustang convertible.

Unsurprisingly, with few opportunities for formal education or informal self-teaching, computer artists were a rare breed. Anybody who could master the esoterica of running an SGI workstation or had the patience to wrestle the beastly programs of the era could find a job with companies who were grappling with the new and unfamiliar technology. The meat markets of the CG business—particularly SIGGRAPH, where Hollywood and big TV production houses did the bulk of their recruiting—were as competitive as the NBA draft. Anybody who could produce a reasonable reel was aggressively pursued with swanky hospitality suites, lavish parties, and signing bonuses. The only dark lining to this silvery cloud was that many artists had to labor (not always unjustly) under the perception that they were merely software jockeys and not "real artists." All in all, though, it was a pretty good time to be in CG.

Fast-forward a decade or two, though, and the picture is a lot less rosy. Wages, as we've said, are slowly falling relative to

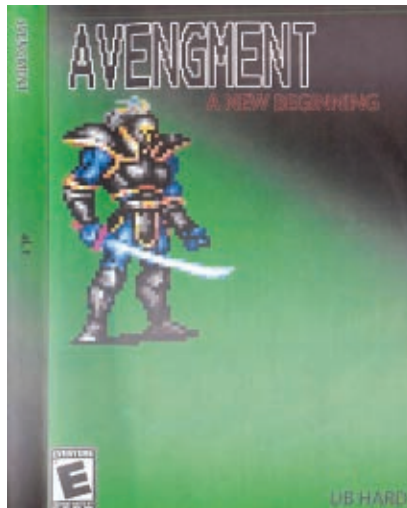


inflation. It's harder to imagine yourself becoming an overnight millionaire as artist #356 working on the seventh iteration of a triple-A behemoth (though, to be fair there are a lot of mobile and casual devs imagining how it will feel to be the next ANGRY BIRDS or LEAGUE OF LEGENDS). Above all, there's no shortage of talented computer artists any more—just ask any former colleague who's had to hit the job market recently.

It used to be hard to find artists because there just weren't that many of us. Not only was the field itself new, it was quite hard to get into. Old-school game artists were largely self-taught, scrounging for information wherever it could be found; they operated on trial-and-error and learned on the job. Today, though, there's a huge (and growing) educational industry that offers coursework, access to the right software, and of course official credentials—which are rapidly becoming a necessity for young artists looking to break into the business. Some schools provide a solid foundation in traditional art and the most up-to-date tech, while others are little more than diploma mills making bank off of cheap student loans and the eagerness of kids who have grown up on games to get into a business they idolize. It's easy for game industry vets to be cynical about the art-education business, particularly given the cost—at \$60,000 for a two-year program, a degree in computer animation from a mediocre vocational school isn't a huge bargain compared to the luxury-car level cost of setting up your own workstation back in the 1990s.

Like them or not, though, the schools have changed the landscape: There are a *lot* of young artists out there with reasonable reels and decent technical skills. That makes it easier for employers to be picky and, if you're a game consumer, it means better-looking games. Unfortunately for us, it also keeps wages down. That's the law of supply and demand for you.

Traditionally, we have not been a degree-heavy business; a lot of us are here because we love games and taught ourselves how to make them. The wannabe factor gives game art enthusiasm and drive—and it also expands the supply of would-be artists. Games have become infinitely more accessible to hobbyists, enthusiasts, and indies. It's easy to find after-school programs and camps that teach the rudiments of 3D and game creation to middle schoolers. Ambitious and motivated kids don't even need to go to school—the internet offers an unimaginable bounty of tutorials, discussion sites, and online classes for every piece of software and genre of art. There are plenty of free, high-quality tools to learn with, from Source Filmmaker to Sketchup to the Unreal SDK.



With online information, better tools, and the rapid growth of game education, it's easier than ever for wannabes to get into the industry. This is great for our creativity and energy... but not so great for our salaries.

Today's vibrant and sophisticated hobbyist communities produce amazing artists and innovative work. Unfortunately this, too, raises the bar for us working schlubs who have to pay the mortgage. Most of us professionals have had to do some soul-searching when we compare our portfolios—done under deadline pressure, technical constraints, and sometimes under debatable art direction—with the highly polished work of enthusiasts who can spend six months perfecting a single sculpt. There is a lot more high-quality CG artwork around now than there was a decade ago—and once again, the iron law of supply and demand keeps prices down.

Depressed yet? We haven't even mentioned automation, outsourcing, or the shift from big console games to smaller, less content-intensive mobile and social titles. Oy vey.

The fact is, the Wild-West boom days are behind us. The Bureau of Labor Statistics is predicting that the employment in computer graphics and animation will grow by about 8% over the next decade—compared to 16% overall. We're not a growth field anymore. The game industry is changing quickly, as smaller studios and smaller projects are replacing the triple-A behemoths of the last decade. Even so, losing the bull market mentality that is built into the game industry's DNA is difficult. Optimism and ambition are such a basic part of what we do that it's emotionally quite hard to face up to the fact that our cozy little niche is a lot more crowded—and hence a lot less lucrative—than it used to be.

Now, this does not mean your job is going to disappear overnight, or that there is no future for the products of all those new computer art degree programs. It does mean prospective students ought to think long and hard about the wisdom of spending \$60,000 or more on a degree! Stock watchers and industry analysts think that the business side of games will start picking up again over the next few years. There will be jobs and opportunities in the future (though a lot of the growth is going to be in the new frontiers of mobile and web gaming, where salaries are lower and triple-A cred counts for less). However, even an upturn in game sales isn't going to reset the clock and push wages back to the level they were in the boom days.

The upshot is that we have to be realistic about our prospects in an increasingly crowded field. Trusting in luck and a vague sense that games are "big business" isn't a viable strategy under today's conditions. This can be especially hard for vets whose emotional ties to the business were forged in headier times. The artists of the next decade will have to be constantly upgrading their skills, keeping up with changing tech and trends in games, and thinking hard about how to market themselves. They'll need to have a clear-eyed approach to things like crunch time; as the rewards of working on a game become smaller for many of us, our willingness to sacrifice family and health for the cause diminishes apace. Many will seek the higher risks and higher rewards of indie and mobile games. Some will throw in the towel and go on to careers that pay better or demand less.

It will be interesting to see how much old-school game culture survives in the less-expansive times ahead. We can hope that the optimism, humor, and enthusiasm that typify game artists won't succumb to the gray fog of lowered expectations. Casual, mobile, and social games still retain a lot of the shoot-from-the-hip gusto of earlier times. Let's hope they keep game culture from getting stale. Even triple-A dinosaurs need to remember that it's pretty damn awesome to spend your days painting scales on dragons and adding fins to spaceships compared to, say, preparing tax forms for a living. And never forget: There's no such thing as an average artist or an average game. We didn't get into this crazy art form to do average work. Kick enough ass, and things will work out. pp

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

RESPECTING THE PLAYER'S WALLET

FREE-TO-PLAY AS A DESIGN OPPORTUNITY

060

It's been a long time since there has been a complete revolution in video game design, but we are in the midst of one now. Free-to-play gameplay and microtransactions used to be limited to indie games and Korean massively multiplayer titles; now, it's broken into the mainstream in a big way, and has increasingly become the way players expect to play their games.

The success of LEAGUE OF LEGENDS is largely credited for this transition; at launch this pioneer allowed players to play a substantial portion of the game without paying a dime; as of July 2012, it had 12 million unique players per day, with over 32 million registrations, putting Riot Games on the map. By comparison, FIFA 13 sold 12 million copies worldwide total in 2012, and WORLD OF WARCRAFT peaked at about the same.

The rest of the industry noticed, and began moving quickly to a f2p model. New titles like TRIBES: ASCEND and the upcoming NEVERWINTER NIGHTS were

built with this billing model in mind. Meanwhile, older titles like LORD OF THE RINGS ONLINE, TEAM FORTRESS 2, and (my employer's own) STAR WARS: THE OLD REPUBLIC have all made quick adjustments to this billing model. All of these games enjoyed strong sales and market presence in their original incarnation, but all reported significant increases in both revenue as well as player populations from the change in billing model.

THE CHANGING MARKET The market conversion has not happened without some bumps in the road, however. Many gamers dislike or resent the microtransaction trend. They tend to be older players, who grew up buying their games for a \$60 price point, and like never being asked for another dollar again. This includes a substantial portion of the game developer community, who feel like the industry is nickel-and-diming customers to death, and then shaking their corpses for loose change.

Kids today grew up in a different world, finding music on YouTube and Spotify, then purchasing songs individually on iTunes. They buy movies on demand from a number of sources if their parents don't subscribe to HBO. They grew up playing CLUB PENGUIN and MAPLE STORY, and now play triple-A games like LEAGUE OF LEGENDS, paying as much money as they feel they can afford, which very frequently is none at all. The next generation of gamers grew up in a world where entertainment was tried for free and bought à la carte.

It's not impossible to see why. While older gamers see free-to-play microtransaction models as an attempt to fleece the customer, the reality is quite different. The vast majority of those who play these games do, in fact, opt to never spend a dime, and this means that these customers never spend \$60 on a game that they dislike. The free-to-play model becomes the ultimate free trial, and it puts a huge onus on the designer to create a quality, polished gameplay experience that the player quickly finds fun and engaging. When looked at through this lens, it is very easy to see the free-to-play model as far kinder to the customer base than was the old premium box model that we all grew up with.

AN EVOLVING DESIGN PERSPECTIVE

Needless to say, f2p requires designers to make significant changes to the way they approach their craft. Some of these are subtle but crucial. As an example, classic MMO design is, first and foremost, designed to encourage subscriptions at all costs. Designers know that once a player cancels their credit card in a game, it can be very difficult to get that customer to re-engage.

By contrast, it's not so distressing if players of a free-to-play game bounce out of the game for a little while. Designers of true free-to-play games no longer care if you quit in April and May if they can get you to bounce back in June to play for a



LEAGUE OF LEGENDS.

while and buy some stuff. Since no credit card information needs to be entered, players are much more likely to stop by when nostalgia for the game kicks in. In my opinion, this model much more accurately maps to real life than the old subscription model—the world is full of competing interests, be it the new hottest competitor to come down the pike, or real-life competition like television, school, or romance. The free-to-play model is less desperate to maintain a player's interest and subscription through all obstacles at all costs, but can instead focus on high-profile events designed to recapture the player's attention.

THE VALUE OF FREE PLAYERS

Designers must also consider the widely disparate ratio of spenders vs. nonspenders. The exact ratio varies wildly from game to game, but can be a huge swing. Facebook games seem to have among the widest disparities, with developers reporting that frequently fewer than 2% of the population pay any money at all. The other 98%, and all of the costs they incur, are effectively subsidized by that small sliver of the population.

It is easy to think of that 98% as shiftless moochers, but in most cases, the game is served well by having a large free-to-play population. Players are content for other players. In MMOs, they make your towns more full and social, and fill your game with more potential party members for your dungeons and player-vs.-player battle scenarios. In *WORLD OF TANKS*, free players fill the world with fodder for your paying customers to destroy.

And this goes beyond the social value that having a large paying population offers. Even if only a fraction are paying for your game, having a free population of a couple million means that you have a couple million people potentially evangelizing the game to their friends and family. These big numbers are easier to market to boot.

MONETIZING YOUR BIG SPENDERS

However, since potentially a small sliver of the population is actually monetizing the game, in many genres, designers need to re-evaluate exactly how money is spent in the game, to allow players to spend what they want to spend.

If you look at most hobbies, they allow spending to scale to level of interest. It is possible to knit on the cheap, picking up only some needles and a ball of yarn at Hobby Lobby. Hardcore knitters, on the other hand, may spend thousands of dollars amassing huge yarn collections, and even fly across the country to go to sock-knitting conventions (yes, they exist). This level of optional spend is found in most major hobbies: woodworking, building model trains,

playing music, golf—you name it. And while shops that cater to these hobbies are more than eager to help newcomers get off the ground with their new hobby, most of them live or die by their regulars, who are more than willing to spend their disposable income on the hobby that gives them so much joy.

Looked at in this light, the classic game model doesn't make sense. No matter how much you loved the original *STARCRAFT*, your spending in the game was pretty much limited to the original box product and the expansion. A hardcore player's spend was going to be in the same order of magnitude as the new player's, while the new player's entry fee was high enough to be a disincentive for many players to try it out. The rise of DLC as a revenue source has attempted to better capitalize on these devoted fans, but since they tend to still be content driven, the hardcore hit a hard ceiling of how much they can spend—even if they want to spend more (and most people like spending money on hobbies they love).

Games that go free-to-play need to better capitalize on their devoted fans. *Magic: The Gathering* has what developers call a "repeatable spend"—players buy random boosters to make their decks better and complete their collections. This is a highly scalable activity—Wizards of the Coast spends a lot of time providing cheap, entry-level decks and creating drafts and "pauper" leagues designed to engage low-level spenders, while high-end tournament decks can have aftermarket values of \$400-600. The top customers are the financial lifeblood of the game, and Wizards goes to great lengths to elevate these customers and decks, but in general, players of any skill and spending level can find a satisfying game experience.

RESPECTING THE WALLET Perhaps the most important lesson for those aspiring to enter the world of microtransactions is figuring out how to do so while still maintaining a healthy respect for the player and their wallet. It is easy to find new and exotic ways to charge your customers money, particularly now while the microtransaction model is still in its infancy in the United States. But just because you can charge a buck for something doesn't mean that you should.

The most common mistake I see in the iPhone games I download is that the games are entirely too aggressive in attempting to charge their customers. Their "free-to-play" games often ask the player to pay for more energy within five minutes of initial download. The human mind is a great pattern-matching machine, and a player will almost certainly extrapolate that pattern to the future—and what they will see is a game that is not



only doomed to be more expensive than what they want, but also one built with the philosophy of nickel-and-diming them to death. Even worse, all of this happens before the customer is

emotionally engaged with the game. It is easy and expected in these cases for the player to walk away.

By contrast, when we converted *STAR WARS: THE OLD REPUBLIC* to follow the free-to-play model last year, we made a concerted effort not to oversell our microtransactions. Players are rarely, if ever, prompted to spend money in the first 10 levels, and the player is likely to forget that doing so is even possible. Even when spending is possible, we make it clear that the entirety of the game can be experienced (albeit with limitations) without spending a dime. Our logic is simple: We think that players are more likely to invest their time and money in a game that they love, so first and foremost we want our players to fall in love with the game. We see the player's respect as something that needs to be earned before we win that dollar.

MICROTRANSACTIONS IN THE WILD WEST

The free-to-play microtransaction game model is coming fast now, and it is not inconceivable to imagine a world in the next decade where the majority of games available to users are delivered in this reality. For the time being, though, we are still in a Wild West game design reality, where every design shop is attempting different ways to earn that dollar—some of those bordering on shady and questionable.

Still, game designers need to start by acknowledging that a small number of heavy users will likely subsidize the majority of the efforts in their microtransaction-oriented game. To me, as a game designer, this is a good thing; making a game that earns the love and devotion of players is the right path. Making a great game with a solid design and a respect for your player's wallet is still the best way forward. **d**

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

THE REAL MONEY OPPORTUNITY FOR THE ODOBO GAMES DEVELOPER



Ashley Lang explains how the Odobo platform provides game developers with an incredible opportunity to join the regulated \$35bn real-money online gambling (RMG) industry where ARPUs can exceed \$100 per player.

Ashley Lang, founder and CEO of Odobo, brings more than 12 years' experience to the online gambling industry and has held senior executive level and directorship positions with two of the largest European online gambling operators. He is regarded as an expert in conversion optimization, player acquisition, affiliate marketing and has been responsible for programs generating millions of customers for regulated gambling operators. Lang has a strong entrepreneurial track record as co-founder of Green Room Media, and is a partner in both poker word game AlphaBet, and iOS game developer, Granville Games.

Odobo launched in late 2012 as the new HTML5 game development platform and marketplace for the regulated gambling industry. Just as the Apple, Android and Facebook developer program models were instrumental in bringing creativity and innovation from new developers - and ultimately driving engagement and sales for the host and device manufacturers - Odobo aims to do the same in the regulated gambling industry. By enabling a wider group of top-tier game developers to more easily bring their game concepts to market in the highly lucrative RMG industry, Odobo aims to fuel innovation and creativity in game production that will delight both existing and new players attracted by the content to the market. In the process, games developers who have previously faced high barriers to entry can enjoy significantly higher revenues than in other forms of gaming (social, casual, console).

The regulated online gambling market is estimated at \$35bn annually today and is forecast to exceed \$40bn by 2015. In the US several states have introduced legislation to allow online gambling. The U.S. market for online gambling alone may reach \$7.4 billion a year by 2017, according to Manchester, U.K.-based researcher H2 Gaming Capital. This is too big an opportunity for game developers to ignore.

Odobo allows games developers to distribute content through a network of licensed established operators around the globe in regulated markets. Qualifying developers can produce games for real-money play without needing to become the licensed operator or having to develop much of the technology required by relying on Odobo's multi-million dollar platform and games development kit (GDK).



Average player values in regulated gambling are over 15x higher than social and casual gaming. The ARPUs can exceed \$100 per player in RMG as opposed to approximately €1.50 blended average income in the App Store.

Producing content for the RMG industry was previously only available to a select few game development studios, because platform fragmentation, poor documentation of development protocols, tightly held specific knowledge of requirements and low transparency on distribution and revenue generation, provided too many hurdles for most developers to overcome.

Odobo has rewritten the rules on the way content is produced, licensed, distributed and monetised for the regulated gambling industry. The Odobo Developers Program provides a transparent business opportunity for the game developer to leverage the core technologies of the Odobo Games Development Kit (GDK) in the production of their game, and the simplified downstream distribution and licensing model of the Odobo Marketplace.

LUCRATIVE NEW REVENUE STREAMS: US AND INTERNATIONAL MARKETS:

In the Odobo model, games developers earn revenue from two models. When gaming operators license games from the Odobo Marketplace they pay the games developer a royalty based upon the gaming revenues generated from their customer's play of the game. Odobo tracks and collects these royalties for participating developers. In addition, games developers can take an active role in marketing their games and driving new players to play their games with participating operators in regulated markets. When the games developer is also the referrer of the new player, the game developer is eligible for an affiliate commissions based upon the lifetime value of the player (not just on their games – affiliate commissions are paid on the lifetime value of the referred player across all games and all gambling products). The combination of game royalties and affiliate commissions can result in ARPUs in excess of \$100 per player to the developer. This new revenue stream can be reinvested in marketing the developer's games across all monetization channels (social, casual, and real-money) "The incoming tide raises all ships."

Regulated online gambling is finally becoming a reality in North America as states such as Nevada, New Jersey and Delaware begin to license and regulate the activity and the potential once other states come online is massive. However, the opportunity outside the US is already there in countries such as the UK, several other European countries, Canada and elsewhere. There will be a big advantage for US game developers who cut their teeth now in open markets outside the US, while waiting for the US to continue on its path of state-by-state regulation.

With the promotion of a standard platform for the development and publication of games content, players across the US, and internationally, will be able to play the same great games despite being serviced by different state licensed operators with point-of consumption regulatory requirements. This is the opportunity Odobo represents for game developers and gaming operators alike.

THE ODOBO GAMES DEVELOPMENT KIT (GDK) AND HTML5:

Core 'commodity' technologies, which are not specific to any game concept, yet are required in the provision of every game, can now be standardized and provided to the developer at the platform level. These include a random number generator, network communications handling, localisation handling (currency and language), persistence (state handling), authentication and player account communications.



ADVERTISEMENT

It is extremely expensive and inefficient for game developers to need to invest in core distribution technologies when they are not specific to their game concept. Odobo provides these components at the platform level.

The 'tectonic shift' to mobile gaming as players migrate from PC and console game-play to tablets and mobile devices, presents a challenge for the regulated gambling industry as native app distribution of gambling-enabled applications is restricted to few markets and significant investment is spent reaching players on tablets and mobiles, only to offer them Flash games which are unsupported on these modern devices. A key part of the answer lies in HTML5 and Odobo is showing that HTML5 enables developers to produce content that rivals the quality of Flash or native applications, whilst being unrestricted by native app store distribution (and fees) and allows play across all HTML5 supported devices. Content produced using the Odobo GDK looks and plays brilliantly on a large desktop full-screen cinema display, tablets and mobile devices and is available for distribution to players in all licensed operator supported markets.

THE ODOBO MARKETPLACE AND ODOBO PLAY:

Content produced using the Odobo GDK is offered to licensed online casino operators via the Odobo B2B Marketplace. These licensed operators in turn publish the games to their vast customer bases of qualified players and reward the games developer with royalties based upon the gaming revenue generated.

The soon to launch Odobo Play is the consumer-facing B2C side of the marketplace, which will showcase the greatest games, the developers that produced them, and the licensed and regulated online casinos that support their play in a discovery-friendly format for players worldwide. This will also provide an optimized marketing channel designed by Odobo for developers and brand owners to promote their games to existing and new audiences and to participate in the lucrative 'new player' affiliate commissions on offer.

ODOBO SUPPORT:

The platform allows developers to manage their games by territory, select royalty rates and to see detailed analytics on player activity and detailed statements on their financial performance. This is designed to give the developers visibility into the player activity that drives their revenues and to help developers to make better products.

An operator who has integrated Odobo can select games and with just a couple of clicks they can license the developers' game from the marketplace and install it in their Odobo integrated and regulated environment, with the confidence that it is built on proven and compliant processes, before launching the game to their players..

A single license agreement covering all content licensed from the Odobo Marketplace, simplifies contractual relationships with developers. The Odobo GDK allows the game developer to focus more on the design, concept and user experience of the game allowing greater innovation.

The combination allows game developers to focus on what they are best at - creating great new games. To make the debut into RMG world easier, Odobo gives the games developer a dedicated account representative to provide direct assistance and support throughout.

RAISE YOUR GAME:

The Odobo model brings together the creative global games development community with the huge revenue opportunity presented by the online regulated gambling industry and to the benefit of games developers, players and operators alike.



odobo
RAISE YOUR GAME
www.odobo.com

STUCK ON REPEAT

A HARD LOOK AT THE USE OF ICONIC SOUNDS IN GAMES

MS. PAC-MAN.

A cavalier attitude surrounds most game development; people treat each game as a special case instead of relying on what has historically “worked” when it comes to best practices. This spirit, coupled with each generation’s limitations, has allowed for constant reinvention during the massive upheaval of creating something new within limitations. The emergence of varied sound-as-representation-of-reality has swiftly replaced most iconic underpinnings of earlier game audio to the point that sound-as-communication is much more subtle. But when it comes to repetitive sounds that speak to the player, are we saying the right things?

STOP ME IF YOU THINK YOU’VE HEARD THIS ONE BEFORE

It’s likely that most of the pop-culture reputation that game audio has achieved hinges on the iconic sounds created during the birth of arcades and home consoles. With cabinets and televisions cranking out chip sounds and 8-bit sonorities, those of us who grew up with it internalized the cues that were being communicated to us in the simple language of sound synthesis. We all remember the sound of MS. PAC-MAN gobbling dots, the multiple explosions exposing monophonic playback limitations, or Q-BERT’S signature synthesized swearing. If you were there back then, these sounds speak to you even today. It was clear that the sound of SUPER MARIO BROS. had reached icon status when I heard it at a basketball game; a free throw never meant as much to me until I heard it coupled with the coin-collect sound from my childhood MARIO. Those sounds don’t just serve as positive feedback; they have gone on to transcend the living rooms and bowling alleys we grew up in and continue to define our modern lives.

Sound designer Mike Niederquell created a resource chronicling the most

memorable and iconic game sounds over at TheSonicSpread.com, which hosts a wide variety of examples that go beyond just the early days of synthesis and “musical” sound design. Tellingly, one of the recurring themes throughout the list is that of frequently repeated sounds. This aspect of game sound was once a limitation imposed by a lack of resources or inability to load multiple variations of a sound into RAM. Hearing the same sound over and over had a way of reinforcing the action it represented and helped to build an association for the player. And that association became so strong that players would tune into these sounds and use them to augment their gameplay.

SOMETHING PULLS ME RIGHT BACK

How many times do you hear the coin sound during a level in SUPER MARIO BROS.? Now think about how many times you hear a footstep, gust of wind, or bullet impact. Chances are good that you’ve heard these sounds just as frequently (if not more frequently), but you wouldn’t be able to match them to a specific game. Real life is infinitely more varied than any current simulation. If reality is part of the design aesthetic for the game, it makes sense to honor that as closely as possible with sound—but that doesn’t mean you can’t still imbue your sound set with iconic aspects that can help “brand” it while still allowing for slight modifications across different versions. Finding the qualities that help differentiate a sound speaks to the core of the sound-design process, but for designers, finding the “voice” of a footstep is secondary to making sure it blends seamlessly (and nonintrusively) into the environment. If we look at the footstep types for differently sized characters, we soon find that not all footsteps are created equal. Whether it’s a lower pitch, a layered impact, or extra element that helps



communicate to the player these differences, the outcome is a clearer indication of the sound’s intention.

As consoles have grown in power through the years, we sound designers have gained the ability to move toward a more realistic representation of sound through variation. Being able to draw upon multiple sounds and randomize volume, pitch, and frequency filtering, for actions that may have a real-life equivalent, lets us more deeply immerse the player in our game by better mirroring our perception of sound in reality. Coupled with this is the desire to convince the player that the worlds we create are real. However, we still need to train the player with audio cues; finding the right iconic heart for a varied sound will enrich the players’ experience and communicate your intention.

MY HEART’S SKIPPING, SKIPPING

We are swiftly approaching an age where audio will be freed from the current file size and quality restrictions, much to the

delight of game audiophiles everywhere. With this increase in space and quality, players will expect more diversity in the sounds we use to represent the worlds we create. As sound designers, we’ll have to balance the use of sound as a mirror for reality, and the use of sound as a tool to get the player to pay attention to something specific. Looking at historical examples of iconic sounds in games, and otherwise, is a good template for what has captured the ear of our culture. In the never-ending quest to leave the player with a lasting impression of their experience, we could do worse than to create memorable, iconic sounds that convey character, while still being varied enough to immerse the player in the world. af

“Communication is never simple, especially when it’s you that’s on the receiving end.”—Little Boots

Damian Kastbauer is a wandering minstrel of game audio, traipsing across the land at LostChocolateLab.com and on Twitter: [@lostlab](https://twitter.com/lostlab).

THE POWER OF RELATIVITY

HOW TO AVOID COMPARISON-PRICING PITFALLS

One of the most frustrating things a game developer will ever hear is “that [PERCEIVED GENRE] game isn’t worth [PRICE]—I can get [OTHER GAME] for [LOWER PRICE].”

It’s frustrating for a whole bunch of reasons. Your game might not be very similar to the games it’s being compared to, or might offer more content or replayability. Heck, you might simply think your game is better and deserves to get a higher price. But it doesn’t matter. The comparisons are being made and now you’re getting two-star reviews calling your game good but your company greedy.

If that sounds familiar, congratulations. You’re part of the very large and growing club of developers who underestimated the power of relativity. No, not $E=mc^2$. I’m talking about the fundamental human tendency to compare everything in our lives to something else we’re familiar with. An organic apple seems ludicrously overpriced at \$1.99 because conventional apples sell for 79 cents, but that same apple would seem cheap if your grocery store carried only the organic variety and if organic mangos were displayed nearby for \$5.99 each. It’s all relative.

Psychologists tell us we can increase our happiness by owning the nicest house in our neighborhood. This is the exact opposite of what many financial advisors will tell you, but the psychologists are right—if you own the nicest house in your neighborhood, you’ll rarely feel jealous of your neighbors or dissatisfied with your lot in life (unless, of course, you spend a bunch of time driving around wealthier neighborhoods).

A huge part of traditional retail marketing is dedicated to countering and exploiting this fundamental human tendency. Retailers very carefully pick what products they sell beside each other. And they attempt to reel you into their stores with advertisements showing a product you are likely to be familiar with for a price you’re likely to perceive as cheap, while then upselling you in person on products (with high margins) that you’re less familiar with. (Who the heck knows what an 18k gold necklace in that shape, containing that stone, by that designer should actually be worth?)

TRIPLE TOWN: THE PUZZLE GAME THAT WISHED IT WASN’T So, to bring this back to game development: You’re making a game. Odds are, whether you like it or not, it’s going to be compared to some other game. And when that comparison is made, you will live with the consequences. In the mobile world, more often than not, the consequences are rather predictable: You’re going to have to sell your game for somewhere between 99 cents and \$2.99, or risk it being perceived as too expensive.

You can try to fight this the way we did in TRIPLE TOWN’s mobile edition. We believed that we had created something special, but



we knew that nobody would pay what we believed was “fair” for unlimited turns. (Tricky word, “fair.” Your definition probably isn’t the same as mine.) So we gave people a limited number of free turns in TRIPLE TOWN every day, with the hope that eventually the quality of the game would win over even the most jaded 99-cent shopper. It kinda worked. We started out with unlimited turns at \$6.99 and consumers absolutely lost their minds with rage. How dare we charge so much?! So we ratcheted that back to \$3.99 and it seemed to work. We still get the occasional complaint, but in general we convert free players to paying users at a higher rate than most casual games achieve, and we do it at a price we feel comfortable with.

With all that said, what I want you to understand is that **we did it wrong in TRIPLE TOWN**. Yes, even though we have a higher conversion rate than most traditional puzzle games.

We did it wrong precisely because we allowed ourselves to be compared to “most traditional puzzle games”; in other words, games that consumers are no longer willing to pay more than 99 cents for, with rare exception. It’s hard to be enthusiastic about spending the time to create something original and beautiful in a market that values it so little.

REALM OF THE MAD GOD: THE ANTI-REFERENCE Now take a moment and compare our experience with TRIPLE TOWN to our experience with REALM OF THE MAD GOD. For starters, ROTMG was generally resistant to comparison. What is an 8-bit bullet hell shooter MMO featuring permadeath and 80-man raids (in Flash!) similar to, exactly? More importantly, the things we sold within ROTMG aren’t easily compared to other products. What, exactly, is a “character slot” in this context worth? What’s more inventory space worth? What’s a health potion worth?

Well, I’ll tell you: **It’s worth what you’re willing to pay for it.** No more, no less. But at least that number is derived from your intrinsic interest in playing ROTMG (relative to any other game) and

your personal opinion of a given item's likely value to you. It is **not** derived from the arbitrary fact that *ANGRY BIRDS* sells for 99 cents, and therefore your game should, too. What a concept—being paid in accordance with how much people like your game!

In other words, free-to-play games—especially original free-to-play games that defy comparison—have an opportunity to lift the goods that you are selling off the “supermarket shelf” and into a context that does not encourage such crude comparison shopping.

To be clear, this doesn't mean “make a F2P game and you can charge whatever you want for in-app purchases.” The perfect example of this is, again, *TRIPLE TOWN*. Unlimited turns are an in-app purchase that is nevertheless perceived by consumers as an upgrade to the “paid version.” Consequently, the game remains on the supermarket shelf and only manages to achieve a slightly higher sale price thanks to how extremely engaging and replayable it is. If you really want to break free of the supermarket, your IAP can't simply be a thinly veiled upsell to the full version of the game. It has to be different.

RELATIVITY CAN HELP, TOO Breaking free of unhelpful comparisons is just one side of this coin. The other side, of course, is to leverage the **helpful** comparisons. For example, in one of our recent games, *HIGHGROUNDS*, we generate essentially all of our revenue by selling booster packs of nonconsumable playable units. *HIGHGROUNDS* has been described as “*Magic: The Gathering* without cards,” and even though there are substantial differences between the games, we like that description so much that we've wholly embraced it.

The reason this comparison is helpful to us is that *MtG* has a very well understood revenue model. There are millions of people out there who appreciate the fundamental promise of the game: You can spend very little (or in our case, zero) and still have fun and be competitive, or you can spend a lot to really flesh out your library and enjoy a much greater diversity of strategic options. You don't “pay to win,” you “pay to play differently.” Players really seem to respond well to this.

I should emphasize that our avoidance of “pay to win” is only part of the equation for *Spry Fox*. When the inevitable angry player complains about our business model in our forums, our fans often spring to our defense by invoking the long and respected history of games like *MtG* that have used booster packs as their revenue system. Even some trolls will grudgingly admit that they “get it” even if they “don't like it.” That's the benefit of comparing ourselves to a positive reference point as opposed to a negative one.

MARKETS MATTER It's worth noting that even if you're planning to develop a traditional, non-F2P game, in a traditional genre, with all the traditional trappings, you can still make decisions that impact how your game will be perceived and compared. Take a game like *FASTER THAN LIGHT (FTL)*—one of my favorite indie releases in the past few years.

If *FTL* had first launched on mobile phones at the exact same quality level and with an appropriate UI, it would have been unlikely to sustain a higher-than-\$2.99 price point, and even that is questionable for an indie game of this scope nowadays. And once *FTL* had launched on mobile phones, it may have been branded as a mobile phone game and therefore somehow not worthy of a higher price without substantial expansion or improvement. Instead, the game launched elsewhere, and was sold for prices as high as \$10 with regular discounts to \$5 on platforms like Steam.

If you ignore relativity, it doesn't make any sense. Why should a game be worth several times more money just because it launches on Steam before it launches on mobile phones? Of course, you can't ignore relativity, because your prospective customers certainly won't.

IF ONLY I WAS SELLING COFFEE It's easy to feel bitter when someone holding a \$4 latte says that your \$2 game is overpriced. Unfortunately, bitterness won't help you sell games, and there's something to be learned from the fact that Starbucks can sell coffee for \$4, or that Evian can sell bottled water for \$3. These companies have marketing machines that spend millions of dollars convincing us to disassociate their products from cheaper and/or less-refined substitutes.

Their marketers are pushing a message: “You can justify spending a fortune on *this* water because we shipped it to you from a mountain spring in Switzerland.” And “You can pay \$4 for *this* cup of coffee because not only is it tastier, but you will enjoy the experience of drinking it in our comfortable and trendy café.” In other words, 7-Eleven sells you coffee; Starbucks sells you coffee++ and strongly suggests that you cannot compare the two.

WRAPPING UP The bottom line is this: You are not purely at the mercy of the market. Every choice you make, from your game's genre, to your game's business model, to your game's launch platform, will have an impact on how your game is perceived and **to what your game is compared.**

You are the first person to describe your game to the public; you decide what, if anything, you'll liken it to. You control

the context of your in-game purchases, if there are any in your game. Think hard about what comparisons those contexts will invoke, and how you might make them more favorable. And of course, you are the one deciding how original your game will be, in general; you don't have to make something with an obvious competitor if you don't want to.

Everything is relative. We simply can't escape that. But relative to *what...* now, that bit is up to you. **b**

David Edery is the CEO of Spry Fox and has worked on games such as REALM OF THE MAD GOD, STEAMBIRDS, and TRIPLE TOWN. Prior to founding Spry Fox, David was the worldwide games portfolio manager for Xbox Live Arcade.





WHO FIGHTS FOR THE USER?

REQUIEM FOR INDUSTRY PIONEER KENJI ENO

On February 20, 2013, the game industry lost one of its few great iconoclasts. This was the date when Kenji Eno, known for games like *D*, *ENEMY ZERO*, and *REAL SOUND*, passed away, leaving a legacy as a creative force that would not be tamed, and which would not bow down. And in our industry, we desperately need more people like him.

WARP TO THE PAST Eno began creating run-of-the-mill action games in the NES era, but quickly became frustrated with both the style of game and the size of his company as it grew from 10 to 30 people. He quit his stable job and formed his own company, EIM, because he wanted to control his creative destiny.

It soon became evident that in order to keep his employees fed, he would have to make licensed games, which, as he told consumer publication *EGM* in his last big interview in 2008, made him so upset that he became “mentally unstable.” This was so far counter to what he

wanted to do that he couldn’t even go into his own office. He wanted to work on original things—after all, that’s why he became independent in the first place—so he closed the company and left the industry entirely for a time. He began working at an auto magazine, but the itch to return to games was too strong. Upon his return, he founded Warp, the developer for which he’s best known.

Though quitting a job or closing a company because of creative integrity is already rather rare, it’s at Warp that he made his biggest impression. When creating the game *D*, he wanted to shock players out of their complacency, and

make them think differently about games. (The shocker, in this case, was a scene involving cannibalism.) He figured the censors wouldn't allow this scene to pass, so he submitted a clean version of his gold master after the deadline, knowing that he would have to hand-deliver the final disc by plane. The clean version was approved, and while he was in flight he switched the discs to the one he really wanted to show the world, sneaking his cannibalism scene into the game under the publisher's nose. The game then went on to sell one million copies in Japan alone.

It's rash, and may seem irresponsible to some—it would most likely be the subject of a lawsuit in this day and age. But how many of us would go so far to create something we believe in? We've seen a few in the last few years, and most of them have gone completely independent, eschewing the traditional game industry entirely. In many ways, Eno was the prototypical indie developer, shouting in the face of authority.

His next big game, ENEMY ZERO, was meant to be for the PlayStation. His team had already started developing for Sony's platform, but he was upset with how Sony had short-changed his shipment of D, shipping less than a third of what they said they would. So he took his revenge at a Sony press conference. As he walked out on stage to announce his new title, the screen behind him showed a PlayStation logo... which quietly warped into a Sega Saturn logo, and he went on to announce his game for Sony's rival platform, at their own event, simply because they'd backed out of a promise. He wasn't going to take any slight lying down. How often have we all had to grin and bear similar indignities, with no way to vent our frustrations?

When ENEMY ZERO was finally released on the Saturn, he made a limited run of 20 special editions, and if you paid \$2,000 for one, he would hand-deliver the game to you. They sold out immediately, and Eno drove his truck across Japan delivering games to his customers. He had always wanted to have a direct relationship with players of his games, and here now was a way. Today, this is a common high-tier Kickstarter reward—hang out with the developers for a bit, and get your game hand-delivered. Eno did this in 1996.

For his next game, Sega wanted to make it an exclusive—whatever it was. Eno had recently met with some sight-impaired folks who liked to play action games, and he asked himself, "What if you made a game that the blind and the sighted could play equally?" So he created the game REAL SOUND, which is an audio-only retail game, and made Sega promise that if he made the game exclusive to them, they would donate 1,000 Saturns to blind people, and he would supply 1,000 copies of the game. Again, this was an unusual idea for 1996, but he felt the stagnancy of the industry, and went to great lengths to shake it up.

His next game, D2, came during the next generation of consoles; D2 had moved to the Dreamcast, and he found himself making a "normal" game again. As he told *EGM*, "I had all of these kinds of ideas [in the past] because I was seeing the game industry from the outside. But around the time of D2, I felt like I was getting too close to the inside; I felt like I was turning into a normal game creator." And so around 1999 he left the game industry yet again, to refresh his perspective, becoming a creative consultant for a variety of industries.

But he just couldn't leave games completely alone. I first met him in 2005 at an E3 event, even though he wouldn't go on to release another game for several years. He impressed me as a wild force of personality and eccentricities, but also of principle. His urge to create was strong, so eventually he did come back to games in 2008, but not as the powerhouse he was before. Some of the fire was gone—Eno was no longer making headlines or fiercely challenging industry norms—but the creative force remained.

Just last year, for example, I was at a potluck in Brooklyn with some interactive media artist-types, and one was telling me about how his company had gotten a lot of press and recognition for an iPhone game sendup about just chasing one pixel around on a screen. They had just made this thing in early 2012. "That's funny," I said, "Kenji Eno did this already, way back in 2009, with

a game called ONE DOT ENEMIES." He may not have been the firecracker he was earlier in his career, but he was still blazing the occasional trail.

After Eno's death, I was speaking to NanaOn-Sha president Masaya Matsuura about the loss, and he told me this: "[English psychedelic rock pioneer] Kevin Ayers also passed away on February 18, and a note was found by his bed which said, 'You can't shine if you don't burn.' These words fit with Kenji's memories for me."

WHO WILL CARRY ON? The traditional game industry—the sector characterized by the developer/publisher relationship, big budgets, and a focus on developing for consoles—is shrinking, and companies pumping millions of dollars into sequels won't save it. If the traditional game industry wants to survive, it needs to identify and support its iconoclasts—people who believe in what they're doing so much they will risk everything to make it happen. Eno proved you could be an iconoclast and still be successful. He proved that you can have a wildly creative career and still be a success.

At the moment, we have plenty of iconoclasts—but aren't they all indies? Notch, for example, has blazed his own path with MINECRAFT and its business model. It's not because he fights the power, but because he doesn't care about the power. He has rocketed past most of the big companies that keep developers under their thumbs. But he's outside the traditional game industry. What does Sega or Sony or Activision mean to him, other than companies that occasionally make games he might like to play? Who will be Activision's

Kenji Eno (middle) and Brandon Sheffield (right).



champion now? When the founders of Infinity Ward go on to sue their former employers, only to get funding from EA, can our current infrastructure even support such dissidents?

Kenji Eno was that sort of champion, and now that he's gone, we need more people like him. Lots more, if we want traditional games to evolve and change with the times. We need more people who are willing to take the world on their backs in order to make it move. Maybe we need to take breaks from games and get some outside perspective, like Eno did. Maybe we need to believe in ourselves more, and believe in the power of our own ideas. But no matter what, we need to stop spinning in place with genre and theme and play style, and move forward. I'll leave you with another line from Eno, given during that *EGM* interview.

"I want to move forward. You have a short life; you're going to die someday. So I don't want to waste my time looking back on something I did in the past." ic

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

CONTRACTOR CORNER

A Genius Alternative

SUPERGENIUS



To Art Outsourcing

Specializing in the 5 disciplines of video game art production

www.supergenius-studio.com

FORGE

STUDIOS

Creativity & Innovation

Tailored to your needs

www.forgestudios.com



Complete Art Development and Production!

- USA Based Sales & Customer Support
- 3 Production Studios in China - (North) Beijing & (South) Ningbo
- Professional Project Management & Production Communication
- Competitive Rates based on Overseas Production
- 3D - High Quality Assets for Game Dev & CG Animation
- 2D - Concept, Design & Illustration - Production Concepts
- Keyframe Animation - 2D and 3D, Layout, Rigging & Weighting

**CENTURION
ART DEVELOPMENT**

1-888-411-9336 or contact@centurionartdev.com www.CenturionArtDev.com - Online Portfolio -



**Premium Art and Animation
Services Provider**

sales@smartboxent.com

www.smartboxent.com



**PRODUCER
CONCEPT
SCENE
CHARATERS**

technicolor



**Your Trusted Partner for
Animation & Sound**

www.technicolor.com/GameServices





GDC Vault

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

STUDIO AND EDUCATION MEMBERSHIP PACKAGES ALSO AVAILABLE.

CONTACT GILLIAN CROWLEY AT [GILLIAN.CROWLEY@UBM.COM](mailto:gillian.crowley@ubm.com)

GDCVAULT.COM

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

Featured Videos

Browse by Category

- Business & Marketing
- Game Career / Education
- Independent Games
- Scriptwriting / Level Design
- Production

Browse by Event

- GDC Dallas 2012
- GDC Europe 2012
- GDC 2012
- GDC China 2012
- GDC China 2011

Vault News

GDC Vault previews EA's debut...
Take from the GDC Vault Weekly...
JAMESON 10.2012

ADVERTISER INDEX

COMPANY NAME	PAGE #		
ART BULLY PRODUCTIONS	070	SMARTBOX ENTERTAINMENT	070
BALLY TECHNOLOGIES	051	SUPERGENIUS	070
CENTURION ART DEVELOPMENT	070	TECHEXCEL INC	005
COSTA IMC	006,007	TECHNICOLOR DIGITAL PRODUCTION	070
E3	052	VANCOUVER FILM SCHOOL	013
EPIC GAMES	014		
FORGE STUDIOS	070	IN ASSOCIATION WITH UBM TECH	PAGE #
ODOBO	C3,C4,001	GDC VAULT	071
ODOBO ADVERTORIAL	062,063,064	GAMASUTRA JOBS	055
PEAK GAMES	002		
RAD GAME TOOLS	C6		
RESEARCH IN MOTION CORPORATION	C5		
SCOTTISH DEVELOPMENT INT.	032		
SHARP SHADOW STUDIO	070		

For more information visit www.jointhegamenetwork.com

gd Game Developer (ISSN 1073-922X) is published monthly by UBM LLC, 303 Second Street, Suite 900 South, South Tower, San Francisco, CA 94107, (415) 947-6000. Please direct advertising and editorial inquiries to this address. Canadian Registered for GST as UBM LLC, GST No. R13288078, Customer No. 2116057, Agreement No. 40011901. SUBSCRIPTION RATES: Subscription rate for the U.S. is \$49.95 for twelve issues. Countries outside the U.S. must be prepaid in U.S. funds drawn on a U.S. bank or via credit card. Canada/Mexico: \$59.95; all other countries: \$69.95 (issues shipped via air delivery). Periodical postage paid at San Francisco, CA and additional mailing offices. POSTMASTER: Send address changes to Game Developer, P.O. Box 1274, Skokie, IL 60076-8274. CUSTOMER SERVICE: For subscription orders and changes of address, call toll-free in the U.S. (800) 250-2429 or fax (847) 647-5972. All other countries call (1) (847) 647-5928 or fax (1) (847) 647-5972. Send payments to gd Game Developer, P.O. Box 1274, Skokie, IL 60076-8274. Call toll-free in the U.S./Canada (800) 444-4881 or fax (785) 838-7566. All other countries call (1) (785) 841-1631 or fax (1) (785) 841-2624. Please remember to indicate gd Game Developer on any correspondence. All content, copyright gd Game Developer magazine/UBM LLC, unless otherwise indicated. Don't steal any of it. Or else.

the visitor from... the future

A TIME TRAVELER PAYS HOMAGE TO THE PAST

BAMF Hey, anybody know what year it is? 2013, you say? Praise Pincus, I made it! Huh, it looks different than it does in the holo-museum. Hey, I know what you're thinking: "This guy is just some kook and not actually a time traveler"—but it's true, I swear! I'm from the year A.P. 50 (that's, uh, 2113 to you). And I've come back to thank you, because this is where it all began—right here in this humble little video game studio. You may not yet realize it, but this very team is on the verge of solving humanity's greatest problem!

FREE TO EAT Okay, who said hunger? Ha ha, you people are real jokers. No, I mean leveraging user engagement to scale multiple revenue streams, obviously. In the future, we live in a utopian society without scarcity. It started with free-to-play games, but that was just the beginning. Every product, every service, everything anyone could ever want is free to use! Take that muffin on your desk. I bet you paid for that, didn't you, you past-dwelling rube? Well in 2113, if you want a muffin, you just go up to any FOUOTM Vendotron and take anything you want!

Now of course, for free, you don't get much of a choice of what's in the muffin. The Basic version doesn't list the ingredients and it isn't really, um, classified as food. That's okay though, the Rand Act of 2039 ensured that GOVERNMENT REGULATION (roll your eyes, everyone) doesn't apply to free goods, or anything else. Anyway, you've got your free muffin and you're enjoying it, but it seems a little bland, huh? You want to upgrade it a bit—well, that's simple! Just fire up your sosh...

oh, sorry, your "social," I guess is the term for you, ha ha. Tell all your frienollowers about the muffin and how awesome it is. If any of them also get a muffin with an upgrade in the next five minutes, you get a free upgrade yourself! Plug that muffin back into the Vendotron and select "Upgrade." After the bit of "Ode to Joy" and fireworks, presto, your muffin is brimming with succulent rehydrated blueberries!

IT'S SO EASY TO SHARE WITH YOUR FRIENDS Any schoolchild knows that stuff, though. Let me tell you how it is for working professionals like us. You're driving to work—well that's wasted unproductive time, isn't it? In the future, you could be earning FOUOCash™ simply by watching a few ads on your Gapple™ Glasses while you drive. Some people even stream two different ads at once, one to each eye. It's not like you actually have to pay attention though—ha ha! What a great way to earn free stuff, huh?

If you really want the FOUOCash to stream in, though, you have to make impressions on the people around you. Online impressions work pretty well—you post something on TwitrBookk+, like, "Gotta get my Stubbs™ Coffee this morning! No other brand will do! #stubbs #coffee #delicious" and every one of your friends who uplikevotes your comment gets a FOUOCoin, and you get one too. If you have a lot of frienollowers it really adds up!

AUGMENTED SOCIAL REALITY The latest thing, though, is real-life impressions. Mention a product by name in person to someone else and your Gapple Glasses will

detect the brand name being mentioned! As long as you say at least one positive thing about it, you'll get another bit of FOUOCash. And if you say THREE positive things, you'll go on a RepStreak—earning DOUBLE POINTS for the next 30 minutes! I know a couple folks who actually wake up every half hour in the middle of the night to call a friend to shout brand names at to keep their Streak going. Their loyalty is really inspiring!

And it's not just food. This morning I bet you showered with water that you had to pay for. Ha! It's amazing to me that people once lived this way. In the future, showering is also completely free. You get 100 seconds of water, and if you talk about showering on TwitrBookk+, mentioning your shampoo brand and everything, you'll get more time and might even be upgraded to the premium experience!

AND MORE... Well, looks like my time is running out. Too bad—I haven't even told you about how people can use RepStreaks to earn free Doritos yet, or how Britney Spears is the prime minister of France! Anyway, I should have been able to make enough impressions about muffins to get back home. See ya! #jumpin #oldscool #backin2013 #timetravel #pastpeopledressweird **ad**

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



G A M E D E S T I N A T I O N :

B L A C K B E R R Y 1 0

It's where your
game belongs.

BlackBerry® 10 offers a powerful and easy platform for game development. It's integrated with major development tools and leading game engines, including Unity, Marmalade and Shiva 3D. Plus, the leading BlackBerry 10 hardware produces a visually stunning and incredibly immersive gaming experience that really lets your masterpiece shine.

Learn more
developer.blackberry.com/games

 **BlackBerry**®



© 2013 Research In Motion Limited. All rights reserved. BlackBerry® RIM® Research In Motion® and related trademarks, names and logos are the property of Research In Motion Limited and are registered and/or used in the U.S. and countries around the world. Used under license from Research In Motion Limited. All other marks are the property of their respective owners.

IT'S HERE. BINK

2

Bink 2 Video has up to 6 times the quality than Bink 1 at the same bandwidth. It's also up to 3x faster due to it's SIMD design (70% of all instructions are SIMD in a frame decode) and perfect two CPU scaling. Available for Windows, Mac, Linux, (or any x86 or x64 system), Xbox 360, Playstation 3, PS Vita, iOS and Android.
www.radgametools.com 425-893-4300

