UBM

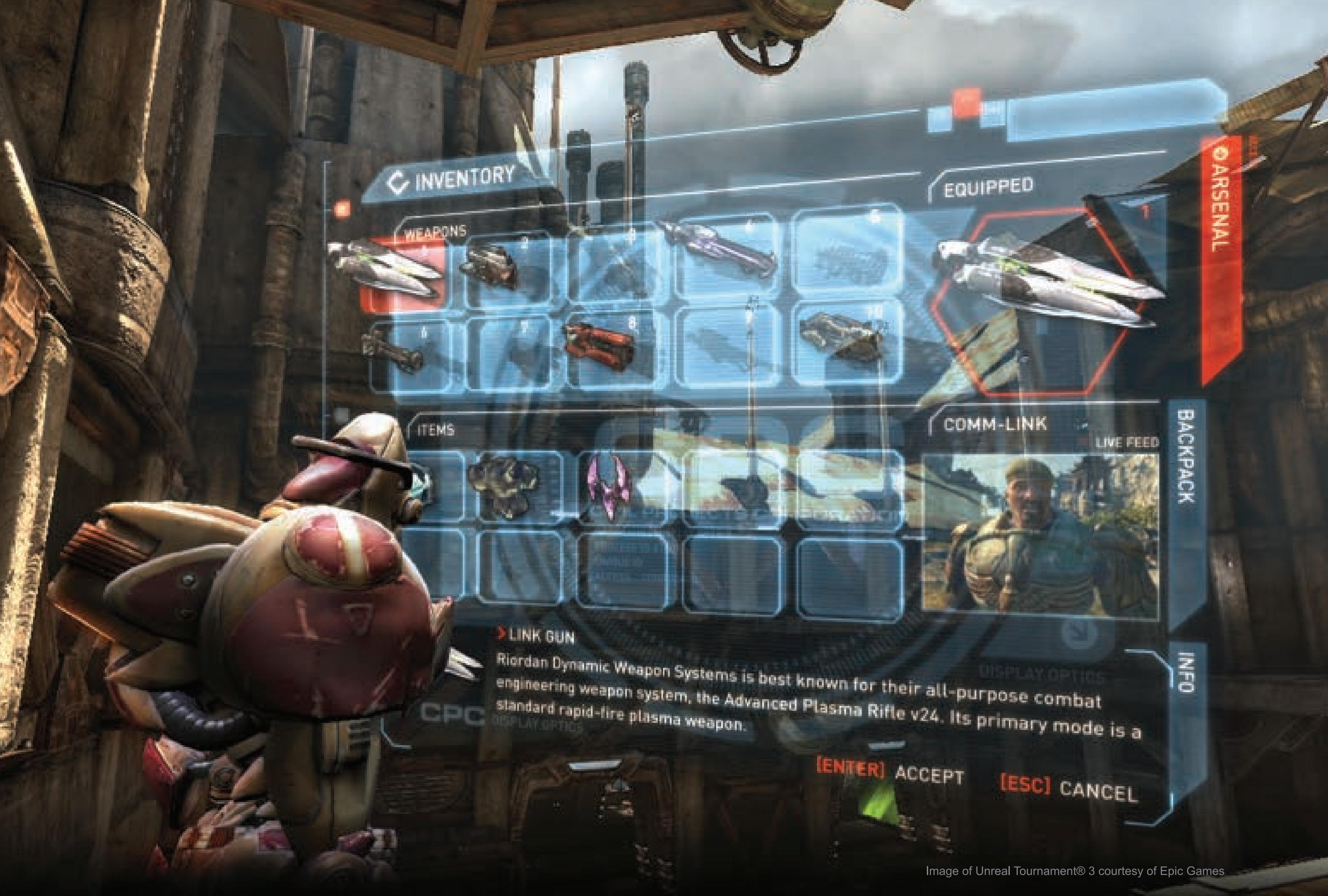
IF I SHOULD DIE TODAY AN ÉRIC CHAHI INTERVIEW

VOL17 NO8 AUGUST 2010

gamedeveloper

THE LEADING GAME INDUSTRY MAGAZINE

DEADLY PREMONITION



LINK GUN

Riordan Dynamic Weapon Systems is best known for their all-purpose combat engineering weapon system, the Advanced Plasma Rifle v24. Its primary mode is a standard rapid-fire plasma weapon.

[ENTER] ACCEPT [ESC] CANCEL

Image of Unreal Tournament® 3 courtesy of Epic Games

3Di is here.

Create animated 3D interfaces using Scaleform GFx 3.2

www.scaleform.com/3di

Scaleform®

CONTENTS.0810
VOLUME 17 NUMBER 7

POSTMORTEM

FEATURES

DEPARTMENTS

COVER ART: ACCESS GAMES AND JOSEPH MITCH

CONSTRAINED DESIGN

THE SIMPLER THE GAME, THE MORE IMPORTANT YOUR DESIGN DECISIONS BECOME

AROUND CHRISTMAS HERE IN the office, a coworker was given a very simple LCD game as a joke. The game was Super Racing from Techno Source, and as we kept it on a divider between our cubicles, the staff of *Game Developer* and Gamasutra wound up playing it more than any other game we have in the office.

Why would this be? There are a few reasons that are beyond the scope of the game itself. First, its simplicity is inviting—we can play it while having conversations about our plan of attack for the day, or simply to take a break from work, and it's a very low time and mental investment. This has obvious parallels to casual and social games, which have low barriers to entry, simple mechanics, and (in some cases) almost no consequence for failure.

Obviously the game is not an amazing world-changing piece of software, but more than the fact that it's easy and available, the game's simple design does a good job of reminding us of the importance of each decision we make when developing a game.

HOW IT PLAYS

» The aim is to race down a track, avoiding (or passing) all other cars along the way. Your car can absorb three hits before it's game over, and there are three waves, each with increasing speed. All you can do is move right and left. So why is this fun? That's where some very simple design and art choices help to elevate this game above the rest (we've since tried other LCD games here, and most are nowhere near as good).

LCD games usually play on a static field with an image, or no on background at all, and there is a set placement for the icons or dots (what we might call sprites in a traditional game), which can simply be on or off. In Super Racing, the background is a road, and approaching cars are indicated by larger and larger identical vehicles that simulate increased proximity. Here's where the first critical design choice comes in—the road is curved, and all the cars are drawn with some perspective, which helps tremendously with the sense of motion, compared to the straight-on perspective of many LCD racers.

Then, as a step to distinguish your car from the others (which are all drawn the same way), other cars never pass into your horizontal space, so it's more like you're aiming your vehicle into open spaces than actually passing cars in real time. The final piece of the puzzle is that the pattern has a degree of randomization, so there's always at least one space for your car, but you pretty much never play the same wave twice.

And that's it! There are a few more flourishes, like beeps for feedback when the cars advance, and a visual hit state if you're struck by another vehicle, and of course the necessity for responsive control, but that's by and large all standard stuff. The game proves that with a combination of a few smartly-realized elements, you can turn even the most simplistic of interactions into something enjoyable.

WHAT DOES IT MEAN?

» This may all sound silly, but the fewer game elements you have, the more important each one becomes, and for me, dissecting what made this game work versus other similar products was a good reminder of what's important in game design. At this scale, basic elements of art, gameplay, and systems design can have a huge impact on fun—and so too in casual games and the social space.

PopCap's Bejeweled, for instance, has had countless match-three imitators, but none have achieved the former's success. Bejeweled wasn't the first game to use the matching of three or more items as a main gameplay feature, but it succeeded because of its polish and attention to small details, from the physics of the jewels, to the satisfying sounds, to the visually uncluttered interface and large, colorful graphics. Without that focus on minutiae, it would've been just another puzzle game.

These lessons can be extended throughout game development. It's obvious that polish helps a game, and that critical design choices work best when finalized early. But when you look at it on the scale of an LCD game, a social game, or a casual game, you begin to appreciate the gravity of each choice.

Applying these lessons to create a game of similar constraints, or even simply paper prototyping an idea seems like a good exercise to help you refocus on what's important, and in fact got me unstuck in a recent project of my own. And it turns out that stepping back and analyzing what makes games fun, even something as simple as Super Racing, can be a real help in terms of getting your priorities straight. So break out your favorite Game & Watch and get on with the analysis!

—Brandon Sheffield

gamedeveloper
www.gdmag.com

600 Harrison St., 6th Fl.,
San Francisco, CA 94107
t: 415.947.6000 f: 415.947.6090

SUBSCRIPTION SERVICES

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

FOR DIGITAL SUBSCRIPTION INFORMATION
www.gdmag.com/digital

EDITORIAL

PUBLISHER
Simon Carless I *scarless@gdmag.com*
EDITOR-IN-CHIEF
Brandon Sheffield I *bsheffield@gdmag.com*
PRODUCTION EDITOR
Jeffrey Fleming I *jfleming@gdmag.com*
ART DIRECTOR
Joseph Mitch I *jmitch@gdmag.com*
PRODUCTION INTERN
Tom Curtis
CONTRIBUTING EDITORS
Jesse Harlin
Steve Theodore
Daniel Nelson
Soren Johnson
Damion Schubert
ADVISORY BOARD
Hal Barwood **Designer-at-Large**
Mick West **Independent**
Brad Bulkley **Neversoft**
Clinton Keith **Independent**
Brenda Brathwaite **Independent**
Bijan Forutanpour **Sony Online Entertainment**
Mark DeLoura **Google**
Carey Chico **Independent**

ADVERTISING SALES

GLOBAL SALES DIRECTOR
Aaron Murawski e: *amurawski@think-services.com*
t: 415.947.6227
MEDIA ACCOUNT MANAGER
John Malik Watson e: *jmwatson@think-services.com*
t: 415.947.6224
GLOBAL ACCOUNT MANAGER, RECRUITMENT
Gina Gross e: *ggross@think-services.com*
t: 415.947.6241
GLOBAL ACCOUNT MANAGER, EDUCATION
Rafael Vallin e: *rvallin@think-services.com*
t: 415.947.6223

ADVERTISING PRODUCTION

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

REPRINTS

WRIGHT'S MEDIA
Ryan Pratt e: *rpratt@wrightsreprints.com*
t: 877.652.5295

AUDIENCE DEVELOPMENT

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

MARKETING

MARKETING SPECIALIST Mellisa Andrade
e: *mandrade@think-services.com*

United Business Media
WWW.UBM.COM

Power up your game with advanced pathfinding and movement – done by the experts.

Download your free evaluation version at xaitment.com

xaitment
INTELLIGENCE ENGINEERED

STUDY DIRECTLY LINKS REVIEW SCORES TO CONSUMER PURCHASE INTENT

PopCap's PLANTS VS. ZOMBIES

THE RELATIONSHIP BETWEEN REVIEW SCORES AND SALES IS OFT-questioned, but now a ground breaking new study examining the impact of review scores on video game sales finds that high scores do indeed move units—even when players make their own quality assessments at the same time.

The Guildhall at SMU, together with analyst group EEDAR, studied a total of 165 people who'd never before played PopCap's popular PLANTS VS. ZOMBIES. PVZ was the only game chosen for the study, due to what was described as its unique combination of high quality and mass appeal.

Study participants were split into three groups. Prior to playing, one group was exposed to high-scored reviews of the game. A second group was exposed beforehand to negative reviews, and a control group was not shown any reviews. After playing for 20 minutes, players were asked to give their own evaluation of the game.

On completing the study, participants got a choice: Receive $10, or a free copy of the game. Fascinatingly, "participants exposed to higher review scores were twice as likely to take a copy of PLANTS VS. ZOMBIES over the $10 cash, and 85 percent more likely to take the game than the control group," says the study. And they were 121 percent more likely to take the game than were those who had seen poor scores beforehand. Players were not told the game's real-world online or retail pricing.

"The EEDAR/SMU study posits that the relationship between video game sales and professional review scores are not correlative but causal," the authors suggest.

Critical opinion not only affects purchase willingness, but also a player's own opinion: The group that saw high scores tended to offer their own scores on average 20 percent higher than those that were shown low scores, but only 6 percent higher than the control group who had not seen any scores.

"Consumer review scores had a greater variance from the mean than professional critic review scores, which had a tighter clustering around the mean," says the study. "The review score standard deviations of all the experimental groups were significantly higher than the standard deviation of professional reviews. Participant review scores ranged from 40 to 100, whilst the range for professional reviewers ranged from 60 to 100."

Being exposed to positive reviews also has a strong effect on whether players recommend a game to a friend or not: "91 percent of participants exposed to high review scores for PLANTS VS. ZOMBIES would recommend the product to a friend if they were asked to recommend a 'good game to play,' compared to only 65 percent from Group B (low review scores) and 80 percent from the control group." *—Leigh Alexander*

"INDIE FUND" OPEN FOR GAME SUBMISSIONS

THE "INDIE FUND," a funding source for independent developers financially backed by indies such as World of Goo's creators, BRAID's Jonathan Blow, and FLOW's Kellee Santiago, is now accepting submissions. The Indie Fund investors, creators behind some of the most successful indie games to hit the market, announced the fund in March this year.

Submissions to the Indie Fund should preferably "introduce something new to gaming," be in playable prototype form and developed by a small financially-efficient team, the group explained. Submitted games should also be able to make money through proven business models. Applicants submit a video and information about the creators.

The goal of the project is to help "encourage the next wave of game developers," according to the official website. The fund also calls itself "a serious alternative to the traditional publisher funding model."

One factor that sets the Indie Fund apart from traditional publishing deals, according to 2D Boy's Ron Carmel, is that it provides the funding "without the overhead or the loss of freedom associated with a publishing deal."

For games signed under the Indie Fund, there are no milestones, although developers must give the investors monthly builds to keep up to date on a game's progress. "You are free to experiment and evolve your design as you go, so long as you don't run too far over budget," the website says. The model also supports proportional payment, and developers have no long-term obligation to the backers.

With submissions now open, the Indie Fund said that it is currently "looking forward to seeing all the weird, amazing, surprising, silly, scary, titillating games you're willing to throw at us." Interested parties can apply at http://indie-fund.com. *—Kris Graft*

GAME DEVELOPER'S ANNUAL CAREER GUIDE

GAME DEVELOPER'S ANNUAL CAREER GUIDE issue, which caters to students and prospective game developers, is now available for free online viewing.

The issue does not ship to subscribers, as the content is entry-level, but is distributed at trade shows where prospective game developers gather, such as GDC and PAX.

This year's Career Guide focuses on getting games made as quickly as possible, offering full tutorials for Flash and Game Maker games. Also featured is a panel interview with student game developers that made it big, and a postmortem on IGNEOUS, an IGF finalist from students at DigiPen University.

The 2010 Game Career Guide can be found here, for online browsing and download: http://gamedeveloper.texterity.com/gamedeveloper/2010cg —*Staff*

SUMMER GHOSTS

FATAL FRAME 2 is just one of the horror games discussed.

LAST YEAR'S THINKING AFTER DARK CONFERENCE, WHICH WAS ORGANIZED BY THE LUDICINE research group at the University of Montreal (see *Game Developer* June/July 2009 issue), yielded a number of academic papers that examine horror video games from a variety of theoretical and critical perspectives. Selected papers from the conference were published last fall in the book *Horror Video Games: Essays on the Fusion of Fear and Play*, edited by Bernard Perron (McFarland, 2009). More recently, further papers from the conference, both in French and English, have been made freely available in pdf form in a special issue of *Loading ...* (http://journals.sfu.ca/loading/index.php/loading/index), the journal of the Canadian Game Studies Association. —*Jeffrey Fleming*

CALENDAR

GDC EUROPE

Koelnmesse | Cologne, Germany
August 16–18 | Price: 180–795 Euros
http://gdceurope.com

2010 IEEE CONFERENCE ON COMPUTATIONAL INTELLIGENCE AND GAMES

IT University of Copenhagen
Copenhagen, Denmark
August 18–21 | Price: See web site
http://game.itu.dk/cig2010

PAX PRIME

Washington State
Convention and Trade Center
Seattle, WA
September 3–5 | Price: $35–$55
www.paxsite.com

FUN AND GAMES CONFERENCE

Leuven, Vlaams-Brabant, Belgium
September 15–17 | Price 400 EUR
http://fng2010.org

FANTASTIC FEST SPAWNS INDIE GAMING FESTIVAL

ANNUAL GENRE FILM FESTIVAL FANTASTIC FEST has announced Fantastic Arcade, a spin-off event dedicated to showcasing new games from independent developers and publishers, at Austin's Highball and famous Alamo Drafthouse theater.

Running alongside Fantastic Fest at nearby venues from September 23–26, Fantastic Arcade's featured indie game developers will have an opportunity to show off their latest titles to an audience of film-goers and gamers.

Along with setups that will allow visitors to play indie games on display, the four-day event will offer a games showcase from major publishers, panel sessions with game industry guest speakers, and live game demos on Alamo Drafthouse's movie screens.

Fantastic Arcade will also feature a machinima film competition judged by *Red Vs. Blue* creator Burnie Burns, chiptune music performances, screenings of games culture films, video game art installations, a Starcade competition, and a "light saber dance party." —*Eric Caoili*

MAKE IT FAST!

BEST PRACTICES FOR GETTING YOUR SOCIAL GAME OUT AND KEEPING IT RUNNING

AMITT MAHAJAN

MAKE IT FAST

FARMVILLE has over 70 million players playing monthly. The game was developed in five weeks by a small team.

Social games have really taken off in the last year. With the release of the Facebook platform in 2007, there has been an abundance of opportunity for small developers to reach massive audiences. Since the platform's release, there has been a dramatic increase in production quality. Games that originally started off as text-based experiences turned into full-fledged Flash clients that spared no expense at visually and aurally delighting players. While this is great for the user experience, game developers are now faced with more upfront work to release a game and subsequent updates.

In order to stay competitive, developers need to be able to react quickly to both market focus and to the ever-changing Facebook platform. New opportunities for growth are being added all the time as Facebook matures. The developers that properly engineer their games and quickly react to these changes are the ones that will make the greatest gains in market share. In addition, given that social gaming is a service rather than a shipped product, it is necessary to dedicate a significant amount of resources and effort to building out a scalable server architecture that can support a user base of millions of daily players.

In spite of all this, developers coming from traditional games need not be intimidated by social games. Many of the techniques used in traditional game development can be applied to social games. Developing a social game is an opportunity to supplement training in developing boxed games with best practices from developing scalable web services.

Given these conditions and our experiences developing games such as FARMVILLE and TREASURE ISLE, Zynga has established several best practices that lead to quickly releasing games and then continually supporting them with updates to provide players with a great experience without any disruption in service. While these methods are not foolproof, we have found that by starting with lessons learned from previous projects, we are able to continually adapt to this growing segment of the game industry.

THINKING DIFFERENTLY ABOUT SOCIAL GAMES

» Social games differ in major ways from traditional games. These differences may throw off developers who are starting to develop social games after coming from a traditional game background.

First, social games are services, not boxed products. Shipping a game is the starting point for a social game's life cycle, and that's when the team is truly tested. Games are meant to evolve and grow with the user base and players have come to expect consistent updates to keep their interest. In the case of Zynga's FARMVILLE, we established an update pattern of twice per week. Prior to that, we were releasing every day as we found issues immediately post-launch.

Second, since people are interacting with your game in real time, real player behavior can be used to guide decisions. In traditional games, experience and designer "gut" tend to drive the decision-making process. Occasionally, playtesting feedback is involved which sometimes helps shift decisions, but overall, it's more of a qualitative process than a quantitative one. For social games, frequently, the result in design meetings is to defer a decision until more real-time data has been gathered. For example, when deciding what items to add to the market in FarmVille, we look at player buying patterns from previous weeks to determine how to best utilize our art team's time.

Finally, social games aren't self-contained experiences. Players want to play these games with their friends and, if properly engaged and entertained, will go out of their way to seek out opportunities to bring their friends to your game. This is a very powerful difference from traditional boxed games where word-of-mouth praise is less likely to result in a sale. In fact, it is this difference that allows a social game to go from zero players to an audience of millions seemingly overnight.

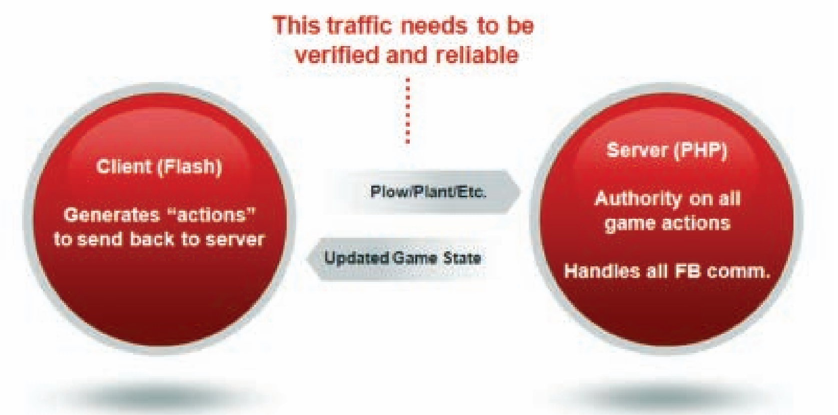

This traffic needs to be verified and reliable
Client (Flash)
Generates "actions" to send back to server
Plow/Plant/Etc.
Updated Game State
Server (PHP)
Authority on all game actions
Handles all FB comm.

FIGURE 1 Newer social games are typically divided into two codebases: a Flash-based client and a web server.

LISTING 1

Example from FarmVille's game settings XML.

```
<item name="strawberry" className="Crop">
        <levelNeeded>1</ levelNeeded >
        <cost>10</cost>
        <timeToGrow>0.174</ timeToGrow >
        <coinYield>35</coinYield>
        <image name="seeded" url="assets/crops/crop_strawberry.swf" />
        <image name="stage1" url="assets/crops/crop_strawberry.swf" />
        <image name="stage2" url="assets/crops/crop_strawberry.swf" />
        <image name="harvestable" url="assets/crops/crop_strawberry.swf" />
        <image name="withered" url="assets/crops/crop_strawberry.swf" />
        <image name="icon" url="assets/crops/crop_strawberry_icon.png" />
</item>
```

THE STRUCTURE OF SOCIAL GAMES

>> In order to provide rich experiences to users, many social games have shifted to a client-server model. Usually, there is a user-facing front end or client written in HTML or Flash and a back end running on a set of web servers written in PHP, Python, or some other form of server language. Players interact with the client and send commands back to the server for processing.

For Zynga's FarmVille, Treasure Isle, and other games, we have a client written in Flash that players interact with. This client gathers commands executed by the player (plow this field, plant strawberries, and so on) and sends them back to the server so we can verify that the commands are valid, and then update our version of the player state accordingly (see Figure 1).

Social games pose a variety of technical challenges that combine the best practices of traditional game development with those from web development. This has created a unique set of challenges not found anywhere else and has made social game development an exciting place to test new ideas. Luckily, this process is repeatable and something that can be invested in, so it will only become easier for experienced developers to create games for social networks as their technologies mature.

REDUCING DEVELOPMENT TIME

>> When FarmVille was first conceived, the original goal was to have a game out in under two months. Focusing on this date, we knew that we would have to take an alternative approach to developing the game. After trying out several different methods unsuccessfully, we finally settled on an agile-based method of development that suited the types of products we were developing.

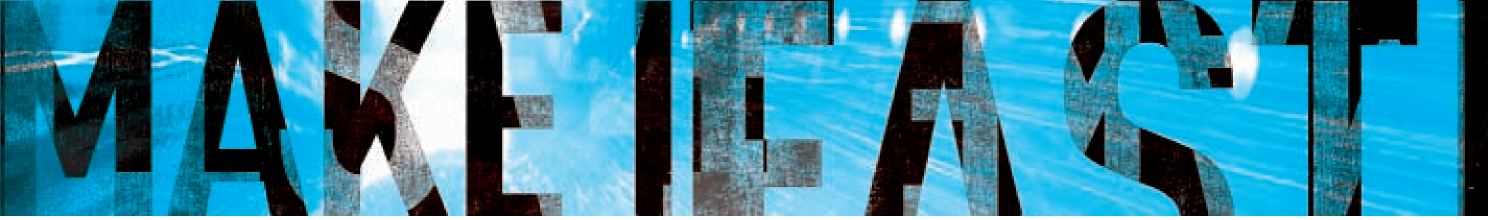
Developers work on both the front end and back end. Having to deal with multiple codebases and languages can be difficult to manage. So a team's natural inclination is to hire engineers that specialize in each codebase separately. In the case of social games, this would be front end (Flash) developers and back end (PHP) developers.

One way to increase efficiency is to have engineers work on features instead of a specific codebase. This means that, while there is an initial cost in training engineers in both codebases, they are able to completely own a feature from top to bottom. Taking this approach has the benefit of reducing the number of people the developer needs to interact with in order to complete their feature.

Pairing designers with coders. Normally, in software development, programmers work on code only, and the design team creates detailed feature specs that are then handed to coders for implementation. Looking at our short timeline for release, we decided that we didn't have the bandwidth to create detailed specs on how the game should be implemented. This method would cause too much back and forth between coders and designers, and would be inefficient in weeding out unrealistic ideas early in the process.

Instead, we experimented with a process in which designers would start creating a new feature by writing a one-page summary of their idea and how it generally should function. They are then paired up with a coder to co-own the feature alongside them. The web developer can push back on design when it would be relatively hard to implement and is held accountable for the performance of the feature as much as the designer is. We found that this method of feature development resulted in better features with less thrashing since programmers often had mutually-exclusive knowledge that could simplify and improve designs.

We took this approach when developing our chicken coop feature for FarmVille. Players could place their chickens inside the coop, from which they could harvest multi-colored eggs. They could share these eggs with their friends on Facebook, allowing them to receive a variety of prizes by opening the eggs. The feature designers started by writing a one-page summary of key bullet points on how the chicken coop should function, allowing the coder to provide feedback and begin work on a prototype of the feature. The designer then incorporated this feedback and fleshed out the rest of the feature, the copy, any key



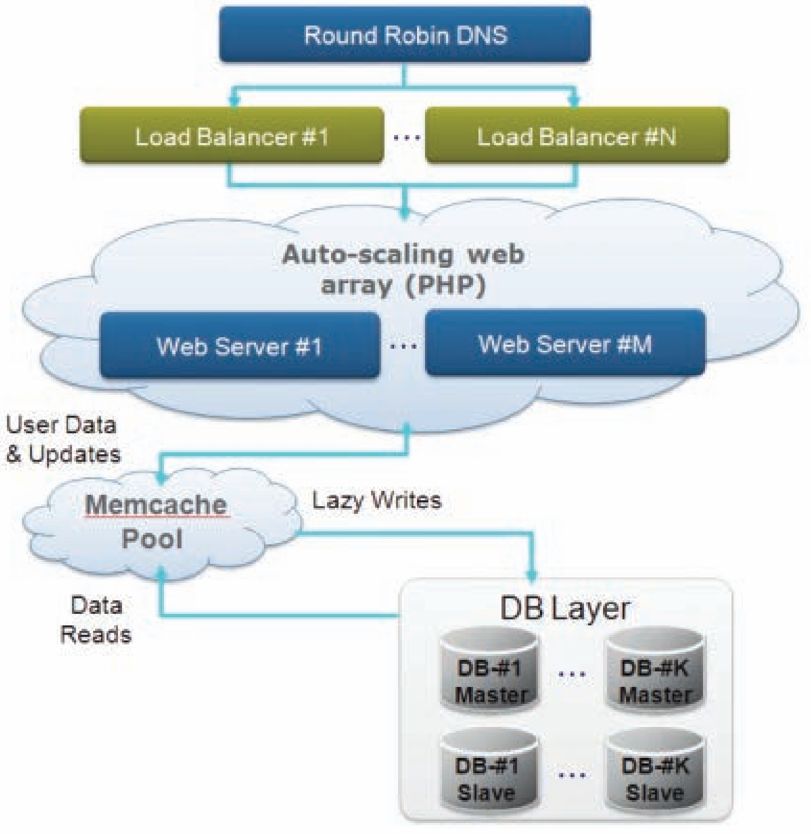


FIGURE 2 **FarmVille's server architecture scales with the number of players currently playing the game.**

metrics to track, and edge cases. After the web developer had a prototype ready, the designer will have already prepared final details allowing them to meet up and finish the feature in less than a week from concept-to-release. Without a tight coupling of a designer and a coder, this would have never been possible in such a short timeframe.

Create systems, not features. A disproportionate amount of the time during our games' development has been devoted to the adding and tweaking of game content. Realizing that it was unrealistic to involve a programmer any time we had to add content to the game, we established a policy of having our programmers create systems instead of features. For us, a system is something that multiple features can be built upon without having to involve any coders. This is commonly known in traditional games as practicing data-driven design.

Generally, when people think of data-driven design they think of tool chains and WYSIWYG editors that allow artists to create content for the game. While these solutions are great and the eventual goal, we needed a way of getting the majority of the benefit of these tools with only a fraction of time to focus on them.

In FarmVille, we use an XML file that defines a lot of the game's mechanics. Exposing the ability to add crops and other items through this XML file allowed the designers to add content to the game with a text editor and without a coder (see Listing 1).

Embracing existing technology. The majority of our games are based on the open source LAMMP stack: Linux, Apache, MySQL, Memcache, and PHP. This has saved us countless hours and dollars. We have also found that by building games on commonly available technologies, it has been easier to hire new developers and acquire talented companies.

INTEGRATING WITH SOCIAL NETWORKS

>> A development challenge that is completely unique to social gaming is integrating with third party networks that form the foundation for the game's features. While social networks provide huge value to the engagement and virality of the games, they also pose another dependency and possible point of failure that your game needs to account for.

In our case, we have developed an abstraction layer that makes social network integration easier to manage. Writing a class that encapsulates all your interaction with Facebook can provide several advantages over allowing feature developers raw access to the Facebook APIs:

SIMPLER INTEGRATION

Almost all features in our games require some sort of social interaction. By simplifying the APIs to the most commonly-used functionality, we give developers the ability to add social network hooks in a couple lines of code.

ADAPTABILITY

The Facebook platform is in flux and constantly changing. Creating a common interface through which to use the libraries allows us to change the implementation in one location as the APIs change. For FarmVille, we are usually able to adapt to changes the day they are made by Facebook by modifying our implementation of the social network layer.

FAULT TOLERANCE

Given that Facebook is a third-party service, fault tolerance is important in the case that our servers can experience any sort of latency while communicating with the Facebook platform. We can build this fault tolerance in at the lowest level so that games can keep running even if the Facebook platform is unresponsive.

SCALING SOCIAL GAMES

>> A large difference between social games and traditional games is the amount of work that needs to go into creating server architectures capable of handling billions of requests a day. Investing in a solid architecture up front will give you peace of mind and a lot of leverage when working on multiple titles. During the creation of FarmVille, the team developed a cloud-based server architecture that enabled us to scale and continues to be Zynga's architecture of choice when deploying new social games.

Auto-scaling Cloud Architecture. Engineers at Zynga are encouraged to evaluate new technologies and are given a large amount of ownership with respect to architecture decisions. Even though Zynga's games have traditionally been hosted within datacenters, this level of autonomy resulted in a cloud-based architecture for FarmVille and Treasure Isle. Running on the cloud has allowed our newer games to scale in a way that they normally would not have been able to within a data center.

We set up our cloud array to auto-scale depending on current load. If the load increased,

the system would automatically add new web servers to the server pool. If it decreased, the system would remove servers. This method of auto-scaling ensured that our max system capacity was always ahead of our current load and that we were not wasting money by having too many extra servers commissioned at once.

Even then, the auto-scaling features of the cloud were not a silver bullet and there were several techniques we had to employ to ensure system reliability.

Horizontal scaling. One of the issues with using the cloud is that you will be inherently limited by the performance of available hardware. There is only so much you can increase within the hardware limits on any given cloud server. To work around this restriction, server architectures would need to scale across several boxes of average performance.

If you look at FarmVille's server diagram (see Figure 2), each box represents a single cloud server. These servers are grouped into sets of specific functionality. You will see that each set of servers is scalable by launching new instances of that server.

For example, when requests come into FarmVille, they are first routed through a Round Robin DNS server. This server simply takes the domain name clients connect to and rotates through a set of IP addresses to route it to. Each IP address points to a load balancer server which monitors a set of web servers running our PHP code. It sends the request to the web server with the lowest load at the time. The web server performs some game logic and then updates the player state as needed.

In the event that any one of these components were to begin experiencing heavy load, more servers of that type could be added without having to take the app down.

Removing the database as a bottleneck. Given the heavy click nature of most social games, it is likely that your user database will have to be constantly updated. To sidestep the database being a possible bottleneck, Memcache can be used as an alternative method of updating user's data. Memcache is an open-source distributed memory storage system that allows fast reads and writes because all of the data is stored in RAM. Using Memcache as a first line of data storage/retrieval and then lazy writing back to the database periodically has proven to be a scalable solution, as user updates will be immediately written to a stored location with little delay.

The database still provides data redundancy and backups. In the case that a Memcache server crashes, users' states can be fully restored without data loss.

By mirroring each master database with a slave using database replication, the databases themselves can also have redundancy built-in. If we were to lose a master database, we simply promote the slave to a master and bring a new slave online.

LAUNCHING

» Launching a social game is a starting point, not the finish line. Several preparations can be made just before launch to help smooth out any issues that may occur once a game is live and servicing players.

Load Testing. After having built out your server architecture, it is worthwhile to test it under real-world conditions to avoid any surprises.

For FarmVille and Treasure Isle, we tested our architecture under simulated loads of up to a couple million users. These simulations helped us locate and resolve bugs in our configuration, saving us costly downtime once we were live.

Monitoring and Alerting. A good goal to have before launch is to ensure that everything keeps running after the game has gone live. And in the event of a server failure, you should have processes in place to alert the team as soon as possible.

At Zynga, we use two core services to give us confidence that our games are still running as expected:

NAGIOS

Nagios is a server monitoring service that watches your servers and alerts you when there are deviations from the norm. We use Nagios to watch our traffic graphs, error rates, and so on. It is infinitely customizable and is set up to be extended with your own specific needs. See www.nagios.org for more information.

MUNIN

Munin is a dashboard that allows you to visually monitor your servers. It provides graphs of every server metric you would ever want to know about and helps you diagnose and resolve problems. We watch our server loads very carefully using Munin because they provide insight on when there are network problems, even if only for a fraction of a second. See http://munin-monitoring.org for more information.

KILL SWITCHES

» By building kill switches into each feature in a game, you are given fine-grained control over how much load is currently on your servers. Having this ability can save you costly downtime by preventing your app from going down.

For example, we noticed one day that our FarmVille Memcache servers were operating under heavy load. We decided to disable the displaying of statistics on the page that shows a player's neighbors. We knew that this page added heavily to the Memcache load because it accessed Memcache quite a bit to display friends' information. By having a simple method to control which features are on, we can have our app adapt to load fairly easily without the assistance of a programmer.

TRACK PLAYER BEHAVIOR

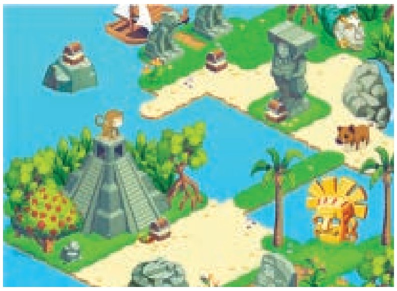
» Social games need to be updated to truly engage audiences. Building event tracking into your game gives you insight into what players are doing and lets you respond in logical ways to player behavior.

We often look at load times and other performance metrics to determine how much engineer time to spend time on perceived problems. By tracking where players stopped playing in FarmVille, we were able to find out that players were dropping off at a high rate before ever entering the game. We traced the problem back to long load times and dedicated engineering resources to resolve the issue. After pushing out an update, we noticed a significant increase almost immediately in the number of users entering the game. Having proper instrumentation gave us the ability to focus on the real problems affecting our game instead of chasing red herrings or theories.

CONCLUSION

» In terms of production values, the rapid progression of social gaming is reminiscent of the advancement path of console games. By employing lessons learned from traditional gaming and applying some knowledge from developing on the web, it is a smooth transition from traditional games to social gaming. With quick development cycles and the possibility to reach millions seemingly overnight, this is an exciting time to be a game developer, and I'm eager to see what the next generation of social games will become.

AMITT MAHAJAN *is a director of engineering at Zynga where he headed up the development of FarmVille and Treasure Isle. He can be reached at* ***amahajan@zynga.com****.*



Zynga's Treasure Isle was developed using several graphics techniques found in traditional games.

CRYTEK
THE POWER TO ACHIEVE YOUR VISION
Visit us at
GDC EUROPE
Aug 16-18, Cologne

FINDING PERSONALITY IN GAMES

WHY MEGA MAN'S JUMPING FACIAL EXPRESSION IS MORE IMPORTANT THAN NORMAL MAPPING

JOSH BEAR

When I first started out in the industry, I worked under the mantra: "gameplay is king." No matter what, the game needed to be fun and interesting through mechanics alone. I figured I should be able to use a box as my main character and find a way for the player to enjoy controlling that box in an environment.

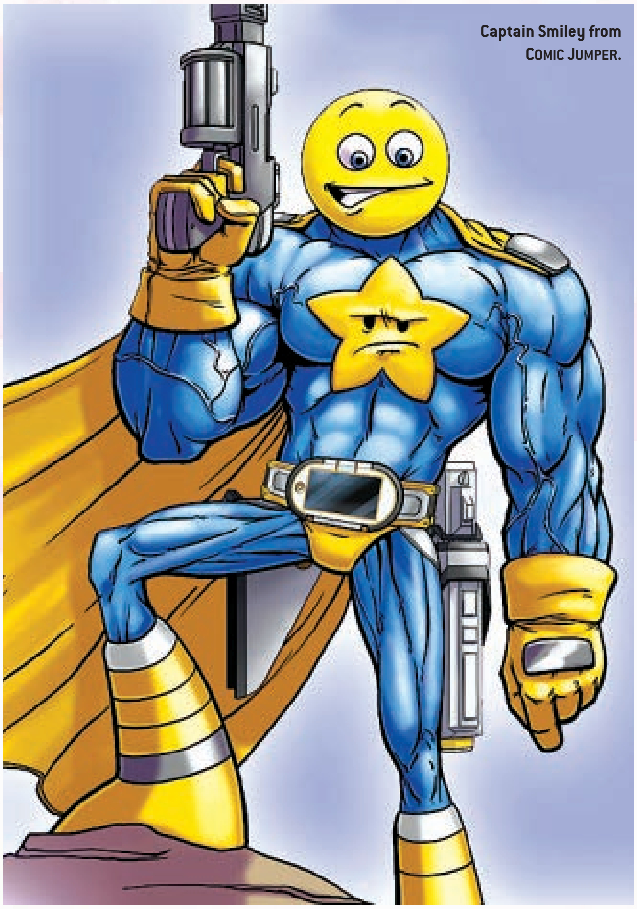
After several years of designing games and co-founding a company (Twisted Pixel), it became obvious that as great as that sounds, it isn't always the case. Having great gameplay can create a fun experience for players, but without support from other aspects of game creation, that gameplay isn't able to do other things, like create fans or memorable experiences.

Our recent games have received some very kind compliments from fans and the gaming press. Our characters were called unique, and were said to bring personality to our games. That word, "personality," seems to only be associated with characters most of the time. While I'm not an expert by any means, this article will discuss some reasons why and how personality has worked in our games and others. I believe that personality isn't just defined by a game's main character; it is something that can be pulled from all aspects of a game. Sometimes this is very meticulously planned out. Other times, it's a complete accident. Taking the time to try to understand how this works has benefited us as a company, and helps our team make sure each of our games is a memorable experience, whether people passionately love it or hate it. Either one is better than being forgotten.

>>>

Super Mario Bros.

Mega Man 9.


Captain Smiley from Comic Jumper.

Is personality important? Absolutely. Is it easy to do? Not always. Too many games (especially licensed games and games aimed at children) assume that personality just comes from the license: if the game character looks like the main character from the television show and they share the same voice actor, then personality is covered, right? It seems like common sense that nobody would think that way, but it happens all the time, even with the coolest characters in the entertainment medium. Games can go so much further when creators take the time to understand how personality works and what it truly means in interactive media.

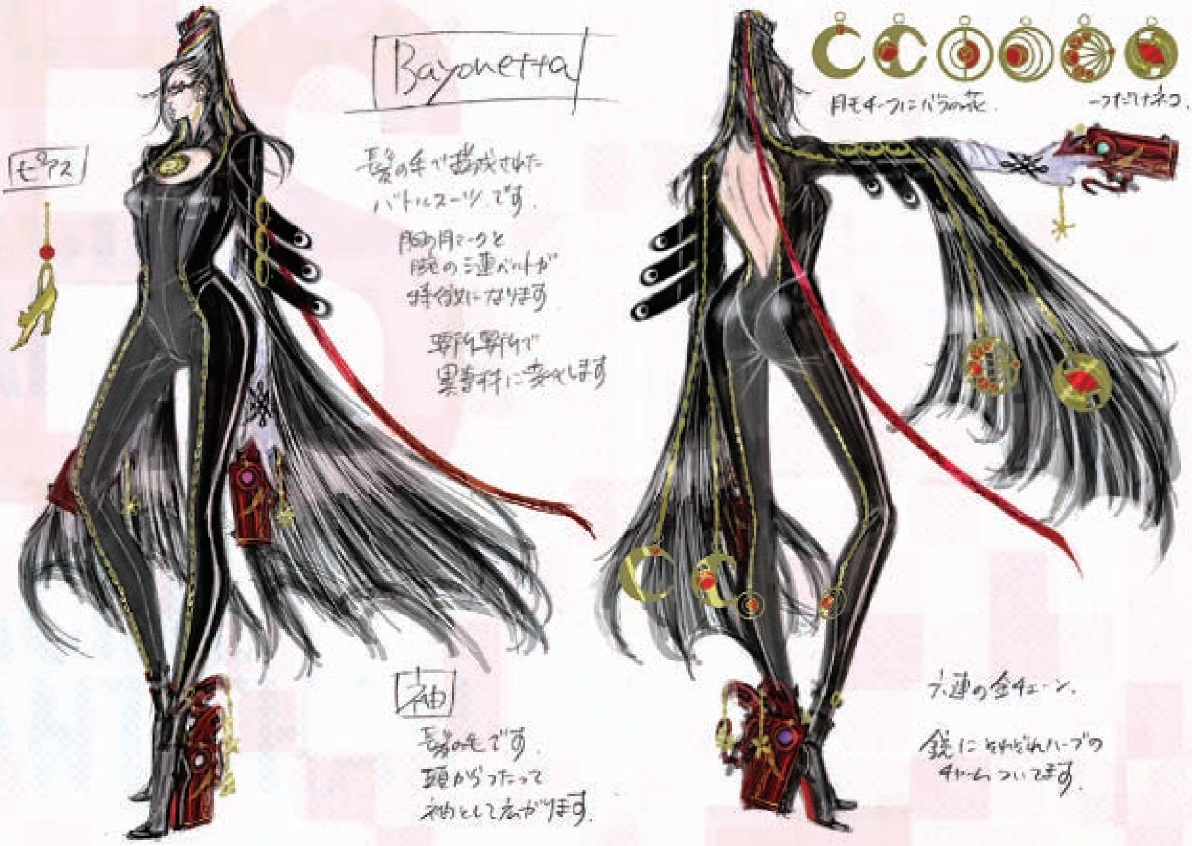
PERSONALITY IS CHARACTER DESIGN

» Character design is a great first place to start figuring out how to infuse personality into your game. Sometimes that design is influenced by limitations of hardware, memory restrictions, and the like. A good example of this is Mario and Luigi from the original Super Mario Bros. Most people already know the story, but it is said that when Shigeru Miyamoto designed Mario, he gave him a mustache to help separate his nose from his mouth and chin area better. Mario's hat was given to him because hair was more difficult to show with the limited pixel space, and the overalls were there simply to help break up his body so that his arms were more visible. This necessity to create an icon within hardware constraints ended up creating arguably the most memorable character in games today.

Mega Man is another great example of character design working within limitations of hardware. The original Mega Man concept art made it clear that the artist had a specific vision for Mega Man's face, but the limitations of the NES hardware made this impossible to realize. So by just enlarging his eyes and forming his mouth into a big, simple "O" when he jumped, Mega Man was instantly recognizable. It didn't matter if you liked him or not, it was more about the recognition, and how he stood out as a character with "better" graphics than other NES characters, because they took the limitations of the platform into account.

Designing memorable and lasting characters isn't an easy thing, and there isn't one process that works. You have to find what works best for you and your team. When we design characters at our studio, lead concept artist Brandon Ford will start roughing out a ton of crazy ideas. After that, the two of us will get together and go over what he started, eliminating elements that we presume won't work for the gameplay style we are going for, or just plain suck. But the most important part of the first step is that there isn't a ton of direction. He just throws down whatever he thinks is cool on paper. That way we aren't limiting ourselves before we even start. Don't go into character design with obstacles. You may have the perfect vision of what you think the character needs to be, but be open to something totally new and radical replacing your original thoughts.

For our character Captain Smiley in our new game Comic Jumper, we designed him to be the most generic character we could think of, since that fit with the story we were trying to tell. Aside from the generic style of it, another reason we decided to go with the smiley-based head is that it would be much easier to show extreme expressions with it through animation than if we went with a human face. With so many game characters out there (and comic book characters which are the basis of the games) it would be hard to come up with something original enough that we would consider memorable just by looking at a still illustration. So love it or hate it, the smiley face head was instantly recognizable and memorable, and fit into what we were trying to accomplish with the character.



BAYONETTA is a good example of distinctive character design.

BioShock.

Recent game characters such as Bayonetta and the Big Daddy from BioShock have done a great job of standing out in a sea of game characters that are all competing for the attention of game players. It would have been easy for a character like Bayonetta to be generic (and perhaps some people think she is), but for my money, her over-the-top design keeps her separate from other similar characters. Her long, crazy hair forms her costume; she holds guns not only with her hands, but on her feet. And then there are those extremely long legs that make her not entirely proportional but aren't pushing it so far that she looks "broken." This is just one example of how a creatively designed character can add a lot of personality to a game.

PERSONALITY IS ANIMATION

» No matter how awesome your graphics technology is, or how many polys a character has, animation is more important, especially when you want to show personality through characters. There have been a lot of recent games that look amazing in screenshots but fall apart when seen in motion. Most players can recognize bad animation right away, especially if it's coming from a humanoid type character.

Earthworm Jim, even though it's an older example, continues to be a great way to show how animation defines a character. Jim and all the other characters in the game had a huge amount of detail to their motion, even down to the idle animations. To do this successfully in your own games, work with what you have and take advantage of your strengths. In our case, we tried to make Splosion Man wacky and crazy through animation to help him stand out from other game characters. We knew early on that we didn't have the manpower to create unique environments for every stage nor a multitude of enemies to go along with them. By focusing on animations for Splosion Man and the scientist characters, we could pull the player's eye more toward that aspect and less toward what we couldn't do that well at the time.

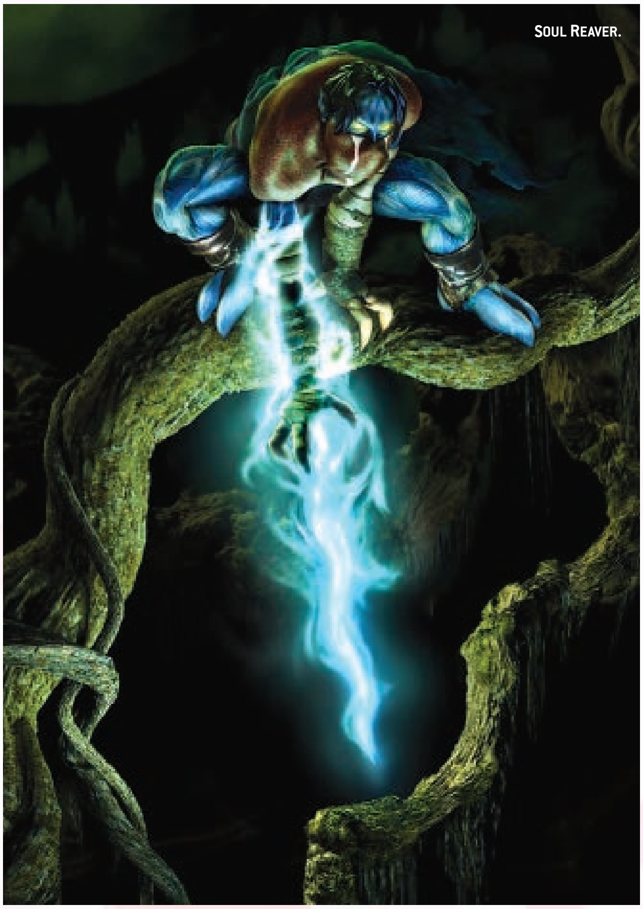
We used the same tactic on our first game, The Maw (see the December 2008 issue of *Game Developer*). With only one artist, we had to choose something to focus on, so we decided that the relationship between Frank and Maw would be more interesting and important than the backgrounds and other objects that populated the game. It would have been great to have a level artist, but that wasn't an important part of getting personality from the end product. Focusing on your strengths, especially if you are a small company, will go a long way. In our case, what we had was a great concept artist and animator, so that's what our game reflected.

Of course, too much animation can be a burden when the animator isn't thinking about how gameplay fits into the equation. There have been a number of games with amazing animation, but where the animator was thinking more in movie terms, and control suffered because the player had to wait for those animations to end. There's a lot of fine-tuning in this process, which is part of what separates beginning animators from the veterans. It usually helps when the animator loves games and gameplay, and doesn't just love animation on its own terms.

Soul Reaver is an older title that did this well, in addition to taking character design into account. The main character Raziel had a great animation style that found the right balance that made him simultaneously fun and fluid to control. While it was a departure in visual style from the first game in the Blood Omen series, this ended up working in the team's favor. By enlarging areas of Raziel's body—like hands and feet—the developers had an easier time expressing his character through animation. This led to a lot of

Earthworm Jim.

Splosion Man.


Soul Reaver.

players remembering Raziel fondly. Animating too much, if it interferes with play, is just as bad as not having enough animation. Knowing the right balance makes all the difference.

PERSONALITY IS GAMEPLAY

» There are a lot of games that ignore the above completely, but manage to have a character all their own. Geometry Wars is a great example of this. Using nothing more than simple shapes with glowing outlines as characters, Geometry Wars still manages to create a sense of personality that separates it from the countless other games that have the exact same gameplay mechanic. This is because of the way the game combined visual effects and feedback to create what amounts to a gorgeous fireworks display. The tight relationship between the controls, the explosive colors, and gameplay rewards made this game stick in peoples' minds visually.

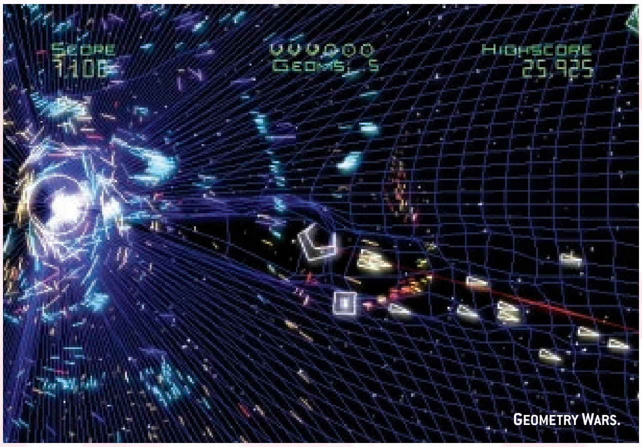
There's a perfect storm of personality brewing in a game like Katamari Damacy. The soundtrack, strange humor, and simple graphics all help to give it a unique identity. At the time the game came out, though, the thing that stood out the most was the way the game played. Rolling a large ball around and collecting junk, animals, and eventually people was very different from what most players were used to seeing. While roaming the convention center during E3 when the game made its debut, people would stop and look at the game and wait in line to play it based solely on the gameplay they were seeing. They couldn't even hear the strange music that most people associate with the game.

Don't confuse infusing personality into your gameplay with overcomplicating your gameplay style. Gameplay should supplement and enhance personality, but not be completely determined by it. We decided that we wanted to make a game that only used one action button (not including directional control) to make it as accessible to as many players as possible. So in Splosion Man, players can make the main character 'splode three times before he must touch the ground or slide down a wall to recharge again. The idea of 'sploding itself is a system that stems from gameplay but is something that people will remember and associate with the character; Splosion Man splodes, that is his thing. But 'splode is essentially just jump, nothing more, and it's used basically the same as it is in Super Mario Bros. But Splosion Man's "jump" has more personality than most characters in similar games, because it has a unique hook that gives the main character more of a memorable feel and look.

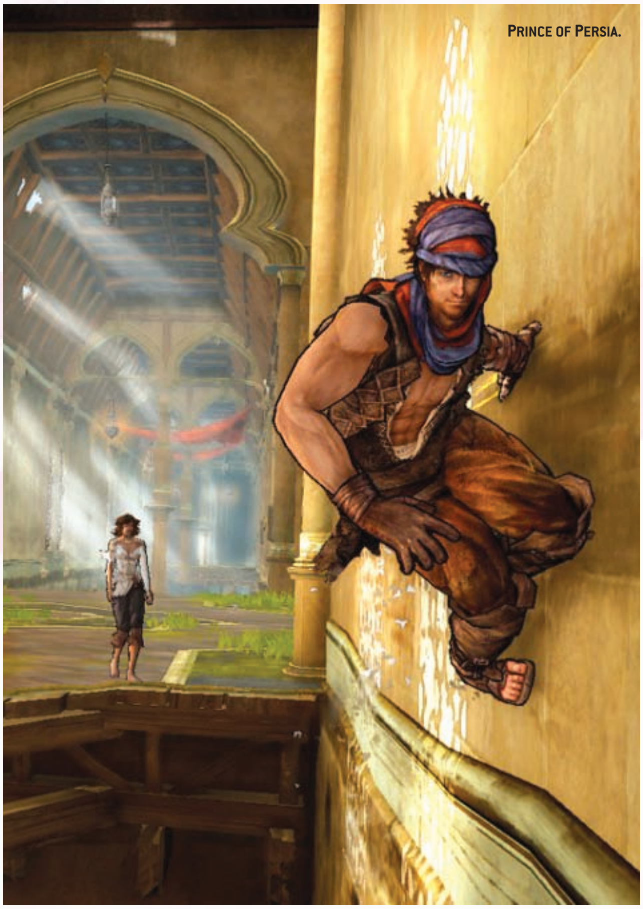
PERSONALITY IS SOUND

» Sound goes a long way toward defining the personality and feel of any game. Let's take the example of Katamari Damacy again. The unexpected vocal-oriented soundtrack defined the tone and humor of the game, and many players associate the music very heavily with the game. As others have said before, music that plays against expectation can create a more memorable experience. Another good example is the announcer in all of the Mortal Kombat games. Combined with the writing of the characters' lines, the deep voice of the narrator let everyone know that the game had a dark sense of humor and was never taking itself too seriously. The team eventually took this idea a step further, with the face of lead audio designer Dan Forden coming out of the corner to yell "Toasty!" every so often. To this day, I don't remember any of the combos or moves that I used to repeat over and over in MK2 and MK3, but I will always fondly remember Forden and his shout out.

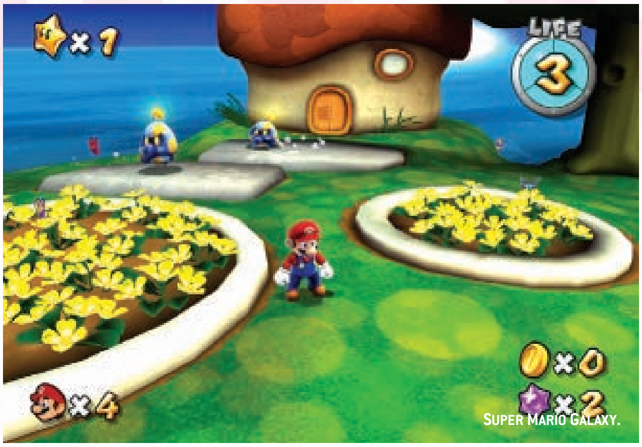
Having a good sound designer on your team who is open to input is essential. The sound



Geometry Wars.



Prince of Persia.



Super Mario Galaxy.

designer should be integrated into the core team, not treated as an outsider. In Splosion Man, we designed the music tracks out in stems, so that in single-player mode, while the music played in the background, when Splosion Man jumped, it would unveil a new portion of the music track which introduced a guitar layer. In the four-player multiplayer levels, each Splosion Man had his own musical instrument stem, so that when all Splosion Men were alive and well, the player would hear the entire score for that level. If a Splosion Man died, the other players would subliminally miss that music stem and want to get the player back.

As cool as these ideas were, and even though they were implemented well, they were a failure compared to *The Donut Song* that was also in the game. Most people who played Splosion Man probably don't even recognize the music stem design at all, or realize how much work was put into it. But *The Donut Song*, a last-second "why not" addition, is the one musical element that people associate with the game. Because it adds so much personality, that song did more for the life span of Splosion Man than any well thought out musical stem idea could. Some people love the song, and an equal amount hate it, but either way, it helps define the game's sense of humor. By making sure our sound designer "Chainsaw" knew that he was a valuable part of the team, he was willing to take risks and go out of his way to make the game bigger and better through music and sound design.

In our current game Comic Jumper, we have a theme song for a villain named Brad. This eponymous song has him singing about how awesome he is. In context this works because it fits the ego of the character, but if we had just thrown a song in for the sake of doing it because *The Donut Song* did well, it would be a disaster. Just randomly throwing stuff in doesn't always work with sound. The personality comes across based on context and how well it fits in with the rest of your world. Games that do this successfully always have a stronger appeal in the personality department.

PERSONALITY IS ENVIRONMENT

» Environments, when handled with care, can have just as much character as any player-controlled entity. Super Mario Galaxy is an example of this sort of unity of level design and environment art. As much as Mario adds to the game (along with all the trimmings that make a Super Mario Bros. game what it is), it could be argued that Mario could be replaced with another character, or even a Mii, and the game would still be just as fun and have just as much personality based solely on what the environments bring to the mix, with their unique gravity, spherical nature, and shifting playing fields. Last year's entry to the Prince of Persia series also did this well, allowing the player to bring the environment back to life through a series of gameplay events. While the levels were typically sparsely populated as far as characters, they were memorable because of layout and the obvious difference between the "dead" and "alive" versions.

PERSONALITY IS WHAT YOU MAKE OF IT

» Personality shows up in every aspect of games. I could go on to show how personality can come from visual effects, writing, programming, and so forth; but the main point is that personality is good. It makes your game stand out, and gives players something to remember. When people mention how much "personality" something has, they don't necessarily mean the main character of your game. Personality can and should come from collaboration between all disciplines. Then you'll really have something to remember!

JOSH BEAR *is the chief creative officer and co-founder of Twisted Pixel Games. Email him at* ***josh@twistedpixelgames.com****.*

Unreal Technology News

by Mark Rein, Epic Games, Inc.

Canadian-born Mark Rein is vice president and co-founder of Epic Games based in Cary, North Carolina.

Epic's Unreal Engine 3 has won Game Developer magazine's Best Engine Front Line Award four times and is also one of the few Hall of Fame inductees.

Epic's internally developed titles include the 2006 Game of the Year "Gears of War" for Xbox 360 and PC; "Unreal Tournament 3" for PC, PlayStation 3 and Xbox 360; "Gears of War 2" for Xbox 360; and "Gears of War 3" for Xbox 360.

Upcoming Epic Attended Events:

GDC Europe
Cologne, Germany
August 16-18, 2010

Gamescom
Cologne, Germany
August 18-22, 2010

GDC Online
Austin, TX
October 5-8, 2010

Please email:
mrein@epicgames.com
for appointments.

TRANSFORMERS: WAR FOR CYBERTRON POWERED BY UNREAL ENGINE 3

High Moon Studios last worked Hollywood magic with Unreal Engine 3 by introducing Jason Bourne to gamers with *The Bourne Conspiracy*.

The developer recently shipped another game based on a blockbuster film franchise. But rather than turning a book-to-film property into a game, they worked with Hasbro to add a new chapter to the Transformers story.

High Moon Studio's *Transformers: War for Cybertron*

The *Transformers: War for Cybertron* game is set on the Transformers home planet before the events seen in the recent Paramount Pictures films.

The Xbox 360, PlayStation 3 and PC versions were created using Unreal Engine 3 by a core team of 68 individuals with a support group that totaled 30.

Matt Tieger, game director at High Moon Studios, said having previously shipped *The Bourne Conspiracy* game on Unreal was a huge advantage, as they were able to hit the ground running from the first day.

"Knowing the pipeline allowed us to work smart and fast," added Tieger. "We were also able to iterate on our internal tools for a second game and leverage the work flow speed that has granted us, especially on the art side. We wanted to continue to leverage what we knew and Unreal Engine 3 is a great engine for us to work with for the type of title *Transformers: War for Cybertron* would become — a third-person action shooter."

High Moon used Unreal Kismet for all of its design scripting and utilized Unreal Matinee sequences for in-game cut scenes and animation vignettes.

The team also took advantage of the UE3 framework for multiplayer and co-op gaming, and the technology helped with the creation of the game across multiple platforms as well.

The game environments were built to accommodate both the hulking Transformers and their hidden vehicles. High Moon worked with Hasbro and used UE3 to bring the fantastical world of Cybertron to life with destructible environments.

High Moon also introduced a lot of firsts with the new Transformers game. Players can create their own Transformers and take them online for head-to-head multiplayer combat. Four character classes and an array of customization options can be used online. Players can also select from a wide variety of weapons and customize their skills loadout, with the ability to level up characters and gain new upgrades and attributes in a deep, rewarding multiplayer experience.

"Gamers are in for something special with multiplayer because the strategies and tactics used are familiar and at the same time unique," said Tieger.

"The foundation of the experience is intuitive controls and gameplay modes, but the tactics change surprisingly when you can transform into a vehicle at a whim. The other key aspect of our multiplayer is that there are several distinct classes to choose from, each with strengths and weaknesses, as well as character growth as you level up each class."

In addition to the online multiplayer combat, High Moon is introducing cooperative gameplay. Never before in a Transformers game have fans been able to play with their friends. The new game offers drop-in and drop-out co-op for the entire campaign game, including all the boss fights.

High Moon Studios has used UE3 to assemble the perfect blend of original action combat for one of the most popular brands out there with *Transformers: War for Cybertron*, allowing fans to explore the home world in an all-new interactive adventure.

Thanks to High Moon Studios for speaking with freelance reporter John Gaudiosi for this story.

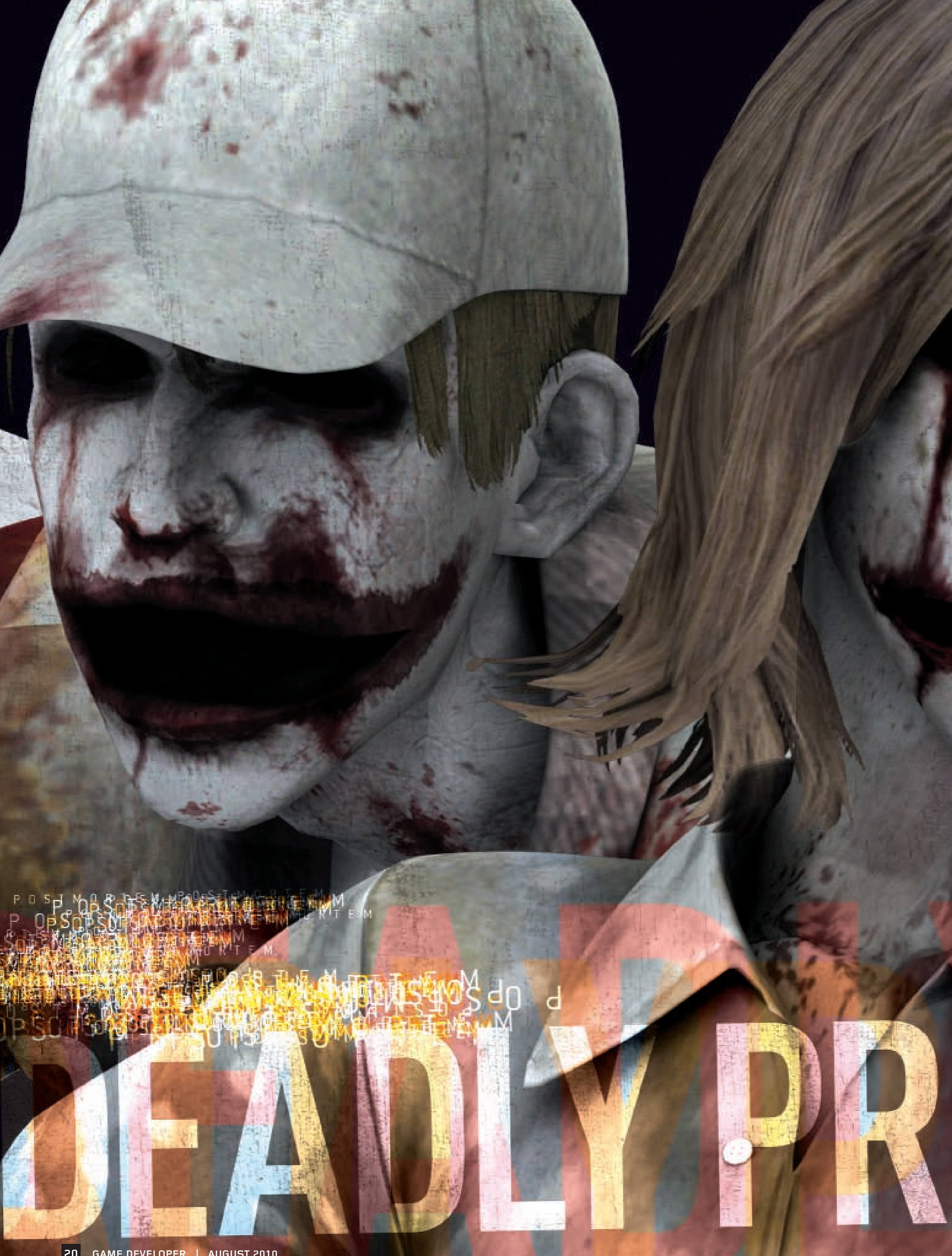
POWERED BY UNREAL TECHNOLOGY

EPIC GAMES

For UE3 licensing inquiries email:
licensing@epicgames.com

For Epic job information visit:
www.epicgames.com/epic_jobs.html

WWW.EPICGAMES.COM



POSTMORTEM

DEADLY PR

This project began in the early fall of 2004. Looking back, it was more than five and a half years ago. After we released SPY FICTION in late 2003, Access Games wasn't lucky enough to have the opportunity to work on another original title, and we wasted nearly a year deciding what our next game would be.

Of course, it's possible we only received the chance to make such an uncommon game because we had so long to charge our batteries. The creativity that built up over so many months was about to explode, looking for an outlet to express itself. Our energy found an explosive release in the form of Marvelous Entertainment, and crystallized into what became DEADLY PREMONITION.

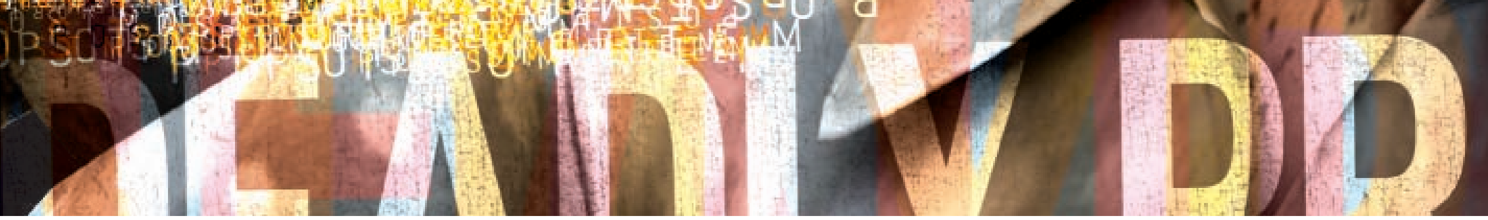
The process was logical, and you might say that we set out to follow a standard development course. But that course turned out to be more precipitous than we could have imagined, and it was, quite literally, a bloody few years. Numerous obstacles stood in our way: unclear next-gen console specs, the decision to go multi-platform, and the threat of project cancellation. It was all we could do to stand up to the hardships and remain steadfast in our implicit faith that the game would eventually be completed.

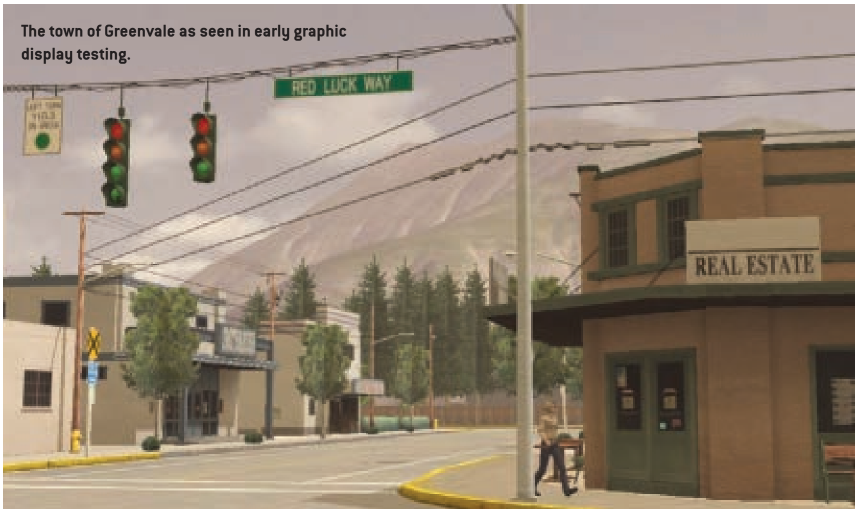
In the end, I learned that true effort is always rewarded. After pushing this project through, we found a gratifying reception waiting for us in a distant, foreign land.

(WHAT WENT RIGHT)

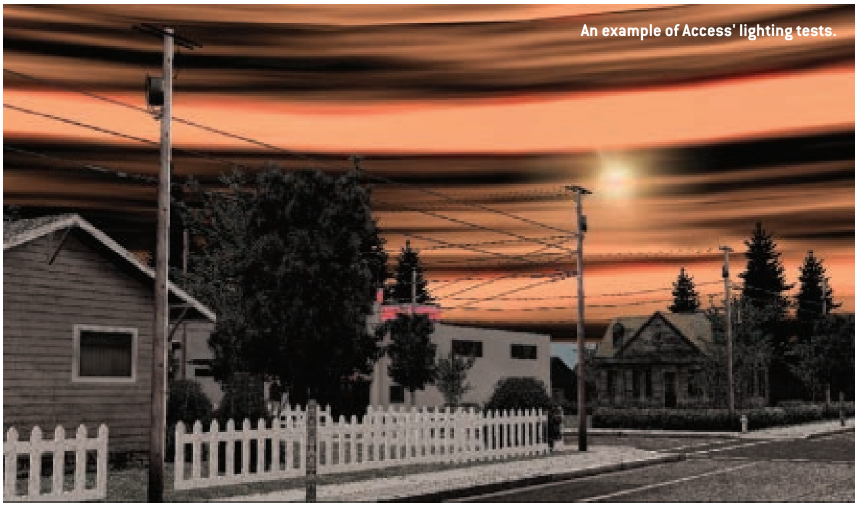
1 | CHARACTER BUILDING AND BACKSTORY (ESPECIALLY YORK AND ZACH) – SWERY (director) & Hitoshi Okamoto (art director) We believe our greatest success in DEADLY PREMONITION was the establishment of our main character. Many different types of protagonists have been created—and loved—in the history of video games, so our primary mission was to devise a

>>>





The town of Greenvale as seen in early graphic display testing.



An example of Access' lighting tests.

type of hero that had never existed before. Deadly Premonition is essentially a mystery, and although quality peripheral characters are crucial in good mysteries, the main character is especially important. I believe players pick up on the fact that while York may appear very handsome, he's a true eccentric inside. York speaks his mind with no regard for the feelings of those around him while constantly muttering to himself ... In other circumstances, it would be no surprise if players grew to dislike him, wondering "Who the hell is Zach?" [York has a second personality, named Zach, living within his mind. There is a strong argument to be made for Zach being "the player."—ed.]

Yet everyone who plays the game seems to love him. Why? Perhaps it's because players recognize that he's extremely charming and reliable—a friend worthy of admiration. Of course, we don't think we gained that recognition for free. There's an important device at work: the invention of the "Zach as Player" relationship. Agent York pursues his murder investigation in Greenvale, the player munches popcorn in his living room, and Zach is the bridge between them.

You might say that the character we were aiming for is established the moment York and the player become true friends, joined through the conduit of Zach. Agent York is a character who only springs into existence with the assistance of the person playing the game ... and that turned out to be exactly the new type of hero we were after.

We knew going in that a character created with the exclusion of the player just wouldn't work. Please show York the same love you've given countless other video game characters before him. That's all we—and he—can ask.

2 | STORY AND WORLDBUILDING (BOTH INITIALLY TOO EXTREME) – SWERY

We spent as much time polishing our storyline and worldview as we did our protagonist. During early development, our setting was much more urban and cynical than the final product. The presentation was also more violent, and decidedly more extreme; I think the one-year gap between original projects may have turned our thoughts toward violence.

Our final world came together only after several iterations, and the overcoming of multiple near-cancellations. For the story, I kept as much sense of reality as possible while embellishing it with a somewhat dreamlike milieu. As a result, I believe the sections depicted as reality have a dreamlike feeling, and likewise, the actual dreams are given a sense of realism—we were able to smear the boundary line. There are very few video games so particular about depicting such things, and I think Deadly Premonition may have become a unique example among them.

A realistic setting was also completely necessary for bringing this story to life. To build it, I posited three "reals:" "real time," "real scale," and "real life." Implementing these with our limited budget and resources proved to be extremely difficult, but despite a great deal of resistance (even from within the team!), I felt these three concepts were essential.

First, to reproduce our five-square-mile town, we visited America for data collection, using measures to determine the width of roads, billboards, and railroad crossings. Second, to depict 24-hour time spans, we calculated weather patterns and the angle of the sun by giving our fictional town a real-world latitude. Finally, we wrote extremely detailed profiles for the townspeople (blood types, birthdays, favorite foods, favorite music, people they disliked, what age they were when they had their first kiss, etc.). While not necessary for the game proper, these were used to create individual 24-hour activity patterns for each character.

For example, when the punctual character Thomas rises in the morning, he goes to the toilet, washes his face, and brushes his teeth. If you have a moment while playing, try spying on his apartment—you'll actually see Thomas performing these actions. Despite not appearing anywhere in the game, we even set the hourly wage for the A&G Diner (about $3.50 per hour, and $25–$30 per day, depending on tips). And, of course, every street in Greenvale has its own name. For a project as rash as this—replicating an entire rural American town from inside Japan—details like these were very important.

These factors layer and combine to form the town of Greenvale, and imbue Deadly Premonition with a sense of reality. This approach is particular to our team, and I wouldn't necessarily recommend it to other developers ... but should you get the chance, why not give it a try?

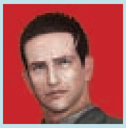
3 | DISTINCTIVE MUSIC – Keiji Teranishi (planner) Music was another focus in Deadly Premonition, and I feel our success in this area was worth the effort. The soundtrack served to further bolster our unique story and world; it may even strike a mood never before heard in games. Our unusual score owes a great deal to our success in helping the composers appreciate the unique

aesthetic behind the game. We showed them countless design documents, played music we thought would fit, and sometimes even hummed song ideas—efforts that resulted in the musicians' understanding of the work.

Doing our utmost to convey the game's aesthetic caused the town of Greenvale to take shape in the minds of our sound team. By the time its citizens began living their daily lives inside our composers' heads, the melodies of Greenvale were created almost automatically ... or so it seemed to me! As proof, I submit "Life Is Beautiful," a memorable whistling tune that conjures up the feel of a peaceful country stroll, and "York And Zach," a song that plays during monologues representing the mysterious image of our hero. Both initial demos were given the okay almost immediately, and both were easily worked into the game. I can think of no other songs that match the worldview of DEADLY PREMONITION so perfectly.

Several other tracks, however, were created only through much trial and error. Particularly difficult were "Red Tree," a theme representing madness heard in the Red Room at the outset of the game, and "Miss Stiletto Heels," Carol's song, which might be considered DEADLY PREMONITION's second theme. We were extremely specific about our orders for these tracks—not only for the arrangements, sound levels, and effects, but also aspects as minute as the timing of hi-hats within single bars! The lunatic improvisational section in "Red Tree" also went through numerous takes with different instruments and arrangements before being judged complete.

YORK'S CHARACTER MODEL SPECS



POLYGONS 13,000

TEXTURES 2048 x 1024 (RGB: Color map, A: Alpha map)
2048 x 1024 (RGB: Normal map)
1024 x 512 (R: Specular diffuse map, G: Specular intensity map, B: Ambient occlusion map)

BONES 141 (29 facial bones, 40 for physics calculations)

LOD Two levels

SPECIAL SHADERS Skin shader/ Eye shader

SHADOW PROCESSING High-resolution shadow processing for the face only

In the end, I believe DEADLY PREMONITION's music succeeded in adding depth to our unique world and characters. If I were a greedy man, I might say that having a few more tracks would have made it even better!

4 | CASTING AND VOICEOVER – Keiji Teranishi I feel that the voice work in DEADLY PREMONITION was even more successful than the music. We spent two weeks recording at WebTone Studios, which was in San Jose at the time (the studio has since relocated to Los Angeles). Over 6,000 total lines needed to be recorded, including pre-loaded and streaming samples. That meant we had to get approximately 500 lines in the can every day. Two weeks may seem like a long time, but Access Games is located in Japan, so it was absolutely imperative we lock down everything required in the time allotted. The pressure was extraordinary.

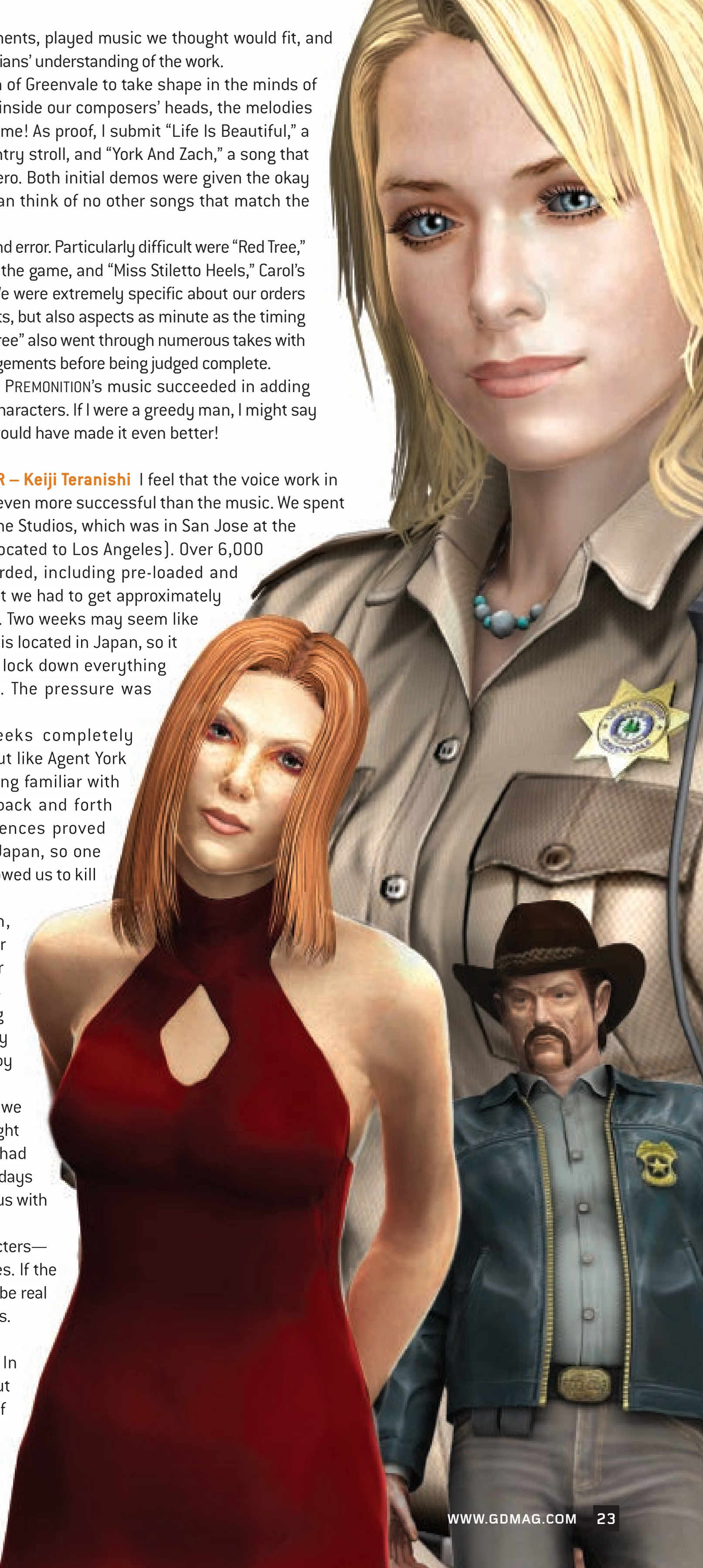
We spent those two weeks completely sequestered in the studio ... but like Agent York himself, we were also becoming familiar with a new town as we shuttled back and forth from our hotel. These experiences proved invaluable after returning to Japan, so one might say the San Jose trip allowed us to kill two birds with one stone.

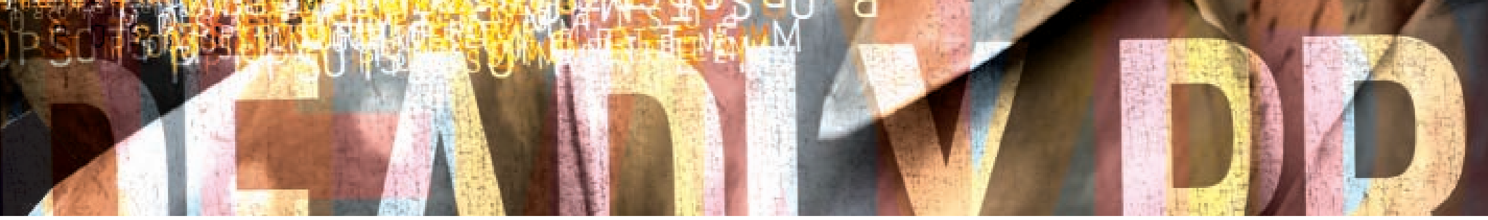
Before recording began, we showed our actors images of the characters to help create their roles. Although we recorded only after explaining in detail other aspects like the general tone, personalities, and family relationships (Greenvale is full of eccentrics with more than a few quirks), directing the performances turned out to be quite a task. When he had difficulty getting his point across, SWERY occasionally gave direct instructions by getting in the booth and performing lines himself.

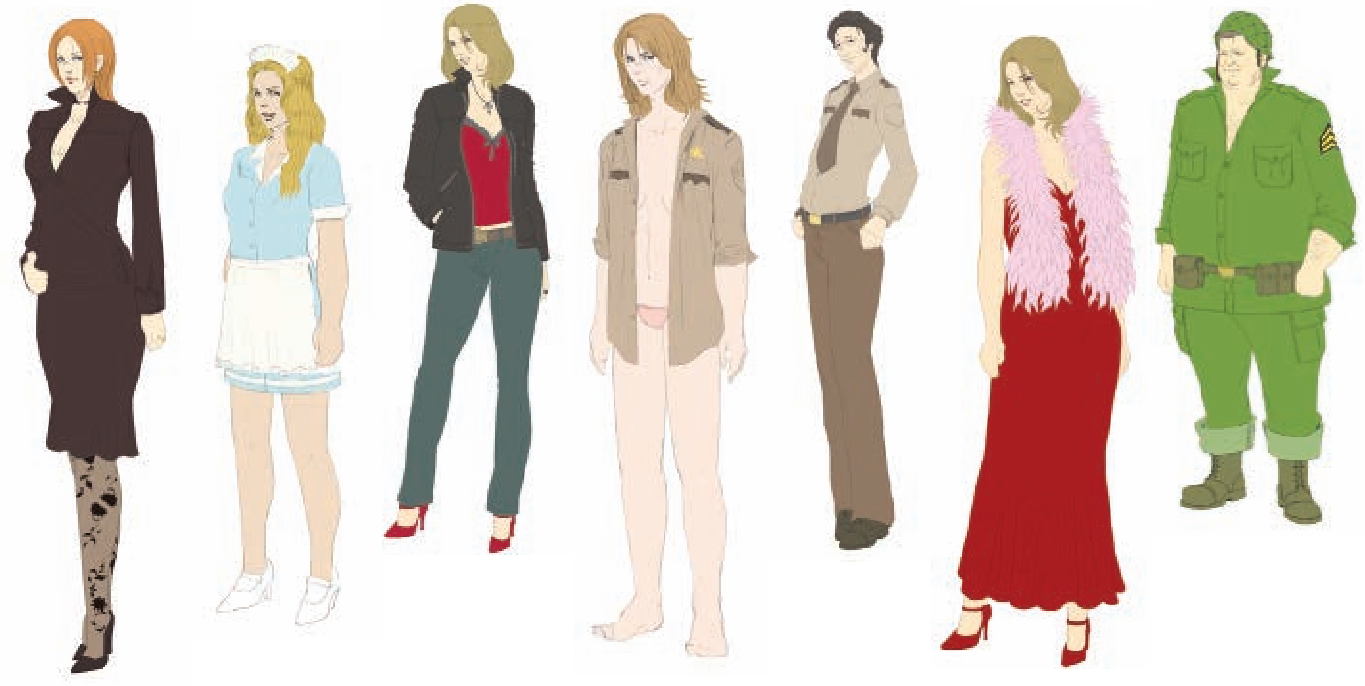
Among all our fine voices, one actor performed his role *exactly* as we imagined ... and perhaps even better: Mr. Jeff Kramer as York. Jeff brought even greater depth and style to the lead role than Access Games had envisioned, for which we are deeply grateful. Despite having only five days to record some 3,000 lines (he was the lead, after all!), Jeff presented us with a truly brilliant performance.

Of course, each of the actors portraying Greenvale's vibrant characters—George, Emily, Thomas and the rest—turned in wonderful performances. If the fans who play DEADLY PREMONITION get a sense that our characters might be real people living their lives somewhere, then our casting was indeed a success.

5 | OUR LOVE AND PASSION FOR DEADLY PREMONITION – SWERY In trying to compile "Five Rights," one might expect some talk about fabulous play control or revolutionary graphics. I can't say either of those were particular successes in DEADLY PREMONITION, so I don't think such topics would be appropriate for our fifth "Right."







GAME DATA

DEVELOPER Access Games

PUBLISHER
Ignition Entertainment, Ltd. (U.S.)
Marvelous Entertainment (Japan)

NUMBER OF DEVELOPERS
Total Staff: 48
Project Scope (Average Number of Staff): 25
Director, Scenario, and Script: 1
Planning and Scripting: 4
Art Director: 1
Object Modeling: 3
Level Modeling: 3
Animation: 3
UI and Effects: 1
Tools and Programming: 7
Sound Director: 1
Composer and Operator: 1
(We received outside staff for localization and voiceovers, and a very small amount of modeling and event scenes were outsourced. Everything else was created in-house.)

BUDGET Not very much

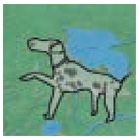
LENGTH OF DEVELOPMENT
Over five years

RELEASE DATE November 1, 2009

DEVELOPMENT TOOLS
Main Tools for Designers: Softimage XSI 4.2, Adobe Photoshop CS
Main Tools for Programmers: Microsoft Visual Studio 2008 Standard, Microsoft Visual SourceSafe 6.0
In-House Tools: Script Editor, Particle Editor, Facial Editor, Object Set Editor, Model Converter, Message Converter, Parameter Converter

After much internal debate, I came to the conclusion that DEADLY PREMONITION's greatest "Right" is the fact that we poured our love into the game. That really says it all, so I'm going to use this space to touch on a few aspects we were particularly obsessive about. I hope this text gives you some idea of how much we cared about our game, and indeed how wasteful some of our efforts may have seemed.

- **Side Missions.** I was insistent that the content of our side missions always draw upon aspects of the NPCs' personalities, thus adding further depth to the world of the game. We built the missions so that players who pursued them might learn more about the personalities and lives of our characters—aspects not revealed in the main story.
- **Observation And York's Monologues.** Examining certain items in DEADLY PREMONITION (products in stores, posters on walls, objects inside NPCs' houses) will reveal messages unrelated to the main story, written from the perspective of York speaking to Zach. We created these messages with an emphasis on the question, "If York were actually investigating this object, what sort of impressions would his peculiar sensibility elicit?" For players who want to get the most out of the game, these insights serve to engender further sympathy with York.
- **Coffee Fortunes.** Coffee is a major factor in the main story, but we also included a feature that allows players to view York's "coffee fortunes" whenever they wished. Like a line in the main story says ("my coffee warned me about it"), we added these interactive "warnings" to bring an extra bit of depth to the concept.
- **Turn Signals And In-Car Navigation.** Although they have no bearing on the game itself, turn signals can be activated on every car. And have you noticed that the car navigation systems are actually functional, and synced to the HUD's automap? Features like these may seem extraneous, but I truly believe they accentuate the game's sense of reality.
- **Conversation While Driving.** DEADLY PREMONITION's game area is five miles square, so a lot of time is spent traveling in cars. What happens on long drives with someone in the passenger seat? Conversation—though not many games have done it. Incorporating the chat feature brought extra realism to these sections of the game. Zach's presence is also given a chance to shine, as he ceases to become a simple internal monologue and actually enables Agent York's conversation with the player.
- **The Greenvale Map.** If you look very closely at the full map of Greenvale displayed in the menu screen, you might notice that the town is shaped like Kaysen's dog, Willie. Willie adds a small accent to the story by appearing with Kaysen, but he is in fact an extremely important character with an intense, unrevealed backstory. We didn't have a chance to expand on Willie in the main plot, so we secretly concealed a hint in the form of the town map for those who are truly perceptive.



(WHAT WENT WRONG)

1 | MEMORY ALLOCATION AND PROCESSING SPEED – Wataru Nishide (lead level artist) & J's Kataoka (lead programmer) DEADLY PREMONITION was our first stab at next-generation development, so we began the project astounded by the vast amount of RAM available—much more than any previous consumer hardware had offered. Astonishment gave rise to overconfidence, and eventually to the worst-case scenario: our data management became sloppy.

Attempting to work on memory allocation in such a state was very dangerous, but we went into production unaware of the risk. The result was a constant struggle with remaining RAM. This was glaringly evident with the motion data in particular—it ended up occupying a truly massive chunk of memory, which led to system restructuring further down the line.

In addition, we set far too many objects in our outdoor scenes, and could not process them all effectively. Dealing with trees and shrubs was especially trying. With that said, cutting too many objects would have reduced the object density of our expansive outdoor map to unacceptable levels, so we had to keep both the optimization of code and the appearance of our world constantly in mind.

Similarly, we had entirely too many points of collision enabled, which became a primary factor in processor bottlenecking. This required scrupulous massaging of collision data right up to the end of the project.

Lighting and shadows (which we will go into more detail about below) had a huge impact on our cycles as well. Our model forced us to do constant optimization throughout all our code, which led to a huge loss of working time.

Many other similar issues can be cited: our water ripple effect, refraction effects, and reflections in mirrors. We do think memory management on this project was an exceptionally useful learning experience, but from the perspective of development, it must be considered a failure. When considering the impact memory allocation has on a game, it goes without saying that this aspect was one of the least successful on the entire project.

2| LIGHTING AND SHADOW PROCESSING —Wataru Nishide & J's Kataoka Since this project allowed the use of pixel shaders, we experimented with various types of lighting—specifically, real-time processes like flat lighting, point lights, and spot lights. Unfortunately, we were a bit overzealous with our lighting calculations which resulted in severe processing bottlenecks.

Even worse, we based our data formats on the assumption that we would be implementing these different lighting techniques as is, so there were absolutely no lighting effects done with textures; this had a huge impact on later code revisions.

Point and spot lights take a heavy toll on graphics rendering, yet brightly lit interiors like the diner had been built with dozens of them. It wasn't just a question of adjusting the positioning or number of lights; the situation escalated to the point that we had to modify our original resources.

Dealing with shadows brought similar hardships. The process of implementing shadows for flat lighting outdoors, point lights indoors, and spot lights used at other specific locations soon became a mighty battle with CPU cycles. A spectacular amount of objects in the vast outdoor areas needed to cast shadows, and because there was little in the way of surface obstructions, the shadows had to appear exceedingly sharp and tight. However, generating such sharp shadows required an enormous amount of VRAM, which simultaneously caused our performance to drop significantly.

A great deal of labor was then expended on repetitive testing and adjustment of various shadowing techniques: perspective shadow maps, light space perspective shadow maps, and cascaded light space perspective shadow maps. In the end, we were not only forced to compromise on the sharp shadows we wanted, but the situation degenerated to the point that we had to modify almost all our resources. Indoor shadows, too, would be processed redundantly several times if an object were lit by multiple light sources, which also led to serious speed drops.

In addition, we faced similar consequences with the self-shadowing on our characters. Self-shadowing can be highly noticeable onscreen, so we strove to render it attractively. Of course, the more high quality our self-shadows became, the more cycles and VRAM were consumed, so we had to work within a delicate balance using jitter textures and the like.

These aspects of game development are commonplace today, but at the time, they were tremendously difficult for us. We simply did not have the proper technical know-how yet.

3| USING THE PHYSICS ENGINE – Yutaka Ohkawa (programming support) At the beginning of development, I was convinced that implementing a physics engine (NVIDIA's PhysX) would allow us to build an exciting, never-before-seen world. It was indeed a great boon to the project, driving the behavior of vehicles, the interaction of objects, and even the fluttering of hair. PhysX did allow us a glimpse into a new world.

However, the use of a physics engine was not all positive. After incorporating too many realistic physics behaviors into the world of our game, we often found ourselves at their mercy. Game worlds require not-infrequent "cheating," but physics behavior is brutally honest. In situations where we had to implement convenient cheats—bending the behavior of a character's prop in a demo scene, or forcing vehicles to navigate properly, for example—physics behavior would conflict.

During development, physics behaviors would often be at odds with what we were expecting, so adjusting the parameters to make things look right was rough indeed. This was especially true for characters, whose physics behavior would change any time their motion data was altered. As a result, even parts of the game that had already been approved would immediately break after motion tweaking. The problems manifested in gameplay, of course, but also in demo scenes; the process of correcting them ended up depleting a tremendous amount of time.

The processing load for physics operations also proved to be much higher than expected, which was another large miscalculation on our part. Because we originally had everything that could possibly be displayed with a physics engine running through PhysX—hair, clothing, even fishing rods—we failed to produce an enjoyable gameplay environment. In our zeal to pursue individually moving objects, I might say we lost sight of the overall processing picture.

To maintain frame rate, it became necessary to optimize; we reduced PhysX actors and adjusted computational loads for each scene individually. However, this made an adversary out of the game's key feature—freedom—as we had to consider an enormous number of variables when making the adjustments. This process was not quick, and when combined with the aforementioned physics behavior tweaks, it consumed even more of our time and energy.

Physics behavior can certainly bring dynamic expression to game worlds, but we learned that it isn't necessary for everything. Determining exactly when to deploy it is of the utmost importance.

4| SOUND EFFECTS AND SURROUND SOUND – Keiji Teranishi As I stated above, the music and voice acting in Deadly Premonition went very well, but I cannot say the same for our sound effects or 5.1 surround sound. With next-generation hardware, we knew sound specs would improve alongside the graphics. The increase in

DEVELOPMENT SCHEDULE

SEPTEMBER, 2004 Initial design document

MARCH, 2005 Project begins

OCTOBER, 2005 Evaluation of prototype

MARCH, 2006 Evaluation of alpha version, Schedule extension due to platform change

AUGUST, 2006 Fatal delay in resource creation, Schedule extension

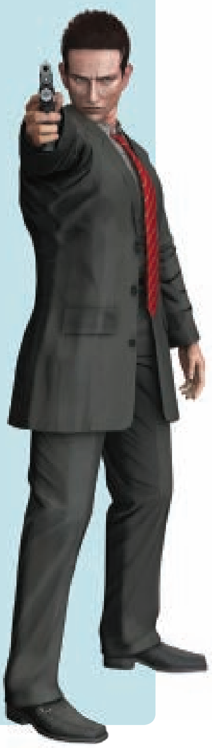
JANUARY, 2007 Resource creation complete

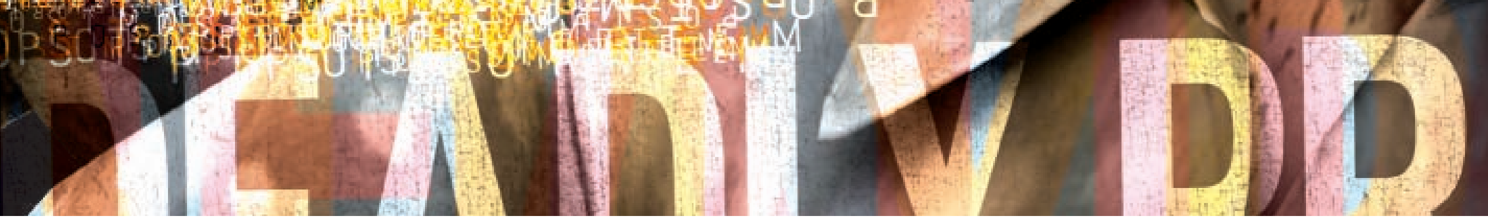
SEPTEMBER, 2007 Trailer shown at Tokyo Game Show

OCTOBER, 2007 Evaluation of beta version: Not approved, Project frozen

FEBRUARY, 2009 Project re-examination, Launch of new project

JANUARY, 2010 Gold master approved: Completion!!





breadth of expression was attractive, but we were incapable of bringing out the full potential of the hardware at the time.

What was the source of this failure? Although there were quality issues with our original sound resources, the biggest problem lies with the fact that Access Games has no sound department. Because we entrust all audio aspects of our games to an outside partner, there is little technical knowledge of sound within the company, which made it tough for us to provide satisfactory quality control. Having turned a blind eye to the problem, we missed the opportunity to try and compensate for the lack of manpower assigned to these issues.

Of course, it goes without saying that many development teams lacking sound departments release games with high quality audio. We should have studied and discussed such games, asking ourselves how we might incorporate their quality sound into our product. Indeed, we should have been closely considering measures to amend our faults. Instead, we put everything into the voice and music, which effectively reduced the priority of the sound effects. This is something I deeply regret.

And why is that? All audio produced by a game is important—music, sound effects, and voice. Only when the three come together as one can we truly speak of a game's "sound."

Looking back, I think something could have been done about the schedule/manpower issue and other problems before they became too unwieldy. Avoiding the creation of problems is certainly an important factor in game development. But this project reminded me that when problems already exist—or loom on the horizon—the ability to work as a team and find the best solutions is even more essential.

5| **SCHEDULE MANAGEMENT AND MOTIVATION – SWERY** I would say that the final and greatest "Wrong" on DEADLY PREMONITION was schedule management. The ideal director is supposed to be able to handle both game quality and project progress, but that was a very difficult proposition for me at the time.

I regret forcing my development team to work longer and harder than was necessary. It may be true that the lack of technical expertise covered above contributed to schedule delays, but as the man surveying the entire scene, the greatest responsibility lies with me.

I have no intention of criticizing a game I obsessed about deeply, and poured my soul into. Those were good things, in fact. However, managing team motivation so that such games can actually be realized is equally important.

On this project, I worked myself to the bone, single-mindedly absorbed in production to the point of blindness, unable to see the people around me—which may be why DEADLY PREMONITION has such a strong sense of "authorship." In the future, however, I want to forge a game production path in which both schedule and motivation management are properly implemented.

I ALMOST FORGOT THE MOST BASIC OF FACTS!

» Somehow we overcame a prolonged production schedule, near-cancellation, and countless other hardships to get DEADLY PREMONITION into gamers' hands. Through the process of producing this game, I learned: never give up.

It sounds so obvious, but the meaning is simple: even though you may believe in what you must do, nothing will come to fruition until you actually do it. DEADLY PREMONITION may not have been a huge financial success, but what may be more important is that it was a work that allowed our staff to grow, strengthen their bonds, and better our relationships with all the companies involved. This is not the goal, but the starting line; now, we embark on our next stage in game development.

York: "*Say, Zach. What do you suppose is waiting for us on the road ahead? No, don't answer that ... Hardships are part and parcel of life. They're what let's appreciate the good times.*

Isn't that right, Zach?"

And that's it. I love you all!

This article was translated by Nick Des Barres.

DEADLY PREMONITION BY THE NUMBERS

[WITH COMMENTARY BY YORK]

568,700
Lines of source code
"Do you have any idea what that figure means, Zach? Neither do I."

263 GB
Total amount of resource data
"The games we played when we were kids weren't even one megabyte! How times have changed."

229,619 words
Number of characters (words) in the script
"Would you say that's a little ... or a lot? I don't know, Zach. Am I a big talker?"

939
Number of times York refers to Zach by name
"Looks like you're pretty important to me, Zach."

8,691
Bugs discovered
"Bug ... that's what they called the lead alien in 'Men In Black.' Too bad those guys don't actually work at the Bureau."

96
Continuous hours spent without sleep
"I have been known to pull some all-nighters during investigations. But things never seem to go very well when I do ..."

4
Times the project was nearly cancelled
"My goodness, Zach ... our investigation was almost cut short."

9,740
Times the director blew his top
"1,948 days in development, five times a day ... This guy has more anger issues than George, Zach!"

1
Times the director cried
"Everyone feels like crying sometimes. Isn't that right, Zach?"

649.33
Times the director shaved
"1,948 days in development, so once every three days ... He must have a light beard. I have to say, from an economical perspective, I'm jealous."

67
Number of movies watched during development
"Why, that's one every three weeks. Now I'm really jealous, Zach."

3,806.75 gallons
Coffee consumed by the team
"Hmm ... 250 cc x 40 cups x 313 days x 4 ½ years = 14,085 liters, divided by 3.7 liters ... These guys almost have me beat, Zach."

338,040
Cigarettes smoked by the team
"So, 40 cigarettes x 6 people x 313 days x 4 ½ years = 338,040 cigarettes. That's fewer than I would have thought. A sign of the times, I suppose?"

4
Turkeys consumed by the team
"That means they celebrated four Christmases together. But you already figured that out, didn't you, Zach?"

2,007 miles
Distance traveled for location scouting (not counting air travel)
"Not much to write home about when compared to our investigation, right Zach?"

TOKYO GAME SHOW 2010

GAME goes to a new chapter.

9/16 [THU] 17 [FRI] 18 [SAT] 19 [SUN]

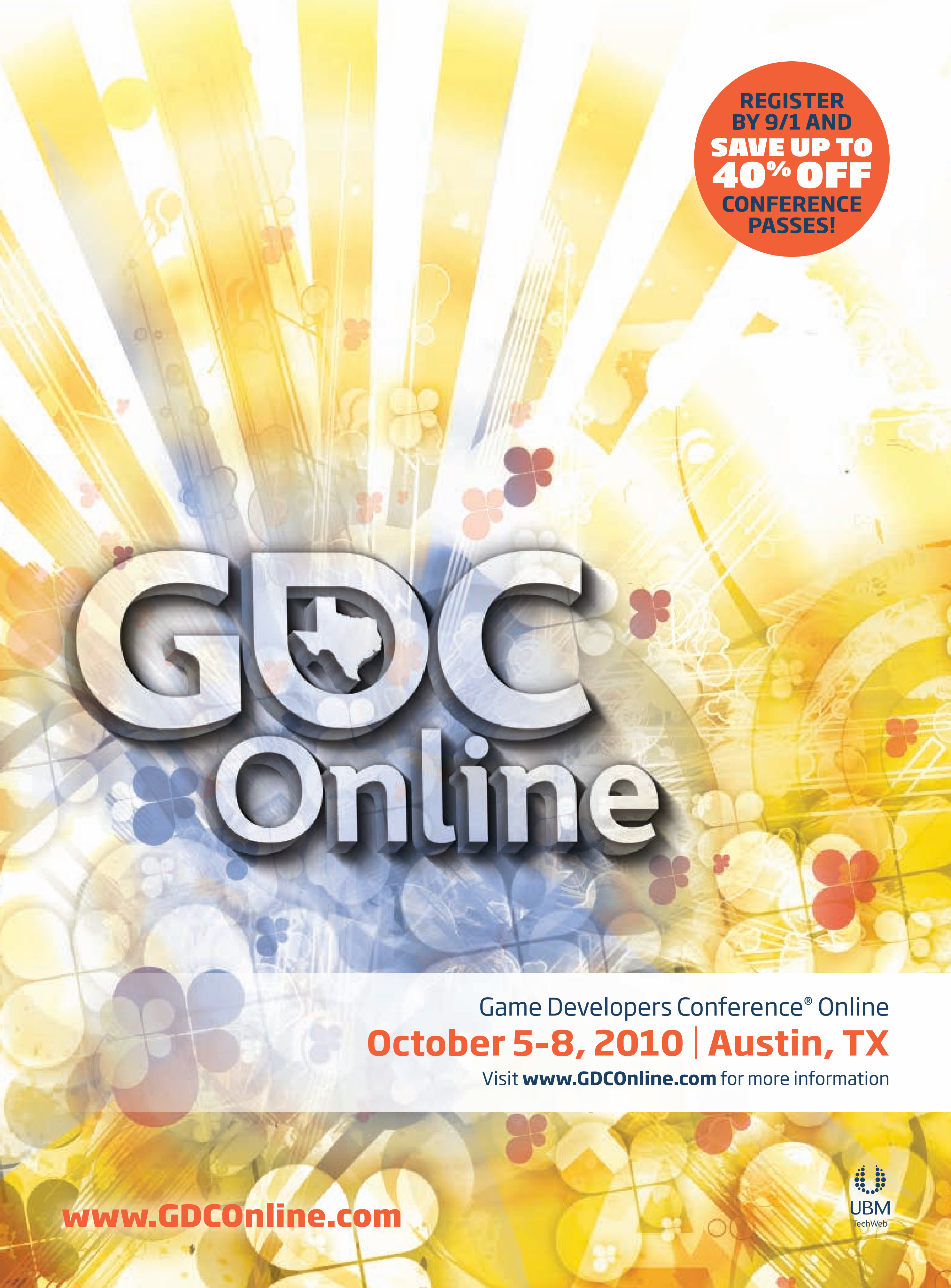
Business Days (16, 17) Open to the Public (18, 19)

http://tgs.cesa.or.jp/english/

Venue: Makuhari Messe

Organizer: Computer Entertainment Supplier's Association(CESA)
Co-organizer: Nikkei Business Publications,Inc.(Nikkei BP)
Supporter: Ministry of Economy, Trade and Industry (METI)

CESA Nikkei Business Publications, Inc. CoFesta 2010



REGISTER BY 9/1 AND **SAVE UP TO 40% OFF** CONFERENCE PASSES!

Game Developers Conference® Online

October 5–8, 2010 | Austin, TX

Visit **www.GDCOnline.com** for more information

www.GDCOnline.com

UBM
TechWeb

BRANDON SHEFFIELD AND CHRIS REMO

AN INTERVIEW WITH ÉRIC CHAHI

IF I SHOULD DIE TODAY

Loneliness, loss, and a shivering sense of possibility. These make up the complex mix of emotions that linger in the player's mind after finishing Éric Chahi's Another World. Working mostly alone, Chahi created a game that was bracingly somber in tone and visually challenging. After Another World's release in 1991, Chahi formed Amazing Studio with a large team of artists and designers to create an ambitious cinematic adventure game called Heart of Darkness. The project took six long years to bring to completion and by the time Heart of Darkness made it to store shelves in 1998, the video game landscape had changed considerably. Although wildly creative, Heart of Darkness seemed out place in the newly emerging world of fast-paced polygonal action and Chahi's studio quietly closed shop.

Chahi walked away from the game industry and little was heard from him until 2004, when a fan-made Game Boy Advance port of Another World renewed interest in Chahi's early masterwork. A year later, Chahi gave his blessing to the GBA project and updated Windows and mobile versions of Another World soon followed. Now returning to games, Chahi spoke with *Game Developer* about his in-progress Project DUST, a real-time terraforming strategy game for Ubisoft. Sitting in on the interview was DUST producer Guillaume Brunier.

BRANDON SHEFFIELD: *What have you been up to in the 10 plus years since you finished Heart of Darkness?*

ÉRIC CHAHI: That's a small question with a big response because ten years is a long time in life, and after Heart of Darkness, which had been a long experience—it took more than six years from initial creation to release—I was exhausted and I wanted to take a step back from video games. I wanted to do something else, so I started to travel. I traveled to some places like deserts, and explored some active volcanos.

Aside from that, I did some painting. I wanted to release something, so I tried painting. I threw paint around and asked "Okay, what does it mean to me?" and then I changed the painting, [and said again] "Okay, what does it mean to me," as an iterative work. And then around 2002–2003, I programmed a tool named Sympheo (www.sympheo.net), which is almost unknown, and not very easy to use, but it's a tool to manipulate sound synthesis. So I programmed this, and then video games started to tempt me again. So I wrote many notes, small ideas, and I tried to connect them together, and all the experiences I had on these volcanos for example, were pushing me. Because when I create, I am creating a lot from feelings. And I felt very, very strong feelings on the volcanos, so that is why we have volcanos in the new game.

From 2004 to 2006, I put all this together, and all of a sudden it came together in one block [in my mind]. I was very happy and I decided to go see Ubisoft, because they're maybe the only big publisher in France and ... well ... I'm not used to English so I preferred to speak to Ubisoft, and I had a feeling that they would be receptive. And they were, but it took time to go to software production. That started in January 2008.

CHRIS REMO: *I've seen that you compared your game to Populous, and that's interesting to me because when I think of what you did in the past, it's a very different thing. Populous is a simulation, you know? Could you explain what the similarities are and why you decided to go down that path?*

ÉC: I love the exploration of new feelings [within games]. I did the same before Another World. I did many games that were unknown, in my youth. I mean, I was searching within myself, but I did many kinds of games—arcade, adventure—but I never did something like a simulation or strategy game. It's something I already wanted to do because I'm passionate about cellular automata, and because of the emergent thing. I just wanted to create a game with something with more organic ... something malleable.

So the thing is, often with this kind of simulation, it is a sandbox but it is sometimes difficult for a [sandbox] game to be challenging. So we really tried to bring together this simulated world, but it's also a challenging world, and that's why the universe of the game is a really intense one. You talk about Populous, and yes there is an influence; I love Populous. And in Populous, there is this ability to manipulate the terrain.

CR: *That's what I was thinking of when you mentioned the volcanoes. Is that related?*

ÉC: Yes, it's almost the same thing because when I saw the earth moving, and the lava flow, which then cools and creates new land, I could see the dynamics of the earth. Usually, with a mountain, it's static, but when you see it exploding, you consider everything differently. So it drives you to play with it. In the game, you can manipulate the terrain by working with ground, water, or lava, so you are kind of terraforming. That is just the basic interaction. Beyond that, there is some challenge because you have to take care of a tribe and their survival in this world, and that is really punishing.

BS: *With your previous popular games, there has been a real focus on this kind of organic and fluid animation and also dynamic gameplay where the same button press in different contexts yields a different action. I can see something similar potentially happening with this game, where you have this very organic world with animation that is automatically deforming. Do you consider this an extension of that style for you? Or is it a different direction?*

ÉC: I think so, because in the game you can take matter, and it's contextual to you ... it depends on where you are. If you are on water, you take water. If you're on ground, you take ground.

GUILLAUME BRUNIER: With the same action? The same button press?
ÉC: Yes, yes.

GB: Do you think you mean it that way? Maybe it's the way you are ...
ÉC: Well, I don't know ... I love simplicity, yes. If it's possible to express many things with simple interaction, I prefer it.

GB: Something interesting you said, which is that it's a simulation ... actually the whole world is based on geological simulation. But the game itself doesn't feel like a simulation because it's simple. I remember, we were doing a lot of presentations internally and on one of the slides, there were three key aspects of the game, and one was brain stimulation. We were working on this presentation with Éric and we wondered, "Should we remove this aspect?" Because when you play the game, you don't feel like you're working and thinking. It's more about feeling things, trying something and if it doesn't work, you try again.

CR: *That's how I think of* ANOTHER WORLD*, that's exactly what it feels like to play. Is the simulation just a setting where things take place as opposed to the point of the game?*
ÉC: Yes, it's true, yes. It's important, but it's just the base matter of the game. Without the simulation, the game would not be the same. And it's a way to express feeling. I talk about tsunamis and volcanoes, but it's not just for show. It's deeper than this because it's a game about the purity of nature and the ambivalence between the beauty of very powerful things and the capacity of destruction that nature has, and how nature is blind, which is also why it's beautiful. And it depends on the context. And so there's no judgment. There's a volcano, and sure, lava sometimes can be really dangerous, but it can be really good to have lava to extend land and to make the earth fertile.

BS: *Nature just is. It's not good or bad. Looking at* PROJECT DUST*, it almost feels like you may have some point, or message, but I wonder if you feel that you have something to communicate, or do you just want to put this scenario out and have people take what they will?*
ÉC: I communicate concepts, not messages. There is another side of gameplay that is very important: the tribe and what happens with this tribe and their quest. It's a quest about knowledge, which is power, indeed, to face nature and to be preserved from some natural aggression. But it's really a place where evolution is stronger. I mean, I could say that the universe of the game is a place where you're looking from above, and it's really dense. The events of this world are very dense, and the lifetime of your men in the game are really short. So you have many generations.

BS: *Interesting, so evolution is happening constantly?*
ÉC: Yes, but it's not a game like CIVILIZATION, where you have an evolution of civilization. This is a world where you have no technology. It's almost a symbolic world. The game, the place of these men, it's more about their memory, their history, and their survival and development in this moving world. It's more about this progression.

GB: In one game session of DUST, you will see your tribe and the individuals of your tribe be born, grow old, and die, and their children will do the same. And you will see maybe, I don't know, in a 20 minute session, you will see a few generations go by.

BS: *Ah yes, so that's why time is compressed.*

ÉC: Yes, it's not an accelerator of time, because you watch it flow normally, but there's a different scale of time change. You can see, as an example, a river being created in five minutes.

BS: *Tell me about your design philosophy and process. Most game developers now pretty much design games based on other games. It's very rare for someone to go out and draw inspiration from the world and from their own creative experience. Most people say, "Well, I'm gonna make a third-person shooter." So*

how does your process work? From inspiration to game ideas, to what ends up being a final product?
ÉC: In general, it's a different influence each time. It can come from a game influence or from other creations or media. For DUST, it comes from my life experience more than anything else.

BS: *So how do you come from that inspiration of being in front of a volcano and seeing the power of it to thinking "Here's how I'm going to communicate this power to other people?" How do you make those steps?*
ÉC: It doesn't happen immediately …

GB: Well he told me it wasn't like he had this creative flash in front of the volcano. It's just part of the experience that you build … progressively, and then you remember the volcano, and then everything comes together, but [snaps] not like this, just an accumulation of different experiences.

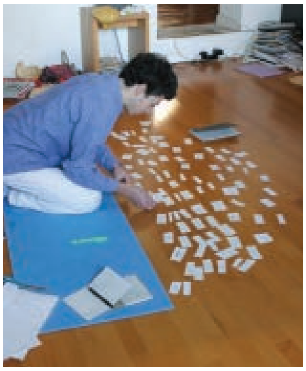
BS: *And then do you do paper prototyping? You said you take a lot of notes.*
ÉC: Yes, notes are very important because it's like food, when you're starting something. Before 2006, when I had to gather this concept together, I thought up all my ideas. And the ideas I thought were cool, I wrote on pieces of paper, and I put them on the ground. Then I grouped ideas together and said, "Okay. This can be a game, this can be a game, this can be a game." After all that, I continued to build on these ideas, and when I felt stuck, I merged two groups of ideas together, and that became Project Dust. But I could be stuck for a long time, and then everything would move quickly, and I'd go, "Ah yes—it's good, it's bad," really quickly.

But the interesting thing is that at the time, I had to choose which project to work on, and there was another project which was very different, and I decided not to start it. Now maybe I'll never do it, I don't know, because people change. And the question I asked myself is "Well, if I should die today, which game do I really want to do?" And I said, "Okay, this one."

GB: If I can add something to this, on the creative process, which Éric just described, it's very, very exciting to work with Éric on such a project, but at the same time, it's quite scary. It took us quite some time to be able to explain what we were doing. And I remember maybe a year ago, people at Ubisoft were saying, "Okay, tell me what your game is in a few words," and we're like, "We don't know! We don't know yet, so we can give you some ideas, bits of ideas … " But I think it comes from the fact that with this game, we didn't say, "Okay, it's gonna target this and this market." No, it's just coming out of the blue.

ÉC: That's why when I have to describe a game in two words, I say, "It's a god game." It's a kind of god game because that's the closest thing, but it's not fair to say it's just a god game, and it's not a clone of Populous. It's just to help us explain it.

BS: *I think that kind of sentiment is part of why people appreciate your games, because many games feel like they're made by a bunch of people and it's just kind of a very generic vision, but in every game that you make, it's very obvious that there is somebody at the helm of this project that is driving it. There's like a captain of the ship, and you can tell who it is, and I think that's really important artistically.*



ÉC: In a team, it's not easy because you have to sometimes let go of ideas. Because if you want it to work in a team, you have to let people have their own ideas. Sometimes I may have to abandon some idea because the team doesn't get it. And if the team doesn't get it [it won't work]. The team has to feel it, some people have to feel it. That's what you see in the illustrations we have, which are done by Bruno Gentile and Sebastien du Jeu. They already feel the game.

GB: The fact that we don't have references forces us to have a team that really works well together. And when you really feel the direction of the game and take part in it … when you're working on a sequel or whatever, if people don't really buy in, it's not a big deal because they can copy [the previous one] and just do better. But on our side, since we don't have references, everybody is putting a bit of themselves into it, and we're very lucky to have a team that's communicating well and understanding Éric's vision and proposing stuff to make it richer. I mean, it's not always easy, right? Sometimes we get in fights, but …

CR: *How big is this team?*
GB: Seventeen.

CR: *Do you find that, over time, it's become easier to communicate that vision? Does it seem like the more you work on this, the more the whole team understands the whole concept?*
GB: It's never easy. But we're progressing, yes.

BS: *It's hard to make someone understand an idea when you're not completely sure what it is yet.*
ÉC: I'm going to tell you why it's not easy. Because there is no bad idea. As an example, an idea can be very cool but it's not in the right direction. So it's difficult because … okay, it's good, but it's not exactly right. And sometimes that's difficult to explain.

BS: *Many people have that dream of being able to do their own projects, their own ideas, and they don't know how. And sometimes I think it's because they don't have the right inspiration. So it's good to hear about someone who can take external, non-video game input and create something powerful from it.*
GB: It's a fight. It's an everyday fight to promote these kind of ideas, this kind of game, because it's easy for people to say, "Ah, no. It's not gonna work."

ÉC: It's easy to go to a standard thing. I mean, it would be so easy to do a fighting game. It's a challenge for me. I have to admit that sometimes I like shooters, because they're good games and all, but I think that there is so much else we could do.

BS: *How much time did you have to spend on the technology side for this? Because it seems quite complicated.*
ÉC: Yes. Two years of work. It's Ronan Bel who coded the simulation especially for this processor in the SPUs. He fought the architecture coding. It's why we can have this real-time simulation.

BS: *When you are working with a sandbox design, how do you make it so that the player has something to identify with? Obviously, there's the tribe, but still, when you are above everything, how do you really connect?*
ÉC: You mean, connect emotionally?

BS: *Yes, to care about the world.*
ÉC: Curiously, it's easier to connect to the land than to the men. The player changes the land, and the simulation takes over. The land evolves, the world evolves, because of what you do to it. Because of events that happen in the world, it becomes your island. That is the surprising thing we discovered after playing the simulation.

GB: One of the best compliments we had when showing the game internally was one day when we sent a version outside of the team, we got a phone call from some guy playing and he said "Hey, how can I save my island?" At the time, we didn't have any save process. When the guy said "my island," when we heard that, Éric and I, were like, "Good. We've reached something."

Because the guy spent maybe 30 minutes, you know, modeling the world, and he took ownership of it emotionally. It's very difficult to convince people that you don't need a hero, like a physical hero, to be attached emotionally to the game. We have the tribe on our side, and we think that people will like them and want to help them, but in the end, the hero of Dust is the world, and you will be attached to the world because you play with it. The fact that people say "my island," for us, that's a really big joy.

{ADVERTISEMENT}

MAKE AN IMPACT JOIN OUR SQUAD

FORGE THE FUTURE OF GAMING'S BIGGEST ACTION SHOOTER FRANCHISE

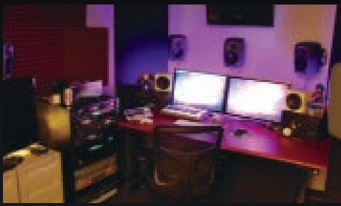
Sledgehammer Games was founded in 2009 by industry veterans Glen Schofield and Michael Condrey and is the newest addition to the Activision family of game development studios. With the exciting announcement of the new studio, Sledgehammer Games immediately generated attention across the industry as the news fueled speculation on the studio's future. Sledgehammer Games recently launched their new website, *www.sledgehammergames.com*, and announced their next title would be for Call of Duty®, the industry's most successful first person action franchise of all time.

"Our studio is made up of the industry's best and brightest talent," said Sledgehammer Games GM Glen Schofield, and creator of Dead Space, 2008's Action Game of the Year. "We are actively looking for top tier talent as we staff for a title in arguably the hottest franchise in games: Call of Duty. The team is proud and pumped to be a part of this franchise. We look forward to creating games that fans will embrace."

Sledgehammer Games also recently unveiled their brand new studio facility in Foster City, located in the heart of San Francisco's Silicon Valley. The modern design and industrial layout of the studio was built to foster the collaborative energy that is at the core of the studio values. "We are happy to announce that we just moved in to the new space," Schofield said. "We're located in a new building in Foster City with views of the Bay and San Francisco. The studio was designed to be state-of-the-art, with equipment and facilities to match. It's a beautiful location, and a space that is set up to be very collaborative and fun, with an open and inviting atmosphere."

Michael Condrey, VP of Development and Studio COO, agreed. "The team, studio space, and opportunity to work on a Call of Duty first person action title are all world class. It's an extremely exciting time for Sledgehammer Games right now."

"We're actively recruiting for top caliber talent," Condrey continues. "We're building a studio that not only delivers exceptional work, but that also feels like a family. We have built a foundation of a team with exceptionally talented individuals, and we established, from the beginning, that our studio is built on the values of teamwork, collaboration, and open and honest dialogue."



Sledgehammer Games is actively recruiting across all disciplines to be a part of their Call of Duty development team. "We work hard and play hard, and strive for nothing less than excellence in everything we do," said Schofield.

The proof is in the results, and it doesn't take long to watch the team in action and know that Sledgehammer Games is dedicated to delivering something special. To learn more about job openings at Sledgehammer Games and how to apply to be part of this amazing opportunity, visit their brand new website: *www.sledgehammergames.com.*

VISIT **SLEDGEHAMMERGAMES.COM/CAREERS** FOR MORE INFORMATION

CALL OF DUTY®

SLEDGEHAMMER GAMES™

REVIEW BY MATTHEW BURNS

UNITY TECHNOLOGIES
UNITY PRO 3

THE UNITY ENGINE'S EXPLOSIVE GROWTH OVER THE LAST FEW YEARS IS NOTHING SHORT OF IMPRESSIVE. This all-in-one game development tool was first released in 2005, and has since become the engine of choice for many game developers across a wide range of experience levels and budgets. Now, Unity Technologies aims to build on that momentum with Unity 3, which adds new platform support, a slew of advanced graphical features, and other improvements, while also addressing some long-standing user requests. The result is a remarkably strong solution for game development that balances ease of use with flexibility and power.

THE BASICS

>> As in previous versions, Unity 3 comes in two basic flavors: a free entry-level variant and the more fully featured $1,500 per-seat "Pro" version. For the purposes of this review, we'll be looking mainly at Unity Pro, which benefits most from the new features, though it should be said that the free version is a complete product in its own right and a good place to start for budget-conscious indies and hobbyists.

If you're not already familiar with it, Unity's interface should be a quick study for anyone who has worked in a 3D modeling program or level editor. The main components are a scene view, a game/camera view, an object hierarchy, and an object inspector. Layouts of these and other windows are customizable and savable, allowing you to find the optimal arrangement for your work style.

Scripting is handled via the open source Mono implementation of the .NET framework, allowing your choice of C#, JavaScript, and the Python-like Boo—or if you want, a combination of the three—to handle game logic. Each compiles into native byte code. If you need to integrate something written in C or C++, Unity Pro will allow you to access functions exposed in a DLL or OS X Bundle as a plug-in to your project. Unity's own source code is not included in the free or Pro versions, but the company does have source-inclusive licenses available through its sales team.

THE NEW HOTNESS

>> No doubt the flashiest upgrades in Unity 3 come in the form of its new graphics features, which are quite advanced for an engine in its price range. For starters, there's a completely revamped lightmap system, now powered by Illuminate Labs' well-known Beast software. This tool creates beautiful, high-quality lightmaps and offers important options like settable per-object resolution, and can fade to real-time lighting with bounce from lightmaps as the camera approaches.

(From top) CIRCUIT DEFENDERS, GLOBAL CONFLICT: PALESTINE, GOOBALL, and FEIST all use the Unity Pro engine.

There are also a handful of new built-in effects to play with such as crepuscular sun rays, depth of field, and outline shading, making a variety of visual options as easy as dragging and dropping their scripts onto your camera. The particle system can now alter the rotation of particles on an individual basis, which is useful for smoke and haze effects. And two new under-the-hood features push Unity 3 even further into "triple-A" territory: an integrated occlusion culling tool from Umbra Software, and deferred lighting, which sharply reduces the calculation load of dynamic lights.

Unfortunately, there's still no visual shader graph editor, which makes the creation of brand new shaders a text-based affair, and more technical than some artists might be comfortable with. This is mitigated by the comprehensive selection of shaders that ships with the engine, though. Chances are you'll find something close to what you need.

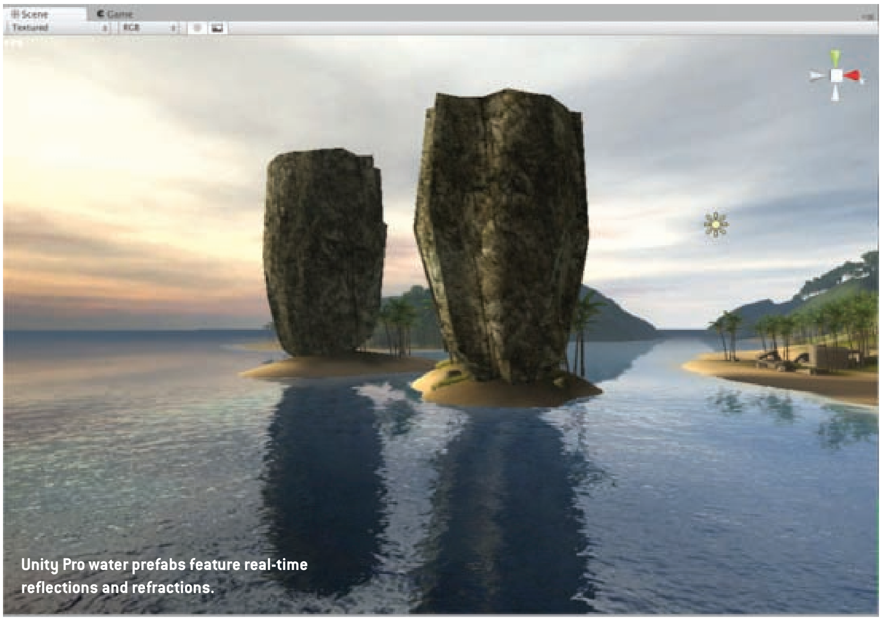
In the sound department, Unity gains audio filters, which work much like the visual effects do. You can add one to an object with an audio source, and the filter will apply it to all audio sources on that object. Add one to an object with an audio listener and the filter will apply to everything that listener can hear. The choices are simple but useful (high pass, low pass, echo, distortion, chorus, and reverb) and make it easy to create real-time effects, such as over-the-radio voice crackle or a muted sound to help indicate low health situations. Additionally, a new "reverb zone" object allows designers to associate individual reverb settings with specific areas, which can enhance indoor/outdoor and other environmental transitions.

Finally, programmers and scripters will be happy to know that serious debugging is now possible in Unity 3 thanks to the included MonoDevelop IDE, which allows you to set breakpoints, single-step through code, and check variable values. Anyone who has felt lost inside a poorly articulated scripting language haphazardly bolted to a game engine will appreciate the dedication to syntactical rigor and clear definition that the Unity team has pursued, and this addition makes the programming environment much stronger.

NON-ALIGNMENT

>> Another important addition to the Unity 3 engine is support for Xbox 360, PlayStation 3, and Android as target platforms. Combined with all the platforms that it could build to before—Windows, Mac OS X, iPhone, iPad, the Wii, and its own browser plug-in—these new deployment options make Unity one of the most universal all-in-one engines available today. Although it should be noted that the console-capable versions are separate licenses that must be acquired from the company along with development kits for those platforms, and the iPhone/Android product is also an additional purchase on top of Unity Pro 3.

These pathways enable game creators to approach development in a more platform-agnostic manner,



Unity Pro water prefabs feature real-time reflections and refractions.

UNITY VS UNITY PRO

Unity Pro offers the following features, which are unavailable in the free version of Unity:

- Render to texture effects
- Full-screen post-processing effects
- Real-time shadows
- Profiler
- Low-level rendering access
- External version control support
- C/C++/Objective-C plug-ins support
- Video playback and streaming
- Fully fledged streaming with asset bundles
- Script access to asset pipeline
- No Unity splash screen or watermark
- May be licensed and used by companies or incorporated entities that had a turnover in excess of $100,000 in their last fiscal year

no matter how deep or shallow their technical resources. This is an exciting prospect; you could put a free demo version of your PC game on your company website for anyone to try via the Unity Web Player with relatively little rework. Or you could enter into discussions to bring your game to one of the console download services with a certain degree of confidence in the porting process before it ever begins.

PUTTING IT ALL TOGETHER

» Day-to-day work in Unity is smooth and enjoyable. The editor was designed from the start to keep iteration in as tight a loop as possible, and it shows: you can modify values in the inspector even while the game is running to test variations immediately (though these edits revert when the game has stopped). Many simple operations, such as instancing prefabs into a scene or associating scripts with game objects, can be performed by dragging and dropping.

Several of the interface enhancements in Unity 3 are a godsend if you worked with complex levels and projects in previous versions. A new asset browser window allows for quick previewing and a selection of large numbers of models and textures. There's also a tagging system which lets you categorize and search your assets easily provided you're disciplined enough to label everything rigorously. The ability to quick search per scene eliminates hunting around for a specific desired object.

Unity also has one of the most friendly asset pipelines of any engine, with wide-ranging format support and synchronization with source files. All it takes is saving a new version of a model in Maya or texture in Photoshop for the asset to be automatically updated, keeping the same settings as before. The only potential downside to this is that the asset metadata format is proprietary and closed. You shouldn't go mucking about in the Library folder on your own, for example, because if that data were to get corrupted somehow, you might not be able to fix it so easily.

Another add-on product, the Unity Asset Server, offers version control and binary file storage to Unity projects and helps coordinate effort among teams. While Unity works fairly well with other version control software such as Perforce or the free Subversion, Asset Server benefits from its close integration with the rest of the Unity platform and doesn't require any special configuration. If you can spring for the $500 additional per seat that it costs (on top of Unity Pro), and don't mind giving up some of Perforce's advanced features or svn's deep command-line control, it is a decent choice for hosting Unity-based projects.

UNITY COMMUNITY

» Another factor that shouldn't be ignored is Unity's user base, which is active and engaged on the official website and a number of independent forums, each of which caters to its own specific community. There's a good selection of user-authored references and tutorials online, along with growing libraries of free scripts, shaders, and plug-ins. Unity-themed conferences and user groups are sprouting up across the world, where you can share tips and tricks with fellow developers in person.

This kind of support is especially important if you're an indie or otherwise struggling to get your start in game development. The increased chances that you'll get some real help from someone—as opposed to the "is anybody out there?" problem you sometimes encounter with questions about other tools—can be a selling point in and of itself.

For more commercially oriented developers, the Unity ecosystem also brings benefits; a selection of add-on packages are sold for the platform, often written by other developers who created the tools while working on their own games. These naturally extend Unity in directions that you might find wanting in the base product: one popular package adds support for visually creating GUI layouts, for example, while another provides functions useful for creating two-dimensional, sprite-based games.

Finally, Unity Technologies itself remains responsive to the

CONTINUED ON PAGE 36

Unity Technologies UNITY PRO 3

http://unity3d.com

¤ **PRICE** $1,500

¤ **SYSTEM REQUIREMENTS**
Windows XP SP2 or later; Mac OS X "Tiger" 10.4 or later
Graphics card with 64 MB of VRAM and pixel shaders or 4 texture units

¤ **PROS**
1 Great balance between intuitive ease of use and open-ended power and flexibility
2 Most major gaming platforms now supported
3 Entry-level version is completely free

¤ **CONS**
1 No visual shader graph or visual GUI editing
2 Metadata format is closed and proprietary
3 Console and mobile support comes at an additional cost

product news

MINDSET SDK ALLOWS FOR EYE BLINK, BRAINWAVE-POWERED GAMES

NeuroSky
www.neurosky.com

NeuroSky announced upgrade v2.1 for its Mindset SDK, which includes eye blink detection in addition to the tool's brainwave measuring features.

The Mindset interface detects electrical impulses to measure brain activity, and can translate certain mental states and brain patterns into digital input for use in games and applications. The tool's SDK allows developers to use unfiltered brainwave measurements for more complicated programs. NeuroSky hopes the inclusion of blink detection will further enhance this interface for use in games and other software. Several applications and games for the Mindset interface are available for download, and the tool was used in several children's toys in 2009.

HANSOFT 6.1 SUPPORTS KANBAN, MULTIPLE WORKFLOWS

Hansoft
www.hansoft.se

Pipeline management tools developer Hansoft has released the latest version of its eponymous solution, Hansoft 6.1, adding support for the Kanban development technique and the ability to handle multiple parallel workflows. The company says Hansoft 6.1 has also improved release and sprint burndowns by offering graphical metrics for milestones and "ideal" days. It also says it's extended XML data importing and exporting, all with the aim of increasing visibility on a project's process and easing long-term planning.

Among the other new features in Hansoft 6.1 is further support for spreadsheet functionality, whereby all information from the tool can be exported to the Excel format. The XML data import-export also includes support for custom columns so that developers have more than one option for migration from other tools. Hansoft says it's also implemented performance enhancements, promising "virtually no" CPU or context-switching when idle, and a higher frame rate for rendering the wall view and displaying cards on-screen.

OPENFEINT SDK COMING TO ANDROID

Aurora Feint
www.openfeint.com

Aurora Feint's OpenFeint platform for virtual goods and social gaming features in mobile games, previously only available on iPhone and iPad, will soon be available for Android game developers. Promising a "complete solution" for Android, OpenFeint will launch later this summer and will include its standard SDK, a "game discovery store," and mobile payment options. OpenFeint for Android will incorporate Google Checkout and relevant community features, as it extends its Game Spotlight discovery app to Android as well.

The Android expansion comes thanks to the backing of Chinese game operator The9's recent investment. Aurora Feint says it will continue to roll out a cross-platform strategy, part of which is serving the 160,000 Android-ready devices that ship daily. The company says OpenFeint will continue to be open source and free for developers and players.

FMOD DESIGNER 2010

Firelight Technologies
www.fmod.org

Audio middleware company Firelight Technologies released FMOD Designer 2010 for Windows and Mac, which offers tools for developers to create and edit game audio. The new version of the FMOD Designer software includes integration with the Unreal engine, the ability to edit multiple audio files at once and use multi-layered sound effects, built in DSP effects, and the ability to implement branching music composition. In addition, the tool also allows developers to mix their music within a game via connectivity to Xbox 360 and PlayStation 3. Performance counters are available during live mixing to monitor performance, audio levels, and resource use.

DIGIMI GAME DEVELOPMENT KIT

DAZ 3D-Gizmoz
www.daz3d.com

DAZ 3D-Gizmoz, the company behind the 3D modeling software DAZ Studio, released the Digimi Game Developer Kit, which allows developers to import models from DAZ studio into a wide range of game engines. The tool also includes features designed specifically for game development such as the ability to reduce a model's polygon count, and to combine, scale, or convert textures into a single map. Animations, morphs, skeletons, cameras, and weightmaps created in DAZ Studio can be imported to game engines along with the 3D models.

MIXAMO SUPPORTS HUMANIK RIGS IN AUTODESK MAYA

Mixamo
www.mixamo.com

Mixamo announced its online animation service now supports Full Body Inverse Kinematics (FBIK) control rigs for Autodesk Maya 2011. Maya users can import animations from Mixamo's online database and use them on FBIK control rigs. This new integration preserves the original control rig, allowing for key frame adjustments to imported animations. Mixamo users can upload rigged models to the service and select and customize animations before importing the animations into Maya. Mixamo can be used with Autodesk Maya 2008 or later.

UNITY PRO 3

CONTINUED FROM PAGE 35

concerns and desires of its user base, and operates a poll system on its website where its users can vote for specific desired features. Each suggestion is well-discussed and several of the top ones have made their way into subsequent releases.

THE SWEET SPOT

» There's much more to Unity that I didn't really have the space to cover fully in this review. An animation editor allows you to create and save simple curves; these curves can in turn be hooked up to any exposed variable of materials and components—it could drive a dynamic light's intensity, the pitch of an audio source, or whatever variables you have defined in your own scripts. There's easily implemented physics powered by NVIDIA PhysX technology, complete with breakable joints, wheel colliders for vehicles, and (new to Unity 3) a cloth simulation. Built-in networking functions give you a good place to start for adding online multiplayer. And a convenient performance profiler tool, introduced in 2.6, speeds up the optimization process.

I could go on, but the best part is that as you begin to absorb the nuances of each individual system, the possibilities do not dwindle with familiarity; instead, they seem to grow exponentially. When taken all together, Unity is more than powerful enough to displace the often dinky-feeling tools that have occupied the lower end of the market, but it's also much simpler and easier to use than the expensive top-line engines that sometimes seem to want to run themselves first and your game second.

Unity Pro 3 offers game developers with all levels of experience a flexible and well-thought-out platform in which to realize ideas. It's little wonder that this engine, which seemed to come out of nowhere just a few short years ago, is so widespread today. If you have not yet evaluated Unity, now is the time to do so.

MATTHEW BURNS *is the founder of Shadegrown Games and formerly a producer at Bungie where he worked on the* HALO *series. Email him at* ***mburns@shadegrowngames.com****.*

DOWNLOADABLE HERO

SCANNING LARGE DLC LIBRARIES

ILLUSTRATION BY RYAN LESSER

DOWNLOADABLE CONTENT IS AN INCREASINGLY IMPORTANT PART OF video games of all genres, as developers look for ways to extend the play experience for their audience. The music game genre, though probably not the first to offer downloadable content, is definitely pushing the idea of a la carte content in new and sometimes dramatic directions. Working at Neversoft Entertainment on the Guitar Hero series of titles, I've watched our games move from offering a few songs a year, to full albums, and now even the entire catalog of previous games as downloadable content. Through this evolution, we've had to come up with solutions to handle this expansion. The most interesting and complex of these problems had to do with scanning vast amounts of content. We currently handle the scanning (in the worst case) of over 1,400 packages and over 17,000 files, and these numbers grow year after year. The thing with downloadable content scanning is that no one notices when it's working well, but you catch hell when it takes ages to complete.

SCANTASTIC

» When we first started doing downloadable content on Guitar Hero III: Legends of Rock, we had to broach the task of scanning what content a player had available to them. Initially, we tackled this in a way that was similar to most every other studio doing downloadable content. We would loop over all folders for our title, and then for each folder, loop over all of the files contained in it. We'd save those results off to a table, as diagramed in Figure 1.

This worked perfectly well. We had about 30 packages maximum, and it took just a handful of seconds in the worst case. Things started to change a year later when we were close to finishing development on Guitar Hero World Tour. It had been decided that Metallica's then-pending *Death Magnetic* album (which was also DLC for our previous game) would need to be compatible with our new title. This would be the start of a push to allow users to bring their music with them from one game to the next. With around 250 singles released yearly, it amounted to an incredibly large library of songs.

Guitar Hero 5 was our first game to support downloadable content packages containing all the content from a previous title—some 50–80 songs—as well as the normal year's downloadable content releases.

We found we had a problem. Our scans were taking far too long in common cases, let alone in the very worst case where the player has purchased every conceivable package. Scanning enough packages for 175 songs was taking some 30 seconds to complete. Worst of all, the poor performance of the scan was hurting our best customers the most! No one wants to sit on a scanning screen for 30 seconds or more every time they start up a game or every time there is a sign-in change. Fairly late in the project, it was decided we couldn't ship our title in its current state. We had to do something to drastically reduce the time it took to scan downloadable content. But how do you speed up scanning over 1,000 folders and 14,000 files?

We started by collecting information on where things were bottlenecking. On the PS3 these operations are faster, so we focused our tests on the Xbox 360. The time it takes to do these operations on either platform is still long enough to benefit from a better solution than a linear scan. For simplicity's sake, I'll only discuss timings for the Xbox 360.

The first thing we realized was that no matter what, we would need to scan all the content folders to see which ones the player had available. Since we always have to scan over all folders, there is no way to speed this process up, so we turned our focus to the inner loop of Figure 1—identifying the files within each content folder. This is where we were surprised. On the Xbox, we found that the first file to be opened took on average 70ms, and up to 300ms, but all subsequent files on average took 16.5ms to be scanned. We're not entirely sure why the first file is particularly bad, but we suspect that when opening a content folder, there is an expensive file system operation going on, as well as network communication with Microsoft to verify DRM. Add that up over our absolute worst case of 14,000 files and you're looking at about 4 minutes to scan just the files! With this new information, our search turned to finding a way to cut down on the number of files we scanned. After some late nights and several brainstorming sessions, we had a viable solution.

BINARY CHOP SHOP

» We borrowed some ideas from simulated annealing and binary search. Our problem of course isn't analogous to binary search; our data set isn't sorted, and we're not searching for an individual item, but we wanted a similar behavior to binary search where we could eliminate large chunks of our data set with each step. We needed to pick a starting point though, and since our data set is not sorted and the names tell us nothing about order, we cannot just choose a median point like in a binary search. Instead, we took an idea from simulated annealing that repeatedly chooses points at random and then walks our dataset greedily, which can be an effective heuristic for exploring the entire data set. We called our technique Binary Chop.

Since scanning for folders needs to be done no matter what, we do that up front, creating a list of the folders found for the set of signed-in users. The real gains are made in how we scan for files within each of those folders, and this is the key to our speed up.

In this Binary Chop, each package has a manifest file that specifies the contents of each package that came before it. This means the manifest for package 5 contains information on what files are in packages 1, 2, 3, 4, and 5. This is key, as it allows us to resolve large chunks of folders at a time by just opening a single file in a single package. We name the manifest files based on the content folder they exist in so that we can generate the file name at run time, and then blindly open the file assuming it is there.

As shown in Figure 2, a package is chosen at random from a pool of unresolved packages, and its manifest file is opened. As you can see in Figure 3, we then use the data contained in the manifest to resolve all the folders that were released before it, which is usually a large chunk of the unresolved files. We narrow down our pool of unresolved files, marking everything found in the manifest data as resolved, and as Figure 4 shows, the process then continues, picking another unresolved package and recording its contents as shown in Figure 5. This continues until all packages have been resolved. Instead of always resolving the contents of the folders by opening each one individually, only a small handful of folders need to be opened to resolve everything.

We can improve on this further with the observation that we have a set amount of downloadable content released every year, and that the majority of the downloadable content for us is forward-compatible content from an older title, so it's already out in the wild with set folder names and files. As long as we can get far enough ahead on the production of downloadable content and by making sure we finish the last packages for the previous title that are to be forward-compatible with our new title, we can include all the information of the previous title's content in the new game's data. This means that for Guitar Hero 5, we were able to include all the manifest entries for Guitar Hero III: Legends of Rock and Guitar Hero World Tour content on the disc. By using those manifest entries, we can resolve all the downloadable content from previous titles without having to open a single folder with the Binary Chop method. Now instead of scanning 1,000 packages, we're scanning around 400—those released just for this title—with the Binary Chop method.

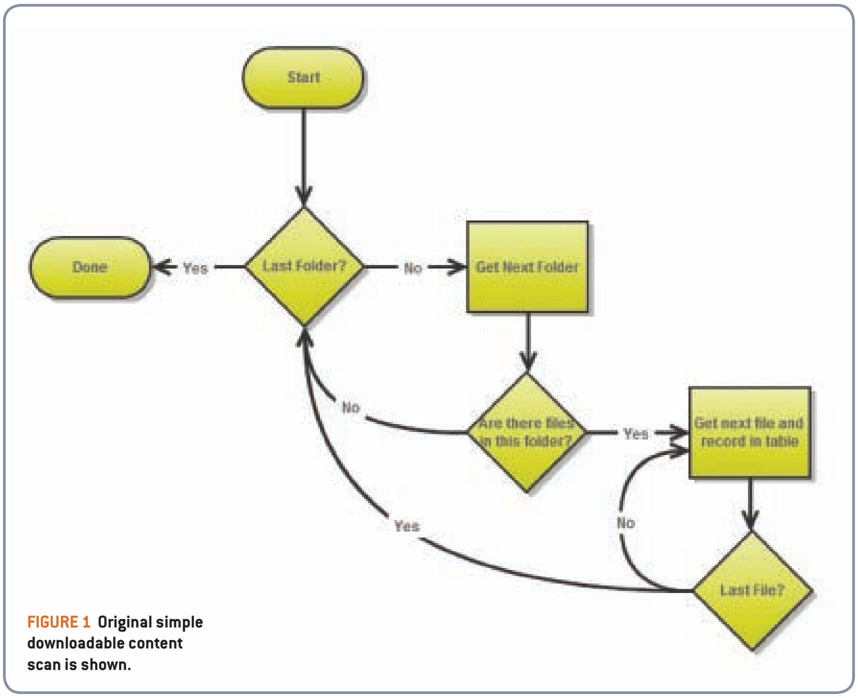

Start
Last Folder?
Yes
Done
No
Get Next Folder
Are there files in this folder?
Yes
Get next file and record in table
No
Last File?
No
Yes

FIGURE 1 **Original simple downloadable content scan is shown.**

PRODUCTION CONSIDERATIONS

» There are some production considerations that come with the improved performance. A number of changes had to be made to the way we produced content in order to avoid breaking assumptions made by the Binary Chop method. Many of these production considerations would eventually be solved through better production tools and a well-documented process.

First, the Binary Chop requires strict ordering of packages in production. The system relies on the fact that when you open a package, it has a manifest that lists everything that came before it. If entries are not correct or missing, packages will end up unresolved. Generally, this means when a change is made to the order, all packages after the one that was moved need to have their manifests re-generated so that they are all correct. This can sometimes cause issues with testing content. Our policy in general is to verify DLC-related bugs with a fresh install of the latest content packages.

Second, if a package name changes, all packages after the one that is changed need to be re-generated as well. This is because the manifest is using the names of the packages to identify them, as well as knowing what file to attempt to open as the manifest. If packages are not re-generated, the manifest will fail to load (because they don't exist with the file name expected), and packages won't be resolved correctly.

Similarly to the previous production rule, because the manifest uses package names to identify the package and then to specify which files are in that package, identical names lead to ambiguity as to which content folder is which in the manifest. For most titles, this is probably something that wouldn't even be considered, but we hit this really early on when we created a "Neversoft Track Pack" for both Guitar Hero World Tour and Guitar Hero 5. Imagine our confusion when the title would on occasion fail to open data

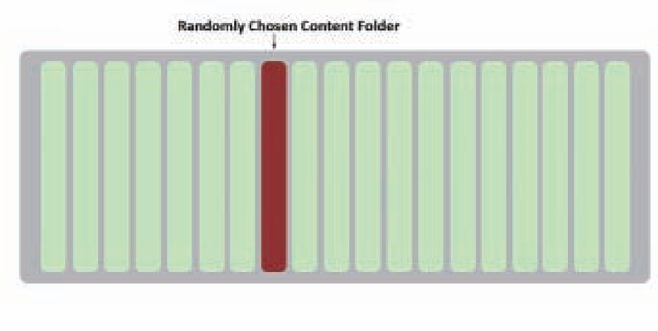


FIGURE 2 **A package is chosen at random from a list of unresolved packages.**

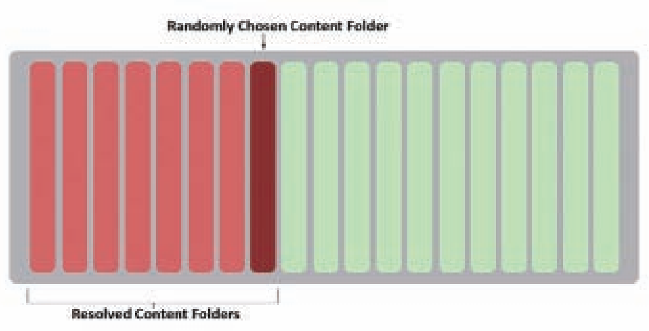


FIGURE 3 **It's important to note that for demonstration purposes the packages are shown in order of release. This shows how the chosen package resolves all previously released packages, but in practice the resolved packages will be distributed randomly. The manifest from the randomly chosen content package is used to resolve all previously released packages.**

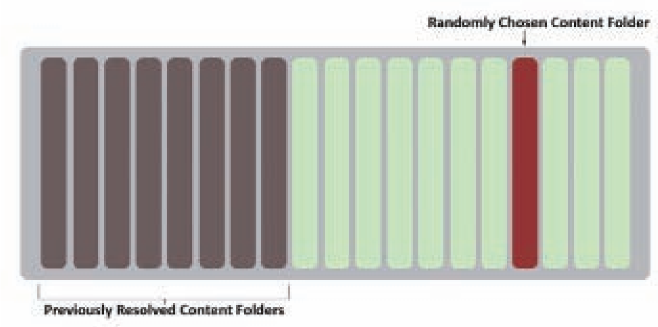

Randomly Chosen Content Folder
Previously Resolved Content Folders

FIGURE 4 **The process continues, and another content package is chosen at random from the remaining unresolved packages.**

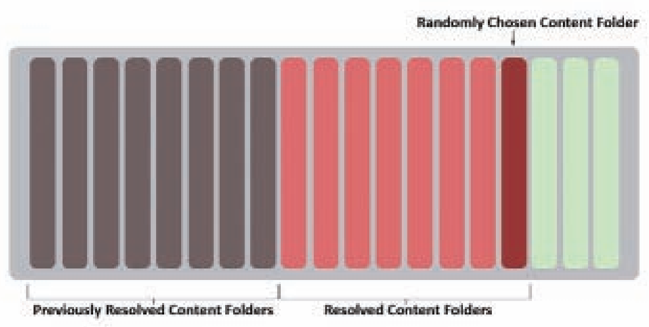

Randomly Chosen Content Folder
Previously Resolved Content Folders
Resolved Content Folders

FIGURE 5 **More packages are resolved.**

from the first package even though the files were clearly there! The manifest system would resolve both the old package and the new GUITAR HERO WORLD TOUR package. This was exacerbated by our DLC system opening the first folder in which the needed file was found—since our scan system builds the content table in a random sequence, this was non-deterministic. Sometimes, when playing a song from the GUITAR HERO WORLD TOUR version of the "Neversoft Track Pack," the correct folder was chosen. Sometimes the wrong one was, which seemed to imply a problem other than with the scan.

Finally, since the folder name on the Xbox 360 is based on the title parameter in the XLAST and can have multiple translations, and the folder name is used to know the name of the manifest file to open, you need multiple manifest files with different names to match these different translations. This is pretty trivial when writing a tool to create the manifests, but it is something to keep in mind.

Ultimately, our solution for the above issues was to improve our production tools, making it difficult if not impossible for packages to be created in a way that would break the scanning system. Our DLC production tool gives visual warnings when packages need to be re-generated, automatically creates all files needed from the data input into it, and locks packages when we release them so that edits cannot be made once packages are finalized. Another benefit of all this work on the tool chain, aside from the reduction in errors, is the decrease in time spent by production creating and managing packages in general.

METHOD COMPARISON

>> Since scanning for folders must happen no matter what, let's ignore it and focus on scanning for files within the folders. With our original system, our average case was the same as our worst case: completely linear. As we added more packages, we had to scan a proportionally greater number of files. With our Binary Chop method, our worst case is no worse; if by some stroke of horrible luck the system were to choose packages in the order in which they were released, we would be back to scanning linearly. However, the statistical chances of our system breaking down to a linear scan are very small. The chances of the randomly chosen package being the very first package released, and then the next package chosen randomly being the second package released and so on is 1/n!, or in our worst case of 1,000 packages that's about 1:4x102, 567. Our best case is to open a single package and resolve everything, but this is equally improbable as our worst case. In our average case, we do get something resembling a logarithmic behavior. There is a 50 percent chance that less than half of the data set will be eliminated in a step, and a 50 percent chance more than half of the data set will be eliminated. Since both of these cases are equally likely, over time, this normalizes to half of the data set being eliminated with each step. We then can show this average behavior taking log(N) steps to eliminate all N packages.

The speed up in practice is pretty remarkable. To scan 420 packages and 2,092 files took 73 seconds with our original system. With just the Binary Chop, without an embedded manifest file on disc, the scan is cut down to 4 seconds.

COLIN CRENSHAW *started at Neversoft Entertainment as a gameplay programmer in 2007 before becoming an engine programmer, which he's done for the last two years. He has felt privileged to work with some excellent people on the GUITAR HERO franchise.*

TRUST ME

OR, HOW I STOPPED WORRYING AND LEARNED TO LOVE THE TOOLS TEAM

EVERY CLOUD, SAYS THE OLD CLICHE, HAS A silver lining. Nobody likes the angst and insecurity of our dubious economy. Still, in these lean times, at least we're far less likely to be dragged away from useful work for chirpy HR seminars on workplace communications or, god help us, to don blindfolds and lean backwards into the clammy arms of that WoW fanatic from IT so we can "learn to trust our teammates." Say what you like about the dire business climate, at least it tamps down the fluff industry.

Honestly, though, the gurus of the workplace have it right about one thing. Blindfolds aside, trust is a key part of any functioning workplace and it's particularly important for us game developers. We make art in a collective medium where all of the disciplines are inextricably tied together. You can make a career as a rock singer with a mediocre backing band. You can prosper as the best actor on a lame sitcom. Cranking out great models for a game that crashes on load, however, isn't going to earn you fame or fortune. Our work can easily be dragged down by a lame graphics engine or a busted pipeline (though, to be fair, we should add that a fun design or a snappy engine can also be torpedoed by inadequate artwork).

Put bluntly, there are no lone geniuses in game art. We depend on other people—graphics engineers, tools coders, designers, and our fellow artists—to do what we do. These relationships are the foundation of our work lives. This doesn't mean that you have to start every whiteboard session with a group hug. In our world, trust is more than just a warm fuzzy feeling. A healthy respect for your teammates—or the suspicion that they don't know what they're doing—shapes how you work in very concrete, unemotional ways.

WHY TRUST MATTERS

>>Consider the morning ritual that greets most working artists at the beginning of the day. Theoretically we all saunter in, fire up the box, and pull down the latest tools and most recent build of the engine. We grab a cup of coffee while the system assimilates yesterday's changes and by the time we're finished with icanhascheezburger.com, we're all caught up and ready for the day's labors.

That's the theory, but it's rarely the practice. Far too many artists will do anything to avoid that daily get. "I don't like the risk," says one, "you never know if it's going to work from one day to the next." Another artist complains, "The tool I need broke last month so I rolled back to a working one and now I don't want to get again." There's the perennial favorite, "It takes too long, so I only do it once a month." And don't forget the oldest standby of them all: "I don't worry about that stuff, I just need my Max files."

Sound familiar? These all-too-common stories are tangible examples of a breakdown in trust, and you don't need to watch a special edition of Oprah to see how it undermines the studio. If the artists don't trust the tools team, they try to bypass the tools and find workarounds. But this goes both ways. Just imagine the dark mutterings coming from the engineering department every time they have to hunt down a "bug" that's really caused by out-of-date tools or hacky workarounds. The artists think the engineers are lost in the clouds, the tools team thinks the artists are big babies. Pretty soon, things bog down in recriminations and buck-passing.

The damage from this kind of breakdown goes far beyond eye-rolling and sarcastic IM messages. It's more than everyday inter-departmental wrangling, too. If you cope with tool and process problems by simply opting out, you aren't just irritating the tools coders or TAs, you're also leaving your fellow artists hanging.

It's perfectly understandable, of course. As an artist, you're stuck with the studio toolset—it's not like you can take your business somewhere else if you don't like your in-house material editor or custom map tool. With deadlines looming and creative problems to solve, it's hard to gin up much enthusiasm for bird-dogging an obscure problem with a wonky exporter plug-in or tracking

down the exact setting that adds 11,000 warning messages to your map imports. The temptation to find a quick workaround and move on is overwhelming. But—and here's the important part—ducking out on the problem doesn't just bypass it. It perpetuates it, for you and for everybody else.

When the culture in the art department deals with tools problems by simply ignoring them, it becomes impossible for the tools team or tech artists to build long-term solutions. Nobody's going to fix a bug that never gets reported, and nobody's going to improve on a bad workflow if the artists don't lobby for a real improvement. Grumbling to each other at lunch doesn't count—unless someone actually makes it clear that there's an issue, things won't get better. Never underestimate how little the other departments know about your problems. The political hassles of scheduling and finding resources for a real fix may be a problem that has to be settled among leads, but it's every line artist's responsibility to make sure both the art leadership and the tools team know when things are broken.

The old chestnut about trusting is that it leaves you vulnerable. Working artists can relate to that, because nothing makes you feel more vulnerable than having a critical part of your pipeline break down when you've a looming deadline and the producer is breathing down your neck. Trusting can be scary—hell, the entire output of the Lifetime Network is devoted to that theme—but learn-to-love-again melodrama teaches you that you need to take some risks if you want to grow ... or to ship good games.

Fortunately, there are some concrete things you can do to strengthen the bond of trust between the content team and the folks who are supposed to help them make the game—and none of them require a sit-down with Dr. Phil.

TAKE IT SLOW

»The most important way tools teams can build trust is to manage their releases better. Nobody would dare release a 99 cent iPhone game into the wild without testing and QA support. All too often, though, tools teams and tech artists roll out changes without doing the legwork needed to make sure the tools work as advertised. When teams were smaller and development was less ponderous, tool writers learned to love fixing bugs and adding features with a simple check-in. It feels great to be able to drop a one-line fix into a script and tell your disgruntled customer, "Just get the latest version and you'll be able to work again."

In a big modern studio, though, that immediacy is an expensive luxury. When you're supporting 50 to 100 artists, the costs of catching bugs late mount up quickly. This is particularly true for tools like Max and Maya plug-ins, which can leave bad data inside art files long after the original bug is fixed. No fix is fast enough to pay for the cost of dozens of angry, idled artists. Bugs will always happen, but an aggressive testing program (complete with real, live testers, whether they're QA folks or tech artists), will go a long way toward easing the pain. Testing will certainly slow the response time for feature requests and trivial fixes, but the benefits in terms of quality and reliability, and therefore trust between the tools team and the content team, are more than enough to pay for the hassle.

PLAN AHEAD

»Ship dates matter, as we know all too well. A planned release schedule for tools enhances trust because it helps hard-working artists brace for changes. Tools that just magically appear in the daily synch are often simply ignored; if a new skeleton editor comes out while you're polishing run cycles, there's a good chance you won't even know it's there. (Note to tools and tech art teams everywhere: *Nobody reads those emails!*) When tools change their behavior or appearance without warning, artists lose faith in the stability of their environment. And of course, if an innocent synch introduces bugs along with UI changes, the trust level will plummet and the vicious cycle of avoidance will begin.

If, on the other hand, tools go out the door at well-advertised times—hopefully, coordinated with production to avoid stepping on sensitive deadlines—everyone can budget the time and energy needed to make sure that the new functionality is supported, new workflows are well understood, and any bugs that slip through test are squashed before they do too much damage. Scheduled releases, accompanied by scheduled learning time, are a safer and less intimidating way to keep the artists and tools builders communicating.

BUY IN

»Trust, as the cliche goes, is a two-way street. If we want the tools to get better, we need to contribute something too. The effort that goes into doing tools with adequate testing and support is hard to sustain if the production people subvert the new tools and insist on their right to opt out of changes. Keeping an art team productive is a herculean labor. There are a million sources of potential problems: out-of-date video drivers, conflicting versions of software, different OS versions—you name it. When every artist has a completely private environment, with personally selected versions of tools and private workarounds, support is exponentially harder. Time that should go to fixing bugs in the current version of the tool will be wasted simply trying to figure out what's going on and why.

Surrendering control over your personal working environment is emotionally difficult for most artists. We're slavishly devoted to customizing our hotkeys, tweaking the layout of our custom toolbars, and enhancing our personal workflows with scripts and widgets we've downloaded from the net. We want the same level of personal control over our in-house tools. Unfortunately, the line between what's personal and what has to be shared with others can be pretty blurry. Out-of-date tools may be producing data that's subtly wrong without being obviously busted, dragging down game performance or leading to crashes. Tools that are made to work together may function poorly if they aren't updated in tandem. And working in a completely private environment means you can't give meaningful feedback to the tools folks to help them do their jobs better. The tools team can only support one good environment at a time—you want it to be the one you're working on.

Besides, when the whole art team is on the same toolset, it's really obvious to everyone what buggy tools cost the studio. Nothing encourages careful attention to tool quality like telling the producer, "None of the artists can work until this is fixed."

TALK SHOP

»As any *Oprah* devotee knows, communication is the key to building trust. Unfortunately, tools providers and line artists don't always communicate as well as they ought to. Technical types tend to be tempted to build tools that make things easier for computers, rather than helping artists. Artists, on the other hand, are frequently too shy to bring their concerns to the attention of the tools team.

If you don't tell people clearly what is broken in your workflow, what's preventing you from iterating, or what kind of changes would make things better, you have no right to bitch about the tools you've got. Both sides need to explain their needs clearly and both sides need to listen attentively. Tech artists, who by nature have a foot in both worlds, are invaluable for helping these discussions along, as are producers who can provide high-level guidance on the distinction between must-have and wouldn't-it-be-nice features.

The bottom line is really very simple: artists, tech artists, and tools engineers all need to commit to making things better. Good tools don't make for good games on their own, but lousy tools certainly make for lousy games. So, if you skip the meetings, don't report bugs, and try to roll your own toolset, you're making your own life and your teammates' lives harder in the long run. Trust me on that.

STEVE THEODORE *has been pushing pixels for 15 years as a modeler, animator, and technical artist. His credits include* HALF-LIFE*,* TEAM FORTRESS CLASSIC*,* COUNTER-STRIKE*,* SOCOM 3 *and* HALO 3*. He's currently the technical art director for Undead Labs in Seattle.*

THE CASUAL/HARDCORE CONTINUUM

THERE HAS BEEN CONSTANT TALK ABOUT CAPTURING CASUAL GAMERS IN THE GAME INDUSTRY EVER since the days that MYST and TETRIS showed eye-popping sales to customers who, by appearances, seemed to be totally normal non-basement-dwellers. When the industry started out, all the advertisements showed a family playing together, and the intention was for all games to be casual (or universal). But as time wore on, the industry kept making more hardcore games for hardcore gamer audiences, until the mainstream opinion of games became "murder simulators." In recent years though, there has been a significant amount of suggestion that games have gone casual again.

And why not? THE SIMS, most assuredly dubbed a game by non-gamer significant others, dominates monthly PC top-ten lists and is one of the best-selling franchises of all time. Casual games like Zynga's MAFIA WARS and FARMVILLE have tens of millions of active players. And while the hardcore players were taking sides in the great 360/PS3 flame wars on the web, no one predicted that the Wii would end up very nearly selling more consoles than both of them combined. Indeed, one could argue that we're now firmly entrenched in the days of the casual gamer.

Presented with all these facts, it's not terribly surprising to see hardcore gamers (and indeed, devoted game developers) trending toward despair. There seems to be a very real fear that games for gamers are dinosaurs, hurtling toward extinction in a world where all games are Nancy Drew, hidden objects, and web apps.

I don't see it that way. I see casual games as creating the next generation of hardcore gamers.

DEFINITIONS

» It has become common practice to conflate casual gamers with women, and casual games as the kind of games that women play—often derisively. The truth is much more complex than that. Casual is not a market, nor is it a genre, as much as it is a *level of investment*. A casual hobbyist does not commit hardcore to her hobby, but a hardcore one does.

Consider knitting. To many, knitting is probably thought of as one of those things that grandma does. In game terminology, any such activity is surely a casual hobby. Walk down the yarn aisle of Hobby Lobby or peruse a knitting website like Ravelry.com and you'll be quickly disabused of this notion. Knitting is a skill that is easy to learn but hard to master. Knitters swap rare patterns, go to conferences and crafting festivals, and spend vast sums of cash on hoards of yarn they can never possibly use in one lifetime, and form communities of like-minded hardcore knitters that talk in indecipherable slang and occasionally slag on the newbies trying to put together their first pair of socks.

Clearly, one can see certain parallels.

The point is that knitting is not a casual activity—but neither is it a hardcore one. Instead, the hobby easily scales itself to the investment level of the person who is interested in it, from the knitter who may pick up the needles once a year to the hardcore yarnslinger obsessed with knitting her own burial shroud. Knitting is a popular hobby because of that scalability. This scalability of experience is mirrored in the most popular games in our industry.

THE SIMS

» I've heard the same story about THE SIMS countless times: Someone I knew who considered themselves a "hardcore gamer" bought it because

WORLD OF WARCRAFT captivates hardcore and casual gamers alike.

The Sims (2000)

it was Will Wright's masterpiece, and played it a couple of days before they realized they were spending their time cleaning virtual toilets. However, their wife, daughter, or significant other had been watching over their shoulder from time to time, and before the friend could turn off the computer in disgust, this person, who normally treated video games with a mixture of disdain and disgust, would ask—"Can I play?"

These stories were told often with a level of sadness and betrayal: the great Will Wright had sold them a *casual game.* But the truth of the matter is not so simple. Many of these "non-gamers" went on to play the game obsessively, spending endless hours in the game nurturing or torturing their Sims, shelling out hard-earned cash for the two sequels and dozens of expansion packs released, publishing their stories on the web, and creating their own (often spectacular) art to import into the game. It turns out that THE SIMS has enough intricacy and depth to turn some of these people into seriously hardcore SIMS players.

And my friends, the husbands and boyfriends? It turns out that for this game, *they* were the casual gamers.

The funny thing, of course, is that the game industry is driven by "classically hardcore" gamers, and as a rule, we tend to have a problem making games that we can't get passionate about ourselves. This can lead to myopia that's kind of staggering, and challenges the notion that the industry is too driven by casual-oriented business-think: 10 years later, THE SIMS has sold 125 million boxes or expansions for more than $2.5 billion in revenue, and yet no one has shipped a serious, well-funded competitor. Meanwhile, a quick tour of the E3 floor shows that all of the triple-A companies are fighting for ever smaller slices of the space marine market, the racing game market, and the World War II shooter market.

Thanks to THE SIMS, millions of new customers are walking down the PC games aisle at Best Buy. Now then, what are you going to do about it?

THE SPECTRUM

» Compared to the MMOs that came before it, WORLD OF WARCRAFT is a pretty player-friendly MMO. Death penalties are light, travel times are quick, players can solo for a large part of their experience, and the degree to which a 14-year-old mouthbreather with too much time on his hands could ruin your whole day were relatively curtailed. Some players who looked back fondly at the wild west-like brutal frontiers of EVERQUEST and ULTIMA ONLINE were quick to deride WORLD OF WARCRAFT as a casual experience for "care bear" players.

It's not, though. Being a top-end raider in WoW takes an enormous amount of skill, teamwork, and devotion of time and energy. Being a PvPer at the top of the Arena rankings is even more brutal and competitive. The difference between WoW and most of its predecessors is not that it's easier, but rather that it provides content and challenges no matter the player's investment level. Casual players can solo, while slightly more invested players can run dungeons and other light group content. Hardcore players run raids and PvP arenas, and the top-end players chase ludicrously difficult achievements in those environments. Furthermore, Blizzard continually works to refine this escalation and coax people up the ladder.

Back in 1971, Nolan Bushnell of Atari said, "All the best games are easy to learn, and difficult to master," a design philosophy now treated as instinctual by nearly every designer in the industry. This describes a philosophy of smoothly escalating levels of investment. Casual to hardcore is not a binary option—it's a sliding scale. We, as designers, want to find ways to smooth out that climb so that players increase their commitment level to our game. Good games convert casual players into hardcore ones, because these games are worth the investment.

REEXAMINING CASUAL VS. HARDCORE

» This idea that casual and hardcore are levels of investment leads to some interesting design thoughts. For example, a game being too casual and being too hardcore could both be considered broken game designs. A casual game is one that lacks the depth for players to get truly invested into it, which means that people quickly wash out of it (something that is definitely not true for THE SIMS or WORLD OF WARCRAFT). On the flip side, a hardcore game is one where the price of investment is more than the player is willing to pay. Typically, the problem here is a poorly managed investment curve—when a player throws down his controller in disgust and says, "That's impossible," it typically means that the difficulty curve has risen too high before the player can bother to care about it.

The truth of the matter is, while players tend to think of themselves as all-around hardcore gamers, in actuality, they tend to be invested in only a few genres or titles. A hardcore racing gamer may only play shooters casually, and be moderately into RTSes. Furthermore, even inside the genre, a game must prove itself to earn a player's devotion. Put another way, whether you're making THE SIMS or DEAD SPACE, FARMVILLE or DEMON'S SOULS, your gamers are starting as relatively uninvested casual gamers. Converting those gamers into devotees means coaxing them inward until they can find the fiendish strategic depths of your games.

This isn't just about making things easier. A game that is too easy, in fact, risks boring people before they can find your secret sauce. The secret really seems to be allowing players to quickly find a place in your game where they can learn, where they can demonstrate mastery, and where their level of skill and investment are rewarded. Can your players find the spot for them in your game's casual-hardcore equilibrium? And can you make them want to go to the next rung up?

DAMION SCHUBERT *is the lead combat 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.*

GDC ONLINE REVEALS NEW BOARD MEMBERS, DEVELOPER'S CHOICE AWARDS

GDC ONLINE ADVISORY BOARD ADDS NEW MEMBERS

» As this October's GDC Online prepares for its first content announcements, the Texas-based show added Playfish and Electric Bat execs to the BioWare, Nexon, SOE, and Zenimax notables already on its Advisory Board.

The GDC Online Advisory Board, which is also helping oversee the newly revealed Game Developers Choice Online Awards, is intimately involved in selecting content for the October 5–8 show at the Austin Convention Center.

Board members individually rate each GDC Online lecture submission. They also meet in person multiple times per year to decide final lectures and keynotes from submissions and invitations ensuring that GDC Online remains the leading worldwide game industry event to concentrate specifically on social games, free-to-play titles, MMOs, and more.

At last week's GDC Online board meeting in Austin, existing board members were joined by new recruits Sebastien de Halleux, VP of EA Interactive and Playfish co-founder, and Cindy Armstrong, Webzen and SOE veteran and Electric Bat Interactive CEO.

The duo joins leading figures from the online game industry on the active Advisory Board, which includes Gordon Walton and Rich Vogel of BioWare Austin (Star Wars: The Old Republic), online design notable Raph Koster (Metaplace), Nexon America VP Min Kim (MapleStory, Combat Arms), and Schell Games' Sheri Graner Ray (The Mummy Online).

Following the recent Advisory Board meeting, first lectures for GDC Online should be announced in the near future, with invitations being extended to many more of the online game industry's leading players.

More information on the event, which will include an Expo Floor, the GDC Online Awards, and co-located Summits on notable topics like iPad/iPhone games and 3D stereoscopic games, is available at the official GDC Online website.

GAME DEVELOPERS CHOICE ONLINE AWARDS

» Organizers of this October's GDC Online conference—formerly GDC Austin—have announced that they will host the first annual Game Developers Choice Online Awards to recognize the rich history, technical excellence, and continued innovation in the arena of online games.

Winners for the Game Developers Choice Online Awards will be selected by a subset of the International Choice Awards Network (ICAN), the same group of over 500 handpicked leading industry creators that pick the Game Developers Choice Awards winners at GDC in San Francisco every year.

The full list of categories for the first annual Game Developers Choice Online Awards include:

[REGULAR AWARDS]

∞ **The Best Online Game Design Award** honors the overall excellence of design in an online game that launched in public beta or full version in the 12 months prior to May 2010. This category recognizes the best social-specific gameplay mechanics, quest design, and other major game design elements.

∞ **The Audience Award** honors the favorite game of the worldwide online game community. After initial developer-specific nominations, members of the public will vote on their favorite currently operating online game, with options from the Game Developers Choice Online finalists and beyond.

Austin, Texas

∞ **The Best Online Visual Arts Award** recognizes the overall excellence of visual art in online games, including, but not limited to, character and animation design, interface design, 2D or 3D art creation, and art direction of all kinds.

∞ **The Best Community Relations Award** honors the currently operating online game that provides the highest quality community feedback and experience, including customer support, forum moderation and leadership, weblog and information updates, real-life events, and other forms of community outreach.

∞ **The Best Online Technical Award** recognizes the overall excellence of technology in an online game. This includes excellence in elements such as complexity of network infrastructure, persistent world coding, graphics technology, or artificial intelligence.

∞ **The Best Social Network Game Award** recognizes the best game that launched on a social network such as Facebook or MySpace in open beta or full version, honoring the most outstanding title in terms of overall depth, execution, and quality in the space.

∞ **The Best Audio for an Online Game Award** recognizes the overall excellence of audio in an online game, including sound effects, musical composition, and sound design for an online title.

∞ **The Best New Online Game Award** recognizes excellence in any online-specific game, including MMOs, free-to-play titles, and social network games that launched to the public in open beta or full versions in the last 12 months.

∞ **The Best Live Game Award** recognizes the best currently operating online game that distinguishes itself with exceptional new content through expansion packs, patches, or other updates, as well as a vibrant player community, high-quality community management, and network operation during that period.

[SPECIAL AWARDS]

∞ **The Online Game Legend Award** recognizes the career and achievements of one particular creator who has made an indelible impact on the craft of online game development.

∞ **The GDC Online Awards Hall of Fame** recognizes a specific online game that has resulted in the long-term advancement of the medium, pioneering major shifts in online game development and games as a whole.

More information about the Game Developers Choice Online Awards, which will take place on October 7 in Austin, is available at the official website for the event. The awards are co-located with the market-leading GDC Online event, which takes place from October 5–8 at the Austin Convention Center, and for which registration is now open.

TOOLS OF THE TRADE

APPS TO MAKE APPS

APPLE'S IPHONE HAS SPARKED a revolution in cellular phone technology. Touch screen smartphones combined with an open and active development community have become the new software gold rush. For game developers, the iPhone rapidly showed itself to be a viable gaming platform.

Like PCs, the iPhone and iPod touch differ greatly from traditional game consoles in one major way: instead of simply being an end-point game platform, the iPhone is also a development platform. While game developers have been shrinking their console titles to the iPhone's touchscreen, app developers have been busy creating a vast number of tools that audio professionals can use to create content and assist production—and not just for Apple platforms either. With the launch of the iPad, the same pool of apps is now available for tablet computing.

THE HARD STUFF

» One of the most interesting shifts the app revolution has brought is the move to create software replacements for traditionally hardware-based external tools such as guitar tuners and metronomes. When I asked audio professionals which apps they were using, the most common reply was Voice Memos, a Dictaphone replacement that comes standard on the iPhone. Across the industry, audio professionals are using Voice Memos to capture melodies, thoughts, and sketches. With the ability to edit and email recordings, some are even using Voice Memos as a basic field recorder. For those looking for a more substantial field recorder, a number of full-featured recording apps are available such as Audiofile Engineering's FiRe and iProRecorder from BIAS.

The combination of touchscreen and microphone has led to the convergence of what used to be many separate tools into collected app suites. Agile Partners' Guitar Toolkit is a great example. Guitar Toolkit offers users a guitar tuner, a metronome, and a smart library of guitar chords and scales plus the ability to add custom tunings. IK Multimedia's Amplitube, as another example, is an all-in-one collection of 11 guitar stomp boxes, 5 amps, 5 cabinets, and 2 microphones.

But guitars aren't the limit of audio apps. From SPL meters like Faber Acoustical's SoundMeter to SEB's RT reverberation timer, the diversity of the once hardware-only tools now represented by software apps continues to grow by the month.

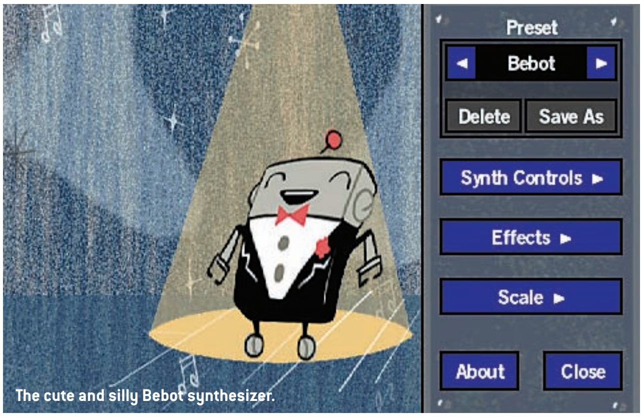
THE SOFT SELL

» By far, the majority of audio apps available are software utilities specifically designed to assist in production and content creation. There are hundreds of these apps that make sounds, collect and arrange sounds, or simply help make your audio job easier.

When it comes to making sounds, the iPhone and iPad have a rich palette of soft synths, emulators, and noise generators. There are acoustic instrument emulators like Peterb's Virtuoso Piano. There are vintage synth emulators like Yonac's megaSynth which allow you to sculpt synth textures and extract the sounds as WAV files. There are even oddball apps like Normalware's cartoon-driven Bebot which is part tone generator, part ribbon controller, and adds an unpredictable—yet undeniably fun—approach to sound design.

While the app store is dominated by smaller developers and hobbyist programmers, even the big boys of software emulation are diving in. In June, Propellerhead Software rereleased its seminal classic Rebirth synth application for the iPhone and iPad. Rebirth returns with its full functionality from the original application, including access to all the user-generated mods which were such a vibrant part of the original Propellerhead community.

Beyond simply being a sound emulator, apps can also transform the iPhone and iPad into audio workstations. For a paltry $10, a number of digital audio workstations (DAWs) are offered. Harmonicdog's MultiTrack DAW presents users with a suite of multitrack mixing and editing tools, though it lacks the MIDI functionality that one would expect. Additionally, INTUA offers BeatMaker, which has a suite of sampling and sequencing tools, though INTUA describes it as "ideally suited for live performances." While neither of these options will replace your Pro Tools HD rig, they may offer DAW options for audio pros on the go.



The cute and silly Bebot synthesizer.

Speaking of Pro Tools, the iPad and iPhone can also function as remote controls for professional DAWs. Far Out Labs' ProRemote turns the iPhone and iPad into a control surface that links directly to Pro Tools, Logic, and Ableton Live. As the name implies, ProRemote is a Mac-oriented remote control with fader interactivity, pan control, solo, and mute buttons, plus transport controls and time display. Far Out Labs claims to "unofficially" also support Digital Performer and Cubase. Cubase users already have access to Steinberg's own Cubase iC, a free control surface solely for use with Cubase.

Lastly, there are the myriad small apps that simply help audio professionals to do their jobs whenever and wherever they want. There are dozens of PDF readers like LLC's Aji Annotate PDF that can be used to replace printed scripts with tablet displays. WattenEarth's time:calc is a calculator specifically formatted for math functions dealing with hours, minutes, and seconds. As a music supervisor, I've even found a mainstream app like Pandora Media's Pandora Radio can be essential when searching for inspiration or music to license.

In a world of software and media convergence, the App Store is at the forefront of changing the way we define our audio tools. With the rising influence of additional platforms like Google's Android, the next few years promise to be a time of ever-increasing invention and innovation.

JESSE HARLIN *has been composing music for games since 1999. He is currently the staff composer for LucasArts.*

TWISTED TEASDALE

HARMONIX SENIOR DESIGNER MAKES INDIE MOVES

Dan Teasdale is perhaps best known for his work at Harmonix, where he led design on the ROCK BAND series, though his roots go back to Pandemic Australia's early days, and further. Teasdale recently left Harmonix to work on smaller games with Twisted Pixel, maker of 'SPLOSION MAN and the upcoming COMIC JUMPER.

BRANDON SHEFFIELD: What prompted you to leave Harmonix?

DAN TEASDALE: It was a tough decision, but the main thing driving me to move on was a nagging desire to be more hands-on while making games.

While I'd been able to keep some hands-on time on ROCK BAND 3 by building tools and designing and implementing some of the features that spanned multiple systems, being a lead on a 150+ person team pretty much saps most of your time to actually make and design things. It wouldn't be uncommon for me to finish up all of my lead work, meetings, system reviews and such, then realize that it was 6pm before I had a chance to start on the "fun" parts of making games.

So, with ROCK BAND 3 wrapping up, I decided to try and find a role that would let me be more hands-on with feature design, implementation, and iteration, while at the same time being able to still help drive the vision and direction of a title. It also had to be an amazing studio with great culture, [and which] made games I played and loved.

BS: Have you ever worked on a team of small size like this? Any concerns about it? Do you think there's more or less stability in a small company?

DT: The first few years of Pandemic Australia were a similar size, actually. We started as a nine-person team of friends and co-workers who'd left Auran Games, and grew to about 18 people over the next project or two before things scaled up like crazy.

There are obviously pros and cons with a small team size, but I think the pros of being in control of your own destiny with other people who you know are awesome dramatically outweigh anything else. If things become unstable, then it'll be because everybody on the team was on the wrong track rather than a random "focus realignment" or "staff reduction."

BS: The character action genre has been flagging for years, but Twisted Pixel is going for it full-force—what's your take on the genre and its future?

DT: I think it's tough to even define "character action" as a genre, let alone say it's flagging. I mean, where do you draw the line? Should UNCHARTED 2 be put in the same genre bucket as NO MORE HEROES, let alone titles like 'SPLOSION MAN?

The key is really focusing on the gameplay experiences people are having. If you can make an amazingly fun game with just grey boxes in your prototype phase, then any kind of well-formed narrative or characters you add to that mix are going to improve the experience dramatically—action, platform, strategy, or otherwise.

I think that's one of the reasons why I'm really excited about working at Twisted Pixel. They're a studio that's proven they know how to make interesting and memorable characters that don't just rest on top of gameplay as an afterthought, but seamlessly integrate right into it.

BS: What would you say to anyone at a larger company looking to join or form an indie team?

DT: I think both AAA and independent development are totally different paths that take a different mindset to enjoy. If you're the kind of person who likes living on the bleeding edge of their craft while still being in control of their destiny, then you should 110% absolutely join or form an independent team to make a game!

new studios

Four veterans from Electronic Arts Los Angeles, who worked on titles such as COMMAND & CONQUER 4, have formed Haunted Temple, where they will develop turn-based strategy game SKULLS OF THE SHOGUN for XBLA and PC. The team is led by Jake Kazdal, known for his early work on games like REZ and SPACE CHANNEL 5.

EVE ONLINE developer and publisher CCP announced its fourth studio, CCP Newcastle, which is working on DUST 514, a console MMO first-person shooter.

Members of shuttered developer GRIN's quality assurance team have formed Trinity QA Studio, a Jakarta, Indonesia-based quality control firm that aims to support a wide variety of platforms.

Media conglomerate News Corporation has created a new game division targeting social games. Sean Ryan, former CEO of the avatar platform MEEZ and the LiveJournal blogging service, has joined on as general manager.

who went where

LucasArts has announced former Epic Games China CEO Paul Meegan will take the reigns as the company's new president, following the departure of Darrell Rodriguez.

Monumental Games, the developer of the FOOTBALL SUPERSTARS MMO, has added former Codemasters CEO Nick Wheelwright to its board in hopes of facilitating studio growth.

Ubisoft announced the departure of creative director Patrice Désilets, who played a key role in the development of the ASSASSIN'S CREED and PRINCE OF PERSIA franchises.

Casual game developer PopCap has added its second outside member to its board, Tech Data CEO Steve Raymund; PopCap hopes his experience as longtime CEO of a public company will help the studio grow.

Microsoft Entertainment & Devices Division president Robbie Bach and former Xbox figurehead J Allard recently announced their departure from Microsoft; the company says Allard plans to take an official advisory role to CEO Steve Balmer following his formal departure.

BECOME
PART
OF
THE
ELITE.

gear

TURNING THE WHEELS OF PRODUCTION

CREATED BY STUDENTS FROM THE DIGIPEN INSTITUTE OF TECHNOLOGY, GEAR WAS A $100,000 WINNER IN THE 2010 INDIE GAME CHALLENGE. WORKING IN AND OUTSIDE OF CLASS, THE GEAR TEAM BUILT THE GAME FROM SCRATCH, ALL WHILE KEEPING UP WITH THEIR REGULAR COURSEWORK. WE TALKED WITH GEAR'S JOSHUA MAICHE AND BRIAN LEE TO FIND OUT MORE ABOUT GAME CREATION IN THIS RESULTS-ORIENTED ENVIRONMENT.

***Jeffrey Fleming:* How did DigiPen prepare you to create GEAR?**

Joshua Maiche: The most instrumental course in helping us make GEAR was our game project class. Every year, we have this project class, so we have to make at least four games before graduating from DigiPen. This meant that all the programmers on our team already had a year's worth of experience working on a game with a team. Furthermore, the project class we were taking at the time was also very useful, as we were submerged in an environment where everyone in our year was trying to make an awesome game, and we had teachers that could give team, tech, and design feedback whenever we asked for their advice.

***JF:* How big was the GEAR team? How did you divide up responsibilities?**

JM: For the first two semesters, September 2008 to April 2009, we had three members: Brian Lee, Andrew Hall, and me. The roles just fell into place, because even though we were programmers, we all had different interests.

Brian Lee worked on general design, art, and music, and graphics and sound programming. Meanwhile, Andrew focused on level editor interface and general programming; he created our original levels. I acted as the producer and the level editor framework, gameplay, and physics programmer.

After those two semesters, we decided to refine GEAR for competitions. We signed on two level designers, Ben Frazier and Mike Halbrook, to remake our levels. Meanwhile, Brian and I kept our old tasks while adding one or two new ones. Brian worked on menus, and I worked on the level editor interface.

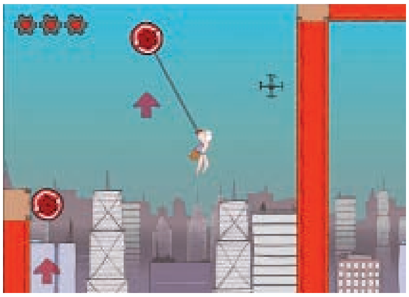
***JF:* What software tools did you use to build GEAR?**

JM: We used Visual Studio 2005 to program, Tortoise SVN to handle version control, and our own in-game level editor to develop levels.

***JF:* GEAR has an interesting look that combines sketchpad art with some pretty sophisticated graphic effects. What were the challenges in achieving the game's visuals?**

Brian Lee: Keeping the style consistent was definitely a challenge, especially with the tight time constraints. Being the graphics programmer, artist, and musician for GEAR while taking a full load of classes forced me to have a very iterative approach to the game's graphical production. Every few days, I would make a new sprite, tileset, or effect and work to ensure it was functional and fit in with the game's aesthetic.

Keeping the more advanced shaders from feeling out of place was somewhat challenging. The water shaders and normal mapped background shader underwent many tweaks during the creation of the game. The background shaders for the factory and ice levels originally looked more realistic, so I had to tone down the lighting a bit and add cel shading to keep those levels more consistent with the game's look. Making sprites and tilesets was somewhat tedious at first, but I eventually set up a nice art pipeline using Flash that allowed me to create, test, and edit tilesets and animations quickly and easily.



***JF:* What are your thoughts on difficulty? Were you able to bring in testers to give you feedback?**

JM: I feel that the appropriate difficulty for a game is affected by its demographic. For GEAR, I wanted everyone to be able to play it. Because of this, we tried to keep the difficulty high enough to keep the game interesting, but low enough that it didn't scare away players. The game on the "Spiky" difficulty was the result of this mentality. Unfortunately, it wasn't easy enough for weaker players. One of our professors, Ben Ellinger, suggested we add an easier mode without spikes, so we created a "Mild" difficulty, and that definitely made the game easier for newer players. We wanted to add a "Lava-Hot" difficulty for people that like being tormented by the difficulty of their game, but we didn't have time, so it had to be left out.

One of the benefits of attending a school centered on video game development is the abundance of gamers. As soon as we had something that could be played, we tested our game relentlessly. Playtesters were always eager to play our game. One of my favorite aspects of DigiPen is its student community. Everyone wants to help everyone else make the best game possible, so most students would gladly play our game and give us feedback, even if they only had five minutes.

***JF:* What was it like entering GEAR into competitions?**

JM: Working to get into the competitions was very challenging. Had we just decided to stop working after the last semester for that project was over, we could have stopped after eight months on the game. Instead, we kept going for another six months, three of which were during the school year, meaning we had classes and another game project to also keep up with. The actual ship date was a mad rush; a couple of us ended up staying up until six that morning trying to get everything we could in the game.

The actual results were a huge rush, too. I remember checking the IGF site the day the finalists were announced; I was a bit disappointed we hadn't made it. It wasn't until the next day that I found out we had actually received an honorable mention. That was a bit of a bittersweet moment: on the one hand, GEAR actually won something! On the other hand, we were so painfully close to making it as a finalist!

I also remember finding out about the results for Indie Game Challenge. I didn't even get to check the website for this one. Brian called me at 6:58 a.m. (the moment he found out we won). It took me a while to actually process what had happened, but once I actually understood what was going on, I was ecstatic to find out we had placed in IGC.

Overall, entering in the competitions was a blast, and it was definitely worth the stress and exhaustion we suffered to get to that point. I would highly recommend it for anyone who's considering it; polishing up a game for competitions is a very different experience compared to actually developing the original game, and it can be extremely rewarding.

—Jeffrey Fleming

GEAR WWW.B-LEE.NET/GEAR

GAMER

ADVERTISER INDEX

COMPANY NAME	PAGE	COMPANY NAME	PAGE
Activision	49	Northeastern University	52
Crytek	12	Perforce Software	6
Epic Games	19	Rad Game Tools	C4
Full Sail Real World Education	54	Scaleform Corporation	C2
Hansoft	43	Sledgehammer Games	33
Havok	50, C3	University of Advancing Technology	53
Madison Media Institute	52	Vancouver Film School	54
Nikkei Business Publications	27	Xaitment	3

Game Developer (ISSN 1073-922X) is published monthly by United Business Media LLC, 600 Harrison St., 6th Fl., San Francisco, CA 94107, (415) 947-6000. Please direct advertising and editorial inquiries to this address. Canadian Registered for GST as United Business Media LLC, GST No. R13288078, Customer No. 2116057, Agreement No. 40011901. **SUBSCRIPTION RATES:** Subscription rate for the U.S. is $49.95 for twelve issues. Countries outside the U.S. must be prepaid in U.S. funds drawn on a U.S. bank or via credit card. Canada/Mexico: $69.95; all other countries: $99.95 (issues shipped via air delivery). Periodical postage paid at San Francisco, CA and additional mailing offices. **POSTMASTER:** Send address changes to Game Developer, P.O. Box 1274, Skokie, IL 60076-8274. **CUSTOMER SERVICE:** For subscription orders and changes of address, call toll-free in the U.S. (800) 250-2429 or fax (847) 647-5972. All other countries call (1) (847) 647-5928 or fax (1) (847) 647-5972. Send payments to *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 *Game Developer* on any correspondence. All content, copyright *Game Developer* magazine/United Business Media LLC, unless otherwise indicated. Don't steal any of it.

TO: ALL_EMPLOYEES@GAMESCOMPANY.COM

A DAY IN THE LIFE OF AN EMAIL ALIAS

Air conditioning update

Team,
Just an update for you all on the broken air conditioner situation. A second service call was placed with the building manager a few days ago. Apparently he has a different phone number now. We are waiting on a response.

In the meantime, to help ease your discomfort, we have purchased a seedless watermelon and placed it in the kitchenette. Please divide it fairly amongst yourselves.

Thanks,
The Management

Taco truck

Team,
For those of you asking where the taco truck was last Friday, good news—we've found out they'll be back just as soon as they pass their Health Department re-inspection and are allowed to serve food in this county again.

Thanks,
The Management

Re: Updating the HR Database

Hey there Jenni-kins,
I think you have it already, but I'm still at the 334 Meadows Apt. 202 address. You know the place! My social security number is 350-17-0405.
P.S. You look great today.

Re: Re: Updating the HR Database

please disregard that last message

Previews tomorrow!

Good morning everyone,

There's going to be a bunch of game journalists walking around the front part of the office tomorrow for our demo event. PLEASE don't touch the decorations, the courier bags, or the buffet tables we're setting up in the foyer. We ran out of food last time and lost a lot of goodwill with those guys. We can NOT afford to let that happen again.

Also, I ask that you don't talk out loud about anything we haven't revealed yet, such as the jetpack, any of the "Gatling Tornado" attacks, or the motion control support—we plan to announce each of those as surprises in the months to come during our run-up to launch.

Finally, NO talking about how dumb game journalists are within earshot of the journalists. Please.

Thanks.

HEAT DISTORTION!!

Hey y'all, sorry for the spam but I have 50 extra tickets to the HEAT DISTORTION show at Shelby's this Saturday. Not to toot my own horn too much, but we are a pretty rocking band and always a good show! Come check us out!!

Heat Distortion.

PLEASE READ

Could whoever who has been microwaving fish curry in the mornings come see me at your earliest convenience? Thanks.

For sale—make me an offer.

Greetings folks,
I'm moving in with my girlfriend this weekend, but there's some stuff I need to get rid of first ... you know how it is. Everything is in great condition, and all reasonable offers will be entertained:

- *Man Up, Bro! How to Stop Being the "Nice Guy" and Score Hot Dates Every Night* (2007 edition)
- Lara Croft cold-cast statue—limited "ripped clothing" version (orig. value $200!)
- *Bikini Tentacle Fighters*, discs 1–13 (includes special bonus disc)

I've got it all sitting on my desk today if you want to take a look.

don't get latest.

i don't know what's going on guys but if u get latest u cannot run anything
get nothing on my dev kit and just a blank screen in the tool
"error code 00E4: target not found"
that sux, i had a lot of work 2 do today...

Re: don't get latest.

nvm guys, my dev kit was not on, haha

Re: Air conditioning update

Team,
We understand your discomfort especially since it is the peak of summer now. However, the more you talk about it, the more warm air escapes your body through your mouth. Therefore, by complaining, you are actually only contributing to the problem.

Thanks,
The Management

Re: HEAT DISTORTION!!

Anyone??? Still got those 50 tix...

Re: Re: HEAT DISTORTION!!

dude, nobody cares about your stupid band

Re: Re: Re: HEAT DISTORTION!!

Look, man, I don't get all up in your grill about your model trains or whatever else it is you do at home by yourself. Unlike you, some of us enjoy actually going out and doing things. Sorry I messed with the tiny little bubble you live inside.

Re: Re: Re: Re: HEAT DISTORTION!!

Hey, could you guys take this offline maybe?

Re: Re: Re: Re: Re: HEAT DISTORTION!!

The all_employees alias is for important studio-wide messages only. Abuse of this alias will result in disciplinary action. In other words, please shut up.

Thanks,
The Management

Re: Re: Re: Re: Re: Re: HEAT DISTORTION!!

/unsubscribe

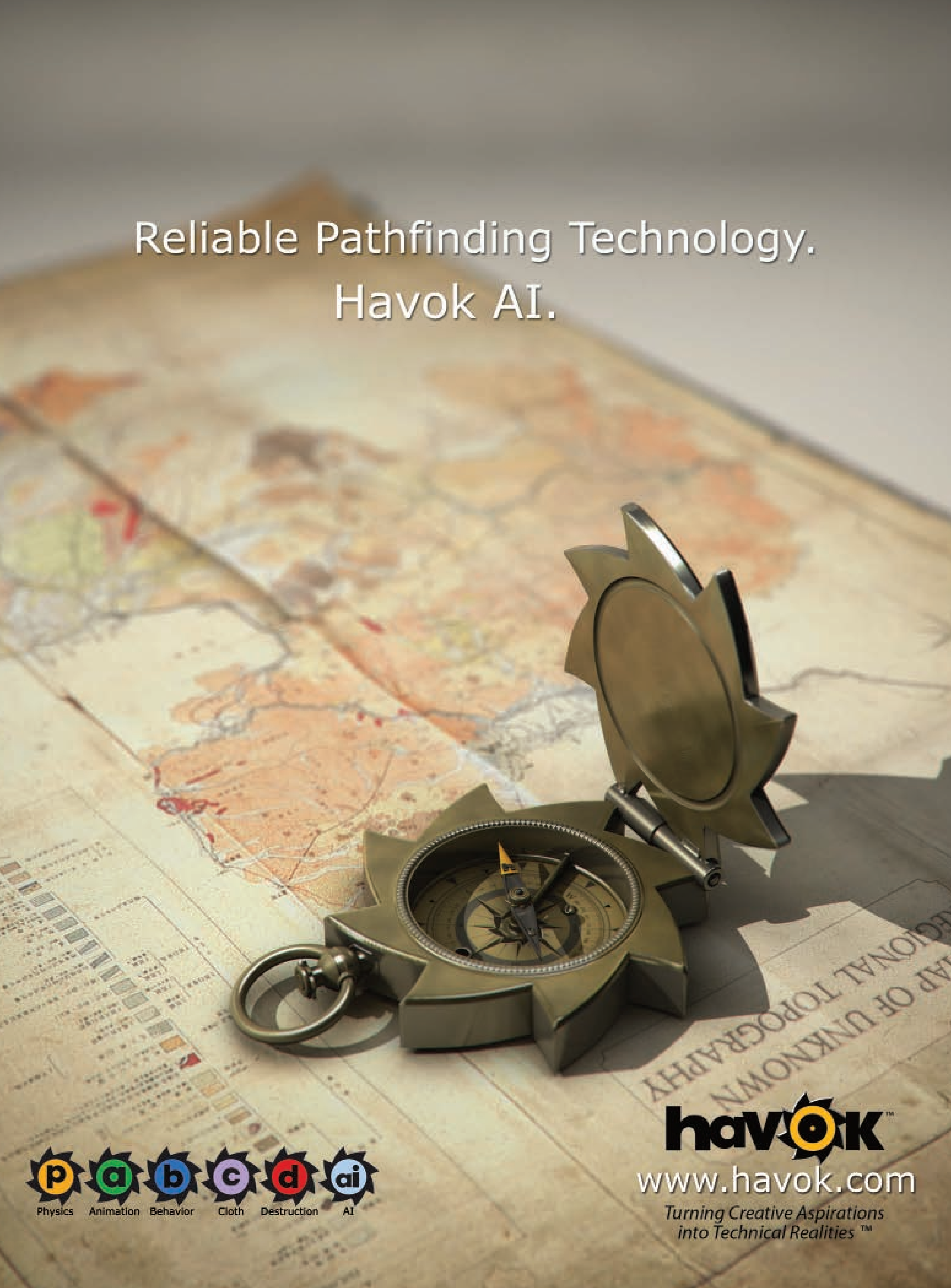
Re: Re: Re: Re: Re: Re: Re: HEAT DISTORTION!!

STOP REPLYING ALL
WHAT IS WRONG WITH YOU PEOPLE

Re: Taco truck

It's here!!

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


Reliable Pathfinding Technology.
Havok AI.
IONAL TOPOGRAPHY
AP OF UNKNOWN
p
Physics
a
Animation
b
Behavior
c
Cloth
d
Destruction
ai
AI
havok™
www.havok.com
Turning Creative Aspirations into Technical Realities ™