Issue 29

February 2009

DEV.MAG

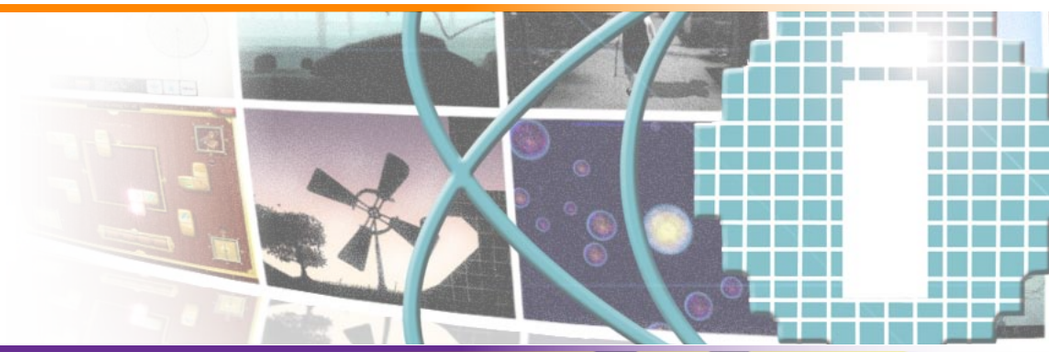
CREATE · DEVELOP · EXPERIENCE

INSIDE:

Out and about at the EuroGamer Career Fair O More Dev! Collision detection, Flash, Blender and Narrative O Reviews: Whodunnit O We look at the IGF finalists O Loads more!

CONTENTS

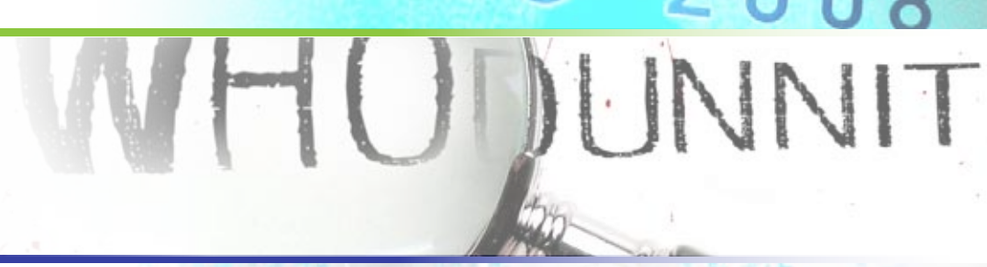
REGULARS



OPINION

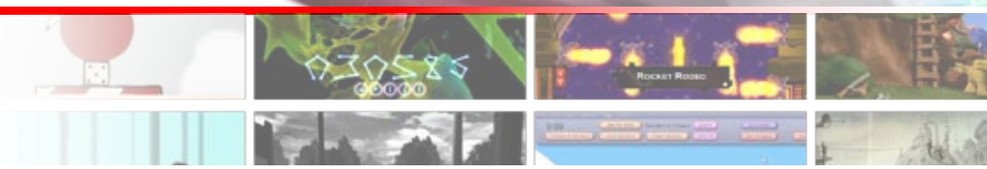
FEATURE

REVIEWS



DEVELOPMENT

TAILPIECE



CLICK ON THE PAGE NUMBERS TO NAVIGATE TO THAT PAGE

EDITOR-IN-CHIEF
Claudio "Chippit" de Sa

COPY EDITOR
James "Nighttimehornets" Etherington-Smith

LAYOUT AND DESIGN
Quinton "Q-Man" Bronkhorst

CONTRIBUTORS
Rodain "Nandrew" Joubert
Chris "LionsInnards" Dudley
Cathy "Kensei" Knights

WEBSITE MANAGER
Robbie "Squid" Fraser

WEBSITE
www.devmag.org.za

EMAIL
devmag@gmail.com

This magazine is a project of the South African Game.Dev community. Visit us at: www.devmag.org.za

I'm struggling not to think of the veritable horde of games I went through in the process of creating our extensive tailpiece this month. I see Dyson in my paragraphs, Cletus Clay in my sleep, Snapshot and Osmos in my article outlines. It's all rather disconcerting. But it's really nice to have a sudden flood of games all in one place, so we can find things to talk about every month. And we'll probably be talking about this lot for quite some time; some of them in particular look very promising. After all, there's always a World of Goo, Braid, Castle Crashers, Darwinia or a Savage in there every year, if you're sharp enough to spot it. Have a gander at our tailpiece and see if you can call out the big hits this year. Also in this month's issue, you'll find the special **return of our Blender series** with our article on getting started with the Blender Game Engine. We also have a special piece from one of our writers abroad who took the opportunity to attend **Eurogamer's Career Fair** and talk to some of the developers there about getting into the business. Have a look at what they're said. There's quite a lot of useful information there for budding developers who are looking to break into one of the 'big' studios.

Finally, we've got some **big changes on the horizon.** The magazine's format will be changing rather drastically over the course of the next two or three months, if all goes well. When it happens you'll know. It'll be hard to miss. Until then, well, it's going to be busy around the virtual Dev.Mag office – or, rather, it would be if we had an office. Until we have one, we'll be clogging up the IM gateways with nefarious plans for Indie World Domination. Nobody really wants to hear that, so we're hoping somebody out there will get upset enough with it and acquire a proper venue for us. And no, that's just me being silly and not a hint about the Big Change. Seriously.

'till next month.

~ Claudio

World of Goo oozes into top PC game sales

http://www.shacknews.com/onearticle.x/56930

During the week of January 11, World of Goo made a remarkable debut into 10th spot in NPD's PC gaming sales figures. That's right, this is the same list that has World of Warcraft: Wrath of the Lich King, Fallout 3 and Call of Duty: World at War. In fact, World of Goo edged EA Games' Red Alert 3 out of the list. Score one for the indies!

Toribash muscles onto WiiWare

http://blog.toribash.com/?p=81

The unique freeware fighter and old IGF finalist that puts players in control of individual muscles and body parts (often resulting in the most hilarious conflicts) may be a familiar name for PC gamers; well, one-man-developer house Nabi Studios hope to make this true for Wii players as too. A port of the game has been under development for quite some time, but, while no release date has been given, it appears mostly functional already. We may just see this in the near future.

IGF mobile finalists

http://www.igfmobile.com/

IGF isn't only for console and PC game entries. Handhelds feature as well, and those finalists under the banner of the Next Great Mobile Game award – on platforms ranging from humble J2ME to the beefy iPhone to the innovative NDS – have been announced recently. Who will clinch the largest part of the $30 000 total Mobile game prize pool?

Mo'Minis Studio is Game Maker for mobiles?

http://www.mominis.com/

Geared to facilitate rapid creation of mobile games, Mo'Minis Studio is a free development tool that looks to make mobile game development as simple as possible, for as many mobile platforms as possible. While it is currently limited to a drag and drop interface only, a scripting language is under development, something which hopes to extend the usability and flexibility of the software even further.

Happy Code for a Happy Coder

Rodain "Nandrew" joubert

I must confess: I hardly ever comment my code. It's the sort of thing that entire lectures are dedicated to – it's important for clarity, ease of use and even national security in the event of nuclear war. This is all true (though perhaps the nuke idea was a bit of a thumb suck) and I will always advise others to write miniature essays within their game code. Unfortunately, I fail horrendously when I need to do it myself.

I'm well aware of the temptation to postpone commentary, skip it for minor sections, or abandon the whole idea due to a sudden irrepressible urge to pee. Of course, such thinking tends to be rather 'shoot-in-the-foot-ish' – experience proves this. These mistakes are continually made by many eager developers. They seem as unavoidable as going off to play Guitar Hero when one is supposed to be writing an opinion column for Dev.Mag.

In a recent game prototype of mine – an Incredible Machine tribute containing little force balls and the Photoshopped head of an unwilling friend – I decided to see what would happen if I threw my own personality into the programming comments. Gone were the dry and crusty "//algorithm x for operation y" explanations – I took it upon myself to slip the colloquial, the abstract and the downright weird into my non-compilable statements. The results were amazing. I immediately felt motivated to comment my work more thoroughly; to think of something witty and interesting for each little code block that I came across.

I brought the "LOL" back into my code. I injected some fun into tired tasks, and not only did I comment more frequently, but I started to enjoy the coding process more. Heck, none of us became programmers so that we could be bored stiff by our duties – it just happens to be an unfortunate reality that we end up frequently trawling through menu generation and file-handling procedures when we want to be doing something really cool instead.

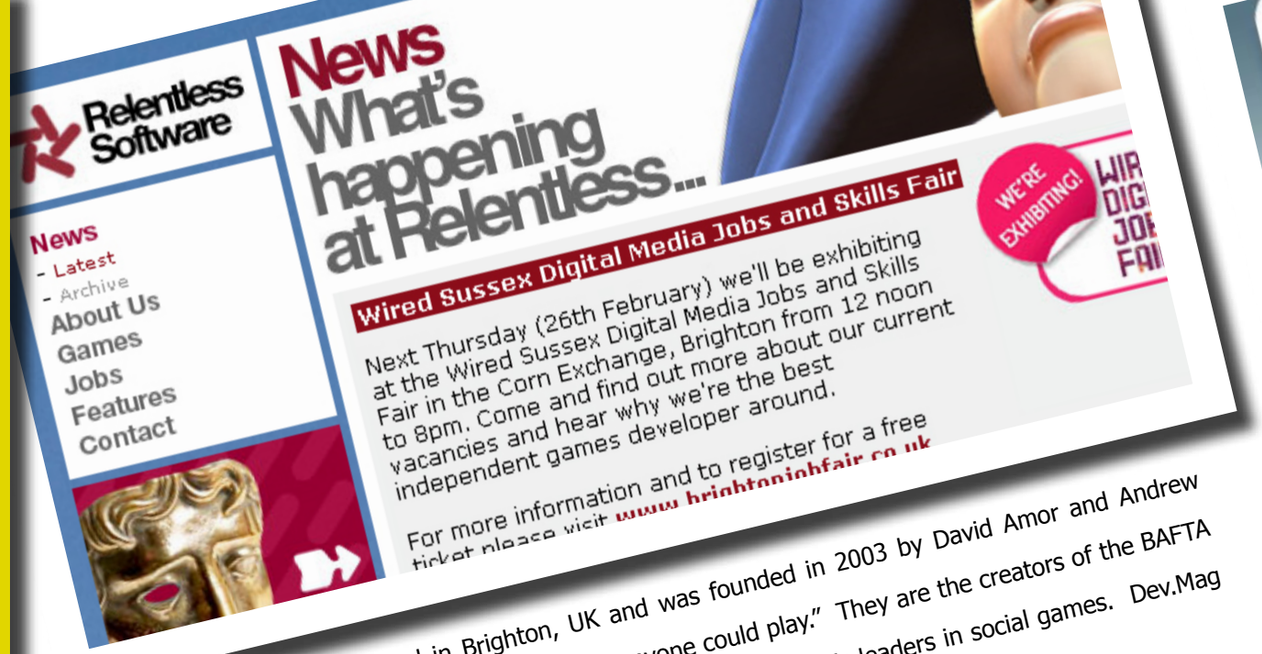
So if you're struggling to comment properly or are lurching through tedious code, try to let go a bit and have a little fun with your work. Make wry remarks about your data structures. Express frustration, joy or relief with individual blocks. Maybe even take a step outside the comment zone and start renaming a few variables or functions instead. It's your program. Do whatever the heck you want with it.

I'm not suggesting that you make your work incomprehensible – far from it. You just need to realise that there's more to coding than tiresome programming platitudes. Have a bit of fun, put a little life into your work, and remember that if you do something often enough, you ought to at least enjoy it.

I brought the "LOL" back into my code. I injected some fun into tired tasks, and not only did I comment more frequently, but I started to enjoy the coding process more. Heck, none of us became programmers so that we could be bored stiff by our duties – it just happens to be an unfortunate reality that we end up frequently trawling through menu generation and file-handling procedures when we want to be doing something really cool instead.

In London, they don't have a weekend of gaming – they have an entire week of festivities. From the 22nd of October to the 2nd of November 2008 the annual London Games Festival took place. Part of this momentous event was the Eurogamer Career Fair, at which European (and one Canadian) game companies come to seek fresh meat for their game development companies. Dev.Mag was fortunate enough to attend the event and interview representatives from various game studios and ask them a few questions.

Cathy "Kensei" Knights



Relentless Software is based in Brighton, UK and was founded in 2003 by David Amor and Andrew Eades, with the mission to "create games that anyone could play." They are the creators of the BAFTA award winning Buzz! franchise, and are considered one of Europe's leaders in social games. Dev.Mag spoke to game designer Jez Harris.

Relentless Software

What do you look for in potential employees?

A degree, a team player and a passion for making games.

Is providing a portfolio important?

Extremely important. People must be able to demonstrate that they are able to create games.

What is the best way to get into the game development industry?

Start at a low level and work your way up. Anything to get experience in the industry.

What are the highs and lows of game design?

The highs are you are making games. Making games is like the new astronaut, everybody wants to do it! The lows are that it is highly stressful at times. There are times when you cannot get something to work

How does an aspiring game developer improve their skills?

Work out what you want to do in the industry and stick to it.

Do formal degrees in Game Development take precedence over degrees in Computer Science?

Nothing beats real experience.

Founded in 2002, this Scottish game development company is a powerhouse in the industry, boasting over 200 staff, many with pedigree such as the Lemmings and Grand Theft Auto franchises. RTW is the creator of Xbox 360 exclusive Crackdown, and they aim to repeat their success with their upcoming title, APB.

Realtime Worlds

What do you look for in potential employees?

A university degree; either in Maths or Software Engineering.

Is providing a portfolio important?

Always.

What is the best way to get into the game development industry?

Applicants should have a degree.

What are the highs and lows of game design?

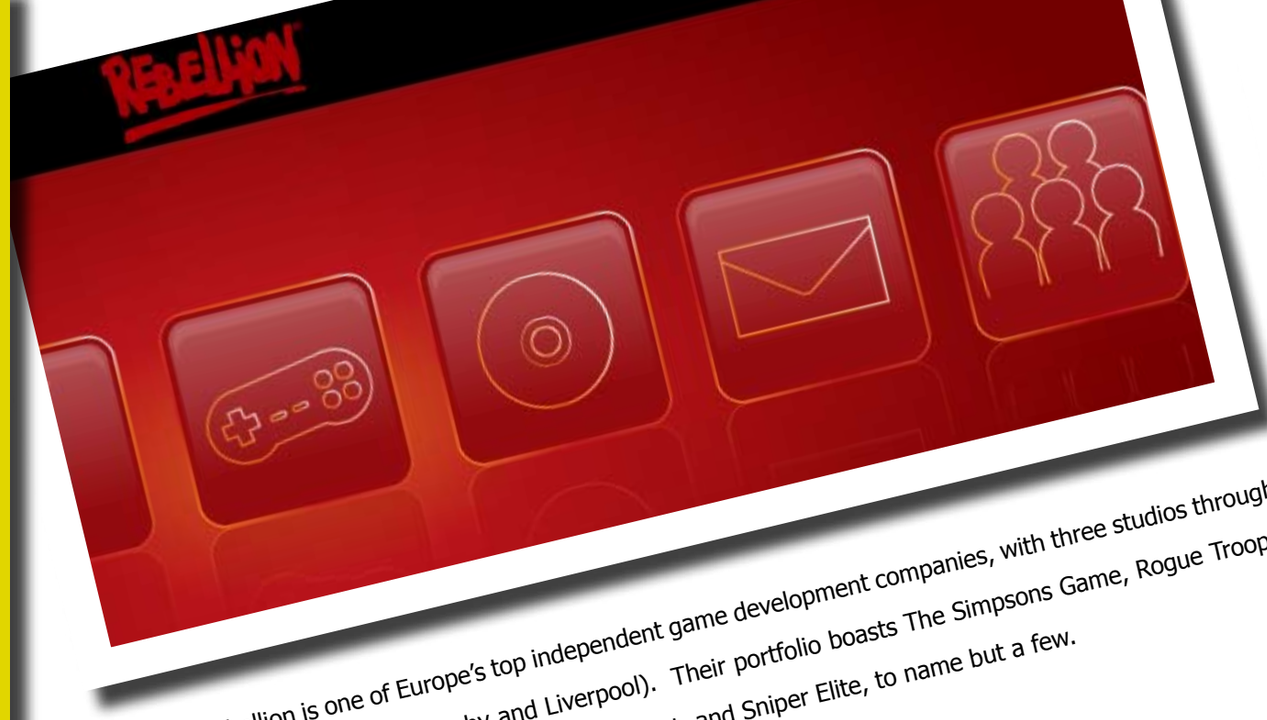
The high is seeing your game make the top ten. The low is when parts of the game get cut out that you spent a lot of effort creating.

How does an aspiring game developer improve their skills?

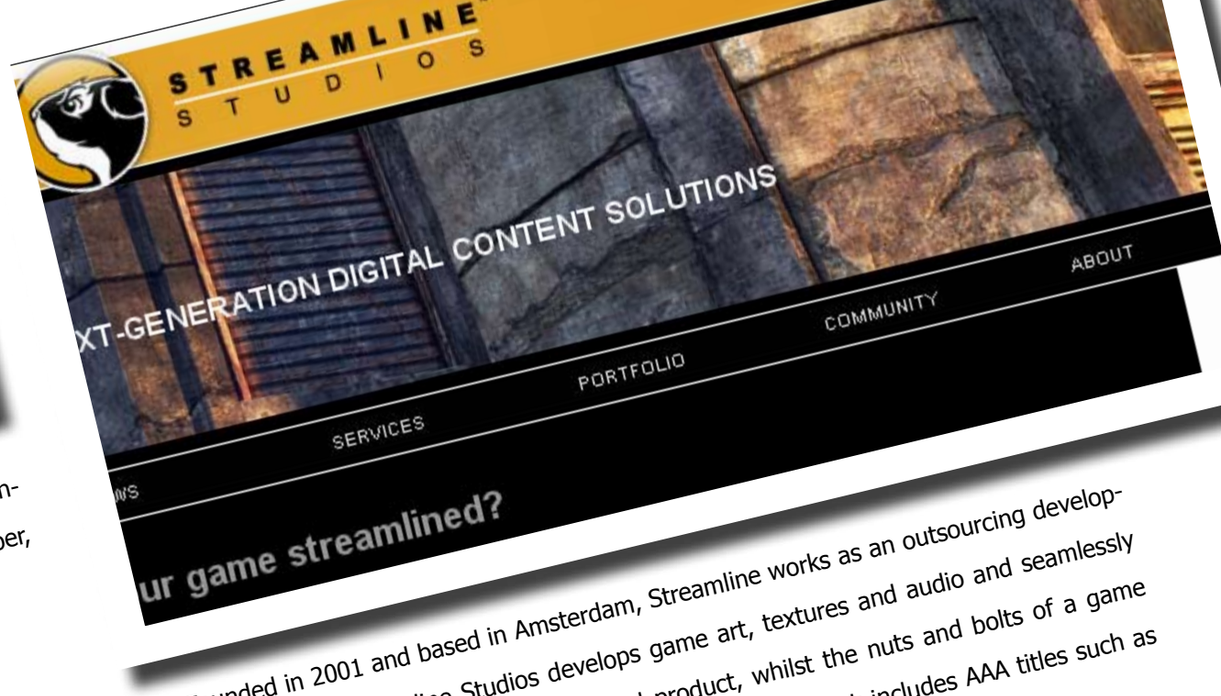
Practice.

Do formal degrees in Game Development take precedence over degrees in Computer Science?

No. Maths or Software Engineering is better.



Rebellion is one of Europe's top independent game development companies, with three studios throughout the UK (Oxford, Derby and Liverpool). Their portfolio boasts The Simpsons Game, Rogue Trooper, Star Wars: Battlefront, and Sniper Elite, to name but a few.



Founded in 2001 and based in Amsterdam, Streamline works as an outsourcing development house. Streamline Studios develops game art, textures and audio and seamlessly integrates their content to create a final product, whilst the nuts and bolts of a game engine are developed by their partner company. Their work includes AAA titles such as Saints Row, Gears of War, Unreal Tournament 3 and Ghost Recon 2.

Rebellion

What do you look for in potential employees?

We look for candidates who have C++ experience.

Is providing a portfolio important?

Yes, very important. Even if they are demos, works-in-progress, or short five minute things, anything that we can download, play and look at the source code.

What is the best way to get into the games industry?

C++ experience for developer jobs is important.

Streamline Studios

What do you look for in potential employees?

An open mind, a passion for game making and an awareness of the industry as a whole.

Is providing a portfolio important?

Yes. Works-in-progress are also important, as we can see how you got to the final product.

What is the best way to get into the game development industry?

Network with those in the industry. Post on forums, such as CG Society [or Game.Dev! - ed].

What are the highs and lows of game design?

The highs include recognition, admiration of your work and sense of completion. You get to work with a great team. The low is crunch time.

How does an aspiring game developer improve their skills?

Seek feedback from community members and stay active on the forums.

Does a degree in Game Development take precedence over a degree in Computer Science?

A degree is not always applicable. Nothing beats good experience.

Starbreeze operates as a development partner for publishers who have secured finance for their projects. This Swedish game development house is known for The Darkness, Chronicles of Riddick: Escape from Butcher Bay, and Enclave.

Founded in 1999, this German company is known for Crysis and Far Cry. They are "dedicated to the creation of high quality video games for the PC and next generation consoles" and have expanded to the UK, Hungry, Ukraine and Bulgaria.

Starbreeze Studios

What do you look for in potential employees?

Experience and a proficiency in their area of expertise.

Is providing a portfolio important?

Very important.

What is the best way to get into the game development industry?

Start in quality assurance, and branch out. For example, Hugo started as a coffee boy and has worked his way up to QA Manager.

What are the highs and lows of game design?

The highs are the parties. We celebrated our tenth anniversary recently. The lows include the dreaded crunch time and, for some, not seeing their family for a while.

How does an aspiring game developer improve their skills?

Practice.

Does a degree in Game Development take precedence over a degree in Computer Science?

Depends on what you want to do. It is important to have a speciality, not just in game design, but in normal development.

Crytek

What do you look for in potential employees?

Pure talent. A knowledge of the industry and what is going on.

Is providing a portfolio important?

Yes.

What is the best way to get into the game development industry?

Pick a discipline and stick to it. Be prepared to start at the bottom and work your way up.

What are the highs and lows of game design?

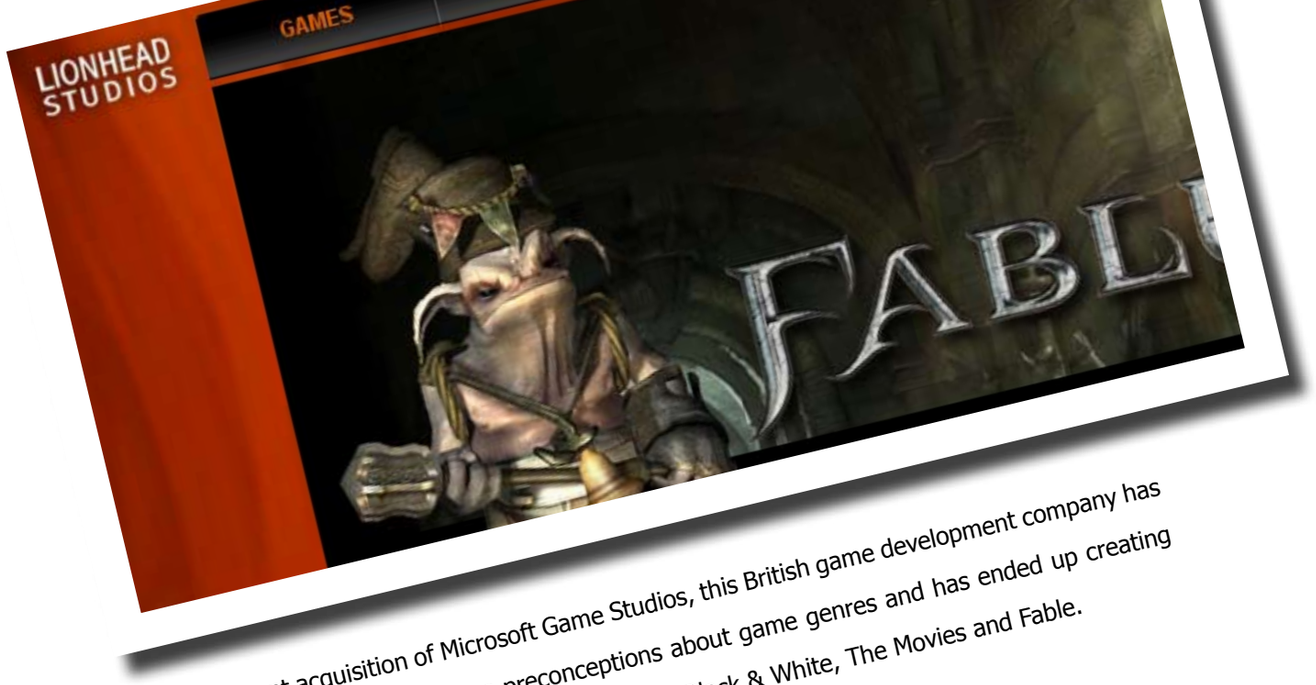
The highs include working in a fast moving industry and getting recognition for a job well done. The lows are that it is still a job and you have to work. It is also important to keep on top of what you are doing.

How does an aspiring game developer improve their skills?

Practice. Read a lot and seek criticism from those in the industry.

Does a degree in Game Development take precedence over a degree in Computer Science?

Not really applicable, talent is more important.



A recent acquisition of Microsoft Game Studios, this British game development company has always challenged existing preconceptions about game genres and has ended up creating their own genres. Games include Black & White, The Movies and Fable.

So that is it, guys and gals, get those portfolios out there, be prepared to start as a grunt and work your way up and, most importantly, keep making games.

Lionhead Studios

What do you look for in potential employees?

An academic background is preferred, but we look for the full package. The applicant needs to be a team player, dedicated and have a good work ethic. We like to see examples of code and games created.

Is providing a portfolio important?

Definitely.

What is the best way to get into the game development industry?

Focus on what you want to do and go for it. It is also important to realize that it is a niche industry.

What are the highs and lows of game design?

The highlight is that you are making games! The lows are crunch time and, because it is a young industry, it is always changing.

How does an aspiring game developer improve their skills?

Get work experience and learn from others.

Does a degree in Game Development take precedence over a degree in Computer Science?

Pure degrees are always better than game design courses.

WHODUNNIT?

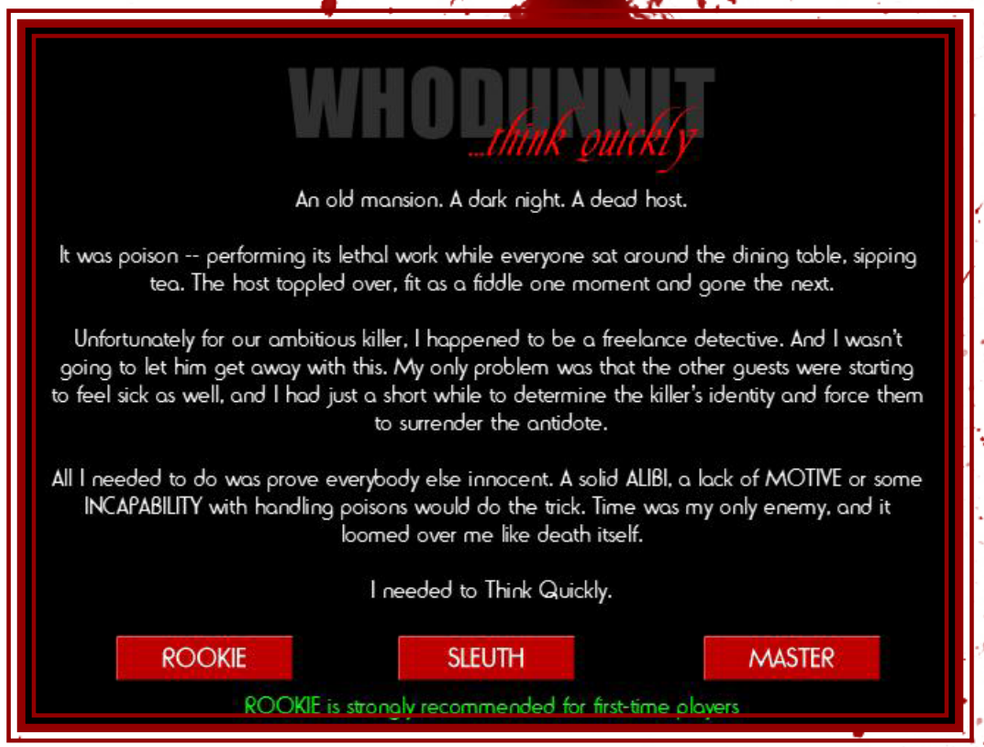
Chris "LionsInnards" Dudley

On the list of "Top Ten Reasons to Admire Rodain "Nandrew" Joubert," coming a close second to his fabulous hair, would be his tendency to consistently roll out absolutely fantastic and innovative games. "Whodunnit…Think Quickly" is no exception.

Whodunnit transposes the player into the role of a tweed jacket wearing a private eye, who is attending a dinner party hosted at a mansion. Whilst the P.I. and a number of guests (which varies depending on the difficulty selected) are busily tucking into their meals, the host suddenly plops face first into his plate! The brilliant detective deduces that he is dead (the living tend to breathe more) and the player now has to sniff out the killer before another succumbs to the poison coursing through everyone's veins.

The fact that one of your character's abilities is taking a smoke break is a testament to the tense experience of the game. The player needs to pose the right questions to the right people, and keep track of all the information handed to them. Being a cerebral experience, the game is played solely with the mouse, with a simple (if not slightly bland) interface displaying all you need to know.

As much fun as this puzzler is, that is not where the true appeal lies. The charm is the atmosphere, as odd as that sounds for a game which is essentially just a collection of buttons and a timer. Nandrew has put a few touches to the game that heighten the experience. Panic and pressure mount up as the timer ticks down; as the player wrongly accuses a guest; or as the prime suspect drops dead. It's a wonderful demonstration of how much atmosphere can be created with limited resources, and it's a great way to pass the time. The ten minute time limit (a literal take on the theme of the Game.Dev competition 17, into which this game was entered) makes it especially suited to coffee breaks and as a quick distraction from work.



When Worlds Collide

Basic Collision Detection in 2D – Part II

Last month we covered some of the very basic ways of testing whether object A hits object B. But, while the techniques we covered could quite likely take you very far, you'll inevitably encounter a time when they simply aren't enough. So this month we'll go a little further.

Claudio "Chippit" de Sa

Line-Line intersection test revised

Occasionally, you'll need slightly more information from your collision tests. Many physical applications will require the precise coordinates of the intersection between lines. Thankfully, this is the easiest test to get that information.

Referring back to last month, we had defined P_a and P_a as points on the respective lines that we were testing as below:

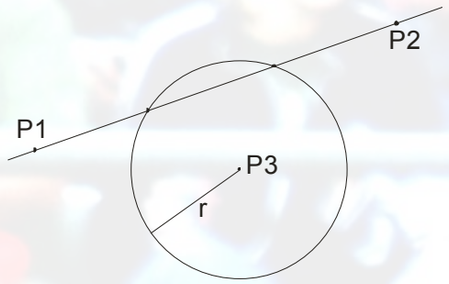
$$P_a = A_1 + u_a (A_2 - A_1)$$
$$P_b = B_1 + u_b (B_2 - B_1)$$

We then solved for points u_a and u_b , or, rather, simply for their denominators to determine their existence. To calculate the precise point of intersection, you'd simply have to calculate and substitute one of these values into the equations for P_a or P_b (it doesn't matter which, they'd be identical) to get your answer. Adjusting our code from last time results in the following:

```
LineLineCollision
Input
    LineA1    Point     First point on line A
    LineA2    Point     Second point on line A
    LineB1    Point     First point on line B
    LineB2    Point     Second point on line B
Output
    The point of the collision, or null if no collision exists.

Method
    denom = ((LineB2.Y - LineB1.Y) * (LineA2.X - LineA1.X)) -
         ((LineB2.X - LineB1.X) * (LineA2.Y - LineA1.Y))

if (denom == 0)
     return null
else
     ua = (((LineB2.X - LineB1.X) * (LineA1.Y - LineB1.Y)) -
          ((LineB2.Y - LineB1.Y) * (LineA1.X - LineB1.X))) / denom
     /* The following 3 lines are only necessary if we are check-
     ing line segments instead of infinite-length lines */
     ub = (((LineB2.X - LineB1.X) * (LineA1.Y - LineB1.Y)) -
     ((LineB2.Y - LineB1.Y) * (LineA1.X - LineB1.X))) / denom
     if (ua < 0) || (ua > 1) || (ub < 0) || (ub > 1)
          return null

     return LineA1 + ua * (LineA2 - LineA1)
```

Circle-line intersection test

This is another common test that should find a use in many games. Did your ball collide with the wall of the level, or a barrier set up by the enemy? Things like that require a circular object to be tested against a line.



This test involves slightly longer equations than previous intersections, but is still in familiar algebraic and geometric ground. And, in fact, the expressions may be simplified greatly by the simple act of working in the circle's local space, effectively removing all circle-centre variables from the equations by setting them all to zero. This is achieved simply by expressing the two coordinates representing the line as positions relative to the centre of the circle. Doing this makes your calculations and equations far simpler.

Using the familiar equation, we define a point P on the line as: $P = P_1 + u(P_2 - P_1)$ with ua being any real number, and P1 and P2 being represented in the circle's local space

And we define the sphere centred on the origin as:

$$x^2 + y^2 = r^2$$

Substituting the equation of the line into the equation for the circle results in an equation in the following familiar form:

$$au^2 + bu + c = 0$$

Where a , b and c are:

$$a = (x_2 - x_1)^2 + (y_2 - y_1)^2$$

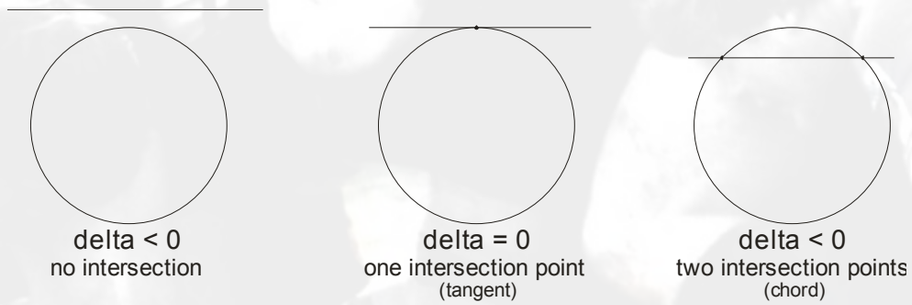
$$b = 2(x_1(x_2 - x_1) + y(y_2 - y_1))$$

$$c = x_1^2 + y_1^2 - r^2$$

Recalling some old high school algebra, the solution to such a quadratic equation is determined by the quadratic formula:

$$u = \frac{-b +/- \sqrt{b^2 - 4ac}}{2a}$$

And the existence of a solution is defined by the delta, or the expression under the square root: $b^2 - 4ac$



If this expression is less than 0, there is no intersection, if it exactly 0, the line is a tangent to the circle, and if it is greater than 0, the line intersects at two points.

Once this has been determined, you can substitute the value(s) for u into the equation for the line to determine the exact coordinates of the intersection. And, similar to the line-line intersection test, you can use this u value to determine whether or not the intersection occurs on the line-segment between points P_1 and P_2.

In pseudocode, this technique may be implemented as follows:

```
CircleLineCollision
Input
    LineP1    Point     First point describing the line
    LineP2    Point     Second point describing the line   CircleCentre
Point    The centre of the circle
    Radius    Floating-point The circle's radius
Output
    The point(s) of the collision, or null if no collision exists.

Method
    // Transform to local coordinates
    LocalP1 = LineP1 - CircleCentre
    LocalP2 = LineP2 - CircleCentre
    // Precalculate this value. We use it often
    P2MinusP1 = LocalP2 - LocalP1

    a = (P2MinusP1.X) * (P2MinusP1.X) + (P2MinusP1.Y) * (P2MinusP1.Y)
    b    = 2 * ((P2MinusP1.X * LocalP1.X) + (P2MinusP1.Y * LocalP1.Y))
    c = (LocalP1.X * LocalP1.X) + (LocalP1.Y * LocalP1.Y) - (Radius * Radius)

    delta = b * b - (4 * a * c)

    if (delta < 0) // No intersection
         return null;

    else if (delta == 0) // One intersection
         u = -b / (2 * a)
         return LineP1 + (u * P2MinusP1)
         /* Use LineP1 instead of LocalP1 because we want our answer in glob-
al space, not the circle's local space */
    else if (delta > 0) // Two intersections
         SquareRootDelta = sqrt(delta)

         u1 = (-b + SquareRootDelta) / (2 * a)
         u2 = (-b - SquareRootDelta) / (2 * a)

         return { LineP1 + (u1 * P2MinusP1) ; LineP1 + (u2 * P2MinusP1)}
```

That's as far as we'll go for this tutorial. The above techniques should be able to solve most of your problems. With clever application of the line-line segment intersection test, you could even determine the intersections between any polygons, even transformed ones. Most specific problems have more efficient solutions, but that one can solve most of them in a simple, though perhaps inefficient, way.

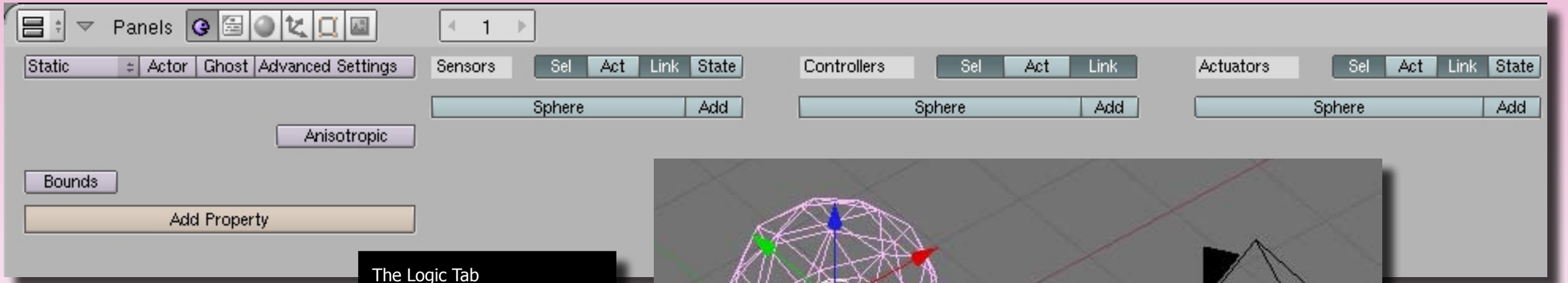
Good luck, and make great games without needing to compromise collision accuracy anymore. And remember, when in doubt, as long as it looks fair to the player, you can usually get away with a few tricks.

Blender

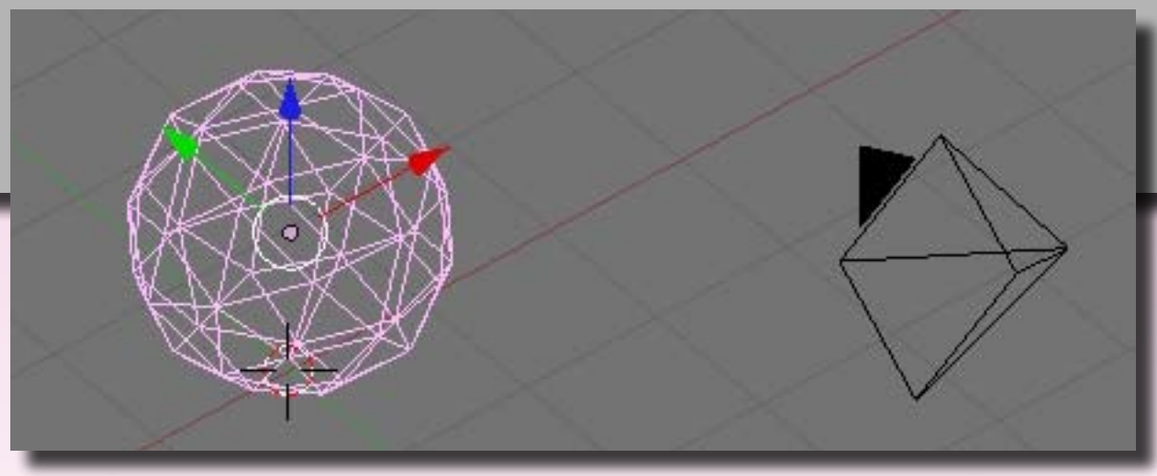
Introduction to Blender Game Engine

Regular readers of our Blender series will have dabbled to a small degree in the Blender Game Engine when we covered rigid body physics. This article will assume some familiarity with these previous articles as we get started with Blender's Game Engine. For the sake of this introduction, we're going to limit the scope of the introduction to Logic Blocks only, without dabbling in Python scripting.

Claudio "Chippit" de Sa



The Logic Tab



Logic tab

Most basic object manipulation is handled on the logic buttons tab, which allows the game engine to respond to events on a per-object basis. After selecting any object in the 3D view, the tab will display options as illustrated (assuming Blender version 2.48 or newer).

The left-most part of the tab will already be familiar to regular readers as this where objects where configured for rigid body collisions, which was also, of course, handled inside the Game Engine itself. Conveniently, the easy availability of the real-time Bullet physics engine will be a boon to anyone planning to use the BGE for game development purposes.

The foreign remainder of the tab, the real meat of the game engine, is divided into 3 logical sections per selected object:

- Sensors are triggers that will activate actuators.
- Controllers allow you to tie any combination of sen sors to specific actuators using a choice of Boolean rules. (AND, OR, XOR, etc.)
- Actuators are actual game actions such as moving objects, spawning other objects, playing sounds and other effects.

Getting started

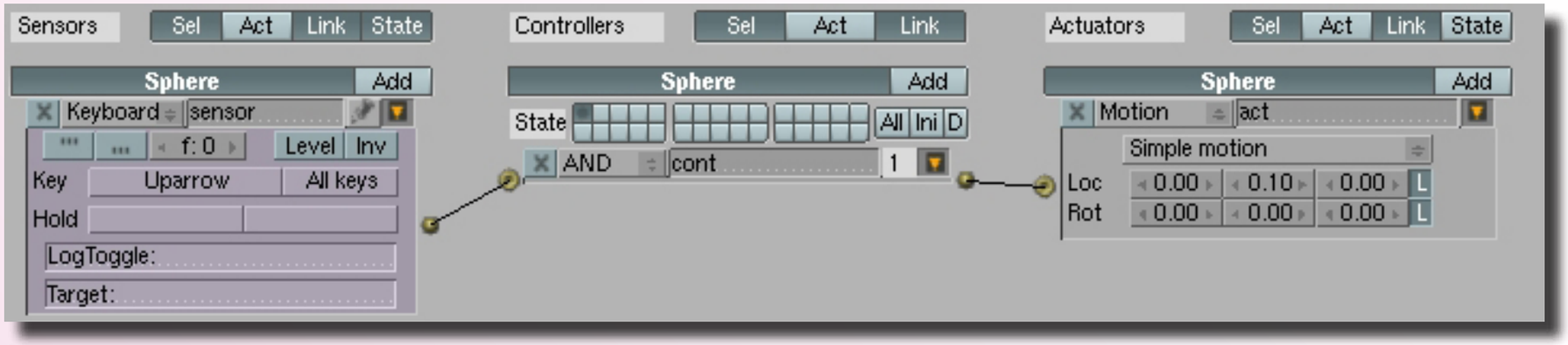
Time to get things moving around. Starting with a blank scene, add a sphere (either Ico or UV spheres will work) which we'll be controlling using the Blender Game Engine and its Logic Blocks.

Back in the logic buttons tab and with the new sphere selected, click the Add button under Sensors to create a new sensor for the object. By default, the sensor will trigger on every frame of the game, but since we want to be able to control the sphere via user input, we'll need to use a different kind of triggering event. Click the drop-down box next to 'Always', and change the sensor to a 'Keyboard' type. Click the box labelled 'key' among the revealed options and press the up-arrow on your keyboard.

Next, create a controller. The default controller is an AND-block, which is fine for our purposes now and can be left as such. The actuator is next. As mentioned above, actuators are actual effects performed on objects in the scene. We want to move our sphere around, so we'll need a 'Motion' actuator.

There are two different kinds of standard motion that can be used here. Change of location (translation), or rotation, and each along the three major axes: X, Y and Z from left to right. The 'L' button signifies whether the change should use local or global co-ordinates. Since we want the up button to move the sphere forward, we'll need a change along the sphere's local y-axis. Enter 0.1 in the second 'Loc' field.

Finally, we need to associate the sensor, controller and actuator with each other. Do this by dragging the connector icons between the sensor and the controller, and again between the actuator and the controller. When you're done, you should have something that looks like the image below.



Forward settings

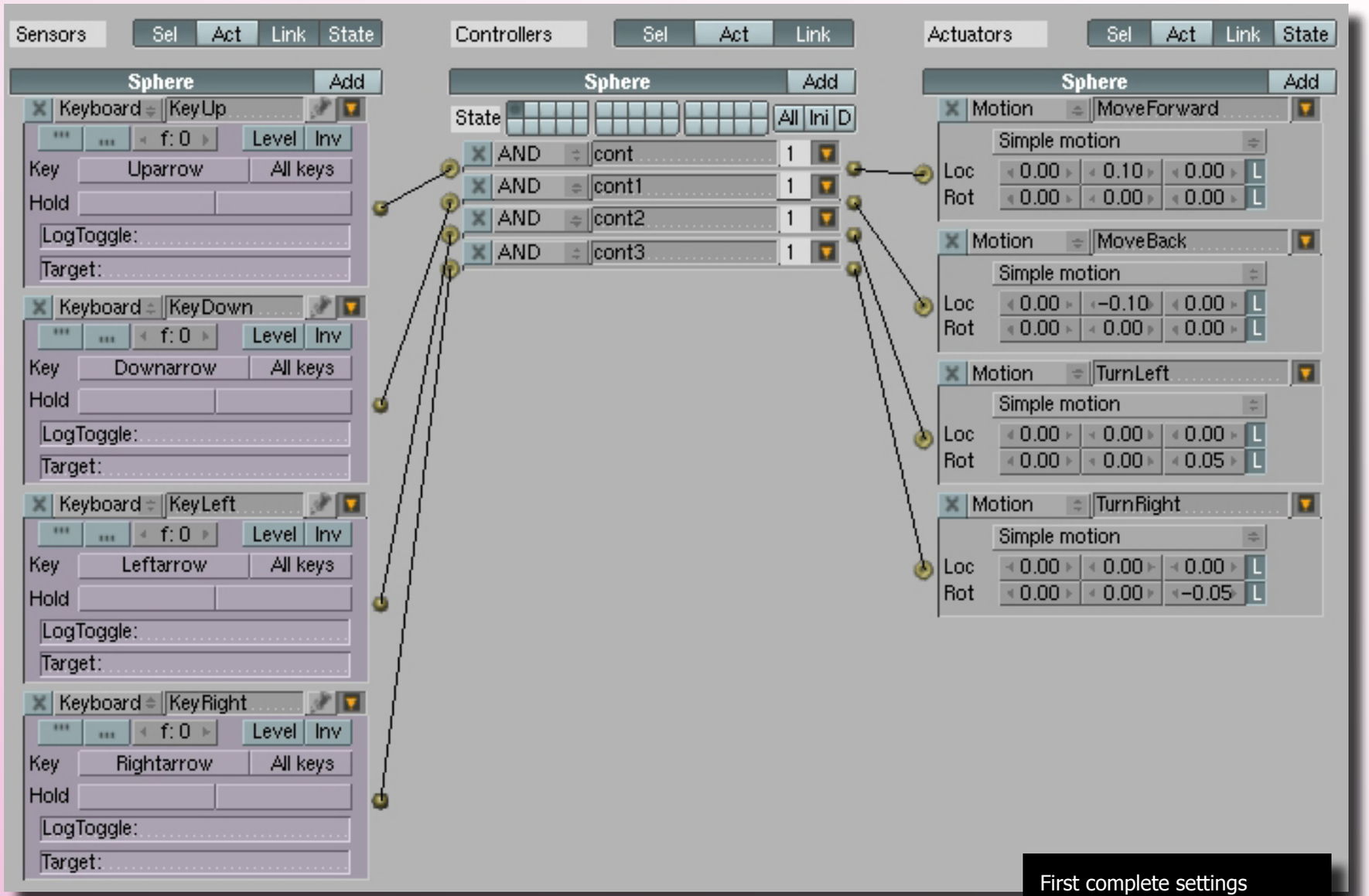
Basic movement

If you test our setup now by pressing P, you should find that you can control the sphere using the up-arrow on your keyboard, as expected. You can now repeat this process for down, left and right controls, using the values as below:

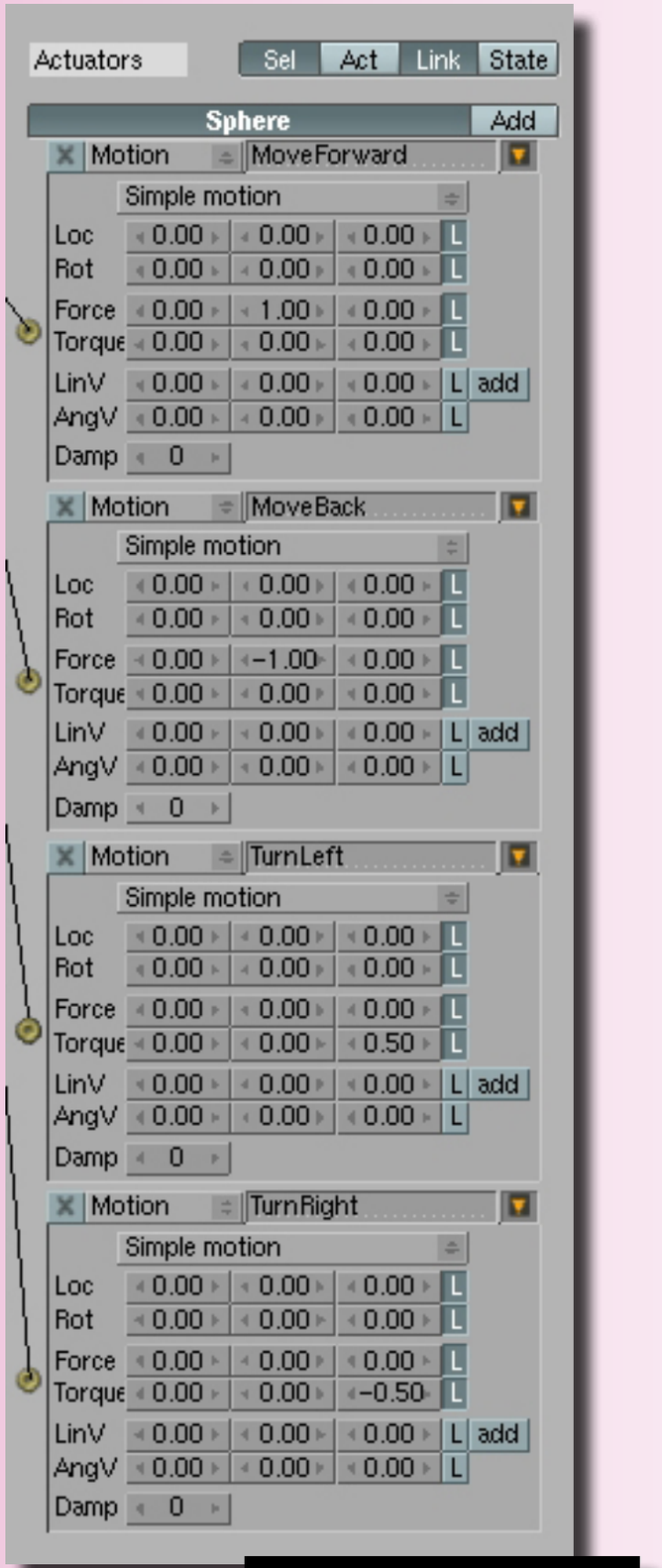
Down:	local Y-position	-0.10
Left:	local Z-rotation	+0.05
Right:	local Z-rotation	-0.05

You'll probably notice that things become rather cluttered at the bottom of your screen as you start adding more sensors and actuators. This is a good time to be aware of the ctrl-up keyboard shortcut, which will toggle the size of whichever panel the mouse is under to use the full window area, hiding other panels temporarily.

When you're done, your logic blocks should look like the image below. I've also taken the opportunity to name every sensor and actuator, which makes them much easier to identify when collapsed.



First complete settings



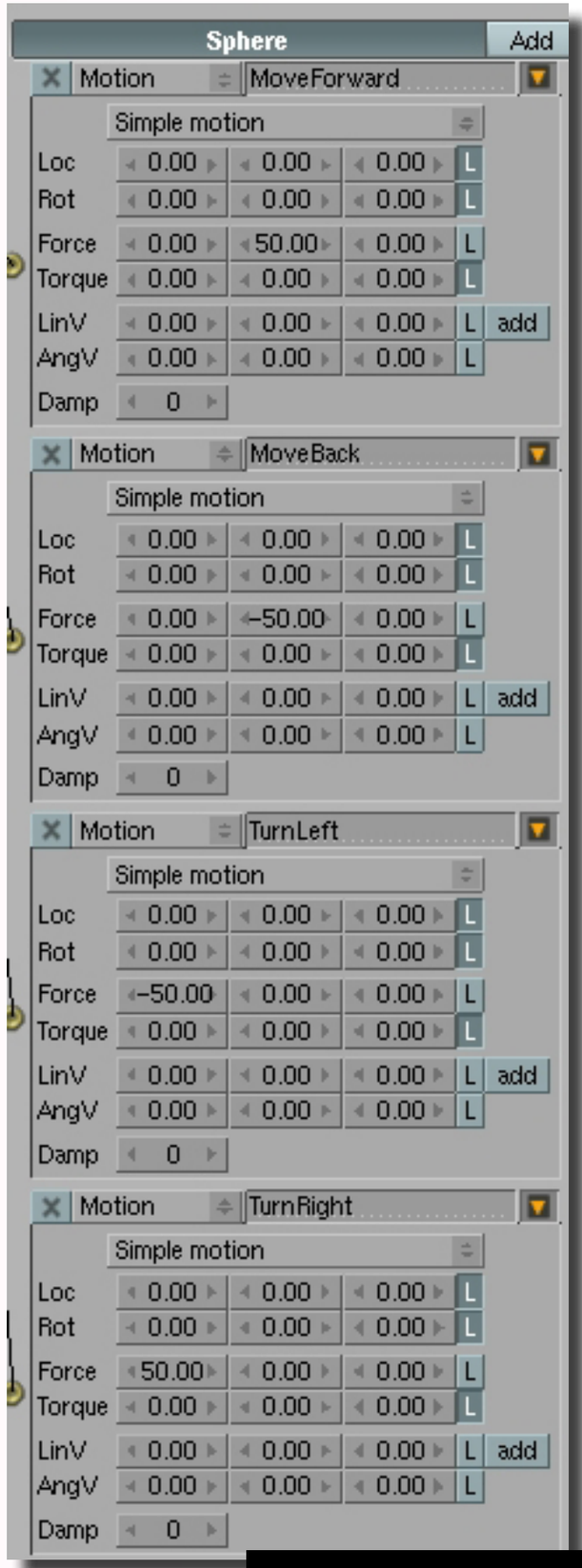
Physical Actuators

Proper physical movement

A keen eye would've noticed that the ball is simply gliding around in space, neither rolling nor colliding with obstacles. What happened to the physical simulation we were promised? For this, we'll have to refer back to some familiar settings that I touched on briefly in the introduction to the logic tab.

With the sphere selected, enable its rigid body dynamics, and set it to spherical collision. Doing so will reveal extra options in your motion actuators. You'll also need to add a fairly large plane to act as a floor, since the sphere will now be subject to gravity. Additionally, we'll need to change our actuators to actually apply a force to the sphere, instead of simply changing its co-ordinates (effectively teleporting it very small distances every frame) and causing it to appear to glide. Adjust your actuators as follows:

Once you're done, you'll notice that testing this reveals some unexpected behaviour. Because the sphere is now rotating as it moves, its local co-ordinates are always changing, and thus the y-axis isn't always aligned parallel to the ground and won't always be pointing forward. In order to counteract this, deselect the 'L' button on each force setting to set the forces to use world coordinates. With a few tweaks to the values for the sake of responsiveness, your sphere should be moving as expected.



Final Actuator Settings

What now?

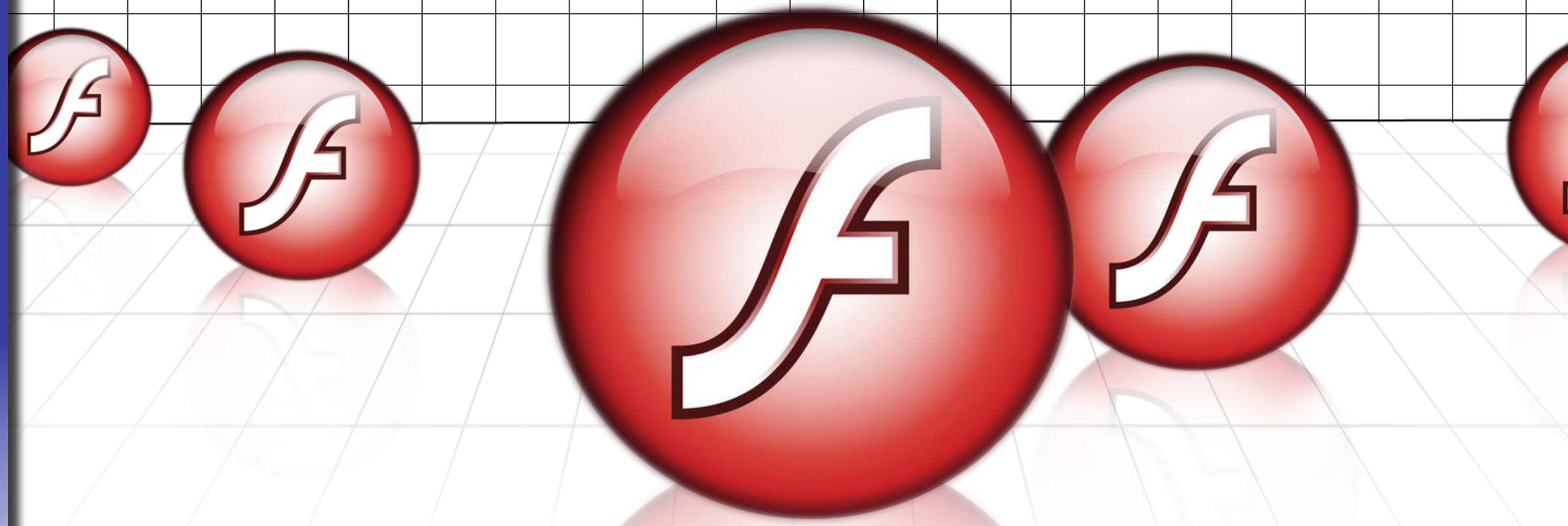
Now that you have a sphere that's controlled under bullet physics, you can add static and dynamic obstacles to the scene. Refer back to the article on rigid body physics for more information on the subtleties of setting up objects for this. Then you can litter your level with obstacles and debris that will be affected by your controllable object.

From here, you can use other logic bricks to respond to different game events, such as collisions or giving the ball the ability to jump (Tip: allow this only while the sphere is on the ground). That's all for the tutorial. Enjoy playing around with the game engine, and good luck with your creations.

Physics Playground

Object Pascal has a lot to offer...

www.PascalGameDevelopment.com



How does Flash Compare?

"Only the unskilled worker blames his tools"... or some malarkey like that. But anybody who has had experience with multiple development kits can testify that every programming language, design environment and creative suite has its own strengths and weaknesses. If you are looking at getting into Flash for game development, but aren't quite certain about whether or not it's the tool for you, here are a few features that may pique your interest.

Flash is geared towards games.

My interest in Flash as a development tool emerged at some point in 2008. Within a week, I had a functional game prototype. Within a month, I had entered a local development competition with a legitimate and reasonably complicated puzzle title. In short, it didn't take me very long to produce pleasing results with a tool that I had to get acquainted with from the ground up.

Flash was by no means created for games. It started off as a low-bandwidth method of getting smooth animations and other visuals onto the Web. However, as people started toying about with the tool and began using it in interesting and surprising ways, the product started to expand to cater for more interactivity and faster, more advanced graphics routines. Unlike many other standard development tools, Flash's core language (currently Actionscript 3) comes out of the box with support for quick and effective sprite manipulation. This is in stark comparison to other tools, which are usually quite painfully constrained to a set of "boring" components for use in standard windows and forms, requiring the user to hunt down specialised libraries before anything can happen.

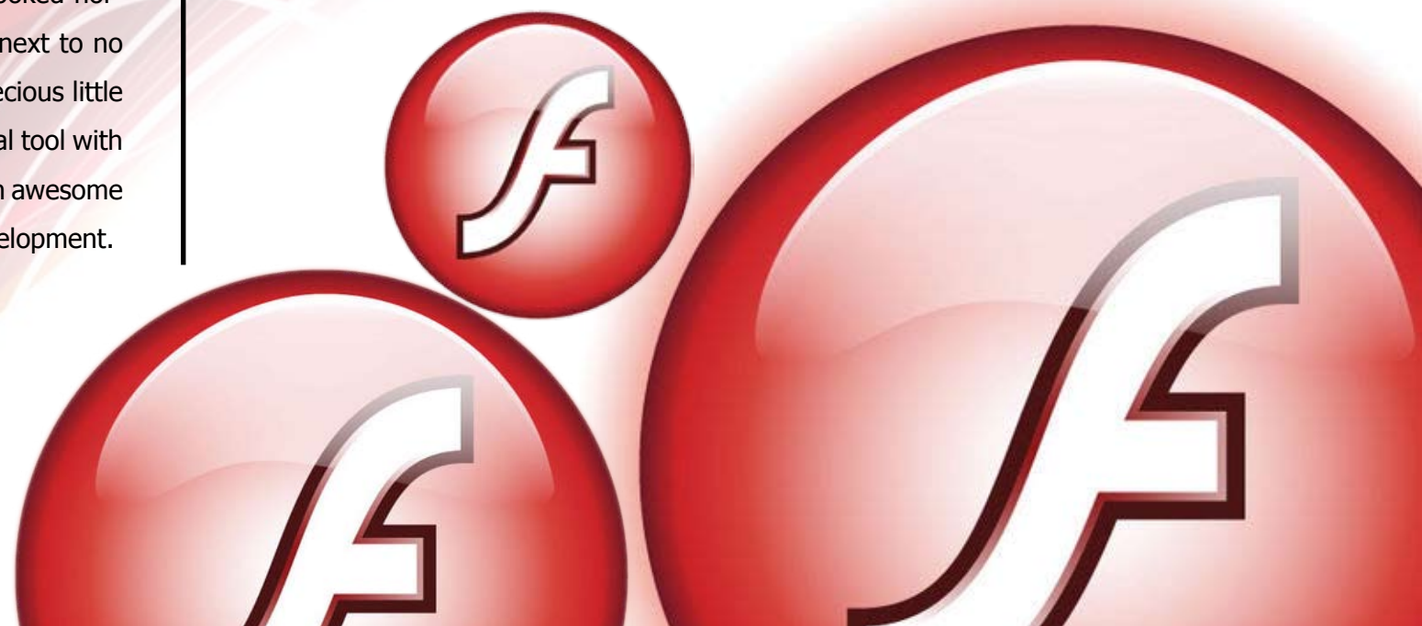
To clarify: most programming tutorials will have you writing a variation of the standard "Hello World". The first Flash tutorial I attempted required me to send a soccer ball bouncing around the screen. A few days later, I had to demonstrate Flash's capabilities as a game creator alongside coders who were using XNA and Game Maker. In one hour, I was able to put together a crude ant extermination simulator complete with rotating sprites, badly-drawn explosions, suitable collision detection and even a few sounds. It wasn't pretty (actually, it looked horrible), but it was made in next to no time by somebody with precious little experience. Flash is a visual tool with flexible results, making it an awesome environment for game development.

Flash has great drawing functions.

Some tools deal wonderfully with sprites and more common visual elements, but choke and fail horribly when it comes to setting up primitives such as shapes and lines. There are three main areas in which most programs can easily crash and burn: the speed of the draw; the flexibility of the primitives; and the ease of use. Flash, being an animation tool at heart, steps up to the mark and shines in this regard.

With my very first game project, I decided to test Flash a little and make the graphics entirely vector-based. That is, every single object in the game consisted of a bunch of interconnected lines, shapes and filled geometry. Of course, I didn't stop there. Objects were powered by crude, un-optimised mathematics, which I put in to allow rotation, colour gradients, transparency and even visual distortion created by elements placed in the game, such as gravitational fields. Flash didn't even break a sweat as it bore this weight, and it turned out that most of the functions which I had tried to code could actually be implemented with one or two built-in calls or value insertions.

The only disappointing factor was the relative difficulty of using text when compared to similar languages. Each bit of text had to have myriad parameters set – as well as its own class instantiation – before it was fit for the screen. I soon found myself forgoing it in favour of the easier-to-use graphics. Perhaps this is avoidable in most cases (and putting the effort into a method or two would probably sort everything out), but it still felt like a bit of an annoyance for a beginner.



Flash has great OO

Actionscript 3 uses a nicely developed object oriented programming structure. The program's entry point is at a standard main class, and from there any actors including sprites, text, shapes and advanced data structures are grouped into classes as well. All of these are declared and instantiated in the same way that you'd expect of any regular object.

Admittedly, the structure can present a slight twist for new users, but this is more of a learning issue than anything else, and once you get used to it, it ends up working very well. All non-static objects in your project ultimately stem from one parent: an object known as the "stage". It's important to note that the stage is not the same as the main class, and cannot be instantiated, replaced or destroyed – it exists as a default entity when you start coding. This means that there's no declaring of an initial "window" or similar framing class for OO projects, and it works out just fine once you understand it. For the most part, all you have to do is add children to the stage for them to show up in your final product.

As of Actionscript 3, Flash's garbage collection has become nothing short of sublime. All classes are periodically checked for a referential connection to the stage (however indirect) and if no such connection is found, they are removed from memory. AS3's collection method also checks properly for cyclical referencing (when classes form a daisy chain of references which can continue indefinitely, but which don't ever link to the primary stage) and ensures that there is no slowdown due to excessive checking. Of course, the coder is still responsible for making sure that classes are removed from the "stage" child list when no longer in use, but there is absolutely no need to go down through the generations and manually delete trees of memory-consuming data.

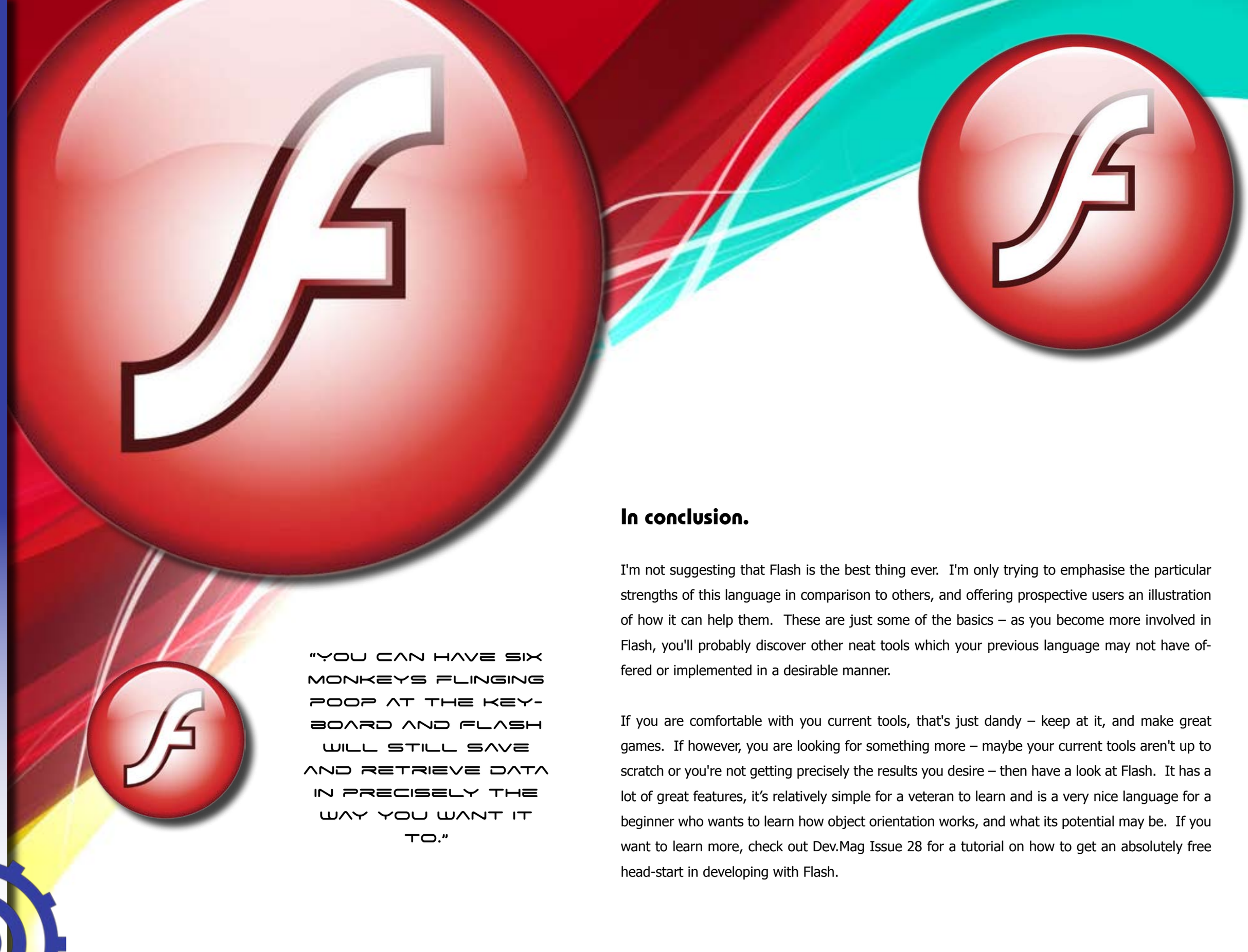
"CHECK OUT DEV.MAG ISSUE 28 FOR A TUTORIAL ON HOW TO GET AN ABSOLUTELY FREE HEAD-START IN DEVELOPING WITH FLASH."

Flash has reach.

If you pick Flash as a tool of choice, you've secured yourself a very easy way to access an audience across several platforms and devices. Most people with a modern web browser know and use Flash on a regular basis. The vast majority of online games are built in Flash. YouTube videos are made possible through the power of Flash. Entire websites driven by Flash are becoming more and more common.

Better still, Flash Lite has great market penetration on cellphones and other mobile devices. Although a bit more restrictive than using Flash for a PC, anybody who conducts sufficient research into the limits and specifications of mobile phones can easily craft something that is deployable to these small, but ubiquitous devices.

Best of all, no matter which platform you use, you won't ever need to worry about IO or file handling. All data is saved to a Flash construct known as the Local Shared Object (LSO). This feature is, for lack of a better term, absolutely unbelievable. It is dynamically-typed, updates itself automatically, and all of its data can be called up from a previous session with a single command. In other words, you can have six monkeys flinging poop at the keyboard and smashing the computer tower and Flash will still save and retrieve data in precisely the way you want it to.



"YOU CAN HAVE SIX MONKEYS FLINGING POOP AT THE KEYBOARD AND FLASH WILL STILL SAVE AND RETRIEVE DATA IN PRECISELY THE WAY YOU WANT IT TO."

In conclusion.

I'm not suggesting that Flash is the best thing ever. I'm only trying to emphasise the particular strengths of this language in comparison to others, and offering prospective users an illustration of how it can help them. These are just some of the basics – as you become more involved in Flash, you'll probably discover other neat tools which your previous language may not have offered or implemented in a desirable manner.

If you are comfortable with you current tools, that's just dandy – keep at it, and make great games. If however, you are looking for something more – maybe your current tools aren't up to scratch or you're not getting precisely the results you desire – then have a look at Flash. It has a lot of great features, it's relatively simple for a veteran to learn and is a very nice language for a beginner who wants to learn how object orientation works, and what its potential may be. If you want to learn more, check out Dev.Mag Issue 28 for a tutorial on how to get an absolutely free head-start in developing with Flash.

DEVELOPMENT

Once upon a time...

Quinton "Q-Man" Bronkhorst

Game Development. The words have a way of sending a flurry of mathematical equations and complex code lines swirling through one's head; after all, it's exactly that which drives a game forward, isn't it? – having your logical pathways set out so that when objects interact with one another they do what is expected. We get so hammered down on programming and gameplay that it's easy to forget, particularly at the introductory level of game development, that there's much more to a game than simply getting the code to compile without any errors. This 6-part guide is here to explain and give you an idea of one of the more abstract areas of game-development: Narrative.

What is Narrative?

Now, you may be wondering why you need to care about something like Narrative. Think about the last five or so game you played. Assuming you breach the sports genre, I will hazard a guess that the majority of those games had a story on some level. Narrative has always been a massive part of the gaming industry, and in recent times, has become even more complex in structure. So naturally, as a game developer, it should be of some interest to you.

So what is it?

To put things bluntly, Narrative boils down to telling a story. Simple enough – but that's an explanation that's only suited for shouting to a stranger as you pass each other by in a crowded room. In the greater scale of things, narrative involves a deeper set of qualities: characterization; dialogue; sequence of events; as well as the world in which all of these things take place. The manner in which all of these qualities interact with each other, forms a narrative.

Narrative is highly flexible, and can stretch as far as your imagination (so if you're a fan of Britney Spears, you may be in trouble here), no idea is off-limits. But there are a few things to take into consideration, and many things to avoid, but these will be covered as we get to them in the coming months. Just remember that you're making a game, and that whatever story you imagine has to be adapted to suit that particular medium.

The Narrative scale

Okay, so now we know what narrative is, how are we going to start implementing it? Well, it's a good idea to first determine what level of narrative structure you're going to have before anything else. This will stop you from going too deep, or not going deep enough, in terms of the world you're going to create.

The first mistake you will probably make is to think that genre determines narrative style. While not completely inaccurate, to let narrative be determined by genre means you will never be able to breach or expand that genre; which means that you're essentially restricting yourself. So instead, we're going to look at narrative on a number of levels, rather than by the genre of the game.

You will find that each level of narrative has an exception to the typical genre – which further presses the point that narrative isn't genre-specific, and that genres, particularly today, are branching out further and further across narrative levels.

*****Remember: DON'T limit yourself, not even by the level explanations that follow – these are simply here to make things easier to understand; the lines are extremely blurred in practice. The levels indicate the level of depth of the story, and not its effectiveness.**

Being a good sport

1

This is the 'no story' level of narrative structure. There is no cohesive tale to be told, no characters to develop and no dialogue to be had. The world in which it takes place has straightforward rules which basically tell the player how things are done, and then leaves them to it. The focus here is gameplay and often nothing else.

Typical Genre Sport, Racing, Old school
Games Any Fifa Game, Gran Turismo, Tetris
Exception Need for Speed Underground

The princess is in another castle

2

This level of narrative structure gives the player a final goal (be it per level, or the game as a whole). The world is basic, offers no elaborate explanation and the characters are recognizable and unique but share no interactions and have extremely limited dialogue. The focus is, again, on gameplay, but players are in a more diverse world.

Typical Genre Platformers
Games Mario Brothers, Sonic the Hedgehog, Crash Bandicoot
Exception Jak and Daxter

The middle-man

3

As the name implies, this is the median level of narrative, where an equal focus is placed on the story and gameplay. The world is recognizable and unique; with characters that are either previously developed or get developed to a degree (but rarely stray too far from their set characteristics such as "badass action hero"). There are dialogue sequences that push the story forward, but interactions are often superficial.

Typical Genre Film tie-ins, Action, First-Person Shooters-
Games Most FPS and Action games out there
Exception Bioshock

An epic journey

4

This level of narrative tips the scale in favour of story. While Gameplay is still a major focus (as it is a game, after all), the story becomes paramount in getting and keeping the player interested. The world is unique, diverse, developed and explained. The characters are deep and develop throughout the game, transforming as the story plays out. The characters interact through dialogue, often learning more about themselves and the world, or making decisions that affect the outcome of the story.

Typical Genre RPG, Action/Adventure
Games Final Fantasy, Resident Evil, Silent Hill, Grand Theft Auto
Exception Fallout, Fable, Mass Effect

The world is yours

5

The deepest level of narration is one that provides you with the option of how things in the story will play out. As it were, this level of Narration has many levels itself, but it ultimately it boils down to the player shaping the story to whatever degree. The characters are whoever the player makes them, and the story goes down a path the players choose. Dialogue happens the way players determine, and the outcome of the story depends entirely on those aforementioned choices.

Typical Genre RPG
Games Elder Scrolls, Fallout, Fable, Mass Effect
Exception The Sims

Filling the void

6

Indeed, you get a level of narration that's completely non-existent, but has the potential to be the deepest level of them all. Basically, the developers have provided everything the player needs to shape the character, world, interactions and story. It is essentially a clean slate that has the potential to be whatever the player can imagine.

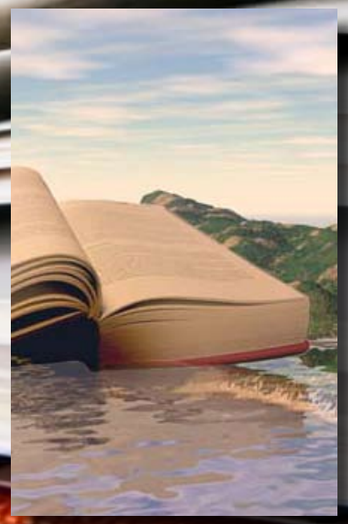
Typical Genre Simulation
Exception Real Life

Finding Direction

Now that you know what you're getting yourself into, we can finally start tackling narrative in more detail, fleshing out each quality and building it up. For the sake of this guide, we're going to be creating a 'middle-man' (level 3) narrative, which over the course of the next few months, we'll develop into a potential indie game-ready story.

In the next Issue: **Once upon a time** – We'll have a look at the story as a whole; from beginning, middle to the end.

The Narrative Series



Introduction *Telling the story* *The World* *Characters* *Dialogue* *Final touches*

IN CASE OF EMERGENCY
BREAK THE MOULD

IGF FINALIST ROUNDUP

Claudio "Chippit" de Sa & Rodain "Nandrew" Joubert

HOW IT WORKS

For your pleasure, here we have a bunch of icons which carry all the information you will need to know about the IGF Finalists

NOMINATIONS

Here are the icons which depict which award the game was nominated for


Excellence in Visial Art


Technical Excellence


Excellence in Audio


Innovation Award


Excellence in Design


Seumas McNally Grand Prize

STATUS

Here are the icons which show at what stage of production the game is in

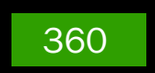

Free to play


Demo Available


In Production

PLATFORM

Here are the icons which show what platform the game operates on


Xbox 360


Playstation 3

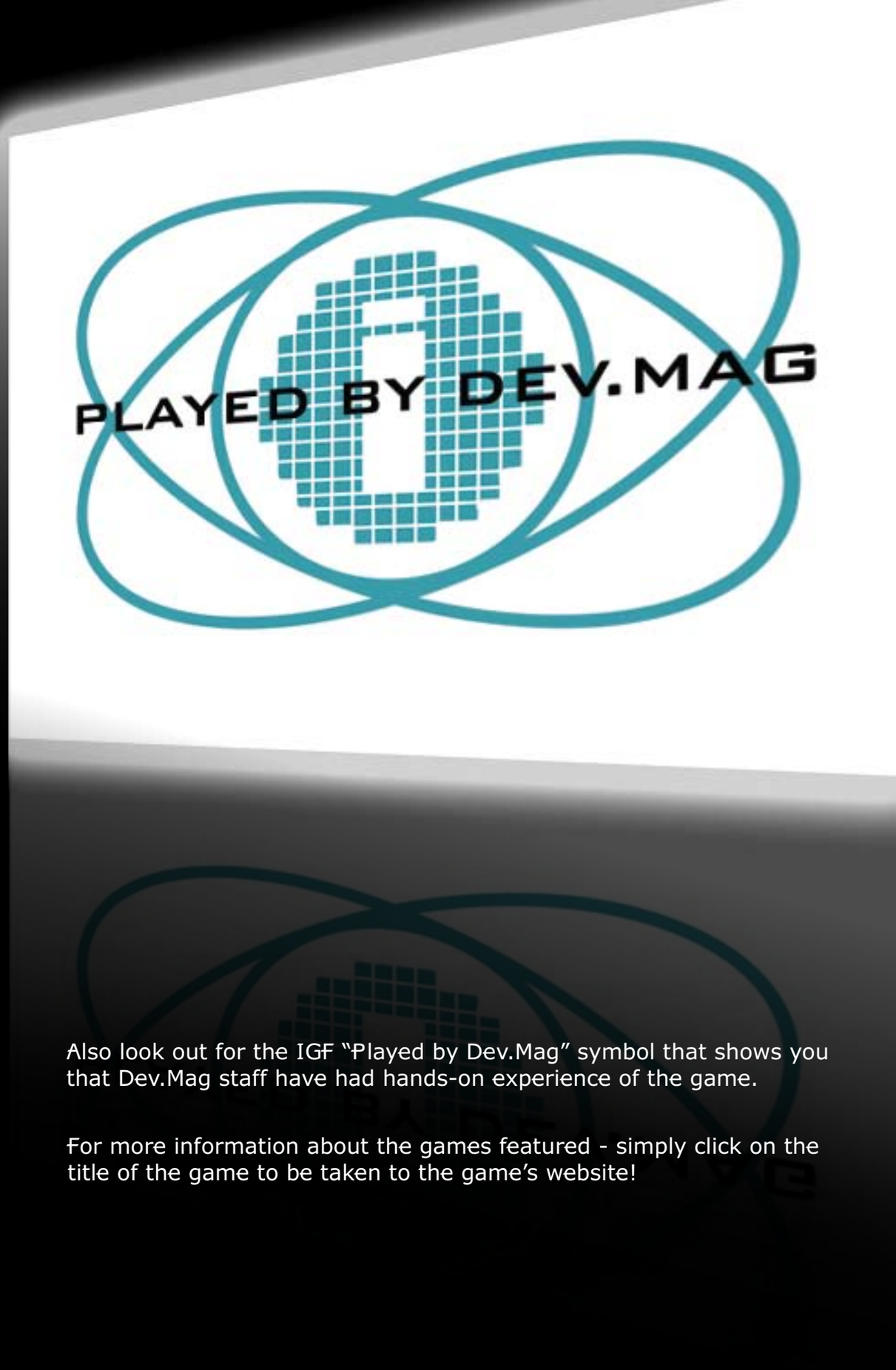

Windows


Apple Mac


Linux

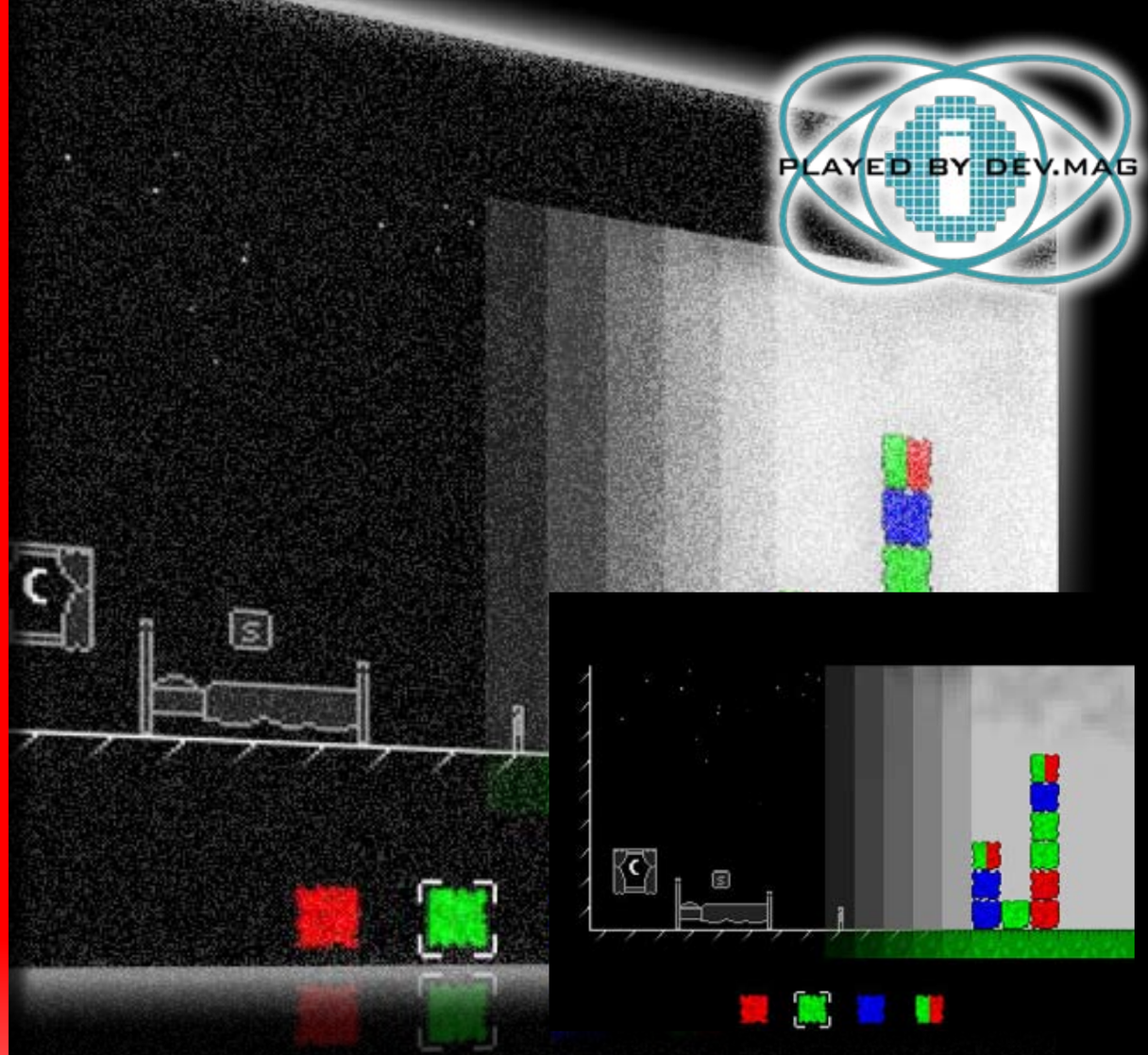

Flash


WiiWare



Also look out for the IGF "Played by Dev.Mag" symbol that shows you that Dev.Mag staff have had hands-on experience of the game.

For more information about the games featured - simply click on the title of the game to be taken to the game's website!



Between

Developer: Jason Rohrer

Between is a particularly strange game, created by someone known for his particularly strange games. Like one of Jason Rohrer's other creations, Passage, Between is an odd, brooding game, simultaneously a puzzle and an experiment in isolation – which is strange considering this is strictly a 2-player game. It may take a little bit of time to work it out, but that's the point.

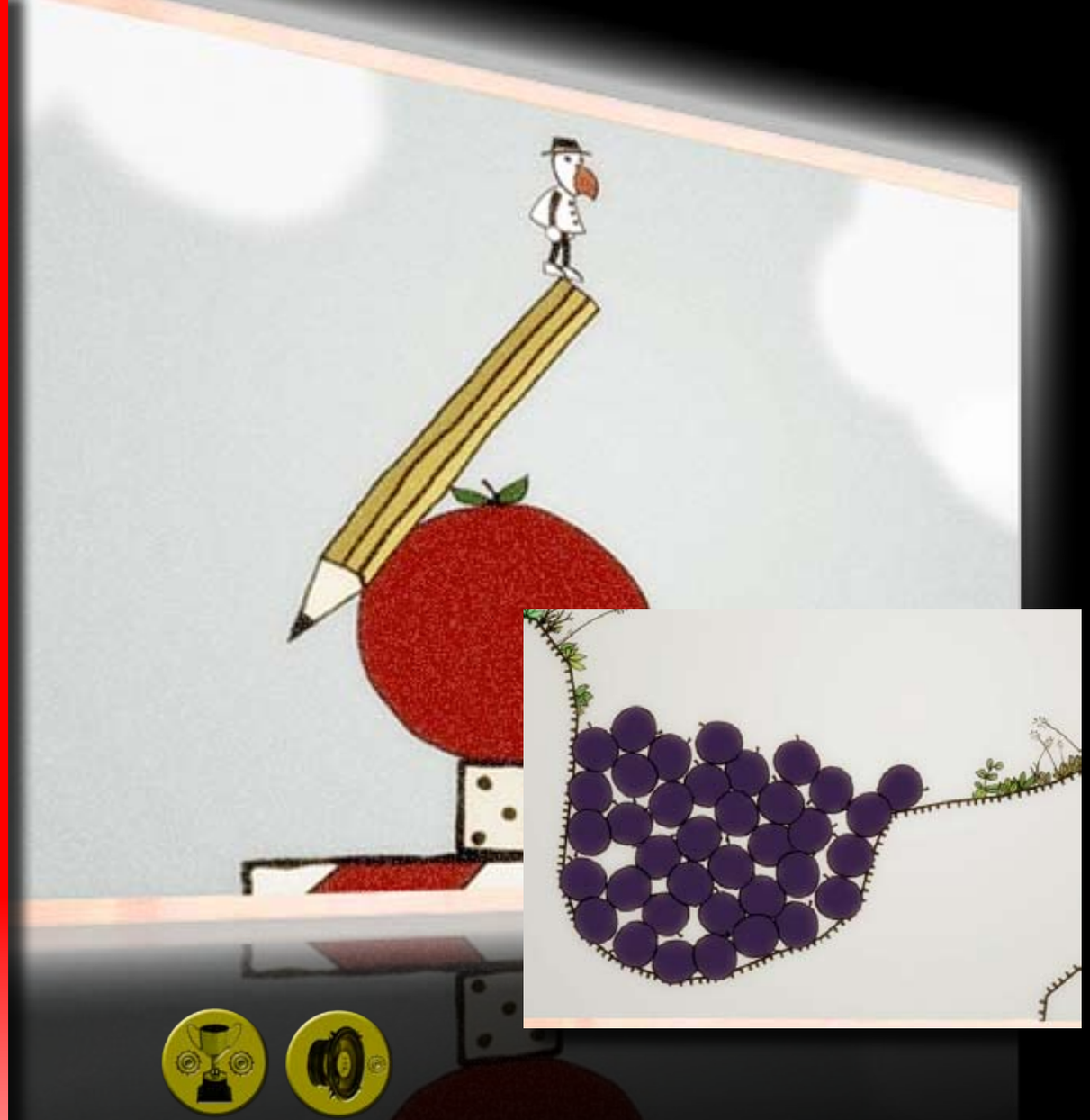
~~360~~ ~~PS3~~ WIN MAC LIN ~~SWF~~

Machinarium

Developer: Amanita Design

Machinarium is a full-scale point and click adventure game being developed by the creators of the famed Samorost web game series. Not much is known about the game yet – aside from information gleaned via interviews and screenshots – but the story of robo-drama combined with the charming hand-drawn artwork typical of these Czech developers has secured it a spot amongst the IGF finalists. The Samorost games have featured in previous IGF competitions and have left an impact on players due to their unique style and great visuals, so fans have high hopes for Machinarium – due for release later this year.

~~360~~ ~~PS3~~ ~~WIN~~ ~~MAC~~ ~~LIN~~ SWF



Blueberry Garden

Developer: Erik Svedang

Blueberry Garden is a secretive game, an experiment about the player's influence on a self-contained eco-system. While little is known about how the game will be played, it has already walked away with an award for innovation from the 2008 Swedish Game Awards, and this arty submission certainly appears to be a feast for the audio and visual senses as well.

360 PS3 **WIN** MAC LIN SWF

The Maw

Developer: Twisted Pixel

A cute and interesting little offering, The Maw is an action/adventure 3D title centred around the adventures of an alien named Frank and a blob called Maw – also known as The Deadliest Organism In The Universe. The Maw uses the basic "you are what you eat" philosophy, granting the player abilities based on the types of creatures consumed. Some have cited the game as being too short and easy, but its technical presentation is beyond reproach. The Maw is available on the Xbox Live Arcade.

360 PS3 WIN MAC LIN SWF

Brainpipe

Developer: Digital Eel

With its vivid yet curiously surreal visual style and use of audio, Brainpipe is like riding a hypnotic rollercoaster – except you also need to steer it. Its simple controls and gradually increasing pace – peaking somewhere just beyond frenetic, assuming you can make it that far – instantly classify the game as one that attempts to be simultaneously casual yet challenging.

360 PS3 **WIN** MAC LIN SWF

Mightier

Developer: Ratloop

Mightier is an innovative title which encourages players to think out of the box – or at least, out of the computer screen. The crowd-winning feature of this title is its use of scanners and printers to present players with pen and paper puzzles that actually affect what occurs in the game. Instead of solving challenges within the game, a map of each level can be physically printed out and changed by drawing on the physical sheet of paper, which is then fed back to the game with a scanner. The game features a fairly high level of customisation (you can even draw your own versions of the avatar and equipment) and for those who aren't within reach of a printer or any spare paper, less adventurous ways to solve the puzzles exist through the mouse draw options. This game is free and pretty fun to play – go check it out!

360 PS3 **WIN** MAC LIN SWF

CarneyVale Showtime

Developer: Singapore-MIT Gambit Game Lab

If CarneyVale Showtime sounds familiar, that's because it is. Regular readers will recognise it as being the winner of Microsoft's DreamBuildPlay competition, and it's now pitting itself against games on platforms other than the Xbox360 in order to clinch grand prize here too. If you haven't tried it yet, don't forget to head over to the Community Games section of the Xbox Live Marketplace and give it a try.

360 PS3 WIN MAC LIN SWF

Musaic Box

Developer: KranX Productions

Musaic Box is an aurally-based puzzle game which has you solving geometric puzzles by listening to musical melodies and gathering the puzzle pieces in such a way that you construct the full audio sample. Those who struggle with sound, however, can still attempt to solve the puzzle visually. A free 1-hour trial enables you to try before you buy – available from Big Fish Games and similar casual portals.

360 PS3 WIN MAC LIN SWF

Cletus Clay

Developer: TunaSnax

Clay animation, something familiar to movie buffs, is a term rarely used in the gaming industry. Cletus Clay hopes to change that with its unique graphical style: everything in the game is modelled out of real, physical clay. Clever use of stop-motion animation results in very smooth looking movements, an excellent complement to the comical nature of the game's hillbilly main character.

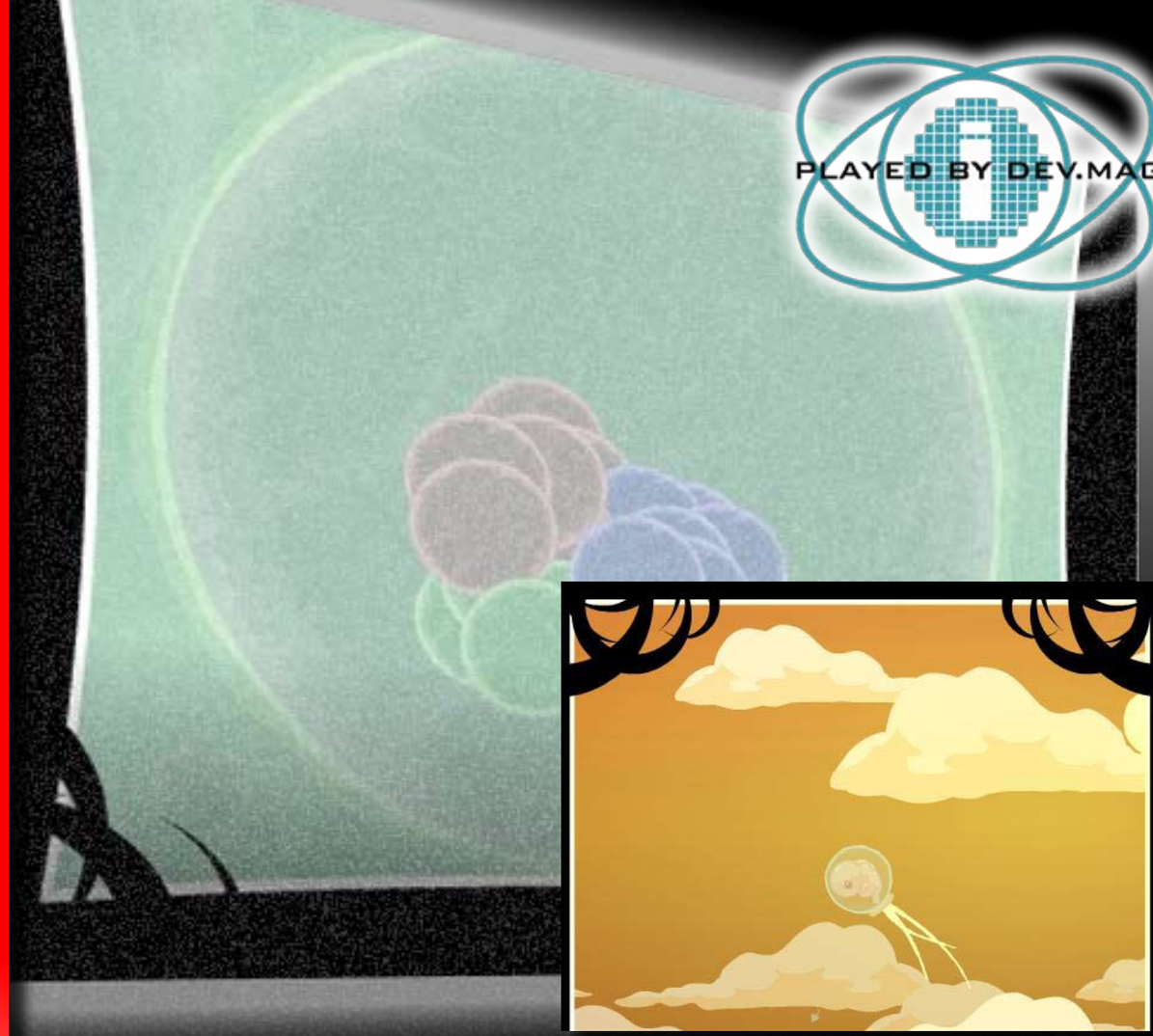
360 PS3 WIN MAC LIN SWF

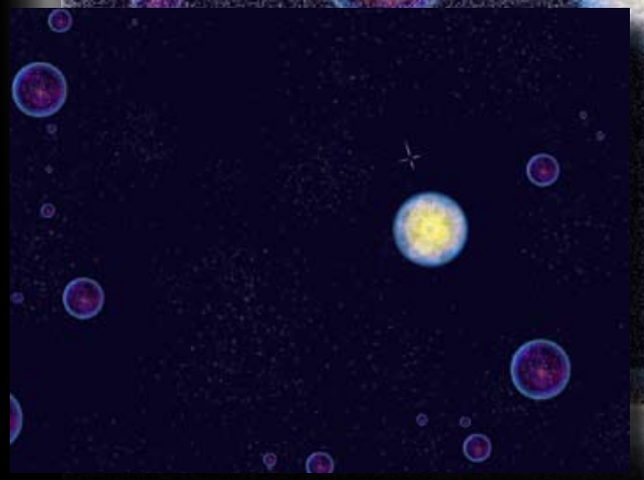
Night Game

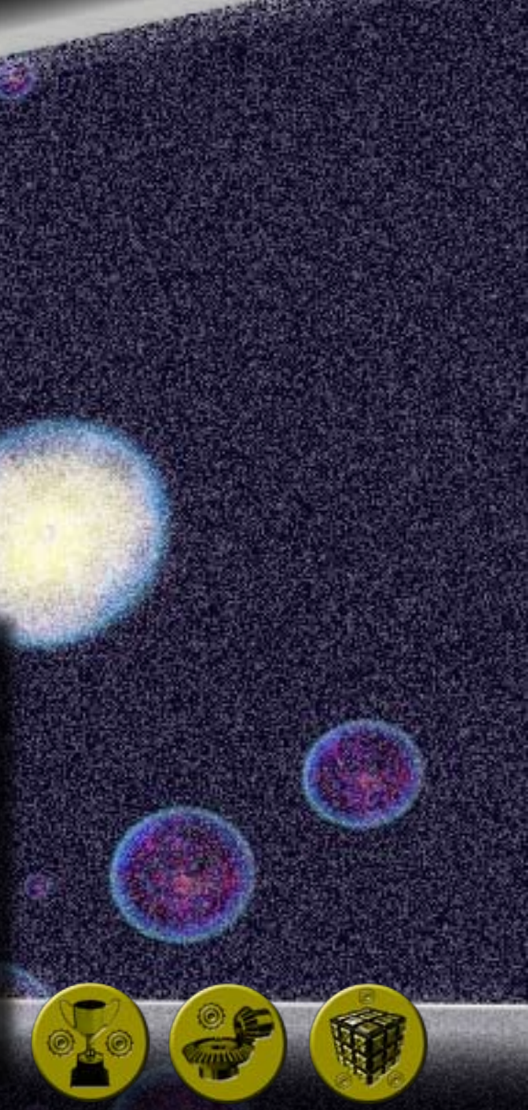
Developer: Nicalis

Night Game is a new offering from game development veteran Nicklas "Niflas" Nygren, the man responsible for platforming gems such as Knytt and Within a Deep Forest. Night Game is a physics-based puzzle game which focuses on navigating a ball through ambient and engaging worlds. The game is slated to be a relaxing – yet challenging – experience that will appeal to a broad base of gamers.

360 PS3 WIN MAC LIN Wii







Coil

Developer: From The Depths

A curious and unique art-game, Coil is a subtly dark attempt at something that is almost more of an experiment than a game; an experiment in the immersion that mystery and the forgoing of traditional instructions can create. The game treads into territory rarely ventured into, and the dual-story and thematic setting make for an interesting experience.

360 PS3 WIN MAC LIN SWF

Osmos

Developer: Hemisphere Games

Osmos has the player taking the role of an organism that needs to grow by absorbing similar organisms (or motes) whenever you collide with them. It sounds simple enough at first, but when you consider that your only means of propulsion is by losing mass (and that larger motes will absorb you on contact) you quickly find yourself trying to maintain that delicate balance between size and mobility to achieve superiority. Although only an alpha version is available, players get to preview the basic game dynamic and a few interesting mote types which exhibit cute AI and some intriguing behaviours.

360 PS3 WIN MAC LIN SWF

Cortex Command

Developer: Data Realms

A familiar game to many followers of the indie scene, Cortex Command has been around for a very, very long time. Being under on-and-off development for up to seven years, Cortex Command has finally reached a state worthy of the IGF competition. Will its excellent physics engine and pixel-perfect collision detection be enough to net it the award for technical excellence?

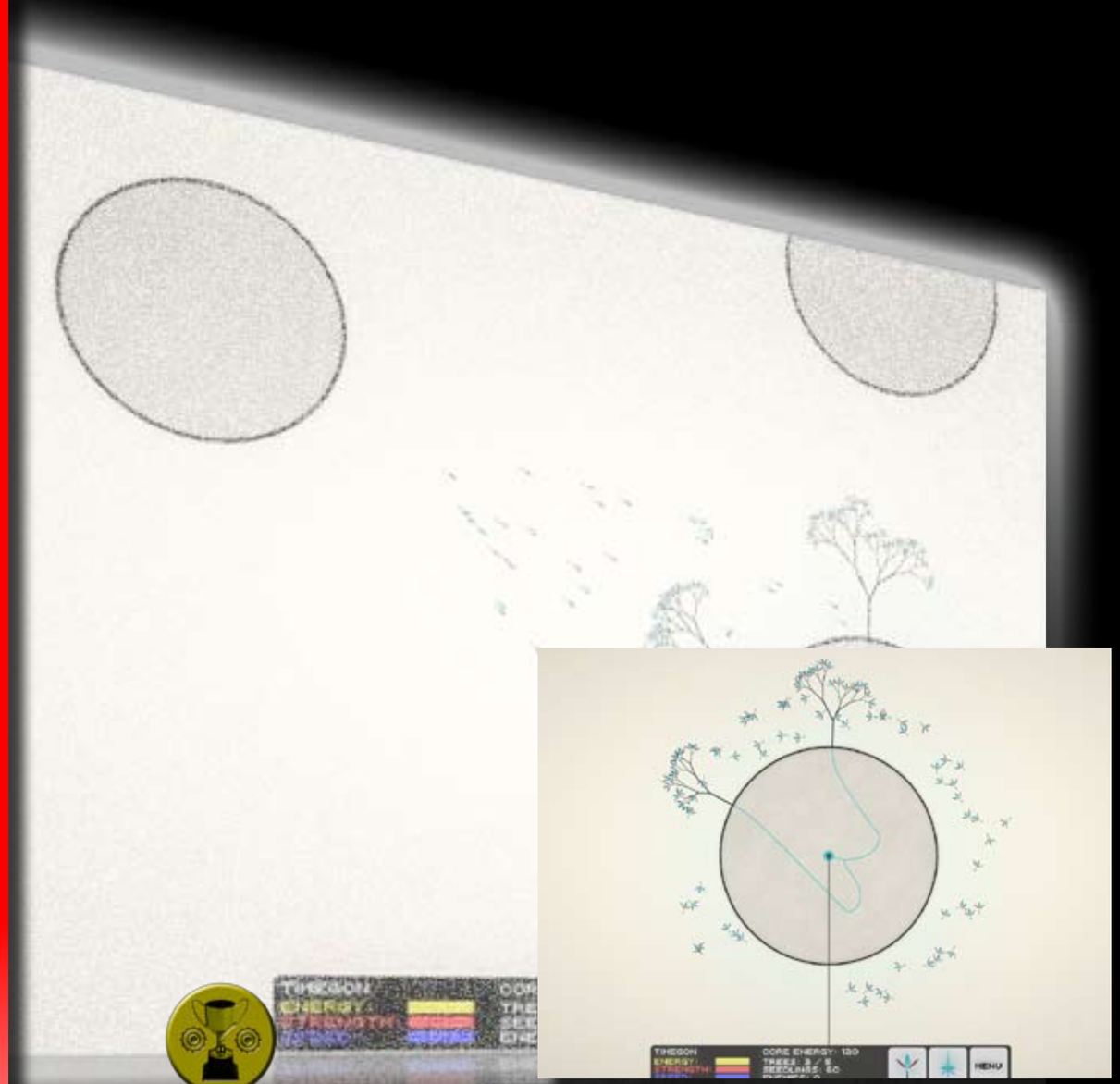
360 PS3 **WIN** MAC LIN SWF

PixelJunk Eden

Developer: Q-Games Ltd.

Several titles have already been churned out under the PixelJunk moniker, and now Eden looks set to outshine them all with a nomination for three separate awards under the IGF judgement criteria. PixelJunk Eden has you hopping around simulated alien plantlife, with up to three players controlling little ... er ... things which can collect helpful objects and get more plants to grow. It looks charming, and hopefully has a strong co-op element. Available for purchase through the PlayStation Network.

360 **PS3** WIN MAC LIN SWF



Dyson

Developer: Rudolf Kremers and Alex May

While we were unfortunately unable to get this interesting-looking RTS game to run, [its persistent crashing on launch driving poor ol' me to murder. Murder zombies, that is, in Left 4 Dead – Ed] it looks to be one to watch out for. The game puts the player in control of self-replicating machines manifested as trees and seedlings in order to try and conquer and colonise an asteroid belt – something often already occupied by rivals.

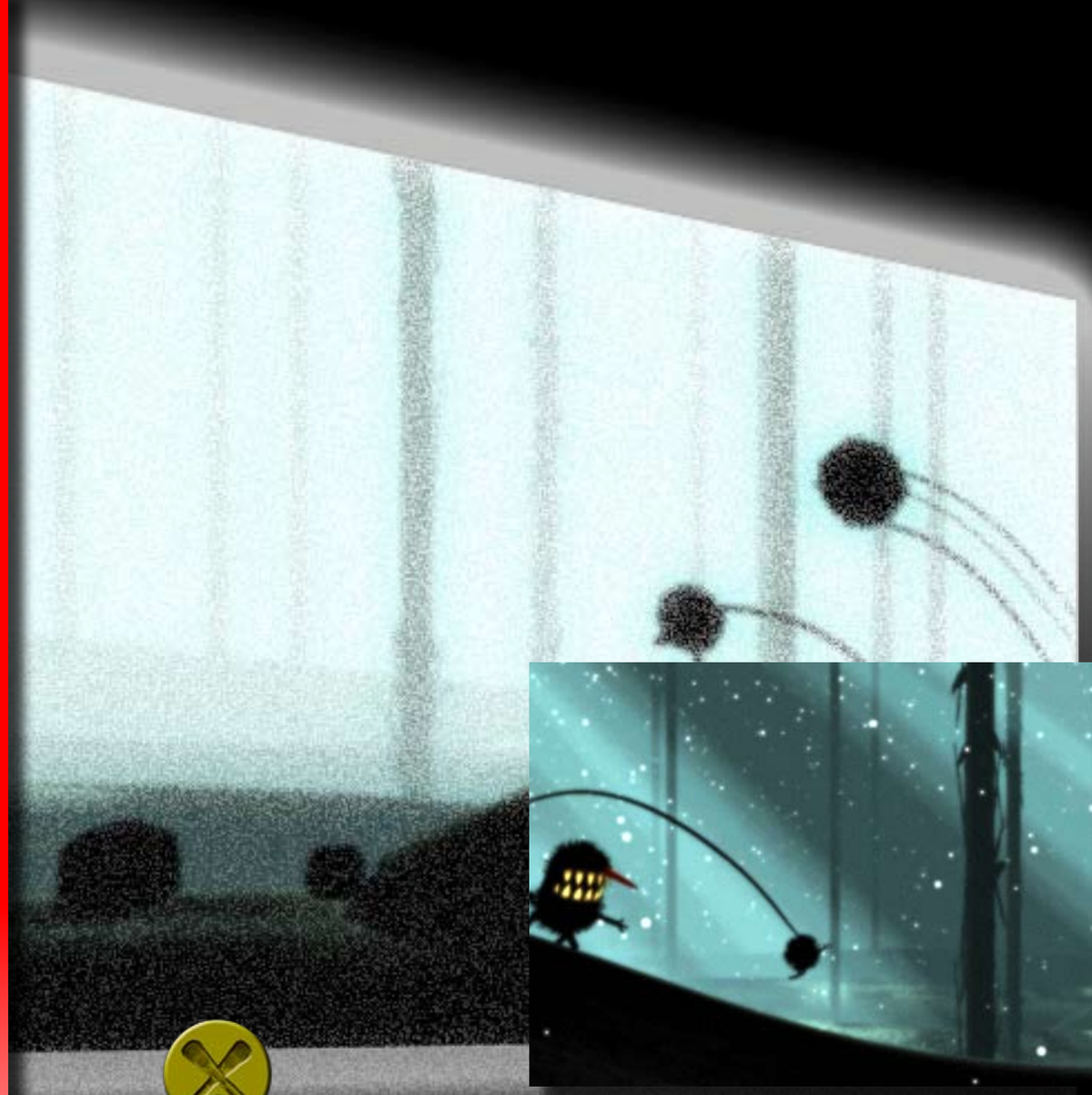
360 PS3 **WIN** MAC **LIN** SWF

Retro/Grade

Developer: 24 Caret Games

Retro/Grade may at first appear to be a simple shooter, but it soon turns out to be a cunningly-disguised rhythm game that surprises players by playing through an entire space battle in reverse. Players need to dodge lasers returning to enemies and "catch" their own gunfire to preserve the integrity of the temporal anomaly that's forcing time to go backwards, assisted by audio cues and the regularity of game events.

360 PS3 **WIN** MAC LIN SWF



FEIST

Developer: Filthy Grip

FEIST sets the player on a platforming journey through stunning locales and tricky obstacles. Featuring an artificial intelligence system that claims complete autonomy in enemy behaviour as well as an evolving and changing world, FEIST promises to be different and unique every time.

360 PS3 WIN MAC LIN SWF

Snapshot

Developer: Kyle Pulver and Peter Jones

The ability to take snapshots of a level and use them later to actively affect the environment may sound quite ambitious – even crazy – but Snapshot promises to pull that off. In this game, the player controls a character called Pic who can capture scenery, objects and even enemies using his camera. The game is still in its early stages of development, but it looks like an intriguing offer.

360 PS3 WIN MAC LIN SWF

The Graveyard

Developer: Tale of Tales

Tale of Tales, a studio with a penchant for preceding all their game names with articles, is another familiar name in this year's IGF. Familiar for two reasons, in fact: one being their unusual MMO, The Endless Forest, and the other their IGF entry from last year, The Path. The Graveyard appears to follow a similar trend to that first presented in The Path, once again being an experiment in interactive storytelling.

~~360~~ ~~PS3~~ WIN MAC ~~LIN~~ ~~SWF~~

You Have to Burn the Rope

Developer: Kian Bashiri

Every once in a while, some bored (or creative) game developer decides to create a tongue-in-cheek title to either make a point or simply poke fun at gaming as a whole. You Have To Burn The Rope is described as "a game about interactivity and false choices" and (more importantly) "a joke". The name pretty much gives you your objective, and the cuteness of the concept is sure to entertain players – even if the game itself is ultimately very short. An interesting and rather unexpected addition to the IGF lineup – you'd have to play it to understand.

~~360~~ ~~PS3~~ ~~WIN~~ ~~MAC~~ ~~LIN~~ SWF

PLAYED BY DEV.MAG

Incredibots

Developer:Grubby Games

Incredibots is another of those browser-based physics construction sandbox games. What separates Incredibots from other similar games is its completeness. The game sports an initially complex user interface that is powerful enough (with familiarity) to allow for some really complex contrivances and visually impressive machines. Its simple tutorial and easy-to-use sharing system makes this a fun way to pass the time.

360 PS3 WIN MAC LIN **SWF**

Zeno Clash

Developer: ACE Team Software

Zeno Clash describes itself as having a "beautiful, disturbing, unprecedented" art style wrapped around a raw close-combat system in a punk fantasy world. Judging by the screenshots and gameplay videos (as well as IGF's seal of approval), it looks like ACE Team may be able to back up these strong words with a proud indie offering featuring brutal melee combat in an interesting and colourful environment.

360 PS3 **WIN** MAC LIN SWF

ONLINE

DEV.MAG

CREATE • DEVELOP • EXPERIENCE

www.devmag.org.za