SOUTH AFRICA'S PREMIER GAME DEVELOPMENT MAGAZINE

JANUARY 2007

DEV.MAG

CREATE ● DEVELOP ● EXPERIENCE

REGULARS

SPOTLIGHT

FEATURE

REVIEW

DESIGN

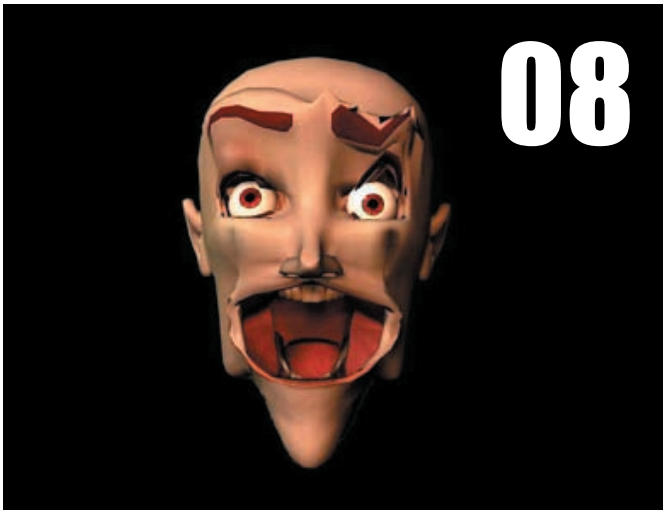
PROJECTS

TECH

HISTORY

TAIL PIECE

COMIC



Righto. Happy New Year! A bit late, perhaps, but considering that this is the first edition of the year, I think we can be forgiven for the delay. So, did you miss us last month? Or were the Christmas prezzies and New Year booze-ups a little too distracting for that? I know that most of the Dev.Mag staff took full advantage of the holidays this year – it seems that even slaves to the game development system enjoy breaks sometimes. Heh, who would have guessed?

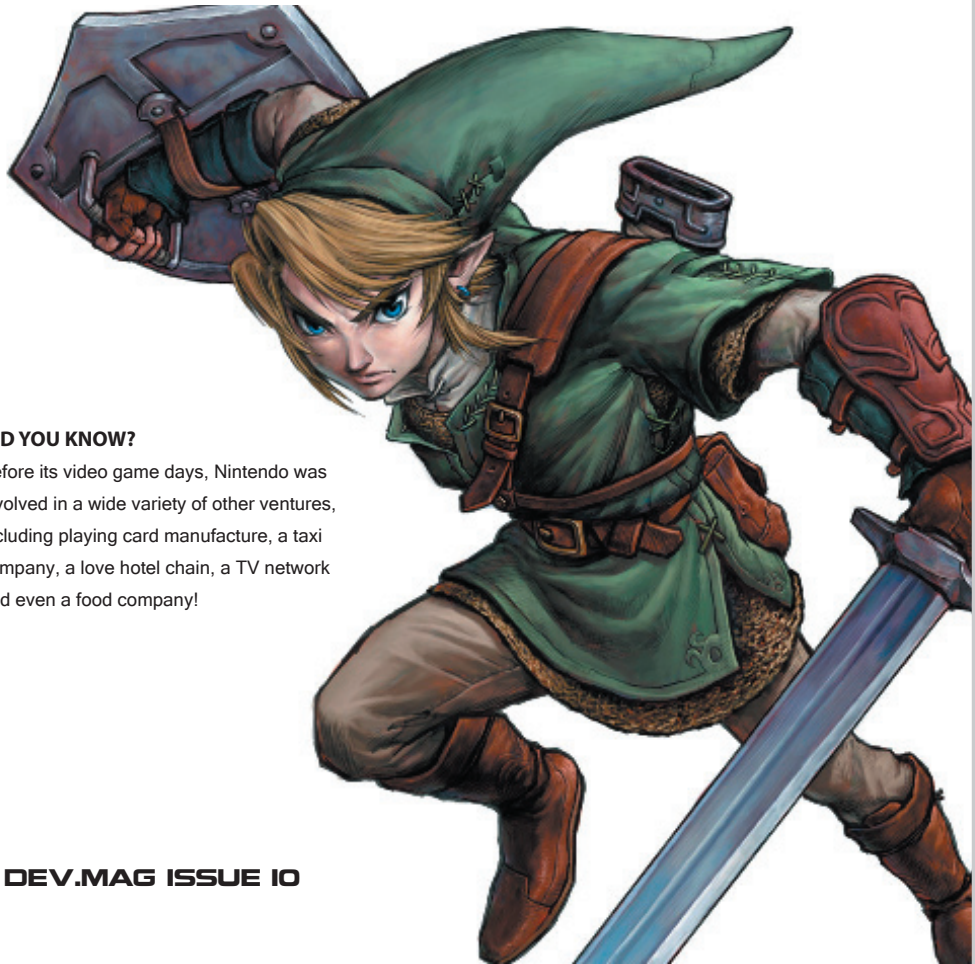
Regardless, everybody was able to spring back this month to get the content in, and it seems that our designer has had a little fun with the first mag of the year as well. Hope you like it. It's really awesome to see where we've managed to get ourselves after only a year of existence, and it's always exciting to know that we've still got a lot of growing to do. Sometimes, I still go back and look at the humble offering that was our first edition, and it encourages me to see what a bunch of dedicated people can achieve with a little time and effort.

Speaking of which, it seems that the Hotlabs are finally moving at full speed in Durban, what with the new influx of ITI students this year and the fact that Game Maker has been adopted as the standard learning tool for the labs. There's even plans underway to have the Hotlabs branch out into other disciplines, including sound engineering and graphics design – open to the public and absolutely free! Being a KZN local myself, I'm quite excited about the developments being made, and joyfully poke a challenge at the idea that Durbs is nothing more than the "baby brother" of the major SA cities. Joburg, Cape Town – eat yer hearts out!

You can check up on the holiday Hotlabs in this month's feature – be sure to e-mail us with requests for labs in your area if you like what you read, and we'll see if we can get something going for interested parties.

Enjoy the mag, and ciao for now.

Editor
Rodain " Nandrew " Joubert



DID YOU KNOW?
Before its video game days, Nintendo was involved in a wide variety of other ventures, including playing card manufacture, a taxi company, a love hotel chain, a TV network and even a food company!

DEVMAG
CREATE DEVELOP EXPERIENCE

EDITOR
Rodain " Nandrew " Joubert

DEPUTY EDITOR
Claudio " Chippit " de Sa

SUB EDITOR
Tarryn "Azimuth" van der Byl

DESIGNER
Brandon " Cyberninja " Rajkumar

MARKETING
Bernard " Mushi Mushi " Boshoff
Andre " Fengol " Odendaal

CARTOONIST
Paul " Higushi " Myburgh

WRITERS
Simon " Tr00jg " de la Rouviere
Ricky " Insomniac " Abell
William " Cairnswm" Cairns
Bernard " Mushi Mushi " Boshoff
Danny " Dislekcia " Day
Andre " Fengol " Odendaal
Heinrich " Himmler " Rall
Matt " Flint " Benic
Luke " Coolhand " Lamothe

WEBSITE DESIGNER
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.gamedotdev.co.za**

Is Vista a boo-boo for indie developers?

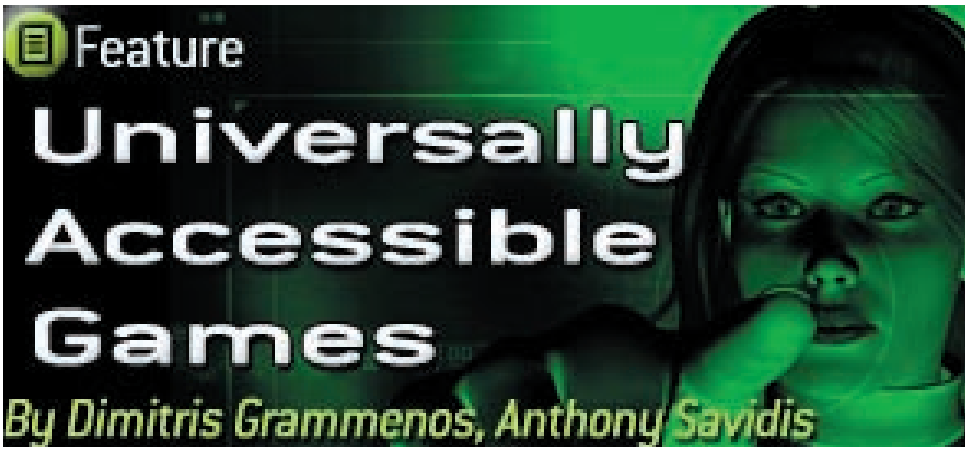
http://www.gamasutra.com/php-bin/news_index.php?story=12314

Original designer of DirectX and founder of WildTangent, Alex St. John, has expressed concerns about Vista's new security features and the risk they pose for the continued development and distribution of independently developed games. "We have found many of the security changes planned for Vista alarming and likely to present sweeping challenges for PC gaming, especially for online distributed games," he claims. Apparently, not only does game compatibility become a much graver issue, but downloadable game distribution – a cornerstone of marketable indie titles – is also going to be severly hampered. The full letter is available on Gamasutra.

Making a playable game

http://www.gamasutra.com/php-bin/news_index.php?story=12013

Just how user-friendly is your game? Playability and a good user interface are important to most self-respecting developers, but this isn't often done with the physically handicapped in mind. Gamasutra recently released a feature about "Universally Accessible Games" -- titles that could, ideally, be played by anyone regardless of physical handicap. Before now, this field of games hasn't been explored too extensively and there is a lot of room for ideas and completed designs. More details in the article.



New XNA video

http://blogs.msdn.com/xna/default.aspx

The XNA team has just announced their latest video release for XNA development. The video, called "Getting started with the XNA Creator Club", is an eleven-minute demonstration on the hows and whats of the Creator Club, dealing with issues such as connecting to the Creator Club, purchasing an account and downloading homebrewed work to the console. The video is available from the Microsoft Download Center in both standard and high resolution.

Java 4K Programming Contest

http://javaunlimited.net/contests/java4k.php

Think big, right? Well, that's not going to work for this competition. The Java 4K Programming Contest, now in its fifth year, is described as "the ultimate byte-squeezing Java challenge". Entrants are required to create a game no larger than 4096 bytes, using all the cunning and resources at their disposal to create something memorable in something minimal. The deadline for this competition is March 1st, and for reference there are archived games from the past two years available on the site.

On Game Design

http://www.gamecareerguide.com/features/327/on_game_design_a_history_of_video_.php

"On Game Design", a new series from www.gamecareerguide.com, looks to be a very promising read for those in the market. The first article, "A history of video game design" takes a look at the rich heritage of game design, detailing events, products and trends that formed the building blocks of modern techniques. The series consists of six parts and includes discussions on the application of good and bad ideas, technical design and interviews with actual game designers.

A games academy?

http://www.gamasutra.com/php-bin/news_index.php?story=12292

More proof that game development is going the way of the film making industry. UK Minister for Creative Industries and Tourism Shaun Woodward has called for the sponsorship of an academy to support students of the video games industry, "similar in function to the successful London Film School." He's expressed an interest in fostering the talent and skills of the industry, and believes that such an academy is the best way to progress. Let's wait and see what happens.

Telling a story ...

http://www.gdcradio.net/2007/01/a_practical_guide_to_the_heros.html

Most of the staff over at Dev.Mag are huge fans of GDCRadio, and we think that you should be, too. Yet another reason for the crew's devotion has just surfaced in the form of "A practical guide to the Hero's Journey", a podcast by Bob Bates which explores the idea of creating a solid and convincing game story that goes beyond simply solving puzzles to continue or running around and killing everything. People interested in this topic can also look up Tim Schafer's "Adventures in Character Design" and Ernest Adams' "Interactive Narratives Revisited: Ten Years of Research"

Game Addiction

http://www.gamasutra.com/php-bin/news_index.php?story=12293

Most of us have heard from time to time about game addiction - a lot of us have probably joked about it or even labelled ourselves as "gaming addicts". But what is the true nature of this beast? Has it been studied enough? Does the success of the video games market hinge on that "addictive" element? Gamasutra's feature on the matter tries to answer questions about this phenomenon, and questions whether or not we truly understand it yet.

Feb 1st - June 3rd 2007

Welcome to the 2007 Pascal Game Development Annual “Multiplexity” Competition! A game development and design competition for Pascal developers. This is our 3rd competition and we hope that it will be bigger than the last two. This year’s theme is called “Multiplexity”. We believe this reflects the idea that you must make a game with multiple genres and which will therefore cause multiple complexity. This year Pascal Game Development have more sponsors on board, bigger prizes than ever and are hoping for many more entrants than ever before. All entries must be developed in a Pascal based language such as Delphi, FreePascal, Chrome or Midlet Pascal. All entries must have a windows version but will score extra points if they run on other platforms.

http://www.pascalgamedevelopment.com/competitions.php?p=details&c=3

The William Cairns Interview

Senior Solutions Architect

Tell us more about yourself. What do you do for a living?

I work for one of the largest IT companies in South Africa. My job title is Senior Solutions Architect. Basically my job is designing new computer systems and ensuring that the developers get the work done to reach the goal I have in mind. I unfortunately get very little opportunity to actually write code anymore, except when it comes to optimization at application or database level. I am married with three kids - who love my games. I'm also going to do the Comrades marathon this year - so with all the training I need to do, game development will be taking a bit of a back seat for the next six months or so.

Where did your interest in game development start?

The first programs I can remember writing were games. From a simple 6-room adventure game to making hang-gliders fly through caves, I have always made computer games. When I got a copy of Star Craft I knew that I wanted to be able to write a game that was similar. Since then I have slowly improved my skills to the point where I believe I could actually write my dream game, given enough time.

You have won esteemed places in international competitions. Tell us more about that.

Actually, I haven't. :) I have entered many competitions but only twice been in the prizes. The first prize I received was for Dragon Flight (A vertical shooter) and the second was for Fantasy Land in the NAG Game Dev contest last year.

What is your most popular/best game? Tell us more about it.

Run-A-War game (4th in the Pascal Game development Contest 2004) is probably my best game and also the one I'm most proud of. I enjoy entering contests as it gives extra incentive to finish the game, and also a deadline that needs to be met. It's difficult to say "tomorrow" when you are working toward a deadline that can't change. Run-A-War got posted on a few "Free Game" sites last year and for three months I was getting over 10000 downloads per month from it.

Mr William " Cairnswm " Cairns, a South African Pascal Games Developer

You recently got a development deal for $1000. How did it all happen?

A company called Online Bandit (www.online-bandit.com) posted on the Pascal Game Development forum asking for Pascal/Delphi developers to write some games for their site. Their games work through a custom portal delivering multiplayer games to their members. Members are then also encouraged to purchase items from the site, thus funding the company. I responded to the post as their site looked a lot more professional and active than most other people that have posted similar requests ("I've designed this uber game - I just need 5 developers to code it for free" type posts). After creating a demo game for them they contracted me to write Spades for their portal. I expected to take about 50hours to write the game, so a US$1000 paycheck for that sounded pretty good.

Needless to say it took a lot longer than I expected. In fact a few times I nearly pulled out completely. In the end I spent nearly 200 hours on getting the game working correctly. A lot of my frustrations were due to having to work within their (undocumented) framework, using their components and based on the contract having limited access to their internals. I have been asked to do another game for them.

I have asked for more access to their source as one of the conditions of me doing the new project. So far it looks like it will be happening.

What would you say is the most important advice you have for indie developers?

Finish your games! I have known so many people that have tried to get into game development by working on something too far beyond their skills, and when they can't get it finished they lose interest in the game and also game development. I know developers whose skills are way better than mine - and they have always wanted to write games, yet have never finished one, all because they try to do too much too soon. A couple of people have told me I sound very negative when I say things like this. I've been told that I say make things BELOW your skill level. I disagree with this sentiment in that you should be making games within your skill level. By tackling something that you can do you'll still learn something each time, and the next game you make will be slightly more complex. By trying to do something above your skill level you'll most likely fail and then lose interest.

What is your goal when you create a game?

My main goal is having a complete game finished at the end of it. Even before I try to make my games fun I try to ensure that the whole game is there - title screens, menu screen, game and game over screens must be present. Another important thing to me when making a game is to ensure that it is playable. Some of my games (like the Boo Game) are pretty pointless but were fun to make :)

How do you currently see the indie industry?

I believe the indie industry is coming under threat from the bigger game companies. The company I wrote the card game for has told me that something like 70% of their community members are women with kids. The big game companies have always targeted younger males (14-25) yet market information shows that the real money resides with mothers aged 30-45 - who do you think buy the games for their kids? So when the big game companies realize that these mothers also want to play games and start targeting their production power at this market, indies are going to find it a lot more difficult to make an income.

Anything else you would like to add?

I complete about 1 in 4 of the games I start developing. Sometimes I lose interest, get distracted or even find that the game is a lot less than I expected it to be. Often a game is canned long before I've written a single line of code. Game development for me is about 75% thinking about the game and 25% coding. I think people should take more time to think about the game ideas and implementation options before they start coding. Games that have been just coded are very difficult to extend while games that have been carefully thought about are typically easy to extend with new ideas. One of my problems about tutorials on the net is that they all only deliver one thing, and they are not designed to be extended to add new ideas to their content.

TR00JG

Quick Questions...3...2...1...Go!

PS3, Wii or Xbox360?

XBox360 - it's the only console I can make games for!

What famous figure would you most like to game with?

Wow - what a question..... Bill Gates - I'd love to see how he'd manage with some sort of management game :)

Winter or Summer?

Both! Winter - I like it cold

Summer - I like doing outdoor sports

Pasta or Pizza?

Pasta in South Africa, Pizza in Italy

The Simpsons or Family Guy?

Neither. But based on my complete dislike of Simpsons probably Family Guy :)

HAPPY HOTLABBING HOLIDAYS!

The summer heat isn't the only thing that's raging over at Durban. Since the start of the KZN Hotlab chapter last year, IT Intellect at Musgrave Centre has played host to monthly sojourns into the world of game development, spearheaded by professionals from all over the country and open to anybody who wants to learn more about this exciting pastime.

Over the holidays, the Durban Hotlabs have adopted Game Maker as a primary tool of education, with great success and some awesome results from learners. Read on to find out what we've been up to, as well as for more details about how to join in with the fun this year. No prior experience necessary!

A Hot Lab in progress. Hots Labs are held every month at IT Intellect in Musgrave.

The first Hotlab after the visit from our friends Danny and Miktar went according to plan, with the die-hard regulars and new fanatics all in attendance. The challenge had been set to our newly appointed guru of game development, Daniel Brewer (AKA Silent-Bob). To come up with a lesson in Game Development, and because we had such an awesome experience with Danny and Miktar the previous month, we decided to pick up where they left off.

Working with Game Maker - our newly acquired tool of the "indie" trade - Daniel set off to describe the theory of state initiators in Artificial Intelligence. This was our mammoth task for the avid game developers: using a turret system of guard houses, our character (or red ball, at least) had to navigate the stage without getting spotted by the evil turrets and destroyed ... easier said than done! The turrets were stationary and had a randomized clockwise and anticlockwise 360 degree view, so one was able to wait behind a wall until their backs were turned and then attempt to make a run for it. Once alerted, however, these bastard turrets would unleash a flurry of proton (aka "green stuff") attacks on our poor Player1. Sound simple enough to the seasoned Game Maker veteran. However, the "execute a piece of code" button had many a would-be programmer (myself included) baffled. Having only a very meek understanding of HTML and JAVA, the programming proved to be a minor bump in the proverbial road. However, once I thought about it - and I mean really thought about it - I realised that the IF statement is actually not that tough to get your mind around, and the jargon and computer speak mishmash soon became somewhat familiar to me. Soon, I was able to follow what in the blazes of green proton (or whatever destructive green gunk flowed out of our turret gun) was going on...

The AI (that's Artificial Intelligence for the layman, ha!), basically had a FOV (field of view) of 45 Degrees, which proved slightly problematic because we never set a cutoff distance (or "fog of war", for you Warcraft fans out there). So, the turrets could easily spot our Player1 even though (mouse_x, mouse_y, x, y) was on the other side of the screen. Thus, we coded a cutoff distance of 300-odd pixels and away we went, making our rooms look like mazes with turrets scanning for intruders, very much like a scene from the first Metal Gear! And thus, we got the AI to go from a passive searching state (which is the keyword here) to a tracking state once the target had been spotted to a skiet-skop-n-donner state once the target was in range of our green stuff!

All in all, the December Hotlab content was a great place to start after the trip by Danny and Miktar had left us with a new toy to play with and understand. Now, we have decided to carry the torch for Game.Dev in KZN. Not only that, but we've decided to include more than just Game Maker for our Game Development fanatics.

More details will follow on those developments at the end of this feature. Remember, kids... stay in school and keep living your dreams! Game Development is for everyone!

MUSHI MUSHI

THE KZN HOTLABS IN JANUARY!

It was a new year and an explosive start to the next annum of game development in Durban. Not only were all our regulars bright-eyed and bushy-tailed for this particular Hotlab, but attendance this month soared with the addition of a whole stack of new developers. Our humble computer lab was filled to the point of bursting, with barely enough workstations available for all the eager new learners. Definitely good reason to expand the facilities next month!!

The size of this month's group meant that the session was divided into two parts - first, an introductory seminar for those new to Game Maker, which went over the basics of this incredibly fun and useful tool and was delivered by Insomniac and Skinklizzard of the Game.Dev crew.

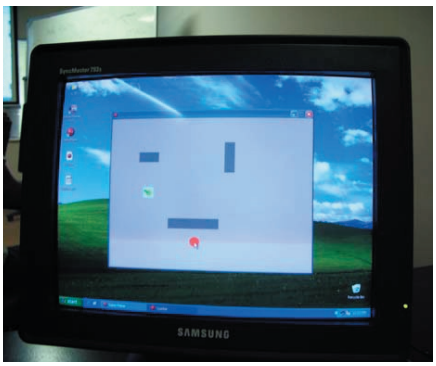
The second part of the Hotlab was once again dealt with by resident programming guru Daniel "SilentBob" Brewer, this time with a lesson on AI flocking and its applications in Game Maker. Afterwards, most people confidently described the seminar as being "flocking brilliant" - amongst some other very bad puns.

The introduction of the "newbie training course" in each Hotlab has proven to be a wise decision, further increasing the accessibility of the seminars and ensuring that even the greenest laymen walk away with the creation of a full, functioning game under their belts. The fresh crowd at the Hotlab definitely proved this point in their own creations - once the main tutorial was over, learners were encouraged to make their own creative additions based on what they learned as well as what they could gain through experimentation. It produced some stunning results from people who had, in some cases, never made a game before in their lives.

January's Hotlab has kept with our goal of being relaxed, friendly, educational and - most importantly - fun. Whether it was teasing the recent misfortunes of "Bernard the breakneck Boshoff", or just sitting down and making small talk between class sessions, that extra little human touch just serves to remind us all that there's more to the world of game development than bunches of sun-deprived social neophytes sitting in front of reams of code.

Be sure to make your way to next month's labs! Chances are that no matter what your level, you'll learn something new - and in the unlikely event that you don't, you still get to have oodles of fun amongst some like-minded game developers. Rock on, KZN, and may we continue to have a fantastic year of game development!

NANDREW



HOTLABS IN ACTION – KZN CHAPTER

2007 will consist of some very interesting seminars on game development and content creation.

These seminars will include game creation using the Game Maker framework, which will be presented and organized by members of the Game.Dev online community. For the n00bs, we have the talented Skinklizzard and FrozenPhoenix on hand to offer some introductory guidance to the framework and rapid game creation using Game Maker. www.gamemaker.nl

For the more experienced game designers, we have SilentBob to demonstrate game mechanics and some very interesting ways of approaching game development. We'll be offering support for engine coders in C++ and C# XNA game programming, as well as offering insight to various API's including FMOD.

For budding artists, animators and graphical content creators, we will have seminars using Adobe Photoshop CS2, Gimp, 3D Studio MAX, and Blender - to name but a few. These GFX Design Labs will be conducted by Grant de Lange of Indigo Child Studios www.indigochildstudios.blogspot.com.

For those who wish to learn the intricacies of Adobe Flash we have another Indigo Child Studios partner - The_Q? -- onboard to teach and inspire those who wish to make flash-based games in Action script.

A seminar on music and sound effects creation will also be conducted every now and again by the final two Indigo Child Studios partners, namely Weazelbub and MushiMushi. These seminars will be teaching the students about EQ applications and VST programming, using applications licensed through the studio such as Steinberg Cubase, Wavelab, and Propellerhead's awesome Reason package.

The goal for 2007 is to create multi-skilled individuals that have the potential to create various forms of content for their games. After each Hotlab, the attendees will be required to complete development tasks pertaining to the information learnt in the previous Hotlab as well as reaching goals for the next Hotlab.

MUSHI MUSHI

Hotlabs will run on a Monthly basis at IT INTELLECT in Durban 031 277 2000 and all participants will be contacted via e-mail.

To be added to the mailing list, please send an e-mail describing your area of interest to bernard.boshoff@gmail.com

NARBACULAR DROP

Developer:
Nuclear Monkey Software

Year of creation:
2005

Website:
http://www.nuclearmonkeysoftware.com

Genre:
Puzzle/Platform

Valve's upcoming title, Portal, has spawned a rather extensive cult following among certain gamers due to its unique gameplay and concept: To use a dynamic portal creation system in an effort to solve puzzles and navigate levels using basic physics with a complex twist. However, this play dynamic is not as unique as it seems.

Introducing Narbacular Drop. Yes, that's actually a name of a game, and a rather innovative game at that. Despite its unwieldy name and shoddy graphics, Narbacular Drop is indeed one of the most unique puzzle experiences ever created.

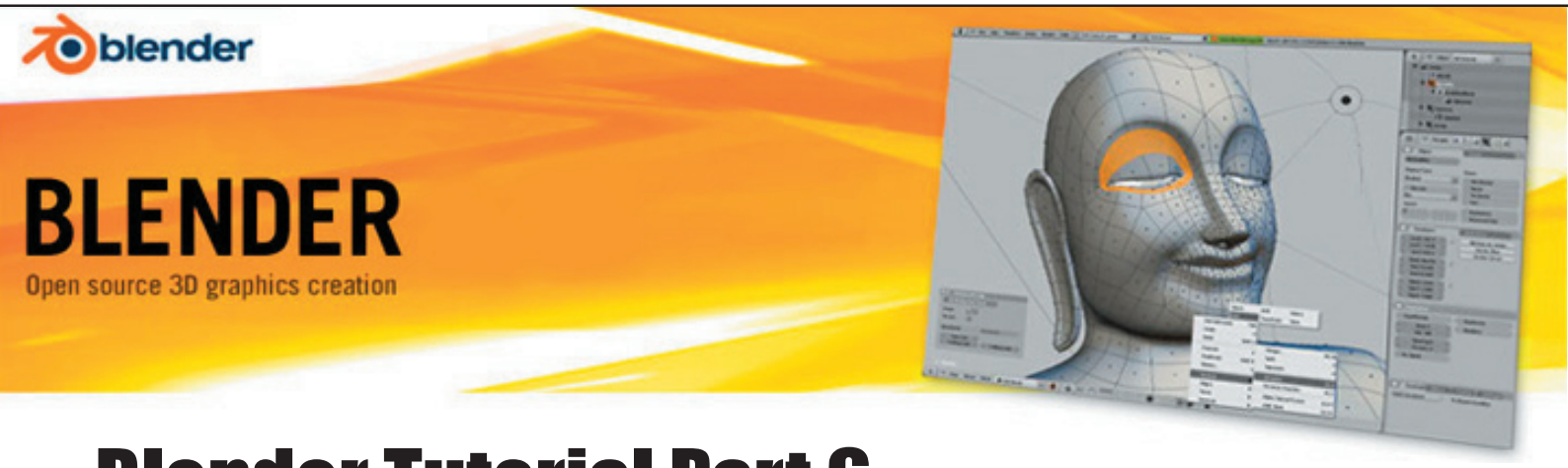
Being the spiritual predecessor of Portal, Narbacular Drop is essentially the experiment that originally founded the dynamics now popularized in Portal. The premise behind Narbacular Drop is simple: The player takes control of a princess - with a suspicious lack of jumping ability - who has been imprisoned by an evil demon. However, her prospective prison turns out to be an actual sentient being, capable of creating rifts within its walls. The princess is granted the ability to create these rifts at will, effectively linking two unrelated points in space by a doorway. Using this rather simple, yet versatile power, the princess must navigate the dungeon and eventually defeat her adversary and escape.

This relatively basic idea leads to near infinite possibilities for puzzle creation. The player is challenged to use this power - and only this power, the player has no other abilities - in innovative ways to progress through the game.

However, it is quickly apparent that Narbacular Drop is intended more as a tech demo than an actual game. There is very little content in the game, and the featured levels will take even a novice player less than half an hour to complete. The focus of the game is more on the functionality and potential of its portal system than anything else. Despite that, the game has earned a remarkable cult following, with hundreds of community-created levels available for download. Even though the game has been completed for over a year, and its developers are now under the employ of Valve, Narbacular Drop's website remains active, with community support carrying the game way past its expected lifetime. This highlights the immense potential that the game holds. Though this is a game with limited appeal, it should at least be played once, just to see the amazing concept in action.

For all those who were even slightly intrigued by the Portal trailer, this is a must have. The website's download are also features a documentation section, useful for budding game developers. It includes well-written design documents that detail the creation of the game, as well as other development related documents. Worth a look.

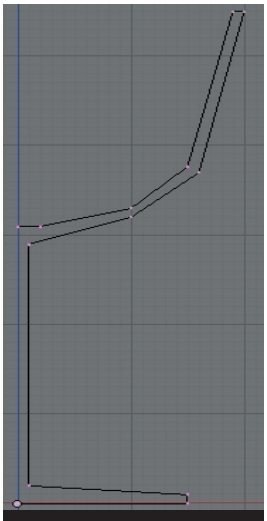
CHIPPIT



Blender Tutorial Part 6 : More Advanced Modelling Techniques

While quite a lot can be achieved with the subtle manipulation of the basic shapes available in Blender, more advanced techniques are available to automate and simplify certain tasks. We'll be using some of these to model seemingly complex objects quite easily. Firstly, we'll use one of the Blender curve objects to define the cross section of our model, a simple wineglass. Curve objects can be used to create subtle curves and bends, but are also ideal for defining complex cross sections of objects. Because we only need a rough approximation, we won't actually be creating curved edges though.

Remove the starting cube and, in front view, add a Path curve using the spacebar menu. In the editing buttons,under CurveTools, click the poly button to make the new curve behave like a basic polygonal shape. You'll notice that the curve is now simply 5 collinear vertices. Manipulate the vertices to create a shape like the one on the right. You'll need to extrude the end point a few times to create enough points.


What the cross section of your model should look like.

Once you've done that, change to object mode and hit Alt+C, or click the Object menu and select Convert Object Type, and select Mesh from the popup menu. Now change to top view and position the 3D cursor at the left edge of the new object.

Make sure you snap it to the grid so that it's exactly at the same location as the left-most vertices of your cross section. Hit Shift+S, or select Snap from the Object menu, and then choose Snap Cursor to Grid to do this. Also make sure that the new object's vertices are also exactly on the grid units.


The 3D Cursor Position

Once you're certain the 3D cursor is in the right position, change back to Edit Mode again. Under mesh tools in the editing buttons screen, change the Steps value to 32, and the Degr value to 360. Then, with all the vertices selected, click the spin button and watch your simple cross section being spun around the 3D cursor to create a glass.

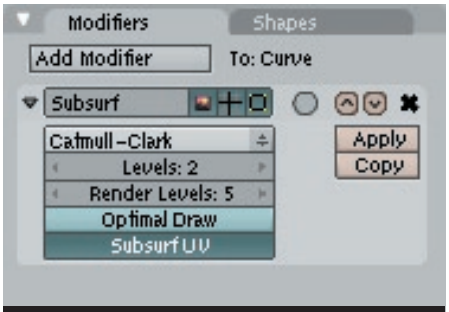
Because of how the Spin function works, it often results in some vertices being 'doubled'. In other words, sometimes there might be more than one vertex occupying the same space. This can

Using and setting up the Spin mesh tool.

cause the shading on the glass to be incorrect. However, this is easy to fix. Still in Edit mode, with all vertices selected, click the Remove Doubles button in the Mesh tools window. Sometimes, depending on how your cross section was created and the spin performed, the vertex normals can occasionally be pointing in the wrong direction, causing the shading to appear incorrect during rendering. If this happens, simply select all the vertices in Edit mode and hit Ctrl+N, or, from the spacebar menu, select Edit, Normals, Recalculate Outside.

Click the Remove Doubles button

Our glass is still a little rough at the moment, so we'll use a little trick known as the subsurface modifier to smooth out the glass and make it look more organic. Change to object mode, and under Modifiers in the Edit tab, click Add Modifier and select Subsurf. Increase the Levels value to see the influence of the modifier in edit mode. The Render Levels value determines how much the modifier will be active when the model is rendered. It is not recommended to use values higher than 5, especially not in edit mode, because this procedure is rather slow at high levels. Bear in mind that the subsurface modifier option has been moved since older versions of Blender, so, if you have an older version it may be in a different place.



The subsurface modifier

Finally, using the methods learned in previous tutorials, create a glass material and set the shading on the object to smooth. You can touch up the lighting and create a floor object for the glass to stand on. When you're finished, your render could look something like the image below.

And that's all for this time. Enjoy experimenting with Blender's more advanced features.

CHIPPIT

This final image for this article was rendered using the third party Yafray renderer. The Yafray renderer is ideal for creating scenes where raytracing is used extensively and may result in more accurate imagery than Blender's internal renderer can create. The file is available for free download from the following link: http://www.yafray.org/

The Final Render. What your glass could look like.

CODE RE-USE - MYTH OR REALITY?

Code re-use has been a catchphrase in the computer community for a long time. However, tutorials seldom or never give examples on how to re-use the tutorial code in other programs. Tutorials typically give an example of how to do something within a very, very narrow frame of reference.

For example, an ASP hello world program would look something like this:

```
[HelloWorld.asp]
<%
response.write “Hello World”
%>
```

Putting this on a server and accessing the page will certainly show you hello world, but how do you now put a “Hello World” message onto your own ASP web site? This little tutorial does not show any way of doing this, nor does it give any indication of how to go about doing it.

Another example of the same ASP-based “Hello World” program could look like this:

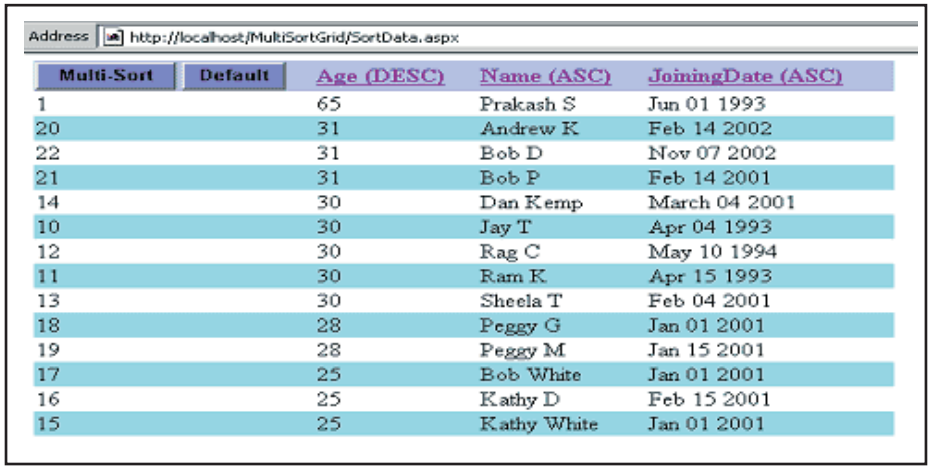
```
[HelloWorld.asp]
<%
Sub DisplayHelloWorld
response.write “Hello World”
end sub
%>

[MainPage.asp]
<%
DisplayHelloWorld()
%>
```

While adding a significant amount of complexity, the example code clearly shows the reader how to use the code that they have built within other applications that require the same logic.

The second example is made significantly small to show the concept of designing tutorials to encourage code re-use rather than designing tutorial code to work only in the specific scenario of the tutorial. When creating a tutorial, there is no reason that the normal programming structures of Classes, Factories, Encapsulation and Abstraction need to fall away.

Unless the tutorial is aimed at the novice programmer who is still learning the basics of a language (such as the above Hello World example), the tutorial code should be able to be copied out of the tutorial and used in the user’s program - this would therefore enable true code re-use.

CAIRNSWM



EDUTAINMENT CAN BE FUN!

The edutainment genre is a very elusive money-maker in the games industry, as parents are always willing to buy something which promises to benefit their future little world leaders. It could be the perfect way for you to start making money as an indie developer, since edutainment games don't require fancy graphics and tricks, just a solid idea to teach. However, the phrase "educational games" can produce a similar groan from gamers as the phrase "licensed games" does - but it doesn't always have to be this way. There are a few gems in the edutainment genre that prove learning can be fun. One old example is a game called Gizmos & Gadgets which was released back in the nineties and had many kids (including me) addicted to a game which involved answering puzzles to gain parts that would be used to build some sort of vehicle to race.

More recently, Dr Kawashima's "Brain Training, How Old Is Your Brain?" for the Nintendo DS has been a huge success across the world and is actually a lot of fun to play. But what makes games like Brain Training stand out from the many stinkers in the genre? In Brain Training, one of the exercises involves doing simple math calculations against the clock - how can a game that involves math sums be such a success? What could it possibly be doing? After having spent some time playing Brain Training, I've studied and learned a lot from it on what makes an edutainment game fun. I'll share here my views on what design points in the game should be learned from and applied to every edutainment game out there.

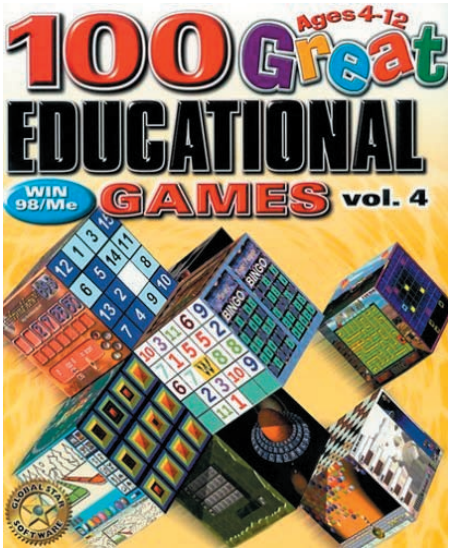
The most important part of a successful edutainment game is having a fun, solid gameplay mechanic to teach its target topic - for example, math or reading. Once you've hopped over that first hurdle and have a basic idea of what to teach, there are three major points which must be weaved into your design.

A sense of progression

In most games things usually start off really simple in a hold-your-hand tutorial before gradually getting harder. For example, in most regular games the player's character starts from doing small amounts of damage with a basic weapon and later grows into a stronger, arse-kicking hero as the player gets better at the game. This basic progression and learning curve must also be paid attention to when designing an edutainment game. In the first part of design, you should plan how you can make the player get better at the topic being taught while keeping the gameplay part fun. In Brain Training, Dr Kawashima gives a clear explanation before you start the training on how and why this will benefit your brain. After each exercise, your performance is scored and your results are graphed so you can get a greater sense of how you really are improving. This really simple but effective approach is a perfect example of giving the user a real sense of progression of their own abilities and brain power - and that is one of the greatest rewards a game can give.

Keep them coming back for more

Every good game has something addictive about it that keeps the player coming back for more. Every good edutainment game needs the same. If you want your game to do well, you're going to need a way to keep the player interested enough to keep playing and learning more. One way is to design the game or exercises so that you can never master



" Every good game has something addictive about it that keeps the player coming back for more."

Learning through edutainment.

them but can always get better and improve. Just like in a racing game, the player can always shave off a few split seconds against the clock and always get that bit better.

Rewarding the player and giving them goals and new challenges for the effort they put into the game is a great way to keep them coming back. A good example of reward from Brain Age is any day that you do training a stamp will be added to your calender and as you get more stamps things are unlocked, such as new exercises and features.

Accessibility

An edutainment game must be easy to use for everyone, whether young or old. Keep your demographic as wide as possible in design. The interface and controls must be simple, keep the focus on what you are teaching and what's happening on the screen rather than rewarding the player for having super-fast reflexes.

Another good design point is to make the game playable in short bursts. No matter how great your concept the majority of people can't put up with learning for marathon-long sessions. Rather try focus your gameplay on five-minute sessions which the player can come back to every day - like in Brain Training, with its daily training calender.

The edutainment genre can be tough to succeed in but with the points I've disscussed above, you can make a great game. Good luck and go out and make that ever-elusive fun edutainment game!

INSOMNIAC

MAKING OF NONEX 2: PART 2 MARKETING

Marketing? Isn't it a bit too early for marketing? Weren't you still busy creating a requirements and design document? You haven't even started programming your game.

Well, yeah - while this is all true, marketing is a very important part of any product, and must be considered from the concept stages right through to the day the product goes off the market.

This concept is very visible in the movie industry. The fact is, you know about most blockbuster movies months before they are released. This generates some hype and gets the audience's attention. However, there are a few key factors to consider when marketing anything. Firstly your product should be something the public would desire - for its functionality, entertainment value and necessity. With gaming, it is clear we are marketing entertainment. But the marketing must be of such a nature that it must feel like a necessity. To make anyone "need" anything, you must make them believe that this product is something they cannot live without, and can't understand how they lived without before.

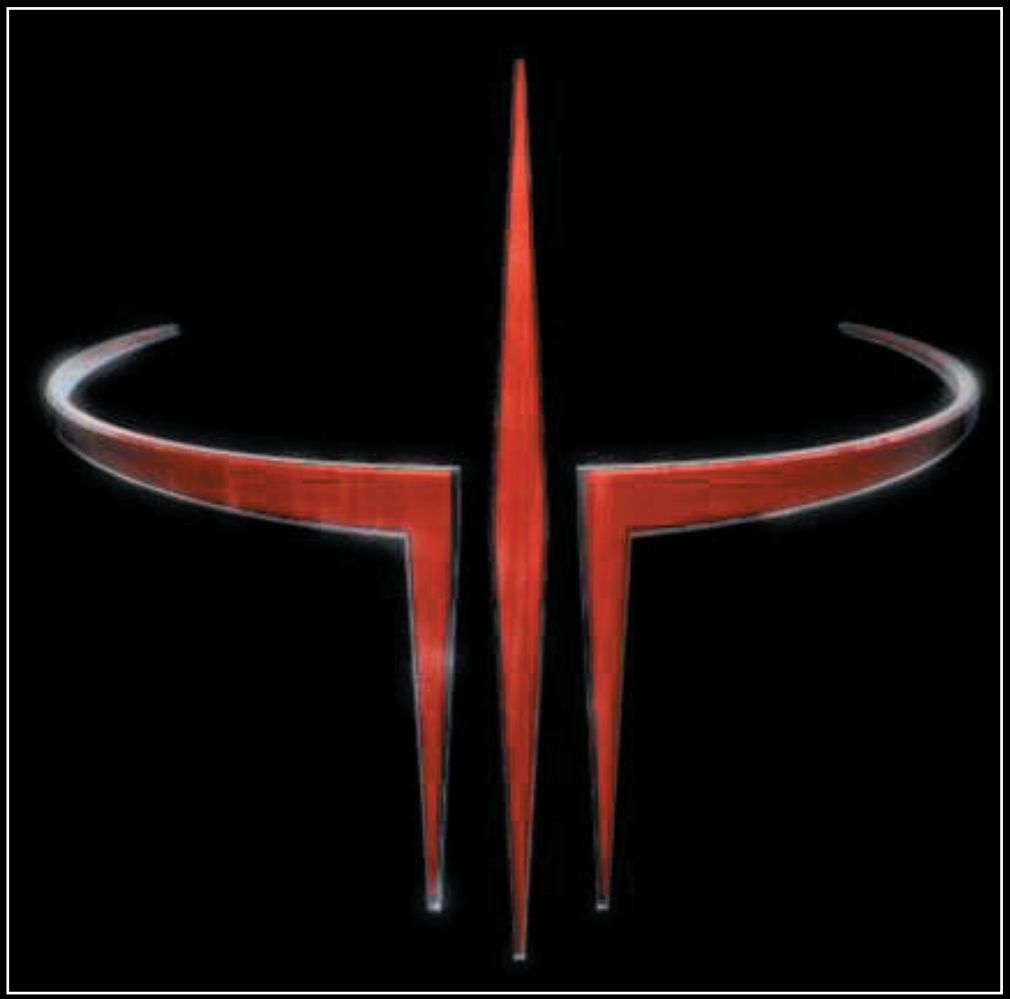
Secondly, there's the "catch phrase", "icon" or logo. Everyone who has an even small history of gaming will immediately be able to tell you what the Quake sign looks like. I liked it so much I made my own Quake 3 sign out of stainless steel as a medal around my neck. Unreal has a similar appeal. All of these symbols and labels are all part of establishing a brand. This is what I am currently busy with and will share some of that shortly.

Thirdly, you need to let the world know of your product/brand and its entertainment value. This is very easy for well-established studios like ID or Blizzard. They already have a great following of fans, and well-established websites. News of Diablo 3 would spread almost instantly across the globe. For a new company or studio, it is unfortunately much harder to get the word out. Even when the word starts to spread, the public will be very skeptical about the quality of the finished product.

To give you an example of a real live situation: how many independent movies have you watched this year in comparison to, say, the movies released by Paramount or Universal? In the same way, no one knew who Matt Damon and Ben Affleck were before "Good Will Hunting", and now they are household names.

The same thing can happen in the gaming industry. That one great idea, that one single game that sets you above the rest, is what can create a brand and following.

This brings me to the fourth step of marketing. You need a finished product. Kind of obvious, isn't it? What are you going to sell if you don't have a product? Well, companies sell things to us long before they exist. We were all "buying" into the game Half-Life 2 two years before it actually came out. Fact is, they could have decided to scrap the game even after all that marketing. But if you know what exactly your product is going to be, you can start "selling" it to the masses.



I am going to stop there for now with the "steps" to marketing, and just put this all into context regarding Nonex 2. I asked myself some questions about my game. What will make my game a must-have product?

Why would people want to play this game so badly that they will buy it off the internet?

How will people identify my product among the thousands of others out there? How will that "logo" identify the product with my studio? How is the public going to know what my game is about, and if it is something that will appeal to them? Will the public "buy" into my gaming concept months before it is released?

There're a lot of questions, but they all need answers. The fact is, if I can't answer these questions now, then I shouldn't waste my time and effort to even make the game. Thus, with all this in mind I have added a new document to my list before I can start making my game.

The documents mentioned last time were the "Design Document" and the "Requirements Document". Now I add the "Marketing portfolio" to that list. All of these documents will become available soon online for your reading "pleasure" ... as soon as Telkom decides to finally install my new ADSL line.

As a final note: Considering the new markets available (i.e. XBOX Live arcade) I still have one major decision to make that may indeed make or break Nonex 2's future. Should I stick with Game Maker, or switch to XNA? This will be the topic of my next article, as I weigh up the positives and negatives. See you all then.

HIMMLER

ROACH TOASTER 2: PICKING OUT THE BUGS

After the totally unexpected success of Roach Toaster 1, it led me to weigh up my options for a Roach Toaster 2. So, after my beauty sleep, I decided, "What the heck, why not?"

Roach Toaster 1 placed you in the command of an elite team of soldiers trained to destroy the neighbourhood's roaches. With each level, you were simply tasked to destroy all the roaches. The gameplay was turn-based, where at the beginning of each turn you could place as many units as you want within the range of your squad's "base". This wasn't without penalty, since placing more units a turn ate up your budget. Thus, you had to defeat the roaches successfully, but within this given budget. Then there was the aded complexity of the roaches breeding each turn. Finally, there were several unique units to choose from, each with their own strengths and weaknesses. Players had to use all the skills at their disposal to first control the situation and then exterminate the roaches.

In developing RT2, I first had to assess RT1. What made it great? What aspects made it so addictive? One of the key "fun" factors in RT1 was, in fact, accidental. Originally, I did not intend the feature to be implemented.

What was it?

It was the "what if" factor... If the player lost a level, which most of the times was a pretty close loss, it induced the thought of "What if I placed that unit slightly left?" Players kept coming back for more, thinking, "What if I had done it differently?"

So, armed with my new-found designing skills, I started design on Roach Toaster 2 immediately. What more would I want from "Roach Toaster"?

After having read countless Gamasutra articles (www.gamasutra.com), I arrived at two key concepts I wanted to try out: multi-level gameplay and meta-gaming.

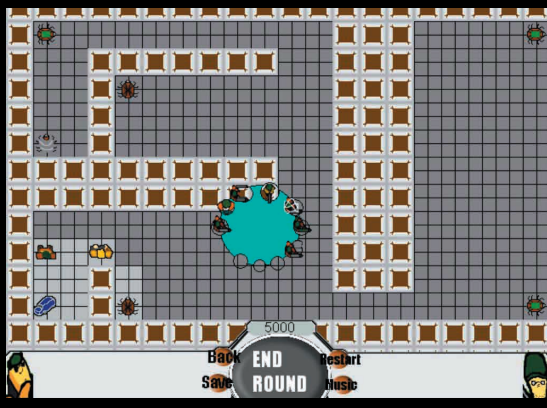
Multi-level gameplay refers to there being literally two (or more) areas of gameplay. In Roach Toaster 2, there is the "micro" level of gameplay, which consists of the levels themselves (like in RT1). Then there is the "macro" level of gameplay, where players have to strategically move their teams across "Big City" to defeat the root of the Roach Infestation once and for all (or at least we hope so)!

Meta-gaming is a difficult concept to explain. One could say that it's when players continue playing the game in hope of "getting more" even though it's not directly tied to the main goal of that game. Good examples are MMORPGs and Diablo 2. How many times have you kept on killing that boss just to complete your armor set?

RT1 had loads of cheesy humour, so of course I had to let it make a return. Two "valiant" soldiers narrate the game - Captain Hetfield and Private Reticer. But they aren't just for show, either - they help in the teams, when at the beginning of each level, they can be placed to help the squad. Each of them can hold one enhancement, which is an addition to RT2 which will ... well, enhance the battles in RT2! What is my goal for Roach Toaster 2? Roach Toaster 1 simply did not have the production value needed to show the world what it was about. With Roach Toaster 2, I intend to make a fully-fledged, polished, quality game with the originality of gameplay from Roach Toaster 1.

Next month, I'll tell you more about the graphics direction I chose and other intended features! If your hunger is increasing for more RT2, or you want to take a look at the original Roach Toaster, gotomysite,http://www.shotbeakgames.za.net..

TROOJG



FUNDAMENTALS OF COMPUTER MATHEMATICS: NUMBER SYSTEMS

Introduction

The mathematics used in the world of computers and computer programming can be quite different to the ones used in everyday life by most people. In a lot of cases, these differences can provide a barrier to new programmers coming to grips with the more intermediate and advanced areas of computer programming. This series of articles, starting with ***Number Systems***, will attempt to cover a few of the fundamentals of computer mathematics that all computer programmers should be aware of.

Decimal

In normal everyday mathematics, people use what is known as the Base-10 or ***decimal*** number system. The meaning of this is that there are 10 unique digits[1] which are used to represent all of the numbers in this system. This number system is also what is used ***most of the time*** for mathematical and numerical operations when writing computer programs. However, there are other number systems that can be also used while programming, some of which are more useful than the standard decimal system, depending on the context of their use.

1. (0-1-2-3-4-5-6-7-8-9)

Binary

The first of these alternate number systems is called Base-2 or ***binary***, and it is also probably the most well known number system when it comes to computers. Following the same rule as the Base-10 system, Base-2 uses only 2 unique digits[2] in order to represent numbers. This can seem a very strange way of storing numbers, but it is in fact just as intuitive as the decimal system once you realise that it follows the exact same principals. In other words, the smallest number is '0', which is then followed by '1'. However, instead of representing two as '2', you need to overflow the single digit column as there are no more unique digits to use. This is done by resetting the column to the base value of '0', and then incrementing the next column by '1'[3] which in turn forms the value '10'. While this may look like the number ten from the decimal system, it also represents the number two in the binary system.

If you are still confused, just think of it in terms of how the value ten comes about being formed in the decimal system. You first start with '9', which is the largest single digit available for decimal numerical values. Now, in order to reach ten, you need to overflow the single digit column by resetting it to '0', and then incrementing the next column by one. This is a standard principal for representing numerical values in all number systems, and if you can understand why it works, it will make working with different number systems simultaneously second nature.

2. (0-1)

3. Even though '9' isn't written as '09', it is always assumed that any 'invisible' number columns are actually filled with the '0' digit

Hexadecimal

While the binary number system may be the most well known alternative one when it comes to computers, it isn't the one most often used by programmers. Next to the decimal system, Base-16 or ***hexadecimal*** has become the number system favoured by computer programmers due to a few reasons. Firstly, it is a power of 2 value, which in turn produces numerical values that are very easy to store as well as read due to the binary[4] nature of computer design. Secondly, it allows for relatively large values to be represented by a relatively small number of digits. Following the rules laid down for the previous number systems, the hexadecimal number system uses 16 unique digits[5] to represent its values, meaning that while a value such as thirteen would be visualised as '13' in the decimal system, it would be represented in the hexadecimal system as merely 'D'.

4. The lowest level of computer logic is designed to run along the lines of the binary math system, with everything either in the on ('1') or off ('0') state

5. (0-1-2-3-4-5-6-7-8-9-A-B-C-D-E-F)

Octal

Often thought of as hexadecimal's ugly sister, the Base-8 number system known as ***octal*** is another alternative number system that is used in some circles of computer programming. However, most of the popularity that it once had has been overtaken by the hip and sexy hexadecimal system. This is mainly due to the fact that octal's 8 unique digits[6] can only store numerical values that are half of the size of the ones that hexadecimal can store. So while octal does conform to the power of 2 standard that makes hexadecimal so useful with computer architecture, it can be messier to work with because of the larger number of digits that are necessary to store the same value.

6. (0-1-2-3-4-5-6-7)

Conclusion

While the mathematics necessary for computer programming can often differ from those that most people use in their normal day-to-day lives, they do in fact have much in common with the elementary level mathematics that everybody has learned from a young age. This point is exemplified greatly in the above number system descriptions, as they are shown to be nothing more than slightly modified versions of the standard 10-digit decimal system, who follow the same rules and are used for the same purpose.

COOLHAND

CODING ETIQUETTE: INTRODUCTION

There are many different schools of thought when it comes to what programming, or "writing code" should be about. The most common one is that programming is merely a means to a end. In other words, it is only the final result that is of any importance, and whatever steps that are taken along the way are inconsequential so long as the program does what it was designed to do.

Programming in this style is usually the quickest way to achieve one's goals. However, more often than not, this method leads to very messy and in most cases completely unmanageable code. At times though, this is actually a completely acceptable result, such as cases when work is being done on small-scale projects by single programmers that will have no need of maintenance or upgrading in the future.

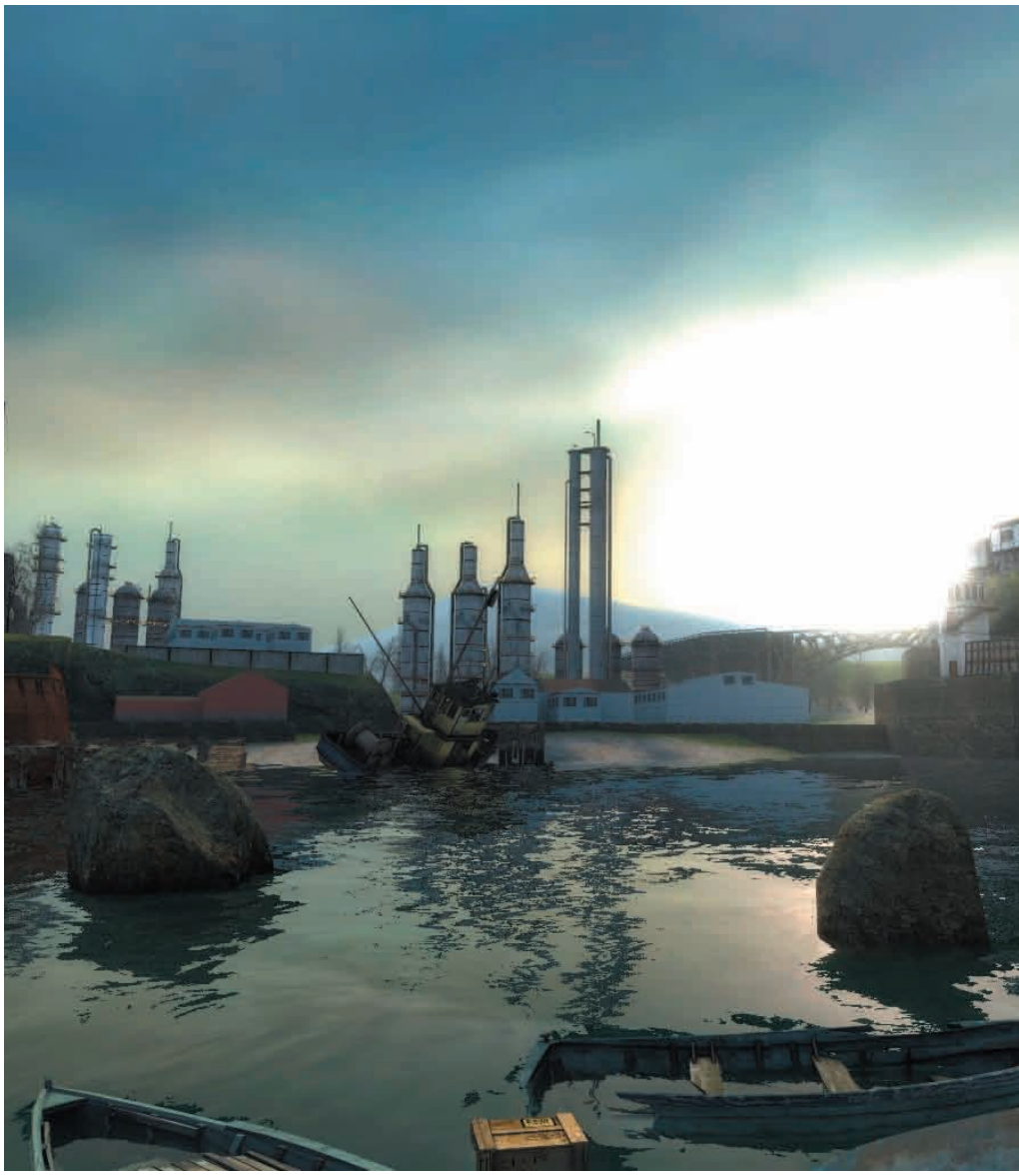
There is another school, however, which teaches that not only are the means important, but that they are just as important if not more so than the end result. The thought behind this method of programming is that most projects are not small ones, created by lone programmers, and that they will have a lifespan that will include maintenance, upgrades, and even full scale overhauls at some point in the future. Coincidentally, this description is best used when talking about the programming involved in game development.

In fact, game programs tend to follow this lifespan more religiously than the average corporate applications that are taught in the average Computer Science program at the average University. In a lot of cases, this lack of understanding with regards to the intricacy, scope, and complexity that is involved in game development can cause Universities to overlook the importance of not merely teaching proper coding etiquette, but the need to stress just how important its use is in the real computer programming world. Unfortunately, this need is an imperative one for programmers interested in game development.

For game programmers, making use of proper etiquette while writing code is just as important as writing efficient code. This is mainly due to the large team sizes and scope that are a part of game development today. When working with many other people on a single project, it doesn't help to have the most efficiently written code possible if the other people on your team need to work with what you have done but are unable to do so, due to such things as a lack of sufficient commenting or incoherent variable naming on your part.

In the coming issues of Dev.Mag, we will be discussing various aspects of coding etiquette that all programmers working in game development should be aware of. Keep in mind that at the end of the day, there is no real right or wrong way in how you go about your programming. The coding etiquette sections that we will bring before you will merely be guidelines and suggestions of practices in order to improve how you write your code, and for you to choose a style that suits how you already work.

COOLHAND



A QUICK HISTORY OF GAME.DEV

It was one overcast morning in January 2005. The NAG forums were still in their first incarnation when the powers that be (which in those times was basically Miktar) created a new section in the forums… <<Game.Dev>>

For me, being an avid NAG poster and wannabe game maker, this was a god-sent call to greatness. The first competition was upon us. All it stated was to create a game…

With the few entrants that entered the first competition (aka "Comp 01"), things looked bleak, but with the support of Danny "Dislekcia" Day, the ultimate driving force behind the Game.Dev community, and his preaching of "make games, not engines" motto, more curious NAG posters were soon snatched up.

"Comp 02" arrived with the goal of creating a game with only circles and squares, which was another example of Dislekcia's motto - to not create engines with awesome graphics, but rather a finished game with good gameplay.

Things slowly began to grow into a very promising endeavour. People were beginning to join NAG not because of the forums themselves, but because of the Game.Dev section alone!

Right before "Comp 03", which prompted us to do a remake, the NAG forum shed its old skin for a new one. This did not deter us in the least, as our community was still growing. "Comp 04" was a great moment of progress for the Game.Dev group. Lots of people joined to participate in the competition, which prompted us to make a game with simple rules that led to complex gameplay. "Comp 05" started in late 2005 soon after the forums shifted to a new home again.

Many heralded our section of the forum as the only sane place to post, away from trolls.

After the 6th competition - which ended our first year as a community - the notion of a game development magazine graced Game.Dev. It was ambitious, with many doomsayers saying that an online mag never lasted beyond the third edition. But we had the power - the skill to defeat that first boss - and thus Dev.Mag was born in early 2006 alongside the seventh competition.

2006 itself was almost a blur at the beginning. Many people joined and tried their hand at game development. Some fell to the wayside, discovering that it simply wasn't for them, while the community slowly paved its way with those who joined and subsequently stayed, participating in another two competitions while keeping the spirit of Game.Dev alive and strong.

Soon, there was lots of excited talk about rAge 2006 floating about the forum. Lots of things were scheduled for rAge, but no-one expected what was to come.

game·dev
It's what you wish you were doing.

Soon, the news struck. To celebrate our tenth competition, an enormous cash prize of R10 000 was up for grabs. This was the turning point, the beginning of a new era for Game.Dev! Lots of new faces tried their hand at making a management game, while previous Game.Dev member rejoined to take part in this legendary competition.

Besides the quality of the games, which were taken to a new high, rAge saw the Game.Dev stand pulling an unexpected amount of visitors. Lots of people, beginners and experts alike came to listen to our talks.

Now at the beginning of 2007, things look better than ever! Currently, Game.Dev is sitting with its own website, has members and Hotlabs spread throughout the country, runs its own publication and has lots of newly fostered game developers to show for it - all of this stemming from a small competition post in a gaming forum two years before ...

TR00JG

THE HISTORY OF I-IMAGINE

I first heard the name "I-Imagine" while sitting in a Mexican restaurant in Redmond, Washington, about 2 kilometers from Nintendo of America's headquarters. I was at a booth table next to a window. You know, the kind with padded benches that you would always rather sit at? There were rabbits outside. Lots of rabbits. So many rabbits, actually, that I quickly resigned myself to the fact that no matter what I ordered from the menu, it would contain in some shape, way, or form ... rabbit.

It was during the fall of 1998 ... September I think. I was having lunch with Dan Wagner, a former classmate and project partner of mine at DigiPen. Not the "DigiPen Institute of Technology" that exists today in Redmond (also about 2 kilometers from the plague of rabbits), but the "original" DigiPen Computer Graphics School that was run up in Vancouver, British Columbia.

Dan and I had both graduated in May of that year and had gone our separate ways, only to meet up again quite close to where it all began. Dan actually left DigiPen after the 3rd semester had ended in order to take up an internship at a company in Las Vegas. Afterwards, he went on to do some contract work for a company out of Portland, Oregon, which is where he was coming from when this meeting was arranged. I happened to be down at the then relatively new DIT campus helping out for the day with some technical stuff if I remember correctly. My normal day-job was as a teaching assistant for the final graduating class at DCGS back up in Vancouver, but every once in a while I took the 3 hour bus ride down to Redmond in order to help out whenever they needed me.

I didn't really know why I was meeting him there, as I stared out the window wondering which ball of white fluff was most likely to end up on my plate with guacamole and sour cream. Dan had arranged with me a few weeks earlier to meet him for lunch on that day as I was going to be closer to Portland... although I suspect it had more to do with him not wanting to deal with the immigration officials on either side of the Canada / US border...

Once we had sat down and finished all of the rabbit-related commentary that we could think of, Dan got to the point. He was looking to move back to his home country of South Africa and he was interested in starting up a game development studio there. He had already started to gauge some interest from potential investors there, and was now looking to convince some of the people with whom he had partnered with at DigiPen, as well as some he had met while working in Las Vegas, to join him in this venture.

Needless to say, I was quite surprised at the end reason for this meeting. However, I had even greater feelings of excitement coursing through my brain at Dan's revelation. Here was this great opportunity for me to not only be a part of starting up a game development company where we could hopefully work on our own

ideas and call our own shots, but to also have the chance to move to somewhere as exotic as South Africa in order to do so. Immediately, I was very interested in the whole idea although I also had quite a few reservations about moving to a place that would, without a doubt, be quite different from anything that I was used to.

Fortunately, Dan wasn't looking to get a solid answer from me at this time, as there was still a lot of leg work for him to do. There were other former colleagues whom he had yet to approach, not to mention that he still had to finalise suitable investment within South Africa that would enable this venture to take off. It was a bit of a catch-22 situation for him, but the thought of those rough seas didn't seem to do anything to diminish the gleam in his eyes when he talked about wanting to be the first person to establish a successful game development company in his native country.

At the end of lunch (which I am glad to say actually did not appear to contain any rabbit-like components or byproducts) I was quite excited. I told Dan that I was very interested in what he was doing, and that he should keep in touch with me when he had managed to get along further and had more concrete facts to tell me. Afterwards, I headed back to DIT to finish my work for the day, then boarded the bus back to Vancouver, all the while thinking about this cool opportunity that potentially lay before me.

I still wasn't sure what I would say if some day Dan got in touch with me and told me that he was ready to start up I-Imagine. I was happy with my job at the moment and I knew that in May, once the last graduating class left DCGS, I would be welcome to join the newly formed division of NOA called Nintendo Technology & Development, as they had very close ties to DigiPen. However, I told myself that I would worry about crossing that bridge if and when I got to it. Just the fact that I had two such terrific opportunities for my future before I had even turned 21 was more than I could have hoped for.

To be continued...

COOLHAND

Fortune telling – game development in 2007

Right, so it's still the tender days of a fresh new year. For some, it's the beginning times of school or university - and some working people out there might also have had the benefit of an extended holiday before putting their noses to the grindstone again. For others, it might mean other, bigger things, or even nothing at all. Whatever your personal scenario, it's almost certain that the year 2007 is going to be an interesting one for game developers. By "interesting", one doesn't necessarily mean good or bad (though the former seems more likely), but when stuff happens, it's bound to be big either way. Internationally, we're looking at some pretty hefty issues that people are going to be occupied with for most of the year. The Game Maker community, for a start, is anxiously awaiting the arrival of GM7 (if it's not here already by the time you read this). It's quite a serious piece of software by now - with 7 major incarnations, a widespread following and even its own book released last year, it's not surprising to realise that the tool's recent foothold in South Africa is rather small compared to its already firm entrenchment in many other countries.

With other game crafting frameworks such as Torque also vying for attention, the average game developer isn't really going to be short of tools (and powerful ones, at that) to practice their art with. Heck, that's not even mentioning Microsoft's beloved XNA, which some people still don't believe exists based on the it's-far-too-good-to-be-true principle.This development tool is pretty much in full swing by now and game devvers all over are ripping into

it like a pack of hungry wolves in a sheep pen. The awesome flexibility and the exclusive hold that the tool has on the 360 platform are partly to blame for such rabid behaviour. And it seems like it can only get better as the year progresses. Considering the above, looking for something to make you a game in 2007 will probably just entail a peek around the proverbial corner. If something's not lying at your feet already. Of course, there could be a nasty little hiccough regarding a particular platform. Sure, the console market has been blown wide open by the presence of XNA, but some worry about what the advent of Vista will do to the faithful old PC indie development. In essence, it seems likely that safety is going to be favoured over flexibility - bad news for the less formal game developers and marketers who have to get their product accepted by a system that is, quite frankly, designed to accept as little unauth third-party produce as possible. Security for the end user is a noble goal, but is it ultimately going to cost the big bad dev machine?

Well, regardless of the new OS's impact, it shouldn't affect the legroom that game development is going to enjoy in South Africa this year. The Game.Dev organisation is by now entering its third year of existence, backed by some very solid successes and some equally solid industry partners, and has some awesome plans for 2007. Hotlabs have already been established in both Johannesburg and Durban (with hopes being held out for Cape Town as well) and the organisation had an undoubtedly strong presence at last year's national rAge event. For 2007, speculation turns to holding workshops based at schools and universities, as well as establishing a game development presence at broader-focus events such as Scifest. The ranks of game developers are also swelling as ITI opens its doors to a whole batch of eager new dev students this year. The formalisation of the industry in South Africa is quickly rising to international standards - albeit on a smaller scale - and soon, perhaps, I-Imagine won't be alone in the arena of official South African game development companies. Dev.Mag itself is recording a fantastic swell of readers in recent months, with official download records doubling in December and the arrival of new game developers to the Game.Dev forums accelerating at an exciting rate.There's no reason why the average South African game developer shouldn't look forward to the potential that 2007 holds. As mentioned already, it should be an interesting year ...

NANDREW

IF LIFE WERE A VIDEO GAME...

(REVOLVING STRANGELY AROUND STICKMEN)

...WE* WOULD ALL BE VIP(pl), WITH HAIR AND FUNKY GLOVES IF WE WERE SO INCLINED.

*APPLIES TO GAME DEVELOPERS ONLY

STOP THE PRESS !

Just before this edition of Dev.Mag got published, the Game.Dev forum was practically flooded with awesome news.

First off, Blender-oriented website, ***www.blendernation.com***, picked up our mag the other day and gave it a front-page news article, literally boosting our readership by thousands. Thanks guys, we're glad that our tutorials have been appreciated on such a large scale!

Secondly, the Game.Dev organisation has had some very promising meetings with a few high-ups which we're all *very* excited about. If all goes well, there could even be some TV appearances involved, so watch this space for more info!

Thirdly ... well, let's just say that we've got even more aces up our sleeves, and what we have to say will probably have explosive results for game development in this country. Expect to find full coverage of some of the best stuff to ever hit SA game development when Issue 11 gets released next month!

http://www.blendernation.com/2007/02/16/blender-user-wins-subdivisionmodelingcom-challenge/