October, 2007

Issue 8

MarkUp

EXCLUSIVE: The Warriors Path!

Pokémon Twilight

Plus...

"The Making of" Dark Hive RTS

Development Journals!

39 DLL Tutorial

Drawing Wavy sprites.

Object Oriented Programming

Book Reviews

And Much More!

Interviews

Resources

Previews

Reviews

Tutorials

Editor's Desk

Game Maker 6 might have failed to impress, Game Maker 7 made few improvements but snobbed off any other project that dared use it's file format. Overall, the past few years of Game Maker have been a bit of a disappointment. But, I'm excited!

Why? Because competition in the "build-you-own-game-without-any-programming-skillz" arena is heating up, and competition tends to bring out the best in all products concerned. With G-Java and Stencyl nearing completion, YoYo Games will begin feeling the heat from the community to provide features that they can't already get elsewhere for free.

Now is a great time to start bantering for the feature you want too. These corporate types rarely think for themselves, so, here are some suggestions that I've had for a while:

- More 3D tools, and more tools that are for registered members put in the free version
- Free premium sprites and sounds for registered members (and lots of them!)
- Support for Game Maker to run on Mac and Linux (and that already is being worked on).
- Make a game source standard, like .ugs for Unified Game Source, so the various game IDEs can co-exist (this will bring in more funds and fame for YoYo!)

Yes, I can hear some of these wishes in the wind, get ready for some real releases soon!

See you next month!

Robin Monks■

You might be confused when looking at the table of contents on right to see that we have a relatively smaller amount of tutorials this issue. But don't worry, we're not turning into a game-oriented magazine, but we're introducing code and technical tutorials in a new form: **Development Journals** and "**Making of**" articles, where the developer of the game describes the cool techniques used to create the game.

Even though we are satisfied with the amount of technical content for this issue, we give you a promise that future issues will be further improved to have the amount of tutorials you are used to in addition to all these new features. To do that, we need your help.

Welcome Back!

Contributors

Robin Monks	Sr. Editor
Eyas Sharaiha	Editor
Andris Belinskis	Editor
Philip Gamble	Writer
Bart Teunis	Writer
José María Méndez	Writer
Jonah Turnquist	Writer
Tarik Abbara	Writer
Jono Alderson	Writer
XD005	Writer
Brad "Revel"	Writer
Veeti Paananen	Writer
Dan Meinzer	Graphic Designer

Table of Contents

Going Further

You may have read the “Setbacks and going Further” wrap up article at the end of the previous issue, where we announced exciting changes to the review system, and new types of articles introduced.

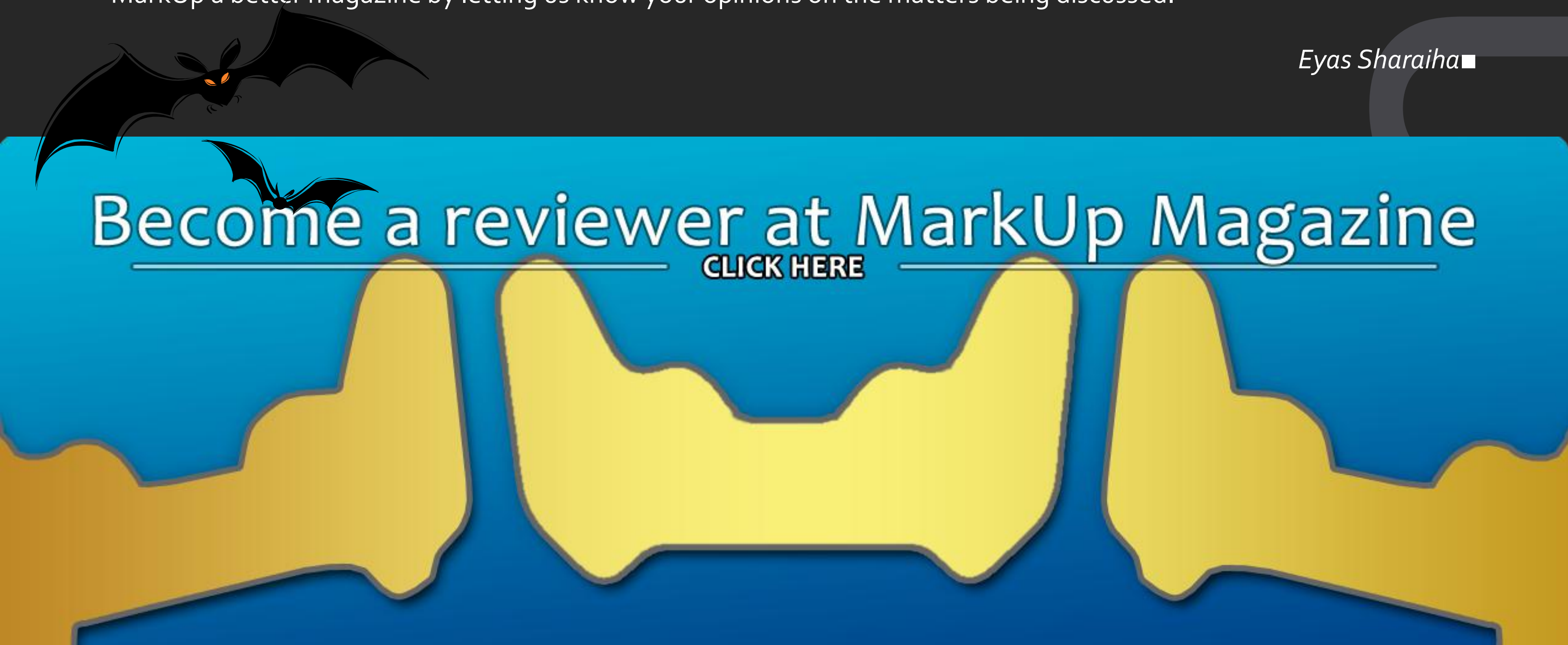
We have finally added the ‘new and fabulous’ **MarDar – the MarkUp Radar –** in two parts: WIP and CREATIONS, so that we could showcase some cool games and review them in an extremely brief way – just so you could know what good games are out there in the Game Maker Community.

We’ve also introduced the new section “**The Making of My Game**”, this time with **Dark Hive RTS**, a strategy Game Maker game created by ArKano. ArKano discusses how he made the game, the techniques he used, and the design concepts he followed.

We also introduced another type of articles which we didn’t previously announced, and that is the **Development Diary** section, where game developers talk to us about the WIP games they are working on *right now*, and give us an insight on what exactly is going on with game development. This section isn’t meant to showcase WIP games, but rather to show the readers how the game is being developed, so that the user could take advantage of all the wonderful techniques in their own games. We will be following the development journal with the same games for a number of issues, which explains why the Journal entries for this issue might be considered more of an introduction, with little code. This will change in the future, as the authors don’t need to introduce you to the various aspects of the game.

Wanna help? Well, you could apply for staff position at MarkUp here, we always need more writers and contributors to the magazine, so that we could have better content in the future. You could simply join the GMking.org Network forums, and participate at the MarkUp forums in there by giving suggestions about future issues and issues in development. You could comment in the MarkUp site about all newly released issues and give us suggestions from there if you don’t want to register at any forums. And finally, you could jump in the GMC discussion topic about MarkUp to give feedback, discuss and debate issues with other members at the GMC, and therefore help make MarkUp a better magazine by letting us know your opinions on the matters being discussed.

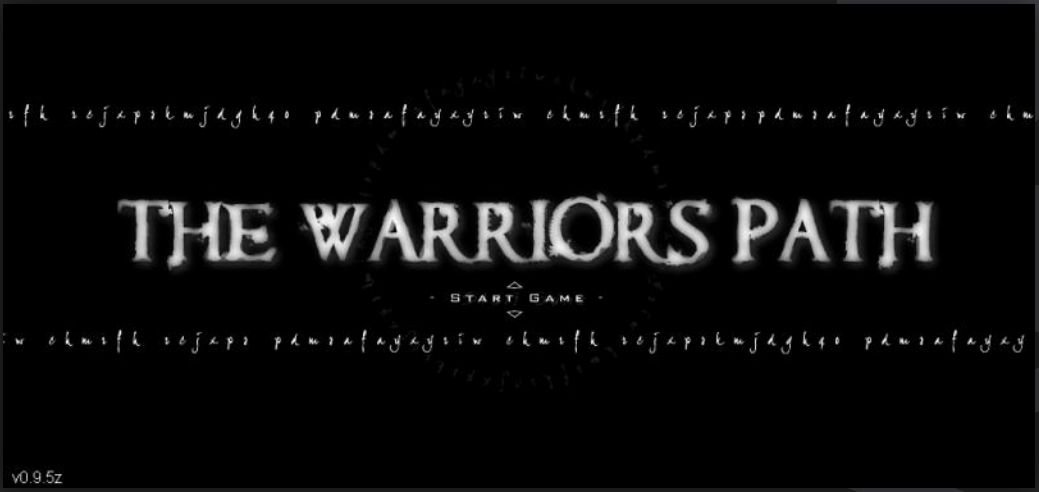
Eyas Sharaiha■



THE WARRIORS PATH

The Warriors Path, by Combustion Entertainment, is an upcoming Game Maker Massively Multiplayer Online game, which aims to deliver the best 3D MMORG Game Maker game graphics.

If anyone looks at the screenshots they have on their site, I think that we can all agree that graphically, this game does break many 'barriers' that many have thought existed.



Overview

We'll examine and preview the game in more detail, and show you – our readers – never before seen screenshots of the game, and discuss other features of the game.

Above, on the top right, you can see the "menu-in-progress" of The Warriors Path. As you can see, the amazing graphical experience stars right from the beginning; the menu is extremely simple, with nothing but a plain black background and an awesome logo, but it still gives you a strange feel: "wow!"

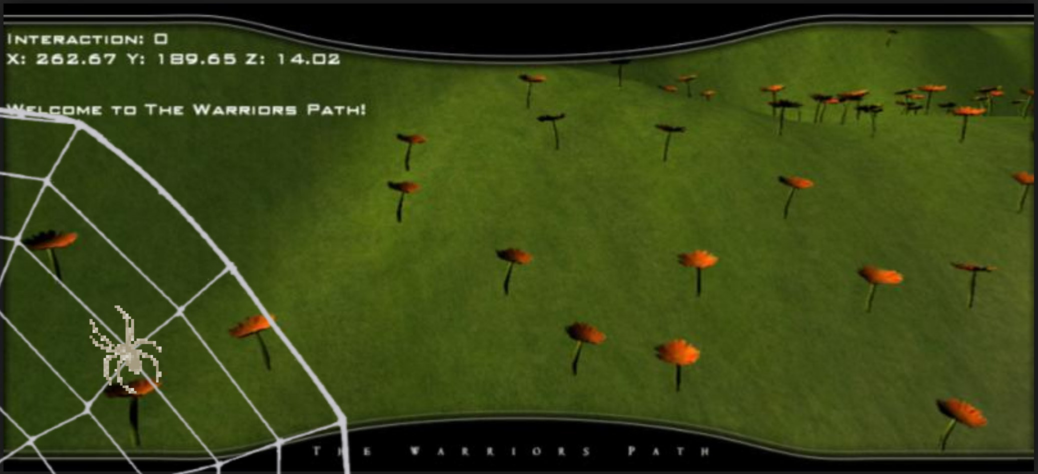
One of the awesome features of the game is the vegetation, it sounds silly, but they are paying a lot of attention to the plants and other types of vegetation which responds to collision and movement. I was lucky enough to get a working demo of the vegetation and movement system, and it seems great.

The way plants on the floor respond to the player moving onto and off of them is marvellous; the graphical quality of the models and textures is also remarkable.

It is quite clear how much the game pays attention to detail. As it could be seen, the graphics are incredible, the plants are great, the trees look really good – but that's not all what's the game about. The game uses Ultimate 3D (ultimate3d.org) 2 as the 3D engine for the game. Ultimate 3D itself employs great features, including vertex tweening, lighting and fog, cel shading, pixel shader and shader management systems, particle system, and full-screen anti-aliasing, as well as much more.

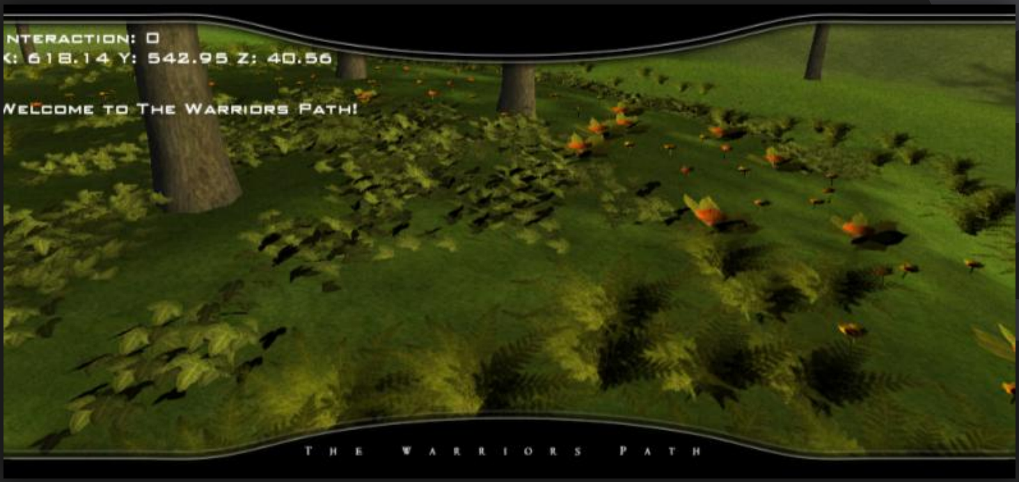
Though the game does succeed with eye candy, there's really more to it than just that, as I have said before, it has really smooth movement and vegetation systems, but that's really just the tip of the iceberg.

When it comes to music, the game is also *magnificent*! I do consider myself



↺

lucky enough to have gotten a sample track, and it's really great. I can't help but remember (from movies and other games) the ancient glory and greatness of the successful warriors and knights when I hear this. It is really well done, high quality and bit rate, and again, awesome tunes!



Other Game Features

Total Interaction and new advancements, including:

- Bump mapping Effects,
- Post Screen Shaders,
- Vegetation that moves under you,
- Custom Players,
- Particle Effects,
- Interaction with the smallest of things.

Game Requirements

A tentative system requirements list has been made; details might change for the final release.

- Windows XP (Home, Pro or Media Edition) [Compatible, but not tested for Windows Vista]
- 2.0ghz + CPU (Dual Core Recommended but not needed)
- 1gb Ram (DDR or higher)
- 128mb+ Video Card (Supporting Pixel Shader 2.0)

Those who use integrated graphic cards (onboard GPUs) are unlikely to be able to properly run the game, unless the integrated card is of high quality (i.e. Intel GMA 950 or higher).

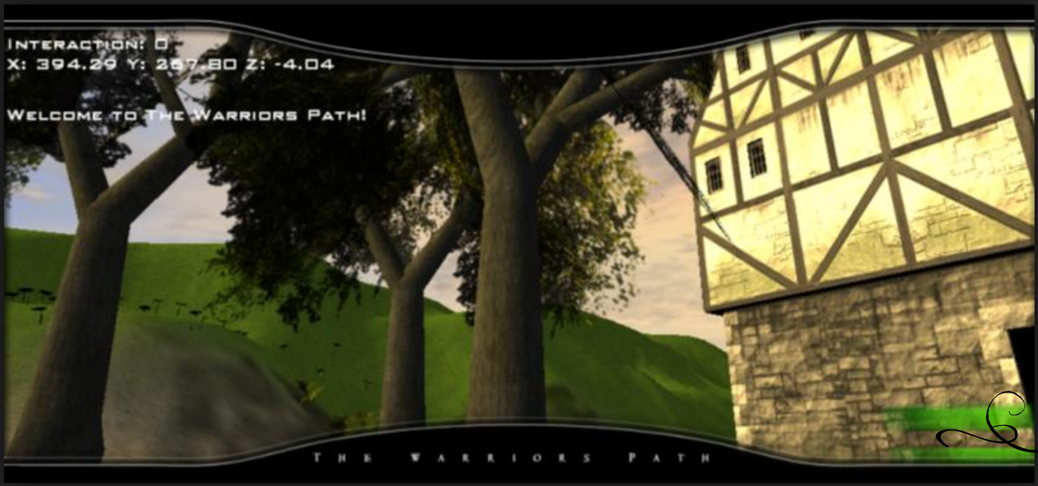
Conclusions

It might seem like I'm a big fan of the game, and that I'm just praising every part of it, but I – as everyone else – have my own concerns about the game.

I *am* excited about the game, it does seem like it might up the bar for Game Maker games. However, I'm still not so sure if the game will come out well. The developers might be overwhelmed by the level of detail and concentrate on interaction with smallest objects (which is a cool feature), graphics, and sound effects, and forget about things like gameplay, a simple 'storyline' (even for an MMO), and good game design (level design).

The game looks promising so far, I just hope it'll keep moving at the same pace at least.

Eyas Sharaiha■

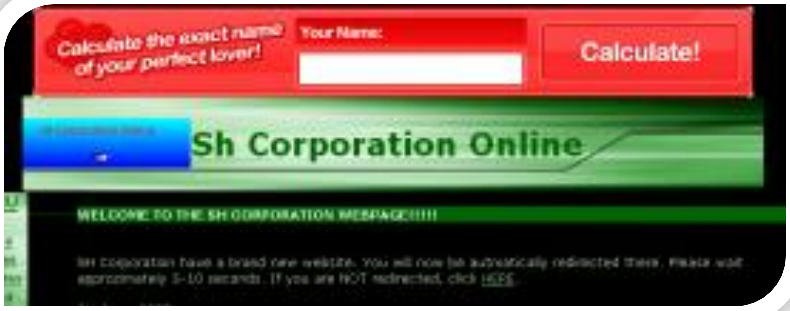


On your Game Maker website

10 things to avoid...

1. Pretending to be something you're not

Major corporations who host with FreeWebs, please. Amazingly ridiculous and yet so many people still do it. Adding 'Corporation' or 'Incorporated' after your hastily thought up, and no-doubt unoriginal, Game Maker team name isn't big or clever. It's just plain ridiculous and in some jurisdictions this is also illegal.



WELCOME TO THE SH CORPORATION WEBPAGE!!!!!

Sh coporation have a brand new website. You will now be automatically redirected there. Please wait approximately 5-10 seconds. If you are NOT redirected, click HERE.

2. A forum with one topic

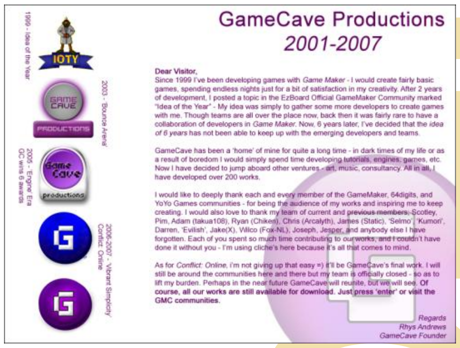
Just because you can add a forum to your website doesn't mean you should. Empty forums with row upon row of zeros going down the page suggest a site is dead. If the only posts on your forum are made by the admin or are two or three months old people will also reach the same conclusion. If this is the case there is no reason to have a forum. It will only be a matter of time before your forum becomes a magnet, attracting your only active users – spam bots who will litter your forum with porn and drug links.

3. Huge graphics

Bitmaps are not designed for web use. Despite the massive growth in the number of people connecting via broadband Internet providers some people still use dial-up. Even so not only do large images and those saved in the wrong format frustrate your users and eat up your bandwidth allowance but also, as is often the case, the images serve little purpose.

4. Using graphics to display text

Why oh why? Some people go to great lengths to create elaborate graphical designs for their sites where all the page text is included as part of an image. Not only does this waste bandwidth, make rectifying mistakes harder and prevent people copying links from your site but your search engine rankings will suffer.



GameCave Productions
2001-2007

Dear Visitor,
Since 1999 I've been developing games with *Game Maker* - I would create fairly basic games, spending endless nights just for a bit of satisfaction in my creativity. After 2 years of development, I posted a topic in the EzBoard Official GameMaker Community marked "Idea of the Year" - My idea was simply to gather some more developers to create games with me. Though teams are all over the place now, back then it was fairly rare to have a collaboration of developers in *Game Maker*. Now, 6 years later, I've decided that the *idea of 6 years* has not been able to keep up with the emerging developers and teams.

GameCave has been a 'home' of mine for quite a long time - in dark times of my life or as a result of boredom I would simply spend time developing tutorials, engines, games, etc. Now I have decided to jump aboard other ventures - art, music, consultancy. All in all, I have developed over 200 works.

I would like to deeply thank each and every member of the GameMaker, 64digits, and YoYo Games communities - for being the audience of my works and inspiring me to keep creating. I would also love to thank my team of current and previous members: Scotley, Pim, Adam (takua108), Ryan (Chiken), Chris (Arcalyth), James (Static), 'Selmo', 'Kumori', Darren, 'Evilish', Jake(X), Wilco (Fox-NL), Joseph, Jesper, and anybody else I have forgotten. Each of you spent so much time contributing to our works, and I couldn't have done it without you - I'm using cliche's here because it's all that comes to mind.

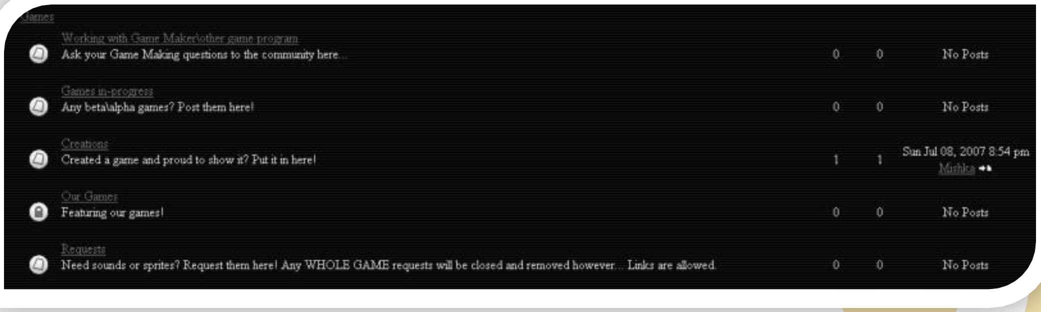
As for *Conflict: Online*, i'm not giving up that easy =) it'll be GameCave's final work. I will still be around the communities here and there but my team is officially closed - so as to lift my burden. Perhaps in the near future GameCave will reunite, but we will see. **Of course, all our works are still available for download. Just press 'enter' or visit the GMC communities.**

Regards
Rhys Andrews
GameCave Founder

5. Firefox incompatible sites

As Microsoft's dominance of the browser share has been steadily decreasing over the past few years (thank goodness) more and more people are using Firefox or other browsers based on the Mozilla/Gecko engine. Whilst Microsoft have helpfully blocked users of browsers other than their own Internet Exploiter from accessing many of their services you shouldn't go the same way.

After all this is the same Microsoft that sells identical copies of Vista in the UK for twice the cost in the United States.

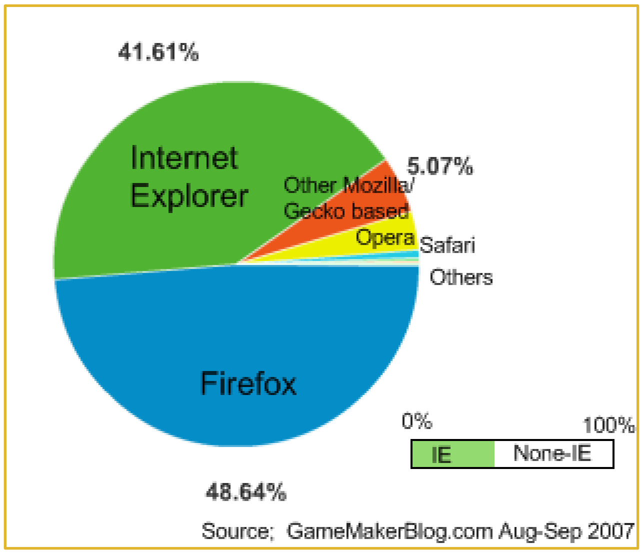


Games

Working with Game Maker\other game program Ask your Game Making questions to the community here...	0	0	No Posts
Games in-progress Any beta\alpha games? Post them here!	0	0	No Posts
Creations Created a game and proud to show it? Put it in here!	1	1	Sun Jul 08, 2007 8:54 pm Mishka
Our Games Featuring our games!	0	0	No Posts
Requests Need sounds or sprites? Request them here! Any WHOLE GAME requests will be closed and removed however... Links are allowed.	0	0	No Posts

10 Things to Avoid On Your Game Maker Website Cont.

This graph on the right indicates the proportion of people you may be neglecting if you foolish decide to choose a design that is incorrectly scripted, or worse still use JavaScript coding to redirect none-IE users away.



6. "Coming soon" pages

If it's not online now do you think I really care? You've taken the time to put up a link to a page with no content and to write those dreaded two words but you can't be bothered to write anything else? Chances are 'soon' will never come anyway.....

7. Your own bad design

If you can't make a decent consistent design for your website – don't. Learn, get someone else to do it or use a template instead. Simple.

8. Clashing colours

Reading white text on black is hard. Reading pink text on gray is harder and reading green text on blue is just plain ridiculous. Why bother writing if no-one is going to be able to read it?

9. Abvns

WTF? Indeed. Abbreviations, text speak - whatever you want to call it. Avoid like the plague it looks unprofessional, tacky and like your website has been written by a five year old.

10. Not reading your content

It happens to all of us. When we are writing we may occasionally repeat a word or use the wrong won. Spelling, grammar and nonsense can be cleared up by proofreading.

Philip Gamble ■
http://gamemakerblog.com

R A N T

Instant Play

While reviewing a game for this issue, I hit the trouble of not having the Firefox extension "Instant Play", by YoYo Games, allowing one to play GM games straight without any "manual downloading"; press the button and play. I hit no trouble with the system, except that Fraps didn't take any screenshots of the game while using Instant Play (although this could be a problem on my side). Basically, I downloaded the extension, restarted Firefox, pushed "play game" and played. All fine.

Veeti Paananen ■

ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT

11 Having broken affiliation banners on your website

This makes your site look tacky and old as images fail to load or link to 404 pages.

The free GameMaker Affiliation Service solves this problem by use of a dynamically updating code which connects your gamemaker site with others.

GameMakerBlog.com/GMA

Pokémon Twilight

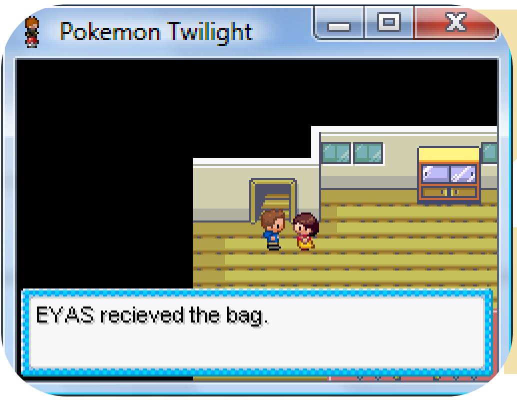
Game Fortress – the development team of Pokémon Twilight, were nice enough to allow me to give an exclusive preview of the game, in addition to a mini-interview and some additional information about the game.

While I consider myself lucky enough to have gotten an early demo of this game in the first place, it should be noted that they were loyal enough to their fans, whom have been waiting months for the new version, and decided to only send me a slightly updated version.

As I told them jokingly, that gives me less to write about, and for the reader less to be excited about, but there's still a lot of great things about this game that'll get you excited – I'm sure!

Pokémon Twilight is – in recent memory – *the* most anticipated Game Maker Work in Progress game, so let hopes it all works out, and from what I'm seeing, it should.

When you run the game, the first thing you could notice would be the all new music. Wow! You hear great tunes and soft music while staying faithful to the original theme and feel of Pokémon games in general.



EYAS recieved the bag.

When beginning the game, as can be seen from the figure above, nothing has changed dramatically. But we'll soon go to deeper aspects of the game and discover all the new additions!

EDITORIAL

Reflect Games: Game in a month

Reflect Games, best known for their online account system, have had their first official competition. The challenge was to create the best arcade game you could in just a month, and the standard was very high! Game Maker superstar FredFrederickson, who ran the competition, has also revealed that he plans to arrange another one in a month to come. The subject is as of yet unknown, but we hope the entries will be of high quality again.

Grego Tyler ■

REVIEW

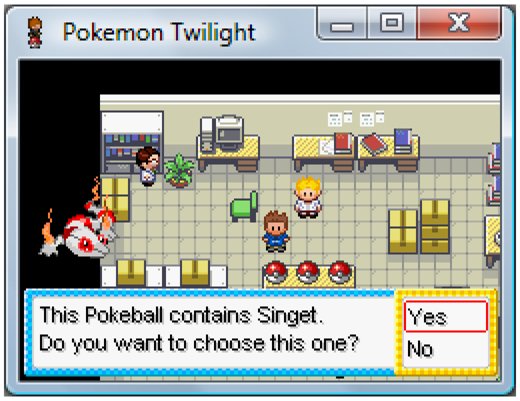
This month's Winner

BenRK was the winner of the first competition, with his game Space 3000. It's a scrolling space shooter where you control a little rocket trying desperately to destroy the oncoming fleet. Unlike many nooby space shooters, this game has graphics with excellent depth and shading, making it look much more appealing than the typical 1945 clones we see so much.

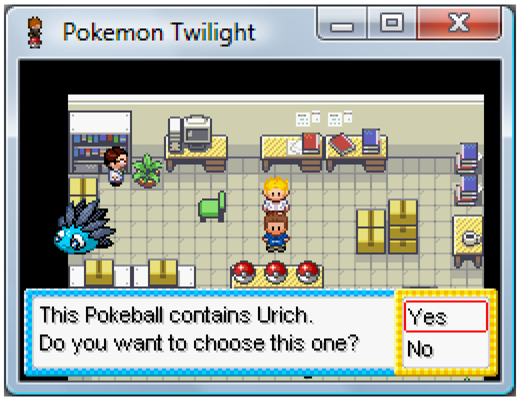
Space 3000 also has added game play features to vastly distinguish it from other shooters. For example, you have to wait for you cannons to recharge before you can fire again, unlike the somewhat more traditional unlimited ammunition. However pick-ups can improve your recovery rate as you go, amongst other things.

Grego Tyler ■

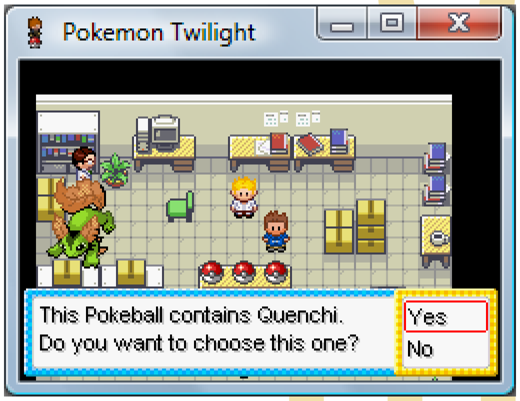
Pokémon Twilight: Exclusive Preview, Cont



This Pokeball contains Singet.
Do you want to choose this one?



This Pokeball contains Urich.
Do you want to choose this one?

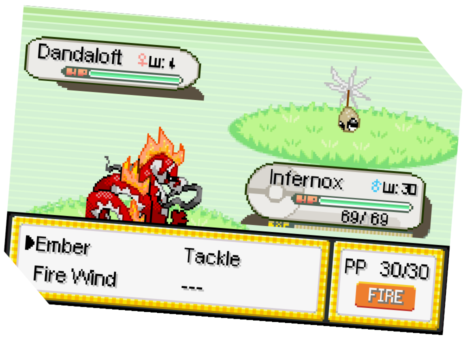


This Pokeball contains Quenchi.
Do you want to choose this one?

As could be seen in the three figures above, the three starters have been updated. Their sprites are really improved and awesome.

Singet, Urich, and Quenchi are the starters for Pokémon Twilight, and as just like all the starters are all-new Pokémon, all of the other Pokémon will be created and sprites without any relation to the classical Pokémon, a move that has been welcomed by the game's supporters: ever for a 'clone game' or a 'fan game', originality never hurts!

When comparing the demo I have with the last public version [in date of release, that is], a very cool thing is that all Pokémon have been re-sprited and improved considerably. The graphics are now more Pokémon-like yet more original as well.



As you can see from the screenshot above, the in-battle graphics have also been improved, but more importantly, the reason I'm showing an in-battle screenshot is because a great improvement has been made to the battle system, something that Game Fortress spoke about for quite some time in the topic: correct damage calculations.

That's right, the game is now using Nintendo's official formulas to calculate damage. This means that all Pokémon need to have base stats and damages are done accordingly.

Everyone seems excited for this and I understand why, but for me, I was shocked that the calculations in the previous version weren't Nintendo's in the first place. Even the Pokémon

↺

R A N T S

GMC Karma (wars)

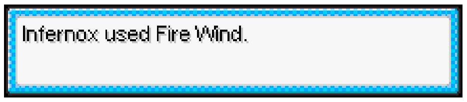
GMC Karma. What a genius idea, especially when it's implemented straight into the forum, easy to use, no hassle! However, the makers of this should have figured something out... There's about less than 50 members in the Game Maker Forum capable of giving constructive karma changes. Basically, whenever you give criticism (or change somebody else's karma), half the time you're facing a drop in your own karma. But instead of whining and nagging, I'll give you my "design document":

1. Karma system only allows positive changes.
2. Positive changes given by "good deeds".
3. Eventually a person's karma drops, as there are no new karma changes (use your own brains to make a variation of this basic idea)

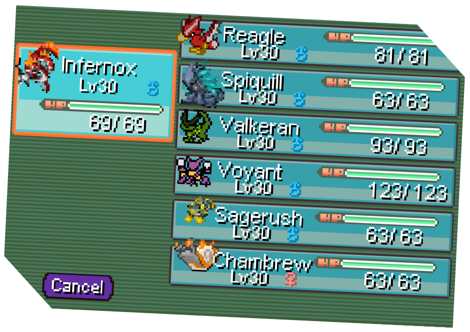
Oh, by the way, it keeps nagging me to "activate" which won't do anything but waste me 10 seconds. Then sometimes it is like "lack of activation" never happened.

Veeti Paananen ■

game I never released had those. Interesting


Infernox used Fire Wind.

Another new addition is, of course, new attacks. The one I'm showing above – Fire Wind – is one of them. It is an attack for "Infernox", which is Pokémon No. 3, the third evolution of the fire starter, Singet.


Infernox Lv30 69/69
Reagle Lv30 81/81
Spiquill Lv30 63/63
Valkeran Lv30 93/93
Voyant Lv30 123/123
Sagerush Lv30 63/63
Chambrew Lv30 63/63
Cancel

I was given a really cool party so that I could try out the new attacks and so on, and so below are some of the information boxes of some of the Pokémon whom I've found to be interesting.

From the new Pokémon shown, we can see some cool new features:

- Double types, like "Fire-Poison", these are all taken into consideration in the new damage calculation.
- Natures, which is a cool feature where the nature of the personality of a Pokémon is mentioned, and it affects the way they act and react.
- A graphically improved interface

Recently Announced Features

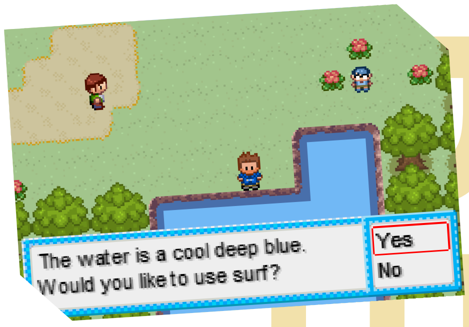
Here are some cool features that have been announced by the Game Fortress team but wasn't included in the demo I received. I've previously talked about my mixed feelings regarding that issue!

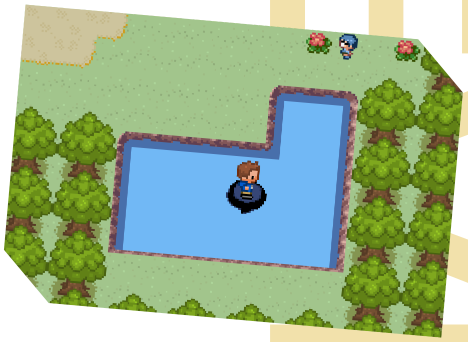
Surf

One of the most basic HMs in all Pokémon games has already been added to the game.

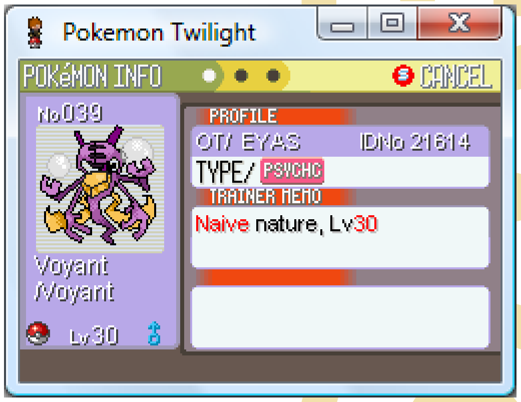
Game Fortress, on a post on the GMC topic for the game – found here, posted cool images of how surf is now, and later posted videos of the surf [as well as glitches created via code injection using cheats].

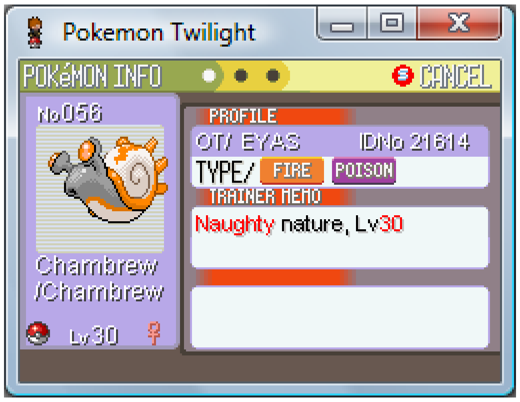
The images are shown below:

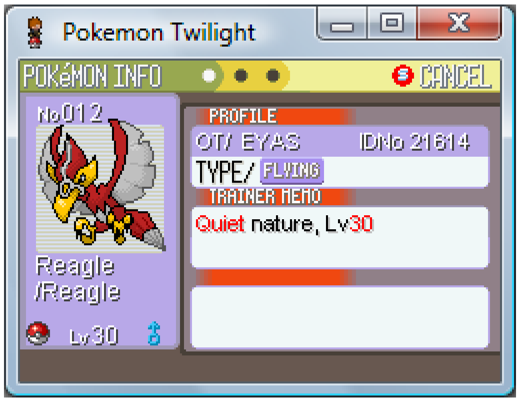

The water is a cool deep blue.
Would you like to use surf?
Yes
No

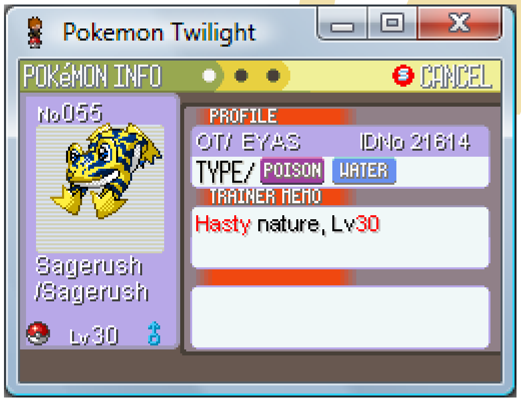


Surf works pretty well, I'm told, and this will give access to many cool parts of the game and the map.

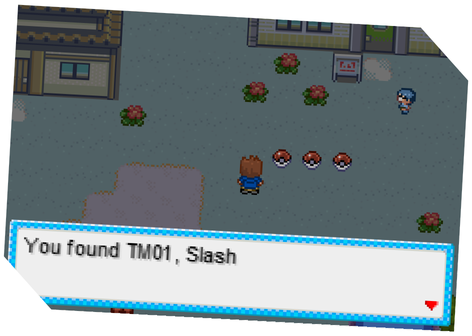

Pokemon Twilight
POKéMON INFO CANCEL
No039
PROFILE
OT/ EYAS IDNo 21614
TYPE/ PSYCHC
TRAINER MEMO
Naive nature, Lv30
Voyant
/Voyant
Lv30


Pokemon Twilight
POKéMON INFO CANCEL
No056
PROFILE
OT/ EYAS IDNo 21614
TYPE/ FIRE POISON
TRAINER MEMO
Naughty nature, Lv30
Chambrew
/Chambrew
Lv30


Pokemon Twilight
POKéMON INFO CANCEL
No012
PROFILE
OT/ EYAS IDNo 21614
TYPE/ FLYING
TRAINER MEMO
Quiet nature, Lv30
Reagle
/Reagle
Lv30


Pokemon Twilight
POKéMON INFO CANCEL
No055
PROFILE
OT/ EYAS IDNo 21614
TYPE/ POISON WATER
TRAINER MEMO
Hasty nature, Lv30
Sagerush
/Sagerush
Lv30

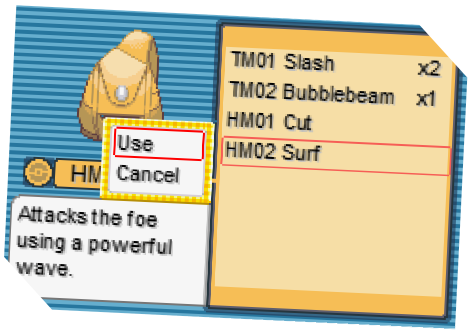
Darkness and Light



While this image was used in the context of referring to the creation of TMs, it also shows a very cool feature of the game: light and darkness, otherwise known as day and night.

The feature depends on the system clock, which on one hand means that we don't need to connect to the internet the whole time, but on the other hand makes cheating easier.

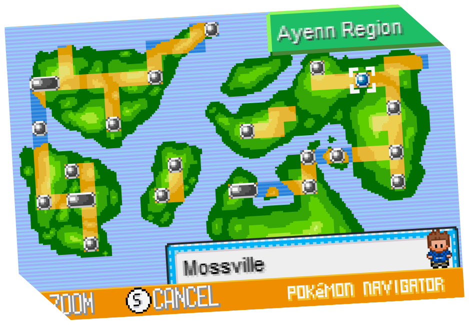
TMs and HMs



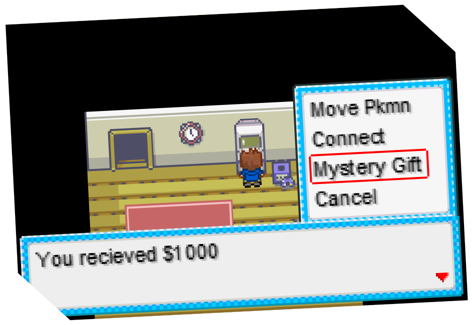
We discussed surf above, but TMs and HMs have – in general – been created and added in, as you can see them in the bag. Pretty cool, I must say.

Finalized Region Map

The map of the entire region has also been finalized. It looks cool, and, well, familiar – don't know about you guys, but that's what I think at least. Anyways, the image is below:



Mystery Gift



Another extremely cool feature that has been added to the game is the mystery gift.

The original plans indicate that one different mystery gift will be offered from Game Fortress. The way someone gets a mystery gift is by going to their computer and choosing "Mystery Gift". The game then connects to the internet and the PT site and downloads information about the gift.

The gift could only be received once a day, and someone who tries to get the mystery gift twice a day gets a message telling him/her that a mystery gift has already been received to this account.

A mystery gift could be anything from cash to objects to attacks to Pokémon.

Glitches

A glitch page could be found here. If you are to report any glitches with the previously released version OR any other versions, you need to check the glitch page first as the error you reported might have been fixed.

Here's a copy of the table:

STATUS	GLITCH
Fixed	Error message appears saying "global.location != location", and the message appears 15+ times
Fixed	Entrance to Glane Cave in the sand outside your house
Fixed	When I read the text for the gamecube and/or other objects, it freezes up
Fixed	I went down the stairs, and when I appeared in the other room, I was facing the wrong way
Fixed	I can surf on land?!? o_O
Finding	Pokemon can get duplicated while being moved in the boxes
Finding	When viewing your parties attacks can sometimes dissapear.
Fixed	Sometimes you get an error message when going into the mart. Apparently encountered when you save after buying an item.
Fixing	Cannot nickname starters.

Interview

I've made an interview with the Game Fortress team. It is less concerned with the game itself and its upcoming features, and more concerned with the way the game is developed, since we believe that is the primary interest for our readership.

How do you plan to take advantage of the online capabilities of the game?

At this stage in the development of Pokémon Twilight, the online aspect of the game could either expand or be completely absent. We haven't come to a final decision on the matter, so it can still go either way depending on how the wind blows. I will say that we would both *like* to see online in PT.

How is development organized? Is the source sent to the developers and updated then sent to the others, or are the various components of the game developed separately?

Usually, the graphics and concepts are developed independently of the engine until they are ready to be implemented, at which point we temporarily move the .gmk file to a different member. Another method we've found to be effective for things like Pokémon stats is to write code without the newest version, where it can then be copy-pasted.

Do you find putting deadlines a good strategy?

Honestly, (and fans who have followed ANY of our games have probably noted this) putting deadlines on our games just makes fans angry when we miss them. ☺

Though the effects are often subtle, it is clear that Pokémon Twilight has some cool GBA-like effects, as well as fading and music -- how important do you feel this is to the game's overall feel?

I think that Pokémon Twilight is the embodiment of all our childhood Pokémon fantasies (which were many), so half the fun has been to recreate a game from the official series as closely as possible to do them justice. I think attention to those details is what really sets a Pokémon game apart. If you are mimicking the Pokémon look and feel, you have to catch the soul as well.

Eyas Sharaiha■

N E W S

Future of Game Maker

I thought, since our main editorial in the introduction talked about the future of Game Maker, as made clear by Sandy Duncan in the YoYo Games GLOG.

First, the Game Maker runner is being rewritten in C++, this allows for games to be ported on to other machines. YoYo are already talking to Microsoft and other companies, and Nintendo is soon going to follow, to negotiate NDS support – and maybe a Wii, who knows?

According to a response Sandy made, Game Maker is planned to remain in DirectX for the time being, and states that this will not mean that games won't work at all in other systems, however obviously, things such as d3d functions will be Windows only.

It is important to make clear that rewriting the runner in C++ does not mean that the syntax and structure of the language will change. To the contrary, it will remain the same, and will include the same looseness as it did before.

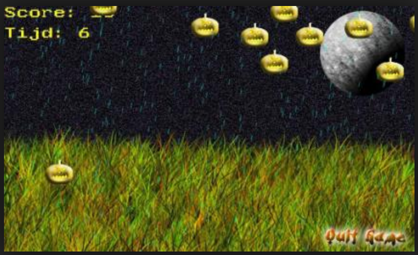
The "maker" itself is being ported to Pascal. This is a relatively easier process, as Game Maker is originally written in Delphi, which is an object oriented variant of Pascal. The reason for porting it to Pascal is to enable it to run on other platforms like the Mac.

Eyas Sharaiha ■

Halloween Night

What they say

This game is a good way to procrastinate from your chores and duties, or just to keep kids entertained. It's a really simple game, but has very little problems.



Description

Halloween Night is a simple 2D minigame. The game opens with a nice, ambient menu. It uses the GM default highscore system.

The graphics consist of Halloween pumpkins, rain, a nice background depicting a full night moon, and grass. A working combination. The objective of the game is to click (shoot?) pumpkins that are "wackily" jumping around the screen, with new ones first falling down, and then bouncing back up from the floor. If a pumpkin flies to the night sky after getting in touch with the floor, the player will lose points.

Clicking the pumpkins is quite hard due to their rapid movement. There are little bugs in the game, which often are to the player's advantage: for example, the Halloween pumpkins get stuck in the corners of the screen very often, making them an easy target for the player.

The game is just a simple minigame, so a full-fledged review obviously can't be done. There are no other special sound effects in the game, with the exception of a Halloweens music clip playing in the background. One problem I noticed mouse clicking lag on the main menu buttons.

Pros and Cons

Halloween Night is a minigame, but quickly becomes un-entertaining.

Pros

- Good basic minigame idea
- Ambience

Cons

- Lack of sound effects
- Dull gameplay after minutes
- No powerups, etc.
- Small bugs

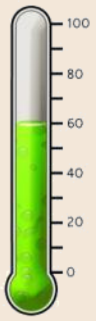
Conclusion

It's a minigame, and is lacking in too many aspects to count, but it's a fun diversion from a stressful day at the office.

Veeti Paananen ■

Ratings (99)

Graphics:	★★★★☆
Sound:	★★★☆☆
Gameplay:	★★★★☆
Storyline:	N/A
Average:	★★★★☆



Stats

Developer:	V'n Gamezz
Version:	1.0
Made with:	Game Maker 6
Filesize:	2.46MB

Download

ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT

Get a free book by writing to MarkUp!

Choose a book related to game development and review it for MarkUp Magazine!

Free Book - Free Shipping
Click for Information

CLICK HERE

39dll

Overview

39ster's (gmc.yoyogames.com/?showuser=234) '39dll' has been around for quite a while. There are some users who do not know how to use it, and some other users know it like the back of their hand. There are many things you can do with this dll (not only the dll, but online functions).

Why not use *Game Maker's* own mplay? Well, 2 reasons. The first is that it is quite a bit slower, and secondly, you need to have the port open on both the server and the client which can be a big mess. -oNL1N3- (gmc.yoyogames.com/?showuser=39135) has created a platform MMORPG example in the open source section, and it is what helped me learn 39dll. I modified the example, and added equipment and hp and stuff, but I got stuck at the monster building part. I also added a login system to it.

Login System

Speaking of login systems, if you need some login help with 39dll, I have created a very simple example, which features a registration system that checks if the user name is in use, and a simple login script. It even comes with the server and the ini file which is used to store all the user data. It is a very simple example and can be modified or 'merged' very easily. Check it out on the GMC sometime: gmc.yoyogames.com/?showtopic=323767.

To start off, you will want to have **2** Game Maker windows open, one for the client and one for the server. Use the various functions to get a connection started. On the server in the step event, you need to check if someone is connecting, and then create an object with a variable with the socket id:

```
stcp = tcpaccept(servertcp,true);
o=instance_create(x,y,obj);
o.tcp=stcp;
```

That accepts the connection and makes an object with the socket id. Then on the object you create you will have all your messages receiving and sending. The client does not need to create another object for the socket id, because you only have 1 connection.

Now, if you have experience with any dll or you have used mplay, you know messages need id's to work properly. For me, when I send a message I do something like this:

```
clearbuffer(); -important. Erases
all the data on the buffer.
Writebyte(); -the id of the message
Writestring(); -writing data
writebyet(); -more data; you can
have lots of these.
sendmessage(socket);
```

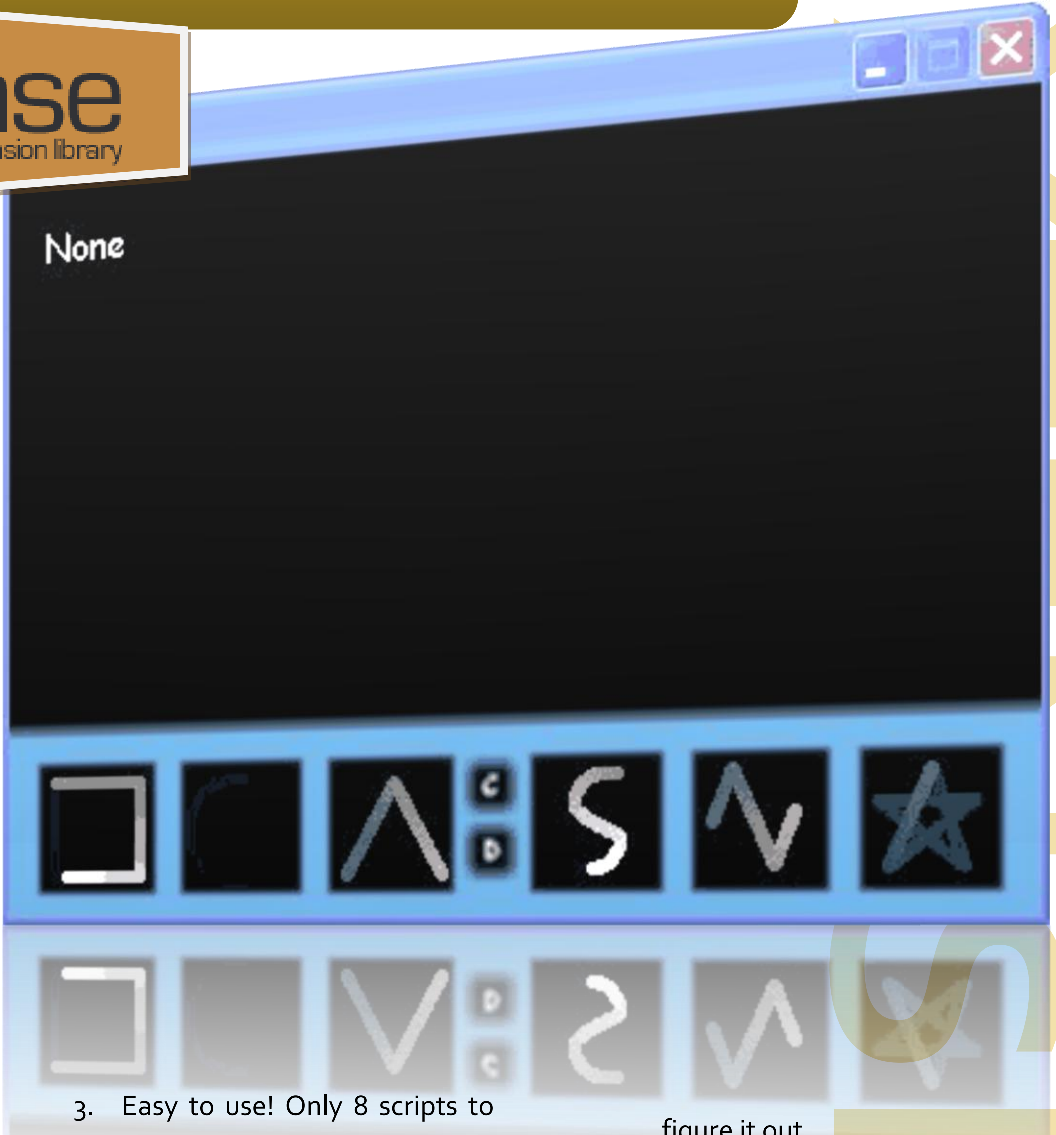
Now when you are receiving the data, you need to read the data in the right order. So after a message is received, `the message id = readbyte()`. That will make the message id the first data we wrote (the message id) and take it off the buffer. Then you should have a switch statement to execute different things on different message ids. And you must take the data off the buffer in the same order you put it on. This is something a lot of people forget to do. Also, make sure you are using the right function, for example I have seen `writebyte(“hello”)`. Why is this invalid? Well, a byte is a Boolean value (a number), and we have written a string. See the problem? So after all this you can experiment. Good luck on your project.

After you get the idea of message sending, you can start to make an MMORPG right? Wrong. Message sending is the easiest part. You still need to program the player adding and synchronization, and all the other things which are very difficult. I suggest you start off with a simple 2-player game and grow from that.

Revel (Brad) ■

Extension of the Month

Powered By: GMbase The user created extension library



Advanced Gesture Recognition System

Ever played Black and White? If so, you must remember the innovative spell system, which required you to trace a spell's symbol to cast it. My gesture system allows just that. With just a few lines of code, you can add shape recognition to your games. It will take you no more than a few minutes to master the system, which contains only 8 scripts, and 3 of those are optional extra features. The help file also has an easy to follow step-by-step tutorial that will guide you through the process of recognizing a shape.

Features:

1. Automatically scales your shapes. If you define a square, it doesn't matter if the user draws a huge square or a tiny one, the system will recognize it.
2. You can make your own shapes! From Spirals to Stars, Squares to Letters, anything you want.
3. Easy to use! Only 8 scripts to learn, but very powerful!
4. Easy to use shape creator allows you to skip the time-consuming process of hard coding your shapes.
5. The tutorial in the help file will walk you through creating a simple shape and recognizing it. Someone with little or no code experience could easily figure it out.

The download includes the gesture extension (.gex), and the Demo/Shape Creator (GM7 registered). The help file is packaged in the extension.

gmbase.cubedwater.com/?page=extension&id=122

Kyle_Solo■

Script of the Month

Powered By: GMLscripts.com

We really wanted to get a Spooky Script for the MarkUp Issue 8 Spooky Special, and to your surprise: we did! The script is called `draw_sprite_wave`, which draws wavy scripts – an incredible effect, incredible!

Draw Sprite Wave

The regular 'draw sprite wave' script also applies for the Lite version of Game Maker. Another script, `draw_sprite_wave_ext`, exists to take advantage of the amazing features of the registered version of Game Maker.

```
/*
**  Usage:
draw_sprite_wave(sprite,subimage,x,y,axis,wavelength,a
mplitude,phase)
**  Arguments:
**      sprite          sprite index
**      subimage        sprite subimage
**      x,y             position in room to draw
sprite
**      axis            0 = horizontal wave, 1 =
vertical wave
**      wavelength      length of wave in pixels,
crest to crest
**      amplitude       half the height of wave in
pixels, crest to trough
**      phase           wave position offset, vary to
animate wave
**  Returns:
**      nothing
**  Notes:
**      Draws a sprite with wave-like distortion.
**      The fractional part of the phase argument
controls the cycle
**      of the wave, a full cycle covers the [0..1]
interval.
**  GMLscripts.com
*/
```

GM6 Example Here

```
{
    var sprite, image, posx, posy, axis, wavelength,
amplitude, phase;
    sprite     = argument0;
    image      = argument1;
    posx       = argument2;
    posy       = argument3;
    axis       = argument4;
    wavelength = argument5;
    amplitude  = argument6;
    phase      = argument7;

    var width,height,xoff,yoff,size,i,sx,sy;
    width  = sprite_get_width(sprite);
    height = sprite_get_height(sprite);
    xoff   = sprite_get_xoffset(sprite);
    yoff   = sprite_get_yoffset(sprite);
    if (axis) size = height else size = width;

    for (i=0; i<size; i+=1) {
        shift =
amplitude*sin(2*pi*((i/wavelength)+phase));
        if (axis) {
            sx = shift-xoff+posx;
            sy = i-yoff+posy;

draw_sprite_part(sprite,image,0,i,width,1,sx,sy);
        }else{
            sx = i-xoff+posx;
            sy = shift-yoff+posy;

draw_sprite_part(sprite,image,i,0,1,height,sx,sy);
        }
    }
}
```

Creator

Special thanks to xot for creating this special script on request for MarkUp's Spooky Special!

Draw Sprite Wave EXT

This script has more arguments, such as x-scale, y-scale, blending, and alpha. The script is on the following page.

Script of the Month

```
/*
**  Usage:
**      draw_sprite_wave_ext(sprite,subimage,x,y,axis,wavelength,amplitude,
**                           phase,xscale,yscale,blend,alpha)
**  Arguments:
**      sprite          sprite index
**      subimage        sprite subimage
**      x,y             position in room to draw sprite
**      axis            0 = horizontal wave, 1 = vertical wave
**      wavelength      length of wave in pixels, crest to crest
**      amplitude       half the height of wave in pixels, crest to trough
**      phase           wave position offset, vary to animate wave
**      xscale,yscale   scaling of sprite along x and y axes (optional)
**      blend           color to blend with sprite (optional)
**      alpha           alpha used to draw sprite (optional)
**  Returns:
**      nothing
**  Notes:
**      Draws a sprite with wave-like distortion. If scaling
**      is used, the wavelength of the wave will also be scaled.
**      The fractional part of the phase argument controls the cycle
**      of the wave, a full cycle covers the [0..1] interval.
**  GMLscripts.com
*/
{
    var sprite,image,posx,posy,axis,wavelength,amplitude,phase,xscale,yscale,blend,alpha;
    sprite     = argument0;
    image      = argument1;
    posx       = argument2;
    posy       = argument3;
    axis       = argument4;
    wavelength = argument5;
    amplitude  = argument6;
    phase      = argument7;
    xscale     = argument8;
    yscale     = argument9;
    blend      = argument10;
    alpha      = argument11;

    var width,height,xoff,yoff,size,i,sx,sy;
    width  = sprite_get_width(sprite);
    height = sprite_get_height(sprite);
    xoff   = sprite_get_xoffset(sprite);
    yoff   = sprite_get_yoffset(sprite);
    if (axis) size = height else size = width;
    for (i=0; i<size; i+=1) {
        shift = amplitude*sin(2*pi*((i/wavelength)+phase));
        if (axis) {
            sx = xscale*(shift-xoff)+posx;
            sy = yscale*(i-yoff)+posy;
            draw_sprite_part_ext(sprite,image,0,i,width,1,sx,sy,xscale,yscale,blend,alpha);
        }else{
            sx = xscale*(i-xoff)+posx;
            sy = yscale*(shift-yoff)+posy;
            draw_sprite_part_ext(sprite,image,i,0,1,height,sx,sy,xscale,yscale,blend,alpha);
        }
    }
}
```

Eyas Sharaiha■

Blinky's Scary School

What they say

There is no GMC topic for this game. One comment has been sent at YoYo:

I remember the Blinky series being popular, but never actually played any of them until now. I was more into "Sorcery +", "Head over heels" and the like. Seems to be nicely done though. Hope to see more remakes in the future.

Description

Blinky's is a remake of an old 8-bit Atari game by Eutechnyx. Blinky is a ghost, and is in a mansion (and I really couldn't figure anything else out).

The graphics have a very retro feeling in them, which fits the game. They are repetitive along the game, but are well done. Blinky's movement is exceptionally smooth. However, the game does a very poor job on first presentation. I had no idea what was going on or what to do; SPACE picks up items (what I don't have a slightest idea of their purpose) and you move with the arrow keys. The game's a good platformer, which could be much better with some introduction to the player. The game is also lacking any sound at all, which is a big minus. The AI is basically monsters walking into two directions or flying. The "damage" system is very well done: it does not make the rookie mistake of draining HP continually as long as a monster is touching the player, but instead lowers the HP and disallows any change from the same monster for a small while.

Pros and Cons

Blinky is an impressive example of almost professional-looking visual presentation, but due to the lack of sound and any help at all, it fails to impress me completely.

Ratings (99)

Graphics:	★★★☆☆
Sound:	★★☆☆☆
Gameplay:	★★★★☆
Storyline:	★★★☆☆
Average:	★★★☆☆

Stats

Developer:	Crossley
Version:	1.0
Made with:	Game Maker 6
Filesize:	874 KB

Download

Pros

- Exceptionally good engine
- Great graphics
- Almost professional-looking

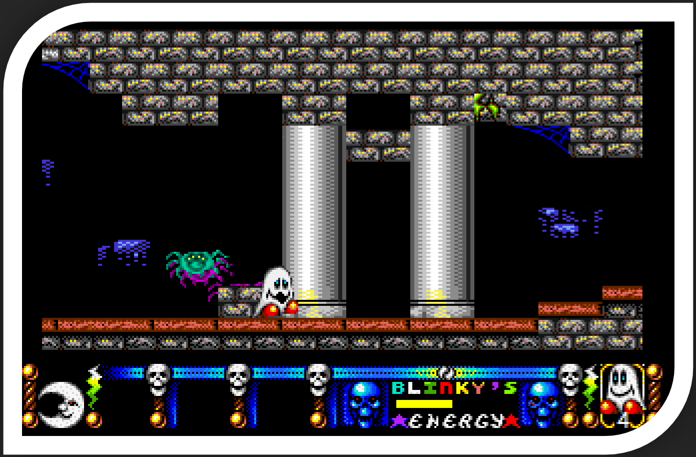
Cons

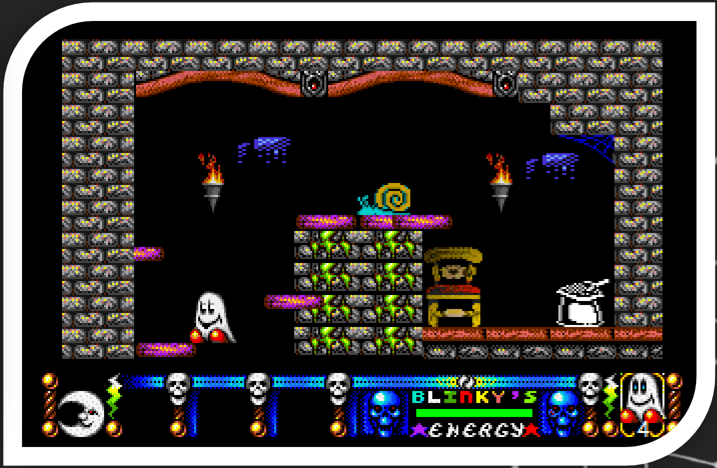
- Lack of sound
- No help to the player

Conclusion

Good idea, bad implementation.

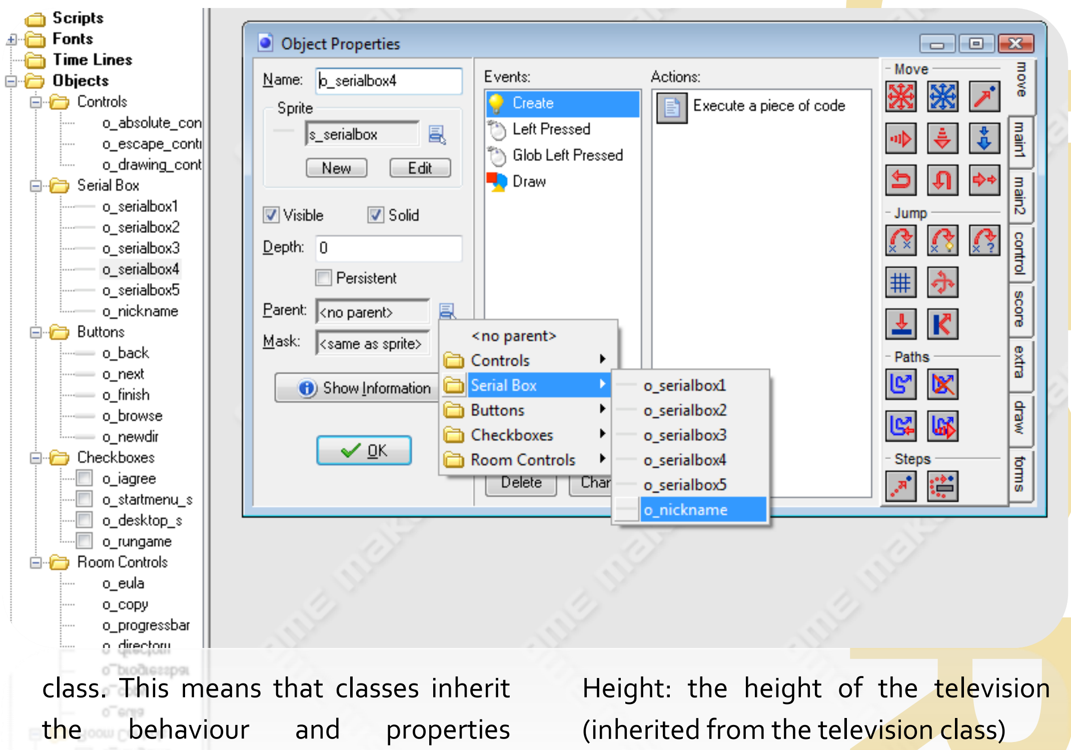
Veeti Paananen





Object Oriented Programming

Objects, Parents, Inheritance, and Instances



The key components in a Game Maker game are objects. Without them, a game doesn't do a thing. When creating large games, the organization of all the objects might get complicated. If there are many objects with almost the same behaviour (events), this could get quite annoying e.g. you need to copy and paste all the code from one object to the other. Luckily, Game Maker allows you to create some sort of hierarchical object structure in which objects can have children and parents and they can inherit events. This is GM's implementation of a more general programming technique that is called object-oriented programming.

Object-oriented programming (OOP)

In object-oriented programming, we use the concept of an "object". An object is an instantiation of a class. What is a class? Simply said, a class is a structure with its own local variables and its own functions. These functions are called "methods" of the class. Very important to mention is that not all other objects have access to these variables and functions. This is called information hiding.

A class can also have derived classes or can itself be derived from another class. This means that classes inherit the behaviour and properties (variables) of the classes they are derived from.

We'll look at an example of how all this can be compared to "objects" in the real world. Let's say we have a television, an LCD display television to be precise. The LCD display television can be seen as a subclass of the television class. Now let's think of some properties and methods the television could have:

Properties

Length: the length of the television (inherited from the television class)

Width: the width of the television (inherited from the television class)

Height: the height of the television (inherited from the television class)

Horizontal resolution: the horizontal resolution of the tv

Vertical resolution: the vertical resolution of the tv

Methods

On: turns on the television (inherited from the television class)

Off: turns off the television (inherited from the television class)

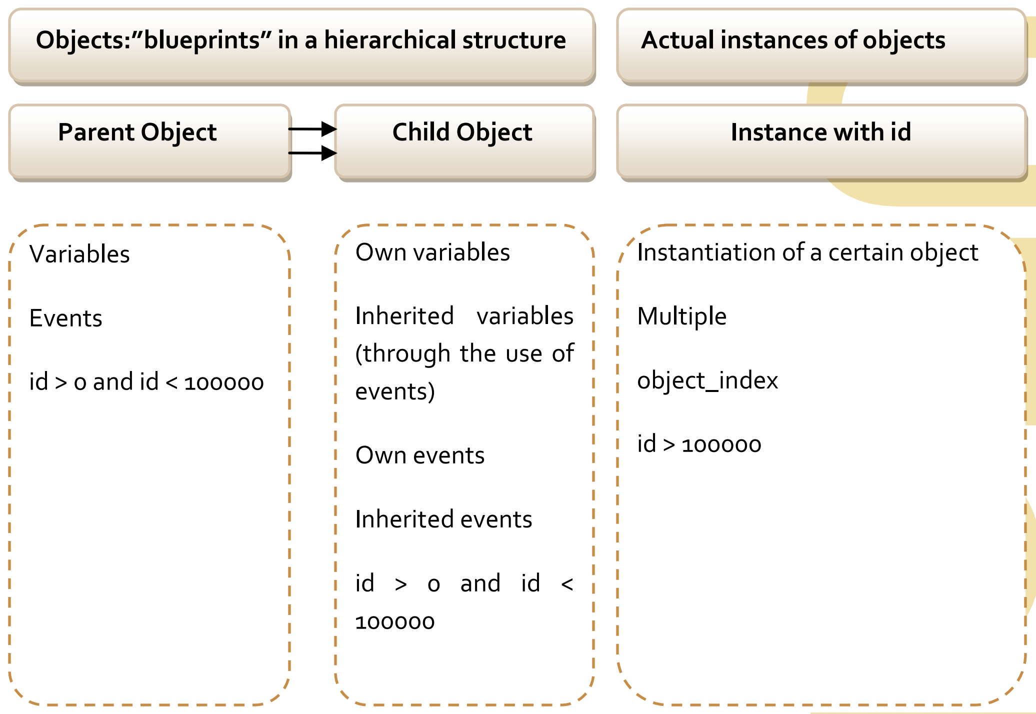
Display: show something on the display (this function is called by the function "On")

These are possible properties and methods that a television could have.

When the user pushes the button to turn on the television, the function On is called. This function calls the method Display. But the user cannot directly access the Display method. So the Display method is "private" in some way. It can only be called by the television itself.

It works the same way with classes in a programming language.

There are some advantages to using an object-oriented approach. A game or program can be split into smaller parts. Each object does what it is supposed to do and conflicts between objects are not likely to occur. The object is independent of other objects (there might be some exceptions, though). By creating a logical class structure, you can work at different "levels". Also, all functions and data required to perform a certain task are now packed together in one handy unit: the class.



GM and object-oriented programming

You could say that GM is very object-oriented. That's pretty obvious: a game made in Game Maker will not do a thing unless you have put an object in a room. The previous sentence is actually wrong. In GM, objects are the classes and instances are the objects (as seen in the previous chapter). Except for that, there is not a big difference between GM instances and objects and objects and classes in other programming languages. Just make sure to remember that in Game Maker:

An object = a class

An instance = an object

And what about the other concepts of OOP, are they present in GM as well?

Objects in Game Maker can also inherit the behaviour and variables of a parent object. But this can only be done through events. An example: an object should have the same sprite as its parent object. Then you could set the sprite for the parent object in the object editor and set that object as the parent for the other object. If you now place an instance of the child object in the room and run the game, you'll see that the object doesn't have a sprite. Now if you would place this piece of code in the create event of the parent object (assume that the sprite is called `spr_character`):

```
sprite_index = spr_character;
```

Then you'll see that the child object now has the same sprite as the parent object. The explanation is simple: objects only inherit events from other objects and not variables. But if you declare a variable in an event of the parent object then that variable will also exist for the child object.

Information hiding is something that's not supported in GM. There is no way to indicate that certain properties of an object should be private, protected or public. In GM, all object properties are available to other objects. Also, objects do not have methods. They can only execute scripts that are available to all objects. For scripts, there is a very simple way to mimic information hiding. Let's say you have a script that can only be executed by `obj_object`. Then you can add this line of code to the script:

```
if (object_index = obj_object)
{
        //script code here
}
```

It's a very simple and effective way to "hide" the script from other objects.

A very important thing about objects and instances in Game Maker is their id. Game Maker uses some arrays to store all the object and instance id's. These arrays can easily be accessed by using some of GM's built-in functions. First, we'll look at a schematic representation of objects and instances in GM.

The scheme shows the objects at the left and the actual instances of them at the right. For simplicity, only one parent and child object are shown, but you can of course create a complete object hierarchy if you want. In the scheme, the objects are called blueprints. The reason for this is simple: objects do never exist in a game. Instances of objects are created and those instances exist.

The parent object has its own local variables and its own events. Object numbering starts at 0. Very important to note here is that the id of any object will most likely be smaller than 100000. The use of this will be explained when we get to instances.

The child object also has its own local variables and its own events. It can also have "inherited variables", but of course this can only be achieved through the use of events. It also has its own events and also inherited events from the parent object. Something to remember is that when you add code to an event of the child object and the same event exists in the parent object, the event from the child object will be executed and not the event of the parent object. If you want to execute the inherited event as well, you can use the function `event_inherited()`, which executes the corresponding event of the parent object.

An instance is, as mentioned before, the instantiation of a certain object. Multiple instances of the same object can exist in a room. If you want to know which object the instance is an instance from, you can use the local variable `object_index`, which contains the object's id. The id of an instance is always higher than 100000. And now it's time to explain the wonderful `with` construction.

Quoted from the Game Maker Manual:

Its global form is:

```
with (<expression>) <statement>
```

`<Expression>` indicates one or more instances. For this you can use an instance id, the name of an object (to indicate all instances of this object) or one of the special objects (all, self, other, noone). `<Statement>` is now executed for each of the indicated instances, as if that instance is the current (self) instance.

As you can read in the underlined line of text, almost every ID can be used in a "`with`" construction. And now we can see the use of the fact that instance id's are always larger than 100000. Whenever a number larger than 100000 is used as the expression, it means that it is the id of a single instance of a certain object. When it is smaller than 100000 and larger than 0, it means that it is the id of an object. When the "`with`" construction is used like that, the statement is executed for each instance of the object. With this, you can possibly save a whole for loop.

All, self, other and noone are in fact constants with a negative value. These can also come in very handy sometimes.

If you combine a logical hierarchical object structure with the use of a with construction, you'll probably save yourself a lot of work. And you'll also have a better view on the structure of your game that way.

To conclude this chapter, we'll have a look at the variables, arrays and functions you can use to find instances, objects, the number of instances of a certain object and parents of objects (all explanations, except pieces of code are quoted from the Game Maker help file).

First of all, you might want to know the id of an instance or the id of the object it is an instance of. For this, you can use the following two variables:

`object_index*` Index of the object this is an instance of. This variable cannot be changed.
`id*` The unique identifier for the instance (>= 100000). (Note that when defining rooms the id of the instance under the mouse is always indicated.)

Normally, instance numbering starts at 100001 (100000 doesn't seem to be included). When you add an instance of an object to the room, 1 is added to the counter each time. But when you remove an instance from the room, nothing is subtracted from the counter. That means that you'll start skipping instances. Luckily, Game Maker has 2 variables to get all the id's of the instances that currently exist in the room:

`instance_count*`: Number of instances that currently exist in the room.
`instance_id[0..n-1]*`: The id of the particular instance. Here n is the number of instance.

Note that the array index n is the number of the instance and not its id.

You might want to create and destroy instances yourself through the use of gml. Then you could use these pieces of code:

`instance_create(x,y,obj)` Creates an instance of obj at position (x,y). The function returns the id of the new instance.
`instance_destroy()` Destroys the current instance.

Note that the `instance_create()` function returns the id of the new instance. This is very useful if you want to keep track of all instances you create yourself.

If you want to destroy all instances of a certain object:

```
with(object_index)
{instance_destroy()}; //destroys
all instances with the same
object_index as the instance that
executes the piece of code
```

You might also need the value of local variables from other instances. For that, you can use the instance id (as a variable) followed by a dot and the variable name:

```
value = inst_id.variable; //finds
the value of the local variable
"variable" of the instance with
id inst_id
```

Many useful functions related to manage objects can be found in the GM help file:

The Game Maker Language (GML) ➔ Changing Resources ➔ Objects

Internet link:
gmlscripts.com/gml/changing_object_resources

And various functions to get information about objects can be found here:

The Game Maker Language (GML) ➔ Resources ➔ Objects

Internet link:
gmlscripts.com/gml/object_resources

And that's about all there is to tell about GM's support for object-oriented programming.

Conclusion

You can create object hierarchies in Game Maker by using GM's features of object-oriented programming. GM has many functions to get object and instance id's and functions (and arrays) to retrieve information (parents, object count) about objects. When you use all those fully to your advantage, it will be easier to manage the objects in your game.

Bart Teunis■

Mr. Pratt's Haunted Mansion 2

What they say

"Grat job". I have to agree to this poorly spelled sentence from YoYo. Mr. Pratt's is certainly a very good game, and much better than many others, hence the Staff's Pick label in YoYo.

Description

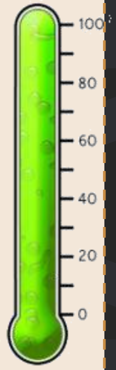
Mr. Pratt's Haunted Mansion 2 is a side-scrolling shooter (although not very fast-paced). You play as , a rich mansion owner. Zombies and other monsters have invaded your mansion, and it's up to you to blast them all to death.

The graphics are very impressive and fit the dark atmosphere of the game. The game also includes a very high quality music library, although lacking in quantity. The music is very professional-sounding, although the sound effects are rather lacking. The enemy AI is simple, not that it needs to be any better. I really liked this game during testing it out, and it kept me addicted. It's also quite hard to play, but still simple. The controls are very simple to use, and do not need any special learning. You can hang from cliffs, etc. which is a plus, yet a simple platformer feature.

Mr. Pratt's contains 20 levels of the action, which is fun to play through. One very big plus is an included level editor, which allows one to make complex custom levels to the game, yet very easily. Presentation is very professional, and Mr. Pratt's could easily be a cheap minigame one could buy.

Ratings (99)

Graphics:	★★★★★
Sound:	★★★★★
Gameplay:	★★★★☆
Storyline:	N/A
Design:	★★★★★
Average:	★★★★★



Stats

Developer:	Bouncing Fox Productions
Version:	1.0
Made with:	Game Maker 6
Filesize:	11 MB

Download

Pros and Cons

Mr. Pratt's is a near-professional level 2D platform game.

Pros

- Very good engine
- Professional graphics
- Level editor
- Keeps you playing

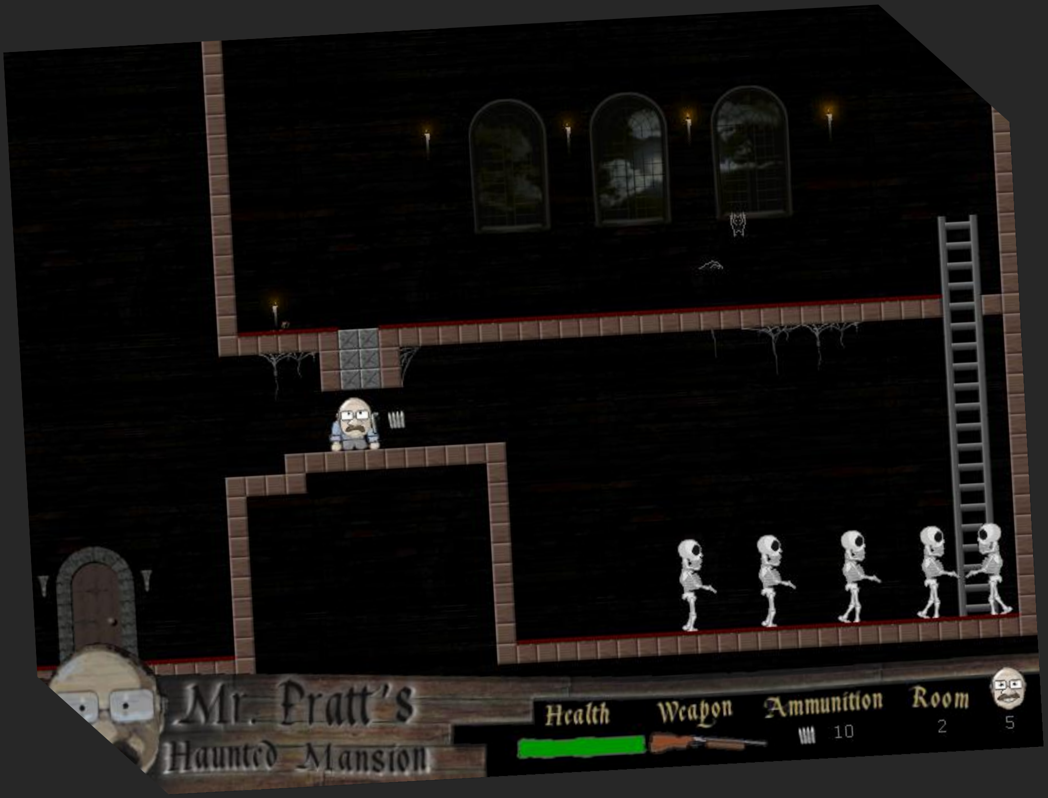
Cons

- Poor sound effects

Conclusion

This is a game anybody should check out.

Veeti Paananen ■



God of Rock

Development Journal

September 15th, 2007

I recently decided to add Sustained notes into the game. The first thing that came to my head was to do use `draw_sprite_stretched` and then have two objects, `S_Note_Start` and `S_Note_End` and stretch the sprite in between the two objects.

I found that that didn't work so I used `draw_rectangle_color` instead. But, after I found that out, I encountered another problem, since I used `instance_nearest`, if the start object detected a end object that was behind the one in front of it, it would draw the rectangle backwards to the end object behind the one in front.

To make this easier to understand, the system works with two basic objects, the start object and the end object. The start object would go in front of the end object, and the start object would use `instance_nearest` to draw a rectangle to the end object that is USUALLY in front of it but, since that is not always the case I had to find another way to do that.

So I tried an alternative method, this method made the game's FPS go all the way down to 5/30. Obviously, that method is not going to work. After that I discovered one more method, this method involved a combination of Arrays and For Loops. This method worked perfectly with no FPS drops.

Then all of the sudden the Pause screen stopped working correctly. I had to totally re-code the pause screen!

The next conflict I had was a problem with the `var` function in Game Maker, what happened was, the game would keep saying "unknown variable" and the game would get stuck in a infinite loop and freeze.

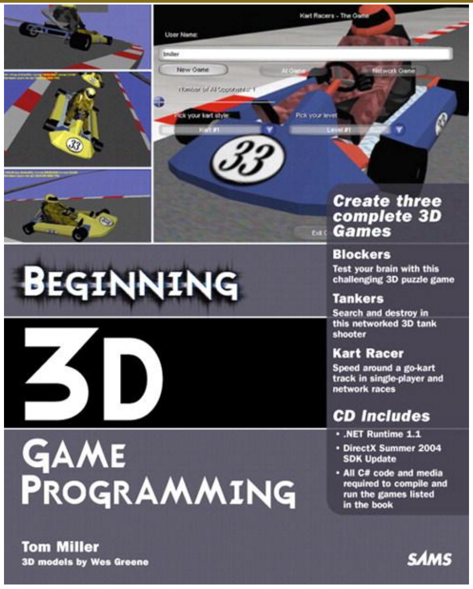
But, someone at the GMC helped me fix that problem; now, it functions with no bugs whatsoever. But, shortly after I encountered another major problem, the game was not correctly writing the data files. And the only way to remedy this was to recode the entire file handing system. So that would delay the release of version 1.06.

So what I did was add a temporary fix by making the game rewrite the data and correct the incorrect values but, I think it would be better for it to write it correctly from the start. And those are some of the problems I faced or are facing in God of Rock, for the month of September.
http://gmc.yoyogames.com/?showtopic=309371

XD005 ■

Beginning 3D Game Programming

Philip got this book for free. Learn how you could do so too. Click here.


Create three complete 3D Games
Blockers
Test your brain with this challenging 3D puzzle game
Tankers
Search and destroy in this networked 3D tank shooter
Kart Racer
Speed around a go-kart track in single-player and network races
CD Includes
• .NET Runtime 1.1
• DirectX Summer 2004 SDK Update
• All C# code and media required to compile and run the games listed in the book
BEGINNING 3D GAME PROGRAMMING
Tom Miller
3D models by Wes Greene
SAMS

I've never made a 3D game before so I was quite excited to be offered the chance to review "Beginning 3D Game Programming". My past programming experience stems from knowledge of Delphi which I use on my Computing course and limited knowledge of Game Maker's inbuilt scripting language GML.

I am far from the most competent programmer but with the increasing number of people creating 3D games in Game Maker and the game development software's 3D possibilities extending I figured this would be a worthy read.

The cover claims that you can create three complete 3D games after reading the 418 page book and using the material included on the accompanying CD-ROM (more on this later). "Blockers" – a challenging 3D puzzle game, "Tankers" – a networked 3D tank shooter and "Kart Racer" where you can race in either single player or networked races.

I think it's best to say now that this book does not use Game Maker to create 3D games but instead makes use of Microsoft's .NET framework with code in the book written in C#.

If you have only ever used Game Maker don't be afraid by this. New language, new rules. To get the most out of this book you really need to start learning C# as soon as possible. Yes, there are a lot of new words flying about but they are all entirely feasible if you have knowledge of programming – there is an introduction to the .NET framework before the main book starts. With author Tom Miller being the designer and development leader of Microsoft's Managed DirectX API he certainly seems like a well qualified person to have written this book.

The title of the book "Beginning 3D game programming" seemed a little daunting in itself, after all programming in 2D can be very challenging indeed which leaves 3D programming sounding even more complex – an extra dimension opens up so many more possibilities, and of course potential problems. I certainly didn't dive into the book with unrealistic perceptions that I would be able to create the Karting game displayed on the cover after just a couple of hours work.

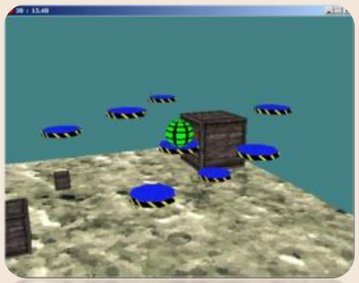
However as someone with no prior knowledge of either C# or the .NET framework I was disappointed to find that the book did not appear to be aimed at my definition of beginner. At first I was able to understand what was going on, but as I read on things became more and more confusing – the book advances very quickly.

Quick Reviews

3D Collision DLL

The 3D collision DLL precisely checks for collisions between 3D objects in Game Maker. A tip from the author: "*it works best with low-poly models, so if you have a mesh with many polygons, create a different one with less polys to use as its collision model.*"

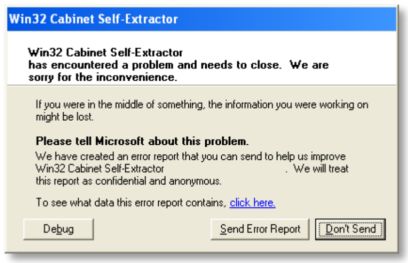
The DLL supports 3 Model formats, either GML Code, the Separated Format, or the Game Maker model format itself.



Get it now!

gmc.yoyogames.com/showtopic=329495

Someone without prior knowledge of programming would be completely lost after just a couple of pages, and from what I can see you would have to be a fairly competent C# programmer before you could make full use of the book and understand all of the code.


Win32 Cabinet Self-Extractor

Win32 Cabinet Self-Extractor has encountered a problem and needs to close. We are sorry for the inconvenience.

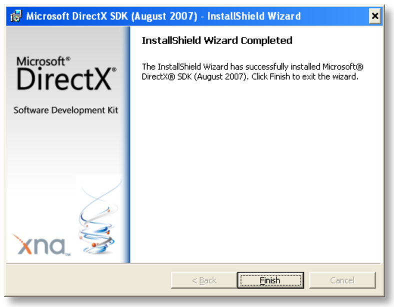
If you were in the middle of something, the information you were working on might be lost.

Please tell Microsoft about this problem.
We have created an error report that you can send to help us improve Win32 Cabinet Self-Extractor . We will treat this report as confidential and anonymous.

To see what data this error report contains, click here.

Not a good start to the installation process.

A major disappointment came when after a brief read of the first few chapters I decided to use the included CD to work through the examples described in the book. The CD contains the graphical resources used in the games as well as some of the utilities needed such as the DirectX Software Development Kit (SDK). There was no mention of what each thing on the CD was, and no installation help whatsoever which left me rather flummoxed especially as I was unable to install the DirectX SDK even after several attempts. A quick Google search revealed that I was not alone with this problem which had been reported on an online discussion as early as 2005. I am disappointed that in two years it appears there have been few, if any, modifications to the CD or book (my copy was printed in 2007) despite the fact that many of these shortcomings are widely reported on the Internet. (You can get around this by downloading the August 2007 version of the DirectX SDK from the Microsoft website).


Microsoft DirectX SDK (August 2007) - InstallShield Wizard

InstallShield Wizard Completed

The InstallShield Wizard has successfully installed Microsoft® DirectX® SDK (August 2007). Click Finish to exit the wizard.

The DirectX SDK is finally installed, but wait – more software is still needed.

The biggest problem was yet to come: Microsoft Visual Studio .NET 2003 (a commercial product) is required to follow the book, and this is not included on the CD. This is not made at all clear in the book, which seems to assume that you already own all the software necessary to make your game without having to be told about it prior to starting.

There are sections explaining the types of tools you will need to create a good 3D game such as 3D modelling software and, probably more importantly, why you need them – I was pleasantly surprised at the level of detail Tom Miller goes into when introducing models and textures. I had thought these would only be briefly touched on - after all this book is about programming rather than graphics. After making "Blockers" object manipulation is introduced however this, like the book in general, gets complex quickly. There are also chapters on 3D maths, basic 3D particle systems and user interfaces.

Other the other hand if you have a good level of experience and skill using C# and are happily creating 2D games and want to take your first steps into 3D programming then this book is probably right for you. Being familiar with the language used in the book will give you a huge advantage over anyone who is not, and you should be able to get going very quickly provided you have all the required software. As ↻

INI Data Structure Extension

This Extension Package provides a pleasant replacement to the INI file handling mechanism in Game Maker, by using Data Structures to achieve that.

The package also includes much more features than the native INI mechanism does, and supposedly takes advantage of the speed of data structures to improve the game's performance.

The extension can open and edit multiple INI files at the same time. It works with INI files outside the working directory, and maintains INI structures without needing corresponding files. Furthermore, it contains functions to explore INI structures step by step and can easily copy data. It supports custom INI syntax, and does much more.

Get it now!
gmc.yoyogames.com/?showtopic=289009

is obviously the case the more knowledge you have prior to reading the easier you will find it.

I feel that the book may have tried to cover too many areas too quickly, trying to appeal to beginners whilst working towards the development of three complete 3D games (two of them networked) in just 400 pages. In truth the book would best be renamed "Making your first 3D game in C#", as this is essentially what it will teach you to do.

Cons

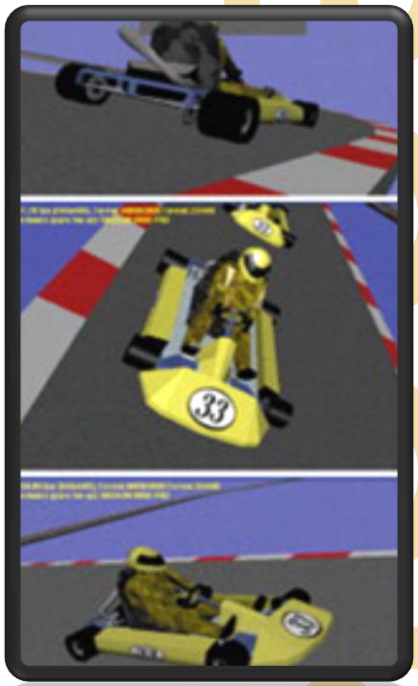
Some issues are skipped over very quickly, for example details of the accompanying CD are extremely limited and there is no installation help. Information about the software required to follow the tutorials is poor, and assumptions are made that you own everything required. However this isn't just true for the CD, there are several gaps in the book as you work through with things missed out and a bit of guess work required when setting things up. The book becomes very complex very quickly (for example the chapter on 3D maths) and you will probably struggle to understand some of the issues being discussed if you have not had prior knowledge of C# to at least an intermediate level. Much of the book is just code which, although vital, may baffle true beginners as things progress so quickly towards the finished games. Considering Tom Miller's position at Microsoft I was disappointed that he has not managed to keep his book up to date with his own SDK.

Pros

Good introduction to some aspects of 3D programming, for example modelling. All the code you need to complete the book and the graphics are included on the CD which will save you time and frustration. The index is very useful, one of the best organised I have ever seen in a book about programming enabling the book to be used frequently as a resource rather than just a series of tutorials. The chapters are also organised in a logical manner, introducing new issues at the same time as the three games develop. If you are a competent C# programmer who can create full 2D games and has all the necessary software installed and can skip over the gaps in the book you should be able to make the games with few problems.

Conclusion

Overall I thought this book was a bit of a letdown, it promised so much, had a good author and led you to making three perfectly good games. If you know what you are doing and keep track of how your games are progressing this will be a good book for you; however instructions I would consider vital are skipped out making this book less user friendly than I would have liked.



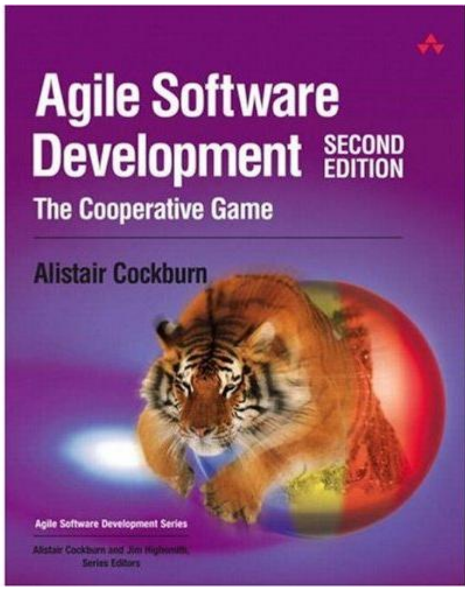
Details

Title
Beginning 3D Game Programming
Author
Tom Miller
Publisher
SAMS
ISBN
0672326612
Year
2004
Pages
418
CD-ROM Included

Philip Gamble ■

Agile Software Development

The Cooperative Game


Agile Software Development
SECOND EDITION
The Cooperative Game
Alistair Cockburn
Agile Software Development Series
Alistair Cockburn and Jim Highsmith, Series Editors

Do you know what Agile Software Development is? Agile Software Development is a framework or model for developing software which encourages close cooperation between team members and management in order to increase productivity and get software developed on schedule.

Reading this book from start to finish would take a good number of hours and would probably result in a sore head and a look of confusion spreading across your face. But this is okay, as I don't believe that series editors Alistair Cockburn and Jim Highsmith intended the book to be read from cover to cover in one sitting. Rather the book is an extremely useful reference guide

The book stresses how in very small groups a set method of working is often not required – eventually, despite the chaos, you will reach your goal. Whereas a good level of interaction with other team members is essential when your team grows, subdivides and sets themselves up in their own enclosed environments.

A number of methodologies are discussed including Extreme Programming (XP) and Crystal , unfortunately the diagrams used to demonstrate these are not always perfectly clear – they certainly aren't the worst quality figures I have come across but some of them do take a little time to understand.

There are some big words in here and no glossary so you may find yourself wanting to keep a dictionary or computer nearby as you read for a second opinion when you get lost.

The back of the book contains several pages of recommended further reading so this book is essentially an introduction to the world of Agile Software Development, but if you are serious about it you will probably be left wanting more. That's not to say that The Cooperative Game misses out vital chunks of knowledge, for someone new to the principle this should prove a valuable and highly useful read.

Parts of the book appear more like a survival guide for management, demonstrating how certain situations can be solved, and answering criticisms which may arise from general workers or even those in more senior positions who may struggle to understand how Agile development is

System Information DLL

The system information DLL is capable of importing information about your system's hardware and software configuration.

Such details would be crucial if a game is to automatically calibrate its graphics and other settings' quality to a degree that is proportional to the configuration of the system.

This includes getting the total amount of RAM, as well as free RAM. Hard Disk Drive usage, total, and free memory. Availability of virtual memory, how much is taken and how much is free. The availability of a pagefile, and how much is taken and how much is free. All of that in addition to battery status

It also gets software information, mostly regarding the Operating System. These include the native language of the operating system, windows directory, system directory, current theme, and name of computer.

Get it now!

gmc.yoyogames.com/?showtopic=324605

Quick Reviews

an improvement on their current system of work.

One of my favourite bits of the book is a sectioned entitled 'Competition Within Cooperation' explaining why you should reward employees who get the job done, one suggested method is printing an in-house currency with the bosses face on it and using it to offer virtual rewards to employees as each target is met. Likewise fines could be imposed if errors are discovered in builds!

504 pages divided and subdivided into 21 sections splits the book up into manageable chunks and enables you to locate particular titbits of information with ease. With bullet pointed lists aplenty the book is presented in a rather unusual two columns per page layout on almost every page.

On the face of it this book appeared dull but in reality it is a hidden gem. Lots of useful ideas can be picked up if you skip over the sections too focused on specific development methods and instead treat the book more as an encyclopaedia to improving productivity and teamwork.

Agile Software Development: The Cooperative Game explores different methods used in collaborative software development projects which, rather helpfully, are compared to other industries showing that programming as a team is connected to events outside the code monkey's desk.

Who is this book for?

This book appears to be aimed more at those in management positions at small and medium software development businesses looking to expand and restructure or reorganise their development team rather than general public usage. That's not to say there isn't some useful information in there, there certainly is but not everything will apply to you.

Pros

An excellent, well compiled reference helped by frequent subdivisions of the book. Much of the content will also be comprehensible to complete novices and aided by regular references to real life situations the book should give you an excellent insight into the world of Agile Software Development.

Cons

Heavy going, this is definitely not a quick read. Some vocabulary may stretch you if you are new to the area, but this shouldn't be too much of a problem.

Philip Gamble ■

Details

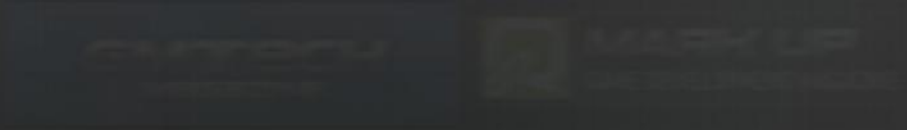
Title
Agile Software Development: The Cooperative Game (second edition)
Author
Alistair Cockburn
Publisher
Addison-Wesley
ISBN
0321482751
Year
2006
Pages
504

ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT - ADVERTISEMENT

Autumnhouse
w·e·b d·e·s·i·g·n

25% off all websites
for GMC Members*
* terms & conditions apply

www.autumnhouseonline.com
webmaster@autumnhouseonline.com



ARE YOU READY?

FOR THE ULTIMATE GAME MAKER CHALLENGE

OCTOBER 15, 2007

www.thegmrace.com

House of the Haunted

What they say

Again, there is no GMC topic for this game. One review has been posted at the YoYo Games game website;

The game is ok, but it need more works. If you try to save the game you might get stuck on it. You should make the grenades more realistic, so they bounce off of walls or something.

Description

House of the Haunted is a simple top down 2D shooter. The game opens with you starting inside a mansion/house (obviously) with a shotgun, and 5 grenades. Zombies will immediately be on your trail, and you will move aiming with the mouse and the up & down arrow keys (which is a minus, because WASD controls would make playing much easier.)

The graphics are simple, and do not capture the mood very well (they don't need to be better, they just need some "atmosphere" added to them, lightning effects perhaps?). The level "building blocks" vary in color every once in a while, but still gets quite dull. The game lacks any sound effects, except for a simple, yet good music tune playing on the background. The level design is in overall simple; some revisions could be made, etc. zombies placed less often, as they are hard to kill in some situations.

The AI is a bit lacking (not that zombies are very intelligent creatures of nature), as what one can see from the game, the zombies on the game screen just sit there until the player is in sight. The small area of view (the player has a small radius of light, the rest being blackness) is very obtrusive in some points of the game, and the effect is not "smooth enough". The game is also very hard in overall, and the energy bar will literally be drained just by one zombie sitting on top of you, as the player does not get any "help" when being hit (e.g. damage protection for a few seconds, zombie kickback...)

Pros and Cons

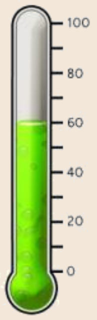
HOTH is a fun minigame, but lacking in many aspects.

Pros

- Basic top down shooter engine well done.
- Fine implementation.

Ratings (99)

Graphics:	★★★☆☆
Sound:	★★☆☆☆
Gameplay:	★★★☆☆
Storyline:	★★★★☆
Design:	★★★☆☆
Average:	★★★★☆



Stats

Developer:	squirrelsattack
Version:	1.0
Made with:	Game Maker 7
Filesize:	2.46 KB

Download

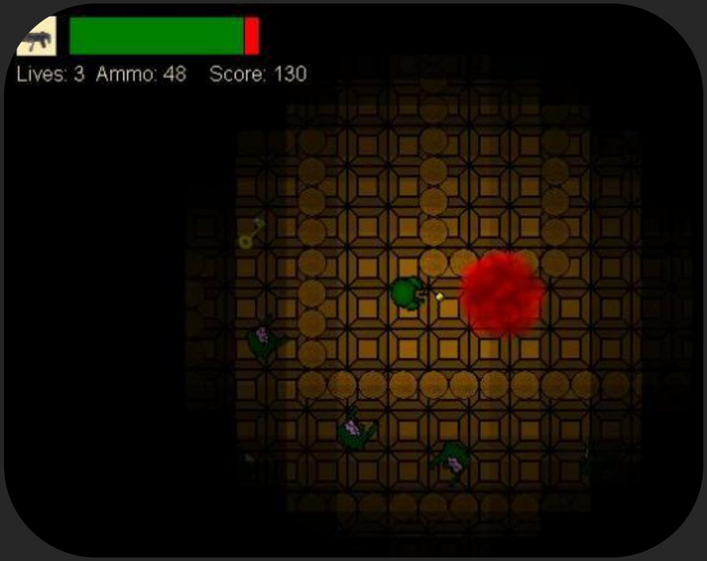
Cons

- Lack of sound effects No ambience.
- Poor AI.

Conclusion

A basic, working idea, but not implemented very well.

Veeti Paananen ■



Dark Hive

Introduction

Making an RTS is not easy; it involves a lot of careful planning, and some trial and error.

Dark Hive was my third Game Maker project. The first one was a hack&slash RPG, which taught me a lot of programming concepts. After finishing it, I thought I was ready to make an RTS (my favourite type of game) and began making one. Well, I wasn´t.

I made a half–cooked Starcraft clone, with poor AI, poor unit management, bad graphics and the code was a complete mess.

But I learned a lot from the mistakes I made there. I took the basic ideas I developed while making that game and started writing another RTS from scratch. Initially it was supposed to take place in an underwater city, and during the first months it had underwater ambientation.

Soon I realised that it was just impossible for me to make a decent underwater explosion sprite. So I decided to change the whole theme. Yeah, really, that was the reason behind a whole theme change.

After that I decided to use 3D prerrendered graphics, and then I came up with the darkness/light idea. Some people from the GM community suggested that a weather/time system would be great too. And that´s how Dark Hive was born.

Reader yawns Ok, ok. I thought that was an interesting story. So you want to know how I made Dark Hive. Well, I will explain the most important parts, showing some pieces of code and explaining their function. The game is open source, so you can download the .gm6 editable file to see how it works:

64digits.com/download.php?name=DHgm6v24.zip&id=10554

Just excuse me a bit because Spanish and English variable names are mixed in the source code!

Part 1: Unit selection/movement and combat system. Part 2: resource/building system. Part 3: IA. Part 4: Map editor. Part 5: Polishing the game and some tips.

Part 1

Dark Hive maps are not grid based. That means that you can have misaligned buildings and that the units can be anywhere, not just in the middle of imaginary cells. Because of this, and some speed considerations, the game doesn´t use A* for pathfinding (implemented as the `mp_grid` functions in Game Maker),

```
/*If I have a clear objetive, and it is within the firing range, shoot at it:*/
if (myobjetive <> 0)
{
    if (distance_to_object(myobjetive) < shotrange)
    {
    mp_potential_step(myx,myy,0,0)//Stop moving.
    direction = point_direction(x,y,myobjetive.x,myobjetive.y)
        if (canshoot = 1)
        {
        alarm[0] = random(2)+8
        myobjetive.life -= attack/myobjetive.shield
        instance_create(x,y,explolight)
        sound_stop(metralla)
        sound_play(metralla)
        myobjetive.agressor = id;
        canshoot = 0
        }
    }
    else //If not, I move normally.
    {
    mp_potential_step(myx,myy,velocity,0)
    }
}
```

FIGURE 1

instead it uses mp_potential step. Due to this design decision, units are a bit 'dumb'. But it is simpler to implement and easier to work with compared to a grid based approach. Now lets see how units move around in Dark Hive (figure 1).

Units have two variables ("myx" and "myy") used to store what we will call their **goal position**, and they move towards it using potential steps when it's not equal to their current position.

When a unit detects that there is an enemy at his goal position, it stores its id in a variable called "myobjective". Then the unit moves towards its goal and stops when the enemy is within its shooting range. After that it begins shooting.

In the following piece of code, notice this line: myobjetive.agressor = id;. That line tells the enemy which unit is attacking him. That way, when we start atacking at an enemy, the enemy knows that we have engaged in battle with him and starts attacking us.

Player units also respond to enemy fire when they are not accomplishing orders given by the player, so they also have an "agressor" variable.

One obstacle I encountered when programming this was: "What if I want to have multiple teams fighting between them? What a mess, how will I identify if a unit is an ally or not?". It´s not a good idea to make a "good" unit object and a "enemy" unit object. Every unit in the game, CPU or not, has a "team" variable which lets the other units know if it's a friend or an enemy. This way we can have multiple CPU/player teams at the same time, and the CPU can attack other CPU teams.

The unit selection system is pretty simple too: Each unit has a variable called "selected" that stores if the unit is selected or not. In their step event I check for collisions with a rectangle, (which is drawn in the draw event from the last clicked position to the current mouse position). Here you can see the code I used (figure 2).

That is the "core" code behind unit management in Dark Hive. It is a bit more complicated, but you can take a look at the editable file.

Part 2

When you have a "pawn" unit selected, a few buttons appear at the lower-right menu. These buttons let you build things.

Again, I tried to keep things as simple as possible. When you click the button, if you have enough resources it creates a "dummy" building that follows the cursor. Instead of having a different dummy for each building type, it has a variable storing which type of building will appear when placing it in the map. Once the player places it, the goal point of the nearest selected pawn becomes the dummy, and when the pawn has been around it for some time, the building appears.

Pawns have also another function: They collect resources. For this they have a variable called "gotowork" (weird name, I know) which stores their current state: Going from the central building to the mine, returning with some resources, returning from a tree, etc. When the goal point of a pawn is a mine or a tree, the state changes and they go towards the mine. Once the mine is reached their state changes again and they return to the central building.

Look at his code to see how they return to the nearest resource object once they have reached the central building (figure 3).

And that´s how pawns gather

↻

```
if mouse_check_button(mb_left) //If the button is pressed
{
    //And i´m inside the rectangle
if(collision_rectangle(cursor.rect_x,cursor.rect_y,cursor.x,cursor.y,id,0,0)
== id  && team == global.my_team)
    {selected = true} //I get selected.
    else if (!keyboard_check(vk_shift))
    {selected = false}  //If shift is pressed, I must stay selected.
}
```

FIGURE 2

resources. They have a variable that indicates their state, and depending on it they change to a different state.This method is usually called a **state machine**, and it´s widely used in programming.

Part 3

CPU controlled units are almost the same as the player controlled ones, the only difference is who sets their goal position.

The AI in Dark Hive is pretty simple, but effective. I made it using just one object, which implements some sort of state machine like the one I used with pawns: Each few steps, the AI checks its resources and its buildings, and depending on that info (state), it decides what to build next. Nothing special, just a big switch instruction and some "ifs". When its army is bigger than the player's, it triggers a variable in every CPU unit of its same team. This variable makes the CPU units enter some sort of "berserk mode". In this state the unit´s goal position becomes the nearest enemy.

And that´s it. But there must be still one more thing bugging you: How the hell does the AI know where to build things? Well, the AI does not decide this; it just creates a building nearby. Then, it´s the building itself who moves around trying to find a suitable place. First it chooses a random place. If it´s too far from an ally, or overlapping something, it chooses a new random place. This proccess is repeated until its position is optimal: Some allies nearby, and no overlapping.

During all this trial-and-error the building is invisible so that the player can´t see it jumping all over the map. It becomes visible only when it has found a suitable place.

```
if distance_to_object(building1) < 10
{
/*If we have just returned from the mine, we update the player resources and
we go to the mine (resource1) again:*/
    if (gotowork = 2) && instance_exists(resource1)
    {
    myy = (instance_nearest(x,y,resource1)).y
    myx = (instance_nearest(x,y,resource1)).x
    if (team == global.playerteam){
    global.ecopos+=10
    drawer.reso1 += 10}
    gotowork = 1
    }
/*If we have just returned from a tree, we update the player resources and
we go to the tree (resource2) again:*/
    if gotowork = 3 && instance_exists(resource2)
    {
    myy = (instance_nearest(x,y,resource2)).y
    myx = (instance_nearest(x,y,resource2)).x
    if (team == global.playerteam){
    global.savia+=8
    drawer.reso2 += 8}
    gotowork = 4
    }
}
```

FIGURE 3

Part 4

Dark Hive´s map editor is just a game that lets you place things on a room. When you save the map, it writes all relevant information about it in a text file which can be read back by Dark Hive.

If every file format is just some text or binary info, what´s the difference between a Dark Hive map and a Command and Conquer map, or a .jpg, for example? Just the rules used to write/read it.

This is Dark Hive´s standard map format, .dhm. It has a campaign variation, called .cml (figure 4, overleaf).

You can open Dark Hive´s maps in notepad, for example, and you´ll see their structure. When you open a map with Dark Hive, it reads the data contained in it using GM´s file reading functions, expecting to find everything organized following the rules I used to write it in the editor. First the map size, which is composed by two real values, then the background, then the teams, etc.

I explain this because map editors

seem a bit mysterious at first, but they are just like any other GM game, with the difference that they store data in external files. This was the first level editor I wrote in my life, so I know that the concept is a bit confusing before you actually start writing one.

Part 5

All the prerrendered graphics in Dark Hive were made using a 3D modelling software known as Maya. There are lots of free 3D programs out there that will allow you to make this kind of graphics, the only thing you´ll need is some 3d skills and patience (since making 3D sprites is a bit more time consuming that just making them with mspaint, for example...but it also looks better). I can´t really say much more about the graphics because 3D modelling is an immensely vast subject that would require a lot of time to explain.

About the music, well, I used a sequencer and some VST plug-ins (midi-controlled software sound synthesizers) another vast subject that could take weeks to explain, just Google for VST and sequencers if you want more info.

Most of the sounds and voices you can hear from the units were made by my girlfriend Lidia "Darkgaze" (who did a great job) and me, so as you can see you don´t need to hire voice actors for your game. Just pick up some friends and give them a microphone!

Finally, the last details and advices:

A well balanced strategy game is a fun strategy game. After you´ve actually finished your game, spend a few days playing it so you can see if it´s fun to play, not very easy or very difficult, etc. I think this is pure common sense but most people seem to forgive about it.

When designing the visual aspect of a game i think the most important thing is to define a consistent style: Never mix realistic graphics with cartoon sprites, for example.

Always try to choose a small colour palette, that will make your game look more polished and will probably give it an unique style. (For instance, Dark Hive´s overall tone is aqua blue, mixed with black and white).

And well, I think that´s all about it. It was fun to make DH, and I learnt a lot. Currently I'm involved in several game projects, although none of them uses GM. But I plan to come back to GM development from time to time. See ya!

Bonus

Dark Hive has two little "cheats" or "Easter eggs" that I think nobody has discovered yet. They´re very simple, but it´s hard to find them, if not by accident:

First, if you press "J" at the main menu, the cursor changes.

Also, if you press "C" at the skirmish menu, you can see the credits, along with some screenshots of early versions of Dark Hive.

The poor voice actors at work.

José María "ArKano" Méndez ■

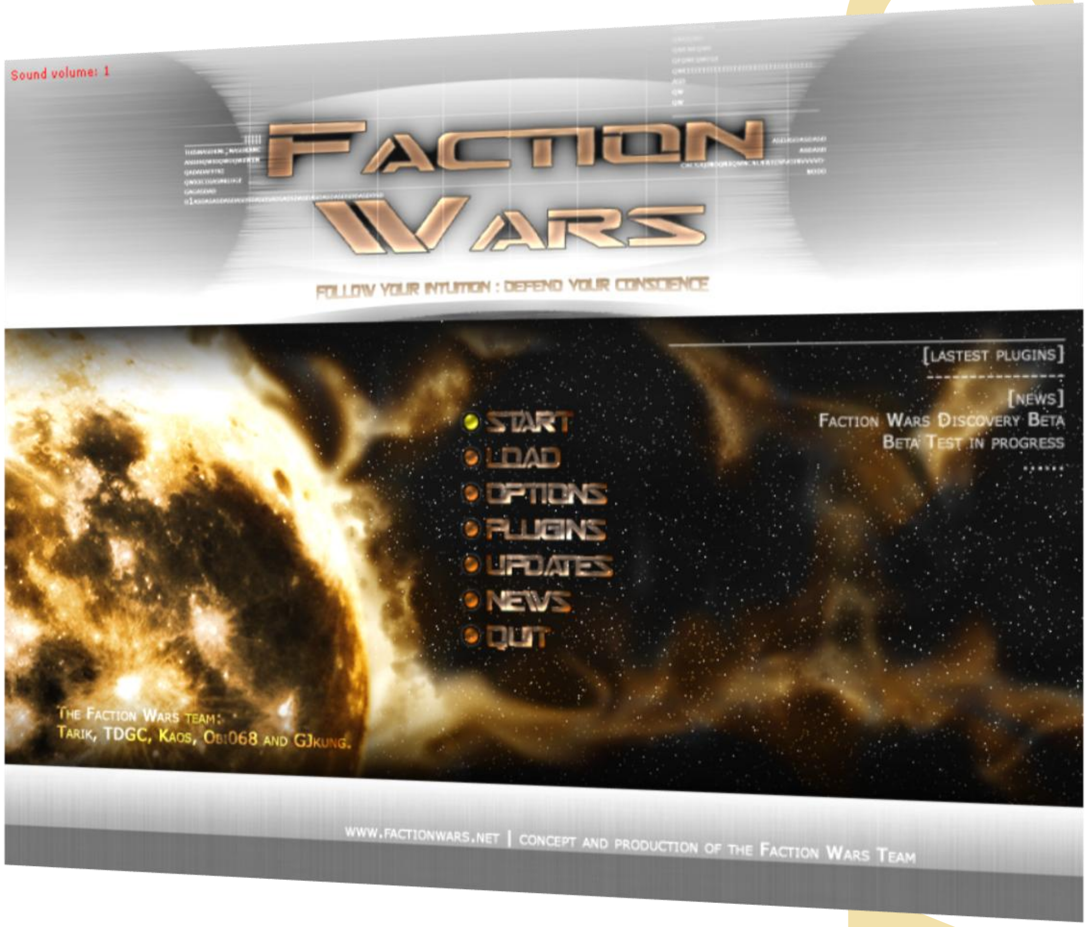
```
//**General Map Info**//
Map_width      Map_height
Background_identifier
Team_array (eight numbers representing the teams that can play the map.)
//**Map data**//
Object_identifier     x_coordinate   y_coordinate   team
.
Object_identifier     x_coordinate   y_coordinate   team
-1 (end of map data)
//**Introduction text (.cml only)**//
Blah blah blah blah blah…
```

FIGURE 4

Faction Wars

Welcome to the first official Development Diary entry of the Faction Wars team. Each week you can expect an exciting update from us documenting the trials and successes that we face as independent, but very serious game developers.

This week we'll be introducing ourselves and bringing you up to date on who we are, what we do, where we're at right now, and where we're aiming to be in the future. As such, this issue focuses on the team and the project, and some of the organisational and structural issues we've had to face and overcome in order to make the Faction Wars project a continuing reality.


Sound volume: 1
FACTION WARS
FOLLOW YOUR INTUITION : DEFEND YOUR CONSCIENCE
START
LOAD
OPTIONS
PLUGINS
UPDATES
NEWS
QUIT
[LASTEST PLUGINS]
[NEWS]
FACTION WARS DISCOVERY BETA
BETA TEST IN PROGRESS
THE FACTION WARS TEAM:
TARIK, TDGC, KAOS, OB:068 AND GJKUNG.
WWW.FACTIONWARS.NET | CONCEPT AND PRODUCTION OF THE FACTION WARS TEAM

The Team

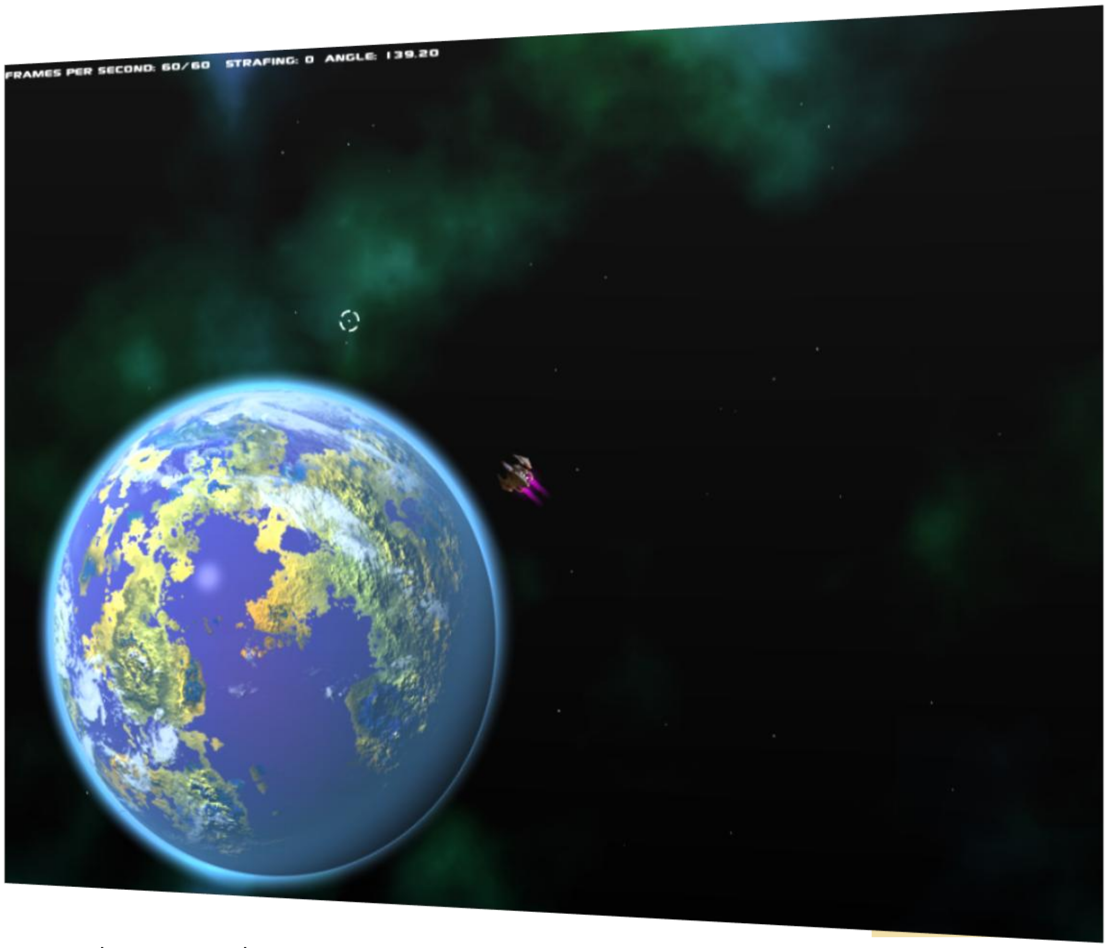
The current core Faction Wars team consists of just two people, Tarik Abbara and Jono Alderson. Between them they like to think they've enough experience of game design and development - and of Game Maker itself - to cover the nearly the whole spectrum of game production. Over their years of working together other team members have come and gone, but Tarik and Jono's vision has guided and developed the project to where we find ourselves today.

The Game – An Overview

FactionWars is a "huge top-down space simulation game" that aims to offer a semi-realistic view of the future, whilst providing the player with a large amount of choice, in depth storylines, sharp graphics and highly addictive gameplay. It's dark, immersive, and extremely dynamic. The decisions, actions and choices of the player in a massive, evolving and constantly changing free-form universe will decide the content, direct, gameplay, focus and outcome of the entire game.

Background

Faction Wars is by no means a new project. In fact, it's been in development in a number of forms and under a variety of different names for well over three years now – and over that period of time the team behind the game has changed just as much as Faction Wars' concept and content. From beginnings buried long ago in ideas and concepts in other games beginning for further expansion and exploration, the game that is now Faction Wars has come from being a 'Space Shooter' with role-play and strategy elements, to arguably one of the most ambitious, large scale and professional undertaking in the Game Maker Community's history. Only recently has Faction Wars has reached the point where we could finally and say with certainty that the concept was finished. The thousands of lines of code that we've developed as ideas for scripts and subsystems in the game can be archived, put away, and the real coding can begin on concepts for systems that have been years in development.



BEGGININGS

Perhaps our approach seems a little excessive. We've been actively designing and developing this game for a number of years now, and we don't really have any significant Gameplay to show for it. However, we've managed to avoid one of the biggest and most dangerous pitfalls that an independent game developer can fall into.

Chances are that if you're reading this, you're one of the many talented, creative and dedicated independent game developers in our community – and chances are that you have a desktop full of half-finished ideas, partially coded systems and unfinished projects that you haven't touched in weeks. Sound familiar?

So, you've a concept for a game that you're happy with – something that you're really excited about. You start coding, start designing, source some lovely graphics. A week later you've a half-designed level, a bunch of code that you can't make heads or tails of now, and no motivation to sit down and go through it all.

So, hang on a minute. There's got to be a better way to approach this.

Before you start coding, how's it all going to work? In the early days of Faction Wars we talked about trading resources between various outposts and colonies. So, we're looking at money, we're looking at inventories, different types of items, different types of *individual items* (Oh yes – not happy with hundreds of different types of weapons, ships, equipment and more, each of those types must also have near-infinite variety from object to object...), and much more.

So, we need to be able to store the information for all of this in a way which makes it easy to retrieve and display it for shops, inventories, and a whole host of GUI interfaces.

But wait, there's more. What happens in a month down the line when it's all finished and working, and somebody says 'Err, wouldn't it be cool to add a cool electric beam gun'?

Suddenly we have a rigid, complex system designed around a massive database that's been designed to process and display only a limited number and certain type of variable throughout the game – a system that's size and complexity means that it's near impossible to incorporate new content without revisiting that system from square one.

See where this is leading?

The Faction Wars team has spent months writing, assessing, rewriting, revising and updating System documents that aim to chart – in fairly technical detail – *everything* in the game, and only now are we really starting to settle down into coding these systems. We know exactly how everything we're making will look, work and function right down to how we're naming, storing and calling each individual variable. We've now a pile of documents as high as our collective heads consisting of pages and pages of technical documentation on how various in-game systems should work, an entire systemised, structured universe with its own physics, inhabitants, politics, economy, wars, resources and more – and these resources have been invaluable.

If we'd just gone ahead at day one and begun to produce Faction Wars on an ad-hoc basis, then we'd either have ended up with a game much, much less than this version will become, or in all likelihood would have given up after repeatedly needing to 'return them to drawing board' as the concept for the game continually evolved.

The moral of the story is a simple one, but one that many independent game makers overlook.

Know what you're developing.

Know how you're going to develop it.

'Finish' as much of your game as possible before you even start it.

Next Month…

So, you've met the team, you've heard a little about the project and how we work. Next month we'll start looking at how we tackle the systems and the code behind the project, and we'll be sharing all sorts of useful snippets, scripts and advice.

In the meantime, we'll leave you with a sneak preview of our first steps into producing the final revision and draft of the game – you can download our current technical demo from our forum post at gmc.yoyogames.com/?showtopic=37012 (website coming soon…) and start to get a feel for what all the fuss is about. We'd love to hear from you, so don't hesitate to get in touch with us!

For now, that's it from us, and we look forward to sharing some time with you next month.

Thanks for reading!
Jono Alderson & Tarik Abbara
***The Faction Wars Team*■**

GM Tween

GM Tween is a GEX Game Maker Extension Package that 'tweens' between variables – such as numbers and colors, to create a great animating effect.

What tweening means is that Game Maker will now take two variables into the function, use them as 'key frames', and according to the length of the time limit between those two frames, the function will fill in the missing frames in between.

This is mostly useful for colors, for example if at time (t=0) the background color is to be blue, and it should fade to red at time (t=30), you could tween between `c_blue` and `c_red`, with a time length of 30 frames between them, and enjoy a fading 'effect' between the two.

GM Tween supports repeating tweens between colors, or repeating the loop of tweening over a certain time period.

Give it a try, it is certainly interesting.

Get it now!

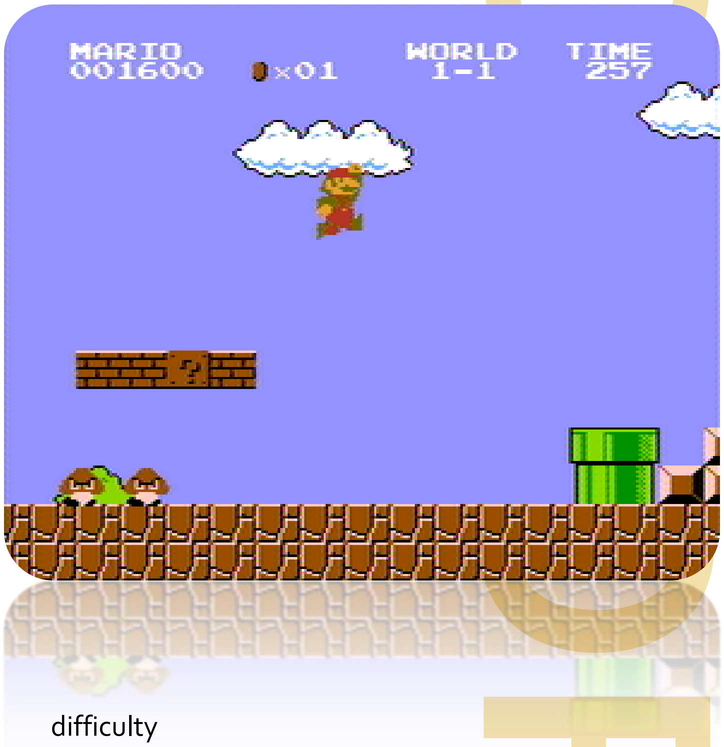
gmc.yoyogames.com/?showtopic=327281

Guide to Level Design

What do you see in the above picture? Some see the first level of **Super Mario Bros**. and others see some random platform game. Level design is an important aspect of game making, and I'm going to discuss about some guidelines in platform levels (and others too, but in a more general sense) which have been formed by the legendary game above, **Super Mario Bros** for the **Nintendo Entertainment System**.

Making levels is a boring task to many, so they get all sloppy with them (as one can see from the numerous bad platformers in the GMC). But keeping a few rules in mind, you can make levels in no time with small time, without making it annoying to the player to play an incomplete level.

When "mapping" (a term to "level making") one should try to avoid major variation. Your player won't be too happy if levels vary in design much every level. Imagine a simple map with slopes and platforms in the air. Fun to play. But the next level, you're facing 5 hard enemies every 20 seconds, impossible jumps and complex terrain. When making **major alternations** to the main map principle in your game, **start slowly**. Don't throw all the new stuff in at once, **but combine the story to it**. For example, try to combine changing terrain with the story. You move to a different territory, start in a "generic area" in this territory that contains only small changes to the previous territory. As the player progresses deeper to the territory, more obvious changes appear.



Try not to make it too hard for a new player. When the game progresses, increase difficulty moderately. Make the first levels obvious on how to play (no hard puzzles, etc.)

Always focus on thinking if it's a good idea to place a certain object in a certain location or if you should reconsider placement. Level design is an important and hard aspect to master. Never place impossible to play scenarios in your levels unless you want to frustrate the player (which isn't really advisable): if you have walls coming to crush the player from 3 directions in high speed, a player will get confused at the movement and not be able to move quickly enough. Avoid "get crushed" situations, especially long ones.

Variation of the main platformer idea once in a while (not in the earlier sense) is a good idea; for example, in **Mario**, you ride on a giant mushroom in one level, and there are coins on the way, and the player can take the challenge of picking them up. A fun challenge for those who think they can do it.

Adding secret Easter eggs to level (such as the extra ways to get wings in **Super Mario Bros. 3**) is a good way to keep the players interested.

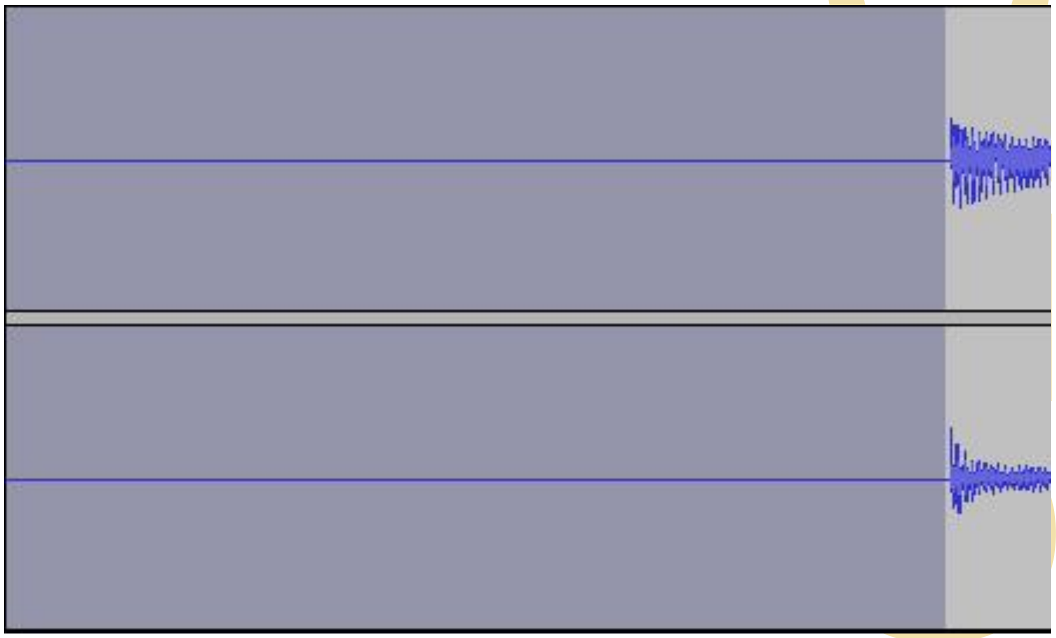
Veeti Paananen ■

Pay attention to Sound Effects

I'll be talking about sound effects' importance in games. For a beginning, imagine playing a game with no sound. It's a good game, and all, you like playing it. But it just feels… "wrong". ***It's lacking sound.*** Maybe it's got a nice ambient tune playing in the background, but… it feels wrong without any effects. You shoot, you don't hear anything. A cool muzzle flash effect is on screen though. Still doesn't feel good. Why is there no sound?

Imagine the same, with great sounds (relevant "gun shoot" sounds to the gun you're using in the game, footsteps…). It's **awesome** and it feels great just to fire the gun. The muzzle flash, combined with a kick ass sound effect = **great.**

I've seen many games that do a bad job with sound effects. Some have hastily picked sounds that don't fit, some don't have any at all, which is ***really bad* and sinks respect for the game immediately.** Finding good sounds that fit in well is very painful, but if you bother to make a good end result, it'll be worth it.



Integration

Free sound clips that you can find in the internet (not that you should use them for copyright reasons) often come in a bad shape for integration in games. If they don't sound good in your game (timing is wrong, e.g. blank space in the beginning of the sound), get a copy of Audacity (<- insert link there) and remove blank spaces, so that the sound is immediately played. A sample image is shown above.

In this picture, I've selected a blank space of about 1 second in the beginning of a sound file. After selecting it, simply press **Delete** on your keyboard. Play around with the file until there's no blank gaps, and it'll play better in your game.

Summary

Sound effects are a very important part of a game. Without them, you're basically screwed: nobody wants to play a game without ey—ear candy. No matter how awesome it **looks**, it's gotta **sound** great too. Bother! Don't leave poor sounds be! Look for the best possible sound, and edit it until it's **perfect!**

Veeti Paananen ■

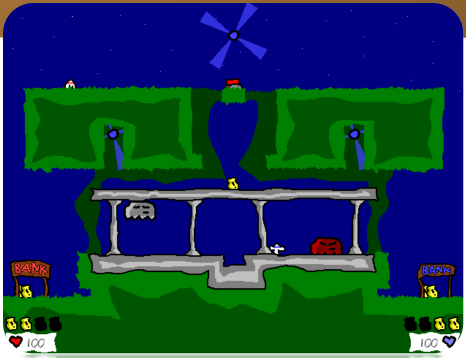
MarDar: CREATIONS

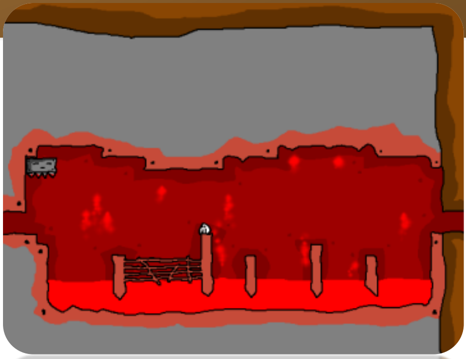
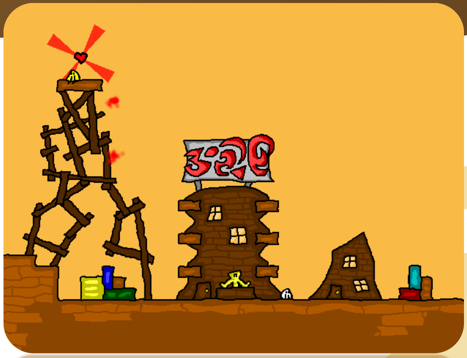
An Untitled Story

"An Untitled Story" is a production of YoMamasMama, most famous for the creation of the Game Maker 'hit', Jumper. You play as an egg in a nest, trying to overcome obstacles in this incredibly cartoonish world, fighting monsters and eighteen unique bosses on the way.

This game – "An Untitled Story" – is not freeware, so if you're interested in playing the full game, you are required to give a donation to Helix Games – at least 1 CAD.

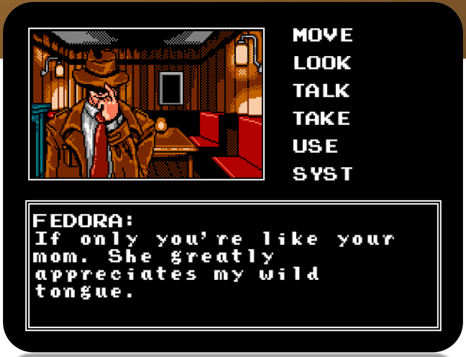
Most community feedback was positive, and negative feedback only concentrated on the way the graphics are and their "unrealism", which according to the creator, is done on purpose to give the game a nice atmosphere – agreed.



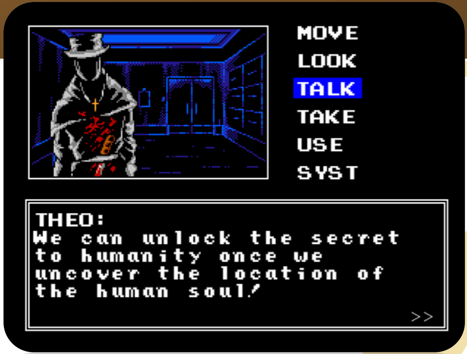



Fedora Spade

"Fedora Spade" was created by rinkuhero and Orchard-L. The game is a NES-style detective game where you have to solve crime cases. The game actually contains two episodes of the same game, or almost a "built in sequel". The graphics are incredibly pixilated, and only 8-bits, which is done on purpose to give the game a unique NES style that was long lost in regular games in this generation.



FEDORA:
If only you're like your mom. She greatly appreciates my wild tongue.



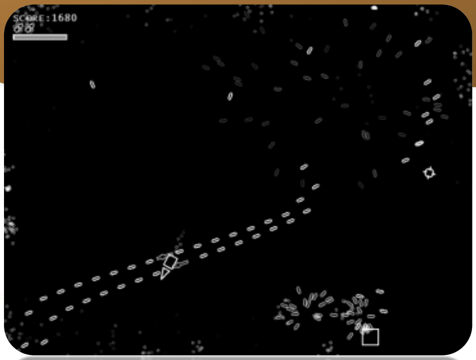
THEO:
We can unlock the secret to humanity once we uncover the location of the human soul!

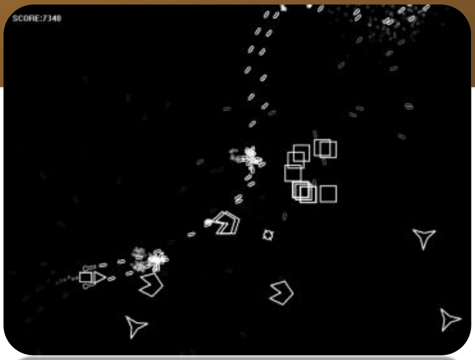
MarDar: WIP

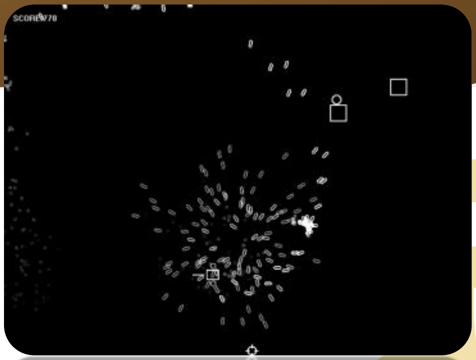
Another Shooter

While originally posted in July, the latest beta of "Another Shooter" has been recently released. Another Shooter could be considered a remake of the famous "Asteroids" game, however in fact it is much more enhanced – both graphically and gameplay wise.

Even though its completely black and white, with only geometric shapes, the graphical experience is certainly innovative to say the least. I release many other games have done that black and white approach, but the fading effects, smoothness, and all that are just amazing!



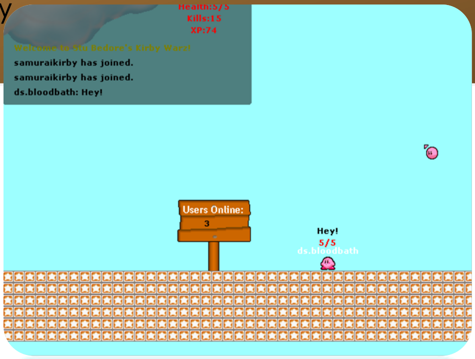


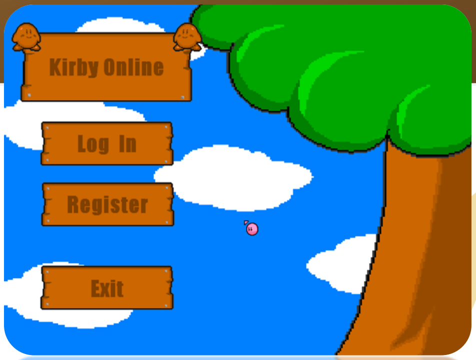


Kirby Warz

Kirby Warz is a massively multiplayer game that takes place in a two-dimensional platform world. The game uses 39dllfor players to connect to the server in order to play against each other.

I guess "cartoonish" and "cute" could describe the game well, although I was disappointed at the lack of a story line – even for an online game, a storyline is necessary.


Welcome to Stu Bedore's Kirby Warz!
samuraikirby has joined.
samuraikirby has joined.
ds.bloodbath: Hey!



Eyas Sharaiha■

Good Bye!

And so it ends, Issue 8 of MarkUp Magazine, for October 2007. This was our Spooky Special; we hope we haven't scared you a lot! The special wasn't meant to be created to celebrate Halloween as a special event or holiday, but rather as a social event that people tend to enjoy in many places of the world.

In this issue we also introduced **Book Reviews**, in which our staff review books related to game development. The staff receive the books for free from all of the major game development book publishers in the world! You can take advantage from this offer by becoming a trusted staff member. Click here for more information.

Remember, MarkUp needs your help and contribution! Please contact us and contribute to MarkUp Magazine by either joining the MarkUp forum, or e-mailing the MarkUp staff.

The MarkUp Staff■■

Check out…

GMking.org is the parent network for MarkUp magazine. It has been recently redesigned to behave as a centralized portal that links to the four main aspects of GMking.org's projects: The GMking.org Site [which is now a sub-site of the main gmking.org page], The GMking.org forums, GMpedia.org, and MarkUp magazines. Come look at, and enjoy the redesign!

MarkUp has sister projects, also developed and maintained by GMking.org, all meant to help Game Developers. To learn more information about your Game Platform of choice, you could check out GMPedia.org. GMPedia is a game development wiki with a growing community-base and content.

GMking.org *Let them make games!*

MarkUp is an open publication made possible by the contributions of people like you; please visit markup.gmking.org for information on how to contribute. Thank you for your support!