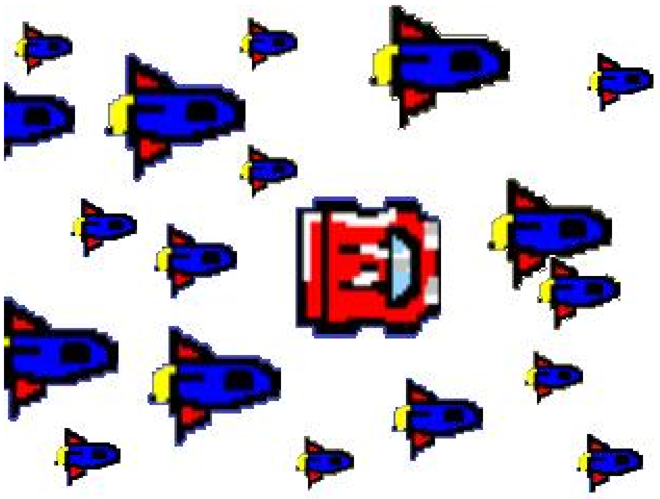
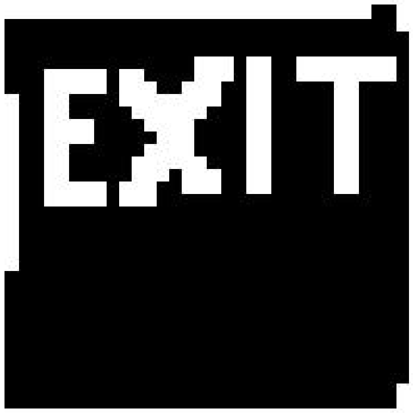




THE MAKING OF…

RANDOMWARE

The ‘next stage’ in computer game innovation, and one step closer to getting the computer to do the work for the designer. We take a look at how the ‘Computer Games Martyr’ created this project

FORMAT: WINDOWS, LINUX PUBLISHER: INNOVATIONS IN COMPUTER ANIMATION DEVELOPER: EDWARD FOWLER ORIGIN: NCCA BOURNEMOUTH, UK RELEASE DATE: 2006

The idea began on a bus journey. I was playing a game called Warioware (Nintendo, 2003) on the Gameboy Advance. I’d already played each of the two hundred so-called “microgames” and completed the other modes. All that was left was to play the same microgames over again, competing against my own score.

Since microgames are as simple as they are, I also had plenty of time to think about what I was doing. I realised that after “exploring” and “unlocking” every part of the game by now, it was no longer as fun as when I had first played it. The reason was because there were no *surprises* left to find which, when unlocked, each feels like a little reward for playing – a new “piece of game”; the reward being a new challenge set – and being new somehow recreates the exciting feeling of playing a game for the first time. (Well, you are playing that “piece of game” for the first time.)

If only there was *something* left to unlock, although I would still be playing the same game, the possibility of a new discovery would give me more incentive – making the same experience more fun. *If only there was more to unlock* – if only it *never ran out of freshness,* then it would *never run out of fun.*

It seemed a totally impossible idea, but it was so crazy that I thought it just might work. I was certain that, even if someone else had already thought of it before, in the modern state of the computer games industry, it’s an idea that sounds far too risky to even attempt – *it’s an idea without proven success.* Even so, it was an idea I became interested in trying out for myself – *it contained the same excitement as playing Warioware did when I was trying to unlock a new level: the reward for taking the risk is the possibility of unlocking a brand new discovery.*

At first, I had planned that this would be my Major Project. However, when the time came to decide, I was afraid of the lack of proof that this idea would work – possibly ending up as a complete failure. So I saved the idea for later, and

Warioware’s concepts: “Infinite Fun”; “Instant Action”. Concepts I believe in, however the fun decreases the more it is played. Its flaw: the design was completed *before the game was published*, meaning that inevitably it can be “finished” by the player. Repetition is its only hope, however it is well known that repetition gets boring after a while.

started my Major Project with the intention of making a straight-forward game, based on tried-and-tested methods. When it came to the Innovations Project, I realised the potential it was giving me to be able to try almost any kind of idea – perhaps even the crazier, the better. So I didn’t hesitate to choose this idea for the project – the difference being that Innovations gives me much less time to work on the project, but at least it was an opportunity at all.

Because of the lack of time, as well as Warioware being my main inspiration, I decided to keep it all simple from the beginning. No 3D graphics, no complex puzzles, physics or algorithms. I just primarily wanted to test the idea to find out the feasibility and implications of it.

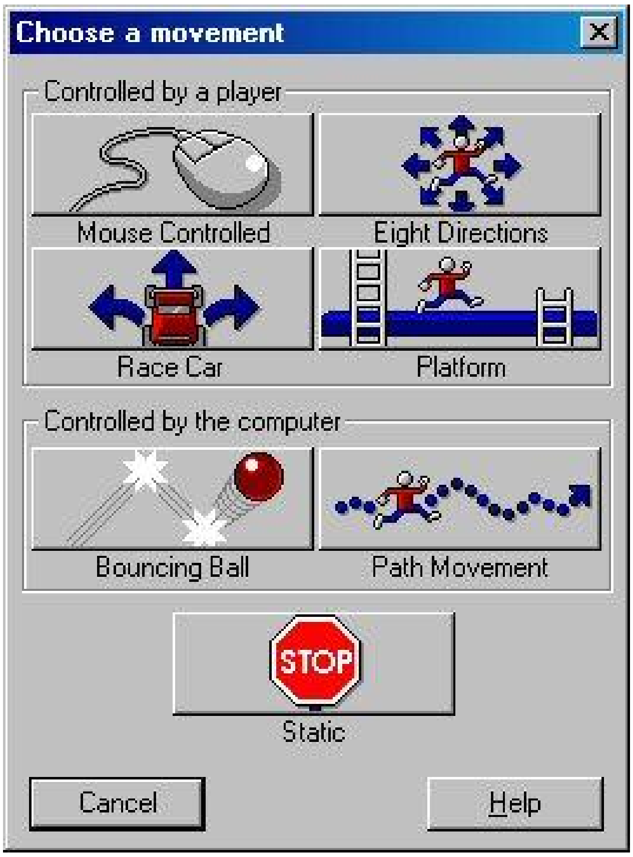
The first stage was quite difficult – I began trying to analyse gameplay mechanics with the goal of generalising any game in terms of one generic set of variables. The plan was, that these variables could later be altered – a small alteration would theoretically change the game slightly, whereas a large alteration would create a “new” game altogether.

For more inspiration, I remembered a tool I had used in the past to create games – The Games Factory. The idea behind The Games Factory is that you are given a wide selection of rules which can be mixed and matched in a huge number of ways, so that without writing any code, you can theoretically create any game you can imagine. However, something notable about The Games Factory is that not everything is variable – some of the rules are already in place. These can be selected by the user, and altered, but only to an extent. This fit with my goal of being able to create a huge range of different games, even though some of the rules are predefined.

So at this stage, the way I went about analysis was listing what is varied between different games, and what is fixed. The list of variations included graphics, the ways in which objects can move, and size, position and number of objects in a level. The list of commonalities between games consisted mostly of fundamental gameplay mechanics, such as “pushing a given button will cause a particular action”, “there must always be an interactive avatar representing the player,” “there must always be at least one collectible object, i.e. ‘goal’.”

This way of thinking is very similar to the way in which Warioware works, but even more fundamentally, “pressing the right button at the right time = win.” Sometimes this is repeated in a sequence per level, but it is always true. For example, jumping over a skipping rope can be generalised into the form of “pressing the button which represents ‘jump’ at the time corresponding to while the rope is lower than the height of the jumping.” The only notable difference between the various games is the appearance of the game. Picking a nose in Warioware also requires you to press the action button at the right time, but in this case, “while the finger is under the nostril.”

Less fundamentally, it could be interpreted as “pressing the button representing ‘pick’ causes the finger to move upwards. If the finger collides with the goal, you win; if the finger misses the goal, you lose.” This is the level of complexity that seemed appropriate to Randomware. Making the “building blocks” too simple may generate incomprehensible scenarios; making them too complex would limit the variation. I required a level of abstraction that I could understand in order to vary it, yet abstract enough to allow a wide difference between two given games that were generated.



The Games Factory (Clickteam, 1996) contains a finite selection of movement types. I decided that if it was hard enough for me to randomly generate movement from scratch, it would be even harder for the player to work out the brand new movement types within a short time limit, thus I decided to also use a finite number of movement types. It is still perfectly possible to create a huge variety of games in TGF—there are nonetheless many other variables to vary within a game.

I began to do some research, although there are few books and articles written that are relevant to this project.

Raph Koster had written a book I’d read, called A Theory of Fun (2005). His theory is that “fun” is a feeling of pleasure people feel in order to encourage them to learn. In other words, the brain “enjoys” learning new patterns. Once a pattern is mastered, it becomes boring. If a pattern appears too complex, the brain is confused, and “gives up.” On the other hand, if a pattern is too simple, the brain sees no need to learn it, so also “gives up.” This is the definition of what we call “boring.” This theory agrees that the concept behind Randomware has a lot of

Randomware's textures—many textures for a few objects. More = better, yet since this was just an experiment, I limited the number just to test the concept. However, perhaps a random image-generator is the next stage in the future of this new genre of game?

potential to be a successful game, i.e. fun, due to its possible lack of repetition.

Also, In The Game Inventor's Guidebook, - a book containing information on how to design games - in an interview with Mike Gray (Senior Director of Product Design for Hasbro games), he says games must contain an "element of luck." This generates "surprises" and makes a game more "relaxing."

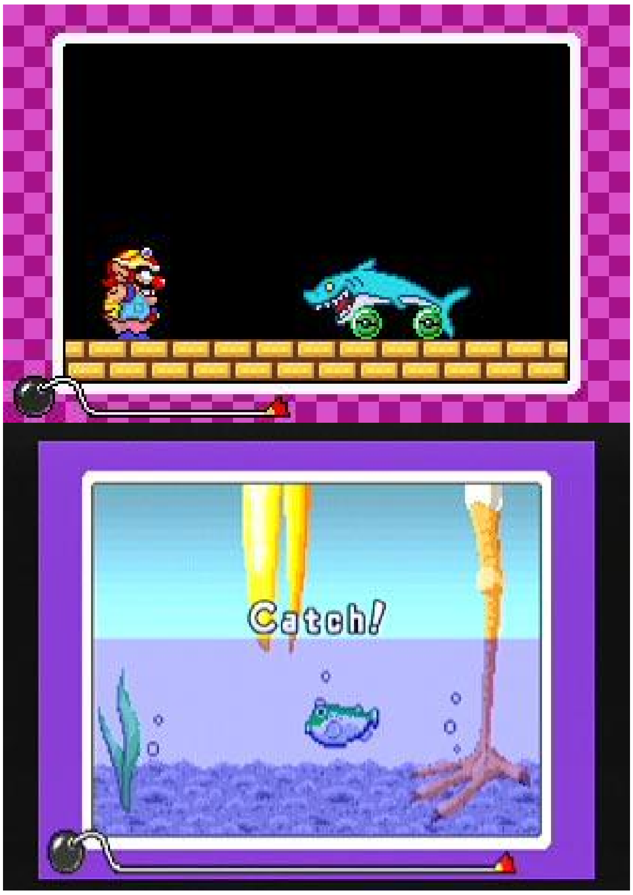
At this point I began working on a prototype in C++. The OpenGL set up tied in with my Major Project, since I already had OpenGL set up for that project. So I simply took the corresponding code and I had the basis for my game ready.

As opposed to the OpenGL set up coming *from* my Major project, later on in the game's creation I had worked out how to make the frame rate of the game independent of the other calculations involved in the game. So this is the one and only direct influence this project has had on my Major Project – although it was a very useful technique to discover.

I began generalising as much as possible, and making classes based on the common attributes I had decided upon, between games. This included "size," "position" and "movement," which became attributes of less abstract, actual physical objects, which would represent obstacles, goals, and the playable character. This allowed the size, position and movement of any particular object to be explicitly varied, providing the ability to drastically change the way objects look and behave. This is rather similar to the concept of The Games Factory, which was a major influence.

In terms of a schedule, I planned on setting myself weekly goals – it didn't seem appropriate to create a complete schedule in advance, when I was working on an experimental prototype – I didn't know how it might change each week. However, as the weeks passed, it turned out that the project was steadily developing and changing, so in reality, it became a continuous "work in progress," as opposed to a sequence of "goals."

Earlier on I had come up with the most fundamental analogy I could think of while analysing gameplay - "hunting" and "being hunted." In other words, there are objects you *must* collect in order to complete the game, and there are also objects you *must* avoid. Every object in a game other than the controllable avatar can be thought of as an obstacle or a collectible – "predator" or "prey." Even where this is not directly apparent, it is quite easy to work out a way of building any particular gameplay situation out of these two types of object. For example, an exit to a level can be thought of as a collectible object that must be collected in order to complete that level.



Warioware: Jumping and catching—two apparently different physical concepts applying exactly the same gameplay mechanic simply by changing the graphics—"press the action button at the corresponding time." Both levels communicate the challenge by the position of the objects on the screen.

After a few days, I'd got a working prototype. All objects were randomised in terms of their position and size. As for player control, I added a few different types of predefined movement, including "race car" movement, "2D directional movement" and platform movement. The plan was, I could test it with a number of different types of movement, and if more different types of control proved to be more fun, I could add more later. For ease, and since I had no graphics ready yet, objects were represented by different coloured spheres – the controllable character was a teapot, just so I could see which direction it was facing in.

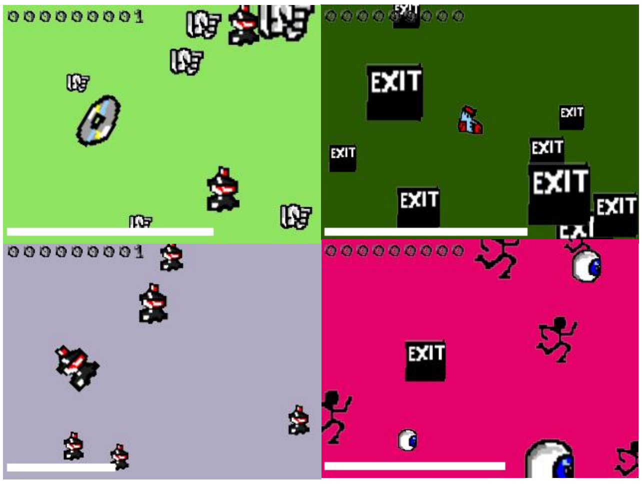
The window size, I decided to keep relatively small—I didn't think gameplay would suffer due to a low resolution, and it would help ensure I kept things simple. In retrospect, changing the shape and size of the game window could open up a whole new area of possibilities—I believe that could become a whole genre of game in itself.

The collision detection was very simple. Obviously more complex collision detection might have made it easier to play, but I got radial collision working, and didn't really notice any major problem with it – it worked as I expected it to throughout the project.

An apparent unique feature which I noted at this stage was that, due to the random nature of the level design, it was not always possible to complete the level. I considered adding some order to the level, however, any order in the level design would directly remove the element of randomness from the game, so I decided to solve the "impossibility" another way.

To make the game fun, it needed more of a challenge – you won if you collected the objects, but there was no incentive to do so. So, since I was basing the project on Warioware, I followed its example and added a timer – which created a challenge, and a score counter – which created an additional incentive other than just progressing to the next level. These proved to be successful overall.

Something I still haven't decided as to whether it was successful or unsuccessful was the alternative to adding order (adding more rules) to levels in order to prevent levels being generated that are impossible to complete. I allowed impossible levels to be generated, but pressing the Insert key skips the level. This gave me several variables to "balance" in order to keep it both fun and challenging. This balance is kept by several aspects of the game: the score only increases once you complete a level, encouraging you to collect everything in the level; the timer is running out during the level, creating a (mental) challenge; the combination of those things means that skipping a level causes a penalisation, since you don't score. This means there is the responsibility to only skip a level when absolutely necessary, encouraging the player to focus harder at what they are doing. I still haven't added a limit yet to end the game – firstly because I still think of it as work in progress and secondly, because of this fact, I haven't come to a decision to how best apply a limit to the game.



In Randomware, the controls, the appearance and the layouts vary at random level to level. There are no clues—it is somewhat an intellectual challenge—you have to re-comprehend the gameplay every level. It is not totally void of repetition, but that isn't to say it is impossible to remove repetition from gameplay altogether. The challenge for the designer is finding the balance of game design between taking charge and letting chance dictate the end product. At what stage does randomness make the game stop being fun, but just confusing?

Throughout this process I got people to play the game. I found out that sometimes they became confused due to the fact that, although the graphics stayed the same, the gameplay changed, which made it difficult to adjust. For this reason, as well as simply out of curiosity, I added graphics. It turned out that I'd underestimated the effect graphics would have on the game. I kept them as simple as I thought they needed to be; I tried to make them as varied as possible; this meant that the game gained a certain kind of charm, which made it more exciting and easier to relate to.

Although the graphics at this point were crudely-drawn icons on 2-dimensional planes, and did not fundamentally alter the gameplay, I was surprised at how much they did alter the game – it was the graphics alone that alter the user's perception of what is "going on." For example, a *car* driving around a level like a car is expected, but what about a *sock* driving around like a car? It's *the same game* in an abstract sense, but it tells an entirely different story to the player. Combinations of graphics also include a number of the same type of object that behave differently, and objects that behave unexpectedly, such as what is apparently the "exit" is in fact controlled by the player.

I made Randomware for Windows, because that was what I use at home, and what the majority of people I know use. This enables me to distribute my project to people easily and get lots of feedback on it, even if it's purely their reaction to the project. However to demonstrate the project to other students in the university, I have also had to get Randomware running on Linux. This is a bonus, because it means a wider range of people can play it than with only the compatibility with one platform – this means that I can potentially get more feedback and, considering the experimental nature of the project, would be highly valuable for any later continuation. Plus, I learnt a lot about the way they each link shared libraries differently.

What I had expected when I first imagined the result of this project was essentially a game like Warioware, except that it never runs out of levels. Over the course of the project, I began to hope for more possibilities, including the possibility of software which creates new genres by combining abstract rules – entirely new games. In a sense, I have failed in not creating as ambitious a result as I had hoped for. The games generated lack the amount of variation I would have ideally liked to see.

At the same time, on the other hand, Randomware is a kind of window into the "world" in which chance is allowed to dictate game

Nintendo "Revolution Controller" (as yet unreleased): Simply by pointing and moving the controller in space will possibly directly control the game accordingly, with the aim being intuitive control. Thus the need for instruction manuals is decreased, a whole possibility of new game genres is created, with the increased intuition acting as connection to people who don't generally play computer games.

design. Every random-controlled possibility I had hoped for does exist in Randomware in a small amount. The new games created are so simple that it might be debatable as to whether they are whole new genres or whether or they are abstract pieces of games that already exist.

In order to be classed as a "game" it must be fun and it must have a challenge. Randomware arguably succeeds at being a game. What I was most pleased about was that the *randomness does enhance the fun* of the game. Although each level is so simple and somewhat limited, it is still surprisingly exciting not knowing exactly what the next level is going to be.

When viewing Randomware critically as a computer game, I think of it as a new genre of game. The unique aspect about this game is that there are no fixed instructions and no fixed mission – the challenge is working out how to actually play each level—whereas a typical computer game would seem lacking without an instruction manual—it is what people who buy games expect. As complex as games are nowadays, it might be a challenge to play one without reading an instruction manual first. This ties in with the concept of Nintendo's new "Revolution" console—the theory is that by making games less complex and more intuitive to control, the popularity of computer games will expand beyond their current audience.

I think of Randomware metaphorically as a superset of all of its non-random generated games. What I mean is, by fixing the variables to the right specific values, it is possible to create a non-random specific game. This is the equivalent of choosing a particular game from a pre-defined list.

Randomware as it is, however, contains many of these games literally mixed together in random quantities. Conversely, there is the possibility of adding even more randomness to Randomware. I believe that too much of this could make the games closer to impossible to complete. Yet judging from my experience with Randomware, I believe that increasing the randomisation would actually make it more fun, when the levels are possible.

The other possibility of future modification is adding more variables and rules that are subject to randomisation but also compliment the game. This could create bigger, more interesting games, with a wider variety of outcomes. Examples might include the ability to shoot bullets, or the ability for collectibles to affect the gameplay during the level, i.e. "power ups." With enough added variables, I think that there is definitely the possibility of a game which creates new genres at random, which is something that I would be excited to see.

Another future possibility was one discussed at the beginning of the project – setting up particular values to be controlled by specific inputs – this could allow for many interesting possibilities, such as a game controlled by weather – perhaps brightness could be controlled by the brightness of the sky; maybe what clothes the player is wearing, or who is playing could determine the game generated; perhaps their mood could affect the game – the possibilities are wide and intriguing.

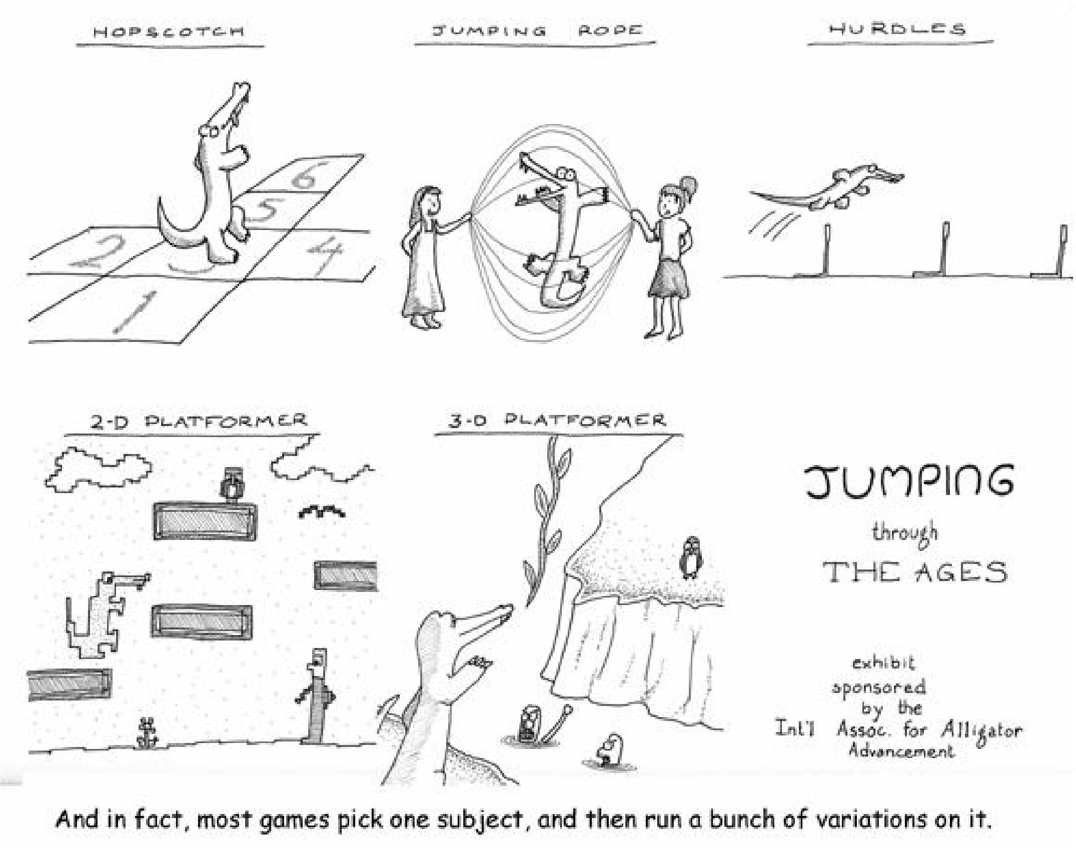
The big obvious aspect that was neglected was sound. I focused on the game mechanics first of all, then the graphics are the next most important feature ("video" game). Instead of add complexity to the project, I decided to simply ignore this aspect of the game altogether. I don't think it impairs the project, however – it is simply not noticed.

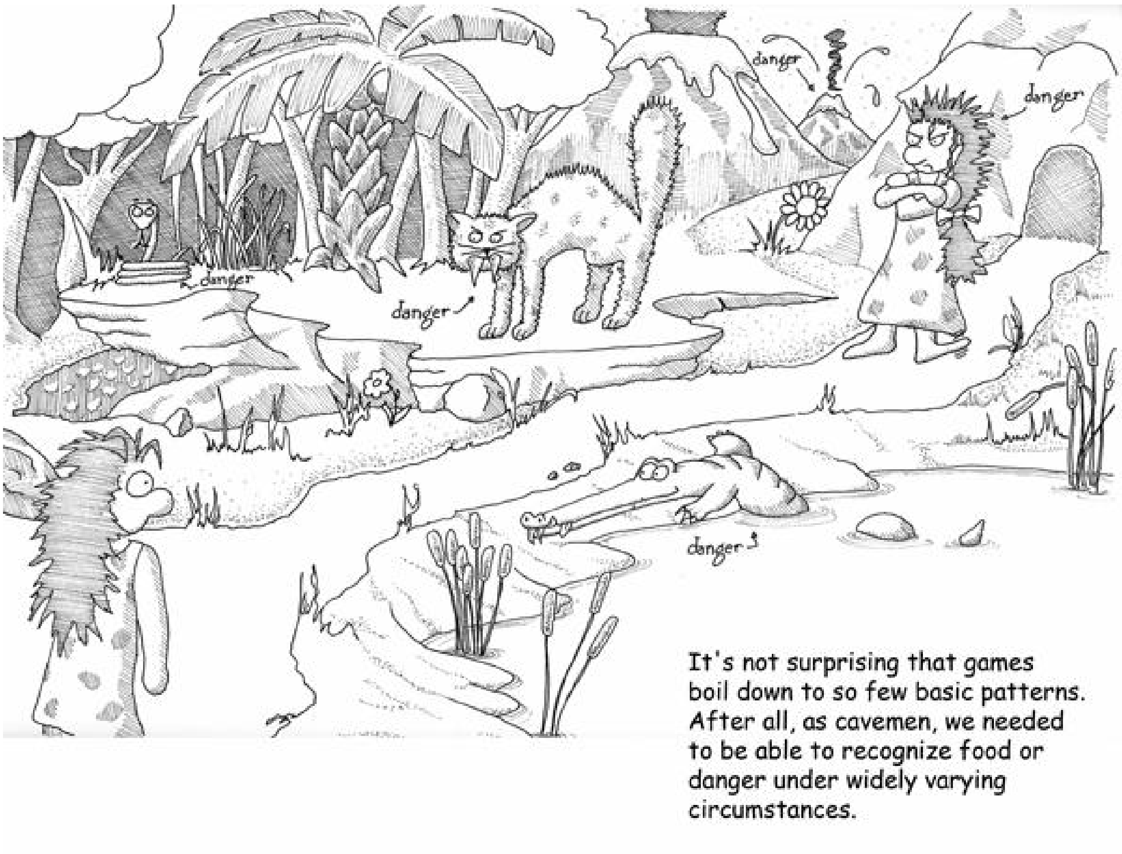
When I first started work on this project, I wasn't even sure whether or not it would be possible to let the design of a game be dictated by chance. I have discovered that if chance is allowed to have a selective effect, when chance affects the appropriate properties of the game, it *is* possible, and in fact, *beneficial* for this to happen, since it adds fun to a game.

It has also made me question, what is the difference between "designing" something and simply allowing something to "form"? The only apparent difference seems to be that a "design" has some inherent source of inspiration. If we are prepared to experience new things, chance can potentially be just as practical – life itself is believed to have happened by chance, so perhaps it isn't so naïve to have faith in it after all? It might even be argued that if nature is dictated by chance, and all art is simply imitating nature, perhaps all art is dictated by chance anyway!

A drawing by Raph Koster, author of A Theory Of Fun. The same gameplay is reused to create “new” games, that are essentially all the same game—the graphics are what is varied in general.

Many people might describe these as a number of different genres, however, from what I have learn by developing Randomware, I would describe this as one genre, with different graphics.


HOPSCOTCH
JUMPING ROPE
HURDLES
2-D PLATFORMER
3-D PLATFORMER
JUMPING through THE AGES
exhibit sponsored by the Int'l Assoc. for Alligator Advancement
And in fact, most games pick one subject, and then run a bunch of variations on it.


danger
danger
danger
danger
danger
It's not surprising that games boil down to so few basic patterns. After all, as cavemen, we needed to be able to recognize food or danger under widely varying circumstances.

This drawing summarises Raph Koster’s view of computer games—an intellectual exercise at heart. He believes that by being aware of this “fact,” people can use games a lot more “usefully,” and they can become much more widely appreciated.

One game that somewhat “proves his point” is a Nintendo DS game series, entitled “Brain Training.” Not only has this become very popular in Japan, but has also begun being used in hospitals as a method of actually suppressing Dementia!

Bibliography

A Theory Of Fun, Raph Koster, 2005 (http://www.theoryoffun.com)
The Game Inventor's Guidebook, Brian Tinsman, 2002
Edge Magazine, Future Publishing, 2005, 2006
Game Architecture And Design, Andrew Rollings, 8 December, 2003

Web

What Warioware Can Teach Us About Game Design, Chain Gingold, October 2005 (http://www.gamestudies.org/0501/gingold/)
Gaming and Playing: Reflections and Classifications, Bo Kampmann Wather, 30 Jan 2006 (http://les.man.ac.uk/cric/gamerz/abstracts/walther.htm)
Playing Research: Methodological approaches to game analysis, Espen Aarseth, 2003 (http://www.msstate.edu/Fineart_Online/Backissues/Vol_17/faf_v17_n08/reviews/aarseth.html)
A Flexible And Expandable Architecture For Computer Games, Jess Plumber, December 2004 (http://www.gamasutra.com/education/theses/20051018/plummer_thesis.pdf)

Other

Warioware, Nintendo, 2003
The Games Factory, Clickteam, 1996 (http://www.clickteam.com/English/tgf.htm)