Game Design Course (March 2010)

Overview

On March 1st 2010, Planet Kodu's free five week Kodu course was held, and it was open to everyone that was interesting in building on their Kodu skills!

The course consisted of live meetings, videos, readings and tasks.

Please note this text has been lightly edited by Kodu Game Lab for clarity and to remove references and links to things that no longer exist.

Week 1 Getting Started with Kodu – Skills and game design

- Planet Kodu's concept of an informal design process
- An exploration of Kodu Game Lab, and what it can be used for
- Overview of basic programming structures and navigation

Week 2 Rapid Prototyping

- A more formal approach to game design is introduced
- A look at ideas, iterative process, brainstorming, identifying risks and more!

Week 3 Game Mechanics – designing challenge and engagement

- An exploration of how to develop game objectives and rules, actions and secrets!

Week 4 The Game Experience – Theme and Story

- Planet Kodu's look at the Kodu game space
- A closer look at the Kodu characters and worlds
- Building atmosphere in your Kodu games
- Play testing your Kodu games, and reward systems

Week 5 Pushing the Boundaries – puzzles and probability

- An exploration of Skills vs. Chance, Head vs. Hands and Cooperation in Kodu games

Course enrollment:

To participate in the course, you'll just need to follow along.

About

By Richard, Planet Kodu

Before we begin…

The course was facilitated by Richard of Planet Kodu and Martin Jorgensen. It ran for five weeks and the outline of the curriculum is on the overview page. Each week consists of delivered content through text, readings, videos and required tasks posted on Monday morning each week but the delivered content is only part of the picture and we expect all participants will contribute knowledge and ideas. The course design has been highly influenced by the work of Stephen Downes and George Siemens and the desire "that the knowledge in this course emerges as a result of the connections among the [participants] … and is not some 'content' shoveled from experts to recipients"

Open to everyone

This course is free and open to anyone. We hope that young people will be involved in the course and therefore we request that appropriate language is used at all times. Thanks!

Interaction is encouraged

The course is designed to enable the course participants to share their learning and discoveries and build upon the learning and ideas of others. To achieve this users are expected to interact by posting reflections, responding to others, sharing content including games designed and developed. Although we will try to present a wide variety of information, ideas and material we know that there is other material in other places that will benefit the course and we hope that this will be unearthed during the course. We hope that participants will share content from other sources and introduce ideas from outside of the course.

We also hope that participants will tailor this course to their own needs and their own situation, that participants will feel free to pursue their unique interests, including modifying and creating their own tasks. When and if this happens we expect that this will greatly increase the knowledge about game design and development using Kodu Game Lab. We don't expect that everyone will be doing the same thing at the same time, rather we expect that course participants will organize and control their own learning while still sharing what they are learning.

But you can learn on your own

Everything you will need to participate in this course, besides a computer capable of running Kodu (sorry, you need to provide this yourself) is provided on this site. All participants have a space to post ideas, reflections and other material, to share links, updates and games.

We know everything you create outside of course won't be about Game Design and Kodu Game Lab, so we suggest specifying a tag or category and using it to identify the content of the course. #koducourse may be a good tag to use!

Why are we running this course?

Firstly, we are passionate about programming and game development and believe that Kodu Game Lab offers a powerful but highly accessible way to make interesting and enjoyable games. We'd love to see more games created and shared using Kodu Game Lab and we hope that this course will result in greater use and higher quality games. Secondly, we're very interested in how online learning takes place and we are interested in understanding how a connectivity approach can be used for scalable online self-directed/networked learning.

Let's get started!

Week 1

If you haven't read the before we begin page and the course overview please read it now.

Use the discussion forums to share links, write blog posts and give updates on your progress (of course you can use your own blog, so.cl and twitter if you like).

Content:

- Welcome to the Planet Kodu, Kodu Game Lab course
- What is Kodu Game Lab?
- Game Making with Kodu Game Lab
- What is Kodu and where do I start?
- The informal process of game design

External links:

Understanding Games 1 2 3 4

Task:

Create your first game.

Welcome to the Planet Kodu, Kodu Game Lab course

(Now hosted by Kodu Game Lab with permission)

We're thrilled to have you onboard!

This course is will take you through many aspects of game design using Kodu Game Lab, a programming platform developed by Microsoft.

We've scaled the materials in this course so that Kodu users of any level of ability can benefit from being involved. During this course, we'll explore the practical nuts and bolts of how to create a game but also discuss the broader theory of games design.

This course will, we hope, give you a greater appreciation for what a game is, and what Kodu Game Lab is all about.

Timeframe

We've designed the course to be flexible to suit your schedule, so you won't have to fit in with ours!

Though the course is over five weeks, we're not expecting you to be in one place at any particular time. Dip in and out of the material during the week, or take it all in at one time. It's up to you.

Giving something back

This is an interactive course, and we'll be counting on your participation!

Each week there will be challenges and requests for your involvement. You can post your thoughts about these in our official Kodu Game Lab Get Connected forum or the forums wherever this material is being taught.

Delivery

We'll deliver the course right here on the Kodu Game Lab website, as Planet Kodu is **no longer in service**.

Of course, you can share your game designs or responses to the course challenges on the website! If you'd prefer to use your blog to reflect on your participation in the course, and upload your game creations there instead of Kodu Game Lab, that's fine!

What do I need to get started?

You'll need Kodu Game Lab installed on your PC, and you can download the most recent version of Kodu here for free.

The Kodu client and Kodu Game Labs site use socl for authentication, so there is only one username and password required. **Existing users will need to create a new socl account** in order to take advantage of this feature if they do not have one. Once (re)registered, you can like and comment on worlds from the Kodu client. Do you have an old client? Don't worry. You should be prompted to upgrade when trying to log in. Feeling shy? No worries – anonymous world uploading is still supported.

What if I run into trouble during the course?

Problems with Kodu? Where can you turn if you need assistance during the course?

There are a number of ways in which you can seek out assistance during the course. Your first port of call should be the Kodu Game Lab community forums.

If you are having issues with the Kodu game client, such as crashing or bugs, contact our support email.

What is Kodu Game Lab?

Richard interviews Rachel Schiff of Microsoft FUSE Lab developers of Kodu Game Lab. Rachel discusses the history of Kodu and the short and longer term plans.

Rachel Schiff from the Kodu Team

Planet Kodu was fortunate enough to speak with Rachel Schiff from Microsoft's FUSE Labs about the development of Kodu Game Lab and the future plans in 2010.

- Watch the video on Kodu Game Lab's official YouTube channel
- Kodu Game Lab
- Posts tagged "kodu" at Microsoft Research FUSElabs
- Cracked Rabbit Gaming Tutorial
- Stuart Ridout's Tutorial

Game Making with Kodu Game Lab

Richard interviews Charles Howell, a respected Kodu Game Lab creator on the Xbox platform.

Game Making with Baaad Dad

Charles Howell (or Baaad Dad as he is known in the Xbox Kodu Game Lab circles) has been creating games with Kodu Game Lab since the launch of Kodu on the Xbox platform in July 2009. In this video, Charles discusses the strengths of Kodu, suggests ways to begin with Kodu and shares some of the games that he has created.

- Watch the first part of the video on Kodu Game Lab's official YouTube channel
- Videos of Charles' games can also be seen on You Tube.
- Inspector Kodeux
- Kolosseum
- Kodu FC vs. Kodu United

What is Kodu and where do I start?

What is it?

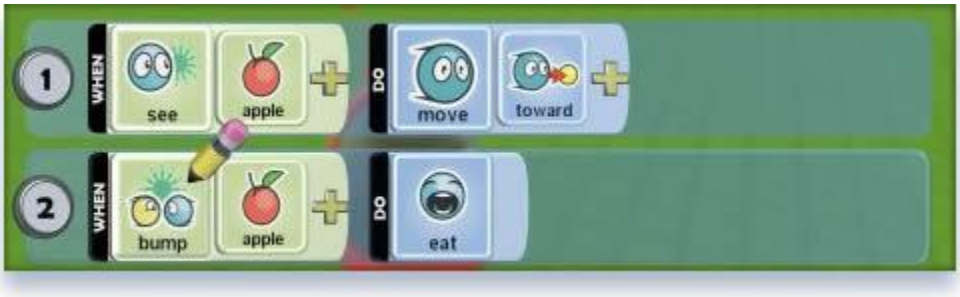
Kodu Game Lab is for building games!

Originally called *Boku*, Kodu is a programming environment designed by Microsoft for Windows and the Xbox 360.

Matthew MacLaurin, at the time director of the Redmond FUSE (Future Social Experiences) Lab and the game's creator, designed Kodu to allow children to actively engage with computers rather than experiencing them passively. Kodu enables you to build games using a series of visual elements in a rich 3D environment, without having to learn a single line of code. Microsoft describes Kodu Game Lab as being '… designed to be accessible for children and enjoyable for anyone.'

The core of Kodu Game Lab is its intuitive user interface.

From the interface, you choose visual commands to determine the actions of the characters and environment, rather than having to type lines of code to build your games. Think of the commands as rules that the game characters and objects must follow.



In the example above, whenever the character sees an apple, they will move toward it. At the same time, whenever the character bumps into an apple, they will eat it.

What isn't it?

While what you can build with Kodu is significant, there are limits to what you can achieve. Building games of expansive size with countless characters will tax the Kodu game engine.

Luckily, Kodu has a built in indicator to let you know when you're stretching the game engine beyond its capacity.

A thermometer to the right of the screen indicates clearly whether your game is becoming too big to manage. If it is, it's a simple matter of reducing the size of the game space, or removing peripheral characters from the game.

That's not to say that Kodu can't be used for building games of surprising complexity! What you can achieve with Kodu can be surprising.

Where do I start?

Start simple

There are a number of ways of approaching building a game, but our suggestion initially is to keep it simple. Our suggestion for your first attempt at building a game is to create a simple environment, and then program a bot (one of the characters you can place within the game) to perform some simple actions.

GDC 2009 review of Kodu by IGN

In this video review from GDC, made shortly after the release of Kodu in 2009, we learn about the basic principles behind the game.

Get some inspiration!

Watching what other users have created is always a great way to get ideas, and see how they have approached the game.

RPG Example

Download and edit

Download some of the games already created on the Planet Kodu website. You'll also want to try the dozens of games that come preloaded with the Kodu Game Lab software. Play the games, look at the way the characters have been programmed, and then try to replicate it in your own game.

The informal process of game design

There are a number of ways you can approach game design, but often an informal approach is the most suitable method to begin with.

Developing a game idea

There are lots of ways to uncover great ideas for game designs. We've outlined a few approaches below that might help inspire you!

You could attempt to replicate an existing game, like Pac Man, where the rules and game design are predetermined. This gives you a framework within which to operate right from the start.

You might also consider replicating an environment that all game players would be familiar with, like a maze. There are numerous elements about the game that you'd still need to determine. Is it a maze from which you have to escape, or one in which you need to reach an item in the middle? Will there be creatures or objects to challenge you as you progress? Whatever decisions you make, the basic concept of how a maze works will offer you a starting point.

You might use the characteristics of a familiar physical game as your starting point. Take tag for example. Numerous games are built around this simple schoolyard activity. Because it is a game that everyone is familiar with, no matter what incarnation you choose, your audience will have an immediate way of engaging with it.

If it's your first attempt at a game with Kodu however, you'll want to keep it simple, and *perhaps aim to achieve one or two simple tasks*.

For example, you may decide that initially that you want to simply build the topography, and insert a character that wanders around the landscape. Once you've achieved that, you might add another layer of complexity, and then another.

Testing

Testing shouldn't be something you do at the end of the game creation process. Making alterations, and then testing to ensure they have worked as you intended should be done as you progress, not at the end of the process.

Try to achieve too many new things at once in your game, and you may find it difficult to rectify your problems all at once. One of the main reasons this is the case, is that your character interactions become unclear. Is the Kodu bot moving because he hears or sees something? Or is he simply moving of his own accord? If you have too many new elements introduced into the game at once, you may find it difficult to determine whether even simple improvements you've made have been successfully achieved.

Getting other users to test your games for you is another solid step in this process. Other users will have different approaches to game design, and will have their own solutions to problems you encounter that you may not have considered previously.

Refinement

It's a good idea to offer up progressive versions of your game creations for the assessment of your peers. With each new version you can more easily see where improvements have been made, and how successful your updates have been.

You can revert to previous attempts if you have to, and try different approaches to find out what the Kodu community thinks of your improvements.

Other users will also benefit from being able to keep track of your progress and understand how you improved your games, enabling them to replicate your achievements in their own games.

Task 1

Your challenge for this week!

This week we're asking you to come up with a game.

The rules?

You can spend as much preliminary thinking and exploring time in Kodu as you wish, but when it comes down to actually building your game, set the clock, and stop when your hour is up!

Now while that might seem easily done, consider how fast time can fly when you're engaged in this sort of task. You'll want to have a clear idea of what you want to achieve, and some idea of how to approach it.

It doesn't matter if your game is unfinished, we want to see them - warts and all!

Upload your games to the Kodu Game Labs worlds page, identifying it as a 'Planet Kodu Course Challenge'. The Kodu client and Kodu Game Labs site use socl for authentication, so there is only one username and password required. **Existing users will need to create a new socl account** in order to take advantage of this feature if they do not have one.

Alternatively, you might decide to upload your course challenge attempt to your own blog or web page. Don't forget to include a link in the comments section of the forums so we can find it and comment, or download it for ourselves!

We'd also love to hear how you approached your design, what you felt were the more difficult elements, and how you resolved them.

Let the games begin!

Week 2

Content:

- Rapid Prototyping
- Video: Designing Kodu Games
- Brainstorming
- Identifying Risks
- The 400 Project
- Design Patterns in Games
- Kodu Game Lab Recipes
- Week 2 Task

Rapid Prototyping

The more formal approach to game design

Iterative process

Last week we looked at the informal approach to game design. This week, we take a look at a game development approach that's more formal, with clearly outlined steps designed to encourage you to consider your game design more broadly.

An iterative process, is a commonly adopted cyclical method you can use to approach designing a game.

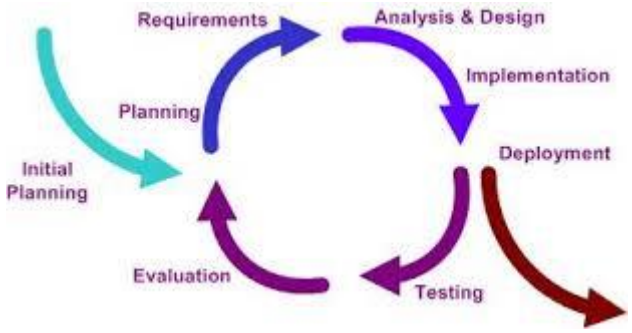


Image from Wikipedia – a typical Iterative Development Model

1. The Planning and Requirements steps are a mission statement of sorts. You'll need to establish the known goals for your game, what you'll need to build it.
2. Next, you design your game, perhaps drawing a paper prototype of it before you start building.
3. Once built, you test the game.

Seems a little like the informal design we explored last week doesn't it?

Well yes, it is like the informal process, except for a few key factors.

1. Firstly, you establish clearly up front what your goals are, and then with each round of testing and evaluation, you return and revisit those goals. There's still a healthy amount of trial and error involved, but this method allows for more formal reflection.
2. As each cycle is completed, you further refine your statements established in the first step. This is often necessary as your hopes for how the game will function, and the reality of how the game actually plays, are often a little different. You may find that as you progress, some of your initial aspirations for the game are either unnecessary, unachievable, or require a number of improvements.
3. With a more formal approach, your known goals will always been front of mind as you progress with development. Not only will you be considering whether they are being met with each new cycle, you'll be determining your games re-playability each time you begin testing.

How to Prototype a Game in Under 7 Days by Kyle Gray, Kyle Gabler, Shalin Shodhan, Matt Kucic

Whether you're taking a formal or informal approach to design, building something quickly introduces challenges that will be present with either method. If you have a time limit ahead of you, like our challenge last week, there are a number of factors you'll want to consider:

a. Choose a theme that underlies your game. It may be for example, that a version of the universally known schoolyard game of tag is the underlying structure, predator and prey. Or, you might decide that the game centers on audible triggers. A theme will help give your game the underlying structure it needs to be engaging, and you'll save time during the games development if you identify it early on.
b. Don't dwell too long on an idea that's not working out – try something new, start something different and come back to your original idea later.
c. Just because you want to build something quickly, doesn't mean you can skip important steps. If your game hasn't been well thought out, if the actual experience of playing the game is less than engaging, no amount of mood lighting, music or clever use of colors will hide that fact.
d. Just because you want to build something quickly doesn't mean you should ignore the mood lighting, music or clever use of colors! A game that looks and feels polished will be more absorbing. It won't save a badly designed game, but it can greatly enhance a well-considered one.
e. Embrace failure! We learn from our mistakes, it's as simple as that. The more you embrace your failures as golden opportunities for learning, the faster you'll develop as a game designer.
f. Never underestimate the power of the simple game. Your game doesn't have to have a thousand different variables to be fun. Pac Man and Pong are great examples; despite having been around for years now, new versions of these games keep emerging because they are simply engaging … or is that engagingly simple? Simple games can be completely and utterly absorbing, it doesn't have to be complicated to be fun.
g. Having an end goal will not only make your game more engaging, you'll find it easier to work toward a target. I've noted a number of RPG attempts in particular built with Kodu that have suffered from this problem. You wander here, you wander there, with no clear idea of what you should be doing. Sure the terrain and mood lighting are impressive, but what am I doing here? People get bored fast, so make sure the goals are clear up front.

Video: Designing Kodu Games

In part 2 of our interview with Charles Howell, we discuss the process of making great games with games, specifically the process that Charles himself uses.

Brainstorming

You can't schedule creativity, there, I've said it. You can schedule the work that goes into building an idea into a solid workable prototype, you can schedule the different aspects of the development cycle you intend to follow … but coming up with new ideas?

You might find these are best uncovered in the traffic on your way to work, while your mind wanders while doing the dishes, or reading a book before bed. Personally, I find with Kodu Game Lab, it's when I'm pulling apart somebody else's game that I'm most often inspired by a new idea.

And where does brainstorming fit in? Well, there are a number of ways you can try to kick start the creative process. It might be simply by messing around with Kodu, or you might find making a list of the things you want to achieve in your game brings out some ideas. There are numerous approaches to brainstorming, the trick is finding the one that works best for you.

Here are a few suggestions to get you started:

Automatic writing

Simply start writing as quickly as you can, all the things you want in a game. Don't stop for spelling or punctuation, don't stop to consider or flesh out your ideas. Just write as fast as you can, everything that pops into your head on the subject. Set yourself a time limit, a couple of minutes perhaps. Now, stop writing, sit back, and put a circle around anything that catches your eye.

Switch off

Sometimes just switching off for a while is the best way for new ideas to spring forth. If you find the new ideas just aren't coming, take a break and go for a run, have a bath, and give your mind a rest for a while. You'll be surprised how often getting off the task for a short time can be the best way to inspire a new idea!

Messing about

Playing with Kodu with no clear goal is another way you can find inspiration for a new game. Don't head in with any clear intentions in mind. Simply start up Kodu, and start building! Try things you'd normally not consider just to see what happens. You can discover a lot about Kodu when you're not placing yourself under any pressure – and you might just find a new idea will surface from all your unplanned messing about.

Paper prototype

Sometimes using a different medium can a great way of inspiring new ideas. I know for example, that I approach problem solving differently when I'm typing, as opposed to when I choose to sit down with paper and pencil. As I tend to type very fast, using paper and pencil forces me to slow down and more closely consider the language I'm using and the ideas I'm exploring. The same applies to designs on paper with Kodu. Draw an approximation of your world and then start putting in characters and ideas. Once you're done, try recreating it in Kodu.

Identifying Risks

When attempting to create any game it is important to consider the design risks and this is especially important when creating games with Kodu Game Lab? Is this game technically possible to make with Kodu Game Lab?

Is there any aspect of the planned game that is not possible to make with Kodu Game Lab?

Can the resources required be kept to size and level of complexity to make it playable with Kodu?

Can the object or bot be programmed to act in a required way?

Can a world be created to in the required form?

When generating ideas for your games it is useful not to be concerned with reality but at some stage it needs to determine if it is actually possible to make the desired game with Kodu. Obviously, for some game ideas that are variations of other Kodu it will be relatively safe to assume that the game will be possible to make. Other ideas, say, creating Second Life with Kodu, will obviously be impossible. So what can we do to minimize design and technical risks?

The Process

1. Identity the major technical and design risks of your design

Once you have decided on the game that you plan to create is important to identify the parts of the game that might not be possible.

2. Build a prototype of the risks

Start by building the parts (the risks) of the game that you feel might not be possible. It is not necessary to build a complete game at this stage or get this part of the game exactly right. This stage is all about determining if this risk is real or not. With Kodu it is not only necessary to determine if it actually possible but also necessary to determine how many resources this part of the game uses.

3. Test

Play testing is the most important part. Testing at this stage only tests the specific identified risks.

4. Modify (or abandon) as needed

Once you have evaluated the risks and determined what is possible, you can either proceed as planned, modify your design with regard to the discovered limitations or abandon the project.

Spending time identifying the risks in your game design will save both time and frustration when making larger and more complex games with Kodu Game Lab.

The 400 Project

The 400 Project is attempting to find and list 400 (just a rough number) rules that apply to Game Design. The project began in 2003 and so far has compiled more than 100 rules. The project is managed by Hal Barwood and Noah Falstein. Most of the rules have been written by Noah Falstein, although anyone can submit a rule.

Each rule has a name, a statement (description) and a domain to which the rule belongs. Each rule has an example or two from well-known games as well as counter-examples that show the consequence of not following the rule.

For example:

Provide Clear Short-Term Goals

Always make it clear to the player what their short-term objectives are. This can be done explicitly by telling them directly, or implicitly by leading them towards those goals through environmental cues. This avoids the frustration of uncertainty and gives player's confidence that they are making forward progress.

Noah suggests that rules are useful when you are stuck with your game design or evaluating the game you are making against each rule in the list.

Good Game, a TV program in Australia, ran a short story on the 400 Project, last year but the clip is no longer available on YouTube.

During the coming weeks we'll look at some of these rules and how they apply to the game experience. We would love to hear your thoughts on these rules.

Design Patterns in Games

Patterns are used by designers to solve problems using specific known solutions and that have been previously proven to work. In 1977, Christopher Alexander wrote the book "A Pattern Language" that outlined how patterns could be used in architecture to enable anyone to design buildings. Design patterns were popularized by the Gang of Four for use to solve common object oriented programming problems.

Design patterns:

- give a general solution to common problems
- are universal to any programming language/environment / framework
- outline the reasons why the pattern is appropriate
- create a shared language and understanding

Staffan Björk & Jussi Holopainen have detailed about 300 game design patterns on their website and in their book. The website only has a basic description, an example and the relations for each pattern but it is still very useful.

Example from the website: Boss Monsters

A more powerful enemy the players have to overcome to reach certain goals in the game.

Sometimes defeating the Boss Monster can be a goal in itself, but usually Boss Monsters are used as sub goals in the game and the high-level goal is of another type of goal. Boss Monsters are almost always used to structure the progress of the game.

Game Design Patterns can positively influence the design choices of game designers by enabling the games to have "authentic game experiences". In particularly investigating the patterns for narrative structures, predictability and immersion, such as tension or planned character development, may be useful in generating ideas for your games.

Many of the patterns and descriptions may seem obvious and some of the patterns may not be applicable to Kodu Game Lab however a deeper knowledge of the design patterns in games will undoubtedly help you create better games.

In weeks 3, 4 and 5 of this course we will unpack some of these design patterns in greater detail looking at how they can be used to build the game experience.

Kodu Game Lab Recipes

(and other ways to make life easier)

While Kodu Game Lab's visual programming language is rather intuitive there are a number of ways that you can minimize the learning curve.

1. Modify or build upon existing games.
The simplest way to create a game with Kodu Game Lab is to modify or improve an existing game. Any Kodu game played can be edited and modified or improved upon make it better.

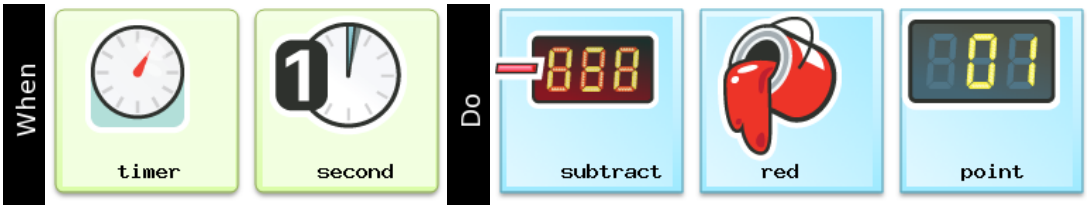
2. You can look at the code of any game and see how it was done.
Alternatively you can build your own game from scratch but learn from others by examining their code. Sometimes it may be obvious where the code from the game is located and

3. Copy and paste code from one project to another.
Kodu allows you to copy and paste objects and bots from one project to another. This enables you to mix and match from different games to create your new game. You can also use the clear tool to remove bots and objects from an existing world that you plan to use as a basis for a new game.

4. Use programming recipes to implement specific elements.
Programming recipes are code snippets that perform a certain task. Recipes are usually short language specific code that achieve a specific task. Often programmers would intuitively know the code to achieve these relatively simple tasks however when starting out, are having understood the basics of the language, recipes are an easy to build your knowledge of syntax and common techniques.

For example, a recipe to add a countdown clock to your game.

To create a countdown timer for 10 seconds use the following Kode to an object that won't be destroyed during the game! Also set the red score to Quiet.



In most common languages cook books of language specific recipes are available on websites and in book form.

Week 2 Task

This week we're offering the choice of two tasks depending upon the time you have.

A. Improve an existing game

Find an existing game, whether it be a game you have previously made, a default game that is bundled with Kodu Game Lab or a game you've found on the community website and then improve it. This may include adding new bots to the game, expanding its scope or improving the AI and game play, it's up to you.

OR

B. A more complex game

This week, with a more formal strategy in game design in mind, we're asking you to take the next step in the development cycle. Using the steps outlined this week, we'd like you to write a list of achievable requirements for your game first, perhaps a mission statement, make a few notes about the design, and then follow the iterative process outlined in this week's course guide.

There are no time limits on this week's challenge. You simply need to submit your game, finished or unfinished, with your notes. It doesn't matter whether your notes are a few simple bullet points, or a considered paragraph or two. What's important is that you have reflected on the development of your game more strategically.

Oh, and it's worth noting that we're particularly interested in your failures! That's right, if you've found little success on your initial attempt, we still want to hear from you, so we can all learn from each other's mistakes!

Week 3

We encourage you to use the official Kodu Game Lab site to share links, write blog posts and give updates on your progress (of course you can use your own blog, so.cl and twitter if you like). The Kodu client and Kodu Game Labs site use socl for authentication, so there is only one username and password required.

Content

Game Mechanics – designing challenge and engagement

Developing an engaging, exciting game is difficult if your objectives are unclear. You need to know where you're going! To ensure immersive game play, in most cases, you'll need to dangle a carrot in front of your player and offer them something to strive for.

Character driven

If your game is driven by an ongoing narrative, then a character driven goal is an obvious option when setting goals in the game. Your players might be set the task of finding hidden apples for Kodu, defeating combatants for bike bot, or catching fish for the tug bot. Perhaps there's a choice of tasks, or a succession of more difficult requests. Trying to fulfill an objective set by a character in the story can be an effective way of delivering the task, and it offers the player a degree of agency in the game by giving the impression, real or imagined, that they are choosing to help.

Ticking clock

Think about the number of movies you've seen where time is the thing that the characters must race against to complete their objectives. A ticking clock is also a wonderful way of adding suspense and increasing the tension in a game.

Simply achieving a game objective can be enough, but add a ticking clock in the corner and you add another layer of difficulty and engagement. The other great thing about using time as a secondary objective, is that you can up the ante, perhaps by reducing the time available to complete a task as the game progresses. A time limit will push the player to improve their game skills as the game progresses and keep them coming back for more!

Open ended

Your game may have objectives, but achieving them doesn't necessarily mean an end to the game play … in fact … your game doesn't have to end at all. Open ended games rely on the activity in the game being absorbing enough that you don't need to set an end goal.

You'll find it difficult to achieve this however, unless the game play remains fresh. One way to do this is continuously change the abilities of either the character or the opponents. Another way is to move the player between a numbers of different challenges in rotation. First I'm shooting fish, next I'm racing a cycle, and then I'm back to the fish again. If you focus on a different theme for each challenge, the player will have to refocus each time they move to a new game.

Object driven

Obtaining a difficult to attain object is a good way to place a goal in front of your players. You might surround the object opponents to build the game challenge, or alternatively, make the terrain the

challenge. A thin path you must stay on to reach your objective for example. It doesn't have to be in plain sight, but they will need to be reminded of the ultimate goal if it's not.

Team play

An additional character can represent your objective. Beat your opponent and win the game. You'll still need to establish something that both characters can strive to achieve, a particular score, collecting the most items etc.

Where two players are working toward a common goal however, your objectives can be a little trickier to establish. The focus with more than one character is usually working as a team.

One simple way to make this engaging is to clearly differentiate the characters. Giving them different abilities, different strengths and weaknesses will encourage them to rely on each other. You'll still need clear objectives, but with team play the aim is to put in place a set of achievable goals most easily gained when the two players work together.

Building intensity

We've all played games where the goals increase in intensity with progress. It's a tried and true method of ensuring that the player's skills are increasing as they move from one challenge to the next. It's also a good approach to draw a character into the game without overwhelming them at the outset.

With Kodu, this can be achieved in a number of ways. For example, with the achievement of a certain score, or once an objective has been met, a more difficult opponent is released into the gameplay. You don't necessarily have to build multiple game spaces in which to achieve this, simply make this 'sleeper' opponent part of the scenery until it's time for them to take part.

Another approach, and an obvious one, is the game Boss. Building a larger more fearsome opponent in the game that you must get past to move to the next level or complete a challenge is also a well-established way to develop intense game play.

You might surprise your players with this more difficult opponent after a set number of achievements. Alternatively you may elect to tease them with the notion of a difficult end game challenge, seed the idea early in their minds as something to work toward. For most players, the Boss character will be a solid inducement to play on as they'll look forward to an ultimate challenge.

Developing your game rules

Building a world with consistency

One of the more frustrating aspects of a game, can be a lack of consistency. If all the fish bots in the river are easily defeated except one, and you haven't established this quirk clearly up front, it can feel like a

bug. Setting clear rules about how your game will progress gives a degree of certainty to the player, and allows them to feel confident in the method of attack that they take.

Revealing the rules to the player as they become necessary is another approach to take. You may not want to hit the player with 12 different requirements at the outset, but release them slowly as it becomes necessary to know them. For example, if you have several game spaces with differing terrain and combatants, you may want to provide guidelines as the player moves to each new challenge.

Changing the rules

Of course, changing the rules on the player is also one of the best ways of adding a fresh element of challenge in the game. Set the rules, play the game, and then at a predetermined point, do a 360 degree turn and force your player to change their approach.

For example, Kodu may have to reach an apple at the end of the road, it's lined with bike bots that stand immobile. If he touches the apple however, they come alive, and become new combatants that Kodu must defeat. Changing the rules is also a great way of quickly establishing a new objective.

There are however, two things you'll need to be careful of when changing the rules.

1. The first is to notify your character that the rules have changed. You can do this up front, warning the player that performing certain actions will trigger a change in game play. Or you can do it at the moment of crisis, for example, when Kodu grabs the apple, it triggers a message that informs him of the new objective.
2. The second is to consider the other implications of a change in game play. Does your music need to change, adding a little tension? Do you need to remind the characters that shooting or running away is now required?

Secrets (And how to reveal them!)

Keeping secrets is something my six year old is becoming adept at, and she takes great relish in telling vital information to some members of the family and not others. Knowledge is power after all!

Earlier we talked about changing the rules, and revealing secrets about the game play is a great way to achieve this. Revealing for example, an opponent's weakness to rockets, can encourage your player to take a new approach mid-way through the game.

There are countless games that rely on this device to keep the game feeling fresh as the player moves through the set objectives. In RPG's for example, it's often a piece of the story, or characters history that's revealed. Often the information allows the game to develop to the next level, and is intended to offer the player an additional incentive to play on.

You can introduce secrets into you Kodu games in many ways. For example, items or opponents that reveal themselves when a particular score is achieved.

PLANET KODU

Video: Tavish Hill on Designing Kodu Worlds

Planet Kodu spoke to Tavish Hill but his webcam wasn't working so they only got audio. They added some of his Kodu worlds as an overlay to the video. Watch now!

MDA Framework

Marc LeBlanc has proposed the MDA Framework as a formal approach to understanding games; MDA stands for Mechanics, Dynamics and Aesthetics. The framework acknowledges that the game designer and game player have different perspectives of the game, with the game.

Rather than look at the model starting from the mechanics, the first stop for the game designer, we will explore the model from the point of view of the player, who cares less for the mechanics of the game (at least initially) and more for the aesthetics of the game.

Game Aesthetics

The game aesthetics are the *desirable emotional responses* evoked by the game dynamics; and it is important that as game designers we understand the emotional factors that make the game fun. To better understand what makes 'fun' fun, Marc describes (a non-exhaustive list of) eight kinds of fun that occur in games.

1. Sensation, where the *game is fun because the player is experiencing something new*.
2. Fantasy, where the game is fun because the player is caught up in a make believe situation.
3. Narrative, where the game is fun because the player gets lost in the game's story.
4. Challenge, where the game is fun because skill and trial and error is required in order to master it.
5. Fellowship, where the game is fun because the player is playing it with others and has a sense of community.
6. Discovery, where the game is fun because the player needs to explore the game and discover its secrets.
7. Expression, where the game is fun because the player is able to leave his/her mark on it and play it according to their preferences.
8. Submission, where the game is fun because the player can immerse themselves in it.

As game designers we can choose the types of fun that we want to occur in our game, and games can (and should) consist of multiple types of fun.

Having decided the type(s) of 'fun' that the game will focus on, we can investigate the game dynamics and game mechanics.

Game Dynamics

The game dynamics cover the process of the game that occurs in any given game session. The run-time behavior of the game can be somewhat predicted when forming the game rules and objectives however the dynamics cannot fully understood until the game is play tested.

- How did the rules create the fun?
- What patterns emerged in the dynamics of the game?
- Then lead to the question:
- What other settings, genres, design patterns or subjects might fit this game?

Here we look at the game design patterns, the 400 rules and other theories of game design that may assist.

Game Mechanics

The game mechanics cover the rules and concepts that formally make the games. The objects and characters that make up the game; and their attributes and states. Also included in the mechanics of the game are the code that make up the game and the rules and objectives that comprise the game.

Of course, the most important rule is the primary objective of the game, followed by the individual rules and constraints of the various objects and parts of the game.

- How is the character controlled? Can it jump? How does it react with other characters and events?
- What do the characters do?
- What different states do the characters have?
- What rules apply to the characters?

The MDA framework is useful for understanding what makes games fun and possibly more importantly may help as you design a game to consider the emotional response of your players.

Objects, Attributes, and States

When we are looking at game mechanics, we are specifically looking at the objects that make up the game. Anything that can be controlled, that reacts to an event, has a position are objects.

In Kodu Game Lab, objects and bots and objects and are added to the game using the object tool. Objects are assigned a starting position upon creation.

Attributes are the settings or information about an object, these may include the color, size and speed.

In Kodu Game Lab, settings can be set by the game designer but can't be altered during the game. Some attributes like speed (e.g. fast, fast) can be altered programmable during the game but not to the extent that they can be by using their settings. An option is to make multiple objects with different settings as creatable objects and then create instances of the object as they are needed during the game.

Objects can be static, in that they don't change during the game or they can have multiple states. Kodu Game Lab only allows an object to have states whereas other platforms may allow each attribute of an object to have various states.

Pages are the metaphor used by Kodu Game Lab to represent the various states of an object with an action triggering a change from one page to another. Object states are also determined by current circumstances of the object such as seeing, hearing or being close to another object. Sometimes it is appropriate to notify the player that the state of an object has changed by changing the color of the object or by playing a sound but on other occasions it is not necessary to signal this to the player. Notifying the player of too many changes to the states of the objects may not always be desirable as it may lead to confusion or the feeling of being overwhelmed.

When planning the mechanics of a game be aware of:

- The objects of your game.
- The various attributes of the objects.
- The various states of each object. Is the player to be notified when an object changes states?

Game play is determined by the mechanics of the game. Thinking about the objects and their attributes and states and how they influence each other is the most important part of a game.

Actions

The reason that games are often uninspiring and appear to be too similar to other games is due to the actions being unoriginal or derivative. Actions define what a player can do, how they control the objects in the game and how they can respond to happenings in the game.

The actions emerge as a result of what is happening during the game and the less predictable and wider in scope that the actions can be, result in a generally more playable game.

Ways in which actions can be used to increase playability:

1. Add more actions.

   If your characters can only perform one action e.g. jump or shoot consider adding more actions to increase the variety in the game play. By adding multiple actions you increase the options available to the player, of course it is possible to overwhelm or add actions that serve little or no purpose.

2. Allow actions to apply to more objects in different ways.

   If actions apply equally and the same way to all objects a game can quickly become repetitive and boring. Can the action that an object performs be used in a variety of ways? Games that focus on discovery or problem solving will particularly benefit from this approach.

3. Add more objects

   Some games allow the player to control multiple objects either concurrently or by switching between them. Some game allow multiple players to work together to achieve the game's goals, goals which are not able to be completed without cooperation. Of course, adding more objects to a game may add complexity that is not needed and will distract from the game.

4. Modify the effect of the action

   When an action is performed it has some effect on other objects in the game. Modifying the effect of an action will most likely increase the interest for the player. Whether the player is notified about the variable impact of certain actions is also worth considering and assessing how this will affect the game.

Questions to ask:

- How many actions can the player perform?
- How many objects can the player control?
- What attributes determine the playability of the game?
- What are the various states of each object?
- How do the effects of the various actions change during the game?

PLANET KODU

Rules, Objectives and Goals

Games have rules and the most important rule is the objective of the game, what the player must achieve in order to win the game. Rules define where players can go, what they can do, and how they interact with other objects and characters in the game. It is critically important that the player clearly understands the objective of the game, a game whose objective is unclear will never be a good game.

It is also important to include short term goals in your game that give the player a sense of progress and accomplishment.

Rules also must be fair, that is, all players should have the same chance of success, and if the fairness of the game changes or is different for different players then this must be clear in the rules of the game.

Rules should generally be clearly communicated so that players know what they have to do to win and why they may lose. In Kodu Game Lab, rules can be conveyed in the description of the game or they can be communicated using in game dialogs. For some games, the rules may be obvious for other games the rules will not be obvious at all.

Games can have different modes, where rules that apply in one mode do not apply in another. Usually these modes are time limited but can also be triggered by location or by certain events.

Good rules are:

1. Clear.
2. Achievable
3. Satisfying

Questions to ask:

- What are the rules of my game?
- Are the goals clear to the player?
- Are there short term goals which build interest in the game?
- Is the player able to choose their own goals?

Task 3

This week's task is to look at a game in light of Marc LeBanc's 8 Kinds of Fun, it is simple task given many of you will be working on your own Kodu games.

Choose a Kodu Game Lab game, either one you have made or one someone else has made, and identify the kinds of fun the game aims for. Assess whether you think the game is successful in this.

Week 4

Content

Game Experience: Theme and Story

We're more than half way through our course! Thanks to everyone that has contributed so far, we've really enjoyed the games you've uploaded, and the discussion, and hope it continues after the course concludes in week 5!

Kodu characters and worlds

There are dozens of different game genres, and one of the wonderful things about Kodu is that it allows you to explore most, if not all of them! Each genre offers unique advantages and challenges to the developer and player.

I've outlined a few of the better known game genres below, and some considerations to keep in mind.

Strategy

Usually turn based, games that require a strategic approach are perhaps one of the more challenging genres to approach in Kodu. Strategy assumes either you have an intelligent opponent, which either means you'll be building a multi-player game, or you're facing a challenge set by the developer.

You can achieve turn based reactions in a number of ways. You might use points, where different scores result in a range of reactions. Alternatively, you might determine that for particular behaviors on the part of your protagonist, specific reactions result, making the game more about devising a strategy to overcome these reactions.

Developing a strategy game is a real challenge, but worth your time as the game play can be absorbing.

Action

Perhaps one of the more common game formats for Kodu games, this style of game often challenges the player's quick reaction times. One of the things to consider in this type of game, is a unique set of properties for each adversary. A weakness and a strength will enhance game play considerably. For example, you might make the bike bot super-fast, but unable to traverse particular obstacles. You might make the tug boat slow, but with superior weaponry.

Considering a strength and weakness for each character will enhance the gameplay, and encourage you to think about their placement and position in the game more carefully.

Because you can also adjust the reactions of game objects (like a ball) as well, consider what their strengths and weaknesses are as well. For example, a ball that moves quickly over one surface, but slowly over another.

RPG and Adventure

You'll need to think about your story more carefully in this genre, as roleplaying games rely on narrative as well as some action to draw the player through the game. Good RPG games can strongly resemble interactive fiction.

In simplest terms, think about your character as the hero in the story. You might even write down some basic story structure to help you create an absorbing narrative. A simple three act story structure is the easiest approach, and you can find supporting material on Lightning Bug.

Another important point when considering the story you intend to structure your game around, is the concept of emergent narratives. This is when the story that evolves is one that occurs because of what the player themselves brings to the game. For example, if the player ignores your suggested path, and takes an unintended shortcut across the river.

You might wish to tightly control the experience of the player in the game in order that they follow your story as you've prescribed it, but keep in mind that the player and may have other ideas, and play in ways you may not have predicted. Providing some side quests is one way of approaching this, accepting that the player may never conclude your story, and prefer to take their own path.

Another challenge when developing a significant world and storyline can be that Kodu Game Lab struggles to cope with the scale of your game. In many of the RPG game examples I've come across, the usage meter is often near or in the red zone. To avoid this, consider what you can leave out of your game up front. This might be simplifying your approach to the topography and relying on moody lighting instead. Or it could be using one adversary with a unique strength and weakness rather than five that just point and shoot.

Construction and management

Games like these rely predominantly on collecting resources and building structures. You'll likely be using the factory bot in this type of game. Though the bots can't create other buildings, they can create large objects like rocks that can be used for all manner of creations. For example. Your player might be tasked with filling a space with rocks to get across it.

New gaming

Your game doesn't necessarily have to have a definitive end resulting in a win or loss. Take a look at the work of Tale of Tales for example. Their games are built for atmosphere and the experience rather than a focus on reaching an evident end goal.

This may be a little harder to achieve with the Game Lab, but you can definitely experiment with some alternate game structures … just ask my 6yo daughter who is obsessed with Kodu at present. Her worlds are an end in themselves. She simply likes placing things in the word that will respond to her. She stands

by the lake, she wanders through the forest, and she chases the clouds. You don't 'win', there are no points to collect, and she just revels in wandering about a space of her own creation.

There are no hard and fast rules about how games must be developed. One of the great things about Kodu, is that you can experiment with combinations of different genres to create unique experiences that you wouldn't encounter anywhere else!

Of course, Kodu has some restrictions, like any development tool, but there's plenty of room for a creative approach once you know what those restrictions are. Getting to know Kodu just that little bit better however, is the best way try something completely different. After all, you need to know the rules before you can figure out how to break them!

Building atmosphere in your Kodu games

Building atmosphere in your game might seem easy on the surface. A little dramatic music here, a dark forest there, and bingo! Instant atmosphere!

To achieve it with some success however, you'll need to develop a holistic approach to building your game, and not rely on one or two single elements to achieve it for you. There are many ways in which you can build atmosphere, and I've outlined a few ideas get you thinking about your own approach.

Character size

Given that you can adjust your characters size, and the size of the objects around them, this presents a wonderful opportunity for introducing atmosphere. Size is inevitably (no jokes here please) linked to power and the level of threat a character is confronted by, or holds over the environment around them.

A small character in a large dark wood immediately introduces the impression of vulnerability for example. Experimenting with your characters size and strength is one way of building atmosphere … without all the dramatic music and dark lighting.

Dramatic music and dark lighting

Of course, you can't go wrong with dramatic music and lighting … as they make a substantial impact on your game experience. Don't consider them an end in themselves however, think of them as simply another layer of atmosphere. It's worth noting that music in particular can be used to great effect if you position it carefully in the game.

Heading off to meet the big Boss? Dramatic music! Wandering over a bridge? Perhaps something a little lighter. You'll be the best judge of how to most effectively use these effects. Just remember that if you use them sparingly, but effectively, they'll have far more impact. Constant music of any kind can wear thin after a while.

Delayed gratification

No, this isn't another paragraph about size, it's about the completely absorbing nature of video games. Of all the forms of entertainment we indulge in, playing computer games demands (and gets!) the most attention from us. Numerous studies have suggested that we're far more likely to remain absorbed, despite distractions, in a good game than we are in TV or even a book!

Delaying gratification in the game is a great way of slowly increasing the stakes, building tension, and keeping your players hooked. Providing just the right amount of challenge to keep the player just out of reach of their goal. Just be careful to allow your audience some wins along the way or they'll eventually get frustrated, and move on to something else.

Color use

Effective color use will influence the mood considerably. Think about your best use of color however, after you've chosen your lighting. If it's a dark night, dark ground won't show up to best effect. Similarly, a dark bot will be hard to see, but then, perhaps that's what you're after? Just consider your lighting first, and then take the color of everything else into account.

POV

The point of view you take in the game can have a considerable influence over the atmosphere and tension in the game. Take for example, a space invaders style shooter like the one that Richard designed a few weeks ago. (Kodu Invader)

A fixed camera angle removes the distraction of a wandering point of view, and introduces the feeling that you can't back away from the action! It also allows you to see the entire playing field, and as you can more easily see the growing number of opponents you're faced with, increases the tension.

A first person perspective on the other hand, is a far more personal point of view, and perfect for RPG games where you want the player investing themselves in the outcome of their characters journey.

Video: Game Design with HaloX

HaloX has created a number of fantastic Kodu Game Lab games on the Xbox. In this interview he shares his game design process and rules for engaging play.

Videos of haloX's games are available on You Tube.

- **Dual** – Gameplay Video
- **Kodu: Portal** – Gameplay Video
- **Kodu or Die** – Gameplay Video
- **Take 'Em Out** – Gameplay Video

The Game Space

Games occur in a physical space. The space defines the playing field both in determining how the game is played and the feel of the game.

Open spaces can allow the player to explore and/or use strategy, while linear games have set paths that players must travel in order to progress in the game. Open spaces usually require secrets to be discovered or tasks accomplished sometimes the order of the tasks matters sometimes they don't.

Obstacles in the game space can be used to allow player to shield or hide from enemies. Barriers and walls can used be used to direct players along certain paths, as can raising or lowering the terrain.

Special Spaces, indicated by color or texture, can alter game play by changing the rules or effecting game play (e.g. the player moves slower in snow)

The camera, also affects the game space. A fixed camera above or to the side of the game space creates a 2 dimensional field but is often useful when creating multiplayer games. First person cameras are useful when creating games with mazes and surprises and much of the game space is often unseen by the player.

Levels

A simple way to give a sense of progression is to have levels in your game. It should be clear to the player that a level has been completed and that a new level has started. Typically a player is unable to return to the previous level once the next level has been started. Usually, a Boss Level is encountered in order to complete the game.

Characters

In Kodu Game Lab, the different bots do not all have the same abilities. Some can fly, some can swim and some can jump. The speed in which the characters can move is also different, our friend Kodu is much slower than the Biker Bot. In some games changing character mid game may be required, when the player boards a boat to cross water.

Interesting games tend to have the characters develop during the game. They might gain additional skills or abilities in order to complete more complex game play. Increasing the skills over time will not only add to the sense of progression within the game but also makes it easier more the player to learn how to play the game. Sometimes these skills will be automatically added to character and sometimes the player might purchase extra skills or tools, predicting what will be needed for the upcoming tasks.

What happens when the characters die?

How many lives does a character get, does the character return to the start of the game or re-spawn at a previous checkpoint? Having a limited number of lives and forcing the player who has "lost" to re-start the

game and then re-do easy parts of the game can make a game feel tedious. If completing the game, rather than say getting a high score, is the main objective then letting the player re-spawn an infinite number of times may be desirable to maximize fun.

Week 4 Task

Create a game that features a Bot that you haven't yet used before in a Kodu Game Lab game.

We encourage you to use the official Kodu Game Lab site to share links, write blog posts and give updates on your progress (of course you can use your own blog, so.cl and twitter if you like). The Kodu client and Kodu Game Labs site use socl for authentication, so there is only one username and password required.

Week 5

Content

Original Final Week Message from Planet Kodu, 2010

It's been a wonderful five weeks running the Planet Kodu course, and we've learnt a lot, observing the games and unique approaches participants have taken. We hope you've gained some new programming skills, have a clearer idea of how to approach game design, and have enjoyed the experience!

We'll be leaving the course material online as a resource for the rapidly growing number of Kodu Game Lab for PC developers, and hope it will continue to be of use to the broader gaming community.

Thanks for playing!
Martin Jorgensen & Richard Olsen

Pushing the Boundaries – Games of skill and chance

Skill

Games of skill challenge the player with clear goals that demand a competent technique in order to win. An accurate aim for example, or the deft control of a joystick to maneuver around obstacles. These games often have a clear challenge and will slowly increase the difficulty, requiring the player to hone their skills.

Kodu is a wonderful platform for creating games of skill. With endless opportunities to reinvent the topography you can create narrow ledges, steep inclines, or simple shooting galleries with ease. Being able to adjust the behaviors of the game surface adds another interesting twist. For example, if you make the blue ground slippery when the player has to shoot a target it increases the challenge.

Considering the strengths and weaknesses of your game bots as we discussed last week, can add an additional element of skill to the this type of game. With bike bot, moving at speed is an obvious skill to master for example. Increase the speed, and you increase the level of difficulty.

You can easily achieve this slowly increase in challenge using point scoring system. For example, when your player reaches a particular score, the opponent bots recognize this, and move to an alternate set of behaviors where the speed at which they move increases.

Adjusting the friction, the amount of bounce, or the player's ability to jump may also be things you consider to add an element of risk and difficulty for the player.

The great thing about games of skill, is that they are often simple in premise, and yet completely absorbing. Players can become obsessive about games of skill, particularly when the goal is clearly in sight. It is the players own level of accomplishment that stands in their way of success, and for competitive players, competing against themselves is the best challenge you can offer. Having an evident point score on the screen can add to this atmosphere

Games of skill may still offer strategy and an element of chance, but move too far away from the skill based requirement and you lose a critical piece of what makes games of skill so addictive.

Chance

When you introduce an element of chance into your game, what you're really doing is offering the player something that they cannot easily predict, or that is impossible to predict. There may be an element of skill or strategy involved in a game of chance, but it is the unpredictable that keeps us coming back for more.

When a game offers the chance for you to win … if you're lucky … the gambler in all of us gets hooked. It's the thrill of the unknown that draws us in, and keeps us playing. Take snakes and ladders for example. Big opportunities to stride ahead of competitors or quickly lose ground make the game exciting.

There are a number of ways you can approach a game of chance in Kodu, and recreating a traditional board game where chance is the main ingredient is one obvious path to take. You can however, find the unpredictable in a game in other ways.

Drawing on the 'wander' movement of opponents in the game for example, make their placement on the board difficult, if not impossible to determine, particularly if they are moving quickly. Adding additional commands to the bot that further compound this unpredictable nature needs to be done slowly to ensure you get just the right balance in how they react, and the threat that they are intended to offer.

The obvious other way of introducing chance, is by using the random point feature, and have the characters behaviors assigned to different point scores. Using the programming 'pages' to best advantage here can allow you to create some very interesting behaviors in your games.

Who are you building for?

The funny thing about games of chance, is that even if the outcomes are entirely random, even if you've warned your players that planning a particular approach won't help them, some players will still look for the best strategy.

Think about players at the roulette table. On some level, they know the game is purely chance, but it doesn't stop them looking for a strategy to try to gain a favorable outcome. Players that love strategy will always look for the best advantage, even when the game is determined by a roll of the dice.

Similarly, in many circumstances, a player that loves skill based games may seek a slender advantage through that ability, despite the outcomes of game they are playing being entirely determined by luck.

You can create a great game with a focus on strategy, skill or chance, but many games will include an element of all three. An element of the unpredictable, an advantage in taking a strategy, and a roll of the dice all bundled together make the game challenging on a number of different levels.

You may decide to make your game to appeal to players that enjoy all these elements in a game, but I'd suggest that leaning toward one of these three will help define the sort of game you're building, and help you better consider what sort of player your building your game for.

Knowing your limitations

There are limitations to Kodu, and when you know them, you'll find it easier to work within the restrictions that the Game Lab has. Some of these limitations are built into the game intentionally to promote more diverse and exciting game play. For example, bots have different abilities, different strengths and weaknesses. Once you know what these are, you can build games that leverage those abilities to best effect.

The game engine has a limit to how big or complex a world it can cope with, and this forces you to work with a more intimate game space. In the same way that telling a story in ten words rather than thousand requires you to focus more keenly on each world and its value in the story, the same is true of this game

space limitations. You must weigh each element of the game carefully, assess its value, and determine whether it is adding to the experience of the game or is unnecessary or even detracting from it.

A good example of this were the soccer games blogged about on the Planet Kodu website. Some had peripheral characters on the sidelines that were there purely to add some extra color. You might say, that these characters added little to the experience of playing the game. They certainly didn't influence the challenge the game offered. It's undeniable however, that they added to the overall atmosphere, and were for me, one of the most memorable elements. As I've stated before, knowing what to include, and what to leave out of the game can be a delicate balance, but can make all the difference in terms of engagement.

Of course, any game you build will have rules and therefore you must set your own limitations. When the player knows the restrictions under which must play, the game play can become more intense. Knowing how many rules to include, and how much freedom you allow your player can greatly affect the playability of your game.

Finally, like any game environment, Kodu has its own quirks and character, and your best games will often be those that use to these unique elements to best effect. PlayStation games are in a completely different category to Nintendo Wii for example, both have their appeal, both have strengths and weaknesses and are satisfying because of them.

Puzzles

Many games use puzzles, game play that can be solved using deductive reasoning, in order to add variation to the game. Often puzzles are used to unlock doors to other game areas or open containers that contain equipment needed by the player.

When creating puzzles, the game designer must make sure that the goal is easily understood and that it is very obvious what the goal of the puzzle is and how the user can proceed. One way to ensure this is to make it easy to get started with the puzzle and give the player feedback on their progress.

It is also very important that the player feels that the puzzle is solvable, using multiple puzzles making the first one easy and then gradually increasing the difficulty in subsequent puzzles is a useful technique. Having said that it is important that we allow the player to give up if they no longer want to work on the puzzle.

Finally, if all else fails it is useful to tell the player the answer to the puzzle. Knowing the answer will help the player with subsequent puzzles and reduce the level of frustration with the puzzle and subsequently the enjoyment of the game.

Refer to the Puzzle Solving game design pattern.

Probability and Chance

Games need chance and probability to be fun. If a game played the same way every time then it wouldn't be worth playing more than once. Also if there isn't any variation in the gameplay then "working through" a level loses its fun because it is predictable and therefore boring.

In game making we use random numbers to add variety to our game play. Randomness may be used to determine when and where opposition characters appear or how the non-playing characters react to certain events. When using chance in our artificial intelligence we need to be careful that we get the level of difficulty correct, the way we determine how likely events are to happen may have a dramatic effect on the skill level needed by the player and therefore how fun the game is.

There are two methods we can use. The first is to look at the code and calculate the likelihood of various events happening, particularly looking at the unlikely events and how game play will be affected when they occur. Using a table may be useful in considering how two random events will work together to determine game play.

The second method is to play test. When play testing to determine how randomness affects your games you will most likely need to play the game a lot. When playing try to see how random events influence each other, if a series of unlikely events happen will this make the game either unplayable or too easy?

Determining the actual possibility of random events in your games won't necessarily help you understand how the random events in your game will affect the perceived playability of your game. The actual differences between various random events may not always be apparent, the subtle differences may either not be apparent or seem to be greater than they actually are. Getting others to play test your games and discussing how random events are perceived is the only way to determine how random events affect the player's mind.

Chance and randomness can be used effectively to increase the level of difficulty, and therefore the level of skill the player needs as the game progresses. Good games get harder and are perceived by the player to get harder as progress is made. Finally, chance and randomness should not be used in way that the player perceives its role to be too great, and therefore believing that luck not skill will determine success in the game. Game design patterns useful when using chance and probability in your games.

- Limited Foresight
- Strategic Knowledge
- Player Balance
- Memorizing
- Balancing Effects
- Risk/Reward
- Imperfect Information
- Tension
- Luck
- Limited Planning Ability