Kodu Curriculum: Xbox Controller

Table of Contents

Kodu Curriculum: Xbox Controller

Program Overview

This program has two components:

- A single in-class session to be held for all 4th graders.
- An 8 week, optional after-school program led by a Microsoft researcher. This program will meet once a week.

Goals

Our goal is to expose students to computer programming in a fun environment, get them excited about potential careers in computer science and allow them to create their own games.

More specifically, a

fter using Kodu students will:

Attitudes

- Have more positive attitudes towards computer science.
- Better understand the steps involved in creating a computer program.
- Have a greater desire to pursue careers and future classes in computer science.

Behaviors

- Improve problem solving skills, especially on word problems.
- Create increasingly complex games showing a deeper understanding for complex coding sequences.

Other

- Show evidence of practice perspective taking and empathy in game play.
- Work together to create innovate solutions.
- Both girls and boys will be excited about pursuing jobs in computer science.

Unique Emphases of the XBox Program

Math and reasoning: Kodu has many uses and this curriculum focuses on using Kodu programming sequences to solve simple word problems.

Journaling: Students in this program will keep journals. They will record information about games they would like to create, things they found difficult and what they liked. Each day, students will be encouraged to write.

Xboxes: Most educational programs to date are designed for PC's. Here we use Xboxes and encourage students to make multiplayer games. This will also encourage group work.

Equipment provided by Microsoft:

- 5 Xboxes equipped with Xbox Live Gold memberships.
- 1 controller per student
- 1 cell modem for networked connectivity

About Sessions

After School Program

Grades: 4-6

Class Dates: 8 Tuesdays starting March 2, from 3:30-5

The goal of the after school program is to track long term attitude and behavioral change with a smaller group of students. There will be 8 total sessions and students will be given a survey at the beginning and end of the program. The curriculum below describes sessions in depth. Sessions include studio pedagogy with mini-lessons interspersed to move the students along in their discovery of the tools while enacting collaborative, problem solving practices. Towards the end of the session, students will work in teams on a longer-term project to create a game. The end of the unit will culminate with student presentations of these games to peers, parents and Microsoft employees.

In Class Session

Grade: 4 (all classes)

Class Date: TBD, 2, 1 hour sessions for each classroom

Teaching programming concepts in a single session can be challenging but goals of a single session program should be modest. We suggest that single sessions aim to pique student's interest in Kodu and programming concepts. This curriculum is divided into two parts. First, we provide a tutorial and student activity in which objects and simple programming concepts are introduced. Next, time permitting, educators can use the second half of this curriculum 'Editing Your World' where landscape creation and editing are discussed. Both aspects of game creation are typically quite interesting to students and we suggest that both concepts be introduced time permitting.

Students will use a survey and solve a programming task in groups. This session will be led by a Microsoft Researcher and a classroom instructor. Two classes of fourth graders at Benjamin Franklin Elementary School will take part in the in class session. Students will be offered the opportunity to take part in the more extensive after school session.

Kodu Curriculum

This curriculum is designed to be a reusable curriculum at Benjamin Franklin Elementary. It is a resource for using Kodu as part of a class, after school program, or summer camp. It is designed specifically for the Xbox but can be modified for the PC.

Kodu Curriculum intended to:

- Introduce instructors and students to the program
- Get students to understand beginning programming concepts
- Suggest ways that Kodu might be used to complement traditional curriculum

Kodu Description

The core of the Kodu project is the programming user interface. The language is simple and entirely icon-based. Programs are composed of pages, which are broken down into rules, which are further divided into conditions and actions. The Kodu language is designed specifically for game development and provides specialized primitives (the nouns, adjectives, and verbs of the language) derived from gaming scenarios. Programs are expressed in physical terms, using concepts like vision, hearing, and time to control character behavior. Kodu can express advanced game design concepts in a simple, direct, and intuitive manner.

Teaching with Kodu

There are many ways that you can introduce Kodu, but in this curriculum, we provide simple lessons and activities. The class begins with more structured instruction and ends with sessions that are centered on game creation. The purpose of this session is to enable students to create their own games. During the last sessions they will work on these games and create here we suggest that you begin using sessions to teach basic programming concepts in Kodu and end using the sessions primarily for game creation. Again, during the last session, students will present their games in a public forum.

There are a couple routines and pedagogical practices we suggest in order to help facilitate the class and enrich the learning experience of the students. First, while explicit instruction about the tools and the criteria for final projects is important, you should also let students experiment with the tool—playfully. This means that they will make mistakes that they must problem solve.

About the Developers

Kodu was developed by a team of research programmers at Microsoft who are passionate about kids having fun and being challenged as they learn how to program.

For More Information

http://fuse.microsoft.com/kodu/default for more information about Kodu and its developers. Visit the Kodu blog to see what others say about Kodu or post your thoughts about the program.

To help facilitate this purposeful mistake-making, they should be continually reminded that they as a collective hold the keys to problem solving. “Three-before-me” is a technique used to assist students in collaborative problem solving conversations –simply put, if students can’t figure out an issue they ask three other students before coming to the instructor. Another method is writing problems on a “Post It” and pasting them to a collective space where students share and answer each other, much like a digital discussion board. As students answer questions with other “Post-Its,” the questions move to solved side of the board.

Lastly, studio pedagogy allows a space for students to discuss their work and receive feedback—this process is integrated throughout the game creation process rather than sequestering the activity at the end of a draft (as in the traditional writing process) or at the end of product creation. There are several ways of establishing this in the class. One way is to have students work in small groups in which each takes a turn discussing his or her project’s intentions, challenges, and successes while others give warm and constructive feedback. (Often, this needs to be modeled.) Another implementation strategy might include, regular whole class exhibitions during which students peruse the games at various stages in development and ask questions of other students’ projects (Again, this strategy would likely need to be modeled.)

Session Scope and Sequence Overview

Session 1	Intro to programming, adding objects, Animating characters, creating landscape

(What do you think about the heading: Basic Navigation and Kodo Principles)

Session 2	Creating Landscape
Session 3	Using the controller to move characters, creating paths, setting behaviors
Session 4	Making clones and creatables and establishing a second player
Session 5	Changing behaviors using pages, establishing and shifting perspectives
Session 6	Power ups, health, timer
Session 7	Scoring basics and to communicate
Session 8	Starting unique stories and characters
Session 9	Presenting your game

Session 1: Basic Navigation and Kodu Principles

Students will be able to:

- Navigate the Kodu macro environment and use the Xbox Controller
- Understand the foundational principles of programming
- Access the programming mode of Kodu, potentially adjusting simple code for a specific purpose

Survey—get to know each other through the survey

Before actually venturing into Kodu, students should take the **'Student Pre-Survey'** found in this packet. The data gathered from the survey helps the developers of Kodu, and it also supplies teachers with some useful information about the types of technology practices in which their kids engage. The survey might also act as a springboard for a conversation that allows you to get to know your students better. It may also help you break your students into groups.

Break into groups

Because students will work together on projects, divide the class into groups of 2-4. These are the groups that students will work with on their activities and on their final projects. You may want to change your groups as group dynamics unfold. Students can self-select groups or, if you feel it is most appropriate, you can divide the students yourself. Your class may divide naturally around grade or other factors.

Basic Navigation

Before having the students log on to Kodu, it is useful to quickly walk them through the game creation environment. Here are some of the main navigation concepts in the game:

Main Menu: This is the first screen you will see after loading Kodu. If you have played games before, the top menu item will be 'Resume'. Selecting this will reload the last world you had open and start playing (PlayMode). If you want to select a new game, you can Load World.

Load Level Menu: Across the center of screen is a list of saved worlds. You can scroll through the list using the left and right shoulder on the controller. Select the world you are interested by pressing the Ⓐ button. Although it is possible to create worlds from scratch, Kodu comes with a number of pre-built worlds. These worlds come with land and may also include pre-programmed characters. In this lesson we will start by modifying a pre-built world.

Play Mode: Each world starts in the Play Mode. You can toggle between Play and Edit mode using the back button on your Xbox Controller.

Edit Mode: To enter into EditMode, you must press the back button. EditMode is where you will probably be spending most of your time. You know you are in EditMode because you can see the Toolbar at the bottom of your screen. Navigate through the Toolbar using your right and left trigger. Here you can create the landscape for your game world, populate it with bots and program their behaviors.

Mini Hub: You can get here by pressing 'Play'. The Mini Hub will let you go back to the world you're currently working with, save your current world and any changes you made, load another existing world, start with a completely blank work or go back to the Main Menu.

Class Play

At this point, let your students explore Kodu Games themselves and allow them to do some game playing so they understand the controller. Observe what they are playing and take note of any navigation issues they are having. Regroup at the end of the session asking the students to Exit the game and turn off their screens in order for them discuss what they found in Kodu and tricks of navigation. Maybe create a list of favorites and list of tricks on the board to track the dialogue.

Exercise in Programming an Unsuspecting "Avatar"

An in-class exercise is handy to prime students with some idea of what it is like to program games in Kodu.

Materials:

- 3 red apples (balls or some other colored object will do)
- 2 green apples
- 1 bag
- 1 or 2 blind folds

Inform the class that in Kodu they will need to:

- Select a character(s) and make it behave and react in certain ways,
- Build an environment in which the characters operate
- Fill that environment with objects that the characters may or may not interact with
- Create rules and reactions that allow the characters to function in the environment

Ask for a student volunteer to act a Kodu character. Have him or her sit in a chair in the middle of the room and blindfolded. Then ask the other students how the room should be arranged to create environment of Kodu. Tell them that you have 5 objects to place in the environment (3 red apples and 2 green). Place them in different spots in the room. Inform them that we need to direct our Kodu character to find and pick up the apples. Note that every action must be outlined and described.

To help organize their "commands" inform them that the character only listens to you, so they need to filter the commands and ask for clarification should something be vague or difficult to enact. Depending on how energetic your students are, you may need to call on students. Keep asking questions about how, when, and where. Feel free to unblindfold the student if they are prompted to see (this is command in Kodu after all). Consider during the exercise that Kodu can move forward, backward, north, south east and west and that it can faster and slow. It can see, hear, and distinguish color. It can jump, turn and bump up against things. It can be programmed to move toward and away from certain objects. It can also express love, anger, sadness, and craziness. Kodu can say things through text as well. Kodu can even wait at places or move between places within a certain amount of time. BUT all these actions and reactions need to be programmed by the user. So as you take commands from the class, be sure to continually ask them how, when, why, and where in order to get increasingly detailed about the action.

A typical scenario might be:

Teacher: We want Kodu to do something that involves these apples. Let's try to just move our Kodu to the apple and put it in his bag. What would be the first we would tell it to do?

Student: Move to the apple?

Teacher: How? Does he walk?

Student: Yes, he walks?

Teacher: Fast or slow?

Student: Fast.

Teacher: How does he know when? What prompts him to move?

As the scenario transpires, it might be helpful to write the commands on the board, so the class can easily recall. Once a coherent set of commands are established, put your Kodu in motion.

Describing an object's program

To prepare for this lesson, the instructor will need to create a simple program in **Empty World.** Add several trees and a cycle with the following program: **'**When: always Do: move wander; When: see tree closeby Do: move avoid.' Rename this, **Demo1**.

Begin the class by opening the world you just created; **Demo1**. Ask the students to observe the cycle's behavior. Have volunteers explain what the cycle is doing.

Explanations might be something like:

- Cycle is moving around.
- Cycle is avoiding the trees.

The Object Tool on the ToolBar is the tool that you will use to add objects to the world and program them. To edit an object, you must hover over that object. Pick up an object using Ⓐ and edit its program withⓎ.

Now, show the students the program that is running this set of behaviors for the cycle. Ask the students to describe what the code is telling the cycle to do (it should be the same set of actions as above). After students successfully explain the code, ask volunteers to modify the code for the following set of behaviors. When appropriate, have the student who made the suggestion come modify the code in front of the class and explain why their solution did or didn't work.

- Add a tree to the world.
- Make this tree blue.
- Adjust the cycle's behavior so that it only avoids the blue tree. (Adjust the second line of code to read 'when: see *blue* tree closeby + Do: move avoid.')

Journal Reflection

After the students work through Activity 1, tell the kids to save their projects, and turn off their monitors. Explain that students keep a record of their progress in their journals. Students may not write about the suggested topics, but this is an important time for them to practice writing. Journal entry: What makes a computer game fun? Discuss journal entries. Also:

- What was challenging?
- What was easy?
- What was a success?
- What did they learn or new?
- What do they want to learn next?

Game Creation

In the first hour of each session, students will learn programming techniques. In the last half-hour they will work on their long-term projects. There will be a short portion of the curriculum each week devoted to these projects. The 'game creation' portion of the curriculum also devotes some time to less structured activity in class.

Start this game creation session by discussing the long-term project. Students should get familiar with their group. This is a good time to discuss the idea video games, like books sometimes tell stories. Ask students to describe some of the stories in their favorite video games. Tell them that when they create their game, it will also tell a story and they can decide what that story will be.

Next, ask students about their favorite books while defining some literary terms. For instance, depending on the age of your students consider reviewing

- Main character (protagonist)
- Opposing character (antagonist)
- Side (peripheral) characters
- Plot—the hook/inciting incident, rising action, and climax (which in literary terms consist of exposition, rising action, climax, resolution and conclusion.)

After or while you are talking about each of the above terms, project **Tedanoo v25** on the screen for students to identify the main character, the opposing character and peripheral characters. This game can be quite challenging so students might not be able to get to the end. They can 'cheat' by looking at the code or even altering the code to make things easier. Let a few different volunteers take a try. See if the students can identify the following literary elements:

Protagonist	Cycle 1
Antagonist	Cycle2
Side Characters	Saucer, Sputnik1, 2, 3 & 4 (note: These might not be considered characters because they don't move. You can discuss this together.)
Plot	
Exposition	Ask the students about the story before the game—how is the story set up? How and why are the characters in opposition to each other? What motivates characters? (Cycle 2 is trying to throw rocks at Cycle1. Cycle 1 should bump into the Sputniks because they create hearts which give him life. The mine also takes away health. The saucer creates hearts and mines.)
Rising Action	What moves the story forward? What events move the game forward? (Collecting hearts from the sputniks, the mine)
Climax	What seems to be the climax in this game? (Interaction with the mine)
Resolution	Does the game supply a resolution or conclusion? And if not, how might it? (If you score above 150 green points you win the game)

Group Activity 1: Eating Apples

Objectives
Add object, change color, select object, create sequential program for object

Directions
We just went over some of these steps. Now work with your groups to complete each of the following, check it off your list. Make sure each member solves at least one TO DO. Check in with your teacher once you are done or if you are having trouble.

To Do Check List

- ❑ Open the world ‘Small with water’.
- ❑ Add an apple. Make this apple blue.
- ❑ Add Kodu to your level.
- ❑ Make Kodu find the apple that you just added.
- ❑ Make Kodu eat the apple once he finds it. (Hint: You may also need to tell Kodu to avoid the water.)

Then, you are free to play. Try to add other objects to this level adjust Kodu’s behavior and change the environment. Remember to work with your team. Ask for help if you need it.

Challenge Activity

As a challenge activity, go to Tutorial 01 v3 and do what the Kodu asks. Also, see if you can reprogram the castle to behave in a different way once it is bumped, and try to figure out how the camera can follow Kodu on its trek to the castle.

Session 2: Creating a Landscape

Ask for student volunteers to operate several of the landscape features—some of them will have likely used the landscape feature during the previous session. Using the same level that was used in Activity 1; **Demo1**. Ask the students who volunteer to verbalize what they are doing so the class can hear their instruction. Volunteers should:

- Add/delete land to the existing landmass [Choose green paintbrush in the toolbar. Select the appropriate landscape material by pressing (Y) and selecting one from the toolbar. Next, select the brush shape (X) and brush size (D-pad). By using the left stick and holding down either the right trigger to add land or the left trigger to delete land, students can draw the landscape.
- Add land of a different color/texture around the perimeter of the landmass [Use same steps as before, only change the landscape material and perhaps the brush size. The speed of land addition or subtraction can be controlled by the degree to which the controller is shifted.]
- Create hills and valleys; use the smoothing feature [Select the raise the lower icon from the toolbar. Again, using the Brush Picker and Brush Size. The speed of land raising and lower can be controlled by the degree to which the triggers are pressed.] Also note to students that the smoothing out feature allows a less jagged landscape which also allows maneuverability.]

Subject Area Link—Geography

If you think it is appropriate, consider making a link to geography. Really amazing landscapes can be designed and demonstrated with Kodu. Depending on the grade of the student, you might ask them to illustrate different geographical terms: hills, valleys, mountains, peninsula, cliff, island, bay, isthmus, volcano, plateau, channel, etc. Kodu can also illustrate changes to landscapes: consider for instance, illustrating erosion, landmass changes do to earthquakes and volcanoes, glacier activity, etc. An interesting exercise for students may be to create a topographic map of a mountain(s), river, state, province or country, perhaps even illustrating the same land mass pre and post geological event.

- Create an island [As students do this, they may be tempted to create a separate landmass in the ether. In order to create an island their needs to be land between the two land bodies, acting as the lake, sea or ocean bed. Water cannot be added unless it has land to rest on.]
- Add water [Select the water tile and use the triggers to add or lower land.]
- Create a plateau and bumpy landscapes using the toolbar selections

What You Can Expect from the Student Activity

The landscape tools are particularly riveting for the students. Some students will easily spend hours creating and revising their environments to create a specific effect. In fact some of the kids will become so engrossed in their world that they will put their efforts in landscaping as opposed to character development, game strategy, plots and usability. This can be alright given the aim of the curriculum. It is important to remember that people tend to compose and create in different ways. Consider for instance, writing: Sometimes the intent of the author is for personal satisfaction over audience interest. This may hold true for the students while working in Kodu as well. If you find that this is the case, it would be good to discuss these approaches to composition and intent as a class once you have entered into discussions of characterization and plot.

Journal Reflection

In the second half of the course, students will be working with teams of 3-4 to create games. They should describe in their journals the mood that they want to create when they make their game. What mood do they want to make? Do they play any games right now that have interesting moods? Discuss entries as a group.

Game Creation

Students will have already written about the games they would like to create in their journals. Now, ask them to create sketches of these landscapes using pen and paper. They should create sketches in their journals. Ask them to select one team member's sketch that they want to actually use as their final game. They can always modify this. Motivated teams can start to create their landscapes in Kodu.

Team Activity 2

Objectives: Create land with texture, add water, trees, rocks, etc.

Making Landscape

Directions: We just went over the different tools for creating a landscape. It is your turn to try it out. As you complete each of the following, check it off your list. Check in with a teacher once you are done or if you are having trouble.

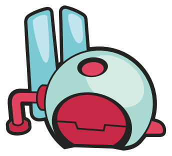
To Do

- ❑ Go to Demo1.
- ❑ Create a landmass with:
- ❑ At least two types of materials
- ❑ Create rolling hills, mountains (with a white peak), and valleys
- ❑ Make an island or two off the coast of your land
- ❑ Add water as either a river, lake, ocean or all of the above
- ❑ Create a magical forest somewhere in your landscape (You can define magical in any way you want. There are number of objects to choose from: trees, rocks, stars, coins, etc.)
- ❑ Create storm clouds over one part of your landscape.

Challenge Level

After you **have** created your world, see if you can find where to change the mood and tone of the **game**. Specifically, try changing some of the settings. Investigate the following settings and note how they change the meaning of your world:

- ❑ Wave height
- ❑ Water strength
- ❑ Sky
- ❑ Lighting
- ❑ Breeze



How do adjustments to the setting change the feeling of your world? Come prepared to share your world with your classmates.

Session 3: Using Controller to Move Characters, Create Paths, and Set Behaviors

Students will be able to:

- Use controller to move characters in a game world
- Create paths on which characters will move
- Give objects behaviors

Thus far in the lesson plans, students have not been asked to use the controller to make characters move. Some students may have completed the challenge in Session 1. In this activity, ask volunteers who have already programmed objects for user control to show the rest of the class how they solved this. Using the projector, ask for a student volunteer to demonstrate how they made Kodu go to the castle in **Tutorial 01 v3**. There are two ways primary ways to do this. In either case, the user must enter edit mode, choose the Kodu icon, put the editing ring under the Kodu on screen and press Y to program.

One way to move Kodu is using sensors. In this case movement is automated. To use sensors, the code would be

When: see, castle (or building) far away Do: move toward quickly

To control the movements of Kodu (or ‘drive’ Kodu), the programming sequence would be

When: gamepad, left stick Do: move, forward, quickly

If the student demonstrates one of the above methods, be sure to suggest an alternative method to show that the problem can be solved in multiple ways.

Also be sure to have the students read the code that precedes the new line that is entered. It is good for the students to practice reading the syntax of the code.

Next go to Idyll v6. This game has a number of components from the lesson. Instead of playing the game right away, have the students read the code collectively. Start with the Kodu—the code is fairly straight forward. (Note: In the PC version of Kodu, you may need to make several edits to clean up the code. The game changes over time but these tutorials are not necessarily updated.) While you can go to Page 2 and read the code, pages will be covered in the next lesson.

Now, run the game. Ask the students what is happening in the game that they didn’t see in the code. They will likely notice that the castles disappear and create wisps. They also notice the blimp moving. And they may also notice that points are accumulated—this is programmed through the tree. Go back and take a look at the code for each of these components. Of most interest to us for this lesson are the castle and the blimp. Scoring will be covered in Lesson 6. A wisp is also coded into the game as a creatable, but this feature will be covered in Lesson 4.

Read the code for the castle. Again, it is straight forward—when kodu bumps the castle it releases a wisp and it blows up. Next, read the code of the blimp—there is only a DO statement, no when. Ask the students what this means.

The idea of a path will probably be new to the students. Show them the blimp can be raised and lowered by "picking it up" and raising and lowering it using the D pad. Also note that the color of the path needs to be specified in order for the blimp to move along the path.

While looking at Idyll v6, ask them how they would create a second path for a hot air balloon. This is done by choosing path from the main tile selection. Then choose plain path. A ball will appear on the landscape, and use the left stick to press A to add another ball. Continue the pattern to create the path you want the balloon to follow. Color the path to distinguish it from the first path—if you want the kids to problem solve consider making the balloon code follow a different color than what has been created.

Subject Area Link –Science

The path feature and the coding tiles Move /Toward and Fast/Slow in Kodu could be particularly useful in animating biological and physical happenings.

Consider, for instance, animating the tribulations of fish or turtle during their migratory paths from sea (dodging debris and fishing vessels) to locks and fish ladders to natural predators. Concepts like over fishing might easily be put into a game format, too.

From a physical science perspective, Kodu might animate stellar patterns, trajectories or atom formation. Animating these concepts can help students to retain concepts with greater depth and complexity.

Journal Reflection

After the student activities, discuss as a class how students solved the TO DO's in their activities. See if they had similar problems and if solutions were found. Discuss how they solved problems—sometimes this entails trial and error, other times this includes looking at code in the other programs. This is also a good time to check whether they are working in groups. Afterwards, ask students to take time to write about what they learned during the lesson in their journals.

Game Creation

Students may have already begun to work on their landscapes for the game that they are creating as a team. Now it is time to think about characters and the narrative. Have them complete the following brainstorming task. They should brainstorm with their groups and they may want to write in their journals. They can start to add a program objects in their game once they have developed their plan but they must come up with the plan first.

Character Brainstorming

Antagonist (Main Character)

- Who or what is your main character?
- What motivates the character?
- What are its likes and dislikes?
- Does it have emotions?
- How does it communicate with others?

Protagonist (Opposing Character)

- Who or what is your protagonist?
- What motivates the character?
- What are its likes and dislikes?
- What are its emotions?
- How does it respond to the main character and why?

Peripheral Characters (side characters)

- What or who are the side characters?
- What role do they play—do they support the main or opposing character?

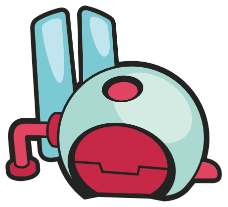
Task: After you have answered the above questions, start putting the objects in game you started creating earlier. If the setting that you made doesn't seem to fit your characters, revise it or chose a new landscape to alter or build.

Student Activity 3

Directions: We just went over the tools for making characters move. It is your turn to try it out. As you complete each of the following, check it off your list. Again, it is important that each team member solves one of these tasks. After you solve a task, alternate who is using the controller. Check in with a Camp Kodu advisor once you are done or if you are having trouble.

To Do

- ☐ Go to the landscape you created during the last session or to an existing game in Kodu. For simplicity sake, choose a world that has land of some sort.
- ☐ Create a character that the user controls with the controller
- ☐ Create a second character that has automated movement
- ☐ Create a path on which a third character moves
- ☐ Create an object that does something either when it is bumped, sees or told to do something using the controller or when it is programmed to do something automatically through a DO statement.



Extra Activity

After you have finished your TO DO for Activity 3, try your hand at 3D Flare Paths to see how to make 3D representation and interesting graphic effect.

Check out the action game Rock Fight v09 for an example of elevated paths. When playing and analyzing the game, think about how the path/ramps are used and add to the intensity of the game.

Session 4: Making Clones and Creatables

Students will be able to:

- Understand cloning and creatables
- Creating a two or more person game

This lesson will take advantage of contrasts between cloning and creatables, and it will use frustration as a key window to learning. Creating armies, groups, hoards, gaggles, swarms, etc. is easy in Kodu. However, giving individuals within the groups the same set of qualities is not readily intuitive and can be time consuming when using the cloning feature alone. Usually, students start by cloning and then soon become tired of programming the same character attributes multiple times should they want to edit the group's behavior.

The in-class lesson will make them use the clone feature for the blimps and then try to modify the behavior of the group—this is not easily done because for the five blimps that they make, they will need to change five different lines of code. As a response to this frustration, you will then walk them through how to use creatables in which code can be changed just once. Have the students run the controls as you talk about what to do. Follow the steps below.

You will need to create a new level 'Blimps and Jets'. To do so, create an empty world with a blimp on one side of the landscape and a jet on the other. Name this level 'Blimps and Jets'.

Cloning and Creatables

- ❑ Go to your new level, Blimps and Jets. The task of the class is to make a battle in which one player controls the blimp and a second player controls the jets. The two can either work together to defeat a common enemy or they can be in opposition. It is up to you.
- ❑ First, program the blimp. Ask a volunteer to put the ring under the blimp and look at the code. Program the blimp so it moves forward quickly by using the left stick.
- ❑ After you have created the blimp, make up to five clones of it using the right trigger when the blimp is glowing blue.
- ❑ Next, make sure that the blimps can move forward.
- ❑ Now give the blimps the ability to shoot blips. How many times do you need to change the code? Ask the class if there is a way in which to make it so you can just enter code in one location. (You can do this with creatables.)
- ❑ Demonstrate this by making a second set of blimps that operate automatically without user input. Have the student put the new set of blimps somewhere else. Make the blimp green or another color to distinguish it from the first set.
- ❑ Next, press X Settings and scroll down the list to Creatables. Select this option.
- ❑ Now, go back to the main programming state and clone this blimp. Point out that lines appear between blimps.

- ❑ Lastly, add code to the “mother” blimp, e.g., shoot blips at the jet when close by. Show the students how this code transferred out to all the blimps. When you play the actual game, the mother creatable will not be on the screen—it only appears during edit mode.

Two (or more) Players

Now that creatables are covered, recap the process by repeating it with the jet on the other side of the field. Clone the jet so that there are two. If you want, you can make it a creatable. At the same time, show them how a second player can be added to the game. Under the Y Program, select:

- ❑ (Jet 1) When: gamepad, left stick, player 1, Do: move forward quickly quickly quickly
- ❑ (Jet 2) When: gamepad, left stick, player 2, Do: Move forward quickly quickly quickly
- ❑ And so forth for additional players. Up to four players can be added when using an Xbox. Show the students how they can create a game that everyone can play. Pass out all the controllers.

Of Note

The use of creatables is powerful, but it can create an impressive effect of a legion of troops all moving the same fashion. Interestingly, however, it can also be used to create a more naturalistic look.

For instance, think about schools of fish with each school being its own creatable set programmed to wander. Then within each set, you can program a set of attributes unique to the school.

Journal Reflection

Consider the game **that** you will be creating for your final project. Describe the following characteristics:

- ❑ Setting (what does the landscape look like?)
- ❑ Mood (whimsical, dark/brooding, does the pace go up and down?)
- ❑ Characters (behaviors, conflicts, friends, alliances?)
- ❑ Objects (do the trees, buildings, rocks, etc. hold a special function?)
- ❑ Plot (how does the story progress?)
- ❑ Why would someone want to play the game? What will make it unique?

Game Creation

During the last ½ hour of class students will work on their games using their journal entries as a springboard. First they should discuss journal reflections with your team members. Does everyone have the same vision for the game? How can you revise your plan so that you incorporate everyone’s ideas? Spend some time making sure that your game incorporates some ideas from each person on your team.

Student Activity 4

Objectives: Make clones, understand the premise of creatables, and craft a game that everyone in your group can play.

Directions: Follow the To Do below and have your instructor check out our progress before moving onto the extra activity. This is a fairly open-ended activity so feel free to be creative in your use of creatables. Make sure you create an activity that everyone can play.

To Do

- ❑ In a world you have already crafted or in a new game all together, add two or three sets of characters—like we did with the two sets of blimps and the set of jets. (Creatables do not necessarily need to march in unison.
- ❑ Make sure that your program uses the move/wander code to craft a more naturalistic effect.)
- ❑ Establish a character or group of characters that are controlled by a second player. You will need a second controller to test the game out. Another person on your team should be able to test out the second player.
- ❑ Now make your game 3 or 4 players depending on how many people are in your group. You should have a game that everyone can play.

Extra Activity

If you finish and you have time to spare, check out Technique: Launching Creatables v02. Note how the apple has to be made into a creatable in order for Kodu to launch it. Take the apple away, and Kodu can launch nothing. Can you reason through why this would be?

Session 5: Starting Unique Stories and Characters

Students will be able to:

- Understand the idea of pages, when they are used and why
- Use nearby and far away features
- Shift camera angles in settings and in the code

This lesson is based on discovery. Have students play Vendura v14. After they have played the game for awhile, ask them to look at the world and see if they can determine how many objects and characters comprise the world and what each object or character does.

Here's a list:

Turtle—we control, acquires wisp shooting ability by interaction with Kodu

Stick—informs the turtle (the user) that Koduwan has been waiting

Kodu—instills power and disappears

Wisps—given to turtle as a power

Castles—unleash mines that shoot

Hearts—creatable that act as food

Mines—move randomly and shoot at Kodu

Factory—acts as the object that triggers the end

Pushpad—informs the turtle what to do

In order to get groups to cooperate, assign each student at least one object or character to evaluate. They should be tasked to reporting back to the group how the object or character is coded. Using the projector, have them present each of their findings. Pay particular attention to pages, and to how nearby, far away, and camera shifting are used.

To help guide the discussion, consider how in Vendura, objects and characters provide the player with an understanding of what's going on. They act as directions for what to do next. However to create the effect of shifting action, pages are necessary. Pages help shift behaviors either based on time or in reaction to some occurrence.

Pages

The stick directing the turtle to Koduwan is the first use of pages. Its use changes the state of stick from closed (page 1) which is its constant state to open (page 2) to express another set of behaviors which run and then return to the initial state of closed (page 1) after a certain amount of time.

The second use of pages comes with the turtle's approach of Koduwan. Page 1 supplies a resting state for the kodu. Page 2 activates a behavior—Koduwan turning and greeting the turtle. After three seconds, Page 3 is enlisted. It supplies another set of text to carry the story along. After eight second, a wisp is created and given to the turtle as Koduwan disappears in a boom.

Close By and Far Away

Point out how dramatic effect is created through this series of exchanges and behaviors. While pages help facilitate the operations, the 'close by' and 'far away' tiles allow users to create an element of surprise. In the entry sequence, both the stick and Koduwan's behaviors are guided by sensing the turtle nearby and then shut off based on the timer. This is effective for moving the story along.

Camera Shift

Also, point out how the camera perspective shifts when the turtle hovers on the red land square. When playing, this shift in perspective seems to be caused by the turtle's approach to Koduwan. This is an effective use of land and camera to jar the user into a suspended like state. Camera positioning is an effective tool for the creation of a particular effect.

Camera Modes

Fixed position: This option keeps the camera from moving at all while you are playing. This is good for arcade style games that require an overhead view. The view will be different on widescreen display.

Fixed Offset: Keeps the camera a fixed distance and direction from whatever object the camera is following. This can be a cursor or a bot.

Free option is the default and lets the camera roam free unless the program for a bot tells the camera to be in first person or to follow it. If you're in Fixed Position or Fixed Offset camera, pressing the X button will let you set the position of the camera.

Camera Mode

Explain that camera angles can be set from the Edit World screen or you can program the camera angle for a particular character. You may also set the position for the camera at the start of your game. To do so, go to where to camera should be and then select 'Edit World'. Discuss how different camera modes can be used to create different perspectives for game play. Here are some types of perspectives:

- ❑ First person- Players view the game through the eyes of the character.
- ❑ Third person- Players exist within the game world- which is less immersive. Feels like you are right behind the character or looking over their shoulder.
- ❑ Top-Down- Player has a global view of the game world.
- ❑ Side-scrolling- Fast paced; action is view from side-view camera; doesn't show a lot about the world.
- ❑ Isometric- Player has a global view.

Journal Reflection

What types of games do you like to play? What camera angles do they have? What kind of game are you creating and what camera angle do you think would work best?

Game Creation

Have students work with their groups to set the camera mode in their game. They will use their journal responses to determine what camera angle would be most appropriate. First, they should set the default camera mode. Next they should set the camera mode for your characters (if they have not done this already). Ask students to think about how this affects story line. Make sure they discuss this in their groups.

Student Activity 5

Objectives: Shift camera perspective, use close by and far away, implement shift in behavior using pages.

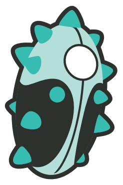
Directions: You have just looked at code and techniques for creating dramatic effects and for crafting changes in behavior within a character. Now is your time to practice.

To Do

- ❑ Consider the three things you talked about during today's lesson—close by and far away, camera angles, and pages.
- ❑ Then explore the various games you have in your Kodu deck and how you might implement the three operations within a single game. For instance, look at Chaotic Orbitals v3 and code in a camera shift to first person—maybe by holding down the A button. The camera shift might also be initiated by bumping, come close to, eating, or grabbing the coin.
- ❑ Now add characters that change behaviors based on their interactions with each other or objects in the world. To create this system of interactions, you may need time to brainstorm and play with the tiles to see what is available to you. If you need assistance, consider adding character and objects in Chaotic Orbitals v3 that have a resting state (Page 1) and then are spurred on to another behavior based on interactions with other characters.

Extra Activity

If you finish the above activity and want to another program challenge, go to Technique, Change Behavior. First play the game and see if you can 'win' the game. Next, program the game, so that you view game play from the perspective of the cycle.



Session 6: Mood and Tone

Students will be able to:

- Use timers, health monitors, and power ups

Prior to this lesson, students may have already seen timers, health monitors and power ups being used. This session will look at these three areas a little more closely. To start, discuss with the class the concept of power ups. In all likelihood they will have a fairly good grasp of how they operate. Basically, a character eats, grabs, bumps, etc something that gives it a special skill or ability for game advantage—often this is temporary. There are also power downs which have the opposite effect. The Level 'Technique: Change Behavior' is an example of a very simple power up—by eating the apple, the bot is able to jump. Students may have already played this game in the Extra Activity in the previous class. First, use the projector to quickly go over this example with your class.

Now have the students explore three games that utilize power ups with their groups. Tell the students you want them to play each of the game and identify the power up (or down) and also identify the code associated with each. After they have played the games, have the students take turns demonstrating and talking about how the power up operates in the game.

It is also important for the kids to start thinking critically about the games. For instance, encourage them to play a Siskel and Ebert routine perhaps to facilitate the discussion. It would be fruitful for them to talk about how the power ups/downs might be better implemented and lend themselves to strategic game play. If they don't enter into this discussion readily, consider inquiring how types of land, timers, allies, thieves, etc might play a role. No special powers are associated with the acquisition of the coins, apples and hearts in these games—which begs the question of how power ups and downs might be better implemented. Health monitors do play a role in the games. Ask them how these add a second strategic element beyond simple scoring.

The three games to consider are:

- ☐ Rock Fight —the player needs to grab hearts for ammo and looses points for shooting; the health monitor goes up has hearts are grabbed and down as ammo is released.
- ☐ Shadow Hunter—the player needs to eat coins and as it does so, the health of the player goes up.
- ☐ Pandemica—the player needs to eat apples to accumulate points and be healed

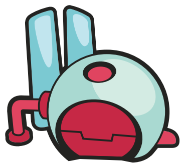
Game Creation

Today the game creation activity should come before the journal reflection. Students will play and critique the games others are working on. It's OK if teams are not completely finished but they should feel OK about presenting their games.

Journal reflection

Students will write in their journals about others' games. They should critique on the following factors. After writing their critiques, they should discuss them with the team who created the game so that the team can make revisions. If possible, they should provide the other teams with this written feedback to be used in Session 7. Students can use the following criteria to critique others' games:

- Do you think the player has a fair chance in the game—is it winnable by skill or luck? If it is more by luck then you may want to revise the game.
- What are the strategies a player can use to work through your game?
- Does the action of the game bring you into the game world?
- Is music or sound effects used in the game? Are they overly annoying or do they create tone or mood that engages the player?



Student Activity 6

Objectives: Use timers, health monitors, and power ups

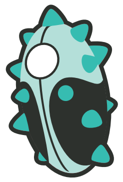
Directions
You have just looked at how power ups operate and can potentially do so. You have also identified how health monitors are applied to characters to activate another level of strategy within game play. Now, it is your turn to implement power ups.

To Do

- ❑ Use an existing world to create a system of power ups and downs.
- ❑ As you are considering how to initiate your system, think about how land, objects, and other characters might be involved.
- ❑ To add a layer of challenge, try to make the power up or down temporary—meaning put it on a timer.

Extra Activity

Look at how the timer is used in Tutorial 3 v.1. To do this you will have to view the program for the saucer. How is this use of the timer different from using the timer for a power up or down? Although there is no health meter being used in the game and although it is not the usual power up and down structure, there is a system of penalty? How is it enacted? What might be the strengths of using this type of penalty system?



Session 7: Changing Behaviors Using Pages, Establishing, and Shifting Perspectives

Students will be able to:

- Use scoring in more complex ways

Your students have likely been using scoring while playing and developing their own games. If they haven't been paying much attention to it as a tool to control behaviors, then this lesson is likely for you.

Create a new level for this lesson. You can call this 'Keep Score'. Use a simple landscape. Add a Kodu and at least 4 cycles. Program Kodu to shoot on command. Kodu should score 5 points every time he shoots a cycle.

It is a useful exercise to add this code in front of students. Ask them how many need to be added before Kodu can win the game (the answer is 4). Now add saucers that move randomly and subtract points from Kodu's score. A volunteer can add these saucers. The volunteer will have to change the program for Kodu so that his score changes when he is hit (bumped) by the saucer (When bump saucer do subtract 5 points). This will make the game more challenging.

Have students play Bonk Out v.18 with their groups. After playing this game discuss if the game was fun. Have them look at the program for each character and discuss the scoring in the game. The group should answer this question in class: How do you win the game?

Journal Reflection

Did you create your game with an audience in mind? What types of people do you think would like your game most? After the journal reflection, have a discussion about the importance of the user in game design.

Game Creation

Students will use the feedback that they received from their peers to make edits to their games.

Student Activity 7

Objectives: Use scoring to change a behavior

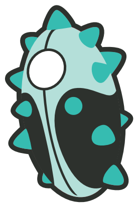
Directions: Each member of your group should solve one of the TO DO's. Check off each TO DO as you complete the task.

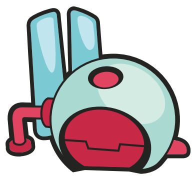
To Do

- ☐ Choose any game environment you want to compete the tasks below
- ☐ Create a simple game in which points are given for doing certain things—like eating, bumping, holding/dropping, etc.
- ☐ Design a situation in which a competitor either subtracts points from your score or has its own score tally comprised of a different color
- ☐ Code a system in which an action is taken or not taken based on a set of scores

Extra Activity

Work with your group to add a Pushpad to the level you are working on. Your pushpad should create a saucer every 0.5 seconds. How do fractions work in Kodu? What happens when you add the tile 'random'?





Session 8: Power Ups, Health, and Timer

Student Presentations

This entire session will be devoted to student presentations. Outsiders (parents, game developers in the community, other students) will be invited to come play the games that students have created. This is a great opportunity for students to showcase and share the games that they worked to create.