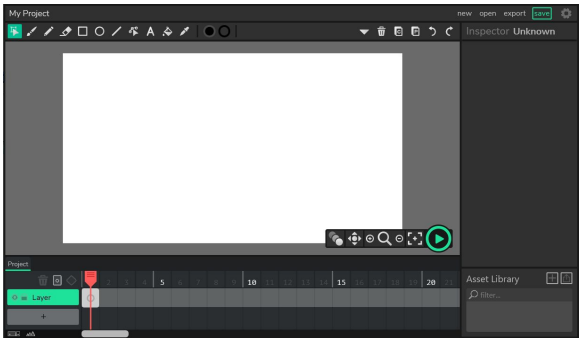
Lesson 1.2

3.7.2020

What is the Wick Editor?

DAILY OBJECTIVE

Students will be introduced to the Wick Editor, a free and open source animation and game creation environment. We'll be using the Wick Editor to create our TEALS end of semester projects.



The tool is accessible from:
www.WickEditor.com

MATERIALS

Educator

- Projector Screen and Computer to show examples.

Students

- Computers with Internet Access
- Note Taking Materials

PREP

Educators should review all examples and ensure they can properly add student codes to examples shown in Wick Editor.

Educators should also attempt to create a short point and click game prior to the session with the help of this guide in order to better understand the common issues students may face. Additional practice with the drawing tools is not required but can certainly help during the lesson if students run into trouble.

For additional support, follow this video tutorial at aka.ms/WickEditorPointAndClick_video

DEFINITIONS

Here are a few terms that you might find useful today.

1. **Wick Editor:** A free, online tool for creating games, animations and interactive projects. This is the tool students will be using to create their own projects throughout the TEALS project.

2. **Point and Click Game:** An interactive game where users are tasked with navigating an environment solely by clicking with a mouse.

3. **JavaScript:** A programming language primarily designed for use on the web. This programming language is used on most modern websites to create interactions, animations, and allow the website to function. In Wick Editor, JavaScript is the primary programming language. This is not the same as Java.

4. **Canvas:** The area of the Wick Editor where creators can place drawings and images to create animations and games.

5. **Timeline:** The Timeline is the “brains” of a Wick Editor project. It contains frames that store the visuals of your animations and games.

6. **Inspector:** The area of the editor where users can view and change the properties of any selected object.
7. **Frame:** An element on the timeline that contains visual and audio information while also describing when to show or play that content.

8. **Scene:** A colloquial term for a frame used to display a setting. For the purpose of this guide, scenes and frames are almost identical!

LESSON PLAN

Section 1: Daily Introduction and Examples

Objective	Teachers will introduce the daily activity, creating an animation and interactive project. Teachers will also demo several example projects to students that incorporate Azure Cognitive Services, into Wick Editor projects.
Duration	10 Minutes
Class Style	Computers should be away. The Classroom should be set up for a full room discussion.
Materials	Note Taking Materials

1.1 Lesson Introduction

Give students a quick introduction to what they'll be making today, a "point and click" game in the Wick Editor.

Point and click games are simple interactive projects that allow a user to explore a game world by only clicking with a mouse. Game designers can combine illustrations, animations, and tiny amounts of code to create very expressive point and click games in Wick Editor.

1.2 Interactive Examples

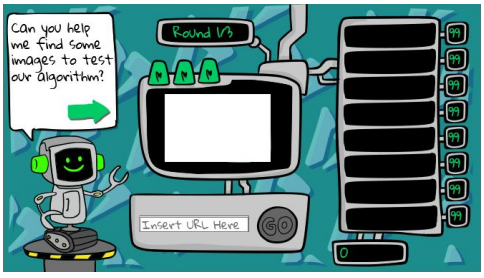
Open the following examples with the class, and ask for volunteers to play with them. The following examples are games that integrate Microsoft Azure Cognitive Services systems into the Wick Editor creation tool we'll be using.

WARN	The following projects must have a "student code" added to them to work! Follow the "How to add a student code" guide at the end of this lesson plan for instructions on how to do this.

1. **"Locked in the Museum"** - A point and click adventure game, with text based puzzles that incorporates AI Text Analysis using Microsoft Azure Cognitive Services.

Direct	aka.ms/WE_TextExample_direct
Download	aka.ms/WE_TextExample

2. **"The Climate Lab"** - An image-finding game that challenges players to find images that an AI image Analysis program can identify!



Direct	aka.ms/WE_ImageExample_direct
Download	aka.ms/WE_ImageExample

3. **"Good Dog"** - A simple role-playing game where players use their facial expressions to simulate a service dog's reaction to their owner having a medical issue.

Direct	aka.ms/WE_VideoExample_direct
Download	aka.ms/WE_VideoExample

While reviewing examples, ask students to reflect on the project elements they may want to include in their own work, such as:

1. World Building
2. Menu Screens

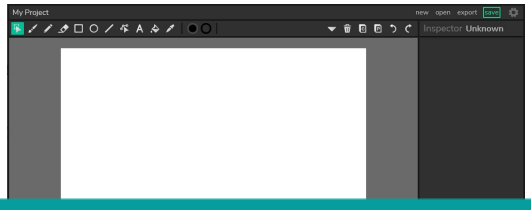
3. Sound Effects and Music
4. Interactivity
5. Animations
6. Timing
7. Puzzles
8. Lighting Effects
9. Animated Text

Section 2: Drawing in the Wick Editor

Objective	Students will learn about the Wick Editor's drawing tool functionality, and how to create basic sketches.
Duration	5 Minutes
Class Style	Student computers should be open to editor.wickeditor.com.
Materials	Student Computers

2.1 Creating a drawing

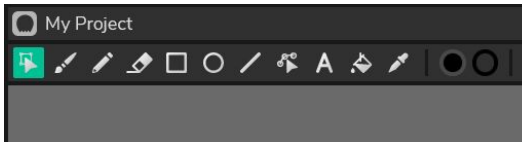
Students will create a basic drawing in Wick Editor. Students should be given a prompt to start this exercise, such as "Create a character from Movies and



Television shows”.

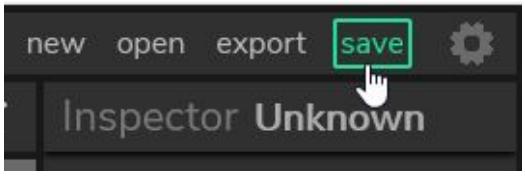
1. Open the Wick Editor by going to editor.wickeditor.com.

2. Start in Wick Editor by experimenting with the drawing tools.



   Be sure to use as many as you can, including:
   a. **Brush**
      i. Create basic lines and shapes.
   b. **Cursor**
      i. Manipulate shapes on the canvas (scale, rotation, position)
   c. **Eraser**
      i. Remove parts of the drawings on canvas.
   d. **Shape Tools**
      i. Create Rectangles, Ellipses, and Lines
   e. **Text Tool**
      i. Add a title, or signature, to your project.

3. Save your work by pressing the “save” option in the upper right of the editor. You should save as often as you can, without slowing down your workflow!



NOTE

Save Often!

Students should be encouraged to save **as often as possible!** Saving is an integral part of the multimedia creation process, and most tools don’t have built-in cloud saving. (Wick Editor is one of these!).

This file will be saved to their device. If you’re using a Learning Management System to track student work, we recommend having students upload this file to their LMS account. Students can reopen these projects at a later time to add additional details. If possible, even if you’re not using an LMS to track progress for this project, we recommend having students have the ability to save these projects across days. This could be on Google Drive, OneDrive, the desktop, or a thumb drive.

Wick Editor File Types

Wick Editor can save many file types. Only the .wick file type can be reopened! Below, we show a breakdown of these file types and how you can describe them to your students.

File Type	Description
.wick	Used to save Wick Editor projects for later editing.
.gif/.mp4	Used to display and share animations on other platforms.
.html/.zip	Used to display and share interactive projects and games on other platforms.

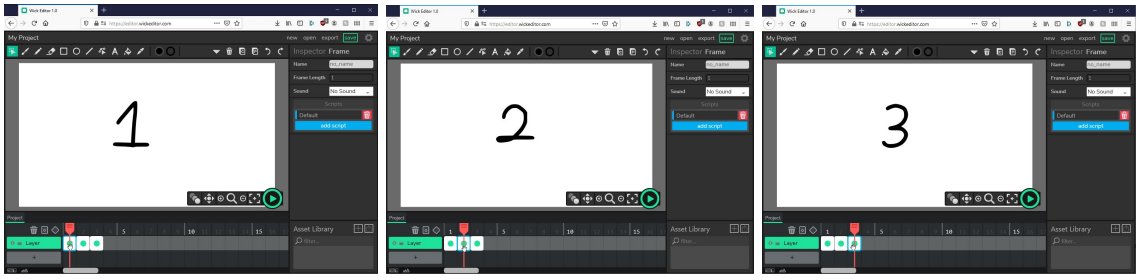
Section 3: Making an Interactive Project

Objective	Students will use the Wick Editor to Create a short, 3-frame interactive project.
Duration	25-30 Minutes
Class Style	Student computers should be open to editor.wickeditor.com on a new, blank project.
Materials	Student Computers

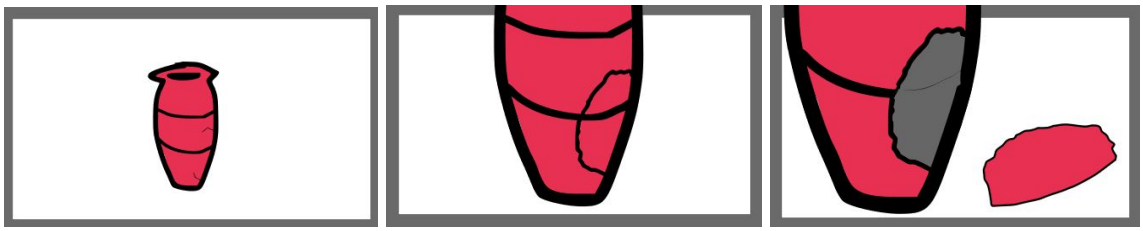
3.1 Making a multi-frame animation.

1. Start by creating 3 frames on the timeline. You can do this by pressing the “add frame” button while highlighting an empty space on the timeline.

2. When creating frames, you will see new frame elements on the timeline. You can move between each frame by selecting each frame object!



3. On each frame, create an image that changes slightly from frame to frame. In our example, we've created an object (a red vase) that cracks and breaks from frame 1 to 3.

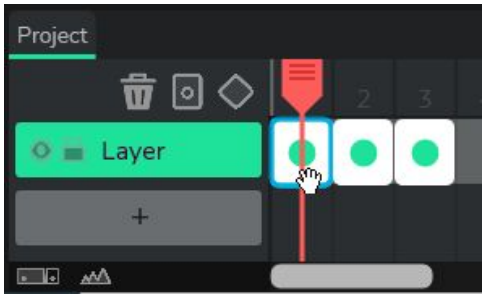


4. Once you're done, press the play button (the large green button on the bottom-right of the canvas). The three frames should play quickly, making an animation!

3.2 Preparing The Project For Interactions

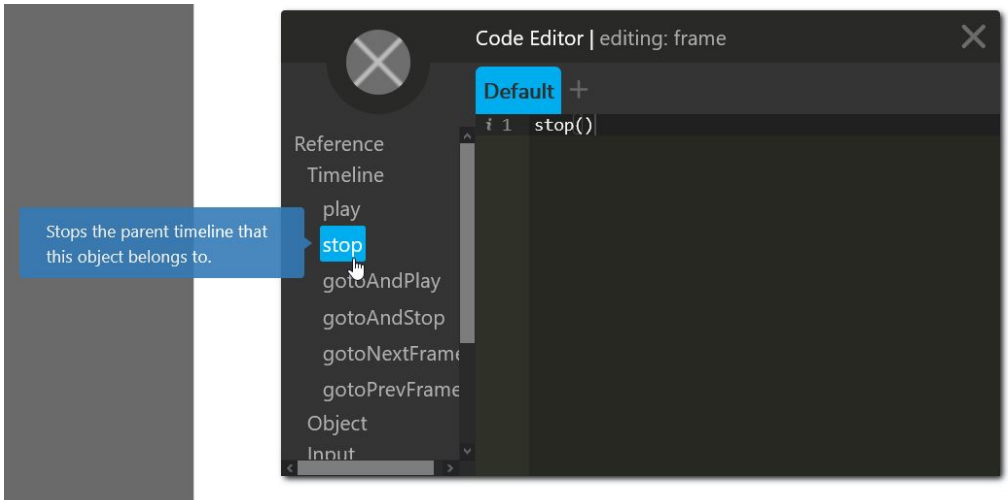
1. To convert the project into an interactive project, we start by adding code that tells the Wick Editor to stop and wait for user input.

   Select the first frame on the timeline.



2. Next, select "default" from the Scripts menu in the Inspector on the right. The code window should now appear.

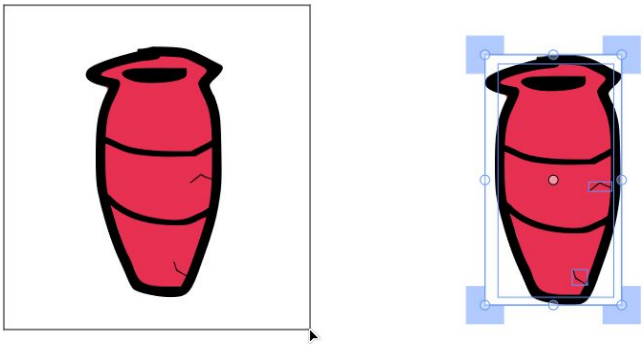
3. We'll add the **stop()** script to this frame. This can be done by typing the command (don't forget the parentheses) or by selecting the **stop** command from the interactive reference on the left of the coding window.



Closing the coding window and pressing the Play button now should cause the project to stop on the first frame!

3.3 Adding Interactive Elements

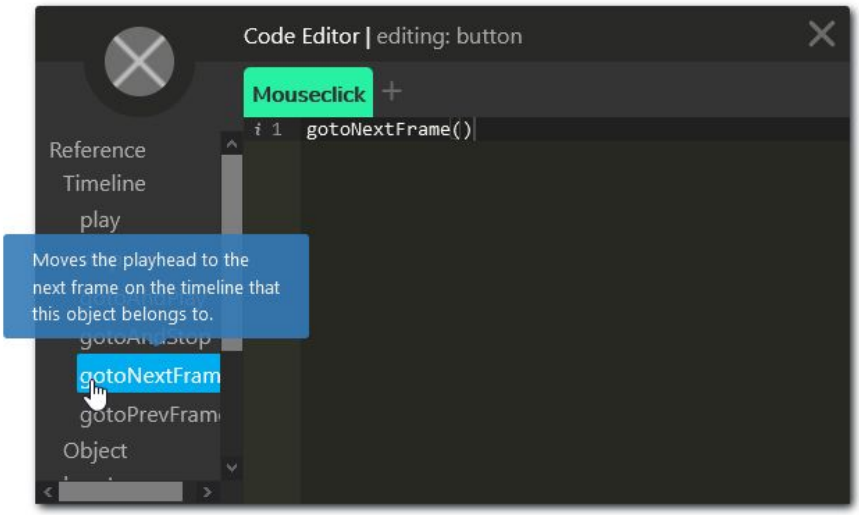
1. Next, we'll add an interactive button to the first frame. Select your object using the Cursor tool. Create a selection box by clicking and dragging from an empty area of the Canvas.



2. With the object selected, convert this object to a button using the **"Make Button"** option in the inspector.

3. With the object converted to a button, and that object selected, there should be a new option for a **mouseclick** script within the inspector on the right of the screen.

4. Open the **mouseclick** script, and insert the command **gotoNextFrame()** by typing the command, or selecting the command from the interactive reference.



5. Closing the code editor and playing the project will now allow you to click the interactive object, and it should bring you to the second frame! With this command, we now have the basis for an interactive storybook.

3.4 Challenges

Students now have the basics of interactive projects under your belt! We'll move on to creating some more complex projects. Educators can follow a video tutorial demonstrating how to create all of the projects shown by watching the video at: aka.ms/WE_PointAndClick_video

C 1	Never get stuck on a frame. Add a button to each frame that allows you to return to the previous frame using the gotoPrevFrame command. Remember, a button can be ANY drawing, or image, you make!

C 1 A	This is easily done by adding an extra item to the canvas on frames 2 and 3, converting that item to a button, and adding the gotoPrevFrame() to the object's mouseclick script. Below, we add a back arrow to the frames. **Completed Project:** - **Direct:** aka.ms/WE_BrokenVaseDemo_remix1_direct - **Download:** aka.ms/WE_BrokenVaseDemo_remix1

C 2	More than one ending. Add new options that allow a user to move through your project in a non-linear way. i.e. Frame 2 can move to either frame 3 OR frame 4, depending on which button is pressed. Use the gotoAndStop(i) command to jump to any frame number!

C 2 A	Create a "branching narrative" that allows a player to explore more than one outcome using the gotoAndStop() command. The gotoAndStop() command can allow a user to go to any frame, by inserting the number of that frame into the command (i.e. to send a user to frame 4, you would use the command gotoAndStop(4)) Below, we create a second outcome to our vase's "story" by giving the user an option to break the vase with a hammer. (The hammer is just another button!) This new option brings us to frame 4, showing the vase broken. **Completed Project:** **Direct:** aka.ms/WE_BrokenVaseDemo_remix2_direct **Download:** aka.ms/WE_BrokenVaseDemo_remix2

3.5 Code Review

Here's a list of all the commands students used today in your challenges.

Command	Description
stop()	Stops the playhead of the animation until some input causes the project to play again or move to another frame.
gotoNextFrame()	Moves the playhead ahead to the next frame.
gotoPrevFrame()	Moves the playhead backward one frame to the previous frame.
gotoAndStop(i)	Moves the playhead to frame i and stops the timeline on that frame.

Common Misconceptions

Below are some common misconceptions that may appear in discussion around today's content.

1. This isn't "Real" Coding

Many students won't realize that what we are doing through creating these small projects is in fact prototyping with actual JavaScript! Prototyping is an important development step, critical to many successful projects.

Wick Editor isn't the only tool that combines visuals and programming. There are hundreds of industry standard development tools like:

a. **Unity**
    i. Combines 3D models and animations with code in a viewport (canvas) system. Used in many industries from Game Design, to Architecture.
b. **Unreal Engine Blueprint System**
    i. Game designers will use the blueprint system, a visual coding system in AAA games.
c. **Processing and P5.js**
    i. Code is used within this visual arts focused programming language to primarily create 2D and 3D projects.
d. **Max MSP**
    i. Electronic Musicians will combine visual node based coding elements with traditional programs to create stunning sound pieces.
e. **Azure Machine Learning Studio**
    i. A professional, GUI-based development environment for creating Machine Learning workflows in Azure!

COMMON PITFALLS

1. Adding Code to the Wrong Objects

a. Students will often add code to the wrong object (such as adding code to a frame, instead of to a button). Make sure code is always added to the correct object while debugging.

2. Spelling and Capitalization of Commands

a. Students will often have trouble when adding commands into projects by misspelling commands, or capitalizing the wrong elements (such as Stop() vs stop()) Always check for spelling and capitalization errors while debugging.

SUCCESS CRITERIA

These success criteria are a simple way to ensure students are on track. They are designed to help educators guide conversations and example development between each day's content.

Discussion	Exploration	Application
Students are able to communicate elements of interactions in demos that they would like to modify and incorporate into projects of their own.	Students can successfully explore all basic elements of the Wick Editor. This includes all of the basic drawing tools described in section 2.	Students have completed all challenges including using all commands in this lesson, such as stop(), gotoNextFrame(), gotoPrevFrame(), and gotoAndStop() in a way which creates linear and branching narratives.