

Lesson 1 - Make Your First Game - Teacher's Guide

<http://www.stencyl.com/teach/act1/>

Objective

Introduce students to the fundamentals of games and creating them. Students will learn the following concepts:

- What is a game?
- How do games work?
- Development happens iteratively (a little bit at a time).

Outcome

Using Stencyl, students will create a simple game from a template.

Lesson Plan (1 hour)

Discussion
15 minutes

Cover the topics under Discussion Notes (Page 2)

Present the topics. Pose questions at appropriate points and encourage students to participate in the discussion.

Demo
20 minutes

Demonstrate Stencyl to the class (Pages 3 - end)

Follow our template.

Alternate Plan: You may prefer to alternate between demoing and having students complete parts of the activity.

Activity
25 minutes

Make a Game

Students will create a simple game using Stencyl. They will become familiar with the Stencyl interface and fundamental game design concepts.

Note: If students don't finish, let them finish it as homework or during the next class.

Discussion Notes

Topic 1: What is a game?

A game is a structured form of play done for enjoyment. Games are bound by rules, goals and interactions between the players.

Discussion Idea: Take a well-known game (physical or video) and dissect it. List out the rules, goals and forms of interaction.

Topic 2: (Discussion) What are examples of games you've played?

Ask this question to students and ask them what they like and don't like about the games they've mentioned.

Topic 3: (Discussion) What makes a game fun to play?

- Gameplay
- Easy to Learn
- Challenge
- Competition / Multiplayer (playing with or against others)
- Story

Tip: Student should describe what makes a game fun by using actual examples.

Topic 4: How do you make a game?

Games are apps (computer programs) that are written using code (language that computers understand). In this class, we'll be using Stencyl, an app that lets you put together games by clicking blocks together. Stencyl writes the code for you, so you don't need to know how to code to create a game.

Aside: Stencyl is similar to MIT Scratch. Mention this if students have used Scratch in the past.

Topic 5: (Optional) How do games work?


Games draw graphics (visuals) to the screen, play sounds and are controlled by the player using a keyboard, mouse and/or a game controller. The game that runs through a set of instructions continually (at least 60 times a second) to update the game's state.

How to Demo Stencyl

Demo steps are divided into an explanation and accompanying actions.

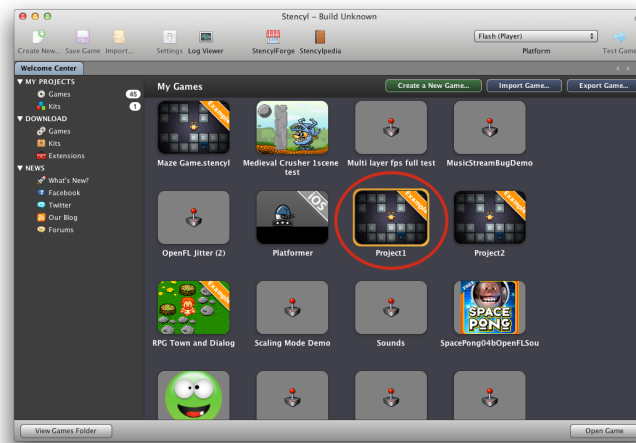
The **explanation** have you (the teacher) **explain** what's happening. The **actions** are **what you do on the computer** to demonstrate how to use Stencyl.

Step 1 of 7: Introduce students to Stencyl

Explanation	<p>Stencyl is an app that lets you make games without knowing how to code. Instead of coding, you click and drag blocks together to form your game's "code."</p>  <p>Stencyl can create any kind of two-dimensional (2D) game within reason and has been used to create platformers, adventure games, shooters and all sorts of other games.</p> <p>Games can be published to the web (Flash), desktop (Windows/Mac/Linux) and mobiles (iPhone, iPad, Android).</p>
Actions	<p>1) Introduce Stencyl to the class.</p> <p>(Optional) If students own/bring their own computers, show them how to download and install Stencyl from our website.</p> <p>2) Show off some games made in Stencyl. Our showcase is a good place to start.</p> <p>3) Finally, launch Stencyl itself.</p> <p>(Optional) If students own/bring their own computers, when Stencyl prompts them to sign in, we recommend that they sign in using a shared classroom account for simplicity.</p>

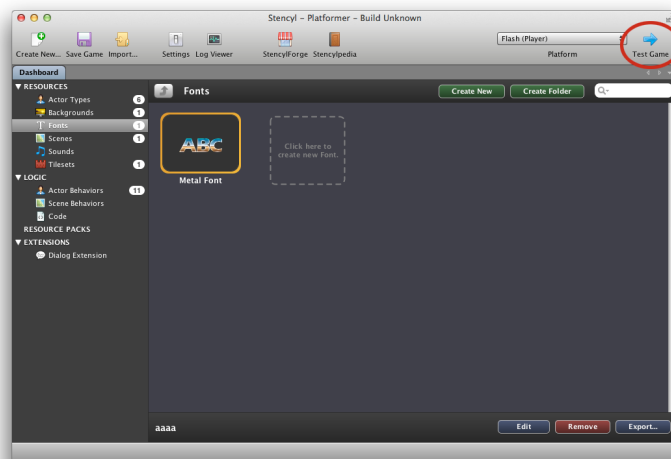
Step 2 of 7: Show off the Welcome Center

Explanation	The Welcome Center lists out all of your games (projects).
Action	Open up Project1 by double-clicking it.



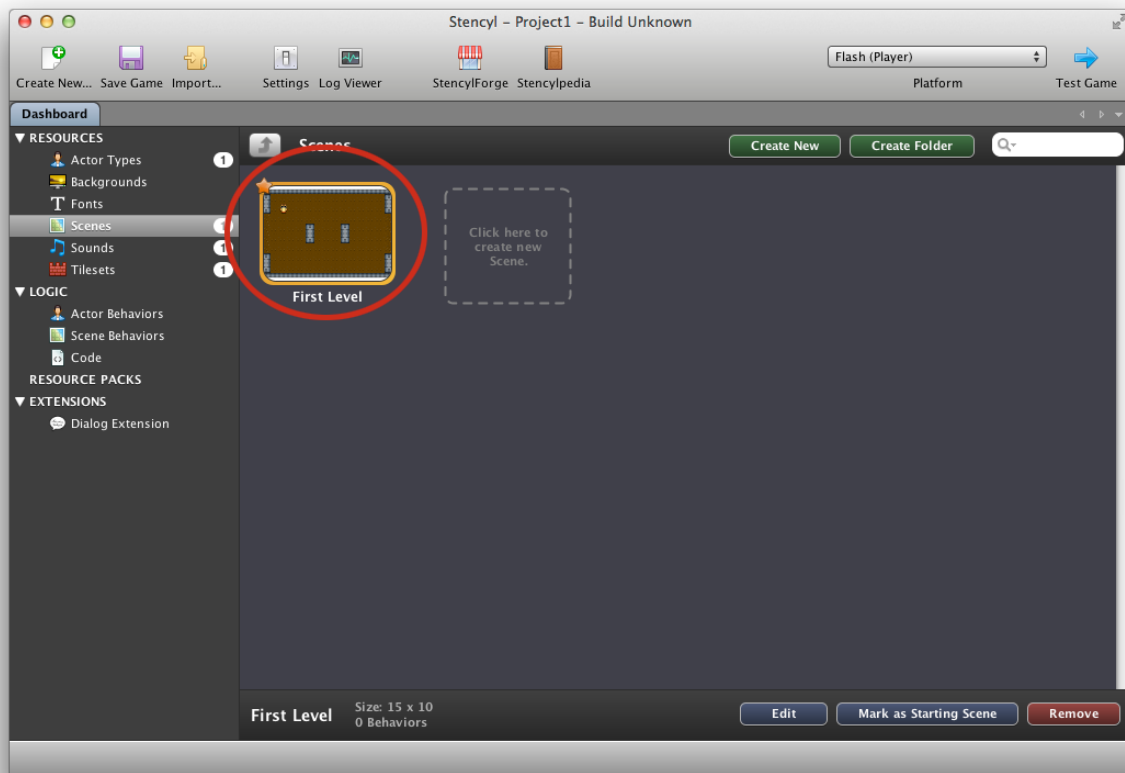
Step 3 of 7: Show how to Run a Game

Explanation	<p>Running a game means telling the computer to begin the game, so that you can play it.</p> <p>By default, Stencyl-made games are powered by Flash. Flash is a technology that lets you run games in a web browser.</p> <p>If you do something "wrong," your game may not run at all, and Stencyl will ask you to fix the problem before proceeding.</p>
Actions	<ol style="list-style-type: none"> 1) Click Test Game at the top-right corner of the app's window. 2) After some time, the game will pop up in a separate window when it's ready to play. 3) Close the game after you are finished playing with it.



Step 4 of 7: Show off the Dashboard

Explanation	<p>The Dashboard is where all of a game's assets (resources) are stored and organized. Common types of game assets include...</p> <ul style="list-style-type: none"> • Actor Types - The “actors” in the game such as the player and the enemies. • Backgrounds - The “wallpaper” that displays behind a level. • Fonts - Ways of styling text • Scenes - The “levels” in your game • Sounds - Music and sound effects • Tilesets - Building blocks for levels. Think of a game like Super Mario Bros and how the terrain inside each level consists of equally-sized squares.
Actions	<p>1) Go over the common types of assets as described above. Flip to the corresponding page for each and open each up to give students a feel for what they are.</p> <p>2) When you are finished, flip to the Scenes page and click Create New.</p> <p>3) Give any name to the scene and click OK to proceed.</p>



Step 5 of 7: Show how to use the Scene Designer

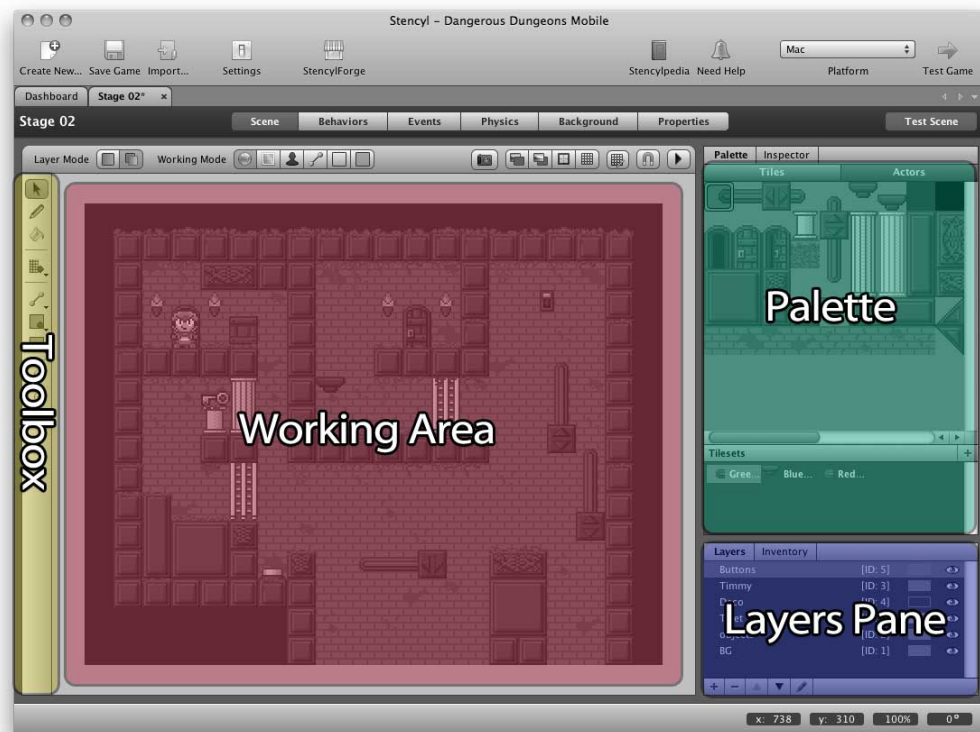
Explanation (Part 1) - What is the Scene Designer?

The Scene Designer is where a game's scenes get built. Scenes are composed of three main elements...



Actors	The "participants" of a game (such as the player and enemies).
Tiles	The tiny, square building blocks that define a scene's terrain.
Backgrounds	The "wallpaper" that displays behind a scene.

Explanation (Part 2) - Terminology

Canvas	The central working area.
Toolbox	The thin strip of icons on the left. Click on a tool to activate it. For this lesson, we'll work with the Pencil and Select tools.
Palette	The area on the right where you pick what game elements (Actors, Tiles) you want to paint with.



Explanation (Part 3) - The Tools

<p>Pencil Tool</p> 	<p>Purpose</p> <p>The Pencil Tool is used for placing actors and tiles.</p> <p>Usage</p> <p>Click and drag to place actors/tiles.</p>
<p>Select Tool</p> 	<p>Purpose</p> <p>The Select Tool is used for selecting and moving existing actors and tiles.</p> <p>Usage</p> <p>Click individual actors/tiles to select them.</p> <p><i>(Holding down Ctrl while doing this will append/remove to your selection)</i></p> <p>Click-and-drag to select multiple actor/tiles in the highlighted area.</p> <p>Click away from your selection to deselect it.</p> <p>Press Backspace/Delete to remove a selection.</p> <p>Press the Arrow Keys to slowly move a selection around.</p>
<p>Switching Tools</p>	<p>Switch between the Pencil and Select tools by pressing Spacebar.</p> <p><i>(Or click on the corresponding buttons in the Toolbox on the left.)</i></p>

Actions (Part 1) - Working with Tiles

1	Activate the Pencil tool.
2	Place tiles by clicking and dragging your mouse, after picking a tile from the Tile palette.
3	Switch between different Tilesets (collections of tiles).
4	Undo/Redo
5	Activate the Select tool
6	Deleting Tiles - Select tiles by clicking on them (or click-dragging several in an area) and then pressing Delete/Backspace.
7	After making something presentable, test the game.

Actions (Part 2) - Working with Actors

1	Activate the Actor palette.
2	Activate the Pencil tool.
3	Place actors by clicking and dragging your mouse, after picking a tile from the Tile palette.
4	Switch between different actors and place them.
5	Undo/Redo
6	Activate the Select tool.
7	Deleting Actors - Select actors by clicking on them (or click-dragging several in an area) and then pressing Delete/Backspace.
8	Test the game to show off the changes.

Step 6 of 7: Talk about Iterative Development

Iterative development means that one should **build a game in small chunks** rather than trying to get the whole thing working all at once. To take an analogy, it's like the tale of the [tortoise vs. the hare](#).

In practice, this means making small changes, testing the game to verify, then repeating again until the game is complete.

Discussion: Why would it be bad to complete the whole game at once and then test it?

Explanation: *Games are complex. If you don't get something correct, the game could fail to work at all or act in very unexpected ways. The more you do between test runs, the more you have mentally sift through to recall what went wrong. This all adds up to significantly more time spent fixing than if you had worked iteratively.*

Step 7 of 7: Begin the Activity

Take questions and turn students over the activity.