**APCSP Activity 1.1.4 Conclusion Questions**

Learning Target: Generate and organize information in order to communicate observations, processes, and results.

**Link to Your Scratch Remix Project:**

|  |
| --- |
| https://scratch.mit.edu/projects/174263910/ |

**Partner(s):**

|  |
| --- |
| none |

**Conclusion Questions**

1. Scratch broadcasts a message to all sprites. This is just how Scratch is set up as a programming language. We are using messages so that they are targeted to a particular sprite. In a large project with hundreds of sprites and scripts, explain why it might help to follow our convention that only one sprite will respond to a message and to follow our naming convention for messages.

The naming convention we were using is object.method() where object would be the sprite name as the method it’s calling upon would be located in that sprite. So if you see a specific method being called you can see which sprite it originates from so you can easily debug various issues.

2. Based on what you learned during this activity, explain what you think events and handlers are.

Events are triggered when specific criteria are met and the event handlers would be the programming that determines what you do with that event after it has been triggered.

3. As computer programs get big and complicated, it is essential to manage the complexity. An important strategy for managing complexity is called encapsulation. With encapsulation some details of one part of a program are hidden from other parts of the program. In cars, for example, the details of the accelerator pedal's actions are encapsulated under the hood. As the user of the accelerator pedal, the driver only gets to press the pedal and is not encouraged to tinker with the internal mechanisms that make the pedal do what the engineer intended.

a. What are some details of the playG() method that are encapsulated by the handler?

The sax.playG() method is encapsulated within the sax object which in this case is the sax sprite.

b. A software developer can make it easier for other developers to tinker with the details of the program she is writing by exposing the details. We'll get into what this means later. For now, just consider: What details mentioned in part a might be useful to expose? In other words, what variations on playG() do you think would be convenient to allow?

Some variables are know as “global” variables and those can be referenced and overwritten no matter where you are in the code. The sax.playG() method executes code that has access to sax specific variables such as the ability to change costumes. Normally when you’re writing object oriented methods like this you would include parameters within the method. For example instead of sax.playG() as it can only play the G note, you could have something like sax.play(“G”) where “G” could be exchanged for any other note. So you would then have a method that can play any note given the parameters. Sometimes you might want to change the costume of the sax outside of the sax specific methods so sax.setCostume(“costume”) would set the current costume of the sax to whatever was specified. Lots of times you wouldn’t want to change the variables contained in the object directly, instead you would just call upon a method that does it for you as there’s usually some extra “setup” code that you need to change the variable without problems.