MS 839 EL Education

Title	GP 2-01 - Introduction to Objects
Grade level	8th
Discipline(s)	Computer Science, Game Programming
Start & end dates	
Author(s)	Adam Driggers

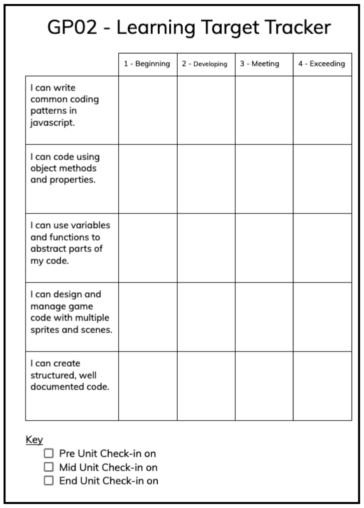
Long-Term Learning Targets Addressed

Long Term Learning Target	This lesson?
I can apply common coding patterns in javascript.	
I can code using object methods and properties.	
I can use variables and functions to abstract parts of my code.	
I can design and manage game code with multiple sprites and scenes.	
I can create structured, well documented code.	

Supporting Learning Targets Assessed	**Ongoing Assessment**
I can access a sprite's properties and methods using the dot operator.	

Agenda Overview	**Teaching Notes**
Warm-Up - 5 min Unplugged Activity - Properties and Methods - 15 min Code Demo - < 10 min Pairing Activity - 10 min Debrief - 10 min Cool Down - 5 min	**Big Ideas** A sprite is an **Object** An Object has actions that it can perform. We call these **methods.** An Object also has attributes that it remembers. We call these **properties.** We access properties and functions using the dot (.) operator.

Lesson Vocabulary	**Materials**
Object Properties Methods	LT Trackers Sprite Reference Character cards for unplugged Slides Code for demo Starter code for pairing Link to Editor

<table>
<tr><td colspan="3"><b>Opening</b>
● <i>How will this lesson or series of lessons help students make progress towards the learning target(s)?</i>
● <i>What will cause students to be curious and want to learn?</i>
● <i>How will I provide students with a vision of the learning targets in a way that gives them ownership of their learning?</i></td></tr>
<tr><td><b>Instructional Plans</b></td><td><b>Duration</b></td><td><b>Meeting Students' Needs</b>
● <i>What support and/or extension do you anticipate students will need?</i>
● <i>How will you adjust the process of and resources used for learning to fit each student's readiness, interest, or learning profile?</i></td></tr>
<tr><td><b>Self Assessment</b>
Start of a new unit. Students self access on the long term learning targets for this unit.

GP02 - Learning Target Tracker
<table>
<tr><th></th><th>1 - Beginning</th><th>2 - Developing</th><th>3 - Meeting</th><th>4 - Exceeding</th></tr>
<tr><td>I can write common coding patterns in javascript.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>I can code using object methods and properties.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>I can use variables and functions to abstract parts of my code.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>I can design and manage game code with multiple sprites and scenes.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>I can create structured, well documented code.</td><td></td><td></td><td></td><td></td></tr>
</table>
Key
☐ Pre Unit Check-in on
☐ Mid Unit Check-in on
☐ End Unit Check-in on
Staple in notebook</td><td>5 min</td><td>Common classroom Practice! Allowing students to track their growth encourages students to see themselves as active participants in their own learning and fosters a growth mindset.

Low stakes opener.</td></tr>
</table>

Work Time

- *What sequenced steps will the students and I take to ensure that all students meet the learning targets?*
- *Is it appropriate to let students grapple first rather than see a model first?*
- *How will students know what quality looks like, and how will I support them in producing quality work?*
- *How will students work or practice together during learning?*

Instructional Plans	Duration	Meeting Students' Needs
Unplugged Groups of students are given a player card with a video Mario, Sonic, Pacman, Galaga Ships, Kirby, Zelda, Donkey Kong, Bowser, Wario, Have students define their object. - What are some **actions** that the character can do? - Ie. mario can jump - Throw fireballs - What are some of the **attributes** that the object has? - Mario is Italian - He wears red - Number of lives - Number of coins Have a few students share out the attributes and the actions for their character.	10 min	Video Game Characters as a hook for learning about object properties and methods. object methods and parameters are going to be a difficult concept so this activity grounds it in something familiar to students. Students collaborate in a think/pair/share discussion regarding the
Big Idea Slide Students write in their notes		Print out slide to fit in student notebooks as a scaffold.
Code Demo Show how to create a sprite and assign to a variable Show how to add controller Show how to console.log values	< 10 min	
Pair Programming Code exploration Students set the velocity of a sprite and see how the gameplay is affected.	10 min	Students already familiar with pairi programming and the idea of navigator/driver. Scaffold: Starter code with Comments and directives and Levels of challenge

Closing and Assessment

- *How will students demonstrate their understanding toward the supporting targets?*
- *What information do I need in order to plan my next instructional steps?*

Instructional Plans	Duration	Meeting Students' Needs
Cool Down On your exit slip write down one example of a method that each sprite has. Write down one attribute that each sprite has.	5 min	Printed exit slips with sentence starters.

Introduction to Objects

GP 2-01

Learning Targets

I can access a sprite’s properties and methods using the dot operator.

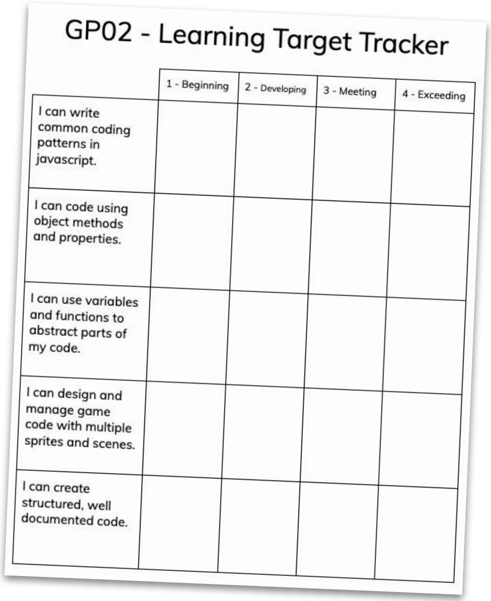
Agenda

Warm-Up (5 min)
Unplugged Programming (15 min)
Code Demo (< 10 min)
Pair Programming (10 min)
Debrief (10 min)
Cool Down (5 min)

Warm Up

Self Assess on the new Learning Targets.

Add your mark to the class poster.



GP02 - Learning Target Tracker

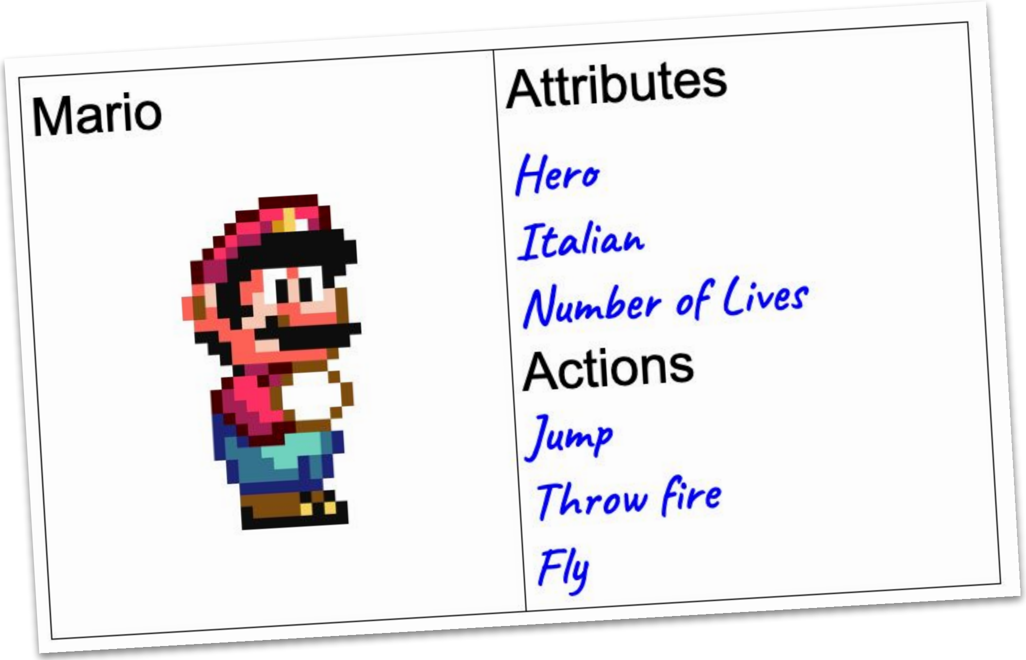
	1 - Beginning	2 - Developing	3 - Meeting	4 - Exceeding
I can write common coding patterns in javascript.				
I can code using object methods and properties.				
I can use variables and functions to abstract parts of my code.				
I can design and manage game code with multiple sprites and scenes.				
I can create structured, well documented code.				

Grab Character Cards

With your table identify the attributes and actions that each character can do.

Bonus!

Write a character card for yourself. What attributes and actions do you have?



Mario

Attributes

Hero

Italian

Number of Lives

Actions

Jump

Throw fire

Fly

Big Idea!

A sprite is an **Object**

An Object has actions that it can perform. We call these **methods**.

An Object also has attributes that it remembers. We call these **properties**.

We access properties and functions using the **dot (.) operator**.

Code Demo

Pair Programming

Cool Down - Exit Slip

Write an attribute that each sprite has.

Closing - How did you feel about today's learning target?

I can access a sprite's properties and methods using the dot operator.

What is a LT?

Confused

Rocked it!

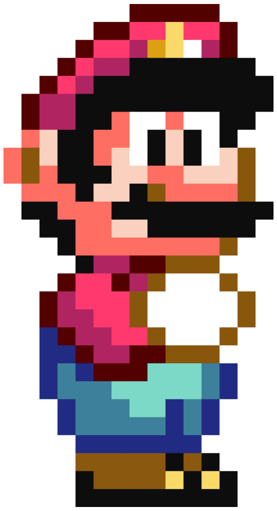
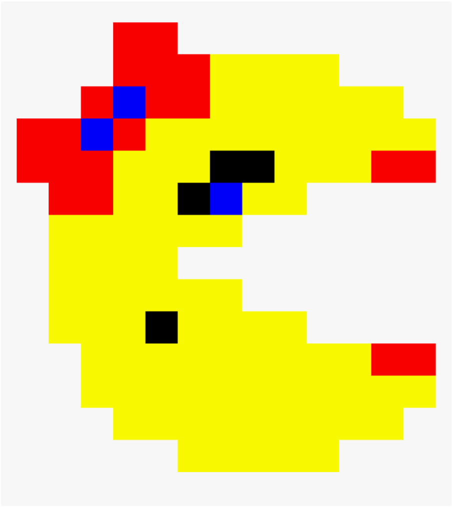
Code for Demo

```
// PROVIDE :: some basic starter code, sprite declaration, controller assingment, and
forever loop
// BE EXPLICIT :: let is creating a variable name, this line means "make a variable
called donut, and assign to it a sprite"
//RUN OFTEN! shows that a donut now appears on the screen
//nothing magic about the name! Change to `let chicken` and run again, still a donut
//STUDENT-PROMPT :: if my sprite will be a donut, why might I choose to name it donut
instead of chicken?
let donut = sprites.create(img`
    ..............bbbbbbb...........
    ...........bb66663333baa........
    .........bb3367776333663aa......
    ........b33333888333389633aa....
    .......b3333333333333389633aa...
    ......b34443333333333338633bae..
    .....b3455433333333334443333ae..
    ....b33322333dddd3333455233daee.
    ...b3d333333dd3bbbb33322333dabe.
    ..b3d333333d3bb33bb33333333da4e.
    ..bd33333333b33aab3333333223a4ee
    .b3d3663333b33aab33366332442b4ee
    .bd3b983333a3aa3333387633ee3b4ee
    .bd6983333baaa333333387633bb4bee
    b3d6833333bba333333333863ba44ebe
    bdd3333333bb3333333333333a44bebe
    add666633333322333366333ba44bbbe
    ad67776333332442336983d3a444b4e.
    add888b333333ee3369833d3a44b44e.
    add333333333333336833d3a444b4e..
    a3dd3333344433333dddd3a444b44e..
    ab33ddd325543333dd33aa444b44e...
    .eabb3dd32233333baaa4444b44e....
    .ebabb3d333d33baa444443b44e.....
    ..ebaab3ddd3aaa4444433b44e......
    ..eebbaab33a44444333b444e.......
    ...eeebbaab444b333b4444e........
    ....ebeeebbbbbbbb4444ee.........
    .....eebbbb44444444ee...........
    .......eeebbb444eee.............
    ..........eeeeee................
    ................................
`, SpriteKind.Player)

// assign the controller to move the donut
// DELIBERATE-ERROR :: mispell as doughnut. computers are so stupid.
```

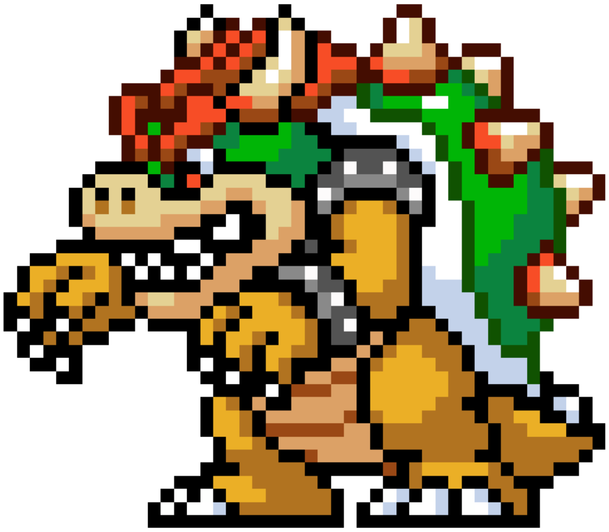
```
controller.moveSprite(donut);
// MUST-ANSWER-Q :: What is donut in the previous line referring to? A: Refering to the donut variable!

// Show using drag and drog. You don't have to remember everything!
controller.A.onEvent(ControllerButtonEvent.Pressed, function () {
    //code to do when the a button is pressed.
    // BE EXPLICIT :: console.logValue prints messages to the console
    // to get at an sprite's property, I use the dot operator.
    console.logValue("x", donut.x); //get the x value, prints to console
    console.logValue("y", donut.y);

    //I can also set a sprites attribute using the dot operator
    donut.x = 0;
    //STUDENT-PROMPT :: why didn't it print the zero? A: order is important.

    //preview pairing exercise.
    //a sprite also has a velocity x and velocity y. change these numbers and see what happens when you run.
})
```

Pairing Code Stem

```
// creates a sprite
let cake = sprites.create(img`
   . . . . . . . . . . b b b . . .
   . . . . . . . . b e e 3 3 b . .
   . . . . . . b b e 3 2 e 3 a . .
   . . . . b b 3 3 e 2 2 e 3 3 a .
   . . b b 3 3 3 3 3 e e 3 3 3 a .
   b b 3 3 3 3 3 3 3 3 3 3 3 3 3 a
   b 3 3 3 d d d d 3 3 3 3 3 d d a
   b b b b b b b 3 d d d d d d 3 a
   b d 5 5 5 5 d b b b a a a a a a
   b 3 d d 5 5 5 5 5 5 5 d d d d a
   b 3 3 3 3 3 3 d 5 5 5 d d d d a
   b 3 d 5 5 5 3 3 3 3 3 3 b b b a
   b b b 3 d 5 5 5 5 5 5 5 d d b a
   . . . b b b 3 d 5 5 5 5 d d 3 a
   . . . . . . b b b b 3 d d d b a
   . . . . . . . . . . b b b a a .
`, SpriteKind.Player);

controller.moveSprite(cake); //assigns the controller to a sprites

//YOU DO: press run, what happens?

//runs the code anytime the a button is pressed
controller.A.onEvent(ControllerButtonEvent.Pressed, function() {
   //Level 1
   //YOU DO: set the x velocity to 200
   //YOU DO: set the y velocity to 200
   //YOU DO: press run, what happens?

   //YOU DO: print the x and y velocity to the console.

   //Level 2
   //YOU DO: change your code to set the y velocity to 0
   //YOU DO: press run, what happens?

   //Boss Level
   //each time the a button is pressed, the velocity increases by 10.
})
```

Mario 	Attributes Actions
Ms. Pacman 	Attributes Actions

Kirby 	Attributes Actions
Ghost 	Attributes Actions

Bowser 	Attributes Actions
Link 	Attributes Actions

GP02 - Learning Target Tracker

	1 - Beginning	2 - Developing	3 - Meeting	4 - Exceeding
I can write common coding patterns in javascript.				
I can code using object methods and properties.				
I can use variables and functions to abstract parts of my code.				
I can design and manage game code with multiple sprites and scenes.				
I can create structured, well documented code.				

Key

- □ Pre Unit Check-in on
- □ Mid Unit Check-in on
- □ End Unit Check-in on

GP02 - Learning Target Tracker

	1 - Beginning	2 - Developing	3 - Meeting	4 - Exceeding
I can write common coding patterns in javascript.				
I can code using object methods and properties.				
I can use variables and functions to abstract parts of my code.				
I can design and manage game code with multiple sprites and scenes.				
I can create structured, well documented code.				

Key

- □ Pre Unit Check-in on
- □ Mid Unit Check-in on
- □ End Unit Check-in on

GP02 - Learning Target Tracker

	1 - Beginning	2 - Developing	3 - Meeting	4 - Exceeding
I can write common coding patterns in javascript.				
I can code using object methods and properties.				
I can use variables and functions to abstract parts of my code.				
I can design and manage game code with multiple sprites and scenes.				
I can create structured, well documented code.				