

# Object Attack

## Outstanding Objects


Dot is so excited to see you back at Decomposphere! Today, you'll learn more about objects and how they can keep their properties out of sight. Let's jump right in!

## Who let the dogs out?

In object-oriented programming, we can choose to keep the variables and functions of an object private. When these features are private, other objects cannot directly control them. Each object can have a combination of both private and public features.

## What's the difference between private and public?

While private properties (variables and functions) cannot be directly controlled by other objects, public properties can. Let's take a look at an example with Dot's pet dog, Rocky!

 <b>Rocky the Labrador</b>			
<b>Variables</b> (Who Rocky is)		<b>Functions</b> (What Rocky does)	
<b>private</b> breed <b>private</b> age	<b>Why private?</b> Dot can't control Rocky's breed and age.	<b>private</b> bark <b>private</b> sleep	<b>Why private?</b> Dot can't naturally control when Rocky barks or sleeps
<b>public</b> owner	<b>Why public?</b> Rocky's owner, Dot, can control who owns Rocky	<b>public</b> feed <b>public</b> play	<b>Why public?</b> Dot can control when to feed or play with Rocky

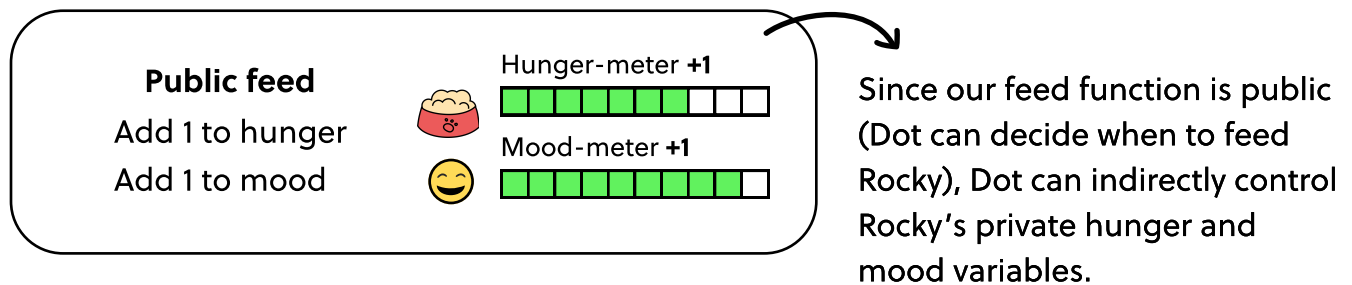
## Why do we use private properties?

In computer science, we use private properties to **protect** features like breed, mood, and hunger from being **directly controlled by other objects**.

## How can we change private variables?

Private variables can only be changed through public functions.

For example, let's take a look at our public function, public feed:




However, some private variables can never be changed.

For example, we cannot change Rocky's breed as a Labrador.

## Grid Attack!

In this action-packed game, you will be creating your own **player object** with its **private/public variables and functions** to attack an opponent!

### Materials


- 2 tokens for you and your friend (This will represent where you are on the board!)
- 1 die 



### How to Play

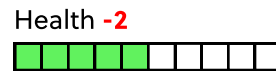
- Define your player object's variables by creating a name, age, and health value. Each player starts the game with a health level of 5 (out of 5). Next to each variable and function, circle whether it is private or public! (*Think: Can these properties be directly changed or used by other objects?*)
- Take turns rolling the die with your friend. Use the **walk** function to move your token the number of steps you rolled in any direction (up, down, left, right). Keep track of your activity in your **activity list**.

(Ex. If you rolled a 6, you can choose to walk 3 steps left and 3 steps up).

If you are on , use the **eat** function to add 1 point to your health.



If you are at the **same location** as your opponent, use the **attack** function to subtract 2 points from your opponent's health.



- Winning the game: You win the game when your opponent's health status reaches 0.

## My Player Object

### Variables

**Private/Public Name**  
(circle one)

---

**Private/Public Age**  
(circle one)

---

**Private/Public Health**  
(circle one)

5

---

### Functions



**Private/Public Walk**  
(circle one)



Move the number of steps rolled



**Private/Public Attack**  
(circle one)



Subtract 2 points from opponent's health



**Private/Public Eat**  
(circle one)



Add 1 point to your own health

## My Activity List

**Function**

**Health**

Ex. Walk left 3 steps

5

---



---



---



---



---



---



---

**Function**

**Health**

---



---



---



---



---



---



---

