# Introduction of 'Snap!'

# What is Snap!

Q1: Why do we choose 'Snap!'? A1: Snap is easy to learn and it shows the combination of the blocks which includes the programming syntax when we write programs.

Q2: What's the connection between 'Snap!' and 'AI' ?
A2: The application of AI to develop programs that do human-like jobs and portray human skills is Machine Learning. 'Snap!' programming provides many possibilities to help us develop artificial intelligence(AI).

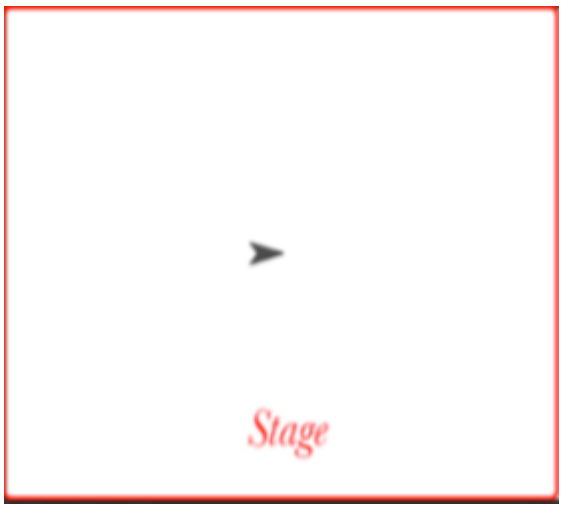
Snap! is a programming language—a notation in which you can tell a computer what you want it to do. Unlike most programming languages, though, Snap! is a visual language; instead of writing a program using the keyboard, the Snap! programmer uses the same drag-and-drop interface familiar to computer users.

## The interface of Snap!




## Five parts of interface :

### 1.Stage



*The stage is on the top of right column*

**What is the stage used for?** It copies the built components into the staging area.

## 2.Sprite

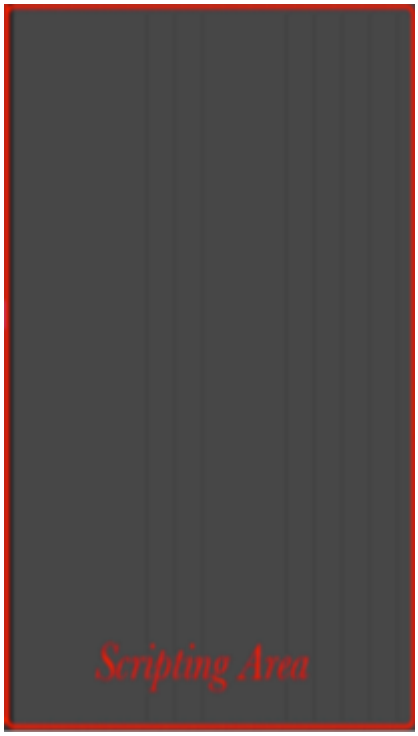


*The Sprite Corral is at the bottom right of the Snap!*

**What is Sprite corral used for?**

It makes a sprite and opens the Paint Editor so that you can used to create a new sprite

## 3.Scripting Area

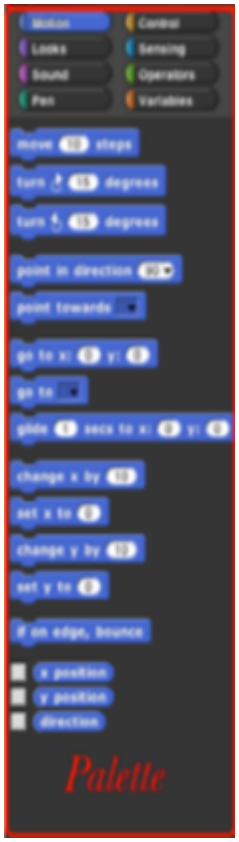


*The scripting area is the middle vertical region of the Snap!*
It contains scripts and also some controls for the appearance and behavior of a sprite.

**What is Script used for?**

Blocks will snap together when one block's indentation is near the tab of the one above it. You should see a white bar appear like the one in the image below, which just shows you where the block will go after you drop it. If you keep attaching blocks together in this way, you will create a script.

## 4.Function




*At the top of the palette area are the eight buttons that select which palette (which block category) is shown: Motion, Looks, Sound, Pen, Control, Sensing, Operators, and Variables (which also includes the List and Other blocks).*

**What is Palette use for?**

The Snaps Palette provides a handy floating palette that holds all the settings and tools related to snapping in a single viewport.

## 5.Tool Bar




*Tool Bar is on the top of the Snap!*

**What is the Tool Bar use for?**

The Snaps toolbar provides access to the most common Snaps settings.

# What is an object?

## Object

Objects are key to understanding object-oriented technology. Look around right now and you'll find many examples of real-world objects: your dog, your desk, your television set, your bicycle.

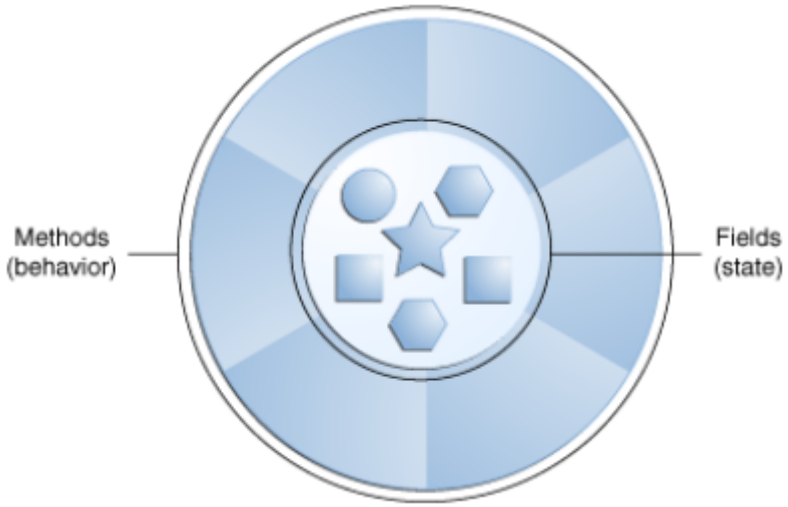
Real-world objects share two characteristics: They all have state and behavior. Dogs have state (name, color, breed, hungry) and behavior (barking, fetching, wagging tail). Bicycles also have state (current gear, current pedal cadence, current speed) and behavior (changing gear, changing pedal cadence, applying brakes). Identifying the state and behavior for real-world objects is a great way to begin thinking in terms of object-oriented programming.

## Observing the real word

Take a minute right now to observe the real-world objects that are in your immediate area. For each object that you see, ask yourself two questions:

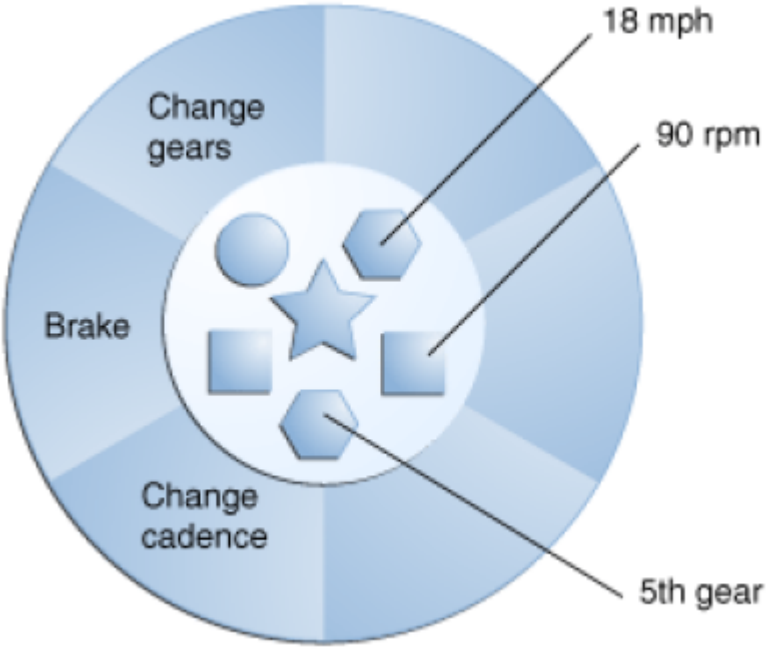
1. “What possible states can this object be in?”
2. “What possible behavior can this object perform?”

*Make sure to write down your observations.*



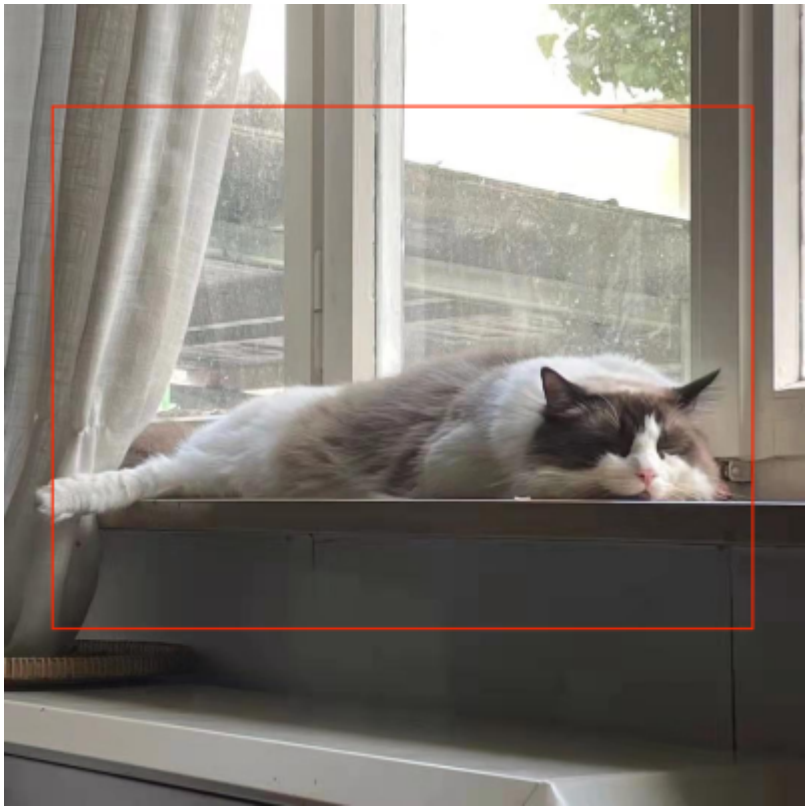

As you do, you'll notice that real-world objects vary in complexity; your desktop lamp may have only two possible states (on and off) and two possible behaviors (turn on, turn off), but your desktop radio might have additional states (on, off, current volume, current station) and behavior (turn on, turn off, increase volume, decrease volume, seek, scan, and tune). You may also notice that some objects, in turn, will also contain other objects. These real-world observations all translate into the world of object-oriented programming.

## Consider a bicycle, for example:




By attributing state (current speed, current pedal cadence, and current gear) and providing methods for changing that state, the object remains in control of how the outside world is allowed to use it. For example, if the bicycle only has 6 gears, a method to change gears could reject any value that is less than 1 or greater than 6.

## Object Detection



Object detection refers to the capability of computer and software systems to locate objects in an image/scene and identify each object. Object detection has been widely used for face detection, vehicle detection, pedestrian counting, web images, security systems and driverless cars.

### Now let's go to the Software objects

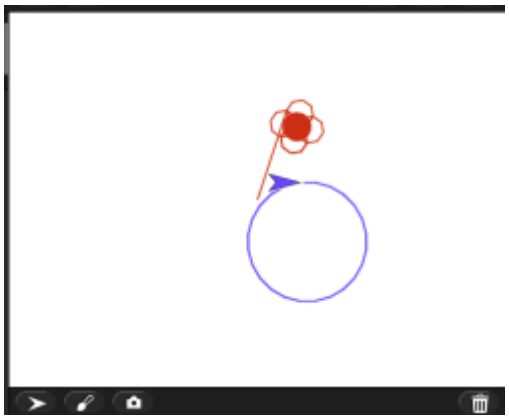
Software objects are conceptually similar to real-world objects: they too consist of state and related behavior. An object stores its state in fields (variables in some programming languages) and exposes its behavior through methods (functions in some programming languages). Methods operate on an object's internal state and serve as the primary mechanism for object-to-object communication. Hiding internal state and requiring all interaction to be performed through an object's methods is known as data encapsulation — a fundamental principle of object-oriented programming.

# Let's do that in Snap

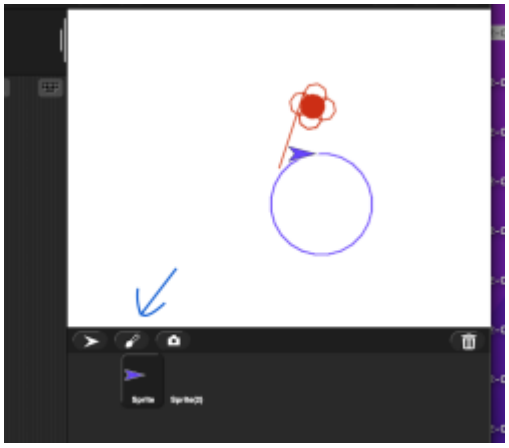
---

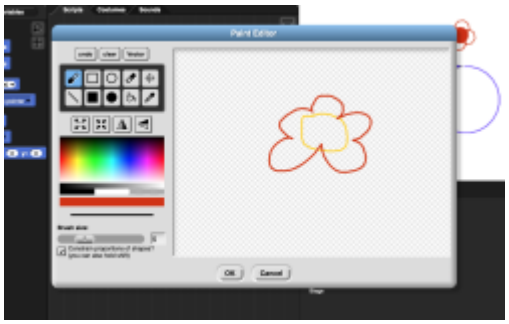
### Example : Combination of Hat blocks and Command blocks to draw a circle.

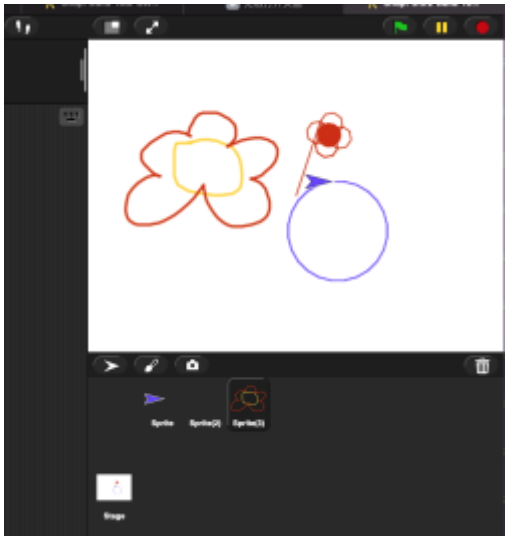




## or using Sprite directly to draw an object







# Now! Do it by yourself!

---

*Task:*

1. Creating your own object and moving it in horizontal direction by using Command blocks.
2. Moving your object in random position.
3. Drawing a star and let's keep it saying 'hello!' and rotating every 0.1 second.

Still improving……

snap_basics.zip

From:
https://wiki.eolab.de/ - **HSRW EOLab Wiki**

Permanent link:
**https://wiki.eolab.de/doku.php?id=ip:ws2021:lets_plaiy:student-documentation:snap:start&rev=1642091325**

Last update: **2022/01/13 17:28**