

Game Programming and Scratch

(An Introduction for
Programmers)

M. Meyer

Programming Languages

- I. Allow us to "talk" to a computer, in a language that we can understand.
 - I. Computers only understand binary (0,1).
 - II. Modern (high-level) languages allow us to write code in a way that we understand.
- II. Require a well-defined syntax and semantics:
 - I. Syntax -> Refers to the rules of grammar, word order and punctuation that must be used. A "syntax error" is usually a punctuation error.
 - II. Semantics -> Refers to the meaning of words that are included in a language. Some words have set meanings, others can be changed (variable mouseX).

Programming Languages

- I. Allow us reuse and share code.
 - I. No need to create code to handle mouse input if someone has already created a function to do that.
 - II. A library is a collection of functions and variables that can be reused when creating a new program.
- II. Fall into different paradigms (types):
 - I. Functional, Logical, Imperative, Procedural and Object-Oriented approaches to programming.
 - II. If you already know one language of a particular type it's easier to learn other languages of that same type.

Note: This is true for spoken languages as well. If you know Spanish it's easier to learn French, Portuguese or Italian. All these languages use similar vocabularies (semantics) and almost the same grammar (syntax).

Scratch

Scratch is an IDE (Integrated Development Environment) application. An IDE is a program that allows users to create, run and debug other programs.

Users create programs in Scratch using an imperative, procedural, object-oriented programming language that has very simple syntax*.

* Don't worry, all of these terms will be made clear to you.

Scratch Interface



Why use Scratch?

1. Scratch is FREE!
2. Simple development environment.
3. Simple syntax.
4. Large library of functions.
5. Supports the basics of programming in 3 important paradigms.
6. Can be used to create ANY simple computer game.

1. Scratch is FREE

- Scratch is developed by the Lifelong Kindergarten group at the MIT Media Lab, with financial support from the National Science Foundation, Microsoft, Intel Foundation, Nokia, and MIT Media Lab research consortia.
- Scratch is free software and will run on Windows, Mac and Linux machines.
- You can download Scratch here:
<http://scratch.mit.edu/>
- The Scratch website has many helpful tutorials as well as a forum for asking questions and getting help.

2. Simple Development

1. Scratch requires very little typing in order to create programs.
2. Visual code creation using drag & drop.
3. FAST FAST FAST development.



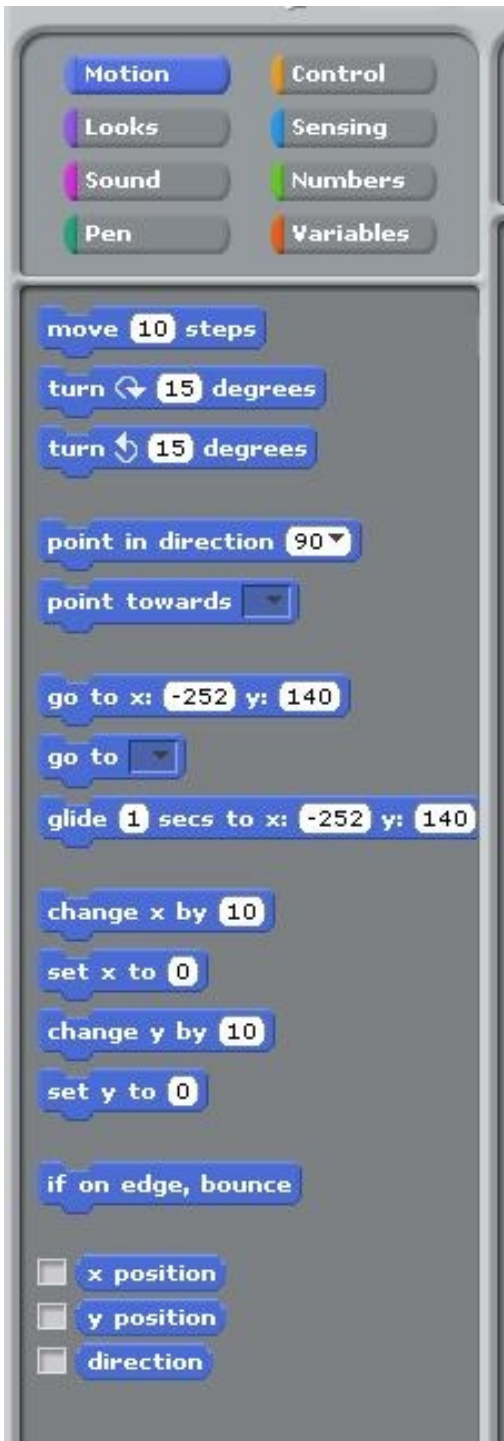
3. Simplified Syntax

- 1.No missing semi-colon problems.
- 2.Code blocks can only fit together in pre-defined way.



4. Large library.

1. Over 100 predefined functions and limited ability to make more.
2. Functions cover vast majority of things that you would want a sprite (object) to be able to do in a game.



5. Basics of Programming

- Just like spoken languages programming languages can be categorized into certain types (or paradigms).
- 3 of the most popular programming paradigms are:
 - A. Imperative -> A 'smart' list.
 - B. Procedural -> Making phone calls.
 - C. Object Oriented -> Programming with objects.

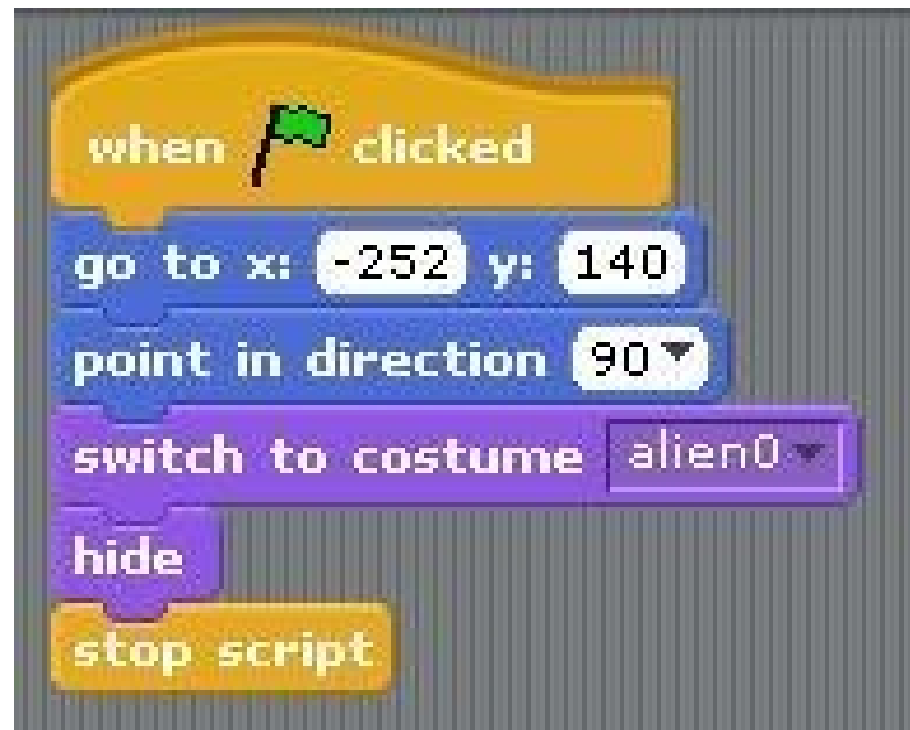
A. Imperative Programming

"a smart list"

1. The imperative paradigm is like giving the computer a list, which tells it step-by-step what to do.
2. To be "smart" your list needs 3 things:
 - i. Sequence -> A predefined order in which to process information. (*English vs. Hebrew*)
 - ii. Selection -> The ability to make a choice. The "IF" statement.
 - iii. Repetition -> The ability to repeat an action. The "WHILE" statement.

i. Sequence

- All "scripts" processed from top down.
- 4 possible start conditions, 3 end.



ii. Selection

- If, If-else and wait_until functions.



iii. Repetition

- Variety of functions including repeat_until.



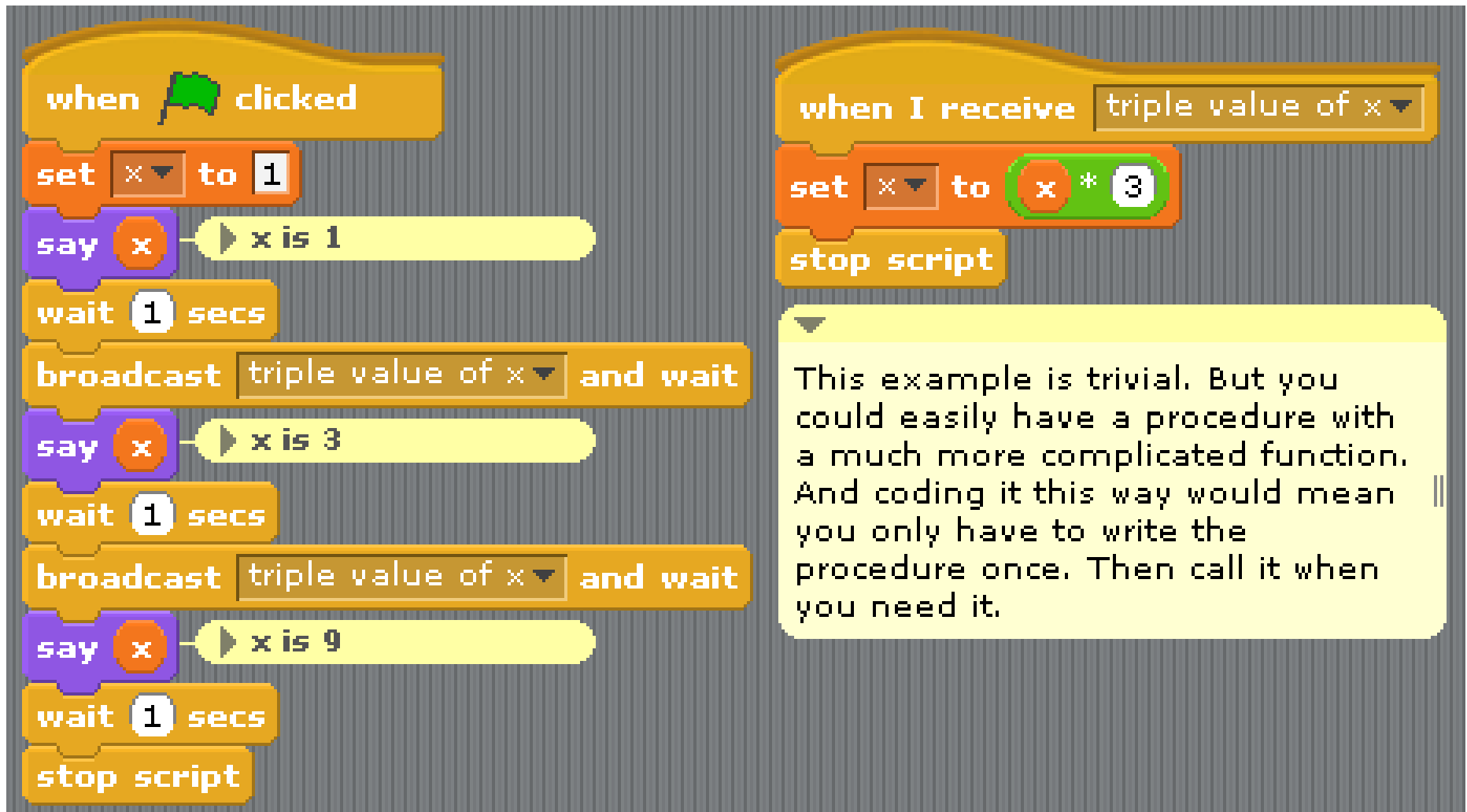
B. Procedural Programming

"making phone calls"

- The procedural programming paradigm is based upon the concept of the “procedure call”: the ability to “send a message” to another section of a program.
- Procedural programming allows us to create sections of code that can be reused over and over .

Broadcast

Scratch allows users to send "broadcasts" which can activate other scripts who are listening for a particular broadcast.



The image displays two Scratch scripts side-by-side. The left script starts with a 'when green flag clicked' block, followed by 'set x to 1', 'say x x is 1', 'wait 1 secs', 'broadcast triple value of x and wait', 'say x x is 3', 'wait 1 secs', 'broadcast triple value of x and wait', 'say x x is 9', 'wait 1 secs', and 'stop script'. The right script starts with 'when I receive triple value of x', followed by 'set x to x * 3', and 'stop script'. A yellow callout box on the right contains the text: 'This example is trivial. But you could easily have a procedure with a much more complicated function. And coding it this way would mean you only have to write the procedure once. Then call it when you need it.'

```
when green flag clicked
  set x to 1
  say x x is 1
  wait 1 secs
  broadcast triple value of x and wait
  say x x is 3
  wait 1 secs
  broadcast triple value of x and wait
  say x x is 9
  wait 1 secs
  stop script

when I receive triple value of x
  set x to x * 3
  stop script
```

This example is trivial. But you could easily have a procedure with a much more complicated function. And coding it this way would mean you only have to write the procedure once. Then call it when you need it.

C. Object-Oriented

1. OO programming is an extremely important programming paradigm.
1. Scratch is not true OO programming, but good example of basic concepts:
 - Creating programs that are composed of interacting objects.
 - These objects have associated properties and functions.

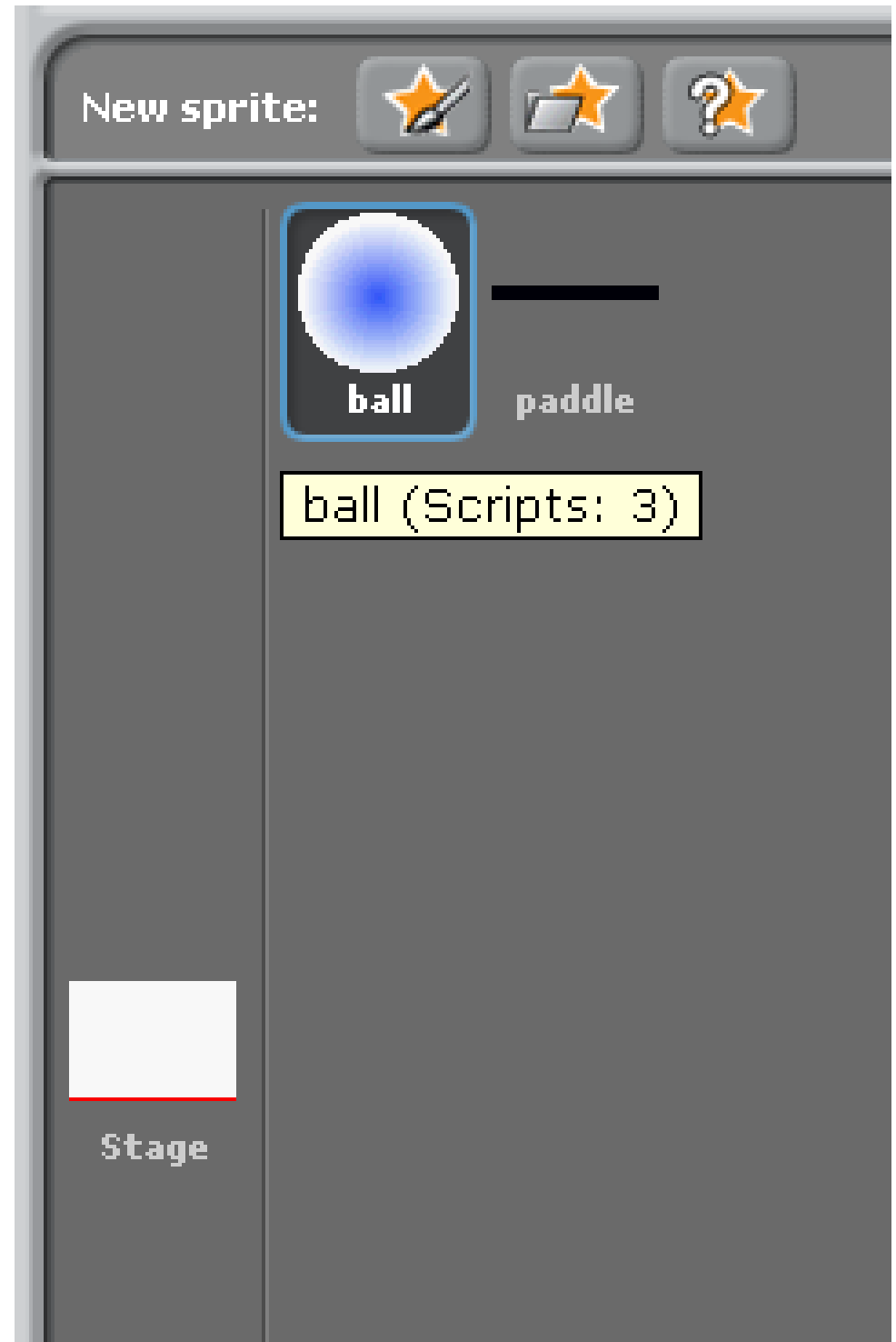
Object-Oriented (cont)

1. Objects in Scratch are called "Sprites".
2. Properties of Sprites include:
 - Location
 - Look
 - User defined properties (variables).
1. Functions of Sprites include:
 - Move
 - Make Sound
 - Detect Collision

Sprites

Found in the lower right corner of the screen.

Click on them to select them and change contents of main window.

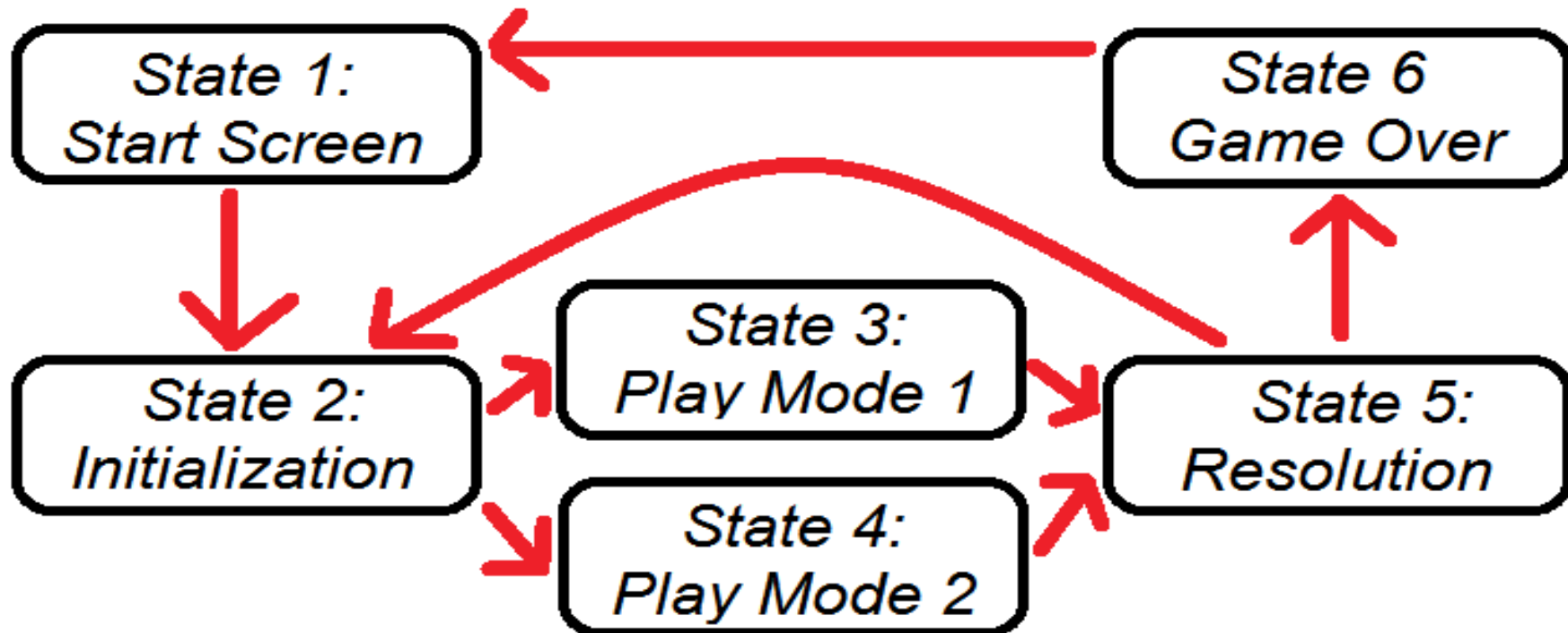


6. Create ANY simple game.

- At this time Scratch does not support Vector Graphics, Multi-threading, and library creation.
- Despite these restrictions it is still possible to create some very interesting and exciting applications/games with Scratch.
- ANY simple arcade game or older console game can be recreated in Scratch.
- Many simple browser-based games (Flash) can also be emulated with Scratch.

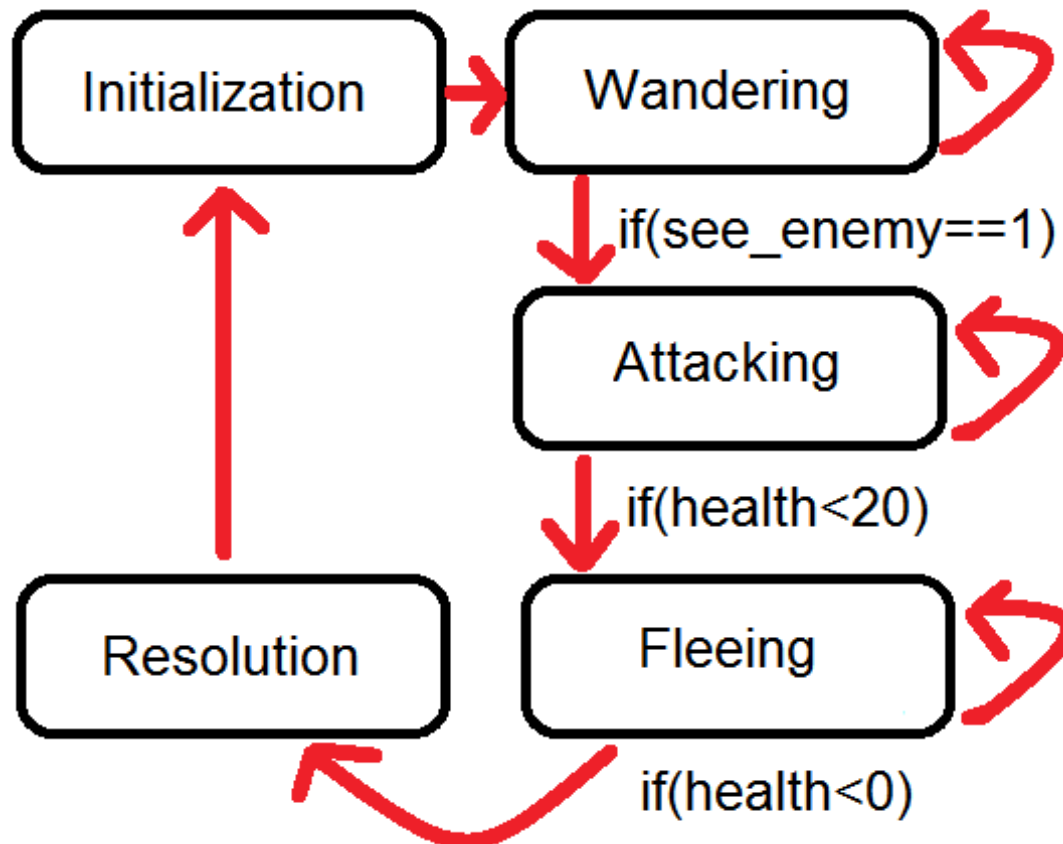
Game State

All games consist of a sequence of states. Each state is characterized by a combination of visual, audio and/or animation effects, as well as a set of rules that are being applied.



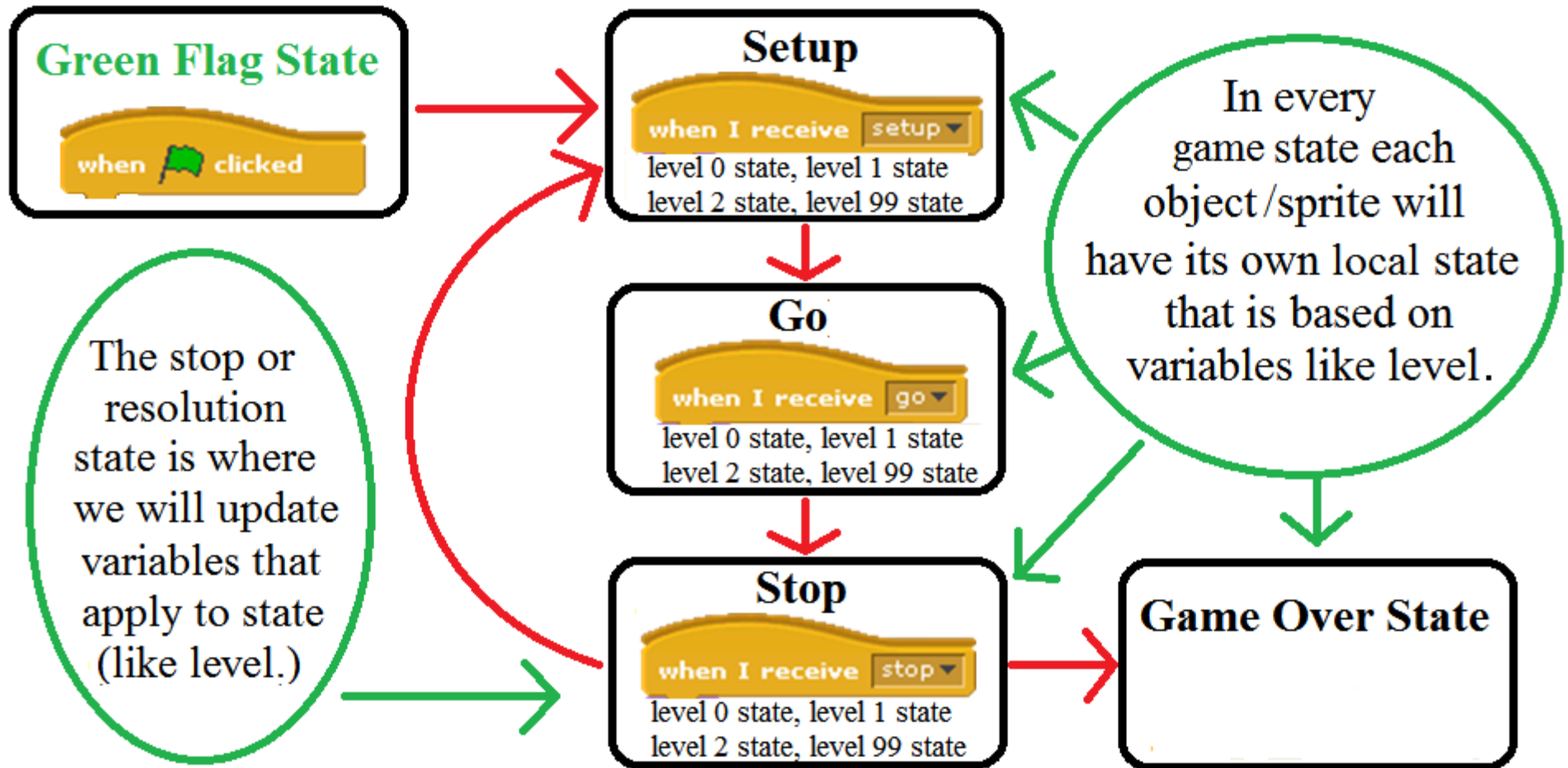
Object State

Objects in the game proceed through their own states as well. These states are defined by the behavior and functionality applied at that time.



Game State in Scratch

A typical game in Scratch might use the following state transition diagram.



Object State in Scratch

In a typical game in Scratch, all of your normal sprites (not stage or any control sprites like buttons) will work very well with only 4 scripts. These scripts (and any associated variables) will control the objects state.

```
when clicked
hide
stop script
```

```
when I receive setup
point in direction 90
go to x: -120 y: 0
set size to 40 %
switch to costume costume1
if level = 0
hide
stop script
```

```
when I receive go
if level = 0
stop script
```

```
when I receive stop
hide
stop script
```

Example Games

Nuclear HenHouse



Traffic Jam



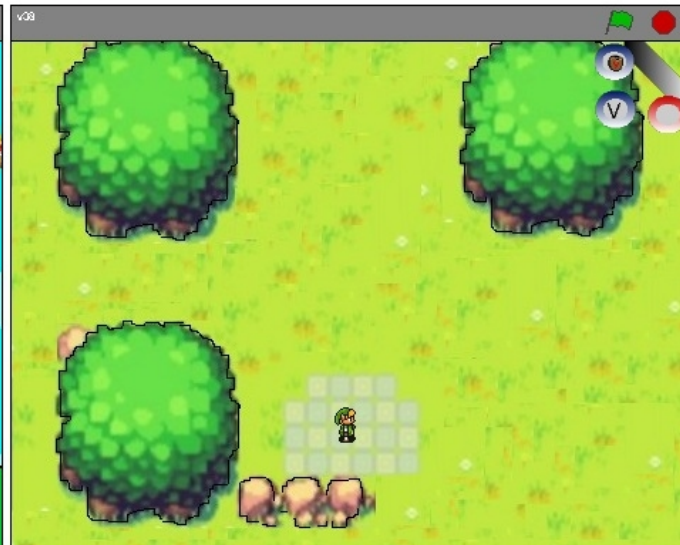
Space Bounty



Super Mario



Zelda



Donkey Kong



More example games

Note: With an account you can post your own games and projects on the Scratch website.

(<http://scratch.mit.edu/>)

<http://scratch.mit.edu/users/MrMeyer>

Additional labs, tutorials and **templates** can be found by going to the “Bridges Program” website and examining the “Game Programming and Design” lesson-plan:

http://bridges.brooklyn.cuny.edu/collegenow/modules/11_GameProgDesign/GameProgDesign.html

The End