Java Programming Project – Asteroids

# Notes from the ‘Project Illustration’ Video

I just took some bullet notes from the TA’s slides. There are no amazing pearls of wisdom here…

* Create the game window: Pane Class, Scene Class
* Create the ship: Polygon class (inherits “Node” class)(setTranslateX, setTranslateY methods)
* Turning the ship: setRotate method, setOnKeyPressed method, onKeyRelased event, AnimationTimer class (helps make animation smoother) (handle method)
* Moving the ship: Point2D class, getRotate method, Math class (Math.cos, Math.sin methods, Math.toRadians method)
* Creating the Asteroid: Same as ship player and it’s functionality
* The Collision Between Ship and Asteroid: Stop method, Shape.intersect method
* Multiple Asteroids: List of objects with random parameters
* Adding points: AtomicInteger class

# Getting Started with JavaFX

## What is Java FX?

“JavaFX is an open source, next generation client application platform for desktop, mobile and embedded systems built on Java. It is a collaborative effort by many individuals and companies with the goal of producing a modern, efficient, and fully featured toolkit for developing rich client applications.”

## Short Introduction to JavaFX:

<https://openjfx.io/openjfx-docs/>

I’m guessing we’ll be using “the JavaFX SDK”?? Unless anyone knows loads about Maven/Gradle and wants to teach the rest of us? JavaFX lets you run java applications on Windows, macOS, Linux etc. and appears to have two parts:

* The java parts (jars containing classes for the objects we’ll be using)
* A platform dependant part (the libraries (“.dll’s”, “.so’s” etc.) that do the magic/make the windows etc. appear on each individual OS.

My laptop and PC use MS Windows. I downloaded “javafx-sdk-17.0.2” from [here](https://gluonhq.com/products/javafx/) and installed it on my machine. To ‘install it’ I just:

* unzipped it to a folder
* added the ‘bin’ folder to my PATH environment variable

This is what I had to do to get it running/ to get any java programs to run with it. I’m sure the instructions are different (but similar) on macOS. If we have a mac user perhaps they might include the instructions here? Or if a Windows user knows a better way – just say so.

## Java FX Scene Builder:

There is a GUI tool for working on JavaFX GUI’s. It’s called Scene-Builder. It used to be distributed by Oracle but when the changed their licence terms (end of Java 8) it was taken over/supported by a company called Gluon. I downloaded it from [here](https://gluonhq.com/products/scene-builder/) and installed it. I installed it to a folder on my PC called “D:\JavaFXSceneBuilder\” – I don’t think it matters where you install it, it just matters (for Eclipse) that you know where you installed it.

## JavaFX Plugin for Eclipse (“e(fx)clipse”):

There is a JavaFX Plugin for Eclipse. I installed it. People seem to recommend it. I did not use it to create the Asteroids project. But it does allow you to integrate the JavaFX Scene Builder software into Eclipse (a GUI tool for working on our Asteroids GUI). To configure Eclipse to use it:

* Window – Preferences
  + JavaFX – SceneBuilder executable (click Browse and select your install folder)

# Hello/Demo JavaFX Applications:

## HelloFX App:

I keyed in the (openjfx) demo ‘HelloFX’ app detailed in the getting started guide [here](https://openjfx.io/openjfx-docs/) and to get that to compile/work I had to create a ‘lib’ folder in our project, add all the jars for JavaFX to that folder, then add the content of those jars to the project classpath. It seems to run fine.

## MainApp/FXMLController:

I keyed in this demo too and it worked. I didn’t follow all of what’s going on at this point– but I didn’t spend too long worrying about it, I just wanted to see it running. Once it was running I moved on! If it works for you, you should see a GUI window on screen that looks something like this when you run it:

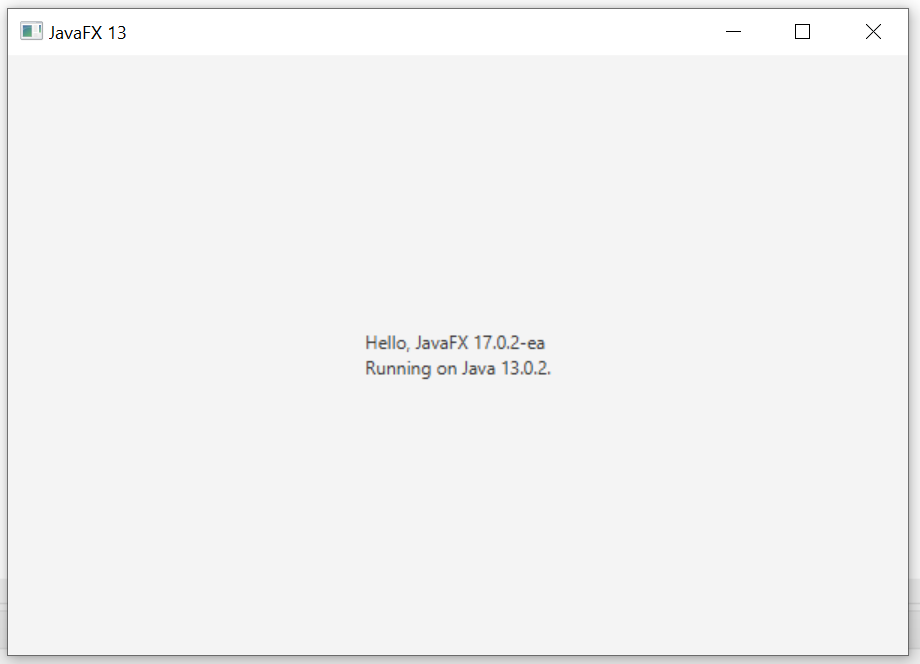


Figure 'MainApp' JavaFX Application Running

# JavaFX – Various Resources:

I have not looked at all of these by any stretch of the imagination. But I thought if I found stuff even remotely useful I’d list it here in case it was useful to the team

## Udemy Course (TL-DR!)

## https://www.udemy.com/course/crash-course-into-javafx-the-best-way-to-make-gui-apps/

## YouTube JavaFX for Beginners:

OK, this course is also “TL-DR” – but I watched a couple of them to get an idea of what was going on. I’m not wild about listening to the guy – but the videos are short and you learn one little thing per video normally.

https://www.youtube.com/playlist?list=PLS1QulWo1RIaUGP446\_pWLgTZPiFizEMq

# Asteroids

## Web Sites:

Following website does a start-to-finish course on how to program Asteroids in JavaFX. It’s very close to our project spec:

https://programming-f20.mooc.fi/part-14/3-larger-application-asteroids

## YouTube Resources:

An American professor called “Lee Stemkoski” did a two-part series on building asteroids using JavaFX. Two parts – two hours long (yikes) This was published two years ago. I’m sure the Prof./TA’s are well aware it is there. I thought it might give us a good starting point. We can try to incorporate all the tips ‘n tricks from the lectures into our design as we go through them.

<https://www.youtube.com/user/ProfStemkoski/search?query=asteroids>

This guy’s whole approach is to use Vectors (co-ordinates with respect to some basis) to calculate screen position for every object. About 12 minutes into the first video Lee uses some math to work out angles etc.. A screenshot of the little diagram he uses to calculate the angles is below, just in case it helps

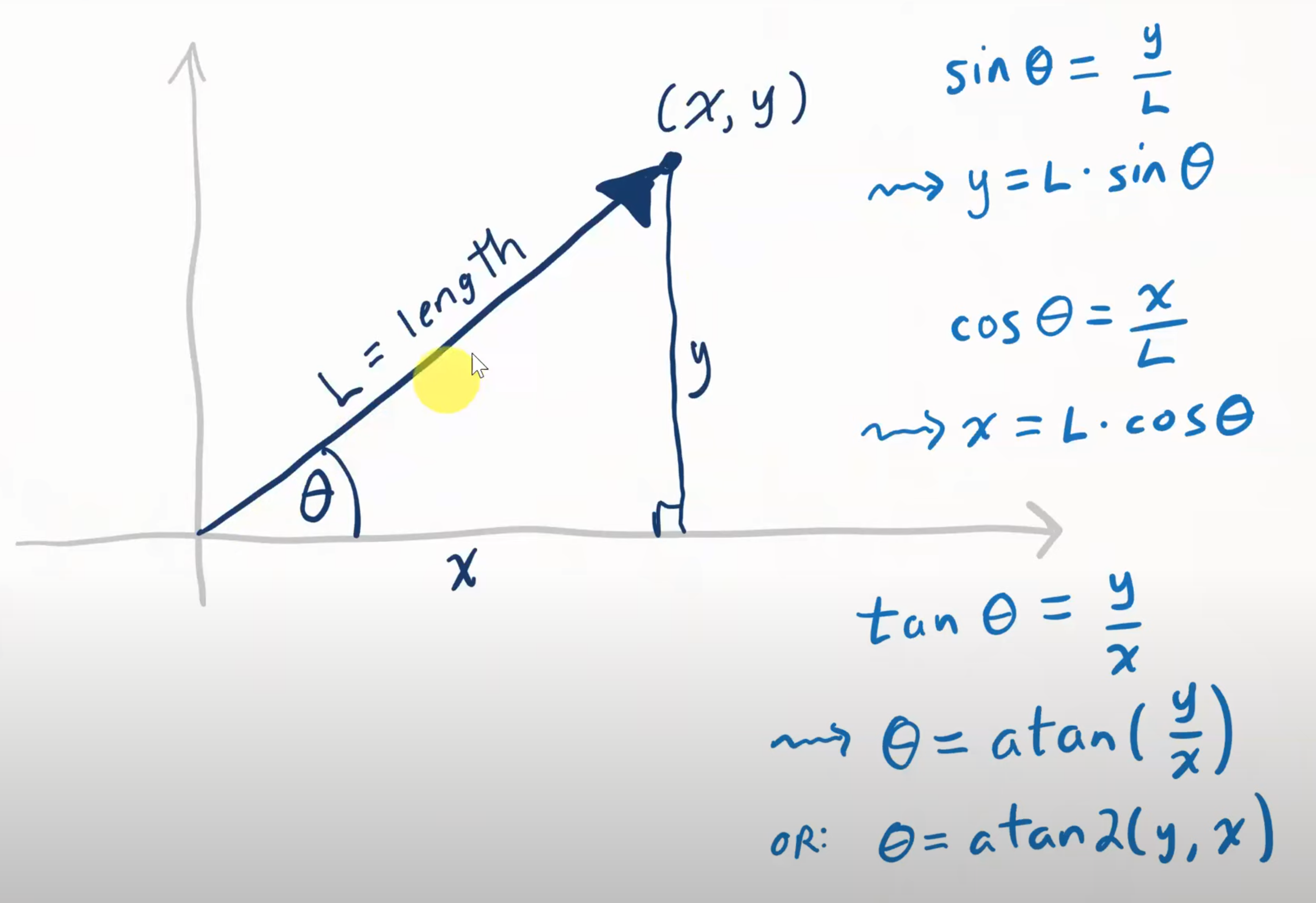


Figure Working out Vector Angle

Then again about 20 minutes into the first video he shows the trigonometry he is using for hit-box detection. NOTE Java FX uses a co-ordinate system starting at the TOP-LEFT - a traditional computer graphics "local" coordinate system in which the x axis increases to the right and the y axis increases downwards.

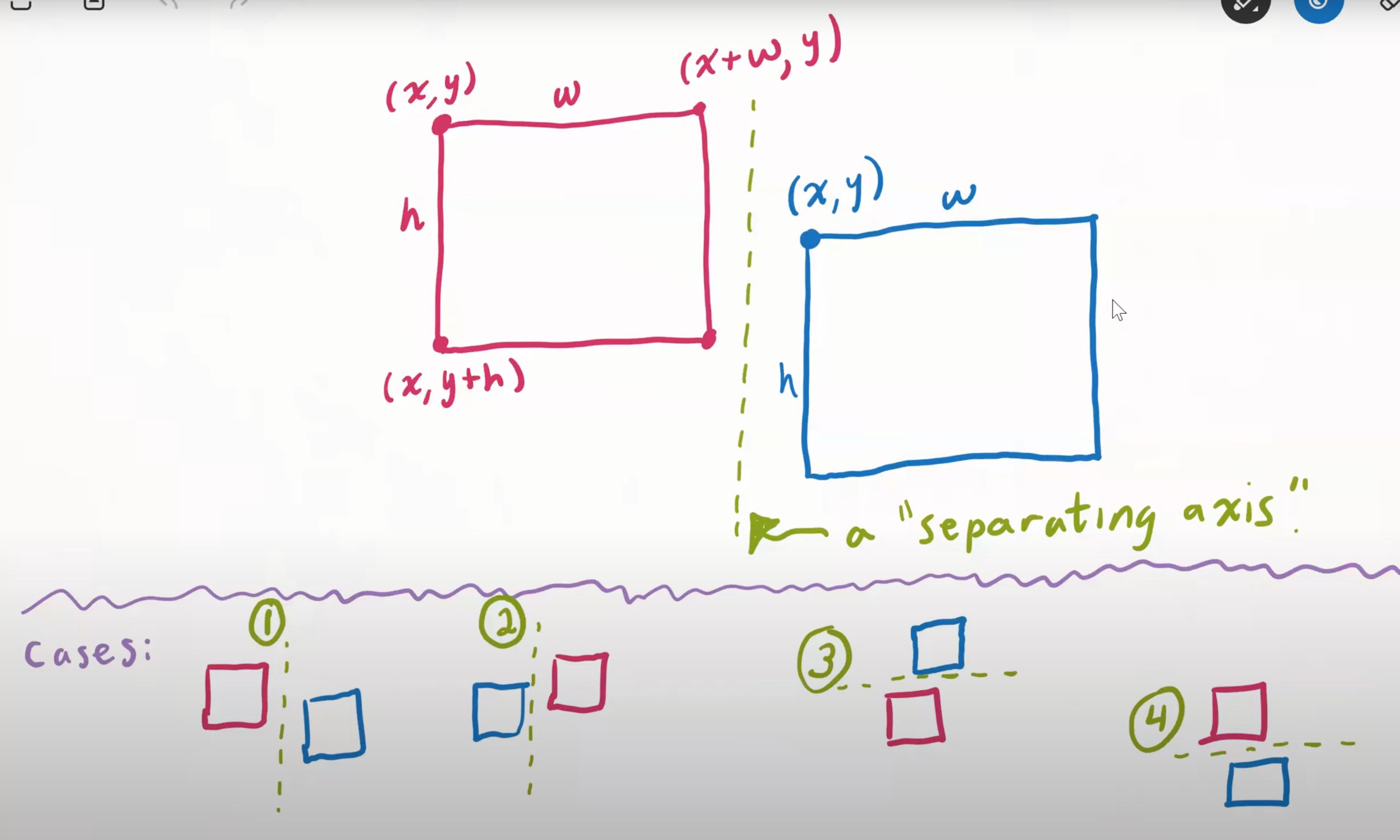


Figure Hitbox detection