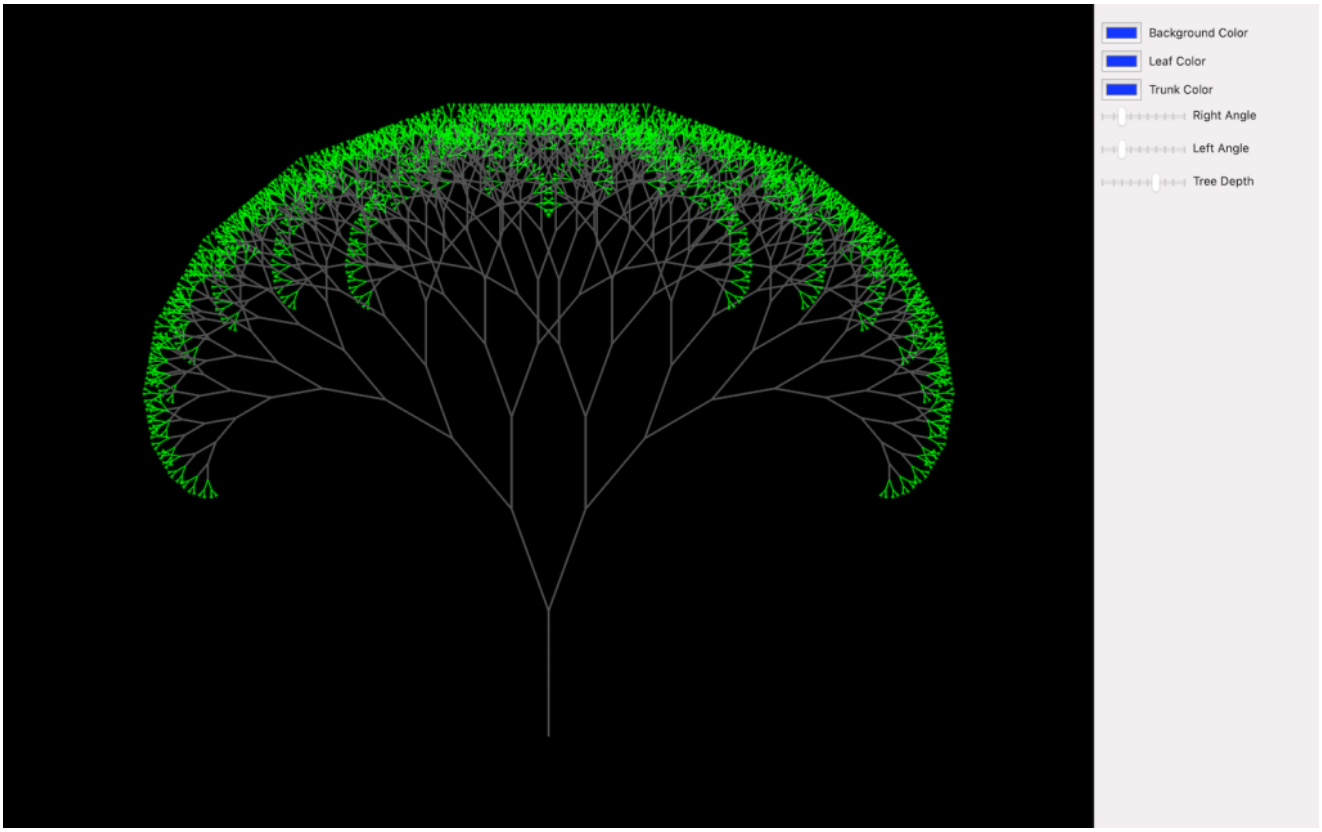
Creating a Fractal Tree in Swift

ddas.tech/creating-a-fractal-tree-in-swift/

September 19, 2022

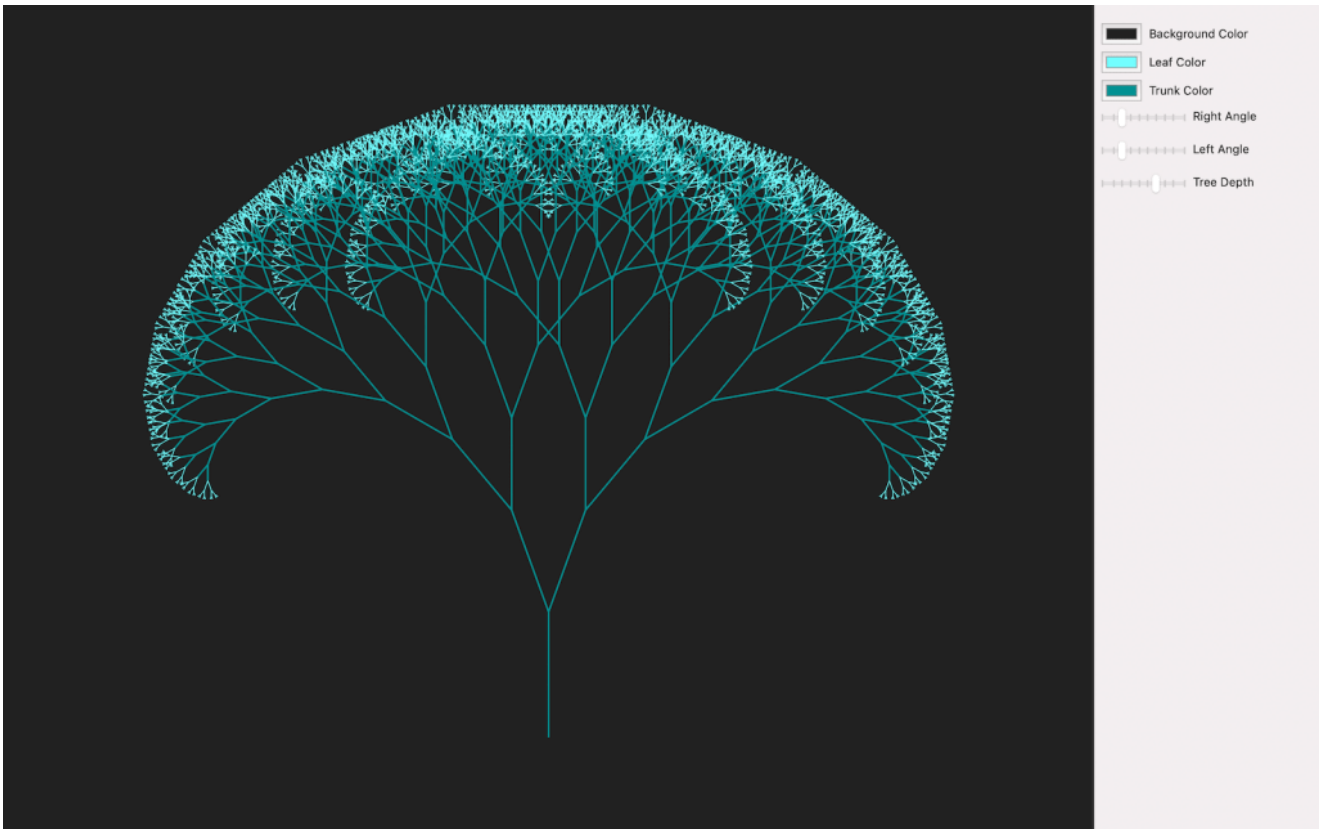
Created By: Debasis Das (Sep – 2022)

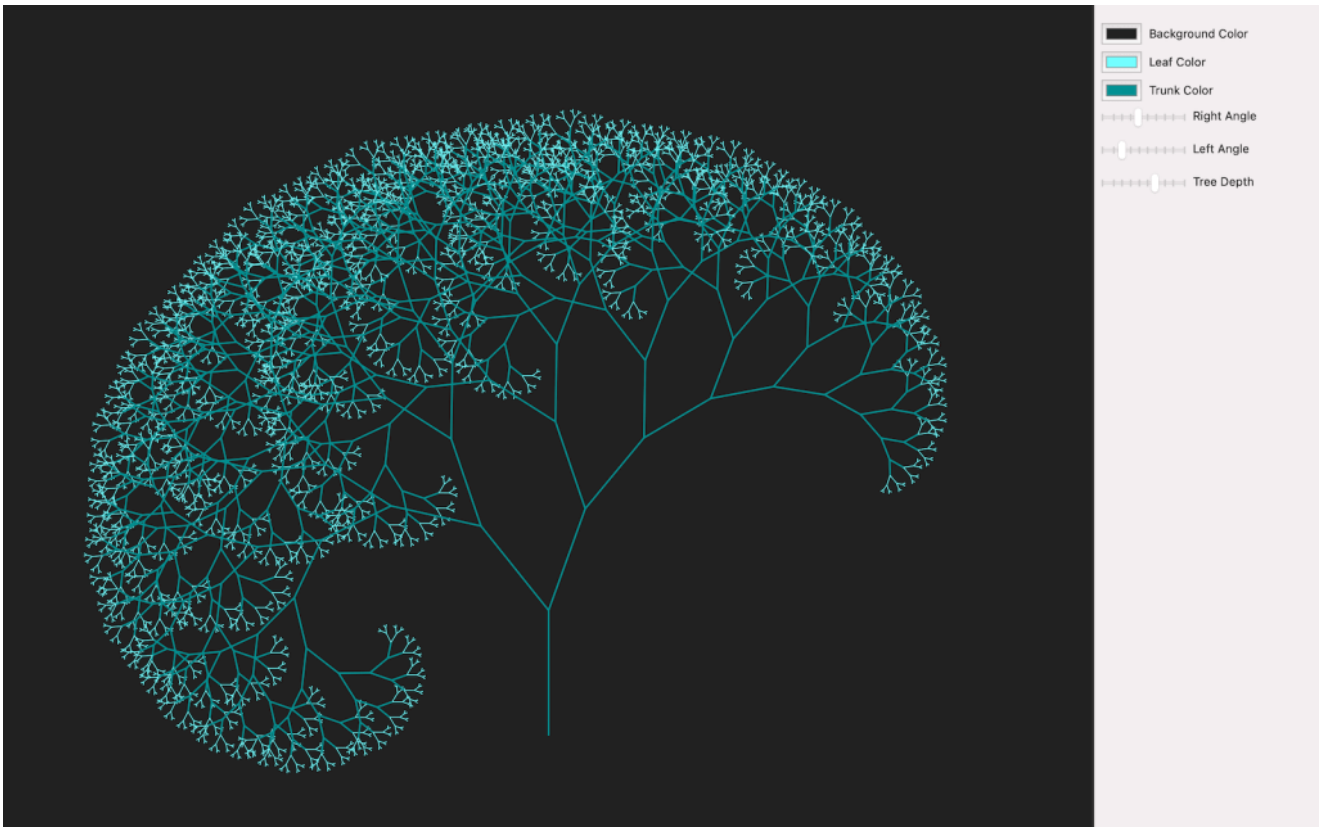
In this post we will create a sample application to draw a fractal tree in swift and customize the branch color, leaf color and angle of left and right branches and the tree depth. The outcome will have the below appearance

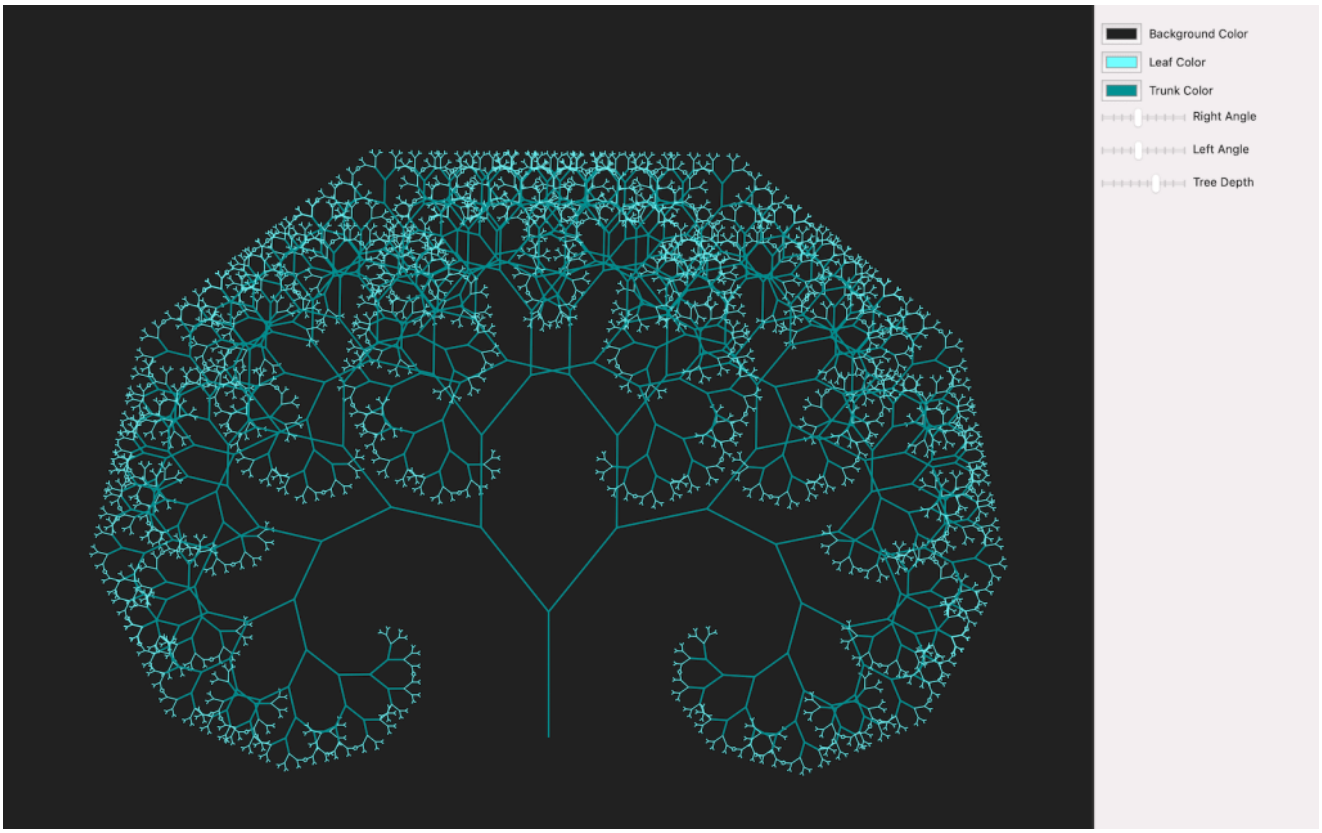


```
//  FractalTreeView.swift
//  SwiftFractalTree
//  Created by Debasis Das on 6/11/22.

import Cocoa

class FractalTreeView: NSView {
    let PI = 3.14156
    var leftAngle:Double = 20
    var rightAngle:Double = 20
    var treeDepth:Float = 12
    var backgroundColor:NSColor = NSColor.black
    var leafColor:NSColor = NSColor(calibratedRed: 0, green: 30, blue: 0, alpha: 1.0)
    var trunkColor:NSColor = NSColor.darkGray

    @IBOutlet weak var backgroundColorWell: NSColorWell!
    @IBOutlet weak var trunkColorWell: NSColorWell!
    @IBOutlet weak var leafColorWell: NSColorWell!

    @IBOutlet weak var leftAngleSlider: NSSlider!
    @IBOutlet weak var rightAngleSlider: NSSlider!
    @IBOutlet weak var treeDepthSlider: NSSlider!

    override class func awakeFromNib() {
        Swift.print("Awake From Nib")
    }


       @IBAction func changeDepth(sender:AnyObject){
           self.treeDepth = Float(sender.intValue)
           self.needsDisplay = true
       }

       @IBAction func changeLeftAngle(sender:AnyObject){
           self.leftAngle = sender.doubleValue
           self.needsDisplay = true
       }

       @IBAction func changeRightAngle(sender:AnyObject){
           self.rightAngle = sender.doubleValue
           self.needsDisplay = true
       }

       @IBAction func changeBackgroundColor(sender:AnyObject){
           self.backgroundColor = ((sender as? NSColorWell)?.color)!
           self.needsDisplay = true
       }

       @IBAction func changeLeafColor(sender:AnyObject){
           self.leafColor = ((sender as? NSColorWell)?.color)!
           self.needsDisplay = true
       }
```

```
       @IBAction func changeTrunkColor(sender:AnyObject){
           self.trunkColor = ((sender as? NSColorWell)?.color)!
           self.needsDisplay = true
       }

    override func draw(_ dirtyRect: NSRect) {
        super.draw(dirtyRect)
        self.backgroundColor.setFill()
        dirtyRect.fill()
        self.drawBranch(x1: Float(self.frame.size.width/2), y1: 100.0, angle: 90.0,
depth: self.treeDepth)
       }

       func drawBranch(x1:Float, y1:Float, angle:Double,depth:Float){
           var branchArmLength = depth
           if depth > 0 && depth < 3{             branchArmLength = depth * 0.7
}         else if depth > 3 && depth < 7{
               branchArmLength = depth * 0.7
           }
           else{
               branchArmLength = depth*0.7
           }
           if depth != 0{
               let x2 = x1 + (Float(cos(angle * PI/180)) * depth * branchArmLength)
               let y2 = y1 + (Float(sin(angle * PI/180)) * depth * branchArmLength)
            drawLine(lineWidth: depth, fromPoint: NSMakePoint(CGFloat(x1),
CGFloat(y1)), toPoint: NSMakePoint(CGFloat(x2), CGFloat(y2)))
            drawBranch(x1: x2, y1: y2, angle: angle - self.leftAngle, depth: depth-1)
            drawBranch(x1: x2, y1: y2, angle: angle + self.rightAngle, depth: depth-
1)
           }
       }

       func drawLine(lineWidth:Float, fromPoint:NSPoint, toPoint:NSPoint){
           let path = NSBezierPath()
           if (lineWidth < 5){
               self.leafColor.set()
               path.lineWidth = 1.0//CGFloat(lineWidth*0.2)
           }
           else{
               self.trunkColor.set()
               path.lineWidth = 2.0//CGFloat(lineWidth*0.5)
           }
        path.move(to: fromPoint)
        path.line(to: toPoint)
        path.stroke()
       }

}
```


Background Color
Leaf Color
Trunk Color
Right Angle
Left Angle
Tree Depth


Background Color
Leaf Color
Trunk Color
Right Angle
Left Angle
Tree Depth



You can check the Fractal tree implementation in HTML here