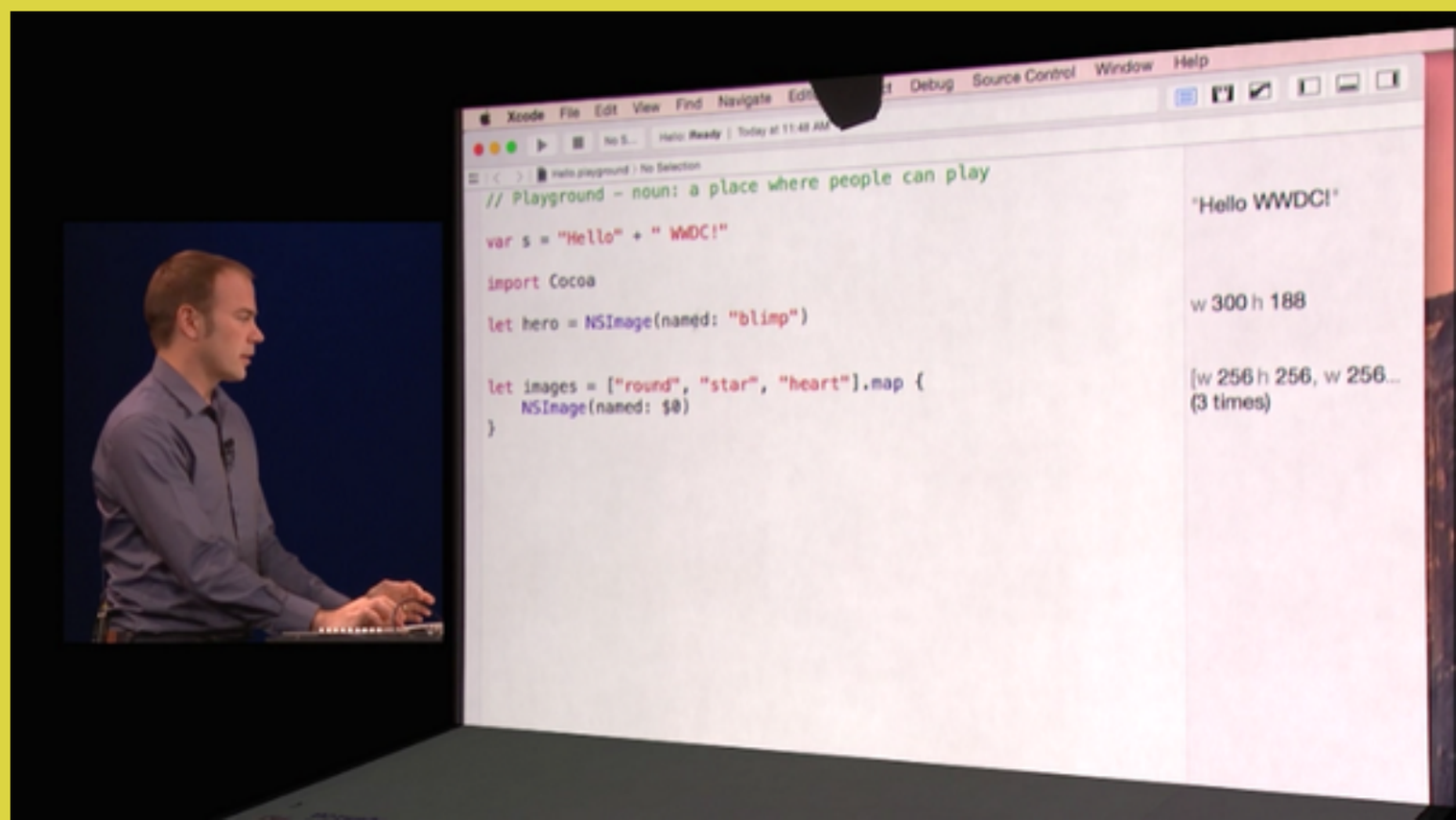
놀아요
Swift Playgrounds

발표자 : github.com/wookay
일시 : 2014년 7월 5일 토요일 1시~6시
장소 : 판교 H스퀘어 N동 4층 - NHN NEXT 홀

Chris Lattner - Playgrounds Demo

Apple SWIFT programming language demo at WWDC 2014
http://youtu.be/l62x80q_QP4?t=3m5s



인물과 배경

- Chris Lattner - LLVM
- Bret Victor
- Chris Granger - Light Table

Chris Lattner

Chris Lattner's Homepage
http://nondot.org/sabre/

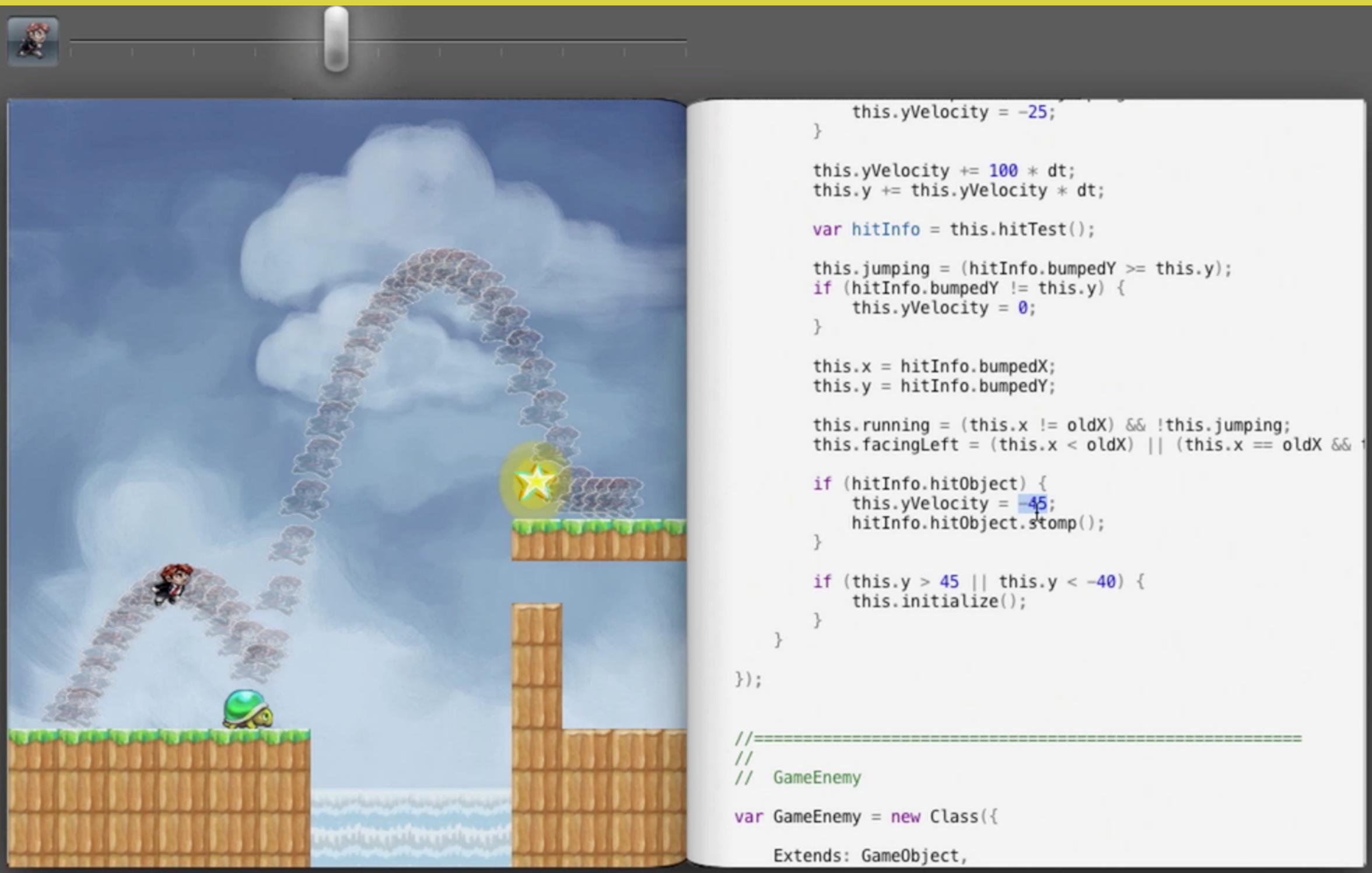
- The Xcode Playgrounds feature and REPL were a personal passion of mine, to make programming more interactive and approachable. The Xcode and LLDB teams have done a phenomenal job turning crazy ideas into something truly great. Playgrounds were heavily influenced by Bret Victor's ideas, by Light Table and by many other interactive systems. I hope that by making programming more approachable and fun, we'll appeal to the next generation of programmers and to help redefine how Computer Science is taught.

Bret Victor - Inventing on Principle

Friday, January 20th, 2012
http://vimeo.com/36579366

CUSEC (Canadian University Software Engineering Conference)
http://2012.cusec.net

Bret Victor - Inventing on Principle (2)



```
            this.yVelocity = -25;
        }

        this.yVelocity += 100 * dt;
        this.y += this.yVelocity * dt;

        var hitInfo = this.hitTest();

        this.jumping = (hitInfo.bumpedY >= this.y);
        if (hitInfo.bumpedY != this.y) {
            this.yVelocity = 0;
        }

        this.x = hitInfo.bumpedX;
        this.y = hitInfo.bumpedY;

        this.running = (this.x != oldX) && !this.jumping;
        this.facingLeft = (this.x < oldX) || (this.x == oldX &&

        if (hitInfo.hitObject) {
            this.yVelocity = -45;
            hitInfo.hitObject.stomp();
        }

        if (this.y > 45 || this.y < -40) {
            this.initialize();
        }
    }

});


//=========================================================
//
//  GameEnemy

var GameEnemy = new Class({

    Extends: GameObject,
```

Bret Victor - Inventing on Principle (3)

```
function binarySearch (key, array) {

    var low = 0;
    var high = array.length - 1;

    while (1) {

        var mid = floor((low + high)/2);
        var value = array[mid];

        if (value < key) {
            low = mid + 1;
        }
        else if (value > key) {
            high = mid - 1;
        }
        else {
            return mid;
        }
    }

}
```

```
   key = 'e'
 array = ['a','b','c','d','e','f']
   low = 0
  high = 5

   low =  0  |  3
  high =  5  |  5
   mid =  2  |  4
 value = 'c' | 'e'

   low =  3  |

 return      |  4
```

Bret Victor - Alesis

2003 ~ 2005

http://worrydream.com/#!/Alesis

* ALESIS ION
* ALESIS MICRON
* ALASIS FUSION

Alesis Micron - Quick demo

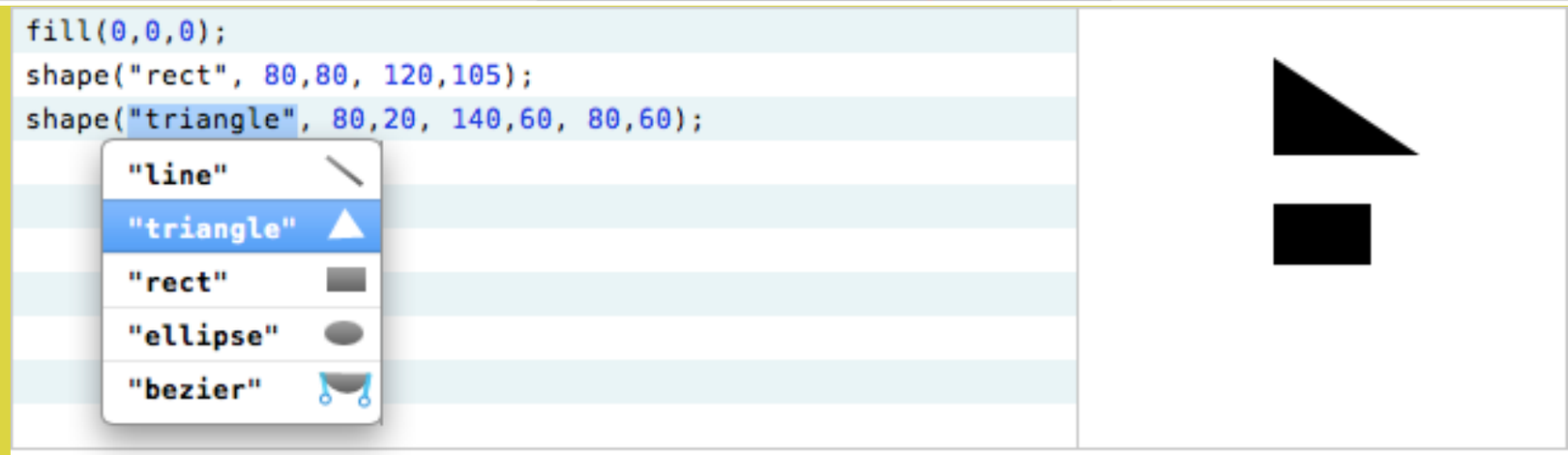
http://www.youtube.com/watch?v=nqoN1dpwA30

Bret Victor - Learnable Programming

September 2012

Learnable Programming

http://worrydream.com/LearnableProgramming/



Chris Granger - Light Table

12 Apr 2012

Light Table - a new IDE concept

http://www.chris-granger.com/2012/04/12/light-table---a-new-ide-concept/

- Bret Victor hinted at the idea that we can do much better than we are now - we can provide instant feedback, we can show you how your changes affect a system. And I discovered he was right.

Instant feedback

- In Inventing on Principle, Bret showed us that we could live-edit games and write binary search in an editor that is constantly evaluating and showing you what's going on. The lispers among us are used to using the REPL to have an environment where we can try things out. But we can do better - we can do it in place and instantaneously. For example here I type the code (+ 3 4) and we immediately see that it evaluates to 7 - no ctrl-enter or anything else.

Chris Granger - Light Table : Playground

24 Jun 2012

It's playtime - Light Table Playground released

http://www.chris-granger.com/2012/06/24/its-playtime/

Instarepl

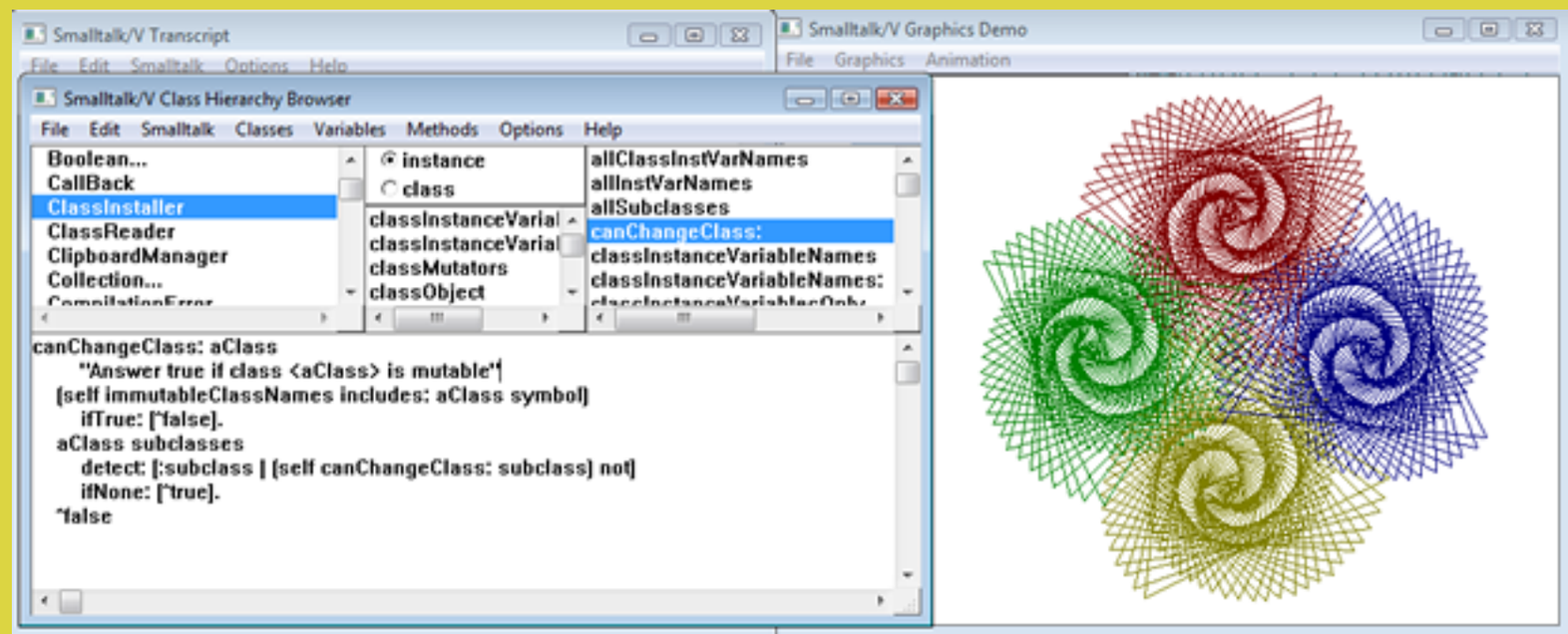
- real-time evaluation

```
;;Welcome to the instarepl.
;;Anything typed into this box will immediately be evaluated
;;with the result being shown on the right hand side.
;;For example:

(+ 3 4) => 7
(- 10 20) => -10

;;This allows you to very quickly see changes to all the
;;things you're working with. You'll notice that the code
;;on the right also has the variables' values filled in in
;;green, making it easy to see how information is flowing
;;through the program.

(defn my-add [3 45]
  (+ 3 45))

(my-add 3 45) => 48

;;If you end up stuck, you can press Ctrl+D to cancel
;;everything that is currently executing. Ctrl+R will also
;;reset the state of the GUI if you want to start over fresh.
```

Live Programming

A History of Live Programming
http://liveprogramming.github.io/liveblog/2013/01/a-history-of-live-A History of Live Programmingprogramming/

- **Live programming is an idea espoused by programming environments from the earliest days of computing, such as Lisp machines, Logo, Hypercard, and Smalltalk. In common with all these systems was liveness: Feedback is nearly instantaneous and evaluation is always accessible. For example, any part of the SmallTalk environment could be modified at any time and reflected instantly. Likewise, in HyperCard, the state of any object was considered to be live and editable at any time.**

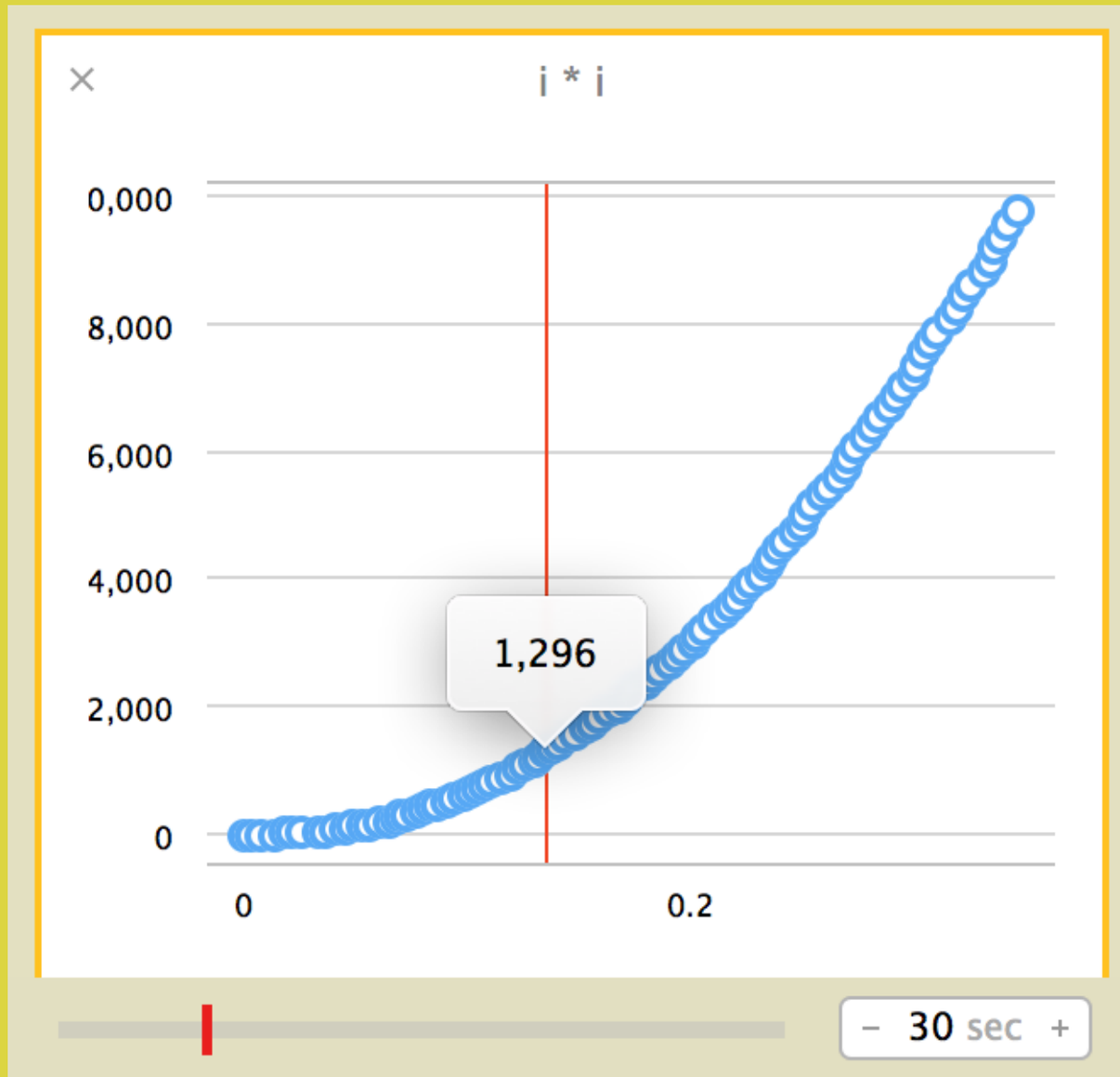


Playgrounds

* Timeline
* Quick Looks
* Settings
* XCPlayground

Timeline

- 타임라인
- 시간 흐름에 대한 값의 변화를 파악

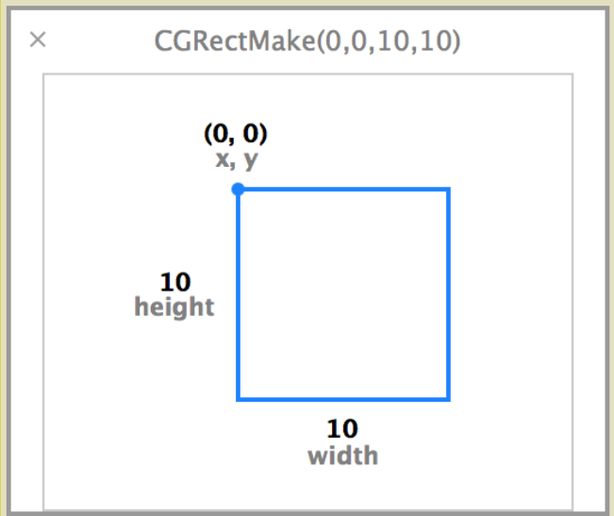


Quick Looks

- 빠른 보기

지원하는 타입

- Colors
- Strings (plain and attributed)
- Images
- Views
- Arrays and dictionaries
- Points, rects, sizes
- Bezier paths
- URLs
- Classes and structs



Settings

파일 인스펙터에서 플랫폼, 리소스 경로 설정

Platform
- OS X
- iOS

Resource Path
- Absolute Path
- Relative to Playground
- Inside Playground

사용
- NSImage(named:)
- NSBundle.pathForResource(_:ofType:)

XCPlayground

OS X에서만 되는 기능

캡쳐할 값 지정

- func XCPCaptureValue<T>(identifier: String, value: T)

타임라인에 라이브 뷰 보이기

- func XCPShowView(identifier: String, view: NSView)

비동기 지원

- func XCPSetExecutionShouldContinueIndefinitely(continueIndefinitely: Bool = true)

예제 모음

Session 408

- GuidedTour.playground
- MyPlayground.playground
- InsertionSort.playground
- TestTableViewCell.playground
- PlaygroundIconView.playground
- Async.playground

Session 408 - Swift Playgrounds

Xcode Engineers : Rick Ballard, Connor Wakamo

WWDC 2014 Session Videos - Apple Developer
https://developer.apple.com/videos/wwdc/2014/

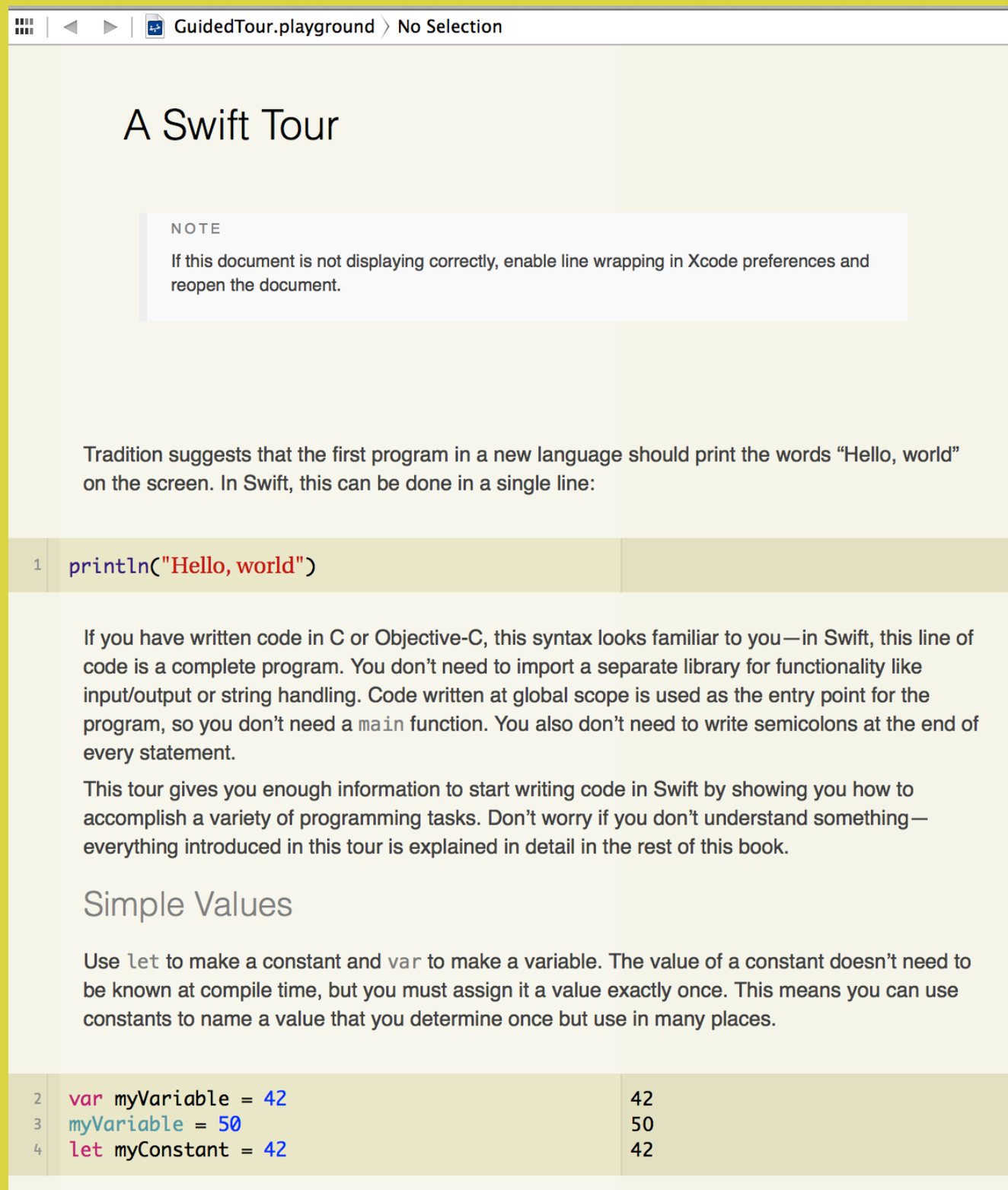
- 408_swift_playgrounds.pdf
- 408_hd_swift_playgrounds.mov

ASCIIwwdc - Swift Playgrounds
http://asciiwwdc.com/2014/sessions/408

WWDC-2014-Subtitle / 408 Swift Playgrounds [English].srt
https://github.com/walkingway/WWDC-2014-Subtitle/blob/master/408%20Swift%20Playgrounds%20%5BEnglish%5D.srt

GuidedTour.playground

```
GuidedTour.playground $ cat contents.xcplayground
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<playground version='3.0' sdk='macosx' allows-reset='YES'>
    <sections>
        <documentation relative-path='section-1.html'/>
        <documentation relative-path='section-2.html'/>
        <code source-file-name='section-3.swift'/>
        <documentation relative-path='section-4.html'/>
        <code source-file-name='section-5.swift'/>
        <documentation relative-path='section-6.html'/>
        <code source-file-name=‘section-7.swift'/>

        …

        <documentation relative-path='section-74.html'/>
        <code source-file-name='section-75.swift'/>
        <documentation relative-path='section-76.html'/>
        <code source-file-name='section-77.swift'/>
        <documentation relative-path='section-78.html'/>
        <code source-file-name='section-79.swift'/>
        <documentation relative-path='section-80.html'/>
        <code source-file-name='section-81.swift'/>
        <documentation relative-path='section-82.html'/>
        <code source-file-name='section-83.swift'/>
        <documentation relative-path='section-84.html'/>
        <code source-file-name='section-85.swift'/>
        <documentation relative-path='section-86.html'/>
        <code source-file-name='section-87.swift'/>
        <documentation relative-path='section-88.html'/>
        <documentation relative-path='section-89.html'/>
    </sections>
    <timeline fileName='timeline.xctimeline'/>
</playground>
```



MyPlayground.playground

Cocoa

- NSAttributedString
  - NSColor
  - NSFont
- NSImageView
  - NSImage

let attrStr = NSAttributedString(...(name: "BM JUA_TTF", size: 50)])

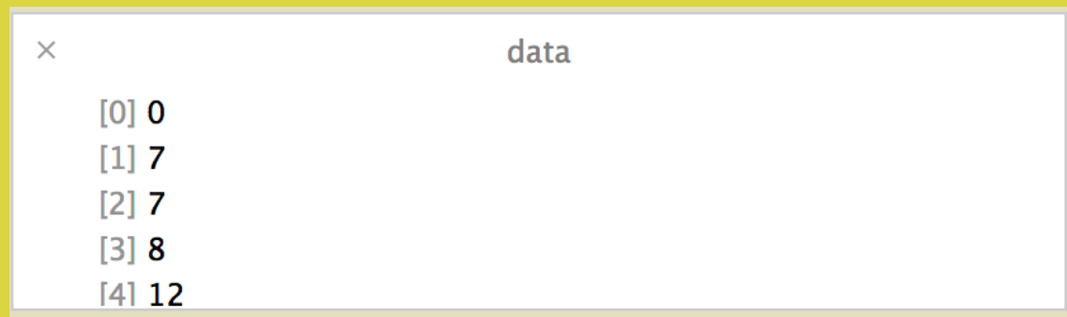
안녕 세미

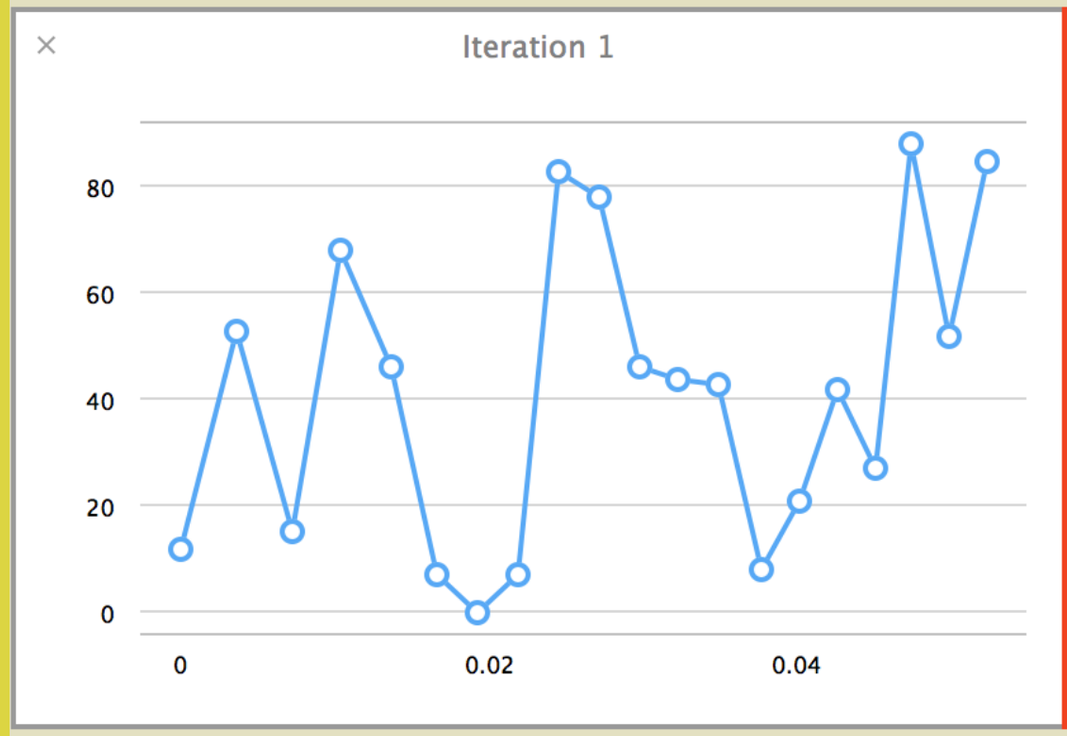
imageView.image = myImage

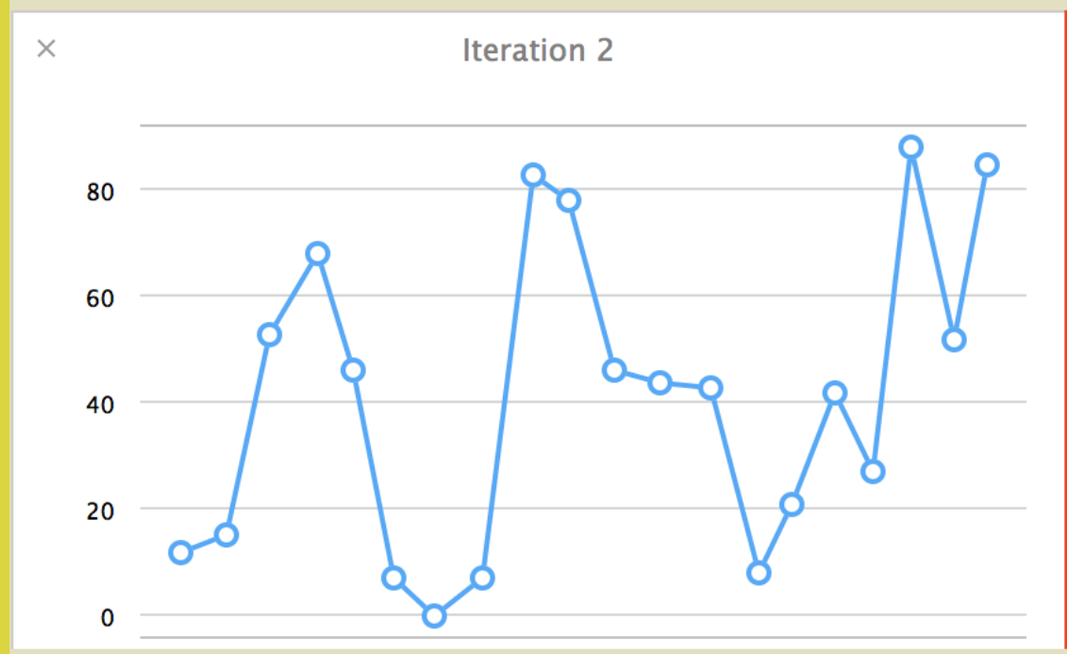
InsertionSort.playground

XCPlayground

- XCPCaptureValue







TestTableViewCell.playground

UIKit

- UITableView
- UITableViewCell

PlaygroundIconView.playground

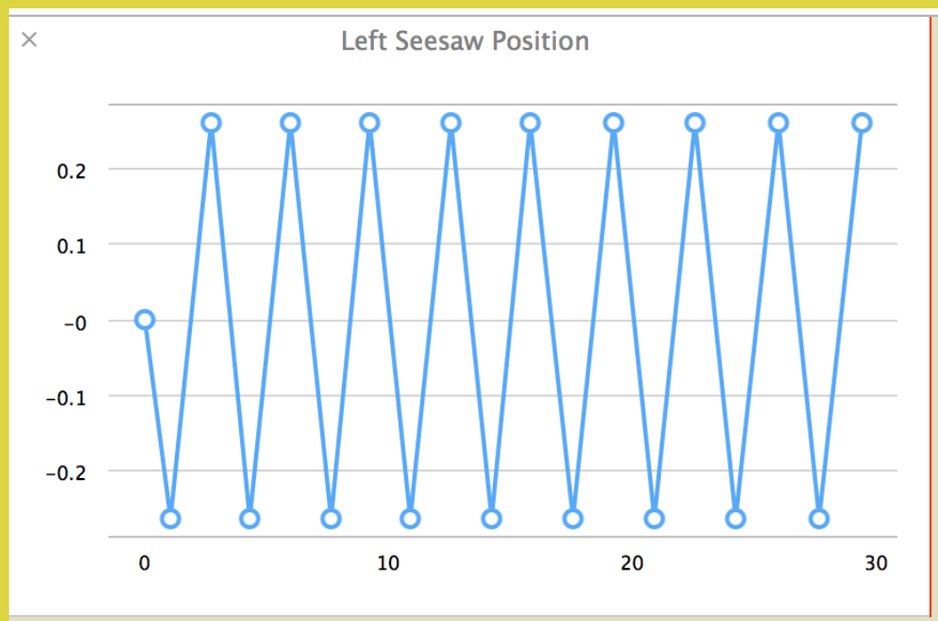
Cocoa

- CAShapeLayer
- CAAnimation
- NSBezierPath

XCPlayground

- XCPShowView

paradin / swift_tutorial
https://github.com/paradin/swift_tutorial



Async.playground

Cocoa

- NSURL
- NSURLSession
- NSURLSessionDelegate

XCPlayground

- XCPSetExecutionShouldContinueIndefinitely

Custom Quick Look Support

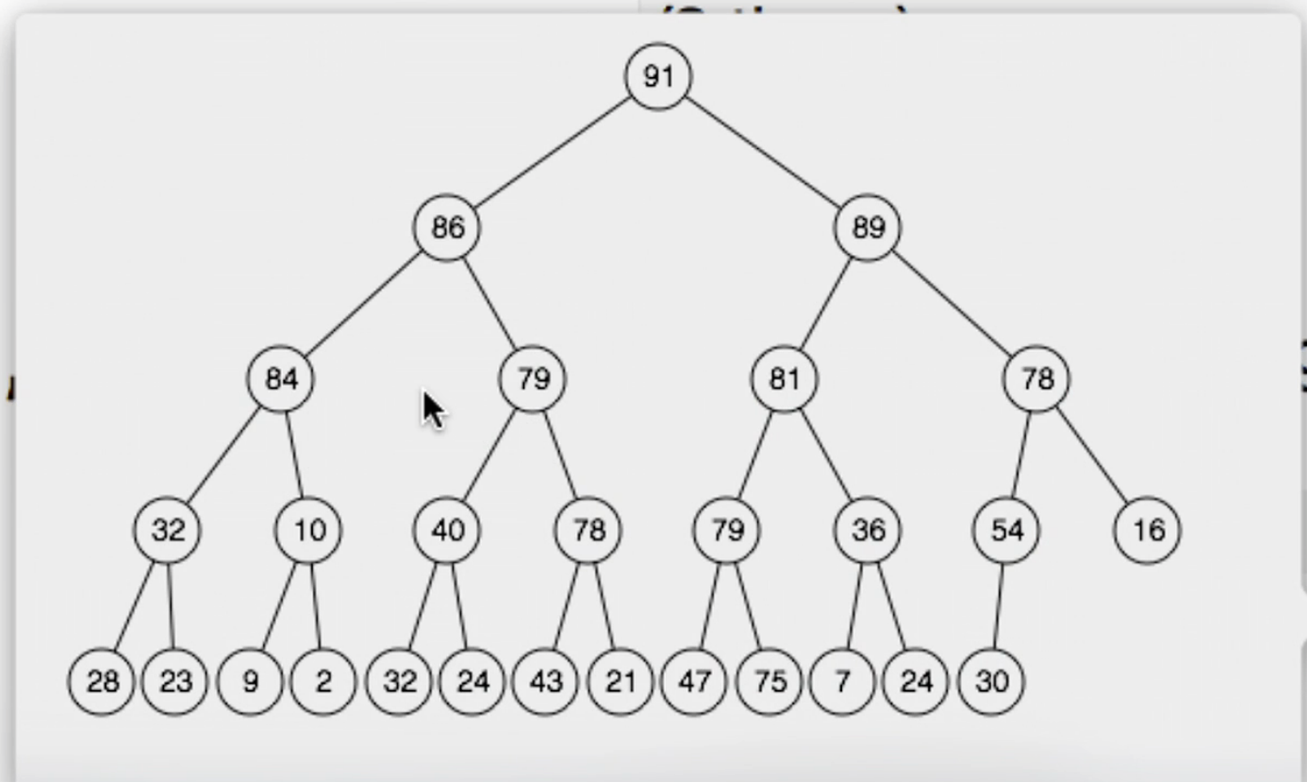
- debugQuickLookObject() 메서드 추가

QuickLook.xcodeproj

(2 times)

6, 28, 84,

9, 8...



SpriteKit

Balloons.playground › setupHero(_:_:)

```
func doDidMoveToView(scene : SKScene,
                     delegate : SKPhysicsContactDelegate) {


    // ================== Blimp Control =====================

    yOffsetForTime = { i in
        return 80 * sin(i / 10.0)
    }

    // ============= Scene Configuration ===================

    // Set up balloon lighting and per-pixel collisions.
    balloonConfigurator = { b in
        b.physicsBody.categoryBitMask = CONTACT_CATEGORY
        b.physicsBody.fieldBitMask = WIND_FIELD_CATEGORY
        b.lightingBitMask = BALLOON_LIGHTING_CATEGORY
    }

    // Load images for balloon explosion.
    balloonPop = (1...4).map {
        SKTexture(imageNamed: "explode_0\($0)")
    }

    // Install turbulant field forces.
    var turbulence = SKFieldNode.noiseFieldWithSmoothness(0.7,
                                 animationSpeed:0.8)
    turbulence.categoryBitMask = WIND_FIELD_CATEGORY
    turbulence.strength = 0.21
    scene.addChild(turbulence)

    cannonStrength = 210.0

    // ============== Scene Initialization =================

    // Do the rest of the setup and start the scene.
    setupHero(scene, delegate)
    setupFan(scene, delegate)
    setupCannons(scene, delegate)
}


func handleContact(bodyA : SKSpriteNode,
                   bodyB : SKSpriteNode)| {

    if (bodyA == hero) {
        bodyB.normalTexture = nil
        bodyB.runAction(removeBalloonAction)
    } else if (bodyB == hero) {
        bodyA.normalTexture = nil
        bodyA.runAction(removeBalloonAction)
    }
}
```

(Function)
(1058 times)

(Function)

(55 times)

[SKTexture, SKTexture, SKTe...
(4 times)

SKNoiseFieldNode

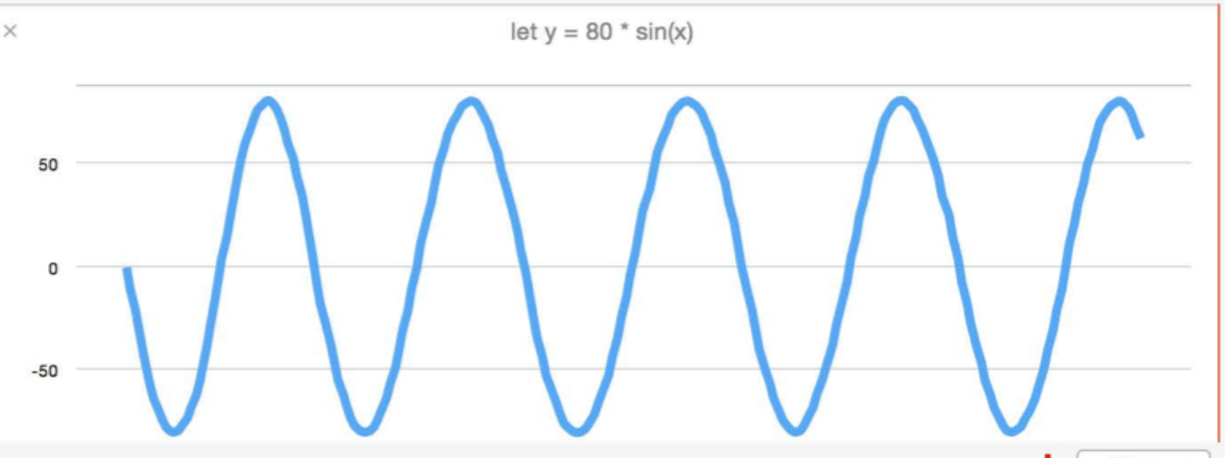
SKNoiseFieldNode
SKNoiseFieldNode
{GameScene {(Function)} {(F...

210.0

Timeline › Balloons.playground (Timeline)

Balloons

let y = 80 * sin(x)



Playgrounds Limitations

- Playground cannot be used for performance testing.
- Does not support User Interaction.
- Does not support On-device execution.
- Cannot use your app or framework code.
- Does not support custom entitlements.

Playgrounds Limitations (2)

추가 단점

- Xcode 자주 사망

질문 답변

쌩유

- 듣느라 수고하셨습니다
- 끝