```
import SpriteKit
import UIKit
import Foundation

class GameScene: SKScene {
    let node = SKLabelNode(fontNamed:"Chalkduster")
    let potionLabel = SKLabelNode(fontNamed:"Chalkduster")
    let counterLabel = SKLabelNode(fontNamed:"Chalkduster")
    let nodeTwo = SKLabelNode(fontNamed: "Chalkduster")
    let health = SKLabelNode(fontNamed: "Chalkduster")
    let resetLabel = SKLabelNode(fontNamed: "Chalkduster")
    let versionLabel = SKLabelNode(fontNamed: "Chalkduster")
    var potion = Int.random(min: 1, max: 75)
    let maxPotion = 75
    var charHealth = Int.random(min: 1, max: 100)
    let maxCharHealth = 100
    var spentPotion = 0
    var timer: dispatch_source_t!

    override func didMoveToView(view: SKView) {
        /* Setup your scene here */
        node.text = "Health:"
        node.fontSize = 45
        node.position = CGPoint(x:CGRectGetMidX(self.frame) - 50, y:CGRectGetMaxY(self.frame)
- 100)
        potionLabel.fontSize = 40
        potionLabel.position = CGPoint(x:CGRectGetMidX(self.frame) + 120,
y:CGRectGetMinY(self.frame) + 100)
        counterLabel.fontSize = 40
        counterLabel.position = CGPoint(x: CGRectGetMidX(self.frame) + 120, y:
CGRectGetMidY(self.frame) - 200)
        nodeTwo.text = "Potion left:"
        nodeTwo.fontSize = 35
        nodeTwo.position = CGPoint(x: CGRectGetMidX(self.frame) - 75, y:
CGRectGetMinY(self.frame) + 150)
        health.fontSize = 45
        health.position = CGPoint(x: CGRectGetMidX(self.frame) + 100, y:
CGRectGetMaxY(self.frame) - 100)
        resetLabel.text = "Resetting..."
        resetLabel.fontSize = 45
        resetLabel.position = CGPoint(x: CGRectGetMidX(self.frame), y:
CGRectGetMidY(self.frame))
        resetLabel.hidden = true
        versionLabel.text = "Version 1.1.1"
        versionLabel.fontSize = 15
        versionLabel.position = CGPoint(x: CGRectGetMinX(self.frame) + 350, y:
CGRectGetMaxY(self.frame) - 20)
        updateText()
```

```
    self.addChild(node)
    self.addChild(potionLabel)
    self.addChild(nodeTwo)
    self.addChild(health)
    self.addChild(counterLabel)
    self.addChild(resetLabel)
    self.addChild(versionLabel)
}

override func touchesBegan(touches: Set<UITouch>, withEvent event: UIEvent?) {
   /* Called when a touch begins */
    //=====OPTIONAL CODE START=====
    //For a bonus, remove "/*" and "*/" at the start and the end of the code below

    /*for touch in touches {
        let location = touch.locationInNode(self)

        let sprite = SKSpriteNode(imageNamed:"Spaceship")

        sprite.xScale = 1
        sprite.yScale = 1
        sprite.position = location

        let resize = SKAction.scaleTo(0.5, duration: 0.25)
        let fadeAction = SKAction.fadeOutWithDuration(0.5)
        let removeAction = SKAction.removeFromParent()
        let fullAction = SKAction.sequence([resize, fadeAction, removeAction])

        self.addChild(sprite)

        sprite.runAction(fullAction)
    }*/

    //=====OPTIONAL CODE END=====
    if charHealth == 100 {
        sleep(1)
        charHealth = Int.random(min: 1, max: 100)
        potion = Int.random(min: 1, max: 75)
        updateText()
    } else if potion == 0 {
        sleep(1)
        charHealth = Int.random(min: 1, max: 100)
        potion = Int.random(min: 1, max: 75)
        updateText()
    } else {
        chugPotion()
    }
}
```

```
    override func update(currentTime: NSTimeInterval) {
        /* Called before each frame is rendered */
    }

    func startTimer() {
        let queue = dispatch_queue_create("com.domain.app.timer", nil)
        timer = dispatch_source_create(DISPATCH_SOURCE_TYPE_TIMER, 0, 0, queue)
        dispatch_source_set_timer(timer, DISPATCH_TIME_NOW, 1 * NSEC_PER_SEC / 50, 0 *
NSEC_PER_SEC) // 50 repeats per second, 0 delay on start
        dispatch_source_set_event_handler(timer) {
            self.charHealth += 1
            self.potion -= 1
            self.updateText()
            if self.charHealth == 100 {
                self.resetStats()
            }
            if self.potion == 0 {
                self.resetStats()
            }
        }
        dispatch_resume(timer)
    }

    func stopTimer() {
        dispatch_source_cancel(timer)
        timer = nil
    }

    func chugPotion() {
        /*spentPotion = potion
        charHealth = charHealth + (maxCharHealth - (maxCharHealth - spentPotion))
        potion = potion - spentPotion
        if charHealth > 100 {
            potion = charHealth - 100
            charHealth = charHealth - (charHealth - 100)
        }*/
        userInteractionEnabled = false
        if potion > 0 {
            if charHealth < 100 {
                startTimer()
            }
        }
    }

    func updateText() {
        let percentCalculate = Int((potion * 100) / maxPotion)
        if percentCalculate < 70 && percentCalculate >= 30 {
```

```
            counterLabel.fontColor = UIColor.yellowColor()
            potionLabel.fontColor = UIColor.yellowColor()
        } else if percentCalculate < 30 && percentCalculate > 0 {
            counterLabel.fontColor = UIColor.redColor()
            potionLabel.fontColor = UIColor.redColor()
        } else if percentCalculate <= 0 {
            counterLabel.fontColor = UIColor.darkGrayColor()
            potionLabel.fontColor = UIColor.darkGrayColor()
        } else {
            counterLabel.fontColor = UIColor.greenColor()
            potionLabel.fontColor = UIColor.greenColor()
        }
        if charHealth < 70 && charHealth >= 30 {
            health.fontColor = UIColor.yellowColor()
        } else if charHealth < 30 && charHealth > 0 {
            health.fontColor = UIColor.redColor()
        } else if charHealth <= 0 {
            health.fontColor = UIColor.darkGrayColor()
        } else {
            health.fontColor = UIColor.greenColor()
        }
        potionLabel.text = "\(potion)/\(maxPotion)"
        counterLabel.text = "\(percentCalculate)%"
        health.text = "\(charHealth)"
    }

    func resetStats() {
        stopTimer()
        sleep(1)
        resetLabel.hidden = false
        sleep(1)
        charHealth = Int.random(min: 1, max: 100)
        potion = Int.random(min: 1, max: 75)
        updateText()
        resetLabel.hidden = true
        userInteractionEnabled = true
    }
}
```