Chapter3

▪66ページ
String型をInt型に変換する

Swift2.0になり、toInt( )が使えなくなりました。String型をInt型に変換する場合は下記の方法で行います。

修正前　var 変数名 = String 型変数 .toInt( )

修正後　var 変数名 = Int(String型変数)

▪67ページ
String型をInt型に変換する

Swift2.0になり、toInt( )が使えなくなりました。String型をInt型に変換する場合は下記の方法で行います。

修正前　var int3 = string1.toInt()

修正後　var int3 = Int(string1)

▪71ページ
String型変数の文字数を調べる

Swift2.0になりutf16Countが使えなくなりました。String型変数の文字数を調べる場合は下記の方法で行います。

修正前　変数.utf16Count

修正後　変数.characters.count

修正前　str.utf16Count

修正後　str.characters.count

Chapter4

▪111ページ
Xcode7になり、値が変わらない変数は定数で宣言しないとWarning(黄色の三角マーク)が出るようになりました。この場合の変数はvarではなくletで宣言します。

修正前　var randomNumber = arc4random() % 3

修正後　let randomNumber = arc4random() % 3

Chapter5

▪122ページ

Swift2.0になり、stringByAppendingPathComponentが使えなくなりました。またAVAudioPlayerクラスのインスタンスはdo - try - catchという構文を使って作成するようになりました。

修正前

```
let soundPath = NSBundle.mainBundle().bundlePath.
        stringByAppendingPathComponent("cowbell.mp3")
let url:NSURL? = NSURL.fileURLWithPath(soundPath)
player = AVAudioPlayer(contentsOfURL:url, error:nil)
player?.play()
```

修正後

```
let url  = NSBundle.mainBundle().bundleURL.URLByAppendingPathComponent("cowbell.mp3")
do {
        try player = AVAudioPlayer(contentsOfURL: url)
}catch{
        print("Error!")
}
player?.play()
```

▪129ページ

122ページの修正と同様に129ページの音を鳴らすためのコードも下記の通り変更になります。

修正前

```
let soundPath =
NSBundle.mainBundle().bundlePath.stringByAppendingPathComponent(soundName)
let url:NSURL? = NSURL.fileURLWithPath(soundPath)
player = AVAudioPlayer(contentsOfURL:url, error:nil)
```

修正後

```
let url  = NSBundle.mainBundle().bundleURL.URLByAppendingPathComponent(soundName)
do {
        try player = AVAudioPlayer(contentsOfURL: url)
}catch{
        print("Error!")
}
```

▪134、135、140、146ページ

Swift2.0になりprintlnが使えなくなりました。デバッグエリアに情報を表示する場合はprintを使用します（146ページのソースコードの例）。

修正前

```
switch sender.tag{
case 1:
   println("ワイングラスボタン\(sender.tag)")
   sound = "1.mp3"
   transform = CGAffineTransformMakeTranslation(0, -20)
case 2:
   println("ワイングラスボタン\(sender.tag)")
   sound = "2.mp3"
   transform = CGAffineTransformMakeScale(1.05, 1.05)
case 3:
   println("ワイングラスボタン\(sender.tag)")
   sound = "3.mp3"
   transform = CGAffineTransformMakeRotation(CGFloat(0.25*M_PI))
case 4:
   println("ワイングラスボタン\(sender.tag)")
   sound = "4.mp3"
   transform = CGAffineTransformMakeScale(0.95, 0.95)
case 5:
   println("ワイングラスボタン\(sender.tag)")
   sound = "5.mp3"
   transform = CGAffineTransformMakeScale(-1, 1)
default:
   println("どのボタンでもありません")
}
```

修正後

```
switch sender.tag{
case 1:
   print("ワイングラスボタン\(sender.tag)")
   sound = "1.mp3"
   transform = CGAffineTransformMakeTranslation(0, -20)
case 2:
   print("ワイングラスボタン\(sender.tag)")
   sound = "2.mp3"
   transform = CGAffineTransformMakeScale(1.05, 1.05)
case 3:
   print("ワイングラスボタン\(sender.tag)")
   sound = "3.mp3"
   transform = CGAffineTransformMakeRotation(CGFloat(0.25*M_PI))
case 4:
   print("ワイングラスボタン\(sender.tag)")
   sound = "4.mp3"
   transform = CGAffineTransformMakeScale(0.95, 0.95)
case 5:
   print("ワイングラスボタン\(sender.tag)")
   sound = "5.mp3"
   transform = CGAffineTransformMakeScale(-1, 1)
default:
```

```
        print("どのボタンでもありません")
    }
```

▪139ページ

122ページの修正と同様に139ページの音を鳴らすためのコードも下記の通り変更になります。

修正前

```
    let soundPath =
NSBundle.mainBundle().bundlePath.stringByAppendingPathComponent(soundName)
    let url:NSURL? = NSURL.fileURLWithPath(soundPath)
    player = AVAudioPlayer(contentsOfURL:url, error:nil)
    player?.numberOfLoops = -1
    player?.prepareToPlay()
    player?.play()
```

修正後

```
    let url  = NSBundle.mainBundle().bundleURL.URLByAppendingPathComponent(soundName)
    do {
        let player = try AVAudioPlayer(contentsOfURL: url)
        soundArray.append(player)
        player.delegate = self
        player.prepareToPlay()
        player.play()
    }catch{
        print("Error!")
    }
```

▪144ページ

Swift2.0になりfindが使えなくなりました。配列のなかの変数を取り出す場合は下記の方法で行います。

修正前

```
func audioPlayerDidFinishPlaying(player: AVAudioPlayer!, successfully flag: Bool) {
    var i:Int = find(soundArray, player)!
    soundArray.removeAtIndex(i)
}
```

修正後

```
func audioPlayerDidFinishPlaying(player: AVAudioPlayer, successfully flag: Bool) {
    let i:Int = soundArray.indexOf(player)!
    soundArray.removeAtIndex(i)
}
```

▪145ページ

値が変わらない変数は定数で宣言します。

修正前

```
var duration:Double = 0.5
```

修正後

```
let duration:Double = 0.5
```

Chapter6

▪163ページ

値が変わらない変数は定数で宣言します。

修正前

```
for var y = 0; y < yButtonCount; y++ {
    for var x = 0; x < xButtonCount; x++ {
        var button = UIButton()
        var buttonWidth = ( screenWidth - ( buttonMargin * ( Double(xButtonCount) + 1 ) ) ) /
Double(xButtonCount)
        var buttonHeight =
        ( screenHeight - resultArea - ( ( buttonMargin * Double(yButtonCount) + 1 ) ) ) /
Double(yButtonCount)
        var buttonPositionX =
        ( screenWidth - buttonMargin ) / Double(xButtonCount) * Double(x) + buttonMargin
        var buttonPositionY =
        ( screenHeight - resultArea - buttonMargin ) / Double(yButtonCount) * Double(y) +
buttonMargin + resultArea
        button.frame = CGRect(x:buttonPositionX,y:
buttonPositionY,width:buttonWidth,height:buttonHeight)
        button.backgroundColor = UIColor.greenColor()
        self.view.addSubview(button)
    }
}
```

修正後

```
for var y = 0; y < yButtonCount; y++ {
    for var x = 0; x < xButtonCount; x++ {
        let button = UIButton()
        let buttonWidth = ( screenWidth - ( buttonMargin * ( Double(xButtonCount) + 1 ) ) ) /
Double(xButtonCount)
        let buttonHeight =
        ( screenHeight - resultArea - ( ( buttonMargin * Double(yButtonCount) + 1 ) ) ) /
Double(yButtonCount)
        let buttonPositionX =
        ( screenWidth - buttonMargin ) / Double(xButtonCount) * Double(x) + buttonMargin
        let buttonPositionY =
        ( screenHeight - resultArea - buttonMargin ) / Double(yButtonCount) * Double(y) +
buttonMargin + resultArea
        button.frame = CGRect(x:buttonPositionX,y:
buttonPositionY,width:buttonWidth,height:buttonHeight)
        button.backgroundColor = UIColor.greenColor()
        self.view.addSubview(button)
```

```
    }
```

■168、170ページ

Swift2.0になりprintlnが使えなくなりました。デバッグエリアに情報を表示する場合はprintを使用します。また値が変わらない変数は定数で宣言します（168ページのソースコードの例）。

修正前

```
func buttonTapped(sender:UIButton){
    var tappedButtonTitle:String = sender.currentTitle!
   println("\(tappedButtonTitle)ボタンが押されました！")
    }
```

修正後

```
func buttonTapped(sender:UIButton){
    let tappedButtonTitle:String = sender.currentTitle!
    print("\(tappedButtonTitle)ボタンが押されました！")
    }
```

■170ページ

修正前

```
func numberButtonTapped(tappedButtonTitle:String){
    println("数字ボタンタップ:\(tappedButtonTitle)")
}
func operatorButtonTapped(tappedButtonTitle:String){
    println("演算子ボタンタップ:\(tappedButtonTitle)")
}
func equalButtonTapped(tappedButtonTitle:String){
    println("等号ボタンタップ:\(tappedButtonTitle)")
}
func clearButtonTapped(tappedButtonTitle:String){
    println("クリアボタンタップ:\(tappedButtonTitle)")
}
```

修正後

```
func numberButtonTapped(tappedButtonTitle:String){
    print("数字ボタンタップ:\(tappedButtonTitle)")
}
func operatorButtonTapped(tappedButtonTitle:String){
    print("演算子ボタンタップ:\(tappedButtonTitle)")
}
func equalButtonTapped(tappedButtonTitle:String){
    print("等号ボタンタップ:\(tappedButtonTitle)")
}
func clearButtonTapped(tappedButtonTitle:String){
    print("クリアボタンタップ:\(tappedButtonTitle)")
}
```

▪172ページ

値が変わらない変数は定数で宣言します。

修正前

```
var tappedButtonNum:Double =(tappedButtonTitle as NSString).doubleValue
```

修正後

```
let tappedButtonNum:Double =(tappedButtonTitle as NSString).doubleValue
```

▪173、176ページ

Swift2.0になりprintlnが使えなくなりました。デバッグエリアに情報を表示する場合はprintを使用します（173ページのソースコードの例）。

修正前

```
switch operatorId {
case "+":
   result = number2.decimalNumberByAdding(number1)
case "-":
   result = number2.decimalNumberBySubtracting(number1)
case "×":
   result = number2.decimalNumberByMultiplyingBy(number1)
case "÷":
   if(number1.isEqualToNumber(0)){
      number1 = 0
      resultLabel.text = "無限大"

      return
   } else {
      result = number2.decimalNumberByDividingBy(number1)
   }
default:
   println("その他")
}
```

修正後

```
switch operatorId {
case "+":
   result = number2.decimalNumberByAdding(number1)
case "-":
   result = number2.decimalNumberBySubtracting(number1)
case "×":
   result = number2.decimalNumberByMultiplyingBy(number1)
case "÷":
   if(number1.isEqualToNumber(0)){
      number1 = 0
      resultLabel.text = "無限大"

      return
   } else {
      result = number2.decimalNumberByDividingBy(number1)
   }
default:
```

```
            print("その他")
        }
```

▪177ページ

Swift2.0になりprintlnが使えなくなりました。デバッグエリアに情報を表示する場合はprintを使用します。

修正前

```
for variable in array {
        if variable is Int{
                println("\(variable) は Int 型 ")
    }
    if variable is Double{
                println("\(variable) は Double 型 ")
        }
        if variable is String{
                println("\(variable) は String 型 ")
        }
}
```

修正後

```
for variable in array {
        if variable is Int{
                print("\(variable) は Int 型 ")
    }
    if variable is Double{
                print("\(variable) は Double 型 ")
        }
        if variable is String{
                print("\(variable) は String 型 ")
        }
}
```

▪179ページ

値が変わらない変数は定数で宣言します。

修正前

```
                var gradient = CAGradientLayer()
                gradient.frame = button.bounds
                var arrayColors = [
                    colorWithRGBHex(0xFFFFFF, alpha: 1.0).CGColor as AnyObject,
                    colorWithRGBHex(0xCCCCCC, alpha: 1.0).CGColor as AnyObject]
```

修正後

```
                let gradient = CAGradientLayer()
                gradient.frame = button.bounds
                let arrayColors = [
```

```
colorWithRGBHex(0xFFFFFF, alpha: 1.0).CGColor as AnyObject,
colorWithRGBHex(0xCCCCCC, alpha: 1.0).CGColor as AnyObject]
```

Chapter7

▪201ページ

CSVファイルなどの外部データを読み込む場合、引数にエラーを設定して初期化することができなくなりました。do - try - catch構文を使って読み込みを行います。

修正前

```
let csvBundle = NSBundle.mainBundle().pathForResource("start", ofType: "csv")
var encodingError:NSError? = nil
let csvData = NSString(contentsOfFile: csvBundle!, encoding: NSUTF8StringEncoding, error:
&encodingError)!
let lineChange = csvData.stringByReplacingOccurrencesOfString("\r", withString: "\n")
let csvArray:Array = lineChange.componentsSeparatedByString(“\n")
println(“\(csvArray)”)
```

修正後

```
let csvBundle = NSBundle.mainBundle().pathForResource("start", ofType: "csv")
do {
  let csvData = try String(contentsOfFile: csvBundle!,
      encoding: NSUTF8StringEncoding)
  let lineChange = csvData.stringByReplacingOccurrencesOfString("\r", withString: "\n")
  let csvArray:Array = lineChange.componentsSeparatedByString("\n")
   print(csvArray)
}
 catch {
    print("エラー")
 }
```

▪207ページ

CSVファイルなどの外部データを読み込む場合、引数にエラーを設定して初期化することができなくなりました。do - try - catch構文を使って読み込みを行います。また戻り値をString型の配列にします。

修正前

```
func loadCSV(filename :String)->[String]{
    var csvArray = []
    let csvBundle = NSBundle.mainBundle().pathForResource(filename, ofType: "csv")
    var encodingError:NSError? = nil
    let csvData = NSString(contentsOfFile: csvBundle!,encoding: NSUTF8StringEncoding, error:
&encodingError)!
    let lineChange = csvData.stringByReplacingOccurrencesOfString("\r",
          withString: "\n")
    let csvArray:Array = lineChange.
      componentsSeparatedByString("\n")
 return csvArray
}
```

修正後

```
func loadCSV(filename :String)->[String]{
    var csvArray = []
    let csvBundle = NSBundle.mainBundle().pathForResource(filename, ofType: "csv")

    do {
        let csvData = try String(contentsOfFile: csvBundle!,
            encoding: NSUTF8StringEncoding)
        let lineChange = csvData.stringByReplacingOccurrencesOfString("\r", withString: "\n")
        csvArray = lineChange.componentsSeparatedByString("\n")

    }
    catch {
        print("エラー")
    }
    return csvArray as! [String]

}
```

▪208ページ

値が変わらない変数は定数で宣言します。

修正前

```
    var viewController = ViewController()
```

修正後

```
    let viewController = ViewController()
```

▪209ページ

Swift2.0になり、toInt( )が使えなくなりました。String型をInt型に変換する場合は下記の方法で行います。

修正前

```
    if sender.tag == mondaiArray[1].toInt(){
```

修正後

```
    if sender.tag == Int(mondaiArray[1]){
```

▪223ページ

値が変わらない変数は定数で宣言します。

修正前

```
    var sortedArray = NSMutableArray(array: csvArray)
```

修正後

```
    let sortedArray = NSMutableArray(array: csvArray)
```

修正前

```
        var randomIndex = arc4random() % UInt32(arrayCount)
```

修正後

```
let randomIndex = arc4random() % UInt32(arrayCount)
```

Chapter8

▪237ページ

Alamofire ライブラリはSwift2.0に対応した最新のものをダウンロードしてください。

▪239ページ

Alamofireのデータの取得方法が変更になりました。

修正前

```
Alamofire.request(.GET,requestUrl).responseJSON {(request, response, json, error) in
    println(json)
}
```

修正後

```
Alamofire.request(.GET, requestUrl, parameters: ["foo": "bar"]).responseJSON { response in
    if let json = response.result.value {
    println(json)
    }
}
```

▪242ページ

Alamofireのデータの取得方法が変更になりました。

修正前

```
Alamofire.request(.GET,requestUrl).responseJSON {(request, response, json, error) in
    let jsonDic = json as! NSDictionary
    let responseData = jsonDic["responseData"] as! NSDictionary
    self.newsDataArray = responseData["results"] as! NSArray
    println("\(self.newsDataArray)")
}
```

修正後

```
Alamofire.request(.GET, requestUrl, parameters: ["foo": "bar"]) .responseJSON { response in
    if let json = response.result.value {
    let jsonDic = json as! NSDictionary
    let responseData = jsonDic["responseData"] as! NSDictionary
    self.newsDataArray = responseData["results"] as! NSArray
    print(self.newsDataArray)
    }
}
```

▪247ページ

値が変わらない変数は定数で宣言します。

修正前

```
var newsDic = newsDataArray[indexPath.row] as! NSDictionary
```

修正後

```
let newsDic = newsDataArray[indexPath.row] as! NSDictionary
```

▪249ページ

Swift2.0になりprintlnが使えなくなりました。デバッグエリアに情報を表示する場合はprintを使用します。

修正前

```
println(" タップされたセルのインデックスパス :\(indexPath.row)")
```

修正後

```
print(" タップされたセルのインデックスパス :\(indexPath.row)")
```

▪250ページ

値が変わらない変数は定数で宣言します。

修正前

```
var newsDic = newsDataArray[indexPath.row] as! NSDictionary
```

修正後

```
let newsDic = newsDataArray[indexPath.row] as! NSDictionary
```

▪255ページ

iOS9からApp Transport Securityという機能により、HTTP通信を行うアプリはXcodeのinfo.plistの設定が必要になりました。

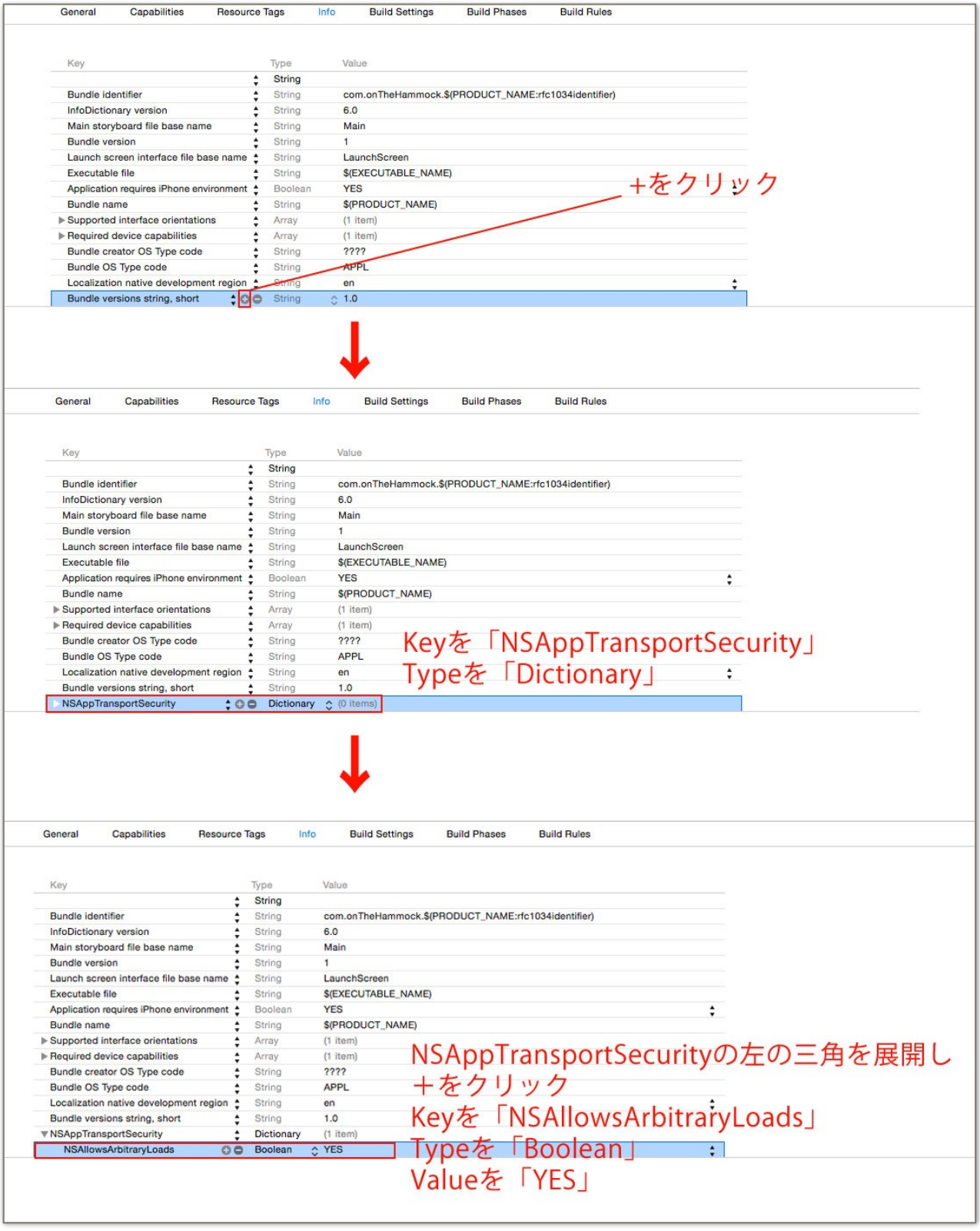
①プロジェクトアイコンをクリックして、設定画面から「Info」のタブを開く

②Custom iOS Target Propetiesの一番下の項目の「＋」をクリック

③新しい項目欄ができるので、Keyを「NSAppTransportSecurity」と入力し、Typeを「Dictionary」に設定する

④新しく作った「NSAppTransportSecurity」の項目の左の三角を展開し、「＋」をクリック

⑤新しい項目欄ができるので、Keyを「NSAllowsArbitraryLoads」と入力し、Typeを「Boolean」に、Valueを「YES」に設定する



■255ページ

値が変わらない変数は定数で宣言します。

修正前

```
var url = NSURL(string :newsUrl)!
var urlRequest = NSURLRequest(URL: url)
```

修正後

```
let url = NSURL(string :newsUrl)!
let urlRequest = NSURLRequest(URL: url)
```

▪258ページ

値が変わらない変数は定数で宣言します。

修正前

```
var wvc = segue.destinationViewController as! WebViewController
```

修正後

```
let wvc = segue.destinationViewController as! WebViewController
```

Chapter9

▪272ページ

値が変わらない変数は定数で宣言します。

修正前

```
var pickerController = UIImagePickerController()
```

修正後

```
let pickerController = UIImagePickerController()
```

▪278ページ

値が変わらない変数は定数で宣言します。

修正前

```
var cell = collectionView.dequeueReusableCellWithReuseIdentifier("Cell", forIndexPath: indexPath) as! UICollectionViewCell
```

修正後

```
let cell = collectionView.dequeueReusableCellWithReuseIdentifier("Cell", forIndexPath: indexPath) as! UICollectionViewCell
```

▪281ページ

Swift2.0でtouchesBeganメソッドの引数の型が変更になりました。

修正前

```
override func touchesBegan(touches: NSSet, withEvent event: UIEvent) {
```

修正後

```
override func touchesBegan(touches: Set<UITouch>, withEvent event: UIEvent?){
```

▪282ページ

Swift2.0でtouchesMovedメソッドの引数の型が変更になりました。これに伴い、画面上のタッチ情報取得時にキャストが必要なくなりました。

修正前

```
    override func touchesMoved(touches: NSSet, withEvent event: UIEvent) {
        let touch = touches.anyobject() as UITouch
```

修正後

```
    override func touchesMoved(touches: Set<UITouch>, withEvent event: UIEvent?) {
        let touch = touches.first!
```

▪283ページ

値が変わらない変数は定数で宣言します。

修正前

```
        var appDelegate = UIApplication.sharedApplication().delegate as! AppDelegate
```

修正後

```
        let appDelegate = UIApplication.sharedApplication().delegate as! AppDelegate
```

▪286ページ

Swift2.0になりfindが使えなくなりました。配列のなかの変数を取り出す場合は下記の方法で行います。

修正前

```
            if let index = find(appDelegate.stampArray, lastStamp){
```

修正後

```
            if let index = appDelegate.stampArray.indexOf(lastStamp){
```

▪287ページ

Swift2.0になりUIGraphicsGetCurrentContext()の返り値がoptional型になりました。

修正前

```
        canvasView.layer.renderInContext(UIGraphicsGetCurrentContext())
```

修正後

```
        canvasView.layer.renderInContext(UIGraphicsGetCurrentContext()!)
```