SCIENCES SORBONNE UNIVERSITÉ

«iSelect»

Fabrice.Kordon@lip6.fr

LIP6

About the example

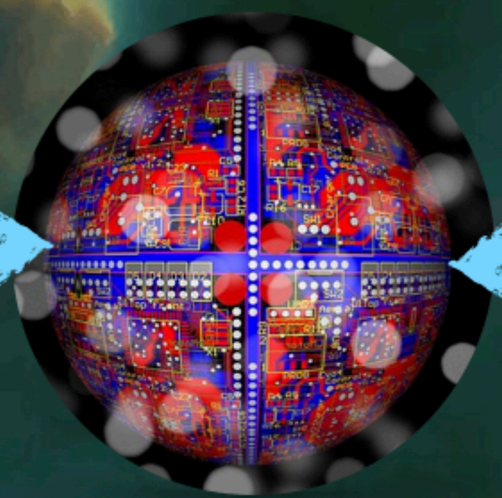
- **Build a UIPickerView**
  - Display some text according to the selected values
- **Example in full MVC**

Demo

Architecture

MyModel

- getColor
- getValue
- handling UIPickerViewDataSource



ViewController

- handling UIPickerViewDelegate

MyView

- displayInSize
- updateLabel

MyModel


```
import UIKit

class MyModel: NSObject, UIPickerViewDataSource {

    private let objects = ["", "Plane", "Computer", "Car",
                           "Boat", "Cube", "iPhone", "iPad",
                           "Paper"]
    private let colors = ["", "red", "green", "yellow", "blue",
                          "black", "purple"]

    func getColor (comp: Int) -> UIColor {
        switch comp {
        case 1 : return .red
        case 2 : return .green
        case 3 : return .yellow
        case 4 : return .blue
        case 5 : return .black
        case 6 : return .purple
        default: return .clear
        }
}
```

MyModel


```
func getValue (comp: Int, row: Int) -> String {
    if comp == 0 && row < objects.count {
        return objects[row]
    } else if comp == 1 && row < colors.count{
        return colors[row]
    } else {
        return "- ERROR -"
    }
}

// UIPickerViewDataSource protocol
    func numberOfComponents(in pickerView: UIPickerView) -> Int {
        return 2
    }

    func pickerView(_ pickerView: UIPickerView,
                    numberOfRowsInComponent component: Int) -> Int {
        if component == 0 {
            return objects.count
        } else if component == 1 {
            return colors.count
        } else {
            return 0
        }
    }
}
```

MyViewController


```
import UIKit

class ViewController: UIViewController, UIPickerViewDelegate {

    private let model = MyModel()
    private var v : MyView?

    override var preferredStatusBarStyle : UIStatusBarStyle {
        return .lightContent
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view
        v = MyView(frame: UIScreen.main.bounds, mm:model, vc:self)
        self.view = v

    }

    override func viewWillTransition(to size: CGSize,
                                     with coordinator: UIViewControllerTransitionCoordinator) {
        super.viewWillTransition(to: size, with: coordinator)
        let v = self.view as! MyView // avoids an error
        v.displayInSize(size:size) // Implemented by MyRoTaTingView
    }
```

MyViewController


```
// UIPickerViewDelegate protocol

func pickerView(_ pickerView: UIPickerView,
                titleForRow row: Int,
                forComponent component: Int) -> String? {
    return model.getValue(comp: component, row: row)
}

func pickerView(_ pickerView: UIPickerView,
                didSelectRow row: Int,
                inComponent component: Int) {
        var val = model.getValue(comp: 0,
                                 row: pickerView.selectedRow(inComponent: 0))
        val = val+" "
        val = val+model.getValue(comp: 1,
                                 row: pickerView.selectedRow(inComponent: 1))
        v?.updateLabel(s: val,
                       col:model.getColor(comp:
                                  pickerView.selectedRow(inComponent: 1)))
    }
}
```

MyView


```
import UIKit

class MyView: UIView {

    private let myModel : MyModel?

    private let back = UIImageView(image: UIImage(named: "bacgimg"))
    private let myPicker = UIPickerView()
    private let pickerSel = UILabel()

    init (frame: CGRect, mm: MyModel, vc: ViewController) {
        pickerSel.textAlignment = .center
        pickerSel.font = UIFont(name: "Helvetica-Bold", size: 18)
        pickerSel.backgroundColor = UIColor(red: 1, green: 1,
                                            blue: 1, alpha: 0.6)
        myPicker.backgroundColor = UIColor(red: 1, green: 1,
                                           blue: 1, alpha: 0.6)
        myPicker.dataSource = mm
        myPicker.delegate = vc
        myModel = mm
        myPicker.selectRow(0, inComponent: 0, animated: true)
        myPicker.selectRow(0, inComponent: 1, animated: true)
        super.init(frame: frame)
        self.addSubview(back)
        self.addSubview(pickerSel)
        self.addSubview(myPicker)
        self.displayInSize(size: frame.size)
}
```

MyView


```
required init?(coder aDecoder: NSCoder) {
    fatalError("init(coder:) has not been implemented")
}

func updateLabel (s: String, col: UIColor) {
    pickerSel.text = s
    pickerSel.textColor = col
}

func displayInSize(size: CGSize) {
    var top = 20;
    if UIDevice.current.userInterfaceIdiom == .phone &&
        size.height >= 812 {
        top = 30
    } else if UIDevice.current.userInterfaceIdiom == .phone &&
        size.width > size.height {
        top = 0
    }
    // To have he image centered
    back.center = CGPoint(x:size.width / 2, y:size.height / 2)
    pickerSel.frame = CGRect(x: 40, y:top + 40,
                             width: Int(size.width) - 80, height: 40)
    myPicker.frame = CGRect(x: Int(size.width / 2) - 140,
                            y: top + 150, width: 280, height: 150)
  }
}
```

As a conclusion…

- **Easy, is'n't it?**

- **You have no excuse to avoid this object**
  - By the way, there exist some specialized pickers
    - **To be developed mater**