SCIENCES SORBONNE UNIVERSITÉ

«Catalog»

Fabrice.Kordon@lip6.fr

LIP6

Goal of the example

Display a product list

- Fetch from a web server
  - **Updated independently from the source code**
- Encoded as an XML file
  - **Load it**
  - **parse it**
- Identify product categories
  - **Mapped to UITableView sections**

You find such examples in the AppStore

- First, the AppStore itself…
- Then, all market-based Apps
  - **Amazon, Ikea, iTunes, etc…**

Demo

BNF of the XML file

List ::= <list>*Category*+</list>

Category ::= <category val= "String">*Product*+</category>

Product ::= <product>*Price Code Name*</product>

Price ::= <price>*String*</price>

Code ::= <code>*String*</code>

Name ::= <name>*String*</name>

String ::= *any string*

BNF of the XML file

List ::= <

Category ::= <

Product ::= <

Price ::= <

Code ::=

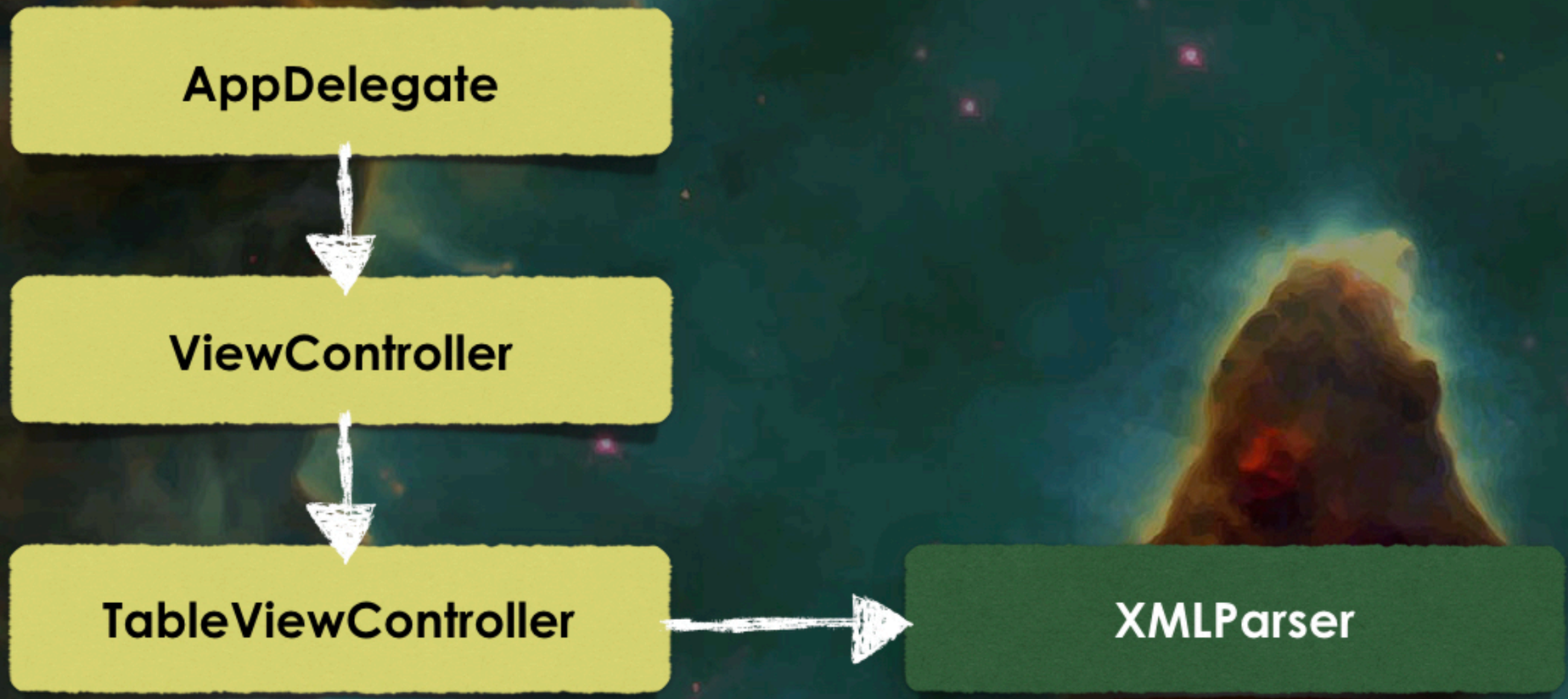
Name ::=

String ::=

…ory>

```
<list>
   <category val="Automobiles">
      <product>
         <price>30 0000€</price>
         <code>1245s854MX5</code>
         <name>Mazda MX5</name>
      </product>
      <product>
         <price>10 000€</price>
         <code>12G5s854LN2</code>
         <name>Lancia Gamma coupe</name>
      </product>
      <product>
         <price>18 000€</price>
         <code>12G5s854AF2</code>
         <name>Alfa Romeo GTV6</name>
      </product>
      <product>
         <price>50 000€</price>
         <code>12GKs854DL5</code>
         <name>DeLorean DMC-12</name>
      </product>
      <product>
         <price>50 000€</price>
         <code>12GKs854FV3</code>
         <name>Facel-Vega Facellia</name>
      </product>
   </category>
 …
</list>
```

Architecture of the application



ViewController


```
import UIKit

class ViewController: UIViewController {

    private let status = UILabel()
    private let allProducts = TableViewController(style: .grouped)
    private let viewTitle = UILabel()

    private let s = UIScreen.main.bounds.size
```

ViewController


```
override func viewDidLoad() {
    super.viewDidLoad()

    let v = UIView()
    v.backgroundColor = UIColor(red: 1.0, green: 0.8, blue: 0.6, alpha: 1.0)
    self.view = v

    status.textAlignment = .center
    status.text = "---"
    status.font = UIFont.boldSystemFont(ofSize: 20.0)
    v.addSubview(status)

    viewTitle.backgroundColor = UIColor.black
    viewTitle.textAlignment = .center
    viewTitle.textColor = UIColor(red: 1.0, green: 0.8, blue: 0.6, alpha: 1.0)
    viewTitle.text = "Catalog"
    viewTitle.font = UIFont.boldSystemFont(ofSize: 20.0)
    v.addSubview(viewTitle)

    allProducts.statusLabel = status
    allProducts.view.frame = CGRect(x: 0.0, // orientation management built-in
                                    y: 110.0,
                                    width: s.width,
                                    height: s.height - 110.0)
    v.addSubview(allProducts.view)
    self.drawInSize(s)
}
```

ViewController


```
    override func viewWillTransition(to size: CGSize,
                      with coordinator: UIViewControllerTransitionCoordinator) {
        self.drawInSize(size)
    }

    func drawInSize (_ size : CGSize) {
        viewTitle.frame = CGRect(x: 0, y: 80, width: size.width, height: 30)
        status.frame = CGRect(x: 20, y: 30, width: size.width - 40, height: 30)
    }
}
```

TableViewController


```
import UIKit

class TableViewController: UITableViewController, XMLParserDelegate {

    var statusLabel : UILabel?

    private var catName = [String]()
    private var name = [[String]]()
    private var price = [[String]]()
    private var code = [[String]]()
    private var buffer = Data()
    private var strBuffer = ""

    // For XML analysis
    private var newCategory = false
    private var crtCategory = 0
    private var crtCode = ""
    private var crtPrice = ""
    private var crtName = ""
```

TableViewController


```
override func viewDidLoad() {
    super.viewDidLoad()
    let url = URL(string: "https://lip6.fr/Fabrice.Kordon/catalog.xml")
    let session = URLSession(configuration: URLSessionConfiguration.default,
                             delegate: nil,
                             delegateQueue: nil)
    let tache = session.dataTask(with: url!) {
        (d: Data?, r: URLResponse?, e: Error?) -> Void in
        DispatchQueue.main.async { // main queue (thread) required
            if e != nil {
                let a = UIAlertController(title: "Network problem",
                                          message: e?.localizedDescription,
                                          preferredStyle: .alert)
                a.addAction(UIAlertAction(title: "OK", style:.default,
                                          handler: nil))
                self.present(a, animated: true, completion: nil)
            } else {
                let parser = XMLParser(data: d!)
                parser.delegate = self
                self.statusLabel?.text = "Data analysis..."
                self.statusLabel?.textColor = UIColor.blue
                if parser.parse() == false {
                    self.statusLabel?.text = "Analysis failed !!!"
                }
            }
        }
    }
    tache.resume()
    statusLabel?.textColor = UIColor.red
    statusLabel?.text = "Loading data..."
}
```

TableViewController


```
override func viewDidAppear(_ animated: Bool) {
    // TableView is smaller than the current window
    self.view.frame = CGRect(x: 0.0, y: 110.0,
                             width: UIScreen.main.bounds.size.width,
                             height: UIScreen.main.bounds.size.height - 110)
}

// MARK: - TableView DataSource

override func numberOfSections(in tableView: UITableView) -> Int {
    return name.count
}

override func tableView(_ tableView: UITableView,
                        numberOfRowsInSection section: Int) -> Int {
    let table = name[section] as [String]
    return table.count
}
```

TableViewController


```
override func tableView(_ tableView: UITableView,
                        cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    var cell = tableView.dequeueReusableCell(withIdentifier: "ident")

    if cell === nil {
        cell = UITableViewCell(style: .subtitle, reuseIdentifier: "ident")
    }

    let tname = name[indexPath.section] as [String]
    let tprice = price[indexPath.section] as [String]
    let tCode = code[indexPath.section] as [String]
    cell?.selectionStyle = .none
    cell?.textLabel?.text = tname[indexPath.row]
    cell?.detailTextLabel?.text = "\(tCode[indexPath.row]) (\(tprice[indexPath.row]))

    cell?.backgroundColor = UIColor.white
    cell?.textLabel?.textColor = UIColor.black
    cell?.detailTextLabel?.textColor = UIColor.red

    return cell!
}

override func tableView(_ tableView: UITableView,
                        titleForHeaderInSection section: Int) -> String? {
    return catName[section]
}
```

TableViewController


```
//  MARK: NSXMLParserDelegate protocol

func parserDidStartDocument(_ parser: XMLParser) {
    strBuffer = "" // Initialiser le buffer
}

func parserDidEndDocument(_ parser: XMLParser) {
    DispatchQueue.main.async {// refresh immediatly
        self.tableView.reloadData()
        self.statusLabel?.textColor = UIColor.black
        self.statusLabel?.text = "Product list loaded"
    }
}

func parser(_ parser: XMLParser,
            validationErrorOccurred validationError: Error) {
    let a = UIAlertController(title: "XML problem",
                              message: validationError.localizedDescription,
                              preferredStyle: .alert)
    a.addAction(UIAlertAction(title: "OK", style:.default, handler: nil))
    self.present(a, animated: true, completion: nil)
}
```

TableViewController


```
func parser(_ parser: XMLParser,
            didStartElement elementName: String,
            namespaceURI: String?,
            qualifiedName qName: String?,
            attributes attributeDict: [String : String] = [:]) {
    if elementName == "category" {
        newCategory = true
        catName.append(attributeDict["val"] ?? "no value")
    }
    strBuffer = "" // flush buffer
}
```

```
func parser(_ parser: XMLParser, foundCharacters string: String) {
    if newCategory {
        newCategory = false
        name += [[String()]]
        code += [[String()]]
        price += [[String()]]
        // To create the new dimension, we added artificially a dummy entry
        name[crtCategory].remove(at: 0)
        code[crtCategory].remove(at: 0)
        price[crtCategory].remove(at: 0)
    } else {
        strBuffer += string // Let's accumulate in the buffer
    }
}
```

TableViewController


```
func parser(_ parser: XMLParser,
            didEndElement elementName: String,
            namespaceURI: String?,
            qualifiedName qName: String?) {
    if elementName == "category" {
        crtCategory += 1
    }
    if elementName == "code" {
        crtCode = strBuffer
    }
    if elementName == "price" {
        crtPrice = strBuffer
    }
    if elementName == "name" {
        crtName = strBuffer
    }
    if elementName == "product" {
        name[crtCategory] += [crtName]
        code[crtCategory] += [crtCode]
        price[crtCategory] += [crtPrice]
    }
  }
}
```

As a conclusion…

That's all…

- You are now ready to interact with web servers