20CYS312 - Principles of Programming Languages
Exploring Programming Paradigms

Assignment-01

Presented by Madhav Harikumar
CB.EN.U4CYS21038
TIFAC-CORE in Cyber Security
Amrita Vishwa Vidyapeetham, Coimbatore Campus

Feb 2024

Outline

1. Paradigm 1 - Dataflow
2. LabVIEW
3. Paradigm 2 - Reactive
4. RxSwift
5. Comparison and Discussions
6. Bibliography

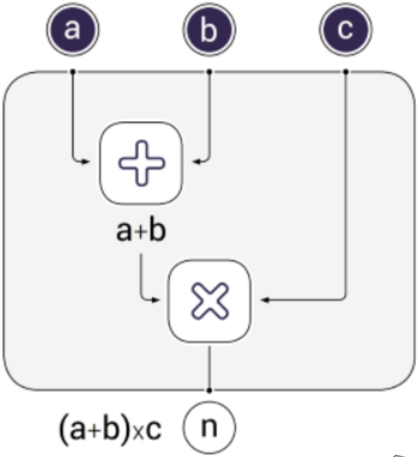
Programming Paradigms

Figure: LabVIEW (DataFlow Paradigm)

Figure: RxSwift (Reactive Paradigm)

DataFlow

- A programming paradigm known as dataflow programming (DFP) views program execution as data passing through a number of operations or transformations. In a graph, every operation might be represented as a node. Data is transferred between nodes via directed arcs. As soon as the input data are accessible, a node operates. On the output arcs, the result is sent.



How is DataFlow different from others?

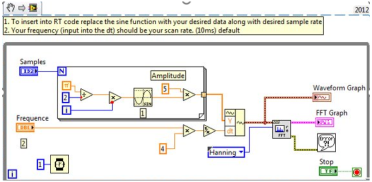
- Dataflow centers around flow of data whereas other paradigms focus on sequence of steps or even the content and the evaluation of the function.
- DataFlow supports parallelism such as executing operations as soon as input data is available. Whereas paradigms like imperative for example, relies on threads and explicit synchronization for concurrency.
- DataFlow emphasizes a data-driven programming style. Whereas paradigms like functional and imperative focuses on steps and also functions expressions.

What is LabVIEW?

- LabVIEW is a software development environment created by National Instruments.
- LabVIEW is a graphical programming environment engineers use to develop automated research, validation, and production test systems.
- LabVIEW is short for Laboratory Virtual Instrument Engineering Workbench.

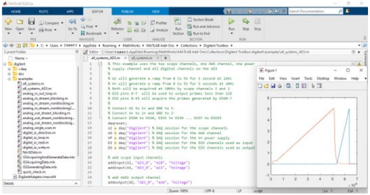
LabVIEW and MATLAB

- When it comes to computing in a programming environment, we use MATLAB.
- When it comes to instrumental control, data acquisition and automated testing, we use LabVIEW.



1. To insert into RT code replace the sine function with your desired data along with desired sample rate
2. Your frequency (input into the dt) should be your scan rate. (10ms) default

Figure: LabVIEW



```
% This example uses the two scope channels, one AWG channel, one power
% supply channel and all digital channels on the AD3
%
% W1 will generate a ramp from 0 to 5V for 1 second at 1KHz
% V+ will generate a ramp from 0 to 5V for 5 seconds at 10Hz
% Both will be acquired at 10KHz by scope channels 1 and 2
% DIO pins 0-7  will be used to output primes less than 128
% DIO pins 8-15 will acquire the primes generated by DIO0-7
%
% Connect W1 to 1+ and GND to 1-
% Connect V+ to 2+ and GND to 2-
% Connect DIO0 to DIO8, DIO1 to DIO9 ... DIO7 to DIO15
daqreset;
AI = daq("digilent") % DAQ session for the scope channels
AO = daq("digilent") % DAQ session for the AWG channel
VP = daq("digilent") % DAQ session for the V+ power supply
DI = daq("digilent") % DAQ session for the DIO channels used as input
DO = daq("digilent") % DAQ session for the DIO channels used as output

% add scope input channels
addinput(AI, "AD3_0", "ai0", "Voltage")
addinput(AI, "AD3_0", "ai1", "Voltage")

% add AWG1 output channel
addoutput(AO, "AD3_0", "ao0", "Voltage")
```

Figure: MATLAB

Functions performed by LabVIEW

- Mathematical and basic logic functions (Logic Gates).
- Basic arithmetic, if/then/else-if conditional statements.
- Fast Fourier Transform (FFTs).
- Filtering functions like Lowpass, Highpass and Bandpass.
- Proportional-Integral-Derivative (PID) control loops.

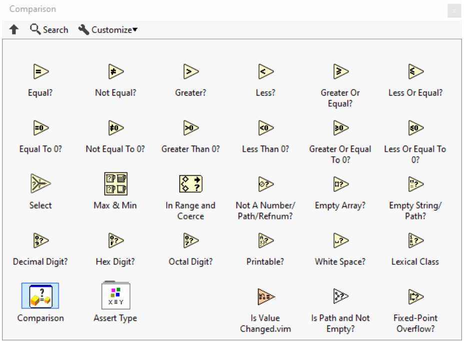


Figure: Your Image Caption

Code Layout(LabVIEW)

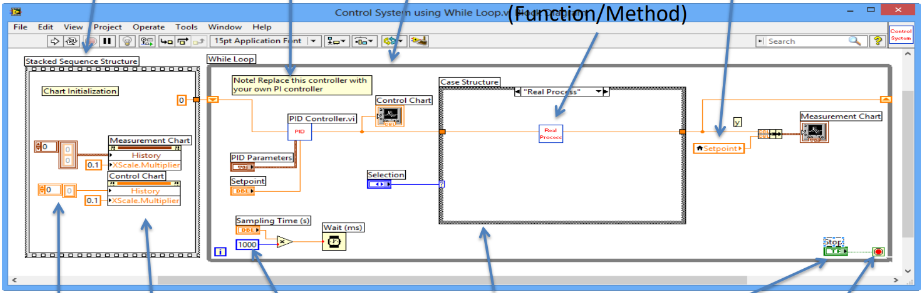
Sequence Structure

Comment

While Loop

Sub VI (Function/Method)

Local Variable



Note! Replace this controller with your own PI controller

Arrays

Property Nodes

Constants

Case Structure (if-else)

Stop Button

Condition (When shall the loop end?)

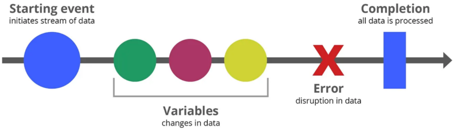
Note! To do something with an object – Right-click on it

Figure: LabVIEW Implemented Diagram

Reactive

- Reactive programming is a programming paradigm that deals with asynchronous data streams and the propagation of changes. It is characterized by the use of observable sequences and the application of functional-style operators to work with these sequences.

Reactive programming uses asynchronous data streams



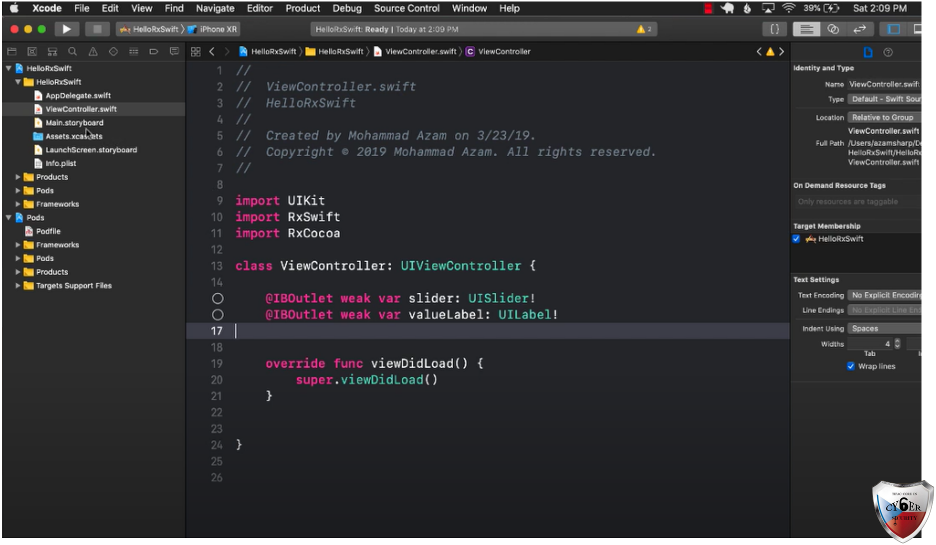
How is Reactive different from the others?

- Reactive focuses on the propagation of changes in the system as streams of data. Whereas other paradigms focuses on data encapsulation, or functional sequences etc.
- Property of asynchronous: Well-suited for handling asynchronous and event-driven code by providing a declarative way to react to changes. Whereas other paradigms explicitly define control flow or even concurrency or reusability of code etc.

What is RxSwift?

- It is a popular programming language used for iOS
- RxSwift is a tool that helps organize and manage code that deals with events and tasks happening at different times.
- It uses observable sequences and functional-style operations, allowing you to control when and how tasks are carried out using schedulers
- It uses reactive programming paradigm. This means that the underlying execution model will automatically propagate changes across the data flow, and that static or dynamic data flows should be easily expressed in the programming languages employed.

Code with Layout (Example) Part-1



```swift
//
//  ViewController.swift
//  HelloRxSwift
//
//  Created by Mohammad Azam on 3/23/19.
//  Copyright © 2019 Mohammad Azam. All rights reserved.
//

import UIKit
import RxSwift
import RxCocoa

class ViewController: UIViewController {

    @IBOutlet weak var slider: UISlider!
    @IBOutlet weak var valueLabel: UILabel!


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



}


```

Identity and Type

Name: ViewController.swift

Type: Default - Swift Sour

Location: Relative to Group

ViewController.swift

Full Path: /Users/azamsharp/D
HelloRxSwift/HelloRx
ViewController.swift

On Demand Resource Tags

Only resources are taggable

Target Membership

☑ HelloRxSwift

Text Settings

Text Encoding: No Explicit Encoding

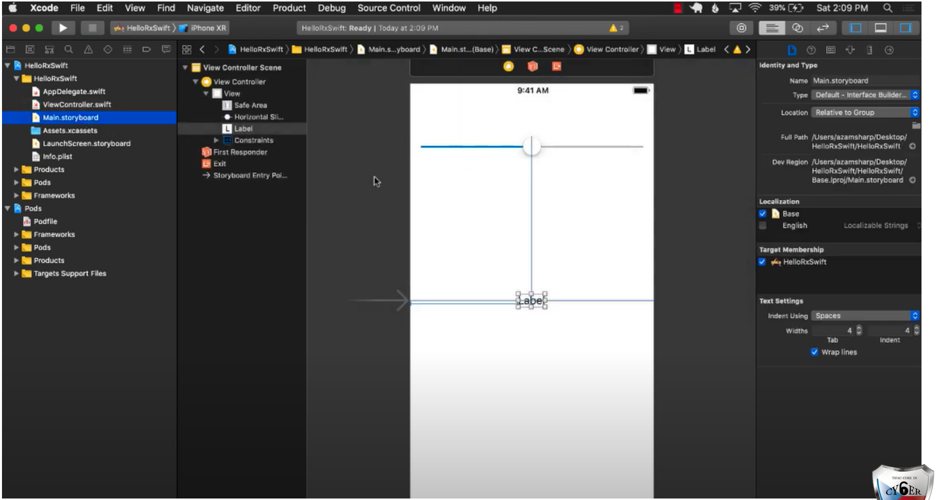
Line Endings: No Explicit Line End

Indent Using: Spaces

Widths: 4 Tab

☑ Wrap lines

Code with Layout (Example) Part-2



Identity and Type

Name Main.storyboard

Type Default - Interface Builder...

Location Relative to Group

Full Path /Users/azamsharp/Desktop/HelloRxSwift/HelloRxSwift

Dev Region /Users/azamsharp/Desktop/HelloRxSwift/HelloRxSwift/Base.lproj/Main.storyboard

Localization

☑ Base

☐ English Localizable Strings

Target Membership

☑ HelloRxSwift

Text Settings

Indent Using Spaces

Widths 4 Tab 4 Indent

☑ Wrap lines

Dataflow VS Reactive

- **Model of Computation:** Reactive programming is often event-driven, reacting to changes in data streams and asynchronous events. Dataflow programming is data-driven, with the execution of components triggered by the availability of input data.
- **Programming Abstraction:** Reactive programming uses observables and operators to manipulate and transform data streams. Dataflow programming abstracts the program as a network of interconnected nodes, with data flowing along the edges.
- **Use Cases:** Reactive programming is well-suited for scenarios where reacting to asynchronous events, such as user input or network responses, is a primary concern. Dataflow programming is often used in scenarios where data processing can be naturally represented as a flow through a network, such as in signal processing, image processing, or workflows

References

- https://devopedia.org/dataflow-programming
- https://bitspark.de/blog/what-the-hell-is-flow-based-programming
- https://www.viewpointusa.com/labview/what-is-labview-used-for/
- https://www.ni.com/en/shop/labview.html
- https://labviewwiki.org/wiki/Functions$_{P}$*alette/Programming/Comparison*
- https://medium.com/@luishrsoares/reactive-programming-principles-explained-437e23eb65df
- https://www.kodeco.com/books/rxswift-reactive-programming-with-swift/v4.0/chapters/1-hello-rxswift
- https://www.techtarget.com/searchapparchitecture/definition/reactive-programming
- https://www.youtube.com/watch?v=k5egUAZiwMwt=273s