//

// ViewController.swift

// Echo

//

// Created by on 2/25/20.

// Copyright © 2020 echotest. All rights reserved.

//

**import** UIKit

**class** ViewController: UIViewController {

**@IBOutlet** **weak** **var** ExpectedOutput: UITextView!

**@IBOutlet** **weak** **var** receivedMsg: UITextView!

**@IBOutlet** **weak** **var** emailPasswordTest: UITextView!

**@IBOutlet** **weak** **var** Attributes: UITextView!

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

// Do any additional setup after loading the view.

}

// Function to test if the email is valid, not if it exists

**func** isValidEmail(testStr:String) -> Bool {

// Minimum 2 characters, longest email length is 64 characters

**let** emailRegEx = "[A-Z0-9a-z.\_%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,64}"

**let** emailTest = NSPredicate(format:"SELF MATCHES %@", emailRegEx)

**return** emailTest.evaluate(with: testStr)

}

// Function to test if the password is valid

**func** isValidPassword(testStr: String) -> Bool {

// Minimum 8 characters, no max at least 1 Alphabet and 1 number

**let** PassRegEx = "^(?=.\*[A-Za-z])(?=.\*\\d)[A-Za-z\\d]{8,}$"

**let** passwordTest = NSPredicate(format: "SELF MATCHES %@", PassRegEx)

**return** passwordTest.evaluate(with: testStr)

}

//MARK: **Echo**

// The echo server

**@IBAction** **func** Echo(**\_** sender: **Any**) {

**var** messageNum = 0

**let** ws = WebSocket("wss://capstone.mcs.uvawise.edu:6969/echo")

**let** send : ()->() = {

messageNum += 1

**let** msg = "\(messageNum): 123"

print("send: \(msg)")

ws.send(msg)

}

ws.event.open = {

print("opened")

send()

}

ws.event.close = { code, reason, clean **in**

print("close")

}

ws.event.error = { error **in**

print("error \(error)")

}

ws.event.message = { message **in**

**if** **let** text = message **as**? String {

print("recv: \(text)")

**if** messageNum == 10 {

ws.close()

} **else** {

send()

}

}

}

}

//MARK: **001**

// email~password

// 001 is the code for creating an account

// email - the email for the user

// password - the password for the user

// Example : 001mmb9ud@uvawise.edu~12345

**@IBAction** **func** createAccount(**\_** sender: **Any**) {

// Declarations for the attributes needed for the 001 code

**let** ws = WebSocket("wss://capstone.mcs.uvawise.edu:6969/echo")

**let** email = "whizkid8198@yahoo.com"

**let** password = "dPassword"

**let** code = "001"

**let** symbol = "~"

**let** sentence = code + email + symbol + password

// Testing if email is valid, not if it exists

**let** emailTest = isValidEmail(testStr: email)

**if** emailTest == **false**{

emailPasswordTest.text = "Invalid email, Please try again "

}

// Testing if the password is valid

**let** passwordTest = isValidPassword(testStr: password)

**if** passwordTest == **false**{

emailPasswordTest.text = " Invalid password, Please try again "

}

// Testing if the sentence is a string

**if** sentence **is** String{

ExpectedOutput.text = sentence + " --- The sentence is a string"

}

// Outputting the attributes for the sentence

Attributes.text = "The Email: \(email)\nThe Password: \(password)\nThe Code: \(code)"

// Sending the sentence to the server

ws.send(sentence)

// Reading the message from the server

ws.event.message = { message **in**

**if** **let** text = message **as**? String {

**self**.receivedMsg.text = "recv: \(text)"

ws.close()

}

}

}

//MARK: **002**

// pin~email

// 002 is the code for verifying an account

// pin - the verification pin sent to the email

// email - the email for the user

// Example: 00212345~email@email.com

**@IBAction** **func** Verifying(**\_** sender: **Any**) {

// Declarations for the attributes needed for the 002 code

**let** email = "whizkid8198@yahoo.com"

**let** ws = WebSocket("wss://capstone.mcs.uvawise.edu:6969/echo")

**let** symbol = "~"

**let** code = "002"

**let** testPin = "4520899"

**let** sentence = code + testPin + symbol + email

// Testing if email is valid, not if it exists

**let** emailTest = isValidEmail(testStr: email)

**if** emailTest == **false**{

emailPasswordTest.text = "Invalid email, Please try again "

}

// Testing if the sentence is a string

**if** sentence **is** String{

ExpectedOutput.text = sentence + " --- The sentence is a string"

}

// Outputting the attributes for the sentence

Attributes.text = "The Email: \(email)\nThe Code: \(code)\nThe Test Pin: \(testPin)"

// Sending the sentence to the server

ws.send(sentence)

// Reading the message from the server

ws.event.message = { message **in**

**if** **let** text = message **as**? String {

**self**.receivedMsg.text = "recv: \(text)"

ws.close()

}

}

}

//MARK: **003**

// email~password

// 003 is the code for logging in

// email - the email for the user

// password - the password for the user

**@IBAction** **func** login(**\_** sender: **Any**) {

// Declarations for the attributes needed for the 003 code

**let** ws = WebSocket("wss://capstone.mcs.uvawise.edu:6969/echo")

**let** email = "whizkid8198@yahoo.com"

**let** password = "dPassword"

**let** code = "003"

**let** symbol = "~"

**let** sentence = code + email + symbol + password

// Testing if email is valid, not if it exists

**let** emailTest = isValidEmail(testStr: email)

**if** emailTest == **false**{

emailPasswordTest.text = "Invalid email, Please try again "

}

// Testing if the password is valid

**let** passwordTest = isValidPassword(testStr: password)

**if** passwordTest == **false**{

emailPasswordTest.text = " Invalid password, Please try again "

}

// Testing if the sentence is a string

**if** sentence **is** String{

ExpectedOutput.text = sentence + " --- The sentence is a string"

}

// Outputing the attributes for the sentence

Attributes.text = "The Email: \(email)\nThe Password: \(password)\nThe Code: \(code)"

// Sending the sentence to the server

ws.send(sentence)

// Reading the messag from the server

ws.event.message = { message **in**

**if** **let** text = message **as**? String {

**self**.receivedMsg.text = "recv: \(text)"

ws.close()

}

}

}

//MARK: **004**

// 004songID~branchSimilarity~onlyLong~minimumDistance~email~password

// 004 is the code for email, password, songID, branchSimiliarity, onlyLong, minDistance, and code

**@IBAction** **func** SongInfo(**\_** sender: **Any**) {

//Declarations for attributes needed for the 004 code

**let** ws = WebSocket("wss://capstone.mcs.uvawise.edu:6969/echo")

**let** email = "whizkid8198@yahoo.com"

**let** password = "dPassword"

**let** songID = "3n3Ppam7vgaVa1iaRUc9Lp"

**let** branchSimiliarity = "50"

**let** onlyLong = "1"

**let** minDistance = "25"

**let** symbol = "~"

**let** code = "004"

**let** sentence = code + songID + symbol + branchSimiliarity + symbol + onlyLong + symbol + minDistance + symbol + email + symbol + password

// Testing if email is valid, not if it exists

**let** emailTest = isValidEmail(testStr: email)

**if** emailTest == **false**{

emailPasswordTest.text = "Invalid email, Please try again "

}

// Testing if the password is valid

**let** passwordTest = isValidPassword(testStr: password)

**if** passwordTest == **false**{

emailPasswordTest.text = " Invalid password, Please try again "

}

// Testing if the sentence is a string

**if** sentence **is** String{

ExpectedOutput.text = sentence + " --- The sentence is a string"

}

// Outputing the attributes for the sentence

Attributes.text = "The Email: \(email)\nThe Password: \(password)\nThe Code: \(code)\nThe SongID: \(songID)\nThe branch similiarity: \(branchSimiliarity)\nThe Only long: \(onlyLong)\nThe Min Distance \(minDistance)"

// Sending the sentence to the server

ws.send(sentence)

// Reading the message from the server

ws.event.message = { message **in**

**if** **let** text = message **as**? String {

**self**.receivedMsg.text = "recv: \(text)"

ws.close()

}

}

}

}