Interview Zen

Harsha Dhammika Liyanachchi

Total duration: 239:59

Question 1

Question 1 of 17

What do you like most about developing mobile Apps? What things don't you like about developing mobile apps?

```
Like:
I like to keep working on something update frequently.
It update the devices and technologies and ability to work different d
That main reason I like to work on mobile application.

Dont Like:
This is not specific to mobile application development.
I don't like doing documentation and comments but I'm doing that beacu
```

0:00 / 7:15

1x 2x 5x

Question 2

Question 2 of 17

Complete the implementation of the following Swift function.

```
func doStuff() {
  let array: [String] = ["one", "two", "three", "four", "five", "six",
"seven", "eight", "nine", "ten"]

  // Print out array in alphabetical order
  <your-code-goes-here>

 // Print out the array in reverse numerical order
 <your-code-goes-here>

 // Print out all the array elements that start with the letter 't'
 <your-code-goes-here>

 // Print out a dictionary of elements grouped by the first letter
```

```
  <your-code-goes-here>
}
```

```
func doStuff() {
  let array: [String] = ["one", "two", "three", "four", "five", "six",

  // Print out array in alphabetical order
  print(array.sort())

 // Print out the array in reverse numerical order
 var numercalArray:Int = Array<Int>()
 let formatter = NumberFormatter()
 formatter.style = .spellOut
 for strnum in array{
    if let number = numberFormatter.number(from: strnum){
        numercalArray.append(number)
    }
 }
 print(numercalArray.sort)

 // Print out all the array elements that start with the letter 't'
  let elementsStartWithT = array.filter{$0.hasPrefix("t")}
  print(elemetsStartWithT)

 // Print out a dictionary of elements grouped by the first letter
 let groupedDict = Dictinary(grouping: array) {$0.first!}
 print(groupedDict)
}
```

0:00 / 21:20

1x 2x 5x

Question 3

Question 3 of 17

Complete the implementation of the following Swift function.

```
func doStuff() {
  let dict = [1:"one", 2:"two", 3:"three", 4:"four", 5:"five", 6:"six",
7:"seven", 8:"eight", 9:"nine", 10:"ten", 11:"eleven", 12:"twelve",
13:"thirteen"]

  // Print out the keys of all the odd numbered entries in descending
order
  <your-code-goes-here>

  // Print out the values of all even numbered entries that start with
the letter 't'
  <your-code-goes-here>

  // Print out all the values that are multiples of 3
  <your-code-goes-here>
}
```

```
func doStuff() {
  let dict = [1:"one", 2:"two", 3:"three", 4:"four", 5:"five", 6:"six"

  // Print out the keys of all the odd numbered entries in descending
  let keysArray = dict.keys
  let oddNumbers = keysArray.filter{ $0 % 2 != 0 }
  print(oddNumbers.sort{$0 > $1})

  // Print out the values of all even numbered entries that start with
  let valuesArray = dict.values
  let startWithT = valueArray.filter{$0.lowercased.hasPrefix(t)}
  let evenNumbersWithT = startWithT.filter{$0 % 2 == 0}
  print(evenNumbersWithT)

  // Print out all the values that are multiples of 3
  let multipleOf3 = keyArray.filter{$0 % 3 == 0}
  for key in multpleOf3{
        print(dict[key])
    }
}
```

0:00 / 11:36

1x 2x 5x

Question 4

Question 4 of 17

Given the following enumeration, write a Swift function to convert an integer to Roman numerals, e.g. 2018 = MMXVIII, 2019 = MMXIX, 2020 = MMXX.

```
enum RomanAlphabet: Int, CaseIterable {
  case I=1, V=5, X=10, L=50, C=100, D=500, M=1000, W=5000, Y=10000
}
```

Using your function what is 12345 in Roman numerals?

```
func intToToman(number:Int) -> String{

    let cases = RomanAlphabet.allCases
    var result:String = ""

    for case in cases{
        let value:Int = number / case.rawValue

        if(value > 0 ){
                result += String(repeating:case, count:value)
                number = number % case.rawValue
                if(number == 9){
                    resut += "IX"
                    break
                }else if(number == 4){
                     resut += "IV"
                     break
                }
            }
        }
        print(result)

        }

        Answer for 12345 = YMMCCCXXXXV
```

0:00 / 42:57

1x 2x 5x

```
Question 5 of 17

Write a Swift protocol for an Animal protocol that has properties for:

- name,
- color,
- number of legs
- number of wings
- has a tail or not

This protocol must also have methods that allow the type to:
- walk a certain distance
- fly a certain distance
- swim a certain distance

Provide default implementations for the three methods in the protocol,
that throw error if the methods (walk/fly/swim) are called on an Animal
that cannot walk, fly or swim.
```

```
protocol Animal{
        var name:String{get}
        var color:String{get}
        var numberOfLegs:Int{get}
        var hasTail:Bool{get}

        func walk(distance:Double) throws
        func fly(distance:Double) throws
        func swim(distance:Double) throws
    }

enum AnimalError:Error{
    case cannotWalk
    case cannotFly
    case cannotSwim
}

class Duck:Animal{
        var name:String
        var color:String
        var numberOfLegs:Int
        var hasTail:Bool

        func walk(distance:Double) throws{
            throw AnimalError.cannotWalk
        }

        func fly(distance:Double) throws{
            throw AnimalError.cannotFly
        }

        func swim(distance:Double) throws{
            throw AnimalError.cannotSwim
        }
```

0:00 / 11:05

1x 2x 5x

Question 6 of 17

The Animal class has the following constraints:

- Animals must have an even number of legs or wings
- Animals may not have more than 8 legs or more than 6 wings
- Animals can fly if they have 2 or more wings
- Animals can walk if they have 2 or more legs
- Animals may not have a negative number of legs or wings

Write a unit test class using XCTest that validates the implementation of the above constraints.

```
func Animal:XCTestCase{
        //Test case for number of legs
        func testEvenNumberOfLegs(){
            let animal = Animal()
            animal.legs = 4
            XCTAssertTrue(animal.legs % 2 == 0)
            }

        func testEvenNumberOfWings(){
            let animal = Animal()
            animal.wings = 4
            XCTAssertTrue(animal.wings % 2 == 0)
            }
        func testNotMoreThan8Legs(){
            let animal = Animal()
            anima.legs = 6
            XCTAssetLessThanOrEqual(animal.legs, 8)
            }

        func testNotMoreThan6Wings(){
            let animal = Animal()
            anima.wings = 6
            XCTAssetLessThanOrEqual(animal.wings, wings)
            }

        func testAnimalCanFly(){
            let animal Animal()
            animal.wings = 2
            XCTAsserTrue(animal.canFly)
            }

          func testAnimalCanWalks(){
            let animal Animal()
            animal.legs = 2
            XCTAsserTrue(animal.canWalk)
            }

            func testNotNegaticeNumberOfLegs(){
                let animal = Animal()
                animal.legs = -2
                XCTAssertTrue(animal.leags == 0)
            }

             func testNotNegaticeNumberOfWings(){
                let animal = Animal()
                animal.wings = -2
                XCTAssertTrue(animal.wings == 0)
            }

    }
```

0:00 / 15:41

1x 2x 5x

Question 7 of 17

Write a Swift extension to String that hyphenates all words in the string so that the expression:

"this is a string".hyphenate()

returns the value:

"t-h-i-s i-s a s-t-r-i-n-g"

```
extention String{
    var hypernate:String{
        var hyperNatedString = ""
        var setHyphen = false
        for char in self{
            if !char.isWhiteSpace{
                hyperNatedString += "\(char)-"
                setHyphen = true
            }else{
                //Drop last hyphen
                if(setHyphen){
                    huperNatedString.dropLast()
                    //To avoid drop last if word contain multiple spac
                    setHyphen = false
                }
            }
        }
        return hyperNatedString
    }
```

0:00 / 15:06

1x 2x 5x

Question 8

Question 8 of 17

Write a function called makePrinter() that takes an array of Strings, and returns a function that will print out each of those elements when it is called.

So given the code:

```
let myFunction = makePrinter(["one", "two", "three"])
myFunction()

// will print the following output:
// one
// two
// three
```

```
func makePrinter(string:[String]) -> () -> Void{
    return{
        guard string.count > 0 else {
            return
        }
        for str in string{
            prting(str)
        }
    }
}
```

0:00 / 4:39

1x 2x 5x

Question 9 of 17

Write a function called makeRepeater(nTimes: Int), that returns a function that takes a string and prints out that string nTimes. So the code:

```
let myFunction = makeRepeater(3)
myFunction("Hello World")

// will print out:
// Hello World
// Hello World
// Hello World
```

```
func makeRepater(nTimes: Int) -> (string)->Void{
    return { theString in
        for _ in 1...nTimes{
            print(inputString)
        }
    }
}
```

0:00 / 4:57

1x 2x 5x

Question 10

Question 10 of 17

Complete the implementation below:

```
enum ValidationError: Error {
  case invalidPassword
  case passwordNotMatch
  case otherError
}

func validate(element: String, _ validator: (String) throws -> ()) {
  do {
    try validator(element)
  } catch let err {
    print(err)
  }
}
```

```
// Write a password validator function named passwordValidator that can
be pass into validate(element, _) so that calling the validate function
with password and passwordValidator will throw a ValidationError if the
password is not valid.
// A valid password must have length of between 8 and 13 characters and
the characters must be alphanumeric and case sensitive.
func passwordValidator() -> (String) throws -> Void {
  <your-code-goes-here>
}

// Test your validator
<your-code-goes-here>

// Write a password confirmation validator function named
passwordConfirmValidator that can be passed into validate(element, _)
so that calling the validate function with password and
passwordConfirmValidator will throw a ValidationError if the password
does not match the confirm password expression.
func passwordConfirmValidator(value: @escaping @autoclosure () ->
String) -> (String) throws -> Void {
  <your-code-goes-here>
}

// Test your validator
<your-code-goes-here>
```

```
func passwordValidator() -> (String) throws -> Void {
  return { password in

    let regx = "^[a-zA-Z0-9]"
    let predicate = NSPredicate(format: "SELF MATCHES %@", passwordReg

     guard (password.count >= 8 && password.count <= 13) && predicate.
       throw ValidationError.invalidPassword
    }
  }
}

validate(element: "Abc1234", passwordValidate())
validate(element: "weak", passwordValidate())

func passwordConfirmValidator(value: @escaping @autoclosure () -> Stri
    return { confirmPassword in
        guard confirmPassword == value() else{
            throe ValidationError.passwordNotMach
            }

    }
}

let password = "Abc12345"
let confirmPassword = "Abc12345"

validate(element: confirmPassword, try passwordConfirmValidator(value:
validate(element: "edcrfv123", try passwordConfirmValidator(value: pas
```

0:00 / 29:42

1x 2x 5x

```
Question 11 of 17

Given the following JSON:

{
  "memberProfile": {
    "memberSystemId": "63c2fe23-90a1-489f-a372-a249a616f1ee",
    "preferredName": "John Doe",
    "email": "john.doe@test.com",
    "emailVerified": true,
    "dateOfBirth": "1957-08-31",
    "sequenceId": 5,
    "mailingAddress": {
      "address1": "Level 11",
      "address2": "199, Jalan Tun Razak",
      "city": "Kuala Lumpur",
      "country": "Malaysia",
      "postcode": "50450"
    },
    "favColor": "#b20040"
  }
}

// Complete the implementation below:

struct MemberAddress: Codable {
  <your-code-goes-here>
}

struct MemberProfile: Codable {
  var memberSystemId: String
  var name: String
  var email: String
  var emailVerified: Bool
  var dateOfBirth: Date?
  var sequenceId: Int
  var address: MemberAddress?
  var favColor: UIColor?

  <your-code-goes-here>
}

struct MemberProfileResponse: Codable {
  var memberProfile: MemberProfile

  static function decoder() -> JSONDecoder {
    <your-code-goes-here>
  }
}
```

```
struct MemberAddress: Codable {
  var address1: String
  var address2: String
  var city: String
  var country:String
  var postcode:String
}

struct MemberProfile: Codable {
  var memberSystemId: String
  var name: String
  var email: String
  var emailVerified: Bool
  var dateOfBirth: Date?
  var sequenceId: Int
  var address: MemberAddress?
  var favColor: UIColor?

  enum CoodingKeys:String, CodingKey{
       case memberSystemId
        case name = "preferredName"
        case email
        case emailVerified
        case dateOfBirth
        case sequenceId
        case address = "mailingAddress"
        case favColor
      }
}

struct MemberProfileResponse: Codable {
  var memberProfile: MemberProfile

  static function decoder() -> JSONDecoder {
    let decoder = JSONDecoder()
    return decoder
  }
}
```

0:00 / 10:37

1x 2x 5x

Question 12

Question 12 of 17

```
struct MemberProfile: Codable {
  var memberSystemId: String
  var name: String
  var email: String
  var emailVerified: Bool
  var dateOfBirth: Date?
  var sequenceId: Int
  var address: MemberAddress?
  var favColor: UIColor?
}
```

Given an array of MemberProfile (from Question 11) and using only higher order functions like Map, Filter, Reduce, etc., return an NSAttributedString object that contains the members' names in system font with font colour of favColor, ordered by the sequenceId. For every even numbered element in the array, include the members' email address in parenthesis after the members' name.

```
func attributedMemberNames(memberProfiles:[MemberProfile]) -> NSAttrib

    let sortedMembers = memberProfile.sort( $0.sequenceId < $1.sequenc

    let attributedString = NSattributedString(string: sortefMembers
        .enumerated()
        .map{
            index, member in

            var text = member.name

            if index % 2 == 0{
                text += "\(member.email)"
            }

            let attributes: [NSAttributedString.Key: Any] = [
                .font: UIfont.systemFont(ofSize: 16),
                .forgroundColor: member.favColor ?? UIColor.black
                ]

                return AttributedString(string: text + "\n", attribute
    }
    .joined

    return attributedString
```

0:00 / 14:31

1x 2x 5x

Question 13

Question 13 of 17

What's your favourite mobile App? Why? How would you improve or change it?

```
I like iMovie.

Why? I used it for editing movie and it very easy to use.
```

```
Improve: More audio editing features.
         Add tutorial and Help section
```

0:00 / 4:33

1x 2x 5x

Question 14

Question 14 of 17

Given the following functions:

func getThingFromSlowService() -> Thing

fund display(_ thing: Thing)

Write a function that calls getThingFromSlowService () on a background thread, and then calls displayThing () on the main thread passing in the return value from getThingFromSlowService().

```
func main(){
        DispatchQueue.global(qos: .background).async{ [weak self] in

            guard let strongSelf = self else{
                return
            }

            let this = getThindFromSlowService()

            DispatchQueue.main.asyc{
                    strongSelf.displa(thing)
                }

            }
    }
```

0:00 / 6:47

1x 2x 5x

Question 15

Question 15 of 17

Given the following functions that asynchronously return Int structs.

func service1() async throws -> Int

func service2() async throws -> Int

func service3() async throws -> Int

Write a function that adds all the integers and displays the result in the main queue when all services complete.

```
func addAndSisplaResut(){

    Task{
            do{
            let result1 = try await service1()
            let result2 = try await service2()
            let result3 = try await service3()

            await Task.withGroup(resultType:Int.self){group in
                    group.add(result1)
                    group.add(result2)
                    group.add(result3)

                    let sum = try await group.reduce(0, +)

                    DispatchQueue.main.async{
                        print(sum)
                    }
                }

            }
            catch{
                       DispatchQueue.main.async{
                            print("Error: \(error))
                         }
                }
        }



    }
```

0:00 / 10:21

1x 2x 5x

Question 16

Question 16 of 17

Write a Swift implementation of a UITableViewController that presents a UITableView that has one section and four cells.

Each cell should display the name of one of your favorite books, with the name of the author beneath it.

Tapping a cell should present an alert giving the Author's name with either his birthdate or one of his quotes that you really like.

Assume that your controller will NOT be instantiated from a xib file or storyboard.

```
struct Book{
        let name:String
        let author:String
        let details:String
    }
class BookListTableViewControler:UITableViewcontroler{

    let cellIdetifier = "cell"
    let bookList = [Book(name: "Book1", author: "Author 1", details: "

    override func viewDidLoad(){
        super.viewDidLoad()
        self.tableView.register(UITableViewCell.self, forCellReuableId
        }

    override func numberOfSections(in tableView:UITableView) -> Int{
        return 1
        }

    override fun numberOfRowsInSection(....){
        return book.count
        }

    override func tableview(tableView:UITableView, cellForRowAtIndex i
        let cell = tableView.dequeueReueableCell(withIdentifier: cellI

        let book = books[indexPath.row]
        cell.textLabel.text = book.name
        cell.authorLabel.text = book.authe

        return cell
    }

    override func tableView(tableView:UITable, didSelectRowAt indexPat
        let book = books[indexPath.row]

        let book = books[indexPath.row]
        let alert = UIAlertController(title: "Author: \(book.authoe)",
        let okAction = UIAlertAction(title:"OK", style.default, handle
        alert.addAction(okAction)

        present(alert, animated:true, completion:nil)
        }
```

0:00 / 16:27

1x 2x 5x

Question 17

```
Question 17 of 17

Complete the implementation below.

public enum HTTPMethod<T: Codable> {
  case get(T)
  case post(T)
  case patch(T)
  case put(T)
  case delete(T)
}

func networkResult<Request: Codable, Response: Codable, Result>
(endPoint: URL, method: HTTPMethod<Request> = .get, type: Request.Type,
subject: @escaping (Response) -> Result) async throws -> Result {

  //guard let urlRequest = request(endPoint: endPoint, params: params,
method: method) else { throw Error() }
  let (data, response) = try await URLSession.shared.data(for:
urlRequest)
  guard let response = response as? HTTPURLResponse else { throw
Error(code: ErrorCode.unknown) }

  // If HTTP response status code is 200 or 204, decode the JSON
response and fulfilled the promise
  // else if HTTP response status code is 4xx, decode the JSON body of
the error response and reject the promise with an error
  <your-code-goes-here>

  throw Error(code: ErrorCode.unknown)
}
```

```
func networkResult<Request: Codable, Response: Codable, Result>(endPoi

  //guard let urlRequest = request(endPoint: endPoint, params: params,
  let (data, response) = try await URLSession.shared.data(for: urlRequ
  guard let response = response as? HTTPURLResponse else { throw Error

  // If HTTP response status code is 200 or 204, decode the JSON respo
  // else if HTTP response status code is 4xx, decode the JSON body of
  if(responce.statusCode = 200 || responce.statusCode == 204){
      let decodeResponce = try JSONDecoder().decode(Response.self, fro
        subject(decodeResponce)
      }else if(400..<500).contains(responce.statusCode){
            let errorResponce = try JSONDecorder().decode(Responce.sel
            throw Error()
          }else{
              throw Error(code: ErrorCode.unknown)
              }


  throw Error(code: ErrorCode.unknown)
}
```

0:00 / 11:39

1x 2x 5x