iOS行動程式基礎開發上架

swift：可nil實體串接

本堂教學重點

1. 可nil串接替代強制打開
2. 定義可實現nil串接的類別
3. 透過可nil串接存取屬性
4. 透過可nil串接呼叫方法
5. 透過可nil串接存取Subscript
   - 可nil類型存取Subscript
6. 多層串接
7. 多層串接呼叫方法

1.可nil串接替代強制打開

```
class Person {
    var residence: Residence?
}

class Residence {
    var numberOfRooms = 1
}


let john = Person()

let roomCount = john.residence!.numberOfRooms
// this triggers a runtime error

if let roomCount = john.residence?.numberOfRooms {
    print("John's residence has \(roomCount) room(s).")
} else {
    print("Unable to retrieve the number of rooms.")
}
// Prints "Unable to retrieve the number of rooms."

```

1.可nil串接替代強制打開

```
john.residence = Residence()

if let roomCount = john.residence?.numberOfRooms {
    print("John's residence has \(roomCount) room(s).")
} else {
    print("Unable to retrieve the number of rooms.")
}
// Prints "John's residence has 1 room(s)."

```

2.定義可實現nil串接的類別

```
class Person {
    var residence: Residence?
}
```

```
class Residence {
    var rooms = [Room]()
    var numberOfRooms: Int {
        return rooms.count
    }
    subscript(i: Int) -> Room {
        get {
            return rooms[i]
        }
        set {
            rooms[i] = newValue
        }
    }
    func printNumberOfRooms() {
        print("The number of rooms is \(numberOfRooms)")
    }
    var address: Address?
}
```

2.定義可實現nil串接的類別

```
class Room {
    let name: String
    init(name: String) { self.name = name }
}

class Address {
    var buildingName: String?
    var buildingNumber: String?
    var street: String?
    func buildingIdentifier() -> String? {
        if let buildingNumber = buildingNumber, let street = street {
            return "\(buildingNumber) \(street)"
        } else if buildingName != nil {
            return buildingName
        } else {
            return nil
        }
    }
}
```

3.透過可nil串接存取屬性

```
let john = Person()
if let roomCount = john.residence?.numberOfRooms {
    print("John's residence has \(roomCount) room(s).")
} else {
    print("Unable to retrieve the number of rooms.")
}
// Prints "Unable to retrieve the number of rooms."
```

```
let someAddress = Address()
someAddress.buildingNumber = "29"
someAddress.street = "Acacia Road"
john.residence?.address = someAddress
```

```
func createAddress() -> Address {
    print("Function was called.")

    let someAddress = Address()
    someAddress.buildingNumber = "29"
    someAddress.street = "Acacia Road"

    return someAddress
}
john.residence?.address = createAddress()
```

4.透過可nil串接呼叫方法

```
func printNumberOfRooms() {
    print("The number of rooms is \(numberOfRooms)")
}


if john.residence?.printNumberOfRooms() != nil {
    print("It was possible to print the number of rooms.")
} else {
    print("It was not possible to print the number of rooms.")
}
// Prints "It was not possible to print the number of rooms."


if (john.residence?.address = someAddress) != nil {
    print("It was possible to set the address.")
} else {
    print("It was not possible to set the address.")
}
// Prints "It was not possible to set the address."
```

5.透過可nil串接存取Subscript

```
if let firstRoomName = john.residence?[0].name {
    print("The first room name is \(firstRoomName).")
} else {
    print("Unable to retrieve the first room name.")
}
// Prints "Unable to retrieve the first room name."

john.residence?[0] = Room(name: "Bathroom")

let johnsHouse = Residence()
johnsHouse.rooms.append(Room(name: "Living Room"))
johnsHouse.rooms.append(Room(name: "Kitchen"))
john.residence = johnsHouse

if let firstRoomName = john.residence?[0].name {
    print("The first room name is \(firstRoomName).")
} else {
    print("Unable to retrieve the first room name.")
}
// Prints "The first room name is Living Room."
```

5.透過可nil串接存取Subscript

可nil類型存取Subscript

```
var testScores = ["Dave": [86, 82, 84], "Bev": [79, 94, 81]]
testScores["Dave"]?[0] = 91
testScores["Bev"]?[0] += 1
testScores["Brian"]?[0] = 72
// the "Dave" array is now [91, 82, 84] and the "Bev" array is now [80, 94, 81]
```

6.多層串接

```
    if let johnsStreet = john.residence?.address?.street {
        print("John's street name is \(johnsStreet).")
    } else {
        print("Unable to retrieve the address.")
    }
    // Prints "Unable to retrieve the address."


    let johnsAddress = Address()
    johnsAddress.buildingName = "The Larches"
    johnsAddress.street = "Laurel Street"
    john.residence?.address = johnsAddress

    if let johnsStreet = john.residence?.address?.street {
        print("John's street name is \(johnsStreet).")
    } else {
        print("Unable to retrieve the address.")
    }
    // Prints "John's street name is Laurel Street."
```

7.多層串接呼叫方法

```
if let buildingIdentifier = john.residence?.address?.buildingIdentifier() {
    print("John's building identifier is \(buildingIdentifier).")
}
// Prints "John's building identifier is The Larches."
```

```
if let beginsWithThe =
    john.residence?.address?.buildingIdentifier()?.hasPrefix("The") {
    if beginsWithThe {
        print("John's building identifier begins with \"The\".")
    } else {
        print("John's building identifier does not begin with \"The\".")
    }
}
// Prints "John's building identifier begins with "The"."
```