Smartphone programming Class - 4

short line

# Swift Programming Language: Classes

# Declaring a class:

enum Country{

case India

case USA

case China

case Canada

case Mexico

}

class Person{

var firstName = "Ashish"

var lastName = "Ashish"

var age = 16

var countryOfBirth: Country = .India

}

This is how you would create an object of class

let ashish = Person();

print(ashish.firstName)

print(ashish.lastName)

print(ashish.age)

print(ashish.countryOfBirth)

Let’s create a new person

let john = Person();

john.firstName = "John"

john.lastName = "Mercer"

john.age = 72

john.countryOfBirth = .USA

print(john.firstName)

print(john.lastName)

print(john.age)

print(john.countryOfBirth)

This is good but I don’t like that every person is created with same last name and first name. I want to initialize the class with these variables.

We will have a new init() function which would be the constructor for the class

init(firstName: String , lastName: String, age: Int, countryOfBirth: Country){

self.firstName = firstName

self.lastName = lastName

self.age = age

self.countryOfBirth = countryOfBirth

}

Now the object of person would be initialized as

let ashish = Person(firstName: "Ashish", lastName: "Ashish", age: 16, countryOfBirth: Country.India)

let john = Person(firstName: "John", lastName: "Mercer", age: 72, countryOfBirth: Country.USA)

Lets assume we want to have default init function and we can have 2 init function.

We would use Convenience init function

init() {}

convenience init(firstName: String , lastName: String, age: Int, countryOfBirth: Country){

self.init()

self.firstName = firstName

self.lastName = lastName

self.age = age

self.countryOfBirth = countryOfBirth

}

We can have multiple convenience init() functions too

# Methods:

When we have functions inside the class they are called methods. Let’s add a method Eat()

override func Eat(foodItem: String){

print("Person: \(firstName) is eating \(foodItem)")

}

All Person are Animals, lets create a class called Animal

enum AnimalType {

case Human

case Monkey

case Dog

case Elephant

case Bird

case Fish

}

class Animal {

var age = 0

var animalType = AnimalType.Human

init(animalType: AnimalType){

self.animalType = animalType

}

}

We can add the Eat function in Animal class

func Eat(foodItem: String){

print("I am an animal, I am hungry and need to eat")

}

Humans can have their own Eat function and override the Animal function

override func Eat(foodItem: String){

super.Eat(foodItem: foodItem)

print("Person: \(firstName!) is eating \(foodItem)")

}

# Optionals:

We cannot have a class variable declared and not have an initial value. In our Person class we are having firstName and lastName variables always initialized to “Ashish” Instead what we can do is that we cannot have anything assigned to the value.

var firstName: String?

var lastName: String?

Problem comes when the variables are not initialized to a value and they are being used.

In that case we will first check if the variables are initialized then we will use them

if firstName != nil {

print("Person: \(firstName!) is eating \(foodItem)")

}

Another way of using it is

if let name = firstName{

print("Person: \(name) is eating \(foodItem)")

}