Cheatsheets / **Learn Swift**

Structures

Structure Creation

Structures, or structs, are used to programmatically represent a real-life object in code. Structures are created with the `struct` keyword followed by its name and then body containing its properties and methods.

```
struct Building {
  var address: String
  var floors: Int

  init(address: String, floors: Int,
color: String) {
    self.address = address
    self.floors = floors
  }
}
```

Default Property Values

A structure's properties can have preassigned default values to avoid assigning values during initialization. Optionally, these property's values can still be assigned a value during initialization.

```
struct Car {
  var numOfWheels = 4
  var topSpeed = 80
}

var reliantRobin = Car(numOfWheels: 3)

print(reliantRobin.numOfWheels) //
Prints: 3
print(reliantRobin.topSpeed)    //
Prints: 80
```

Structure Instance Creation

A new instance of a structure is created by using the name of the structure with parentheses `()` and any

```
struct Person {
```

necessary arguments.

```
  var name: String
  var age: Int

  init(name: String, age: Int) {
    self.name = name
    self.age = age
  }
}

// Instance of Person:
var morty = Person(name: "Morty", age:
14)
```

Checking Type

The built-in function `type(of:)` accepts an argument and returns the type of the argument passed.

```
print(type(of: "abc")) // Prints: String
print(type(of: 123))   // Prints: 123
```

`init()` Method

Structures can have an `init()` method to initialize values to an instance's properties. Unlike other methods, The `init()` method does not need the `func` keyword. In its body, the `self` keyword is used to reference the actual instance of the structure.

```
struct TV {
  var screenSize: Int
  var displayType: String

  init(screenSize: Int, displayType:
String) {
    self.screenSize = screenSize
    self.displayType = displayType
  }
}

var newTV = TV(screenSize: 65,
displayType: "LED")
```

Structure Methods

Methods are like functions that are specifically called on an instance. To call the method, an instance is appended with the method name using dot notation followed by parentheses that include any necessary arguments.

```
struct Dog {
  func bark() {
    print("Woof")
  }
}

let fido = Dog()
fido.bark() // Prints: Woof
```

Mutating Methods

Structure methods declared with the `mutating` keyword allow the method to affect an instance's own properties.

```
struct Menu {
  var menuItems = ["Fries", "Burgers"]

  mutating func addToMenu(dish: String)
{
    self.menuItems.append(dish)
  }
}

var dinerMenu = Menu()

dinerMenu.addToMenu(dish: "Toast")
print(dinerMenu.menuItems)
// Prints: ["Fries", "Burgers", "Toast"]
```