Cheatsheets / **Learn Swift**

Classes

Swift Class

A class is used to programmatically represent a real-life object in code. Classes are defined by the keyword `class` followed by the class name and curly braces that store the class's properties and methods.

```swift
// Using data types:
class Student {
  var name: String
  var year: Int
  var gpa: Double
  var honors: Bool
}


// Using default property values:
class Student {
  var name = ""
  var year = 0
  var gpa = 0.0
  var honors = false
}
```

Instance of a Class

Creating a new instance of a class is done by calling a defined class name with parentheses `()` and any necessary arguments.

```swift
class Person {
  var name = ""
  var age = 0
}

var sonny = Person()

// sonny is now an instance of Person
```

Class Properties

Class properties are accessed using dot syntax, i.e. `.property` .

```swift
var ferris = Student()

ferris.name = "Ferris Bueller"
ferris.year = 12
ferris.gpa = 3.81
ferris.honors = false
```

`init()` Method

Classes can be initialized with an `init()` method and corresponding initialized properties. In the `init()` method, the `self` keyword is used to reference the actual instance of the class assign property values.

```swift
class Fruit {
  var hasSeeds = true
  var color: String

  init(color: String) {
    self.color = color
  }
}

let apple = Fruit(color: "red")
```

Inheritance

A class can inherit, or take on, another class's properties and methods:

- The new inheriting class is known as a subclass.
- The class that the subclass inherits from is known as its superclass.

```swift
// Suppose we have a BankAccount class:

class BankAccount {
  var balance = 0.0

  func deposit(amount: Double) {
    balance += amount
  }

  func withdraw(amount: Double) {
    balance -= amount
  }
}


// And we want a new SavingsAccount
class that inherits from BankAccount:

class SavingsAccount: BankAccount {
  var interest = 0.0

  func addInterest() {
    let interest = balance * 0.005
    self.deposit(amount: interest)
  }
}

// Here, the new SavingsAccount class
(subclass) automatically gains all of
the characteristics of BankAccount class
(superclass). In addition, the
SavingsAccount class defines a .interest
property and a .addInterest() method.
```

Overriding

A subclass can provide its own custom implementation of a property or method that is inherited from a superclass. This is known as overriding.

```
// Suppose we have a BankAccount class:

class BankAccount {
  var balance = 0.0

  func deposit(amount: Double) {
    balance += amount
  }

  func withdraw(amount: Double) {
    balance -= amount
  }
}


// Suppose we want a new SavingsAccount
class and we want to override the
.withdraw() method from its superclass
BankAccount:

class SavingsAccount: BankAccount {
  var interest = 0.0
  var numWithdraw = 0

  func addInterest() {
    let interest = balance * 0.01
    self.deposit(amount: interest)
  }

  override func withdraw(amount: Double)
{
    balance -= amount
    numWithdraw += 1
  }
}
```

Reference Types

Classes are reference types, while structures are value types, classes are reference types.
Unlike value types, reference types are not copied when they are assigned to a variable or constant, or when they are passed to a function. Rather than a copy, a reference to the same existing instance is used.

Save Print Share