

OOP - Class



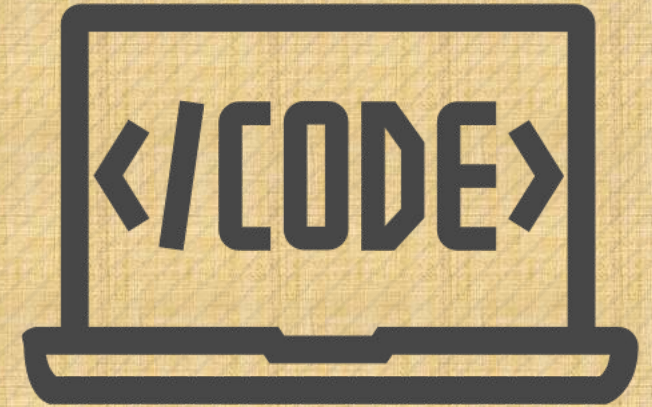
Class Definition as on brilliant.org -

In object-oriented programming, a class is a blueprint for creating objects (a particular data structure), providing initial values for state (member variables or attributes), and implementations of behavior (member functions or methods).

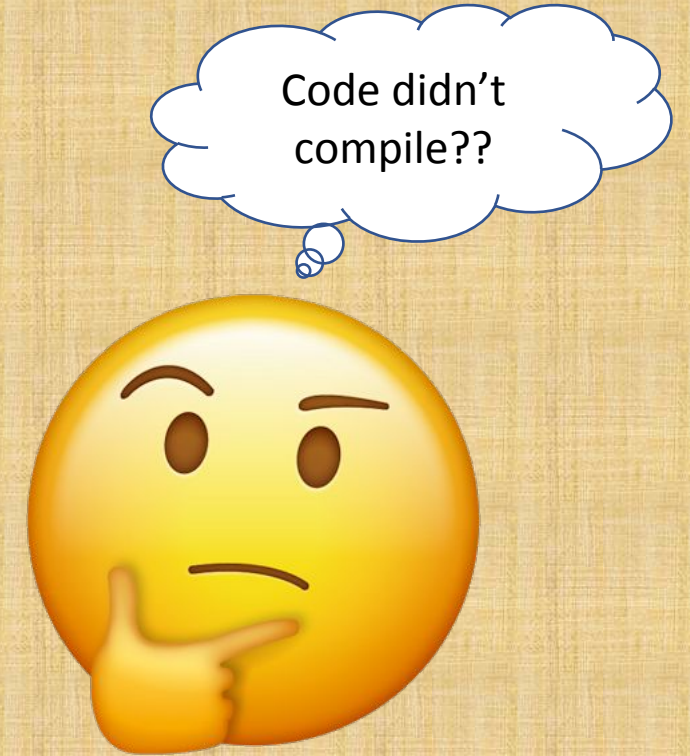
OOP -Class



```
class ComputerLanguage{  
    //Member to Store Language Name  
    var languageName: String  
  
    //Member to identify if it is a compiled language  
    var isCompiled: Bool  
  
    //Member storing the market adoption %ages  
    var adoptionRate: Double  
  
    //Member identifying if languages has OOP support  
    var supportsOOP: Bool  
  
    //and so on...  
}
```



OOP -Class





OOP -Class

```
class ComputerLanguage{  
    var languageName: String  
    var isCompiled: Bool?  
    var adoptionRate: Double  
    var supportsOOP: Bool?  
  
    init(languageName: String, adoptionRate: Double){  
        self.languageName = languageName  
        self.adoptionRate = adoptionRate  
        self.isCompiled = true  
        self.supportsOOP = true  
    }  
}
```

```
ComputerLanguage(languageName: "Swift",  
adoptionRate: 5.9)
```



Either add initialization
or make the member
optional

OOP -Objects



An instance is a specific object created from a particular class. Classes are used to create and manage new objects and support inheritance.

OOP -Objects



```
var java =  
  ComputerLanguage(lan  
    guageName: "Java",  
    isCompiled: true,  
    adoptionRate: 40.2,  
    supportsOOP: true)
```



```
var swift =  
  ComputerLanguage(l  
    anguageName:  
    "Swift", isCompiled:  
    true, adoptionRate:  
    5.9, supportsOOP:  
    true)
```



```
var kotlin =  
  ComputerLanguage(lan  
    guageName: "Kotlin",  
    isCompiled: true,  
    adoptionRate: 7.8,  
    supportsOOP: true)
```

OOP – Comment on classes and files



We are playing with playgrounds, so you will see me defining classes (sometimes multiple classes) and objects in the same file. For all professional implementations you will include each class in its individual file.

OOP -Encapsulation



Encapsulation refers to the bundling of data with the methods that operate on that data, or the restricting of direct access to some of an object's components. Encapsulation is used to hide the values or state of a structured data object inside a class, preventing unauthorized parties' direct access to them.

Think Access Specifiers

OOP -Encapsulation



```
class AmortizationCalculator{  
    private var myInterestRate: Double = 3.4 //I may be pulling this from  
    network call or from DB
```

```
    public func getInterestRateApplied() -> Double{  
        return self.myInterestRate  
    }  
}
```

```
var calculator = AmortizationCalculator()  
print (calculator.getInterestRateApplied())
```



OOP -Inheritance

As the name says –

Pass on the wealth





OOP -Inheritance

```
class ComputerLanguage{
    var languageName: String
    init() {
        languageName = ""
    }
}

class OOPLanguage: ComputerLanguage{
    public func teachClassMeaning(){}
    public func teachObjects(){}
    public func teachInheritance(){}
}

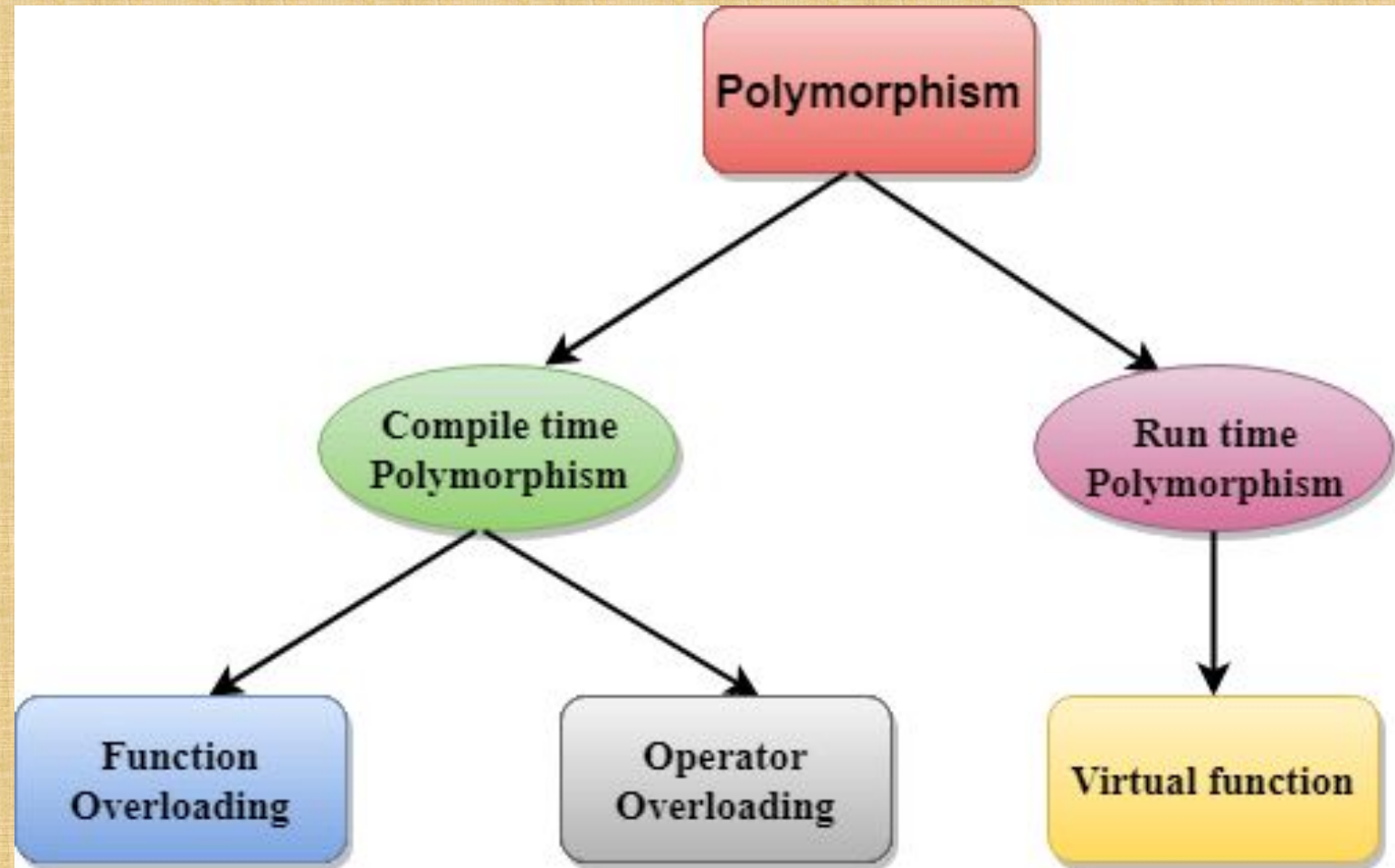
class Swift: OOPLanguage{
    public func coverOOPBasics(){
        self.teachClassMeaning();
        self.teachObjects();
        self.teachInheritance();
    }
}

var swift = Swift()
swift.languageName = "Swift"
```


OOP - Polymorphism



OOP -Polymorphism





OOP –Polymorphism - Compile time

```
class ComputerLanguage{  
    var languageName: String  
    init(languageName: String){  
        self.languageName = languageName  
    }  
    public func doSyntaxAnalysis(){  
        print ("My default syntax analysis implementation for: \(languageName)")  
    }  
  
    public func doSyntaxAnalysis(ignoringWarnings: Bool){  
        print ("My default syntax analysis implementation for: \(languageName) while ignoring  
warnings: \(ignoringWarnings)")  
    }  
}
```

var swift = ComputerLanguage(languageName: "Swift")
swift.doSyntaxAnalysis()
swift.doSyntaxAnalysis(ignoring Warnings: **true**)

Method overloading



OOP –Polymorphism - Compile time

```
class ComputerLanguage{
    var languageName: String
    init(languageName: String){
        self.languageName = languageName
    }
    public func doSyntaxAnalysis(){
        print ("My default syntax analysis implementation for: \(languageName)")
    }
}

class Swift: ComputerLanguage{
    init(){
        super.init(languageName: "Swift")
    }
    override public func doSyntaxAnalysis(){
        print ("Doing my custom implementation for: \(languageName)")
    }
}

var swift = Swift()
swift.doSyntaxAnalysis()
```

Method overriding

OOP –Polymorphism - Run time



```
protocol ComputerLanguageProtocol{  
    var languageName: String{ get set }  
    func doSyntaxAnalysis()  
}
```

```
class ComputerLanguage: ComputerLanguageProtocol{  
    var languageName: String  
    init(languageName: String){  
        self.languageName = languageName  
    }  
    public func doSyntaxAnalysis(){  
        print ("My default syntax analysis implementation for: \(languageName)")  
    }  
}
```



OOP – Polymorphism - Run time

```
class Swift: ComputerLanguage{
    init(){
        super.init(languageName: "Swift")
    }
    override public func doSyntaxAnalysis(){
        print ("Doing my custom implementation for: \(languageName)")
    }
}

class Java: ComputerLanguage{
    init(){
        super.init(languageName: "Java")
    }
}

var swift = Swift()
swift.doSyntaxAnalysis()
var java = Java()
java.doSyntaxAnalysis()
```


OOP -Abstraction



ABSTRACTION is the concept of object-oriented programming that "shows" only essential attributes and "hides" unnecessary information. The main purpose of abstraction is hiding the unnecessary details from the users. Abstraction is selecting data from a larger pool to show only relevant details of the object to the user. It helps in reducing programming complexity and efforts. It is one of the most important concepts of OOPs.

OOP -Abstraction



You don't need to worry about
what Xbox does on hitting X



OOP -Abstraction

```
protocol ComputerLanguageProtocol{
    var languageName: String{ get set }
    func doSyntaxAnalysis()
}
extension ComputerLanguageProtocol{
    public func doSyntaxAnalysis(){
        print ("My default syntax analysis implementation for: \(languageName)")
    }
}
class Swift: ComputerLanguageProtocol{
    var languageName = "Swift"
    public func doSyntaxAnalysis(){
        print ("Doing my custom implementation for: \(languageName)")
    }
}
class Java: ComputerLanguageProtocol{
    var languageName = "Java"
}
Swift().doSyntaxAnalysis()
Java().doSyntaxAnalysis()
```