Access Control

-Devendranath

Access control **restricts access** **to your code from code in other source files and modules. This feature enables you to** **hide the implementation details of your code** **, and to** **specify a preferred interface** **through which that code can be accessed and used.**

Abstraction **can be achieved through Access Control.**

NOTE: Don't play with Access Specifiers in Playgrounds

Swift Provides 5 types of Access Specifiers.

OPEN, FilePrivate

Public:(*Higher level access)* This means _everyone_ can read and write the property.

Internal:(*Medium Level Access*) This means only accessible in _same module_ in which it is defined. Doesn't work in other frameworks.

Private:(*Low Lever Access*) This means that only accessible in _same class_ where it is declared. Outside of that class is not accessible

Default access specifier is **internal**.

```swift
class AccessSpecifiersV3
{
    public var aPublicVariable = 10
    internal var aInternalVariable = 20
    private var aPrivateVariable = 30

    public func aPublicMethod()
    {
        print("Public Method: \(aPublicVariable) , \
(aInternalVariable) , \(aPrivateVariable)")
    }

    internal func aInternalMethod()
    {
        print("Internal Method: \(aPublicVariable) , \
(aInternalVariable) , \(aPrivateVariable)")
    }

    private func aPrivateMethod()
    {
        print("Private Method: \(aPublicVariable) , \
(aInternalVariable) , \(aPrivateVariable)")
    }
}
NOTE: All three are accessible in same class where those are
declared
```

Accessing in it's subclass

```
class SubClass: AccessSpecifiersV3
{
    func accessSuperClassVarsAndMethods()
    {
        aPublicMethod()
        aInternalMethod()
//ERROR: aPrivateMethod() // Not Accessible due to
private

        print(aPublicVariable)
        print(aInternalVariable)
//ERROR: print(aPrivateVariable)
    }
}
```

Thank You