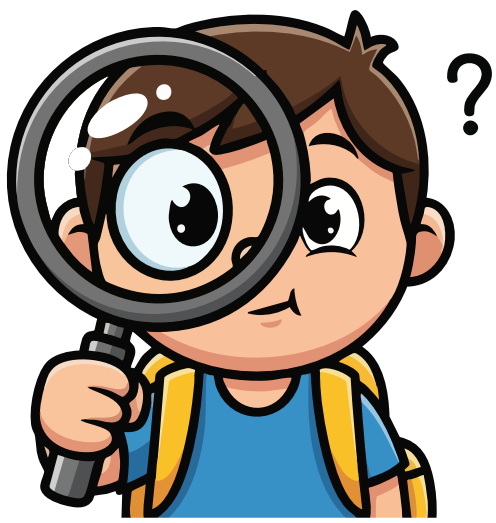
Access Control

Access control restricts where your code can be accessed from.It helps in **encapsulation**, **safety**, and **clean architecture**.



By Shubam Gupta

🔐 Access Control in Swift

1.OPEN (MOST PERMISSIVE)

✅ Where accessible?

- Same module
- Other modules
- Can be subclassed & overridden outside the module

📌 Used when:

- Creating frameworks / SDKs
- APIs meant for external customization.

open example

```
open class BaseViewController {
    open func setupUI() { }
}
```

⚠️ Only class and class members can be open

2.PUBLIC

✅ Where accessible?

- Same module
- Other modules
- ❌ Cannot be overridden or subclassed outside module

📌 Used when:

- Exposing functionality but preventing modification.

public example

```
public struct User {
    public let name: String
}
```

🔍 OPEN VS PUBLIC

Feature	open	public
Access outside module	✅	✅
Subclass outside module	✅	❌
Override outside module	✅	❌

3.INTERNAL (DEFAULT)

✅ Where accessible?

- Anywhere within the same module.

internal (Default) example

```
class NetworkManager {
    func fetchData() { }
}
```

📌 If you don’t specify access level → it is internal.

4.FILEPRIVATE

✅ Where accessible?

- Only within the same file.

fileprivate example

```
fileprivate class CacheManager { }

fileprivate func logError() { }
```

📌 Used when:

- Multiple types in same file share logic
- Helpers not meant to escape the file.

5.PRIVATE (MOST RESTRICTIVE)

✅ Where accessible?

- Only inside the enclosing declaration.

private example

```
class LoginViewModel {
    private var password: String = ""
}
```

📌 Used for:

- Sensitive data
- Implementation details
- Prevent misuse.

🧠 Access Control Hierarchy

- open → public → internal → fileprivate → private.

⚠️ Important Swift Rule (Interview Favorite)

A member cannot be more accessible than its enclosing type.

Invalid example

```
private class A {
    public func test() { } // ❌
}
```

📦 Access Control in Real iOS Projects.

Scenario	Recommended
ViewModels internal	private
Helpers in same file	fileprivate
App-level services	internal
SDK / Framework API	public / open
Prevent subclassing	public

⭐ Interview One-Liner

“Access control in Swift defines the visibility of types and members across files and modules, helping enforce encapsulation and safer APIs.”

Thank you

By Shubam Gupta