The private(set) access control level in Swift allows you to define a property that can only be set from within the scope in which it is defined. This can be useful for preventing unauthorized changes to a property, or for ensuring that a property is only set in a specific way.

To use the private(set) access control level, you simply add the private(set) keyword before the property declaration. For example, the following code defines a property called myProperty that can only be set from within the MyClass class:

Swift .

```
class MyClass {
    private(set) var myProperty: String = "Hello, world!"
}
```

Even though the myProperty property is declared as public, it can only be set from within the MyClass class. This is because of the private(set) access control level.

Here is an example of how to use the private(set) access control level:

```
class MyClass {
    private(set) var myProperty: String = "Hello, world!"

    func setMyProperty(to newProperty: String) {
        myProperty = newProperty
    }
}

// Accessing the myProperty property from outside of MyClass will
not work.
// let myClass = MyClass()
// myClass.myProperty = "New value" // Error: 'myProperty' is
immutable

// Setting the myProperty property from within MyClass will work.
let myClass = MyClass()
myClass.setMyProperty(to: "New value")

// Accessing the myProperty property from outside of MyClass will
now return the new value.
print(myClass.myProperty) // Prints "New value"
```

The private(set) access control level is a powerful tool that can help you to write more secure and maintainable code. By using the private(set) access control level, you can restrict access to properties and prevent unauthorized changes.