



Class modifiers



sealed class

Class hierarchy

```
interface Expr
```

```
class Num(val value: Int) : Expr
```

```
class Sum(val left: Expr, val right: Expr) : Expr
```

```
fun eval(e: Expr): Int { ... }
```

```
// 1 + (2 + 3)
```

```
eval(Sum(Num(1), Sum(Num(2), Num(3)))) // 6
```

Class hierarchy

```
interface Expr
class Num(val value: Int) : Expr
class Sum(val left: Expr, val right: Expr) : Expr

fun eval(e: Expr): Int = when (e) {
    is Num -> e.value
    is Sum -> eval(e.left) + eval(e.right)
}
```

Compiler error: 'when' expression must be exhaustive, add necessary 'else' branch

Class hierarchy

```
interface Expr
class Num(val value: Int) : Expr
class Sum(val left: Expr, val right: Expr) : Expr


fun eval(e: Expr): Int = when (e) {
    is Num -> e.value
    is Sum -> eval(e.left) + eval(e.right)
    else -> throw IllegalArgumentException(
        "Unknown expression")
}
```

sealed modifier

Restricts class hierarchy:
all subclasses must be located in the same file

```
sealed class Expr
class Num(val value: Int) : Expr()
class Sum(val left: Expr, val right: Expr) : Expr()

fun eval(e: Expr): Int = when (e) {
    is Num -> e.value
    is Sum -> eval(e.left) + eval(e.right)
}
```





Inner and nested classes



Which class (nested or inner) stores a reference to an outer class?

In Java	In Kotlin	Class declared within another class
<code>static class A</code>	<code>class A</code> (by default)	nested class
<code>class A</code> (by default)	<code>inner class A</code>	inner class





Which class (nested or inner) stores a reference to an outer class?

In Java	In Kotlin	Class declared within another class
<code>static class A</code>	<code>class A</code> (by default)	nested class
<code>class A</code> (by default)	<code>inner class A</code>	inner class



Which class (nested or inner) stores a reference to an outer class?

In Java	In Kotlin	Class declared within another class
<code>static class A</code>	<code>class A (by default)</code>	nested class
<code>class A (by default)</code>	<code>inner class A</code>	inner class



Which class (nested or inner) stores a reference to an outer class?

In Java	In Kotlin	Class declared within another class
<code>static class A</code>	<code>class A</code> (by default)	nested class
<code>class A</code> (by default)	<code>inner class A</code>	inner class



Which class (nested or inner) stores a reference to an outer class?

In Java	In Kotlin	Class declared within another class
<code>static class A</code>	<code>class A</code> (by default)	nested class
<code>class A</code> (by default)	<code>inner</code> class A	inner class

inner modifier

adds reference to the outer class:

```
class A {  
    class B  
    inner class C {  
        ...this@A...  
    }  
}
```



Class delegation

Class delegation

```
interface Repository {  
    fun getById(id: Int): Customer  
    fun getAll(): List<Customer>  
}  
  
interface Logger {  
    fun logAll()  
}  
  
class Controller(  
    val repository: Repository,  
    val logger: Logger  
) : Repository, Logger {  
    ...  
}
```


Class delegation

```
class Controller(  
    val repository: Repository,  
    val logger: Logger  
) : Repository, Logger {  
  
    override fun getById(id: Int) = repository.getById(id)  
  
    override fun getAll(): List<Customer> = repository.getAll()  
  
    override fun logAll() = logger.logAll()  
}
```

Class delegation

```
class Controller(  
    repository: Repository,  
    logger: Logger  
) : Repository by repository, Logger by logger
```

implements an interface by
delegating to ...

Class delegation

```
interface Repository {  
    fun getById(id: Int): Customer  
    fun getAll(): List<Customer>  
}
```

```
interface Logger {  
    fun logAll()  
}
```

```
class Controller(  
    repository: Repository,  
    logger: Logger  
) : Repository by repository, Logger by logger
```

```
fun use(controller: Controller) {  
    controller.logAll()  
}
```