Coroutines

Light-weight Threads

Like threads, Coroutines can run in parallel, wait for each other, and communicate with each other

Coroutine !=Thread

Coroutines are very very cheap-almost free. Create thousand of them without any memory issue. “Best for modern applications”

How to use Coroutines?

Unline threads, for coroutines the application by default does not wait for it to finish its execution

What is ‘Suspend’ modifier?

What are suspending functions?

Suspending Function

1. A Function with a ‘Suspend’ modifier is known as suspending function
2. The suspending functions are only allowed to be called from a coroutine or from another suspending function
3. They cannot be called from outside a coroutine

GlobalScope.launch() is non-blocking in nature whereas runBlocking() blocks the thread in which it operates

Summary

1. What are coroutines?
2. Threads vs Coroutines
3. Integrate coroutines to project
4. What are suspending functions?
5. Create coroutines: GlobalScope.launch{} and runBlocking {}

Keep track of threads

Kotlin Coroutine Builders: launch, async, and runBlocking along with GlobalScope companion object

What You Should Already Know?

1. What are Coroutines?
   -Purpose of using Coroutines

What You Should Already Know?

1. What are Coroutines?
   -Purpose of using Coroutines
2. Coroutines vs. Threads

What You Should Already Know?

1. What are Coroutines?
   -Purpose of using Coroutines
2. Coroutines vs. Threads
3. What is suspend modifier and what are suspending functions?

What You Should Already Know?

1. What are Coroutines?
   -Purpose of using Coroutines
2. Coroutines vs. Threads
3. What is suspend modifier and what are suspending functions?
4. Integrate coroutine in kotlin project using Gradle Maven

What You Should Already Know?

1. What are Coroutines?
   -Purpose of using Coroutines
2. Coroutines vs. Threads
3. What is suspend modifier and what are suspending functions?
4. Integrate coroutine in kotlin project using Gradle Maven
5. Create your first coroutine
   -GlobalScope.launch { } and runBlocking { }

What are Coroutine Builders?

Coroutine builders are used for creating coroutines

Coroutine Builders

Launch

Coroutine Builders

Launch

async

Coroutine Builders

Launch

async

runblocking

Coroutine Builders

Launch

GlobalScope.launch {}

async

runblocking

Coroutine Builders

Launch

GlobalScope.launch {}

async

runblocking

Coroutine Builders

Launch

GlobalScope.launch {}

async

GlobalScope.async {}

runblocking

Coroutine Builders

Launch

GlobalScope.launch {}

async

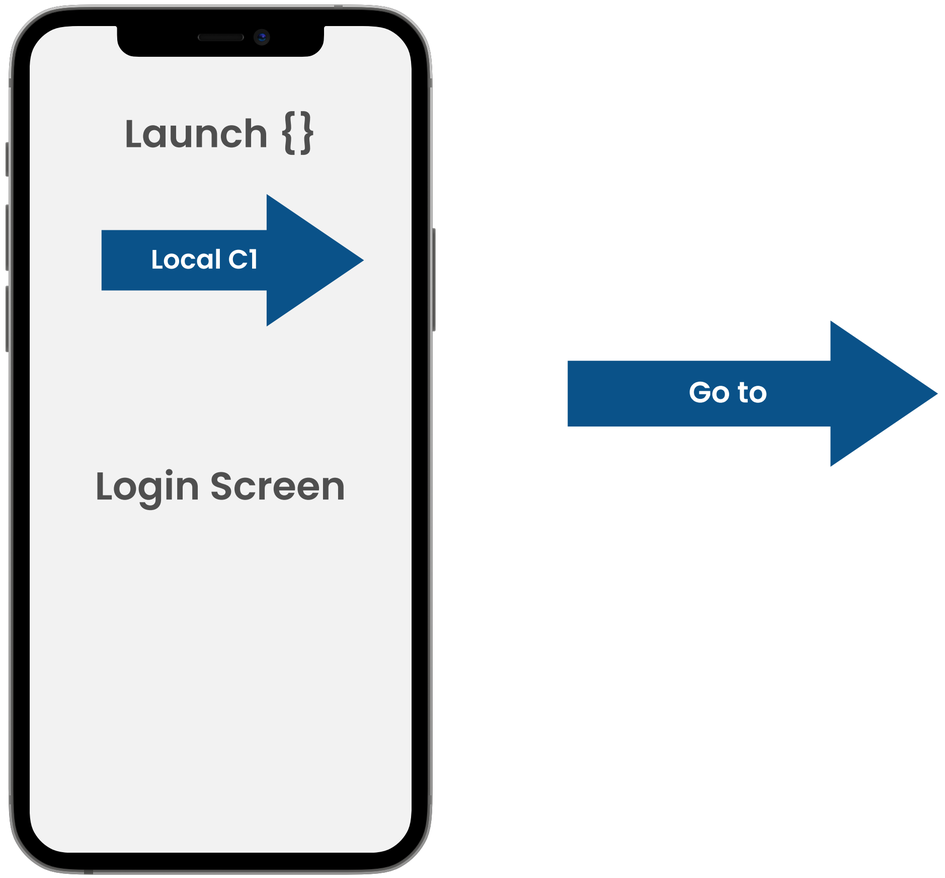
GlobalScope.async {}

runblocking

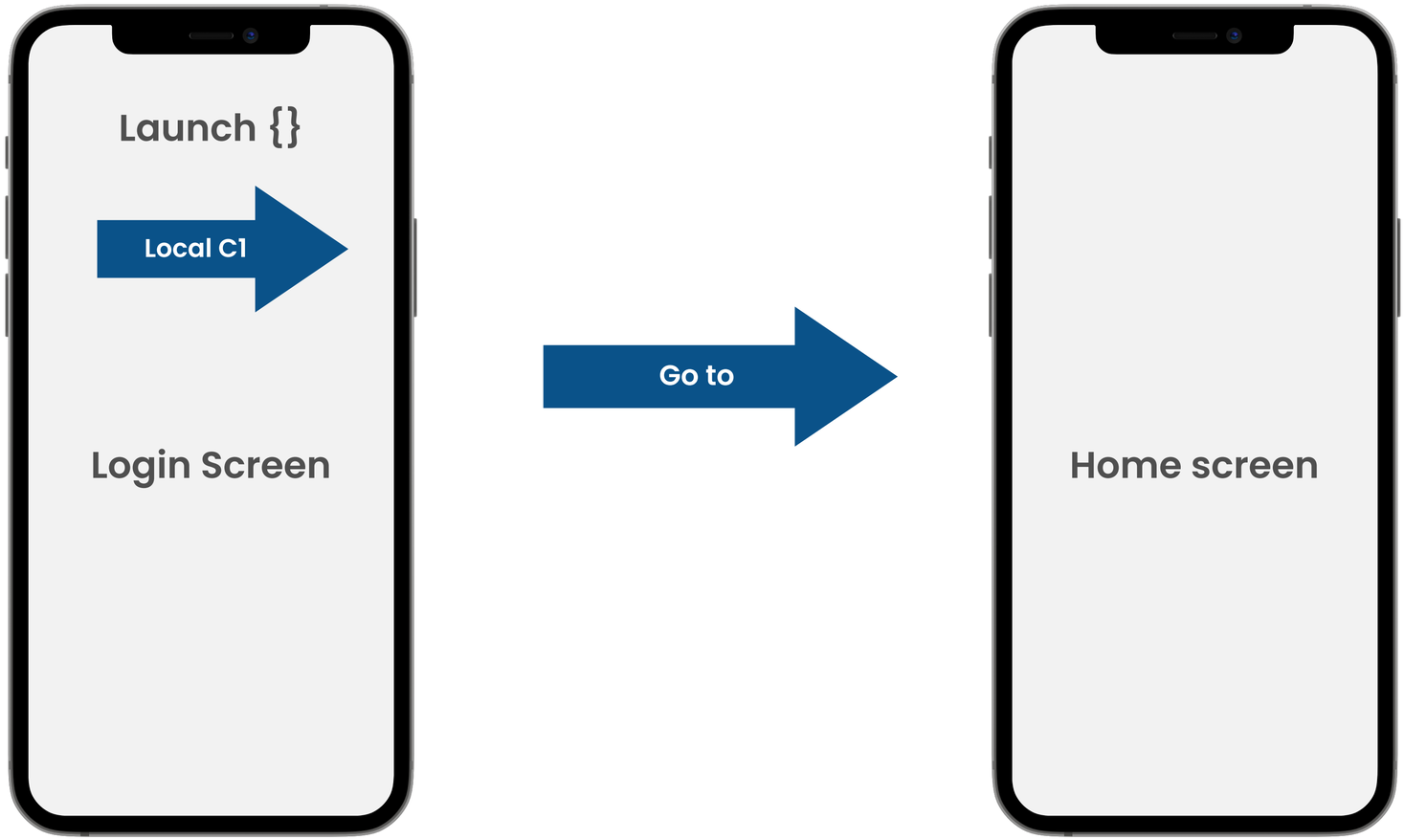
Life time of Application

Life time of Application

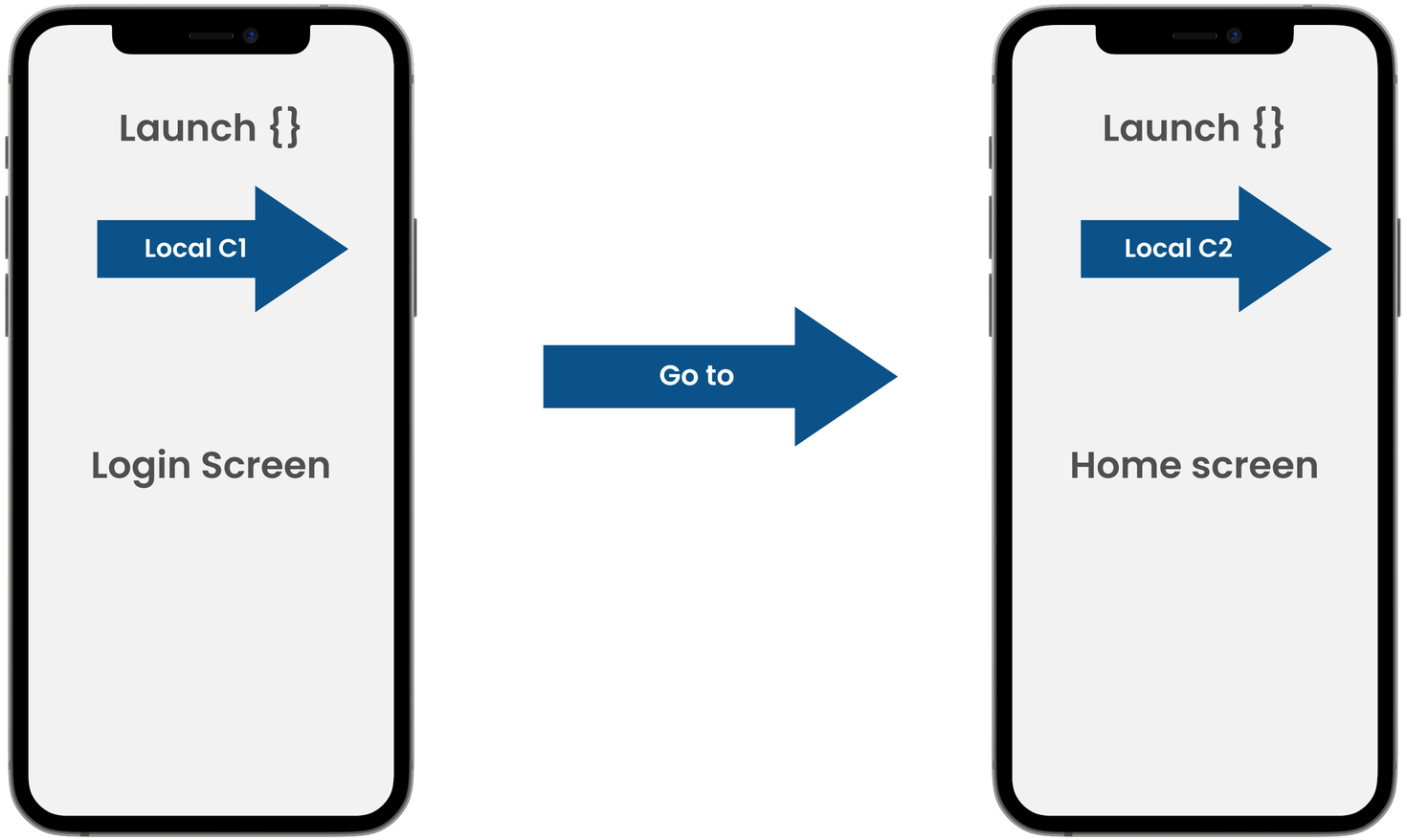
Life time of Application



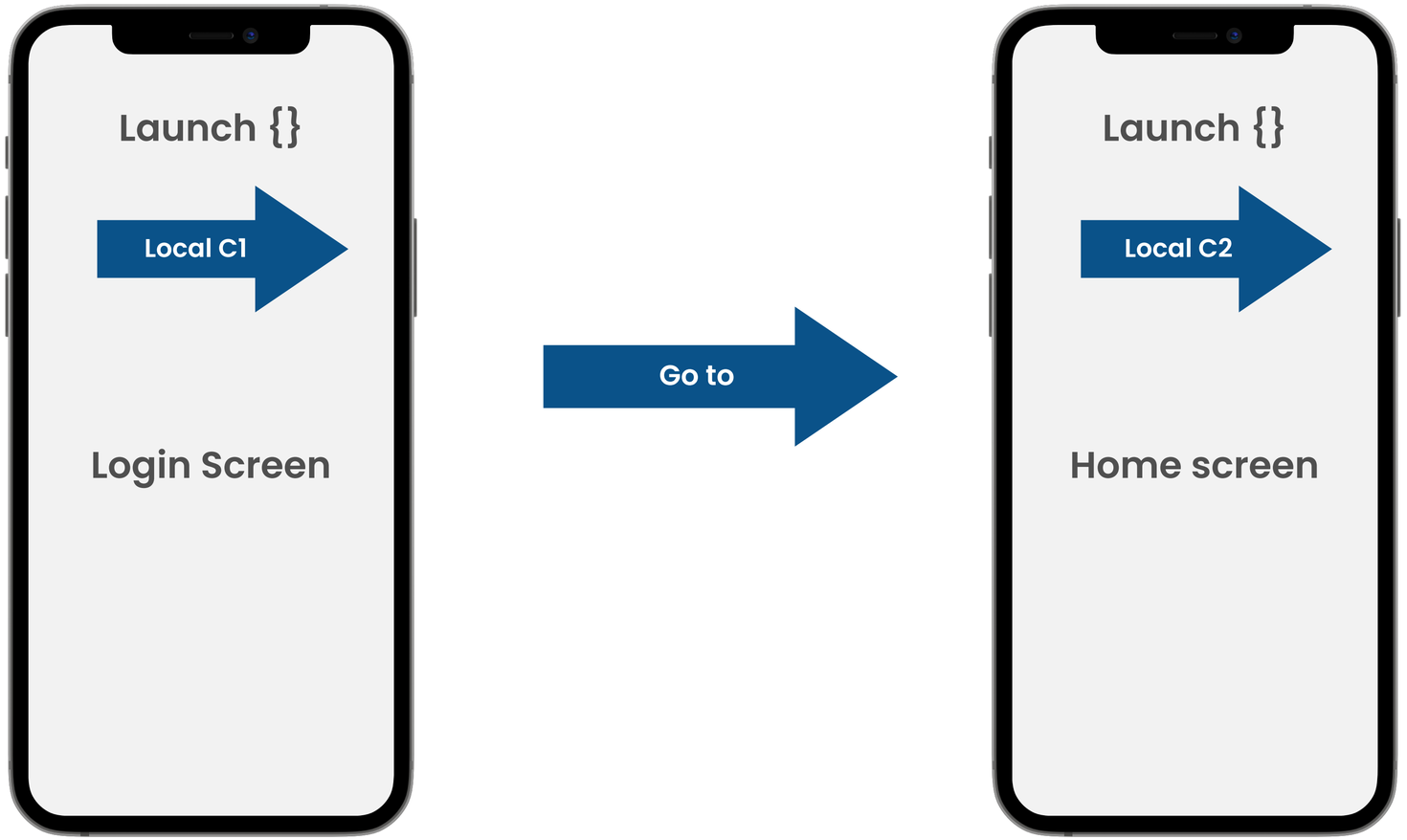
Life time of Application



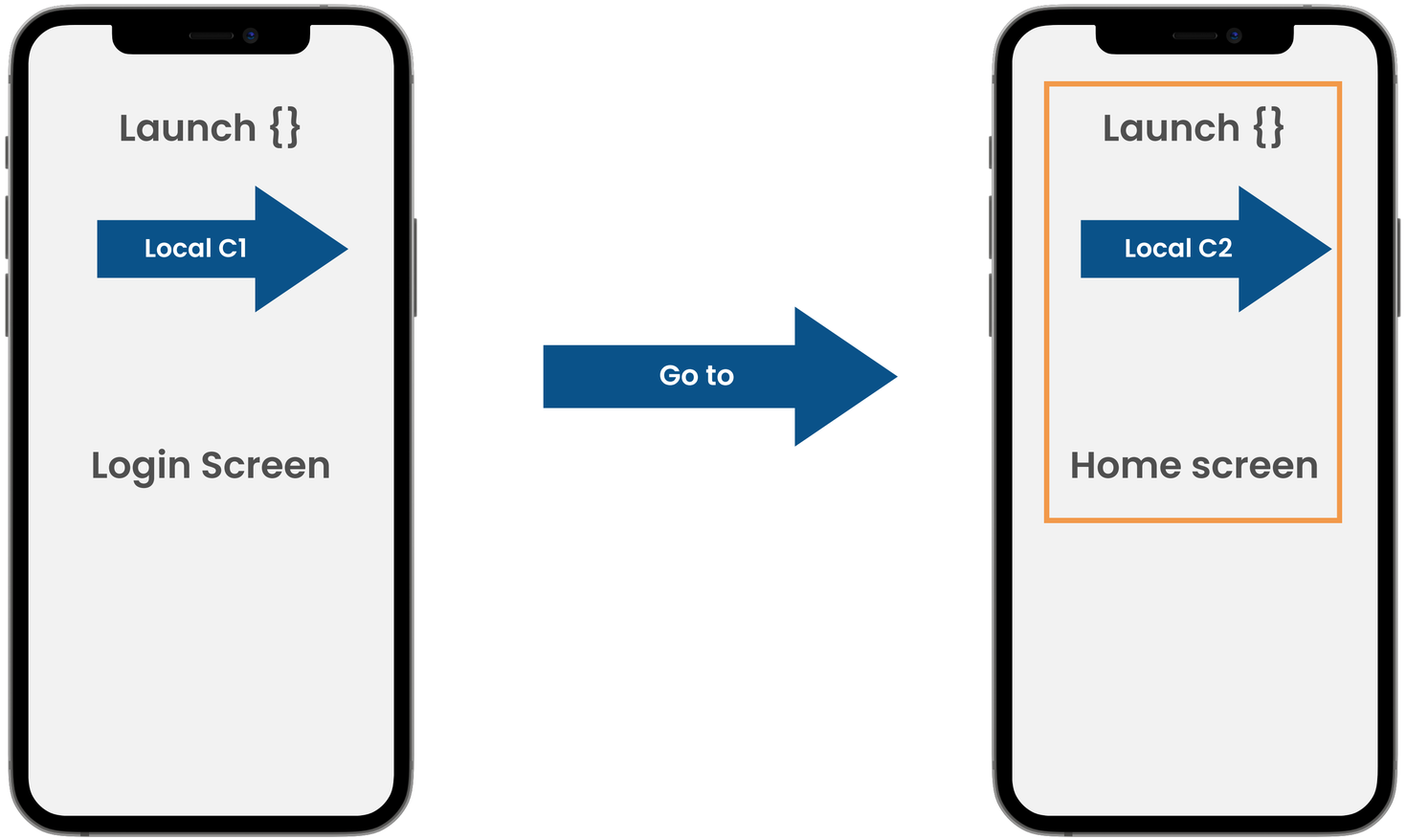
Life time of Application



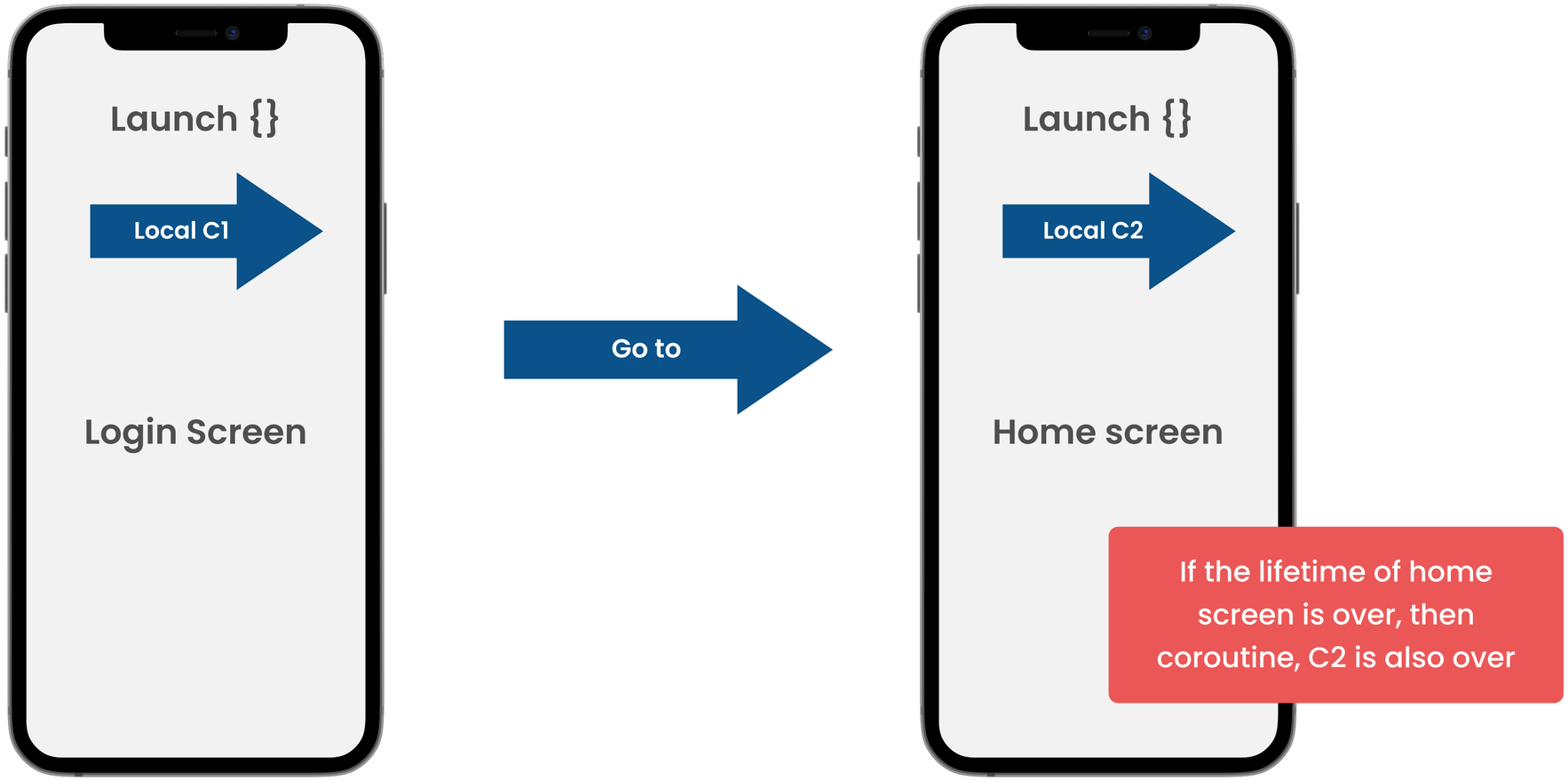
Life time of Application



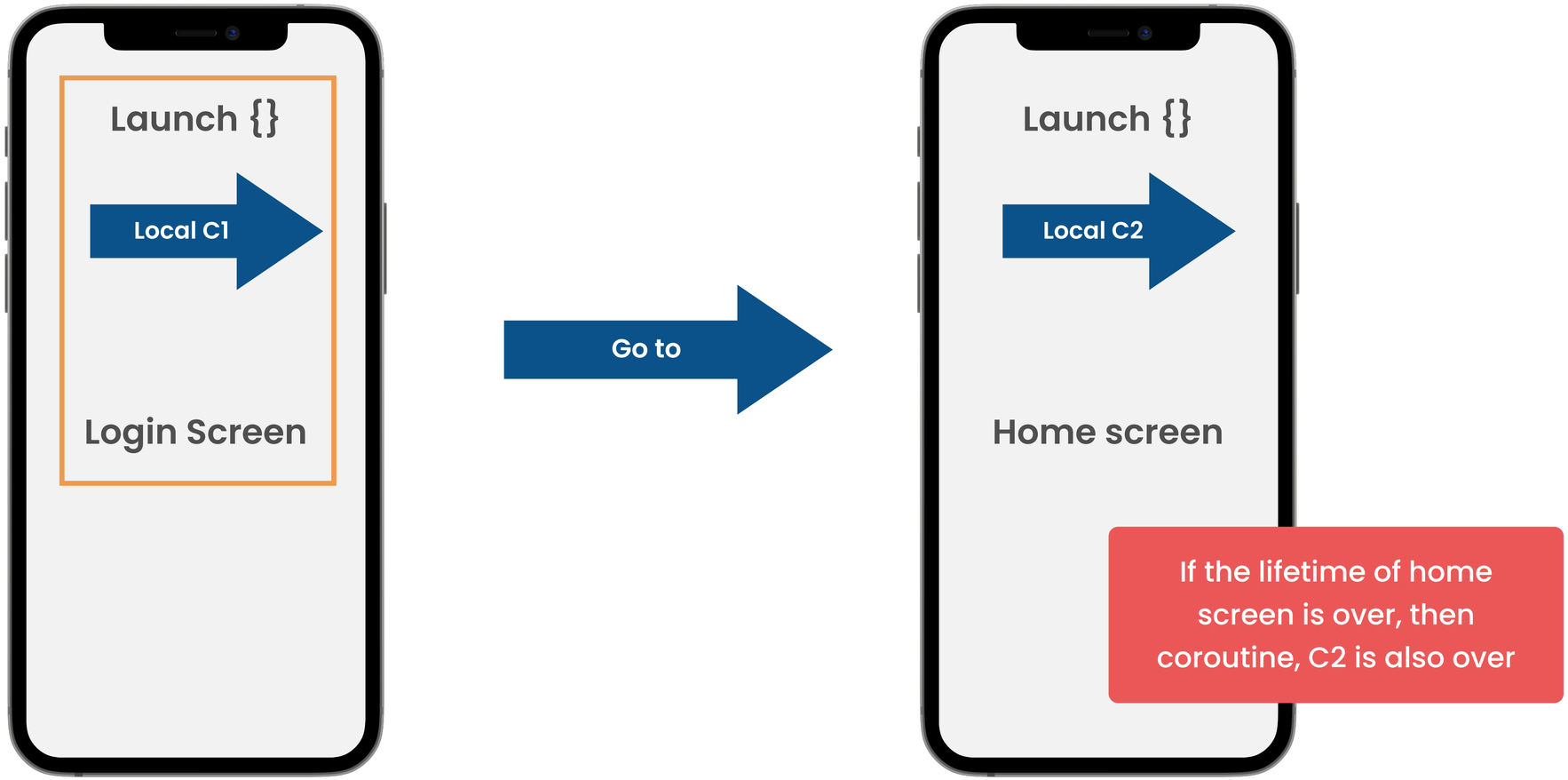
Life time of Application



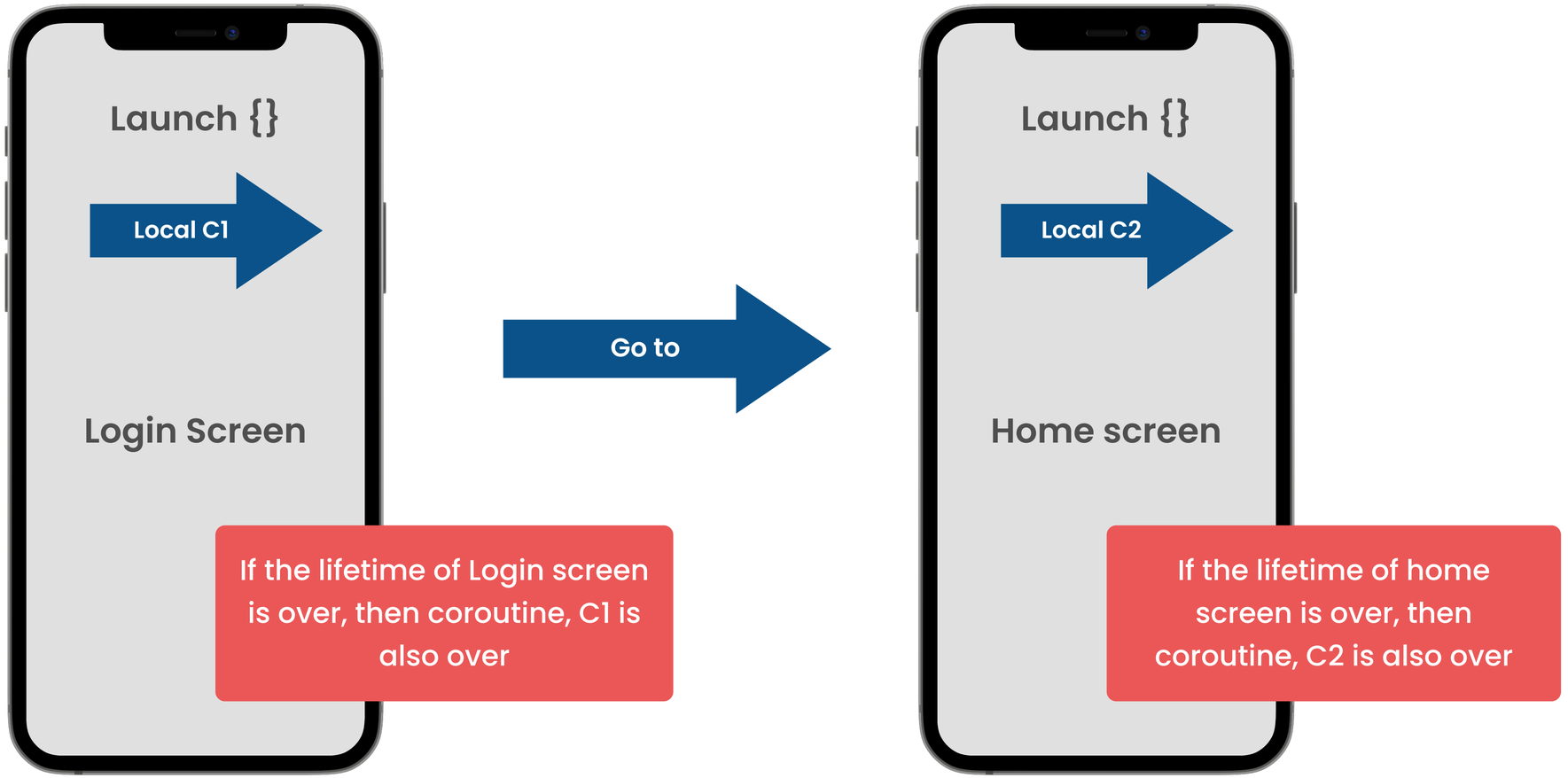
Life time of Application



Life time of Application

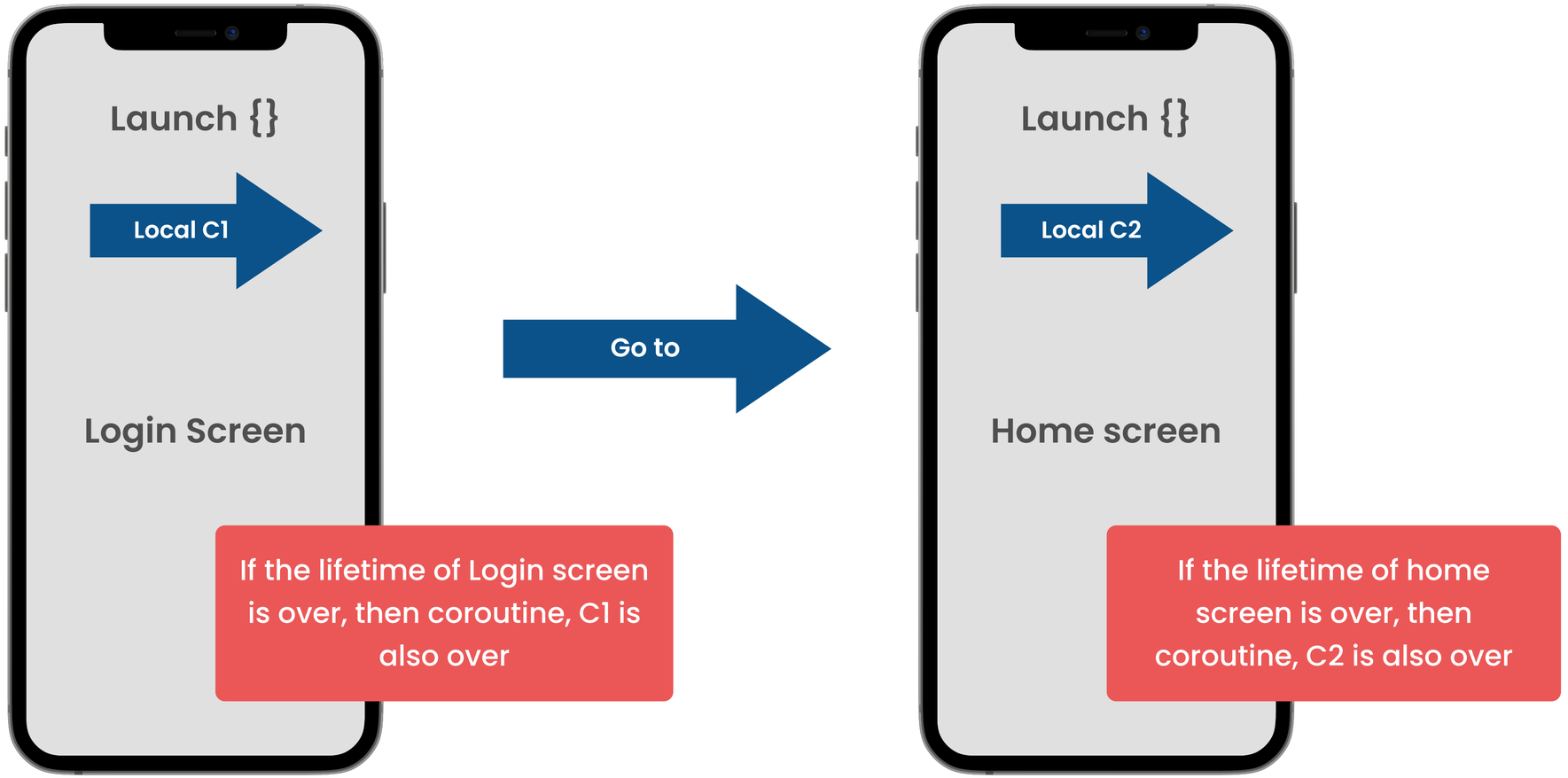


Life time of Application



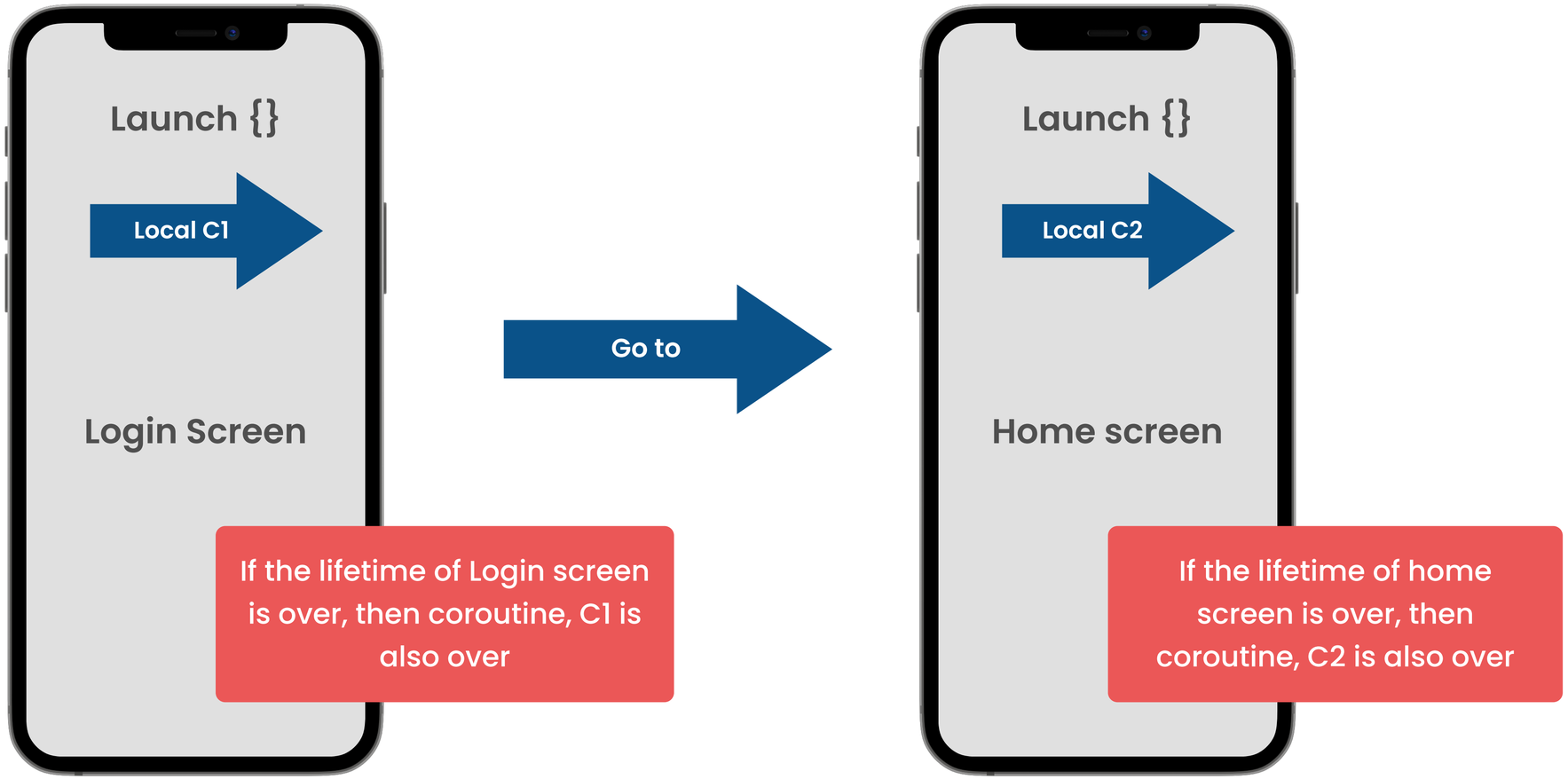
Life time of Application

GlobalScope.launch {} // Create coroutines at global (app) level



Life time of Application

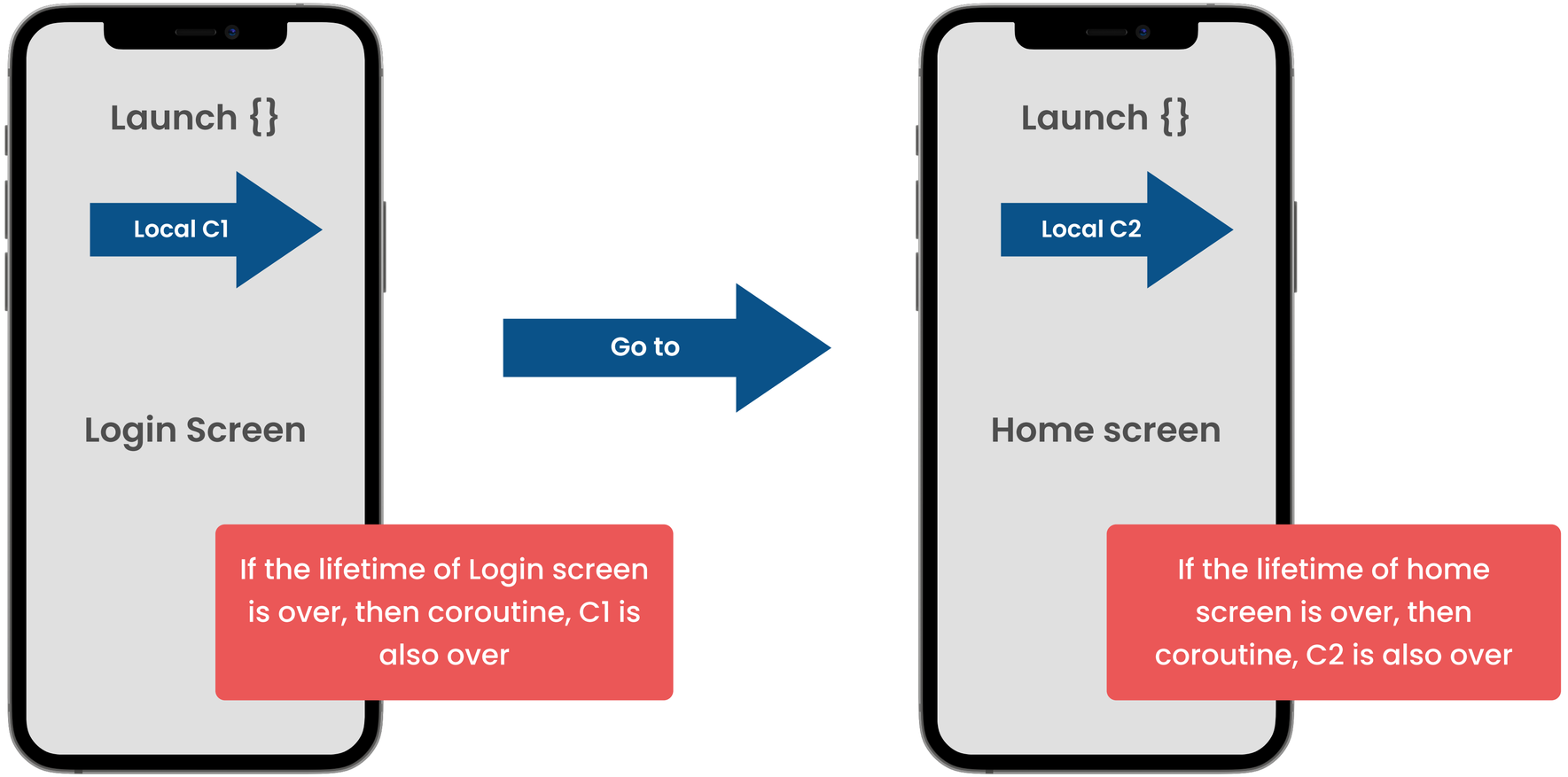
```
GlobalScope.launch {}          // Create coroutines at global (app) level
```



Life time of Application

GlobalScope.launch {} // Create coroutines at global (app) level

Global Coroutines are top-level coroutines and survive the entire life of the application.



```
// Creates coroutine at global (app) level
GlobalScope.launch {




}

 // Creates coroutine at local scope
launch {




}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {



}
```

```
 // Creates coroutine at local scope
launch {



}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {
                        // File download
                        // Play music
}

 // Creates coroutine at local scope
launch {



}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                                // File download
                                // Play music

}

 // Creates coroutine at local scope
launch {




}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                        // File download

                        // Play music

}
```

```
 // Creates coroutine at local scope
launch {




}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                    // File download

                    // Play music

}


 // Creates coroutine at local scope
launch {

                    // some data computation

                    // Login operation

}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                    // File download

                    // Play music

}


 // Creates coroutine at local scope
launch {

                    // some data computation

                    // Login operation

}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                    // File download

                    // Play music

}


 // Creates coroutine at local scope
launch {

                    // some data computation

                    // Login operation

}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                    // File download

                    // Play music

}
```

Discouraged. Use only when needed

```
 // Creates coroutine at local scope
launch {

                    // some data computation

                    // Login operation

}
```

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                    // File download

                    // Play music

}
```

Discouraged. Use only when needed

```
 // Creates coroutine at local scope
launch {

                    // some data computation

                    // Login operation

}
```

By default, use launch{}

```
// Creates coroutine at global (app) level
GlobalScope.launch {

                    // File download

                    // Play music

}
```

Discouraged. Use only when needed

```
 // Creates coroutine at local scope
launch {

                    // some data computation

                    // Login operation

}
```

By default, use launch{}