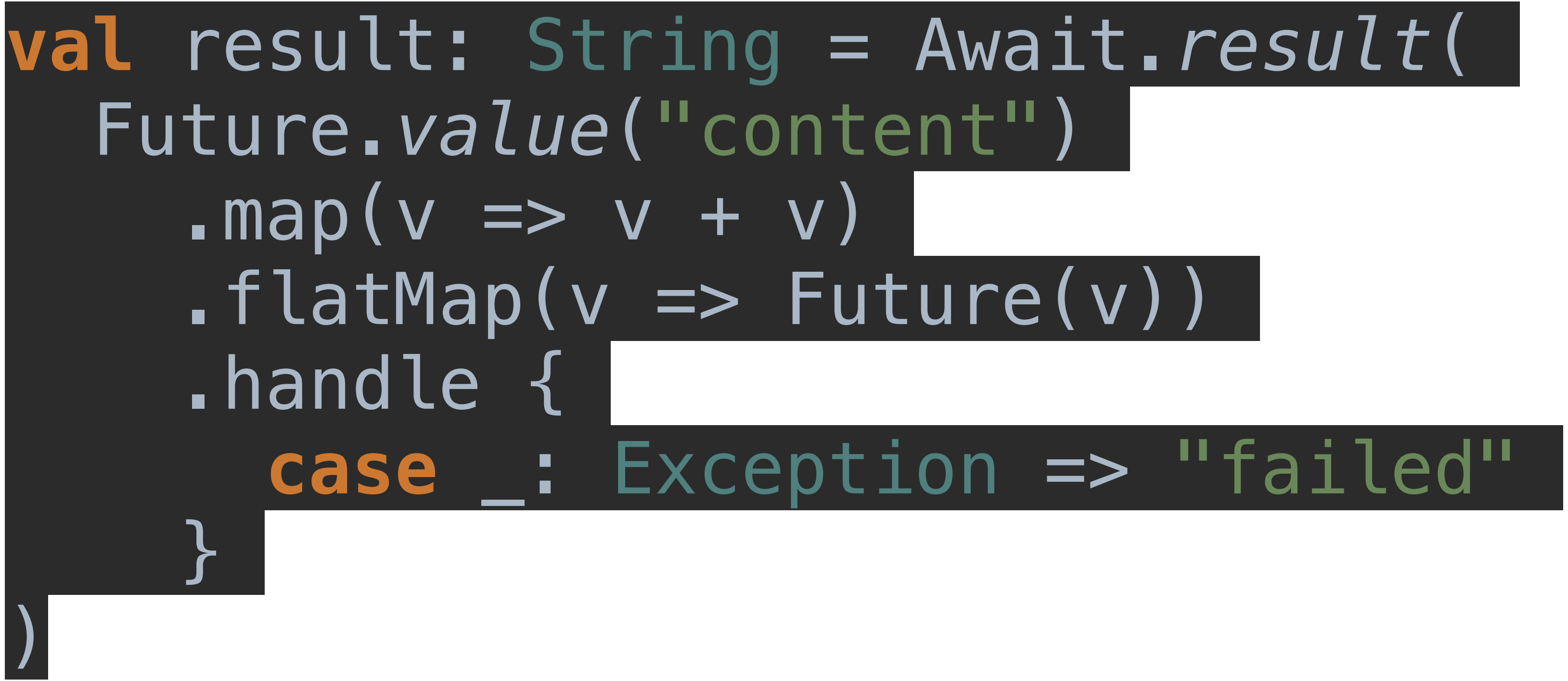

```scala
val result: String = Await.result(
  Future.value("content")
    .map(v => v + v)
    .flatMap(v => Future(v))
    .handle {
      case _: Exception => "failed"
    }
)
```

Concept: Future

“Represents the result of an Asynchronous operation.”

- 2 end states: Value and Exception
- Manipulate with: map() / flatMap() / handle() / rescue()
- Can get the result with Await - but never block!!

```
val result: String = Await.result(
  Future.value("content")
    .map(v => v + v)
    .flatMap(v => Future(v))
    .handle {
      case _: Exception => "failed"
    }
)
```

Concept: Service

“It turns an Request into an (eventual) Response”

ie. it’s a function!

```
Service: Request => Future[Response]
```