Key points

- Combine is a declarative, reactive framework for processing asynchronous events over time. It aims to solve existing problems, like unifying tools for asynchronous programming, dealing with mutable state and making error handling a starting team player.
- Combine revolves around three main types: **publishers** to emit events over time, **operators** to asynchronously process and manipulate upstream events and **subscribers** to consume the results and do something useful with them.

SINK operator:

it simply provides an easy way to attach a subscriber with closures to handle output from a publisher

```
var subscriptions = Set<AnyCancellable>()

let just = Just("Hello world!")
_ = just
  .sink(
    receiveCompletion: {
      print("Received completion", $0)
    },
    receiveValue: {
      print("Received value", $0)
    }).store(in: subscriptions)
```

Output:
Received value Hello world!
Received completion finished

ASSIGN operator:

the built-in assign(to:on:) operator enables you to assign the received value to a KVO-compliant property of an object.

```swift
func exampleOfAssign() {
  // 1
  class SomeObject {
    var value: String = "" {
      didSet {
        print(value)
      }
    }}
  let object = SomeObject()
  let publisher = ["Hello", "world!"].publisher
  _ = publisher
    .assign(to: \.value, on: object).cancel()
}
```

Output:
Hello
World!

Code Explanation:

1. **Define a class with a property that has a didSet property observer that prints the new value.**
2. **Create an instance of that class.**
3. **Create a publisher from an array of strings.**
4. **Subscribe to the publisher, assigning each value received to the value property of the object.**