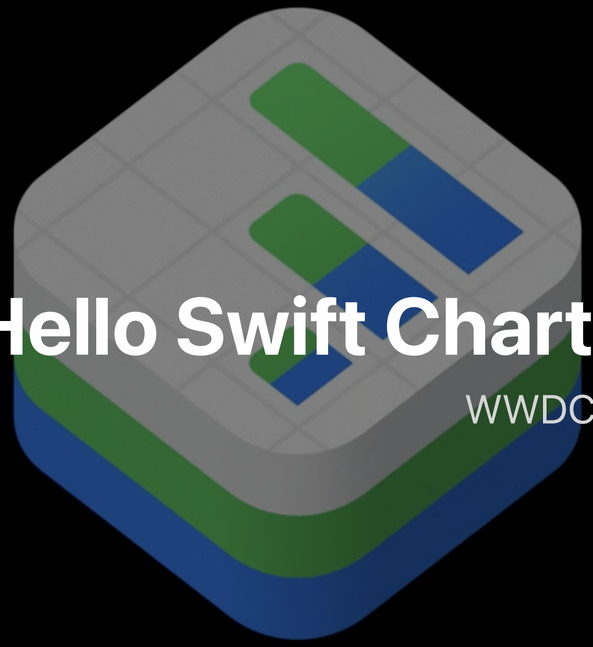


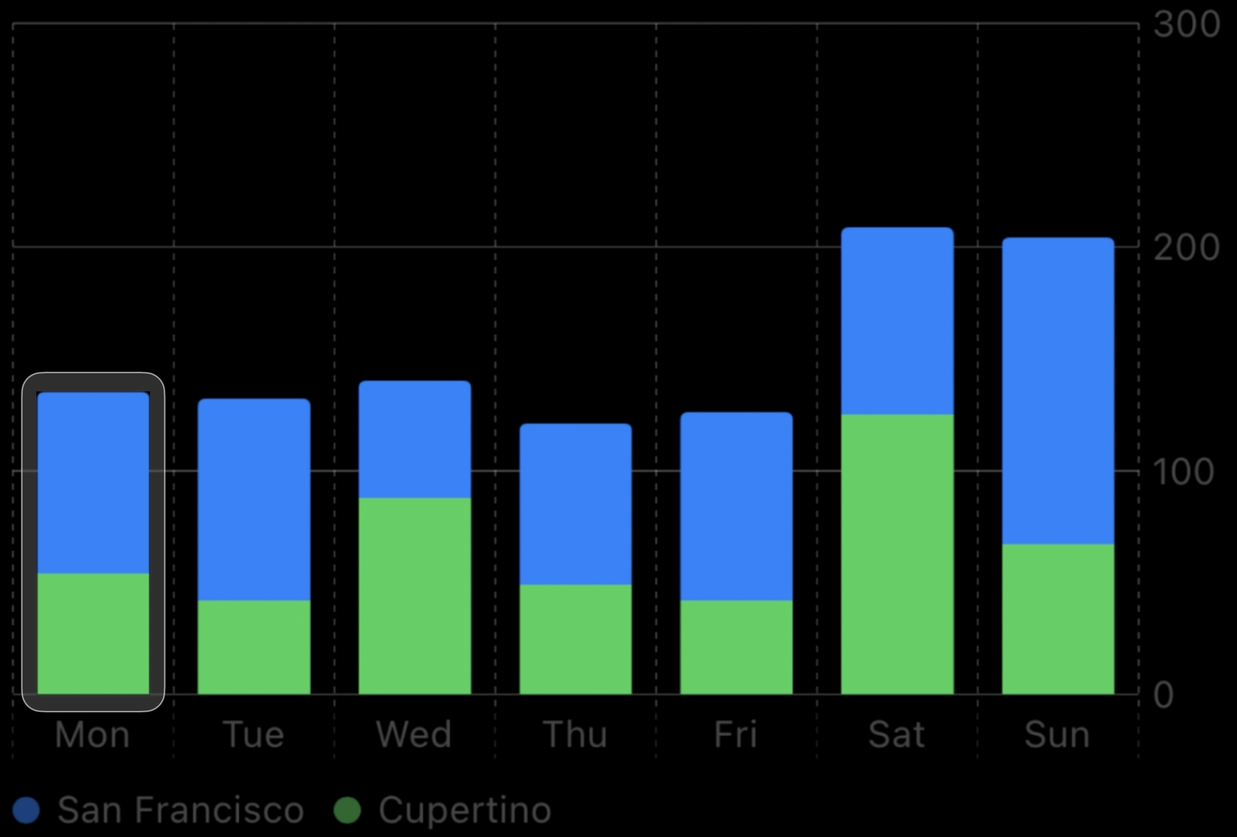
Hello Swift Charts

WWDC22

Apple-designed charts

Same syntax as SwiftUI

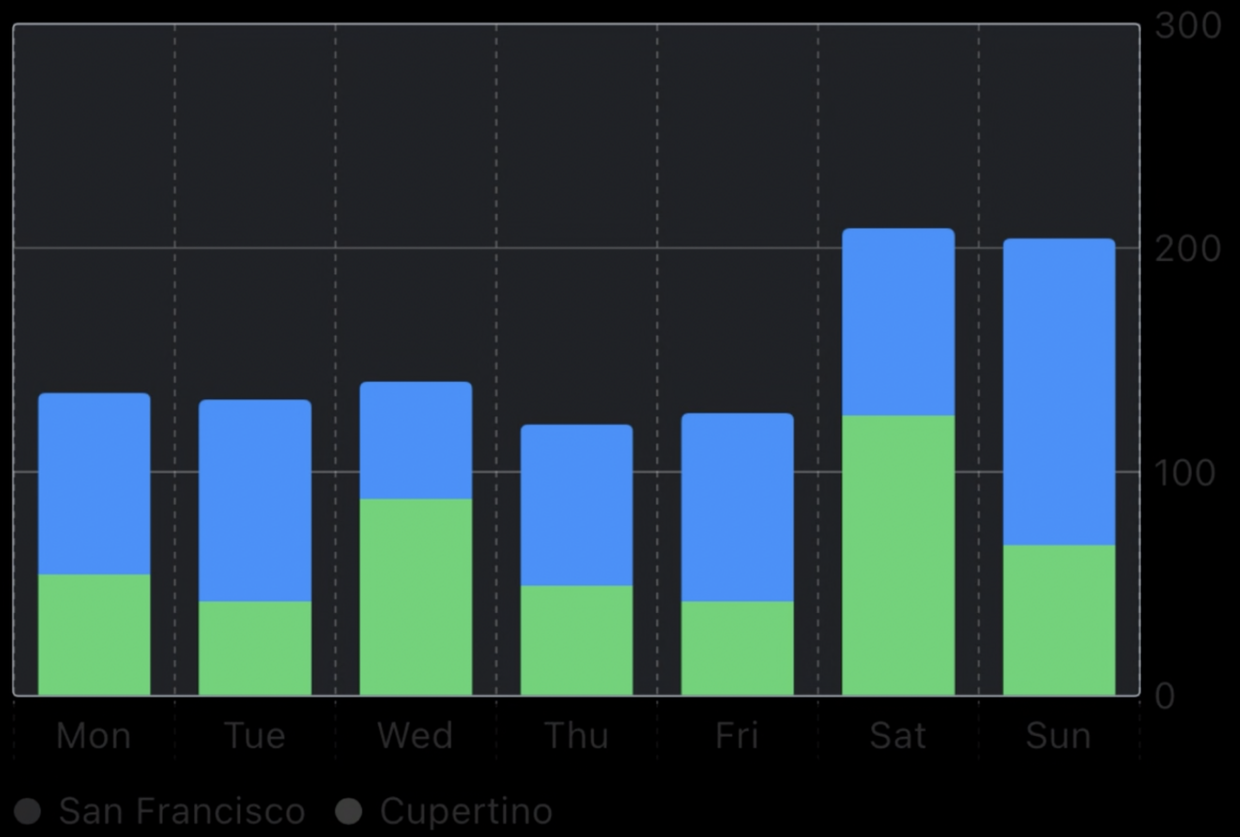
Marks


300
200
100
0
Mon
Tue
Wed
Thu
Fri
Sat
Sun
San Francisco
Cupertino

Axes

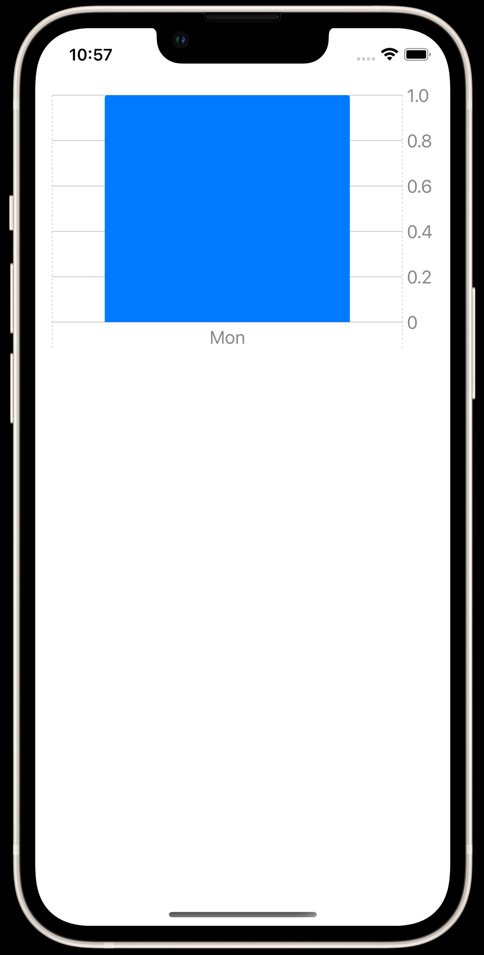
Legends

Plot Area

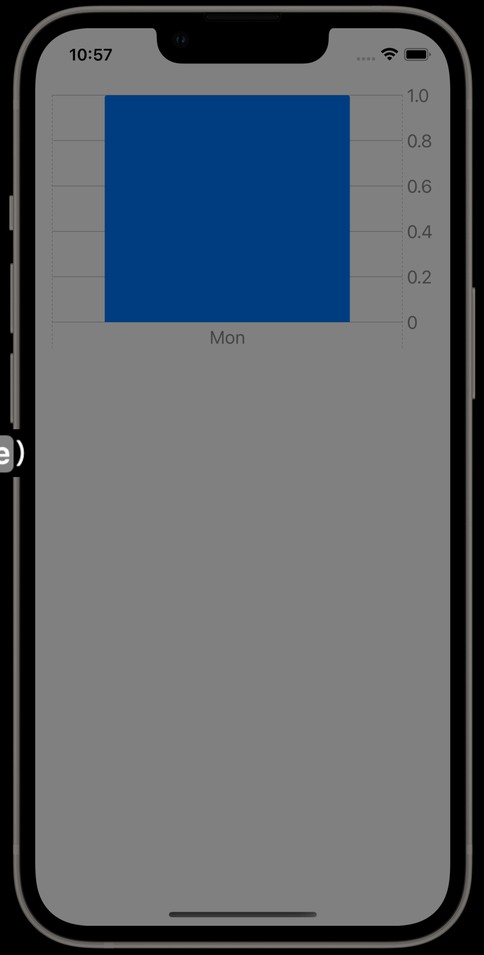

300
200
100
0
Mon
Tue
Wed
Thu
Fri
Sat
Sun
San Francisco
Cupertino

```
import Charts
import SwiftUI


Chart {

}
```

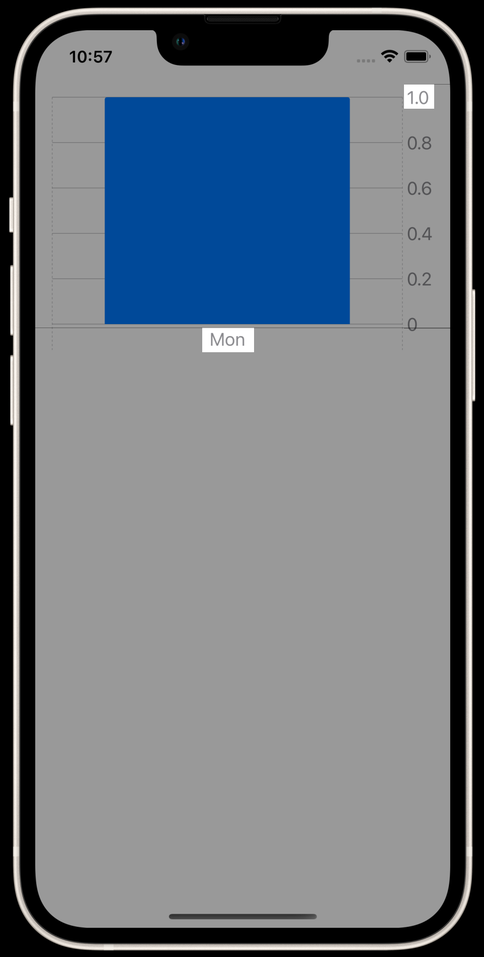
```swift
import Charts
import SwiftUI


Chart {
    BarMark(
        x: .value("Day", "Mon"),
        y: .value("Count", 1)
    )
}
```



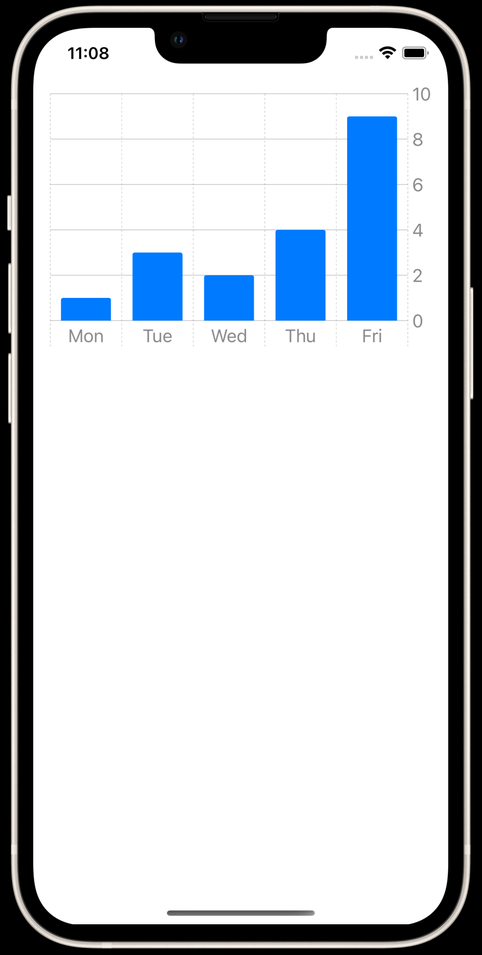
```swift
import Charts
import SwiftUI


Chart {                         .value(labelKey: LocalizedStringKey, value: Plottable)
    BarMark(
        x: .value("Day", "Mon"),
        y: .value("Count", 1)
    )
}
```



```swift
import Charts
import SwiftUI


Chart {
    BarMark(
        x: .value("Day", "Mon"),
        y: .value("Count", 1)
    )
}
```



```
Chart {
    BarMark(
        x: .value("Day", "Mon"),
        y: .value("Count", 1)
    )
    BarMark(
        x: .value("Day", "Tue"),
        y: .value("Count", 3)
    )
    BarMark(
        x: .value("Day", "Wed"),
        y: .value("Count", 2)
    )
    BarMark(
        x: .value("Day", "Thu"),
        y: .value("Count", 4)
    )
    BarMark(
        x: .value("Day", "Fri"),
        y: .value("Count", 9)
    )
}
```



Model

```
struct AlarmSummary: Identifiable {
    let weekday: Date
    let count: Int

    var id: Date { weekday }
}
```

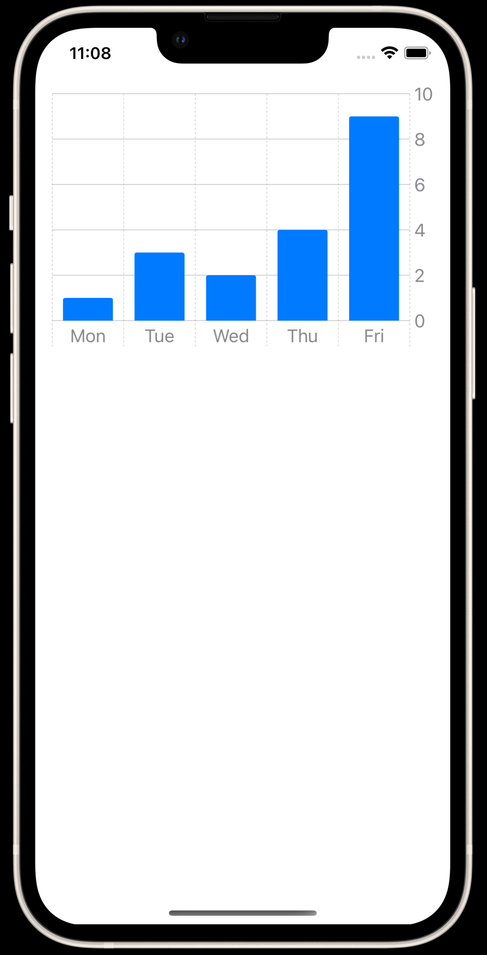
Model

```
struct AlarmSummary: Identifiable {
    let weekday: Date
    let count: Int

    var id: Date { weekday }
}
```

Data

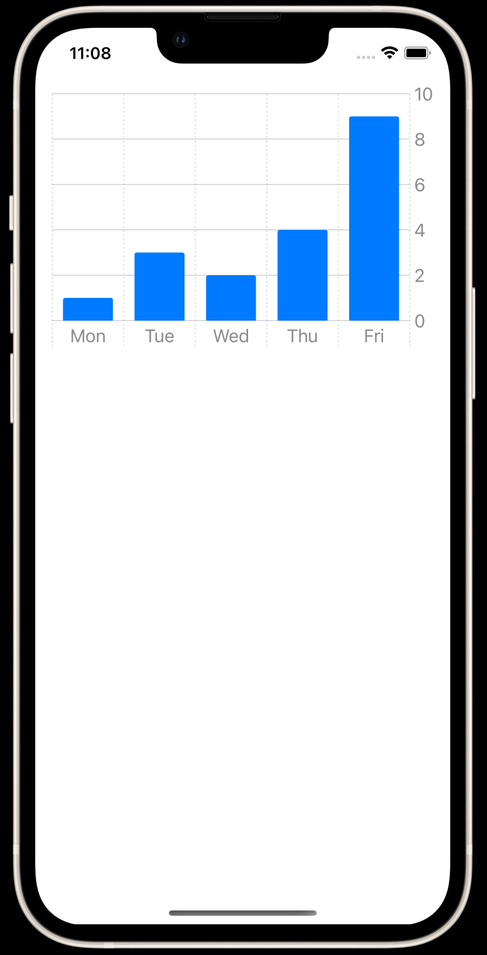
```
let clockAlarmData: [AlarmSummary] = [
    .init(weekday: date(2022, 06, 20), count: 1),
    .init(weekday: date(2022, 06, 21), count: 3),
    .init(weekday: date(2022, 06, 22), count: 2),
    .init(weekday: date(2022, 06, 23), count: 4),
    .init(weekday: date(2022, 06, 24), count: 9),
]
```

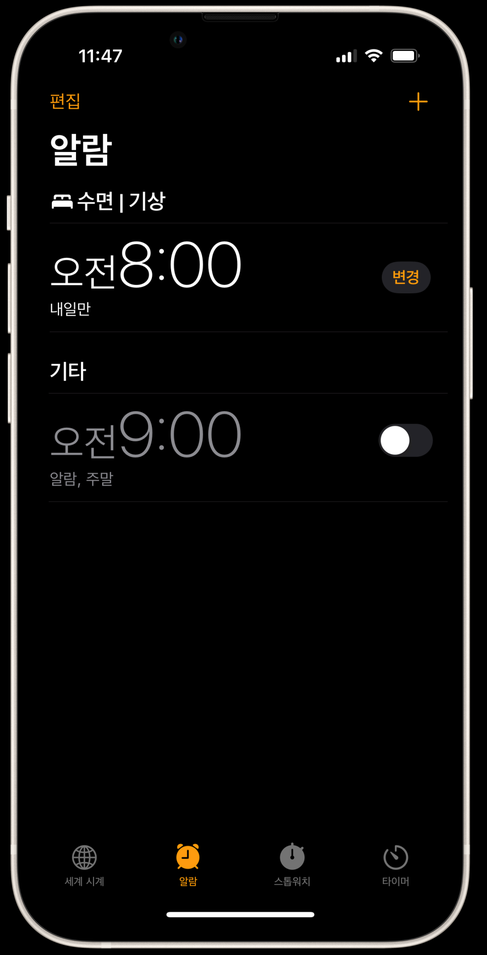
```
Chart {
    ForEach(clockAlarmData) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
    }
}

let clockAlarmData: [AlarmSummary] = [
    .init(weekday: date(2022, 06, 20), count: 1),
    .init(weekday: date(2022, 06, 21), count: 3),
    .init(weekday: date(2022, 06, 22), count: 2),
    .init(weekday: date(2022, 06, 23), count: 4),
    .init(weekday: date(2022, 06, 24), count: 9),
]
```



```
Chart(clockAlarmData) { element in
    BarMark(
        x: .value("Day", element.weekday, unit: .day),
        y: .value("Count", element.count)
    )
}
```

```
let clockAlarmData: [AlarmSummary] = [
    .init(weekday: date(2022, 06, 20), count: 1),
    .init(weekday: date(2022, 06, 21), count: 3),
    .init(weekday: date(2022, 06, 22), count: 2),
    .init(weekday: date(2022, 06, 23), count: 4),
    .init(weekday: date(2022, 06, 24), count: 9),
]
```

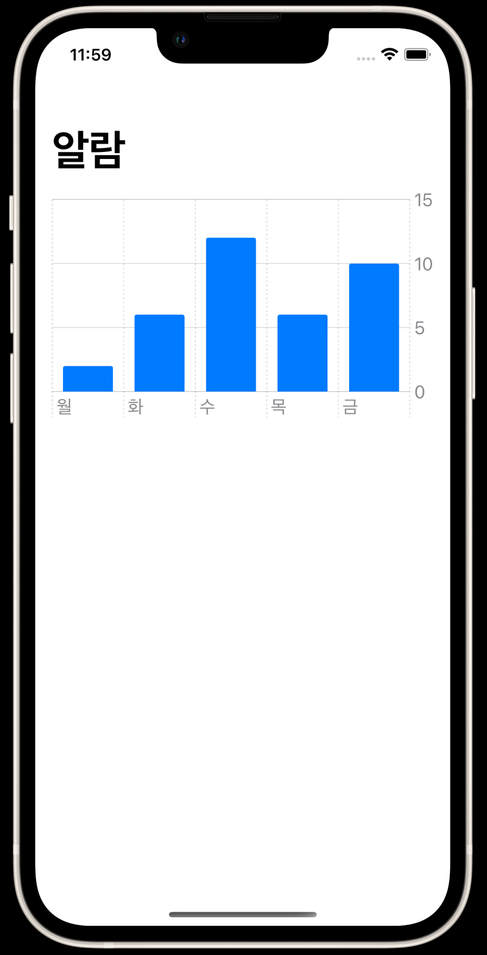



11:47
편집
+
알람
수면 | 기상
오전 8:00
변경
내일만
기타
오전 9:00
알람, 주말
세계 시계
알람
스톱워치
타이머

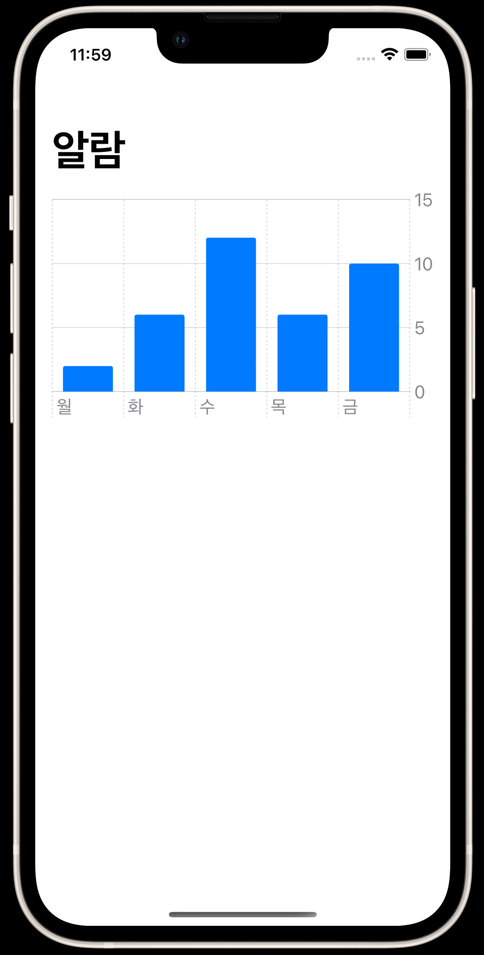
```
let phoneAlarmData: [AlarmSummary] = [
    .init(weekday: date(2022, 06, 20), count: 1),
    .init(weekday: date(2022, 06, 21), count: 3),
    .init(weekday: date(2022, 06, 22), count: 10),
    .init(weekday: date(2022, 06, 23), count: 2),
    .init(weekday: date(2022, 06, 24), count: 1),
]
```

```
let phoneAlarmData: [AlarmSummary] = [
    .init(weekday: date(2022, 06, 20), count: 1),
    .init(weekday: date(2022, 06, 21), count: 3),
    .init(weekday: date(2022, 06, 22), count: 10),
    .init(weekday: date(2022, 06, 23), count: 2),
    .init(weekday: date(2022, 06, 24), count: 1),
]


let alarmData: [AlarmSeries] = [
    .init(device: "Clock", alarms: clockAlarmData),
    .init(device: "Phone", alarms: phoneAlarmData)
]
```

```
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
    }
}

let clockAlarmData: [AlarmSummary] = [
    .init(weekday: date(2022, 06, 20), count: 1),
    .init(weekday: date(2022, 06, 21), count: 3),
    .init(weekday: date(2022, 06, 22), count: 2),
    .init(weekday: date(2022, 06, 23), count: 4),
    .init(weekday: date(2022, 06, 24), count: 9),
]
```

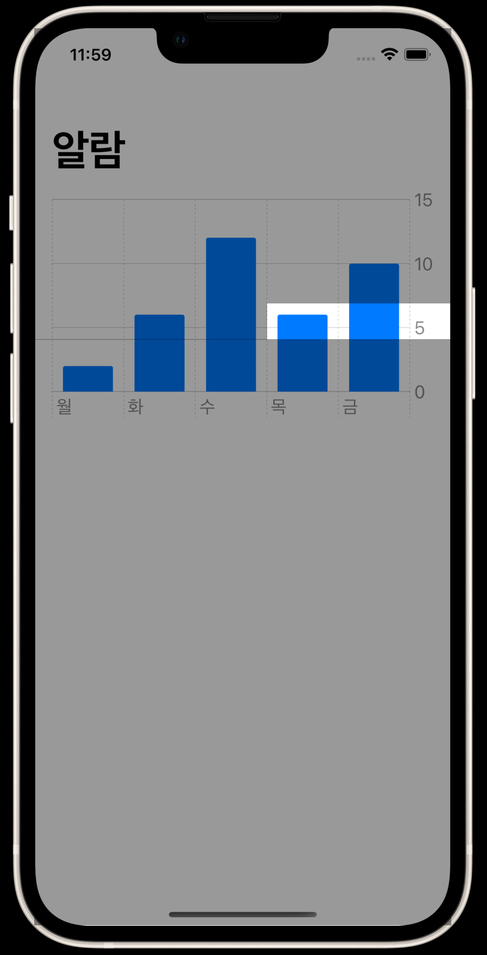


```swift
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
    }
}


let alarmData: [AlarmSeries] = [
    .init(device: "Clock", alarms: clockAlarmData),
    .init(device: "Phone", alarms: phoneAlarmData)
]
```

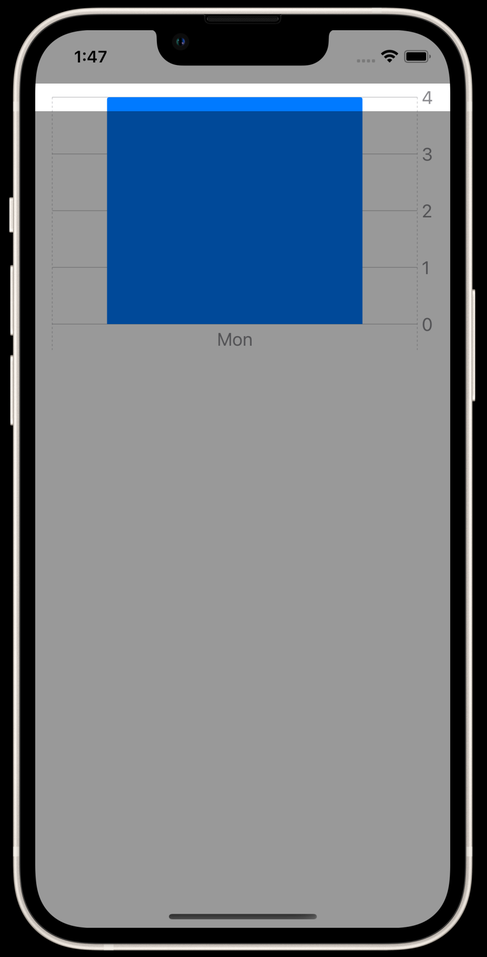


```swift
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
    }
}
```

목요일 clockAlarmData: .init(weekday: date(2022, 06, 23), count: 4),
목요일 phoneAlarmData: .init(weekday: date(2022, 06, 23), count: 2),

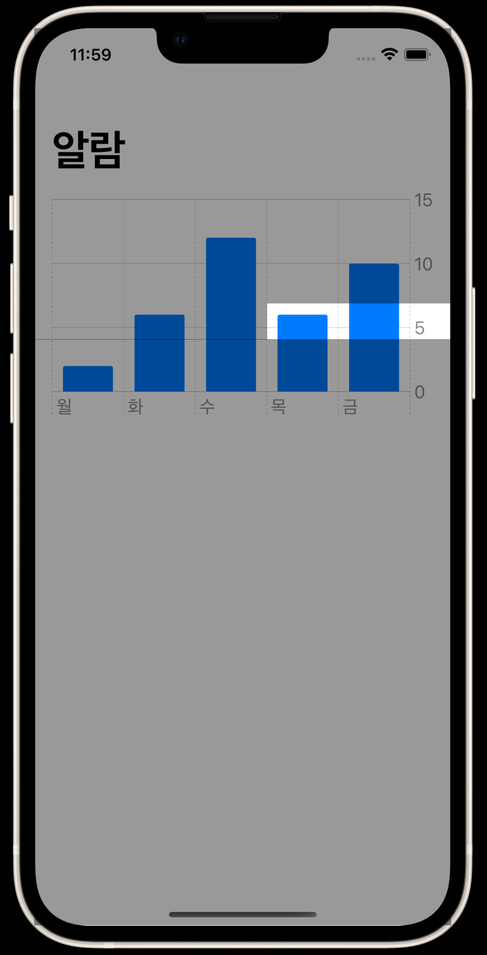


```
Chart {
    BarMark(
        x: .value("Day", "Mon"),
        y: .value("Count", 1)
    )
    BarMark(
        x: .value("Day", "Mon"),
        y: .value("Count", 3)
    )
}
```

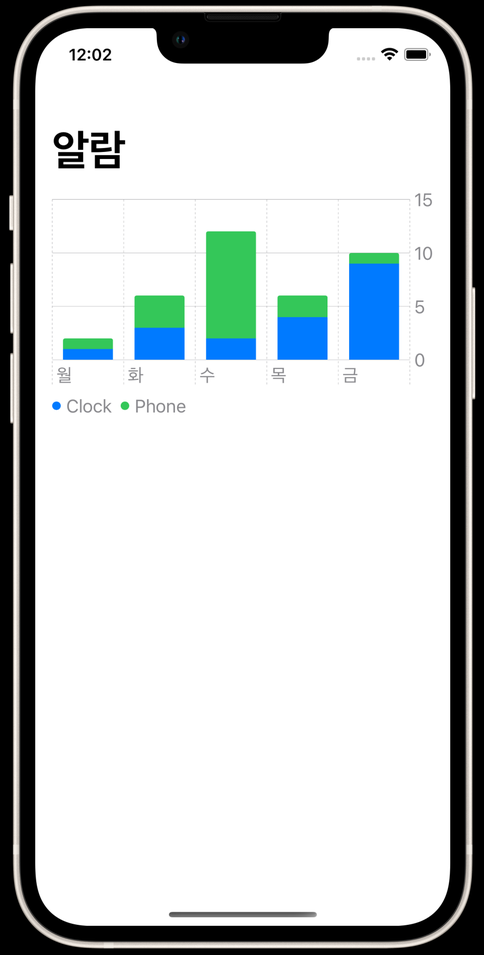


```swift
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
    }
}
```

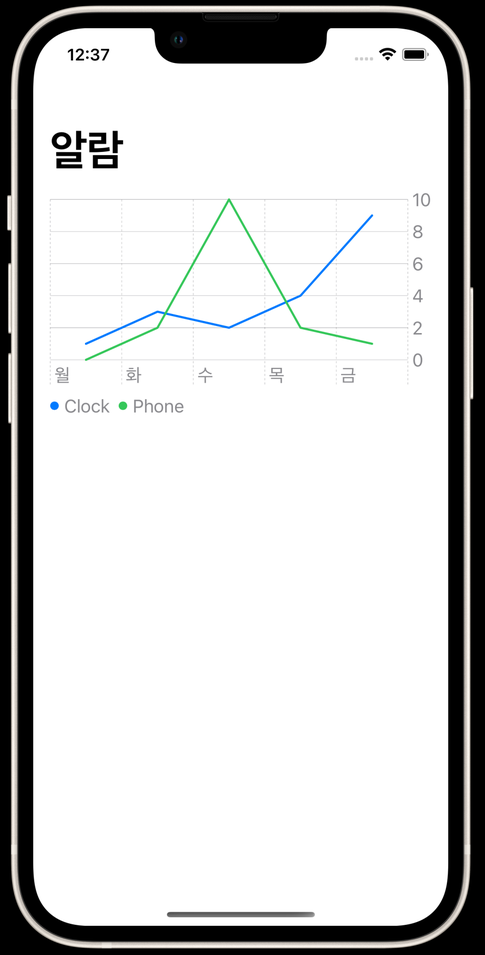
목요일 clockAlarmData: .init(weekday: date(2022, 06, 23), count: 4),
목요일 phoneAlarmData: .init(weekday: date(2022, 06, 23), count: 2),



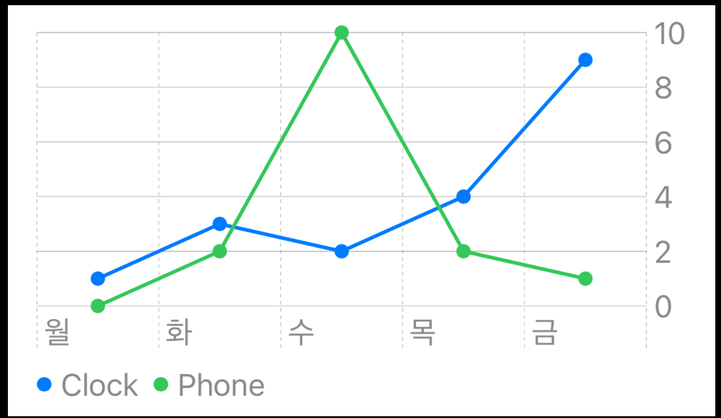
```
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
        .foregroundStyle(by: .value("Device", series.device))
    }
}

let alarmData: [AlarmSeries] = [
    .init(device: "Clock", alarms: clockAlarmData),
    .init(device: "Phone", alarms: phoneAlarmData)
]
```



```
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        LineMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
        .foregroundStyle(by: .value("Device", series.device))
    }
}

let alarmData: [AlarmSeries] = [
    .init(device: "Clock", alarms: clockAlarmData),
    .init(device: "Phone", alarms: phoneAlarmData)
]
```



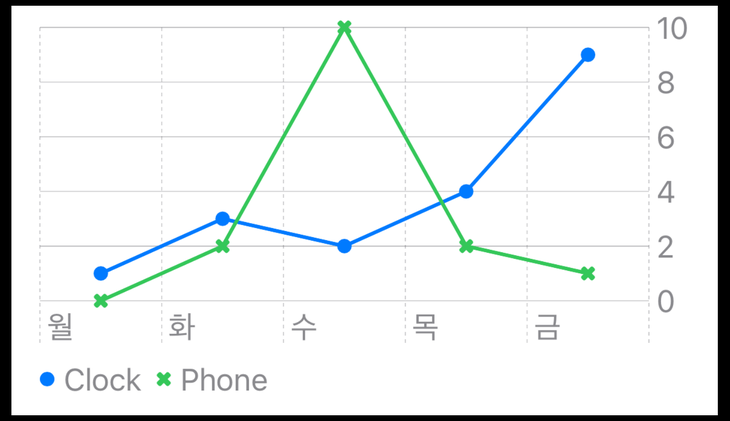
```swift
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        LineMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
        .foregroundStyle(by: .value("Device", series.device))

        PointMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
        .foregroundStyle(by: .value("Device", series.device))
    }
```



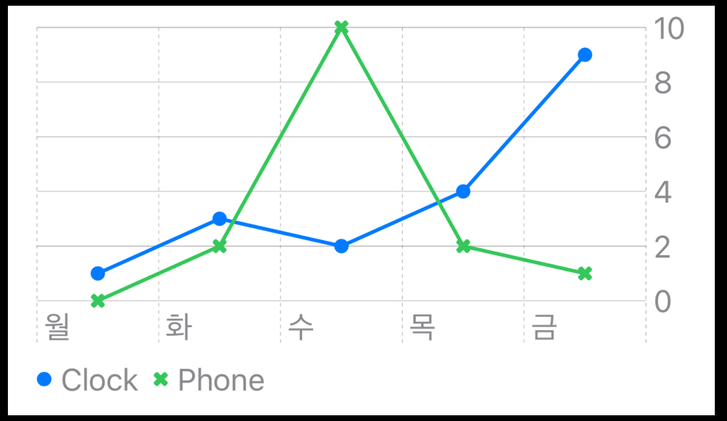

```swift
let alarmData: [AlarmSeries] = [
    .init(device: "Clock", alarms: clockAlarmData),
    .init(device: "Phone", alarms: phoneAlarmData)
]
```

```swift
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        LineMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
        .foregroundStyle(by: .value("Device", series.device))

        PointMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
        .foregroundStyle(by: .value("Device", series.device))
        .symbol(by: .value("Device", series.device))
    }
```

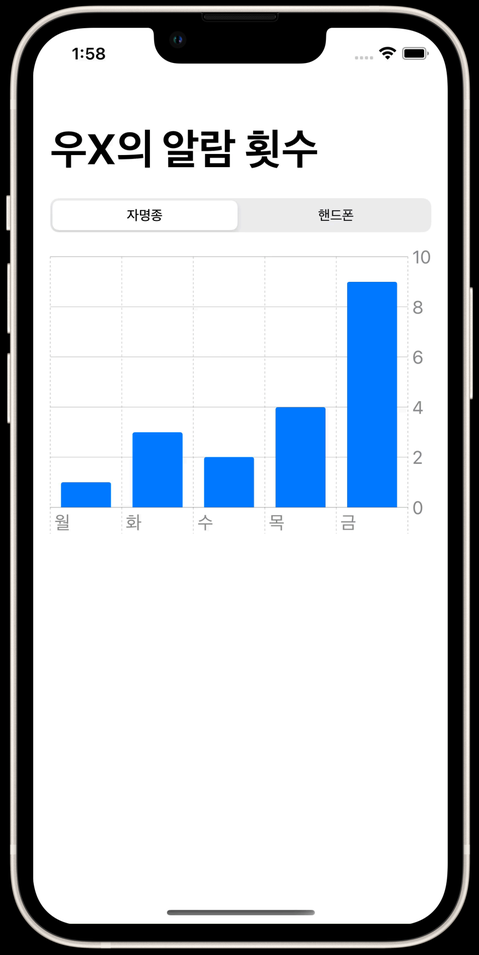
```swift
let alarmData: [AlarmSeries] = [
    .init(device: "Clock", alarms: clockAlarmData),
    .init(device: "Phone", alarms: phoneAlarmData)
]
```



```
Chart(alarmData) { series in
    ForEach(series.alarms) { element in
        LineMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
        .foregroundStyle(by: .value("Device", series.device))
        .symbol(by: .value("Device", series.device))
    }
}
```




```
let alarmData: [AlarmSeries] = [
    .init(device: "Clock", alarms: clockAlarmData),
    .init(device: "Phone", alarms: phoneAlarmData)
]
```


1:58
우X의 알람 횟수
자명종
핸드폰
10
8
6
4
2
0
월
화
수
목
금

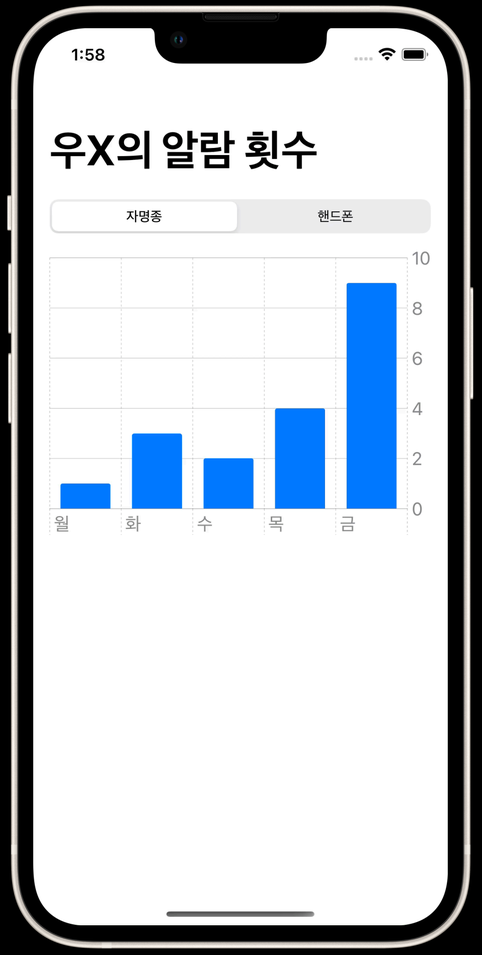
```
enum Device {
    case clock
    case phone
}
```

```swift
enum Device {
    case clock
    case phone
}

@State private var device: Device = .clock

Picker("Device", selection: $device) {
    Text("자명종").tag(Device.clock)
    Text("핸드폰").tag(Device.phone)
}
.pickerStyle(.segmented)
```

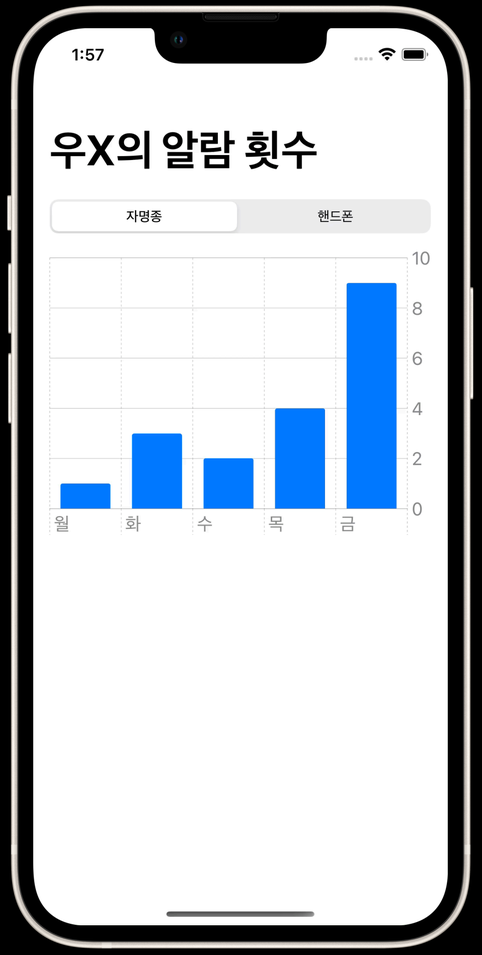
```swift
enum Device {
    case clock
    case phone
}

@State private var device: Device = .clock

Picker("Device", selection: $device) {
    Text("자명종").tag(Device.clock)
    Text("핸드폰").tag(Device.phone)
}
.pickerStyle(.segmented)

var data: [AlarmSummary] {
    switch device {
    case .clock:
        return clockAlarmData
    case .phone:
        return phoneAlarmData
    }
}
```

```swift
@State private var device: Device = .clock

Picker("Device", selection: $device) {
    Text("자명종").tag(Device.clock)
    Text("핸드폰").tag(Device.phone)
}
.pickerStyle(.segmented)

var data: [AlarmSummary] {
    switch device {
    case .clock:
        return clockAlarmData
    case .phone:
        return phoneAlarmData
    }
}

Chart(data) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
}
```

```
Picker("Device", selection: $device) {
    Text("Clock").tag(Device.clock)
    Text("Phone").tag(Device.phone)
}
.pickerStyle(.segmented)

Chart(data) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
}
```



```
Picker("Device", selection: $device.animation(.easeInOut)) {
    Text("Clock").tag(Device.clock)
    Text("Phone").tag(Device.phone)
}
.pickerStyle(.segmented)

Chart(data) { element in
        BarMark(
            x: .value("Day", element.weekday, unit: .day),
            y: .value("Count", element.count)
        )
}
```



Dark Mode / Device Screen Sizes / Dynamic Type /
Voice Over / Audio Graphs / High-Contrast /
Localization / Multi-Platform

Marks

Bar

Point

Line

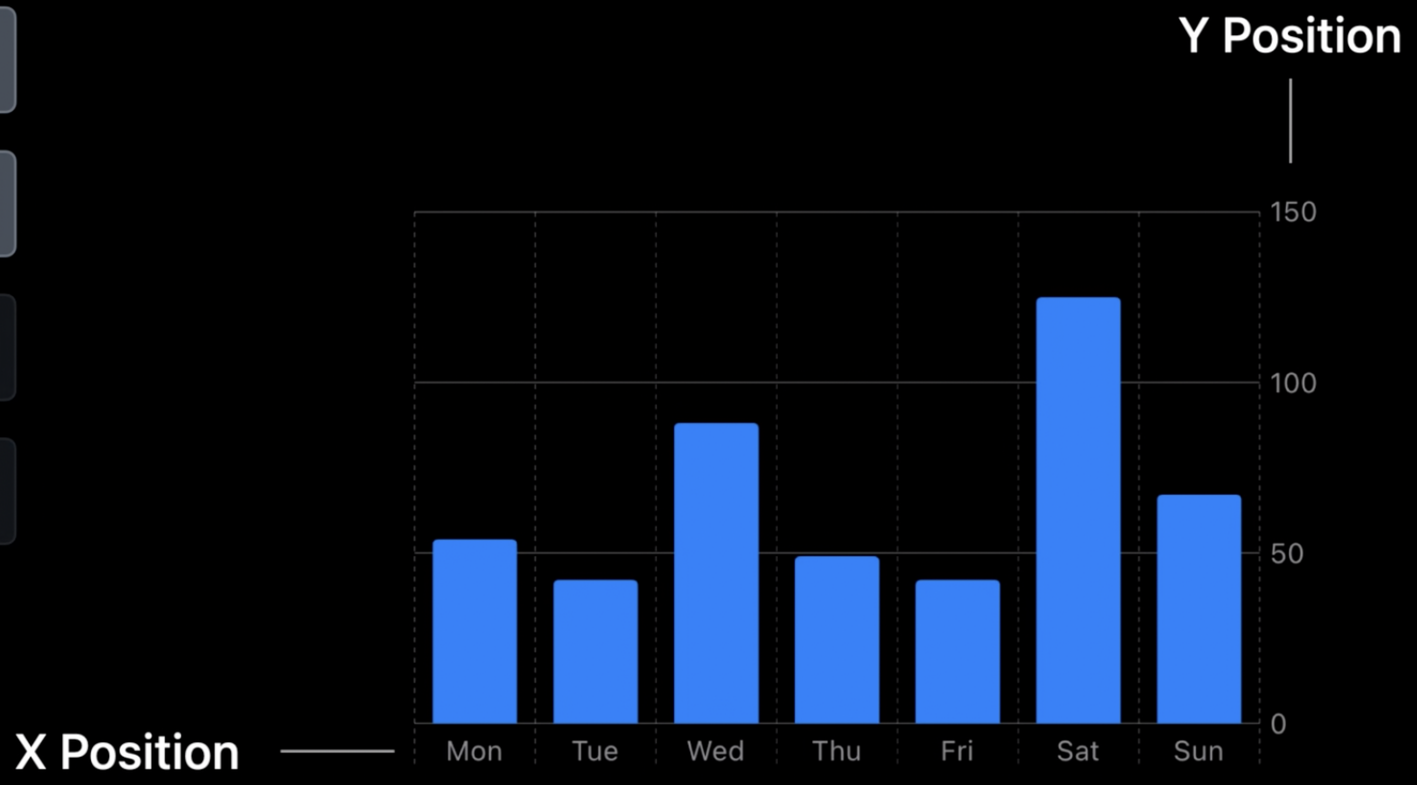
Properties

X Position

Y Position

Foreground Style

Symbol



Marks

Bar

Point

Line

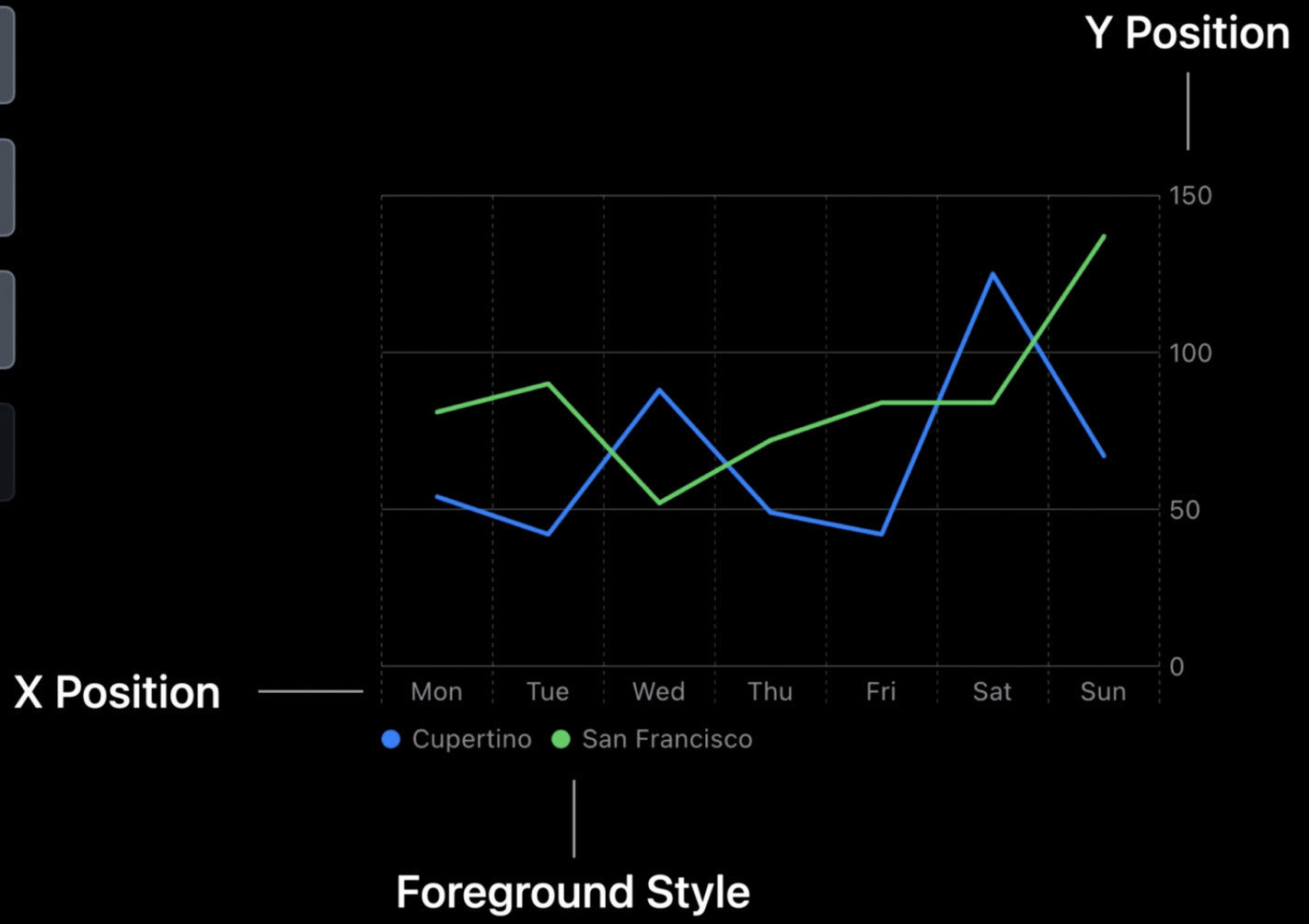
Properties

X Position

Y Position

Foreground Style

Symbol



Marks

Bar

Point

Line

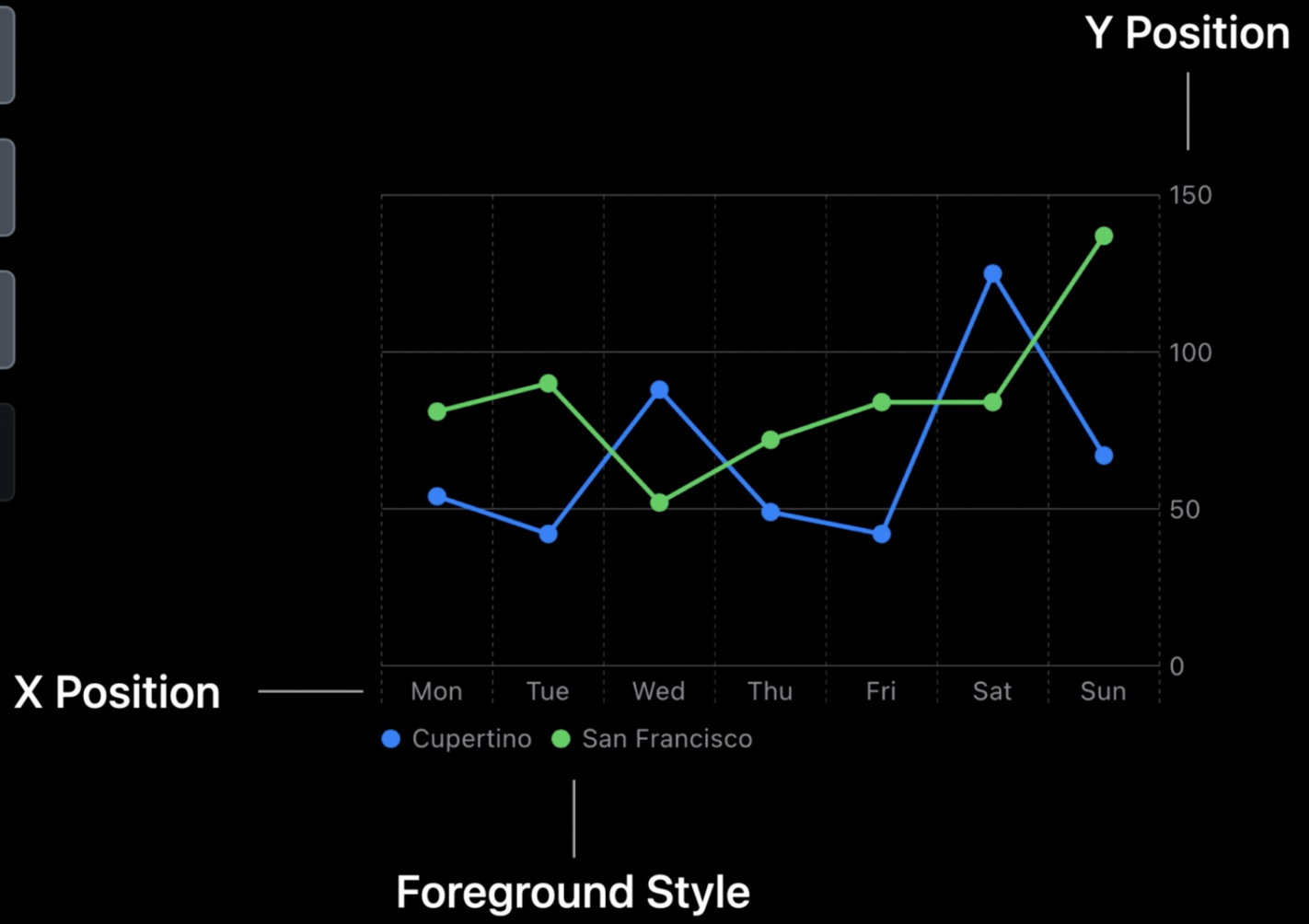
Properties

X Position

Y Position

Foreground Style

Symbol



Marks

Bar

Point

Line

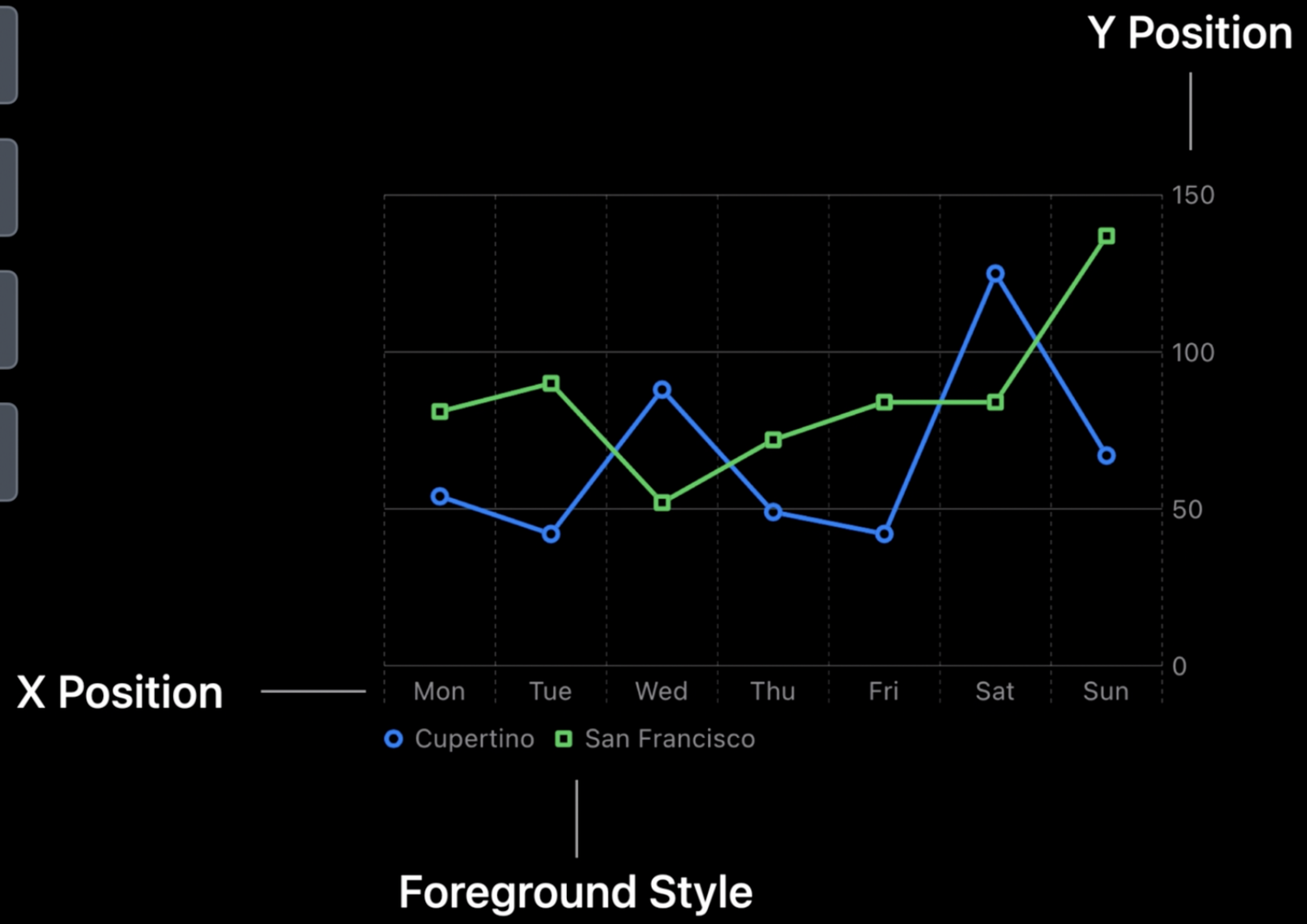
Properties

X Position

Y Position

Foreground Style

Symbol

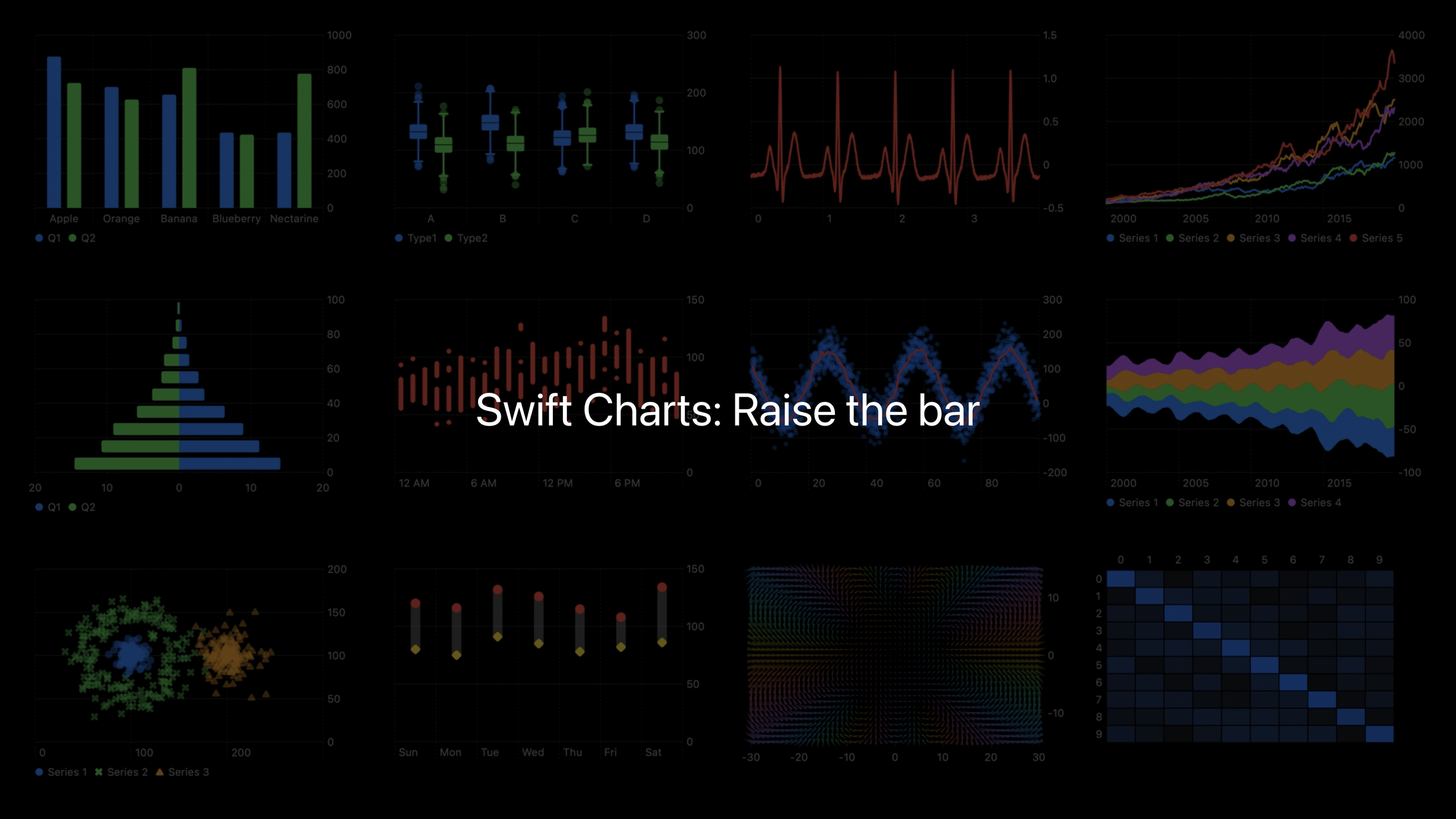


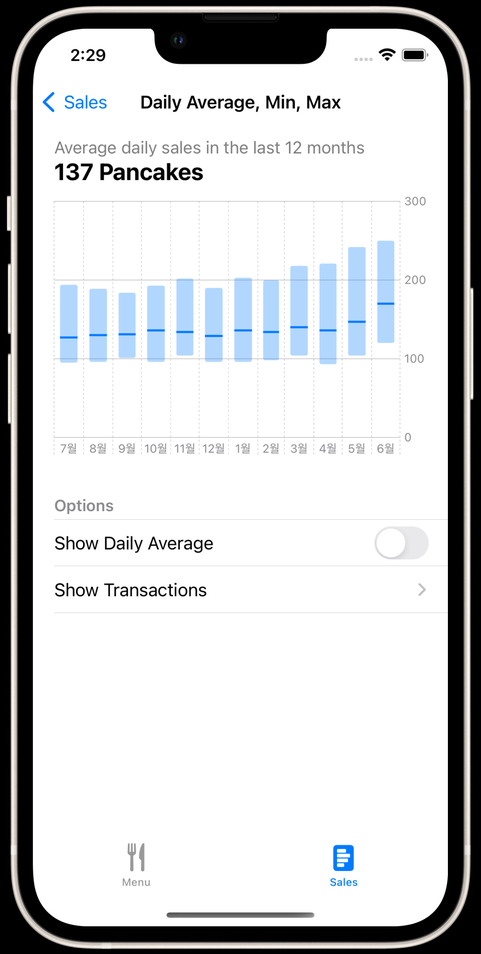
Marks

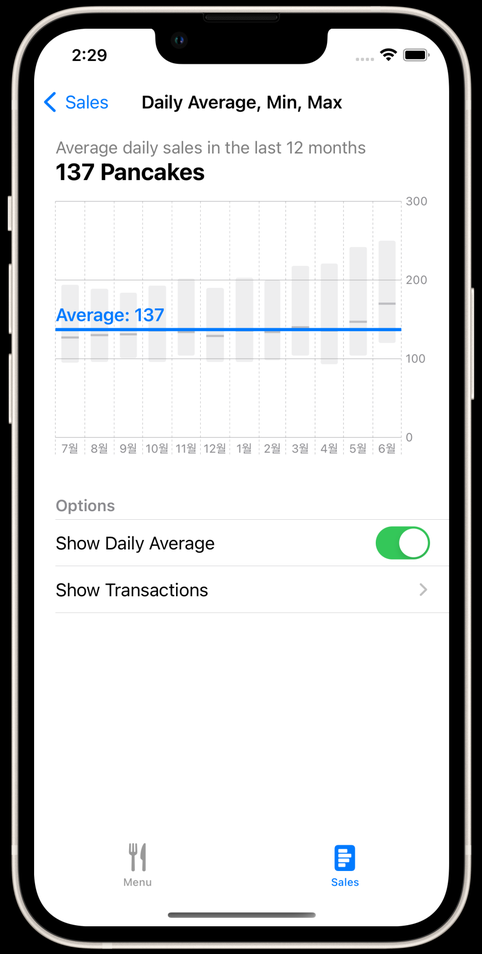
- Bar
- Point
- Line
- Area
- Rule
- Rectangle

Properties

- X Position
- Y Position
- Foreground Style
- Symbol
- Symbol Size
- Line Style


Swift Charts: Raise the bar
Apple
Orange
Banana
Blueberry
Nectarine
Q1
Q2
A
B
C
D
Type1
Type2
Series 1
Series 2
Series 3
Series 4
Series 5
12 AM
6 AM
12 PM
6 PM
Sun
Mon
Tue
Wed
Thu
Fri
Sat


2:29
Sales
Daily Average, Min, Max
Average daily sales in the last 12 months
137 Pancakes
300
200
100
0
7월 8월 9월 10월 11월 12월 1월 2월 3월 4월 5월 6월
Options
Show Daily Average
Show Transactions
Menu
Sales


2:29
Sales
Daily Average, Min, Max
Average daily sales in the last 12 months
137 Pancakes
300
200
Average: 137
100
0
7월 8월 9월 10월 11월 12월 1월 2월 3월 4월 5월 6월
Options
Show Daily Average
Show Transactions
Menu
Sales


2:25
◀ SwiftChart
Sales
Interactive Lollipop
Total Sales
4,593 Pancakes
300
200
100
0
5월 08일
5월 15일
5월 22일
5월 29일
...
Options
Show Transactions
Menu
Sales