Докладчик

Антон Романчук
iOS Developer

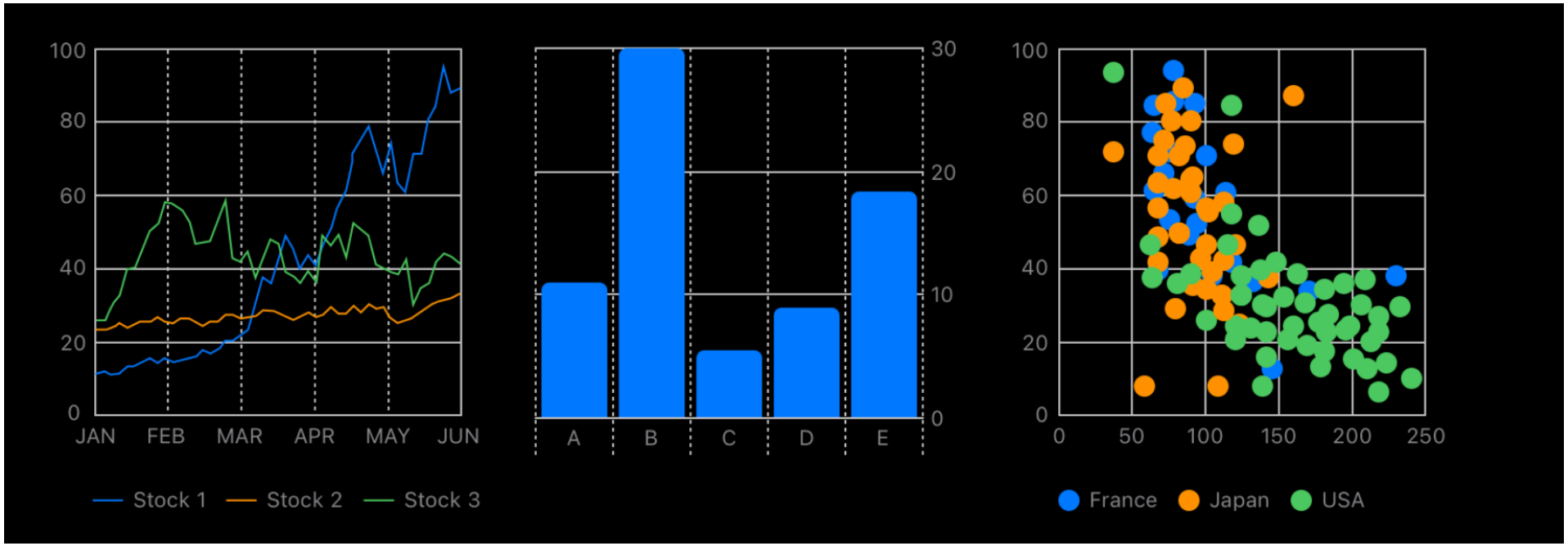
Swift Charts

Возможности, реализация и обновления с WWDC 2023

Содержание

1. Что такое Swift Charts?

На WWDC 2022 Apple представила новую библиотеку Swift Chart, которая позволяет создавать графики на SwiftUI



1. Что такое Swift Charts?

Можем ли мы использовать Swift Charts сейчас?

- iOS 16.0+
- iPadOS 16.0+
- macOS 13.0+
  - Mac Catalyst 16.0+
- tvOS 16.0+
- watchOS 9.0+
- visionOS 1.0+

1.1 SwiftUI

SwiftUI

```
var body: some View {
    Button("Title") {
        print("Action")
    }
}
```

UIKit

```
let button = UIButton(type: .custom)
button.frame = CGRect(x: 0, y: 0, width: 100, height: 50)
button.setTitle("Title", for: .normal)
button.target(forAction: #selector(someFunciton), withSender: self)
view.addSubview(button)
```

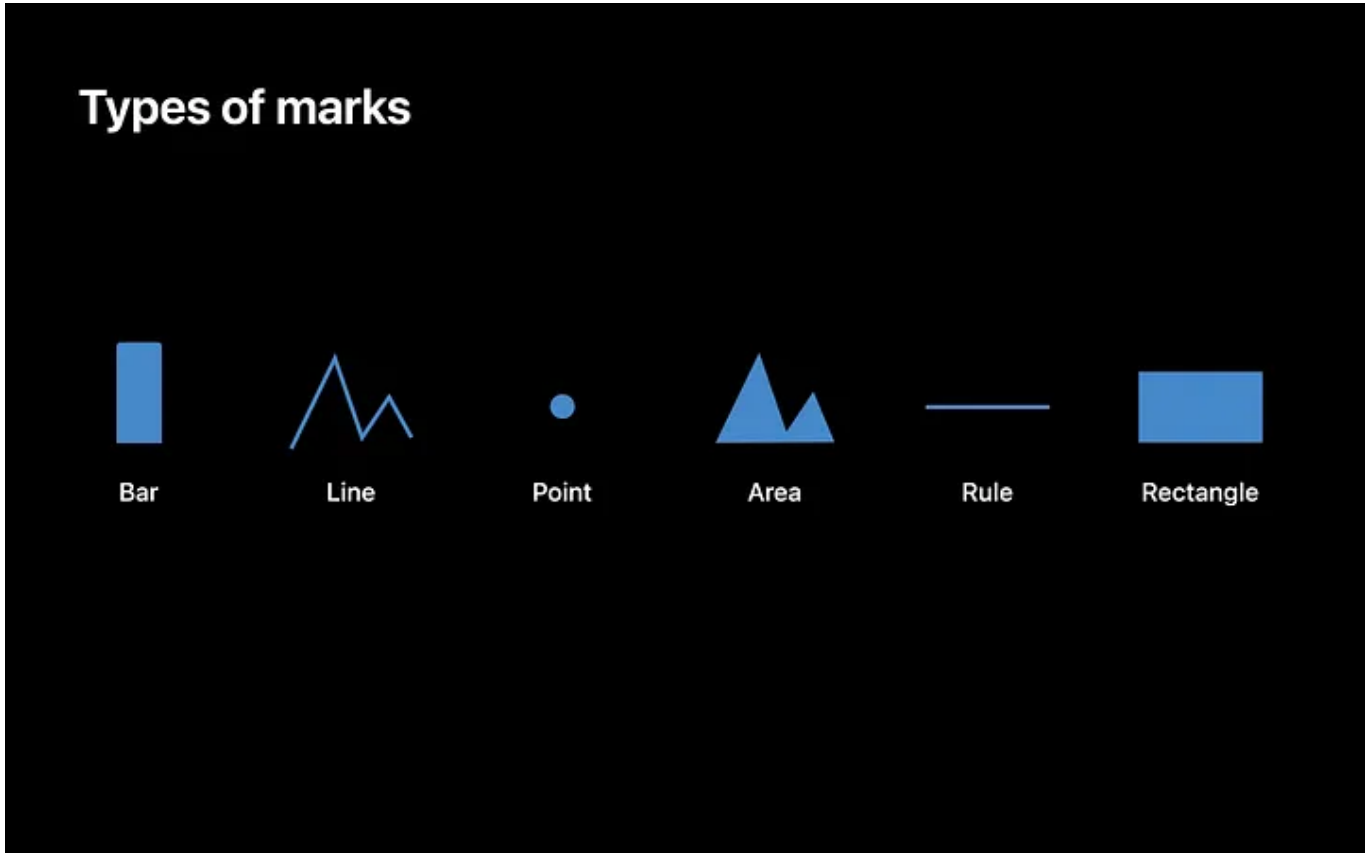
1.2 Компоненты библиотеки Swift Charts

Chart

```
Chart {

}
```

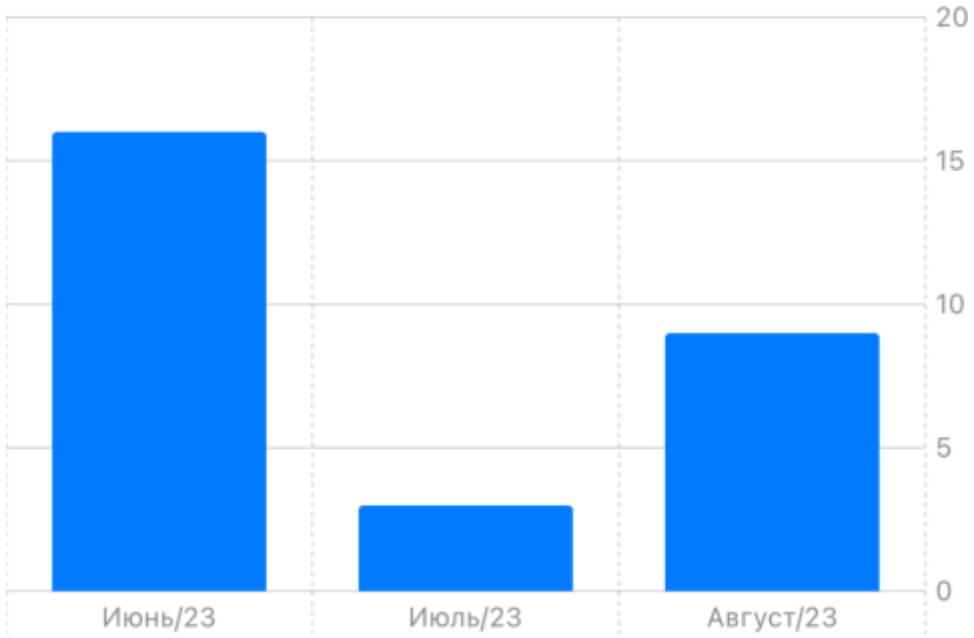
1.2 Компоненты библиотеки Swift Charts

Mark



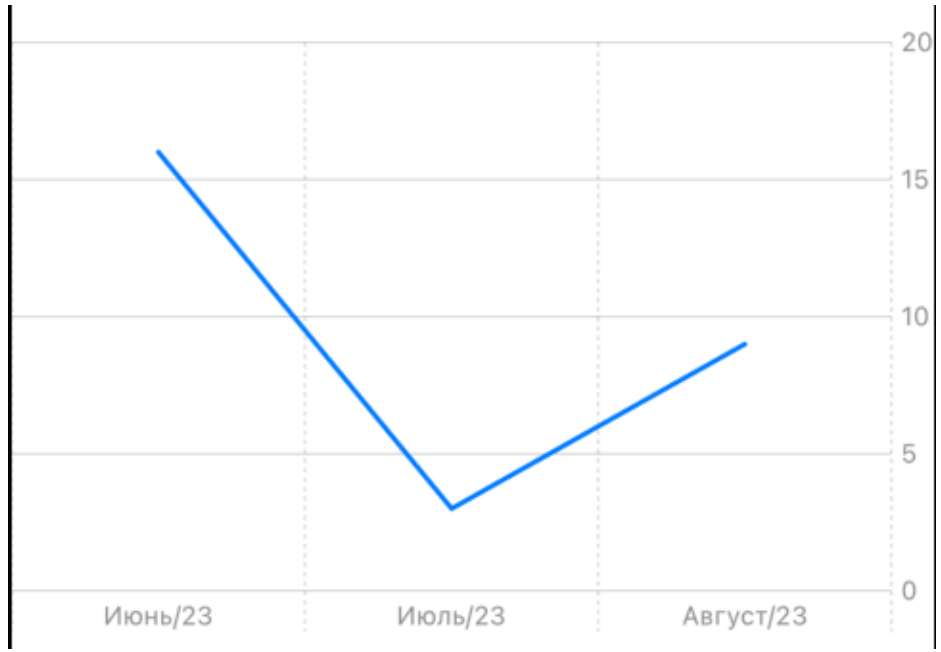
1.2 Компоненты библиотеки Swift Charts

BarMark



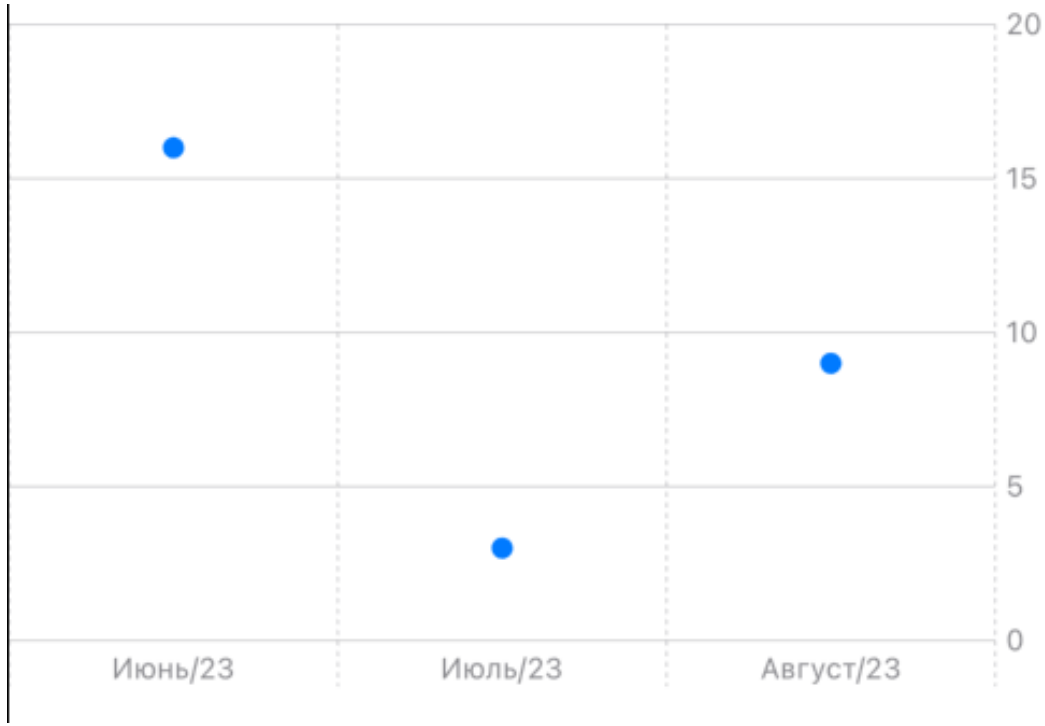
1.2 Компоненты библиотеки Swift Charts

LineMark



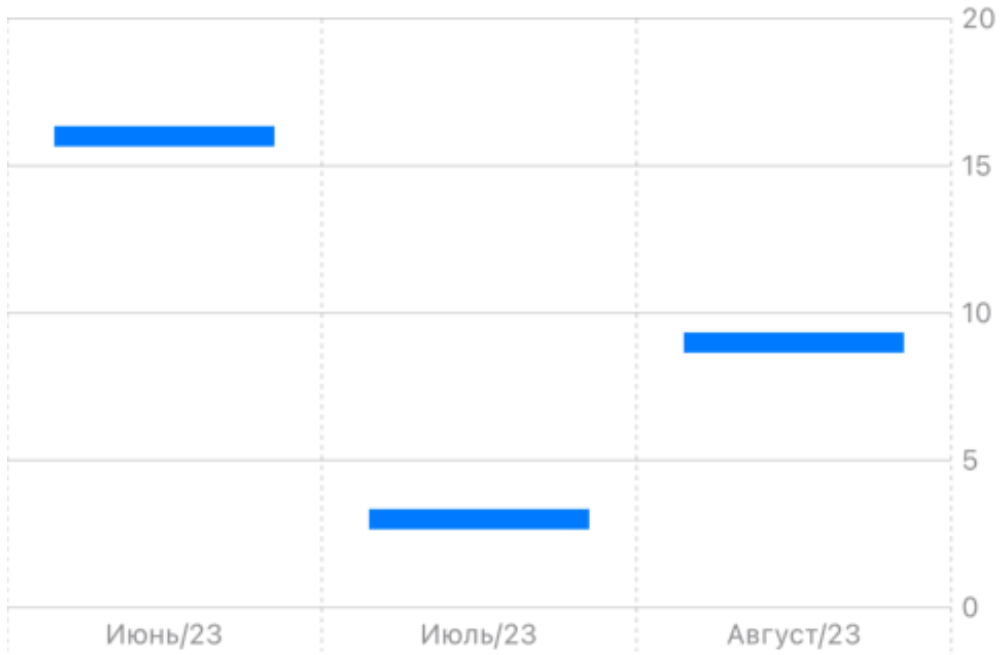
1.2 Компоненты библиотеки Swift Charts

PointMark



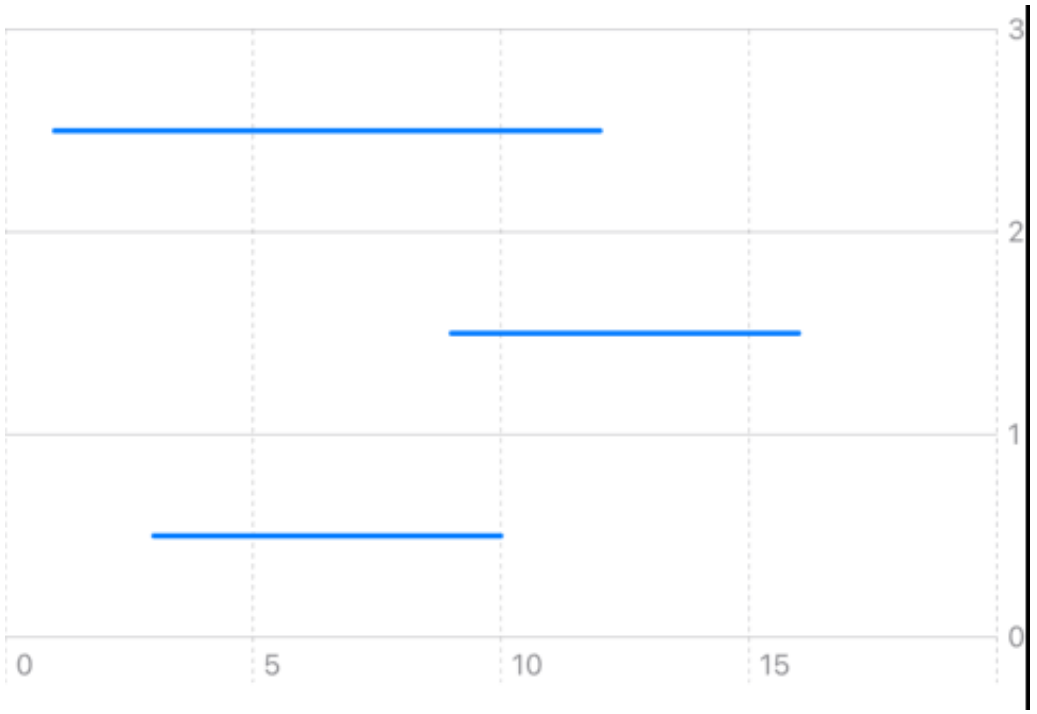
1.2 Компоненты библиотеки Swift Charts

RectangleMark



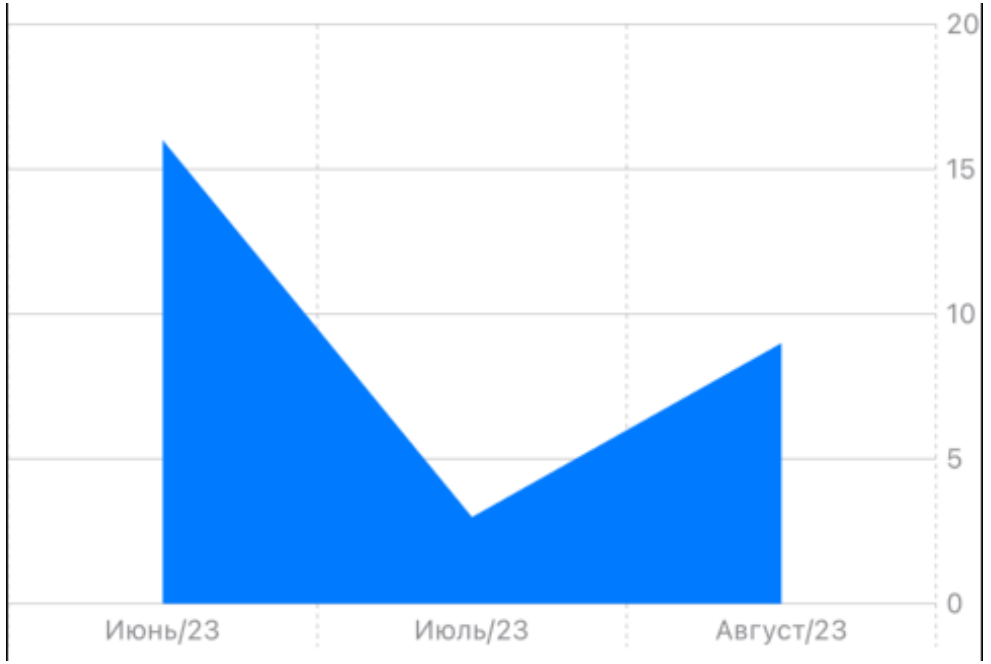
1.2 Компоненты библиотеки Swift Charts

RuleMark



1.2 Компоненты библиотеки Swift Charts

AreaMark



2. Реализация Swift Charts

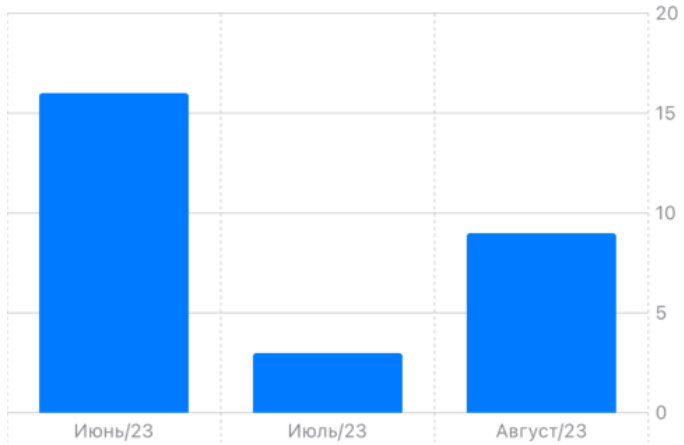


График с одним типом Mark

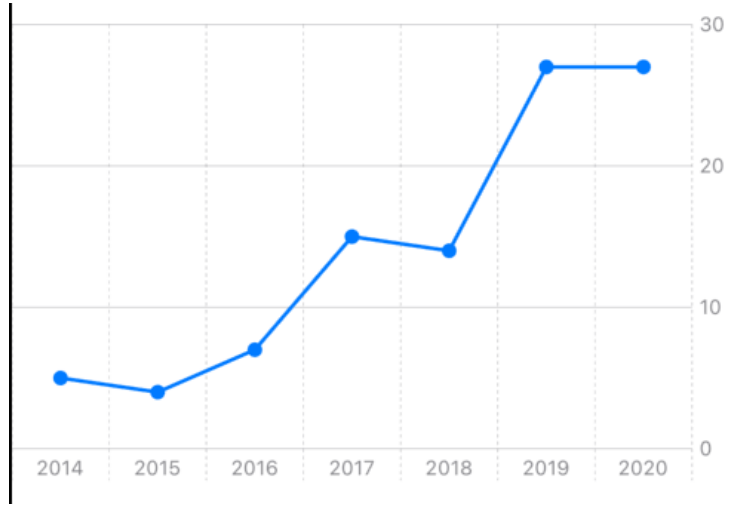


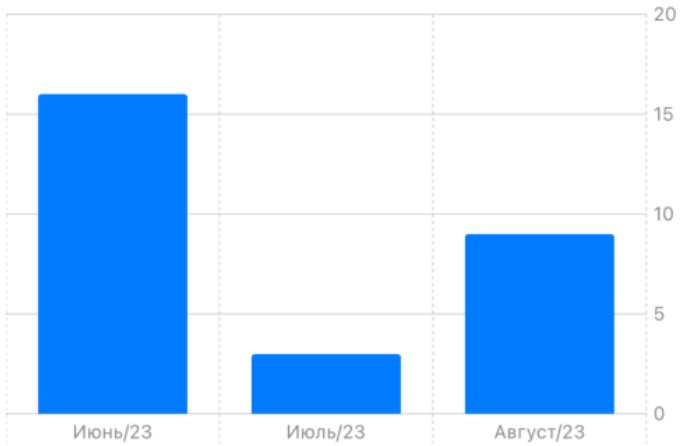
График с LineMark + PointMark

2. Реализация Swift Charts

Типы данных

2.1 Реализация однотипных графиков

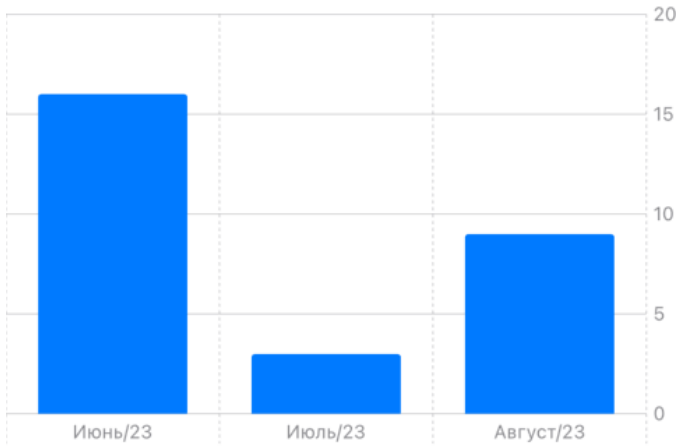
BarMark




```swift
import SwiftUI
import Charts

struct BarMarkExample: View {
    var body: some View {
        Chart {
            BarMark(
                x: .value("Month", "Июнь/23"),
                y: .value("Value", 16)
            )
            BarMark(
                x: .value("Month", "Июль/23"),
                y: .value("Value", 3)
            )
            BarMark(
                x: .value("Month", "Август/23"),
                y: .value("Value", 9)
            )
        }
        .frame(height: 250)
    }
}
```

2.1 Реализация однотипных графиков

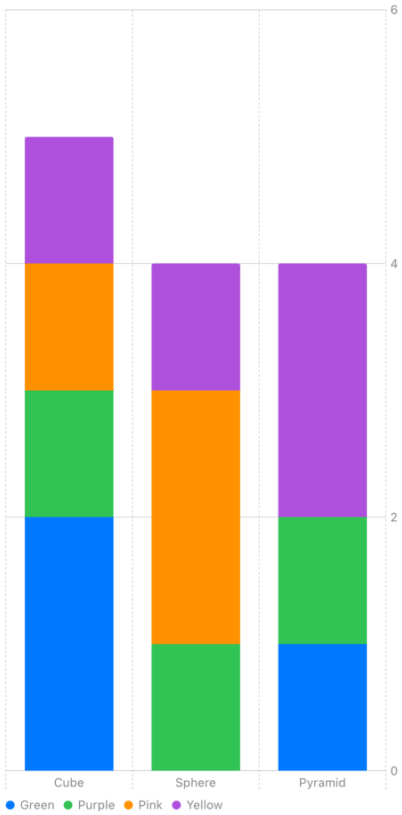
BarMark




```swift
import SwiftUI
import Charts

struct CustomBarMarkChart: View {
    let monthHoursArray: [MonthHours] = [
        .init(month: "Июнь/23", hours: 16),
        .init(month: "Июль/23", hours: 3),
        .init(month: "Август/23", hours: 9)
    ]

    var body: some View {
        Chart(monthHoursArray) {
            BarMark(
                x: .value("Month", $0.month),
                y: .value("Value", $0.hours)
            )
        }
        .frame(height: 250)
    }
}

// MARK: - Data

extension CustomBarMarkChart {
    struct MonthHours: Identifiable {
        let month: String
        let hours: Int
        let id: UUID = .init()
    }
}
```

2.1 Реализация однотипных графиков

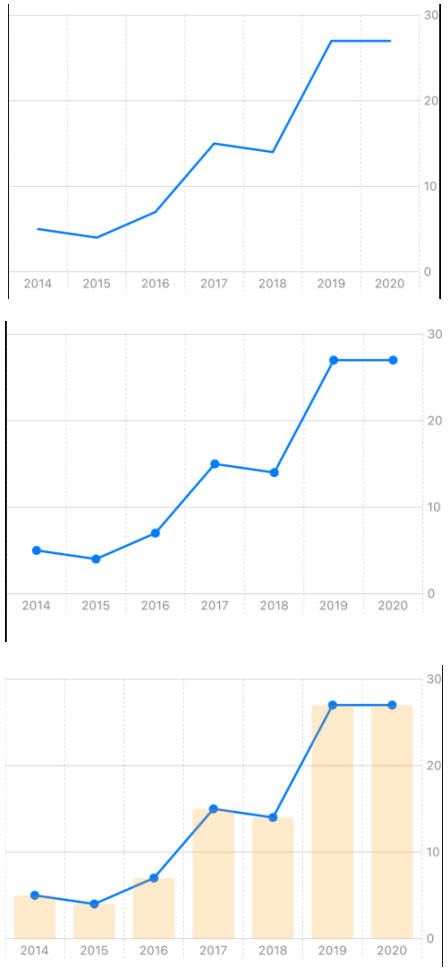
BarMark




```swift
import SwiftUI
import Charts

struct DataBarMarkChart: View {
    let shapes: [Shape] = [
        .init(color: "Green", type: "Cube", count: 2),
        .init(color: "Green", type: "Sphere", count: 0),
        .init(color: "Green", type: "Pyramid", count: 1),
        .init(color: "Purple", type: "Cube", count: 1),
        .init(color: "Purple", type: "Sphere", count: 1),
        .init(color: "Purple", type: "Pyramid", count: 1),
        .init(color: "Pink", type: "Cube", count: 1),
        .init(color: "Pink", type: "Sphere", count: 2),
        .init(color: "Pink", type: "Pyramid", count: 0),
        .init(color: "Yellow", type: "Cube", count: 1),
        .init(color: "Yellow", type: "Sphere", count: 1),
        .init(color: "Yellow", type: "Pyramid", count: 2)
    ]

    var body: some View {
        Chart(shapes) {
            BarMark(
                x: .value("Shape Type", $0.type),
                y: .value("Total Count", $0.count)
            )
            .foregroundStyle(by: .value("Shape Color", $0.color))
        }
    }
}

// MARK: - Data

extension DataBarMarkChart {
    struct Shape: Identifiable {
        var color: String
        var type: String
        var count: Double
        var id = UUID()
    }
```

2.2 Комбинирование разных типов графиков

LineMark + PointMark + BarMark




```swift
import SwiftUI
import Charts

struct CombinedPointAndLineMarksChart: View {
    private let data: [PriceHistory] = [
        PriceHistory(year: "2014", value: 5),
        PriceHistory(year: "2015", value: 4),
        PriceHistory(year: "2016", value: 7),
        PriceHistory(year: "2017", value: 15),
        PriceHistory(year: "2018", value: 14),
        PriceHistory(year: "2019", value: 27),
        PriceHistory(year: "2020", value: 27)
    ]

    var body: some View {
        Chart(data) {
            LineMark(
                x: .value("Year", $0.year),
                y: .value("Value", $0.value)
            )
            PointMark(
                x: .value("Year", $0.year),
                y: .value("Value", $0.value)
            )
            BarMark(
                x: .value("Year", $0.year),
                y: .value("Value", $0.value)
            )
            .foregroundStyle(.orange.opacity(0.2))
        }
        .frame(height: 250)
    }
}


// MARK: - Data

extension CombinedPointAndLineMarksChart {
    struct PriceHistory: Identifiable {
        var id = UUID()
        var year: String
        var value: Double
    }
}
```

2.3 Возможности кастомизации графиков

BarMark

2.3 Возможности кастомизации графиков

BarMark

.foregroundStyle(Color.red)

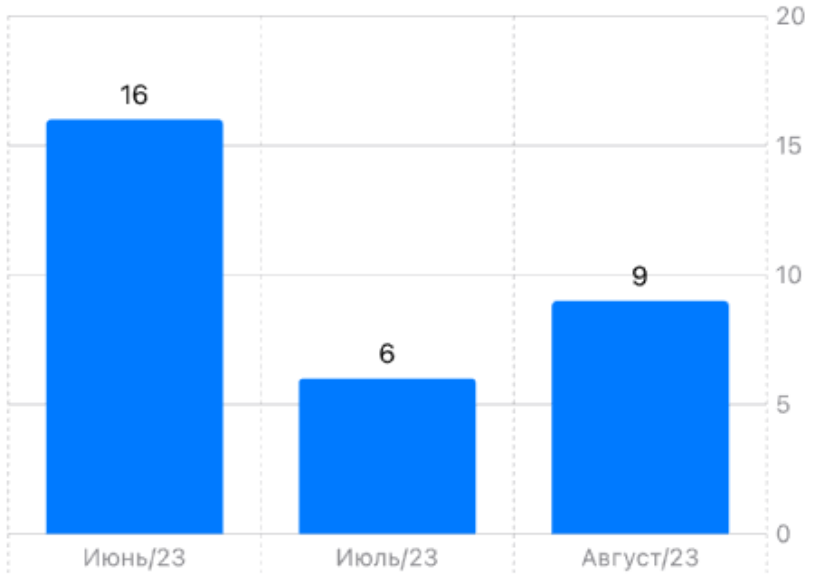
2.3 Возможности кастомизации графиков

BarMark

.clipShape(RoundedRectangle(cornerRadius: 32))

2.3 Возможности кастомизации графиков

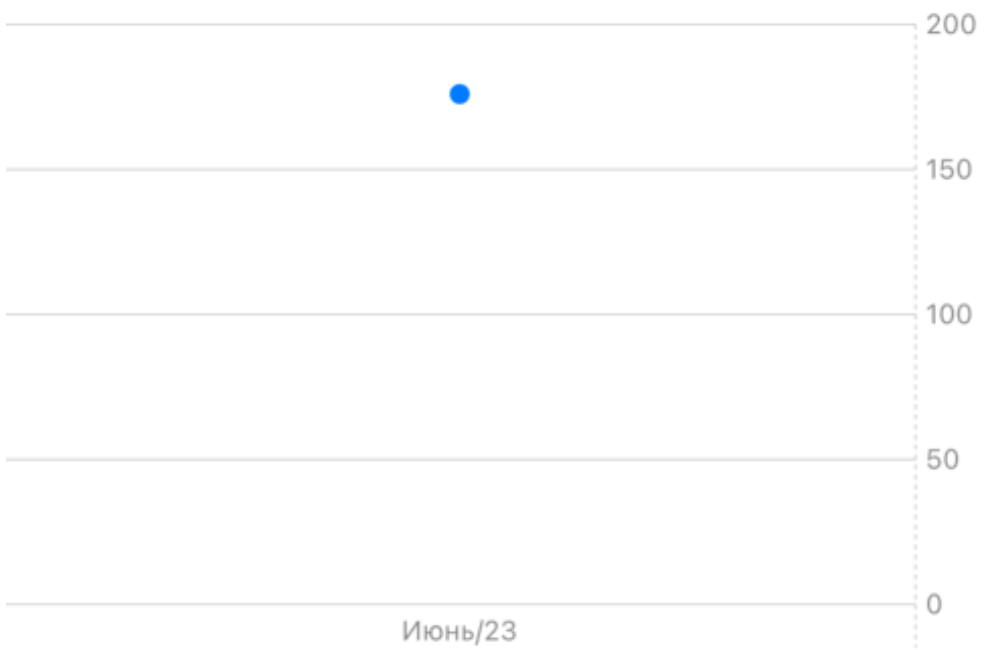
BarMark



.annotation { Text("\(hoursValue)") }

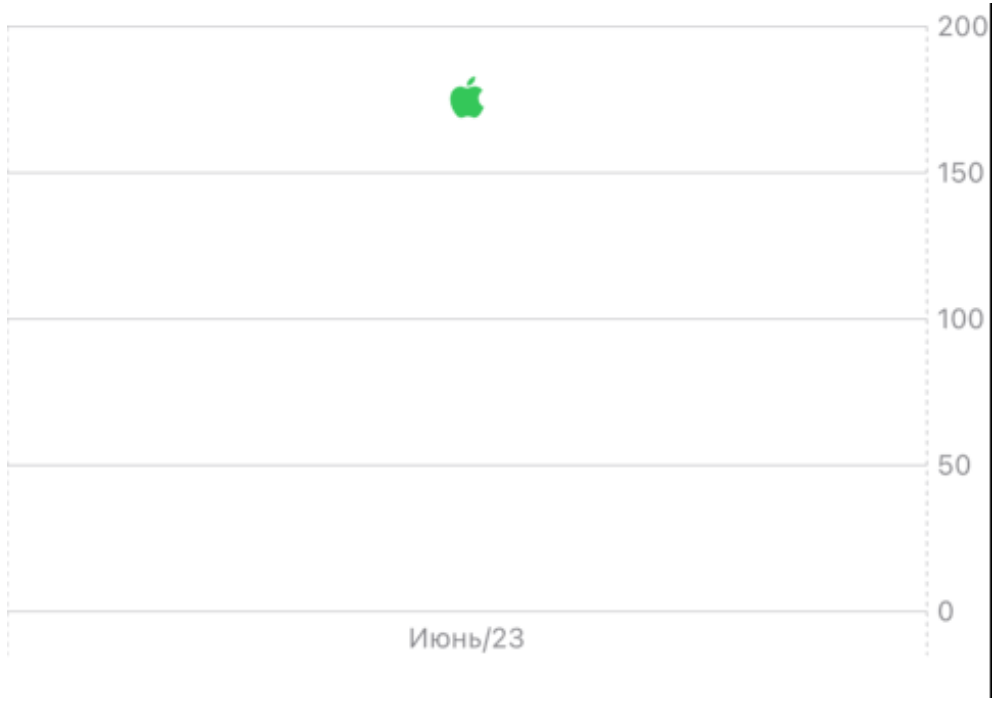
2.3 Возможности кастомизации графиков

PointMark



2.3 Возможности кастомизации графиков

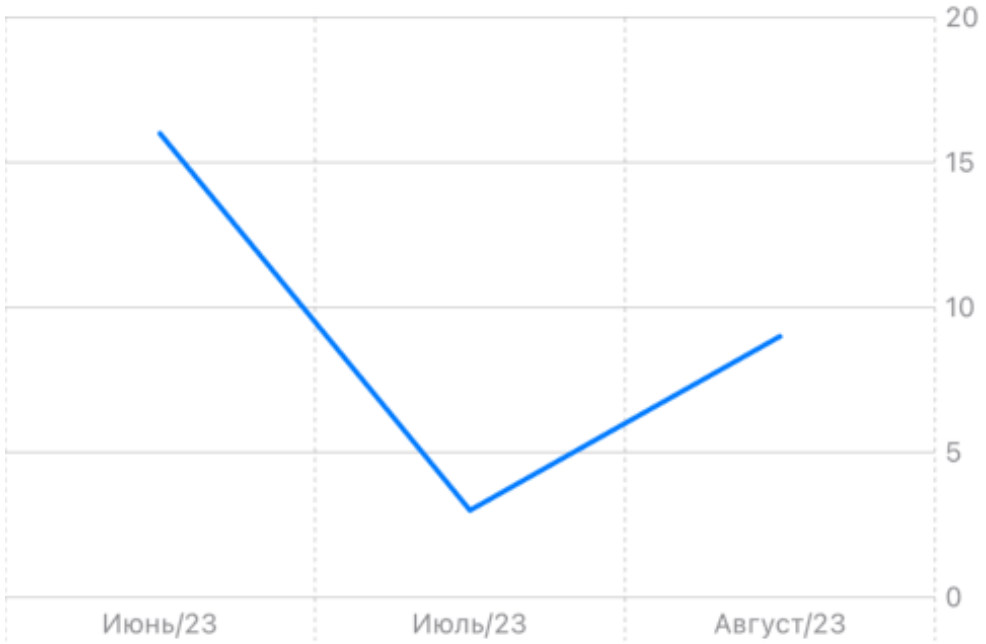
PointMark



.symbol { Image(systemName: "apple.logo").foregroundColor(.green) }

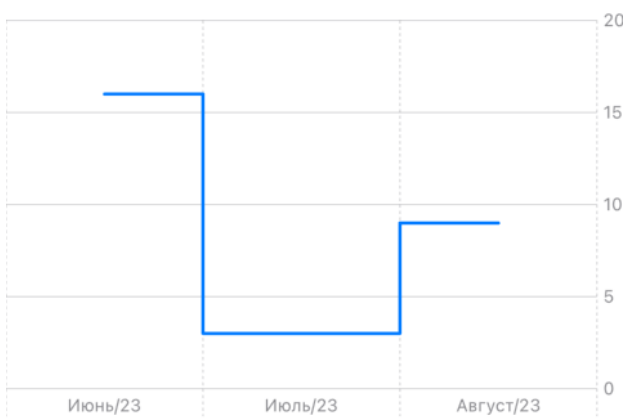
2.3 Возможности кастомизации графиков

LineMark

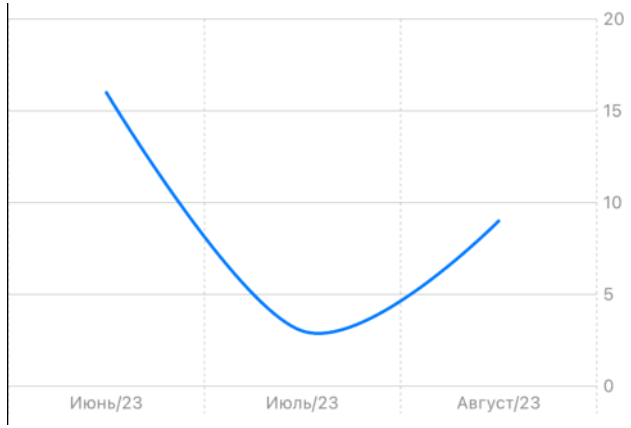


2.3 Возможности кастомизации графиков

LineMark



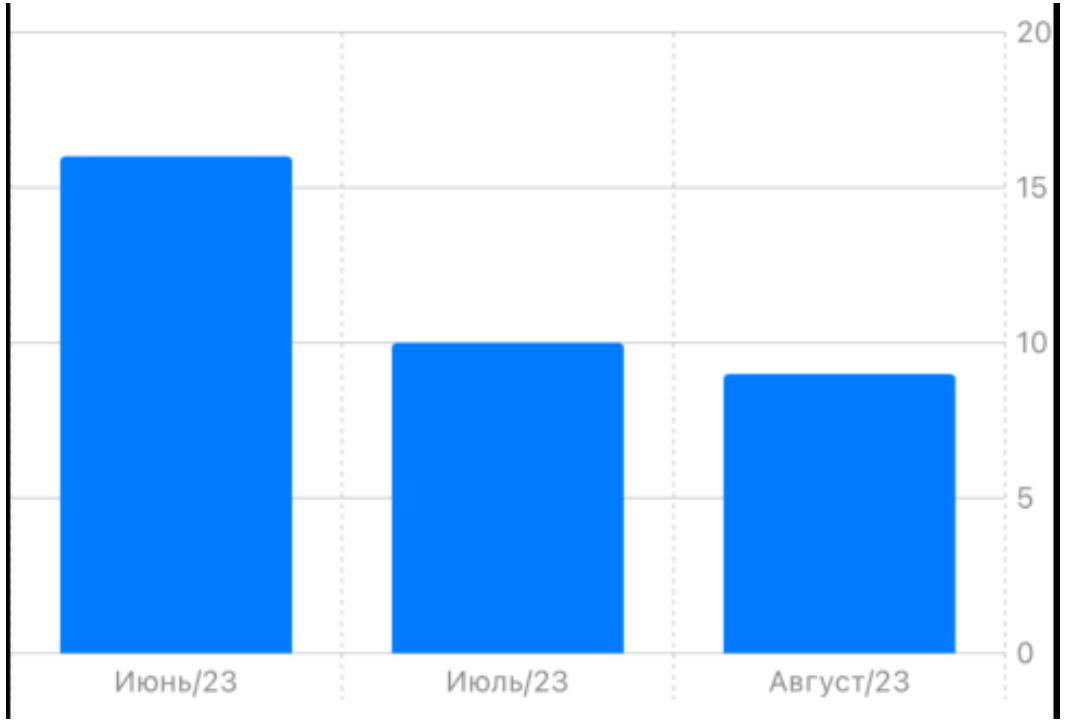
.interpolationMethod(.stepCenter)



.interpolationMethod(.cardinal)

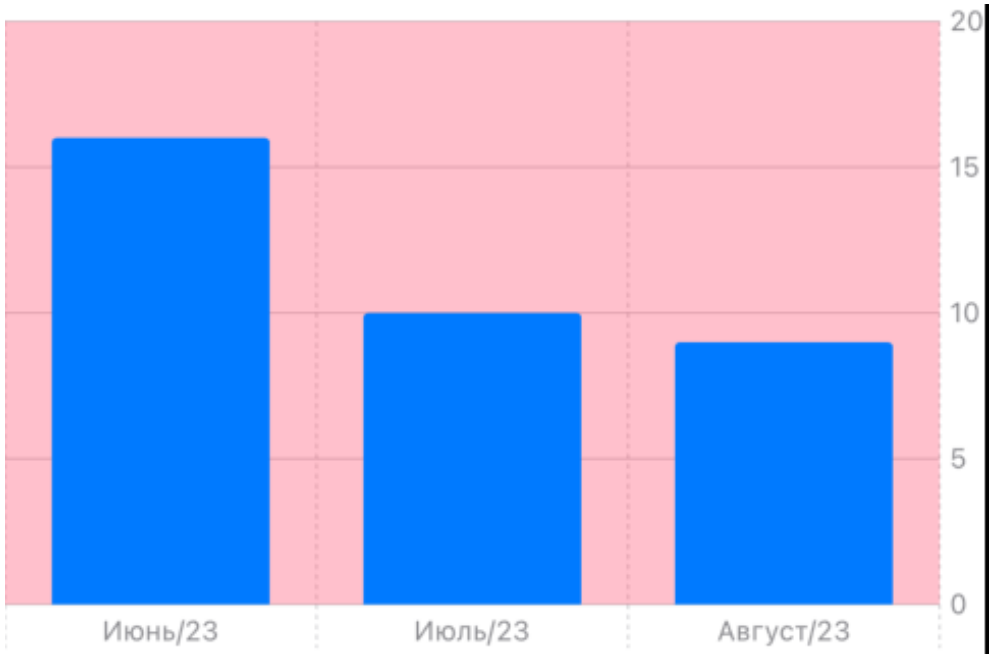
2.3 Возможности кастомизации графиков

BarMark



2.3 Возможности кастомизации графиков

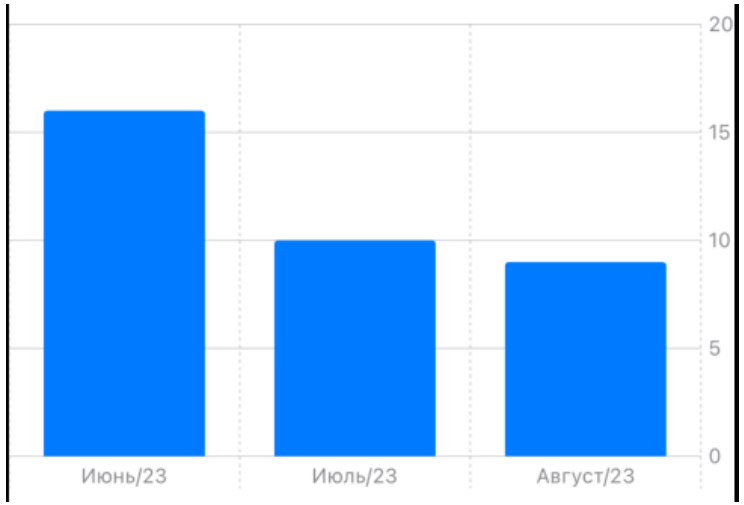
BarMark



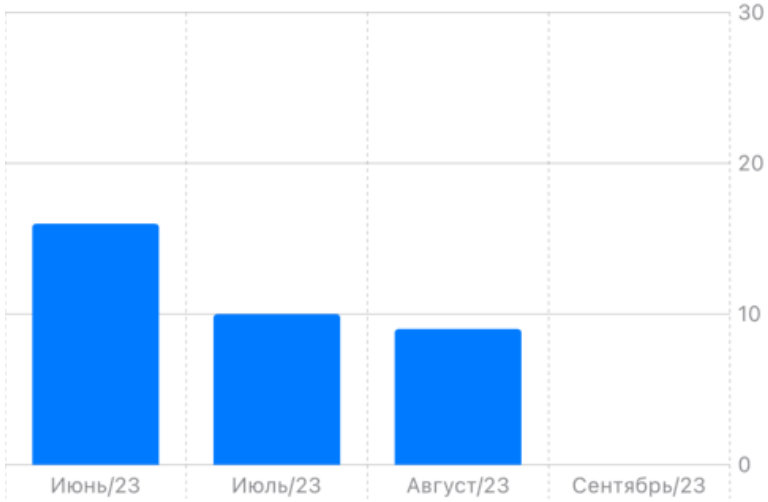
.chartPlotStyle { plotArea in plotArea.background(.pink.opacity(0.3)) }

2.3 Возможности кастомизации графиков

BarMark



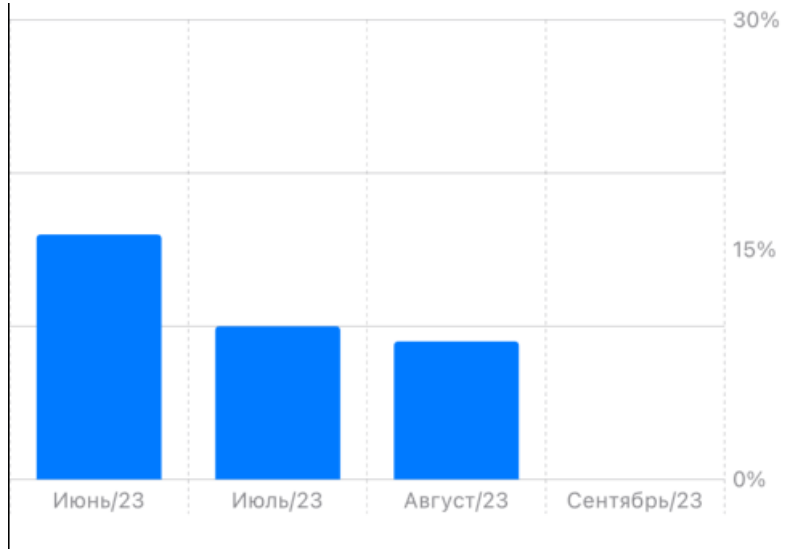
Default




```
.chartYScale(domain: [0, 30])
.chartXScale(domain: [
"Июнь/23",
“Июль/23",
 “Август/23",
 “Сентябрь/23"])
```

2.3 Возможности кастомизации графиков

BarMark




```
.chartYAxis {
    AxisMarks( values: [0, 15, 30] ) {
        AxisValueLabel(format: Decimal.FormatStyle.Percent.percent.scale(1))
    }
}
```

3. Обновления WWDC 2023

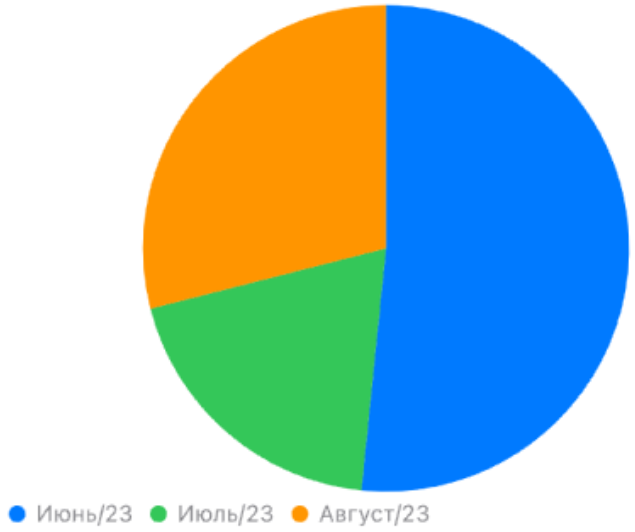
SectorMark

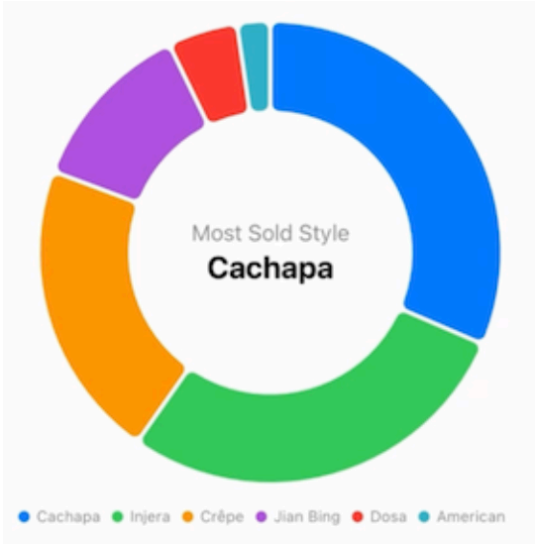
Selection

Scrolling

3. Обновления WWDC 2023

SectorMark

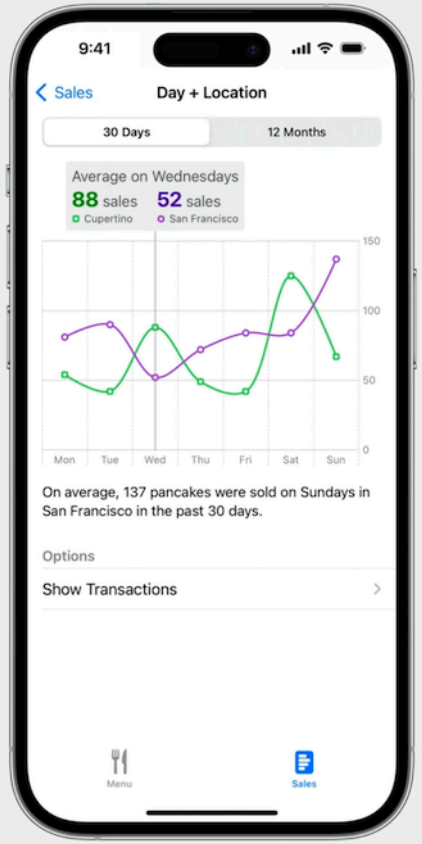




3. Обновления WWDC 2023

Selection

```
if let selectedDate {
  RuleMark(
    x: .value("Selected", selectedDate, unit: .day)
  )
  .foregroundStyle(Color.gray.opacity(0.3))
  .offset(yStart: -10)
  .zIndex(-1)
  .annotation(
    position: .top, spacing: 0,
    overflowResolution: .init(
      x: .fit(to: .chart),
      y: .disabled
    )
  ) {
    valueSelectionPopover
  }
}
```



3. Обновления WWDC 2023

Scrolling

.chartScrollableAxes(.horizontall l .vertical)

.chartXVisibleDomain(length:) и chartYVisibleDomain(length:)

.chartScrollPosition(x: $scrollPosition) и .chartScrollPosition(y: $scrollPosition)

На этом всё.

Полезные ссылки

https://github.com/AntonRomanchuk/SwiftUI-Charts-Aurora

https://developer.apple.com/documentation/charts/visualizing_your_app_s_data

Ваши вопросы