```swift
//
//  ColourSelectorApp.swift
//  ColourSelector
//

import SwiftUI

@main
struct ColourSelectorApp: App {

    var body: some Scene {
        WindowGroup {
            BrowsingView()
        }
    }
}


//
//  Functions.swift
//  ColourSelector
//

import Foundation

func filtering(originalList: [Palette], on desiredFloor: Hue) -> [Palette] {

    if desiredFloor == .allHues {
        return originalList
    }

    let desiredCeiling = desiredFloor.rawValue + 60.0
    let desiredHueRange = desiredFloor.rawValue...desiredCeiling

    var filteredListOfPalettes: [Palette] = []

    for palette in originalList {

        if desiredHueRange.contains(palette.base.hue) {
            filteredListOfPalettes.append(palette)
        }

    }

    return filteredListOfPalettes

}


//
//  Hue.swift
//  ColourSelector
//

import Foundation

enum Hue: Double {
    case allHues = -30
    case red = 0
    case yellow = 60
    case green = 120
    case teal = 180
    case blue = 240
```

```
    case purple = 300
}


//
//  Tile.swift
//  ColourSelector
//

import SwiftUI

struct Tile: Identifiable {

    let id = UUID()
    let hue: Double
    let saturation: Double = 80
    let brightness: Double

    var hueFormattedAsString: String {
        return "\(hue.formatted(.number.precision(.fractionLength(1))))°"
    }

    var color: Color {

        return Color(
            hue: hue / 360.0,
            saturation: saturation / 100.0,
            brightness: brightness / 100.0
        )

    }

}

let redTile = Tile(hue: 0.0, brightness: 90)
let yellowTile = Tile(hue: 60.0, brightness: 90)
let blueTile = Tile(hue: 240.0, brightness: 90)


//
//  Palette.swift
//  ColourSelector
//

import SwiftUI

struct Palette: Identifiable {

    let id = UUID()
    let base: Tile

    var darker: Tile {
        return Tile(hue: base.hue, brightness: base.brightness - 30.0)
    }
    var darkest: Tile {
        return Tile(hue: base.hue, brightness: base.brightness - 60.0)
    }

}

let examplePalette = Palette(
    base: Tile(hue: 0.0, brightness: 90.0)
)
```

```swift


//
//  TileView.swift
//  ColourSelector
//

import SwiftUI

struct TileView: View {

    var tile: Tile
    var size: Double

    var body: some View {

        Rectangle()
            .fill(tile.color)
            .frame(width: size, height: size)

    }
}

#Preview {
    TileView(tile: blueTile, size: 100)
}


//
//  PaletteView.swift
//  ColourSelector
//

import SwiftUI

struct PaletteView: View {

    let palette: Palette

    var body: some View {

        HStack(spacing: 0) {

            TileView(
                tile: palette.base,
                size: 50
            )

            TileView(
                tile: palette.darker,
                size: 50
            )

            TileView(
                tile: palette.darkest,
                size: 50
            )

            Spacer()

        }

    }
```

```swift
}

#Preview {
    PaletteView(palette: examplePalette)
}


//
//  BrowsingView.swift
//  ColourSelector
//

import SwiftUI

struct BrowsingView: View {

    @State var selectedBaseHue = 0.0

    @State var history: [Palette] = []

    @State var selectedHueRange: Hue = .allHues

    var tileCreatedFromSelectedHue: Tile {
        return Tile(hue: selectedBaseHue, brightness: 90)
    }

    var paletteCreatedFromBaseTile: Palette {
        return Palette(base: tileCreatedFromSelectedHue)
    }

    var body: some View {

        HStack {

            VStack(alignment: .leading, spacing: 20) {


                TileView(
                    tile: tileCreatedFromSelectedHue,
                    size: 150.0
                )
                .padding(.trailing)


                Text(tileCreatedFromSelectedHue.hueFormattedAsString)
                    .font(.largeTitle)

                Slider(value: $selectedBaseHue,
                       in: 0...360,
                       label: { Text("Base Hue") },
                       minimumValueLabel: { Text("0") },
                       maximumValueLabel: { Text("360") })

                HStack {

                    PaletteView(palette: paletteCreatedFromBaseTile)

                    Spacer()

                    Button(action: {
                        savePalette()
                    }, label: {
                        Text("Save")
```

```
                            .font(.subheadline.smallCaps())
                    })
                    .buttonStyle(.bordered)

                }

                Spacer()

            }

            VStack(alignment: .leading) {

                Picker("Filtering on", selection: $selectedHueRange) {
                    Text("All hues (no filtering)").tag(Hue.allHues)
                    Text("Red to yellow (0° to 60°)").tag(Hue.red)
                    Text("Yellow to green (60° to 120°)").tag(Hue.yellow)
                    Text("Green to teal (120° to 180°)").tag(Hue.green)
                    Text("Teal to blue (180° to 240°)").tag(Hue.teal)
                    Text("Blue to purple (240° to 300°)").tag(Hue.blue)
                    Text("Purple to red (300° to 360°)").tag(Hue.purple)
                }
                .padding(10)

                Image("IllustrationOfHSBColorModel")
                    .resizable()
                    .scaledToFit()
                    .frame(width: 200)

                List(
                    filtering(
                        originalList: history,
                        on: selectedHueRange
                    )
                ) { currentPalette in
                    PaletteView(palette: currentPalette)
                }

            }

        }
        .padding()
    }

    func savePalette() {
        history.append(paletteCreatedFromBaseTile)
    }
}

#Preview {

    BrowsingView()
        .frame(height: 600)

}
```