🎨 HarborDot Color Customization Guide

📍 Where to Customize Colors

1. Task Row Colors (TaskRowView.swift)

Background Colors:

swift

```swift
private var taskRowBackground: Color {
    switch task.status {
    case .complete:
        return Color.green.opacity(0.08)  // ← Change opacity (0.05-0.15)
    case .inProgress:
        return Color.orange.opacity(0.08)  // ← Change color/opacity
    case .notCompleted:
        return Color.red.opacity(0.08)
    default:
        return Color(uiColor: .secondarySystemGroupedBackground)
    }
}
```

Text Colors:

swift

```swift
private var textColor: Color {
    switch task.status {
    case .complete:
        return Color.green.opacity(0.8)  // ← Adjust green shade (0.6-1.0)
    case .inProgress:
        return .primary
    case .notCompleted:
        return .red  // ← Try .red.opacity(0.9) for softer red
    default:
        return .primary
    }
}
```

Color Tag Circles:

swift

```swift
Circle()
    .fill(Color.fromString(primaryTag.returnColorString()))
    .frame(width: 20, height: 20)  // ← Size already increased
```

---

2. Next/Previous Day Buttons (DayView.swift)

Look for the date navigation section:

swift

```swift
// Find this code in DayView.swift:
Button(action: { currentDate = Calendar.current.date(byAdding: .day, value: -1, to: currentDate) ?? currentDate }) {
    Image(systemName: "chevron.left")
        .foregroundColor(.blue)  // ← Change this color
        .font(.title2)
}

Button(action: { currentDate = Calendar.current.date(byAdding: .day, value: 1, to: currentDate) ?? currentDate }) {
    Image(systemName: "chevron.right")
        .foregroundColor(.blue)  // ← Change this color
        .font(.title2)
}
```

Popular alternatives:

- `.foregroundColor(.accentColor)` - Uses your app's accent color
- `.foregroundColor(Color(red: 0.2, green: 0.6, blue: 0.9))` - Custom blue
- `.foregroundColor(.indigo)` - Built-in color
- `.foregroundColor(Color(hex: "#4A90E2"))` - Hex color (requires extension)

---

3. App-Wide Theme (AppTheme.swift or ColorExtensions.swift)

Create a centralized theme file for consistency:

Option A: Create AppTheme.swift (Recommended)


swift

```swift
import SwiftUI

struct AppTheme {
    // Primary Colors
    static let accentColor = Color.blue  // ← Main app accent
    static let completedTaskGreen = Color.green.opacity(0.8)  // ← Completed tasks
    static let inProgressOrange = Color.orange
    static let notCompletedRed = Color.red.opacity(0.9)

    // Button Colors
    static let navigationButtonColor = Color.blue
    static let addButtonColor = Color.green

    // Background Colors
    static let primaryBackground = Color(uiColor: .systemBackground)
    static let secondaryBackground = Color(uiColor: .secondarySystemGroupedBackground)
    static let taskRowBackground = Color(uiColor: .secondarySystemGroupedBackground)

    // Text Colors
    static let primaryText = Color.primary
    static let secondaryText = Color.secondary
    static let completedText = Color.green.opacity(0.8)
}
```

Then use throughout your app:


swift

```
.foregroundColor(AppTheme.navigationButtonColor)
.background(AppTheme.taskRowBackground)
```

---

4. Status Badge Colors (TaskRowView.swift)

swift

```
case .inProgress:
    HStack(spacing: 4) {
        Image(systemName: "clock.fill")
        Text("In Progress")
    }
    .foregroundColor(.orange)  // ← Change badge text color
    .background(Color.orange.opacity(0.15))  // ← Change badge background

case .notCompleted:
    HStack(spacing: 4) {
        Image(systemName: "xmark.circle.fill")
        Text("Not Done")
    }
    .foregroundColor(.red)  // ← Change badge text color
    .background(Color.red.opacity(0.15))  // ← Change badge background
```

---

5. Custom Tag Colors (TaskRowView.swift)

swift

```
Text(tag.name ?? "")
    .font(.caption)
    .foregroundColor(.blue)  // ← Change tag text color
    .background(Color.blue.opacity(0.15))  // ← Change tag background
    .cornerRadius(8)
```

Try different color schemes:

- Modern: `.purple` text with `.purple.opacity(0.12)` background
- Professional: `.indigo` text with `.indigo.opacity(0.1)` background
- Vibrant: `.cyan` text with `.cyan.opacity(0.15)` background

🎨 Recommended Color Palettes

Palette 1: Ocean Blue

swift

```
// Navigation/Buttons: Color(hex: "#0077BE")
// Completed: Color(hex: "#00A86B")
// In Progress: Color(hex: "#FFA500")
// Not Completed: Color(hex: "#E74C3C")
```

Palette 2: Forest Green

swift

```
// Navigation/Buttons: Color(hex: "#2E7D32")
// Completed: Color(hex: "#66BB6A")
// In Progress: Color(hex: "#FFA726")
// Not Completed: Color(hex: "#EF5350")
```

Palette 3: Sunset Purple


swift

```
// Navigation/Buttons: Color(hex: "#7B2CBF")
// Completed: Color(hex: "#10B981")
// In Progress: Color(hex: "#F59E0B")
// Not Completed: Color(hex: "#EF4444")
```

Palette 4: Minimal Gray


swift

```
// Navigation/Buttons: Color(hex: "#374151")
// Completed: Color(hex: "#059669")
// In Progress: Color(hex: "#D97706")
// Not Completed: Color(hex: "#DC2626")
```

🛠️ How to Use Hex Colors

Add this extension to **ColorExtensions.swift**:

swift

```swift
extension Color {
    init(hex: String) {
        let hex = hex.trimmingCharacters(in: CharacterSet.alphanumerics.inverted)
        var int: UInt64 = 0
        Scanner(string: hex).scanHexInt64(&int)
        let a, r, g, b: UInt64
        switch hex.count {
        case 3: // RGB (12-bit)
            (a, r, g, b) = (255, (int >> 8) * 17, (int >> 4 & 0xF) * 17, (int & 0xF) * 17)
        case 6: // RGB (24-bit)
            (a, r, g, b) = (255, int >> 16, int >> 8 & 0xFF, int & 0xFF)
        case 8: // ARGB (32-bit)
            (a, r, g, b) = (int >> 24, int >> 16 & 0xFF, int >> 8 & 0xFF, int & 0xFF)
        default:
            (a, r, g, b) = (255, 0, 0, 0)
        }
        self.init(
            .sRGB,
            red: Double(r) / 255,
            green: Double(g) / 255,
            blue: Double(b) / 255,
            opacity: Double(a) / 255
        )
    }
}
```

Then use:

swift

```
.foregroundColor(Color(hex: "#4A90E2"))
```

📝 Quick Testing Tips

1. **Try one change at a time** - See what you like before changing everything
2. **Use opacity wisely** - 0.08-0.15 for backgrounds, 0.7-0.9 for text
3. **Maintain contrast** - Ensure text is readable on backgrounds
4. **Test in both light and dark mode** - Colors look different!
5. **Use the built-in Color Picker** - In Xcode, click on any Color literal to see a color picker

---

🎯 Files to Modify

Element	File	Search For
Task row colors	TaskRowView.swift	taskRowBackground
Completed text color	TaskRowView.swift	textColor
Color tag circles	TaskRowView.swift	Circle().fill
Next/Prev buttons	DayView.swift	chevron.left and chevron.right
Status badges	TaskRowView.swift	statusBadge
Custom tag chips	TaskRowView.swift	ForEach(task.customTags
App-wide theme	Create AppTheme.swift	N/A - new file

💡 Pro Tips

1. **Create AppTheme.swift** for centralized color management
2. **Use semantic naming** - `accentColor` not `blueColor`
3. **Consider accessibility** - Test with high contrast mode
4. **Save your favorites** - Comment out alternative color schemes
5. **Match your brand** - Use colors from your app icon

Happy customizing! 🎨