RATing CONTROL

import UIKit

@IBDesignable class RatingControl: UIStackView {

//MARK: Properties

private var ratingButtons = [UIButton]()

var rating = 0 {

didSet {

updateButtonSelectionStates()

}

}

@IBInspectable var starSize: CGSize = CGSize(width: 44.0, height: 44.0) {

didSet {

setupButtons()

}

}

@IBInspectable var starCount: Int = 5 {

didSet {

setupButtons()

}

}

//MARK: Initialization

override init(frame: CGRect) {

super.init(frame: frame)

setupButtons()

}

required init(coder: NSCoder) {

super.init(coder: coder)

setupButtons()

}

//MARK: Button Action

func ratingButtonTapped(button: UIButton) {

guard let index = ratingButtons.index(of: button) else {

fatalError("The button, \(button), is not in the ratingButtons array: \(ratingButtons)")

}

// Calculate the rating of the selected button

let selectedRating = index + 1

if selectedRating == rating {

// If the selected star represents the current rating, reset the rating to 0.

rating = 0

} else {

// Otherwise set the rating to the selected star

rating = selectedRating

}

}

//MARK: Private Methods

private func setupButtons() {

// Clear any existing buttons

for button in ratingButtons {

removeArrangedSubview(button)

button.removeFromSuperview()

}

ratingButtons.removeAll()

// Load Button Images

let bundle = Bundle(for: type(of: self))

let filledStar = UIImage(named: "filledStar", in: bundle, compatibleWith: self.traitCollection)

let emptyStar = UIImage(named:"emptyStar", in: bundle, compatibleWith: self.traitCollection)

let highlightedStar = UIImage(named:"highlightedStar", in: bundle, compatibleWith: self.traitCollection)

for index in 0..<starCount {

// Create the button

let button = UIButton()

// Set the button images

button.setImage(emptyStar, for: .normal)

button.setImage(filledStar, for: .selected)

button.setImage(highlightedStar, for: .highlighted)

button.setImage(highlightedStar, for: [.highlighted, .selected])

// Add constraints

button.translatesAutoresizingMaskIntoConstraints = false

button.heightAnchor.constraint(equalToConstant: starSize.height).isActive = true

button.widthAnchor.constraint(equalToConstant: starSize.width).isActive = true

// Set the accessibility label

button.accessibilityLabel = "Set \(index + 1) star rating"

// Setup the button action

button.addTarget(self, action: #selector(RatingControl.ratingButtonTapped(button:)), for: .touchUpInside)

// Add the button to the stack

addArrangedSubview(button)

// Add the new button to the rating button array

ratingButtons.append(button)

}

updateButtonSelectionStates()

}

private func updateButtonSelectionStates() {

for (index, button) in ratingButtons.enumerated() {

// If the index of a button is less than the rating, that button should be selected.

button.isSelected = index < rating

// Set accessibility hint and value

let hintString: String?

if rating == index + 1 {

hintString = "Tap to reset the rating to zero."

} else {

hintString = nil

}

let valueString: String

switch (rating) {

case 0:

valueString = "No rating set."

case 1:

valueString = "1 star set."

default:

valueString = "\(rating) stars set."

}

button.accessibilityHint = hintString

button.accessibilityValue = valueString

}

}

}