Sample Code

Configuring the PencilKit tool picker

Incorporate a custom PencilKit tool picker with a variety of system and custom tools into a drawing app.

Download

iOS 18.0+ | iPadOS 18.0+ | visionOS 2.0+ | Xcode 16.0+

Overview

This sample app shows how to configure a `PKToolPicker`. The app is a simple drawing tool that allows a person to draw on a canvas using tools from the tool picker. In addition to showing system tools, the tool picker provides a custom tool that a person can use to place stamps on the canvas. The tool picker also contains an accessory item for adding a signature to the canvas.

> **Note**
>
> This sample code project is associated with WWDC24 session 10214: Squeeze the most out of Apple Pencil.

Configure the sample code project

Although this sample app's basic functionality is available in Simulator, running the project with physical devices is recommended to demonstrate the full capabilities of the hardware. Run this project with Apple Pencil Pro and a compatible iPad.

Create a system tool

This sample app creates and configures a system tool to show in the tool picker: a pen, which is a PKToolPickerInkingItem with the PKInkingTool.InkType.pen ink type.

```
let penWidth = PKInkingTool.InkType.pen.defaultWidth
let secondPen = PKToolPickerInkingItem(type: .pen,
                                        color: toolPickerRed,
                                        width: penWidth,
                                        identifier: "com.example.apple-samplecode.sec
```

This sample app uses an `identifier` to differentiate between this pen and the default system pen so they can both show up in the picker. If you don't specify an identifier to differentiate between multiple system tools of the same type, the picker treats those tool items as duplicates and only shows the first item in the list.

Create a custom tool

The PencilKit tool picker can contain a combination of system tools and custom tools. To add a custom tool to the picker, this sample app creates a PKToolPickerCustomItem.Configuration to define details about the custom tool. This sample app creates a custom tool to place stamps in the shape of animals on the canvas.

```
let identifier = "com.example.apple-samplecode.animal-stamp"
let name = NSLocalizedString("Animal Stamp", comment: "Name of the animal stamp tool
var config = PKToolPickerCustomItem.Configuration(identifier: identifier, name: name
```

The imageProvider is a closure that generates an image for the custom tool. The system automatically calls this closure to fetch a new image for the custom tool when PencilKit attributes like color or width change. If a custom tool has additional custom properties that affect the appearance of the tool, fetch a new image manually by calling reloadImage() when those properties change.

```
config.imageProvider = { item in
    return ImageProvider.shared.drawImage(color: item.color,
                                          width: item.width,
                                          attributeImage: attributeVC.attributeModel
}
```

The viewControllerProvider is a closure that provides an additional view controller to display above the system controls in the tool attributes popover. In this sample app, this view controller shows additional customization options for the stamp tool, like being able to choose which animal to use for the custom stamp tool.

```
config.viewControllerProvider = { item in
    attributeVC.attributeModel.color = item.color
    attributeVC.reload()
    return attributeVC
}
```

Specify the order of the tools in the picker

Tools in the tool picker can appear in any order you specify when you create the picker using `init(toolItems:)`. This sample app places the custom tool item first, but it can appear in any order in the `toolItems` array. After the custom tool item, this sample app adds the second pen and the default set of `PKToolPicker` tools.

```
let toolItems = [self.animalStampWrapper.toolItem, secondPen] + PKToolPicker().toolI

self.toolPicker = PKToolPicker(toolItems: toolItems)
```

Add an accessory item

An `accessoryItem` adds an optional button to the tool picker to provide quick access to additional behavior. This sample app adds an accessory item that allows placing a signature on the canvas.

```
toolPicker.accessoryItem = UIBarButtonItem(image: UIImage(systemName: "signature"),
    self.signDrawing(sender: nil)
}))
```

Draw a hover preview for tools

A hover preview can provide useful feedback about where a tool might touch down on the canvas. This sample app checks the value of `prefersHoverToolPreview` to see the user preference for displaying a hover tool preview, and uses that to determine whether to draw the preview. Then, it draws the preview view in response to the state of a `UIHoverGestureRecognizer` gesture. This sample app uses `rollAngle` to rotate the preview view in response to the barrel-roll angle of Apple Pencil Pro.

```
let point = sender.location(in: stampView)
// Subtract the `rollAngle` instead of adding it, because it's defined to go in the
// direction from `azimuthAngle`.
let angleInRadians = sender.azimuthAngle(in: sender.view) - sender.rollAngle

switch sender.state {
case .changed:
    hoverPreviewView?.removeFromSuperview()
    hoverPreviewView = nil
    if let imageView = animalStampWrapper.stampImageView(for: point, angleInRadians:
        imageView.alpha = 0.5
        stampView.addSubview(imageView)
        hoverPreviewView = imageView
    } else {
        hoverPreviewView = nil
    }
default:
    hoverPreviewView?.removeFromSuperview()
    hoverPreviewView = nil
}
```

See Also

Tools

`class PKToolPicker`

A tool palette that displays a selection of drawing tools and colors for tools that a person can choose from.

`struct PKInkingTool`

A structure that defines the drawing characteristics (width, color, pen style) to use when drawing lines on a canvas view.

`struct PKEraserTool`

A tool for erasing previously drawn content in a canvas view.

`struct PKLassoTool`

A tool for selecting stroked lines and shapes in a canvas view.

`protocol PKTool`

An interface adopted by drawing and writing tools used by a canvas view.