Fill Extrustion Layer

Add a fill extrustion layer and adjust the light dynamically with a slider.

> Note: This example uses UIKit.

This examples adds a `MHFillExtrusionStyleLayer`. The to be rendered height of the buildings is read from the vector data.

The global `MHStyle/light` property is adjusted as the user changes a slider, which affects the fill extrustion layer.

```
class BuildingLightExample: UIViewController, MHMapViewDelegate {
    var mapView: MHMapView!
    var light: MHLight!
    var slider: UISlider!
    override func viewDidLoad() {
        super.viewDidLoad()

        mapView = MHMapView(frame: view.bounds, styleURL:
AMERICANA_STYLE)
        mapView.autoresizingMask = [.flexibleWidth, .flexibleHeight]
        mapView.delegate = self

        // Center the map on the Flatiron Building in New York, NY.
        mapView.camera = MHMapCamera(lookingAtCenter:
CLLocationCoordinate2D(latitude: 40.7411, longitude: -73.9897),
altitude: 1200, pitch: 45, heading: 0)

        view.addSubview(mapView)

        addSlider()
    }

    // Add a slider to the map view. This will be used to adjust the
map's light object.
    func addSlider() {
        slider = UISlider()
        slider.translatesAutoresizingMaskIntoConstraints = false
        slider.autoresizingMask = [.flexibleTopMargin,
.flexibleLeftMargin, .flexibleRightMargin]
        slider.minimumValue = -180
        slider.maximumValue = 180
        slider.value = 0
        slider.isContinuous = true

        slider.addTarget(self, action: #selector(shiftLight), for:
.valueChanged)
        view.addSubview(slider)
        NSLayoutConstraint.activate([
            slider.leadingAnchor.constraint(equalTo: view.leadingAnchor,
constant: 40.0),
            slider.trailingAnchor.constraint(equalTo:
view.trailingAnchor, constant: -40.0),
            slider.bottomAnchor.constraint(equalTo: view.bottomAnchor,
constant: -75.0),
        ])
    }

    func mapView(_: MHMapView, didFinishLoading style: MHStyle) {
        // Add a MHFillExtrusionStyleLayer.
        addFillExtrusionLayer(style: style)

        // Create an MHLight object.
        light = MHLight()

        // Create an MHSphericalPosition and set the radial, azimuthal,
and polar values.
        // Radial : Distance from the center of the base of an object to
its light. Takes a CGFloat.
        // Azimuthal : Position of the light relative to its anchor.
Takes a CLLocationDirection.
        // Polar : The height of the light. Takes a CLLocationDirection.
        let position = MHSphericalPositionMake(5, 180, 80)
        light.position = NSExpression(forConstantValue:
NSValue(mhSphericalPosition: position))

        // Set the light anchor to the map and add the light object to
the map view's style. The light anchor can be the viewport (or rotates
with the viewport) or the map (rotates with the map). To make the
viewport the anchor, replace `map` with `viewport`.
        light.anchor = NSExpression(forConstantValue: "map")
        style.light = light
    }

    @objc func shiftLight() {
        // Use the slider's value to change the light's polar value.
        let position = MHSphericalPositionMake(5, 180,
CLLocationDirection(slider.value))
        light.position = NSExpression(forConstantValue:
NSValue(mhSphericalPosition: position))
        mapView.style?.light = light
    }

    func addFillExtrusionLayer(style: MHStyle) {
        // Access the OpenMapTiles source and use it to create a
`MHFillExtrusionStyleLayer`. The source identifier is `openmaptiles`.
Use the `sources` property on a style to verify source identifiers.
        guard let source = style.source(withIdentifier: "openmaptiles")
else {
            print("Could not find source openmaptiles")
            return
        }
        let layer = MHFillExtrusionStyleLayer(identifier: "extrusion-
layer", source: source)
        layer.sourceLayerIdentifier = "building"
        layer.fillExtrusionBase = NSExpression(forKeyPath:
"render_min_height")
        layer.fillExtrusionHeight = NSExpression(forKeyPath:
"render_height")
        layer.fillExtrusionOpacity = NSExpression(forConstantValue: 0.8)
        layer.fillExtrusionColor = NSExpression(forConstantValue:
UIColor.white)

        // Access the map's layer with the identifier "poi" and insert
the fill extrusion layer below it.
        let symbolLayer = style.layer(withIdentifier: "poi")!
        style.insertLayer(layer, below: symbolLayer)
    }
}
```