BY
SHUBAM GUPTA

UX/UI

SWIFT UI - LIST

WHAT IS LIST IN SWIFTUI?

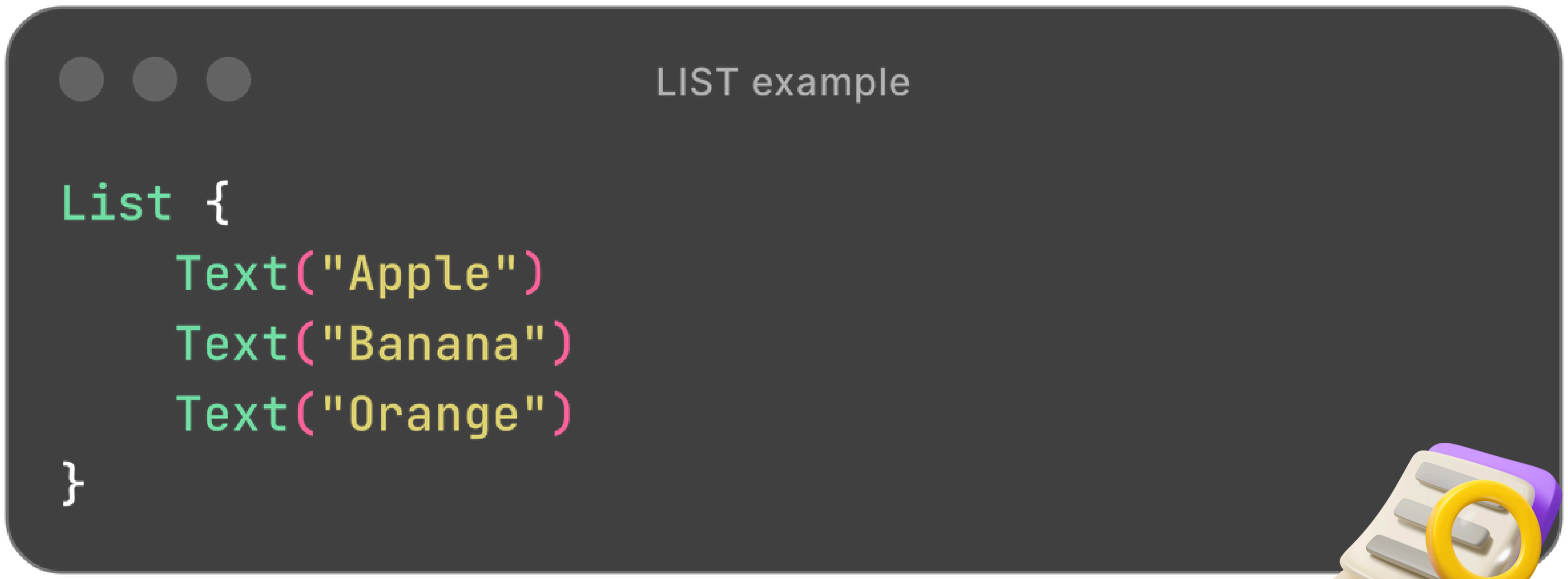
List is a **scrollable**, **data-driven** container optimized for:

- Large data sets
- Dynamic updates
- System behaviors (swipe, delete, reorder, accessibility)

Think of it as SwiftUI's version of:

👉 **UITableView**

LIST (EXAMPLE)



```swift
List {
    Text("Apple")
    Text("Banana")
    Text("Orange")
}
```

DYNAMIC LIST

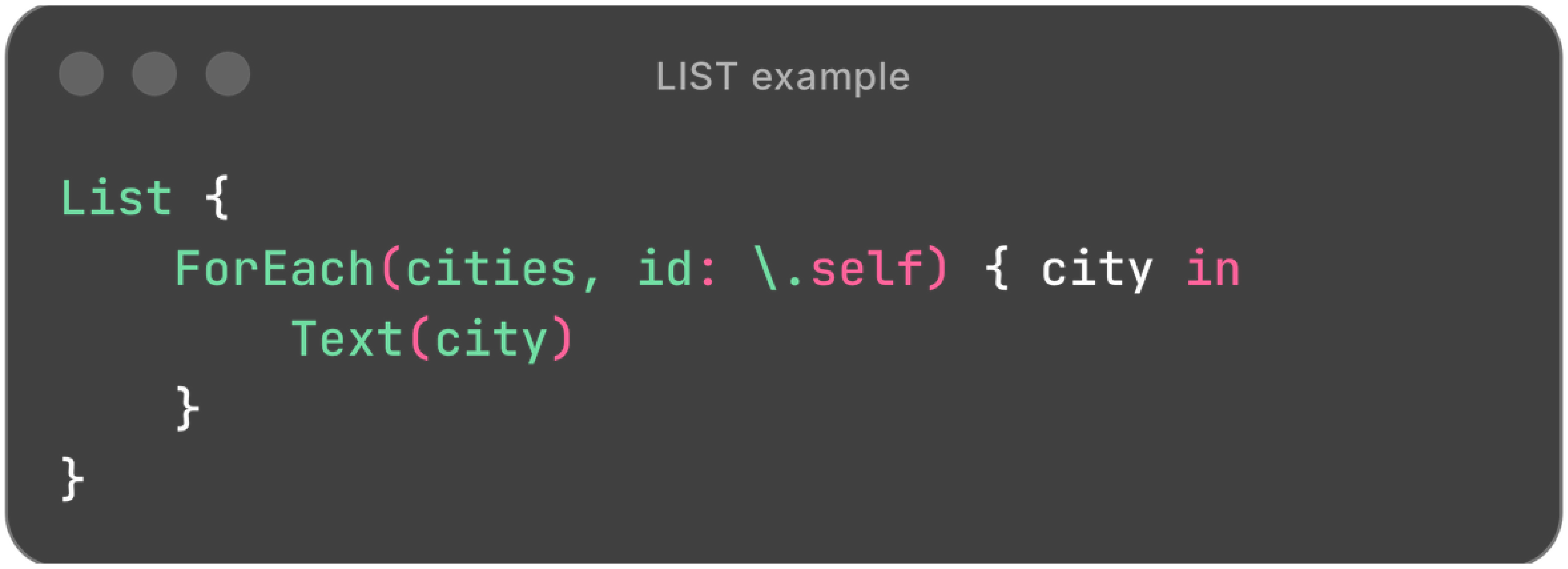
LIST example

```
let cities = ["Delhi", "Mumbai", "Bangalore", "Chennai"]

List(cities, id: \.self) { city in
    Text(city)
}
```

- **Cities** is an array of city names that we want to display on the screen.
- **List(cities, id: \.self)** creates a scrollable list, and **id: \.self** tells SwiftUI that each city name itself is a unique identifier.
- **Text(city)** shows each city name as one row in the list.

FOREACH INSIDE LIST

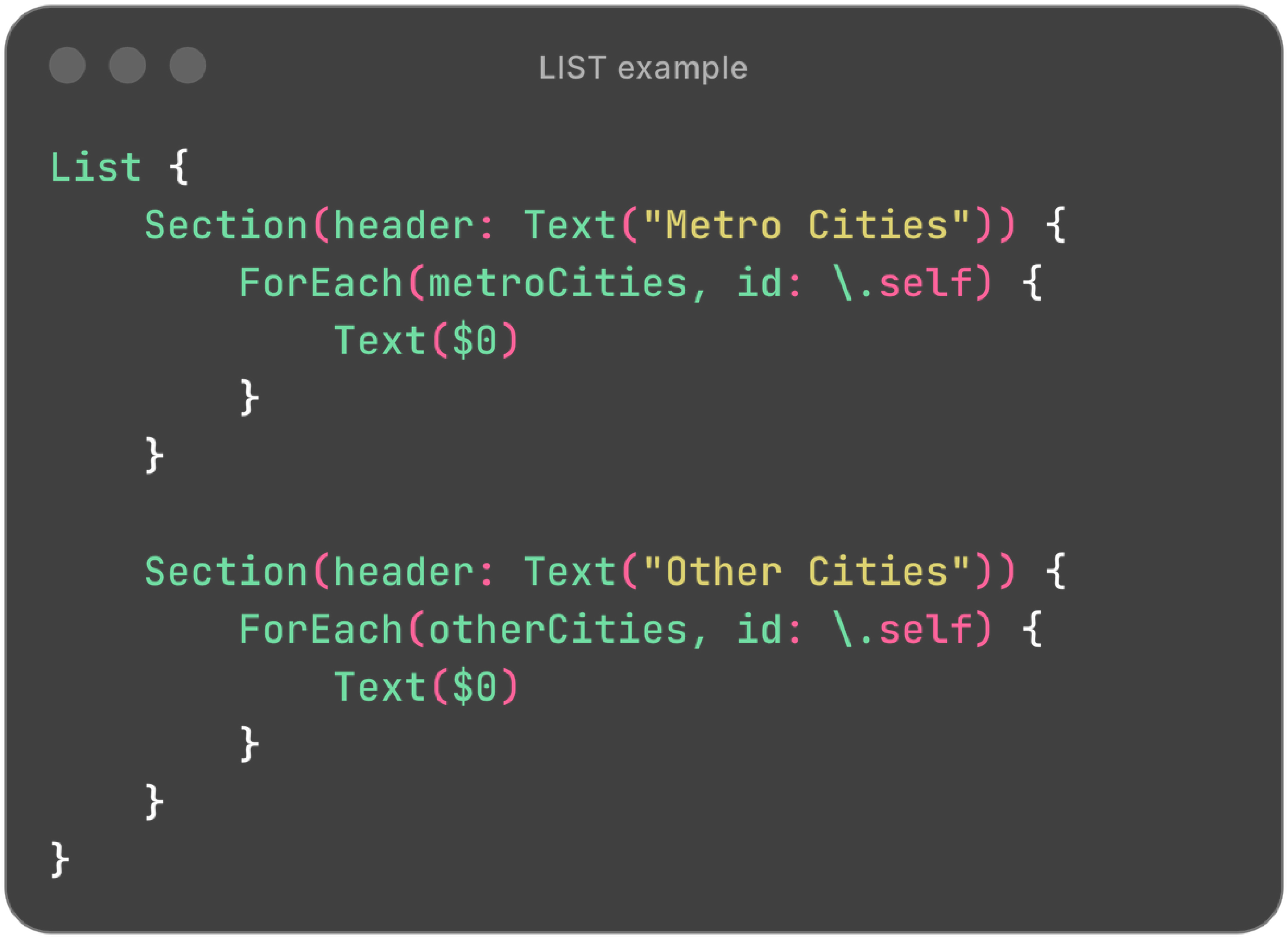


```swift
List {
    ForEach(cities, id: \.self) { city in
        Text(city)
    }
}
```

- **List {}** creates a scrollable list container in SwiftUI.
- **ForEach(cities, id: \.self)** loops through each city in the **cities** array and uses the city name itself as a unique ID.
- **Text(city)** displays each city as one row inside the list.

👉 In short: List shows the UI, ForEach provides the data, Text shows the content.

SECTIONS IN LIST



```swift
List {
    Section(header: Text("Metro Cities")) {
        ForEach(metroCities, id: \.self) {
            Text($0)
        }
    }

    Section(header: Text("Other Cities")) {
        ForEach(otherCities, id: \.self) {
            Text($0)
        }
    }
}
```

- **List {}** creates a scrollable list.
- **Section(header:)** divides the list into groups with titles like Metro Cities and Other Cities.
- **ForEach** loops through each array and **Text($0)** displays each city as a row.

SWIPE ACTIONS

LIST example

```
.swipeActions(edge: .trailing) {
    Button(role: .destructive) {
        deleteItem()
    } label: {
        Label("Delete", systemImage: "trash")
    }
}
```

- **swipeActions(edge: .trailing)** adds swipe options when you swipe a list row from right to left.
- **Button(role: .destructive)** shows a red Delete action, indicating a dangerous action.
- When tapped, it calls **deleteItem()** and removes the item.

REORDER ROWS

LIST example

```
.onMove { source, destination in
    cities.move(fromOffsets: source, toOffset: destination)
}
```

- onMove allows the user to drag and reorder rows in a List.
- cities.move(fromOffsets:toOffset:) updates the cities array to match the new order.

PULL TO REFRESH

LIST example

```
.refreshable {
    await loadData()
}
```

- **refreshable** adds pull-to-refresh functionality to a **List** or **ScrollView**.
- When the user pulls down, **loadData()** is called asynchronously to reload the data.

DELETE USING .ONDELETE

LIST example

```swift
List {
    ForEach(cities, id: \.self) { city in
        Text(city)
    }
    .onDelete { indexSet in
        cities.remove(atOffsets: indexSet)
    }
}
```

- **ForEach** displays each city as a row inside the **List**.
- **onDelete** enables swipe-to-delete on list rows.
- **cities.remove(atOffsets:)** removes the selected city from the array.

WHY WE USE LIST IN SWIFTUI

- List gives us a **ready-made**, native iOS table with scrolling, cell reuse, swipe actions, delete, move, and refresh all with very little code.
- Features like swipe to delete, pull to refresh, and reordering come almost for free, which earlier required a lot of code in UIKit.
- List automatically adapts to different screen sizes, accessibility, and system styles, reducing UI bugs.
- 👉 In short: Less code, more built-in behavior, native iOS feel.

FOLLOW US

THANK YOU
BY SHUBAM GUPTA