iOS 應用程式開發

Week 5

解決你的問題

解決問題的步驟

- 列出問題所在
- 定義要解決的範圍
- 規劃你的計劃
  (如何用 Apps 解決你的問題)
- 實現你的計劃

例子

- 小明平常有不少支出，買東西、食飯、坐車等等...
- 他覺得自己應該沒有很多支出，但想找一個方法統計一下
- 他希望在他完成消費計立即記錄下來
- 如果可以還想幫他買的東西拍照，最好記下地點

定義要解決的範圍

- 列出所有消費
- 快速輸入消費的內容 (如何輸入?)
  - 拍攝條碼就能知道多少錢 (能做到嗎?)
  - 手動輸入
    - 拍攝照片作記錄 (未學😅)
    - 記下消費的地點 (未學😅)
- 統計每個月消費了多少?
- 統計每個月哪種消費最多?

➡ 還有時間的限制 (有多少時間去完成?)

定義要解決的範圍

- 列出所有消費
- 快速輸入消費的內容 (如何輸入?)
  - ~~拍攝條碼就能知道多少錢 (能做到嗎?)~~ (暫時未有能力做)
  - 手動輸入
    - ~~拍攝照片作記錄 (未學😅)~~ (時間不夠)
    - ~~記下消費的地點 (未學😅)~~ (時間不夠)
- ~~統計每個月消費了多少?~~ (留待下一個版本)
- ~~統計每個月哪種消費最多?~~ (留待下一個版本)

➡ 還有時間的限制 (有多少時間去完成?)

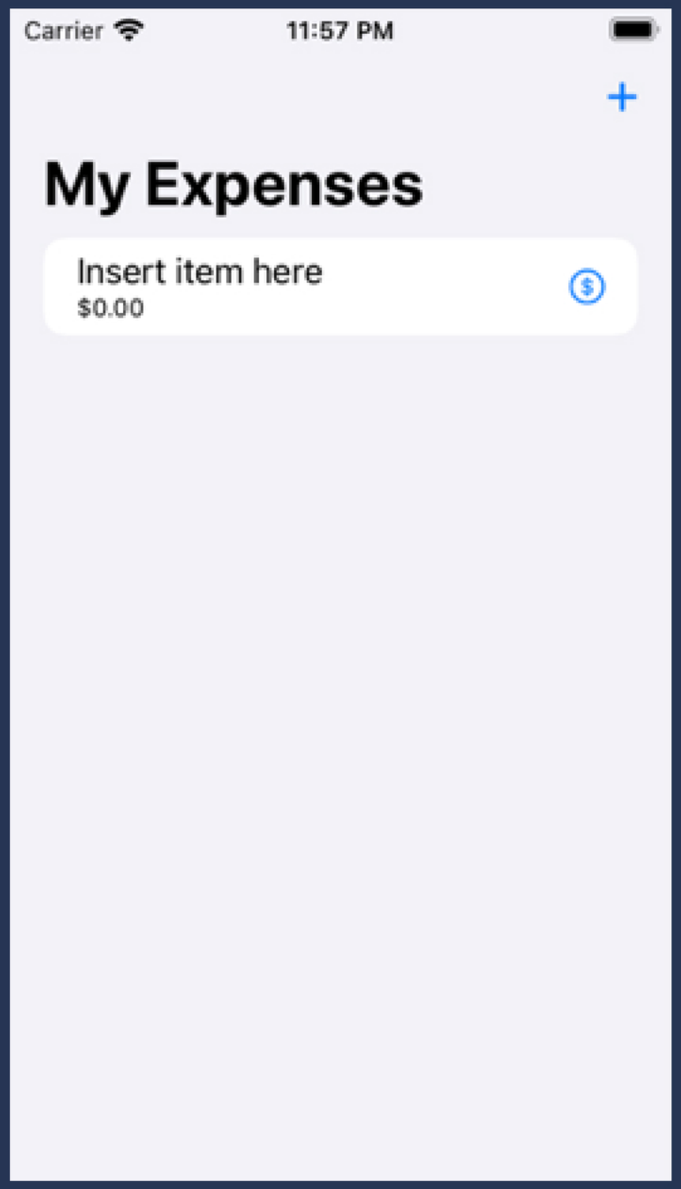
定義要解決的範圍 (續)

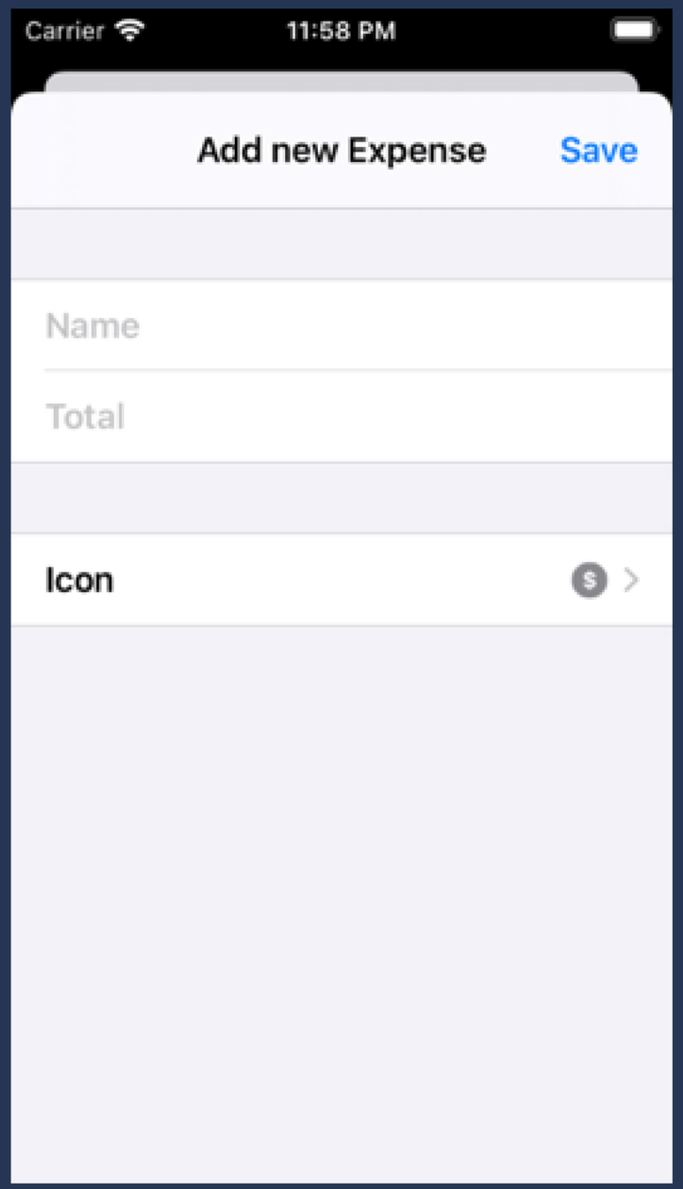
- 要記錄哪些資料
  - 消費內容
  - 消費金額
  - 消費類型 (買東西/交通/遊戲...)

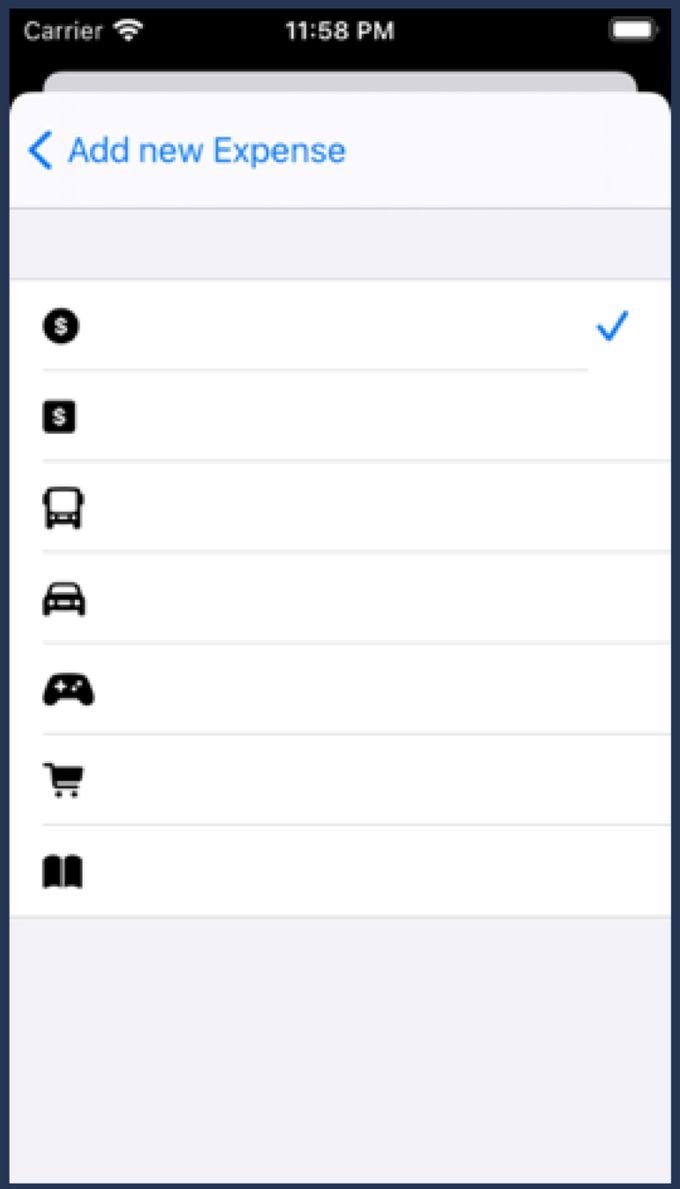
規劃你的計劃

- 繪畫設計 (Mockup) 或製作原型 (Prototype)
- 比較小型的程式可以直接紙上設計或用 SwiftUI 製作原型
- 預先規劃可避免來回修改

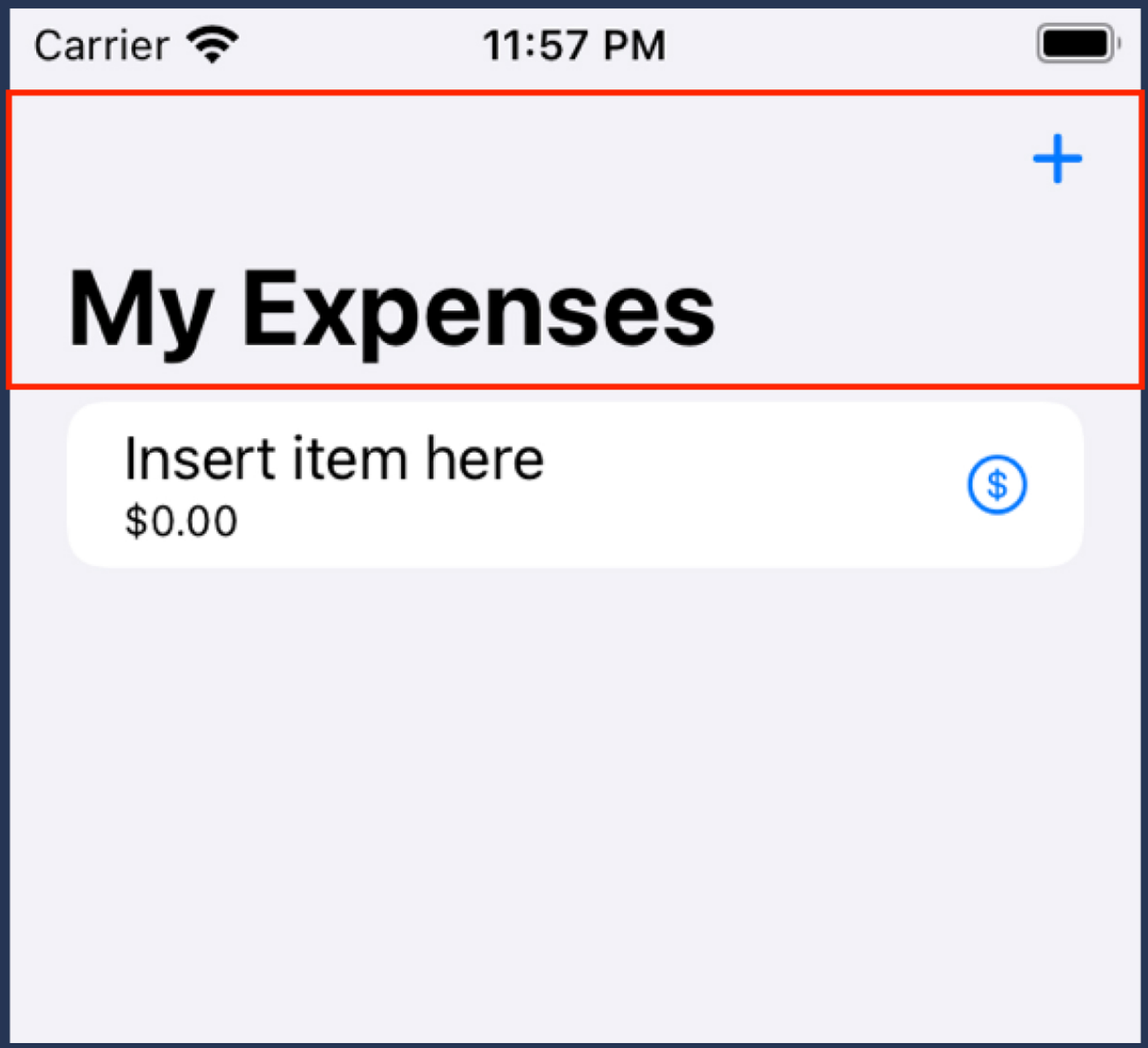
在 XCode 製作原型
(未實現功能)


Carrier
11:57 PM
My Expenses
Insert item here
$0.00

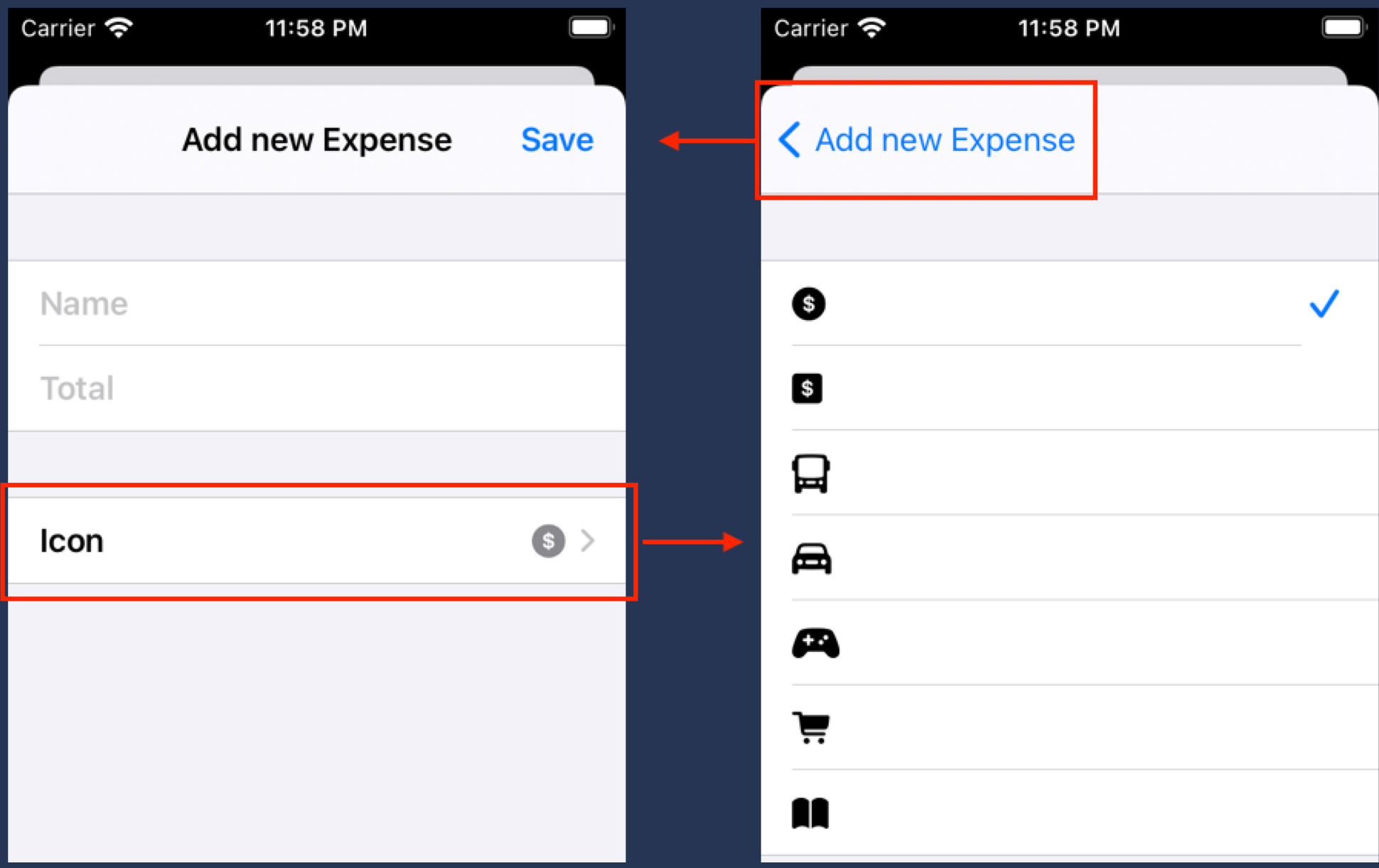

Carrier
11:58 PM
Add new Expense
Save
Name
Total
Icon


Carrier
11:58 PM
Add new Expense

Navigation View



Navigation View



Navigation View

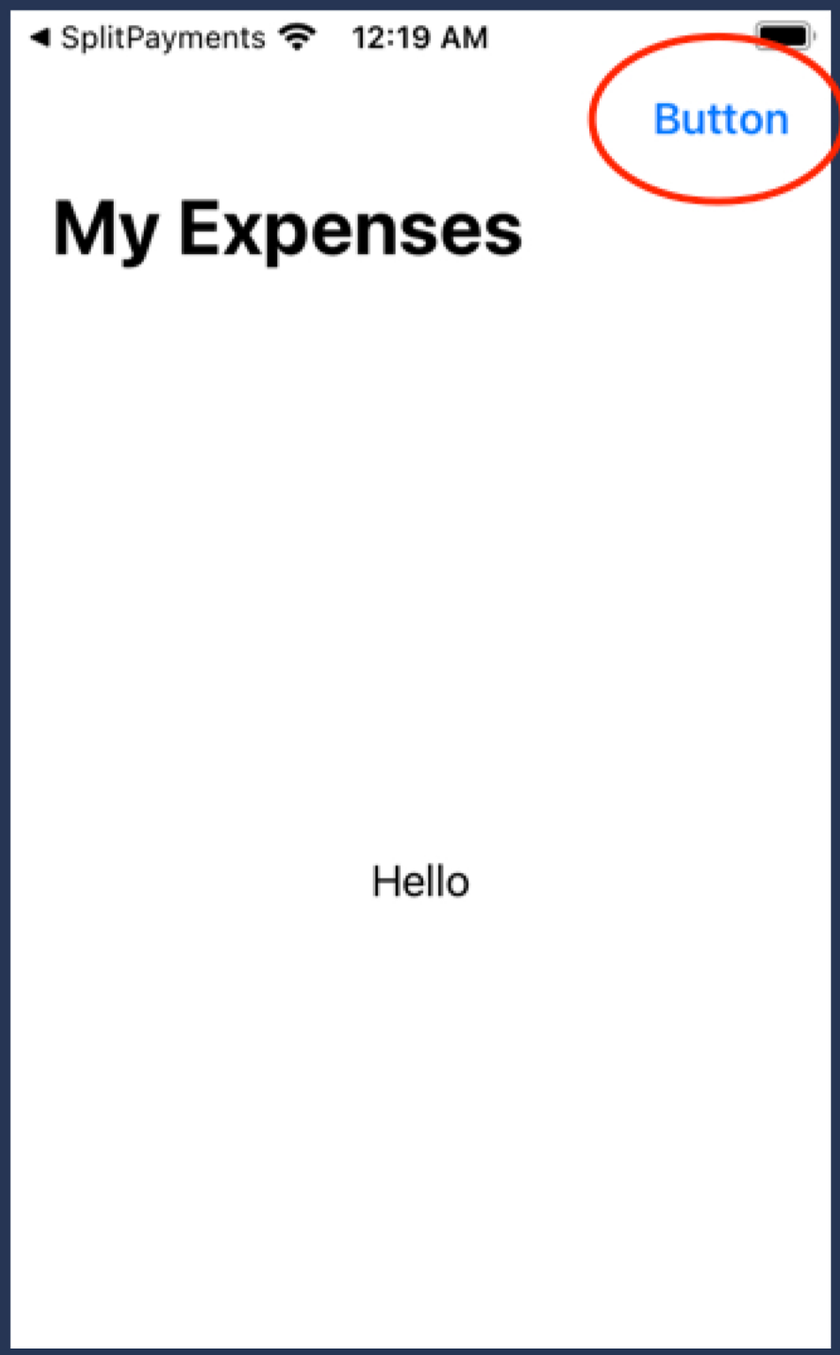
```
NavigationView {
    Text("Hello")
        .navigationTitle(Text("My Expenses"))
}
```

Navigation View

```
.navigationBarItems(trailing:
    Button(action:{
      // ...
    },
    label: {
        Text("Button")
    })
)
```

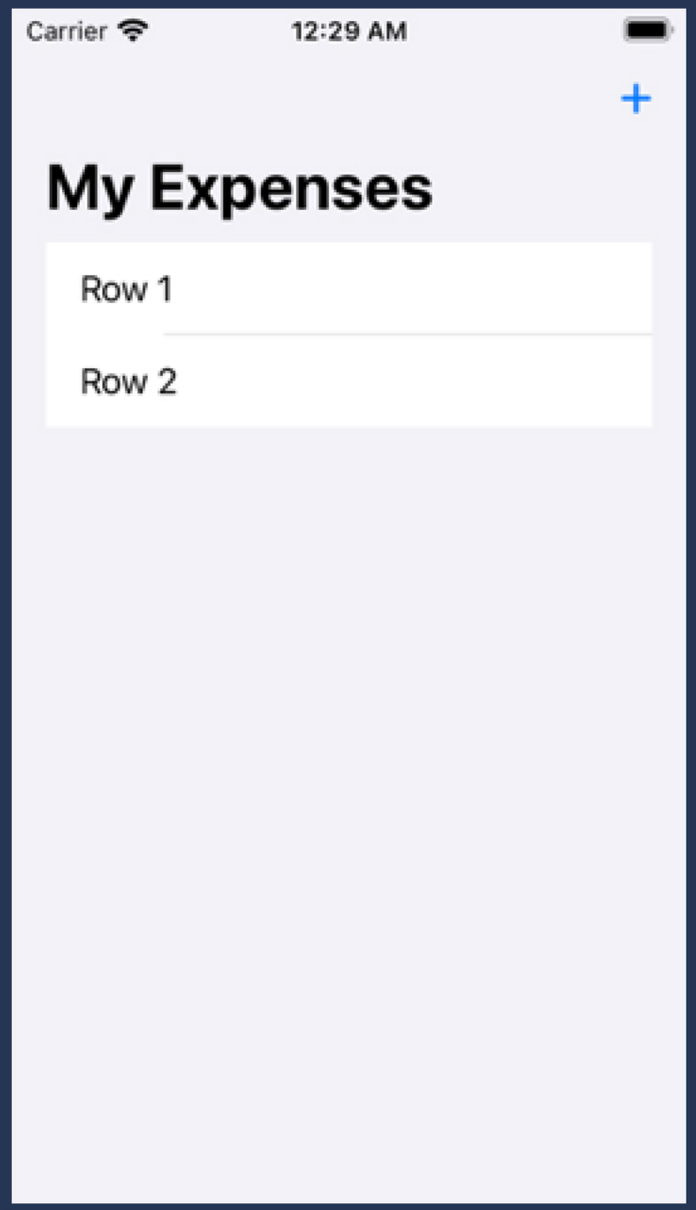
trailing: / leading:

(在左至右的環境，trailing 為右邊，leading 為左邊。)



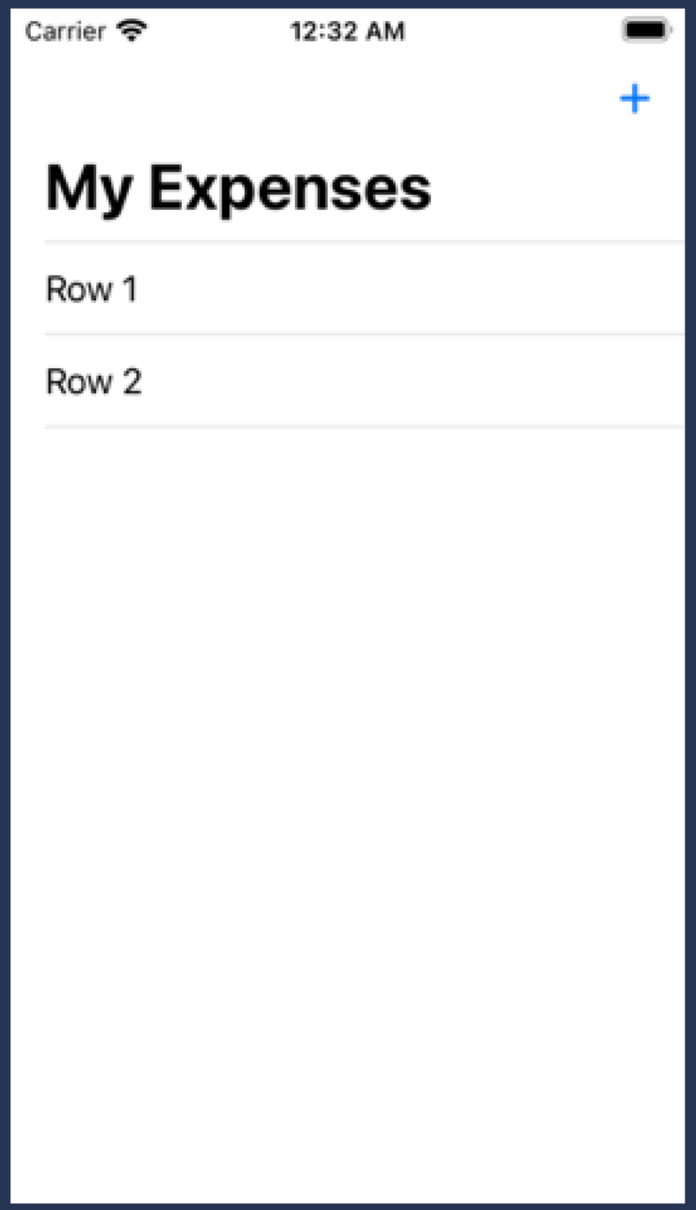
List

```
List {
    Text("Row 1")
    Text("Row 2")
}
```



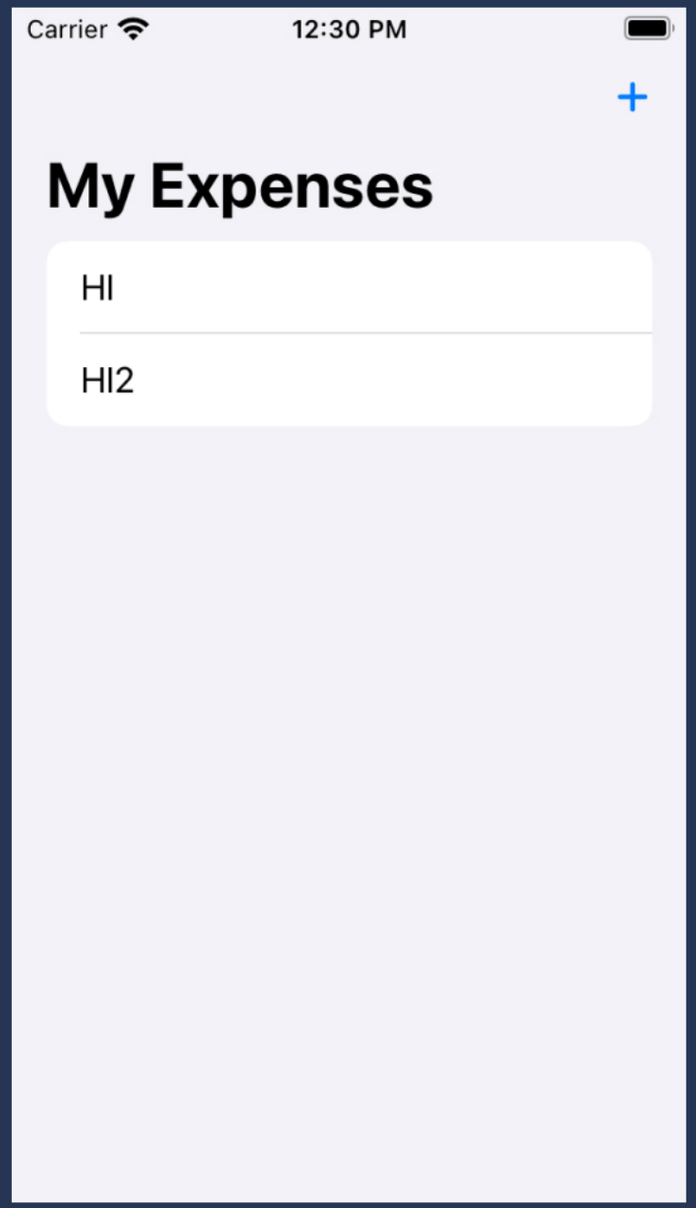
.listStyle()

```
List {
    Text("Row 1")
    Text("Row 2")
}
.listStyle(
    PlainListStyle()
)
```



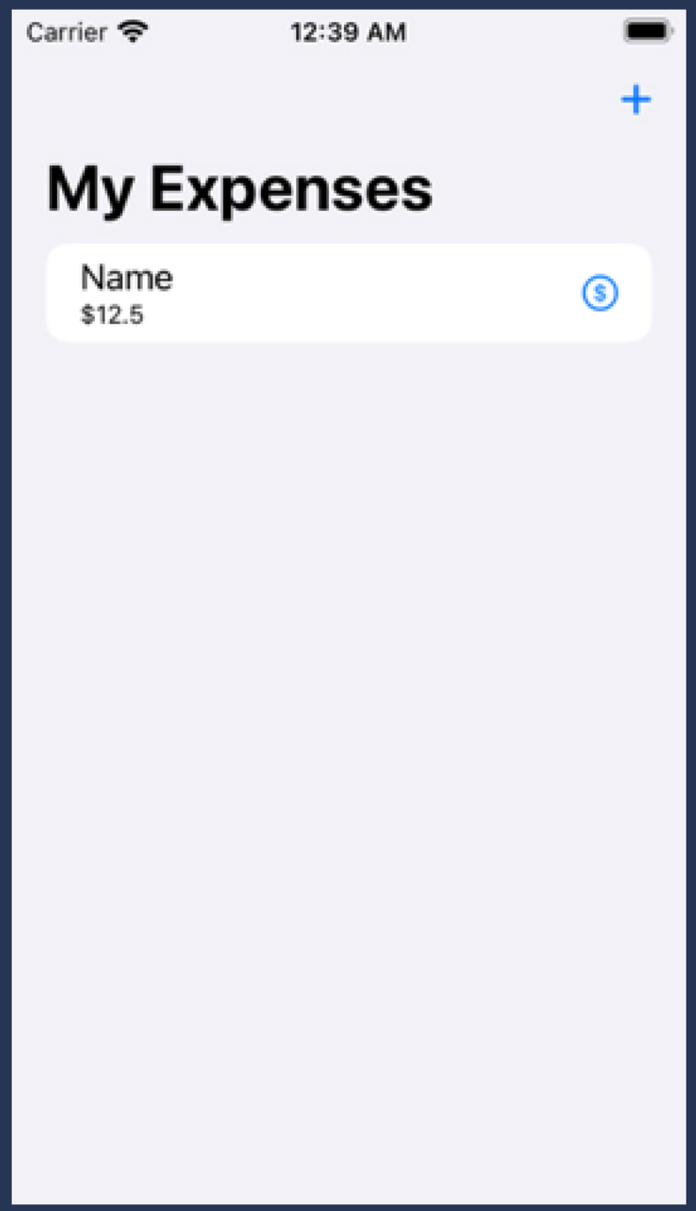
.listStyle()

```
List {
    Text("Row 1")
    Text("Row 2")
}
.listStyle(
    InsetGroupedListStyle()
)
```



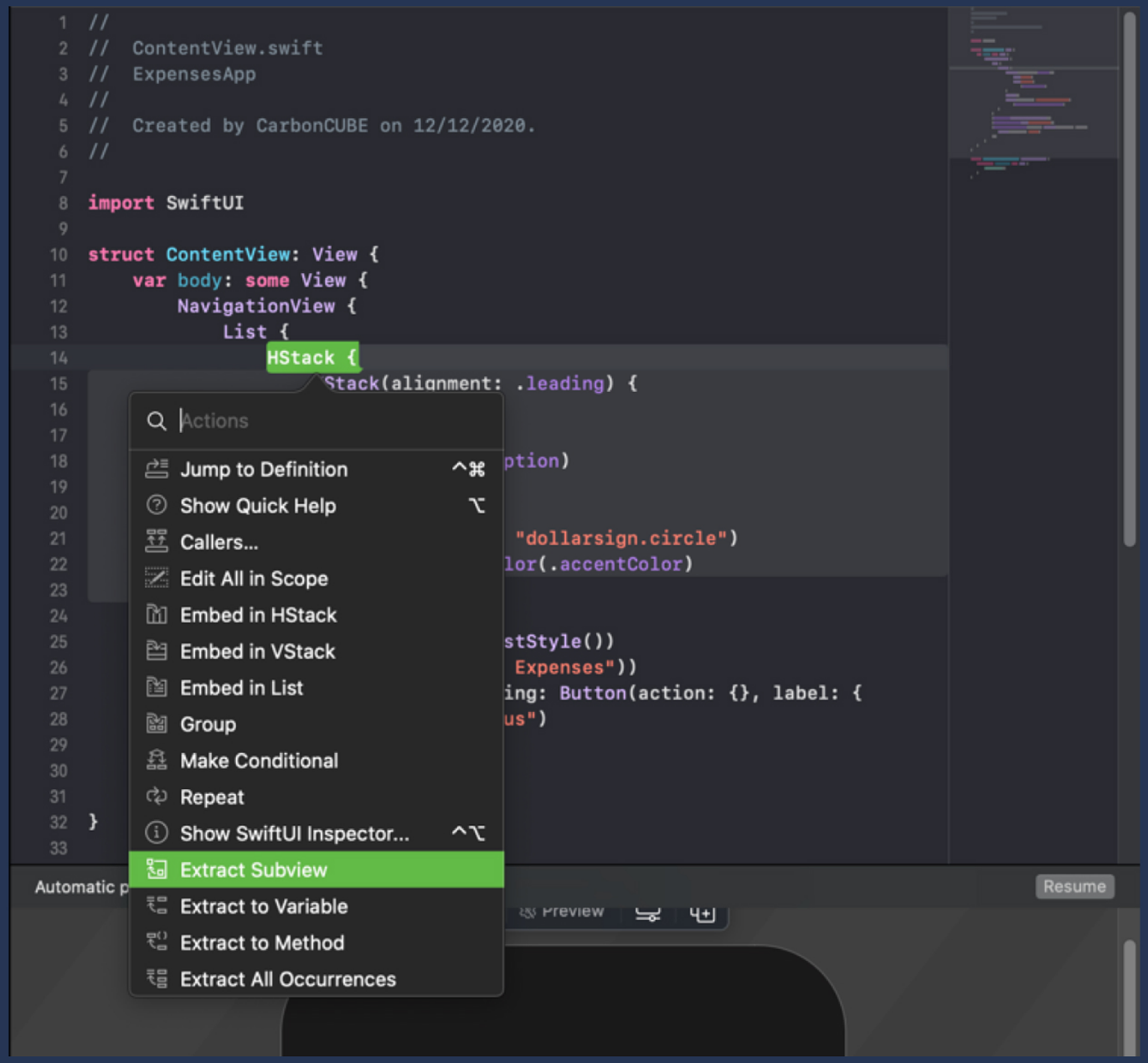
Row

```
List {
  HStack {
      VStack(alignment: .leading) {
          Text("Name")
          Text("$12.5")
              .font(.caption)
      }
      Spacer()
      Image(
        systemName: "dollarsign.circle"
      )
          .foregroundColor(.accentColor)
  }
}
```



Extract Sub View

```swift
//
//  ContentView.swift
//  ExpensesApp
//
//  Created by CarbonCUBE on 12/12/2020.
//

import SwiftUI

struct ContentView: View {
    var body: some View {
        NavigationView {
            List {
                HStack {
                    VStack(alignment: .leading) {
                        Text("Name")
                        Text("$12.5")
                            .font(.caption)
                    }
                    Spacer()
                    Image(systemName: "dollarsign.circle")
                        .foregroundColor(.accentColor)
                }
            }
            .listStyle(InsetGroupedListStyle())
            .navigationTitle(Text("My Expenses"))
            .navigationBarItems(trailing: Button(action: {}, label: {
                Image(systemName: "plus")
            }))
        }
    }
}

```


//
// ContentView.swift
// ExpensesApp
//
// Created by CarbonCUBE on 12/12/2020.
//
import SwiftUI
struct ContentView: View {
var body: some View {
NavigationView {
List {
HStack {
Stack(alignment: .leading) {
ption)
"dollarsign.circle")
lor(.accentColor)
stStyle())
Expenses"))
ing: Button(action: {}, label: {
us")
}
Actions
Jump to Definition
Show Quick Help
Callers...
Edit All in Scope
Embed in HStack
Embed in VStack
Embed in List
Group
Make Conditional
Repeat
Show SwiftUI Inspector...
Extract Subview
Extract to Variable
Extract to Method
Extract All Occurrences
Automatic p
Preview
Resume

ExpensesApp 〉 ExpensesApp 〉 ContentView.swift 〉 P body

```swift
//  Created by CarbonCUBE on 12/12/2020.
//

import SwiftUI

struct ContentView: View {
    var body: some View {
        NavigationView {
            List {
                ExtractedView()
            }
            .listStyle(InsetGroupedListStyle())
            .navigationTitle(Text("My Expenses"))
            .navigationBarItems(trailing: Button(action: {}, label: {
                Image(systemName: "plus")
            }))
        }
    }
}

struct ContentView_Previews: PreviewProvider {
    static var previews: some View {
        ContentView()
    }
}

struct ExtractedView: View {
    var body: some View {
        HStack {
            VStack(alignment: .leading) {
                Text("Name")
                Text("$12.5")
                    .font(.caption)
```

重覆使用 Sub View

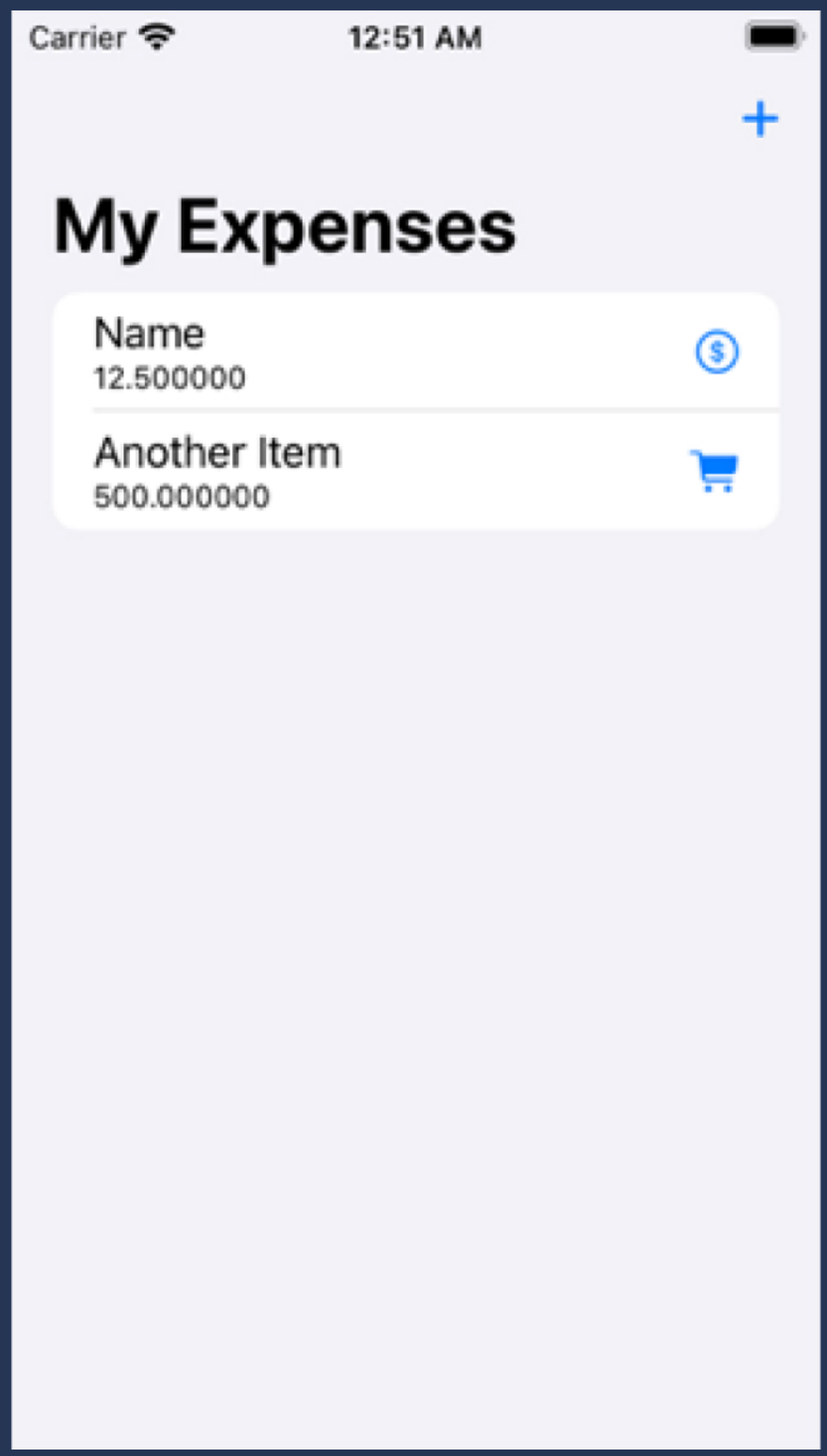
```
struct ExpenseRow: View {

    var name = ""
    var total = 0.0
    var icon = ""

    var body: some View {
        HStack {
            VStack(alignment: .leading) {
                Text("Name")
                Text("$12.5")
                    .font(.caption)
            }
            Spacer()
            Image(systemName: "dollarsign.circle")
                .foregroundColor(.accentColor)
        }
    }
}
```

重覆使用 Sub View

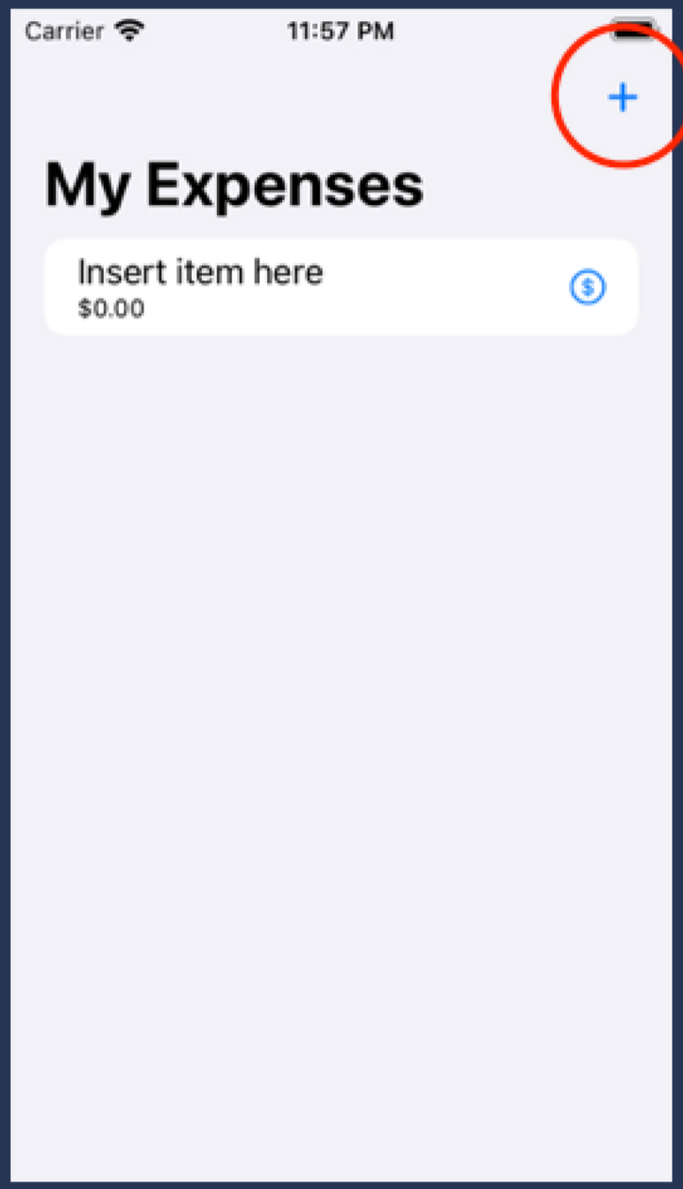
```
struct ExpenseRow: View {

    var name = ""
    var total = 0.0
    var icon = ""

    var body: some View {
        HStack {
            VStack(alignment: .leading) {
                Text("\(name)")
                Text("\(total)")
                    .font(.caption)
            }
            Spacer()
            Image(systemName: icon)
                .foregroundColor(.accentColor)
        }
    }
}
```

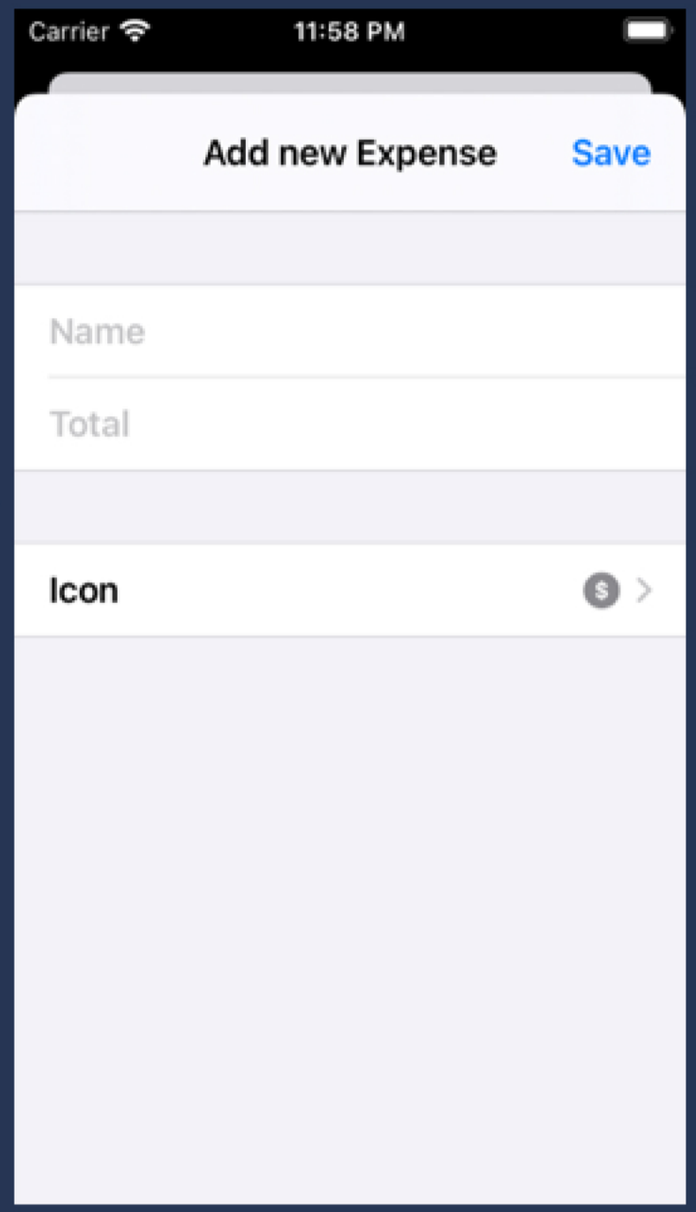
ExpenseRow

```
List {
    ExpenseRow(
        name: "Name",
        total: 12.5,
        icon: "dollarsign.circle"
    )
    ExpenseRow(
        name: "Another Item",
        total: 500,
        icon: "cart.fill"
    )
}
```

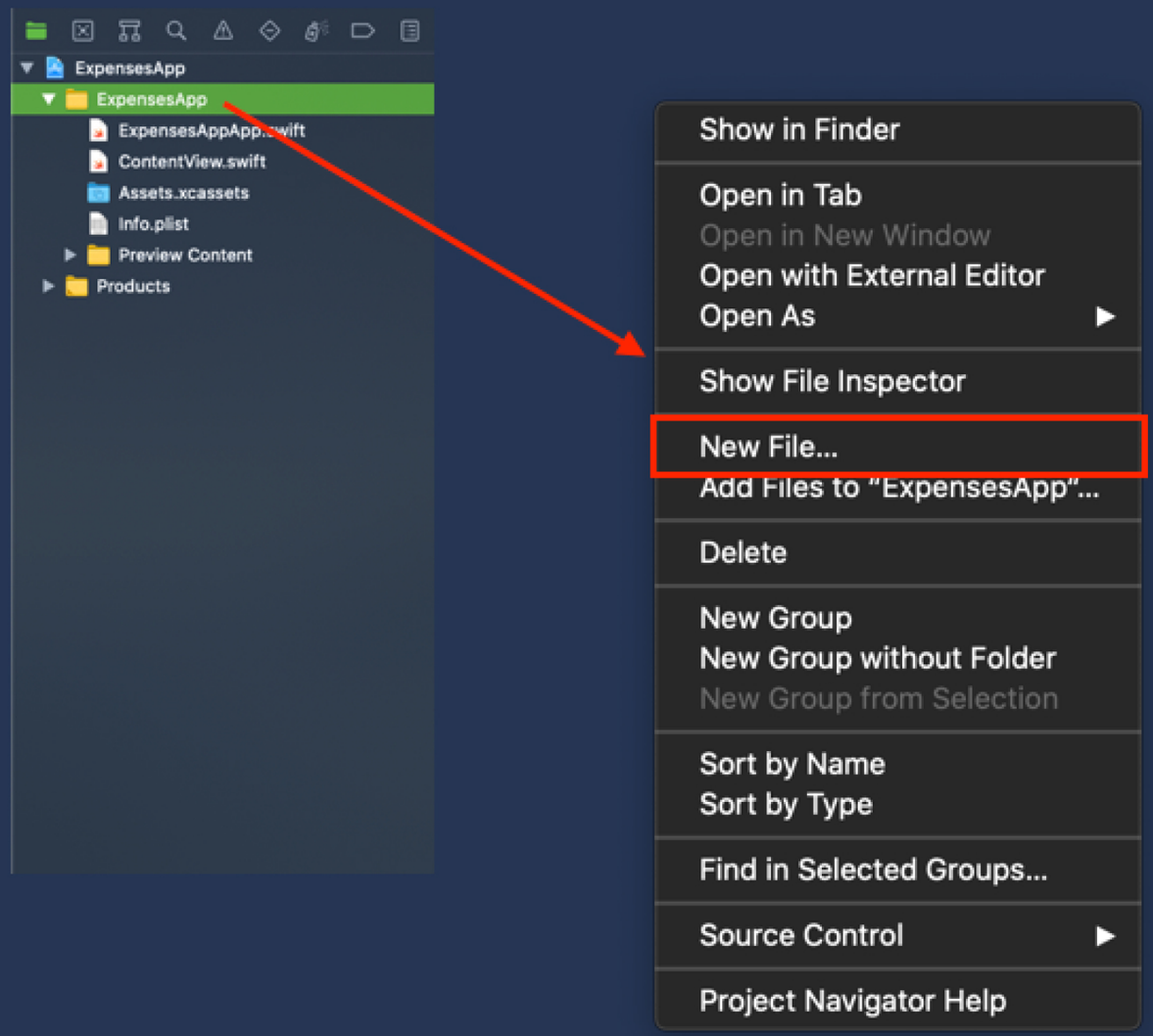


彈出新頁

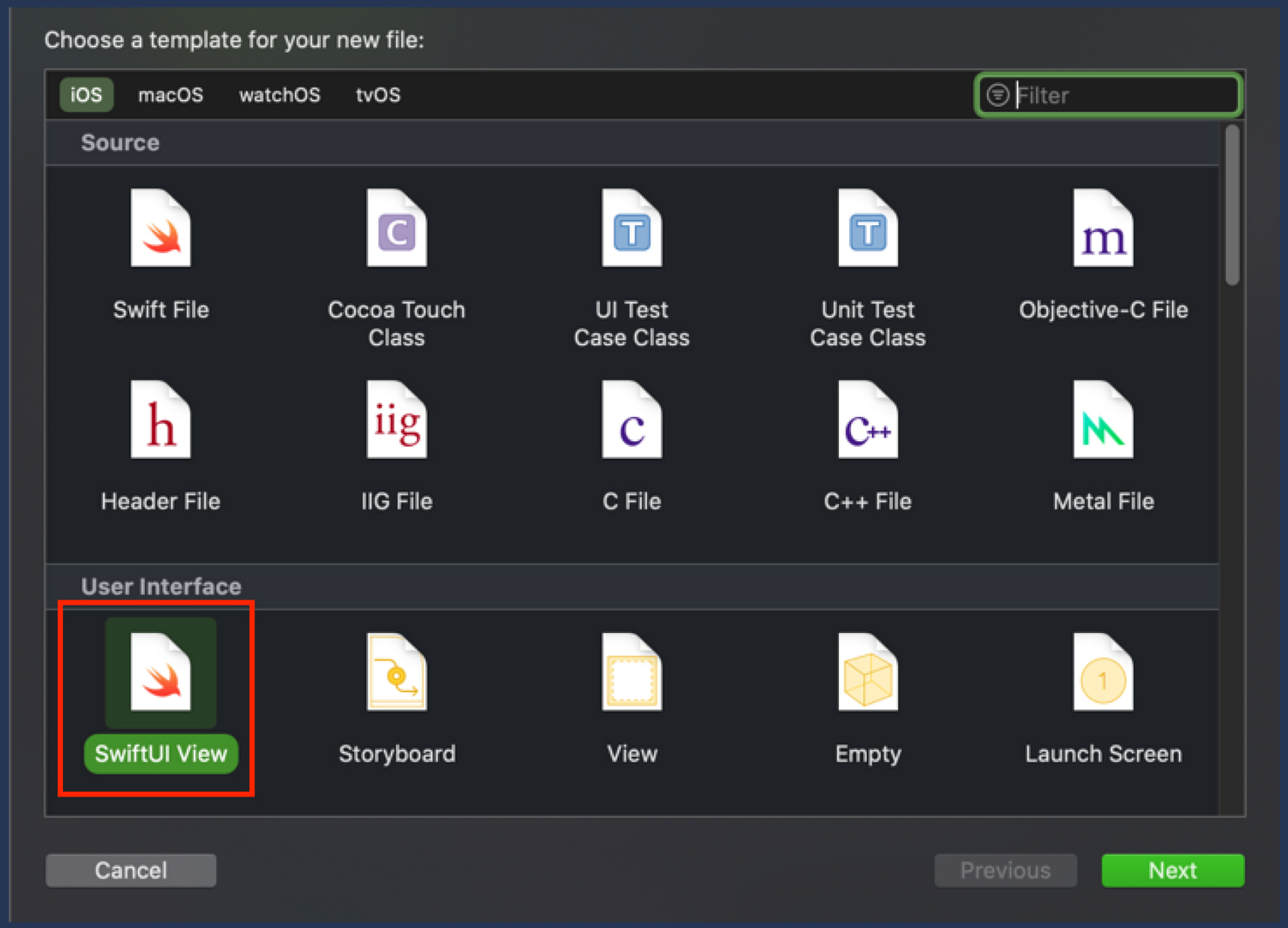

Carrier
11:57 PM
My Expenses
Insert item here
$0.00


Carrier
11:58 PM
Add new Expense
Save
Name
Total
Icon

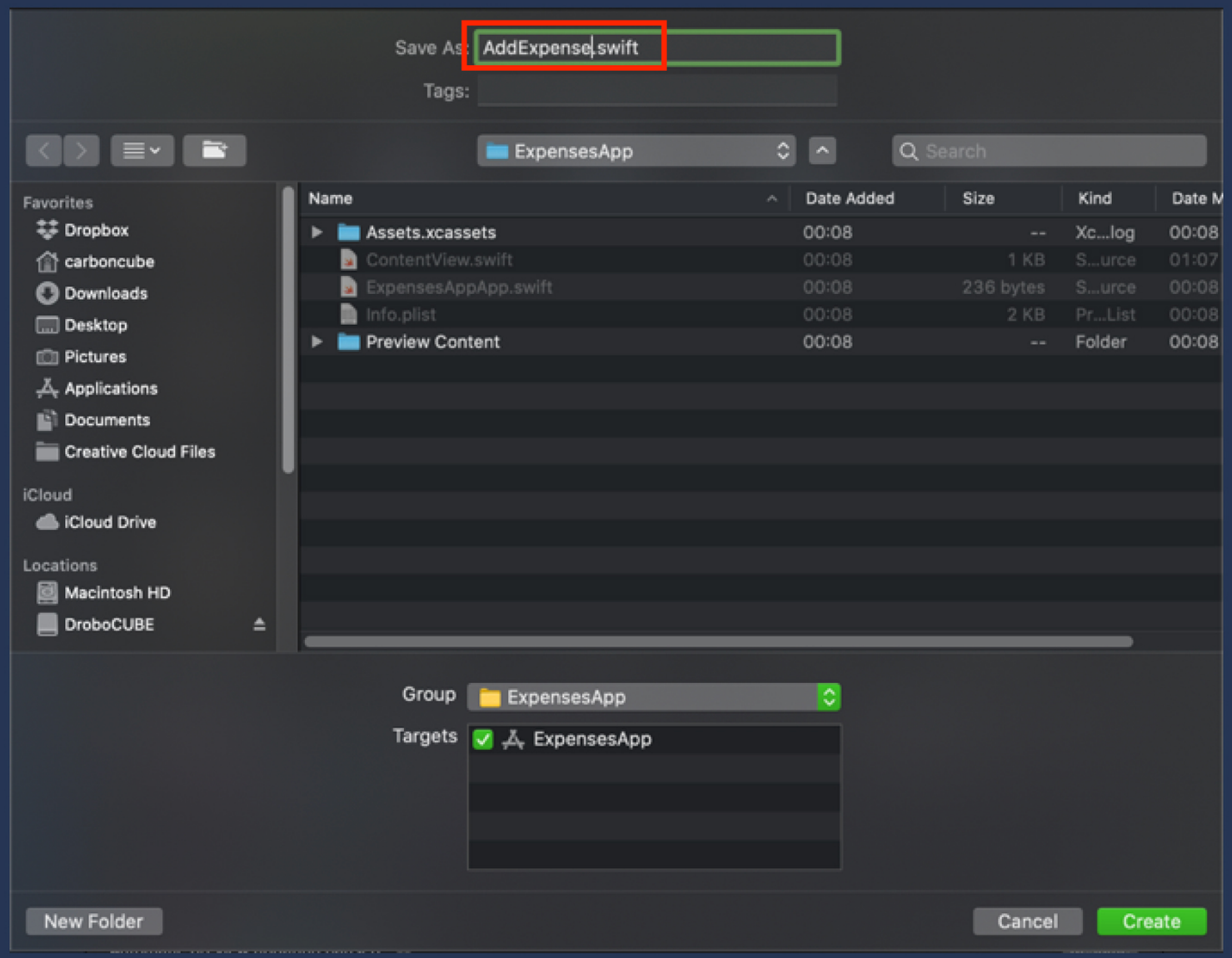
建立新一頁


ExpensesApp
ExpensesApp
ExpensesAppApp.swift
ContentView.swift
Assets.xcassets
Info.plist
Preview Content
Products
Show in Finder
Open in Tab
Open in New Window
Open with External Editor
Open As
Show File Inspector
New File...
Add Files to "ExpensesApp"...
Delete
New Group
New Group without Folder
New Group from Selection
Sort by Name
Sort by Type
Find in Selected Groups...
Source Control
Project Navigator Help

Swift UI View



Swift UI View



Sheet

```
// 定義控制 Sheet 顯示狀態的變量
@State var isShowingSheet = false

// ~~~

// 按鈕
Button(action: {
    isShowingSheet = true // 👈 按下按鈕時將變量設為 true
}, label: {
    Image(systemName: "plus")
})
    .sheet(isPresented: $isShowingSheet, content: {
        AddExpense() // 👈 希望彈出的 View
    })
```

Form

```
// 定義儲存的內容
@State var name = ""
@State var total = ""
@State var icon = ""

// ~~~

NavigationView {
    Form {
        Section {
            TextField("Name", text: $name)
            TextField("Total", text: $total)
                .keyboardType(.decimalPad) // 👈 將輸入方式變成數字 Keypad
        }
    }
    .navigationBarTitle(Text("Add Expense"), displayMode: .inline)
    .navigationBarItems(trailing: Button(action: {
        // 儲存的的動作
    }, label: {
        Text("Save")
    }))
}
```

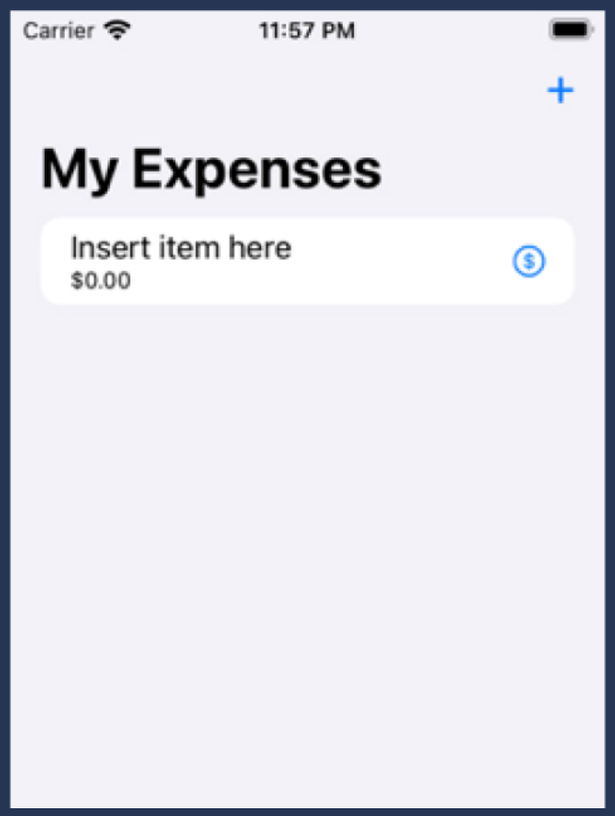
Picker

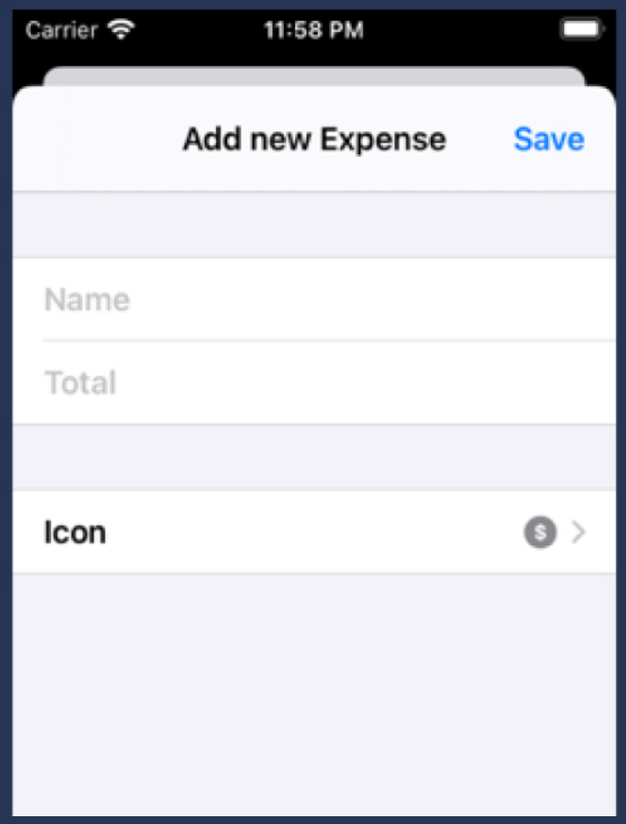
```
Section {
    Picker(selection: $icon, label: Text("Icon"), content: {
        Image(systemName: "dollarsign.circle.fill")
            .tag("") // 👈 .tag() 用於分辨選取的 Item
        Image(systemName: "dollarsign.square.fill")
            .tag("dollarsign.square.fill")
        Image(systemName: "bus.fill")
            .tag("bus.fill")
        Image(systemName: "car.fill")
            .tag("car.fill")
        Image(systemName: "gamecontroller.fill")
            .tag("gamecontroller.fill")
        Image(systemName: "cart.fill")
            .tag("cart.fill")
        Image(systemName: "book.fill")
            .tag("book.fill")
    })
}
```

實現功能

取得資料

記錄資料


Carrier
11:57 PM
My Expenses
Insert item here
$0.00


Carrier
11:58 PM
Add new Expense
Save
Name
Total
Icon

Expense Item

class ***ExpenseItem*** {

name	total	icon
Item	20.0	cart.fill

}

Expense Item

```
struct ExpenseItem {
    var name: String = ""
    var total: Double = 0.0
    var icon: String = ""
}
```

...只儲存一項資料

資料結構

使用 Array 儲存多個 *ExpenseItem*

Expenses

```
class Expenses : ObservableObject {
    @Published var items = [ExpenseItem]()
    // 👆當被標記為 @Published 的屬性有所改變，就會作出通知
}
```

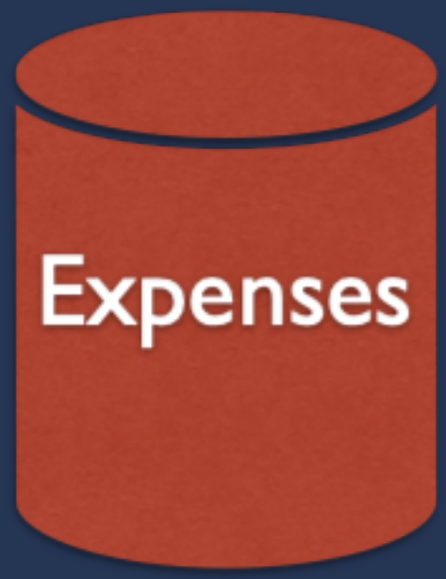
Expenses

```
struct ContentView: View {


    @State var isShowingSheet = false
    @ObservedObject var expenses = Expenses()
```

@ObservedObject

重點

- 容許在幾個 View 之間共通資料
- 標記 ObservableObject 可以被觀察
- 標記 @Published 的屬性被改變時，通知觀察人



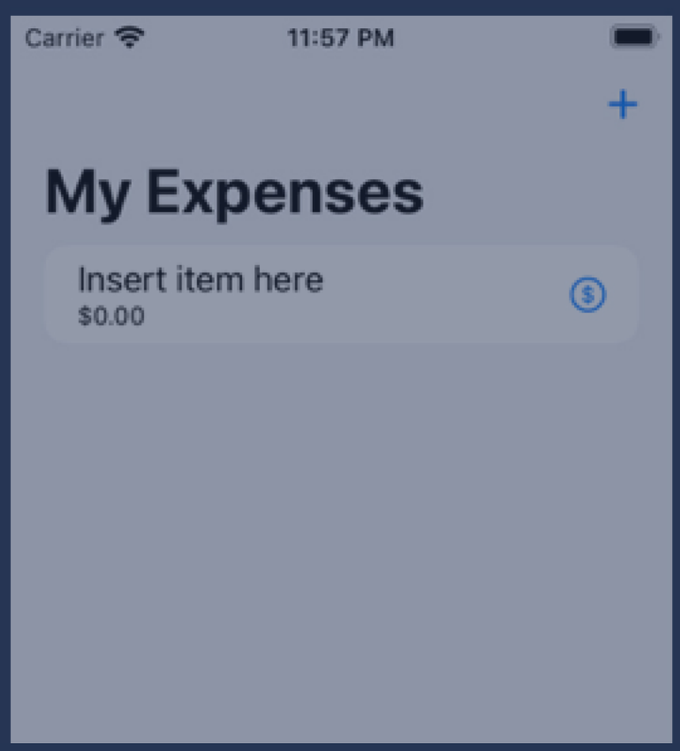
在 Array 加入新資料

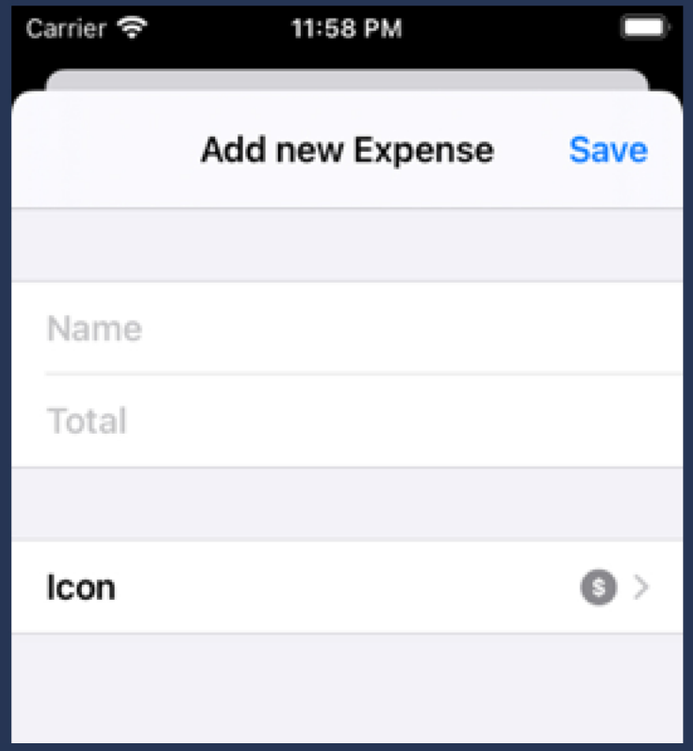
```
// 👇 item 為 expenses 的屬性, append 為 Array 的方法
expenses.items.append(
    ExpenseItem(
        name: "New Item",
        total: 100.0,
        icon: "dollarsign.square"
    )
    // 👆 加入一項 ExpenseItem
)
```

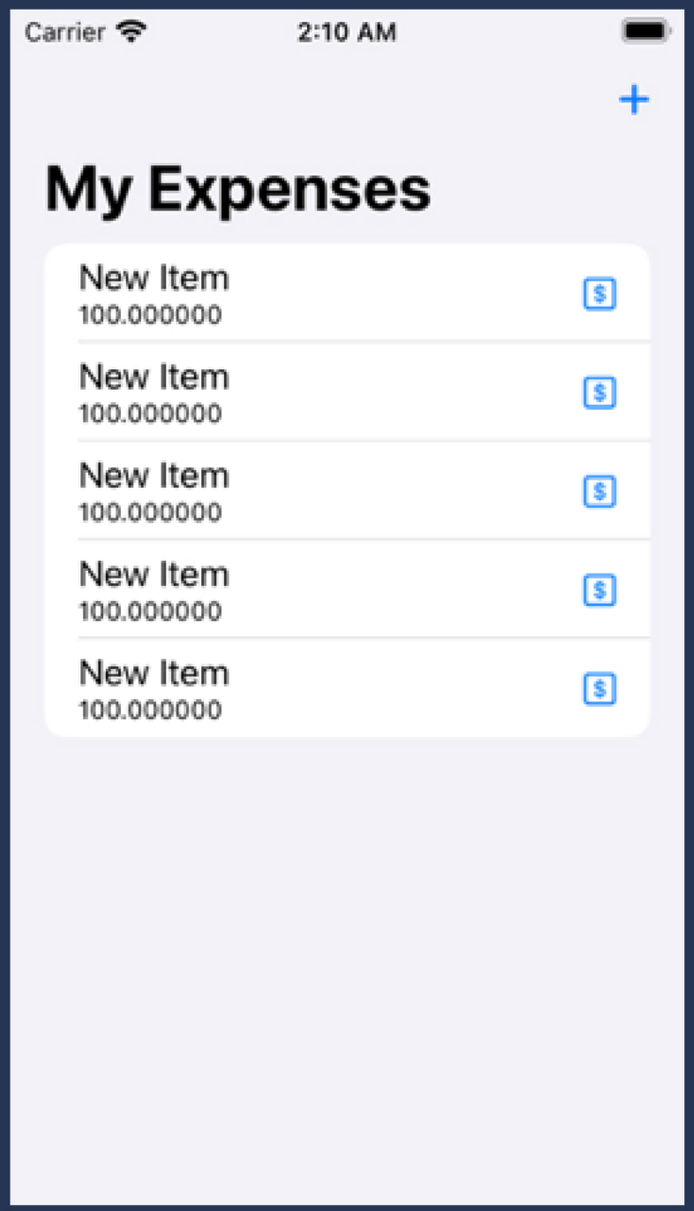
列表中顯示資料

```
ForEach(expenses.items){ item in
    ExpenseRow(name: item.name, total: item.total, icon: item.icon)
}

// ~~~
struct ExpenseItem : Identifiable {
    var id = UUID() // 👈 獨特的編號
    var name: String = ""
    var total: Double = 0.0
    var icon: String = ""
}
```

取得資料


Carrier
11:57 PM
My Expenses
Insert item here
$0.00


Carrier
11:58 PM
Add new Expense
Save
Name
Total
Icon


Carrier
2:10 AM
My Expenses
New Item
100.000000
New Item
100.000000
New Item
100.000000
New Item
100.000000
New Item
100.000000
New Item
100.000000

使 AddExpense 觀察相同的物件

```
struct AddExpense: View {
    @ObservedObject var expenses : Expenses
    // ...
```

在打開 AddExpense 頁時...

```
// ContentView.swift
Button(action: {
    isShowingSheet = true
}, label: {
    Image(systemName: "plus")
})
    .sheet(isPresented: $isShowingSheet, content: {
        AddExpense(expenses: expenses)
        // 👆 需要將同一個 expenses 傳遞
    })
```

儲存資料

```
// AddExpense.swift
if let totalValue = Double(self.total) {
    self.expenses.items.append(ExpenseItem(
        name: self.name,
        total: totalValue,
        icon: self.icon
    ))
}
```

@Environment

```
struct AddExpense: View {
    @Environment(\.presentationMode) var presentationMode
// ...

// ~~~
presentationMode.wrappedValue.dismiss()
// 👆 關閉 Sheet
```

接下來...

- 將資料儲存在電話
- 加入可以儲存的資料 (相片? 位置?)
- 利用資料進行統計

作業要求

1. 檔案命名：[你的全名].week5.zip
2. 檔案格式：必須將 Project 壓縮為 ZIP 檔後才上傳
3. 上傳至連結：https://www.dropbox.com/request/xl1nzJLl3MOW37bSHWiF
4. 到期日：2021 年 8 月 4 日 (下下週三) 晚上 11 時前
5. 評分標準：基本完成要求滿分、空白 Project 或欠交沒有分數。