struct CountViewContainer

```
let viewModel: Redux_SquadViewModel

func incrementWon() { viewModel.increment(.won) }

func incrementLoss() { viewModel.increment(.lost) }

var body: some View {
    HStack {
  CountView(count: viewModel.winCount,
    action: incrementWon)

  CountView(count: viewModel.lossCount,
    action: incrementLost)

   Button(...) { resetWonLost() }
  }
}
```

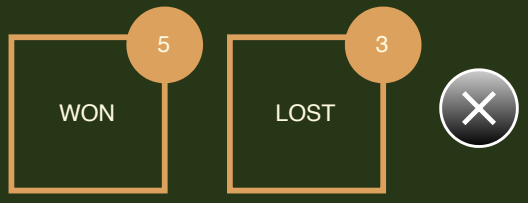
struct Redux_SquadViewModel

```
@Published private(set) var winCount: Int32 = 0

@Published private(set) var lossCount: Int32 = 0

let store: MyAppStore

let squadData: SquadData
```

```
func updateSquad(squadData: SquadData)

func increment(type: CountType)

func decrement(type: CountType)

- func updateCount(type: CountType, newCount: Int32)

buildViewProperties()
```

struct CountView

```
let action: () -> Void

let count: Int32

var body: some View {
  BannerButton<Int32>(bannerText: count,
    action: action)
}
```



enum CountType

```
case won

case lost
```

struct BannerButton<T>

```
let bannerText: T

let action: () -> Void
```