STRATEGY PATTERN FOR MULTI-ACTION QUICKVIEW

INTRO

- Marcin Wojciechowski
- Marcin Wojciechowski | LinkedIn
- Marcin Wojciechowski (@mgwojciech) / Twitter
- https://github.com/mgwojciech/

WHAT'S THE PROBLEM?

- With complex ACEs the onAction method can grow to a point it becomes difficult to manage
- Keeping big ifs/switch statements makes code difficult to follow and debug
- Logic in SPFx entry-point classes is difficult to isolate and test (QuickView.ts, WebPart.ts, Extension.ts)
- Adding additional logic gets more difficult with each code branch

HOW TO DO IT?

- Abstract QuickView to IQuickView as we will only case about state and setState
- Abstract action handler as we need two thing to handle any action
  - shouldHandle method
  - handleAction method
- Compose action handlers in ActionExecutor where we will call relevant action

DEMO TIME

WHY DO IT LIKE THIS?

+ Easy way to add new actions with zero impact on QuickView code
+ Easily isolated and testable without any need of mocking SharePoint dependencies
+ Handlers related to specific business action (pagination, sharing) can be logically grouped together