Building form

- remove CSS padding for subtitle
- replace title with "SAFE Demo", subtitle with "Score my talk @...", remove `safeComponents`
- add image (Level -> item -> Image -> img -> Src), 128x128, `Style [ Border "2px solid" ]`
- remove contents of `containerBox` and `show` function, add field helper function
- add comment (Textarea) and name (Input.text)
- add submit (Button.a), make it primary color + full width
- add scores: Level (ismobile) -> item -> button.a -> `Icon.faIcon [] [ Fa.icon Fa.I.SmileO ]`
- add 2x (`Fa.fa2x` to contents), color and outlined to button
- add function `scoreIcon`, add function `scoreColor`

Client side debugging

- change Model, Msg, init and update
- `Fable.Core.JsInterop`, `let onChange action = OnChange (fun e -> action !!e.target?value)`
- bind comment, name - **!!! use `DefaultValue` instead of `Value` !!!**
- bind score, `scoreColor` function: `IsWhite` for `None` and `Some s when s <> score`
- demonstrate client side debug - console, HMR, redux, react

Talking to server side

- add `Submit` to Msg, add `Loading` to Model, init, update
- bind submit button, disable all inputs when loading
- move `Score` to Shared, add `Vote` and `VotingProtocol` with `seeResults` types
- Server: `let votes = System.Collections.Concurrent.Bag<Vote>()`
- add `countVotes function` - **validate first**, `vote` async function with 1000 sleep
- server adapter: counter -> voting, client proxy: counter -> voting
- add `Results` of `Result<VotingResults,exn>` to model
- add `mkVote` function,
- `GotResults` to Msg, update, handle both Ok and Error
- add cmd | Submit `Cmd.ofAsync Server.api.vote (mkVote model') (Ok >> GR)...`
- `resultsBox` (empty), `formBox` and `containerBox` with pattern match
- fill out `resultsBox` -> copy from scores, but div instead of button
- add contents (small) for comments (quotes in italics), `Style [ TextAlign "left" ]`
- add "See results": getResults to protocol, Msg, update, results button

Deploy

- build.fsx: **change docker user and image name**
- Copy and adjust Deploy target (push imageFullName, add to chain)
- `build deploy` fast! copy image name
- create repository in docker hub, create web app in azure