SRE designing UI:
Agenda

- Get your hands off my CLI!!!
- Who are you users?
- GUIs are really not that hard…
- Architect with user in-mind from the start
- Build a toolbox of components
- Profit!
Whenever my Bash scripts aren't quite working, I say to add a few more quotes
And 99% of the time that fixes it. Somehow.
So what’s really wrong with the CLI?
First question - who is the user?
Well… I’m the user!
A Story of a successful tool

- I once needed to find something in a log file
- So I used `grep`
- Then I needed to find something in lots of log files
- So I used `grep -r`
- Then I needed to find something in lots of log files for service `x`
- So I used `find . -name "*x.log" -exec grep`
- Then I wanted to find something in lots of log files on lots of machines
- So I started a company, and called it Splunk
- PROFIT
The Step Before Starting A Company...
Your Tool Is Beloved!
But what if your tool works as intended… but has a bad interface?
So you say - “CLIs are much less frustrating than bad UIs”
Well… that is true sometimes, but most of the time, a very small group of people actually know how to operate it.
CLI Limitations

- Interface is not clear
- Is it `-help` or `--help` or `--?` or `-?` or `help` or `<blank>`
- What about “non technical users”? They do exist! They could be your manager, or your manager’s manager.
- What is the input format?
- What is the output format?
- Where is the progress indication?
- How do I know when flags are mutually exclusive?
You can build bad UIs as well, but the issues are more glaring and there are less places where to hide
UIs are not that hard, I promise
State of the world

- Focus on Web UIs, they also work on mobile if done right
- Prototype & Connect Pieces Quickly: *AngularJS*
- Make it look pretty - easily: *Bootstrap*
- Keep your backend
- Template UI - Closure Templates aka Soy:
Basic UI Concepts

- Minimize Clutter - Break information into palatable pieces
- Flow - Top-Down & Left-to-Right (or Right-to-Left)
- Specific set of choices in drop-downs or radio button
- Free text input should be minimized
- Highlight important areas
- Minimize Text - Easy to read, large font
- Give your UI to a neighbor to test drive
What really matters is architecture
MVC According to Wikipedia

- **MODEL** updates and manipulates the **VIEW**.
- **VIEW** sees the **USER**.
- **CONTROLLER** uses the **MODEL**.
MVC According to @avrukin

View
- CLI
- UI
- Other Tool

Controller

Core Library

Model
- DB
- Files
- Queue
- Remote Data

View-Controller Interface
- Light Local Server or Remote Call
Your Toolbox
You can build a small set of reusable patterns and tools, don’t have to start from scratch every time.
Example: UI Component + Command Line Flag

- **UI Component**
  - One-of
  - Multi-Option 1
  - Multi-Option 2
  - Multi-Option 3

- **CLI Flags**
  - `> app --flag1=one-of --flag2=option1 --flag2=option2`

- **Generated From**
  - Flags Config
  - Library Call
  - RPC
  - CLI Flag Parser
  - CLI Flags
  - Flags
  - CLI Renderer
  - UI Renderer

- **1:1 Mapping**

- **Generates**
  - UI Renderer
  - Flags Validator
  - CLI Flag Parser

- **Parses**
  - Flags Config
Example: Deploying a binary to production across many servers
Progress Provider and Monitors - Deploy Image to Servers

- A built image is placed in storage - S3
- Trigger of image deployment
  - Note the number of machines to deploy to
  - For each deployed machine, publish status to shared db - (Progress Provider)
- CLI tool can monitor progress through RPC (Progress Monitor)
- UI can monitor progress through JSON API (Progress Monitor)
- Additional actions - Cancel & Rollback
  - CLI - ctrl-c
  - CLI - another console
  - UI - Big red button
Architecture is Generic
Progress Provider & Monitor Architecture

Job Runner Page
- Go!
- Cancel

Job Status Queue
- Job Executor
- Job Status Provider
- Job Status Updater
- Job Status DB

> start job
~ 1 / x finished
~ 2 / x finished
~ 3 / x finished
~ 4th failed
^c
Some Common Components - Use Architectural Patterns

- Progress indicator: Progress Provider + Progress Broadcaster
- File Parser
  - Configuration Parser + Viewer + Validator
  - Log Parser + Indexer + Searcher + Viewer
- Remote State: Process Viewer + System State + *Providers
- Dependency Resolution + Interconnection
- UI
  - Tabs
  - Input Configuration - Dropdowns, Radio, Checkbox
  - Fields - Typed - with validation
Thank You for Listening
Q & A