Building the Monorepo

kchau@microsoft.com
Flywheel Team

About Monorepos

- JS codebases have grown, and are maintained by **thousands of devs**
- Modularity is needed: **package as unit of modularity**
- Related packages are updated at the **same commit**
- Monorepo require a **Monorepo Management Stack**

Monorepo Management Stack

1. Workspace-enabled package manager

- **Installs dependencies** for all packages
- **Links internal packages** to satisfy the node resolution algorithm
- Handles **dependency resolution** for all packages
- Optional: **hoisting**, **strictness** enforcement (phantom deps)
- On the market: `yarn`, `pnpm`, `rush`, `lerna + npm`

2. Task scheduler & runner

- **Runs npm scripts** for all packages
- **Optimize** task run speeds at the dev machine and CI
- Optionally in **topological** order or in **parallel**
- On the market: `lerna`, `wsrun`, `rush`, `pnpm recursive`, `lage`

3. Package publish tool

- Automated management of **semver**
  - Change description files or commit messages
- **Validation** of description of changes
- **Synchronize versions** between npm registry and git repository
- Automated **changelog creation**
- On the market: `rush`, `lerna`, `semantic-release`, `beachball`

Our Focus: Task scheduler & runner

Problem statement

Create a task runner that optimizes package tasks in a monorepo for a single machine

Current state

- JS monorepos in the wild run with **all kinds of workspaces**
- The state-of-the-art monorepo task runners are **not optimized**
- **CPU cores sit idle** for topological scripts
- Large monorepos generally have **clustered graph** of related packages

Philosphy

- Distribute work via smaller libraries with multiple owners
- Leverage OSS as much as possible
- Support package.json scripts as the script runner

Requirements of a task runner

- **Open sourced**
  - easily shared, public development demands **polish**
  - **easily contributed** to by many groups
- Works with all workspace implementations
- Easy setup
- Minimize idle CPU cores
- **Sublinear increase** in build time per package

Prior Art

- This should sound familiar because Vincent made a version for Midgard

Lage

“ *v. to make (Norwegian); pr. LAH-geh* ”

- Open sourced: https://github.com/microsoft/lage
- Easy to integrate with existing codebase
- Scales up with pipelining
- Scales out with caching and scoping

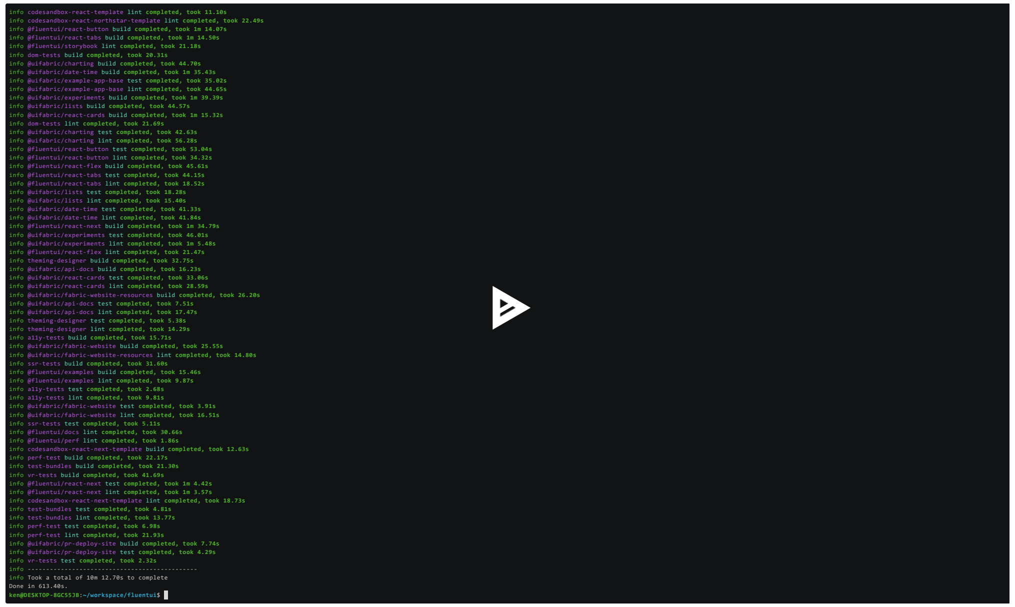
Collaboration

- **OneDrive/SharePoint**: rush showed us incremental builds
- **Midgard**: backfill cache, task-scheduler
- **Flywheel**: pipeline config, workspace-tools, `lage` tool
- **FluidX**: p-graph promise graph that supports priority queuing

What does it look like?

Full Build

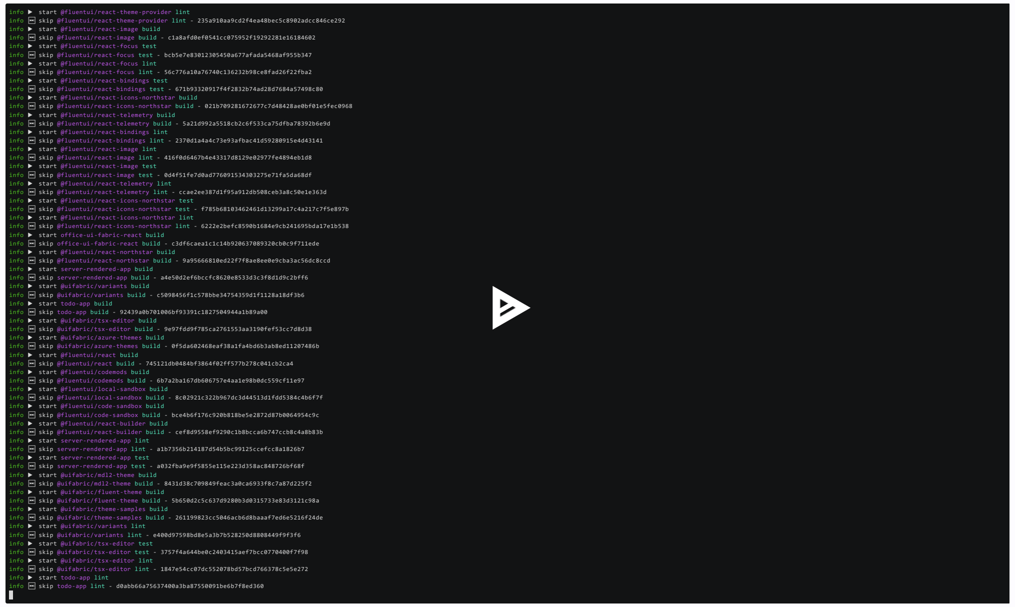
```
$ lage build test lint --grouped --verbose --reset-cache
```



```
info codesandbox-react-template lint completed, took 11.10s
info codesandbox-react-northstar-template lint completed, took 22.49s
info @fluentui/react-button build completed, took 1m 14.07s
info @fluentui/react-tabs build completed, took 1m 14.50s
info @fluentui/storybook lint completed, took 21.18s
info dom-tests build completed, took 20.31s
info @uifabric/charting build completed, took 44.70s
info @uifabric/date-time build completed, took 1m 35.43s
info @uifabric/example-app-base test completed, took 35.02s
info @uifabric/example-app-base lint completed, took 44.65s
info @uifabric/experiments build completed, took 1m 39.39s
info @uifabric/lists build completed, took 44.57s
info @uifabric/react-cards build completed, took 1m 15.32s
info dom-tests lint completed, took 21.69s
info @uifabric/charting test completed, took 42.63s
info @uifabric/charting lint completed, took 56.28s
info @fluentui/react-button test completed, took 53.04s
info @fluentui/react-button lint completed, took 34.32s
info @fluentui/react-flex build completed, took 45.61s
info @fluentui/react-tabs test completed, took 44.15s
info @fluentui/react-tabs lint completed, took 18.52s
info @uifabric/lists test completed, took 18.28s
info @uifabric/lists lint completed, took 15.40s
info @uifabric/date-time test completed, took 41.33s
info @uifabric/date-time lint completed, took 41.84s
info @fluentui/react-next build completed, took 1m 34.79s
info @uifabric/experiments test completed, took 46.01s
info @uifabric/experiments lint completed, took 1m 5.48s
info @fluentui/react-flex lint completed, took 21.47s
info theming-designer build completed, took 32.75s
info @uifabric/api-docs build completed, took 16.23s
info @uifabric/react-cards test completed, took 33.06s
info @uifabric/react-cards lint completed, took 28.59s
info @uifabric/fabric-website-resources build completed, took 26.20s
info @uifabric/api-docs test completed, took 7.51s
info @uifabric/api-docs lint completed, took 17.47s
info theming-designer test completed, took 5.38s
info theming-designer lint completed, took 14.29s
info a11y-tests build completed, took 15.71s
info @uifabric/fabric-website build completed, took 25.55s
info @uifabric/fabric-website-resources lint completed, took 14.80s
info ssr-tests build completed, took 31.60s
info @fluentui/examples build completed, took 15.46s
info @fluentui/examples lint completed, took 9.87s
info a11y-tests test completed, took 2.68s
info a11y-tests lint completed, took 9.81s
info @uifabric/fabric-website test completed, took 3.91s
info @uifabric/fabric-website lint completed, took 16.51s
info ssr-tests test completed, took 5.11s
info @fluentui/docs lint completed, took 30.66s
info @fluentui/perf lint completed, took 1.86s
info codesandbox-react-next-template build completed, took 12.63s
info perf-test build completed, took 22.17s
info test-bundles build completed, took 21.30s
info vr-tests build completed, took 41.69s
info @fluentui/react-next test completed, took 1m 4.42s
info @fluentui/react-next lint completed, took 1m 3.57s
info codesandbox-react-next-template lint completed, took 18.73s
info test-bundles test completed, took 4.81s
info test-bundles lint completed, took 13.77s
info perf-test test completed, took 6.98s
info perf-test lint completed, took 21.93s
info @uifabric/pr-deploy-site build completed, took 7.74s
info @uifabric/pr-deploy-site test completed, took 4.29s
info vr-tests test completed, took 2.32s
info ---------------------------------------------------
info Took a total of 10m 12.70s to complete
Done in 613.40s.
ken@DESKTOP-8GC55JB:~/workspace/fluentui$
```

Cached Build

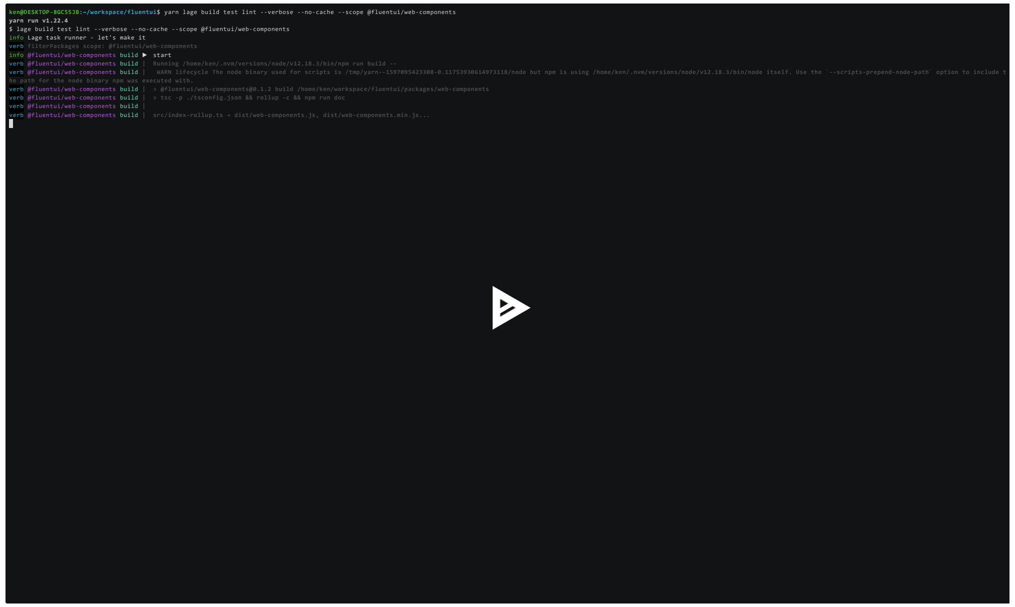
```
$ lage build test lint --grouped --verbose
```



```
info ▶ start @fluentui/react-theme-provider lint
info ⏭ skip @fluentui/react-theme-provider lint - 235a910aa9cd2f4ea48bec5c8902adcc846ce292
info ▶ start @fluentui/react-image build
info ⏭ skip @fluentui/react-image build - c1a8afd0ef0541cc075952f19292281e16184602
info ▶ start @fluentui/react-focus test
info ⏭ skip @fluentui/react-focus test - bcb5e7e83012305450a677afada5468af955b347
info ▶ start @fluentui/react-focus lint
info ⏭ skip @fluentui/react-focus lint - 56c776a10a76740c136232b98ce8fad26f22fba2
info ▶ start @fluentui/react-bindings test
info ⏭ skip @fluentui/react-bindings test - 671b93320917f4f2832b74ad28d7684a57498c80
info ▶ start @fluentui/react-icons-northstar build
info ⏭ skip @fluentui/react-icons-northstar build - 021b709281672677c7d48428ae0bf01e5fec0968
info ▶ start @fluentui/react-telemetry build
info ⏭ skip @fluentui/react-telemetry build - 5a21d992a5518cb2c6f533ca75dfba78392b6e9d
info ▶ start @fluentui/react-bindings lint
info ⏭ skip @fluentui/react-bindings lint - 2370d1a4a4c73e93afbac41d59280915e4d43141
info ▶ start @fluentui/react-image lint
info ⏭ skip @fluentui/react-image lint - 416f0d6467b4e43317d8129e02977fe4894eb1d8
info ▶ start @fluentui/react-image test
info ⏭ skip @fluentui/react-image test - 0d4f51fe7d0ad776091534303275e71fa5da68df
info ▶ start @fluentui/react-telemetry lint
info ⏭ skip @fluentui/react-telemetry lint - ccae2ee387d1f95a912db508ceb3a8c50e1e363d
info ▶ start @fluentui/react-icons-northstar test
info ⏭ skip @fluentui/react-icons-northstar test - f785b68103462461d13299a17c4a217c7f5e897b
info ▶ start @fluentui/react-icons-northstar lint
info ⏭ skip @fluentui/react-icons-northstar lint - 6222e2befc8590b1684e9cb241695bda17e1b538
info ▶ start office-ui-fabric-react build
info ⏭ skip office-ui-fabric-react build - c3df6caea1c1c14b920637089320cb0c9f711ede
info ▶ start @fluentui/react-northstar build
info ⏭ skip @fluentui/react-northstar build - 9a95666810ed22f7f8ae8ee0e9cba3ac56dc8ccd
info ▶ start server-rendered-app build
info ⏭ skip server-rendered-app build - a4e50d2ef6bccfc8620e8533d3c3f8d1d9c2bff6
info ▶ start @uifabric/variants build
info ⏭ skip @uifabric/variants build - c5098456f1c578bbe34754359d1f1128a18df3b6
info ▶ start todo-app build
info ⏭ skip todo-app build - 92439a0b701006bf93391c1827504944a1b89a00
info ▶ start @uifabric/tsx-editor build
info ⏭ skip @uifabric/tsx-editor build - 9e97fdd9f785ca2761553aa3190fef53cc7d8d38
info ▶ start @uifabric/azure-themes build
info ⏭ skip @uifabric/azure-themes build - 0f5da602468eaf38a1fa4bd6b3ab8ed11207486b
info ▶ start @fluentui/react build
info ⏭ skip @fluentui/react build - 745121db0484bf3864f02ff577b278c041cb2ca4
info ▶ start @fluentui/codemods build
info ⏭ skip @fluentui/codemods build - 6b7a2ba167db606757e4aa1e98b0dc559cf11e97
info ▶ start @fluentui/local-sandbox build
info ⏭ skip @fluentui/local-sandbox build - 8c02921c322b967dc3d44513d1fdd5384c4b6f7f
info ▶ start @fluentui/code-sandbox build
info ⏭ skip @fluentui/code-sandbox build - bce4b6f176c920b818be5e2872d87b0064954c9c
info ▶ start @fluentui/react-builder build
info ⏭ skip @fluentui/react-builder build - cef8d9558ef9290c1b8bcca6b747ccb8c4a8b83b
info ▶ start server-rendered-app lint
info ⏭ skip server-rendered-app lint - a1b7356b214187d54b5bc99125ccefcc8a1826b7
info ▶ start server-rendered-app test
info ⏭ skip server-rendered-app test - a032fba9e9f5855e115e223d358ac848726bf68f
info ▶ start @uifabric/mdl2-theme build
info ⏭ skip @uifabric/mdl2-theme build - 8431d38c709849feac3a0ca6933f8c7a87d225f2
info ▶ start @uifabric/fluent-theme build
info ⏭ skip @uifabric/fluent-theme build - 5b650d2c5c637d9280b3d0315733e83d3121c98a
info ▶ start @uifabric/theme-samples build
info ⏭ skip @uifabric/theme-samples build - 261199823cc5046acb6d8baaaf7ed6e5216f24de
info ▶ start @uifabric/variants lint
info ⏭ skip @uifabric/variants lint - e400d97598bd8e5a3b7b528250d8808449f9f3f6
info ▶ start @uifabric/tsx-editor test
info ⏭ skip @uifabric/tsx-editor test - 3757f4a644be0c2403415aef7bcc0770400f7f98
info ▶ start @uifabric/tsx-editor lint
info ⏭ skip @uifabric/tsx-editor lint - 1847e54cc07dc552078bd57bcd766378c5e5e272
info ▶ start todo-app lint
info ⏭ skip todo-app lint - d0abb66a75637400a3ba87550091be6b7f8ed360
```

Scoped Build

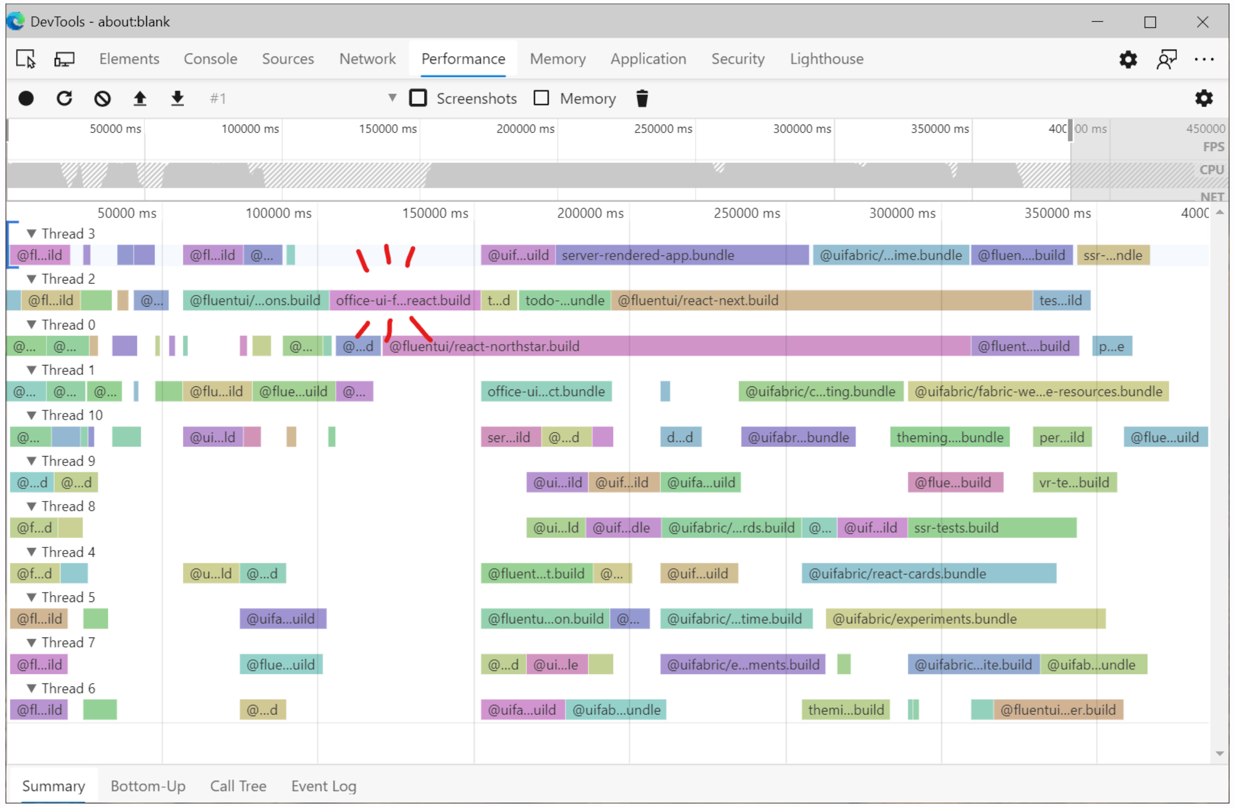
```
$ lage build test lint --grouped --verbose --scope @fluentui/web-components
```



```
ken@DESKTOP-8GC55JB:~/workspace/fluentui$ yarn lage build test lint --verbose --no-cache --scope @fluentui/web-components
yarn run v1.22.4
$ lage build test lint --verbose --no-cache --scope @fluentui/web-components
info Lage task runner - let's make it
verb filterPackages scope: @fluentui/web-components
info @fluentui/web-components build ▶ start
verb @fluentui/web-components build |  Running /home/ken/.nvm/versions/node/v12.18.3/bin/npm run build --
verb @fluentui/web-components build |   WARN lifecycle The node binary used for scripts is /tmp/yarn--1597095423308-0.11753930614973118/node but npm is using /home/ken/.nvm/versions/node/v12.18.3/bin/node itself. Use the `--scripts-prepend-node-path` option to include t
he path for the node binary npm was executed with.
verb @fluentui/web-components build |  > @fluentui/web-components@0.1.2 build /home/ken/workspace/fluentui/packages/web-components
verb @fluentui/web-components build |  > tsc -p ./tsconfig.json && rollup -c && npm run doc
verb @fluentui/web-components build |
verb @fluentui/web-components build |  src/index-rollup.ts → dist/web-components.js, dist/web-components.min.js...
```

Profiling

```
$ lage build test lint --profile
```



How does it work?

https://microsoft.github.io/lage/guide/levels.html

How try it at home?

https://microsoft.github.io/lage/guide/getting-started.html

1. npm scripts (build, test, lint) are at package level
2. `npx lage init`
   - creates a `lage.config.js` - configure it
   - adds `lage` as a dep
3. `yarn lage build` or `npm run lage build`

Future

- `lage` is a great solution for a **single machine**, not distributed builds
- For MSFT is buildxl
  - lage needs to spit out dscript or json config for buildxl
- Alternative: also investigate bazel
  - lage can potentially spit out WORKSPACE & BUILD

More info

Github:

https://github.com/microsoft/lage

Documentation:

https://microsoft.github.io/lage/

Complex Configuration:

https://github.com/microsoft/fluentui/blob/master/lage.config.js