

# Building the Monorepo

“

[kchau@microsoft.com](mailto: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` , `wsrn` , `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

# Philosophy

- 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: codecademy/react-template lint completed, took 11.10s
info: codecademy/react-orchestrator-template lint completed, took 12.09s
info: @pinstudio/react-button build completed, took 16.87s
info: @pinstudio/react-tabs build completed, took 16.79s
info: @pinstudio/storybook lint completed, took 21.18s
info: app-tests build completed, took 20.15s
info: @uiFabric/charting build completed, took 44.79s
info: @uiFabric/date-time build completed, took 19.43s
info: @uiFabric/example-app-base test completed, took 15.42s
info: @uiFabric/example-app-base lint completed, took 40.42s
info: @uiFabric/experiments build completed, took 16.10s
info: @uiFabric/lite build completed, took 46.71s
info: @uiFabric/react-carsa build completed, took 16.11s
info: app-tests lint completed, took 11.09s
info: @uiFabric/charting test completed, took 42.41s
info: @uiFabric/charting lint completed, took 50.19s
info: @pinstudio/react-button test completed, took 11.86s
info: @pinstudio/react-button lint completed, took 16.11s
info: @pinstudio/react-fire build completed, took 46.40s
info: @pinstudio/react-tabs test completed, took 44.11s
info: @pinstudio/react-tabs lint completed, took 19.27s
info: @uiFabric/lite test completed, took 18.28s
info: @uiFabric/lite lint completed, took 15.48s
info: @uiFabric/date-time test completed, took 41.31s
info: @uiFabric/date-time lint completed, took 42.58s
info: @pinstudio/react-next build completed, took 16.79s
info: @uiFabric/experiments test completed, took 46.40s
info: @uiFabric/experiments lint completed, took 16.10s
info: @pinstudio/react-fis lint completed, took 11.45s
info: thewing-designer build completed, took 11.71s
info: @uiFabric/api-docs build completed, took 10.23s
info: @uiFabric/react-carsa test completed, took 10.86s
info: @uiFabric/react-carsa lint completed, took 18.19s
info: @uiFabric/fabric-white-resources build completed, took 28.28s
info: @uiFabric/api-docs test completed, took 7.11s
info: @uiFabric/api-docs lint completed, took 17.47s
info: thewing-designer test completed, took 5.18s
info: thewing-designer lint completed, took 16.27s
info: ally-tests build completed, took 19.71s
info: @uiFabric/fabric-white-resources lint completed, took 14.89s
info: @uiFabric/fabric-white-resources test completed, took 16.86s
info: app-tests build completed, took 11.09s
info: @pinstudio/example build completed, took 15.44s
info: @pinstudio/example lint completed, took 9.87s
info: ally-tests test completed, took 2.44s
info: ally-tests lint completed, took 9.81s
info: @uiFabric/fabric-white test completed, took 3.81s
info: @uiFabric/fabric-white lint completed, took 16.11s
info: app-tests test completed, took 1.11s
info: @pinstudio/docs lint completed, took 18.44s
info: @pinstudio/pure lint completed, took 1.46s
info: codecademy/react-next-template build completed, took 12.61s
info: perf-test build completed, took 22.17s
info: text-handlers build completed, took 11.38s
info: app-tests build completed, took 40.40s
info: @pinstudio/react-next test completed, took 16.42s
info: @pinstudio/react-next lint completed, took 16.11s
info: codecademy/react-next-template lint completed, took 18.77s
info: text-handlers test completed, took 4.41s
info: text-handlers lint completed, took 19.71s
info: perf-test test completed, took 6.18s
info: perf-test lint completed, took 21.81s
info: @uiFabric/pr-deploy-lite build completed, took 7.78s
info: @uiFabric/pr-deploy-lite test completed, took 4.72s
info: app-tests test completed, took 2.12s
info:
info: Took a total of 104:12.79s to complete
Done in 104.10s.
/home/USER/.vscode/.workspace/fabricwhite
```



# Cached Build

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

```

1047 #start gfloatint/react:these-provider:link
1048 #map gfloatint/react:these-provider:link - 235a18b2d42f6e8b0c190210c846c292
1049 #map gfloatint/react:image-build
1050 #map gfloatint/react:image-build - 1a8af086f0941c875923f302281a1184802
1051 #start gfloatint/react:focus:best
1052 #map gfloatint/react:focus:best - 5b5e57b8238542e6d7afada5468a955b407
1053 #map gfloatint/react:focus:link
1054 #map gfloatint/react:focus:link - 56c776a18a7676b116232008c9eaf2d2729a2
1055 #map gfloatint/react:bindings:best
1056 #map gfloatint/react:bindings:best - 6703512037747c2812b42e87678a57488:0
1057 #start gfloatint/react:controller:build
1058 #map gfloatint/react:controller:build - 021a7091873277c774082a86e0f45cf9e058
1059 #map gfloatint/react:solamary:build
1060 #map gfloatint/react:solamary:build - 5a1d0992a015181c93518c37a0fa781203b6b6
1061 #map gfloatint/react:bindings:link
1062 #map gfloatint/react:bindings:link - 23781a2a6b73e73a5fa1c10280011e6d3141
1063 #map gfloatint/react:image:link
1064 #map gfloatint/react:image:link - 435f9686a07b63117c123a6203777e9d9b018
1065 #map gfloatint/react:image:best
1066 #map gfloatint/react:image:best - 8a0711c768d0776081515a102751715afcd6d0
1067 #start gfloatint/react:solamary:link
1068 #map gfloatint/react:solamary:link - c3c2ae387512f55a2058b1c6a510eac35d
1069 #start gfloatint/react:controller:best
1070 #map gfloatint/react:controller:best - 4780d8189494261132091b74e4217cf7e597b
1071 #map gfloatint/react:bindings:link
1072 #map gfloatint/react:bindings:link - 42232a3cf63789018d84b3a1695baf7e310b
1073 #start office-us-fabric:react:build
1074 #map office-us-fabric:react:build - c3f6a6a1c144922807328b10c8b3cf711e6
1075 #map gfloatint/react:controller:build
1076 #map gfloatint/react:controller:build - 9a955683382567886b3a4e16d8:0:0
1077 #start server-rendered-app:build
1078 #map server-rendered-app:build - 9e5822ef0cf678326a5131613f841012bfc
1079 #map server-rendered-app:build
1080 #map server-rendered-app:build - C598454cf15780b4a57453d6f112181b7304
1081 #start top-app:build
1082 #map top-app:build - 521a3ab702889593313170704544a1b9a0e
1083 #start Buifabric/fix-editor:build
1084 #map Buifabric/fix-editor:build - 9a076d60780a325841338f6c370d050
1085 #start Buifabric/scene:build
1086 #map Buifabric/scene:build - 076d4020d0e728cf4e6b018b3120700a
1087 #start gfloatint/react:build
1088 #map gfloatint/react:build - 92121a0b840cf188492cf979737b7841121cf
1089 #start gfloatint/scene:build
1090 #map gfloatint/scene:build - 0a31a5109080757a1eaf080b550cf1a07
1091 #start gfloatint/local-scene:build
1092 #map gfloatint/local-scene:build - 8a29212c32207b7c3a851f1cf45336a4ed7ff
1093 #start gfloatint/code-scene:build
1094 #map gfloatint/code-scene:build - 6a4e6d371c5286181bfce2875a4505454c6
1095 #start gfloatint/real-builder:build
1096 #map gfloatint/real-builder:build - cef4d5105e71298118c6a0747c4c8b4a83b
1097 #start server-rendered-app:link
1098 #map server-rendered-app:link - a3b7535a141870c8a0591251c9fc84a13267
1099 #start server-rendered-app:link
1100 #map server-rendered-app:link - a8129a9e7957851351233535e8a477b76d8
1101 #start Buifabric/real3:build
1102 #map Buifabric/real3:build - 8a3133b8708897f9e3a04933f67b7e7d2252
1103 #start Buifabric/float:scene:build
1104 #map Buifabric/float:scene:build - 9a528d2c5e737268018013573b3121310a
1105 #start Buifabric/scene:scene:build
1106 #map Buifabric/scene:scene:build - 2e11998231c584a6a6a77edde1211f2d6
1107 #start Buifabric/scene:link
1108 #map Buifabric/scene:link - e0a00d739b0a5a1b75128188884459f3f6
1109 #start Buifabric/fix-editor:link
1110 #map Buifabric/fix-editor:link - 13757464d4a0b3a04041a570cc7870a00776b
1111 #start Buifabric/fix-editor:link
1112 #map Buifabric/fix-editor:link - 18a7c34c01c152870a57b7c7e4637615e5272
1113 #start top-app:link
1114 #map Buifabric/fix-editor:link - dba0b6d676517408a3b4755080517b4e010
1115 #start top-app:link

```

# Scoped Build

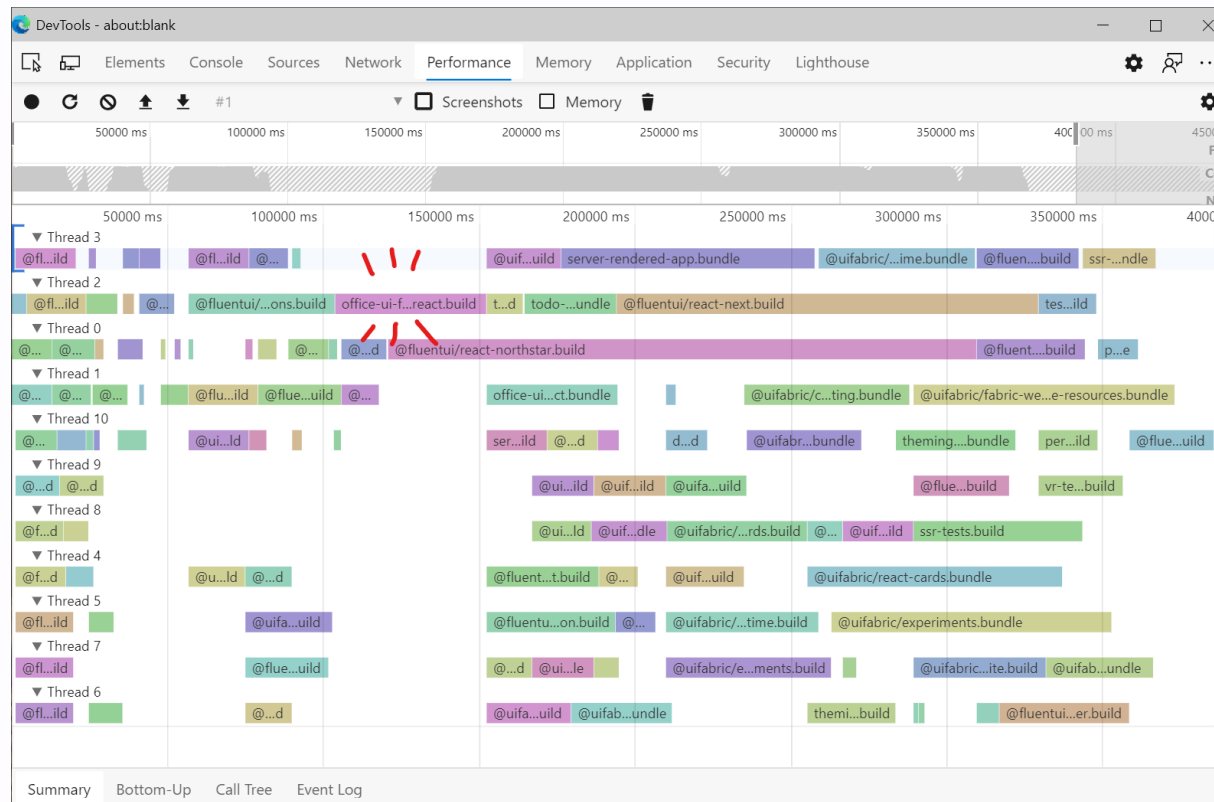
```
$ lage build test lint --grouped --verbose --scope @fluentui/web-components
```

```
~/workspace/fluentui$ npm run build test lint --verbose --no-cache --scope @fluentui/web-components
npm run v1.22.4
$ lage build test lint --verbose --no-cache --scope @fluentui/web-components
info lage test runner - let's make it
verb @fluentui/web-components build start
verb @fluentui/web-components build Running /home/ken/.npm/_nodedir/node/v12.18.1/bin/node run build --
verb @fluentui/web-components build with --script-path The node binary used for scripts is /home/ken/.npm/_nodedir/node/v12.18.1/bin/node itself. Use the --script-path option to include a
verb @fluentui/web-components build path for the node binary you are executing 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 - & tar -p ./fluentui_lint.sh && run run dev
verb @fluentui/web-components build - npm/run dev -- 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 to 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
- Microsoft solution is `buildxl`
  - `lage` spits out info for `buildxl` now!

```
lage info build --reporter json
```

- Another popular distributed build solution: `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>