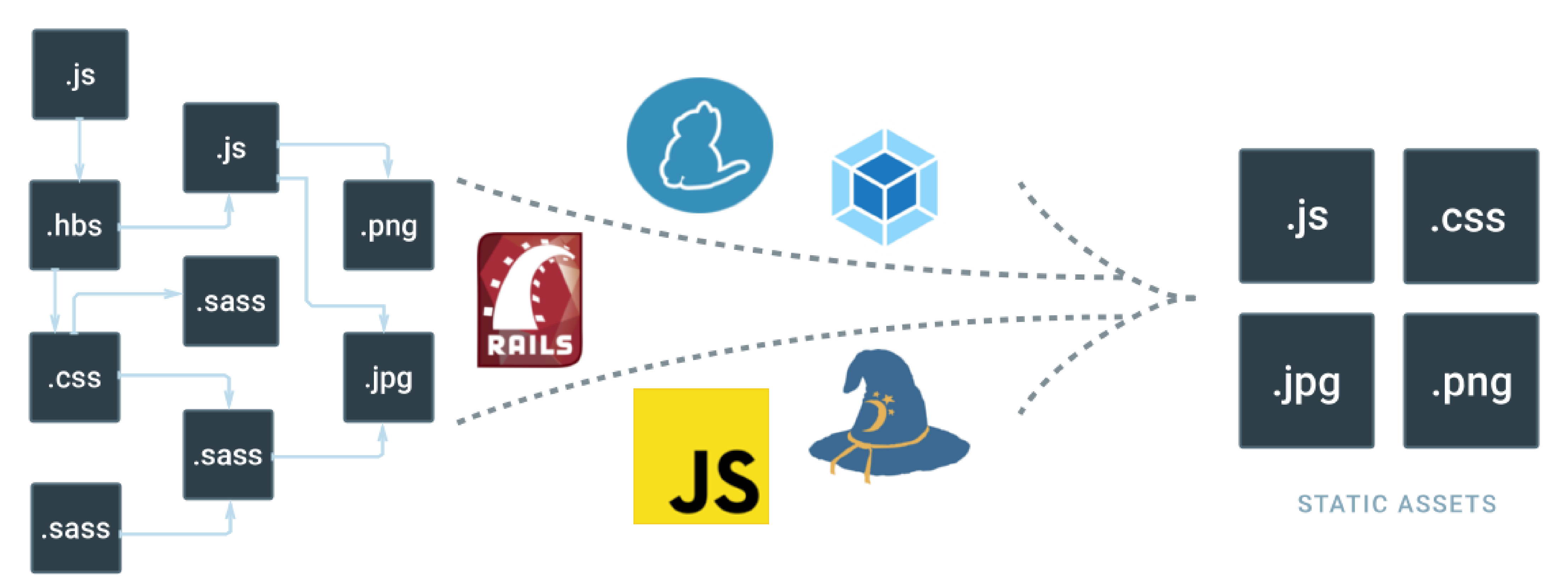
Updating and improving the build-chain and -tools

Introduction: What does a build-chain do?



https://jes.al/2017/03/incorporating-modern-javascript-build-tools-with-rails/

Analysis: How the current build-chain works

- source = .scss + .hbs + .js + media → build = .css + .html + .js + media
- grunt tasks: *build, test, measure, dist, cordova (build, run)*
    - Compile, autoprefix and minify sass
    - Check dependecies (bower/requirejs)
    - Optimize images
    - Copy files
    - Generate data (persons + context)
    - Create android package
    - Helpers for development (watch)

Evaluation: Problems with the build-chain?

- outdated packages
  - npm audit reports:

```
545 vulnerabilities (95 low, 311 moderate, 134 high, 5
critical) in 8967 scanned packages
357 vulnerabilities require semver-major dependency updates.
43 vulnerabilities require manual review
```

Package	Project Version	Latest Version
Nodejs	0.12	12.x
Cordova	5.3.1	9.0.0
Grunt	0.4.5	1.0.4
Grunt Packages	< 1.x	> 1.x or 2.x
jquery (dev)	2.1.4	3.4.1
jquery (dist)	1.8.2	3.4.1
Mocha	2.3.0	6.1.4

Evaluation: Problems with the build-chain?

- many different tools making it complex
  - 66 packages (without dependecies!): ruby: 6 | node: 49 | bower: 11
  - autoprefixer and composer??
  - grunt-scss-lint and ruby scss-lint??
  - bower: another package manager??

Evaluation: Problems with the build-chain?

- tools that seem to be used only rarely
  - compass/ruby (not used at all??)
  - modernizr (only no-touch?)
  - assemble (just 3 templates)

- abandoned projects
  - bower says: “we recommend using Yarn and Webpack or Parcel“
  - compass: deprecated
  - jquery: 1.x and 2. branches no longer receive patches

Evaluation: Problems with the build-chain?

- requirements on a local development machine
  - nodejs 0.12 (latest is 12.x)
  - ruby x.x?
  - android sdk 22
  - android studio (for templatefiles) x.x?
  - Java jdk 8 (latest is 12)
  - git
  - lots of little tweaks (windows paths, android list targets output ...)

Evaluation: Conclusion

- Its complex
- Its insecure
- Its error-prone

Consequences – How to improve?

- reduce complexity
  - consolidate
    - for example by using webpack = package manager and dependency manager → could replace bower and requirejs
    - could have a look in to parceljs (could replace grunt postcss …)
  - check for packages that are not used or not needed anymore
    - compass (modern browsers, autoprefixing)
    - ruby (only there for compass)
    - assemble (only 3 html files generated)

Consequences – How to improve?

- improve security
  - update packages → adapt sourcecode if necessary

- reduce error susceptibility
  - isolate development environment by using docker
  - optional: use CI/CD

Roadmap

1. migrate current build-chain to docker (1 week)
2. CI/CD (1 week)
3. analyze packages (2 weeks)
   1. check which packages are obsolete
   2. check which packages can be consolidated
   3. document what sourcecode changes will be needed

Milestone 2 - 17.06.2019

Roadmap

1. migrate bower/requirejs to webpack (1 week)
2. update packages, adapt sourcecode (1 week)
3. tests & documentation (1 week)
4. (optional: improve CI/CD if timetable allows it)

Milestone 3 - 08.07.2019