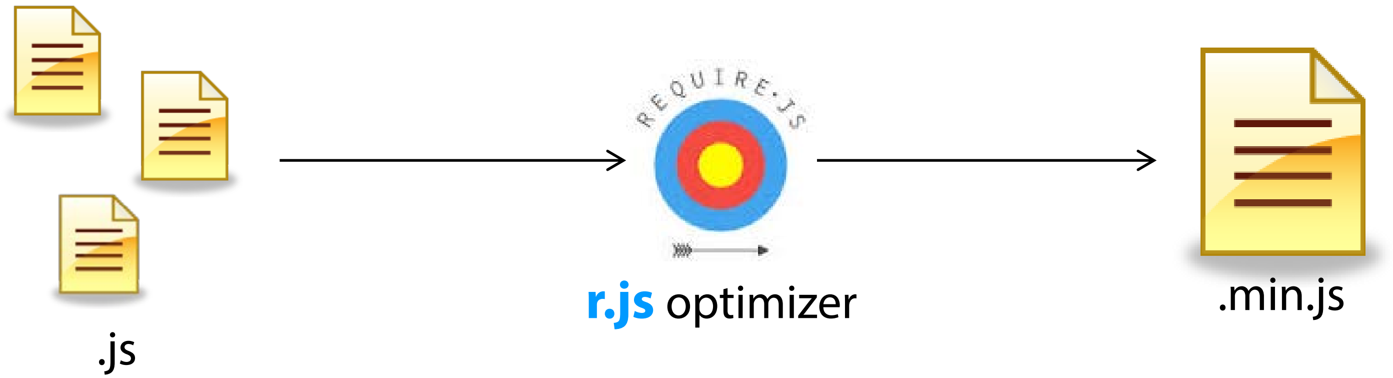
Optimization

Jeff Valore
CodingWithSpike.wordpress.com
@CodingWithSpike

pluralsight
hardcore dev and IT training



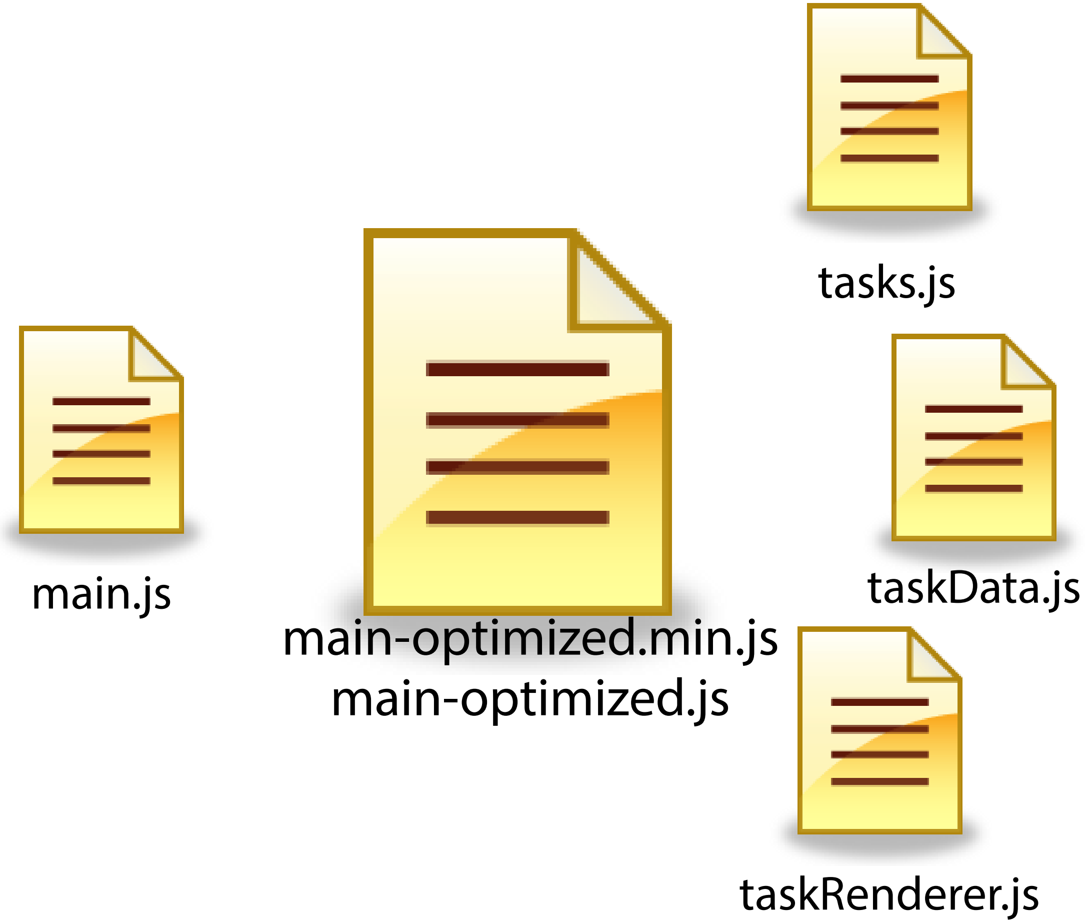
Development

Deployment

.js

r.js optimizer

.min.js



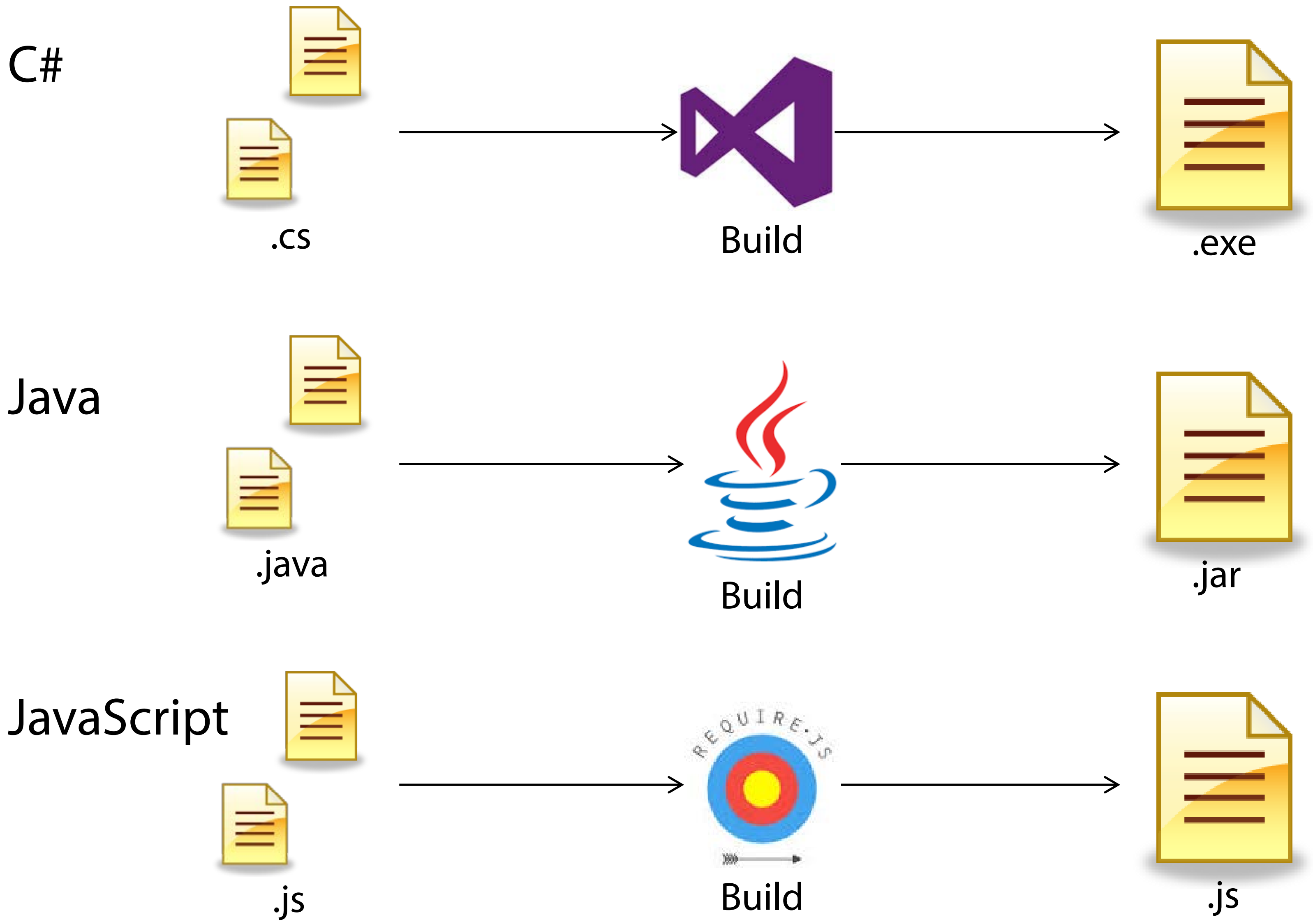
main.js

main-optimized.min.js
main-optimized.js

tasks.js

taskData.js

taskRenderer.js

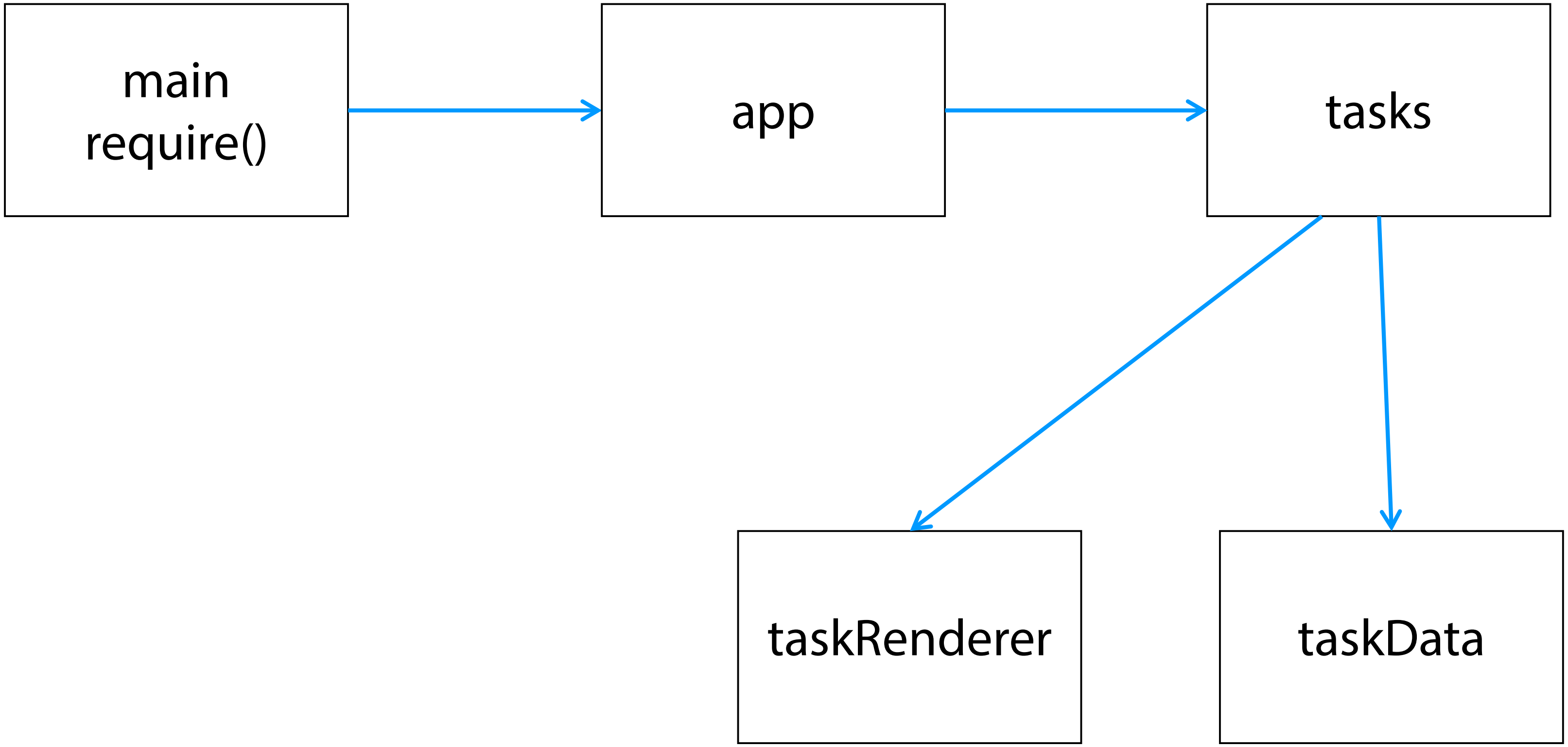


C# .cs → Build → .exe

Java .java → Build → .jar

JavaScript .js → Build → .js

Module Dependencies



main
require()

app

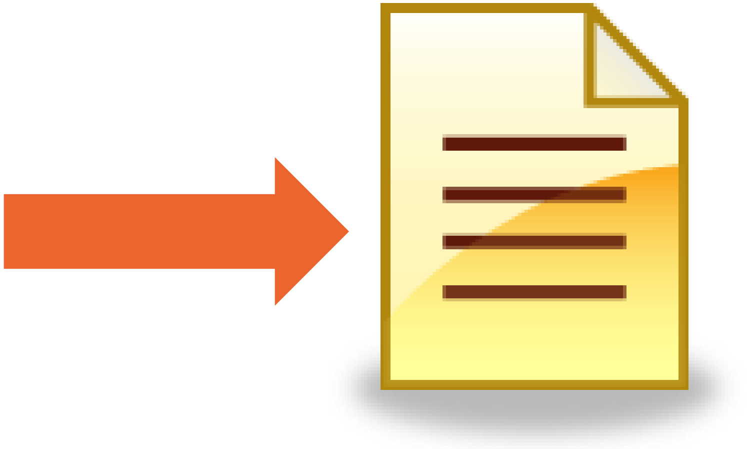
tasks

taskRenderer

taskData

Current r.js Command Line Arguments

```
node build\r.js –o
   name=main
   baseUrl=src/js
   mainConfigFile=src/js/main.js
   out=src/js/main-optimized.min.js
   generateSourceMaps=true
   preserveLicenseComments=false
   optimize=uglify2
```



Build Profile
(JavaScript file)

Options to Ease Debugging

1. **Use un-optimized files in development**

   Dev: <script src="js/require.js" data-main="js/main.js"></script>
   Prod: <script src="js/require.js" data-main="js/main-optimized.min.js"></script>

2. **Use source maps**

   Maps lines in compressed files back to their original source.
   Enable using generateSourceMaps=true option in r.js

Summary

- The r.js optimizer allows us to have multiple files for development and a single file for release.
- The r.js optimizer runs in Node.js.
- The optimizer can optionally minify the output files.
- Source maps ease debugging by mapping optimized and minified code to their original source lines.
- Optimization options can be passed to r.js on the command line or through a build profile file.