TRION

Modern Java Front-end development

Thomas Kruse

INTRODUCTION

TRION

- Thomas Kruse
  - Consultant (@ trion.de)
  - Leader JUG Münster
  - twitter - @everflux

SHOW CASE

TRION

Social App (not only) for JUGs

- Twitter
- Facebook
- …

- Keeping credentials centralized
- Timed publications
- …

- Showcase Netbeans ;-)
- And modern Java

Illustrate architecture decision making

SETTING

TRION

Environment

- More complex systems
- Mobile devices
- New major players / trend setters

Architecture requirements

- Decoupled front-end/back-end
- Low latency
- Manage front-end complexity
- Leverage the cloud

User expectations

- Low latency
- Cross-device
- Ease of use
- Time to market
- Long lasting systems

FRONTEND OPTIONS

TRION

Java Swing / JavaFX

- No production quality cross device
- Requires Java on user device
- Valid for special use cases

JSF

- Tight backend coupling
- High resource consumption
- R.I.P. !
- … same for Wicket, Vaadin, …

HTML5+ REST

- Server side rendering or …
- … HTTP API driven
- Ajax, Websockets, SSE, …
- Vanilla or Framework?
- jQuery: Already legacy

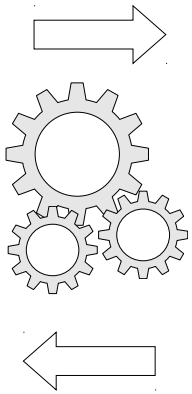
TOOLING OPTIONS

TRION

NetBeans IDE

IDE

- Code-completion
- Navigation
- Hints
- Round-trip integration
- Instant Live-reload
- Debugging / breakpoints
- gulp / grunt / npm



HTML5 diagnostic APIs

Chrome dev tools

Debugger

Browser

gulp / grunt / live-reload

DEMO

TRION

- Netbeans round-trip editing

INTEGRATION IMPLICATIONS

TRION

Front-End

- Finer grained
- Parallel
- Out-of-order
- Composition of multiple APIs
- Same API for M2M

Back-End

- Coarse grained (JSF, MVC)
- Sequential
- Single-Threaded
- Container services
  - Security
  - Transactions
- Separate API for M2M

PROJECT STRUCTURE

	Single project	Separate front end project
Dependency	• WebJars • Manual – add to SCM	• npm • bower
Build system	• Maven • Gradle	• grunt • gulp
Coupling	• Highly coupled • Pace of evolution mismatch	• Loosely coupled • Enables separate evolution • Encourages testability • Easy replacement of one part
Deployment	• Part of application • Simultaneous release	• Separate • Low requirements for plain HTML

CHOOSING A FRAMEWORK

TRION

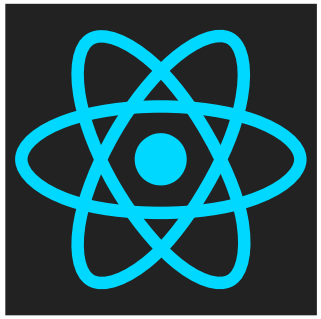
Why a front-end framework?

- jQuery is dead
- … and was never a framework
- Testability
- Ease of development
- Team scaling
- Even outsourcing

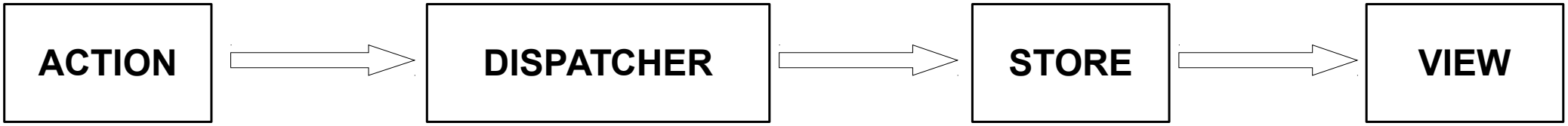
REACT + FLUX

TRION

- Facebook
  - … and others
- React.js
  - Library
  - (virtual) DOM manipulation
  - Unidirectional data-flow concept
  - Server-side rendering possible
- No framework!
  - HTTP Service - superagent



REACT/FLUX BLOCKS

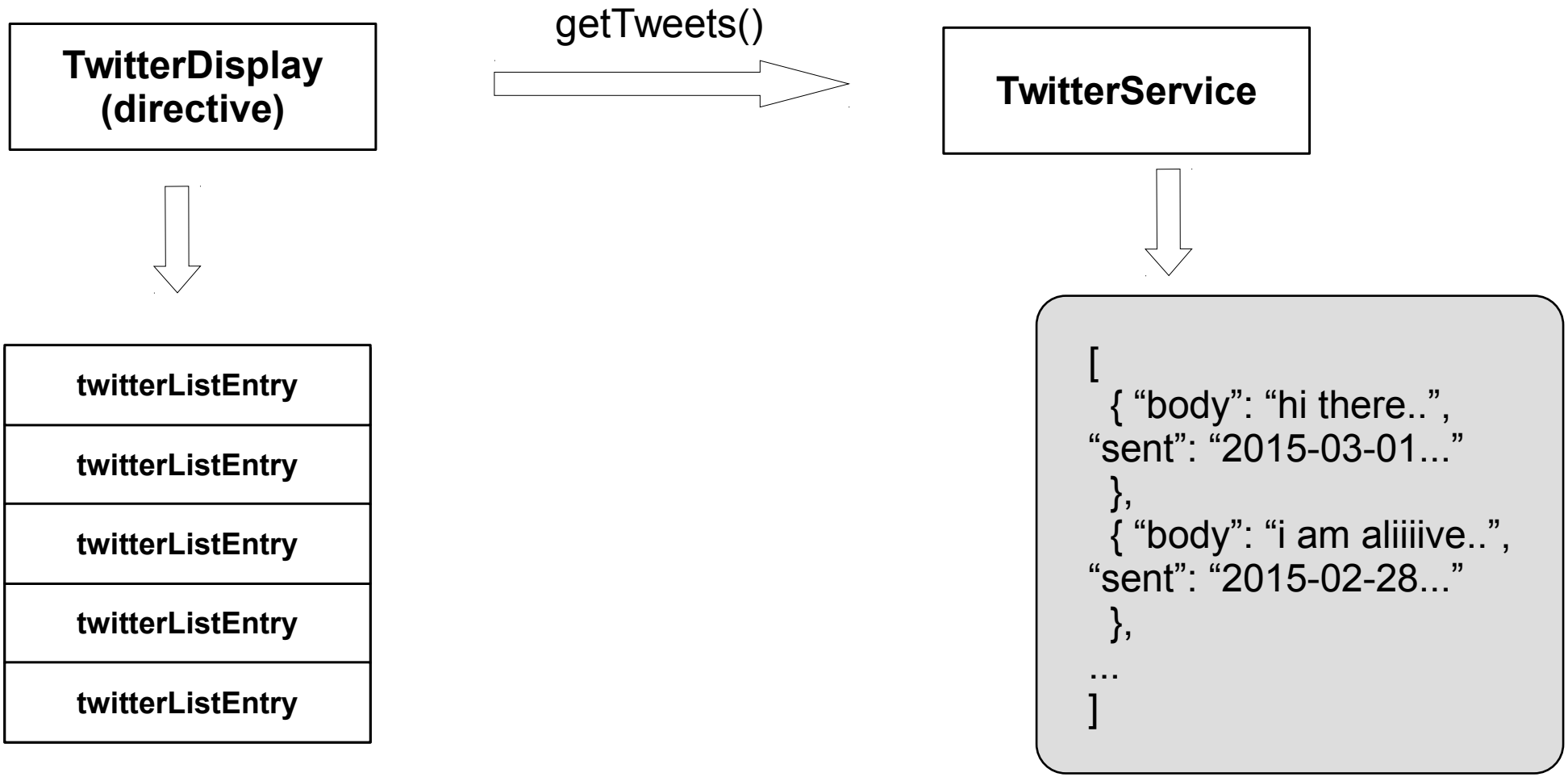


- View components
  - JSX (optional, no Netbeans support)
  - Properties (immut.) input
- Dispatcher
  - Receives actions, notifies callbacks
- Store
  - Container for state and callbacks
- Action
  - Passed to dispatcher (payload)

Angular

- Front-end framework
- Opinionated
- Dependency injection
- Model-View-ViewModel
- Two-way data binding
- Routing
- Directives, Services
- Java developers feel at home

ANGULAR DESIGN



DEMO

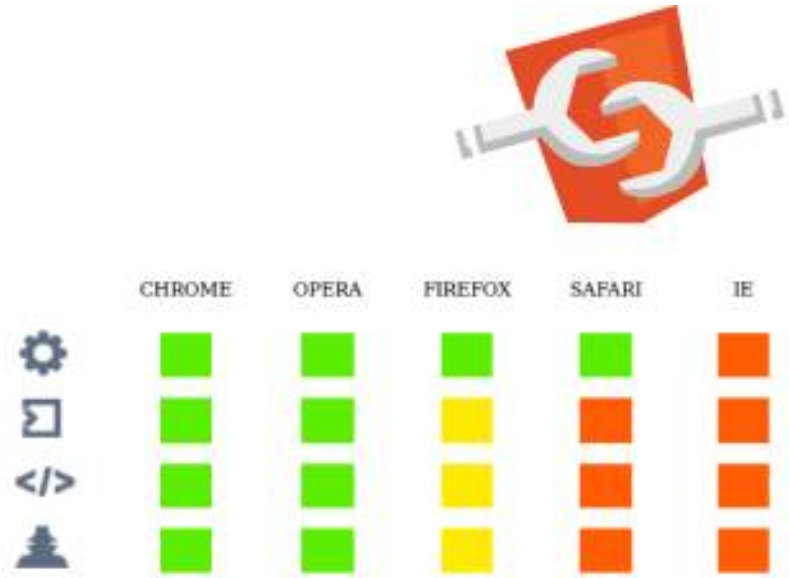
TRION

- Angular message service – fake or real

WEB COMPONENTS

TRION

- Mostly implemented across relevant platforms
- Elements
  - Templates
  - HTML import
  - Custom elements
  - Shadow DOM



- Foundation for libraries like x-tag / Polymer
  - Polyfill for IE / stable spec: Bosonic

DEMO

- Netbeans web-component editing

BACKEND LANGUAGE

Java

- Java EE
- Spring Boot
- dropwizard
- Fancy other options like vert.x

PHP / Ruby / Python

- Sometimes a good choice
- Consider tooling and operations

node / io.js

- All developers use JavaScript anyway
- Same language on all layers
- Consider evolution speed
- Avatar: Uncertain future

go

- Sweet spot for JSON micro services

OUR STACK

TRION

	Backend	Frontend
Tooling	• Netbeans IDE • curl	• Netbeans IDE • Chrome / Firefox
Build system/ Dependency mgmt	• Maven	• npm / bower • grunt
Frameworks	• Spring Boot • Spring Integration	• Foundation CSS • Angular • Angular Foundation
Runtime	• Tomcat	• Browser

BACKEND BOOTSTRAP

TRION

jhipster

- Single project
- Maven
- Spring Boot
- Angular

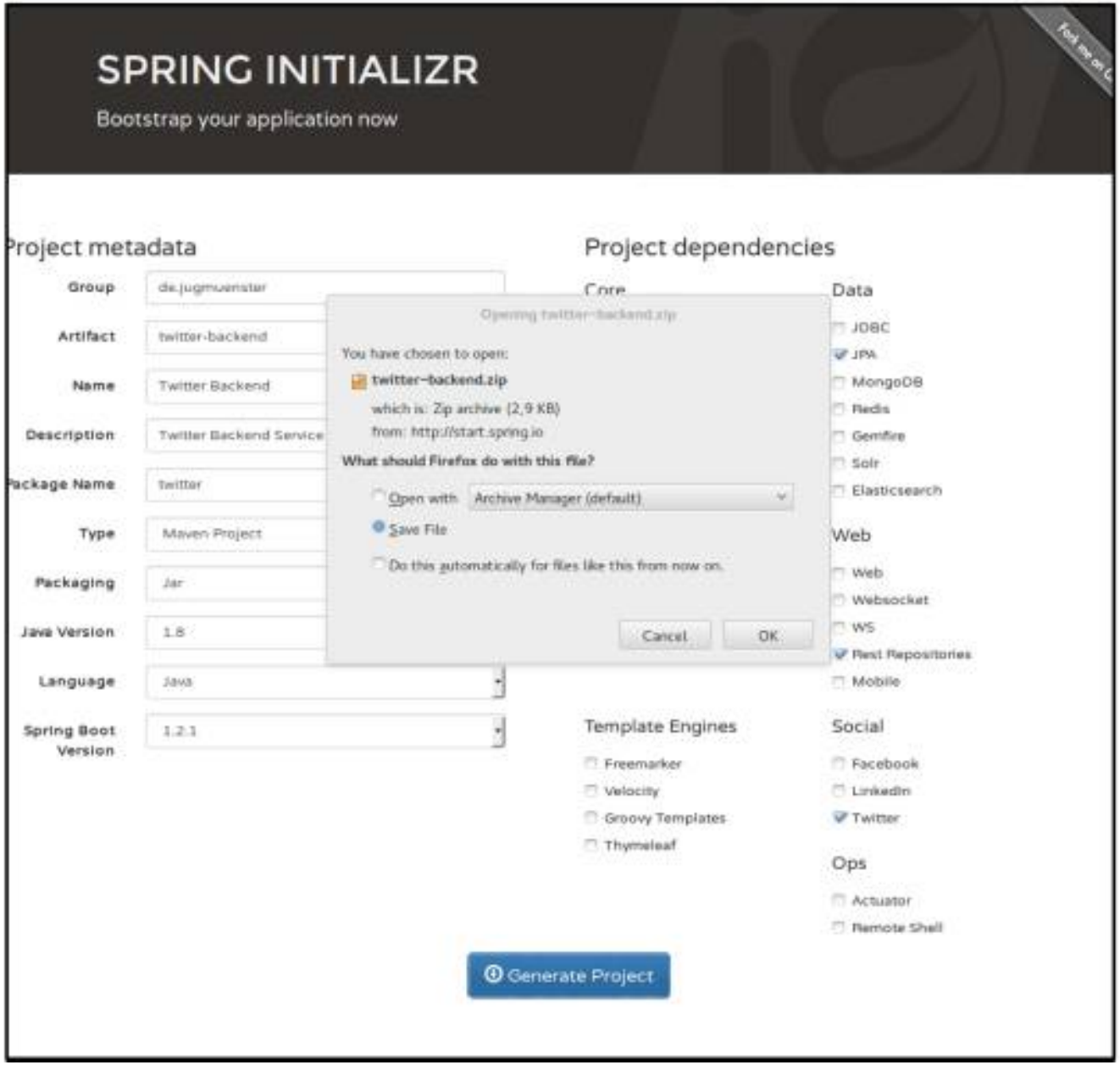
spring initializr

- Code generation maintained by spring
- Maven + Gradle
- Cloud service

maven archetype

- No broad support
- Hard to implement
- Often not well maintained
- One time scaffolding

SPRING BOOT - start.spring.io

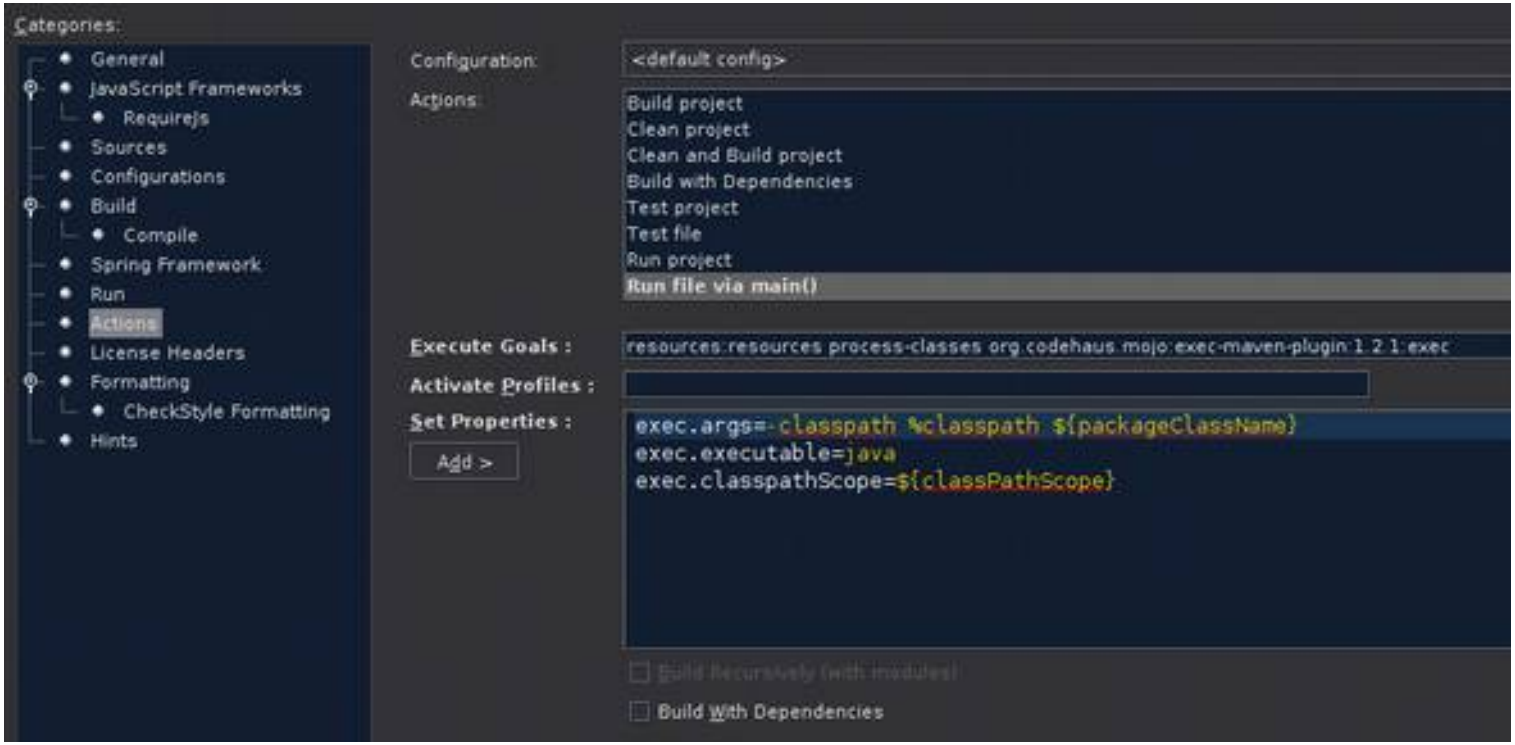


DEMO

TRION

Run spring-boot in Netbeans

Configuration hint: resources:resources



FRONTEND BOOTSTRAP

TRION

yeoman

- Code generator

YEOMAN

angular seed

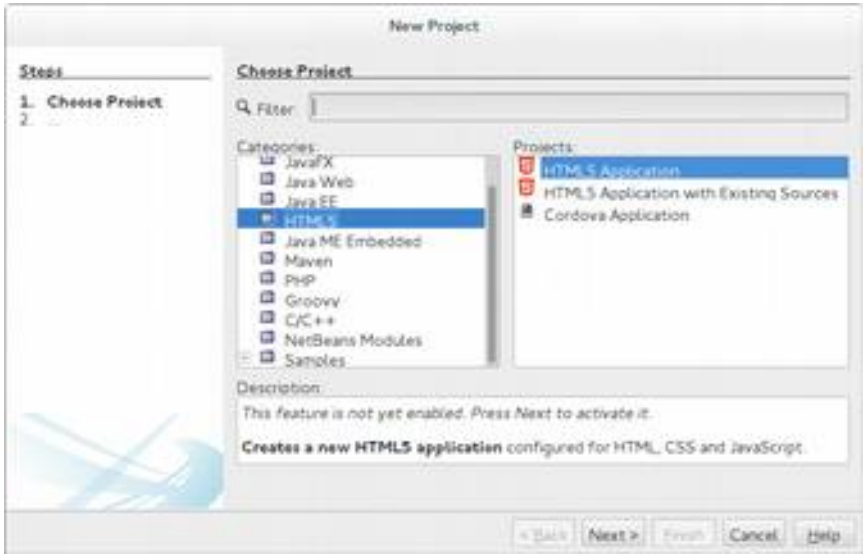
- Static template project

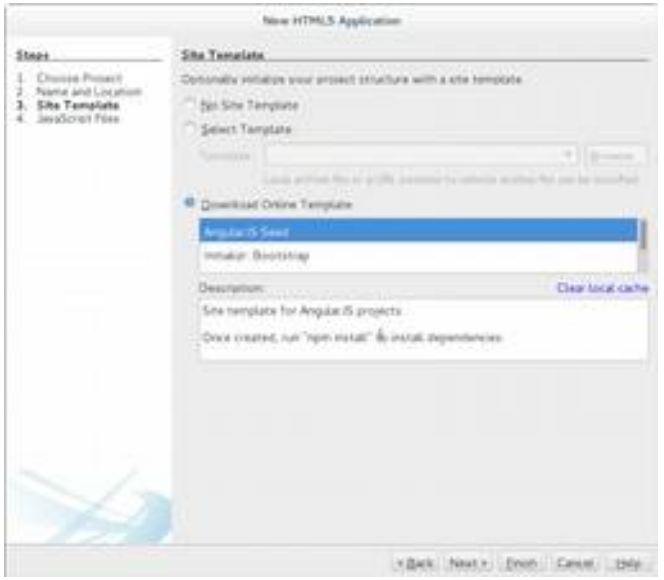
GRUNT

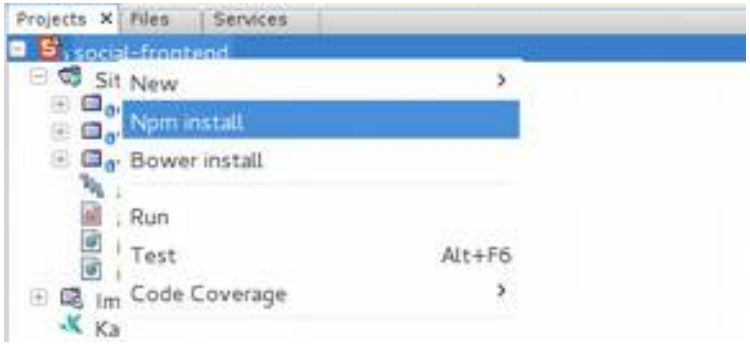
npm

DEMO

- Netbeans angular seed bootstrap







IMPLICATIONS

TRION

Teams / Roles

- Develop stateless back-end services
- Develop front-end components
- Different skill set

Paradigm Change

Development

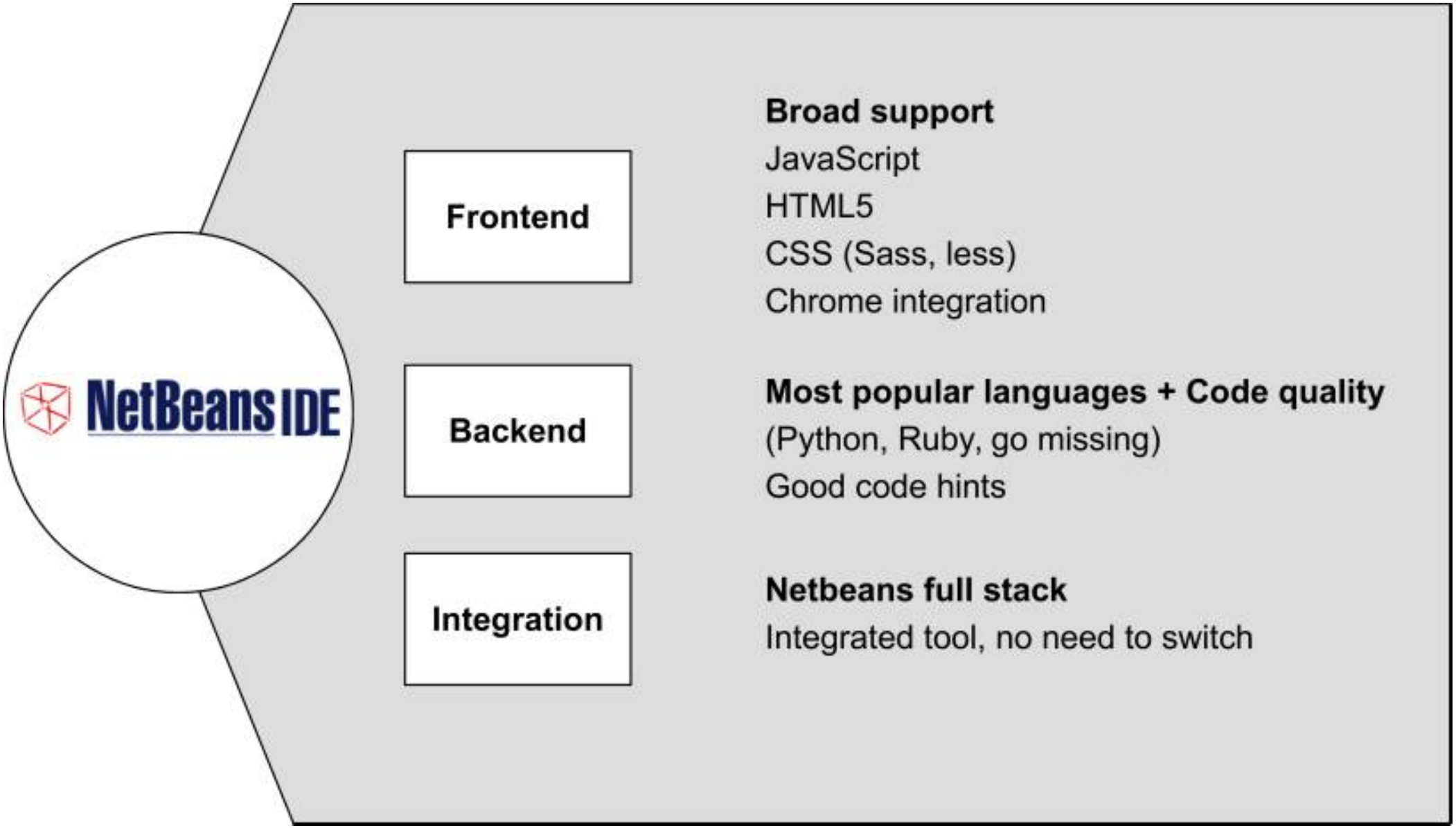
- Testability
- Flexibility
- API management
- Different level of re-use
- Speed of change

Operations

- Different infrastructure
- Distributed logging and diagnosis
- Scaling can leverage cloud infrastructure

IDE TOOLING

TRION



Q&A

TRION

Your Questions?

Thomas Kruse @everflux

STOP BACKUP STOP

DEMOS

TRION

- HTML Round-trip-editing
- Angular message service
- Webcomponent editing
- Spring-boot-init opening, running on tomcat
- Angular-seed opening