Pick Technologies & Tools Faster by Coding with JHipster:

Talk page at:
https://bpf.li/cin

Please start with the **problem description** on the top left, then work your way down. Then move onto the **solution description** on the right and work your down from there.

Java developers today **pick more technologies & tools more often**

- More technologies & tools?
  - Changes for Java developers from 10 years ago
    - Microservices & Container
    - Fullstack Developer
    - Shift Left
    - DevOps
    - Single Page Applications
    - Mobile Apps
    - Cloud
  - Before and after
    - **Before:** Either front-end or back-end developer inside app server with corporate DB
    - **After:** Code, test, run & support up to 4 applications
      - Java back-end
      - Web front-end
      - iOS front-end
      - Android front-end
  - Questions for four applications
    - Which framework?
    - How to build?
    - How to test?
    - How to debug in IDE?
    - How to run in Continuous Integration?
    - How to deploy?
    - How to monitor?
    - How to support?
    - How does it look with our data?
    - How productive can our team be?
- More often?
  - More releases
    - Java: From once every 3 years to every 6 months
    - Typescript: every 2-3 months for the last four years
  - More alternatives
    - Application frameworks
      - 5 years ago
        - JEE
        - Spring Boot
      - now
        - JEE
        - Spring Boot
        - MicroProfile
        - Micronaut
        - Quarkus
        - Helion
    - Java distributions
      - 5 years ago
        - Oracle
        - IBM
      - now
        - Oracle
        - IBM
        - AdoptOpenJDK
        - Azul
        - Redhat
        - Alibaba
        - Bell Soft
        - Amazon
        - SAP
- How to pick technology & tools?
  - Process
    - Analysis steps
      - Pick candidates
      - Does it work?
      - Does it scale?
      - Is it popular enough?
      - What's the adoption risk?
    - Code steps
      - How to build?
      - How to test?
      - How to debug in IDE?
      - How to run in Continuous Integration?
      - How to deploy?
      - How to monitor?
      - How to support?
      - How does it look with our data?
      - How productive can our team be?
  - Pick for 4 applications
    - Java back-end
    - Web front-end
    - iOS front-end
    - Android front-end
  - How can we learn?
    - Materials
      - Articles
      - Books
      - Videos
      - Talks
      - Online training
      - On-Site training
      - Consultants
      - Run code
    - Typical Issues
      - Problems
        - Outdated
        - Incomplete
        - Expensive (consultant, training)
      - Hard
        - How to debug in IDE?
        - How to run in Continuous Integration?
        - How to deploy?
        - How does it look with our data?
        - How productive can I be?
  - Dream
    - Develop complete application multiple times, once with each candidate
    - Use our data
    - For free

We **pick technologies & tools faster** by coding with JHipster

- What is it?
  - "Superstar developer"
    - Picks, configures, and updates best technologies & tools
    - Writes all the "boring plumbing code": Production grade, all layers
  - Open-source Java application generator
    - Full applications with front-end & back-end
    - No mobile apps
- What does it do?
  - Generates complete application with user management, tests, continuous integration, deployment & monitoring
    - Create it by running wizard or import application configuration from JHipster Domain Language (JDL) file
  - Generates CRUD front-end & back-end for our entities
    - Import data model from JDL file
    - Re-import after JDL file changes
  - Updates application
    - Re-generates application & entities with new JHipster version
    - Overwriting your own code changes can be painful!
    - Receive security patches or framework updates (like Spring Boot)
    - Sometimes switches out library: yarn => npm, JavaScript test libraries, Webpack => Angular CLI
- Live Demo
  - Generate application
    - JHipster picked and configured technologies & tools for us
    - Generated project
      - We picked architecture: monolith
      - We picked technologies & tools (like MongoDB or React)
    - Started to generate CRUD screens
  - Generate CRUD
- What is JHipster?
  - Technologies & tools?
    - Monolith and microservices
    - Java and Kotlin
    - Maven and Gradle
    - Spring Boot, Micronaut (alpha), Quarkus (beta), NodeJS (alpha), Dot.NET (alpha)
    - SQL: H2, MySQL, MariaDB, PostgreSQL, Oracle, MS SQL Server
    - NoSQL: MongoDB, Cassandra, Couchbase, Neo4J (beta)
    - Caching: Ehcache, Caffeine, Hazelcast, Infinispan, Memcached, Redis
    - Imperative and reactive APIs (Spring WebFlux)
    - Authentication: JWT, OAuth 2.0, HTTP session
    - Optional: Elasticsearch, WebSockets, Apache Kafka messaging
    - SPA Framework: React, Angular, Vue
    - Java & SPA in one project
    - Testing: Junit, Cucumber, Jest, Protractor, Gatling
    - Continuous Integration: Jenkins,Travis, CircleCI, Github, Gitlab, Azure
    - Deployment: Docker, Kubernetes, OpenShift, AWS, Microsoft Azure, Google Cloud, Cloud Foundry, Heroku
    - Multi-language support
  - CRUD with JHipster Domain Language (JDL)
    - Text file
    - Describes data model through entities
      - Entities are JPA entities with typical types & constraints
      - Entity Relationships: 1:1, 1:m, m:1, m:n
      - Entity configuration: Service classes, DTOs, pagination
    - Can also configure application
  - What's the risk?
    - 6 Years of Steady Releases
    - No proprietary Layers
    - Best-Practice code, Production-Ready
    - JHipster Lead now at Microsoft
    - Diverse Contributors
    - Weekly downloads went from 3,000 in early 2016 to 40.000 at the end of 2020
- How does it help?
  - Makes picking technologies & tools easier
    - Analysis steps
      - Pick candidates
      - Does it work?
      - Does it scale?
      - Is it popular enough?
      - What's the adoption risk?
      - (JHipster picks)
    - Code steps
      - How to build?
      - How to test?
      - How to run in Continuous Integration?
      - How to deploy in Docker?
      - How to monitor?
      - How to support?
      - How to debug in IDE?
      - How does it look with our data?
      - How productive can our team be?
      - How to deploy into cloud?
      - (Copy & paste from generated applications)
      - (Use full application created by JHipster)
      - (Get inspired by cloud deployment options)
  - Learn by running unknown code
    - We don't need to know a technology in detail to debug it, deploy it or monitor it
    - JHipster creates running applications with unknown technologies, so answer most questions
      - Analysis steps
        - Pick candidates
        - Does it work?
        - Does it scale?
        - Is it popular enough?
      - Code steps
        - How to build?
        - How to test?
        - How to run in Continuous Integration?
        - How to deploy in Docker?
        - How to monitor?
        - How to debug in IDE?
        - How does it look with our data?
    - SQL vs NoSQL: Entities
      - SQL has more annotations and different data type for "id"
      - NoSQL not so different from SQL!
    - Angular vs React
      - Angular separates presentation code to template, uses standard HTML tags
      - React mixes business logic & presentation code to template, doesn't use standard HTML tags
  - Live Demo
    - Review application
      - Started MongoDB through generated Docker Compose file
      - Application
        - Spring Boot & React Application in one Git project
        - Login & user administration
        - CRUD screens
        - Responsive layout for mobile
        - Monitoring & Administration
        - Multi-language support
    - Generate CI file
  - The dream comes true with JHipster
    - Develop complete application multiple times, once with each candidate
      - Monolith and microservices
      - Java and Kotlin
      - Maven and Gradle
      - Application Framework: Spring Boot, Micronaut, Quarkus
      - Relational DB and NoSQL
      - Imperativ and reactive APIs
      - SPA Framework: React, Angular, Vue
    - Use our data
      - Generates CRUD front-end & back-end for our entities with JDL files
    - For free
      - Open-source