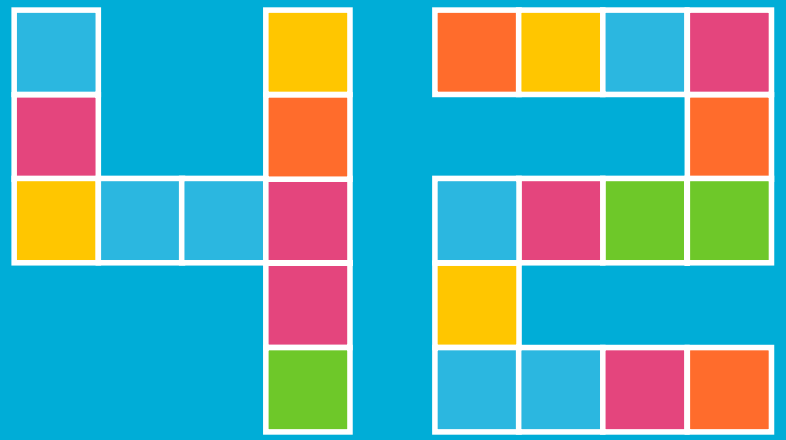
Exploring Spring Boot Clients
A Comprehensive Overview

Patrick Baumgartner
42talents GmbH, Zürich, Switzerland

@patbaumgartner
patrick.baumgartner@42talents.com



Abstract

Exploring Spring Boot Clients: A Comprehensive Overview

The extensive Spring Boot landscape is the preferred solution for building robust enterprise applications today. This talk will explore the role of Spring Boot clients in seamlessly connecting to a variety of services, including databases and RESTful APIs. We will learn about Templates, Interface Clients, and other client abstractions and discover how they work and how to extend their functionality.

Join us to deepen your understanding of how to use Spring Boot clients for seamless integration and efficient communication within your application.

Exploring Spring Boot Clients

A Comprehensive Overview

Patrick Baumgartner
42talents GmbH, Zürich, Switzerland

@patbaumgartner
patrick.baumgartner@42talents.com

Exploring Spring Boot Clients
A Comprehensive Overview

Patrick Baumgartner / 42talents GmbH

Patrick Baumgartner

Technical Agile Coach @ **42talents**

My focus is on the **development of software solutions *with* humans**.

Coaching, Architecture, Development, Reviews, and Training.

Lecturer @ **Zurich University of Applied Sciences ZHAW**

Co-Organizer of **Voxxed Days Zurich**, **JUG Switzerland**, Oracle ACE Pro Java ...

@patbaumgartner

Agenda

Agenda

- A Brief History of Programming Styles
  - … Java
  - … Spring
- Design Patterns
- Spring Client Examples
- Summary


DMC

A Brief History of Programming Styles


0:00 / 0:08

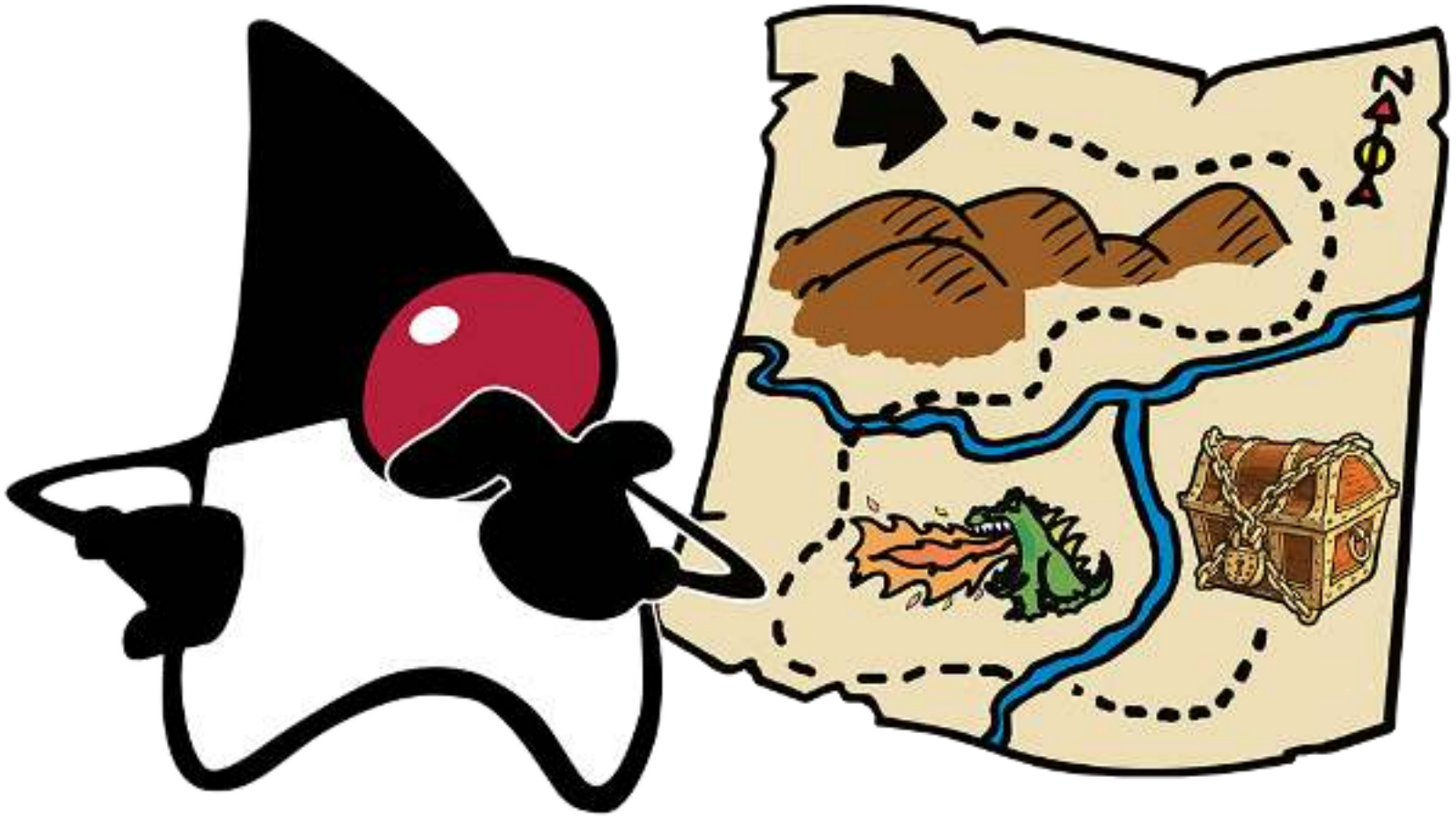
A Brief History of Programming Styles

- **1940s - 1950s** - Machine Code and Assembly Language
- **1950s - 1970s** - Procedural Programming
- **late 1960s - 1970s** - Structured Programming
- **1970s - 1980s** - Object-Oriented Programming
- **1980s - 1990s** - Declarative Programming
- **1950s - present** - Functional Programming
- **1990s - present** - Event-Driven and Reactive Programming
- **2000s - present** - DSLs and Language Polyglotism
- **2010s - present** - Modern Trends


Java™

A Brief History of Java

- **late 1990s and early 2000s** - Java 2 (J2EE and J2SE)
- **2004** - Java 5 with Generics and Metadata Annotations
- **2006** - Java 6 with Scripting Support
- **2011** - Java 7 with Project Coin Features
- **2014** - Java 8 with Lambda Expressions and Stream API
- **2017** - Java 9 with Project Jigsaw and Module System
- **2018 - 2019** - Java 10-11 with Local-Variable Type Inference, ...
- **2019 - 2020** - Java 12-14 with several Preview Features and Records
- **2020 - 2021** - Java 15-17 with Sealed Classes, Records, and Pattern Matching
- **2022 - 2023** - Java 18-21 with Project Loom and Valhalla
- **2024 - 2025** - Java 22-23 with Project Panama and Amber




spring®

Celebrating 20 Years of Spring

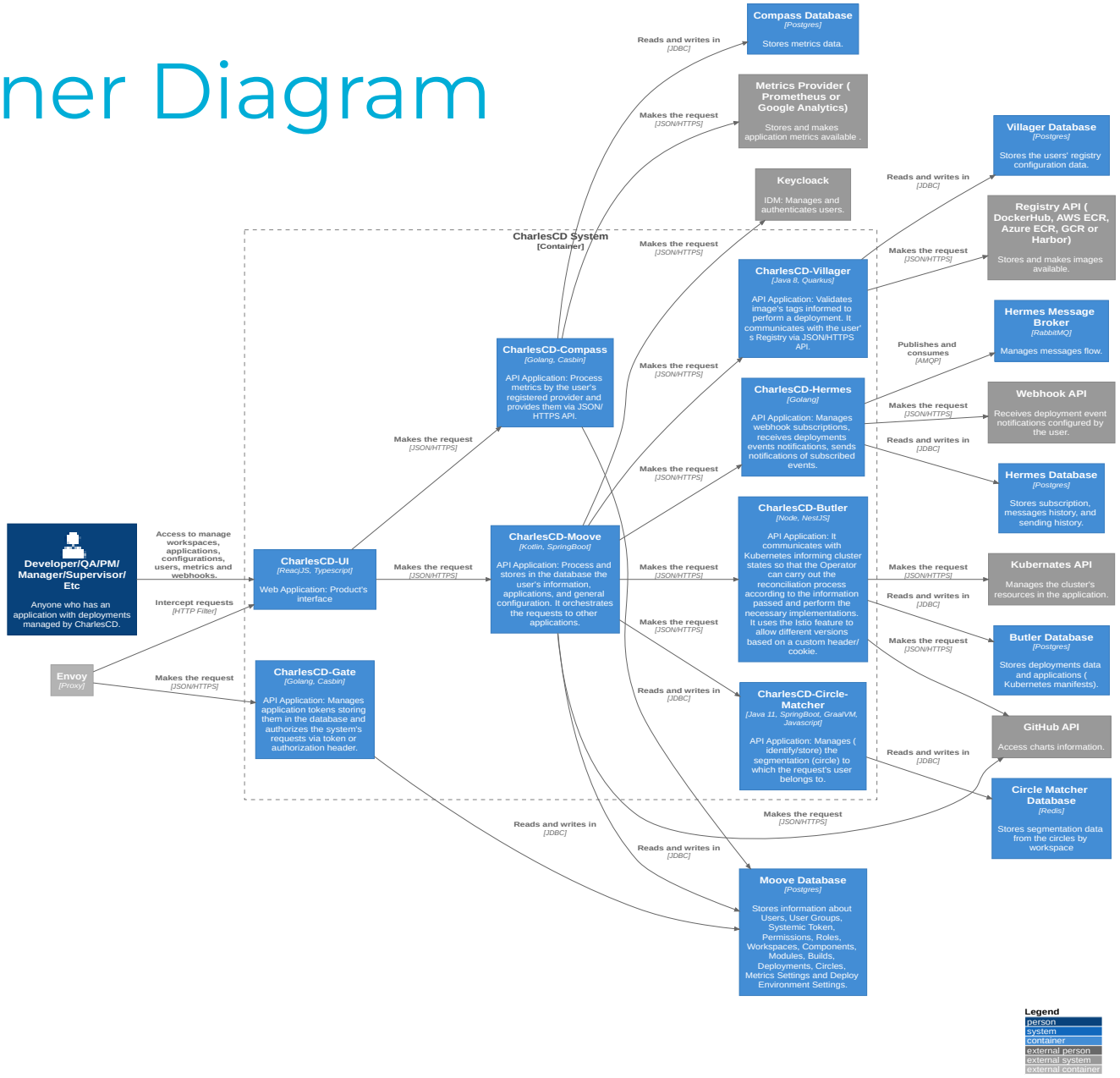
spring
by VMware Tanzu

vmware

Why do I need to know all this?! 

We Are Living in a Complex World ...

C4 Container Diagram



Design Patterns

Template Method is a behavioral design pattern that defines the skeleton of an algorithm in the superclass but lets subclasses override specific steps of the algorithm without changing its structure.

- Instead of extending the template, callbacks can be used.
- They signal, that the desired action is completed.
- It can also be used to perform actions such as mapping results.

Proxy is a structural design pattern that lets you provide a substitute or placeholder for another object. A proxy controls access to the original object, allowing you to perform something either before or after the request gets through to the original object.

- A Spring proxy typically hides cross-cutting concerns.
- It's also used to hide repetitive infrastructure code.
- Implemented with Spring AOP or AspectJ.

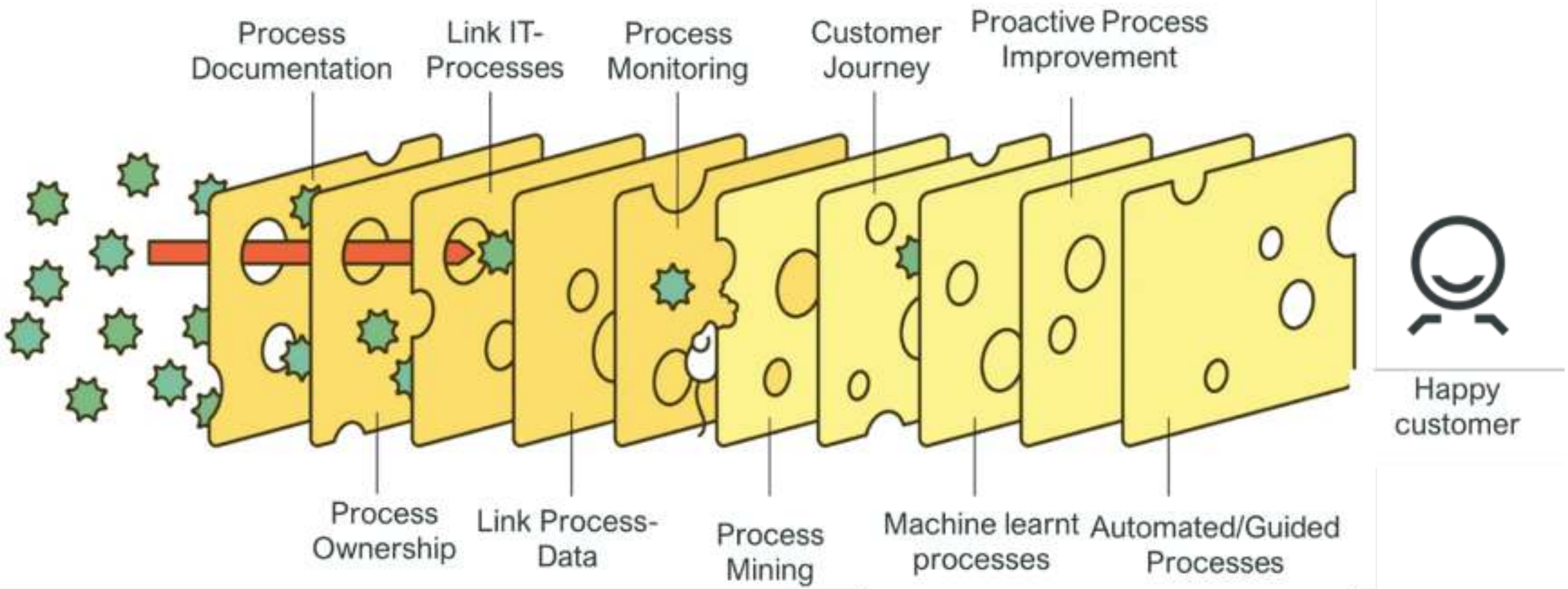
Builder is a creational design pattern that lets you construct complex objects step by step. The pattern allows you to produce different types and representations of an object using the same construction code.

- ➡️ Allows multiple optional parameters or configurations.
- ➡️ Provides a fluent and readable interface.
- ➡️ Provides more control and lots of flexibility.

Facade is a structural design pattern that provides a simplified interface to a library, a framework, or any other complex set of classes.

➡️ Reduces internal conflict and cognitive load.

➡️ Often used to hide ugly code.


Process Documentation
Link IT-Processes
Process Monitoring
Customer Journey
Proactive Process Improvement
Process Ownership
Link Process-Data
Process Mining
Machine learnt processes
Automated/Guided Processes
Happy customer

Spring Clients

Data Access

JdbcTemplate

- Introduced 3. Mai 2001
- Hides lots of infrastructure code
- NamedParameterJdbcTemplate implementation
- Huge list of methods with parameters
- Uses Callbacks like RowMapper, ResultSetExtractor, RowCallbackHandler
- Full control over SQL
- Threadsafe

Simple ...

SimpleJdbcInsert

- Introduced in Spring 2.5
- Multithreaded, reusable object
- Easy (batch) insert capabilities for a table
- An Abstraction over the JdbcTemplate
- Meta-data processing, and manual parameter mapping
- "Fluent" style

SimpleJdbcCall

- Introduced in Spring 2.5
- Multithreaded, reusable object
- Call a stored procedure or a stored function
- An Abstraction over the JdbcTemplate
- Meta-data processing, and manual parameter mapping
- "Fluent" style

Spring Data Proxy

- Introduced in 2011.
- Simplify data access in Java applications.
- Provides a high-level abstraction for interacting with various data stores.
- Covers relational databases, NoSQL databases, key-value stores, etc.
- Includes modules like JPA, JDBC/R2DBC, KeyValue, LDAP, MongoDB, Redis, REST, Apache Cassandra, Apache Geode, Couchbase, Elasticsearch, Neo4j

JdbcClient

- Introduced in Spring 6.1
- Fluent API
- AutoConfiguration for JdbcClient since Spring Boot 3.2
- Deals with classic and named parameters
- Uses internally the JdbcTemplate

Web

RestTemplate

- Introduced in Spring 3.0
- Feature Complete → Maintenance mode
- Method Overloading
- Apache HTTPClient, OkHttp, JDK HttpURLConnection, JDK HttpClient, ...

WebClient

- Introduced in Spring 5.0
- Fluent API
- Asynchronous
- Reactive Stack
- AutoConfiguration for WebClient.Builder since Spring Boot 2.0
- Netty HttpClient, Jetty HttpClient

HTTP Interface Client

- Introduced in Spring 6.0
- Asynchronous with WebClient
- Synchronous with RestClient since Spring 6.1
- Needs still some manual configuration
- Spring Boot AutoConfiguration is still in discussion

RestClient

- Introduced in Spring 6.1
- Fluent API
- Synchronous like the RestTemplate
- AutoConfiguration for RestClient.Builder since Spring Boot 3.2
- Can be created from RestTemplate
  - Copies Handlers, Intercepters, Converters, ...
- Naming → Rest, ... really?!

Spring Integration

Spring Integration

- Lightweight messaging within Spring-based applications.
- Supports integration with external systems via declarative adapters.
- Integrates with many technologies:
    - AMQP, Apache Camel, Apache Cassandra, Debezium CDC, Spring ApplicationEvent, RSS and Atom, File, FTP/FTPS, GraphQL, Hazelcast, HTTP, JDBC, JPA, JMS, JMX, Apache Kafka, Mail, MongoDb, MQTT, R2DBC, Redis, Resource, RSocket, SFTP, SMB, STOMP, Stream, Syslog, TCP and UDP, WebFlux, WebSockets, Web Services, XML, XMPP, ZeroMQ, Zookeeper.

Summary

Templates

- High-level abstraction
- Templates are full of overloads
- Use the SpringBoot provided Builder if available
- There are templates for:
  - Redis, Neo4j, Cassandra, MongoDB, Solr, Gemfire, Ldap, Couchbase, ...
  - JMS, RabbitMq, Kafka ...
  - Rest, Webservice, Websocket, ...
  - *Build your own!*

Templating API is resilient to time. 

Interface Clients

- Implements generic (data) access pattern
- Declaration via Interfaces
- Hides a lot of boilerplate code
- Examples:
  - HTTP Interface Clients
  - Spring Data Proxies
  - OpenFeign (Spring Cloud)
  - ...

Clients

- A client has typically logic
  - Retry, OAuth2, ...
- Reduces complexity by using a fluent API
- Examples:
  - JdbcClient
  - DatabaseClient (R2DBC)
  - WebClient
  - RestClient
  - ...

Clients are here to stay! 

Last Famous Words ...

Style Matters!


42

Wir bitten um
dein Feedback!

entwickler.de

Exploring Spring Boot Clients

A Comprehensive Overview

Patrick Baumgartner
42talents GmbH, Zürich, Switzerland

@patbaumgartner
patrick.baumgartner@42talents.com