FunctionalJ

Adding functional spices to Java

Ever since Java 8 released over 5 years ago, we – Java programmers – have gotten some tastes of programming in a more functional way. Additions such as functional interfaces, method references, Optional type and especially Stream API change the way many of us code. We use less loops, do more immutability, for example. But Java is not a functional language, never designed to be and never will become one. Introducing *FunctionalJ* – a pure Java library that migrates many of the functional ideas and patterns to Java. *FunctionalJ* is designed with Java programmers in mind. It does not force in lambda calculus or category theory or any other abstract math concepts. Instead, it adds Java-friendly functional elements such as function types, lazy lists and maps, pipe, immutable types, lens, algebraic data types, rule types, function-friendly dependency-injection mechanism and side-effect management. Together, they provide many benefits of functional programming such declarative programming, reuse thought composition, greater type safety (make illegal state unrepresentable) and decoupling of side-effect and logic code – all these in your Java comfort zone. This talk discusses *FunctionalJ*, its features and example codes with live coding demo.