ELVIS

THE ERLANG STYLE CODE CHECKER

Each developer has their own preferred style of writing code, based on personal preferences and habits formed over time. In a team environment with developers of all abilities, variations in programming style can end up costing a lot of time (not to mention nerves and money).

A unitary programming style will increase project speed, improve code maintainability and significantly reduce the technical debt your team will have to deal with in the future. Having an agreement upon common guidelines is incredibly useful, but no developer can be asked to remember 100 style rules while focusing on writing good code.

BENEFITS

STYLCE CODE
CONSISTENCY

CODE CONVENTIONS
ENCOURAGEMENT

CONTINUOUS
MONITORING

AUTOMATIC ERROR
DETECTION

HOW DOES ELVIS WORK?

Elvis is a code style checker that enforces common guidelines in your GitHub-hosted Erlang project by detecting where guidelines are not being met. Once Elvis is activated for your repository, the rules you configure will be executed every time a pull request is created or updated, and Elvis will add comments to the lines that don't comply with these rules.

By enabling this verification for your project and automating part of the review process, you help reviewers focus on the most important details. Since only those lines that have been modified in the pull request are considered when adding comments, you can use Elvis as a component in the incremental improvement of your code base.

GitHub

RULES

The set of rules can be selected by specifying a configuration file (elvis.config) in your project. Some rules accept one or more configuration values, allowing for a large amount of flexibility when applying them. Here are a few rules examples:

• Nesting Level: there shouldn't be any nested expressions that exceed the limit specified in the configuration
• God Modules: there shouldn't be any modules that export a number of functions greater than the limit specified in the configuration
• Used Ignored Variable: don't use a variable whose name indicates it is actually an ignored variable (i.e. it starts with an underscore _Var).
• Save work by creating your tests automatically
• Shorten your time to market by testing more effectively
• Improve the quality of your product by testing more scenarios
• Reduce risk by detecting errors early in development
• Save time by automatic error analysis
• Reduce maintenance cost of large test suites by introducing generic properties and removing "bespoke" tests
• Find latent bugs within your existing software products

OUR EXPERTISE

Elvis is free of charge for public repositories, while using it for private repositories involves a small annual fee per each private repository. Elvis was developed by Erlang Solutions' mobile division, Inaka.

Inaka have extensive experience in system architecture and design, migration to OTP, prototyping, coding, integration, testing, profiling, code rewrites, code and architecture reviews and maintenance. By combing this experitise with Inaka's, we expanded our capabilities to designing end-to-end web and mobile apps in Java, Objective-C, Ruby, and use other leading technologies i.e. Android, iOS, Rails, Riak, Elasticsearch, Redis, MySQL, and Heroku.   **www.erlang-solutions.com**

ABOUT ERLANG SOLUTIONS

Founded in 1999 soon after Erlang was released as open source. We specialiSe in providing businesses with truly scalable solutions through the creation, integration, delivery and lifetime support of products and services based on the Erlang programming language. Our HQ are in London and our team has worked with over 300 clients from our offices in Stockholm, Krakow, Budapest, Berlin, San Francisco, Seattle and Buenos Aires.

OUR OFFERING

SERVICES End-To-En Solutions | Consulting | Support | Development | Mobile | Training
PRODUCTS MongooseIM | Riak KV | WombatOAM | Sardina | LSD | QuickCheck
MegaLOAD | Linc & Loom | Elvis | RabbitMQ
INDUSTRIES Online Gambling & Betting | Telecoms | Financial | Advertising & New Media | Gaming
EVENTS Erlang Factory | Erlang User Conference | Code Mesh | ElixirConf
COMMUNITY Development | Education | Support | Research | Recruitment

e: info@erlang-solutions.com
w: www.erlang-solutions.com