Язык программирования: Elixir

Компилятор – Elixirc. Интерпретатор – IEx. Система сборки – Mix. Система автоматического форматирования – mix format. Lint tools – Credo. Инструменты тестирования – ExUnit. Стиль кодирования - Elixir Style Guide.

В качестве технологии, которую я буду изучать во время прохождения курса по функциональному программированию, мне интересно выбрать ту, которая больше применяется в реальных производственных системах. В этом плане Elixir, основанный на Erlang/OTP, гораздо более прагматичен и востребован на рынке, чем, например, Haskell, который имеет более абстрактную концепцию, связанную с академическими науками и математикой.

Также, если Haskell является чисто функциональным языком программирования, несовместимый с остальными парадигмами, то Elixir – это гибридный и динамически типизированный язык, в котором при этом используется знакомая мне по JAVA концепция байт-кода. Также использование модели акторов делает этот язык самым объектно-ориентированным. Всё это должно сделать понимание парадигмы функционального программирования после изучения ООП более простым, по сравнению с другими технологиями.

Но главной причиной выбора этого языка я бы назвал большие возможности распараллеливания, библиотеки OTP и особый подход к параллелизму через модель акторов (В рамках этой модели, между процессами (акторами) нет ничего общего. Каждый процесс поддерживает свое собственное внутреннее состояние, и единственный способ общения между различными процессами — отправка сообщений). Это делает язык Elixir наиболее удобным для работы с высоконагруженными распределёнными системами и параллельных вычислений, что в наш век Интернета и многоядерных процессоров крайне актуально.

Также к плюсам Erlang/OTP и Elixir я бы отнёс высокую надёжность (некоторые программы, работающие на виртуальной машине Erlang, смогли достичь надежности 99.9999999%), скорость (виртуальная машина Erlang запускается быстро, производительность хорошая) и масштабируемость (Elixir легко справляется с такими вещами как: кластеризация, RPC и сетевые взаимодействия).

Выбор в пользу Elixir по сравнению с Erlang был сделан из-за нескольких причин. Во-первых, Elixir является более современным языком, поэтому его сообщество быстрее растёт и развивается. В нём более современный синтаксис (Ruby vs Prolog), а также ряд новых инструментов. Благодаря фреймворку Phoenix Elixir более удобен и популярен в веб-разработке (причем не только в сравнении с Erlang, но и с многими другими языками), разработке микросервисов и распределенных приложений.

В качестве основной книги, по которой я буду знакомиться с технологией, я выбрал “Elixir in Action” Саши Юрича. Во-первых, она очень популярна и её рекомендует само сообщество, утверждается, что по ней можно усвоить все фундаментальные знания о языке. Во-вторых, совсем недавно вышло 3 издание этой книги с обновлениями и дополнениями, из чего можно сделать вывод, что она не устарела.

Что касается соображений на предмет лабораторной работы №4, то меня заинтересовали примеры: peer2peer сервиса чатов с шифрованием и хранением истории (можно использовать фреймворк Phoenix), системы распределённого хранения данных, системы управления вычислениями на кластере (звучит похоже на Kubernetes).

Краков Кирилл P3331