

# Erlang Virtual Machine

Эликсир и Эрланг объединяет виртуальная машина EVM (Erlang Virtual Machine),

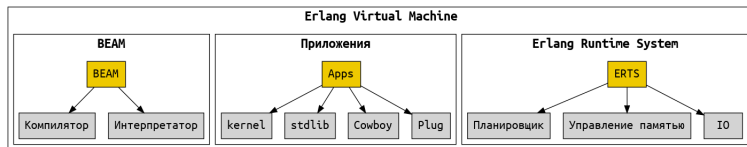
которую обычно называют **BEAM** (Bogdan's Erlang Abstract Machine).

# Erlang Virtual Machine

Операционная система в миниатюре:

- ▶ планировщик процессов,
- ▶ управление памятью,
- ▶ ввод-вывод,
- ▶ сетевой стек,
- ▶ и др.

# Erlang Virtual Machine



# Erlang Virtual Machine

Основные свойства машины:

- ▶ Многопоточность (Concurrency);
- ▶ Устойчивость к ошибкам (Fault Tolerance);
- ▶ Поддержка распределенных систем (Distribution);
- ▶ Горячее обновление кода (Hot Code Upgrade).

# Erlang Virtual Machine

Еще несколько свойств:

- ▶ Симметричная multiprocessing (Symmetric Multiprocessing);
- ▶ Модель акторов (Actor Model);
- ▶ Система реального времени (Soft Real Time);
- ▶ Сборщик мусора (Garbage Collector);
- ▶ Интерактивная консоль (Erlang/Elixir Shell);
- ▶ Трассировка (Tracing).

# Многопоточность

Процессы (thread) и планировщики (scheduler),  
независимые от ОС.

Процессы легковесные, их можно создавать десятки  
и сотни тысяч.

# Многопоточность

Каждый процесс имеет свою изолированную память, разделяемой памяти (shared memory) нет.

Процессы изолированы, блокировка или падение одного процесса не влияет на работу остальных.

Данные передаются отправкой сообщений (message passing).

# Многопоточность

BEAM способна запускать до **134,217,727** ( $2^{27}$ ) процессов.

Запуск нового процесса занимает **3-5 микросекунд**.

На старте процесс резервирует **2696 байт** памяти, включая стек, кучу и метаданные.



# Многопоточность

Отдельный планировщик на каждом ядре CPU.

Балансируют нагрузку, передавая процессы друг другу.

# Устойчивость к ошибкам

Исключения не основной способ обработки ошибок.

Основной способ – это механизм мониторинга одних процессов другими.

**supervisor** (наблюдатель) и **worker** (рабочий процесс).

# Устойчивость к ошибкам

Супервизоры наблюдают за рабочими процессами и друг за другом.

Организованы в дерево, где узлами являются супервизоры, а листьями – рабочие процессы.

# Устойчивость к ошибкам

Следующий уровень устойчивости к ошибкам – объединение узлов в кластер.

Если узел падает, то его функцию берет на себя другой узел.

# Поддержка распределенных систем

Устойчивость к аппаратным авариям.

Железо выходит из строя не часто,

но в дата центре из сотен серверов это случается регулярно

и является штатной ситуацией.

# Поддержка распределенных систем

Горизонтальное масштабирование позволяет  
строить большие системы, справляющиеся  
с большими нагрузками и большими данными.

# Поддержка распределенных систем

Сетевая прозрачность (location transparency).

Процессы на одном узле и на разных узлах

общаются одинаково.

# Поддержка распределенных систем

Доверенная среда (trusted environment).

Любой процесс может посылать любые сообщения кому угодно.

Любой код выполняется с равными правами, без ограничений.



# Горячее обновление кода

Важно для телекоммуникационного оборудования 80-90-х годов.

Обновление без прерывания обслуживания клиентов,

без разрыва текущих телефонных сессий.

# Горячее обновление кода

В современном мире мы умеем поддерживать работу кластера при выходе из строя части узлов.

Используем это для обновления.

# Горячее обновление кода

Полезно как инструмент разработки.

Используем локально на машине разработчика.