



「いま」と「これから」

Davide Bettio
名古屋 | piyopiyo.ex

自己紹介 (Davide Bettio)

<https://github.com/bettio/> | davide@uninstall.it | <https://uninstall.it/>

- 2004年頃からハードウェア／組込みシステムを触っています
- 2005年頃からオープンソース開発を続けています (KDE Plasma など)
- 2017年に Astarte Platform を作る中で Elixir に魅了されました
- 2017年に AtomVM プロジェクトを開始
- ハイキングが好きです

はじめまして!

本日の内容

- 今日は講演（ハンズオンはありません）
- AtomVMの「いま」と「これから」を紹介します
- ハンズオンが気になる方へ：
 - 3月21日に対面型ワークショップ「AtomVM入門～大須散策を添えて～」が開催されます
 - piyopiyo.ex イベントページ: <https://piyopiyoex.connpass.com/event/373137/>



本日のお題

- いまのAtomVM
 - 最近の主な変更点
- これから AtomVM
 - v0.7
 - V0.8
 - Zephyr／NuttX
 - AtomVMとNerves
 - AtomGL
 - v1.0に向けて

30秒でわかるAtomVM

- Elixir/Erlangを書いて、マイコン（ESP32 / RP2040 / STM32...）や制約のある環境（例：WASM）へデプロイできます
- 軽量プロセス、スーパービジョン、メッセージパッシングといったErlangのプログラミングモデルはそのまま使えます
- 目的：制約のあるハードウェア上で、「組込み+分散+耐障害」システムを実用化すること



2025年は
AtomVMにとって
すばらしい年でした

2025年

- 2025年はVMコアの強化に注力しました
- Erlang分散、JIT、大きい整数、互換性向上、OTP-28対応など多くの改善が入りました
- Popcornの利用形態を支えるための作業も進みました
- 2025年の成果は AtomVM v0.7 としてリリースされます

2025 =

分散・高速化・互換性向上

Erlang分散 (Distribution)

- AtomVMノードは AtomVM / BEAM (Erlang/OTP 24以降) ノードを含むクラスタに参加できます
- 対応していること
 - すべての型のシリализ
 - epmdプロトコル (クライアント／サーバ)
 - メッセージパッシング
 - プロセス監視
 - I/O分散 (group leader)
 - Erlang/OTP から AtomVM へのRPC
- シェルも動作 (ただし追加モジュールの提供が必要)

なぜErlang分散か

- 一言でいうと：可能性が広がります
- アイデア：
 - シリアル/I²C/SPI を TCP の代わりに使う
 - 強力なCPU上のNervesシステムと組み合わせたハイブリッド構成
 - AtomVM搭載のマイコンを賢いI/O拡張ボードとして使う
 - AtomVM搭載のマイコンを常時稼働の低消費電力ドメインとして使う
 - ひょっとするとロボット工学にも？

JIT

- 2025年11月以降、AtomVMは JIT (Just-in-Time) と AOT (Ahead-of-Time) の両方のコンパイラ方式をサポート
- コードをネイティブコードへ変換し、実行速度を最適化
 - 代償：フラッシュまたはRAMを多く消費
- 対応ターゲット
 - arm6m (Cortex-M)
 - aarch64 (64-bit ARM)
 - RISC-V (32-bit)
 - x86_64

AOT / JIT

AOT（事前コンパイル）：開発機やCIで実行できます

- 起動・モジュール読み込みが高速
- JITよりRAMが少ない機器に向く
- JITエンジンの同梱は任意
- 特定のターゲット（アーキテクチャ）向けにコード生成

JIT（実行時コンパイル）：デバイス上で実行できます

- 事前の追加手順が不要
- 実行中にモジュールを読み込む

数字で見るAtomVM

- VMに組み込みのコアネイティブ関数：267（NIF 196、BIF 71）
 - うち167は erlang モジュール（OTPのerlangモジュールには350の関数があるので47%）
- 利用可能なErlang stdlib モジュール：51
 - erlang、gen_server、gen_event、gen_statem、gen_tcp、gen_udp、lists、queue など
- サポート対象プラットフォーム：5
 - generic_unix（Linux/macOS/FreeBSD）、rp2（Raspberry Pi Pico）、ESP32、STM32、emscripten
- 新規コントリビュータ：11名

erlangモジュール関数の47%で足りるのか

- 未対応の183個の関数は、厳密には必須ではありません
 - 非公開／内部用途のもの
 - マイコンでは起こりにくい特殊な場面向けのもの
 - いくつか有用なものはありそう（それらの追加は歓迎）
- 例：
 - erlang:set_cpu_topology/1（非公開）
 - erlang:phash/2（非推奨）

結論として、全部なくても概ね困りません



納得できない。
証拠を見せて。

証拠

事実：Elixirコンパイラはかなり複雑

AtomVMのmainブランチは、

AtomVM上でElixirコンパイラを動かせる
程度に十分完成しています。

セルフホストは良い目標です。

現状提供しているOTP部分集合でも、
より複雑なアプリケーションが
十分に動かせることを示します。

Elixir Eval Demo

Simple Elixir REPL. Write some code or use our examples and click "Evaluate" (Ctrl+Enter) to see results. Entirely offline.

Code

```
case :erlang.system_info(:machine) do
  -c"BEAM" -> "Running on the BEAM"
  -c"ATOM" -> "Running on AtomVM"
end
```

Evaluate Example: case Example: pipes

Results

```
"Running on AtomVM"
```

23.3 ms ✓

Logs

Popcorn

- ブラウザで動く Elixir
- WebAssembly (WASM) にコンパイルされた AtomVM を利用します
 - Elixirコードは通常どおりコンパイルされ、ローカルにElixirツール群は不要
- Popcorn はライブラリであり、Elixir フロントエンド Web アプリを作るためのツール群 (mix タスク) も提供します
- Popcornの実現のためにAtomVMに取り込まれた改善の多くは、様々な場面で役立ちます

この新しい
Elixir Language Tour なら、
Webブラウザだけで
Elixirを学べます。

ローカル環境への
インストールは不要です！

The screenshot shows a browser window displaying the Elixir Language Tour website at elixir-language-tour.swmansion.com. The page title is "Elixir Language Tour". The main content area features a "Welcome to the Elixir Language Tour" section with a subtitle "This guide is your first step into the world of Elixir, a modern, functional programming language designed for building scalable and maintainable applications." Below this is a "Navigation" section and a "Code editor" section. The code editor contains the following Elixir code:

```
1 # A line that starts with # is a comment
2 # We'll use comments for short explanations and small exercises
3 # The exercises are additionally marked with '
4
5 # Try changing IO.puts to IO.inspect. What's the difference?
6 IO.puts("Hello, Elixir!")
7
8 IO.inspect(1 + 2)
9
10 # You can specify a 'label' to include additional info:
11 IO.inspect(1 + 2, label: "Added 1 to 2")
12
13 # You can also omit braces:
14 IO.inspect 1 + 2, label: "Added 1 to 2"
15
```

At the bottom of the code editor, there are "Reset Code" and "Run Code" buttons. The output panel shows the results of running the code:

(23.780 ms)

Output:
Hello, Elixir!
3
Added 1 to 2: 3
Added 1 to 2: 3

3

Phoenix LiveView を
Webブラウザで動かす
というのはどうでしょうか？

Software Mansion チームが
実現しました。

form demo

/*! 🌟 daisyUI 5.0.35 */
Done in 122ms
[info] GET /
[debug] Processing with FormDemoWeb.PageController.home/2
Parameters: %{}
Pipelines: [:browser]
[info] Sent 200 in 43ms
[info] CONNECTED TO Phoenix.LiveView.Socket in 22μs
Transport: :websocket
Serializer: Phoenix.Socket.V2.JSONSerializer
Parameters: %{"_csrf_token" => "dxchQBs0JRoQbAuNRIHixQhJX8R8AIIFGc5HBARZ5_j0hDHLglFKNRY", "_live_referer" => "undefined", "_mount_attempts" => "0", "_mounts" => "0", "_track_static" => "%{"0" => "http://localhost:4000/assets/css/app.css", "1" => "http://localhost:4000/assets/js/app.js"}", "vsn" => "2.0.0"}
[debug] MOUNT FormDemoWeb.FormDemoLive
Parameters: :not_mounted_at_router
Session: %{"_csrf_token" => "1PBuSvdHJYoDzzCkXFl9zJPQ"}
[debug] Replied in 161μs
[debug] HANDLE EVENT "llv_local_message" in FormDemoWeb.FormDemolive
Parameters: %{"payload" => %{"type" => "sync_request", "view" => "FormDemolocal"}
[debug] Replied in 94μs
[debug] HANDLE EVENT "llv_local_message" in FormDemoWeb.FormDemolive
Parameters: %{"payload" => %{"type" => "new_user", "user" => %{"email" => "user421@example.com", "username" => "user421"}}, "view" => "FormDemolocal"}
[debug] Replied in 43μs
[debug] HANDLE EVENT "llv_local_message" in FormDemoWeb.FormDemolive
Parameters: %{"payload" => %{"type" => "new_user", "user" => %{"email" => "user281@example.com", "username" => "user281"}}, "view" => "FormDemolocal"}
[debug] Replied in 41μs
[debug] HANDLE EVENT "llv_local_message" in FormDemoWeb.FormDemolive
Parameters: %{"payload" => %{"type" => "new_user", "user" => %{"email" => "user286@example.com", "username" => "user286"}}, "view" => "FormDemolocal"}
[debug] Replied in 442μs
^C
BREAK: (a)abort (A)abort with dump (c)continue (p)roc info (i)nfo
(l)oaded (v)ersion (k)ill (D)b-tables (d)istribution
AtomVM + form_demo git:(local_live_view_phoenix_connection) x

Webサーバが停止した後も
そのまま動作します。

AtomVM



2026年

- 2026年はハードウェア対応に注力します
- 2026年中に v0.7 と v0.8 をリリースすることを目指します
- 拡張されたハードウェア対応を多くの人が利用できるようにしたい

もう少し詳しく見ていきます

v0.7

- ここまで述べた機能を含む安定版リリース
- かなり近いところまで来ています
- まだ足りない変更
 - エラー処理の改善（スタックトレースに関数引数が表示するなど）
 - 新しいモジュールローダ（フラッシュ使用量の削減につながるはず）
 - その他の小さなPRや修正
 - OTP 26未満／古いElixir対応コードの削除
 - Bitstring（例：`<<42::size(7)>>`）は v0.7 か v0.8 か未確定

v0.8

- プラットフォーム、ハードウェア対応、ツール群が中心
- コアVMに大きな変更は入らない見込み



各種ハードウェアや
プラットフォームへの
サポートは？

Zephyr

- 課題：generic_unix、esp32、rp2040 など、特定プラットフォーム向けのポートが複数あります
- Nordic Semiconductor、NXPなど、未対応のシリコンプラットフォームはまだたくさんあります
- それぞれに個別のプラットフォーム対応を書くのは大変です
- Zephyrはハードウェア抽象化層（HAL）を提供し、プラットフォーム対応を一度書けば済むようにできます

Zephyr：これまでの取り組み

- 2023年に初期検証に着手
- 当初の課題
 - Zephyrプロジェクト内でAtomVMコアをどうコンパイルするか
 - 2つの異なるビルド方式をどう統合するか
 - 時間と優先度

Zephyr : 次にやること

- 今年、Zephyr 対応を再開
- 私の最優先のタスクとして進める予定
- Zephyr経験者の協力を歓迎
 - 実装の貢献はもちろん、PRレビューも頂ければ大きな助けになります

Zephyr：期待できること

- Zephyr 対応により、幅広い SoC（システムオンチップ）で AtomVM を動かせる可能性が広がります
- 対応候補：Microchip、Nordic Semiconductor、NXP、Silicon Labs、STMicroelectronics、Renesas など
- 実際に対応できる範囲は、Zephyr側のポートと利用可能なRAMに依存します



Apache NuttX ?

- Zephyrの代替候補として位置づけ
- AtomVM 上の開発者体験は、できる限り同じにする方針
- 選択肢が増える

AtomVM と Nerves

- Nerves との一定の互換性を目指す
- 理想：同じ I2C / SPI / GPIO ドライバを、変更なしで両方で動かす
 - 例：温度センサーなどのデバイスドライバ
- 互換性の範囲や到達点はまだ検討中
 - できる限り良い互換性を提供したい

開発支援ツールと開発体験

- 現状：exatomvm などで、面倒な作業を自動化できている
 - 例：mix atomvm.esp32.install のような便利タスク
- ただ「便利」だけでは足りない
 - 目標：「とても良い」開発者体験（迷わず進められること）
- さらなる自動化
- 事前ビルド済みイメージを増やす
 - 安定版リリースとmainブランチの両方
 - AtomGLなどネイティブ拡張を含むもの

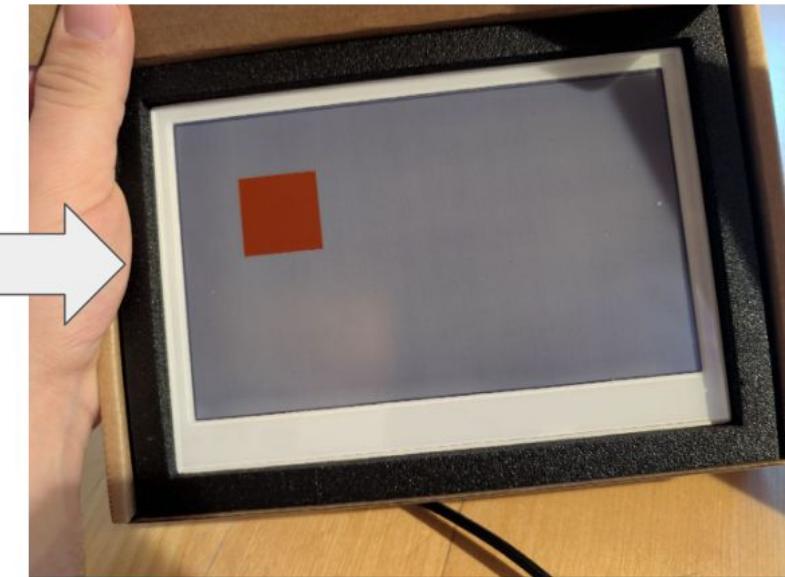
AtomGL

- AtomVMは、コアVMだけではなく、周辺コンポーネントを含むエコシステムとして提供しています。その一つが、AtomGLです。
- AtomGLはディスプレイ／ディスプレイコントローラの抽象化を提供。
- 対応ディスプレイは増えてきています。
 - SPIカラーLCD : ILI9341 / ILI9342C、ILI9486 / ILI9488、ST7789 / ST7796
 - SPIモノクロLCD : Sharp Memory LCD
 - SPIカラー電子ペーパー : Waveshare 5.65" ACeP 7色、E Ink Spectra 6 (reTerm E1002)
 - I²CモノクロOLED : SSD1306 / SH1106
 - 更に追加可能

AtomGLの宣言的UI

「どう描くか」ではなく、
「何を表示するか」をAtomGLに伝える

```
def handle_info(:show_hello, state) do
  items = [
    {:text, 10, 20, :default16px, 0x000000, @bg_color,
     "Hello"},
    {:rect, 100, 100, 128, 128, 0xFF0000},
    {:rect, 0, 0, 800, 480, @bg_color}
  ]
  {:noreply, state, [{:push, items}]}
end
```



AtomGLの内部動作

- アイテム一覧は末尾（最下層）から先頭（最上層）の順で描画
- 最上層へのアイテム追加は、[new_top_item | items]
- フレームバッファ用を持たず、必要になった時点で走査線ごとに描画
- この方法なら、通常ならメモリ不足になりがちなディスプレイも扱える
- 色変換、ディザリングなどもすべて必要になった時点で実行

v1.0に向けて

v1.0で目指す状態

- APIを凍結し、安定した互換性を提供
- OTPの中核機能がそろう
- 主要プラットフォームで必須機能を揃える

要するに、v0.xだから本番で使えないという意味ではありません

すでにメンテナンスブランチでバグ修正版リリースを継続している

スポンサー

スポンサーの支援により、AtomVM エコシステムに副業・専業で携わる人々が生まれ、Popcorn のすべての素晴らしい成果につながりました。

ありがとうございます！



Dashbit



新しい年、新しいスポンサーを募集します

開発スピードを加速させるため、新しいスポンサーを募集しています。

プロジェクトは日々複雑さを増し、毎日PRやIssueが届き、本番で使う開発者も現れています。

これまで余暇で進めてきましたが、それでは限界があるため、今年からはフルタイムでAtomVMに取り組みます。

小規模な支援でも大きな助けになります。ご協力をお待ちしています。



Join Us

<https://atomvm.org/>

Discord: <https://discord.gg/QA7fNjm9Nw>

Telegram: <https://t.me/atomvm>

Documentation: <https://doc.atomvm.org/>

ご清聴ありがとうございました



AtomVM 支援のご案内



ご連絡はお気軽にどうぞ