環境変数リリース設定

セクターブレンダン　センヌエ株式会社 代表取締役社長

tokyo.ex #6

2016年9月22日　木曜日

sgeos.github.io

Storing Elixir Release Configuration in Environment Variables with Distillery

Storing Elixir Release Configuration in Environment Variables with exrm

リリース環境

環境変数

リリース設置

Elixir アプリ
Phoenix アプリ

データベース

発表内容

POST JSON/HTTP

GET JSON/HTTP

distillery リリース

mix は使えないよ！

テストアプリ作成

```
mix phoenix.new phoenix_environment_settings --no-brunch
cd phoenix_environment_settings
mix ecto.create
mix phoenix.gen.json Memo memos title:string body:string
```

テストアプリ作成

```
mix phoenix.new phoenix_environment_settings --no-brunch
cd phoenix_environment_settings
mix ecto.create
mix phoenix.gen.json Memo memos title:string body:string
```

web/router.ex 一部

```
  scope "/api", PhoenixEnvironmentSettings do
    pipe_through :api

    resources "/memos", MemoController, except: [:new, :edit]
  end
```

テストアプリ作成

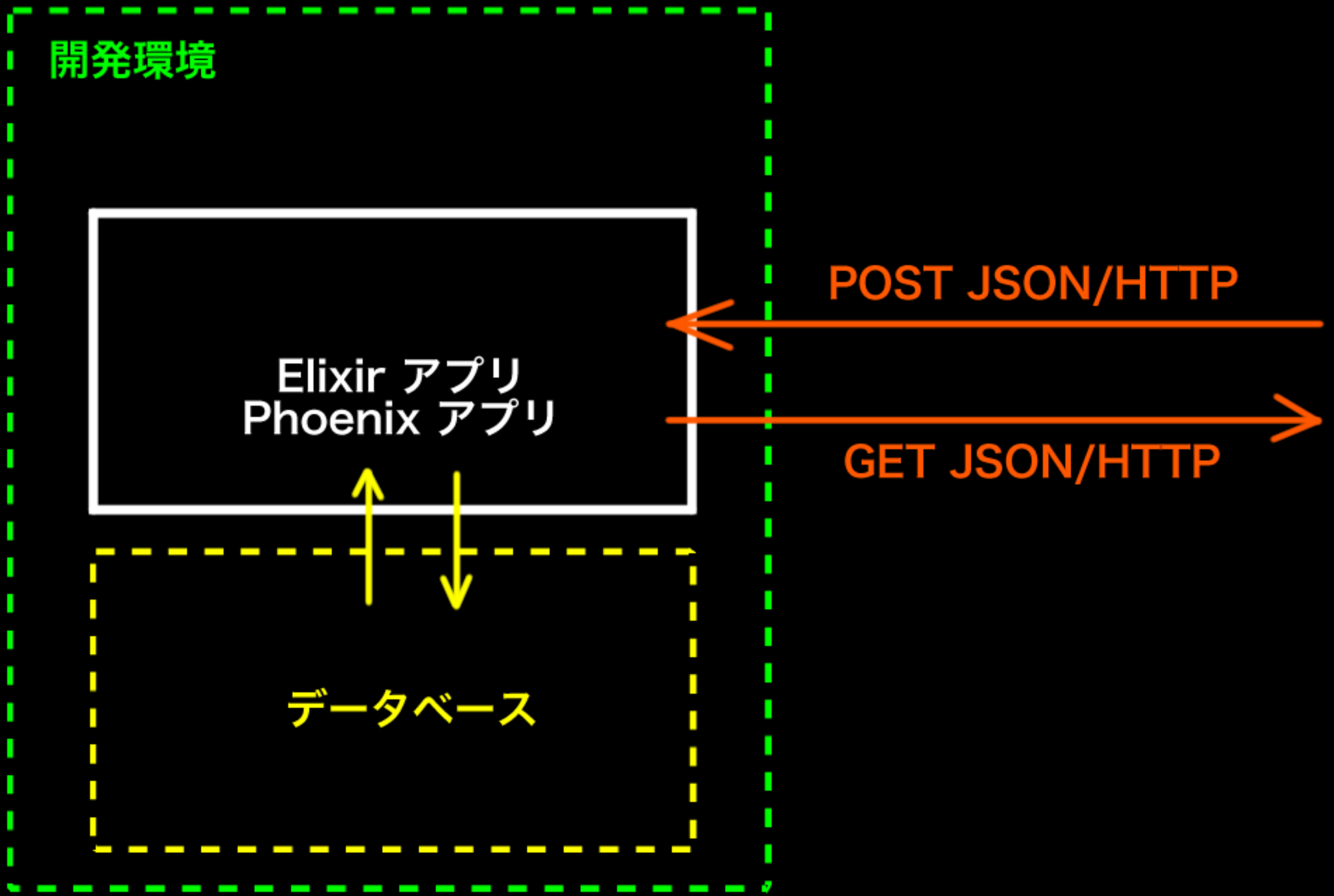
```
mix phoenix.new phoenix_environment_settings --no-brunch
cd phoenix_environment_settings
mix ecto.create
mix phoenix.gen.json Memo memos title:string body:string
```

web/router.ex 一部

```
  scope "/api", PhoenixEnvironmentSettings do
    pipe_through :api

    resources "/memos", MemoController, except: [:new, :edit]
  end
```

```
mix ecto.migrate
mix test
mix phoenix.server
curl -H 'Content-Type: application/json' -X POST \
  -d '{"memo": {"title": "メモのタイトル", "body": "メモの内容で す。"}}' \
  http://localhost:4000/api/memos
curl -H 'Content-Type: application/json' http://localhost:4000/api/memos/1
```


開発環境
Elixir アプリ
Phoenix アプリ
データベース
POST JSON/HTTP
GET JSON/HTTP

環境変数設定

```
export NODE_NAME=leaf
export COOKIE=thin_mints
export DB_USER=phoenix_environment_settings
export DB_PASSWORD=phoenix_environment_settings_password
export DB_NAME=phoenix_environment_settings_prod
export DB_HOST=localhost
export HOST=host.example.org
export PORT=7777
export SECRET_KEY_BASE=$(elixir -e ":crypto.strong_rand_bytes(48) |>
  Base.encode64 |> IO.puts")
```

ウェブアプリ設定

config/prod.exs 一部

```
config :phoenix_environment_settings, PhoenixEnvironmentSettings.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: "${DB_USER}",
  password: "${DB_PASSWORD}",
  database: "${DB_NAME}",
  hostname: "${DB_HOST}",
  pool_size: 20
```

ウェブアプリ設定

config/prod.exs 全部

```
use Mix.Config

config :phoenix_environment_settings, PhoenixEnvironmentSettings.Endpoint,
  http: [port: {:system, "PORT"}],
  url: [host: "${HOST}", port: {:system, "PORT"}],
  cache_static_manifest: "priv/static/manifest.json",
  server: true,
  root: ".",
  version: Mix.Project.config[:version]

config :logger, level: :info

config :phoenix_environment_settings, PhoenixEnvironmentSettings.Endpoint,
  secret_key_base: "${SECRET_KEY_BASE}"

config :phoenix_environment_settings, PhoenixEnvironmentSettings.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: "${DB_USER}",
  password: "${DB_PASSWORD}",
  database: "${DB_NAME}",
  hostname: "${DB_HOST}",
  pool_size: 20
```

ウェブアプリ設定

config/prod.exs 一部

```
config :phoenix_environment_settings, PhoenixEnvironmentSettings.Endpoint,
http: [port: {:system, "PORT"}],
url: [host: "${HOST}", port: {:system, "PORT"}],
cache_static_manifest: "priv/static/manifest.json",
server: true,
root: ".",
version: Mix.Project.config[:version]
```

ウェブアプリ設定

config/prod.secret.exs 全部

```
use Mix.Config

config :phoenix_environment_settings, PhoenixEnvironmentSettings.Endpoint,
  secret_key_base: "${SECRET_KEY_BASE}"

config :phoenix_environment_settings, PhoenixEnvironmentSettings.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: "${DB_USER}",
  password: "${DB_PASSWORD}",
  database: "${DB_NAME}",
  hostname: "${DB_HOST}",
  pool_size: 20
```

ウェブアプリ設定

config/dev.exs 一部

```
config :phoenix_environment_settings, PhoenixEnvironmentSettings.Endpoint,
  http: [port: System.get_env("PORT") || 4000],

# ... ファイルの最後の方 ...

# Configure your database
config :phoenix_environment_settings, PhoenixEnvironmentSettings.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: System.get_env("DB_USER") || "postgres",
  password: System.get_env("DB_PASSWORD") || "postgres",
  database: System.get_env("DB_NAME") || "phoenix_environment_settings_dev",
  hostname: System.get_env("DB_HOST") || "localhost",
  pool_size: 10
```

リリース生成

mix.exs 一部

```
  defp deps do
    [{:phoenix, "~> 1.2.1"},
     {:phoenix_pubsub, "~> 1.0"},
     {:phoenix_ecto, "~> 3.0"},
     {:postgrex, ">= 0.0.0"},
     {:phoenix_html, "~> 2.6"},
     {:phoenix_live_reload, "~> 1.0", only: :dev},
     {:gettext, "~> 0.11"},
     {:distillery, "~> 0.9"}, # 新しく追加した
     {:cowboy, "~> 1.0"}]
  end
```

リリース生成

```
mix deps.get
mix deps.compile
mix release.init
```

リリース生成

rel/vm.args 全部

```
## ノード名
-name ${NODE_NAME}

## 分散 erlang の為のクッキー
-setcookie ${COOKIE}

## アプリ設定
-phoenix_environment_settings port ${PORT}
```

リリース生成

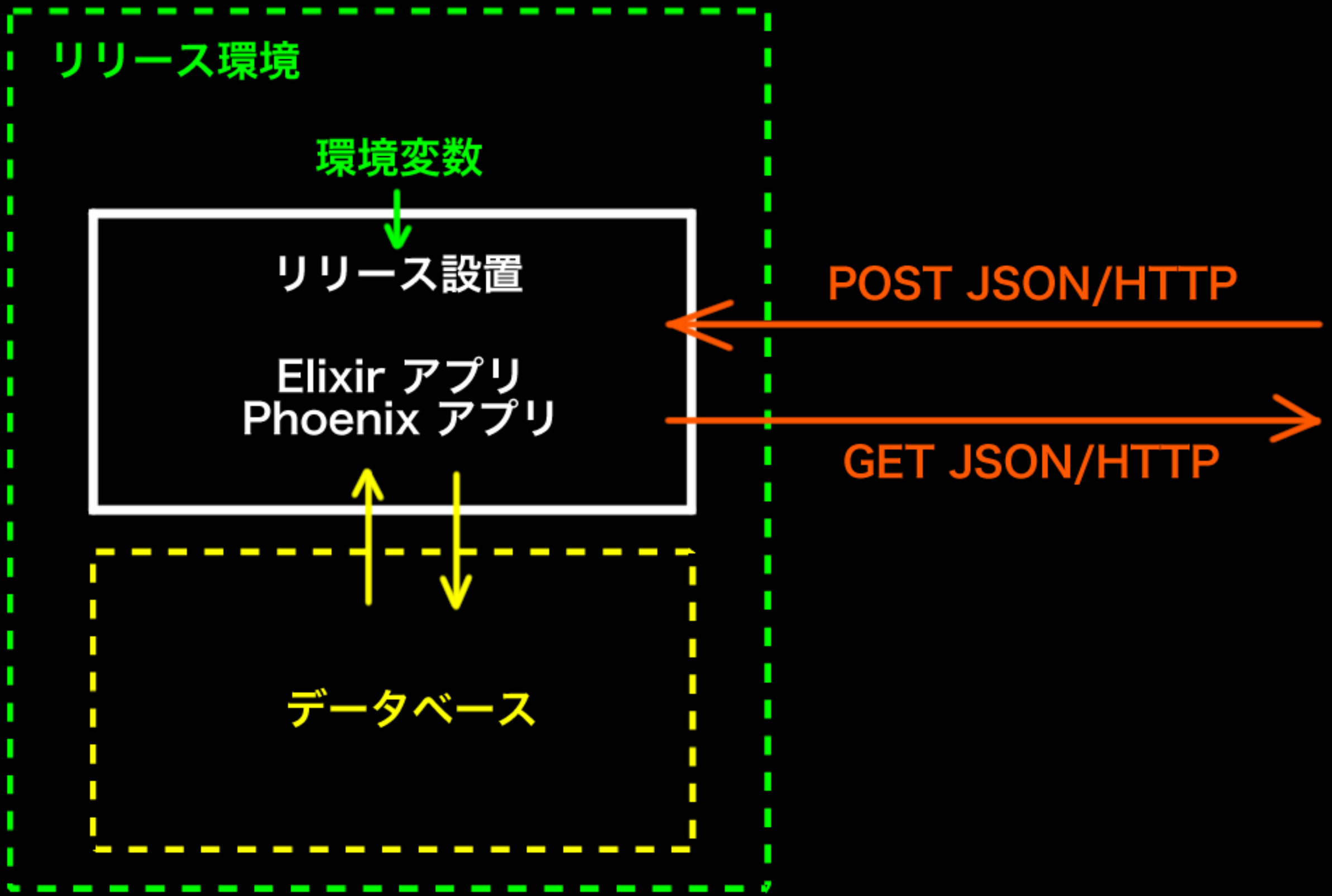
rel/sys.config 全部

```
[
  {phoenix_environment_settings, [
    {port, "${PORT}"}
  ]}
].
```

config/config.exs
config/prod.exs
config/prod.secret.exs

リリース生成

```
export REPLACE_OS_VARS=true
```


リリース環境
環境変数
リリース設置
Elixir アプリ
Phoenix アプリ
POST JSON/HTTP
GET JSON/HTTP
データベース

リリース環境で mix は使えません！

リリースタスク追加

lib/release_tasks.ex 全部

```
# `mix ecto.migrate` の同等タスク
defmodule :release_tasks do
  def migrate do
    {:ok, _} = Application.ensure_all_started(:phoenix_environment_settings)
    path = Application.app_dir(:phoenix_environment_settings,
      "priv/repo/migrations")
    Ecto.Migrator.run(PhoenixEnvironmentSettings.Repo, path, :up, all: true)
    :init.stop()
  end
end
```

```
REPLACE_OS_VARS=true \
  rel/phoenix_environment_settings/bin/phoenix_environment_settings \
  command release_tasks migrate
```

リリースタスク追加

```
# `mix ecto.create`の同等コマンド、PostgreSQL版
psql -c "CREATE USER ${DB_USER} WITH PASSWORD '${DB_PASSWORD}';"
createdb "${DB_NAME}"
psql -c "GRANT ALL PRIVILEGES ON DATABASE ${DB_NAME} to ${DB_USER};"

# `mix ecto.drop`の同等コマンド、PostgreSQL版
dropdb "${DB_NAME}"
dropuser "${DB_USER}"

# PostgreSQL 対話型端末
PGPASSWORD="${DB_PASSWORD}" psql -U "${DB_USER}" "${DB_NAME}"
```

リリースタスク追加

```
# `mix ecto.create` の同等コマンド、MySQL版
mysql -e "CREATE USER '${DB_USER}'@'localhost' IDENTIFIED BY '${DB_PASSWORD}';"
mysql -e "CREATE DATABASE ${DB_NAME};"
mysql -e "GRANT ALL PRIVILEGES ON ${DB_NAME}.* TO ‘${DB_USER}'@'localhost' \
  IDENTIFIED BY '${DB_PASSWORD}';"
mysql -e "FLUSH PRIVILEGES;"

# `mix ecto.drop` の同等コマンド、MySQL版
mysql -e "DROP DATABASE ${DB_NAME};"
mysql -e "DROP USER '${DB_USER}'@'localhost';"

# MySQL 対話型端末
mysql -u"${DB_USER}" -p"${DB_PASSWORD}" "${DB_NAME}"
```

改造コマンド追加

```
mkdir rel/commands
```

rel/commands/ecto_migrate 全部

```
#!/usr/bin/env sh

# `mix ecto.migrate` の同等コマンド
"${SCRIPT}" command release_tasks migrate
```

改造コマンド追加

rel/commands/ecto_create 全部

```
#!/usr/bin/env sh

# `mix ecto.create` の同等コマンド、PostgreSQL版
psql -c "CREATE USER ${DB_USER} WITH PASSWORD '${DB_PASSWORD}';" &&
createdb "${DB_NAME}" &&
psql -c "GRANT ALL PRIVILEGES ON DATABASE ${DB_NAME} to ${DB_USER};" &&
echo "The database '${DB_NAME}' and role '${DB_USER}' have been created."
```

rel/commands/ecto_drop 全部

```
#!/usr/bin/env sh

# `mix ecto.drop` の同等コマンド、PostgreSQL版
dropdb "${DB_NAME}" &&
dropuser "${DB_USER}" &&
echo "The database '${DB_NAME}' and role '${DB_USER}' have been dropped."
```

改造コマンド追加

rel/commands/ecto_create 全部

```
#!/usr/bin/env sh

# `mix ecto.create` の同等コマンド、MySQL版
mysql -e "CREATE USER '${DB_USER}'@'localhost' IDENTIFIED BY '${DB_PASSWORD}';" &&
mysql -e "CREATE DATABASE ${DB_NAME};" &&
mysql -e "GRANT ALL PRIVILEGES ON ${DB_NAME}.* TO '${DB_USER}'@'localhost' \
  IDENTIFIED BY '${DB_PASSWORD}';" &&
mysql -e "FLUSH PRIVILEGES;" &&
echo "The database '${DB_NAME}' and role '${DB_USER}' have been created."
```

rel/commands/ecto_drop 全部

```
#!/usr/bin/env sh

# `mix ecto.drop` の同等コマンド、MySQL版
mysql -e "DROP DATABASE ${DB_NAME};" &&
mysql -e "DROP USER '${DB_USER}'@'localhost';" &&
echo "The database '${DB_NAME}' and role '${DB_USER}' have been dropped."
```

改造コマンド追加

rel/config.exs 全部

```
use Mix.Releases.Config,
    default_release: :default,
    default_environment: :prod

environment :prod do
  set include_erts: true
  set include_src: false
  set commands: [
    "ecto.migrate": "rel/commands/ecto_migrate",
    "ecto.create": "rel/commands/ecto_create",
    "ecto.drop": "rel/commands/ecto_drop"
  ]
end

release :phoenix_environment_settings do
  set version: current_version(:phoenix_environment_settings)
end
```

リリース実行

```
# rm -rf rel/phoenix_environment_settings/ # 任意、古いリリースファイル強制削除
MIX_ENV=prod mix compile
# brunch build --production # brunch を使っている場合
MIX_ENV=prod mix phoenix.digest
MIX_ENV=prod mix release --env=prod
```

リリース実行

```
# rm -rf rel/phoenix_environment_settings/ # 任意、古いリリースファイル強制削除
MIX_ENV=prod mix compile
# brunch build --production # brunch を使っている場合
MIX_ENV=prod mix phoenix.digest
MIX_ENV=prod mix release --env=prod
```

```
REPLACE_OS_VARS=true \
  rel/phoenix_environment_settings/bin/phoenix_environment_settings ecto.create
REPLACE_OS_VARS=true \
  rel/phoenix_environment_settings/bin/phoenix_environment_settings ecto.migrate
```

リリース実行

```
REPLACE_OS_VARS=true \
  rel/phoenix_environment_settings/bin/phoenix_environment_settings start
```

リリース実行

```
REPLACE_OS_VARS=true \
  rel/phoenix_environment_settings/bin/phoenix_environment_settings start
```

```
curl -H 'Content-Type: application/json' -X POST \
  -d '{"memo": {"title": "メモα", "body": "メモαの内容で す。"}}' \
  "http://localhost:${PORT}/api/memos"

curl -H 'Content-Type: application/json' -X POST \
  -d '{"memo": {"title": "メモβ", "body": "メモβの内容で す。"}}' \
  "http://localhost:${PORT}/api/memos"

curl "http://localhost:${PORT}/api/memos"
```

リリース実行

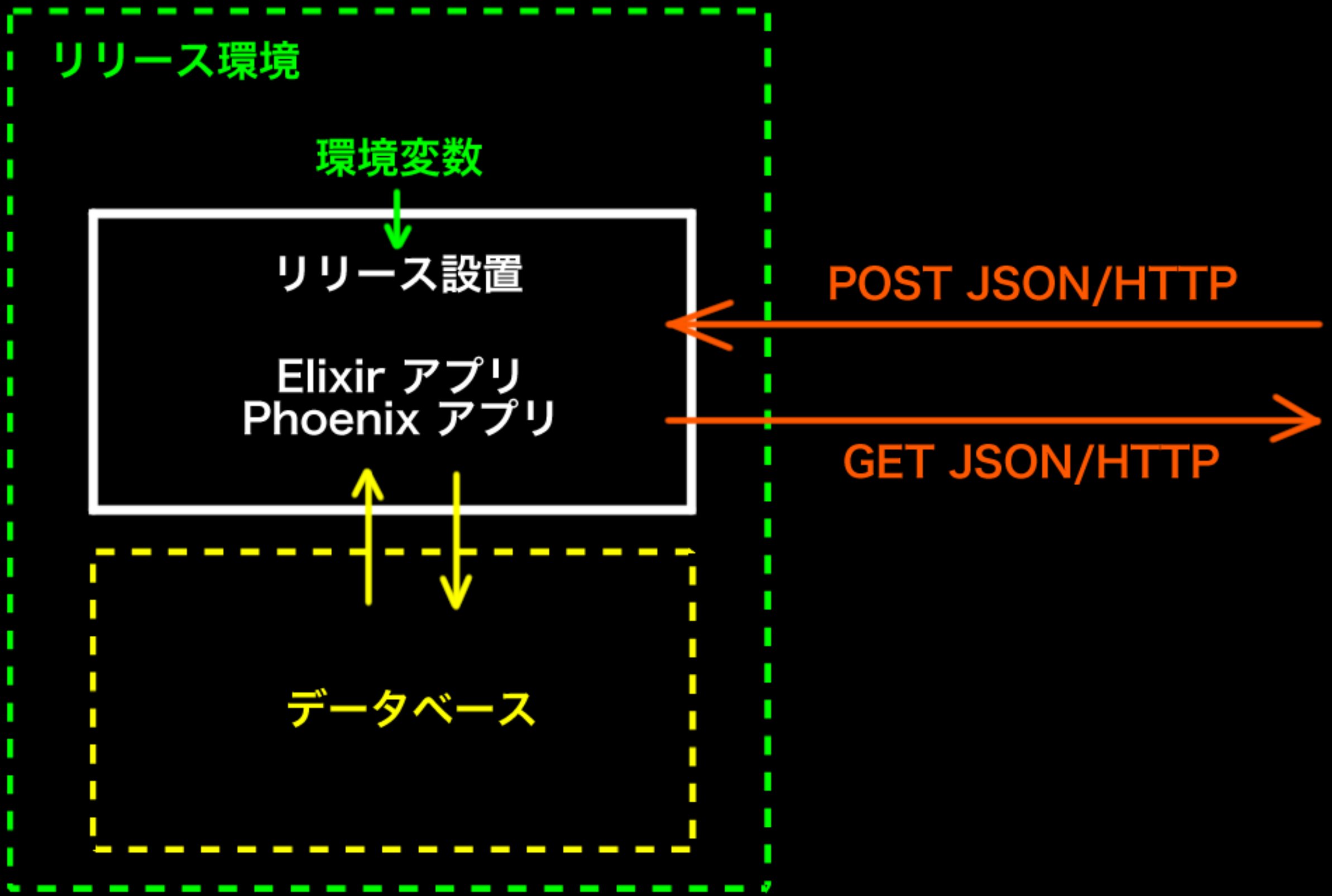
```
REPLACE_OS_VARS=true \
  rel/phoenix_environment_settings/bin/phoenix_environment_settings start
```

```
curl -H 'Content-Type: application/json' -X POST \
  -d '{"memo": {"title": "メモα", "body": "メモαの内容で す。"}}' \
  "http://localhost:${PORT}/api/memos"

curl -H 'Content-Type: application/json' -X POST \
  -d '{"memo": {"title": "メモβ", "body": "メモβの内容で す。"}}' \
  "http://localhost:${PORT}/api/memos"

curl "http://localhost:${PORT}/api/memos"
```

```
REPLACE_OS_VARS=true \
  rel/phoenix_environment_settings/bin/phoenix_environment_settings stop
```


リリース環境
環境変数
リリース設置
Elixir アプリ
Phoenix アプリ
POST JSON/HTTP
GET JSON/HTTP
データベース

ウェブアプリ設定

config/prod.exs 一部

```
config :phoenix_environment_settings, PhoenixEnvironmentSettings.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: "${DB_USER}",
  password: "${DB_PASSWORD}",
  database: "${DB_NAME}",
  hostname: "${DB_HOST}",
  pool_size: 20
```

config/dev.exs 一部

```
config :phoenix_environment_settings, PhoenixEnvironmentSettings.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: System.get_env("DB_USER") || "postgres",
  password: System.get_env("DB_PASSWORD") || "postgres",
  database: System.get_env("DB_NAME") || "phoenix_environment_settings_dev",
  hostname: System.get_env("DB_HOST") || "localhost",
  pool_size: 10
```

ウェブアプリ設定

lib/config.ex 全部

```
defmodule Config do
  def get(app, key, default \\ nil) when is_atom(app) and is_atom(key) do
    case Application.get_env(app, key) do
      {:system, env_var} ->
        case System.get_env(env_var) do
          nil -> default
          val -> val
        end
      {:system, env_var, preconfigured_default} ->
        case System.get_env(env_var) do
          nil -> preconfigured_default
          val -> val
        end
      nil ->
        default
      val ->
        val
    end
  end

  def get_integer(app, key, default \\ nil) do
    case get(app, key, nil) do
      nil -> default
      n when is_integer(n) -> n
      n ->
        case Integer.parse(n) do
          {i, _} -> i
          :error -> default
        end
    end
  end
end
```

ウェブアプリ設定

lib/config.exs 全部

```
defmodule Config do
  def get(app, key, default \\ nil) when is_atom(app) and is_atom(key) do
    case Application.get_env(app, key) do
      {:system, env_var} ->
        case System.get_env(env_var) do
          nil -> default
          val -> val
        end
      {:system, env_var, preconfigured_default} ->
        case System.get_env(env_var) do
          nil -> preconfigured_default
          val -> val
        end
      nil ->
```

https://gist.github.com/bitwalker/a4f73b33aea43951fe19b242d06da7b9

```
        val
    end
  end

  def get_integer(app, key, default \\ nil) do
    case get(app, key, nil) do
      nil -> default
      n when is_integer(n) -> n
      n ->
        case Integer.parse(n) do
          {i, _} -> i
          :error -> default
        end
    end
  end
end
```

ウェブアプリ設定

config/config.exs 一部

```
config :phoenix_environment_settings,
  ecto_repos: [PhoenixEnvironmentSettings.Repo],
  welcome_message: {:system, "WELCOME_MESSAGE", "Hello, world!"},
  magic_number: {:system, "MAGIC_NUMBER", 42}
```

ウェブアプリ設定

config/config.exs 一部

```
config :phoenix_environment_settings,
  ecto_repos: [PhoenixEnvironmentSettings.Repo],
  welcome_message: {:system, "WELCOME_MESSAGE", "Hello, world!"},
  magic_number: {:system, "MAGIC_NUMBER", 42}
```

iex -S mix

```
Config.get :phoenix_environment_settings, :welcome_message
Config.get_integer :phoenix_environment_settings, :magic_number
```

ウェブアプリ設定

config/config.exs 一部

```
config :phoenix_environment_settings,
  ecto_repos: [PhoenixEnvironmentSettings.Repo],
  welcome_message: {:system, "WELCOME_MESSAGE", "Hello, world!"},
  magic_number: {:system, "MAGIC_NUMBER", 42}
```

iex -S mix

```
Config.get :phoenix_environment_settings, :welcome_message
Config.get_integer :phoenix_environment_settings, :magic_number
```

```
export WELCOME_MESSAGE=“ようこそ！逃げても良いけど、出口は有りませんよ。”
export MAGIC_NUMBER=-1
```

ウェブアプリ設定

config/config.exs 一部

```
config :phoenix_environment_settings,
  ecto_repos: [PhoenixEnvironmentSettings.Repo],
  welcome_message: {:system, "WELCOME_MESSAGE", "Hello, world!"},
  magic_number: {:system, "MAGIC_NUMBER", 42}
```

iex -S mix

```
Config.get :phoenix_environment_settings, :welcome_message
Config.get_integer :phoenix_environment_settings, :magic_number
```

```
export WELCOME_MESSAGE="ようこそ！逃げても良いけど、出口は有りませんよ。"
export MAGIC_NUMBER=-1
```

iex -S mix

```
Config.get :phoenix_environment_settings, :welcome_message
Config.get_integer :phoenix_environment_settings, :magic_number
```

其の他の考察

export DB_POOL_SIZE=20 # × 残念ながらリリース環境で整数やアトムは使えない！

其の他の考察

export DB_POOL_SIZE=20 # × 残念ながらリリース環境で整数やアトムは使えない！

config/prod.exs か **config/prod.secret.exs** 一部

```
  pool_size: "${DB_POOL_SIZE}"
```

其の他の考察

export DB_POOL_SIZE=20 # × 残念ながらリリース環境で整数やアトムは使えない！

config/prod.exs か **config/prod.secret.exs** 一部

```
  pool_size: "${DB_POOL_SIZE}"
```

残念ながらリリース環境で整数やアトムは使えない！

其の他の考察

export DB_POOL_SIZE=20 # × 残念ながらリリース環境で整数やアトムは使えない！

config/prod.exs か **config/prod.secret.exs** 一部

```
  pool_size: "${DB_POOL_SIZE}"
```

残念ながらリリース環境で整数やアトムは使えない！

config/dev.exs 一部

```
  pool_size: (System.get_env("DB_POOL_SIZE") || "10") |> String.to_integer
```

exrm と distillery の違い

distillery
- REPLACE_OS_VARS=true
- 改造コマンド有り
- rel/sys.config か config/config.exs が使える

exrm
- RELX_REPLACE_OS_VARS=true
- 改造コマンド無し
- rel/sys.config と config/config.exs は同時に使える

次のステップ

edeliver

conform