PHP コードを隔離された環境で安全に動かす (on WebAssembly)

nsfisis (いまむら)

第 157 回 PHP 勉強会@東京

自己紹介

nsfisis (いまむら)

@ デジタルサーカス株式会社

任意コードを安全に実行したい

任意の (危険かもしれない) コードを実行したい

任意コードを安全に実行したい

任意の (危険かもしれない) コードを実行したい

WebAssembly による隔離

WebAssembly とは

ポータブルな仮想マシンのコード (別名 Wasm)

元々はブラウザにおける処理の高速化を目的として制定されたが、さまざまなプラットフォームに活躍の場を広げつつある

プラットフォームに依存しない、言語に依存しない、外から隔離された実行環境

先行研究

PHP の処理系を WebAssembly にコンパイル

PHPerKaigi 2023:「PHP をブラウザで動かす技術」
Zenn:「WebAssembly を使った PHP のリアルタイム Playground を作りました。」
WordPress Playground

WebAssembly にビルドする

C 言語でエントリポイントを書く

```c
#include <emscripten.h>
#include <Zend/zend_execute.h>


int EMSCRIPTEN_KEEPALIVE php_wasm_run(const char* code) {
    zend_result result =
        zend_eval_string_ex(code, NULL, "php.wasm code", 1);
    return result == SUCCESS ? 0 : 1;
}
```

Emscripten

C 等から WebAssembly へのコンパイルに使用するツールチェイン

LLVM を利用し、C や C++ などから WebAssembly のコードを生成する

WebAssembly にビルドする

```
$ ./buildconf
$ ./configure
$ make
```

PHP の通常のビルド手順

WebAssembly にビルドする

```
$ ./buildconf
$ emconfigure ./configure
$ emmake make
```

configure の代わりに emconfigure

make の代わりに emmake

ビルドを楽にするために

```
$ emconfigure configure \
    --disable-all \
    --disable-mbregex \
    --disable-fiber-asm \
    --disable-cli \
    --disable-cgi \
    --disable-phpdbg \
    --without-iconv \
    --without-libxml \
    --without-pcre-jit \
    --without-pdo-sqlite \
    --without-sqlite3
```

なるべく disable/without して、ビルド対象を減らす

ビルドを楽にするために

```
$ EMCC_CFLAGS='-s ERROR_ON_UNDEFINED_SYMBOLS=0' \
    emmake make
```

ERROR_ON_UNDEFINED_SYMBOLS を 0 にして、未定義のシンボルがあってもエラーにならないように

WebAssembly にビルドする

```
$ emcc \
    -s ERROR_ON_UNDEFINED_SYMBOLS=0 \
    -s ENVIRONMENT=node \
    -s INITIAL_MEMORY=16777216 \
    -s EXPORT_ES6=1 \
    -s INVOKE_RUN=0 \
    -s MODULARIZE=1 \
    -o php-wasm.js \
    php-wasm.o libphp.a
```

ENVIRONMENT で動かす環境を指定、INITIAL_MEMORY で確保するメモリを指定、等

php-wasm.wasm と php-wasm.js が生成される

使ってみる

```
import PHPWasm from './php-wasm.js'


const { ccall } = await PHPWasm();
const result = ccall(
  'php_wasm_run',
  'number', ['string'],
  [`echo "Hello, World!";`],
);
console.log(`exit code: ${result}`);
```

`ccall` に、関数名、返り値の型、仮引数の型、実引数を渡して呼び出す

まとめ

- Emscripten を使って、PHP の処理系を WebAssembly にコンパイルできる
- WebAssembly を使って、環境を隔離できる
- Git リポジトリ : https://github.com/nsfisis/tiny-php.wasm
  - Docker があれば構築可能