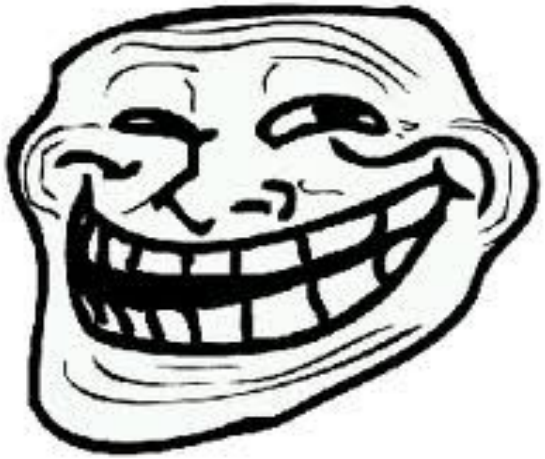
Демонизированный PHP - before it was cool

Arvīds Godjuks
Areto Development
Москва, 2015

О себе

- PHP
  - Разработчик?
  - Архитектор?
  - Web Developer?
  - Сделай то, незнаю что!

О себе



Если вы...

- Думаете потоками
- Писали на phpDaemon
- Использовали libevent
- Просто человекоподобный робот

Доклад – баян ;)


ONE DOES NOT
SIMPLY
WRITE A DEAMON IN
PHP
memegenerator.net

Лёгкий экскурс в историю

PHP < 5.3

- GC
- Утечки памяти
- Баги

PHP >= 5.3

- GC как ультимативное средство
- Стабилизация 5-ки как таковой
- Libevent, phpDaemon – поспособствовали развитию

Однако

В большинстве случаев – обыкновенно отсутствие знаний и/или интереса

Демон – это не сложно

Практически не отличается от того, как это делается в C/C++ под Linux.

Демон – это не сложно

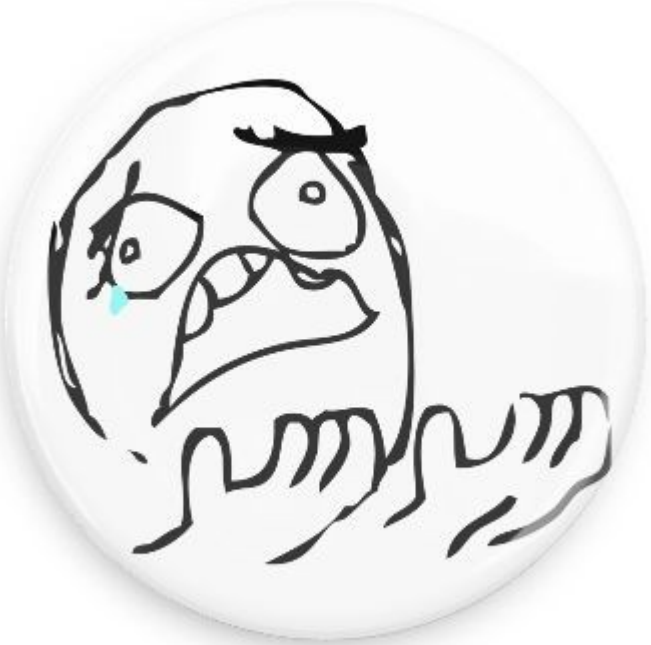
Материал для разработки демонизированных приложений на C/C++ похож на то, как это нужно делать в PHP.

Демон – это не сложно

Но дъявол в деталях :)

Хорошо, убедил...

НО ВО ИМЯ КТУЛХУ



Архитектура

- Единая кодовая база
  - Бизнес логика
  - Валидация
  - Переиспользование кода

Издержки

- Время на синхронизацию проектов
- Поиск персонала
  - Не редко большие зарплаты
- Обучение существующего персонала

С точки зрения здравого смысла, гораздо быстрее и дешевле реализовать прототип на PHP и судить о необходимости смены инструмента по результатам

PHP - быстрый

- Особенно с выходом 5.4
- Ну а тесты PHP7 вообще показывают цифры, в которые не сразу вериться
  - 50%-90% прироста скорости

PHP - быстрый

- Тонкая прослойка над C/C++ библиотеками
  - Иногда накладные расходы минимальны
- Сам язык прилично оптимизирован

Когда не стоит

- У вас много данных
  - С PHP7 сильно улучшит ситуацию
- Вам нужна математика
- Производительность любой ценой
- Ну очень большой и сложный проект
- Инстинкт говорит “ни-наааа-до!”

Когда не только можно, но и нужно

Если у вас проект на фреймворке, который имеет приличные CLI компоненты.

В таких проектах до 60-70% кода может реюзаться между WEB и CLI частями, что экономит время и силы комманды

Когда не только можно, но и нужно

Вы начинаете новый проект, в котором можете изначально заложить необходимый фундамент

PhpDaemon

- Не об этом речь
- Создание FastCGI приложения на PHP возможно и неплохая идея, но тут я вам не могу помочь.
  - Сделайте доклад, если реализовывали успешно :)

Libevent

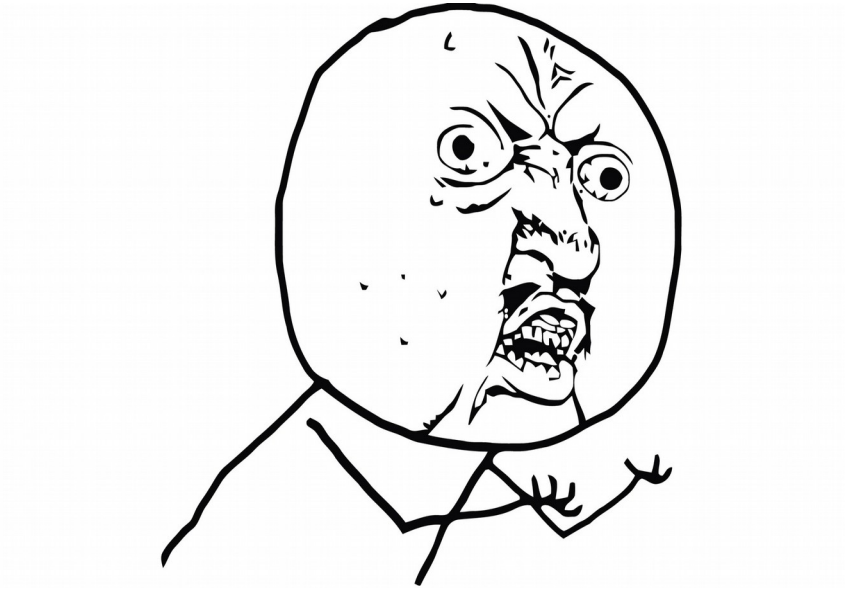
- Опять же, не о том доклад
- С ней не работал

pnctl_fork()

- Это только начало
- Хабр, StackOverflow и документация содержат все ответы, какие только можно придумать.

pnctl_fork()

- Кто , прочитав статью про сабж, задались вопросом “А что дальше?”



Что-ж, побояним...

Прежде чем начать, прочитайте

- Документацию
- Статьи
- Особенности работы PHP в CLI режиме

Полезняшки

- cli_set_process_title
  - PHP 5.5+
- PECL – proctitle
  - Если у вас PHP < 5.5

Обработка ошибок

- Логируйте всё.
- Валидируйте все данные, не зависимо откуда они пришли.
- Пишите параноидальный код.
  - Всё вокруг хочет вас нае**ть.
- error_reporting(E_ALL), только хардкор
  - Это вам не веб скриптики...

Exceptions

- Хорошо продуманная система исключений сделает вашу жизнь простой
- Возможность выбрасывать исключения разных типов и их ловить на практике очень сильно упрощает и облегчает код
  - Однако не стоит увлекаться их кол-вом – слишком много тоже плохо.
- Если у вас фреймворк – как правило у него уже есть структура исключений – не надо велосипедить.

PHP ошибки в Exception

- Весьма полезный механизм
  - Помогает записать все обстоятельства и валидно отключить демон.
  - Само собой бывают исключения

Управление памятью

- Да-да! Глаза вас не подводят.

Управление памятью

- Всегда удаляйте данные за собой
  - unset() ваш друг

Управление памятью

- По возможности явно удаляйте объекты
  - Поможет от проблем с зацикленными ссылками

Управление памятью

- Проверяйте компоненты на утечки памяти
  - Пишите патчи
  - Шлите в апстрим

Управление памятью

- GC
  - Периодически запускайте gc_collect_cycles() сами

Сетевые соединения

- У них есть таймауты
  - Их нужно конфигурировать
  - А так же учитывать в разработке
  - Поддерживать соединения в активном состоянии
- Ping? Pong!

Сетевые соединения

- Подключение к MySQL тоже сетевое соединение
  - Даже если это Unix socket
  - У MySQL сервера есть таймаут на inactivity, по которому он закрывает соединение со своей стороны
    - wait_timeout опция

Сетевые соединения

- Memcached
- Mongodb
- и.т.д.

Я обещал HHVM?

- Так и не добрался :(

Вопросы?