Производительность Python Frameworks для веб-приложений

Часть первая

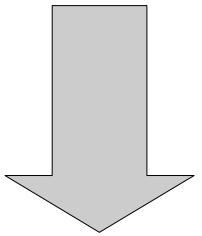
Общие сведения о современных тенденциях веб-разработки

WEB 2.0

Ретивое программирование :-)

agile development

- итерации: кодирование, тестирование, обсуждение;
- ориентир на бизнес заказчика;
- уход от формальностей в пользу успешного проекта и довольного заказчика.



ГИБКОСТЬ

Гибкая платформа

- гибкий язык;
- скорость написания кода;
- оперативное обновление и тестирование;
- производительность приложения;
- кроссплатформенность;
- правильный framework.

PHP

Perl

Java

Ruby

Python

Python + Agile = ?

Производительность Python Frameworks для веб-приложений

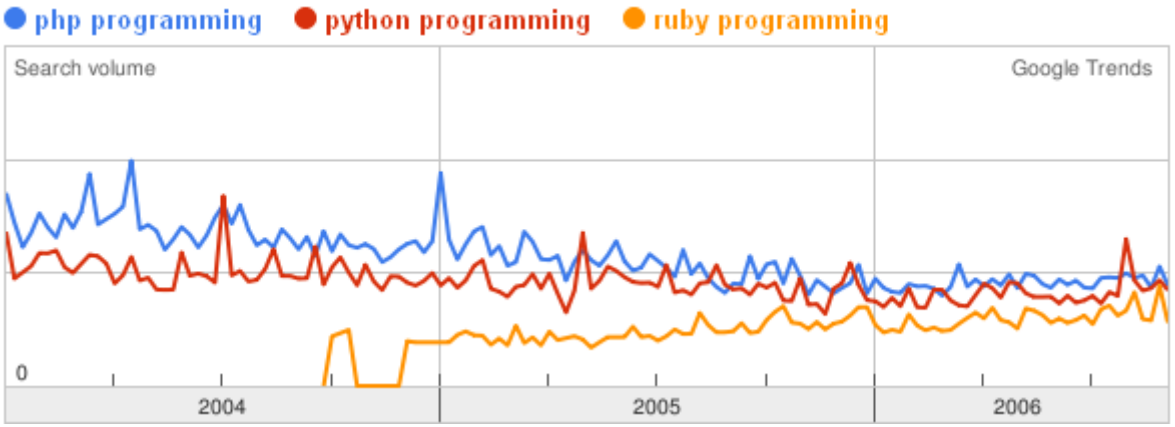
Часть вторая

Выбираем платформу или особенности интерпретатора Python

Преимущества Python

- относительно быстрый интерпретатор;
- экономная работа с памятью (сборка мусора);
- портабельность (все мажорные ОС);
- отличный ООП;
- работа с модулями;
- возраст проекта;
- большое сообщество пользователей.

Статистика Google о запросах: **php, python, ruby**



Возраст проектов

Python – 1990
PHP – 1994
Ruby – 1995

Модули в Python

- большая коллекция стандартных модулей;
- большое количество дополнительных внешних модулей, которые обычно распространяются как open source.

Есть репозиторий модулей.
Прошу на Python Cheese Shop
http://python.org/pypi

Python не течет!

В отличии от известных Perl и Ruby, у Python нет проблем со сборкой мусора, поэтому процессы не текут и соответственно память используется по назначению.

Это особо важно на продакшн проектах, т.к. там перезагрузка приложения или его понижающаяся скорость выполнения влияет прямо на успешность бизнеса.

Пример (два реальных веб-сайта)**:**
Первый построен на **Django** + FastCGI + Apache.
Второй на **RoR** + FastCGI + Apache.

В первом варианте процесс работает 11 дней без перезагрузки и имеет размер общий **21508Kb**, а размер резидента **9940Kb**.

Во втором примере процесс работает 5 дней и имеет общий размер в **212Mb**. А резидент в **9500Kb**.

Делайте выводы сами. :-)

Производительность Python Frameworks для веб-приложений

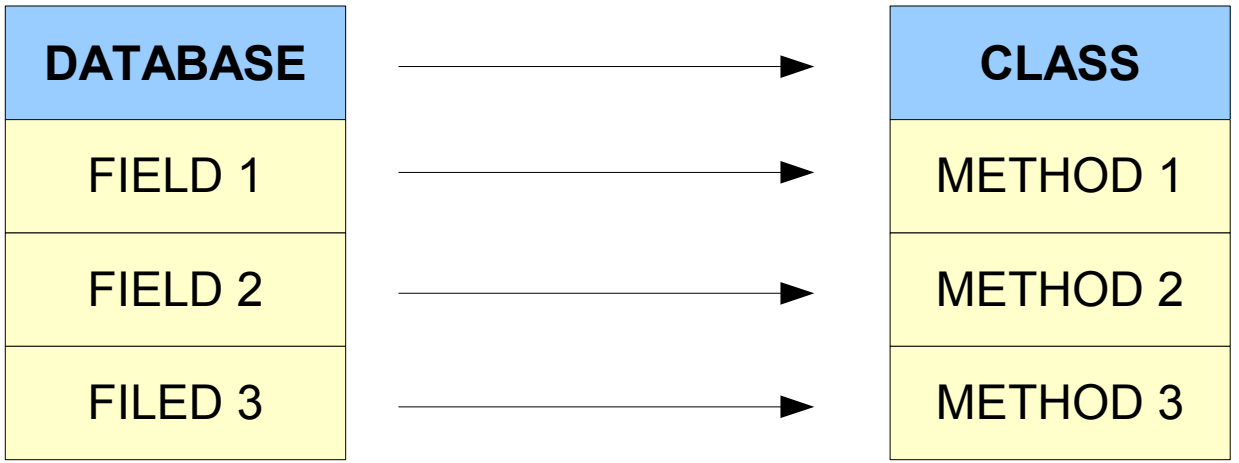
Часть третья

Существующие web frameworks для Python

Замолвим слово про **OOP** и **ORM**

OOP – Object Oriented Programming

ORM – Object Relational Mapping



Python Web Framework

Pylons

- Легкий и гибкий фреймворк;
- ORM: SQLAlchemy/SQLObject;
- встроенная поддержка AJAX;
- темплейты на выбор (Myghty/Kid/Cheetah);
- кеширование на уровне темплейтов, кода и HTTP (Etag);
- закос под Rails.

Python Web Framework

Clever Harold

- Амбициозный фреймворк?
- ORM: SQLAlchemy/SQLObject;
- темплейты на выбор (Kid/Cheetah);
- кеширование: в памяти, через memcached, dbm file.

Python Web Framework

Web.py

- Готично! Ой, аскетично;
- ORM отсутствует, прямая работа с базой;
- темплейты на Cheetah;
- кеширование на уровне темплейтов.

Python Web Framework

Turbo Gears

- Мощный модульный фреймворк;
- ORM: SQLAlchemy/SQLObject;
- встроенная поддержка AJAX;
- темплейты на выбор (все что только можно);
- кеширование на уровне SQLObject;

Python Web Framework

Django

- Продуманный и аккуратный фреймворк;
- ORM: свой и появилась поддержка SQLAlchemy;
- темплейты свои похожие на Smarty (можно использовать другие);
- кеширование отдельная инфраструктура с поддержкой memcached, database caching, filesystem caching, memory caching;
- отличная работа с HTTP-кешированием и экспайрингом + обработка Conditional GET.

Производительность на примере Django Python Framework

- История создания;
- фреймворк для журналиста;
- slashdotting;
- как всё оптимизировать...
    - Всё начинается с данных. Выбор базы данных.
    - Кеширование отрендереных страниц.
    - HTTP-кеширование.

Производительность Python Frameworks для веб-приложений

Часть четвертая

Серверные технологии для разворачивания производительных веб-приложений

mod_python

Встраивает интерпретатор с подключенными модулями Python прямо в процесс Apache

+ интерпретатор постоянно в памяти и очень быстро работает
- может уложить полностью веб-сервер в случае глюка
- работает только с Apache

Правильный сборщик мусора

FastCGI

Запускает откомпилированный байткод как отдельный процесс

+ работает быстро за счет постоянно загруженного скомпилированного байткода;
+ отдельный от веб-сервера процесс;
+ может работать на отдельном сервере и от отдельного пользователя в системе;
- медленнее чем mod_python;

Главное преимущество - гибкость

SCGI

Тот же FastCGI, только с другим протоколом общения с веб-сервером.

Python web-server

Веб-сервер написанный на Python

+ более тесная интеграция приложения и веб-сервера;
+ удобство отладки;
+ всегда под рукой;
- не такой быстрый как веб-сервер на Си.

Удобный для разработки и тестирования

Немного о веб-серверах

- Классический Apache говорят в связке с mod_python самый быстрый вариант.
- Lighttpd в связке с FastCGI дает сравнимую производительность и меньшее потребление памяти.
- Nginx в связке с FastCGI шустрее чем Lighttpd и потребляет еще меньше памяти. У него также есть ряд других преимуществ.

Производительность Python Frameworks для веб-приложений

Часть пятая

Итог: Python и Веб – спортивный интерес или бизнес-решение?

Как всё связать?

Agile development

Производительность

Framework

Бизнес

В любом бизнесе главной движущей идеей является решение проблемы **заказчика** / **покупателя**.

Эту же идею и берет технология «**Agile development**».

И для достижения этой цели используются средства, в нашем случае **Framework**, который позволит быстрее и качественнее решить задачу.

Вот на этом этапе приходится решать какую платформу выбрать.

И тут как раз стоит заглянуть вперед и посмотреть на момент внедрения и работы приложения в условиях продакшена, чтоб принять правильное решение.

Поэтому в итоге вопросы производительности, легкости внедрения и сопровождения становятся на первый план.

И на закуску пример из реальной жизни

Почему на PHP так много проектов?

Легко внедрять...

Почему на PHP всё больше плюются?

Глючность и низкая производительность...

У Python много других преимуществ, про которые можно долго рассказывать...

Советую его выучить как минимум для личного развития ;-)

Успехов в разработке!

Илья Хамушкин

E-mail: ilya@khamushkin.com
GTalk: dobrych@gmail.com
ICQ: 2211123

Меня можно почитать:
http://tophost.com.ua/blog/
http://livedev.org/