Подготовка расширений к распространению

В этой главе вы узнали, как создавать различные типы расширений Yii. Сейчас мы поговорим о том, как поделиться своими результатами с людьми, и почему это важно.

Подготовка

Давайте сначала сформируем контрольный список для хорошего расширения. Хороший продукт программирования должен следовать этим пунктам:

* Хороший стиль кодирования
* Люди должны быть в состоянии найти его
* Последовательный, легкий для чтения, и легкий для использования API
* Хорошая документация
* Расширение должно применяться к наиболее распространенным случаям использования
* Следует сохранить
* Хорошо протестированный код, в идеале с модульными тестами
* Вы должны обеспечить поддержку для него

Конечно, все это требует много работы, но она необходима для создания хорошего продукта.

Как это сделать...

1. В каждом современном продукте PHP должен соответствовать стандартам PSR4 автозагрузки и PSR1 и PSR2 стандарты стиля кодирования от http://www.php-fig.org/psr/.
2. Давайте рассмотрим наш список более подробно, начиная с API. API должен быть последовательным, легким для чтения и простой в использовании. Непротиворечивость означает, что общий стиль не должен изменяться, поэтому не должно быть других имен переменных, несогласованных имен, таких как isFlagi() и isNotFlag2 (), и так далее. Все должно подчиняться правилам, определенным для кода. Это позволяет меньше проверять документацию и позволяет сосредоточиться на кодировании.
3. Код без документации практически бесполезен. Исключением является сравнительно простой код, но даже если это всего несколько строк, это не правильно, если нет ни одного слова о том, как установить и использовать его. Что делает хорошую документацию? Цель кода и его плюсы должны быть как можно более заметны и должны быть написаны громко и ясно.
4. Код бесполезен, если разработчики не знают, куда ее девать и что должно быть в конфигурации приложения. Не ожидайте, что люди знают, как делать специфичные для фреймворка вещи. Руководство по установке должно быть подробным. Большинство разработчиков предпочитают пошаговую форму. Если для работы кода требуется схема SQL, предоставьте ее.
5. Даже если ваши методы и свойства API названы правильно, вам все равно нужно документировать их комментариями PHPDoc с указанием типов аргументов и возвращаемых типов, предоставляя краткое описание для каждого метода. Не забывайте о защищенных и закрытых методах и свойствах, так как иногда необходимо прочитать их, чтобы понять детали того, как работает код. Кроме того, рассмотрите возможность перечисления открытых методов и свойств в документации, чтобы ее можно было использовать в качестве ссылки.
6. Предоставьте примеры вариантов использования с хорошо прокомментированным кодом. Попробуйте охватить наиболее распространенные способы использования расширения.
7. В примере не пытайтесь решить несколько проблем одновременно, так как это может сбить с толку.
8. Важно сделать код гибким, чтобы он применялся ко многим случаям использования. Однако, поскольку невозможно создать код для всех возможных вариантов использования, попробуйте охватить наиболее распространенные.
9. Важно, чтобы люди чувствовали себя комфортно. Предоставление хорошей документации-это первый шаг. Второй-это доказательство того, что ваш код работает должным образом и будет работать с дальнейшими обновлениями. Лучший способ сделать это-набор модульных тестов.
10. Расширение должно поддерживаться, по крайней мере, до тех пор, пока оно не станет стабильным, и больше не будет запросов функций и сообщений об ошибках. Поэтому ждите вопросов и отчетов, и оставляйте некоторое время для дальнейшей работы над кодом. Если Вы не можете посвятить больше времени поддержке расширений, но это очень инновационно, и никто не делал этого раньше, все равно стоит поделиться. Если сообществу это нравится, кто-то обязательно предложит свою помощь.
11. Наконец, необходимо сделать расширения доступными. Создайте пакет Composer из своего расширения, вставьте его в GitHub или другое общее хранилище репозитория и опубликуйте на сайте https://packagist.org.
12. Каждое расширение должно иметь номер версии и журнал изменений. Это позволит сообществу проверить, если у них есть последняя версия и проверить, что изменилось перед обновлением. Мы рекомендуем следовать семантическим правилам управления версиями с сайта http://semver.org.
13. Даже если ваше расширение относительно простое и документация хорошая, могут возникнуть вопросы, и в первый раз единственный человек, который может ответить на них, - это вы. Обычно вопросы задают на официальных форумах, поэтому лучше создать тему, где люди смогут обсудить ваш код и предоставить ссылку на странице расширения.

Как это работает...

Если вы хотите поделиться расширением с сообществом и быть уверены, что оно будет полезным и популярным, вам нужно сделать больше, чем просто написать код. Сделать распространение расширений готовым гораздо больше работы. Он может быть даже больше, чем создание самого расширения. Итак, почему хорошо делиться расширениями с сообществом в первую очередь?

Создание кода, который вы используете в своих собственных проектах с открытым исходным кодом, имеет свои плюсы. Вы получаете людей, намного больше людей, чем вы можете получить, чтобы проверить свой проект с закрытым исходным кодом. Люди, которые используют ваше расширение, тестируют его, дают ценные отзывы и сообщают об ошибках. Если ваш код популярен, будут страстные разработчики, которые попытаются улучшить ваш код, сделать его более обширным, более стабильным и многоразовым. Кроме того, вы просто чувствует себя хорошо, потому что вы делаете доброе дело.

Мы рассмотрели самые важные вещи. Тем не менее, есть больше вещей, чтобы проверить. Попробуйте существующие расширения, прежде чем писать свои собственные. Если расширение почти подходит, попробуйте связаться с автором расширения и внести свои идеи. Просмотр существующего кода поможет вам найти полезные трюки, dos и Don'TS. Кроме того, время от времени проверяйте Вики-статьи и официальный форум; есть много полезной информации о создании расширений и разработке с использованием Yii в целом.

Смотрите так же

* Для современной информации о стандарты кодирования PHP, см. [http://www.php-fig. org/psr/](http://www.php-fig.org/psr/)
* Дополнительные сведения о семантическом управлении версиями см. в разделе <http://semver.org>