Редактирование шаблона стартер

Создание шаблонов для большинства веб-приложений на самом деле не является сложной задачей. Но если вы планируете поделиться своей работой с общественностью, то время от времени вы сталкиваетесь с небольшой проблемой, а именно, когда появляется новая версия приложения: вам нужно отслеживать, что изменилось с момента последнего релиза, чтобы иметь возможность обновить свой шаблон соответствующим образом. Конечно, не у всех есть время следить за разработкой приложения достаточно внимательно, чтобы замечать каждое небольшое изменение, которое влияет на его работу. IMHO, это может быть причиной того, что некоторые из доступных шаблонов DokuWiki немного устарели и, вероятно, не на 100% совместимы с последней версией. Однако я хотел бы показать способ воспользоваться преимуществами обработки таблиц стилей DokuWiki и свести усилия, необходимые для поддержания такого шаблона в актуальном состоянии, к минимуму (мы все ленивые люди, верно? ;-)).

Создание нового шаблона

Самый простой способ создать новый шаблон — это шаблон Starter от Anika Henke. Просто скачайте и распакуйте его. Переименуйте папку, поместите ее ниже **<dokuwiki>/lib/tpl/templates** и выберите в менеджере конфигурации DokuWiki.

Идея: воспользоваться преимуществами обработки CSS DokuWiki

DokuWiki предоставляет расширенную систему обработки CSS , которая использует файл PHP .ini с именем style.ini, который является частью шаблона. Он используется для определения того, какие файлы загружаются для определенного режима отображения. Итак, давайте взглянем на style.ini, предоставленный шаблоном Starter. Сейчас нас интересует первый раздел файла:

style.ini

```
[stylesheets]

css/basic.css             = screen
css/structure.css         = screen
css/design.css            = screen
css/content.css           = screen
css/_imgdetail.css        = screen
css/_media_popup.css      = screen
css/_media_fullscreen.css = screen
css/_fileuploader.css     = screen
css/_tabs.css             = screen
css/_links.css            = screen
css/_toc.css              = screen
css/_footnotes.css        = screen
css/_search.css           = screen
```

```
css/_recent.css           = screen
css/_diff.css             = screen
css/_edit.css             = screen
css/_modal.css            = screen
css/_forms.css            = screen
css/_admin.css            = screen
css/includes.css          = screen

css/rtl.css               = rtl
css/print.css             = print
```

Выше описано, какой файл .css загружается для определенного режима отображения. « screen» — это обычный режим отображения, когда вы просматриваете страницу в браузере, « rtl» загружается для языков с письмом справа налево, а « print», очевидно, когда вы хотите распечатать страницу вики. 2) Для получения дополнительной информации о порядке загрузки таблиц стилей обратитесь к документации .

Итак, вот ключевая часть:

Мы не будем редактировать ни один из файлов .css, скопированных из шаблона Starter!

Вместо этого мы просто добавляем наши собственные файлы **.css** для каждого режима **<dokuwiki>/lib/tpl/template/<mytemplate>/css** и изменяем их **style.ini** так, чтобы они загружались после исходных:

style.ini

```
[stylesheets]

css/_imgdetail.css        = screen
css/_media_popup.css      = screen
css/_media_fullscreen.css = screen
css/_fileuploader.css     = screen
css/_tabs.css             = screen
css/_links.css            = screen
css/_toc.css              = screen
css/_footnotes.css        = screen
css/_search.css           = screen
css/_recent.css           = screen
css/_diff.css             = screen
css/_edit.css             = screen
css/_modal.css            = screen
css/_forms.css            = screen
css/_admin.css            = screen
```

```
css/mytemplate_screen.css = screen

css/rtl.css                = rtl
css/mytemplate_rtl.css     = rtl
css/print.css              = print
css/mytemplate_print.css   = print
```

Вместо редактирования любого из файлов .css, скопированных из шаблона Starter, мы добавляем наши правила CSS в файлы, которые мы только что создали, и используем ключевую функцию CSS : так называемый каскад. 3) Выражаясь упрощенно, правила CSS просто переопределяют предыдущие правила, влияющие на тот же элемент. 4) А в некоторых случаях, когда специфичности вашего правила недостаточно, вы можете использовать !important5) или собственный класс для решения таких проблем.

Поэтому мы просто переопределяем любое правило CSS шаблона Starter , которое нам нравится менять в наших собственных mytemplate_screen.css, mytemplate_rtl.cssи mytemplate_print.css. Разработка таблицы стилей таким образом немного более продвинута, чем если бы вы просто редактировали существующие файлы .css. Но у этого есть преимущество, как вы увидите позже. Кроме того, вы могли заметить, что я удалил некоторые исходные файлы шаблона Starter, которые не начинаются с подчеркивания (basic.css, structure.css, design.css, content.cssи includes.css). Причина этого проста: только файлы, начинающиеся с подчеркивания плюс rtl.cssи , print.cssявляются важными основными стилями, остальные должны быть скорректированы. Поэтому используйте их в качестве отправной точки для ваших собственных mytemplate_screen.css.

Еще одна вещь, которую вы, возможно, захотите учесть при изменении CSS DokuWiki, заключается в том, что лучше всего получать доступ к элементам так же, как это сделано в файлах CSS шаблона Starter , и добавлять большинство идентификаторов или классов к .dokuwiki. И взгляните на цветные заполнители . Если вы случайно отредактируете main.phpсвоего шаблона, чтобы добавить или изменить что-то, вам не следует удалять , так как это нарушит всю таблицу стилей!

Чтобы получить представление о том, какие правила следует перезаписать, вам может быть полезно взглянуть на CSS- файлы моих шаблонов prsnl10 и mnml-blog :

- prsnl10_screen.css
- mnml-blog_screen.css

Как вы могли заметить, у этого подхода есть один недостаток. Размер окончательной таблицы стилей, доставленной в браузер, будет больше, чем если бы вы просто редактировали существующие файлы .css. Это должно быть около ~2 КБ до 15 КБ (несжатый), в зависимости от того, что вы изменяете. Поскольку весь CSS сжимается и обычно доставляется только один раз, а затем кэшируется, я не думаю, что это такая уж большая проблема. 6) И это не «нечисто» или что-то в этом роде, потому что данные Wiki не затрагиваются, и шаблон не вызывает никаких зависимостей для ваших данных в целом.

Поддержание актуальности

Если вы будете следовать подходу, изложенному выше, поддержание вашего шаблона в актуальном состоянии после нового релиза DokuWiki — задача, которую можно выполнить очень быстро! Единственное, что вам нужно сделать, — это запустить diffдля всех .css-файлов, которые вы скопировали из шаблона Starter, и сравнить их с исходными (или взглянуть на Commits , например, если вы не знакомы с такими инструментами, как diff). Проверка журнала изменений шаблона Starter также является хорошей идеей. Если были изменения, просто скопируйте измененные .css-файлы шаблона Starter в папку вашего шаблона, перезаписав старые, и, только если необходимо, добавьте несколько дополнительных правил в ваши пользовательские .css-файлы, чтобы настроить все в соответствии с вашими потребностями. Лучшее место для проверки того, все ли в порядке, — это страница синтаксиса , включенная в каждый релиз DokuWiki. 7) Также рекомендуется взглянуть на .php-файлы, предоставляемые шаблоном Starter, чтобы определить, нужны ли какие-либо изменения для новых функций. Официальный журнал изменений разработчиков также может помочь вам в поддержке новых функций.

Некоторые дополнительные примечания

Вот и все! По крайней мере, этот подход работает довольно хорошо для меня ;-). Конечно, это касается только проблем с CSS , но это гарантирует, что все не будет полностью испорчено. Если вы добавили новую функциональность в шаблон, вам, безусловно, придется потратить больше времени на тестирование/отладку. Но если вы создаете другой облик для DokuWiki и хотите поделиться им , это один из способов упростить все с течением времени. Поддерживать все в рабочем состоянии обычно гораздо сложнее, чем создавать что-то новое, поэтому дополнительная работа при запуске нового шаблона должна быстро окупиться. Этот подход даже достаточно мощный, чтобы сделать возможными такие вещи, как vector 8) или шаблон этого блога mnml-blog, не создавая особых проблем при выпуске новой версии DokuWiki.

Раньше лучшим способом создать собственный шаблон было скопировать шаблон DokuWiki по умолчанию. Причина проста: не было такой полезной вещи, как шаблон Starter. Поэтому все мои существующие шаблоны были основаны на шаблоне CSS по умолчанию . Но все немного изменилось. Во-первых, DokuWiki получит новый шаблон по умолчанию под названием «dokuwiki» в ближайшем будущем. Поэтому я спросил в списке рассылки, лучше ли использовать шаблон CSS «Starter» или подождать новый шаблон CSS «dokuwiki» в качестве технической базы для подхода к разработке, описанного в этом тексте. И ответ ясен: использовать шаблон Starter, поскольку новый шаблон по умолчанию также основан на Starter. Поэтому я переключил все свои шаблоны со старого шаблона CSS по умолчанию на шаблон CSS Starter в качестве технической базы. 9) Я уверен, что подобная ситуация больше не повторится. Полагаться на шаблон по умолчанию было просто менее перспективно, чем полагаться на существующий шаблон Starter, который существует только для того, чтобы упростить разработку шаблонов.

Ссылки

- Оригинальная статья
- Начальный шаблон
- Журнал изменений шаблона Starter
- Журнал изменений разработчиков DokuWiki
- Каскадные таблицы стилей
- Назначение значений свойств, каскадирование и наследование: 6.4 Каскад
- Kaskade — Рекомендации по таблицам стилей в документации :язык_de:
- prsnl10_screen.css
- mnml-blog_screen.css

Примечания

Аника Хенке№ 4 @ 2011/06/19 17:09

Спасибо за поддержку шаблона Starter. :)

Есть одна вещь, которую я бы сделал по-другому: не сохраняйте basic.css, structure.css, design.css и content.css, если вы их не используете! Только файлы, начинающиеся с подчеркивания, являются важными основными стилями, остальное должно быть скорректировано. Я понимаю, что вы хотели бы упростить обновление. Но в этом случае вам лучше удалить (или закомментировать) эти 4 css-файла из style.ini и создать свой «mytemplate_screen.css», как предлагается. (Или создать mytemplate_basic.css, mytemplate_structure.css и т. д.) Стили rtl и print могут быть исключением из этого правила, потому что они уже достаточно универсальны.

Еще одно: считается плохой практикой использовать «!important». Вместо этого вы можете добавить класс к своему <body> (или к div вокруг всего) и переопределить другие правила, например body.mytemplate .dokuwiki { font-size: 90%; }, .