УДК 004.42(07)
Христинець Н.А., Багнюк Н.В., Матейчук Н.В., Дзюбинський В.В.
Луцький національний технічний Університет

АНАЛІЗ PHP-ШАБЛОНІЗАТОРІВ ТА ВИКОРИСТАННЯ TWIG У DRUPAL 8

Христинець Н.А., Багнюк Н.В., Матейчук Н.В., Дзюбинський В.В. Аналіз PHP-шаблонізаторів та використання Twig у Drupal 8. У статті розкрито суть шаблонізаторів для відокремлення логіки від представлення, розглянуто їх основні переваги та недоліки. Здійснено порівняння характеристик поширених шаблонізаторів: Blade, Mustache, Smarty, Twig і Volt. Обгрунтовано використання Twig в якості двигуна шаблонів у Drupal 8.
Ключові слова: шаблонізатор, PHP, HTML, Twig, Drupal 8, шаблон, фреймворк, Blade, Mustache, Smarty,Volt.

Христинец Н.А., Багнюк Н.В., Матейчук Н.В., Дзюбинский В.В. Анализ PHP-шаблонизаторов и использования Twig в Drupal 8. В статье раскрыта суть шаблонизаторов для отделения логики от представления, рассмотрены их основные преимущества и недостатки. Проведено сравнение характеристик распространенных шаблонизаторов: Blade, Mustache, Smarty, Twig и Volt. Обосновано использование Twig в качестве двигателя шаблонов в Drupal 8.
Ключевые слова: шаблонизатор, PHP, HTML, Twig, Drupal 8, шаблон, фреймворк, Blade, Mustache, Smarty, Volt.

Khrystynets N.A., Bahniuk N.V., Mateychuk N.V., Dzyubinsky V.V. Analysis of PHP Templates and Using Twig in Drupal 8. The article discusses the essence of templates for separating logic from presentation, examining their main advantages and disadvantages. A comparison of the characteristics of common templates: Blade, Mustache, Smarty, Twig and Volt. The use of Twig as a template engine in Drupal 8 has been substantiated.
Keywords: template, PHP, HTML, Twig, Drupal 8, Template, Framework, Blade, Mustache, Smarty, Volt.

Постановка проблеми. Вже багато років іде дискусія на рахунок шаблонізаторів. Одні заявляють, що це хороша практика в програмуванні, а інші – що це не має виражених переваг, а тільки ускладнює розробку web-додатків. В роботі розглянуто розгорнуті характеристики популярних шаблонізаторів, переваги їх використання для різноманітних задач.

Аналіз досліджень. Антон Шевчук в своїй статті «Шаблонизаторы и PHP» [3] розкритикував існуючий популярний шаблонізатор – Smarty. Він заявив, що Smarty не вирішує головної задачі шаблонізатора – відділення логіки від представлення, адже Smarty дає можливість використовувати php код в шаблонах. Ще одним недоліком шаблонізатора він виділив складний синтаксис, який дуже часто є незрозумілим для верстальщиків і дизайнерів. Аналізуючи цю проблему, нами було виділено групу шаблонізаторів, що мають зручний синтаксиси та застосування.

Виклад основного матеріалу й обґрунтування отриманих результатів. Однією із основних характеристик шаблонізатора є можливість опису класу програм, що займаються заміною строкових послідовностей деякими даними. Слово «шаблонізатор» часто сприймається синонімом слова «подання». Заочно вважається, що якщо у проектах застосовується шаблонізатор, то у них відділено логіку представлення від бізнес-логіки. Існує думка, що шаблонізатори створені для дизайнерів/верстальників, а програмісту вони не потрібні. Насправді ж, використання шаблонізаторів полягає в тому, щоб зробити поділ логік зручніше, що надає деякі розширені функції готового веб-продукту. Можна використовувати будь-який шаблонізатор або можна взагалі обійтися без нього, але при цьому варто добре відділяти представлення від логіки. І навпаки, можна використовувати Smarty, але при цьому остаточно все заплутати. Тому, використання конкретного шаблонізатора єх позитивним рішенням для створення веб-продукту.

Поділ HTML і PHP

PHP і HTML тісно взаємодіють: PHP може генерувати HTML, а HTML може передавати інформацію PHP. Проте, складні чергування PHP і HTML ставали занадто складними для сприйняття. Зміна дизайну взагалі перетворювалася на кропітку роботу. Код дуже часто повторювався на різних сторінках і знайти його відповідний кусок було складно. Ще складніше було витягнути його звідти і вставити в новий проект.

Врешті програмісти прийшли до розуміння, що потрібно код HTML відокремити від PHP коду. Саме тоді з'явилося поняття шаблонізатора. Фактично, це є програмне забезпечення , що дозволяє використовувати html-шаблони для генерації кінцевих html-сторінок. Основна мета використання шаблонізатора – це відділення представлення даних від виконуваного коду. Часто це необхідно для забезпечення можливості паралельної роботи програміста і дизайнера-

верстальника. Використання шаблонізатора покращує читаність коду і внесення змін у зовнішній вигляд, коли над проектом працює група розробників.

PHP шаблонізатор корисний для проектів, що включають роботу розробників та дизайнерів.

Для складних програм необхідно розділити всі шари, починаючи з бізнес-логіки та закінчуючи презентацією. Це була основна причина створення шаблону дизайну MVC , який пізніше був прийнятий для веб-додатків. Проекти, створені в останні роки, просунуті на багатьох рівнях і вимагають сильної взаємодії з розробниками та дизайнерами. Обидва повинні співпрацювати, тому дуже важливим є пошук спільної мови для власної компетенції.

Переваги використання шаблонного двигуна

1. Відокремлення

Коли код додатка зростає і стає нечитабельним, розробники продумають структуру файлів проекту. Відокремлення моделі, перегляду та контролера вимагається за схемою дизайну, однак розробникам та дизайнерам все-таки потрібно працювати над їх загальним кодом. І шаблонний файл є найкращим місцем для такої співпраці.

2. Читабельність

Логіка та презентація, об'єднані в файли проекту, часто не є достатньо чіткими. Це виглядає непривабливо, і для дизайнера це дуже важко зрозуміти. Розробники розділяють код на окремі файли, оскільки обидва шари використовують різні технології, наприклад PHP та HTML. Найчастіше бекенд і фронтенд коди повинні підтримувати різні фахівці. Бекенд розробник PHP не зацікавлений в коді CSS, який використовується для стилізації веб-сторінки і, з точки зору дизайнера, те саме стосується написання запиту SQL на зворотному боці. Ось чому шаблони краще для всіх.

3. Кешування

Хороші двигуни шаблонів пропонують кешування для статичних файлів. Спеціальний синтаксис надає безліч функцій, але його потрібно проаналізувати та скласти з вихідною мовою. Шаблонні движки, такі як Smarty та Twig , мають кращу продуктивність зі скомпільованими файлами, що є ще одним кешуванням для веб-додатків.

4. Співпраця

Розробникам і дизайнерам доведеться працювати разом. Для складних проектів потрібні багато людей і досвідчені фахівці. Розробникам і дизайнерам потрібні свої ділянки в проекті, щоб працювати краще. Використання популярних інструментів допомагає їм усім розуміти один одного та легко співпрацювати.

5. Фільтри, модифікатори та багато інших

Шаблонні двигуни мають вбудовані функції, які дозволяють легко обробляти популярні сценарії. Якщо постає необхідність скоротити текст, форматувати рядок або дату, встановити значення за замовчуванням та багато інших, то рішення пропонують модифікатори та фільтри для простих завдань, які також дозволяють створювати власні плагіни та розширення. Всі ці приклади – це ще один шар абстракції, який допомагає проекту.

6. Легко вивчити

Розробники PHP дуже часто кажуть, що "PHP є мовою шаблону, тому немає сенсу вивчати іншу". З огляду на всі переваги, безумовно, добре вивчити мову двигуна шаблону, особливо, якщо мови набагато простіше для тих розробників, які раніше вивчали PHP.

Дизайнери можуть навчитися простому та інтуїтивно зрозумілому синтаксису шаблонів набагато легше порівняно з цілою складною мовою. Друк змінних, умовних інструкцій або циклів є дуже природним і простим у використанні навіть для нетехнічної людини.

7. Менший шанс зламати більше

Чим більше людей працюють з тим самим кодом, тим більша ймовірність того, що щось не так. Щоб запобігти конфліктам, розробники створили версію коду. Наступним кроком, що обмежує можливість фатальної помилки, є доступ до зони обмеженого доступу. Дизайнерам не потрібно читати та редагувати бекенд код. Місце для їх роботи обмежується файлами шаблонів.

Чому варто використовувати шаблонізатори

Усі зазначені аргументи є аналізом використаних літературних джерел, вони є результатом багаторічного досвіду роботипрофесіоналів. Можна не погодитися з ними. Але, з іншого боку, важко погодитися з використанням PHP як шаблону в кожній ситуації.

Порівняння шаблонізаторів

До шаблонізаторів застосовувалися такі критерії: вони повинні бути написані на PHP, активно підтримуватися і бути визнаними спільнотою PHP. При виборі шаблонізатора потрібно враховувати наступні фактори: синтаксис, логіка, розширюваність, документація, активність розробки, підтримка спільноти і продуктивність.

Blade

Цей шаблонізатор використовується в Laravel - PHP-фреймворку, що почав своє життя в 2011 р і став одним з популярних PHP-фреймворків. Причиною швидкості Blade є невеликий список регулярних виразів для заміни.

Mustache

Mustache доступний для практично необмеженої кількості мов, в тому числі і PHP. Також він містить мінімум логіки: заміна, цикл foreach, перевірка на null.

Smarty

Smarty з'явився в початку нульових, до цих пір розвивається і конкурує з більш молодими проектами.

Twig

Даний шаблонізатор знайшов свою популярність завдяки Фаб'єну Потенцеруб, який впровадив його в систему представлень фреймворка Symfony. Проте, Twig може бути впроваджений практично в будь-який проект, тобто незалежно від фреймворка.

Volt

Volt використовується в фреймворкуPhalcon (фреймворк, написаний на C і поширюваний як PHP-розширення). З недоліків можна відзначити лише те, що Volt можна використовувати тільки в Phalcon, тобто немає можливості використовувати в проекті на іншому фреймворку.

Отже, найбільш оптимальним варіантом шаблонізатора є Twig. Він не залежить від фреймворка, запускається дуже швидко, містить досить функціоналу, має відмінну документацію і активно розвивається.Як відомо, у Drupal 8 шаблонізатор Twig використовується замість PHPTemplate. Тепер замість звичних нам *.tpl.php файлів шаблонів використовуються *.html.twig файли.

TwigvsPHPTemplate

Розглянемо основні особливості Twig та його відмінності від PHPTemplate.

Виведення змінної:

PHPTemplate:<?phpprint$variable; ?>

Twig:{{ variable }}

Оператор If:

PHPTemplate:

```
<?php
 if($variable_1== “1”): ?>  <div>
 <?phpprint$variable_2; ?>
    </div>
<?phpendif; ?>
```

Twig:

```
{% ifvariable_1 %}
  <div>
{{ variable_2 }}
  </div>
{% endif%}
```

```
<?phpif(!empty($variable)):
$variable2= ‘value’;
 endif; ?>
```

```
{% ifvariableisnotempty%}
set variable2 = ‘value ’
{% endif%}
```

```
<?phpif(isset($variable)):
$variable2= ‘value’;
 endif; ?>
```

```
{% ifvariableisdefined %}
set variable2 = ‘value ’
{% endif%}
```

```
<?phpif($variable< 0):
$variable2= ‘value’;
 endif; ?>
```

```
{% ifvariable< 0 %}
set variable2 = ‘value’
{% endif%}
```

Надання змінній значення:

PHPTemplate:
```
<?php $variable= 'some_value'; ?>
```
Twig:
```
{% setvariable = 'some_value '%}
```

Створення масиву:

PHPTemplate:
```
<?php $my_array= array(1,2,3,4,); ?>
<?php $my_array= array('!element1'=>$var1,
'!element2'=>$var2); ?>
```
Twig:
```
{% setmy_array = [1,2,3,4,]%}
{% setmy_array = {'!element1': var1,
'!element2': val2} %}
```

Цикли:

PHPTemplate:
```
<?phpforeach($usersas$user) {
} ?>
```
Twig:
```
{% foruserinusers %} {%
endfor%}
```

ДебагінгTwig-шаблонів

Для того, щоб увімкнути можливість дебажититwig шаблони, необхідно у файлі *services.yml* у блоці налаштувань *twig.config* параметру debug надати значення *true*. Для зручності розробки можна також параметру cache встановити значення *false*. В цьому разі не потрібно буде після змін у шаблонах робити очищення кешу.

```
twig.config:
  debug: true
  cache: false
```

Встановлення власного шаблону для елемента

Розглянемо, як виглядає встановлення власного шаблону для елемента:

```
functionmy_theme_theme($existing, $type, $theme, $path) {
 returnarray(
  'block__system_menu_block'=>array(
   'template'=> 'custom-menu-template',
  ),
 );
}
```

де block__system_menu_block — елемент, для якого потрібно використати шаблон;
custom-menu-template — назва шаблону (ім'я файлу шаблону буде виглядати так: *custom-menu-template.html.twig*).

Переваги використання Twig у Drupal 8

Шаблонізатор Twig дозволяє веб-дизайнерам без будь-яких навичок в PHP модифікувати розмітку веб-сторінки. Це спрощує темізацію в Drupal 8. Це прекрасна альтернатива обробнику шаблонів PHPTemplate.

Стислість

Існують проекти, для яких реалізований програмний код на мові PHP. Але, в порівнянні з нею, Twig має більш стислий синтаксис, що робить шаблони більш читабельними. Він більш зрозумілий і простіший в компіляції. Крім того, Twig дає можливість створювати більш ефективні шаблони, завдяки синтаксису, який дуже схожий на синтаксис HTML.

Швидкість

Висока швидкість завантажння сайту дає змогу миттєво отримати необхідну інформацію. Швидкий процес розробки сайту також має вагоме значення для успіху всього проекту. Одною з цілей Twig є його швидкість. Для її досягнення Twig компілює шаблони в PHP-код. Крім того, вищезгаданий, оптимізований та зручний для читання синтаксис також допомагає веб-дизайнерам швидко виконувати свою роботу.

Безпека

Коли мова йдеться про безпеку, це має значення в будь-якому випадку і в будь-якому місці. Безпека веб-сайту настільки ж важлива, як і безпека процесу його розробки. У цьому аспекті Twig має кілька унікальних можливостей.

1. HTML escaping, або автоматичні виведення тексту у форматі HTML з міркувань безпеки можна застосувати у якості автоматичного виведення для частини коду або повністю для всього шаблону.

2. «Пісочниці, у середовищі якихробник визначає певний обмежений набір фільтрів, тегів та методів об'єктів. Користувач має доступ до нього, що дозволяє використовувати Twig як мову шаблонів для додатків, де користувачі можуть самостійно змінювати дизайн шаблону. Twig здатний оцінювати та автоматично ізолювати будь-який ненадійний код шаблону. Можна застосовувати «пісочниці» для окремих або для всіх шаблонів.

3. Бібліотека Twig повністю протестована та схвалена спільнотою. Вона є стабільною і нею можна користуватись при створенні різних проектів (додатків або сайтів).

Розширюваність

Ця корисна функція може задовольнити всі ваші потреби, чи то вони прості, чи складні. Завдяки гнучкості Twig і відкритій архітектурі можна додати нові функції, фільтри, теги, навіть синтаксис та ключові слова. Можливість розширення забезпечує розробників та веб-дизайнерів кращими інструментами для створення структури їх проекту в Drupal 8.

Наслідування

Найбільш потужною частиною Twig є наслідування шаблонів. Після впровадження Twig не доведеться копіювати файли батьківських шаблонів у власні користувацькі шаблони.

Висновки та перспективи подальшого дослідження.

Серед розглянутих шаблонізаторів в Drupal 8 найкращим варіантом є Twig. Він надає широкі можливості для розробки і до того ж він мультиплатформенний. Саме тому його вибрали розробники Drupaly своїй останній версії замість попереднього .tpl.php.

1. Addison Berry, Angela Byron, Bruno De Bondt. Using Drupal (2nd Edition). – O'Reilly. 2012. – 500p.
2. Al-Darwish, N.: PageGen: AnEffectiveSchemeforDdynamicGenerationofWebPages.InformationandSoftwareTechnology 45(10), 15 July 2003, Pages 651-662
3. Cynthia McCourt. Drupal: The Guide to Plainning and Building Websites. – Wrox. 2011. – 504 p.
4. Douglas Vernon Denny. Drupal 7 Webform Cookbook. – Packt Publishing. 2012. – 276 p.
5. FabienPotencier. TemplatingEnginesin PHP (переклад), Templatingenginesin PHP — Follow-Up (переклад)
6. Jennifer Hodgdon. A Programmer's Guide to Drupal. – O'Reilly. 2012. – 114p.
7. Ric Shreves, Brice Dunwoodie. Drupal 7 Bible. – Wiley. 2011. – 768 p.
8. Smarty 3.1.29 Released — 2015.
9. Trevor James. Migrating to Drupal 7. – Packt Publishing. 2012. – 158p.
10. Байрон А., Берри Э. и др. Drupal: создание и управление сайтом / Пер. с англ. – СПб.: Символ-Плюс, 2010. – 576 с.,ил.
11. Ромашов В.Р. CMS Drupal: система управления содержимым сайта. – СПб.: Питер, 2010. – 256 с.:ил.
12. http://anton.shevchuk.name/php/php-template-engin/
13. https://drudesk.com.ua/blog/funktsionalni-mozhlyvosti-twig