УДК 004.415+004.5

ПРОЕКТУВАННЯ СТРУКТУРИ ГЕНЕРАТОРА СІТОК ДЛЯ ПІДГОТОВКИ ВЕБ-ІНТЕРФЕЙСУ

Р. В. Олійник

Українська академія друкарства,
вул. Під Голоском, 19, Львів, 79020, Україна

У статті розглянуто задачу проектування інтерфейсу користувача за допомогою модульних сіток. Виявлено, що наявні засоби генерування мають ряд недоліків, які можуть ускладнити процес розробки багатокористувацьких інтерфейсів та запропоновано методи їх усунення. Описана структура ядра проектованого фреймворку для автоматизованого генерування систем класів. Обумовлено шляхи пришвидшення проектування не тільки каскадних таблиць стилів, а й описано шляхи збирання супутніх елементів, зокрема, шрифтового наповнення з використанням технології GULP і препроцесорів.

Ключові слова: *модульна сітка, фреймворк, система збірки, інтерфейс.*

Постановка проблеми. Розвиток мережевих інформаційних технологій, зокрема веб-орієнтація програмного забезпечення, а також ріст-вимог щодо доступності не тільки для різних пристроїв, а й для людей з обмеженими можливостями обумовили появу нових підходів до проектування користувацького інтерфейсу [1]. Частково проблему вирішують frontend фреймворки, наприклад, Bootstrap, Foundation, тощо. Проте ці засоби несуть ряд обмежень, зокрема неможливість зміни кількості колонок, які використовуватимуться у проектованому інтерфейсі та потребу у створенні дизайну під сітку[2]. Також тут часто присутня надлишковість компонентів. Тому задача проектування такого середовища, яке матиме змінну систему генерування сіток, а також обмежений набір часто використовуваних операції при створенні користувацького інтерфейсу є своєчасною та актуальною.

Аналіз останніх досліджень та публікацій. У наукових працях за темою дослідження висвітлюються різні аспекти проблематики. Зокрема закордонні видання акцентують увагу переважно на проблемі комунікації людини з технікою та веб-доступності як елементі дизайну без акцентування механізму дизайн-реалізація. Робота Йозефа Мюлле-ра-Брокманна «Модульні системи в графічному дизайні. Посібник для графіків, типографів і оформлювачів виставок» є докладним посібником з використання модульної системи в графічному дизайні та дизайнерській роботі. Тут автор переконливо доводить переваги у використанні модульного методу проектування. У праці Я. Нільсена «Веб-дизайн: книга Якоба Нільсена» визначено основні базові елементи, які беруть участь у створенні нових трендів. Ітан Маркотт у книзі «Responsive web design» розглядає можливості адаптації веб-сайту під різноманітність існую-

чих пристроїв для того, щоб задовольнити запити користувача. У роботі також наводяться приклади використання техніки «плаваючої» сітки, гнучких зображень і медіазапитів [2]. У совїй монографії «Adaptive Web Design: Crafting Rich Experiences with Progressive Enhancement» Аарон Густафсон розкриває ключові принципи і техніку створення адаптивного макету, який повинен враховувати перш за все можливості смартфона і бути простим з використанням сіток [6]. Цікавим є те, що тут А. Густафсон пропонує застосування техніки Progressive Enhancement, суть якої полягає в створенні такого сайту, який би відображав його основний зміст не тільки незалежно від засобів взаємодії з користувачем а й взагалі за відсутності інтернету. Серед Українських дослідників створення веб-інтерфейсів можна відзначити роботу Ю.О. Сирих, який у своїй праці «Сучасний веб-дизайн. Епоха Веб 3.0» аналізує популярні веб-стилістики та їх основні особливості. Також багато українських вчених досліджують системи модульнх сіток, зокрема у веб, можна відзначити Д.В Бородаева., В. В. Турчина, А. І. Рожкова, З.М. Сельменська, О.Г. Хамула, та ін.

Мета статті — вивчення сучасних тенденцій у процесі розробок веб-інтерфейсів з використанням сіток та створення підходів до компонування інтерфейсів.

Виклад основного матеріалу дослідження. Інтерфейс користувача, можна вважати ключовим елементом у будь-якій інформаційній системі [3]. Щоб вважтись якісною сторінка повинна бути лаконічною, чіткою та відповідати вимогам як дизайну так і юзабіліті і адаптивності [6]. Відтак для забезпечення обумовлених вимог до дизайну та сприйняття необхідним є використання нових концепцій, підходів та розробок. Однією з таких концепцій є використання модульних сіток у веб-дизайні [2]. З розвитком технологій зростає також і мобільність користувачів, що призводить до появи нових пристроїв відображення, зокрема, екранів з високою чіткістю, де класичний веб-інтерфейс може виглядати не презентабельно, а також такий він не здатен адаптуватись під пристрій відображення [4]. Тому сьогодні разом з класичною блочною розміткою появляються нові методи для версти, зокрема, Flexbox та Css grid. Проте, поки рано говорити, що з приходом нових стандартів потреба у модульній сітці та вендорних префіксах відпадуть. Такий стан речей зумовлено стрімким розвитком стандартів веб розробок до яких поки не можуть оновитись постачальники веб-браузерів. Тому задача створення легких сіток з мінімально необхідним набором компонентів для розробника є актуальною та своєчасною.

Проведені дослідження типових інтерфейсів дозволили виявити складність проектування функціональних елементів, адже сам інтерфейс для різної ранговості користувачів одного і того ж ресурсу може суттєво відрізнятись за функціоналом. Окрім того повинен бути передбачений функціонал для людей з особливими потребами. Тому для забезпечення логічної впорядкованості необхідно розробити бібліотеку елементів з якої власне і буде здійснено генерування візуального та функціонального автоматизованого інтерфейсу користувача. Дослідження ж наявних фреймворків показало, що найбільш доцільним при

проектуванні такого спеціалізованого засобу є використання модульної структури [5]. Тут модулем можна вважати певний компонент або набір компонентів, які за потребою підключаються розробником та вирішують ту чи іншу задачу. Також на відмінно від існуючих фреймворків, наприклад, bootstrap, повинні мати змогу як використовуватись окремо так і підключатись одразу в повному обсязі без необхідності включення залежностей, оскільки ускладнення підключення компонентів часто переобтяжує існуючий код та може призвести до конфліктів з стилями або скриптами розробника.

Оскільки проектований програмний засіб буде доволі об'ємним та повинен виконувати математичні операції хоч і не складні, зокрема, обчислення відношень ширини та висоти дочірніх блоків при зміні розмірів батьківського елементу, генерувати сітку для будь-якої кількості колонок, то доцільно скористатись препроцесорами, які являють собою надбудови для динамічного генерування каскадних таблиць стилів. Як показали проведені дослідження оптимальним варіантом є препроцесор LESS, адже він написаний на Node.js, а у випадку інтегрування системи збірки фронт-енд , наприклад GULP, розробник може динамічно не тільки генерувати код, а й оптимізувати зображення, здійснювати мінімізацію та перевірку коду на валідність стандартам.

Ґрунтуючись на вище сказаному проектований фреймворк (рис. 1.) являтиме собою ядро у яке за допомогою less директиви @include файли-модулі, а також зовнішні Less модулі або плагіни наприклад FontAwesome. Як видно з (рис. 1.) (суцільна лінія) до обов'язкових компонентів пропонується включити систему сіток та елементи Framework, котрі являють собою набір готових блоків коду для автоматичного стилізування форм та інших елементів проектованого інтерфейсу. Решта ж компонентів є необов'язковими та інтегруються за вимогою розробника з репозиторіїв постачальників цих компонентів. javascript та інші функціонально необхідна засоби розташовуються незалежно, що дозволяє розмежувати фреймворк та скриптову частину.

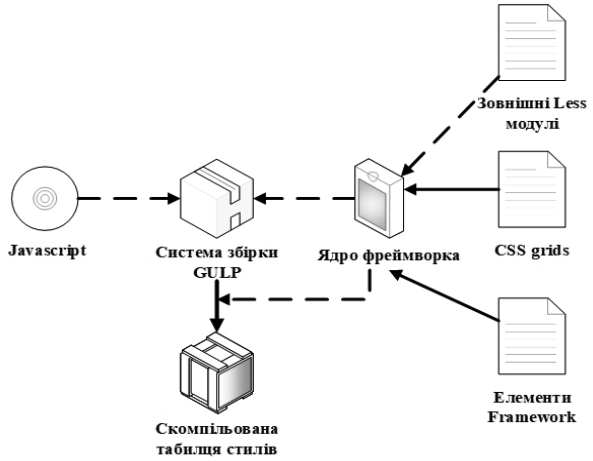

Рис. 1. Структура розроблюваного фреймворку

Для автоматизації процесу мінімізації, об'єднання стилів, картинок та ін. зручно ввести у проектований засіб систему збірки проектів GULP. Такий підхід дозволятиме контролювати не тільки сам фреймворк генеруючи з ядра необхідний CSS код, а й може долучати зовнішні CSS бібліотеки, що постачаються з сторонніми скриптами. Також важливо зауважити, що система збірки GULP здійснює генерування CSS-map яку зручно використовувати для процесу відлагодження уже сформованого фрагменту коду.

Проведені дослідження показали, що ядро проектованого фреймворка доцільно поділити на блоки обов'язкових та необов'язкових модулів. До таких модулів відноситься зокрема блок глобальних змінних. Тут описуються залежності які є сталими в процесі використання програмного засобу, наприклад, кількість колонок та між колонкові відстані, стилі для placeholder-елементів форм, системні кольори посилань, назви додаткових модулів які включені у збірку, але не активовані (рис. 2). У блоці локальних змінних користувач може здійснити індивідуальне налаштування або перевизначити глобальні змінні шляхом застосування більш специфічних селекторів або просто перевизначивши глобальну змінну. Також локальні змінні містять індивідуальні параметри для проектованого інтерфейсу у вигляді готових елементів коду – mixin, які можуть мати як свої параметри так і виконувати роль генераторів вендорних префіксів.

Як відомо анімаційні ефекти з використанням CSS являють собою функцію декількох змінних. Тому у проектованому засобі побудови інтерфейсів анімаційні ефекти пропонується використовувати окремим блоком-модулем (рис. 2.), часові властивості змінних тут зручно зберігати в блоці локальних змінних, що дозволить більш гнучко здійснювати динамічне регулювання як затримок так і часу виконання ефектів.

Як показали проведені дослідження сітку та позиціювання найкраще сприймати як пов'язані елементи, котрі мають спільні властивості, незважаючи на їх методи реалізації [1, 6]. Тому блок у якому містяться інструкції для box-моделей можуть бути перевизначені блоком сіток і навпаки. Останній і сьогодні напевно найважливіший блок параметрів є модуль адаптивностей. Тут можна здійснювати перевизначення всіх компонентів, які використовуються при створенні веб-інтерфейсу. Перевизначення відбувається за допомогою специфічності селекторів реперних точок, а також місця їх підключення. Решта ж модулів таких як шрифти, шрифтові іконки, тощо підключаються як додаткові блоки за потребою. Пізніше ядро фреймворку у вигляді одного файлу визначень можна передати або компілятору LESS або системі збірки GULP. В обох випадках буде повернуто скомпільований єдиний CSS-файл, щоправда у випадку використання системи збірки разом із згенерованими стилями буде повернуто оптимізовані параметри, які було налаштовано на початковому етапі для GULP розробником.

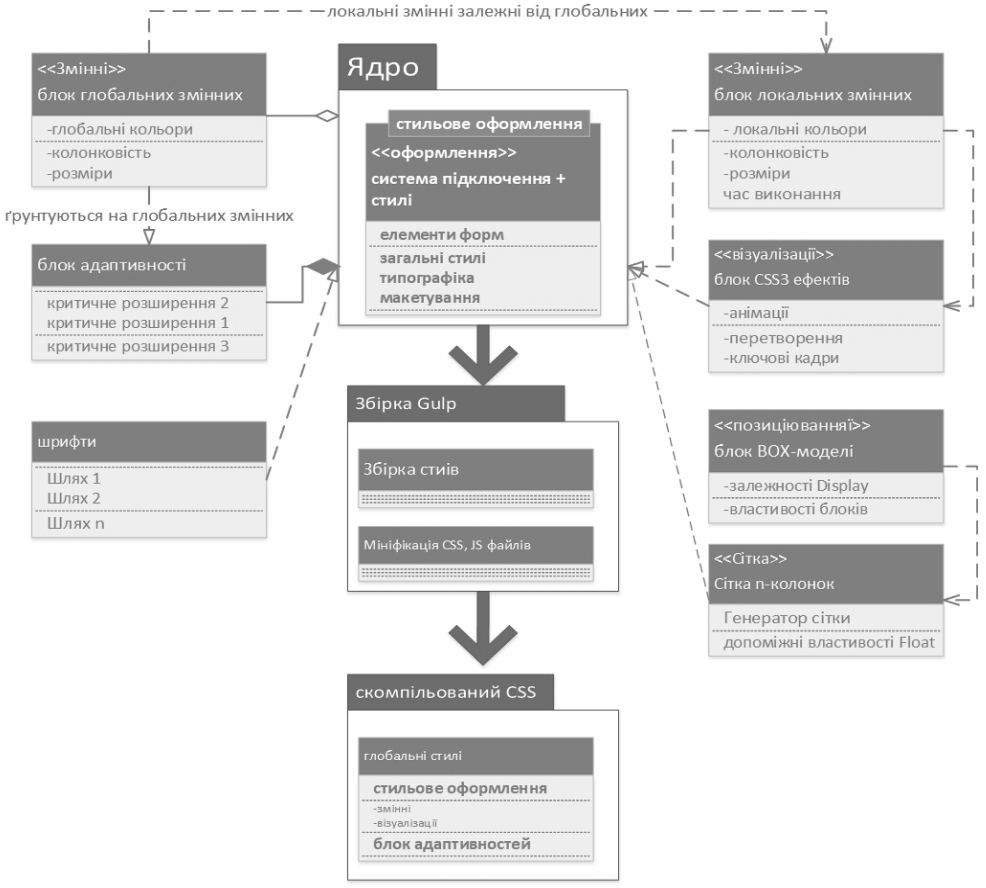

Рис. 2. Ядро фреймворка з набором модулів

Висновки. Дослідженнями підтверджено, що модульна сітка є необхідним інструментом для композиційного рішення проектування веб-інтерфейсів користувача. Головним призначенням модульної сітки є впорядкування інформації у відповідності до її призначення та релевантності, створення композиційної структури веб-сторінки та всього веб-сайту [4]. Описаний підхід у проектуванні фреймворка для створення веб-інтерфейсів дозволяє вирішити питання як створення модульної сітки так і її налаштування у відповідності до потреб проектованого середовища разом з можливістю під'єднання сторонніх скриптів без необхідності переконфігуровування всієї системи. Також використання модульної структури у процесі роботи з проектованим фреймворком дозволяє автоматизувати процес створення багатьох елементів дизайну, зокрема, веб-форм, груп меню тощо. Застосувавши систему збірки GULP можна отримати повністю автоматизований засіб створення користувацьких інтерфейсів з автономною системою класів придатною для багаторазового використання у проектах різної складності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мандел Т. Разработка пользовательского интерфейса / Т. Мандел, Пер. с англ. – М.: ДМК Пресс, 2001. – 416 с.
2. Маркотт И. Отзывчивый веб-дизайн/ Итан Маркотт. – М. : Манн, Иванов и Фербер, 2012. – 176 с. – (Серия A Book Apart).
3. Нерода Т.В. Обґрунтування користувацьких профілів у комп'ютеризованій навчальній системі ВНЗ / Т.В. Нерода, Р.В. Олійник // Науково-практична конференція «Сучасні освітні технології у професійній підготовці майбутніх фахівців»: матеріали доп. – Львів : Львівськ. Наук.-практ. центр, 2011. – С.89-90
4. Стрижак О. Є. Методика створення онтологічного інтерфейсу у середовищі WEB-порталу / О. Є. Стрижак, М. А. Попова, К. В. Ляшук // Радіоелектронні і комп'ютерні системи. - 2014. - № 2. - С. 78–84.
5. Торрес Р. Практическое руководство по проектированию и разработке пользовательского интерфейса/ Р. Торрес. – Москва: Вильямс, 2002. – 400 с
6. Gustafson A. Adaptive Web Design: Crafting Rich Experiences with Progressive Enhancement/ Aaron Gustafson. – Chattanooga : Easy Readers, 2011. – 144 с .

REFERENCES

1. Mandel T. (2001). Rozrobka korystuval′nyts′koho interfeysu / T. Mandel, Per. s anhl. - M .: DMK Press, - 416s. (in Russian)
2. Markott Y. (2012). Otzyvchyvyy veb-dyzayn / Itan Markott. - M. : Mann, Yvanov y Fer-ber, - 176 s. - (Seriya Knyha Apart). (in Russian)
3. Neroda T.V. (2011). Obhruntuvannya korystuvats′kykh profiliv u komp'yuterniy na-vchal′niy systemi VNZ / T.V. Neroda, R.V. Oliynyk // Naukovo-praktychna konferentsiya «Suchasni osvitni tekhnolohiyi u profesiynomu pidhotovtsi maybutnikh fakhivtsiv»: ma-terialy dop. - L′viv: L′vivs′k. Nauk.-prakt. tsentr, - S.89-90. (in Ukrainian)
4. Stryzhak O. YE. (2014). Metodyka stvorennya ontolohichnoho interfeysu v seredovyshchi WEB-portalu / O. YE. Stryzhak, M. A. Popova, K. V. Lyashuk // Radioelektronni ta komp'yuterni systemy. - № 2. - S. 78-84. (in Ukrainian)
5. Torres R. (2002). Praktychne kerivnytstvo po rozrobtsi ta rozrobtsi korystuval′nyts′-koho interfeysu / R. Torres. - Moskva: Vyl′yams, - 400 s. (in Russian)
6. Gustafson A. (2011). Adaptive Web Design: Crafting Rich Experiences with Progressive Enhancement/ Aaron Gustafson. – Chattanooga : Easy Readers,– 144 c. (in English)

UDC 004.415+004.5

DESIGN OF THE GRID GENERATOR STRUCTURE FOR PREPARATION OF THE WEB INTERFACES

R. V. Oliynyk

Ukrainian Academy of Printing,
19, Pid Holoskom St., Lviv, 79020, Ukraine,
enigmus@ukr.net

The article deals with the problem of user interface parsing with the help of modular grids. It has been found that the existing generating tools have some of shortcomings that can complicate the development of multi-user interfaces and suggest the ways to eliminate them. The kernel structure of the projected framework for automated generation of class systems has been described. The ways of speeding up the design of not only cascading stylesheets, but also the ways of assembling related elements, in particular, the font content using the technology of GULP and preprocessors, have been described.

Keywords: *modular grid, framework, code building systems, user intrface.*

Стаття надійшла до редакції 25.05.2017
Received 25.05.2017