x-project: next-generation API-centric HTML5 Web Components based Web Application and CMS framework

Ben Trovato[*]
Institute for Clarity in Documentation
1932 Wallamaloo Lane
Wallamaloo, New Zealand
trovato@corporation.com

G.K.M. Tobin[†]
Institute for Clarity in Documentation
P.O. Box 1212
Dublin, Ohio 43017-6221
webmaster@marysville-ohio.com

Lars Thørväld[‡]
The Thørväld Group
1 Thørväld Circle
Hekla, Iceland
larst@affiliation.org

Lawrence P. Leipuner
Brookhaven Laboratories
Brookhaven National Lab
P.O. Box 5000
lleipuner@researchlabs.org

Sean Fogarty
NASA Ames Research Center
Moffett Field
California 94035
fogartys@amesres.org

Charles Palmer
Palmer Research Laboratories
8600 Datapoint Drive
San Antonio, Texas 78229
cpalmer@prl.com

ABSTRACT

This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings. It is an *alternate* style which produces a *tighter-looking* paper and was designed in response to concerns expressed, by authors, over page-budgets. It complements the document *Author's (Alternate) Guide to Preparing ACM SIG Proceedings Using LaTeX2$_\epsilon$ and BibTeX*. This source file has been written with the intention of being compiled under LaTeX2$_\epsilon$ and BibTeX.

The developers have tried to include every imaginable sort of "bells and whistles", such as a subtitle, footnotes on title, subtitle and authors, as well as in the text, and every optional component (e.g. Acknowledgments, Additional Authors, Appendices), not to mention examples of equations, theorems, tables and figures.

To make best use of this sample document, run it through LaTeX and BibTeX, and compare this source code with the printed output produced by the dvi file. A compiled PDF version is available on the web page to help you with the 'look and feel'.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

Web Application, Web Platform, Single Page Application, Content Management System, Web Components, API

1. INTRODUCTION

A web application framework is a software framework that is designed to support the development of dynamic web applications. To speed up web application development, action based frameworks mostly rely on external configuration files Regarding to numerous open source web frameworks, it becomes so difficult for developers to select a suitable framework for web applications development. Because of that, the purpose of this paper is to help web developers to easier choose the right framework for development of their web applications. The paper will analyze a few open source Java component based frameworks. Additionally, the paper will provide basic features of the analyzed frameworks as well as represent their most important characteristics. Also, in this paper the analyzed web frameworks will be compared and summarized [1, 2, 3].

2. STATE OF THE ART

Web ÃÍ una collezione di standard allâĂŹavanguardia che ci fornisce la possibilitÃă di creare widget, ÃÍ totalmente riusabile e non interferisce con il funzionamento della pagina in caso di cambiamento dellâĂŹimplementazione interna del componente. Nello sviluppo di un widget ÃÍ importante utilizzare abilmente Javascript e HTML per la visualizzazione dinamica dei contenuti e gli standard di Web Components sono orientati verso ciÃš. LâĂŹunico, fondamentale, problema relativo allo sviluppo di Web Components ÃÍ dettato dal DOM. Il DOM interno a un widget non ÃÍ incapsulato

[*]Dr. Trovato insisted his name be first.
[†]The secretary disavows any knowledge of this author's actions.
[‡]This author is the one who did all the really hard work.

dal resto della pagina. Questa mancanza potrebbe portare a problemi di sovrascritture da parte della pagina su elementi allâĂŹinterno del widget. Open-source software (OSS) is computer software with its source code made available with a license in which the copyright holder provides the rights to study, change and distribute the software to anyone and for any purpose.[21] Open-source software is developed in a collaborative public manner. Open-source software is the most prominent example of open-source development and often compared to (technically defined) user-generated content or (legally defined) open-content movements.[22]

2.1 Web Application and CMS platforms

One universal definition of Content Management System is: "A system that lets you apply management principles to content"[4]. WordPress is a free and open-source CMS based on PHP and MySQL. Features include a plugin architecture and a template system [5]. WordPress was used by more than 23.3Keystone.js is a Node.js CMS and Web Application Platform. It is an open source framework for developing database-driven websites, applications and APIs in Node.js. It is built on Express and MongoDB [9]. The LoopBack framework is a set of Node.js modules that you can use independently or together. An application interacts with data sources through the LoopBack model API, available locally within Node.js, remotely over REST, and via native client APIs for iOS, Android, and HTML5. Using these APIs, apps can query databases, store data, upload files, send emails, create push notifications, register users, and perform other actions provided by data sources and services. Clients can call LoopBack APIs directly using Strong Remoting, a pluggable transport layer that enables you to provide backend APIs over REST, WebSockets, and other transports. The following diagram illustrates key LoopBack modules, how they are related, and their dependencies. [10]

2.2 Single Page Applications

A single-page application (SPA), is a web application or web site that fits on a single web page with the goal of providing a more fluid user experience akin to a desktop application. In an SPA, either all necessary code âĂŞ HTML, JavaScript, and CSS âĂŞ is retrieved with a single page load, or the appropriate resources are dynamically loaded and added to the page as necessary, usually in response to user actions. The page does not reload at any point in the process, nor does control transfer to another page, although modern web technologies (such as those included in the HTML5 pushState() API) can provide the perception and navigability of separate logical pages in the application. Interaction with the single page application often involves dynamic communication with the web server behind the scenes.

3. NEXT GENERATION TECHNOLOGIES AND FRAMEWORKS

3.1 WebComponents

Web Components are a collection of standards which are working their way through the W3C and landing in browsers as we speak. In a nutshell, they allow us to bundle markup and styles into custom HTML elements. What's truly amazing about these new elements is that they fully encapsulate all of their HTML and CSS. That means the styles that you write always render as you intended, and your HTML is safe from the prying eyes of external JavaScript. [11]

Custom Elements This specification describes the method for enabling the author to define and use new types of DOM elements in a document.

HTML Imports HTML Imports are a way to include and reuse HTML documents in other HTML documents.

Templates This specification describes a method for declaring inert DOM subtrees in HTML and manipulating them to instantiate document fragments with identical contents.

Shadow DOM This specification describes a method of establishing and maintaining functional boundaries between DOM trees and how these trees interact with each other within a document, thus enabling better functional encapsulation within the DOM.

3.2 Web Components Frameworks

Web Components usher in a new era of web development based on encapsulated and interoperable custom elements that extend HTML itself. There are three major frameworks built atop these new standards: Polymer, X-Tag and Bosonic. Moreover there are some Web Application Frameworks that are moving their philosophy towards Web Components such as AngularJS [101, 200, 201] and EmberJS [102]. Polymer makes it easier and faster to create elements, from a button to a complete application across desktop, mobile, and beyond. The Polymer core provides a thin layer of API on top of web components. It expresses PolymerâĂŹs opinion, provides the extra sugaring that all Polymer elements use, and is meant to help make developing web components much easier [103]. X-Tag allows to easily create elements to encapsulate common behavior or use existing custom elements to quickly get the behavior desired. X-Tag provides several powerful features that streamline element creation such as: custom events and delegation, mixins, accessors and component lifecycle functions [104]. Bosonic is a set of tools that enable the built and use reusable Web Components. By leveraging the power of DOM to build high-level elements, it allows to simplify applicationâĂŹs code and benefits from 3rd-party elements [105].

3.3 WebComponents libraries

(customelements.io, comp kitchen, paper element)

4. CONTRIBUTION

The real contribution of the work presented in this paper isâĂę la realizzazione di un API-centric CMS/SPA Application framework Web Components based.API-centrico basato su API builder strongloop loopback vantaggi immediati, separazione degli interessi,

5. X-PROJECT

X-Project isâĂę

5.1 Philosophy

âĂIJEverything is an elementâĂİ, even a page. ogni parte del sito Ãĺ incapsulata in un elemento, anche componenti funzionali come servizi http (chiamate ajax) possono essere incapsulate in un elementoâĂę

5.1.1 Components

da Web services and web components Component-based Software Engineering denotes the process of building software by (re)using pre-built software components. The main benefit of component-based software is the âĂIJtime-to-marketâĂİ [12], thus reducing the cost of developing the software. A software component is a unit of composition with contractually specified interfaces and explicit context dependencies. Additionally, a component is a software element that conforms to a component model and can be independently deployed and composed [13]. Component based software is developed interconnecting building blocks, therefore, a high degree of reusability and modularity is achieved by this type of applications [14].

5.2 Architecture

Client/Server API-centric architecture for Single Page Applications.

5.2.1 Server-side

Node.js web framework Node.js ÃÍ un framework event-driven per il motore JavaScript V8, su piattaforme UNIX like. Si tratta quindi di un framework relativo all'utilizzo server-side di Javascript.

MongDB MongoDB ÃÍ un database non relazionale, orientato ai documenti. Classificato come un database di tipo NoSQL, MongoDB si allontana dalla struttura tradizionale basata su tabelle dei database relazionali in favore di documenti in stile JSON con schema dinamico (MongoDB chiama il formato BSON), rendendo l'integrazione di dati di alcuni tipi di applicazioni piÃź facile e veloce.

NoSQL NoSQL (often interpreted as Not only SQL[1][2]) database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. Motivations for this approach include simplicity of design, horizontal scaling, and finer control over availability. The data structures used by NoSQL databases (e.g. key-value, graph, or document) differ from those used in relational databases, making some operations faster in NoSQL and others faster in relational databases. The particular suitability of a given NoSQL database depends on the problem it must solve.

Strongloop StrongLoop offers a subscription-based product known as StrongLoop Suite. StrongLoop Suite includes three components: an open source private mobile Backend-as-a-Service mBaaS named LoopBack; a second component called StrongOps, which provides operations and real-time performance monitoring in a console; and a supported package of Node.js called StrongNode, containing advanced debugging, clustering and support for private npm registries.

5.2.2 Client-side

Polymer.js

Web Components

5.3 X-Elements components

Come, polymer-project, offre una libreria di elementi, chiamata paper-elements, x-project offre una libreria di elementi, illustrati in questa sezione possiamo classificare gli elementi in strutturali, applicativi, form, di stile.

5.3.1 Strctural elements

elementi strutturali compongono la pagina

x-header , barra di navigazione

x-footer , per le informazioni a piÃÍ di pagina come autore, copyright, madeby, social buttons

x-drawer , struttura per menu laterale a scomparsa

x-page , struttura per una pagina

x-content , struttura per un contenuto generico

5.3.2 Routing elements

x-link , estensione dellâĂŹelemento anchor (`<a>`) per evitare il comportamento di default e richidere la pagina al server, ma gestire il routing localmente

5.3.3 Form elements

x-map (google maps)

x-location (google place API wrapper)

x-contact (form di elementi di input aggregati per le informazioni di un contatto)

5.3.4 Input elements

x-input, x-textarea, x-number, x-date, x-datetime elementi rispettivamente per testo breve, testo lungo, numeri, date, date e orario

5.3.5 Style elements

x-bootstrap e la libreria bootstrap adattata per stilizzare elementi dello shadow dom nota su stilizzare gli elementi. lo stile di un elemento si puÃš mettere inline

ma si possono anche stilizzare elementi dallâĂŹesterno, accedendo tramite i selettori CSS allo shadow dom, usando /deep/

link a github

5.4 Model definition

To speed up web application development, action based frameworks mostly rely on external configuration files and less on Java code [15] avviene tramite JSON

CMS definito tramite modelli. Un modello ÃÍ definito da un json, estensione dei models di loopback. Il generatore di API di loopback usa questi modelli. Hanno un campo type. NellâĂŹestensione, per non entrare in conflitto con loopback, viene aggiunto un campo `$type`. A ogni tipo di dato ÃÍ associato un campo di input corrispondente. Tali type sono:

- number: x-number

- string: x-input,
- select: x-select,
- date: x-date,
- enum: x-radio,
- number: x-number,
- email: x-email,
- textarea: x-textarea,
- url: x-url,
- boolean: x-checkbox,
- money: x-money,
- password: x-password

6. CONCLUSION

The advantages of this approach ar

6.1 Statistical comparison

In the present time open source content management system (CMS) has gained a big market. Lots of varieties are available based on functionality and platform. As told in there are different performance criteria like page load time (PLT), page size (PS), number of request, number of CSS and JS files etc. while comparing all these parameters we could come to conclusion that which CMS should be used under what conditions [19].

Generally all CMSs fulfill common task of content like create, edit, publish. But above mention CMS are providing good user support, security, more plug-ins, documentation etc than other.

As shown in [20] there are about thirty features to analyze to make a comparison of CMSs. We can design six important classes of features that summarize all the features.

Admin Management, Data Management, User Management, UI Management, Web Content Management, Multimedia Data Management.

6.2 Security

Third-party code inclusion is rampant, potentially exposing sensitive data to attackers. Protected Web components can keep private data safe from opportunistic attacks by hiding static data in the Document Object Model (DOM) and isolating sensitive interactive elements within a Web component. The Web has evolved from including static images and document links to comprising Web applications with individual components provided by numerous service providers. When a Web application incorporates third-party components using remote scripts, the userâĂŹs browser will run the third-party code within the security context of the Web application. This not only exposes the codeâĂŹs functionality to the Web application but also gives the included code full access to the Web applicationâĂŹs client-side context, including the pageâĂŹs content, local data, and origin-protected functionality. This lack of code isolation can have severe consequences if the included code doesnâĂŹt behave correctly.

Consequently, by including potentially untrusted remote scripts, a Web application developer accepts a certain risk, both for the siteâĂŹs integrity and for the safekeeping of user data [16].

Web components are the most viable starting point for creating a protection mechanism for private data and sensitive elements against opportunistic attackers. They offer the required flexibility to cope with the highly dynamic requirements of modern Web applications, as opposed to iframes, and already possess the capability to host a separate DOM tree using the shadow DOM, a property that is hard to achieve using JavaScript sandboxing technologies [17].

To leverage Web components to create protected Web components, we must be able to hide static data in the DOM tree, without it being accessible to opportunistic attackers. Second, protected Web components should be able to host interactive elements, without being vulnerable to script-based compromisesâĂŤfor example, through function-overriding or prototype-poisoning attacks [18].

7. FUTURE DEVELOPMENT

estensione libreria elementi