Journal of Technological Innovation

Etd. 2020

Employing Build Tools: Optimizing Frontend Development with Webpack, Gulp, or Grunt

Chakradhar Avinash Devarapalli
Software Engineer
E-mail: avinashd7@gmail.com

Abstract:

Frontend development with continuous progress has reached a point where the objective is to not only achieve the required output but now other attributes like optimization are also associated with modern development. To achieve these additional objectives, developers need to consider multiple domains of the project like scalability, dependencies, loading, code organization, project configuration, performance, and optimization. However, it is not possible to manually perform all the necessary actions for these factors. Therefore, suitable build tools are required to effectively automate these tasks. This will not only make different complex tasks possible but will save resources both for organizations and individuals. The highlighted tools are webpack, gulp, and grunt. This research explores these tools in terms of solving development-related challenges. The study aims to discuss the effective tools that can help to improve the development as well as maintenance process. The recommendations with their benefits help to identify the problems being faced by the individual and present the most suitable tool to avoid them in future development.

Keywords: scalability, performance, optimization, tools, frontend development, webpack, gulp, grunt.

1. Introduction

The frontend development evolved through different stages and rapid changes were adopted in the recent decade for better user experience. As the complexity of the web application increases, different challenges are given to the developers specifically in terms of project management and optimization. The build tools like webpack, gulp, and grun solved most of the problems with effectiveness in the development process and enhanced user experience [1]. Each of these tools is the aggregation of some benefits.

The webpack can handle JavaScript as well as other front-end entities like HTML, CSS, Visuals, and other media. The task is to generate static assets against the modules to avoid dependency and optimization problems. The webpack is suitable for large projects and capable of performing multitasking to efficiently generate static assets. It comes with already-integrated plugins that help in dependency management and asset optimization [2], and [3].

The gulp is helpful in the enhancement of workflow that could be difficult to handle otherwise. The repetitive tasks can be performed with this JavaScript toolkit efficiently in frontend development. Thus, it helps in the task automation. Gulp is powered with Node.JS and uses streams to automate and run tasks quickly and more efficiently. It uses plugins to work on project files. These plugins can be integrated to carry out more complex tasks [4].

The grunt also helps in task automation like gulp and helps in minification and unit testing of a project. It is equipped with numerous plugins and uses a Command Line Interface (CLI) to achieve a particular objective. It is based on Node.js and is distributed with the help of node package manager (npm). As the name suggests, grunt can focus on the completion of development tasks and perform tasks like generating minified files. It also provides automation by handling multiple repetitive tasks. It is easy to use and doesn't generate ambiguity after completing its job [5], and [6].

The current research article is mainly focused on the utilization of build tools like webpack, gulp, or grunt to optimize the process of frontend development to yield better results. The benefits associated with each of these build tools are discussed to make the developer informed decisions while choosing by using the tools. The challenges that can appear in the development industry are mentioned to analyze the essence of building tools. If not used properly, it can cost both individuals and organizations in terms of many resources. Thus, there is a need to correctly identify the requirements of the ongoing project and choose effective tools that can contribute to solving the potential problems.

The following table 1 demonstrates the comparison between these build tools,

Table 1: Comparison between tools (webpack, gulp, and grunt)

Features	**Webpack**	**Gulp**	**Grunt**
Description	Module Bundling	Task Automation	Task Automation
Source Data Format	JavaScript + Other Formats like images or CSS	JavaScript	JSON
Multitasking	Yes	Yes	No
Usability	Suitable for Large Projects	Suitable for Small or Medium Projects	Suitable for Small or Medium Projects
Plugin	built-in	Need External Plugins	Need External Plugins

2. Literature Review

Software Development in this era is a complicated process in which multiple attributes are involved. The developer has to consider the effective management of the project rather than just achieving the desired output of the feature. The third-party libraries and dependencies are used to reduce the cost and handle complex tasks in a short period. This is achieved with the permission of open-source services that are allowed to be integrated into the project to utilize the built-in features rather than writing a program from scratch. But the problems can arise with the use of these external systems like version conflicts, incompatibilities, and other vulnerabilities. Therefore, there is a need for seamless integration of useful code into the project [7].

The load time largely depends on the optimization of the front end and the assets integrated with the system. The web traffic is also dependent on the initial loading time of a website and can make or break the business based on that platform. A large number of websites online are suffering because of high waiting times in loading the components of the home page. The large assets contribute much and thus these need to be compacted in size. The large files need only to be loaded when the user desires to get a response in return for an action [8].

3. Problem Statement

Modern frontend development with complexities in all domains of the project requires updated tools that can help to meet the deadlines, increase the performance to achieve the tasks in a short time, and provide ease in managing or debugging the code. The assured tasks like dependency management, automation in identical activities to save human effort, loading, and code organization also increase the demand for efficient build tools. The problem is to identify the suitable tools that can be used to help the developers optimize their development process avoiding the possible challenges that can hinder the completion of the ongoing project.

4. Industrial Challenges

Performance and Optimization

The performance of the project is largely dependent on how the components of it are being managed and classified. Code optimization however is about how well the code is organized for later modification or debugging. Both of these tasks required serious efforts in the past without appropriate tools. The spaghetti code is more difficult to manage or modify. The efficiency can only be increased to a certain limit until development is revolutionized.

Scalability in Large Projects

If not divided or bundled properly, the project requires more effort for smaller results. The modifications in the code are not possible if the code is written in the same packages. For instance, there is a limit to the management of code built on the principle of non-modular programming. The human mind cannot go beyond a specific threshold which restricts the scalability of the projects. Apart from that, a combination of CSS and JS files in large projects can reduce the load time if no appropriate tools are used for bundling the code.

Dependency Management

It is difficult to download, integrate, and manage dependencies manually without the help of internal or external tools in large projects. Sometimes serious conflicts lead to the ultimate failure of the system. The inefficient dependencies can also be reflected in the lazy loading of the components or assets in the front end. This as a result becomes a source of bad user experience and lower return rate of users to the system. Not only that, but a lack of dependency management in the project can lead to other connected challenges as well.

Components and Assets Loading

The response of the user to a product depends on the response time of the frontend. The increased initial loading time of the page is one of the primary reasons for losing the traffic from the websites. The identification of the components and assets that are contributing to the reduction of response time is important. Only then it would be possible to solve this problem. This whole procedure requires a lot of effort from developers and therefore the build tools can help to automate this task and make possible changes directly into the project.

Organization and Configurations

It is a cumbersome task to organize code without using any tool that can help to achieve the task. If the code is not organized properly then it becomes very difficult to debug, modify, or integrate additional features in the project in the later stages. Thus, this is also linked with the scalability of the project as larger projects require continuous integration and modifications. Further, the project configuration requirement occurs in the larger project when the developers need to handle build settings.

5. Solutions with Build Tools

The build tools like webpack, gulp.js, and grunt are responsible for providing suitable solutions against the challenges discussed in the previous section. Each of these has its benefits that can help to avoid the problems faced by the developers as suggested below:

Webpack

The webpack offers the following benefits that can help the industry:

Bundling and Code Splitting
The dependencies as mentioned are difficult to manage without the use of build tools and therefore webpack helps to manage these complexities. This can be achieved by bundling the modules in terms of their association with others. Other than that, webpack is also responsible for modularization (code splitting) in which the code is divided into chunks for improved management and debugging [9]. Therefore, webpack solves the problem of dependency management, code organization, and scalability of large projects.

Optimized Loading
Webpack contributes to the initial fast loading of components and assets. This is done by rendering the necessary code only and later using the resources to execute code based on user action. Therefore, webpack helps to avoid the challenge of components and assets loading. This in turn optimizes the project and increases the performance for end users.

Gulp
The following advantages are provided by gulp.js to avoid some programming challenges

Usability
Gulp is easier to understand and read than other libraries with the same features as Grunt. Developers with basic build tools skills can also efficiently use this library. Apart from that, it is easy to debug due to its simplicity.

Task Automation

The manual efforts can be saved with the use of gulp and a significant part of development can be automated to save the resources on the development end. This is because Gulp provides interoperated plugins for the smooth build process.

Faster Builds

Being the ability to operate on streams, gulp.js can execute the files without the interposed requirements. This increases the build speed and is useful in projects where more dependencies are required to manage and contain larger files. So, it solves the issue of dependency management in addition to the better performance of the project.

Grunt

The grunt tool is accompanied by useful benefits and helps solve the challenges in the following manner. Most of the advantages of this tool are similar to what we can get with the use of Gulp with few exceptions.

Project Configuration

The configuration of the project settings becomes easy with this tool as it provides full control for easiness. The plugin's intercommunication like Gulp is also efficient in this tool and therefore it also contributes to the configuration of the overall project and saves extra efforts from the development end.

Task Automation

Similar to the previously discussed tool, it also assists in task automation and saves human efforts from manual tasks.

Diverse Handling

The responsibilities are spread in multiple dimensions as grunt can handle a diverse range of actions and provides a broad range of options to the developers for customization in the project.

6. Research Impact

This research explores the benefits of different build tools specifically Webpack, Gulp, and Grunt to help mitigate certain challenges that the software industry was facing in the past. Each of these tools is equipped with benefits and is discussed as a solution to the respective problems. One can easily identify the problem being faced in the project with the help of this and can choose a tool against the problem. With the effective use of these tools, the developers can save resources both on the personal and organizational levels.

7. Conclusion

Considering the overall discussion, each of the mentioned tools is equipped with advantages that can revolutionize the way of development and are helpful specifically in large projects that are difficult to scale as well as efficient in terms of loading assets and returning expected results. It has become inevitable to use build tools like these to increase productivity and achieve output faster by meeting all the requirements. Almost all organizations and individuals working on frontend projects are using build tools according to their interests and if not this can help identify the reasons to start using these build tools.

References

[1] E. Elrom, "Platform Deployment," in *Pro MEAN Stack Development*, New York, USA, Apress, Berkeley, CA, Dec. 2016, pp. 249-279.

[2] M. Clow, "Introducing Webpack," in *Angular 5 Projects*, Sandy Springs, Georgia, USA, Apress, Berkeley, CA, Feb. 2018, pp. 133-137.

[3] S. Chen, U. R. Thaduri and V. K. R. Ballamudi, "Front-End Development in React: An Overview," *Engineering International*, vol. 7, no. 2, pp. 117-126, 2019.

[4] T. Maynard, Getting Started with Gulp, Birmingham-Mumbai: Packt Publishing Ltd. , 2017.

[5] D. Reynolds, Learning Grunt, Birmingham, UK: Packt Publishing Ltd., Mar. 2016.

[6] J. Cryer, "Using Grunt with JavaScript," in *Pro Grunt.JS*, Caerphilly, United Kingdom, Apress, Berkeley, CA, 2015, pp. 59-83.

[7] E. E. Eghan, "Dependency Management 2.0 – A Semantic Web Enabled Approach," Nov. 2019.

[8] M. Selakovic and M. Pradel, "Performance issues and optimizations in JavaScript: an empirical study," in *ICSE '16: Proceedings of the 38th International Conference on Software Engineering*, May. 2016.

[9] V. Subramanian, "Modularization and Webpack," in *Pro MERN Stack*, Bangalore, Karnataka, India, Apress, Berkeley, CA, Mar. 2017, pp. 115-150.