International Journal For Technological Research In Engineering
Volume 9, Issue 10, June-2022 *ISSN (Online): 2347 - 4718*

MODERN JAVASCRIPT

[1]Aman Rawat, [2]Prabhav, [3]Utkarsh Tiwari, [4]Prof. Gurpreet Kaur
[1,2,3]Students, [4]Assistant Professor
Department of Computer Science Engineering
Bhagwan Mahaveer College of Engineering and Management, Sonipat, India

Abstract: - JavaScript is the main script language for Web browsers, and it is essential to modern Web applications. Programmers have begun to use it to write complicated applications, but there is still little tool support available during development. We present a static program analysis infrastructure which can infer detailed and acoustic information for JavaScript programs using abstract interpretation. The analysis is designed to support the entire language as defined in the ECMAScript standard, including its particular object model and all embedded features. The results of the analysis can be used to detect current programming errors – or rather, to prove their absence, and to produce type information for program understanding.

Preliminary experiments conducted on real JavaScript code indicate that the approach is promising in terms of precision of analysis on small and medium-sized programs, which make up the majority of JavaScript applications. With additional enhancement potential, we offer analytics as a basis to build tools that can help JavaScript programmers.

What is JavaScript?

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was initially known as LiveScript, but Netscape changed its name to JavaScript, probably due to the excitation generated by Java. JavaScript first appeared in Netscape 2.0 in 1995 and was named LiveScript. The general core of the language has been incorporated into Netscape, Internet Explorer and other web browsers.

JavaScript was developed by Brendan Eich in 1995. It was developed for Netscape 2 and became ECMA-262 by 1997. After Netscape shared JavaScript with ECMA, the Mozilla Foundation continued to develop JavaScript for the Firefox browser.

In 1995, Netscape announced JavaScript as a "user-friendly object scripting language designed to create live online applications that link objects and resources on clients and servers". Since then, it's become the de facto standard for client-side scripts in web browsers, but many other applications also include a JavaScript engine. This prevalence has led developers to write large programs in a language that has been designed for scripting, but not for programming in the large. Therefore, the support of the tool is much needed to help debug and maintain such programs.

Developing programming tools that go beyond the simple verification of syntax properties requires a kind of program analysis. In particular, type analysis is crucial for detecting representational errors, e.g. mistake numbers for strings or boolean for functions at the beginning of the development process. Type analysis is a precious tool for a programmer because it completely excludes this class of programming mistakes.

ECMA-262 has defined a standard version of the base JavaScript language.

JavaScript is a simple and interpreted programming language.

Built to create network-centric apps.

Complementary and built-in Java.

Complements and integrated into HTML.

Open and multiplatform.

Client-Side JavaScript

JavaScript on the client side is the most common language form. The script must be included or referenced by an HTML document before the code can be interpreted by the browser. This means that a web page does not need to be static HTML, but may include programs that interact with the user, control the browser, and dynamically create HTML content.

The JavaScript client-side mechanism offers many benefits over conventional CGI server-side scripts. For example, you can use JavaScript to verify whether the user has entered a valid e-mail address into a form field. The JavaScript code is run when the user submits the form, and only if all entries are valid, they will be submitted to the web server. JavaScript may be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user explicitly or implicitly initiates.

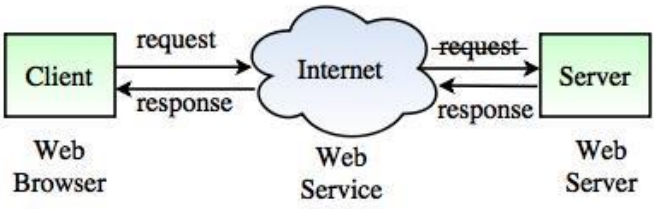


Fig. Client-Side Scripting

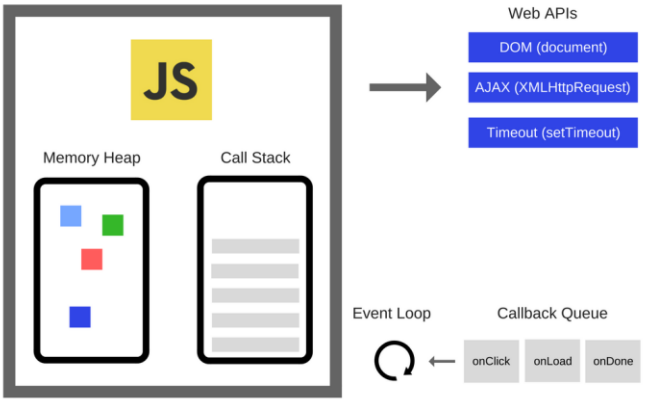
Where is JavaScript Today?

ECMAScript Edition 5 will be the first update published in over four years. JavaScript 2.0 complies with version 5 of the ECMAScript standard, and the difference between the two is extremely minimal.

JavaScript 2.0 specs are available at http://www.ecmascript.org/

Nowadays, Netscape JavaScript and Microsoft JScript are compliant with the ECMAScript standard, although both languages still support features that are not part of the standard.

How JavaScript Works?



Applying type analysis to JavaScript is a subtle business because, like most other scripting languages, JavaScript has a weak, dynamic typing discipline which resolves many representation mismatches by silent type conversions. As JavaScript supports objects, first-class functions, and exceptions, tracking the flow of data and control is nontrivial. Moreover, JavaScript's peculiarities present a number of challenges that set it apart from most other programming languages: – JavaScript is an object-based language that uses prototype objects to model inheritance. As virtually all predefined operations are accessed via prototype objects, it is imperative that the analysis models these objects precisely. – Objects are mappings from strings (property names) to values. In general, properties can be added and removed during execution and property names may be dynamically computed. – Undefined results, such as accessing a non-existing property of an object, are represented by a particular value undefined, but there is a subtle distinction between an object that lacks a property and an object that has the property set to undefined. – Values are freely converted from one type to another type with few exceptions. In fact, there are only a few cases where no automatic conversion applies: the values null and undefined cannot be converted to objects and only function values can be invoked as functions. Some of the automatic conversions are non-intuitive and programmers should be aware of them. – The language distinguishes primitive values and wrapped primitive values, which behave subtly different in certain circumstances. – Variables can be created by simple assignments without explicit declarations, but an attempt to read an absent variable results in a runtime error. JavaScript's with statement breaks ordinary lexical scoping rules, so even resolving variable names is a nontrivial task. – Object properties can have attributes, like ReadOnly. These attributes cannot be changed by programs but they must be taken into account by the analysis to maintain soundness and precision. – Functions can be created and called with variable numbers of parameters. – Function objects serve as first-class functions, methods, and constructors with subtly different behavior. An analysis must keep these uses apart and detect initialization patterns. – With the eval function, a dynamically constructed string can be interpreted as a program fragment and executed in the current scope. – The language includes features that prescribe certain structures (the global object, activation objects, argument objects) in the implementation of the runtime system. These structures must be modeled in an analysis to obtain sufficient precision.

Applications of JavaScript

JavaScript is the widely used programming language, all over the world. It has the largest open-source package repository in the world (npm). Every type of software uses JavaScript, including the server code (Node.js), productivity apps, 3D games, robots, IOT devices. JavaScript follows the principle of: write once, run anywhere.

JavaScript is mainly used in the following:-

- Web Development -

JavaScript lets you add behaviour to the web page where the page responds to actions without loading a new page to request processing. It enables the website to interact with visitors and execute complex actions

- Web Servers-

With the advent of Node.js a few years ago, JavaScript made its way from the browser into the server. Since then Node is adopted by major companies such as Wal-Mart, as a key part of back end infrastructure

- Games-

While the browser hasn't been the traditional games platform in the past, recently it has become robust for games. Additionally, with the addition of HTML 5 canvas, the level of complexity that is possible in the browser-based games has increased exponentially.

- Mobile Applications-

One of the most powerful things you can do with JavaScript is to build an application for non-web contexts. Mobile devices are now the most popular way to access the internet. What this means is all of the websites should be responsive.

- IOT-

Several commercially available Quadcopters, some outfitted with a simple OS, make it possible to install Node.js. This means that you can program a flying robot with JavaScript.

Web-Frames of JavaScript

JavaScript Framework is an application skeleton, a complete structure that provides the programmer with the basic tools of creating a website or a web application. They consist of a huge collection of various JavaScript libraries that supplies the programmers with pre-written code and thus allows even those who don't have a lot of programming knowledge.

Angular Js

AngularJS is a very powerful JavaScript Framework. It is used in Single Page Application (SPA) projects. It extends HTML DOM with additional attributes and makes it more responsive to user actions. AngularJS is open source, completely free, and used by thousands of developers around the world.

- AngularJS is a efficient framework that can create Rich Internet Applications (RIA).

- AngularJS provides developers an options to write client side applications using JavaScript in a clean Model View Controller (MVC) way.

- Applications written in AngularJS are cross-browser compliant. AngularJS automatically handles JavaScript code suitable for each browser.

- AngularJS is open source, completely free, and used by thousands of developers around the world.

Though AngularJS comes with a lot of merits, here are some points of concern −

- Not Secure − Being JavaScript only framework, application written in AngularJS are not safe. Server side authentication and authorization is must to keep an application secure.

- Not degradable − If the user of your application disables JavaScript, then nothing would be visible, except the basic page.

React Js

ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front end library which is responsible only for the view layer of the application. It was initially developed and maintained by Facebook and later used in its products like WhatsApp & Instagram.

The main objective of ReactJS is to develop User Interfaces (UI) that improves the speed of the apps. It uses virtual DOM (JavaScript object), which improves the performance of the app. The JavaScript virtual DOM is faster than the regular DOM. We can use ReactJS on the client and server-side as well as with other frameworks. It uses component and data patterns that improve readability and helps to maintain larger apps.

A ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code. The components are the heart of all React applications. These Components can be nested with other components to allow complex applications to be built of simple building blocks. ReactJS uses virtual DOM based mechanism to fill data in HTML DOM. The virtual DOM works fast as it only changes individual DOM elements instead of reloading complete DOM every time.

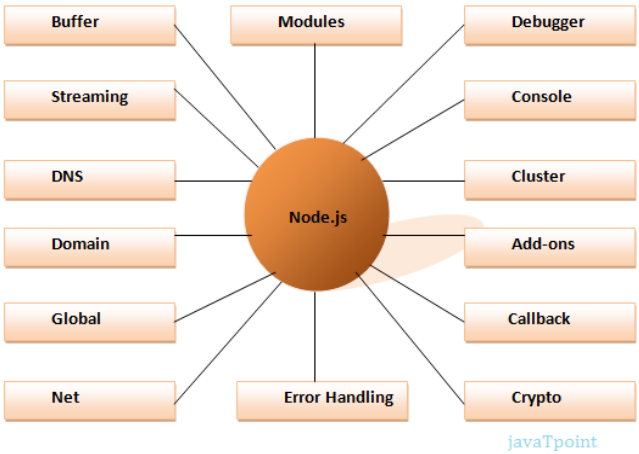
In React application, there are several files and folders in the root directory. Some of them are as follows:

- node_modules: It contains the React library and any other third party libraries needed.
- public: It holds the public assets of the application. It contains the index.html where React will mount the application by default on the element.
- src: It contains the App.css, App.js, App.test.js, index.css, index.js, and serviceWorker.js files. Here, the App.js file always responsible for displaying the output screen in React.
- package-lock.json: It is generated automatically for any operations where npm package modifies either the node_modules tree or package.json. It cannot be published. It will be ignored if it finds any other place rather than the top-level package.
- package.json: It holds various metadata required for the project. It gives information to npm, which allows to identify the project as well as handle the project?s dependencies.
- README.md: It provides the documentation to read about React topics.

Node Js

Node.js is a cross-platform runtime environment and library for running JavaScript applications outside the browser. It is used for creating server-side and networking web applications. It is open source and free to use Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js = Runtime Environment + JavaScript Library



Features of NodeJs

- Extremely fast: Node.js is built on Google Chrome's V8 JavaScript Engine, so its library is very fast in code execution.
- I/O is Asynchronous and Event Driven: All APIs of Node.js library are asynchronous i.e. non-blocking. So a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call. It is also a reason that it is very fast.
- Single threaded: Node.js follows a single threaded model with event looping.
- Highly Scalable: Node.js is highly scalable because event mechanism helps the server to respond in a non-blocking way.
- No buffering: Node.js cuts down the overall processing time while uploading audio and video files. Node.js applications never buffer any data. These applications simply output the data in chunks.
- Open source: Node.js has an open source community which has produced many excellent modules to add additional capabilities to Node.js applications.

Express Js
Express is a fast, assertive, essential and moderate web framework of Node.js. You can assume express as a layer built on the top of the Node.js that helps manage a server and routes. It provides a robust set of features to develop web and mobile applications.

Core features of Express framework:

- It can be used to design single-page, multi-page and hybrid web applications.
- It allows to setup middlewares to respond to HTTP Requests.
- It defines a routing table which is used to perform different actions based on HTTP method and URL.
- It allows to dynamically render HTML Pages based on passing arguments to templates.

Why use Express

- Ultra fast I/O
- Asynchronous and single threaded
- MVC like structure
- Robust API makes routing easy

React Native
React Native JavaScript framework inherits all the main features of the ReactJS library, like code reusability. You already know all the features of ReactJS from our prior tutorial on JavaScript Libraries. Many popular applications like Facebook, Instagram, Pinterest, Skype, etc. use React Native for development.

- It is an open-source project, thus increasing its community support daily.
- It provides various ready-to-develop components, thus it has a shorter development time.
- Cross-platform applications are possible to some extent as it supports iOS and Android.
- It creates fast, stable and reliable applications very easily.
- It saves the developers time as well as money with only a minimum code.

The pitfalls of the framework are:

- There is only a small collection for the readymade components, limiting the developers to only simple applications.
- Its testing is still in the beta phase since the navigation is not so smooth.

Future and Scope

Although JavaScript is an old technology, just like our computer is. It was established a relatively long time ago and suffered some repercussions some time ago. But with the arrival of ECMAScript and Node.js, things have changed considerably, and JavaScript is now the third most used language in the world with java and python above. With the development of CSS and with the growing importance of Vue.js, Node.js and other frameworks and API driven web applications, it's absolutely stupid to think that JavaScript has no future.
Other languages arrive via Web Assembly, but this is far from mature in the future. Currently, you can write C code that compiles to web assembly and talks to JavaScript to do intense process tasks. There are also a number of languages that transfer to JavaScript, to name a few: Scala.js, ClojureScript, TypeScript.
That said JavaScript is the first and most fashionable languages right now. And since the body that controls the ECMAscript standard has moved to a yearly publishing cycle, JavaScript has many wonderful updates and modern language features. This has become a beautiful language day after day. The language will always have a historical craft because web standards strive for 100% backward compatibility with every piece of code ever written for the web since it began.HTML, CSS and JavaScript don't have the luxury of tossing out the old for this reason.
Any frontEnd code is inherently insecure. Sure the app ecosystem of IOS, Android, etc is more locked down and reverse engineering computer code into something human-readable is a huge pain in the ass but it's still doable. The point is, you can never trust the client. This is why OAuth and other authentications were made. Principally the front-End manages state while the back-End gives and takes your data and authorizes data access. Assuming your Front-End is compromised and API endpoints are exposed, this doesn't mean anything because without the right JWT or whatever you can't get or give data from the API anyway. And if the frontEnd is compromised, a big deal if you can fake manage state. The Front-End should never be your source of truth.

REFERENCES

1. https://www.codecademy.com/articles/language/javascript
2. Norris Boyd et al. Rhino: JavaScript for Java. http://www.mozilla.org/rhino/
3. https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics
4. https://reactjs.org/
5. Adobe. JSEclipse. http://labs.adobe.com/technologies/jseclipse/
6. Robert Cartwright and Mike Fagan. Soft typing. In Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI '91, June 1991.
7. https://developer.ibm.com/languages/javascript/articles/

