Bracket is a new, easy, visual, live web programing language that interacts between html, js, and nodejs, and renders on both server and browser sides, where a developer writes only bracket code to build any desired web application. Bracket is live on internet. To start building an application using bracket, you need to follow these steps:

1. Signup on bracketjs.com

2. Create a project

3. Start creating pages and views

When a developer arrives the developer editor page in bracket, he is ready now to start coding. By default a page (named "main") and a "main" view are created. The main page is the landing page of the developer web application with path "/", and the "main" view is a view id that is included in the main page.

The structure of bracket code is JSON, where every file has specific fields and values. The JSON files will be converted to html and js code and will interact according to bracket logic. Every string value is structured as follows: "view\_type?params?conditions". Which means a view is conditional and has params related will be assigned or executed. So a field value has the value itself, params, and conditions that are separated by question marks. ex.

{ type: "Text?style.color=red;style.fontSize=15px;text=Hello World!" }.

Here there are no conditions to be considered, the view type is Text, and the text has styles and its innerHTML is "Hello World!". Params are separated by semicolons as shown in the example above. There are two store types. The first type is global and is symbolized as ")(" which is accessible all over the app, and the other is related to each view and is symbolized as "()". To access any view over the app, there are several ways, most popular is by id like follows "():id". Adding no id means accessing the local view.

Every page has a unique path, unique title, controls, any params, and could include multiple views. ex. the home page has a navbar, sidebar, body, footer... these are considered as views that could be created individually or included all in a one view.

A view is an object component with many attributes that has a parent view and may have children views, where a child could be also a parent of other children. A bracket view has 3 main fields: type, children, and controls. Regarding the type field, there are several view types like: View, Text, Input, Map.... and its value includes the view type, parameters related to the view (like styles, events, attributes, variables, html element...), and conditions as mentioned before. Any view could be included in any and many pages.

Development environment is live on internet. When editing your code and after saving, the changes could be watched immediately on the app, ex: on my\_project\_name.bracketjs.com/page\_name.

Data of each project (such as views, pages, collections, products, and any other data) is stored on firestore, and project is hosted on firebase (for now and soon on AWS EC2) as well. A developer using bracket could create many collections of products, hotels, reviews, or any collection type, and could store files and images.

Bracket language compilation is very high in speed, efficient, and very stable. It is still under development, even though a developer starting from now could build lots of applications using Bracket.

Views

Let us look deeper in views. Consider the following example. We want to create an app that retrieves data from all users (first name, last name, email, phone number, and hobbies), then submitted and stored and lists all users data in a below the form.

The bracket code of this app is as follows: