Anatomy of a component

Pre-requisites

We consider that you are familiar with Vue development.

You can see the excellent Vue documentation here.

> **TIP**
>
> WeWeb relies on **version 3** of Vue.

There are two files you will edit:

- `src/wwElement.vue` or `src/wwSection.vue` which is the Vue component displayed in the Editor and the published web-app.
- `ww-config.js` which describes the different properties of your component so that WeWeb knows how to interact with your component

Vue component

See the Development process to load a base component in dev mod.

For now your Vue component is simple. It received the `content` props from the Editor, and uses it to add some style on text:

```js
export default {
  // [...]
  props: {
    content: { type: "Object", required: true },
  },
```

```
    computed: {
      textStyle() {
        return {
          color: this.content.textColor,
        };
      },
    },
    // [...]
  };
```

```html
<h1 :style="textStyle">My Title</h1>
```

The `content` props is where your component's editable data lives. Each time this content changes via the Editor, the component will receive a new updated content object and, thanks to Vue's reactivity, your component will refresh its template.

> **TIP**
>
> WeWeb handles a lot of things for you like translation, responsive or binding so that you will always receive a simple `content` object without having to think about it.

`content` can have any shape you want, but we encourage you the keep it as flat as possible, because it interacts better with a lof of WeWeb features. You can put any style and html you want to your component. Just keep in mind that your root element will be editable via WeWeb's default editor.

It is also required that your template has a root element, and is not a fragment.

> **WARNING**
>
> Avoid using link or defining styles like padding which are handled by the Editor.
>
> See a complete list here

---

ww-config.js

This file describes the metadata of your component. All the desired properties of your `content` need to be listed here. You also have a lot of options to customize the Editor's

side panel, the different menus and interactions inside WeWeb's Editor:

```js
export default {
  editor: {
    label: {
      en: "My Element",
    },
  },
  properties: {
    textColor: {
      label: {
        en: "Text color",
      },
      type: "Color",
      defaultValue: "#F23636",
    },
  },
};
```

- `editor` describes the interaction of your component inside the WeWeb Editor
  - `label` is a translated text used by the Editor for the menu, side panel titles, or navigator
  - A lot of other options are available, there are listed here
- `properties` is an object which describes all your content properties
  - Each content property will describe the corresponding property inside content (if your property name is `textColor`, it will be available in `this.content.textColor`)
  - `type` lets WeWeb's Editor display the correct user interface to edit your property. A list of all available types and their options are available here

TIP

Defining all your properties inside `ww-config.js` is mandatory. You can decide to handle yourself the edition update by using a custom editor interface.

For that, you can use the hidden option to hide it from the edition panel.

Using the configuration file is still the quickest way to make your content editable by users and offer a uniform edition experience for users across all the components.

Your turn!

If you want to add more properties take a look at our cookbook.

- Add a property
- Add a dropzone
- Add a responsive property
- Add a bindable property
- Add a translated text property

Previous page
Development process

Next page
Using Claude Code