How to implement a widget

Implementing a widget means to develop an user interface (UI) element that is reusable by other microservice developers. This guide limits this approach to reusable UI elements for websites (HTML, CSS and JS) and it is using Vue.js to implement it, since it supports our core concept of decomposed UI elements as its own core concept. Furthermore, Vue.js advertises itself with versatility, thus it can be used to implement a library of UI elements, what is exactly what we like to do. Additionally, guides using alternative technologies (e.g. React) should be written.

At first, you have to setup vue.js. Please use the Vue CLI and follow the official instructions to create a new project. For this guide instantiate the project

```
vue create hello-cosmos
```

> Since many use cases imply the handling of data, you should use vuex stores.

Now, you are able to generate a set of vue components that will be the code base for widgets. A project repository could export multiple widgets. You can simply implement them as different components and test them by importing it in the App component. The set of components representing a widget will be exported as library.

> A widget should be implemented as a component that maybe uses other components.

> Avoid to use globally imported components, since they will possibly not be available in the library.

Exporting the components as library requires an additional file: `./src/lib.js`

```
import WidgetA from './components/widget_a'
import WidgetB from './components/widget_b'

WidgetA.install = function(Vue, options) {
   Vue.component('widget-a', WidgetA)
}

WidgetB.install = function(Vue, options) {
   Vue.component('widget-b', WidgetB)
}

// Install by default if using the script tag
if (typeof window !== 'undefined' && window.Vue) {
    window.Vue.use(WidgetA)
    window.Vue.use(WidgetB)
}

export default WidgetA
const version = '__VERSION__'
// Export all components too
export {
    WidgetA
    WidgetB
}
```

The `vue-cli-service` will build the library, if we add the following line to the `package.json`:

```
"scripts": {
  ...
  "lib": "vue-cli-service build --target lib --name vca-widget-base src/lib.js"
  ...
}
```

Now, you can build the library by executing `npm run lib` at the console.

> Avoiding mistakes, you can also add an additional line:

```
"scripts": {
  ...
  "prepublish": "npm run lib",
  ...
}
```

Documentation

Sharing code requires to write an appropriate documentation. The widgets are documented at three different layers. First of all, a `readme.md` containing a detailed overview has to be composed. At the second layer, the `readme.md` has to be referenced in this documentation (section widgets) as a new page. As the last step, all components (widgets) have to be described in the widgets table.