Custom Components Upp

The goal of this project is to learn how to customize SAP Upscale with extensions. Services have been prepared to handle event data to user-defined properties, SAP help tips have been included, a distinction has been made for different platforms (Android, iOS...) and several example components have been added to customize SAP Upscale.

Before implement this project is recommended see the next video: 🎞

Implementation of a new component

1. Extend AbstractComponent class

```
[...]
import { AbstractComponent } from '../abstract.component';
[...]
export class SimilarProductPreferencesComponent
  extends AbstractComponent
  implements OnInit
{
  [...]
```

2. Create a constructor: It's important to call to super method and pass the Upscale event handler service, the zone and the initial height the container will have.

```
  constructor(
    public zone: NgZone,
    public uppEventHandlerService: UppEventHandlerService,
    private _localStorage: LocalStorageService // Optional
  ) {
    super(zone, uppEventHandlerService, 0);
  }
```

> **Note** Declare LocalStorageService is optional, use it when you want keep or retrieve som information from him

3. Create the following attribute for the class and its corresponding view:

```
  @ViewChild('container') container: ElementRef | undefined;
```

```
<div #container class="grid-container">
  [...]
</div>
```

4. Finally, it calls to `perform` method after view init. This method after view initialisation. This function applies events to send the height of the inner elements or when the `container` is resized.

```
  ngAfterViewInit() {
    this.perform(this.container);
  }
```

Manage Events

The following example contains how to manage an event in TypeScript

```
initData() {
    window.addEventListener('message', (event:
UppEvent<ComponentContextData>) => {
        // Check that the type is as expected
        if (event.type == EVENT_COMPONENT_CONTEXT) {
          // Do something
        }
    }, false);
}
```

Recommendations:

- Build a model class to map the inbound data from event
- Parametrized `UppEvent<T>`
- Use declared contants or create new ones in `src/app/constants`

Examples of components

- `similar-product-preferences`: This component receives an event called `component_context` from Upscale and then performs a request to Upscale to get the first 5 products of the same selling tree (it's a model in Upscale).

- `navigation-menu` component: you can use to navigate throught the differents components within this application.

> **Note**: Use this component to test application indepently of Upscale, you don't configure it as extension in the Workbench

- `simple-form` component: simple example form

- `external-data` component: it consumes a external URL and show the information in card style

Finally, you must configure extension Upscale to consume one of the next URL's:

- `/example-form`
- `/example-external-data`

- `/example-product-preference`

References SAP Upscale

- SAP Upscale
- Custom components

Installation - Arch Linux

- Install Node.JS

```
pacman -S nodejs
```

> Tested with the version `16.1.0`

- Install NPM

```
pacman -S npm
```

> Tested with the version `7.13.0`

- Add path variables to `~/.profile` to allow user-wide installations

```
PATH="$HOME/.local/bin:$PATH"
export npm_config_prefix="$HOME/.local"
```

- Restart terminal
- Install Angular CLI 12.0.1 for global scope

```
npm install -g @angular/cli
```

- Move to the root repository and install the dependencies `npm install`
- Use the `ng` commands for Angular CLI

Guide

Development server

Run `ng serve` for a dev server. Navigate to `http://localhost:4200/`. The app will automatically reload if you change any of the source files.

Code scaffolding

Run `ng generate component component-name` to generate a new component. You can also use `ng generate directive|pipe|service|class|guard|interface|enum|module`.

Build

Run `ng build` to build the project. The build artifacts will be stored in the `dist/` directory. Use the `--prod` flag for a production build.

Running unit tests

Run `ng test` to execute the unit tests via Karma.

Running end-to-end tests

Run `ng e2e` to execute the end-to-end tests via a platform of your choice.

Further help

To get more help on the Angular CLI use `ng help` or go check out the Angular CLI Overview and Command Reference page.