CLP App

Folder Structure

The application uses yarn with workspaces to organize all the single apps.

```
.
|-- apps - all single react apps
|-- assets - files like PDFs, vendor files etc.
|-- bin - node scripts for tooling
|-- build - production ready builds (auto generated)
|-- libs - shared code consumed by apps
|-- node_modules - all required dependencies (auto generated)
|-- units - static files for units and descriptions
|-- README.md - documentation (current file)
|-- README.pdf - documentation as PDF file
|-- package.json - workspace configuration
`-- yarn.lock - dependency lockins
```

Installation

- To run the application you need to download Node JS
- Open your terminal and change into the `clp` directory
- Execute this command: `npx yarn`

Commands

There are some predefined commands to make app management easier:

Running an app locally

To run an app locally and preview changes you can use the following command: `npx yarn start [app]`

Example: `npx yarn start @clp/main` - starts main application with all sub systems.
Example: `npx yarn start @clp/ele` - starts only electrical app.

Optimizing SVG images

To optimize all SVG images within an app use: `npx yarn svgo [app]`

Example: `npx yarn svgo apps/ele`

Creating a production build

To create an optimized build to be shipped on web servers use: `npx yarn build`

This will create a build in the folder `/build` which can be shipped on web servers.

Apps

All single apps are contained in the `/apps` folder for further configuration.

Folder structure

All apps use a generic folder structure like:

```
.
|-- config - individual config files
|-- node_modules - app dependencies (auto generated)
|-- public - keeps the driver file (index.html)
|-- src - source code written in React/JSX
|-- jsconfig.json - CRA configuration
|-- package.json - app configuration
`-- yarn.lock - dependency lockins
```

Configuration structure

Each app consists of single pages which can be configured via a `config.js` file:

```
.
|-- ac-system
|   |-- pic
|   `-- config.js
|-- cb-location
|   |-- desc
|   |-- pic
|   `-- config.js
|-- dc-system
|   |-- pic
|   `-- config.js
|-- overview
|   |-- pic
|   `-- config.js
|-- relay-finder
|   |-- desc
|   |-- pic
|   `-- config.js
`-- package.json
```

For each page there is a dedicated folder with the name of the page. Each page has a `config.js` for fine-grained configuration. Images like SVG etc. live in `pic`, whereas all custom HTML files should live in `desc` - these files are mostly referenced from `config.js` file.

Root and relative paths

Please make sure that you always use relative paths across all files, whereas they have to be relative to `clp` folder, which acts as root folder. Some examples may help:

- link to a file in app's config: `/apps/ele/config/overview/pic/some-svg.svg`
- link to a global unit file: `/units/ele/overview/units/unit.html`

Components

For convenience there are some web components available when writing HTML files to keep workload low and keep consistency. You can use them across all HTML files like units and descs.

Images

To link to images or svgs with relative paths use: `<clp-img src="./relative/path/to/file">Text</clp-img>`

Example: `<clp-img src="./units/ele/overview/unit/unit.svg"></clp-img>`

`width` : Set a custom width in px.
`height` : Set a custom height in px.
`icon` : Inlines and aligns vertically centered.

Text colors

To make part of a text orange use: `<clp-text-orange>Orange</clp-text-orange>`
To make part of a text green use: `<clp-text-green>Green</clp-text-green>`
To make part of a text blue use: `<clp-text-blue>Blue</clp-text-blue>`
To make part of a text red use: `<clp-text-red>Red</clp-text-red>`

Example: `I am <clp-text-red>a red text</clp-text-red>.`

Text alignment

To align text left use: `<clp-text-left>Left text</clp-text-left>`
To align text centered use: `<clp-text-centered>Centered Text</clp-text-centered>`
To align text right use: `<clp-text-right>Right text</clp-text-right>`

Example: `<clp-text-centered>I am centered text.</clp-text-centered>`

Text size

To make text xx-large use: `<clp-text-xx-large>XX-Large text</clp-text-xx-large>`
To make text x-large use: `<clp-text-x-large>X-Large text</clp-text-x-large>`
To make text large use: `<clp-text-large>Large text</clp-text-large>`
To make text medium use: `<clp-text-medium>Medium text</clp-text-medium>`
To make text small use: `<clp-text-small>Small text</clp-text-small>`
To make text x-small use: `<clp-text-x-small>X-Small text</clp-text-x-small>`
To make text xx-small use: `<clp-text-xx-small>XX-Small text</clp-text-xx-small>`

HTML links

To create links relative to CLP root directory use: `<clp-a href="./path/to/file.pdf">My Link Text</clp-link>`

Example: `<clp-a href="units/pdf/epu.pdf">EPU</clp-a>`

Nav links

To create links navigating within the system use: `<clp-link-nav path="/path/to/page"></clp-link-nav>`

Example: `<clp-link-nav path="/engine/apu/">APU system</clp-link-nav>`

PDF links

To create static links to PDF files use: `<clp-link-pdf>[SYS] TASK [NUM]</clp-link-pdf>`

Example: `<clp-link-pdf>AMM TASK 12-10-32-210-801</clp-link-pdf>`

TR links

To create static links to TR files use: `<clp-link-tr>(TR-[SYS])</clp-link-tr>`

Example: `<clp-link-tr>(TR-AIPC)</clp-link-tr>`

Ref links

To create interactive links for references use: `<clp-link-ref id="refId">Text</clp-link-ref>`

Example: `<clp-link-ref id="clp-rjb-1">Relay Junction Box 1</clp-link-ref>`

Info messsages

To make text appear in a orange info box use: `<clp-info-orange>Orange</clp-info-orange>`
To make text appear in a green info box use: `<clp-info-green>Green</clp-info-green>`
To make text appear in a blue info box use: `<clp-info-blue>Blue</clp-info-blue>`
To make text appear in a grey info box use: `<clp-info-grey>Grey</clp-info-grey>`
To make text appear in a red info box use: `<clp-info-red>Red</clp-info-red>`
To make text appear in a white info box use: `<clp-info-white>White</clp-info-white>`
To make text appear in a bordered box use: `<clp-info-border>Border</clp-info-border>`

To make info box auto-size to content use: `<clp-info-white inline="">White</clp-info-white>`

Example 1 (text only):

```
<clp-info-red>Im an informative text.</clp-info-red>
```

Example 2 (with header & image):

```
<clp-info-red>
  <strong>HEADER</strong><br>
  <clp-table>
    <clp-tr>
      <clp-td>Im an informative text.</clp-td>
      <clp-td><div><clp-img src="./path/to/an/image.svg"></clp-img></div></clp-td>
    </clp-tr>
  </clp-table>
</clp-info-red>
```

Popup links

To create a link which opens a popup with configured HTML file use: `<clp-link-popup path="./path/to/html.file">Text</clp-link-popup>`

Example: `<clp-link-popup path="./units/cockpit/units/epcu.html">EPCU</clp-link-popup>`

`path` : Path to html file for popup content
`name` : Optional title for popup shown at the top
`size` : Optional predefined size: `size-1` , `size-2` , `size-3` (default: auto)

Files

An element to load and display HTML files in-place:

```
<clp-file path="./path/to/file.html"></clp-file>
```

Example: `<clp-file path="./units/cockpit/units/epcu.html"></clp-file>`

Tables

An element to make consistent tables with or without border:

```
<clp-table>
  <clp-tr>
    <clp-td>Cell #1</clp-td>
    <clp-td>Cell #2</clp-td>
  </clp/tr>
</clp-table>
```

For tables with transparent background: `<clp-table bg-transparent="">`
For tables with borders add the following attribute: `<clp-table border="">`
For tables with hover effect add the following attribute: `<clp-table hover="">`
For tables with 100% width add the following attribute: `<clp-table fullwidth="">`
For tables avoiding wrapping of column (one from 1-10) use: `<clp-table nowrap="1>`
For tables with vertically centered columns: `<clp-table vertical-align="middle">`
For tables with vertically bottomed columns: `<clp-table vertical-align="bottom">`
For specifying width of columns use (% from 1-100): `<clp-td width="50%"></clp-td>`

Cards

An element to make a blue card with a name:

```
<clp-card name="Name">
  <p>
    Some HTML here
  </p>
</clp-card>
```

Cards can also contain buttons by placing `<clp-card-button>` elements inside `<clp-card>`:

```
<clp-card name="Name">
  <!-- button opening another page -->
  <clp-card-button path="/ele" icon="INFO" mode="LINK"></clp-card-button>

  <!-- button opening a unit within card -->
  <clp-card-button path="./units/apu/units/apu_gen.html" icon="INFO" mode="INLINE">
</clp-card-button>

  <!-- button opening a unit in a popup -->
  <clp-card-button path="./units/apu/units/apu_gen.html" icon="WRENCH" mode="POPUP"
title="APU UNIT"></clp-card-button>
</clp-card>
```

`path`: Path to html/popup (`INLINE` / `POPUP`) or to page (`LINK`).
`icon`: One of `INFO` (default) or `WRENCH`
`mode`: One of `INLINE`, `POPUP` or `LINK`
`name`: Optional name for popup (read more at `<clp-link-popup>`).
`size`: Optional size for popup (read more at `<clp-link-popup>`).

If you want a card to not automatically expand to its parent container's height use `stretch`:

```
<clp-card name="Name" stretch="false"></clp-card>
```

Grids

An element to make consistent grids:

```
<clp-grid>
  <clp-grid-column>Column #1</clp-grid-column>
  <clp-grid-column>Column #2</clp-grid-column>
  <clp-grid-column>Column #3</clp-grid-column>
</clp-grid>
```

Accordion

An element to wrap contents into expandable items to save space and toggles opened items.

```
<clp-accordion>
  <clp-expandable name="Expandable #1">Expandable #1</clp-expandable>
  <clp-expandable name="Expandable #2">Expandable #2</clp-expandable>
  <clp-expandable name="Expandable #3">Expandable #3</clp-expandable>
</clp-accordion>
```

`name`: Name for expandable to be shown in header

Note: You can also use `<clp-expandable>` standalone!

Tabs

An element to wrap contents into single tabs to save space.

```
<clp-tabs>
  <clp-tab name="Tab #1">Tab #1</clp-tab>
  <clp-tab name="Tab #2">Tab #2</clp-tab>
  <clp-tab name="Tab #3">Tab #3</clp-tab>
</clp-tabs>
```

`name` : Name for tab to be shown in navbar.

SVG

An element to display interactive SVG with config options to navigate, open popup or show inline HTML.

```
<clp-svg path="./path/to/file.svg">
  <clp-svg-config ref="clp-id1" mode="LINK" path="/path/to/system"></clp-svg-config>
  <clp-svg-config ref="clp-id2" mode="POPUP" path="./path/to/file.html" name="MY
POPUP"></clp-svg-config>
  <clp-svg-config ref="clp-id3" mode="INLINE" path="./path/to/file.html" name="MY
HTML"></clp-svg-config>
  <clp-svg-intro path="./path/to/intro.file"></clp-svg-intro>
</clp-svg>
```

Attributes for `<clp-svg-config>` :

`ref` : ID of the layer in the SVG.
`path` : path to html (`INLINE` or `POPUP`) or link (`LINK`).
`mode` : `INLINE` , `POPUP` or `LINK`
`name` : text for box or popup (`INLINE` or `POPUP`).
`size` : `size-1` , `size-2` , `size-3` for `POPUP`

Attributes for `<clp-svg-intro>` :

`path` : path to html file shown initially. Toggles if mode is `INLINE`