Feature Toggles

The configuration api provides an endpoint for collecting feature toggles defined in the github repo configuration-api-files.

We primarily use Feature Toggles as a release mechanism to support continous delivery as well as timed releases.

Adding a new feature toggle

To add a new feature toogle, create a PR in the `configuration-api-files` in the main file that relates to your project/team, in each environment directory.

NB: As a general rule of thumb, its good to enable to feature toggle for `development` intially, whil having it off for the others until you need to toggle.

In the `@mtfh/common/lib/configuration` you will need to define the scope, as well as the feature toggles the frontend needs to support.

```
const initialFeatureToggles = {
  MMH: {
    Test: false,
    TenureActivityHistory: false,
    RefactorComments: false,
    CreateTenure: false,
    EditTenure: false,
  },
};
```

This step allows us to enforce feature toggle names in our ui.

Using a feature toggle

In react, we provide a hook to access feature toggles within React components:

```
import { useFeatureToggle } from '@mtfh/common/lib/hooks'

const View = () => {
    const hasEditTenure = useFeatureToggle('MMH.EditTenure');
}
```

Outside of React:

```
import { hasToggle } from '@mtfh/common/lib/configuration'

const hasEditTenure = hasToggle('MMH.EditTenure')
```

Released Deployments

Our micro-frontends are setup for continous delivery through trunk based git flow, and as a result all unreleased features that introduce change should be feature toggled. This is so developers can continosly work on features and still allow hot fixes.

A helpful strategy to reduce complexity and avoid deeply nested feature toggles, we recommend duplicating the top level views and marking the current version as legacy.

```
mtfh-frontend-tenure
├── node_modules
├── src
│   ├── components
│   └── views
│       ├── edit-tenure-legacy
│       └── edit-tenure
└── app.tsx
```

In app.tsx:

```
import { useFeatureToggle } from '@mtfh/common/lib/hooks';
import { EditTenureViewV2, EditTenureView } from './views';

export default function App(): JSX.Element {
  const hasEditTenureV2 = useFeatureToggle('MMH.EditTenureV2');

  return (
    <Switch>
      <Route path="/tenure/:tenureId/edit">
        {hasEditTenureV2 ? <EditTenureView /> : <EditTenureViewLegacy />}
      </Route>
    </Switch>
  )
}
```