Universal-React-Redux-Boilerplate-PLUS / React Helment

- We should use React-Helmet to add information to the `head` of the document which can be written to when certain components are loaded.
- nested components will take precedence over parent elements.
- allows for server-side rendering.
- `react-helmet` needs to know where to load this stuff in when it is created in your components, so we need to set-up some parameters when we create our `html` object.

```
  const Html = (props: Object) => {
    const { markup, state, assets: { styles, javascript, assets }, helmet } = props

    const htmlAttrs = helmet.htmlAttributes.toComponent();
    const bodyAttrs = helmet.bodyAttributes.toComponent();

    return (
      <html {...htmlAttrs}>
        <head>
          <meta charSet="UTF-8" />
            {helmet.title.toComponent()}
            {helmet.base.toComponent()}
            {helmet.meta.toComponent()}
            {helmet.link.toComponent()}
            {helmet.script.toComponent()}
            {helmet.noscript.toComponent()}
            {helmet.and.toComponent() style}
          <meta name="viewport" content="width=device-width, initial-scale=1.0" />
          {
            Object.keys(styles).map(key => (
              <link key={ key } rel="stylesheet" href={ styles[key] } />
            ))
          }
          { __DEV__ && <style dangerouslySetInnerHTML={ { __html: getStyles(assets) } } /> }
        </head>
        <body {...bodyAttrs}>
          <div id="app" dangerouslySetInnerHTML={ { __html: markup } } />
          <script dangerouslySetInnerHTML={ { __html: getInitialState(state) } } />
          <script src={ javascript.app } />
        </body>
      </html>
    )
  }
```

react-helmet properties:

```
 /**
      * @param {Object} base: {"target": "_blank", "href": "http://mysite.com/"}
      * @param {Object} bodyAttributes: {"className": "root"}
      * @param {String} defaultTitle: "Default Title"
      * @param {Boolean} encodeSpecialCharacters: true
      * @param {Object} htmlAttributes: {"lang": "en", "amp": undefined}
      * @param {Array} link: [{"rel": "canonical", "href": "http://mysite.com/example"}]
      * @param {Array} meta: [{"name": "description", "content": "Test description"}]
      * @param {Array} noscript: [{"innerHTML": "<img src='http://mysite.com/js/test.js'"}]
      * @param {Function} onChangeClientState: "(newState) => console.log(newState)"
      * @param {Array} script: [{"type": "text/javascript", "src": "http://mysite.com/js/test.js"}]
      * @param {Array} style: [{"type": "text/css", "cssText": "div { display: block; color: blue; }"}]
      * @param {String} title: "Title"
      * @param {Object} titleAttributes: {"itemprop": "name"}
      * @param {String} titleTemplate: "MySite.com - %s"
      */
```

react-helmet - All Examples

```
<Helmet
    encodeSpecialCharacters={true}
    titleTemplate="MySite.com - %s"
    // fallback if not template is inheriting title
    defaultTitle="My Default Title"
    // `newState` returns an object with all the new Helmet attributes in the Component
    onChangeClientState={(newState) => {console.log(newState) }}
>
    <html lang="en" amp />
    <body className="bodywrapper-blog" />
    <title itemProp="name" lang="en">My Title</title>
    {/* sets the base for relative resources/links */}
    <base target="_self" href="//localhost:3000/" />
    {/* add meta-data */}
    <meta name="description" content="Helmet application" />
    <meta property="og:type" content="article" />

    <link rel="canonical" href="https://localhost:3000/" />
    <link rel="apple-touch-icon" href="http://localhost:3000/img/apple-touch-icon-57x57.png" />
    <link rel="apple-touch-icon" sizes="72x72" href="http://localhost:3000/img/apple-touch-icon-72x72.png" />
    {/* add resources */}
    <link href="https://cdn.rawgit.com/michalsnik/aos/2.1.1/dist/aos.css" rel="stylesheet" />
    <script src="https://cdn.rawgit.com/michalsnik/aos/2.1.1/dist/aos.js" type="text/javascript" />
    {/* inline elements */}
    <script type="application/ld+json">{`
        {
            "@context": "http://schema.org"
        }
    `}</script>
    <noscript>{`
        <link rel="stylesheet" type="text/css" href="foo.css" />
    `}</noscript>
    <style type="text/css">{`
        body {
            background-color: blue;
        }
    `}</style>
</Helmet>
```

Let's look at a Example:

- going to be updating the `title` tag and `body` CSS className on each Component

`App.js` (The App Wrapper - will be shown on all pages unless nested components override it.)

- `%s` : *Useful when you want titles to inherit from a template:*
- also notice the `bodyAttributes` property needs to be encapsulated in TWO pairs of curly braces, once to parse as JS, 2nd time becuase it is an Object!

```
<Helmet
    titleTemplate="%s | Your Site Name"
    defaultTitle="Your Site Name"
    bodyAttributes={{"className": "pagewrapper-app"}}
    >
    <meta name="description" content="This will show up on SERPs" />
</Helmet>
```

- include `defaultTemplate` for the actual page as a fallback title

`BlogIndex.js`

```
<Helmet>
    <title>How to Use React-Helmet</title>
    <meta name="description" content="This is our blog" />
    <bodyAttributes className="pagewrapper-blog" />
</Helmet>
```

When you switch to ANY page that does not have `<Helmet />` be aware that the react-helmet code in `App.js` will be run each time, meaning all pages will have a `body` classname of `pagewrapper-app` and a `title` of `Your Site Name` unless otherwise specified.

… the `title` on the Blog page would be shown in the browser tab as `How to Use React-Helmet | Your Site Name`

EXTRA SHIT - Appending a bodyAttr instead of replacing it:

- what's interesting about this? Remember that this code is isomorphic–meaning it runs on both the server and the client. When this runs on the server there is no `document` object since this comes from the browser and node clearly is not! Typically you wouldn't need to check if `window` exists, but we need to be sure the server doesn't throw an error when looking for `document`. This ensures that chunk of code is only run on the client.

```
const bodyClassesToAdd = "pagewrapper-blog something-else"
let bodyClassNames = ""
if(typeof window !== 'undefined') {
    bodyClassNames = !document.querySelector("body").getAttribute("class").includes(bodyClassesToAdd) ? (document.querySelector("body").getAttribute("class") + " " + bodyClassesToAdd)  : document.querySelector("body").getAttribute("class")
};

<Helmet titleTemplate="%s | Your Site Knowledge Base" bodyAttributes={{"class":bodyClassNames}}>
```