React to PDF

React Van Jan 2020
Brian Zinn

yarn add @react-pdf/renderer

Text

Images

Resume

Fractals

Knobs

Page wrap

Examples from repo
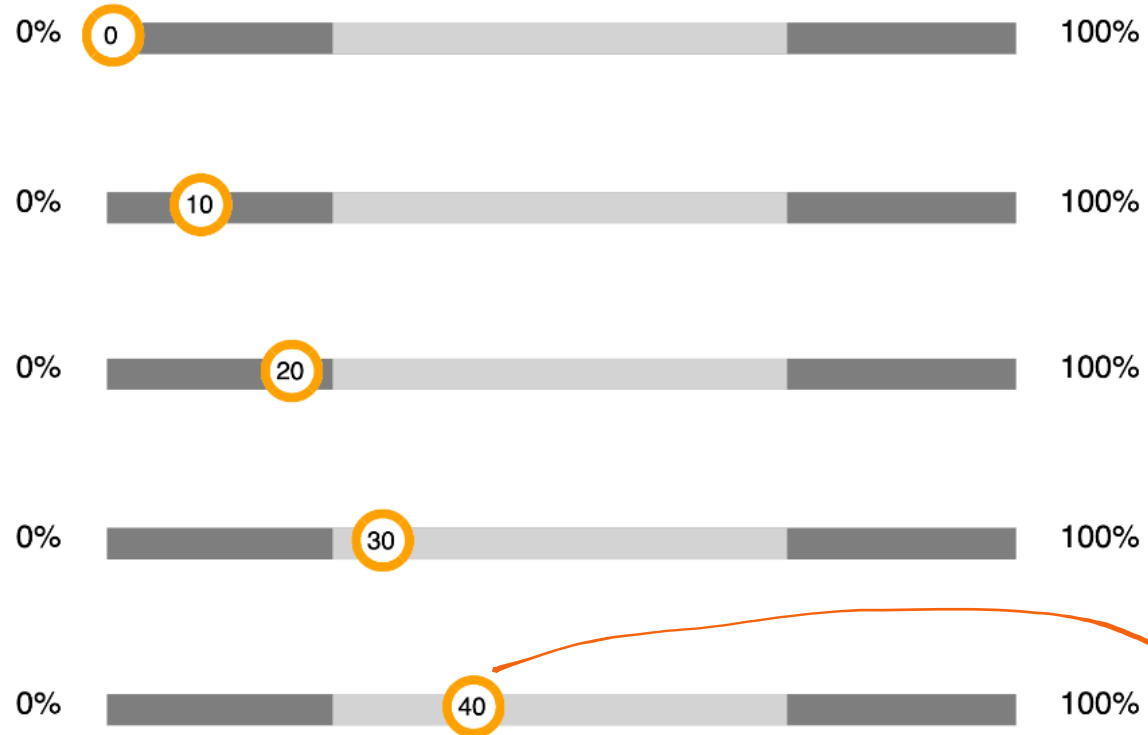
```
const styles = StyleSheet.create({
    knob: {
        alignItems: 'center',
        justifyContent: 'center',
        width: '20px',
        height: '20px',
        borderRadius: 10,
        borderColor: 'orange',
        /* ... */
    }
});
```

```jsx
const Knob = ({ value }) => (
  <View style={[styles.knob, { left: `${value - 3}%` }]}>
    <Text style={{ fontSize: 8, marginTop: 4 }}>{value}</Text>
  </View>
);

const Select = props => (
  <View style={styles.select}>
    <Text style={[styles.text, { marginRight: '15px' }]}>0%</Text>
    <View style={styles.bar}>
      <View style={styles.barMiddle} />
      <Knob {...props} />
    </View>
    <Text style={[styles.text, { marginLeft: '15px' }]}>100%</Text>
  </View>
);

const doc = (
  <Document>
    <Page size="A5">
      <Select value={0} />
      <Select value={10} />

      ...
      <Select value={90} />
      <Select value={100} />
    </Page>
  </Document>
);
```

This project is deprecated in favor for Puppeteer. Thanks to all the contributors who made this project possible.

Chromeless

npm `v1.5.2` downloads `6.6k/month` circleci codecov `unknown` dependencies `out of date` node `>= 6.10.0` semantic-release

Chrome automation made simple. Runs locally or headless on AWS Lambda. (**See Demo**)

Chromeless can be used to...

- Run 1000s of **browser integration tests in parallel** ⚡
- Crawl the web & automate screenshots
- Write bots that require a real browser
- *Do pretty much everything you've used* **PhantomJS, NightmareJS or Selenium** *for before*

Puppeteer

1. Generate screenshots and PDFs of pages.

2. Maintained by Chrome dev tools team
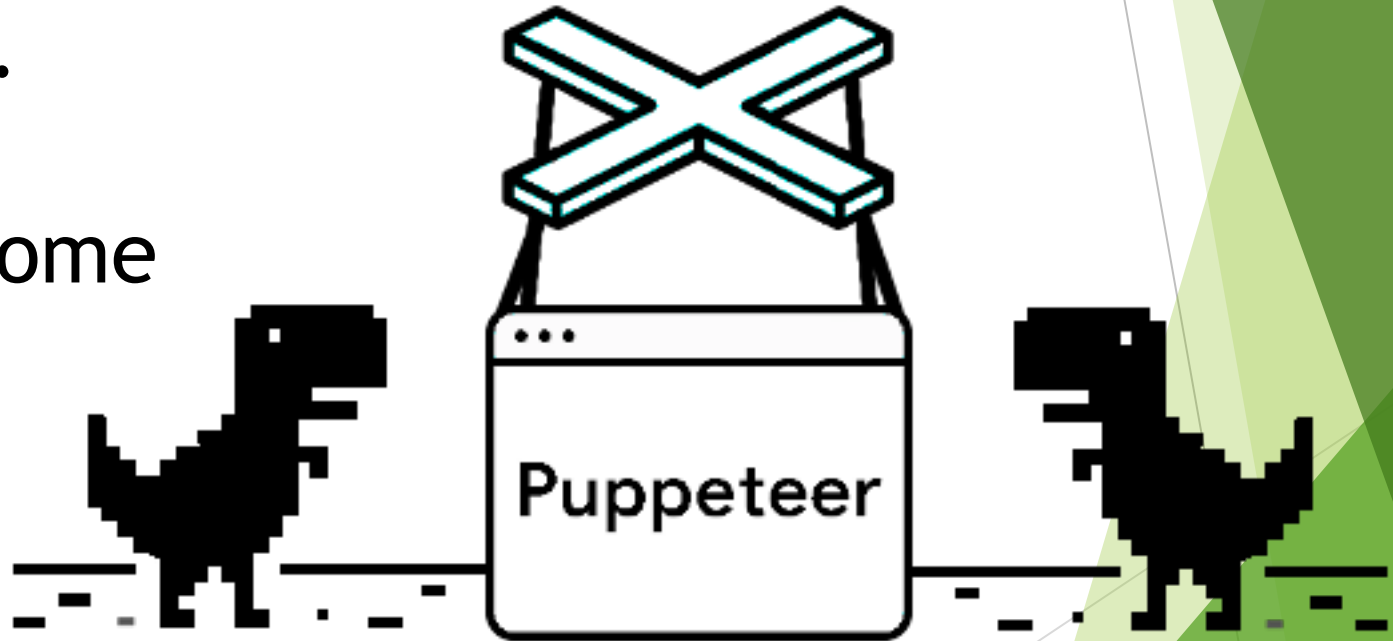
Check the releases

v2.0.0

mathiasbynens released this on Oct 24, 2019 · 19 commits to master since this release

Big changes

- Chromium 79.0.3942.0 (706915)
- Puppeteer now requires Node.js v8+; Node.js v6 is no longer supported

time to code...

Additional Considerations

▶ Not running in context of browser, so you may need:

  ▶ auth-context – don't do security as an afterthought

  ▶ browser headers (from request)

  ▶ client-browser settings (need to interrogate browser Time Zone / Language)

    ▶ From browser (can I use): timezone: Intl.DateTimeFormat().resolvedOptions().timeZone

    ▶ Fallback: new Date().getTimezoneOffset()

▶ When running automated consider adding user profile settings for ie: Time Zone/Language

  ▶ ie: as a weekly digest

  ▶ on event completion (online order billed/shipped, incident PM complete)

▶ Use CSS for to control breaking – consider adding @media print rules. ie:

  ▶ { break-inside: avoid }
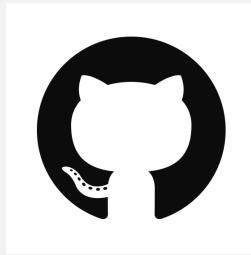
  ▶ { break-after: always}

pros

Puppeteer

- ▶ Automated generation
- ▶ SVG
- ▶ All React components and libraries, CSS
- ▶ 3rd party image integrations (slack)
- ▶ Same code base as website
- ▶ Any website - even Angular ☺
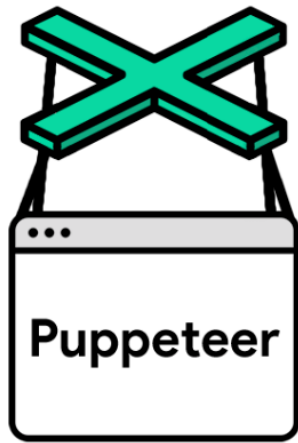
cons

- ▶ node (server-side)
- ▶ additional work to pass browser settings
- ▶ new security

thank-you

github.com/brianzinn/crapdf