Sign up

wojtekmaj / **react-pdf** Public

Sponsor | Notifications | Fork 567 | Star 5.5k

Code | Issues 84 | Pull requests 7 | Discussions | Actions | Wiki | Security | ...

main | Go to file | Code

wojtekmaj Replace cloudflare CDN and jsDelivr CDN with UNPKG ... ✓ 2 days ago 1,088

.github	Fix Environment rendered as markdown in YAML issu…	17 days ago
.yarn/releases	Update Yarn to 3.1.0	2 months ago
__mocks__	Update mock PDF	7 months ago
sample	Add copy-cmaps script to Parcel and Parcel 2 samples	3 days ago
src	Update instructions on PDF.js worker	3 days ago
test	Add dest and pageIndex to onItemClick callback (#924)	3 days ago
.babelrc	Remove unnecessary Babel plugin	4 months ago
.eslintignore	Extend ESLint to cover test folder	4 years ago
.eslintrc.json	Replace eslint-config-airbnb with eslint-config-wojtekmaj	15 months ago
.gitattributes	Update .gitattributes	3 years ago

About

Display PDFs in your React app as easily as if they were images.

projects.wojtekmaj.pl/react-pdf

react | pdf | pdf-viewer

Readme

MIT License

5.5k stars

50 watching

567 forks

Releases 98

v5.6.0 Latest
on Dec 13, 2021

.gitignore	Upgrade Yarn to Yarn Berry (v2) (#643)	16 months ago
.mailmap	Create .mailmap	17 days ago
.yarnrc.yml	Update Yarn to 3.1.0	2 months ago
LICENSE	Update year in LICENSE	13 months ago
README.md	Replace cloudflare CDN and jsDelivr CDN with UNPKG	2 days ago
copy-styles.js	[breaking] Create ESM builds (#429)	2 years ago
jest.config.json	Update Jest to 27.0.0	4 months ago
jest.env.js	Update Jest to 27.0.0	4 months ago
jest.setup.js	Update React to 17.0.0	13 months ago
package.json	Migrate from Error/errorOnDev/warnOnDev to tiny-inv…	3 days ago
test-utils.js	Fix restoreConsole test util not working properly	15 months ago
yarn.lock	Migrate from Error/errorOnDev/warnOnDev to tiny-inv…	3 days ago

README.md

npm v5.6.0 | downloads 21M | CI passing | tested with jest

React-PDF

Display PDFs in your React app as easily as if they were images.

+ 97 releases

Sponsor this project

wojtekmaj Wojciech Maj

opencollective.com/**react-pdf-w…**

Learn more about GitHub Sponsors

Packages

No packages published

Used by 14.6k

+ 14,543

Contributors 47

+ 36 contributors

Lost?

This package is used to *display* existing PDFs. If you wish to *create* PDFs using React, you may be looking for @react-pdf/renderer.

tl;dr

- Install by executing `npm install react-pdf` or `yarn add react-pdf`.
- Import by adding `import { Document } from 'react-pdf'`.
- Use by adding `<Document file="..." />`. `file` can be a URL, base64 content, Uint8Array, and more.
- Put `<Page />` components inside `<Document />` to render pages.

Demo

A minimal demo page can be found in `sample` directory.

Online demo is also available!

Before you continue

React-PDF is under constant development. This documentation is written for React-PDF 5.x branch. If you want to see documentation for other versions of React-PDF, use dropdown on top of GitHub page to switch to an appropriate tag. Here are quick links to the newest docs from each branch:

- v4.x

Languages

● **JavaScript** 95.0% ● **CSS** 3.2%
● **Less** 1.7% ● **HTML** 0.1%

- v3.x
- v2.x
- v1.x

Getting started

Compatibility

React

To use the latest version of React-PDF, your project needs to use React 16.3 or later.

If you use an older version of React, please refer to the table below to a find suitable React-PDF version. Don't worry - as long as you're running React 15.5 or later, you won't be missing out a lot!

React version	Newest compatible React-PDF version
≥16.3	latest
≥15.5	4.x

Internet Explorer

Internet Explorer is not supported in React-PDF 5.x or later. If you need Internet Explorer support, you will need to use React-PDF 4.x instead. Don't worry - it still receives bug fixes and even occasional small features!

Installation

Add React-PDF to your project by executing `npm install react-pdf` or `yarn add react-pdf`.

Usage

Here's an example of basic usage:

```
import React, { useState } from 'react';
import { Document, Page } from 'react-pdf';

function MyApp() {
  const [numPages, setNumPages] = useState(null);
  const [pageNumber, setPageNumber] = useState(1);

  function onDocumentLoadSuccess({ numPages }) {
    setNumPages(numPages);
  }

  return (
    <div>
      <Document
        file="somefile.pdf"
        onLoadSuccess={onDocumentLoadSuccess}
      >
        <Page pageNumber={pageNumber} />
      </Document>
      <p>Page {pageNumber} of {numPages}</p>
    </div>
  );
}
```

Check the sample directory in this repository for a full working example. For more examples and more advanced use cases, check Recipes in React-PDF Wiki.

Configure PDF.js worker

For React-PDF to work, PDF.js worker needs to be provided.

To make it easier, special entry files were prepared for most popular bundlers. You can find them in the table below.

For example, if you want to use React-PDF with Webpack 5, instead of writing:

```
import { Document, Page } from 'react-pdf';
```

write:

```
import { Document, Page } from 'react-pdf/dist/esm/entry.webpack';
```

Bundler	Entry file
Parcel 1	`react-pdf/dist/esm/entry.parcel`
Parcel 2	`react-pdf/dist/esm/entry.parcel2`
Webpack 4	`react-pdf/dist/esm/entry.webpack`
Webpack 5	`react-pdf/dist/esm/entry.webpack5`

Create React App

Create React App 4 (`react-scripts@4.0.0`) uses Webpack 4 under the hood, so you can use the entry file built for Webpack 4.

Create React App 5 (`react-scripts@5.0.0`) uses Webpack 5 under the hood, so the aim is to use the entry file built for Webpack 5. However, the way Webpack is configured in CRA 5 causes it to crash at build time on most machines with *JavaScript heap out of memory* error.

Standard instructions will also work with Create React App. Please note that in CRA, you can copy `pdf.worker.js` file from `pdfjs-dist/legacy/build` to `public` directory in order for it to be copied to your project's output folder at build time.

Standard (Browserify, esbuild and others)

If you use Browserify, esbuild, or other bundlers, you will have to make sure on your own that `pdf.worker.js` file from `pdfjs-dist/legacy/build` is copied to your project's output folder.

For example, you could use a custom script like:

```
import path from 'path';
import fs from 'fs';

const pdfjsDistPath = path.dirname(require.resolve('pdfjs-dist/package.json'));
const pdfWorkerPath = path.join(pdfjsDistPath, 'legacy', 'build', 'pdf.worker.j

fs.copyFileSync(pdfWorkerPath, './dist/pdf.worker.js');
```

If you don't need to debug `pdf.worker.js` , you can use `pdf.worker.min.js` file instead, which is roughly half the size. For this to work, however, you will need to specify `workerSrc` manually like so:

```
import { pdfjs } from 'react-pdf';
pdfjs.GlobalWorkerOptions.workerSrc = 'pdf.worker.min.js';
```

Alternatively, you could use the minified `pdf.worker.min.js` from an external CDN:

```
import { pdfjs } from 'react-pdf';
pdfjs.GlobalWorkerOptions.workerSrc = `//unpkg.com/pdfjs-dist@${pdfjs.version}/
```

Support for annotations

If you want to use annotations (e.g. links) in PDFs rendered by React-PDF, then you would need to include stylesheet necessary for annotations to be correctly displayed like so:

```
import 'react-pdf/dist/esm/Page/AnnotationLayer.css';
```

Support for non-latin characters

If you want to ensure that PDFs with non-latin characters will render perfectly, or you have encountered the following warning:

```
Warning: The CMap "baseUrl" parameter must be specified, ensure that the "cMapU
```

then you would also need to include cMaps in your build and tell React-PDF where they are.

Copying cMaps

First, you need to copy cMaps from `pdfjs-dist` (React-PDF's dependency - it should be in your `node_modules` if you have React-PDF installed). cMaps are located in `pdfjs-dist/cmaps`.

Webpack

Add `copy-webpack-plugin` to your project if you haven't already:

```
npm install copy-webpack-plugin --save-dev
```

Now, in your Webpack config, import the plugin:

```
import path from 'path';
import CopyWebpackPlugin from 'copy-webpack-plugin';
```

and in `plugins` section of your config, add the following:

```
new CopyWebpackPlugin({
  patterns: [
    {
      from: path.join(path.dirname(require.resolve('pdfjs-dist/package.json')),
      to: 'cmaps/'
    },
  ],
}),
```

Parcel, Browserify and others

If you use Parcel, Browserify or other bundling tools, you will have to make sure on your own that cMaps are copied to your project's output folder.

For example, you could use a custom script like:

```
import path from 'path';
import fs from 'fs';

const cMapsDir = path.join(path.dirname(require.resolve('pdfjs-dist/package.jso

function copyDir(from, to) {
  // Ensure target directory exists
  fs.mkdirSync(to, { recursive: true });

  const files = fs.readdirSync(from);
  files.forEach((file) => {
    fs.copyFileSync(path.join(from, file), path.join(to, file));
  });
}

copyDir(cMapsDir, 'dist/cmaps/');
```

Setting up React-PDF

Now that you have cMaps in your build, pass required options to Document component by using `options` prop, like so:

```
<Document
  options={{
    cMapUrl: 'cmaps/',
    cMapPacked: true,
  }}
/>
```

Alternatively, you could use cMaps from external CDN:

```
import { pdfjs } from 'react-pdf';

<Document
  options={{
    cMapUrl: `//unpkg.com/pdfjs-dist@${pdfjs.version}/cmaps`,
    cMapPacked: true,
  }}
/>
```

User guide

Document

Loads a document passed using `file` prop.

Props

Prop name	Description	Default value	Exam

Prop name	Description	Default value	Exam
className	Class name(s) that will be added to rendered element along with the default `react-pdf__Document`.	n/a	• String: `"custom-clas` `class-name-2` • Array of string `["custom-cla` `"custom-clas`
error	What the component should display in case of an error.	`"Failed to load PDF file."`	• String: `"An error oc` • React elemen `<div>An erro` • Function: `this.renderE`
externalLinkTarget	Link target for external links rendered in annotations.	unset, which means that default behavior will be used	One of valid values • `"_self"` • `"_blank"` • `"_parent"` • `"_top"`
file	What PDF should be displayed. Its value can be an URL,	n/a	• URL: `"http://exam`

Prop name	Description	Default value	Exam
	a file (imported using `import ... from ...` or from file input form element), or an object with parameters (`url` - URL; `data` - data, preferably Uint8Array; `range` - PDFDataRangeTransport; `httpHeaders` - custom request headers, e.g. for authorization), `withCredentials` - a boolean to indicate whether or not to include cookies in the request (defaults to `false`). **Warning**: Since equality check (`===`) is used to determine if `file` object has changed, it must be memoized by setting it in component's state, `useMemo` or other similar technique.		• File: `import samp` `'../static/s` `sample` • Parameter ob `{ url:` `'http://exam` `httpHeaders:` `'40359820958` `withCredenti`

Prop name	Description	Default value	Exam
imageResourcesPath	The path used to prefix the src attributes of annotation SVGs.	n/a (pdf.js will fallback to an empty string)	`"/public/images`

inputRef	A prop that behaves like ref, but it's passed to main `<div>` rendered by `<Document>` component.	n/a	• Function: `(ref) => { t` `ref; }`
			• Ref created u
			`React.create`
			`this.ref = R`
			`…`
			`inputRef={th`
			• Ref created u
			`const ref =`
			`…`
			`inputRef={re`

loading	What the component should display while loading.	`"Loading PDF…"`	• String: `"Please wai` • React elemen `<div>Please` • Function: `this.renderL`

Prop name	Description	Default value	Exam
noData	What the component should display in case of no data.	`"No PDF file specified."`	• String: `"Please sel` • React elemen `<div>Please </div>` • Function: `this.render`
onItemClick	Function called when an outline item has been clicked. Usually, you would like to use this callback to move the user wherever they requested to.	n/a	`({ dest, pageIn => alert('Clicke ' + pageNumber +`
onLoadError	Function called in case of an error while loading a document.	n/a	`(error) => aler loading document error.message)`
onLoadProgress	Function called, potentially multiple times, as the loading progresses.	n/a	`({ loaded, tota alert('Loading a (loaded / total)`

Prop name	Description	Default value	Exam
onLoadSuccess	Function called when the document is successfully loaded.	n/a	`(pdf) => alert(` `' + pdf.numPages`
onPassword	Function called when a password-protected PDF is loaded.	A function that prompts the user for password	`(callback) =>` `callback('s3cr3t`
onSourceError	Function called in case of an error while retrieving document source from `file` prop.	n/a	`(error) => aler` `retrieving docum` `error.message)`
onSourceSuccess	Function called when document source is successfully retrieved from `file` prop.	n/a	`() => alert('Do` `retrieved!')`
options	An object in which additional parameters to be passed to PDF.js can be defined. For a full list of possible parameters, check PDF.js documentation on DocumentInitParameters.	n/a	`{ cMapUrl: 'cma` `true }`

Prop name	Description	Default value	Exam
renderMode	Rendering mode of the document. Can be `"canvas"`, `"svg"` or `"none"`.	`"canvas"`	`"svg"`
rotate	Rotation of the document in degrees. If provided, will change rotation globally, even for the pages which were given `rotate` prop of their own. `90` = rotated to the right, `180` = upside down, `270` = rotated to the left.	n/a	`90`

Page

Displays a page. Should be placed inside `<Document />`. Alternatively, it can have `pdf` prop passed, which can be obtained from `<Document />`'s `onLoadSuccess` callback function, however some advanced functions like linking between pages inside a document may not be working correctly.

Props

Prop name	Description	Default value	Example
canvasBackground	Canvas background color. Any valid `canvas.fillStyle` can be used. If you set `renderMode` to `"svg"` this prop will be ignored.	n/a	`"transparen`
canvasRef	A prop that behaves like ref, but it's passed to `<canvas>` rendered by `<PageCanvas>` component. If you set `renderMode` to `"svg"` this prop will be ignored.	n/a	• Function: `(ref) =` `this.myP` `}` • Ref creat `React.c` `this.re` `React.cr` `…` `inputRe` `{this.re` • Ref creat `React.u` `const r` `React.us` `…` `inputRe`

Prop name	Description	Default value	Example
className	Class name(s) that will be added to rendered element along with the default `react-pdf__Page`.	n/a	• String: `"custom name-1 c class-na` • Array of s `["custor name-1", class-na`
customTextRenderer	A function that customizes how a text layer is rendered. Passes itext item and index for item.	n/a	`({ str, ite => { return {str}</mark>`
error	What the component should display in case of an error.	`"Failed to load the page."`	• String: `"An erre occurred` • React ele `<div>An occurred` • Function: `this.re`

Prop name	Description	Default value	Example
height	Page height. If neither `height` nor `width` are defined, page will be rendered at the size defined in PDF. If you define `width` and `height` at the same time, `height` will be ignored. If you define `height` and `scale` at the same time, the height will be multiplied by a given factor.	Page's default height	`300`
imageResourcesPath	The path used to prefix the src attributes of annotation SVGs.	n/a (pdf.js will fallback to an empty string)	`"/public/ima`

Prop name	Description	Default value	Example
inputRef	A prop that behaves like ref, but it's passed to main `<div>` rendered by `<Page>` component.	n/a	<ul><li>Function: `(ref) =` `this.myP` `}`</li><li>Ref creat `React.c` `this.re` `React.cr` `...` `inputRe` `{this.re`</li><li>Ref creat `React.u` `const r` `React.us` `...` `inputRe`</li></ul>

Prop name	Description	Default value	Example
loading	What the component should display while loading.	`"Loading page…"`	• String: `"Please` • React ele `<div>Pl` `</div>` • Function: `this.re`
noData	What the component should display in case of no data.	`"No page specified."`	• String: `"Please` `page."` • React ele `<div>Pl` `a page.<` • Function: `this.re`
onLoadError	Function called in case of an error while loading the page.	n/a	`(error) =>` `alert('Error` `loading page` `error.messag`

Prop name	Description	Default value	Example
onLoadSuccess	Function called when the page is successfully loaded.	n/a	`(page) => a` `displaying a` `number ' +` `page.pageNum`
onRenderError	Function called in case of an error while rendering the page.	n/a	`(error) =>` `alert('Error` `loading page` `error.messag`
onRenderSuccess	Function called when the page is successfully rendered on the screen.	n/a	`() => alert` `the page!')`
onGetAnnotationsSuccess	Function called when annotations are successfully loaded.	n/a	`(annotation` `alert('Now d` `' + annotati` `+ ' annotati`
onGetAnnotationsError	Function called in case of an error while loading annotations.	n/a	`(error) =>` `alert('Error` `loading anno` `+ error.mess`

Prop name	Description	Default value	Example
onGetTextSuccess	Function called when text layer items are successfully loaded.	n/a	`(items) => a` `displaying '` `items.length` `layer items!`
onGetTextError	Function called in case of an error while loading text layer items.	n/a	`(error) =>` `alert('Error` `loading text` `items! ' +` `error.messag`
pageIndex	Which page from PDF file should be displayed, by page index.	`0`	`1`
pageNumber	Which page from PDF file should be displayed, by page number. If provided, `pageIndex` prop will be ignored.	`1`	`2`
renderAnnotationLayer	Whether annotations (e.g. links) should be rendered.	`true`	`false`

Prop name	Description	Default value	Example
renderInteractiveForms	Whether interactive forms should be rendered. `renderAnnotationLayer` prop must be set to `true`.	`false`	`true`
renderMode	Rendering mode of the document. Can be `"canvas"`, `"svg"` or `"none"`.	`"canvas"`	`"svg"`
renderTextLayer	Whether a text layer should be rendered.	`true`	`false`
rotate	Rotation of the page in degrees. `90` = rotated to the right, `180` = upside down, `270` = rotated to the left.	Page's default setting, usually `0`	`90`
scale	Page scale.	`1.0`	`0.5`

Prop name	Description	Default value	Example
width	Page width. If neither `height` nor `width` are defined, page will be rendered at the size defined in PDF. If you define `width` and `height` at the same time, `height` will be ignored. If you define `width` and `scale` at the same time, the width will be multiplied by a given factor.	Page's default width	`300`

Outline

Displays an outline (table of contents). Should be placed inside `<Document />`. Alternatively, it can have `pdf` prop passed, which can be obtained from `<Document />`'s `onLoadSuccess` callback function.

Props

Prop name	Description	Default value	Example values

Prop name	Description	Default value	Example values
className	Class name(s) that will be added to rendered element along with the default `react-pdf__Outline`.	n/a	• String: `"custom-class-name-1 custom-class-name-2"` • Array of strings: `["custom-class-name-1", "custom-class-name-2"]`

Prop name	Description	Default value	Example values
inputRef	A prop that behaves like ref, but it's passed to main `<div>` rendered by `<Outline>` component.	n/a	• Function: `(ref) => { this.myOutline = ref; }` • Ref created using `React.createRef`: `this.ref = React.createRef();` … `inputRef={this.ref}` • Ref created using `React.useRef`: `const ref = React.useRef();` … `inputRef={ref}`

Prop name	Description	Default value	Example values
onItemClick	Function called when an outline item has been clicked. Usually, you would like to use this callback to move the user wherever they requested to.	n/a	`({ dest, pageIndex, pageNumber }) => alert('Clicked an item from page ' + pageNumber + '!')`
onLoadError	Function called in case of an error while retrieving the outline.	n/a	`(error) => alert('Error while retrieving the outline! ' + error.message)`
onLoadSuccess	Function called when the outline is successfully retrieved.	n/a	`(outline) => alert('The outline has been successfully retrieved.')`

Useful links

- React-PDF Wiki

License

The MIT License.

Author

	Wojciech Maj kontakt@wojtekmaj.pl https://wojtekmaj.pl

Thank you

This project wouldn't be possible without awesome work of Niklas Närhinen niklas@narhinen.net who created its initial version and without Mozilla, author of pdf.js. Thank you!

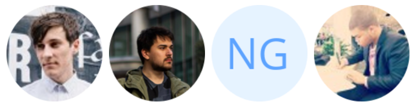
Sponsors

Thank you to all our sponsors! Become a sponsor and get your image on our README on GitHub.

Become a Sponsor

Backers

Thank you to all our backers! Become a backer and get your image on our README on GitHub.



Become a **Backer**

Top Contributors

Thank you to all our contributors that helped on this project!