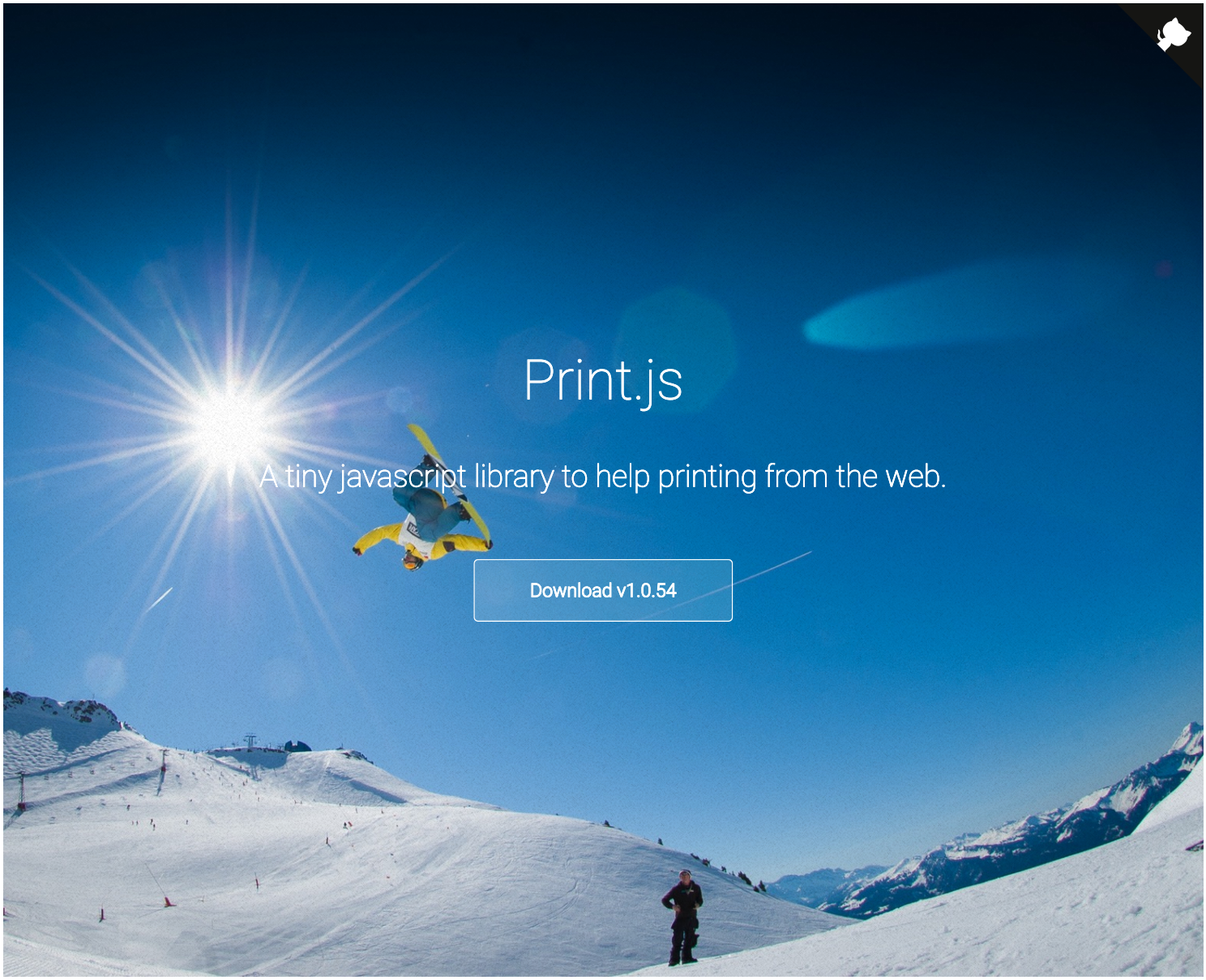

PDF Printing

Print.js was primarily written to help us print PDF files directly within our apps, without leaving the interface, an unique situations where there is no need for users to open or download the PDF files, and instead, they just need

One scenario where this is useful, for example, is when users request to print reports that are generated on the s are sent back as PDF files. There is no need to open these files before printing them. Print.js offers a quick way our apps.

> *PDF files must be served from the same domain as your app is hosted under. Print.js uses iframe to load files therefore, it is limited by the Same Origin Policy. This helps preventing Cross-site scripting (XSS) attacks.*

Example

Add a button to print a PDF file located on your hosting server:

```
<button type="button" onclick="printJS('docs/printjs.pdf')">
   Print PDF
</button>
```

Result:

Print PDF

> *Firefox currently doesn't allow printing PDF documents using iframes. There is an open bug in Mozilla's websit*
> *Firefox, Print.js will open the PDF file into a new tab.*

For large files, you can show a message to the user when loading files.

```
<button type="button" onclick="printJS({printable:'docs/xx_large_printjs.pdf', type:'pdf', showModal:true})">
   Print PDF with Message
</button>
```

Result:

Print Large PDF (5mb file)

Print Extra Large PDF (16mb file)

HTML Printing

Sometimes we just want to print selected parts of a HTML page, and that can be tricky. With Print.js, we can
element that we want to print. The element can be of any tag, as long it has a unique id. The library will try to pı
looks on screen, and at the same time, it will create a printer friendly format for it.

Example

Add a print button to a HTML form:

```
<form method="post" action="#" id="printJS-form">
   ...
</form>

<button type="button" onclick="printJS('printJS-form', 'html')">
   Print Form
</button>
```

Result:

Name:

John Doe

Email:

john@doe.com

Message:

Print HTML Elements

Print Form

Print.js accepts an object with arguments. Let's print the form again, but now we will add a header to the page:

```
<button type="button" onclick="printJS({ printable: 'printJS-form', type: 'html', header: 'PrintJS - Form Element Selection' })">
   Print Form with Header
</button>
```

Result:

Print Form with Header

Image Printing

Print.js can be used to quickly print any image on your page, by passing the image url. This can be useful when y
on the screen, using a low resolution version of the images. When users try to print the selected image, you can
url to Print.js.

Example

Load images on your page with just the necessary resolution you need on screen:

```
<img src="images/print-01.jpg" />
```

In your javascript, pass the highest resolution image url to Print.js for a better print quality:

```
printJS('images/print-01-highres.jpg', 'image')
```

Result:

> *Print.js uses promises to make sure the images are loaded before trying to print. This is useful when printing h that are not yet loaded, like the example above.*

You can also add a header to the image being printed:

```
printJS({printable: 'images/print-01-highres.jpg', type: 'image', header: 'My cool image header'})
```

Result:

Print Image With Header

To print multiple images together, we can pass an array of images. We can also pass the style to be applied on ea

```
printJS({
 printable: ['images/print-01-highres.jpg', 'images/print-02-highres.jpg', 'images/print-03-highres.jpg'],
 type: 'image',
 header: 'Multiple Images',
 imageStyle: 'width:50%;margin-bottom:20px;'
})
```

Result:

Print Multiple Images

JSON Printing

A simple and quick way to print dynamic data or array of javascript objects.

Example

We have the following data set in our javascript code. This would probably come from an AJAX call to a server AP

```
someJSONdata = [
   {
      name: 'John Doe',
      email: 'john@doe.com',
      phone: '111-111-1111'
   },
   {
      name: 'Barry Allen',
      email: 'barry@flash.com',
      phone: '222-222-2222'
   },
   {
      name: 'Cool Dude',
      email: 'cool@dude.com',
      phone: '333-333-3333'
   }
]
```

We can pass it to Print.js:

```
<button type="button" onclick="printJS({printable: someJSONdata, properties: ['name', 'email', 'phone'], type: 'json'})">
   Print JSON Data
</button>
```

Result:

Print JSON Data

We can style the data grid by passing some custom css:

```
<button type="button" onclick="printJS({
           printable: someJSONdata,
           properties: ['name', 'email', 'phone'],
           type: 'json',
           gridHeaderStyle: 'color: red;  border: 2px solid #3971A5;',
           gridStyle: 'border: 2px solid #3971A5;'
       })">
   Print JSON Data
</button>
```

Result:

Print Styled JSON Data

Download and Install

You can download the latest version of Print.js from the GitHub releases.

Download v1.0.54

To install using npm:

```
npm install print-js --save
```

To install using yarn:

```
yarn add print-js
```

When installing via npm or yarn, import the library into your project:

```
import print from 'print-js'
```

CDN is also available, thanks to KeyCDN:

```
https://printjs-4de6.kxcdn.com/print.min.js
https://printjs-4de6.kxcdn.com/print.min.css
```

Getting Started

First we need to include the Print.js library on the page.

```
<script src="print.js"></script>
```

If you will use the modal feature, also include Print.css on the page.

```
<link rel="stylesheet" type="text/css" href="print.css">
```

That's it. You can now use Print.js in your pages.

When writing your javascript code, remember that the library occupies a global variable of `printJS`.

Using Print.js

There are four print document types available: `'pdf'`, `'html'`, `'image'` and `'json'`.

The default type is `'pdf'`.

It's basic usage is to call `printJS()` and just pass in a PDF document url: `printJS('docs/PrintJS.pdf')`.

For image files, the idea is the same, but you need to pass a second argument: `printJS('images/PrintJS.jpg', 'image'`

To print HTML elements, in a similar way, pass in the element id and type: `printJS('myElementId', 'html')`.

When printing JSON data, pass in the data, type and the data properties that you want to print:
`printJS({printable: myData, type: 'json', properties: ['prop1', 'prop2', 'prop3']})`;

Configuration

Print.js will accept an object as argument, where you can configure some options:

Argument	Default Value	Description
printable	null	Document source: pdf or image url, html ele object.
type	'pdf'	Document type.
header	null	Optional header to be used with HTML, Ima will be placed on the top of the page.
headerStyle	'font-weight: 300;'	Optional header style to be applied to the he
maxWidth	800	Max document width in pixels. Change this printing HTML, Images or JSON.
css	null	This allow us to pass one or more css files applied to the html being printed. Value can URL or an array with multiple URLs.
style	null	This allow us to pass a string with custom s applied to the html being printed.
scanStyles	true	When set to false, the library will not proces html being printed. Useful when using the c
targetStyle	null	By default, the library process some styles c HTML elements. This option allows you to p that you want to be processed. Ex.: ['paddin
targetStyles	null	Same as `targetStyle`, however, this will proc styles. Ex.: ['border', 'padding'], will include 'b top', 'border-left', 'border-right', 'padding-top', You can also pass ['*'] to process all styles.
ignoreElements	[ ]	Accepts an array of html ids that should be parent html element.
properties	null	Used when printing JSON. These are the ob
gridHeaderStyle	'font-weight: bold;'	Optional style for the grid header when print
gridStyle	'border: 1px solid lightgray;	Optional style for the grid rows when printin

	margin-bottom: -1px;'	
repeatTableHeader	true	Used when printing JSON data. When set to header will show in first page only.
showModal	null	Enable this option to show user feedback w processing large PDF files.
modalMessage	'Retrieving Document...'	Message displayed to users when `showModal`
onLoadingStart	null	Function to be executed when PDF is being
onLoadingEnd	null	Function to be executed after PDF has loade
documentTitle	'Document'	When printing html, image or json, this will b document title. It will also be the name of th tries to save the print job to a pdf file.
fallbackPrintable	null	When printing pdf, if the browser is not com compatibility table), the library will open the allow you to pass a different pdf document t the original passed in `printable`. This may b javascript in your alternate pdf file.
onPdfOpen	null	When printing pdf, if the browser is not com compatibility table), the library will open the callback function can be passed here, which this happens. It may be useful in some situa handle your print flow, update user interface
onPrintDialogClose	null	Callback function executed once the browse
onError	error => throw error	Callback function to be executed when an e
honorMarginPadding (deprecated)	true	This is used to keep or remove padding and that are being printed. Sometimes these sty on screen but it looks pretty bad when printi by setting this to false.
honorColor (deprecated)	false	To print text in color, set this property to true be printed in black.
font (deprecated)	'TimesNewRoman'	Typeface used when printing HTML or JSON

font_size (deprecated)	'12pt'	Font size used when printing HTML or JSON
imageStyle (deprecated)	'width:100%;'	Used when printing images. Accepts a string be applied to each image.

Browser Compatibility

Currently, not all library features are working between browsers. Below are the results of tests done with these m latest versions.

	Google Chrome	Safari	Firefox	Edge	Opera	
PDF	✔	✔	✖	✖	✔	✖
HTML	✔	✔	✔	✔	✔	✔
Images	✔	✔	✔	✔	✔	✖
JSON	✔	✔	✔	✔	✔	✔

Thank you BrowserStack for the support. Amazing cross-browser testing tool.

BrowserStack

Development and Support

Please report issues and feature requests in GitHub Issues.

If you have questions when implementing or using the library, ask about it in StackOverflow.

Pull requests are very welcome! Make sure your patches are well tested: README.md.