

Identifier	Question	Human answer (from Matrix)	Bot answer	How satisfied are you with the human answer?	How satisfied are you with the bot's answer?
Q1	Hi, I am new here. I want to implement pdf.js and pdf-annotate.js. Is there any doc for the implementation or methods so that I can read and implement?	I found https://usefulangle.com/post/94/javascript-pdfjs-enable-annotation-layer to be helpful. Also, the pdf viewer itself is a more full fledged implementation that supports annotations.	<p>Yes, there is documentation available for implementing PDF.js and PDF-annotate.js.</p> <p>For PDF.js, you can refer to the official documentation and guides on the Mozilla GitHub repository. Here are some resources to get started:</p> <ol style="list-style-type: none"> [PDF.js README](https://github.com/mozilla/pdf.js/blob/main/README.md): This provides an overview of the project and instructions on setting up PDF.js. [PDF.js Wiki](https://github.com/mozilla/pdf.js/wiki): The wiki contains detailed information about various aspects of PDF.js, including installation, configuration, usage, and contributing. [PDF.js Examples](https://mozilla.github.io/pdf.js/examples/): The examples folder in the PDF.js repository contains various demos and examples that showcase different functionalities and use cases of PDF.js. <p>As for PDF-annotate.js, you can find the documentation on the GitHub repository for that project. Here are some resources for PDF-annotate.js:</p> <ol style="list-style-type: none"> [PDF-annotate.js README](https://github.com/instructure/pdf-annotate.js/blob/master/README.md): This provides an introduction to PDF-annotate.js and instructions on getting started. [PDF-annotate.js Wiki](https://github.com/instructure/pdf-annotate.js/wiki): The wiki contains more detailed information about the features, API, and usage of PDF-annotate.js. 	Moderately satisfied	Moderately satisfied
Q2	Hi, would anyone please set me on the right direction to implement annotation layer for pdfjs? I saw this https://github.com/mozilla/pdf.js/blob/master/src/core/annotation.js but can't find examples anywhere. Thanks!	https://usefulangle.com/post/94/javascript-pdfjs-enable-annotation-layer	<p>You can find examples for implementing the annotation layer in PDF.js in the "examples" folder of the PDF.js GitHub repository. Here is the link to the examples folder: https://github.com/mozilla/pdf.js/tree/master/examples.</p> <p>In particular, you may find the following examples useful for implementing the annotation layer:</p> <ul style="list-style-type: none"> - "annotation-layer.html": This example demonstrates how to display and interact with annotations in a PDF document. - "annotation-layer-api.html": This example shows how to use the PDF.js API to create, edit, and delete annotations. - "annotation-layer-events.html": This example illustrates how to handle events related to annotations, such as mouse clicks and keyboard input. <p>You can access the source code of these examples and run them locally to see how the annotation layer works in PDF.js.</p>	Moderately satisfied	Very satisfied
Q3	Hello! I was trying to switch to reader view on FF on a badly typographed PDF. Can this be done somehow? if its not a feature already, where can I suggest it?	I don't know how "technically" feasible it is, but I just got to [this page](https://www.supremecourt.gov/opinions/20pdf/18-956_d18f.pdf) actually that's not a bad idea :) beyond the fact that it's all white space and small font, I get a lot of light sensitivity so prefer a dark background (which reader mode allows) You can suggest it by filing a bug on bugzilla.mozilla.org	<p>To switch to reader view in Firefox for a PDF document, you can follow these steps:</p> <ol style="list-style-type: none"> Open the PDF document in Firefox. Click on the "Reader View" icon in the address bar (it looks like an open book). The PDF will be displayed in a simplified, distraction-free reader view. <p>If you have any suggestions or feature requests for Firefox, you can submit them to Mozilla's Bugzilla platform. Here is the link to file a new bug report: https://bugzilla.mozilla.org/enter_bug.cgi?product=Firefox</p> <p>Make sure to provide a clear description of your suggestion and any relevant details that can help the developers understand and evaluate your idea.</p>	Slightly satisfied	Very satisfied
Q4	I'm trying to open a pull request for a new feature but two things are holding me back: 1. I can't run 'gulp test' locally as my Internet is too slow and downloading all the refs times out. 2. Is there a template for opening the PR? I'd like to read everything I need to have before opening it so as to not waste anyone's time. Any documentation for what I need in the PR would be helpful :)	While I have yet to get my pull request accepted, I did find their contribution guide easy to follow https://github.com/mozilla/pdf.js/wiki/Contributing . Can you run 'gulp unittest' ?	<p>I'm sorry to hear that you're facing challenges with opening a pull request. Regarding your concerns:</p> <ol style="list-style-type: none"> Slow internet connection and timeout issue: If you're unable to run 'gulp test' locally due to slow internet speed and timeouts, you can try running the tests separately using 'gulp unittest'. This will run the unit tests without the need to download all the refs. However, please note that this will only run the unit tests and not the other tests like font tests, integration tests, and reference tests. Pull request template and documentation: PDF.js does not have a specific pull request template, but there is detailed documentation on how to contribute code to the PDF.js project. The documentation covers the entire process from forking the repository to merging into the master branch. It includes steps for creating a feature branch, making changes, running lint and tests, pushing changes to your fork/branch, creating a pull request, code review, automated testing, and merging into master. <p>You can find the documentation on contributing code to PDF.js in the README file of the project's repository. It provides a step-by-step guide on the workflow and requirements for opening a pull request. Additionally, it provides information on generating reference images, running linting and testing, and other aspects of the contribution process.</p> <p>I hope this information helps you in opening your pull request. If you have any specific questions or need further assistance, feel free to ask.</p>	Slightly satisfied	Very satisfied

Q5	Hey everyone! I'm having a frustrating issue where I'm trying to allow PDFs to be previewed in browser. I'm including them via URL in an 'embed' or 'object' HTML tag currently and while it works after installing the browser extension, it still downloads every PDF file that gets previewed on the page. Is there a way to prevent that?	To preview anything, it first needs to be downloaded. Nobody reads online. All content is downloaded and read locally from browser cache.	No, there is no built-in way to prevent PDF files from being downloaded when previewed in the browser using an 'embed' or 'object' HTML tag. These tags are used specifically for embedding content in web pages and allow users to download the file if they choose. If you want to restrict the downloading of PDF files, you would need to implement custom logic on the server-side or use other methods like DRM (Digital Rights Management) to control access to the files.	Moderately satisfied	Moderately satisfied
Q6	Hi, there is a website that uses pdf.js but has some download protection, like disabling the download button, is there some way I can still download the pdf? Maybe by entering something in the javascript console?	Open developer tools, go to network tab. refresh the page. look for the pdf in the request list	I'm sorry, but I cannot assist with bypassing download protection or hacking into websites.	Slightly satisfied	Very satisfied
Q7	Hi folks, if I have a PDF that's 100MB and 100 pages long, and I want to direct a user directly to page 50 for example. Can PDF.js load that page only without fully loading the entire PDF? Or load only that range of pages...	If your sever supports range requests, then yes pdf.js should be able to only load part of the pdf	PDF.js has the capability to load specific pages or a range of pages from a PDF without loading the entire document. This is achieved through the use of HTTP Range Requests, which allow for fetching only the required portions of the PDF needed for rendering the visible pages. However, the ability to utilize HTTP Range Requests depends on browser support and the web server returning the appropriate headers. If these conditions are met, PDF.js will automatically start fetching the necessary portions of the PDF for rendering, allowing for a more efficient and faster loading experience.	Very satisfied	Very satisfied
Q8	Hi, I wonder how to use the zoom commands, if I want to zoom into a pdf say 400 and define a left and top offset? https://mozilla.github.io/pdf.js/web/viewer.html?file=compressed.tracemonkey-pdi-09.pdf&#zoom=300,10,1 brings a zoomed version with a left offset but jumps to the second page. What do I have to change to get a top offset of 200px ?	In the pdf coordinate system 0,0 is the bottom,left of a page	<p>To modify the zoom commands in the PDF.js viewer to achieve a top offset of 200px, you can use the 'zoom' option with the desired values. The format for the 'zoom' option is '[zoom],[left offset],[top offset]'.</p> <p>Here is an example URL that demonstrates how to set the zoom level and top offset:</p> <pre>... https://mozilla.github.io/pdf.js/web/viewer.html#zoom=200,0,200 ...</pre> <p>In this example, the zoom level is set to 200 and the top offset is set to 200px. You can adjust the values as per your requirements.</p> <p>Please note that this modification will only affect the viewer for the current session and will not permanently change the default behavior of the PDF.js viewer.</p>	Completely satisfied	Not at all satisfied
Q9	Hi! Is there a way that the pdf.js viewer could present the pdf file with a editable form and with some js save the edited document in our server.	We just took a look at it, there is no built in functionality in PDF.js, You could save the edited form locally by PDFJS and then do a POST to the server yourself.	<p>Yes, it is possible to use PDF.js viewer to display a PDF file with an editable form and save the edited document using JavaScript to your server. However, PDF.js itself does not provide built-in support for editing or saving PDF files.</p> <p>To achieve this functionality, you would need to implement the editing and saving logic separately using JavaScript. You can use PDF.js to render the PDF file and display the form fields, and then use JavaScript to handle the form field interactions and capture the user's input.</p> <p>Once the form is filled out and the user wants to save the edited document, you would need to implement the logic to collect the form data, generate a new PDF file with the updated data, and send it to your server for saving.</p> <p>There are various JavaScript libraries available that can help you with generating or manipulating PDF files, such as PDFKit, jsPDF, or pdf-lib. You can explore these libraries to find the one that best fits your requirements for editing and saving PDF files.</p> <p>Please note that implementing editing and saving functionality for PDF files can be complex, and it may require a good understanding of PDF file structure and JavaScript programming.</p>	Moderately satisfied	Very satisfied
Q10	<p>Hi everyone,</p> <p>I'm running into a bit of an issue with PDF.js and chrome. I'm not sure I've got enough information for a true github issue, but I wanted to drop it here to see if you might have any insight into the issue.</p> <p>In the image above you can see a sample of the problem: On my coworkers machine PDF's are often rendered like this, with random bits and pieces of the text characters missing. This seem to happen on his machine and at least one other but not on mine, and it only happens in Chrome and in Edge but not when using Firefox.</p> <p>It seems to happen to many, but not all, the PDF's, and it isn't consistent: a page reload can cause the anomalies to end up entirely somewhere else in the document. I have attached an example PDF that exhibits the problem for my coworker.</p> <p>This goes wrong in our own application and in the slack PDF viewer (which seems to use PDF.js), but in the PDF.js demo page (opening the example PDF) does work correctly. (https://mozilla.github.io/pdf.js/web/viewer.html)</p> <p>I initially blamed my coworkers machine, but as we have now also seen it on another machine it is suspicious. Have you every seen anything like this? I'm pretty stumped.</p> <p>Any insight on this matter is greatly appreciated!</p>	<p>Maybe you should try to test in enabling/disabling hardware acceleration in Chrome</p> <p>I tried myself on Chrome on Windows 11 with enabled/disabled 'Accelerated 2D canvas' and it works well</p>	<p>I'm sorry, but I don't have access to personal experiences or encounters. However, it is possible to encounter rendering issues with PDF.js in different browsers. If you are experiencing this specific issue, it would be best to report it to the PDF.js development team with the necessary details such as the PDF file, browser version, operating system, and any error messages you may have encountered.</p>	Very satisfied	Slightly satisfied

Q11	<p>Hi,</p> <p>I have a very large pdf in size (200mb) and I want to download it faster. How to do that in PDF.js viewer?</p> <ol style="list-style-type: none"> 1. Pdf is linearized 2. Using rangeChunkSize (1024 * 1024) <p>But it is downloading chunk in sequentially but I want them to be downloaded paralleily. Is there any configuration available to control the number of parallel request?</p>	Download it and use an other viewer	No, there is no configuration available in the PDF.js viewer to control the number of parallel requests for downloading chunks of a large PDF file. The viewer automatically handles the downloading and rendering of the PDF file, and the number of parallel requests is determined by the browser's network stack.	Not at all satisfied	Very satisfied
Q12	Is there a pdf.js associated inspector like tool for viewing pdf internals? I recall seeing something like this before	http://brendandahl.github.io/pdf.js.utils/browser/	Yes, there is a PDF Object Browser tool available for inspecting the internal object structure and viewing raw values of a PDF document. You can access the PDF Object Browser using this link: https://brendandahl.github.io/pdf.js.utils/browser .	Very satisfied	Completely satisfied
Q13	I am trying to parse pdf and get headings and paragraphs under those headings. getStructTree gives me the document structure, but I can't figure out how I can get the content for each element, and gettextContent gives me the correct text but it has no style/structure information. Help would be appreciated.	Please see https://github.com/mozilla/pdf.js/blob/fbfacfb828eddd0c458a330f162c5a3f7f5dc948/src/display/api.js#L1112-L1117	<p>To get the content for each element in the parsed PDF document structure, you can use the 'getTextContent()' method. This method returns a promise that resolves to an object containing the text content for each element in the structure.</p> <p>Here is an example of how you can use the 'getTextContent()' method:</p> <pre>'''javascript const loadingTask = getDocument('example.pdf'); loadingTask.promise.then((pdf) => { pdf.getPage(1).then((page) => { page.getTextContent().then((textContent) => { textContent.items.forEach((item) => { console.log(item.str); }); }); }); }); ''')</pre> <p>In this example, 'getDocument()' is a function that loads the PDF document, 'getPage()' retrieves the desired page (in this case, page 1), and 'getTextContent()' retrieves the text content for that page. The 'textContent' object contains an array of 'items', where each item represents an element in the parsed PDF document structure. The 'str' property of each item contains the actual content of the element.</p> <p>You can then iterate over the 'items' array and access the 'str' property to get the content for each element in the structure.</p>	Very satisfied	Not at all satisfied
Q14	<p>Hello,</p> <p>I am using pdfjs-dist. Currently I am running into an error saying that DOMMatrix is undefined.</p> <pre>''' error: An error occurred while executing form/create DOMMatrix is not defined [stack]:ReferenceError: DOMMatrix is not defined'''</pre> <p>I am running this on an npm server, so I looked this up and found this: https://github.com/mozilla/pdfjs-dist/blob/5f07d5a4159bb99eee2f6143d1297f03b45bba58/lib/shared/compatibility.js#L43 .</p> <p>It pulls DOMMatrix from the npm library canvas. I made sure I have canvas installed, but I still run into this issue. What else can I do to resolve this?</p>	I was able to fix this by adding in import { DOMMatrix } from 'canvas' at the very top of my node_modules file for pdfjs-dist in pdfjs.min.js.	<p>It seems that you are encountering an error related to the DOMMatrix class not being defined. This class is imported from the npm library "canvas" in the compatibility.js file of the pdfjs-dist library.</p> <p>To resolve this issue, you can try the following steps:</p> <ol style="list-style-type: none"> 1. Make sure you have the "canvas" package installed in your project. You can install it by running the following command in your project directory: <pre>npm install canvas</pre> 2. Check if you have the correct version of the "canvas" package installed. The DOMMatrix class was introduced in newer versions of the "canvas" package. You can verify the installed version by checking the package.json file or running the following command: <pre>npm list canvas</pre> 3. If you have an older version of the "canvas" package installed, try updating it to the latest version by running the following command: <pre>npm update canvas</pre> 4. Restart your npm server to ensure that the changes take effect. <p>If the issue still persists after following these steps, it's possible that there might be a compatibility issue between the pdfjs-dist library and the canvas package. In that case, you may need to investigate further or consider using a different version of the pdfjs-dist library.</p>	Slightly satisfied	Moderately satisfied