To add an image upload toolbar for uploading images directly in addition to the existing image toolbar in Quill editor, you can customize the toolbar and add a custom button for image upload functionality. Additionally, you can customize the image toolbar to include properties like width, height, alt text, etc., when inserting an image.

Here's how you can achieve this:

1. **Add Image Upload Button to Toolbar**: You can add a custom button to the toolbar for uploading images. When this button is clicked, you can trigger a file input dialog to upload the image and then insert it into the editor.
2. **Customize Image Toolbar**: You can customize the image toolbar by creating a custom module that extends the default image module of Quill. This custom module can provide options for setting properties like width, height, alt text, etc., when inserting an image.

Below is an example implementation to achieve the above functionalities:

<!-- Add a custom button for image upload -->

<button class="ql-image-upload">Upload Image</button>

<!-- Customize the image toolbar with width, height, alt text fields -->

<div id="image-toolbar">

Width: <input type="text" id="image-width">

Height: <input type="text" id="image-height">

Alt Text: <input type="text" id="image-alt">

</div>

// Initialize Quill editor with custom modules

const quill = new Quill('#editor', {

modules: {

toolbar: {

container: [

[{ 'header': [1, 2, false] }],

['bold', 'italic', 'underline'],

['image', 'link'],

[{ 'align': [] }],

['image-upload'] // Custom image upload button

]

},

imageResize: {

modules: ['Resize', 'DisplaySize']

}

},

theme: 'snow'

});

// Add functionality for custom image upload button

const imageUploadButton = document.querySelector('.ql-image-upload');

imageUploadButton.addEventListener('click', () => {

const input = document.createElement('input');

input.setAttribute('type', 'file');

input.setAttribute('accept', 'image/\*');

input.onchange = () => {

const file = input.files[0];

// Handle file upload and insert image into editor

};

input.click();

});

// Customize image toolbar with width, height, alt text fields

quill.getModule('toolbar').addHandler('image', () => {

const toolbar = document.querySelector('#image-toolbar');

toolbar.style.display = 'block';

// Add functionality to set width, height, alt text of the image

});

In this implementation:

* The custom image upload button triggers a file input dialog for uploading images.
* The image toolbar is customized to include fields for setting width, height, and alt text for the image.
* You can handle the file upload and image insertion logic in the respective event handlers.

You can further enhance this functionality by adding validation for image properties, error handling, and styling as per your requirements. Feel free to modify and expand upon this example to suit your specific needs.

Ask GPT-4 Turbo

please fills two parts (// Handle file upload and insert image into editor) and ( // Add functionality to set width, height, alt text of the image) in your code suggestions with some sample codes. for example with Axios send the uploaded image to url '/api/upload' and then insert in to the editor content, etc. or set image width, height based on user input choice.

**GPT-3.5 Turbo**

Certainly! Here are the sample codes to handle file upload, insert the image into the editor, and set the width, height, and alt text of the image based on user input:

// Handle file upload and insert image into editor

input.onchange = () => {

const file = input.files[0];

const formData = new FormData();

formData.append('image', file);

axios.post('/api/upload', formData, {

headers: {

'Content-Type': 'multipart/form-data'

}

})

.then(response => {

const imageUrl = response.data.imageUrl;

const range = quill.getSelection();

quill.insertEmbed(range.index, 'image', imageUrl);

})

.catch(error => {

console.error('Error uploading image: ' + error.response);

});

};

// Add functionality to set width, height, alt text of the image

quill.getModule('toolbar').addHandler('image', () => {

const toolbar = document.querySelector('#image-toolbar');

toolbar.style.display = 'block';

const setAttributesButton = document.createElement('button');

setAttributesButton.innerHTML = 'Set Attributes';

setAttributesButton.addEventListener('click', () => {

const width = document.querySelector('#image-width').value;

const height = document.querySelector('#image-height').value;

const alt = document.querySelector('#image-alt').value;

const selectedImage = quill.root.querySelector('.ql-image');

if (selectedImage) {

selectedImage.setAttribute('width', width);

selectedImage.setAttribute('height', height);

selectedImage.setAttribute('alt', alt);

}

toolbar.style.display = 'none';

});

toolbar.appendChild(setAttributesButton);

});

In the above code snippets:

* When a user selects an image to upload, the file is sent to the server endpoint /api/upload using Axios. Upon successful upload, the image URL is received and inserted into the Quill editor at the current cursor position.
* The image toolbar now includes input fields for setting the width, height, and alt text of the selected image. When the user clicks on "Set Attributes", the selected image's attributes are updated accordingly.

You can further customize these functionalities based on your specific requirements and server-side image upload handling.