کلاس FileReader برای خواندن محتوای فایل

کلاس FileReader

کلاس `FileReader` در نسخه‌های جدید Appsan برای مدیریت عملیات خواندن فایل‌ها به صورت متنی، باینری و Data URL استفاده می‌شود. این کلاس برای خواندن محتوای یک Blob و فایل‌ها در محیط جاوااسکریپت طراحی شده است. در اینجا به معرفی متدهای این کلاس و نحوه استفاده از آن‌ها پرداخته می‌شود.

1. متدها و ویژگی‌ها

1.1 `readAsText`

این متد برای خواندن محتوای یک Blob یا فایل به صورت متن استفاده می‌شود.

- **ورودی‌ها**:
  - `blob` (نوع: `JSObject`): شیء Blob که می‌خواهید محتوای آن را به صورت متن بخوانید.
  - `encoding` (اختیاری، نوع: `String`): نوع کدگذاری که به صورت پیش‌فرض "utf-8" است.
- **خروجی**: این متد خروجی خاصی ندارد اما پس از خواندن موفقیت‌آمیز، مقدار `result` در شیء `FileReader` به متن خوانده شده تنظیم می‌شود. همچنین اگر تابع `onload` تعریف شده باشد، پس از تکمیل خواندن، این تابع فراخوانی می‌شود.

1.2 `readAsArrayBuffer`

این متد برای خواندن محتوای یک Blob یا فایل به صورت باینری (آرایه‌ای از بایت‌ها) استفاده می‌شود.

- **ورودی‌ها**:
  - `blob` (نوع: `JSObject`): شیء Blob که می‌خواهید محتوای آن را به صورت آرایه بایت بخوانید.
- **خروجی**: مانند `readAsText`، این متد خروجی خاصی ندارد اما مقدار `result` در شیء `FileReader` به آرایه بایت‌های خوانده شده تنظیم می‌شود. تابع `onload` نیز در صورت تعریف، پس از تکمیل خواندن فراخوانی می‌شود.

1.3 `readAsDataURL`

این متد برای خواندن محتوای یک Blob یا فایل به صورت Data URL استفاده می‌شود. این متد در نسخه‌های بعدی Appsan ارائه خواهد شد.

- **ورودی‌ها**:
  - `blob` (نوع: `JSObject`): شیء Blob که می‌خواهید محتوای آن را به صورت Data URL بخوانید.
  - `encoding` (اختیاری، نوع: `String`): نوع کدگذاری که به صورت پیش‌فرض "utf-8" است.
- **خروجی**: این متد نیز خروجی خاصی ندارد اما مقدار `result` در شیء `FileReader` به Data URL تنظیم می‌شود و تابع `onload` پس از تکمیل عملیات فراخوانی می‌شود.

2. نمونه کدها

در زیر چند نمونه کد از استفاده کلاس `FileReader` آورده شده است.

```
<script id="script-upload">
    var selectedFile;

    function readAsText() {
        if (!selectedFile) {
            Appsan.runAction("snackbar/'No file selected.'");
            return;
        }

        const reader = new FileReader();
        reader.onload = function() {
            console.log(reader.result);
            Appsan.setProperty('result', 'text', reader.result);
        }
        reader.readAsText(selectedFile);
    }

    function readAsArrayBuffer() {
        if (!selectedFile) {
            Appsan.runAction("snackbar/'No file selected.'");
            return;
        }

        const reader = new FileReader();
        reader.onload = function() {
            console.log(reader.result);
            const ab = reader.result;
```

```
            const uint8Array = new Uint8Array(ab);

            const byteString = Array.from(uint8Array).map(byte =>
byte.toString(16).padStart(2, '0')).join(' ');

            console.log(byteString);
            Appsan.setProperty('result', 'text', byteString);
        }
        reader.readAsArrayBuffer(selectedFile);
    }

    function readAsDataURL() {
        if (!selectedFile) {
            Appsan.runAction("snackbar/'No file selected.'");
            return;
        }

        const reader = new FileReader();
        reader.onload = function() {
            console.log(reader.result);
            Appsan.setProperty('result', 'text', reader.result);
        }
        reader.readAsDataURL(selectedFile);
    }
</script>
```

3. نکات مهم

- در نسخه‌های آینده، متد `readAsDataURL` به طور کامل در دسترس خواهد بود.
- توجه داشته باشید که متدها در `FileReader` به صورت ناهمگام (asynchronous) اجرا می‌شوند و نتیجه عملیات پس از اتمام در ویژگی `result` ذخیره می‌شود.

Revision #2
Created 20 August 2024 13:56:17 by a.dadgari
Updated 20 August 2024 13:58:17 by a.dadgari