Projektuppgift

Introduktion till programmering med JavaScript

Sveriges Radio API

Elias Eriksson

Mittuniversitetet
MID SWEDEN UNIVERSITY

Campus Härnösand Universitetsbacken 1, SE-871 88. **Campus Sundsvall** Holmgatan 10, SE-851 70 Sundsvall.
Campus Östersund Kunskapens väg 8, SE-831 25 Östersund.
Phone: +46 (0)771 97 50 00, Fax: +46 (0)771 97 50 01.

MITTUNIVERSITETET
Avdelningen för informationssystem och -teknologi

Författare: Elias Eriksson, eler2006@student.miun.se
Utbildningsprogram: Webbutveckling, 120 hp
Huvudområde: Datateknik
Termin, år: HT, 2020

Sammanfattning

I detta projekt byggs en webbplats som visar kanaler, dess tablåer och har möjlighet att spela ljudströmmar från Sveriges Radio. Detta utförs genom att använda SRs API för att hämta det innehållet och sedan lägga upp innehållet på webbplatsen genom modifiering utav DOM.

Innehållsförteckning

Terminologi

Akronymer/Förkortningar

HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
HTTP	HyperText Transfer Protocol
DOM	Document Object Model
JSON	JavaScript Object Notation
AJAX	Asynchronous JavaScript and XML
API	Application Programming

1 Introduktion

1.1 Bakgrund och problemmotivering

I detta projekt används SRs (Sveriges Radios) API för att bygga en interaktiv webbplats. Webbplatsen har möjlighet att visa ett visst antal kanaler från SR samt dagens kanalprogram fram till midnatt. Det kommer även att finnas möjlighet att spela upp en vald kanals ljudström via en spelare.

Detta för att påvisa kunskaper att kunna hantera funktionaliteter i JavaScript såsom: webb förfrågningar till externa resurser, hantering utav JSON data, DOM manipulering och allmänt kontroll flöde utav JavaScript.

1.2 Detaljerad problemformulering

Allt innehåll som ska finnas på webbplatsen är innehåll från en extern resurs så allt innehåll måste först hämtas till användaren. Innehållet som hämtas kommer att hämtas som JSON format. JSON datan måste sedan gås igenom och placeras ut på webbplatsen enligt specifikation nedan.

Ett antal kanaler ska visas i vänsterspalten på given webbplats som länkar. Antalet kanaler kan styras med hjälp av ett inmatningsfält under vänsterspalten. Vid svävning av muspekaren över länkarna ska en beskrivning utav radiokanalen visas. Vid klick på länken ska radiokanalens dags tablå visas i mitten utav webbplatsen. Varje program i tablån ska ha en rubrik, sändningstid, information om programmet samt en under rubrik om programmet har en. Utöver tablån finns det även möjlighet att välja och spela radiokanal i sidhuvudet.

2 Teori

I detta projekt används SRs webb API (Application Programming Interface) för att kunna hämta information om SRs radiokanaler, dess tablåer och för att kunna spela upp kanalens ljudström. API är ett sätt för en applikation att exponera funktionalitet för andra programmerare att använda i sina applikationer [1]. SRs API exponerar information om kanalerna som de erbjuder och kan hämtas genom webb anrop [2].

Webb anrop kan göras i JavaScript med JavaScripts inbyggda XMLHttpRequests [3]. Genom att använda XMLHttpRequests görs ett webb anrop till SRs webbserver via bas URLen http://api.sr.se/api/v2/ som kan förlängas med en utav de metoder beskrivna i SRs dokumentation och extra information via SRs generella parametrar [2].

Svaret från SR hämtas i JSON format som är ett textbaserat datalagringsformat som bygger på strukturen av JavaScript objekt [4].

När innehållet i svaret ska skrivas ut på webbplatsen används JavaScript för att manipulera DOM. DOM är en trädstruktur som representerar hela HTML dokumentet och en ändring i DOM ändrar HTML strukturen och därmed webbplatsen [5].

3 Metod

Allt innehåll kommer att hämtas från SRs API med hjälp utav JavaScripts XMLHttpRequest och svaren kommer att analyseras med JavaScripts JSON analyserare för att skapa JavaScript objekt. Detta kommer att utföras när sidan först laddas och när användaren ändrar antalet kanaler som ska synas i vänsterspalten. Samt när användaren klickar på en utav kanalerna i vänsterspalten.

Nya HTML element kommer att skapas med JavaScripts createElement metod och kommer populeras med innehåll från svaret från SRs API och placeras ut som ett barn element i det HTML element med det ID som innehållet ska läggas till i.

4 Konstruktion

När sidan först laddas hämtas inställningar för hur många kanaler som ska hämtas och volymen på spelaren från localstorage till en inställningsobjekt från klassen `Settings`. Det definieras även tre olika eventlyssnare som triggas när: spelarens spelar knapp blir klickad, antalet kanaler som ska visas ändras och sidladdningen är klar.

4.1 Sidladdning

När sidan har laddats klart kommer först inställningen för hur många kanaler som ska visas att sättas till inmatningsfältet med id `numrows`. Därefter kommer https://api.sr.se/api/v2/channels?format=json&pagination=false att efterfrågas med en XMLHttpRequest för att hämta alla kanaler SR erbjuder i JSON format. Svaret kommer att analyseras med JavaScripts JSON analyserare för att skapa ett JavaScript objekt. Utifrån det nyskapade objektet kommer alla kanaler som finns i det att läggas till i listan utav kanaler som kan spelas upp. Men endast de första kanalerna upp till mängden kanaler i inställningarna kommer att läggas till på vänsterspalten [Figur 1]. Kanalerna som läggs till i menyn att spelas upp kommer bli representerade av följande HTML struktur:

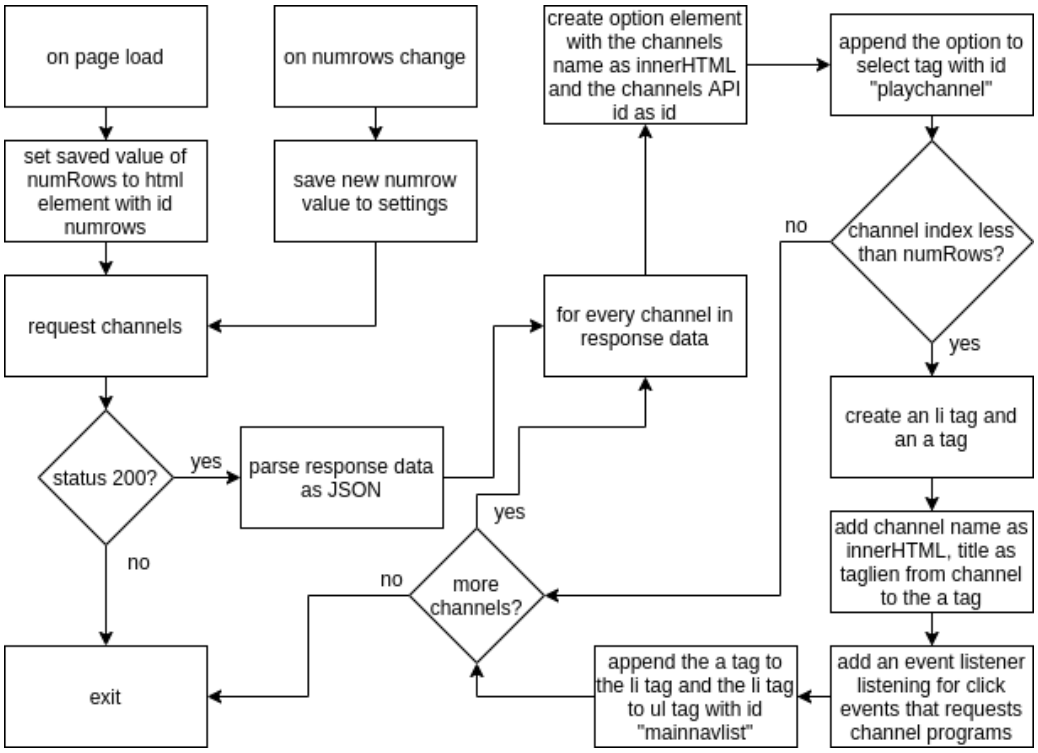
<option id="id">Kanal namn</option>

där värdet på option taggens id attributet är kanalens id som läses ut från svaret från SRs API och inre HTML är svarets kanal namn. Denna struktur skapas med JavaScripts `createElement` metod och det returnerade elementet modifieras med dess egenskaper `setAttribute` och `innerHTML`. Denna struktur läggs sedan till i webbplatsens element med id `playchannel` som hämtas med JavaScript `getElementById` och läggs till med det returnerade elementets egenskap `appendChild`.[Figur 1]

Kanalerna som läggs till i vänsterspalten kommer att representeras av följande HTML struktur:

Kanal namn

där värdet på a taggens id attribut är kanalens id som läses ut från svaret från SRs API, title läses ut från svarets kanal tagline och inre HTML är svarets kanal namn. Denna struktur skapas med JavaScripts `createElement` metod och det returnerade elementet modifieras med dess egenskaper `setAttribute` och `innerHTML`. En eventlyssnare som lyssnar efter klick events läggs även till på a taggen för att hämta tablåer för kanalen när den blir klickad på som beskrivs i avsnitt 4.4. Denna struktur läggs sedan till i webbplatsens element med id `mainnavlist` som hämtas med JavaScripts `getElementById` och läggs till med det returnerade elementets egenskap `appendChild` [Figur 1].



Figur 1: Flödesschema som beskriver programmets gång när användaren först hämtar sidan samt när användaren ändrar på antalet kanaler som ska visas i vänsterspalten.

4.2 Ändring av antal kanaler som ska visas

När antalet kanaler har ändras på grund av att användaren har interagerat med inmatningsfältet med id `numrows` kommer det nya värdet att sparas till inställningsobjektet och exakt samma procedur som beskrivits i avsnitt 4.1 kommer utföras med undantaget att inställningarna inte sparas [Figur 1].

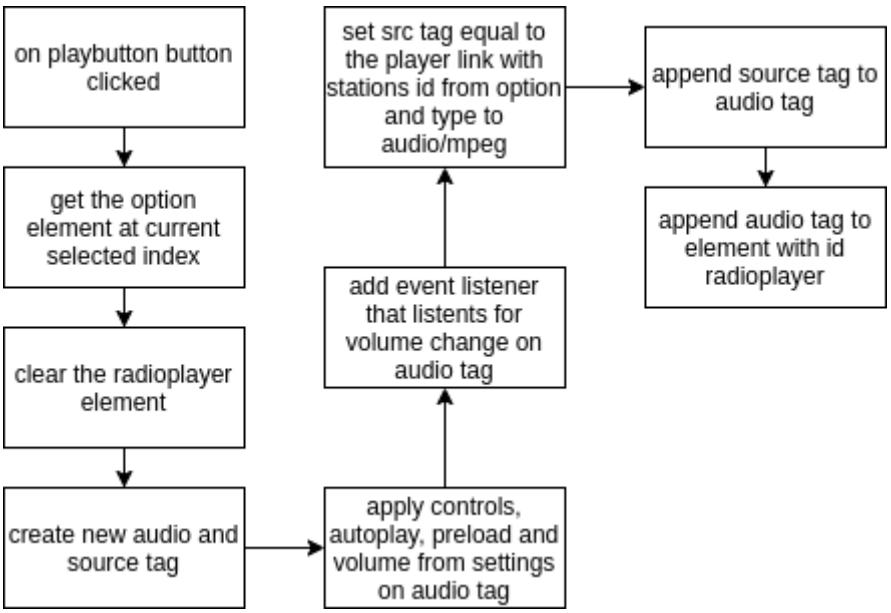
4.3 Spelning av kanal

När användaren trycker på knappen med id `playbutton` kommer innehållet i elementet med id `radioplayer` först att raderas [Figur 2]. Sedan kommer en ljudspelare att skapas med följande HTML struktur:

```
<audio controls="" autoplay="true" preload="auto">
        <source src="https://sverigesradio.se/topsy/direkt/id.mp3"
                type="audio/mpeg">
</audio>
```

där id i urlen är kanalens id som är det valda elementet i `playchannel`s id. Denna struktur skapas med JavaScripts `createElement` metod för att skapa en audio tagg och en source tagg. Audio taggen modifieras sedan med dess egenskaper `setAttribute` och `addEventListener` för att lägga till attributen `controls=""`, `autoplay=”true”` och `preload=”auto”`. En eventlyssnare läggs även till som lyssnar efter att användaren ändrar volymen på ljudspelaren var funktion beskrivs

i avsnitt 4.5 samt att spelarens volym sätts till volymen sparad i inställningarna. Source taggen modifieras med dess egenskap `setAttribute` för att lägga till attributen `src="https://sverigesradio.se/topsy/direkt/id.mp3"` där id är kanalens id samt `type="audio/mpeg"`. Source taggen läggs sedan in i audio taggen med audio taggens egenskap `appendChild`. Audio taggen läggs sedan till i elementet med id `radioplayer` med dess egenskap `appendChild` [Figur 2].



Figur 2: Flödesschema som visar vad som händer när användaren börjar spela en kanal.

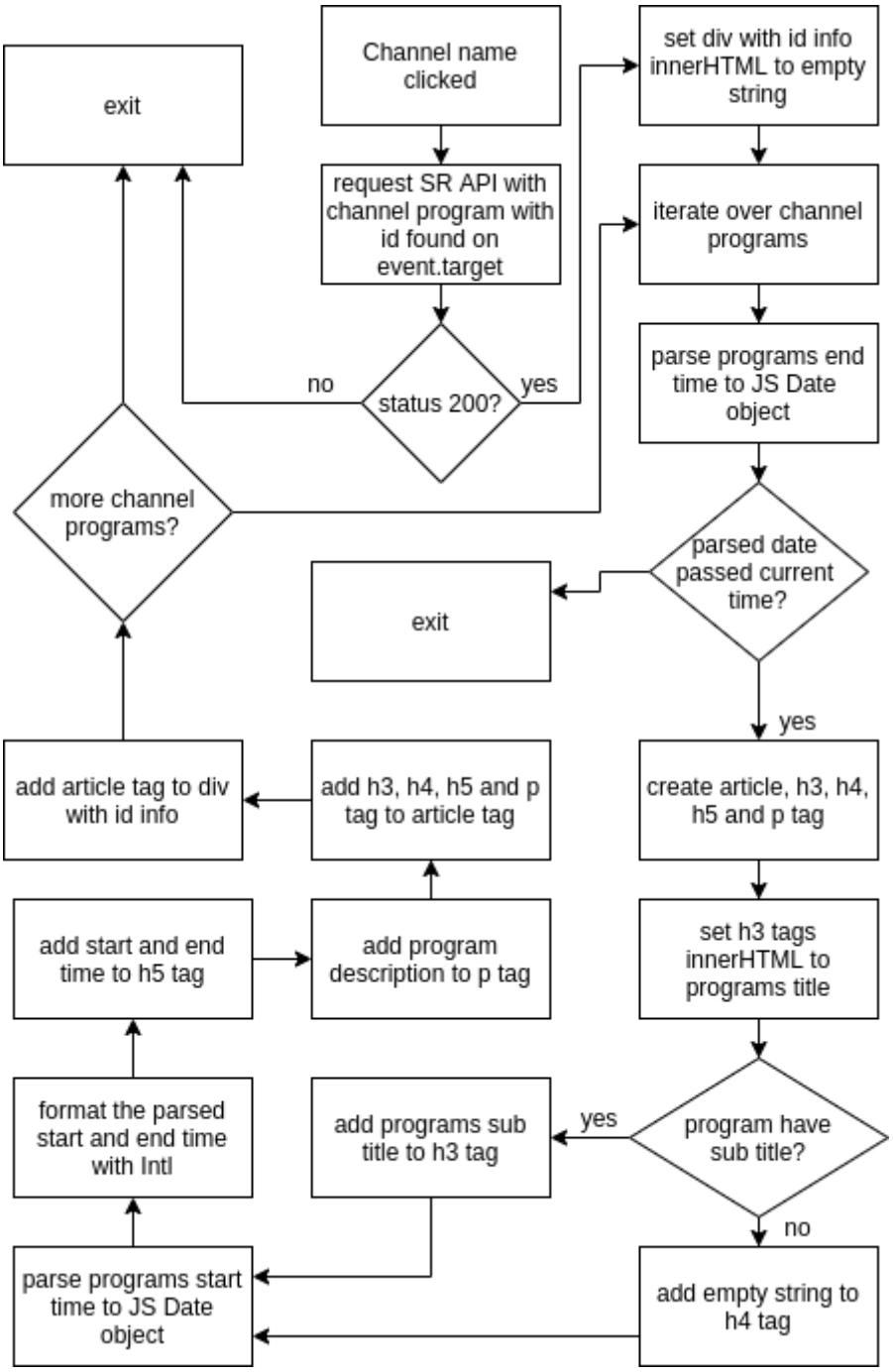
4.4 Hämtning utav tablå

När en a tagg i vänsterspalten klickas på kommer innehållet i elementet med id `info` först att raderas. Sedan kommer https://api.sr.se/api/v2/scheduledepisodes?channelid=id&format=json efterfrågas med en XMLHttpRequest där id i URLen är kanalens id som finns i det klickade elementets id attribut [Figur 3]. Svaret från efterfrågan innehåller en tablå med program och alla program kommer representeras med följande HTML struktur:

```
<article>
        <h3>Program rubrik</h3>
        <h4>Under rubrik</h4>
        <h5>start och slut tid</h5>
        <p>Program beskrivning</p>
</article>
```

där h3 taggens inre HTML är programmets title som läses ut från svaret från SRs API, h4 elementets inre HTML är svarets program subtitle om det finns, h5 elementets inre HTML är svarets program starttime och endtime formaterat till svenskt tidformat och p elementets inre HTML är svarets program description. Denna struktur skapas med JavaScripts `createElement` metod och de skapade h3, h4, h5 och p elementen läggs till i article elementet via article elementets

egenskap `appendChild`. article elementet läggs sedan till i elementet med id `info` genom elementets egenskap `appendChild` [Figur 3].



Figur 3: Flödesschema som beskriver vad som händer när användaren klickar på en länk i vänsterspalten.

4.5 Volym ändring i spelaren

När spelaren känner av en volymändring kommer den nya volymen att sparas till inställnings objektet.

5 Resultat

Allt innehåll till webbplatsen har hämtats från SRs API med hjälp av JavaScripts XMLHttpRequest. Svaren analyserades med JavaScripts JSON analyserare och placerades ut på webbplatsen genom att manipulera DOM med JavaScript.

Innehållet ligger på rätt plats på webbplatsen och följer strukturerna beskrivna i avsnitt 4.

6 Slutsatser

Den byggda webbplatsen uppfyller det krav som fanns och fungerar som den ska. Om webbplatsen skulle byggas om skulle det vara bättre att bygga den med fetch API och async/await för att på så sätt kunna ha en intern datastruktur med all information som hämtas som kan agera som en cache då det görs väldigt många anrop till SR användaren ändrar mängden kanaler i sidan hela tiden. Informationen hålls uppdaterad då men webbplatsen är inte tidskänslig och skulle spara resurser på SRs sida.

Källförteckning

[1] W3schools, ”web APIs – introduction” https://www.w3schools.com/js/js_api_intro.asp Hämtad 2020-10-22.

[2] Sveriges Radio ”Sveriges Radios öppna API – (version 2)” https://sverigesradio.se/api/documentation/v2/ Hämtad 2020-10-22.

[3] MDN ”XMLHttpRequest” https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest Hämtad 2020-10-22.

[4] JSON org ”Introducing JSON” https://www.json.org/json-en.html Hämtad 2020-10-22.

[5] MDN ”Document Object Model (DOM)” https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model Hämtad 2020-10-22.