[Samsung PRISM] Mid Review Report

OD153MS - Read Aloud

Team

1. College Professor(s):
2. Students:
   1. Aakash P
   2. Utkarsh Agarwal
   3. Vridhi Kamath
   4. Mohak Khowal
3. Department: Computer Science and Engineering

Date: 5 Aug 2019

Text manipulation, Bixby Capsule Development | Read Aloud

SAMSUNG

Problem Statement

- Modes of media consumption keeps changing. In many situations (e.g. working out, cooking) reading an article or a book is impractical, however listening to the same information is feasible.

- Existing solutions for listening to a body of text often require active user interaction and have limited features for navigating the text.

- Proposed solution – Read Aloud:
- Read Aloud enables a user to transition from reading a webpage or a local document to listening to it.
- It does so by leveraging voice assistant and on-device TTS support
- Using the voice assistant, the listener can

    - Utter voice commands to navigate through the webpage or the document.
    - request to repeat segments of text, or to pause or stop listening.
    - Ask for the text summarization (similar to skimming through the document)

Utkarsh Ray

Naga Koteswara

Additional Documentation:

- https://bixbydevelopers.com/
- https://www.nltk.org/
- https://developer.android.com/reference/android/speech/tts/TextToSpeech
- https://developer.android.com/reference/android/graphics/pdf/PdfRenderer

Work-let expected duration – 6 months

2 Members

Expectations

- Solution with Android document reader app and a Bixby capsule which on user's request performs the following actions:
    - Obtain webpage URL/document filename (PDF, EPUB) from active application or directly given by user
    - Extract readable text from the requested parts of the webpage/document (e.g. specific section, chapter, page, entirety)
    - Read aloud the resulting text to user
    - Navigate the text based on user's voice interaction (e.g. previous line, skip the section, repeat last paragraph)

- Optional/Stretch goals
    - Summarise the text given specific constraints (e.g. summary in N sentences/seconds)
    - Optimise any required ML components to run on-device

Training / Pre-requisites

- Android app development
- Good knowledge of text manipulation
- Optional/stretch goals:
    - Familiarity with ML applications in text domain

VOICE Assistant

Student Learning

- Bixby capsule development
- Text manipulation
- TTS solutions on Android

Kick Off < 1st Month >

- Understanding relationships between required modules
- Getting acquainted with the required APIs
- Sample solution to use TTS functionality to read aloud text

Milestone 1 < 2nd Month >

- Obtain text from a given URL/local document
- Extract specific portions of the text
- Obtain URL/document from active application

Milestone 2 < 4th Month >

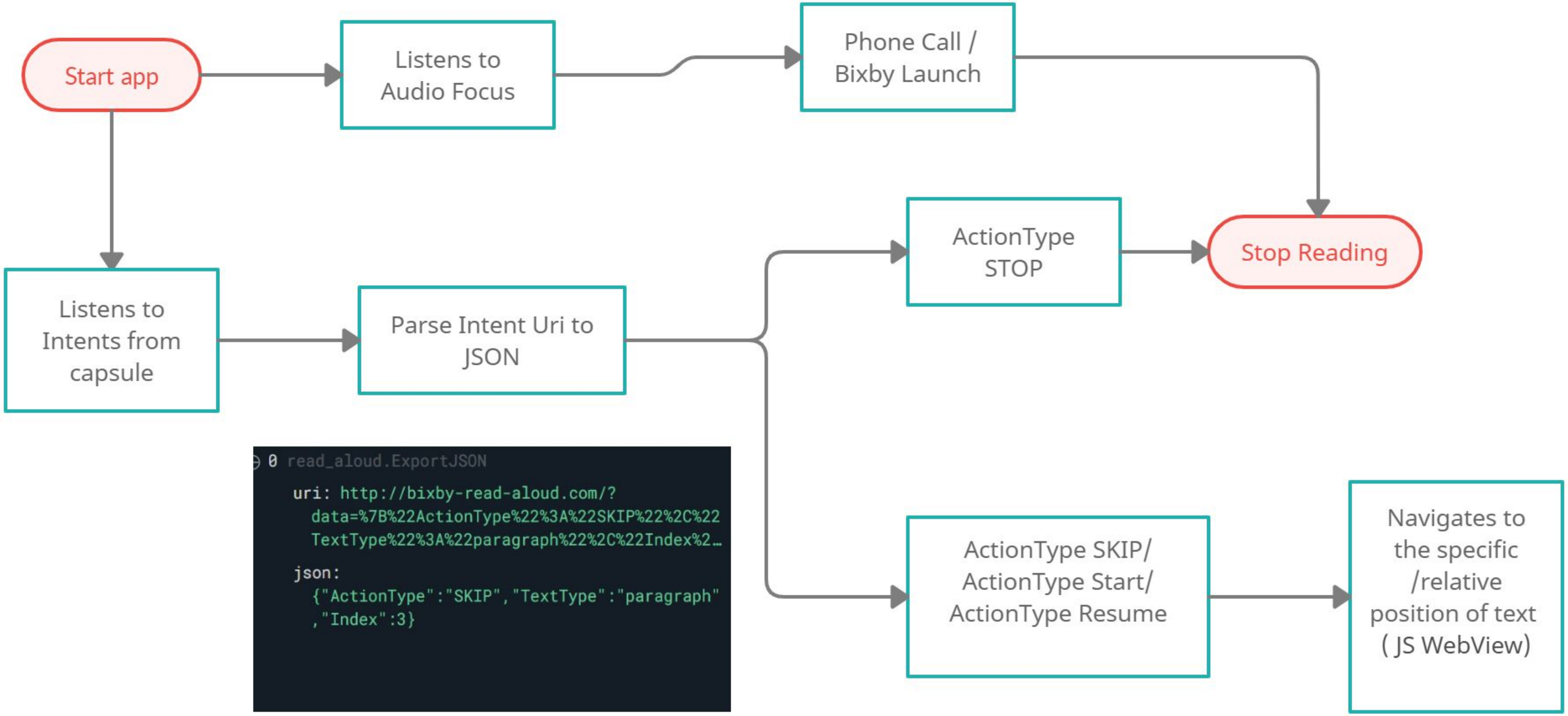
- Navigate the text as per user interactions
- Summarise the text given time/length constraints

Closure < 6th Month >

- Optimisation for network payload/latency
- Optimise any ML solutions for on-device application

Proposed Approach / Solution

- **Concept Diagram** :
  (Clear detailed schematic / block diagram / flow chart depicting the proposed concept / solution)

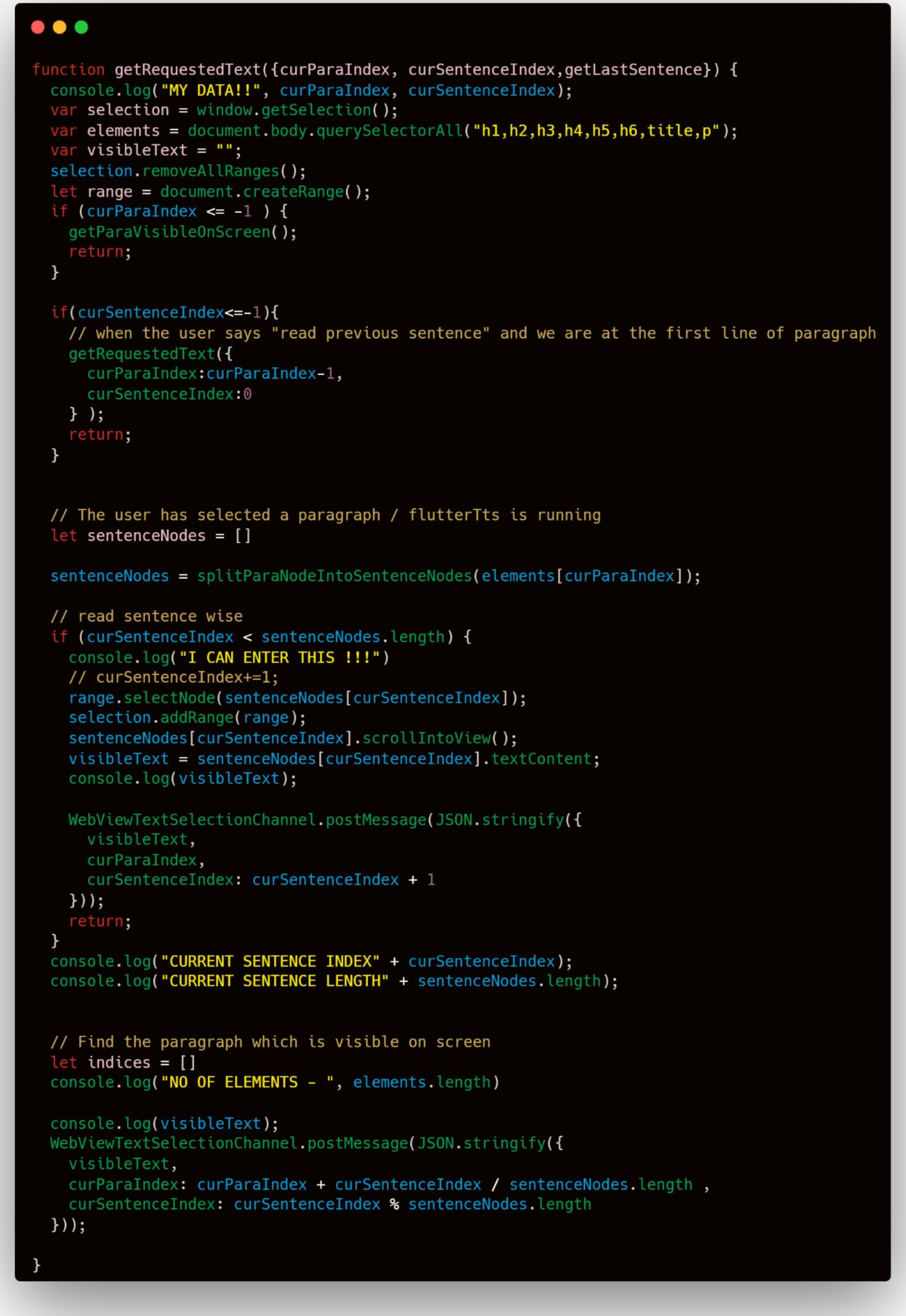

Start app
Listens to Audio Focus
Phone Call / Bixby Launch
Stop Reading
Listens to Intents from capsule
Parse Intent Uri to JSON
ActionType STOP
ActionType SKIP/ ActionType Start/ ActionType Resume
Navigates to the specific /relative position of text (JS WebView)

```
0 read_aloud.ExportJSON
  uri: http://bixby-read-aloud.com/?
    data=%7B%22ActionType%22%3A%22SKIP%22%2C%22
    TextType%22%3A%22paragraph%22%2C%22Index%2…
  json:
    {"ActionType":"SKIP","TextType":"paragraph"
    ,"Index":3}
```

Goal : ConvertRelativeActionToJson

Utterance	**ActionType**	**Text type**	**NavigationInfo**	**CurParaIndex**	**CurSentenceIndex**
Start Reading	START	PARAGRAPH	THIS	0	0
Read next paragraph	RESUME	PARAGRAPH	NEXT	+1	0
Repeat previous paragraph	RESUME	PARAGRAPH	PREVIOUS	-1	0
Repeat last sentence	RESUME	SENTENCE	PREVIOUS		-1
Go to next paragraph	RESUME	PARAGRAPH	NEXT	+1	0
Skip 3 sentences	SKIP	SENTENCE	Not applicable		(+3+1)
Skip 3 paragraphs	SKIP	PARAGRAPH	Not applicable	(+3+1)	
Stop Reading	STOP	Not applicable	Not applicable	SET TO NULL	Not applicable
Repeat this paragraph	RESUME	PARAGRAPH	THIS	No change (+ 0)	0

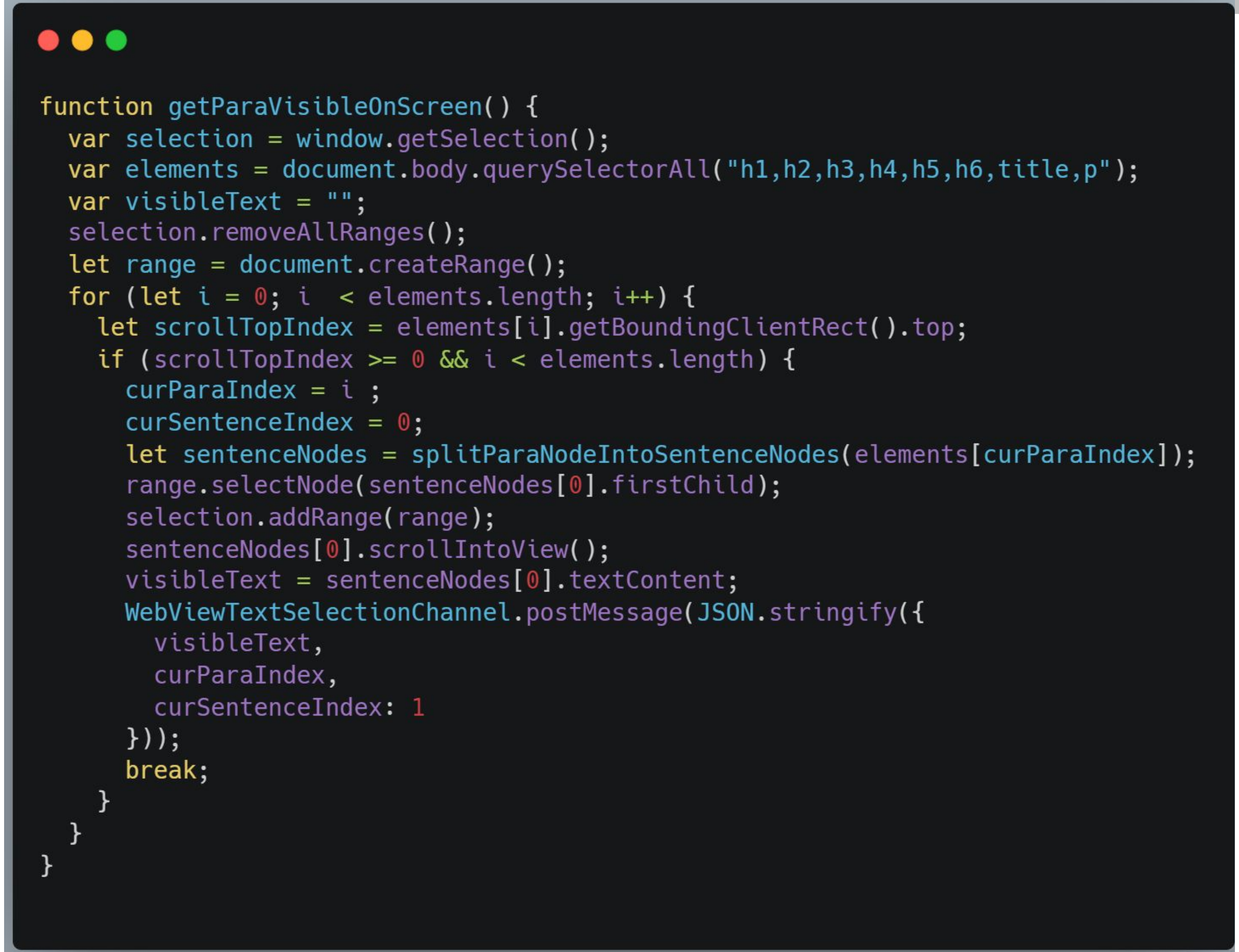
Text Manipulation- Extracting Paragraph



```
function getRequestedText({curParaIndex, curSentenceIndex,getLastSentence}) {
  console.log("MY DATA!!", curParaIndex, curSentenceIndex);
  var selection = window.getSelection();
  var elements = document.body.querySelectorAll("h1,h2,h3,h4,h5,h6,title,p");
  var visibleText = "";
  selection.removeAllRanges();
  let range = document.createRange();
  if (curParaIndex <= -1 ) {
    getParaVisibleOnScreen();
    return;
  }

  if(curSentenceIndex<=-1){
    // when the user says "read previous sentence" and we are at the first line of paragraph
    getRequestedText({
      curParaIndex:curParaIndex-1,
      curSentenceIndex:0
    } );
    return;
  }


  // The user has selected a paragraph / flutterTts is running
  let sentenceNodes = []

  sentenceNodes = splitParaNodeIntoSentenceNodes(elements[curParaIndex]);

  // read sentence wise
  if (curSentenceIndex < sentenceNodes.length) {
    console.log("I CAN ENTER THIS !!!")
    // curSentenceIndex+=1;
    range.selectNode(sentenceNodes[curSentenceIndex]);
    selection.addRange(range);
    sentenceNodes[curSentenceIndex].scrollIntoView();
    visibleText = sentenceNodes[curSentenceIndex].textContent;
    console.log(visibleText);

    WebViewTextSelectionChannel.postMessage(JSON.stringify({
      visibleText,
      curParaIndex,
      curSentenceIndex: curSentenceIndex + 1
    }));
    return;
  }
  console.log("CURRENT SENTENCE INDEX" + curSentenceIndex);
  console.log("CURRENT SENTENCE LENGTH" + sentenceNodes.length);


  // Find the paragraph which is visible on screen
  let indices = []
  console.log("NO OF ELEMENTS - ", elements.length)

  console.log(visibleText);
  WebViewTextSelectionChannel.postMessage(JSON.stringify({
    visibleText,
    curParaIndex: curParaIndex + curSentenceIndex / sentenceNodes.length ,
    curSentenceIndex: curSentenceIndex % sentenceNodes.length
  }));

}
```

We are extracting the paragraph as per curParaIndex, which is updated every time the user specifies the paragraph to be read
Eg: Read Next Paragraph

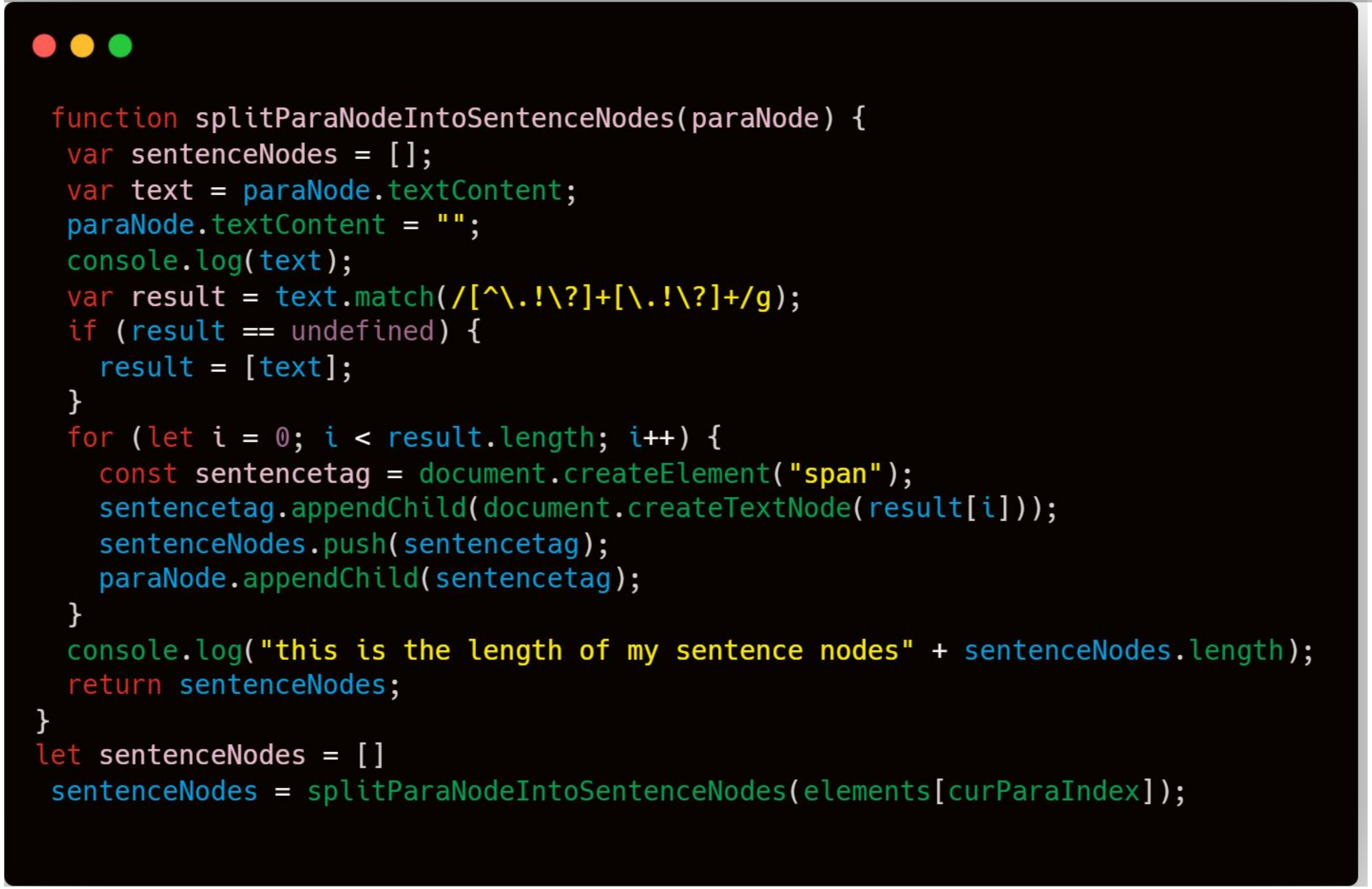
Text Manipulation- Extracting Paragraph



```
function getParaVisibleOnScreen() {
  var selection = window.getSelection();
  var elements = document.body.querySelectorAll("h1,h2,h3,h4,h5,h6,title,p");
  var visibleText = "";
  selection.removeAllRanges();
  let range = document.createRange();
  for (let i = 0; i  < elements.length; i++) {
    let scrollTopIndex = elements[i].getBoundingClientRect().top;
    if (scrollTopIndex >= 0 && i < elements.length) {
      curParaIndex = i ;
      curSentenceIndex = 0;
      let sentenceNodes = splitParaNodeIntoSentenceNodes(elements[curParaIndex]);
      range.selectNode(sentenceNodes[0].firstChild);
      selection.addRange(range);
      sentenceNodes[0].scrollIntoView();
      visibleText = sentenceNodes[0].textContent;
      WebViewTextSelectionChannel.postMessage(JSON.stringify({
        visibleText,
        curParaIndex,
        curSentenceIndex: 1
      }));
      break;
    }
  }
}
```

We are extracting the paragraph as per curParaIndex, which is updated every time the user specifies the paragraph to be read
Eg: Read Next Paragraph

Text Manipulation- Extracting Sentence



```
 function splitParaNodeIntoSentenceNodes(paraNode) {
  var sentenceNodes = [];
  var text = paraNode.textContent;
  paraNode.textContent = "";
  console.log(text);
  var result = text.match(/[^\.!\?]+[\.!\?]+/g);
  if (result == undefined) {
    result = [text];
  }
  for (let i = 0; i < result.length; i++) {
    const sentencetag = document.createElement("span");
    sentencetag.appendChild(document.createTextNode(result[i]));
    sentenceNodes.push(sentencetag);
    paraNode.appendChild(sentencetag);
  }
  console.log("this is the length of my sentence nodes" + sentenceNodes.length);
  return sentenceNodes;
}
let sentenceNodes = []
 sentenceNodes = splitParaNodeIntoSentenceNodes(elements[curParaIndex]);
```

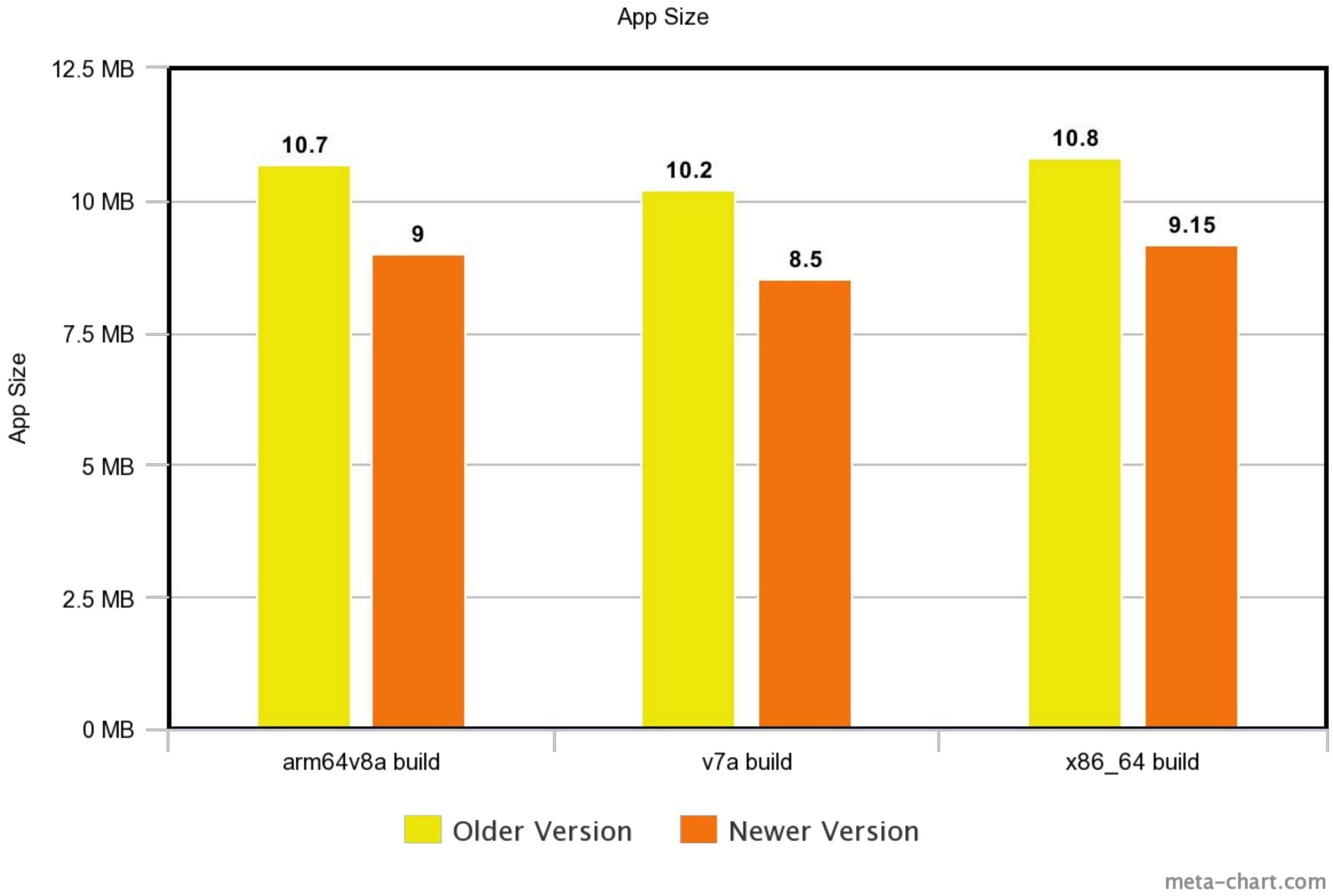
We are extracting sentences from the current paragraph and storing that in an array

Text Manipulation- Extracting pages

Using pdf.js to display the pdf in web view.

Experimental Results / Simulations / Observations

- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)



Reduced App Sizes in different
Application Binary Interfaces (ABI)
After optimization of used storage space.

Further Plan to Complete Project

- **Final Probable Deliverables :**

(Discuss in the form of bullets, what are the next steps to complete the solution, any road blocks / bottlenecks, any support needed from SRIB)

- Complete implementation of pdf navigation
- Testing
- Cleaning Code

- **IP Target / Plan :**

(Any possibility of papers / patentable ideas / innovative aspects that can lead to patentable ideas)

Thank you