PDF.js manual for search function

# **Environment setup**

1. Get the pdf.js in mozilla.
2. Follow the github setup procedure.
3. A black background with white text

   Description automatically generated
4. A screenshot of a computer

   Description automatically generated
5. The edited code file will be viewer.js, viewer.html, viewer.css in build->generic->web. A screenshot of a computer program

   Description automatically generated
6. Server will start when we type “gulp server” in the terminal. 

# **Function or code for search**

Search function in viewer.js : call this function and set the search term in query,

    this.eventBus.dispatch("find", {

      source: this,

      query: matchword,

      caseSensitive: this.caseSensitive.checked,

      entireWord: true,

      highlightAll: true,

      findPrevious: true,

      matchDiacritics: this.matchDiacritics.checked

    });

the terms in query can be like ->

query: "matchword"

For **multiple searches** in pdf.js, we can save it into an array and put it into the query.

let matchwordz = ["a","b"];

query: matchwordz,

An example that make it work (in about line 4875)

  dispatchMatchWord() {

    let matchword1 = "numbers";

    let matchword2 = "example";

    let matchword = [];

    if(matchword1check){

      matchword[0] = matchword1;

    }

    if(matchword2check){

      matchword[1] = matchword2;

    }

    console.log(matchword);

    this.eventBus.dispatch("find", {

      source: this,

      query: matchword,

      caseSensitive: this.caseSensitive.checked,

      entireWord: true,

      highlightAll: true,

      findPrevious: true,

      matchDiacritics: this.matchDiacritics.checked

    });

  }

## Multiple search-term example (whole code added)

### In viewer.js

The default search bar’s function is stored inside the class PDFFindBar (line 4708)

Code added for multiple queries searching:

In (line 4762)

1. this.MatchWord1.addEventListener("click", () => {
2. if(matchword1check){
3. matchword1check = false;
4. }
5. else{
6. matchword1check = true;
7. }
8. this.dispatchMatchWord();
9. });

this.MatchWord2.addEventListener("click", () => {

1. if(matchword2check){
2. matchword2check = false;
3. }
4. else{
5. matchword2check = true;
6. }
7. this.dispatchMatchWord();
8. });

Line 4874

1. dispatchMatchWord() {
2. let matchword1 = "numbers";
3. let matchword2 = "example";
4. let matchword = [];
5. let matchwordz = ["a","b"];
6. if(matchword1check){
7. matchword[0] = matchword1;
8. }
10. if(matchword2check){
11. matchword[1] = matchword2;
12. }
13. console.log(matchword);
14. this.eventBus.dispatch("find", {
15. source: this,
16. query: matchword,
17. caseSensitive: this.caseSensitive.checked,
18. entireWord: true,
19. highlightAll: true,
20. findPrevious: true,
21. matchDiacritics: this.matchDiacritics.checked
22. });
23. }

Code added to activate the button and connect it to the html,

(line 4729)

    this.MatchWord1 = options.MatchWordButton1;

    this.MatchWord2 = options.MatchWordButton2;

(line 13939)

      MatchWordButton1: document.getElementById("MatchWord1"),

      MatchWordButton2: document.getElementById("MatchWord2"),

### In viewer.html

Code for the corresponding button (line 117-118):

A screen shot of a computer code

Description automatically generated

# Language translation

### Normal setup for changing the language of the code:

      const {

        webL10n

      } = document;

      this.l10n.\_ready = new Promise((resolve) => {

        webL10n.setLanguage('en-US', () => {

          resolve(webL10n);

        });

      });

      this.l10n.translate(document.documentElement).then(() => {

        this.eventBus.dispatch("localized", {

          source: this

        });

      });

    });

(P.S: the upper code is just the function to make changes of languages, not the whole code for the button)

(P.S.2: Every time changing the language, make sure it is inside the class/constructor that contains l10n.)

In this case, the function is setup in class PDFFindBar.

### How to choose the language that we want to change:

Inside the setLanguage function, rename the ‘en-US’ as ‘zh-TW’, then it will change from English to Chinese.

      this.l10n.\_ready = new Promise((resolve) => {

        webL10n.setLanguage('en-US', () => {

          resolve(webL10n);

        });

      });

To find out all the language and the corresponding code that they could change to, we can reference from -> pdf.js-master -> l10n.

A screenshot of a computer

Description automatically generated