Global

Members

base64

this Event listener is listening for base64 images

Source: app.js, line 4

color

this file contains the functions to control the drawing on the canvas

Source: canvas.js, line 4

db

class Annotation{ constructor (name, roomId, pixel_pair, message, date) { this.name= name; this.roomId= roomId, this.pixel_pair=pixel_pair; this.canvas = canvas; this.message = message; this.date= date; } }

Source: annotationDatabase.js, line 18

db

using a data model: class Story{ constructor (first_name, family_name, story_title, story_image, story_description, date) { this.first_name= first_name; this.family_name= family_name, this.story_title=story_title; this.story_image= story_image; this.story_description= story_description; this.date= date; } }

Source: database.js, line 18

Methods

(async) addStory() → {Promise.<void>}

if offline, it will only add the story to database, if online, it will also add the story to server.

Source: app.js, line 20

Returns:

Type
Promise.<void>

(async) addStory() → {Promise.<void>}

if offline, it will only add the story to database, if online, it will also add the story to server.

Source: history.js, line 7

Returns:

Type
Promise.<void>

addToStoryList(dataR)

if forceReload is true, using the story data returned by the server, if forceReload is false, using the story data returned by indexDB given the story data, it adds a row of weather forecasts to the story-list div

Parameters:

Name	Type	Description
dataR	object	the data returned by the server, a Story object

Source: app.js, line 176

addToStoryList(dataR)

if forceReload is true, using the story data returned by the server, if forceReload is false, using the story data returned by indexDB given the story data, it adds a row of weather forecasts to the story-list div

Parameters:

Name	Type	Description
dataR		

Source: history.js, line 148

checkLoginData(form) → {boolean}

check login data is not null

Parameters:

Name	Type	Description
form	form	a login form including two string: username and password

Source: users.js, line 6

Returns:

return false if login data is null else return ture

Type
boolean

connectToRoom()

used to connect to a room. It gets the user name and room number from the interface. It connects chat and load story in room and initialize canvas and knowledge graph.

Source: chat.js, line 88

deleteDB(name)

Delete a database.

Parameters:

Name	Type	Description
name		Name of the database.

Source: idb/index.js, line 36

drawImageScaled(img, canvas, ctx)

called when it is required to draw the image on the canvas. We have resized the canvas to the same image size so ti is simpler to draw later

Parameters:

Name	Type	Description
img	HTMLElement	image html element in the html page
canvas		the canvas element obtained by jquery
ctx	HTMLElement	the canvas context

Source: canvas.js, line 168

drawOnCanvas(ctx, canvasWidth, canvasHeight, prevX, prevY, currX, currY, color, thickness)

this is called when we want to display what we (or any other connected via socket.io) draws on the canvas note that as the remote provider can have a different canvas size (e.g. their browser window is larger) we have to know what their canvas size is so to map the coordinates

Parameters:

Name	Type	Description
ctx	HTMLElement	the canvas context
canvasWidth	number	the originating canvas width
canvasHeight	number	the originating canvas height
prevX	number	the starting X coordinate
prevY	number	the starting Y coordinate
currX	number	the ending X coordinate
currY	number	the ending Y coordinate
color	string	color of the line
thickness	number	thickness of the line

Source: canvas.js, line 195

get_date(dataR) → {string}

given the local cached annotation data, it returns the value of the field date

Parameters:

Name	Type	Description
dataR	object	the cached annotation object

Source: annotationDatabase.js, line 259

Returns:

return 'unavailable' if dataR is null else return the value of the field date

Type
string

get_date(dataR) → {string}

given the server data, it returns the value of the field date

Parameters:

Name	Type	Description
dataR	object	the story object returned by the server

Source: database.js, line 165

Returns:

return 'unavailable' if dataR is null else return the date of the story

Type
string

get_family_name(dataR) → {string}

given the server data, it returns the value of the field family_name

Parameters:

Name	Type	Description
dataR	object	the story object returned by the server

Source: database.js, line 139

Returns:

family name of the author

Type
string

get_first_name(dataR) → {string}

given the server data, it returns the value of the field first_name

Parameters:

Name	Type	Description
dataR	object	the story object returned by the server

Source: database.js, line 127

Returns:

first name of the author

Type
string

get_message(dataR) → {string}

given the local cached annotation data, it returns the value of the field message

Parameters:

Name	Type	Description
dataR	object	the cached annotation object

Source: annotationDatabase.js, line 271

Returns:

return 'unavailable' if dataR is null else return the message of the object

Type
string

get_name(dataR) → {string}

given the local cached annotation data, it returns the value of the field name

Parameters:

Name	Type	Description
dataR	object	the cached annotation object

Source: annotationDatabase.js, line 222

Returns:

return 'unavailable' if dataR is null else return the value of the field name

Type
string

get_pixel_pair(dataR) → {array}

given the local cached annotation data, it returns the value of the field pixel_pair

Parameters:

Name	Type	Description
dataR	object	the cached annotation object

Source: annotationDatabase.js, line 246

Returns:

return 'unavailable' if dataR is null else return pixel_pair

Type
array

get_roomId(dataR) → {string}

given the local cached annotation data, it returns the value of the field roomId

Parameters:

Name	Type	Description
dataR	object	the cached annotation object

Source: annotationDatabase.js, line 234

Returns:

return 'unavailable' if dataR is null else return the value of the field roomId

Type
string

get_story_description(dataR) → {string}

given the server data, it returns the value of the field description

Parameters:

Name	Type	Description
dataR	object	the story object returned by the server

Source: database.js, line 189

Returns:

description of the story

Type
string

get_story_image(dataR) → {string}

given the server data, it returns the value of the field image

Parameters:

Name	Type	Description
dataR	object	the story object returned by the server

Source: database.js, line 177

Returns:

image data coded in base64

Type
string

get_story_title(dataR) → {string}

given the server data, it returns the value of the field story_title

Parameters:

Name	Type	Description
dataR	object	the story object returned by the server

Source: database.js, line 151

Returns:

title of the story

Type
string

(async) getAllAnnotation() → {Promise.<*>}

it retrieves all annotations

Source: annotationDatabase.js, line 127

Returns:

Type
Promise.<*>

(async) getCachedData(name, roomId) → {array}

it retrieves the annotation for the specific roomId from the database

Parameters:

Name	Type	Description
name	string	user id in a chat room
roomId	string	id of a chat room

Source: annotationDatabase.js, line 154

Returns:

using Promise to get a list of cached data

Type
array

(async) getKnowledgeGraphFromCachedData(dataR) → {string|array}

<<<<<<< HEAD given the local cached annotation data, it returns the value of the field name

Parameters:

Name	Type	Description
dataR	object	the cached annotation object

Source: annotationDatabase.js, line 191

Returns:

- return 'unavailable' if dataR is null else return the value of the field name ======= it retrieves all knowledge graph from the database

  Type
  string

- using Promise to get a list of cached data

Type
array

getRndColor()

obtain a random color when a knowledge graph result is selected

Source: knowledgeGraph.js, line 9

(async) getStoryFromCachedData(story_title) → {array}

it retrieves the story for a story_title from the database

Parameters:

Name	Type	Description
story_title	string	title of a story

Source: database.js, line 80

Returns:

list of stories

Type
array

hideLoginInterface(room, userId)

it hides the initial form and shows the chat, story and knowledge graph

Parameters:

Name	Type	Description
room		the selected room
userId		the user name

Source: chat.js, line 164

hideOfflineWarning()

hide offline warning

Source: app.js, line 276

init()

called by it initialises the interface and the expected socket messages plus the associated actions

Source: chat.js, line 12

initCanvas(roomNo, name)

it inits the image canvas to draw on. It sets up the events to respond to (click, mouse on, etc.) it is also the place where the data should be sent via socket.io

Parameters:

Name	Type	Description
roomNo	number	the number of chat room
name	string	the id of user

Source: canvas.js, line 12

initChatSocket()

it initialises the socket for /chat

Source: chat.js, line 30

(async) initDatabase()

it inits the database

Source: annotationDatabase.js, line 27

(async) initDatabase()

it inits the database

Source: database.js, line 26

initHistory()

called by the HTML onload showing any cached story data and declaring the service worker

Source: history.js, line 56

initKG(skt)

it inits socket to listen event of selecting a knowledge graph result

Parameters:

Name	Type	Description
skt		socket object, using chat here

Source: knowledgeGraph.js, line 20

initStories()

called by the HTML onload showing any cached story data and declaring the service worker

Source: app.js, line 69

initStory(title, author, img_url, description)

load story to the story container div

Parameters:

Name	Type	Description
title	string	the story title to display
author	string	the story author to display
img_url	string	the URL of story image to display
description	string	the story description to display

Source: chat.js, line 144

loadData(forceReload)

if forceReload is true, load stories from server and store the stories to database if forceReload is false, load stories from database

Parameters:

Name	Type	Description
forceReload	boolean	true if the data is to be loaded from the server

Source: app.js, line 90

loadData(forceReload)

if forceReload is true, load stories from server and store the stories to database if forceReload is false, load stories from database

Parameters:

Name	Type	Description
forceReload		true if the data is to be loaded from the server

Source: history.js, line 72

(async) loadStoryData(story_title, forceReload)

given one story_title it queries the indexDB to get the story information

Parameters:

Name	Type	Description
story_title	string	title of a story
forceReload	boolean	false if the data is to be retrieved from the database

Source: app.js, line 153

(async) loadStoryData(story_title, forceReload)

given one story_title it queries the indexDB to get the story information

Parameters:

Name	Type	Description
story_title		
forceReload		false if the data is to be retrieved from the database

Source: history.js, line 114

onLoadLoginPage()

load message in login page

Source: users.js, line 45

openDB(name, version, callbacks)

Open a database.

Parameters:

Name	Type	Description
name		Name of the database.
version		Schema version.
callbacks		Additional callbacks.

Source: idb/index.js, line 11

putItem(itemId, itemName, itemRc, itemGc, borderColor)

append a knowledge graph result into result panel div

Parameters:

Name	Type	Description

Name	Type	Description
itemId	string	id of a knowledge graph result
itemName	string	name of a knowledge graph result
itemRc	string	description of a knowledge graph result
itemGc	string	URL link to website
borderColor	string	color of border

Source: knowledgeGraph.js, line 90

queryMainEntity(id, type)

currently not used. left for reference

Parameters:

Name	Type	Description
id		
type		

Source: knowledgeGraph.js, line 119

refreshStoryList()

it removes all stories from the story-list div

Source: app.js, line 218

removeDuplicates(storyList) → {Array}

Given a list of stories, it removes any duplicates

Parameters:

Name	Type	Description
storyList	Array	a list of stories

Source: app.js, line 286

Returns:

return unique stories

Type
Array

retrieveAllStoriesData(storyList, forceReload)

it cycles through the list of stories and requests the data from the server for each story

Parameters:

Name	Type	Description
storyList		the list of the cities the user has requested
forceReload	boolean	true if the data is to be retrieved from the server

Source: app.js, line 125

selectItem(event)

callback called when an element in the widget is selected tell others the selected result

Parameters:

Name	Type	Description
event		the Google Graph widget event https://developers.google.com/knowledge-graph/how-tos/search-widget

Source: knowledgeGraph.js, line 61

sendChatText()

called when the Send button is pressed. It gets the text to send from the interface and sends the message via socket

Source: chat.js, line 65

showOfflineWarning()

show offline warning

Source: app.js, line 267

signup()

post signup data to add a new user redirect to login page after signup successfully

Source: users.js, line 22

(async) storeCachedData(name, roomId, AnnotationObject) → {Promise.<void>}

it saves the annotation in localStorage

Parameters:

Name	Type	Description
name	string	user id in a chat room
roomId	string	id of a chat room
AnnotationObject	object	object of annotation

Source: annotationDatabase.js, line 61

Returns:

set item

Type
Promise.<void>

(async) storeKnowledgeGraphToCachedData(KnowledgeGraphObject) → {Promise.<void>}

it saves the knowledge graph in localStorage

Parameters:

Name	Type	Description
KnowledgeGraphObject		

Source: annotationDatabase.js, line 86

Returns:

Type
Promise.<void>

(async) storeStoryToCachedData(story_title, storyObject)

it saves the stories in localStorage

Parameters:

Name	Type	Description
story_title	string	title of a story
storyObject	object	a story object

Source: database.js, line 49

widgetInit()

it inits the widget by selecting the type from the field myType and it displays the Google Graph widget it also hides the form to get the type

Source: knowledgeGraph.js, line 34

writeOnChatHistory(text)

it appends the given html text to the history div

Parameters:

Name	Type	Description
text		the text to append

Source: chat.js, line 129

Documentation generated by JSDoc 3.6.10 on Fri May 20 2022 20:56:11 GMT+0100 (British Summer Time)