CS5590 APS
-
Programming for Web/Cloud based Application
ASSIGNMENT 3

-Saidu Babu Dosapati
ID: 14

Description:

YouTube videos

Aim:

Create an application which will display the desired videos based on the keywords entered by the user in the search tab (Hint: Youtube API)

Objective:

- When user searches with a key word in the application, this application would display the list of videos which matches the keyword searched
- Every result will have the details of the video and was embedded on to the application in a html page
- On clicking the video, we used the hyperlink
- Handling any unexpected values with displaying in appropriate values

Approach:

Steps:

1. Designed a HTML page Where we can add code on demand from Javascript which was done in a separate file in a directory
2. We designed this application using Bootstrap for making the website responsive and Beautiful looking
3. Using Bootstrap makes the website and the user feel the good user experience and also with a good user interface
4. We designed a
    a. Login page
    b. Registration page

   New users can register/sign up and
   existing users can login

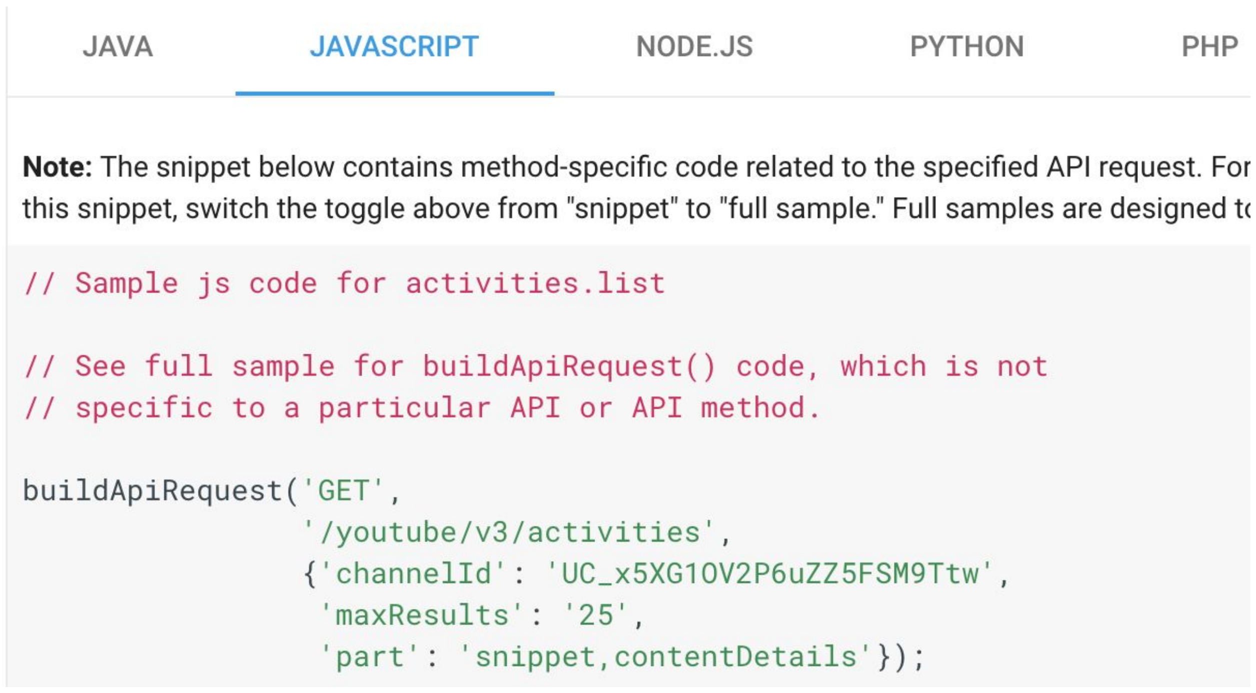
5. We designed the application using the youtube API
6. We made the AJAX for making an API call on the background
7. We get the response in the format of JSON file, when we call the api using a get method, we generally recieve the file in the format of javascript object notation
8. Inorder to evaluate the data we got and make it useful we need follow the steps. Next, We parse the JSON data using JSON.parse(response.text)
9. Then we append the data into HTML page which we already created containing nothin, for displaying the users
10. When the user enters a source key into the application and selecting the news source and start searching
11. The results get displayed below by using the API call Successfully

```
// The client ID is obtained from the {{ Google Cloud Console }}
// at {{ https://cloud.google.com/console }}.
// If you run this code from a server other than http://localhost,
// you need to register your own client ID.
var OAUTH2_CLIENT_ID = '__YOUR_CLIENT_ID__';
var OAUTH2_SCOPES = [
  'https://www.googleapis.com/auth/youtube'
];

// Upon loading, the Google APIs JS client automatically invokes this callback.
googleApiClientReady = function() {
  gapi.auth.init(function() {
    window.setTimeout(checkAuth, 1);
  });
}

// Attempt the immediate OAuth 2.0 client flow as soon as the page loads.
// If the currently logged-in Google Account has previously authorized
// the client specified as the OAUTH2_CLIENT_ID, then the authorization
// succeeds with no user intervention. Otherwise, it fails and the
// user interface that prompts for authorization needs to display.
function checkAuth() {
  gapi.auth.authorize({
    client_id: OAUTH2_CLIENT_ID,
    scope: OAUTH2_SCOPES,
    immediate: true
  }, handleAuthResult);
}

// Handle the result of a gapi.auth.authorize() call.
function handleAuthResult(authResult) {
  if (authResult && !authResult.error) {
    // Authorization was successful. Hide authorization prompts and show
    // content that should be visible after authorization succeeds.
    $('.pre-auth').hide();
    $('.post-auth').show();
    loadAPIClientInterfaces();
  } else {
    // Make the #login-link clickable. Attempt a non-immediate OAuth 2.0
    // client flow. The current function is called when that flow completes.
    $('#login-link').click(function() {
      gapi.auth.authorize({
```



Note: The snippet below contains method-specific code related to the specified API request. For
this snippet, switch the toggle above from "snippet" to "full sample." Full samples are designed t

```
// Sample js code for activities.list

// See full sample for buildApiRequest() code, which is not
// specific to a particular API or API method.

buildApiRequest('GET',
                '/youtube/v3/activities',
                {'channelId': 'UC_x5XG1OV2P6uZZ5FSM9Ttw',
                 'maxResults': '25',
                 'part': 'snippet,contentDetails'});
```

Result:

The Application were designed Successfully

Conclusion:

- This application can be made much more beautiful by focusing on the CSS
- We can also add animations to improve the user experience and interface.

TASK 2

Description: Twitter friends list visualization using D3.js

Aim:

- Get the twitter friends list using twitter API (https://developer.twitter.com/en/docs/accounts-and-users/follow-search-get-users/overview) and
- Visualize them through D3.JS Marks will be distributed between logic, implementation and UI

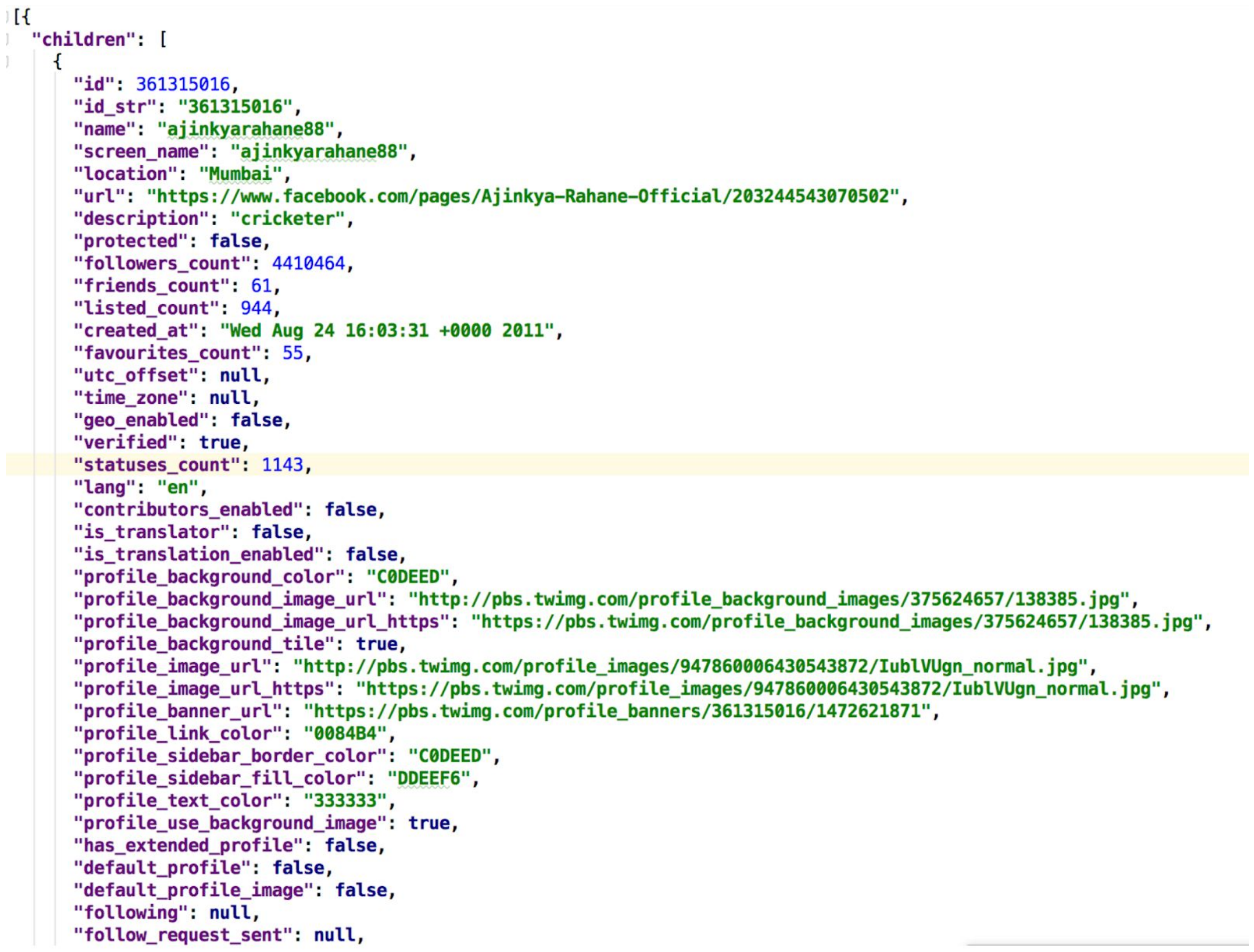
Objective:

- We get the friends list of the user in the twitter
- We then visualize the friends list with D3.js after getting the results in the format of JSON

Approach:

Steps:

1. First we create the developers account in twitter
2. We then get the credentials necessary to call/use the twitter application api
3. After getting the data output in the format of JSON
4. We then use the D3.js library to visualize the json format of the output data
5. We create the visualization of the application of twitters friends list
6. We also added the implementation of the application by adding the features of edit/modifying and moving the data by dragging and dropping
7. We are also using mongo db for the data generally, if to store the data in the cloud mlab
8. The JSON data we obtained will contain



```
[{
  "children": [
    {
      "id": 361315016,
      "id_str": "361315016",
      "name": "ajinkyarahane88",
      "screen_name": "ajinkyarahane88",
      "location": "Mumbai",
      "url": "https://www.facebook.com/pages/Ajinkya-Rahane-Official/203244543070502",
      "description": "cricketer",
      "protected": false,
      "followers_count": 4410464,
      "friends_count": 61,
      "listed_count": 944,
      "created_at": "Wed Aug 24 16:03:31 +0000 2011",
      "favourites_count": 55,
      "utc_offset": null,
      "time_zone": null,
      "geo_enabled": false,
      "verified": true,
      "statuses_count": 1143,
      "lang": "en",
      "contributors_enabled": false,
      "is_translator": false,
      "is_translation_enabled": false,
      "profile_background_color": "C0DEED",
      "profile_background_image_url": "http://pbs.twimg.com/profile_background_images/375624657/138385.jpg",
      "profile_background_image_url_https": "https://pbs.twimg.com/profile_background_images/375624657/138385.jpg",
      "profile_background_tile": true,
      "profile_image_url": "http://pbs.twimg.com/profile_images/947860006430543872/IublVUgn_normal.jpg",
      "profile_image_url_https": "https://pbs.twimg.com/profile_images/947860006430543872/IublVUgn_normal.jpg",
      "profile_banner_url": "https://pbs.twimg.com/profile_banners/361315016/1472621871",
      "profile_link_color": "0084B4",
      "profile_sidebar_border_color": "C0DEED",
      "profile_sidebar_fill_color": "DDEEF6",
      "profile_text_color": "333333",
      "profile_use_background_image": true,
      "has_extended_profile": false,
      "default_profile": false,
      "default_profile_image": false,
      "following": null,
      "follow_request_sent": null,
```

Result:

We successfully created the Twitter application of friends list

Conclusion:

- This application can be made much more beautiful by focusing on the CSS
- We can also add animations to improve the user experience and interface