CS 646 Android Mobile Application Development
Spring Semester, 2017
Assignment 4

San Diego State University -- This page last updated 2/28/17

Assignment 4 - SDSU Hometown Locations

Due March 19 23:59

Goals

Network
Maps

Relation to Assignment 5

Assignment 4 will be continued in assignment 5. In assignment 5 you will add functionality and provide a tablet version of the app. So you will find using fragments in this assignment useful in assignment 5.

App Description

This app will allow users to post where they are from and allow them to see on a map where others are from. The app has two basic parts. First part is posting the user information and second is the viewing of data posted by other users.

Posting User Information

The users post eight pieces of information described below.

nickname

> A string used to identify the user to others. The two users can not use the same nickname. Nicknames are case insensitive. That is “FooBar” is considered the same as “foobar”. Required.

password

> A string determined by the user. Will be used later to allow the user to enter/modify their information. Passwords must be at least three characters long. Required.

country

> A string which is the country the user is from. The list of supported countries is available from the server (see server protocol below). The server does not accept posts containing countries not supported by the server. Required.

state

A string which is the state/providence the user is from in their country. The list of supported states is available from the server (see server protocol below). The server does not accept posts containing states not supported by the server. Required.

city

A string which is the city the user is from. The server does not have a list of cities. Required.

year

An integer representing the year the user started attending SDSU. The year needs to between 1970 and 2017. Required.

longitude & latitude

The latitude and longitude of the users hometown or if the user prefers of their actual home. Only people in a few professions knows the latitude and longitude of anything, so do not ask the user to enter these numbers. Show them a map and let them mark a point on the map. Use this point to determine the latitude and longitude. Latitude values are doubles in the interval [-90, 90] and Longitude values are doubles in the interval [-180, 180). These values are optional in the sense that the user data can be sent to the server without them. You will lose points if you can not do this.

Viewing User's Hometown

The app gives the user two ways to view the hometowns posted to the server. The first is as a list and the second is markers on a map. When showing the data as a list the viewer needs to be able to access the nickname, country, state and city of each entry. In the map tapping on the marker should provide the nickname of the person represented by the marker. The user needs to be able to filter data by country, state and year. The user may only want to see locations of users in California, or users that started SDSU in 2016, or users in California that started SDSU in 2015.

Server Protocol

The server supports six GET requests and one POST request detailed below.

When a request is successful it will return a response with the HTTP status 200. If there is a problem with the request the response will have a HTTP status of 404 or 400. When you request a URL that does not exists you will get a 404 response and the body of the response will be HTML. This can happen when you mistype the URL. Other errors will return a response with a 400 or 404 status and the body of the response will be JSON. This can happen if you do not include a required parameter, misspell a parameter or have a value that is out of range or the wrong type. All 200 responses indicate that their bodies contain JSON data.

Note that the URLs given below are case sensitive. The path and all query parameter names are lower case.

GET URLs

http://bismarck.sdsu.edu/hometown/countries

Parameter: None

Returns JSON array of strings, the names of the countries currently supported

Sample return value:

```
["China","India","Mexico","USA"]
```

http://bismarck.sdsu.edu/hometown/states?country=ACountryName

Parameter: ACountryName = one of the country names returned from the first URL

Sample: http://bismarck.sdsu.edu/hometown/states?country=USA

Return JSON array of string, names of the states/providences in given country

Sample return value (Shortened for space reasons

```
["Alabama","Alaska","Arizona","Wyoming"]
```

http://bismarck.sdsu.edu/hometown/nicknameexists?name=ANickName

Parameter: ANickName = A nickname that you wish to see if is already used

Sample: http://bismarck.sdsu.edu/hometown/nicknameexists?name=foo

Returns: JSON boolean. Returns true if nickname is already in use.

Sample return value:

false

http://bismarck.sdsu.edu/hometown/count

Parameter: none

Returns: Integer number of current users. As new users can be added to the server at any time this value may not be correct for very long. The actual number of users may be larger

Sample return value:

12

http://bismarck.sdsu.edu/hometown/nextid

Parameter: none

Returns: The id that will be used for the next user. This is only useful when using the page query parameter with the reverse query.

Sample return value:

3

http://bismarck.sdsu.edu/hometown/users

The URL returns a JSON array of the users that have been added to the server. Each user is a JSON object. The keys in the object are: nickname, country, state, city, year, longitude, latitude, id and timestamp. The values at the keys nickname, country, state, city, year are as sent in the adduser post (see below). If the longitude and latitude are not give in the adduser post the values returned are 0. Each user has an id, which is added by the server. Ids are integers. The ids are unique and are increasing. Below is an sample response from the server. It is what the server returns after the two samples in the POST URL section are sent to the server. Since rew2 was sent after rew that user has an larger id. The time stamp is when the user data was received by the server in UTC.

Note that the list of users by default is sorted by increasing id. That is users that were added to the server earlier are listed earlier. The list is sorted by increasing time stamp.


```
[{
   "nickname": "rew",
   "city": "Devils Lake",
   "longitude": 0,
   "state": "North Dakota",
   "year": 1985,
   "id": 1,
   "latitude": 0,
   "time-stamp": "2017-03-01T02:44:03.247Z",
   "country": "USA"
 },
```

```
  {
    "nickname": "rew2",
    "city": "Minneapolis",
    "longitude": -93.293753,
    "state": "Minnesota",
    "year": 1985,
    "id": 2,
    "latitude": 44.96606,
    "time-stamp": "2017-03-01T20:31:00.632Z",
    "country": "USA"
  }
]
```


This request has nine optional query parameters. These are used to further specify which users to return or to reverse the order of the list. These are described below.

country

Sample: http://bismarck.sdsu.edu/hometown/users?country=USA

Returns only the users in the given country

state

Sample: http://bismarck.sdsu.edu/hometown/users?state=Minnesota

Returns only the users in the given state.

city

Sample: http://bismarck.sdsu.edu/hometown/users?city=Minneapolis

Returns only the users in the given city. Since different cities can have the same name (Paris, France & Paris, California) it is best to combine this with country and state query parameters. For example:

http://bismarck.sdsu.edu/hometown/users?country=USA&state=Minnesota&city=Minneapolis

year

Sample: http://bismarck.sdsu.edu/hometown/users?year=1985

Returns only the users that started SDSU in the given year.

afterid

Sample: http://bismarck.sdsu.edu/hometown/users?afterid=1

Returns only the users with ids strictly greater than the one given.

beforeid

Sample: http://bismarck.sdsu.edu/hometown/users?beforeid=2

Returns only the users with ids strictly less than the one given.

reverse

Sample: http://bismarck.sdsu.edu/hometown/users?reverse=true

When the value of reverse is true the order of the list returned is in reverse order. That is the most recent user added is now first in the returned list rather then last. That is the return list is sorted by timestamp with time decreasing, which is the same as sorting by decreasing id.

page

Sample: http://bismarck.sdsu.edu/hometown/users?page=0

When the list of users gets long it may not be practical to get the entire list in one request. Then this query parameter is present the list is divided into groups or pages. The default size of the group or page is 25 users. This can be change. See pagesize below. When page=0 you get the first 25 users. When page=1 you get the second 25 users. To get the entire list you increase the value by 1 until you get back an empty list. If you use pages with the reverse set to true you have to be careful. Pages are computed on the current list of users. On a reverse list first query to get the nextid then always use the beforeid with the page as in the following:

http://bismarck.sdsu.edu/hometown/users?page=2&reverse=true&beforeid=30

pagesize

Sample: http://bismarck.sdsu.edu/hometown/users?page=0&pagesize=5

This changes the size of pages from the default 25. Only has an affect when used with the page query.

The query parameters above can be used individual or combined. You can use all or none of them. The order they are listed the URL does not matter.

POST URL

http://bismarck.sdsu.edu/hometown/adduser

Body of the request: The body of the request is a JSON object with six required keys and two optional keys which are nickname, password, country, state, city, year, longitude, latitude. The latter two are optional. These are described above in the “Posting User Information” section. Two samples are:

{"nickname" : "rew",
"password" : "catman",
"country" : "USA",
"state" : "North Dakota",
"city" : "Devils Lake",
"year" : 1985}

{"nickname" : "rew2",
"password" : "guess",
"country" : "USA",
"state" : "Minnesota",
"city" : "Minneapolis",
"year" : 1985,
"latitude" :44.966060,
"longitude" : -93.293753
}

If the post is successful then the server response has status code 200 and the body contains the string “ok”.

Grading

Points	Item
5	Can enter required data for new user
5	Can successfully get list of countries & states from server
15	Can submit new user data to server
10	Can display list of all current users. Can see nickname, state, city, and year
10	Can filter the list of users based on country, state and year
10	Can generate latitude and longitude location for new user from map
10	Display current user's location on a map with marker when have latitude and longitude
10	Display current user's location on a map with marker when user did not enter latitude and longitude when submitting their information.
20	Display all current users from country or state on a single map. No users outside the state or country are shown on the map. Each user has a marker on the map
5	Can see the user's nickname when tap on marker on map

What to Turn in

Create a zip file of your entire android project. Please no rar files. Turn in your assignment at: http://bismarck.sdsu.edu/CoursePortal. Don't forget to reduce the size of your project before you turn it in.