CSC 372/472: Mobile Applications Development for Android

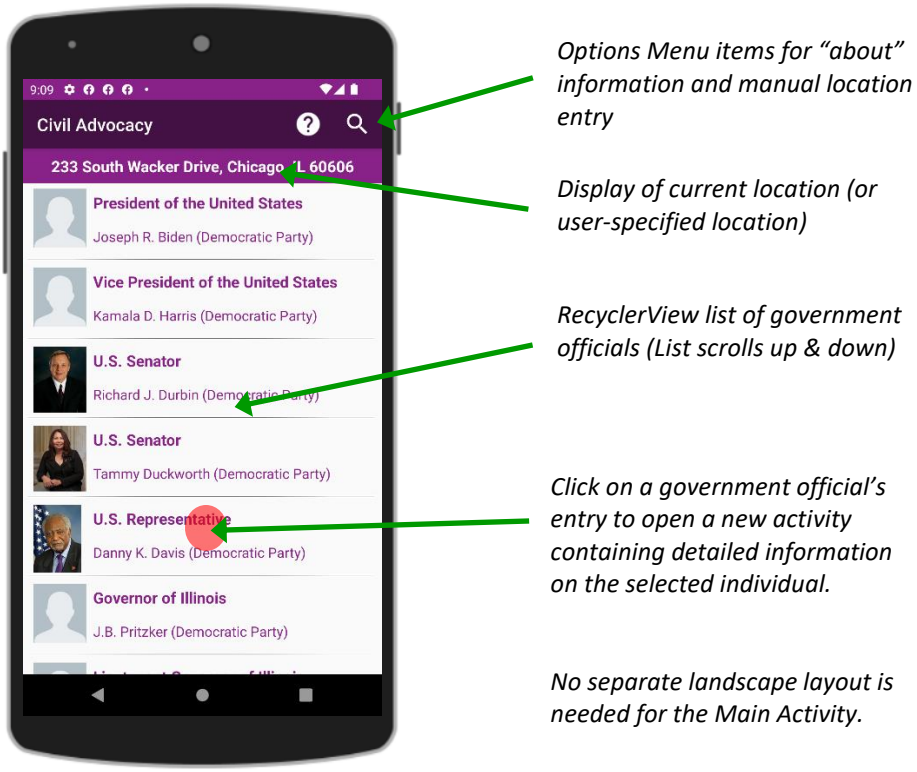
Assignment 4 – Civil Advocacy App *(400 pts)*

Uses: Location Services, Internet, Google APIs, Images, Picasso Library, Implicit Intents, TextView Links

App Highlights:

- This app will acquire and display an interactive list of political officials that represent the current location (or a specified location) at each level of government.
- Android location services will be used to determine the user's location.
- The Google Civic Information API will be used to acquire the government official data (via REST service and JSON results).
- You *will* need to use a different layout for landscape orientation for 2 of the activities in this application. Those details are specified later in the document.
- Clicking on an official's list entry opens a detailed view of that individual government representative.
- Clicking on the photo of an official will display a Photo Activity, showing a larger version of the photo.
- An "About" activity will show application information (Author, Copyright data & Version)
- Your manifest should add permissions for ACCESS_FINE_LOCATION and INTERNET
- The application is made up of 4 activities, shown below:

1) Main Activity



NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

© Christopher Hield

2) *Official Activity*

Display of current location (or user-specified location)

Basic official data
-Name
(i.e., Danny K. Davis)
-Office
(i.e., U.S. Representative)
-Party
(i.e., Democratic Party)

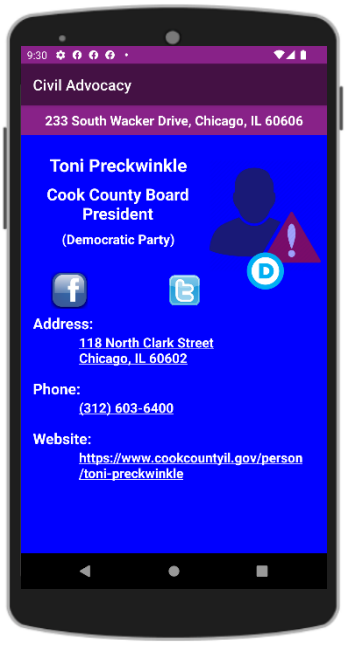
The background color of this activity is based upon the official's political party:
Republican = RED,
Democratic = BLUE,
Otherwise use BLACK

Social Media (only display data that is provided).
All are clickable – implicit intents.
Possible data items:
- Facebook
- Twitter
- YouTube

Party logo for Democratic & Republican. Otherwise, no logo

Contact Information (only display data that is provided). All are clickable – implicit intents. Possible data items:
- Office Address
- Phone Number
- Email address
- Website

- *Photo of official (where available). NOTE, a default image should be displayed when a photo is not specified (see below, middle)*
- *Error image should be displayed if the provided image URL doesn't exist.*
- *Clicking on the photo opens Photo Detail Activity*

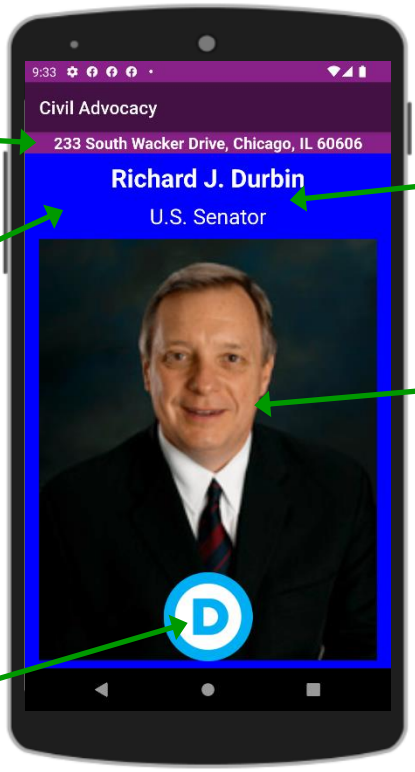


3) Photo Detail Activity

Display of current location (or user-specified location)

The background color of this activity is based upon the official's political party:
Republican = RED,
Democratic = BLUE,
Otherwise use BLACK

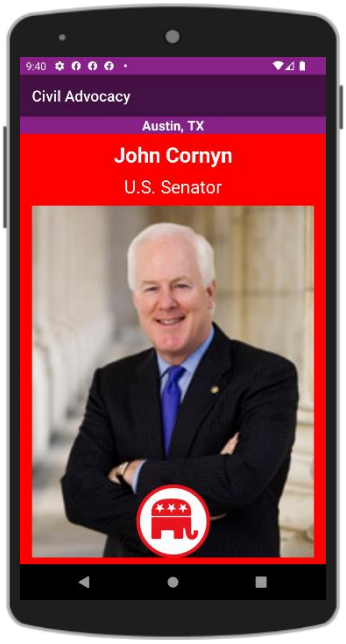
Party logo for Democratic & Republican. Otherwise, no logo

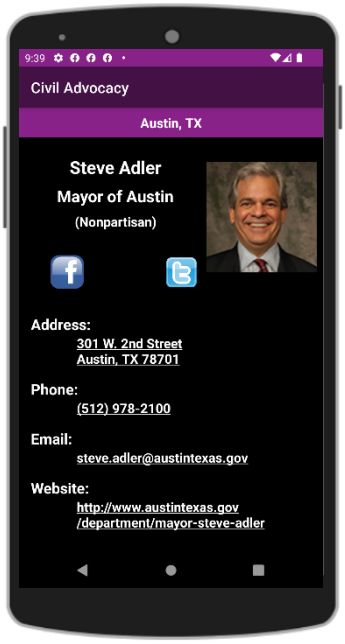


Basic official data
1) *Name (i.e., Richard J. Durbin)*
2) *Office (i.e., U.S. Senator)*

"Full-sized" image of official (where available).

NOTE: This activity should <u>not</u> open if the representative's image is not specified.

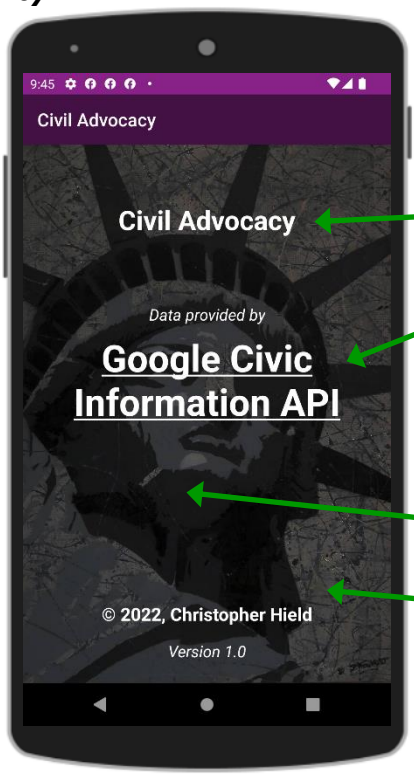




NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

© Christopher Hield

4) About Activity



App Title

API Attribution (clickable link to https://developers.google.com/civic-information/). Note: This is a TextView with the text underlined, and an onClick defined to go to the Google Civic Information URL).

Full-sized background image

Application information (Author, Copyright data & Version)

Required Landscape Layouts

Official Activity

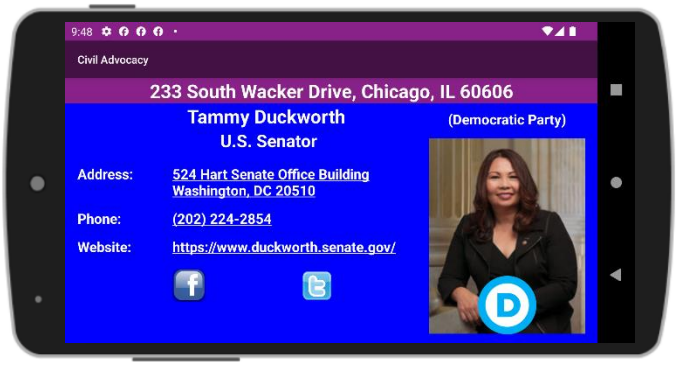
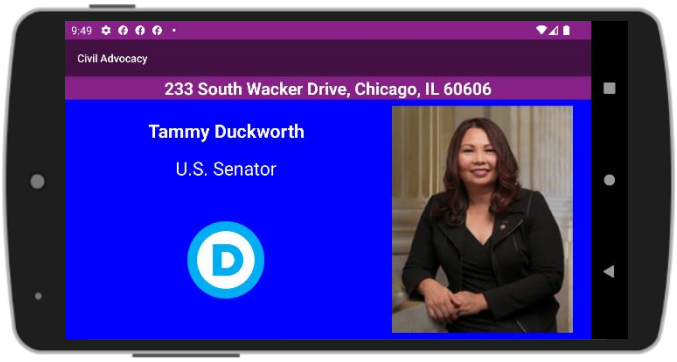


Photo Activity



NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

Internet Data:

Downloading data for the government officials requires a download via the Google Civic Information API. The Civic Information API lets you enter a zip code or city/state name to look up the data properties for the elected officials in those districts.

NOTE: You MUST sign up with Google to get an API key. Your API KEY must be supplied with all Google Civic Information API queries. You can get a free API key by following the instructions at:

https://developers.google.com/civic-information/docs/using_api

Follow the instructions for "Acquiring and using an API key". The type of credential you need to create is "API Key" (not "OAuth client ID" and not "Service account key"). This is a very quick and easy process. The API key will be a long string of characters. The results of Civic Information API calls are returned in JSON format. The content of the returned results is described later in this section.

Query Format: https://www.googleapis.com/civicinfo/v2/representatives?key=*Your-API-Key*&address=*zip-code*

https://www.googleapis.com/civicinfo/v2/representatives?key=*Your-API-Key*&address=*city*

For example, if your API Key was "ABC123xyz" and the zip code was 60605, your full URL would be:

https://www.googleapis.com/civicinfo/v2/representatives?key=ABC123xyz**&address=**60605

For example, if your API Key was "ABC123xyz" and the city was Chicago, your full URL would be:

https://www.googleapis.com/civicinfo/v2/representatives?key=ABC123xyz**&address=**Chicago

Google Civic Information API Results Example:

The JSON results you receive contains 4 sections, described in detail below. The *normalizedInput* section contains location details for the results provided. The *divisions* section lists political geographic divisions, like a country, state, county, or legislative district *(we do not need this section for our application).* The *offices* section lists the political positions governing the specified location. The *officials* section lists people presently serving in the offices specified in the *offices* section.

High-level view of JSON results:

```
{
    "normalizedInput": {   ← We need this section, it is accessed as a JSONObject (described later)
        … ← Data will be here
    },
    "kind": "civicinfo#representativeInfoResponse", ← We do not need this
    "divisions": { ← We do not need this
        … ← Data will be here, we do not need this section
    },
    "offices": [   ← We need this section, it is accessed as a JSONArray (described later)
        … ← Data will be here
    ],
    "officials": [   ← We need this section, it is accessed as a JSONArray (described later)
        … ← Data will be here
    ]
}
```

JSON Section Detail:

1) The "normalizedInput" JSONObject contains the following:

```
"normalizedInput": {
        "line1": "",
        "city": "Chicago",      ← We want this for the location display in our activities
        "state": "IL",          ← We want this for the location display in our activities
        "zip": "60654"          ← We want this for the location display in our activities
},
```

2) The "divisions" section comes next, but we do not need this section, so it is not described here.

3) The "offices" JSONArray contains multiple instances of the following:

```
"offices": [
  {
   "name": "President of the United States",   ← We want this, the "office" title
   "divisionId": "ocd-division/country:us",
   "levels": [
    "country"
   ],
   "roles": [
    "headOfState",
    "headOfGovernment"
   ],
   "officialIndices": [                        ← We want this, it is the index into the "officials" JSONArray (see the next section) which contains the details of the person that holds this office.
    0                                          NOTE: There can be more than one index as time offices have multiple representatives.
   ]
  },

  {
   "name": "United States Senate",             ← We want this, the "office" title
   "divisionId": "ocd-division/country:us/state:il",
   "levels": [
    "country"
   ],
   "roles": [
    "legislatorUpperBody"
   ],
   "officialIndices": [                        ← We want this, it is the index into the "officials" JSONArray (see the next section) which contains the details of the person that holds this office.
    2,                                         NOTE: There can be more than one index as time offices have multiple representatives.
    3
   ]
  },

  …   ← The above sections repeat many times, once per government office

],
```

NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

© Christopher Hield

4) The "officials" JSONArray contains multiple instances of the following:

```
"officials": [
 {
  "name": " Joseph R. Biden",
  "address": [
   {
    "line1": "The White House",
    "line2": "1600 Pennsylvania Avenue NW",
    "city": "Washington",
    "state": "DC",
    "zip": "20500"
   }
  ],
  "party": " Democratic Party",
  "phones": [
   "(202) 456-1111"
  ],
  "urls": [
   "http://www.whitehouse.gov/"
  ],
  "emails": [
   "email@address.com"
  ]
  "photoUrl": "https://www.whitehouse.gov/sites/whitehouse.gov/files/images/45/PE%20Color.jpg",
  "channels": [
   {
    "type": "Facebook",
    "id": "whitehouse"
   },
   {
    "type": "Twitter",
    "id": "whitehouse"
   },
   {
    "type": "YouTube",
    "id": "whitehouse"
   }
  ]
 },
 {
    ... ← The above section repeats many times, once per government official. Remember – the first official
          in the JSONArray is index 0. The second official in this JSONArray is index 1, the third second
          official in this JSONArray is index 2, and so on.
 }
]
```

This is the first JSONObject is the first official (index 0). This index corresponds to the "officialIndices" we found in the "offices" section earlier.

This is the name of the person that holds this office.

This is the Address (check for possible line1, line2 & line3 – concatenate them into one String), City, State & Zip Code of this person's office

This is the political party of this person: "Republican", "Democratic/Democrat", or "Unknown". Note this section might also be omitted – consider that as party "Unknown".

This person's office phone number. There may be more than one – just use the first entry. Note this section might also be omitted.

This person's office web site. There may be more than one – just use the first entry. Note this section might also be omitted.

This person's office email address. There may be more than one – use only the first entry. This section might also be omitted.

This is the URL to the person's photo. Note this section might also be omitted. In this case, use a "place holder" photo.

These are the user ids for the related social media channels. There will be up to four entries. Possible entries are:

- *Facebook*
- *Twitter*
- *YouTube*

Some or all of these 4 items might be omitted.

NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

Use a Photo Library for Photo Downloads

Downloading the photos of the officials in the Official Activity and the Photo Activity must use either the Glide or Picasso libraries (as was discussed in class).

Social Media Implicit Intent Examples

The following are examples of how you should code the social-media implicit intents (some of these have been seen already in class examples):

- **Twitter** (define an onClick image to execute the Twiter implied intent as used in class examples)
- **Facebook** (define an onClick image to execute the Facebook implied intent as used in class examples):
- **YouTube** (example onClick method to be associated with the YouTube ImageView icon):

```
public void youTubeClicked(View v) {
    String name = <the official's youtube id from download>;
    Intent intent = null;
    try {
        intent = new Intent(Intent.ACTION_VIEW);
        intent.setPackage("com.google.android.youtube");
        intent.setData(Uri.parse("https://www.youtube.com/" + name));
        startActivity(intent);
    } catch (ActivityNotFoundException e) {
        startActivity(new Intent(Intent.ACTION_VIEW,
                        Uri.parse("https://www.youtube.com/" + name)));
    }
}
```

Provided Icons

To insure a consistent and professional appearance of our application, the following image files are being provided to you for use in this assignment. You can simply download them, add them to your Android Studio project, and use them with your ImageViews.

About Activity Background

Default Official Image

Bad photo URL image

Separator Image (to add to the end of your list entry layout)

Facebook icon

Twitter icon

YouTube icon

Launcher Icon

Democratic Party Logo

Republican Party Logo

Application Behavior Diagrams:

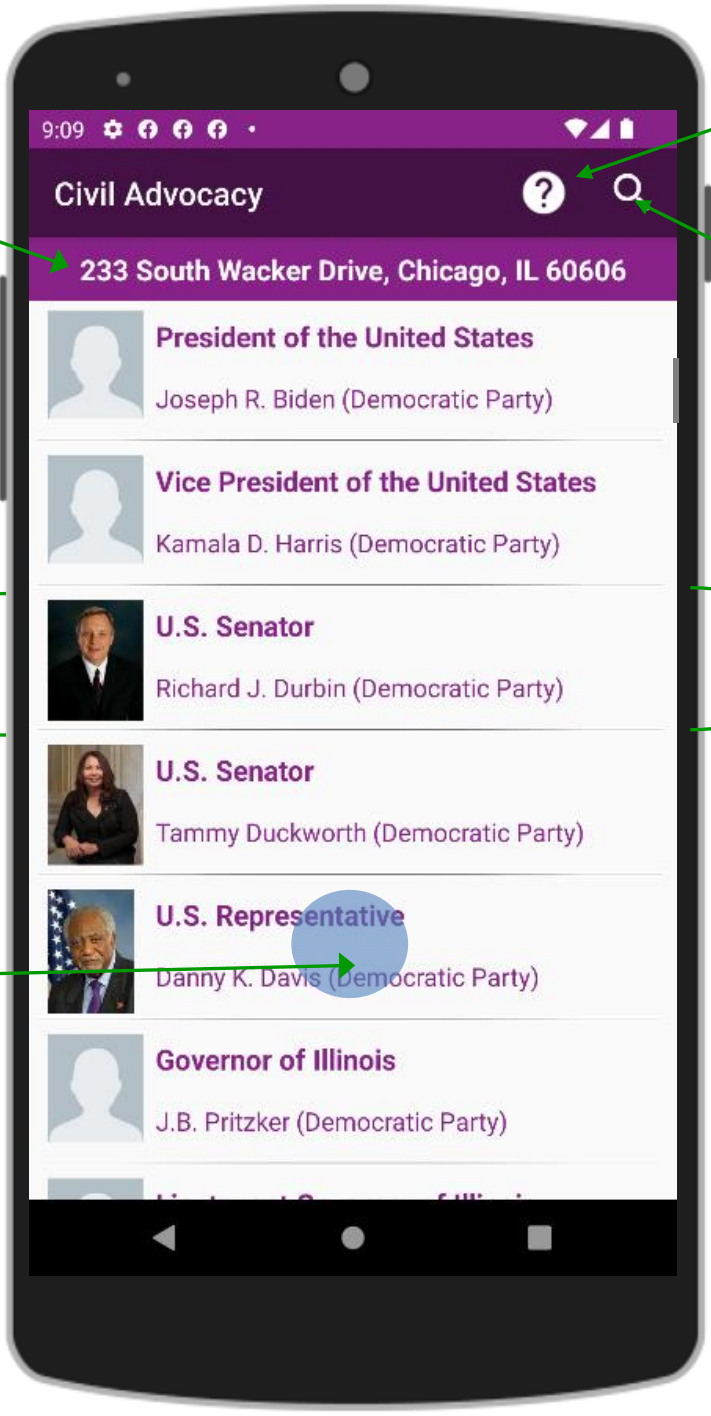
1) App MainActivity

On startup, the application should use the device's current location zip code to populate the list with officials.

Location information comes from the normalizedInput section of the JSON download.

The "info" options menu item opens the "About" activity.

The "Location" options menu item opens an alert dialog that allows the user to enter a city/state or zip code manually.



Each political official entry displays the image, office, and the official's name and political party. The political party is in parenthesis. A separator image should be added to the end of each list entry to create a nice break between entries.

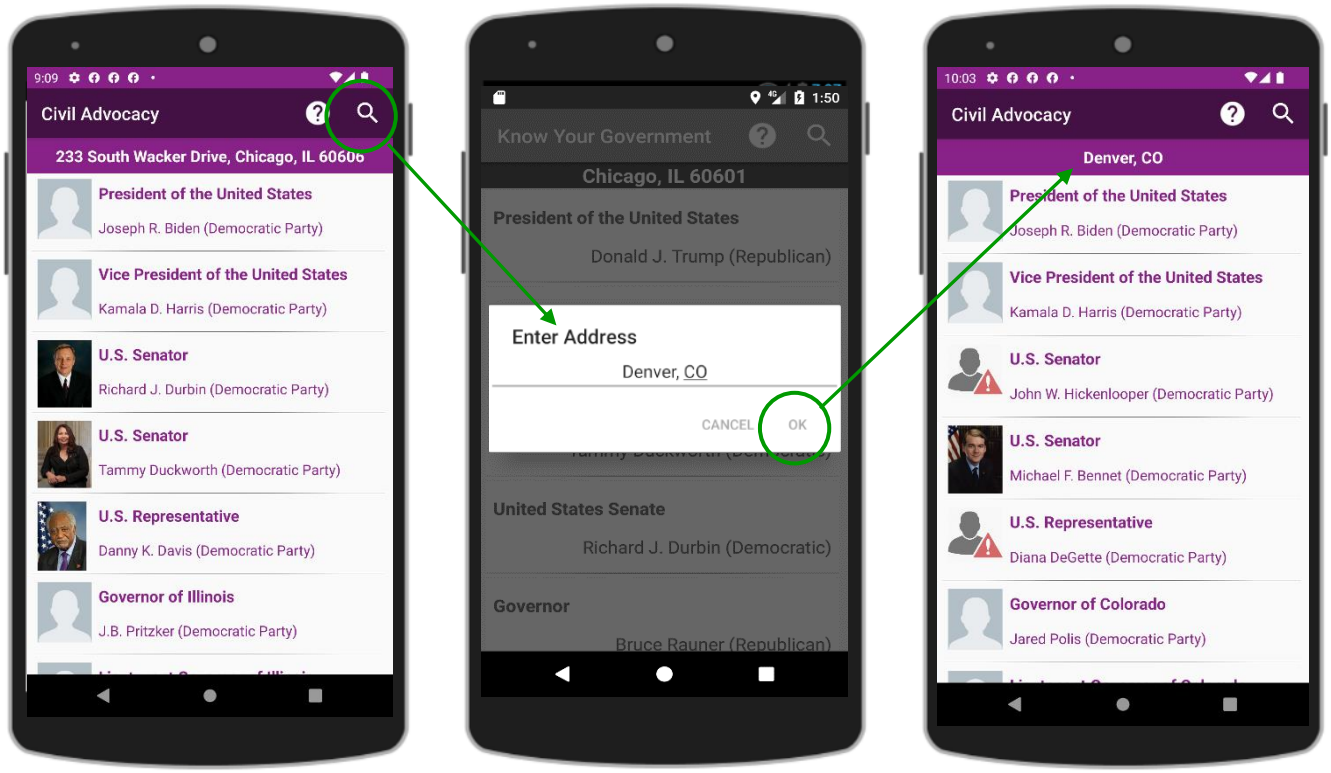
RecyclerView list entries have their own layout

Click on an entry to open the individual Representative Activity

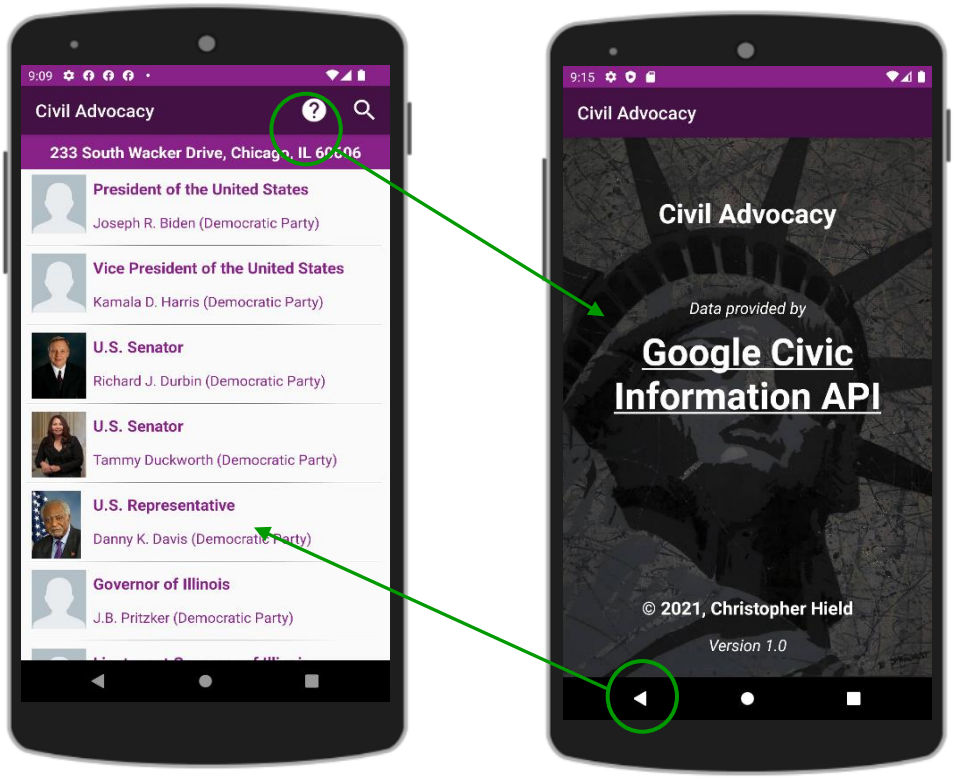
There is no need for a long-press/click capability here.

NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

2) Manually Setting the location:

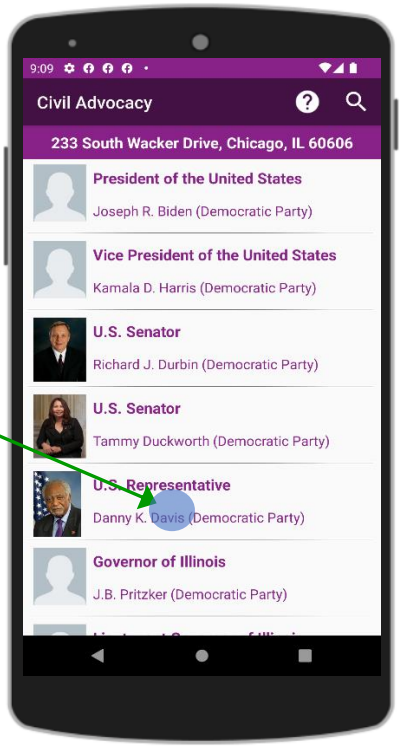


3) Opening the About Activity



4) Viewing an individual Official:

Click on an entry to open the individual Official Activity



5) Interactive links on the official's activity:

Click on the address (where available) to open the office location in Google Maps

Click on the phone number (where available) to open the Phone App with the phone number pre-loaded)

Click on the Email Address (where available) to open the email app with this address pre-loaded in the new message

Click on the Website address (where available) to open the website in a browser.

Click on the photo (where available) to open the Photo Activity

Click the Party Logo when present to go to the political party's website:

Dem: https://democrats.org

Rep: https://www.gop.com

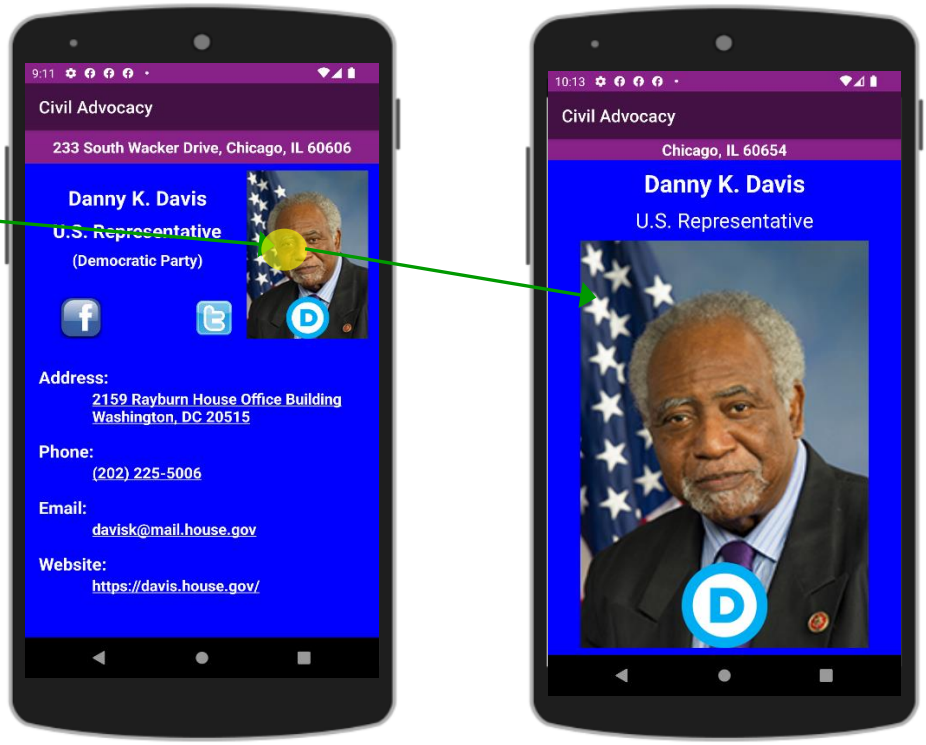
Social media links. Where supplied, the official's social media links will be displayed using the social media's icon. If available, the icon is present - if not it is absent (the icon should not be displayed).

Clicking on these will open the related app (if installed) or will default to opening the site in a web browser.

NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

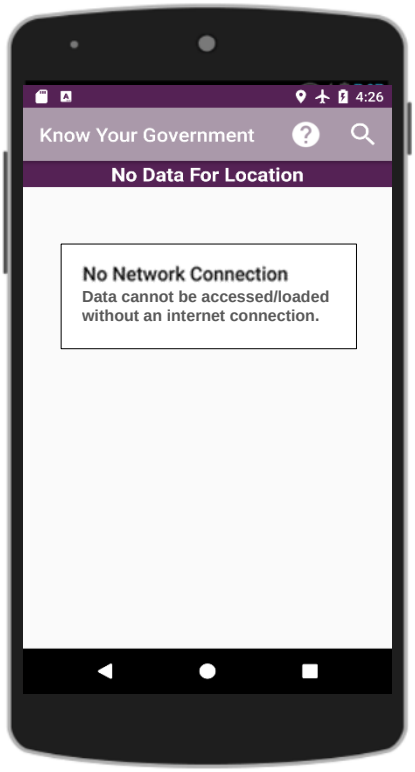
6) Viewing an individual Official's Photo Activity:

Click on the Photo to open the Photo Activity



7) If no location or no internet connection:

On StartUp or new specified location query



NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

© Christopher Hield

Development Plan

1) Create the base app:
    a. MainActivity with RecyclerView
    b. Official Class
    c. RecyclerView Adapter
    d. RecyclerView ViewHolder
    e. Create fake "dummy" officials to populate the list in the MainActivity onCreate.
    f. Add the onClick methods. The onClick can open a Toast message for now.
    g. Zip/City options-menu item opens the dialog, on entry you can open a Toast message for now.
    h. Create the About Activity – opens when MainActivity "About" options menu item is selected.

2) Add the location code.
    a. Add the location code that determines the device's zip code (this happens in onCreate).
    b. Instead of using that zip code to make the Google API call, you can open a Toast message that displays the zip code for now.

3) Add the Official Activity:
    a. This activity opens when an entry in the list of elected officials is clicked on.
    b. You can use the data in your test (dummy) official objects to test this activity.
    c. Any test data you don't have (i.e., missing data) should be properly handled in your activity.
    d. The social media ImageView onClick's should open a Toast message that displays the name of the social media for now.
    e. Remember to create the separate layout for landscape orientation.

4) Add the Google Civic Information API elements:
    a. Create the Google Civic Information API Runnable class.
    b. Remove the use of dummy data from your app (now you will have real data).
    c. Remove the location Toast message. Instead, here you use the device's zip code (or a manually entered location) to make the API call.
    d. This should result in a populated list of Official objects in your MainActivity that is then displayed in the RecyclerView.
    e. Add the Photo Activity, opened when an official's photo is clicked in the Official Activity
    f. Remember to create the separate layout for landscape orientation.

5) Test the app very thoroughly and review your implementation against all requirements multiple times.

NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

© Christopher Hield

Assignment Assistance

If you are stuck on an assignment problem that you have exhaustively researched and/or debugged yourself, you can email me a ZIP file of your entire project so that I can examine the problem. All emailed assistance requests must include a detailed description of the problem, and the details of what steps you have already taken in trying to determine the source of the problem.

Note: To make your submission zip file smaller, before zipping your project file, you can remove the ".gradle" folder (found in your project's root directory), and remove the "build" folder (found in the "app" folder in your project's root directory).

Submissions & Grading

1) Submissions must consist of your zipped project folder (please execute Build =>Clean Project before generating the zip file).
2) Submissions should reflect the concepts and practices we cover in class, and the requirements specified in this document.
3) NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED
4) Grading will be based upon the presence and proper functionality of *all features and behaviors* described in this document.

- Grading will be performed with the following SDK details:
    - Project Minimum SDK Version: 25

5) Grading will be performed on emulator devices with the following characteristics:

Resolution	Details	Example Emulators to Use
1080 x 1920	With Playstore	Pixel, Pixel 2, Nexus 5, Nexus 5X
1080 x 2220 or 2280	With Playstore	Pixel 3a, Pixel 4

NOTE

This assignment is worth 400 points. This means (for example) that if you get 89% on this assignment, your recorded score will be:

(89% * 400 points = 356 points)

If you do not understand anything in this handout, please ask.

Otherwise the assumption is that you understand the content.

Unsure? Ask!

NO LATE ASSIGNMENT 4 SUBMISSIONS WILL BE ACCEPTED

© Christopher Hield