Contents

[Introduction 3](#_Toc312316454)

[What’s Included 4](#_Toc312316455)

[Before you start: 5](#_Toc312316456)

[Software Environment 5](#_Toc312316457)

[Let’s go! Download Template and Unzip the Contents 5](#_Toc312316458)

[Template Contents 6](#_Toc312316459)

[Configuring your PollMap 8](#_Toc312316460)

[Configure the user interface to start using your poll’s questions 9](#_Toc312316461)

[Review and Learn about the Data 9](#_Toc312316462)

[Determine your geographic scope and how you want the maps to look 10](#_Toc312316463)

[Edit Maps behind Your Services to Match Your Questions and Responses 10](#_Toc312316464)

[Serving Your Newly Edited Maps 11](#_Toc312316465)

[Configure Additional Services 12](#_Toc312316466)

[Configuring the Social Media Widgets 13](#_Toc312316467)

[Determine How People Will Learn About Your Map 13](#_Toc312316468)

[Additional Branding 14](#_Toc312316469)

[System Architecture 15](#_Toc312316470)

[Considerations for Deploying your PollMap on a Production Server 16](#_Toc312316471)

[How Voting Works 16](#_Toc312316472)

[Analyzing the Results 17](#_Toc312316473)

[Additional Resources 17](#_Toc312316474)

[Known Issues 18](#_Toc312316475)

[Appendix A: Definition Query Examples 19](#_Toc312316476)

[Radio (2 Choices) 19](#_Toc312316477)

[Radio (3 or more Choices) 19](#_Toc312316478)

[Allocation 19](#_Toc312316479)

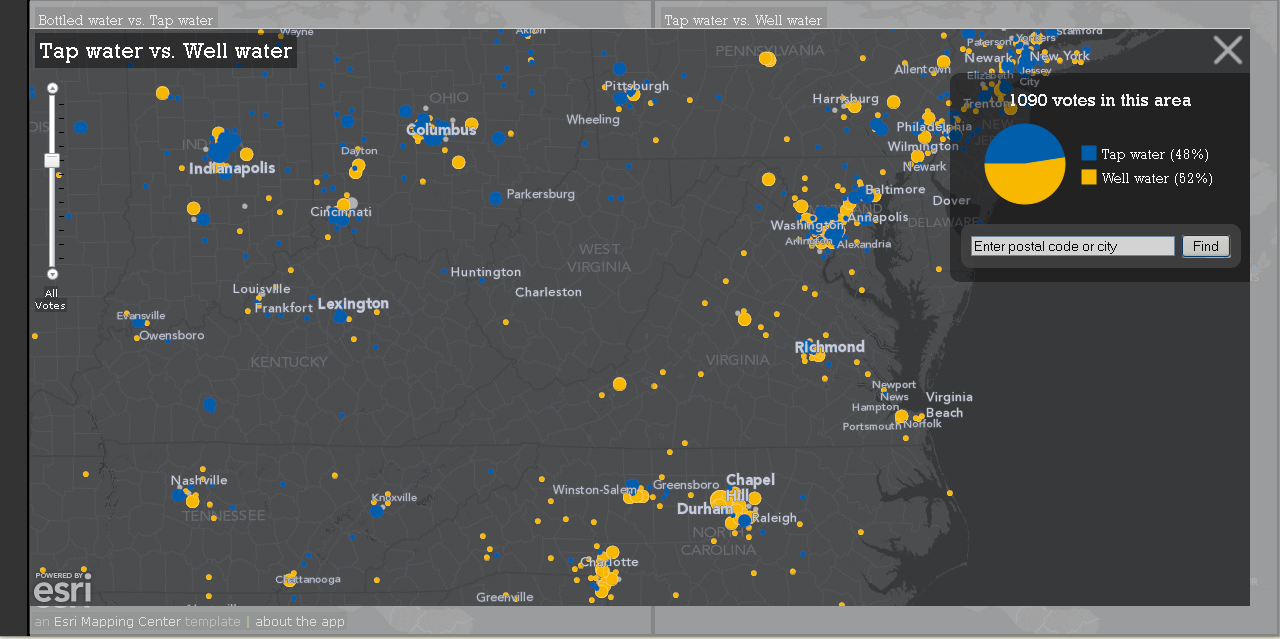
[Appendix B: Config.js question type examples 20](#_Toc312316480)

[Radio (2 Choices) 20](#_Toc312316481)

[Autocomplete (*n* Choices) 20](#_Toc312316482)

# Introduction

This Public Opinion Map, also known as the Esri FanMap™ or PollMap, allows you to capture and map public opinion on any issue or topic you configure the template to address. It encourages participation through a simple user experience that focuses on simple things: reading the question(s) and choosing among the options presented, and then immediately viewing results on the map. This gives the visitor instant feedback and an option to explore the map to see how their vote fits within local, regional and even national and global contexts.

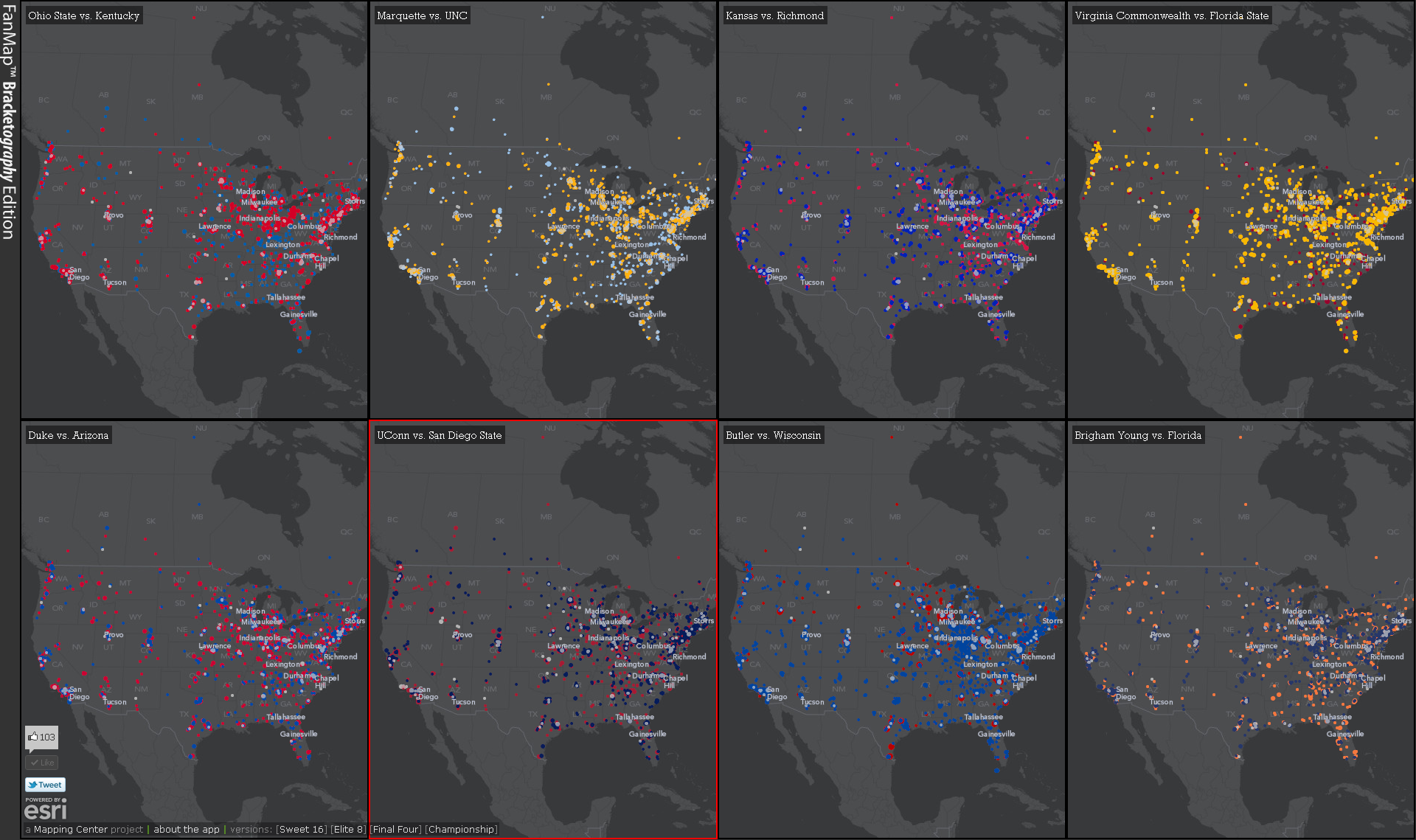
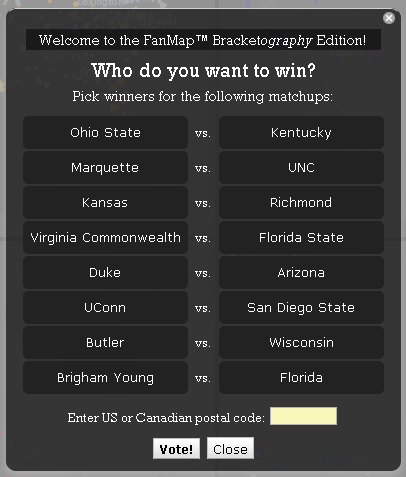
****

**Figure 1: The sample poll this template includes allows people to vote their preference.**

This template is built using the Esri JavaScript API. It is an interactive map primarily intended for use in a standard web browser or tablet device. It requires no plug-ins or installs by the user. Though it may function in smaller mobile devices like an iPhone or Droid phone, it is not specifically built to support those types of smaller devices.

The template includes sample data to record votes by postal code in the U.S. and Canada, and URLs to basemaps hosted by Esri on basic cloud servers. Contact us if you intend to deploy anything which may attract more than 100 visitors an hour so that we can work with you to arrange proper support. Additional data for vote collection in other countries or globally by city is available at the [Public Opinion Poll ArcGIS Online Group](http://www.arcgis.com/home/group.html?owner=MappingCenterTeam&title=Public%20Opinion%20Maps).

This is version 2.01 of the template. Two kinds of questions are supported: a basic multiple choice “A vs. B”, and an allocation of resources question. The documentation is written with the multiple choice type of question in mind. Once you are familiar with how the template works, you will start to see how and where the allocation type of question is supported, and gain insight for customizing the experience to support other kinds of questions. This documentation will not spell out that path for you.

**Figure 2: FanMap: Bracketography Edition, an example of what this template can support**

The template includes everything you need to have a working application on a single computer within a couple of hours.

The application works well with the following options:

* Microsoft Internet Explorer 8/9, Mozilla Firefox 3.6 or higher, Google Chrome 10 or higher, and Safari 5.1 and higher.
* 1280x1024 Screen Resolution. It can work on smaller screens, it mainly depends on height of the Vote box (see Figure 2 above)

## What’s Included

* Web application source code
* Geoprocessing source code
* Map service MXDs
* Data for postal codes in the U.S. and Canada
* URLs to basemaps you can use during development
* Simple documentation

## Before you start:

This template’s documentation presumes you’ll get everything running on a single machine using the sample application, maps and data; and then configure that deployment to accommodate your needs. You’ll need to use a computer running Microsoft IIS and ArcGIS Server 10. See the section near the end that discusses options for server architectures. The idea for now is that you are working on what can be considered a development environment and will be configuring that until you’re ready to move your work into a production environment.

## Software Environment

The following software must be installed and configured:

* ArcGIS Server 10 w/ SP 1 for the Microsoft .Net Framework – Standard or Advanced
* ArcGIS Desktop 10 w/ SP 1 (and Maplex Labeling Extension if you want to cache your own basemaps from the “canvas” basemap templates, to be made available summer 2011)
* (recommended for final deployment) ArcSDE for Microsoft SQL Server 2008

Note: ArcGIS Server can be configured on other supported web servers but this template has been optimized for ArcGIS Server the Microsoft.Net Framework. Other configurations have not been tested.

Firefox with the Firebug plug-in (or Google Chrome) can really help diagnose any issues as they are encountered.

Once you have the sample web app and maps running (a few steps from now, instructions are below), you’ll want to modify colors to suit your subject, and have the ArcMap colors of those features match the charts produced in the app. ColorPic is a great tool for checking colors onscreen and converting RGB values (used in ArcMap) to HEX values (used in the web pages).

ArcSDE is recommended for deployments that expect to generate higher amounts of traffic. The main consideration is concurrent use, meaning more than 1 user per 1-2 seconds. ArcSDE is multi-user and designed to accommodate this kind of traffic without dropping or shutting out users.

# Let’s go! Download Template and Unzip the Contents

Presumably you’ve already downloaded the template; hence you are reading this document. If not, search [www.arcgis.com](http://www.arcgis.com) for “Public Opinion Map” and get the latest version, and work with that documentation. This documentation covers Version 2.01.

Unzip the file to your wwwroot directory, under /WaterPoll (or any other name desired – this documentation assumes /WaterPoll. You can use anything you want, but fair warning: you won’t want to change this later unless you absolutely have to.

After you unzip everything successfully, go to your web browser and check if he URL works. For example, my machine name is called “skyray” so my URL is <http://skyray.esri.com/WaterPoll>. You should see a dark screen and a Vote window in front of that. No maps will appear at this point.

If you try to submit a vote at this point, you’ll get an error message. That makes sense – we have not started any services yet.

If you don’t see the dark background and Vote window popup, troubleshoot based on the errors you see, or get an IIS expert to take a look. A good strategy is to type in the exact error message you are seeing into a search engine and look for possible causes.

If the site otherwise works, let’s move on.

## Template Contents

The following files are provided in the ZIP file. This is for an overview only – the sequence by which you need to make changes is described in the next section. This table is for reference only.

|  |  |  |
| --- | --- | --- |
| **Template Directory** | **Item** | **Description** |
| (root) | index.html | The web page used to launch the PollMap / FanMap. Edit this file so that it contains the title, description, tags etc. you want. |
|  | PollMap documentation.docx | That’s what you’re reading. I hope. |
| css | style.css | Controls many UI elements. Strictly speaking, you should not need to edit anything here. |
| database | votes.gdb | This file geodatabase captures the incoming votes and also captures the summarizations by postal code. Out of the box it contains simulated votes, so that you’ll see something on the sample maps. It also contains a “blank” database so that you can start fresh when ready, and a copy of a simulated votes table. |
| gp | Schema.gdb | An empty schema that’s needed for the GP to function. |
|  | summarize.py | Summarizes the vote count in a given map extent. Results are used to update the pie chart onscreen. Edit this file if you change the location of your votes tables. |
|  | vote.py | Records the vote and increments the totals in a given geography. Edit this file if you change the location of your votes tables. |
|  | vote.tbx | Toolbox which contains the above scripts. This toolbox is served in ArcGIS Server. |
| js | config.js | Edit this file to configure the choices available to the user, and their colors in the pie charts. |
|  | plugins.js | Should not need to edit this. |
|  | script.js | Should not need to edit this. |
| /libs directory |  | Should not need to edit anything in this directory. |
| /esri/PollMap/Templates directory | VoteWindow.html | Edit this file to configure the question asked on the Vote Window. |
|  | AboutWindow.html | Edit this file to give details of your implementation. |
|  | GeocodeView.html | Edit this file if desired. |
|  | SummaryPane.html | Edit this file if desired. |
| /js/esri/pollmap/img directory | 1sidebar.png | This has an example of Esri’s branding of the sidebar. Create your own version of this file and rename it “sidebar.png” for the app to display. |
| lib |  | Dojo items are here. |
| media |  | Sample graphics files. Nothing to change here. |
| mxds | Question01\_highvote.mxd | The map document used to author the map for Question01. Contains symbology for Question01’s choices. Update this .MXD and use it to create your .MSD file for serving. |
|  | Question01\_highvote.msd | The map service definition used to author the Question01 map service. |
|  | Question02\_highvote.mxd | The map document used to author the map for Question02. Contains symbology for Question02’s choices. Update this .MXD and use it to create your .MSD file for serving. |
|  | Question02\_highvote.msd | The map service definition used to author the Question02 map service. |
|  | Zips\_Winkel.mxd | Simple map service to serve postal code points. No editing needed. |
|  | Zips\_Winkel.msd | The map service definition used to author the Zips\_Winkel map service. |

## Configuring your PollMap

After you’ve confirmed that the site works (you can submit a vote and see a map afterwards with no error messages), then you can start the process of configuring the application files to suit your situation. The general sequence is below and while it can all be accomplished in a few days, it takes most people 2-3 weeks before they’re comfortable sharing their Public Opinion Map with the world, particularly the first time.

1. **Determine your questions:** The important thing here is to write them down once, taking no more than 20 minutes to complete the poll. You’ll learn more quickly by diving in and configuring the poll you just unzipped and tested. Expect at least a week of refining and adjusting of words and phrases.
2. **Set the geographic scope of your poll:** Currently you can support a global poll by collecting the city name respondents provide, or a national or smaller scope poll based on postal codes. The default poll (water map) targets U.S. and Canadian postal codes (currently there is no support for multiple countries when two or more countries share the same format for postal codes). Cities and additional postal code data can be found at the [Public Opinion Maps group](http://www.arcgis.com/home/group.html?owner=MappingCenterTeam&title=Public%20Opinion%20Maps) on ArcGIS Online. This should take no more than an hour or two.
3. **Set up your map services:** It’s likely that your questions will have a different number of answers than our example, so you will need to edit these map documents, changing the fields, definition queries, and symbols for the reporting layers. This work will take anywhere from a few hours to two days depending on the number of questions and responses you have.
4. **Reset the votes table and zero out the responses:** Initially use the default votes table and results supplied in the example to see what your map will look like. This allows you to set the symbology for your map effectively. Once you are ready to test your map for the final time, you’ll want to verify using a zero vote count scenario.
5. **Deploy your map:** Once the map works with zero votes, it’s ready to deploy on to a public server. This means you will copy your website, the maps, and data to that server. Then you’ll create new services and edit the configuration files to use those services, edit your geoprocessing scripts to point to your new data location (if different), make sure all social media widgets correctly point to your new url, and set up analytics (such as Google Analytics). Your Public Opinion Map should be ready to use.
6. **Promote your map:** Just because your map is deployed and working doesn’t mean people will find it. Planning for marketing your map will be ongoing throughout the above phases of work. The general idea is to gradually tell people about your map. Initially you will monitor all activity, usually having a few close colleagues using the map, and then you may spread the word to a slightly larger community. For instance, we first test the map with just the development team (5 people), and then we share the map to a group of about 20 people, and then to about 100 people. We have had to reset the votes and were surprised by many things that are now fixed in the 2.01 version during these early releases. It’s better to get the initial issues worked out on a small friendly audience.
7. **Fix issues and refine:** The Public Opinion Map is flexible and can be edited while it is running. The first week of operation is usually characterized by a set of issues that come in fast, but then dwindle to just a few items by the end of that week. So far this is normal—every poll is different and the issues will vary.

## Configure the user interface to start using your poll’s questions

***Edit the VoteWindow.html file*** to tweak the question and set context for responses. Especially early on, experiment, doing one change at a time to learn where each item you’ve edited is used as part of the experience. Decisions you make here flow into subsequent edits in other files, so it’s good to get this squared away early.

***Edit the config.js file*** to change the possible responses (aka “CHOICE01”, “CHOICE02” etc) and their options and colors.

1. Under “**app”**,start with the **type** variable. Pollmaps cannot mix types of questions, so what you set here will apply to all of your questions
2. See the examples for each autocomplete, radio and allocation types of questions. You will see which variables are needed.
3. Experiment with subtitles, which can be used as section breaks between radio type questions. (needs to be brought over from your lunchpollmap)

**Edit index.html** to change the page title, description, tags etc. to suit your situation.

**Edit SummaryPane.html** to tweak the text. This is the pane that shows the pie chart on the large map window.

**Edit GeocodeView.html** to tweak the text. This is the FIND box on the user interface underneath the chart.

**Edit AboutWindow.**html to enter new text needed here to suit your map. Once your application is deployed on the production server, update Facebook, Twitter, etc.

## Review and Learn about the Data

For now, you don’t need to do anything to the sample data. Later, when you are ready to start with a clean slate, read this again. For now, just have a look at the data in ArcCatalog to get familiar with the schema and content.

Votes.gdb contains five datasets:

* Votes Table, which is how votes are recorded. When a person clicks the submit button in the application a record is added to this table indicting the location they selected and which options they chose. For radio or auto-complete type questions a 1 is set in the appropriate CHOICExx field to indicate what they chose. For allocation type questions, the amount the user allocated is recorded. The cookie field is a way to determine who voted (summarizing the table based on this will tell you how many unique users (or at least, browsers) participated). The DateTime field records when the vote was submitted.
* Votes\_blank Table, which is the schema for the votes table and will be used to create the clean slate when you deploy the application. If you ever need to recreate this table, just make a copy of any current Votes table and then run the [Delete Rows tool](http://help.arcgis.com/en/arcgisdesktop/10.0/help/index.html#//0017000000n6000000) on the copy.
* PostalCodes Points, which is the sample data that you saw when you set up the template.
* PostalCodes\_Empty. This is the dataset you’ll use to collect responses to your questions once you deploy. The easiest way to reset the vote is to have this table handy to copy and rename to PostalCodes—that way you are only managing data and not needing to reconfigure anything to get started.
* PostalCodes\_TestData, which is copy of the sample data. Sometimes it is useful to go back to this data, like if you need to reset any symbology or if you change the number of potential answers to your questions after you have been testing your work.

Once you have tested and verified everything is working with the sample data and you are ready to deploy the application you will need to stop all services, then copy and rename the votes table and PostalCodes feature class. Then restart ArcGIS Server services.

## Determine your geographic scope and how you want the maps to look

Now is the time to decide what the initial geographic extent of your maps will be and what projection to use. We have used both Web Mercator as a project because we could use Esri’s Canvas Basemaps and we have also made custom base maps that depicted our area of interest the most geographically responsible way given that we were creating a multi-scale web map application. To help you learn more about this decision we creating some blog entries that discuss the decisions we made:

[Alternative to Web Mercator: Winkel Tripel](http://blogs.esri.com/Support/blogs/mappingcenter/archive/2011/02/17/an-alternative-to-web-mercator-winkle-triple.aspx)

[Making the FanMap: Super Bowl Edition](http://blogs.esri.com/Support/blogs/mappingcenter/archive/2011/02/11/making-the-fanmap-super-bowl-edition.aspx)

Once you have determined the extent and coordinate system, you need to enter this information in the config.js file. There are some standard entries already there that are commented out, to save you time.

## Edit Maps behind Your Services to Match Your Questions and Responses

Each question your poll asks requires a map to be served in order to display results. Each map has one layer for each possible response to a question and an additional layer for showing ties. Because these maps need to look good at multiple scales, we’ve provided three designs, one for large scales, another for small scales, and one for very small scales. The rationale is that at small scales the dots must be relatively small to have a legible, or readable, map. As a user zooms in the dots can be larger, and once a user zooms into large scales the dots scale. Try zooming in from a global extent to one cluster of dots—you will see the subtle change we introduce. The way this works is by having three group layers in the map. That means you will need to make edits to each of the layers within these group layers to make your map work.

1. Open Question01.mxd and rename the data frame to reflect the question the map corresponds to. Then rename the layers to reflect the possible responses. Question01.mxd is a map for a radio type question where there are two possible responses which are derived from fields CHOICE01 and CHOICE02. If you do not need to add more responses or changed the fields used to symbolize the data, follow the procedure in step 2, just change the colors of the symbol to match what is defined in the config.js file for the question..
2. To add more responses, or if you need to change which fields are being used to symbolize the data follow these steps:
   1. Open the layers properties and remove the definition query. Click Apply to set the new definition query.
   2. Go to the symbology tab and first edit or just save the template symbol. This will give you a good symbol to work with.
   3. Change the field – notice that this will reset how the symbology is applied. The test datasets were specially prepared so that each field contains exactly the same data, making it possible for you to set exactly the same symbol size ranges for each layer, which will allow them to be visually comparable. Change the color of the symbol to match what is defined in the config.js file for the question.
   4. Click Apply in the Layer Properties dialog to set the new symbology.
   5. Go back to the definition query tab and set the correct definition query for the layer.
   6. Repeat for each layer and then skip to step 4.

## Serving Your Newly Edited Maps

When ready to serve your modified map, here is the procedure to follow to ensure the maps look right and are easily maintainable:

1. Turn on the Map Publishing toolbar and check that your Options (right most button on the toolbar) for Anti-aliasing is set to “Best” and for Text Anti-Aliasing is “Force”.
2. From the Map Publishing toolbar, Preview your map and set the properties Image Format to PNG32. This required and ensures the map service will produce the best looking display of your data.
3. Then export to MSD. Be sure you pay attention to what directory you are writing the MSD to, so that you don’t save it in a place which the service isn’t drawing from on restarts. This is a backup copy.
4. To publish your map using the Publish to ArcGIS Server button. If this is a new service, edit the config.js file to add the Service URL (from the REST endpoint of your server) to the questions and maps sections. If you are updating a service, still use this option because it saves your MSD in your arcgisserver/input folder, and the Publish to ArcGIS Server button automatically updates that file.
5. You may need to clear the REST cache for your server to see your changes. The rest cache is at a URL like this: <http://skyray.esri.com/arcgis/rest/admin>. For some changes, it is also necessary to clear your browser’s cache too.

Remember, if you rename or create a new service, that’s fine – just edit the config.js file listed above. In fact this is a good time to determine the names for the map services that will support your Public Opinion Map.

We’ve found we spent a fair portion of the time to deploy a Public Opinion Map is in editing the maps. One tip for saving time on editing maps is to prototype your questions using the sample data with the goal of getting the look of the VoteWindow.html finalized. Only once you are certain that you have final list of potential responses to each question should you begin the process of setting up your maps. The operation that takes the longest is adding a response to a given question. The best way to know you have the questions done is to have people you respect try to vote and give their honest reactions to the questions. They will tell you if something is missing or phrased in an awkward way.

The votes table, the PostalCodes points, and the citytown points are set up to handle up to 30 potential responses. If you need more… well, consider dropping a question or refining your poll’s objective; 30 potential responses is a lot to ask Web people to do spend time assimilating. 30 responses will generate a large amount of work (870 possible pair-wise summaries, and that does not include grouping). These polls should bequick and fun to use, even if the topic is serious. If you find you need more responses, remember to add fields to both the votes table and your points feature class. If you do not have expertise in survey design, consider consulting someone who does—they will be of great help.

## Configure Additional Services

1. Set up feature service next. Use postalcodes.msd, serve as a normal map service. Do not turn on “Feature Access” as it is unnecessary while you are in a development environment (see the System Architecture section below for further information).
2. Set up a geometry service next. In ArcCatalog, right click on your server name, and click Add New Service. Select “Geometry” service and hit Next until completion, accepting all defaults for now.
3. Set up GP services next. Edit the two python scripts to make sure the path to the database matches your development or production database. After editing, or if you have no changes at this point, go to ArcCatalog and navigate to the scripts’ folder. Right-click on the votes.tbx toolbox and choose “Publish to ArcGIS Server.” Accept the defaults for now. After it starts, stop the service and right-click on it, choose Properties. Change the Execution type to synchronous. Set the timeouts for these services to be 6-8 seconds, any longer will cause the map to look like it is hung. If you have a lot of traffic, it’s better to be at 8 seconds just in case it takes longer to calculate the summary for the chart that is drawn when a user pans or zooms. Usually that takes 1-2 seconds.
4. If you are using SDE, you will need to use the Create ArcSDE Connection File tool (in Data Management Tools 🡪 Workspace) in order to make a file that will be the basis for your python scripts to have a complete path string that points to your data.

## Configuring the Social Media Widgets

The default PollMap contains connections to two Social Media outlets: FaceBook and Twitter. These are configured in three files:

1. Index.html where you will need to uncomment two sections of code. In the first section you will customize the parameters to match your map’s topic. In the 2nd section you will provide the parameters needed to initialize the FaceBook Like button, and the Tweet button.
2. AboutWindow.html where you will uncomment and customize the section of code at the top to match the first section of #1 above.
3. Config.js where you will customize the value of the megaMapSocialWidgetsHTMLvariable. This is a little more difficult for having all the code for the script contained in one string, so save this for last, after you have the code working.

For Twitter, you do not need your own account, though it is helpful to test. Depending on how you want to leverage Twitter, you can get help here: <https://dev.twitter.com/docs/twitter-for-websites>.

For Facebook, you will need an account and you will need to create an application, which will provide you with the information needed to update the code in index.html. Go to: <http://developers.facebook.com/> to get started, you will want to create an app for your PollMap. The idea is that users will be “Liking” your app.

## Determine How People Will Learn About Your Map

If you work for an organization that has a marketing or public relations department, now is a good time to let them know what you’re doing and that their help would be very much appreciated. Hopefully they were the ones who asked you for this map in the first place.

Another scenario is that you’re doing research as a student or faculty at a university. Social media, including email, is going to be your primary means of telling people about your map. Start drafting content early. You’ll want at least three different pitches for at least three different audiences. The pitches should fit within the following lengths: 140 characters, 300 characters, and 300 words. The three audiences are: your friends and family, people you know such as colleagues, and people you don’t know. The latter category will be identified more as groups that may be interested in your map’s content. Some of your messages will be generated from your map if you are using the social media widgets, so tailor those messages as if the users of your map are sending them to their friends, family, and colleagues (think to eliminate work on their part in helping you get the word out).

If you’re in this latter category of people without a marketing department to help you, here are some things to consider:

1. Getting 500 responses in your first week is a really good start, and it will take you about 8-10 hours of concerted effort to make that happen during the course of that week.
2. Most organizations will not help you, but try anyway. For the most part, they will be wary of any risks in being associated with your map.
3. Do press releases for the local papers, including campus newspapers—each will use a tailored version of your 300 word pitch. Follow up with the editors until you have confirmation of go or no-go.
4. Get friends to advocate your map on social media promotion sites like Reddit,Digg or StumbleUpon.
5. To leverage sites like FaceBook or LinkedIn, look for groups or pages that have tens of thousands of people who like them. Expect that a post here will be acted upon by 0.001% to 0.1% of people. Yes, that means you’re trolling for 1s, 5s, and 10s of submissions.
6. To leverage Twitter, it’s good to have well connected credible friends. If you’re new to Twitter, that may be hard to achieve quickly. Either way, here are a few tips:
   1. Search for hash-tags (starts with a #) that relate to your topic. Make sure you leverage those hash tags in your messaging from the app and in tweets you generate.
   2. Message well connected people who are obviously potentially interested in your map. Ask them to have a look and if they like your map to please tweet it.
   3. Get TweetDeck to monitor activity. The Twitter web site can help too.
   4. It’s important to have or find several credible well connected friends on Twitter. These people can generate dozens and even hundreds of hits for your map.
7. The time of day that you use social media matters. 9:00 AM to 3:00 PM on weekdays is most effective, with early to mid morning being the best. Weekends and Evenings will often produce less than 10% of the results that a weekday message produces.
8. Consider setting up an email account to deal with the feedback and management of promoting your map. However, do not set up a new social media account (unless your map is part of a larger marketing or product promotion campaign) because nobody will know about it, and you’ll find yourself promoting that account and your map.
9. Make sure you have an audience of people to initially populate your map with responses. You’ll want about 50 people to start things off. An empty Pollmap is rather pathetic. Consider putting in some responses yourself to seed things if you don’t have enough people to get things started. You’re in control, and you can, once the poll is done summarize the first 100 responses in the votes table to determine how many to subtract from the final totals (so you don’t analyze your seed responses).

## Additional Branding

Edit file 1sidebar.png (or use one of the sidebar.ai files in the media directory) with your content, and rename it to “sidebar.png” to have the app pick up and display it. Result appears on left side of app.

In the text at the bottom of the map you have the opportunity to solicit feedback and brand yourself or your sponsors by including text that links to appropriate web sites. The About window (AboutWindow.html) is another place you have an opportunity to tell your story and additional brand or distinguish your Public Opinion Poll.

## System Architecture

If you’ve implemented the system as we’ve described above, you have a single server configuration that is capable of handling about 250 users per hour without affecting performance provided you using a mid-sized server. For the sake of this discussion, a mid-sized server is one with 4 CPUs, 8Gb of RAM and is running a 64-bit OS. The only remaining issue to consider is peak load, particularly if users are concurrently accessing your map. Concurrent use is when two or more people simultaneously ask your map to do something. Our testing has shown this one-server configuration will handle the occasional burst of two concurrent users.

We monitor activity on our servers by looking in the ArcGIS Server properties to see the response times for our services. Our expectation is that responses for map service will be under 0.2 seconds, and we like to see the GP services at under 1.8 seconds. Under peak load it is okay for these to be double. But if they are consistently higher, then add more processes to these services to handle that load. It is not uncommon to have 6-12 processes per map service and 10-20 processes for the GP service. Once you have this many services, it is good to monitor overall system resource usage to ensure you do are not coming close to the limits of your system’s CPU or memory resources. If you are, you will need to add other servers.

We have also implemented Public Opinion Maps on a more secure three-server configuration:

1. Geoprocessing services on one server. The GP Services tend to be CPU-intensive, meaning that of memory, Disk I/O, or CPU, the latter resources is most heavily used.
2. Map Services on one server: Map Services tend to be memory and File I/O intensive rather than CPU intensive.
3. Database on one server. The database is the only bottleneck in this application. By isolating it on it’s own server, it’s use is maximized. It is also more secure because there is no need to expose the database to direct use. In our architecture, the only access to the database is through the GP server and through the map services, which can be on other servers.

We are usually asked how much use can this application scale to handle? For the one server configuration, we have found that two concurrent users, or about 4-500 users per hour is where noticeable performance issues cannot be avoided. Up to that point, more processes can be added to handle usage. The multi-server configuration was designed much higher loads; and based on our experience we believe it could handle as many as 250,000-500,000 users per hour, with the database server ultimately being the bottleneck. The only recourse to scale further would be to employ a database replication strategy.

As more people submit votes, your database will grow. We have learned that has two significant implications. First is performance for the summary operations that occur when users change map extents. Second, as more data is collected and processing the summary operations takes more resources, it will also take longer to process votes when they are submitted. The database is the main bottleneck. We shipped this template based on a file geodatabase, which is considered fast and works well in a scenario that does not involve concurrent use. File Geodatabases are single user databases, and we rely on ArcGIS Server’s load balancing of the geoprocessing services to ensure system integrity. However, there is a hard limit to what that system can scale to. To get past that limit ArcSDE is required, and the reasoning is that a mulit-user DBMS can become the backend and your database.

## Considerations for Deploying your PollMap on a Production Server

Deploy, and then test, test, test. Never publicize a website that is intended for public consumption unless you have thoroughly tested it in the production environment. For example in most cases we used a 3-tier system. Our developers create on their development computers, and we ask them not to share those to us. We do our internal testing on what we call a staging server, which is internal, i.e., not available to the public. The reason for this is to ensure we learn exactly what files are required for deployment, having the goal of finding the minimum set.

Once that staging site works as expected, then we copy the data and maps to the production environment and set up the services. Back in our staging environment we edit our config.js file to use those new production services and we test. Then we deploy our website on the production server and test that. All of this is still happening with the test data. Once we verify the site is working correctly, we stop the services (so there are no locks on the data), we delete the test data and copy the zero vote count data, naming it to match our service requirements. Now the site is technically ready for a trial run.

Try voting and see if it works. If so, remove the test votes and copy the zero vote count data again. Now it’s time to have a few close colleagues or friends get your poll started. See the section about determining how people will learn about your map (above).

# How Voting Works

Here is how this application works:

1. A user loads the page and chooses their responses to the questions, enters a postal code, and clicks the Submit button.
2. The application sends that vote to the server which adds a record to the votes table. That record includes a date/time and a unique identifier for the voter along with setting the value of the CHOICExx field to one for each response the user selected.
3. Another process is also triggeredthat selects the current geography in the points table and increments the CHOICExx fields for each response the user selected. In both cases load is handled by ArcGIS server’s load balancing capabilities.
4. Next, the user is shown the collection of maps that correspond to each question. They can click on any map to see it in the ‘MegaMap’ pane.
5. When the MegaMap pane is loaded, panned, or zoomed, the resulting extent is sent to the server and the current votes within that extent are summarized and presented in the form of a pie chart. That means every time a user changes the extent an uptodate summary is produced.

# Analyzing the Results

Analyzing your results is done using the votes table and the Summary Statistics tool (Analysis toobox🡪Statistics). The idea is to Selection by Attributes to select rows in the votes table using queries like: CHOICE01 = 1. You can also use more complex queries like: CHOICE01 = 1 AND CHOICE05 = 1. It’s also possible to group responses within questions by using queries like: (CHOICE01 = 1 OR CHOICE02 = 1) AND CHOICE05 = 1. For Allocation question types, you will use a slightly different query, which is CHOICExx IS NOT NULL.

For radio question type polls, use the Summary Statistics tool by adding each of the CHOICExx fields as Statistics fields with SUM being the statistic type. You will not need Case fields or other types of statistics because all the values in the votes table are either null or 1.

For allocation question type polls, use the Summary Statistics tool by adding each of the CHOICExx fields by SUM, MEAN, STD, and COUNT.

# Additional Resources

Conducting a poll of any sort requires thought and expertise. [The American Association for Public Opinion Research](http://www.aapor.org) offers many useful resources for both experienced pollsters and those who may be conducting their first poll.

If you are considering adding functionality to your map, the [ArcGIS API for JavaScript](http://help.arcgis.com/en/webapi/javascript/arcgis/) will be needed.

We used the [dojo toolkit](http://dojotoolkit.org/documentation/) for our graphs, and this is the place to start if you wish to customize the graph for your map.

# Known Issues

Internet Explore 7 is not currently supported. The user is warned of this on the way in.

We have not secured services properly in this version of the template. It is possible for someone who is enterprising to hack and disrupt with your data, votes table, or zips\_winkel table. See the system architecture section above for advice.

On vote, you are limited to the postal codes in this particular database, which comes from GeoNames It’s actually pretty easy to add postal code points if you need to. Just copy the postcodes feature class, then run Delete features on the copy, Add empty dataset to ArcMap, Start Editing, and add the code(s) you need. Save your edits, and then use the Append tool to add your new codes.

Thanks for trying out this template. Send comments about this document to [jherries@esri.com](mailto:jherries@esri.com) and I’ll try to incorporate them quickly into the next version. We can also correspond about the template itself.

# Appendix A: Definition Query Examples

Each question type may require a different strategy for your layer’s definition queries. Here are examples for each question type:

## Radio (2 Choices)

**Regular layers**: CHOICE15 > CHOICE14

**Tied layers:** CHOICE14 = CHOICE15 AND CHOICE14 > 0

## Radio (3 or more Choices)

**Regular layers:**

The idea here is to detect when choice 12 has the highest total among all these fields:

CHOICE12 > CHOICE08 AND CHOICE12 > CHOICE09 AND CHOICE12 > CHOICE10 AND CHOICE12 > CHOICE11 AND CHOICE12 > CHOICE13

**Tied layers:**

The idea here is to detect all possible ties between the following fields: CHOICE08, CHOICE09, CHOICE10, CHOICE11, CHOICE12, and CHOICE13:

(CHOICE08 > 0 AND (CHOICE08 = CHOICE09 OR CHOICE08 = CHOICE10 OR CHOICE08 = CHOICE11 OR CHOICE08 = CHOICE12 OR CHOICE08 = CHOICE13)) OR (CHOICE09 > 0 AND (CHOICE09 = CHOICE10 OR CHOICE09 = CHOICE11 OR CHOICE09 = CHOICE12 OR CHOICE09 = CHOICE13)) OR (CHOICE10 > 0 AND (CHOICE10 = CHOICE11 OR CHOICE10 = CHOICE12 OR CHOICE10 = CHOICE13)) OR (CHOICE11 > 0 AND (CHOICE11 = CHOICE12 OR CHOICE11 = CHOICE13)) OR (CHOICE12 > 0 AND CHOICE12 = CHOICE13)

Or, formatted for readability (do not copy and paste this section):

(CHOICE08 > 0 AND (CHOICE08 = CHOICE09 OR CHOICE08 = CHOICE10 OR CHOICE08 = CHOICE11 OR CHOICE08 = CHOICE12 OR CHOICE08 = CHOICE13))

OR (CHOICE09 > 0 AND (CHOICE09 = CHOICE10 OR CHOICE09 = CHOICE11 OR CHOICE09 = CHOICE12 OR CHOICE09 = CHOICE13))

OR (CHOICE10 > 0 AND (CHOICE10 = CHOICE11 OR CHOICE10 = CHOICE12 OR CHOICE10 = CHOICE13))

OR (CHOICE11 > 0 AND (CHOICE11 = CHOICE12 OR CHOICE11 = CHOICE13))

OR (CHOICE12 > 0 AND CHOICE12 = CHOICE13)

## Allocation

"CHOICE04" > "CHOICE01" AND "CHOICE04" > "CHOICE02" AND "CHOICE04" > "CHOICE03" AND "CHOICE04" > "CHOICE05" AND "CHOICE04" > "CHOICE06"

# Appendix B: Config.js question type examples

Each question type requires a different use and subset of the possible variables. Here are examples of what is required for each of the question types supported at version 2.01

## Radio (2 Choices)

questions : [{

type : "radio", //must match config.app.questionMode

label : "Gender: ", //for the voteWindow.html

title : "Gender", //for the multi-map view

subtitle : "About your most recent…" //Optional

active : true,

values : [{

label : "Male",

value : "CHOICE01", // the field name in the votes table

title : "Male" //for the pie chart

},{

label : "Female",

value : "CHOICE02",

title : "Female"

}]

},{

## Autocomplete (*n* Choices)

This option has not been tested yet, as soon as it has, we will add documentation.