EmpowerENGINE

Dynamic Raster-Tiles

Helping progressives view and understand their geographic data.

Julie Goldberg
julie@empowerengine.com
Noah Glusenkamp
noah@empowerengine.com

Outline

1. Problem Space & Demo
2. Our Stack
3. Dynamic Tile Challenges
4. 3 Lessons Learned “Caching is your Friend”
5. Q&A

Campaign Data is hot

2012 ELECTION

Inside the Secret World of the Data Crunchers Who Helped Obama Win

Data-driven decisionmaking played a huge role in creating a second term for the 44th President and will be one of the more closely studied elements of the 2012 cycle

By Michael Scherer @michaelscherer | Nov. 07, 2012 | 273 Comments

The Atlantic

20 Percent of American Workers Feel Overqualified

Image Apples, Okto

POLITICS | BUSINESS | TECH | ENTERTAINMENT | HEALTH | EDUCATION | SEXES | NATIONAL | GLOB

Special Reports In Focus Events E-books Newsletters JUST IN 1 in 5 U.S. Workers: I'm Too Educated for My Job

When the Nerds Go Marching In

ALEXIS C. MADRIGAL | NOV 16 2012, 7:00 AM ET

18 Like

How a dream team of engineers from Facebook, Twitter, and Google built the software that drove Barack Obama's reelection

POLITICO

Sign in / Register | Mobile | POLITICO Jobs

POLITICO Home

2012 44 CONGRESS BLOGS ARENA OPINION POLICY VIDEO PHO

Navigate: POLITICO | Obama's data advantage

Obama's data advantage

c|net

English Reviews News Download CNET TV How To Deal

CNET › News › Politics and Law › Why Romney's Orca killer app beached on ...

Why Romney's Orca killer app beached on Election Day

Project Orca was supposed to give the Romney campaign a technical advantage over Obama on election day. It got harpooned instead.

ars technica

MAIN MENU MY STORIES: 25 FORUMS SUBSCRIBE JOBS

TECHNOLOGY LAB / INFORMATION TECHNOLOG

Built to win: Deep inside Obama's campaign tech

How Obama's tech team built a "force multiplier" with Amazon and a narwhal.

by Sean Gallagher - Nov 14 2012, 8:15am PST

BIG DATA CLOUD DEVELOPMENT 189

Campaigns are data-intensive

- Efficiently target and track communication with voters: phone calls, door knocks, mailings, emails, etc.
- Success is measured daily: how many contacts, attempts, 1s and 5s.
- There's now a campaign job title of "Data Director"
- All campaign data is tracked by district.

Why maps?

- Help us see patterns in data that are otherwise hard to see.

- District lines are arbitrary but significant in politics
  - parties organize along them
  - polling locations (most states) determined by them
  - ballots are tallied within them

- Data is a motivating factor for volunteers and donors. Maps help communicate data and thereby **engage supporters.**

- No good solution right now. The wheel is reinvented every campaign.

The Goal

An easy and fast way to generate and securely share maps of demographic, electoral, and campaign data at a high level of detail (precincts/blocks).

Why a custom web-application?

There are a lot of great tools for web-maps (Mapbox, CartoDB, Google Fusion tables) so why build our own?

1. Security - obfuscated links aren't good enough. Full user authentication system is needed.

2. Time & Skill - you still need to know more than a little about data and mapping to use the above tools. Campaigns don't have that expertise, especially down-ballot (e.g. state legislature, city council, etc.).

3. Cost - The high granularity and scale of what we want to map prices us out.

   CartoDB Enterprise L Instance = $2,499 / mo for 100 GB ~ $25 / GB / mo

   Mapbox Premium = $499 / mo for 30 GB **of tile data**

   The census has GB's of data we want to show. Add precincts and voter-file data = a lot.

   Tile disk requirements increase exponentially with zoom level.

      WA State at Zoom Level 17 > 1 GB

      WA State at Zoom Level 19 > 100 GB

Tiles!

- Browsers just can’t handle rendering very complex geometries as vectors. (although d3 is helping push the limits here, see OSM id Editor)

- We want to map many complex geometries like census blocks and precincts.

- We want to map these geometries at large extents (even all precincts in King County is pretty taxing and we want to be able to do things like all precincts in the state).

The Problem

Raster tiles are usually pre-generated (Mapbox) or statically specified (Tilestache).

We want it both ways: the speed of raster tiles with the dynamic nature of vectors.

Demo

http://www.empowerengine.com

Our Stack

Django app

- Interface to login, view allowed maps, create/edit/share maps (with paid account), purchase account, etc.

Postgres/postgis

- Stores district shapes and info about districts (e.g. Obama's votes per precinct)

Leaflet

- Django templates use this library to request and display map tiles

Tilestache + Mapnik

- Tilestache server serves up tiles for requested map. Hacked to allow unlimited number of maps.
- Tilestache provider uses Mapnik to render new tiles (color tiles and utf grids) as needed. Caches them on s3.
- Uses django settings and models to pull appropriate data.

Problem: Need Dynamic Tiles

Tilestache designed for a fixed set of “data layers” to make a fixed set of tiles. These are specified in the config.

```
{
  "cache": {"name": "Test"},
  "logging": "debug",
  "layers":{
    "district_value":{
      "provider": {
        "class": "provider.DistrictValueColorProvider"
      },
    },
    "district_attributes":{
      "provider": {
        "class": "provider.DistrictUTFGridProvider"
      }
    }
  }
}
```

Tilestache Providers

Documentation: *“A Provider is the part of TileStache that stores static files to speed up future requests.”*

Providers can be an mbtiles file, a url template, a proxy, etc.

You can write your own provider class that implements `renderArea()`.

Problem: Need Dynamic Tiles

- Layers (maps) are added to tilestache through the config.
- BUT we cannot recreate the tilestache configuration every time a map is created.
- Our Tilestache “provider” needs to know what map it is pulling data for.
  - Colorize each map based on different “district attributes” (Obama vote count, 74 win percent, % Asian, etc.)
  - Populate UTF grid based on the same information
  - Map specifies scope and colors

Solution: Custom Tilestache TileServer

- “Decorated” WSGITileServer, so it expects ids in its url.
- `gunicorn --pythonpath $PWD "TileServer: DynamicLayersTileServer('tilestache.cfg', 'foo', 'bar')"`
- Passes the value of “foo” as 25 and “bar” as 34 to provider and to cache.
- http://mytileserver.com/district_value/foo/25/bar/34
- Effectively dynamically updates the config every time it’s called.
- Available at https://gist.github.com/JulieGoldberg

```
class DynamicLayersTileServer():
     def __init__(self, config, *parameter_names):
          self.tile_server = WSGITileServer(config)
          self.parameter_names = parameter_names
          #regex to parse out the parameter value and the path tilestache expects if it starts
with the parameter name
          parameter_regex = ""
          for parameter_name in parameter_names:
               regex_variable_name = self.__get_regex_variable_name__(parameter_name)
               parameter_regex += "\/" + parameter_name + "\/(?P<" + regex_variable_name +
">\w+)"
          parameter_regex += "(?P<tilestache_path_info>\/.+$)"
          self.parameter_regex = re.compile(parameter_regex)

     def __call__(self, environ, start_response):
          path_info = environ['PATH_INFO']
          parsed_path = self.parameter_regex.match(path_info or '')
          if parsed_path:
               self.__update_config_with_dynamic_parameter_values__(self.tile_server.config,
environ, parsed_path)
          response = self.tile_server.__call__(environ, start_response)
          return response

     def __get_regex_variable_name__(self, parameter_name):
          return parameter_name + "_value"

     def __update_config_with_dynamic_parameter_values__(self, config, environ, parsed_path):
          #Set PATH_INFO to what it would have been if we hadn't added our dynamic param
          environ['PATH_INFO'] = parsed_path.group("tilestache_path_info")
          #set our dynamic parameter on the provider for this layer
          layer_name = splitPathInfo(environ['PATH_INFO'])[0]
          layer = config.layers[layer_name]
          parameter_values = {}
          for parameter_name in self.parameter_names:
               regex_variable_name = self.__get_regex_variable_name__(parameter_name)
```

Solution: Custom Provider Classes

- Much more standard practice to write providers than tileservers.
- Just have to write `renderArea()` function, which uses mapnik.
- Based on `Mapnik.ImageProvider` and `Mapnik.GridProvider` classes. Unfortunately, those classes do a lot in init, so I couldn't subclass. Write modular code and architect carefully!
- My custom tileserver calls `set_tileserver_parameters(self, parameter_values)` at the start of each request, so we know what map we're rendering before we get to `renderArea()`
- We use django models to figure out what data to pull and (for images) how to colorize it.

Caching is Your Friend!

Tip 1: Only compute all the data for a map once.

- Postgres stores geometries in a different way than fixed-size data, so it's slower to access.
- When Mapnik renders a tile, it has to determine which districts (if any) it should pull data for.
- Once it has the districts, it needs to get the appropriate data (if any).
- Using joins is pricy, especially when your districts table is huge.
- **When we create a map, we copy all the data for it into its own cache table. All data for map 39 is in "*map_caches.map_39*"**

Caching is Your Friend!

Tip 2: Avoid on-the-fly spatial joins.

- Once you specify map scope and granularity, figuring out what districts are on the map is a spatial query. That's pricey.
- We pre-compute a "lookups" table that specifies all the precincts/blocks in every city/county/LD.
- We add to it when we add new districts.
- **When we add a map, we use these lookup tables to create that map's cache table.**

Caching is Your Friend!

Tip 3: Avoid on-the-fly projection transformations.

- Transforming district projections is pricy.
- Much of our data arrives as NAD83/Census projection `(SRID=4269)` and that's how we store it.
- Need to match baselayer tiles that are in web mercator `(SRID=3857)`
- We used to transform all the districts when making the cache table, but that slowed initial map creation way down.
- **We have two geometric columns on our districts table: "`geometry`" and "`map_cache_geometry`"**

EmpowerENGINE

Q&A

Julie Goldberg
julie@empowerengine.com
Noah Glusenkamp
noah@empowerengine.com