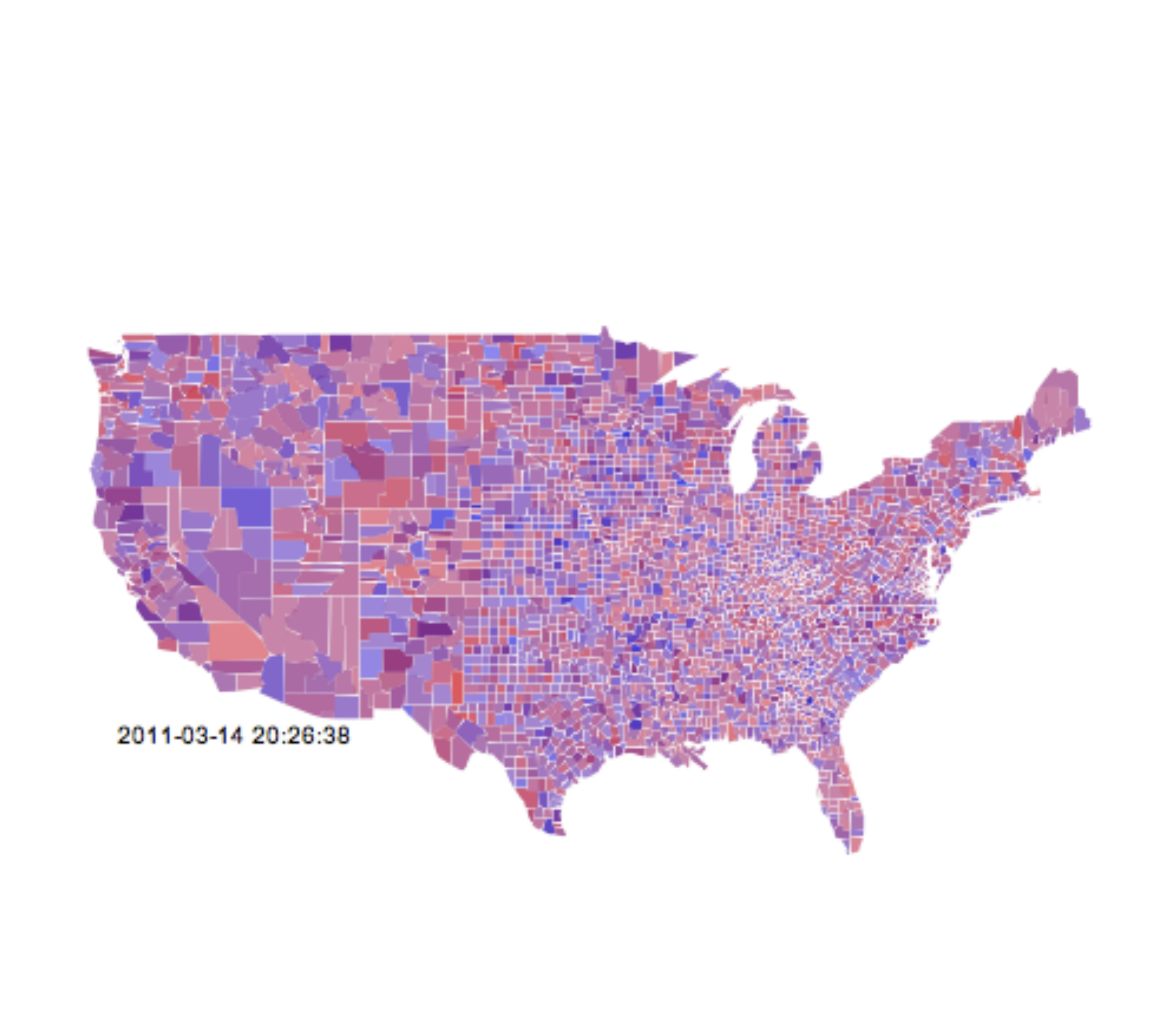
A Few of My Favorite Things

A Modern Data Environment
Byron Ellis

About Me...



- Director of Analytics at adBrite (an ad exchange)
- R user since early 2000
- Before that it was LispStat
- More about stacks than specific applications

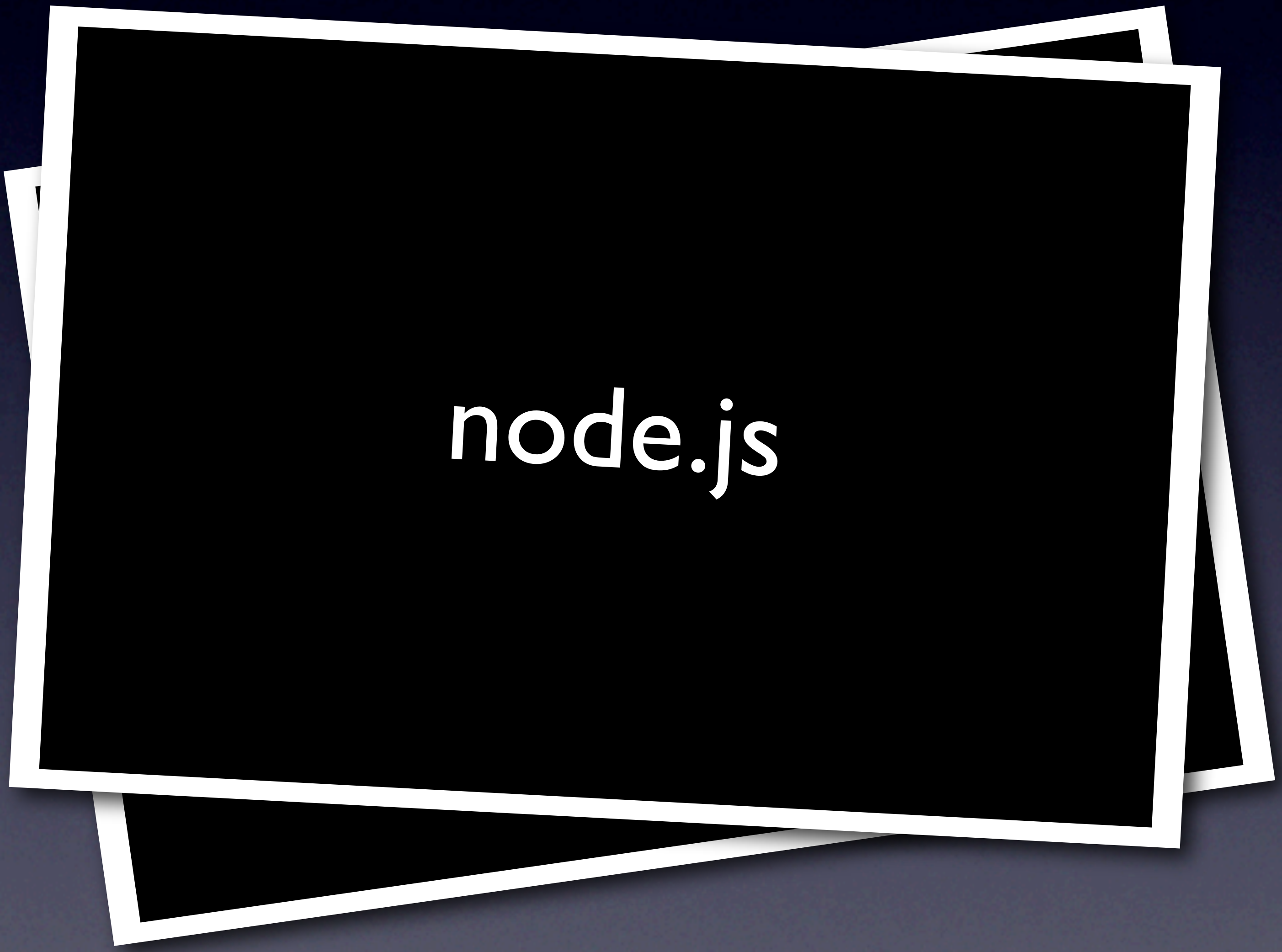
What about the map?

- Dynamically rendered in a WebView
  - No Flash, 84 lines of Javascript and HTML
- Looks suspiciously like R (it’s running the background)
- Wanted to keep the interfaces very light
  - Simple data formats (tab delimited, JSON)
  - Exploration, not architecture

```
x = table(read.delim
("http://localhost:3000/
df/elections"))

>head(x)

                  party
county             Democrat Republican
  alabama,autauga         2          0
  alabama,baldwin         0          2
  alabama,bibb            1          0
  alabama,blount          1          1
  alabama,calhoun         0          1
  alabama,cleburne        0          1
```



node.js

- Chrome’s V8 Javascript Engine
- “Continuation”-style convention for blocking events.
- CommonJS module system.
- Young, but with a growing ecosystem.

node.js

```
http.createServer(function(req,res) {
    var uri   = url.parse(req.url,true);
    var parts = uri.pathname.split("/");
    var cmd   = parts[1] || "";
    if(cmd == "obs") {
    d.observe(parts
[2],uri.query,function(err) {
            if(err) {
                    res.writeHead(500);
                    res.end();
            } else {
                    res.writeHead(200);
                    res.end();
            }
    });
    } else if(cmd == "df") {
            d.report(parts
[2],uri.query,res);
    } else if(cmd.substring(0,3) == "dev")
            graphicsCommand(cmd,parts[2] ||
null,uri,req,res);
    else
            serveStaticFile(uri.pathname,res);
}).listen(3000);
```

MongoDB

- Key-Value store
- “Document” oriented values
- Secondary indexing on arbitrary document keys
- Javascript query engine
- Some Map/Reduce capability.

MongoDB

```
db.metadata.findOne
({_id:”elections”});
```

```
{ "_id" : "elections",
"cat" : { "county" : 1,
"party" : 1 }, "columns" :
[ "county", "party" ] }
```

```
db.data.find({d:”elections”});
```

```
{ "d" : "elections", "t" :
{ "floatApprox" :
1300126097341 }, "r" :
{ "county" :
"kentucky,daviess",
"party" : "Republican" },
"_id" : ObjectId
("4d7e5991f30562bf1000
0418") }
```

node.js

```
var cols = [];
var meta = {};
var rec  = {};
for(var i in value) {
      var name = i.split(".").join
("_");
      cols.push(name);
      var num  = 1*value[i];
      rec[name] = isNaN(num) ?
value[i] : num;
      if(isNaN(num)) meta
["cat."+name] = 1;
}

dataColl.insert({d:df,t:new
(Date)().getTime(),r:rec});

metaColl.update({_id:df},
{$set:meta,$addToSet:
{columns:{$each:cols}}},
{upsert:true});
```

node.js

```
coll.find({d:id},
function(err,cursor) {
cursor.each(
function(err,x) {
  if(err || x == null) {
     sink.end();
  } else {
     var line = [];
     for(var i in order) {
       line.push(x[order[i]]);
     }
    sink.write(line.join(“\t”)
      +”\n”);
  }
});
});
```

Prediction APIs

- “observe” function implements collection of training data
- Add a little logic/indexing to the observation collection to get time batches
- Fit models and POST them back to the node instance.
  - Lexically scoped anonymous functions

```
with(list(x=x+1),{
a = 127 + 128*log((x))/log(max(x))
shade = rgb(
ramp(
sweep(x,1,rowSums(x),"/")[,2]
),
alpha=a,maxColorValue=255)
map("county",names(shade),fill=TRUE,boundary=FALSE,lty=0,interior=FALSE,col=shade)
text(-110,30,Sys.time())})
```

```
library(RGraphicsDevice)
postCmd = function(cmd,from) {
        new.attr = to.attr(from)
        if(length(new.attr) > 0)
            cmd$attr = new.attr
        con = socketConnection(port=3000,host="localhost")
        j = toJSON(cmd)
        l = sprintf("POST /dev.cmd/%s HTTP/1.1\nContent-Type: application/json\nContent-Length:%d\n\n%s",.Gfx.Id,nchar(j),j)
        writeLines(l,con)
        close(con)
    }
```

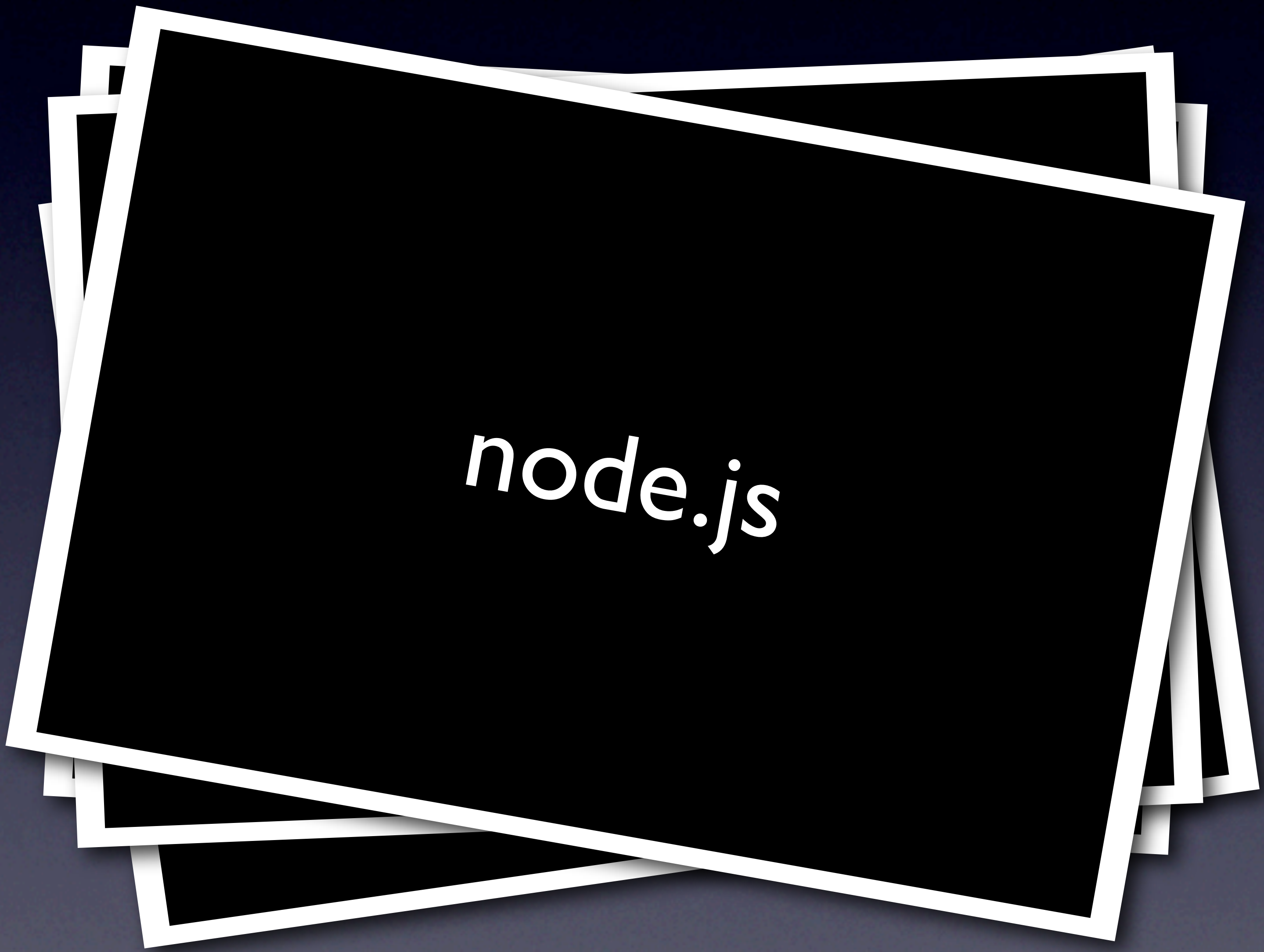
```
dev@polygon =
function(n,x,y,gcontext,dev) {
      postCmd(list(
        type="polygon",
        x=round(x[1:n],2),
        y=round(y[1:n],2)),
        gcontext)
}
```

node.js

```
var buffer = "";

source.addListener("data",
function(newData) {
      buffer += newData;
});

source.addListener("end",
function() {
      if(buffer.length > 0)
           insert(JSON.parse(buffer));
      cb(null,count);
});
```



http://localhost:3000/dev.js/elections

```
{"_id":"elections",
width:500,
height:500,
cmds:[
{type:”polygon”,
 x:[...],
 y:[...],
 attr:{fill:”#6600999A”}},
...
]}
```

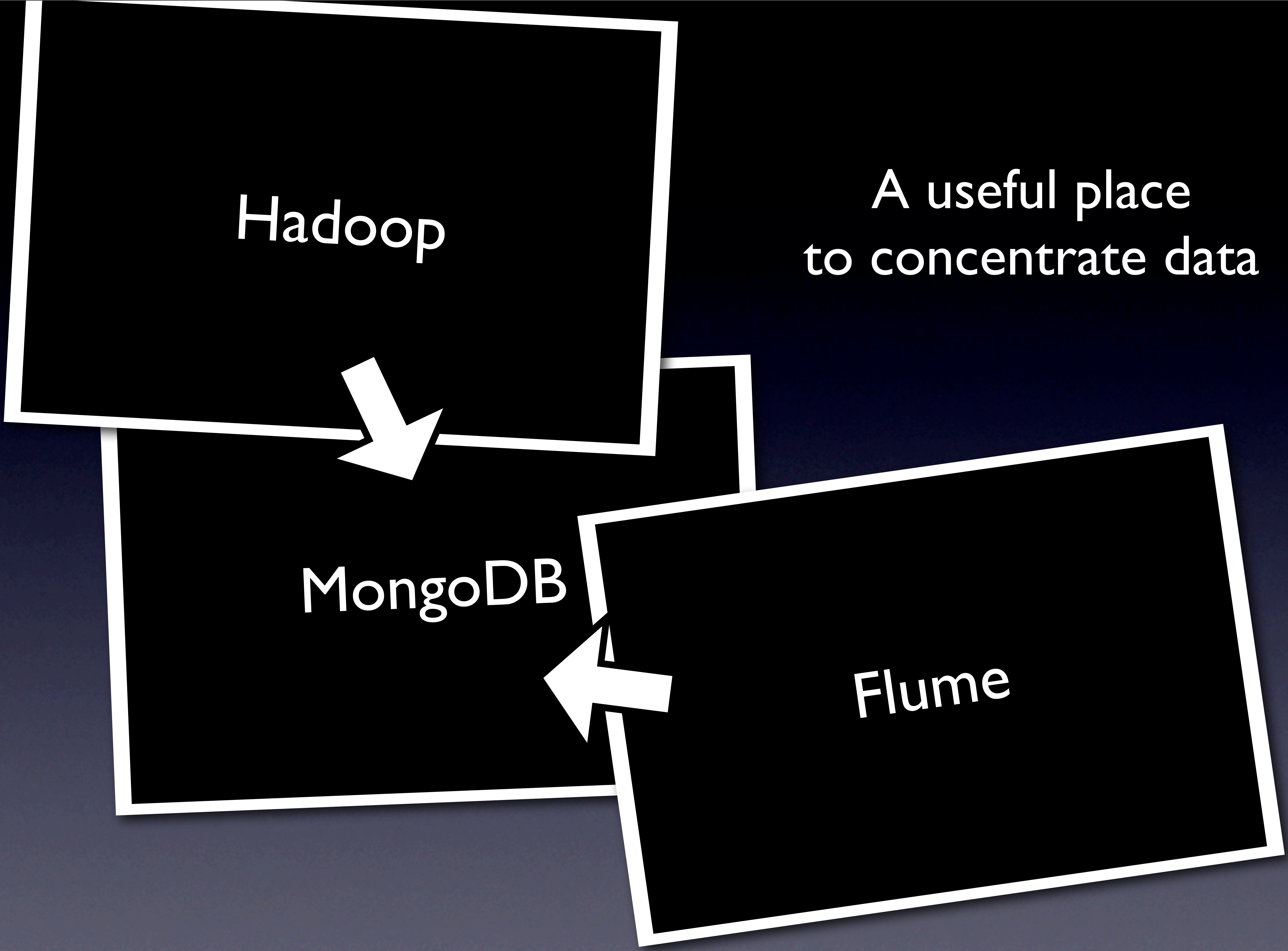
jQuery + Raphaël

```
for(i in data.cmd) {
var cmd = data.cmd[i];
mergeAttr(cmd.attr);
...
if(cmd.type == "polyline" ||
   cmd.type == "polygon")
{
 var x = cmd.x;
 var y = cmd.y;
 var path = [];
 for(var i=0;i<x.length;i++) {
  path.push(
    (i==0?"M":"L")
    +x[i]+","+(y[i]));
}
r.path(path.join("")).attr(attr);
}
}
```

Collabotron 3000!

- Share links to your graphics device
  - Other folks don’t need R to see it
- Comments are easy to implement
- Hook it up to R-node and you have (another) web interface to R.
- Record pages—Gapminder minus Flash.

A useful place
to concentrate data



Hadoop

MongoDB

Flume

Summary

- MongoDB and node.js maintain the model
  - Acts as the model with which controllers interact
- Transform the data so that it is convenient for the tool
  - Going to HTML? JSON
  - Going to R? Delimited files.
- Keep it simple.

http://github.com/byronellis/mixnmatch