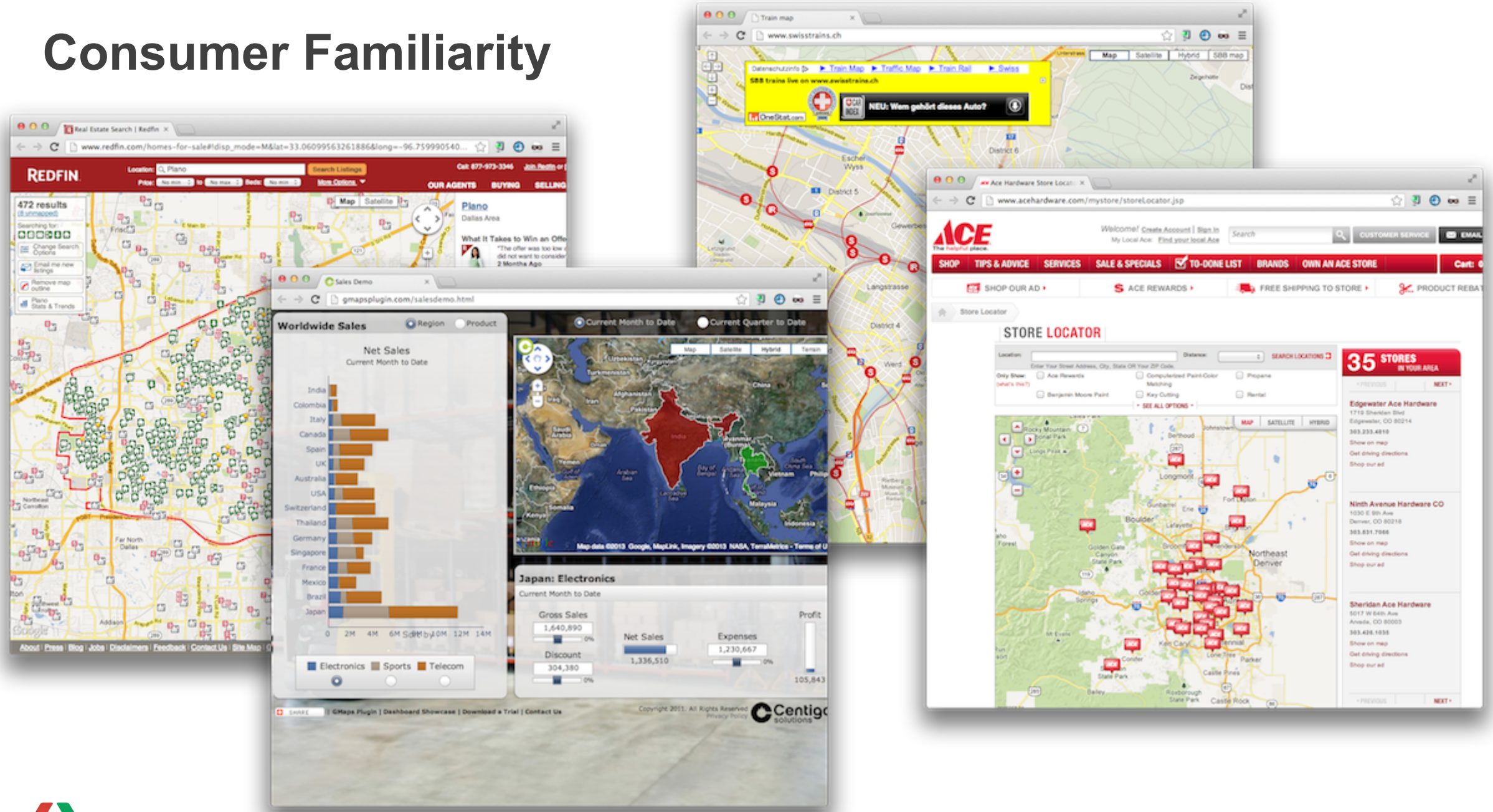

Google
Developers

Off the Grid:
Going Offline with Maps and Earth

Sean Maday, Product Manager
Avnish Bhatnagar, Technical Solutions Engineer

Consumer Familiarity



Portable Use Cases

Emergency Response

Utility, Construction & Energy

Research and Outreach

Defense & Public Safety

Portable Overview

Product and Architecture

Portable

The familiarity of Google Maps & Google Earth Offline

Builds for Windows, Mac, Linux

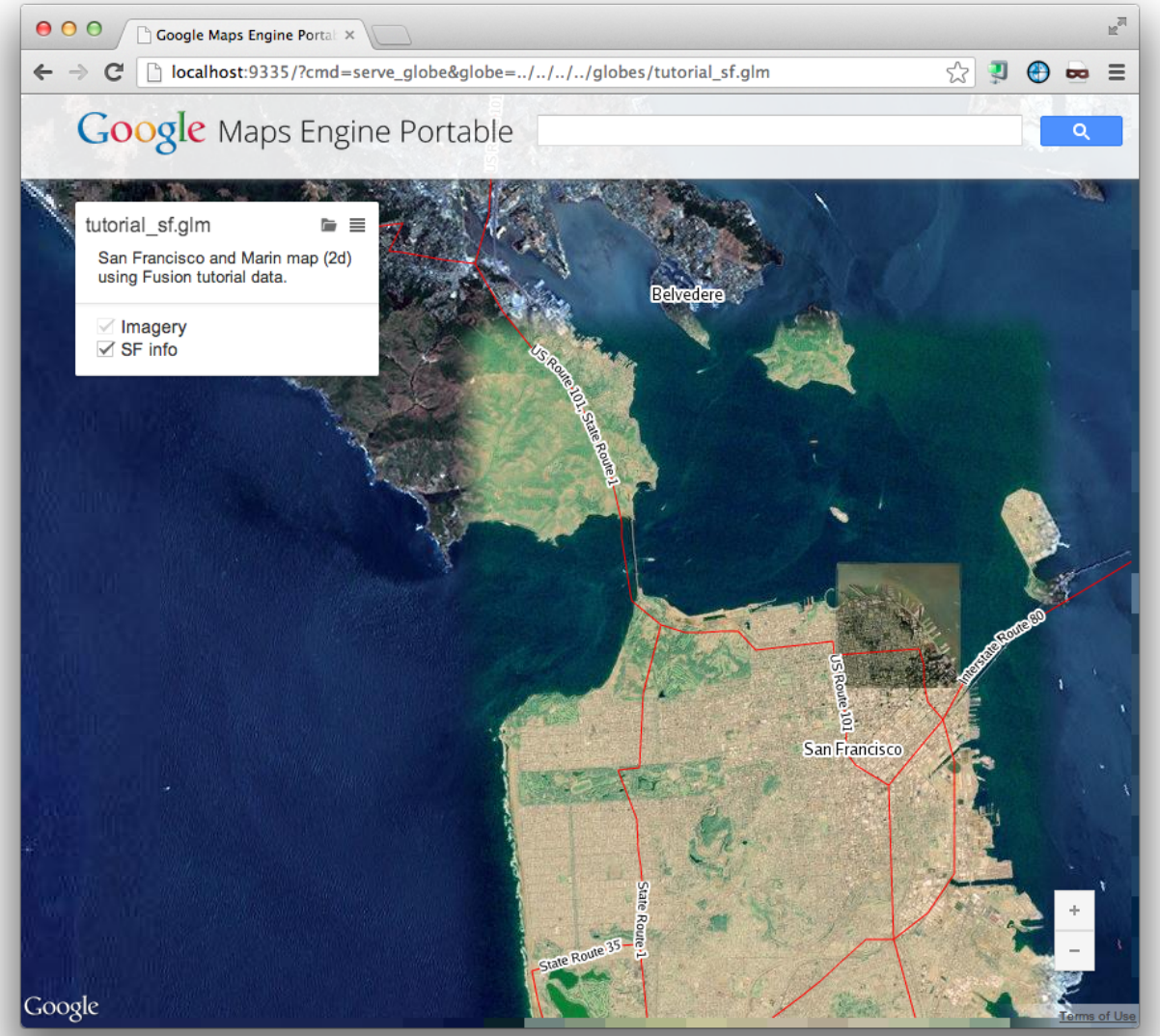
Android access with the Maps for Business Mobile App

Portable

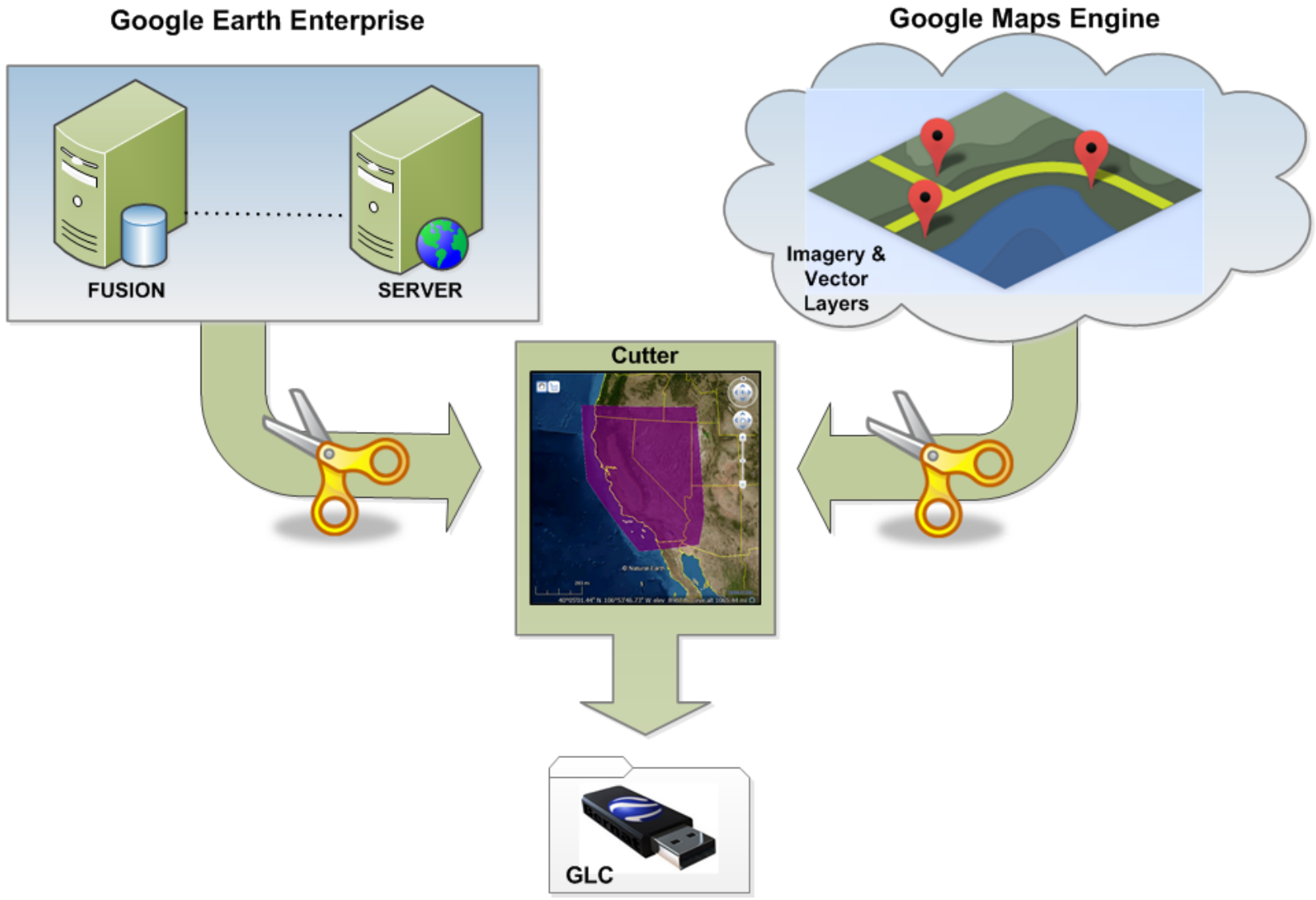
Lightweight Tornado web server

Serve 2D Maps or 3D Earth Portables

Broadcast to share with up to 10 users



Architecture & Workflow - Cutting

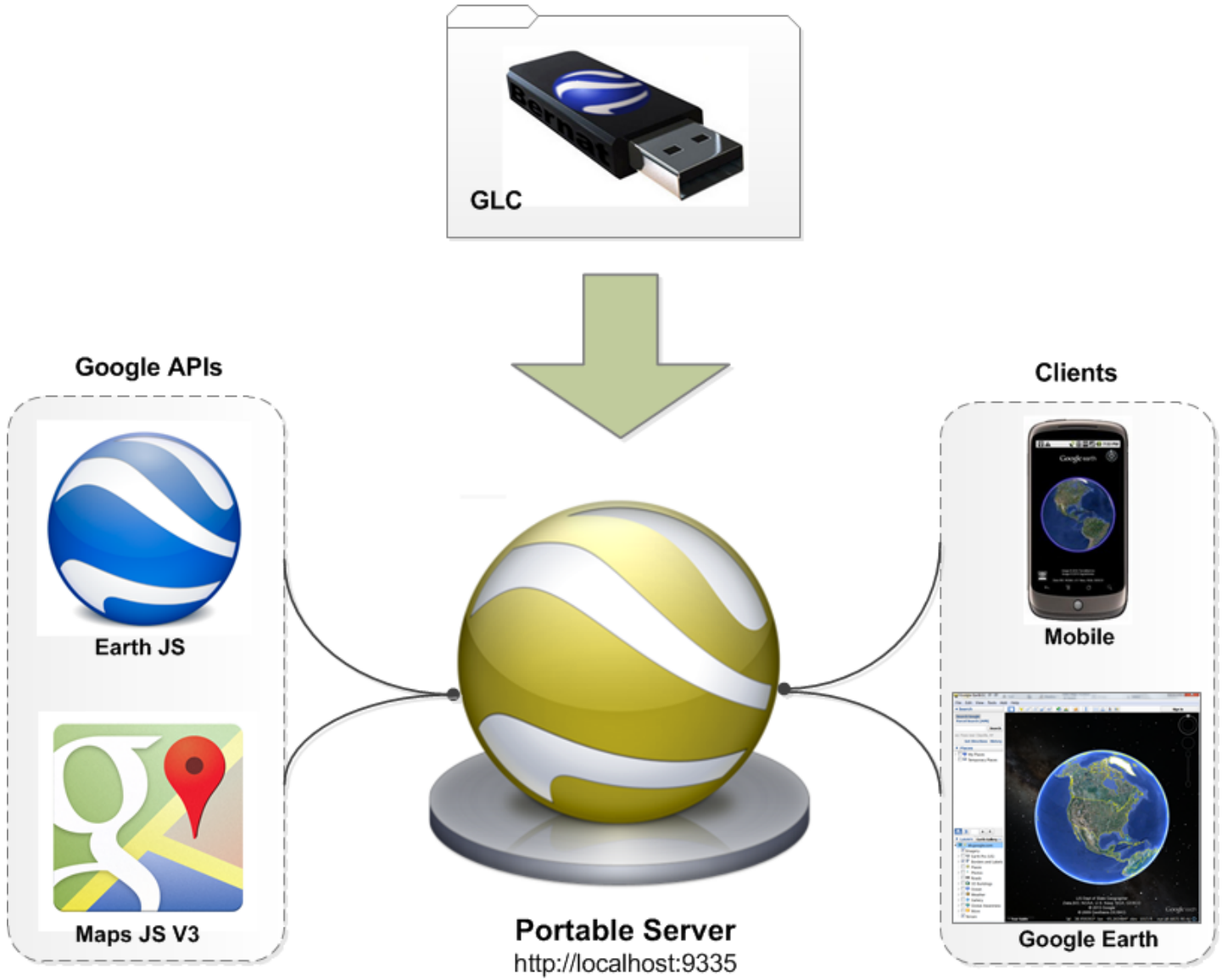


A Portable "Cut"

Self contained 2D Map or 3D Globe

- Single binary file
- Carries a GLC extension
- Encompasses legacy GLM and GLB file formats
- Includes custom POI search indexes

Architecture & Workflow - Serving



Creating a Portable Map/Globe

Cutting Concepts

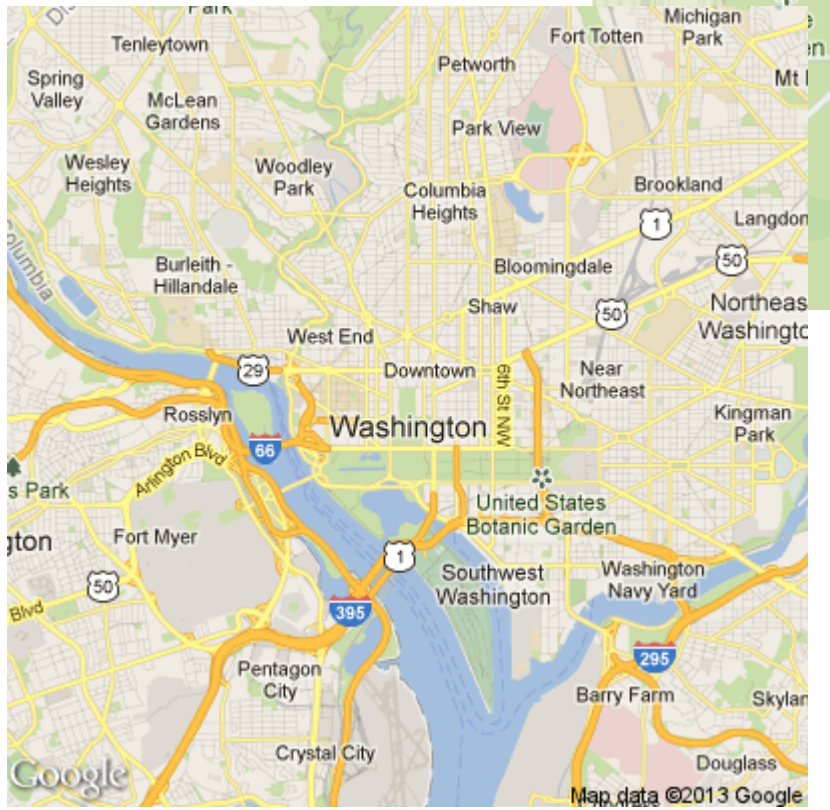
Define a cut area

- Single and multi polygon support
- Specify using a simple KML

Set cut parameters

- Zoom level inside the polygon
- Zoom level outside the polygon

Zoom Level 18



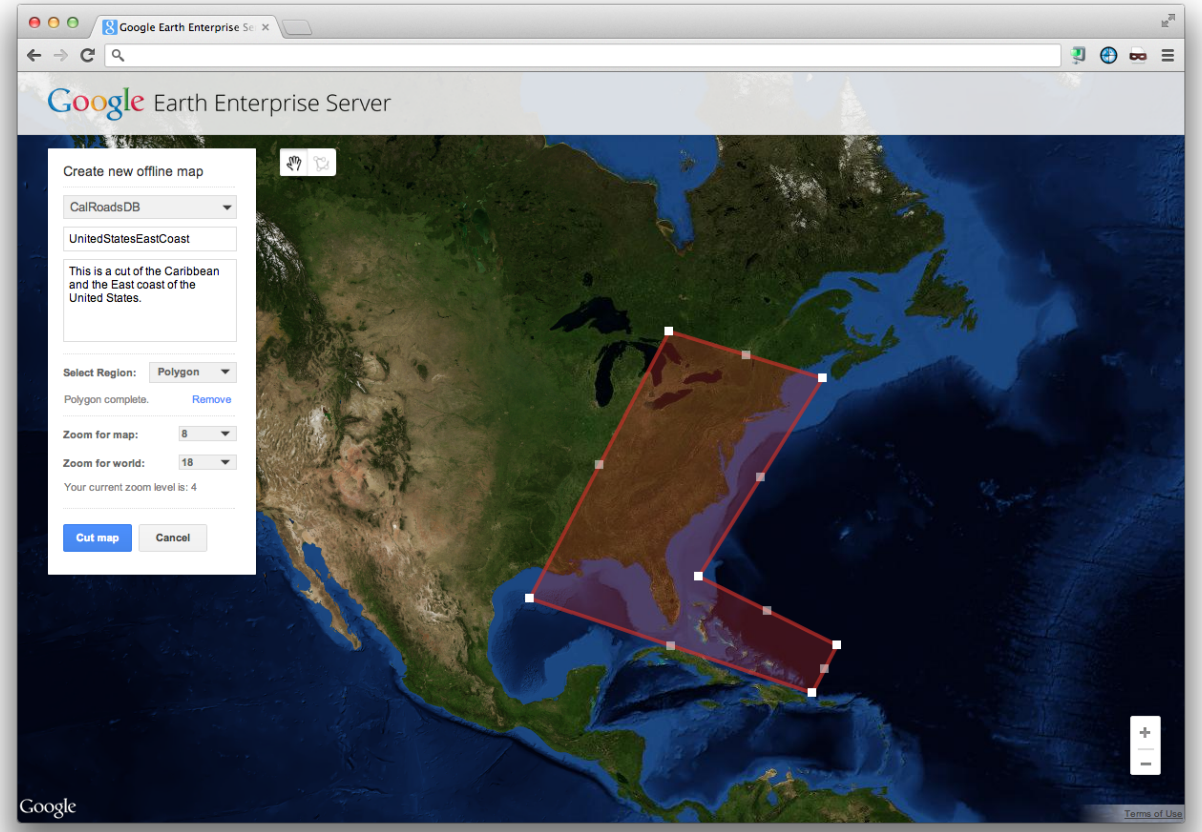
Zoom Level 12

Cutting a Portable from GEE

Creating a Portable with Google Earth Enterprise (GEE)

- HTML5 user interface
- Command line utilities

Support for up to 5 imagery layers in Earth

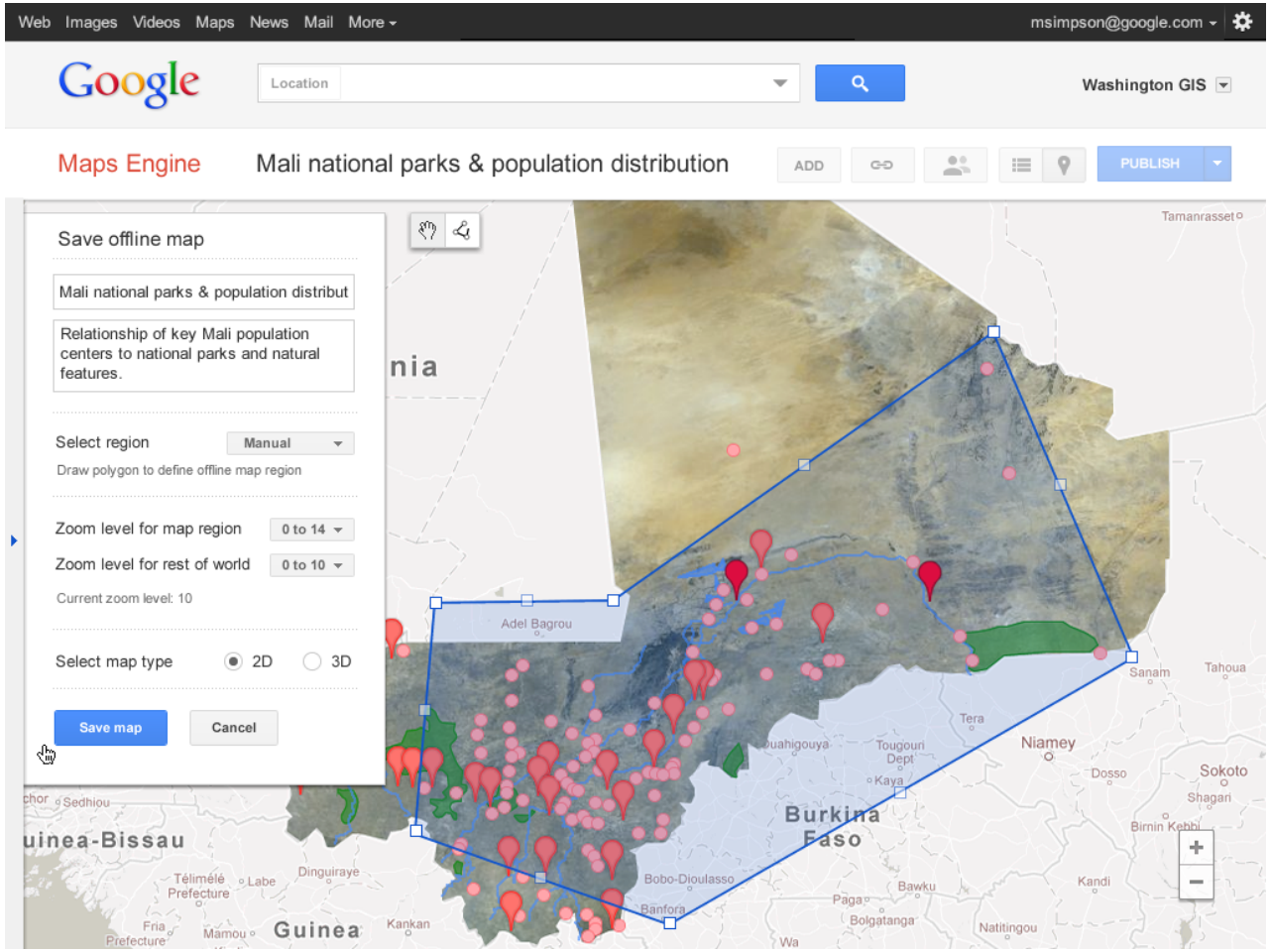


Cutting a Portable from GME - Coming Soon

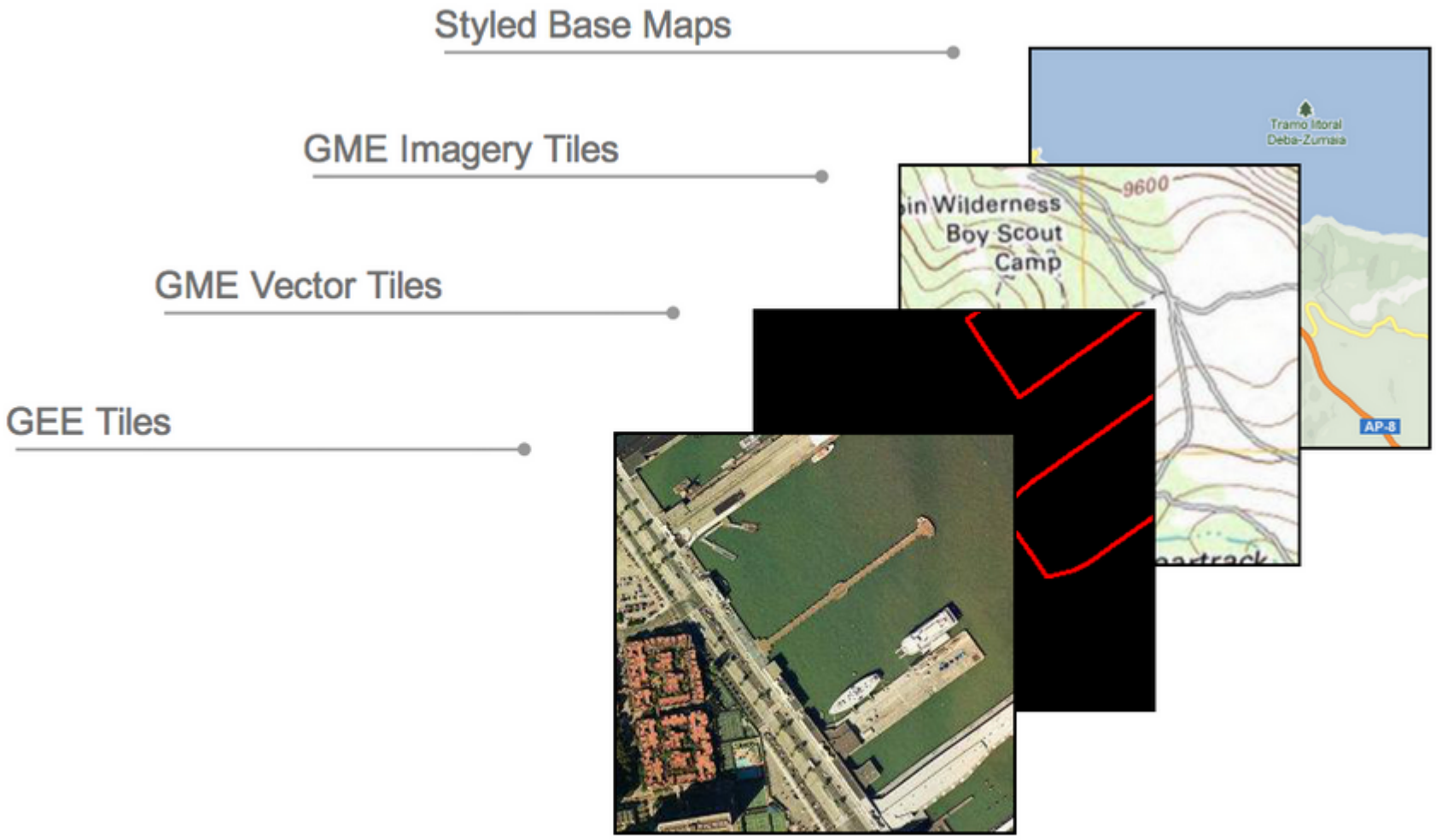
Creating a Portable with Google Maps Engine (GME)

- Browser-based user interface
- API for programmatic access

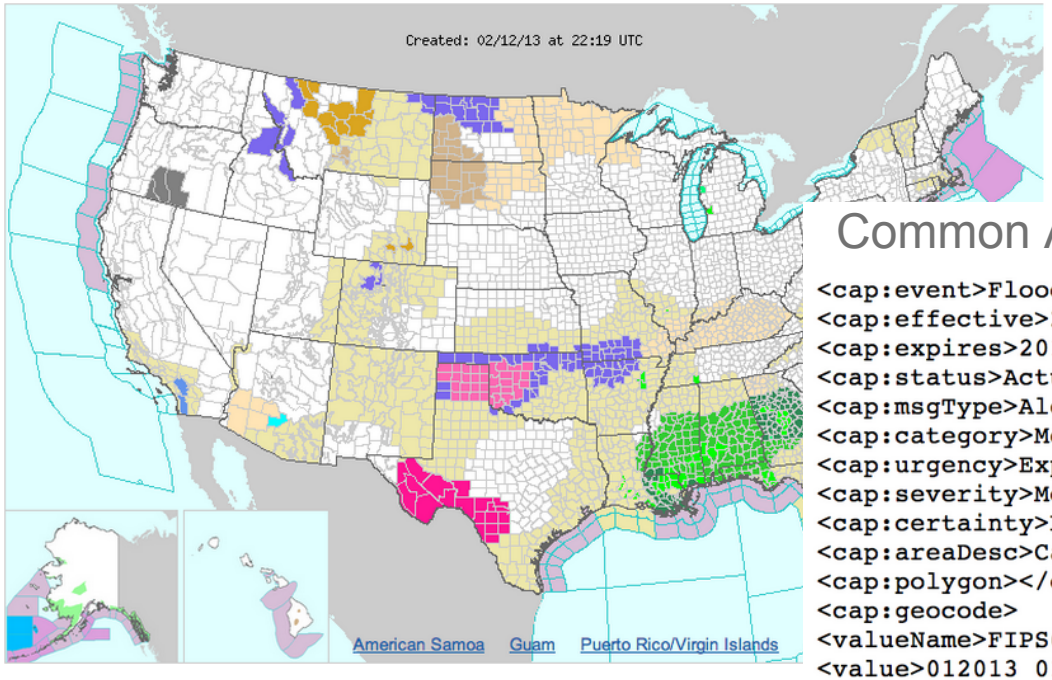
Launching to trusted testers later in 2013



Composite Portable Solutions

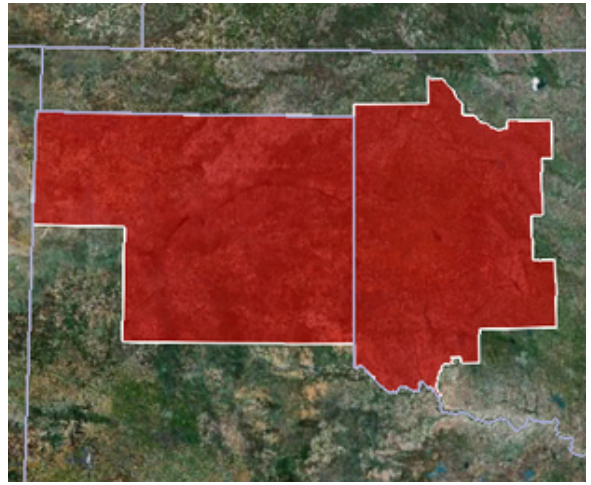


Programmatic Cutting



Common Alerting Protocol (CAP)

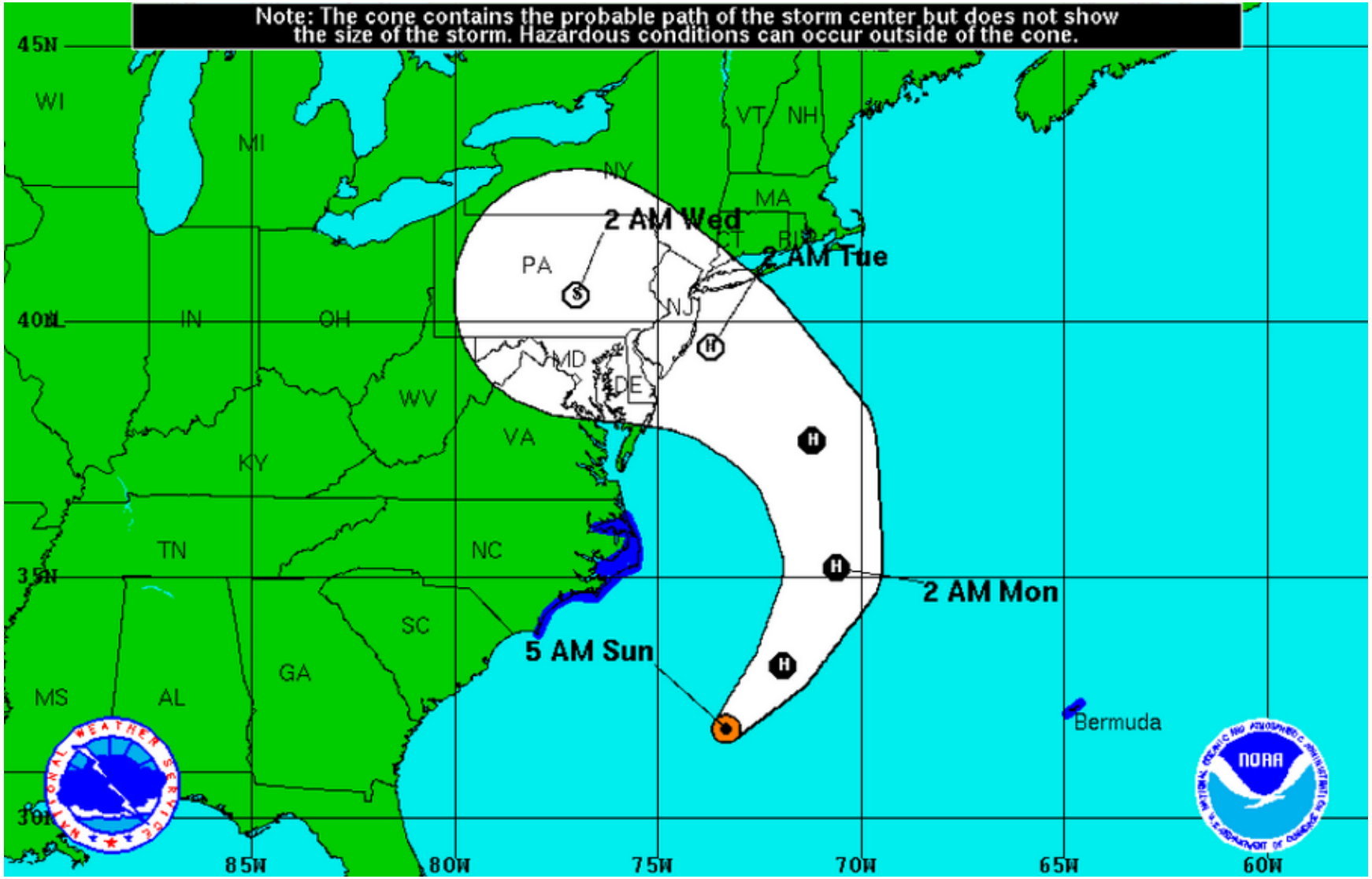
```
<cap:event>Flood Warning</cap:event>
<cap:effective>2013-02-13T09:27:00-05:00</cap:effective>
<cap:expires>2013-02-16T15:34:00-05:00</cap:expires>
<cap:status>Actual</cap:status>
<cap:msgType>Alert</cap:msgType>
<cap:category>Met</cap:category>
<cap:urgency>Expected</cap:urgency>
<cap:severity>Moderate</cap:severity>
<cap:certainty>Likely</cap:certainty>
<cap:areaDesc>Calhoun; Franklin; Gulf; Liberty</cap:areaDesc>
<cap:polygon></cap:polygon>
<cap:geocode>
<valueName>FIPS6</valueName>
<value>012013 012037 012045 012077</value>
```



Programmatic Cutting

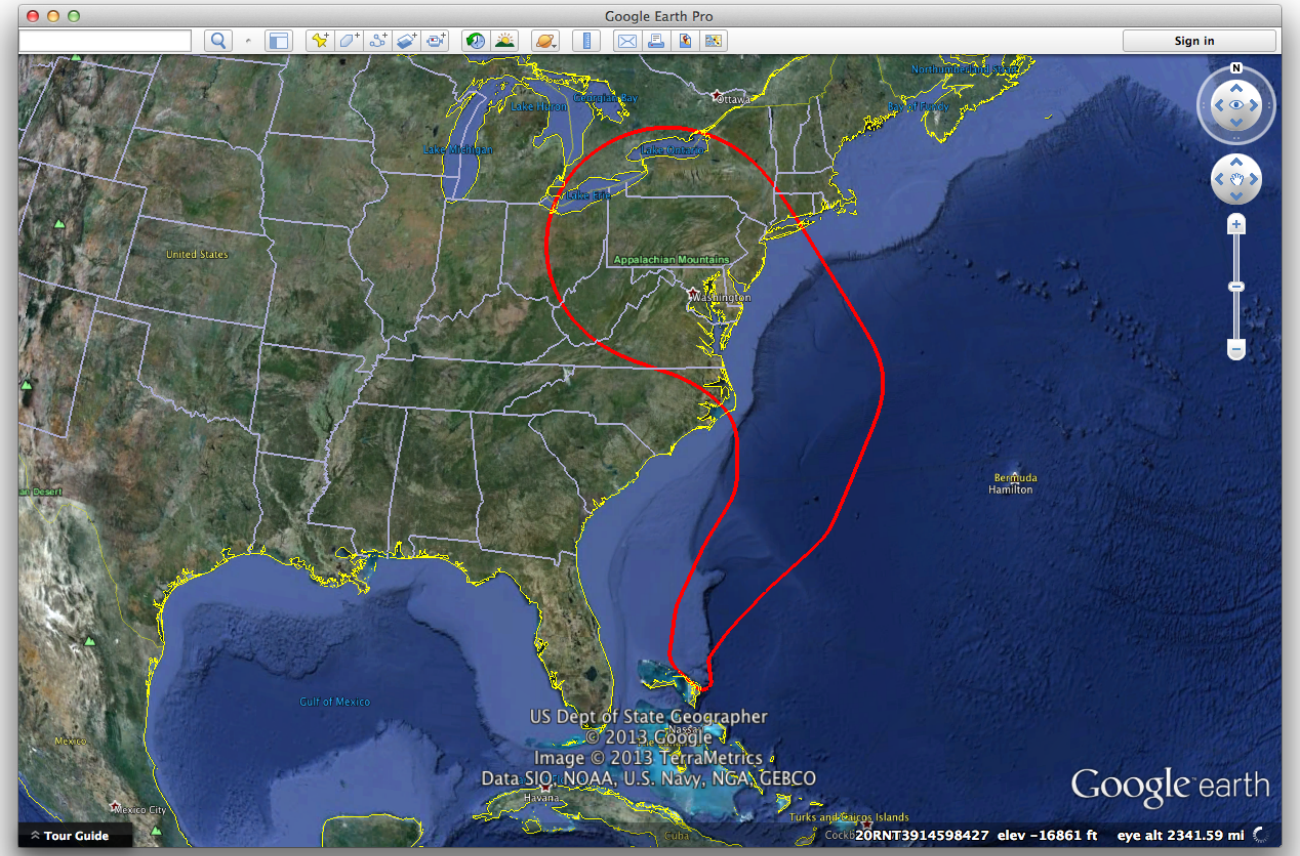
```
#!/usr/bin/python

import urllib2
from xml.dom.minidom import parse, parseString


CAP = "http://alerts.weather.gov/cap/us.php?x=0"

data = urllib2.urlopen(CAP).read()

dom = parseString(data)

items = dom.getElementsByTagName("entry")
Max = len(items)

for i in range(0,Max):
      Find which alerts match your criteria ...
```

Programmatic Cutting



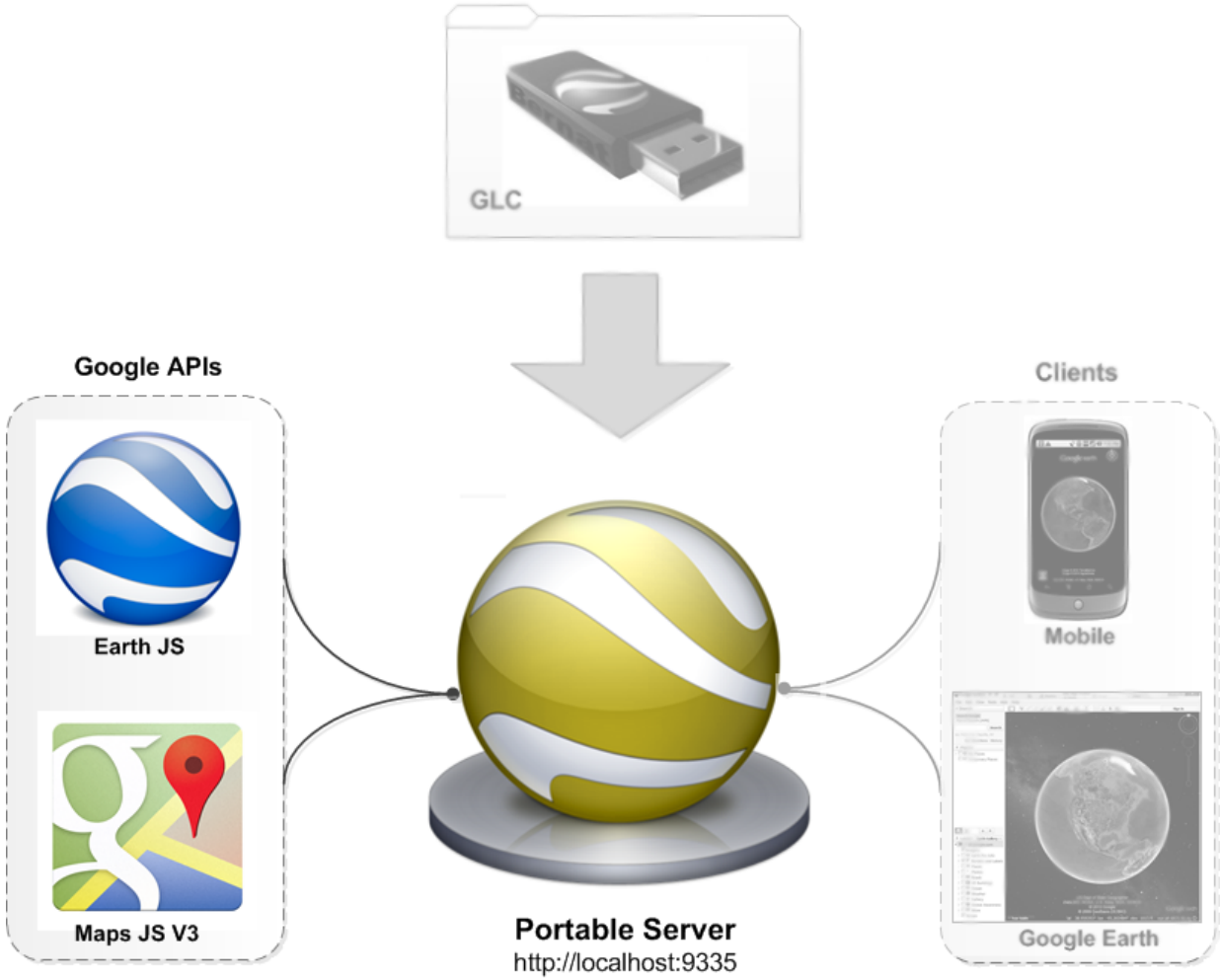
Polygon Filetype Conversion

```
wget http://www.nhc.noaa.gov/gis/forecast/archive/al182017_5day_017.zip
unzip al182012_5day_001.zip
ogr2ogr -f "KML" al182012.017_5day_pgn.kml al182012.017_5day_pgn.shp
```



Offline Extensibility

The key: leverage the Google APIs!



Offline 2D - Google Maps API

Google Maps Javascript API V3:

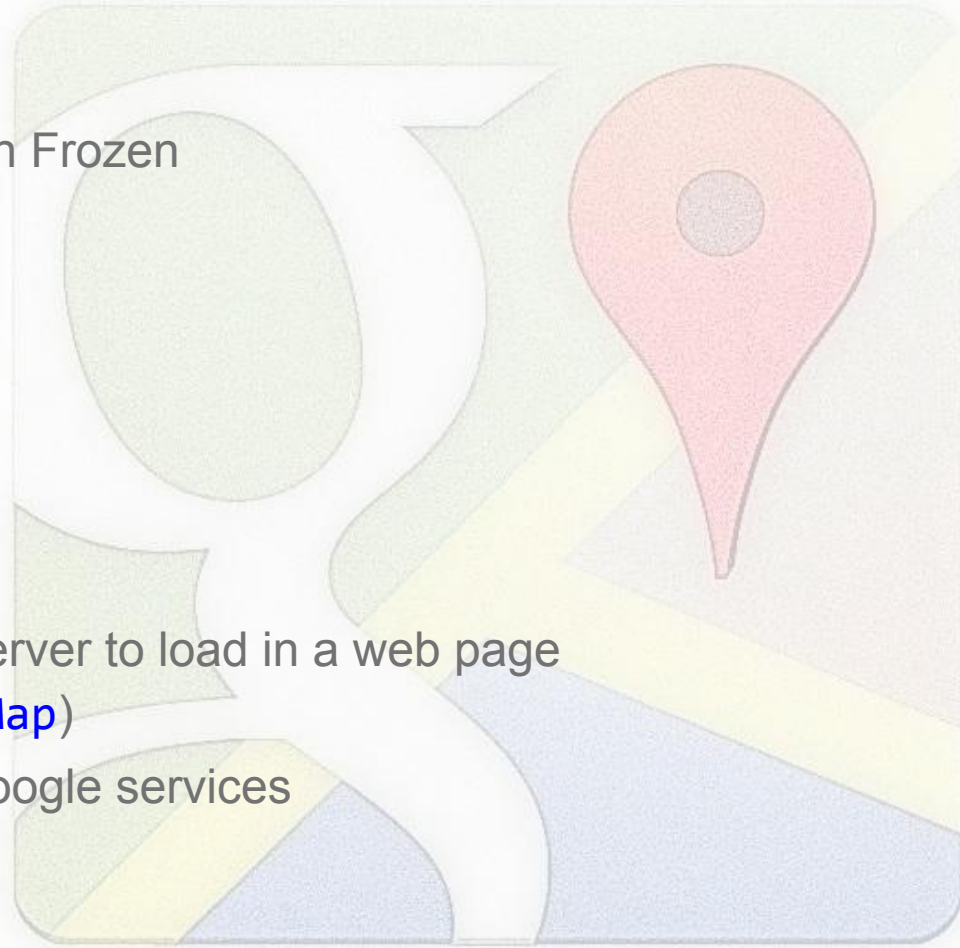
- Bundled with GEE Server and Portable Server
- Currently v3.9.19, but updated periodically to align with Frozen

Same core Javascript V3 API supporting:

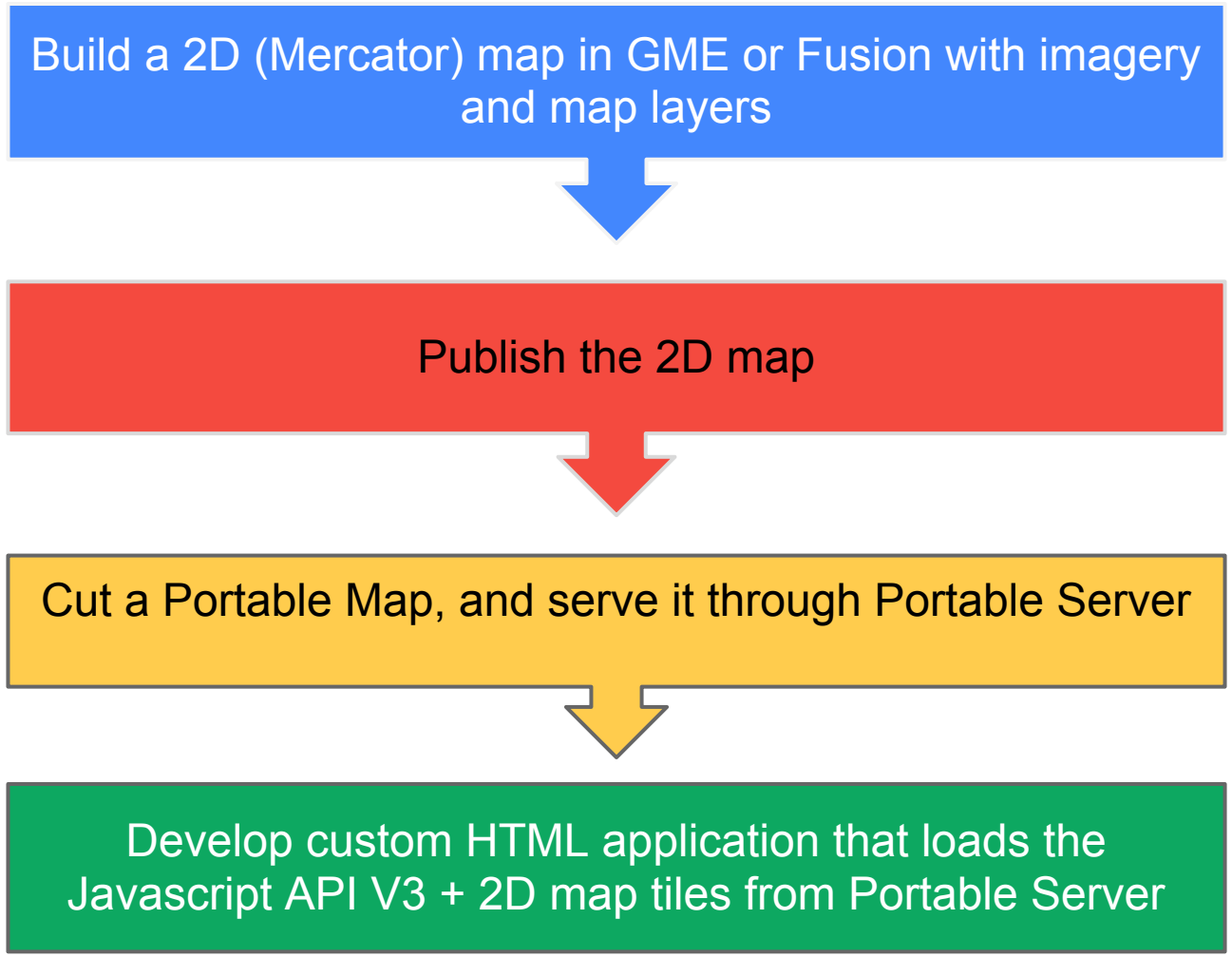
- Events
- Controls
- Overlays

but with some differences:

- Method of invoking the API from a GEE or Portable Server to load in a web page
- The main class used for the map instances (`GFusionMap`)
- No support for objects that require access to online Google services



Hosting a 2D Map from Portable Server



"Hello World" 2D offline

```
<!DOCTYPE html>
<html>
<head>
<title>GEE Maps JavaScript API v3 Example: Simple Map</title>
<script type='text/javascript' src='/local/maps/api/bootstrap_loader.js'></script>
<script type='text/javascript' src='/local/maps/api/fusion_map_obj_v3.js'></script>
<script type='text/javascript' src='/local/maps/api/fusion_maps_v3.js'></script>
<script type="text/javascript">
  function initialize() {
    var latlng = new google.maps.LatLng(-37, 115);
    var myOptions = {
      zoom: 6,
      minZoom: 4,
      maxZoom: 8,
      center: latlng,
      mapTypeId: google.maps.MapTypeId.ROADMAP,
      disableDefaultUI: true,
      scaleControl: true,
      navigationControl: true
    };
```

"Hello World" 2D offline (cont.)

```
var geemap = new GFusionMap("map_canvas", geeServerDefs, myOptions);
  }
</script>
</head>
<body onload="initialize()">
  <div id="map_canvas" style="width:100%; height:100%"></div>
</body>
</html>
```

class GFusionMap

- Instantiate class `GFusionMap` in order to create a map. This is an extension of the google.maps.Map class
- Use `GFusionMap` class instead of google.maps.Map to create applications that use layers from GME or GEE
- `GFusionMap(container, opts?)` creates a new map inside of the given HTML container, typically a DIV element. Options are the same as that of google.maps.Map.

Visualization demos - 2D Offline

Adapted from these Google online examples, but rendered *completely* offline against your own 2D map served by Portable Server

- Polyline styling with Symbols:
  http://localhost:9335/local/polyline-symbols.html
- Heatmap Layer:
  http://localhost:9335/local/heatmap.html
- Temporal visualization:
  http://localhost:9335/local/walmarts.html

Offline 3d - Google Earth API

- Compatibility with Windows and Mac
- Javascript API supporting:
  - Placemarks
  - Balloons
  - Geometries & Overlays
  - Touring
  - Historical imagery
  - KML import
- Load from multiple globe sources

  (`window.google.earth.addSideDatabase`)

"Hello World" 3D offline

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
  <head>
    <title>The Google Earth Plugin Offline Demo</title>
      <script type="text/javascript" src="earth_plugin_loader.js"></script>
    <script type="text/javascript">
    var ge = null;

    function init() {
      google.earth.createInstance('earth', initCB, failureCB,
         {database:'http://localhost:9335'});
    }

    function initCB (earth) {
      ge = earth;
      ge.getWindow().setVisibility(true);
    }

    </script>
  </head>
  <body onload="init();">
    <div id="earth" style="border: 1px solid #000; width:800px; height:600px;"></div>
    </body>
</html>
```

Python on Portable

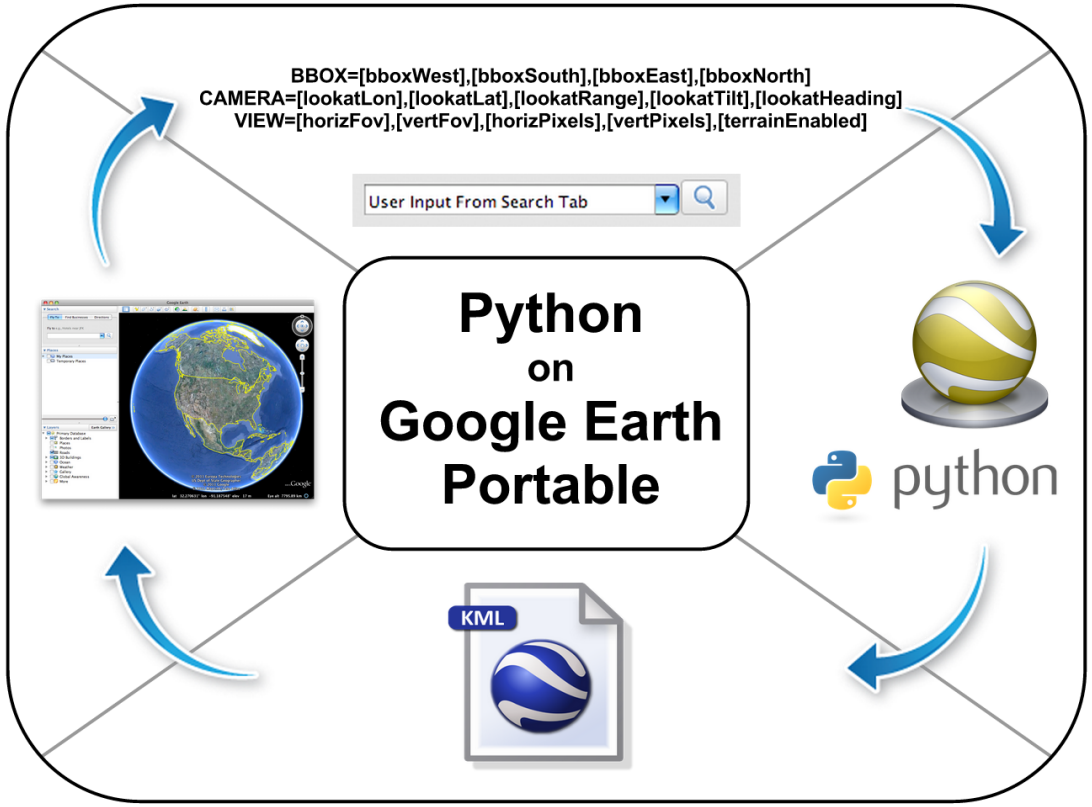
A framework for embedding ext services

Execute custom Python scripts

Install standard Python libraries

Extend the Google Earth search interface

- Generate KML for Google Earth
- Generate JSON for Google Maps

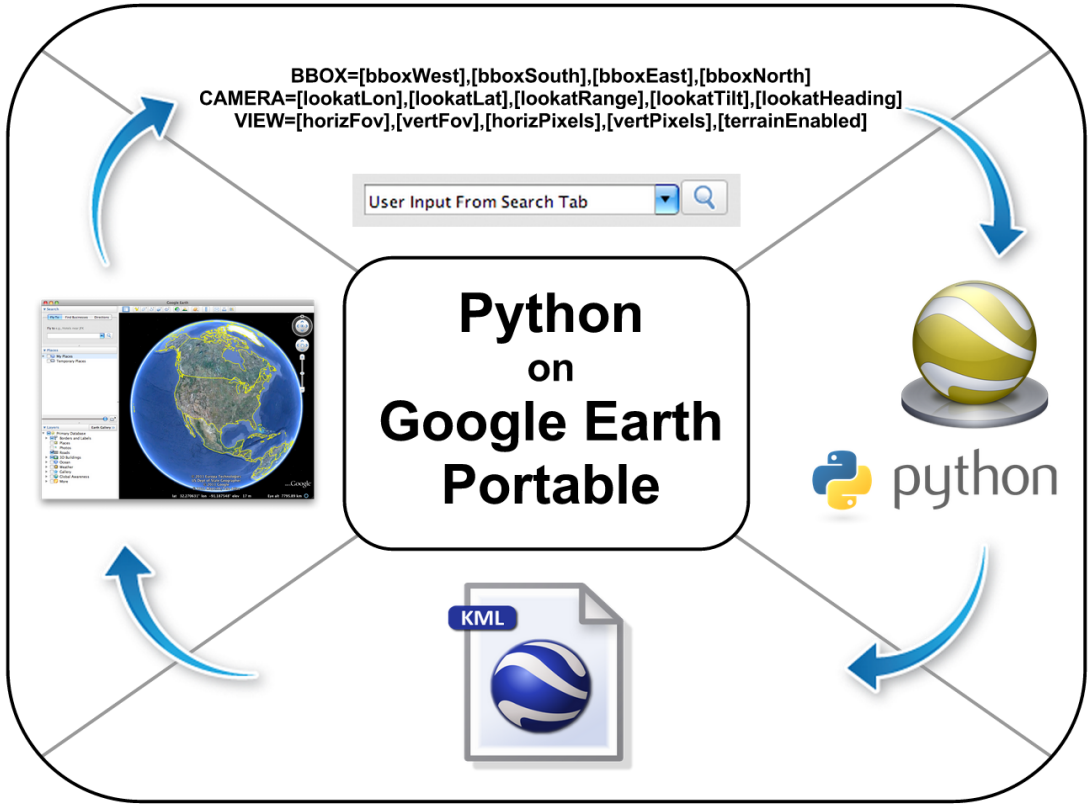


Python on Portable

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2" xmlns:gx="http://www.google.com/kml/ext/2.2" xmlns:kml="http://www.opengis.
net/kml/2.2" xmlns:atom="http://www.w3.org/2005/Atom">

<NetworkLink>
      <name>PIP Geocoder</name>
      <open>1</open>
      <Link>
            <href>http://www.sigacts.com/PIP/mkOverlay.cgi?</href>
            <viewRefreshMode>onStop</viewRefreshMode>
            <viewRefreshTime>0</viewRefreshTime>
            <viewFormat>
                  BBOX=[bboxWest],[bboxSouth],[bboxEast],[bboxNorth]&amp;LookAt=
                  [lookatLon],[lookatLat],[lookatRange]&amp;LookatTerrain=
                  [lookatTerrainLon],[lookatTerrainLat],[lookatTerrainAlt]&amp;
                  LookatHeading=[lookatHeading]&amp;LookatTilt=[lookatTilt]&amp;fovHoriz=
                  [horizFov]&amp;fovVert=[vertFov]&amp;horizPixels=[horizPixels]&amp;
                  vertPixels=[vertPixels]&amp;terrain=[terrainEnabled]
            </viewFormat>
      </Link>
</NetworkLink>

</kml>
```

Python on Portable

```
#!/usr/bin/python

import cgi
form = cgi.FieldStorage()

try:
        Terrain = str(form.getvalue('terrain'))
except:
        Terrain = "0"

if Terrain == "0":
        TSetting = "LookAt"
        TSetText = "Terrain Setting Off"
else:
        TSetting = "LookatTerrain"
        TSetText = "Terrain Setting On"

try:
        LaT = form.getvalue(TSetting)
except:
        LaT = "1,1"

Lat = float(LaT.split(',')[1])
Lon = float(LaT.split(',')[0])
```



Questions?

Thank You!

Sean Maday
smaday@google.com
@seanmaday

Avnish Bhatnagar
avnish@google.com


Google
Developers