
Google™
09
I/O

# Programming with and for Google Wave

Douwe Osinga
May 28th, 2009

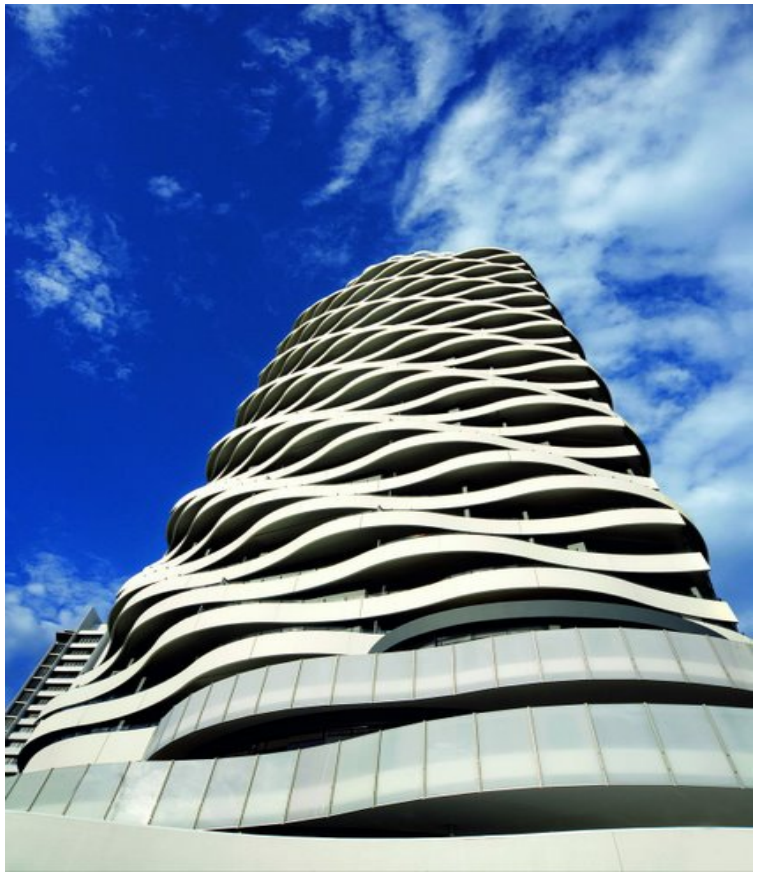
# Wave Developer Preview

# Introducing WaveSandbox.com

## Getting access to the sandbox and how to use it

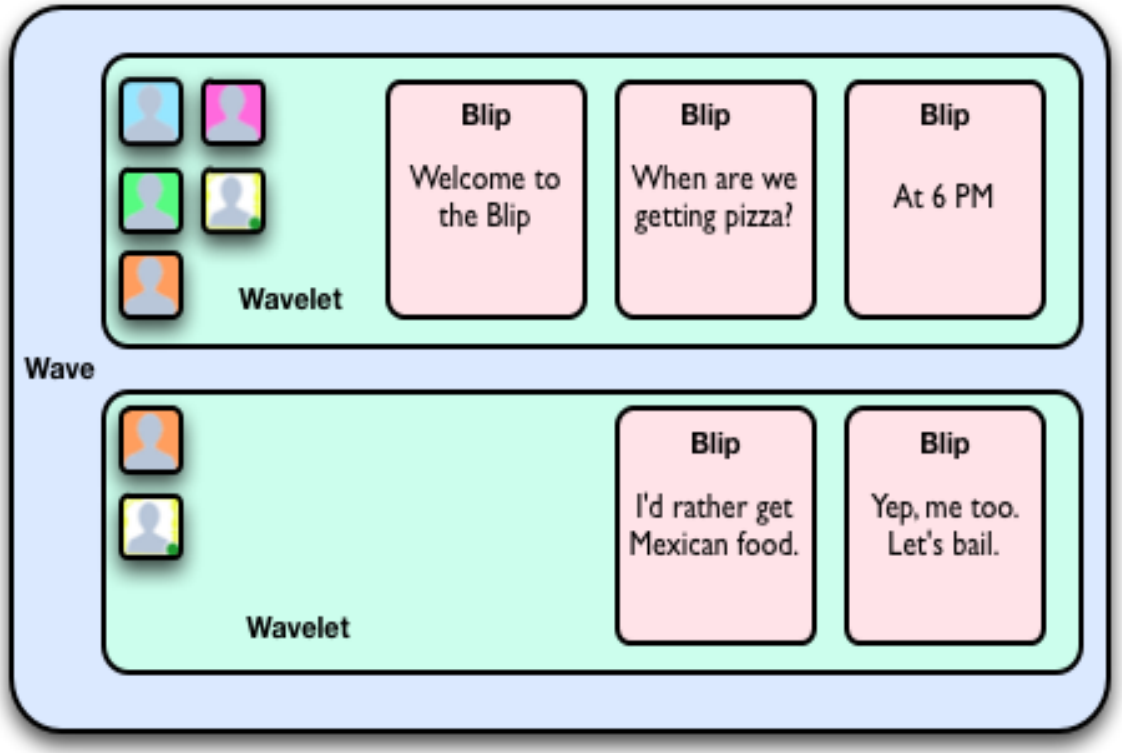
- Developer accounts initially on wavesandbox.com
- Explore the docs on the Google Code site
- Check out the new Google Wave blog site
- You will get an email inviting you to sign up
- Complete the form to pick your desired user name
- Accounts will be created starting next week!

# Wave Building



# Anatomy of a wave

## Waves, wavelets, participants, and blips

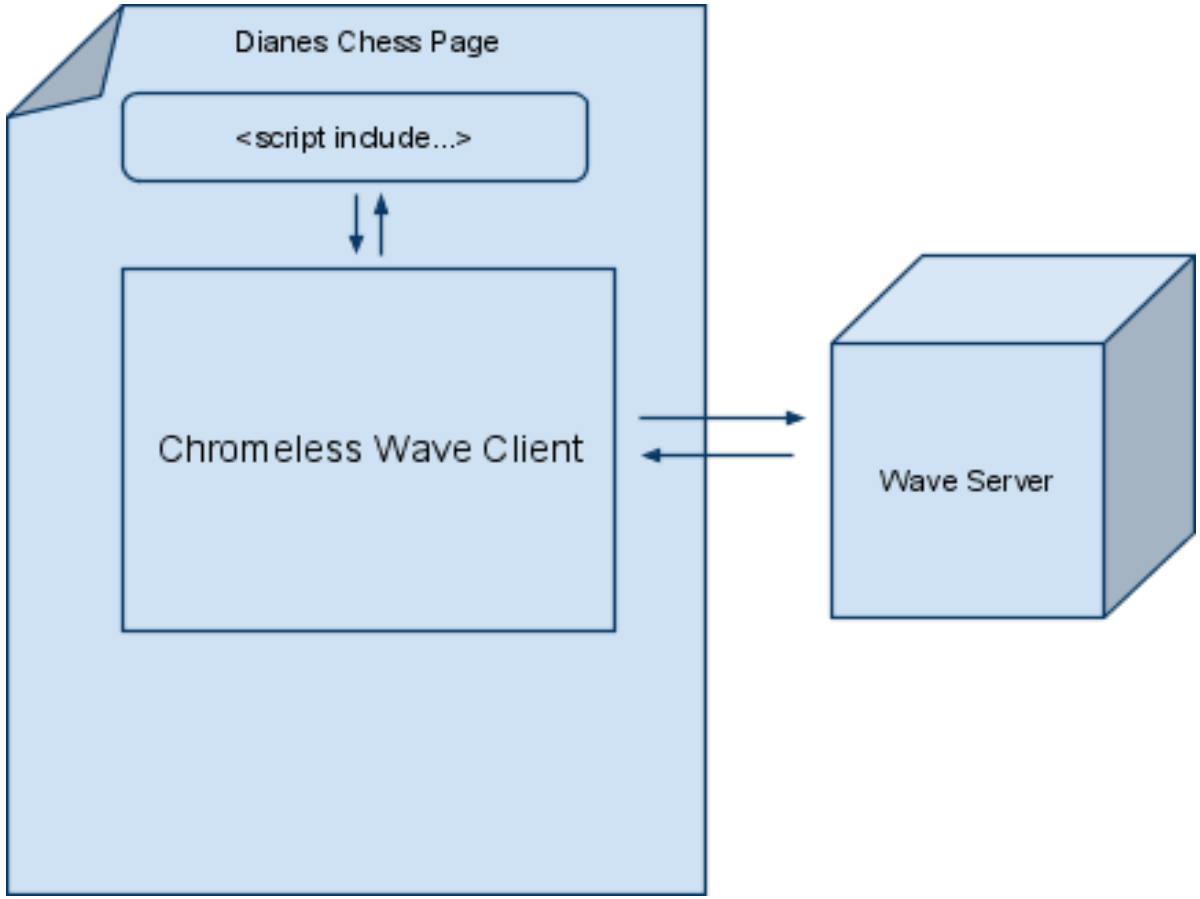



- A wave can be considered the root
- Waves contain wavelets
- Wavelets contain blips
- Permissions are applied at the wavelet level

# Embedding

# Embedding architecture

## How scripts communicate with the Wave Server



# Embed API

How scripts communicate with the Wave Server

# Example Code for Embedding

## Embedding a wave without customization

```
<div id='waveframe' style='height:500px;width:100%'/>
<script type='text/javascript'
    src='http://wave-api.appspot.com/public/embed.js'> </script>
<script>
  var wavePanel = new WavePanel(
        'http://wave.google.com/a/wavesandbox.com/');
  wavePanel.loadWave('wavesandbox.com!w+PfYnNrZk%1');
  wavePanel.init(document.getElementById('waveframe'));
</script>
```

# Customizing the UI

Example of using the *setUIConfig* method and addParticipant

```
<div id='waveframe' style='height:500px;width:100%'/>
<button type="button"
        onclick="wavePanel.addParticipant()">
    Add comment</button>
<script type='text/javascript'
    src='http://wave-api.appspot.com/public/embed.js'> </script>
<script>
  var wavePanel = new WavePanel(
        'http://wave.google.com/a/wavesandbox.com/');
  wavePanel.setUIConfig('red', 'black', 'courier new', '18px');
  wavePanel.loadWave('wavesandbox.com!w+PfYnNrZk%1');
  wavePanel.init(document.getElementById('waveframe'));
</script>
```

# Embedding Within a Google Gadget

Used to embed a wave on iGoogle and other gadget containers

```
<ModulePrefs title="Google Wave" height="600" ../>
<UserPref name="waveID" display_name="Wave ID" required="true" default_value="
wavesandbox.com!w+t7KgqNmw%1" />
<UserPref name="font" display_name="Font" required="false" default_value="" />
...
<Content type="html" view="home,canvas,default,profile">
<![CDATA[
<div id='waveframe' style='height:650px; width:100%;'></div>
<script type='text/javascript'
    src='http://wave-api.appspot.com/public/embed.js'></script>
<script type="text/javascript">
  var wavePanel = new WavePanel("http://wave.google.com/a/wavesandbox.com/");
  wavePanel.loadWave("__UP_waveID__");
  wavePanel.setUIConfig(
    '__UP_bgcolor__', '__UP_color__', '__UP_font__', '__UP_fontsize__');
  wavePanel.init(document.getElementById('waveframe'));
</script>
]]>
</Content>
</Module>
```

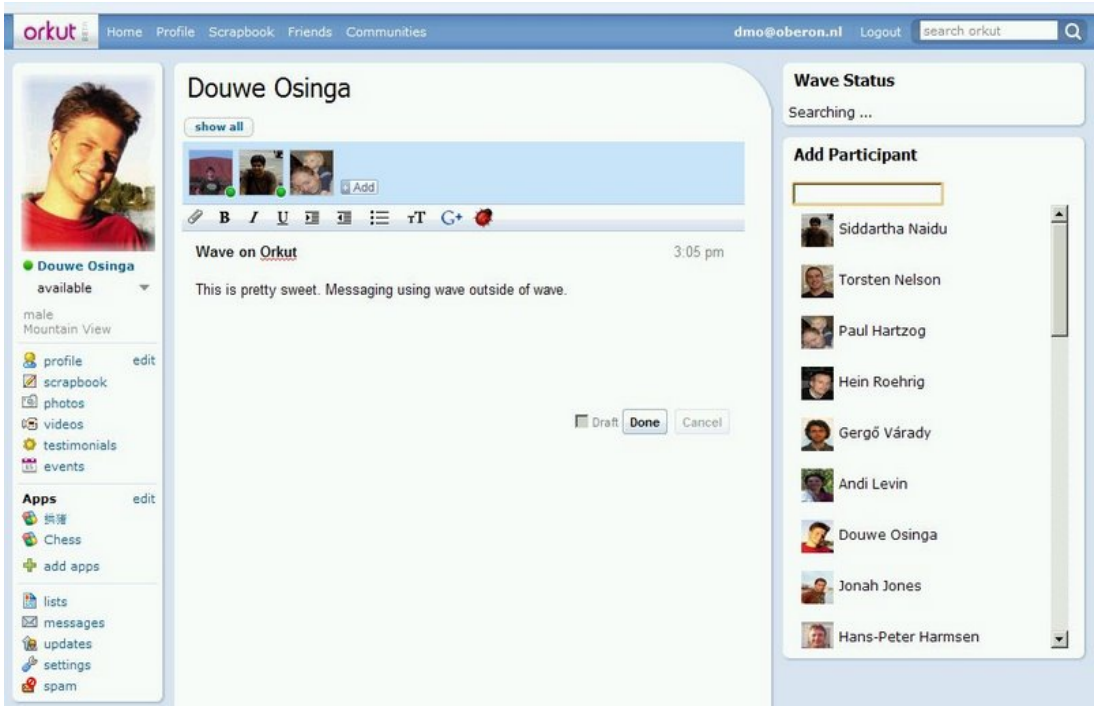
# Wave Embed Gadget Settings

Users don't have to copy any code


- Users can change settings without knowing JavaScript or HTML
- Allows waves to be on gadget containers that don't allow raw scripts

# Future Plans for Embedding

An overview of potential future enhancements




- Anonymous Access
- Search Panel
- Provide Participants
- Communication Hub

# Extensions: Gadgets and Robots

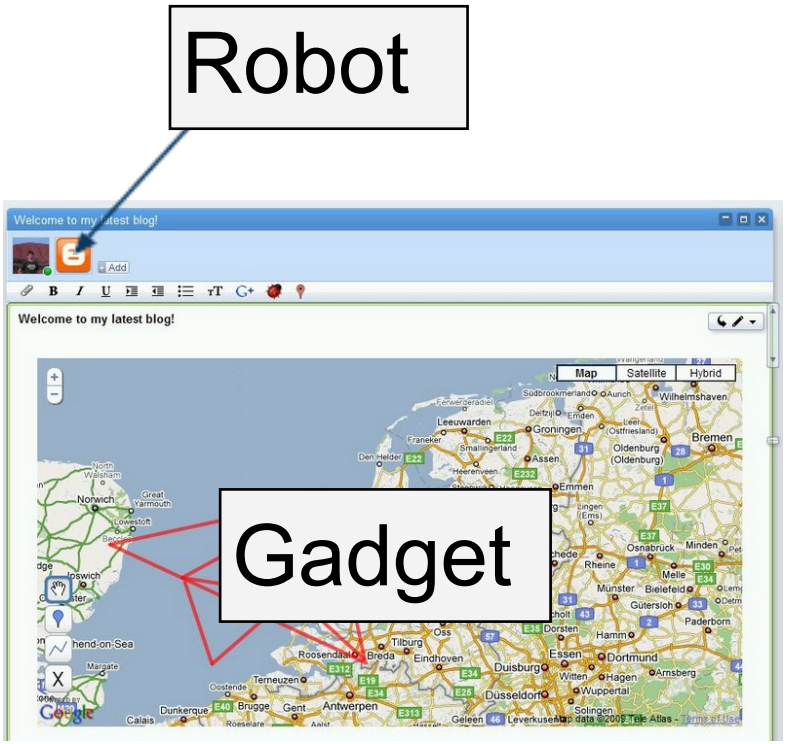
# Intro to Gadgets and Robots

The difference between a robot and a gadget

- Robot: participant, runs in the **cloud**
- Gadget: wave element, runs on the **client**

- Robot: interacts with the wave
- Gadget: interacts with the user, saves state in the wave

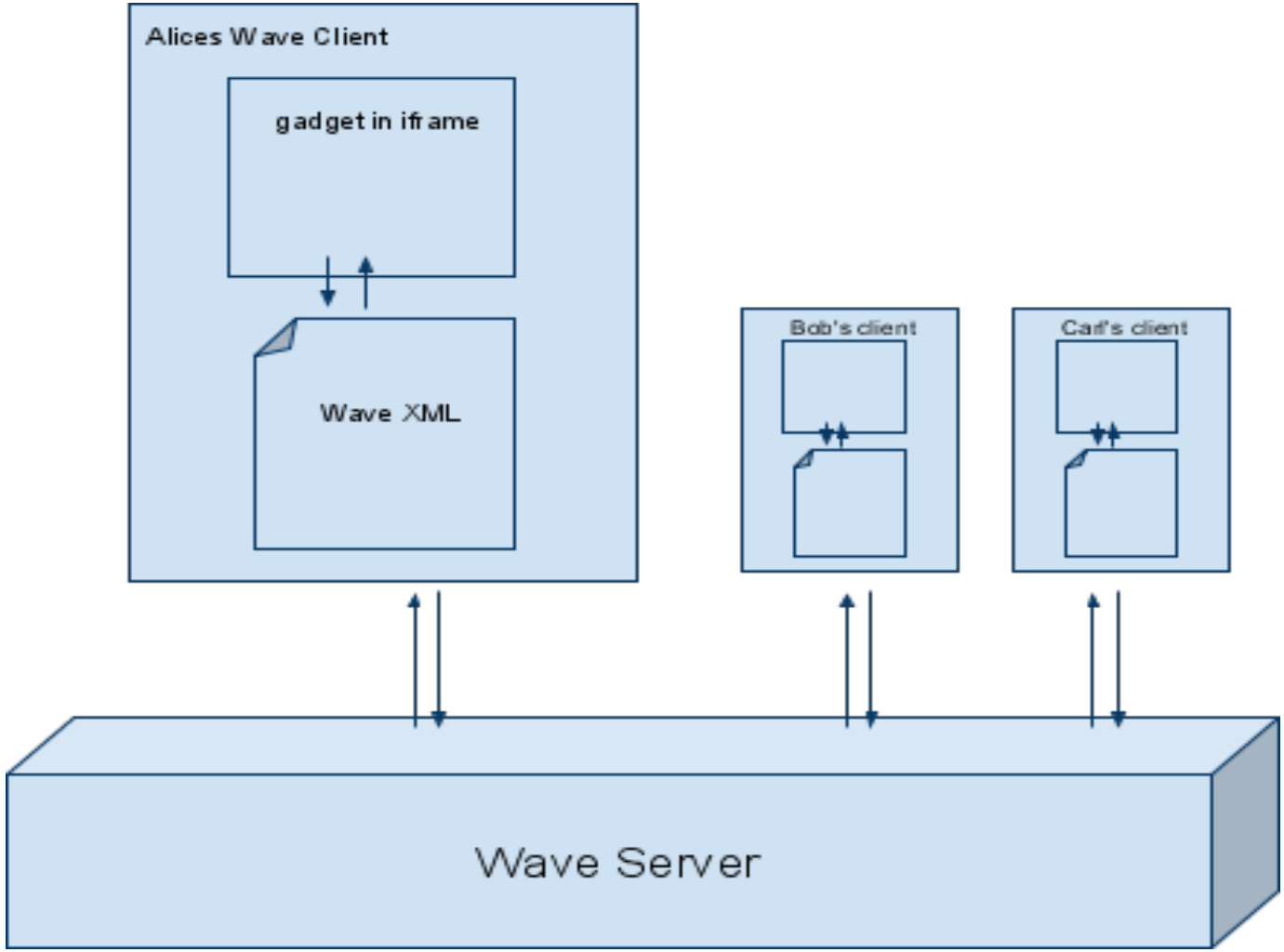
- Robot: observes and modifies the wave
- Gadget: has limited view of the wave, modifies only its own state



# Gadgets Deep Dive

# Gadget Architecture

Multiple clients talking to the Wave Server using Wave XML



# Wave Gadget API

Interacting with wave participants and state

## Wave

getParticipants()
getState()
getMode()
getTime()
getViewer()
getHost()
setParticipantCallback()
setStateCallback()

## State

getKeys()
submitDelta(delta)
submitValue(delta)
get(key, opt_default)

## Participant

getId()
getDisplayName()
getThumbnailUrl()

# Hello World Gadget

## Bare bones Gadget

```
<Module>
  <ModulePrefs title="Basemap" height="400" ../>
    <Require feature="rpc"/>
  </ModulePrefs>
  <Content type="html">
    <![CDATA[
      <script type="text/javascript"
        src="http://wave-api.appspot.com/public/wave.js">
      </script>
      <div id="map_canvas">Hello Wave</div>
    ]]>
  </Content>
</Module>
```

# Adding a Map

Paste in a standard Google Map API example

```
...
 <script type="text/javascript">
   var map;
   function main() {
     map = new GMap2(document.getElementById("map_canvas"));
     map.setCenter(new GLatLng(35, 135), 2);
   }
   gadgets.util.registerOnLoadHandler(main);
 </script>
 <div id="map_canvas" style="width: 100%; height: 100%"></div>
 ]]>
 </Content>
</Module>
```

# Making the Map Shared

Using *getState* , *setStateCallback* , *addListener* , and more

```
 ...
function stateChanged() {
  var state = wave.getState();
  map.setCenter(
      new GLatLng(state.get('lat', 35),
                  state.get('lng', 135)), 2);
  }

function main() {
  if (wave && wave.isInWaveContainer()) {
    wave.setStateCallback(stateChanged);
  }
  ...
  GEvent.addListener(map, "dragend", function() {
    wave.getState().submitDelta({
        lat: map.getCenter().lat(),
        lng: map.getCenter().lng()
    });
  });
```

# Adding Avatars

Using *getState* , *setStateCallback* , *addListener* , and more

```
var participants = wave.getParticipants();
for (var i = 0; i < participants.length; ++i) {
  var p = participants[i];
  var ploc = state.get(p.getId(), i * 2 + '|' + i * 2).split('|');
  var lat = parseFloat(ploc[0]);
  var lng = parseFloat(ploc[1]);
  var Icon = new GIcon();
  Icon.image = p.getThumbnailUrl();
  var marker = new GMarker(new GLatLng(lat, lng), {draggable:true, icon:Icon});
  map.addOverlay(marker);
  if (p.getId() == wave.getViewer().getId()) {
    marker.pid = p.getId();
    GEvent.addListener(marker, "dragend", function() {
      var d = {}
      d[this.pid] = this.getLatLng().lat() + '|' + this.getLatLng().lng();
      wave.getState().submitDelta(d);
    });
  }
```

# Other Examples of Gadgets

## Games, polls, puzzles, and more

| Hungry engineer | Order |
|---|---|
| Adam Schuck | In and Out Burger: No. 1 (Double double meal) |
| Luke Macpherson | In and Out Burger: No. 1 (Double double meal) |
| Alexandre Mah | Arby's: Beef 'n Cheddar Sandwich |
| Sam Thorogood | Pizza: Gourmet Veggie |
| Stephanie Hannon | In and Out Burger: Grilled Cheese, Chocolate Shake |
| Jonas S Karlsson | In and Out Burger: No. 2 (Cheeseburger meal), French Fries, Shake |

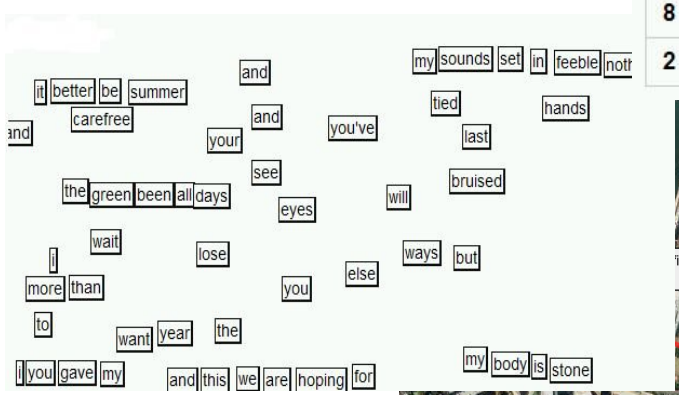
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 4 | 5 | 2 | 9 | 8 | 1 | 3 | 6 | 7 |
| 7 | 8 | 9 | 6 | 3 | 2 | 4 | 1 | 5 |
| | | 1 | 7 | 5 | 4 | 8 | 2 | 9 |
| 5 | 2 | | 1 | 6 | 7 | 9 | 4 | 3 |
| 1 | 3 | 4 | 8 | 2 | 9 | 5 | 7 | 6 |
| 9 | | | 5 | 4 | 3 | 1 | 8 | 2 |
| 6 | 4 | 3 | 2 | | 8 | 7 | 5 | 1 |
| 8 | 1 | 5 | 3 | 7 | 6 | 2 | 9 | 4 |
| 2 | 9 | 7 | 4 | 1 | 5 | 6 | 3 | 8 |

**Game updates:**

cadams@google.com has gained a point (7)
cadams@google.com has gained a point (6)
cadams@google.com has gained a point (5)
cadams@google.com has gained a point (4)
cadams@google.com has gained a point (3)

**Ranking:**

(7) cadams@google.com
(1) hannon@google.com
(1) lars@google.com
(0) austinchau@google.com



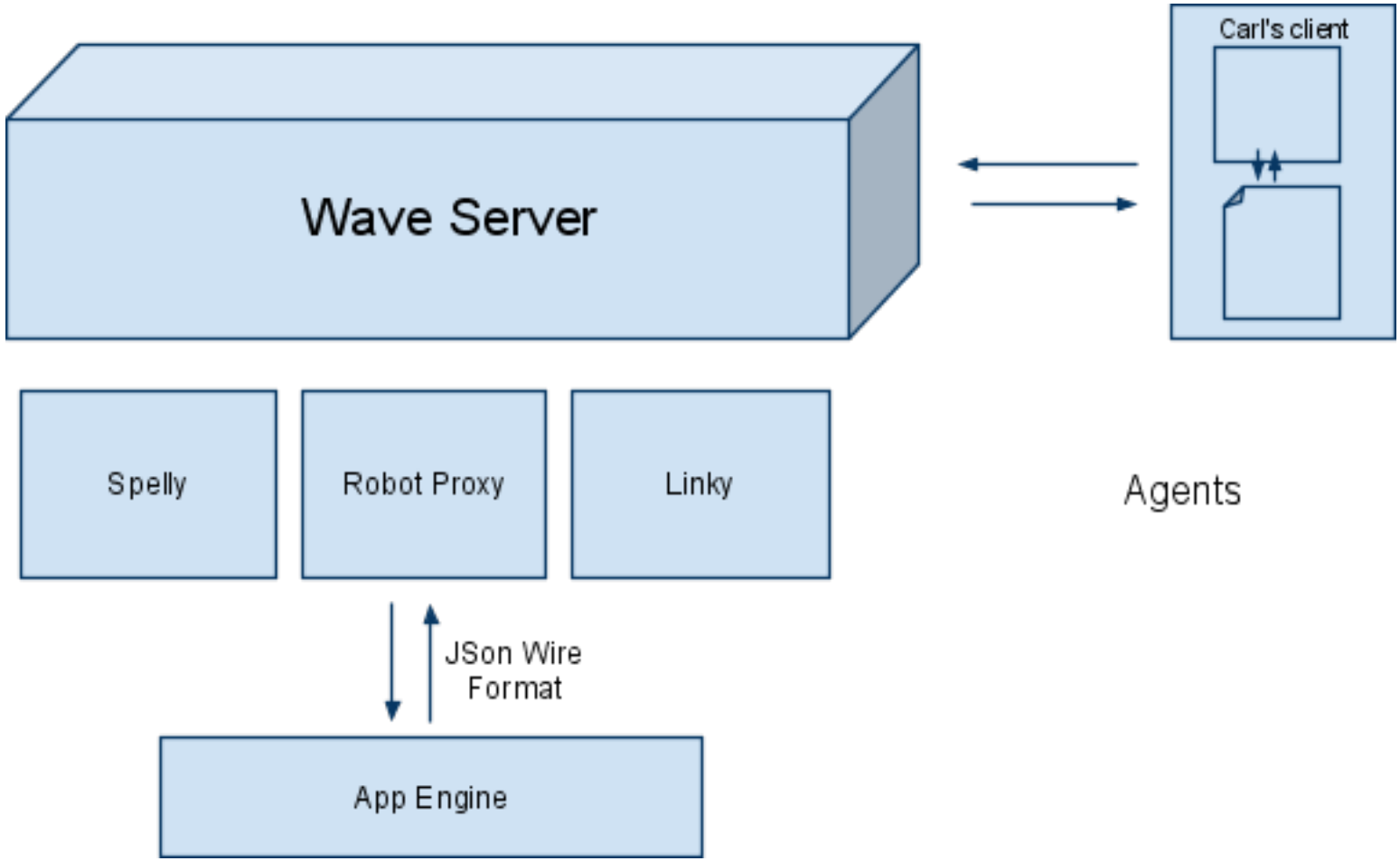



1. e2_e4 e7_e5
2. f2_f4 e5xf4
3. Bf1_c4 Qd8_h4
4. Ke1_f1 b7_b5
5. Bc4xb5 Ng8_f6
6. Ng1_f3 Qh4_h6
7. d2_d3 Nf6_h5
8. Nf3_h4 Qh6_g5
9. Nh4_f5 c7_c6
10. g2_g4 Nh5_f6
11. Rh1_g1 c6xb5
12. h2_h4 Qg5_g6
13. h4_h5 Qg6_g5
14. Qd1_f3 Nf6_g8
15. Bc1xf4 Qg5_f6
16. Nb1_c3 Bf8_c5
17. Nc3_d5 Qf6xb2
18. Bf4_d6 Bc5xg1
19. e4_e5 Qb2xa1
20. Kf1_e2 Nb8_a6
21. Nf5xg7 Ke8_d8
22. Qf3_f6 Ng8xf6
23. Bd6_e7

# Robots Deep Dive

# Robot Architecture

How robots interact with clients and the Wave Server

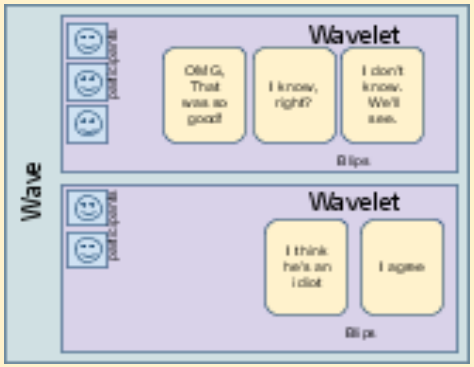



* All robots run on App Engine today but that will be opened up

# Wave Robot API

Overview of the model, events, and operations

## Model




TextView
GadgetView
FormView

## Events

**Wavelets**: BlipCreate, ParticipantChanged, TitleChanged
**Blips**: ContributorsChanged, Deleted, Submited, DocumentChanged
**Form**: ButtonClicked

## Operations

**Wavelet**: AppendBlip, AddParticipant, Create, RemoveSelf, SetDataDoc, SetTitle, CreateBlip
**Blip**: CreateChild, Delete
**Annotation**: Delete, Set
**Document**: Append, AppendStyled, Insert, Delete, **Replace**, Elements, InlineBlip

# Simple Robot

## Example of Smiley, the emoticon robot

```
"""Yasr: Yet another smiley robot"""
from api import events
from api import robot

def OnBlipSubmitted(properties, context):
  blip = context.GetBlipById(properties['blipId'])
  contents = blip.GetDocument().GetText()
  contents = contents.replace(':-(', unichr(0x2639)).replace(':-)', unichr(0x263A))
  blip.GetDocument().SetText(contents)

if __name__ == '__main__':
  yasr = robot.Robot('Yasr')
  yasr.RegisterHandler(
      events.BLIP_SUBMITTED, OnBlipSubmitted)
  yasr.Run()
```

# Completey

## Using the Google Search API and a wave robot

```
 if '???' in contents:
   q = '"%s"' % contents.replace('???', '*').replace('"', ' ')
   start = 0
   res = {}
   for i in range(6):
     url = 'http://ajax.googleapis.com/ajax/services/search/web?v=1.0&start=%d&q=%s'
% (start, urllib.quote(q))
     js = urlfetch.fetch(url=url).content
     for fragment in simplejson.loads(js)['responseData']['results']:
       for m in re.findall('\<b\>([^\<]*)', fragment['content']):
         m = m.lower()
         if m == '...':
           continue
         res[m] = res.get(m, 0) + 1
     start += 5
   if res:
     res = res.items()
     res.sort(lambda a,b: -cmp(a[1], b[1]))
     blip.GetDocument().SetText(res[0][0])
```

# More Example Robots

Some sample robots written to demo the API

- Polly
  - handles the flow of polling
  - demonstrates forms within a wave
  - uses waves as the data store
- Bloggy
  - publishes waves to a blog
- Buggy
  - connects wave to Issue tracker
- Searchy
  - does the heavy lifting for, uh, search
- Tweety
  - syncs between waves and Twitter

# Extensions Distribution

# Extensions API

## Extending the Google Wave client

### Hooks

**New Wave**
**Toolbar**

Keyboard short cut
File menu

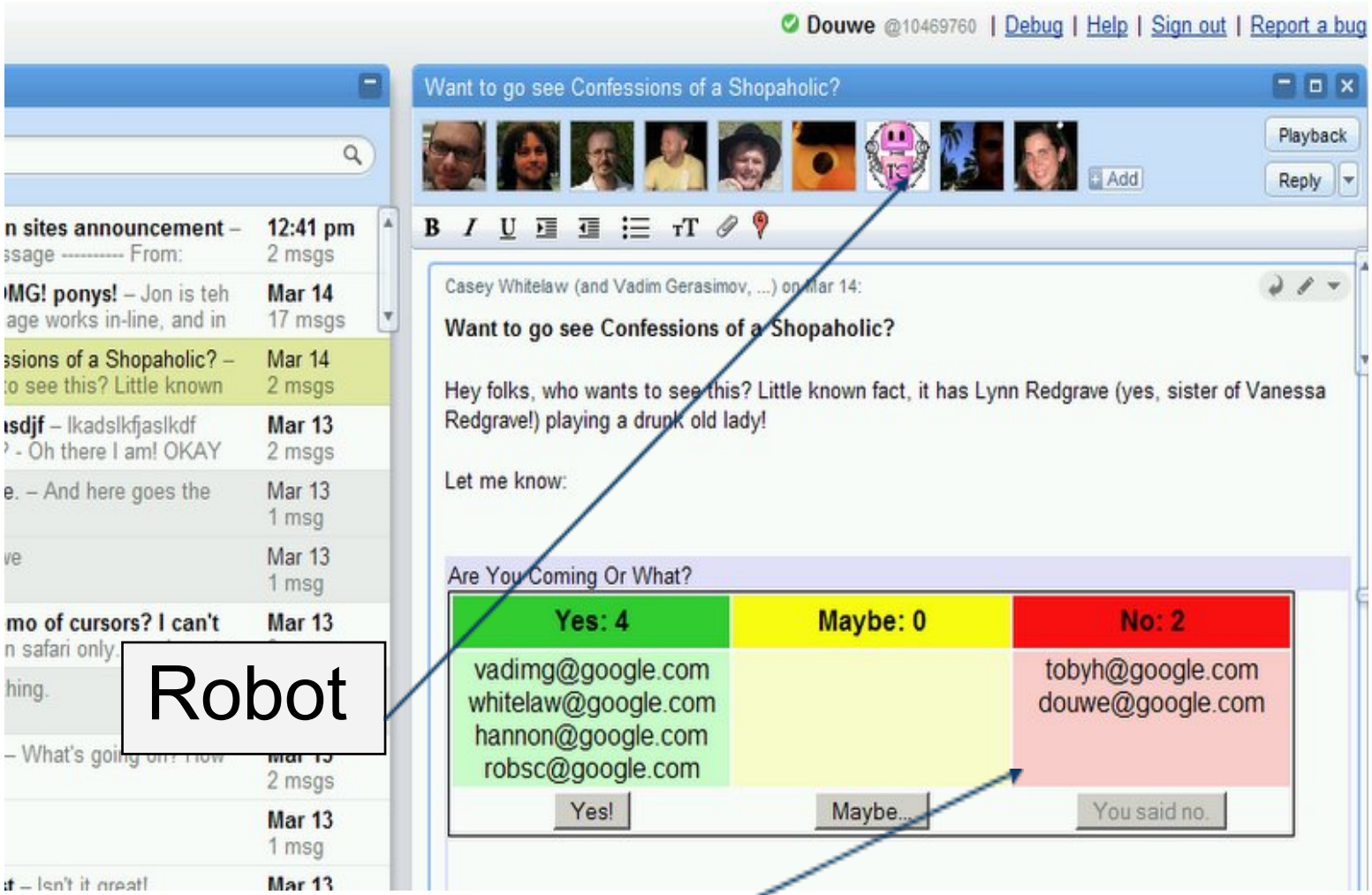
### Actions

**Insert Gadget**
**Add Participant**
**Create New Wave**

Apply Annotation

# Integrating Extensions

## Extensions allow easy access to robots and gadgets



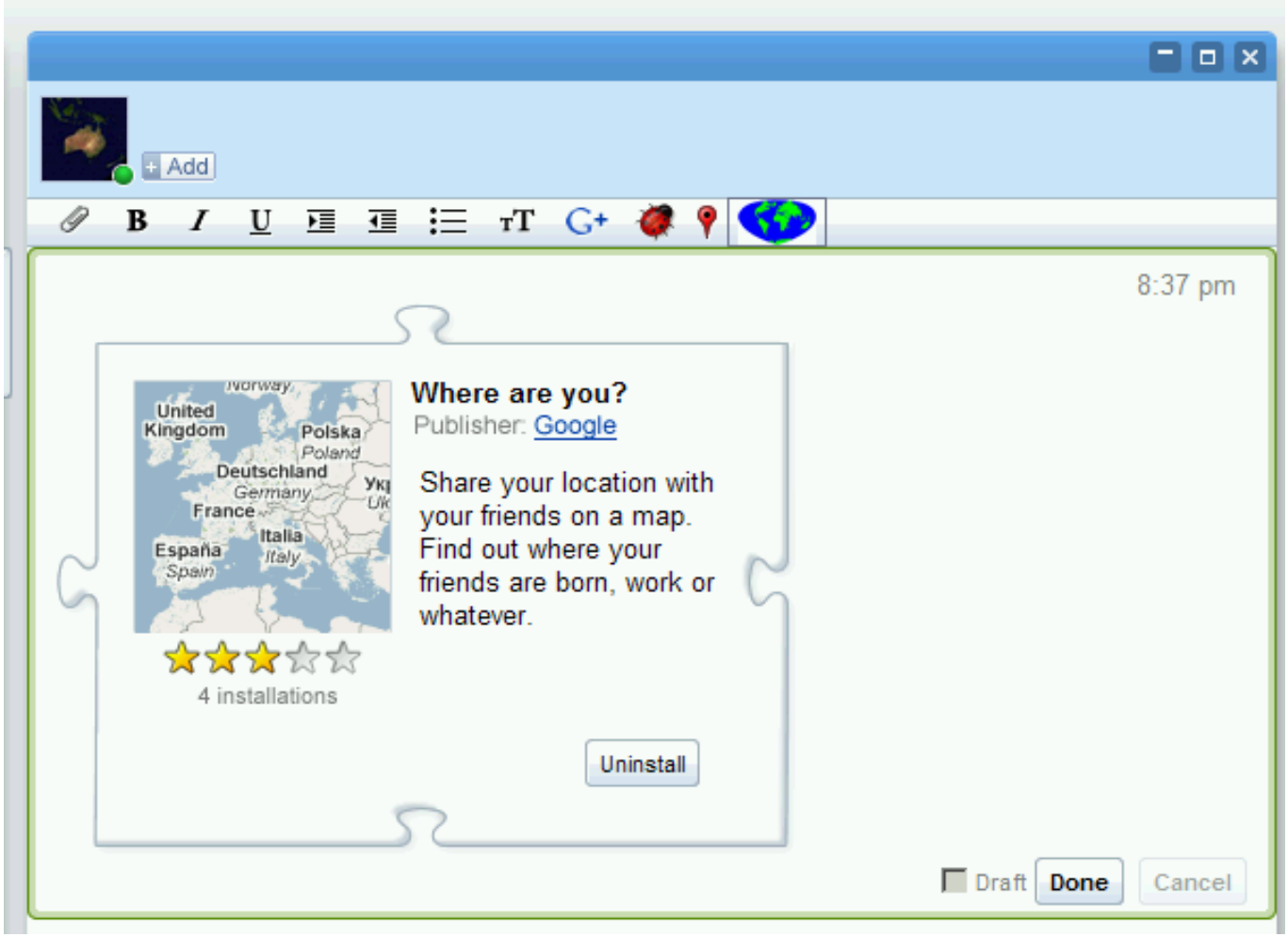
# Example Extension Installer

New Menu button for participant showing map gadget

```
<extension location="Toolbar">
  <info id="where-are-you-gadget"
        text="Where Are You?"
        description="Insert the Where Are You? gadget."
        imageUrl="http://wave-api-dmo.appspot.
com/public/simplemap/whereicon.png"/>
      <insertGadget
        url="http://wave-api-dmo.appspot.
com/public/simplemap/participantmap.xml"/>
</extension>
```

# Extension Installer Screen Shot

## Installer shows install state for current user



# Thank You!

Thanks for listening and be sure to check out the code site

for more info:

http://code.google.com/apis/wave/


Google™
09
I/O