Wrap up & Experimentation

CS147L Lecture 8
Mike Krieger

Intro

Welcome back!

By the end of today...

- Questions from implementations
- A few implementation loose ends
- A/B testing primer
- Google Analytics

Questions?

Loose ends

Floaty bar

Canonical implementation

- Gmail's mobile web app

Gmail Demo

Fitts Thumb

- Though hand input not quite the same as mouse, general **principle** applies:
  - Minimize thumb-moving distance
  - Make targets even larger than you think they need to be (thumbs are clumsy)

Getting plugin

- Included with jQTouch under extensions/
- Copy jqt.floaty.js into your JS folder

Integrating & customizing

```
<body>
    <div id="floaty">
        Options: <input type="button" value="Action"/>
    </div>
```

Initializing

```
<script type="text/javascript" charset="utf-8">
var jQT = new $.jQTouch();
$(document).ready(function(){
    $("#floaty").makeFloaty();
});
</script>
```

Styling

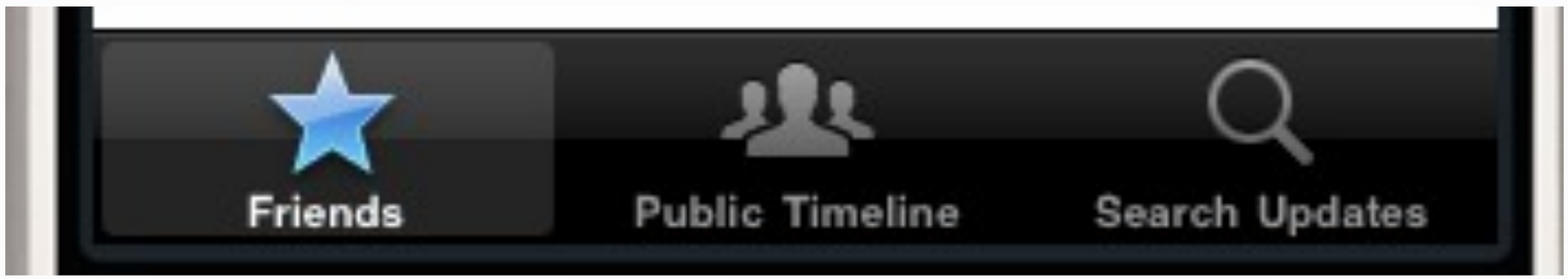
```
<style>
    #floaty {
        z-index: 100;
        left: 190px;
        width: 120px;
        padding: 10px 5px;
        background: #666;
        opacity: 0.9;
        -webkit-border-radius: 8px;
        font-size: 90%;

    }
</style>
```

Demo

- floaty.html

Bottom bar

UITabBarController on iPhone



This won't work...

- Traditional approach: position: fixed at bottom:0
- Or, div with overflow:hidden and bottom bar with absolute position and bottom:0

On the iPhone

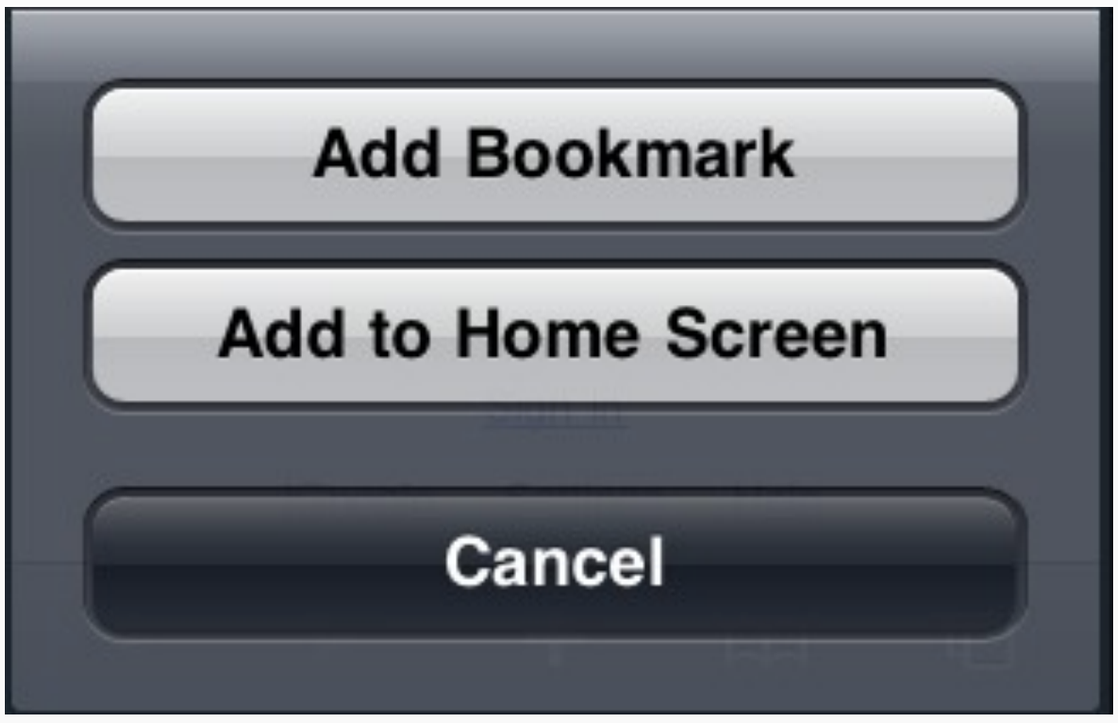
- Scrolling scrolls entire page
  - Floaty bar is probably the way to go...
  - You could hack it up, but your users would have to learn to two-finger scroll for everything

Full screening

App mode

- Only engaged when users click home button (might want to prompt them, or do it before hand)

App mode



A/B Testing

Why A/B test?

- You can have the best designers & great PMs...
- But nothing beats seeing what on earth people actually do
- Differences in *usability*, *virality*, and *revenue*

Framework

- Selecting an experiment
- Choosing variations
- Selecting / sampling users
- Deploying & serving variations
- Measuring user behavior
- Analyzing results

What makes a good experiment?

- Measuring user funnels through a sale
- Click-through rates for links
- Time spent / time until an action is taken
- Performance questions
- Email newsletters

(continued)

- Minor tweaks to site design
- **Flows with a clear goal**

In sum—

- a **measurable** user behavior that you believe will be **modulated** by tweaks in design

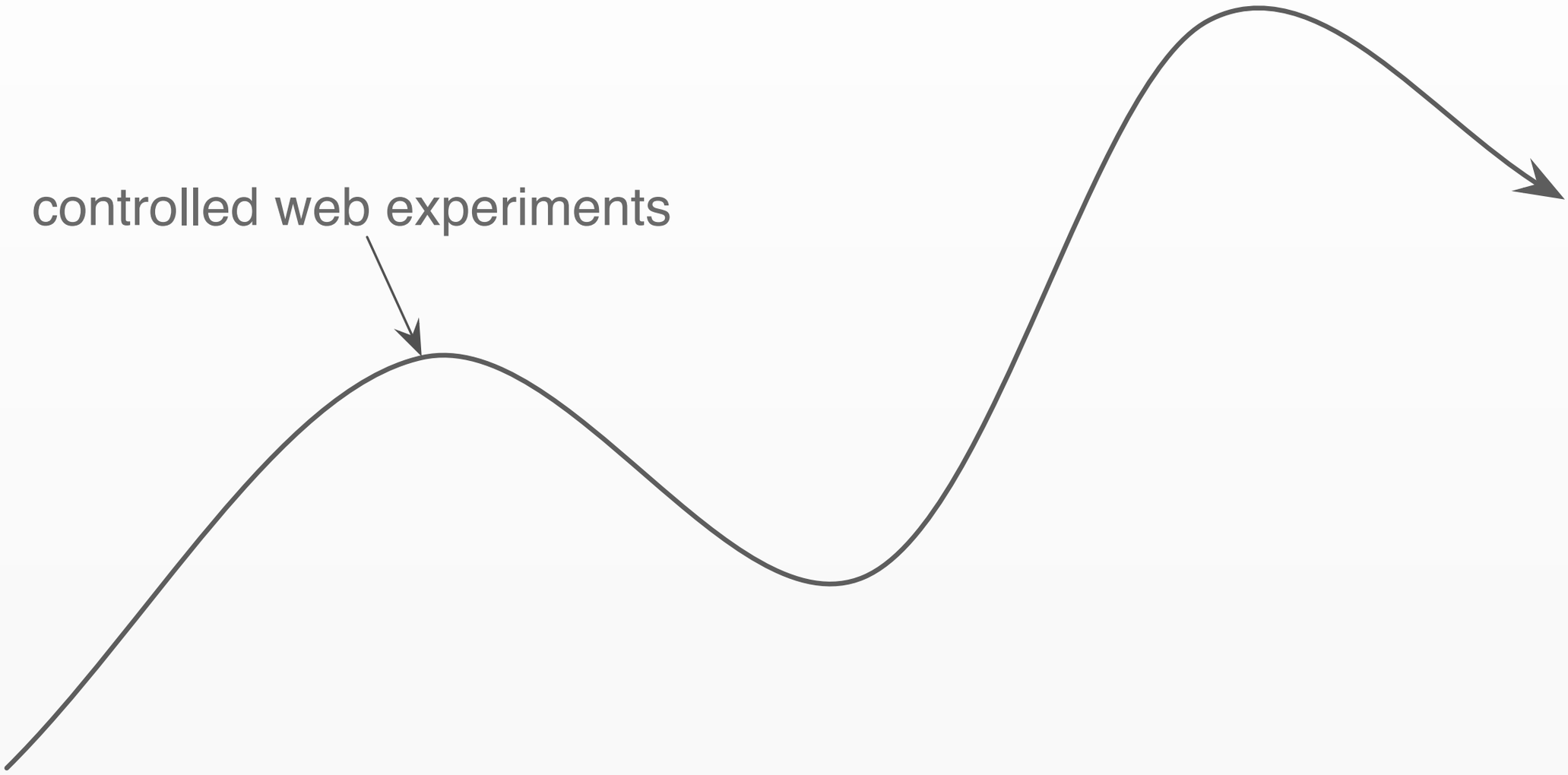
What A/B testing won't tell you

- Is it aesthetically pleasing?
- Is it fun?
- Is it accessible?
- Is this even what my company should be doing?

As Buxton would say

- A/B testing will help you get the **design right**, but can't help you get the **right design** in the first place

Hillclimbing



But really…

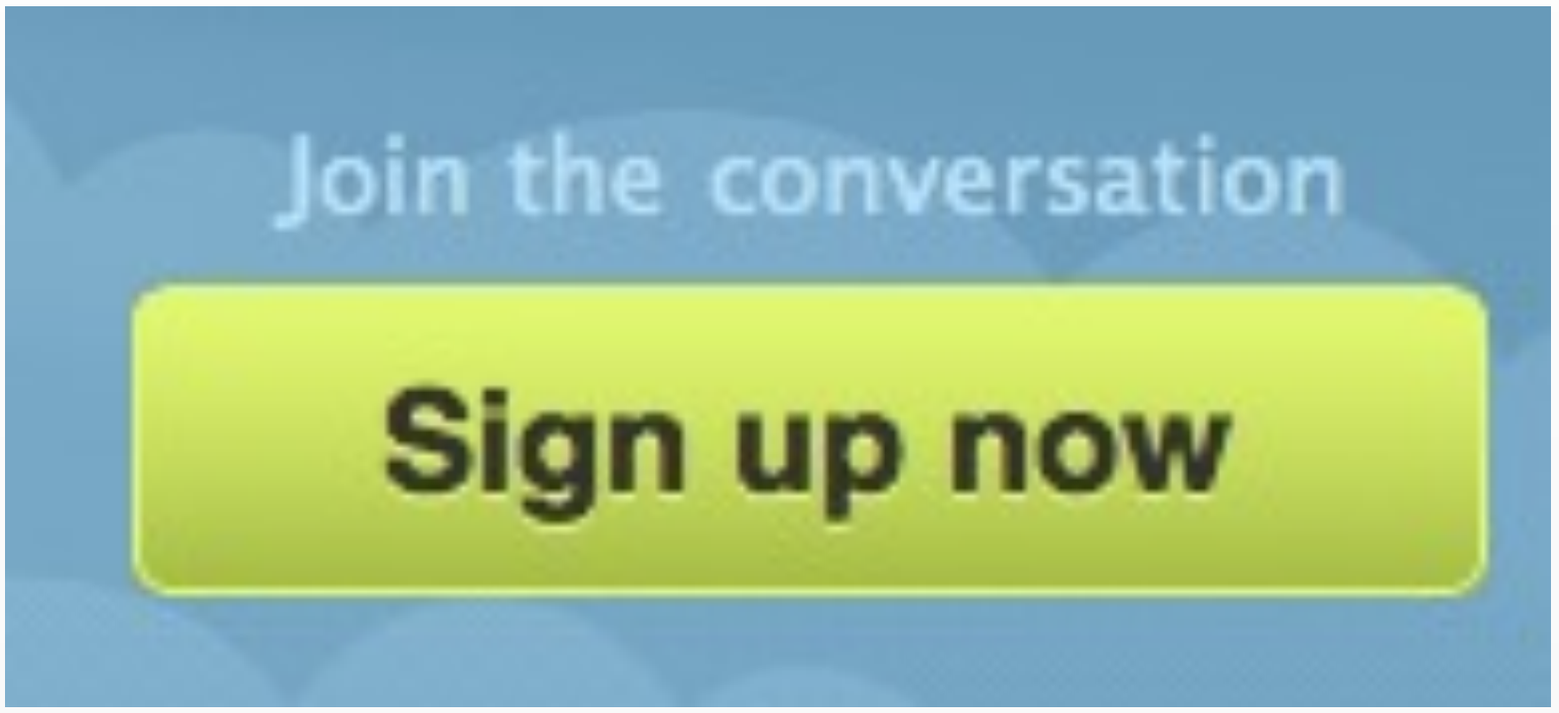
Choosing Variations

- Think in terms of **variables**
- Spectrum of choices
- If time (and participant pool), look at interactions, too

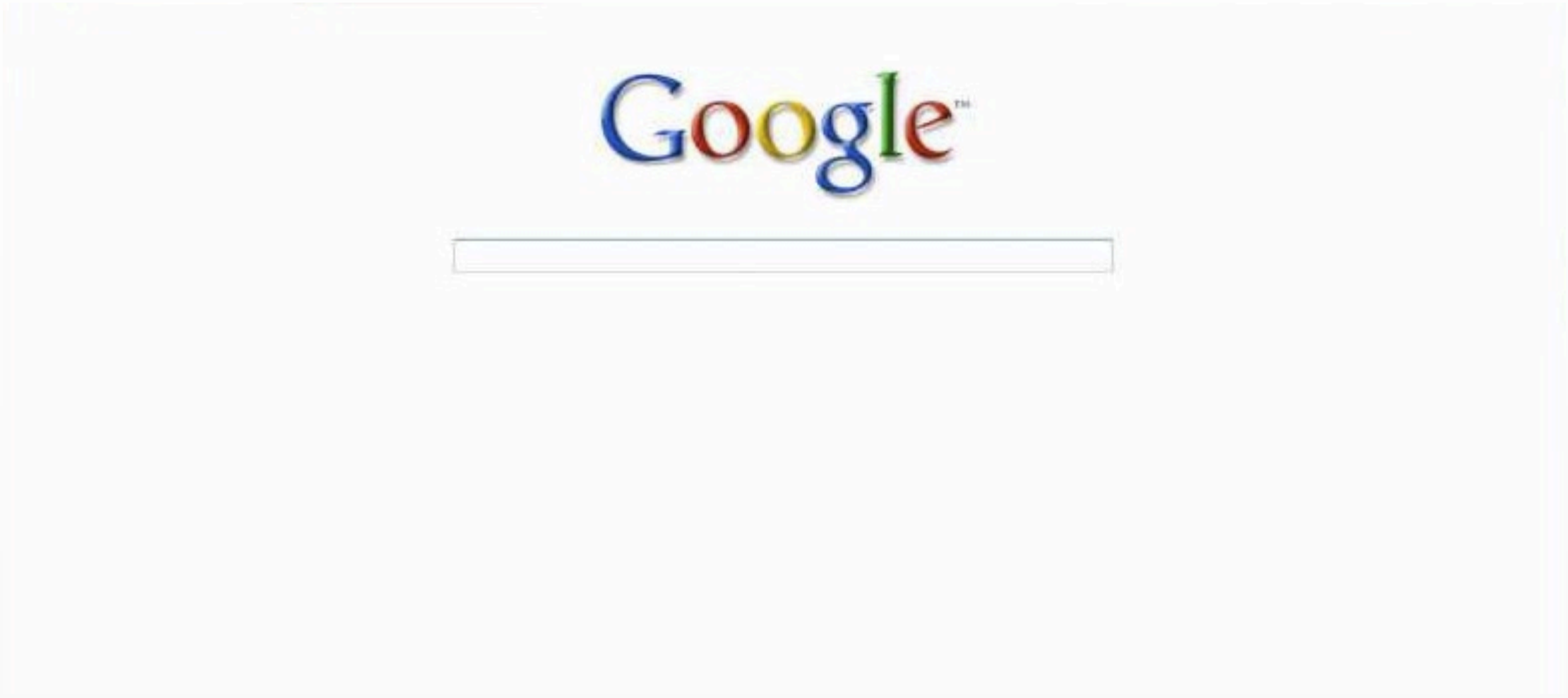
Examples

- Twitter homepage call to action
  - **Join the Conversation**
  - or
  - **Get Started**

Iteration



Google Homepage



...experiment?

- People actively want to join Google's A/B tests
- But can use interest/reactions as proxies for results in this case

Selecting/sampling users

- Two general approaches

The naive way

- Every time a user loads a page, they have a random chance of ending up in a bucket

Why doesn't this work?

- Order effects
- Confuse the users, who want a consistent experience
- Random functions not so random

Using a hashing function

- Suggested by Kohavi in his Web Experimentation paper
- How it's implemented at Meebo

General idea

- Be as consistent as possible per user
- If we can, use User ID (across computers)
- If we can't, use a cookie (at least consistent at one computer)
- At the very worst, assign randomly

MD5

- Hashing function; not great for encryption but fine for our purposes
- Problem: hashes will be long strings and we actually want a probability distribution

Solution

- Hash the unique identifier plus the experiment name
- Get the hexadecimal digest of the resulting hash
- Convert to a decimal and see where it falls along the range of 0 to the Max number in the distribution

In other words...

```
$unique_identifier = "mike";
$experiment_id = "linkcolor";
$hashing_value = $unique_identifier.$experiment_id;
```

(continued)

```
$hashed = md5($hashing_value);
$substring = substr($hashed, 0, 7);
$hashed_as_dec = hexdec($substring);
$max_value = hexdec("FFFFFFF");
$probability = $hashed_as_dec / $max_value;
```

Notes

- Will be evenly distributed from 0 to 1.0
- We can use this probability to bucket people
- Given the same input, will result in same number every time

Deploying & serving variations

- For prototypes, much can be hard-coded
- For real production use, infrastructure can make life easier in the long run

One easy way

```php
if ($probability < 0.5) {
    include_once 'treament1.php';
} else {
    include_once 'treatment2.php';
}
```

In the long term

- Build out front-end to turn experiments on/off or config file

Overriding Javascript

- Problem: you already have most of your functions defined, but want your treatment to do something slightly different

Encapsulation

```
var myApp = {

    sendTweet : function(tweet) {
        ...
    }
};
```

Overwriting

```
myApp.sendTweet = function(tweet) {
    //do something different
}
```

Monkeypatching

- Idea: we want to do mostly the same thing, but do something before/ afterwards that's slightly different, or modify the input

How to

```
(function(){
    var oldFunction = myApp.sendTweet;
    myApp.sendTweet = function(tweet) {
        tweet += '!!!';
        oldFunction(tweet);
    }
})();
```

Measuring behavior

- Are people actually doing something different?
- Using log lines, or writing straight to DB

Normal Log lines

```
10.32.109.7 - - [18/Nov/2009:22:36:32 -0800] "GET /courses/
cs147/images/media.jpg HTTP/1.1" 200 191910 "http://
hci.stanford.edu/courses/cs147/" "Mozilla/5.0 (Macintosh; U;
Intel Mac OS X 10_6_2; en-us) AppleWebKit/531.21.8 (KHTML,
like Gecko) Version/4.0.4 Safari/531.21.10"
```

Tracking Log Lines

```
10.32.109.7 - - [18/Nov/2009:22:36:32 -0800] "GET /track?
condition=bluebutton&event=click&timebeforeclick=500 HTTP/1.1"
200 191910 "http://hci.stanford.edu/courses/cs147/" "Mozilla/
5.0 (Macintosh; U; Intel Mac OS X 10_6_2; en-us) AppleWebKit/
531.21.8 (KHTML, like Gecko) Version/4.0.4 Safari/531.21.10"
```

How to process?

- In the small: use Python
- In the large: use Hadoop and Pig

Pig, ultrabriefly

- (because I think this will be huge in a year or so)

Pig

- SQL-like language built on top of Hadoop
- Makes writing Map/Reduce tasks really quick
- In use at Yahoo!, Twitter, Meebo, etc

Pig sample code

```
logs = LOAD "logs.txt" USING PigStorage("\t") AS
    (user_id:chararray, timestamp:int, event:chararray,
        value:chararray, condition:chararray)

grouped = GROUP logs BY event;

FOREACH grouped {
    GENERATE group, COUNT($1);
}
```

And the best part…

- Will compile & run for you over as many machines as necessary

PHP Logging scripts

- Wouldn't work for production data
- Fine for any A/B tests or just logging / instrumentation you want to do

DB Schema

```
if ($db = new SQLiteDatabase('logger.db')) {
    $result = $db->query("SELECT name FROM sqlite_master WHERE type='table' AND name='events'");
    if ($result->numRows() == 0) {
        $db->queryExec('CREATE TABLE events (
            ip text,
            ts int,
            category text,
            event text,
            condition text,
            extra text)');
    }
}
```

Logging events

```
$ip = $_SERVER['REMOTE_ADDR'];
$category = sqlite_escape_string($_REQUEST['category']);
$event = sqlite_escape_string($_REQUEST['event']);
$condition = sqlite_escape_string($_REQUEST['condition']);
$extra = sqlite_escape_string($_REQUEST['extra']);
$timestamp = time();

$statement = "INSERT INTO events (ip,ts, category,event,condition,extra) VALUES('%s', %d, '%s',
, '%s', '%s')";
$to_execute = sprintf($statement, $ip, $timestamp, $category, $event, $condition, $extra);
$db->queryExec($to_execute);
```

Example

```php
// for now, just placeholder
$hashing_value = "visitor"."buttoncolor";
$hashed = md5($hashing_value);
$substring = substr($hashed, 0, 7);
$hashed_as_dec = hexdec($substring);
$max_value = hexdec("FFFFFFF");
$probability = $hashed_as_dec / $max_value;

if ($probability < 0.33) {
    $buttoncolor = "#5c77af";
    $condition = "blue";
} else if ($probability < 0.66) {
    $buttoncolor = "#af423c";
    $condition = "red";
} else {
    $buttoncolor = "#32af43";
    $condition = "green";
}
```

HTML

```
    <a href="#" id="mybutton" style="background-color:<?php echo
$buttoncolor ?>; padding: 10px; color:white">Click me please!</a>
```

JS

```
$(document).ready(function(){
    $("#mybutton").click(function(){
        $.get('logger.php', {
            'event':'buttonclick',
            'category':'ui-logs',
            'condition': '<?php echo $condition ?>',
            'extra': ""
        }, function(){
            console.log("logged!");
        })
    })
})
```

Closing the loop

- Get the data out & aggregate
- Visualize!

Reading data from SQLite

- report.php

Basic code

```
$event = sqlite_escape_string($_REQUEST['event']);

$prepared = "SELECT * FROM events WHERE event = '%s'";
$query = sprintf($prepared, $event);
$data = $db->query($query);
$counts = array();
while($data->valid()) {
    $current = $data->current();
    $condition = $current['condition'];
    if (!isset($counts[$condition])) {
        $counts[$condition] = 1;
    } else {
        $counts[$condition]++;
    }
    $data->next();
}
```

Reformat as data series

```
$series = array();
$i = 0;
foreach ($counts as $key => $value) {
    array_push($series, array(
        "label"=>$key,
        "data"=>array(array($i, $value))
    ));
    $i++;
}
echo json_encode($series);
```

Result

[{"label":"blue","data":[[0,2]]},{"label":"red","data":
[[1,12]]},{"label":"green","data":[[2,1]]}]

Analysis options

- Excel
- Tableau
- R
- Javascript or Flash graphing/visualization libraries

Briefly: flot

- jQuery plugin
- Super useful for basic graphing & charting needs
- Also handles time-series data well
- http://code.google.com/p/flot/

From report.php to flot

```
$.get("report.php",
{'event':'buttonclick'},
function(response){
    var json = JSON.parse(response);
    var labels = [];
    // reformat for flot
    for(var i = 0; i < json.length; i++) {
        json[i]['bars'] = {
            'show': true,
            'fillColor': json[i]['label'],
            'lineWidth': 0
        };
        json[i]['lines'] = {
            'show': false
        }
        labels.push(json[i]['label']);
    }
    var labels =
    $.plot("#graph", json, {
        colors: labels
    });
})
```

Demo

`flot.html`

Even better: Protovis

- Stanford Graphics lab project
- http://vis.stanford.edu/protovis/

Significant change?

- Chi-Squared test

Chi-Squared

- Idea: measure whether a particular distribution of measures deviates *significantly* from expected

Null hypothesis

- Button color has no impact on click-through rate

Testing

$$X^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

Where:
O(i) is the *observed* frequency,
E(i) is the *expected* frequency
degrees of freedom = (number of categories) - 1

Sample data

Blue	Red	Green	Total
267	267	266	800

Significant?

Sample data

Blue	Red	Green	Total
270	250	260	800

Significant?

Sum the differences

```
Blue: (270-267)^2 / 267 = 0.03
Red: (250-267)^2 / 267 = 1.08
Green: (280-267)^2 / 267 = 0.63

0.03 + 1.08 + 0.63 = 1.74 = χ2
```

Look it up in table

- (or use R/SPSS/something fancier)
- http://www2.lv.psu.edu/jxm57/irp/chisquar.html

Significant?

Degrees of Freedom (*df*)	**Probability (p)**										
	0.95	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.01	0.001
1	0.004	0.02	0.06	0.15	0.46	1.07	1.64	2.71	3.84	6.64	10.83
2	0.10	0.21	0.45	0.71	1.39	2.41	3.22	4.60	5.99	9.21	13.82

p ~ 0.4 (we want 0.05)
not significant

Sample data

Blue	Red	Green	Total
240	230	330	800

Significant?

Sum the differences

```
Blue: (240-267)^2 / 267 = 2.73
Red: (230-267)^2 / 267 = 5.12
Green: (330-267)^2 / 267 = 14.86

2.73 + 5.12 + 14.86 = 22 = χ2
```

Significant?

Degrees of Freedom (*df*)	**Probability (p)**										
	0.95	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.01	0.001
1	0.004	0.02	0.06	0.15	0.46	1.07	1.64	2.71	3.84	6.64	10.83
2	0.10	0.21	0.45	0.71	1.39	2.41	3.22	4.60	5.99	9.21	13.82

$p < 0.001$

highly significant

Wrap-up

- Can be a bit of work
- But can lead to amazing insights
- Plus, data analysis & visualization is really fun

Google Analytics

Why?

- It's free!
- For a small company or project, much better than rolling out some of these analysis tools for yourself

Signing up

- http://analytics.google.com

Integrating JS

```
<script type="text/javascript">
var gaJsHost = (("https:" == document.location.protocol) ? "https://ssl." :
"http://www.");
document.write(unescape("%3Cscript src='" + gaJsHost + "google-analytics.com/ga.js'
type='text/javascript'%3E%3C/script%3E"));
</script>
<script type="text/javascript">
try {
var pageTracker = _gat._getTracker("UA-3549939-1");
pageTracker._trackPageview();
} catch(err) {}</script>
```

Insert before </body>

Tip: Tracking JS events

```
_trackEvent("useractions", "click", "condition", "bluebutton");
```

_trackevent function, takes
(category, event, optional_key, optional_value)

Using Web Optimizer

- Attached to AdSense
- Provides A/B testing tools with integrated confidence interval and significant different calculations

Demo

Final notes

Where to go from here?

- Four fun things to explore

Building full web apps in Django

- Templating
- Wrapping Request/Response
- Interfacing with the database
- Forms

Some code from courseapp

```
class Assignment(models.Model):
    """ A class assignment. Can be individual/group, submitted off/online """
    title = models.CharField(max_length=255)
    date_assigned = models.DateTimeField()
    date_due = models.DateTimeField()
    # raw points score that it's out of
    points = models.IntegerField()
    # how many points its worth in the long run, if different
    # (if not specified, points will be used)
    scaled_points = models.IntegerField(null=True, blank=True)
    submitted_online = models.BooleanField(default=True)
    group_submission = models.BooleanField(default=False)
    extra_credit = models.BooleanField(default=False)
    nonpublic = models.BooleanField(default=False)
```

and the view...

```
@authenticate_user
@login_user
@staff_member_required
def reviewassignment(request, assignment, studio=None):
    assignment = get_object_or_404(Assignment, pk=assignment)
    if studio is not None:
        submissions = Submission.objects.filter(Q(user__userinfo__section=studio) |
Q(group__section=studio), assignment__id=assignment.id)
        students = User.objects.filter(userinfo__section=studio)
        groups = Group.objects.filter(section=studio)
    else:
        submissions = Submission.objects.filter(assignment__id=assignment.id)
        students =
User.objects.filter(is_staff=False).exclude(userinfo=None).filter(userinfo__section__id__gte=1)
.order_by('userinfo__section__id')
        groups = Group.objects.all()
```

Trying out Google App Engine

- Great for weekend projects that could become something more
  - Django templating built-in
  - Can also use (most) of Django with app-engine-patch (http://code.google.com/p/app-engine-patch/)
  - And, it'll scale if you need it to

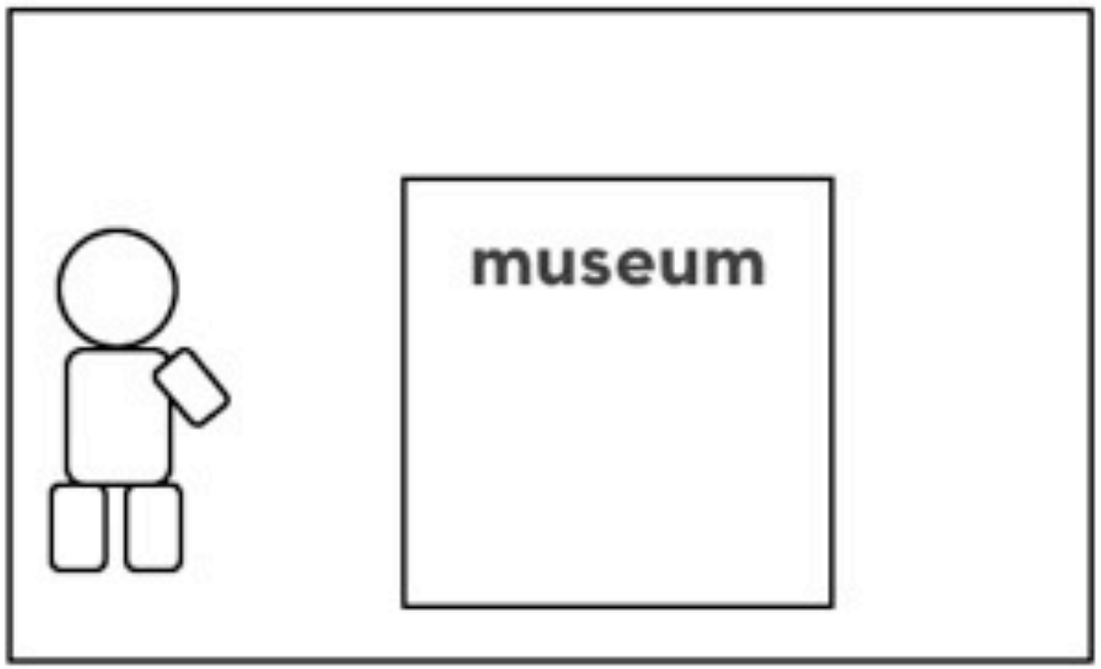
Adapting for native app

- PhoneGap lets you access native app features from JavaScript
- Can continue your class projects if you want to take them further
- Caveat: will have to get dev account

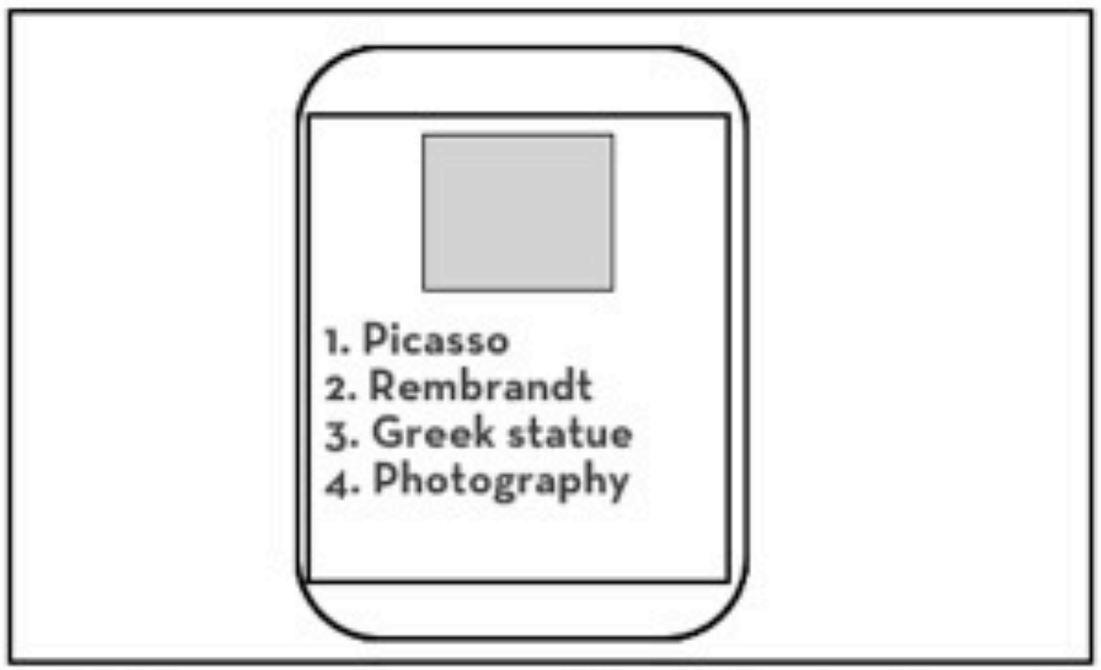
Using Mechanical Turk for app feedback

- And quick testing of ideas

Storyboard 1

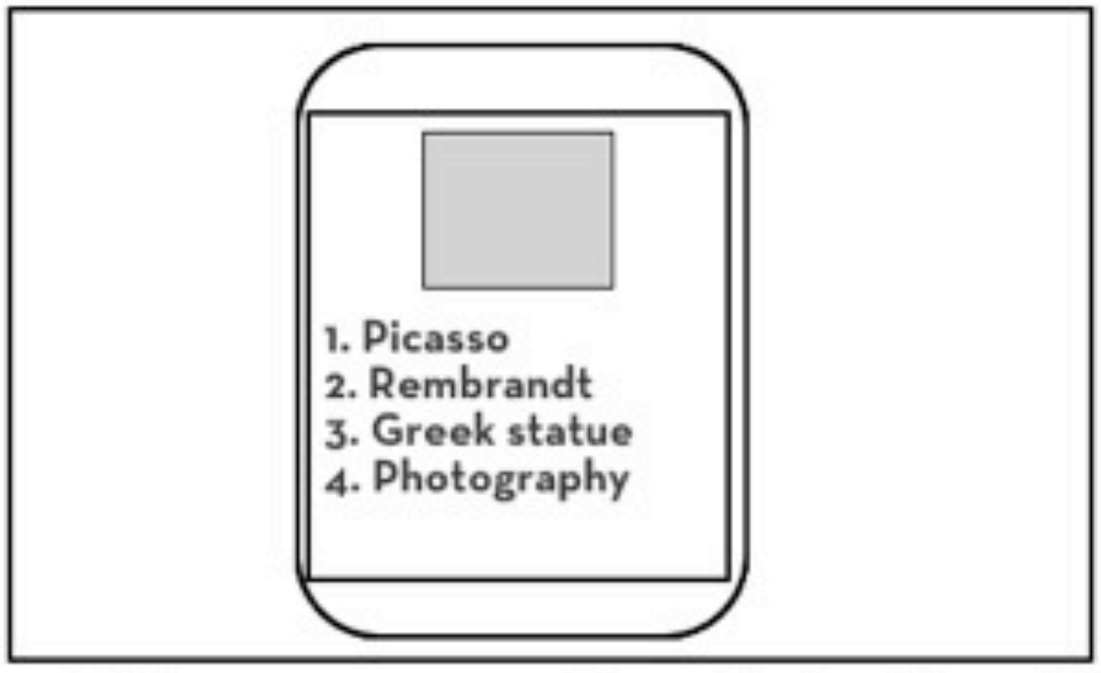


the user arrives at the museum

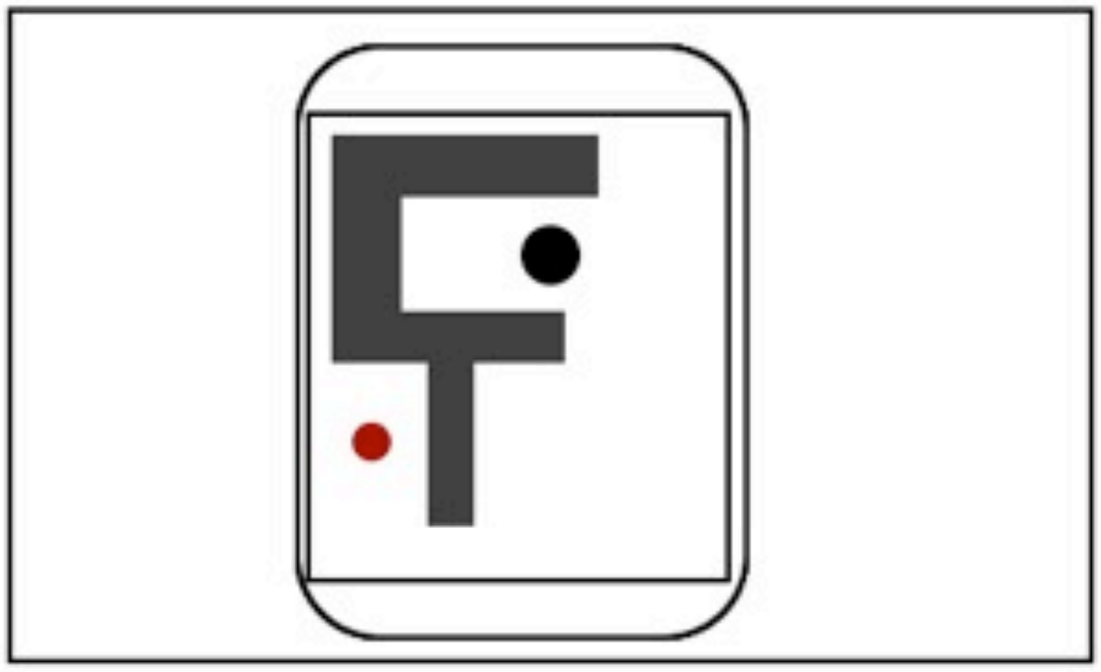


her iPhone presents a list of exhibits

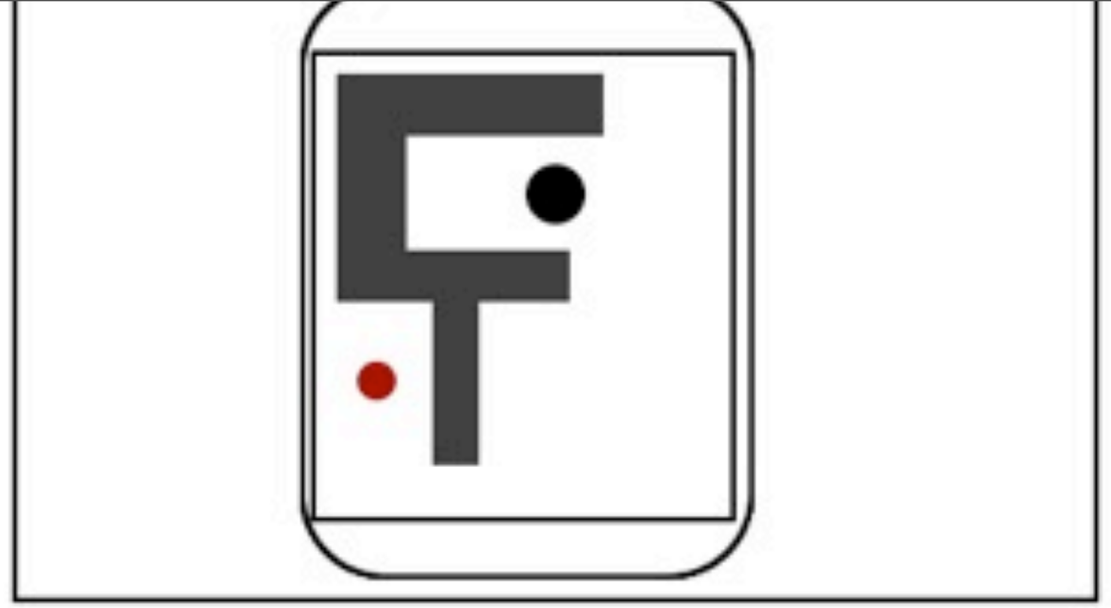
the user arrives at the museum



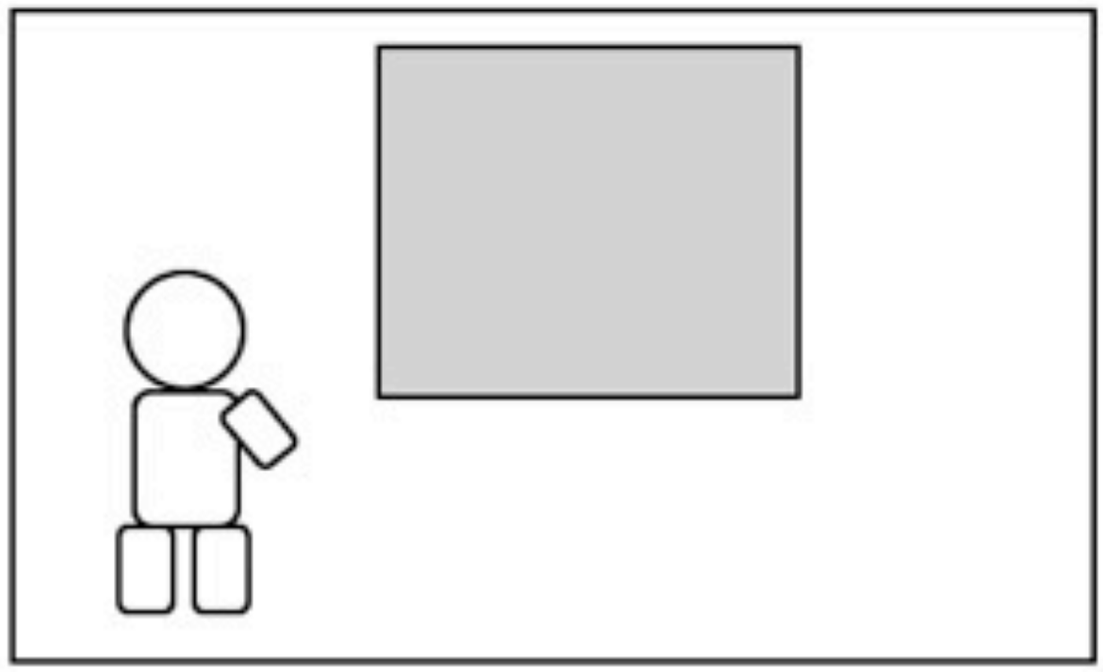
her iPhone presents a list of exhibits



a map is presented, leading to the chosen exhibit

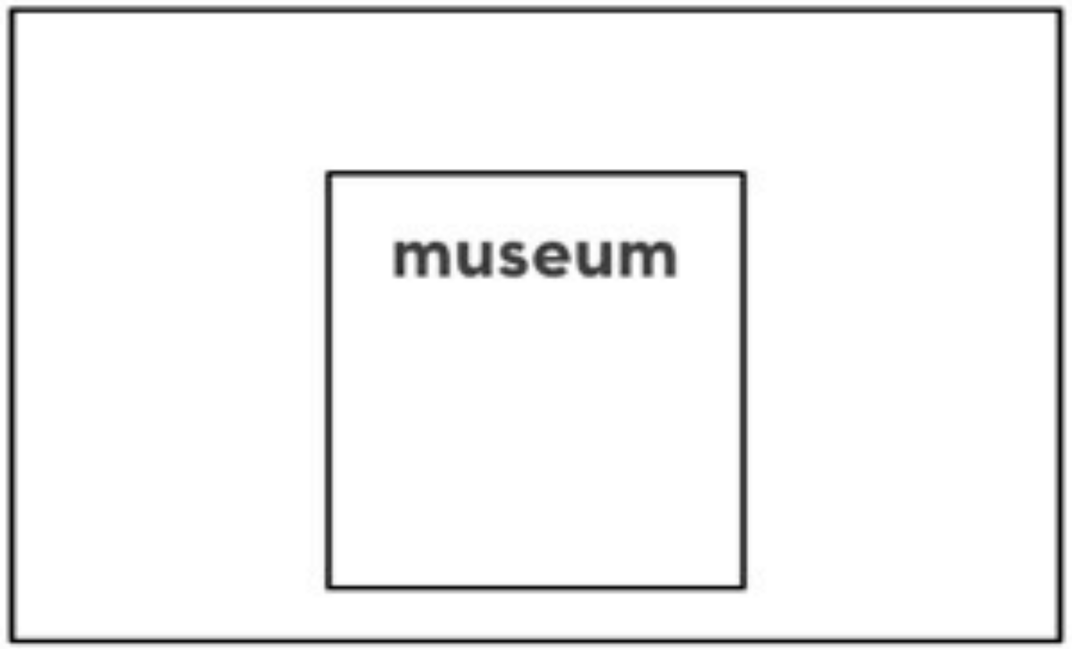


a map is presented, leading to the chosen exhibit

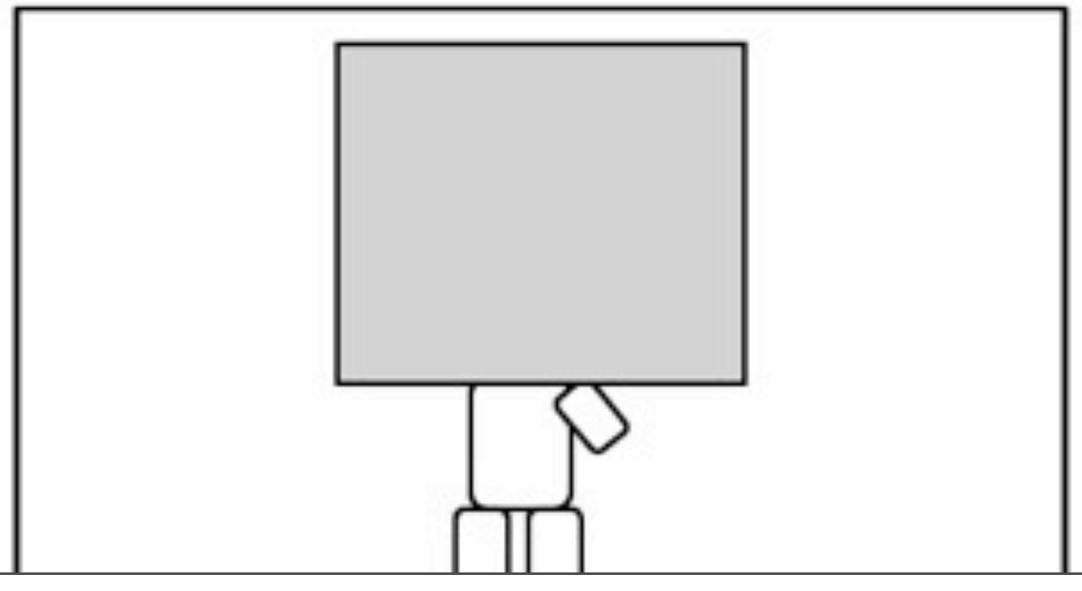


the user walks to the exhibit she wants to view

Storyboard 2

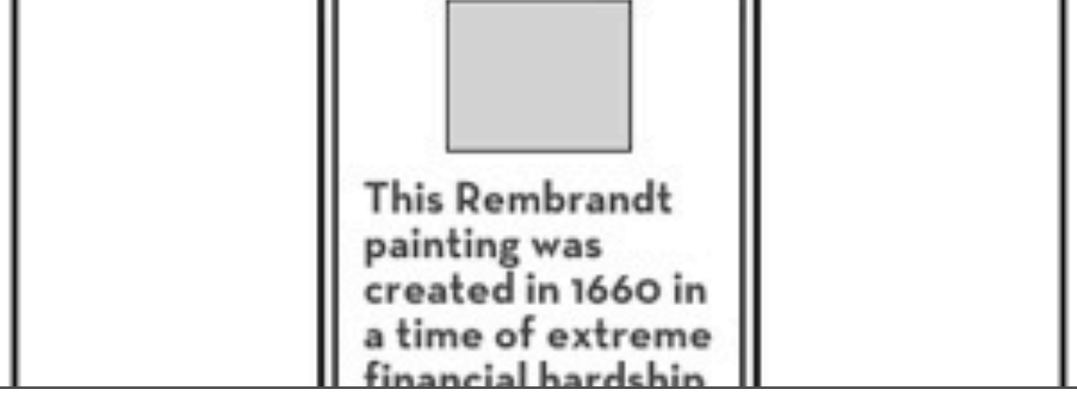


the user arrives at the museum

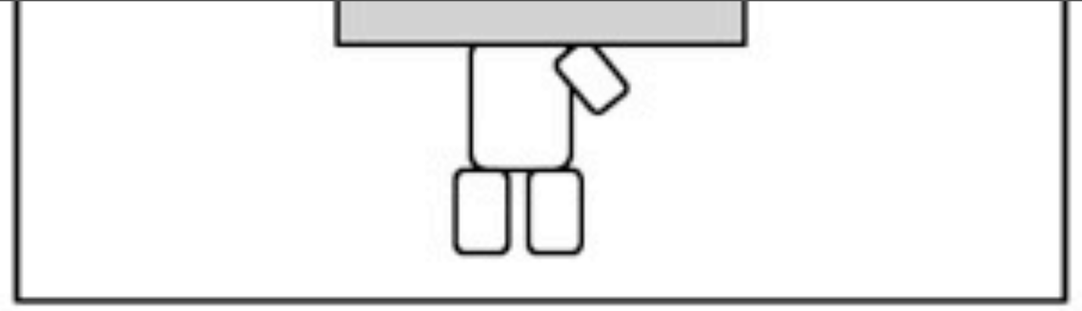


museum

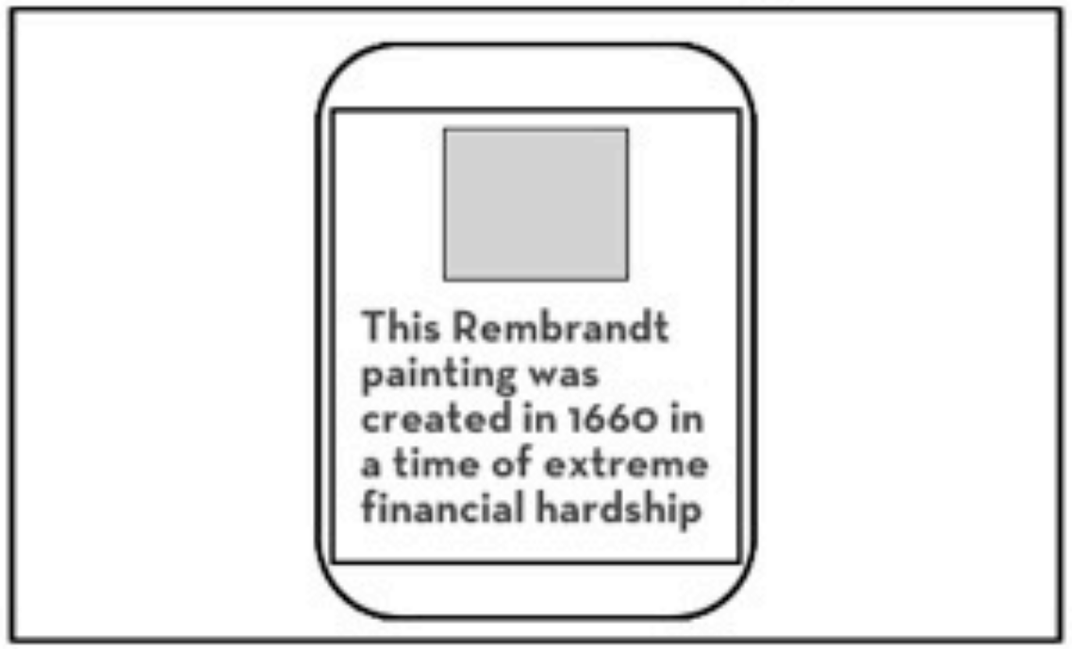
the user arrives at the museum



as the user moves through the exhibit,
her iPhone occasionally vibrates



as the user moves through the exhibit, her iPhone occasionally vibrates



the iPhone displays information relevant to a particular piece of art

Reactions

- “Personally I prefer the idea of storyboard one. This is because the user freely walks around the museum as they would traditionally, yet automatically receive info about exhibits - a virtual guide without user input. Much more impressive.”

- “Personally I think the idea from storyboard one is more compelling. The reason for this is that I would be interested in finding out interesting information about a piece of artwork or a particular artist that I couldn't just get at the museum. The map of the museum is something that I can get at the museum on a piece of paper that doesn't require me to be pulling out my phone and wasting the battery to get to an exhibit.”

Reactions

- "Students who are visiting for a school assignment and have limited time to partake in all the exhibits would definitely find that option helpful."
- "I liked storyboard 2 because of [the use of] cell phones in a physical space."
- Self-reported as non-designers

Final plug

- We're hiring at Meebo!
  - User Experience, Usability, UI...
  - meebo.com/jobs or email me directly at mike.krieger@meebo-inc.com

Thanks!

Q's?