HTML5で

目指せ!
低コストで高機能なWEB開発

第14回　ITテクニカルセミナー

2013/5/31

アジェンダ

OPEN TONE
金融ソリューション事業部

1．自己紹介
2．HTML5とは？
3．Meteorとは？
4．Meteorをさわってみよう
5．Meteorで
　　本格アプリを作ってみよう
6．Meteorおさらい
7．まとめ

自己紹介

OPEN TONE
金融ソリューション事業部

名前：伊東　直喜
所属：株式会社オープントーン
　　金融ソリューション事業部
業界：銀行、TV業界 等の開発に関わってきました
言語：Java、C、C#、COBOL、Ruby
　　HTML、JavaScript など
メール：itou@opentone.co.jp

自己紹介

OPEN TONE
金融ソリューション事業部

得意料理：極厚肉汁ハンバーグ

最近読んだ本：アジャイルサムライ　西村直人

弱点　：腰

アジェンダ

OPEN TONE
金融ソリューション事業部

1．自己紹介
2．HTML5とは？
3．Meteorとは？
4．Meteorをさわってみよう
5．Meteorで
　　本格アプリを作ってみよう
6．Meteorおさらい
7．まとめ

HTML5とは？

OPEN TONE
金融ソリューション事業部

HTML5って・・・
今までのHTML4と何が違う？

Webページを見ているだけだと分かりにくい…

HTML4

HTML5

でも、出来ることが違う!!

HTML5でできること

OPEN TONE
金融ソリューション事業部

HTML5で追加されるカテゴリの分類

カテゴリ	内容
セマンティックス	HTML5の新タグ、RDFa、マイクロデータ、マイクロフォーマット
オフラインとストレージ	App Cache、Web Storage、Indexed Database API、File API
デバイスアクセス	Geolocation API、マイク・カメラ、アドレス帳・カレンダー、端末の向き
接続性	WebSocket、Server-Sent Events
マルチメディア	audio、video要素、3D、グラフィックス、エフェクト - SVG、canvas要素、WebGL、CSS3 3D
パフォーマンスと統合	Web Workers、XMLHttpRequest Level 2
CSS3	WOFFも含む

HTML5 news

OPEN TONE
金融ソリューション事業部

2012/8/16　Windows8発売。Windows8ストアアプリは
HTML5/JavaScriptで開発が可能

2012/12/17 W3CがHTML5仕様策定完了
勧告候補に

2013年　アプリをHTML5で開発するFirefoxOS
Tizenがリリース（既にAndroid,iPhone
はHTML5ハイブリッドアプリが流通）

2014年　HTML5が勧告予定

HTML5を取り巻く環境は整ってきている

HTML5でMeteor

OPEN TONE
金融ソリューション事業部

でも、HTML5でWebアプリケーションを
作るには、それなりにコストがかかる・・・

そんなアナタに **Meteor!!**
開発を「高機能」＆「低コスト」で

アジェンダ

OPEN TONE
金融ソリューション事業部

1．自己紹介
2．HTML5とは？
3．Meteorとは？
4．Meteorをさわってみよう
5．Meteorで
　　本格アプリを作ってみよう
6．Meteorおさらい
7．まとめ

Meteorとは？

OPEN TONE
金融ソリューション事業部

Meteorとは、下記７つのコンセプトに開発が行われているフレームワークです。

Structuring your app →アプリケーションの構造化
Data and security →データ同期
Reactivity →反応性
Live HTML →動的HTML
Templates →テンプレート機能
Smart packages →ライブラリ統合
Deploying →簡易なデプロイ

Meteorとは？

OPEN TONE
金融ソリューション事業部

・技術
　ベースはNode.js(サーバサイドJavaScript)

　データベースはMongoDB
　(オープンソースのドキュメント志向データベース)

・言語
　サーバ／クライアントともにJavaScriptで記述

Meteorとは？

OPEN TONE
金融ソリューション事業部

・メリット
　環境構築が容易（DB周りの設定が容易）
　外部ライブラリが使用できる
　サーバと各クライアントとのデータ同期が容易

・少しデメリット
　まだ正式リリースされていない。
　（しかし、もうじきリリースされる予定）

Meteor インストール

OPEN TONE
金融ソリューション事業部

ターミナル上で以下のコマンドを打ち込むだけで、
Meteorのインストールが完了します。

```
$ curl install.meteor.com | sh
```

すぐに使えます！

アジェンダ

OPEN TONE
金融ソリューション事業部

1．自己紹介
2．HTML5とは？
3．Meteorとは？
4．Meteorをさわってみよう
5．Meteorで
　　本格アプリを作ってみよう
6．Meteorおさらい
7．まとめ

チャットアプリ　デモ

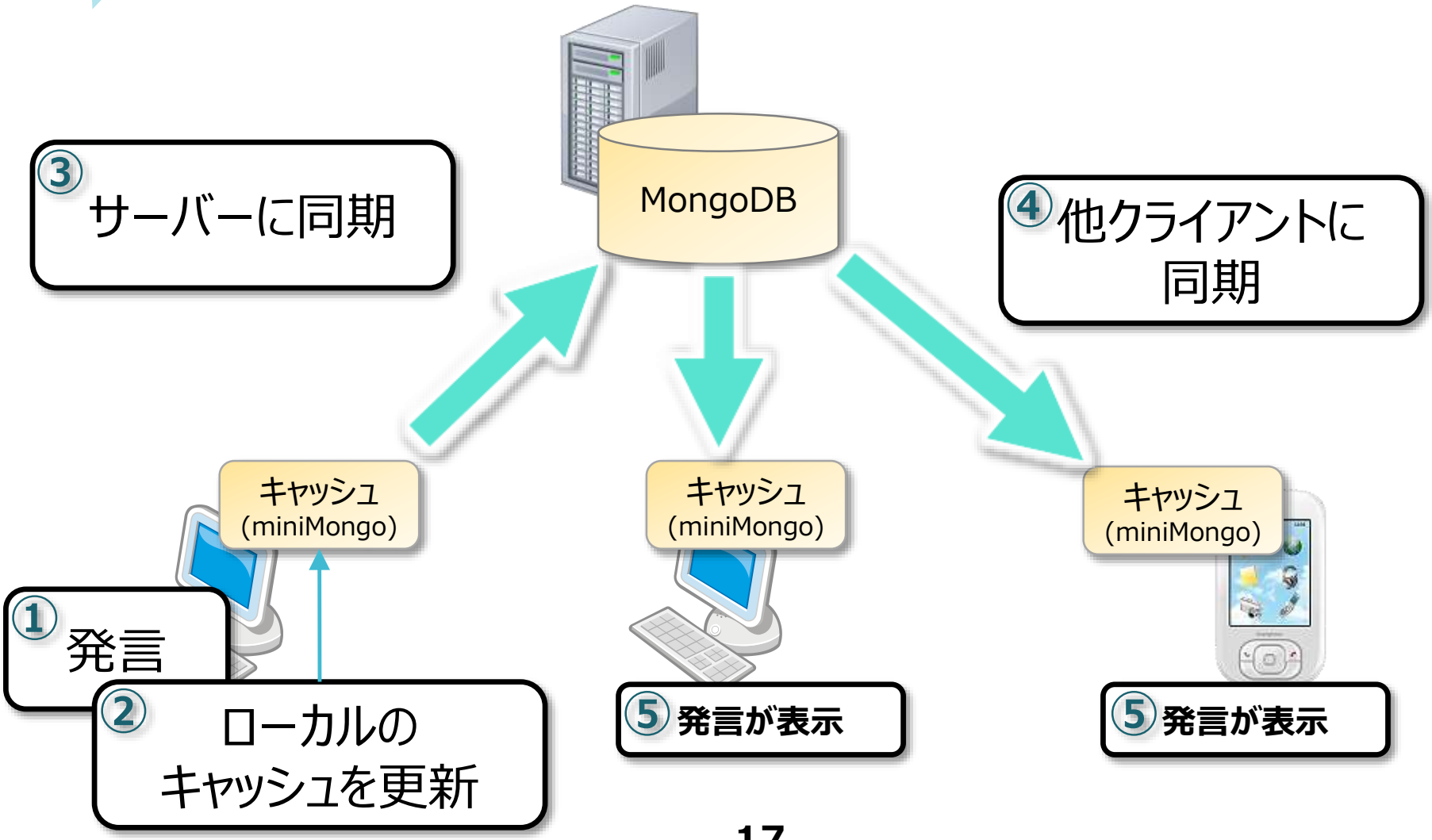
OPEN TONE
金融ソリューション事業部

デモURL http://ot-sample-chat.meteor.com/

よろしければ、URLもしくはQRコードからご参加ください。

Meteor Reactivity

OPEN TONE
金融ソリューション事業部



チャットデモでの使用技術

OPEN TONE
金融ソリューション事業部

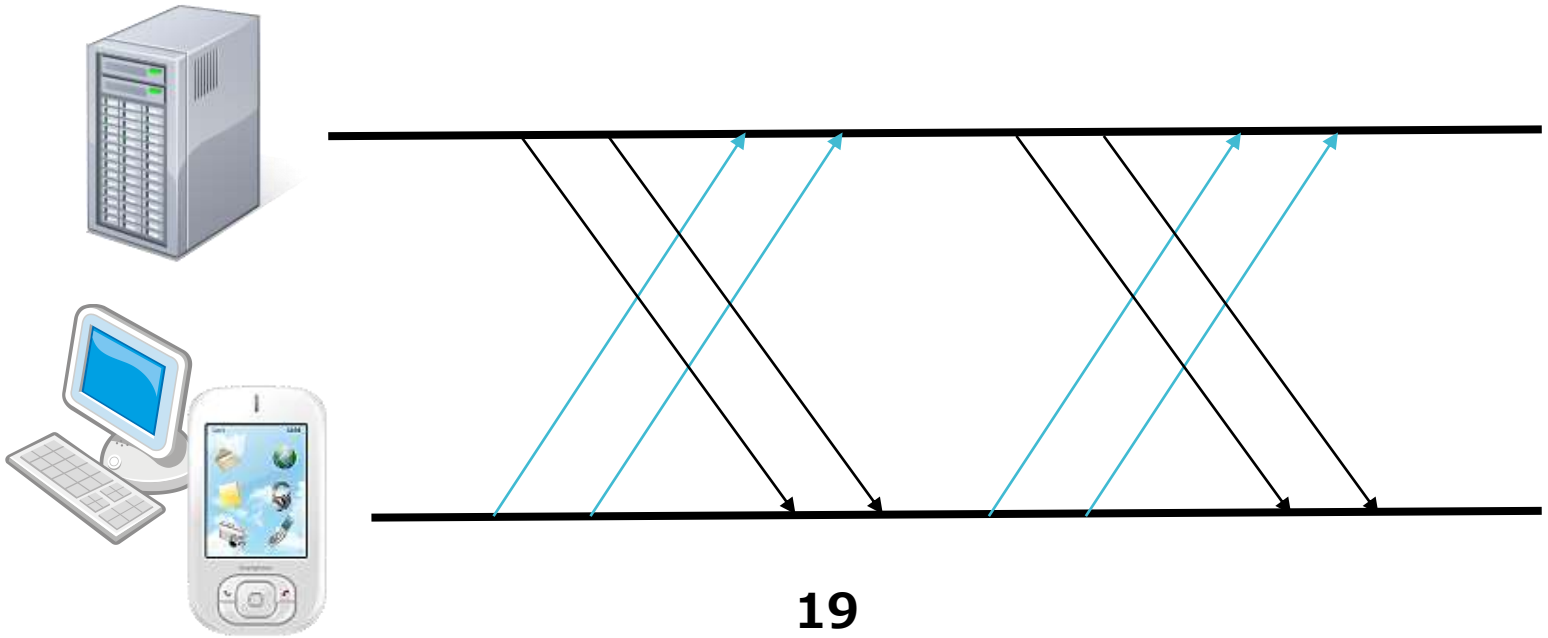
HTML5で追加されるカテゴリの分類

カテゴリ	内容
セマンティックス	HTML5の新タグ、RDFa、マイクロデータ、マイクロフォーマット
オフラインとストレージ	App Cache、Web Storage、Indexed Database API、File API
デバイスアクセス	Geolocation API、マイク・カメラ、アドレス帳・カレンダー、端末の向き
接続性	**WebSocket**、Server-Sent Events
マルチメディア	audio、video要素、3D、グラフィックス、エフェクト - SVG、canvas要素、WebGL、CSS3 3D
パフォーマンスと統合	Web Workers、XMLHttpRequest Level 2
CSS3	WOFFも含む

WebSocket

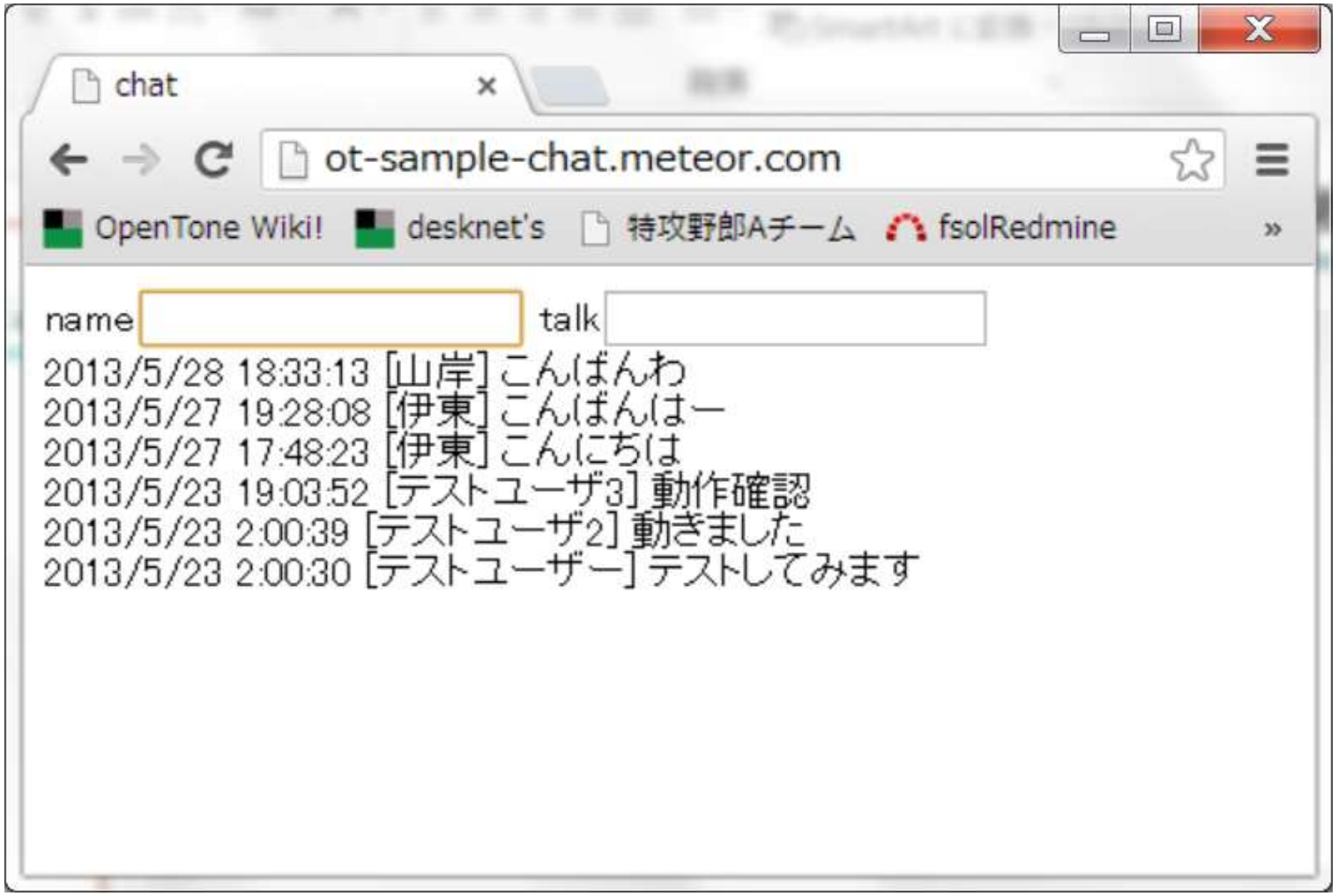
OPEN TONE
金融ソリューション事業部

・WebSocketでは双方向(サーバ/クライアント)
からデータを送受信できる。（Meteor内部で実施）
・同じリアルタイム通信である、Ajax(Googleマップ等で使用)では、クライアントからリクエストがない限り更新しない。



チャットデモのソース

OPEN TONE
金融ソリューション事業部



HTMLファイル

OPEN TONE
金融ソリューション事業部

```
<body>
  {{> form}}
  {{> log}}
</body>
```

<template name="log">タグの内容を表示する

```
<template name="log">
<div>
{{#each logs}}
  <span>{{displayDate}} [{{name}}] {{talk}}</span><br />
{{/each}}
</div>
</template>
```

「Template.log.logs」オブジェクトとしてアクセスできる

JavaScriptファイル

OPEN TONE
金融ソリューション事業部

```
Template.log.logs = function() {
  return Logs.find({}, {sort: {date: -1}}).fetch();
}

Template.log.displayDate = function() {
  return new Date(this.date).toLocaleString()
}
```

降順

Template.log.logsオブジェクトはMongoDBの
Logsコレクションと永続化されています。
（nameやtalkなどのKEYを持っています）

チャットアプリを作ろう

OPEN TONE
金融ソリューション事業部

ライブコーディング

Hello　World!作成　　（１分）
↓
HTMLファイル編集　　（４分）
↓
JavaScriptファイル編集（５分）

Meteorは低コスト

OPEN TONE
金融ソリューション事業部

今回のチャットデモを作る場合……

比較項目	Meteor以外	Meteor
開発対象アプリ	サーバとクライアントの2つ	**1つのみ**
同期処理	イベント発生時に動くような同期処理を実装	**実装する必要なし**
DB処理	別途DBサーバを用意し、接続処理を実装	**事前準備ほぼ不要で、MongoDBへのread/writeが可能**

アジェンダ

OPEN TONE
金融ソリューション事業部

1．自己紹介
2．HTML5とは？
3．Meteorとは？
4．Meteorをさわってみよう
5．Meteorで本格アプリを作ってみよう
6．Meteorおさらい
7．まとめ

アンケートアプリ　デモ

OPEN TONE
金融ソリューション事業部

デモURL http://ot-sample-enquete.meteor.com/

アンケートデモでの使用技術

OPEN TONE
金融ソリューション事業部

HTML5で追加されるカテゴリの分類

カテゴリ	内容
セマンティックス	HTML5の新タグ、RDFa、マイクロデータ、マイクロフォーマット
オフラインとストレージ	App Cache、Web Storage、Indexed Database API、File API
デバイスアクセス	Geolocation API、マイク・カメラ、アドレス帳・カレンダー、端末の向き
接続性	**WebSocket**、Server-Sent Events
マルチメディア	audio、video要素、3D、グラフィックス、エフェクト - **SVG、canvas要素**、WebGL、CSS3 3D
パフォーマンスと統合	Web Workers、XMLHttpRequest Level 2
CSS3	WOFFも含む

マップアプリ　デモ

OPEN TONE
金融ソリューション事業部

デモURL http://ot-sample-map.meteor.com/

目的

ある場所を目的としたユーザが
リアルタイムに情報共有すること

アプリ画面イメージ

OPEN TONE
金融ソリューション事業部

目的地

- セミナー会場
- 懇親会会場

Xさん発言1
Yさん発言1
Zさん発言1
Zさん発言2
Yさん発言2
Zさん発言3

参加者 2 人
名前:
発言: 発言
lat: 35.685187 lng: 139.753763 指定の場所を公開する 現在位置を公開する

name	comment	date

アプリ画面イメージ

OPEN TONE
金融ソリューション事業部

登録されている目的地を表示

目的地
- セミナー会場
- 懇親会会場

参加者 2 人
名前:
発言: 発言
lat: 35.685187 lng: 139.753763 指定の場所を公開する 現在位置を公開する

name	comment	date

アプリ画面イメージ

OPEN TONE
金融ソリューション事業部

発言情報を表示するマップ

目的地

- セミナー会場
- 懇親会会場

参加者 2 人
名前:
発言:
発言
lat: 35.685187 lng: 139.753763
指定の場所を公開する
現在位置を公開する

name	comment	date

アプリ画面イメージ

OPEN TONE
金融ソリューション事業部

発言のログを表示

地図アプリでの使用技術

OPEN TONE
金融ソリューション事業部

HTML5で追加されるカテゴリの分類

カテゴリ	内容
セマンティックス	HTML5の新タグ、RDFa、マイクロデータ、マイクロフォーマット
オフラインとストレージ	App Cache、Web Storage、Indexed Database API、File API
デバイスアクセス	**Geolocation API**、マイク・カメラ、アドレス帳・カレンダー、端末の向き
接続性	**WebSocket**、Server-Sent Events
マルチメディア	audio、video要素、3D、グラフィックス、エフェクト - SVG、canvas要素、WebGL、CSS3 3D
パフォーマンスと統合	Web Workers、XMLHttpRequest Level 2
CSS3	WOFFも含む

アプリ例

OPEN TONE
金融ソリューション事業部

大規模鬼ごっこアプリ

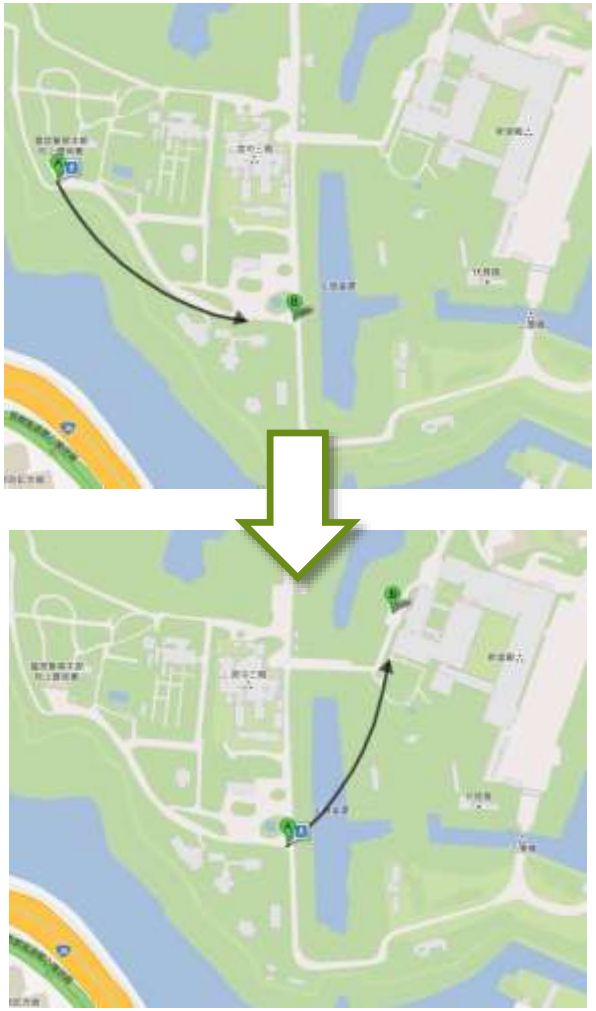
- 子は鬼に定期的に自分の位置情報を提供
- 鬼は、子の動きから行動範囲を予測
- 鬼達が連携して対象とする子を捕まえる
  （参考：TOKIOの鉄腕ダッシュ）

戦略的で　大規模な鬼ごっこが実現できる

イメージ[鬼の端末]

OPEN TONE
金融ソリューション事業部

皇居で鬼ごっこ



アプリ例

防災アプリ

・被害状況、トイレ貸出、施設の臨時提供
　などの情報発信
・社員同士の安否確認や情報共有など

アプリイメージ

OPEN TONE
金融ソリューション事業部

アプリイメージ

OPEN TONE
金融ソリューション事業部

アプリイメージ

OPEN TONE
金融ソリューション事業部

アジェンダ

OPEN TONE
金融ソリューション事業部

1．自己紹介
2．HTML5とは？
3．Meteorとは？
4．Meteorをさわってみよう
5．Meteorで
　　本格アプリを作ってみよう
6．Meteorおさらい
7．まとめ

Meteorおさらい

OPEN TONE
金融ソリューション事業部

環境構築が容易（DB周りの設定が容易）
→インストール:$ curl install.meteor.com | sh
デプロイ:$ meteor deploy ot-sample-chat
⇒http://ot-sample-chat.meteor.comに公開

外部JavaScriptライブラリが使用できる
→d3.jsなど

サーバと各クライアントとのデータ同期が容易
→WebSocketを使った通信・MongoDBによる同期

低コストで高機能なWeb開発が可能

アジェンダ

OPEN TONE
金融ソリューション事業部

1．自己紹介
2．HTML5とは？
3．Meteorとは？
4．Meteorをさわってみよう
5．Meteorで
　　本格アプリを作ってみよう
6．Meteorおさらい
7．まとめ

まとめ

OPEN TONE
金融ソリューション事業部

Meteorで作成したアプリは
HTML5対応ブラウザで動作可能

クロスプラットフォームで動作可能

低コストで高機能なWeb開発が可能！

オプトンラボご紹介

OPEN TONE
金融ソリューション事業部

今回ご紹介した技術ネタ以外に
マネジメント系など、多数紹介しております。
ぜひ、ご覧ください。

http://labs.opentone.co.jp/

チャットアプリ HTML

```
<head>
  <title>chat</title>
</head>

<body>
  {{> form}}
  {{> log}}
</body>

<template name="form">
<div>
 name<input type="text" id="name" />
 talk<input type="text" id="talk" />
</div>
</template>

<template name="log">
<div>
{{#each logs}}
  <span>{{displayDate}} [{{name}}] {{talk}}</span><br />
{{/each}}
</div>
</template>

[EOF]
```

チャットアプリ JavaScript

```
Logs = new Meteor.Collection("logs");

if (Meteor.isClient) {
  Template.form.events({
    'keyup input#talk' : function (e){
      if (e.which != 13) return;

      name = $("input#name").val();
      talk = $("input#talk").val();

      Logs.insert({"name":name, "talk":talk, "date":new Date().getTime()});

      $("input#talk").val("");
      $("input#talk").focus();
    }
  });

  Template.log.logs = function() {
    return Logs.find({}, {sort: {date: -1}}).fetch();
  }

  Template.log.displayDate = function() {
    return new Date(this.date).toLocaleString();
  }
}

if (Meteor.isServer) {
  Meteor.startup(function () {
  });
}
```

地図アプリ HTML 全119行(コメント含む)

```
<head>
  <title>AreaSample</title>
  <script type="text/javascript" src="http://maps.google.com/maps/api/js?key=
AIzaSyAKaMB4rFH4AtpV91paRkHrm8i_bSlHPm8&sensor=true">
  <script language="javascript"></script>
</head>

<body>
  <h4>Map</h4>
  <table width="100%">
    <tr>
      <td width="50%">{{> rooms}}</td>
      <td>{{> story}}</td>
    </tr>
  </table>
  {{> users}}
  {{> input}}
  {{> geo}}
  <table width="100%">
    <tr>
      <td width="80%">
        <div id="map_canvas" style="width:100%; height:450px"></div>
      </td>
      <td style="vertical-align: top; padding-left: 10px;">
        {{> comments}}
      </td>
    </tr>
  </table>
</body>

<template name="rooms">
  <div>目的地</div>
  <ul class="menu">
    {{#each rooms}}
    <li class="{{selected}}"><a href="#" data-name="{{name}}">{{display}}</a></
li>
    {{/each}}
  </ul>
  <br />
</template>

<template name="users">
  <div class="users">
    <span>参加者 {{count}} 人</span><br />
  </div>
  {{#each users}}
    {{> user}}
  {{/each}}
</template>

<template name="user">
  {{marker}}
</template>

<template name="input">
  <div class="input">
    名前：<input type="input" id="name" value=""/><BR>
    発言：<input type="input" id="text" size="80"/>
    <input type="button" id="submit" value="発言" />
  </div>
</template>

<template name="comments">
  <table cellspacing="0" cellpadding="0" border=1>
  <thead>
  <tr>
    <th nowrap>name</th>
```

全体のレイアウト定義

目的地のレイアウト定義

ユーザのレイアウト定義

入力(名前、発言)のレイア

```
66      <th nowrap>comment</th>
67      <th nowrap>date</th>
68    </tr>
69    </thead>
70    <tbody>
71      {{#each comments}}
72        {{> comment}}
73      {{/each}}
74    </tbody>
75  </table>
76  </template>
77
78  <template name="comment">
79  <tr>
80    <td><a href="#" id="open-comment" data-id="{{_id}}">{{name}}</a></td>
81    <td>{{text}}</td>
82    <td>{{date}}</td>
83  </tr>
84  </template>
85
86  <template name="geo">
87    <div class="input">
88      lat : <input type="input" id="lat" value="{{geo_lat}}"/>
89      lng : <input type="input" id="lng" value="{{geo_lng}}"/>
90      <input id="set-position" type="button" value="指定の場所を公開する" />
91      <input id="set-current-position" type="button" value="現在位置を公開する" />
92    </div>
93  </template>
94
95
96  <template name="story">
97    <div class="story">
98      <table>
99        <tr>
100         <td><a href="#" id="story1" data-lat="35.695853" data-lng="139.724409" data-name="X" data-text="現場到着。みんなの状況は？">Xさん発言1</a></td>
101       </tr>
102       <tr>
103         <td><a href="#" id="story2" data-lat="35.693204" data-lng="139.724121" data-name="Y" data-text="予定通り到着できそうです。">Yさん発言1</a></td>
104       </tr>
105       <tr>
106         <td><a href="#" id="story3" data-lat="35.690834" data-lng="139.707084" data-name="Z" data-text="電車遅延のためまだ新宿三丁目です。電車が動きだしたらまた連絡します。">Zさん発言1</a></td>
107       </tr>
108       <tr>
109         <td><a href="#" id="story4" data-lat="35.691531" data-lng="139.708586" data-name="Z" data-text="電車が動きだしました。この分だとXX時ぐらいになりそうです。">Zさん発言2</a></td>
110       </tr>
111       <tr>
112         <td><a href="#" id="story5" data-lat="35.695225" data-lng="139.724448" data-name="Y" data-text="現場に到着しました。">Yさん発言2</a></td>
113       </tr>
114       <tr>
115         <td><a href="#" id="story6" data-lat="35.692838" data-lng="139.723327" data-name="Z" data-text="やっと最寄駅に着きました。">Zさん発言3</a></td>
116       </tr>
117     </table>
118   </div>
119 </template>
```

発言一覧のレイアウト定

位置情報のレイアウト定

入力値代入のレイアウト定義

地図アプリ JavaScript 全329行(コメント含む)

```
1  User = new Meteor.Collection("users")
2  Comments = new Meteor.Collection("comments");
3  //Rooms = new Meteor.Collection("rooms");
4  map = null;
5  markers = {};
6  commentMarkers = {};
7  info_windows = {};
8  main_info_windows = {};
9
10
11 if (Meteor.is_client) {
12   Meteor.startup(function () {
13     map = new google.maps.Map(document.getElementById("map_canvas"),
14       {zoom: 13,
15       center: new google.maps.LatLng(35.685187, 139.753763),
16       mapTypeId: google.maps.MapTypeId.ROADMAP});
17
18     count = User.find({_id: Session.get("user_id")}).count();
19     if (count == 0) {
20       user_id = User.insert({last_keepalive: new Date().getTime(), name: '',
   refresh: false});
21       Session.set("user_id", user_id);
22     }
23   });
24
25   // 1分毎にkeepaliveを更新
26   Meteor.setInterval(function () {
27     User.update({_id: Session.get('user_id')}, {$set: {last_keepalive: (new
   Date()).getTime(), refresh: false}});
28     console.log('refresh')}, (60 * 1000)
29   );
30
31 // Template function
32   Template.users.count = function() {
33     return User.find().count();
34   }
35
36 //** rooms
37   Template.rooms.selected = function() {
38     return current_room() == this.name ? "current" : ""
39   }
40
41 //  Template.rooms.rooms = function() {
42 //    if (localRoom.length == 0) {
43 //      _rooms = Rooms.find({});
44 //      if (_rooms.count() != 0) {
45 //        _rooms.forEach(addRoom)
46 //        localRoom = _rooms.fetch();
47 //      }
48 //    }
49 //    return localRoom;
50 //  }
51   troom = {
52             'rooma' : { name : 'rooma', display: 'セミナー会場', title: '勉強会',
    lat: 35.695853, lng: 139.724409, comment: '開催日: 5月31日 (金) <br>13:00～18:
   00<br>会場:健保会館市ヶ谷会議室 F室', zindex: 5000},
53             'roomb' : { name : 'roomb', display: '懇親会会場', title: '勉強会',
   lat: 35.691701, lng: 139.73729, comment: '開催日: 5月31日 (金) <br>18:30～<br>
   会場:花びし', zindex: 5001}
54           };
55   Template.rooms.rooms = [
56                           troom['rooma'],
57                           troom['roomb'],
58                          ];
59
60   Template.rooms.sessionid = function() {
61     return Session.get("user_id");
62   }
```

初期化処理
約10行

クライアント初期処理
約20行

目的地に関する処理
約40行

```
63
64    Template.rooms.events = {
65      'click a': function(evt) {
66
67        deleteRoom();
68        clearOverlays();
69        deleteCommentInfo();
70        Session.set("current_room", $(evt.target).data("name"));
71        addRoom(troom[$(evt.target).data("name")]);
72
73      }
74    };
75
76  //** input
77
78    Template.users.users = function() {
79      return User.find({refresh: true});
80    }
81
82    Template.user.marker = function() {
83  //    if (!map || !this.lng || this.lat) {
84  //      return;
85  //    }
86
87      position = new google.maps.LatLng(this.lat, this.lng);
88
89  //    if (Session.get("user_id") == this._id) {
90  //      markerName = localStorage.getItem("name");
91  //    } else {
92        markerName = this.name;
93  //    }
94      if (markers[this._id]) {
95        markers[this._id].setPosition(position);
96        markers[this._id].setMap(map);
97  //      if (info_windows[Session.get("current_room")+this._id]) {
98  //        info_windows[Session.get("current_room")+this._id].close();
99  //      }
100       console.log("markers user_id:" + this._id);
101     }
102     else if (markerName != ''){
103       var icon = 'http://chart.apis.google.com/chart?chst=d_map_pin_letter&
    chld=' + markerName + '|7FFF00|000000';
104       markers[this._id] = new google.maps.Marker({position: position, map: map,
     icon: icon});
105       info_windows[Session.get("current_room")+this._id] = new google.maps.
    InfoWindow();
106       console.log("new markers user_id:" + this._id + " name:" + markerName);
107     }
108   };
109
110   Template.input.events = {
111     'click input#submit, keyup input#text' : function (evt) {
112       //  Returnキー以外は何もしない
113       //if(evt.type == "keyup" && evt.which != 13) return;
114       //暫定でkeyup禁止
115       if(evt.type == "keyup") return;
116
117       name = $("input#name").val();
118       text = $("input#text").val();
119
120       if(name == "" || text == "") return;
121
122       id = Session.get("user_id");
123 //      if (info_windows[Session.get("current_room")+id]) {
124 //        msg = name + "さんは書きました:<br>" + text;
125 //        info_windows[Session.get("current_room")+id].setContent(msg);
126 //        info_windows[Session.get("current_room")+id].open(map, markers[id]);
127 //      }
128
129       //位置情報を取得
```

ユーザに関する処理
約30行

入力処理に関する処理
約40行

```
130       var geo = null;
131       if (Session.get('lastGeo')) {
132         geo = Session.get('lastGeo');
133       } else {
134         geo = {lat: $("input#lat").val(), lng: $("input#lng").val()};
135       }
136       console.log("geolocation" + geo['lat'] + " : " +  geo['lng']);
137
138       Comments.insert({
139         room: current_room(),
140         user_id: id,
141         name: name,
142         text: text,
143         date: new Date().getTime(),
144         lat:  geo['lat'],
145         lng: geo['lng']});
146       $("input#text").val("");
147       $("input#text").focus();
148     }
149
150
151   };
152 //** geo
153   Template.geo.events = {
154     'click #set-position': function () {
155       lat = $("input#lat").val();
156       lng = $("input#lng").val();
157         name = $("input#name").val();
158         if (name == "") {
159           alert('名前を入れください');
160           return;
161         }
162       User.update({_id: Session.get('user_id')}, {$set: {lat: lat, lng: lng,
name: name, refresh: true}});
163       Session.set('lastGeo', {lat: lat, lng: lng});
164     },
165
166     'click #set-current-position': function () {
167       navigator.geolocation.getCurrentPosition( function(geo) {
168         name = $("input#name").val();
169         if (name == "") {
170          alert('名前を入れください');
171         }
172         User.update({_id: Session.get('user_id')}, {$set: {lat: geo.coords.
latitude, lng: geo.coords.longitude, name: name, refresh: true}});
173         Session.set('lastGeo', {lat: geo.coords.latitude, lng: geo.coords.
longitude});
174       })
175     }
176   };
177
178   Template.comments.comments = function() {
179     comments = Comments.find({room: current_room()}, {sort: {date: -1}}).fetch(
);
180     //最新5件
181     newComments = comments.slice(0, 5);
182     executedFlg = new Array();
183     for (var i = 0; i < newComments.length; i++) {
184       user_id = newComments[i].user_id;
185
186       if (!(user_id in executedFlg)
187           && markers[user_id]) {
188         executedFlg[user_id] = true;
189         pop(newComments[i], 5 - i);
190       }
191
192     }
193     return newComments;
194     //return comments;
195   };
```

位置情報に関する処理
約25行

発言一覧に関する処理
約30行

```
196
197   Template.comment.date = function() {
198     return new Date(this.date).format("yyyy/mm/dd HH:MM:ss")
199   };
200
201   Template.comment.events = {
202     'mouseenter #open-comment': function(evt) {
203       comment = Comments.findOne({_id: $(evt.target).data("id")})
204       onePop(comment);
205     },
206     'mouseleave #open-comment': function() {
207       onePopClear();
208     }
209   };
210
211   Template.geo.geo_lat = function() {
212     return "35.685187";
213   };
214
215   Template.geo.geo_lng = function() {
216     return "139.753763";
217   };
218
219 // ** story
220   Template.story.events = {
221     'click a': function(evt) {
222
223       $("input#lat").val($(evt.target).data("lat"));
224       $("input#lng").val($(evt.target).data("lng"));
225       $("input#name").val($(evt.target).data("name"));
226       $("input#text").val($(evt.target).data("text"));
227     }
228   };
229
230 // private function
231   function addRoom(room) {
232     console.log(room.title + " " + room.comment + " " + room.lat + " " + room.
    lng);
233
234     position = new google.maps.LatLng(room.lat, room.lng);
235     main_info_windows[room._id] = new google.maps.InfoWindow({
236     content: '<div style="background-color:#ffe0e0;"><strong>- ' + room.title +
     ' -<br>' + room.comment + '</strong></div>',
237     position: position,
238     zIndex: room.zindex});
239     main_info_windows[room._id].open(map);
240   }
241
242   function deleteRoom(room) {
243     for (var i in main_info_windows){
244      main_info_windows[i].close();
245     }
246   }
247
248   function clearOverlays() {
249     if (markers) {
250       for (i in markers) {
251         markers[i].setMap(null);
252       }
253     }
254   }
255
256   function deleteCommentInfo() {
257     for (var i in info_windows){
258      info_windows[i].close();
259     }
260   }
261
262   function current_room() {
263     console.log("current_room:" + Session.get("current_room"));
```

入力値代入に関する処理
約20行

共通処理
（噴出しやマーカーに関するgoogle Map API）

約80行

```
264     return Session.get("current_room") || Template.rooms.rooms[0].name;
265   }
266
267   function pop(comment, index) {
268         _date = new Date(comment.date);
269         _marker = markers[comment.user_id];
270         if (index != null) {
271           _marker.setZIndex(index);
272         }
273         msg = new Date(_date).format("yyyy/mm/dd HH:MM:ss") + ":" + comment.
name + "さんは書きました:<br>" + comment.text;
274         info_windows[Session.get("current_room")+comment.user_id].setContent(
msg);
275         info_windows[Session.get("current_room")+comment.user_id].setZIndex(
index);
276         info_windows[Session.get("current_room")+comment.user_id].open(map,
_marker);
277         console.log("user_id:" + comment.user_id + " "+comment.name);
278   }
279
280   oneMarker = null;
281   oneInfoWin = null;
282   function onePop(comment) {
283         position = new google.maps.LatLng(comment.lat, comment.lng);
284         icon = 'http://chart.apis.google.com/chart?chst=d_map_pin_letter&chld='
 + comment.name + '|7F3000|FFFFFF';
285         oneMarker = new google.maps.Marker({position: position, map: map, icon:
 icon});
286         _marker = oneMarker;
287         _marker.setZIndex(99999);
288
289         msg = new Date(comment.date).format("yyyy/mm/dd HH:MM:ss") + ":" +
comment.name + "さんは書きました:<br>" + comment.text;
290         oneInfoWin = new google.maps.InfoWindow();
291         oneInfoWin.setContent(msg);
292         oneInfoWin.setZIndex(99999);
293         oneInfoWin.open(map, _marker);
294         console.log("onePop user_id:" + comment.user_id + " "+comment.name + "
" + comment._id);
295   }
296   function onePopClear(){
297     if (oneMarker) {
298       oneMarker.setMap(null);
299       oneMarker = null;
300     }
301     if (oneInfoWin) {
302       oneInfoWin.close();
303       oneInfoWin = null;
304     }
305     console.log("onePopClear");
306   }
307 }
308
309 if (Meteor.is_server) {
310   Meteor.startup(function () {
311     // 初期データ
312 //     if (Rooms.find({}).count() == 0) {
313 //       console.log(Rooms.insert({name: 'rooma', display: 'セミナー会場',
title: '勉強会', lat: 35.695853, lng: 139.724409, comment: '開催日: 5月31日 (金
) <br>13:00～18:00<br>会場:健保会館市ヶ谷会議室 F室', zindex: 5000}));
314 //     }
315   });
316
317
318
319   batch_interval = 60 * 1000;
320
321   Meteor.setInterval(function() {
322     User.remove({last_keepalive: undefined});
323     User.remove({last_keepalive: {$lt: new Date().getTime() - batch_interval *
```

サーバ処理
（バッチ処理）
約20行

```
2}})
324     console.log("server event [user remove]");
325   }, batch_interval
326   );
327 }
328
```