

Google™



# Enterprise Workflow with Apps Script

Eric Zhang and Evin Levey  
May 11<sup>th</sup> 2011

# Enterprise Workflow with Apps Script

Hashtags: [#io2011](#) [#GoogleApps](#)

Feedback: <http://goo.gl/hwvJk>

# Introduction

- Workflow
  - *Management of a sequence of steps in a business process*
  - Examples: vacation request process, document approval
- Google Sites
  - Easy way to create and share web pages, and web sites
- Google Apps Script
  - Google's cloud scripting service
  - Extends and automates Google Apps

# Agenda

- **Script in Sites**
  - The basics
- Simple Workflow
  - Posting private comments to a site owner
- Document Approval
  - A publishing approval workflow
- Advanced Document Approval
  - Multiple approvers, reminders, escalation, etc.

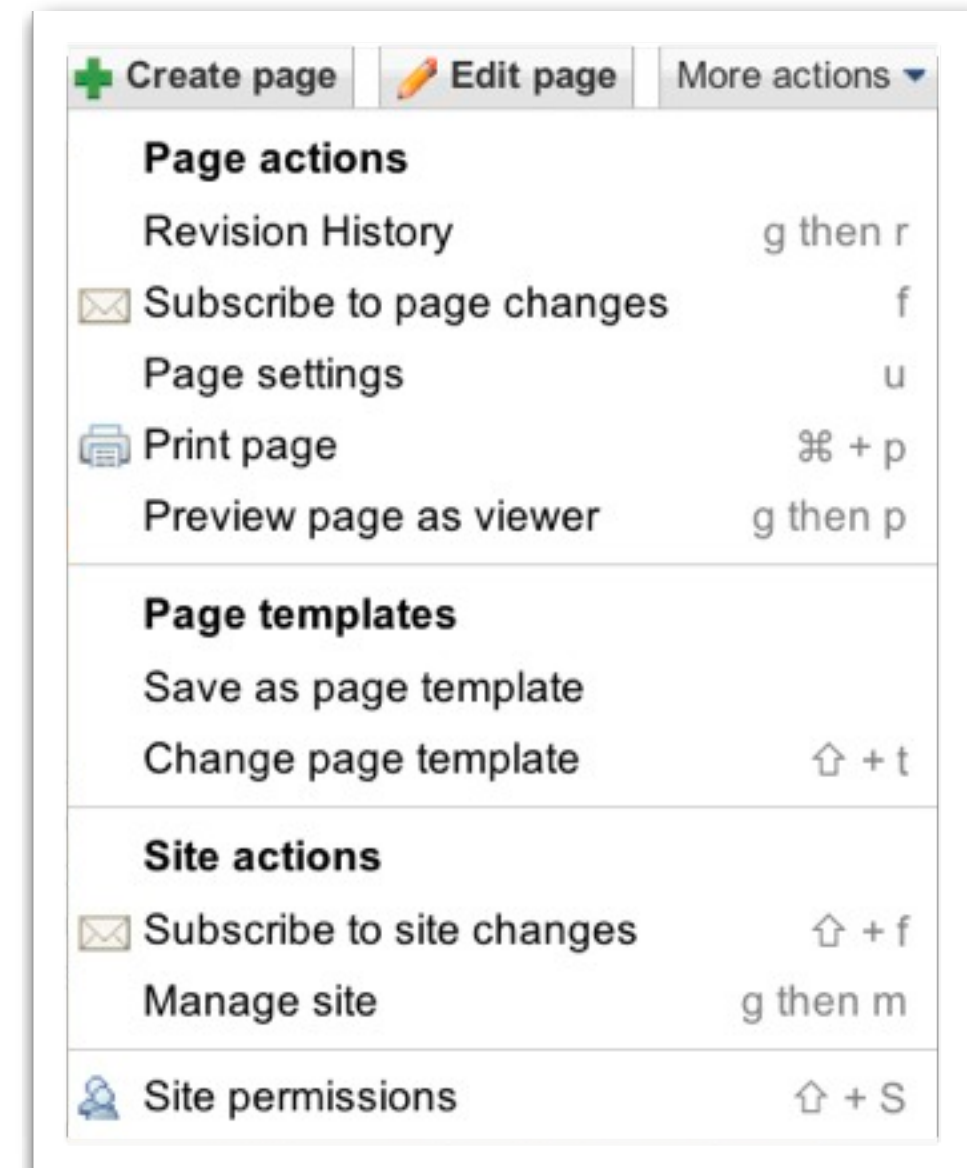
# Script in Sites

- Workflows permeate all apps, but live in Sites
  - HR Site, Team Site, Expense ‘App’
- Google Apps Script is now available in Google Sites
  - Write scripts that interact with a site directly
  - Embed **Script Gadgets** into a page

# Sites Service

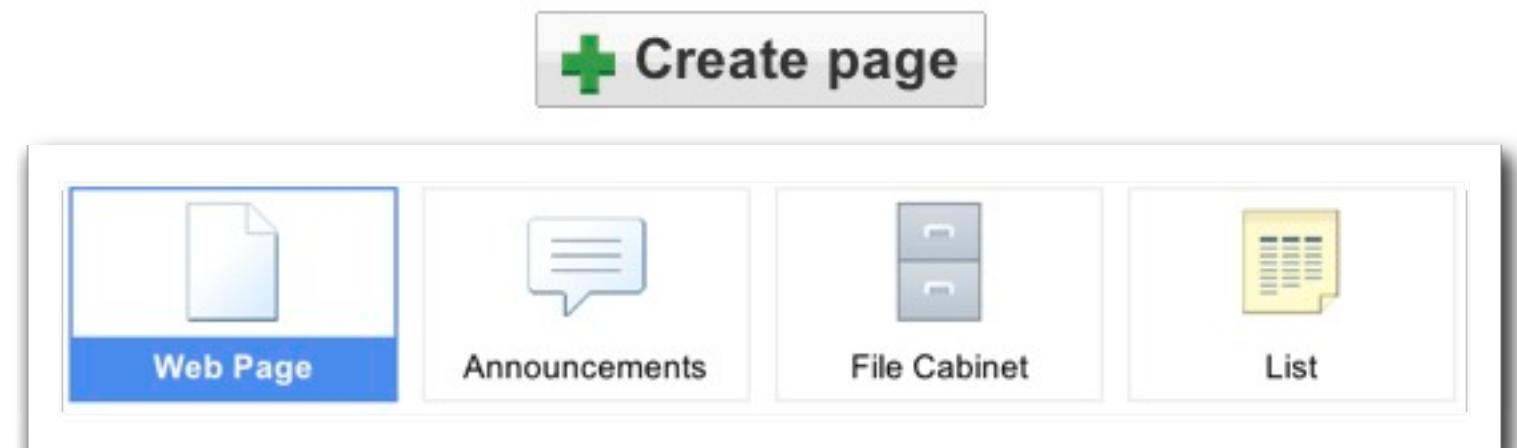
If you use Sites, it's easy!

- Page Creation
- Sharing
- Themes
- Search
- Navigation



# SitesApp

- There are just 4 page types:
  - WebPage
  - ListPage
  - AnnouncementPage
  - FileCabinetPage

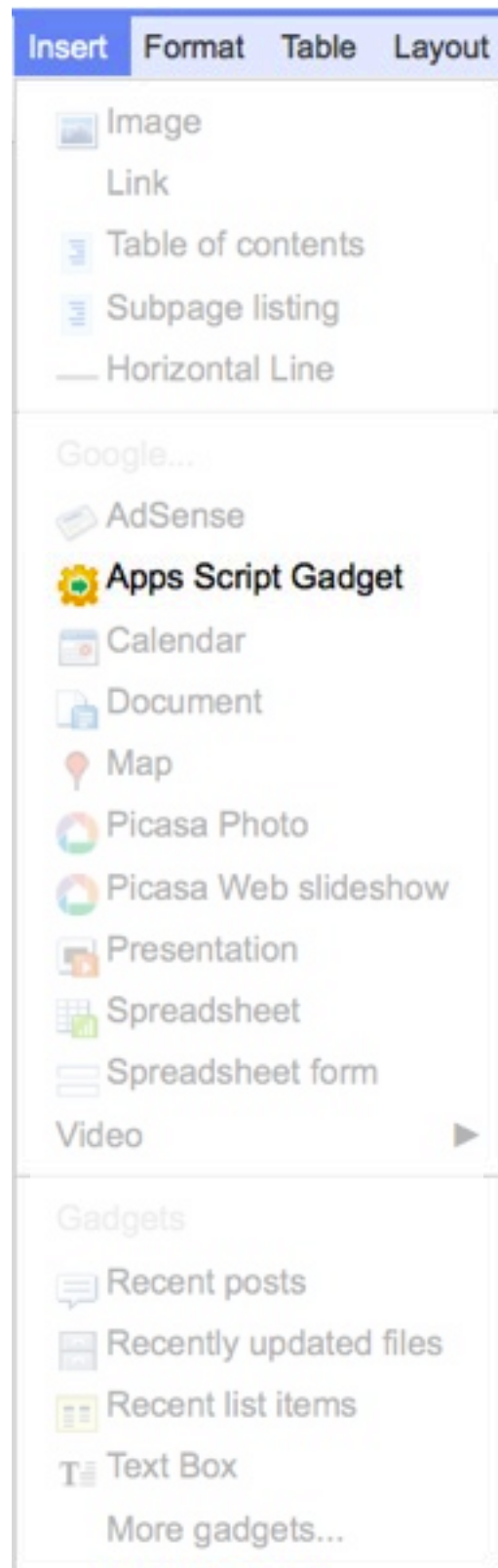


- Themes are applied to pages
- Pages are *mostly* standard html



# SitesApp

- Sharing is simple - at *Site Level*:
  - addViewer, getViewers
  - addCollaborator, getCollaborators
  - addOwner, getOwners



# Script In Sites

- Insert Gadgets in Edit mode
- Script Gadgets behave just like other Sites' Gadgets
  - Scripts that access user services need special authorization
- Manage Scripts in the Admin pages
- Write Scripts in the Apps Script Editor

# Managing Scripts

Google sites Manage example

[« Return to site](#)

## Site content

Recent site activity

Pages

Attachments

Page templates

## Apps Script

Deleted items

## Site settings

General

[Share this site](#)

Monetize

## Site appearance

Site layout

Colors and Fonts

Themes

Launch Editor

Name	Description
<a href="#">Data Input</a>	Enter and validate spreadsheet data through an embedded form
<a href="#">Publishing Workflow</a>	Manage Announcement review and publishing
<a href="#">Contact Owners</a>	Gadget to email a site owner

evin@example.com

Create page

Edit page

More actions

### Page actions

Revision History g then r

Subscribe to page changes f

Page settings u

Print page ⌘ + p

Preview page as viewer g then p

### Page templates

Save as page template

Change page template ↑ + t

### Site actions

Subscribe to site changes ↑ + f

Manage site g then m

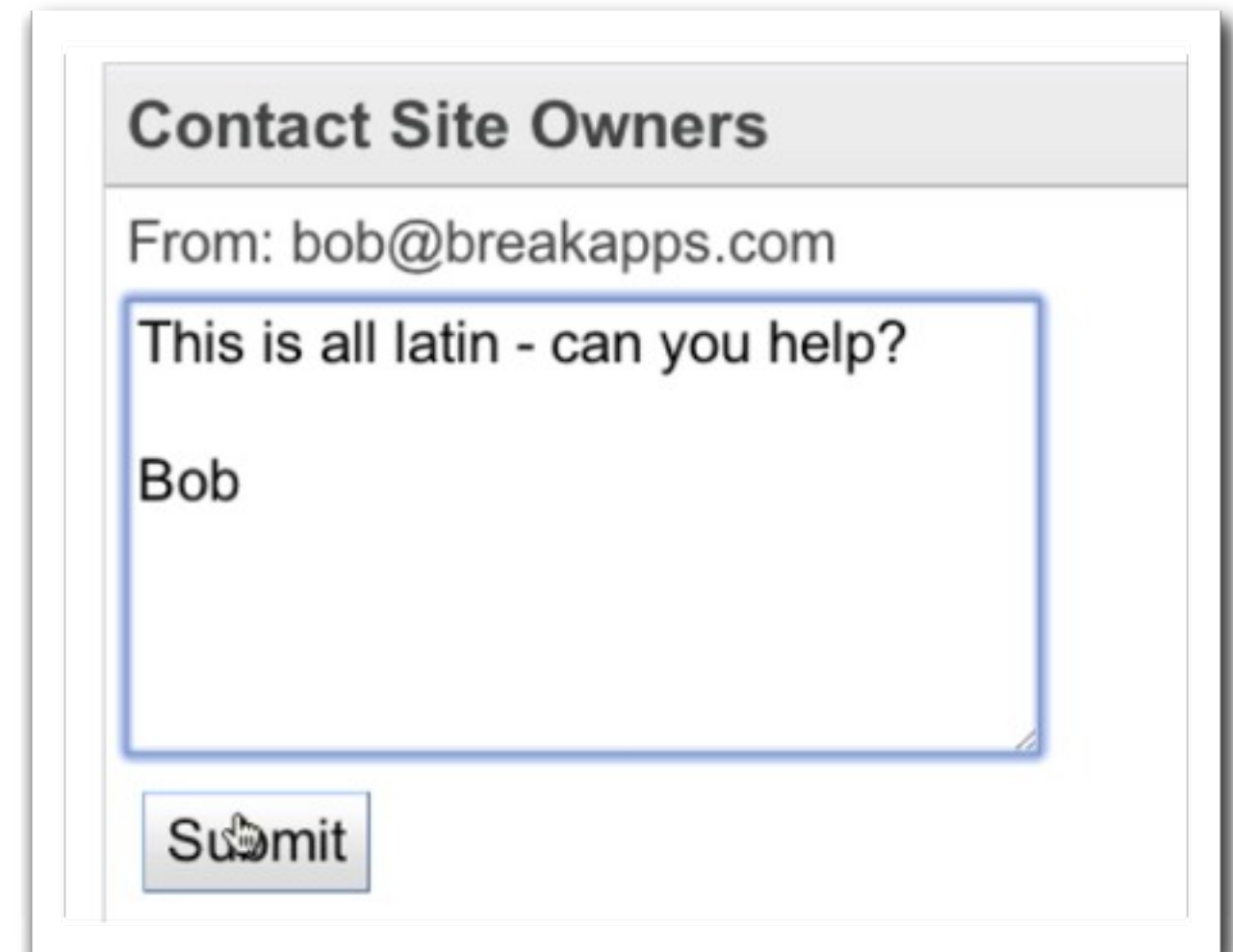
Site permissions ↑ + S

# Agenda

- Script in Sites
  - The basics
- **Simple Workflow**
  - Posting private comments to a site owner
- Document Approval
  - A publishing approval workflow
- Advanced Document Approval
  - Multiple approvers, reminders, escalation, etc.

# Contact Site Owner

- Script Gadgets allow dynamic UI embedded in Sites pages



Contact Site Owners

From: bob@breakapps.com

This is all latin - can you help?

Bob

Submit

```
function sendEmail_(e) {  
    var owners = SitesApp.getSite("domain", "site").getOwners();  
    var mail = MailApp.sendEmail(owners.join(","),  
        "hello", e.parameter.email);  
}
```

# Contact Owner Form

```
function doGet(e) {  
  
    var app = UiApp.createApplication().setTitle("Contact owners");  
    var mainPanel = app.createVerticalPanel().setId("mainPanel");  
    mainPanel.setSpacing(5);  
    app.add(mainPanel);  
  
    var user = Session.getActiveUser().getEmail();  
    mainPanel.add(app.createLabel("From: " + user));  
    mainPanel.add(app.createTextArea()  
        .setWidth("200px")  
        .setHeight("100px")  
        .setName("email"));  
  
    var submitButton = app.createButton('Submit');  
    var clickHandler = app.createServerClickHandler('sendEmail_');  
    clickHandler.addCallbackElement(mainPanel);  
    submitButton.addClickHandler(clickHandler);  
    mainPanel.add(submitButton);  
  
    return app;  
}
```

# Feedback

- Need to update the UI to indicate the email is sent...

```
function sendEmail_(e) {  
  var owners = SitesApp.getSite("breakapps.com", "example").getOwners();  
  var mail = MailApp.sendEmail(owners.join(","),  
    "hello", e.parameter.email);  
  
  var app = UiApp.getActiveApplication();  
  app.getElementById("mainPanel").setVisible(false);  
  
  var label = app.createLabel("Thanks - your feedback has been recorded!");  
  app.add(label);  
  
  return app;  
}
```

# Agenda

- Script in Sites
  - The basics
- Simple Workflow
  - Posting private comments to a site owner
- Document Approval
  - A publishing approval workflow
- Advanced Document Approval
  - Multiple approvers, reminders, escalation, etc.



# Document Approval

- Requestor submits a doc for publication
- Approver reviews and publishes
- Builds easily on top of *Contact Owners*

**Write an Article!**  
~~Contact Site Owners~~

From: bob@breakapps.com

This is all latin - can you help?

Bob

~~Submit~~ **Submit for Publication**

# Submission

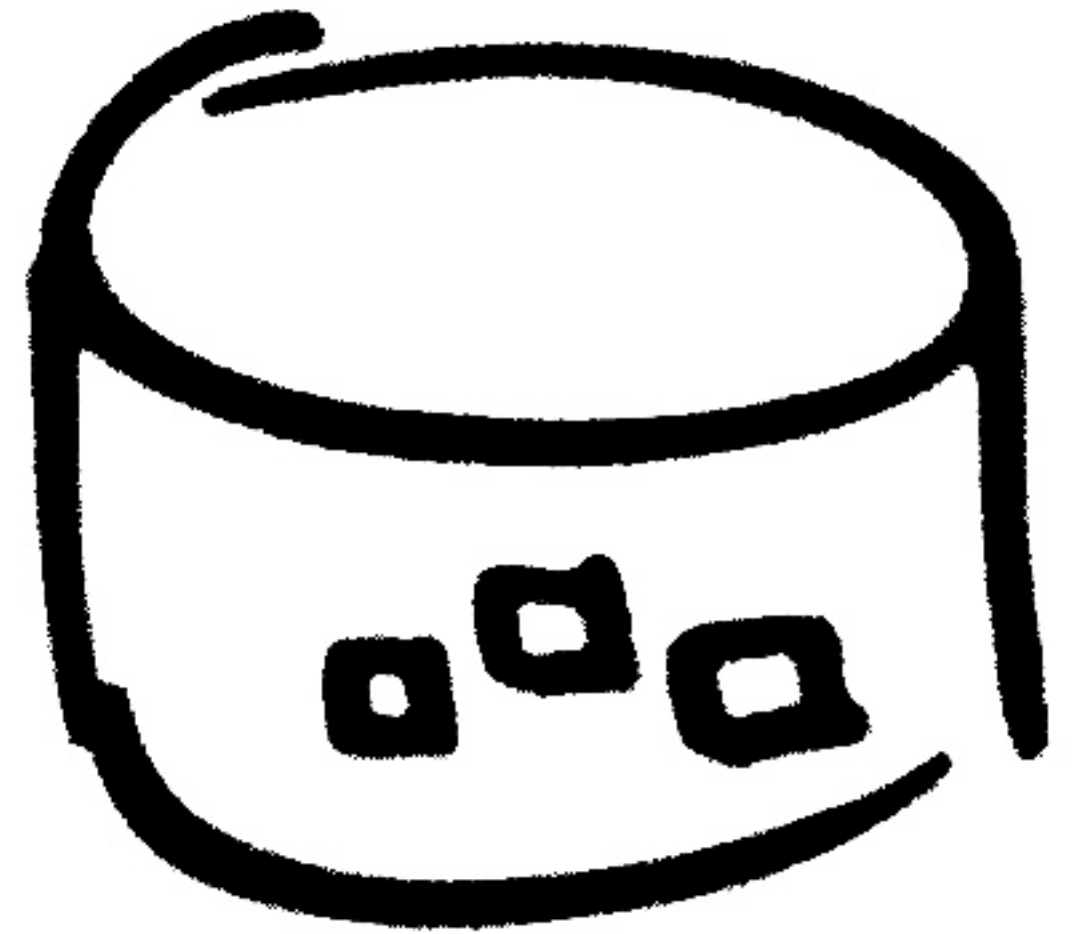
**Write an Article!**

From: [evin@breakapps.com](mailto:evin@breakapps.com)

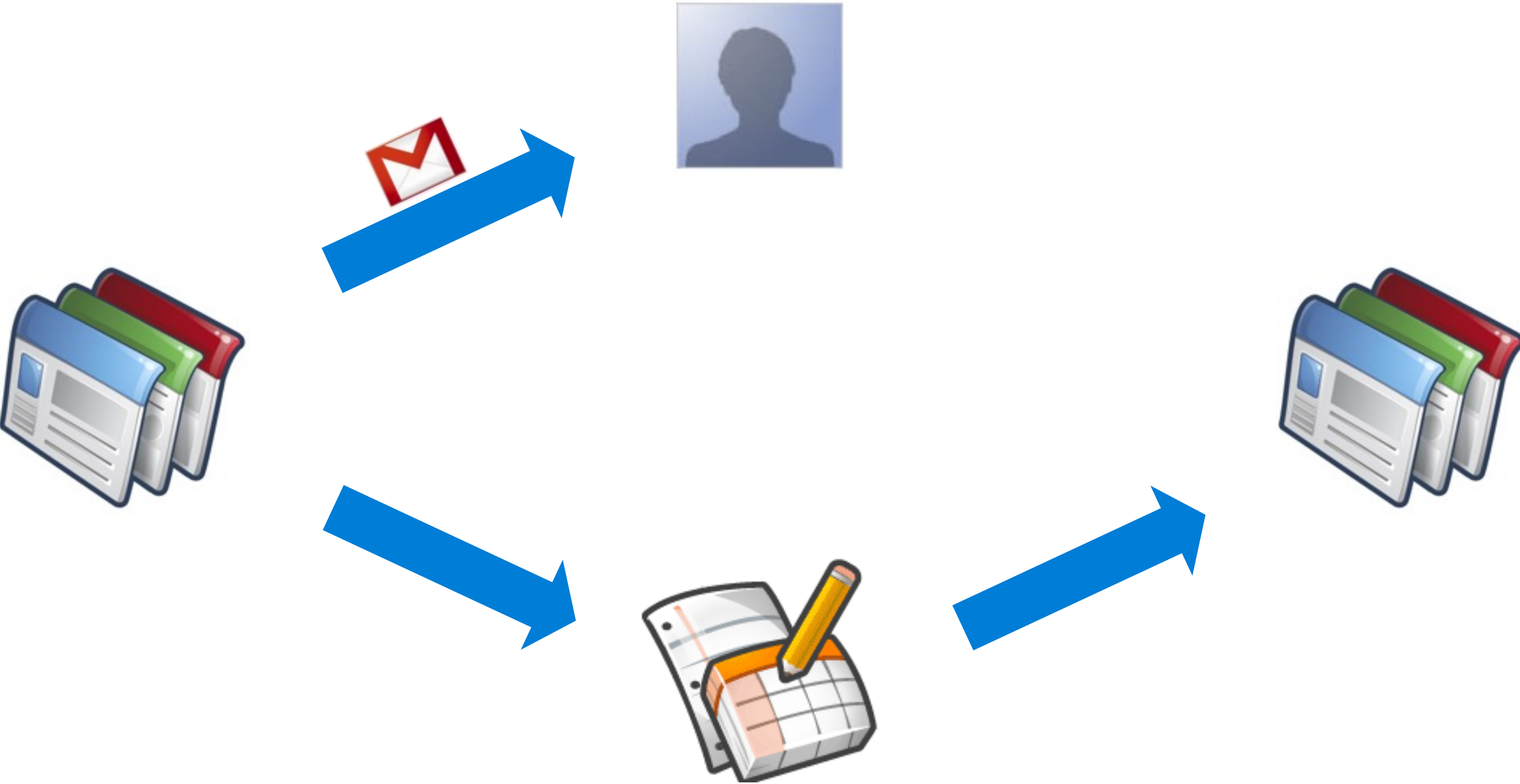
**Submit for Publication**

# Approval

- Document now lives in an email
- Emails get deleted, and overlooked, etc
- We really need to push to a data store
- For simplicity use a Spreadsheet...



# Approval Flow



# Spreadsheet Data Store

Google docs ☆ Approvals Private to only me Saved seconds ago Saved Share

File Edit View Insert Format Data Tools Help

Formula:  Show all formulas

	A	B	C	D	E	F	G
1	From	Date Submitted	Target Publication Date	Title	Approver	Status	Content
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							

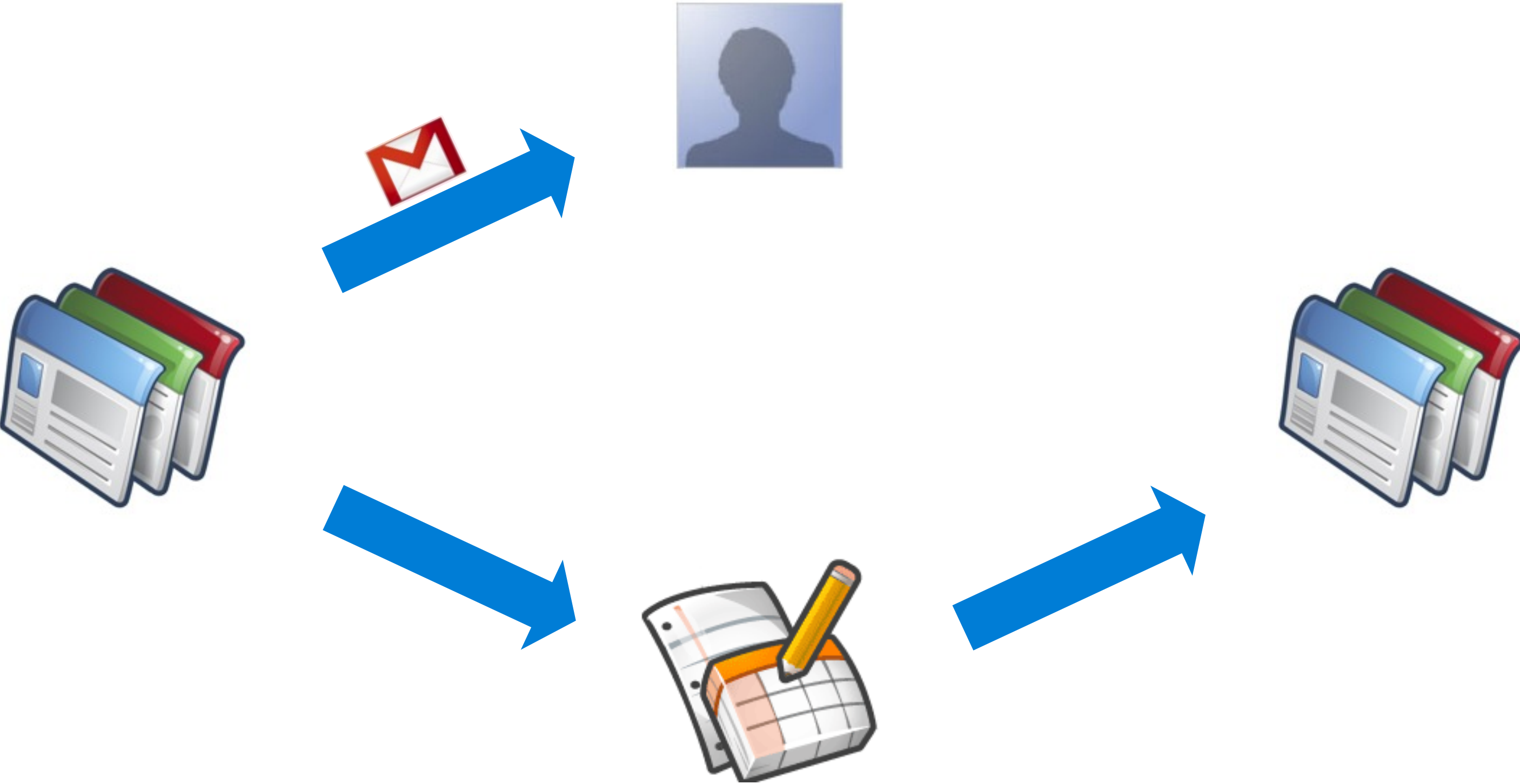
# Updating Data Store

```
function sendEmail_(e) {  
    var owners = SitesApp.getSite("breakapps.com", "example").getOwners();  
    var mail = MailApp.sendEmail(owners.join(","),  
        "hello", e.parameter.email);  
  
    // Also post to spreadsheet  
    var sheet = SpreadsheetApp.openById("tA9iTpVcWbTdzjmnGM1WkSQ").getActiveSheet();  
    var range = sheet.getRange(sheet.getLastRow()+1,1,1,7);  
    range.setValues([[Session.getActiveUser().getEmail(), Date(), "", "", "",  
        "Unapproved", e.parameter.email]]);  
}
```

# Publishing

```
function publishRow(r) {  
  // Read from the spreadsheet  
  var sheet = SpreadsheetApp.openById("tA9iTpVcWbTdzjmnGM1WkSQ")  
    .getActiveSheet();  
  var values = sheet.getRange(r, 1, 1, 7).getValues();  
  
  // Push to the Site  
  var site = SitesApp.getSite("breakapps.com", "example");  
  var p = site.createWebPage(values[0][3], "", values[0][6]);  
  
  // Inform author  
  MailApp.sendEmail(values[0][0], "Article published!",  
    "Your post is now live here " + p.getUrl());  
  
  // Update state to Published  
  sheet.getRange(r, 6).setValue("Published");  
}
```

# Approval Flow





# Approving

- Document now lives in Data Store
- Approver has details in an Email
- Use Spreadsheet as *Control Panel* for Approvals?

# Approval from Spreadsheet

```
function publish() {  
    var r = SpreadsheetApp.getActiveRange().getRow();  
    publishRow(r)  
}
```

```
function onOpen() {  
    var subMenus = [{name: "Approve", functionName: "publish"}];  
    SpreadsheetApp.getActiveSpreadsheet().addMenu("Publish", subMenus);  
}
```

# Approving from Email

- Far more efficient for the approver
- Send message with an embedded *html form*
- Process becomes a one-click approval

# HTML Forms

User name:

Password:

```
<form action="http://www..." method="post">
```

User name:

```
<input type="text" name="username" size="20" />
```

```
<br />
```

Password:

```
<input type="password" name="password" size="20" />
```

```
<br />
```

```
<input type="submit" value="Submit" />
```

```
</form>
```

# HTML Form Actions

```
<form action="http://www..." method="post">
```

User name:

```
<input type="text" name="username" size="20" />
```

```
<br />
```

Password:

```
<input type="password" name="password" size="20" />
```

```
<br />
```

```
<input type="submit" value="Submit" />
```

```
</form>
```

# Server?

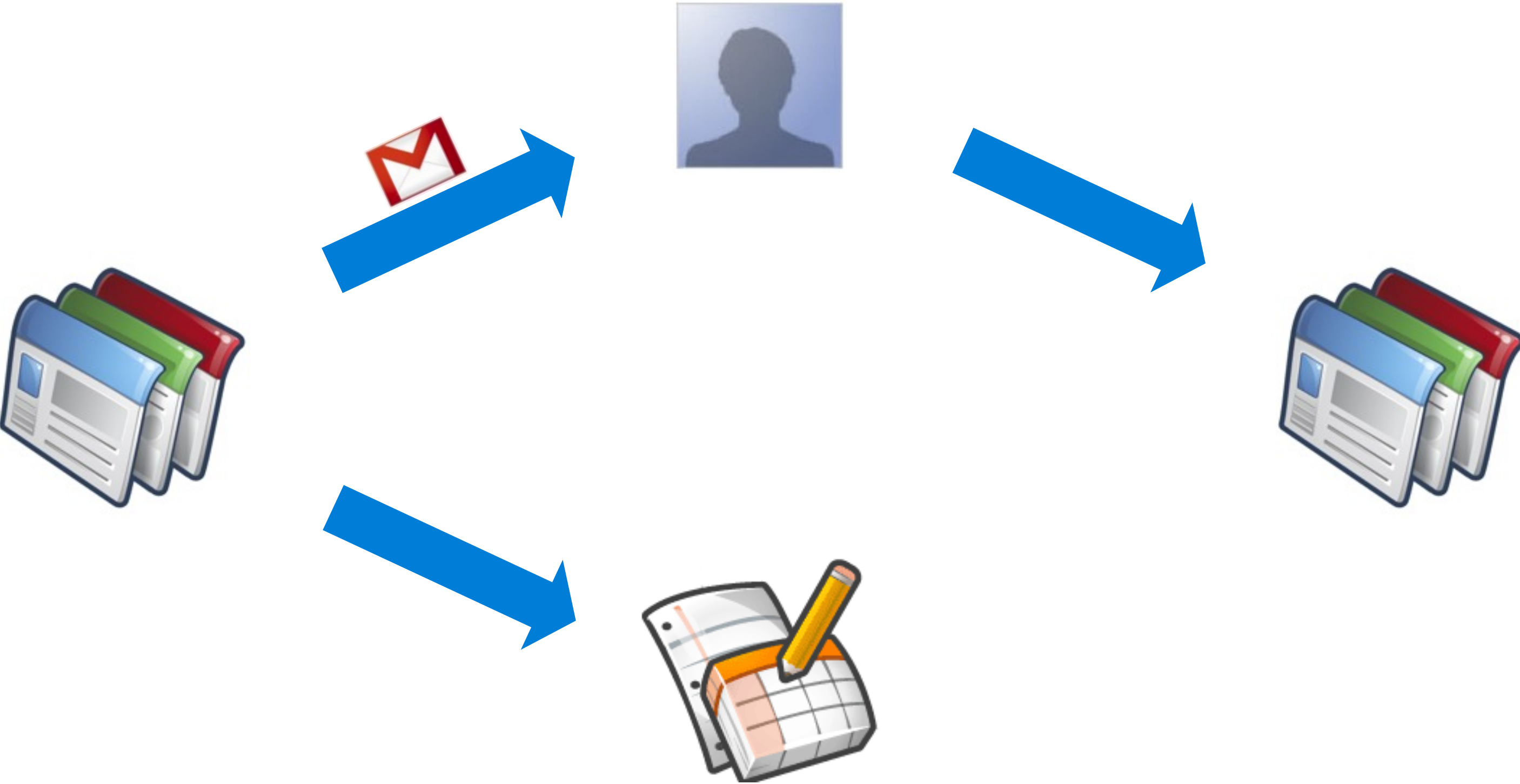


```
function doPost(e) {  
    var app = UiApp.createApplication();  
    var panel = app.add(app.createVerticalPanel());  
  
    for (var i in e.parameter)  
        panel.add(app.createLabel(i + ": " + e.parameter[i]));  
  
    return app;  
}
```

# Approving from Email

- Embed the article, and an approval button into the email
- Embed a hidden field that points to the correct article
- Ensure our 'doPost' checks the user, and Publishes

# Approval Flow





# Request

```
function sendEmail_(e) {  
  
    ...  
    // Now send email form  
    var body = e.parameter.email + '<br><br>' +  
        '<form action="http://..." method = "post">' +  
        '<input type="hidden" name="row" value="' + row + '" />' +  
        '<input type="submit" value="Approve" />' +  
        '</form>';  
  
    MailApp.sendEmail(owners.join(","),  
        "Draft Post" , , {htmlBody: body});  
  
    ...  
}
```

# Approval

```
function doPost(e) {  
  
    // call the old publish function  
    publishRow(e.parameter.row);  
  
    // let the user know it hapened  
    var app = UiApp.createApplication();  
    app.add(app.createLabel("Published!"));  
  
    return app;  
}
```

# Security

- How do we know who clicked the Approve button?
- Service is restricted to our 'domain' or 'organization'
- We need a user-level check

# Security

```
function doPost(e) {  
  
    var message = "Access Denied!";  
    if (Session.getActiveUser() == "evin@breakapps.com") {  
        publishRow(e.parameter.row);  
        message = "Published";  
    }  
  
    // let the user know what hapened  
    var app = UiApp.createApplication();  
    app.add(app.createLabel(message));  
  
    return app;  
}
```

# More Security

- *Effective User* is the account running the script
- *Active User* is at the keyboard
- Script can only get the *Active User* in special cases
  - When both users are part of the same *domain* or *organization*
  - *Domain* is **not** a special edition (partner, or edu, etc)

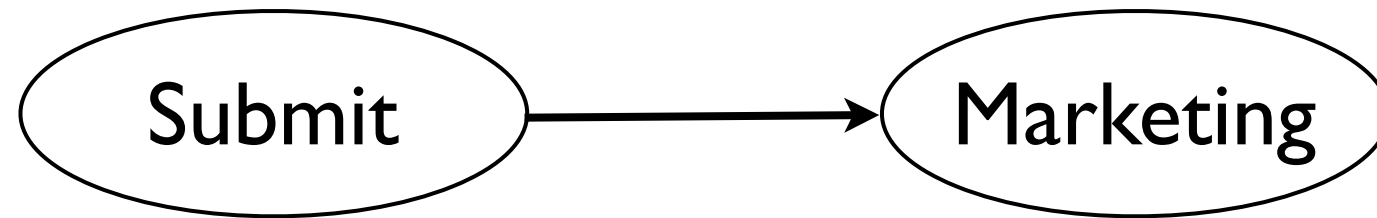
# Agenda

- Script in Sites
  - The basics
- Simple Workflow
  - Posting private comments to a site owner
- Document Approval
  - A publishing approval workflow
- **Advanced Document Approval**
  - Multiple approvers, reminders, escalation, etc.

# Multiple Approvers

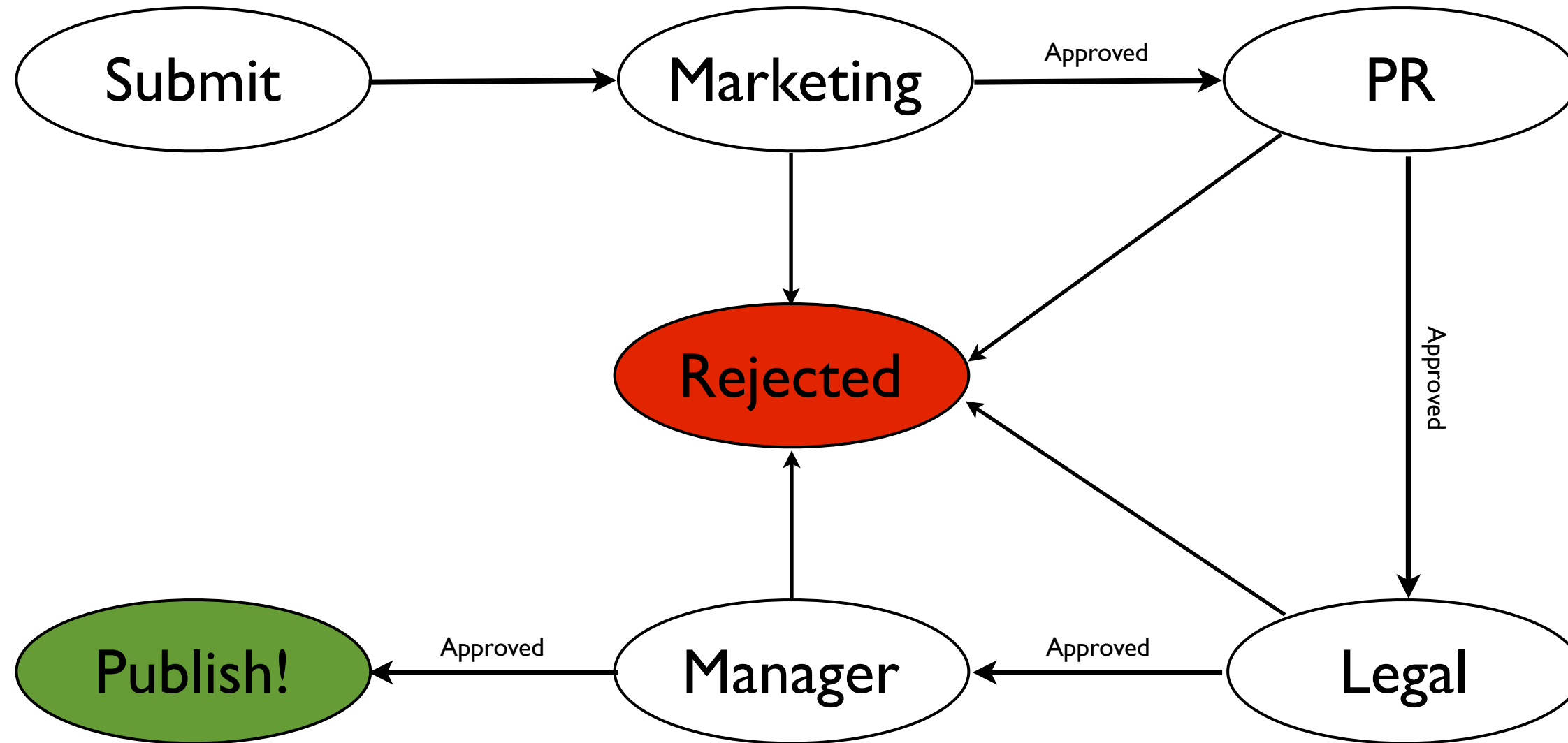
- Real value of workflow is to codify a complex process
- Often need Marketing, Legal, PR and Managerial approvals
- How do we handle the explosion of complexity?
- Introduce a better representation of the **state**

# State

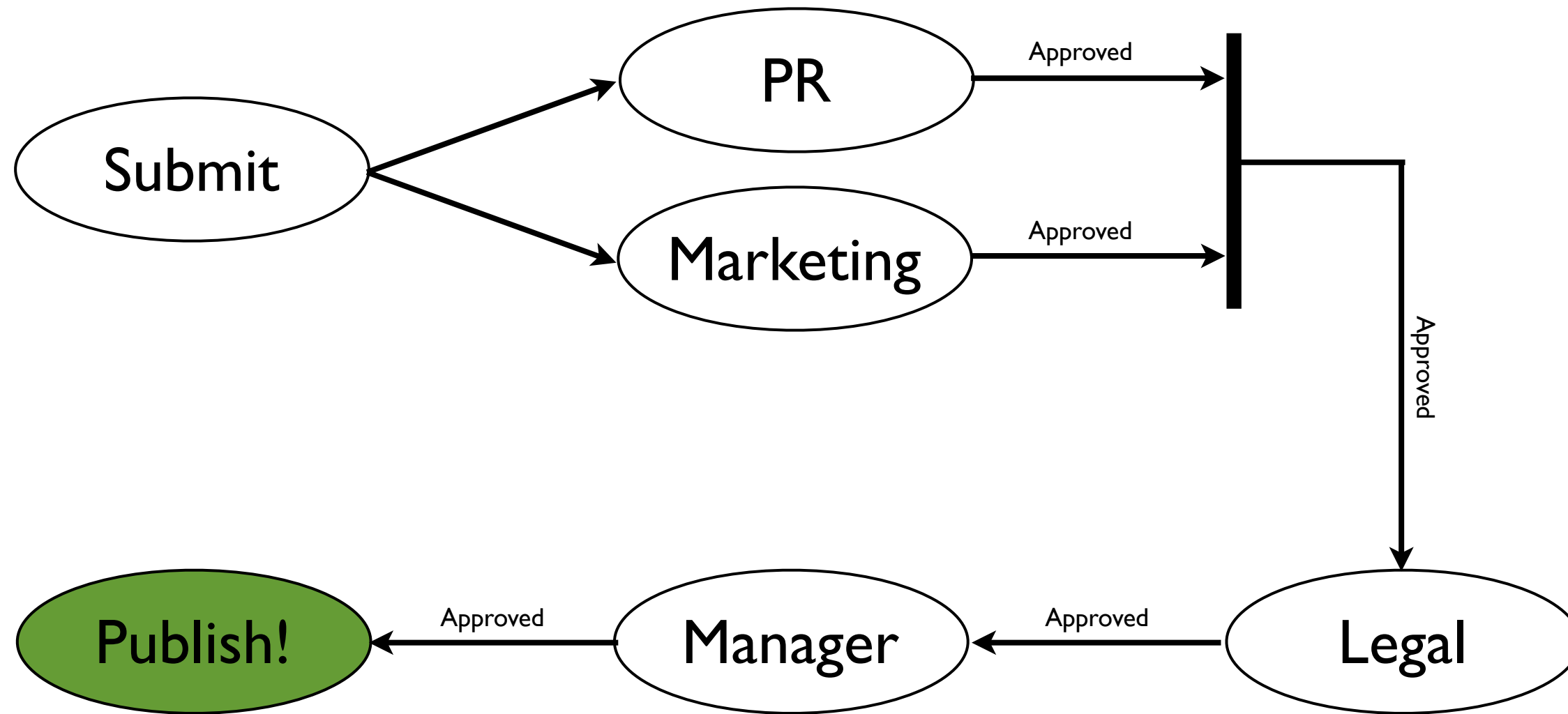




# State



# More Realistically



# Representing State

- Process is a sequence of Steps
- At least one approver per Step, perhaps multiple
- Approval state maintained in the Step
- Use JavaScript Objects to represent Steps
- Process is simply an Array of Steps

# Step Creation

```
function createStep(approvers) {  
  
    var step = new Object();  
  
    step['numberOfApprovers'] = approvers.length;  
    step['approvers'] = approvers.slice(0);  
    step['status'] = 'pending';  
  
    for (var i in approvers)  
        step[approvers[i]] = 'pending';  
  
    return step;  
}
```

```
createStep([ "evin@breakapps.com",  
            "bob@breakapps.com" ] );  
  
{ status: "pending",  
  numberOfApprovers: 2,  
  approvers: [ "evin@breakapps.com",  
              "bob@breakapps.com" ],  
  evin@breakapps.com: "pending",  
  bob@breakapps.com: "pending"  
}
```

# Process Creation

```
function createProcess() {  
  var process = [];  
  
  // step 1  
  process.push(createStep([ 'marketing@breakapps.com',  
                           'pr@breakapps.com' ]));  
  
  // step 2  
  process.push(createStep([ 'legal@breakapps.com' ]));  
  
  // step 3  
  process.push(createStep([ 'manager@breakapps.com' ]));  
  
  return process;  
}
```



# Saving State

```
function saveProcess(process, sheet, row) {  
    var status = sheet.getRange(row, 6);  
    status.setValue(JSON.stringify(process));  
}
```

```
function loadProcess(sheet, row) {  
    var status = sheet.getRange(row, 6);  
    return eval(status);  
}
```

# Algorithm

## On Submit:

- create appropriate approval Process
- write to Data Store
- email approvers in first Step

## On Approval:

- get the active Step, update status
- if Step fully approved, start next Step
- if all Steps approved, publish!

# Dynamic Approvals

- Our Process structure is very flexible
- Steps can be added and removed trivially
  - Simple Array manipulation
- Automatically add Legal Step *if* post contains profanity
- Enable approvers to add steps - 'Refer to Legal', etc.



# Reminders

- Emails often get overlooked or ignored
- Delays are annoying and potentially quite costly
- Issue a daily reminder of pending approvals...

# Who Needs A Reminder?

```
function getPendingApprovers(process) {  
    var activeStep = getActiveStep(process);  
    var pendingApprovers = [];  
  
    // find each pending user  
    for (var u in activeStep.approvers) {  
        var user = activeStep.approvers[u];  
        if (activeStep[user] == 'pending')  
            pendingApprovers.push(user);  
    }  
  
    return pendingApprovers;  
}
```

```
function getActiveStep(process) {  
    for (var s in process)  
        if (process[s].status == 'pending')  
            return process[s];  
}
```

# Sending Reminders

```
function sendReminders() {
  var sheet = SpreadsheetApp
    .openById( "tA9iTpVcWbTdzjmnGM1WkSQ" )
    .getActiveSheet();

  var data = sheet.getDataRange().getValues();

  for (var row=1; row<data.length; row++) {
    var process = data[row][6];
    var users = getPendingApprovers(process);
    sendReminder(users, data[row][7]);
  }
}
```

# Reminders

- This is a basic 'reminder' strategy
- Easy to extend:
  - Store last date sent in the Step
  - Check on length of delay, before sending reminder
- Solid foundation for Escalations...

# Roles & User Attributes

- Often need to know a user's manager, etc.
- In general, need to find user responsible for Approval
- Important to get access to org. hierarchy

# Google Profiles


← → ↻ <https://www.google.com/accounts/b/0/ManageAccount> ☆ 🔍

[Web](#) [Images](#) [Videos](#) [Maps](#) [News](#) [Shopping](#) [Mail](#) [more](#) ▼ [evin@breakapps.com](#) ▼ | [My Account](#) | [Help](#) | [Sign out](#)

## Google accounts

---












### Profile



### Personal Settings

<b>Security</b>	<a href="#">Changing your password</a> <a href="#">Authorizing applications &amp; sites</a>
<b>Dashboard</b>	<a href="#">View data stored with this account</a>
<b>Email addresses</b>	evin@breakapps.com (Primary email)
<b>Multiple sign-in</b>	On - <a href="#">Edit</a>
<b>Connected accounts</b>	<a href="#">View and manage your accounts from other services</a>

### My products

 <a href="#">App Engine</a> - <a href="#">My Applications</a>	 <a href="#">Calendar</a> - <a href="#">Settings</a>	 <a href="#">Custom Search</a> - <a href="#">My Search Engines</a>
 <a href="#">Developer Consoles</a>	 <a href="#">Docs</a> - <a href="#">Settings</a>	 <a href="#">Gmail</a> - <a href="#">Settings</a>
 <a href="#">Groups</a> - <a href="#">Manage subscriptions</a>	 <a href="#">Sites</a>	 <a href="#">Subscribed Links</a> - <a href="#">My Subscribed Links</a>
 <a href="#">Talk</a>	 <a href="#">Webmaster Tools</a>	

©2011 Google - [Google Home](#) - [Terms of Service](#) - [Privacy Policy](#) - [Help](#)

# Google Apps Directory Sync

**LDAP User Profile Attributes**

Specify the LDAP attributes that contain profile information for users. Directory Sync will import the information in these LDAP attributes into User Profiles in Google Apps. [Learn More](#)

Primary email:	<input type="text" value="mail"/>	Work phone numbers:	<input type="text" value="telephoneNumber"/>
Job title:	<input type="text" value="title"/>	Home phone numbers:	<input type="text"/>
Company name:	<input type="text" value="company"/>	Fax numbers:	<input type="text"/>
Assistant's DN:	<input type="text" value="assistant"/>	Mobile phone numbers:	<input type="text" value="cellphoneNumber"/>
Manager's DN:	<input type="text" value="manager"/>	Work mobile phone numbers:	<input type="text"/>
Department:	<input type="text" value="department"/>	Assistant's Number:	<input type="text"/>
Office location:	<input type="text" value="division"/>	<b>Primary Work Address Attributes</b>	
Employee ids:	<input type="text" value="employeeID"/>	Street Address:	<input type="text"/>
Websites:	<input type="text"/>	P.O. Box:	<input type="text"/>
Notes:	<input type="text"/>	City:	<input type="text"/>
		State/Province:	<input type="text"/>
		Zip/Postal Code:	<input type="text"/>
		Country/Region:	<input type="text"/>

[← Previous](#) [Next →](#)

# Google Apps Directory & Profiles

- Key information now available in the cloud
- We can use it to look up managers
  - [http://code.google.com/apis/apps/profiles/developers\\_guide\\_protocol.html](http://code.google.com/apis/apps/profiles/developers_guide_protocol.html)
- How do we call that API from Apps Script?



# Google Apps Directory & Profiles

```
function getProfile(user) {  
  
    // Setup OAuthServiceConfig  
    var oAuthConfig = UrlFetchApp.addOAuthService("Directory");  
    oAuthConfig.setAccessTokenUrl("https://www.google.com/accounts/OAuthGetAccessToken");  
    oAuthConfig.setRequestTokenUrl(  
        "https://www.google.com/accounts/OAuthGetRequestToken?scope=https://www.google.com/m8/feeds/");  
    oAuthConfig.setAuthorizationUrl("https://www.google.com/accounts/OAuthAuthorizeToken");  
    oAuthConfig.setConsumerKey("anonymous");  
    oAuthConfig.setConsumerSecret("anonymous");  
  
    // Set Options  
    var options =  
        {  
            "oAuthServiceName" : "Directory",  
            "oAuthUseToken" : "always",  
            "headers": { "GData-Version": "1.0"}  
        };  
  
    // Make call  
    var result = UrlFetchApp.fetch(  
        "https://www.google.com/m8/feeds/profiles/domain/breakapps.com/full/"  
        + user, options);  
  
    return result.getContentText();  
}
```

# Google Apps Directory & Profiles

```
// Set Options
var options =
{
  "oAuthServiceName" : "Directory",
  "oAuthUseToken" : "always",
  "headers" : { "GData-Version" : "1.0" }
};

// Make call
var result = UrlFetchApp.fetch(
  "https://www.google.com/m8/feeds/profiles/domain/breakapps.com/full/"
  + user, options);

return result.getContentText();
```

# My Profile

```
▼<feed>
  ▶<id>...</id>
  <updated>2011-05-02T18:38:32.393Z</updated>
  <link rel="http://schemas.google.com/g/2005#feed" type="application/atom+xml" href="https://www.google.com/m8/feeds/gal/google.com/full"/>
  <link rel="self" type="application/atom+xml" href="https://www.google.com/m8/feeds/gal/google.com/full?q=evin"/>
  <generator version="1.0" uri="http://www.google.com/m8/feeds">Contacts</generator>
  <openSearch:startIndex>1</openSearch:startIndex>
  ▼<entry gd:etag="W/"YDwpeyI." ">
    ▶<id>...</id>
    <published>1970-01-01T00:00:00.000Z</published>
    <updated>1970-01-01T00:00:00.000Z</updated>
    <title>Evin Levey</title>
    ▼<content>
      Product Manager, Google NYC<div> - Docs Platform</div> - Google Script</div><div><br></div>
    </content>
    <link rel="self" type="application/atom+xml" href="https://www.google.com/m8/feeds/gal/google.com/full/P.5a216486045b8c6c"/>
    <gal:type type="profile"/>
    ▼<gd:name>
      <gd:fullName>Evin Levey</gd:fullName>
      <gd:givenName>Evin</gd:givenName>
      <gd:familyName>Levey</gd:familyName>
    </gd:name>
    <gContact:nickname>Evin</gContact:nickname>
    <gContact:gender value="male"/>
    <gd:where valueString="US-NYC-9TH 10C354A"/>
    ▶<gd:organization rel="http://schemas.google.com/g/2005#other">...</gd:organization>
    ▼<gd:organization rel="http://schemas.google.com/g/2005#other">
      <gd:orgName>Google</gd:orgName>
      <gd:orgTitle>Product Manager</gd:orgTitle>
    </gd:organization>
    <gd:phoneNumber rel="http://schemas.google.com/g/2005#home">+1 (555) 425 2341</gd:phoneNumber>
    <gd:phoneNumber rel="http://schemas.google.com/g/2005#work">+1 (555) 565 2550</gd:phoneNumber>
    ▼<gd:structuredPostalAddress rel="http://schemas.google.com/g/2005#work">
      <gd:formattedAddress>76 9th Ave, New York, NY</gd:formattedAddress>
    </gd:structuredPostalAddress>
    <gContact:relation rel="manager">Jonathan Rochelle</gContact:relation>
  </entry>
</feed>
```

# Roles & User Attributes

- Single source of truth for org. structure
- Available now via Profiles API (Admin only)
- Available via new Directory API very soon

# Recap

- **Script in Sites**
  - Actions on a Site, and embedding Script into a Site's page
- **Simple Workflow**
  - Private comments to a site owner
- **Document Approval**
  - A simple publishing approval workflow

# Recap

- Approval Mechanisms
  - From a Spreadsheet
  - From email using HTML Forms
- Multi-Step Approvals
  - Saving the State of the approval
  - Dynamically altering the process
- Reminders & Escalation
  - Integration with Google Apps Directory

# Questions



Hashtags: **#io2011** **#GoogleApps**

Feedback: **<http://goo.gl/hwvJk>**

Google™

