PharmaSUG 2019 – Paper PO-225

Badge in Batch with Honeybadger: Generating Conference Badges with Quick Response (QR) Codes Containing Virtual Contact Cards (vCards) for Automatic Smart Phone Contact List Upload

Troy Martin Hughes

ABSTRACT

Quick Response (QR) codes are widely used to encode information such as uniform record locators (URLs) for websites, flight passenger data on airline tickets, attendee information on concert tickets, or product information on product packaging. The proliferation of QR codes is due in part to the broad dissemination of smart phones and the accessibility of free smart phone applications that scan QR codes. With the ease of QR code scanning has come another common QR code usage—the identification of conference attendees. Conference badges, emblazoned with attendee-specific QR codes, can communicate attendee contact information to other conference goers, including organizers, vendors, potential customers or employers, and others. Conference badges that contain QR codes make it easy for attendees to link up with each other because snapping a photo of a badge can immediately capture contact information (that could not otherwise be printed on the badge itself). To that end, this text introduces flexible Base SAS® software that dynamically creates attendee QR codes from a data set containing contact and other information. This data-driven approach could be used to create attendee badges by conference organizers rather than costly third-party vendors. When a badge QR code is scanned by a conference goer, the attendee's personal information—including name, job title, company, phone number, email address, city, state, website, and biographical statement—is ported into a variant call format (VCF) file (or vCard) that can be uploaded automatically into a smart phone's contact list. Attendees are able to select what personal information is contained within their QR code and conference organizers are able to customize and configure badge format and content through an external cascading style sheets (CSS) file that dynamically alters badges without the necessity to modify the underlying code. This end-to-end system offers conference organizers potential cost savings of thousands of dollars—money that can be diverted from costly, third-party badge vendors to open bars and other necessities.

INTRODUCTION

Quick response (QR) codes, launched in 1994, are barcodes that contain information encoded in boxy black and white patterns. Some QR codes are unique. For example, if you've purchased an airline ticket, concert ticket, or sporting event ticket in the past decade, the ticket probably had a QR code (unique to you) that was scanned as you entered the event. The QR code in part demonstrates that the ticket was valid and unique (so that would-be thieves cannot attempt to enter the same venue with the same ticket). In other cases, a QR code may not be unique, such as a code used on product packaging, in which the code may be printed on thousands of boxes. For example, in a recent trip to Best Buy (aka, the Amazon showroom), I scanned QR codes on a number of printers, which linked me to online product information.

In addition to other uses, QR codes have become ubiquitous at trade, technical, and other professional conferences. With individualized QR codes emblazoned on attendee badges, conference organizers can ensure that attendees belong in designated events. Moreover, other conference goers can snap photos of attendee badges or scan them with QR-code readers, and immediately upload the associated contact information into a cellular phone contact list. The use of QR codes on conference badges can tremendously increase the ability of conference goers to interact with each other and to exchange and receive contact information.

This text introduces a data-driven method that generates variant call format (VCF) files (or vCards) that can be uploaded automatically to contact lists of smart phones. Contact information is collected when attendees register for a conference or other event, and eventually makes its way to a database or table. The SAS® Honeybadger takes those data, converts them to a QR code that can be placed dynamically on individuals'

badges, and creates those badges as a file that can be saved and printed. This allows conference organizers to print badges themselves rather than paying a third-party vendor. When conference attendees want to exchange contact information, they only have to snap a picture of a badge (or use a QR code reader application) to have the vCard uploaded in their phones.

Special thanks to Rob Allison for his blog post that wholly inspired this endeavor![i] Rob's code is posted here, dated September 12, 2016, and formed the basis of this exploration.[ii] The Honeybadger relies on the Google Infographics QR code generator that is still functional, but which has been deprecated.[iii]

ATTENDEE PERSONAL DATA

Personal data for conference attendees is commonly collected during the online conference registration process. Although some of this information is confidential (e.g., credit card information), other information is considered public (e.g., first and last name, company) and is commonly printed on conference badges. And in the middle may lie personal information (e.g., phone number, email, city of residence, social media username) that some attendees don't mind releasing although others will want held confidential. In other words, some attendees would prefer to be recognized by name only whereas others would prefer to walk around with a business card emblazoned on their chest. Thus, the first step is to ensure that attendees can "opt in" to all personal data that will be released—either displayed in plain text on a badge (like company name) or encoded within the QR code.

Behind the scenes, the data are collected and amassed into a database or table. This table, converted into a SAS data set, can be used to drive the dynamic creation of QR codes and subsequent badges that can be printed for all conference attendees. Note that conference organizers are responsible for converting their specific attendee table into the format of the Attendees data set, depicted in Table 1.

Information	Variable Name	Format
Prefix	prefix	$5
First Name	firstName	$30
Middle Name	middleName	$30
Last Name	lastName	$30
Suffix	suffix	$5
City	city	$30
State	state	$2
Cell Phone	cellPhone	$12
Work Phone	workPhone	$12
Email Address	email	$40
Website URL	URL	$100
Biography	bio	$100
Company Name	company	$30
Professional Title	jobTitle	$40

Table 1. Attendees Data Set Collected from Theoretical Conference Registration Table

Within the Biography field, carriage returns are changed to "/n" so that line breaks and multiple lines are supported within the vCard NOTE field. Blank lines are also supported, so long as they occur between text blocks (i.e., paragraphs), and are represented by two successive "/n" escape codes.

SAMPLE ATTENDEES DATA SET

The following DATA step creates a sample data set (HONEYBGR.Attendees) that includes some loveable characters from NBC's hit comedy, *The Office*:

```
%let loc=D:\SAS\honeybadger\;  * USER MUST CHANGE THIS VALUE!!! *;
libname honeybgr "&loc";

data honeybgr.attendees;
   infile datalines dsd;
   length prefix $5 firstName $30 middleName $30 lastName $30
      suffix $5 city $30 state $2 cellPhone $10 workPhone $10
      email $40 URL $100 bio $100 company $30 jobTitle $40;
   input prefix $ firstName $ middleName $ lastName $
      suffix $ city $ state $ cellPhone $ workPhone $
      email $ URL $ bio $0 company $ jobTitle $;
   label prefix='Prefix' firstName='First Name' middleName='Middle Name'
      lastName='Last Name' suffix='Suffix' city='City' state='State'
      cellPhone='Cell Phone' workPhone='Work Phone'
      email='Email' URL='URL' bio='Biography' company='Company' jobTitle='Job
Title';
datalines;
Mr.,Michael,Gary,Scott,,Scranton,PA,,570-344-
1212,mscott@dundermiff.com,https://theoffice.fandom.com/wiki/Michael_Scott,Michael
Scott is the Regional Manager of Dunder Mifflin.,Dunder Mifflin,Regional Manager
Mr.,Jim,,Halpert,,Scranton,PA,,570-344-
1214,jhalpert@dundermiff.com,https://theoffice.fandom.com/wiki/Jim_Halpert,Jim
Halpert has been a paper salesman for far too long.,Dunder Mifflin,Salesman
Mr.,Dwight,K,Schrute,,Scranton,PA,,570-344-
1269,dschrute@dundermiff.com,https://theoffice.fandom.com/wiki/Dwight_Schrute,"Dwig
ht is a fan of Battlestar Galactica, bears, and beets.",Dunder Mifflin,Assistant to
the Regional Manager
Ms.,Pam,,Halpert,,Scranton,PA,,570-344-
1279,phalpert@dundermiff.com,https://theoffice.fandom.com/wiki/Pam_Beesly,"Pam
Halpert has held various roles, including receptionist, salesman, and office
manager.",Dunder Mifflin,Office Manager
Ms.,Angela,,Martin,,Scranton,PA,,570-344-
1206,amartin@dundermiff.com,https://theoffice.fandom.com/wiki/Angela_Martin,Angela
Martin is the head of the Accounting Department and loves her kitties.,Dunder
Mifflin,Accountant
Ms.,Meredith,,Palmer,,Scranton,PA,,570-344-
1208,mpalmer@dundermiff.com,https://theoffice.fandom.com/wiki/Meredith_Palmer,Mered
ith Palmer sleeps with suppliers for steak coupons and discounted products.,Dunder
Mifflin,Supplier Relations
;
```

Note also that the &LOC global macro variable must be changed to reflect the user's local SAS infrastructure. In reality, conference data would have been maintained in a database, table, or SAS data set, from which those data could be ported to the HONEYBGR. Users will also need to ensure that only those data approved for release (by attendees) are included in the Attendees data set. Thus, this represents the finalized, culled data set in which personal information that attendees wish to remain private (like personal cell phone numbers) have been removed.

CREATING A VIRTUAL CONTACT CARD (VCARD)

The vCard format specification is defined as an ANSI standard to ensure compatibility across devices.[iv] All possible vCard fields are not incorporated into the QR codes that are utilized; thus, only those fields listed in Table 1 are utilized by the Honeybadger. The metadata content contained within the QR codes represent the typical contact information (e.g., name, phone number, email, job title, company) that would be printed on a business card, as well as additional (optional) information (e.g., short biography) that might be

exchanged in introductory conversation. And, for those conference goers who value privacy, their QR codes can be generated to include only first and last name, with no additional information.

When a conference attendee scans a QR code on a badge, the smart phone recognizes the vCard format, and prompts the user to create (or update) a contact (based on the specific phone make and model). The contact information is automatically saved to the phone and, if the badge wearer has provided optional biographical information, that information can be reviewed later. vCard files can also contain one URL, so attendees can choose to highlight their LinkedIn, Facebook, Twitter, or other social media handle, or to link to a blog or other location. In this way, although many conference goers may still choose to carry and distribute business cards at conferences, they can also rely on their badge to dynamically (and in a much "greener" fashion) convey that same information.

The following DATA step iteratively creates a vCard file for each person in the Attendees data set:

```
data _null_;
   set honeybgr.attendees end=eof;
   length fname $200;
   array line $200 line1-line13;
   fname="&loc" || 'vCard' || strip(put(_n_,8.)) || '.vcf';
   file f filevar=fname;
   line1='BEGIN:VCARD';
   line2='VERSION:4.0';
   line3=cats('N:',lastName,';',firstName,';;',prefix,';');
   line4=catx(' ','FN:',firstName,lastName);
   line5=cats('ORG:',company);
   line6=cats('TITLE:',jobTitle);
   line7=cats('TEL;TYPE=work:tel:+',workPhone);
   line8=cats('TEL;TYPE=cell:tel:+',cellPhone);
   line9=cats('ADR;TYPE=work:',city,';',state);
   line10=cats('EMAIL;TYPE=work:',email);
   line11=cats('URL:',url);
   line12=cats('NOTE:',bio);
   line13='END:VCARD';
   do over line;
      put line;
      end;
   if eof then call symputx('numpeeps',strip(put(_N_,8.)),'g');
run;
```

Thus, given the sample data set, the DATA step creates six vCard files (vCard1.vcf through vCard6.vcf). For example, the first observation (Michael Scott) is saved as vCard1.vcf:

```
BEGIN:VCARD
VERSION:4.0
N:Scott;Michael;;Mr.;
FN: Michael Scott
ORG:Dunder Mifflin
TITLE:Regional Manager
TEL;TYPE=work:tel:+570-344-1212
TEL;TYPE=cell:tel:+
ADR;TYPE=work:Scranton;PA
EMAIL;TYPE=work:mscott@dundermiff.com
URL:https://theoffice.fandom.com/wiki/Michael_Scott
NOTE:Michael Scott is the Regional Manager of Dunder Mifflin.
END:VCARD
```

CREATING A QR CODE FROM A VCARD

The QR macro (thank you again Rob Allison!) contacts the Google API and converts a text file—in this case, the vCard text files that were dynamically created—to a QR code PNG file:

```
%macro qr(in,out);
data;
```

```
   infile &in recfm=f lrecl=4096 length=l;
   length url_encoded $8192;
   keep url_encoded;
   input @1 all $varying4096. l;
   url_encoded='chs=500x500&cht=qr&chl='||urlencode(strip(all))||'&chld=l';
   call symputx('qrtextl',length(url_encoded));
   output;
   stop;
run;

filename qrtext temp recfm=f lrecl=&qrtextl;

data _null_; set;
   file qrtext;
   put url_encoded;
run;

proc delete;
run;

proc http in=qrtext out=&out method='post'
   url='https://chart.googleapis.com/chart?'
   ct='application/x-www-form-urlencoded';
run;
filename qrtext clear;
%mend;
```

The CALLQR macro iterates through the list of vCard files and calls the QR macro to build a QR PNG file from each vCard file:

```
%macro callqr();
%local i;
%do i=1 %to &numpeeps;
   filename vcardin "&loc.vCard&i..vcf";
   filename qrout "&loc.qrcode&i..png";
   %qr(vcardin,qrout);
   filename vcardin clear;
   filename qrout clear;
   %end;
%mend;

%callqr;
```

This invocation of CALLQR, for example, creates QRcode1.png through QRcode6.png. Figure 1 demonstrates QRcode1.png, which represents the vCard data for Michael Scott.

Figure 1. vCard Data for Michael Scott (QRcode.png)

SCANNING A QR CODE AND ADDING A CONTACT

When a user (such as a conference organizer, security guard, or other attendee) scans the QR code in Figure 1 (using a free cell phone app), the user is immediately prompted with choices. For example, Figure 2 demonstrates how this appears on a Samsung smart phone.

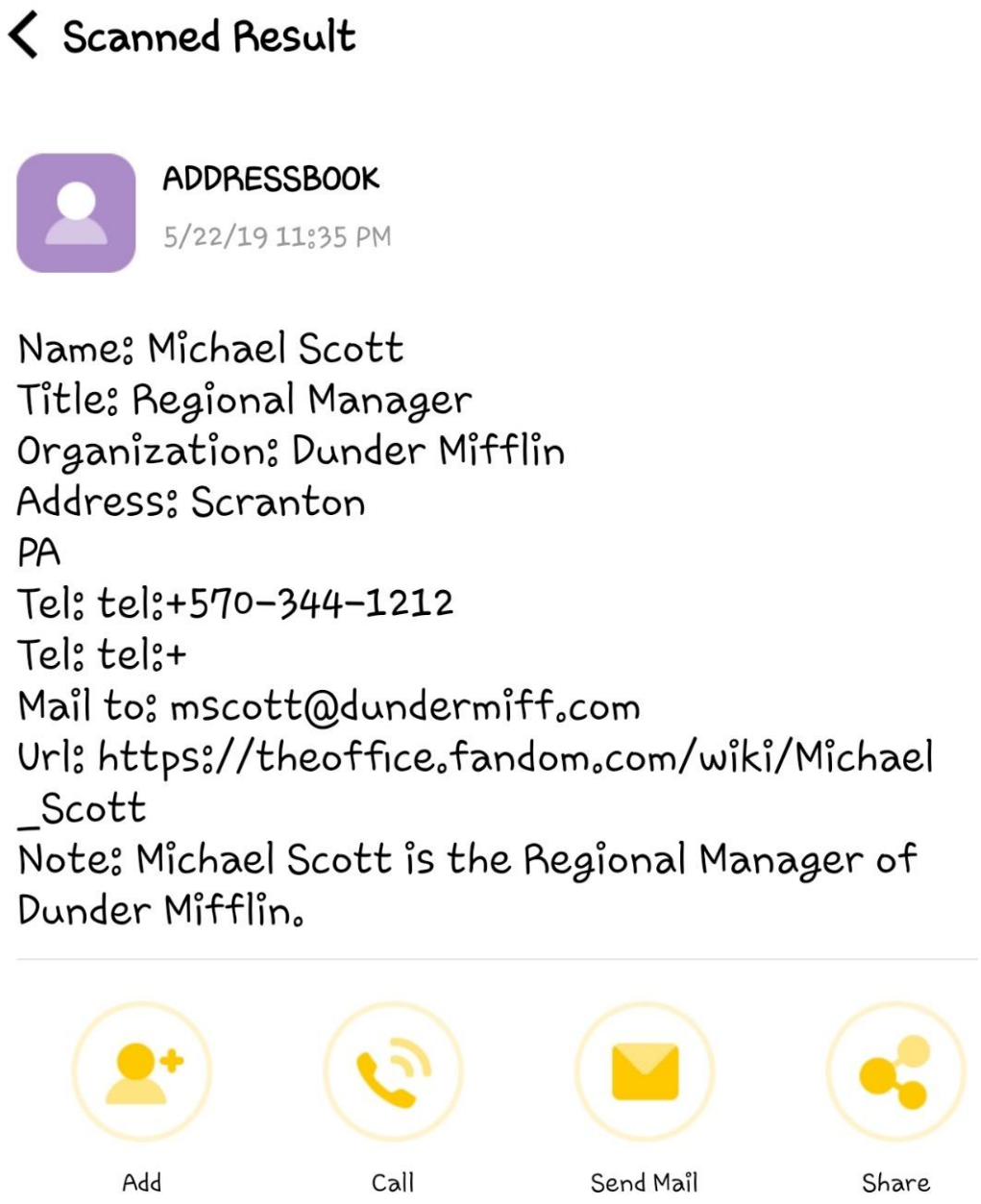

Figure 2. Scanned QR Code Import on a Samsung Smart Phone

Once the vCard information is populated, clicking “Add” will create a contact (i.e., phone book entry) that can be edited. For example, clicking “Add” on Michael Scott is demonstrated in Figure 3.

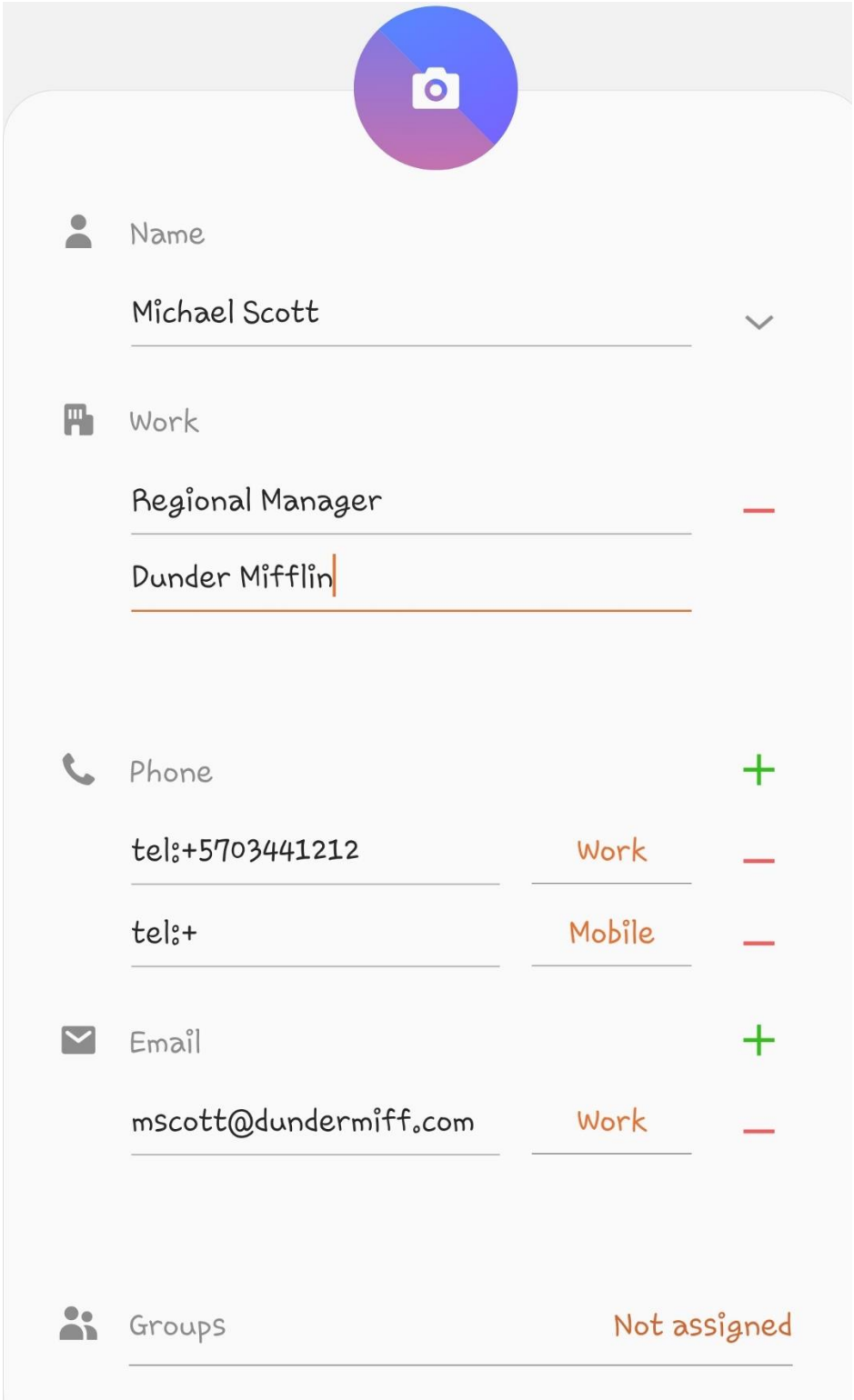

Figure 3. Creating a New Contact

If no edits are required, the contact can be saved immediately. Thus, in five seconds, a conference goer can snap a picture with a QR code scanning app, import the vCard, save the vCard as a new contact, and have all information represented in Table 1 imported into their phone. Figure 4 demonstrates the new contact, again as it appears within a Samsung smart phone.

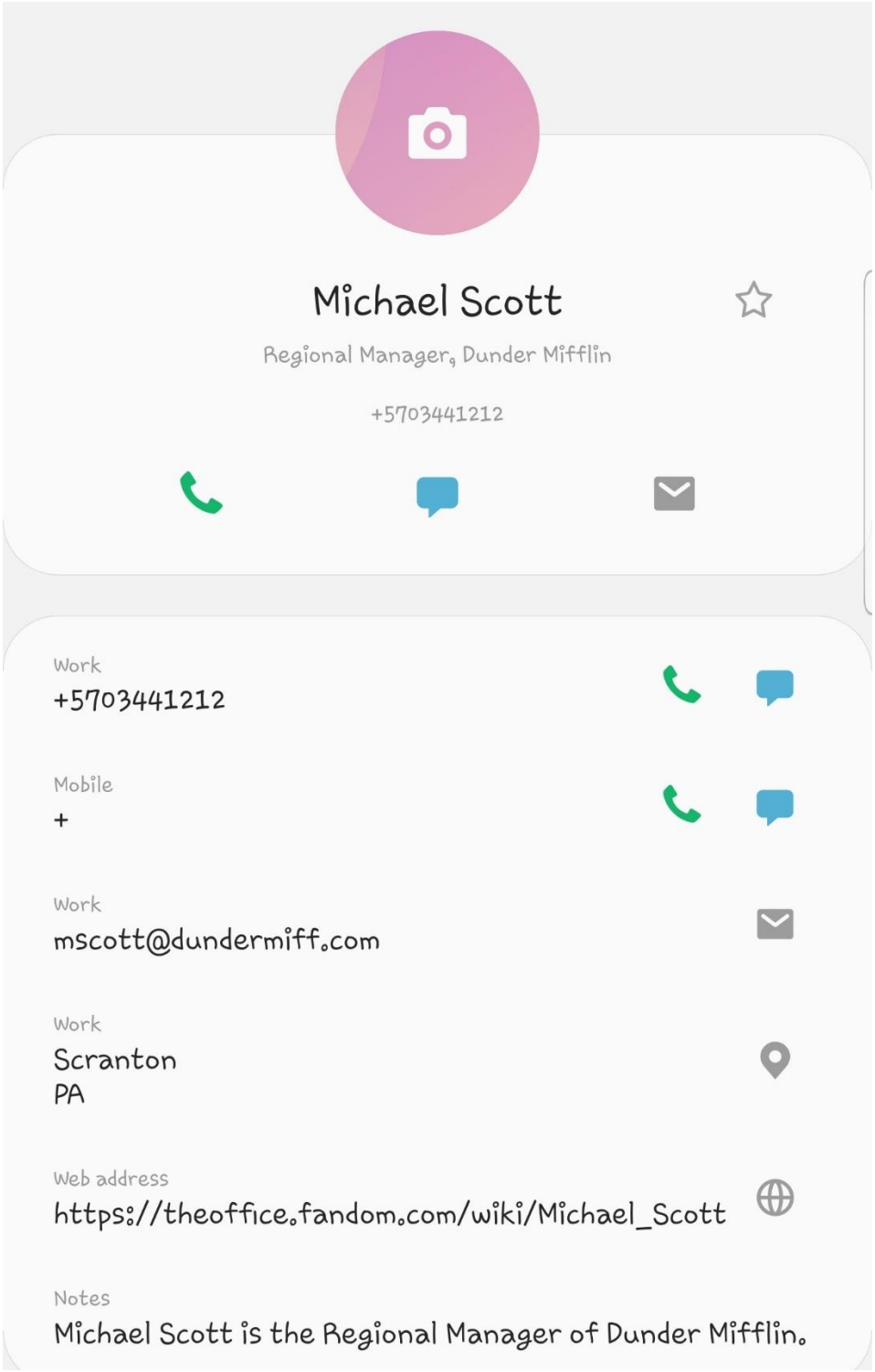

Figure 4. New Contact Saved in Phone

CREATING BADGES

Once the Attendees data set has been populated, the vCard text files have been created, and the QR code images have been created, the final step (for conference organizers) is to run the Honeybadger to create the HTML output that contains the badges:

```
data _null_;
   set honeybgr.attendees end=eof;
   array line $200 line1-line5;
   length cssfile qrfile logofile $200;
   cssfile="&loc.badges.css";
   qrfile="&loc.QRcode" || strip(put(_n_,8.)) || '.png';
   logofile="&loc.pharmasug_logo.jpg";
   file "&loc.badges.html";
   if _n_=1 then do;
      put '<!DOCTYPE html>';
      put '<html>';
      put '<head>';
      put '<link rel="stylesheet" href="' cssfile '">';
      put '</head>';
      put '<body>';
      put '<table>';
      end;
   line1='<div class="f1">' || strip(firstName) || ' ' || strip(lastName)
      || '</div><br>';
   line2='<div class="f2">' || strip(jobTitle) || ', ' || strip(company)
      || '</div><br>';
   line3='<div class="f3">' || strip(city) || ', ' || strip(state)
      || '</div><br><br>';
   line4='<img class="qr" src="' || strip(qrfile) || '">';
   line5='<img class="logo" src="' || strip(logofile) || '">';
   if mod(_n_,2)=1 then put '<tr>';
   put '  <td>';
   do over line;
      put '    ' line;
      end;
   put '  </td>';
   if mod(_n_,2)=0 then put '</tr>';
   if mod(_n_,2)=1 and eof then put '</tr>';
   if eof then put '</table></body></html>';
run;
```

When the DATA step executes, the Badges.html file is created, which references the Badges.css stylesheet from which all style attributes are derived:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet"
href="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\badges.css ">
</head>
<body>
<table>
<tr>
  <td>
    <div class="f1">Michael Scott</div><br>
    <div class="f2">Regional Manager, Dunder Mifflin</div><br>
    <div class="f3">Scranton, PA</div><br><br>
    <img class="qr"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\QRcode1.png">
    <img class="logo"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\pharmasug_logo.jpg"
>
  </td>
  <td>
    <div class="f1">Jim Halpert</div><br>
    <div class="f2">Salesman, Dunder Mifflin</div><br>
    <div class="f3">Scranton, PA</div><br><br>
```

```
    <img class="qr"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\QRcode2.png">
    <img class="logo"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\pharmasug_logo.jpg"
>
  </td>
</tr>
<tr>
  <td>
    <div class="f1">Dwight Schrute</div><br>
    <div class="f2">Assistant to the Regional Manager, Dunder Mifflin</div><br>
    <div class="f3">Scranton, PA</div><br><br>
    <img class="qr"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\QRcode3.png">
    <img class="logo"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\pharmasug_logo.jpg"
>
  </td>
  <td>
    <div class="f1">Pam Halpert</div><br>
    <div class="f2">Office Manager, Dunder Mifflin</div><br>
    <div class="f3">Scranton, PA</div><br><br>
    <img class="qr"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\QRcode4.png">
    <img class="logo"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\pharmasug_logo.jpg"
>
  </td>
</tr>
<tr>
  <td>
    <div class="f1">Angela Martin</div><br>
    <div class="f2">Accountant, Dunder Mifflin</div><br>
    <div class="f3">Scranton, PA</div><br><br>
    <img class="qr"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\QRcode5.png">
    <img class="logo"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\pharmasug_logo.jpg"
>
  </td>
  <td>
    <div class="f1">Meredith Palmer</div><br>
    <div class="f2">Supplier Relations, Dunder Mifflin</div><br>
    <div class="f3">Scranton, PA</div><br><br>
    <img class="qr"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\QRcode6.png">
    <img class="logo"
src="C:\Users\manny\stuff\SAS\20190615_PharmaSUG_Phily\QR_codes\pharmasug_logo.jpg"
>
  </td>
</tr>
</table></body></html>
```

Badges.html is demonstrated in Figure 5. Note that six badges are printed per page, allowing conference organizers to print multiple pages on conveniently accessible 3” x 4” perforated (or sticky) badge paper. In Figure 5, a black outline (optional) has been added only to differentiate badge edges. This and other stylistic customizations can be made by modifying the Badges.css file.

Figure 5. Badges.html

INPUT FILES REQUIRED TO PRODUCE BADGES

The previous DATA step requires three input files to complete: a CSS file that prescribes the HTML style and formatting, the QR code (unique to each attendee, and saved as PNG files), and the PharmaSUG logo (saved as d:\sas\honeybadger\pharmasug_logo.jpg).

The default CSS file (d:\sas\honeybadger\badges.css) follows:

```
body {
  margin: 25px;
  background-color: rgb(256,256,256);
  font-family: arial, sans-serif;
  font-size: 14px;
}

div
{
  display: inline-block;
}

table {
  table-layout: fixed;
}

td {
  width:4in;
  height:3in;
  max-width:4in;
  min-width:4in;
  max-height:3in;
  min-height:3in;
  border:1px solid black;
  text-align: center;
  vertical-align: bottom;
}

.f1 {
  font:arial, sans-serif;
  font-size: 20pt;
  font-style: normal;
  font-weight: bold;
}

.f2 {
  font:arial, sans-serif;
  font-size: 14pt;
  font-style: italic;
  font-weight: normal;
}

.f3 {
  font:times new roman, sans-serif;
  font-size: 10pt;
  font-style: normal
  font-weight: normal;
}

.qr {
  border: 0in;
  padding: 0in;
  display: block;
  float: left;
  vertical-align: bottom;
  margin-bottom: 0px;
  bottom: 0px;
  width: 1.25in;
  height: 1.25in;
}

.logo {
  border: 0in;
```

```
    padding: 0in;
    float: right;
    vertical-align: text-bottom;
    left: 2.75in;
    height: 1.25in;
  }
```

By modifying the styles specified in the CSS file (not described in this text), users can customize every aspect of the badges that are printed. The Honeybadger DATA step likewise can be modified so that specific attendee metadata are included or omitted. Through these two mechanisms, conference organize have total autonomy to customize badges with very little effort.

CONCLUSION

This data-driven solution to conference badge creation puts conference organizers squarely in the driver's seat. Honeybadger lets organizers control the information printed on the badge, the font and other style elements, and the size and location of the QR code and optional conference logo. Attendees also have control over badge information, and are able to indicate what metadata should be made available to conference attendees (in the QR code) versus metadata that should be available only to conference organizers (in the conference registration database). Finally, given the often-exorbitant cost of third-party badge vendors, this home-grown solution offers a huge cost savings to conference organizers who can redirect it toward more meaningful activities.

REFERENCES

[i] Robert Allison. *How to create your own QR codes with SAS!* Retrieved from https://blogs.sas.com/content/sastraining/2016/09/12/how-to-create-your-own-qr-codes-with-sas/.

[ii] Robert Allison. Code, including his SAS macro TO_QR. Retrieved from http://robslink.com/SAS/democd88/qr_robslink.sas.

[iii] Google Charts Infographics: QR Codes. Retrieved from https://developers.google.com/chart/infographics/docs/qr_codes.

[iv] vCard Format Specification. Retrieved from https://tools.ietf.org/html/rfc6350.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Troy Martin Hughes
E-mail: troymartinhughes@gmail.com