PhUSE 2015

Paper TT02

How to create a mobile WebApp? Walkthrough example including SAS output

Katja Glaß, Bayer Pharma AG, Berlin, Germany

ABSTRACT

Apps have become a major part of our lives. Also in our clinical working environment we would like to see and show results and perform data entries quickly via mobile phones or tablets. Of course we have the webpage options, but they look so old-fashioned. Today it is state of the art to provide an app for various displays and tasks.

Creating native apps is always restricted to one platform and involves quite complex programming especially compared with the alternative. Why not create a WebApp? These are as easy to create as HTML pages (as it is HTML) and are platform independent! They have a native look and feel and can even be compiled to native apps and uploaded to the app stores!

This presentation will give an introduction and an example on how to create a nice WebApp without any specific tool. It will provide a starting point for developing your own WebApps and will show how SAS output can be included.

HOW TO GET STARTED

There is always a reason behind creating an App, so what should people do with the App? We might want to create an App to display some SAS outputs like tables and graphics. We would also like to have easy navigation, display some free text and of course it must be stylish!

If you have a more business related topic in mind like using Apps for research or marketing, you might want to save time and resources by checking what tools are already available. For example, SAS provides an App development tool called SAS App Dev Studio. There are plenty of tools and companies out there which are specialized on App creation.

WEBAPP VS. NATIVE APP

When creating an app, there are two major ways to do so, namely creating a native App or a WebApp. The native Apps are programmed for specific platforms, e.g. for iOS, Android and Windows separately. So the corresponding programming languages and interfaces have to be used. A WebApp is an HTML5 page which looks like an app but is actually just a web page. As most devices support web views, WebApps can be used on most devices.

WebApps do not support all the functions which are available through native apps. Furthermore WebApps are typically not as performant as native apps. However, a WebApp is sufficient for most of our purposes, and as this is easier to create and it supports various devices, this is what we will create.

PROVIDING WEBAPPS

Hosting. Another question is how users can access the WebApp. The WebApp can be stored on a web server and simply accessed through a web browser. Then no installation or anything is required. This means setup is very simple, but users still see the web browser's navigation bars. There are options available which enable those webpages which are WebApps to be downloaded for offline usage. But this requires some user knowledge, which the user might not have.

Getting Native. A much nicer way to deploy a WebApp is through the stores. The stores require native app versions compiled for the specific operating systems like iOS, Android and Windows. Luckily there are ways to convert WebApps into native apps. There is a free tool called PhoneGap which needs some configuration and installation. There is also the Adobe® PhoneGap™ Build service available where you can easily convert your WebApps to native apps without any installation. Plenty of other providers and services like “Go Native” also provide easy solutions for generating native Apps out of WebApps.

Store. To be able to create native apps for a store, you need special developer accounts, e.g. an iDeveloper account. You must generate the keys required to create, deploy and upload the App to the store. So there is still a certain amount of configuration necessary. Be aware that for Android you can provide a file which can simply then be installed on any device (if the security settings allow this), but for iOS your app can only be deployed on a device through the Apple Store.

WHAT AM I LOOKING FOR?

The main question is still, how to get started? Everything we need to know is on the internet. The major problem with the internet is that it is often tough to find what you need, especially if you do not know the technical terms. If your background knowledge is quite low, a book might be the right place to start.

A book will hopefully provide you with a good understanding about what's possible and also give you the technical terms which can then be searched for on the internet. There are intriguing things like "JQuery Mobile", "Sencha Touch", "Theme Roller" and many more to look out for! So these are some keywords you might want to google. If you know how to start, for example with "JQuery Mobile", you will easily find the tutorials you need to move forward.

SETTING UP THE ENVIRONMENT

Theoretically you do not need anything special to start programming a WebApp. As this is "just" an HTML page you can easily use a text editor and create your page there. But it might be a good idea to use special software with features like auto completion, highlighting and similar. As an example, Eclipse® is always a nice tool for any kind of programming, including HTML, but there are plenty of other HTML editors available.

You will have to select a browser for the development. Depending on the final platform used, Google Chrome or Safari might be good browsers for development. Some versions of Internet Explorer do not support major HTML5 functions and should be avoided. For fine tuning it is recommended to check the WebApp with various browsers to see that there are no major issues.

As soon as you start using more complex webpage functions you will require web services. JQuery Mobile itself already requires this for some functions. So you might want to install XAMPP to support local development on your PC. When you have a webhosting service from T-Online, GMX or similar, web services are automatically pre-installed, so the related files can be uploaded and displayed in a browser with web services running in the background.

In case you want to work with other colleagues and share your WebApp development you might want to use code repositories like GitHub, Google Code, Bit Bucket or other tools. This paper's WebApp is available in the PhUSE repository[1] which can also be used for co-working and sharing anything related to our working environment.

CREATE THE WEBAPP

START THE PROGRAMMING

After knowing how to start and after setting up the environment, the programming can begin. So, in time-honored tradition, let's get started with a simple "Hello World" WebApp!

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Web App Example</title>
</head>
<body>
Hello World
</body>
</html>
```

Right now, this looks just like a webpage, but as we develop the code, it will become more and more a WebApp. A web page contains various HTML tags which define how the page is displayed in browsers. The tags typically come in pairs, a beginning and an ending tag of the same type. Many layout and basic functions can be associated with tags, for example defining font sizes. There are various HTML tags available which are responsible for different tasks.[2] This example has tags for defining that this is an HTML file containing a head with general information and a body containing the text.

Having read that JQuery Mobile might be a nice solution to easily create a WebApp we are searching for tutorials to try it out and start understanding how it works.[3] The following example contains a nice layout and will be our starting point.

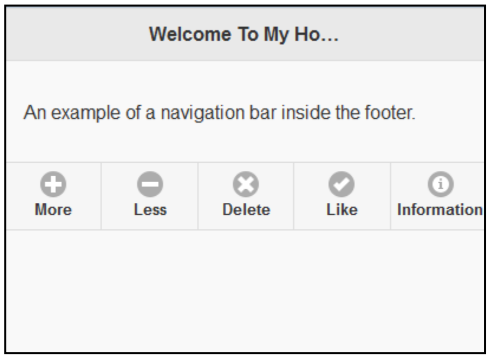
```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.css">
<script src="http://code.jquery.com/jquery-1.11.3.min.js"></script>
<script src="http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.js"></script>
</head>
<body>

<div data-role="page" id="pageone">
  <div data-role="header">
    <h1>Welcome To My Homepage</h1>
  </div>

  <div data-role="main" class="ui-content">
    <p>An example of a navigation bar inside
       the footer.</p>
  </div>

  <div data-role="footer">
    <div data-role="navbar">
      <ul>
        <li><a href="#" data-icon="plus">More</a></li>
        <li><a href="#" data-icon="minus">Less</a></li>
        <li><a href="#" data-icon="delete">Delete</a></li>
        <li><a href="#" data-icon="check">Like</a></li>
        <li><a href="#" data-icon="info">Information</a></li>
      </ul>
    </div>
  </div>
</div>

</body>
</html>
```


Welcome To My Ho...

An example of a navigation bar inside the footer.

Figure 1: Screenshot - JQuery Mobile Example

The source already looks a little more complex. The head now contains some additional settings. There is a Style Sheet (CSS) and two Java Script files (JS). JQuery Mobile files provide intuitive classifications such as the “data-role” container. The body contains a “page” div-container and the page itself is divided into header, content and footer. Furthermore there is a special “navbar” container with a list.

The JQuery Mobile documentation provides very good examples for the different elements this tool supports. But how does it work? What are CSS and JS about?

AVAILABLE ELEMENTS – HTML, CSS AND JS

HTML is a markup language. It describes the structure and content of information. It is like a word document containing texts, graphics and hyperlinks. Cascading Style Sheets (CSS) describe the style of the displayed page. Using CSS, style elements like font sizes can be defined in this separate style defining files. Furthermore very special styles like edgy squares or the detailed sizes and layouts of buttons and layout effects can be described in CSS files. Java Script (JS) makes the pages dynamic. Here programming elements like loops and if-then-else conditions can be programmed. Plugins like JQuery Mobile are typically a combination of CSS and JS files which contain the complete functionality. Those components can then be embedded via HTML tags or called via Java Script commands to modify the layout.

SKETCHING

As we are overwhelmed by many possible nice looking layouts and examples we have to think about what we want to display and then search for the corresponding functionality. So we first sketch our WebApp on paper.

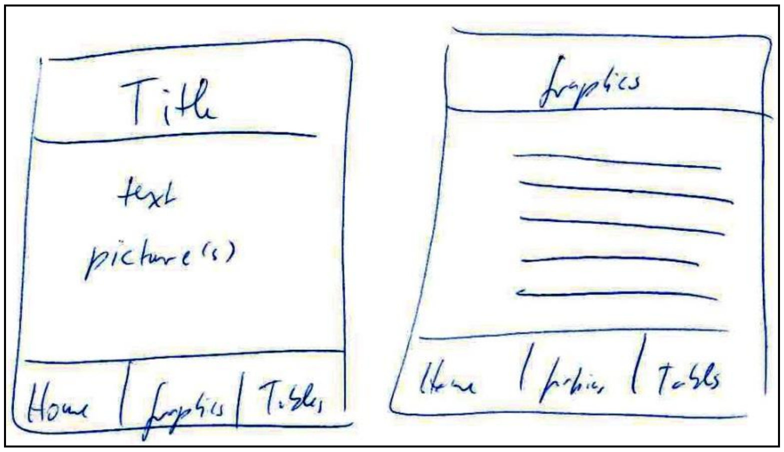


Figure 2: WebApp Sketch via Hand

When working in collaboration it might be a good idea to use a tool to create the sketch as people might not be able to read the handwriting of other programmers. You can find plenty of tools if you search for “free web app prototyping tool”.[4]

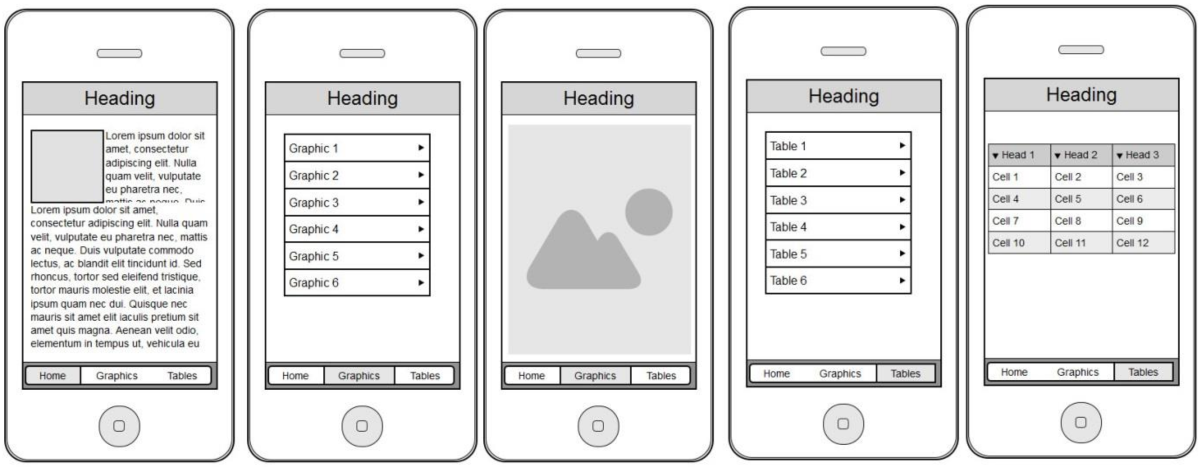
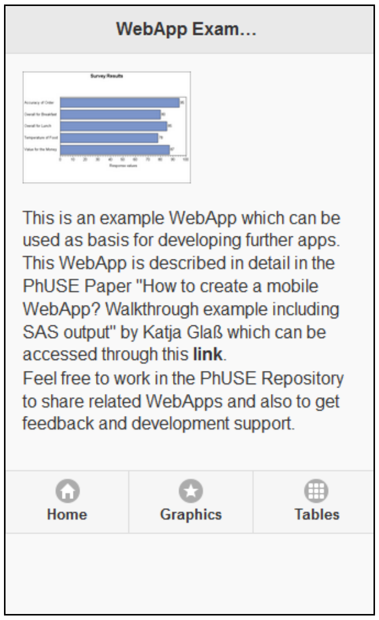


Figure 3: WebApp Sketch via Tool

This WebApp should contain a “Home” area with general text and probably a picture. Then we need a “Graphics” area where we have a list of graphics to select and another page to display the selected graphic. Finally we also want another “Table” area with a list of tables as well as pages to display the tables.

INCLUDE CONTENT

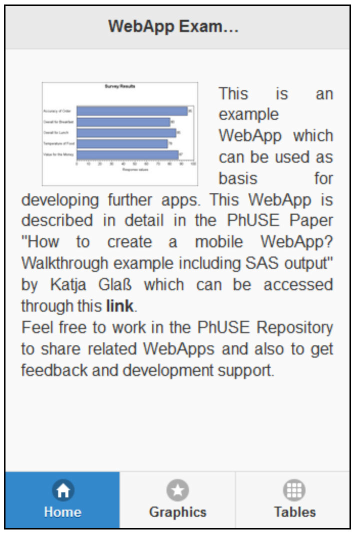
As we see the elements we want to use, we can copy and modify the bits out of the examples available. We use the skeleton and fill this with our “Home” area. “WebApp Example” is used as the page title; the footer is modified to contain our three areas “Home”, “Graphics” and “Tables”; the icons are exchanged to fit our purpose.[5] Then we include a graphic and text. We use the simple <img> and <p> HTML tags and define the width of the image as 50%. Somehow the display is not as pretty as sketched (see right).



This is an example WebApp which can be used as basis for developing further apps. This WebApp is described in detail in the PhUSE Paper "How to create a mobile WebApp? Walkthrough example including SAS output" by Katja Glaß which can be accessed through this **link**.
Feel free to work in the PhUSE Repository to share related WebApps and also to get feedback and development support.

Figure 4: Screenshot first draft "Home"

To finally create a nice styling some internet searching is required. The footer should be fixed at the bottom of the page regardless of whether the content fits the page or not. The “Home” area in the footer is marked blue to provide the users a simple view of the current location. And finally the text should float around the image and some free space between the image and the text should be provided. After finding the correct HTML tags and attributes the initial page looks as specified (see left).



This is an example WebApp which can be used as basis for developing further apps. This WebApp is described in detail in the PhUSE Paper "How to create a mobile WebApp? Walkthrough example including SAS output" by Katja Glaß which can be accessed through this **link**.
Feel free to work in the PhUSE Repository to share related WebApps and also to get feedback and development support.

Figure 5: Screenshot "Home"

GRAPHICS

The “Graphics” page should be programmed next. Here a list should be created linking to the various graphics. Instead of creating separate HTML files per “page”, those pages can also be included in one file. We copy the “Home” area to use the same skeleton for the “Graphics” area and modify where required.

The JQuery Mobile “List View” looks very promising, so we copy an example[6] and modify the content to include the six selected example pictures from the SAS Graph Gallery.[7]

```
<div data-role="main" class="ui-content">
      <ul data-role="listview" data-inset="true">
    <li data-role="list-divider">SAS Example Graphics</li>
    <li><a href="#example1">Survay Results</a></li>
    <li><a href="#example2">Height of Students</a></li>
    <li><a href="#example3">Centered Subgroup Labels</a></li>
    <li><a href="#example4">Default Graph of Sales Totals</a></li>
    <li><a href="#example5">Annotate standard error bars</a></li>
    <li><a href="#example6">Overlay a line on top of GCHART …</a></li>
  </ul>
</div>
```

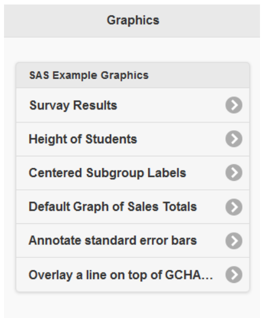

Graphics
SAS Example Graphics
Survay Results
Height of Students
Centered Subgroup Labels
Default Graph of Sales Totals
Annotate standard error bars
Overlay a line on top of GCHA...

Figure 6: Graphics List View

The next step is to create the single example1 … example6 pages containing the graphics themselves. We can copy the skeleton again and just include an <img> tag to include the corresponding image. We use a weight of 100% as the majority of our images are landscape formatted.

```
<div data-role="main" class="ui-content">
     <img src="sas_output/06_sample_24865.gif" width="100%"></img>
</div>
```

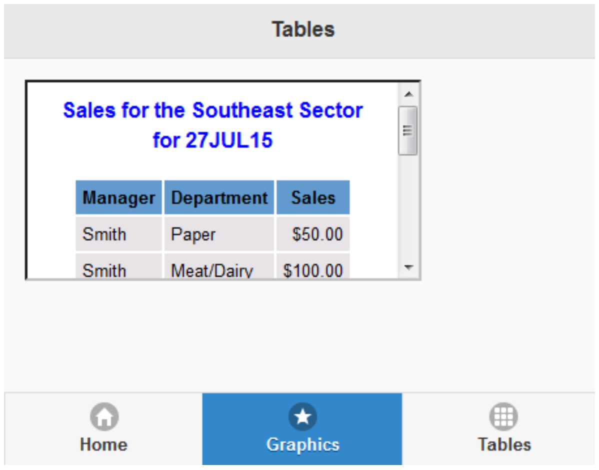
TABLES

For the table examples, SAS examples are used.[8] With SAS the tables can be created in various formats. The simple text output format might not be fancy enough for the WebApp, so we might want to use real tables. When looking into the most common ODS formats – PDF, RTF and HTML – we might be able to include the HTML output directly within our WebApp. And as ODS provides great styling options, we can easily use a nice style.

When searching for “embed html in html” it seems that “IFRAME” might be a good solution here. First of all we create a list again to link to the various table outputs, just the same as for the graphics. Then we create a first page using an IFRAME to include one of the SAS HTML output examples.

This might look like the following:

```
<div data-role="main" class="ui-content">
  <iframe src="../sas_output/example1.html"></iframe>
</div>
```



Sales for the Southeast Sector
for 27JUL15

Manager	Department	Sales
Smith	Paper	$50.00
Smith	Meat/Dairy	$100.00

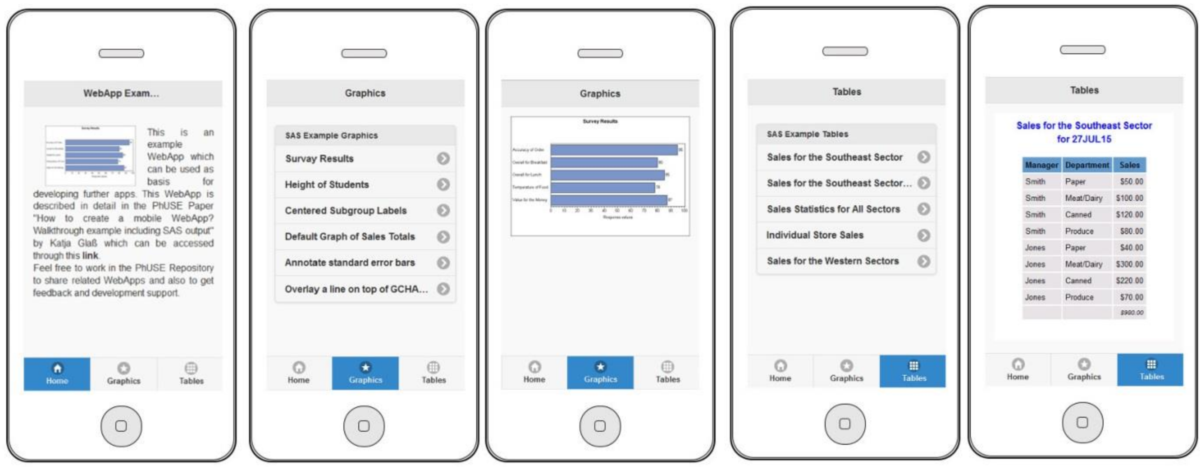
Figure 7: Screenshot first draft IFRAME

The result is not optimal as we would like the IFRAME to fill the complete content area. After searching for some IFRAME attributes, we try the width and height set to 100% and a frame border of zero.

The height does not work as the page does not know 100% of what. The easiest solution is to set the height to a specific value. As mobiles and browsers have different heights, the specific “window-height” can be used. We need the Java Script language to investigate and set this value for the IFRAME tag. The script itself should be executed at a specific time. Using “pageshow” is a common technique. So finally our IFRAME and Java Script look like the following:

```
<div data-role="main" class="ui-content">
    <iframe id="iframe_1" src="../sas_output/example1.html" width="100%" frameborder="0">
    </iframe>
</div>
<script type="text/javascript">
       $('#t_example1').on("pageshow", function () {
              document.getElementById("iframe_1").height = ($(window).height()) - 140;
       });
</script>
```

As the final layout for the display of HTML pages is now done, we copy and update this first HTML example page to create one for each HTML output. And that is it! The app is running and looking very good.


WebApp Exam...

This is an example WebApp which can be used as basis for developing further apps. This WebApp is described in detail in the PhUSE Paper "How to create a mobile WebApp? Walkthrough example including SAS output" by Katja Glaß which can be accessed through this **link**.
Feel free to work in the PhUSE Repository to share related WebApps and also to get feedback and development support.

Graphics

SAS Example Graphics
- Survay Results
- Height of Students
- Centered Subgroup Labels
- Default Graph of Sales Totals
- Annotate standard error bars
- Overlay a line on top of GCHA...

Tables

SAS Example Tables
- Sales for the Southeast Sector
- Sales for the Southeast Sector...
- Sales Statistics for All Sectors
- Individual Store Sales
- Sales for the Western Sectors

Sales for the Southeast Sector for 27JUL15

Manager	Department	Sales
Smith	Paper	$50.00
Smith	Meat/Dairy	$100.00
Smith	Canned	$120.00
Smith	Produce	$80.00
Jones	Paper	$40.00
Jones	Meat/Dairy	$300.00
Jones	Canned	$220.00
Jones	Produce	$70.00
		$980.00

Figure 8: Screenshots - final App

NEXT STEPS

GETTING DYNAMIC

As a programmer it seems strange to copy the same text over and over again with just minimal changes. To start with it might be an acceptable solution, but there should be better ways. Of course there are. Java Script is designed for such tasks.

A list of graphics can be created as a variable and out of this the HTML elements can be constructed. For the graphics display one dynamic page can be generated having something like a “parameter” which defines the graphic it should contain. This process can be done for the display of the tables as well.

Finally instead of updating the HTML and/or java script whenever the content changes, even the content can be provided more dynamically. As an example, the graphic and table metadata can be stored in a so-called JSON file. JSON is just like XML, a file describing objects or variables. If programmed with java script, the HTML reads which graphics and tables should be displayed and creates the corresponding lists and calls.

GIVE IT A STYLE

The current style as the default style is quite common. As this WebApp should be a bit special, a different style would be nice. To modify all elements of the pages is quite an overwhelming task. Luckily we can use the Cascading Style Sheets we mentioned above. With such a CSS file the general layout and specific style elements can be specified.

But even CSS file creation looks quite complicated as we would need to learn and understand the CSS principles. Luckily there is a tool! With ThemeRoller® a custom style for JQuery mobile can easily be clicked together and used.

MEDIA QUERIES

“Media Queries” are nice HTML elements. With these CSS style elements the layout for different devices can be controlled. So we can easily adapt the layout for smartphones and tablets. For our graphic displays we might want to display more information on one page for tablets, e.g. to display the list of graphics on the left-hand side and the selected graphic on the right-hand side on one screen.

GETTING INTERACTION

Possible interactions with the users are another major element of WebApps. User information should be stored and used. A database like for normal web pages can be used to store user information and to provide content. The “Local Storage” option has enhanced and powerful storage capabilities and are much better than cookies. With this storage user-provided information can be stored on the user device itself and no database connection is required. For example, users might want to mark special graphics where they want to focus further research on their PCs.

USING EXTENSIONS

As web applications and web pages are extremely common and used nearly everywhere, there are many examples and also extensions available as add-ons or plugins. An extension is a tool package which can typically be “installed” simply by including the corresponding Java Script (JS) and Cascading Style Sheet (CSS) files. There are many tools, for example to create a nice gallery, to include a star rating and moreover JQuery Mobile is “just” an extension. Some of them are not compatible with each other or require other add-ons, so some investigation is always required, but these provide great results with little effort.

GETTING SAS CONNECTED

SAS provides a range of options for web pages and these can of course also be used for WebApps which are basically “just” web pages too. The SAS stored processes should theoretically be usable by WebApps you create yourself. The SAS Connect functionality should also work. With these options many potentially useful applications in our working environment are possible.

POSSIBLE AREAS FOR SUCH WEBAPPS IN CLINICAL RESEARCH & DEVELOPMENT

There is still the main question of whether to buy some WebApp software or to develop this from scratch. There are some areas where self-development is quite useful. For example a WebApp can be developed to display standard layouts or a standard table catalog.

Also manual and training materials could be provided via WebApp to give users quick glances of available options. For detailed investigations and studying of user manuals the classical paper formats like Word or PC optimized HTML pages are to be preferred, but to provide simple overviews a WebApp might be a nice gimmick.

CONCLUSION

As shown here, it is easily possible to create a WebApp. The major initial problem is finding the right technical terms and suitable examples on the internet. This paper at least provides you with a good starting point. The source code for our WebApp is in the PhUSE Repository and maybe you would like to enhance it further or develop another WebApp using the repository.

SOURCE

```
<!DOCTYPE html>
<html>
<head>
        <!-- creating the "HOME" area -->
        <meta charset="ISO-8859-1">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <link rel="stylesheet" href="http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.css">
        <script src="http://code.jquery.com/jquery-1.11.3.min.js"></script>
        <script src="http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.js"></script>
</head>
<body>

<div data-role="page" id="home">
  <div data-role="header">
    <h1>WebApp Example</h1>
  </div>

  <div data-role="main" class="ui-content">
        <p align="justify">
    <img src="sas_output/01_sample_52468.png" width="50%" align="left" hspace="20">
    This is an example WebApp which can be used as basis for developing further apps. This WebApp
    is described in detail in the PhUSE Paper "How to create a mobile WebApp? Walkthrough example
    including SAS output"
    by Katja Glaß which can be accessed through this <b>link</b>.<br>
    Feel free to work in the PhUSE Repository to share related WebApps and also to get feedback and
    development support.</p>
  </div>

  <div data-role="footer" data-position="fixed" data-tap-toggle="false">
    <div data-role="navbar">
      <ul>
        <li><a href="#home" data-icon="home" class="ui-btn-active ui-state-persist">Home</a></li>
        <li><a href="#graphics" data-icon="star">Graphics</a></li>
        <li><a href="#tables" data-icon="grid">Tables</a></li>
      </ul>
    </div>
  </div>
</div>

<div data-role="page" id="graphics">
  <div data-role="header">
    <h1>Graphics</h1>
  </div>

  <div data-role="main" class="ui-content">
        <ul data-role="listview" data-inset="true">
      <li data-role="list-divider">SAS Example Graphics</li>
      <li><a href="#example1">Survay Results</a></li>
      <li><a href="#example2">Height of Students</a></li>
      <li><a href="#example3">Centered Subgroup Labels</a></li>
      <li><a href="#example4">Default Graph of Sales Totals</a></li>
      <li><a href="#example5">Annotate standard error bars</a></li>
      <li><a href="#example6">Overlay a line on top of GCHART output using Annotate</a></li>
    </ul>
  </div>

  <div data-role="footer" data-position="fixed" data-tap-toggle="false">
    <div data-role="navbar">
      <ul>
        <li><a href="#home" data-icon="home">Home</a></li>
        <li><a href="#graphics" data-icon="star" class="ui-btn-active ui-state-persist">Graphics</a></li>
        <li><a href="#tables" data-icon="grid">Tables</a></li>
      </ul>
    </div>
  </div>
</div>

<div data-role="page" id="example1">
```

```
  <div data-role="header">
    <h1>Graphics</h1>
  </div>

  <div data-role="main" class="ui-content">
        <img src="sas_output/01_sample_52468.png" width="100%"></img>
  </div>

  <div data-role="footer" data-position="fixed" data-tap-toggle="false">
    <div data-role="navbar">
      <ul>
        <li><a href="#home" data-icon="home">Home</a></li>
        <li><a href="#graphics" data-icon="star" class="ui-btn-active ui-state-persist">Graphics</a></li>
        <li><a href="#tables" data-icon="grid">Tables</a></li>
      </ul>
    </div>
  </div>
</div>

<div data-role="page" id="example2">
  <div data-role="header">
    <h1>Graphics</h1>
  </div>

  <div data-role="main" class="ui-content">
        <img src="sas_output/02_sample_51732.gif" width="100%"></img>
  </div>

  <div data-role="footer" data-position="fixed" data-tap-toggle="false">
    <div data-role="navbar">
      <ul>
        <li><a href="#home" data-icon="home">Home</a></li>
        <li><a href="#graphics" data-icon="star" class="ui-btn-active ui-state-persist">Graphics</a></li>
        <li><a href="#tables" data-icon="grid">Tables</a></li>
      </ul>
    </div>
  </div>
</div>

<div data-role="page" id="example3">
  ...
</div>

<div data-role="page" id="example4">
  ...
</div>

<div data-role="page" id="example5">
    ...
</div>

<div data-role="page" id="example6">
    ...
</div>

<div data-role="page" id="tables">
  <div data-role="header">
    <h1>Tables</h1>
  </div>

  <div data-role="main" class="ui-content">
        <ul data-role="listview" data-inset="true">
      <li data-role="list-divider">SAS Example Tables</li>
      <li><a href="#t_example1">Sales for the Southeast Sector</a></li>
      <li><a href="#t_example2">Sales for the Southeast Sector (2)</a></li>
      <li><a href="#t_example3">Sales Statistics for All Sectors</a></li>
      <li><a href="#t_example4">Individual Store Sales</a></li>
      <li><a href="#t_example5">Sales for the Western Sectors</a></li>
    </ul>
  </div>
  <div data-role="footer" data-position="fixed" data-tap-toggle="false">
    <div data-role="navbar">
      <ul>
        <li><a href="#home" data-icon="home">Home</a></li>
        <li><a href="#graphics" data-icon="star">Graphics</a></li>
        <li><a href="#tables" data-icon="grid" class="ui-btn-active ui-state-persist">Tables</a></li>
      </ul>
    </div>
  </div>
</div>

<div data-role="page" id="t_example1">
  <div data-role="header">
    <h1>Tables</h1>
  </div>

  <div data-role="main" class="ui-content">
    <iframe id="iframe_1" src="sas_output/example1.html" width="100%" frameborder="0"></iframe>
  </div>

  <div data-role="footer" data-position="fixed" data-tap-toggle="false">
    <div data-role="navbar">
      <ul>
        <li><a href="#home" data-icon="home">Home</a></li>
        <li><a href="#graphics" data-icon="star">Graphics</a></li>
        <li><a href="#tables" data-icon="grid" class="ui-btn-active ui-state-persist">Tables</a></li>
      </ul>
    </div>
  </div>
</div>

<div data-role="page" id="t_example2">
    ...
</div>

<div data-role="page" id="t_example3">
    ...
</div>

<div data-role="page" id="t_example4">
    ...
</div>

<div data-role="page" id="t_example5">
    ...
</div>

<script type="text/javascript">
        $('#t_example1').on("pageshow", function () {
                document.getElementById("iframe_1").height = ($(window).height()) - 140;
        });
        $('#t_example2').on("pageshow", function () {
                document.getElementById("iframe_2").height = ($(window).height()) - 140;
        });
        $('#t_example3').on("pageshow", function () {
                document.getElementById("iframe_3").height = ($(window).height()) - 140;
        });
        $('#t_example4').on("pageshow", function () {
                document.getElementById("iframe_4").height = ($(window).height()) - 140;
        });
        $('#t_example5').on("pageshow", function () {
                document.getElementById("iframe_5").height = ($(window).height()) - 140;
        });
</script>

</body>
</html>
```

REFERENCES

[1] PhUSE Script Repository – containing the paper's WebApp example
http://code.google.com/p/phuse-scripts/
http://code.google.com/p/phuse-scripts/source/browse/#svn%2Ftrunk%2Flang%2Fhtml%2Freport
https://github.com/phuse-org/phuse-scripts
https://github.com/phuse-org/phuse-scripts/blob/master/lang/html/report/webAppExample.html

[2] HTML Element Reference
https://developer.mozilla.org/en-US/docs/Web/HTML/Element

[3] JQuery Mobile Examples
http://www.w3schools.com/jquerymobile/tryit.asp?filename=tryjqmob_navbars_footer
https://learn.jquery.com/jquery-mobile/getting-started/
http://www.w3schools.com/jquerymobile/jquerymobile_examples.asp
http://demos.jquerymobile.com/1.4.5/

[4] WebApp prototyping example tool (free for pure clicking / no account required for basic prototyping)
https://moqups.com

[5] JQuery Mobile Icons
https://api.jquerymobile.com/icons/

[6] JQuery Mobile ListView example
http://www.w3schools.com/jquerymobile/jquerymobile_list_views.asp

[7] SAS Graph Gallery
http://support.sas.com/sassamples/graphgallery/PROC_GCHART.html

[8] Used table output examples from SAS
http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm#a000063977.htm

CONTACT INFORMATION

Author:	Katja Glaß
Institution:	Bayer Pharma AG
Address:	Sellerstr. 32, 13353 Berlin, Germany
E-mail:	Katja.Glass@Bayer.com