David Cowart

David.cowart@telerik.com

Abstract

List of instructions for setting up the v1.0 mobile app in Sitefinity. Sept 13, 2013. This document is intended for developers and advanced business users and assumes that the reader has a fairly broad understanding of the Sitefinity platform.

Footprints Setup Instructions

v1.0

Table of Contents

[Summary 2](#_Toc378147726)

[Public Github Repository 2](#_Toc378147727)

[Components 3](#_Toc378147728)

[Module Builder 3](#_Toc378147729)

[Sitefinity Roles & Users 3](#_Toc378147730)

[Sitefinity Pages 4](#_Toc378147731)

[Mobile App Builder 4](#_Toc378147732)

[Custom APIs & Widgets 5](#_Toc378147733)

[Widgets 5](#_Toc378147734)

[Web API 6](#_Toc378147735)

[Drag and Drop your Widgets 7](#_Toc378147736)

[Using & Demoing the App 7](#_Toc378147737)

[Gotchas 8](#_Toc378147738)

[Users & Roles 8](#_Toc378147739)

[Demoing on Mobile Device 9](#_Toc378147740)

[Images 10](#_Toc378147741)

# Summary

The primary purpose of the Footprints mobile app is to demonstrate the power and possibilities of the Sitefinity Mobile App Builder, thus driving customer engagement. The Footprints app acts as a way to track, and report on, outdoor sales people’s activities. It does this by creating a mobile app, which a sales person can:

* Add metadata about their sales call, like: Title, Date, Address and Summary
* Take an image of the physical address of the building
* Create a widget report that outlines the actives of sales people
* Create a Google map using the Sitefinity (SF) geolocation services that gives a physical representation of where sales people have been

Even though the app was built for a fictitious sales company the functionality could easily be extended to such activities as:

* Charity and Political neighborhood canvassing
* Delivery services
* Municipal damage reporting
  + Documenting vandalism and civil defects and dangers
* Etc…

The app consists of following parts:

* The Footprints custom module
* The Footprints mobile app
* The Footprints custom report widget
* The Footprints custom map widget
* The Footprints Role

## Public Github Repository

* <https://github.com/dpcowart/FootprintsMobileApp.git>

# Components

## Module Builder

The first thing that you will need to do is create a custom module using the Module Builder. This is just a standard custom module so there’s nothing tricky here.

**Name**: Footprints

**Content type (singular):** Footprint

**Parent content type:** None

**Fields of this item:**

* Title
  + Short text
* Summary
  + Long text
* Name
  + Short text
* Image
  + Media (images, videos, files)
* Address
  + Address
    - Address form and map

**Which field is the identifier of the content:** Title

*~ see figure 1 in Images section*

## Sitefinity Roles & Users

Go into the Administration of your Sitefinity site and create a Role called “Footprints” and assign the following permissions:

* Create footprints
* Modify footprints
* View footprints

Then assign any users who you will be demoing with. Not really important who, but use 3 or more for demo purposes.

## Sitefinity Pages

Create the following pages:

* Footprints Reporter
  + This page will house the custom Footprint Reporter widget (described below)
  + This page has a rad-tabstrip and grid that will display Footprints items by Role and User
  + This page can export data to a csv file
  + After adding the widget to the page, click the widget Edit button and then point the page selectors to the appropriate locations of the Footprint Details and Footprints Map View pages.
* Footprint Details
  + This page will house the custom module Footprints
  + This page will display details about individual Footprints
* Footprints Map View
  + This page will house the custom Geolocation Map widget (described below)
  + This page will show a Google Map representation of “stops” by Role and User. It can also be given a date range limit the data returned.

You can use whatever page names you want, the above are just suggestions.

*~ see figures 4, 5, 6 and 7*

## Mobile App Builder

Open the Mobile App Builder and create a new mobile app with the following items:

* **App name**: Footprints
* **Based on**: Footprints
* **This app allows**: Reading, creating, editing and deleting content
* **App icon**: Choose accordingly

When you’re ready, publish the app to “Modify the app in Icenium”. If you feel so inclined the app will work with Sitefinity Box, as well.

Unless you are going to demo this on an actual mobile device, then there aren’t any other tasks that need to be done for the mobile app piece. If you do want to demo this on a mobile device, then you will need to setup your provisions with iOS and Android and then build and publish to your device.

There are certain advantages of publishing to SF Box or Icenium. SF Box seems to be more stable, but has less functionality. Icenium has more errors, but has more functionality… I suggest publishing to both and running through the functionality to see what works, what throws errors and what to avoid all together. If for some reason you can’t publish to Icenium, then just use SF Box. It can do most of everything.

*~ see figure 8*

## Custom APIs & Widgets

The file should already be added to your project, but in case they’re not, then the next thing you will want to do is extract the zip files and copy the zip’s “Custom” folder and paste it into the root of your SF project. **If you already have a folder called “Custom”** within the root of your project**,** then just copy/paste the folders and files within the zip’s “Custom” folder instead. Once you have done this, then you will rebuild your project.

### Widgets

There are two custom Thunder built widgets with designers that have been created for this module. They are:

* FootprintsReporter
  + This widget is responsible for pulling all of the Footprints items by Role via its API. The users in the role are added to the widget’s tab and the Footprint items are added to the grid. Included files:
    - FootprintsReporter.ascx
    - FootprintsReporter.cs
    - FootprintsReporterDesigner.ascx
    - FootprintsReporterDesigner.cs
    - FootprintsReporterDesigner.js
* FootprintsMapLauncher
  + This widget is responsible for displaying Footprint items as map points on a Google map. The points on the map are retrieved via the Web API described below and based off of the User(s) selected. It can also be filtered by a specified time range. The default time range is 3 months. The user defaults to the user selected on the Footprint Reporter page, but can be changed by selecting a new user(s) in its dropdown. Included files:
    - FootprintsMapLauncher.ascx
    - FootprintsMapLauncher.cs
    - FootprintsMapLauncher Designer.ascx
    - FootprintsMapLauncher Designer.cs
    - FootprintsMapLauncher Designer.js

### Web API

The Footprint Web API code is used to pull data for the FootprintsMapLauncher. There is only 1 Get method, but it is overloaded 3 times. The API will pull data based on the following filters:

* User
* From Date – if specifiied
* To Date – if specified

The overloaded Get method will pull data off of the following filters:

* Get All
* Get by User
* Get by User and Dates

**Please note** that you will always have a designated user, whether it is a single user or all users.

There are 2 files in the zip’s “Services” folder:

* FootprintsController.cs
* FootprintsModel.cs

In order for the API to work you must have the following routing information (highlighted) in your Global.asax. Should already be there, but here’s the code just in case.

*Add the following using statements:*

using System.Web.Routing;

using System.Web.Http;

*Add the following code:*

protected void Bootstrapper\_Initialized(object sender, Telerik.Sitefinity.Data.ExecutedEventArgs args)

{

ObjectFactory.Container.RegisterType(

typeof(ICriterionEvaluator),

typeof(CustomEvaluator),

PersonalizationConstants.CriteriaName.SearchKeywords,

new ContainerControlledLifetimeManager(),

new InjectionConstructor());

if (args.CommandName == "Bootstrapped")

{

RegisterRoutes(RouteTable.Routes);

}

}

public static void RegisterRoutes(RouteCollection routes)

{

routes.Ignore("{resource}.axd/{\*pathInfo}");

routes.MapHttpRoute(

name: "DefaultApi",

routeTemplate: "api/{controller}/{action}/{id}/{dateFrom}/{dateTo}",

defaults: new { id = RouteParameter.Optional, dateFrom = RouteParameter.Optional, dateTo = RouteParameter.Optional }

);

}

### Drag and Drop your Widgets

If everything is as described thus far in the document, then you should not have to do anything else. If you, however, move things around or change names and what not, then you may need to go in and tinker a bit with the code. If the widgets are missing you may need to register them using Thunder.

If you do need to register the widgets, then rebuild the project, go into your SF website and drag and drop the following widgets on to your previously created pages:

* Footprints Reporter 🡪 Footprints Reporter widget
* Footprint Details 🡪 Custom Footprints Module
* Footprints Map View 🡪 Footprints Map Launcher widget

## Using & Demoing the App

Now that you have the app installed, the project rebuilt and the pages created along with their respective widgets, then you are ready to start using it. The app is for the most part pretty self-explanatory, but here are some basic guidelines.

There should be data, but if not, then the first thing you will want to do is create some items for the Footprints content type. You can do this by either adding it directly to the SF backend 🡪 Content 🡪 Footprints content item, or via the Icenium simulator and/or mobile device with the mobile app installed on it. I think about 10 items using 3 to 5 people should be good for demo purposes.

Once that’s done, then you can demo the mobile piece using the Icenium simulator of your choice. This part is pretty straight forward as well. Basically a user is adding some metadata to the app and then submitting it to the SF dynamic content module. **Please note** that you will not be able to take an image with the simulator, but you can select one from your computer and/or SF library(s). No image will be saved in Sitefinity, though.

From there you can click on the FootprintsReporter to view the list of content items. This page will list the items by either “all users” or by “individual users” via a RadTabStrip and Grid. Users will be listed in the tabs and the Footprint items will be in the grid. All columns and tabs allow for sorting. A button underneath the grid labeled “Export…” will export the data to a csv file.

If you click on one of the items in the Summary column you will be taken to the FootprintDetails page, which lists out all the info as entered into the mobile app/simulator.

If you click on an item in the “Launch Map” column of the FootprintsReporter you will be taken to the FootprintMapLauncher page. **Please note** that whatever user you click on, as denoted by either the tab text or the grid’s Name column, is the user-data that you will see in the map page.

The Map Launcher page is used to give a visual representation of either all users (Role) or individual users (name) that have previously added Footprint items. For each Footprint item there will be a Google marker on the map, as long as it falls within the following filters:

* Name - required
* Start Date – if specified
* End Date – if specified

**Please note** that you do not have to use dates for your search, but if you do want to use dates you have to use both Start and End dates. An alert will popup if you forget. You will always have a designated user Name as a filter. If the Name dropdown is set to “Select” you will get an alert telling you to select a user before submitting.

The Google map will have all of the standard map functionality that we’ve all come to know and love. You can drill down and up as you’d like. You can switch it to a Satellite or Map view. You can drag and reposition the map as you will and you can click on each red marker for details about that “stop”.

At this time I do have the map defaulted to centering on the continental USA. If you were to add a spot in another country you will have to reposition the map to view it.

*“Todo”: Figure out a new default map center mechanism.*

## Gotchas

### Users & Roles

There are two main items that you need to setup correctly for this app to work at this time.

* Creating a role called “Footprints”
* Adding user’s to this role whose names will match what they put into the Mobile app.

For whatever reason the dynamic content api does not return a system author based off of the logged in user credentials. Since I cannot filter off of that author name coming from the system I have to depend on the Role 🡪 User 🡪 User-Name matching the user’s input in the Mobile App 🡪 Name field. This is obviously not optimal…

### Demoing on Mobile Device

If you decide to demo this app via an actual mobile device you will need a website that is accessible from the world-wide-web. You will also need to generate a device certificates and/or provisions for your mobile devices.

# Images

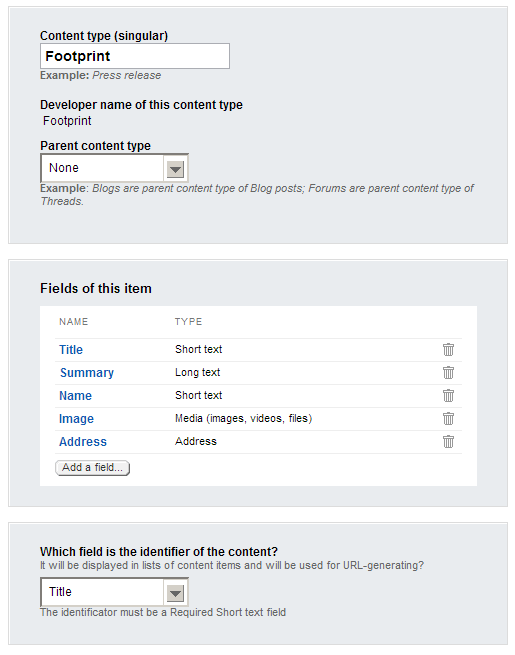


Figure 1

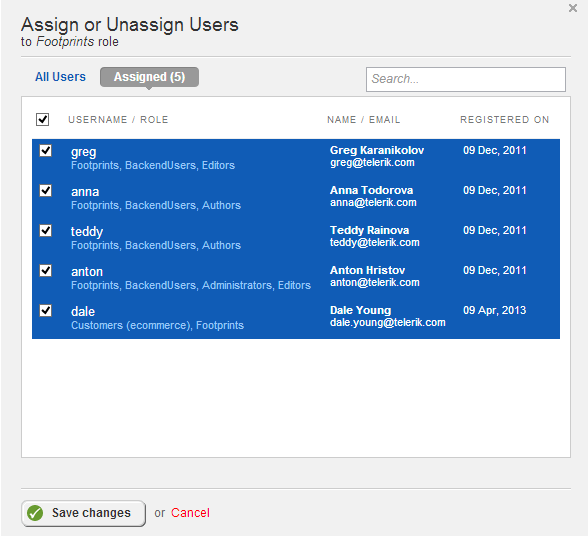


Figure 2

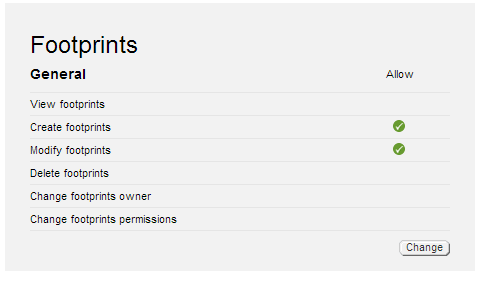


Figure 3

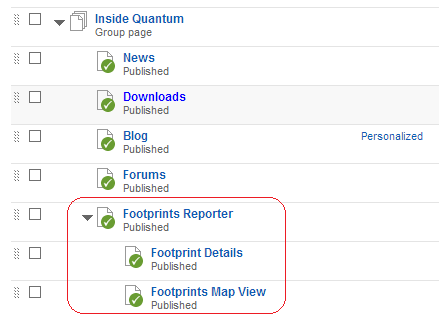


Figure 4 - Footprints Page Setup

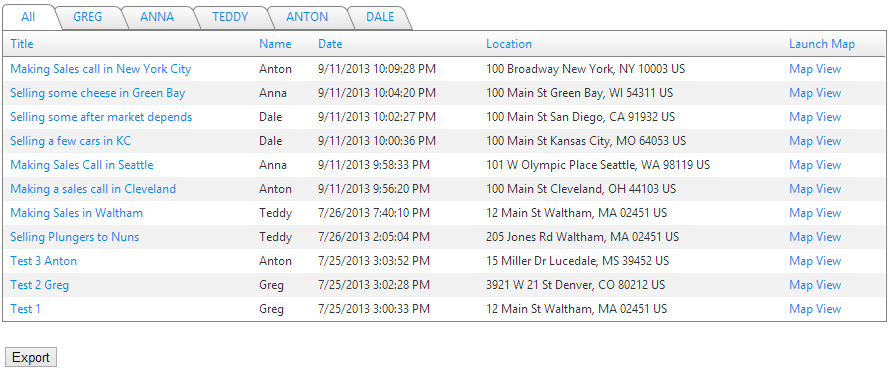


Figure 5 - Footprints Reporter Page

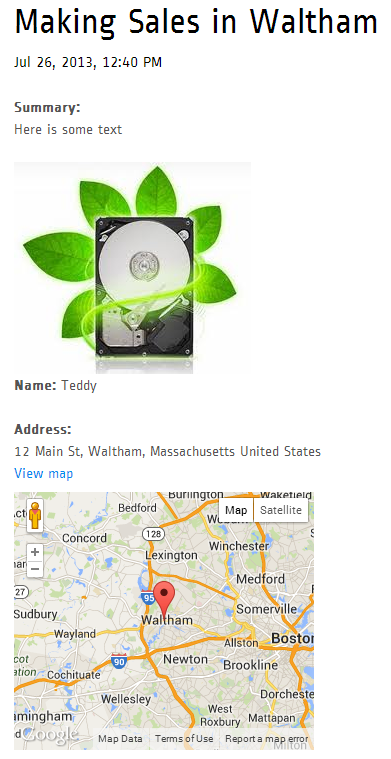


Figure 6- Footprint Details Page

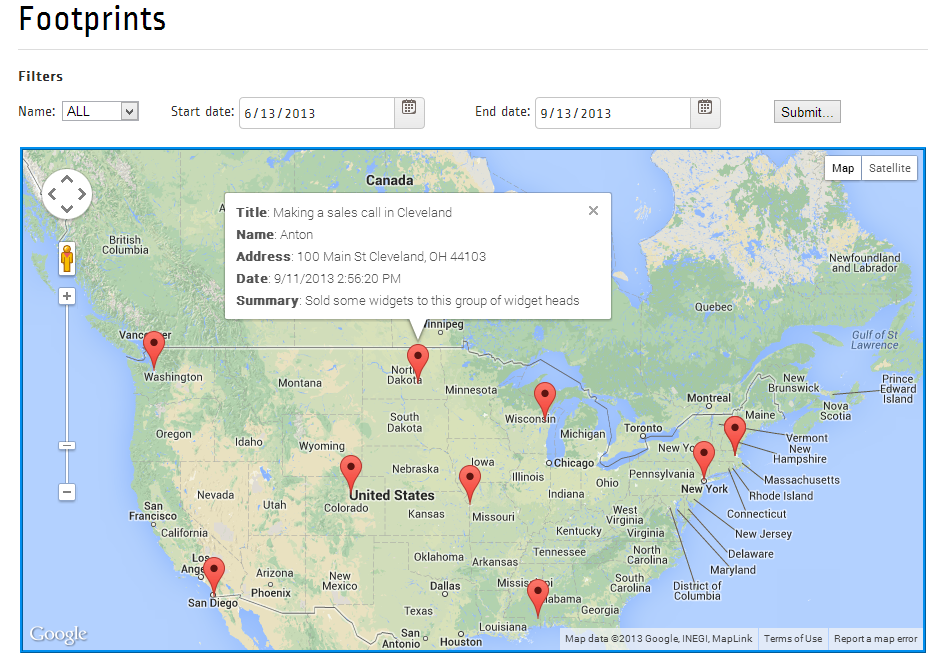


Figure 7 - Footprints Map Launcher Page

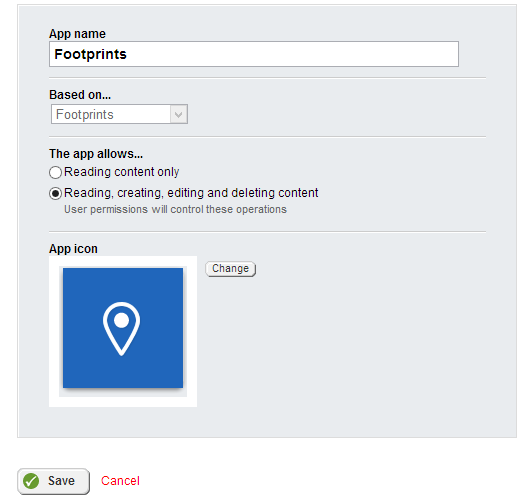


Figure 8



Figure 9- Global.asax Updates