The Return:

WikiNinjas Guide to SharePoint 2013

About the eBook

This eBook is provided "as is". The information and views expressed in this eBook, including URL and other web site references, may change without notice. You assume the entire risk of use.

This eBook does not provide you with legal rights to the ownership of a Microsoft product, but just the use, unless this is explicitly stated in the document. "Trial" keys are provided for a single purpose of test.

For more information, please contact **Gokan Ozcifci** or **Ed Price** at the following address ozcifci.gokan@live.be and edprice@microsoft.com

What is TechNet WIKI?

The TechNet Wiki is a library of information about Microsoft technologies, written by the community for the community. Whether you write code, manage servers, keep mission-critical sites up and running, or just enjoy digging into details, we think you will be at home in the TechNet Wiki.

- This is a community site. For official documentation, see MSDN Library, TechNet Library or contact Microsoft Support.
- The Wiki is focused on Microsoft technologies. The community will edit or remove topics that get too far off track.
- We are inspired by Wikipedia.
- Anyone who joins can participate and contribute content.

How Can I Participate?

The simplest way to participate is to use the information in this Wiki. The community is providing how-to guides, troubleshooting tips and techniques, practical usage scenarios, scripting pointers as well as overview, conceptual and technology overview topics.

- Read the terms of use.
- Sign in, upload an avatar and configure your profile.
- Review the Code of Conduct. It takes after the Ubuntu Code of Conduct and guides our behavior.
- Visit Getting Started and What Makes a Great Article to get the basics.
- Find topics using search, the tag cloud or by visiting the article spotlight page.
- Create a topic. Contribute boldly, edit gently!

We welcome your feedback. Head over to the TechNet Wiki Discussion forum, connect with us on the Wiki, or Tweet feedback using #TNWiki (and follow WikiNinjas).

Help us write the future.

Our Lead - Ed Price

You can find below the interview with the one and only Mr. Wiki Ed Price:

First question… I think for a lot of the readers (including me), you are mister Wiki. You have been there since, what I call, the First Light article (http://social.technet.microsoft.com/wiki/contents/articles/695.wiki-about-technet-wiki-en-us.aspx), and your activity levels are beyond-normal. Who is your Wiki "mr. Miyagi"? And outside of Wiki, who is your example?

That's interesting. What is the first article on TechNet Wiki? They're numbered you know. I'll hunt a little… 275… 115… 114… That's the lowest I can find for now.
Eric Battalio had the vision and the stubbornness to get this going. Tony Soper, Monica Rush, and Kim Ditto-Ehlert were all vital to getting the wiki ball rolling. And now we have important contributors that are both in Microsoft, like Tom Shinder, Nathaniel Scharer, Kurt Hudson, and Roger Doherty, and out of Microsoft, like Fernando Veltem, Patris, Luciano Lima, Luigi Bruno, Richard Mueller, Thiago Luiz, you, and Susan Bradley.
But if I was going to name one person who inspires me the most, my Mr. Miyagi, it would be the shirtless man… Yuri Diogenes.
Outside of Wiki… I'm inspired by Benjamin Franklin. If he could write the wildly popular Silence Dogood letters when he was 16, then is anything I write or edit all that impressive?

You're a SQL Server Experience Program Manager at Microsoft. Why did you start working for Microsoft and what does an Experience Program Manager do?
To be honest, I was starting a family, so I needed to think of working for a larger company. Microsoft takes care of its employees, so I naturally looked here.
I started working here back in 2005, focusing on assistance design and content for Microsoft Surface, our touch computer (we just announced a line of pretty sweet tablets). I've got five patents filed for Surface (4 pending). I later worked on Hardware (mice, keyboards, webcams) where I got to redesign our manual (working with our designer, Azy), removing the text and making it more of an IKEA or Lego like instruction booklet (pictures and arrows). Then I moved to SQL to work in the content team. I had a fun time of Wiki work, videos, redesigning Help layouts, and driving efforts to integrate more assistance in the UI.

Then I moved over closer to our UX team to be an xPM.
Experience program managers (at least in our group) focus on end-to-end experience envisioning, working with our Designers, Product Planners, and product PMs to help build out the experiences and scenarios and help make sure the customer is at the center of it all (here's an example of what focusing on customers feels like in a design). We also do a lot cross-team collaboration building, communication, and we sometimes own other Design-focused programs.

Personally, I own our personas program, I'm trying to help redefine and redesign the future of Help, I'm working to put a stronger focus on our customers, and I'm really driving toward some extreme team collaboration. I also sometimes make fun
videos like this one I made with Ehren (that's my voice as the stick figure).

Your TN Stats are insane: a total of 77,855 points, 1000s of forum replies, you've received more than 300 4 star ratings for your blog posts, 20,000+ Wiki activities… How do you fit this into your normal working schedule/life?
I broke 80K points. Woot.
I use clones. A whole army of them. I dress them in white armor and give them blasters.
Some people play videogames. Community is my videogame. =^)
Hey you didn't mention my achievement awards. I have the most of those in the whole world. I've got 17 gold ones.

There's one thing I've noticed about your TN Stats: no translations at all! If you had to learn a foreign language to get this number up, which one would it be?
I actually translate Spanish articles for TechNet Wiki. I lead a team of folks who help me refine the translations. Here's one: Wiki: Acerca de TechNet Wiki (es-ES)
That other stat on the profiles (Translation Wiki) is for translating on MSDN/TechNet Library, using a Translation Widget that's similar to what we have on TechNet Wiki and blogs. So on TechNet Wiki, you can translate an English article, and then similarly any edits you make go out to a moderator to double check them. They call the Library version the Translation Wiki, which is a little confusing because we also translate articles on TechNet Wiki, and then we have the Translation Widget with the same wiki-like features on the Wiki and blogs. So there are three different types of "translation wiki".

I believe you are married and have kids too. If so, does Wiki mean anything to them or is this just a "weird hobby" of daddy?
They're too young to really know. But my one year old is involved. I sit her on my lap while I write or edit sometimes. I put two stickers on my shirt, she takes them off and puts them on hers. Then I take them off and put them somewhere else on my shirt or hers. We go on like that for hours. Days even.
For my wife, the interaction is more like, "Are you working?" "Sort of." Then she gives me the look. I can see it even when I'm not looking at her. You can always see the look.

You often sign a blog post as "Ninja Ed". Now, to find a proper definition, I've looked it up and a ninja (or shinobi) was a mercenary in feudal Japan specializing in unorthodox warfare, including espionage, assassination, and open combat. Should we be afraid of you?
Yes. Be afraid of my wiki editing skills. You know how they say the pen is mightier than the sword? Well that was before they invented the keyboard.

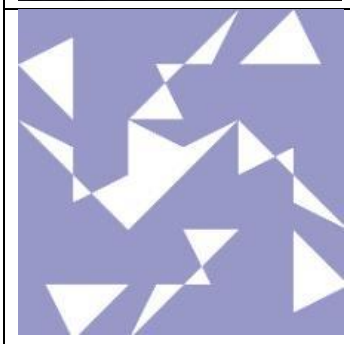
I'm very fond of the Ninja stick figure, I think it's hilarious. Where did this idea come from?
Eric Battalio, the grand master of TechNet Wiki. I think he's a fan of stick figure online comic strips and stick fighter animations… either way he likes the simplicity of it. He started out making a ninja stick figure icon for the Twitter account. Then he made some stick figure images for stickers to promote TechNet Wiki. Yuri followed with the Brazil Wiki Ninjas Twitter account and a ninja with the Brazil flag in the corner. Other Brazil members also made stick figures. I brought the concept of the Wiki Ninjas name and stick figures over to the blog.
I got the collection of ninja images from Eric, and I began adding to it, like the image above.

If it was possible to get a present from the TN Wiki community on your birthday, what would you like to have?
A medallion that grants peace to everyone you hit it with. Or… A Wiki Ninja stick figure image of me… tall, beard, glasses, and wearing a nametag that says "EDitor".

Any famous last words?
In the famous words of Winston Churchill, "Madam, you are ugly. In the morning, I shall be sober."
In the famous words of Eleanor Roosevelt or someone else, "Great minds discuss ideas; average minds discuss events; small minds discuss people."
In the famous words of Benjamin Franklin, "Necessity never made a good bargain."

Guest authors

	**Joe Davies**	Microsoft	Principal Writer at Microsoft, currently working on the Office Solutions writing team.
	**Matthew Yarlett**	Microsoft Partner	I'm a Senior Consultant with OBS, focused on delivering business solutions built on the SharePoint platform. Working with business requirements and drivers, I help design solutions that deliver clear business benefit, and fit in with existing IT architectures.
	**Thuan Soldier**	MVP Microsoft Most Valuable Professional	A 23-year-old man loving Microsoft technologies and making crazy ideas on business journey.

	**Craig Lussier**	Microsoft Partner	SharePoint Architect and Information Management Consultant / Bermuda SPUG Founder and Leader / TechNet Wiki Community Council Member / Father
	**Benoit Jester**	MVP Microsoft Most Valuable Professional	Admin / Expert SharePoint Freelance Blog technique : http://spasipe.wordpress.com contact@asipe.net
	**Margriet Bruggeman**	MVP Microsoft Most Valuable Professional	Margriet is a technical architect and software developer specializing in Microsoft technology. Margriet has worked with SharePoint since the beta release of the first version (SharePoint Portal Server 2001) and was one of the first people world-wide to receive the SharePoint MVP title (in 2002-2003). Margriet has written several books about SharePoint
	**Inderjeet Singh Jaggi**	Microsoft Partner	I am working on SharePoint technology since last 8 yrs. Worked on nearly all the version of Sharepoint from sts 2001 to SP 2013.

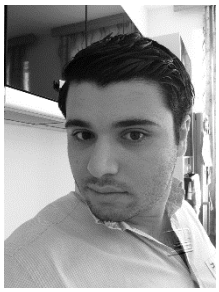
	**RaghuAriga**	Microsoft Partner	I like playing with latest gadgets. Love designing websites and creating communities for sharing ideas. Enjoy listening music and watching football. Apart from them i work on SharePoint, Love it.
	Brent Groom	Microsoft	
	**Dan Christian**		
	**Aulakh Amardeep- MVP SharePoint Server**	MVP Microsoft Most Valuable Professional	I am a working professional as a Project Lead in Collaboration Technologies which includes SharePoint Server, Office Communicator Server with Wipro LTD. My experience back ground started with the System Administration specially Windows Servers Administration and Network Management for CISCO devices. Now I am dedicatedly working on SharePoint technologies and had severed clients from different domain such as Microsoft SharePoint Product Support Services, Pharmacy, Semiconductor Manufacturing, Banking and helped them designing, defining, configuring and deploying the new processes using SharePoint best practices. I am very fond of enhancing my learning skill in every subject concerned to

			above listed technologies as I believe learning has no end point and I always tend to be a newbie for SharePoint.
	**Rashu,Rahul**		I am in Microsoft Technologies from over 3 years. Other than sharepoint 2007 I have experience in SSIS also. I am currently involved in a huge Sharepoint 2007 project. I do work in SPS 2010 as well
	**Melick**		A person started programming in 2002 by engaging freelance developments in VB6. Skilled at .NET technologies, Database technologies as well SharePoint by completing two degrees in information technology with high performance. Involved in developing, designing, consulting and teaching in Microsoft related technologies. Never hesitate to share knowledge among the community and resulting to become a MVP – SharePoint in Leadership Technology.
	**Jason Barkes**		Software Architect and Developer currently focused on SharePoint, InfoPath, ASP.NET and Android.
	**Steven Andrews**		SharePoint Professional

	**Jesper Arnecke**	Microsoft Partner	Worked with the following MS technologies more than once :) Hyper-V, SCVMM 2008, SQL 2000/2005/2008/2012, Failover Clustering 2008, NLB 2003/2008, Win O/S 2000/2003/2008/2008R2/2012, SharePoint 2003/2007/2010/2013, Project 2010, IIS 6.0/7.0, CRM 4.0/2011, Exchange 2003/2010, TMG 2010 TFS 2010 UAG 2010
	**Nikolas Charlebois-Laprade**		Software Engineer, Speaker, Author, Geek Microsoft Certified Professional http://NikCharlebois.com

Principal Author

Gokan Ozcifci



Gokan, working for **Neoxy Consultancy** is one of the few people in Belgium to hold the prestigious **Microsoft Most valuable Professional** Award and is one of the retired **MCC** badge holder.

Leading the Turkish community on the TechNet Wiki International Council, including leading the Turkish blog, Turkish forum (for Wiki discussions), and progress for the TAT team to make in content on TechNet Wiki. Gokan is also an MVP and is known for his SharePoint blogs, Gallery scripts, representing SharePoint in the TechNet Wiki Advisory Board, and his work on the TechNet Wiki Community Council, focusing on Community Evangelism (through TechNet Wiki White Papers and TechNet Wiki TV) and TechNet Wiki Featured Articles.

Gokan is blogging on SharePoint since 2011 at **http://gokanx.wordpress.com** and you can follow him on Twitter – **@gokanozcifci**.

Table of Contents

SharePoint 2013 Server

SharePoint Server 2013 is designed to help you achieve new levels of reliability and performance, delivering features and capabilities that simplify administration, protect communications and information, and empower users while meeting their demands for greater business mobility.

The RTM version of SharePoint 2013 was announced on October 11, 2012. (Historical Note: The SharePoint 2013 Preview was released on July 16, 2012.)

Discontinued Features and Modified Functionality

- See: Discontinued features and modified functionality in Microsoft SharePoint 2013

Downloads

- SharePoint 2013 Download Portal
  This Wiki page contains links to RTM downloads for SharePoint 2013, related Servers, Office/Developer Applications, Language Packs and Developer SDKs
- Microsoft Download Center: "SharePoint 2013" Search Results

Forums (MSDN/TechNet)

- SharePoint 2013 Preview for Developers (MSDN)
- Developing Apps for SharePoint (MSDN)
- SharePoint 2013 - General Discussions and Questions (MSDN)
- SharePoint 2013 - Development and Programming (MSDN)
- SharePoint 2013 - Using SharePoint Designer, InfoPath and Other Customizations (MSDN)
- SharePoint 2013 - Search (MSDN)
- SharePoint 2013 - Setup, Upgrade, Administration and Operations (MSDN)
- SharePoint 2013 - General Discussions and Questions (TechNet)
- SharePoint 2013 - Setup, Upgrade, Administration and Operations (TechNet)
- SharePoint 2013 - Search (TechNet)

Best Practices

- SharePoint 2013 Best Practices

Install/Configuration Procedures

- SharePoint 2013 Best Practices - Installation
- SharePoint 2013: Install SharePoint 2013 with SQL 2012 on Windows Server 2012
- Install and Configure SharePoint Designer 2013 on Windows Server 2012
- Download/Install SharePoint 2013 Prerequisites on Windows Server 2012 with PowerShell

Troubleshooting SharePoint 2013

- Troubleshoot

PowerShell

- SharePoint 2013 PowerShell Reference and Resources
- Windows PowerShell for SharePoint 2013 Learning Roadmap

Resources for Developers

SharePoint 2013 - Resources for Developers **-** this Wiki page includes links to the following topics:

- What's New
- API's, SDK's and Reference
- Apps
- Build Numbers
- PowerShell
- Service Applications
- Training
- Workflow

SharePoint 2013 new features and capabilities Dev Center on MSDN

Resources for IT Pros

SharePoint 2013 - Resources for IT Pros - this Wiki page includes links to the following topics:

- Build Numbers
- PowerShell
- Reference
- Service Applications
- Training
- Upgrade
- Office Blogs

Scenario Pages

- eDiscovery in SharePoint Server 2013 and Exchange Server 2013
- Personal sites (My Sites) in SharePoint Server 2013
- Create SharePoint sites by using cross-site publishing in SharePoint Server 2013

SharePoint in the Cloud

- SharePoint 2013 in the Cloud - Office 365
- SharePoint 2013 - SkyDrive Pro

SharePoint Hybrid Configuration

- Hybrid for SharePoint Server 2013
- Overview of Hybrid for SharePoint Server 2013 and Office 365 video (4 minutes)

WCF Service Call

- WCF REST Service: Integration of HTTP/ HTTPs WCF Services (REST & SOAP) in SharePoint 2013

Scenario Pages

- eDiscovery in SharePoint Server 2013 and Exchange Server 2013

System Center Operations Manager Monitoring Packs

- System Center Monitoring Pack for SharePoint Server 2013
- System Center Monitoring Pack for SharePoint Foundation 2013

Service Applications

- SharePoint 2013 - Service Applications
  - Access Services
  - Access Services 2010
  - App Management Service
  - Business Data Connectivity Service
  - Excel Services Application
  - Machine Translation Service
  - PerformancePoint Service Application
  - Managed Metadata Service Application
  - Search Service Application
  - Secure Store Service
  - User Profile Service Application
  - Visio graphics Service
  - Word Automation Services
  - Work Management Service Application

System Requirements

- SharePoint 2013 - System Requirements (Hardware and Software)

Test Lab Guides

- SharePoint Server 2013 Test Lab
- SharePoint Server 2013 Business Intelligence Test Lab

Learning Roadmaps

- Authentication in SharePoint 2013
- Database Management for SharePoint 2013
- Permissions for SharePoint 2013
- Upgrade for SharePoint 2013
- User Profiles in SharePoint 2013
- Virtualize SharePoint 2013
- Windows PowerShell for SharePoint 2013

SharePoint 2013 Licensing

- SharePoint 2013 Volume Licensing brief
- Licensing Internet Sites Built on SharePoint 2013

Wiki Article Tag Filtering

- SharePoint 2013 - How to Find and Filter Wiki Articles by Language (en-US)

Stub Articles

As the SharePoint 2013 Preview was released on July 16th, 2012, it will take time to properly represent the SharePoint 2013 platform on the Wiki. Some stub articles have been created for major features/aspects of the platform to get the article creation process started. Please help out by adding relevant content to the SharePoint 2013 stub pages.

- Listing of SharePoint 2013 related stub articles

Notes on 'Stub' pages:

- If you are going to create a stub article related to SharePoint 2013, please add "SharePoint 2013" and "stub" as tags to the article. This way, others can easily find the SharePoint 2013 related stub articles using the link above.
- If you add content to a stub page, please remove the "stub" tag so it no longer appears in the SharePoint 2013 related stub articles list.
- Read - Wiki: How to Create a Stub Page

SharePoint: Site Collection cannot be restored due to Insufficient Space

Problem Definition

We get error "Site collection could not be restored. Make sure the content databases are available and have sufficient space" when try to restore site collection from a backup

Steps to recreate the issue

I observed this issue in below scenarios:

Scenario 1

- A user takes a backup of a site collection.
- Creates a new blank site collection in same web Application (content database).
- Tried to restore it the backup to this site collection

Scenario 2

- A user takes a backup of a site collection and then deletes site collection from central admin.
- Creates a new blank site collection in same web Application (content database).
- Tried to restore it the backup to this site collection

Cause

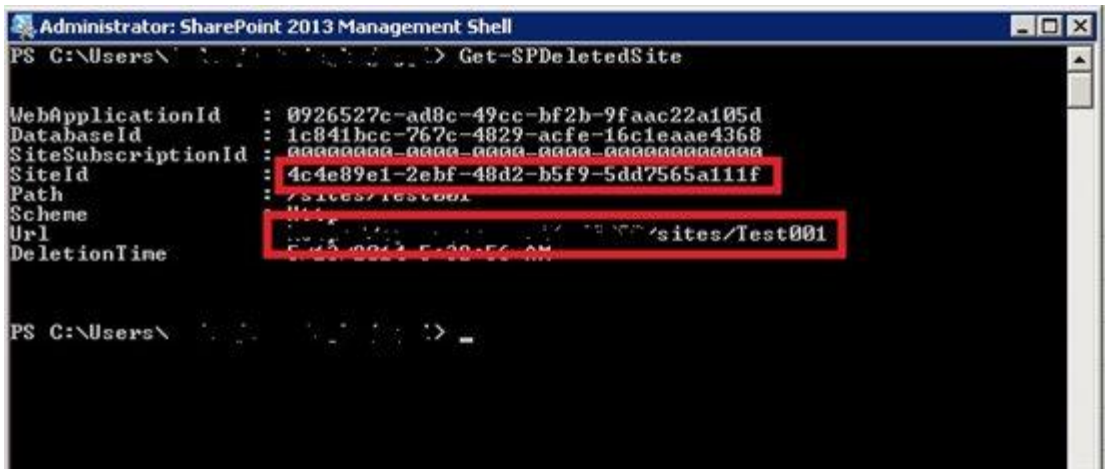
There are 2 causes to this issue:

1. Site is marked to be restored in future using Restore-SPDeletedSite command. When a site collection is deleted. It can be restored with same Site ID and content without any backup. You just need to restore it using Restore-SPDeletedSite and delete sites SiteID
2. Timerjob : Gradual Site Delete. "*When a site collection is deleted, the site collection entry (pointer) in dbo.SiteMap is removed from the configuration database and from dbo.Sites in the corresponding content database. For all purposes of user access to the site collection Url or its content the site collection no longer exists and is inaccessible, or otherwise, the Url is no longer reserved. The site collection deletion is then queued into a new table (dbo.SiteDeletion) in the hosting content database where it is marked to be gradually deleted. At this point in the operation, a new Timer Job Definition [Gradual Site Delete] executes on a daily schedule [configurable], and will continuously attempt to delete all the data for all the site collections in its queue (dbo.SiteDeletion). It will delete the data in small enough batches of a maximum of 1000 rows through multiple transactions to avoid lock escalation, and can be resumed in the event any failure occurs so that it can attempt the delete process again if needed. Once the site collection is fully deleted the dbo.SiteDeletion entry is removed.*" [Bill Baer, MS]

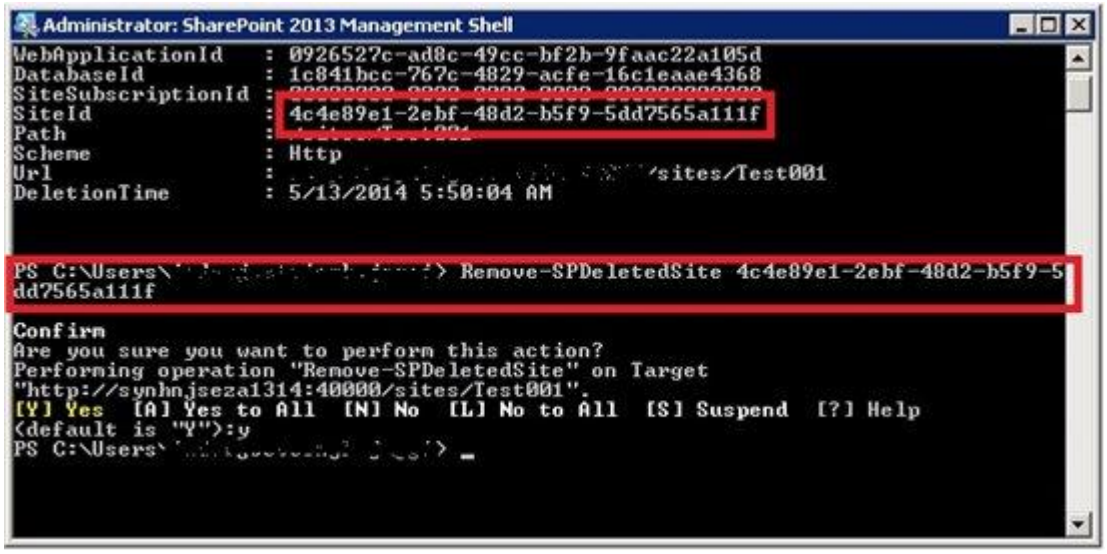
Resolutions

1. Create a new database from Central Admin.
    - Go to Central Admin > Application Management
    - Manage Content databases.
    - Add a content database.
    - Add a new content database name and click OK.
      Note: Make sure that the new blank site collection is created in new content database. Now you will be able to restore site collection in this new database.
2. Use Remove-spdeletedsite.

    - Open Powershell and run get-spdeletedsite. You will see your site collection here with Site ID



- Now delete the site collection using this site ID “remove-spdeletedsite f5f7639d-536f-4f76-8f94-57834d177a99”



- Now you should be able to restore site collection in same database.

Mitigation

Wait for some time while the timerjob "Gradual Site Delete" delete all reference from all databases and table before site collection is restored.

References

- http://social.technet.microsoft.com/Forums/sharepoint/en-US/548a6966-8d8b-4b6c-8b82-485d0d4d634d/site-collection-could-not-be-restored-make-sure-the-content-databases-are-available-and-have?forum=sharepointgeneralprevious#7a3e5fd4-670a-4cd3-9b14-87d483feb4d4
- http://blogs.technet.com/b/wbaer/archive/2010/08/02/gradual-site-delete-in-sharepoint-2010.aspx

SharePoint 2013: Use a List Template Model with the REST API

Introduction

This article describes how to use the List Template model with the REST API. The article's aim is to save you considerable time and effort when building a solution that uses the List Template model.

Model

```
TemplateModel = {
    element: '',
    listname: '',
    url: '',
    addurl: '',
    editurl: '',
    displayurl: '',
    filtervalue: '',
    filteronly: '&$filter=',
    selectonly: '&$select=',
    orderby: '',
    orderbyonly: '&$orderby=',
    toponly: '&$top=',
    topandskip: '',
    topcount: 7,
    skipcount: 0,
    typeofdata: '',
    result_set: undefined,
    callbackhandler: undefined,
    pageelement: '',
    listpageindex: new Array,
    resettotalcount: false,
    totalcount: 0,
    init: function (pelement, cbhandler, pcountelement) {
        TemplateModel.element = pelement;
```

```
        TemplateModel.callbackhandler = cbhandler;
        TemplateModel.pageelement = pcountelement;
        TemplateModel.resettopandskip();
        TemplateModel.resetfiltervalue();
    },
    load: function () {
        TemplateModel.url =
TemplateModel.listpageindex[TemplateModel.skipcount];
        TemplateModel.getdatafromlist();
    },
    additem: function (dialogResult, returnValue) {
        if (dialogResult == SP.UI.DialogResult.OK) {
            TemplateModel.url = appweburl + svc_apiurl +
svc_sitecollurl +"lists/getByTitle('" + TemplateModel.listname
+ "')/items?@target='" + hostweburl + "'";
            var rqheaders = {
                "content-
type": "application/json;odata=verbose",
                "X-RequestDigest": $("#__REQUESTDIGEST").val()
            }
            var setdata = {
                __metadata: { "type": TemplateModel.typeofdata
},
                Title: returnValue.Title
            };
            var senddata = JSON.stringify(setdata);
            TemplateModel.createorupdateitem(senddata,
rqheaders);
        }
    },
    getbyId: function (id, phandler) {
        TemplateModel.url = appweburl + svc_apiurl +
svc_sitecollurl + "lists/getByTitle('"+ TemplateModel.listname
+ "')/items(" + id + ")?@target='" + hostweburl + "'";
        var executor = new SP.RequestExecutor(appweburl);
        executor.executeAsync(
            {
                url: TemplateModel.url,
                method: "GET",
                headers: { "Accept": "application/json;
odata=verbose" },
                success: phandler,
                error: TemplateModel.errorHandler
            }
        );
    },
```

```
    updatebyId: function (dialogResult, returnValue) {
        if (dialogResult == SP.UI.DialogResult.OK) {
            var getetag = returnValue.etag;
            TemplateModel.url = appweburl + svc_apiurl +
svc_sitecollurl +"lists/getByTitle('" + TemplateModel.listname
+ "')/items(" + returnValue.Id + ")?@target='" + hostweburl
+ "'";
            var rqheaders = {
                "content-
type": "application/json;odata=verbose",
                "Accept": "application/json; odata=verbose",
                "X-RequestDigest": $("#__REQUESTDIGEST").val(),
                "X-HTTP-Method": "MERGE",
                "If-Match": getetag
            }
            var setdata = {
                __metadata: { "type": TemplateModel.typeofdata
},
                Title: returnValue.Title
                };
            var senddata = JSON.stringify(setdata);
            TemplateModel.createorupdateitem(dataString,
rqheaders);
        }
    },
    deletebyId: function (id) {
        TemplateModel.url = appweburl + svc_apiurl +
svc_sitecollurl + "lists/getByTitle('"+ TemplateModel.listname
+ "')/items(" + id + ")?@target='" + hostweburl + "'";
        var requestHeaders = {
            "Accept": "application/json; odata=verbose",
            "X-RequestDigest": $("#__REQUESTDIGEST").val(),
            "X-HTTP-Method": "DELETE",
            "If-Match": "*"
        };
        var executor = new SP.RequestExecutor(appweburl);
        executor.executeAsync(
            {
                url: TemplateModel.url,
                contentType: "application/json; odata=verbose",
                method: "POST",
                headers: requestHeaders,
                success: TemplateModel.reload,
                error: TemplateModel.errorHandler
            }
        );
```

```
    },
    getdatafromlist: function () {
        var executor = new SP.RequestExecutor(appweburl);
        executor.executeAsync(
            {
                url: TemplateModel.url,
                method: "GET",
                headers: { "Accept": "application/json;
odata=verbose" },
                success: TemplateModel.onSuccess,
                error: TemplateModel.errorHandler
            }
        );
    },
    createorupdateitem: function (data, rqheaders) {
        var executor = new SP.RequestExecutor(appweburl);
        executor.executeAsync(
            {
                url: TemplateModel.url,
                contentType: "application/json;odata=verbose",
                method: "POST",
                body: data,
                headers: rqheaders,
                success: TemplateModel.reload,
                error: TemplateModel.errorHandler
            }
        );
    },
    onSuccess: function (data) {
        var jsonObject = JSON.parse(data.body);
        if (TemplateModel.resettotalcount &&
TemplateModel.pageelement != null) {
            $(TemplateModel.pageelement.substring(0,
TemplateModel.pageelement.length - 1) + 'f').hide();
            $(TemplateModel.pageelement.substring(0,
TemplateModel.pageelement.length - 1) + 'b').hide();
            $(TemplateModel.pageelement).text('1');
            $(TemplateModel.pageelement.substring(0,
TemplateModel.pageelement.length - 1)).text('1');
        }

        if (jsonObject.d.results != undefined) {
            TemplateModel.result_set = jsonObject.d.results;
            if (jsonObject.d.__next != null) {
                TemplateModel.listpageindex[TemplateModel.skipco
unt + 1] = jsonObject.d.__next;
```

```
                if (TemplateModel.resettotalcount) {
                    var removPage =
TemplateModel.listpageindex[TemplateModel.skipcount];
                    removPage =
removPage.replace(TemplateModel.topandskip, "");
                    var executor
= new SP.RequestExecutor(appweburl);
                    executor.executeAsync(
                        {
                            url: removPage,
                            method: "GET",
                            headers:
{ "Accept": "application/json; odata=verbose" },
                            success:
TemplateModel.onFilterCountSuccess,
                            error:
TemplateModel.errorCountHandler
                        }
                    );
                    TemplateModel.resettotalcount = false;
                }
            }
        }
        else if (jsonObject.d != undefined) {
            TemplateModel.result_set = jsonObject.d;
        }
        TemplateModel.callbackhandler(TemplateModel);
    },
    errorHandler: function (data, errorCode, errorMessage) {
        alert("Unable to load model: " + errorMessage);
    },
    errorCountHandler: function (data, errorCode, errorMessage) {
        alert("Unable to get count for model: " + errorMessage);
    },
    onFilterCountSuccess: function (data) {
        var jsonObject = JSON.parse(data.body);
        if (jsonObject.d.results != undefined) {
            TemplateModel.totalcount =
jsonObject.d.results.length;
            var pagenum = Math.floor(TemplateModel.totalcount /
TemplateModel.topcount);
            if (TemplateModel.totalcount % TemplateModel.topcount
> 0) {
                pagenum += 1
            }
            if (pagenum > 1) {
```

```
                $(TemplateModel.pageelement.substring(0,
TemplateModel.pageelement.length - 1) + 'f').show();
            }
            else {
                $(TemplateModel.pageelement.substring(0,
TemplateModel.pageelement.length - 1) + 'f').hide();
            }
            $(TemplateModel.pageelement).text(pagenum == 0 ? 1 :
pagenum);
            $(TemplateModel.pageelement.substring(0,
TemplateModel.pageelement.length - 1)).text('1');
            $(TemplateModel.pageelement.substring(0,
TemplateModel.pageelement.length - 1) + 'b').hide();
        }
    },
    reload: function (data) {
        TemplateModel.resettopandskip();
        TemplateModel.resettotalcount = true;
        TemplateModel.load();
    },
    settopandskip: function (ptop) {
        TemplateModel.topandskip = TemplateModel.toponly +
(TemplateModel.topcount);
        TemplateModel.skipcount = ptop - 1;
    },
    seteqfilter: function (column, data) {
        TemplateModel.filtervalue = TemplateModel.filteronly +
column + " eq '" + data +"'";
        TemplateModel.skipcount = 0;
        TemplateModel.resettotalcount = true;
        TemplateModel.listpageindex[TemplateModel.skipcount] =
appweburl + svc_apiurl + svc_sitecollurl + "lists/getByTitle('" +
TemplateModel.listname + "')/items?@target='" + hostweburl
+ "'" + TemplateModel.orderby + TemplateModel.topandskip +
TemplateModel.filtervalue;
    },
    setorderby: function (column) {
        TemplateModel.orderby = TemplateModel.orderbyonly +
column;
        TemplateModel.skipcount = 0;
        TemplateModel.listpageindex[TemplateModel.skipcount] =
appweburl + svc_apiurl + svc_sitecollurl + "lists/getByTitle('" +
TemplateModel.listname + "')/items?@target='" + hostweburl
+ "'" + TemplateModel.orderby + TemplateModel.topandskip +
TemplateModel.filtervalue;
    },
```

```
    resetorderby: function (column) {
        TemplateModel.orderby = TemplateModel.orderbyonly
+ 'Title';
        TemplateModel.skipcount = 0;
        TemplateModel.listpageindex[TemplateModel.skipcount] =
appweburl + svc_apiurl + svc_sitecollurl + "lists/getByTitle('" +
TemplateModel.listname + "')/items?@target='" + hostweburl
+ "'" + TemplateModel.orderby + TemplateModel.topandskip +
TemplateModel.filtervalue;
    },
    resettopandskip: function () {
        TemplateModel.topandskip = TemplateModel.toponly +
TemplateModel.topcount;
        TemplateModel.skipcount = 0;
        TemplateModel.listpageindex[TemplateModel.skipcount] =
appweburl + svc_apiurl + svc_sitecollurl + "lists/getByTitle('" +
TemplateModel.listname + "')/items?@target='" + hostweburl
+ "'" + TemplateModel.orderby + TemplateModel.topandskip +
TemplateModel.filtervalue;
    },
    resetfiltervalue: function () {
        TemplateModel.filtervalue = '';
        TemplateModel.orderby = TemplateModel.orderbyonly
+ 'Title';
        TemplateModel.skipcount = 0;
        TemplateModel.resettotalcount = true;
        TemplateModel.listpageindex[TemplateModel.skipcount] =
appweburl + svc_apiurl + svc_sitecollurl + "lists/getByTitle('" +
TemplateModel.listname + "')/items?@target='" + hostweburl
+ "'" + TemplateModel.orderby + TemplateModel.topandskip +
TemplateModel.filtervalue;
    },
    displayaddform: function (pclientid) {
        var args1 = {
            command: 'add'
        };
        var options = {
            url: TemplateModel.addurl,
            showClose: true,
            title: 'Title',
            dialogReturnValueCallback: TemplateModel.additem,
            args: args1
        };
        SP.UI.ModalDialog.showModalDialog(options);
    },
    displayeditform: function (id) {
```

```
        TemplateModel.displayobjbyid(id, 'edit');
    },
    displayviewform: function (id) {
        TemplateModel.displayobjbyid(id, 'view');
    },
    displayobjbyid: function (id, pcommand) {
        var objtosend = null;
        if (TemplateModel.result_set != undefined) {
            objtosend =
$.grep(TemplateModel.result_set, function (item, i) {
                return item.Id == id;
            });
        }
        if (objtosend != null) {
            var args1 = {
                command: pcommand,
                sendobjinfo: objtosend
            };
            var options = {
                url: TemplateModel.editurl,
                showClose: true,
                title: 'Title',
                dialogReturnValueCallback:
TemplateModel.updatebyId,
                args: args1
            };
            SP.UI.ModalDialog.showModalDialog(options);
        }
    }
};
```

Setting the Page Elements

```
<a href="javascript:refreshModel(TemplateModel);">Refresh</a>
<a id="splistpageb" href="javascript:pagebackGrid(TemplateModel);
">Page Down</a>
<span id="splistpage">1</span> of <span id="splistpaget">1</span>
<a id="splistpagef" href="javascript:pageupGrid(TemplateModel);">
Page Up</a></li>
<div id="divlist"></div>
```

Accessing the Model

Before we can access the model, we need to set a few variables and add scripts to our page. If you have created a separate .js file for the model, then include the link.

```
<script
type="text/javascript" src="http://ajax.microsoft.com/ajax/jQuery
/jquery-1.8.1.min.js "></script>
<script
type="text/javascript" src="/_layouts/15/SP.UI.Dialog.js"></scrip
t>
<script
type="text/javascript" src="../Scripts/templatemodel.js"></script
>
<script
type="text/javascript" src="../Scripts/jsrender.min.js"></script>
var hostweburl;
var appweburl;
var scriptbase;
var svc_apiurl = "/_api/SP.AppContextSite(@target)/";
var svc_sitecollurl = "web/";  //"site/rootweb/";
var temp_list;
var sortUp = false;
$(function () {
    hostweburl =
decodeURIComponent(getQueryStringParameter("SPHostUrl"));
    appweburl =
decodeURIComponent(getQueryStringParameter("SPAppWebUrl"));
    temp_list = TemplateModel;
    temp_list.init("#divlist",
loadtemplatedataingrid, "#splistpaget");
    temp_list.load();
```

```
});

function getQueryStringParameter(paramToRetrieve) {
    var params = document.URL.split("?")[1].split("&");
    var strParams = "";
    for (var i = 0; i < params.length; i = i + 1) {
        var singleParam = params[i].split("=");
        if (singleParam[0] == paramToRetrieve)
            return singleParam[1];
    }
}

function loadtemplatedataingrid(model) {
    $(model.element).empty();
    var tableHeader = "<thead><tr><td width=\"40%\"><a
href=\"javascript:sortResults('Title');\">Title</a></td></thead>
";
    var ntable = $("<table>", { id: "customTable", width: "100%",
border: "0", cellspacing:"0",
cellpadding: "0" }).append(tableHeader);
    var templ = "<tr><td>{{>Title}}</td></tr>";
    $.templates({ "tmplTable": templ });
    ntable.append($.render.tmplTable(model.result_set));
    $(model.element).append(ntable);
}
```

Passing and Retrieving Items to Modal Pop-up

The model allows passing the item to another page and show it in a modal pop-up window. When accessing value on the page and passing value back from the page:

```
var getpassedargs;
var etagvalue = 0;
var itemid = 0;
$(function () {
    getpassedargs = window.frameElement.dialogArgs;
    if (getpassedargs.command == 'add') {
        $('#cmdedit').attr('disabled', true);
        $('#cmdedit').removeClass('popup-edit').addClass('popup-
edit-nohover');
    }
    else {
        if (getpassedargs.command == 'edit') {
            $('#cmdedit').attr('disabled', true);
```

```
            $('#cmdedit').removeClass('popup-
edit').addClass('popup-edit-nohover');
        }
        else{
            $('.unlock').attr('disabled', true);
        }
        var setobj = getpassedargs.sendobjinfo;
        etagvalue = setobj[0].__metadata.etag;
        itemid = setobj[0].Id;
        $('#textboxtitle').val(setobj[0].Title);
    }

    $('#cmdedit').click(function () {
        $('#cmdedit').attr('disabled', true);
        $('#cmdedit').removeClass('popup-edit').addClass('popup-
edit-nohover');
        $('.unlock').removeAttr('disabled');
    });

    document.onsubmit = function () {
        var result = SP.UI.DialogResult.OK;
        var value = {
            Id: itemid,
            Title: 'Title',
            etag: etagvalue
        };
        SP.UI.ModalDialog.commonModalDialogClose(result, value)
    };

    $('#cmdcancel').click(function () {
        var result = SP.UI.DialogResult.cancel;
        var value = '';
        SP.UI.ModalDialog.commonModalDialogClose(result, value)
    });
});

<input type="submit" id="cmdsave" value="Save" />
<input type="button" id="cmdcancel" value="Cancel"/>
```

Paging the Model

```
function pagebackGrid(model) {
    var pcurrent = $(model.pageelement.substring(0,
model.pageelement.length - 1)).text();
    var pmax = $(model.pageelement).text();
    pcurrent--;
```

```
    $(model.pageelement.substring(0, model.pageelement.length -
1)).text(pcurrent);
    model.settopandskip(pcurrent);
    model.load();
    if (pcurrent == 1) {
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'b').hide();
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'f').show();
    }
    else {
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'b').show();
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'f').show();
    }
}

function refreshModel(model) {
    model.resettopandskip();
    model.load();
    $(model.pageelement.substring(0, model.pageelement.length -
1)).text('1');
    $(model.pageelement.substring(0, model.pageelement.length -
1) + 'b').hide();
}

function pageupGrid(model) {
    var pcurrent = $(model.pageelement.substring(0,
model.pageelement.length - 1)).text();
    var pmax = $(model.pageelement).text();
    pcurrent++;
    $(model.pageelement.substring(0, model.pageelement.length -
1)).text(pcurrent);
    model.settopandskip(pcurrent);
    model.load();
    if (pcurrent == pmax) {
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'f').hide();
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'b').show();
    }
    else {
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'b').show();
```

```
        $(model.pageelement.substring(0,
model.pageelement.length - 1) + 'f').show();
    }
}
```

Sorting the Model

```
function sortResults(pcolumn) {
    if (sortUp) {
        temp_list.setorderby(pcolumn);
        sortUp = false;
    }
    else {
        temp_list.setorderby(pcolumn + ' desc');
        sortUp = true;
    }
    temp_list.load();
}
```

See Also

- How to: Complete basic operations using JavaScript library code in SharePoint 2013
- Use OData query operations in SharePoint REST requests
- Publish apps for SharePoint
- Boris Moore - JSRender (GitHub)

SSRS: How to Add JPEG and PNG Report Export when SSRS 2012 is integrated with SharePoint 2013

Introduction

Users like to embed report charts in PowerPoint by including an image snapshot of the chart with the correct parameters specified.

SQL Server Reporting Services (SSRS) does not include export to JPEG or PNG in the SharePoint report viewer web part by default, even though SSRS supports report rendering in these formats. For standalone SSRS servers you can enable this capability by editing the RSReportServer.config file (http://technet.microsoft.com/en-us/library/ms157273.aspx#bkmk_rendering), but this does not work for SSRS integrated with SharePoint.

Using the following PowerShell, you can successfully add the extensions.

Workaround

1. Get your application ID (GUID):

```
Get-SPRSServiceApplication
```

2. Add the extensions:

```
New-SPRSExtension -identity "{INSERT YOUR APPLICATION ID HERE}" -
ExtensionType "Render" -name "JPEG" -TypeName
"Microsoft.ReportingServices.Rendering.ImageRenderer.ImageRenderer,Mic
rosoft.ReportingServices.ImageRendering" -ServerDirectives
"<OverrideNames><Name
 Language='en-US'>JPEG</Name></OverrideNames>" -ExtensionConfiguration
"<DeviceInfo><OutputFormat>JPEG</OutputFormat></DeviceInfo>"
 New-SPRSExtension -identity "{INSERT YOUR APPLICATION ID HERE}" -
ExtensionType "Render" -name "PNG" -TypeName
"Microsoft.ReportingServices.Rendering.ImageRenderer.ImageRenderer,Mic
rosoft.ReportingServices.ImageRendering" -ServerDirectives
"<OverrideNames><Name
```

```
 Language='en-US'>PNG</Name></OverrideNames>" -ExtensionConfiguration
"<DeviceInfo><OutputFormat>PNG</OutputFormat></DeviceInfo>"
```

3. You can list the loaded Render extensions:

```
Get-SPRSExtension -Identity "{INSERT YOUR APPLICATION ID HERE}" -
ExtensionType "Render"
```

4: If you make a mistake you can remove them with:

```
Remove-SPRSExtension -Identity "{INSERT YOUR APPLICATION ID HERE}" -
ExtensionType "Render" -Name PNG
```

See Also

- RSReportServer Configuration File
- Report Viewer Web Part (Reporting Services in SharePoint Integrated Mode)

References

- Originally posted to the TechNet Forum here: http://social.technet.microsoft.com/Forums/sqlserver/en-US/1920bbad-5201-4dfa-9aeb-fbd4baf26b7c/how-to-add-jpeg-and-png-report-export-when-ssrs-2012-is-integrated-with-sp-2013

SharePoint 2013: Save Publishing Site as Template

Save Site as a Template

1. Open the site in SharePoint Designer 2013.
2. From the tool bar click on Site Options.
3. Select SaveSiteAsTemplateEnabled, click Modify and set value to *true* if it is false. Click OK, Click OK.
4. Click Administration Web Page from tool bar.
5. On the site settings page, click Master Page from Look and Feel Section.
6. Expand Alternate CSS URL section, select *Use default styles and any CSS files associated with your Master Page,*click OK. This will change look and feel of the site. Look and Feel will be reset after creating template.
7. From SharePoint Designer 2013 tool bar click Save as Template.
8. Enter File Name, Template Name, Template Description, select Include Content (Include Content MUST be selected else site created from this template will not have any page created.)
9. Click OK.
10. Once the site template is created, from SharePoint Designer 2013 tool bar, click Administration Web Page.
11. On the site settings page, click Master Page from Look and Feel Section.
12. Expand Alternate CSS URL section, select *Inherit Alternate CSS URL from parent of this site,* click OK. This action will reset Look and Feel of the site.

Note: Creating Site from Saved Template

After creating new site with this template, on new site look and feel should be changed to get NMIT Navigation. Follow these steps to reset look and feel.

1. On new site click, Site Settings from Gear menu.
2. Click Master Page from Look and Feel section.
3. Expand Alternate CSS URL section, select *Inherit Alternate CSS URL from parent of this site,* click OK.
4. Click Manage Site Features from Site Actions section on Site Settings page.
5. Deactivate SharePoint Server Publishing feature and activate again.

SharePoint 2013: Move the Trace, Usage and IIS Logs to a Log Farm

By default, SharePoint 2013 writes logs to the same drive and partition on which it was installed. Because diagnostic logging can use a large amount of drive space and compromise drive performance, you should configure SharePoint 2013 to write to another drive on which SharePoint 2013 is not installed. You should also consider the connection speed to the drive on which SharePoint 2013 writes the logs. If verbose-level logging is configured, the server records a large amount of data. Therefore, a slow connection might result in poor log performance.http://technet.microsoft.com/en-us/library/ee748656(v=office.15).aspx
Why should you move the SharePoint Logs or IIS Logs from their default directory?

- You'll definitely win space on the root drive (80GB is sometimes not enough).
- You can't read any local log file if your Web Front End or Application Server is down due to Network related issues, due to Windows Server related issues, …
- …

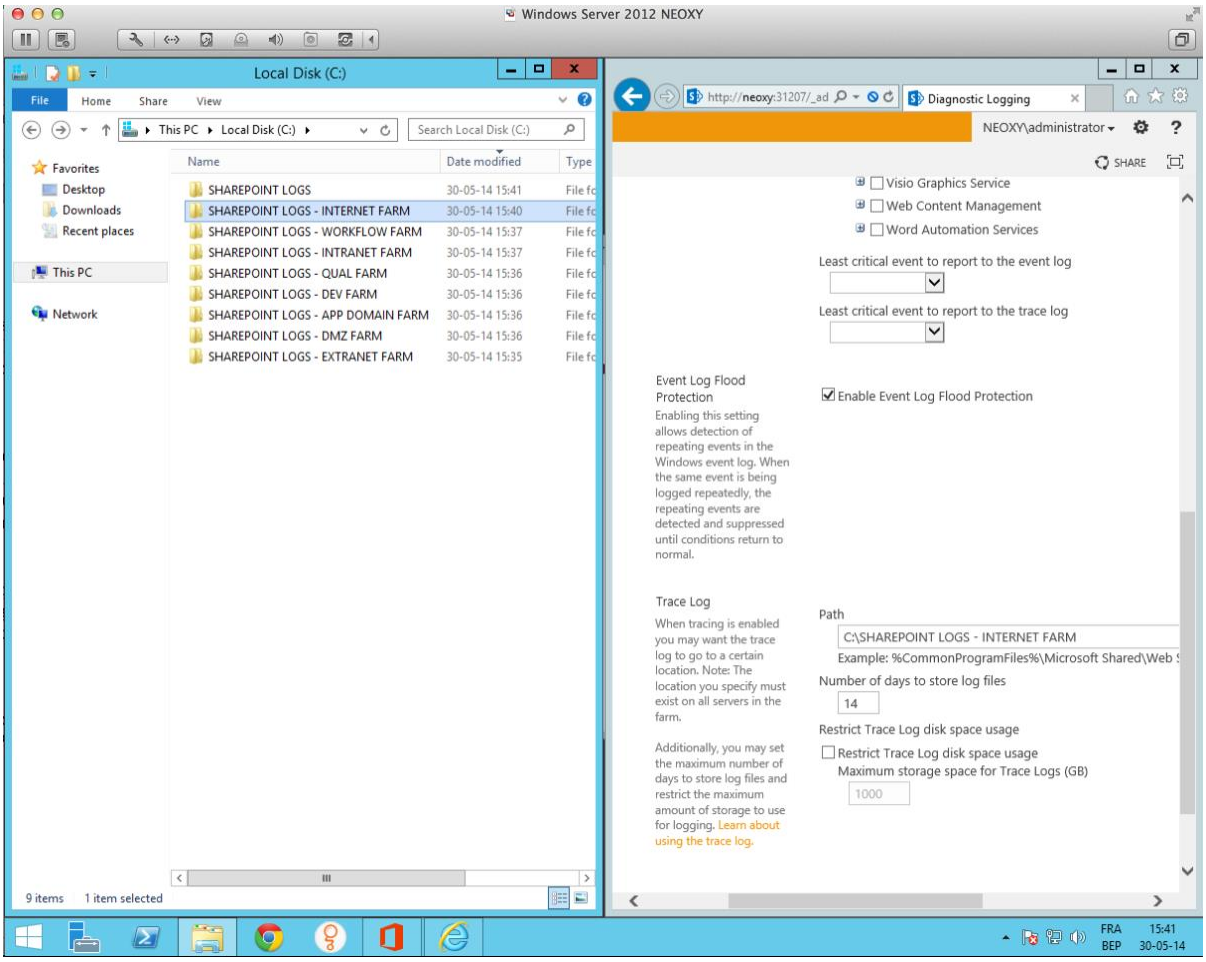
Why this blog post? I recently saw something that I never saw in my life. A customer created a dedicated Log Farm. The customer was legally forced to keep their log files at least for 7 years. So they decided to create a single point of server for all SharePoint and IIS Related logs.

How did they proceed? For each SharePoint Environment, a Shared SharePoint folder was created on a Dedicated Server where logs where available.

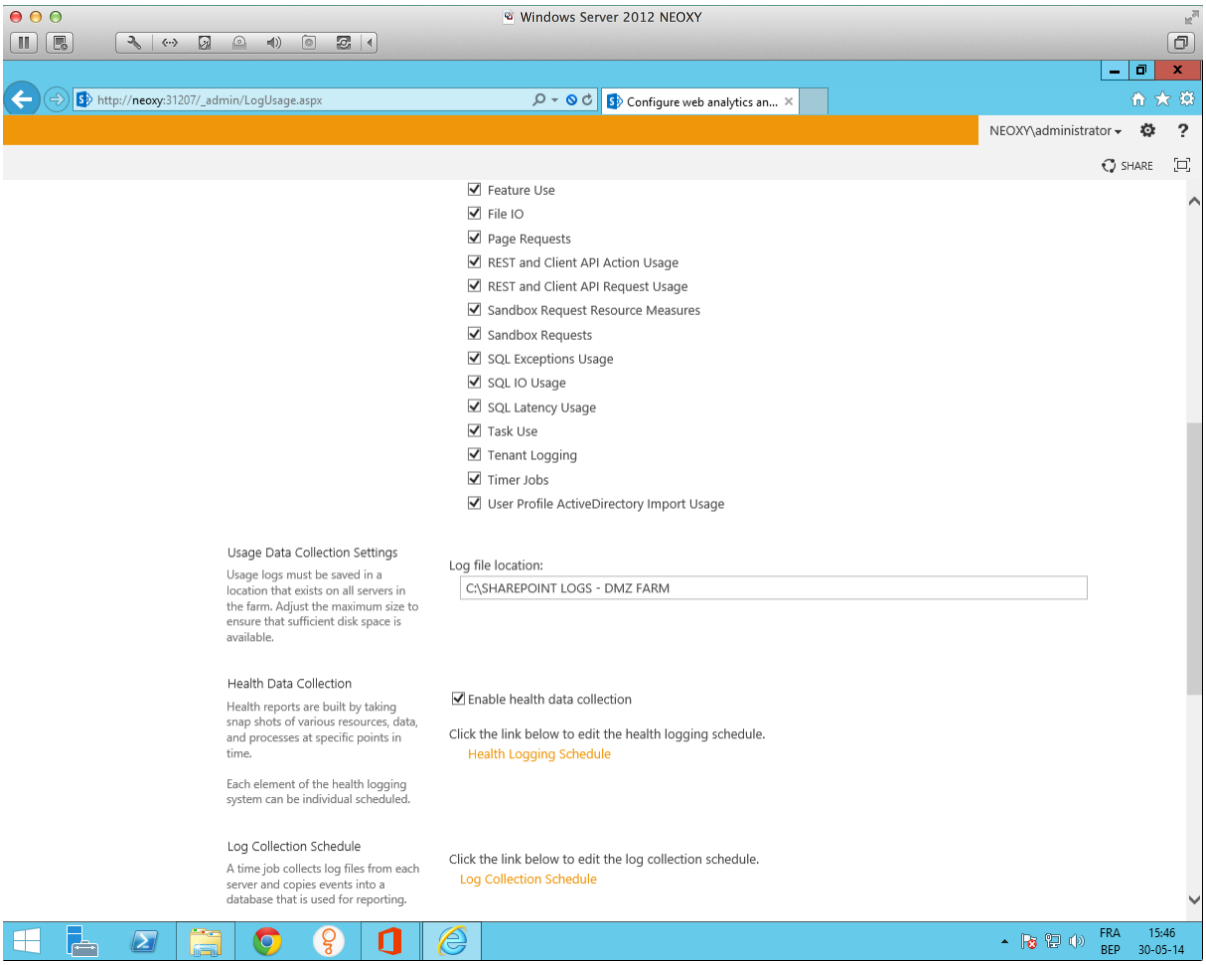
SharePoint Logs

SharePoint has 2 different logs. The Trace and Usage Logs. To move the Trace Logs Under Central Administration navigate to Monitoring and select Configure diagnostic logging (/_admin/metrics.aspx).

In the Trace Log section, in the **Path** box, type the path of the folder to which you want logs to be written.



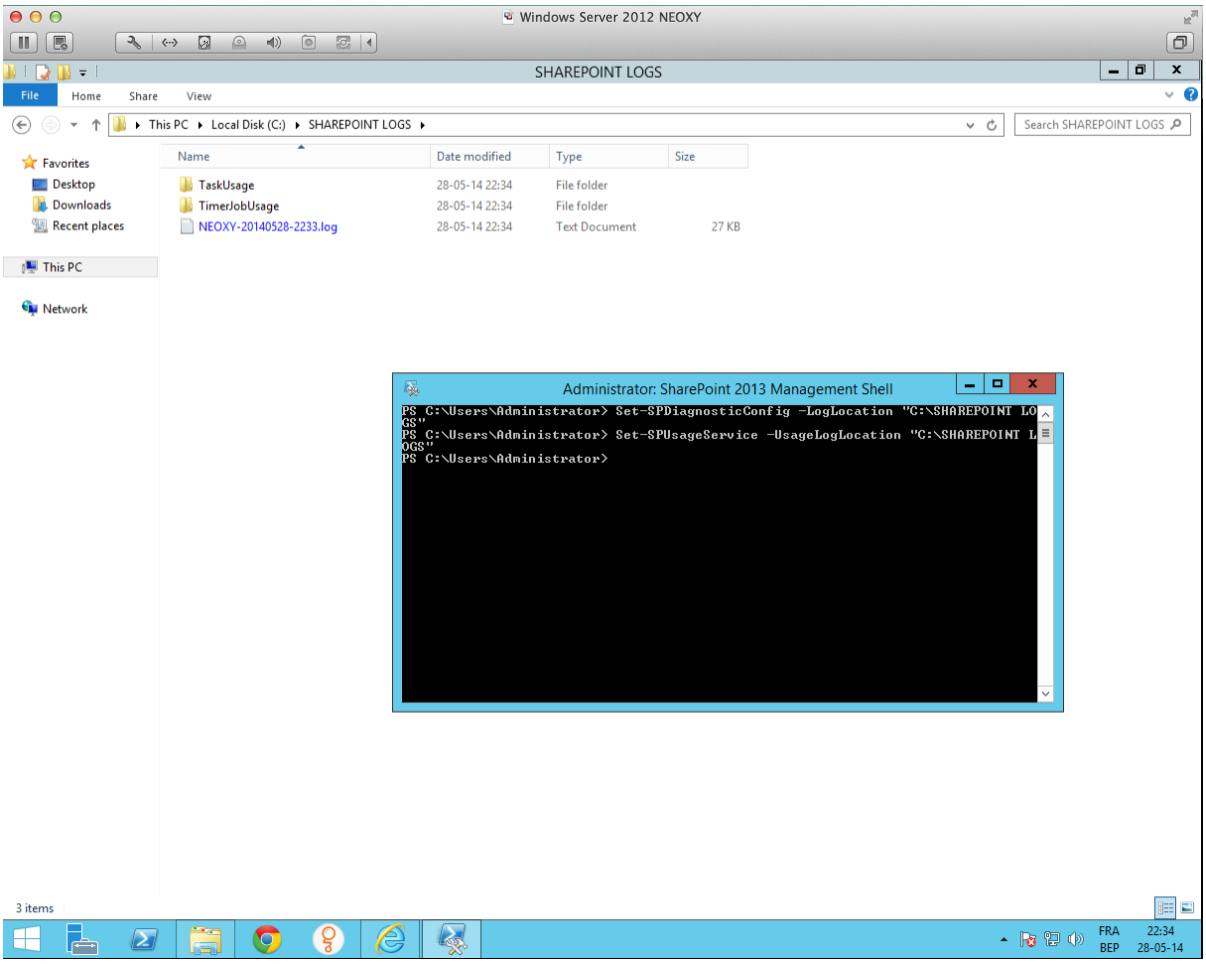
You can also do this for the Usage Data Collection Settings and forward these logs to another server or to another drive.



If you are not very underpad with the Central Administration, you still can use PowerShell to manage the logs.

The Set-SPDiagnosticConfig and Set-SPUsageService cmdlets are the two that you need to configure the new paths for SharePoint Logs. The user needs to be Farm Administrator to run following cmdlets.

- **Set-SPDiagnosticConfig –LogLocation "…"**
- **Set-SPUsageService – UsageLogLocation "…"**

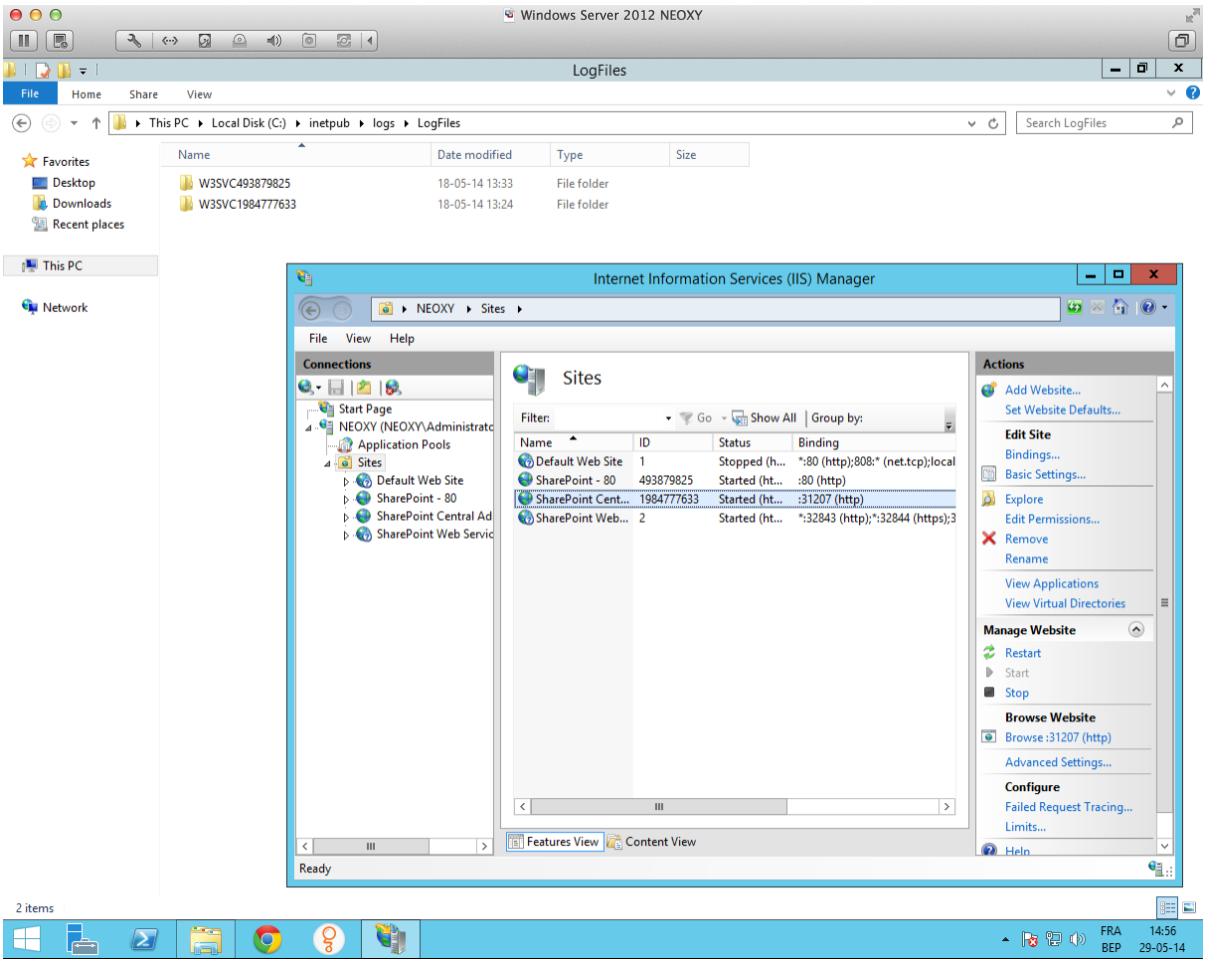


IIS Logs

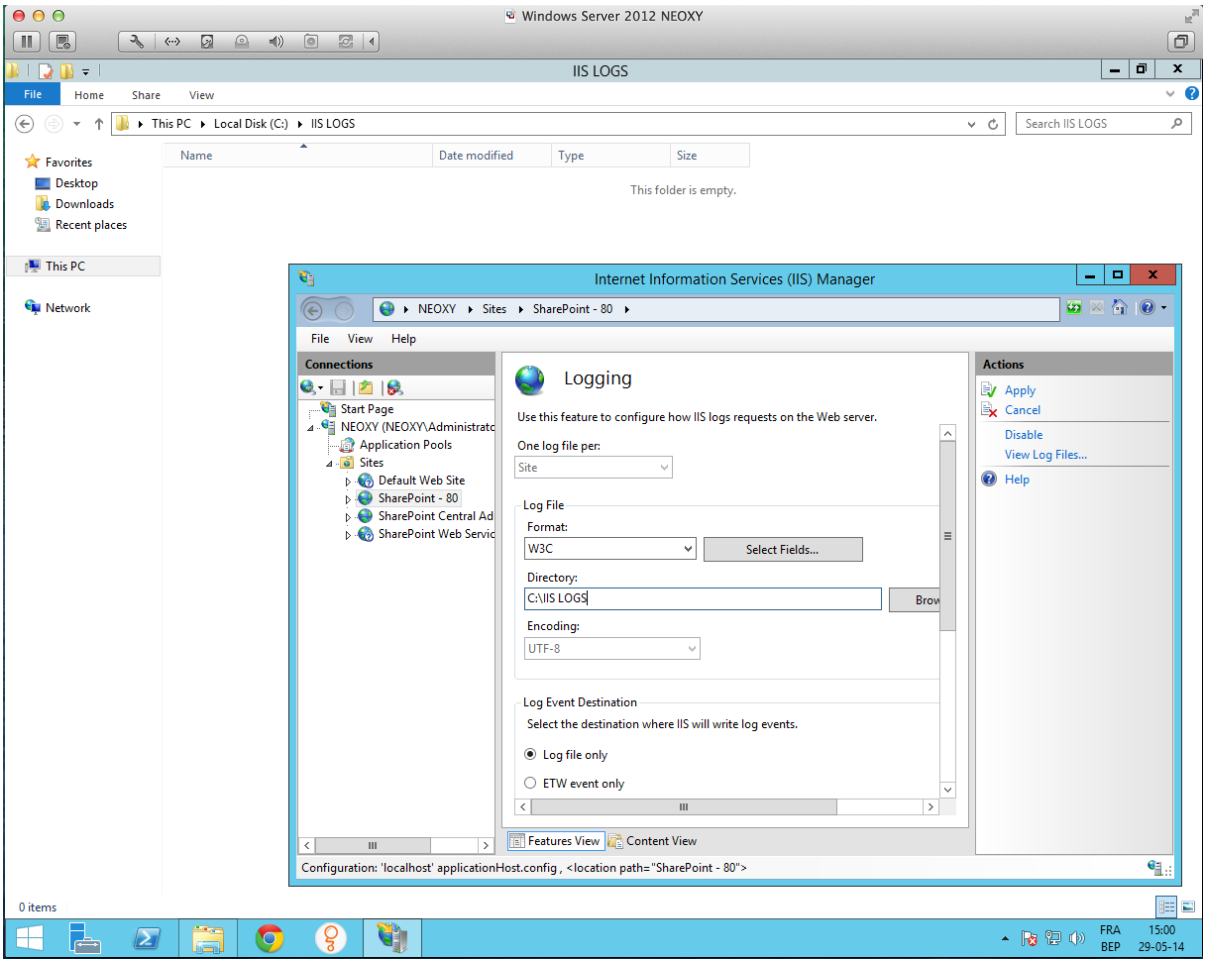
The IIS Server logs are a little bit more complicated but after using only once you'll be able to understand the way of work. The IIS Logs are located under the root => Inetpub => Logs => LogsFiles

Strange isn't it? You'll see some guid's maybe never saw in your Farm. Actually they are the ID's of your IIS Websites. Open IIS Server and under **Sites** you'll see the ID of each site.
The IIS logs files are also located in the root and it isn't a bad idea to move them also to the dedicated log farm.

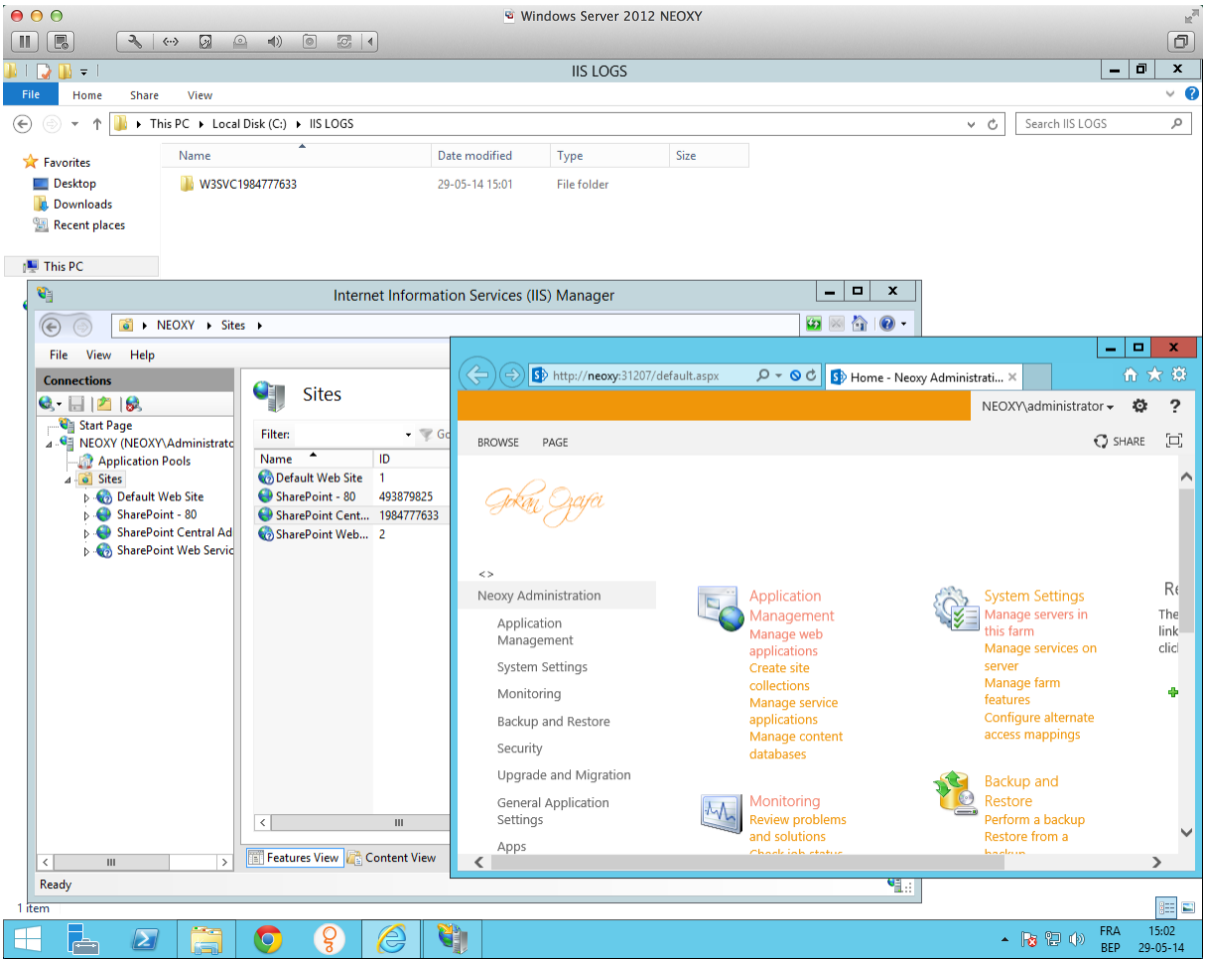
Navigate to the SharePoint Site and open **Logging** in the right pane.The **by default** logging Directory is %SystemDrive% =>Inetpub => Logs => LogFiles.



If you want to apply the same settings as we did for SharePoint – meaning move the logs – well we have to specify the local or shared Log Directory in the Directory Case.



When I surf to the Central Administration, you'll see that a new entry will be done in the local or shared Log Farm. Again another way of gouvernance.


Windows Server 2012 NEOXY
IIS LOGS
File Home Share View
This PC ▸ Local Disk (C:) ▸ IIS LOGS
Search IIS LOGS
Favorites
Desktop
Downloads
Recent places
This PC
Name Date modified Type Size
W3SVC1984777633 29-05-14 15:01 File folder
Internet Information Services (IIS) Manager
NEOXY ▸ Sites ▸
File View Help
Connections
Start Page
NEOXY (NEOXY\Administrato
Application Pools
Sites
Default Web Site
SharePoint - 80
SharePoint Central Ad
SharePoint Web Servic
Sites
Filter:
Name ID
Default Web Site 1
SharePoint - 80 493879825
SharePoint Cent... 1984777633
SharePoint Web... 2
Features View
Content View
Ready
1 item
http://neoxy:31207/default.aspx
Home - Neoxy Administrati...
NEOXY\administrator
BROWSE PAGE
SHARE
Neoxy Administration
Application Management
System Settings
Monitoring
Backup and Restore
Security
Upgrade and Migration
General Application Settings
Apps
Application Management
Manage web applications
Create site collections
Manage service applications
Manage content databases
System Settings
Manage servers in this farm
Manage services on server
Manage farm features
Configure alternate access mappings
Monitoring
Review problems and solutions
Backup and Restore
Perform a backup
Restore from a
FRA
BEP
15:02
29-05-14

SharePoint 2013: Common PowerShell Snippets

Introduction

This wiki has been created to consolidate the SharePoint PowerShell snippets we use on our day to day SharePoint activities.

General

Adding the SharePoint SNAPIN

```
# Add SharePoint Snapin to PowerShell
if((Get-PSSnapin | Where {$_.Name -eq
"Microsoft.SharePoint.PowerShell"}) -eq $null)
{
 Add-PSSnapin Microsoft.SharePoint.PowerShell
}
```

Solution Deployment

Add solution to farm

```
#Replace the wsp path with the actual path of your wsp
Add-SPSolution -LiteralPath C:\Rajesh\Sample.wsp
```

Deploy Solution

```
#Replace the wsp name and WebApplication url
Install-SPSolution -Identity Sample.wsp -WebApplication
"http://contoso.com" –GACDeployment
```

There are multiple parameters to the install-spsolution cmdlet, refer to http://technet.microsoft.com/en-us/library/ff607534.aspx

Remove Solution

```
#Replace the wsp name and WebApplication url
Uninstall-SPSolution -Identity Sample.wsp -WebApplication
```

Enable Feature

```
#Replace the feature identity and WebApplication url
Enable-SPFeature –identity "Rajesh.CustomFeature1" -URL
"http://contoso.com"
```

Disable Feature

```
#Replace the feature identity and WebApplication url
Disable-SPFeature –identity "Rajesh.CustomFeature1" -URL
"http://contoso.com"
```

Uninstall Feature

```
#Removes the feature definition from the farm.
#Replace the feature identity
Uninstall-SPFeature –identity "Rajesh.CustomFeature1”
```

Branding

Changing the master page on all sites at site collection

```
#Replace the site-url with the actual site url
$site = Get-SPSite "http://contoso.com"
foreach ($web in $site.AllWebs) {
#change the master url accordingly
$web.CustomMasterUrl = "/_catalogs/masterpage/V4.master";
$web.Update();
$web.Dispose();
}
foreach ($web in $site.AllWebs) {
#change the master url accordingly
$web.MasterUrl = "/_catalogs/masterpage/v4.master";
$web.Update();
$web.Dispose();
}
$site.Dispose();
```

Set Alternate CSS

```
#Replace the site-url with the actual site url
$web = Get-SPWeb "http://contoso.com"
#Replace the css url accordingly
$web.AlternateCssUrl = "/Style Library/MyStyles/main.css";
$web.AllProperties["__InheritsAlternateCssUrl"] = $True;
$web.Update();
$web.Dispose();
```

Set regional setting/locale

```
#Replace the site-url with the actual site url
$site = Get-SPSite "http://contoso.com"
foreach ($web in $site.AllWebs) {
#Change the Web locale as required, below is for US Eng
$web.Locale = 1033;
$web.Update();
$web.Dispose()
}
$site.Dispose()
```

Document Library

Delete document by name from Document Library

```
$site = new-object Microsoft.SharePoint.SPSite("site-url")
$web = $site.openweb()
$list=$web.Lists["Document-Library-Name "]
$listItems = $list.Items
$listItemsTotal = $listItems.Count
Write-Host $listItemsTotal
for ($x=$listItemsTotal-1;$x -ge 0; $x--)
{
  if($listItems[$x].name.Contains("file")) # file refers to the name
of the document
  {
    Write-Host("DELETED: " + $listItems[$x].name)
    $listItems[$x].Delete()
  }
}
```

Site Management

Delete all sites in the given site

```
# completely deletes the specified Web (including all sub sites)
function RemoveSPWebRecursively( [Microsoft.SharePoint.SPWeb] $web)
{
    Write-Debug "Removing site ($($web.Url))..."
    $subwebs = $web.GetSubwebsForCurrentUser()
    foreach($subweb in $subwebs)
    {
        RemoveSPWebRecursively($subweb)
        $subweb.Dispose()
    }

    $DebugPreference = "SilentlyContinue"
    Remove-SPWeb $web -Confirm:$false
    $DebugPreference = "Continue"
}

$DebugPreference = "SilentlyContinue"
#Replace the site-url with the actual site url
$web = Get-SPWeb "http://contoso.com/subsite1"
$DebugPreference = "Continue"

If ($web -ne $null)
{
    RemoveSPWebRecursively $web
    $web.Dispose()
}
```

PowerShell SharePoint 2010 Developer Dashboard

Enable the Developer Dashboard

```
$service =
[Microsoft.SharePoint.Administration.SPWebService]::ContentService
$addsetting =$service.DeveloperDashboardSettings
$addsetting.DisplayLevel =
[Microsoft.SharePoint.Administration.SPDeveloperDashboardLevel]::OnDem
and
$addsetting.Update()
```

Other languages

This article is also available in the following languages:

- SharePoint 2010: Extraits de code PowerShell courants (fr-FR)

SharePoint 2013: SharePoint and Enterprise Search Survival Guide

Introduction

This is a list of SharePoint 2013 resources related to Enterprise Search that you can add too, and feel free to re-arrange.

Quickly find SharePoint 2013 Enterprise Search Resources!!

- Bing SP2013ES http://www.bing.com/search?q=sp2013es

Main TechNet and MSDN Resources

- Small, medium, and large-size farm architectures for enterprise search. It contains search and farm topology examples, scaling guidance, and hardware requirements (TechNet) Link to Technical Diagrams page . Direct link to Search PDF .
- Search in SharePoint Server 2013 Resource Center (TechNet)
- What's new in Search in SharePoint 2013 (TechNet)
- Search architectural diagrams (posters) (TechNet)
- Administer search in SharePoint Server 2013 (TechNet)
- Index of all PS cmdlets for SharePoint 2013 (TechNet)
- Index of all PS Search cmdlets for SharePoint 2013 (TechNet)
- Survival Guide: FAST Search Server 2010 for SharePoint
- CodePlex - Projects related to SharePoint Search 2013

Migration

- Moving from Search Server 2010 for SharePoint
- Moving from Fast Search Server 2010 for SharePoint (TechNet wiki)
- Moving from ESP 5.3/FSIS
- Moving from Google Search Appliance

Operations

- Procedure to apply OS updates, hotfixes, CU, PU, SP, etc (TechNet Blog)
- Managing data spills (TechNet wiki)

Hybrid Search with SharePoint 2013

- Configure a one-way hybrid environment with SharePoint Server 2013 and Office 365 (white paper)
- Configure a two-way hybrid search environment with SharePoint Server 2013 and Office 365 (white paper)

SPC2012 sessions - Cliff Notes

- Search Architecture in SharePoint 2013: SPC202 (Rune Zakariassen and Thomas Molbach)
- What's new for Search in SharePoint 2013: SPC259 (Glen Anderson and Cem Aykan)
- Capacity Planning, Sizing and High Availability for Search in SharePoint 2013: SPC172 (Barry Waldbaum and Olaf Birkeland)
- Effective Search Deployment and Operations in SharePoint 2013: SPC 095 (Knut Brandrud and Darrin Allred)
- Search Content Enrichment and Extensibility: SPC 203 (K.Hammervold, Runar Olsen and Niels Petter Rasch Olsen)
- Install and Configure SharePoint 2013
- Auto SP Installer

PowerShell scripts on TechNet Gallery

- Import-QueryRules
- Maintain-SPEnterpriseSearchMetadataProperties
- Migrate SharePoint 2010 Search Thesaurus to SharePoint 2013

Topology and Diagnostics

- Determine search service status
- Manage Search Topology in SharePoint 2013 (TechNet)
- Creating the Search SSA in PowerShell
- Deploying SharePoint 2013 SSA Across Multiple Servers Using PowerShell (MSDN Blog)
- Recommended server architecture for small, medium, and large-size farms (Microsoft Download Center)
- What are the sizing recommendations for the search databases? (wiki)
- SharePoint 2013 IO Recommendations
- What is the high availability (HA) story for SharePoint 2013 and search? (wiki)

Content Integration, Connectors and Processing

- XML Mapper
- Customize the SharePoint 2013 search experience with a Content Enrichment web service (MSDN Blog)
- Brilliant Blog outlining Asynchronous Content Enrichment (MSDN Blog)
- IIS Web Sites and User ACLs (TechNet wiki)
- CEWS Pipeline Toolkit (TechNet wiki)
- Supported file types (TechNet Wiki)
- SharePoint 2013 - Configure Content Source and Search Result Source using Powershell

Query Integration

- SharePoint 2013 Document Preview
- Adaptive experiences (MSDN Blog)
- Query Suggestions (TechNet wiki)
- Blog entry for CodePlex SharePoint 2013 Search Query Tool (CodePlex)

Security

- Custom Security Trimming for Search in SharePoint 2013 (MSDN Blog)

Blogs

- Enterprise Search - Authoring Schedule
- Alex Smith: Enterprise Search Blog (MSDN Blog) Some great entries on Documentum!
- Anders Fagerhaug: Adaptive Experiences in SharePoint 2013 (MSDN Blog)
- Anders Fagerhaug and Armen Kirakosyan : XML Connector for SharePoint 2013 (MSDN Blog)
- Balamurugan Kailasam's Blog SharePoint-community.net
- Beyond Search Stephen Arnold (Blog)
- Brian Jackett: The Frog Pond of Technology (Blog)
- Brian Pendergrass: SharePoint Strategery (MSDN Blog)
- Brandon Atkinson's Blog
- Do More With Search (Blog)
- Enterprise Search (Blog)
- Kris Loranger: Enterprise Search Blog (MSDN Blog)
- Nadeem Ishqair: Search Space (MSDN Blog) Connectors, REST API, Search Query Tool on CodePlex
- Mark Sullivan: Enterprise Search Blog (TechNet Blog)
- Max Melcher: SharePoint, Search and other stuff (Blog)
- Mikael Svensen's Blog (blogspot)
- Microsoft Enterprise Search Blog (MSDN Blog)
- Peter Dempsey: Enterprise Search Blog (MSDN Blog)
- Ragnar Heil: SharePoint Nuggets (MSDN Blog)
- Richard DiZerega: SharePoint Blog (MSDN Blog)
- Sezai Komur: SharePoint Blog
- Share-n-dipity Steve Peschka (TechNet Blog)
- SharePoint Brew Russ Maxwell (MSDN Blog)
- SharePoint Dev Blog (MSDN Blog)
- SharePoint for Squirrels Natalya Voskresenskaya (BlogSpot)
- SharePoint IT Pro Blog: ToTheSharePoint (MSDN Blog)
- Sveinar Rasmussen: Security Trimming in SharePoint 2013 (MSDN Blog)
- The Search Guys - Paul Branson, Paul Summers, Matt King (TechNet Blog)

FBA

- Nice Webpart for creating FBA users from SharePoint site.

Building Custom Forms Using SharePoint Designer 2013

Description

In this article I am going to walk through the process of building custom forms using SharePoint Designer 2013. We are going to create three separate forms for a custom list. First a custom new item form, second a display item form and finally an edit item form.

Setting the Scene

I have created a custom list call Request Form which a student will fill to submit an event request. The counselor will use the edit item form and either approve or deny the request.

Building the List

Below is a screenshot of the columns created in the custom list.

Column (click to edit)	Type	Required
Title	Single line of text	✓
Student Name	Single line of text	✓
Student Comment	Multiple lines of text	
Student Date Request	Date and Time	✓
Counselor Name	Single line of text	✓
Counselor Comment	Multiple lines of text	
Counselor Date	Date and Time	
Approved By Counselor	Choice	
Event Type	Choice	✓
New Event Description	Multiple lines of text	
Previous Event Description	Multiple lines of text	
Address Line 1	Single line of text	✓
Address Line 2	Single line of text	
City	Single line of text	✓
State	Choice	✓
Zip Code	Single line of text	
Phone Number	Single line of text	✓
Fax Number	Single line of text	
Email Address	Single line of text	✓
Current Status	Choice	
Modified	Date and Time	
Created	Date and Time	
Created By	Person or Group	
Modified By	Person or Group	

The columns I have shown above are self-explanatory; however, I am going to provide the choices I have added to the Choice column. The default value of the below choice columns are blank.

Approved By Counselor

Name and Type

Type a name for this column.

Column name:

Approved By Counselor

The type of information in this column is:

- ○ Single line of text
- ○ Multiple lines of text
- ◉ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

Require that this column contains information:

○ Yes ◉ No

Enforce unique values:

○ Yes ◉ No

Type each choice on a separate line:

Approved
Denied

Display choices using:

- ◉ Drop-Down Menu
- ○ Radio Buttons
- ○ Checkboxes (allow multiple selections)

Event Type

Name and Type

Type a name for this column.

Column name:

Event Type

The type of information in this column is:

- ○ Single line of text
- ○ Multiple lines of text
- ◉ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

Require that this column contains information:

◉ Yes ○ No

Enforce unique values:

○ Yes ◉ No

Type each choice on a separate line:

New Event
Previous Event

Display choices using:

- ◉ Drop-Down Menu
- ○ Radio Buttons
- ○ Checkboxes (allow multiple selections)

State

Name and Type

Type a name for this column.

Column name:

State

The type of information in this column is:

- ○ Single line of text
- ○ Multiple lines of text
- ◉ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

Require that this column contains information:

◉ Yes ○ No

Enforce unique values:

○ Yes ◉ No

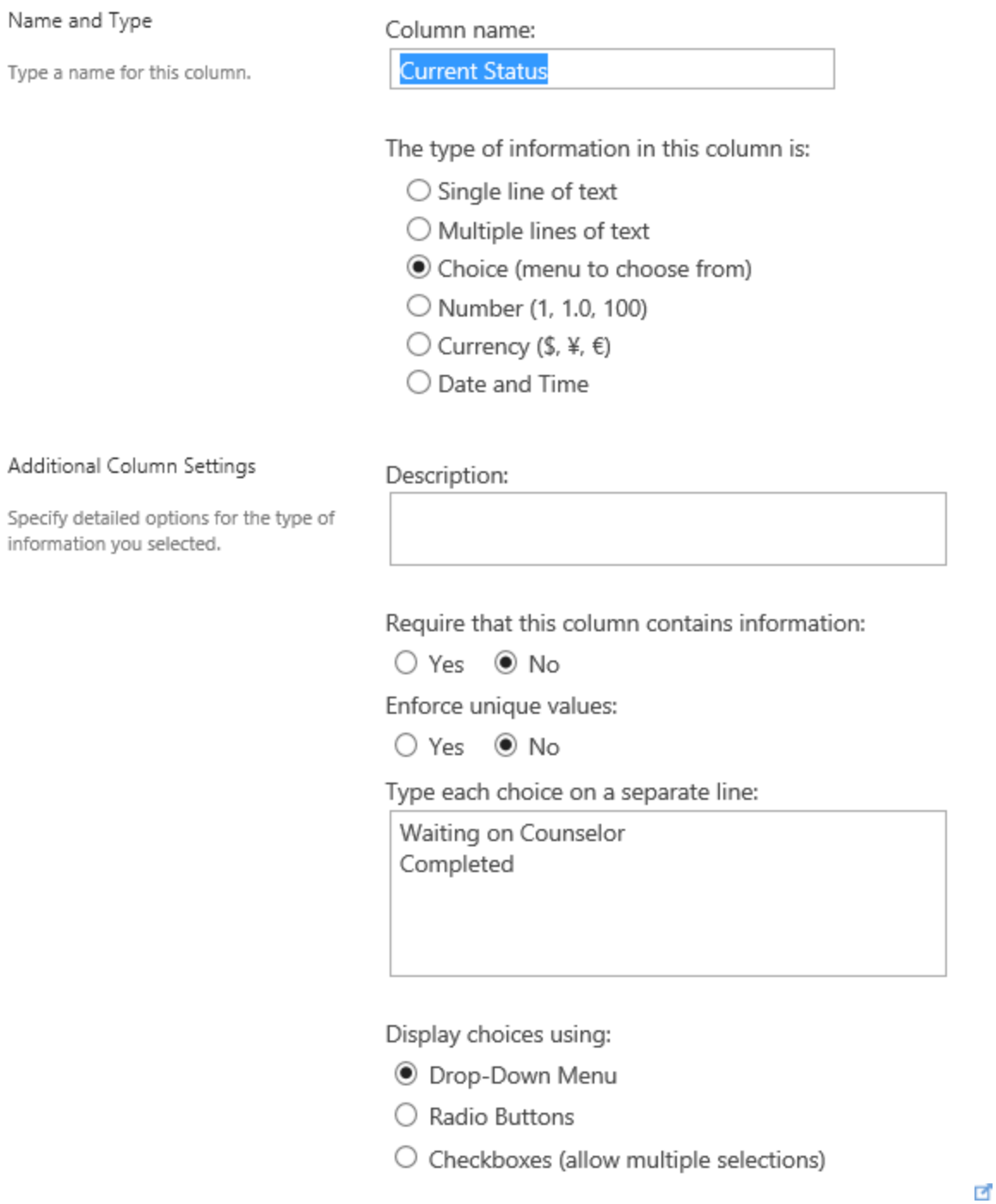
Type each choice on a separate line:

Alabama
Alaska
Arizona
Arkansas
California

Display choices using:

- ◉ Drop-Down Menu
- ○ Radio Buttons
- ○ Checkboxes (allow multiple selections)

Current Status

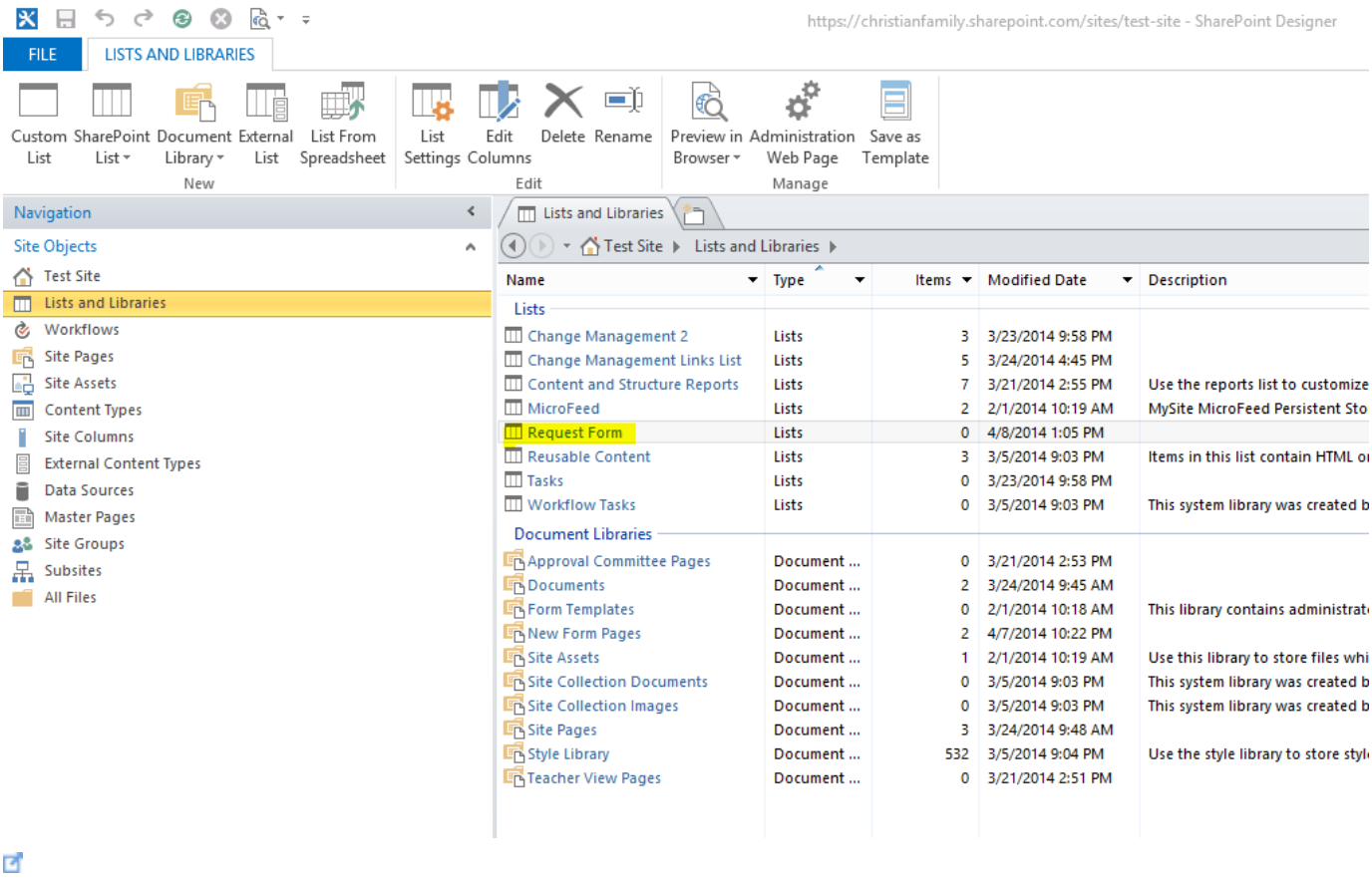


Building the Form

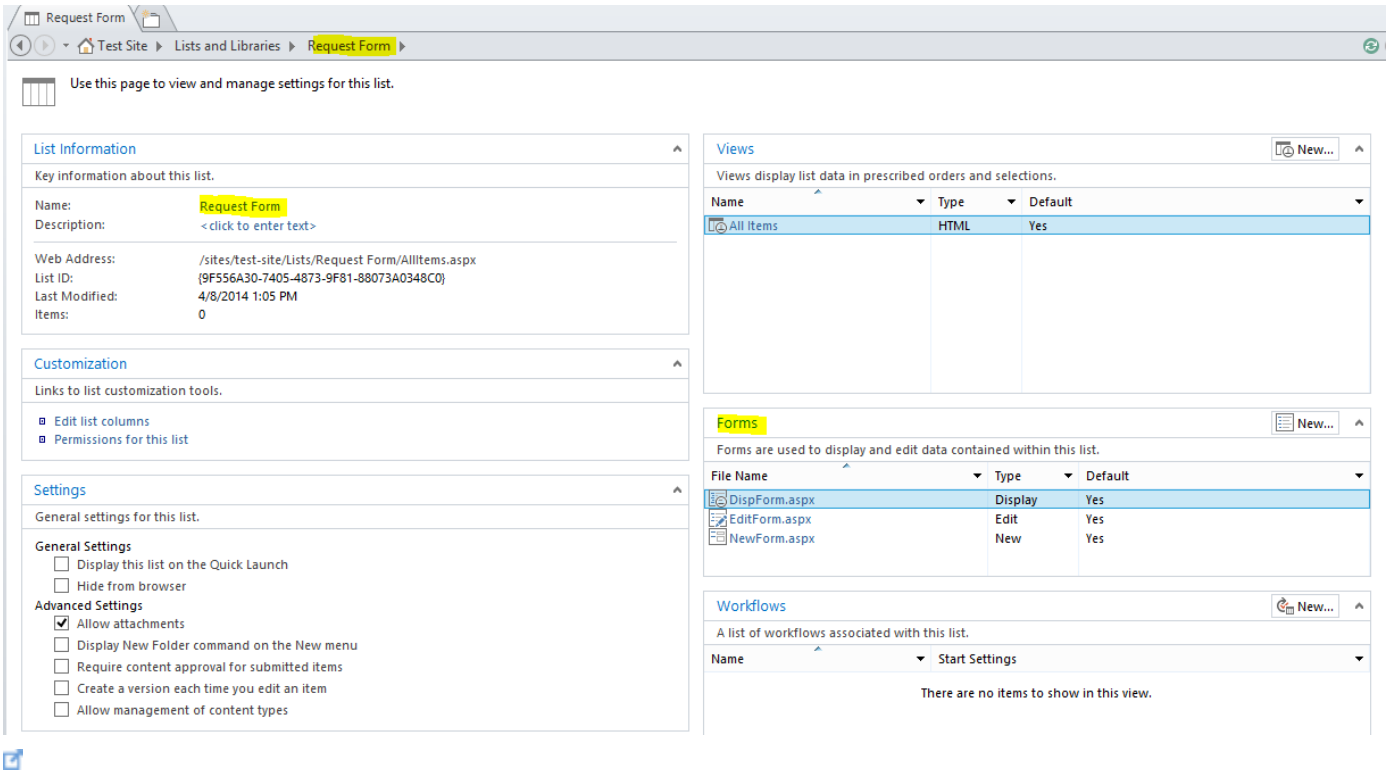
This section describes the process of building custom new, display and edits forms using SharePoint Designer 2013.

New Item Form

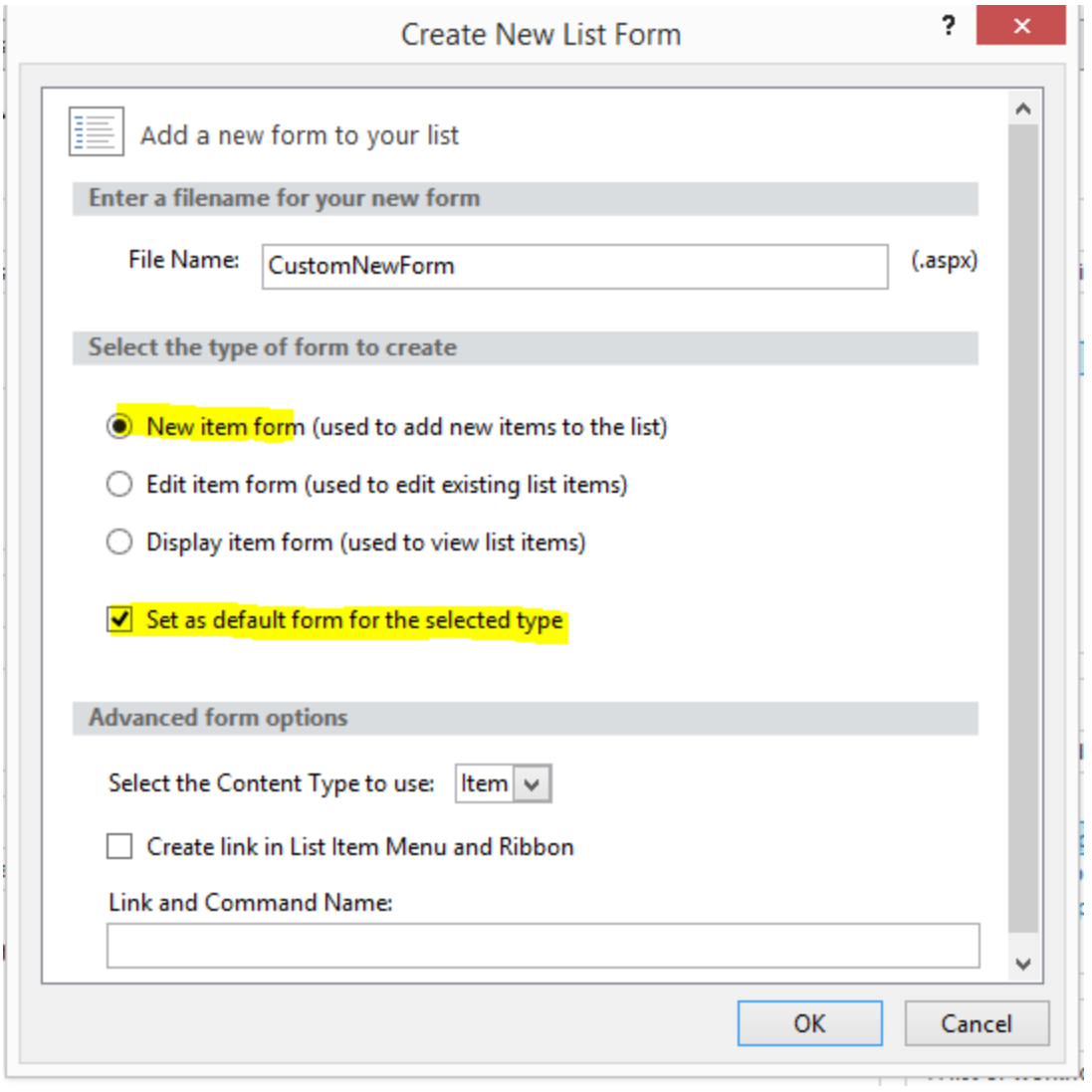
Below is a screenshot of what SharePoint Designer 2013 looks like when I have opened the site. Request Form is the list I am focusing on.



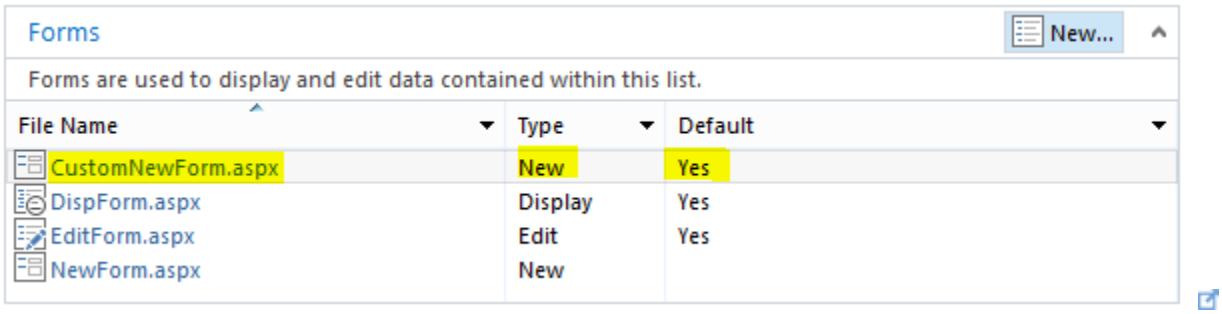
Below are the settings of the list. We are going to use the Forms section to build the custom forms.



Click on the Form's New button and fill the information as shown below. Select the 'Set as default form for the selected type' . Click OK.

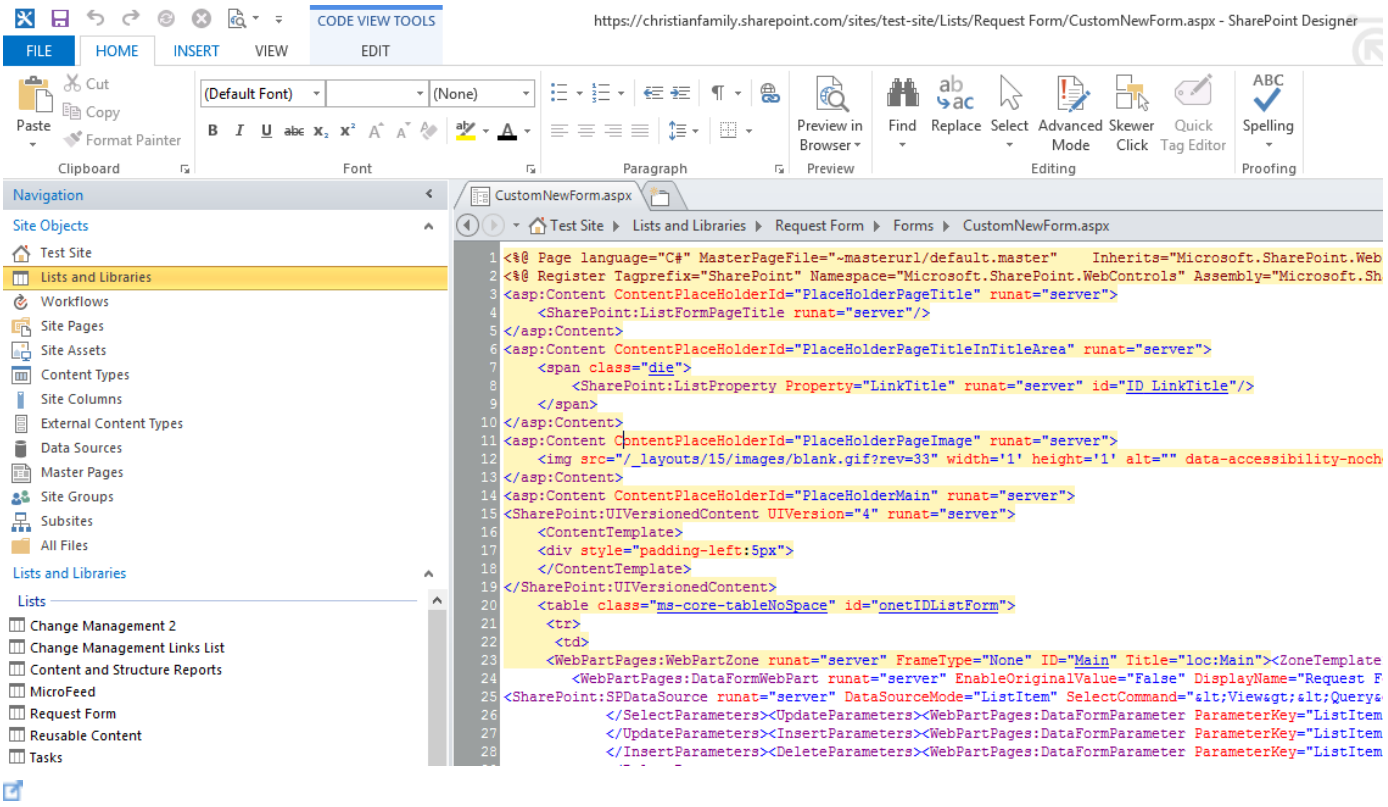


Now in Forms you should see your new custom form. Click on the form. You should now be presented with the CustomNewForm code. Below is a screenshot.

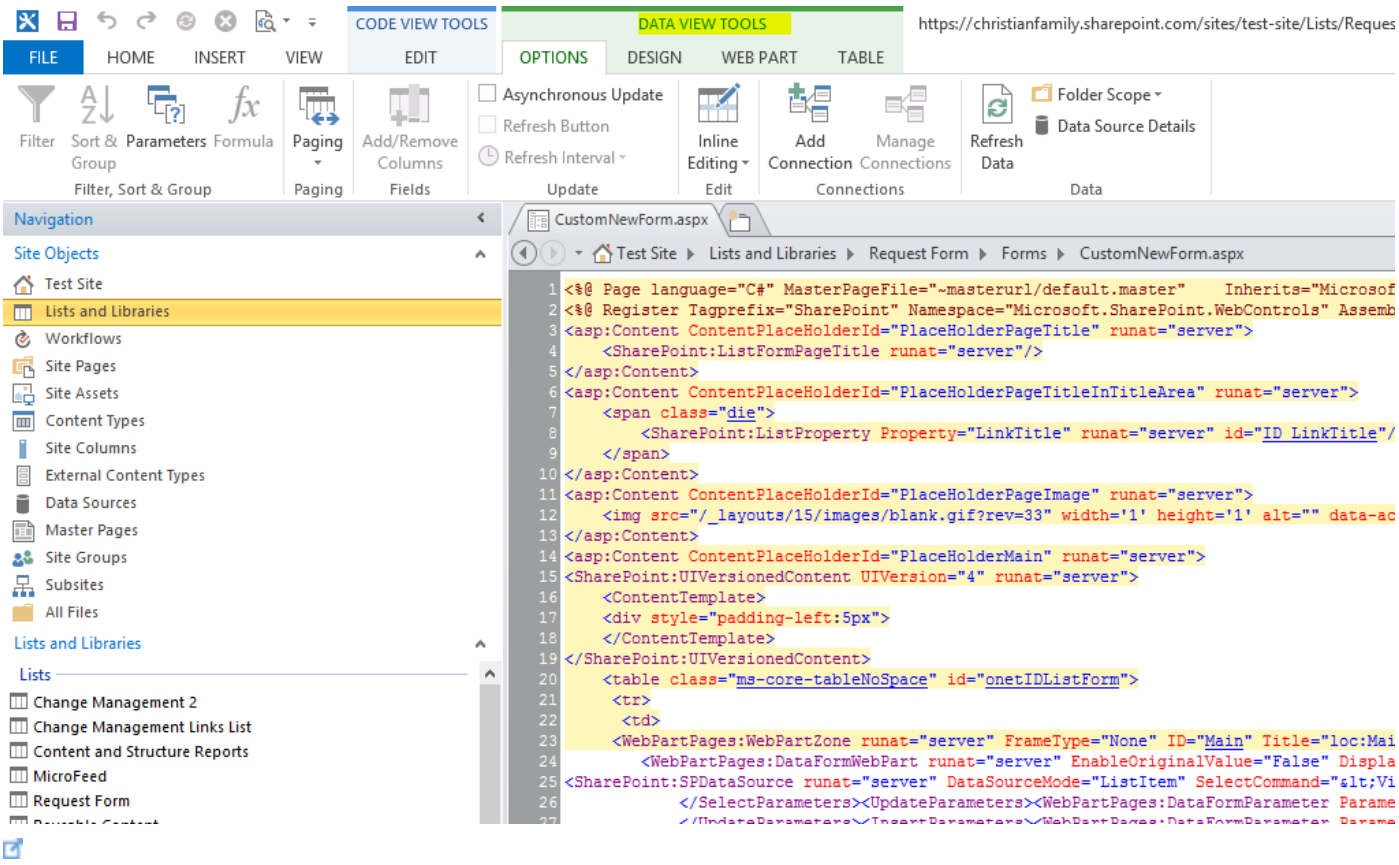


Click on 'CustomNewForm.aspx' to view the code. The first 23 lines are highlighted

yellow which you ***cannot*** edit and hence when you click on it, the Data View Tools will not be available. I have provided screenshots of what the ribbon looks like with and without the tools.
Below is as screenshot without Data View tools. This is because the cursor is on the highlighted yellow code.



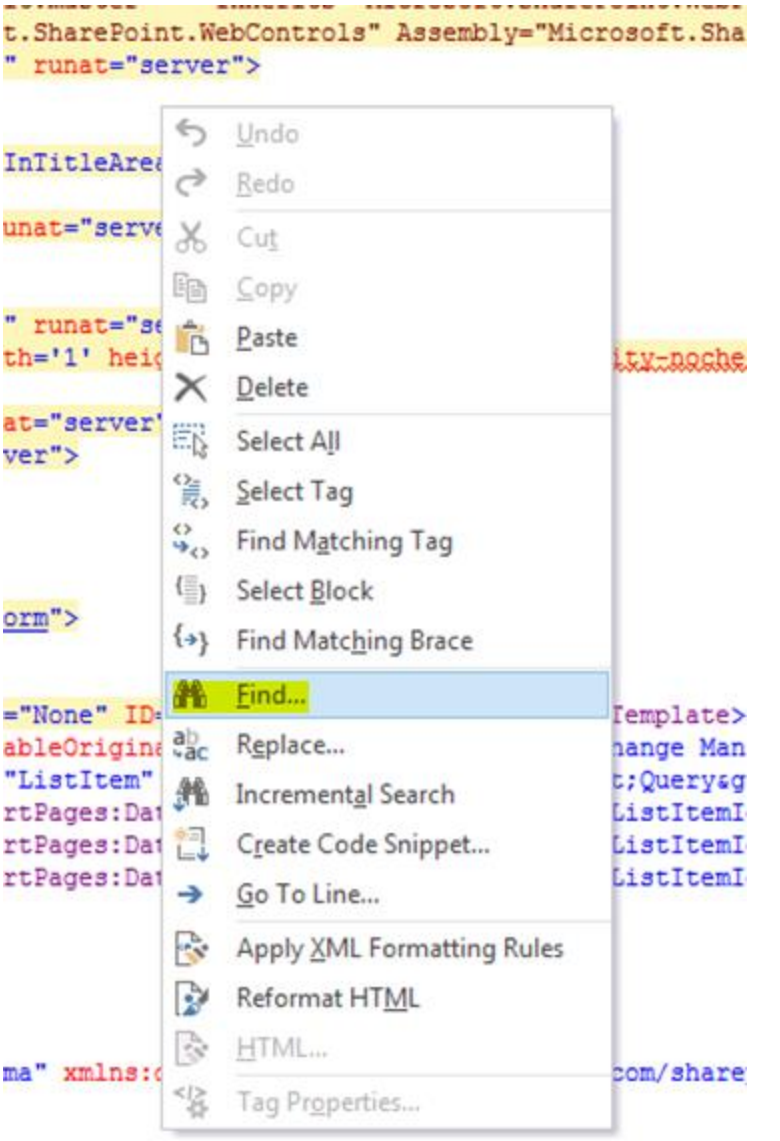
Below is a screenshot with Data View Tools

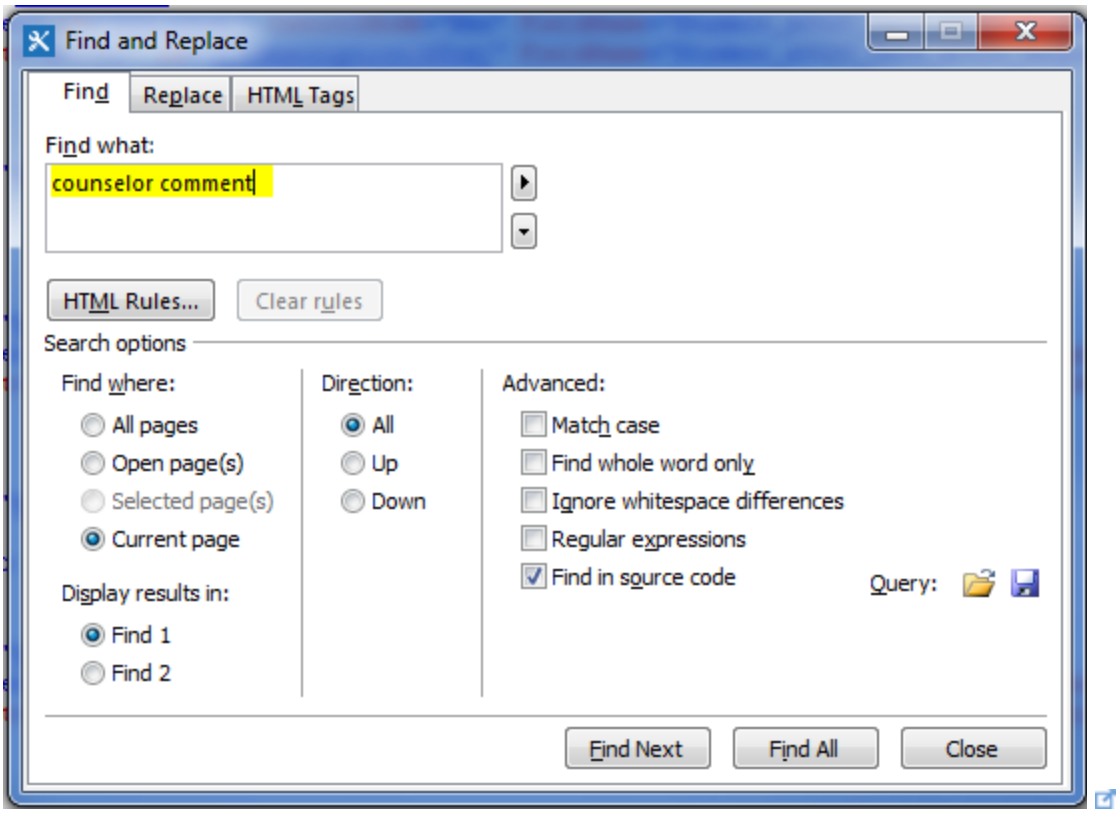


The below table contains the columns that the new form should contain, the rest can be deleted.

List Columns	Student
Title	X
Student Name	X
Student Comment	X
Student Date Request	X
Counselor Name	X
Counselor Comment	delete
Counselor Date	delete
Approved By Counselor	delete
Event Type	X
New Event Description	X
Previous Event Description	X
Address Line 1	X
Address Line 2	X
City	X
State	X
Zip Code	X
Phone Number	X
Fax Number	X
Email Address	X
Current Status	X

In the code right-click and choose 'Find…' from the drop-down list. Type in 'Counselor comment'. Below are screenshots


t.SharePoint.WebControls" Assembly="Microsoft.Sha
" runat="server">
InTitleArea
unat="serve
" runat="se
th='1' heig
at="server"
ver">
orm">
="None" ID=
ableOrigina
"ListItem"
rtPages:Dat
rtPages:Dat
rtPages:Dat
Template>
hange Man
t;Query&g
ListItemI
ListItemI
ListItemI
ma" xmlns:o
com/share
Undo
Redo
Cut
Copy
Paste
Delete
Select All
Select Tag
Find Matching Tag
Select Block
Find Matching Brace
Find...
Replace...
Incremental Search
Create Code Snippet...
Go To Line...
Apply XML Formatting Rules
Reformat HTML
HTML...
Tag Properties...



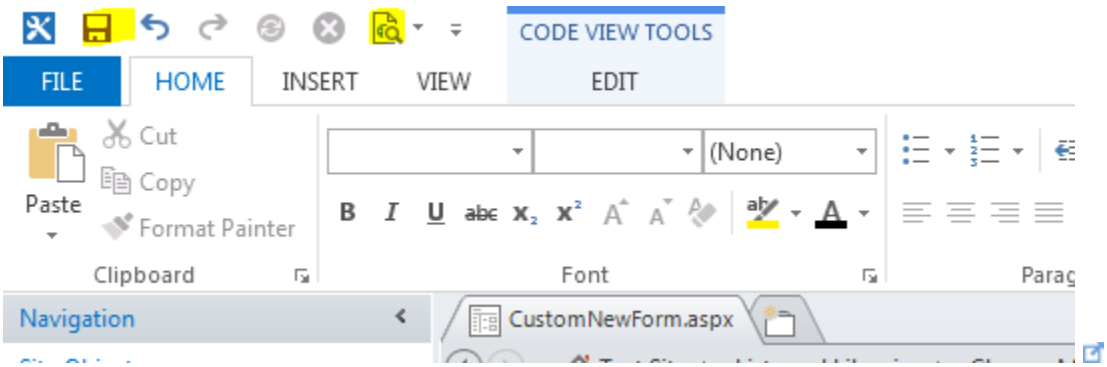
We need to delete both the ms-formlabel and the ms-formbody for this column. Below is the code that **NEEDS TO BE DELETED**. I have also provided a screenshot as well.

```
<tr>
<td width="190px" valign="top" class="ms-formlabel">
<H3 class="ms-standardheader">
<nobr>Counselor Comment</nobr>
</H3>
</td>
<td width="400px" valign="top" class="ms-formbody">
<SharePoint:FormField runat="server" id="ff6{$Pos}" ControlMode="New" FieldName="Counselor_x0020_Comment" __designer:bind="{ddwrt:DataBind('i',concat('ff6',$Pos),'Value','ValueChanged','ID',ddwrt:EscapeDelims(string(@ID)),'@Counselor_x0020_Comment')}"/>
<SharePoint:FieldDescription runat="server" id="ff6description{$Pos}" FieldName="Counselor_x0020_Comment"ControlMode="New"/>
```

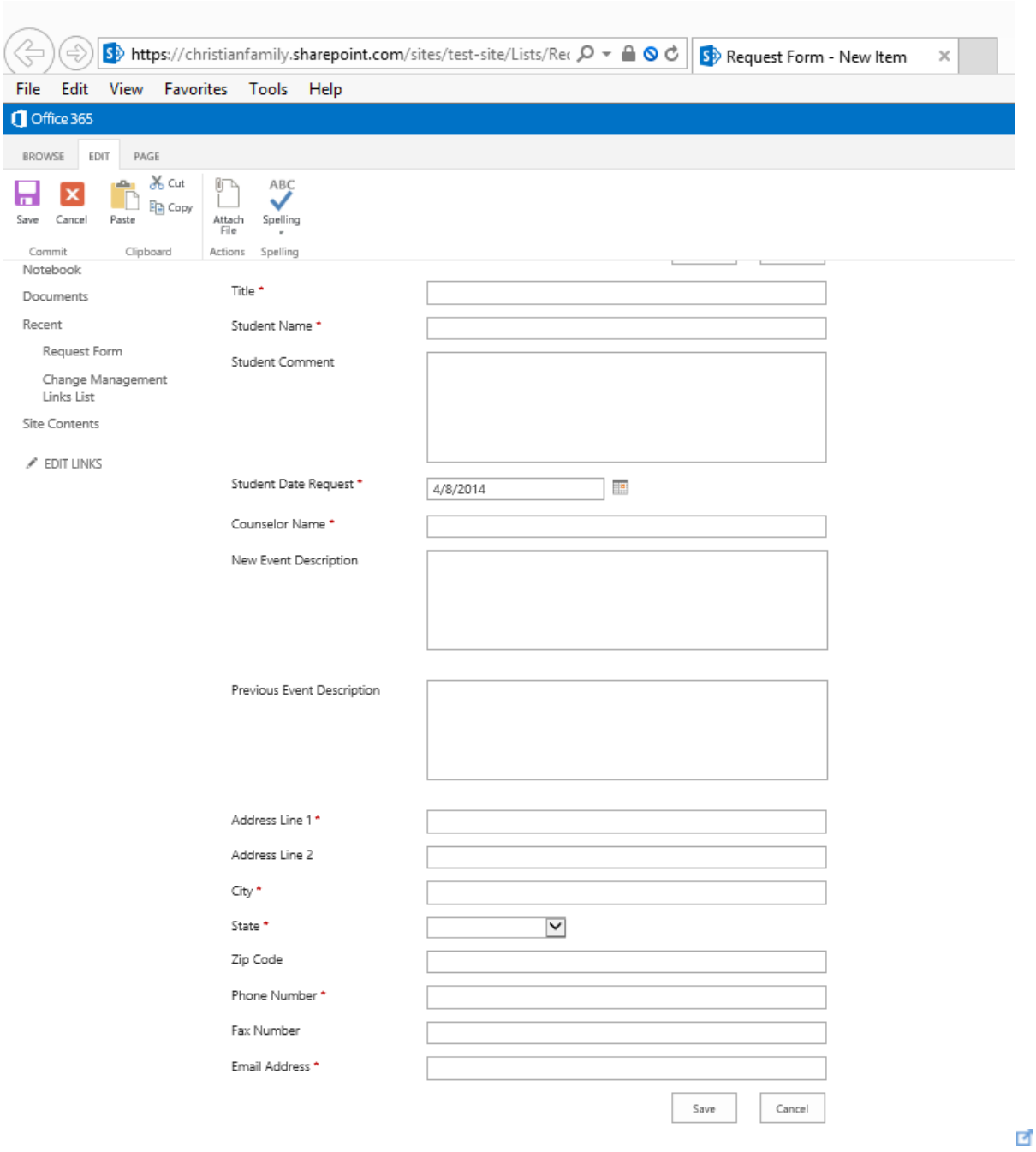
```
</td>
</tr>
```

```
<tr>
    <td width="190px" valign="top" class="ms-formlabel">
        <H3 class="ms-standardheader">
            <nobr>Counselor Comment</nobr>
        </H3>
    </td>
    <td width="400px" valign="top" class="ms-formbody">
        <SharePoint:FormField runat="server" id="ff6{$Pos}" ControlMode="New" FieldName="Counselor_x0020_Comment" __des
        <SharePoint:FieldDescription runat="server" id="ff6description{$Pos}" FieldName="Counselor_x0020_Comment" Contr
    </td>
</tr>
<tr>
```

Do the same for the other columns that need to be deleted for this view.
Once you have deleted the columns, click Save and then Preview in Browser. Saving the changes you made in the form can take a few seconds.



Below is a screenshot of the CustomNewForm.aspx

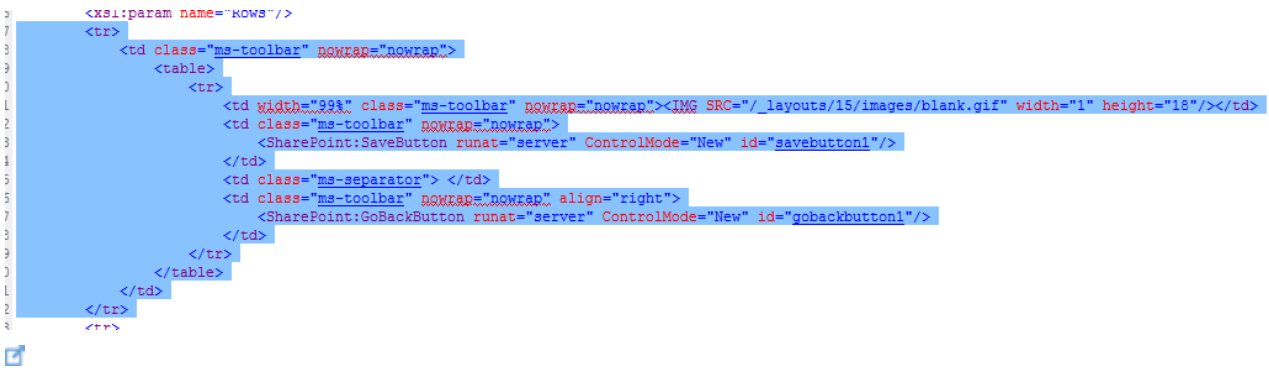


Remove the Save and Cancel button

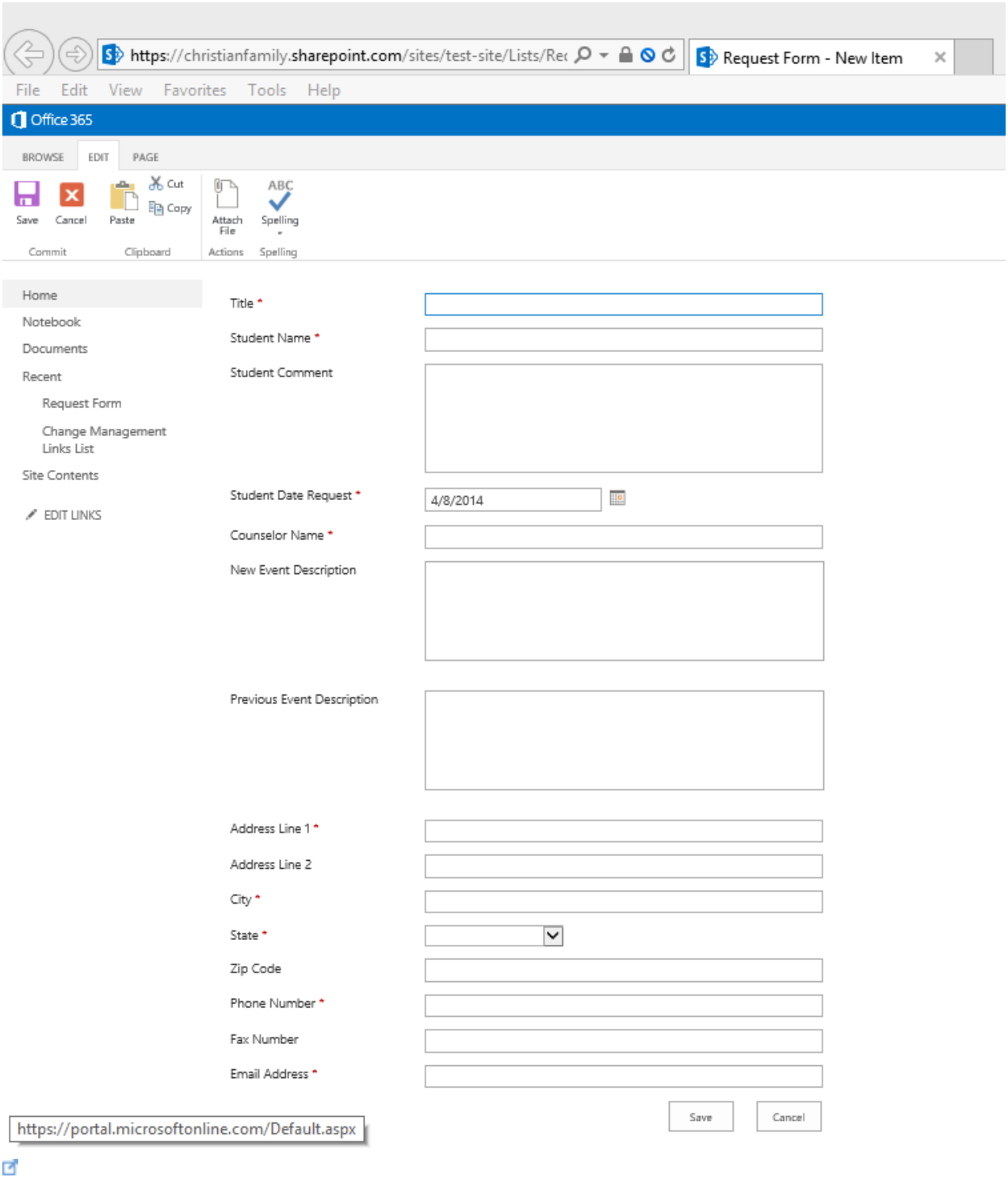
I am now going to remove the Save and Cancel button from the top and give this form a title. In SharePoint Designer 2013, in the code right-click and choose 'Find...' from the drop-down list. Type in 'savebutton1'. Below is what you are looking for. Delete that

code. I have also provided a copy of that code below.

```
<tr>
<td class="ms-toolbar" nowrap="nowrap">
<table>
<tr>
<td width="99%" class="ms-toolbar" nowrap="nowrap"><IMG SRC="/_layouts/15/images/blank.gif" width="1"height="18"/></td>
<td class="ms-toolbar" nowrap="nowrap">
<SharePoint:SaveButton runat="server" ControlMode="New" id="savebutton1"/>
</td>
<td class="ms-separator"> </td>
<td class="ms-toolbar" nowrap="nowrap" align="right">
<SharePoint:GoBackButton runat="server" ControlMode="New" id="gobackbutton1"/>
</td>
</tr>
</table>
</td>
</tr>
```



Again, click Save and then Preview in Browser. Now, as shown below, you should not see the Save and Cancel button on the top.



Add Title

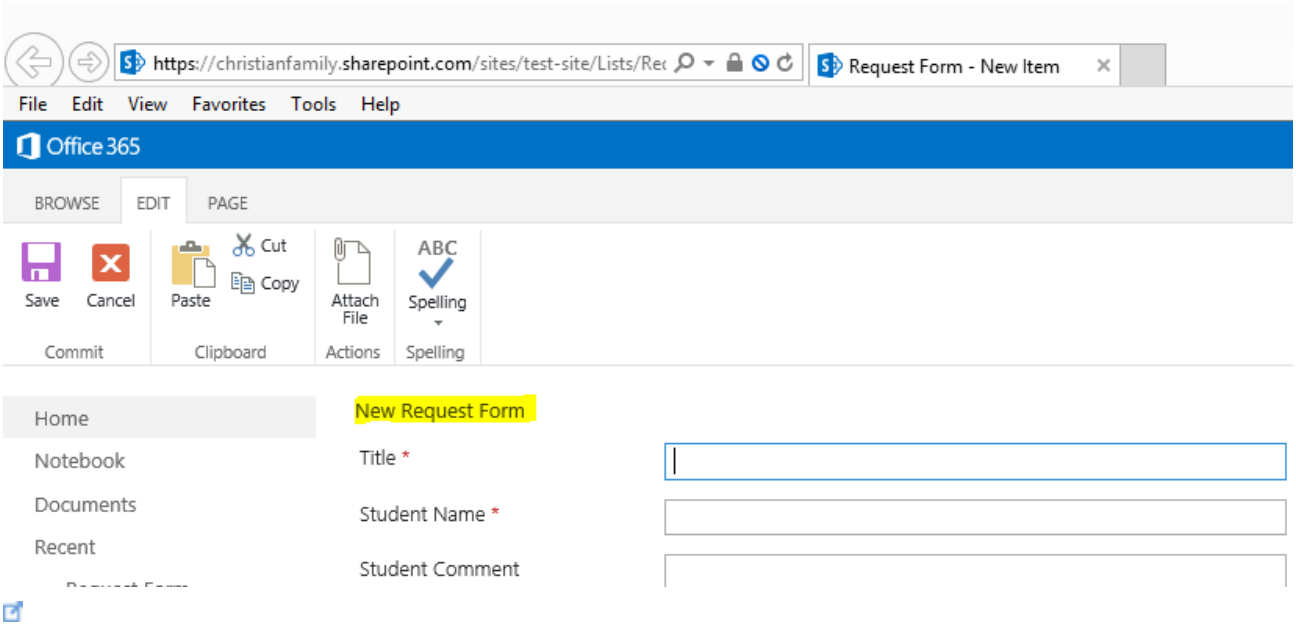
In the CustomNewForm.aspx search for dvt_1.body. Below is what you should find.

```
    </div>
</xsl:template>
<xsl:template name="dvt_1.body">
    <xsl:param name="Rows"/>
            <tr>
        <td class="ms-toolbar" nowrap="nowrap">
            <SharePoint:FormToolBar runat="server" ControlMode="New"/>
            <SharePoint:ItemValidationFailedMessage runat="server" ControlMode="New"/>
        </td>
    </tr>
    <xsl:call-template name="dvt_1.rowedit"/>
    <tr>
        <td class="ms-toolbar" nowrap="nowrap">
            <table>
```

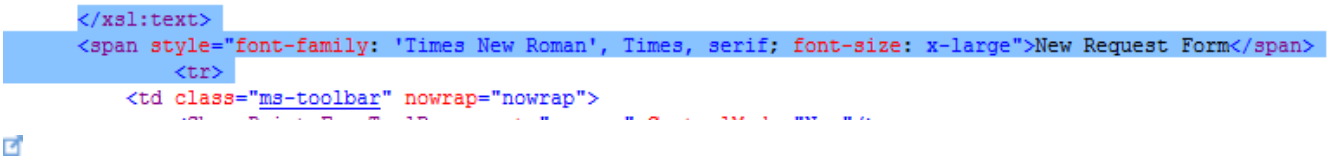
Right below <xls:template name="dvt_1.body"> , type in a form name, for example 'New Request form'. Below is a screenshot

```
</xsl:template>
<xsl:template name="dvt_1.body">
    <xsl:param name="Rows"/>
    New Request Form
            <tr>
        <td class="ms-toolbar" nowrap="nowrap">
            <SharePoint:FormToolBar runat="server" ControlMode="New
            <SharePoint:ItemValidationFailedMessage runat="server"
        </td>
    </tr>
    <xsl:call-template name="dvt_1.rowedit"/>
    <tr>
```

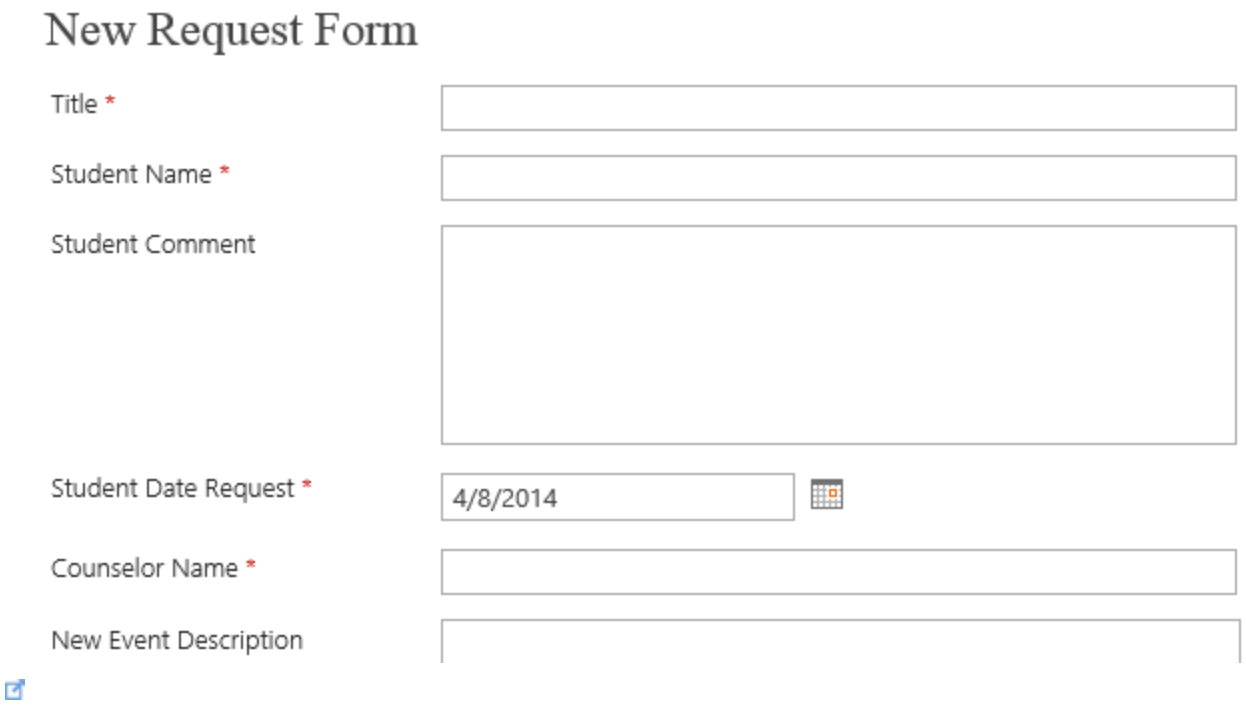
Again, click Save and then Preview in Browser. Below is what you will see.



To make the text look like a prominent title, go back to SharePoint Designer, highlight the 'New Request Form', click on the Home ribbon and select the font type and size. I selected my font type to be 'Times New Roman' and font size as x-Large which will automatically change the code. Below is what you will see



Again, click Save and then Preview in Browser. Below is what you will see.

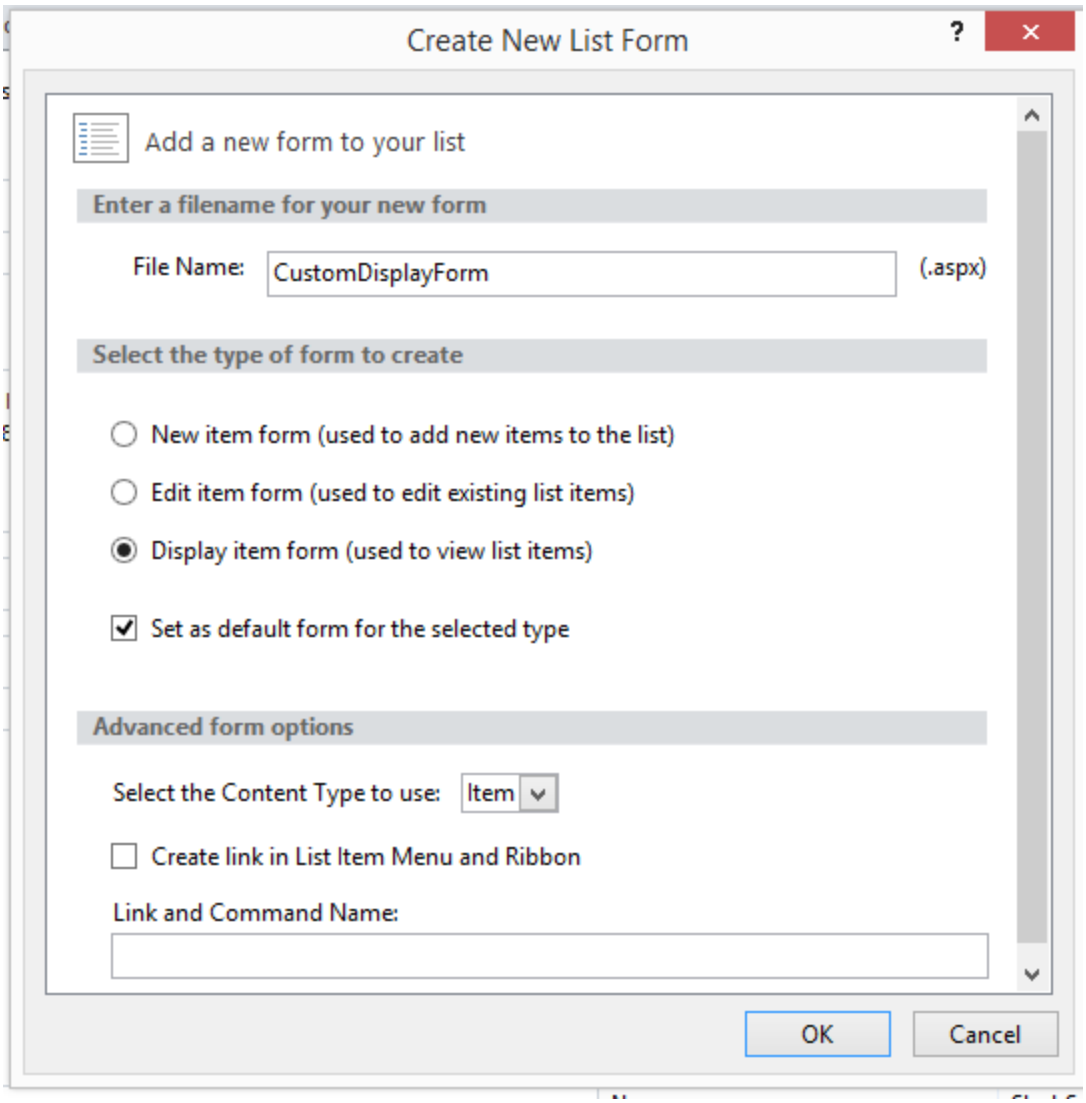


Below is a video which walks you through the steps show above. (Click Here to view a high resolution video)

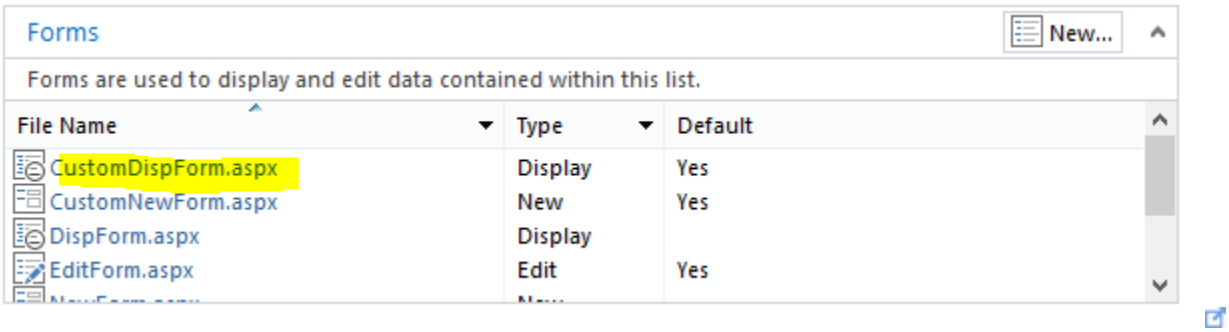
Display Item Form

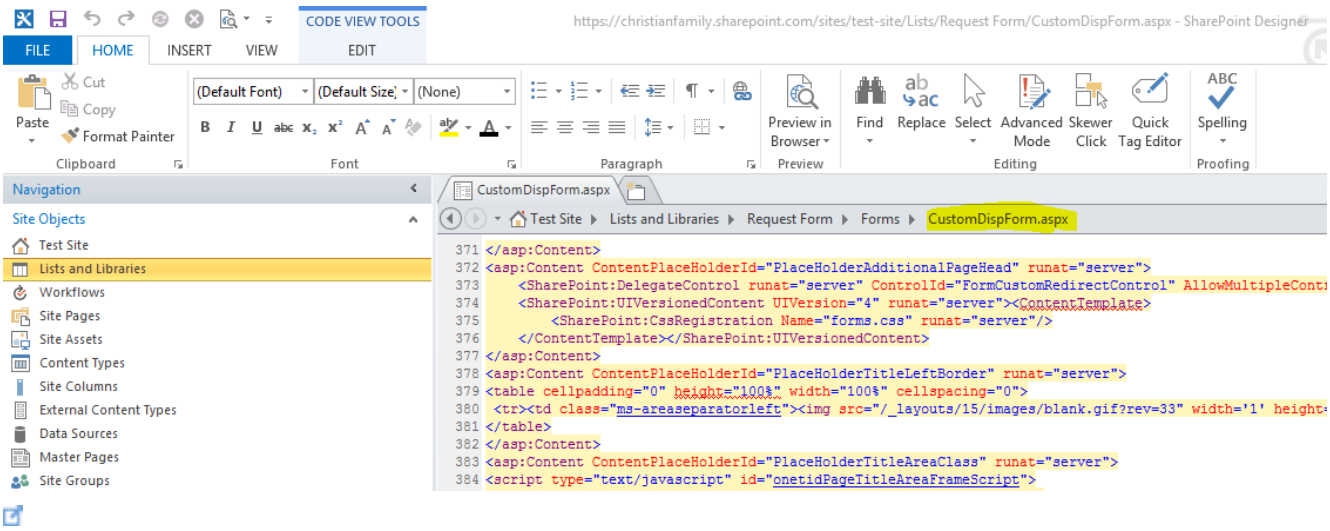
In this section we are going to build the Display Form. In this form we are ***not going to delete*** any columns, however, we are going to remove the 'Close' button from the top and give the form a different name.

Click on the Form's New button and fill the information as shown below. Select the 'Set as default form for the selected type' . Click Ok.



Now in Forms you should see your new custom form. Click on the form. You should now be presented with the CustomDisplayForm code. Below are screenshots.





In the code right-click and choose 'Find...' from the drop-down list. Type in 'gobackbutton1'. Below is what you are looking for. Delete that code. I have also provide a copy of that code below.

```
<tr>
<td class="ms-toolbar" nowrap="nowrap">
<table>
<tr>
<td width="99%" class="ms-toolbar" nowrap="nowrap"><IMG SRC="/_layouts/15/images/blank.gif" width="1"height="18"/></td>
<td class="ms-toolbar" nowrap="nowrap" align="right">
<SharePoint:GoBackButton runat="server" ControlMode="Display" id="gobackbutton1"/>
</td>
</tr>
</table>
</td>
</tr>
```

```
<tr>
    <td class="ms-toolbar" nowrap="nowrap">
        <table>
            <tr>
                <td width="99%" class="ms-toolbar" nowrap="nowrap"><IMG SRC="/_layouts/15/images/blank.gif" width="1" height="18"/>
                <td class="ms-toolbar" nowrap="nowrap" align="right">
                    <SharePoint:GoBackButton runat="server" ControlMode="Display" id="gobackbutton1"/>
                </td>
            </tr>
        </table>
    </td>
</tr>
```

Again, click Save and then Preview in Browser. Now, you should ***not*** see the Close button on the top.

Below is a video which walks you through the process of deleting the top Close button and adding a title. (Click Here to view a high resolution video)

Edit Item Form

In this section we are going to build the Edit form. In this form we are ***not going to delete*** any columns, however, make changes so that some of them cannot be ***edited***. We are also going to remove the Save and Cancel buttons from the top and change the title.

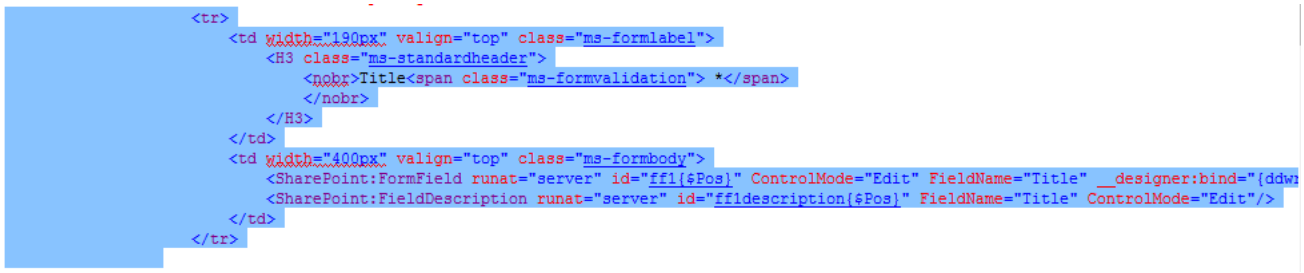
Below is a table which shows which columns can and cannot be edited.

List Columns	Edit View Form
Title	Read Only
Student Name	Read Only
Student Comment	Read Only
Student Date Request	Read Only
Counselor Name	Read Only
Counselor Comment	Edit
Counselor Date	Edit
Approved By Counselor	Edit
Event Type	Read Only
New Event Description	Read Only
Previous Event Description	Read Only
Address Line 1	Read Only
Address Line 2	Read Only
City	Read Only
State	Read Only
Zip Code	Read Only
Phone Number	Read Only
Fax Number	Read Only
Email Address	Read Only
Current Status	Read Only

In the CustomEditForm.aspx code, for all the columns we want to setup as read only, we need to replace the Edit code with the Display code i.e. replace the Edit code shown below (both text and screenshot are provided)

```
<tr>
<td width="190px" valign="top" class="ms-formlabel">
<H3 class="ms-standardheader">
<nobr>Title<span class="ms-formvalidation"> *</span>
</nobr>
</H3>
</td>
```

```
<td width="400px" valign="top" class="ms-formbody">
<SharePoint:FormField runat="server" id="ff1{$Pos}" ControlMode="Edit" FieldName="Title" __designer:bind="{ddwrt:DataBind('u',concat('ff1',$Pos),'Value','ValueChanged','ID',ddwrt:EscapeDelims(string(@ID)),'@Title')}"/>
<SharePoint:FieldDescription runat="server" id="ff1description{$Pos}" FieldName="Title" ControlMode="Edit"/>
</td>
</tr>
```

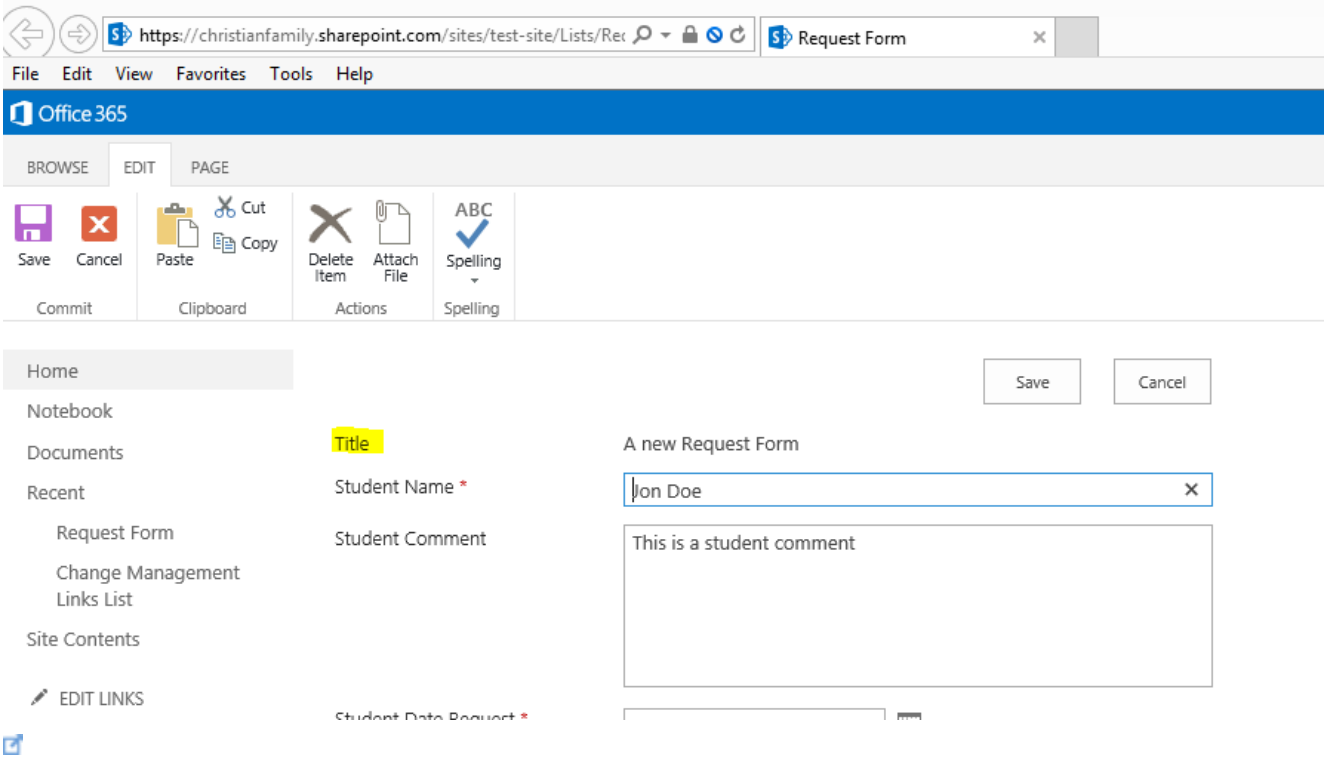


With the Display code as shown below (both text and screenshot are provided)

```
<tr>
<td width="190px" valign="top" class="ms-formlabel">
<H3 class="ms-standardheader">
<nobr>Title</nobr>
</H3>
</td>
<td width="400px" valign="top" class="ms-formbody">
<xsl:value-of select="@Title"/>
</td>
</tr>
```

```
<tr>
    <td width="190px" valign="top" class="ms-formlabel">
        <H3 class="ms-standardheader">
            <nobr>Title</nobr>
        </H3>
    </td>
    <td width="400px" valign="top" class="ms-formbody">
        <xsl:value-of select="@Title"/>
    </td>
</tr>
```

Below is a screenshot of what the Edit view looks like when the Title column has been setup as display only.



Below is a video which walks you through the process of editing the edit item form to force some of the fields to be in display mode only. In this video I have first purposefully left the Title column as it is and also copied the code over from the display form to show you both. Then I have made all the changes to only allow the Counselor Comment, Counselor Date and Approved by Counselor fields to be editable. (Click Here to view a high resolution video)

Conclusion

In this article I have not described the use of the Current Status column that we created. This column is primarily used for workflows i.e. you can use that choice field to setup workflow conditions to perform desired actions.
This article is to provide a quick, easy and little to no-code solution to create custom design forms for a SharePoint list or document library using only SharePoint Designer 2013. Once the custom forms have been built and set as default, it is at your discretion to delete the original three forms.

See Also

- SharePoint 2013 Portal
- SharePoint 2013 - Service Applications
- SharePoint 2013 - Resources for Developers
- SharePoint 2013 - Resources for IT Pros

SharePoint 2013: A no Code Solution to Build a Change Management Request

Brief Synopsis

A request had come in by a department to create a Change Management Request. This request is somewhat similar to a vacation (time off) request, however it does have some differences. Below is the basic outline of the request.

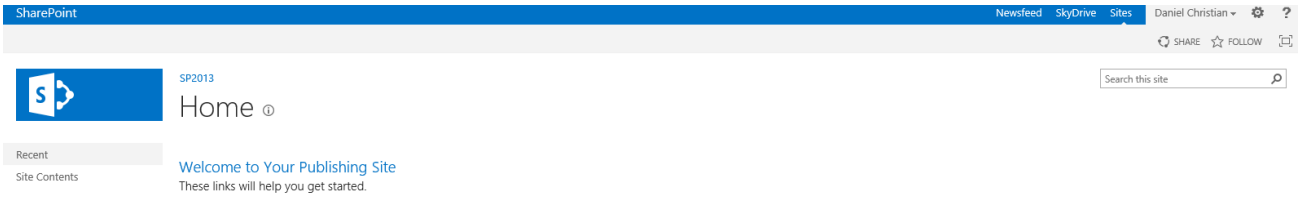
1. The users should have the ability to fill a form. This form has certain questions which the users must answer. Let's call this a 'Change Management Request form'
2. A "Management Committee" team will receive an alert which tells them that a form has been filled, submitted and waiting to be reviewed. They check to see if the criteria has been met and then forwards the request to an 'Awards committee'.
3. The "Awards Committee" can only view the questions and answers, but does not have any access to who filled the form. Also, each awards committee member needs to approve or deny this request. Once the number of approved (or denied) reaches a certain quantity, the Management Committee receives an alert.

Sounds simple enough!...well...kinda. The primary and most important criteria is that all users, including the 'Awards Committee' should be able to view the existing suggestions that have already been made, however, they should not be to see who put in those suggestions. To explain how I went about setting this up I am going to break it into six separate sections.
I am going to use a SharePoint (SP) 2013 enterprise environment with SharePoint designer 2013 and InfoPath designer 2010 to explain the steps. I am going to create three separate lists, three separate InfoPath forms and three separate workflows. One of the workflow is going to use the impersonation step. Below is the setup of my SharePoint 2013 farm.

1. SharePoint 2013 standalone environment setup on Windows 2012 and SQL 2012 SP1.
2. SharePoint 2013 Enterprise license.

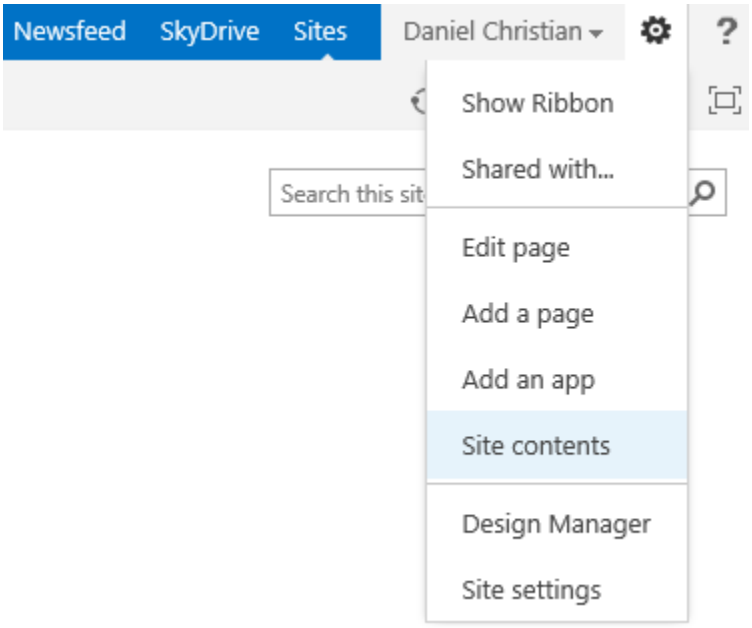
Building the three lists

I have a SharePoint (SP) 2013 publishing site already setup as shown below:

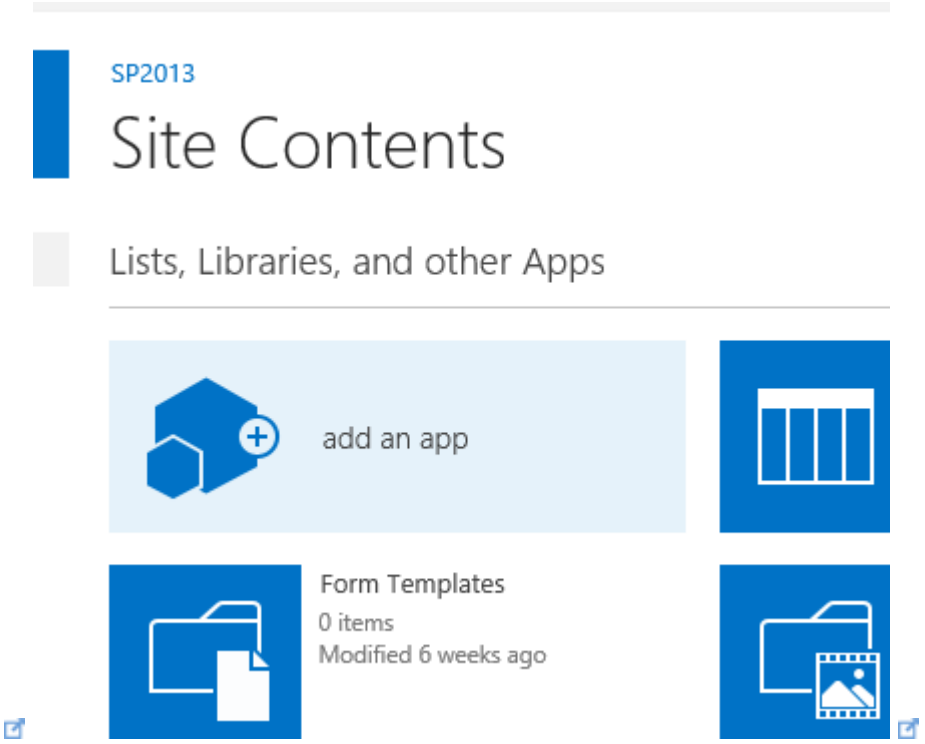


User Input List

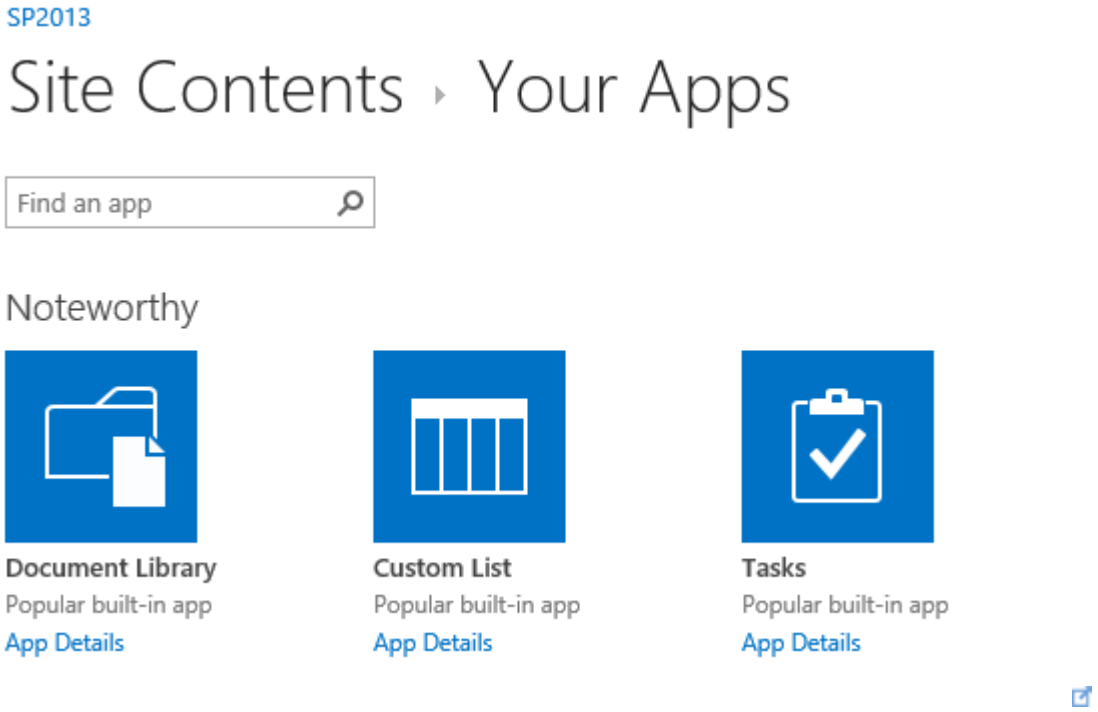
Create the first list called 'User Input'. Since we will be creating three separate lists, I am going to walk-through the process of setting up the first list in detail. You can create the remaining two lists using the same steps as shown below.
Click on the settings option and then choose 'Site contents' as shown below



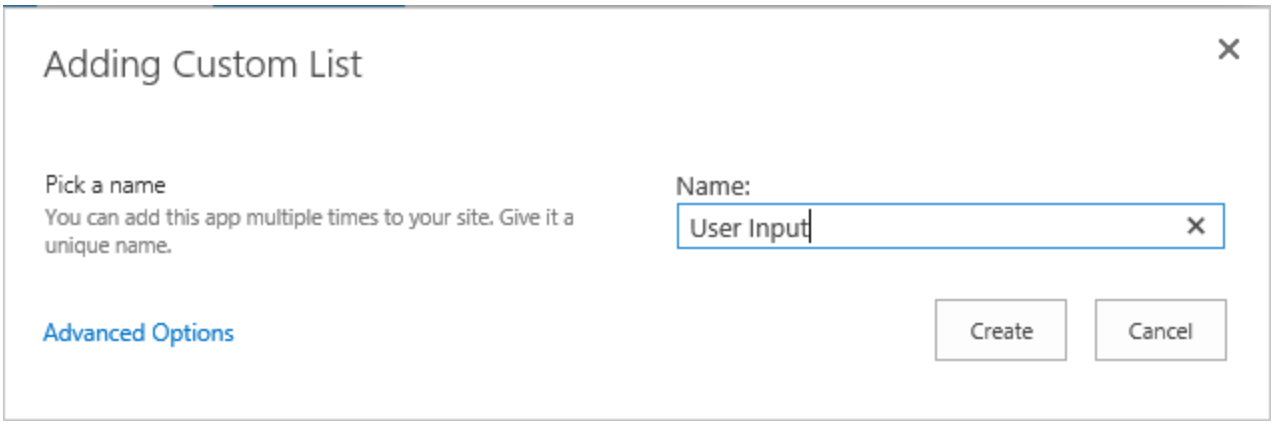
In 'Site Contents' choose 'add an app' as shown below:



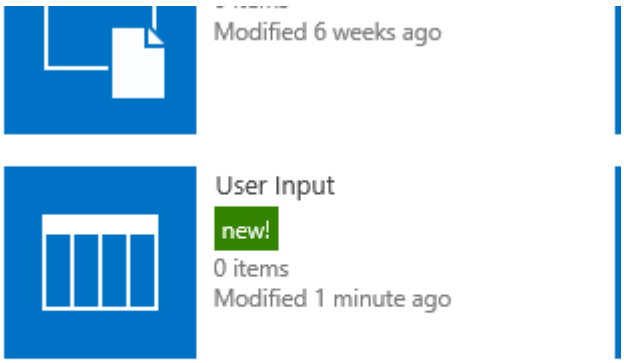
In 'Your Apps' choose 'Custom List'

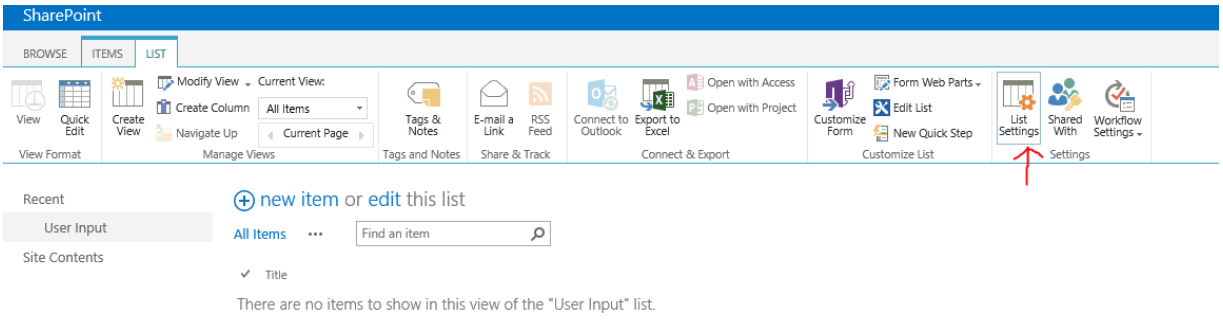


Give your custom list a name. In my case I have named it as 'User Input':

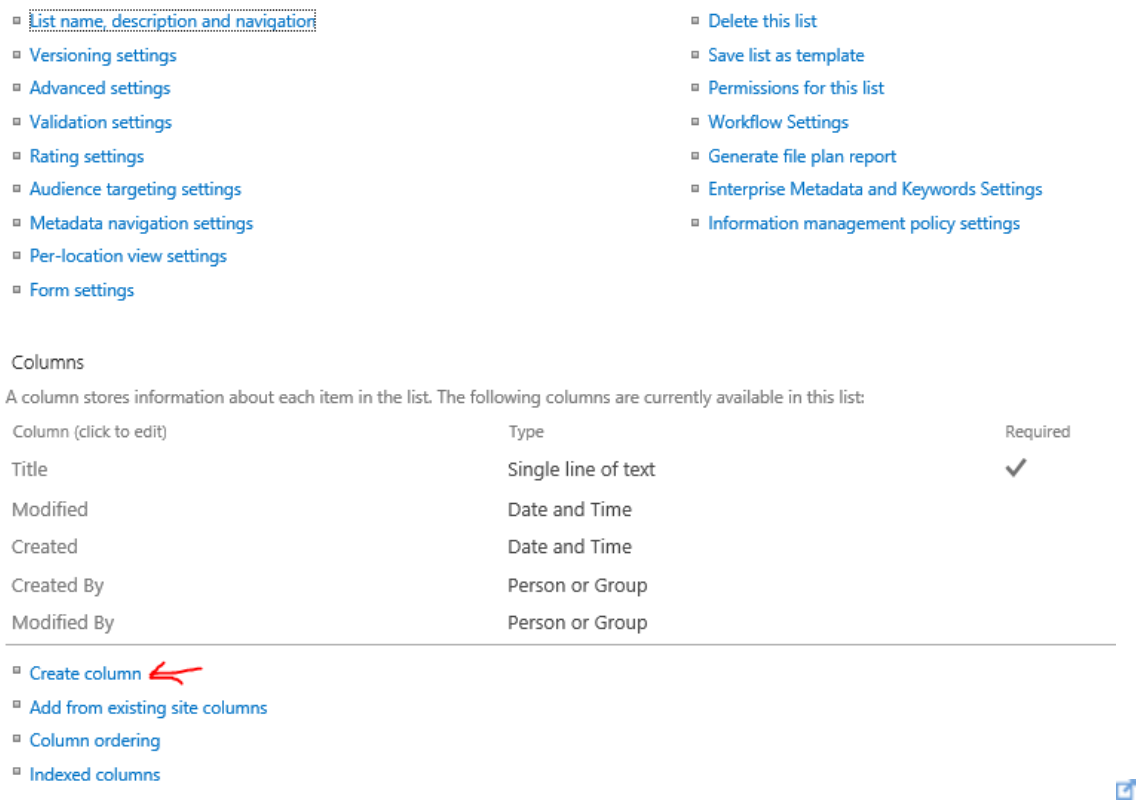


You will now see the new list created. Click on it and then access the List Settings option.





Now we will be creating columns needed. Choose the 'Create column' option as shown below:



Below are the lists of columns which we will need. Some of these columns are type Choice and hence I have provided the choices as well. When you create a list by default the Title column will be created. Rename the Title column to as Suggestion Topic. It is important that this change is carried out because the Title column also has a hyperlink built into it.

- List name, description and navigation
- Versioning settings
- Advanced settings
- Validation settings
- Rating settings
- Audience targeting settings
- Metadata navigation settings
- Per-location view settings
- Form settings
- Delete this list
- Save list as template
- Permissions for this list
- Workflow Settings
- Generate file plan report
- Enterprise Metadata and Keywords Settings
- Information management policy settings

Columns

A column stores information about each item in the list. The following columns are currently available in this list:

Column (click to edit)	Type	Required
Title	Single line of text	✓
Action	Choice	
Requester Name	Person or Group	
Suggestion Topic	Single line of text	✓
Description	Multiple lines of text	✓
Solution	Multiple lines of text	✓
Result	Multiple lines of text	✓
State	Choice	
Modified	Date and Time	
Created	Date and Time	
Created By	Person or Group	
Modified By	Person or Group	

The Action column has the following choices.

SU - Submitted by User
AMT - Approved by Management Team
RMT - Rejected by Management Team
AAC - Approved by Awards Committee
RAC - Rejected by Awards Committee

Name and Type

Type a name for this column.

Column name:

Action

The type of information in this column is:

- ○ Single line of text
- ○ Multiple lines of text
- ◉ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

SU - Submitted by User
AMT - Approved by Management Team

Require that this column contains information:

○ Yes ◉ No

Enforce unique values:

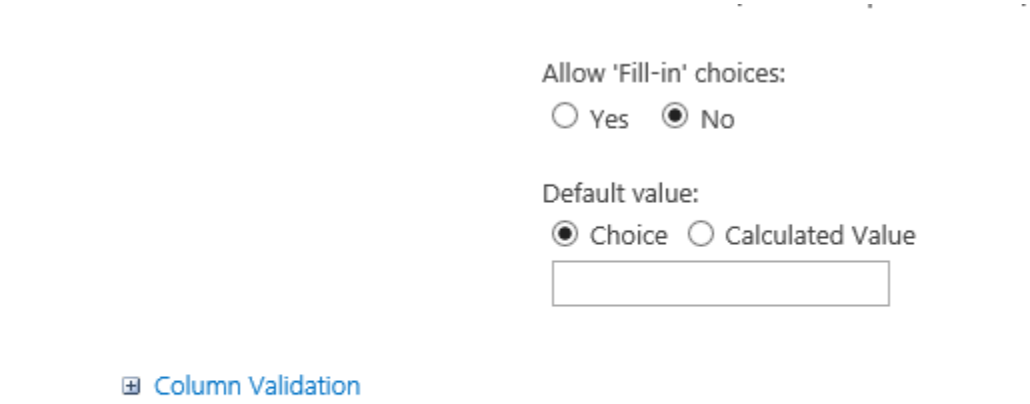
○ Yes ◉ No

Type each choice on a separate line:

SU
AMT
RMT
AAC
RAC

Display choices using:

- ◉ Drop-Down Menu
- ○ Radio Buttons
- ○ Checkboxes (allow multiple selections)



The State column has the following choices.

WMT - Waiting on Management Team
WAC - Waiting on Awards Committee
DONE

Settings ▸ Edit Column

Name and Type

Type a name for this column.

Column name:

State

The type of information in this column is:

- ○ Single line of text
- ○ Multiple lines of text
- ◉ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

WMT - Waiting on Management Team
WAC - Waiting on Awards Committee

Require that this column contains information:

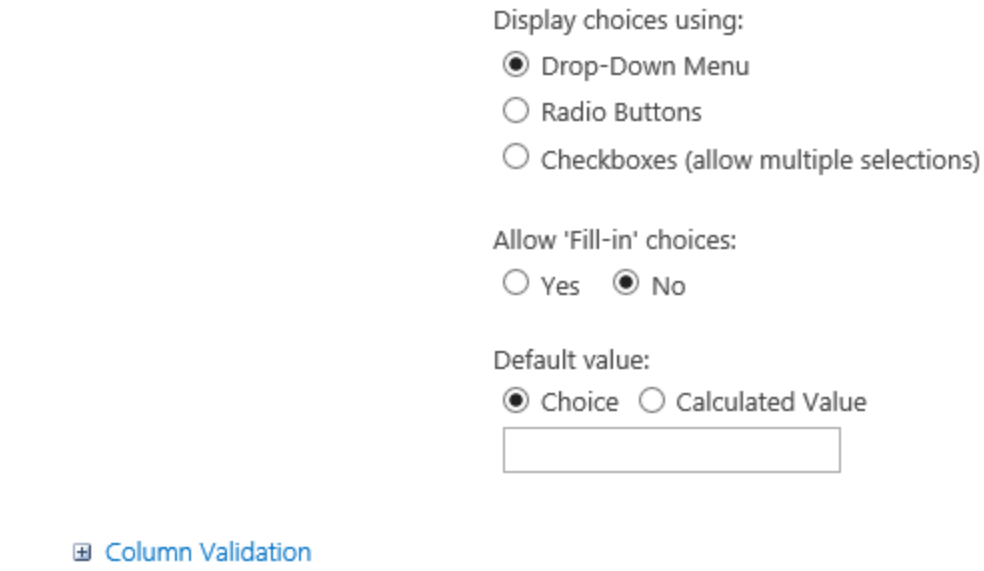
○ Yes ◉ No

Enforce unique values:

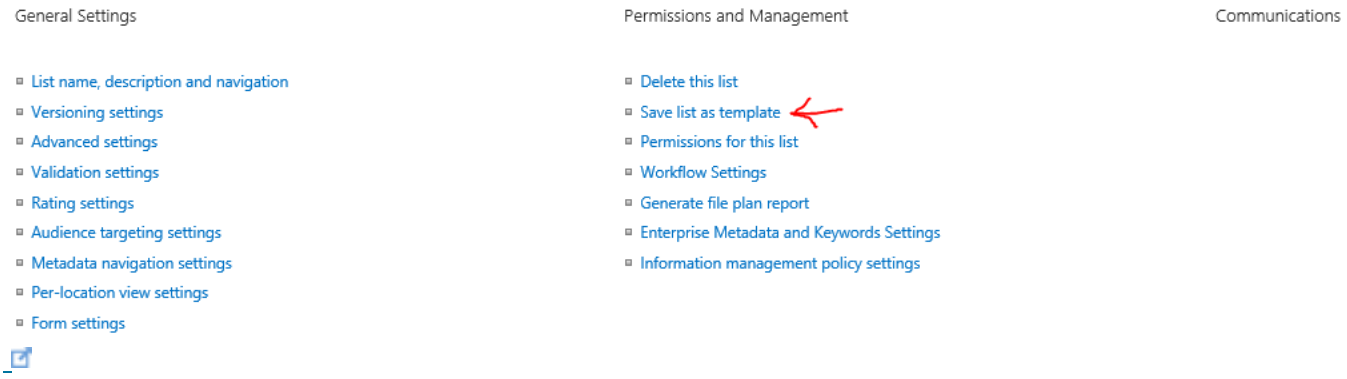
○ Yes ◉ No

Type each choice on a separate line:

WMT
WAC
DONE



We will be reusing this list again so let's save it as a template. In the List Settings page choose the 'Save list as template option' as shown below:



Settings ▸ Save as Template

File Name

Enter the name for this template file.

File name:

ChangeManagement

Name and Description

The name and description of this template will be displayed on the Create page.

Template name:

ChangeManagement

Template description:

Include Content

Include content in your template if you want new lists created from this template to include the items in this list. Including content can increase the size of your template.

Caution: Item security is not maintained in a template. If you have private content in this list, enabling this option is not recommended.

☐ Include Content

SP2013 ▸ Operation Completed Successfully

The template has successfully been saved to the list template gallery. You can now create lists based on this template.

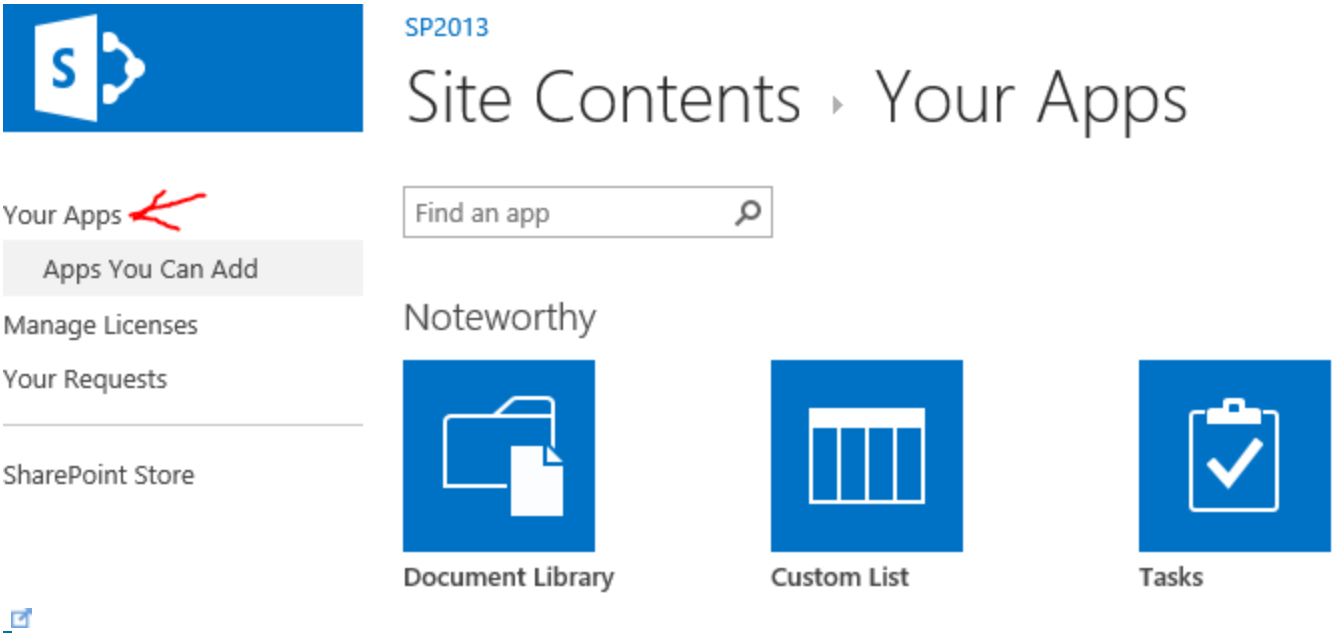
To manage templates in the gallery, go to the list template gallery.

To return to the list customization page, click **OK**.

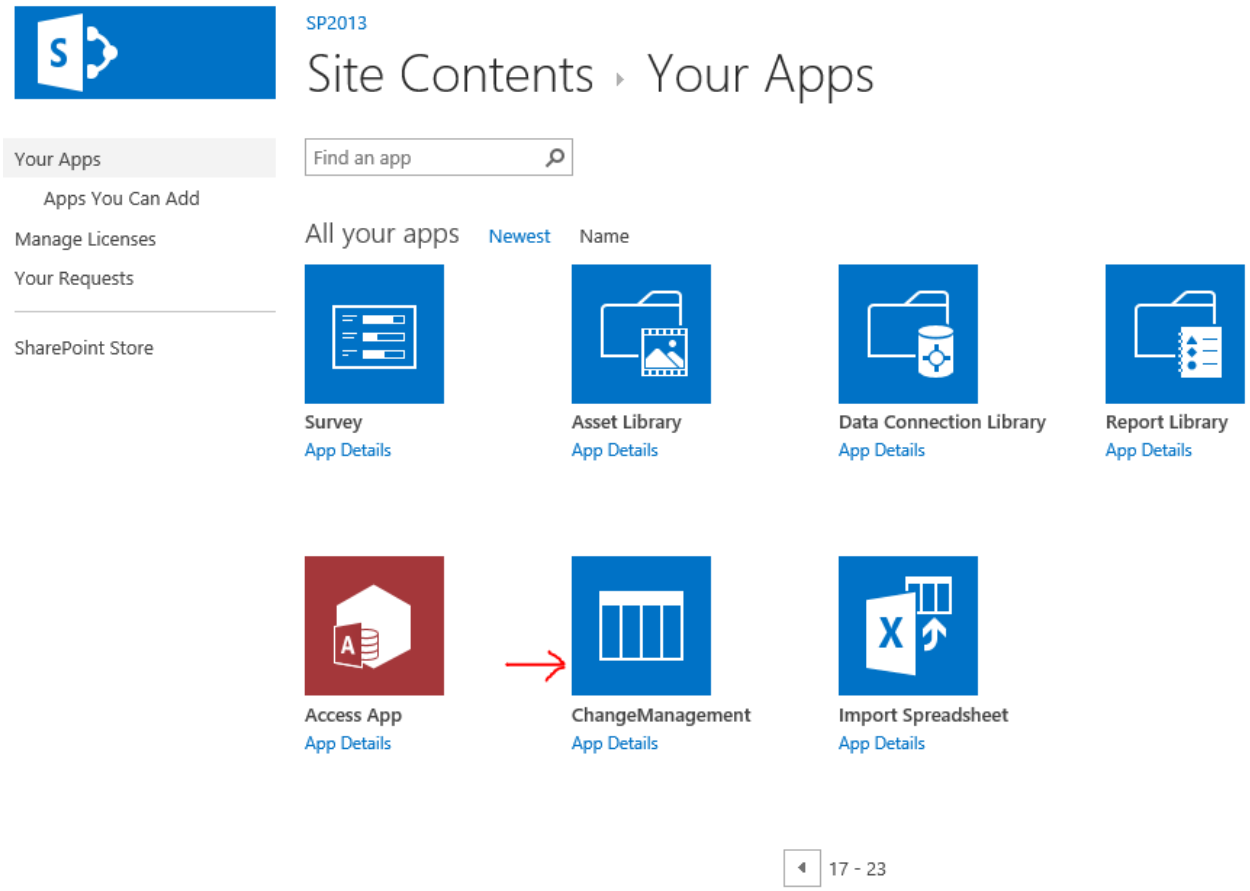
OK

Management Committee Input List

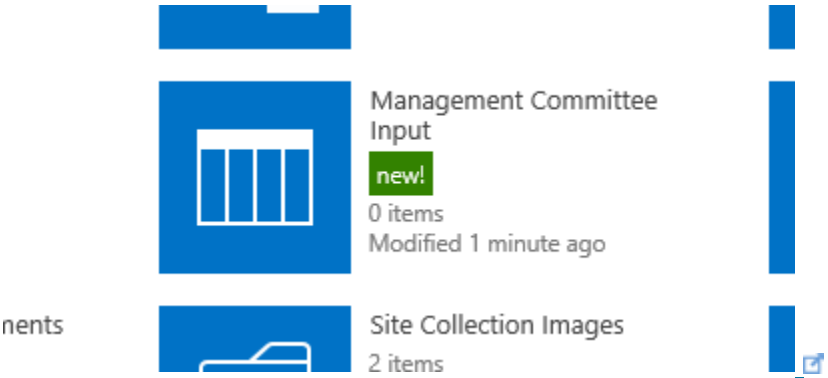
Now, we are going to create our second list, however, this time we are going to use the template that we saved above.
Follow the steps provided above to 'Add an app'. By default, the 'Apps You Can Add' option is selected; instead, choose the 'Your Apps' option as shown below:

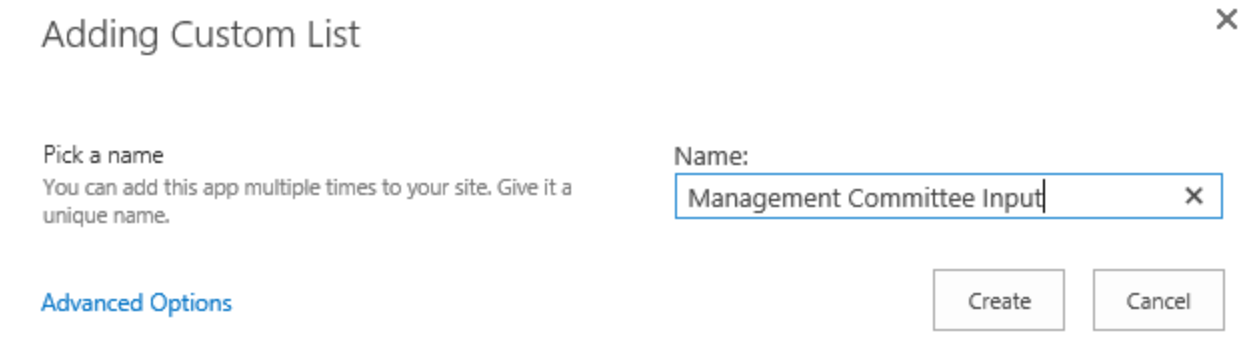


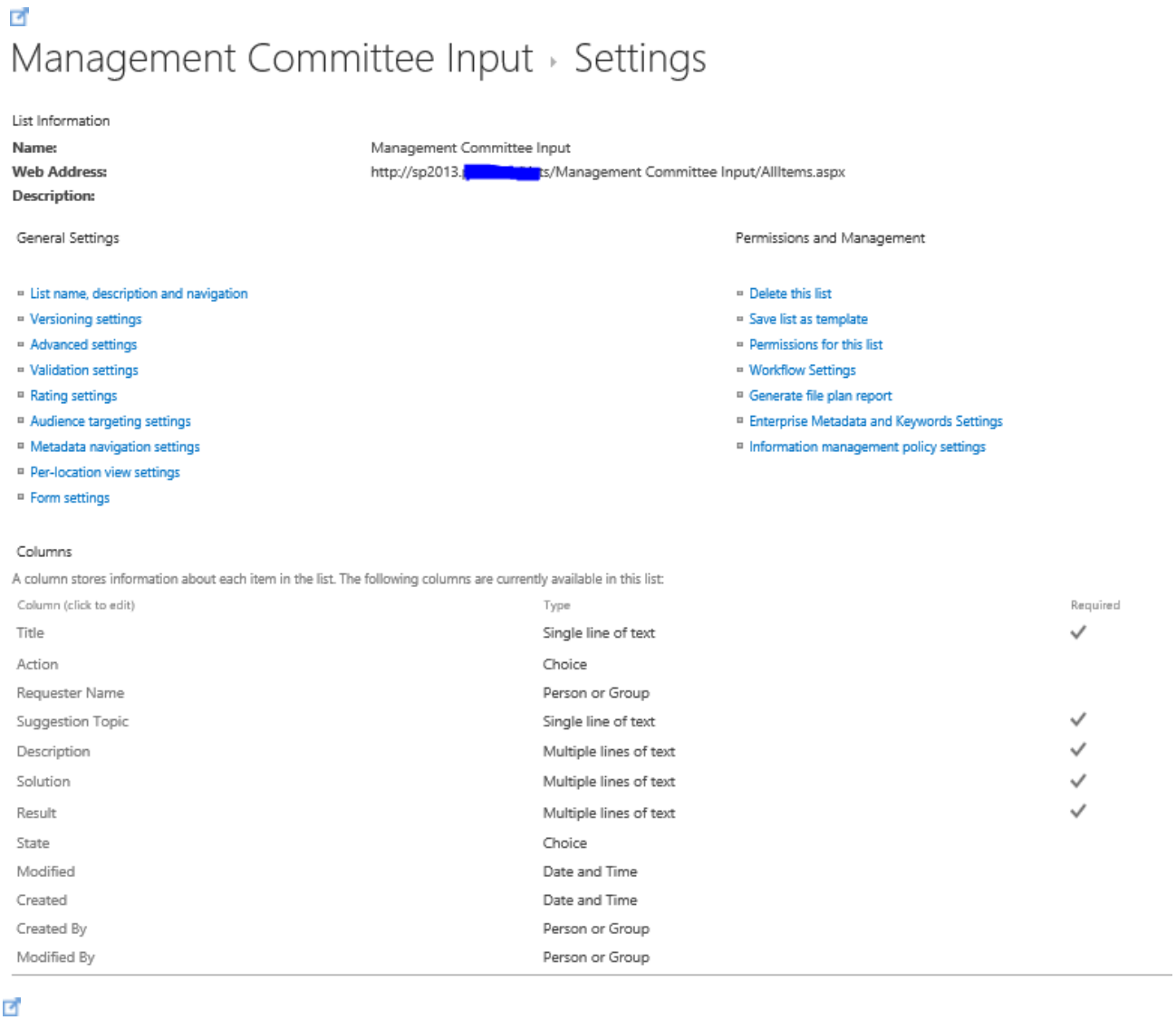
Now you should be able to find the new 'ChangeManagement' template we saved.



Click on the 'ChangeManagement' app and give it a name. In my case I have named it as 'Management Committee Input'. Now you should see the new list you have created which will contain the columns we built before.







In this list we are going to add a column. This will be a Multiple lines of text column called 'Management Team Comments' as shown below:

SP2013

Settings ▸ Create Column

Name and Type

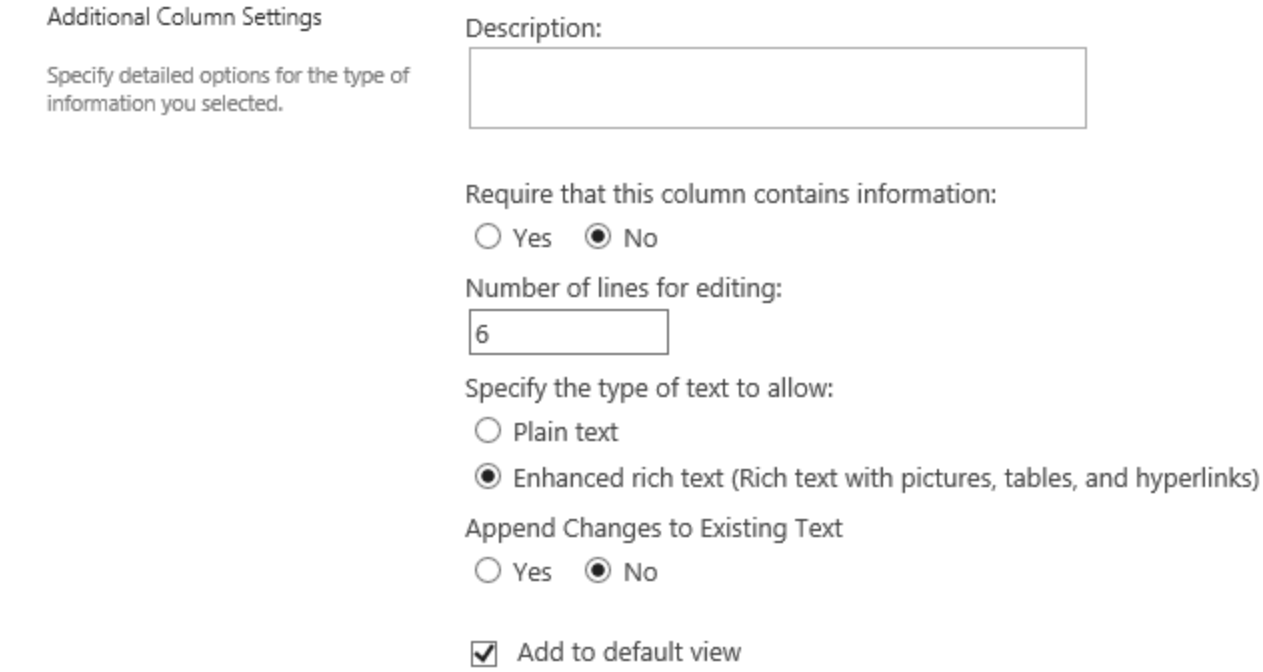
Type a name for this column, and select the type of information you want to store in the column.

Column name:

Management Team Comments

The type of information in this column is:

- ○ Single line of text
- ◉ Multiple lines of text
- ○ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time
- ○ Lookup (information already on this site)
- ○ Yes/No (check box)
- ○ Person or Group
- ○ Hyperlink or Picture
- ○ Calculated (calculation based on other columns)
- ○ Task Outcome
- ○ External Data
- ○ Managed Metadata



Awards Committee List

Now we are going to build our third and final list. Build the list in the same manner as the second list. The third list is called 'Awards Committee'
This list will be used to store the information of the Awards Committee. Each Awards committee member is going to have to choose a Yes or a No and leave a comment. We also need to a State column for each and every Awards committee. Below are the three columns which need to be created for every Awards committee member.

User1	Choice
User1-State	Choice
User1-Comment	Multiple lines of text

Below are the screen shots for the User1 column

Name and Type

Type a name for this column.

Column name:

User1

The type of information in this column is:

- ○ Single line of text
- ○ Multiple lines of text
- ◉ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

Require that this column contains information:

○ Yes ◉ No

Enforce unique values:

○ Yes ◉ No

Type each choice on a separate line:

No
Yes

Display choices using:

- (•) Drop-Down Menu
- ( ) Radio Buttons
- ( ) Checkboxes (allow multiple selections)

Allow 'Fill-in' choices:

- ( ) Yes (•) No

Default value:

- (•) Choice ( ) Calculated Value

Below are screenshots of the User1-State column:

Name and Type

Type a name for this column.

Column name:

User1-State

The type of information in this column is:

- ○ Single line of text
- ○ Multiple lines of text
- ◉ Choice (menu to choose from)
- ○ Number (1, 1.0, 100)
- ○ Currency ($, ¥, €)
- ○ Date and Time

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

Require that this column contains information:

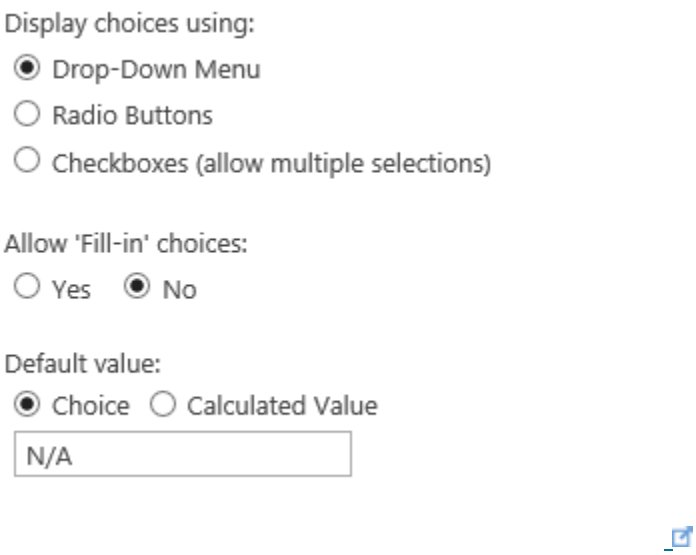
○ Yes ◉ No

Enforce unique values:

○ Yes ◉ No

Type each choice on a separate line:

N/A
Approved
Denied



Below is a screenshot of the User1-Comment column:

Name and Type

Type a name for this column.

Column name:

User1-Comment

The type of information in this column is:

Multiple lines of text

Additional Column Settings

Specify detailed options for the type of information you selected.

Description:

Require that this column contains information:

Yes No

Number of lines for editing:

6

Specify the type of text to allow:

Plain text

Rich text (Bold, italics, text alignment, hyperlinks)

Enhanced rich text (Rich text with pictures, tables, and hyperlinks)

Append Changes to Existing Text

Yes No

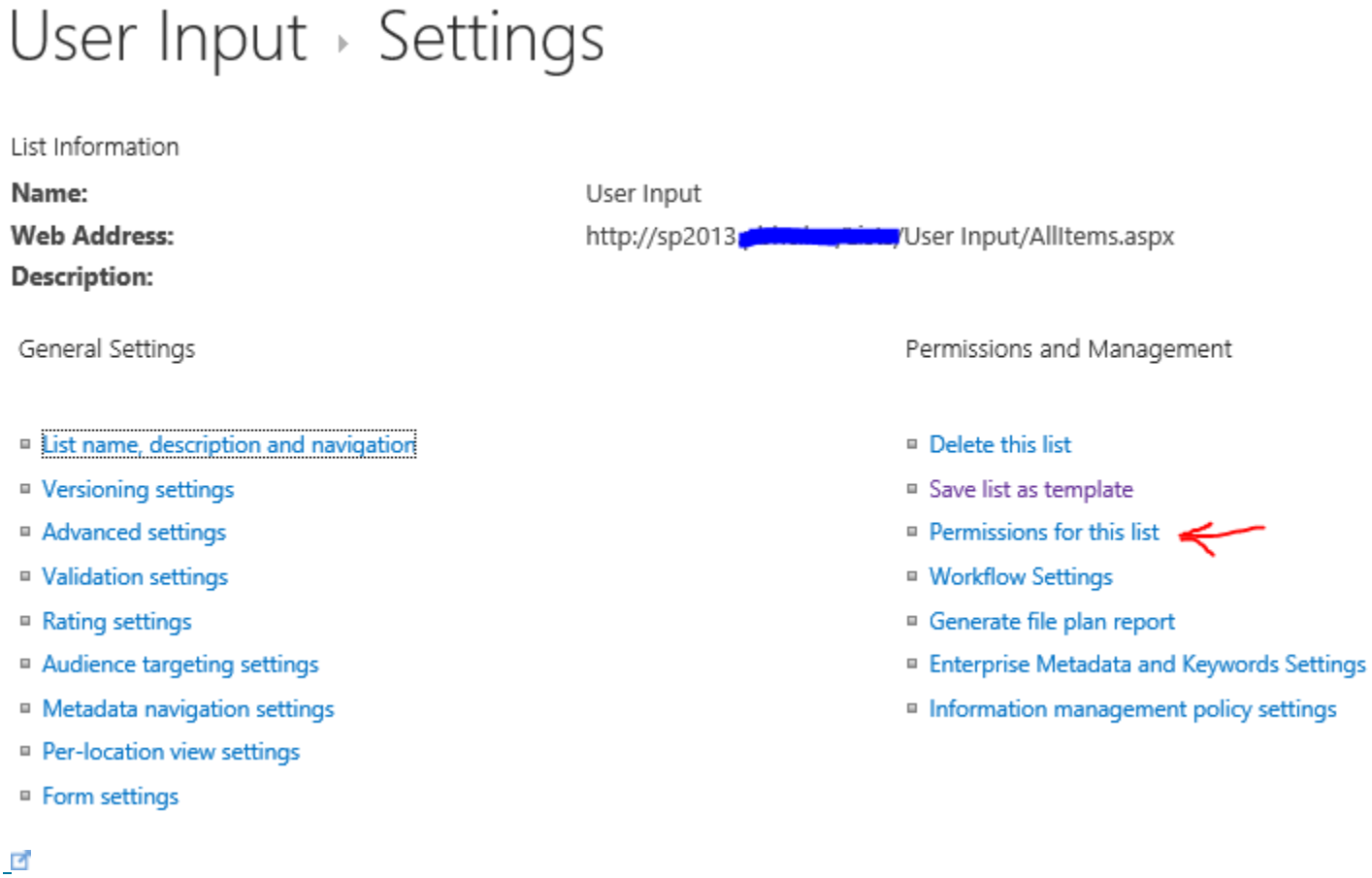
You can add as many Awards committee members as you need. I have created a total of 5. Below is a screenshot of all my columns.

Solution	Multiple lines of text
Result	Multiple lines of text
State	Choice
Approved Number	Number
Denied Number	Number
User1	Choice
User1-State	Choice
User1-Comment	Multiple lines of text
User2	Choice
User2-State	Choice
User2-Comment	Multiple lines of text
User3	Choice
User3-State	Choice
User3-Comment	Multiple lines of text
User4	Choice
User4-State	Choice
User4-Comment	Multiple lines of text
User6	Choice
User6-State	Choice
User6-Comment	Multiple lines of text
Modified	Date and Time

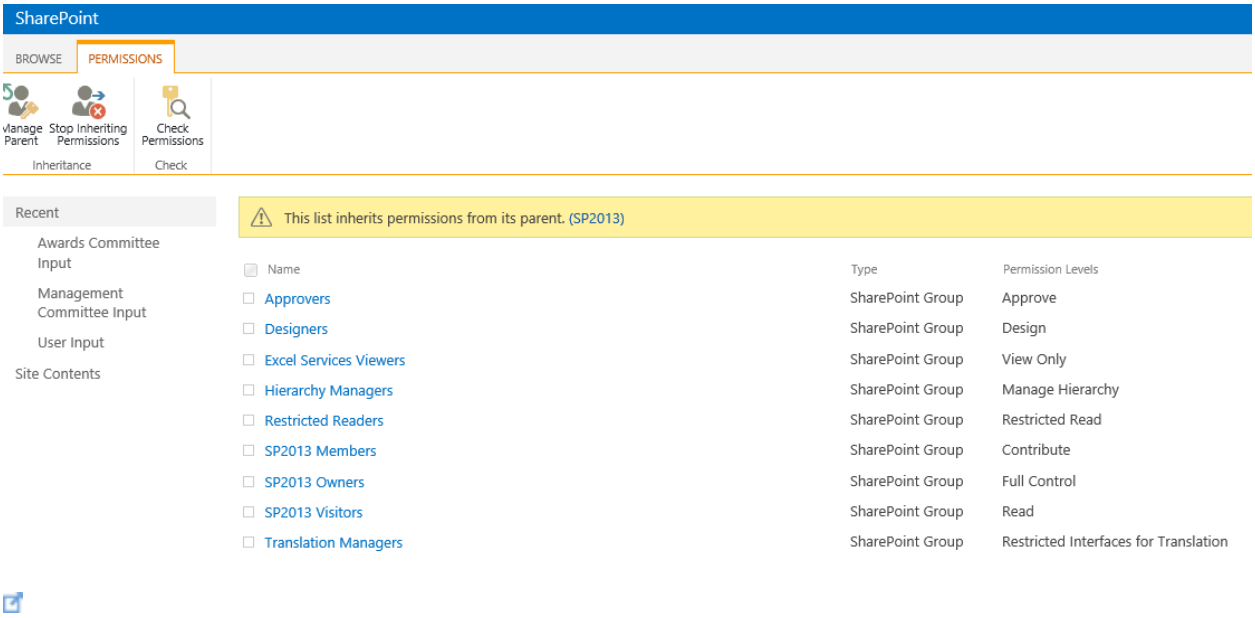
Setting up the permissions

As I had mentioned before, the primary and most important criteria is that all users, including the 'Awards Committee' should be able to view the existing suggestions that have already been made, however, they should not be to see who put in those suggestions. Hence the permissions inheritance needs to be broken and specific permissions needs to be setup for each list. Since we will be changing the permissions of three separate lists, I am going to walk-through the process of setting up the permissions of the first list in detail. You can setup the permissions of the remaining two lists in the same manner.

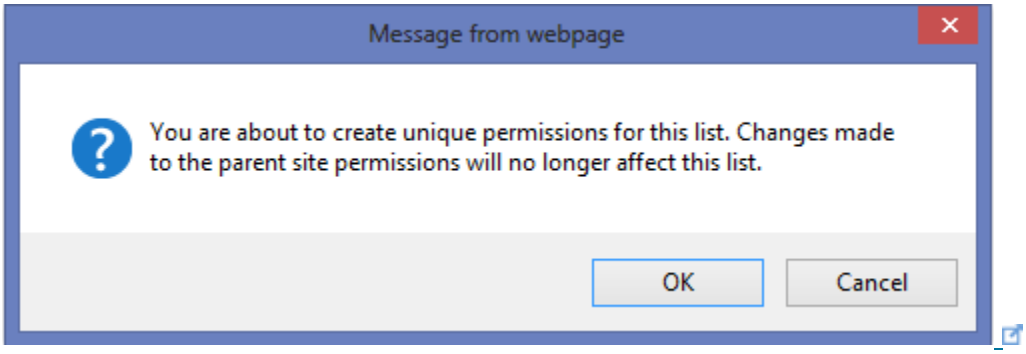
User Input list permissions
Go to the List Settings for the 'User Input' list and click on the 'Permissions for this list' as shown below:



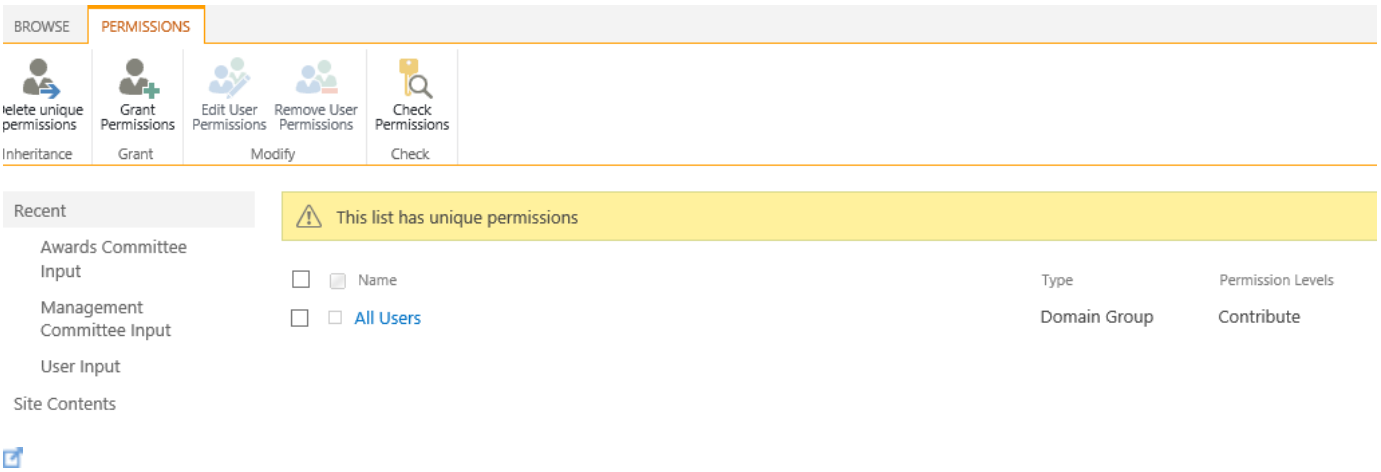
Below is the default permissions settings. As you can see, currently this list is inheriting the permissions from its parent.



Click on the 'Stop Inheriting Permissions' option. You will receive the below message, click OK.

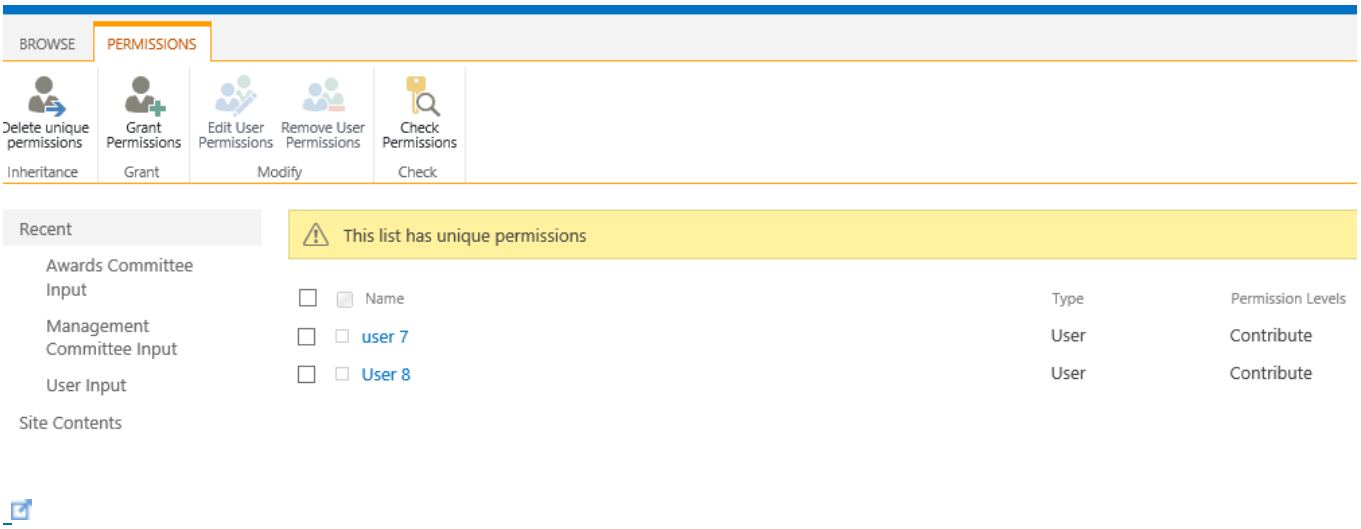


For the sake of simplicity, I am going to delete all the existing groups I have and add the individual users/groups only. Since all users will need access to fill the form, I have given All Users Contribute level permission for this list. Below is a screenshot which also shows that this list now has unique permissions.



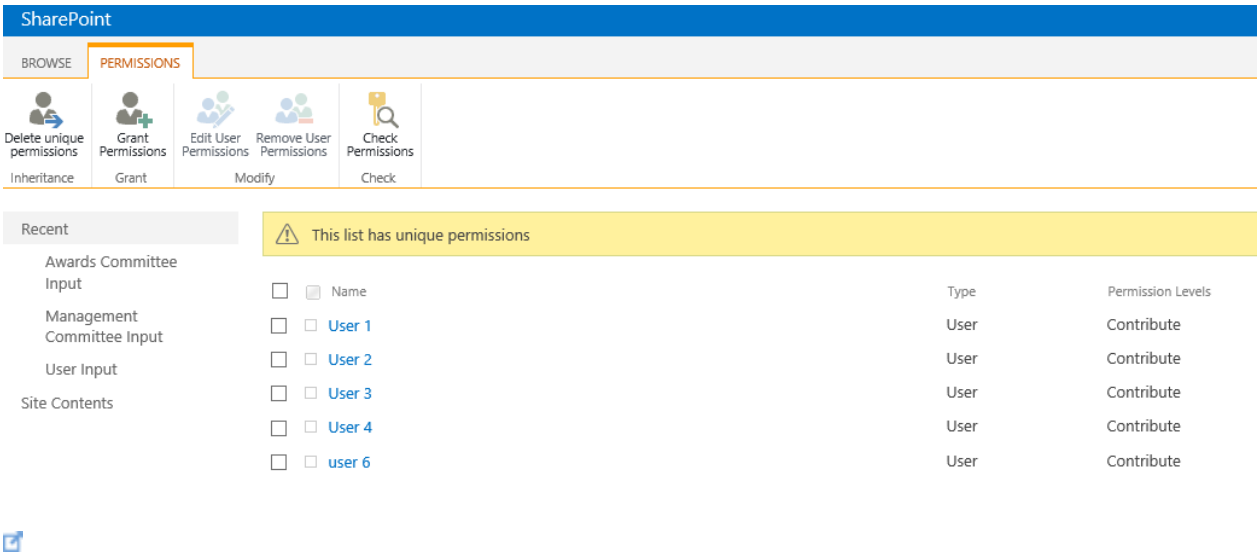
Management Committee Input list permissions

Below is a screenshot of the permission for the 'Management Committee Input' list. User7 and User8 are the two members of this committee and hence I have given them contribute access.



Awards Committee Input list permissions

Below is a screenshot of the permission for the 'Awards Committee Input' list. User1, User2, User3, User4 & User6 are the members of this committee and hence I have given them contributed access.

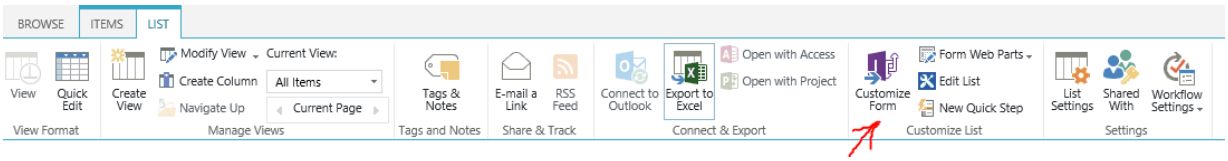


Building the InfoPath forms

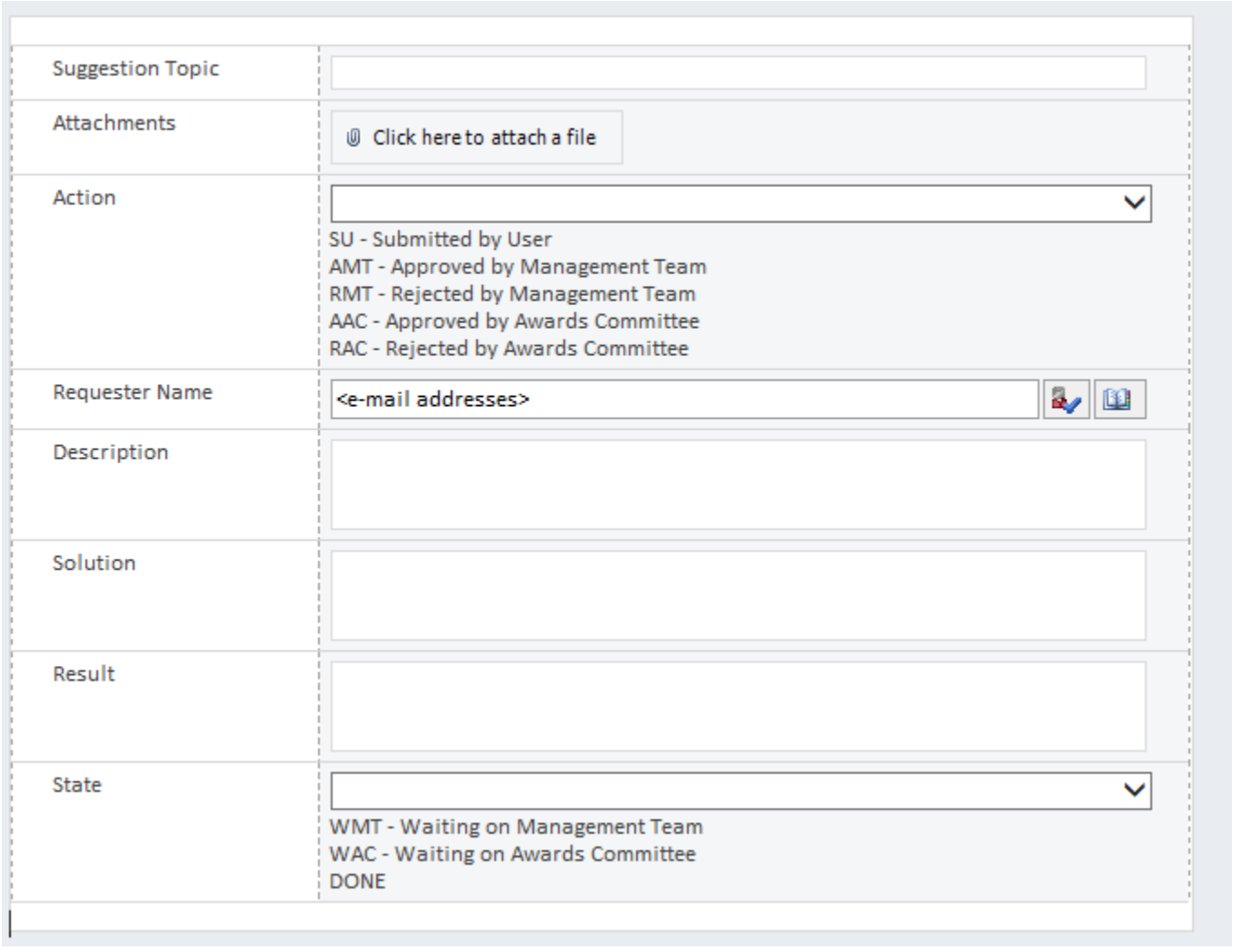
I will be spending more time explaining step-by-step how to create the forms using InfoPath designer since it plays an important role in making the workflows work successfully. This section comprises of three separate forms for each list we created before. I have provided written steps to create these forms and also videos for each. As a reminder, you must have SharePoint 2013 enterprise level license to use the embedded forms features.

Building the User Input form

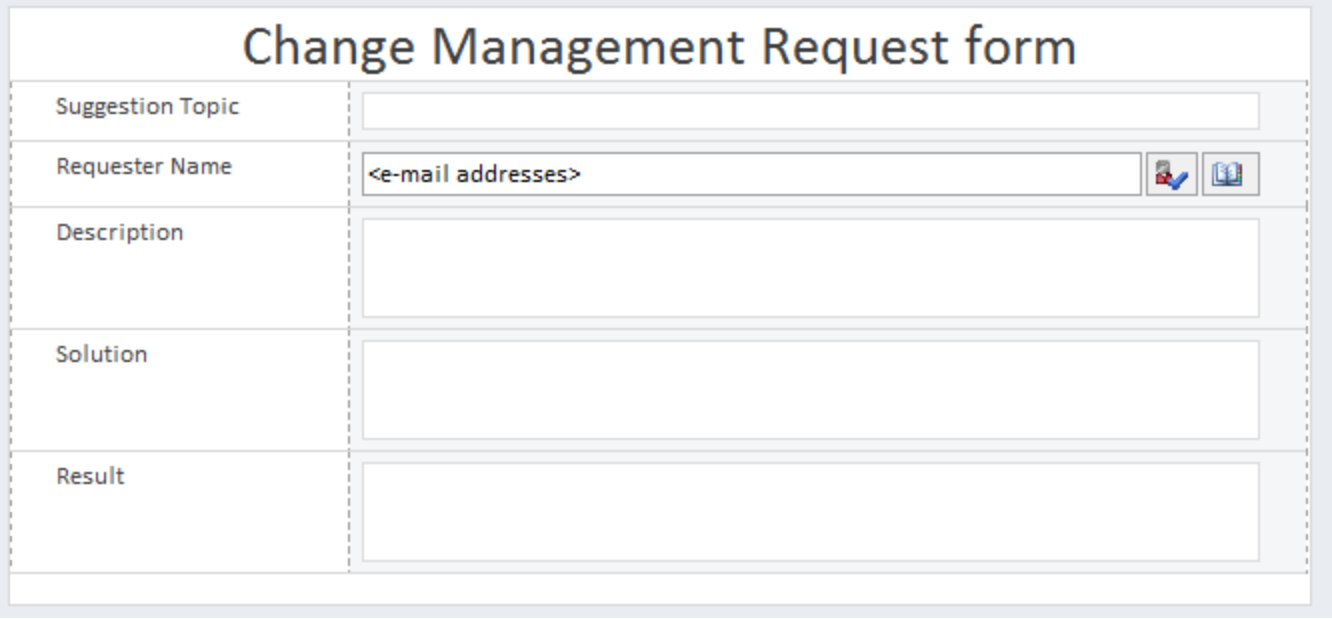
Go to the User Input list and choose the Customize Form option (as shown below) which is on the top ribbon.



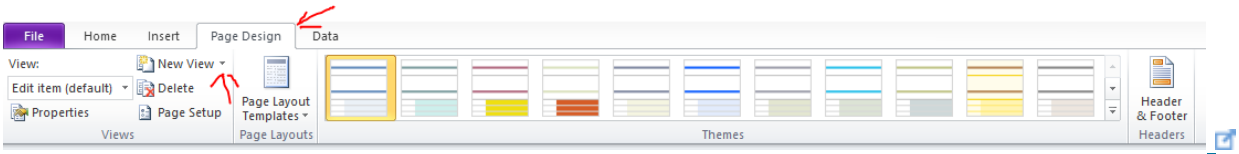
When the form is first created on InfoPath Designer, it contains all of the fields that were created in the User Input list as shown below.



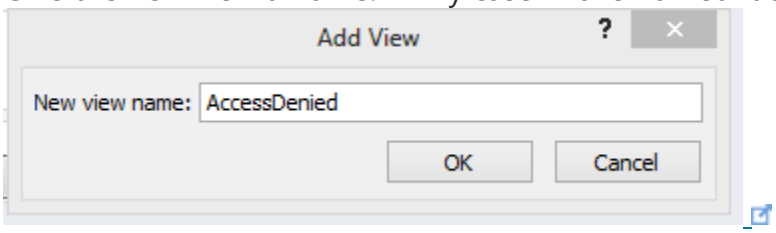
For the initial Change Management Request form, we need only need the Suggestion Topic, Requester Name, Description, Solution and Result columns. The rest can be removed from the form. You can also add a title to the form to signify its use as shown below.



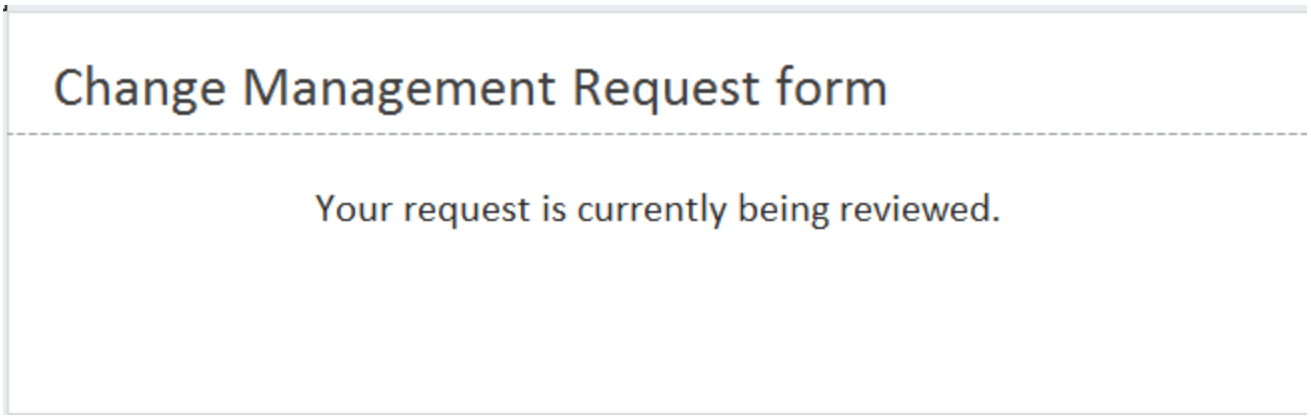
On the horizontal ribbon, choose Page Design and then click on New view as shown below.



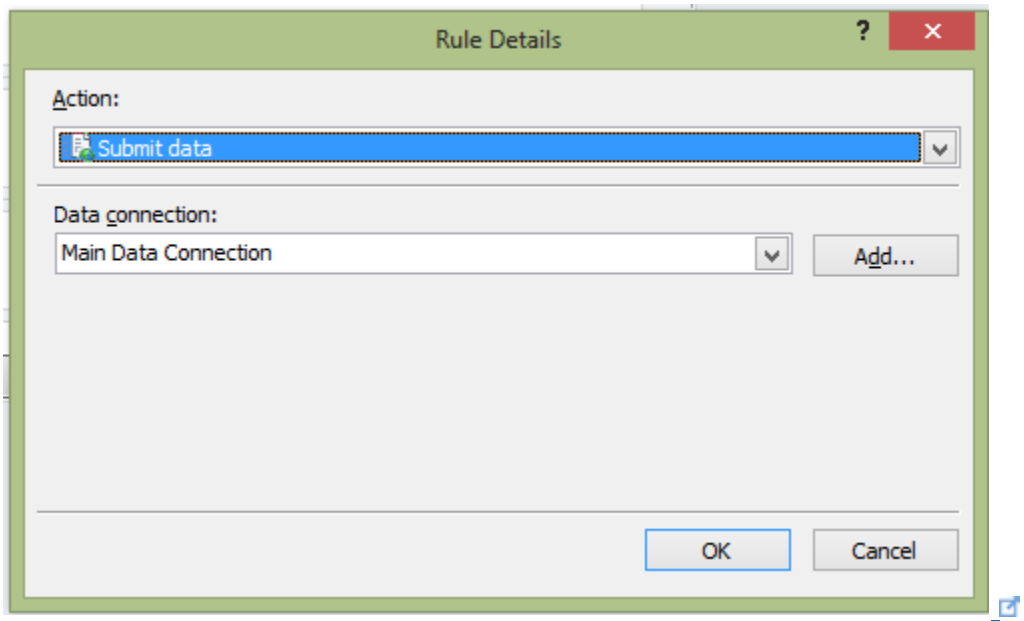
Give the new view a name. In my case I have named it as 'AccessDenied'



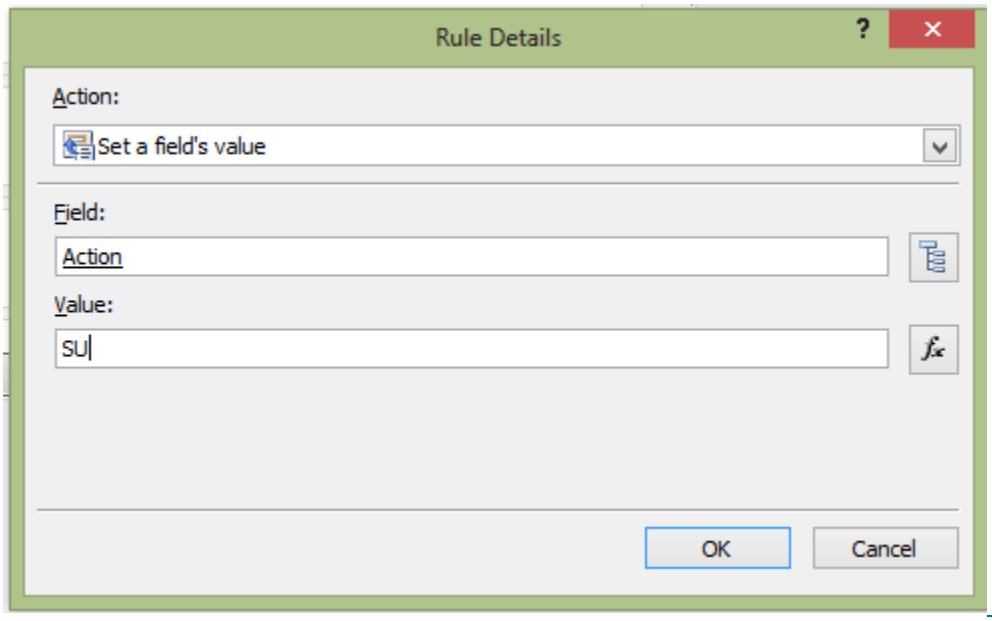
Add some text similar to what I have shown below:



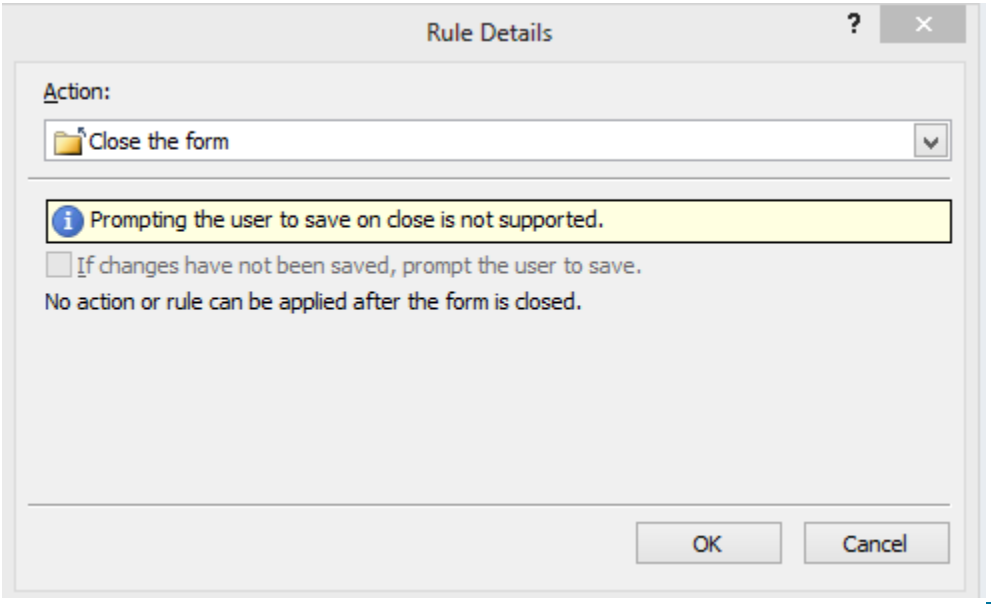
Go back to the Edit Item view and add two button control objects to the form. Name one as Submit and the other as Cancel.
For the Submit button add an Action rule which runs the following actions-
Submit Data



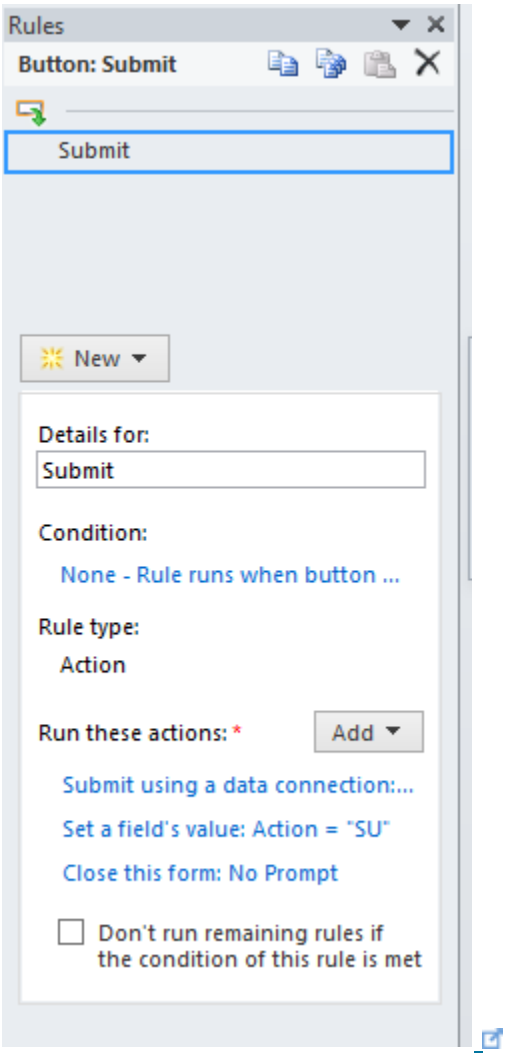
Set the Action to SU



Close the form

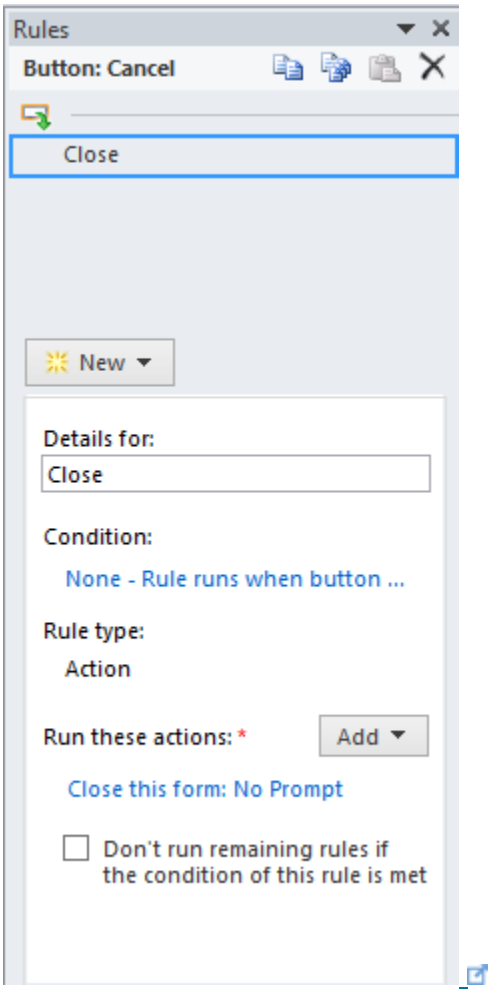


Below is what the Rule should look like

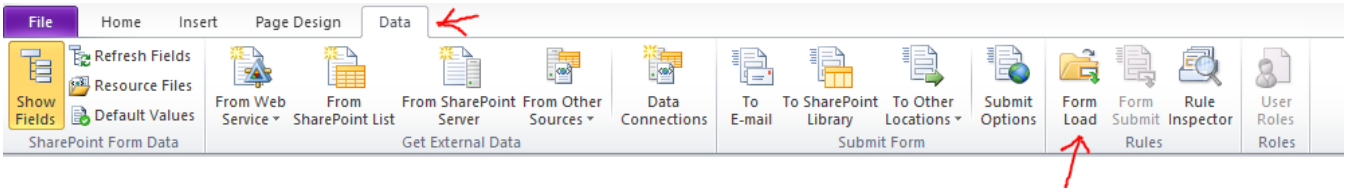


For the Cancel button add the Close This form action rule. Below is what the rule should

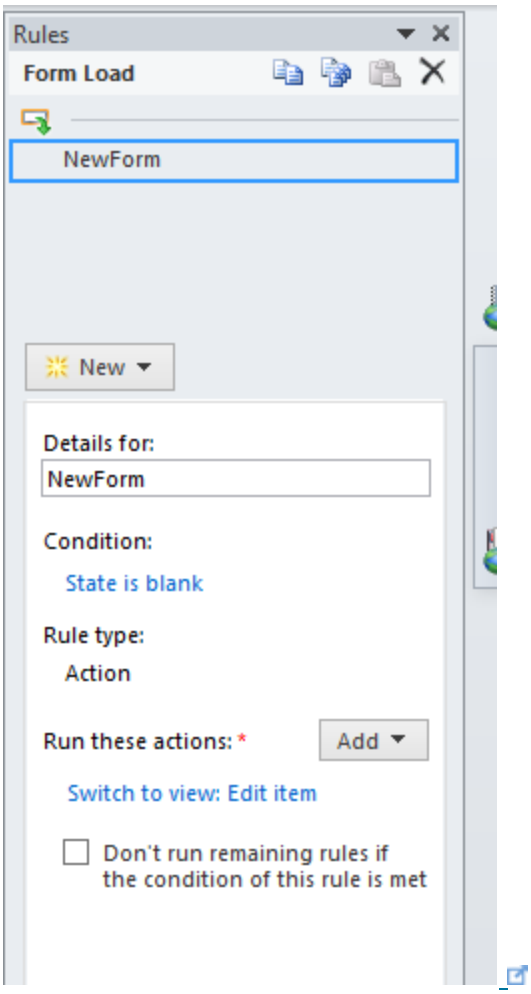
look like.



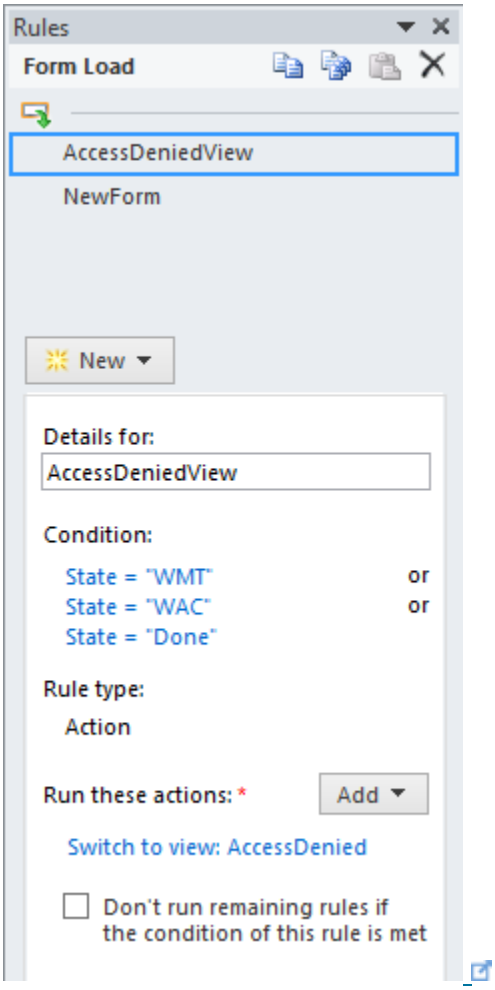
On the horizontal ribbon, choose Data and then click on Form Load as shown below



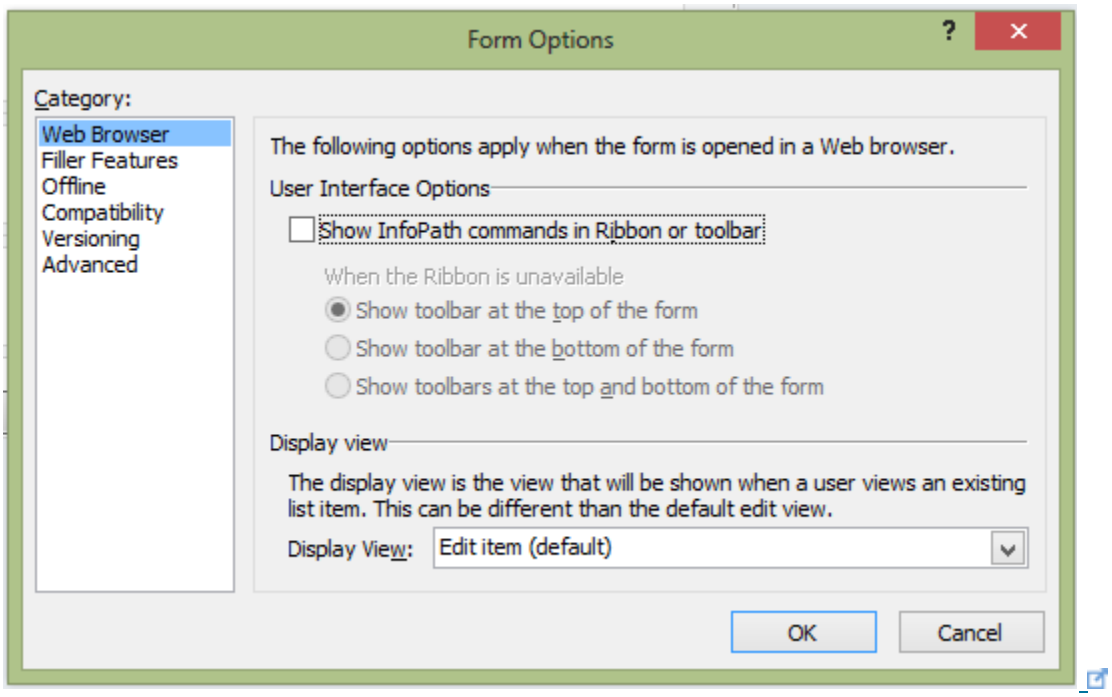
Add the following two rules.
NewForm rule as shown below


Rules
Form Load
NewForm
New
Details for:
NewForm
Condition:
State is blank
Rule type:
Action
Run these actions: *
Add
Switch to view: Edit item
Don't run remaining rules if the condition of this rule is met

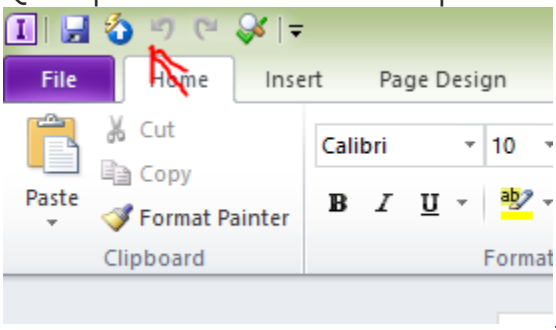
An AccessDeniedview as shown below



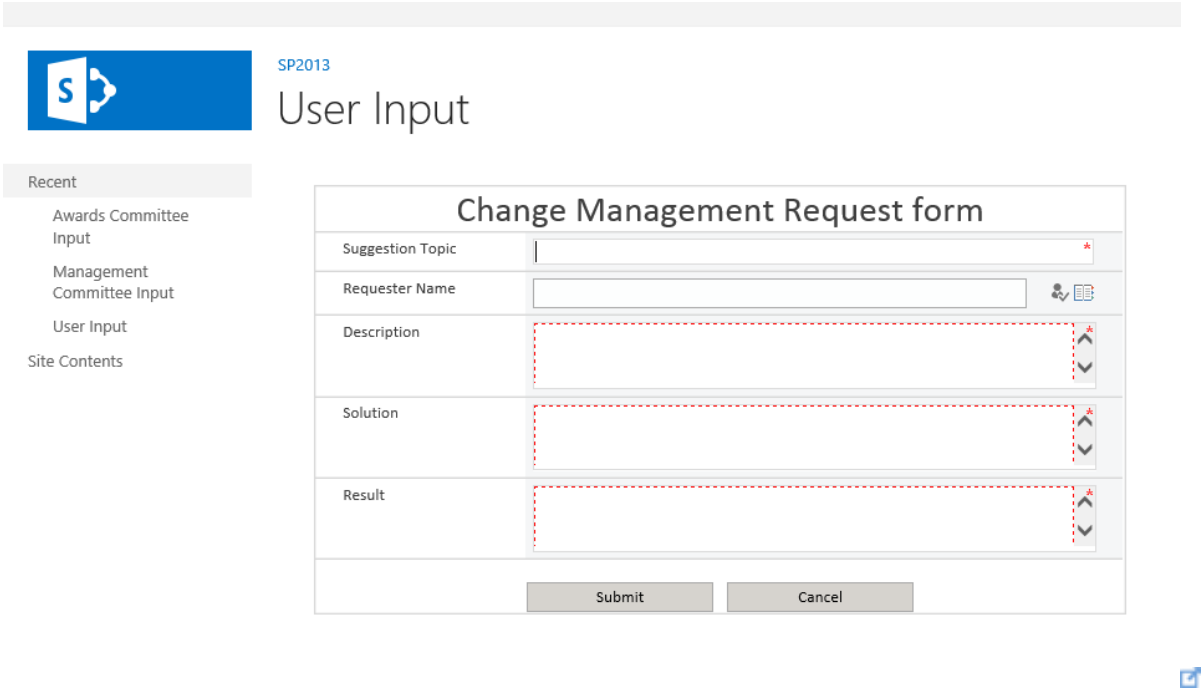
Go to File>Form Options and UNCHECK the Show InfoPath commands in Ribbon or toolbar as shown below. Confirm that Edit Item (default) is the Display view. This will prevent the end user from switching back from the AccessDeniedview to the Edit view.



You can now publish the page by either going to File>Quick Publish or simply using the Quick publish button on the top left corner.



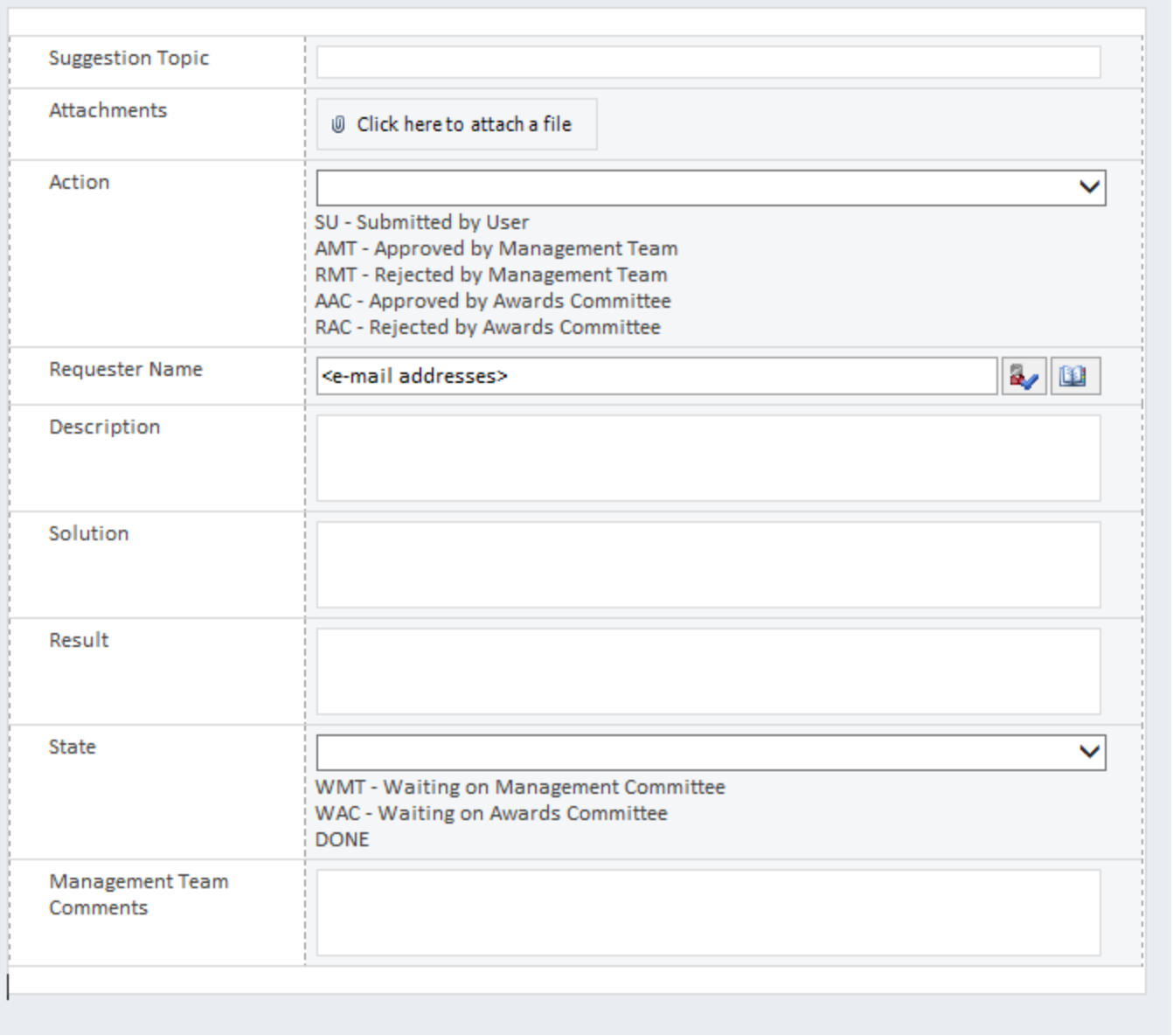
Now when you go back to your User Input list on SharePoint and choose New Item, below is the embedded form you should see.



Below is a video which walks you through the process described above to build the embedded form for Change Management Request form? Click Here to view the same video in full size.

Building the Management Committee Input form

As we did before for the User Input form, go to the Management Committee Input list and choose the Customize Form option.
When the form is first created on InfoPath Designer, it contains all of the fields that were created in the Change Management Request list as shown below.

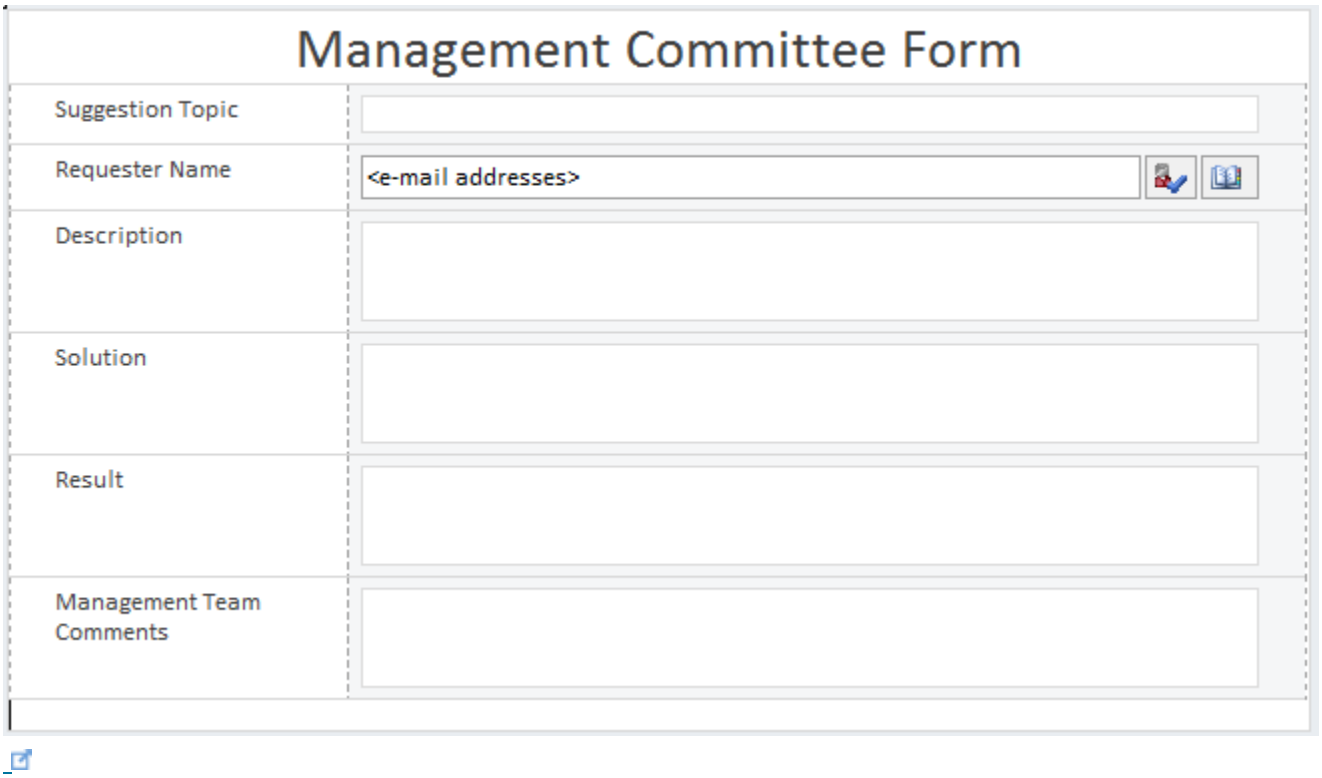


In this form there are a few new features we are going to add.

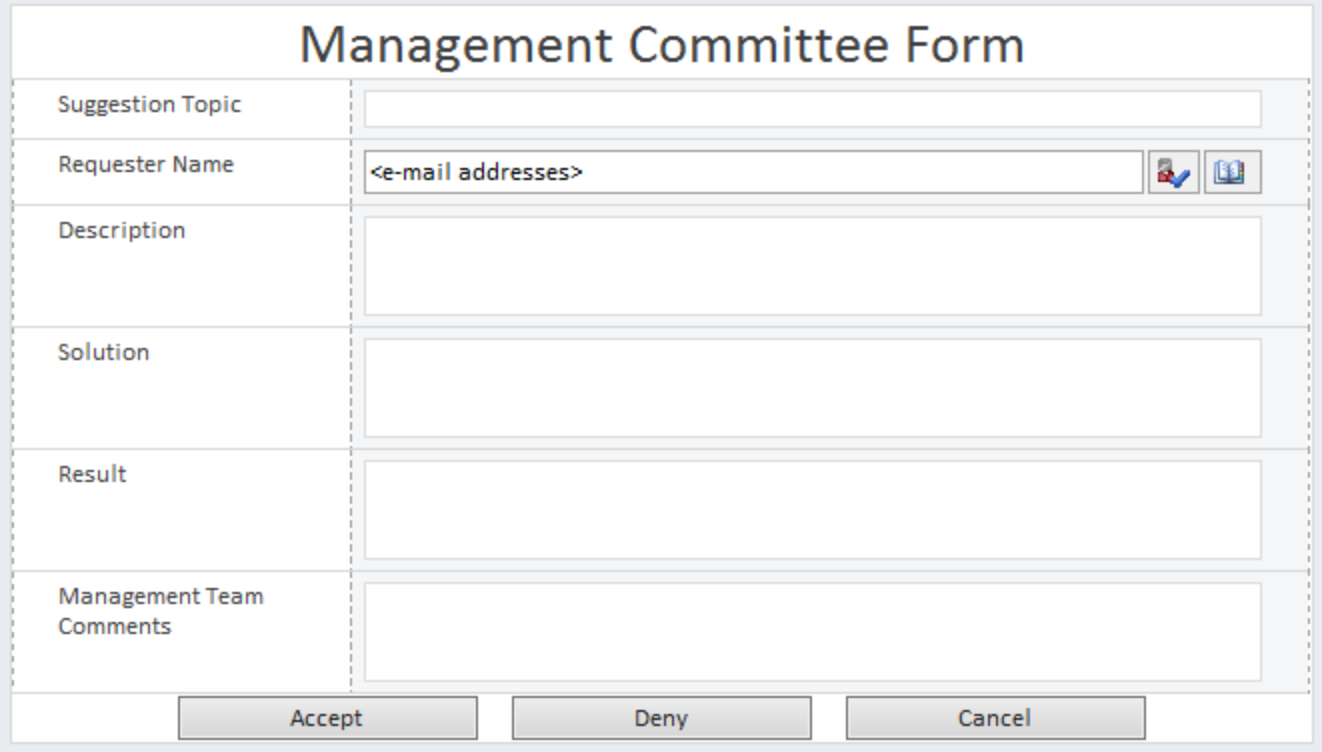
- We are going to add a two new rules in this form. The first new rule will NOT ALLOW the Management team to edit the Suggestion Topic, Requester name, Description, Solution and Result column. The second rule will FORCE the management committee to fill a comment if they deny the request.
- Add three buttons, Accept, Deny and Cancel.

We need only need the Suggestion Topic, Requester Name, Description, Solution, Result and Management Team Comments columns. The rest can be removed from the form.

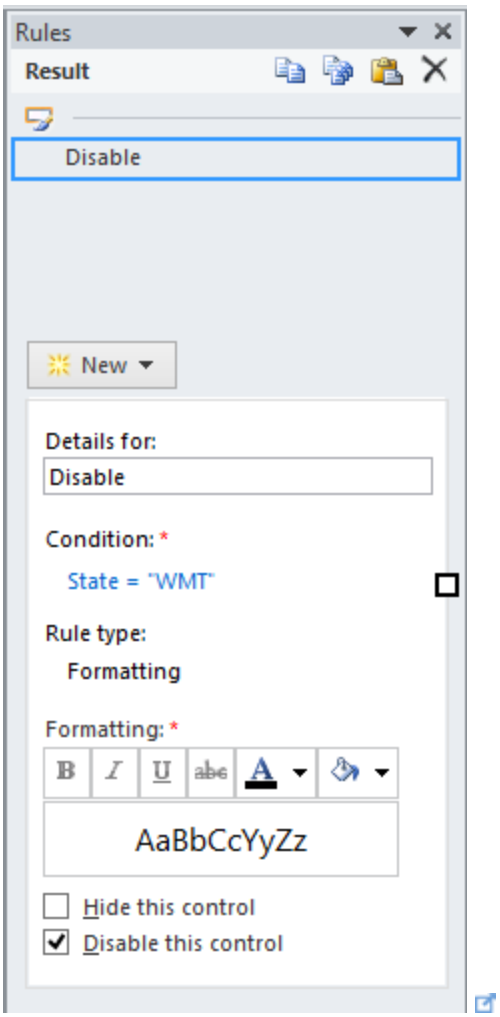
You can also add a title to the form to signify its use as shown below.



Add three Control buttons to this form and name them as Accept, Deny and Cancel. After adding the buttons below is what the form should look like.

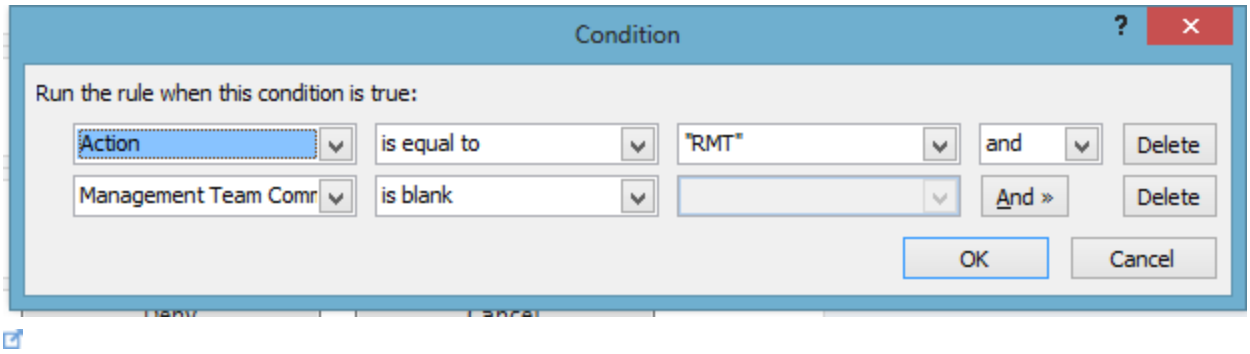


Let's add the first new rule which will NOT ALLOW the Management team to edit the Suggestion Topic, Requester name, Description, Solution and Result column.
This rule needs to be applied separately to Suggestion Topic, Requester Name, Description, Solution and Result field. I'm going to provide the steps for the Suggestion Topic field only, but the process is the same for the remaining.
Select the Suggestion Topic field, Click New, and then choose Formatting. Below are the settings for this rule.
Details for: Disable
Condition: State = WMT
Formatting: check the 'Disable this control' box.
Below is a screenshot of what the settings are.

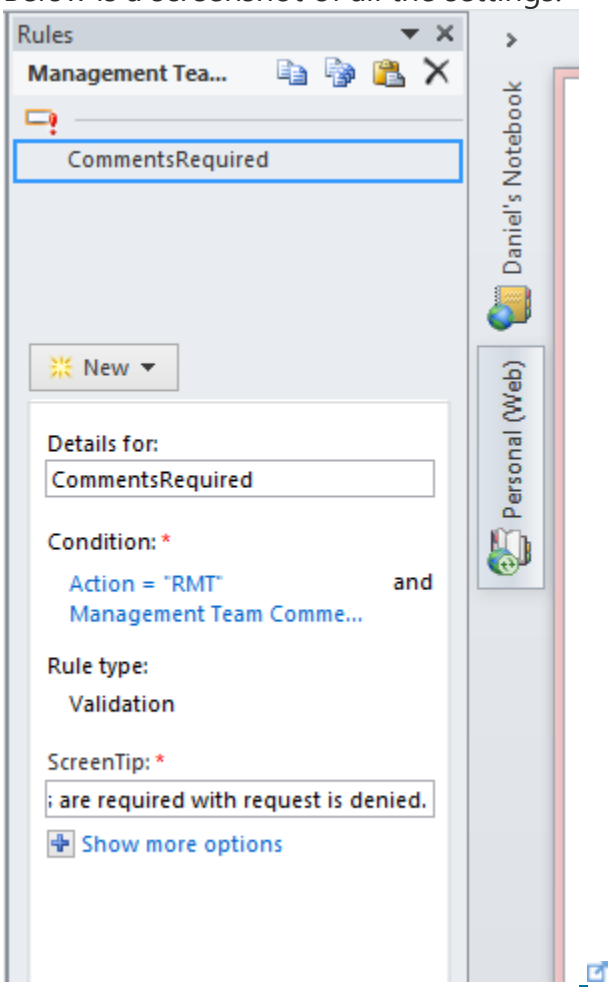


The same rule needs to be applied for remaining we mentioned above. To quicken the process, you can Copy the rule in the Suggestion Topic field and paste it in the remaining fields.

Now, lets add the second rule will FORCE the management committee to fill a comment if they deny the request. We are going to use the Validation Rule.
Select the Management Team Comments field, Click New, and then choose Validation.
Below are the settings for this rule.
Details for: CommentsRequired
Condition: Action = "RMT" and Management Team Comments is Blank

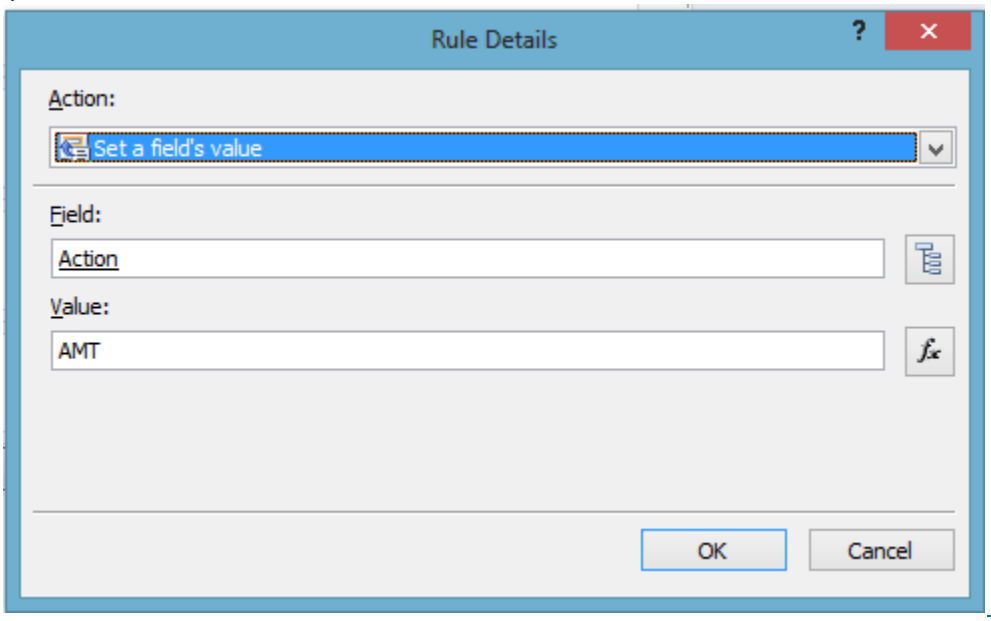


ScreenTip: Comments are required when request is denied.
Below is a screenshot of all the settings.

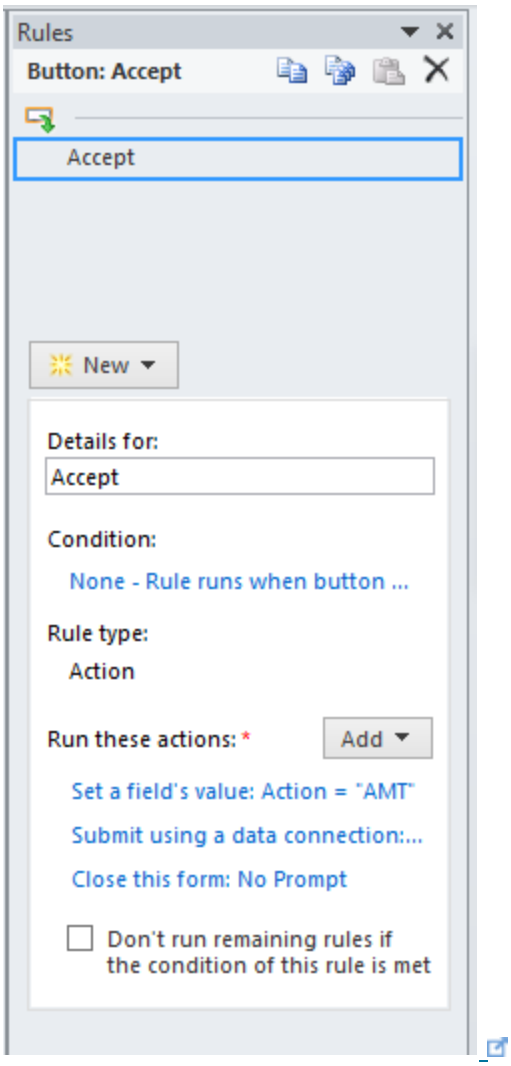


Now lets add the rules for each button.
Select the Accept button and add a new Action Rule. Below are the settings for this rule.
Details for: Accept
Condition: None -Rule runs when the button is clicked. (This is setup by default when you choose the Action rule for a button)

Run these actions:
i) Set a field's value

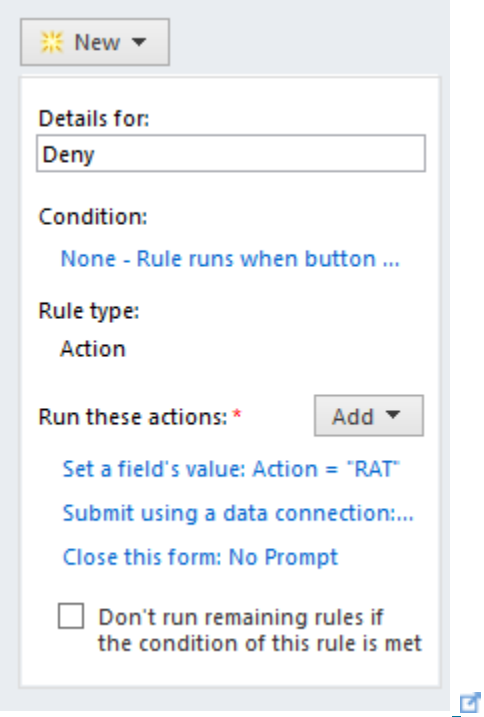


ii) Submit Data
iii) Close this form.
Below is a screenshot of all the settings.

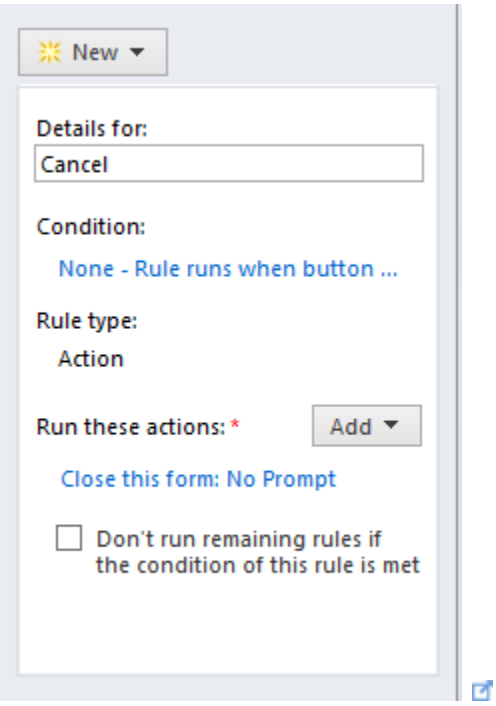


The rules for the Deny button is very much the same except the Set a field's value Action

= RMT. Below is a screenshot of all the settings.



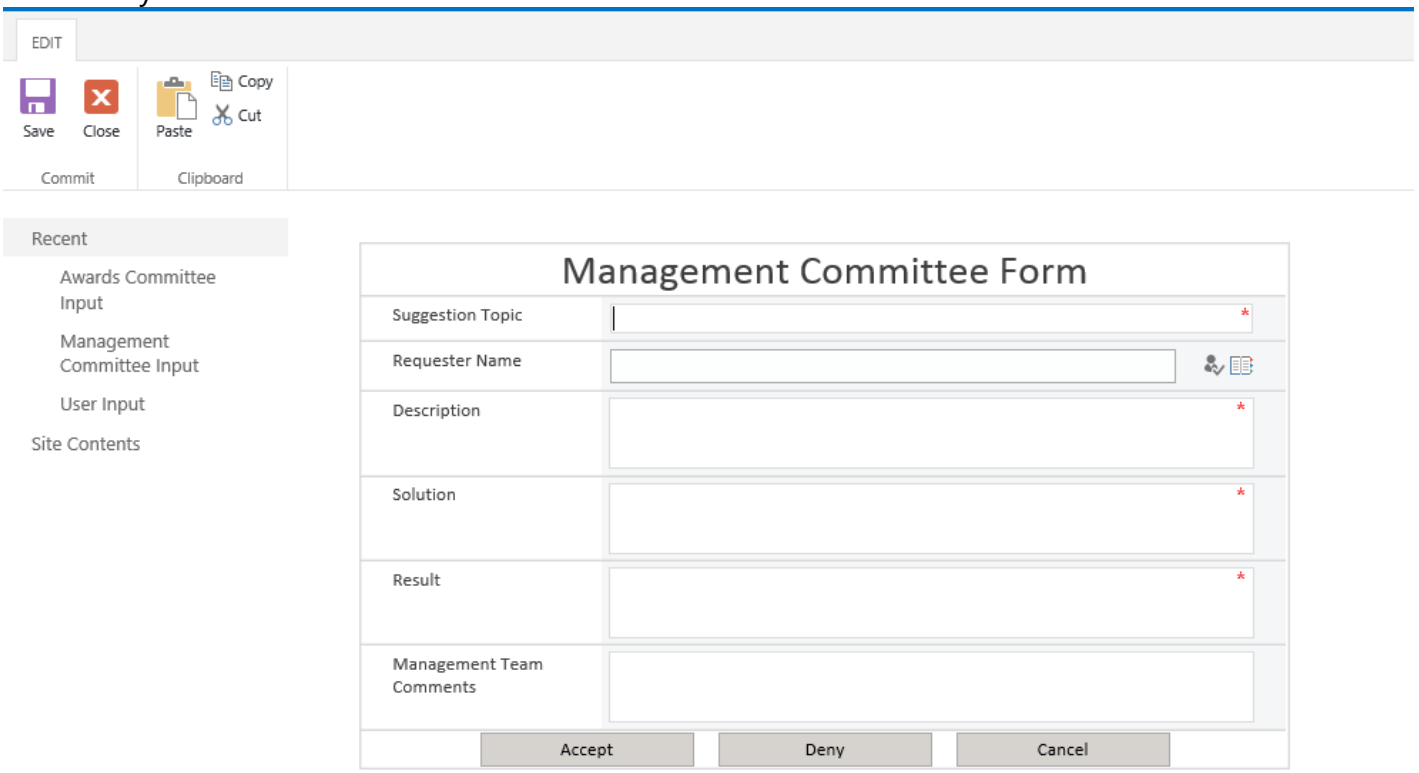
For the Cancel button use the Action rule and use settings below as shown in the screenshot.



Go to File>Form Options and UNCHECK the Show InfoPath commands in Ribbon or toolbar. You can now publish the page by either going to File>Quick Publish or simply

using the Quick publish button on the top left corner.

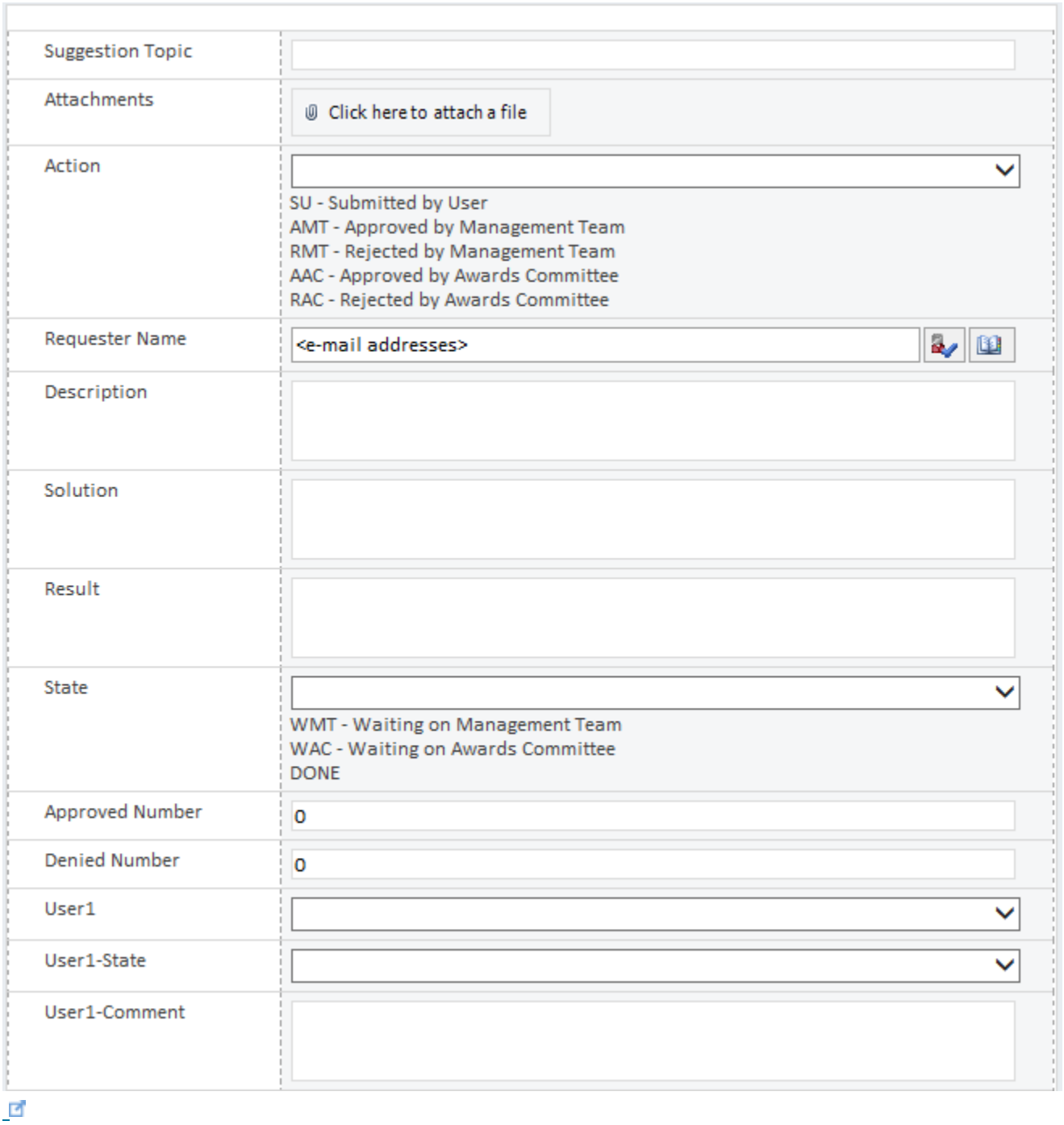
Now when you go back to your Management Committee list on SharePoint and choose New Item, below is the embedded form you should see. Some of you may be wondering as to why we can still edit the Suggestion Topic, Requester Name, Description, Solution and Result even though we set up the Actions in InfoPath designer. I'm glad you thought of that. The simple answer is because the we haven't setup the workflows yet to change the State. Hence, currently the State is blank and if you recall, the disable feature works only when the State is set to WMC.

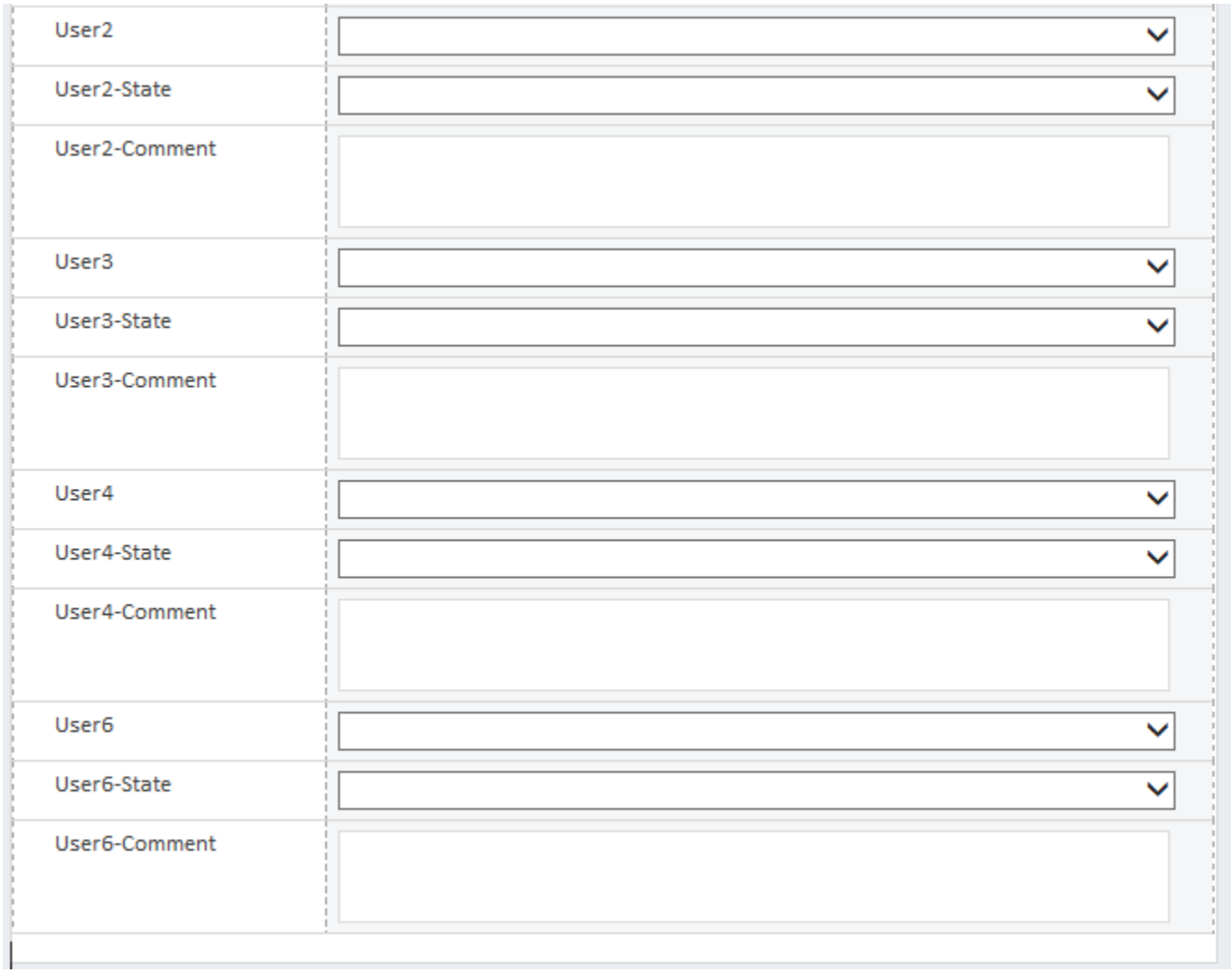


Below is a video which walks you through the process described above to build the embedded form for the Management Committee form. Click Here to view the same video in full size.

Building the Awards Committee Input form

As we did before for the Management Committee form, go to the Awards Committee Input list and choose the Customize Form option.
When the form is first created on InfoPath Designer, it contains all of the fields that were created in the list as shown below.

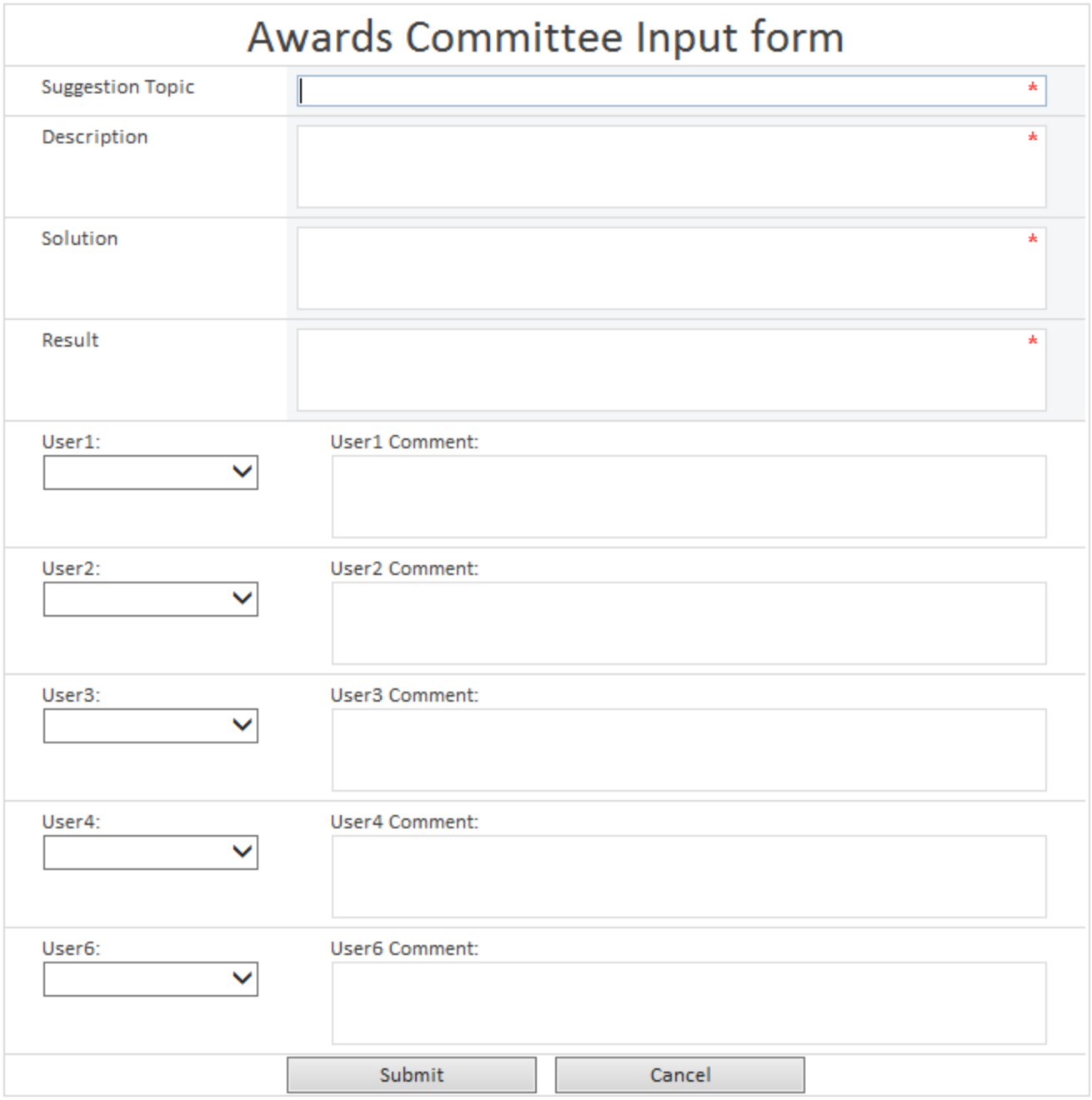




In this form there are a few new features we are going to add.

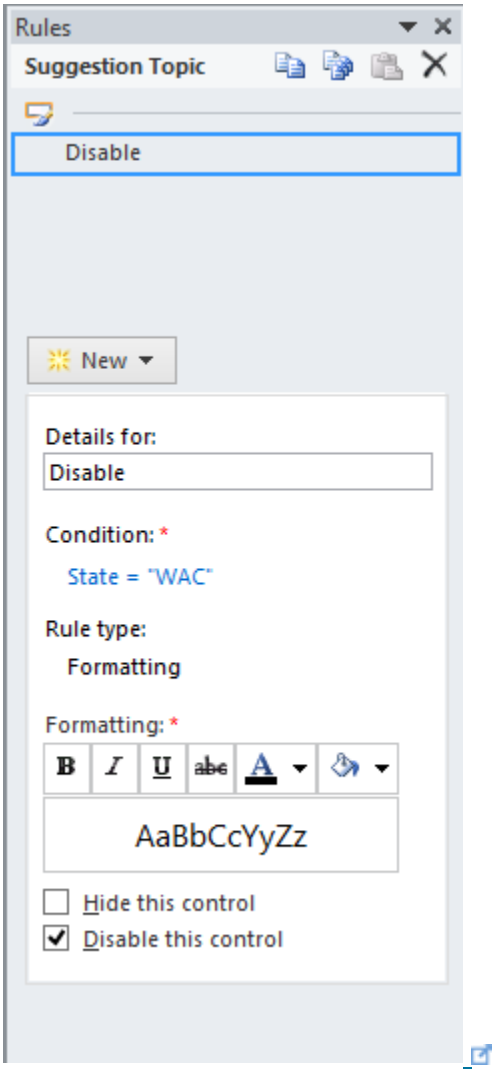
- We are going to add a two new rules in this form. The first new rule will NOT ALLOW the Management team to edit the Suggestion Topic, Description, Solution and Result column. The second rule will FORCE the awards committee to fill a comment if they choose No.
- Add two buttons, Submit and Cancel.

There are selected fields that DO NOT need to be present in the form and hence after removing them and re-arranging the fields below is what the form looks like.



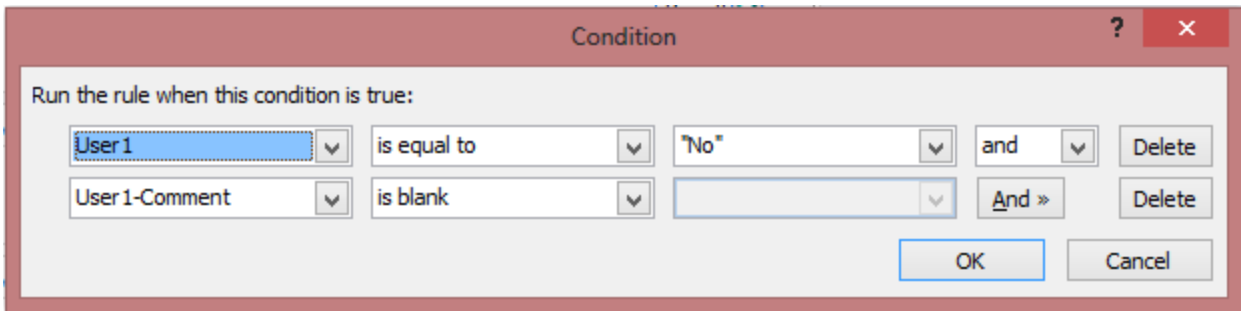
Let's add the first new rule which will NOT ALLOW the Management team to edit the Suggestion Topic, Description, Solution and Result column.
This rule needs to be applied separately to Suggestion Topic, Description, Solution and Result field. I'm going to provide the steps for the Suggestion Topic field only, but the process is the same for the remaining.
Select the Suggestion Topic field, Click New, and then choose Formatting. Below are the settings for this rule.

Details for: Disable
Condition: State = WAC
Formatting: check the 'Disable this control' box.
Below is a screenshot of what that looks like.

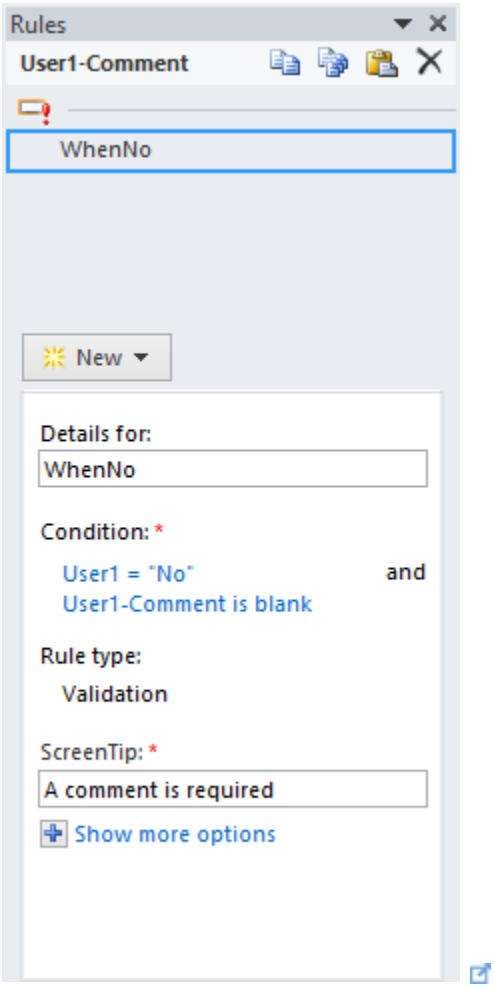


The same rule needs to be applied for remaining we mentioned above. To quicken the process, you can Copy the rule in the Suggestion Topic field and paste it in the remaining fields.

Now, lets add the second rule will FORCE the awards committee to fill a comment if they select No. We are going to use the Validation Rule.
Select the Comments field, Click New, and then choose Validation. Below are the settings for this rule.
Details for: WhenNo
Condition: User1 = "No" and User1-Comment is Blank



ScreenTip: Comments are required when request is denied.
Below is a screenshot of all the settings.

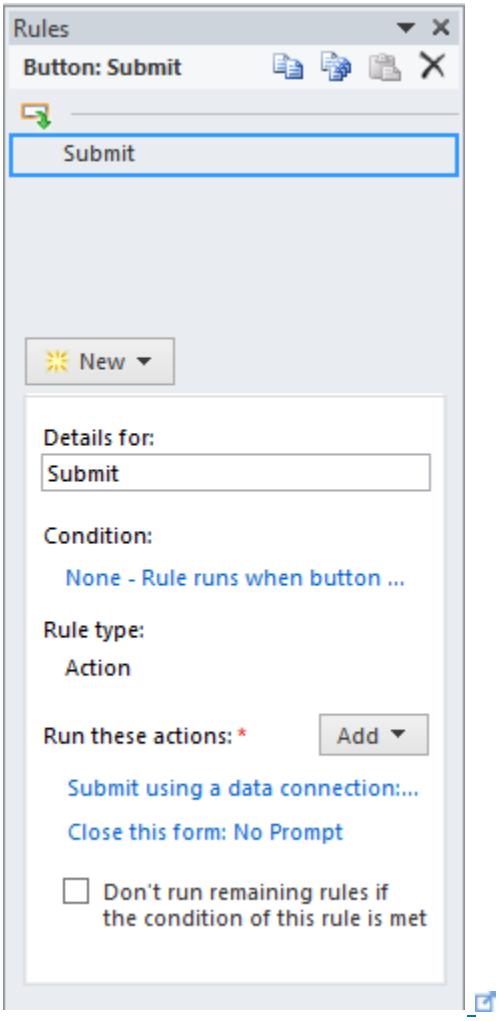


The same rule needs to be applied to remaining four comments fields. To quicken the process, you can use the Copy and Paste feature.

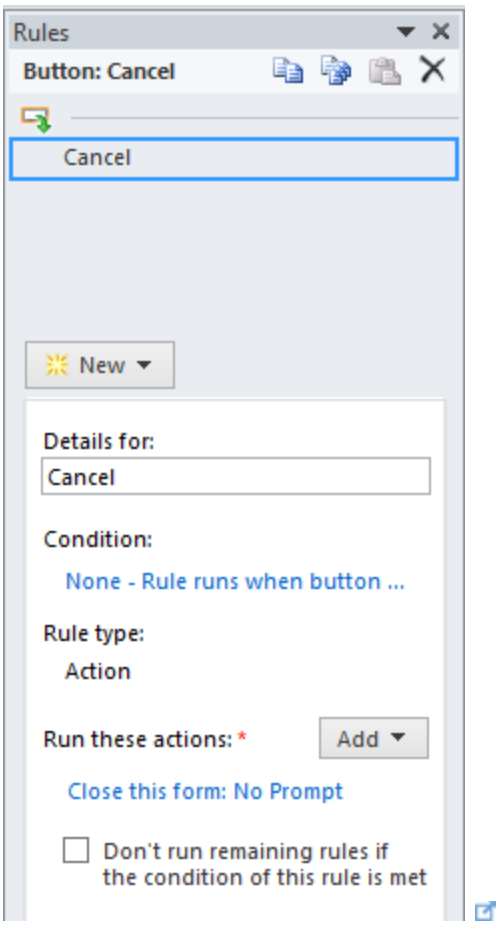
Now lets add the rules for each button.
Select the Accept button and add a new Action Rule. Below are the settings for this rule.

Details for: Accept
Condition: None -Rule runs when the button is clicked. (This is setup by default when you choose the Action rule for a button)
Run these actions:

i) Submit Data
ii) Close this form.
Below is a screenshot of all the settings.

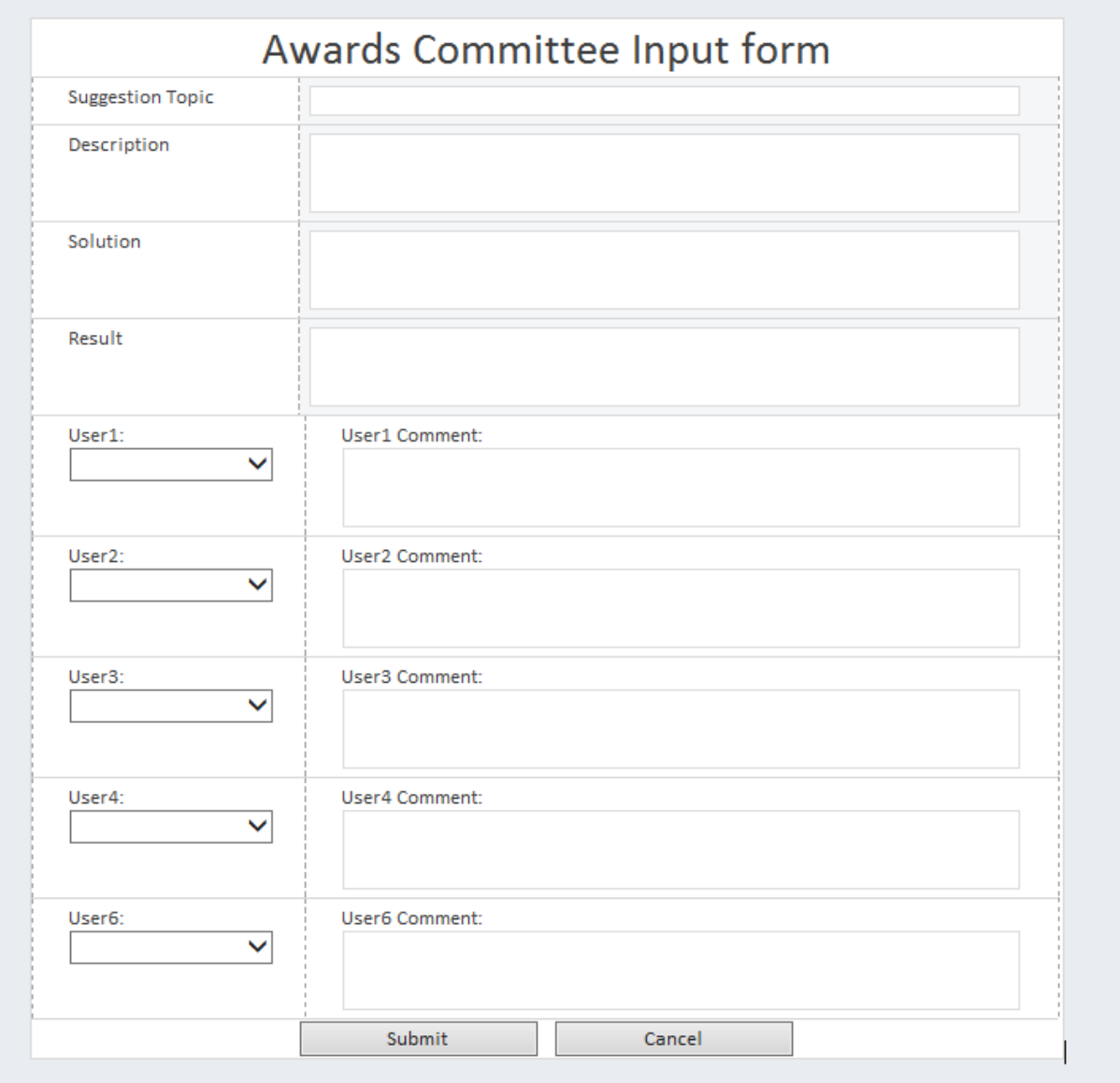


For the Cancel button use the Action rule and use settings below as shown in the screenshot.



Go to File>Form Options and UNCHECK the Show InfoPath commands in Ribbon or toolbar. You can now publish the page by either going to File>Quick Publish or simply

using the Quick publish button on the top left corner.

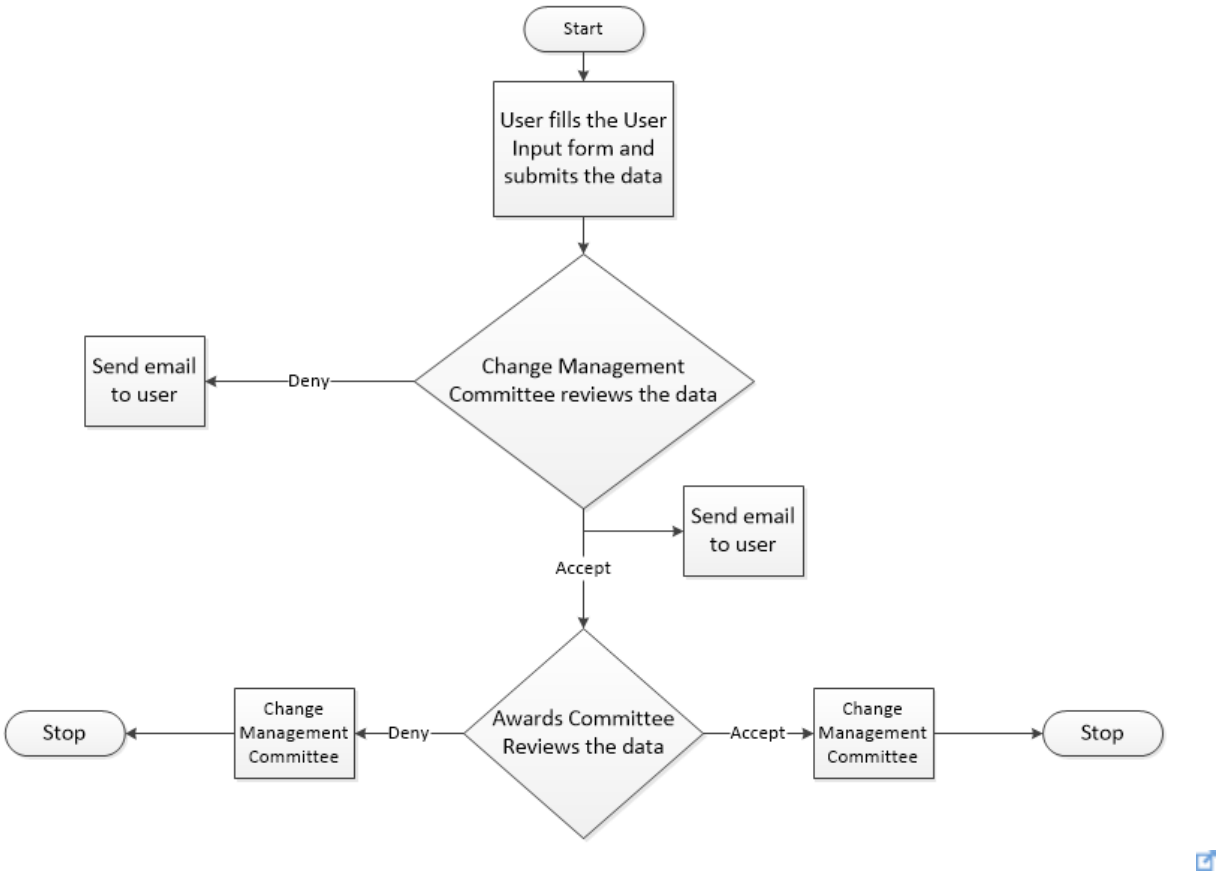


Below is a video which walks you through the process described above to build the embedded form for Awards Committee Input form

Click Here to view the same video in full size

Building the workflows

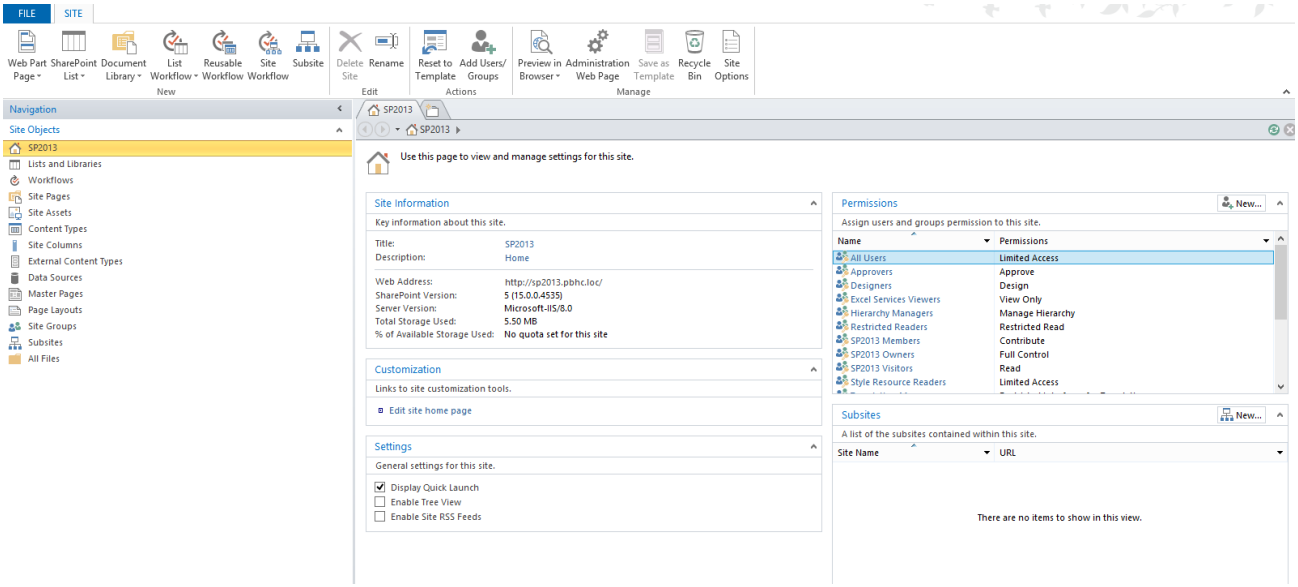
Below is a diagrammatic view of what the workflow is going to be. I am excited to show you the new Copy and Paste feature in SharePoint Designer 2013 which makes things much, much easier and faster.



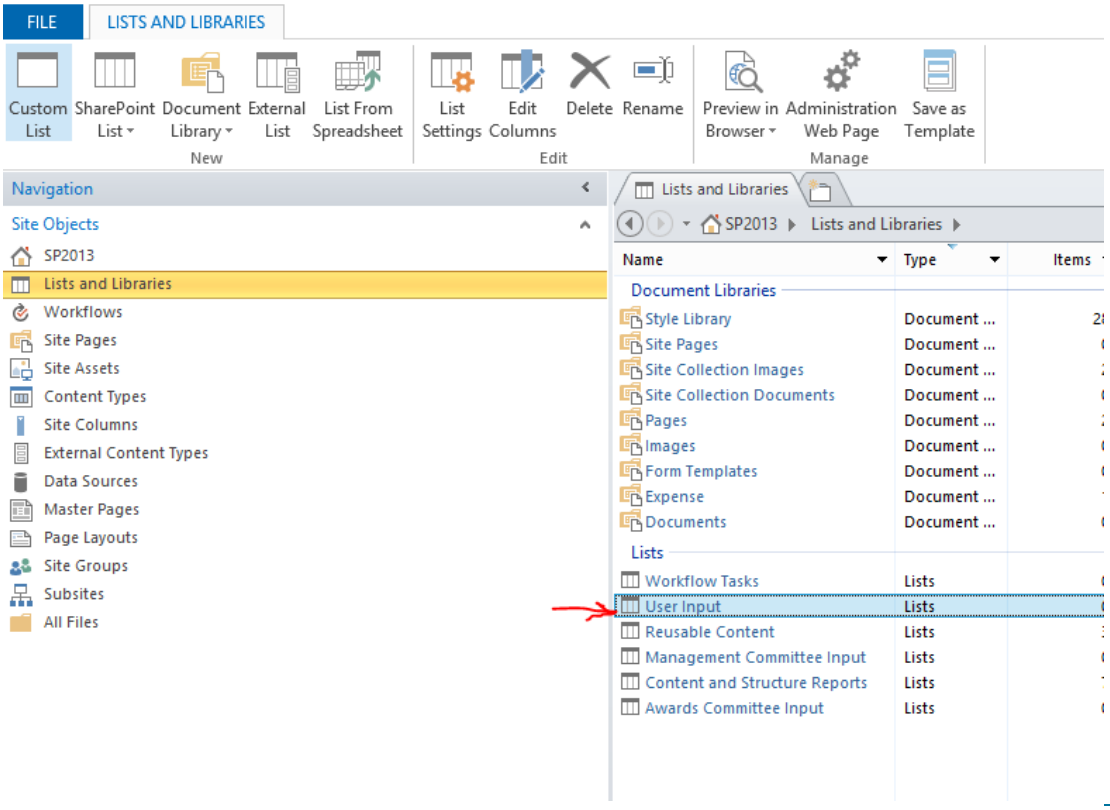
We are going to create three separate workflows whose functions are as follows
a. Once a user fills the form and submits it, the Change Management Committee receives an email alert with information and links. A copy of this list item is created in the Change Management Committee list. Since the permissions of both the lists are NOT THE SAME, we will be using the impersonation step.
b. Change Management Committee is going to review the user input. If the committee approves this request, then an alert email is sent to the end user and all the Awards Committee members.
c. Awards Committee members will access the form and make a decision. As the committee members are making the decision a count is kept on the total number of Yes and No. Once either one of them reaches at the value 3, an alert email is sent to the Management Committee.

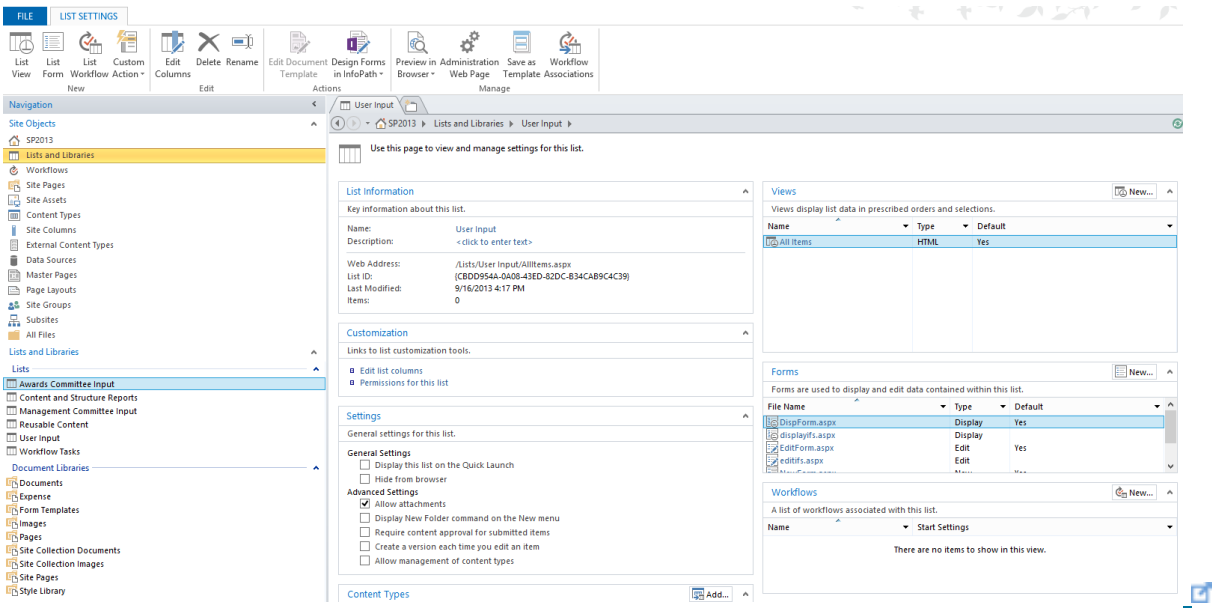
Building the User Input Workflow

Open your SharePoint 2013 site using SharePoint Designer 2013. Below is what the designer should look like.



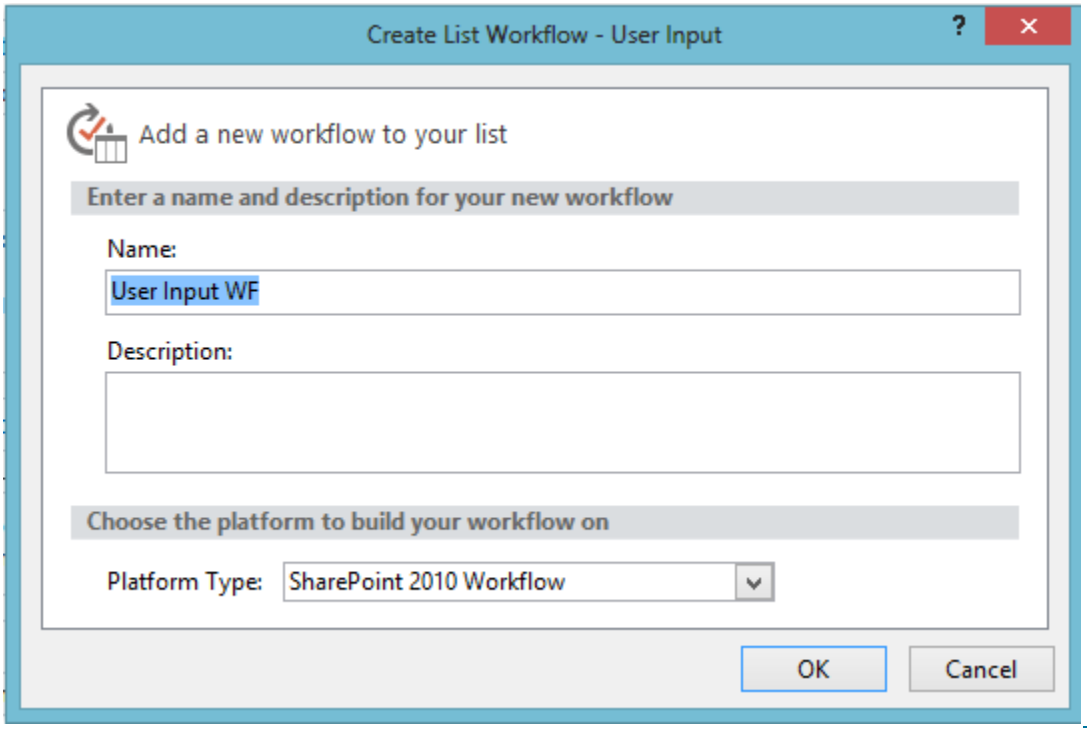
Under Navigation>Site Objects click on Lists and Libraries. Select the User Input List. Below are the two screenshots.

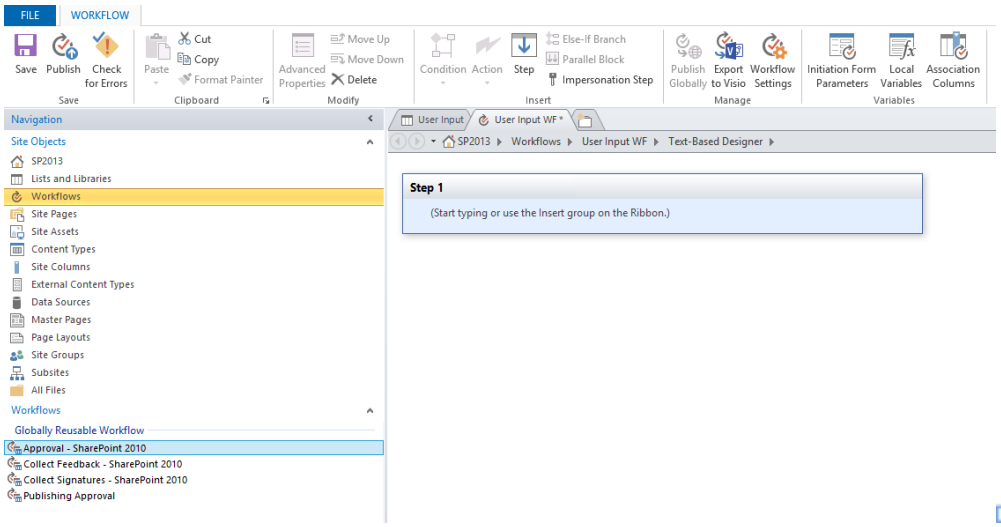




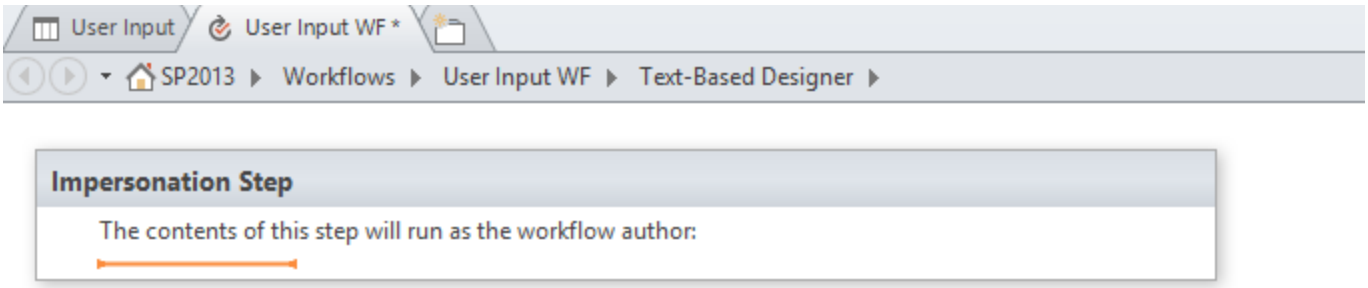
On the top horizontal ribbon, click on List Workflow, select SharePoint 2010 Workflow as the Platform type and give it a name. In my case I am naming it as User Input WF. Below

are the screenshots.





In the horizontal ribbon, in Insert, click on Impersonation Step. You can now delete the Step 1.



Follow the steps provided in the below video to add the Conditions and Actions.

In the Click Here URL you can put in either DispForm.aspx?=ID or EditForm.aspx?ID= Click Here to view the same video in full size.

Building the Management Committee Workflow
Part of this workflow is to copy a list item into the Awards Committee list. In this case we will NOT BE USING the impersonation Step since the Management Committee users already have access to the Awards Committee list.

Follow the steps provided in the below video to add the Conditions and Actions.

In the Click Here URL you can put in either DispForm.aspx?=ID or EditForm.aspx?ID= Click Here to view the same video in full size.

Building the Awards Committee Workflow
Since SharePoint 2013 Designer now has the copy and paste feature available, building this workflow is much much easier. This workflow is keeping a count of to total number of approved and denied counts. If any one of them reach a total of 3 then an alert email is sent to the Management Committee.

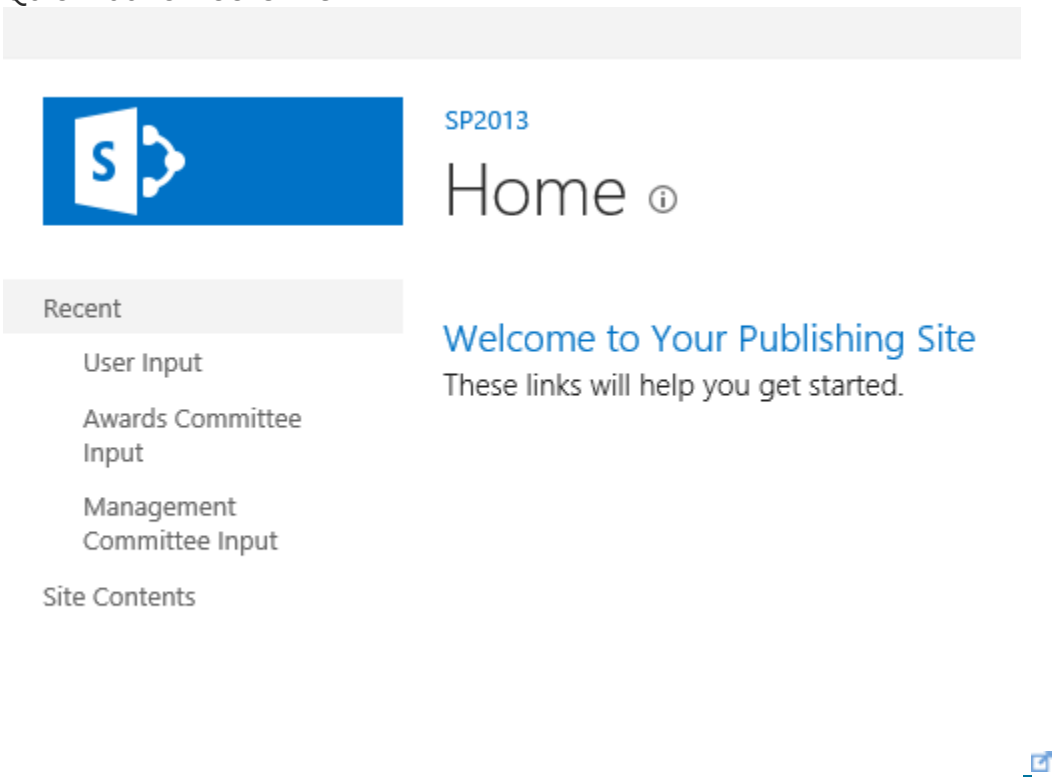
Follow the steps provided in the below video to add the Conditions and Actions.

In the Click Here URL you can put in either DispForm.aspx?=ID or EditForm.aspx?ID=

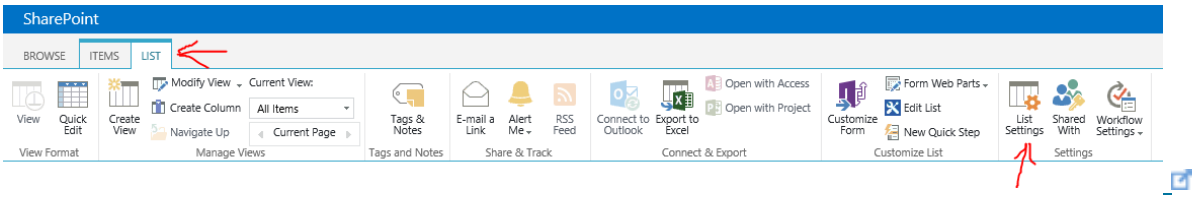
Click Here to view the same video in full size.

Finishing touches

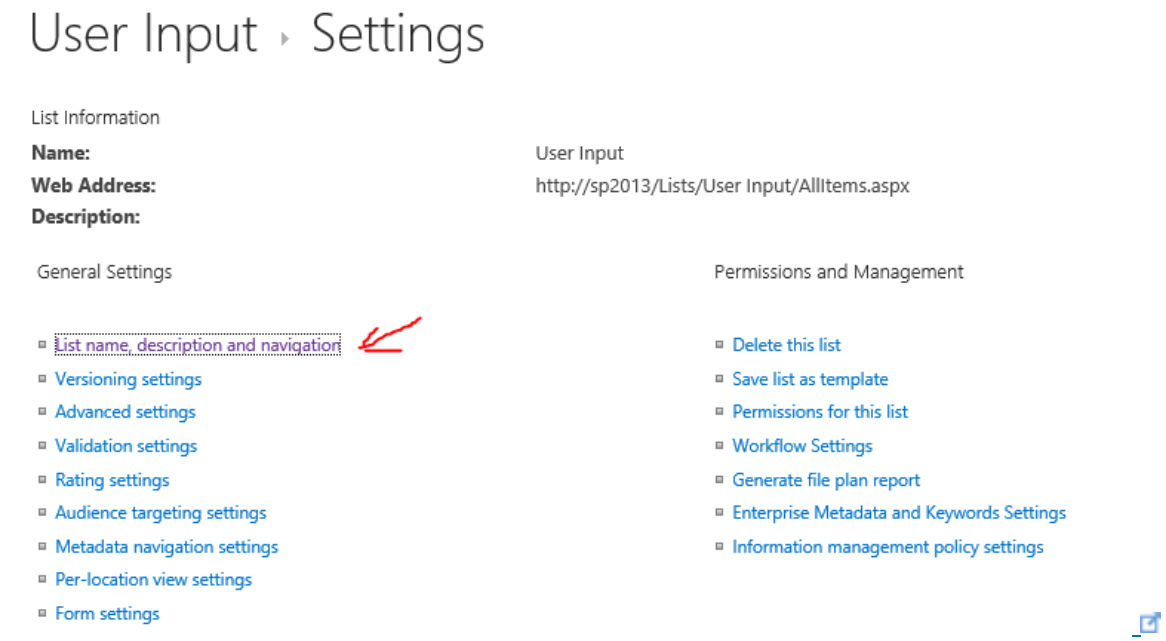
Now that we have all the forms and workflows setup, we need to cleanup the look and feel of the default page and list views.
Home the home page, hide the lists from the Quick Launch as follows. Below is what the Quick Launch looks like



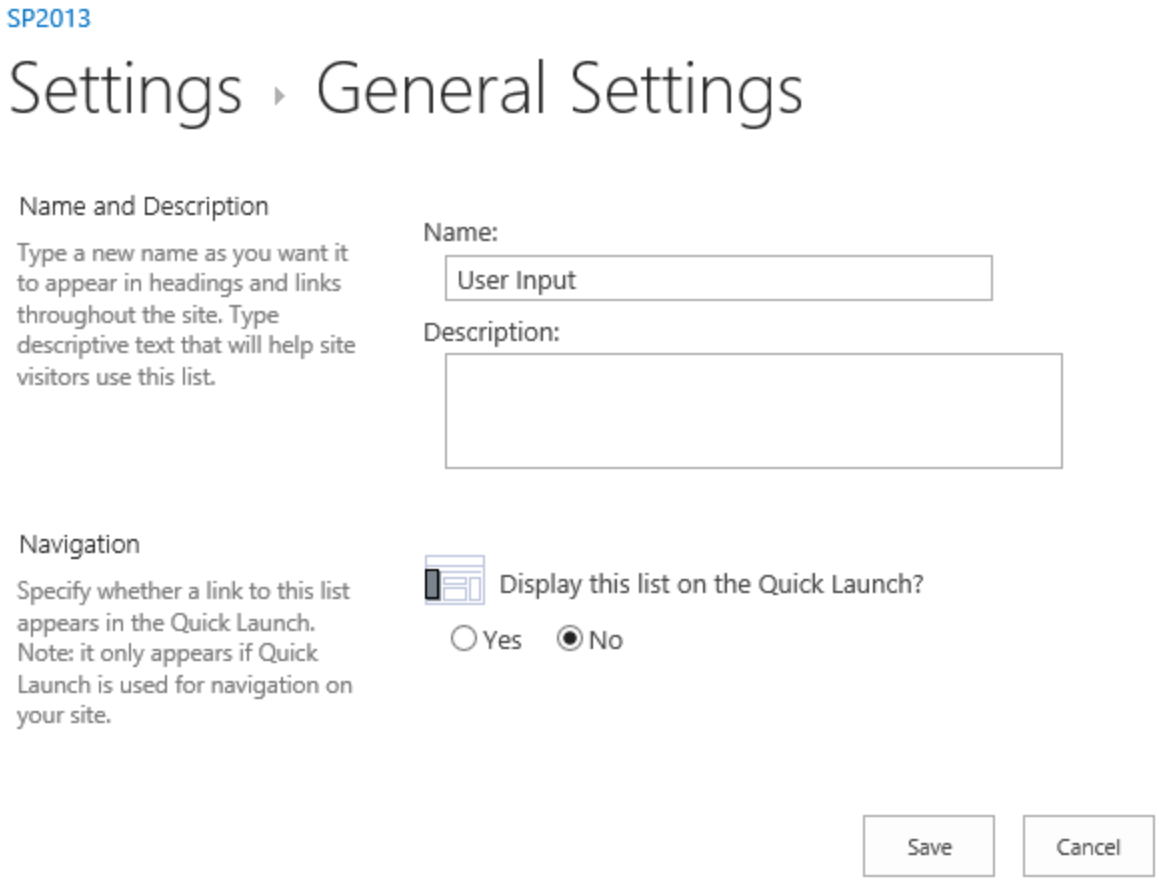
Go to User Input list, click on the List Tab and then choose List Settings as shown below



Under General Settings click on 'List name, description and navigation'

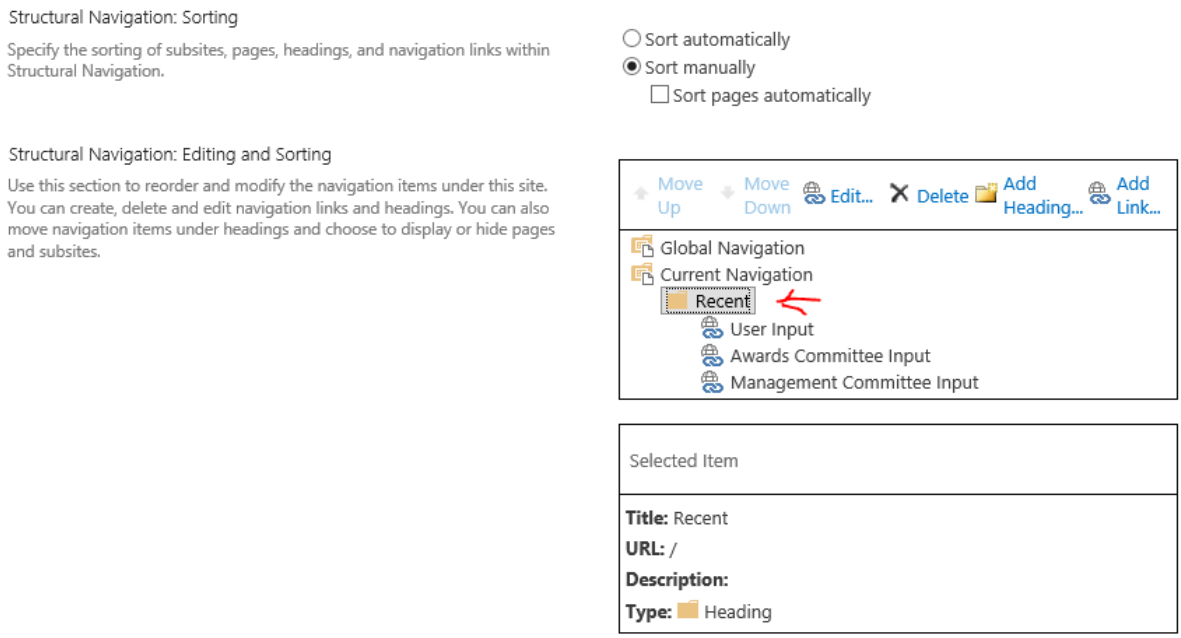


In Navigation, choose No and Save.

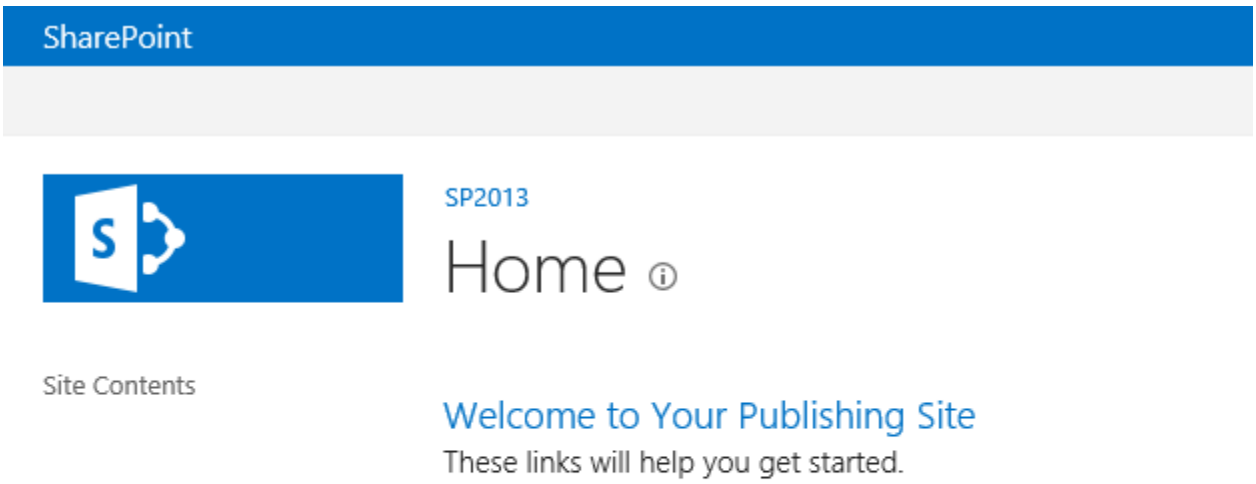


Do the same for the Awards Committee Input and the Management Committee Input list as well.

Also, in SharePoint 2013 the lists will still show up even though you have not displayed it due to the Recent navigation setting.
You can manually remove the Recent navigation from Settings-> Site Settings -> Navigation (under Look and Feel).
Select the Recent navigation option and click Delete as shown below

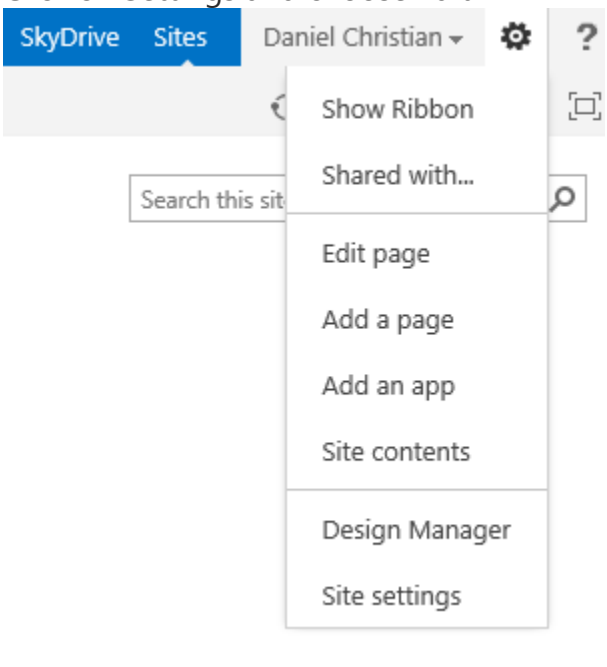


Now when you visit the home page you will no longer see the lists in the Navigation as shown below

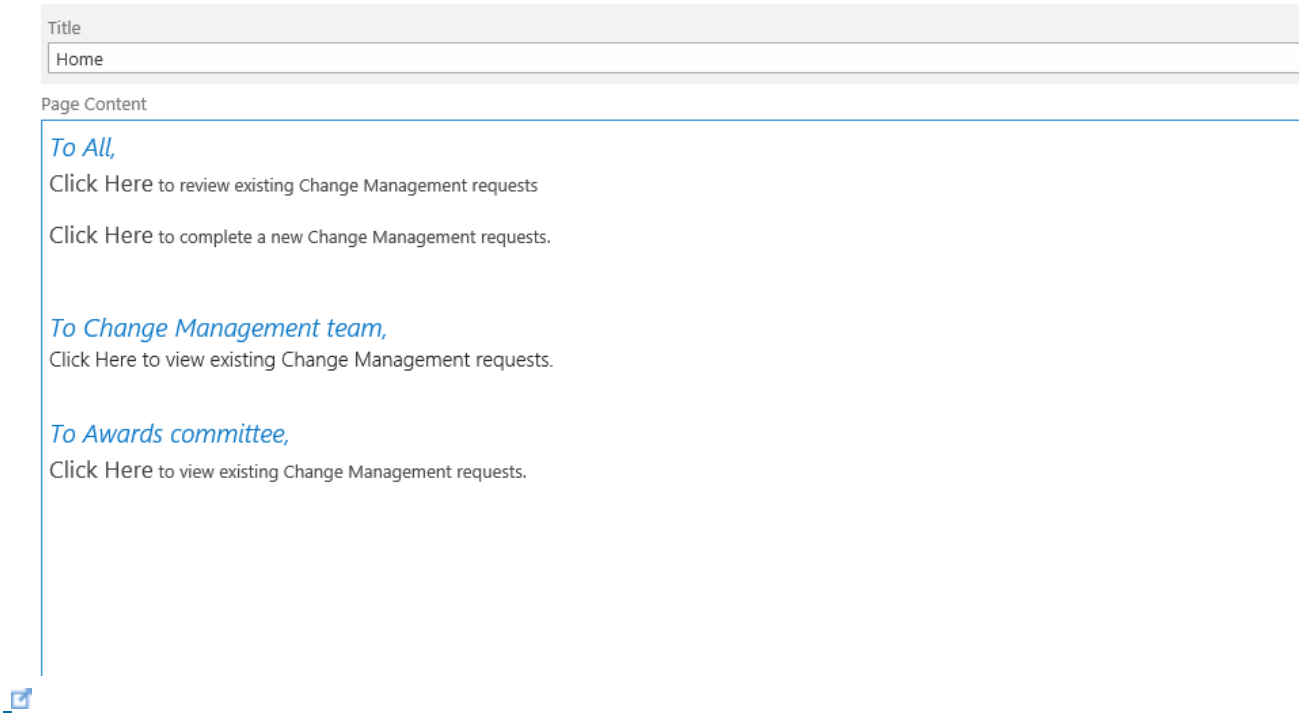


Lets add a few options to select from keeping in mind permissions.

Click on Settings and choose Edit

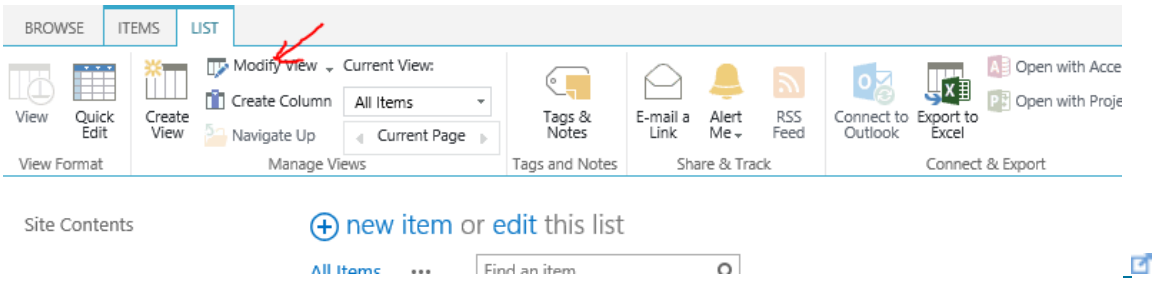


In the Page Content section, you can delete any unwanted text there and add something similar to the following.

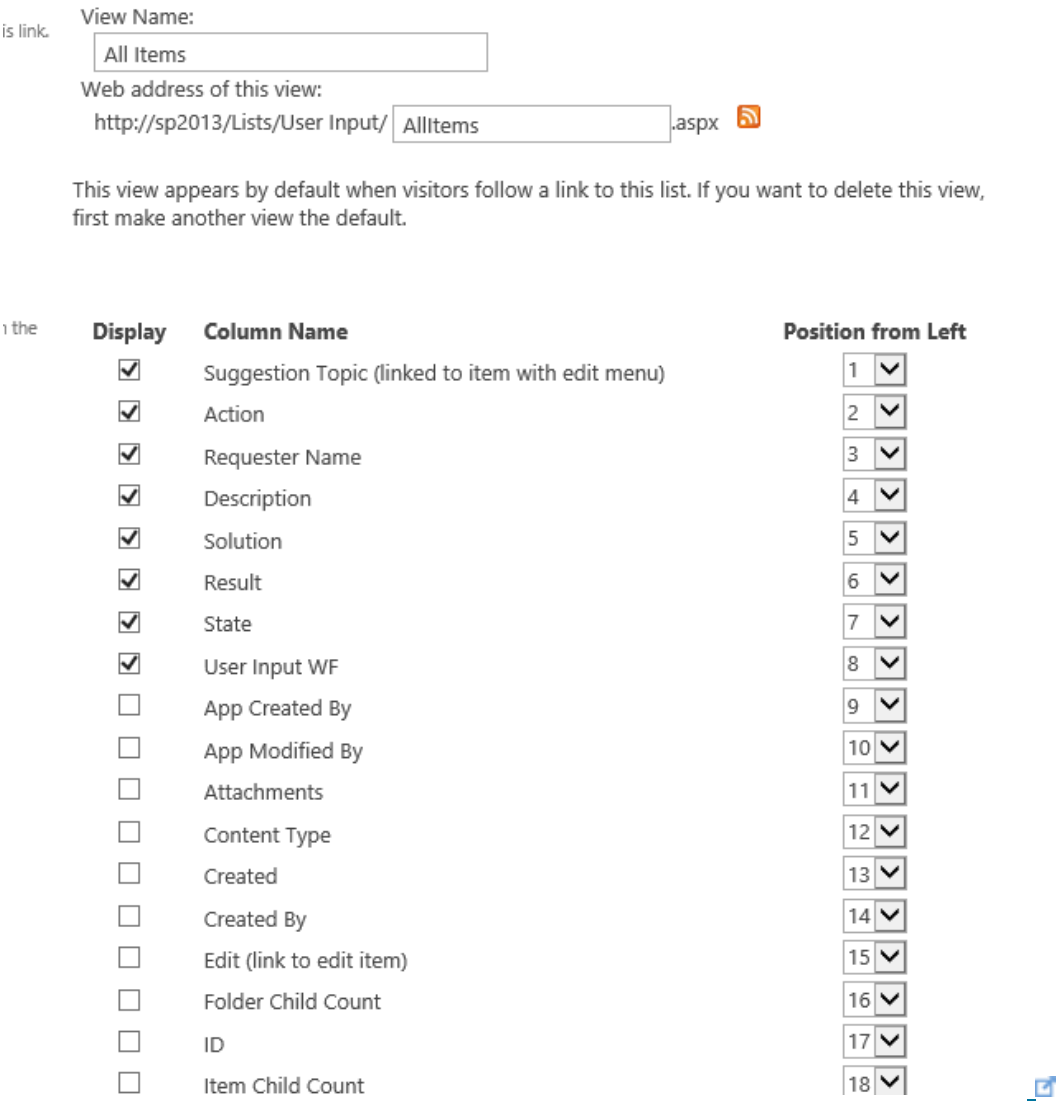


Save, check-in and publish the changes. We will be returning back to this home page to add URLs.

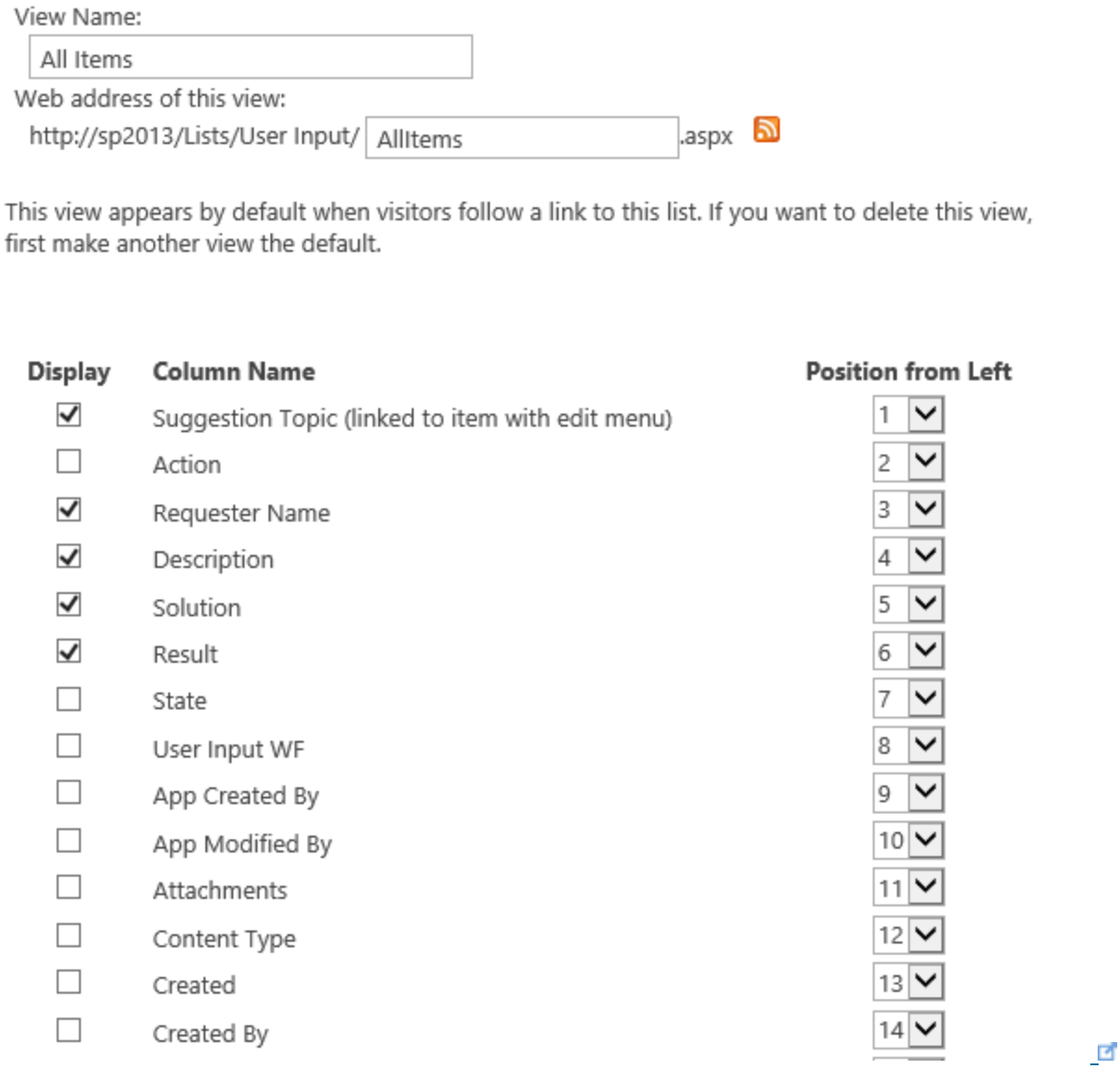
Go to User Input list, click on the List Tab and then choose Modify View as shown below



Below are the default settings of this list view.

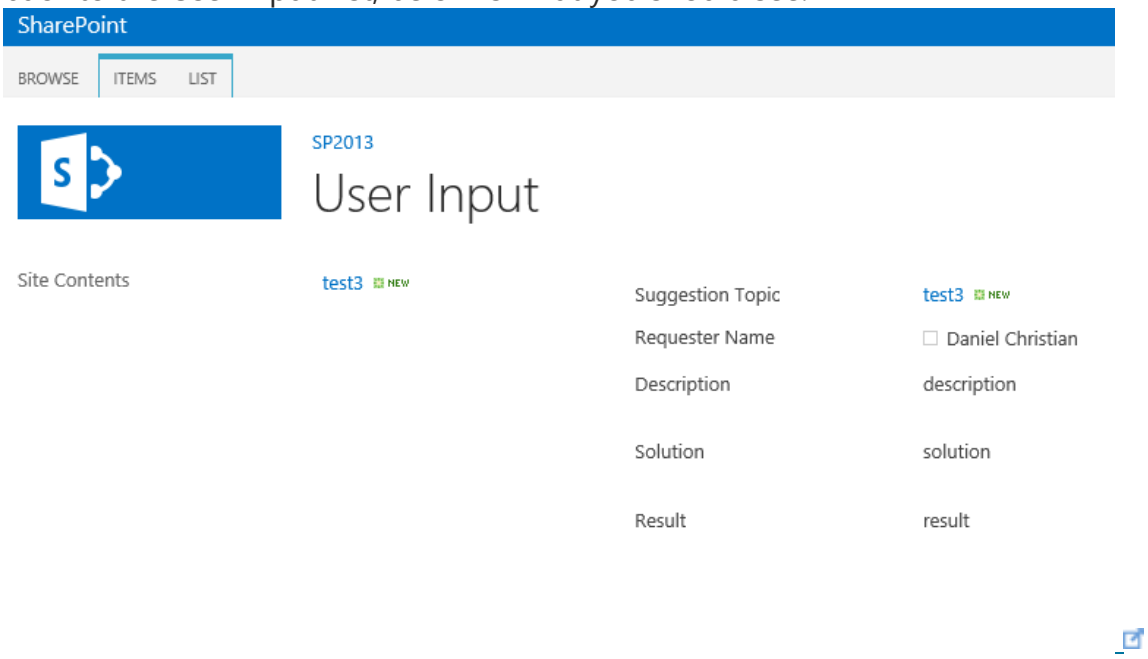


Make the below changes:

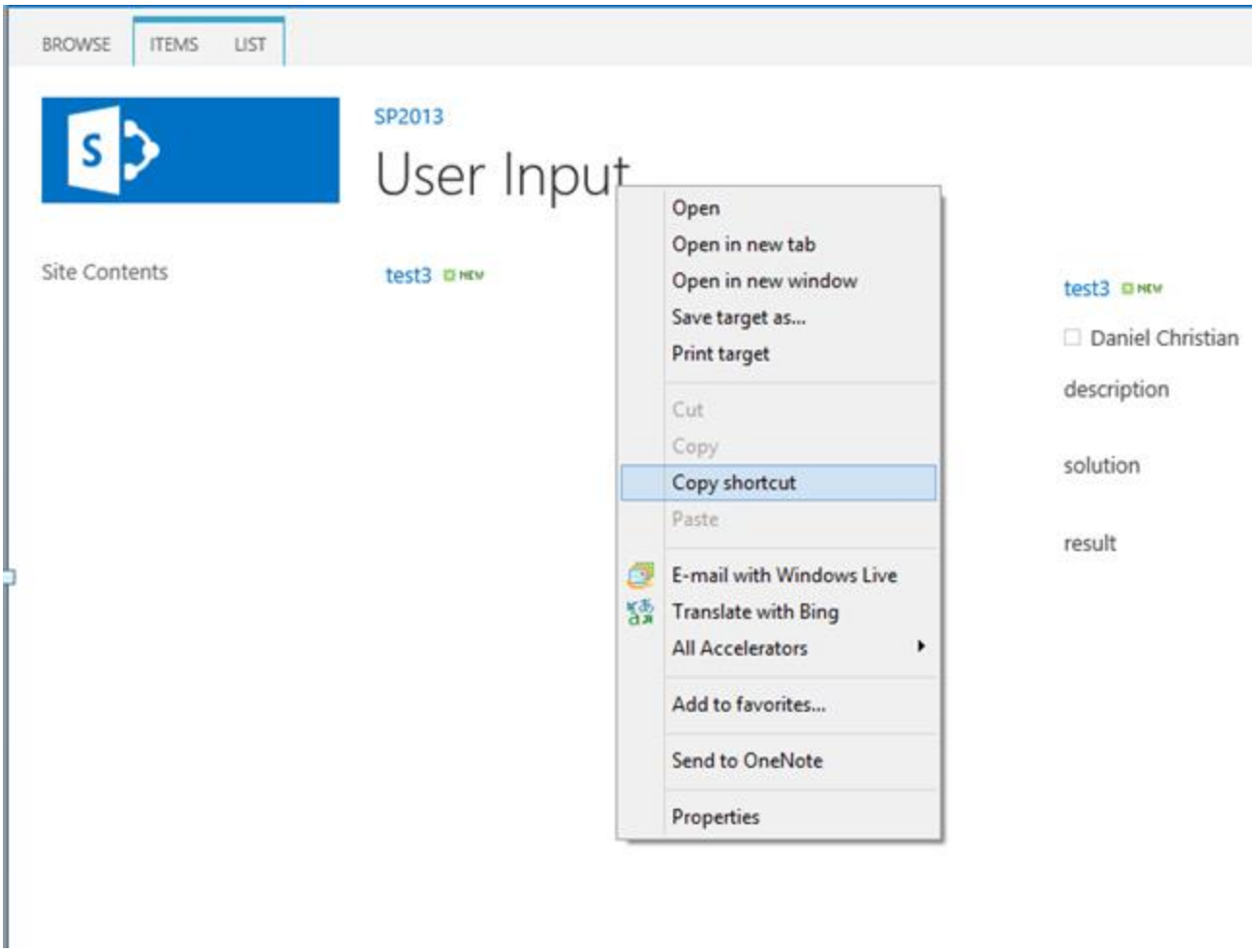


Scroll down below and expand Style and choose Preview. Click Ok. Now when you go

back to the User Input List, below is what you should see.

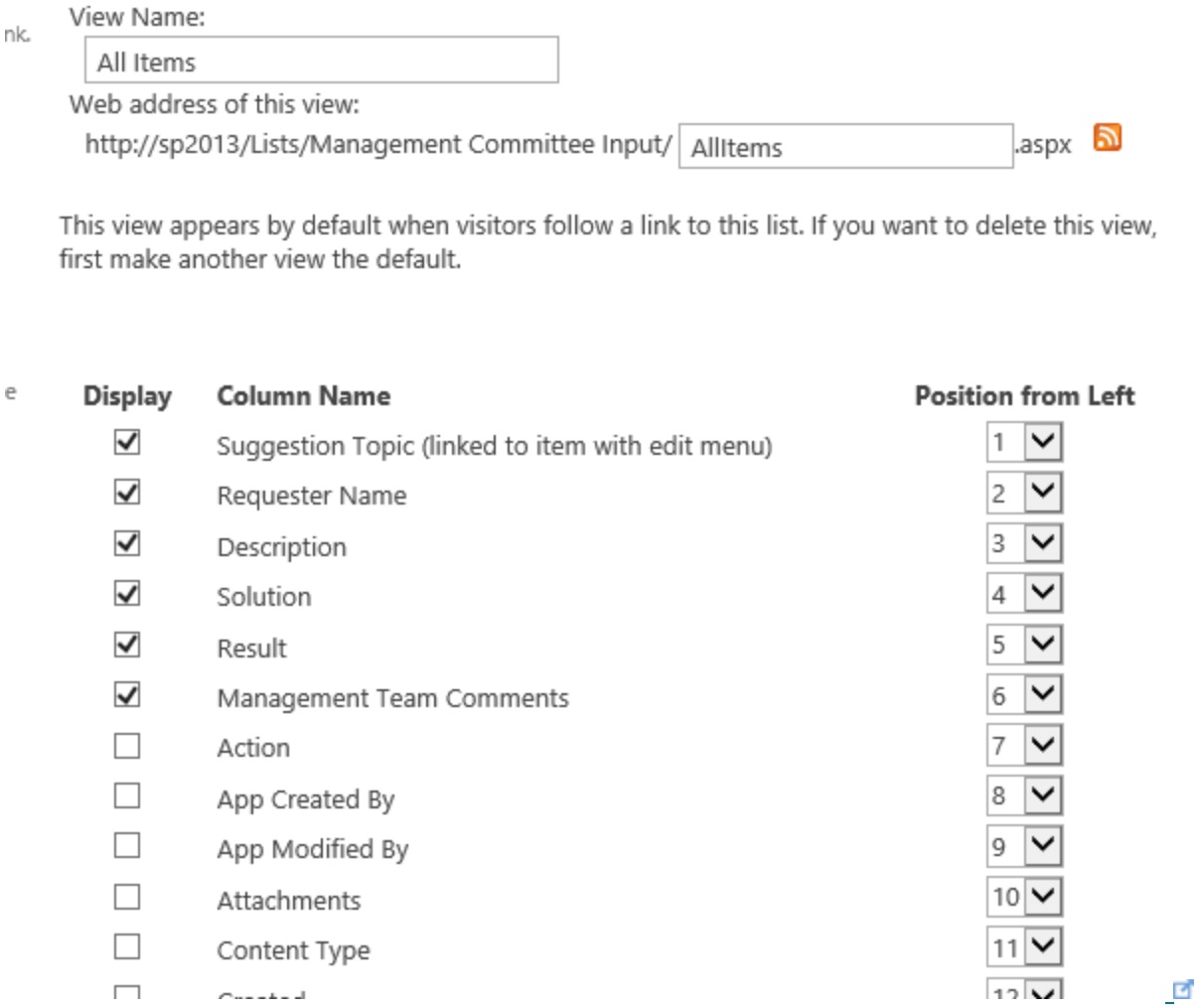


Right-Click on the User Input title, and copy shortcut as shown below

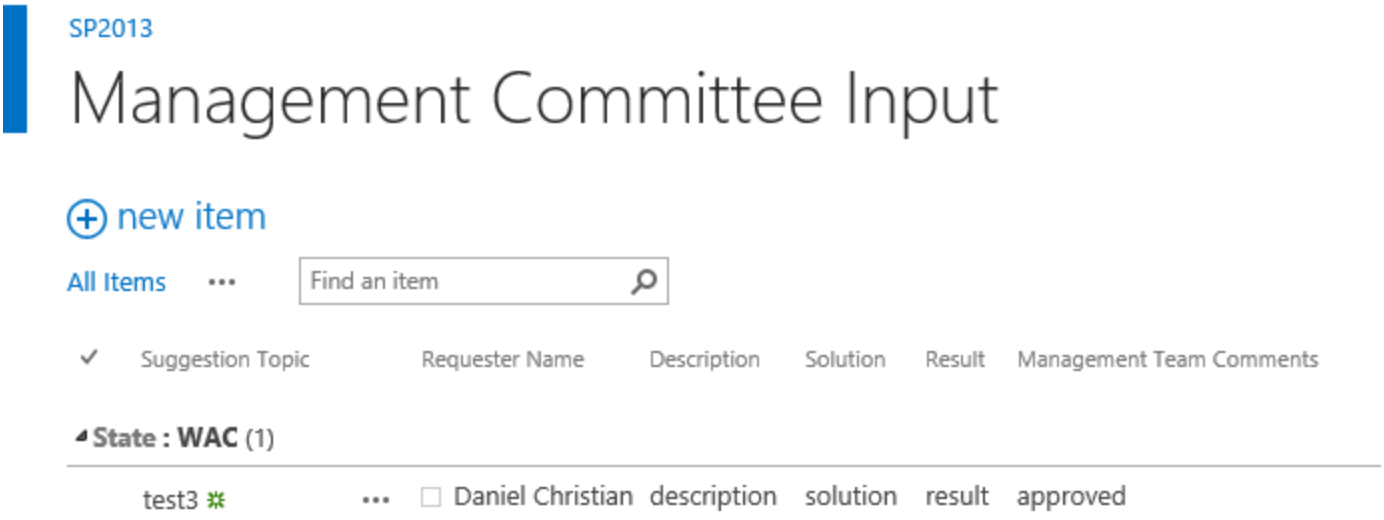


Go back to the home page, and paste that URL to the 'Click Here' in 'Click Here to review existing Change Management request' .
Add the following URL to the 'Click Here' in 'Click Here to complete a new Change Management request.'
http://sp2013/Lists/User%20Input/NewForm.aspx

Go to Management Committee Input list, click on the List Tab and then choose Modify View. Make the below changes.



Scroll down below and expand Group By, choose State in the 'First Group by the column'. Click Ok. Now when you go back to the Management Committee List, you should see a view similar to the below screenshot.

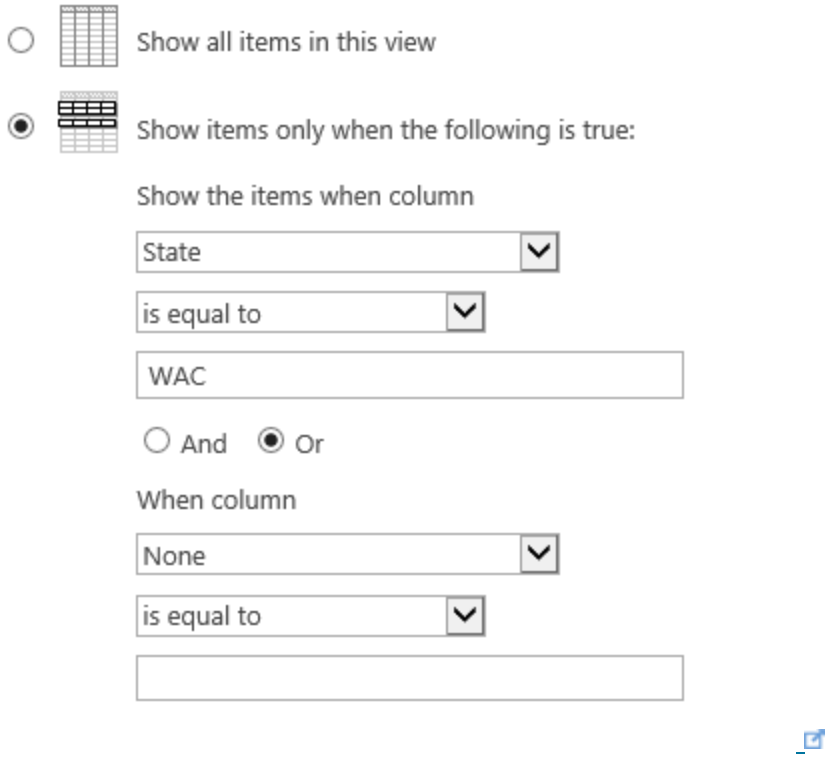


Right-Click on the Management Committee title, and copy shortcut.
Go back to the home page, and paste that URL to the 'Click Here' in 'Click Here to review existing Change Management request' in To Change Management Form.

Finally, Go to Awards Committee Input list, click on the List Tab and then choose Modify View. Make the below changes.

Display	Column Name	Position from Left
☑	Suggestion Topic (linked to item with edit menu)	1
☐	Action	2
☐	Requester Name	3
☑	Description	4
☑	Solution	5
☑	Result	6
☐	State	7
☐	Approved Number	8
☐	Denied Number	9
☑	User1	10
☐	User1-State	11
☑	User1-Comment	12
☑	User2	13
☐	User2-State	14
☑	User2-Comment	15
☑	User3	16
☐	User3-State	17
☑	User3-Comment	18
☑	User4	19
☐	User4-State	20
☑	User4-Comment	21
☑	User6	22
☐	User6-State	23
☑	User6-Comment	24
☐	Awards Committee Input WF	25

Scroll down below and expand Filter, select the 'Show items only when the following is true:' radio button and make the following changes. This will allow the Awards committee to only view those Change Management requests that are waiting for them to be reviewed and none other.



Right-Click on the title, and copy shortcut.
Go back to the home page, and paste that URL to the 'Click Here' in 'Click Here to review existing Change Management request' in To Award Committee Form.

Below is a brief video which walks you through the process of adding the text on the home page, filter the lists and adding the links. Click Here to view the same video in full size

Demo

Finally, the below video shows a demo of how an everyday user (user10) would put in a request, the Change Management team (User 7 or User 8) would approve it and the Awards committee members would log in to approve it.
For sake of demonstration I have created a personal view for lists with which I can view the control columns such as State and Action. This is not needed once the workflow goes live on your production farm.

(Here's a tidbit - In SharePoint 2013 the 'Sign in as a Different User' is not available, however you can use http://yoursite/_layouts/closeConnection.aspx?loginasanotheruser=true to do that.)
Click Here to view the same video in full size.

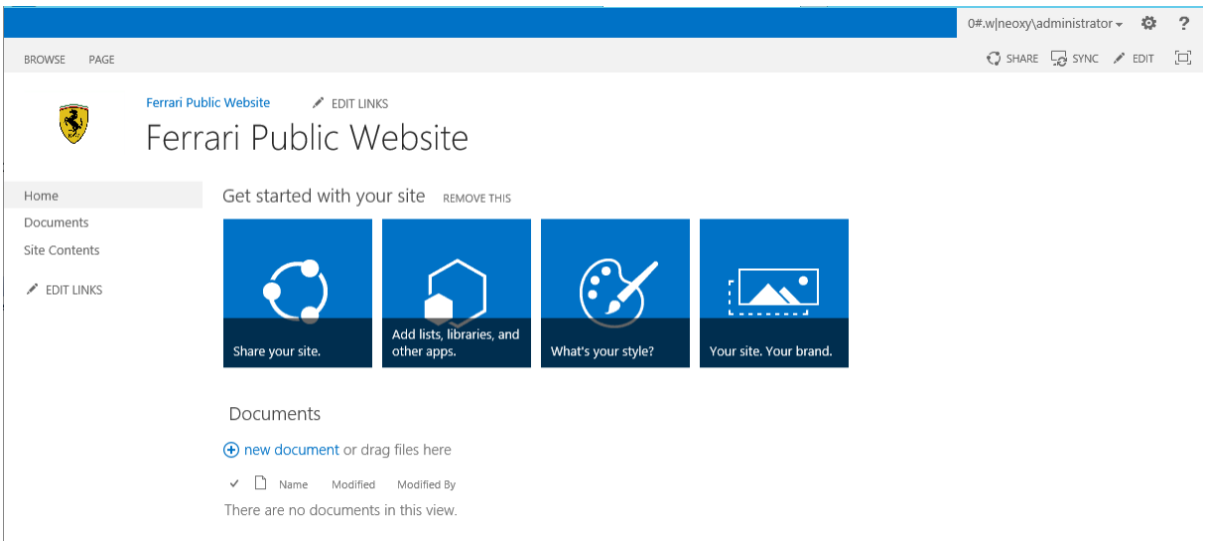
I would like to end by saying that there are several ways to perform this same function and tighten the security further, however, this demonstration should give you a quick and easy way to create a Change Management type workflow without any code!

See Also

- SharePoint 2013 Portal
- SharePoint 2013 - Service Applications
- SharePoint 2013 - Resources for Developers
- SharePoint 2013 - Resources for IT Pros

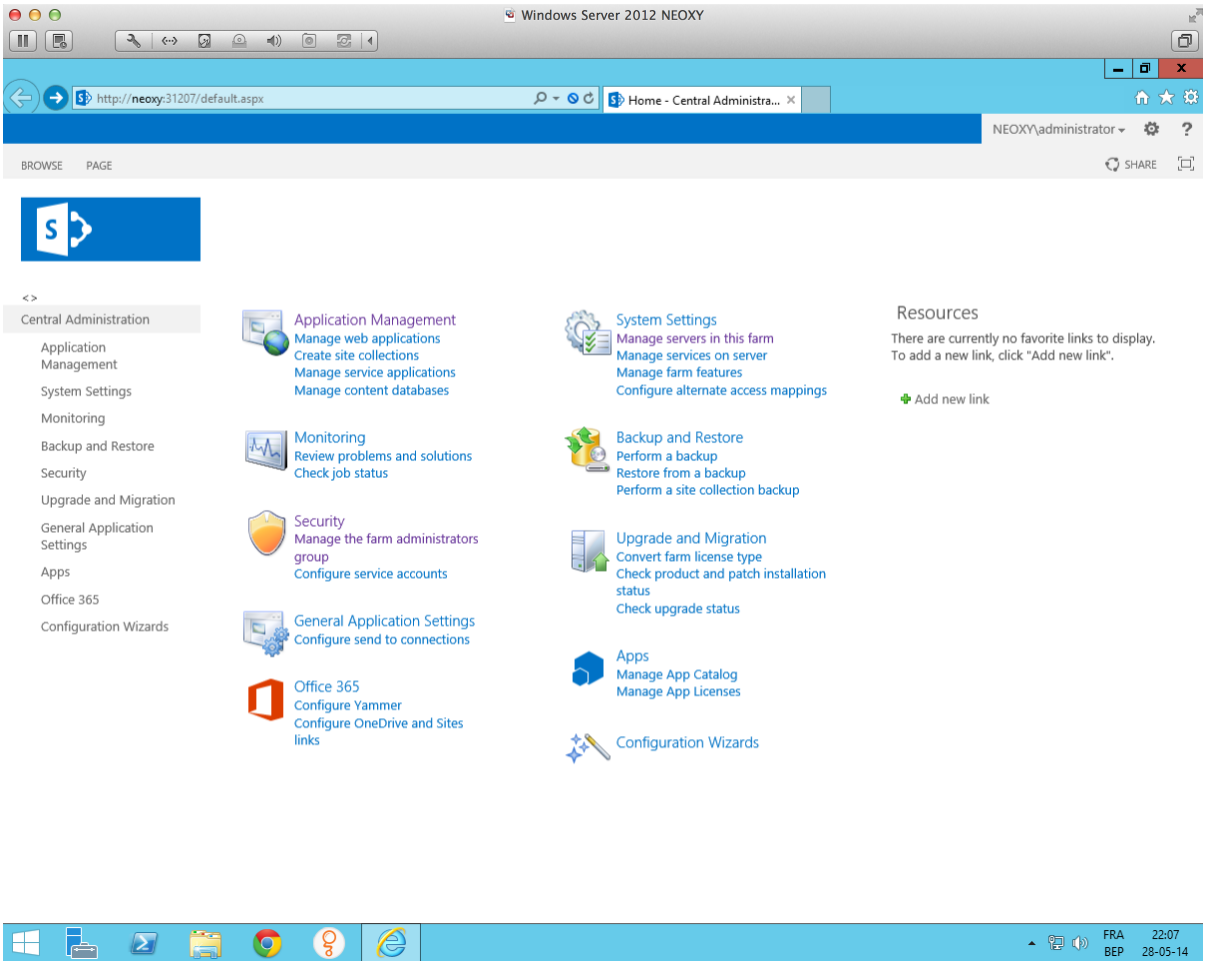
SharePoint Server 2013: Customize your Central Administration

Once a time **Jeff Teper** – Senior Vice President for SharePoint said that we should "use SharePoint as an out-of-box application whenever possible. We designed SharePoint [2013's] UI to be clean, simple and fast and work great out-of-box. We encourage you not to modify it, which could add complexity, performance and upgradeability, and to focus your energy on working with users and groups to understand how to use SharePoint to improve productivity and collaboration and identifying and promoting best practices in your organization."
This is **ABSOLUTELY** true for intranet or extranet sites. We've all seen awful intranet/extranet designs. Flashing lights, animated gifs and so many other things. But thankfully they're now less common or even disappeared. A corporate Intranet or Extranet should be as simple as possible. This should be even considered a best practice
This is **NOT** true for Public Facing Internet Sites. Hopefully I guess, because otherwise the world well knows SharePoint Public Facing Website should be something like that:

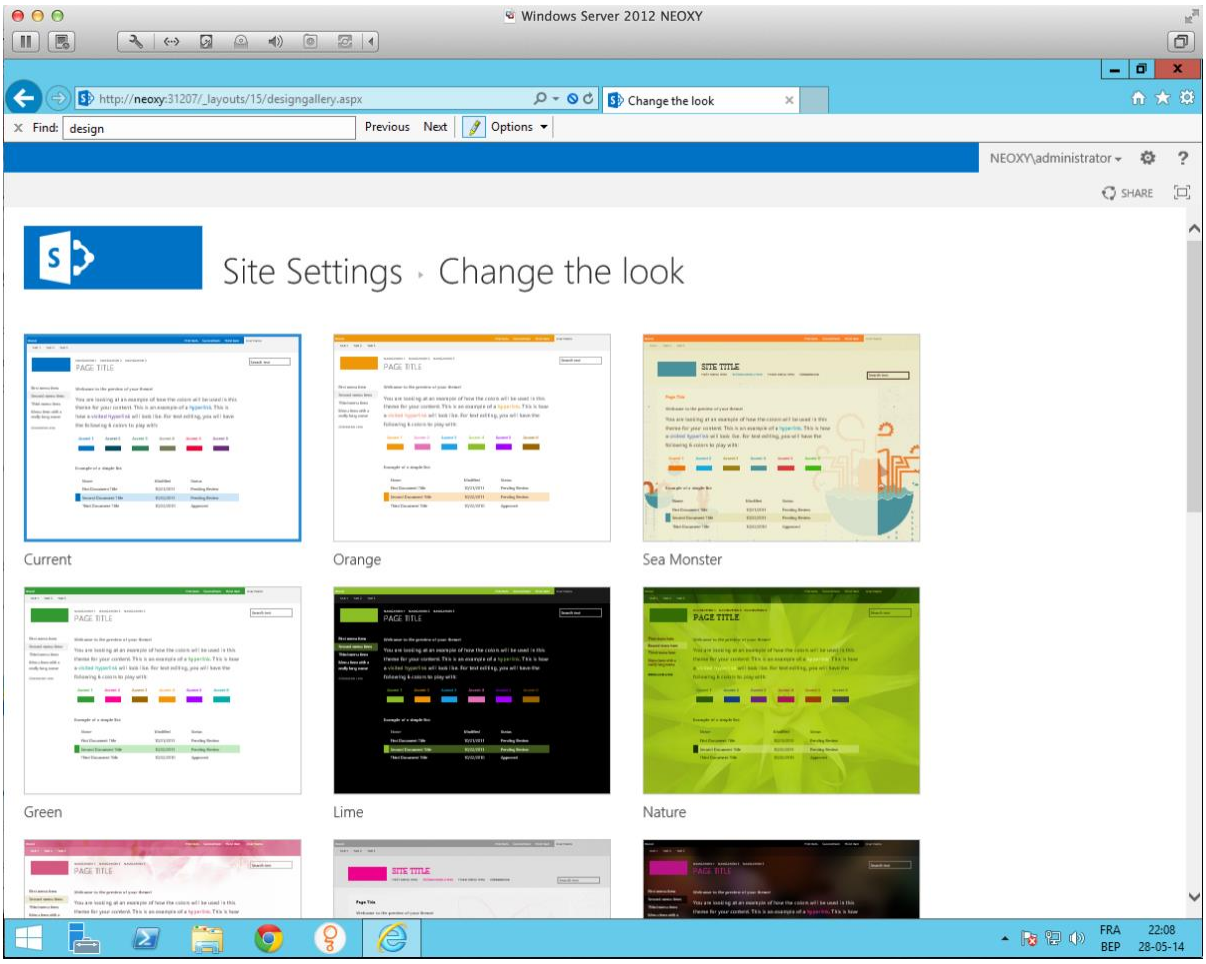


Well, if a customer wants to buy a Ferrari and sees this website, I can guarantee you that he will buy another car. Not professional at all. Everybody is talking about customizing Public facing Internet Web Sites or even intranets, but with this blog post I want to show you how to customize your Central Administration.

Therefore connect to your machine and open SharePoint 2013 Central Administration. As you can see the User Interface is very classic and same as all other Central Administrations that you ever saw in your life.

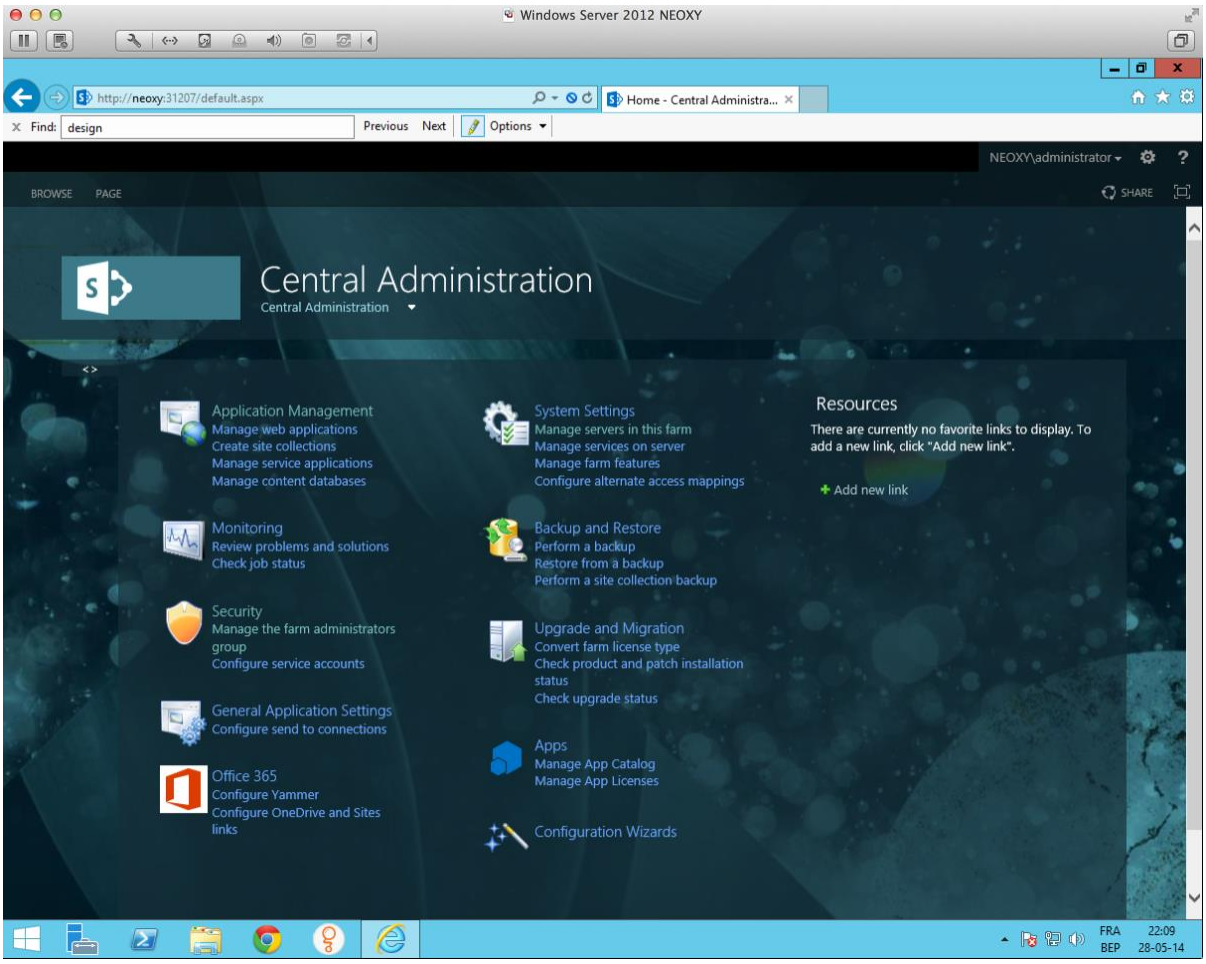


You can't see the Design Gallery under the Site Settings of your Central Administration but it's not forbidden to navigate to it by hitting the **/15/designgallery.aspx** in the browser.
You can now select a predefined template for your Central Administration and hit ok.



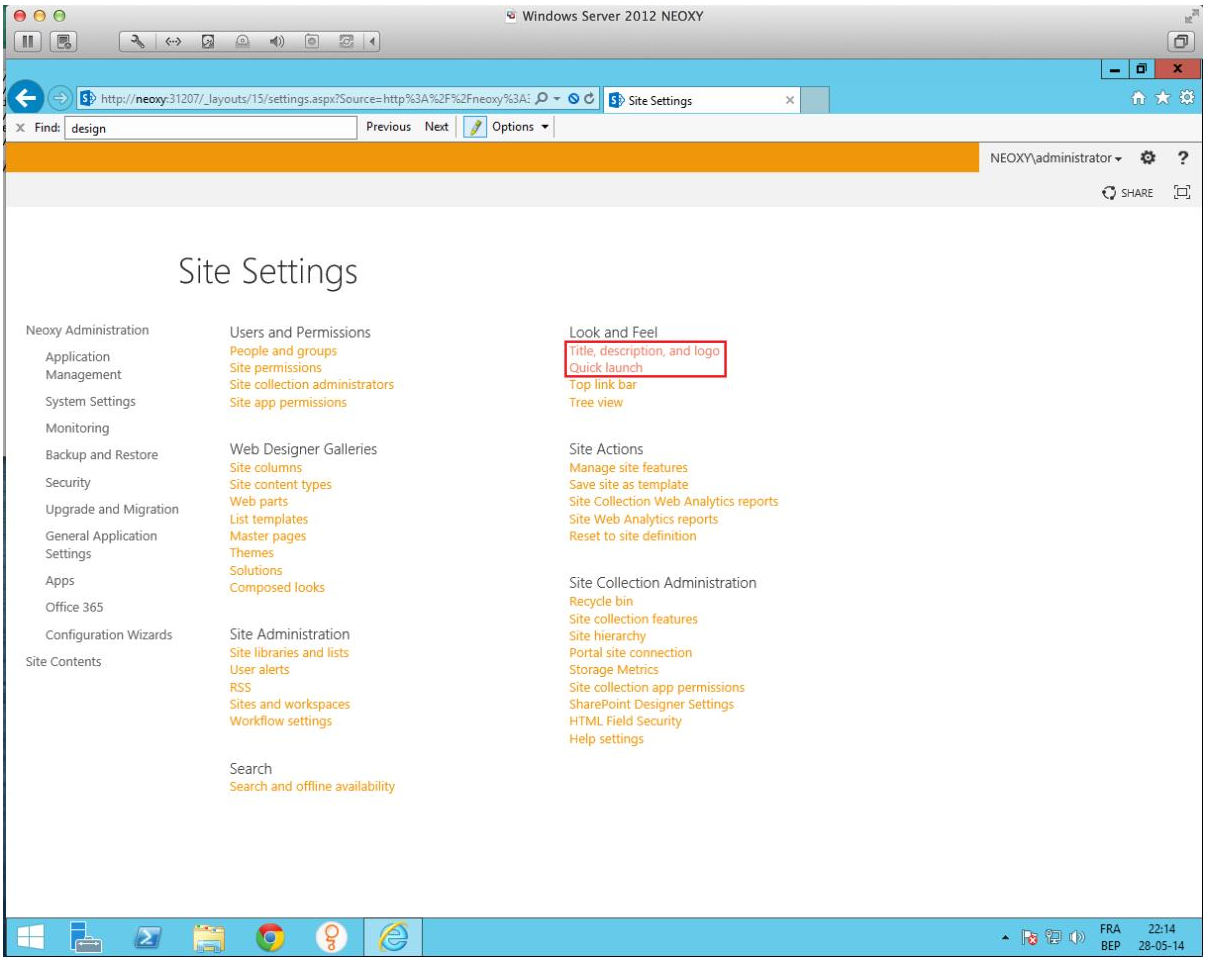
Well look this now. Your Central Administration is now a little bit different. Cool isn't it?

If you don't like the black template you can always navigate to the Design Gallery and select another one.



To be honest it's, still not personalized enough. What I want is to see is, also my company name on my Central Administration and a small logo. Why not ?

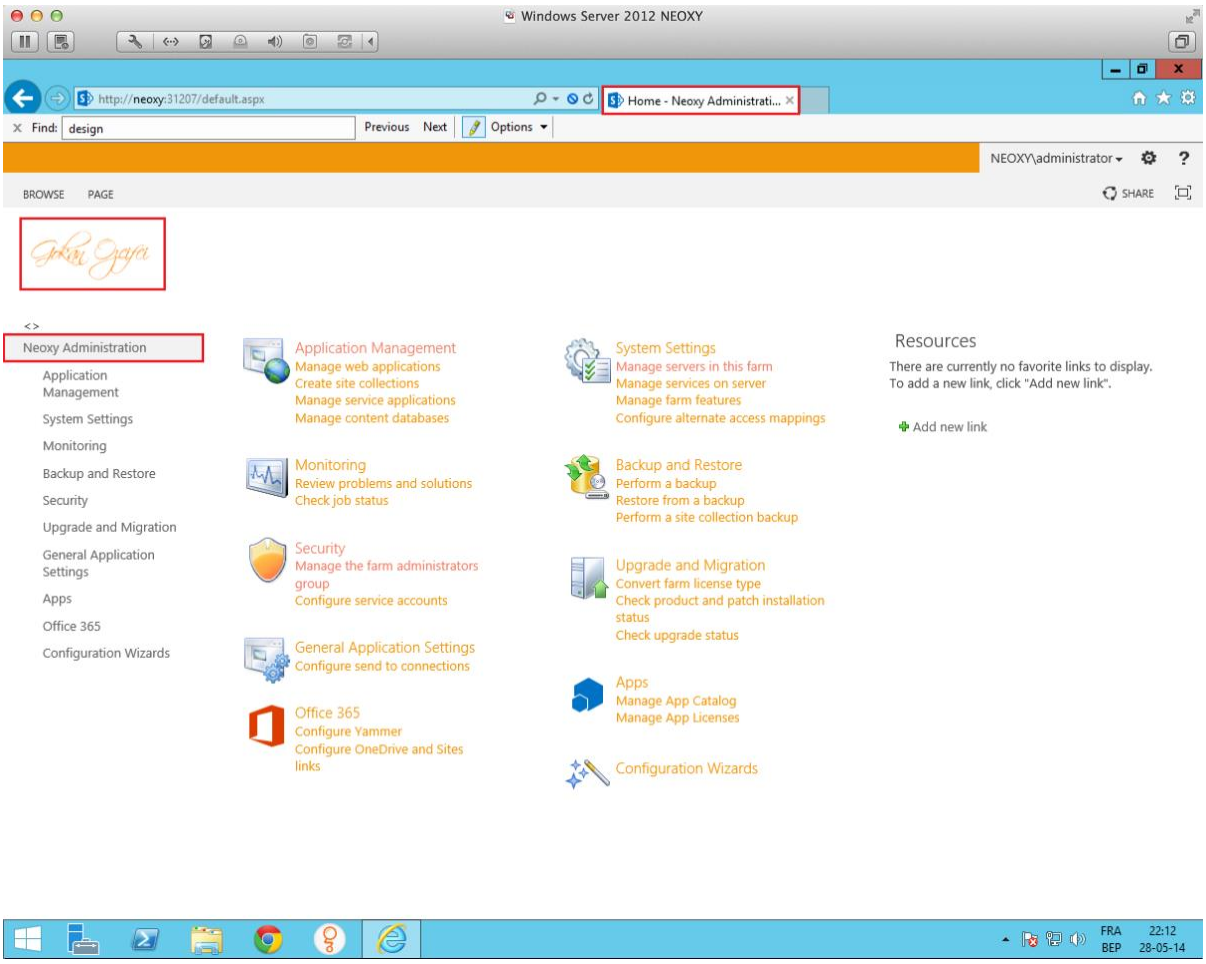
Navigate to the Site Settings and change the **Title, Description, Logo and the Quick Launch**



This is now a 100% personalized (No custom Code) Central Administration.

If you have many farms and multiple environments you can also image that the Development environment should have a « red », the Qualifying a « orange » and the Production environment a « black » template.

This will help you – visually although – remember where you are and limit the User Error Impact! A different way of governance :-):-)


Windows Server 2012 NEOXY
http://neoxy:31207/default.aspx
Home - Neoxy Administrati...
Find: design
Previous
Next
Options
NEOXY\administrator
BROWSE
PAGE
SHARE
Neoxy Administration
Application Management
System Settings
Monitoring
Backup and Restore
Security
Upgrade and Migration
General Application Settings
Apps
Office 365
Configuration Wizards
Application Management
Manage web applications
Create site collections
Manage service applications
Manage content databases
Monitoring
Review problems and solutions
Check job status
Security
Manage the farm administrators group
Configure service accounts
General Application Settings
Configure send to connections
Office 365
Configure Yammer
Configure OneDrive and Sites links
System Settings
Manage servers in this farm
Manage services on server
Manage farm features
Configure alternate access mappings
Backup and Restore
Perform a backup
Restore from a backup
Perform a site collection backup
Upgrade and Migration
Convert farm license type
Check product and patch installation status
Check upgrade status
Apps
Manage App Catalog
Manage App Licenses
Configuration Wizards
Resources
There are currently no favorite links to display.
To add a new link, click "Add new link".
Add new link
FRA
BEP
22:12
28-05-14

Adding Master Page to Auto-Hosted Apps

When we build a SharePoint hosted app, all pages and other resources are served by SharePoint from within the app web. For this reason, SharePoint hosted apps automatically inherit the look and feel of the host web and present custom functionality within the familiar SharePoint environment.

This inheritance is not automatic when we create a remote hosted app because the remote web is served from outside the SharePoint environment. There is no requirement to ensure that your remote web looks like the host web but a similar look and feel reassures users, positions familiar links and controls where users expect them, and can ease the adoption of your apptime.

Steps to add host webmaster page to app pages

1. To use the Chrome Control, copy the sp.ui.controls.js JavaScript library into your remote web project.
2. Ensure each page in app includes <script> element to the above JavaScript file.

```
<script
type="text/javascript" src="../Scripts/sp.ui.controls.js"></
script>
```

3. Add ChromeLoader.js to appweb project and provide the reference to each page in app

```
<script
type="text/javascript" src="../ChromeLoader.js"></script>
```

4. Add following div element to .aspx page

Best Practice

Use the Chrome Control whenever you want your app to blend with the host web's look and feel.
You cannot be sure that the host web will always use the familiar SharePoint look and feel so, if you hard-code the usual SharePoint design into your app, the app may not look like the customer's host web. The Chrome Control avoids this problem by downloading the style sheet and headers elements from the host web at run time.

Code: ChromeLoader.js

```
001.var hostweburl;
002.
003.//load the SharePoint resources
004.
005.$(document).ready(function () {
006.
007.    //Get the URI decoded URL.
008.
009.    hostweburl =
010.
011.        decodeURIComponent(
012.
013.            getQueryStringParameter("SPHostUrl")
014.
015.    );
016.
017.    // The SharePoint js files URL are in the form:
018.
```

```
019.     // web_url/_layouts/15/resource
020.
021.     var scriptbase = hostweburl + "/_layouts/15/";
022.
023.     // Load the js file and continue to the
024.
025.     //    success handler
026.
027.     $.getScript(scriptbase + "SP.UI.Controls.js",
renderChrome)
028.
029.});
030.
031.//Function to prepare the options and render the control
032.
033.function renderChrome() {
034.
035.     // The Help, Account and Contact pages receive the
036.
037.     //    same query string parameters as the main page
038.
039.     var options = {
040.
041.         "appIconUrl": "siteicon.png",
```

```
042.
043.         "appTitle": "My Reports",
044.
045.         //"settingsLinks": [
046.
047.         //     {
048.
049.         //         "linkUrl": "Account.html?"
050.
051.         //             + document.URL.split("?")[1],
052.
053.         //         "displayName": "Account settings"
054.
055.         //     },
056.
057.         //     {
058.
059.         //         "linkUrl": "Contact.html?"
060.
061.         //             + document.URL.split("?")[1],
062.
063.         //         "displayName": "Contact us"
064.
```

```
065.         //     }
066.
067.         //]
068.
069.     };
070.
071.     var nav = new SP.UI.Controls.Navigation(
072.
073.                                 "chrome_ctrl_placeholder",
074.
075.                                 options
076.
077.                             );
078.
079.     nav.setVisible(true);
080.
081.}
082.
083.// Function to retrieve a query string value.
084.
085.// For production purposes you may want to use
086.
087.//  a library to handle the query string.
```

```
088.
089.function getQueryStringParameter(paramToRetrieve) {
090.
091.    var params =
092.
093.        document.URL.split("?")[1].split("&");
094.
095.    var strParams = "";
096.
097.    for (var i = 0; i < params.length; i = i + 1) {
098.
099.        var singleParam = params[i].split("=");
100.
101.        if (singleParam[0] == paramToRetrieve)
102.
103.            return singleParam[1];
104.
105.    }
106.
107.}
```

HTML

```
01.<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="Default.aspx.cs" Inherits="AppWeb.Default" %>
02.
```

```
03.<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd ">

04.<html xmlns="http://www.w3.org/1999/xhtml ">

05.

06.<head runat="server">

07.

08.     <title>Home</title>

09.

10.       <script

11.

12.         src="http://ajax.aspnetcdn.com/ajax/4.0/1/MicrosoftAj
ax.js "

13.

14.         type="text/javascript">

15.

16.     </script>

17.

18.     <script

19.

20.         type="text/javascript"

21.

22.         src="../Scripts/jquery-1.9.1.min.js">

23.

24.     </script>
```

```
25.
26.       <script
27.
28.         type="text/javascript"
29.
30.         src="../Scripts/sp.ui.controls.js">
31.
32.     </script>
33.
34.     <script
35.
36.         type="text/javascript"
37.
38.         src="../ChromeLoader.js">
39.
40.     </script>
41.
42.</head>
43.
44.<body>
45.
46.     <form id="form1" runat="server">
47.
```

```
48.          <div id="chrome_ctrl_placeholder"></div>
49.
50.     <!-- The chrome control also makes the SharePoint
51.
52.           Website stylesheet available to your page -->
53.
54.         <div style="padding-left:100px">
55.
56.     <h1 class="ms-accentText"> Reports</h1>
57.
58.     <h2 class="ms-accentText"> </h2>
59.
60.     <div id="MainContent">
61.
62.     <div>
63.
64.         <h2>  </h2>
65.
66.     <hr />
67.
68.         <div>
69.
70.     </div></div></div>
```

```
71.
72.     </form>
73.
74.</body>
75.
76.</html>
```

SharePoint 2007, SharePoint 2010 and SharePoint 2013: View GUID via PowerShell

Issue: View from the View drop down on SharePoint list disappears.

Background: We had a user who had an issue with a view loading slow, very slow on a SharePoint list. This view was set as default view for the list. We noticed that there were close to 50 odd webparts on the page that was in closed state. We asked the user to delete them from webpart maintenances page by appending URL with? contents=1.

Now the problem starts

Once you try to load the list again, it loaded with the new view as lending page, and to our surprise the previous view vanished from the View drop down.

Troubleshooting

It seems that one of the webparts which was deleted from the default view had any link-up with the view so deleting webpart deleted this as well.

However, this was difficult to recreate the view as it was having allotted of customization on the page.

We checked the whole list and there was no view the same name and we are clue less what to do. We were able to open the view by typing its name after the list name; however, that was not acceptable solution for the customer.

Though of editing the view using the GUID from the URL as form the GUI GUID of lost view is not available.

To get Views from the SharePoint site URL: you can refer to the below references:

http://blogs.msdn.com/b/ronalus/archive/2007/09/08/a-little-guid-picker.aspx

http://blogs.technet.com/b/sharepointcomic/archive/2008/12/09/url-decoder-encoder-guid-converter.aspx

http://allthingsdotnet.net/960/how-to-find-list-and-view-guids

http://www.surfpointtech.com/2013/10/14/sharepoint-list-id-and-view-id-calculator

However, among these none o f my use as we had lost the edit capability from GUI being view not available in View drops down list.

So here not next reference come up to take some help from Share Master Controller, which I refer as PowerShell …! To get View GUID via Powershell and amend in the List URL to get to Views Edit mode to set it as default view to resolve the problem.

I got a very good reference of script while searching online.

```
$sourceWebURL = "http://sharepointsite&#8221;

$sourceListName = "MyList"

$spSourceWeb = Get-SPWeb $sourceWebURL

$spSourceList = $spSourceWeb.Lists[$sourceListName]

$spView = $spSourceList.Views

$spView >C:\a.txt
```

Open txt file you can see <View Name>

http://microsoftechies.com/2013/06/25/powershell-get-list-views-guid

It was my hard luck to say that problem was not limited here as this PowerShell code is for SharePoint 2010 environment. The issue were experiencing was Customer with 2007 environment.

We all know MOSS has a limited capabilities when it comes to PowerShell and everything needs to be done by calling SharePoint Object model.

Further challenge was to prepare SharePoint 2007 to work with PowerShell-

Read the post on forum to check the limited capabilities of what SharePoint 2007 got with PowerShell:http://social.technet.microsoft.com/Forums/sharepoint/en-US/01ff1eda-ee2e-4723-b375-aa54b19b00b7/how-to-install-sharepoint-powershell-to-moss-2007?forum=sharepointgenerallegacy

A old way of preparing SharePoint 2007 for PowerShell: http://nickgrattan.wordpress.com/2007/09/03/preparing-powershell-for-sharepoint-and-moss-2007/

CodePlex Reference for SharePoint 2007 for PowerShell: https://spps.codeplex.com

Instead following all these great source of information we were clue less. Thanks to our MS PFE who wrote a PoweShell code for me to get this working.

```
# load in the SharePoint assemblies
[void][System.Reflection.Assembly]::LoadWithPartialName("Microsoft.SharePoint")


# equivalent of Get-SPSite
[string]$url = "http://SharePointPortal/sites/Amar"
$collSite = New-Object Microsoft.SharePoint.SPSite($url)


# get the sub site that we want (need to change the title to whatever site you are trying to get)
#$web = $collSite.AllWebs | where {$_.Title -eq 'subSite1'}

# get the list that we need to work on (need to change the title to whatever list you are trying to get)
$rootWeb = $collSite.RootWeb

$list = $rootWeb.Lists | where {$_.Title -eq 'Tasks'}

$viewColl = $list.Views

    foreach ($view in $viewColl)

    {write-host $view.ID $view.Title $view.DefaultView  }

$collSite.Dispose()
```

Some more reference to manage SharePoint lib using PowerShell:

Deploy a PowerShell Module with SharePoint Cmdlets

http://blogs.technet.com/b/heyscriptingguy/archive/2010/09/29/deploy-a-powershell-module-with-sharepoint-cmdlets.aspx

Use PowerShell Cmdlets to Manage SharePoint Document Libraries

http://blogs.technet.com/b/heyscriptingguy/archive/2010/09/23/use-powershell-cmdlets-to-manage-sharepoint-document-libraries.aspx

SharePoint 2013: The SDDL string contains an invalid sid or a sid that cannot be translated

Recently an issue was reported to me where some of my site admins were facing an issue in setting up a SharePoint 2013 standalone environment. They were facing this error in the step 8 of SharePoint configuration wizard "The SDDL string contains an invalid sid or a sid that cannot be translated".

After spending significant amount of time this issue was finally resolved in this way:

1. Add the account you want to use for search service in WSS_Admin_WPG group. If you are going to use built in administrator account this should be present by default.
2. Then execute this command from sharepoint powershell console:
$SearchService = Get-Credential Domain\UserName

in our scenario of standalone installation the Domain should be the machine name.
3. Then execute this command
New-SPManagedAccount -Credential $SearchService
4. Now Go to "C:\Program Files\Microsoft Office Servers\15.0\Data\Office Server" and find the folder that starts with Analytics with a postfix guid.
Note: *If you selected a different location to store index files, check this path!*
5. Right-click --> Properties-->Sharing-->Advanced Sharing
6. Check the "Share this folder' check box and click on Permissions
7. Add the Search Service account created in step 2 and select Full Control. Do the same for WSS_ADMIN_WPG
8. Run the SharePoint Configuration Wizard again. It should now complete successfully.

An Example of Using Write-Progress in a Long Running SharePoint PowerShell Script

Introduction

Ever written a really cool PowerShell script that performs lots of actions, or iterates over hundreds or thousands of items? Wouldn't it be nice to see the status of the script as it's running to get an indication of the progress? You could use Write-Host to output lots of messages, like "File X just got updated...". But then your status messages are mixed through your scripts output, and you still have no clear indication of the scripts progress, or how many actions remain. In steps Write-Progress...

Write-Progress is a PowerShell Utility Cmdlet, and allows you to write a status and progress message to the top of the PowerShell console window. Write-Progress is fairly well documented on the TechNet site, and you can read about it here: Write-Progress

This article covers a simple scenario of using Write-Progress with a long running script that performs iterations of all the item versions in a SharePoint list, within a web and all of its sub-webs. It highlights how to nest progress bars and dynamically set the percent (%) complete. It also demonstrates using informative progress status messages.

The full script that this example is based on has additional functions (that allows you to delete old versions) as well as help files. The script is available for download from the TechNet Gallery: List or Delete excess ListItem Versions in a SharePoint List or Document Library

Example PowerShell Function

The following (minimized) section of the PowerShell script is commented, and contains 2 functions, and some statements (at the bottom) that call the Get-ExcessListItemVersions function. The comments should give you an understanding of how to call the Write-Progress cmdlet using different ID's and Parent ID's (for nesting the progress bars) and displaying dynamic messages and "percent complete" information.

You can copy this code into a PowerShell window on a SharePoint server and run the command (just change the SiteUrl parameter and ListTitle parameter).

```
function Get-ExcessListItemVersions{
    [CmdletBinding()]
        Param(
                [parameter(Mandatory=$true)][string]$SiteUrl,
                [parameter(Mandatory=$true)][string]$ListTitle,
                [parameter(Mandatory=$true)][int]$MaxVersions,

[parameter(Mandatory=$false)][int]$ParentProgressBarId,
                [parameter(Mandatory=$false)][switch]$Recurse
            )

    if($Recurse){
        #Display the first progress message.
        #Set the Percentage Complete to 1% using the "-PercentComplete
(1)" parameter
        #Set the Id of the progress bar to 1
        #For the activity, display a message to the user, inserting
the URL of the root site being checked, by using the $SiteUrl variable
in the "-Activity" parameter.
        Write-Progress -Id 1 -Activity "Recursively checking the web,
$SiteUrl, and all sub webs." -PercentComplete (1) -Status "Recursively
enumerating the webs to process.";
        #Because the -Recurse switch has been set, get a list of Urls,
containing the current webs Url, and any of its Sub-Webs
        $websToSearch = @();
        $websToSearch = Get-Webs $SiteUrl;
        #To update the progress bar, we need to pass the -
PercentComplete parameter a number between 1 - 100, representing the
progress of the script. Because the -Recurse switch has been set, we
need to dynamically set the maximum number of progress actions (or
```

```
steps) this script will take as it enumerates through the list of
webs.
        #Set the default number of actions and webs to 1.
        $progressActions = 1;
        $totalWebCount = 1;
        #If the Get-Webs function returns an array, the Web (specified
by -SiteUrl) contained sub-webs. So we need to set the number of
progress actions and the number of webs to the number of sub-webs.
        if($websToSearch.GetType().ToString() -ne "System.String")  {
            $progressActions = $websToSearch.Count;
            $totalWebCount = $websToSearch.Count;
        }
        $currentProgress=1;
        $currentweb =1;
        #Loop through each web in $websToSearch, and call Get-
ExcessListItemVersions using the URL of the current item.
        #When making the call to Get-ExcessListItemVersions, pass in
the ID of the current Progress Bar, as the -ParentProgressBarId
        #By setting the -ParentProgressBarId parameter, you ensure
that the current progress bar is not updated by the recursive call to
the Get-ExcessListItemVersions function. The -ParentProgressBarId
parameter tells the Get-ExcessListItemVersions to increment the ID's
of the progress bars it uses, in effect, nesting the progress bars.
        foreach($weburl in $websToSearch)
        {
            Write-Host "Searching web: $weburl $Current Item:
$currentProgress Number of Actions: $progressActions";
            #Update the parent progress bars progress indicator. You
can work out the current progress by dividing the current item in the
foreach loop ($currentProgress) by the total number of items to be
checked ($progressActions), then multiplying it by 100 to get a
percentage. This is represented by "($currentProgress/$progressActions
* 100)", which is passed to the -PercentComplete parameter
            Write-Progress -Id 1 -Activity "Recursively checking the
web, $SiteUrl, and all sub webs." -PercentComplete
($currentProgress/$progressActions * 100) -Status "Checking web
$currentweb of $totalWebCount ($weburl) ";
            Get-ExcessListItemVersions -SiteUrl $weburl -ListTitle
$ListTitle -MaxVersions $MaxVersions -ParentProgressBarId 1;
            #Increment the $currentProgress variable
            $currentProgress++;
            $currentweb++;
        }
        #Once we have finished looping through the collection of webs
to search, update the progress bar one final time, setting the percent
complete to 100%.
```

```
        #The Sleep cmdlet is used here to keep the progress bar on the
screen for a few seconds as a visual indicator that the script has
finished.
        Write-Progress -Id 1 -Activity "Recursively checking the web,
$SiteUrl, and all sub webs." -PercentComplete (100) -Status
"Finished!";
        Sleep 3;
        return;
    }

    $itemversionobj = New-Object psobject
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"WebUrl" -value ""
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"ListTitle" -value ""
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"ItemTitle" -value ""
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"ItemId" -value ""
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"Author" -value ""
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"VersionAuthor" -value ""
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"VersionLabel" -value ""
    $itemversionobj | Add-Member -MemberType NoteProperty -Name
"VersionComment" -value ""
    $versionList = $null;
    $versionList = @();

    #Set the initial ID's used for the two progress bars that will be
displayed as the script checks items.
    $outerProgressBarId = 1;
    $innerProgressBarId = 2;
    #If the -ParentProgressBarId parameter has been set, update the
$outerProgressBarId and $innerProgressBarId variables so that they
don't clash with the progressbar ID of the calling function. If we
don't do this and the progress bars we display in this function have
the same ID as the calling functions progress bar, the calling
functions progress will be overwritten when we make new calls to
Write-Progress
    if($ParentProgressBarId -ne $null)
    {
        $outerProgressBarId = $ParentProgressBarId + 1;
        $innerProgressBarId = $outerProgressBarId + 1;
    }
```

```
    else
    {
        $ParentProgressBarId = 0;
    }
    #In this part of the function, we are going to use two progress
bars.
    #The first progress bar, the Outer (or parent) progress bar, is
going to display the overall status of the function, which has four
stages (steps), which are: 1. "Getting the web", 2. "Getting the
list", 3. "Checking list items", and finally, 4. Successfully finished
parsing the list
    #The second progress bar, the Inner progress bar, is going to
display progress about each item being iterated. This progress bar
will show a message that contains the "current item of totals items",
as a visual indicator as to where the script is up (how many items of
the total number of items has been checked). The second progress bar
has its parent progress bar  (-ParentId) set to the ID of the Outer
progress bar, so that it is nested underneath it.
    $numberOfActions = 4;
    #Set the percentage of the first progress bar to 25% (calculated
from (1/$numberOfActions *100), which is effectively 1/4*100)
    Write-Progress -Id $outerProgressBarId -ParentId
$ParentProgressBarId -Activity "Processing items in $SiteUrl" -
PercentComplete (1/$numberOfActions *100) -Status "Getting the
$SiteUrl web.";
    $w = get-spweb $siteUrl
    try
    {
        #Set the percentage of the first progress bar to 50%
(calculated from (2/$numberOfActions *100), which is effectily
2/4*100)
        Write-Progress -Id $outerProgressBarId -ParentId
$ParentProgressBarId -Activity "Processing items in $SiteUrl" -
PercentComplete (2/$numberOfActions *100) -Status "Getting the list.";
        $l = $w.Lists.TryGetList($ListTitle);
        if($l -eq $null)
        {
            #If the list was not found in the current web, set the
progress to 100%, complete, and return.
            Write-Progress -Id $outerProgressBarId -ParentId
$ParentProgressBarId -Activity "Processing items in $SiteUrl" -
PercentComplete (100) -Status "List, $ListTitle, not found in the
current web, $SiteUrl";
            return;
        }
```

```
        #Get the count of items. This will be used by the second
progress bar (the inner progress bar) to calculate the percent
complete value as the items are iterated.
        $items = $l.Items;
        $count = $items.Count;
        $currentItem =1;
        #Set the percentage of the first progress bar to 75%
(calculated from (3/$numberOfActions *100), which is effectively
3/4*100). Show the user how many items are being checked by setting
the status (-Status) of the progress bar using the count of items.
        Write-Progress -Id $outerProgressBarId -ParentId
$ParentProgressBarId -Activity "Processing items in $SiteUrl" -
PercentComplete (3/$numberOfActions *100) -Status "Found the
'$ListTitle' List. Checking $count items.";
        $mltf = $l.Fields["Check In Comment"];
        foreach($item in $items)
        {
            $itemTitle = $item.Title;
            $itemAuthor = ($item.Fields["Created
By"]).GetFieldValueAsText($item["Created By"]);
            $itemId = $item.ID;
            #As each item in the collection of items is iterated
through, update the progress bars percent complete value by dividing
the current item by the total number of items and multiplying it by
100. This is represented by ($currentItem/$count*100), which is passed
to the -PercentComplete parameter. Also update the status message,
using the -Status parameter, with a message about the position of the
current item being checked within the total number of items. This is
represented by  -Status "Checking item $currentItem of $count
($itemTitle)", which uses the $currentItem, $count and $itemTitle
variables to display a dynamic message.
            Write-Progress -Id $innerProgressBarId -ParentId
$outerProgressBarId -Activity "Enumerating List Items" -
PercentComplete ($currentItem/$count*100) -Status "Checking item
$currentItem of $count ($itemTitle)";
            $excessVersions = $false;
            $versionsDeleted = 0;
            if($item.Versions.Count -gt $MaxVersions){
                $vtr = $item.Versions.Count;
                $versionsDeleted = $vtr - $MaxVersions;
                $excessVersions = $true;
                Write-Host "[$SiteUrl] $itemTitle has $vtr versions.";
                while($vtr -gt $MaxVersions){
                    $vtr--;
                    $comment = "";
```

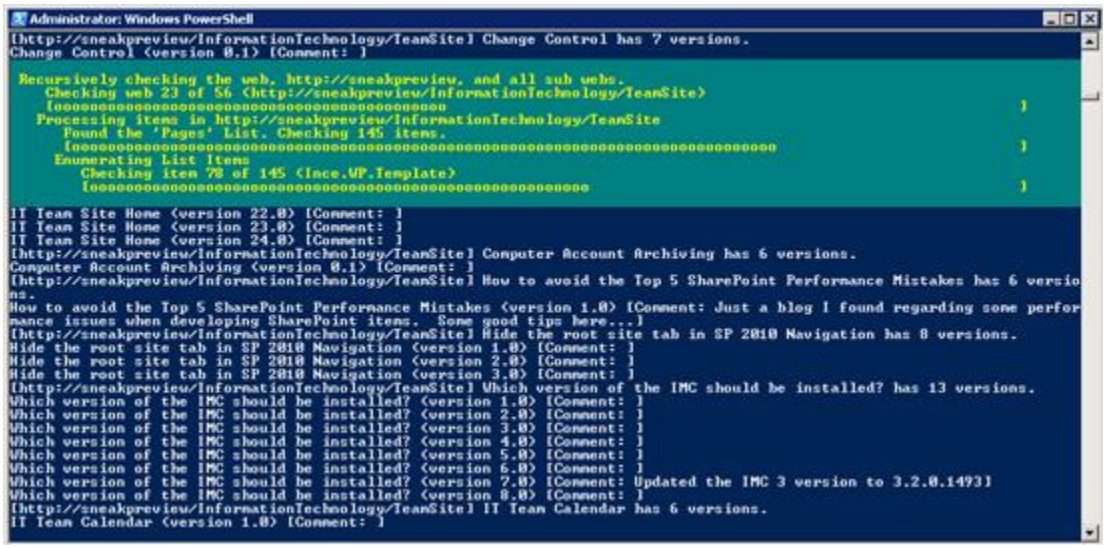
```
                    [Microsoft.SharePoint.SPListItemVersion]$iv =
$item.Versions[$vtr];
                    $versionLabel = $iv.VersionLabel;
                    $versionAuthor = $iv.CreatedBy.User.DisplayName;
                    $comment =
($mltf.GetFieldValueAsText($item.File.Versions[($versionLabel)-
1].CheckInComment)).Replace("`r`n"," ").Replace("`n"," ");
                    Write-Host "$itemTitle (version $versionLabel)
[Comment: $comment]";
                    $nvi = $itemversionobj | Select-Object *;
$nvi.WebUrl=$SiteUrl;$nvi.ListTitle=$ListTitle;$nvi.ItemTitle=$itemTit
le;$nvi.VersionLabel=$versionLabel;$nvi.VersionComment=$comment;$nvi.A
uthor=$itemAuthor;$nvi.VersionAuthor=$versionAuthor;$nvi.ItemId =
$itemId;
                    $versionList += $nvi;
                }
            }
            if($excessVersions)
            {
                #If excess versions were found, update the progress
bars current status message, without changing the percent complete
value.
                Write-Progress -Id $innerProgressBarId -ParentId
$outerProgressBarId -Activity "Enumerating List Items" -
PercentComplete ($currentItem/$count*100) -Status "Found
$versionsDeleted excess versions from the list item '$itemTitle'";
            }
            #Increment the $currentItem value before the next loop.
            $currentItem++;
        }
    }
    finally
    {
        $w.Dispose();
    }
    #Finally, update the outer (parent) progress bar with a success
message, and set the percent complete to 100%
    Write-Progress -Id $outerProgressBarId -ParentId
$ParentProgressBarId -Activity "Processing items in $SiteUrl" -
PercentComplete (100) -Status "Successfully finished enumerating items
in the $ListTitle list.";
    return $versionList;
}

function Get-Webs{
    [CmdletBinding()]
```

```
    Param(
            [parameter(Mandatory=$true)][string]$WebUrl
        )
    $w = Get-SPWeb $WebUrl;
    try
    {
        $webCollection = @();
        $webCollection += $w.Url;
        if($w.Webs.Count -gt 0)
        {
            foreach($web in $w.Webs)
            {
                $webCollection += Get-Webs -WebUrl $web.Url;
            }
        }
    }
    finally
    {
        $w.Dispose();
    }
    return $webCollection;
}
```

Example: Enumerate all of the publishing pages in the Pages library of the input web, and all sub-webs

Enumerate all of the publishing pages in the Pages library (if found) of the sneakpreview web, and all sub-webs of sneakpreview. Store all versions of a page in excess of 5 versions in the $excessVersions variable.

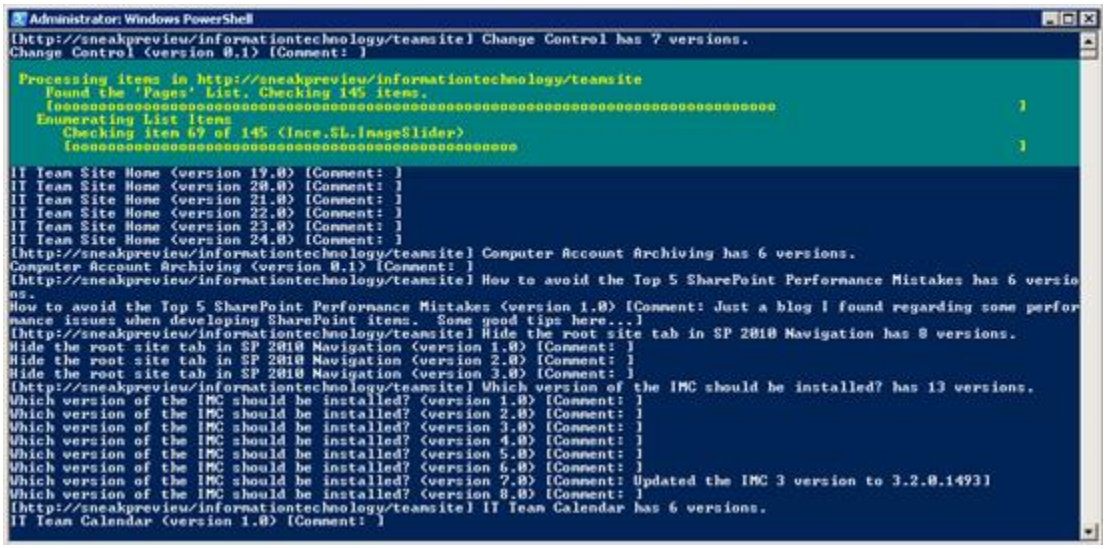
```
$excessVersions = Get-ExcessListItemVersions -SiteUrl
"http://sneakpreview" -ListTitle "Pages" -MaxVersions 5 -Recurse
```



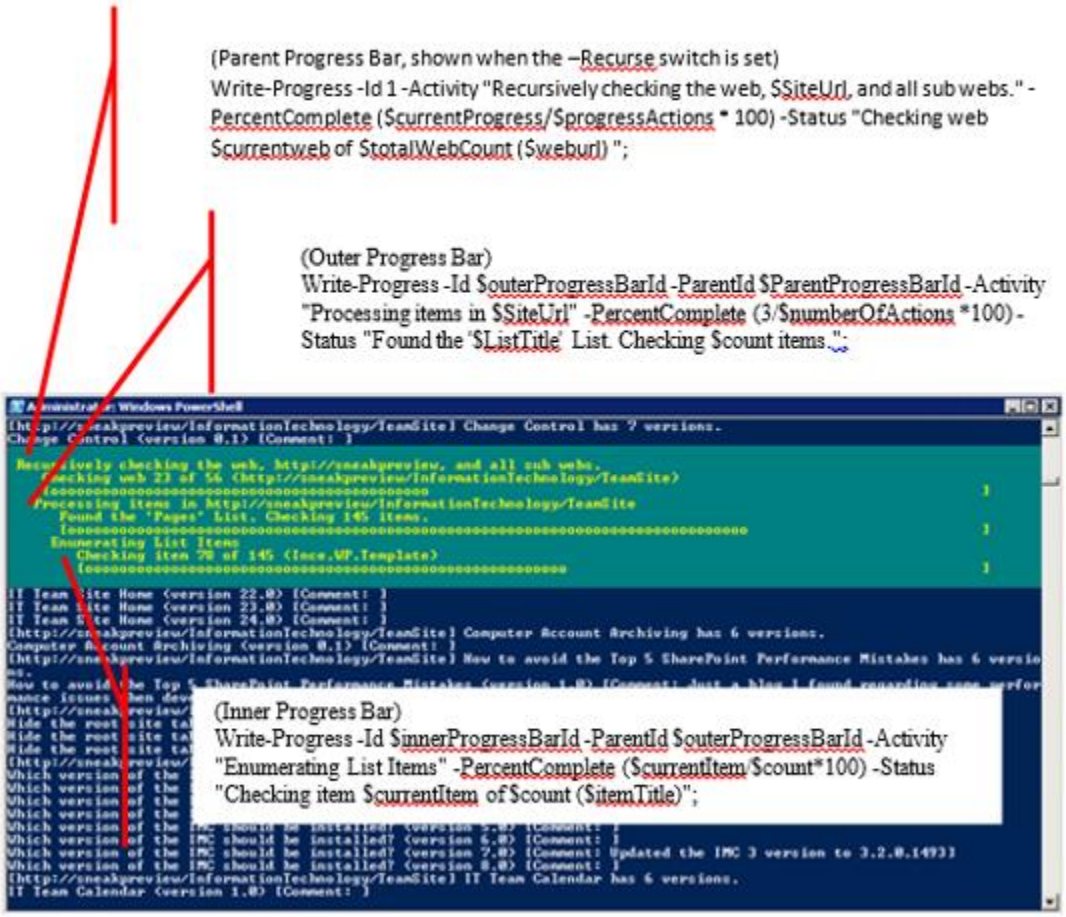
Example: Enumerate all of the publishing pages in the Pages library

Enumerate all of the publishing pages in the Pages library (if found) of the sneakpreview/informationtechnology/teamsite web.

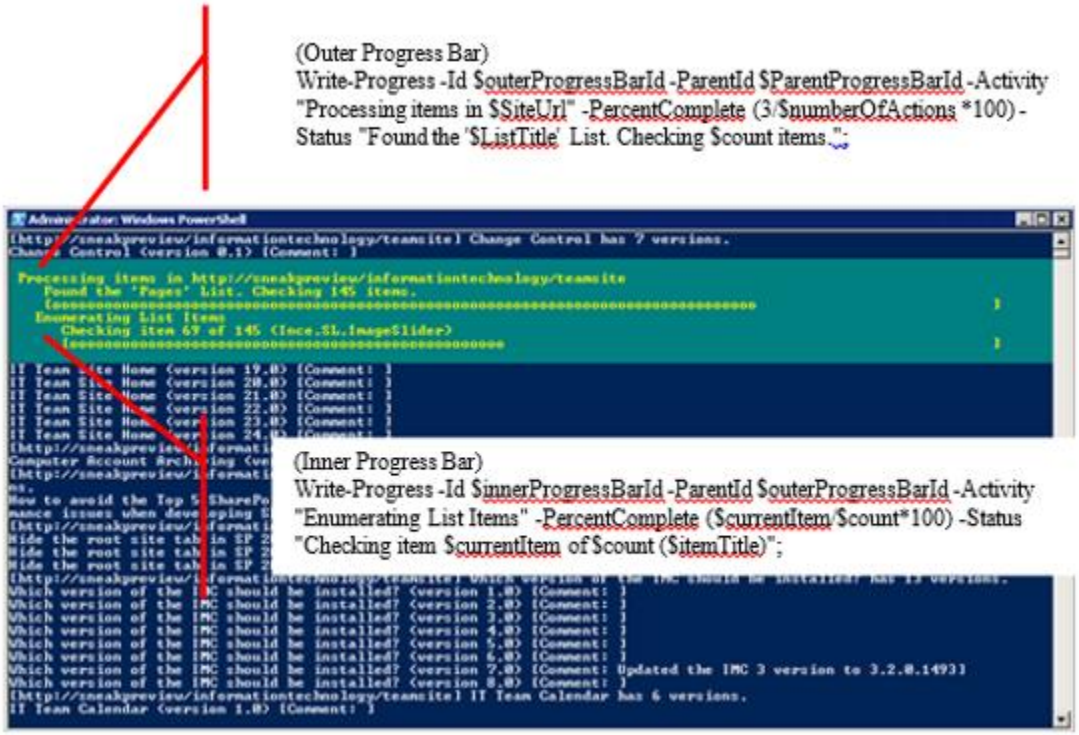
```
$excessVersions = Get-ExcessListItemVersions -SiteUrl
"http://sneakpreview/informationtechnology/teamsite" -ListTitle
"Pages" -MaxVersions 5
```



Example: The three progress bars used (with -Recurse)



Example: Two progress bars used (without -Recurse)



See Also

- Write-Progress
- PowerShell Portal
- Wiki: Portal of TechNet Wiki Portals

References

- From Forum (Progress report of my powershell script)

SharePoint: Get Set and Copy User Profile Properties using PowerShell

Introduction

Whether you're a SharePoint Administrator or SharePoint Developer, being able to quickly read, update or copy User Profile fields is a handy skill to have. Using PowerShell to get and set User Profile fields is both quick and easy. This post outlines how to do it!

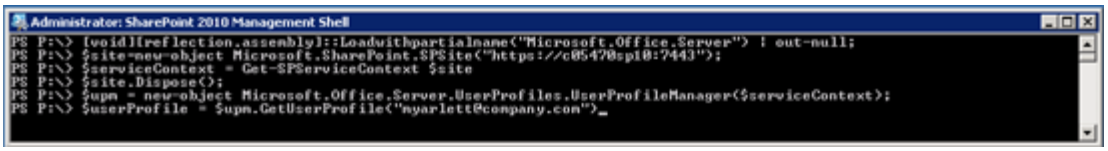
Applies To

- SharePoint 2010
- SharePoint 2013

Getting the User Profile

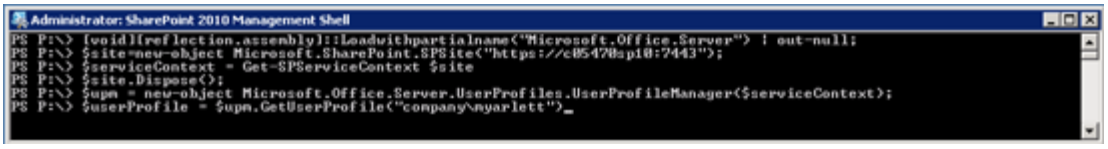
The basic PowerShell code for getting a user profile, using a users UPN (User Principal Name):

```
[void][reflection.assembly]::Loadwithpartialname("Microsoft.Office.Server");
$site=new-object Microsoft.SharePoint.SPSite("https://c05470sp10:7443");
$serviceContext = Get-SPServiceContext $site;
$site.Dispose();
$upm = new-object Microsoft.Office.Server.UserProfiles.UserProfileManager($serviceContext);
$userProfile = $upm.GetUserProfile("myarlett@company.com");
```



The basic PowerShell code for getting a user profile, using the user's login name:

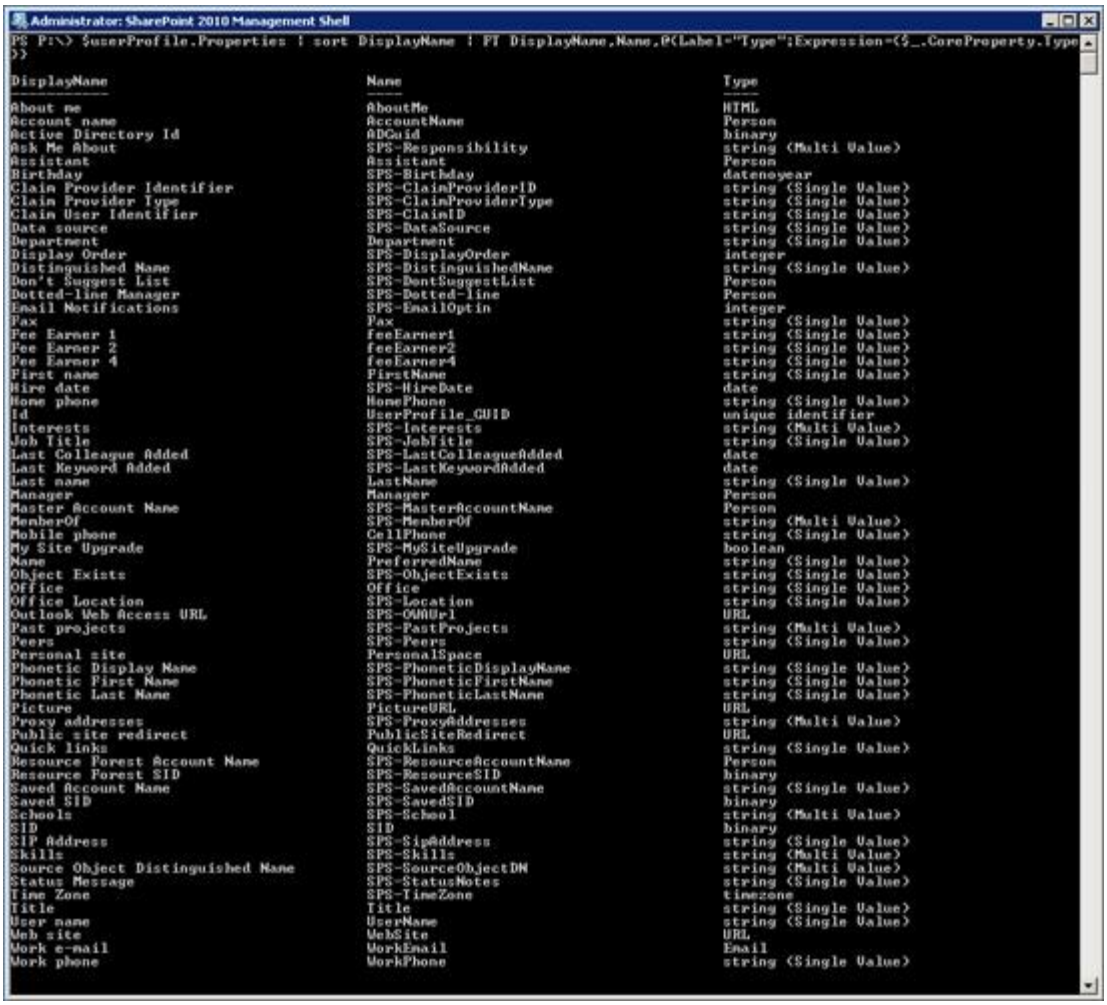
```
[void][reflection.assembly]::Loadwithpartialname("Microsoft.Office.Server");
$site=new-object Microsoft.SharePoint.SPSite("https://c05470sp10:7443");
$serviceContext = Get-SPServiceContext $site;
$site.Dispose();
$upm = new-object Microsoft.Office.Server.UserProfiles.UserProfileManager($serviceContext);
$userProfile = $upm.GetUserProfile("company\myarlett");
```



Listing all the Profile Properties (Fields) and their Types

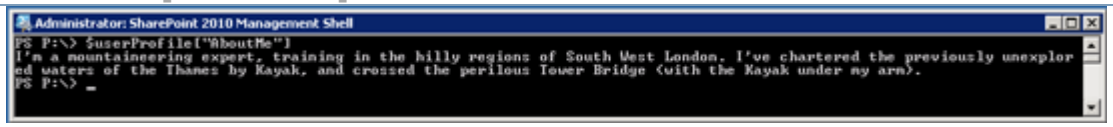
List the user profile properties (including the field type). This is handy, because we'll need to know what the field type is before trying to set it's value:

```
$userProfile.Properties | sort DisplayName | FT DisplayName,Name,@{Label="Type";Expression={$_.CoreProperty.Type}}
```



Getting the Value of a Property
Get the users About Me property (HTML):

```
$userProfile["AboutMe"].Value
```



Setting the Values of Properties

Update the users Location (String field):

```
$userProfile["SPS-Location"].Value = "London";
$userProfile.Commit();
```

Update the users Manager (Person field):

```
$userProfile["Manager"].Value = (Get-SPWeb
https://c05470sp10:7443).EnsureUser("company\fred");
$userProfile.Commit();
```

Note that in the above example, we have retrieved an SPUser object (for the manager) from the Central Admin site, using the EnsureUser method.

Add a value to the About Me property (multi-string)

```
$rp = $userProfile["SPS-Responsibility"]
#Print out the current values
foreach($s in $rp){$s}
#Add a new value to the UserProfileValueCollection
$rp.Add("Awesomeness");
#Set the SPS-Responsibility property with the
UserProfileValueCollection
$userProfile["SPS-Responsibility"].Value = $rp;
#Save the profile changes back to the User Profile store
$userProfile.Commit()
```

Clear all values in the About Me property

```
#Delete all values in the UserProfileValueCollection
$rp.Clear()
#Set the SPS-Responsibility property with the
UserProfileValueCollection
$userProfile["SPS-Responsibility"].Value = $rp;
#Save the profile changes back to the User Profile store
$userProfile.Commit()
```

Copying User Profile Properties between Profiles

Copy fields from one user profile to another:

```
$userProfile2 = $upm.GetUserProfile("company\matthewette");
$userProfile2["AboutMe"].Value = $userProfile["AboutMe"];
$userProfile2.Commit();
```

See Also

- UserProfileManager class
- SPWeb.EnsureUser method
- PowerShell Portal

References

- Original content from Matthew Yarlett's blog, A Quick Guide to Getting, Setting and Copying User Profile Properties using PowerShell
- Forum content; Copy Sharepoint User Profile properties from one account to another

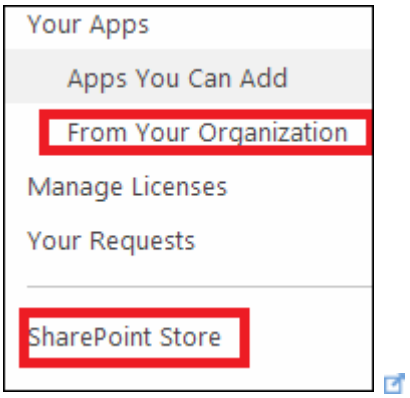
SharePoint 2013 and Office 365 (SharePoint Online) App Provisioning and Installation Options

Introduction

App deployment is different from app development. App deployment depends on how administrator is going to install the app. App deployment can be done in:

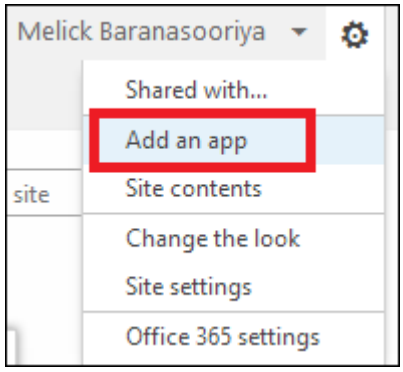
- Web Scope
- Tenant Scope

For both deployments we need **App Catalog** in functioning. Apps are currently coming from two main sources as **From Organization** and **From SharePoint Store.**

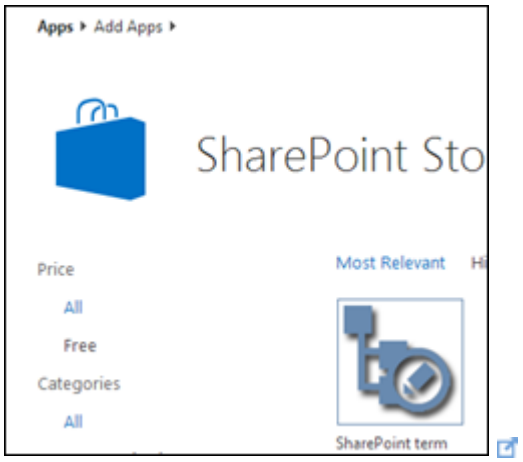


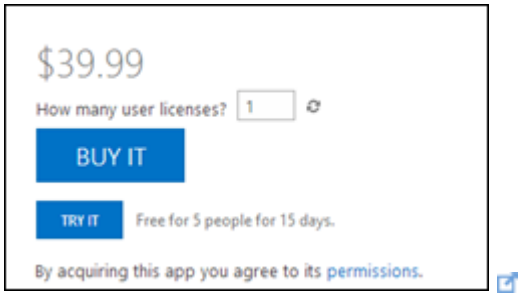
Web Scope Installation

This is what we generally do. Here we are navigating to desired web (assume the web is http://www.sharepoint.com/sites/myweb) and click the setting button and launch Add an app:

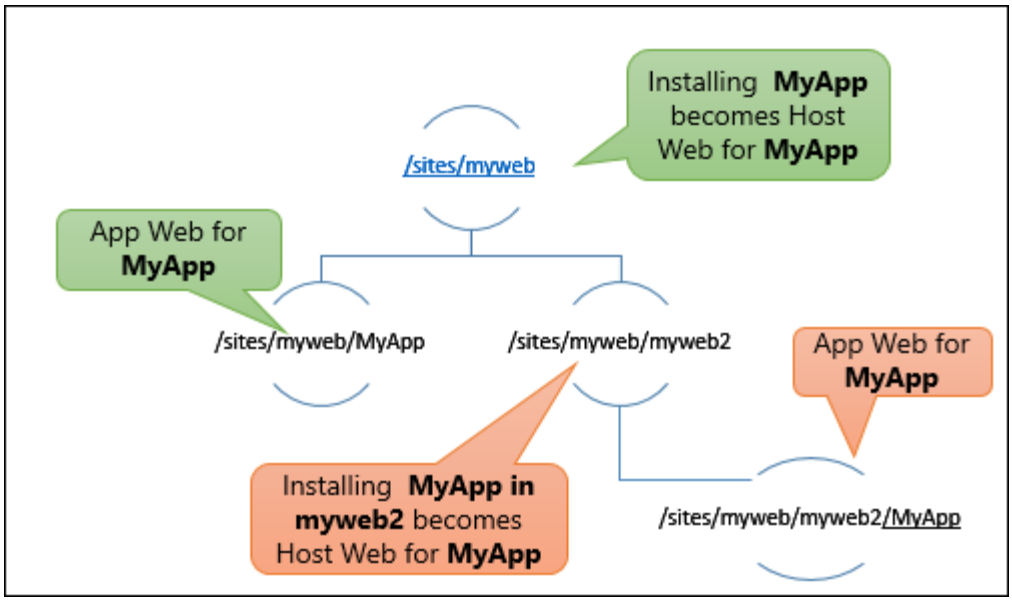


Then you can either browse or install app from office store or organization store:



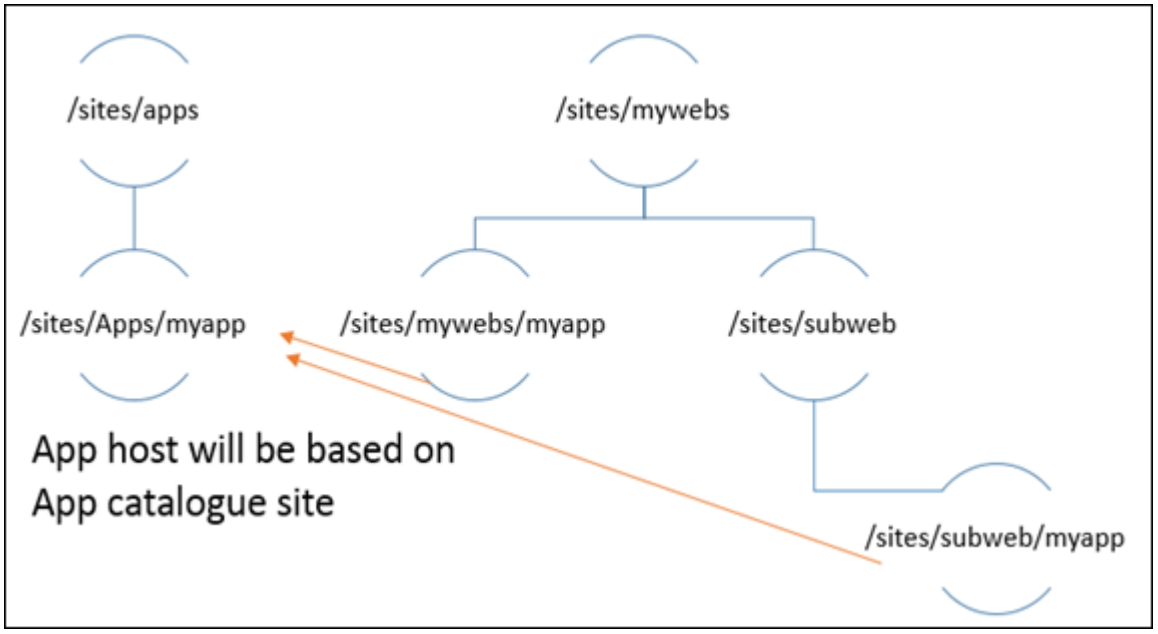


In this approach app is installing as a **sub web** under **current web** and data will be scoped to the particular web we installed. The web site which actually holds app data and functionality is called App Web (sub web) and the web site we installed the app is called as Host Web (current web):



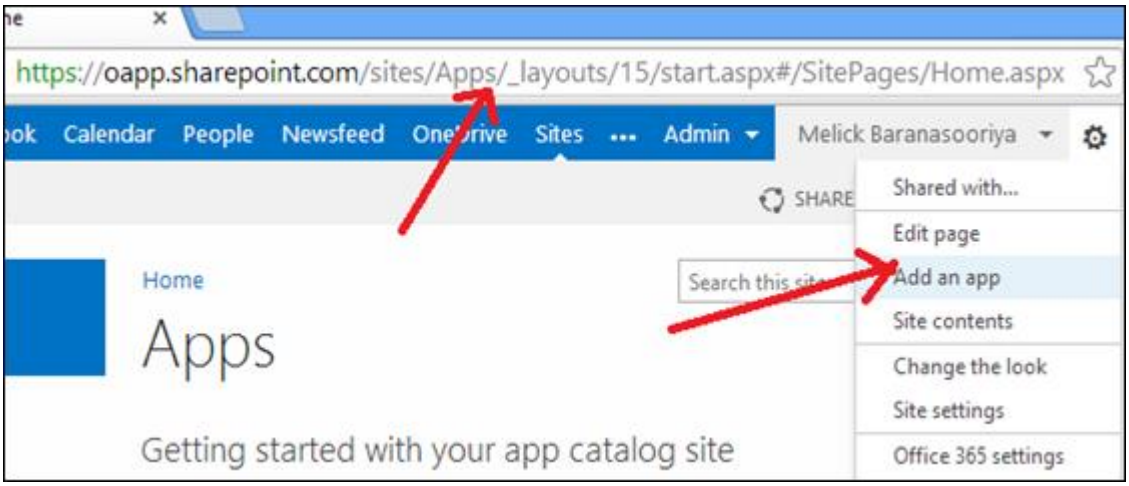
For an example assume installing MyApp under myweb. In this scenario **/sites/myweb** becomes a **Host Web** for myApp and installing MyApp provisioning a sub web **(/sites/myweb/MyApp)** that becomes a **App Web** for particular MyApp. Installing MyApp under sub site; **sites/myweb/myweb2** resulting;**sites/myweb/myweb2** been a Host Web for particular MyApp instance and **/myweb/myweb2/Myapp** as a App Web for that MyApp instance.

Tenant Scope Installation

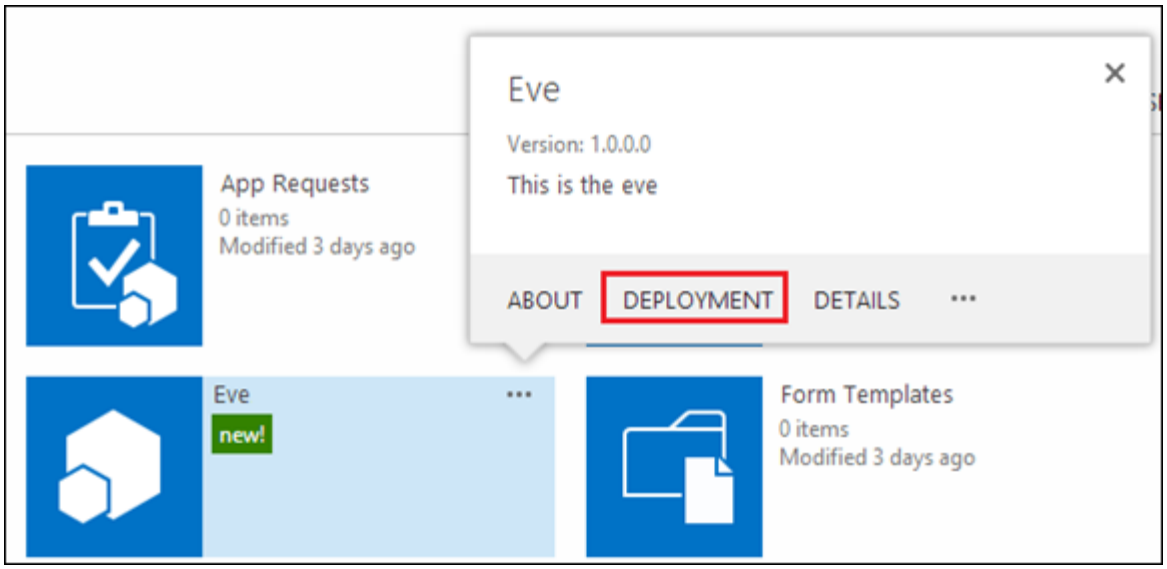


This installation is different from web scope installation. Here we are installing app in **App Catalog Site** and then it becomes Host Web for that application.

First you need to navigate to your **App Catalog Site Collection** and perform app installation in that site.



Then select and installed the app. After that go to **Site Contents** and open the call out menu for that app. You can see Deployment Menu.

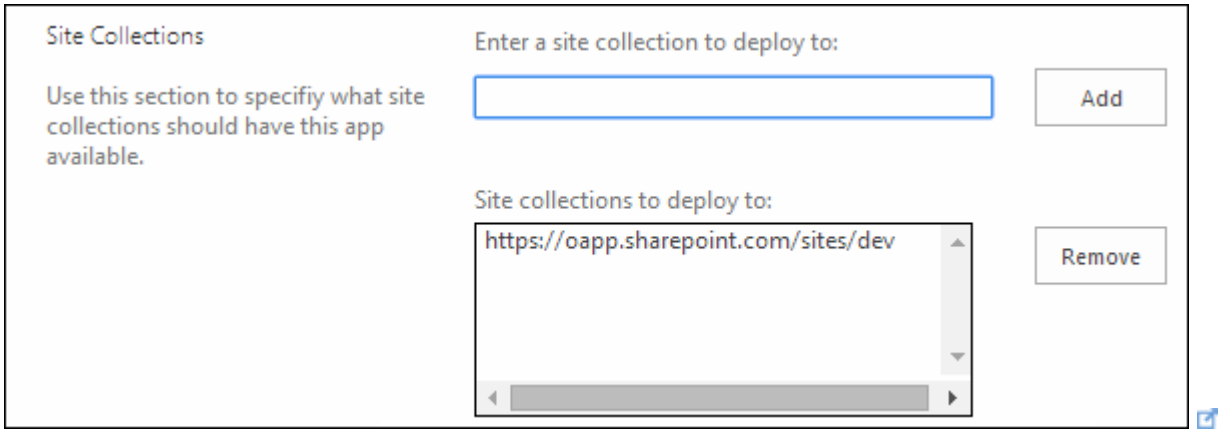


If you click on that you can see following tenant deployment options.

- Site Collections
- Managed Paths
- Site Templates

Site Collections

Here you can specify which site collections you need to deploy. You can enter multiple sets of site collections. Thus app will be automatically provisioned under all webs inside that site collection.



Managed Paths

This will provision the app in webs under that manage path.

Site Templates

This will provision the app for webs that inherits from mentioned template.

See Also

- SharePoint 2013 Portal
- SharePoint 2013 - Service Applications
- SharePoint 2013 - Resources for Developers
- SharePoint 2013 - Resources for IT Pros

Step by step Installation & Configuration of Workflow Manager (SharePoint 2013)

Introduction

I have uploaded a document in TechNet Gallery which contains detail steps for configuration & Installation of SharePoint 2013 Workflow Manager .

Content

- Overview of Workflow Manager
- Prerequisites
- Pre-install steps
- Installation of Workflow Manager
- Configuration of Workflow Manager & Service Bus
- Install and configure Workflow Manager Client
- Configure the SharePoint farm with the workflow farm

Overview

- Workflow Manager 1.0 allows you to host and manage long-running Windows Workflow Foundation (WF) applications.
- It extends the functionality originally delivered with Windows Workflow Foundation.
- Designed to meet the scale and multi-tenant needs of your modern enterprise applications
- Workflow Manager is the backing technology for SharePoint Workflows in SharePoint Server 2013 and next version of Office 365.
- Increase developer and administrator productivity
- Workflow Manager 1.0 is also supported in Windows Azure Virtual Machines.

Prerequisites

- Web Platform installer checks for these prerequisites.
    - .Net Framework 4 Platform Update 3 or .NET Framework 4.5
    - Service Bus 1.0
    - Workflow Client 1.0
    - PowerShell 3.0
- Additional requirements must be met before you can run Configuration Wizard to configure Workflow Manager
    - Instance of SQL Server 2008 R2 SP1, SQL Server Express 2008 R2 SP1, or SQL Server 2012
    - TCP/IP Connections or named pipes must be configured in SQL Server.
    - Windows Firewall must be enabled.
    - Ports 12290 and 12291 must be available.

Pre-install Steps

- Create setup, service accounts and Workflow Group in AD
- Create Accounts on AD
    - *WFSetup*
    - *WFService*
- Create Workflow Administration Group in AD
    - *WFGroup*
- Add SharePoint setup Admin account to WFGroup in AD
- Add WFSetup account to SQL Server security group and give him SysAdmin permission
- Add WFSetup to Workflow server in machine administrator group, so user can login as workflow administrator.

Document Source

You can find further detail on Installation & configuration of workflow Manager in TechNet Gallery .

SharePoint: Testing Email Alerts in UAT and DEV Environments

Introduction

This post describes installing an SMTP server (SMTP4Dev) that can be used to capture all email sent from a SharePoint farm, without delivering the email to the recipients mailbox.

This is handy to have setup in a Development or UAT (user acceptance testing) environment, where you want to test or analyse email alerts sent from SharePoint, without actually having the emails delivered to the end user.

Applies To

- SharePoint 2013
- SharePoint 2010

The two scenarios presented here

1. Installing and configuring SMTP4Dev on a SharePoint server (or other server) that doesn't already have a process listening on port 25 (e.g. the IIS SMTP service)
2. Installing and configuring SMTP4Dev on a SharePoint server that already has a service listening on port 25 (e.g. the IIS SMTP service)

Introduction to SMTP4Dev

SMTP4Dev is a console application used to receive email via SMTP. Email received by SMTP4Dev can be inspected or deleted. However, SMTP4Dev does not deliver email to a destination mailbox. It can listen on any port (the default port is 25), and will accept email while it's running. It can be configure for anonymous or authenticated connections.

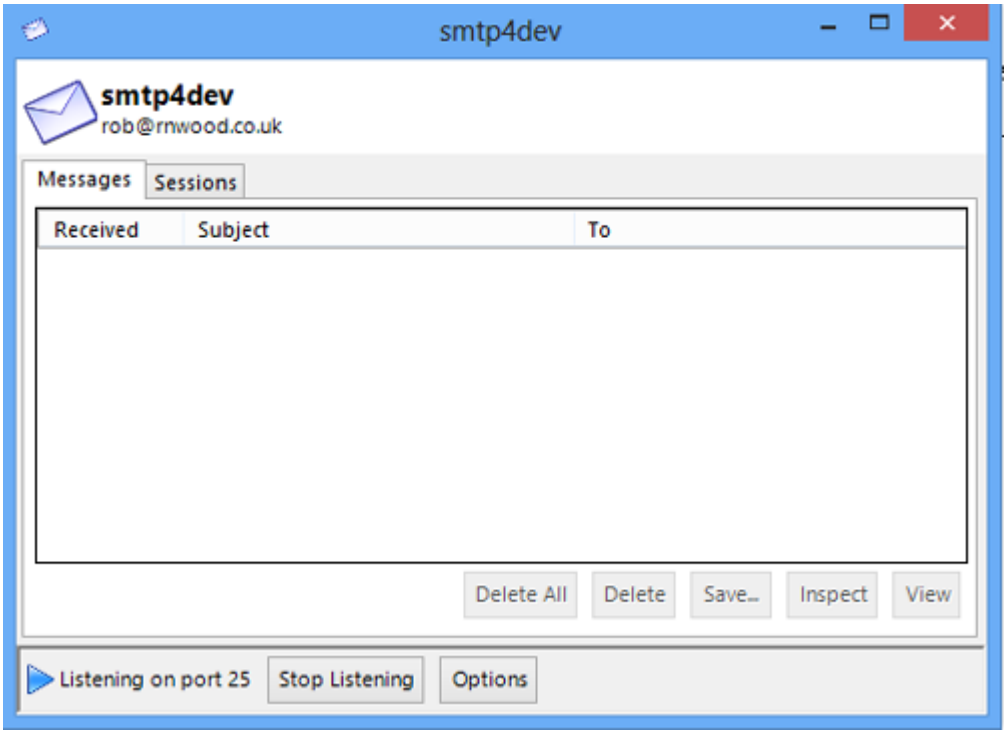
SMTP4Dev is great for being able to capture and analyse emails sent in a UAT or Development environment, without needing an email infrastructure (like Exchange and Outlook). Downloaded SMTP4Dev here: http://smtp4dev.codeplex.com/

Installing SMTP4Dev

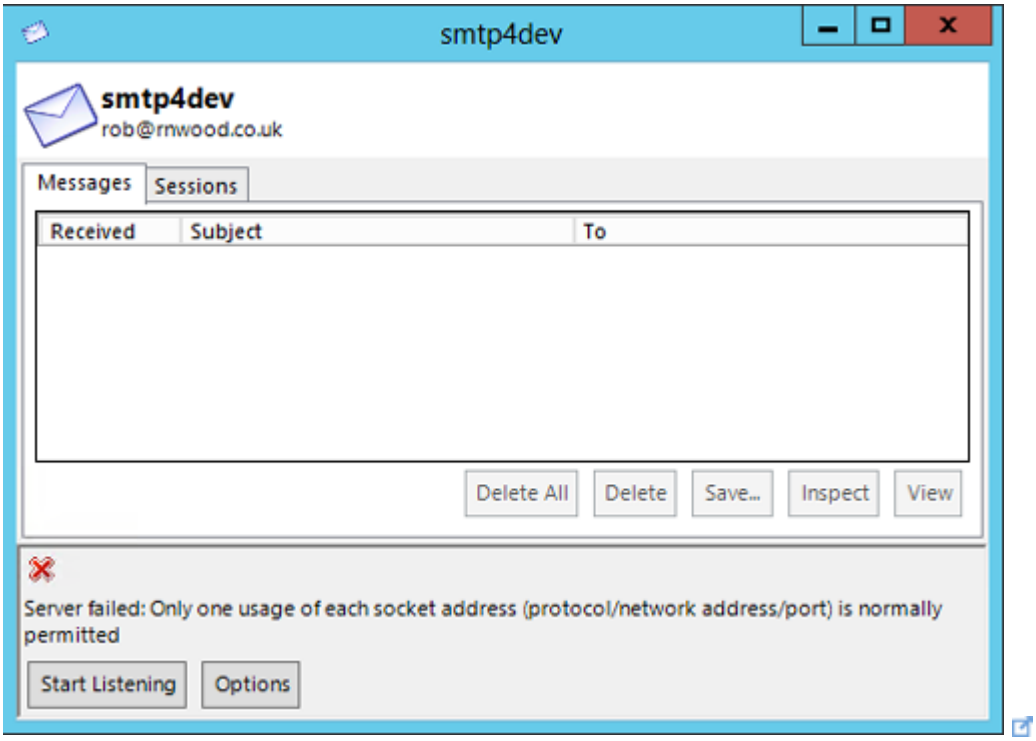
Installing and running SMTP4Dev is very simple. Follow these steps:

1. Download SMTP4Dev from codeplex (http://smtp4dev.codeplex.com/) and save the zip file to the local file system. E.g. C:\Tools\smtp4dev.zip
2. Extract the zip files contents to the same location
3. Double click the smtp4dev.exe
4. Done!

When SMTP4Dev is opened, it will start listening on Port 25 by default.



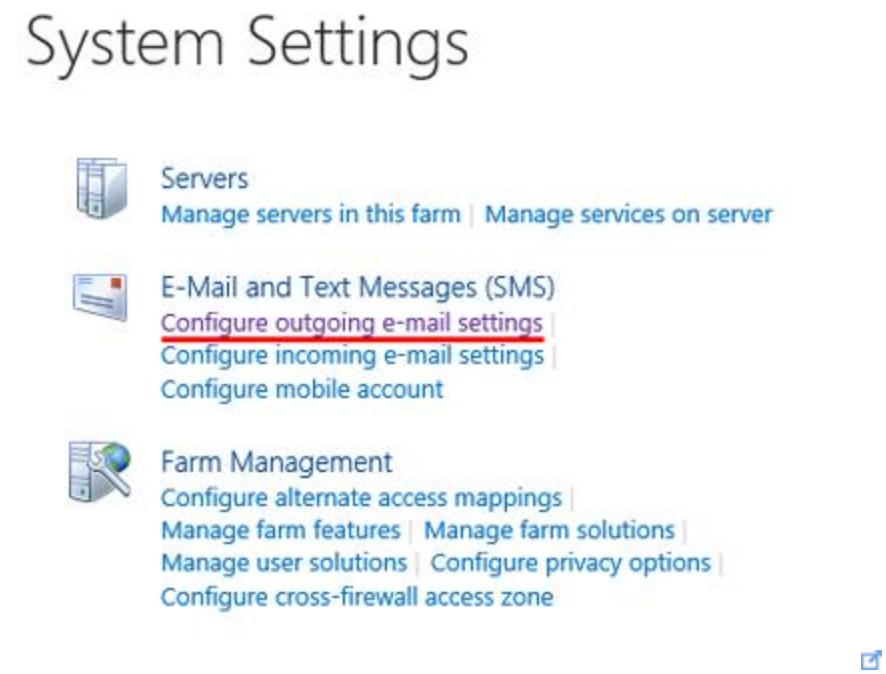
If there is already another process using port 25 (e.g. the IIS SMTP service), SMTP4Dev will show an error message about a socket address (the configure port is already in use).



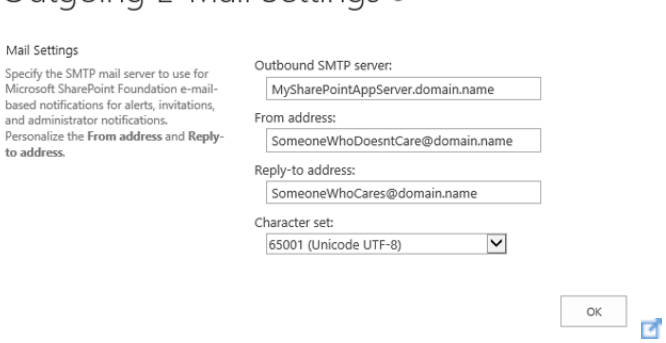
Configure SharePoint Outbound email with SMTP4Dev running on Port 25
If no other process is running on port 25, SMTP4Dev will use the default SMTP port (25) to listen on. Once you start SMTP4Dev, by default it will begin to listen for email on port 25, using anonymous authentication.

To finish the configuration, set the Outbound Email server for the farm to the FQDN (Fully Qualified Domain Name) of the SharePoint application server running SMTP4Dev.

1. Open the Central Admin site
2. Click on System Settings
3. Click on Configure outgoing e-mail settings



4. Enter the FQDN for the SharePoint application server that is running SMTP4Dev, into the Outbound SMTP Server textbox.



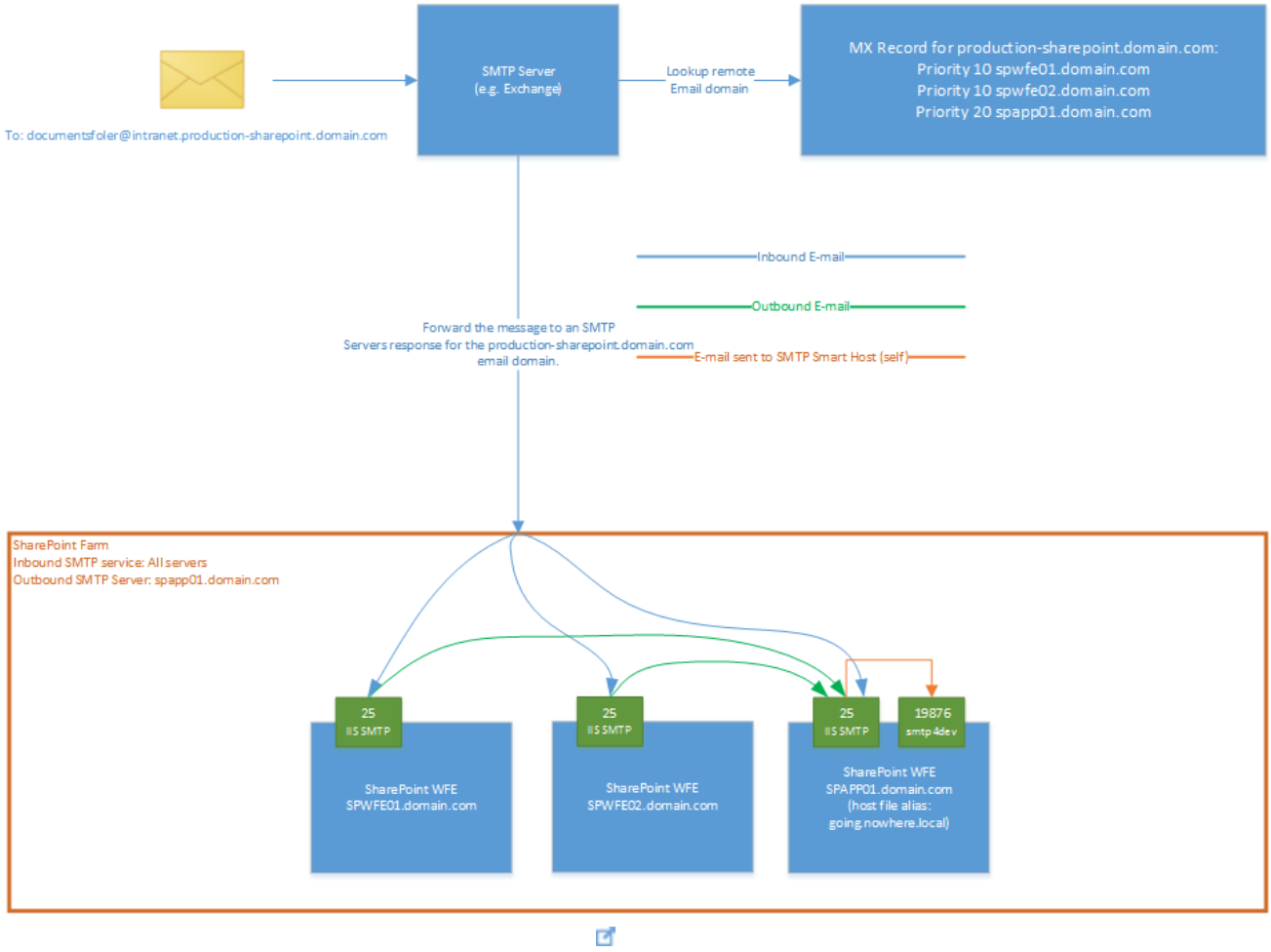
5. Set the From address and Reply-to address
6. Click OK to save the settings.

Configure SharePoint Outbound email with SMTP4Dev running on a Custom Port

If there is already another process listening port 25, you will need to configure SMTP4Dev to listen on another port. Configuring SMTP4Dev to listen on another port is easy. However, SharePoint will only send outbound email to an SMTP server listening on Port 25. So additional configuration is required to get this working.

Configuring SMTP4Dev to listen on a custom port might happen in a scenario where all of the SharePoint servers in a UAT or Development farm already have the IIS SMTP service installed (for inbound email).
Consider the following scenario:

- The SharePoint application server that SMTP4Dev is installed on has the host name SP13App01.mydomain.com
- The SharePoint Application server, SP13App01.mydomain.com, already has the IIS SMTP service configured (on Port 25) for Inbound Email (functionality that allows people to email a SharePoint folder).
- The IIS SMTP service is configured to receive email for the SharePoint farms domain alias (e.g. production-sharepoint.mydomain.com)
- The Outbound email SMTP server for the farm is set to sp13app01.mydomain.com
- The IIS SMTP server on sp13app01 is configured to relay email for all remote domains to the SMTP smart host going.nowhere.local, over port 19876.
- There is a dns entry in the hosts file on sp13app01 for going.nowhere.local that uses the localhost IP address 127.0.0.1
- SMTP4Dev is configured to listen on Port 19876



In this scenario, any email sent to an address in the production-sharepoint.mydomain.com will be picked up by one of the SharePoint servers running the IIS SMTP service (on Port 25) and saved into the IIS mail Drop folder (for SharePoint to process). If an email sent to the SMTP server isn't addressed to the production-sharepoint.mydomain.com domain, the IIS SMTP server will;

- (WFE servers): discard the email
- (Application server): forward (relay) the message to the SMTP smart host.

An SMTP smart host is an SMTP server that will accept email for any domain from a source SMTP server (the sending SMTP server), and forward that email to a destination SMTP server responsible for the emails domain.

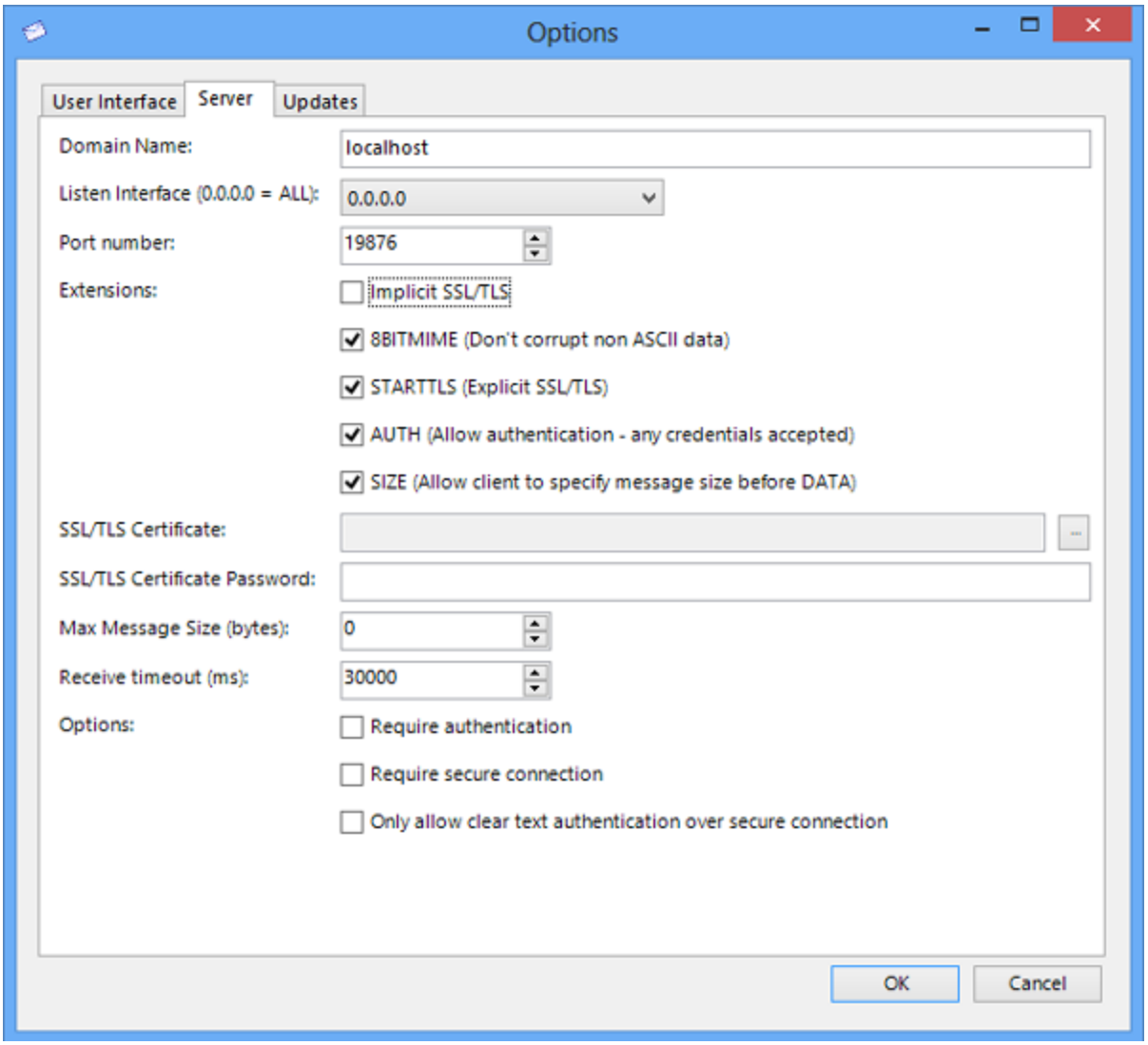
In practice, the IIS SMTP on the SharePoint WFE servers should never receive email addressed to a foreign domain. However, the IIS SMTP service on the SharePoint application server will receive email addressed to the production-sharepoint.domain.com domain, as well as emails sent from the SharePoint farm itself. This is because the SharePoint server sp13app.domain.com has been set as the outgoing email server for the farm.

In this scenario, SMTP4Dev is configured as the Smarthost. It will receive all emails sent to the SharePoint Application server's IIS SMTP service (running on Port 25) that are destined for a foreign domain (e.g. a list alert configured to send new list item alerts to a domain user).
To configure SharePoint for this type of scenario, follow these steps;

1. Configure SMTP4Dev to listen on a custom port
2. Add a host alias to the hosts file of the server running SMTP4Dev
3. Configure the IIS SMTP service on the SharePoint application server to relay all email destined for a foreign domain to the host alias configured in step 2
4. Configure the SharePoint application server as the outbound SMTP server for the farm.

Configure SMTP4Dev to listen on a custom port

1. Open SMTP4Dev
2. Click Options
3. From the Options dialog, click the Server tab
4. Change the Port Number to a custom port value between 1025 and 65000. In the example, we use port 19876



Note: Whatever port you choose, make sure no other process is listening on that port. To check this, you can use the NETSTAT command, and pipe the results to the FIND command. For example, so check if a process is listening on port 19786, use the following command at a command prompt.

Netstat –a | Find ":19876"

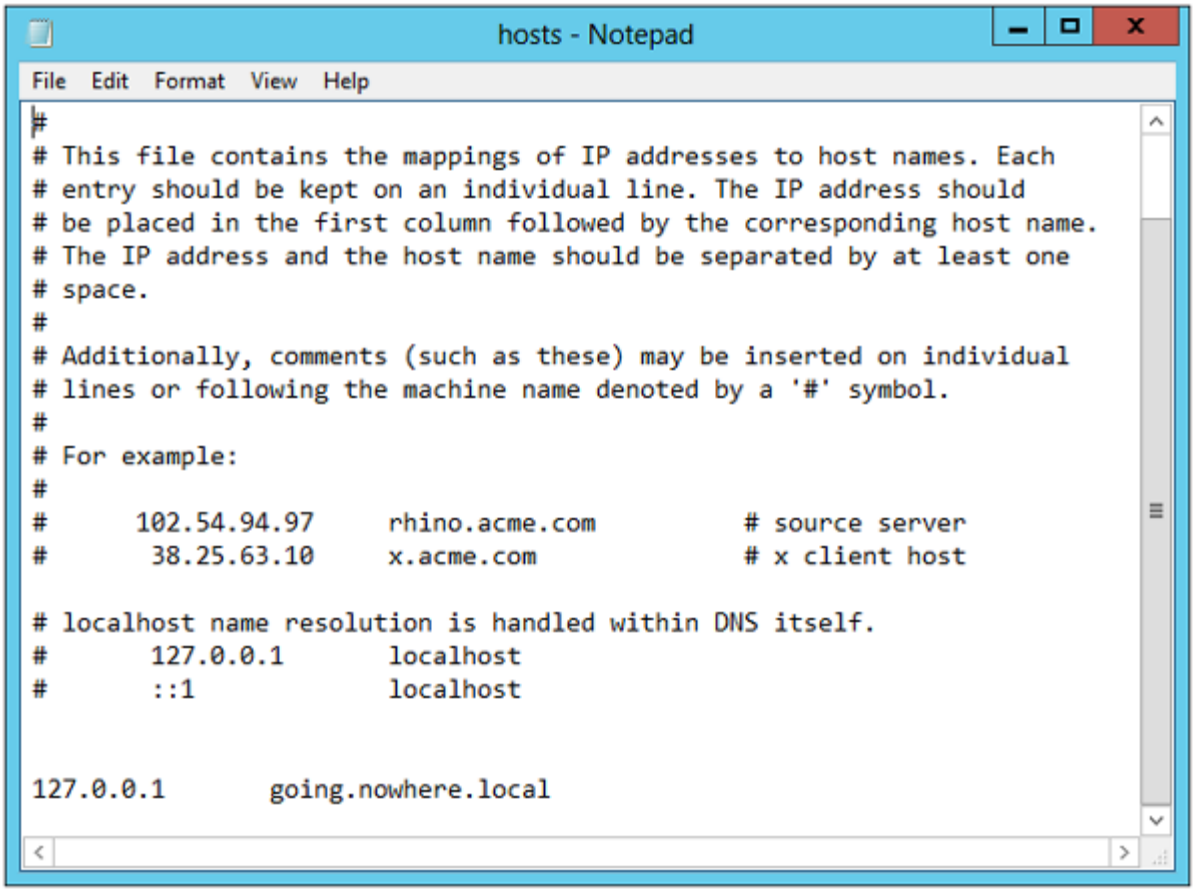
Add a host alias to the host file

Each SharePoint server in the farm is configured to send outbound email to a single SharePoint application server. The IIS SMTP service on the SharePoint application server is configured to forward (relay) all e-mail for foreign domains to a Smart Host. The "Smart Host" is actually an SMTP service (SMTP4Dev) running on itself, listening on a different port.

When you configure the Smart Host in the IIS SMTP services, the UI prevents you from adding the hostname of the current server as the Smart Host.

To work around this limitation, add an alias to the hosts file on the SharePoint application server. The alias can be anything (though it shouldn't be a hostname used anywhere else), but the IP address must be set to the local server (127.0.0.1).

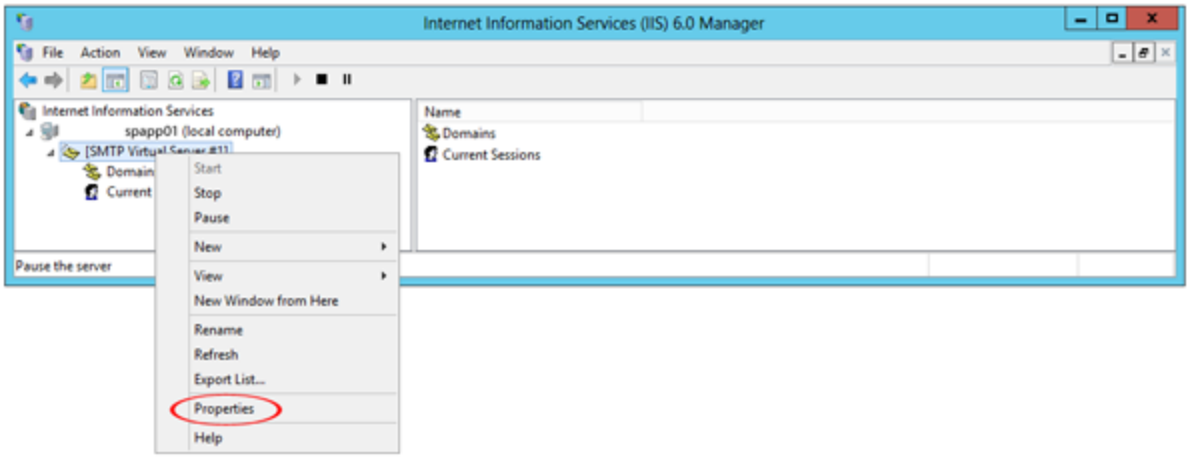
In this example, the alias is set to going.nowhere.local



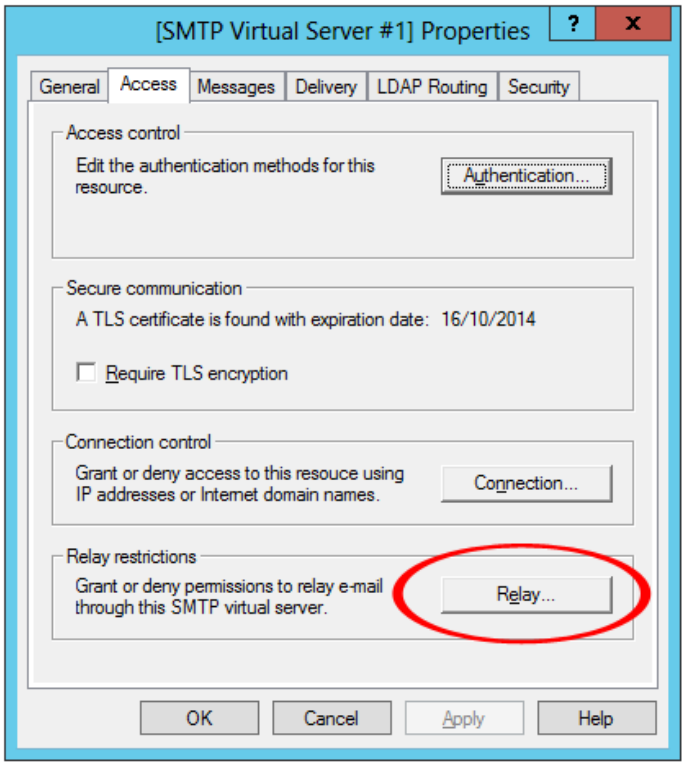
Configure IIS SMTP service to relay email to a Smart Host

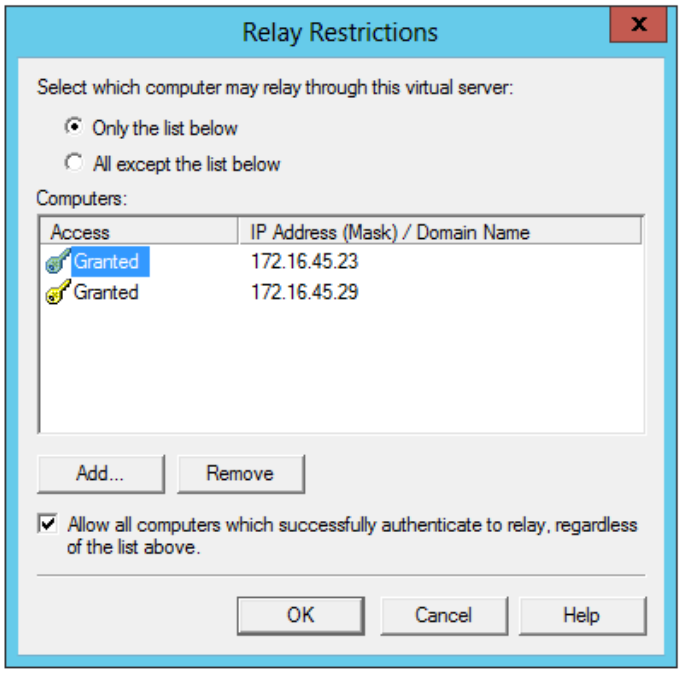
Configure the IIS SMTP service (on the SharePoint application server) to forward email (for remote domains) to itself.

1. Open the IIS Admin 6.0 console
2. Expand the local server
3. Expand the SMTP Virtual Server



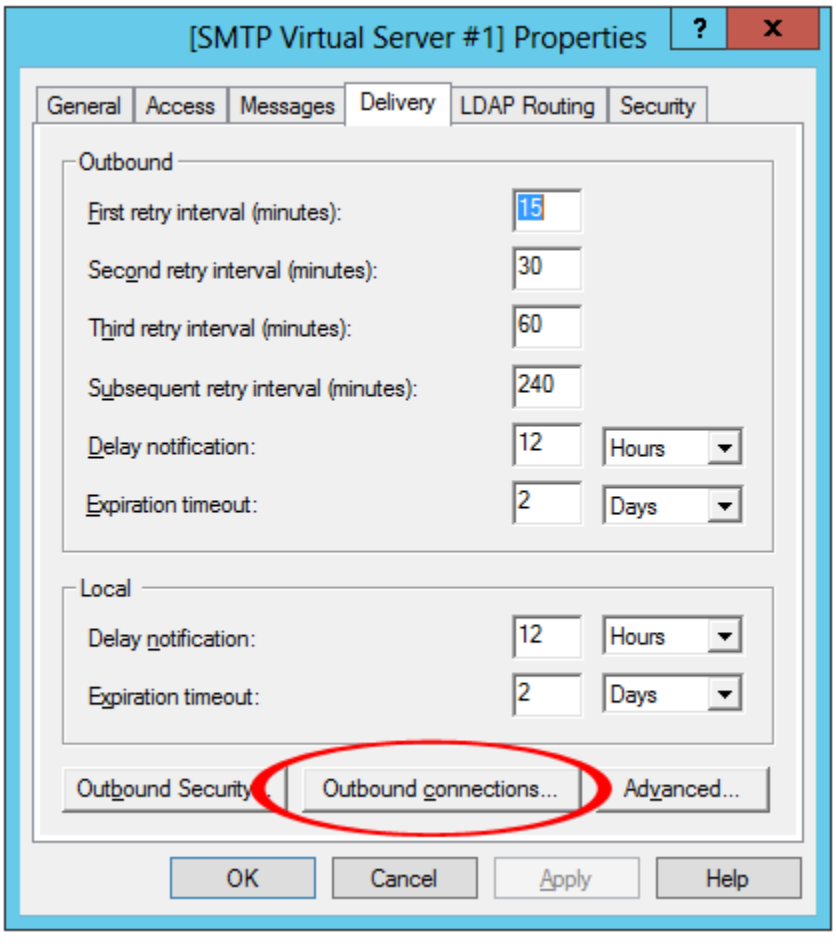
4. Right click the virtual server, and click Properties
5. Click on the Access tab
6. Configure the Relay settings
   Note: Be careful configuring relay settings. Ensure that you restrict the list of servers (IP Addresses) allowed to use this server as an open (unauthenticated) SMTP relay.



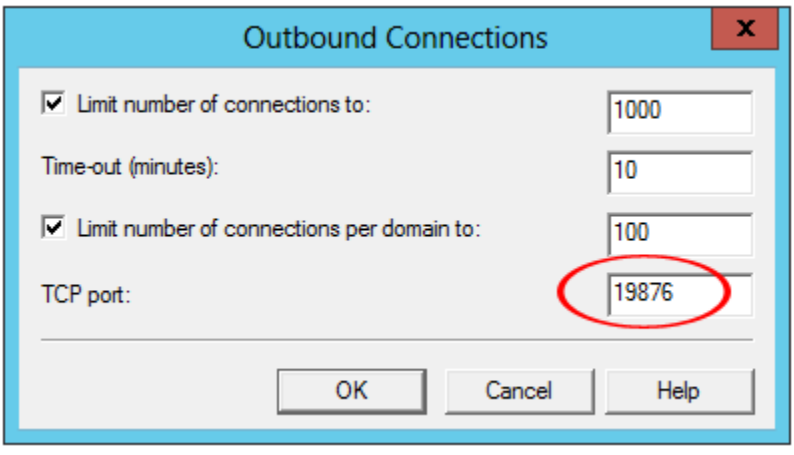


7. From the example we've been using, we are going to allow the two SharePoint WFE servers to relay through this SMTP server un-authenticated.
8. After configuring the SMTP Relay settings, you need to configure the Smart Host that will be used to forward all foreign email to. Click on the Delivery tab.

9. Click Outbound connections

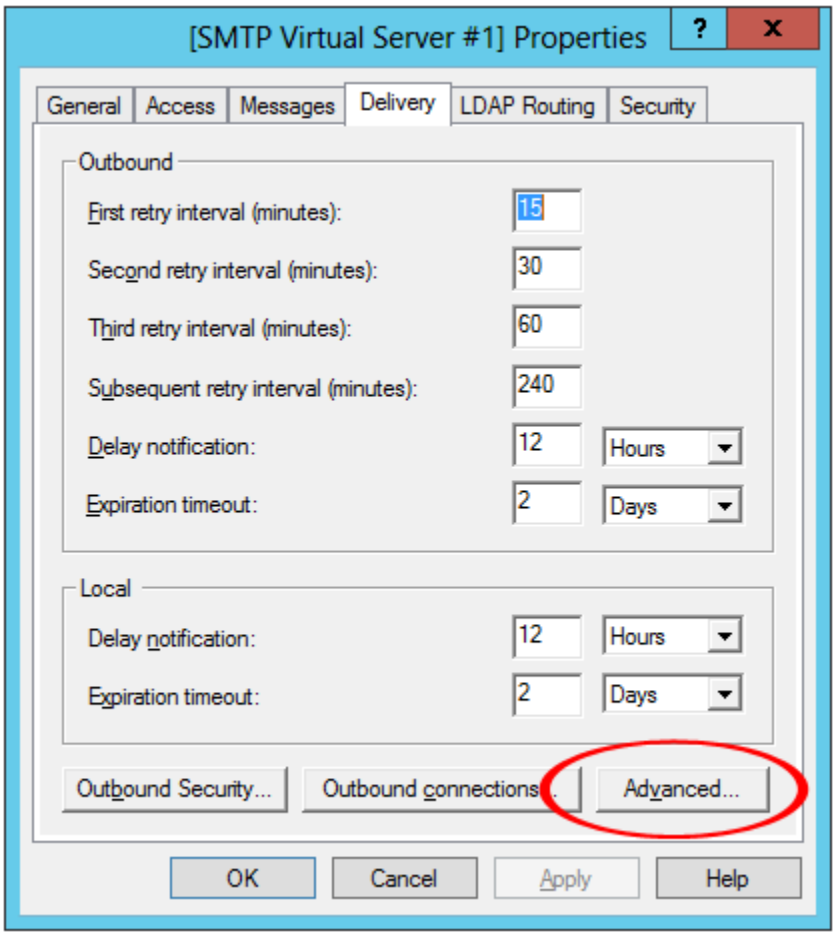


10. In the Outbound Connections dialog, enter the TCP port that SMTP4Dev is listening on. In the example, this is port 19876

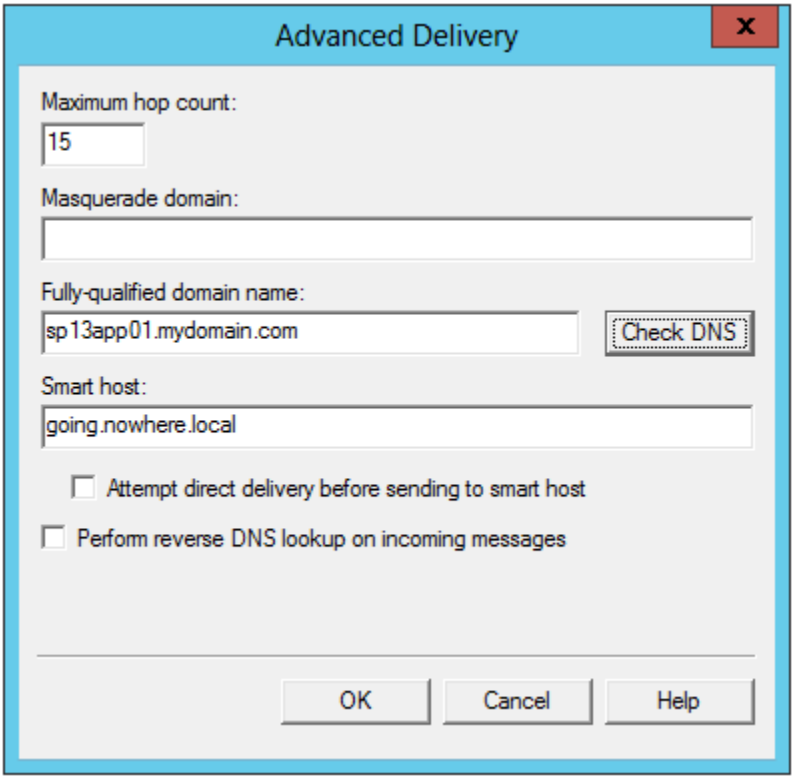


11. Click Ok to save the changes.

12. From the Delivery tab, click Advanced.



13. In the Advanced Delivery dialog, set the Smart Host. This will be the alias name you added to the hosts file. In the example, we used going.nowhere.local



14. Click Ok to save the changes.
15. This completes the configuration of the IIS STMP service.

Summary

1. The IIS SMTP Service on all SharePoint servers is configured to accept emails sent to SharePoint web applications. In the examples, the SharePoint servers accept email sent to the production-sharepoint.mydomain.local domain. E.g. an email addressed to myshareddocumentlibrary@production-sharepoint.mydomain.local
2. The IIS SMTP Service on the SharePoint application server is configured to forward all foreign email (that is, email sent to other domains) to an SMTP Smart Host (hosted on the same server), called going.nowhere.local, over port 19876.
3. The SMTP service listening on port 19876 is SMTP4Dev. SMTP4Dev will receive all email sent to it, so that it can be viewed. E-mail will never reach the mailbox of the intended recipient, which is the behaviour we want.

Configure the Outbound email settings for the farm

The final step is to configure the farm to send all outbound email to the SharePoint application server . After doing this, the IIS SMTP service on the SharePoint application server will receive all outbound email, and forward it on to the testing SMTP service, SMTP4Dev (listening on port 19876).

References

- Original content from Matthew Yarlett's blog, Testing SharePoint Email Alerts in UAT and DEV Environments
- This solution is based on using SMTP4Dev , a community project that can be found on Codeplex.
- Netstat

SharePoint 2013: Deploy and apply theme to SharePoint sites with PowerShell

Introduction

From the beginning Theme in SharePoint is an interesting feature for developer and UI designer. In WSS3.0 and MOSS 2007 all themes related files resides under theme folder inside 12 hives. So if you want to create new theme then you have to make new folder under theme and use it. So it was deployment directly on the file system and they could be available to other web applications even though they were not needed. And also after every change you need to deploy it and IIS reset was needed to take complete effect. So to remove those headache of deployment process Microsoft introduces Theme gallery in SharePoint 2010 and .thmx file. It actually was an answer to theme deployment(in 2007) with simple deployment approach in 2010(no need of IIS reset also); and also somehow it took away burden of creating theme from the SharePoint developer. But the problem with .thmx file was the resource needs to be PowerPoint expert to create those thmx files(I am not sure about other tools :)). At last in SharePoint 2013 it makes easy to make and apply theme; it makes clear two roles and their responsibilities: SharePoint UI Designer- he/she has to create the theme and SharePoint Developer/Admin – he/she has to deploy and apply the theme. Deployment process is also simple; you can have preview available to check how your site going to look. So life in SharePoint 2013 is little bit easy for developers :).

Case Statement

As we know all over the world organizations are now transforming their SharePoint platforms to the SharePoint 2013 platform to gain the power of new features. Now as a part of migration you have to migrate the theme which is currently applied to the all sites in your SharePoint 2010 to the new sites in SharePoint 2013 environment. In first look it looks very simple to migrate theme to 2013. But actually the truth is different; you are not migrating theme to 2013 rather you are creating it from the scratch. Now after creating the theme next challenge is how to deploy and apply them to the thousands of sites? Do not get panic, I will explain here how to migrate(actually

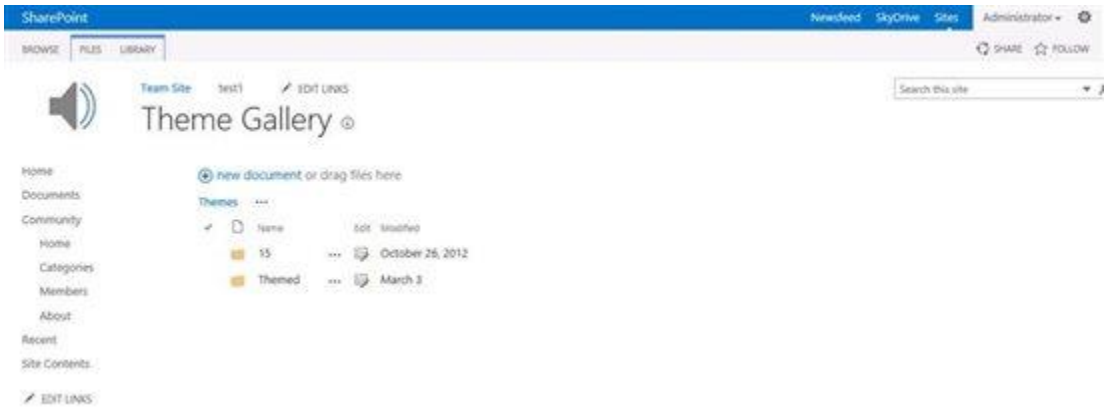
recreate J) theme to 2013 and apply them with one or two click.

About SharePoint 2013 Theme

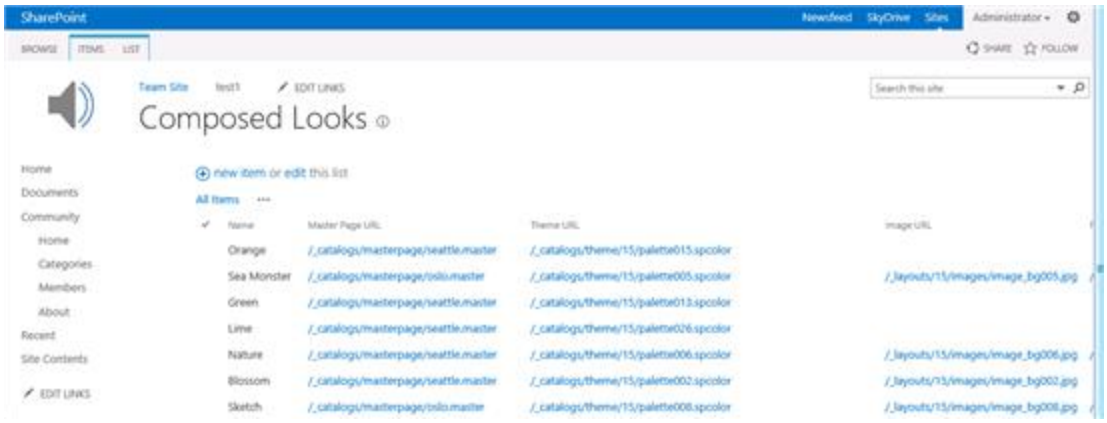
Let's learn SharePoint 2013 theme. As I explained in Introduction section that in 2007 we were using some files as artifacts of the theme, then in 2010 we are using .thmx file to create custom theme. So what inside SharePoint 2013?

So following galleries are important part of the theme:

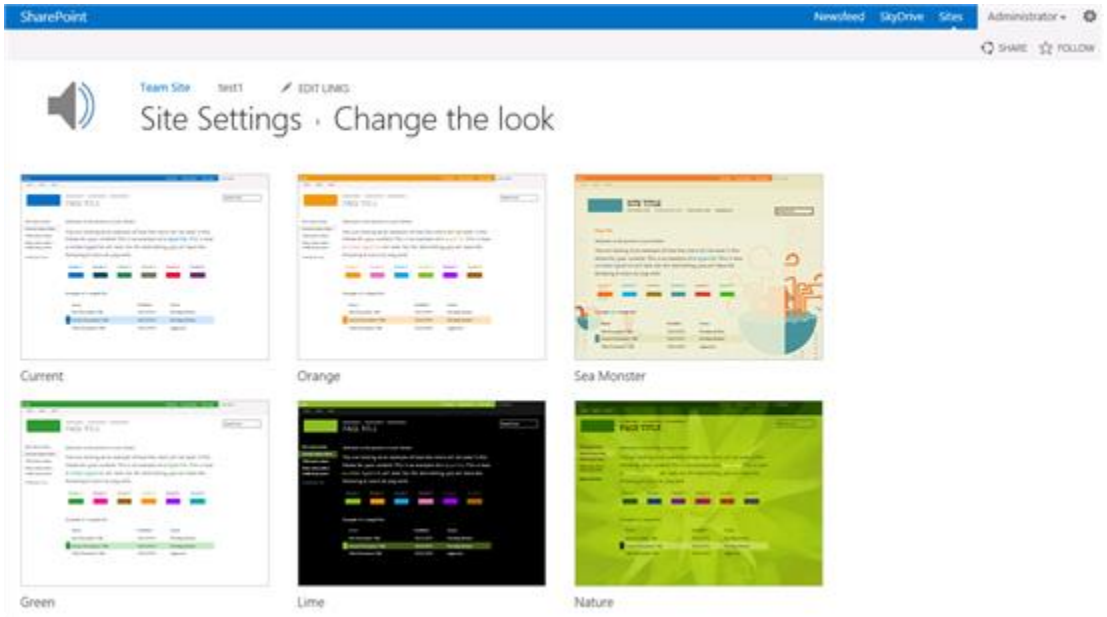
1. Theme gallery – you can access this from here: /_catalogs/theme/Forms/AllItems.aspx . This gallery generally resides on the root web level and has theme files – Color palette and font schema inside 15 folder.



2. Composed looks(Web Designer) gallery – As name suggest it has several composition of looks. This library actually contains master page URL, image URL, theme (color palette) URL, Font Scheme URL and Display Order. And all of them together is a theme. So when you are creating a theme you also have to make new entry in this gallery with respective information.



So once you have entry for your theme in this library you can see your composed look/theme available on applying page.



So this is all about the galleries which you must know while working on theme. Now let's see the files which actually makes the theme:

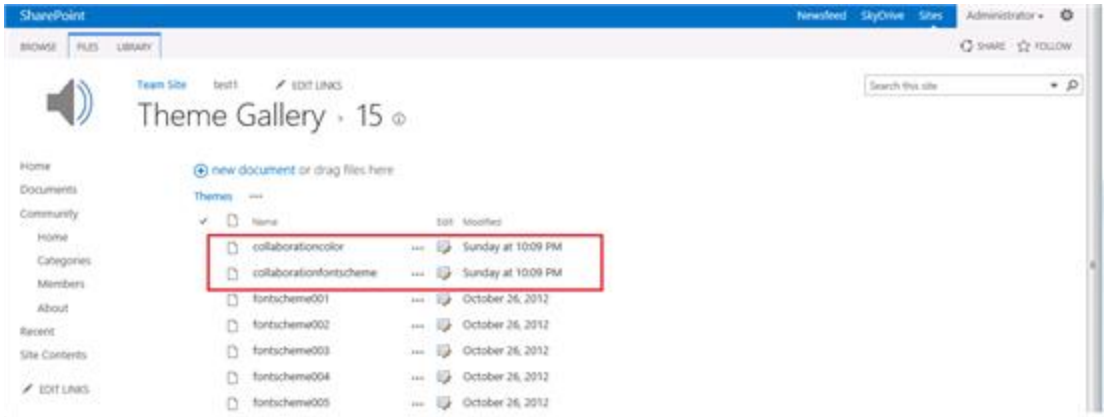
1. Color palette file: this file defines the combination of color which is going to be used in the site. You can define color scheme for your site and modify the look.
2. Font scheme file: this file defines the font scheme for your site. SharePoint has provided you slots and according to your requirement you have to apply font to them.
3. Background Image: this is image used as the back ground of pages of your site.
4. Master page: this is master page which defines the layout and structure of site. We will not focus more on this.

You can learn in detail about theme here: http://msdn.microsoft.com/en-us/library/office/jj927174.aspx

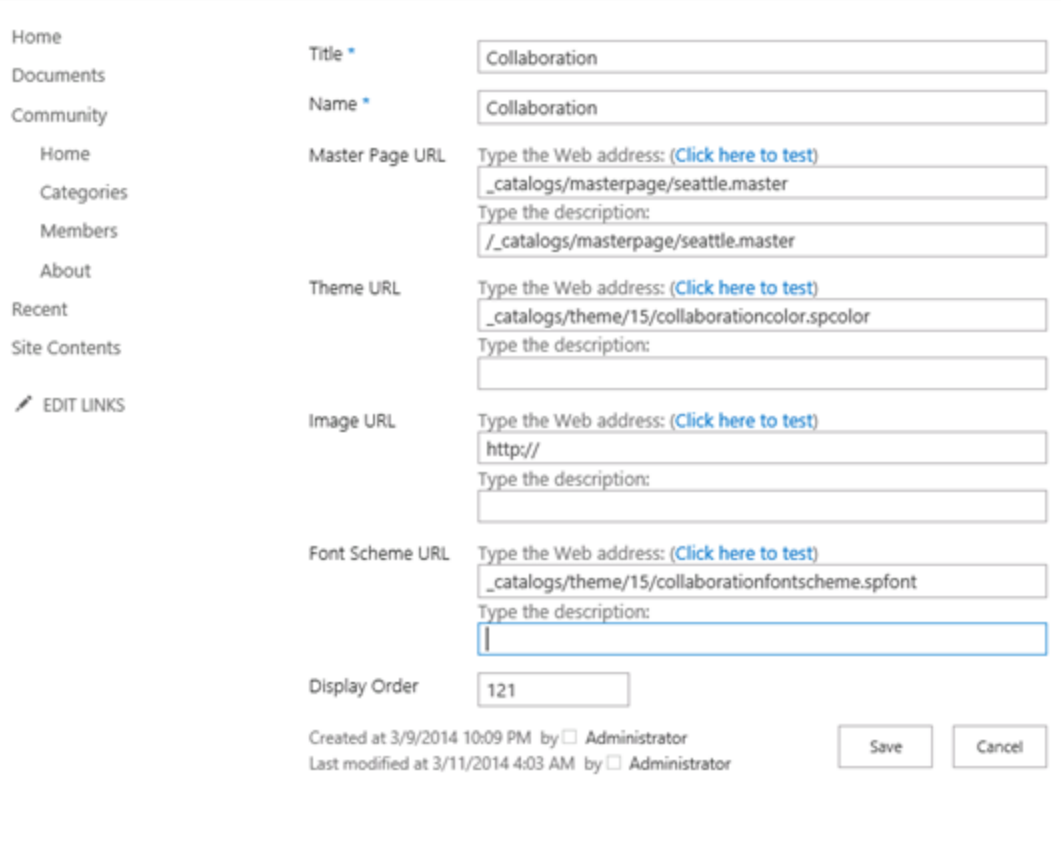
Now you have brief overview about theme. So if you compare SharePoint 2010 theme and SharePoint 2013 theme structure then you will realize that they are different and you have to create theme from the scratch for SharePoint 2013. It is not as-is migration. So whenever got this requirement do not hurry to mark it easy.

So once theme is ready and you want to deploy it then you have to follow some of manual steps:

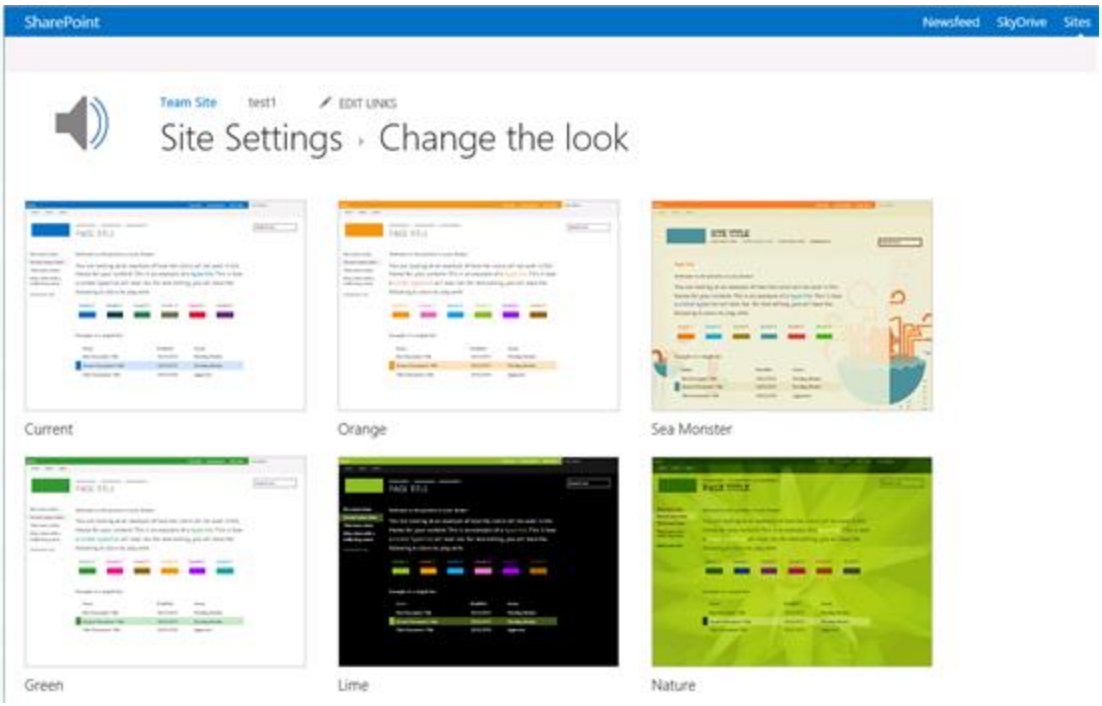
1. Go to the theme gallery and upload the color palette and font scheme file. Here I have created collaboration theme for my team site.



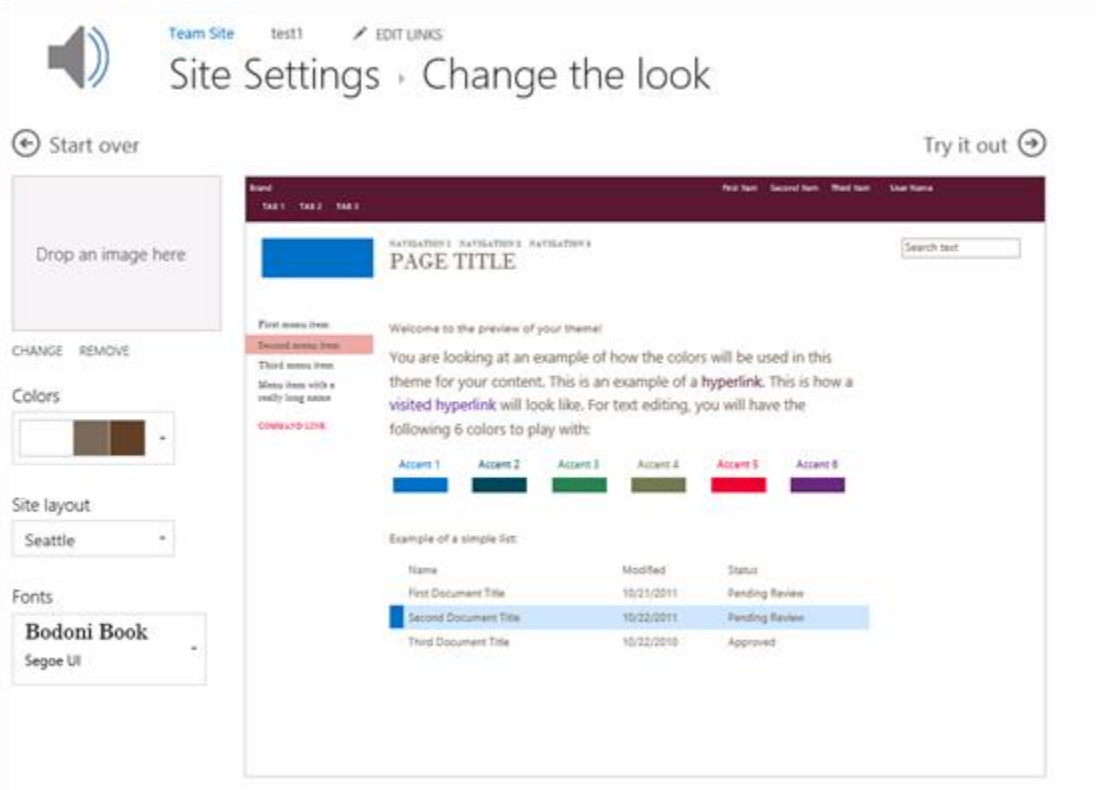
2. The next step is creating new composition in Composed looks(Web Designer) gallery . Give the correct URL of master page, theme page and font schema page.



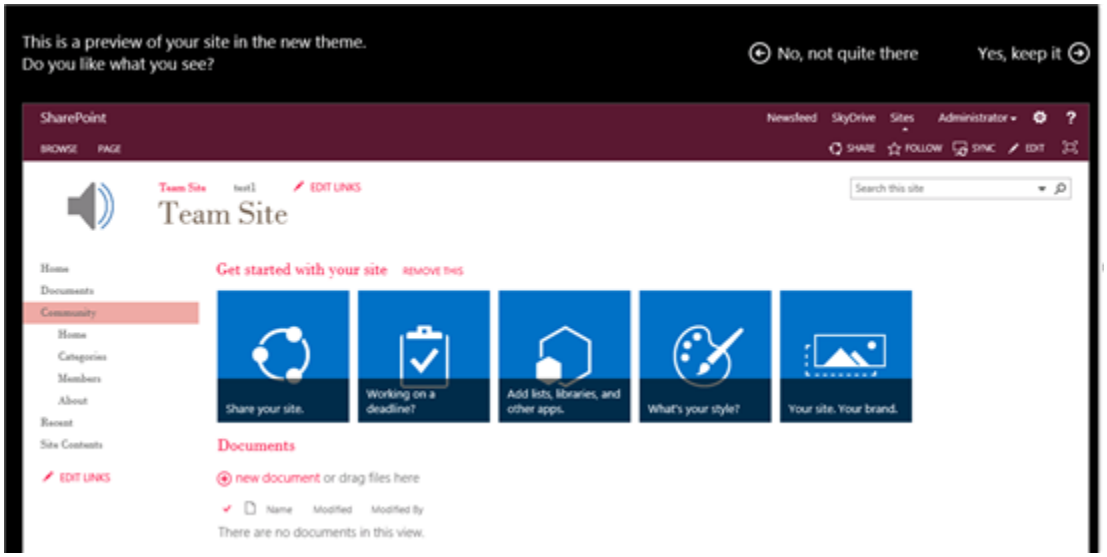
3. Now go to the Site Settings page and then click on the 'Change the look' link under Look and Feel section. You will see all composition here including new one(if there is no error).



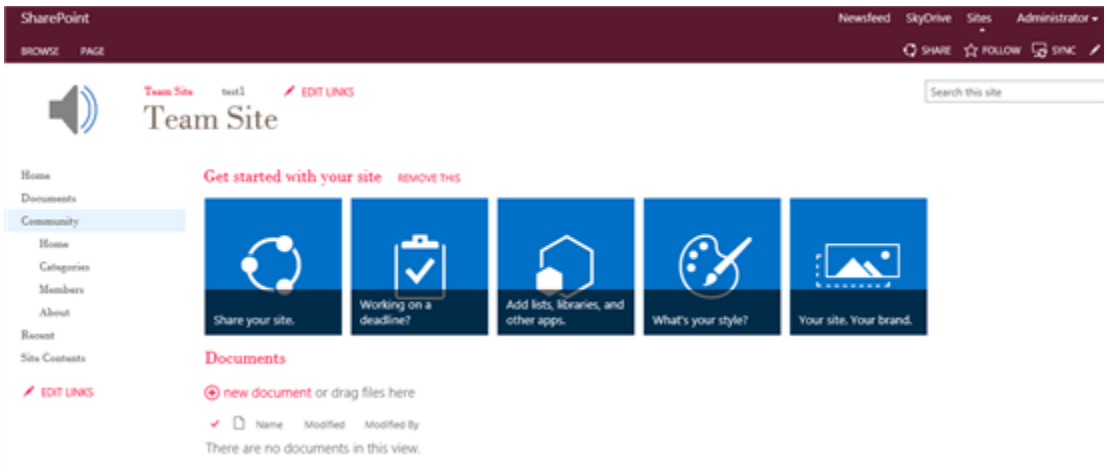
4. Now find out new theme(here in my case: Collaboration) and click on it. You will see first preview how site looks with theme.



5. Now click on the Try it Out and you will see the final preview of your site. Really very cool feature.



6. Now if you are happy with your theme then just click on 'Yes Keep it'. And theme gets applied to your site.



Hushh. So these are steps to apply theme. Now you have thousands of sites which are migrated and waiting for Theme to get applied. Now??? Got scared of repeating same steps on thousands of sites?? Don't worry PowerShell is there with its power to help you.

Complete PowerShell Script

This is complete PowerShell snippet which you can use to deploy and apply theme to your environment. I have not tested it for 'My Sites' but it should work practically.

```
Add-PSSnapin "Microsoft.SharePoint.PowerShell" -ErrorAction
SilentlyContinue;
[xml]$inputfile = Get-Content C:\Data\theme\ThemeScriptInput.xml

$ColorFilePartUrl = "/_catalogs/theme/15/";
$FontFilePartUrl = "/_catalogs/theme/15/";
$MasterPagePartUrl = "/_catalogs/masterpage/";

foreach( $site in $inputfile.SiteCollections.site)
{
    $siteurl = $site.url;
    $themeName = $site.theme;
    $colorfile = $site.colorfile;
    $fontfile = $site.fontfile;
    $masterpage = $site.masterpage;
    $colorshemeFilePath = $site.colorFilePath;
    $colorfontFilePath = $site.fontFilePath;
    $IsfilesToUpload = $site.filesToUpload;

    Write-Host "$siteurl + " :: " + $themeName + " :: " + $colorfile +
" :: " + $fontfile + " :: " + $masterpage";

    $site = Get-SPSite $siteurl;
    $rootweb = $site.RootWeb;

    if($IsfilesToUpload -eq "true")
    {
        #region Upload theme files to the root web gallery

        $colorSchemeBytes =
[System.IO.File]::ReadAllBytes($colorshemeFilePath);
        $fontSchemeBytes =
[System.IO.File]::ReadAllBytes($colorfontFilePath);

        $rootweb.allowunsafeupdates = $true;
        $themeList =
$rootweb.GetCatalog([Microsoft.SharePoint.SPListTemplateType]::ThemeCa
talog);
        $folder = $themeList.RootFolder.SubFolders["15"];
```

```
        $folder.Files.Add($colorfile,$colorSchemeBytes,$true);
        $folder.Files.Add($fontfile,$fontSchemeBytes, $true);
        $rootweb.allowunsafeupdates = $false;
        #endregion
    }


$colorfile=$site.AllWebs[0].GetFile("$siteurl$ColorFilePartUrl$colorfile");

$fontfile=$site.AllWebs[0].GetFile("$siteurl$FontFilePartUrl$fontfile");

    foreach ($Web in $site.AllWebs)
    {
       Write-Host "processing: " $Web.Title;
       $Web.allowunsafeupdates = $true;
       $relativeUrl = $Web.ServerRelativeUrl;
       $spList = $Web.GetCatalog([Microsoft.SharePoint.SPListTemplateType]::DesignCatalog);

       #region Add New theme to the Composed Looks gallery

       $SPQuery1 =  New-Object Microsoft.SharePoint.SPQuery;
       $SPQuery1.Query = "<Where><Eq><FieldRef Name='Name'/><Value Type='Text'>$themeName</Value></Eq></Where>";
       $SPQuery1.RowLimit = 1;
       $SPQuery1.ViewFields = "<FieldRef Name='Name'/>";
       $SPQuery1.ViewFieldsOnly = $true;

       $spListItems1 = $spList.GetItems($SPQuery1);


       if($spListItems1.Count -eq 0)
       {

       $newThemeItem = $spList.AddItem();

       $newThemeItem["Name"] = $themeName;
       $newThemeItem["Title"] = $themeName;
       $newThemeItem["MasterPageUrl"] = "$relativeUrl$MasterPagePartUrl$masterpage";#$Web.MasterUrl;
       $newThemeItem["ThemeUrl"] = "$ColorFilePartUrl$colorfile";
       $newThemeItem["FontSchemeUrl"] = "$FontFilePartUrl$fontfile";
       $newThemeItem["DisplayOrder"] = 121;
```

```
        $newThemeItem.Update();

        }
        #endregion

        #region Set the theme



$theme=[Microsoft.SharePoint.Utilities.SPTheme]::Open($themeName,
$colorfile);
         Write-Host $theme.Name "to" $Web.Title;
         $theme.ApplyTo($Web, $false);



        #endregion

        #region Set applied theme as current theme

        $SPQuery =  New-Object Microsoft.SharePoint.SPQuery;
        $SPQuery.Query = "<Where><Eq><FieldRef
Name='DisplayOrder'/><Value Type='Number'>0</Value></Eq></Where>";
        $SPQuery.RowLimit = 1;
        $SPQuery.ViewFields = "<FieldRef Name='DisplayOrder'/>";
        $SPQuery.ViewFieldsOnly = $true;

        $spListItems = $spList.GetItems($SPQuery);


        if($spListItems.Count -eq 1)
        {
             $spListItems[0].Delete();
        }

        $currentThemeItem = $spList.AddItem();

        $currentThemeItem["Name"] =
[Microsoft.SharePoint.SPResource]::GetString([System.Threading.Thread]
::CurrentThread.CurrentUICulture,
[Microsoft.SharePoint.Strings]::DesignGalleryCurrentItemName);
        $currentThemeItem["Title"] =
[Microsoft.SharePoint.SPResource]::GetString([System.Threading.Thread]
::CurrentThread.CurrentUICulture,[Microsoft.SharePoint.Strings]::Desig
nGalleryCurrentItemName);
```

```
        $currentThemeItem["MasterPageUrl"] =
"$relativeUrl$MasterPagePartUrl$masterpage";#$Web.MasterUrl;
        $currentThemeItem["ThemeUrl"] = "$ColorFilePartUrl$colorfile";
        $currentThemeItem["FontSchemeUrl"] =
"$FontFilePartUrl$fontfile";
        $currentThemeItem["DisplayOrder"] = 0;
        $currentThemeItem.Update();

        #endregion

        $Web.allowunsafeupdates = $false;
        Write-Host "Set" $theme.Name "theme to :" $Web.Title "("
$Web.Url ")" ;
    }

}
```

Input File

```
<SiteCollections>
  <site>
    <url>http://local/sites/MyBlog< ;/url>
    <theme>Collaboration</theme>
   <filesToUpload>false</filesToUpload>
    <colorFilePath>C:\theme\collaborationcolor.spcolor</colorFil
ePath>
    <fontFilePath>C:\theme\collaborationfontscheme.spfont</fontF
ilePath>
    <colorfile>collaborationcolor.spcolor</colorfile>
   <fontfile>collaborationfontscheme.spfont</fontfile>
    <masterpage>seattle.master</masterpage>
  </site>
  <site>
    <url>http://local< ;/url>
    <theme>Collaboration</theme>
```

```
    <filesToUpload>true</filesToUpload>
    <colorFilePath>C:\theme\collaborationcolor.spcolor</colorFilePath>
    <fontFilePath>C:\theme\collaborationfontscheme.spfont</fontFilePath>
    <colorfile>collaborationcolor.spcolor</colorfile>
    <fontfile>collaborationfontscheme.spfont</fontfile>
    <masterpage>seattle.master</masterpage>
  </site>
</SiteCollections>
```

PowerShell Script walk through

Now let's go through the code snippet and understand it.

```
Add-PSSnapin "Microsoft.SharePoint.PowerShell" -ErrorAction SilentlyContinue;
```

Above command adds the SharePoint snap in to the PowerShell console.

```
[xml]$inputfile = Get-Content C:\Data\theme\ThemeScriptInput.xml
```

Above command will load the input xml file in context.

```
foreach( $site in $inputfile.SiteCollections.site)
```

Above foreach loop will traverse through all site collections provided in xml file.

Now once we have all input ready then first task is: upload the color palette and font scheme files to the theme gallery of site collection. We need to upload those two files to

root web and then you can reuse them in all sub sites. Region “Upload theme files to the root web gallery ” has complete snippet which reads file from provided file location and uploads to theme library.

Now once file gets uploaded get their references which are required while applying theme. Following lines gives you files from the theme gallery:

```
$colorfile=$site.AllWebs[0].GetFile("$siteurl$ColorFilePartUrl$colorfile");
$fontfile=$site.AllWebs[0].GetFile("$siteurl$FontFilePartUrl$fontfile");
```

Next foreach loop traverse through the all sub sites of the site collection and applies themes. First of all it gets the web design gallery/list with following command:

```
$spList = $Web.GetCatalog([Microsoft.SharePoint.SPListTemplateType]::DesignCatalog);
```

Now once design gallery is available script will add new composition/theme in the gallery/list. It first checks whether same composition is present or not, if it is present then it skips the creation. Region “Add New theme to the Composed Looks gallery ” adds new composition/theme.

Once composition/theme gets added successfully next step is to apply it. And region “Set the theme ” has code snippet to apply theme. Following are important command lines:

```
$theme=[Microsoft.SharePoint.Utilities.SPTheme]::Open($themeName, $colorfile);
Write-Host $theme.Name "to" $Web.Title;
$theme.ApplyTo($Web, $false);
```

‘ApplyTo’ method is important here which actually applies theme to the site. Before that you have to get the reference of theme as shown in above snippet.

The last step is once you apply theme successfully you have to make it current theme in design gallery. If you check the manual steps does same then we also have to do it in our script. And our region "region Set applied theme as current theme " does the same. It first deletes the old current theme and add new entry with our theme parameter as current theme.

Input File Walk through

The description of each node is as follow:

SiteCollections – this is parent node which can have multiple 'site' nodes. It means you can put 'n' numbers sites here.
Site – this node contains all required parameter for that specific site.
url – this is URL of site collection. Do not add '/' at the end.
Theme – this is name of theme by which your theme will be known in SharePoint.
filesToUpload – this is flag which decides whether file upload is needed or not. true- files needs to upload, false- no need to upload file.
colorFilePath – this is the file location of '.spcolor' file which needs to upload to theme gallery.
fontFilePath – this is the file location of '.spfont' file which needs to upload to theme gallery.
Colorfile – it contains name of the color or .spcolor file.
fontfile – it contains name of the font or .spfont file.
masterpage – it contains name of the master page file.

Generate input xml file carefully and enjoy the painless deployment.

So this is all about the SharePoint theme and deployment script.

Interact with Office 365 using PowerShell and the Client Side Object Model

One of my biggest frustration with Office 365 over the past few months, is the fact that we are now prohibited from creating new subsites under the main Internet Public facing site collection that is given to us by SharePoint online. Using SharePoint Online Management Shell only gives us 30 PowerShell cmdlets to use, and none of them allow us to create new SPWeb objects. The only option left to create a subsite on an Office 365 public site is to write custom code, but custom code also means that this can be achived using PowerShell. Remember that PowerShell is able to leverage all of the .NET framework components. We can then use the .NET Managed Client-Side Object Model (CSOM) of SharePoint and make remote calls to the Office 365 SharePoint Online object model. The following PowerShell script will prompt you to enter the URL of your SharePoint Online public site collection (using the https:// prefix), the credentials for your account, as well as the name and relative URL for this new Web you are trying to create (see Figure 1 for a real-life example).

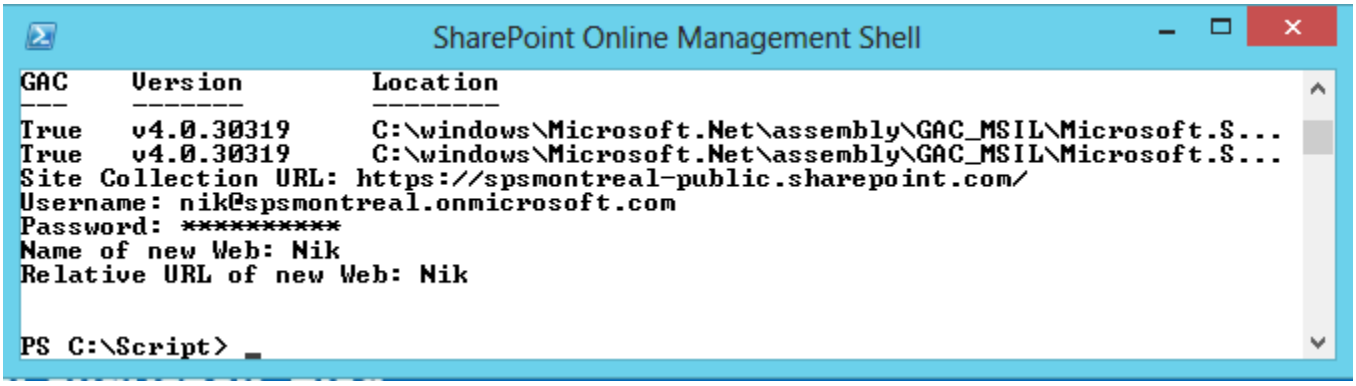


Figure 1 - Creating a new Office 365 SharePoint online Web using a custom PowerShell script

Full PowerShell Script:

```
[System.Reflection.Assembly]::LoadWithPartialName("Microsoft.SharePoin
t.Client")
[System.Reflection.Assembly]::LoadWithPartialName("Microsoft.SharePoin
t.Client.Runtime")

$url = Read-Host -Prompt "Site Collection URL"
$username = Read-Host -Prompt "Username"
```

```powershell
$password = Read-Host -Prompt "Password" -AsSecureString
$webTitle = Read-Host -Prompt "Name of new Web"
$webUrl = Read-Host -Prompt "Relative URL of new Web"

$ctx = New-Object Microsoft.SharePoint.Client.ClientContext($url)
$ctx.Credentials = New-Object
Microsoft.SharePoint.Client.SharePointOnlineCredentials($username,
$password)
$webInfo = New-Object
Microsoft.SharePoint.Client.WebCreationInformation
$webInfo.Title = $webTitle
$webInfo.Url = $webUrl
$webInfo.WebTemplate = "STS#0"
$webInfo.Language = 1033

$newWeb = $ctx.Web.Webs.Add($webInfo)
$ctx.Web.Update()
$ctx.Load($newWeb)
$ctx.ExecuteQuery()
```

SharePoint 2013 – Workflow Management – Starting a workflow using Powershell

Introduction

As it is right now, the SharePoint 2013 workflow management isn't exactly straight up using Powershell. The below code should aid in the understanding on how to work with it in its current state.

The solution

```
$sourceWebURL = '<URL>'
$sourceListName = '<List Name>'
$TargetWorkflow = '<Workflow Name>'
$spSourceWeb = Get-SPWeb $sourceWebURL
$spSourceList = $spSourceWeb.Lists[$sourceListName]

#-- Getting a Workflow manager object to work with.
$wfm = New-object
Microsoft.SharePoint.WorkflowServices.WorkflowServicesManager($s
pSourceweb)
#-- Getting the subscriptions
$sub = $wfm.GetWorkflowSubscriptionService()
#-- Getting the specific workflow within the list of
subscriptions on the specific list. (SP2010 associated workflows
basically)
$WF = $sub.EnumerateSubscriptionsByList($spSourcelist.ID) |
Where-Object {$_.Name -eq "$TargetWorkflow"}
#-- Getting a Workflow instance in order to perform my commands.
$wfis=$wfm.GetWorkflowInstanceService()

Foreach($item in $spSourceList){
    #-- Creating the dictionary object I need to parse into
StartWorkflow. This could be most other workflow commands.
    $object = New-Object
'system.collections.generic.dictionary[string,object]'
    $object.Add("WorkflowStart", "StartWorkflow");
    $wfis.StartWorkflowOnListItem($WF, $item.ID, $object)
```

References

- Microsoft.SharePoint.Client.WorkflowServices namespace
- http://msdn.microsoft.com/en-us/library/microsoft.sharepoint.client.workflowservices%28v=office.15%29.aspx

Additional credit

Frej Laursen, Joachim Bach & Per Jakobsen.

SharePoint: PowerShell Runas

Introduction

Have you ever had a SharePoint PowerShell script where you need to run a section of the script with elevated privileges (for example, under the security context of the farm service account), without being prompted to enter credential information?

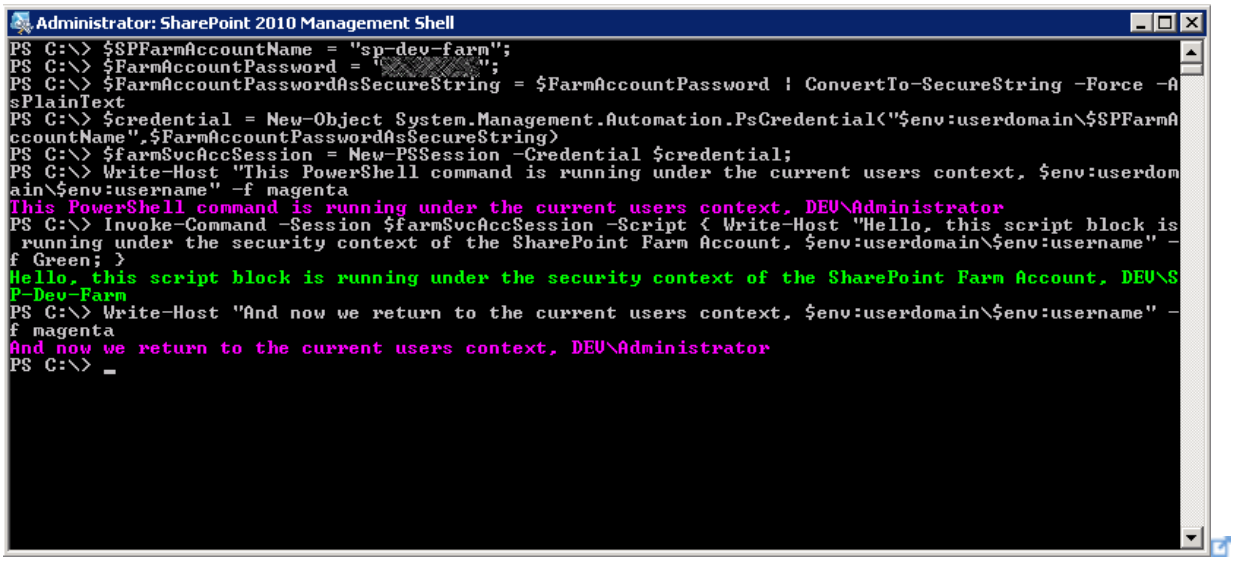
You can achieve this by running the code in a script block using the Invoke-Command cmdlet. The Invoke-Command cmdlet can take a PSSession object as a parameter. Using a PSSession object, you can create a PowerShell session with a different security context (as long as you have the username and password of the different user account) by passing the alternate credentials as a PSCredential object. It's all very easy, and here are a few examples.

Example

Basic PowerShell code to run a script block using alternate credentials:

```
#Variables to store the username and password for the alternate
account
$SPFarmAccountName = "sp-dev-farm";
$FarmAccountPassword = "NotMyPswd";

#Convert the plain text password to a SecureString, required to create
the PSCredential object
$FarmAccountPasswordAsSecureString = $FarmAccountPassword | ConvertTo-
SecureString -Force -AsPlainText

#Create the PSCredential object using the alternate users username and
password
#Note that the Domain name has been prepended to the username, using
the $env:userdomain variable, which represents the current domain.
$credential = New-Object
System.Management.Automation.PsCredential("$env:userdomain\$SPFarmAcco
untName",$FarmAccountPasswordAsSecureString)
```

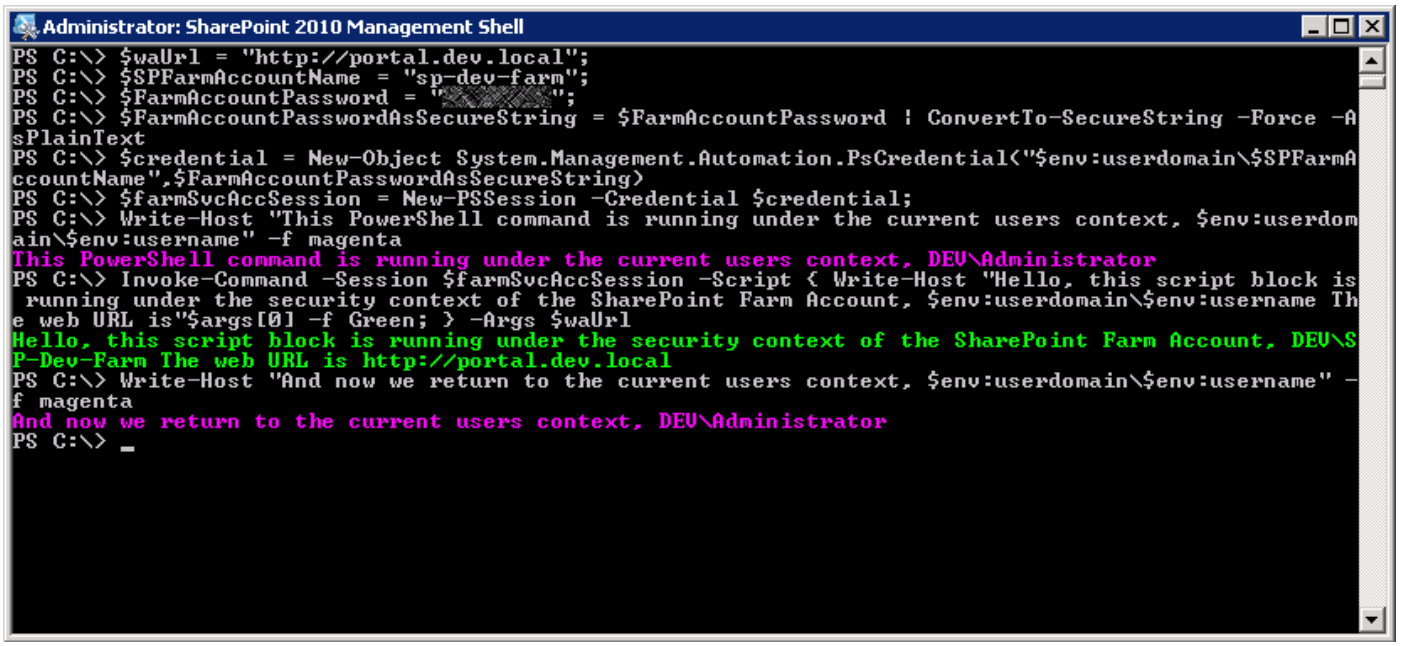
```powershell
#Create a new PowerShell session in the security context of the
alternate user, using the PSCredential object we just created
$farmSvcAccSession = New-PSSession -Credential $credential;

#Write some text to the PowerShell Window, that prints the username
from the current context
Write-Host "This PowerShell command is running under the current users
context, $env:userdomain\$env:username" -f magenta

#Pass the PSSession object to Invoke-Command, and write some text to
the PowerShell Window, that prints the username from the current
context of the PSSession object (which will be the security context of
the alternate user)
Invoke-Command -Session $farmSvcAccSession -Script { Write-Host
"Hello, this script block is running under the security context of the
SharePoint Farm Account, $env:userdomain\$env:username" -f Green; }

#Write some more text to the PowerShell Window, that shoes the
security context has returned to the original user
Write-Host "And now we return to the current users context,
$env:userdomain\$env:username" -f magenta
```



Run a PowerShell script block using alternate credentials, passing a parameter to the elevated session (in this case, a URL)

```powershell
$waUrl = "http://portal.dev.local";
```

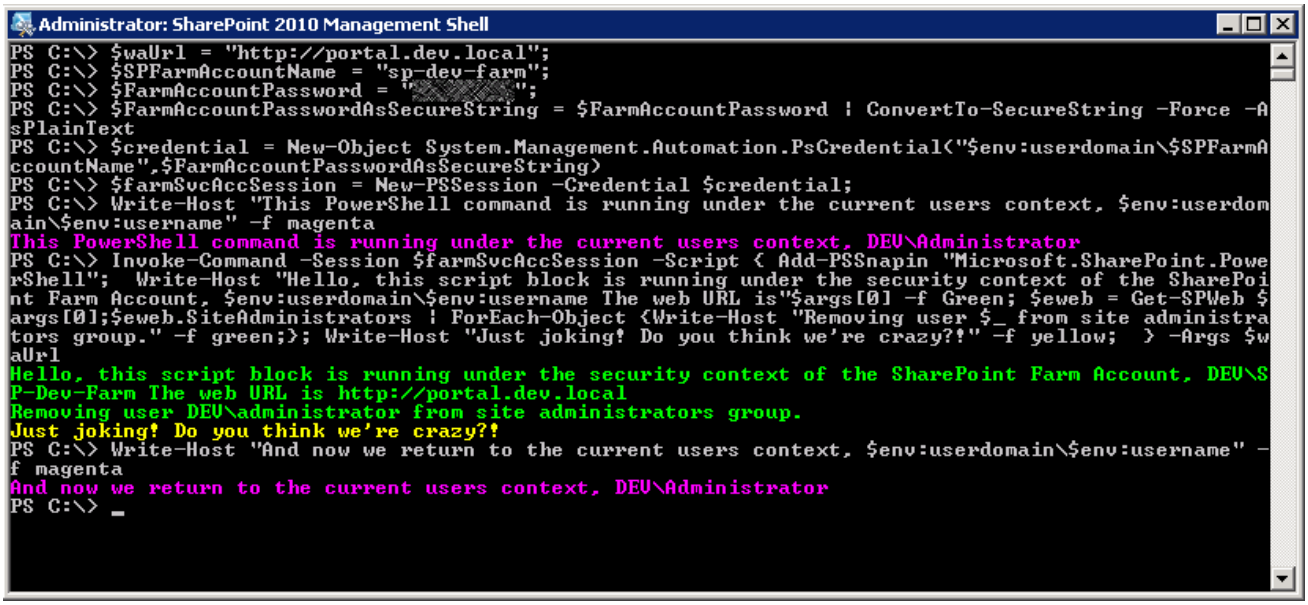
```
$SPFarmAccountName = "sp-dev-farm";
$FarmAccountPassword = "NotMyPswd";
$FarmAccountPasswordAsSecureString = $FarmAccountPassword | ConvertTo-
SecureString -Force -AsPlainText
$credential = New-Object
System.Management.Automation.PsCredential("$env:userdomain\$SPFarmAcco
untName",$FarmAccountPasswordAsSecureString)
$farmSvcAccSession = New-PSSession -Credential $credential;
Write-Host "This PowerShell command is running under the current users
context, $env:userdomain\$env:username" -f magenta
Invoke-Command -Session $farmSvcAccSession -Script { Write-Host
"Hello, this script block is running under the security context of the
SharePoint Farm Account, $env:userdomain\$env:username The web URL
is"$args[0] -f Green; } -Args $waUrl
Write-Host "And now we return to the current users context,
$env:userdomain\$env:username" -f magenta
```



Run a PowerShell script block using alternate credentials, passing a URL as a parameter to the elevated session, and then running some SharePoint commands (notice that the SharePoint PowerShell snapin needs to be loaded).

```
$waUrl = "http://portal.dev.local";
$SPFarmAccountName = "sp-dev-farm";
$FarmAccountPassword = "NotMyPswd";
```

```
$FarmAccountPasswordAsSecureString = $FarmAccountPassword | ConvertTo-SecureString -Force -AsPlainText
$credential = New-Object System.Management.Automation.PsCredential("$env:userdomain\$SPFarmAccountName",$FarmAccountPasswordAsSecureString)
$farmSvcAccSession = New-PSSession -Credential $credential;
Write-Host "This PowerShell command is running under the current users context, $env:userdomain\$env:username" -f magenta
Invoke-Command -Session $farmSvcAccSession -Script { Add-PSSnapin "Microsoft.SharePoint.PowerShell";  Write-Host "Hello, this script block is running under the security context of the SharePoint Farm Account, $env:userdomain\$env:username The web URL is"$args[0] -f Green; $eweb = Get-SPWeb $args[0];$eweb.SiteAdministrators | ForEach-Object {Write-Host "Removing user $_ from site administrators group." -f green;}; Write-Host "Just joking! Do you think we're crazy?!" -f yellow;  } -Args $waUrl
Write-Host "And now we return to the current users context, $env:userdomain\$env:username" -f magenta
```



See Also

- Invoke-Command
- New-PSSession
- How to create a PSCredential Object

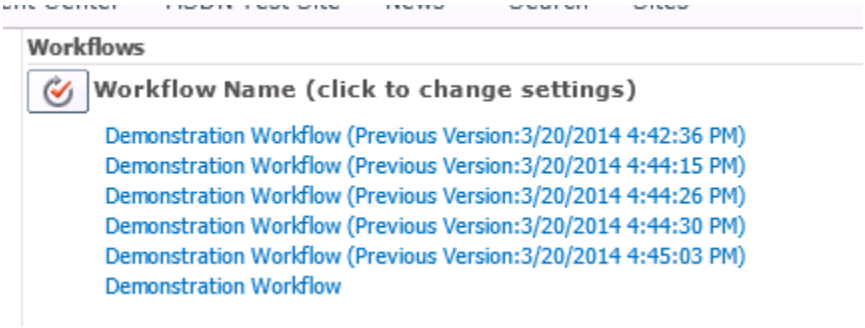
References

- Original content from Matthew Yarlett's blog, PowerShell Runas

Removing Old SharePoint Designer Workflow Instances

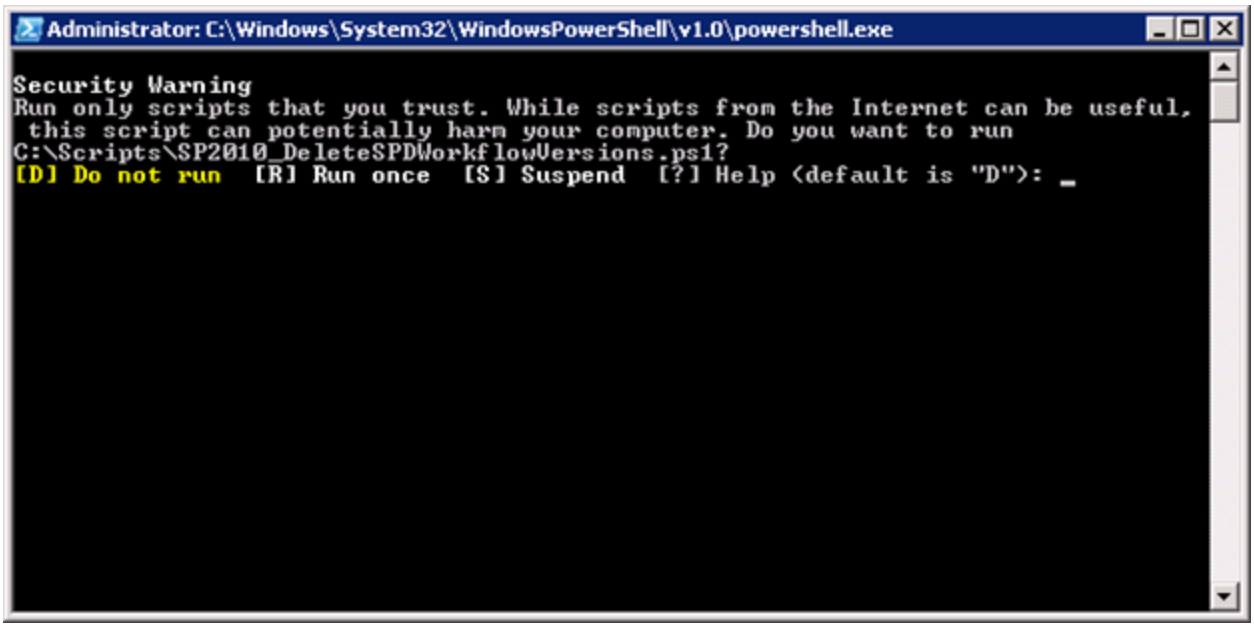
As most of you will know, there are two basic types of workflow in SharePoint. Some workflows are authored in SharePoint Designer whilst others are created within Visual Studio. This article is aimed at those who author workflows in SPD.

Each time a new workflow is published within SPD, a new instance is added. What this means is that each time a workflow is published, it overwrites a previous version but the previous versions are not removed. If an administrator were to check the active workflows, they'd see a history that looks like this. This is to allow any items that still have "in-progress" workflows to complete successfully.

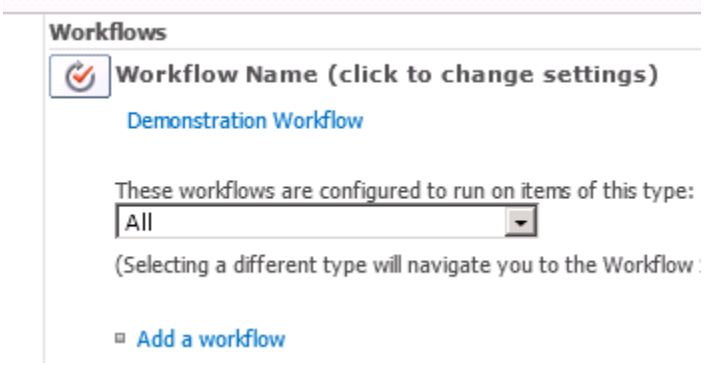


I don't believe that there is any performance overhead for having these historical workflows listed but wouldn't it be nice to have a way to remove these. This article is going to show you how.

1. Download this PowerShell Script from the TechNet Gallery. Download it somewhere that is easy to find, such as C:/Scripts
2. Right Click on the script and select Open
3. Look for the following variables and change them as required
    a. $web: Include the URL of your Site
    b. $listToCancel: Use the display name of the list / library that has the workflow versions you want to delete
    c. $wfToKeep: Use the name of the Workflow to be pruned
4. Save and close the script
5. Right click and select Run with PowerShell. Click Yes to the script. You'll see this as the script has been downloaded from TechNet rather than written locally



1. Once the script has run, return to your workflow history page. You should see only one version of the workflow as below



Caveats

Please be mindful of any workflow instances that may still be in operation. The script won't be too choosy in what it removes.

Credits

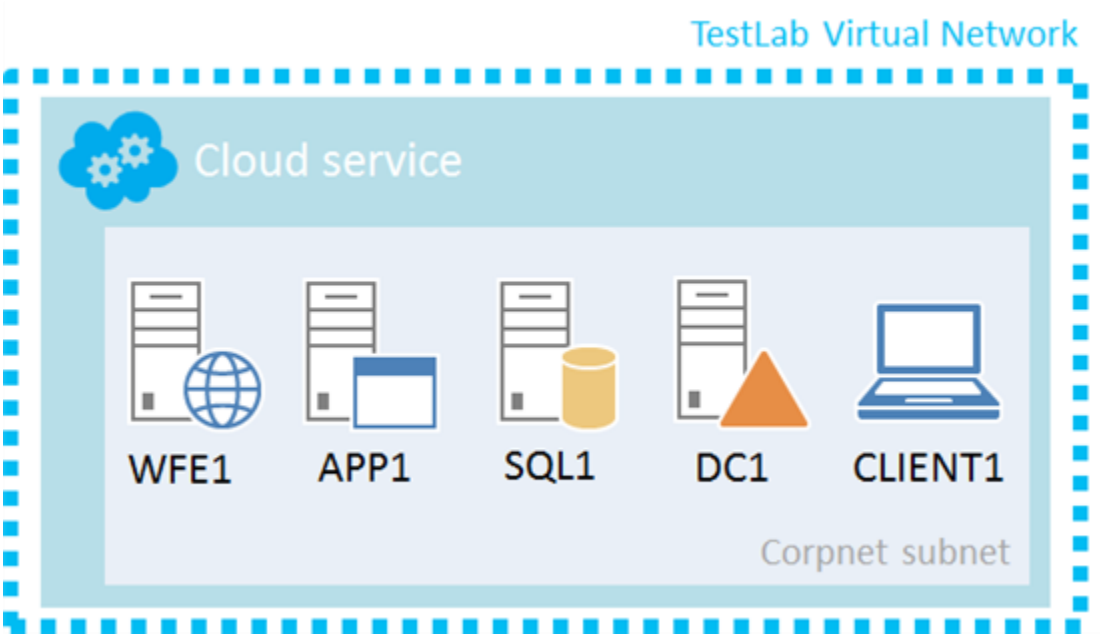
Thanks to Matthew Yarlett for looking over this script for me.

SharePoint Server 2013 Test Lab in Azure

With the Test Lab Guide: SharePoint Server 2013 Three-Tier Farm in Azure , you can use a free trial subscription , your MSDN subscription , or your paid Azure subscription to build out a SharePoint Server 2013 farm in a cloud-only virtual network in Azure.

The advantage of creating your SharePoint 2013 test lab in Azure is that you don't have to acquire the physical or virtual machines and create an isolated subnet. You can create the virtual machines for the test lab in Azure within minutes. Additionally, you can create the virtual machines for the lab in various sizes, and you have the ability to change their size after they are created.

Here is the resulting configuration:



Here is a short video that shows how to build this out:

Once you have configured this base configuration for SharePoint Server 2013, you can use the following TLGs to perform additional testing and learning (subject to the modifications described):

- Test Lab Guide: Configure Intranet and Team Sites with SharePoint Server 2013 To use this TLG, start with"Step 2: Configure the intranet and team sites on APP1". In addition, run **ipconfig** in a Windows PowerShell prompt on the APP1 computer to obtain its IP address, then substitute that address for "10.0.0.3" in the procedure named "To enable DNS name resolution for the new web applications".

- Test Lab Guide: Demonstrate permissions with SharePoint Server 2013
- Test Lab Guide: Demonstrate Profile Synchronization for SharePoint Server 2013 To use this TLG, start with "Step 2: Create a managed metadata service application, My Site web application, and site collection and configure settings".

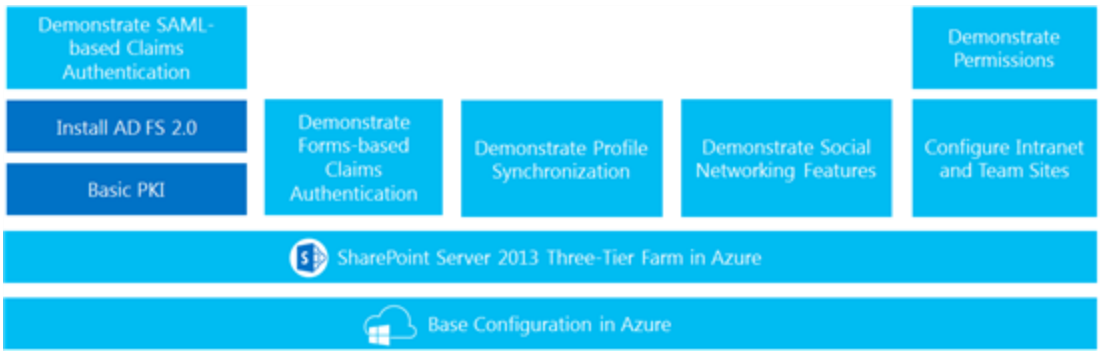
- Test Lab Guide: Demonstrate Social Networking Features of SharePoint Server 2013 To use this TLG, start with "Step 2: Create a My Site site collection and configure settings".

- Test Lab Guide: Demonstrate Forms-based Authentication with SharePoint Server 2013 To use this TLG, start with "Step 2: Configure forms-based authentication".

- Test Lab Guide: Demonstrate SAML-based Claims Authentication with SharePoint Server 2013 To use this TLG, use the Test Lab Guide Mini-Module: Basic PKI for Windows Server 2012 to install an enterprise root certification authority (CA) on APP1 and configure the corp.contoso.com domain for computer certificate autoenrollment. Run **gpupdate** at the Windows PowerShell prompt of each computer in the test lab. Then, start with "Step 2: Install AD FS 2.0 on DC1".

Also keep in mind that these TLGs were written for the Windows Server 2008 R2 base configuration, so you might have to translate some steps of some procedures to the new user interface in Windows Server 2012.

Here is the resulting SharePoint 2013 on Azure TLG stack (click on it for a larger version):



For additional Azure TLGs, see the Azure Test Lab portal page.

See the following for more information:

- SharePoint Server 2013 Business Intelligence Test Lab
- SharePoint 2013 Portal
- Test Lab Guides portal page
- Test Lab Guides blog
- Test Lab Guides FAQ

SharePoint 2013: Unattended Installation Gotchas

An unattended installation is an automated installation of any kind of Microsoft product that you can use to install or upgrade. It can be an operating system or a software application with minimal or NO user intervention. Unattended installations are typically used during large-scale rollouts when it would be too slow and costly to have administrators or technicians interactively install the operating system on individual computers.

As we are now migrating all our SharePoint 2010 Servers to SharePoint 2013 and installing every server separately could be very dangerous like have different installations (settings that you forget to adapt, 3th party tools that you forget to install, a firewall exception etc…) and will cost many dollars to the company. A full automated unattended script will never forget these and will always do the same on all servers. For more information about the unattended installation please refer to:
http://technet.microsoft.com/en-us/library/cc785644(v=ws.10).aspx

Unattended installations for SharePoint Server?

Yes, it's absolutely possible. A good friend of mine from the @Wikininjas (Wiki Ninjas is a group of authors who celebrate and evangelize the community catalyst and social-authoring excellence that is called TechNet Wiki.) wrote an excellent script that you can download on the Gallery of Microsoft. The PowerShell scripts provided by Craig will automate the download and installation of the SharePoint 2013 Prerequisites on Windows Server 2012 etc… The scripts will assist those who need to install SharePoint 2013 'offline' or wish to manually install its Prerequisites on Windows Server 2012.
Not only Craig from the @Wikininjas provided such scripts but ALL the SharePoint Community created several scripts in different ways to download and install in a few clicks SharePoint 2013 and configure it.
The real question here is "are they all FULL unattended?" Can I with only "1 click" download, install and configure my SharePoint Farm?

Full Unattended installations for SharePoint Server?

Several scripts on the internet are not full unattended; they just are “scripts”. The scripts are working great but if you want to create your own script due to the company’s own corporate best practices like “1 click only” you can have several issues.

Questions that you can ask are; Are these scripts able to bypass any TMG, PROXY or a FIREWALL rule to download the prerequisites? What if the UAC isn’t deactivated on the server?

Now you can say that as a SharePoint Administrator or Developer you can’t think to everything but you’re task is to create a NO user intervention needed script. And there are many ways to achieve that goal.

Here are a few gotchas about a full unattended scripting for a SharePoint download, install and configuration.

-NoNewWindow

When you have to install several executables or MSI’s please be sure to use the –NoNewWindow in the Start-Process commend line with PowerShell. With this parameter you’ll ensure that no any new window will appear during the installation. This option will start the new process in the current console window. By default Windows PowerShell opens a new window.

-Language Packs needs configuration file

Language Packs are very important in SharePoint. Especially in Belgium. As there are 3 official languages; every person wants to see his SharePoint site in his language (French, Dutch or German). When you try to install a specific language pack on SharePoint 2010; a simple command line was enough as they were simple executables.

Start-Process -filepath “$SourcePathLANG\ServerLanguagePack_NL.exe” -Argument “/quiet” -PassThru –Wait
With the /quit argument you’re actually specifying that the installation should be “invisible” for the user. The QuietArg attribute is also appended if the Setup is running. In such cases, you can specify the QuietArg attribute with the “/quiet” switch. For example, you can use: QuietArg=”/quiet”.

When you try to install SharePoint 2013 language pack as on his old way, you get an error message saying that the **/quiet** argument isn't supported anymore. You have to specify one of the following arguments to install the language pack.

- /Config
- /Modify
- /Repair
- /Unistall

As you can see for installing the Language Pack you need a **Configuration File**.
According to TechNet you can create your own configuration file: http://technet.microsoft.com/en-us/library/cc261668(v=office.14).aspx (Still valid for SharePoint 2013)
Just copy a configuration file and add the following arguments

<Configuration>

…

<Setting Id="SETUPTYPE" Value="CLEAN_INSTALL"/>

<Display Level="Basic" CompletionNotice="No" AcceptEULA="Yes" />

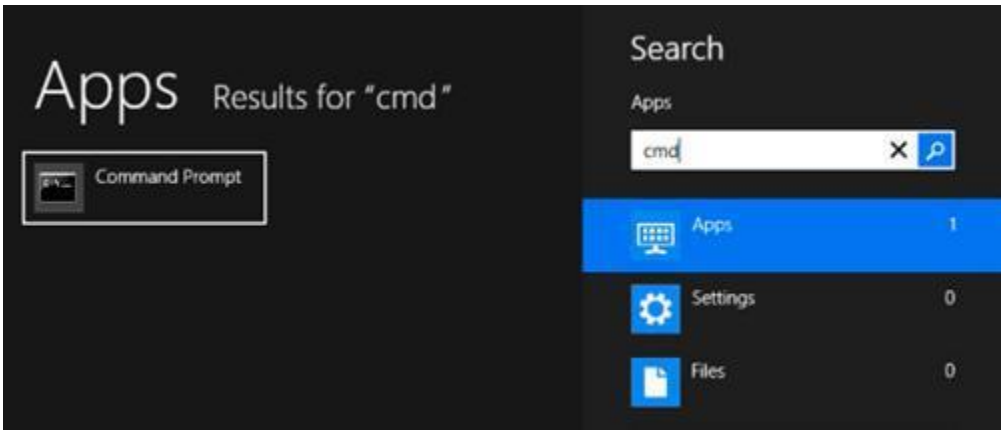
…

</Configuration>

With the following arguments you're saying that you accept the EULA and that there won't be any information (message box) about your installation.

-Use CALL

You should use **call** when you need to call another batch program (cmd script). Using 'call' will have no effect on any other executable file.
You should consider creating 1 batch file with all your scripts herein. This will become your central point for all your scripts. The CALL command is extremely important if you'll have a sequential installation and will ensure that every script gets executed even if in a script you'll have an "EXIT".



-Slipstream

Before the release of Microsoft SharePoint Server 2013 Service Pack 1 (SP1), Microsoft did not support the installation of SharePoint Server 2013 in a Windows Server 2012 R2-based environment. However, with the release of SharePoint Server 2013 SP1, this configuration is supported in Windows Server 2012 R2.

There are still few issues with the Service Pack 1 (SP1) of SharePoint but still, the main idea is the same. Please use a slipstreamed version that you can download from the Microsoft Developer Network (MSDN), the Volume Licensing Service Center (VLSC), and the Microsoft Partner Network (MPN). You should avoid creating your own slipstream versions because of a change in the package layout that was introduced with the March 2013 product update.

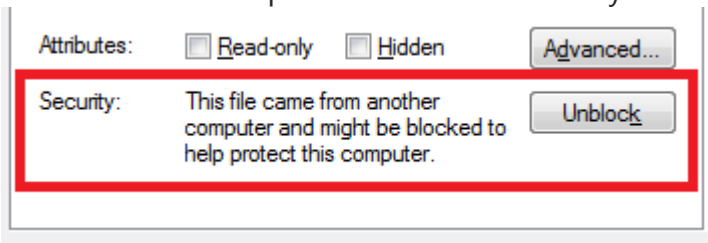
SharePoint 2013 "with" SP1 is the supported installation scenario on Server 2012 R2. SharePoint 2013 RTM and upgrade to SP1, or self-slipstreaming SP1 is not supported for an installation Server 2012 R2.

-Local Intranet and files that came from another computer

Not all servers in companies have access to the “internet”. Security wise we can’t better but this isn’t the best User Experience ever. If you didn’t download and copied the sources (Prerequisites and install files) locally, bah, you could have a serious problem.

If you’ll have several server with no internet access please download the necessary files and place them somewhere. This somewhere can be somewhere local or a shared folder.

- If you’re using a Shared Folder please be sure to add the \\Share\Folder to Local Intranet otherwise you’ll receive a **pop up during** the installation that the sources are on an unsafe location.
- If you downloaded the executables please also be sure that you unblock these.



-Start-Sleep

We noticed that even if the script finishes his job that SQL Server was still busy with the configuration of his components. I’m not pretty sure if it’s a best practice but I’ll recommend you to use the –Start-Sleep cmdlet.

The **Start-Sleep** cmdlet (which is probably more useful in scripts than it is as a command-line command) enables you to pause Windows PowerShell activity for a specified period of time. For example, this command pauses Windows PowerShell for 600 seconds:
Start-Sleep -s 600

This kind of cmdlets will prevent you from unnecessary errors as creating your SharePoint Farm even when SQL Server did not finish his job etc…

-/unattended

```
set PreReqPath=c:/_installation/prerequisites
PrerequisiteInstaller.exe /SQLNCli:%PreReqPath%\sqlncli.msi  ^
    /PowerShell:%PreReqPath%\Windows6.1-KB2506143-x64.msu  ^
    /NETFX:%PreReqPath%\dotNetFx45_Full_x86_x64.exe  ^
    /IDFX:%PreReqPath%\Windows6.1-KB974405-x64.msu  ^
    /Sync:%PreReqPath%\Synchronization.msi  ^
    /AppFabric:%PreReqPath%\WindowsServerAppFabricSetup_x64.exe  ^
    /IDFX11:%PreReqPath%\MicrosoftIdentityExtensions-64.msi  ^
    /MSIPCClient:%PreReqPath%\setup_msipc_x64.msi  ^
    /WCFDataServices:%PreReqPath%\WcfDataServices.exe  ^
    /KB2671763:%PreReqPath%\AppFabric1.1-RTM-KB2671763-x64-ENU.exe
```

This is the code that you'll see in many scripts. When defining your prerequisites in a variable please be sure that you add the **/unattended** argument. This will ensure that all the software's are installed in a NO user intervention way.

$param=**'/unattended**' + `
'/SQLNCli:"C:\temp\SP2013-prereq\sqlncli.msi" ' + `

'/IDFX:"C:\temp\SP2013-prereq\Windows6.1-KB974405-x64.msu" ' + `

'/IDFX11:"C:\temp\SP2013-prereq\MicrosoftIdentityExtensions-64.msi" ' + `

'/AppFabric:"C:\temp\SP2013-prereq\WindowsServerAppFabricSetup_x64.exe" ' + `

'/KB2671763:"C:\temp\SP2013-prereq\AppFabric1.1-RTM-KB2671763-x64-ENU.exe" ' + `

'/MSIPCClient:"C:\temp\SP2013-prereq\setup_msipc_x64.msi" ' + `

'/WCFDataServices:"C:\temp\SP2013-prereq\WcfDataServices.exe" ' + `

'/Sync:"C:\temp\SP2013-prereq\Synchronization.msi"'

Start-Process -filepath $SourcePRE -Argument $param -PassThru -Wait –NoNewWindow

Conclusion

As you can see there are many ways to improve an unattended script. I just wanted to share real problems I had during the scripting of my fully automated unattended script!

PowerShell: Get Process for SharePoint 2013 NodeRunners

Save the code below to a file named something like getNodeRunnerProcess.ps1 and then run it using any of the following:

- To get all the noderunner processes on the **current server**:
  - **getNodeRunnerProcess.ps1**
- To get all the noderunner processes on a **specified server**:
  - **getNodeRunnerProcess.ps1 someServerName**
- To get the noderunner for a **specific component** name (assuming you only have one SSA in the farm and the name specified matches at least one of the active components):
  - **getNodeRunnerProcess.ps1** IndexComponent1

Code Sample:

```
if ($args[0] -ne $null) {

    $SSA = Get-SPEnterpriseSearchServiceApplication

    $firstParam = $args[0]

       $targetComponent = $SSA.ActiveTopology.GetComponents() | Where
{$_.Name -ieq $firstParam}


       #if the command line arg specifies an existing search component

       if ($targetComponent -ne $null) {


              #Set the server name where this component is running

              $serverName = $targetComponent.ServerName


              #And set the $targetComponent as this parameter

              $targetComponentName = $firstParam
```

```
        } else {
                #assume the command line arg references another server
                $serverName = $firstParam
        }
} else {
        #default to all components on this local server
        $serverName = $ENV:COMPUTERNAME
}

Write-Host ("Server: " + $serverName)
if ($targetComponentName -ne $null) {
    Write-Host ("Comp: " + $targetComponentName)
}
Write-Host ("======================================")

$nodeRunnerProcesses = (Get-Process noderunner -ComputerName
$serverName -ErrorAction SilentlyContinue)
foreach ($noderunner in $nodeRunnerProcesses) {
        $noderunner | Add-Member -Force -MemberType NoteProperty -Name
_ProcessCommandLine -Value $(
                (Get-WmiObject Win32_Process -ComputerName $serverName |
where {$_.processId -eq $noderunner.id}).CommandLine
        )
        if ($targetComponentName -ne $null) {
```

```
                if ($noderunner._ProcessCommandLine -ilike $("*" +
$targetComponentName + "*")) {

                        $noderunner

                }

        } else {

                $delimPos =
$($noderunner._ProcessCommandLine).IndexOf("Nodes")

                $stub =
$($noderunner._ProcessCommandLine).SubString($delimPos)

                $delimPos = $stub.IndexOf("`"")

                $derivedCompName = $stub.SubString(0,$delimPos)

                $derivedCompName

                Write-Host ("------------------------")

                $noderunner

            Write-Host

        }

}
```

----------Example Output----------

```
PS D:\toolbox> .\getNodeRunnerProcess.ps1


Server: SPServer5


=======================================
```

```
IndexComponent1

------------------------------
Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
   1612     157   511112     326368  1781 6,776.89   2488 noderunner
QueryProcessingComponent1

------------------------------
   1885     208   365876     372716 -1142   651.38   2564 noderunner

PS D:\toolbox> .\getNodeRunnerProcess.ps1 spserver3

Server: SPServer3

========================================

AdminComponent1
```

```
-----------------------------
Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
   2070     192   543016     552396  1542            2680 noderunner
AnalyticsProcessingComponent1

-----------------------------
   1930     188   259624     272752 -1285            2912 noderunner
ContentProcessingComponent1

-----------------------------
   1959     208   847500     904156    47            2920 noderunner

PS D:\toolbox> .\getNodeRunnerProcess.ps1 AdminComponent1

Server: SPServer3

Comp: AdminComponent1
```

```
=======================================

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName

-------  ------    -----      ----- -----   ------     -- -----------

   2136     192   543084     552432  1544            2680 noderunner
```

SharePoint 2013: Multilingual user interface (MUI) supported features

Following are the list of features that are supported in multilingual user interface (MUI).

Component	MUI Support
Settings pages, such as those in the _layouts and the _admin virtual directories	✓
SharePoint Help	✓
Site Actions	✓
Site Title and description	✓
List or Library Title & description	✓
Top link bars links	✓
Menus	✓
Controls	✓
Features and Solutions	✓
Quick Launch links	✓
Local breadcrumbs	✓
Site and list content types	✓
Site and list columns	✓
List-based Web Parts like, **Announcements** and **Shared Documents**	✓
Links to list titles in the Quick Launch	✗
Global breadcrumbs	✗
List item data	✗
Documents	✗
Web pages in libraries	✗
Custom permissions levels	✗
Groups	✗
Views	✗
User Web Parts	✗
Site template - Blog template	✗
Site Template - Meeting workspace templates	✗
Site Template - Web database templates	✗
User Information such as ,**About Me**, **Title** and **Department**	✗
Non-list based Web Parts like ,Content Editor' and ,Content Query	✗
In web parts its component such as web part title, description, custom properties and import error message	✗

SharePoint 2013: Create a Custom WCF REST Service Hosted in SharePoint and Deployed in a WSP

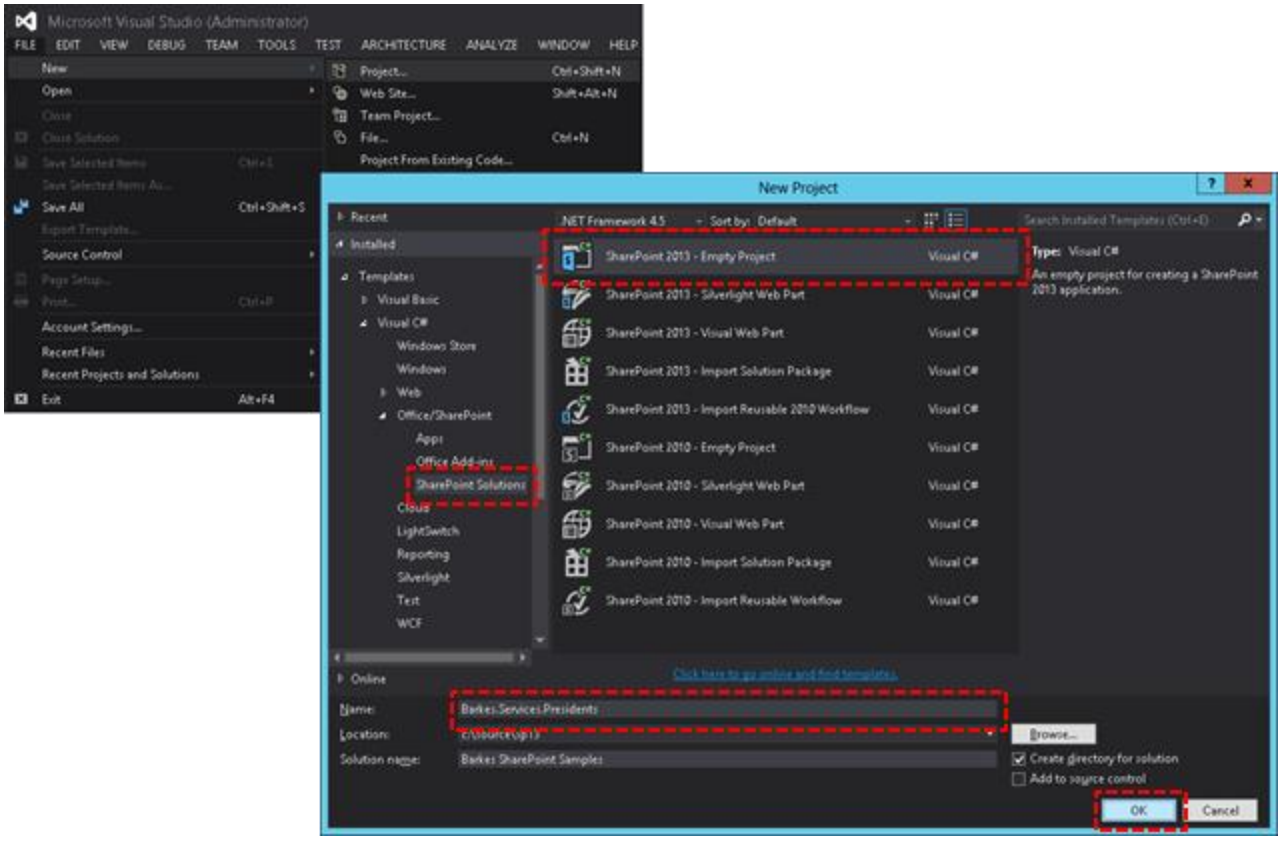
SharePoint 2013 provides a robust Representational State Transfer (REST) interface that allows any technology that supports standard REST capabilities to interact with SharePoint (sites, libraries, lists, etc). In addition to the built-in SharePoint REST API, you can create your own custom Windows Communication Foundation (WCF) REST services that are hosted in SharePoint. In this example, we'll explore the steps necessary to create a SharePoint-hosted WCF service with a REST interface that is deployed via a SharePoint solution (.wsp).

Download the source (50.8 KB)

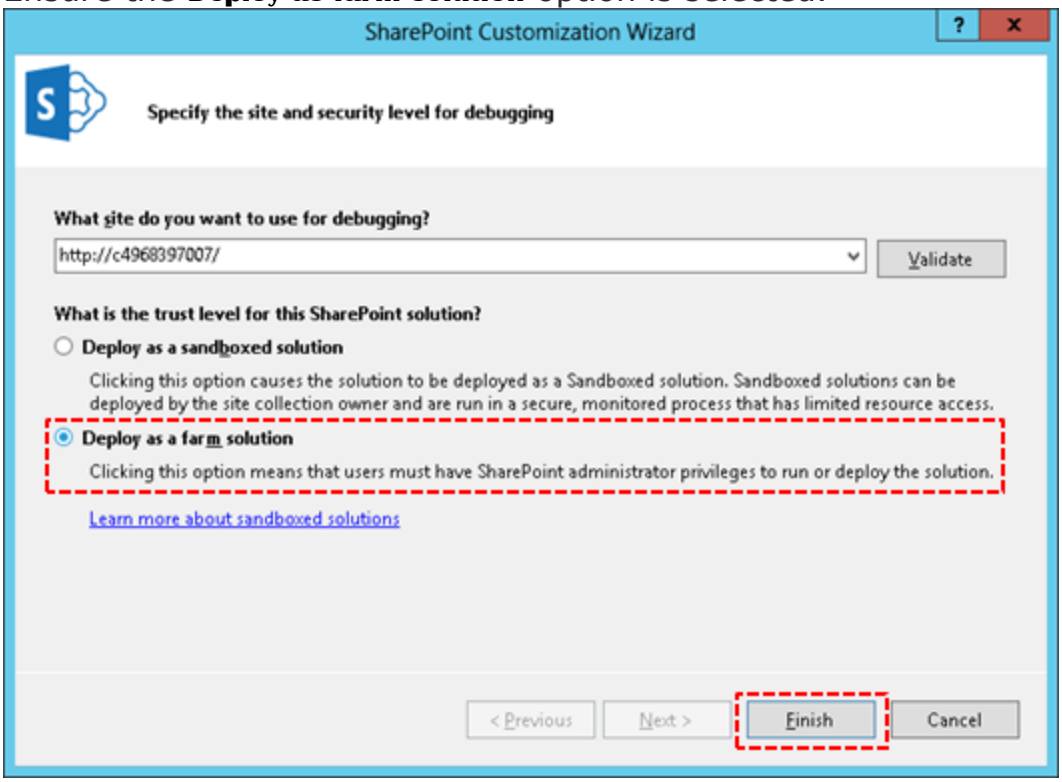
Step-by-Step Instructions

1. Run **Visual Studio 2013** as an Administrator.

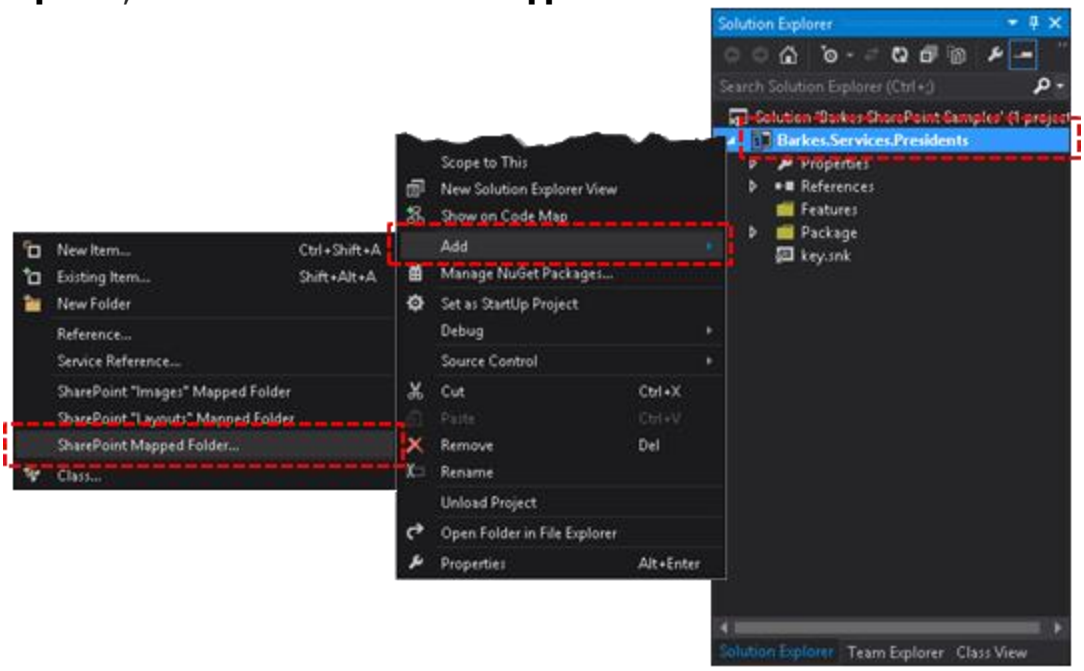
2. Create a new project named **Barkes.Services.Presidents** using the **SharePoint 2013 - Empty Project** template from the Visual C# **SharePoint Solutions** category.


Microsoft Visual Studio (Administrator)
FILE EDIT VIEW DEBUG TEAM TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
New
Open
Close
Close Solution
Save Selected Items Ctrl+S
Save Selected Items As...
Save All Ctrl+Shift+S
Export Template...
Source Control
Page Setup...
Print... Ctrl+P
Account Settings...
Recent Files
Recent Projects and Solutions
Exit Alt+F4
Project... Ctrl+Shift+N
Web Site... Shift+Alt+N
Team Project...
File... Ctrl+N
Project From Existing Code...
New Project
Recent
Installed
Templates
Visual Basic
Visual C#
Windows Store
Windows
Web
Office/SharePoint
Apps
Office Add-ins
SharePoint Solutions
Cloud
LightSwitch
Reporting
Silverlight
Test
WCF
Online
.NET Framework 4.5
Sort by: Default
SharePoint 2013 - Empty Project Visual C#
SharePoint 2013 - Silverlight Web Part Visual C#
SharePoint 2013 - Visual Web Part Visual C#
SharePoint 2013 - Import Solution Package Visual C#
SharePoint 2013 - Import Reusable 2010 Workflow Visual C#
SharePoint 2010 - Empty Project Visual C#
SharePoint 2010 - Silverlight Web Part Visual C#
SharePoint 2010 - Visual Web Part Visual C#
SharePoint 2010 - Import Solution Package Visual C#
SharePoint 2010 - Import Reusable Workflow Visual C#
Search Installed Templates (Ctrl+E)
Type: Visual C#
An empty project for creating a SharePoint 2013 application.
Click here to go online and find templates.
Name: Barkes.Services.Presidents
Location: c:\source\sp1\
Solution name: Barkes SharePoint Samples
Browse...
Create directory for solution
Add to source control
OK
Cancel

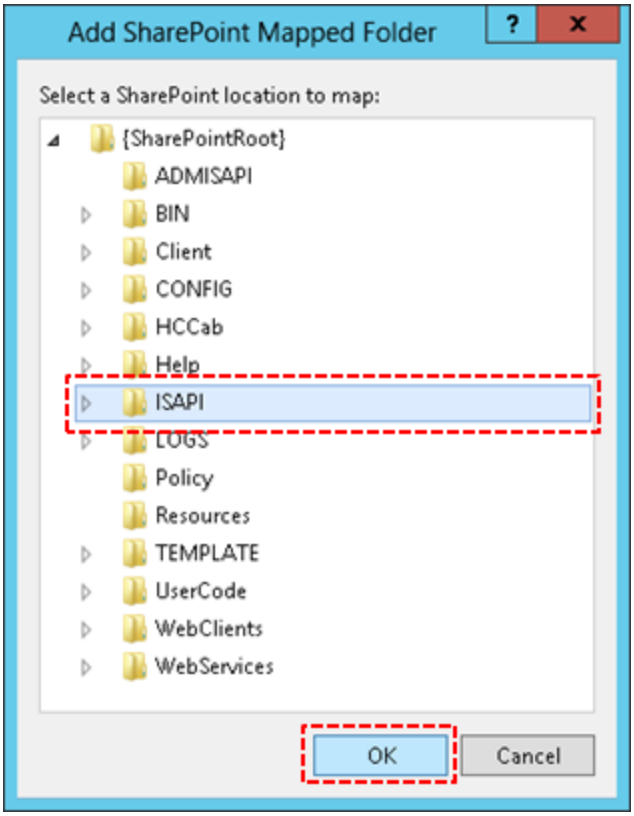
3. Ensure the **Deploy as farm solution** option is selected.



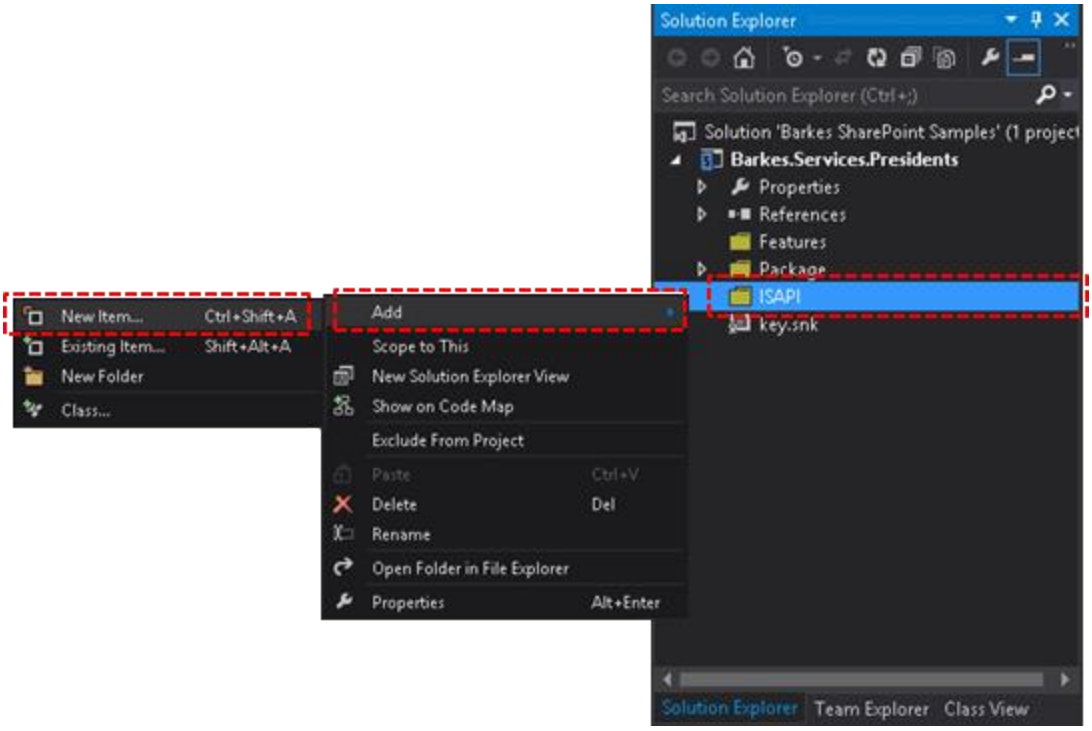
4. After the project is created, right-click on the project in the **Solution Explorer**, then **Add -> SharePoint Mapped Folder**.



5. On the **Add SharePoint Mapped Folder** dialog, select the **ISAPI** folder and click **OK**.

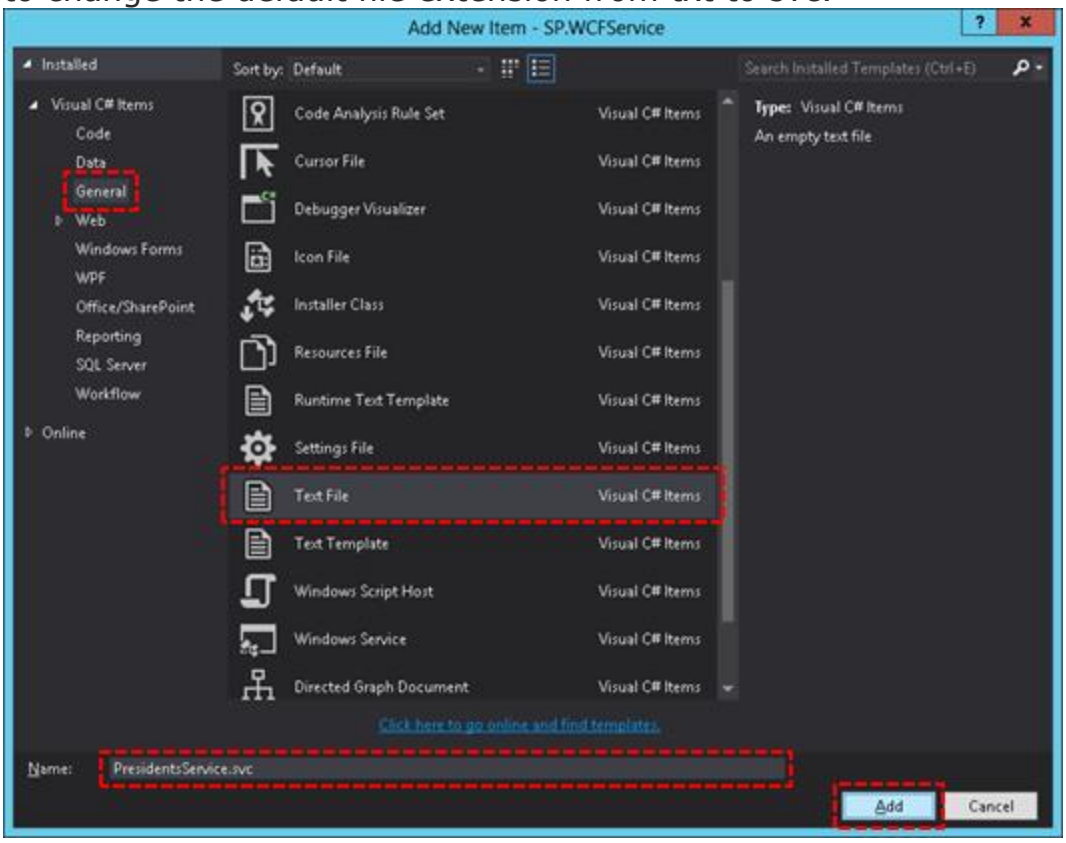


6. Right-click on the **ISAPI** folder in the **Solution Explorer**, then **Add -> New Item.**



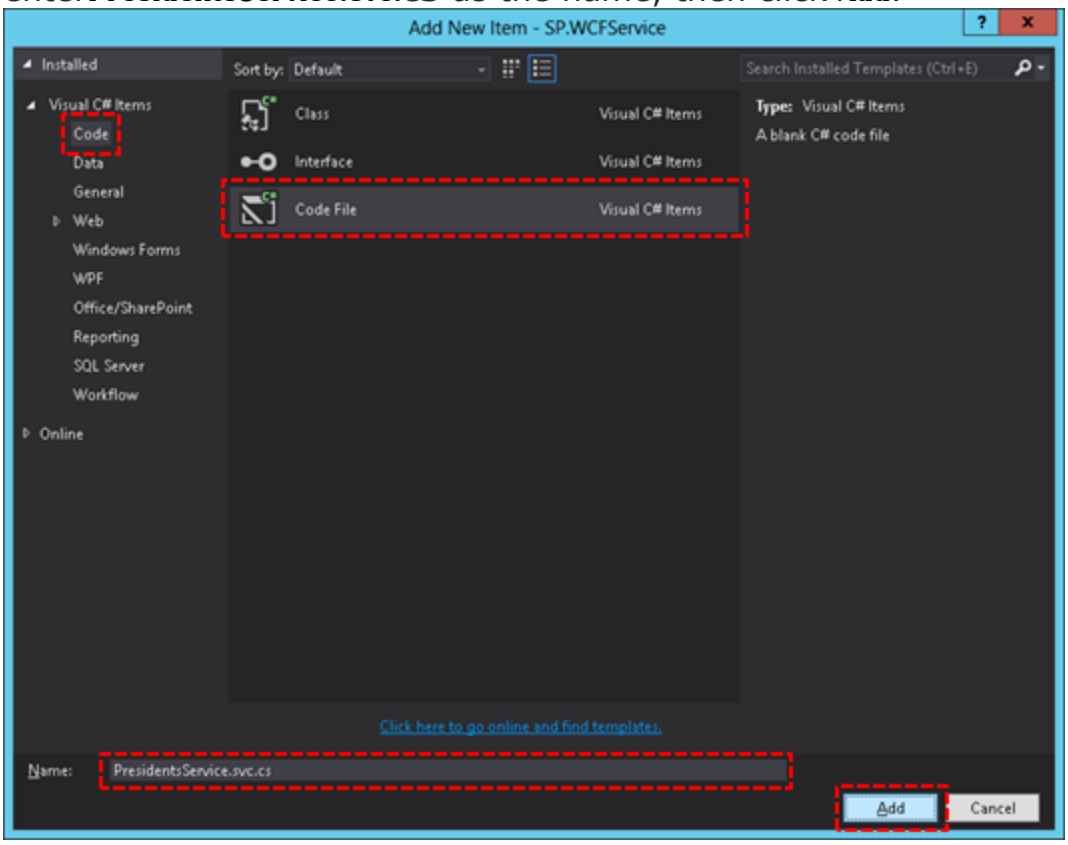
7. On the **Add New Item** dialog, select **Text File** from the **General** category and enter **PresidentsService.svc** *as the name, then click* **Add**. Make sure

to change the default file extension from txt to svc.



8. Right-click on the **ISAPI** folder in the **Solution Explorer**, then **Add -> New Item.**

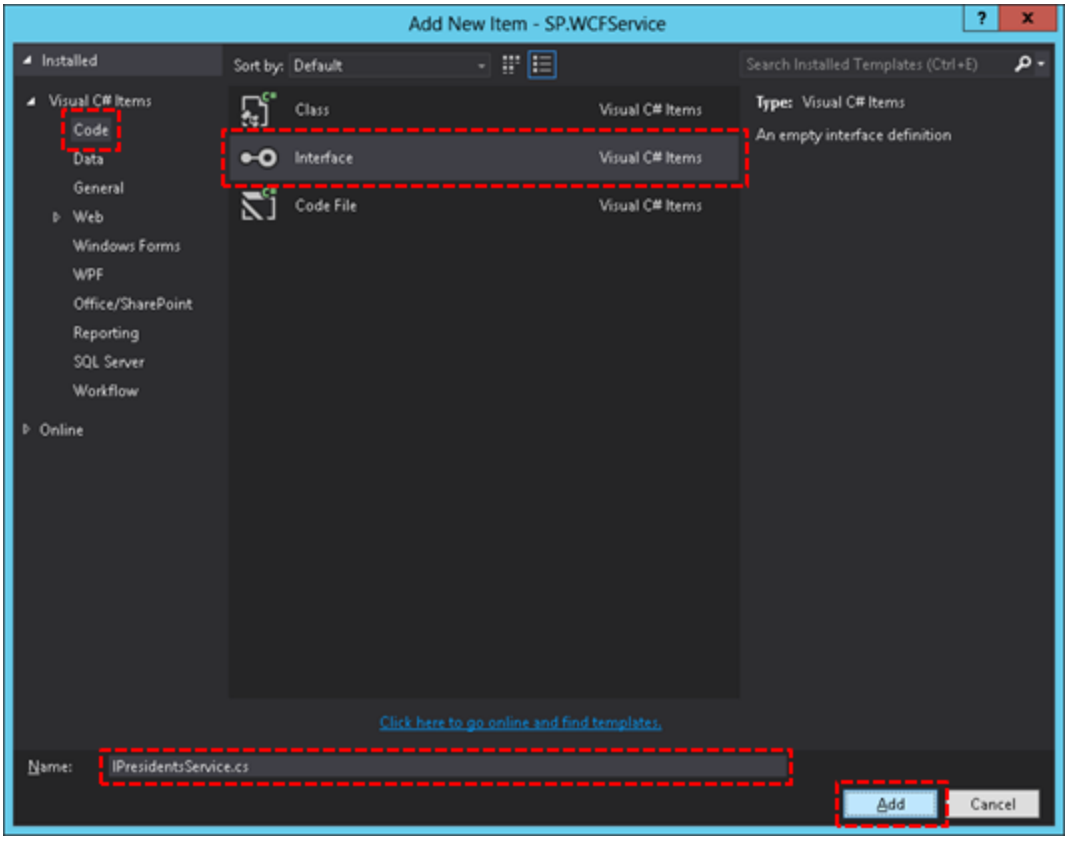
9. On the **Add New Item** dialog, select **Code File** from the **Code** category and enter**PresidentsService.svc.*cs*** as the name, then click **Add**.



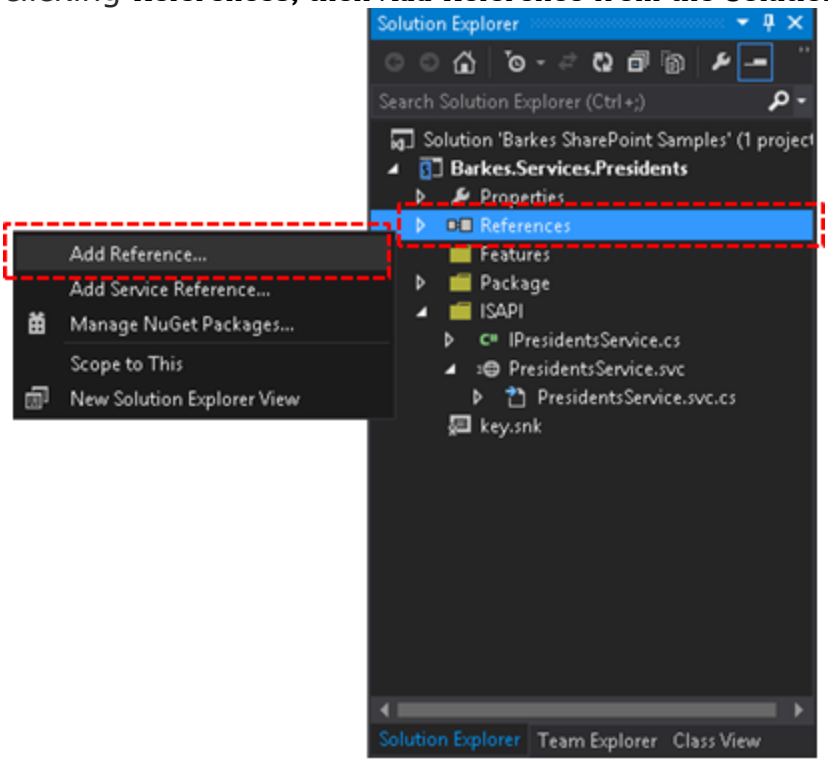
10. Right-click on the **ISAPI** folder in the **Solution Explorer, then Add -> New Item.**

11. On the **Add New Item** dialog, select **Interface** from the **Code** category and enter**IPresidentsService.cs** as the name, then

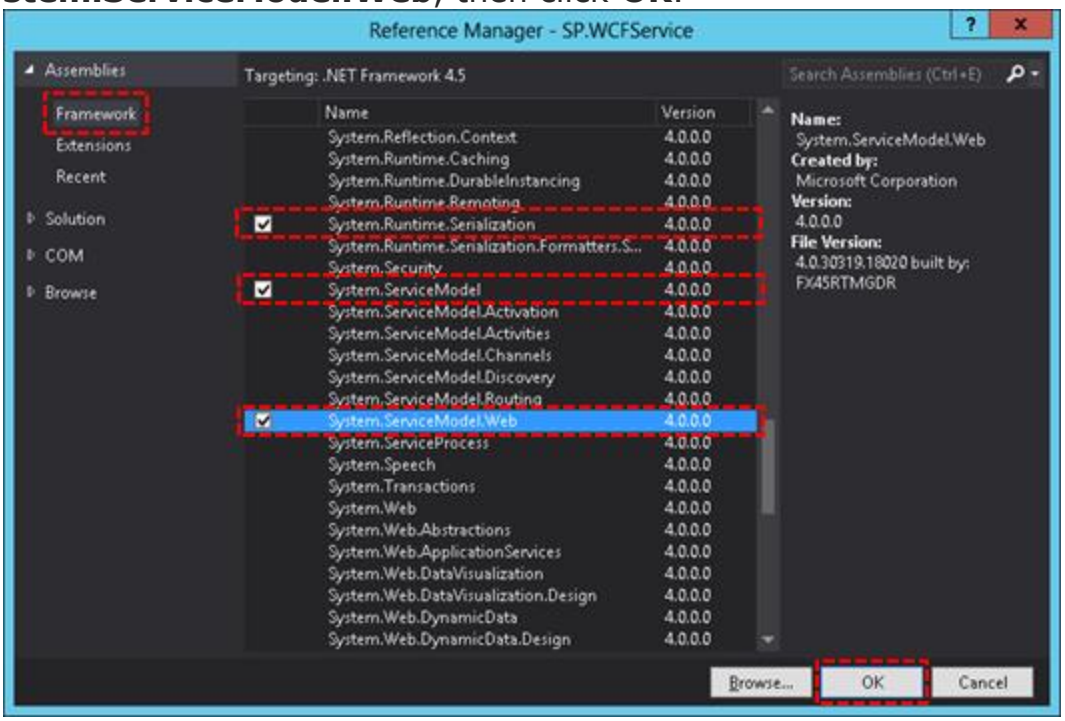
click **Add**.


Add New Item - SP.WCFService
Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
Office/SharePoint
Reporting
SQL Server
Workflow
Online
Sort by: Default
Class
Visual C# Items
Interface
Visual C# Items
Code File
Visual C# Items
Search Installed Templates (Ctrl+E)
Type: Visual C# Items
An empty interface definition
Click here to go online and find templates.
Name:
IPresidentsService.cs
Add
Cancel

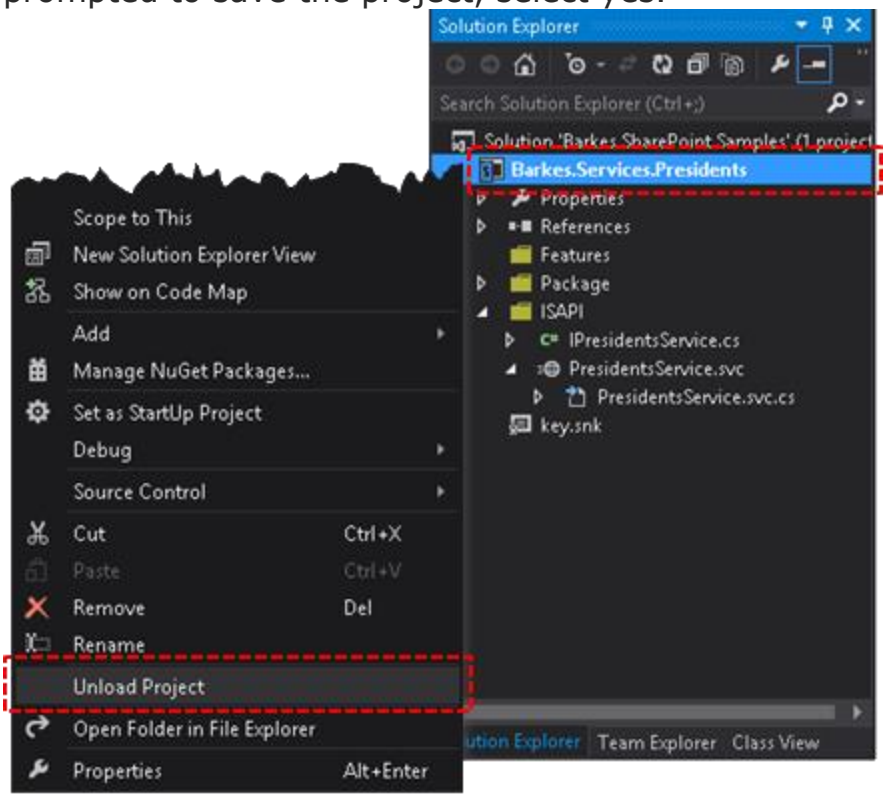
12. Add the required assembly references by right-clicking **References, then Add Reference from the Solution Explorer**.



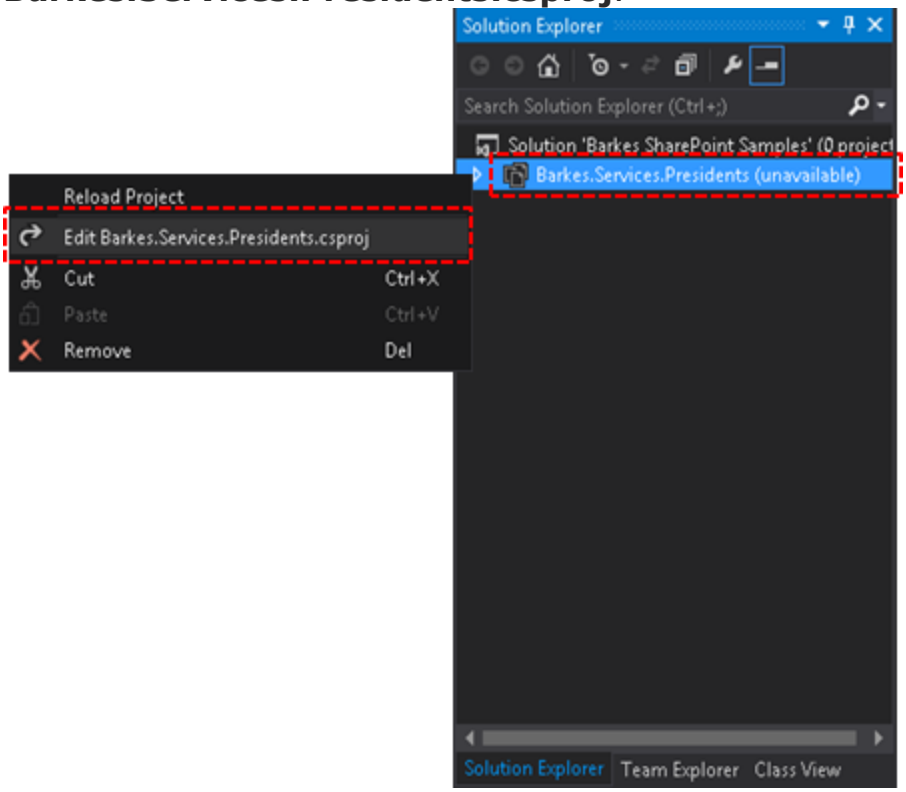
13. On the **Reference Manager** dialog, select **Framework** and check**System.Runtime.Serialization**, **System.ServiceModel** and**System.ServiceModel.Web**, then click **OK**.



14. By default Visual Studio does not support token replacements in .SVC files. In order to use the **$SharePoint.Project.AssemblyFullName$** token, right-click on the project in**Solution Explorer**, then **Unload Project**. If you are prompted to save the project, select yes.



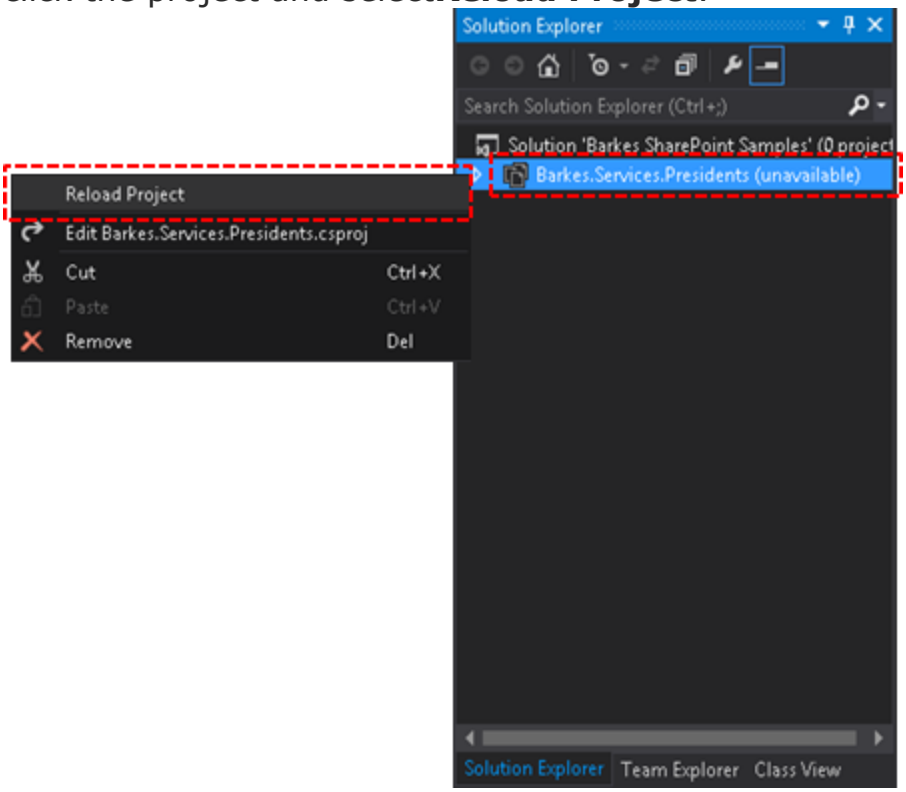
15. Right-click the project in the **Solution Explorer**, then **Edit Barkes.Services.Presidents.csproj**.



16. In the first **PropertyGroup** (toward the top of the project file), add the**TokenReplacementFileExtensions** element beneath the SandboxedSolution element and set its value to **svc**. Don't forget to save the changes to the project file.

```
<PropertyGroup>
    ...
    <SandboxedSolution>False</SandboxedSolution>

<TokenReplacementFileExtensions>svc</TokenReplacementFileExtensions>
</PropertyGroup>
```

21. After you've made the required manual project changes, right-click the project and select**Reload Project**.



22. Open ***PresidentsService.svc*** and enter the following service declaration. Note that the use of the SharePoint-specific ***MultipleBaseAddressWebServiceHostFactory*** replaces the need to specify endpoint configurations in a web.config.

```
<%@ ServiceHost Language="C#" Debug="true"
    Service="Barkes.Services.Presidents.PresidentsService,
$SharePoint.Project.AssemblyFullName$"
    CodeBehind="PresidentsService.svc.cs"
    Factory="Microsoft.SharePoint.Client.Services.MultipleBaseAddressWebServiceHostFactory,
    Microsoft.SharePoint.Client.ServerRuntime,
Version=15.0.0.0, Culture=neutral,
    PublicKeyToken=71e9bce111e9429c" %>
```

28. Open ***IPresidentsService.cs*** and enter the following interface definition, with associated**ServiceContract** and **OperationContracts**.

```
1. using System;
2. using System.Collections.Generic;
3. using System.ServiceModel;
4. using System.ServiceModel.Web;
5. using Barkes.Services.Presidents.Model;
6.
7. namespace Barkes.Services.Presidents
8. {
9.     [ServiceContract]
10.     interface IPresidentsService
11.     {
12.         [OperationContract]
13.         [WebGet(UriTemplate = "GetAllPresidents",
14.             ResponseFormat = WebMessageFormat.Json)]
15.         List<President> GetAllPresidents();
16.
17.         [OperationContract(Name =
    "GetPresidentsByLastName")]
18.         [WebGet(UriTemplate =
    "GetPresidentsByLastName/{lastName}",
19.             ResponseFormat = WebMessageFormat.Json)]
20.         List<President> GetPresidentsByName(string
    lastName);
21.
22.         [OperationContract(Name =
    "GetPresidentsByLastFirstName")]
23.         [WebGet(UriTemplate =
    "GetPresidentsByLastFirstName/{lastName}/{firstName}",
24.             ResponseFormat = WebMessageFormat.Json)]
25.         List<President> GetPresidentsByName(string
    lastName, string firstName);
26.
27.         [OperationContract]
28.         [WebGet(UriTemplate = "GetPresidentById/{id}",
29.             ResponseFormat = WebMessageFormat.Json)]
30.         President GetPresidentById(string id);
31.
32.         [OperationContract]
33.         [WebInvoke(Method = "POST", UriTemplate =
    "AddPresident",
34.             RequestFormat = WebMessageFormat.Json,
35.             ResponseFormat = WebMessageFormat.Json)]
36.         bool AddPresident(President president);
```

```
37.     }
38. }
39.
```

29. Open ***PresidentsService.svc.cs*** and enter the following code to implement the service interface.

```
1. using Microsoft.SharePoint.Client.Services;
2. using System;
3. using System.Collections.Generic;
4. using System.Linq;
5. using System.ServiceModel.Activation;
6. using Barkes.Services.Presidents.Model;
7.
8. namespace Barkes.Services.Presidents
9. {
10.     [AspNetCompatibilityRequirements(RequirementsMode =
   AspNetCompatibilityRequirementsMode.Required)]
11.     public class PresidentsService : IPresidentsService
12.     {
13.         #region Private Members
14.
15.         private List<President> _presidents;
16.         private List<President> Presidents
17.         {
18.             get
19.             {
20.                 // If there aren't any presidents in our
   list, populate with samples
21.                 _presidents = _presidents ?? new
   List<President>(SampleData.SamplePresidents);
22.                 return _presidents;
23.             }
24.         }
25.
26.         #endregion
27.
28.         #region IPresidentsService Implementation
29.
30.         public List<President> GetAllPresidents()
31.         {
```

```
32.             return Presidents;
33.         }
34.
35.         public List<President> GetPresidentsByName(string
  lastName)
36.         {
37.             return GetPresidentsByName(lastName,
  string.Empty);
38.         }
39.
40.         public List<President> GetPresidentsByName(string
  lastName, string firstName)
41.         {
42.             var query = from President p in Presidents
43.                         where
  p.LastName.ToLower().Contains(lastName.ToLower())
44.                               &&
  (string.IsNullOrWhiteSpace(firstName)
45.                                   ? true
46.                                   :
  p.FirstName.ToLower().Contains(firstName.ToLower()))
47.                         select p;
48.
49.             return query.ToList();
50.         }
51.
52.         public President GetPresidentById(string id)
53.         {
54.             var query = from President p in Presidents
55.                         where p.Id == id
56.                         select p;
57.
58.             return query.FirstOrDefault();
59.         }
60.
61.         public bool AddPresident(President president)
62.         {
63.             Presidents.Add(president);
64.             return true;
65.         }
66.
67.         #endregion
68.
```

```
69.      }
70. }
```

30. Add a new folder named ***Model*** to the project by right-clicking on the project and selecting Add, then New Folder.

31. Add a new class in the **Model** folder named ***President.cs*** and enter the following class definition, with associated DataContract and DataMembers.

```
1. using System.Runtime.Serialization;
2.
3. namespace Barkes.Services.Presidents.Model
4. {
5.     [DataContract]
6.     public class President
7.     {
8.         [DataMember]
9.         public string Id { get; set; }
10.
11.          [DataMember]
12.          public string LastName { get; set; }
13.
14.          [DataMember]
15.          public string FirstName { get; set; }
16.
17.          [DataMember]
18.          public string EmailAddress { get; set; }
19.     }
20. }
```

32. Add a new class in the **Model** folder named ***PresidentsData.cs*** and enter the following sample data code. In a production application, this would typically come from a database. The presidents array is purposely abbreviated for readability - all the presidents are in the complete source.

```
1. namespace Barkes.Services.Presidents.Model
2. {
3.     public static class SampleData
4.     {
```

```
5.          // This array is purposely abbreviated for
  readability in this article.
6.          // The complete list of presidents is available in
  the source download.
7.          public static President[] SamplePresidents = new
  President[]
8.          {
9.              new President {
10.                 Id =  "1", FirstName = "George", LastName
  = "Washington",
11.                 EmailAddress = "gwashington@email.com" },
12.             new President {
13.                 Id =  "2", FirstName = "John", LastName =
  "Adams",
14.                 EmailAddress = "jadams@email.com" },
15.             new President {
16.                 Id =  "3", FirstName = "Thomas", LastName
  = "Jefferson",
17.                 EmailAddress = "tjefferson@email.com" },
18.             new President {
19.                 Id =  "4", FirstName = "James", LastName =
  "Madison",
20.                 EmailAddress = "jmadison@email.com" },
21.             new President {
22.                 Id =  "5", FirstName = "James", LastName =
  "Monroe",
23.                 EmailAddress = "jmonroe@email.com" },
24.             new President {
25.                 Id = "43", FirstName = "George W.",
  LastName = "Bush",
26.                 EmailAddress = "gbush@email.com" },
27.             new President {
28.                 Id = "44", FirstName = "Barack", LastName
  = "Obama",
29.                 EmailAddress = "bobama@email.com" },
30.         };
31.     }
32. }
```

33. Now you're ready to build the solution and deploy the WSP. After deployment, you'll find the **PresidentsService.svc** service declaration in the 15 hive at

C:\Program Files\Common Files\microsoft shared\Web Server Extensions\15\ISAPI\BarkesServices.

Call the Service from Managed Code

There are a number of different options (tools, libraries, etc) and articles available to help you consume a WCF REST service from managed code. The following is an excerpt from a Visual Studio unit test project that calls the service to return all presidents. The custom JSON helper class used to simplify the object (de)serialization is shown below as well.

```
1. // Be sure to update the url to point to the Presidents Service
   in your SP farm.
2. string url =
   "http://sp13.dev/_vti_bin/BarkesServices/PresidentsService.svc/G
   etAllPresidents";
3. string response = CallService(url);
4. List<President> presidents =
   JsonHelper.Deserialize<List<President>>(response);
5.
6. private string CallService(string serviceUrl)
7. {
8.     WebClient client = new WebClient();
9.     client.UseDefaultCredentials = true;
10.          client.Headers["Content-type"] = "application/json";
11.          client.Encoding = Encoding.UTF8;
12.          string response = response =
   client.DownloadString(serviceUrl);
13.
14.          return response;
15.      }
```

```
1. public class JsonHelper
2. {
3.     public static string Serialize<T>(T obj)
4.     {
5.         MemoryStream stream = new MemoryStream();
6.         DataContractJsonSerializer serializer = new
   DataContractJsonSerializer(typeof(T));
7.         serializer.WriteObject(stream, obj);
8.         stream.Position = 0;
9.         StreamReader reader = new StreamReader(stream);
10.              return reader.ReadToEnd();
```

```
11.          }
12.
13.          public static T Deserialize<T>(string data)
14.          {
15.              if (string.IsNullOrWhiteSpace(data)) return
   default(T);
16.              DataContractJsonSerializer serializer = new
   DataContractJsonSerializer(typeof(T));
17.              MemoryStream stream = new
   MemoryStream(Encoding.UTF8.GetBytes(data));
18.              return (T)serializer.ReadObject(stream);
19.          }
20.      }
```

Call the Service from JQuery

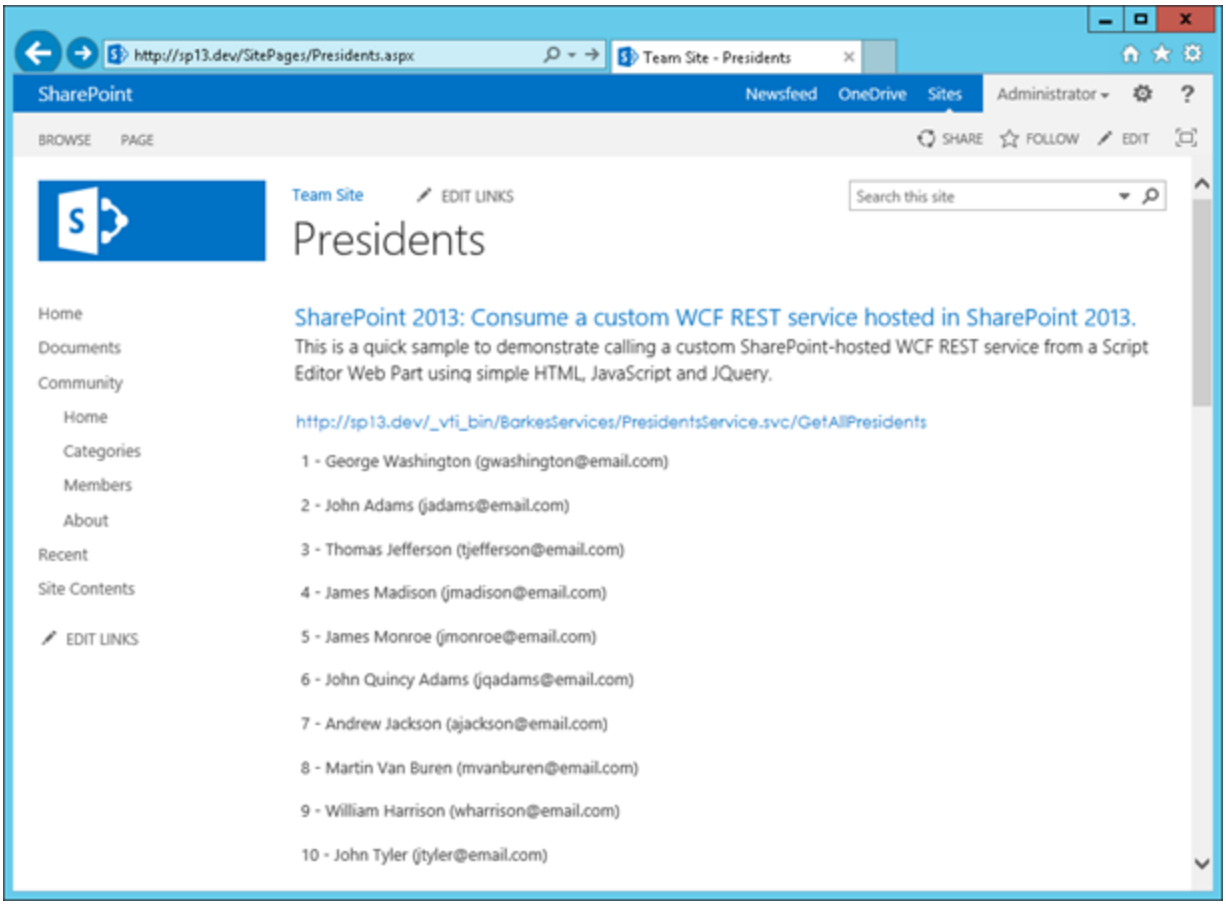
The following demonstrates how to call the service from a **Script Editor Web Part** using simple HTML, JavaScript and **JQuery**.

```
1. <script src="http://sp13.dev/SiteAssets/jquery-
   1.10.2.min.js"></script>
2.
3. <h2>SharePoint 2013: Consume a custom WCF REST service hosted in
   SharePoint 2013.</h2>
4. <h3>This is a quick sample to demonstrate calling a custom
   SharePoint-hosted WCF REST service from a
5.     Script Editor Web Part using simple HTML, JavaScript and
   JQuery.
6. </h3>
7.
8. <div>
9.     <br />
10.          <p id="message">Loading presidents...</p>
11.      </div>
12.
13.      <div id="resultsPanel"></div>
14.
15.      <script type="text/javascript">
16.          $(document).ready(function () {
17.              getPresidentsData();
18.          });
19.
20.      function getPresidentsData() {
```

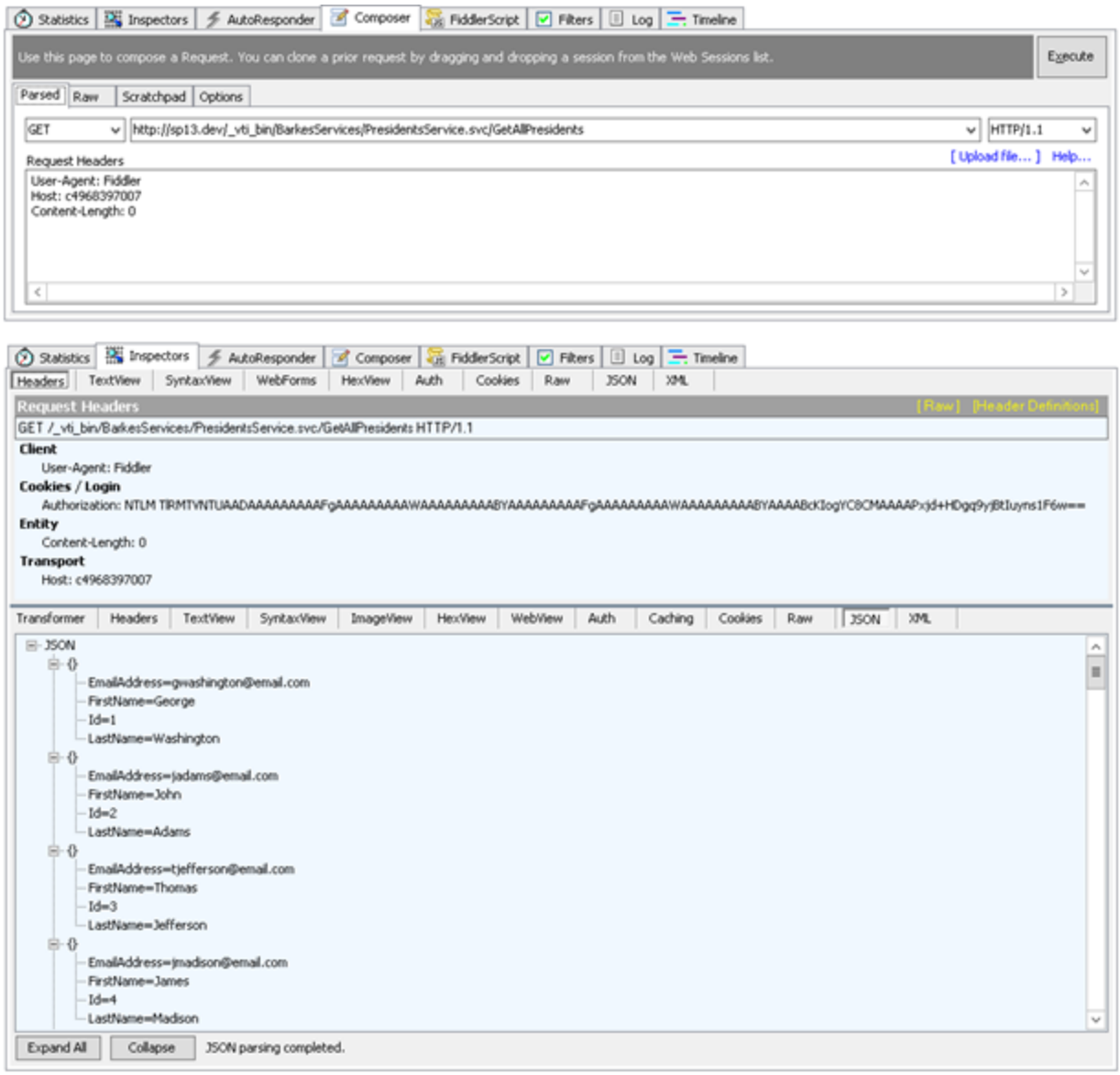
```
21.          var serviceUri = _spPageContextInfo.webAbsoluteUrl +
22.
   "/_vti_bin/BarkesServices/PresidentsService.svc/GetAllPresidents
   ";
23.          $.ajax({
24.              type: "GET",
25.              contentType: "application/json",
26.              url: serviceUri,
27.              dataType: "json",
28.              success:
29.                  function (response) {
30.                      showPresidentsList(response);
31.                      $('#message').html("<a href=" + serviceUri
   + ">" + serviceUri + "</a>");
32.                  },
33.              error:
34.                  function (err) {
35.                      alert(err);
36.                  }
37.          });
38.      }
39.
40.      function showPresidentsList(presidentsData) {
41.          $.each(presidentsData, function () {
42.              $('#resultsPanel').append($(this)[0].Id + ' - ');
43.              $('#resultsPanel').append($(this)[0].FirstName + '
   ');
44.              $('#resultsPanel').append($(this)[0].LastName + '
   (');
45.              $('#resultsPanel').append($(this)[0].EmailAddress
   + ')');
46.              $('#resultsPanel').append('<br><br>');
47.          });
48.      }
49.      </script>
```

Results Screenshots

Calling the Presidents Service from a **Script Editor Web Part** using simple HTML, JavaScript and**JQuery**. Of course you can use the resulting JSON data with Knockout and a variety of JavaScript/JQuery grids (JS Grid , simpleGrid , jqGrid , etc).



Interacting with the Presidents Service in Fiddler :



Download the source (50.8 KB)

For more SharePoint articles and samples, check out the Software Development Outpost .

Configure Power View (Reporting service Features) on existing SharePoint 2013 Farm With BI Features (Small Farm – APP & DB)

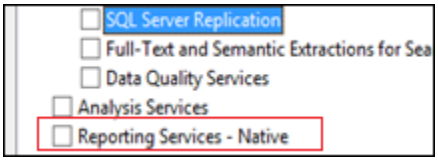
Install SQL Server 2012 SP1 CTP3

First you need need at least SQL Server 2012 SP1 CTP 3 installed in your farm. If not you can download and install it from here

Better if you can restart the machine after installation.

Uninstall Report Services – Native

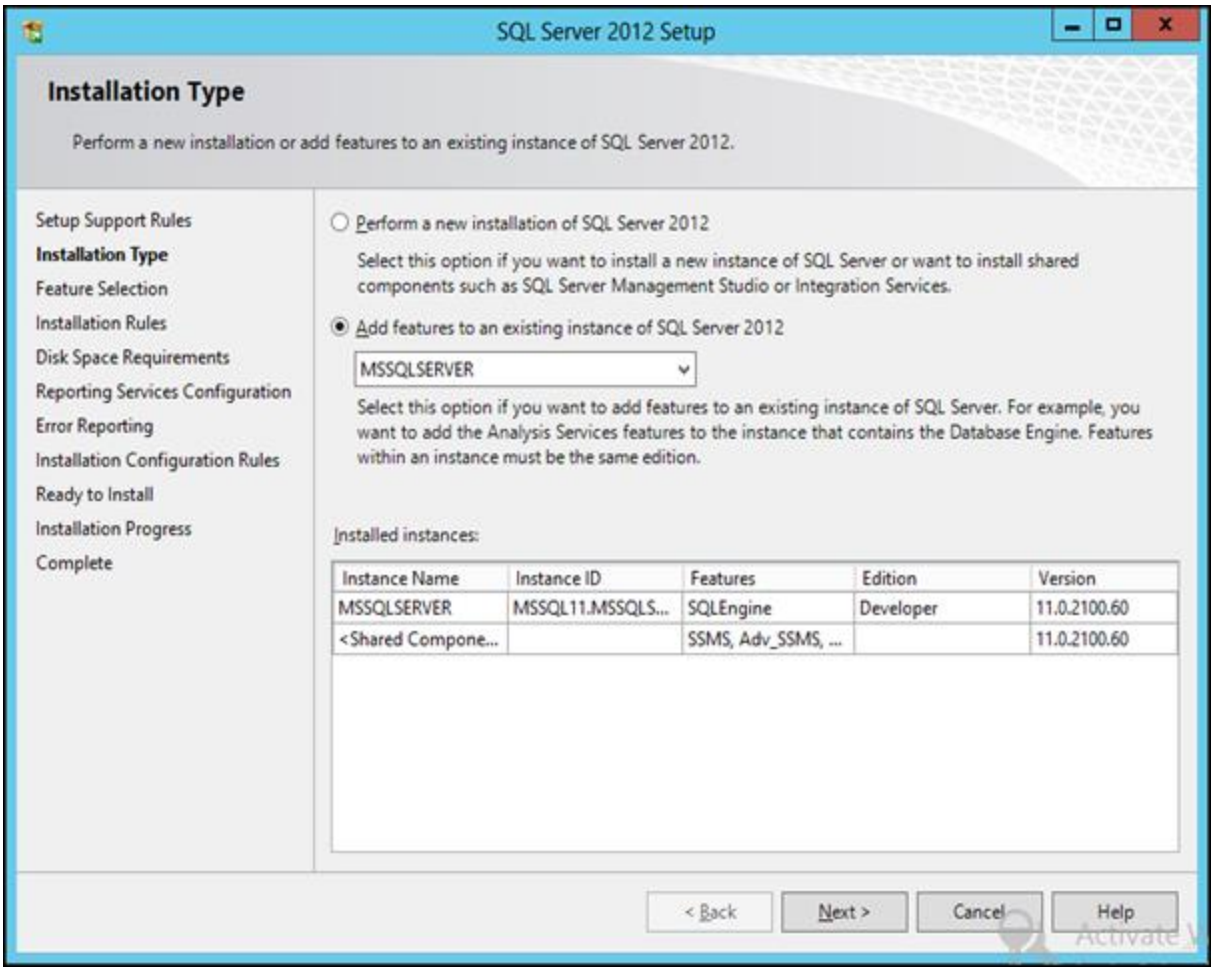
If You have already installed Reporting services Native go to Control Panel and Uninstall SQL Server and remove the Reporting Services – Native Feature.



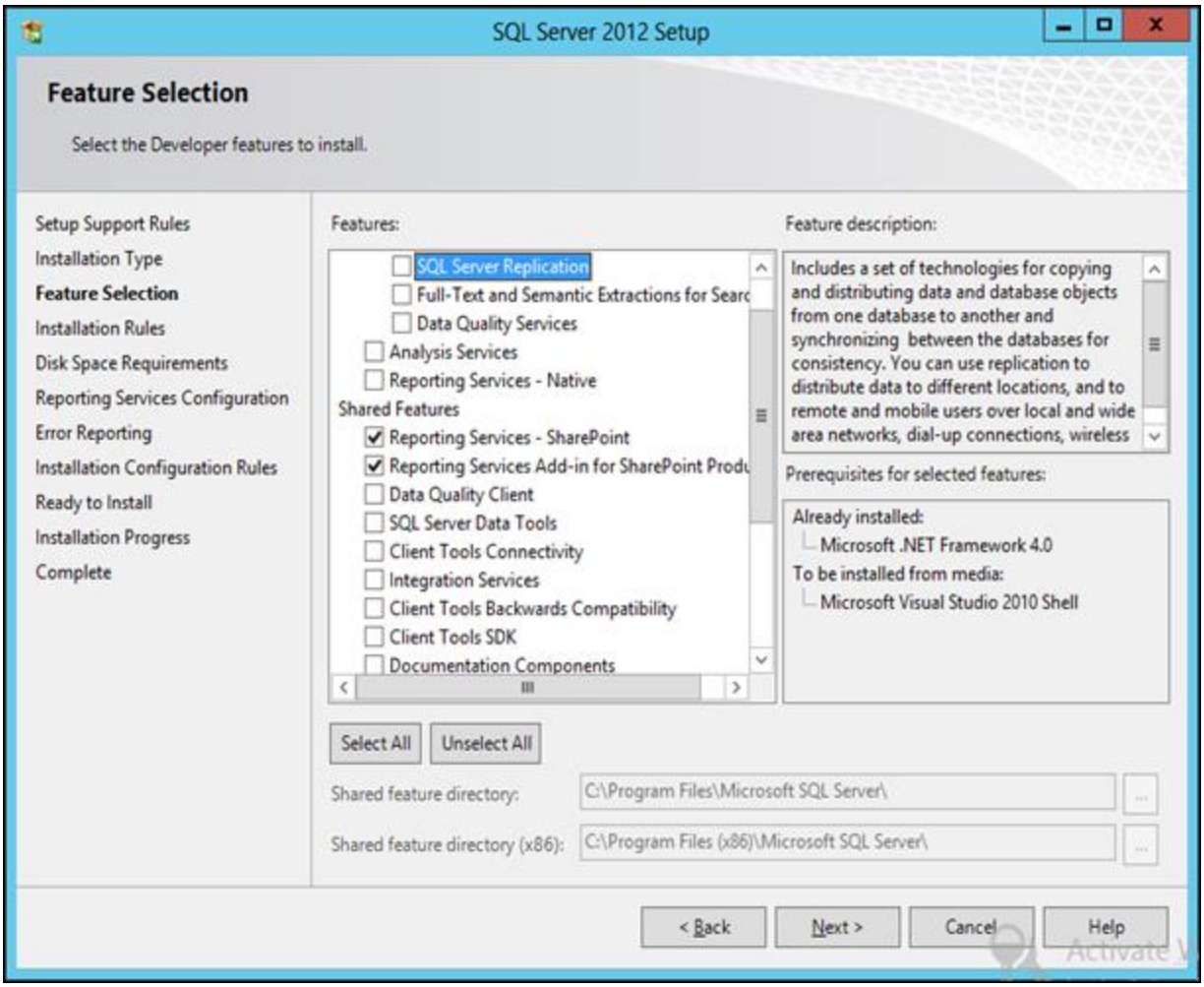
Install Reporting Services – SharePoint & Reporting Services Add-In for SharePoint Product

If you are not installed Reporting Services – SharePoint and Reporting services Add –ins

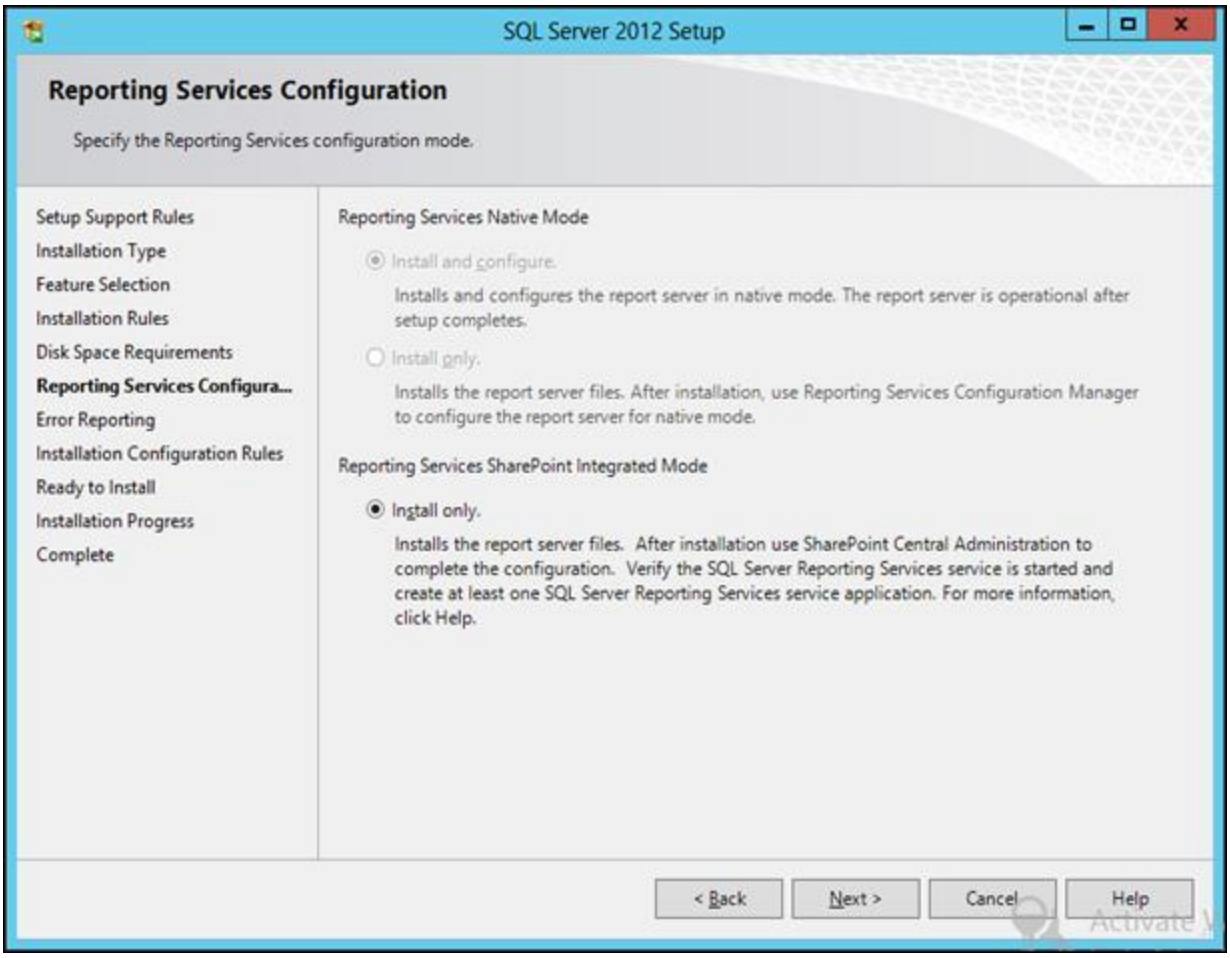
you need to run SQL Server 2012 and select **Add features to an existing instance of SQL Server 2012**



Select **Reporting Services – SharePoint and Reporting services Add –ins for SharePoint Products** and Proceed

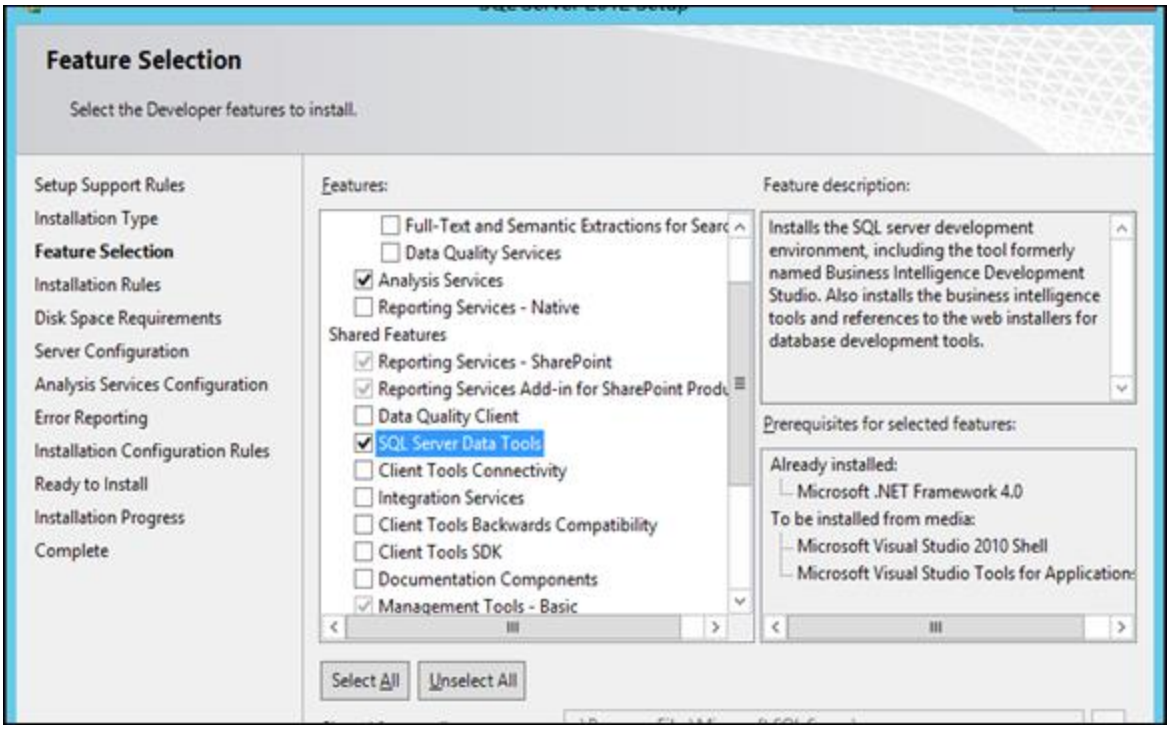


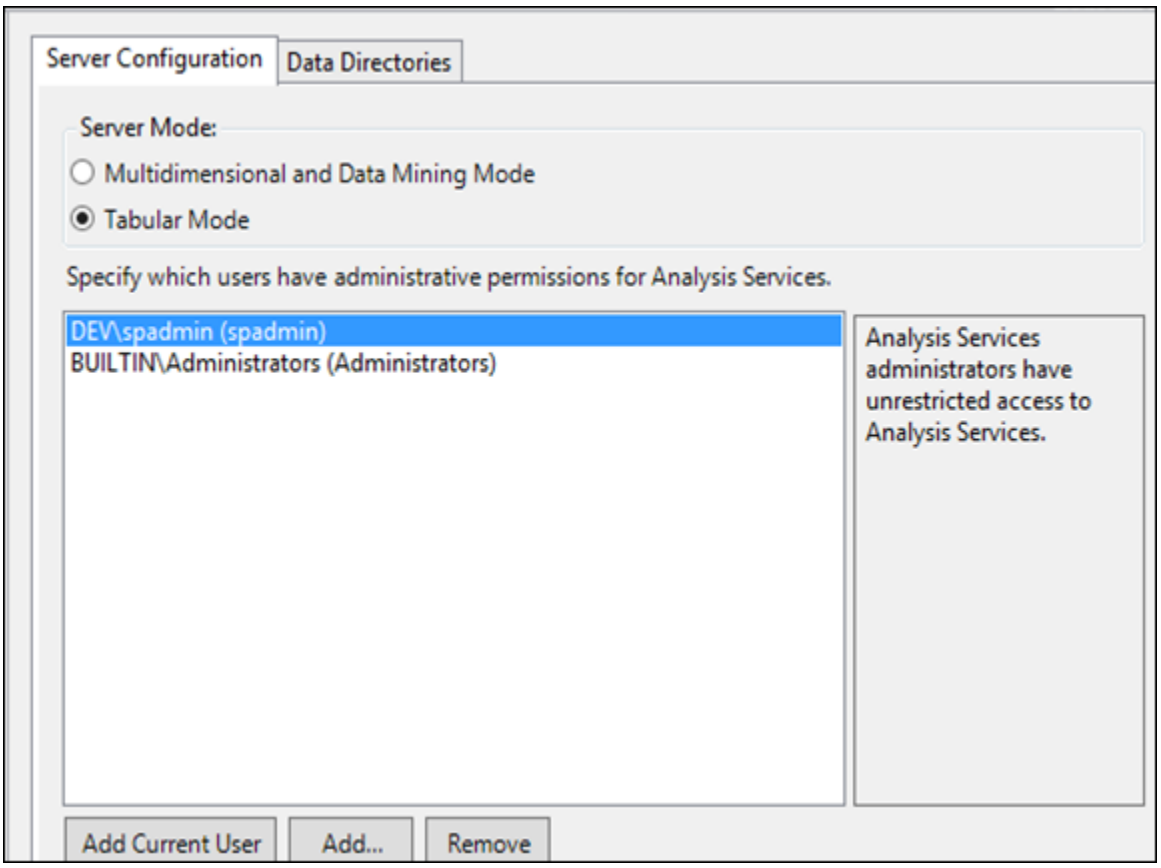
Then install as **Reporting Services SharePoint Integrated Mode**



Install Analysis Services In Tabular Mode and Data Tools

If you have already installed the Analysis services uninstall in and install it in **Tabular Mode.**



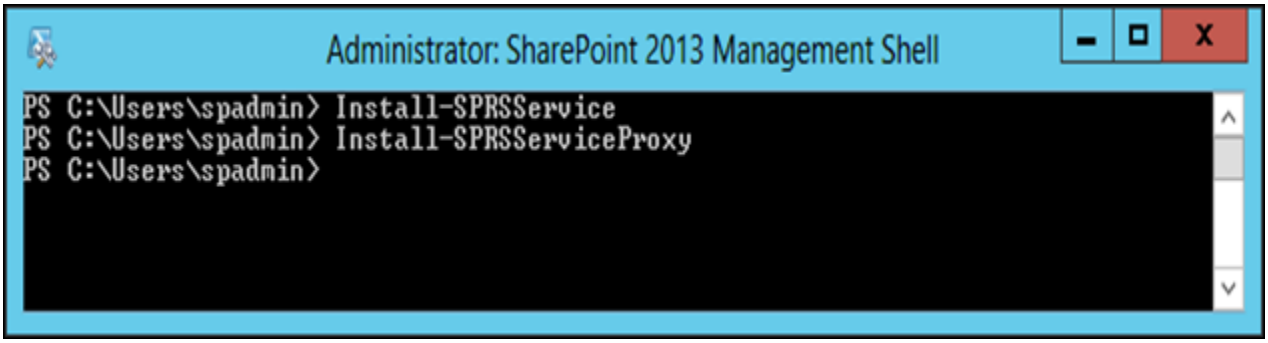


After the installation restart the machine . (This is not mandatory)

Install SharePoint Reporting Service

Open the SharePoint Management Shell and run following commands If every thing is fine it will return no errors.

- Install-SPRSService
- Install-SPRSServiceProxy



Note: If you come across this error you need to install SQL Server 2012 SP1 CTP3 again and restart the machine again.

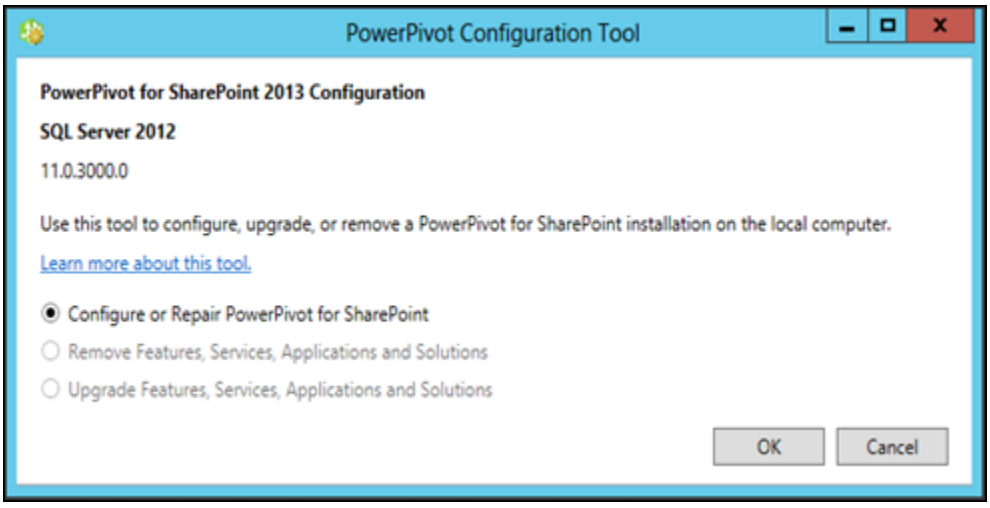
Install-SPRSService : The term 'Install-SPRSService' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the

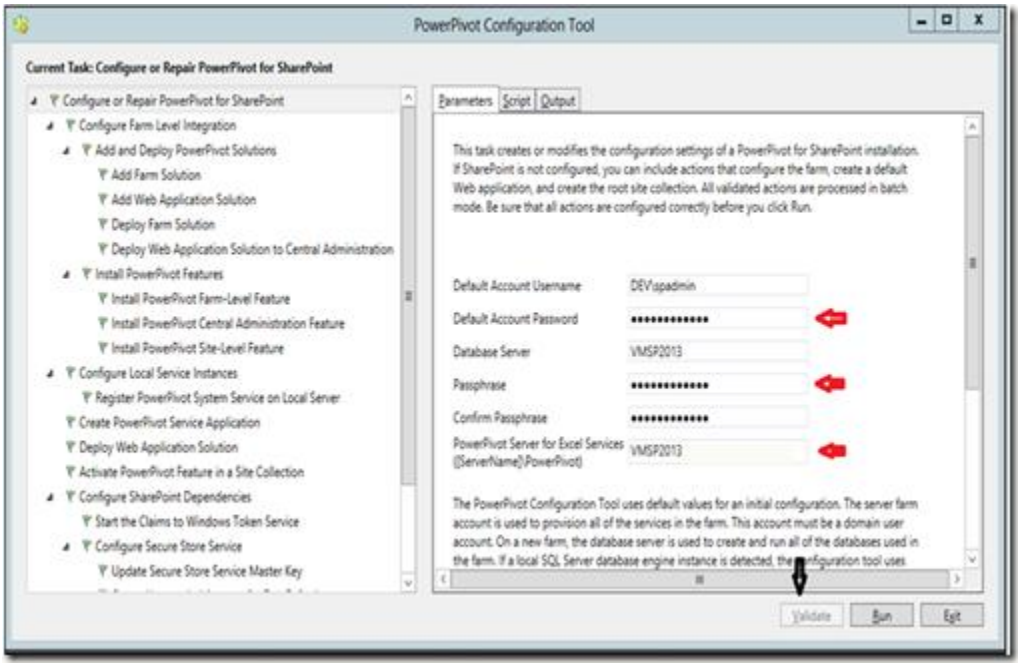
Install and Configure Power Pivot for SharePoint

Download PoverPivot for SharePoint. (http://www.microsoft.com/en-us/download/confirmation.aspx?id=35577)

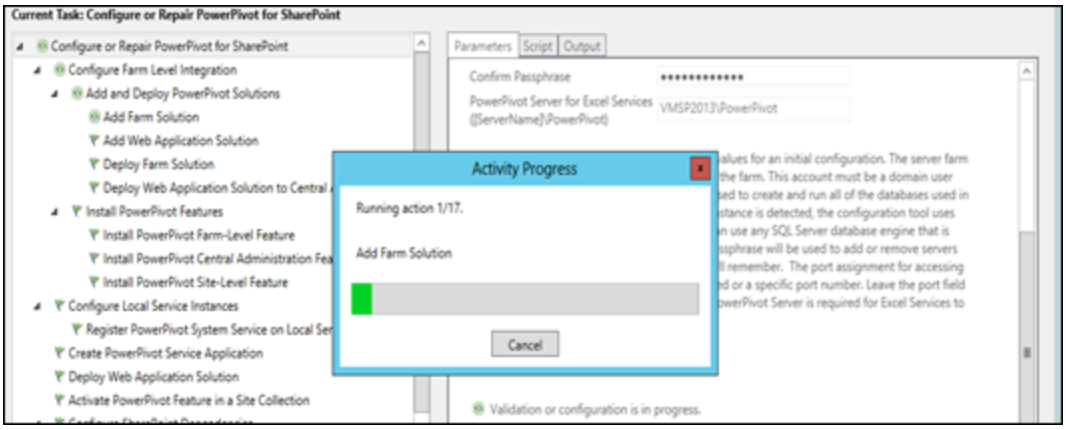
Run the setup as administrator.

After the installation you need to run the PowerPivot Configuration as a admin.



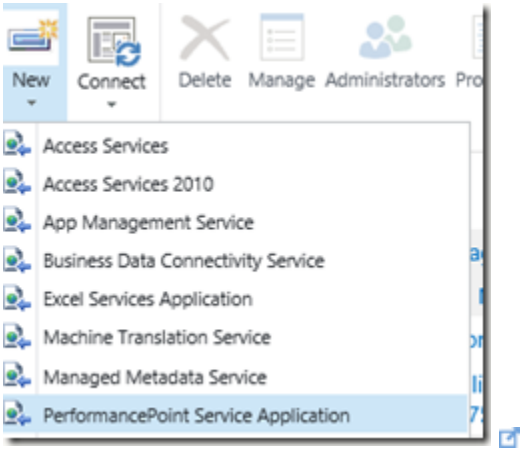


Validate and run the configuration.

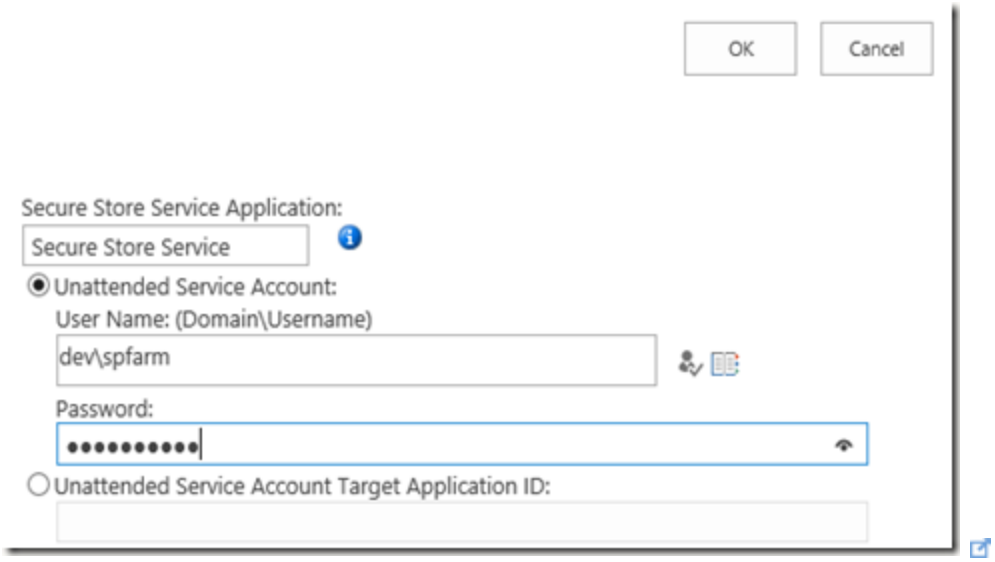


Configure Performance Point Services

Go to Central Admin and go to Manage service applications and create Performance Point Service Application.



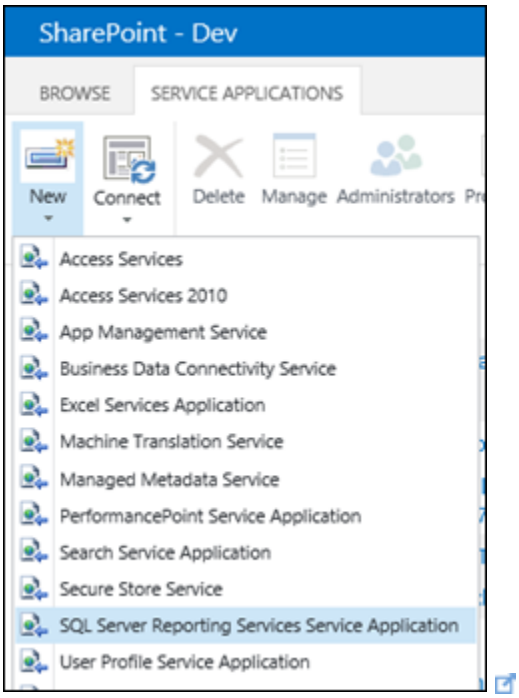
When creating the Service Application make sure you inserted the Unattended Service Account.



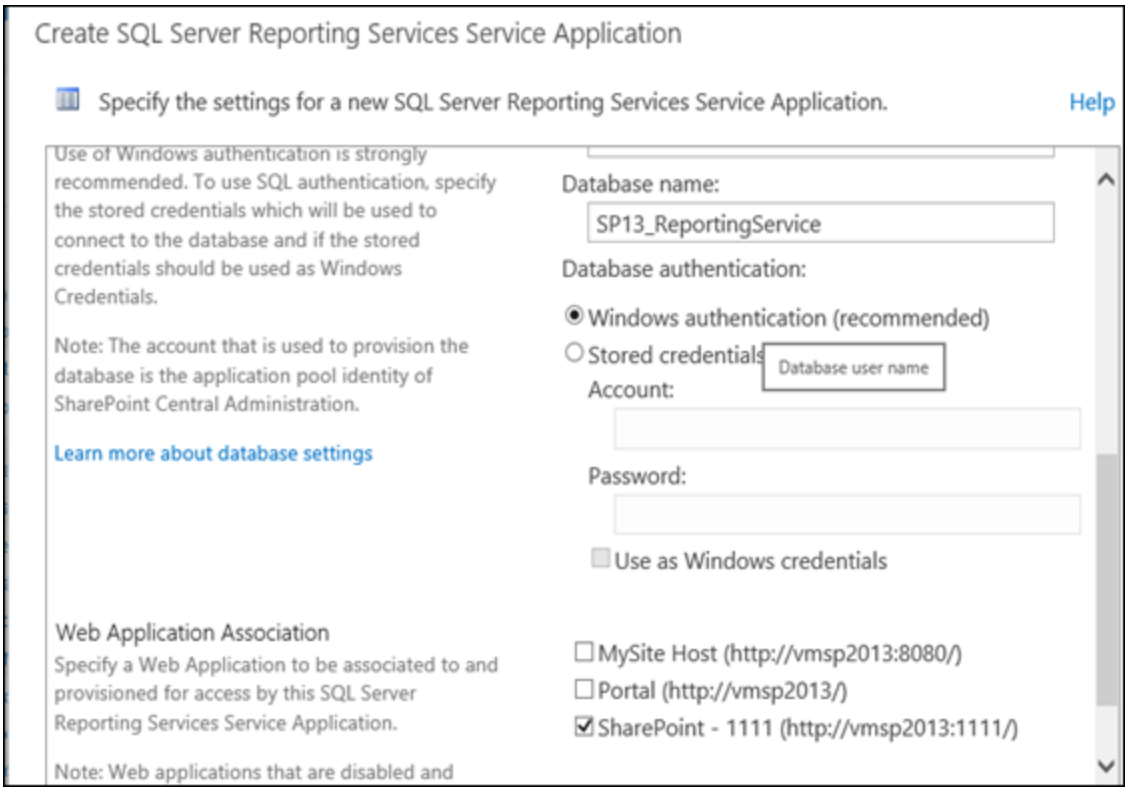
Ultimately check **PerformancePoint Service is up and running.**

Configure Reporting Services

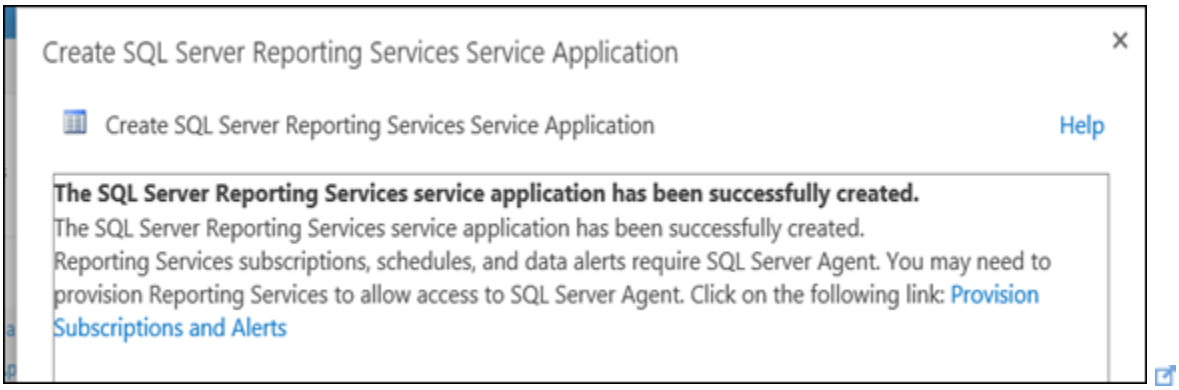
Go to Central Admin and go to Manage service applications and create Reporting Service Application.



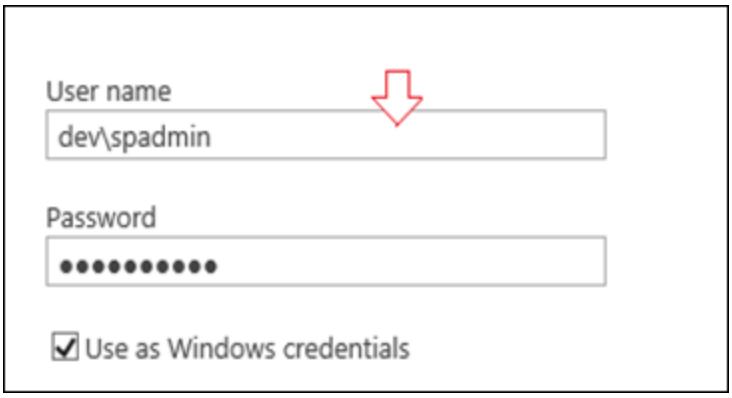
When you are creating make relevant web associations to your web applications.



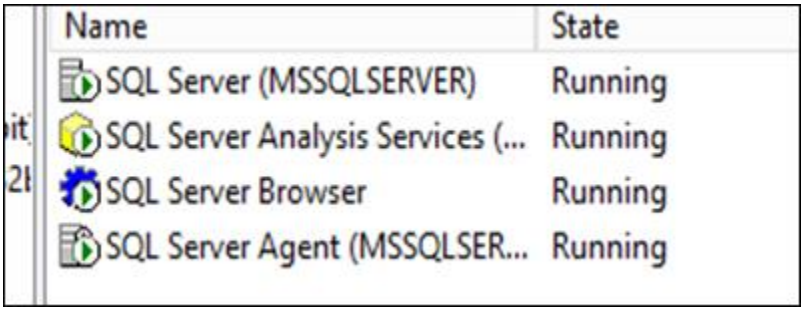
When service application created navigate to **Provision Subscriptions and Alerts**



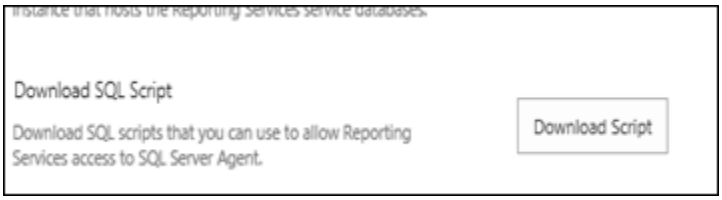
and enter spadmin account and make in as Windows Credentials. (This account should have **sysadmin** permission in the Database)



Then make sure SQL server Agent is up and running



Then download the Script and execute the SQL.



and finally make sure SQL Server Reporting Services is running.

Now You are almost over.

Now you need to go to site collection and activate features

Site Collection Features

Performance Point Site Collection Feature

PowerPivot Feature Integration for Site Collections

Site Features

PerformancePoint Site Feature

SharePoint Server Enterprise site feature

Configure Power View (Reporting service Features) on existing SharePoint 2013 Farm With BI Features (Small Farm – APP & DB)

How to create Custom Web Service WCF (REST) in SharePoint 2013

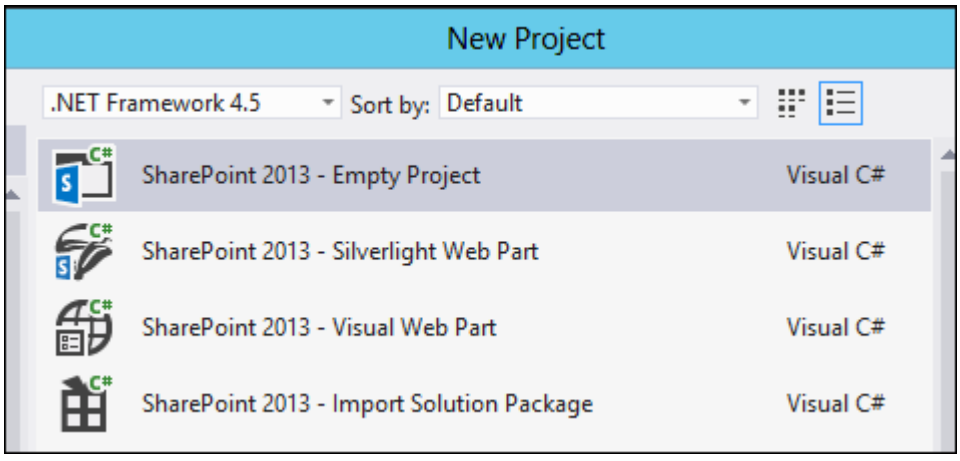
SharePoint native web services can be access via **_vti_bin** mapped path. Deploying our custom service to this mapped path enables mapped features such as

- Access service using relative path
- Ex – Assume service is custom.svc; therefore it is possible to access it
- http://server/**_vti_bin/custom.svc**
- http://server/sites/site/**_vti_bin/custom.svc**
- http://server/sites/site/web/**_vti_bin/custom.svc**
- http://server/sites/site/web/pages/**_vti_bin/custom.svc**

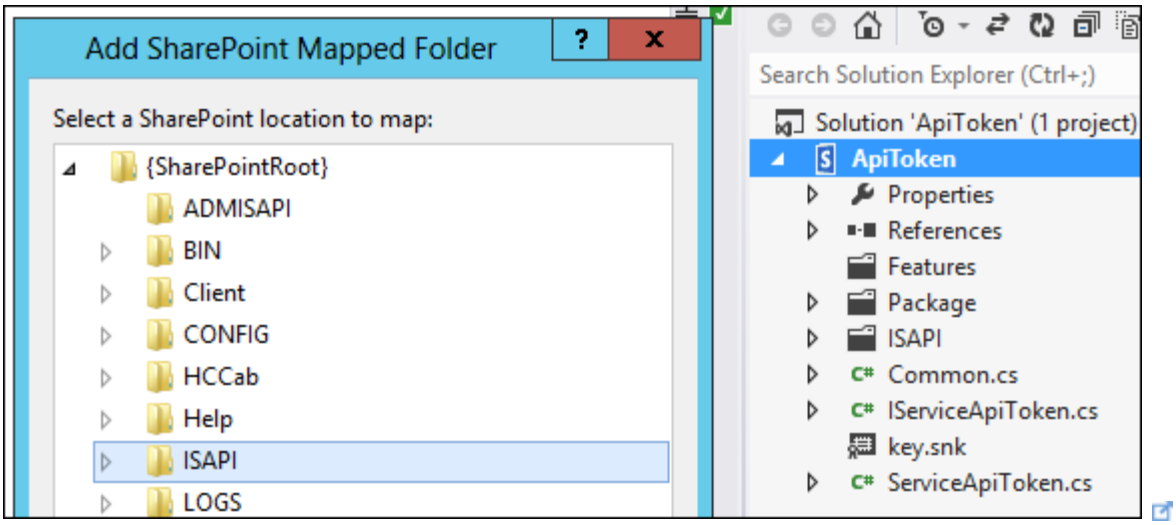
But SharePoint Project template does not contain wcf project therefore we need to do have some work around to archive this.

Step by Step guide to create a custom SharePoint REST service

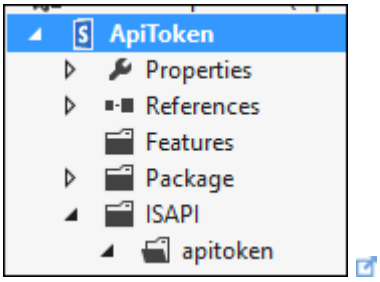
Create a **Empty SharePoint Project** by Selecting as **Farm Solution**



Then right the Project and **Add –> SharePoint Mapped Folder** –> select **ISAPI** and click ok



Then create a **sub folder under ISAPI** that is use to deploy our service.



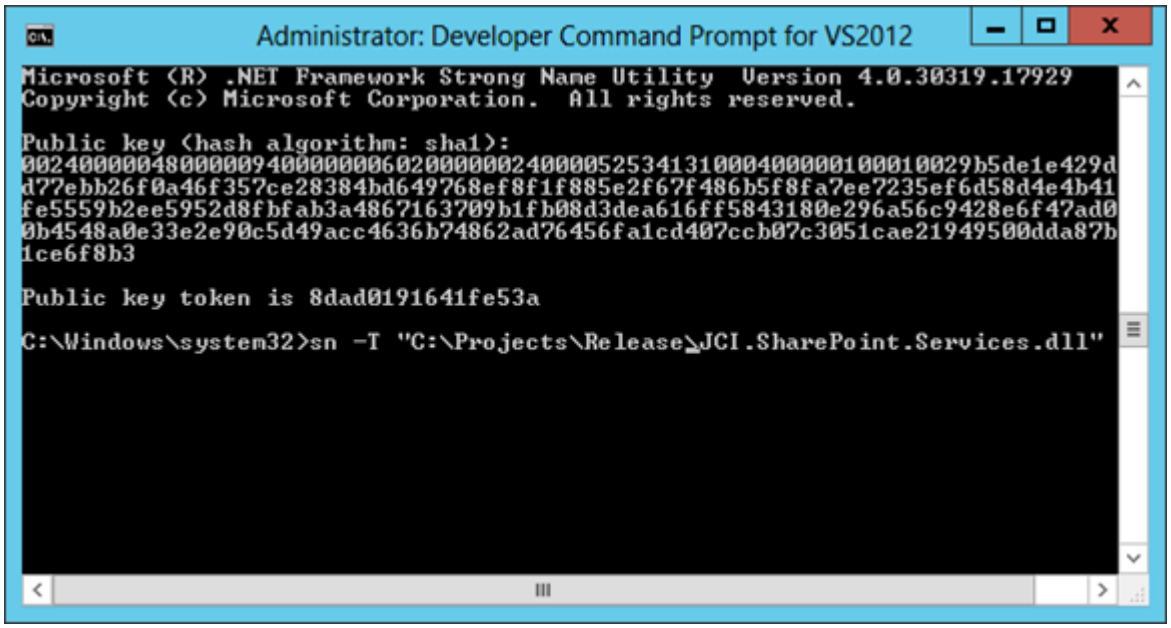
Now we need to create wcf service and it's interface. therefore add two classes named **ServiceApiToken.cs and IServiceApiToken.cs.**

```
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;

namespace ApiToken
{
    [ServiceContract]
    public interface IServiceApiToken
    {
        [OperationContract]
        [WebGet(ResponseFormat = WebMessageFormat.Json)]
        string GetResponse();
    }
};
```

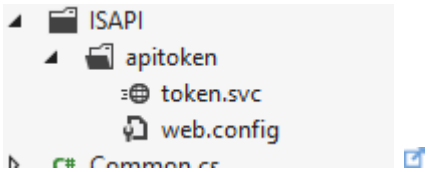
```
using System;
using System.Net;
using System.Runtime.Serialization.Json;
using System.Text;

namespace ApiToken
{
    [AspNetCompatibilityRequirements(RequirementsMode =
AspNetCompatibilityRequirementsMode.Allowed)]
    public class ServiceApiToken : IServiceApiToken
    {

        public string GetResponse()
        {
            return "Executed";
        }
    }
}
```

After implementing the Service and Service interface you need to build the project and find the public key of the assembly. (You need to find the fully qualified assembly name)

For an example you can **open visual studio command prompt and use sn –T and dll path to find the public key token.**



finally we need add service end points and web.config



you can not find svc template in SharePoint Project Template. Thus you need to add txt file and rename as a svc. **In my case i renamed it as a token.svc.**

Token.svc file you need to specify the Public token you found.

```
<%@ Assembly Name="ApiToken, Version=1.0.0.0, Culture=neutral,
PublicKeyToken=fe8eeb150d52c287"%>
<%@ ServiceHost Service="ApiToken.ServiceApiToken" %>
```

for web config you need to add txt file and rename it as a web.config

```
<configuration>
  <system.serviceModel>
    <behaviors>
      <serviceBehaviors>
        <behavior>
          <serviceMetadata httpGetEnabled="true"
httpsGetEnabled="true" />
          <serviceDebug includeExceptionDetailInFaults="false"
/>
        </behavior>
      </serviceBehaviors>
      <endpointBehaviors>
        <behavior name="customApiToken">
          <webHttp/>
        </behavior>
      </endpointBehaviors>
    </behaviors>
    <services>
      <service name="ApiToken.ServiceApiToken" >
        <endpoint address=""
behaviorConfiguration="customApiToken" binding="webHttpBinding"
contract="ApiToken.IServiceApiToken">
          <identity>
            <dns value="localhost" />
          </identity>
        </endpoint>
```

```
        <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange" />
      </service>
    </services>
  </system.serviceModel>
 </configuration>
```

After you can deploy the solution and access from _vti_bin folder.

Publish SharePoint site in Azure to the Internet

As part of testing process, you may need to publish your SharePoint site whose application server is hosted in Microsoft Azure to the Internet. This is to allow your tester or end-user to access to do pilot testing. This post is going to cover quick steps to complete the publishing.

You may need to read 3 reasons why start-up companies should use Microsoft Azure .

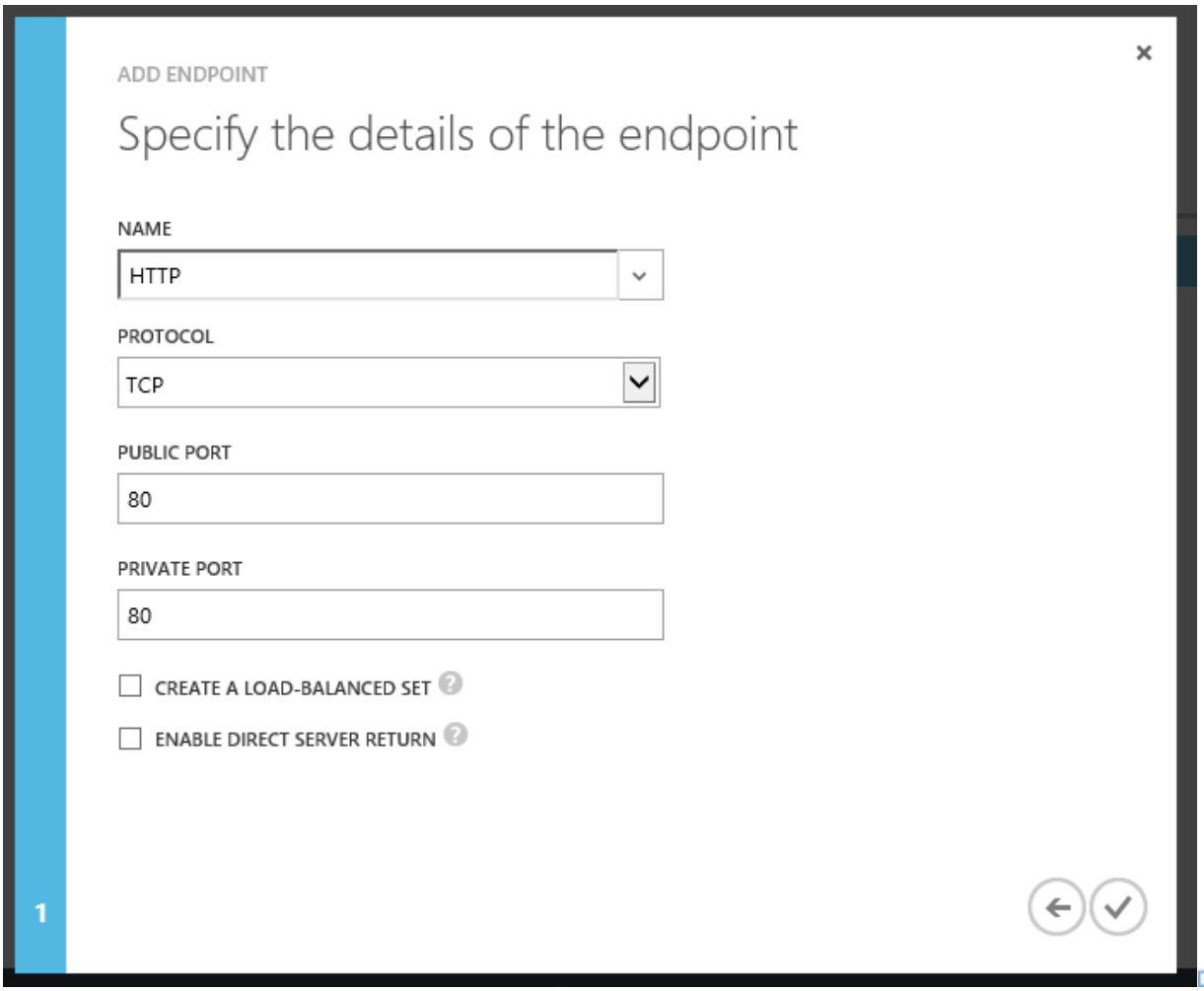
Step1: Create an HTTP endpoint

Your virtual machines in Microsoft Azure by default only communicate each other using a private network channel in the same cloud service. You cannot have it be accessible from the Internet without setting an HTTP or HTTPs endpoint for them. Microsoft Azure creates Remote Desktop endpoint that you need to remotely connect to your virtual machine. PowerShell is also created upon your demand during your setup.

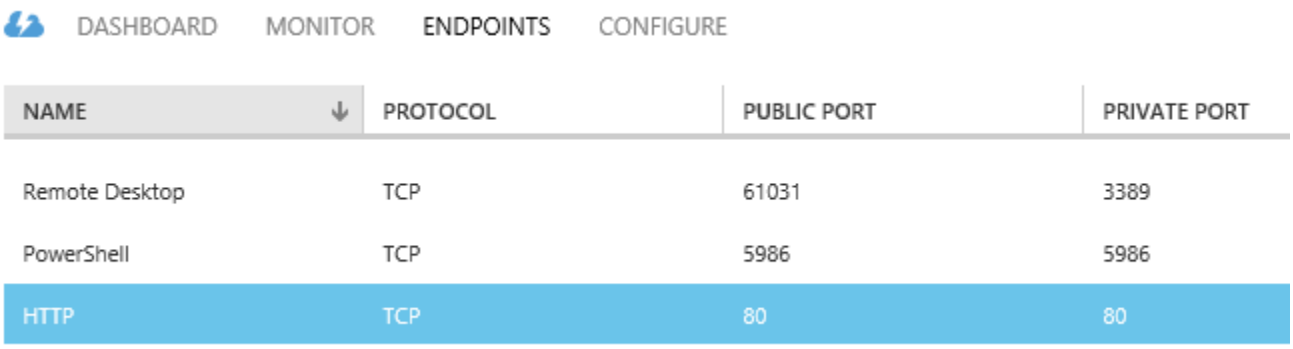
To view default endpoints associated to your virtual machine in Microsoft Azure, click ENDPOINTS.

To create a new endpoint, click Add in the bottom bar. Select Add a stand-alone endpoint in case you have not set up a Load-Balanced Set in your Microsoft Azure infrastructure. In the next windows, set up the following parameters:

- **Name**: HTTP
- **Protocol**: TCP
- **Public Port**: 80
- **Private Port**: 80 (optional)



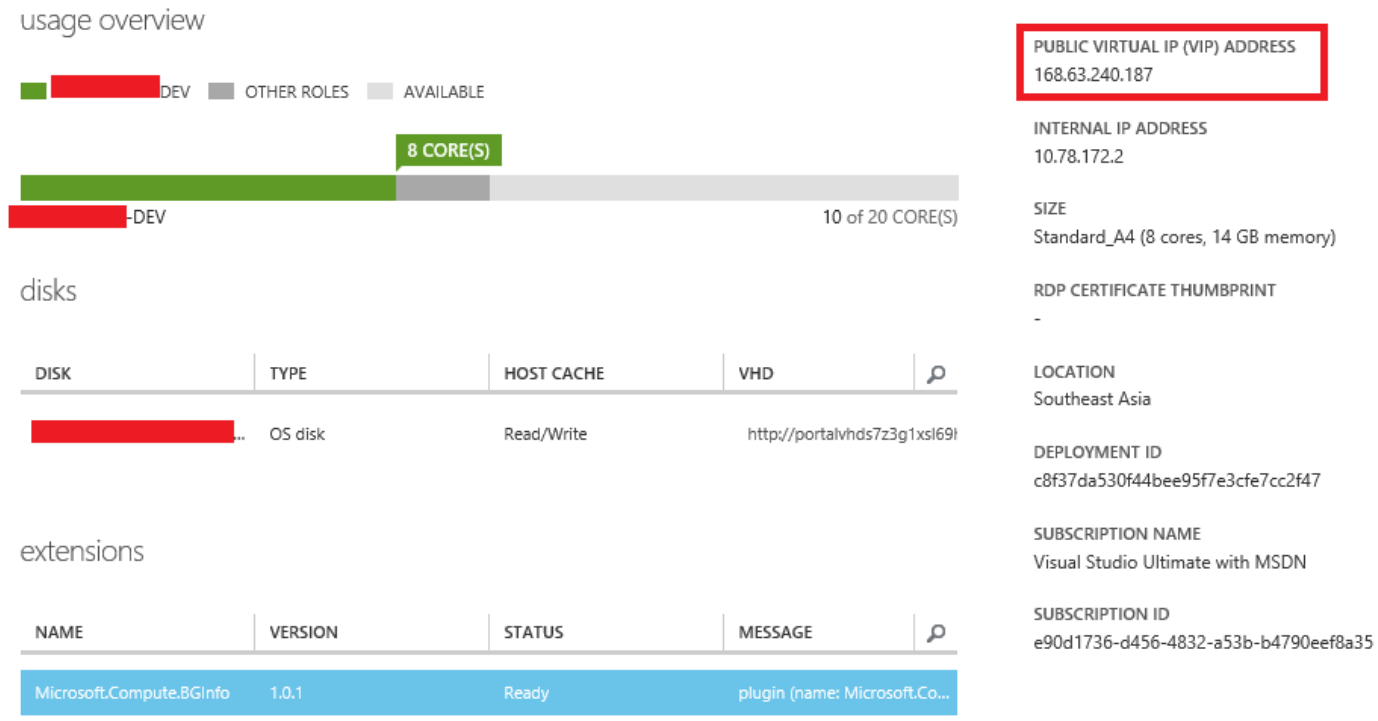
Both check-box settings are optional. If you do not intend to build high availability, skip them. Wait for Microsoft Azure a few minutes.



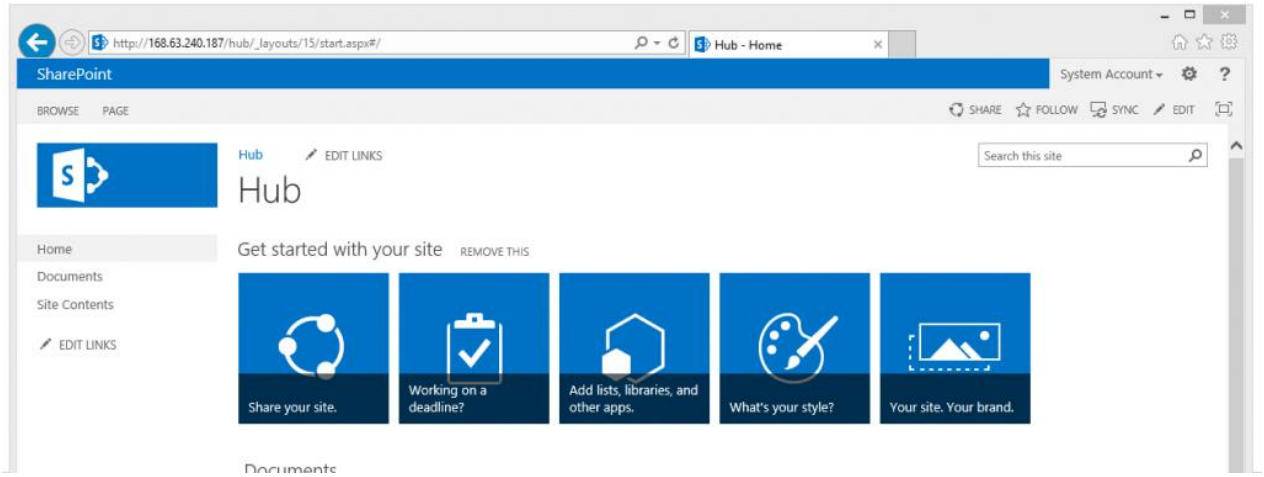
Make sure HTTP endpoint is available in the endpoint list.

Step 2: Configure AAM for SharePoint site

Now you need to configure AAM (Alternative Access Mapping) for SharePoint site. You need to add public virtual IP (VIP) address to the Internet zone in AAM setting. Find the VIP address at the left panel.



If you do not add VIP address to internet or extranet zone, you may encounter 401 Authorization errors. After AAM configuration, check your SharePoint site from the Internet.



You couldn’t create multiple HTTP endpoints without configuring host header for each SharePoint site that needs publishing.

Additional references

- How to Set Up Endpoints to a Virtual Machine
- Adding multiple endpoints to your Windows Azure Virtual Machines by using a CSV file
- Windows Azure Virtual Machine Endpoints Explained

SharePoint: Use PowerShell to find Fields using a Managed Metadata TermSet

Introduction

Ever wondered how many fields are referencing a Managed Metadata Termset? It's going to be a long and boring job using the Web UI to click through every web... and every list in every web... and every field in every list, looking for all the fields referencing a particular termset. Just writing that in a sentence was long enough!

This is the sort of job where PowerShell really shines!

The example below demonstrates creating a script (with a number of functions) to recurse through a site collection, creating a report of all the fields using a termset.

Applies To

- SharePoint 2010
- SharePoint 2013

Download

- Download the script from the TechNet Gallery: Find all SPFields that are using a Managed Metadata TermSet

Walkthrough

The basic PowerShell used to check if a Managed Metadata field is using a particular Managed Metadata Termset is:

```
$termSetId = "e07cab2f-ef85-473e-a4a7-1104b5daf192"
$field = (Get-SPWeb
"http://mdysp13").Lists["Documents"].Fields["Country"]
if($field.GetType().Name -eq "TaxonomyField"){
 if($field.TermSetId.ToString() -eq $termSetId){
  Write-Host "Houston, we have a match!" -foregroundcolor darkyellow;
 }
}
```

First the Managed Metadata Termset ID is stored in a variable (more about getting a termsets ID later). Then the field is checked to see if it's a Taxonomy Field. Finally, the fields TermSetId property is compared to the Termset ID stored, to see if they match.

The PowerShell can be extended to check a collection of fields:

```
$fieldCollection = (Get-SPWeb
"http://mdysp13").Lists["Documents"].Fields
$termSetId = "e07cab2f-ef85-473e-a4a7-1104b5daf192"
foreach($field in $fieldCollection)
{
 if($field.GetType().Name -ne "TaxonomyField"){
  continue;
 }
 if($field.TermSetId.ToString() -ne $termSetId){
  continue;
 }
 #if we get to here, we have a match!
}
```

To get the Taxonomy TermSet ID, you can use the Get-SPTaxonomySession cmdlet for the current site, and pipe it to Format-Table (FT), to list all of the Termsets.

```
$w = Get-SPWeb "http://mdysp13";
$tsession = Get-SPTaxonomySession -Site $w.Site;
$tsession.GetTermSets("Countries",1033) | FT Name,ID
#Or, if you want to get a term set based on the SPWeb's default
language ID
$tsession.GetTermSets("Countries",$w.Language) | FT Name,ID
```

Pretty cool huh?

If you want to get a list of all the termsets, then you can write a simple function to return all the termsets as a list.

```
function List-AllTermSets{
 [CmdletBinding()]
  Param(
    [parameter(Mandatory=$true,
ValueFromPipeline=$true)][Microsoft.SharePoint.SPWeb]$web
   )
 $termSetInfo = New-Object psobject
 $termSetInfo | Add-Member -MemberType NoteProperty -Name "Store" -
value ""
 $termSetInfo | Add-Member -MemberType NoteProperty -Name "StoreId" -
value ""
 $termSetInfo | Add-Member -MemberType NoteProperty -Name "Group" -
value ""
 $termSetInfo | Add-Member -MemberType NoteProperty -Name "GroupId" -
value ""
 $termSetInfo | Add-Member -MemberType NoteProperty -Name "TermSet" -
value ""
 $termSetInfo | Add-Member -MemberType NoteProperty -Name "TermSetId"
-value ""

 $tsession = Get-SPTaxonomySession -Site $web.Site;
 $tstores =  $tsession.TermStores;
 $list = @();
 foreach($tstore in $tstores)
 {
  $tgroups = $tstore.Groups;
  foreach($tgroup in $tgroups)
```

```
  {
   $tsets = $tgroup.TermSets;
   foreach($tset in $tsets)
   {
    $tinfo = $null;
    $tinfo = $termSetInfo | Select-Object *;
    $tinfo.Store = $tstore.Name;
    $tinfo.StoreId = $tstore.ID;
    $tinfo.Group = $tgroup.Name;
    $tinfo.GroupId = $tgroup.ID;
    $tinfo.TermSet = $tSet.Name;
    $tinfo.TermSetId = $tSet.ID;
    $list += $tinfo;
   }
  }
 }
 return $list;
}
```

So, what if you wanted all of this scripted? A function you can call that generates a report. Well, prepare to roll up your sleeves and poise your fingers over the Ctrl+C key combo!

We need a couple of functions for this, performing the following tasks;

1. A function to get a list of all the taxonomy (managed metadata) fields in a field collection referencing a termset
2. A function to call which will report on all the taxonomy (managed metadata) fields in the web, the webs lists, and the webs sub webs, that are referencing a given termset.

Each function is outlined below.

Note: If you'd rather download the script, download it from the Microsoft TechNet Gallery here: Find all SPFields that are using a Managed Metadata TermSet

1. Get a list of all the fields (in a field collection) using a termset

```
function Get-FieldsUsingTermSet
{
 [CmdletBinding()]
  Param(
```

```
    [parameter(Mandatory=$true, ValueFromPipeline=$true,
Position=1)][Microsoft.SharePoint.SPFieldCollection]$fieldCollection,
    [parameter(Mandatory=$true,
Position=2)][Microsoft.SharePoint.Taxonomy.TermSet]$TermSet
   )
 $MetadataField = New-Object psobject
 $MetadataField | Add-Member -MemberType NoteProperty -Name
"ParentListUrl" -value ""
 $MetadataField | Add-Member -MemberType NoteProperty -Name
"ParentListTitle" -value ""
 $MetadataField | Add-Member -MemberType NoteProperty -Name
"FieldTitle" -value ""
 $MetadataField | Add-Member -MemberType NoteProperty -Name "FieldId"
-value ""

 $matches = @();
 foreach($field in $fieldCollection)
 {
  if($field.GetType().Name -ne "TaxonomyField"){
   continue;
  }
  if($field.TermSetId.ToString() -ne
$TermSet.Id.ToString()){continue;}
  $tf = $MetadataField | Select-Object *;
  $tf.ParentListUrl = $field.ParentList.ParentWeb.Url;
  $tf.ParentListTitle = $field.ParentList.Title;
  $tf.FieldTitle = $field.Title;
  $tf.FieldId = $field.ID;
  $matches += $tf;
 }
 return $matches;
}
```

2. A parent function to bring it together, that will give you some options (like recursively checking the web, searching just web level fields)

```
function Get-ManagedMetadataFieldUses
{
 [CmdletBinding()]
  Param(
    [parameter(Mandatory=$true, ValueFromPipeline=$true,
Position=1)][Microsoft.SharePoint.SPWeb]$web,
    [parameter(Mandatory=$true,
Position=2)][Microsoft.SharePoint.Taxonomy.TermSet]$TermSet,
```

```
    [parameter(Mandatory=$false, Position=4)][switch]$Recurse,
    [parameter(Mandatory=$false,
Position=5)][switch]$WebLevelFieldsOnly
   )

 $matches = @();
 $matches += Get-FieldsUsingTermSet $web.Fields $TermSet;

 if($WebLevelFieldsOnly -eq $false)
 {
  foreach($list in $web.Lists)
  {
   $matches += Get-FieldsUsingTermSet $list.Fields $TermSet
  }
 }

 if($Recurse)
 {
  foreach($subweb in $web.Webs)
  {
   $matches += Get-ManagedMetadataFieldUses $subweb $TermSet $Recurse
$WebLevelFieldsOnly;
  }
 }

 return $matches
}
```

Examples of using the script to create some reports.

1. Download the script from here: Find all SPFields that are using a Managed Metadata TermSet
2. Save the script somewhere. "C:\Temp" is a good place!
3. If you haven't already, set the PowerShell execution policy to Bypass (this will allow you to import all PowerShell scripts, but be aware of the security implications)

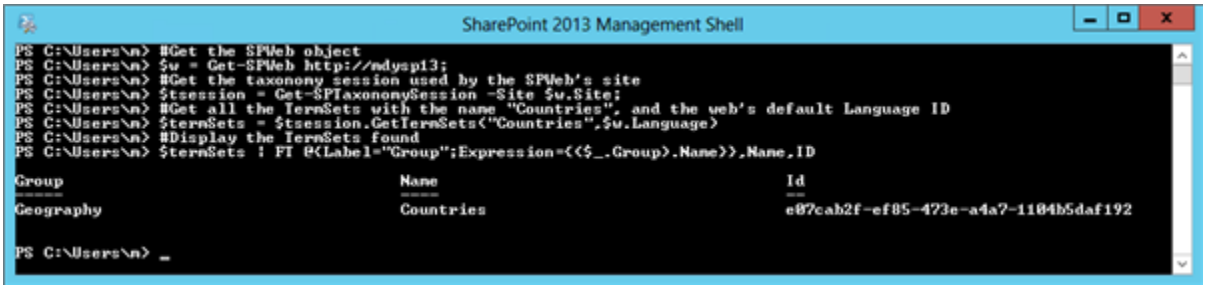
```
Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope CurrentUser
```

4. Import the script into PowerShell.

```
Import-Module C:\Temp\Get-ManagedMetadataFieldUses.ps1
```

5. Run a few commands to get a termset to report on. In this example, we get a termset called "Countries"

```
#Get the SPWeb object
$w = Get-SPWeb http://mdysp13;
#Get the taxonomy session used by the SPWeb's site
$tsession = Get-SPTaxonomySession -Site $w.Site;
#Get all the TermSets with the name "Countries", and the web's default
Language ID
$termSets = $tsession.GetTermSets("Countries",$w.Language)
#Display the TermSets found
$termSets | FT @{Label="Group";Expression={($_.Group).Name}},Name,ID
#Select the first TermSet
$termSet = $termSets[0]
```



6. Call the Get-ManagedMetadataFieldUses function, and store the results in the $matchingFields variable.

```
$matchingFields = Get-ManagedMetadataFieldUses -web $w -TermSet
$termSet -Recurse
```

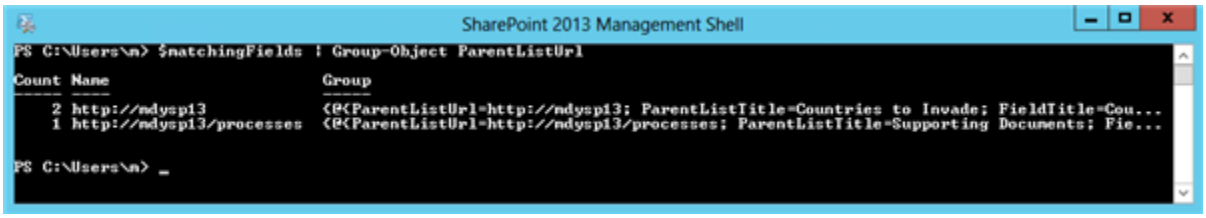
Now do some reporting!!

Display all of the results in the raw format.

```
$matchingFields | FT
```

Display all of the results, grouping them by the Site. This view of the data will show you how many fields in each site (or web) are referencing the termset)

```
$matchingFields | Group-Object ParentListUrl
```



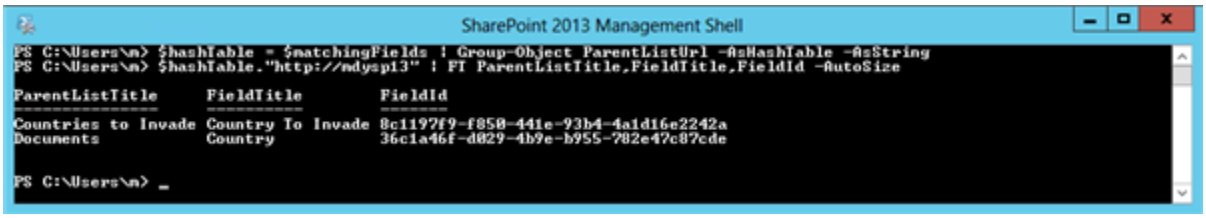
This improves on the previous command, displaying all of the results, grouping them by the Site. In this view, all the fields are listed, grouped under the site they belong to.

```
$matchingFields | Group-Object ParentListUrl | Select -ExpandProperty Group | Format-Table -GroupBy ParentListUrl
```

Group the objects into a Hash Table. This will allow you to directly reference a web URL, to a get a list of fields in that web that reference the termset.

```
$hashTable = $matchingFields | Group-Object ParentListUrl -AsHashTable -AsString
```

```
$hashTable."http://mdysp13" | FT ParentListTitle,FieldTitle,FieldId -
AutoSize
```



Finally, you can export your results to a CSV file for further analysis!

```
$matchingFields | Export-CSV  -Path C:\temp\fieldreport.csv -
NoTypeInformation -Delimiter "`t"
```

Download the full script from the Microsoft TechNet Gallery here: Find all SPFields that are using a Managed Metadata TermSet

References

- Original content from Matthew Yarlett's blog, Get a List of Fields in a Site Collection that are using a Managed Metadata TermSet

SharePoint 2013: Integrate Yammer with SharePoint for Social Feature(s)

Yammer?

Yammer is a social communication platform for organization. Employees can communicate with each other to share valuable information; they can get in touch with the right group within the organization to get right information, to get their issues resolved. Yammer is social site like other social sites: Facebook, etc. but this is a private social network for the organization only. It is a social network for a company, where employees can post messages, respond to a message, like a message and share the message. Employees who belong to the same department can create groups in yammer; and can start the conversation and sharing information. Yammer introduces a real time work in organization. And so many organizations use Yammer as their social platform today. Sometimes they also want yammer integrated with their application; and this article talks about how to integrate Yammer with SharePoint.

Yammer Integration with SharePoint

Everybody knows that Yammer will be integrated with the SharePoint 2013. There is couple of phases in which integration will be done: Basic Integration and Deep Integration.

In **Phase 1,** they are just planning to replace the NewsFeed with the Yammer feed. So for this you can either use the Web Part (which was provided for SP2010) or embed code provided by yammer. Also App for SP 2013 has been released; so we can have this option.

In **Phase 2**, we will have single sign on and more features. In this article we will see how to use Yammer Web Part to replace the NewsFeed in SharePoint 2013, how you can use yammer web part to display search results from yammer, how you can see feeds of particular user, etc. Before continuing with this article please make sure Yammer web part is installed / activated in the web application, if not then contact your administrator.

Integration with Yammer app for SharePoint

Yes this documentation on TechNet: http:// aka.ms/yamapp Till now we see how to replace the Newsfeed of SharePoint or how to configure Yammer web part for two way communication in web part. But what about starting conversation around a particular list item or document or commenting on the list item or document? What about liking and following SharePoint list item, document on Yammer or with the help of Yammer? Yes we can do this using the Yammer Embed Code Commenting. To learn more about Yammer Embed, click here . Before starting using the embed code commenting you should understand how it works.

Integration with Yammer embed code

So once you put the embedded code on a page (here for case study we will put embedded code on the DispForm.aspx or display page of list) , it renders the Yammer UI which displays conversation and provides the controls to comments and for other social things. Initially it is empty as no message has been posted, but as soon as the first message get posted from that page from the UI which is generated by embed code, the Object gets created in Yammer. And all conversation and social activities on this page (DispForm.aspx) happens around this Yammer Object only. This Object is created uniquely and remembered using the actual URL on which Embed code is placed (here in our case it is DispForm.aspx page). In SharePoint the list or document library has single display form page but it has different URL for different list items or documents. And we can use these URLs and start commenting on the particular list item or document.

Steps to configure it are very simple, let's see them one by one:

1. Create one html file using Visual studio.
2. Put the following code inside it and save it:

```
yam.connect.embedFeed({
container: "#embedded-feed",
    network: "Network Name",
```

```
    feedType: "open-graph",
    objectProperties: {
        type: "page",
},
    private: true,
});
<div id="embedded-feed"></div>
```

Here you need to put the name of your network. And choose the appropriate type.

3. Upload this html file to the SharePoint library and remember the address of file which will require afterwards.

4. Go to the DispForm.aspx (choosing any document or list item).

5. Start editing the page. You can also choose the SharePoint designer to edit this page.

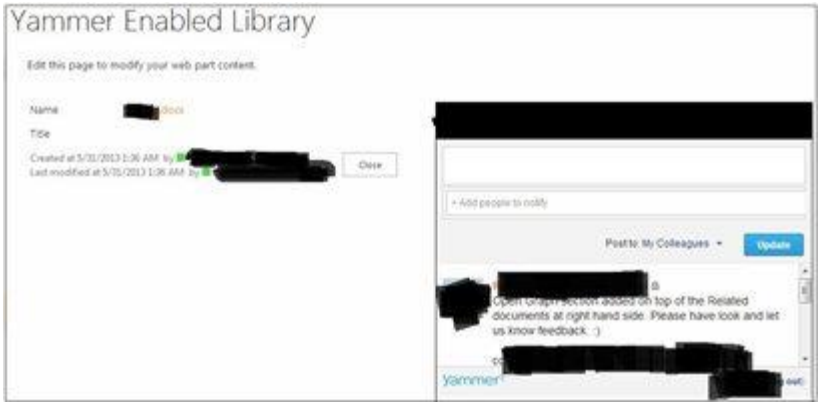
6. Add the Content Editor Web part in the web part zone.

7. Point content editor web part to the html file using the URL of file by editing web part.

8. Save the web part page settings.

9. Save the page.

After following these steps you can start commenting on each list item or document in yammer separately as shown in below screenshot.



Same way you can use Like and Follow's embed code. You can use this to replace OOB Comment web part provided by SharePoint 2013. You can use this for list, document library, task list, calendar, blogs, etc. In this way without so much coding you can start Yamming on your SharePoint environment.

See Also

An important place to find a huge amount of SharePoint related articles is the TechNet Wiki itself. The best entry point is SharePoint Resources on the TechNet Wiki

Form based Authentication (FBA) in SharePoint 2013

What is Form based Authentication (FBA)?

Form Based Authentication (FBA) provides your own authentication method using a web form. More and more companies are using FBA as a way of extending a site for non-Active Directory (AD) users.

SharePoint standard installation uses as default AD to query the Domain Controller and to check user credentials through Windows Authentication. FBA uses a custom database created separate from AD to store user credentials. Authentication using FBA is executed by a SQL DB query. When FBA is used to extend SharePoint sites, external users (non AD users) have access to SharePoint.

Why would you allow access to external users?

A practical reason for extending a SharePoint site through FBA is collaboration on documents with your clients or vendors.

How to achieve FBA in SharePoint 2013?

There are lot of guide are there in internet to explain farm based authentication in SharePoint 2013. But here I hope simple steps through which we can easily achieve the Farm Based Authentication in SharePoint 2013.

Below are the 13 simple steps to achieve FBA.

Step 1. Create **aspnetdb** using **aspnet_regsql**

Step 2. Assign SharePoint administrator as a **db_owner** in **aspnetdb** using SQL Management Studio

Step 3. Create **connection string** in IIS Global

Step 4. Add Providers in **.Net Roles** and **.Net User .**

Step 5. Do step 4 in **SecurityTokenServiceApplication**

Step 6. Create Web Application with claim based authentication

Step 7. Create Site Collection with windows authentication

Step 8. Modification in **webConfig** file in Web Application & Central Admin

Step 9. Change your web application to FBA

Step 10. Select your web application in IIS and add .net Users, if error change the default provider in .net Roles and .net Users.

Step 11. Add the .net user in IIS.

Step 12. Open the web application with windows authentication and share your site with FBA users.

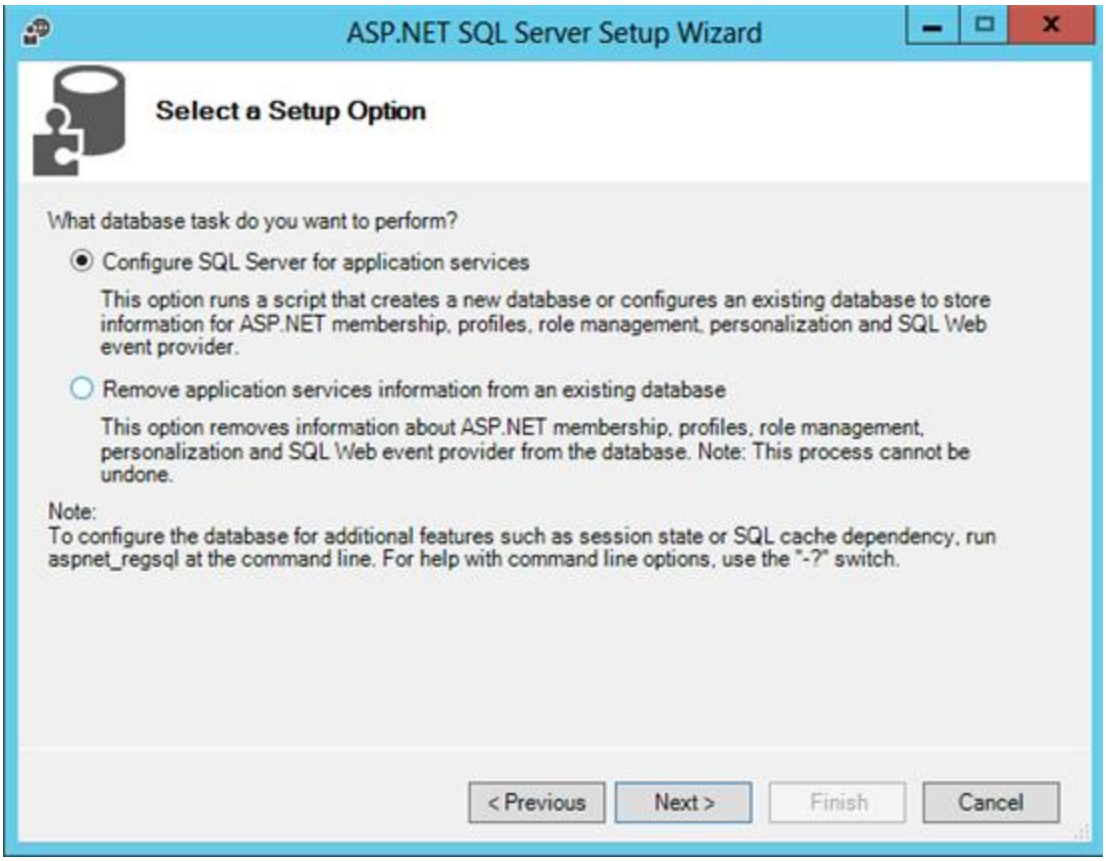
Step 13. Now login with FBA account.

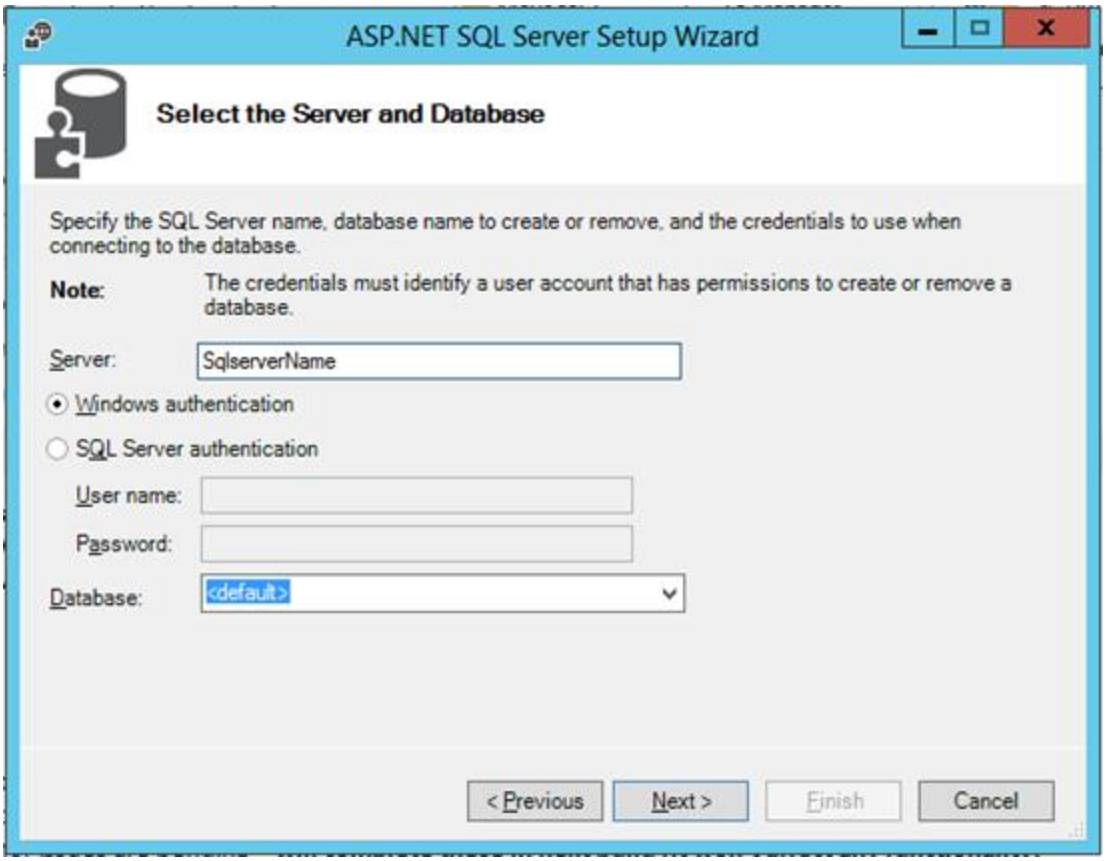
Step 1. Create aspnetdb using aspnet_regsql in Command Prompt

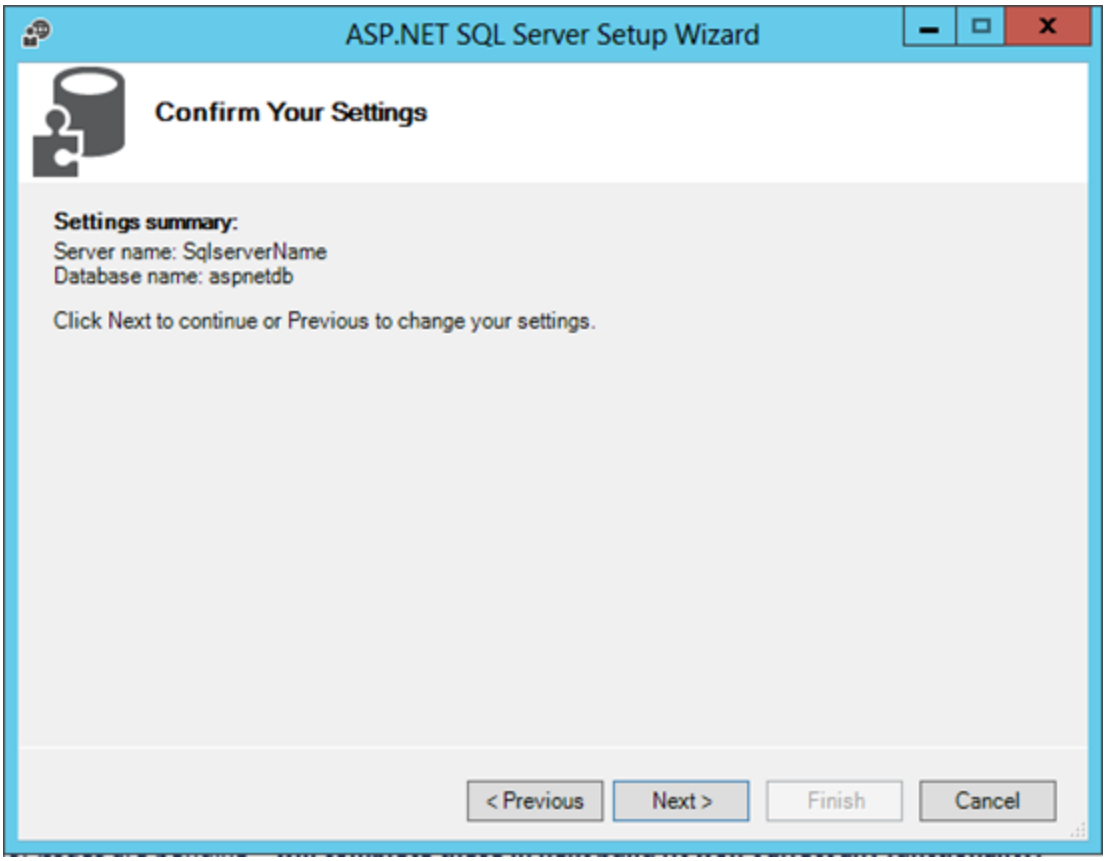
Open command prompt as run as administrator. Then move the directory to **C:Windows\Microsoft.NET\Framework64\v4.0.30319 ,** Type aspnet_reqsql and click enter.

Then the following window will open. Follow the steps one by one.


ASP.NET SQL Server Setup Wizard
Welcome to the ASP.NET SQL Server Setup Wizard
This wizard creates or configures a SQL Server database that stores information for ASP.NET applications services (membership, profiles, role management, personalization and SQL Web event provider).
To configure the database for these features individually or for additional features such as session state or SQL cache dependency, run aspnet_regsql at the command line. For help with command line options, use the "-?" switch.
Click Next to continue.
< Previous
Next >
Finish
Cancel


ASP.NET SQL Server Setup Wizard
Select a Setup Option
What database task do you want to perform?
Configure SQL Server for application services
This option runs a script that creates a new database or configures an existing database to store information for ASP.NET membership, profiles, role management, personalization and SQL Web event provider.
Remove application services information from an existing database
This option removes information about ASP.NET membership, profiles, role management, personalization and SQL Web event provider from the database. Note: This process cannot be undone.
Note:
To configure the database for additional features such as session state or SQL cache dependency, run aspnet_regsql at the command line. For help with command line options, use the "-?" switch.
< Previous
Next >
Finish
Cancel

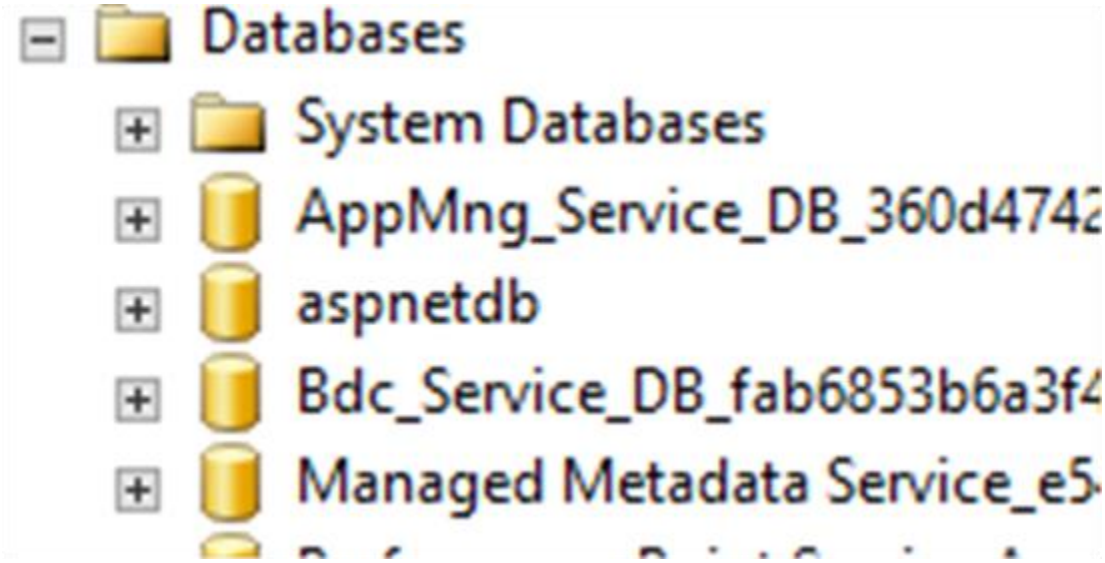

ASP.NET SQL Server Setup Wizard
Select the Server and Database
Specify the SQL Server name, database name to create or remove, and the credentials to use when connecting to the database.
Note:
The credentials must identify a user account that has permissions to create or remove a database.
Server:
SqlserverName
Windows authentication
SQL Server authentication
User name:
Password:
Database:
<default>
< Previous
Next >
Finish
Cancel



Then Click Finish. Now Database has been created in sql server

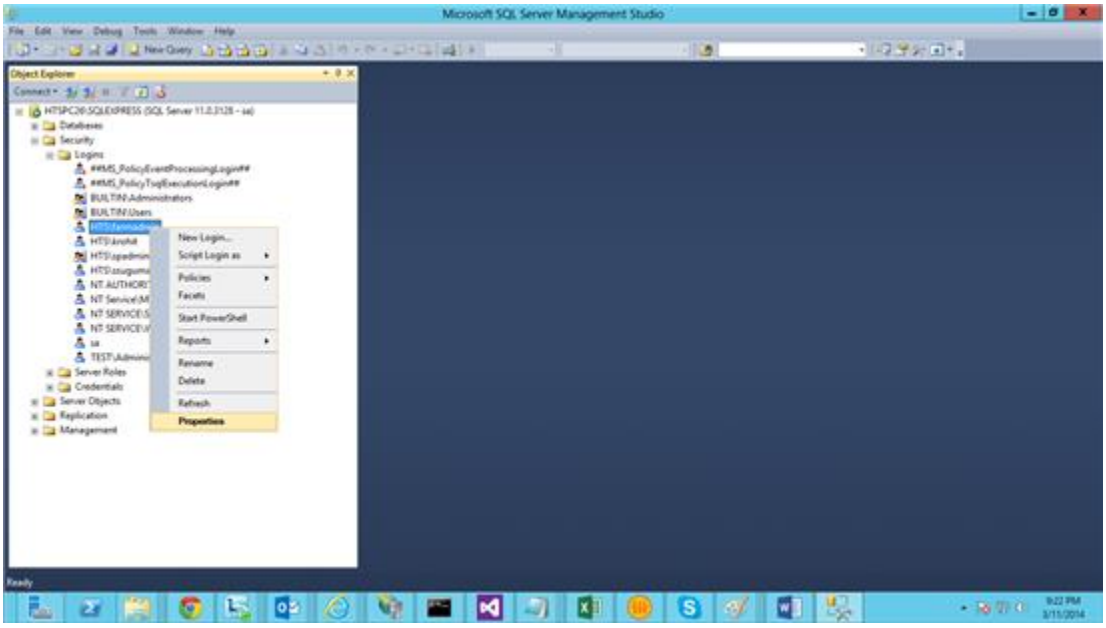
Step 2. Assign SharePoint administrator as a db_owner in aspnetdb using SQL ManagementStudio

Open Sql Server Management Studio and check for **aspnetdb** is available under database



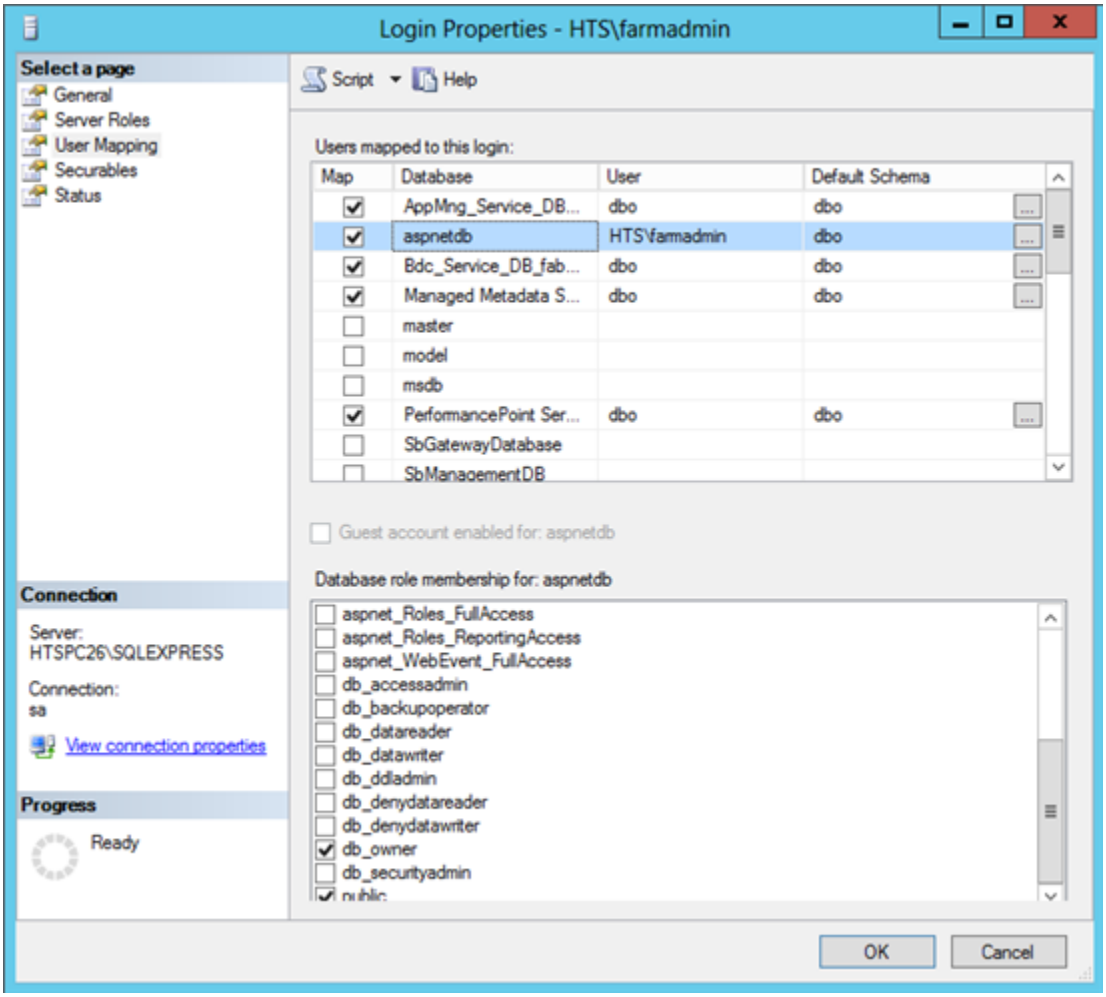
We need to give access to farmadmin to access this database. Below are the steps to give dbowner rights to farmadmin.

Expand security in left panel, and the expand Logins, check for farmadmin name in logins. If not add it. Else right click on the user name and open the Properties.



Properties Pop up will open, in that select **User Mapping** and select the DB name in right Panel. In database roles membership select the tick mark for db_owner &

Click ok.

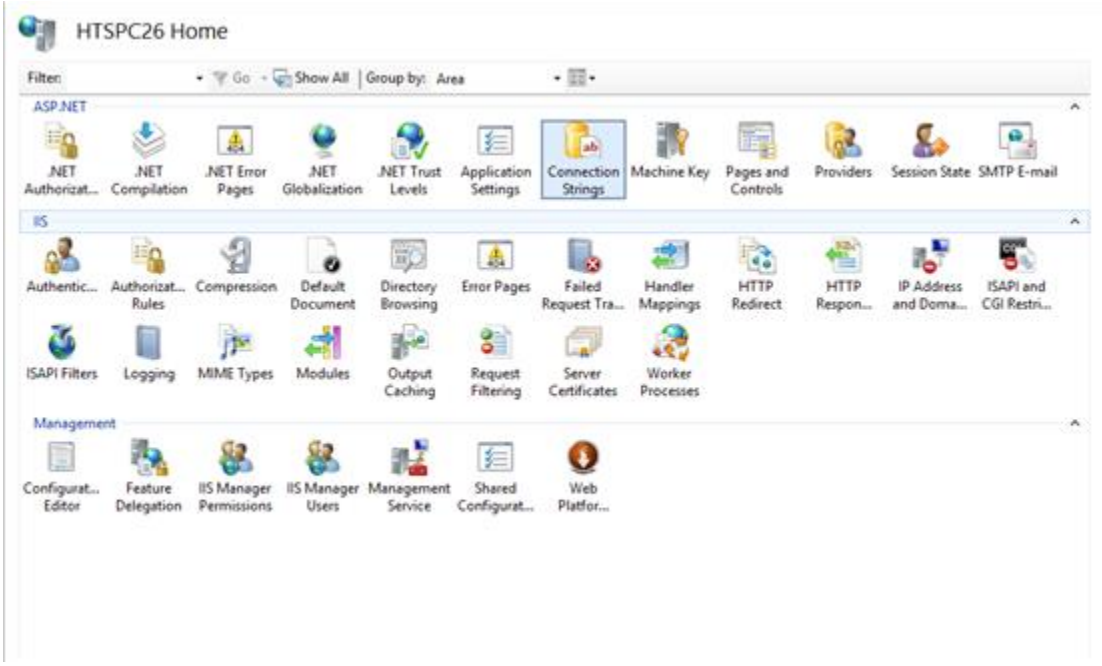


Now step 2 completed. The farm admin has access to **aspnetdb.**

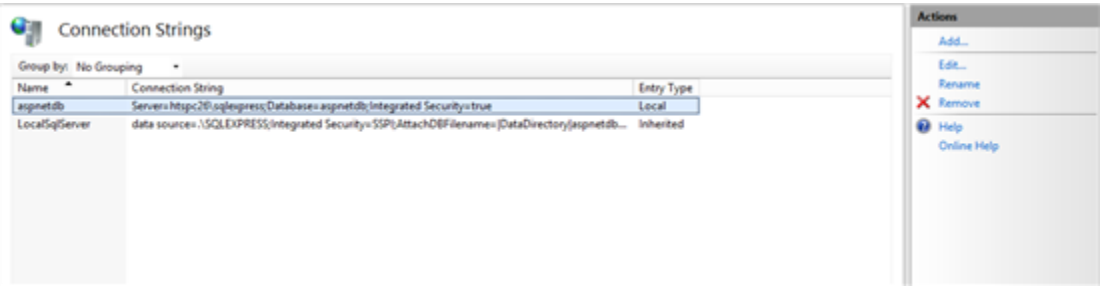
Step 3. Create connection string in IIS Global

We need to add connection string in our SharePoint application to access the created DB

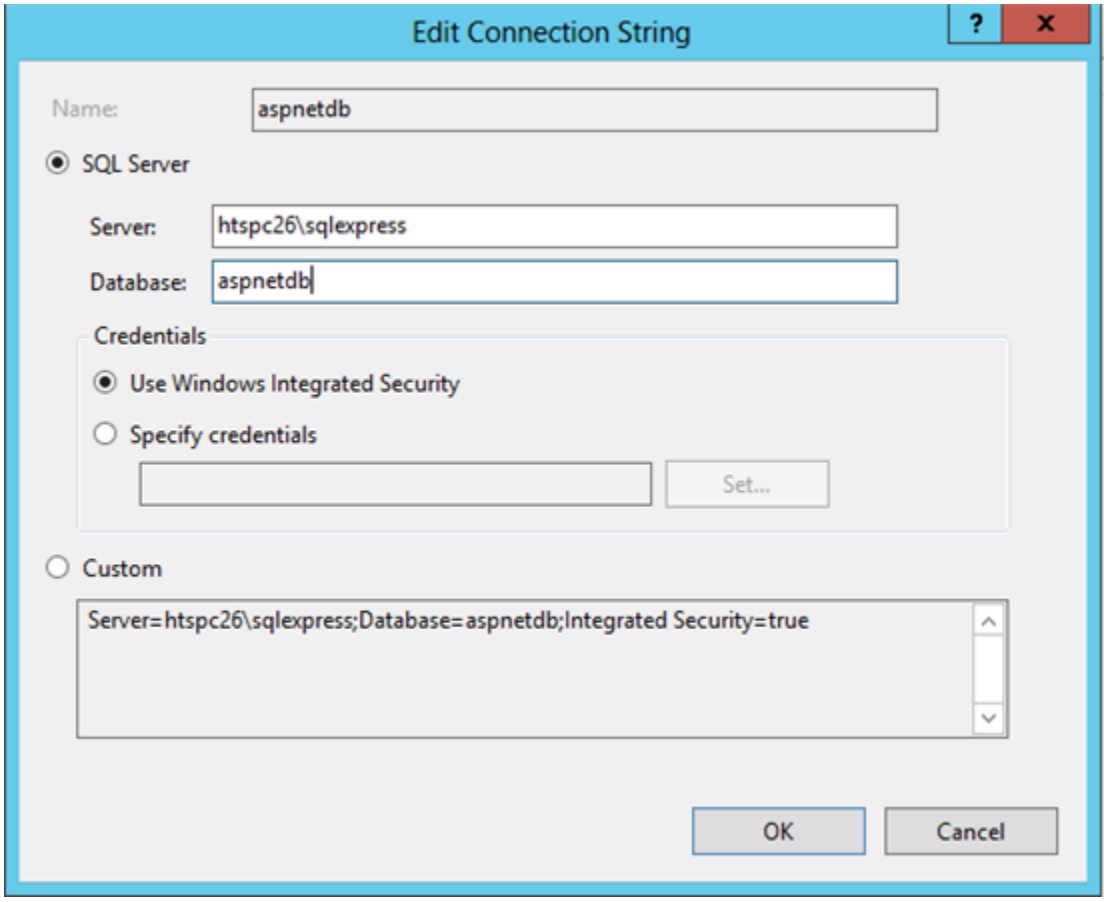
Open IIS in your server, and click o connection strings in home page.



Click add button at right side.

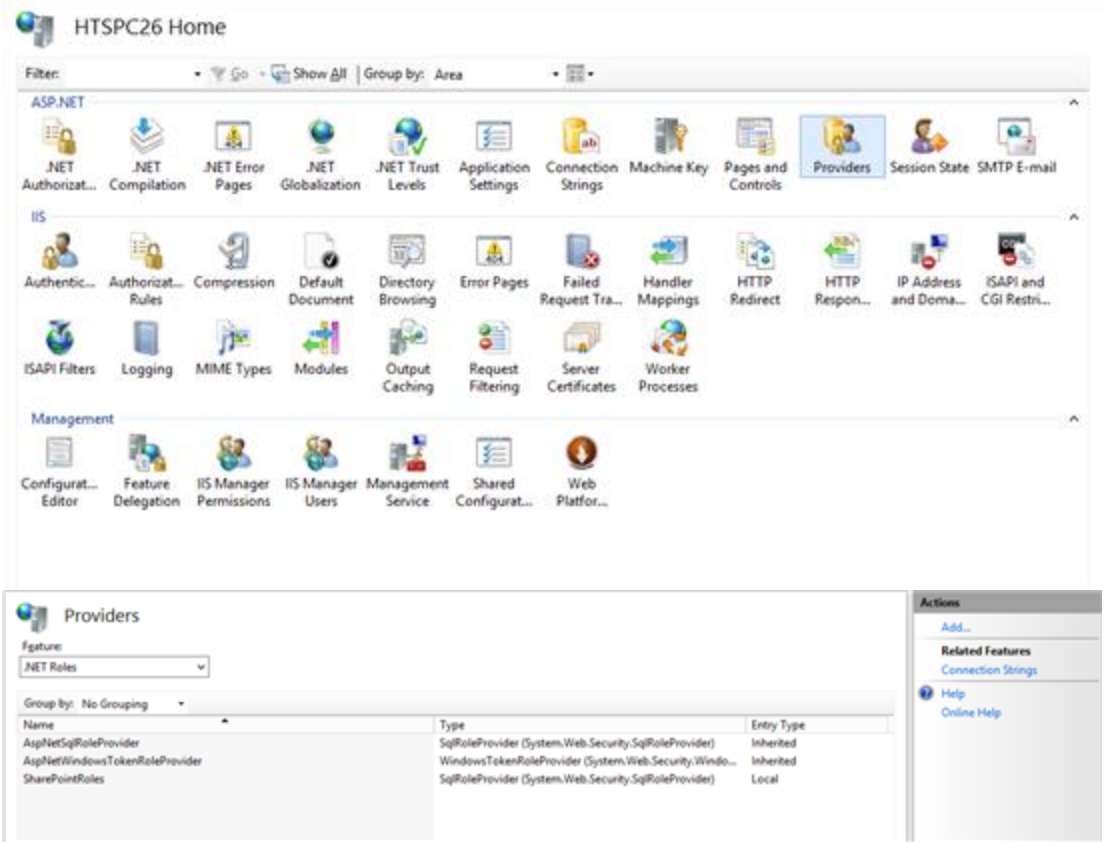


Below pop up will come for adding new connection string. Give Connections string name, sql server name, database name correctly as like below fig.



Step 4. Add Providers in .Net Roles and .Net User.

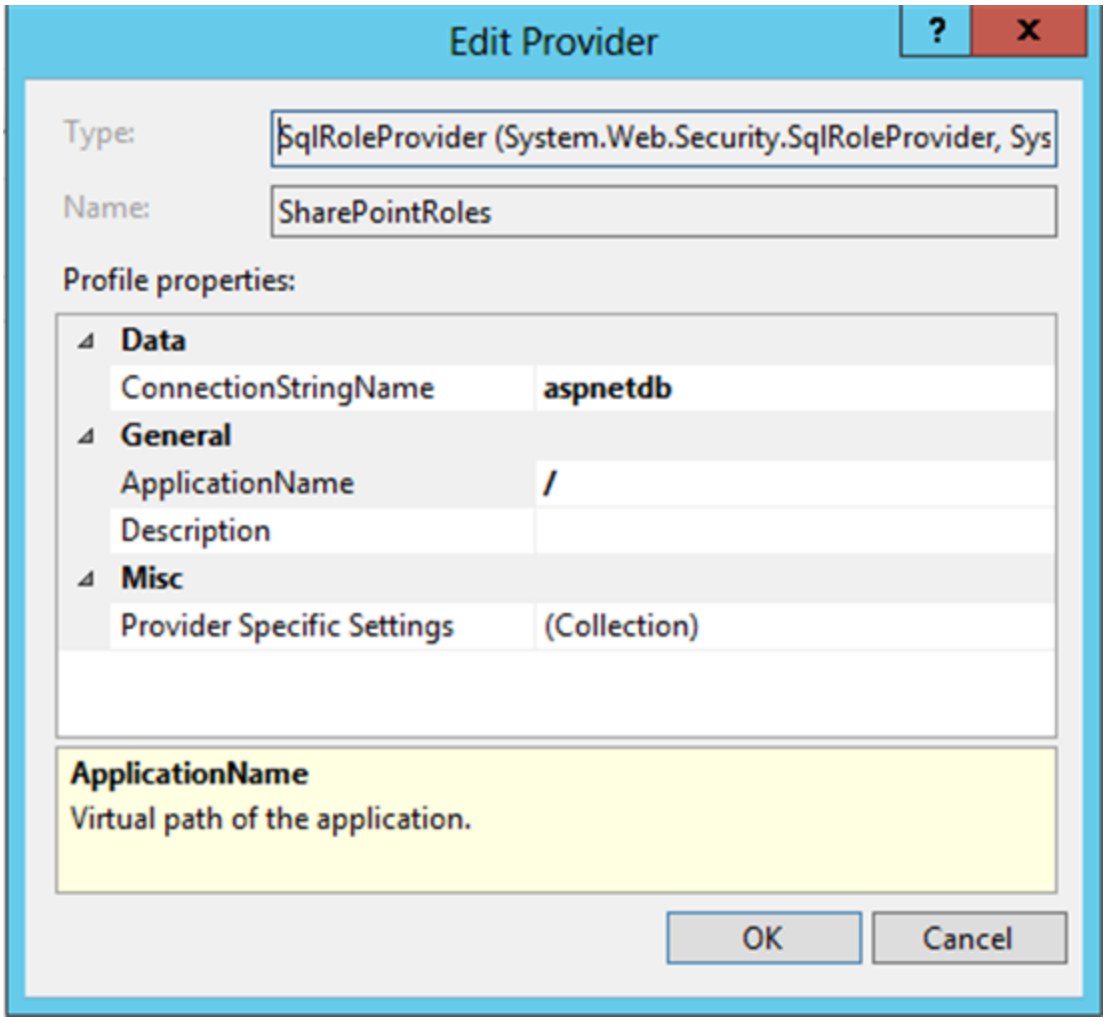
Now add roles and membership in providers. Click providers in home page.



Select .Net Roles in Features drop down and click add to add new roles.

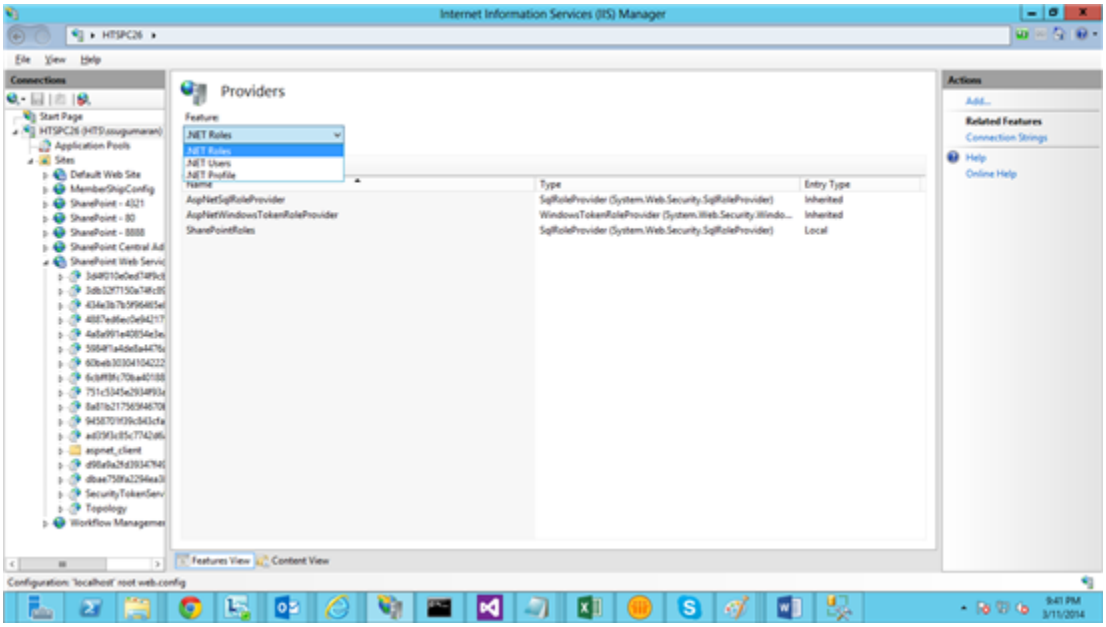
Select SqlRoleProvider as the type, and give role name and make a note on that **Role Name.** Because we need that name at a time of changing our web application to FBA.

Select the connection string the we going to use. And type**"/"** in application control to affect fab in Whole web application and click ok.

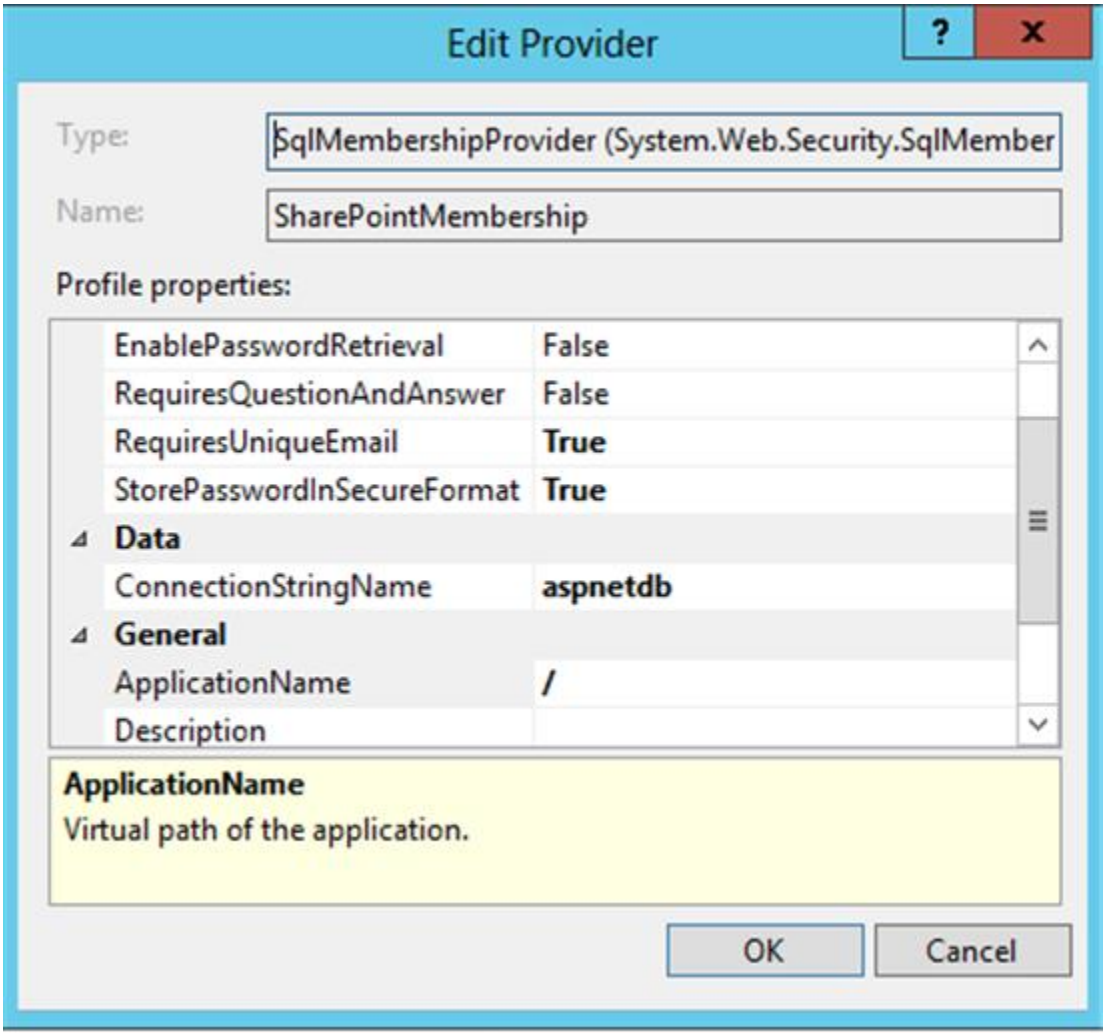


As like creating Roles Provider, we need to create the Membership provider. In provider Page Select .Net Users in Features. And click add link to add new membership provider.

Select SQL Membership Provider as the type & Give Membership Name. we need this name at a time of changing our web application to FBA.

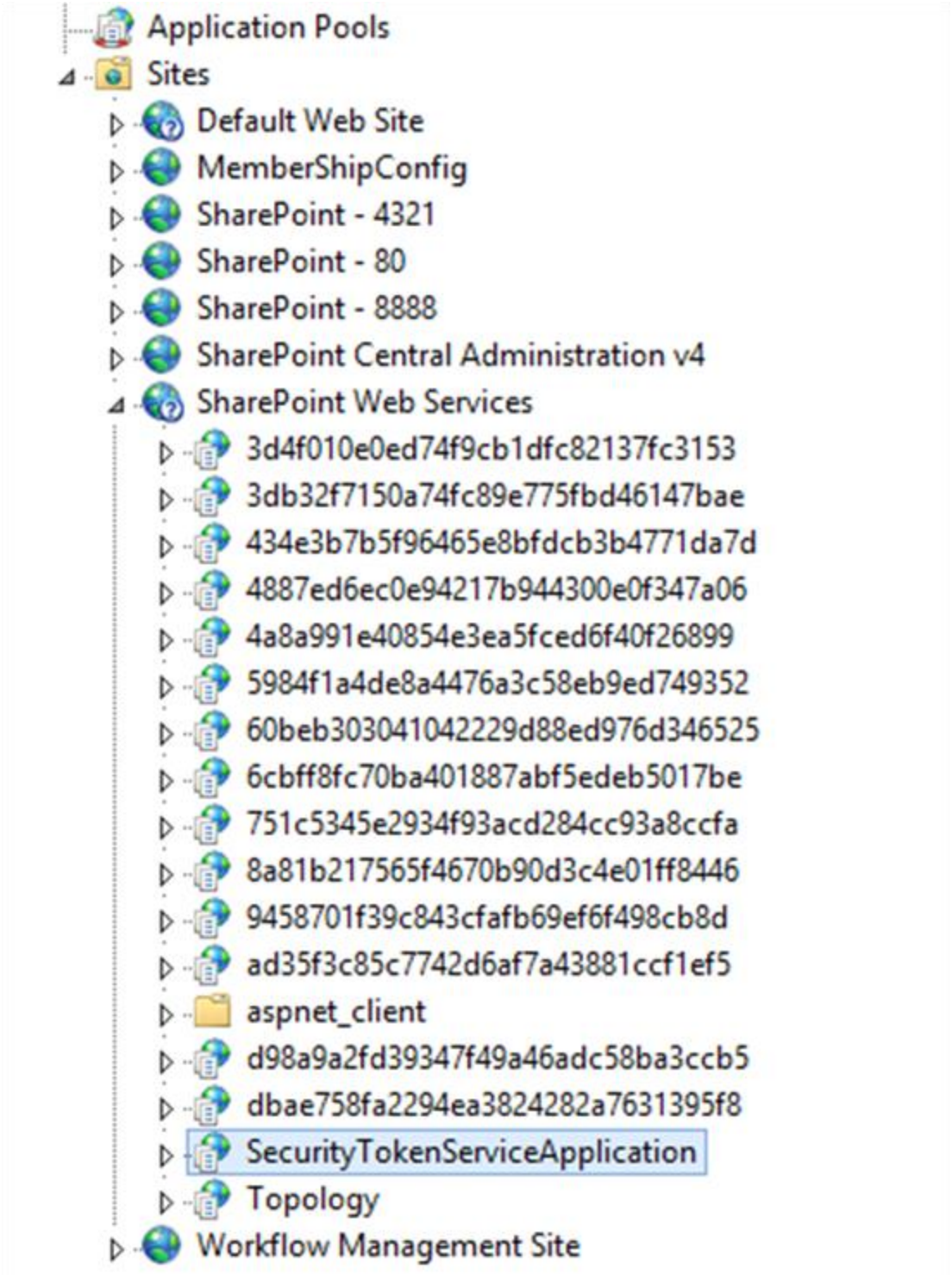


Select Connection string &
make **true on RequiresUniqueEmail , StorepasswordInSercuredManner.**



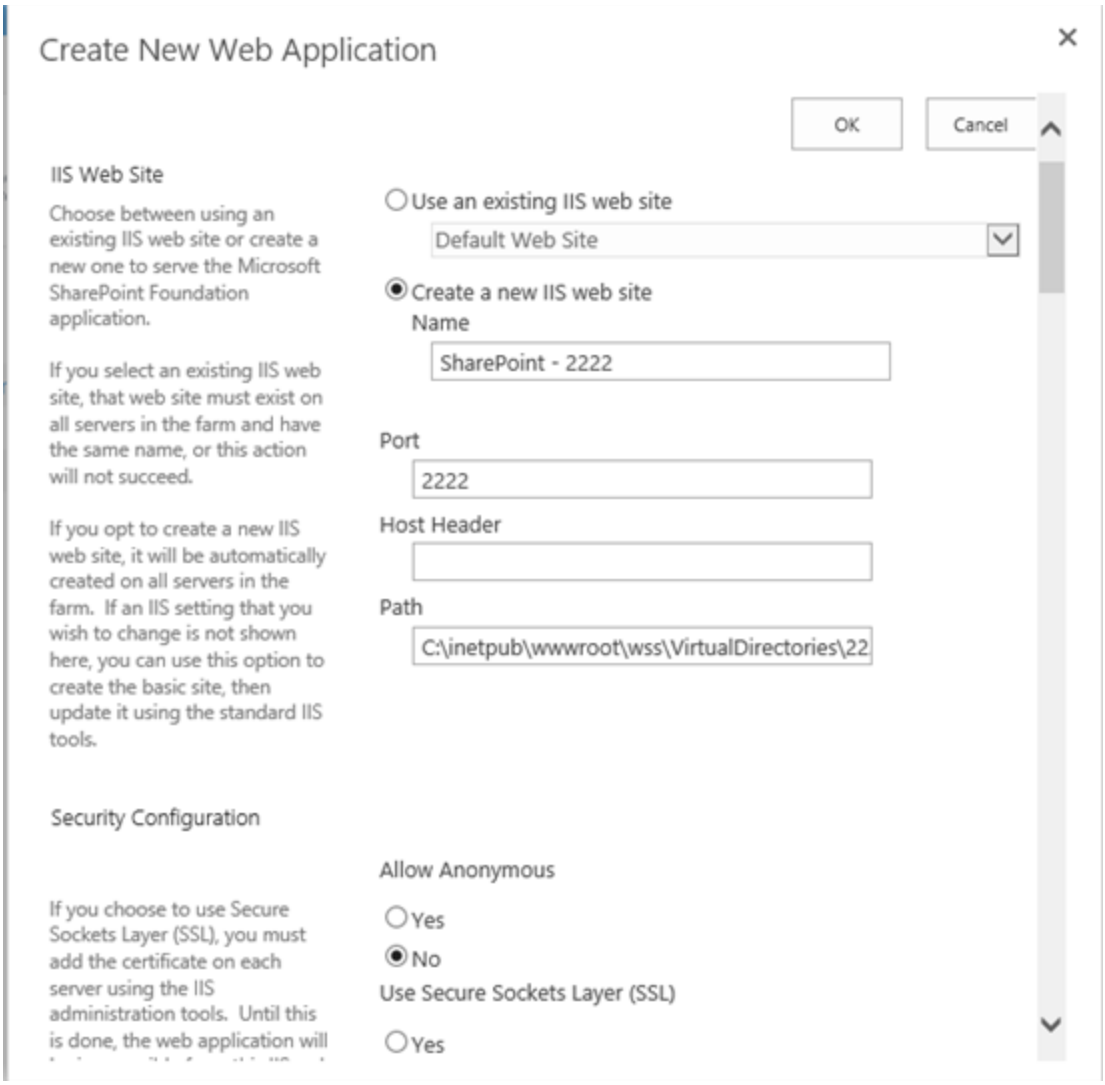
Step 5. Do step 4 in SecurityTokenServiceApplication

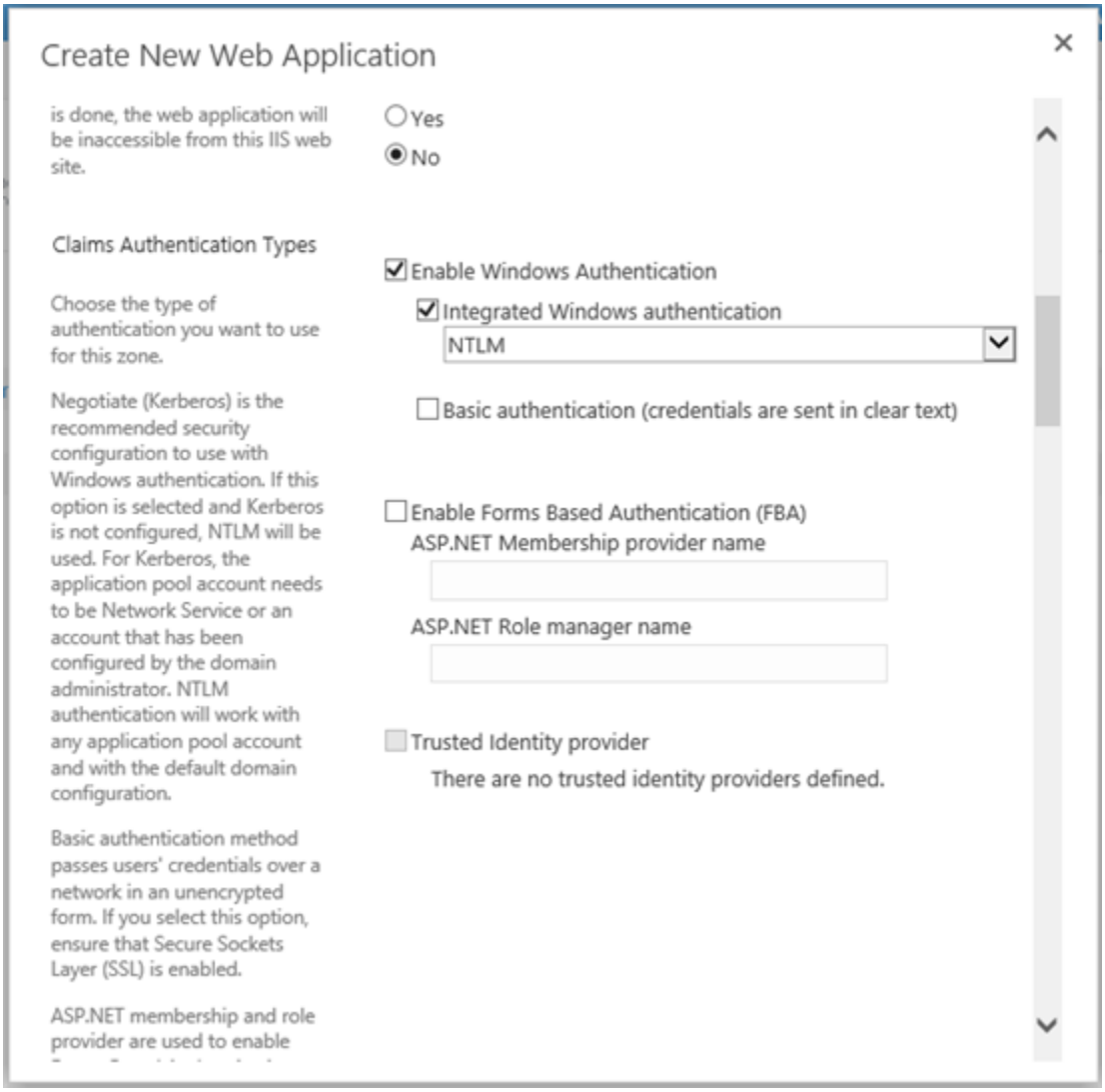
Select your **SharePoint SecurityTokenServiceApplication under SharePoint Web Services.** And create the Roles and Membership providers for this **SecurityTokenServiceApplication as like in step 4.**


Application Pools
Sites
Default Web Site
MemberShipConfig
SharePoint - 4321
SharePoint - 80
SharePoint - 8888
SharePoint Central Administration v4
SharePoint Web Services
3d4f010e0ed74f9cb1dfc82137fc3153
3db32f7150a74fc89e775fbd46147bae
434e3b7b5f96465e8bfdcb3b4771da7d
4887ed6ec0e94217b944300e0f347a06
4a8a991e40854e3ea5fced6f40f26899
5984f1a4de8a4476a3c58eb9ed749352
60beb303041042229d88ed976d346525
6cbff8fc70ba401887abf5edeb5017be
751c5345e2934f93acd284cc93a8ccfa
8a81b217565f4670b90d3c4e01ff8446
9458701f39c843cfafb69ef6f498cb8d
ad35f3c85c7742d6af7a43881ccf1ef5
aspnet_client
d98a9a2fd39347f49a46adc58ba3ccb5
dbae758fa2294ea3824282a7631395f8
SecurityTokenServiceApplication
Topology
Workflow Management Site

Step 6. Create Web Application with claim based authentication

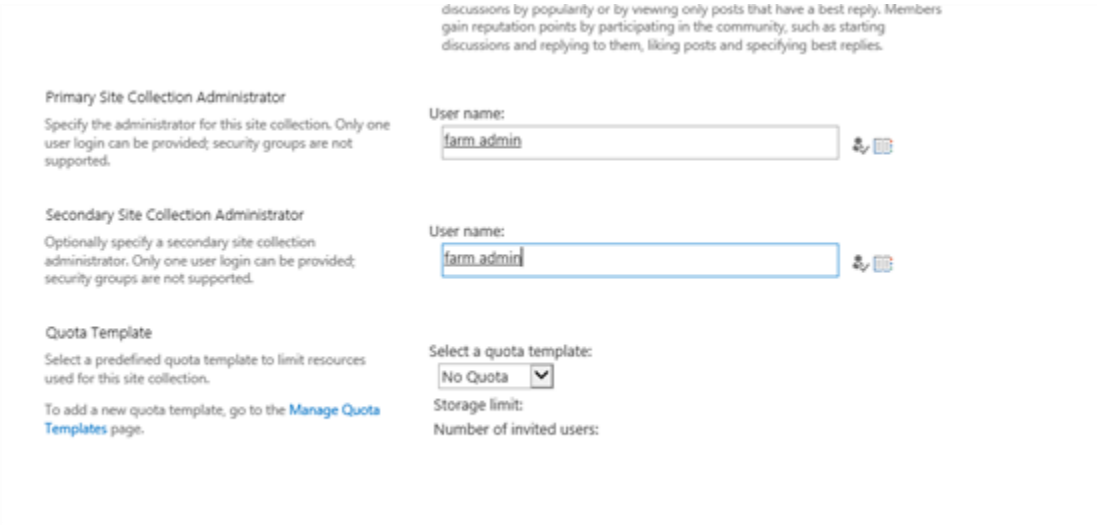
Create the new application with normal claim Based authentication as like below image. Don’t select Enable FAB .


Create New Web Application
OK
Cancel
IIS Web Site
Choose between using an existing IIS web site or create a new one to serve the Microsoft SharePoint Foundation application.
If you select an existing IIS web site, that web site must exist on all servers in the farm and have the same name, or this action will not succeed.
If you opt to create a new IIS web site, it will be automatically created on all servers in the farm. If an IIS setting that you wish to change is not shown here, you can use this option to create the basic site, then update it using the standard IIS tools.
Use an existing IIS web site
Default Web Site
Create a new IIS web site
Name
SharePoint - 2222
Port
2222
Host Header
Path
C:\inetpub\wwwroot\wss\VirtualDirectories\22
Security Configuration
If you choose to use Secure Sockets Layer (SSL), you must add the certificate on each server using the IIS administration tools. Until this is done, the web application will
Allow Anonymous
Yes
No
Use Secure Sockets Layer (SSL)
Yes



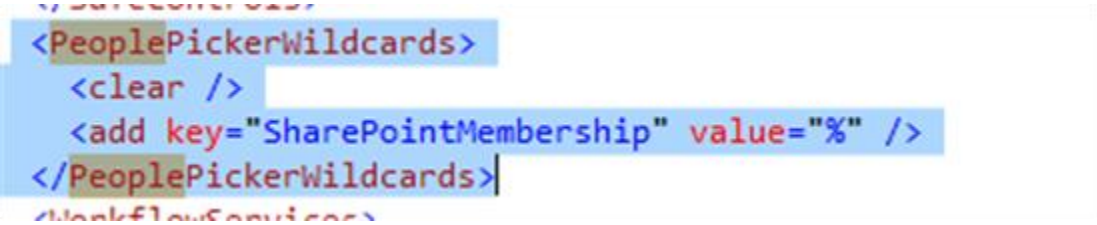
Step 7. Create Site Collection with windows authentication

Create the site collection for create web application and give farmadmin as site collection administration.



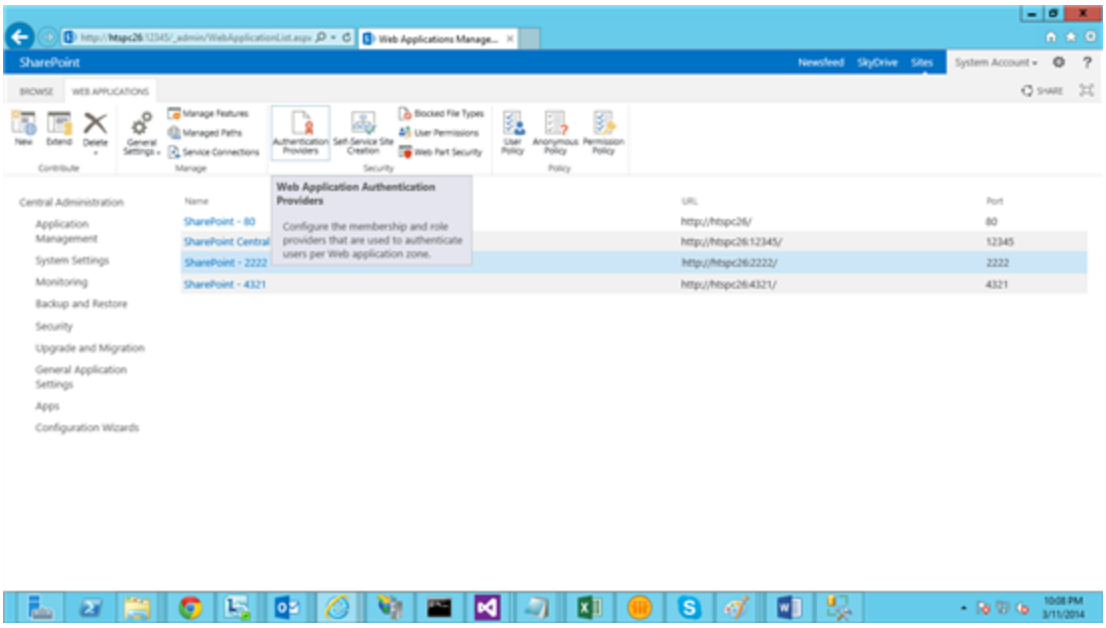
Step 8. Modification in webConfig file in Web Application & Central Admin

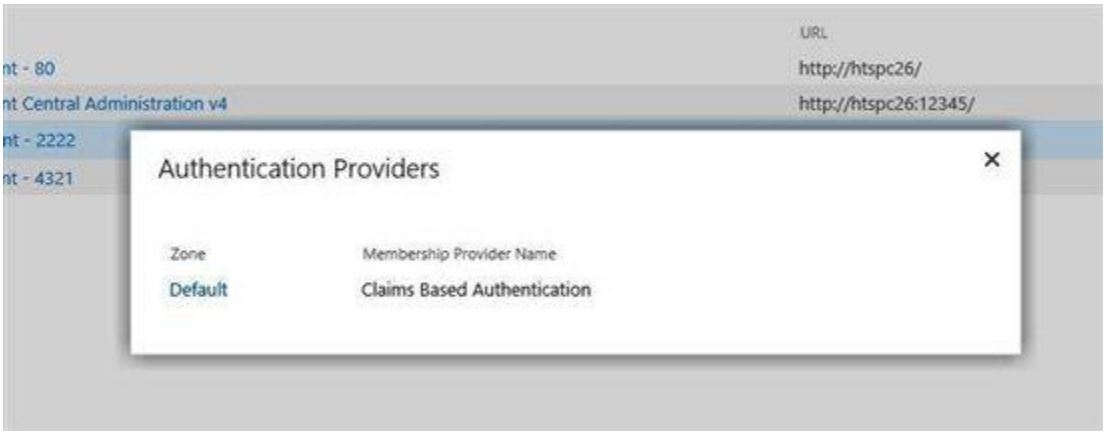
Open wenconfig file of Central admin & Web application and Find the name " **PeoplePickerWildCards".** And change the Key name to our Membership Provider name that we created in IIS.

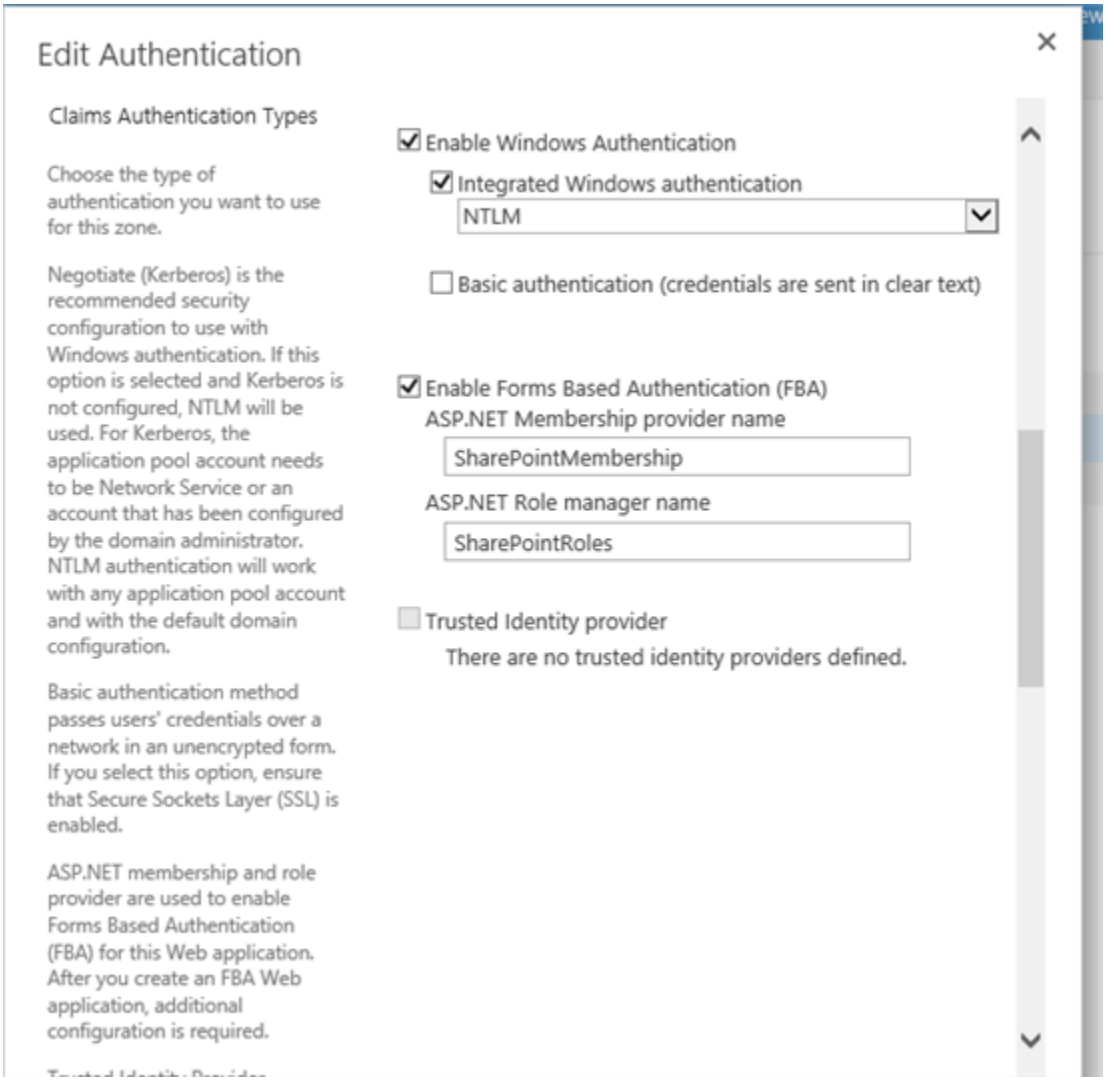


Step 9. Change your web application to FBA

Now Change your Web application in FAB by fllowing steps. Give the Membership provider name & Roles Provider name correctly in this step.

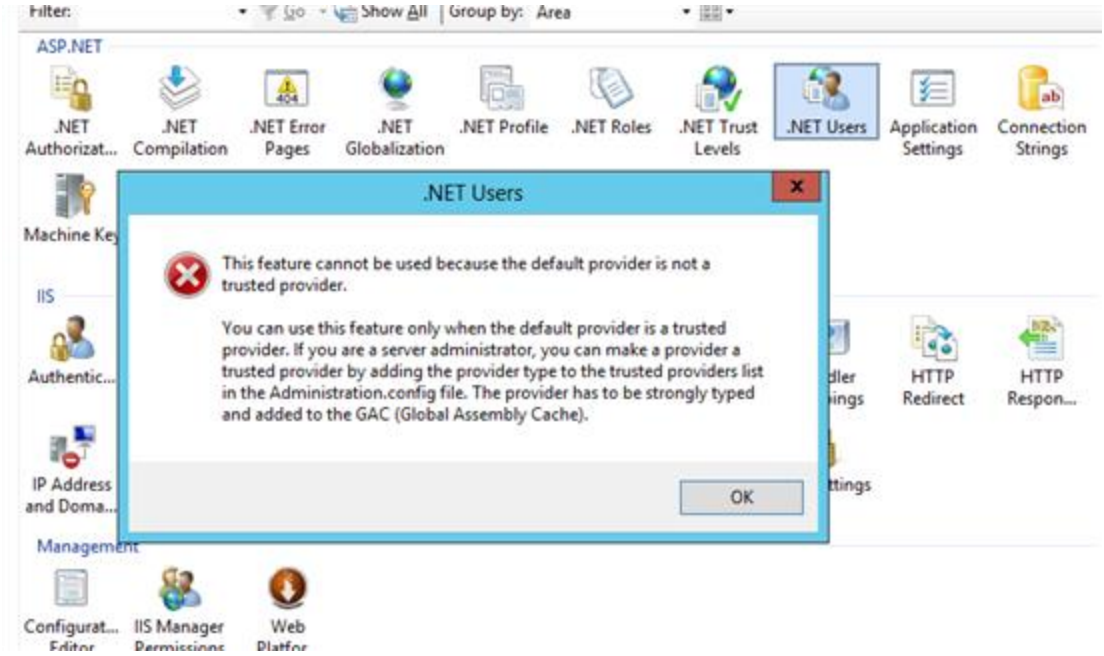

Web Applications Manage...
SharePoint
Newsfeed
SkyDrive
Sites
System Account
BROWSE
WEB APPLICATIONS
Authentication Providers
Web Application Authentication Providers
Configure the membership and role providers that are used to authenticate users per Web application zone.
Central Administration
Application Management
System Settings
Monitoring
Backup and Restore
Security
Upgrade and Migration
General Application Settings
Apps
Configuration Wizards
Name
URL
Port
SharePoint - 80
http://htspc26/
80
SharePoint Central
http://htspc26:12345/
12345
SharePoint - 2222
http://htspc26:2222/
2222
SharePoint - 4321
http://htspc26:4321/
4321

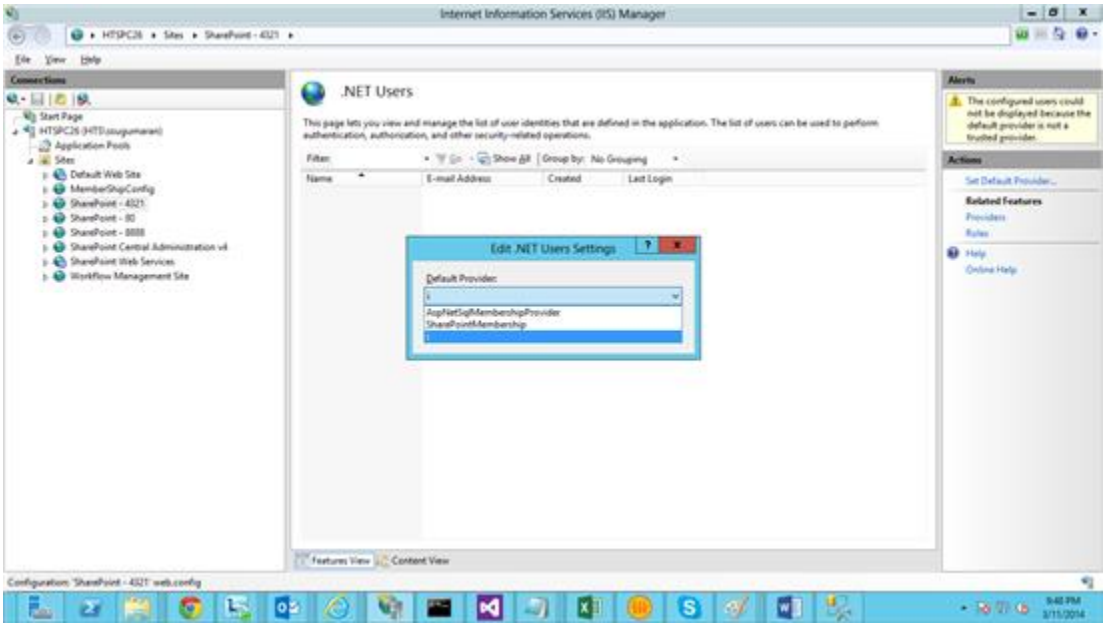

URL
nt - 80
http://htspc26/
nt Central Administration v4
http://htspc26:12345/
nt - 2222
nt - 4321
Authentication Providers
Zone
Membership Provider Name
Default
Claims Based Authentication



Step 10. Select your web application in IIS and add .net Users, if error changes the default provider in .net Roles and .net Users.

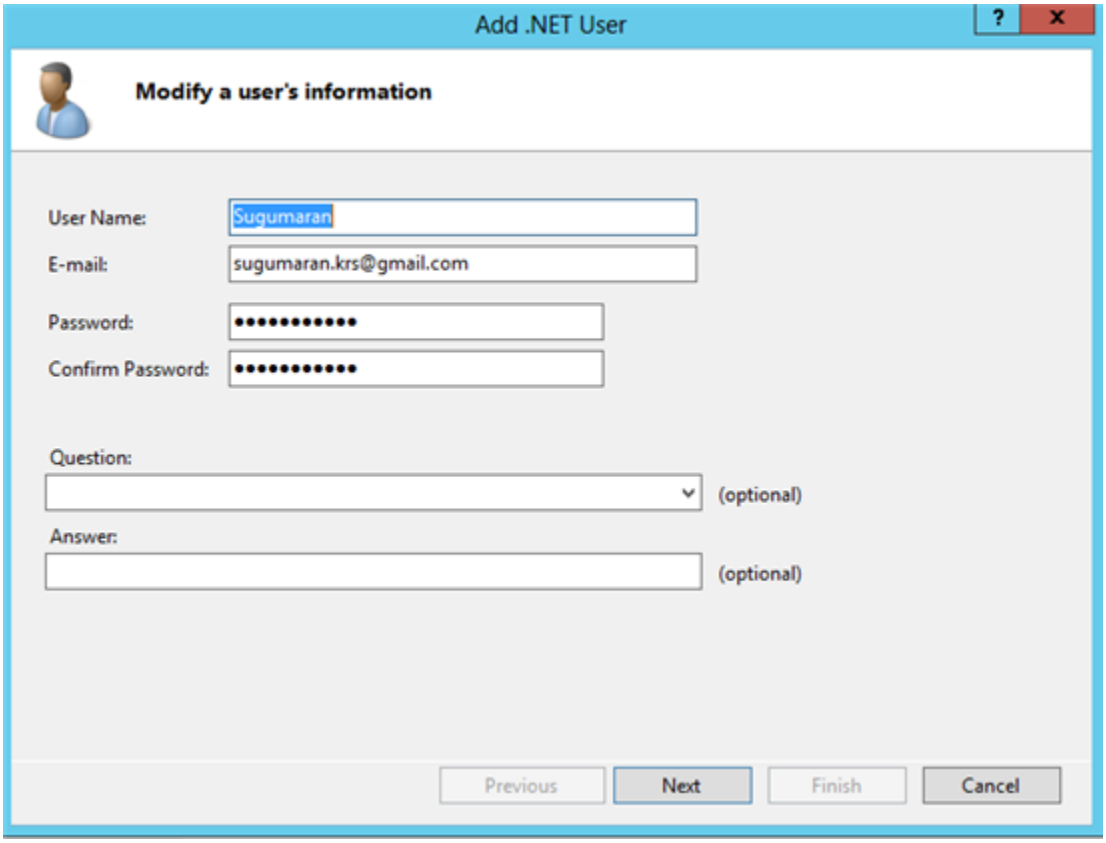
Change the default provider as our membership provider and roles providers in .Nets Users & .Net Roles respectively.





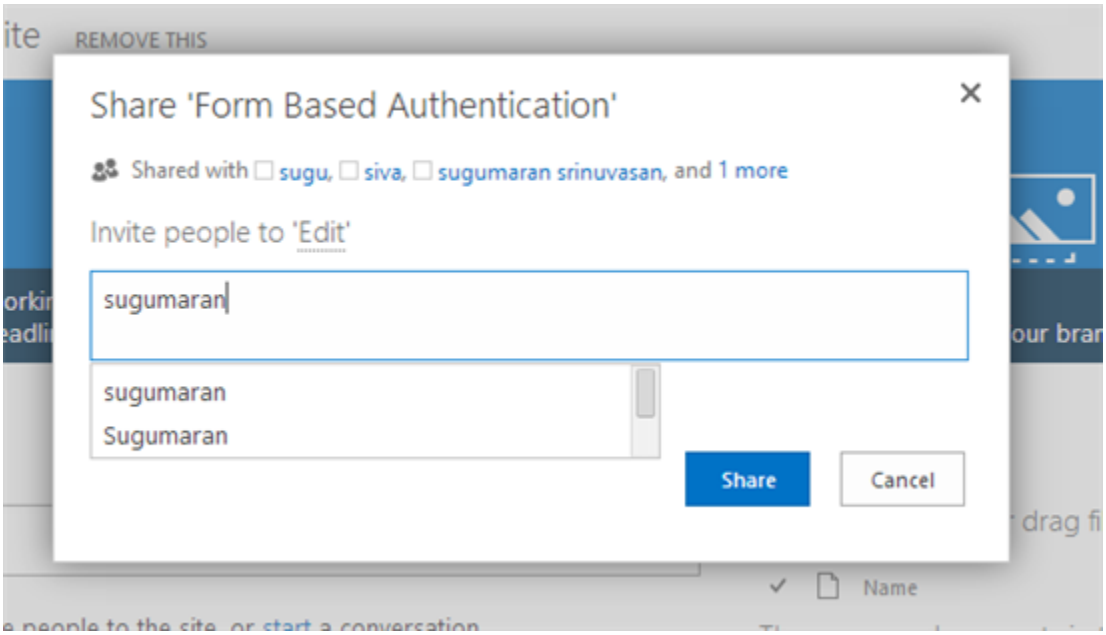
Step 11. Add the .net user in IIS.

Now add new users to this member ship by click .Net Users and Click Add Users.



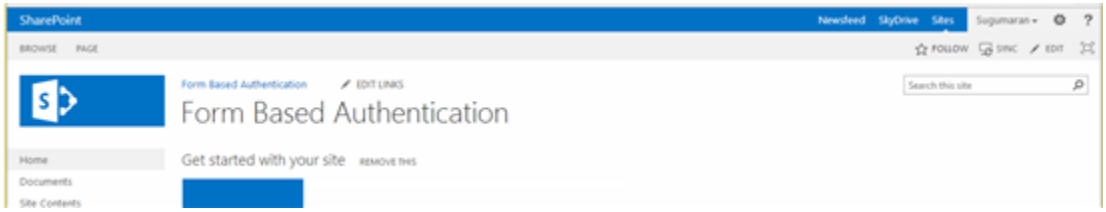
Step 12. Open the web application with windows authentication and share your site with FBA users.

Open web application with windows authentication (farmadmin account) and give site acces to FBA user that we created.



Step 13. Now login with FBA account.

Great, Now we can sign in with different user and we can open our site with Farm Authentication.



SharePoint 2013: Organization Browser Web Part does not render in HTML View for Windows Claims Users

Organization Browser Web Part does not render in HTML View for Windows Claims Users

I have a requirement to display HTML View of Organization Browser in one of the SharePoint Site So I have inserted Organization Browser web part in SharePoint 2013 page and observed that organization hierarchy of the logged-in user is not rendered anything either in HTML View or Silverlight View.

I was searching about this issue in Microsoft sites and figured out one KB articlehttp://support.microsoft.com/kb/2643420 about organization browser issue and saw a fix for Silverlight View in this article but no fix for HTML View. In the same KB article at the bottom of the page, they mentioned about HTML view issue saying "it not possible to overwrite the Html View of the Organization Browser web part from the client". I am not stopped myself after seeing that message and continued further investigating about HTML View fix.

After some much of R & D, I am successful fixing HTML View issue by overriding the following HTML View function

```
<script type="text/javascript">
SP.UI.Portal.SimpleProfileBrowser.prototype.$2S_0 = function ($p0,
$p1, $p2) {
var i = $p0.indexOf("|");
$p0 = $p0.substr(i+1,$p0.length-i-1);

        var $v_0 = $get($p1);
        if ($v_0) {
            $v_0.innerHTML = '<DIV></DIV><DIV></DIV><DIV></DIV>';
            this.$h_0 = $v_0.firstChild;
            this.$16_0 = this.$h_0.nextSibling;
            this.$1e_0 = this.$16_0.nextSibling;
        }
        if ($p2) {
            this.$h_0.innerHTML =
String.format(SpsClient.ScriptResources.silverlight_Install_Mess
```

```
age, '<a
href=\'javascript:Silverlight.getSilverlight(\"2.0\");\'>Silverl
ight</a>');
            Sys.UI.DomElement.addCssClass(this.$h_0, 'ms-
profileBrowserHeaderText');
            Sys.UI.DomElement.addCssClass(this.$h_0.firstChild.n
extSibling, 'ms-profileBrowserSilverlightLink');
        }
        this.$2V_0($p0);
}
</script>
```

Below are the steps to add Organization Browser web part to the page and then adding a HTML View fix to the page.

Steps to add Organization Browser in a SharePoint 2013

1. Edit the page
2. Click on Insert web part at the top navigation bar
3. Go to “Social Collaboration” category and select “Organization Browser” and click on Add button
4. Edit the Organization Browser web part and Expand Default Values and check “Only show HTML View” option and click Ok button

Steps to fix HTML View of Organization Browser

1. Edit the page (if the page is not in edit mode).
2. Click on Insert web part at the top navigation bar
3. Go to “Media and Content” category and select “Content Editor” Web Part and click on Add button.
4. Edit the content.
5. In the Ribbon, click on Edit Source under the Markup tab.
6. Copy-paste the code above.
7. Hide the Chrome and the content of the web part so that it would not appear on the page.

I am hoping Microsoft will take care of Claims Users issue for Organization Browser web part in the next SharePoint releases..

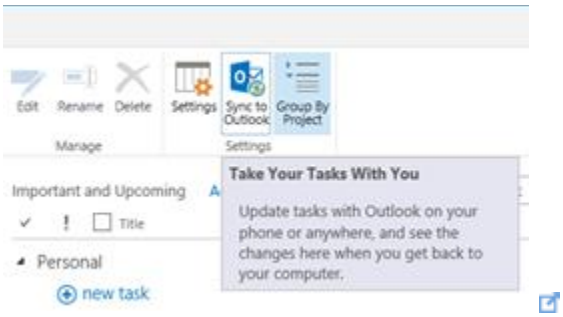
SharePoint 2013: Integration between SharePoint and Exchange can only be disabled on-premises

Accessing the My Tasks feature

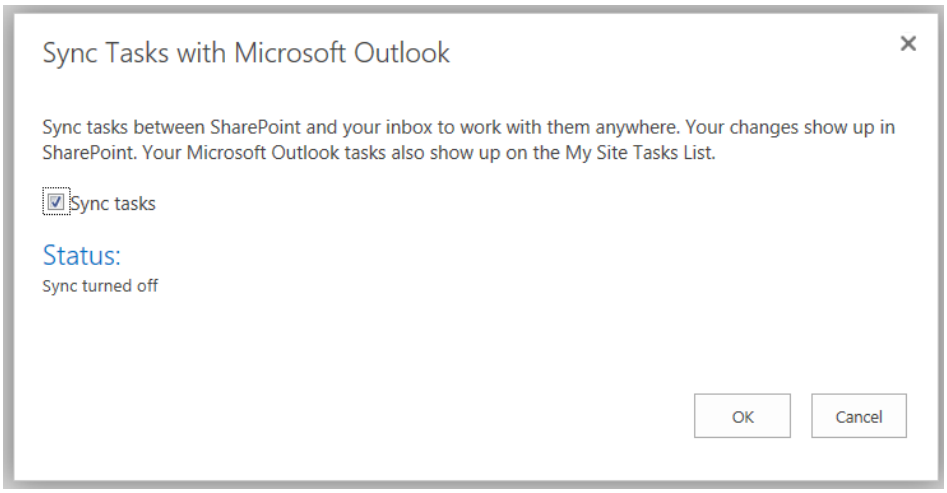
If the server is set with all the prerequisites, then the user can access the **My Tasks** page by navigating to his or her **Personal** page and then clicking the **Task** link in the left column.

Issue from a PowerUser

When a user navigates to the SharePoint's **My Task** area, that user can find the **Sync to Outlook** button in the **Tasks** pane of the ribbon.

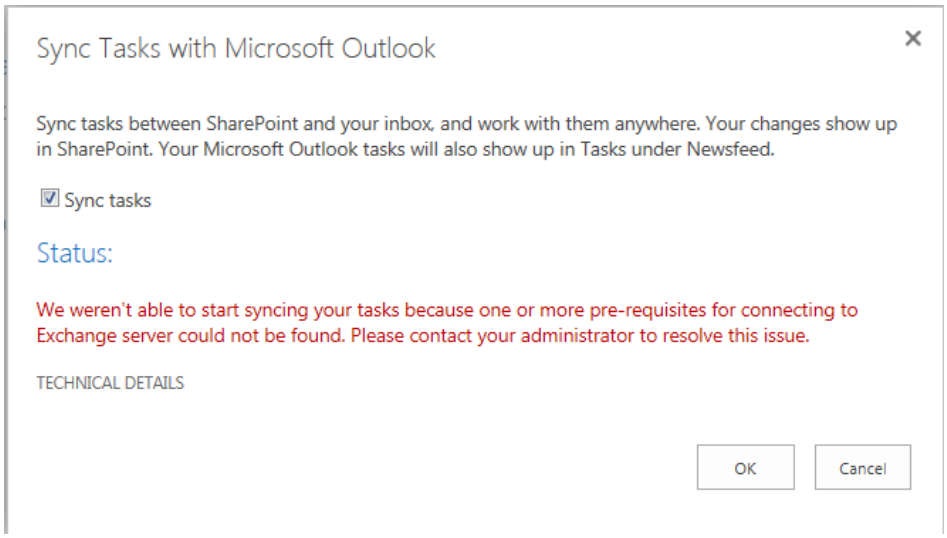


After clicking that button, the opt-in dialog box is shown. Select the **Sync tasks** check box to opt in.



When try to synchronize **My Tasks** with **Outlook 2010** you get the following error:

We weren't able to start syncing your tasks because one or more prerequisites for connection to Exchange server could not be found. Please contact your Administrator to resolve the issue.



According to some posts

A little search on the internet gave me this blog post: (http://netwovenblogs.com/2013/07/27/sharepoint-2013-outlook-tasks-synchronization-issue/) who confirms that I need the following prerequisites:

- You will need to have Exchange 2013 in your environment.
- You will also need to install Exchange 2013 Web Services Managed API 2.0

Strange issue isn't it? This is also meaning that all the corporates using Office 2007 or Office 2010 couldn't synchronize any task list anymore.

According to Microsoft TechNet

According to Microsoft TechNet: Before you begin this operation, we have review the following information about prerequisites:

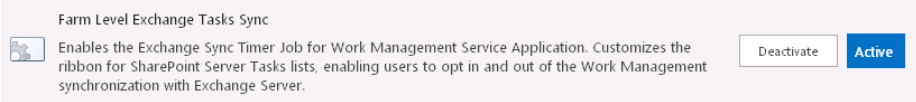
- Task Synchronization requires that user profile synchronization be configured in the farm. For information about configuring user profile synchronization, see Plan user profiles and identities (SharePoint Server 2013) , and Manage user profile synchronization in SharePoint Server 2013 .
- Task Synchronization requires that the work management service application be configured in the farm. For information about creating the work management service application, see New-SPWorkManagementServiceApplication
- **Task Synchronization requires Exchange Server 2013.**
- Secure Sockets Layer (SSL) is a requirement for web applications that are deployed in scenarios that support server-to-server authentication and app authentication. This is such a scenario. As a prerequisite for configuring Task Synchronization, the computer that is running SharePoint Server must have SSL configured. For more information, see Create claims-based web applications in

SharePoint 2013 and follow the steps for creating an SSL site collection and server certificate. **Even TechNet confirms that we need Exchange Server 2013!**

Resolution

After a while of searching we could found that Integration between SharePoint Server and Exchange Server can only be disabled on-premises who will resolve our issue.

To disable it, you turn off the **Farm Level Exchange Task Sync feature by going to Central Administration è Manage farm features**. This action deletes the timer job and disables the UI integration, bringing back the legacy sync behavior to Outlook.



How does it work after deactivating the farm feature?

Refreshing or accessing the **My Tasks** page launches the aggregator code, if the last time the aggregator ran was more than **five minutes** prior. This artificial delay is put in place to preserve performance.

There's no requirement to refresh the page after inline edits, as inline changes are instantly replicated on the original provider. The five-minute delay doesn't apply to them.

This value can be changed from on-premises deployments by altering the property on the WMA service application object by using Windows PowerShell:**minimumTimeBetweenProviderRefreshes 00:05:00**

Reference

http://social.technet.microsoft.com/Forums/sharepoint/en-US/bd7446be-cd01-4098-9ebd-6b67fc2cc1e2/error-when-trying-to-sync-tasks-to-outlook-from-sharepoint-2013?forum=sharepointgeneral

SharePoint 2010: Release Distribution Process – Gotchas

A **Release Distribution Process** is actually not more than governance that you adopt depending your business needs and IT requirements.

A simple example can be; the business requirement is a new Secure (HTTPS) SharePoint site in the DMZ-Legacy zone for tomorrow with a few globally deployed solutions.

You can't do this at 3:00PM when everybody is using your SharePoint platform. Installing or Removing "not" targeted Web Application Solutions will perturb the Application Pools. This means potentially that there can be a "downtime" of few minutes.

A release distribution process can be:

- All modifications are done after 6:00PM
- Content Deployment is done after 8:00PM
- Only if I'm impacted by a problem and a Cumulative Update resolves it then we install it in the WK between 0:00AM and 0:00PM
- Only Emergency deploys can be done at 1PM (during the lunch)
- Service Packs are installed every 6 months during the monthly reboot
- …

This list can be very huge and long depending your IT and SharePoint governance. Not all Release Distributions processes are fitting the Best Practices of SharePoint and there is not a need too! If your Business is happy with a roll-out of a Service Pack during the work hours you don't have to wait until 6:00PM!

A major rollout without downtime; this is the requirement of our business. Unfortunately it's not possible to update your SharePoint Platform without having some amount downtime. But we can optimize our **Release Distribution Process to minimize downtime.**

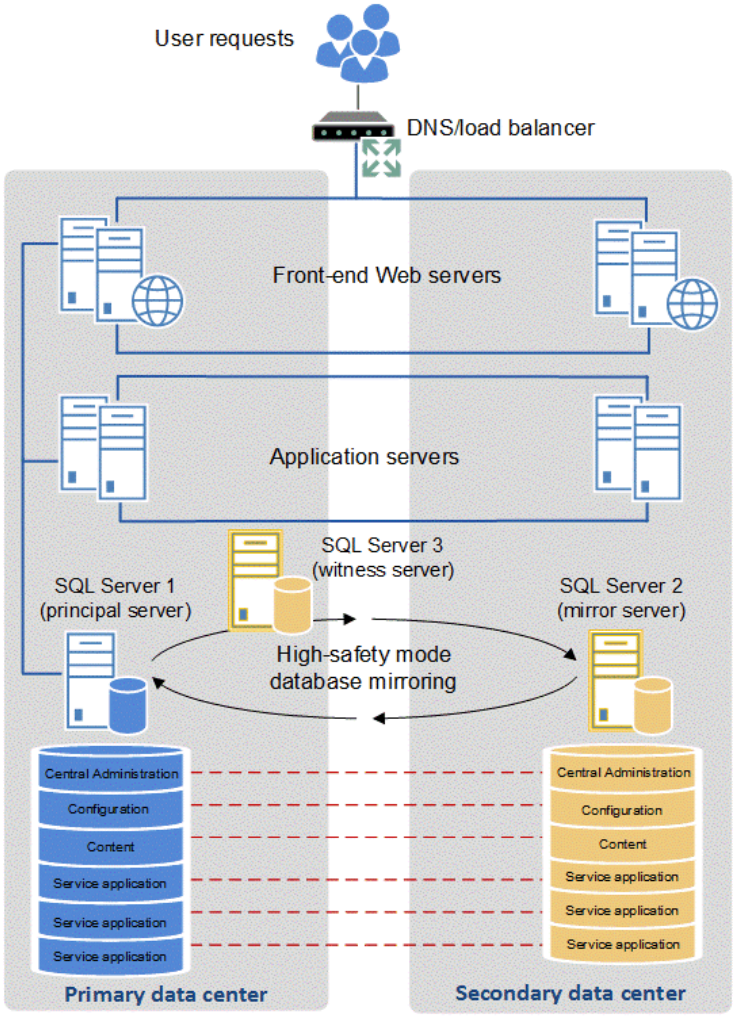
Possible solutions

The use of a common data store for all SharePoint farm members and the dependency on IIS, means that you cannot update a SharePoint farm without “some” downtime unless a fully synchronized standby farm is available (so, deploying SharePoint Packages means outage to your Web Applications).

There are a few methods available to minimize the amount of downtime; however it’s just not possible to achieve **a zero downtime** solution for your upgrade. A common way to minimize downtime is via implementation of a parallel upgrade farm.

Stretched Farms

A great example is the Stretched Farm, even if it is not a perfect solution.



For stretched farm architecture to work as a supported high availability solution the follow prerequisites must be met:

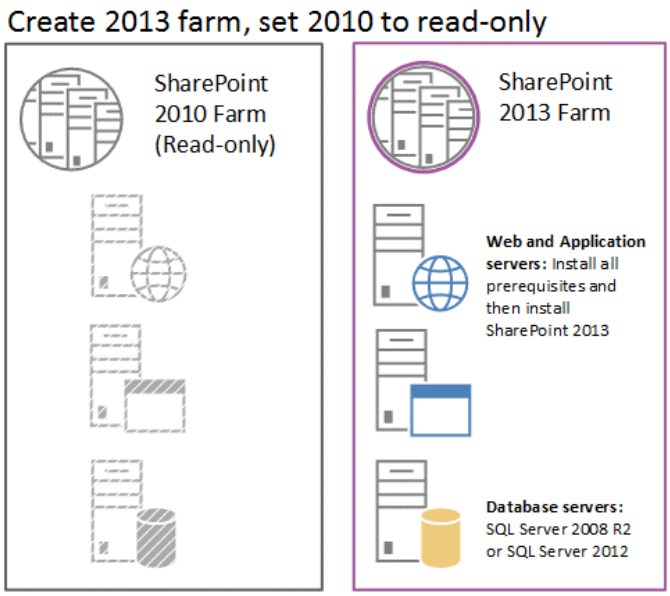
- There is a highly consistent intra-farm latency of <1ms (one way), 99.9% of the time over a period of ten minutes. (Intra-farm latency is commonly defined as the latency between the front-end web servers and the database servers.)
- The bandwidth speed must be at least 1 gigabit per second.

The TechNet Article has more information about the Stretched Farms: here .

SharePoint Read-Only Farms

The functionality and user experience in a read-only farm depends on which databases are set to read-only. A farm that uses read-only content and service application databases is likely to be part of a disaster recovery environment or a highly available maintenance, update, or upgrade environment.

You may choose a switching mechanism (DNS, load balancing, etc), that meets your needs. Read more about this configuration, its caveats, & the end user experience it provides: Run a farm that uses read-only databases (SharePoint Server 2010) http://technet.microsoft.com/en-us/library/dd793608(v=office.14).aspx



Local deployment

In a local deployment, solution files are deployed only to the computer from which the deployment operation was initiated. The solution is not marked as “deployed” in the configuration database until the solution files are deployed to all applicable servers in the server farm. Then solution features are installed, and schema and definition files are committed to the configuration store.

There are some situations where this way of deployment just does not work, especially when the solution updates the content and code at the same time. The content is in the DB and shared by all the WFEs. So, you should first consider what type of customizations you do and which allow for zero downtime and which not. Note Web services can be deployed and only recycle the used app pool.

Another example, assuming you’re only updating a .dll:

- Disable the timer job on all front-ends
- On the CA server (not load balanced), Update-SPSolution
- For each of the front-ends
    - Take the front-end out of rotation
    - Launch stsadm –o execadmsvcjobs. This will cause the upgrade to execute on that particular front-end only.
    - Bring the front-end back into rotation
    - Enable the timer job

Release Distribution checklist

Although this approach will for most cases work there are a few gotchas that you need to watch out for. In some of these occasions all SharePoint WFEs will have their application pools restarted.

1) Try to avoid -Force

Use the Force switch only for fixing broken deployments of SharePoint Packages. Not only it makes SharePoint stop affected Application Pools but also it prevents you from seeing errors should there be any.

2) Retracting packages

There is a recycle on the Application Pool, Why don't you do this at 04:00AM. On Each front-end Web server, the following occurs:

- Microsoft Internet Information Services (IIS) is disabled.
- Files are removed from the system.
- IIS is re-enabled and Windows SharePoint Services is reloaded when a user browses to a page.

3) Scope your solution

When a solution is deployed globally, all SharePoint application pools, including Central Administration's, are recycled automatically. This can be good and bad. This is good because any GAC installed DLL that has been upgraded needs to be reloaded. This can be bad though with regards to the availability of your entire SharePoint Farm.

4) Recycle

Set the ResetWebServerModeOnUpgrade attribute to **Recycle**. You have to do this explicitly in the SharePoint Package configuration. Without it, the setting will default to the StartStop.

5) Good resource

Who wrote my solution? Anybody who I can fully trust? Does he write great code? Does he an IISReset in his code?

6) Reduce the amount

Do not create a WSP for layout Pages, CSS. Try to minimize the amount of solutions. 'Aspx' files, DLL and controls are going to the application domain and these files have to be compiled. Layouts, CSS, resources not!

7) Time limit

Schedule retracting of SharePoint Packages when there is the least traffic on your Web Applications as it always stops affected Application Pools.

8) Hardware and software boundary limits

On many TechNet articles you can see that per Application Pool you need 2GB of RAM. A simple calculus could be for 40 Application Pools => 80GB or RAM. This is not true! Try to respect the 12 Application Pools per server and be generous with the RAM (depending your hardware).

9) Separated Application Pools

Many websites can be hosted on one Application Pool, but Many Application Pools cannot be used by a Web Application. So the question can be how can I manage my Web Applications while keeping in mind the 12 Web Application Pools Limit? Well, all the Application Pools should be together by usage or anything else and divided by authentication model (claims, anonymous …)

10) Avoid creating a lot of global SharePoint Packages

Avoid creating a lot of global SharePoint Packages and try instead to provision as much as you can to specific Web Applications. Every time you touch a global SharePoint Package all Applications Pools will be stopped/recycled.

According to Microsoftbe

After several days of discussion with @MicrosoftBe and several **tests on our Dev machines**, here are the results of our tests on the behavior of pools Application.. A special thanks for my colleague and Solution Architect @Tomvanrousselt for that !

Please hit ME or the image to download the Excel Sheet!

The deployment process is a black box

According to @Microsoftbe: (Note these remarks are valid for globally deployed solutions as deployments targeted towards particular applications should not have any impact on other web applications: http://msdn.microsoft.com/en-us/library/ms412929(v=office.14).aspx)

1. If you retract/deploy the solution which marked as "DeploymentServerType = ApplicationServer" then the**application will never stopped or recycle.**
2. If you retract the solution which marked as "DeploymentServerType = WebFrontEnd" from SharePoint then SharePoint will first stop all the **application pools and then start then one by one**
3. If we upgrade the solution which marked as "DeploymentServerType = WebFrontEnd" from SharePoint then**SharePoint will first stop all the application pool and then start then one by one**
4. If we deploy the solution to SharePoint with -force then SharePoint then **SharePoint will first stop all the application pool and then start then one by one**
5. If we deploy the solution to SharePoint without –force (or in the central admin) then **SharePoint will recycle the application at the end of the progress**
6. If we deploy the solution to SharePoint without –force (or in the central admin) and "ResetWebServer = false" then**SharePoint will not recycle the application pool any more**. But to make your new solution works you need to recycle the application pool manually.

Then in your scenario if you want to reduce the downtime please do the following:

1. Make sure you are using the right "DeploymentServerType" value to **avoid the unnecessary restart on the application server**
2. Avoid to retract/upgrade the solution with "DeploymentServerType = WebFrontEnd" in the business time. As this scenario which **stops all the application pool and start them again**
3. Deploy the solution without –force (or in the central admin) SharePoint **will recycle them rather than stop and start them again**
4. Deploy the solution without –force and "ResetWebServer = false" in the solution manifest file will make the**application pool with no recycle**. But you need to recycle the application by yourself accordingly. Be sure that this has to be done manually, because Visual Studio creates for you the Manifest file.

Conclusion

The deployment process on each front-end Web server involves copying files into system directories, installing features, and adding assembly DLLs to the GAC. Furthermore, SharePoint always runs a command to restart all the IIS worker processes that might be affected by the new components that have just been installed. For this reason, in production environments you should deploy solution packages during off hours with the provided gotchas when the fewest number of users are using the system.

References and Additional Info

- Optimizing the process of deploying SharePoint Packages to minimize the impact on farm's availability
- The Zero Downtime SharePoint Patching Myth
- How we deployed SharePoint WSP Solutions without downtime

SharePoint 2010: Performance Differences of Search verse Recursively Looping

Introduction

A question about returning all the sites (SPWeb's) "the current user" has access to in a given site collection comes up regularly in the TechNet SharePoint forums. The question usually asked is, "is there a method that returns all the sub-webs" of a site collection that a user has access to, or do we need to recursively loop through each web in the site collection, checking if the user has a specific permission to view the web?

The answer is regularly that you need to loop through the collection of webs (recursively), to determine the list of webs the user has access to.

Depending on the size of a site collection, this can be a very expensive and time consuming operation.

There is another way to achieve this requirement, using Search. This article explores using Search to generate a list of webs a user has access to, examines the performance differences between Search and Looping through collections, as well as some potential pros and cons.

Creating a Webpart to Test the Performance of Both Methods.

To compare the difference in performance and the results produced from each method, we are going to create a test webpart. The webpart is very simple, containing two main methods. One method is used for generating the list of webs by looping (calling SPWeb.GetSubwebsForCurrentUser() on each web), and the other method is using the

SharePoint Search infrastructure, via the KeywordSearch class. Each of these methods is wrapped in an SPMonitoredScope block, enabling the performance of the each method to be tracked. The results can be seen in the Developer Dashboard.

The method that uses SPWeb.GetSubwebsForCurrentUser() starts at the root web for the site collection, an traverse down, calling GetSubwebsForCurrentUser() on each child web of the current web, until it finishes enumerating all the webs the current user has access to.

The search query used in the search method, queries the search engine for "ALL sites AND webs WHERE the webapplication hostname STARTS WITH the current sites hostname". You can test out the results of this search query using the SharePoint UI, via a standard Enterprise Search site. The search command would look something similar to this, if you were searching for all sub-sites you had access to on the http://corporate site collection:

(contentclass:STS_SITE OR contentclass:STS_Web) AND sitename:http://corporate

WebPart Code for Testing the Performance of Both Methods

```
namespace SearchVerseLoop.GetTheSitesIHavePermissionsToSee
{
    [ToolboxItemAttribute(false)]
    public class GetTheSitesIHavePermissionsToSee : WebPart
    {
        private Label _sitesFromSearch;
        private Label _sitesFromLooping;

        protected override void CreateChildControls()
        {
            _sitesFromSearch = new Label();
```

```
            _sitesFromLooping = new Label();
            Controls.Add(_sitesFromSearch);
            Controls.Add(_sitesFromLooping);
        }

        protected override void OnPreRender(EventArgs e)
        {
            base.OnPreRender(e);
            _sitesFromLooping.Text = GetAllWebs();
        }

        private String GetAllWebs()
        {
            try
            {
                var output = new StringBuilder();
                var websFromLooping = new ArrayList();
                using (new SPMonitoredScope("using a loop"))
                {
                    GetListOfWebs(SPContext.Current.Site.RootWeb
, SPContext.Current.Site.RootWeb.GetSubwebsForCurrentUser(),
websFromLooping);
                    output.Append(String.Format("<p>There are
{0} webs I have access to (retrieved from looping through the
rootwebs sub-webs)</p>", websFromLooping.Count));
                    foreach (var web in websFromLooping)
                    {
                        output.Append(String.Format("<span>{0}</
span><br/>", web));
                    }
                }
```

```
                var websFromSearching = new ArrayList();
                using (new SPMonitoredScope("using search"))
                {
                    GetListOfWebsFromSearch(websFromSearching);
                    output.Append(String.Format("<p>There are
{0} webs I have access to (retrieved from search, filtering on
the current site)</p>", websFromSearching.Count));
                    foreach (var web in websFromSearching)
                    {
                        output.Append(String.Format("<span>{0}</
span><br/>", web));
                    }
                }

                return output.ToString();
            }
            catch (Exception e)
            {
                return e.Message;
            }
        }

        private void GetListOfWebs(SPWeb currentWeb,
IEnumerable<SPWeb> webCollection, ArrayList webs)
        {
            webs.Add(currentWeb.Url);
            foreach (SPWeb web in webCollection)
            {
                if (web.GetSubwebsForCurrentUser().Count > 0)
                {
```

```
                    GetListOfWebs(web,
web.GetSubwebsForCurrentUser(), webs);
                }
                else
                {
                    webs.Add(web.Url);
                }
            }
        }

        private void GetListOfWebsFromSearch(ArrayList webs)
        {
            var ssaProxy =
(SearchServiceApplicationProxy)SearchServiceApplicationProxy.Get
Proxy(SPServiceContext.GetContext(SPContext.Current.Site));
            var keywordQuery = new KeywordQuery(ssaProxy)
                {
                    RowLimit = 500,
                    TrimDuplicates = true,
                    ResultsProvider = SearchProvider.Default
                };
            keywordQuery.SelectProperties.Clear();
            keywordQuery.SelectProperties.Add("Path");
            keywordQuery.ResultTypes |=
ResultType.RelevantResults;
            keywordQuery.QueryText =
String.Format("(contentclass:STS_SITE OR contentclass:STS_Web)
AND sitename:{0}", SPContext.Current.Site.HostName);
            ResultTableCollection searchResults;
            try
            {
                searchResults = keywordQuery.Execute();
```

```
            }
            catch (Exception)
            {
                //"Your query is malformed. Please rephrase your
query."
                return;
            }

            if (!searchResults.Exists(ResultType.RelevantResults)
) return;
            var searchResult =
searchResults[ResultType.RelevantResults];
            var results = new DataTable { TableName
= "SearchResults" };
            results.Load(searchResult,
LoadOption.OverwriteChanges);
            foreach (DataRow dataRow in results.Rows)
            {
                webs.Add(dataRow["Path"]);
            }
        }
    }
}
```

Using SPMonitoredScope

In the code above we have two main functions that get called during the PreRender event. Both of these functions are wrapped in an SPMonitoredScope, which will enable us to track performance information about each method, namely the time each method takes to produce the list webs the current user has access to.

Using SPMonitoredScope also allows us to see other valuable information, such as the number and type of SQL calls, and expensive object allocations, like SPRequest allocations.

Turning on the Developer Dashboard with PowerShell

To see the results, we need to enable the Developer Dashboard. There is no user interface in SharePoint for enabling the Developer Dashboard, but thankfully, it's easily enabled using PowerShell.

To enable the Developer Dashboard, logon to your (test) SharePoint server, and open the SharePoint Management Shell.

Execute the following commands to enable the Developer Dashboard.

```
$ds =
[Microsoft.SharePoint.Administration.SPWebService]::ContentService.Dev
eloperDashboardSettings;
$ds.DisplayLevel = "On";
$ds.Update();
```

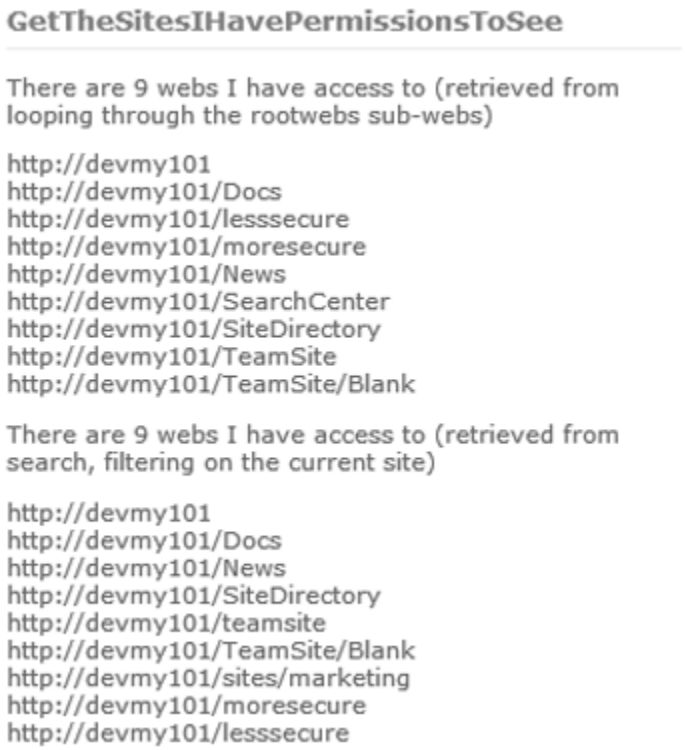
Testing the Example Webpart

To test the performance differences, we will run the following tests.
1. User A, with the webpart on a site collection with 9 sites (all webs are indexed)
2. User A, with the webpart on a site collection with 54 sites (some webs are NOT indexed)
3. User B, who has more restricted permissions than User A, with the webpart on a site collection with 54 sites (some webs are NOT indexed)

Each test will be run three times (by refreshing the page), the results (time taken for each method) will be aggregate to produce an average time.

Test 1: User A, on a site collection with 9 sites (all webs are indexed)

The output from the webpart shows both methods return the same number of sites.



This image shows part of the Developer Dashboard output. Using it, we can see the time taken for various parts of the page to load. The highlighted section shows the time taken to execute the two methods we wrapped in the SPMonitoredScope blocks.
You can see immediately that the search method is much faster, even on a small site collection.

The results from the first page refresh.

The results from the second page refresh.

The results from the third page refresh.

From the five screen shots above, we can see that both methods returned the same number of sites, and the differences in the time taken by each method.

Average time for the Looping Method to generate the result set: **84.3ms** (93.68, 90.96, 68.20)
Average time for the Search Method to generate the result set: **23.37ms** (25.62, 22.04, 22.45)

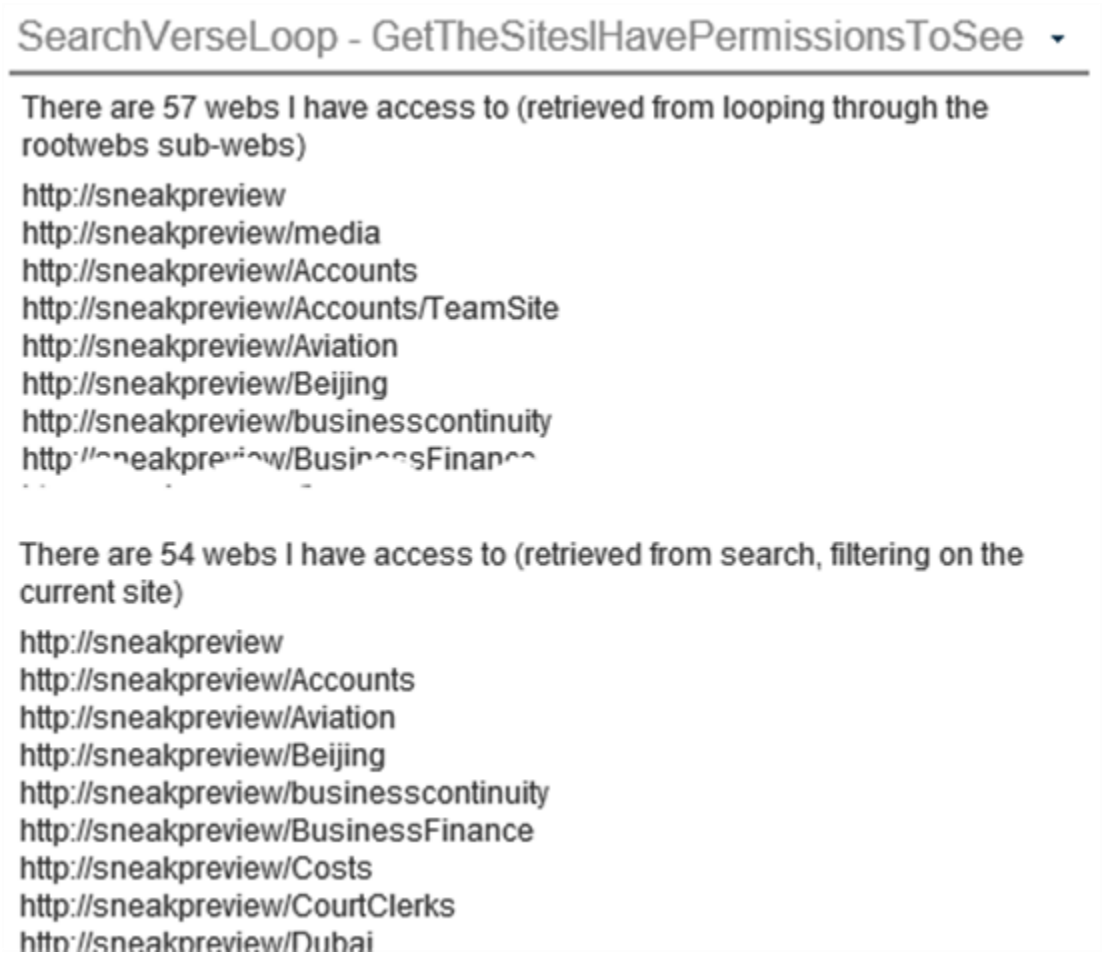
From this test, we can already see that using search is considerably faster, even though we are dealing with a small site collection.

Test 2: User A, with the Webpart on a Site Collection with 57 Sites (some Webs are not Indexed)

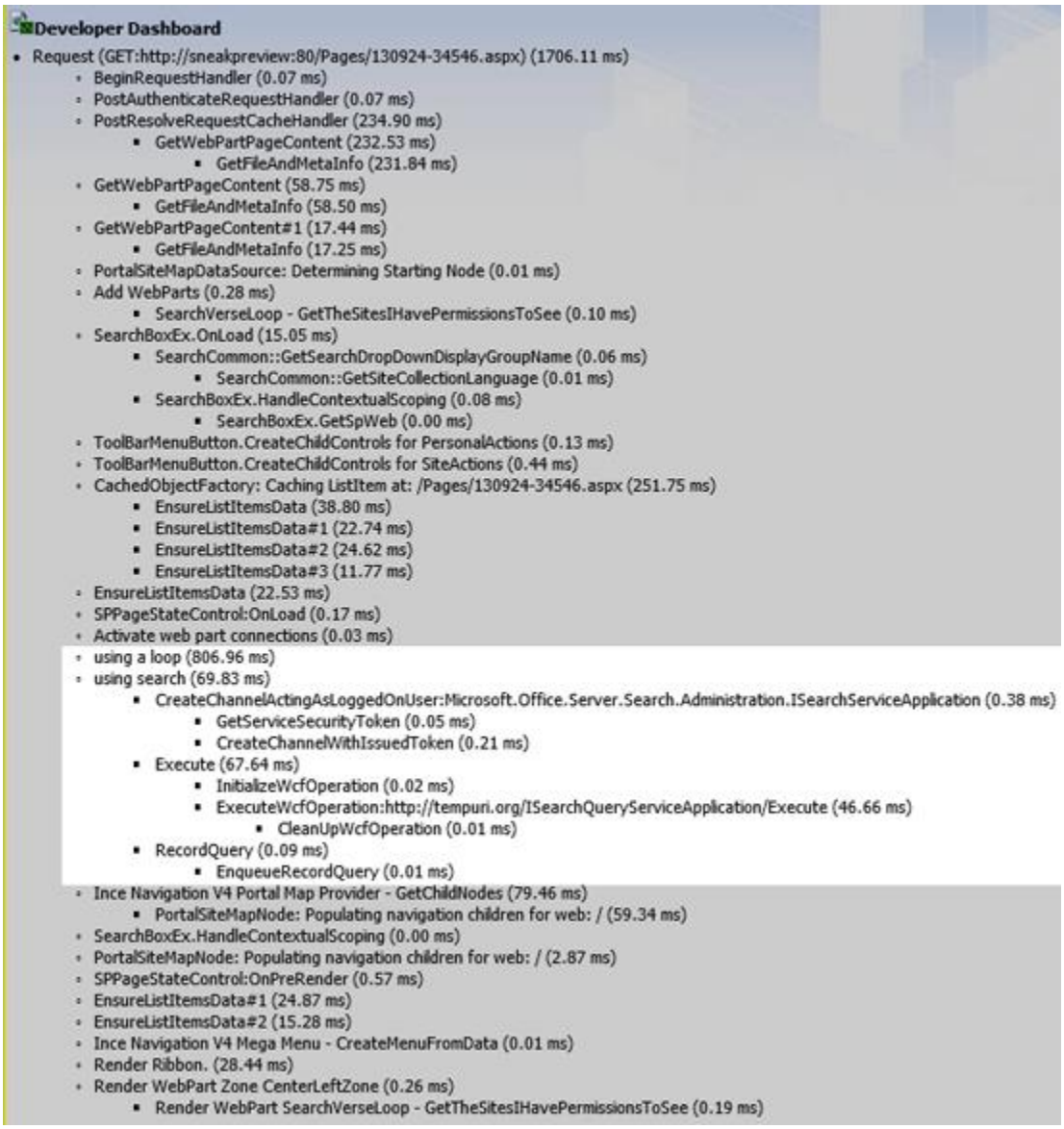
From the output of the webpart you can see there is a difference in the number of sites returned. The loop method returns 57 webs, but the search method only returns 54 webs.

The difference in the search results is mainly down to a setting on an SPWeb that controls if the site is included in the search index. That setting, "Allow this site to appear in search results?", is set via the site settings page of a site (SPWeb). For example: The "search center" (http://sneakpreview/searchcenter) is not returned in the result set, as this site is excluded from appearing in search results.

This is one of the caveats of using the search method, and could be seen as either a dis-benefit, or a benefit.



Looking at the Developer Dashboard for this test, we can see that the Search method clearly out-performs the loop based method. Test 2 has approximately 6 times the number of sites to retrieve. Using search, the query takes about 3 times longer than it did in the first example. Using looping, the query takes nearly 10 times longer than it did in the first test. Ouch!



The results from the first page refresh.

- using a loop (806.96 ms)
- using search (69.83 ms)
    - CreateChannelActingAsLoggedOnUser:Microsoft.Office.Server.Search.Administration.ISearchServiceApplication (0.38 ms)
        - GetServiceSecurityToken (0.05 ms)
        - CreateChannelWithIssuedToken (0.21 ms)
    - Execute (67.64 ms)
        - InitializeWcfOperation (0.02 ms)
        - ExecuteWcfOperation:http://tempuri.org/ISearchQueryServiceApplication/Execute (46.66 ms)
            - CleanUpWcfOperation (0.01 ms)
    - RecordQuery (0.09 ms)
        - EnqueueRecordQuery (0.01 ms)

The results from the second page refresh.

- using a loop (786.25 ms)
- using search (60.80 ms)
    - CreateChannelActingAsLoggedOnUser:Microsoft.Office.Server.Search.Administration.ISearchServiceApplication (0.40 ms)
        - GetServiceSecurityToken (0.06 ms)
        - CreateChannelWithIssuedToken (0.21 ms)
    - Execute (58.92 ms)
        - InitializeWcfOperation (0.02 ms)
        - ExecuteWcfOperation:http://tempuri.org/ISearchQueryServiceApplication/Execute (56.92 ms)
            - CleanUpWcfOperation (0.01 ms)
    - RecordQuery (0.07 ms)
        - EnqueueRecordQuery (0.01 ms)

The results from the third page refresh.

- using a loop (815.11 ms)
- using search (57.91 ms)
    - CreateChannelActingAsLoggedOnUser:Microsoft.Office.Server.Search.Administration.ISearchServiceApplication (0.38 ms)
        - GetServiceSecurityToken (0.05 ms)
        - CreateChannelWithIssuedToken (0.21 ms)
    - Execute (55.98 ms)
        - InitializeWcfOperation (0.02 ms)
        - ExecuteWcfOperation:http://tempuri.org/ISearchQueryServiceApplication/Execute (53.86 ms)
            - CleanUpWcfOperation (0.01 ms)
    - RecordQuery (0.09 ms)
        - EnqueueRecordQuery (0.01 ms)

From the five screen shots above, we can see that both methods returned approximately the same number of sites, and the different time taken by each method. Looping returns all 57 sites the user has access to, while the Search method returns 54 sites (because some sites are excluded from the Search Index).

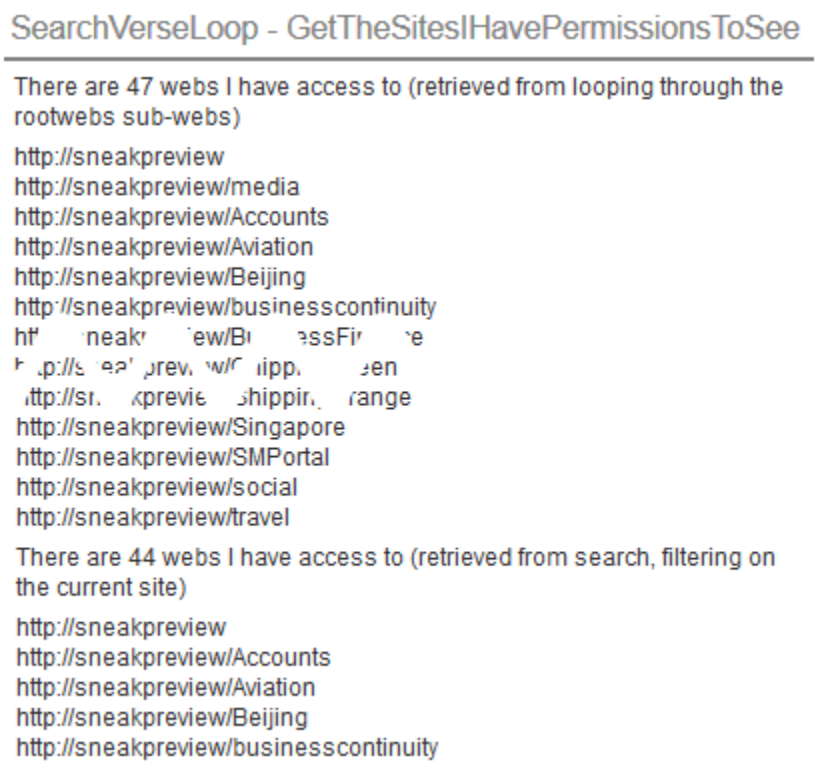
Average time for the Looping Method to generate the result set: **803.8ms** (806.64, 806.96, 786.25, 815.11)
Average time for the Search Method to generate the result set: **64ms** (69.83, 67.64, 60.80, 57.91)

In this case, where we are searching a slightly larger site collection, the differences in performance are very noticeable!

Test 3: User B, who has more Restricted Permissions than User A, with the Webpart on a Site Collection with 57 sites (some Webs are not Indexed)

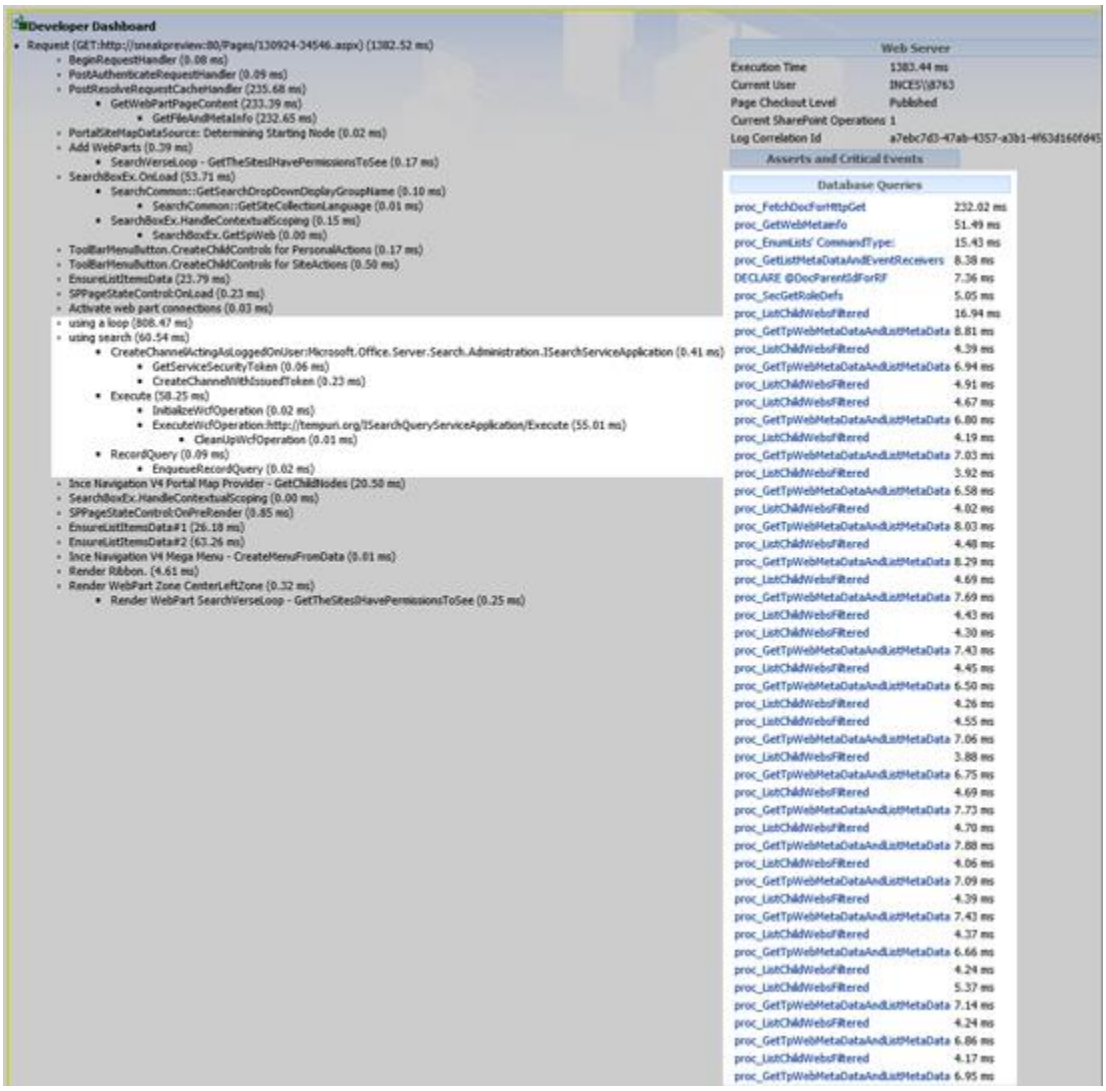
In this test, we focus on some other performance statistics that are highlighted by the Developer Dashboard.

We can see that this user has access to fewer sites than the user used in the previous test (47 sites, as compared with 57 sites for the user in Test 2). While the results are similar to Test 2 (the Search method returns 3 fewer results than the looping method, and the performance time statistics are similar), we want to look at what else is going on behind the scenes.



If we have a closer look at the Developer Dashboard's output, under the Database Queries, we can that the Looping method (calling SPWeb.GetSubwebsForCurrentUser()) makes two calls to the SQL database (proc_ListChildWebsFiltered, and

proc_GetTpWebMetaDataAndListMetaData) for each Web that is checked.



Further down the Developer Dashboard page, we have the SPRequest Allocations listed. Here we can see that the Looping method (which calls the SPWeb.GetSubwebsForCurrentUser()) creates an SPRequest allocation for each web that is checked.

SPRequest Allocations

SPWeb: http://sneakpreview/Pages/130924-34546.aspx
SPWeb: http://sneakpreview/media
SPWeb: http://sneakpreview/Accounts
SPWeb: http://sneakpreview/Accounts/TeamSite
SPWeb: http://sneakpreview/Aviation
SPWeb: http://sneakpreview/Beijing
SPWeb: http://sneakpreview/businesscontinuity
SPWeb: http://sneakpreview/BusinessFinance
SPWeb: http://sneakpreview/Costs
SPWeb: http://sneakpreview/Costs/TeamSite
SPWeb: http://sneakpreview/CourtClerks
SPWeb: http://sneakpreview/CourtClerks/TeamSite
SPWeb: http://sneakpreview/Dubai
SPWeb: http://sneakpreview/EnergyAndOffshore
SPWeb: http://sneakpreview/Hamburg
SPWeb: http://sneakpreview/HongKong
SPWeb: http://sneakpreview/HKMarineDB
SPWeb: http://sneakpreview/HumanResources
SPWeb: http://sneakpreview/HumanResources/TeamSite
SPWeb: http://sneakpreview/Incisive
SPWeb: http://sneakpreview/InformationTechnology
SPWeb: http://sneakpreview/InformationTechnology/Development
SPWeb: http://sneakpreview/InformationTechnology/Help
SPWeb: http://sneakpreview/InformationTechnology/TeamSite
SPWeb: http://sneakpreview/Insurance
SPWeb: http://sneakpreview/ITC
SPWeb: http://sneakpreview/KnowledgeManagement
SPWeb: http://sneakpreview/LatinAmericaGroup
SPWeb: http://sneakpreview/LeHavre
SPWeb: http://sneakpreview/learning
SPWeb: http://sneakpreview/Library
SPWeb: http://sneakpreview/Library/TeamSite
SPWeb: http://sneakpreview/LitigationSupport
SPWeb: http://sneakpreview/LitigationSupport/TeamSite
SPWeb: http://sneakpreview/Lloyds
SPWeb: http://sneakpreview/London
SPWeb: http://sneakpreview/Marketing
SPWeb: http://sneakpreview/Marketing/TeamSite
SPWeb: http://sneakpreview/Monaco
SPWeb: http://sneakpreview/officeservices
SPWeb: http://sneakpreview/officeservices/TeamSite
SPWeb: http://sneakpreview/Paris
SPWeb: http://sneakpreview/Piraeus
SPWeb: http://sneakpreview/riskandcompliance
SPWeb: http://sneakpreview/riskandcompliance/riskregister
SPWeb: http://sneakpreview/SearchCenter
SPWeb: http://sneakpreview/Secretarial
SPWeb: http://sneakpreview/Secretarial/TeamSite
SPWeb: http://sneakpreview/Shanghai
SPWeb: http://sneakpreview/Shipping
SPWeb: http://sneakpreview/ShippingBlue
SPWeb: http://sneakpreview/ShippingGreen
SPWeb: http://sneakpreview/ShippingOrange
SPWeb: http://sneakpreview/Singapore
SPWeb: http://sneakpreview/SMPortal

Test Summary

From the three tests above, the Search method out performs the Looping method, and uses less resources in doing so. This makes the search method more scale-able, both in terms of simultaneous users loading the page, and in terms of how large the site collection can be.

The caveat to the Searching method is that the result set might not include all of the sites a user has access to, if one or more sites has been excluded from the Search Index. This may or may not be a problem, depending on why the sites have been excluded from the search index.

The looping method puts more load on the SharePoint infrastructure, and performance issues are bound to occur as the number of users using the code (or webpart) increases and/or the number of sites in a site collection increases.

Quick Summary of Pros and Cons

Search Pros

- It's fast
- It can handle a very large site collection, returning results very quickly

Search Cons

- If a site has the "Allow this site to appear in search results?" set to No, the site won't be returned in the search results. This could be a pro (in some scenarios) or a con.
- There are limited properties that can returned about an SPWeb object using Search. If you need to query additional properties, for example a property from the SPWeb.Properties collection, you would need to use the looping method.

Loop (iteratively calling GetSubwebsForCurrentUser) Pros

- You can query additional properties of each SPWeb object as you parse the collection of webs the user has access to. For example, you could query a custom property from the SPWeb.Properties collection.

Loop (iteratively calling GetSubwebsForCurrentUser) Cons

- As the number of webs in a site collection increases, the performance becomes a big issue, causing the page to load slower.
- Make calls to SPWeb.GetSubwebsForCurrentUser() increases load on the SQL server. This could cause a performance problem (albeit, depending on the size of your environment, number of webs in the site collection and the frequency in which the code is called).
- Creates a lot of SPRequest allocations.

See Also

- KeywordQuery
- SPWeb.GetSubwebsForCurrentUser()
- Using the Developer Dashboard
- Using SPMonitoredScope
- SPMonitoredScope

References

- Original content from Matthew Yarlett's Blog
- Forum Content (Retrieve List Of Sites)
- Forum Content (how to get the login user details from subsites)

SharePoint 2013: The SEO Friendly Site

Introduction

Search engines rely heavily on page keywords and meta descriptions to rank your web site. SharePoint 2013 helps you make your site pages SEO friendly, by providing an out-of-the-box mechanism to output your meta keywords and description.

Enabling Keywords and Meta Descriptions

On his blog , Yaroslav Pentsarskyy shared an interesting 5-step method to Enable Keywords & Meta Descriptions on Pages.

"When content authors create or edit and existing page, they can specify meta keywords for that page right there and then. Assuming you have a site of publishing site template,

1. *As a site collection administrator, navigate to the Site settings -> Site collection features*
2. *Activate Search Engine Sitemap feature*
3. *Navigate to an existing or a new page and enter its edit mode*
4. *From the Page ribbon tab, click on the Edit Properties fly-out and select Edit SEO Properties*
5. *Check additional settings*
    a. *Browser title*
    b. *Keywords*
    c. *Exclude from internet Search engines*"

Fields are rendered on the page, using the following meta tags.

```
<meta name="description" content="Sunny's meta description" />
<meta name="keywords" content="Sunny's keywords" />
```

See Also

- How To: improve SharePoint/Office 365 site SEO with page keywords and meta description

SharePoint 2013: SharePoint Community Site as Real Time Social Communities or Groups

Introduction

One of my friends asked me how admin group from organization can create, delete or modify the communities. His requirement was the following: he wanted to create a social application on top of SharePoint 2013 where admins can create as many communities or groups for various departments, people; at the same time he also wanted to provide community management feature to them so that admin can delete, purge or modify the community. By default these features are available there, only thing is: you need to exploit them to best as per your requirement. Also by default SharePoint has provided web forms to create, delete or manage the Communities though they are here and there and you want everything at one place.

Approaches

You can do this by following some of the ways:

With One community site

If the usage of your social application is very limited or less then create one community site and create multiple discussions threads and categorized them with proper category. Here category will act as a boundary and you will have felt like there are different groups or social communities. But the headache here is you need to put custom layer of access; so that users can see intended discussions only. Still this is an option, although I don't recommend it.

With One web application, Site collection and multiple communities (sub sites)

In this option you need to create a web application and a site collection; site collection should have Community template available to create Community sub sites. So my idea

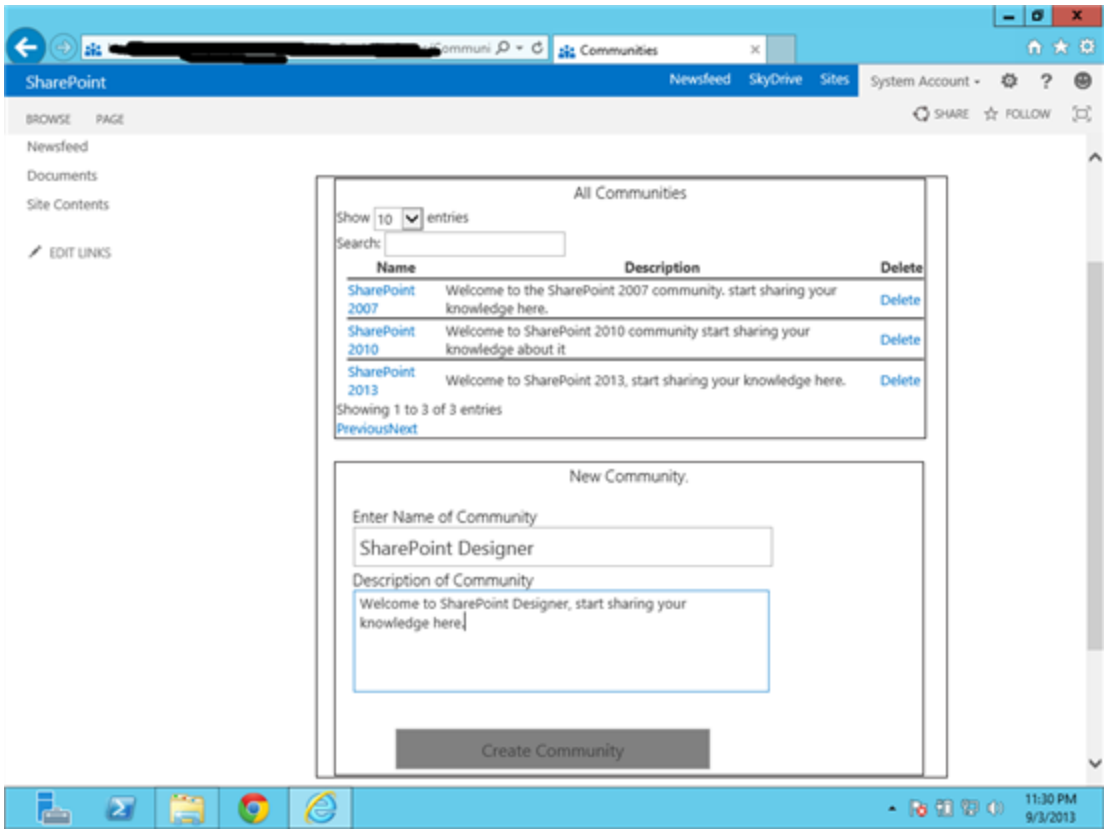
here is the root web will act as the landing site from where users can see multiple communities. Don't forget that Communities itself are nothing but sub sites. And you can create as many sub sites in SharePoint site provided that your FARM supports (performance wise). You will manage all communities from the root site. In this approach communities or social groups are well separated; you have option of unique permissions, you can give, take permission of users, managing individual community is also easy.

With One web application, multiple communities (site collections)

In this approach you need to create a web application and you will create many communities as site collection. The idea is the same as option 2 above, the only difference is here communities are site collections.

So after understanding SharePoint 2013 community sites I have concluded this; SharePoint experts may have different views and ideas as compare to above one.

So now we will see how you can use option No. 2 and create a social application for your organisation. In this example I have created one web application and a site collection (root web). And it will have all communities and a page where admins can manage all communities; this script is very basic which provides option to see all existing communities and admins can create them. Below screenshot depicts how it looks.



Example Approach 2

Steps for creating social site using option 2:

1. Create web application from the central administration.
2. Create root site collection.
3. Create a community.html file by using below code snippet.

```
<script src="/SiteAssets/js/jquery.js"></script>
<script src="/js/jquery.dataTables.min.js"></script>
<script type="text/javascript">

    var siteTitle;
    var siteTemplate; //template for Community.
    var siteURL;
    var sitePermission ;
    var siteDescription;
    var allwebs, allwebsToShow;
    function CreateSite() {
        siteTitle =  $("#siteTitle").val();
        siteTemplate = "COMMUNITY#0"; //template for Community.
```

```
        siteURL = siteTitle.replace(" ","");
        sitePermission = true;
        siteDescription = $("#siteDesc").val();

        var clientContext = new SP.ClientContext.get_current();
        this.Web = clientContext.get_web(); //.get_current();

        var webInfo = new SP.WebCreationInformation();
        webInfo.set_webTemplate(siteTemplate);
        webInfo.set_description(siteDescription);
        webInfo.set_title(siteTitle);
        webInfo.set_url(siteURL);
        webInfo.set_language("1033");
        webInfo.set_useSamePermissionsAsParentSite(sitePermissio
n);

        allwebs = this.Web.get_webs();
        //allwebs.add(webInfo);

        clientContext.load(this.Web);
        clientContext.load(allwebs);

        clientContext.executeQueryAsync(Function.createDelegate(
this, this.onSuccess),
                Function.createDelegate(this, this.onFail));

    }

    function onSuccess(sender, args) {
        alert("Community Created successfully.");
        $("#siteTitle").val("");
        $("#siteDesc").val("");
    }

    function onFail(sender, args) {
        alert('Failed:' + args.get_message());
    }

    $(document).ready(function () {
        ExecuteOrDelayUntilScriptLoaded(GetAllCommunities,
"sp.js");
    });


    function GetAllCommunities() {
```

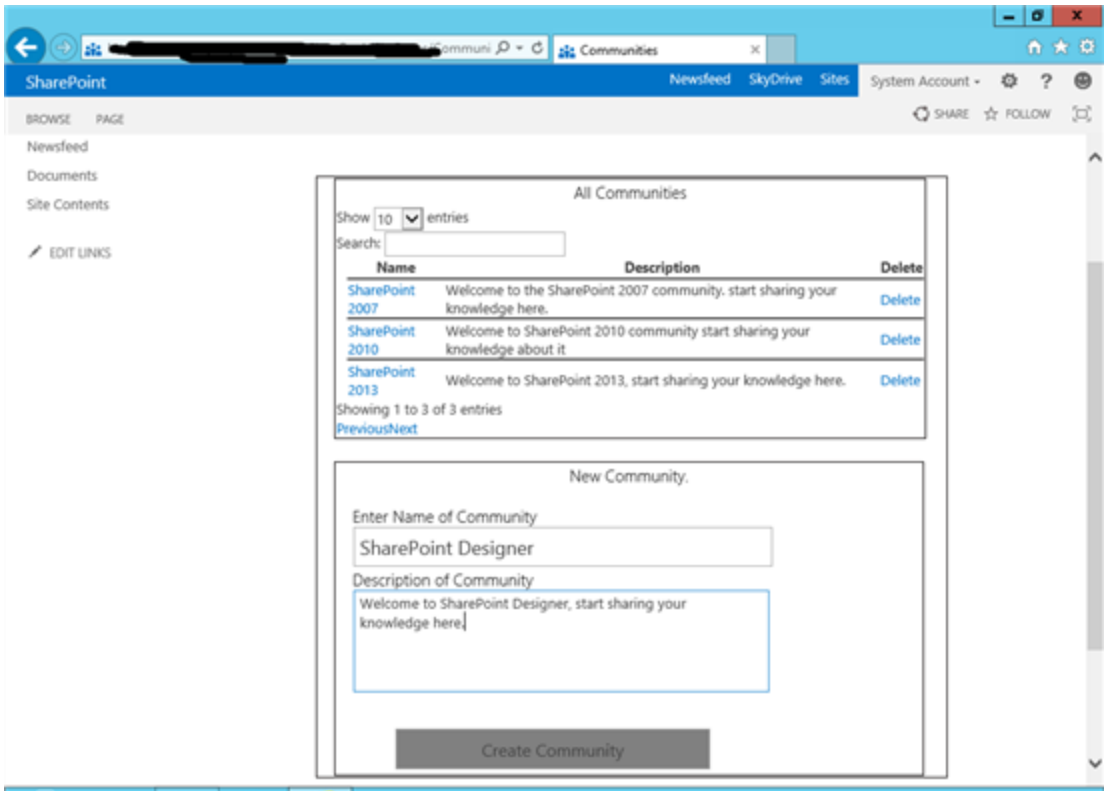
```
        debugger;
        var clientContext = new SP.ClientContext.get_current();
        this.Web = clientContext.get_web();
        allwebsToShow = this.Web.get_webs();

        clientContext.load(this.Web);
        clientContext.load(allwebsToShow);

        clientContext.executeQueryAsync(Function.createDelegate(
this, this.onSuccessAllWebs),
                Function.createDelegate(this,
this.onFailAllWebs));
    }

    function onSuccessAllWebs(sender, args) {

        var htmlStart = "<table id='communityTable' style='margin-
left:10px;'><thead><tr><th>Name</th><th>Description</th><th>Dele
te</th></tr></thead>"
        var htmlEnd = "</table>"
        var htmlstr = "";
        for (var i = 0 ; i < allwebsToShow.get_count() ; i++)
{
            if (allwebsToShow.get_item(i).get_webTemplate() ==
"COMMUNITY") {

                htmlstr = htmlstr + "<tr style='border-top:1pt
solid black;'><td><a href='" +
allwebsToShow.get_item(i).get_url() + "'>" +
allwebsToShow.get_item

(i).get_title() + "</a></td><td>" +
allwebsToShow.get_item(i).get_description() + "</td><td> "+
"<ahref='javascript:alert('Delete')'>Delete</a>" + "</td></tr>"
            }
        }

        $("#allCommunities").html(htmlStart + htmlstr +
htmlEnd);

        $('#communityTable').dataTable();

    }
```

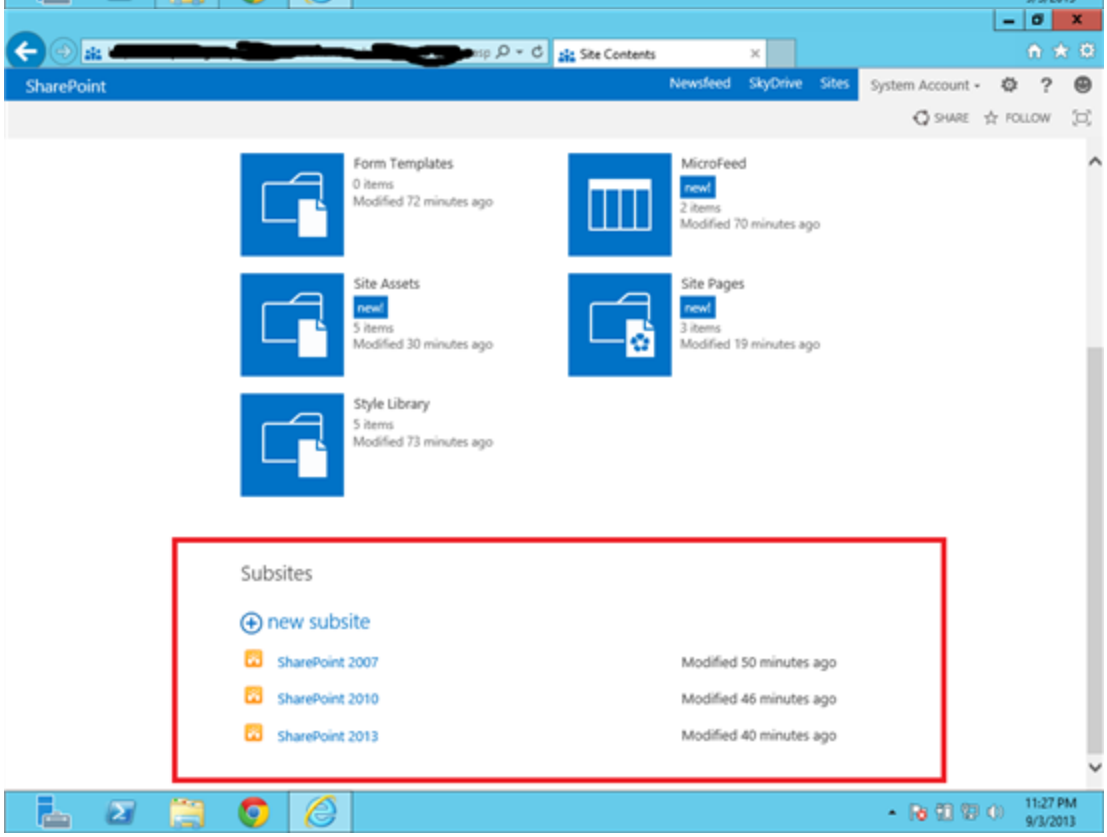
```
    function onFailAllWebs(sender, args) {
        alert('Failed:' + args.get_message());
    }

</script>

    <style type="text/css">
        #btnCreate {
            height: 38px;
            width: 293px;
            font-size: medium;
            background-color: #808080;
            text-align: center;
        }
        #siteTitle {
            height: 31px;
            width: 378px;
            font-size: large;
        }
        .auto-style1 {
            padding: 0px;
            width: 498px;
        }
        .auto-style2 {
            padding-top: 0px;
            width: 498px;
        }
        .auto-style3 {
            height: 44px;
            padding-top:30px;
            width: 498px;
        }
        #siteDesc {
            font-size: small;
            width: 375px;
            height: 89px;
        }
        .auto-style6 {
            width: 405px;
        }
        .auto-style7 {
            height: 12px;
        }
    </style>
<div style="border:solid 1px #070303;width: 586px;">
    <table style="width: 574px">
```

```
        <tr>
            <td>
                <table style="border:1px solid #070303;padding-
left:25px;margin-left:15px; width: 551px;">
                    <tr>
                        <th class="auto-style6">
                            <h3>All Communities</h3>
                        </th>
                    </tr>
                    <tr>
                        <td class="auto-style6">
                            <div id="allCommunities" style="width:
100%">

                            </div>
                        </td>
                    </tr>
                </table>
            </td>
        </tr>
        <tr>
            <td class="auto-style7">

            </td>
        </tr>
        <tr>
            <td>
                <table style="border:1px solid #070303;padding-
left:25px;margin-left:15px; width: 549px;">
                    <tr>
                        <th class="auto-style6">
                            <h3>New Community.
                        </h3>
                        </th>
                    </tr>
                    <tr>
                        <td class="auto-style6">
                            <table style="padding-
left:25px;margin-left:15px; width: 517px;">
                            <tr>
                                <td class="auto-
style1" style="padding-top:15px;">
                                    <h3>Enter Name of Community
</h3>
```

```
                                    </td>
                                </tr>
                                <tr>
                                    <td class="auto-style2">
                                        <input type="text" id="siteTit
le" />
                                    </td>
                                </tr>
                                <tr>
                                    <td class="auto-style1">
                                        <h3>Description of Community
</h3>
                                    </td>
                                </tr>
                                <tr>
                                    <td class="auto-style2">
                                        <textarea id="siteDesc"></tex
tarea>
                                    </td>
                                </tr>
                                <tr>
                                    <td class="auto-
style3" style="padding-left:30px;">
                                        <input type="button" id="btnCr
eate" value="Create Community"onclick="CreateSite();" />
                                    </td>
                                </tr>
                            </table>
                            </td>
                        </tr>
                    </table>
                </td>
            </tr>
        </table>

</div>
```

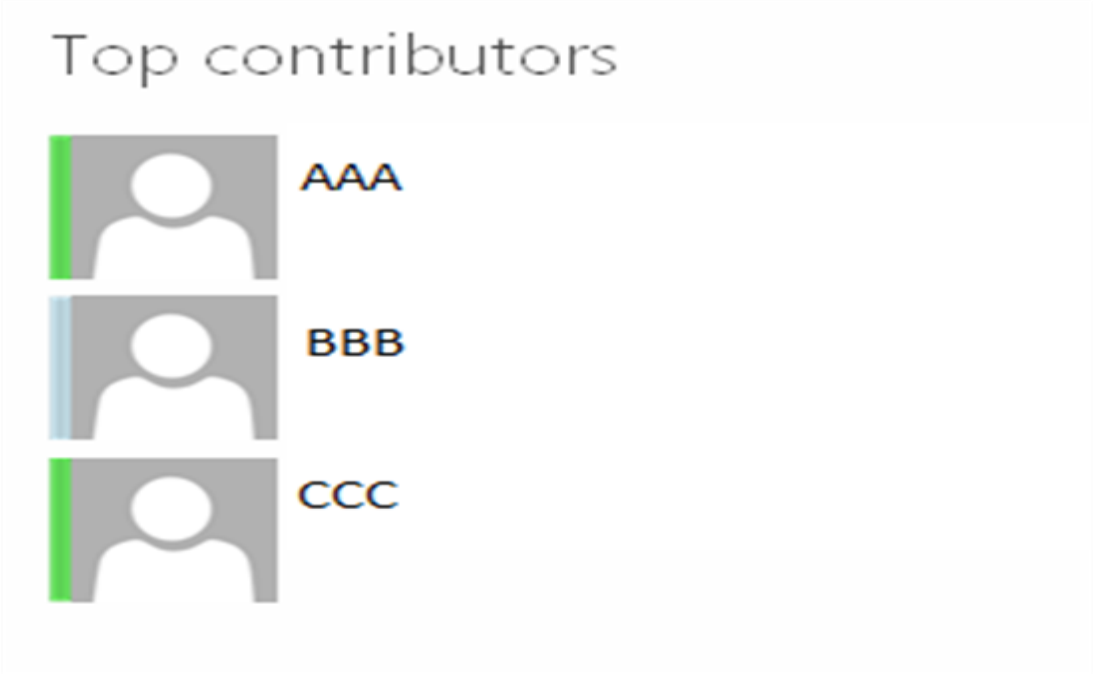
4. Upload this file to **Site Asset library.**
5. **Open** the home page of site collection in edit mode.
6. Add content editor web part.
7. Provide the **URL** of 'community.html' to the content editor web part.
8. **Save** and apply settings.


Communities
SharePoint
Newsfeed
SkyDrive
Sites
System Account
BROWSE
PAGE
SHARE
FOLLOW
Newsfeed
Documents
Site Contents
EDIT LINKS
All Communities
Show 10 entries
Search:
Name
Description
Delete
SharePoint 2007
Welcome to the SharePoint 2007 community. start sharing your knowledge here.
Delete
SharePoint 2010
Welcome to SharePoint 2010 community start sharing your knowledge about it
Delete
SharePoint 2013
Welcome to SharePoint 2013, start sharing your knowledge here.
Delete
Showing 1 to 3 of 3 entries
PreviousNext
New Community.
Enter Name of Community
SharePoint Designer
Description of Community
Welcome to SharePoint Designer, start sharing your knowledge here.
Create Community
11:30 PM
9/3/2013


Site Contents
SharePoint
Newsfeed
SkyDrive
Sites
System Account
SHARE
FOLLOW
Form Templates
0 items
Modified 72 minutes ago
MicroFeed
new!
2 items
Modified 70 minutes ago
Site Assets
new!
5 items
Modified 30 minutes ago
Site Pages
new!
3 items
Modified 19 minutes ago
Style Library
5 items
Modified 73 minutes ago
Subsites
new subsite
SharePoint 2007
Modified 50 minutes ago
SharePoint 2010
Modified 46 minutes ago
SharePoint 2013
Modified 40 minutes ago
11:27 PM
9/3/2013

Consolidated Top Contributors of Communities

Everyone including administrator wants to see the top contributors of community. In some organization it is used as a parameter to recognize and reward a person. So how will you get top contributors? Yes, SharePoint 2013 already provides this OOB using member's lists view. You can see top contributors for community and this is specific to that community only and it is calculated on the basis of Reputation Score/points earned by members.



Now what if you want to see consolidated top Contributors amongst all communities? As in your social application, there are number of communities and members have joined more than one community. And now you want to find out top contributor amongst all communities, now how you can achieve it?

So here I have write a script which gives you consolidated top contributors and displays on the screen with their Reputation score, Name, number of replies and number of post.

```
    <style type="text/css">
        .auto-style1 {
            width: 169px;
        }
    </style>

<script src="/jquery-1.7.1.min.js"></script>
```

```
<script src="/jquery.dataTables.min.js"></script>
<script type="text/javascript">

    var allSitesToShow;
    var ArrayOfMembers = new Array();
    var ArrayOfListItemCollection = new Array();
    var context;

    $(document).ready(function () {
        ExecuteOrDelayUntilScriptLoaded(GetAllCommunitySites,
"sp.js");
    });

    function GetAllCommunitySites() {
        debugger;
        context = new SP.ClientContext.get_current();
        this.Web = context.get_web();
        allSitesToShow = this.Web.get_webs();

        context.load(this.Web);
        context.load(allSitesToShow);

        context.executeQueryAsync(Function.createDelegate(this,
this.onSuccessAllCommunitySites),
                Function.createDelegate(this,
this.onFailAllWebs));
    }

    function onSuccessAllCommunitySites(sender, args) {
        debugger;

        var camlQuery = new SP.CamlQuery();
        //var query = '<View/>';
        var query =
"<View><Query><OrderBy><FieldRef Name='ReputationScore' Ascending=
'False'></FieldRef></OrderBy></Query><RowLimit>5</RowLimit></Vie
w>";
        camlQuery.set_viewXml(query);
        context = new SP.ClientContext.get_current();
        for (var i = 0 ; i < allwebsToShow.get_count() ; i++) {
            if (allwebsToShow.get_item(i).get_webTemplate() ==
"COMMUNITY") {

                //var context = new SP.ClientContext.get_current()
;
```

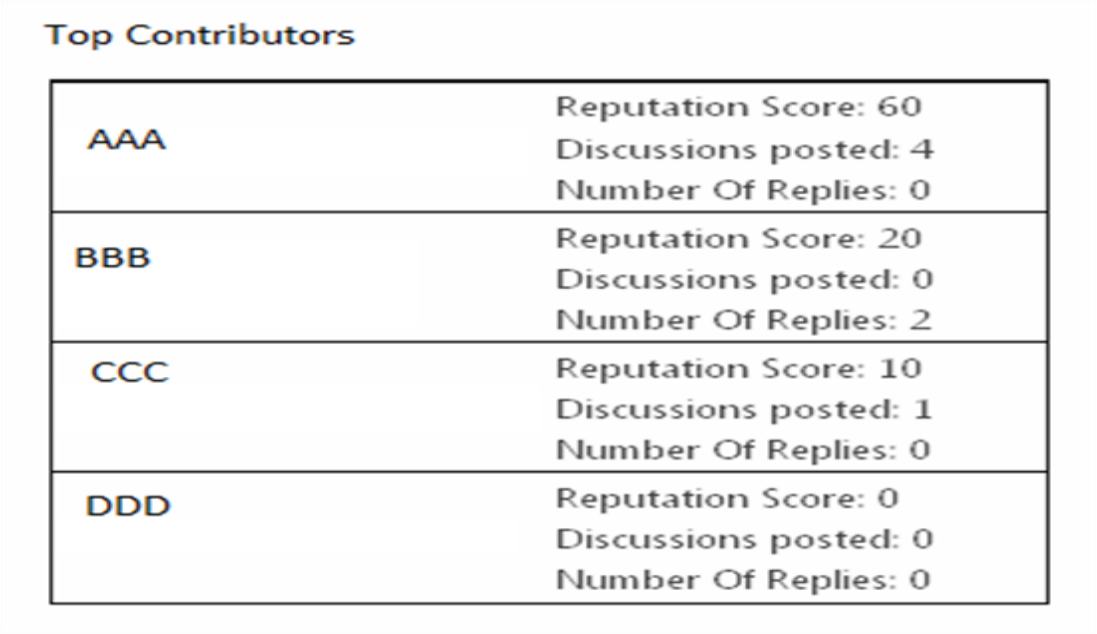
```
                var list = allwebsToShow.get_item(i).get_lists().
getByTitle("Community Members");
                var listItems_1 = list.getItems(camlQuery);
                ArrayOfListItemCollection.push(listItems_1);
                context.load(listItems_1);
            }
        }
        context.executeQueryAsync(Function.createDelegate(this,
this.onSuccessListItems), Function.createDelegate(this,
this.onFailAllWebs));
    }

    function onSuccessListItems(sender, args) {
        debugger;
        var flag = false;
        for (var i = 0 ; i < ArrayOfListItemCollection.length ;
i++) {

            for (var j = 0 ; j <
ArrayOfListItemCollection[i].get_count() ; j++) {

                var Data ={
                    MemberName:
ArrayOfListItemCollection[i].get_item(j).get_fieldValues().Title
,
                    ReputationScore:
ArrayOfListItemCollection[i].get_item(j).get_fieldValues().Reput
ationScore,
                    NumberOfDiscussions:
ArrayOfListItemCollection[i].get_item(j).get_fieldValues().Numbe
rOfDiscussions,
                    NumberOfReplies:
ArrayOfListItemCollection[i].get_item(j).get_fieldValues().Numbe
rOfReplies,
                    NumberOfBestResponses:
ArrayOfListItemCollection[i].get_item(j).get_fieldValues().Numbe
rOfBestResponses,
                    LookupId:
ArrayOfListItemCollection[i].get_item(j).get_fieldValues().Membe
r.get_lookupId()
                }


                for (var k = 0 ; k < ArrayOfMembers.length ; k++)
{
```

```
                    if (ArrayOfMembers[k].MemberName ==
Data.MemberName)
{
                        flag = true;
                        break;
                    }
                }

                if (flag == true) {
                    ArrayOfMembers[k].ReputationScore = ArrayOfMe
mbers[k].ReputationScore + Data.ReputationScore;
                    ArrayOfMembers[k].NumberOfDiscussions = Array
OfMembers[k].NumberOfDiscussions + Data.NumberOfDiscussions;
                    ArrayOfMembers[k].NumberOfReplies = ArrayOfMe
mbers[k].NumberOfReplies + Data.NumberOfReplies;
                }
                else {
                    ArrayOfMembers.push(Data);
                }
                flag = false;
            }
            //alert(ArrayOfMembers);
        }

            ArrayOfMembers = ArrayOfMembers.sort(function (a, b)
{ return b.ReputationScore - a.ReputationScore });

            var htmlStart = "<table style='width:300px;'>"
            var htmlEnd = "</table>"
            var htmlstr = "";

            for (var i = 0 ; i < ArrayOfMembers.length ; i++) {
//ArrayOfMembers.length
                if (i == 5)
                    break;
                htmlstr = htmlstr + "<tr style='border-top:1pt
solid
black;'><tdstyle='width:140px;vAlign:Top;'><a href='/_layouts/15/
userdisp.aspx?ID=" + ArrayOfMembers[i].LookupId + "'>" +
ArrayOfMembers[i].MemberName +
"</a></td><td style='width:140px;'><table><tr><td> Reputation
Score: " + ArrayOfMembers[i].ReputationScore +
"</td></tr><tr><td> Discussions posted: " +
ArrayOfMembers[i].NumberOfDiscussions + "</td></tr><tr><td>
```

```
Number Of Replies: " + ArrayOfMembers[i].NumberOfReplies +
"</td></tr></table></td></tr>";
            }

            $("#contribDiv").html(htmlStart + htmlstr +
htmlEnd);

    }

    function onFailAllWebs(sender, args) {
        alert('Failed:' + args.get_message());
    }

</script>

<div>

    <div id="contribDiv" style="border:solid 1px #070303;">

    </div>

</div>
```

How does this script work?

This script is written for the scenario where approach 2 is used (please refer to the above example). So it first finds out all community sites. And then it collects all top 5 contributors from all community, while doing this it checks whether member has joined more than one community and sums the Reputation score of all community. In last step it sorts the member collection on the basis of reputation score and displays on the screen.



Conclusion

Now you are ready to use this page as management console for all communities. This is very basic and initial level of script. I am planning to add following features in it: configure permissions for users from this console, list out top communities, top contributors, health of communities, etc.

See Also

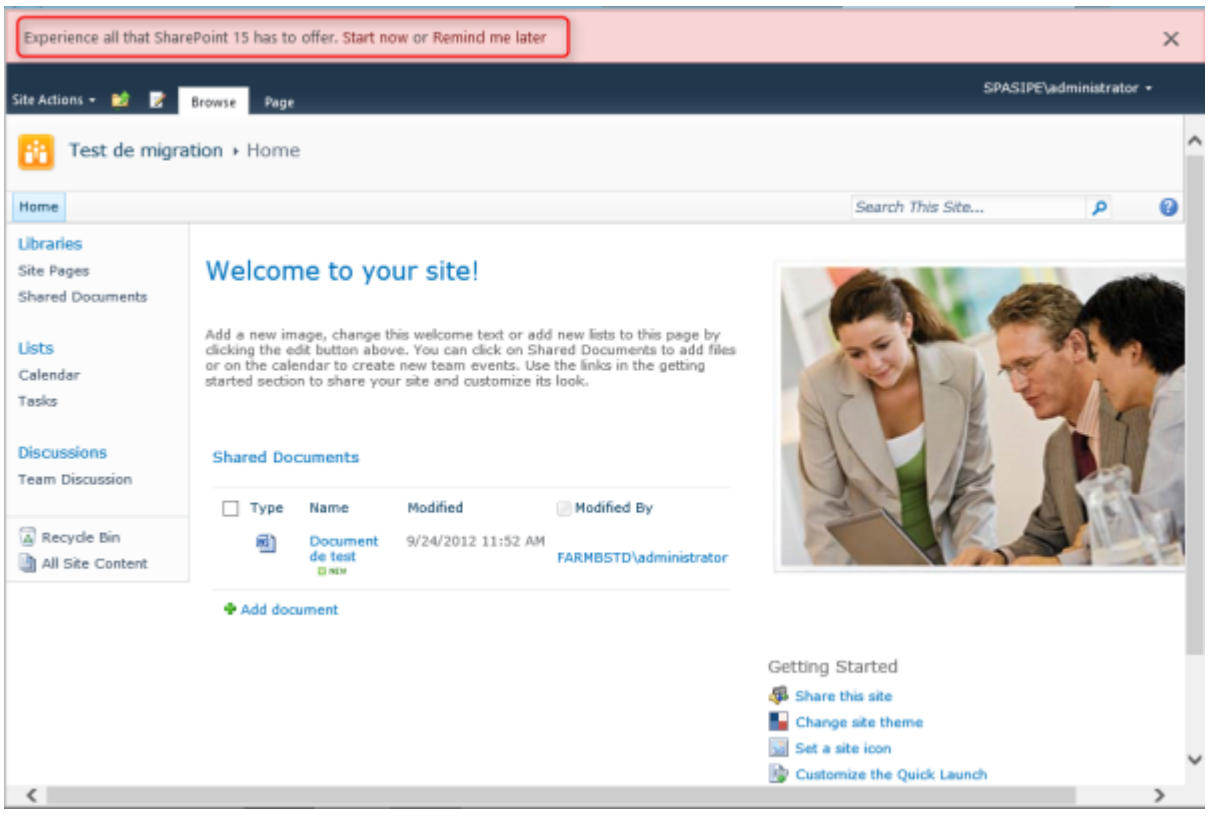
- SharePoint 2013: Social Features, Community Site.

SharePoint 2013: Upgrade a Site Collection (User Interface and PowerShell)

Introduction

This article will explain how to upgrade a site collection to SharePoint 2013, by using the user interface and PowerShell.

I consider here that my site collection has already been migrated to the SharePoint 2013 server and is in actually in "SharePoint 2010" mode, like this:



I divided the functionalities in different "Points", each of them having a * *PowerShell* * section which will detail the available modifications or interactions by using PowerShell.

Point 1: Self-service site upgrade

To access the self-service upgrade options, your site collection must be eligible, and must also respect 2 rules defined at the web application level:

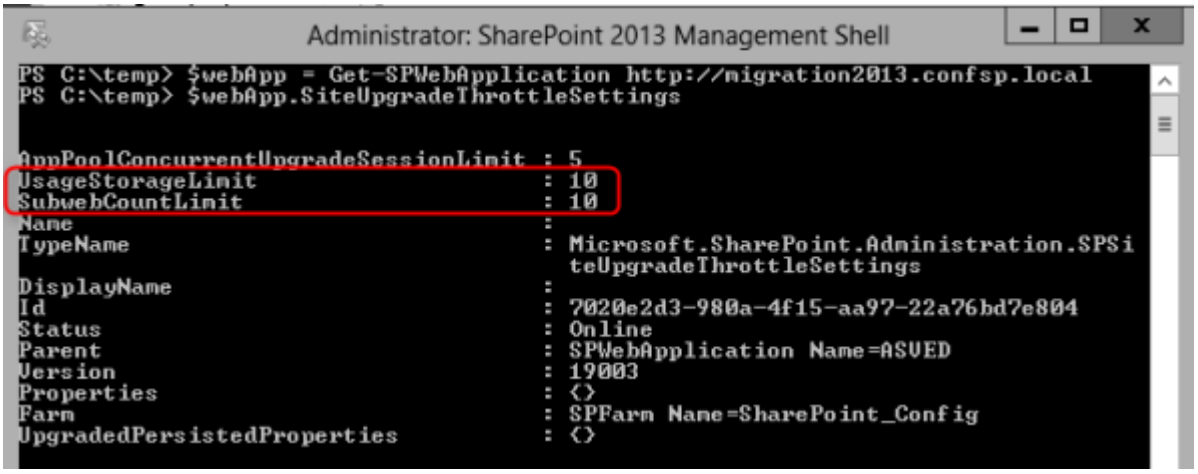
- A maximal size (10 MB by default),
- A maximum number of sites (10 by default).

Si one rule isn't respected, you had to migrate via PowerShell.

* *PowerShell* *

- Get the web application parameters

$webApp = Get-SPWebApplication < Web Application URL >
$webApp.SiteUpgradeThrottleSettings



- Modify web application parameters

$webApp = Get-SPWebApplication -URL < Web Application URL >

$webApp.SiteUpgradeThrottleSettings.UsageStorageLimit = <value>

$webApp.SiteUpgradeThrottleSettings.SubwebCountLimit = <value>

$webApp.Update()

Point 2: The notification bar

The notification bar at the top of the site displays a message indicating that the site can be migrated.

Two options are available:

- "Start now": Access to the page which allow to migrate the site or request an evaluation site,
- "Remind me later": Hide the notification bar, for 30 days (by default).

* *PowerShell* *

1. Prevent users from using the self-service upgrade:

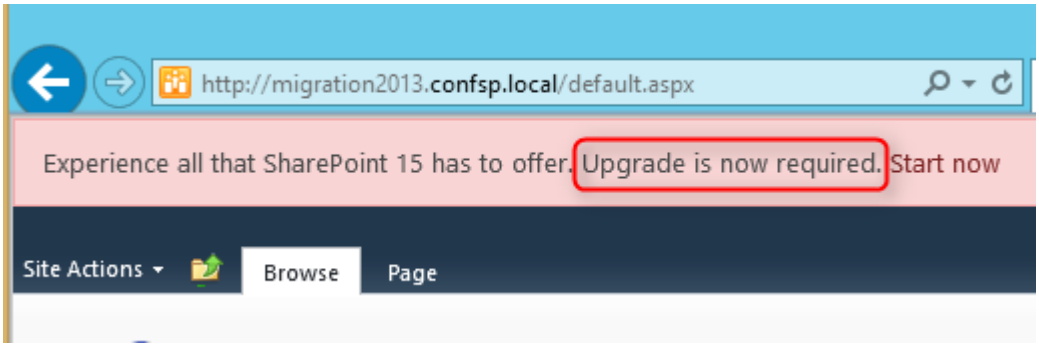
$site = Get-SPSite <URL>

$site.AllowSelfServiceUpgrade = $FALSE

2. Changer the reminder delay / force upgrade
$webApp = Get-SPWebApplication < Web Application URL >
$webApp.UpgradeReminderDelay <Number of days>
$webApp.Update();

If you set the "UpgradeReminderDelay" parameter to 0:

- The link to hide the notification bar is removed,
- The message indicates that the upgrade is now required.



Point 3: The Health Checks

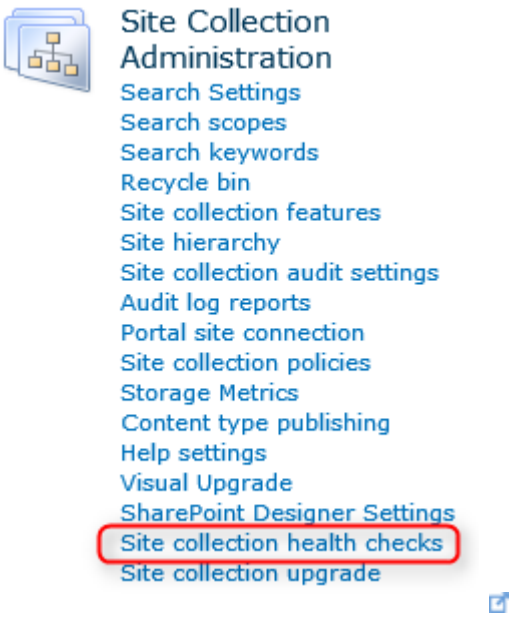
The "Health Checks" are a major step of the migration, because they will identify the potentials problems:

- Conflicting content types,
- Customized/unghosted files,
- Missing galleries,
- Missing parent content types,
- Missing site templates,
- Unsupported language pack references,
- Unsupported MUI (multi-user interface) references.

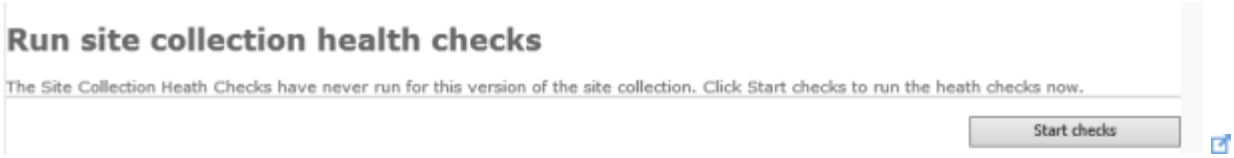
Note that these Health Checks can be executed before and after the site collection upgrade.

To execute the "Health Checks":

1. Click on "Site collection health checks" link in the "Site Settings":

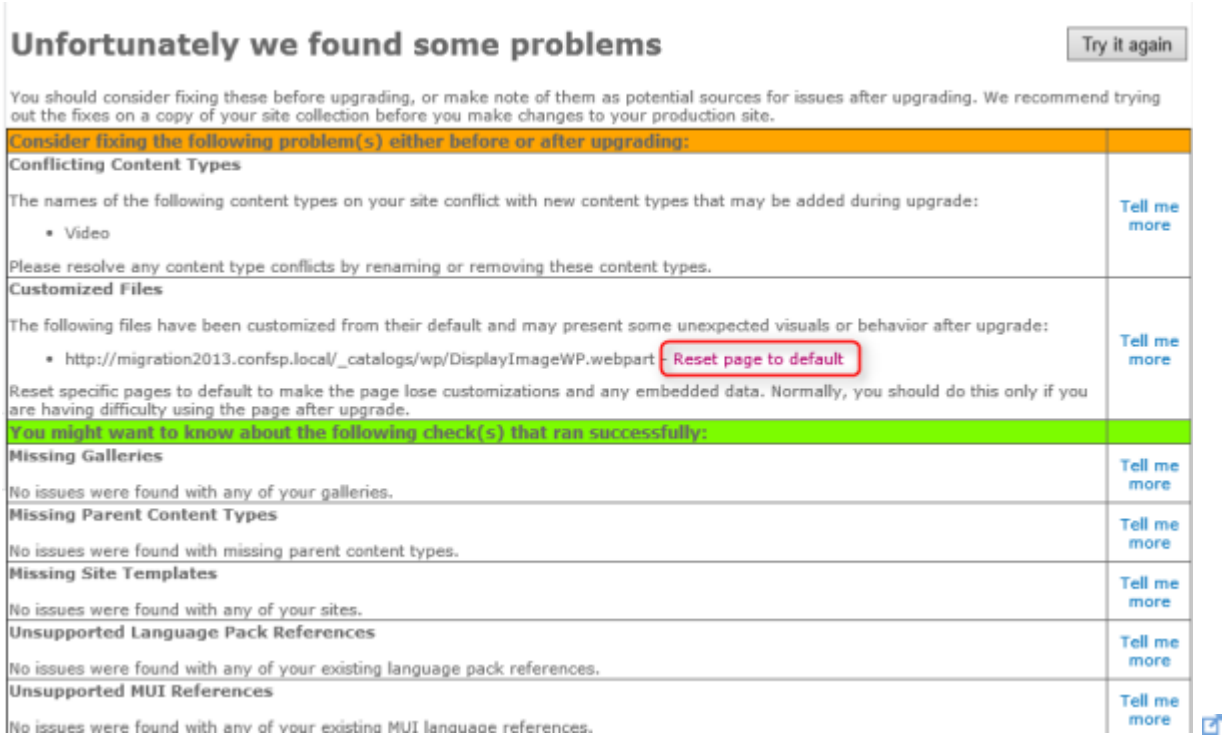


2. Click on "Start checks" to launch the process.

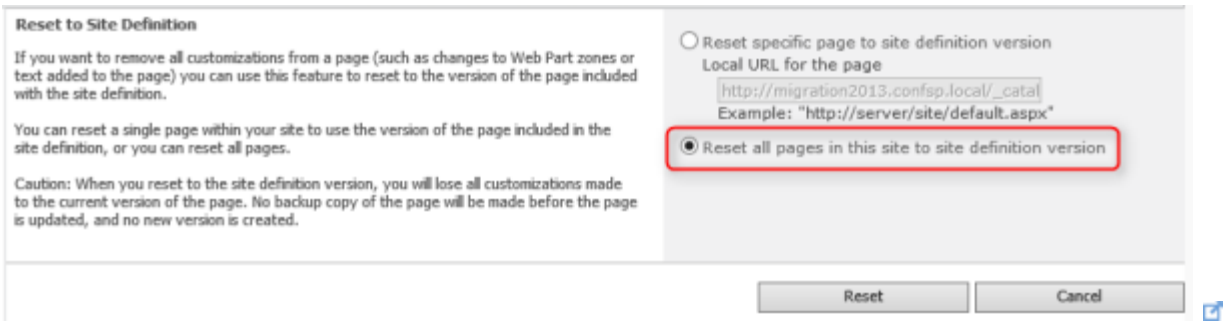


3. A summary is displayed, in our case 2 problems have been found:

- The "Video" content type is in conflict with the new one – Delete or rename it,
- A webpart definition is considered as customized file – In this case you can fix this problem by clicking on "Reset page to default".



When you click on the "Reset page to default" link, you can choose to reset a specific page or all pages in the site by using the existing site template.



** PowerShell **

- Test-SPSite : Launch the Health Checks on a site collection, with the ability to choose a specific rule to be verified.

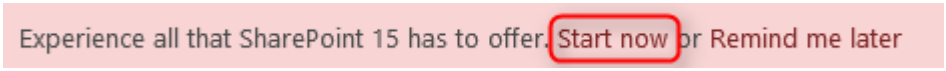
Test-SPSite -Identity <SiteURL> [-Rule <RuleID>]

- Repair-SPSite : Launch the repair, with the ability to choose a specific rule to be repaired.

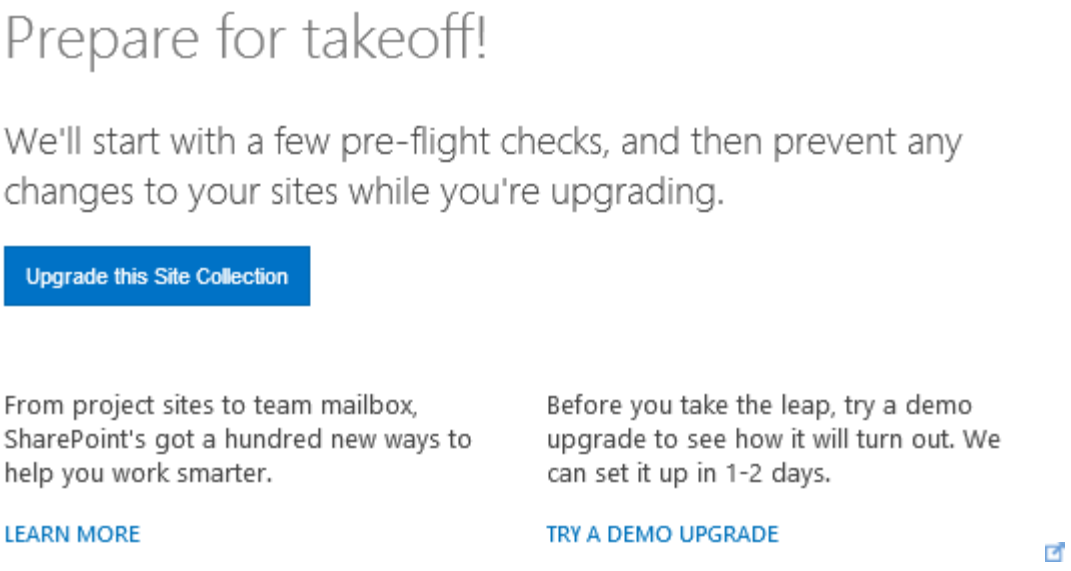
Repair-SPSite -Identity <SiteURL> [-Rule <RuleID>]

Point 4: Access the migration / Evaluation site collection request page

Click on "Start now":



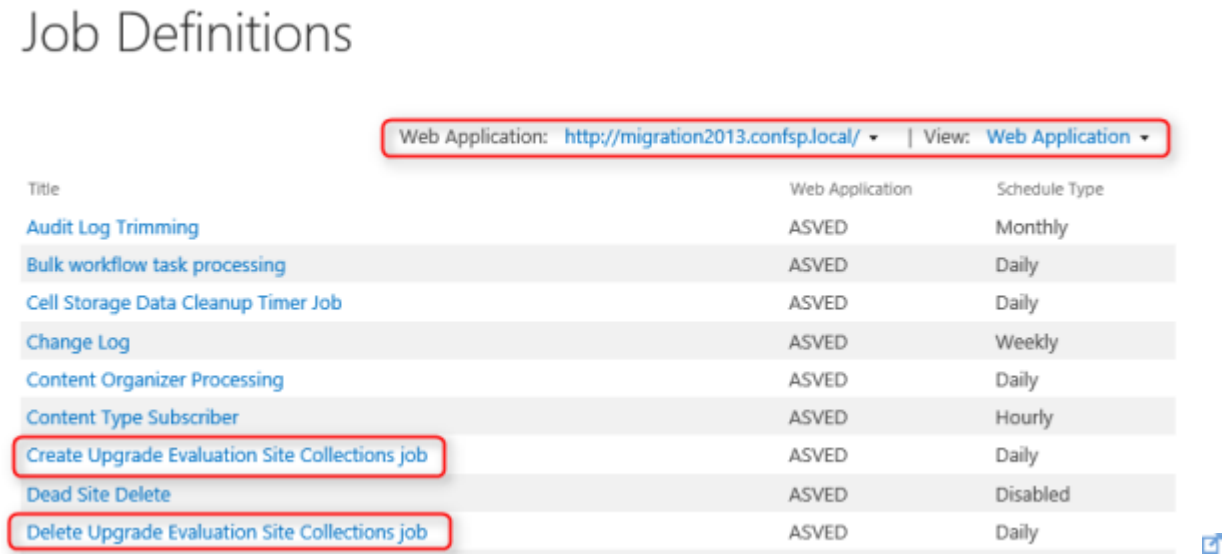
To access the migration page:



Point 5: Evaluation site request

One new capability in SharePoint 2013 is to request an upgrade evaluation site, which will be a copy of your site, in SharePoint 2013 version.

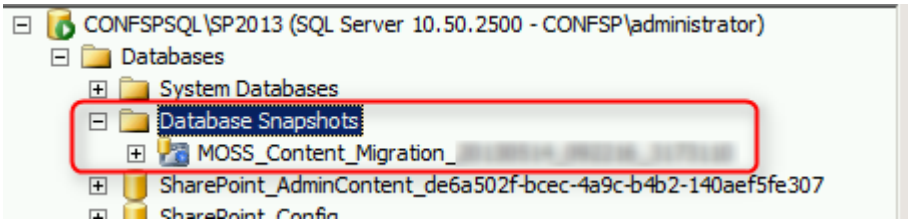
This site will be available during 30 days (by default and configurable), before being deleted by a timer job (1 for each web application).



Note : No data will be replicated from the « source » site to the copy, or the opposite; users should also be aware of the fact that this site is "short-lived".

Concerning the evaluation site creation, 2 possibilities:

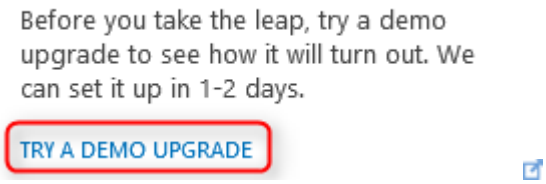
- You SQL Server version supports snapshots (Enterprise or Datacenter version) and in this case a snapshot will be used,



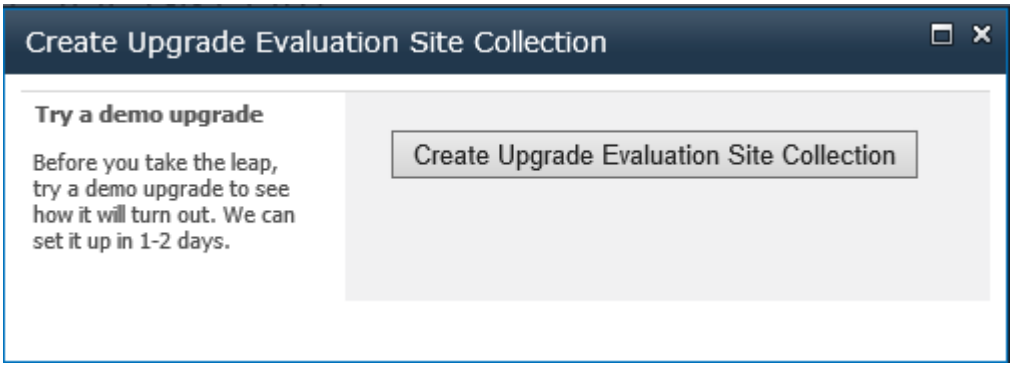
- If not, a backup-restore will be used, and your site collection will be read-only during the backup phase.

From the interface:

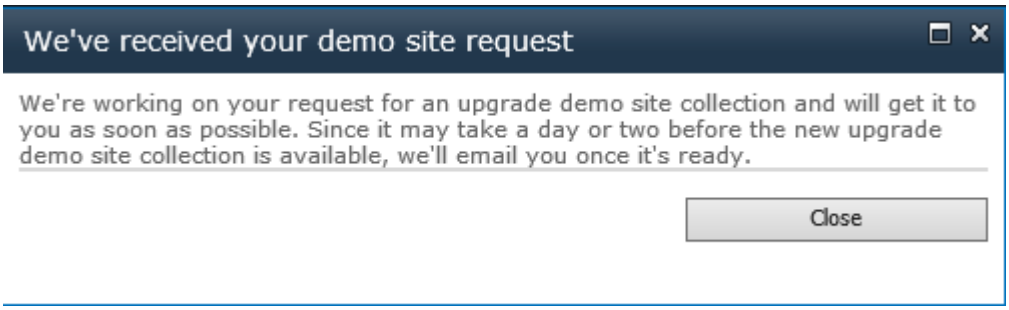
1. Click on "TRY A DEMO UPGRADE" to request a site evaluation request:



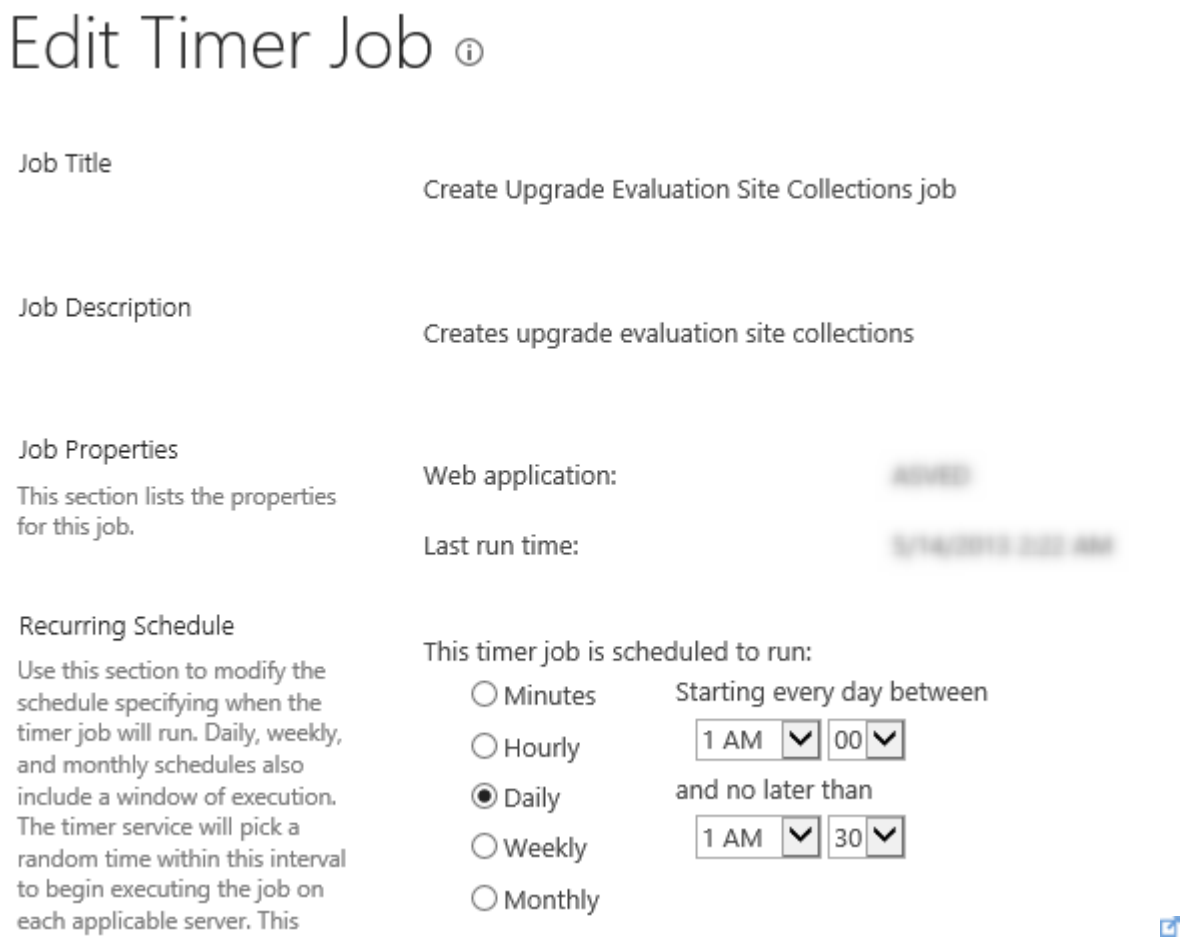
2. Click on the button :



3. Site collection administrator(s) is/are alerted by email when the upgraded copy is available.



Why a day or two ? Because the creation process are managed by timer jobs (1 for each web application) which runs once a day:



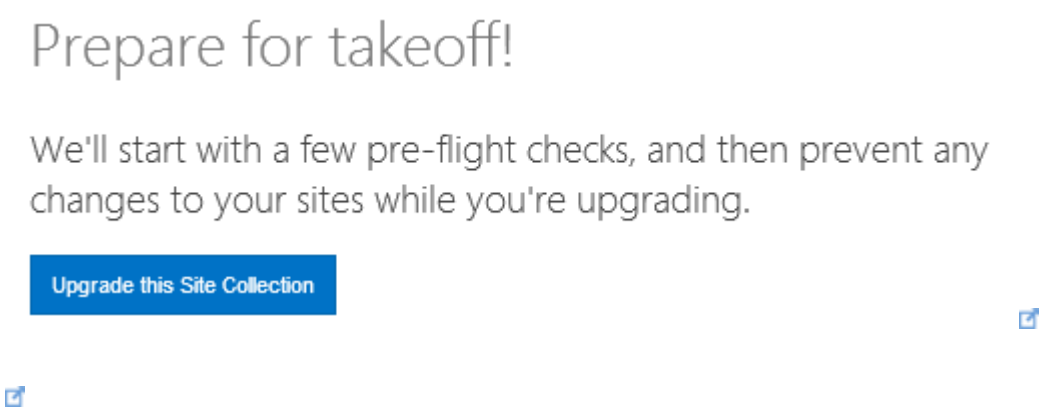
* *PowerShell* *

- Evaluation site request

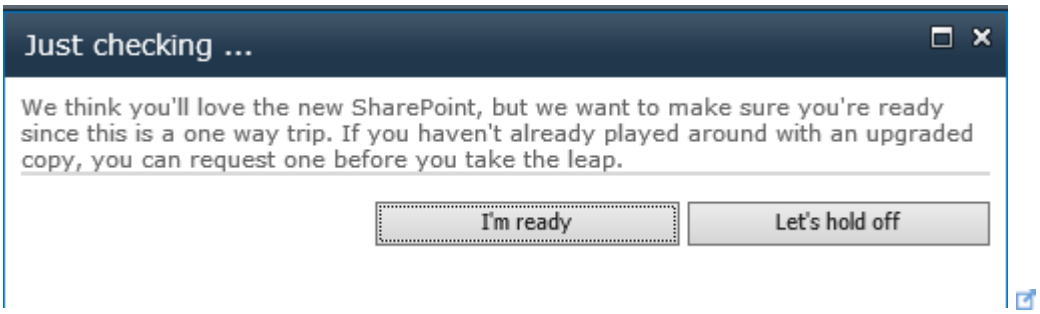
Request-SPUpgradeEvaluationSiteCollection -Identity <Site collection URL>

Point 6: Site upgrade

1. On the migration page, you can launch the upgrade by clicking on "Upgrade this Site Collection"

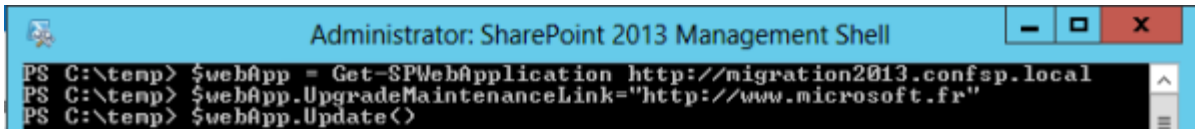


2. A message is displayed, inviting the user to request an evaluation copy, if he hasn't "played" with it before.

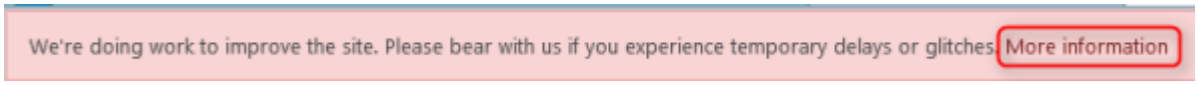


3. The migration starts, with the possibility to add a link in the status bar, with the "UpgradeMaintenanceLink" web application property.

Example :



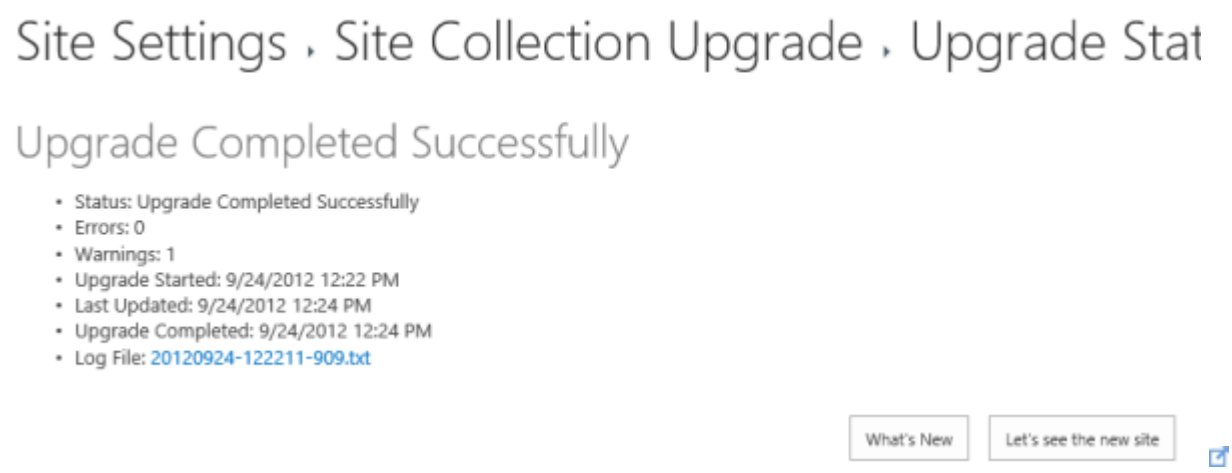
Result :



Note: The link opens in the current page.

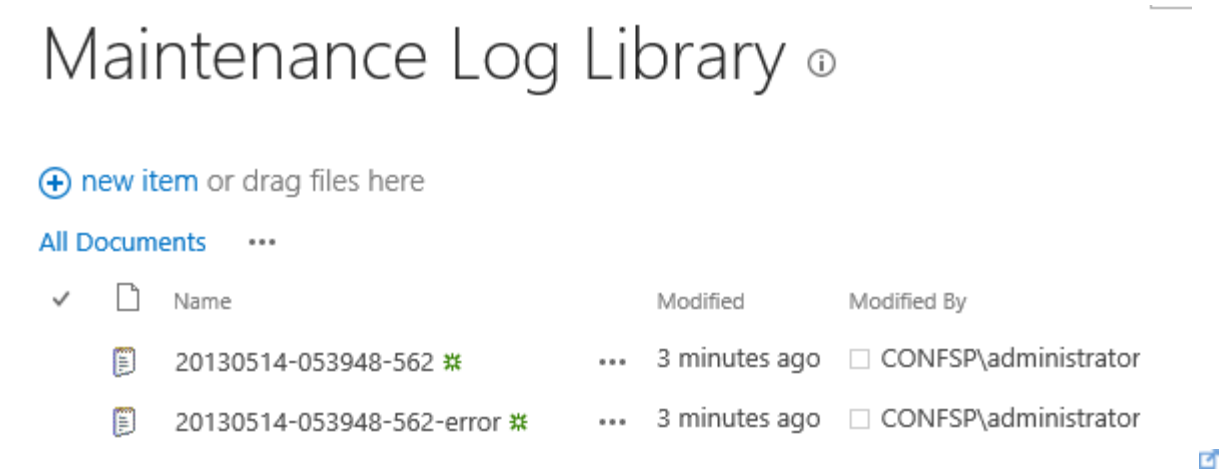
Point 7: End of the migration

1. The migration ends, a summary page is displayed:



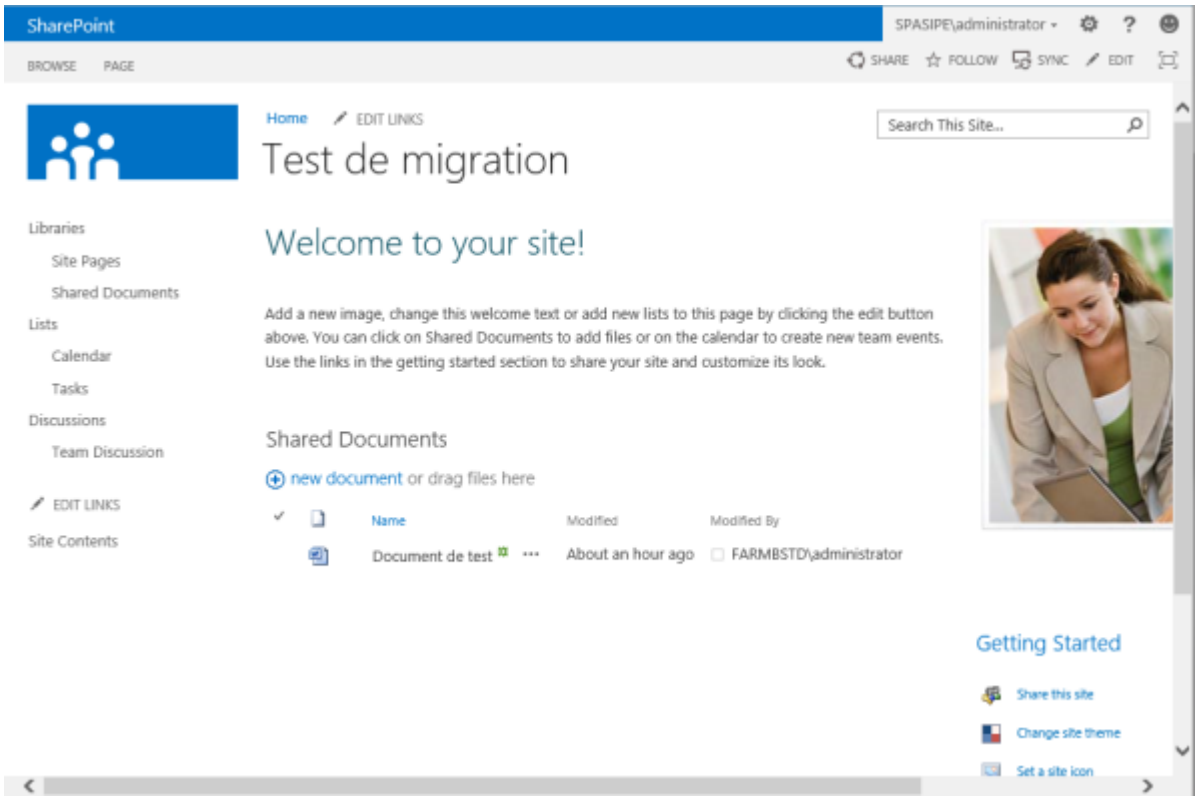
With :

- A link to log files (verbose + errors/warnings), stored in a hidden library named "Maintenance Log Library",



- A "What's New" button which opens a help page,
- A "Let's see the new site" button which opens the homepage.

2. The upgraded site is displayed correctly



* *PowerShell* *

- Launch a site collection upgrade, even if throttling limits are reached,

Upgrade-SPSite <Site collection URL> -VersionUpgrade -Unthrottled

- Launch a site collection upgrade, with queuing the upgrade

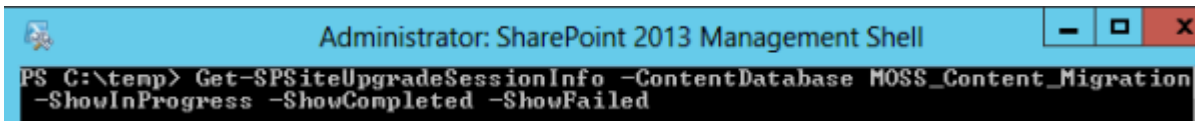
Upgrade-SPSite < Site collection URL > -VersionUpgrade **-QueueOnly**

Point 8: Upgrade queue management

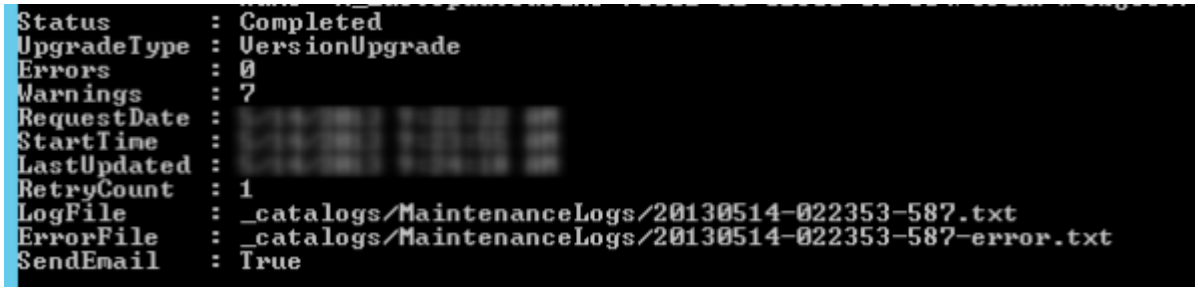
Last but not least, the SharePoint administrator can manage the upgrade queue, only with PowerShell !

- Site collections in the queue (Note the ShowInProgress / ShowCompleted / ShowFailed parameters)

Get-SPSiteUpgradeSessionInfo -ContentDatabase <Database name> -ShowInProgress -ShowCompleted -ShowFailed



Several informations are displayed (the image is just an extract):



- Site collection upgrade status

Get-SPSiteUpgradeSessionInfo -Site < Site collection URL>

For the informations displayed, see the screenshot above.

- Add a site collection in the queue

Upgrade-SPSite < Site collection URL> -VersionUpgrade **-QueueOnly**

- Remove a site collection from the queue (if it's not being upgraded)

Remove-SPSiteUpgradeSessionInfo -Identity < Site collection URL>

Point 9: Throttling

- Number of simultaneous upgrades allowed per application pool

$wa = Get-SPWebApplication -URL < Web Application URL >

$wa.SiteUpgradeThrottleSettings.AppPoolConcurrentUpgradeSessionLimit=<Value>

- Number of simultaneous upgrades allowed per database

$db = Get-SPContentDatabase <Database name>
$db.ConcurrentSiteUpgradeSessionLimit

- Maximum size to allow self-service upgrade

$wa.SiteUpgradeThrottleSettings.UsageStorageLimit=<Value>

- Maximum number of sub-sites to allow self-service upgrade

$wa.SiteUpgradeThrottleSettings.SubwebCountLimit=<Value>

References

- Use Windows PowerShell cmdlets to upgrade to SharePoint 2013 and to migrate data

Other articles

This article is part of a series of articles dedicated to the Migration to SharePoint 2013.

They are actually written in French and hosted on my blog, but I will translate and transfer them on the TechNet Wiki.

- How upgrade to SharePoint 2013 ?
- SharePoint 2013 : Migrate a simple SharePoint 2010 site

- SharePoint 2013 : Migrate a simple SharePoint 2007 site
- Migrer vers SharePoint 2013 : Upgrade des collections de sites (UI et PowerShell) (This article)
- SharePoint 2013 : Prepare the migration of specific developments

SharePoint 2013: Going Up in the Navigation

Going up can be a little bit difficult in SharePoint 2013, especially for newcomers. To address the issue, here's a list of things you can do:

- **Create** global navigation by inheriting navigation from parent sites
- Manually, **add** navigation links to the top link bar
- **Turn on** the tree view
- **Work** with hyperlinks and favorites
- **Add** the site map control using SharePoint Designer:http://www.manageprojectsonsharepoint.com/blog/2012/12/19/restore-navigate-up-on-sharepoint-2013/ /
- **Use** Managed Navigation: http:// technet.microsoft.com/en-us/library/dn194311.aspx
- **Use** browser back and forward buttons, in some scenarios this works fine.

See Also

Visit the SharePoint 2013 Best Practices page athttp://social.technet.microsoft.com/wiki/contents/articles/12438.sharepoint-2013-best-practices.aspx

Other Languages

This article is also available in the following languages:

- SharePoint 2013: Remonter Dans la Navigation (fr-FR)