|  |
| --- |
|  |

17/3/2015

|  |
| --- |
|  |
| https://uknowit.uwgb.edu/images/group91/41011/Office365.jpg |
|  |



|  |  |
| --- | --- |
|  | Curso de Office 365 - Modelo de Objetos en Cliente para SPO |

**Indice**

[1 Introducción 2](#_Toc414021903)

[2 Modelo de Objetos en Cliente y API REST para SPO 2](#_Toc414021904)

[2.1 Pre-requisitos para el Laboratorio 2](#_Toc414021905)

[2.1.1 Creación de una Colección de Sitios para el laboratorio 2](#_Toc414021906)

[2.1.2 Configurar el catálogo de Aplicaciones de SPO 4](#_Toc414021907)

[2.2 Uso de la API REST de SPO 6](#_Toc414021908)

[2.2.1 Preparación del entorno del escenario de uso de la API REST 6](#_Toc414021909)

[2.2.2 Uso de la API REST desde el navegador 8](#_Toc414021910)

[2.2.3 Instalación de NAPA en el sitio de desarrollador 10](#_Toc414021911)

[2.2.4 Uso de la API REST desde una Aplicación 12](#_Toc414021912)

[2.3 Uso del CSOM para SPO 17](#_Toc414021913)

[2.3.1 Consultar los elementos de una lista con CSOM especificando las credenciales de acceso 17](#_Toc414021914)

[2.3.2 Operaciones CRUD en Listas con CSOM especificando las credenciales de acceso 20](#_Toc414021915)

[2.3.3 Simulación de un Timer Job con el CSOM sin especificar credenciales de acceso 21](#_Toc414021916)

[2.3.4 Referencias 28](#_Toc414021917)

[3 Uso de las APIs de SPO en PowerShell 28](#_Toc414021918)

[3.1 Entorno de PowerShell para SharePoint Online 28](#_Toc414021919)

[3.1.1 SharePoint Online Management Shell 29](#_Toc414021920)

[3.1.2 ISE 30](#_Toc414021921)

[3.2 Uso del Modelo de Objetos en Cliente de SharePoint desde PowerShell 32](#_Toc414021922)

[3.3 Como utilizar el CSOM de SPO en PowerShell 33](#_Toc414021923)

[3.4 Como obtener todas las Colecciones de Sitios del tenant mediante CSOM 33](#_Toc414021924)

[3.5 Como obtener todos los sitios de una Colección de Sitios mediante CSOM 35](#_Toc414021925)

[3.6 Uso de la API REST de SPO desde PowerShell 36](#_Toc414021926)

[3.7 Como depurar Scripts PowerShell con el ISE 37](#_Toc414021927)

[4 Encuesta de satisfaction 38](#_Toc414021928)

# Introducción

El presente laboratorio recoge una serie de ejercicios y laboratorios en los que se trata de poner en práctica las posibilidades que las APIs de Cliente de SharePoint Online (SPO) proporcionan a los desarrolladores para poder extender SPO desde dos perspectivas:

* Programas y Aplicaciones creados desde Visual Studio / Otras herramientas de desarrollo.
* Scripts PowerShell en los que se haga uso de las APIs de Cliente para interactuar con SPO.

# Modelo de Objetos en Cliente y API REST para SPO

En esta sección se incluyen una serie de prácticas relativas al uso del Modelo de Objetos en Cliente (CSOM) y API REST para SPO.

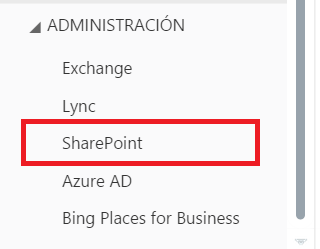
## Pre-requisitos para el Laboratorio

Para este laboratorio se requiere que tenga una cuanta activa de Office 365, si no la tiene deberá crearse una de prueba por un mes la siguiente URL: <http://products.office.com/es-es/business/office-365-enterprise-e3-business-software>

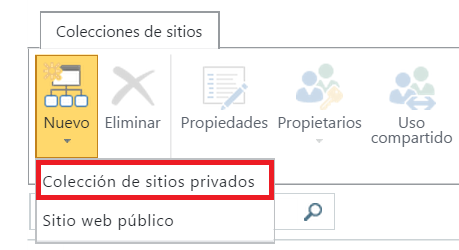
### Creación de una Colección de Sitios para el laboratorio

Una vez aprovisionada la cuenta de Office 365, debe crear una colección de sitios basada en la plantilla “Sitio de desarrollador”, los pasos que deberá ejecutar son los siguientes:

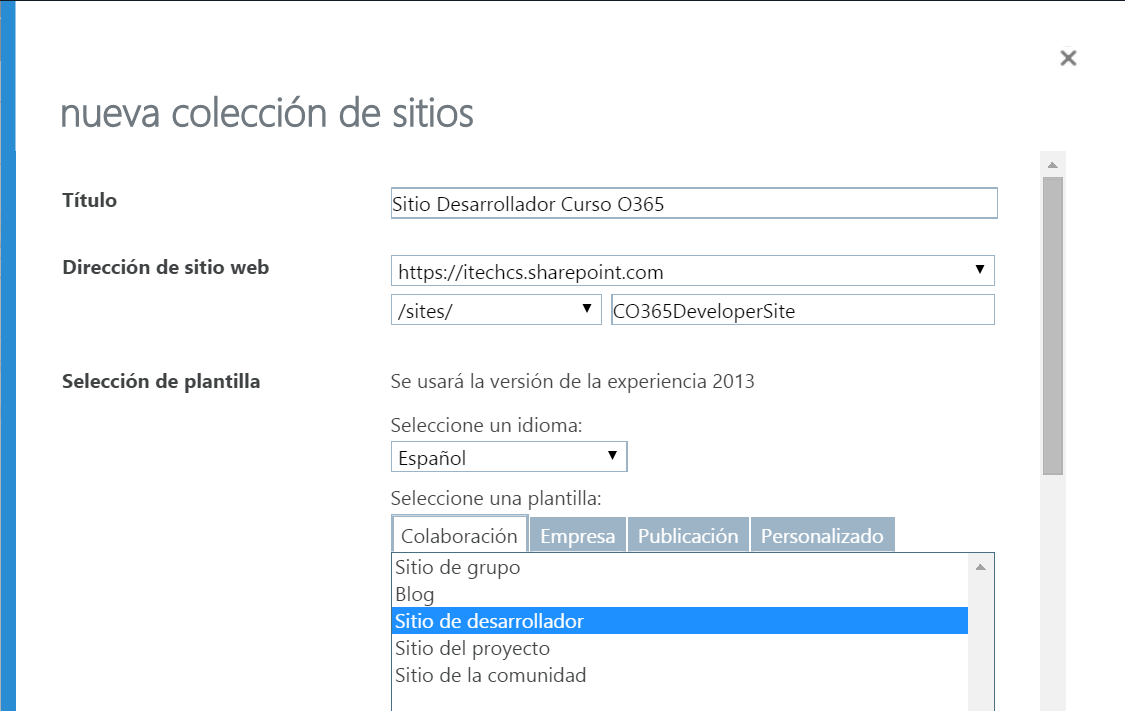
* Acceda a su cuenta de Office 365.
* Acceda a la administración de SharePoint utilizando el menú ADMIN -> SharePoint disponible en el Centro de Administración de Office 365:



* En la sección colección de sitios seleccione “Nuevo” en la barra de tareas y luego “Colección de Sitios”.



* Ingrese los datos de la nueva colección de sitios, seleccione la plantilla mencionada y presione “Aceptar”.



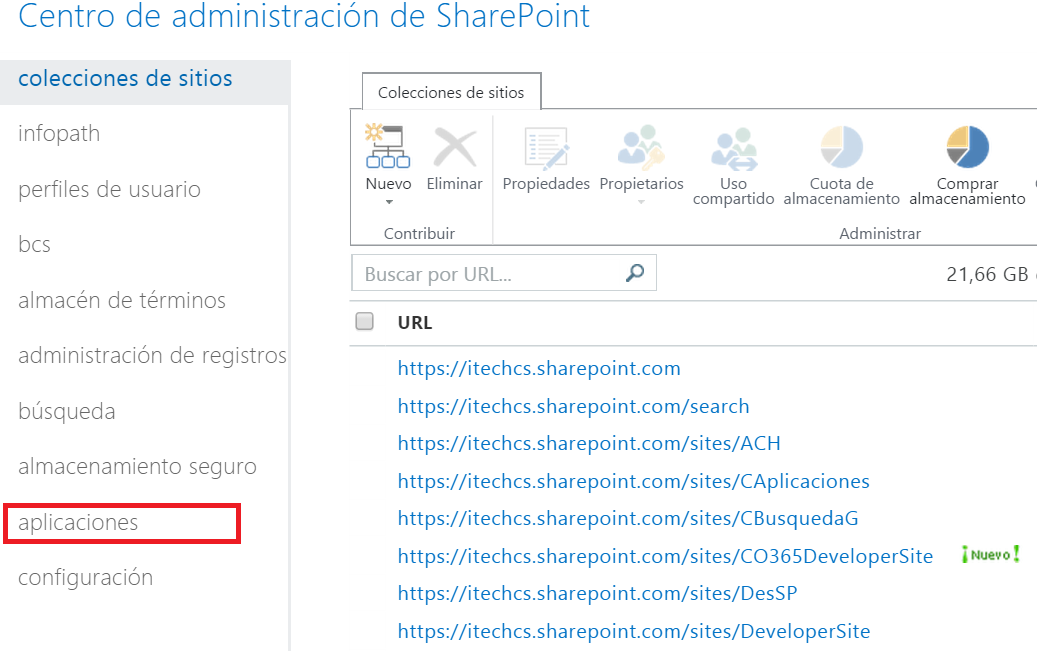
* Una vez la Colección de Sitios está creada, verifique que puede acceder a la misma sin problemas desde la Administración de SharePoint Online.



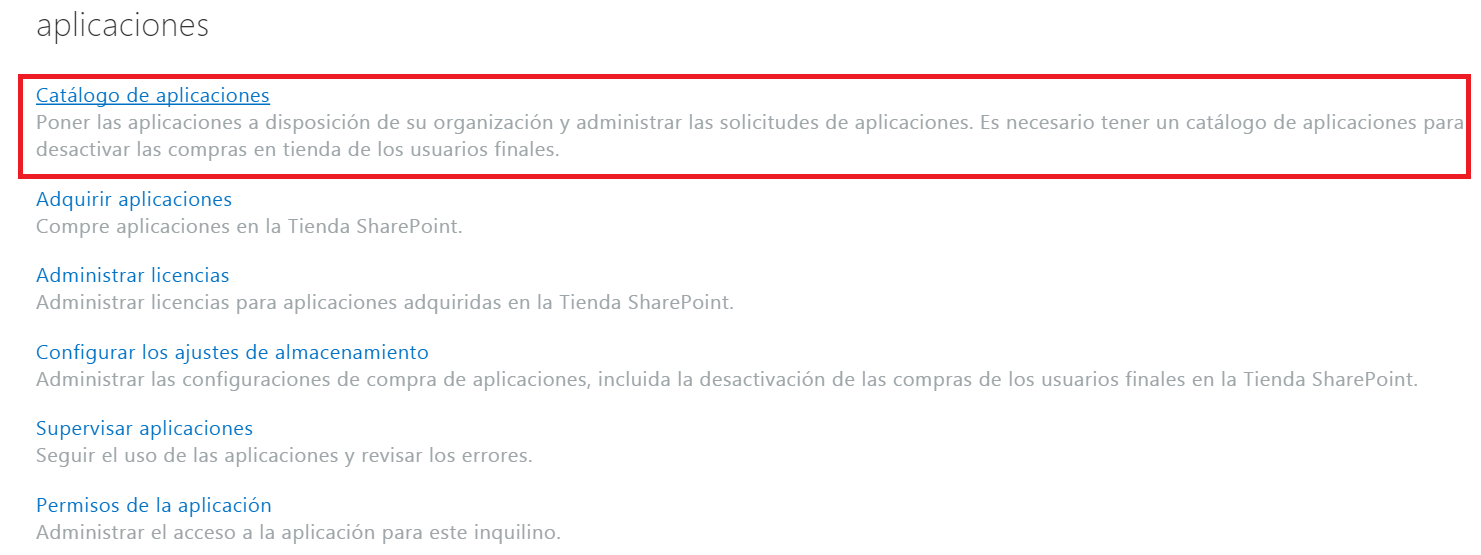
### Configurar el catálogo de Aplicaciones de SPO

Si ya tiene una colección de sitios basada en la plantilla “Sitio de desarrollador” proceda a configurar el catálogo de Aplicaciones, si no ejecute el paso anterior. Para configurar el catálogo de Aplicaciones ejecute los siguientes pasos en su subscripción Office 365.

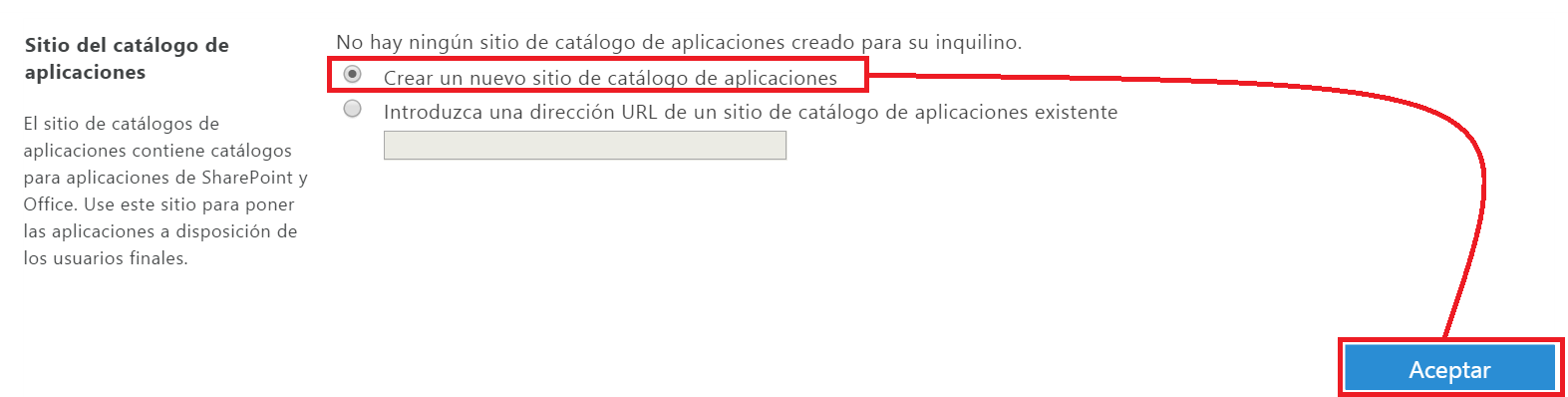
* Acceda a su cuenta de Office 365.
* Acceda a la administración de SharePoint utilizando el menú ADMIN -> SharePoint disponible en el Centro de Administración de Office 365.
* Acceda a la sección “Aplicaciones” utilizando la opción de menú que se encuentra en el menú vertical de la izquierda.



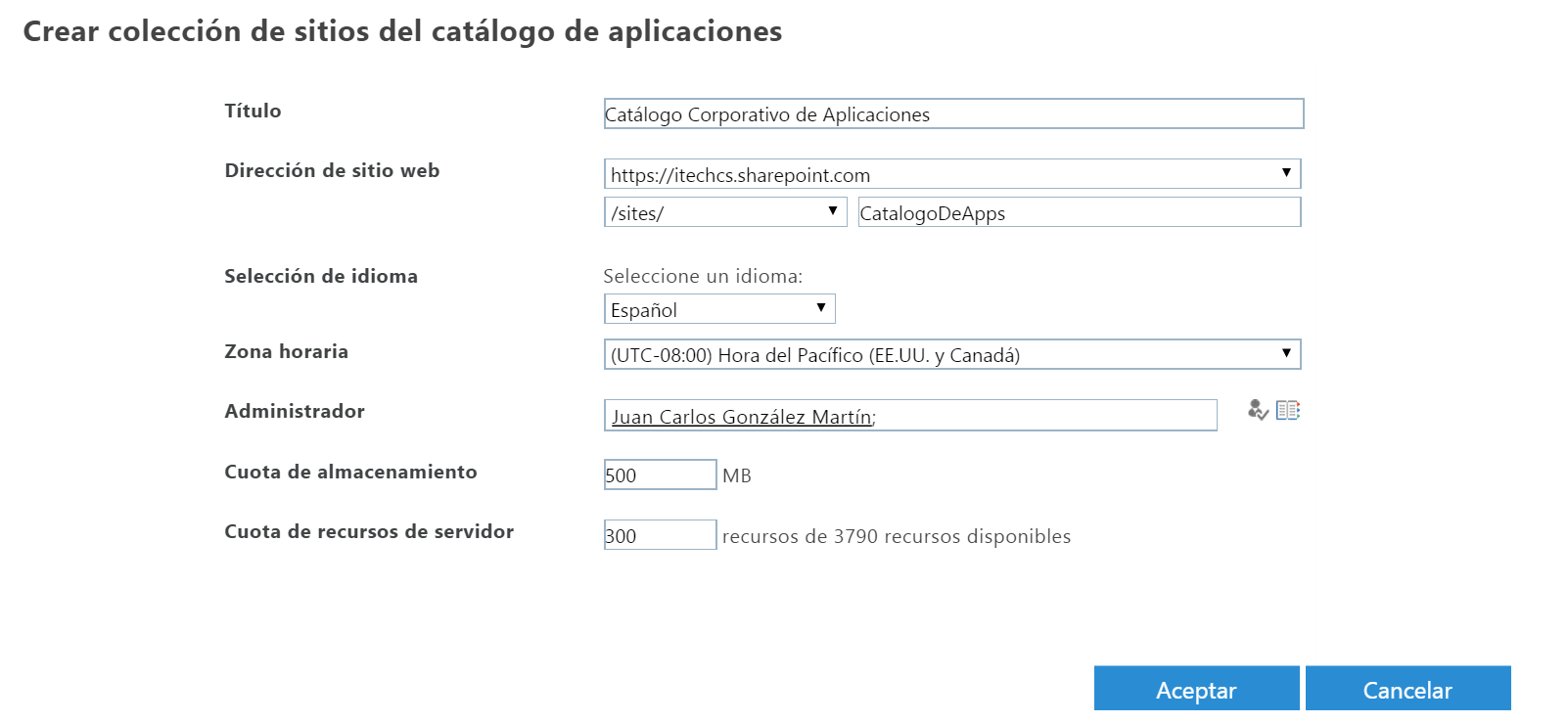
* Acceda a la sección “Catálogo de aplicaciones”. Si ya tiene un catálogo configurado está opción navegará hasta el sitio creado por SharePoint, si no tendrá la opción para configurar uno.



* Selecciones la opción “Crear un nuevo sitio de catálogo de aplicaciones” y después seleccione “Aceptar”.



* Añada los datos del nuevo Sitio que utilizara para el catálogo de aplicaciones y presione “Aceptar”.



* El nuevo sitio será aprovisionado en unos minutos y el catálogo de aplicaciones estará disponible para su uso.

## Uso de la API REST de SPO

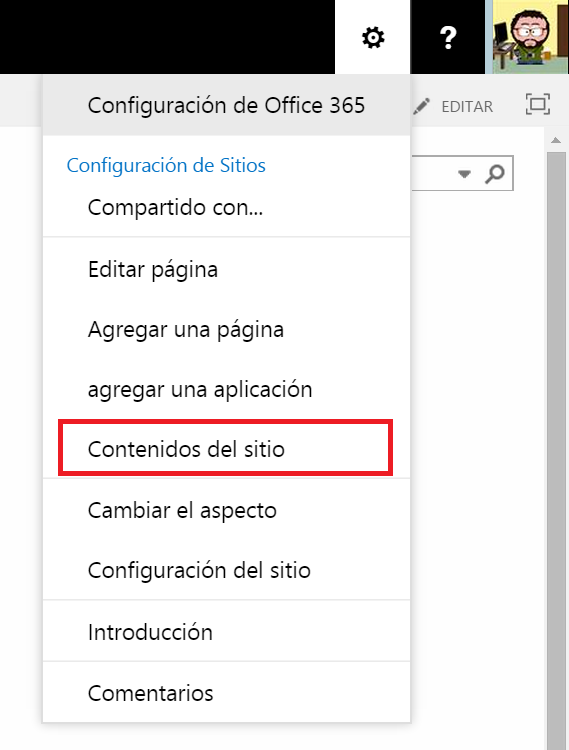
En esta sección vamos a ver ejemplos de cómo hacer uso de la API REST de SharePoint Online de acuerdo a dos posibilidades:

* A través del uso del navegador como cliente de consulta de información de SPO mediante REST.
* A través de una sencilla aplicación que permita consultar información de una lista del sitio de SharePoint Online.

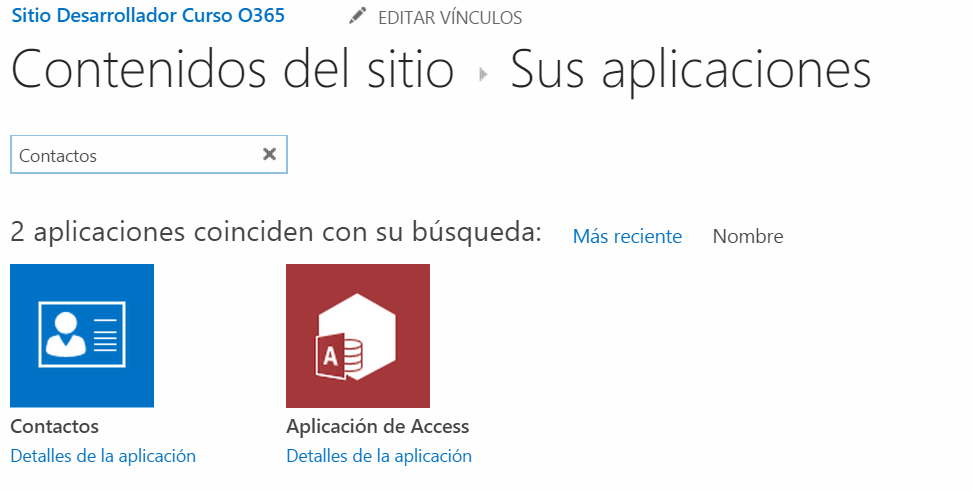
### Preparación del entorno del escenario de uso de la API REST

En primer lugar, vamos a preparar el escenario de uso de la API REST añadiendo al sitio de desarrollador los elementos necesarios:

* Navegue al sitio del desarrollador al sitio de desarrollador creado anteriormente para agregar una Aplicación de tipo “Contactos” al mismo. Para añadir una Aplicación acceda a la sección “Contenido del sitio” que se encuentra en el menú vertical izquierdo como se puede ver en la siguiente imagen.



* Seleccione “Agregar Aplicación” en la parte superior de la pantalla. En la sección “Aplicaciones que usted puede agregar” seleccione “Contactos”.



* En la ventana modal que se abre, en el campo “Seleccione un Nombre” ingrese “Contactos Curso Office 365” y presione crear.
* En la seccione “Contenido del sitio” seleccione la nueva aplicación creada que se llama “Contactos Curso Office 365” y está marcada con un icono de color verde “Nuevo”. Añada varios contactos en la aplicación como se puede ver en la siguiente imagen.



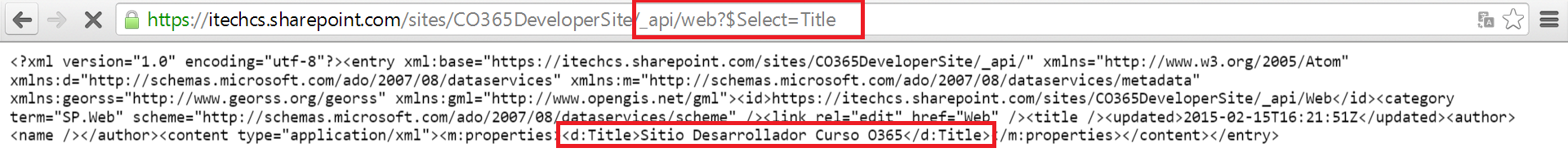
### Uso de la API REST desde el navegador

Una forma rápida de hacer uso de la API REST es a través de lanzar consultas en el navegador:

* Para acceder a la propiedad “Title” del sitio de desarrollador, simplemente tendremos que especificar la siguiente consulta REST que devuelve el título del sitio haciendo uso de OData:

|  |
| --- |
| https://itechcs.sharepoint.com/sites/CO365DeveloperSite/\_api/web?$Select=Title |

* La salida por pantalla que se obtiene es la siguiente:



* Para obtener un listado de todas las listas del Sitio, simplemente especificamos la siguiente Url en el navegador:

|  |
| --- |
| https://itechcs.sharepoint.com/sites/CO365DeveloperSite/\_api/web/lists?$Select=Title |

* En este caso, la salida por pantalla que se obtiene es la siguiente:



* Para mostrar el conjunto de elementos de una lista mediante REST y OData, la URL a especificar en el navegador es la siguiente:

|  |
| --- |
| https://itechcs.sharepoint.com/sites/CO365DeveloperSite/\_api/web/lists/getbytitle('Contactos Curso Office 365')/items/?$select=Created,Title&Orderby=Created |

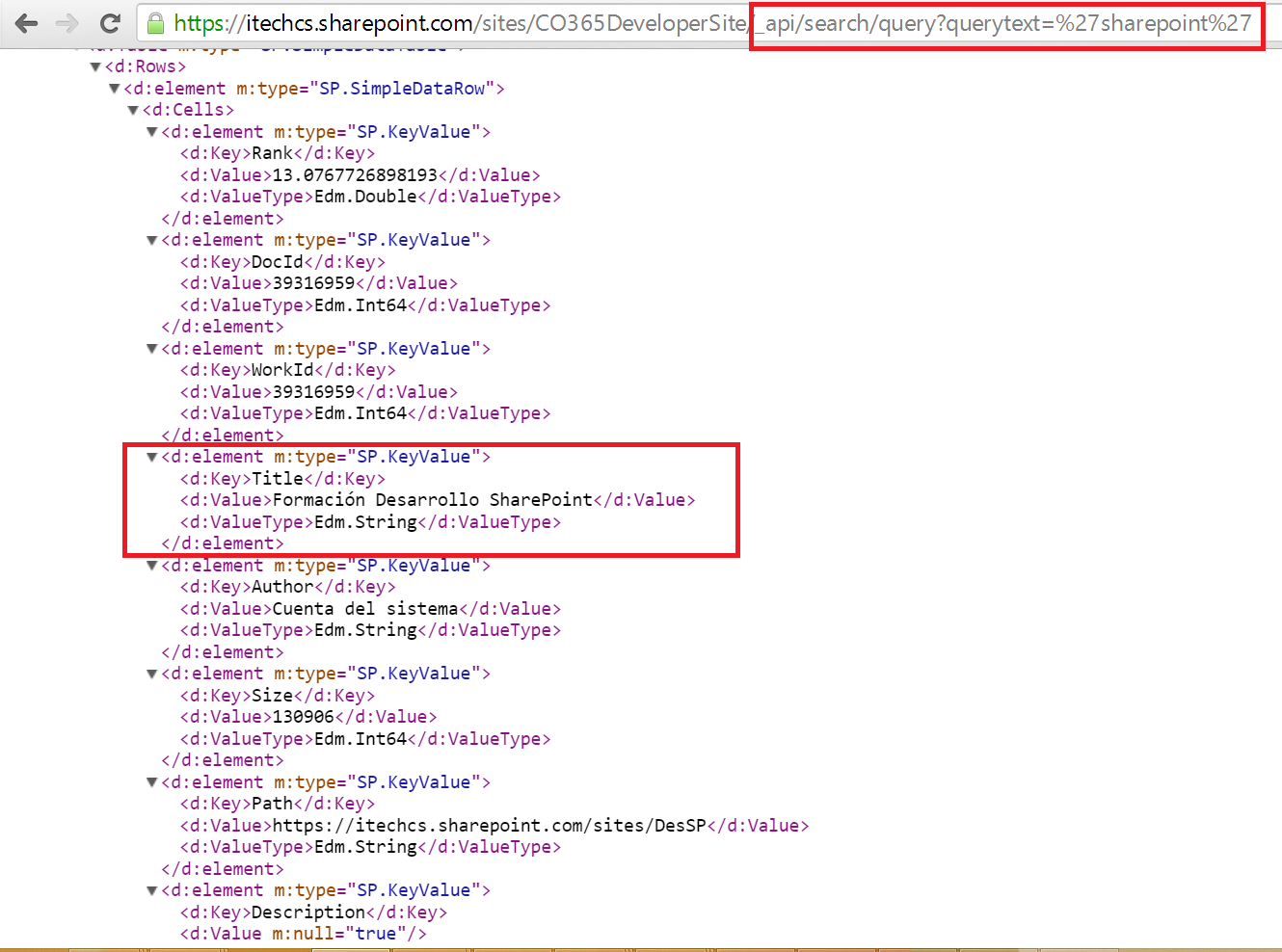
* Y la salida que se obtiene es la siguiente:



* De la misma forma que podemos trabajar con contenedores de SharePoint como Sitios, Listas o elementos de lista, podemos hacer uso de los servicios de la plataforma. Por ejemplo, para hacer uso mediante REST del servicio de búsqueda podemos especificar una consulta REST como la siguiente:

|  |
| --- |
| https://itechcs.sharepoint.com/sites/CO365DeveloperSite/\_api/search/query?querytext='sharepoint' |

* La salida por pantalla que se obtiene es la siguiente:



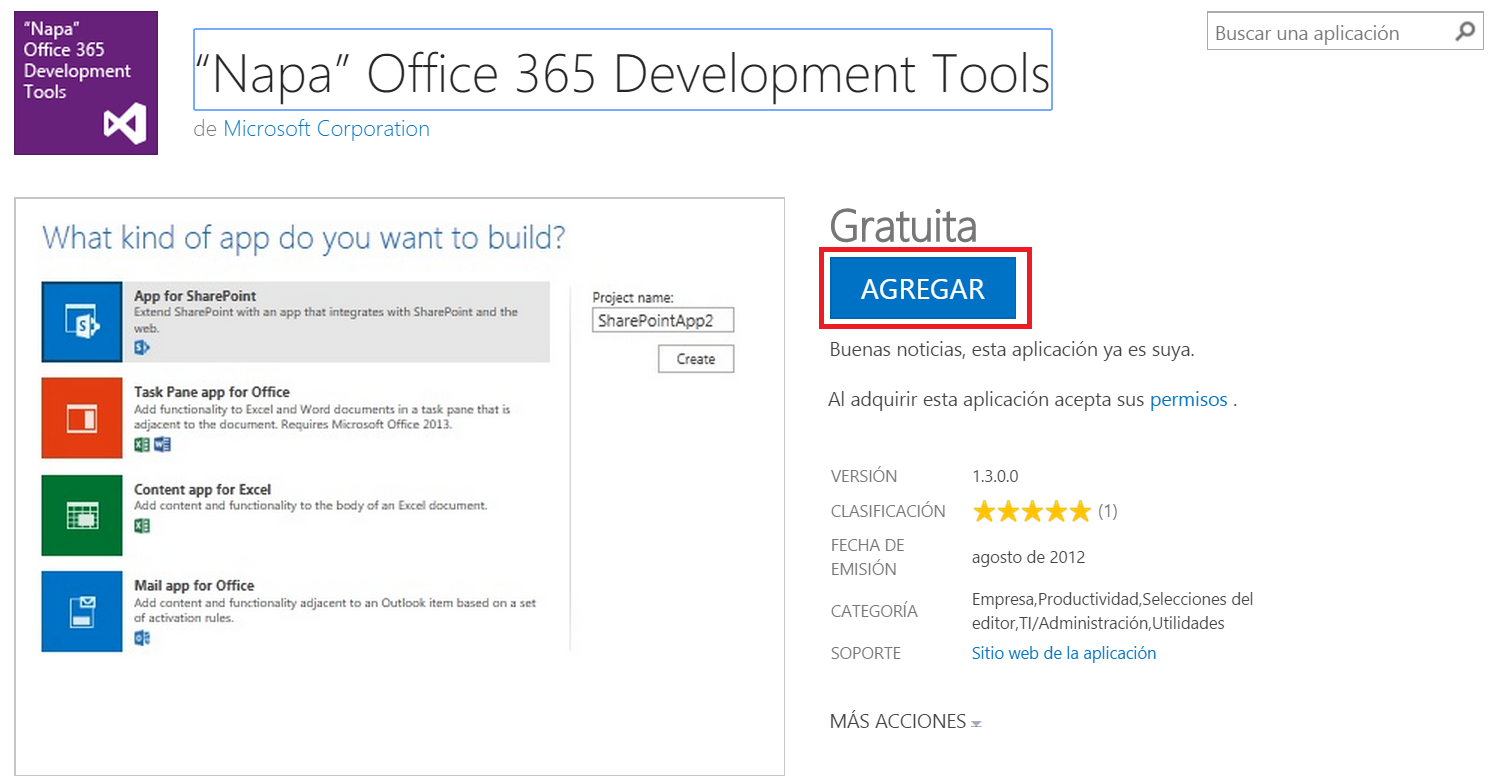
### Instalación de NAPA en el sitio de desarrollador

En primer lugar, vamos a preparar el entorno de desarrollo en el navegador añadiendo la aplicación NAPA al sitio de desarrollador creado en la sección 2.1.1:

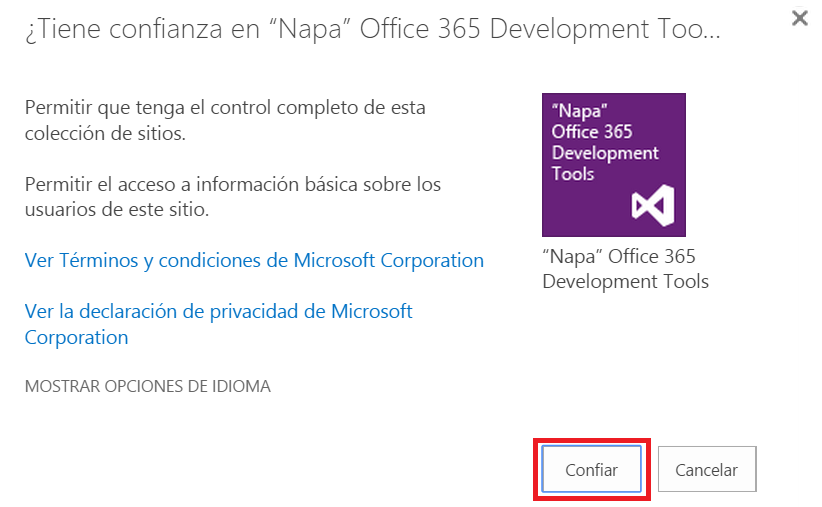
* Navegamos al sitio del desarrollador y en la página principal hacemos clic el Tile “Crear una aplicación”.



* A continuación seremos redirigidos a la página de detalle de NAPA para proceder a su instalación. Hacemos clic en el botón “AGREGAR”.



* En el diálogo modal que se abre, simplemente hacemos clic en “Confíar”.



* Una vez NAPA se instala, tendremos disponible el correspondiente acceso a la herramienta en la página de “Contenidos del sitio”.

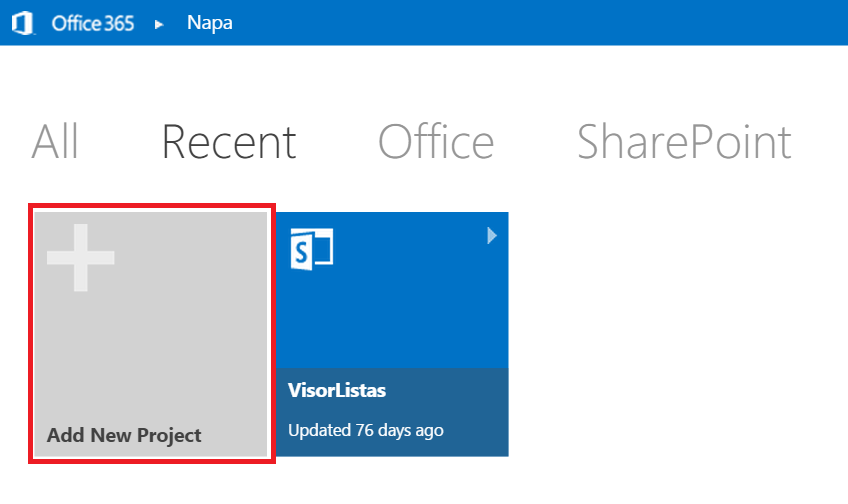


### Uso de la API REST desde una Aplicación

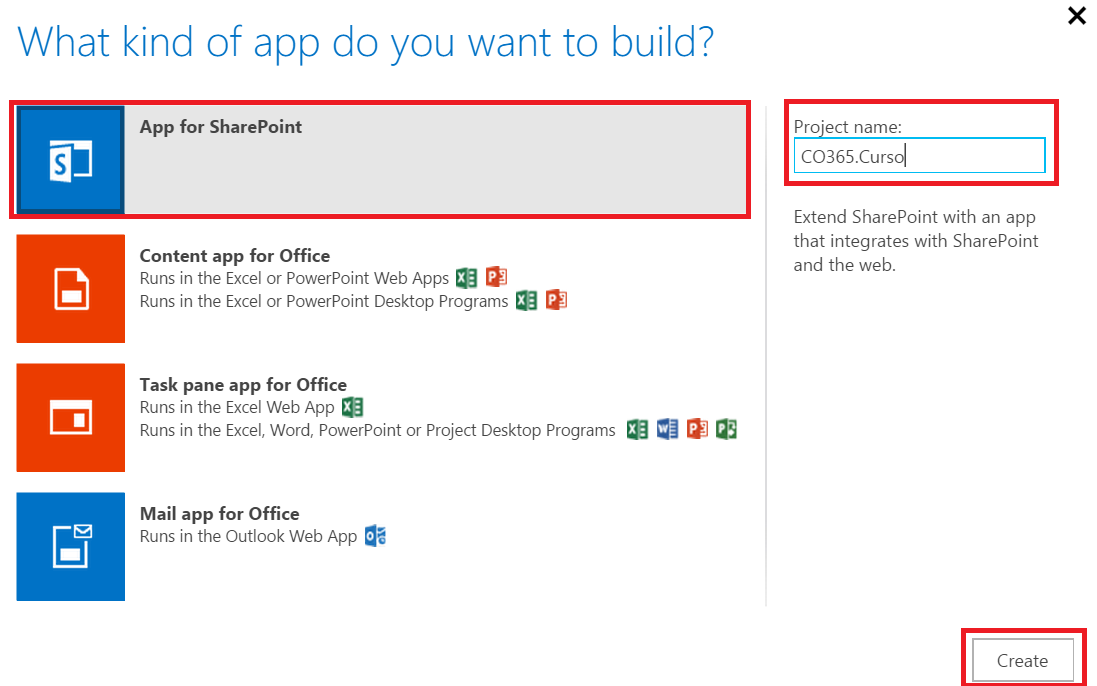
En esta sección vamos a ver como hacer uso de la API REST de SPO desde código JavaScript en una Aplicación creada en NAPA:

En esta tarea creará una aplicación que mediante JavaScript, jQuery y OData cargue él nombre de un contacto en la página.

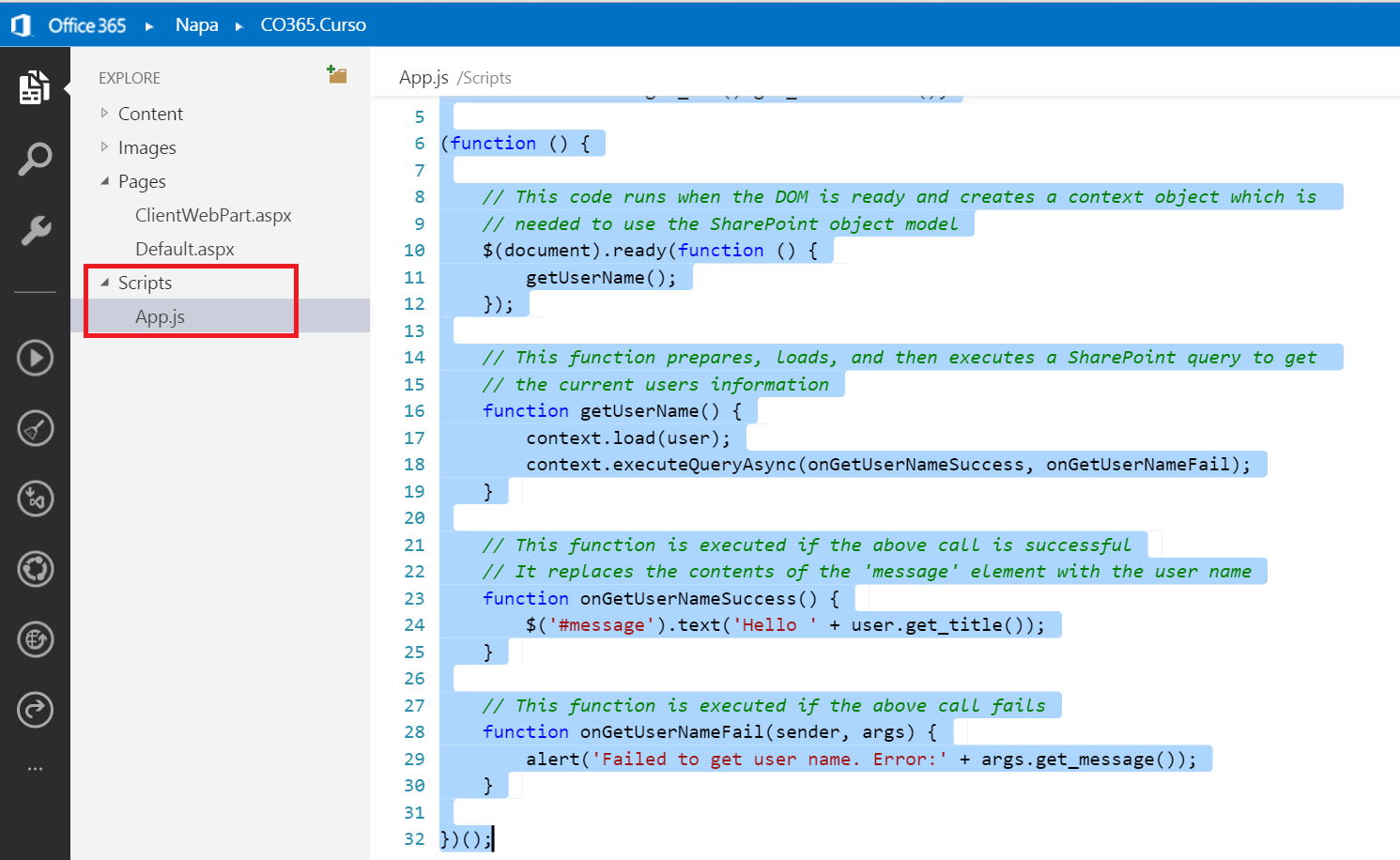
* En el sitio de desarrollador, hacemos clic en el “Tile” “Crear una Aplicación” de forma que se muestra el entorno de creación de Aplicaciones de NAPA. En la página principal de la aplicación seleccione “Add New Project”.



* En la ventana modal que se abre seleccione “App for SharePoint”, en la campo “Project name” especifique “CO365.Contactos” y presione el botón “Create”.



* Editamos el archivo App.js ubicado en la carpeta “Scripts” del entorno de NAPA y borramos su contenido:



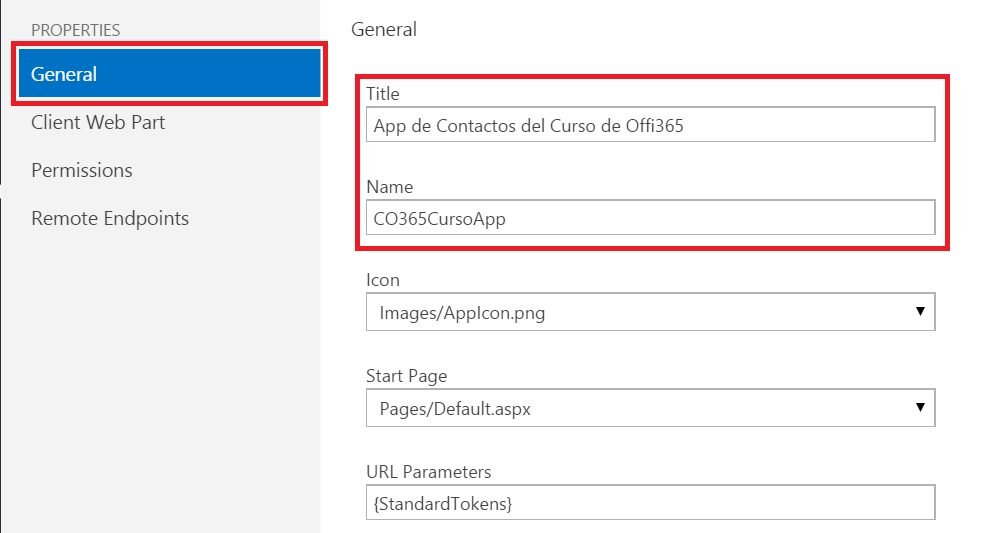
* Añadimos el siguiente código JavaScript en App.js:

|  |
| --- |
| 'use strict';  var mSPAppWebUrl = decodeURIComponent(getUrlParametros()["SPAppWebUrl"]);  alert(mSPAppWebUrl);  var mSPHostUrl = decodeURIComponent(getUrlParametros()["SPHostUrl"]);  alert(mSPHostUrl);  var scriptbase = mSPHostUrl + "/\_layouts/15/";  $(document).ready(function () {  $.getScript(scriptbase + "SP.RequestExecutor.js", ObtenerDatosListaContactosCrossDomain);  });  function ObtenerDatosListaContactosCrossDomain() {  var executor = new SP.RequestExecutor(mSPAppWebUrl);  executor.executeAsync(  {  url:  mSPAppWebUrl +  "/\_api/SP.AppContextSite(@target)/web/lists/getbytitle('Contactos Curso Office 365')/items?$select=Title,FirstName&$filter=Title eq 'Imaz'&@target='" +  mSPHostUrl + "'",  method: "GET",  headers: { "Accept": "application/json; odata=verbose" },  success: ObtenerDatosExitoso,  error: ObtenerDatosError  }  );  }  function ObtenerDatosExitoso(pElementos) {  var jsonObject = JSON.parse(pElementos.body);  $.each(jsonObject.d.results, function (value, key) {  $('#message').text('Apellido: ' + key.Title + ' Nombre: ' + key.FirstName);  });  }  function ObtenerDatosError(data, errorCode, errorMessage) {  alert('Se produjo un error al cargar los datos: ' + errorMessage);  }  function getUrlParametros() {  var lParametros = [], hash;  var hashes = window.location.href.slice(window.location.href.indexOf('?') + 1).split('&');  for (var i = 0; i < hashes.length; i++) {  hash = hashes[i].split('=');  lParametros.push(hash[0]);  lParametros[hash[0]] = hash[1];  }  return lParametros;  } |

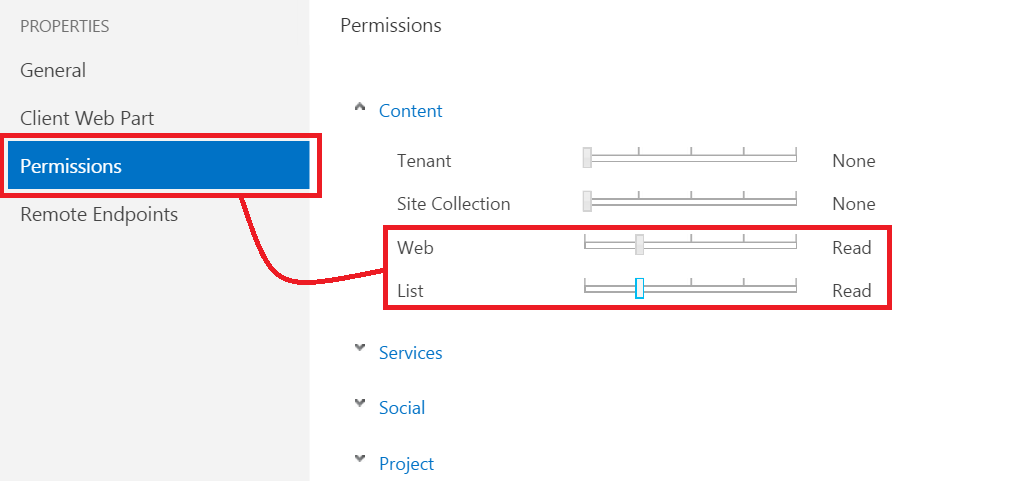
* En el código cambie la consulta OData para establecer el apellido de la persona que desea cargar y mostrar, ya que la consulta tiene un filtro como se puede apreciar a continuación.

|  |
| --- |
| /\_api/SP.AppContextSite(@target)/web/lists/getbytitle('MisContactos')/items?$select=Title,FirstName&$filter=Title eq 'Imaz'&@target='" |

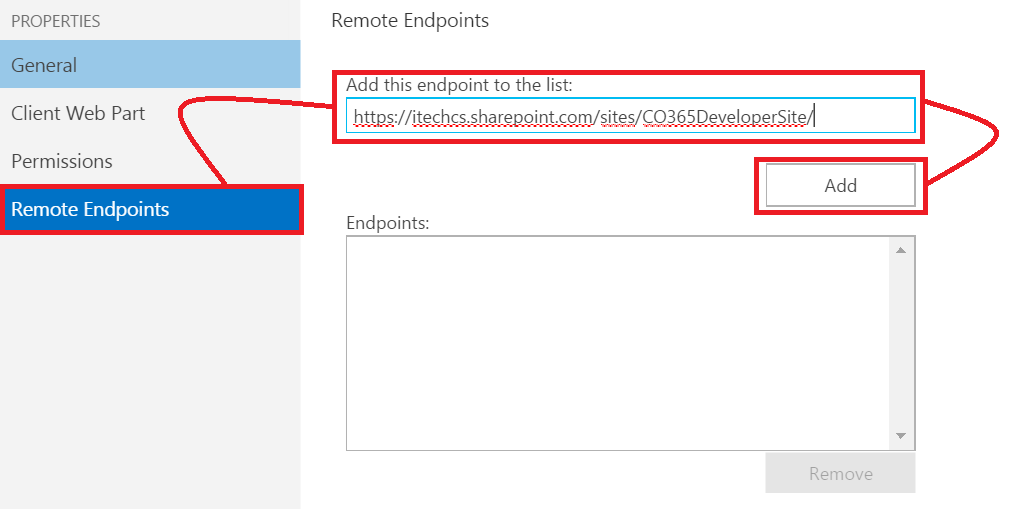
* Antes de ejecutar la aplicación se debe configurar los permisos. Acceda a pantalla de configuración haciendo click en el icono “” que permite acceder a las propiedades de la aplicación.
* En la página de configuración realice los siguientes cambios:
  + En la sección “General”, configure los parámetros “Title” y “Name” con valores adecuados.



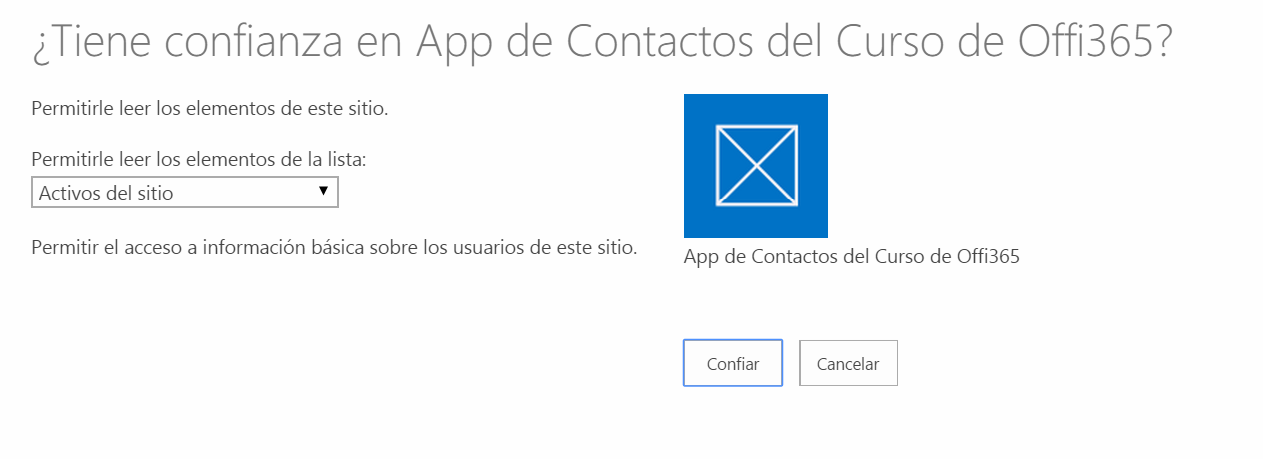
* + En la sección “Permissions” establezca los permisos “Readd” para los ámbitos “Web” y “List”.



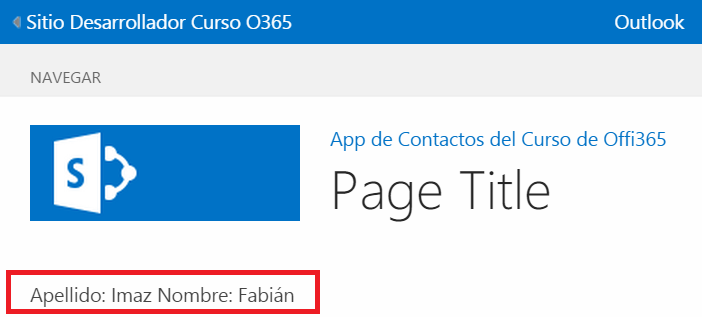
* + En la sección “Remote Endpoints” añada la Url del sitio dónde se ubica la lista “Contactos Curso Office 365” y pulse “Add”.



* Cierre la ventana de configuración. La aplicación al estar ejecutándose en un dominio diferente al dominio donde se creó la lista realizara una llamada de dominio cruzado y para que la misma se ejecute correctamente debemos estipular cual es dicho domino.
* Ejecute la aplicación de forma que en primer lugar se muestra la página intermedia solicitando confirmación relativa a los permisos que necesita para su ejecución. Note como se especifican los permisos de lectura a nivel de Sitio y de Lista. Pulse el botón “Confiar”.



* Compruebe como la Aplicación se ejecuta de forma correcta y muestra la información que se espera:



## Uso del CSOM para SPO

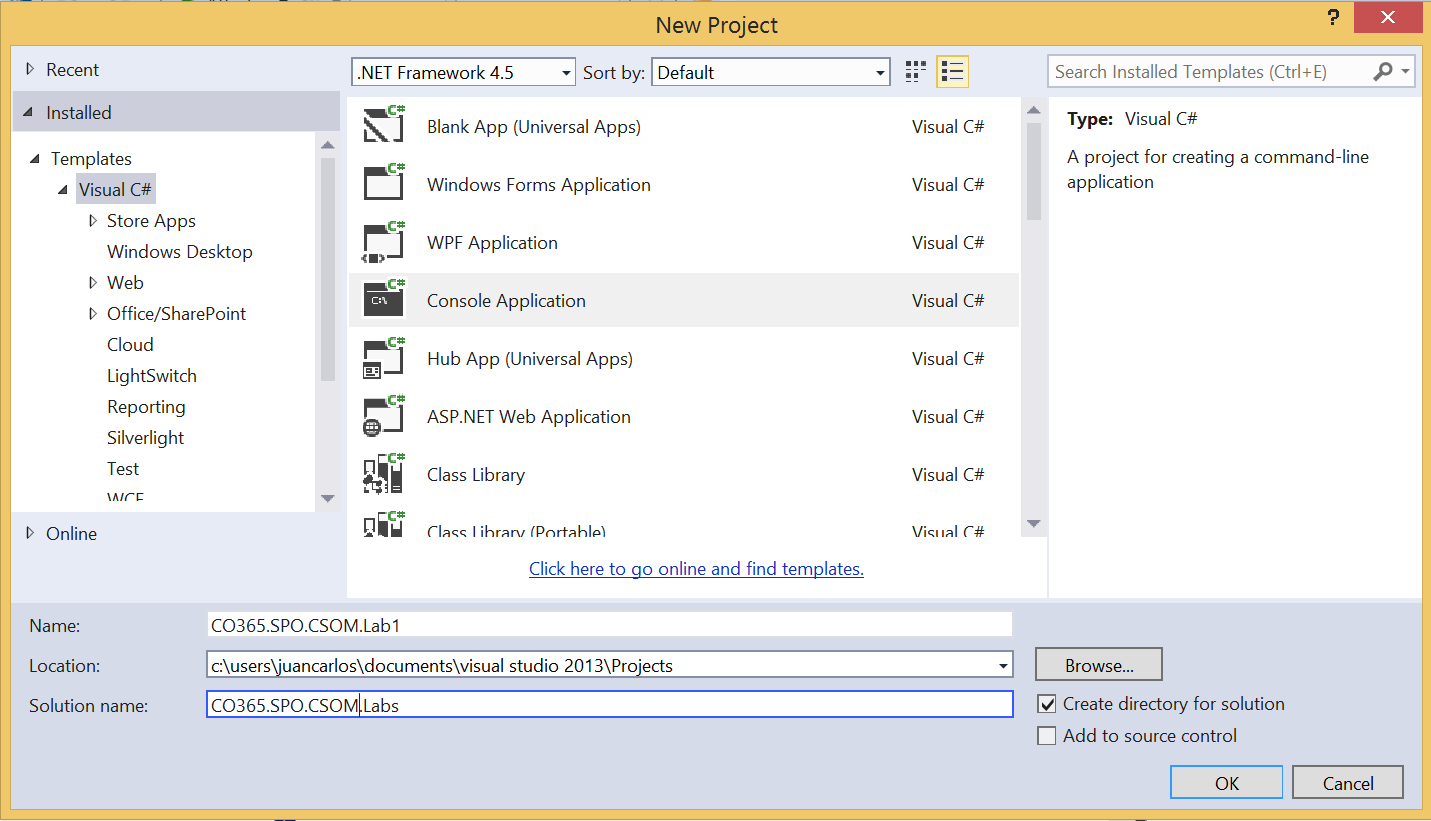
En esta sección vamos a ver ejemplos de cómo hacer uso del CSOM para SPO. Antes de empezar, necesitaremos disponer de los ensamblados del CSOM:

* En primer lugar, tenemos que descargarnos e instalar en nuestro equipo de trabajo el [SharePoint Online Cliente SDK](•%09En%20primer%20lugar,%20tenemos%20que%20descargarnos%20e%20instalar%20en%20nuestro%20equipo%20de%20trabajo%20el%20SharePoint%20Server%202013%20Client%20Components%20SDK%20que%20se%20puede%20descargar%20desde%20el%20siguiente%20enlace:%20http:/www.microsoft.com/en-us/download/details.aspx?id=35585%20y%20también%20el%20SharePoint%20Server%202013%20Client%20Components%20SDK.%20(Nota:%20Opcionalmente,%20si%20se%20dispone%20de%20un%20entorno%20de%20SharePoint%202013%20se%20pueden%20copiar%20del%20mismo%20los%20ensamblados%20relativos%20al%20CSOM).) y también el [SharePoint Server 2013 Client Components SDK](http://www.microsoft.com/en-ie/download/details.aspx?id=35585). (**Nota:** Opcionalmente, si se dispone de un entorno de SharePoint 2013 se pueden copiar del mismo los ensamblados relativos al CSOM).
* A continuación, nos aseguramos de que tenemos perfectamente localizada la ruta de dichos ensamblados en nuestro equipo local ya que la necesitaremos posteriormente para hacer uso de los objetos del CSOM en nuestros scripts PowerShell.

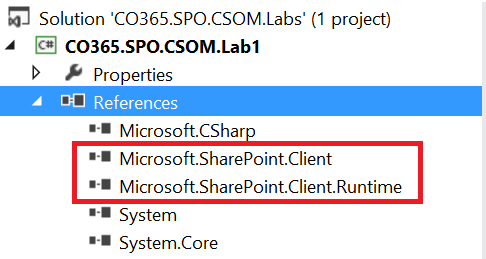
### Consultar los elementos de una lista con CSOM especificando las credenciales de acceso

El primer ejemplo a realizar consiste en crear una Aplicación de Consola que haga uso del CSOM de SPO para consultar los elementos de una lista especificando las credenciales de acceso a la lista:

* Inicie Visual Studio 2013 y cree un proyecto de tipo Aplicación de Consola.



* Agregamos referencias a Microsoft.SharePoint.Client, Microsoft.SharePoint.Client.Runtime y System.Configuration.



* En Program.cs añada una directiva using a Microsoft.SharePoint.Client, System.Security y System.Configuration:

|  |
| --- |
| using SPCSOM = Microsoft.SharePoint.Client;  using System.Security;  using System.Configuration; |

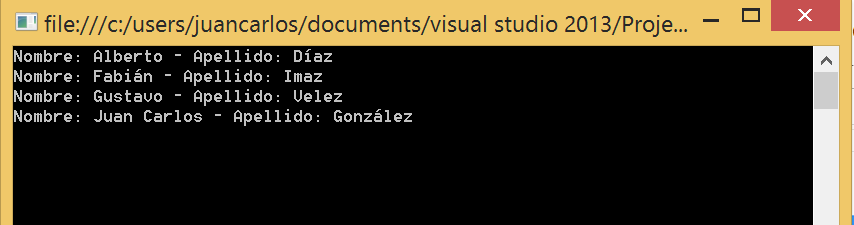
* Añada el siguiente método estático que permite consultar los elementos de la lista “Contactos Curso Office 365” y mostrarlos por pantalla.

|  |
| --- |
| static void ConsultarContactosCSOM()  {  try  {  string sSiteUrl = "https://itechcs.sharepoint.com/sites/CO365DeveloperSite/";  using (SPCSOM.ClientContext spoCtx=new SPCSOM.ClientContext(sSiteUrl))  {  //  //SharePoint Online Credentials  //  string sSPOUser =  ConfigurationManager.AppSettings["SPOUser"];  string sPassword=  ConfigurationManager.AppSettings["SPOPassword"];  SecureString ssPassword = new SecureString();  foreach (char c in sPassword.ToCharArray())  ssPassword.AppendChar(c);  spoCtx.Credentials =  new SPCSOM.SharePointOnlineCredentials(  sSPOUser, ssPassword);  SPCSOM.List spoList =  spoCtx.Web.Lists.GetByTitle("Contactos Curso Office 365");  spoCtx.Load(spoList);  spoCtx.ExecuteQuery();  if (spoList!=null && spoList.ItemCount>0)  {  SPCSOM.CamlQuery spocqConsulta =  new SPCSOM.CamlQuery();  spocqConsulta.ViewXml =  @"<View>  <Query>  <OrderBy>  <FieldRef Name='FirstName' />  </OrderBy>  </Query>  <ViewFields>  <FieldRef Name='LinkTitle' />  <FieldRef Name='FirstName' />  </ViewFields>  </View>";  SPCSOM.ListItemCollection spoliItemCollection =  spoList.GetItems(spocqConsulta);  spoCtx.Load(spoliItemCollection);  spoCtx.ExecuteQuery();  //Procesado de los datos  foreach (SPCSOM.ListItem spoliItem in spoliItemCollection)  {  Console.WriteLine("Nombre: {0} - Apellido: {1}",  spoliItem["FirstName"].ToString(),  spoliItem["Title"].ToString());  }  }  }  }  catch (Exception ex)  {  Console.WriteLine("Error: {0}", ex.Message);  }  } |

* Modifique el archivo App.Config del proyecto para añadir las propiedades relativas al usuario y contraseña para SharePoint Online de acuerdo a la siguiente estructura:

|  |
| --- |
| <?xml version="1.0" encoding="utf-8" ?>  <configuration>  <startup>  <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />  </startup>  <appSettings>  <add key="SPOUser" value="<SPOUser>"/>  <add key="SPOPassword" value="<SPOPassword>"/>  </appSettings>  </configuration> |

* Añada una llamada al método estático en el método Main() de la Aplicación de Consola, ejecute el proyecto y compruebe que la salida generada es la esperada.



### Operaciones CRUD en Listas con CSOM especificando las credenciales de acceso

Continuando con el ejemplo anterior, en esta sección vamos a ver cómo realizar operaciones CRUD con elementos de lista haciendo uso del CSOM para SPO:

* Añada el siguiente método estático que a Program.cs que permite realizar operaciones de tipo CRUD con listas de SharePoint usando el CSOM.

|  |
| --- |
| static void OperacionesCRUD(string sTipoDeOperacion)  {  try  {  string sSiteUrl = "https://itechcs.sharepoint.com/sites/CO365DeveloperSite/";  using (SPCSOM.ClientContext spoCtx = new SPCSOM.ClientContext(sSiteUrl))  {  //  //SharePoint Online Credentials  //  string sSPOUser =  ConfigurationManager.AppSettings["SPOUser"];  string sPassword =  ConfigurationManager.AppSettings["SPOPassword"];  SecureString ssPassword = new SecureString();  foreach (char c in sPassword.ToCharArray())  ssPassword.AppendChar(c);  spoCtx.Credentials =  new SPCSOM.SharePointOnlineCredentials(  sSPOUser, ssPassword);  SPCSOM.List spoList =  spoCtx.Web.Lists.GetByTitle("Contactos Curso Office 365");  spoCtx.Load(spoList);  spoCtx.ExecuteQuery();  if (spoList != null && spoList.ItemCount > 0)  {  switch (sTipoDeOperacion)  {  //Crear  case "C":  Console.WriteLine("\*\*\*\*Creando Elemento en la lista\*\*\*\*");  SPCSOM.ListItemCreationInformation liciContacto =  new SPCSOM.ListItemCreationInformation();  SPCSOM.ListItem liElemento =  spoList.AddItem(liciContacto);  liElemento["FirstName"] = "Adrián";  liElemento["Title"] = "Díaz";  liElemento.Update();  spoCtx.ExecuteQuery();  ConsultarContactosCSOM();  break;  //Crear  case "A":  Console.WriteLine("\*\*\*\*Actualizando Elemento en la lista\*\*\*\*");  SPCSOM.ListItem liElementoAActualizar =  spoList.GetItemById(4);  liElementoAActualizar["FirstName"] = "Adrián";  liElementoAActualizar["Title"] = "Díaz Cervera";  liElementoAActualizar.Update();  spoCtx.ExecuteQuery();  ConsultarContactosCSOM();  break;  //Crear  case "B":  Console.WriteLine("\*\*\*\*Borrando Elemento en la lista\*\*\*\*");  SPCSOM.ListItem liElementoABorrar =  spoList.GetItemById(4);  liElementoABorrar.DeleteObject();  spoCtx.ExecuteQuery();  ConsultarContactosCSOM();  break;  default:  break;  }  }  }  }  catch (Exception ex)  {  Console.WriteLine("Error: {0}", ex.Message);  }  } |

* Añada las siguientes llamadas al método estático en el método Main() de Program.cs y compruebe que las operaciones se realizan de forma correcta.

|  |
| --- |
| OperacionesCRUD("C");  OperacionesCRUD("A");  OperacionesCRUD("B"); |

### Simulación de un Timer Job con el CSOM sin especificar credenciales de acceso

Como alternativa a hacer uso del CSOM de SPO especificando las credenciales de acceso, vamos a ver cómo es posible hacer uso del mismo sin especificar dichas credenciales y haciendo uso de las capacidades que introduce el Modelo de Aplicaciones para SPO.

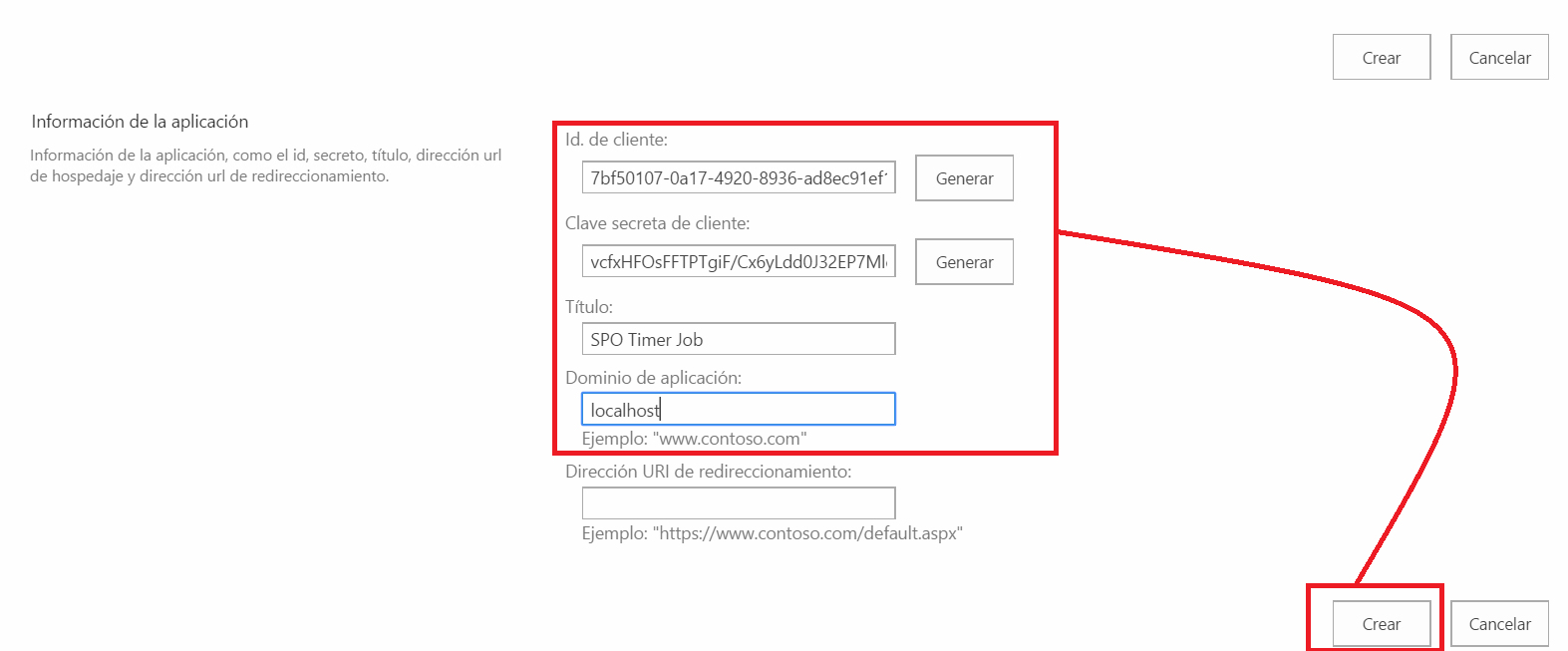
#### Creación de un App Security Principal y Configuración de los permisos

Lo primero que tendremos que hacer es configurar a nivel de SPO un App Security Principal y sus permios de forma que pueda ser utilizado por la Aplicación de Consola para poder interactuar con el sitio de SPO:

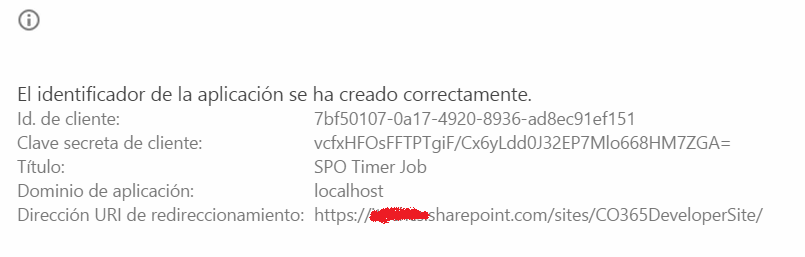
* Accedemos al sitio de desarrollador creado en el apartado 2.1.1 del laboratorio y a continuación indicamos la siguiente Url en el navegador:

|  |
| --- |
| https://<DominioOffice365>.sharepoint.com/sites/CO365DeveloperSite/**\_layouts/15/appregnew.aspx** |

* A continuación se muestra la página que nos permite crear un nuevo App Security Principal para ser utilizado en una App. En esta página:
  + Hacemos clic en el botón “Generar” de Id. de cliente.
  + Repetimos la operación para Clave secreta de cliente.
  + Especificamos “SPO Timer Job” en Título.
  + Añadimos “localhost” en Dominio de aplicación.
  + Finalmente, en el campo Dirección URI de redireccionamiento especificamos la dirección de la página principal del sitio de desarrollador.



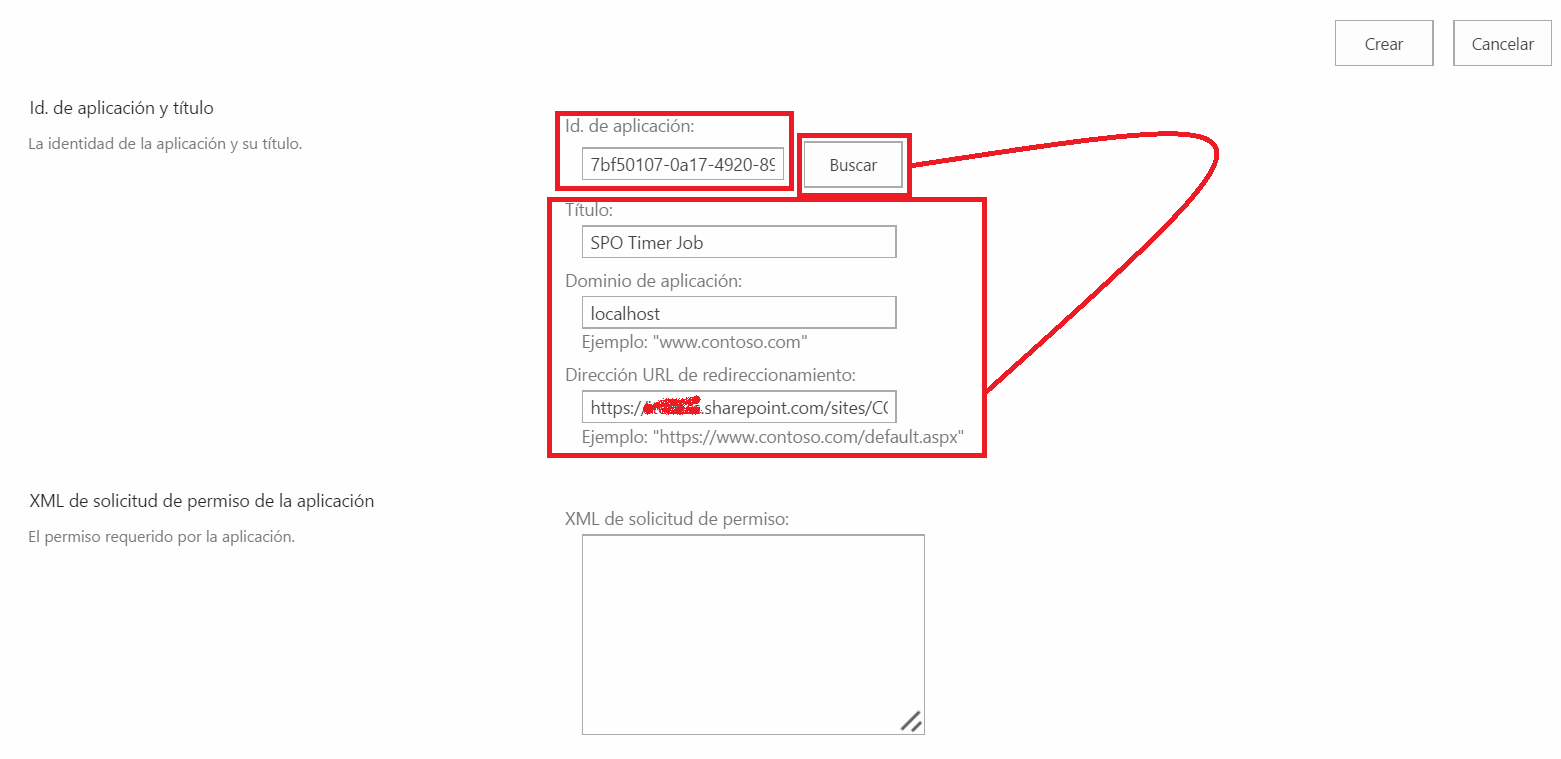
* Copiamos los valores del Id. de Cliente y Clave secreta de cliente en un archivo del bloc de notas (también aparecerán en la siguiente página) puesto que los necesitaremos más adelante y pulsamos “Crear”. A continuación se muestra la información del App Security Principal generado.



* Hacemos clic en “Aceptar” y navegamos a la página de aplicación de SPO que nos permite configurar un App Security Principal:

|  |
| --- |
| https://<DominioOffice365>.sharepoint.com/sites/CO365DeveloperSite/**\_layouts/15/appinv.aspx** |

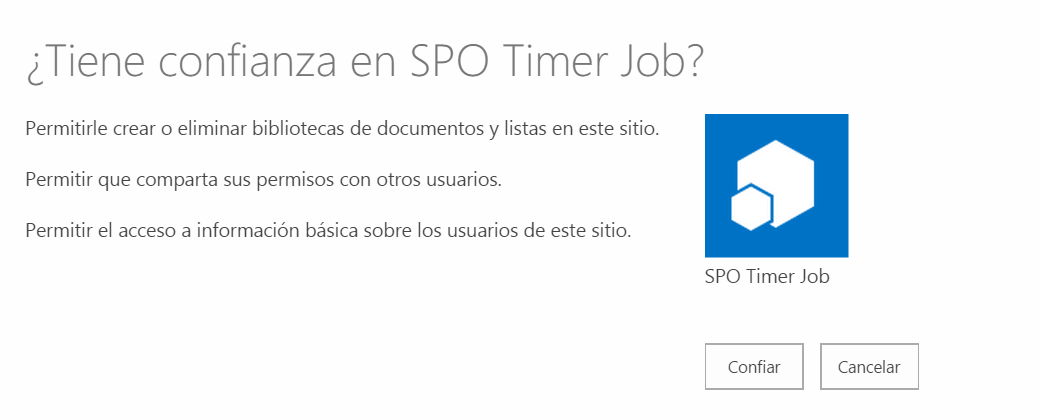
* En la página que se muestra, realizamos las siguientes configuraciones:
  + Id. de aplicación, pegamos el valor generado en la página **appregnew.aspx**.Al pulsar el botón “Buscar”, la página devuelve el resto de información relativo al App Security Principal creado.



* + En la sección “XML de solicitud de permiso” pegamos la siguiente definición XML y pulsamos el botón “Crear”. Como se puede apreciar, estamos configurando la App para que se pueda ejecutar sin necesidad de especificar las credenciales del usuario. De esta forma, estamos concediendo los permisos específicos para la App que se creará en la siguiente sección.

|  |
| --- |
| <AppPermissionRequests AllowAppOnlyPolicy="true">  <AppPermissionRequest  Scope="http://sharepoint/content/sitecollection/web"  Right="Manage" />  </AppPermissionRequests> |

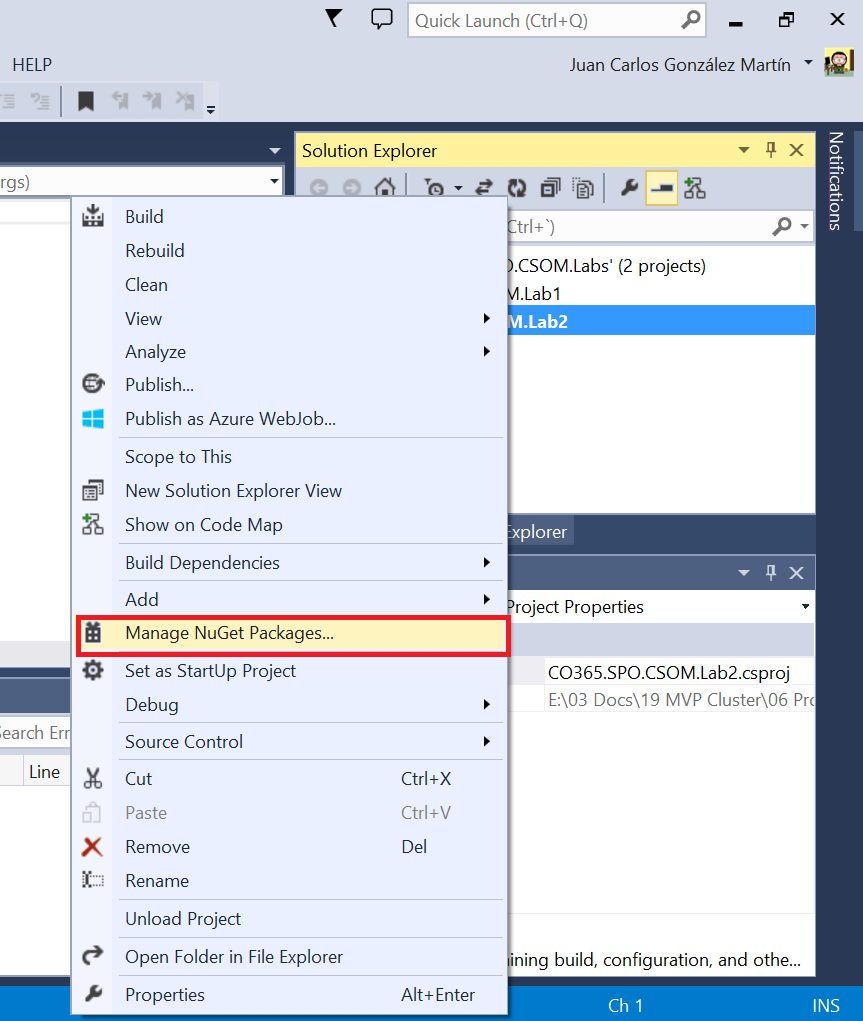
* A continuación, se muestra la típica página en la que se pide confirmación en torno al uso de la Aplicación. Hacemos clic en “Confiar”.



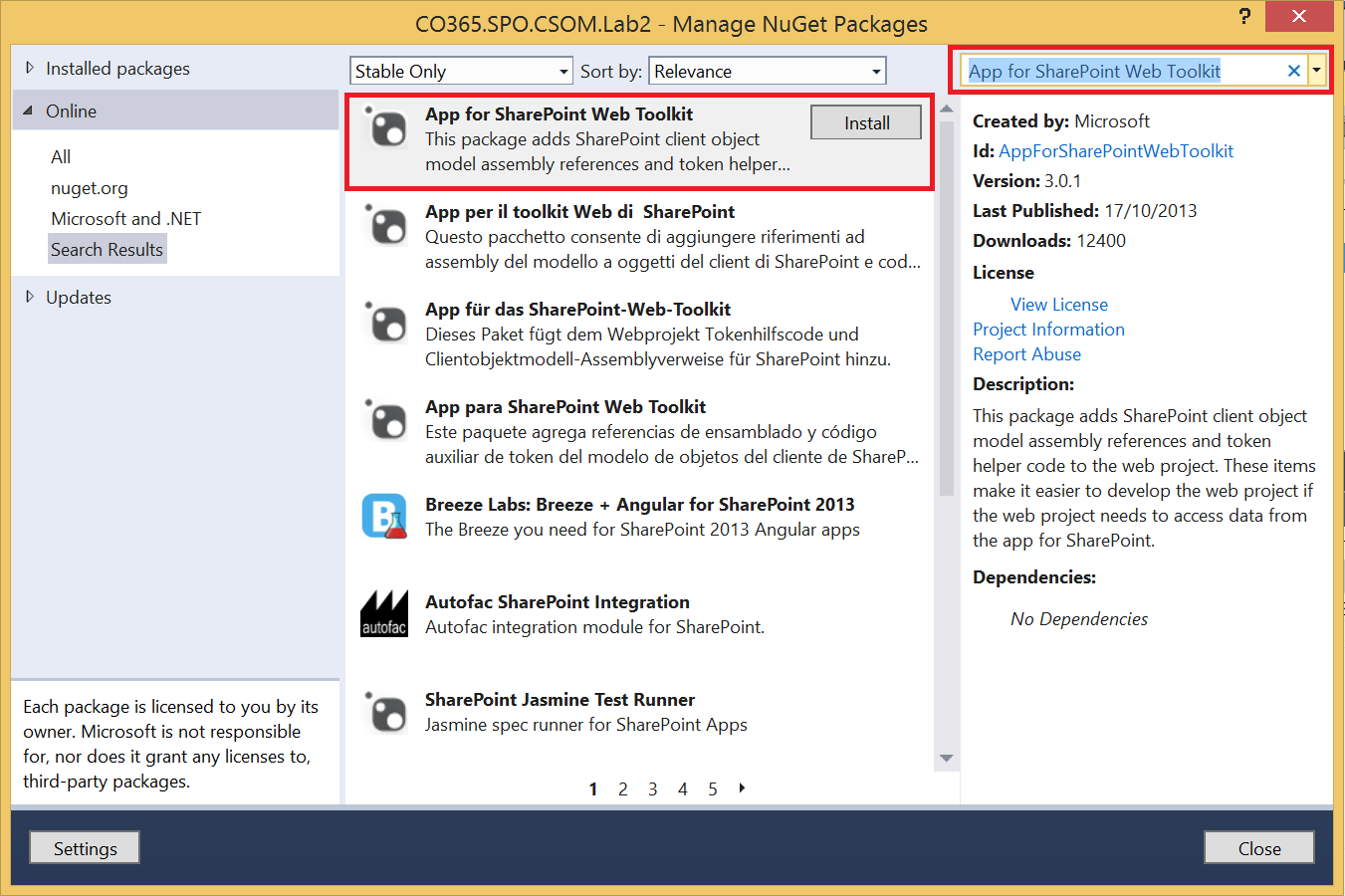
#### Creación del proyecto de Aplicación de Consola

En esta sección vamos a crear la Aplicación de Consola que permite simular un Timer Job para SPO:

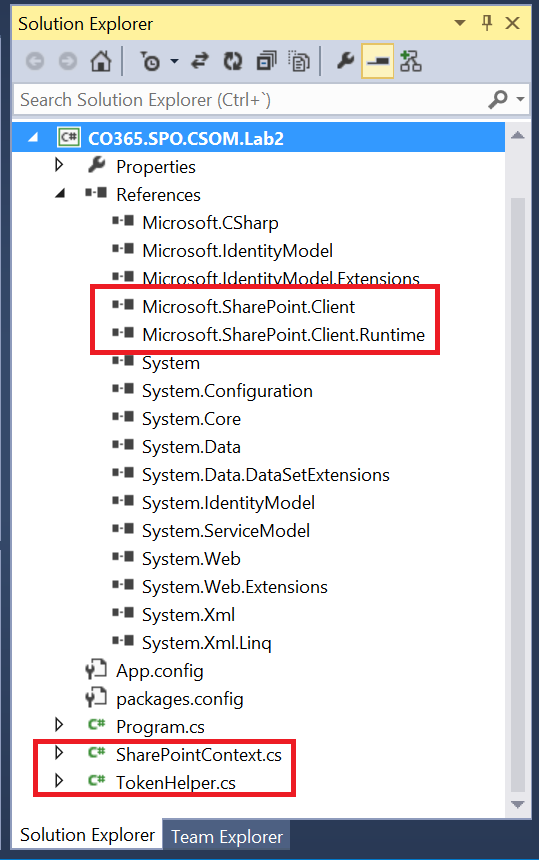
* En Visual Studio creamos un proyecto de tipo Aplicación de Consola. Seleccionamos el nombre del proyecto, hacemos clic con el botón derecho del ratón y a continuación hacemos clic en la opción “Manage NuGet Packages…”.



* En la ventana de “Manage NuGet Packages…” bucamos el paquete App for SharePoint Web Toolkit y hacemos clic en “Install”.



* Una vez se instala el paquete, podremos comprobar a través del explorador de soluciones que se han añadido nuevos elementos al proyecto como:
  + Referencias al modelo de objetos en cliente (ensamblados Microsoft.SharePoint.Client y Microsoft.SharePoint.Runtime).
  + Clases para trabajar con métodos y objetos específicos para el Modelo de Apps para SPO: clase SharePointContext y clase TokenHelper.



* Dentro del proyecto de aplicación de consola, localizamos el archivo App.Config y lo modificamos para incluir la siguiente sección:

|  |
| --- |
| <appSettings>  <add key="ClientId" value="[[YOUR\_VALUE]]"/>  <add key="ClientSecret" value="[[YOUR\_VALUE]]"/>  </appSettings> |

* Modificamos los valores de los atributos ClientId y ClientSecret generados en la sección 2.3.3.1 del manual de forma que el archivo de configuración debería quedar de la siguiente forma:

|  |
| --- |
| <?xml version="1.0" encoding="utf-8" ?>  <configuration>  <startup>  <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />  </startup>  <appSettings>  <add key="ClientId" value="7bf50107-0a17-4920-8936-ad8ec91ef151"/>  <add key="ClientSecret" value="vcfxHFOsFFTPTgiF/Cx6yLdd0J32EP7Mlo668HM7ZGA="/>  </appSettings>  </configuration> |

* Añadimos al proyecto las siguiente directiva using en Program.cs:

|  |
| --- |
| using SPCSOM=Microsoft.SharePoint.Client; |

* Añadimos a continuación el siguiente método estático que permite combinar por un lado la URL relativa del sitio, y por otro la Url relativa de los archivos de tema que se van a usar para cambiar el aspecto de un sitio

|  |
| --- |
| private static string URLCombine(string baseUrl, string relativeUrl)  {  if (baseUrl.Length == 0)  return relativeUrl;  if (relativeUrl.Length == 0)  return baseUrl;  return string.Format("{0}/{1}",  baseUrl.TrimEnd(new char[] { '/', '\\' }),  relativeUrl.TrimStart(new char[] { '/', '\\' }));  } |

* Añadimos un nuevo método estático que permite cambiar el tema del sitio:

|  |
| --- |
| private static void ApplyTheme(SPCSOM.ClientContext spoCtx)  {  try  {  SPCSOM.Web wWeb = spoCtx.Web;  spoCtx.Load(wWeb);  spoCtx.ExecuteQuery();  //Aplicación del tema Sketch  wWeb.ApplyTheme(  URLCombine(wWeb.ServerRelativeUrl,  "/\_catalogs/theme/15/palette007.spcolor"),  URLCombine(wWeb.ServerRelativeUrl,  "/\_catalogs/theme/15/fontscheme002.spfont"),  URLCombine(wWeb.ServerRelativeUrl,  "/\_layouts/15/images/image\_bg007.jpg"),  false);  spoCtx.ExecuteQuery();  }  catch (Exception ex)  {  Console.WriteLine("Error: {0}", ex.Message);  }  } |

* Modificamos el método Main() de Program.cs para que ejecute el método estático anterior:

|  |
| --- |
| Uri siteUri =  new Uri("https://<O365Domain>.sharepoint.com/<SiteCollectionPath>");  string realm =  TokenHelper.GetRealmFromTargetUrl(siteUri);  string accessToken =  TokenHelper.GetAppOnlyAccessToken(  TokenHelper.SharePointPrincipal,  siteUri.Authority, realm).AccessToken;  using (var spoCtx = TokenHelper.GetClientContextWithAccessToken(siteUri.ToString(), accessToken))  {  //ApplyTheme(spoCtx);  //Console.WriteLine("El tema ha sido actualizado...");  ConsultarContactosCSOM(spoCtx);  Console.ReadLine();  } |

* Ejecutamos en modo depuración el código y comprobamos qué el tema se cambia en el sitio de Office 365.



### Referencias

* <https://github.com/OfficeDev/TrainingContent/blob/master/O3651-4%20Moving%20Full%20Trust%20Code%20to%20the%20cloud%20using%20repeatable%20patterns%20and%20best%20practices/Lab.md>

# Uso de las APIs de SPO en PowerShell

Esta sección trata de mostrar las posibilidades que PowerShell para hacer uso de las APIs de Cliente (CSOM y REST) de SPO para realizar operaciones comunes.

## Entorno de PowerShell para SharePoint Online

Para poder administrar SharePoint Online por medio de PowerShell [es necesario cumplir unos pre-requisitos que pasan por](http://office.microsoft.com/en-us/sharepoint-help/set-up-the-sharepoint-online-management-shell-environment-HA102919087.aspx?CTT=5&origin=HA102915057) que pasan por:

* Descargar e instalar [Windows Management Framework 3.0](http://www.microsoft.com/en-us/download/details.aspx?id=34595).
* Descargar e instalar [SharePoint Online Management Shell](http://www.microsoft.com/en-us/download/details.aspx?id=35588).

**Nota:** La instalación de estos componentes se pude realizar tanto en un equipo con S.O de cliente (Windows 7 en adelante) como de servidor (Windows Server 208 R2) en adelante.

Una vez que tenemos listo el entorno de Administración de SharePoint Online con PowerShell, el siguiente paso consiste en elegir la herramienta de Administración que se va a utilizar. Las posibilidades son dos:

* El SharePoint Online Management Shell.
* El entorno aportado por el ISE (Integrated Shell Script Environment)

### SharePoint Online Management Shell

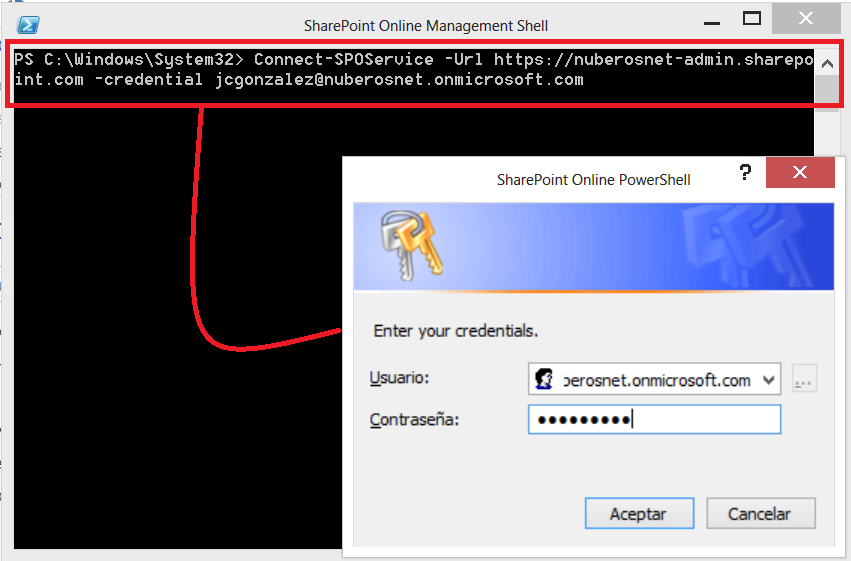
Para administrar SharePoint Online con el SharePoint Online Management Shell, tenemos que seguir los siguientes pasos:

* Iniciamos el Shell y nos conectamos en primer lugar a nuestro tenant de SharePoint Online de Office 365 haciendo uso del siguiente comando:

|  |
| --- |
| Connect-SPOService -Url https://<Dominio\_Office365>-admin.sharepoint.com -credential usuario@<Dominio\_Office365>.com |

Dónde:

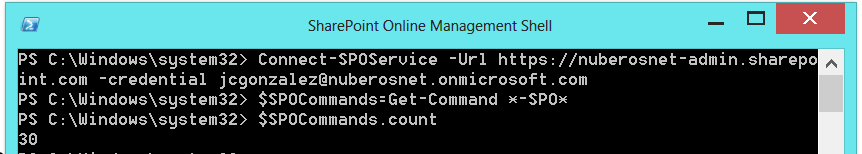
* + Url es la URL de la Centro de Administración de SharePoint Online.
  + Credential es el nombre del usuario que va a realizar la administración de SharePoint Online desde línea de comandos.
* Una vez ejecutado el comando anterior, tendremos que introducir las credenciales de Office 365:



* Para comprobar que la conexión a SharePoint Online es correcta, simplemente ejecutamos la siguiente sentencia PowerShell que nos devuelve todos los comandos PowerShell disponibles para SharePoint Online:

|  |
| --- |
| $SPOCommands=Get-Command \*-SPO\*  $SPOCommands.count |

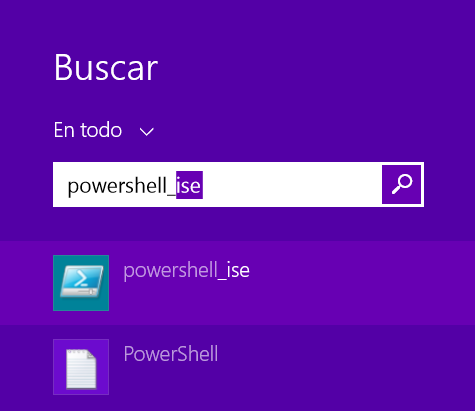
* La salida por pantalla correspondiente es la siguiente



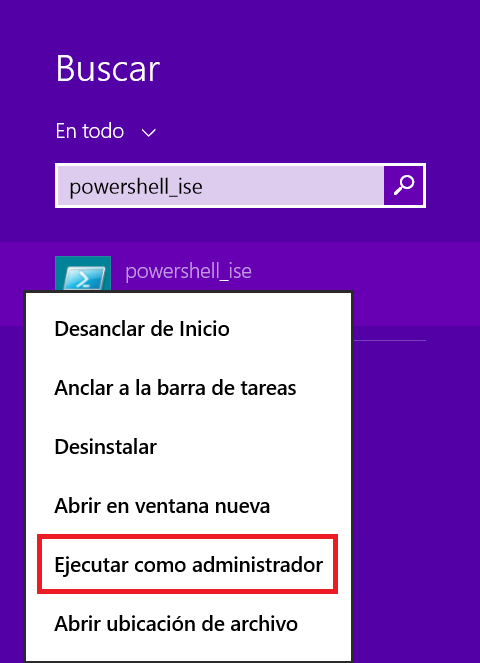
### ISE

Para poder ejecutar Scripts PowerShell para SharePoint Online por medio del ISE, tenemos que seguir los siguientes pasos:

* Ejecutar el ISE en nuestro equipo de trabajo. En el caso de qué se vaya a ejecutar PowerShell ISE en equipos con Windows 8 / 8.1 instalado, es necesario localizar en primer lugar el ISE haciendo uso de las opciones de búsqueda del Sistema Operativo como muestra la siguiente figura:



* Ejecutamos powershell\_ise como administradores para poder tener acceso al juego de comandos PowerShell por defecto para SPO:



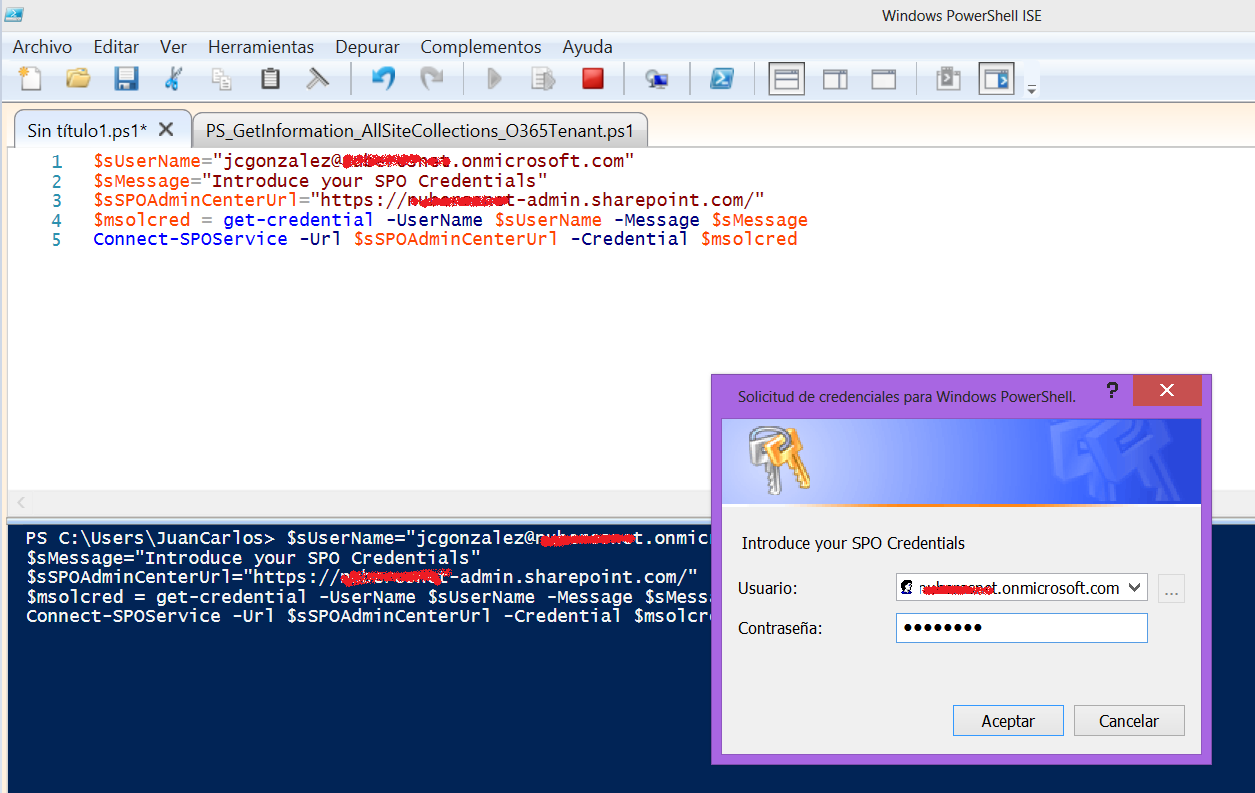
* Una vez que tenemos abierto PowerShell ISE, en el editor de scripts añadimos las siguientes sentencias reemplazando la cuenta de Office 365 a utilizar por la que corresponda:

|  |
| --- |
| $sUserName="<Usuario>@<Dominio\_Office365>.onmicrosoft.com"  $sMessage="Introduce your SPO Credentials"  $sSPOAdminCenterUrl="https://<Dominio\_Office365>-admin.sharepoint.com/"  $msolcred = get-credential -UserName $sUserName -Message $sMessage  Connect-SPOService -Url $sSPOAdminCenterUrl -Credential $msolcred |

**Notas:**

1. Es recomendable revisar el siguiente post sobre el [uso de Windows PowerShell ISE en Windows 8](http://geeks.ms/blogs/jcgonzalez/archive/2014/03/07/powershell-como-hacer-uso-del-ise-en-windows-8.aspx) en caso de encontrar problemas a la hora de ejecutar el ISE en un equipo con Windows 8 instalado.
2. El número de comandos PowerShell disponible para SharePoint Online puede variar en el tiempo, ya que Microsoft puede añadir nuevos comandos.

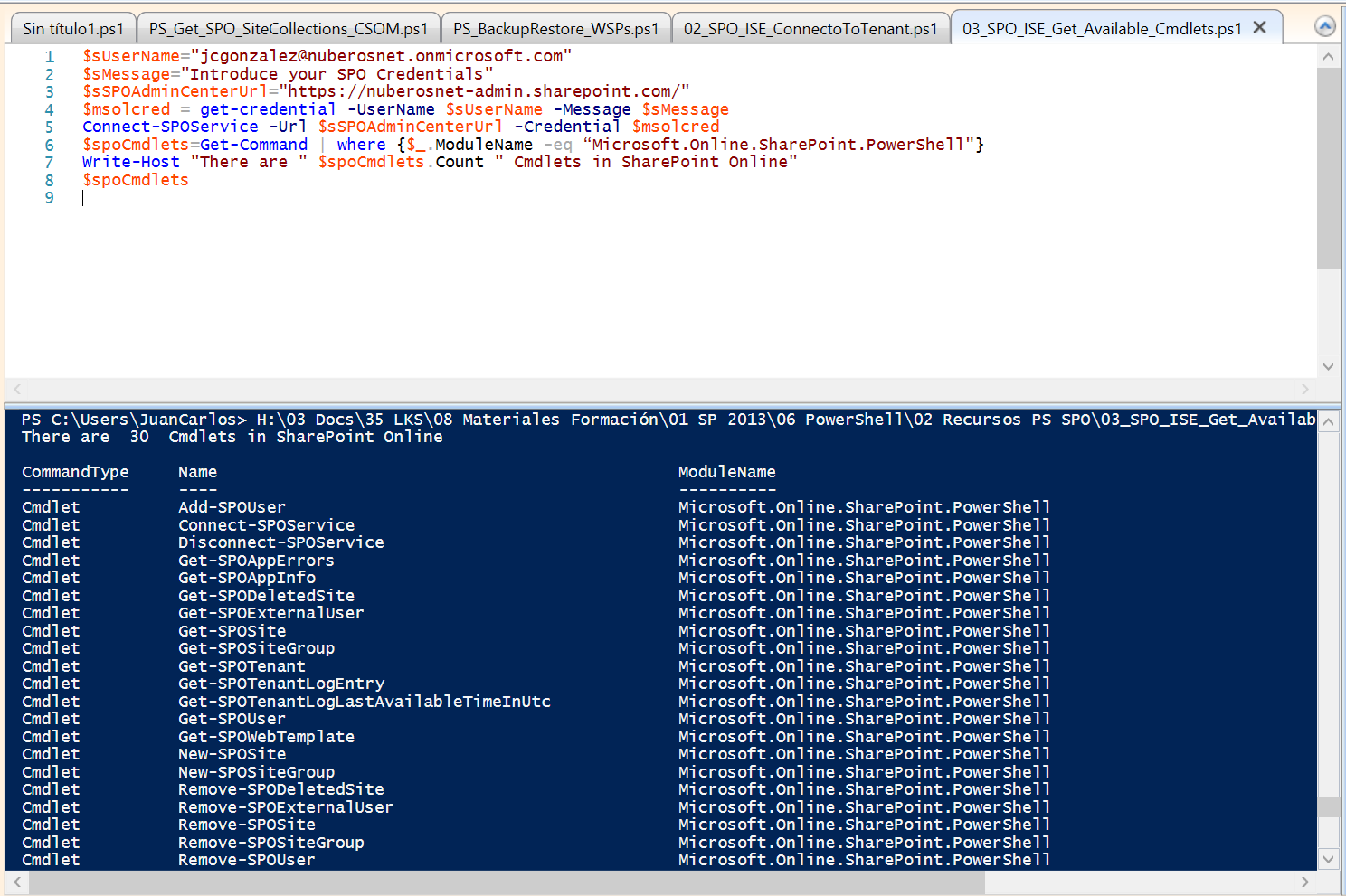
* La ejecución de la secuencia anterior produce como resultado la petición de credenciales de acceso al tenant de SharePoint Online en Office 365:



* A partir de aquí, ya estamos listos para poder hacer uso de los comandos PowerShell disponibles para SharePoint Online. Por ejemplo, para obtener un listado rápido de los mismos añadimos la siguiente secuencia PowerShell a la secuencia anterior:

|  |
| --- |
| $spoCmdlets=Get-Command | where {$\_.ModuleName -eq “Microsoft.Online.SharePoint.PowerShell"}  Write-Host "There are " $spoCmdlets.Count " Cmdlets in SharePoint Online"  $spoCmdlets |

* La salida por pantalla que se debería obtener es la siguientes:



## Uso del Modelo de Objetos en Cliente de SharePoint desde PowerShell

Para poder hacer uso del Modelo de Objetos en Cliente (CSOM) de SharePoint en PowerShell, necesitamos una serie de pre-requisitos:

* En primer lugar, tenemos que descargarnos e instalar en nuestro equipo de trabajo el [SharePoint Online Cliente SDK](•%09En%20primer%20lugar,%20tenemos%20que%20descargarnos%20e%20instalar%20en%20nuestro%20equipo%20de%20trabajo%20el%20SharePoint%20Server%202013%20Client%20Components%20SDK%20que%20se%20puede%20descargar%20desde%20el%20siguiente%20enlace:%20http:/www.microsoft.com/en-us/download/details.aspx?id=35585%20y%20también%20el%20SharePoint%20Server%202013%20Client%20Components%20SDK.%20(Nota:%20Opcionalmente,%20si%20se%20dispone%20de%20un%20entorno%20de%20SharePoint%202013%20se%20pueden%20copiar%20del%20mismo%20los%20ensamblados%20relativos%20al%20CSOM).) y también el [SharePoint Server 2013 Client Components SDK](http://www.microsoft.com/en-ie/download/details.aspx?id=35585). (**Nota:** Opcionalmente, si se dispone de un entorno de SharePoint 2013 se pueden copiar del mismo los ensamblados relativos al CSOM).
* A continuación, nos aseguramos de que tenemos perfectamente localizada la ruta de dichos ensamblados en nuestro equipo local ya que la necesitaremos posteriormente para hacer uso de los objetos del CSOM en nuestros scripts PowerShell.

## Como utilizar el CSOM de SPO en PowerShell

Para poder hacer uso del CSOM de SPO en PowerShell:

* Una vez que tenemos disponibles los ensamblados, tenemos que cargar los mismos en el entorno de ejecución de PowerShell utilizando la siguiente secuencia de comandos que hacen uso de Add-Type para agregar un ensamblado a partir de la ruta del mismo:

|  |
| --- |
| Add-Type -Path "<CSOM\_Path>\Microsoft.SharePoint.Client.dll"  Add-Type -Path "<CSOM\_Path>\Microsoft.SharePoint.Client.Runtime.dll"  Add-Type -Path "<CSOM\_Path>\Microsoft.Online.SharePoint.Client.Tenant.dll" |

* A partir de aquí, sólo hay que seguir las reglas conocidas en el uso del Modelo de Objetos en Cliente:
  + Definir en primer lugar un objeto de tipo ClientContext a partir de la Url de una Colección de Sitios de SharePoint Online.
  + Indicar unas credenciales de conexión validas a la Colección en cuestión.
  + Comenzar a realizar operaciones en la Colección teniendo en cuenta que en primer lugar hay que inicializar las operaciones mediante el método Load() de ClientContext para a continuación proceder a ejecutar dicha operación con el método ExecuteQuery().

## Como obtener todas las Colecciones de Sitios del tenant mediante CSOM

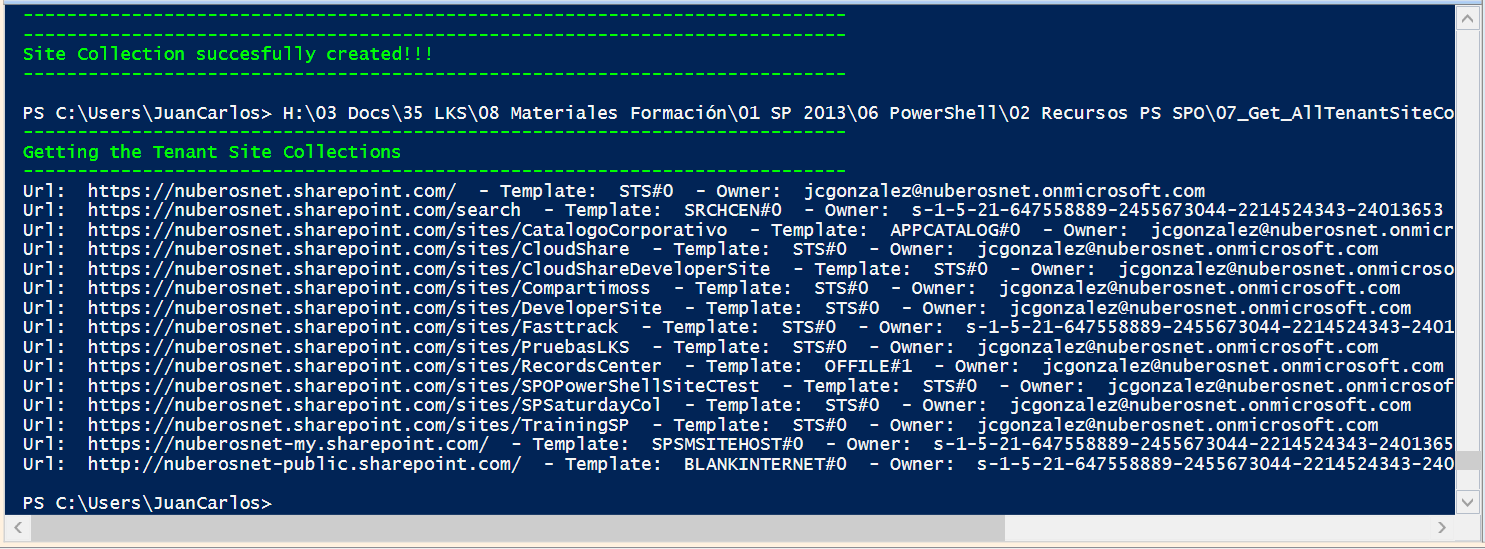
Para listar todas las Colecciones de Sitios a nivel de Tenant de SharePoint Online:

* En el ISE, añadimos el siguiente script:

|  |
| --- |
| $host.Runspace.ThreadOptions = "ReuseThread"  #Definition of the function that gets the list of site collections in the tenant using CSOM  function Get-SPOTenantSiteCollections  {  param ($sCSOMPath,$sSiteUrl,$sUserName,$sPassword)  try  {  Write-Host "----------------------------------------------------------------------------" -foregroundcolor Green  Write-Host "Getting the Tenant Site Collections" -foregroundcolor Green  Write-Host "----------------------------------------------------------------------------" -foregroundcolor Green    #Adding the Client OM Assemblies  $sCSOMRuntimePath=$sCSOMPath + "\Microsoft.SharePoint.Client.Runtime.dll"  $sCSOMTenantPath=$sCSOMPath + "\Microsoft.Online.SharePoint.Client.Tenant.dll"  $sCSOMPath=$sCSOMPath + "\Microsoft.SharePoint.Client.dll"  Add-Type -Path $sCSOMPath  Add-Type -Path $sCSOMRuntimePath  Add-Type -Path $sCSOMTenantPath  #SPO Client Object Model Context  $spoCtx = New-Object Microsoft.SharePoint.Client.ClientContext($sSiteUrl)  $spoCredentials = New-Object Microsoft.SharePoint.Client.SharePointOnlineCredentials($sUsername, $sPassword)  $spoCtx.Credentials = $spoCredentials  $spoTenant= New-Object Microsoft.Online.SharePoint.TenantAdministration.Tenant($spoCtx)  $spoTenantSiteCollections=$spoTenant.GetSiteProperties(0,$true)  $spoCtx.Load($spoTenantSiteCollections)  $spoCtx.ExecuteQuery()    #We need to iterate through the $spoTenantSiteCollections object to get the information of each individual Site Collection  foreach($spoSiteCollection in $spoTenantSiteCollections){    Write-Host "Url: " $spoSiteCollection.Url " - Template: " $spoSiteCollection.Template " - Owner: " $spoSiteCollection.Owner -ForegroundColor Red  }  $spoCtx.Dispose()  }  catch [System.Exception]  {  write-host -f red $\_.Exception.ToString()  }  }  #Required Parameters  $sSiteUrl = "https://<O365Domain>-admin.sharepoint.com/"  $sUserName = "<SPOUser>@<O365Domain>.onmicrosoft.com"  $sPassword = Read-Host -Prompt "Enter your password: " -AsSecureString  $sCSOMPath="<CSOM\_Assemblies\_Path"  Get-SPOTenantSiteCollections -sCSOMPath $sCSOMPath -sSiteUrl $sSiteUrl -sUserName $sUserName -sPassword $sPassword |

Como se puede apreciar, resulta muy sencillo listar las colecciones de sitios de un tenant de SPO gracias a los objetos disponibles en el ensamblado [Microsoft.Online.SharePoint.Client.Tenant](http://msdn.microsoft.com/en-us/library/microsoft.online.sharepoint.tenantadministration(v=office.15).aspx) que nos proporciona un objeto global “Tenant” que entre otras cosas dispone del método GetSiteProperties() que nos permite acceder al listado de colecciones de sitios de nuestro Tenant de SharePoint Online.

* La salida por pantalla correspondiente es la que se muestra a continuación:



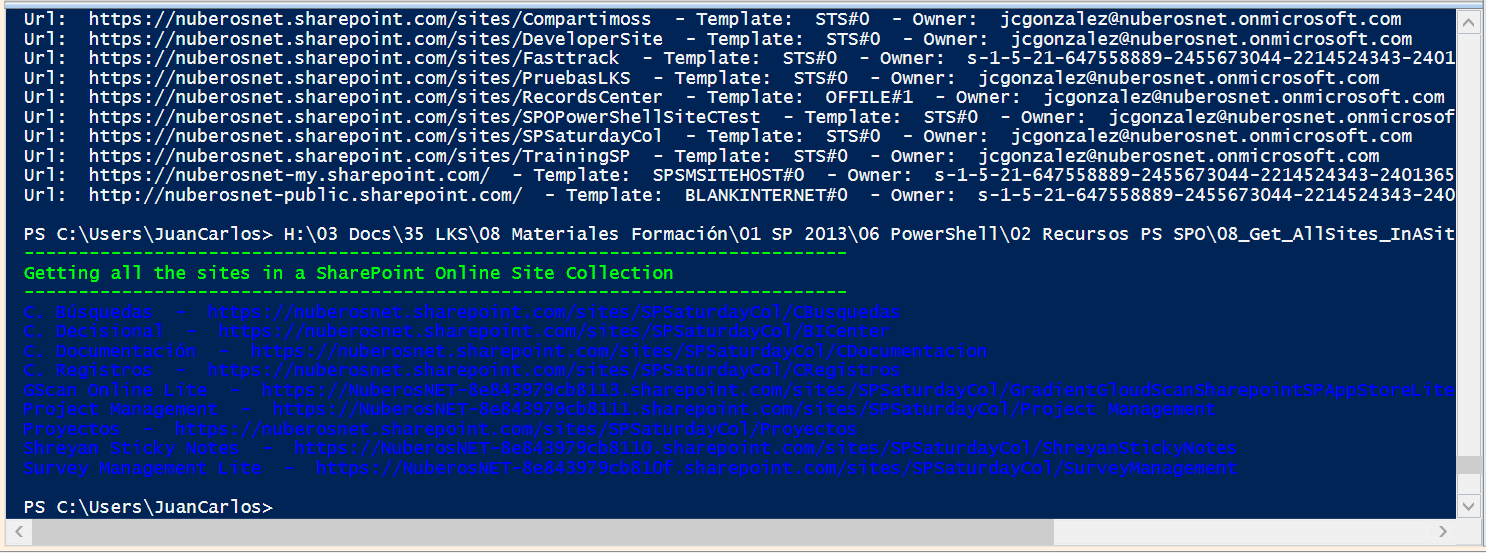
## Como obtener todos los sitios de una Colección de Sitios mediante CSOM

Para obtener todos los sitios bajo una Colección de Sitios mediante el CSOM y PowerShell:

* En el ISE añadimos el siguiente script PowerShell:

|  |
| --- |
| $host.Runspace.ThreadOptions = "ReuseThread"  #Definition of the function that gets all the site collections information in a SharePoint Online tenant  function Get-SPOSitesInSC  {  param ($sCSOMPath,$sSiteColUrl,$sUserName,$sPassword)  try  {  Write-Host "----------------------------------------------------------------------------" -foregroundcolor Green  Write-Host "Getting all the sites in a SharePoint Online Site Collection" -foregroundcolor Green  Write-Host "----------------------------------------------------------------------------" -foregroundcolor Green    #Adding the Client OM Assemblies  #Adding the Client OM Assemblies  $sCSOMRuntimePath=$sCSOMPath + "\Microsoft.SharePoint.Client.Runtime.dll"  $sCSOMPath=$sCSOMPath + "\Microsoft.SharePoint.Client.dll"  Add-Type -Path $sCSOMPath  Add-Type -Path $sCSOMRuntimePath  #SPO Client Object Model Context  $spoCtx = New-Object Microsoft.SharePoint.Client.ClientContext($sSiteColUrl)  $spoCredentials = New-Object Microsoft.SharePoint.Client.SharePointOnlineCredentials($sUsername, $sPassword)  $spoCtx.Credentials = $spoCredentials  #Root Web Site  $spoRootWebSite = $spoCtx.Web  #Collecction of Sites under the Root Web Site  $spoSites = $spoRootWebSite.Webs  #Loading operations  $spoCtx.Load($spoRootWebSite)  $spoCtx.Load($spoSites)  $spoCtx.ExecuteQuery()  #We need to iterate through the $spoSites Object in order to get individual sites information  foreach($spoSite in $spoSites){  $spoCtx.Load($spoSite)  $spoCtx.ExecuteQuery()  Write-Host $spoSite.Title " - " $spoSite.Url -ForegroundColor Red  }  $spoCtx.Dispose()  }  catch [System.Exception]  {  write-host -f red $\_.Exception.ToString()  }  }  #Required Parameters  $sSiteColUrl = "https://<O365Domain>.sharepoint.com/sites/SPSaturdayCol"  $sUserName = "<SPOUser>@<O365Domain>.onmicrosoft.com"  $sPassword = Read-Host -Prompt "Enter your password: " -AsSecureString  $sCSOMPath="<CSOM\_Assemblies\_Path>"  Get-SPOSitesInSC -sCSOMPath $sCSOMPath -sSiteColUrl $sSiteColUrl -sUserName $sUserName -sPassword $sPassword |

* El resultado que deberíamos obtener al ejecutar el script es el siguiente:



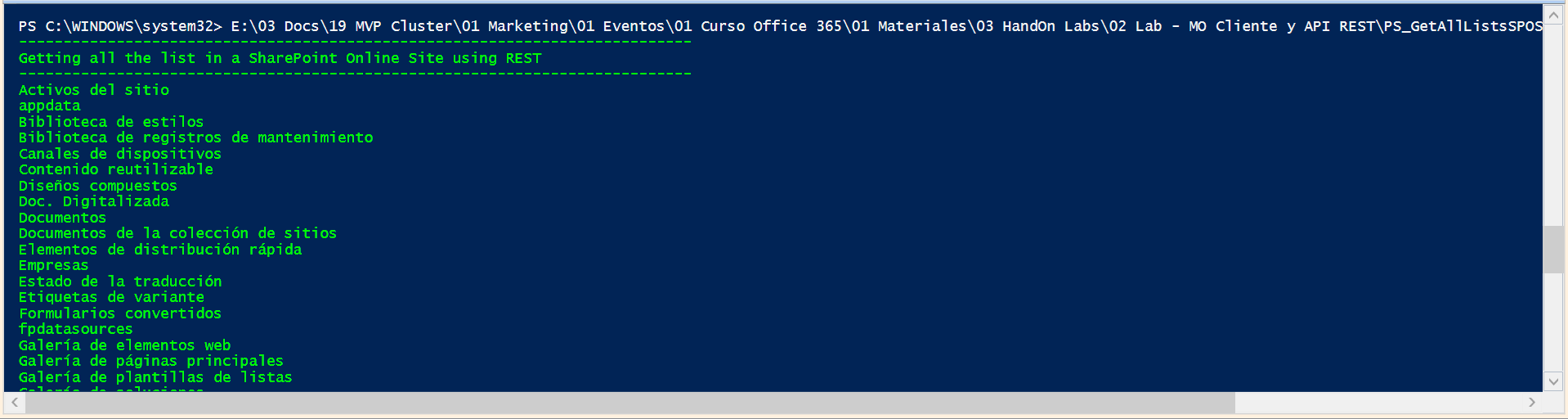
## Uso de la API REST de SPO desde PowerShell

De la misma forma que se puede hacer uso del CSOM para SPO desde PowerShell, se puede hacer uso de la API REST. El siguiente script demuestra cómo hacer uso de dicha API:

* En el ISE añadimos el siguiente script PowerShell que permite listar todas las listas de un sitio de SPO:

|  |
| --- |
| $host.Runspace.ThreadOptions = "ReuseThread"  #Definition of the function that gets all the lists in a SharePoint Online Site using REST  function Get-SPListsUsingRESTAPI  {  param ($sCSOMPath,$sRESTUrl,$sUserName,$sPassword, $WebRMethod)  try  {  Write-Host "----------------------------------------------------------------------------" -foregroundcolor Green  Write-Host "Getting all the list in a SharePoint Online Site using REST" -foregroundcolor Green  Write-Host "----------------------------------------------------------------------------" -foregroundcolor Green    #Adding the Client OM Assemblies  $sCSOMRuntimePath=$sCSOMPath + "\Microsoft.SharePoint.Client.Runtime.dll"  $sCSOMPath=$sCSOMPath + "\Microsoft.SharePoint.Client.dll"  Add-Type -Path $sCSOMPath  Add-Type -Path $sCSOMRuntimePath  $spCredentials = New-Object Microsoft.SharePoint.Client.SharePointOnlineCredentials($sUserName, $sPassword)  $spWebRequest = [System.Net.WebRequest]::Create($sRESTUrl)  $spWebRequest.Credentials = $spCredentials  $spWebRequest.Headers.Add("X-FORMS\_BASED\_AUTH\_ACCEPTED", "f")  $spWebRequest.Accept = "application/json;odata=verbose"  $spWebRequest.Method=$WebRMethod  $spWebResponse = $spWebRequest.GetResponse()  $spRequestStream = $spWebResponse.GetResponseStream()  $spReadStream = New-Object System.IO.StreamReader $spRequestStream  $spData=$spReadStream.ReadToEnd()  $spResults = $spData | ConvertFrom-Json  $spLists=$spResults.d.results  foreach($spList in $spLists)  {  Write-Host $spList.Title -ForegroundColor Green  }  }  catch [System.Exception]  {  write-host -f red $\_.Exception.ToString()  }  }  #Required Parameters  $sRESTUrl = "https://<SPO\_Site\_Url>/\_api/web/lists"  $sUserName = "<SPOUser>@<O365Domain>.onmicrosoft.com"  $sPassword = Read-Host -Prompt "Enter your password: " -AsSecureString  $WebRMethod=[Microsoft.PowerShell.Commands.WebRequestMethod]::Get  $sCSOMPath="<CSOM\_Assemblies\_Path>"  Get-SPListsUsingRESTAPI -sCSOMPath $sCSOMPath -sRESTUrl $sRESTUrl -sUserName $sUserName -sPassword $sPassword -WebRMethod $WebRMethod |

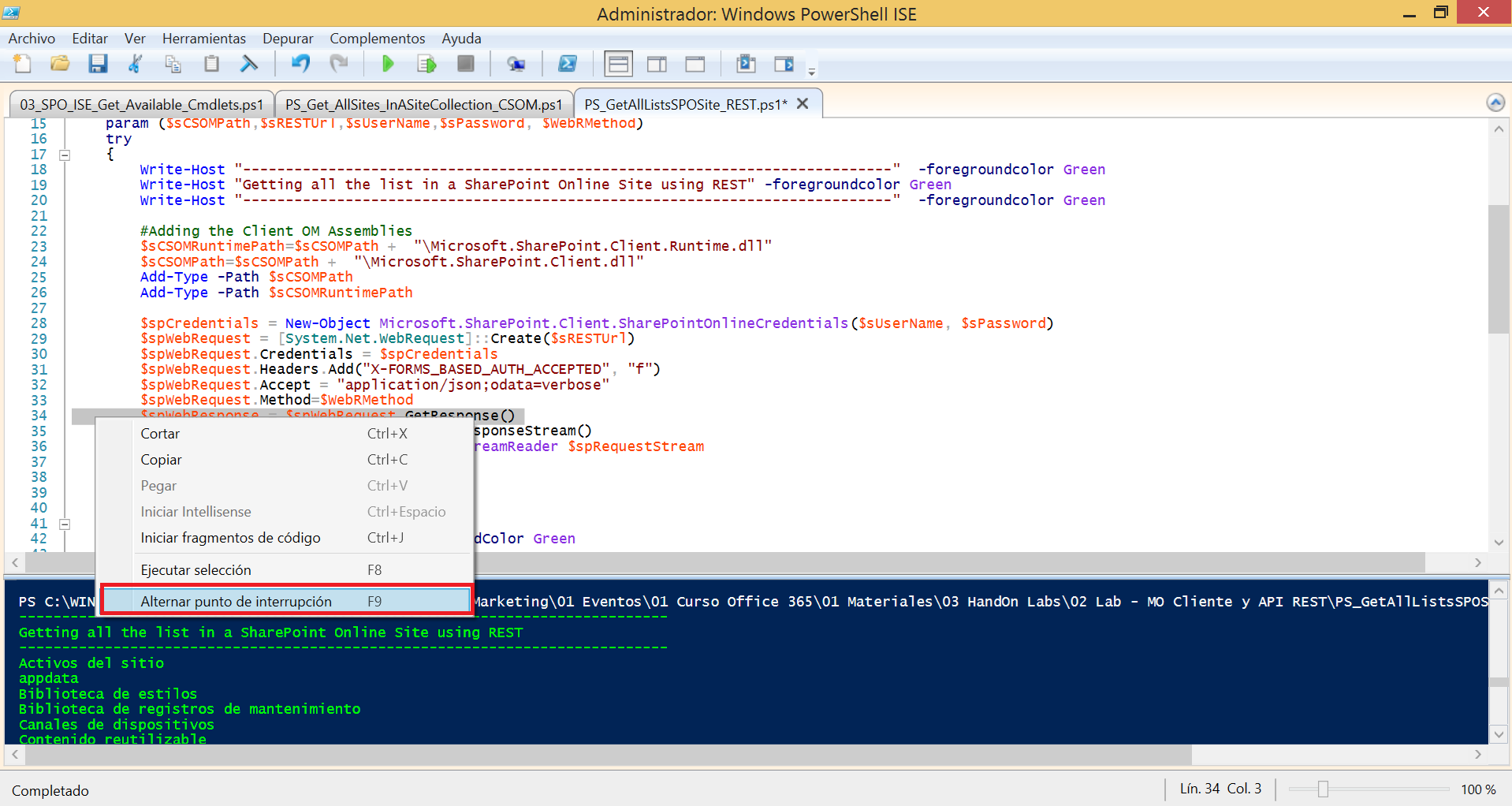
* La salida por pantalla que se obtiene es la siguiente:



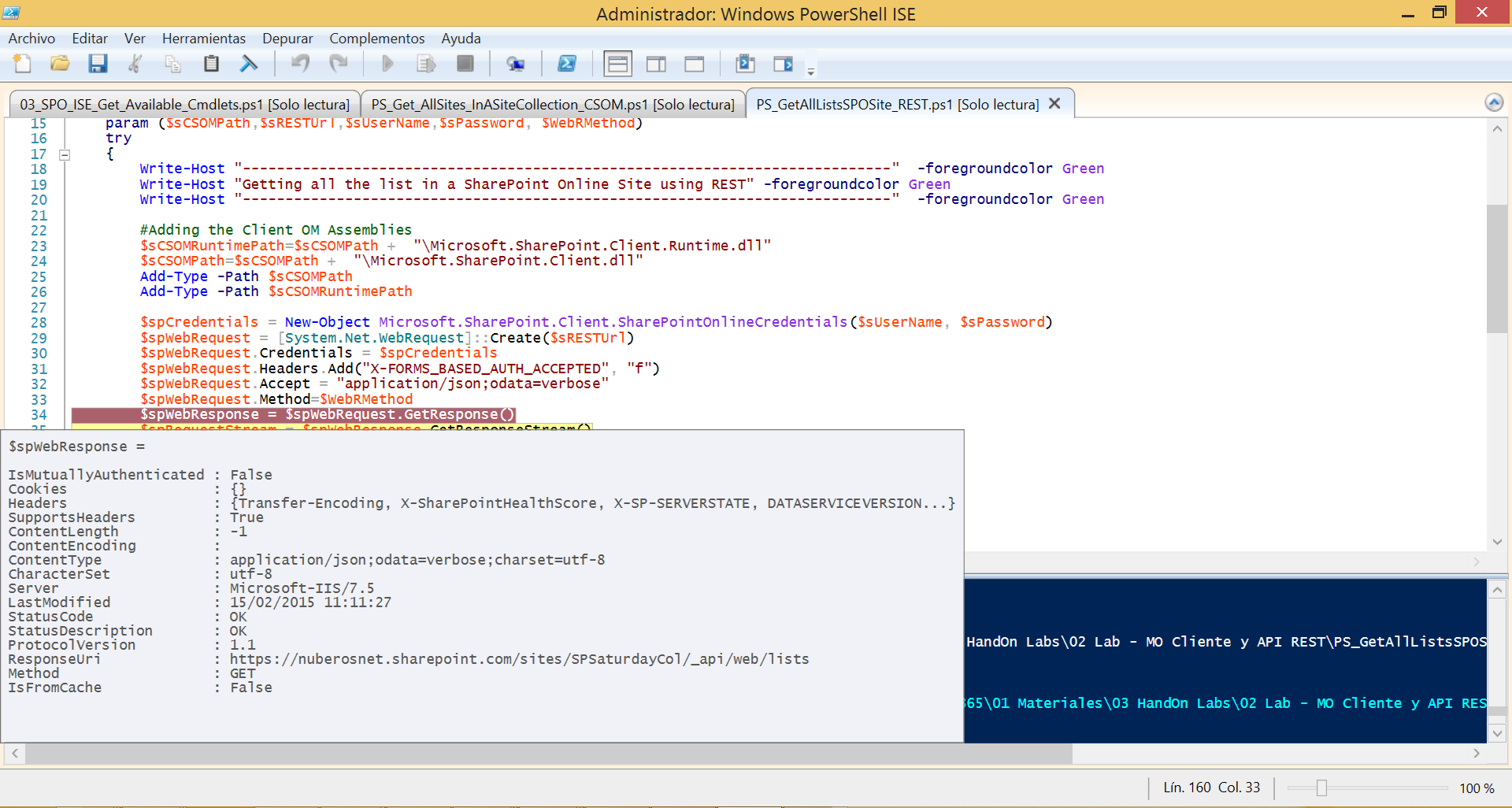
## Como depurar Scripts PowerShell con el ISE

Una de las características del ISE desde la perspectiva del programador es que permite depurar los Scripts añadiendo puntos de ruptura en los mismos:

* Para añadir un punto de ruptura, simplemente hacemos clic con el botón derecho del ratón en la línea del script en la que se quiere agregar y hacemos clic en “Alternar punto de interrupción”.



* A continuación ejecutamos el script de forma que veremos cómo su ejecución se para en la línea en la que hemos añadido el punto de interrupción y podremos inspeccionar pasando el ratón por encima de los objetos involucrados el contenido de los mismos:



# Encuesta de satisfaction

* <http://www.metricsthatmatter.com/student/evaluation.asp?k=16302&i=892359>