|  |
| --- |
|  |

17/3/2015

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



|  |  |
| --- | --- |
|  | Laboratorio Aplicaciones SharePoint Hosted |

**Indice**

[1 Introducción 2](#_Toc413750889)

[2 Requisitos 2](#_Toc413750890)

[3 Creación de la aplicación 2](#_Toc413750891)

[4 Creación de una Lista 5](#_Toc413750892)

[5 Accediendo a la Lista programáticamente 7](#_Toc413750893)

[6 Despliegue en el Catalogo de Aplicaciones 9](#_Toc413750894)

# Introducción

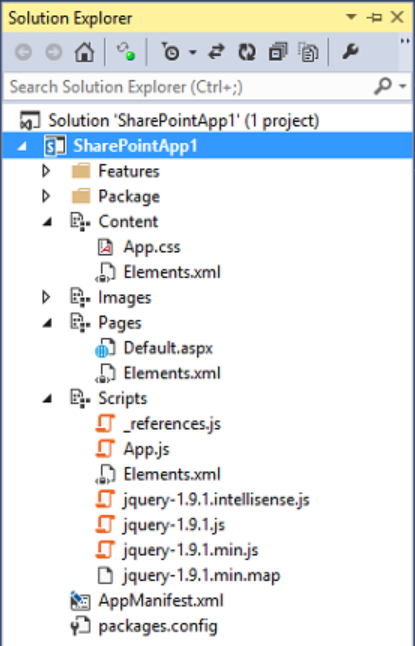
SharePoint 365 permite crear aplicaciones SharePoint Hosted. En este laboratorio se creará una aplicación de este tipo utilizando la plantilla por defecto de Visual Studio, que al momento de creación provisiona una Lista personalizada. Posteriormente le agregara código para poder trabajar con la Lista programáticamente.

# Requisitos

* Cuenta activada de Office 365
* SharePoint 365 configurado con una Colección de Sitios de Desarrollo y otra para el Catalogo de Aplicaciones
* Computador con Visual Studio 2013 (cualquier update). Puede ser en el computador local, en un computador virtualizado o un computador en Azure

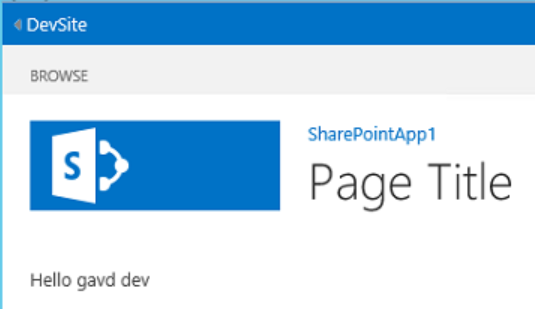
# Creación de la aplicación

1. Inicie Visual Studio 2013 con la cuenta de administrador del computador
2. Seleccione "New project"
3. En la lista de plantillas de Visual Studio seleccione "Visual C#" 🡪 "Office/SharePoint" 🡪 "Apps" 🡪 "App for SharePoint"
4. Acepte el nombre por defecto ("SharePointApp1")
5. En la ventana de "New app for SharePoint":
   1. Indique en la casilla de "What SharePoint site do you want to use for debugging your app" el URL base de la Colección de Sitios de Desarrollo en SharePoint (Office) 365, por ejemplo <http://[nombre].sharepoint.com/sites/devsite/>
   2. Seleccione "SharePoint-hosted" en la sección "How do you want to host your app for SharePoint"
   3. "Finish"
6. En la ventana de "Connect to SharePoint" indique la cuenta ("User ID") y contraseña de Office 365 y seleccione "Keep me signed" 🡪 "Sign in"
7. Revise la organización de la aplicación creada en el Solution Explorer de Visual Studio:

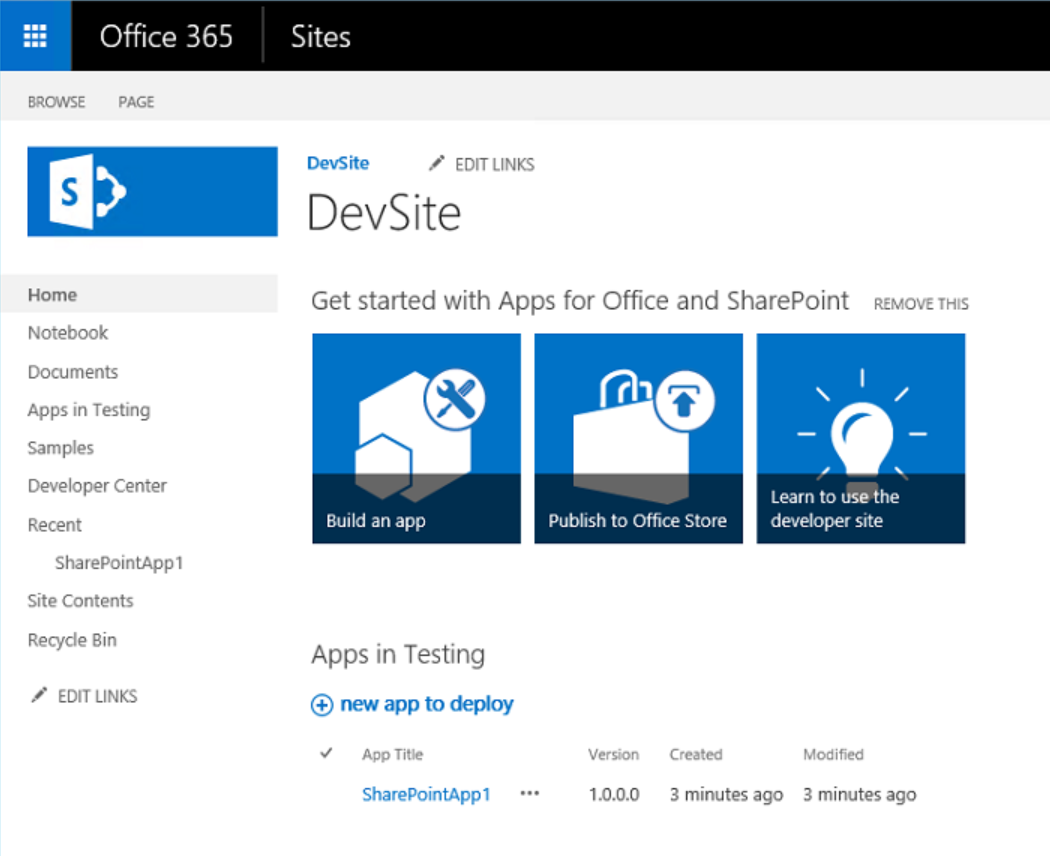


* 1. La carpeta "Content" contiene la hoja de estilo a utilizar en la aplicación
  2. Una aplicación consiste de por lo menos una página aspx que se encuentra en la carpeta "Pages"
  3. La carpeta "Scripts" contiene todos los archivos con scripts de JavaScript. El archivo "App.js" que contiene el código de ejemplo de la aplicación creada por Visual Studio
  4. El archivo "AppManifest.xml" contiene la configuración de la Aplicación

1. Examine el código de la página "Default.aspx". En las líneas 11 a 20 se registran los archivos de JavaScript a utilizar. Líneas 31 a 36 crean un parágrafo de HTML con un identificador ("message") que va a mostrar el código generado por el código de JavaScript
2. Examine el código de la página "App.js":
   1. Usando el Modelo de Objetos de Cliente (CSOM) se encuentra el contexto de la Aplicación (línea 3) y la cuenta del usuario logeado (línea 4)
   2. Líneas 7 a 9 indican que cuando la página ha terminado de cargar, se debe ejecutar la función "getUserName"
   3. La función "getUserName" envía la consulta a SharePoint y captura el resultado en la función delegada "onGetUserNameSuccess" (si el servidor no genera un error) o en la función delegada "onGetUserNameFail" (en caso contrario)
3. Haga ejecutar la Aplicación (F5, o el botón de "Start" o usando el menú "Debug 🡪 Star Debugging"). El proceso de despliegue se puede seguir por medio de la ventana de "Output" en Visual Studio
4. Visual Studio despliega la Aplicación en SharePoint 365 y abre un nuevo navegador mostrándola:



1. Abra el Sitio de Desarrollo de SharePoint 365. En la Biblioteca de "Apps in Testing" se encuentra desplegada la Aplicación que se acaba de crear



1. En Visual Studio (sin detener el debuggeo), agregue un Breakpoint in la línea 8 de App.js. Refresque la página de la Aplicación en Internet Explorer. Revise que la ejecución se detiene en el Breakpoint y se pueden examinar todos los objetos creados
2. Detenga el debugueo (cerrando Internet Explorer)

# Creación de una Lista

1. En el "Solution Explorer" de Visual Studio seleccione el Proyecto y usando el menú contextual cree un folder llamado "Listas"
2. En el folder de "Listas", usando el menú contextual, seleccione "Add" 🡪 "New Item"
3. Seleccione "List" y utilice el nombre por defecto ("List1") 🡪 "Add"
4. En la ventana de "SharePoint Customization Wizard" utilice la configuración por defecto ("List1" and "Default (Custom List") 🡪 "Finish"
5. Visual Studio abre el archivo "Schema.xml" para configurar la Lista:
   1. Haga clic sobre "Type a new or existing column name" y escriba "Nombre"
   2. Repita el proceso para crear una nueva columna llamada "Apellido" (utilice el "Type" por defecto)
   3. Esta pantalla permite la creación no solo de columnas, sino también de Tipos de Contenido y Vistas. Mantenga la configuración por defecto
6. Inicie la Aplicación de nuevo en modo de debugeo (F5)
7. Cuando la Aplicación termine de desplegar y se abra Internet Explorer con su primera página, vaya al URL "http://[nombre] -[ID].sharepoint.com/sites/devsite/SharePointApp1/Lists/List1" para ver la Lista creada
8. Cierre el modo de debugeo y regrese al código de Visual Studio
9. Abra el archivo "Default.aspx" y agregue el siguiente código en la línea 37 (después del parágrafo de "message") para crear un nuevo botón en la página:

<div>

<input id="Button1" type="button" value="Ir A Lista" onclick="IrALista('SPAppWebUrl');" />

</div>

1. Abra el archivo "App.js" y agregue el siguiente código al final del archivo:

function getParameterByName(name) {

name = name.replace(/[\[]/, "\\[").replace(/[\]]/, "\\]");

var regex = new RegExp("[\\?&]" + name + "=([^&#]\*)"),

results = regex.exec(location.search);

return results === null ? "" : decodeURIComponent(results[1].replace(/\+/g, " "));

}

function IrALista(HostUrl) {

var siteUrl = getParameterByName(HostUrl);

var listUrl = siteUrl + "/Lists/List1"; //==> Coloque un Breakpoint aqui para debuggear

window.location = listUrl;

}

1. La primera función permite extraer el valor de un parámetro en el QueryString. La segunda función concatena el URL de la Lista y redirige el usuario a la página de la Lista
2. Debugee la Aplicación desde Visual Studio. Al apretar el botón se debe redirigir hacia la página de la Lista
3. Detenga el debugeo y de regreso en Visual Studio abra el archivo (doble clic) "Manifest.xml" desde el Explorador de Soluciones. Visual Studio abre un ayudante de configuración en donde, en la primera pestaña ("General") se encuentra el parámetro "Query string" y un macro como valor ("{StandardTokens}"). Este macro genera cinco valores en el QueryString: SPHostUrl, SPLanguage, SPClienteTag, SPProductNumber y SPAppWebUrl. Si es necesario se pueden agregar valore en el QueryString, o modificar los valores por defecto
4. Es posible agregar valores en la Lista automáticamente en el momento de creación. Expanda el folder de "Listas" en el Solution Explorer de Visual Studio, expanda "List1", expanda "ListInstance" y abra el archivo "Elements.xml". Reemplace todo el código en el archivo con el siguiente código que le agrega una sección "Data" a la etiqueta "ListInstance":

<?xml version="1.0" encoding="utf-8"?>

<Elements xmlns="http://schemas.microsoft.com/sharepoint/">

<ListInstance Title="List1"

OnQuickLaunch="TRUE"

TemplateType="100"

Url="Lists/List1"

Description="My List Instance">

<Data>

<Rows>

<Row>

<Field Name="Title">Una</Field>

<Field Name="Nombre">Sophia</Field>

<Field Name="Apellido">Loren</Field>

</Row>

<Row>

<Field Name="Title">Otra</Field>

<Field Name="Nombre">Gina</Field>

<Field Name="Apellido">Lollobrigida</Field>

</Row>

</Rows>

</Data>

</ListInstance>

</Elements>

1. Debugee la Aplicación desde Visual Studio. Use el botón para ir a la página de la Lista. La Lista contiene dos valores agregados por defecto

# Accediendo a la Lista programáticamente

1. Abra la página "Default.aspx" en Visual Studio y agréguele el siguiente fragmento entre el código del parágrafo del botón para ir a la Lista y "</asp:Content>" (línea 41). El código agrega un espacio para mostrar algún texto y un nuevo botón para mostrar las Listas en el sitio de la Aplicación:

<div>

<input type='button' onclick='TodasLasListas()' value='Enumerar Listas' /><br />

<textarea id="TextArea1" cols="40" rows="20"></textarea>

</div>

1. Abra el archivo "App.js" y agregue el siguiente código al final de la página:

function TodasLasListas() {

var myContext = new SP.ClientContext.get\_current();

var myWeb = myContext.get\_web();

this.myLists = myWeb.get\_lists();

myContext.load(myLists);

myContext.executeQueryAsync(

Function.createDelegate(this, this.onQuerySucceeded),

Function.createDelegate(this, this.onQueryFailed));

}

function onQuerySucceeded(sender, args) {

document.getElementById("TextArea1").value = "";

var myValues = '';

var myEnumerator = this.myLists.getEnumerator(); //==> Coloque un Breakpoint aqui para debuggear

while (myEnumerator.moveNext()) {

var oneList = myEnumerator.get\_current();

myValues += '\nTitulo: ' + oneList.get\_title();

}

document.getElementById("TextArea1").value = myValues.toString();

}

function onQueryFailed(sender, args) {

alert('Communication Failed '

+ args.get\_message() + '\n'

+ args.get\_stackTrace());

}

1. El código agregado contiene varias funciones de JavaScript que utilizan el Modelo de Objetos de Cliente de SharePoint:
   1. La función "TodasLasListas" crea un objeto basado en el contexto y objetos para contener el sitio Web y las Listas en él. Finalmente la función envía la consulta al servidor y recibe la respuesta en dos funciones delegadas
   2. La función delegada "onQuerySucceeded" comienza a funcionar si el servidor envía una respuesta válida. La función simplemente enumera el título de las Listas en la página Web
   3. La función delegada "onQueryFailed" funciona si el servidor envía un mensaje de error, y lo muestra en una alerta
2. Ejecute la Aplicación y cuando Internet Explorer termine de cargar la página, utilice el botón de "Enumerar Listas" para ver todas las Listas que contiene el sitio Web (debe incluir la Lista creada programáticamente en el punto anterior)
3. Agregue el siguiente código al final de la página "Default.aspx" (línea 46) para crear otro botón:

<div>

<input type='button' onclick='AgregarElemento()' value='Agregar Elemento' />

</div>

1. En el archivo "App.js" agregue al final el siguiente código:

function AgregarElemento() {

var myContext = new SP.ClientContext.get\_current();

var myWeb = myContext.get\_web();

this.myList = myWeb.get\_lists().getByTitle('List1');

var myItemCreationInfo = new SP.ListItemCreationInformation();

this.newItem = myList.addItem(myItemCreationInfo);

newItem.set\_item('Title', 'Una mas');

newItem.set\_item('Nombre', 'Brigitte');

newItem.set\_item('Apellido', 'Bardot');

newItem.update();

myContext.executeQueryAsync();

}

1. Esta función utiliza el contexto y dos objetos conteniendo el sitio Web y la Lista personalizada creada anteriormente. Luego, usando el método "ListItemCreationInformation" agrega un elemento nuevo en la Lista y modifica los campos del elemento creado. Finalmente envía la consulta al servidor. Note que no es obligatorio utilizar las funciones delegadas, aunque eso significa que el usuario no va a recibir ningún mensaje si la consulta ha sido ejecuta con éxito o no
2. Ejecute la Aplicación. Utilice el botón de "Agregar Elemento". Vaya a la página de la Lista utilizando el botón de "Ir a Lista". El nuevo elemento debe ser visible

# Despliegue en el Catalogo de Aplicaciones

1. Desde Visual Studio desinstale la Aplicación totalmente: usando el menú contextual en el Solución Explorer de Visual Studio, seleccione la Solución y utilice el botón de "Retract"
2. Cuando la Aplicación termine de ser removida totalmente de la Colección de Sitios de Desarrollo en SharePoint 365 (lo puede ver en la ventana de "Output" de Visual Studio), utilice la opción de "Publish" desde el menú contextual del Solution Explorer de Visual Studio
3. En la página de "Publish" que abre, utilice el botón de "Package the app". Una nueva instancia de Windows Explorer abre con el paquete de despliegue (extensión ".app")
4. Vaya a SharePoint 365 y abra la página del sitio del Catalogo
5. Abra la Biblioteca de "Apps for SharePoint"
6. Utilice el botón de "Upload" y suba el archivo ".app" generado por Visual Studio
7. Cuando la Aplicación termine de subir en la Biblioteca, abra una página cualquiera en un sitio de trabajo de la implementación de SharePoint 365
8. Utilice el vínculo de "Site Contents" (menú de acceso rápido al lado Izquierdo de la página)
9. Utilice el botón de "Add an app"
10. Utilice el vínculo de "From Your Organization" en el menú de acceso rápido
11. Haga clic sobre la Aplicación creada anteriormente
12. Cuando aparezca la ventana de "Do you trust SharePointApp1?" utilice el botón de "Trust it"
13. SharePoint regresa a la página de "Site Contents" y después de algunos segundos aparece la Aplicación instalada y lista para funcionar
14. Haga clic sobre "SharePointApp1"
15. Para desinstalar la Aplicación, desde la ventana de "Site Contents" utilice el botón de elipse ("...") al lado derecho de la Aplicación "SharePointApp1" y el vínculo de "Remove"
16. Utilice el botón de "OK" en la ventana de confirmación
17. Para remover la Aplicación del Catálogo, desde el sitio de Catálogo, Biblioteca "Apps for SharePoint" utilice el botón de elipse de la Aplicación, de nuevo el botón de elipse en la segunda pantalla y luego "Delete" y acepte los cambios