無料電子ブック

学習

sharepoint

Free unaffiliated eBook created from
Stack Overflow contributors.

#sharepoint

1
1: 2
2
2
Examples 2
SharePoint 2016 2
2
3
3
3
4
SharePoint FrameworkWeb 5
SharePoint ULS 5
5
5
SPMonitoredScope 5
2: JavaScriptJSOM 7
7
Examples 7
7
8
8
8
9
10
ID 10
CAML 11
11
CAML 11
3: JavaScript 13

13

13

14

Examples 14

14

14

14

4: REST 16

16

RESTURL 16

REST 16

XMLHttpRequest 16

jQuery AJAX 16

Examples 17

17

18

18

18

18

20

21

SharePointID 22

SharePoint 2010 RESTCRUD 23

5: SharePoint 2013 26

26

Examples 26

CSR/ 26

CSRSharePoint 27

CSR/ 27

CSR 32

6: SharePoint 33

33

33

Examples 33

SharePoint 2013SharePoint 2013JSOM 33

7: 35

Examples 35

35

36

Visual Studio 37

41

44

8: CSOM 47

47

Examples 47

47

Web 48

Web 48

Web 48

Web 48

Web 49

49

49

Web 50

50

51

51

51

Include 51

52

Web 52

53

53

54

54

SharePoint 54

55

55

SharePoint 55

56

Web 56

SharePoint 57

57

57

58

58

59

Web 59

Web Web 59

WebWeb 60

WebWeb 61

61

9: 63

Examples 63

SharePoint 2016 63

SharePoint 2013 63

10: 65

65

65

65

Examples 65

Hello World 65

SharePoint 66

66

66

url.....67
.....67
.....68

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: sharepoint

It is an unofficial and free sharepoint ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official sharepoint.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: シェアポイントの

SharePointは、Microsoft SharePointファミリの1つのをできます。

- **SharePoint Foundation** これはすべてのSharePointサイトのとなるテクノロジであり、SharePoint 2016ではできなくなりました
- **SharePoint Server** これは、SharePointのオンプレミスです。 1つのSharePointサーバーをできます。 BI、エンタープライズコンテンツなど、SharePoint Foundationのをします
- **SharePoint Online** クラウドベースのSharePoint。は、サーバーのインフラストラクチャまたはスケーラビリティをにするはありません。

Office 365は、SharePoint Onlineサービスをむのマイクロソフトですが、すべてのがすべてのSharePointをサポートするわけではありません。

のリンクは、なSharePointのバージョンでなをします。

- SharePoint 2013オンプレミスとSharePoint 2016オンプレミス SharePointのオンプレミスでのの
- Office 365のSharePoint SharePointでのの
- SharePoint OnlineOffice 365なしのSharePoint SharePointのスタンドアロンでのの
- SharePoint 2013とSharePoint Onlineののを http : //www.buckleyplanet.com/2014/06/sharepoint-online-vs-onprem-feature-comparison.html

バージョン

バージョン		
2003	SharePoint Portal Server	200279
2003	SharePoint Portal Server 2003	20031123
2007	SharePoint Server 2007	20070127
2010	Microsoft SharePoint Server 2010	2010-07-15
2013	Microsoft SharePoint Server 2013	2013-01-09
2016	Microsoft SharePoint Server 2016	2016-05-01

Examples

シングルサーバーファームの**SharePoint 2016**のインストール

き

SharePoint 2016は、SharePointファミリのバ―ジョン16リリ―スです。 201654にリリ―スされました。このでは、サ―バ―ファ―ムをしたSharePoint 2016のインスト―ルについてします。このでは、のサ―バ―をとせずにSharePointファ―ムをするためのについてします。サ―バ―ファ―ムのとなるシナリオは、、シナリオとになシナリオにられています。

SharePointをインスト―ルするに、なをするがあります。 SharePointは、ドキュメント、メタデ―タ、ログ、カスタムアプリケ―ション、カスタマイズなどをします。ベ―スラインのをえるなディスクスペ―スとRAMがあることをしてください。

- 64ビットプロセッサの4つのコア
- 1224 GBのRAMテストまたはプロダクトに
- システム80GBハ―ドドライブ
- 2のドライブとして100GBのハ―ドドライブ
- 64ビットWindows Server 2012 R2またはテクニカルプレビュ―「しきい」をつサ―バ―
- SQL Server 2014またはSQL Server 2016
- .NET Framework 4.5.2または.NET Framework 4.6
- ドメインにしたコンピュ―タアカウントとされたファ―ムサ―ビスアカウント

のすべてのは、でインスト―ルすることも、SharePointインスト―ルにまれているSharePointインスト―ラをしてすることもできます。

インスト―ル

- インスト―ラをします。するにサ―バ―のをすることがあります
- SharePointインスト―ルからSetup.exeをします。
- ライセンスキ―をしてください
- ライセンスにする
- [サ―バ―の]タブで[]をします。
- セットアップがにするはずです
- ペ―ジで、[のウィザ―ド]のにあるチェックボックスをオンのままにして、[じる]をクリックします。

のをしているは、SharePoint 2016ウィザ―ドがにきます。ボックスがされないや、でをするは、スタ―ト - > SharePoint 2016 - > SharePoint 2016ウィザ―ドのにしてウィザ―ドをきます。

- ウェルカムペ―ジでへをクリックします。
- にされるいくつかのサ―ビスをすモ―ダルダイアログがされます。もインスト―ルされていないので、「はい」をクリックします

- ファームのデータベースサーバーをする
  - SQL Serverをしているマシンのをします。この、それはローカルマシンです
  - データベースのをするか、デフォルトのままにするSharePoint_Config
  - データベースにアクセスするドメインサービスユーザーのユーザーDOMAIN \ userのをします*ドメインユーザーのパスワードをします
  - したらへをクリックします
- ファームパスワードをします。のサーバーをしいファームにさせるときにされます
- サーバーファームのをする
- Central Admin Web AppSharePointがファームによってされるをし、ポートをし、フェデレーションのタイプをしますNTLMまたはNegotateKerberose。
- ページのをし、にじてします
- ができたら、をします。にはかかることがあります
- すると、ウィザードがき、サイトをくことができます
- したは、COMMONPROGRAMFILES\ Microsoft Shared \ Web Server Extensions \ 16 \ LOG フォルダのログをべることができます

ファーム

のWebアプリケーション、データベース、およびがされると、ユーザまたはにするようにファームをするがいます。 Central Adminサイトのをブックマークするか、ウィザードとじにあるショートカットからアクセスできます。

- でをするは、「クイック」->「ウィザード」->「ファームウィザード」をクリックします。
- インストールからウィザードをするは、ウィザードのをクリックします。
- はいまたはいいえをクリックしてプログラムにするかどうかをします
- ファームページで、ファームでバックグラウンドサービスをするドメインアカウントをします
  - このアカウントはデータベースアカウントとじでもかまいませんが、ロールとのではなるもあります
  - DOMAIN \ userとしてアカウントをしてください
- [サービス]ページでファームでなサービスをする
- ファームでのサイトコレクションをしますこのはスキップしてでできます
  - サイトコレクションのタイトル、、Webアドレス、のサイトはサーバールートにあります、およびテンプレート
  - ほとんどのものはすることができますタイトル、にすることができますが、Web URL のようなのものは、テンプレートをにロールバックすることもできませんが、SharePointではのカスタマイズがで、テンプレートをしてサイトのスタイルとレイアウトをできます
- がしたら、[]をクリックします

ファームとのサイトコレクションができるようにされました。

SharePoint FrameworkでWebパーツをする

dev.office.com/sharepointは、SharePoint Frameworkをするのになです。

SharePoint Frameworkは、にOffice 365でSharePoint Onlineをターゲットとした、SharePointにするなクライアントサイドのアプローチです。SharePoint FrameworkでされたWebパーツはしいタイプのWebパーツであり、のSharePointページとしいSharePointページ

のSharePointクライアントサイドWebパーツをするHello Worldパート1でホストされているこのプロセスのらしいhello worldのがあります 。 dev.office.comのすべてのは、githubをしてコミュニティのにできます。

SharePoint FrameworkのHello Worldのなはのとおりです。

1. Yeoman SharePoint Generatorをしてプロジェクトのスケルトンをします 。

   yo @ microsoft / SharePoint

2. したエディタでされたコードをします。 Visual Studioコードのサポートは、プラットフォームでです。

3. gulpとローカルSharePoint WorkbenchをしてWebパーツをプレビューする

   ガルプスサービス

4. SharePoint Onlineでのプレビュー

   のURLにアクセスしてください ' https//your-sharepoint-site/_layouts/workbench.aspx '

SharePoint ULSのログとログ

SharePoint Unified Logging ServiceULSは、opsとのにサポートとデバッグをします。ログをみるをすることは、をするためのなです。

ツーリング

マイクロソフトはULSビューアをして、ファームがされているときにきまれているいログとログのみりをします。また、フィルタリングしてログにをして、のりみにてることもできます。

をするには、のIDだけをべるとです。IDは、システムのまたはアクションジョブバーなどにけられます。レンダリングのWebページにがある、ULSログにをしてのIDにすると、のログからすべてのノイズがされ、をするのにちます。

マイコードにSPMonitoredScopeをする

ロギングのとパフォーマンスモニタリングをするの1つは、コードにSPMonitoredScopeをすることです。

```
using (new SPMonitoredScope("Feature Monitor"))
{
    // My code here
}
```

このコードでは、リクエストのめとわり、およびのパフォーマンスデータがされます。
ISPScopedPerformanceMonitorをするのカスタムモニタをすることで、のコードのトレースレベルまたはをできます。

オンラインでシェアポイントのをむ https://riptutorial.com/ja/sharepoint/topic/950/シェアポイントの

2: JavaScriptクライアントオブジェクトモデルJSOMの

バックグラウンド

JavaScriptオブジェクトモデルはSharePoint 2010でされました。はサーバーのコードまたはのWebサービスをじてアクセスだったくのオブジェクトをクライアントでしています。

SharePointページへの**JavaScript**のめみ

SharePoint 2013では、JavaScriptをスクリプトエディタのWebパーツにできます。

SharePoint 2010では、コンテンツエディタWebパーツの「コンテンツリンク」プロパティをして、めみスクリプトをむHTMLファイルにリンクすることができます。

オブジェクト

SPにまれるすべてのオブジェクトのコンストラクタ、メソッド、およびプロパティは、 ここではSharePoint 2013クライアントオブジェクトモデルのリファレンスにされています。

ここでは、SharePoint 2010のJavaScriptクライアントオブジェクトモデルリファレンスをできます 。

JSOMのプログラミングパターン

JavaScriptクライアントオブジェクトモデルをする、コードはにのパターンをとります。

1. ClientContextオブジェクトをします。
2. ClientContextオブジェクトをして、SharePointオブジェクトモデルのエンティティリスト、フォルダ、ビューなどをすオブジェクトをします。
3. オブジェクトにしてされるをキューイングします。これらのはまだサーバーにはされません。
4. loadをして、サーバーからするをClientContextにします。
5. ClientContextオブジェクトのexecuteQueryAsyncをびして、 ClientContextれたをサーバーにし、またはにされる2つのコールバックをします。
6. コールバックでは、サーバーからされたをします。

JSOMのクライアントのには、SharePointのWebサービス、 RESTエンドポイント 、 .NETクライアントオブジェクトモデルなどがあります 。

Examples

ライブラリをしてライブラリコンテンツタイプをする

```
function getContentTypes(site_url,name_of_the_library){
    var ctx = new SP.ClientContext(site_url);
    var web = ctx.get_web();
    list = web.get_lists().getByTitle(name_of_the_library);

    // You can include any property of the SP.ContentType object (sp.js), for this example we
are just getting the name
    ctx.load(list,'ContentTypes.Include(Name)');
    ctx.executeQueryAsync(onQuerySucceeded, onQueryFailed);
}

function onQuerySucceeded(sender, args) {
    // var list is the one that we used in function "getContentTypes"
    var contentTypesEnumerator = (list.get_contentTypes()).getEnumerator();

    while (contentTypesEnumerator.moveNext()) {
        var contentType = contentTypesEnumerator.get_current();
        alert(contentType.get_name());
    }
}

function onQueryFailed(sender, args) {
    alert('Request failed. ' + args.get_message() + '\n' + args.get_stackTrace());
}
```

リストのをする

```
SP.SOD.executeOrDelayUntilScriptLoaded( function(){ deleteItem(1); }, "sp.js");

function deleteItem(id){
    var clientContext = new SP.ClientContext();
    var list = clientContext.get_web().get_lists().getByTitle("List Title");
    var item = list.getItemById(id);
    item.deleteObject();
    clientContext.executeQueryAsync(function(){
        alert("Item #"+id+" deleted successfully!");
    },function(sender,args){alert(args.get_message());});
}
```

アイテムまたはフォルダの

リストの

```
SP.SOD.executeOrDelayUntilScriptLoaded(createItem,"sp.js");

function createItem(){
    var clientContext = new SP.ClientContext();
    var list = clientContext.get_web().get_lists().getByTitle("List Title");
    var newItem = list.addItem();
    newItem.set_item("Title","Example Title");
    newItem.update();
    clientContext.load(newItem); // only needed to retrieve info from newly created item
    clientContext.executeQueryAsync(function(){
        var itemId = newItem.get_item("ID");
        alert("Item #"+itemId+" Created Successfully!");
```

```
    },function(sender,args){
        alert(args.get_message());
    });
}
```

のは、のをすることによってリストがされることをしています。

1. リストオブジェクトの`addItem`メソッドをびしてアイテムオブジェクトをする
2. のリストアイテムオブジェクトにして`set_item`メソッドをびして、フィールドをにじてします
3. リストアイテムオブジェクトの`update`メソッドをびして、がコミットされることをします。
4. キューにされたをするには、クライアントコンテキストオブジェクトの`executeQueryAsync`メソッドをびします。

アイテムをするために、しいアイテムオブジェクトをクライアントコンテキストの`load`メソッドにすはありません。このは、サーバーからアイテムのフィールドをするにのみです。

フォルダの

フォルダをすることは、リストにをすることにています。いは、に`ListItemCreationInformation`オブジェクトをし、その`underlyingObjectType`プロパティを`SP.FileSystemObjectType.folder`し、その`leafName`プロパティをしいフォルダのするにするがあることです。

オブジェクトは、ライブラリの`addItem`メソッドのパラメータとしてされ、フォルダがされます。

```
// ...
var itemCreateInfo = new SP.ListItemCreationInformation();
itemCreateInfo.set_underlyingObjectType(SP.FileSystemObjectType.folder);
itemCreateInfo.set_leafName(folderName);
var newItem = list.addItem(itemCreateInfo);
// ...
```

をコミットするには、ライブラリがアクセスされた`ClientContext`オブジェクトの`executeQueryAsync`メソッドをびします。

のなは、のタイムスタンプにづいたのフォルダをし、そのフォルダをモーダルダイアログできます。

```
SP.SOD.executeOrDelayUntilScriptLoaded(createFolder,"sp.js");

function createFolder(){
    var now = new Date();
    var timeStamp = now.getYear() + "-" + (now.getMonth()+1) + "-" + now.getDate()
        + "T" + now.getHours()+"_"+now.getMinutes()+"
"+now.getSeconds()+"_"+now.getMilliseconds();
    var clientContext = new SP.ClientContext();
    var list = clientContext.get_web().get_lists().getByTitle("Library Title");
    var itemCreateInfo = new SP.ListItemCreationInformation();
    itemCreateInfo.set_underlyingObjectType(SP.FileSystemObjectType.folder);
```

```
    itemCreateInfo.set_leafName(timeStamp);
    var newItem = list.addItem(itemCreateInfo);
    newItem.update();
    clientContext.load(newItem);
    var rootFolder = list.get_rootFolder(); // Note: use a list's root folder to determine its
server relative URL
    clientContext.load(rootFolder);
    clientContext.executeQueryAsync(function(){
        var itemId = newItem.get_item("ID");
        var name = newItem.get_item("FileLeafRef");
        SP.UI.ModalDialog.showModalDialog(
            {
                title: "Folder \""+name+"\" (#"+itemId+") Created Successfully!",
                url: rootFolder.get_serverRelativeUrl() + "/" + name
            }
        );
    },function(sender,args){alert(args.get_message());});
}
```

のユーザーをする

```
SP.SOD.executeOrDelayUntilScriptLoaded(showUserInfo,"sp.js");

function showUserInfo(){
    var clientContext = new SP.ClientContext();
    var user = clientContext.get_web().get_currentUser();
    clientContext.load(user);
    clientContext.executeQueryAsync(function(){
            var details = "ID: "+user.get_id()+"\n"+
                "Title: "+user.get_title()+"\n"+
                "Login: "+user.get_loginName()+"\n"+
                "Email: "+user.get_email();
            alert(details);
        },function(sender,args){alert(args.get_message());})
}
```

IDでリストアイテムをする

```
SP.SOD.executeOrDelayUntilScriptLoaded(myFunction,"sp.js");

function myFunction(){
    var clientContext = new SP.ClientContext();
    var list = clientContext.get_web().get_lists().getByTitle("List Title");
    var item = list.getItemById(1); // get item with ID == 1
    clientContext.load(item);
    clientContext.executeQueryAsync(
        function(){ // onSuccess
            var title = item.get_item("Title");
            alert(title);
        },
        function(sender,args){ // onError
            alert(args.get_message());
        }
    );
}
```

な

SP.CamlQueryオブジェクトの`set_viewXml`メソッドをして、アイテムをするCAMLクエリをします。

```
SP.SOD.executeOrDelayUntilScriptLoaded(showListItems,"core.js");

function showListItems(){
    var clientContext = new SP.ClientContext();
    var list = clientContext.get_web().get_lists().getByTitle("List Title");
    var camlQuery = new SP.CamlQuery();
    camlQuery.set_viewXml(
        "<View><Query>" +
            "<Where>" +
                "<Eq><FieldRef Name=\"Title\"/><Value Type=\"Text\">Value</Value></Eq>" +
            "</Where>" +
            "<OrderBy><FieldRef Name=\"Modified\" Ascending=\"FALSE\"/></OrderBy>" +
        "</Query>"+
        //"<RowLimit>5000</RowLimit>" +
        "</View>");
    var items = list.getItems(camlQuery);
    clientContext.load(items);
    clientContext.executeQueryAsync(function(){
        var itemArray = [];
        var itemEnumerator = items.getEnumerator();
        while(itemEnumerator.moveNext()){
            var item = itemEnumerator.get_current();
            var id = item.get_item("ID");
            var title = item.get_item("Title");
            itemArray.push(id + ": " + title);
        }
        alert("ID: Title\n"+itemArray.join("\n"));
    },function(sender,args){alert(args.get_message());});
}
```

CAMLせののページング

CAMLせの`RowLimit`をして、せでのサブセットのみをできます。

リストアイテムコレクションの`get_listItemCollectionPosition`メソッドをしてのをし、そのをSP.CamlQueryオブジェクトの`set_listItemCollectionPosition`メソッドのパラメータとしてして、ののバッチをします。

```
SP.SOD.executeOrDelayUntilScriptLoaded(showListItems,"sp.js");

function showListItems(){
    var itemArray = [];
    var clientContext = new SP.ClientContext();
    var list = clientContext.get_web().get_lists().getByTitle("List Title");
```

```
    var viewXml =
        "<View><Query>" +
            "<OrderBy><FieldRef Name=\"Modified\" Ascending=\"FALSE\"/></OrderBy>" +
        "</Query>"+
          "<RowLimit>1</RowLimit>" +
        "</View>";
    var camlQuery = new SP.CamlQuery();
    camlQuery.set_viewXml(viewXml);
    var items = list.getItems(camlQuery);
    clientContext.load(items);
    clientContext.executeQueryAsync(loadResults,showError);

    function loadResults(){
        var resultsFound = false;
        var itemEnumerator = items.getEnumerator();
        while(itemEnumerator.moveNext()){
            var item = itemEnumerator.get_current();
            var id = item.get_item("ID");
            var title = item.get_item("Title");
            itemArray.push(id + ": " + title);
        }
        var pos = items.get_listItemCollectionPosition();// <- get position
        if(pos !== null){ // <-- position is null when no more results are returned
            if(confirm("Results so far: \nID: Title\n"+itemArray.join("\n"))){
                camlQuery = new SP.CamlQuery();
                camlQuery.set_listItemCollectionPosition(pos);// <- set position for next
batch
                camlQuery.set_viewXml(viewXml);
                items = list.getItems(camlQuery);
                clientContext.load(items);
                clientContext.executeQueryAsync(loadResults,showError);
            }
        }else{
            alert("Total Results: \nID: Title\n"+itemArray.join("\n")); // <- display when no
more results
        }
    }
    function showError(sender,args){
        alert(args.get_message());
    }
}
```

オンラインでJavaScriptクライアントオブジェクトモデルJSOMのをむ
https://riptutorial.com/ja/sharepoint/topic/1316/javascriptクライアントオブジェクトモデル-jsom-の

3: JavaScriptをしたモーダルダイアログボックスの

- var options = SP.UI. $ create_DialogOptions;
- var modalDialog = SP.UI.ModalDialog.showModalDialogoptions;

パラメーター

optionsプロパティ	
タイトル	ダイアログのタイトルをむ
URL	ダイアログにされるページのURLをむ。 **url**または**html**のいずれかをするがあります。 **url**が**html**よりもされます。
html	ダイアログにするHTML。
バツ	ダイアログのxオフセットをとしてします。
y	ダイアログのyオフセットをとしてします。
	ダイアログのをとしてします。されていない、サイズが**false**の、は768pxにされます。
さ	ダイアログのさをでします。されていない、サイズが**false**の、さは576pxにされます
allowMaximize	[ ]ボタンをするかどうかをするブール。
showMaximized	ダイアログがするかどうかをするブール。
ショークローズ	ダイアログにじるボタンをするかどうかをするブール。
autoSize	ダイアログプラットフォームがダイアログサイジングをにするかどうかをするブール。
dialogReturnValueCallback	リターンコールバックをするポインタ。タイプSP.UI.DialogResultの*dialogResult*、およびダイアログによってされたすべてのデータがまれているの*ReturnValue*オブジェクトは2つのパラメータをります。
args	ダイアログにされるデータをむオブジェクトです。

`SP.UI.ModalDialog`は、SharePoint 2010をしてJavaScriptオブジェクトモデルにされました。そのSharePointバージョン2013、Office365、および2016でできます。

そのの

- SP.UI.ModalDialog.showModalDialogのMSDNリファレンスオプション
- SP.UI.DialogResultのMSDNリファレンス

Examples

ダイアログボックスがじているときにアクションをする

```
SP.SOD.executeOrDelayUntilScriptLoaded(showDialog,"sp.js");

function showDialog(){
    var options = SP.UI.$create_DialogOptions();
    options.url = "/mySite/lists/myList/NewForm.aspx";
    options.dialogReturnValueCallback = myCallBackFunction;
    SP.UI.ModalDialog.showModalDialog(options);
    function myCallBackFunction(result,data){
        switch(result){
            case SP.UI.DialogResult.invalid:
                alert("The dialog result was invalid");
                break;
            case SP.UI.DialogResult.cancel:
                alert("You clicked cancel or close");
                break;
            case SP.UI.DialogResult.OK:
                alert("You clicked OK, creating an item in the list.");
                break;
        }
    }
}
```

のページをダイアログにする

```
SP.SOD.executeOrDelayUntilScriptLoaded(showDialog,"sp.js");

function showDialog(){
    SP.UI.ModalDialog.showModalDialog(
        { url: "/org/it/web/wik/Lists/ExampleCode/DispForm.aspx?ID=6" }
    );
}
```

カスタムダイアログをする

```
SP.SOD.executeOrDelayUntilScriptLoaded(showDialog,"sp.js");

function showDialog(){
    var dialogOptions = SP.UI.$create_DialogOptions();
    dialogOptions.title = "Your Title Here!";
    var dummyElement = document.createElement("div");
```

```
    dummyElement.style.textAlign = "center";
    dummyElement.appendChild(document.createElement("br"));
    dummyElement.appendChild(document.createTextNode("Some beautifully crafted text."));
    dummyElement.appendChild(document.createElement("br"));
    dialogOptions.html = dummyElement;
    SP.UI.ModalDialog.showModalDialog(dialogOptions);
}
```

オンラインでJavaScriptをしたモーダルダイアログボックスのをむ
https://riptutorial.com/ja/sharepoint/topic/6868/javascriptをしたモーダルダイアログボックスの

4: RESTサービス

RESTサービスのエンドポイントURL

RESTクライアントアクセスAPIは、SharePoint 2010でめてされましたが、SharePoint 2013ではにされました。SharePoint 2010のREST APIは、 `/_vti_bin/ListData.svc` URLのListData Webサービスからアクセスされます 。 SharePoint 2013では`/_api/lists/`および`/_api/web`エンドポイントURLがされました。これらのURLはがなります。

のエンドポイントURLはがされなければならない`http://server/site` `server`サーバーのをし、 `site`の、またはのサイトへのパスをします。

のURLの	SharePoint 2010	SharePoint 2013
リストをする	`/_vti_bin/ListData.svc/ListName`	`/_api/lists('ListGuid')`
アイテムをする	`/_vti_bin/ListData.svc/ListName(1)`	`/_api/lists('ListGuid')/items(1)`
Webをする		`/_api/web`

リストとリストアイテムへのアクセスのいにもかかわらず、それらのをうはのバージョンでにています。

`ListData.svc`サービスは、のためにSharePoint 2013できできます。

RESTの

RESTリクエストは、ネイティブJavaScript XMLHttpRequestまたはjQuery AJAXラッパーをしてできます。

XMLHttpRequestの

```
var xhr = new XMLHttpRequest();
xhr.open(verb, url, true);
xhr.setRequestHeader("Content-Type","application/json");
xhr.send(data);
```

jQuery AJAXの

```
$.ajax({
    method: verb,
```

```
    url: url,
    headers: { "Content-Type":"application/json" },
    data: data
});
```

AJAXでリクエストをするのについては、JavaScript AJAXのドキュメントをしてください。

Examples

リストの

リストアイテムの

このでは、すべてのリストアイテムをし、それらのアイテムをするをします。 `top`パラメーターをして、ののをすることができます。 `select`パラメータをしてのフィールド `$select=id, Title, uri` をすることも`select`ます。

JavaScript

```
function GetListItems(){
    $.ajax({
        url: "../_api/web/lists/getbytitle('List Title')/items?$top=50"
        contentType: "application/json;odata=verbose",
        method: "GET",
        headers: { "accept": "application/json;odata=verbose" },
        success: function (data) {
            $.each(data.d.results, function(index,item){
                //use item to access the individual list item
                console.log(item.Id);
            });
        },
        error: function(error){
            console.log(error);
        }
    });
    }
```

々のリストをする

JavaScript

```
function GetListItem(){
    $.ajax({
        url: "../_api/web/lists/getbytitle('List Title')/items(1)",
        contentType: "application/json;odata=verbose",
        method: "GET",
        headers: { "accept": "application/json;odata=verbose" },
        success: function (data) {
            console.log(data.d.Id);
        },
        error: function(error){
            console.log(error);
        }
    });
```

```
    }
```

ルックアップをつリストアイテムをする

によっては、のようなリストをつことがあります。

リスト

	タイプ	
タイトル	テキスト	の
		の
		の
タイプ	ルックアップテーブル	ルックアップフィールドテーブルからののドロップダウン

テーブル

	タイプ	
タイトル	テキスト	/のブタ
NumLegs		のの

らくもなではないかもしれませんが、ここのはとしてです。 SharePointリストからをするためにのリクエストをする、の`Type`については、JSONレスポンスに`TypeId`というフィールドしかされません。これらのを1のAJAXびしでするには、URLパラメータになマークアップがです。

このは、のものにもされます。 `People/Groups`カラムをしているは、にはルックアップだけなので、 `Title` 、 `EMail`などのアイテムをに`EMail`できます。

コード

な ルックアップからすフィールドをするときは、のテーブルのルックアップフィールドのをフィールドのにけるがあります。たとえば、ルックアップから`NumLegs`を`NumLegs`は、「 `Type/NumLegs`するがあります。

JavaScript

```
// webUrl: The url of the site (ex. https://www.contoso.com/sites/animals)
// listTitle: The name of the list you want to query
// selectFields: the specific fields you want to get back
// expandFields: the name of the fields that need to be pulled from lookup tables
// callback: the name of the callback function on success
function getItems(webUrl,listTitle,selectFields, expandFields, callback){
    var endpointUrl = webUrl + "/_api/web/lists/getbytitle('" + listTitle + "')/items";
    endpointUrl+= '?$select=' + selectFields.join(",");
    endpointUrl+= '&$expand=' + expandFields.join(",");
    return executeRequest(endpointUrl,'GET', callback);
}

function executeRequest(url,method,callback,headers,payload)
{
    if (typeof headers == 'undefined'){
        headers = {};
    }
    headers["Accept"] = "application/json;odata=verbose";
    if(method == "POST") {
        headers["X-RequestDigest"] = $("#__REQUESTDIGEST").val();
    }

    var ajaxOptions =
    {
    url: url,
    type: method,
    contentType: "application/json;odata=verbose",
    headers: headers,
    success: function (data) { callback(data) }
    };
    if(method == "POST") {
    ajaxOptions.data = JSON.stringify(payload);
    }

    return $.ajax(ajaxOptions);
}

// Setup the ajax request by setting all of the arguments to the getItems function
function getAnimals() {
    var url = "https://www.contoso.com/sites/animals";
    var listTitle = "AnimalListing";

    var selectFields = [
        "Title",
        "Age",
        "Value",
        "Type/Title",
        "Type/NumLegs"
    ];

    var expandFields = [
        "Type/Title",
        "Type/NumLegs"
    ];

    getItems(url, listTitle, selectFields, expandFields, processAnimals);
}

// Callback function
```

```
// data: returns the data given by SharePoint
function processAnimals(data) {
    console.log(data);
    // Process data here
}

// Start the entire process
getAnimals();
```

のフィールドにをする

このでは、ルックアップのが`MultiLookupColumnName`で、マルチルックアップフィールドをID 1と2のをするようにするとしています。

jQuery AJAXの

2010

```
var listName = "YourListName";
var lookupList = "LookupListName";
var idOfItemToUpdate = 1;
var url = "/server/site/_vti_bin/ListData.svc/"+listName+"("+idOfItemToUpdate+")";
var data = JSON.stringify({
    MultiLookupColumnName:[
        {__metadata:{uri:"http://yoursiteurl/_vti_bin/ListData.svc/"+lookupList+"(1)"}},
        {__metadata:{uri:"http://yoursiteurl/_vti_bin/ListData.svc/"+lookupList+"(2)"}}
    ]
});
$.ajax({
    method: 'POST',
    url: url,
    contentType: 'application/json',
    headers: {
      "X-HTTP-Method" : "MERGE",
      "If-Match" : "*"
    },
    data: data
});
```

2013

```
var listGuid = "id-of-list-to-update"; // use list GUID here
var lookupGuid = "id-of-lookup-list"; // use lookup list GUID here
var idOfItemToUpdate = 1;
var url = "/server/site/_api/lists('"+ listGuid + "')/items("+ idOfItemToUpdate + ")";
var data = JSON.stringify({
    MultiLookupColumnName:[
        {__metadata:{uri:"http://yoursiteurl/_api/lists('" + lookupGuid + "')/items(1)"}},
        {__metadata:{uri:"http://yoursiteurl/_api/lists('" + lookupGuid + "')/items(2)"}}
    ]
});
$.ajax({
    method: 'POST',
    url: url,
    contentType: 'application/json',
    headers: {
      "X-HTTP-Method" : "MERGE",
```

```
      "If-Match" : "*"
    },
    data: data
});
```

XMLHttpRequestの

2010

```
var listName = "YourListName";
var lookupList = "LookupListName";
var idOfItemToUpdate = 1;
var url = "/server/site/_vti_bin/ListData.svc/YourListName("+idOfItemToUpdate+")";
var data = JSON.stringify({
    MultiLookupColumnName:[
       {__metadata:{uri:"http://yoursiteurl/_vti_bin/ListData.svc/"+lookupList+"(1)"}},
        {__metadata:{uri:"http://yoursiteurl/_vti_bin/ListData.svc/"+lookupList+"(2)"}}
    ]
});
var xhr = new XMLHttpRequest();
xhr.open("POST",url,true);
xhr.setRequestHeader("X-HTTP-Method", "MERGE");
xhr.setRequestHeader("If-Match", "*");
xhr.setRequestHeader("Content-Type","application/json");
xhr.send(data);
```

2013

```
var listGuid = "id-of-list-to-update";
var lookupGuid = "id-of-lookup-list";
var idOfItemToUpdate = 1;
var url = "/server/site/_api/lists('"+ listGuid + "')/items("+ idOfItemToUpdate + ")";
var data = JSON.stringify({
    MultiLookupColumnName:[
        {__metadata:{uri:"http://yoursiteurl/_api/lists('" + lookupGuid + "')/items(1)"}},
        {__metadata:{uri:"http://yoursiteurl/_api/lists('" + lookupGuid + "')/items(2)"}}
    ]
});
var xhr = new XMLHttpRequest();
xhr.open("POST",url,true);
xhr.setRequestHeader("X-HTTP-Method", "MERGE");
xhr.setRequestHeader("If-Match", "*");
xhr.setRequestHeader("Content-Type","application/json");
xhr.send(data);
```

クエリからされたリストアイテムのページング

RESTをしてページングをシミュレートするには、のようにします。

1. `$orderby`パラメータにっての`n`エントリをスキップするには、 `$skip=n`パラメータをします
2. `$top=n`パラメータをして、 `$orderby`および`$skip`パラメータにって`n`エントリをします。

```
var endpointUrl = "/_api/lists('guid')/items"; // SP2010: "/_vti_bin/ListData.svc/ListName";
$.getJSON(
    endpointUrl + "?$orderby=Id&$top=1000",
```

```
    function(data){
        processData(data); // you can do something with the results here
        var count = data.d.results.length;
        getNextBatch(count, processData, onComplete); // fetch next page
    }
);

function getNextBatch(totalSoFar, processResults, onCompleteCallback){
    $.getJSON(
        endpointUrl + "?$orderby=Id&$skip="+totalSoFar+"&$top=1000",
        function(data){
            var count = data.d.results.length;
            if(count > 0){
                processResults(data); // do something with results
                getNextBatch(totalSoFar+count, callback); // fetch next page
            }else{
                onCompleteCallback();
            }
        }
    );
}
```

SharePointリストでしくされたアイテムのIDをする

このは、SharePoint REST APIをしてしくされたアイテムのIDをするをしています。

listName - このにはあなたのリストのがまれます。

newItemBody - これは、リストにしいアイテムをするためのリクエストボディです。

var newItemBody = {__metadata{'type' 'SP.Data.MyListNameItem'}、タイトル 'Some title value'};

```
function CreateListItemWithDetails(listName, newItemBody) {

    var item = newItemBody;
    return $.ajax({
        url: _spPageContextInfo.siteAbsoluteUrl + "/_api/web/lists/getbytitle('" + listName +
"')/items",
        type: "POST",
        contentType: "application/json;odata=verbose",
        data: JSON.stringify(item),
        headers: {
            "Accept": "application/json;odata=verbose",
            "X-RequestDigest": $("#__REQUESTDIGEST").val(),
            "content-Type": "application/json;odata=verbose"
        }

    });
}

CreateListItemWithDetails(listName, newItemBody)
    .then(function(data){
        //success callback
        var NewlyCreatedItemId = data.d.ID;
    }, function(data){
        //failure callback
    });
```

SharePoint 2010 REST インターフェイスをしてCRUDをする

する

RESTをしてCreateをするには、のをするがあります。

POSTをしてHTTPをします。 POSTのとしてエンティティをするリストのサービスURLをします。コンテンツタイプをapplication/jsonしapplication/json。 しいリストをすJSONオブジェクトをとしてシリアルし、このをリクエストにします。JavaScriptの

```
function createListItem(webUrl,listName, itemProperties, success, failure) {

    $.ajax({
        url: webUrl + "/_vti_bin/listdata.svc/" + listName,
        type: "POST",
        processData: false,
        contentType: "application/json;odata=verbose",
        data: JSON.stringify(itemProperties),
        headers: {
            "Accept": "application/json;odata=verbose"
        },
        success: function (data) {
            success(data.d);
        },
        error: function (data) {
            failure(data.responseJSON.error);
        }
    });
}
```

```
var taskProperties = {
    'TaskName': 'Order Approval',
    'AssignedToId': 12
};

createListItem('https://contoso.sharepoint.com/project/','Tasks',taskProperties,function(task){

    console.log('Task' + task.TaskName + ' has been created');
  },
  function(error){
    console.log(JSON.stringify(error));
  }
);
```

む

RESTでみりをするには、のをするがあります。

GET verbをしてHTTPをします。 GETのとしてエンティティをするリストのサービスURLをします。コンテンツタイプをapplication/jsonしapplication/json。 JavaScriptの

```
function getListItemById(webUrl,listName, itemId, success, failure) {
    var url = webUrl + "/_vti_bin/listdata.svc/" + listName + "(" + itemId + ")";
    $.ajax({
```

```
        url: url,
        method: "GET",
        headers: { "Accept": "application/json; odata=verbose" },
        success: function (data) {
            success(data.d);
        },
        error: function (data) {
            failure(data.responseJSON.error);
        }
    });
}
```

```
getListItemById('https://contoso.sharepoint.com/project/','Tasks',2,function(taskItem){
    console.log(taskItem.TaskName);
  },
  function(error){
    console.log(JSON.stringify(error));
  }
);
```

のエンティティをするには、のをするがあります。

`POST`をしてHTTPをします。が`MERGE` `X-HTTP-Method`ヘッダーをし`X-HTTP-Method`。 するリストのサービスURLをPOSTのターゲットとしてする`If-Match`ヘッダーにエンティティののETagまたは*のをします。 JavaScriptの

```
function updateListItem(webUrl,listName,itemId,itemProperties,success, failure)
{
   getListItemById(webUrl,listName,itemId,function(item){

      $.ajax({
         type: 'POST',
         url: item.__metadata.uri,
         contentType: 'application/json',
         processData: false,
         headers: {
                "Accept": "application/json;odata=verbose",
                "X-HTTP-Method": "MERGE",
                "If-Match": item.__metadata.etag
         },
         data: Sys.Serialization.JavaScriptSerializer.serialize(itemProperties),
         success: function (data) {
                success(data);
         },
         error: function (data) {
                failure(data);
         }
      });

   },
   function(error){
       failure(error);
   });

}
```

```
var taskProperties = {
    'TaskName': 'Approval',
    'AssignedToId': 12
};


updateListItem('https://contoso.sharepoint.com/project/','Tasks',2,taskProperties,function(item){

    console.log('Task has been updated');
  },
  function(error){
    console.log(JSON.stringify(error));
  }
);
```

エンティティをするには、のをするがあります。

`POST`をしてHTTPをします。が`DELETE` `X-HTTP-Method`ヘッダーをし`X-HTTP-Method`。 するリストのサービスURLをPOSTのターゲットとしてする`If-Match`ヘッダーにエンティティののETagのをします。 JavaScriptの

```
function deleteListItem(webUrl, listName, itemId, success, failure) {
    getListItemById(webUrl,listName,itemId,function(item){
        $.ajax({
            url: item.__metadata.uri,
            type: "POST",
            headers: {
                "Accept": "application/json;odata=verbose",
                "X-Http-Method": "DELETE",
                "If-Match": item.__metadata.etag
            },
            success: function (data) {
                success();
            },
            error: function (data) {
                failure(data.responseJSON.error);
            }
        });
    },
   function (error) {
       failure(error);
   });
}
```

```
deleteListItem('https://contoso.sharepoint.com/project/','Tasks',3,function(){
    console.log('Task has been deleted');
  },
  function(error){
    console.log(JSON.stringify(error));
  }
);
```

オンラインでRESTサービスをむ https://riptutorial.com/ja/sharepoint/topic/3045/restサービス

5: SharePoint 2013クライアントのレンダリング

き

クライアントサイドレンダリングCSRは、SharePoint 2013でされたしいコンセプトです。SharePointページでホストされているのコントロールにして、のレンダリングをできるメカニズムをしますリストビュー、リストフォーム。クライアントサイトのレンダリングは、サーバーではなくクライアントをしてデータがされるときだけです。これは、XSLTをするのではなく、HTMLやJavaScriptなどのクライアントのをすることをします。

Examples

CSRをしてリストビューのフィールド/のハイパーリンクをする

のは、CSRをしてリストビューの " **ID** "と " **TitleLinkTitle** "フィールドのハイパーリンクをするをしています。

ステップ**1JS**ファイルをし、のコードをりけます

```
(function () {

    function registerRenderer() {
        var ctxForm = {};
        ctxForm.Templates = {};

        ctxForm.Templates = {
            Fields : {
                'LinkTitle': { //------ Change Hyperlink of LinkTitle
                    View : function (ctx) {
                        var url = String.format('{0}?ID={1}',
"/sites/Lists/testlist/EditItem.aspx", ctx.CurrentItem.ID);
                        return String.format('<a href="{0}" onclick="EditItem2(event,
\'{0}\');return false;">{1}</a>', url, ctx.CurrentItem.Title);
                    }
                },
                'ID' : { //------ Change Hyperlink from ID field
                    View : function (ctx) {
                        var url = String.format('{0}?ID={1}',
"/IssueTracker/Lists/testlist/DisplayItem.aspx", ctx.CurrentItem.ID);
                        return String.format('<a href="{0}" onclick="EditItem2(event,
\'{0}\');return false;">{1}</a>', url, ctx.CurrentItem.ID);
                    }
                },

            }
        };
        SPClientTemplates.TemplateManager.RegisterTemplateOverrides(ctxForm);
    }
```

```
    ExecuteOrDelayUntilScriptLoaded(registerRenderer, 'clienttemplates.js');

})();
```

ステップ**2List View**の**Web**パーツのプロパティにし、このしくされた**js**ファイルに**JS Link**リファレンスをします**sitecollection / SiteAssets / CSRCodeFile.js**

JSlinkはこのでのみしてください。 "sitecollection / YourJSfFilePath"

ステップ3Appy and Done

CSRをしてSharePointリストビューからをにします。

このは、CSRをしてSharePointリストビューから「」フィールドをにするをしています。

```
(function () {

    function RemoveFields(ctx) {
        var fieldName = "Date"; // here Date is field or column name to be hide
        var header = document.querySelectorAll("[displayname=" + fieldName +
"]")[0].parentNode;
        var index = [].slice.call(header.parentNode.children).indexOf(header) + 1;
        header.style.display = "none";
        for (var i = 0, cells = document.querySelectorAll("td:nth-child(" + index + ")"); i <
cells.length; i++) {
            cells[i].style.display = "none";
        }
    }

    function registerRenderer() {
        var ctxForm = {};
        ctxForm.Templates = {};
        ctxForm.OnPostRender = RemoveFields;
        SPClientTemplates.TemplateManager.RegisterTemplateOverrides(ctxForm);
    }
    ExecuteOrDelayUntilScriptLoaded(registerRenderer, 'clienttemplates.js');

})();
```

CSRをして/フォームでをする

SharePointのリストがあり、それには4つのフィールドがあるとします。タイトル、、メール、などをします。/アイテムのフォームでカスタムをするは、にCSRコードをできます。のフォームは、のをすることができます

- フィールドの
- によるメールIDのチェック
- モバイルのでチェック
- [フルネーム]フィールドにをめることはできません

ステップ**1** JSファイルをして、 `CSRValidations.js`とし、のコードをJSファイルにコピーします

```
        (function () {

        // Create object that have the context information about the field that we want to
change it's output render
        var fieldContext = {};
        fieldContext.Templates = {};
        fieldContext.Templates.Fields = {
            // Apply the new rendering for Email field on New and Edit Forms
            "Title": {
                "NewForm": titleFieldTemplate,
                "EditForm":  titleFieldTemplate
            },
            "Full_x0020_Name": {
                "NewForm": fullNameFieldTemplate,
                "EditForm":  fullNameFieldTemplate
            },
            "Email": {
                "NewForm": emailFieldTemplate,
                "EditForm":  emailFieldTemplate
            },
            "Mobile_x0020_Phone": {
                "NewForm": mobilePhoneFieldTemplate,
                "EditForm":  mobilePhoneFieldTemplate
            }
        };

        SPClientTemplates.TemplateManager.RegisterTemplateOverrides(fieldContext);

    })();

    // This function provides the rendering logic
    function emailFieldTemplate(ctx) {

        var formCtx = SPClientTemplates.Utility.GetFormContextForCurrentField(ctx);

        // Register a callback just before submit.
        formCtx.registerGetValueCallback(formCtx.fieldName, function () {
            return document.getElementById('inpEmail').value;
        });

        //Create container for various validations
        var validators = new SPClientForms.ClientValidation.ValidatorSet();
        validators.RegisterValidator(new emailValidator());

        // Validation failure handler.
        formCtx.registerValidationErrorCallback(formCtx.fieldName, emailOnError);

        formCtx.registerClientValidator(formCtx.fieldName, validators);

        return "<span dir='none'><input type='text' value='" + formCtx.fieldValue + "'
maxlength='255' id='inpEmail' class='ms-long'> \ <br><span id='spnEmailError' class='ms-
formvalidation ms-csrformvalidation'></span></span>";
    }

    // Custom validation object to validate email format
    emailValidator = function () {
        emailValidator.prototype.Validate = function (value) {
            var isError = false;
            var errorMessage = "";

            //Email format Regex expression
```

```
            //var emailRejex = /\S+@\S+\.\S+/;
            var emailRejex = /^(([^<>()[\]HYPERLINK
"\\.,;:\s@\"\\.,;:\s@\"]+(\.[^<>()[\]HYPERLINK "\\.,;:\s@\"\\.,;:\s@\"]+)*)|(\".+\"))@((\[[0-
9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/;

            if (value.trim() == "") {
                isError = true;
                errorMessage = "You must specify a value for this required field.";
            }else if (!emailRejex.test(value) && value.trim()) {
                isError = true;
                errorMessage = "Please enter valid email address";
            }

            //Send error message to error callback function (emailOnError)
            return new SPClientForms.ClientValidation.ValidationResult(isError, errorMessage);

        };
    };

    // Add error message to spnError element under the input field element
    function emailOnError(error) {
        document.getElementById("spnEmailError").innerHTML = "<span role='alert'>" +
error.errorMessage + "</span>";
    }

    // This function provides the rendering logic
    function titleFieldTemplate(ctx) {

        var formCtx = SPClientTemplates.Utility.GetFormContextForCurrentField(ctx);
        // Register a callback just before submit.

        formCtx.registerGetValueCallback(formCtx.fieldName, function () {
            return document.getElementById('inpTitle').value;
        });

        //Create container for various validations
        var validators = new SPClientForms.ClientValidation.ValidatorSet();
        validators.RegisterValidator(new titleValidator());

        // Validation failure handler.
        formCtx.registerValidationErrorCallback(formCtx.fieldName, titleOnError);

        formCtx.registerClientValidator(formCtx.fieldName, validators);

        return "<span dir='none'><input type='text' value='" + formCtx.fieldValue + "'
maxlength='255' id='inpTitle' class='ms-long'> \ <br><span id='spnTitleError' class='ms-
formvalidation ms-csrformvalidation'></span></span>";
    }

    // Custom validation object to validate title format
    titleValidator = function () {
        titleValidator.prototype.Validate = function (value) {
            var isError = false;
            var errorMessage = "";

            if (value.trim() == "") {
                isError = true;
                errorMessage = "You must specify a value for this required field.";
            }

            //Send error message to error callback function (titleOnError)
```

```
            return new SPClientForms.ClientValidation.ValidationResult(isError, errorMessage);

        };
    };

    // Add error message to spnError element under the input field element
    function titleOnError(error) {
        document.getElementById("spnTitleError").innerHTML = "<span role='alert'>" +
error.errorMessage + "</span>";
    }

    // This function provides the rendering logic
    function mobilePhoneFieldTemplate(ctx) {

        var formCtx = SPClientTemplates.Utility.GetFormContextForCurrentField(ctx);

        // Register a callback just before submit.
        formCtx.registerGetValueCallback(formCtx.fieldName, function () {
            return document.getElementById('inpMobilePhone').value;
        });

        //Create container for various validations
        var validators = new SPClientForms.ClientValidation.ValidatorSet();
        validators.RegisterValidator(new mobilePhoneValidator());

        // Validation failure handler.
        formCtx.registerValidationErrorCallback(formCtx.fieldName, mobilePhoneOnError);

        formCtx.registerClientValidator(formCtx.fieldName, validators);

        return "<span dir='none'><input type='text' value='" + formCtx.fieldValue + "'
maxlength='255' id='inpMobilePhone' class='ms-long'> \ <br><span id='spnMobilePhoneError'
class='ms-formvalidation ms-csrformvalidation'></span></span>";
    }

    // Custom validation object to validate mobilePhone format
    mobilePhoneValidator = function () {
        mobilePhoneValidator.prototype.Validate = function (value) {
            var isError = false;
            var errorMessage = "";

            //MobilePhone format Regex expression
            //var mobilePhoneRejex = /\S+@\S+\.\S+/;
            var mobilePhoneRejex = /^[0-9]+$/;

            if (value.trim() == "") {
                isError = true;
                errorMessage = "You must specify a value for this required field.";
            }else if (!mobilePhoneRejex.test(value) && value.trim()) {
                isError = true;
                errorMessage = "Please enter valid mobile phone number";
            }

            //Send error message to error callback function (mobilePhoneOnError)
            return new SPClientForms.ClientValidation.ValidationResult(isError, errorMessage);

        };
    };

    // Add error message to spnError element under the input field element
    function mobilePhoneOnError(error) {
```

```
        document.getElementById("spnMobilePhoneError").innerHTML = "<span role='alert'>" +
error.errorMessage + "</span>";
    }

    // This function provides the rendering logic
    function fullNameFieldTemplate(ctx) {

        var formCtx = SPClientTemplates.Utility.GetFormContextForCurrentField(ctx);

        // Register a callback just before submit.
        formCtx.registerGetValueCallback(formCtx.fieldName, function () {
            return document.getElementById('inpFullName').value;
        });

        //Create container for various validations
        var validators = new SPClientForms.ClientValidation.ValidatorSet();
        validators.RegisterValidator(new fullNameValidator());

        // Validation failure handler.
        formCtx.registerValidationErrorCallback(formCtx.fieldName, fullNameOnError);

        formCtx.registerClientValidator(formCtx.fieldName, validators);

        return "<span dir='none'><input type='text' value='" + formCtx.fieldValue + "'
maxlength='255' id='inpFullName' class='ms-long'> \ <br><span id='spnFullNameError' class='ms-
formvalidation ms-csrformvalidation'></span></span>";
    }

    // Custom validation object to validate fullName format
    fullNameValidator = function () {
        fullNameValidator.prototype.Validate = function (value) {
            var isError = false;
            var errorMessage = "";

            //FullName format Regex expression
            var fullNameRejex = /^[a-z ,.'-]+$/i;

            if (value.trim() == "") {
                isError = true;
                errorMessage = "You must specify a value for this required field.";
            }else if (!fullNameRejex.test(value) && value.trim()) {
                isError = true;
                errorMessage = "Please enter valid name";
            }

            //Send error message to error callback function (fullNameOnError)
            return new SPClientForms.ClientValidation.ValidationResult(isError, errorMessage);

        };
    };

    // Add error message to spnError element under the input field element
    function fullNameOnError(error) {
        document.getElementById("spnFullNameError").innerHTML = "<span role='alert'>" +
error.errorMessage + "</span>";
    }
```

ステップ**2**ブラウザでアイテムフォームをきます。ページをしてWebパーツをします。

3 Webパーツのプロパティで、[その] - > [JSリンク] - > jsファイルのパスをりけますsitecollection

/ SiteAssets / CSRValidations.js

4 Webパーツのプロパティとページをします。

CSRをしてリストビューでのをする

リストビューでのをするがあるがあります

たとえば、ビューにされるが「IsApprovalNeeded」で、「がですか」としたいなどです。

もちろん、リストのでのタイトルをすることで、のをすることはできますが、リストのままにして、ページプレビューでするは、 CSRクライアントレンダリング

ここにコードがあります...

```
(function () {

    function preTaskFormRenderer(renderCtx) {
       modifyColumns(renderCtx);
    }

    function modifyColumns(renderCtx)
    {
      var arrayLength= renderCtx.ListSchema.Field.length;
        for (var i=0; i < arrayLength;i++)
        {
           if(renderCtx.ListSchema.Field[i].DisplayName == 'IsApprovalNeeded')
             {
               var newTitle= "Is Approval Needed?";
               var linkTitleField = renderCtx.ListSchema.Field[i];
               linkTitleField.DisplayName = newTitle;
             }
         }
    }

    function registerRenderer()
    {
      var ctxForm = {};
      ctxForm.Templates = {};
      ctxForm.OnPreRender = preTaskFormRenderer;
      SPClientTemplates.TemplateManager.RegisterTemplateOverrides(ctxForm);
    }

    ExecuteOrDelayUntilScriptLoaded(registerRenderer, 'clienttemplates.js');

})();
```

オンラインでSharePoint 2013クライアントのレンダリングをむ
https://riptutorial.com/ja/sharepoint/topic/8317/sharepoint-2013クライアントのレンダリング

6: SharePointアプリケーション

き

SharePoint Hosted App

サイトからのがです http : //www.letsharepoint.com/what-is-user-information-list-in-sharepoint-2013/

Examples

SharePoint 2013SharePoint 2013でJSOMをしてユーザープロファイルサービスデータにアクセスする

SharePoint 2013SharePoint 2013でJSOMをしてユーザープロファイルサービスデータにアクセスする

このでは、JSOMJavascript Object ModelをしてUser Profile ServiceUPSアプリケーションをまたはアクセスし、なアプリケーションをするをびます。めるに、になUPSをしてみましょう。

ユーザープロファイル - の々のすべてのをしたでっています。これは、ユーザーにするAccountName、FirstName、LastName、WorkEmailなどのすべてのプロパティをします。

ユーザープロファイルサービスアプリケーション - すべてのユーザープロファイルをするされたとなされ、はプロファイル、プロファイルの、サイト、ソーシャルタグなどをまたはできます。また、Active Directoryなどのディレクトリサービスからをきすこともできます。

サイト -々のユーザーがのをし、、リンクなどをするためのパーソナライズされたサイト。ユーザーがやのにするをできるようにすることでなネットワーキングとソーシャルをします。のサイトにアクセスするには、SharePointページのにあるユーザーをクリックします。

ユーザープロファイルデータのとアクセス

JSOMをしてするので、JSOMまたはCSOMまたはRESTをしてプロフィールをできるというをいて、「みり」のみをできます。

*サーバーサイドコードにより、ののみり/きみがです。

JSOMをしてユーザープロファイルのプロパティをする

SharePoint Hosted Appをし、そのアプリでユーザーをできます。

Visual Studio 2013をし、[プロジェクト]から[SharePoint 2013のアプリケーション]をします。のプロジェクトタイプをすると、SharePointサイトにし、するアプリケーションのタイプをするウ

ィンドウがされますのスクリーンショットを。ここでは、SharePoint OnlineのサイトURLとSharePoint Hosted Appをしました。 Finishをクリックします。

3.プロジェクトがされると、デフォルトでソリューションエクスプローラにされたのフォルダ/ファイラーがプロジェクトにされます。

4. "Default.aspx"ページをくと、すでにページにされているJavaScriptライブラリがいくつかつかります。

ここで、もう1つのライブラリをして、User Profilesでのをするがあります

オンラインでSharePointアプリケーションをむ
https://riptutorial.com/ja/sharepoint/topic/9876/sharepointアプリケーション

7: プロバイダのアプリケーションをする

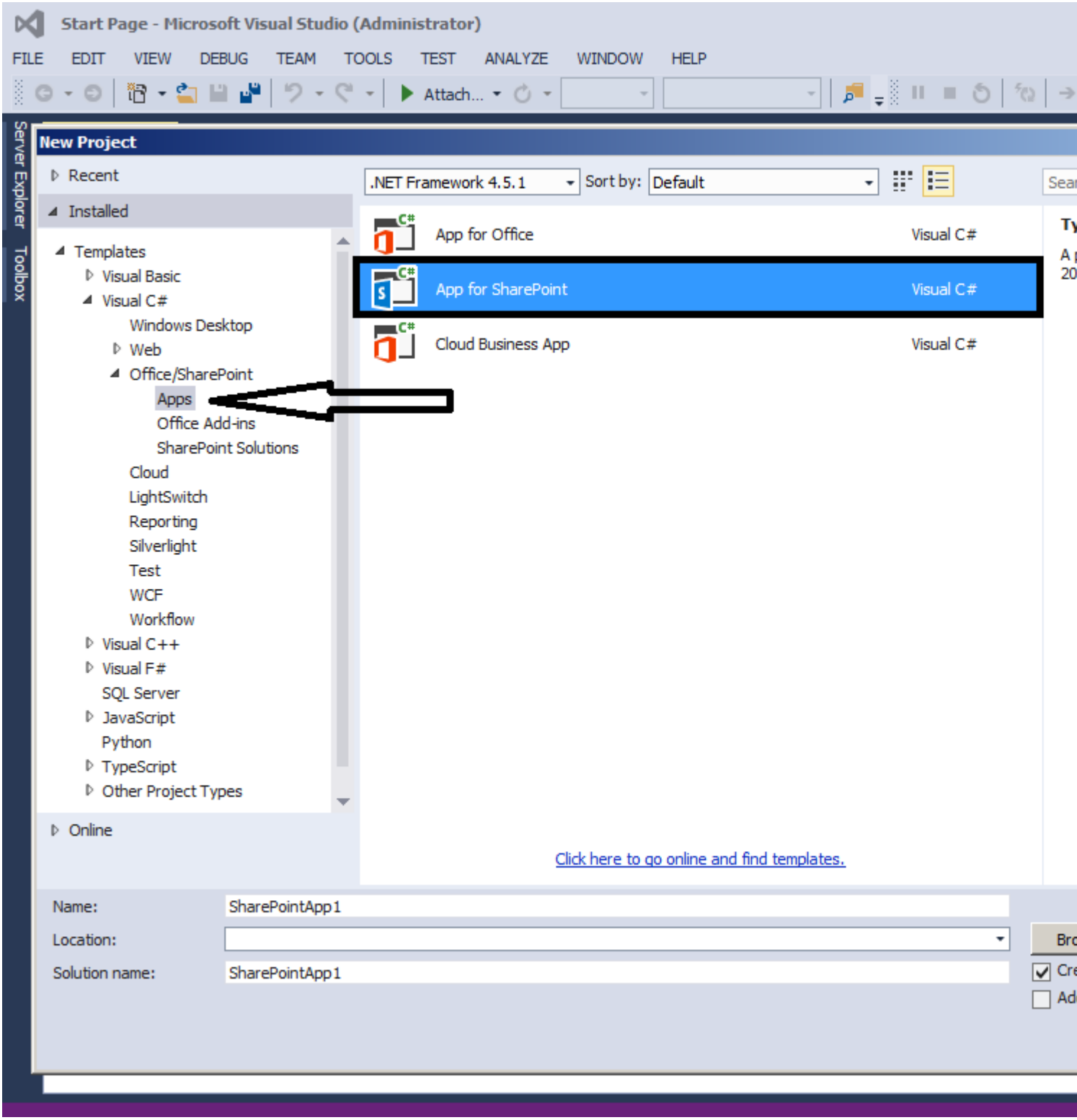
Examples

の

アプリケーションからめるには、Visual studio 2013がです。のコミュニティやのはこちらからダウンロードできます。> https://www.visualstudio.com/products/free-developer-offers-vs

それがダウンロードされ、インストールされると

> いてクリックしてしいプロジェクトをする
>
> Office / SharePointセクションをすると、のようにAppのオプションがされます。

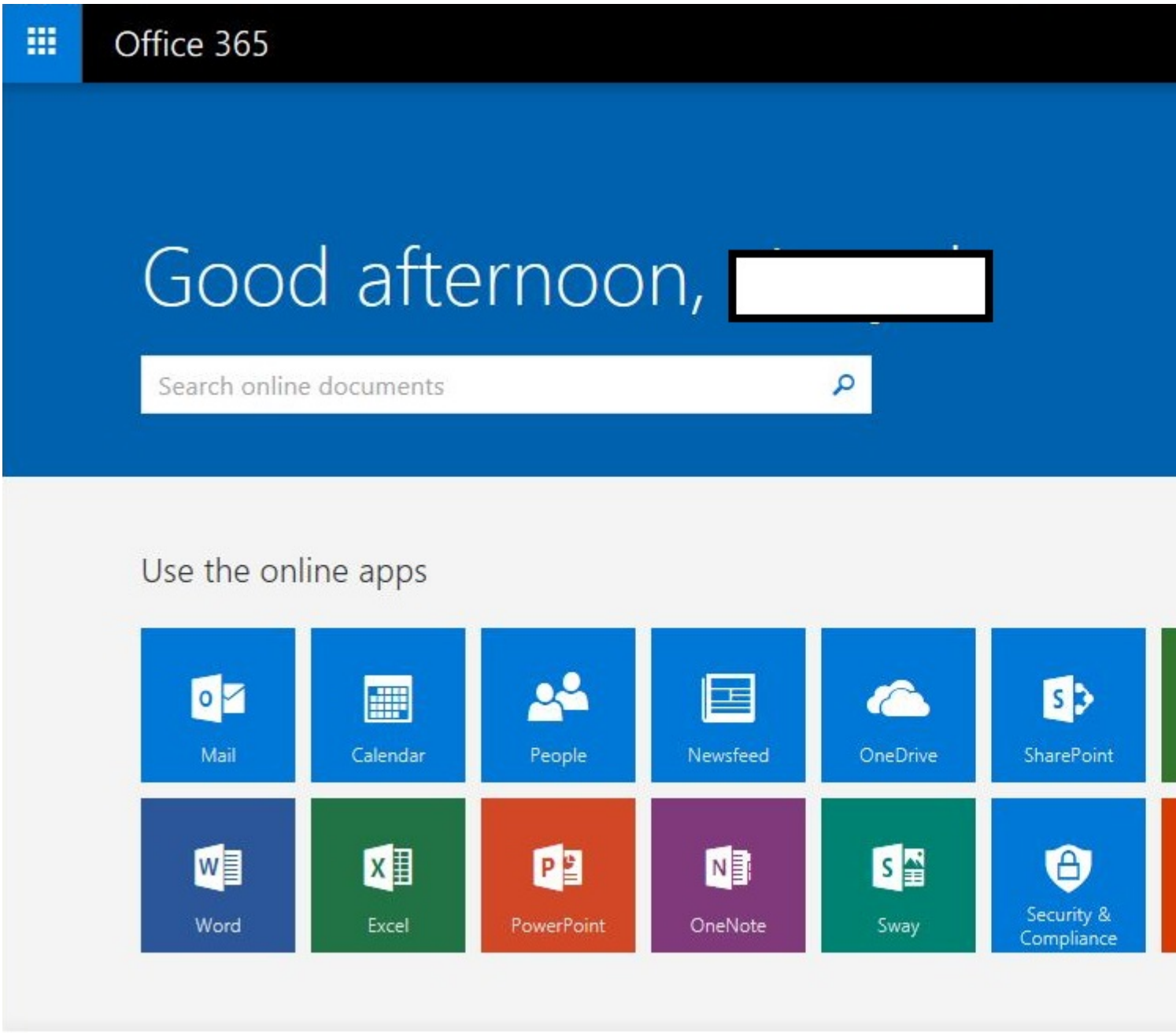


Appオプションができないvsをし、 **Microsoft Office Developer Tools**をダウンロードしてインストールします。https://www.visualstudio.com/en-us/features/office-tools-vs.aspx

サイトの

ビジュアルスタジオをしたら、アプリケーションをSharePointにするためのサイトがです。もなはのとおりです>の1Office 365デベロッパーアカウントにサインアップする https://profile.microsoft.com/RegSysProfileCenter/wizardnp.aspx?wizid=14b845d0-938c-45af-b061-f798fbb4d170&lcid=1033

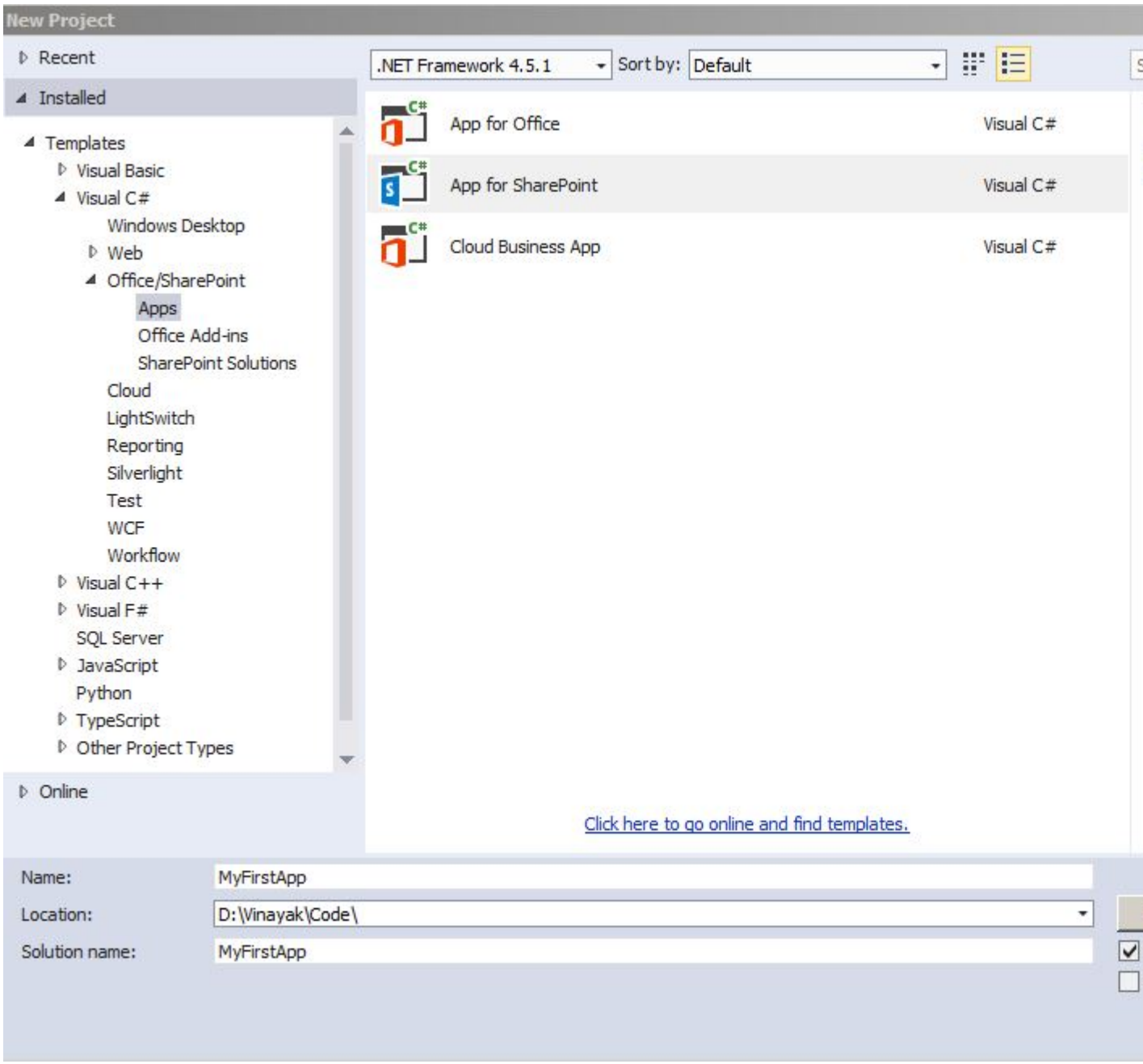
きがしたら、すべてのアプリのhttps://www.office.com/ center URL



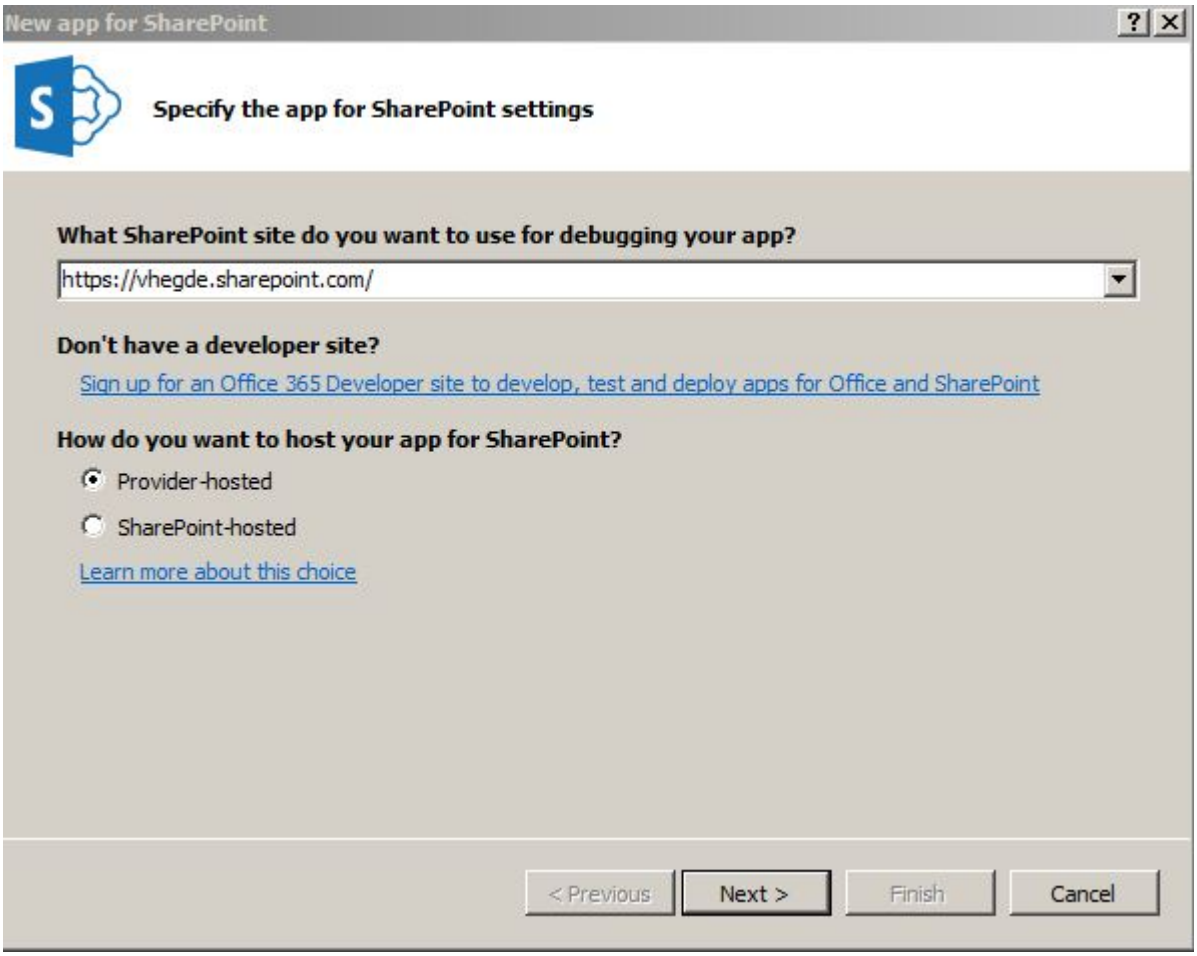
Visual Studioでアプリケーションをする

のアプリのからめましょう

1. ビジュアルスタジオをき、しいプロジェクトをする
2. とを

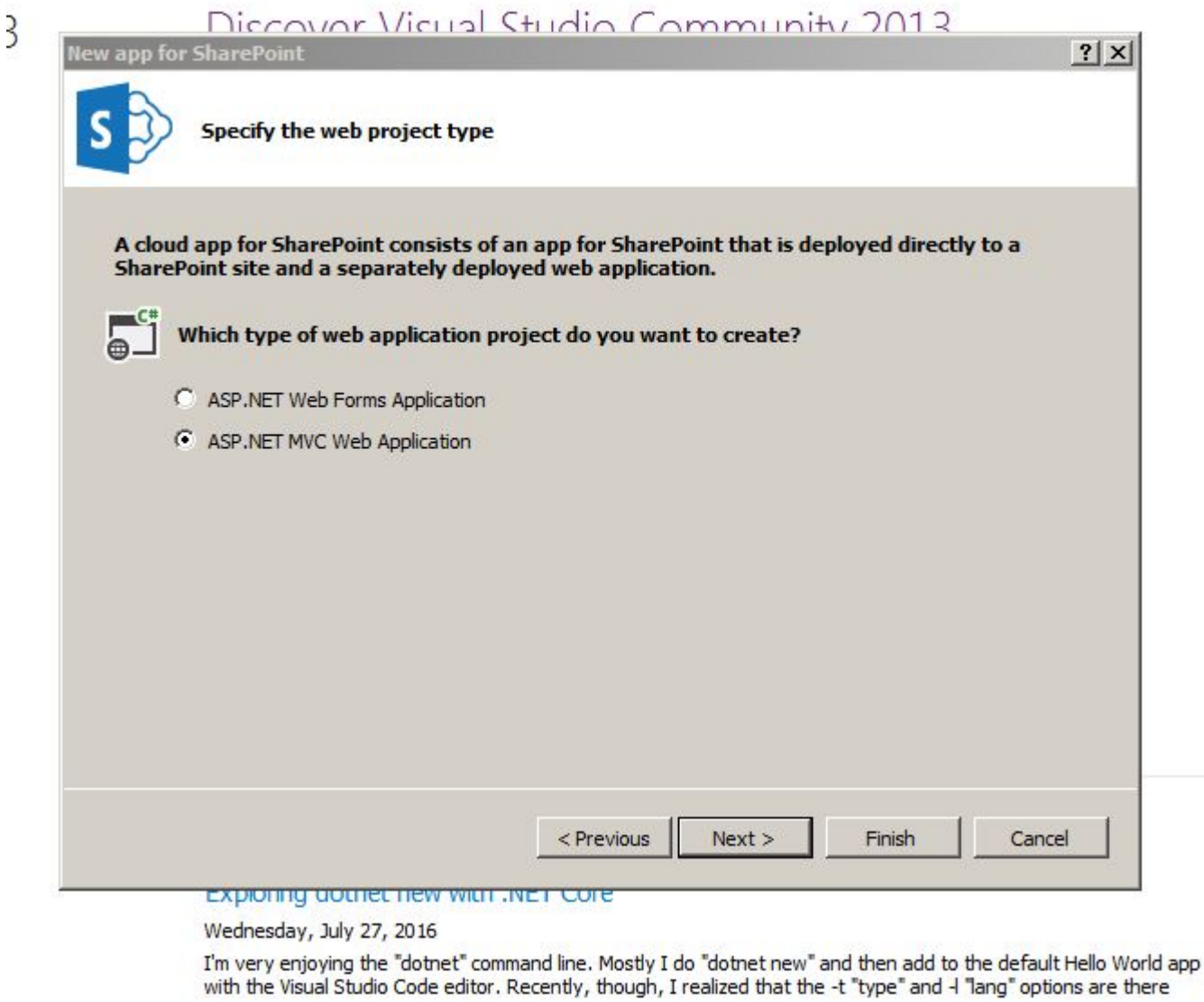


3. のでしたデベロッパーサイトのURLをし、[プロバイダのホスト]をします



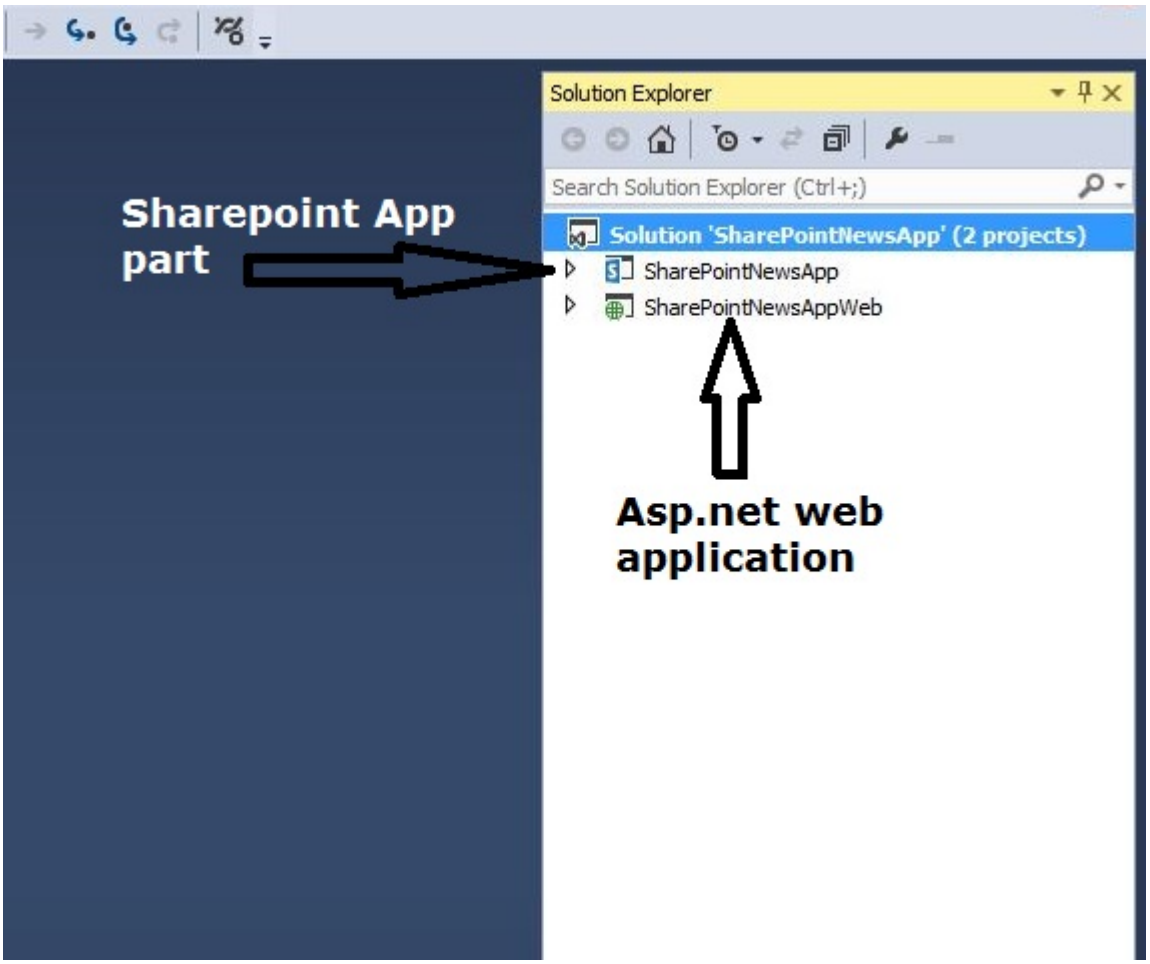
4. ポップアップがき、ログインにされます

5. のステップでは、アプリケーションのタイプとして、MVCまたはWebformをします。ここでMCVをしています



6. どのようにアドインをするのですかで、Windows Azure Access Control Serviceををし、をクリックします。

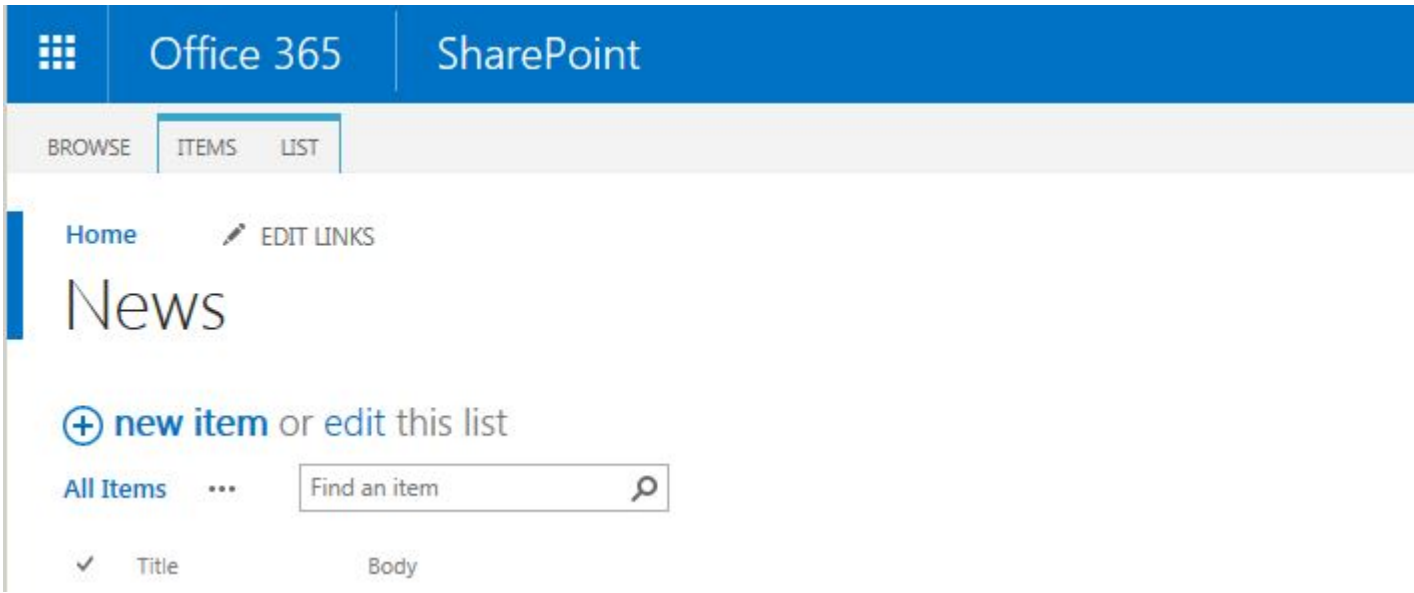
7. ソリューションエクスプローラでは、2つのプロジェクトがされていることがわかります。1つはSharePoint app-partで、もう1つはasp.net web appです



コーディングをします

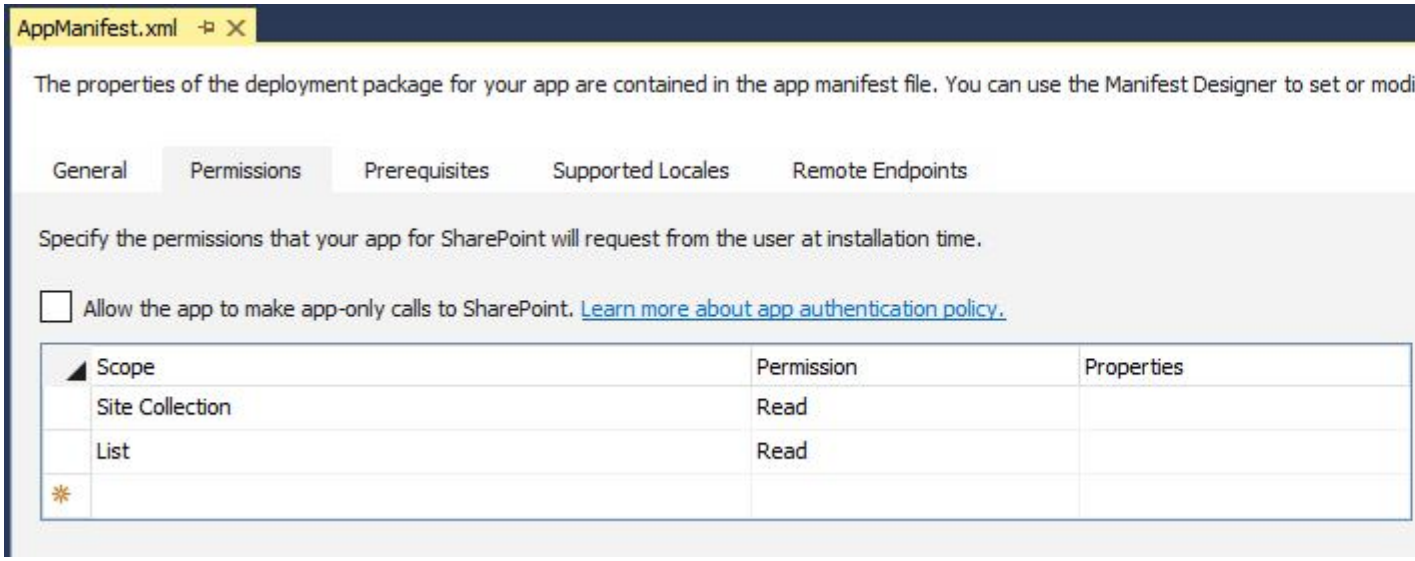
ここではなニュースアプリのをっています

1. SharePointデベロッパーサイトをき、ニュースをするためのリストをする
2. カスタムリストをし、さらに3つのをしますBody、Summery、ThumbnailImageUrl



3. たちのSharePointアプリケーションにり、AppManifest.xmlファイルをき、タブをクリック

し、サイトコレクションにみりをえてします。



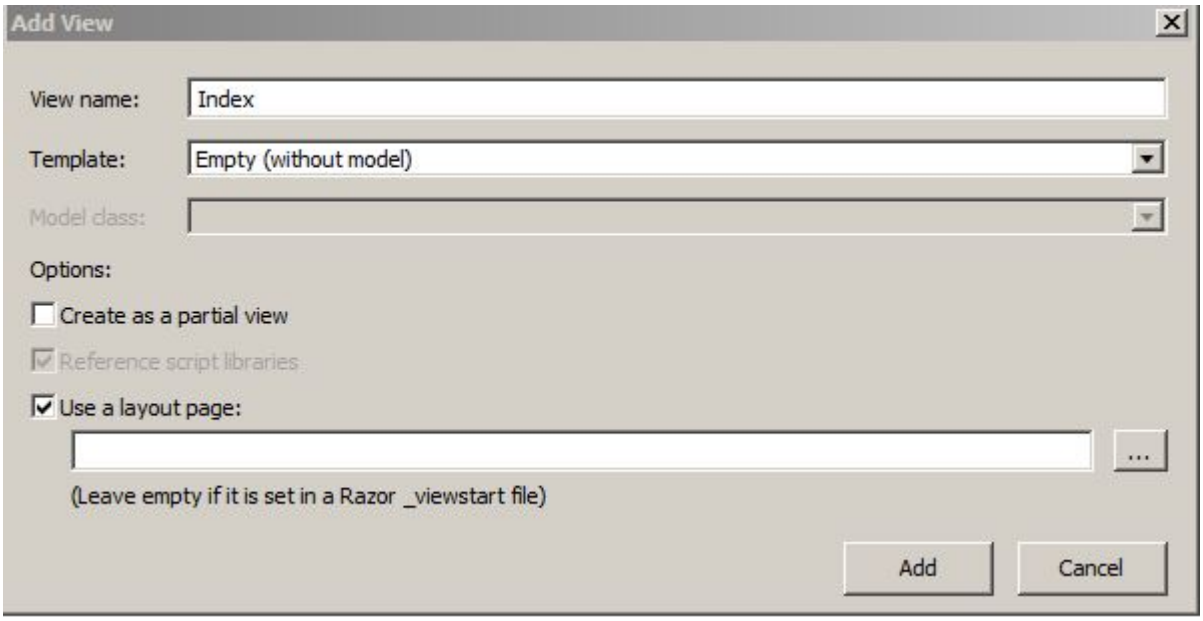
4. WebアプリケーションからHomeControllerをきます。の、MVCアプリケーションです。Webformアプリケーションをするは、default.aspx.csページにコードをれるがあります

5. は、リストからのニュースをするためのコードスニペットです。インデックスページの

```
[SharePointContextFilter]
public ActionResult Index()
{
    User spUser = null;

    var spContext = SharePointContextProvider.Current.GetSharePointContext(HttpContext);
    List<NewsList> newsList = new List<NewsList>();
    using (var clientContext = spContext.CreateUserClientContextForSPHost())
    {
        if (clientContext != null)
        {
            spUser = clientContext.Web.CurrentUser;

            clientContext.Load(spUser, user => user.Title);

            clientContext.ExecuteQuery();

            ViewBag.UserName = spUser.Title;

            List lst = clientContext.Web.Lists.GetByTitle("News");
            CamlQuery queryNews = CamlQuery.CreateAllItemsQuery(10);
            ListItemCollection newsItems = lst.GetItems(queryNews);
            clientContext.Load(newsItems, includes => includes.Include(i => i.Id, i =>
i.DisplayName, i => i["ThumbnailImageUrl"], i => i["Summery"]));

            clientContext.ExecuteQuery();

            if (newsItems != null)
            {
                foreach (var lstProductItem in newsItems)
                {
                    newsList.Add(
                        new NewsList
                        {
                            Id = Convert.ToInt32(lstProductItem.Id.ToString()),
```

```
                                Title = lstProductItem.DisplayName.ToString(),
                                Summery = lstProductItem["Summery"].ToString(),
                                Thumbnail = lstProductItem["ThumbnailImageUrl"].ToString()
                            });
                    }
                }
            }

        return View(newsList);
}
```

6. **Index**をクリックし、 **Add View**をクリックします。に、[]をクリックします



7. **Index.cshtml**ファイルをきます。Views **> Home directory**から

8. は、index.cshtmlファイルのコードスニペットです

```
@model List<SharePointNewsAppWeb.Models.NewsList>
 @{
 ViewBag.Title = "My News - browse latest news";
}
 <br />
 @foreach (var item in Model)
 {
 <div class="row panel panel-default">
  <div class="col-xs-3">
    <a href="/home/aticle?ArticleId=@item.Id">
        <img class="img-responsive" style="max-height:200px;max-width:100%;"
src="@item.Thumbnail" alt="@item.Title" />
    </a>
</div>
<div class="col-xs-9 panel-default">
    <div class="panel-heading">
        <h4><a href="/home/aticle?ArticleId=@item.Id">@item.Title.ToUpper()</a></h4>
    </div>
    <div class="panel-body">
        <p>@item.Summery</p>
```

```
    </div>
</div>
```

9. ソリューションのModelフォルダをクリックし、CSクラスファイルをします。のモデルクラスをする

```
 using System;
 using System.Collections.Generic;
 using System.Linq;
 using System.Web;

 namespace SharePointNewsAppWeb.Models
{
  public class NewsApp
   {

   }
 public class NewsList
  {

public int Id { get; set; }

public string Title { get; set; }

public string Summery { get; set; }

public string Thumbnail { get; set; }
 }
 public class FullArticle
 {

  public int Id { get; set; }

  public string Title { get; set; }

  public string Body { get; set; }

 }
}
```

10. アドインをしてするには、F5キーをします。ローカルホストをするようにめるセキュリティウィンドウがされたは、[はい]をします。

そして、のAppがです

なページをする

たちはすでにすべてのニュースをするのページをしました。このページにはながされます。

1. HomeControllerにもう1つのアクションメソッドをする

```
[SharePointContextFilter]
public ActionResult Aticle(int ArticleId)
{
    User spUser = null;
```

```
    var spContext = SharePointContextProvider.Current.GetSharePointContext(HttpContext);
    FullArticle article = new FullArticle();
    using (var clientContext = spContext.CreateUserClientContextForSPHost())
    {
        if (clientContext != null)
        {
            spUser = clientContext.Web.CurrentUser;

            clientContext.Load(spUser, user => user.Title);

            clientContext.ExecuteQuery();

            ViewBag.UserName = spUser.Title;

            List lst = clientContext.Web.Lists.GetByTitle("News");
            CamlQuery queryNews = new CamlQuery();
            queryNews.ViewXml = @"<View><Query><Where><Eq><FieldRef Name='ID'/>" +
"<Value Type='Number'>" + ArticleId + "</Value></Eq></Where></Query>" +
                "<ViewFields><FieldRef Name='ID'/><FieldRef Name='Title'/><FieldRef
Name='Body'/></ViewFields></View>";//
            ListItemCollection newsItems = lst.GetItems(queryNews);
            clientContext.Load(newsItems, includes => includes.Include(i => i.Id, i =>
i.DisplayName, i => i["Body"]));

            clientContext.ExecuteQuery();

            if (newsItems != null)
            {
                foreach (var lstProductItem in newsItems)
                {
                    article.Id = Convert.ToInt32(lstProductItem.Id.ToString());
                    article.Title = lstProductItem.DisplayName.ToString();
                    article.Body = lstProductItem["Body"].ToString();
                }
            }
        }
    }
    return View(article);
}
```

2. アクションをクリックし、じのアクションメソッドをつビューをします。の、ビューは **Aticle**とばれます

```
  @model SharePointNewsAppWeb.Models.FullArticle

@{
 ViewBag.Title = "Aticle";
}

<br />
<div class="panel panel-default">
 <div class="panel-heading"><a style="font-size:20px;" href="/"><i   class="glyphicon
glyphicon-chevron-left"></i> <i class="glyphicon glyphicon-home"></i>      </a></div>
 <div class="panel-heading"><h1 class="h2">@Model.Title.ToUpper()</h1></div>
 <div class="panel-body">@Html.Raw(@Model.Body)</div>
</div>
```

これはニュースのをすなページのコードです

オンラインでプロバイダのアプリケーションをするをむ
https://riptutorial.com/ja/sharepoint/topic/6301/プロバイダのアプリケーションをする

8: マネージドクライアントサイドオブジェクトモデルCSOMの

- ほとんどのはMSDNのものです。
- クライアントオブジェクトモデルをする.NETマネージクライアントアプリケーションをするには、2つのクライアントライブラリDLL、Microsoft.SharePoint.Client.dllおよびMicrosoft.SharePoint.Client.Runtime.dllへのをするがあります。 ProgramFiles\ Common Files \ Microsoft Shared \ webサーバーエクステンション\ 16 \ ISAPIフォルダまたはSharePointサーバーでつけることができます。
- またはMicrosoft.SharePointOnline.CSOM NuGetパッケージをインストールします。これは、SP O365とに「オン」でします。
- ほとんどのプロパティはvalueプロパティです。これらのプロパティにアクセスするに、clientContext.LoadおよびclientContext.ExecuteQueryをにびすがあります。ここでの のプロパティにアクセスするのコールロードとExecuteQuery

Examples

こんにちはサイトタイトルを

SharePointのすべてのバージョンは、サイトSPSiteSSOMまたはサイトCSOMとWebSPWeb SSOMまたはWebCSOMをベースにしています。サイトには、にされるメタデータとがまれていますが、UIにレンダリングされません。 Webは、サイトにアクセスするユーザーにUIをするなです。すべてのサイトには、ドキュメントライブラリなどのやメタデータをするルートWebがあります。このでは、パス`sites`にあるサーバー`MyServer`あるWebをフェッチするためのなびしをし`sites`。

```
using System;
using Microsoft.SharePoint.Client;

namespace Microsoft.SDK.SharePointServices.Samples
{
    class RetrieveWebsite
    {
        static void Main()
        {
            // This is the URL of the target web we are interested in.
            string siteUrl = "http://MyServer/sites/MySiteCollection";
            // The client context is allows us to queue up requests for the server
            // Note that the context can only ask questions about the site it is created for
            using (ClientContext clientContext = new ClientContext(siteUrl))
            {
                // To make it easier to read the code, pull the target web
                // context off of the client context and store in a variable
                Web oWebsite = clientContext.Web;
                // Tell the client context we want to request information about the
                // Web from the server
                clientContext.Load(oWebsite);
```

```
                // After we are done creating the batch of information we need from the sever,
                // request the data from SharePoint
                clientContext.ExecuteQuery();
                // Print the results of the query
                Console.WriteLine("Title: {0} Description: {1}", oWebsite.Title,
oWebsite.Description);
            }
        }
    }
}
```

ウェブ。 Webサイトのプロパティの

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;
clientContext.Load(oWebsite);
clientContext.ExecuteQuery();
Console.WriteLine("Title: {0} Description: {1}", oWebsite.Title, oWebsite.Description);
```

ウェブ。 Webサイトのされたプロパティのみをする

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;
clientContext.Load(
                oWebsite,
                website => website.Title,
                website => website.Created);
clientContext.ExecuteQuery();
Console.WriteLine("Title: {0} Created: {1}", oWebsite.Title, oWebsite.Created);
```

ウェブ。 Webサイトのタイトルとの

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = context.Web;
oWebsite.Title = "Updated Web Site";
oWebsite.Description = "This is an updated Web site.";
oWebsite.Update();
clientContext.ExecuteQuery();
```

ウェブ。 Webサイトの

```
string siteUrl = "http://MyServer/sites/MySiteCollection";
string blogDescription = "A new blog Web site.";
int blogLanguage = 1033;
string blogTitle = "Blog Web Site";
string blogUrl = "blogwebsite";
bool blogPermissions = false;
string webTemplate = "BLOG#0";

ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;

WebCreationInformation webCreateInfo = new WebCreationInformation();
```

```
webCreateInfo.Description = blogDescription;
webCreateInfo.Language = blogLanguage;
webCreateInfo.Title = blogTitle;
webCreateInfo.Url = blogUrl;
webCreateInfo.UseSamePermissionsAsParentSite = blogPermissions;
webCreateInfo.WebTemplate = webTemplate;

Web oNewWebsite = oWebsite.Webs.Add(webCreateInfo);

clientContext.Load(
    oNewWebsite,
    website => website.ServerRelativeUrl,
    website => website.Created);

clientContext.ExecuteQuery();

Console.WriteLine("Server-relative Url: {0} Created: {1}", oNewWebsite.ServerRelativeUrl,
oNewWebsite.Created);
```

リスト。 Webサイトのすべてのリストのすべてのプロパティをする

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;
ListCollection collList = oWebsite.Lists;

clientContext.Load(collList);

clientContext.ExecuteQuery();

foreach (List oList in collList)
{
    Console.WriteLine("Title: {0} Created: {1}", oList.Title, oList.Created.ToString());
}
```

リスト。されたリストのプロパティのみをする

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;
ListCollection collList = oWebsite.Lists;

clientContext.Load(
    collList,
    lists => lists.Include(
        list => list.Title,
        list => list.Id));

clientContext.ExecuteQuery();

foreach (List oList in collList)
{
    Console.WriteLine("Title: {0} ID: {1}", oList.Title, oList.Id.ToString("D"));
}
```

リスト。されたリストをコレクションにする

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;
ListCollection collList = oWebsite.Lists;

IEnumerable<List> resultCollection = clientContext.LoadQuery(
    collList.Include(
        list=>list.Title,
        list=>list.Id));

clientContext.ExecuteQuery();

foreach (List oList in resultCollection)
{
    Console.WriteLine("Title: {0} ID: {1}", oList.Title, oList.Id.ToString("D"));
}
```

リスト。 Webサイトからのリストフィールドの

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;
ListCollection collList = oWebsite.Lists;

IEnumerable<SP.List> listInfo = clientContext.LoadQuery(
    collList.Include(
        list => list.Title,
        list => list.Fields.Include(
            field => field.Title,
            field => field.InternalName)));

 clientContext.ExecuteQuery();

foreach (SP.List oList in listInfo)
{
    FieldCollection collField = oList.Fields;

    foreach (SP.Field oField in collField)
    {
        Regex regEx = new Regex("name", RegexOptions.IgnoreCase);

        if (regEx.IsMatch(oField.InternalName))
        {
            Console.WriteLine("List: {0} \n\t Field Title: {1} \n\t Field Internal Name: {2}",

                oList.Title, oField.Title, oField.InternalName);
        }
    }
}
```

リスト。リストのと

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;

ListCreationInformation listCreationInfo = new ListCreationInformation();
listCreationInfo.Title = "My Announcements List";
listCreationInfo.TemplateType = (int)ListTemplateType.Announcements;
```

```
List oList = oWebsite.Lists.Add(listCreationInfo);

clientContext.ExecuteQuery();
```

リスト。フィールドをリストにする

```
ClientContext clientContext = new ClientContext(siteUrl);

SP.List oList = clientContext.Web.Lists.GetByTitle("Announcements");

SP.Field oField = oList.Fields.AddFieldAsXml("<Field DisplayName='MyField' Type='Number' />",

    true, AddFieldOptions.DefaultValue);

SP.FieldNumber fieldNumber = clientContext.CastTo<FieldNumber>(oField);
fieldNumber.MaximumValue = 100;
fieldNumber.MinimumValue = 35;

fieldNumber.Update();

clientContext.ExecuteQuery();
```

リスト。リストの

```
ClientContext clientContext = new ClientContext(siteUrl);
Web oWebsite = clientContext.Web;

List oList = oWebsite.Lists.GetByTitle("My Announcements List");

oList.DeleteObject();

clientContext.ExecuteQuery();
```

。リストからアイテムをする

```
ClientContext clientContext = new ClientContext(siteUrl);
SP.List oList = clientContext.Web.Lists.GetByTitle("Announcements");

CamlQuery camlQuery = new CamlQuery();
camlQuery.ViewXml = "<View><Query><Where><Geq><FieldRef Name='ID'/>" +
    "<Value Type='Number'>10</Value></Geq></Where></Query><RowLimit>100</RowLimit></View>";
ListItemCollection collListItem = oList.GetItems(camlQuery);

clientContext.Load(collListItem);

clientContext.ExecuteQuery();

foreach (ListItem oListItem in collListItem)
{
    Console.WriteLine("ID: {0} \nTitle: {1} \nBody: {2}", oListItem.Id, oListItem["Title"],
oListItem["Body"]);
}
```

。アイテムの**Include**メソッドを

このは、サ―バ―からアイテムをすると、リストのよりいプロパティをするをしています。デフォルトでは、サ―バ―はオブジェクトをすのデ―タのみをします。サ―バ―からをするのは、びしのです。

```
ClientContext clientContext = new ClientContext(siteUrl);
List oList = clientContext.Web.Lists.GetByTitle("Announcements");

CamlQuery camlQuery = new CamlQuery();
camlQuery.ViewXml = "<View><RowLimit>100</RowLimit></View>";

ListItemCollection collListItem = oList.GetItems(camlQuery);

// The first line of this request indicates the list item collection to load from the server
// The second line uses a lambda to request that from the server
// also include additional properties in the response
// The third though fifth lines are the properties being requested from the server
clientContext.Load(collListItem,
     items => items.Include(
        item => item.Id,
        item => item.DisplayName,
        item => item.HasUniqueRoleAssignments));

clientContext.ExecuteQuery();

foreach (ListItem oListItem in collListItem)
{
    Console.WriteLine("ID: {0} \nDisplay name: {1} \nUnique role assignments: {2}",
        oListItem.Id, oListItem.DisplayName, oListItem.HasUniqueRoleAssignments);
}
```

。されたのからのフィ―ルドをする

```
ClientContext clientContext = new ClientContext(siteUrl);
SP.List oList = clientContext.Web.Lists.GetByTitle("Announcements");

CamlQuery camlQuery = new CamlQuery();
ListItemCollection collListItem = oList.GetItems(camlQuery);

clientContext.Load(
    collListItem,
    items => items.Take(5).Include(
    item => item["Title"],
    item => item["Body"]));

clientContext.ExecuteQuery();

foreach (ListItem oListItem in collListItem)
{
    Console.WriteLine("Title: {0} \nBody: {1}\n", oListItem["Title"], oListItem["Body"]);
}
```

。 Webサイトのすべてのリストからアイテムをする

```
ClientContext clientContext = new ClientContext(siteUrl);
ListCollection collList = clientContext.Web.Lists;
```

```
clientContext.Load(
    collList,
    lists => lists.Where(
        list => list.Hidden == false).Include(
        list => list.Title,
        list => list.Items.Take(10)));

clientContext.ExecuteQuery();

foreach (SP.List oList in clientContext.Web.Lists)
{
    string listTitle = oList.Title;
    int itemCount = oList.Items.Count;

    Console.WriteLine("List {0} returned with {1} items", listTitle, itemCount);
}
```

。リストアイテムのコレ クションをしてアイテムをする

```
ClientContext clientContext = new ClientContext(siteUrl);
SP.List oList = clientContext.Web.Lists.GetByTitle("Announcements");

ListItemCollectionPosition itemPosition = null;

while (true)
{
    CamlQuery camlQuery = new CamlQuery();

    camlQuery.ListItemCollectionPosition = itemPosition;

    camlQuery.ViewXml = "<View><ViewFields><FieldRef Name='ID'/>" +
        "<FieldRef Name='Title'/><FieldRef Name='Body'/>" +
        "</ViewFields><RowLimit>5</RowLimit></View>";

    ListItemCollection collListItem = oList.GetItems(camlQuery);

    clientContext.Load(collListItem);

    clientContext.ExecuteQuery();

    itemPosition = collListItem.ListItemCollectionPosition;

    foreach (ListItem oListItem in collListItem)
    {
        Console.WriteLine("Title: {0}: \nBody: {1}", oListItem["Title"], oListItem["Body"]);
    }

    if (itemPosition == null)
    {
        break;
    }

    Console.WriteLine("\n" + itemPosition.PagingInfo + "\n");
}
```

。リストの

しいリストアイテムをするとき、そのフィールドはにたをしてできます。これらのフィールドは

オンザフライではされず、リストのスキ―マによってされることにしてください。これらのフィ―ルドまたははサ―バ―にしていなければなりません。そうでない、createはします。すべてのリストアイテムにはタイトルフィ―ルドがあります。のリストには、アイテムがリストにされるにするのあるフィ―ルドがあるがあります。

このでは、リストはAnnouncementsテンプレ―トをしています。タイトルフィ―ルドにえて、リストには、アナウンスのをリストにするBodyフィ―ルドがまれています。

```
ClientContext clientContext = new ClientContext(siteUrl);
List oList = clientContext.Web.Lists.GetByTitle("Announcements");

ListItemCreationInformation itemCreateInfo = new ListItemCreationInformation();
ListItem oListItem = oList.AddItem(itemCreateInfo);
oListItem["Title"] = "My New Item!";
oListItem["Body"] = "Hello World!";

oListItem.Update();

clientContext.ExecuteQuery();
```

。リストの

```
ClientContext clientContext = new ClientContext(siteUrl);
SP.List oList = clientContext.Web.Lists.GetByTitle("Announcements");
ListItem oListItem = oList.Items.GetById(3);

oListItem["Title"] = "My Updated Title.";

oListItem.Update();

clientContext.ExecuteQuery();
```

。リストをする

```
ClientContext clientContext = new ClientContext(siteUrl);
SP.List oList = clientContext.Web.Lists.GetByTitle("Announcements");
ListItem oListItem = oList.GetItemById(2);

oListItem.DeleteObject();

clientContext.ExecuteQuery();
```

グル―プ。 **SharePoint** グル―プからすべてのユ―ザ―をする

```
ClientContext clientContext = new ClientContext("http://MyServer/sites/MySiteCollection");
GroupCollection collGroup = clientContext.Web.SiteGroups;
Group oGroup = collGroup.GetById(7);
UserCollection collUser = oGroup.Users;

clientContext.Load(collUser);

clientContext.ExecuteQuery();
```

```
foreach (User oUser in collUser)
{
    Console.WriteLine("User: {0}  ID: {1} Email: {2} Login Name: {3}",
        oUser.Title, oUser.Id, oUser.Email, oUser.LoginName);
}
```

グループ。ユーザーののプロパティの

```
ClientContext clientContext = new ClientContext("http://MyServer/sites/MySiteCollection");
GroupCollection collGroup = clientContext.Web.SiteGroups;
Group oGroup = collGroup.GetById(7);
UserCollection collUser = oGroup.Users;

clientContext.Load(collUser,
    users => users.Include(
        user => user.Title,
        user => user.LoginName,
        user => user.Email));

clientContext.ExecuteQuery();

foreach (User oUser in collUser)
{
    Console.WriteLine("User: {0} Login name: {1} Email: {2}",
        oUser.Title, oUser.LoginName, oUser.Email);
}
```

グループ。サイトコレクションのすべてのグループのすべてのユーザーをする

```
ClientContext clientContext = new ClientContext("http://MyServer/sites/MySiteCollection");
GroupCollection collGroup = clientContext.Web.SiteGroups;

clientContext.Load(collGroup);

clientContext.Load(collGroup,
    groups => groups.Include(
        group => group.Users));

clientContext.ExecuteQuery();

foreach (Group oGroup in collGroup)
{
    UserCollection collUser = oGroup.Users;

    foreach (User oUser in collUser)
    {
        Console.WriteLine("Group ID: {0} Group Title: {1} User: {2} Login Name: {3}",
            oGroup.Id, oGroup.Title, oUser.Title, oUser.LoginName);
    }
}
```

グループ。 **SharePoint** グループにユーザーをする

```
ClientContext clientContext = new ClientContext("http://MyServer/sites/MySiteCollection ");
```

```
GroupCollection collGroup = clientContext.Web.SiteGroups;
Group oGroup = collGroup.GetById(6);

UserCreationInformation userCreationInfo = new UserCreationInformation();
userCreationInfo.Email = "alias@somewhere.com";
userCreationInfo.LoginName = @"DOMAIN\alias";
userCreationInfo.Title = "John";

User oUser = oGroup.Users.Add(userCreationInfo);

clientContext.ExecuteQuery();
```

。 ロールの

```
ClientContext oClientContext = new ClientContext("http://MyServer/sites/MySiteCollection");

Web oWebsite = clientContext.Web;

BasePermissions permissions = new BasePermissions();
permissions.Set(PermissionKind.CreateAlerts);
permissions.Set(PermissionKind.ManageAlerts);

RoleDefinitionCreationInformation roleCreationInfo = new RoleDefinitionCreationInformation();

roleCreationInfo.BasePermissions = permissions;
roleCreationInfo.Description = "A new role with create and manage alerts permission";
roleCreationInfo.Name = "Create and Manage Alerts";
roleCreationInfo.Order = 4;

RoleDefinition oRoleDefinition = oWebsite.RoleDefinitions.Add(roleCreationInfo);

clientContext.ExecuteQuery();

Console.WriteLine("{0} role created.", oRoleDefinition.Name);
```

。 Webサイトのロールにユーザーをりてる

```
ClientContext oClientContext = new
ClientContext("http://MyServer/sites/MySiteCollection/MyWebSite");
Web oWebsite = clientContext.Web;

Principal oUser = oWebsite.SiteUsers.GetByLoginName(@"DOMAIN\alias");

RoleDefinition oRoleDefinition = oWebsite.RoleDefinitions.GetByName("Create and Manage
Alerts");
RoleDefinitionBindingCollection collRoleDefinitionBinding = new
RoleDefinitionBindingCollection(clientContext);
collRoleDefinitionBinding.Add(oRoleDefinition);

RoleAssignment oRoleAssignment = oWebsite.RoleAssignments.Add(oUser,
collRoleDefinitionBinding);

clientContext.Load(oUser,
    user => user.Title);

clientContext.Load(oRoleDefinition,
    role => role.Name);
```

```
clientContext.ExecuteQuery();

Console.WriteLine("{0} added with {1} role.", oUser.Title, oRoleDefinition.Name);
```

。 SharePointグループのとロールへのグループの

```
ClientContext oClientContext = new
ClientContext("http://MyServer/sites/MySiteCollection/MyWebSite");
Web oWebsite = clientContext.Web;

GroupCreationInformation groupCreationInfo = new GroupCreationInformation();
groupCreationInfo.Title = "My New Group";
groupCreationInfo.Description = "Description of new group.";
Group oGroup = oWebsite.SiteGroups.Add(groupCreationInfo);

RoleDefinitionBindingCollection collRoleDefinitionBinding = new
RoleDefinitionBindingCollection(clientContext);

RoleDefinition oRoleDefinition = oWebsite.RoleDefinitions.GetByType(RoleType.Contributor);

collRoleDefinitionBinding.Add(oRoleDefinition);

oWebsite.RoleAssignments.Add(oGroup, collRoleDefinitionBinding);

clientContext.Load(oGroup,
    group => group.Title);

clientContext.Load(oRoleDefinition,
    role => role.Name);

clientContext.ExecuteQuery();

Console.WriteLine("{0} created and assigned {1} role.", oGroup.Title, oRoleDefinition.Name);
}
```

パーミッション。リストのセキュリティをる

```
string siteUrl = "http://MyServer/sites/MySiteCollection";
ClientContext oContext = new ClientContext(siteUrl);
SP.List oList = oContext.Web.Lists.GetByTitle("Announcements");

oList.BreakRoleInheritance(true, false);

oContext.ExecuteQuery();
```

パーミッション。ドキュメントのセキュリティをり、ユーザーをリーダーとしてする

```
ClientContext clientContext = new ClientContext(siteUrl);
SP.List oList = clientContext.Web.Lists.GetByTitle("MyList");

int itemId = 3;
ListItem oListItem = oList.Items.GetById(itemId);
```

```
oListItem.BreakRoleInheritance(false);

User oUser = clientContext.Web.SiteUsers.GetByLoginName(@"DOMAIN\alias");

RoleDefinitionBindingCollection collRoleDefinitionBinding = new
RoleDefinitionBindingCollection(clientContext);

collRoleDefinitionBinding.Add(clientContext.Web.RoleDefinitions.GetByType(RoleType.Reader));

oListItem.RoleAssignments.Add(oUser, collRoleDefinitionBinding);

clientContext.ExecuteQuery();
```

パーミッション。ドキュメントのセキュリティをり、ユーザーのアクセスをする

```
ClientContext clientContext = new ClientContext(siteUrl);
SP.List oList = clientContext.Web.Lists.GetByTitle("MyList");

int itemId = 2;
ListItem oListItem = oList.Items.GetById(itemId);

oListItem.BreakRoleInheritance(true);

User oUser = clientContext.Web.SiteUsers.GetByLoginName(@"DOMAIN\alias");
oListItem.RoleAssignments.GetByPrincipal(oUser).DeleteObject();

RoleDefinitionBindingCollection collRollDefinitionBinding = new
RoleDefinitionBindingCollection(clientContext);

collRollDefinitionBinding.Add(clientContext.Web.RoleDefinitions.GetByType(RoleType.Reader));

oListItem.RoleAssignments.Add(oUser, collRollDefinitionBinding);

clientContext.ExecuteQuery();
```

カスタムアクション。リストアイテムのユーザーカスタムアクションの

```
string urlWebsite = "http://MyServer/sites/MySiteCollection";
ClientContext clientContext = new ClientContext(urlWebsite);
Web oWebsite = clientContext.Web;

List oList = oWebsite.Lists.GetByTitle("My List");
UserCustomActionCollection collUserCustomAction = oList.UserCustomActions;

UserCustomAction oUserCustomAction = collUserCustomAction.Add();
oUserCustomAction.Location = "EditControlBlock";
oUserCustomAction.Sequence = 100;
oUserCustomAction.Title = "My First User Custom Action";
oUserCustomAction.Url = urlWebsite + @"/_layouts/MyPage.aspx";
oUserCustomAction.Update();

clientContext.Load(oList,
    list => list.UserCustomActions);

clientContext.ExecuteQuery();
```

カスタムアクション。ユーザーカスタムアクションの

```
string urlWebsite = "http://MyServer/sites/SiteCollection";
ClientContext clientContext = new ClientContext(urlWebsite);
Web oWebsite = clientContext.Web;

List oList = oWebsite.Lists.GetByTitle("My List");
UserCustomActionCollection collUserCustomAction = oList.UserCustomActions;

clientContext.Load(collUserCustomAction,
    userCustomActions => userCustomActions.Include(
        userCustomAction => userCustomAction.Title));

clientContext.ExecuteQuery();

foreach (UserCustomAction oUserCustomAction in collUserCustomAction)
{
    if (oUserCustomAction.Title == "My First User Custom Action")
    {
        oUserCustomAction.ImageUrl = "http://MyServer/_layouts/images/MyIcon.png";
        oUserCustomAction.Update();

        clientContext.ExecuteQuery();
    }
}
```

カスタムアクション。 Webサイトのサイトアクションにユーザーカスタムアクションをする

```
string urlWebsite = "http://MyServer/sites/MySiteCollection";
ClientContext clientContext = new ClientContext(urlWebsite);

Web oWebsite = clientContext.Web;
UserCustomActionCollection collUserCustomAction = oWebsite.UserCustomActions;

UserCustomAction oUserCustomAction = collUserCustomAction.Add();

oUserCustomAction.Location = "Microsoft.SharePoint.StandardMenu";
oUserCustomAction.Group = "SiteActions";
oUserCustomAction.Sequence = 101;
oUserCustomAction.Title = "Website User Custom Action";
oUserCustomAction.Description = "This description appears on the Site Actions menu.";
oUserCustomAction.Url = urlWebsite + @"/_layouts/MyPage.aspx";

oUserCustomAction.Update();

clientContext.Load(oWebsite,
    webSite => webSite.UserCustomActions);

clientContext.ExecuteQuery();
```

Webパーツ。 Webパーツのタイトルの

```
ClientContext oClientContext = new ClientContext("http://MyServer/sites/MySiteCollection");
File oFile = oClientContext.Web.GetFileByServerRelativeUrl("Default.aspx");
```

```
LimitedWebPartManager limitedWebPartManager =
oFile.GetLimitedWebPartManager(PersonalizationScope.Shared);

oClientContext.Load(limitedWebPartManager.WebParts,
    wps => wps.Include(
    wp => wp.WebPart.Title));

oClientContext.ExecuteQuery();

if (limitedWebPartManager.WebParts.Count == 0)
{
    throw new Exception("No Web Parts on this page.");
}

WebPartDefinition oWebPartDefinition = limitedWebPartManager.WebParts[1];
WebPart oWebPart = oWebPartDefinition.WebPart;
oWebPart.Title = "My New Web Part Title";

oWebPartDefinition.SaveWebPartChanges();

oClientContext.ExecuteQuery();
```

Webパーツ。ページにWebパーツをする

```
ClientContext oClientContext = new ClientContext("http://MyServer/sites/MySiteCollection");
File oFile = oClientContext.Web.GetFileByServerRelativeUrl("Default.aspx");
LimitedWebPartManager limitedWebPartManager =
oFile.GetLimitedWebPartManager(PersonalizationScope.Shared);

string xmlWebPart = "<?xml version=\"1.0\" encoding=\"utf-8\"?>" +
    "<WebPart xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"" +
    " xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"" +
    " xmlns=\"http://schemas.microsoft.com/WebPart/v2\">" +
    "<Title>My Web Part</Title><FrameType>Default</FrameType>" +
    "<Description>Use for formatted text, tables, and images.</Description>" +
    "<IsIncluded>true</IsIncluded><ZoneID></ZoneID><PartOrder>0</PartOrder>" +
    "<FrameState>Normal</FrameState><Height /><Width /><AllowRemove>true</AllowRemove>" +
    "<AllowZoneChange>true</AllowZoneChange><AllowMinimize>true</AllowMinimize>" +
    "<AllowConnect>true</AllowConnect><AllowEdit>true</AllowEdit>" +
    "<AllowHide>true</AllowHide><IsVisible>true</IsVisible><DetailLink /><HelpLink />" +
    "<HelpMode>Modeless</HelpMode><Dir>Default</Dir><PartImageSmall />" +
    "<MissingAssembly>Cannot import this Web Part.</MissingAssembly>" +
    "<PartImageLarge>/_layouts/images/mscontl.gif</PartImageLarge><IsIncludedFilter />" +
    "<Assembly>Microsoft.SharePoint, Version=13.0.0.0, Culture=neutral, " +
    "PublicKeyToken=94de0004b6e3fcc5</Assembly>" +
    "<TypeName>Microsoft.SharePoint.WebPartPages.ContentEditorWebPart</TypeName>" +
    "<ContentLink xmlns=\"http://schemas.microsoft.com/WebPart/v2/ContentEditor\" />" +
    "<Content xmlns=\"http://schemas.microsoft.com/WebPart/v2/ContentEditor\">" +
    "<![CDATA[This is a first paragraph!<DIV> </DIV>And this is a second
paragraph.]]></Content>" +
    "<PartStorage xmlns=\"http://schemas.microsoft.com/WebPart/v2/ContentEditor\"
/></WebPart>";

WebPartDefinition oWebPartDefinition = limitedWebPartManager.ImportWebPart(xmlWebPart);

limitedWebPartManager.AddWebPart(oWebPartDefinition.WebPart, "Left", 1);

oClientContext.ExecuteQuery();
```

Webパーツ。ページからのWebパーツの

```
ClientContext oClientContext = new ClientContext("http://MyServer/sites/MySiteCollection");
File oFile =
oClientContext.Web.GetFileByServerRelativeUrl("/sites/MySiteCollection/SitePages/Home.aspx ");
LimitedWebPartManager limitedWebPartManager =
oFile.GetLimitedWebPartManager(PersonalizationScope.Shared);

oClientContext.Load(limitedWebPartManager.WebParts);

oClientContext.ExecuteQuery();

if (limitedWebPartManager.WebParts.Count == 0)
{
    throw new Exception("No Web Parts to delete.");
}

WebPartDefinition webPartDefinition = limitedWebPartManager.WebParts[0];

webPartDefinition.DeleteWebPart();

oClientContext.ExecuteQuery();
```

コンテキスト。コードのされたにキャッシュをする

サーバーのコードはされたでできますが、なセキュリティのから、クライアントのコードでをさせるのはありません。または、のユーザーまたはサービスアカウントのアクセスをエミュレートするためのをすることもできます。

をするには、 `CredentialCache`オブジェクトをしてし、それを`ClientContext`オブジェクトの`Credentials`プロパティにりてます。

のでは、アプリケーションプールアカウントをエミュレートし、NTLMをするオンプレミスSharePoint 2013をしています。

```
using System.Net;
using Microsoft.SharePoint.Client;

using (ClientContext ctx = new ClientContext("https://onpremises.local/sites/demo/"))
{
    // need the web object
    ctx.Load(ctx.Web);
    ctx.ExecuteQuery();

    // here the default network credentials relate to the identity of the account
    // running the App Pool of your web application.
    CredentialCache credCache = new CredentialCache();
    cc.Add(new Uri(ctx.Web.Url), "NTLM", CredentialCache.DefaultNetworkCredentials);

    ctx.Credentials = credCache;
    ctx.AuthenticationMode = ClientAuthentication.Default;
    ctx.ExecuteQuery();

    // do stuff as elevated app pool account
}
```

SharePointでアプリケーションプールアカウントのされたをすることはベストプラクティスにしますが、するネットワークはそのわりにできることにしてください。

オンラインでマネージドクライアントサイドオブジェクトモデルCSOMのをむ
https://riptutorial.com/ja/sharepoint/topic/2679/マネージドクライアントサイドオブジェクトモデル-csom-の

9: メジャーリリース

Examples

SharePoint 2016

ビルド		
16.0.4366.1000	20164の	SharePoint Server 2016
16.0.4336.1000	RTM	SharePoint Server 2016
16.0.4327.1000	リリース	SharePoint Server 2016
16.0.4266.1001	16.0.4306.1002 Beta 2	SharePoint Server 2016

SharePoint 2013

ビルド	
15.0.4623.1001	20146
15.0.4631.1001	20147
15.0.4641.1001	20148
15.0.4649.1001	20149
15.0.4659.1001	201410
15.0.4667.1000	201411
15.0.4675.1000	201412
15.0.4693.1001	20152
15.0.4701.1001	20153
15.0.4711.1000	20154SP1 REQ
15.0.4719.1002	20155
15.0.4727.1001	20156
15.0.4737.1000	20157
15.0.4745.1000	20158

ビルド	
15.0.4753.1003	20159
15.0.4763.1002	201510
15.0.4771.1000	201511
15.0.4779.1000	201512
15.0.4787.1000	MS16-004
15.0.4787.1000	Januar 2016
15.0.4797.1001	20162
15.0.4805.1000	20163
15.0.4815.1000	20164
15.0.4823.1003	20165
15.0.4833.1000	20166

SharePoint 2013ビルドとCU

オンラインでメジャーリリースをむ https://riptutorial.com/ja/sharepoint/topic/2737/メジャーリリース

10: サーバーサイドオブジェクトモデルフルトラストの

SharePointでは、サイトコレクションにはサイトがまれ、サイトにはがまれます。サイトコレクション `SPSite` にはなUIはありませんが、に1つのルートレベルサイト `RootWeb` プロパティでアクセスとそのルートサイトのにあるのサブサイトがにまれています。サイトまたはWeb `SPWeb` にはUIがあり、リスト/ドキュメントライブラリ `SPList` 、Web `SPWeb` をむページ、およびアイテム/ドキュメント `SPListItem` がまれています。

サーバーサイドの

- Visual Studioプロジェクトで、SharePointサーバーオブジェクトモデルをするアプリケーションをするには、Framework AssemblieにリストされているMicrosoft.SharePointアセンブリへのをするがあります。
- サーバーサイドオブジェクトモデルをするアプリケーションは、SharePointをホストしているWindows Serverでのみできます。
- アプリケーションがされているサーバーのSharePointサーバーにすることはできません。

Examples

Hello Worldサイトタイトル

2013

SharePoint 2013のバージョンは64ビットのみであるため、アセンブリ/プログラムも64ビットプロセッサにするがあります。

プロジェクトがされたに、 **Platform** ターゲットをの**CPU**から**x64**にりえるがあります。そうしないと、エラーがします。

```
using System;
using Microsoft.SharePoint;

namespace StackOverflow
{
    class Samples
    {
        static void Main()
        {
            using (SPSite site = new SPSite("http://server/sites/siteCollection"))
            using (SPWeb web = site.OpenWeb())
            {
                Console.WriteLine("Title: {0} Description: {1}", web.Title, web.Description);
```

```
            }
        }
    }
}
```

SharePointファームをループする

SharePoint WebサーバーからされたPowerShellの

```
$wacoll = get-spwebapplication
foreach($wa in $wacoll){
    if($wa.IsAdministrationWebApplication -eq $false){
        foreach($site in $wa.Sites){
            foreach($web in $site.AllWebs){
                # your code here
                $web.Dispose()
            }
            $site.Dispose()
        }
    }
}
```

リストアイテムをする

```
using (SPSite site = new SPSite("http://server/sites/siteCollection"))
using (SPWeb web = site.OpenWeb())
{
    SPList list = web.Lists["Some list"];

    // It is always better and faster to query list items with GetItems method with
    // empty SPQuery object than to use Items property
    SPListItemCollection items = list.GetItems(new SPQuery());
    foreach (SPListItem item in items)
    {
        // Do some operation with item
    }
}
```

ページングをしてアイテムをする

```
using (SPSite site = new SPSite("http://server/sites/siteCollection"))
using (SPWeb web = site.OpenWeb())
{
    SPList list = web.Lists["Some list"];
    SPQuery query = new SPQuery()
    {
        RowLimit = 100
    };

    do
    {
        SPListItemCollection items = list.GetItems(query);
        foreach (SPListItem item in items)
        {
            // Do some operation with item
```

```
        }

        // Assign current position to SPQuery object
        query.ListItemCollectionPosition = items.ListItemCollectionPosition;
    } while (query.ListItemCollectionPosition != null);
}
```

urlによるリストの

```
using (SPSite site = new SPSite("http://server/sites/siteCollection"))
using (SPWeb web = site.OpenWeb())
{
    string listUrl = string.Format("{0}{1}", web.ServerRelativeUrl, "Lists/SomeList");
    SPList list = web.GetList(listUrl);
}
```

リストの

しいリストアイテムをするとき、そのフィールドはにたをしてできます。これらのフィールドはオンザフライではされず、リストのスキーマによってされることにしてください。これらのフィールドまたははサーバーにしていなければなりません。そうでない、createはします。すべてのリストアイテムにはタイトルフィールドがあります。のリストには、アイテムがリストにされるにするのあるフィールドがあるがあります。

このでは、リストはAnnouncementsテンプレートをしています。タイトルフィールドにえて、リストには、アナウンスのをリストにするBodyフィールドがまれています。

```
using (SPSite site = new SPSite("http://server/sites/siteCollection"))
using (SPWeb web = site.OpenWeb())
{
    SPList list = web.Lists["Announcements"];

    SPListItem item = list.AddItem();
    item[SPBuiltInFieldId.Title] = "My new item";
    item[SPBuiltInFieldId.Body] = "Hello World!";
    item.Update();
}
```

オンラインでサーバーサイドオブジェクトモデルフルトラストのをむ
https://riptutorial.com/ja/sharepoint/topic/7543/サーバーサイドオブジェクトモデル-フルトラスト-の

クレジット

S. No		Contributors
1	シェアポイントの	Community, Marco, Ryan Gregg, Thriggle, Tom Resing, Zach Koehne
2	JavaScriptクライアントオブジェクトモデルJSOMの	Thriggle, yngrdyn
3	JavaScriptをしたモーダルダィアログボックスの	Thriggle
4	RESTサービス	Aaron, Brock Davis, ocelotsloth, R4mbi, Rohit Waghela, Thriggle
5	SharePoint 2013クライアントのレンダリング	Rohit Waghela, Yayati
6	SharePointアプリケーション	Sunil sahu
7	プロバィダのアプリケーションをする	vinayak hegde
8	マネージドクライアントサイドオブジェクトモデルCSOMの	InvoiceGuy, Lukáš Nešpor, MikhailSP, RamenChef, Thriggle, Zach Koehne
9	メジャーリース	jjr2527, MikhailSP
10	サーバーサイドオブジェクトモデルフルトラストの	Lukáš Nešpor, Thriggle