**How To Integrate QuickBooks Desktop Using Web Service (QBXML)**

This sample is a C# ASP.NET web service application that communicates with QuickBooks Point of Sale via QBWebConnector. The sample focuses primarily on demonstrating how to set up all web service web methods to run against QBWebConnector and does not focus on any particular use case.

**Step 1** Install QuickBooks desktop from [here](https://community.intuit.com/articles/1207255-quickbooks-desktop-trial-links)

**Step 2** Install QuickBooks Web Connector from [here](https://developer.intuit.com/app/developer/qbdesktop/docs/get-started/get-started-with-quickbooks-web-connector#download-and-install-the-quickbooks-web-connector)

**Step 3**

Create a new web service in Visual Studio and give the name of this web service as QWCPOSWebService

**Step 4 - QuickBooks WebService**

Open your Service class file and paste the below code into that.

1. //
2. // QuickBooks Web Connector Sample: WCWebService
3. // Copyright (c) 2006-2007 Intuit, Inc
4. //
5. // This sample is a C# ASP.NET web service application that
6. // communicates with QuickBooks Point of Sale via QBWebConnector. The
7. // sample focuses primarily on demonstrating how to setup all web service
8. // web methods to run against QBWebConnector and does not focus on any
9. // particular use case. For simplicity, it sends three request XMLs:
10. // CustomerQuery, ItenInventoryQuery and PurchaseOrderQuery.
11. //
12. // This sample assumes that you have configured IIS with ASP.NET and
13. // have a functional system to deploy this web service sample. If you have
14. // not yet configured ASP.NET with IIS, you may need to run the
15. // following command from c:\windows\Microsoft.NET\Framework\
16. // your\_asp\_dot\_net\_version path: -
17. // aspnet\_regiis /i
18. // This will help avoid the occasional message from microsoft development
19. // environment such as "VS.NET has detected that the specified web server
20. // is not running ASP.NET version 1.1. You will be unable to run ASP.NET
21. // web applications or services)".

24. /\*
25. \* Useful note about using OwnerID and FileID in a real-world application
26. \*
27. \* As part of your QB Web Connector configuration (.QWC) file, you include
28. \* OwnerID and FileID. Following note on these two parameters may be useful.
29. \*
30. \* OwnerID -- this is a GUID that represents your application or suite of
31. \* applications, if your application needs to store private data in the
32. \* company file for one reason or another (one of the most common cases
33. \* being to check if you have communicated with this company file before,
34. \* and possibly some data about that communication) that private data will
35. \* be visible to any application that knows the OwnerID.
36. \*
37. \* FileID -- this is a GUID we stamp in the file on your behalf
38. \* (using your OwnerID) as a private data extension to the "Company" object.
39. \* It allows an application to verify that the company file it is exchanging
40. \* data with is consistent over time (by doing a CompanyQuery with the field
41. \* set appropriately and reading the DataExtRet values returned.
42. \*
43. \* \*/

46. **using** System;
47. **using** System.Collections;
48. **using** System.ComponentModel;
49. **using** System.Data;
50. **using** System.Diagnostics;
51. **using** System.Web;
52. **using** System.Web.Services;
53. **using** System.IO;
54. **using** System.Security.Cryptography;
55. **using** Microsoft.Win32;
56. **using** System.Xml;
57. **using** System.Text.RegularExpressions;

60. **namespace** QWCPOSWebService
61. {
62. /// <summary>
63. ///  Web Service Namespace="http://developer.intuit.com/",
64. ///  Web Service Name="QWCPOSWebService",
65. ///  Web Service Description="Sample WebService in ASP.NET to
66. ///  demonstrate QBWebConnector with QuickBooks POS
67. /// </summary>
68. [WebService(
69. Namespace = "http://developer.intuit.com/",
70. Name = "QWCPOSWebService",
71. Description = "Sample WebService in ASP.NET to demonstrate " +
72. "QBWebConnector with QuickBooks POS")]

75. // Important Note:
76. // You should keep the namespace as http://developer.intuit.com/ for all web
77. // services that communicates with QuickBooks Web Connector.
78. **public** **class** QWCPOSWebService : System.Web.Services.WebService
79. {
80. #region GlobalVariables
81. System.Diagnostics.EventLog evLog = **new** System.Diagnostics.EventLog();
82. **public** **int** count = 0;
83. **public** ArrayList req = **new** ArrayList();
84. #endregion

87. #region Constructor
88. **public** QWCPOSWebService()
89. {
90. //CODEGEN: This call is required by the ASP.NET
91. //Web Services Designer
92. InitializeComponent();
93. // Initializing EventLog for logging
94. initEvLog();
95. }
96. #endregion

99. #region AutoGeneratedMethods
100. //Required by the Web Services Designer
101. **private** IContainer components = **null**;

104. /// <summary>
105. /// Required method for Designer support - do not modify
106. /// the contents of this method with the code editor.
107. /// </summary>
108. **private** **void** InitializeComponent()
109. {
110. }

113. /// <summary>
114. /// Clean up any resources being used.
115. /// </summary>
116. **protected** **override** **void** Dispose(**bool** disposing)
117. {
118. **if** (disposing && components != **null**)
119. {
120. components.Dispose();
121. }
122. **base**.Dispose(disposing);
123. }

126. #endregion

129. #region WebMethods
130. [WebMethod]
131. /// <summary>
132. /// WebMethod# 1 - clientVersion()
133. /// To enable web service with QBWC version control
134. /// Signature: public string clientVersion(string strVersion)
135. ///
136. /// IN:
137. /// string strVersion
138. ///
139. /// OUT:
140. /// string errorOrWarning
141. /// Possible values:
142. /// string retVal
143. /// - NULL or <emptyString> = QBWC will let the web service update
144. /// - "E:<any text>" = popup ERROR dialog with <any text>
145. /// - abort update and force download of new QBWC.
146. /// - "W:<any text>" = popup WARNING dialog with <any text>
147. /// - choice to user, continue update or not.
148. /// </summary>
149. **public** **string** clientVersion(**string** strVersion)
150. {
151. **string** evLogTxt = "WebMethod: clientVersion() has been called " +
152. "by QBWebconnector" + "\r\n\r\n";
153. evLogTxt = evLogTxt + "Parameters received:\r\n";
154. evLogTxt = evLogTxt + "string strVersion = " + strVersion + "\r\n";
155. evLogTxt = evLogTxt + "\r\n";

158. **string** retVal = **null**;
159. **double** recommendedVersion = 1.5;
160. **double** supportedMinVersion = 1.0;
161. **double** suppliedVersion = Convert.ToDouble(**this**.parseForVersion(strVersion));
162. evLogTxt = evLogTxt + "QBWebConnector version = " + strVersion + "\r\n";
163. evLogTxt = evLogTxt + "Recommended Version = " + recommendedVersion.ToString() + "\r\n";
164. evLogTxt = evLogTxt + "Supported Minimum Version = " + supportedMinVersion.ToString() + "\r\n";
165. evLogTxt = evLogTxt + "SuppliedVersion = " + suppliedVersion.ToString() + "\r\n";
166. **if** (suppliedVersion < recommendedVersion)
167. {
168. retVal = "W:We recommend that you upgrade your QBWebConnector";
169. }
170. **else** **if** (suppliedVersion < supportedMinVersion)
171. {
172. retVal = "E:You need to upgrade your QBWebConnector";
173. }
174. evLogTxt = evLogTxt + "\r\n";
175. evLogTxt = evLogTxt + "Return values: " + "\r\n";
176. evLogTxt = evLogTxt + "string retVal = " + retVal;
177. logEvent(evLogTxt);
178. **return** retVal;
179. }
180. [WebMethod]
181. /// <summary>
182. /// WebMethod# 2 - authenticate()
183. /// To verify username and password for the web connector that is trying to connect
184. /// Signature: public string[] authenticate(string strUserName, string strPassword)
185. ///
186. /// IN:
187. /// string strUserName
188. /// string strPassword
189. ///
190. /// OUT:
191. /// string[] authReturn
192. /// Possible values:
193. /// string[0] = ticket
194. /// string[1]
195. /// - empty string = use current company file
196. /// - "none" = no further request/no further action required
197. /// - "nvu" = not valid user
198. /// - any other string value = use this company file
199. /// </summary>
200. **public** **string**[] authenticate(**string** strUserName, **string** strPassword)
201. {
202. **string** evLogTxt = "WebMethod: authenticate() has been called by QBWebconnector" + "\r\n\r\n";
203. evLogTxt = evLogTxt + "Parameters received:\r\n";
204. evLogTxt = evLogTxt + "string strUserName = " + strUserName + "\r\n";
205. evLogTxt = evLogTxt + "string strPassword = " + strPassword + "\r\n";
206. evLogTxt = evLogTxt + "\r\n";
207. **string**[] authReturn = **new** **string**[2];
208. // Code below uses a random GUID to use as session ticket
209. // An example of a GUID is {85B41BEE-5CD9-427a-A61B-83964F1EB426}
210. authReturn[0] = System.Guid.NewGuid().ToString();

213. // For simplicity of sample, a hardcoded username/password is used.
214. // In real world, you should handle authentication in using a standard way.
215. // For example, you could validate the username/password against an LDAP
216. // or a directory server
217. **string** pwd = "password";
218. evLogTxt = evLogTxt + "Password locally stored = " + pwd + "\r\n";
219. **if** (strUserName.ToUpper().Trim().Equals("USERNAME") && strPassword.ToUpper().Trim().Equals(pwd.ToUpper()))
220. {
221. // An empty string for authReturn[1] means asking QBWebConnector
222. // to connect to the company file that is currently openned in QB
223. authReturn[1] = "Company Data=IqbalStore";
224. }
225. **else**
226. {
227. authReturn[1] = "nvu";
228. }
229. // You could also return "none" to indicate there is no work to do
230. // or a company filename in the format C:\full\path\to\company.qbw
231. // based on your program logic and requirements.

234. evLogTxt = evLogTxt + "\r\n";
235. evLogTxt = evLogTxt + "Return values: " + "\r\n";
236. evLogTxt = evLogTxt + "string[] authReturn[0] = " + authReturn[0].ToString() + "\r\n";
237. evLogTxt = evLogTxt + "string[] authReturn[1] = " + authReturn[1].ToString();
238. logEvent(evLogTxt);
239. **return** authReturn;
240. }
241. [WebMethod(Description = "This web method facilitates web service to handle connection errors between QuickBooks and QBWebConnector", EnableSession = **true**)]
242. /// <summary>
243. /// WebMethod# 3 - connectionError()
244. /// To facilitate capturing of QuickBooks error and notifying it to web services
245. /// Signature: public string connectionError (string ticket, string hresult, string message)
246. ///
247. /// IN:
248. /// string ticket = A GUID based ticket string to maintain identity of QBWebConnector
249. /// string hresult = An HRESULT value thrown by QuickBooks when trying to make connection
250. /// string message = An error message corresponding to the HRESULT
251. ///
252. /// OUT:
253. /// string retVal
254. /// Possible values:
255. /// - “done” = no further action required from QBWebConnector
256. /// - any other string value = use this name for company file
257. /// </summary>
258. **public** **string** connectionError(**string** ticket, **string** hresult, **string** message)
259. {
260. **if** (Session["ce\_counter"] == **null**)
261. {
262. Session["ce\_counter"] = 0;
263. }

266. **string** evLogTxt = "WebMethod: connectionError() has been called by QBWebconnector" + "\r\n\r\n";
267. evLogTxt = evLogTxt + "Parameters received:\r\n";
268. evLogTxt = evLogTxt + "string ticket = " + ticket + "\r\n";
269. evLogTxt = evLogTxt + "string hresult = " + hresult + "\r\n";
270. evLogTxt = evLogTxt + "string message = " + message + "\r\n";
271. evLogTxt = evLogTxt + "\r\n";

274. **string** retVal = **null**;
275. //-2147418113 = Can't connect to the database
276. **const** **string** CANT\_CONNECT\_TO\_DB = "0x8000FFFF";
277. // Add more as you need...

280. **if** (hresult.Trim().Equals(CANT\_CONNECT\_TO\_DB))
281. {
282. evLogTxt = evLogTxt + "HRESULT = " + hresult + "\r\n";
283. evLogTxt = evLogTxt + "Message = " + message + "\r\n";
284. retVal = "DONE";
285. }
286. **else**
287. {
288. // Depending on various hresults return different value
289. **if** ((**int**)Session["ce\_counter"] == 0)
290. {
291. // Try again with this company file
292. evLogTxt = evLogTxt + "HRESULT = " + hresult + "\r\n";
293. evLogTxt = evLogTxt + "Message = " + message + "\r\n";
294. evLogTxt = evLogTxt + "Sending connection string as \"Company Data=\" to QBWebConnector.";
295. retVal = "Company Data=";
296. }
297. **else**
298. {
299. evLogTxt = evLogTxt + "HRESULT = " + hresult + "\r\n";
300. evLogTxt = evLogTxt + "Message = " + message + "\r\n";
301. evLogTxt = evLogTxt + "Sending DONE to stop.";
302. retVal = "DONE";
303. }
304. }
305. evLogTxt = evLogTxt + "\r\n";
306. evLogTxt = evLogTxt + "Return values: " + "\r\n";
307. evLogTxt = evLogTxt + "string retVal = " + retVal + "\r\n";
308. logEvent(evLogTxt);
309. Session["ce\_counter"] = ((**int**)Session["ce\_counter"]) + 1;
310. **return** retVal;
311. }
312. [WebMethod(Description = "This web method facilitates web service to send request XML to QuickBooks via QBWebConnector", EnableSession = **true**)]
313. /// <summary>
314. /// WebMethod# 4 - sendRequestXML()
315. /// Signature: public string sendRequestXML(string ticket, string strHCPResponse, string strCompanyFileName,
316. /// string Country, int qbXMLMajorVers, int qbXMLMinorVers)
317. ///
318. /// IN:
319. /// int qbXMLMajorVers
320. /// int qbXMLMinorVers
321. /// string ticket
322. /// string strHCPResponse
323. /// string strCompanyFileName
324. /// string Country
325. /// int qbXMLMajorVers
326. /// int qbXMLMinorVers
327. ///
328. /// OUT:
329. /// string request
330. /// Possible values:
331. /// - “any\_string” = Request XML for QBWebConnector to process
332. /// - "" = No more request XML
333. /// </summary>
334. **public** **string** sendRequestXML(**string** ticket, **string** strHCPResponse, **string** strCompanyFileName,
335. **string** qbXMLCountry, **int** qbXMLMajorVers, **int** qbXMLMinorVers)
336. {
337. **if** (Session["counter"] == **null**)
338. {
339. Session["counter"] = 0;
340. }
341. **string** evLogTxt = "WebMethod: sendRequestXML() has been called by QBWebconnector" + "\r\n\r\n";
342. evLogTxt = evLogTxt + "Parameters received:\r\n";
343. evLogTxt = evLogTxt + "string ticket = " + ticket + "\r\n";
344. evLogTxt = evLogTxt + "string strHCPResponse = " + strHCPResponse + "\r\n";
345. evLogTxt = evLogTxt + "string strCompanyFileName = " + strCompanyFileName + "\r\n";
346. evLogTxt = evLogTxt + "string qbXMLCountry = " + qbXMLCountry + "\r\n";
347. evLogTxt = evLogTxt + "int qbXMLMajorVers = " + qbXMLMajorVers.ToString() + "\r\n";
348. evLogTxt = evLogTxt + "int qbXMLMinorVers = " + qbXMLMinorVers.ToString() + "\r\n";
349. evLogTxt = evLogTxt + "\r\n";

352. ArrayList req = buildRequest();
353. **string** request = "";
354. **int** total = req.Count;
355. count = Convert.ToInt32(Session["counter"]);

358. **if** (count < total)
359. {
360. request = req[count].ToString();
361. evLogTxt = evLogTxt + "sending request no = " + (count + 1) + "\r\n";
362. Session["counter"] = ((**int**)Session["counter"]) + 1;
363. }
364. **else**
365. {
366. count = 0;
367. Session["counter"] = 0;
368. request = "";
369. }
370. evLogTxt = evLogTxt + "\r\n";
371. evLogTxt = evLogTxt + "Return values: " + "\r\n";
372. evLogTxt = evLogTxt + "string request = " + request + "\r\n";
373. logEvent(evLogTxt);
374. **return** request;
375. }
376. [WebMethod(Description = "This web method facilitates web service to receive response XML from QuickBooks via QBWebConnector", EnableSession = **true**)]
377. /// <summary>
378. /// WebMethod# 5 - receiveResponseXML()
379. /// Signature: public int receiveResponseXML(string ticket, string response, string hresult, string message)
380. ///
381. /// IN:
382. /// string ticket
383. /// string response
384. /// string hresult
385. /// string message
386. ///
387. /// OUT:
388. /// int retVal
389. /// Greater than zero  = There are more request to send
390. /// 100 = Done. no more request to send
391. /// Less than zero  = Custom Error codes
392. /// </summary>
393. **public** **int** receiveResponseXML(**string** ticket, **string** response, **string** hresult, **string** message)
394. {
395. **string** evLogTxt = "WebMethod: receiveResponseXML() has been called by QBWebconnector" + "\r\n\r\n";
396. evLogTxt = evLogTxt + "Parameters received:\r\n";
397. evLogTxt = evLogTxt + "string ticket = " + ticket + "\r\n";
398. evLogTxt = evLogTxt + "string response = " + response + "\r\n";
399. evLogTxt = evLogTxt + "string hresult = " + hresult + "\r\n";
400. evLogTxt = evLogTxt + "string message = " + message + "\r\n";
401. evLogTxt = evLogTxt + "\r\n";

404. **int** retVal = 0;
405. **if** (!hresult.ToString().Equals(""))
406. {
407. // if there is an error with the response received, web service could also return a -ve int
409. evLogTxt = evLogTxt + "HRESULT = " + hresult + "\r\n";
410. evLogTxt = evLogTxt + "Message = " + message + "\r\n";
411. retVal = -101;
412. }
413. **else**
414. {
415. evLogTxt = evLogTxt + "Length of response received = " + response.Length + "\r\n";

418. ArrayList req = buildRequest();
419. **int** total = req.Count;
420. **int** count = Convert.ToInt32(Session["counter"]);

423. **int** percentage = (count \* 100) / total;
424. **if** (percentage >= 100)
425. {
426. count = 0;
427. Session["counter"] = 0;
428. }
429. retVal = percentage;
430. }
431. evLogTxt = evLogTxt + "\r\n";
432. evLogTxt = evLogTxt + "Return values: " + "\r\n";
433. evLogTxt = evLogTxt + "int retVal= " + retVal.ToString() + "\r\n";
434. logEvent(evLogTxt);
435. **return** retVal;
436. }

439. [WebMethod]
440. /// <summary>
441. /// WebMethod# 6 - getLastError()
442. /// Signature: public string getLastError(string ticket)
443. ///
444. /// IN:
445. /// string ticket
446. ///
447. /// OUT:
448. /// string retVal
449. /// Possible Values:
450. /// Error message describing last web service error
451. /// </summary>
452. **public** **string** getLastError(**string** ticket)
453. {
454. **string** evLogTxt = "WebMethod: getLastError() has been called by QBWebconnector" + "\r\n\r\n";
455. evLogTxt = evLogTxt + "Parameters received:\r\n";
456. evLogTxt = evLogTxt + "string ticket = " + ticket + "\r\n";
457. evLogTxt = evLogTxt + "\r\n";

460. **int** errorCode = 0;
461. **string** retVal = **null**;
462. **if** (errorCode == -101)
463. {
464. retVal = "QuickBooks was not running!"; // This is just an example of custom user errors
465. }
466. **else**
467. {
468. retVal = "Error!";
469. }
470. evLogTxt = evLogTxt + "\r\n";
471. evLogTxt = evLogTxt + "Return values: " + "\r\n";
472. evLogTxt = evLogTxt + "string retVal= " + retVal + "\r\n";
473. logEvent(evLogTxt);
474. **return** retVal;
475. }
476. [WebMethod]
477. /// <summary>
478. /// WebMethod# 7 - closeConnection()
479. /// At the end of a successful update session, QBWebConnector will call this web method.
480. /// Signature: public string closeConnection(string ticket)
481. ///
482. /// IN:
483. /// string ticket
484. ///
485. /// OUT:
486. /// string closeConnection result
487. /// </summary>
488. **public** **string** closeConnection(**string** ticket)
489. {
490. **string** evLogTxt = "WebMethod: closeConnection() has been called by QBWebconnector" + "\r\n\r\n";
491. evLogTxt = evLogTxt + "Parameters received:\r\n";
492. evLogTxt = evLogTxt + "string ticket = " + ticket + "\r\n";
493. evLogTxt = evLogTxt + "\r\n";
494. **string** retVal = **null**;

497. retVal = "OK";

500. evLogTxt = evLogTxt + "\r\n";
501. evLogTxt = evLogTxt + "Return values: " + "\r\n";
502. evLogTxt = evLogTxt + "string retVal= " + retVal + "\r\n";
503. logEvent(evLogTxt);
504. **return** retVal;
505. }

508. #endregion

511. #region UtilityMethods
512. **private** **void** initEvLog()
513. {
514. **try**
515. {
516. **string** source = "WCWebService";
517. **if** (!System.Diagnostics.EventLog.SourceExists(source))
518. System.Diagnostics.EventLog.CreateEventSource(source, "Application");
519. evLog.Source = source;
520. }
521. **catch** { };
522. **return**;
523. }

526. **private** **void** logEvent(**string** logText)
527. {
528. **try**
529. {
530. evLog.WriteEntry(logText);
531. }
532. **catch** { };
533. **return**;
534. }

537. **public** ArrayList buildRequest()
538. {
539. **string** strRequestXML = "";
540. XmlDocument inputXMLDoc = **null**;

543. // CustomerQuery
544. inputXMLDoc = **new** XmlDocument();
545. inputXMLDoc.AppendChild(inputXMLDoc.CreateXmlDeclaration("1.0", **null**, **null**));
546. inputXMLDoc.AppendChild(inputXMLDoc.CreateProcessingInstruction("qbposxml", "version=\"1.0\""));

549. XmlElement qbposXML = inputXMLDoc.CreateElement("QBPOSXML");
550. inputXMLDoc.AppendChild(qbposXML);
551. XmlElement qbposXMLMsgsRq = inputXMLDoc.CreateElement("QBPOSXMLMsgsRq");
552. qbposXML.AppendChild(qbposXMLMsgsRq);
553. qbposXMLMsgsRq.SetAttribute("onError", "stopOnError");
554. XmlElement customerQueryRq = inputXMLDoc.CreateElement("CustomerQueryRq");
555. qbposXMLMsgsRq.AppendChild(customerQueryRq);
556. customerQueryRq.SetAttribute("requestID", "1");
557. XmlElement maxReturned = inputXMLDoc.CreateElement("MaxReturned");
558. customerQueryRq.AppendChild(maxReturned).InnerText = "1";

561. strRequestXML = inputXMLDoc.OuterXml;
562. req.Add(strRequestXML);

565. // Clean up
566. strRequestXML = "";
567. inputXMLDoc = **null**;
568. qbposXML = **null**;
569. qbposXMLMsgsRq = **null**;
570. maxReturned = **null**;

573. // ItemInventoryQuery
574. inputXMLDoc = **new** XmlDocument();
575. inputXMLDoc.AppendChild(inputXMLDoc.CreateXmlDeclaration("1.0", **null**, **null**));
576. inputXMLDoc.AppendChild(inputXMLDoc.CreateProcessingInstruction("qbposxml", "version=\"1.0\""));

579. qbposXML = inputXMLDoc.CreateElement("QBPOSXML");
580. inputXMLDoc.AppendChild(qbposXML);
581. qbposXMLMsgsRq = inputXMLDoc.CreateElement("QBPOSXMLMsgsRq");
582. qbposXML.AppendChild(qbposXMLMsgsRq);
583. qbposXMLMsgsRq.SetAttribute("onError", "stopOnError");
584. XmlElement itemInventoryQueryRq = inputXMLDoc.CreateElement("ItemInventoryQueryRq");
585. qbposXMLMsgsRq.AppendChild(itemInventoryQueryRq);
586. itemInventoryQueryRq.SetAttribute("requestID", "2");
587. maxReturned = inputXMLDoc.CreateElement("MaxReturned");
588. itemInventoryQueryRq.AppendChild(maxReturned).InnerText = "1";

591. strRequestXML = inputXMLDoc.OuterXml;
592. req.Add(strRequestXML);

595. // Clean up
596. strRequestXML = "";
597. inputXMLDoc = **null**;
598. qbposXML = **null**;
599. qbposXMLMsgsRq = **null**;
600. maxReturned = **null**;

603. // PurchaseOrderQuery
604. inputXMLDoc = **new** XmlDocument();
605. inputXMLDoc.AppendChild(inputXMLDoc.CreateXmlDeclaration("1.0", **null**, **null**));
606. inputXMLDoc.AppendChild(inputXMLDoc.CreateProcessingInstruction("qbposxml", "version=\"1.0\""));

609. qbposXML = inputXMLDoc.CreateElement("QBPOSXML");
610. inputXMLDoc.AppendChild(qbposXML);
611. qbposXMLMsgsRq = inputXMLDoc.CreateElement("QBPOSXMLMsgsRq");
612. qbposXML.AppendChild(qbposXMLMsgsRq);
613. qbposXMLMsgsRq.SetAttribute("onError", "stopOnError");
614. XmlElement purchaseOrderQueryRq = inputXMLDoc.CreateElement("PurchaseOrderQueryRq");
615. qbposXMLMsgsRq.AppendChild(purchaseOrderQueryRq);
616. purchaseOrderQueryRq.SetAttribute("requestID", "3");
617. maxReturned = inputXMLDoc.CreateElement("MaxReturned");
618. purchaseOrderQueryRq.AppendChild(maxReturned).InnerText = "1";

621. strRequestXML = inputXMLDoc.OuterXml;
622. req.Add(strRequestXML);

625. // InvoiceQuery
626. //inputXMLDoc = new XmlDocument();
627. //inputXMLDoc.AppendChild(inputXMLDoc.CreateXmlDeclaration("1.0",null, null));
628. //          inputXMLDoc.AppendChild(inputXMLDoc.CreateProcessingInstruction("qbxml", "version=\"4.0\""));
630. //          qbXML = inputXMLDoc.CreateElement("QBXML");
631. //          inputXMLDoc.AppendChild(qbXML);
632. //          qbXMLMsgsRq = inputXMLDoc.CreateElement("QBXMLMsgsRq");
633. //          qbXML.AppendChild(qbXMLMsgsRq);
634. //          qbXMLMsgsRq.SetAttribute("onError", "stopOnError");
635. //          XmlElement invoiceQueryRq = inputXMLDoc.CreateElement("InvoiceQueryRq");
636. //qbXMLMsgsRq.AppendChild(invoiceQueryRq);
637. //          invoiceQueryRq.SetAttribute("requestID", "2");
638. //          maxReturned=inputXMLDoc.CreateElement("MaxReturned");
639. //          invoiceQueryRq.AppendChild(maxReturned).InnerText="1";
641. //          strRequestXML = inputXMLDoc.OuterXml;
642. //          req.Add(strRequestXML);

645. **return** req;
646. }
647. **private** **string** parseForVersion(**string** input)
648. {
649. // This method is created just to parse the first two version components
650. // out of the standard four component version number:
651. // <Major>.<Minor>.<Release>.<Build>
652. //
653. // As long as you get the version in right format, you could use
654. // any algorithm here.
655. **string** retVal = "";
656. **string** major = "";
657. **string** minor = "";
658. Regex version = **new** Regex(@"^(?\d+)\.(?\d+)(\.\w+){0,2}$", RegexOptions.Compiled);
659. Match versionMatch = version.Match(input);
660. **if** (versionMatch.Success)
661. {
662. major = versionMatch.Result("${major}");
663. minor = versionMatch.Result("${minor}");
664. retVal = major + "." + minor;
665. }
666. **else**
667. {
668. retVal = input;
669. }
670. **return** retVal;
671. }
672. #endregion
673. }
674. }

**Step 5 - Generate QWC File**

Now, create the QuickBooks file for making the connection between QuickBooks desktop and your web service.

As part of your QB Web Connector configuration (.QWC) file, you include OwnerID and FileID. The following note on these two parameters may be useful.

**OwnerID**

This is a GUID that represents your application or suite of applications if your application needs to store private data in the company file. Another reason (one of the most common cases being to check if you have communicated with this company file before, and possibly some data about that communication) is that private data will be visible to any application that knows the OwnerID.

**FileID**

This is a GUID we stamp in the file on your behalf (using your OwnerID) as a private data extension to the "Company" object. It allows an application to verify that the company file it is exchanging data with is consistent over time (by doing a Company Query with the field).

An example of GUID is {85B41BEE-5CD9-427a-A61B-83964F1EB426}

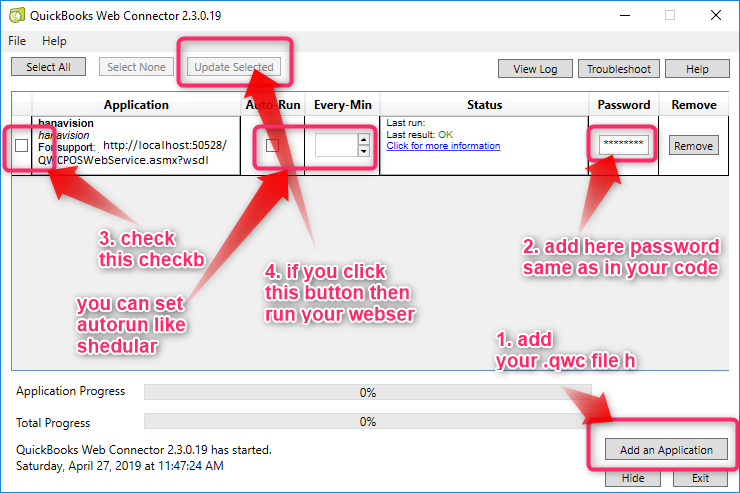
Given below is the code for them .QWC file.

1. <?xml version="1.0"?><QBWCXML>
2. <AppName>hanavision</AppName>
3. <AppID>1</AppID>
4. <AppURL>http://localhost:50528/QWCPOSWebService.asmx</AppURL>
5. <AppDescription>A **short** description **for** QWCPOSWebService</AppDescription>
6. <AppSupport>http://localhost:50528/QWCPOSWebService.asmx?wsdl</AppSupport>
7. <OwnerID>{87EDAAF8-0000-1111-2222-4BA79C2F8998}</OwnerID>
8. <FileID>{CA1C3EB8-0000-1111-2222-8D5B438B83AC}</FileID>
9. <UserName>Bhavdip</UserName>
10. <QBType>QBFS</QBType>
11. <Style>Document</Style>
12. <AuthFlags>0xF</AuthFlags>
13. </QBWCXML>

**Note**Don’t forget to change the localhost path.

**Step 6 - Add QWC file into your web connector**

Now, open your web connector and add this file into your web connector. After successfully adding this, now add your file to a web connector.

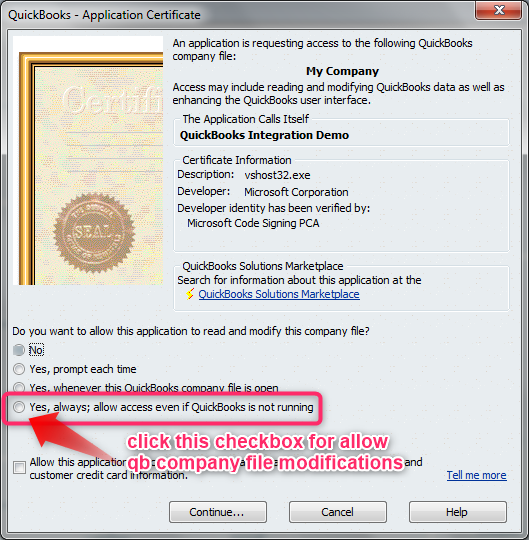


1. Add your file using the "Add an Application" button; then choose your QBC file.
2. Add password which is similar to the one in your code of web services.
3. Select this web service in web connector using the checkbox.
4. Click the "Update selected" button for running your code of web service.

If you want to run like scheduler, then you can set autorun and give the timing in every Min text box. So now, your QuickBooks desktop and your web service are connected using this web connector.

**Authorizing the Application**

On the first time that an application connects to a QuickBooks company file, QuickBooks should be opened and a user must be there to authorize the access. When we try to save a customer again and QuickBooks is running this time, a dialog will appear in QuickBooks, as shown in the figure below. We should take note that this will block the current thread so if we used the UI thread, then the UI will become unresponsive.



Example qbXML Request to "Add Customer".

1. <?xml version="1.0" encoding="utf-8"?>
2. <?qbxml version="2.0"?>
3. <QBXML>
4. <QBXMLMsgsRq onError="stopOnError">
5. <CustomerAddRq requestID="15">
6. <CustomerAdd>
7. <Name>20706 - Eastern XYZ University</Name>
8. <CompanyName>Eastern XYZ University</CompanyName>
9. <FirstName>Keith</FirstName>
10. <LastName>Palmer</LastName>
11. <BillAddress>
12. <Addr1>Eastern XYZ University</Addr1>
13. <Addr2>College of Engineering</Addr2>
14. <Addr3>123 XYZ Road</Addr3>
15. <City>Storrs-Mansfield</City>
16. <State>CT</State>
17. <PostalCode>06268</PostalCode>
18. <Country>United States</Country>
19. </BillAddress>
20. <Phone>860-634-1602</Phone>
21. <AltPhone>860-429-0021</AltPhone>
22. <Fax>860-429-5183</Fax>
23. <Email>keith@consolibyte.com</Email>
24. <Contact>Keith Palmer</Contact>
25. </CustomerAdd>
26. </CustomerAddRq>
27. </QBXMLMsgsRq>
28. </QBXML>

Example qbXML Response to "Add Customer".

1. <?xml version="1.0" ?>
2. <QBXML>
3. <QBXMLMsgsRs>
4. <CustomerAddRs requestID="15" statusCode="0" statusSeverity="Info" statusMessage="Status OK">
5. <CustomerRet>
6. <ListID>F540000-1197683154</ListID>
7. <TimeCreated>2007-12-14T20:45:54-05:00</TimeCreated>
8. <TimeModified>2007-12-14T20:45:54-05:00</TimeModified>
9. <EditSequence>1197683154</EditSequence>
10. <Name>20706 - Eastern XYZ University</Name>
11. <FullName>20706 - Eastern XYZ University</FullName>
12. <IsActive>**true**</IsActive>
13. <Sublevel>0</Sublevel>
14. <CompanyName>Eastern XYZ University</CompanyName>
15. <FirstName>Keith</FirstName>
16. <LastName>Palmer</LastName>
17. <BillAddress>
18. <Addr1>Eastern XYZ University</Addr1>
19. <Addr2>College of Engineering</Addr2>
20. <Addr3>123 XYZ Road</Addr3>
21. <City>Storrs-Mansfield</City>
22. <State>CT</State>
23. <PostalCode>88130</PostalCode>
24. <Country>USA</Country>
25. </BillAddress>
26. <Phone>860-634-1602</Phone>
27. <AltPhone>860-429-0021</AltPhone>
28. <Fax>860-429-5183</Fax>
29. <Email>keith@consolibyte.com</Email>
30. <Contact>Keith Palmer</Contact>
31. <Balance>0.00</Balance>
32. <TotalBalance>0.00</TotalBalance>
33. <JobStatus>None</JobStatus>
34. </CustomerRet>
35. </CustomerAddRs>
36. </QBXMLMsgsRs>
37. </QBXML>

Example qbXML to "Add an Invoice".

1. <?xml version="1.0" encoding="utf-8"?>
2. <?qbxml version="2.0"?>
3. <QBXML>
4. <QBXMLMsgsRq onError="stopOnError">
5. <InvoiceAddRq requestID="2">
6. <InvoiceAdd>
7. <CustomerRef>
8. <ListID>F560000-1197683156</ListID>  <!-- or
9. <Name>Bhavdip</Name> -->
10. </CustomerRef>
11. <TxnDate>2007-12-14</TxnDate>
12. <RefNumber>9668</RefNumber>
13. <BillAddress>
14. <Addr1>56 Cowles Road</Addr1>
15. <City>Willington</City>
16. <State>CT</State>
17. <PostalCode>06279</PostalCode>
18. <Country>United States</Country>
19. </BillAddress>
20. <PONumber></PONumber>
21. <Memo></Memo>
23. <InvoiceLineAdd>
24. <ItemRef>
25. <FullName>Downloaded Invoice</FullName>
26. </ItemRef>
27. <Desc>Item 1 Description Goes Here</Desc>
28. <Quantity>1</Quantity>
29. <Rate>295</Rate>
30. </InvoiceLineAdd>
32. <InvoiceLineAdd>
33. <ItemRef>
34. <FullName>Downloaded Invoice</FullName>
35. </ItemRef>
36. <Desc>Item 2 Description Goes Here</Desc>
37. <Quantity>3</Quantity>
38. <Rate>25</Rate>
39. </InvoiceLineAdd>
41. </InvoiceAdd>
42. </InvoiceAddRq>
43. </QBXMLMsgsRq>
44. </QBXML>

**Note**Make sure of the Ref like below,

1. <CustomerRef>
2. <ListID>F560000-1197683156</ListID> <!-- or
3. <Name>Bhavdip</Name> -->
4. </CustomerRef>

If your customer "Bhavdip" is already added in your QuickBooks desktop, then your invoice will be added. Otherwise, it will return an error.

Thus, first add a customer, item etc. which is required for the invoice. Then and only then make an invoice.