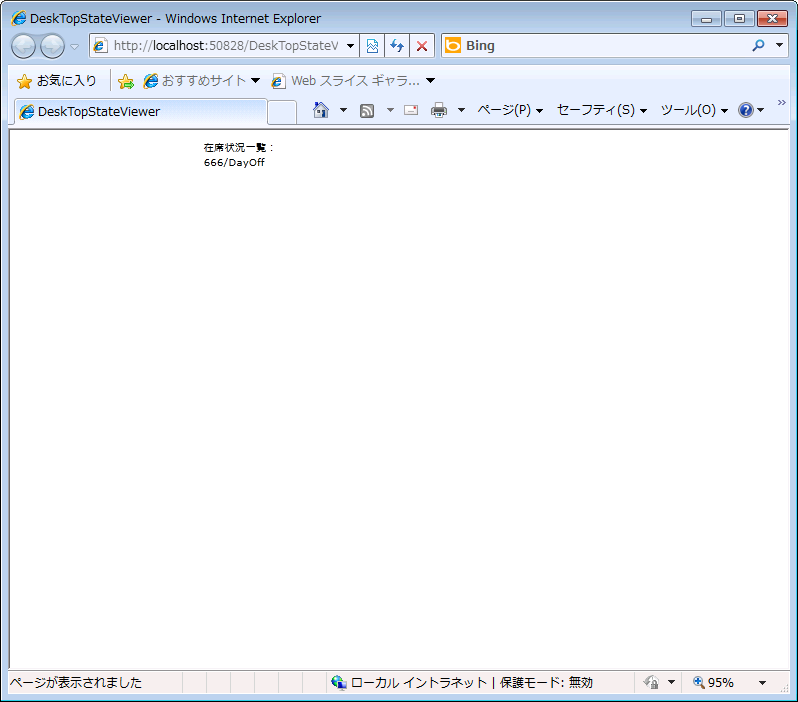
**WCF + Silverlightの双方向通信で、サーバーの状態を全員に通知する仕組み**

記事の「第4回 WCF+Silverlightで作るリアルタイムWeb（後編）」の最後の画面が表示出来ることを確認してください。特に、クライアント側のコードでサーバーアドレスをハードコーディングしている部分があります。

|  |
| --- |
| static EndpointAddress address = new EndpointAddress("http://**localhost:50828**/DeskTopStateService.svc"); |

※サーバーアドレスは、ASP.NET開発サーバーのアドレスになっているため、実行環境で異なります。



画面が表示出来ることを確認したら、次にサーバーに保存した在席状況を全員に通知するように変更します。現在は、サービス実装側のDeskTopStateServiceクラス内で自分のIDと在席状況をXMLファイルに書き込んで、再度読み出し、クライアントのGetStateRecievedを呼び出しています。

|  |
| --- |
| client.GetState(states);//サーバー側でこのように呼び出すとクライアント側の  //proxy.GetStateReceivedにセットしたproxy\_GetStateReceived  //が呼ばれる仕組みです。 |

これ（「client.GetState(states);」）を全クライアントに行いますから、この呼び出し部分で使用している「client」をコレクションにして保持しておくことで、他のユーザーの在籍状態の変更（SetState）で全員に通知する仕組みを構築できます。

まず、クラスのルートにコレクションを宣言しますが以下のコレクションの違いに注意してください。

表：代表的なコレクション【C】System.Collections【G】System.Collections.Generic

|  |
| --- |
| 【C】単にオブジェクトをグループ化したICollectionを実装したコレクション |
| 【C】固定長配列であるArray |
| 【C】IListを実装した可変長コレクションであるArrayList |
| 【C】並べ替え済みのArrayListであるSortedList |
| 【C】LINQでクエリ可能なIEnumerableを実装しているコレクション |
| 【C】キーバリューペアのコレクションであるHashtable |
| 【C】先入れ先出しのコレクションであるQueue |
| 【C】後入れ先出しのコレクションであるStack |
| 【G】重複を許容しないキーバリューペアのコレクションであるHashSet<T> |
| 【G】シリアライズ可能なキーバリューペアのコレクションであるDictionary<TKey,TValue> |
| 【G】リストの検索、並べ替え、操作を実装したList<T> |
| 【G】並べ替え済みのArrayListであるSortedSet<T> |
| 【G】先入れ先出しのコレクションであるQueue<T> |
| 【G】後入れ先出しのコレクションであるStack<T> |
| ...その他 |

今回は、特に難しいこと（コレクションを操作する際の細かな制御など）をするわけではないので、System.Collections.Genericを使います。また、特別なインデックスを必要としないためキーバリューペアは必要ありません（在籍状態を部署単位で表示したいといった場合は、WCF側でなくビュー側で表示を操作することをお勧めします）。Set操作という観点に注目すると、HashSetの持つSet操作は集合の差分等（N対N）に強いため、今回のように社員の誰かが在籍状態を更新するといったタイミングでのコレクション変更（N対1）には過大な機能となり、格納順、並び順等も関係ないことから一般的なList<T>を使うのが良いかと思われます。IDeskTopStateReceiverオブジェクトをコレクションしますので、DeskTopStateServiceクラスのルートレベルでusersコレクションを作成します。

|  |
| --- |
| static List<IDeskTopStateReceiver> users = new List<IDeskTopStateReceiver>(); |

client.GetState(states);として、XMLファイルを変更したユーザーに対して通知していた部分をコメントアウトし、コレクションに追加してから全員に通知するメソッドを呼びます。

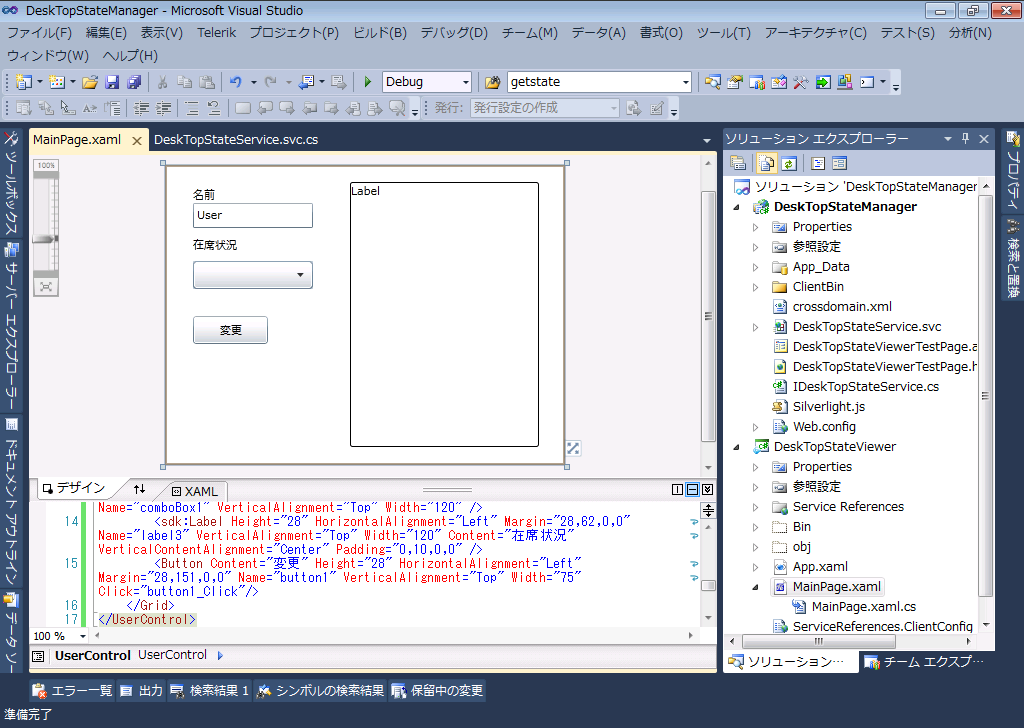
|  |
| --- |
| //client.GetState(states);  users.Add(client);  announcements(states);//全員に通知する |

全員に通知するannouncementsメソッドをDeskTopStateServiceクラスに作成し、コレクション全員にXMLの内容を通知しています。

|  |
| --- |
| private void announcements(IEnumerable<DeskTopStateSet> states)  {  foreach(IDeskTopStateReceiver u in users)  {  u.GetState(states);  }  } |

現在は、クライアント側から固定的なIDを発行するようハードコーディングしていますので、在席状態をクライアントから変更できるようにします。MainPage.xamlのLayoutRootグリッドを以下のように変更します。

|  |
| --- |
| <Grid x:Name="LayoutRoot" Background="White">  <sdk:Label Height="265" HorizontalAlignment="Left" Margin="185,17,0,0" Name="label1" VerticalAlignment="Top" Width="189" BorderThickness="1" BorderBrush="Black" />  <TextBox Height="25" HorizontalAlignment="Left" Margin="28,38,0,0" Name="textBox1" VerticalAlignment="Top" Width="120" Text="User" />  <sdk:Label Height="28" HorizontalAlignment="Left" Margin="28,12,0,0" Name="label2" VerticalAlignment="Top" Width="120" Content="名前" Padding="0,10,0,0" />  <ComboBox Height="28" HorizontalAlignment="Left" Margin="28,96,0,0" Name="comboBox1" VerticalAlignment="Top" Width="120" />  <sdk:Label Height="28" HorizontalAlignment="Left" Margin="28,62,0,0" Name="label3" VerticalAlignment="Top" Width="120" Content="在席状況" VerticalContentAlignment="Center" Padding="0,10,0,0" />  <Button Content="変更" Height="28" HorizontalAlignment="Left" Margin="28,151,0,0" Name="button1" VerticalAlignment="Top" Width="75" Click="button1\_Click"/>  </Grid> |



在席状況のリストの値は、MainPage.xaml.csのコンストラクタのInitializeComponent();の直後に以下のコードを追加して、リストにセットします。既定値はユーザーを「”user01”」が在席（DeskTopStateService.DeskTopStateType.On）としておきます。

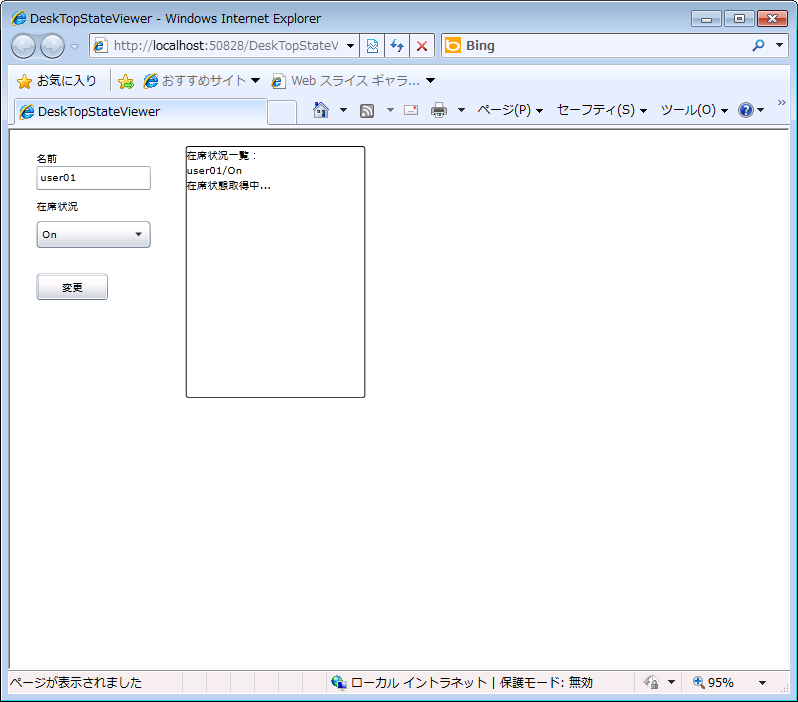
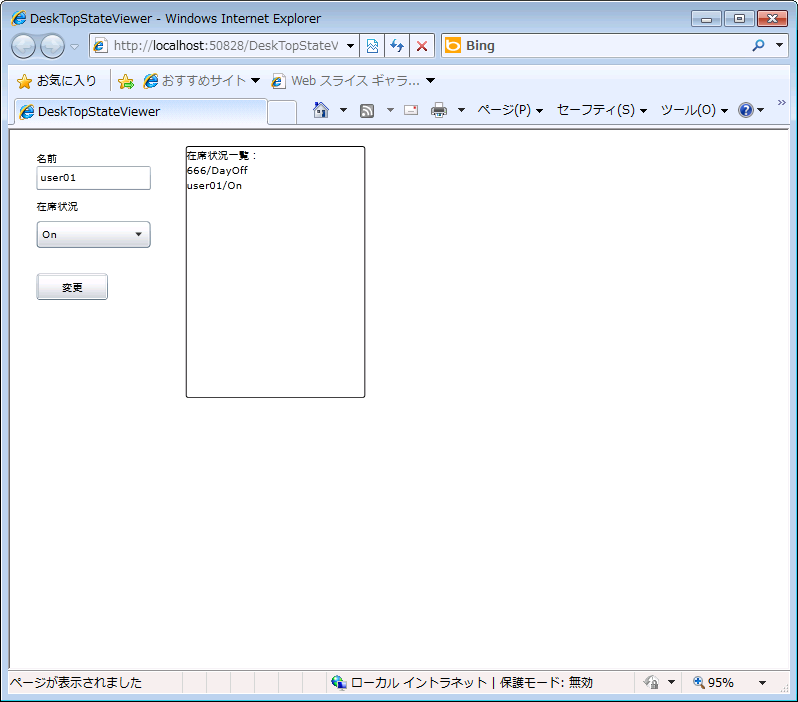
双方向通信によって、自分が送信した既定のステータスはサーバーから通知されてきますが、初期表示ではサーバーからの通知を待たず、送ったステータスを表示しておきます。

|  |
| --- |
| public partial class MainPage : UserControl  {  …  DeskTopStateService.DeskTopStateType currentState = DeskTopStateService.DeskTopStateType.On;  public MainPage()  {  InitializeComponent();  this.comboBox1.Items.Add("選択してください");  Type enumType = typeof(DeskTopStateService.DeskTopStateType);  var fields = from field in enumType.GetFields()  where field.IsLiteral  select field;  foreach (System.Reflection.FieldInfo item in fields)  {  this.comboBox1.Items.Add(item.GetValue(enumType).ToString());  }  this.comboBox1.SelectedItem = currentState.ToString();  this.textBox1.Text = "user01";  proxy.SetStateAsync(this.textBox1.Text, currentState);  this.label1.Content = "在席状況一覧：" + Environment.NewLine + this.textBox1.Text + "/" + currentState.ToString() + Environment.NewLine + "在席状態取得中...";  … |

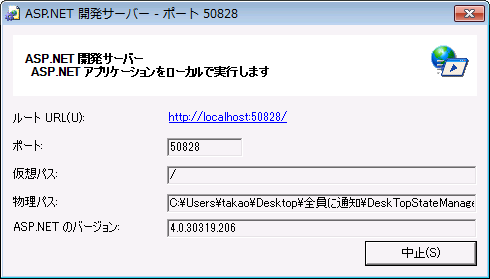
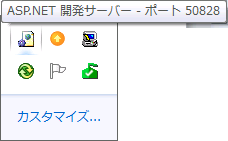
また、[変更]ボタンのクリックで在席状態を変更できるようにしておきます。

|  |
| --- |
| private void button1\_Click(object sender, RoutedEventArgs e)  {  Type enumType = typeof(DeskTopStateService.DeskTopStateType);  System.Reflection.FieldInfo item = enumType.GetField(this.comboBox1.SelectedValue.ToString());  currentState = (DeskTopStateService.DeskTopStateType)item.GetValue(enumType);  proxy.SetStateAsync(this.textBox1.Text, currentState);  } |

F5を押して動作を確認してみましょう。

うまく動かない場合は、ASP.NET開発サーバーを再起動します（一度終了し、再度F5キーで起動させます）。



※タスクトレイのASP.NET開発サーバーのアイコンを右クリックすると中止できます。

一連の挙動を確認したら、VisualStudioをもう一つ起動して、同じソリューションを開き同じようにF5をクリックします。

（サーバーアドレスをハードコーディングしているので、F5でWCFサービスはもう一つ起動しますが、新しく起動したWCFサービスでなく、ハードコーディングしたWCFサービスを利用したデバッグが行えます）

