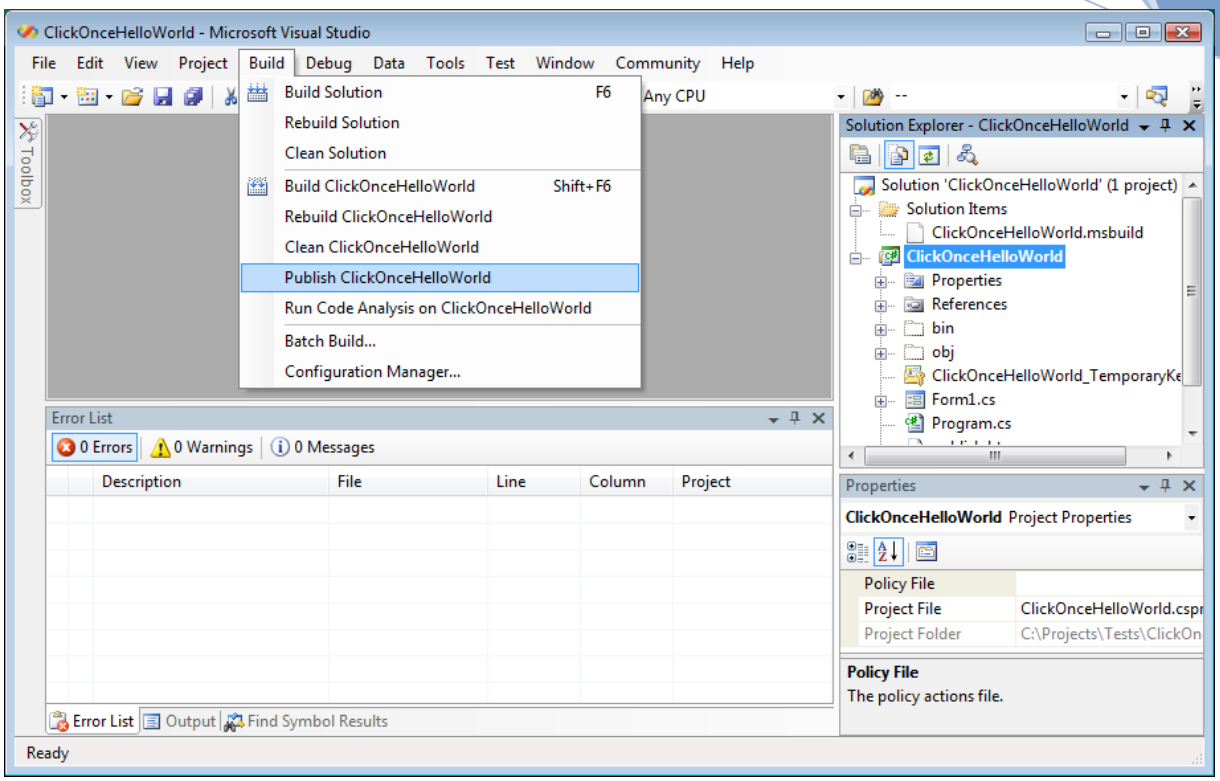
Continuous Integration z ClickOnce

To, że *Continuous Integration* (z ang. ciągła integracja) to proces polegający na stałym, możliwie częstym konsolidowaniu (kompilacji, testowaniu, publikacji) wyników pracy programistycznej możesz przeczyta w moim artykule w Software Developers Journal 06/2007. Tekst ten jest również dostępny na stronie internetowej wydawnictwa. Opisałem w nim podstawowe kroki prowadzące do pełnego zastosowania procesu CI na przykładzie aplikacji webowej.

Ostatni krok w dostarczeniu strony internetowej użytkownikowi końcowemu to zadanie często sprowadzające się do skopiowania wyników pracy do określonego folderu na serwerze WWW. W przypadku programu okienkowego trzeba się bardziej nagimnastykować. Należy na przykład utworzyć program instalacyjny, który trzeba jakoś dostarczyć użytkownikowi. Technologią łączącą te dwa podejścia jest ClickOnce Deployment. Za pomocą ClickOnce można automatycznie opublikować aplikację okienkowo na stronie WWW. Aplikacja taka może być zainstalowana wprost ze strony internetowej. Podczas każdego startu aplikacja taka pamiętając „skąd" została zainstalowana sprawdzi czy dostępna jest nowa wersja i jeśli użytkownik sobie tego życzy zainstaluje ją. Aplikacja dystrybuowana poprzez ClickOnce uruchamiana jest domyślnie z ograniczonymi prawami (dotyczącymi na przykład dostępu do plików). Nie jest ona również instalowana w katalogu *Program Files*, ale dość głęboko w prywatnym katalogu użytkownika, który dokonał instalacji. Jeśli odpowiada ci taka forma dystrybucji to warto postarać się żeby odbywała się ona automatycznie. Dobrym pomysłem (w pewnych okolicznościach) może być zasada, że nowa wersja publikowana jest po każdym przekazaniu kodów do systemu kontroli wersji. Nic nie stoi na przeszkodzie by publikację poprzedzić dokładnymi testami albo/i wygenerować dokumentacje. Dobre wprowadzenie znajduje się we wspomnianym już artykule. Teraz zajmiemy się tylko automatyczną publikacją poprzez ClickOnce.

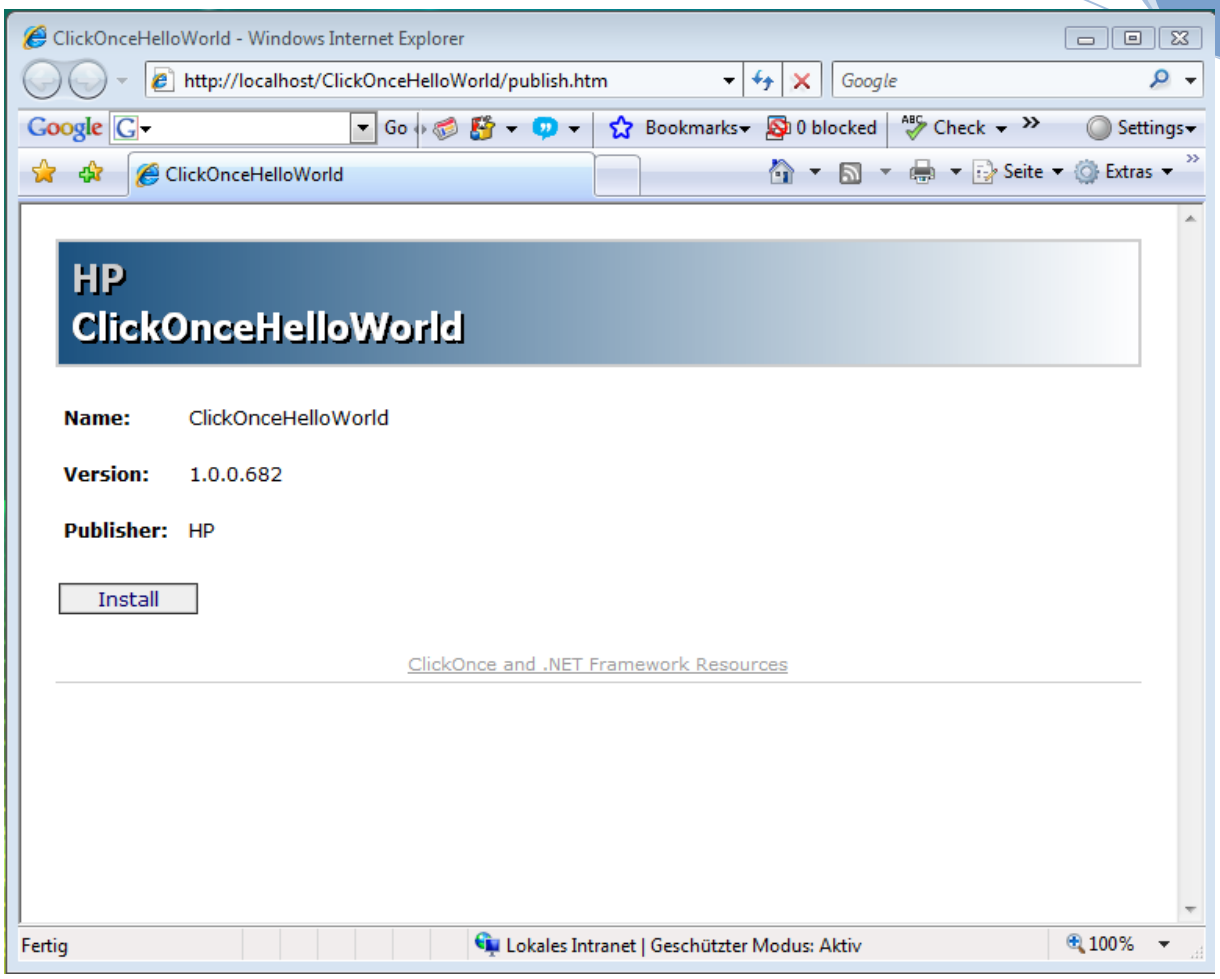
Tworzymy dowolną aplikację okienkową w Visual Studio. Nazwijmy ją *ClickOnceHelloWorld*. Aplikację taką można opublikować wprost ze środowiska wybierając z menu *Build->Publish ClickOnceHelloWorld* tak jak to pokazano na rysunku 1.



Rysunek 1. Publkacja ClickOnce w VS2005

Powinniśmy mieć zainstalowany serwer IIS który będzie stanowił platformę dla naszych publikacji. Po potwierdzeniu domyślnych ustawień otwieramy przeglądarkę podając http://localhost/ClickOnceHelloWorld/publish.htm

Patrz rysunek 2.



Rysunek 2. Opublikowana strona

Po wybraniu Install nasz świeżutki program zostanie zainstalowany na naszym komputerze. Prawda, że proste? Spróbujmy zautomatyzować te kroki.

Aby tego dokona musimy być w stanie dokonać publikacji z poziomu znaku zachęty systemu operacyjnego (command line). Można to zrobi w bardzo prosty sposób.

Msbuild /target:Publish

Voila! Pierwsza publikacja dokonana. Ale nie tak szybko. Pliki publikacji, które znajdują się w podkatalogu *bin/Debug/ ClickOnceHelloWorld/publish.htm* muszą znaleźć się jeszcze na serwerze WWW. Można je tam skopiować ręcznie. Niestety plik *publish.htm* nie jest tworzony automatycznie i brakuje numeracji wersji w podkatalogu *ClickOnceHelloWorld*. Ostatni problem łatwo rozwiążemy przekazując numer wersji poprzez parametr.

msbuild /target:Publish /p: ApplicationVersion=1.0.0.2

No tak, ale jak serwer CI będzie w stanie nada kolejny numer wersji naszej aplikacji? Jednym z rozwiązań jest nadanie jej numeru zgodnego z numerem rewizji na serwerze SVN.

Dotarliśmy, więc do punktu, w którym musimy przygotować sobie „front robót”. Po pierwsze tworzymy miejsce na naszą aplikację na serwerze SVN i przygotowujemy serwer CruiseControl.NET

(patrz wspomniany wcześniej artykuł z SDJ). Kiedy wszystko będzie już gotowe tworzymy skrypt msbuild, który przeprowadzi za nas cały proces integracji.

Załóżmy że nasze repozytorium znajduje się pod tym adresem *http://svn-server/svn/ClickOnceHelloWorld/trunk* w takim przypadku pierwsza wersja naszego skryptu msbuild będzie wyglądał następująco.

```
<Project DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <UsingTask AssemblyFile="lib\MSBuildCommunityTasks\MSBuild.Community.Tasks.dll"
TaskName="MSBuild.Community.Tasks.Subversion.SvnInfo"></UsingTask>

  <PropertyGroup>
    <ApplicationVersion Condition=" '$(ApplicationVersion)' == ''
">1.0.0.</ApplicationVersion>
    <RevisionNumber Condition=" '$(RevisionNumber)' == '' ">0</RevisionNumber>


  <Target Name="Build">
    <SvnInfo RepositoryPath=" http://svn-server/svn/ClickOnceHelloWorld/trunk">
      <Output TaskParameter="Revision" PropertyName="RevisionNumber" />
    </SvnInfo>

    <MSBuild Targets="Publish" Projects="ClickOnceHelloWorld.sln" ContinueOnError="false"
Properties="ApplicationVersion=$(ApplicationVersion)$(RevisionNumber)"/>
  </Target>

</Project>
```

Pierwszym krokiem jest import zadania *SvnInfo* z pakietu MSBuild Communitz Tasks (który zgodnie z wytycznymi z artykułu z SDJ umieszczamy w podkatalogu *lib* naszego rozwiązania (solucji). Deklarujemy zmienne *ApplicationVersion* i *RevisionNumber*, z których złożymy właściwy numer wersji naszej aplikacji. *ApplicationVersion* to domyślnie 1.0.0. pozostaje dowiedzieć się, jaki numer ma ostatnia rewizja. W celu *Build* posłużymy się zadaniem *SvnInfo* który za pomocą parametru Revision wypełni naszą zmienną RevisionNumber. Kiedy wszystko jest gotowe wystarczy wywołać *MSBuild* z celem Publish i odpowiednimi parametrami.

Wystarczy spojrzeć do katalogu *ClickOnceHelloWorld.publish* by przekonać się, że nasza publikacja jest zaopatrzona we właściwe numery wersji i jest gotowa do umieszczenia na serwerze.

Do zrealizowania tego zadania użyjemy zadania *Copy*, wcześniej deklarując jednak ścieżkę docelową w części PropertyGroup

```
<DeploymentFolder>c:\Inetpub\wwwroot\ClickOnceHelloWorld</DeploymentFolder>
```

Oraz nową *ItemGroup* z definicją zbioru plików do skopiowania

```
  <ItemGroup>
    <DeploymentSourceFiles
Include="ClickOnceHelloWorld\bin\$(Configuration)\ClickOnceHelloWorld.publish\**\*.*" />
  </ItemGroup>
```

Po czym uzupełniamy cel build o kopiowanie plików.

```
<Copy SourceFiles="@(DeploymentSourceFiles)"
      DestinationFiles="@(DeploymentSourceFiles-
>'$(DeploymentFolder)\%(RecursiveDir)%(Filename)%(Extension)')"/>
```

Zmieńmy teraz coś w kodzie naszego programu i umieśćmy nowa wersję na serwerze SVN. Po czym zabawmy się w serwer CI i wystartujmy nasz plik *msbuil*. Ponowne uruchomienie

CliclOnceHelloWorld poprzez menu start. Zaowocuje ściągnięciem i zainstalowaniem najnowszej wersji naszego oprogramowania. Brawo!

Pozostaje jedna drobnostka. Niestety na stronie http://localhost/ClickOnceHelloWorld/publish.htm

Widnieje wciąż stary numer wersji (mimo, że w środku czeka najnowsze oprogramowanie). By skorygować to drobne niedociągnięcie użyjemy wyrażeń regularnych.

Kopiujemy plik *publish.htm* do katalogu naszej polucji. Edytujemy go i znajdujemy miejsce gdzie widnieje numer wersji 0.0.0.1 i zamieniamy go na ciąg znaków *ApplicationVersion* i dodajemy następujący urywek na końcu naszego pliku msbuil.

```
    <Copy SourceFiles="ClickOnceHelloWorld\publish.htm"
DestinationFiles="$(DeploymentFolder)\publish.htm"/>
    <FileUpdate Files="$(DeploymentFolder)\publish.htm"
                IgnoreCase="true"
                Multiline="true"
                Singleline="false"
                Regex="ApplicationVersion"
                ReplacementText="$(ApplicationVersion)$(RevisionNumber)"/>
```

Najpierw kopiujemy nasz plik wzorcowy do katalogu docelowego, a później zmieniamy jego treść za pomocą zadania *FileUpdate* wyszukując za pomocą *Regex* wyrażenia *AplicationVersion* i zamieniając je na *ReplacementTest*, który jest znanym nam już numerem wersji. *FileUpdate* jest częścią MSBuild Communitzy taska I tako taki musi zostać zaimportowany.

```
<UsingTask AssemblyFile="lib\MSBuildCommunityTasks\MSBuild.Community.Tasks.dll"
TaskName="MSBuild.Community.Tasks.Subversion.SvnInfo"></UsingTask>
```

Wszystko gotowe można teraz skonfigurować CruiseControl.net tak by używał naszego skryptu. Jak tego dokonać opisałem we wspomnianym wcześniej artykule.

Dzięki tym prostym krokom, od tej pory po każdym przekazaniu nowej wersji do SVN otrzymamy świeżutką stronę kopię ClickOnce naszego oprogramowania.