Subtext Installation Specification

Getting from No Subtext to Subtext

Phil Haack  
April 2005

**Summary:** A proposal and specification for the Subtext installation process.

# Introduction

Installation of a web-based blogging engine is tricky problem as there are many variables involved. For example, some users may only have file access to their hosting provider via FTP and SQL access via a web-based console. This would make an MSI installation process unworkable for this scenario. This document outlines a proposal (and potential alternatives) for the Subtext installation process.

# Zip File and Web Database Install

## Description

This method is inspired by DotNetNuke and Wordpress.

In this version, we simply supply a zip folder with instructions. The user unzips the files to a directory, configures IIS to point to that directory, modifies the connection string (specifying a user with permissions to create and drop objects), and then navigates to the root page. The root page will detect that an installation is necessary and redirect to Install.aspx, unless the Connection string is invalid. In that case, it will display a friendly error message with instructions.

Install.aspx is a web-based wizard that checks to see if the connection string has the requisite properties, checks the existence of the blog\_version table, if such table exists, it then checks the stored version in that table with a version info file. Afterwards, it walks the user through the installation process. In the end, it instructs the user to lower the privileges of the user.

## Justification

For users of hosting providers, a website will already have been set up, so they merely have to copy the files over, adjust the connection string, and they’re on their way.

For users hosting their own, they will easily be savvy enough to configure a website.

## Options

We could provide a simple application to help the user configure this initial setup.

One option is after the user unzips the files, the user would run an application that helps the user configure the connection string. Afterwards the app opens a browser to the Install.aspx page.

## 

## Detecting The Need For Installation

To detect that an installation is necessary, the system will try to check the existence the blog\_version table. If it doesn’t exist, then we’re in for an installation. If it does exist, then we…

## Detect the Need For an Upgrade

On Application start, the system will store the values within blog\_version into the cache. On each request, it will check to see that the Subtext.Framework version match the value stored in blog\_version. If they don’t match, it’s still possible we don’t require an upgrade. If it’s a build or revision number change, then we simply ignore it. If it’s a minor version change, we then check to see if there are schema changes necessary.

At this point, an embedded xml resource will be checked to see if there are any schema changes. It will be up to the development team to make sure that this file is maintained. It might look like…

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

<Versions>

<Version number="0.95">

<ContainsSchemaChange>true</ContainsSchemaChange>

<UpgradeScript>InstallSubtextSchema095.sql</UpgradeScript>

</Version>

</Versions>

At this point, we find the node that matches the version stored in blog\_version and the node that matches the Subtext.Framework version number. This will tell us every script that needs to run in order to upgrade the installation of Subtext. Since a schema upgrade is necessary, the user is redirected to Install.aspx and the upgrade wizard.

The scripts themselves are embedded resources. This way, if someone copies a new version of the Subtext.Framework assembly into the the bin directory, the installation process will occur. We might consider moving the scripts into a separate assembly.

Once an upgrade is complete, the blog\_version record is updated to the current version.

# Custom Installation Application

This idea takes the previous idea, but embeds the zip file as a resource within a custom application. The application gives the user the option to specify a directory to unpack the zip file. It can also create a website or virtual directory pointing to the location of the just unzipped files. It would also help configure the connection string. Once the connection string is configured, it opens a browser to Install.aspx.

# WiX MSI Installer

Clearly, the idea of using an MSI installer package is compelling. We could offer one easily for setting up Subtext in a dev environment or for users who control their own server. The drawback is for users who do not have interactive access to their hosting server to install software.

Likewise, I’m stuck on getting this implemented.

# Conclusion

I think the Zip File method might be the best thing going.