MFPluggerService

MFPluggerService : Background and Concept:

During the programming life in different projects, we developed multiple windows services for doing scheduled & batch processing jobs. This requires to code for creating a windows service, which is ever-running, and execute some business logic with it at a periodic interval (say in once every 30 mins or once in every day etc..). Each such implementation involve developer to learn and create a windows service and maintain its code, apart from coding the business logic for specific need.

With **MFPluggerService**, the concept is to remove the burden from the developer from developping the core windows service code every time, and just to focus on the business need. The **MFPluggerService** works as a service framework and can accept the assemblies (.DLLs) as hot plugins. The plugins are auto detected by the **MFPluggerService** framework and loaded into it dynamically along with configuration parameters (e.g. when to start execution and the interval time etc). Then **MFPluggerService** framework takes the responsibility of creating the instance of the business assembly and invoke appropriate method to initiate the execution.

MFPluggerService : Setup Information:

The package comes with a Windows Service Installer, that installs **MFPluggerService** on the machine. The default installation directory is "`[system drive]\Program Files\mindfire solutions\MFPluggerService\`". The service is configured to auto start after installation with a scheduled timer of 30 seconds interval. This means, the parent service will check status of each individual plugin and invoke them at every 30 seconds interval (only if they are waiting to be executed).

The settings of the parent service framework can be altered by editing the **MFPluggerService.exe.config** file placed within the root directory which will take effect after restarting the service from the windows service manager.

Listing 1: MFPluggerService.exe.config sample.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <appSettings>
        <add key="PluginsFolderName" value="Plugins" />
        <add key="PluginInfoFileName" value="PluginInfos.xml" />
        <!-- Service timer loop interval in secs -->
        <add key="TimerInterval" value="30" />
    </appSettings>
</configuration>
```

The plugins developed for **MFPluggerService** need to be placed inside "`[system drive]\Program Files\mindfire solutions\MFPluggerService\`**`Plugins`**" directory when the service is running, to get auto detected by service and get registered with service framework. How to create a plugin for **MFPluggerService** is described below.

MFPluggerService : Creating and deploying plugin:

Now the developer only need to focus and write code for the scheduled & batch jobs to be executed for business need as a .net class library project, which will be a plugin to the service framework. The plugin class should inherit from `MFPluggerService.`**`IMFServicePlugin`** interface provided within **MFServicePlugin.dll**.

The interface contains only one method **ExecutePlugin()**, which should be implemented by the custom plugin as the entry point method. This is the method which will be called by the core service framework at the scheduled interval. The execution interval can be configured by the plugin's config file, with a name in the format **MFDemoPlugin.dll.config**, assuming MFDemoPlugin is the name of the plugin assembly.

A sample plugin project code is provided in zipped format along with the installation package for reference. This is developed with .net framework 3.5 and needs Visual Studio 2008 or higher for opening.

Listing 2: MFDemoPlugin.dll.config sample.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <appSettings>
        <add key="Name" value="MFDemoPlugin" />
        <add key="Description" value="Writes the execution time MFDemoPlugin.txt file in C drive." />
        <add key="InitialLoadTime" value="01/19/2010 04:45:00 PM" />
        <!-- Execution time interval in mins e.g. 120 = 2 hrs below -->
        <add key="ExecutionInterval" value="120" />
    </appSettings>
</configuration>
```

For deploying the plugin and associating it with the service, one need to put the plugin assembly and the related config file within **Plugins** folder. The service is responsible to detect and start executing from there as per the schedule.

The service framework is capable to load and manage multiple plugins with different application domains, so to keep each one independent of other for performance monitoring and application security. All the registered plugin information is kept inside **PluginInfos.xml** file at the installation root, which can be altered later for managing individual the plugin execution cycle. A node of the **PluginInfos.xml** file contains the data in the following format.

Listing 3: PluginInfos.xml node format.

```
  <Plugin FileName="MFDemoPlugin.dll">
    <Name>MFDemoPlugin</Name>
    <Description>Writes the execution time MFDemoPlugin.txt file in C drive.</Description>
    <NextLoadTime>1/17/2010 9:13:09 PM</NextLoadTime>
    <ExecutionInterval>120</ExecutionInterval>
    <IsActive>True</IsActive>
    <LastLoadedTime>1/17/2010 7:13:09 PM</LastLoadedTime>
  </Plugin>
```

Hope this framework will help you to focus on the business aspect of your service need without concerning the service development management.

~~~~~~

Developed by: Abhishek Bhadoria & Sumit Dhal
Concept by: Chinmoy Panda
Mindfire Solutions
~~~~~~