Tasker

Система исполнения заданий в виде Службы Windows. Описание.

1. Цель документа

Настоящий документ содержит необходимые сведения по запуску, отладке, расширению функционала, а также описанию работы приложения.

2. Системные требования

- Microsoft Windows 7, 8, 8.1, Server 2008R2, Server 2012
- Microsoft SQL Server 2012 и выше
- Microsoft .Net Framework 4.5

3. Используемые библиотеки:

- Castle.Core версия 3.3.3
- Castle.Windsor версия 3.4.0
- EntityFramework версия 6.1.3

4. Описание применения

Настоящее приложение содержит механизм выполнения типизированного задания, параметры которого хранятся в БД.

Типы заданий, реализованных в текущей сборке:

- **Createfile** - Выдерживание паузы 10 секунд с момента получения задания, создание пустого файла в файловой системе с именем, содержащимся в параметре filename. Директория сохранения указывается в файле конфигурации приложения App.Config с ключем PathToSave.
- **Emailsender** - отправка E-mail сообщения по указанным реквизитам в параметрах: *Sender* – e-mail отправителя, *Recipient* - получатель, *Subject* - Тема письма, *Body* - Текст письма. Параметры, необходимые для подключения к SMTP-серверу содержаться в файле конфигурации приложения App.Config с ключами: *SMTPHost* - адрес SMPT-сервера, *SMTPPort* - порт SMTP-сервера, *SMTPUser* - имя пользователя, SMTPPassword - пароль пользователя, *SMTPUseSSL* - признак использования *SSL* для подключения (true - *И*спользовать/ false - не использовать).

Для запуска приложения используется файл Tasker.Runner.exe. Запуск может быть произведен в нескольких режимах:

- Служба Windows (параметр запуска: -Service).
- Установка службы Windows (параметр запуска: -Install).
- Удаление службы Windows (параметр запуска: -Uninstall).
- Консольное приложение (запускается без параметров).

5. Описание процесса создания отдельного модуля, реализующего тип задачи. Руководство разработчика

Для добавления нового типа задач необходимо создать сборку, в которой реализованы интерфейсы Tasker.Core.IModule и Tasker.Core.IJobType (Tasker.Core.dll). После компиляции сборки необходимо скопировать .dll файл в директорию Tasker.Runner\bin\modules. При запуске приложения все сборки из данной директории будут загружены.

Tasker.Core.IModule - определяет модуль. Содержит метод инициализации модуля и его валидации. Данные методы вызываются при старте приложения. При инициализации модуля необходимо зарегистрировать в контейнере реализации типов задач (Tasker.Core.IJobType) используя метод RegisterJobType<T>();

```
public interface IModule
    {

        string Id { get; }

        string Name { get; }

        string Description { get; }

        void InitModule();

       void ValidateModule();
}
```

Пример реализации:

```
namespace Tasker.JobType.ExampleModule
{
    using System.Configuration;
    using System.Linq;
    using Castle.Windsor;
    using Core;

    public class Module : IModule
    {
        private readonly IWindsorContainer container;

        public Module(IWindsorContainer container)
        {
            this.container = container;
        }

        public string Id
        {
            get { return "Tasker.JobType.ExampleModule"; }
        }

        public string Name
        {
            get
            {
                return "JobType.ExampleModule";
            }
        }

        public string Description
        {
            get { return string.Empty; }
        }

        public void InitModule()
        {
            this.container.RegisterJobType<ExampleJob>();
        }

        public void ValidateModule()
        {
        }
```

Tasker.Core.IJobType - определяет тип задачи. Содержит метод Execute, в котором описываются необходимые инструкции, для выполнения задачи.

```
namespace Tasker.Core
{
    public interface IJobType
    {
        string Description { get;  }

        void Execute(JobParameters parameters);
    }
}
```

Пример реализации:

```
namespace Tasker.JobType.ExampleJob
{
    using System;
    using System.Configuration;
    using System.IO;
    using System.Threading;
    using Core;

    public class ExampleJob: IJobType
    {
      public string Description
       {
          get { return "Описание типа задачи."; }
       }

       public void Execute(JobParameters parameters)
       {
          // Инструкции исполнения

       }
    }
}
```

6. Описание функциональных ограничений.

- Нет возможности передавать в параметрах исполнения задач знаки ":" и ";" из-за особенностей сериализации.
- В режиме консольного приложения устанавливать и удалять службу можно только с правами администратора