С целью экономии времени файл настроек находиться в статически привязанном маршруте: "C:\Temp\Settings.xml";

Сохранять настройки в реестр посчитал неуместным.

В проекте есть 2 сервиса пустышки для проверки.

Эскизное проектирование под задачей.

Сервис «Watch Dog»

Необходимо создать сервис под ОС WINDOWS для слежения за работоспособностью других сервисов и при определенном условии перезапускать (restart) нужный сервис.

Контроль осуществляется двумя различными способами:

1. Для WEB сервисов: с заданной периодичностью посылаем GET запрос сервису на определенный адрес и порт (URL) и на основе полученного ответа, например статус 200 (OK), принимаем решение о работоспособности сервиса.
2. Для консольного приложения: с заданной периодичностью проверяем время создания файла-флага, который сервис регулярно перезаписывает и если время создания файла отличается от текущего времени не более чем на N-количество минут, то считаем что сервис работает корректно.

Для каждого варианта проверки необходимо учитывать параметры:

* Тип сервиса (WEB/Console)
* Название сервиса
* Адрес (для WEB), файл-флаг и путь к нему (для Console)
* Количество попыток, после которых считаем что сервис «завис»
* Интервал проверки (различный для каждого сервиса)
* Статус (для WEB) который считается корректным ответом для сервиса
* «Возраст» файла-флага в минутах (для Console)

После запуска сервиса, он через заданные интервалы времени «опрашивает» сервисы и в случае некорректного ответа за определенное количество попыток, рестартует «зависший».

**Проектирование.**

**Модель:**

//Класс параметров для сервисов WEB и Console

public class ServiceParameters

{

public int ServiceType { get; set; } //Тип сервиса (WEB/Console)

public string ServiceName { get; set; } //Название сервиса

public string ServicePath { get; set; } //Адрес (для WEB), файл-флаг и путь к нему (для Console)

public int AttemptsQuantity { get; set; } //Количество попыток, после которых считаем что сервис «завис»

public int CheckInterval { get; set; } //Интервал проверки (различный для каждого сервиса)

public int CorrectState { get; set; } //Статус (для WEB) который считается корректным ответом для сервиса, «Возраст» файла-флага в минутах (для Console)

}

//Для быстрой сериализации/десериализации создадим клас объединяющи параметры для обоих сервисов

[Serializable]

public class Services

{

public ServiceParameters WEBserviceParams = new ServiceParameters();

public ServiceParameters ConsoleServiceParams = new ServiceParameters();

}

**Контроллер1 (установка обновление параметров):**

**1. Оконное приложение** для управления параметрами сервиса **ServiceParameters,** параметры считываются и сохраняться в **xml файл.**

**Контроллер2 (сервис проверки):**

**1. Модуль считывания из** **xml файла параметров сервиса,** запускается при запуске сервиса и периодически с перепроверки (для образца взял 1 минуту) перечитывает параметры;

**2. Модуль проверки Web сервиса.** Процесс отправляющий через **dCheckInterval GET** запрос на Адрес (для WEB: **ServicePath**) проверяющий статус полученного ответа и в случае **dAttemptsQuantity** ответов не соответствующих **dCorrectState** перезапускающий сервис **sServiceName;**

**3. Модуль проверки Console сервиса.** Процесс проверяет время создания файла  **sServicePath** через **dCheckInterval** интервал если время создания не более **dCorrectState** в течении **dAttemptsQuantity** ответов то перезапускаем сервис **sServiceName;**