Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Уфимский Государственный Авиационный Технический Университет»

Кафедра «Вычислительной математики и кибернетики»

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Процент выполнения работы |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Номер учебной недели | | | | | | | | | |

Курсовая работа

по дисциплине «Операционные системы»

Тема

Служба Windows

Выполнил: *студент группы ПРО-201в*

*Доронин С.Г.*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(подпись)*

Руководитель: к.т.н.,

*кафедры ВМиК*

*Карамова Л.М.*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(дата и подпись)*

Уфа 2013

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Уфимский Государственный Авиационный Технический Университет»

Кафедра «Вычислительной математики и кибернетики»

Задание

на курсовую работу по дисциплине «Операционные системы»

Студент Доронин С.Г. Группа ПРО-201вКонсультант Фридлянд А.М.

1. Тема курсовой работы Служба Windows

2. Основное содержание: Разбор работы алгоритма службы для MS Windows и проверка на программе, написанной для данной курсовой работы.

3. Требования к структуре и оформлению курсовой работы

3.1. Курсовая работа должна содержать следующие разделы:

Введение

Глава 1. Постановка задачи

Глава 2. Основные понятия служб

Глава 3. Область применения служб

Глава 4. Задача с решением

Глава 5. Написание алгоритма на Java

Глава 6. Обзор работы программы

Глава 7. Текст программы

Заключение

Список использованной литературы

4. Источники информации

Книги, учебники, учебные пособия, материалы сети Интернет.

На все источники, приведенные в списке литературы, должны быть ссылки по тексту курсовой работы.

Дата выдачи задания \_\_\_\_\_\_\_\_\_ Дата окончания работы

Руководитель  (дата и подпись)

СОДЕРЖАНИЕ

[Введение 4](#_Toc358736675)

[1 Постановка задачи 5](#_Toc358736676)

[2 Основные понятия служб 6](#_Toc358736677)

[3 Область применения служб 7](#_Toc358736678)

[4 Задача с решением 8](#_Toc358736679)

[5 Написание алгоритма на Java 9](#_Toc358736680)

[6 Обзор работы программы 11](#_Toc358736681)

[7 Текст программы 13](#_Toc358736682)

[Заключение 18](#_Toc358736683)

[Список литературы 19](#_Toc358736684)

# Введение

Службы Windows (англ. Windows Service, сервисы) — приложения, автоматически запускаемые системой при запуске Windows и выполняющиеся вне зависимости от статуса пользователя. Имеет общие черты с концепцией демонов в Unix.

В большинстве случаев службам запрещено взаимодействие с консолью или рабочим столом пользователей (как локальных, так и удалённых), однако для некоторых сервисов возможно исключение — взаимодействие с консолью (сессией с номером 0, в которой зарегистрирован пользователь локально или при запуске службы mstsc с ключом /console).

Существует четыре режима для сервисов:

* Запрещён к запуску;
* ручной запуск (по запросу);
* автоматический запуск при загрузке компьютера;
* обязательный сервис (автоматический запуск и невозможность (для пользователя) остановить сервис).

# 1 Постановка задачи

* Изучить теорию сервисов;
* разработать алгоритм для службы генерации отчета об MP3 файлах;
* написать программу на языке Java, которая бы при заданных пользователем параметрах, генерировала бы отчет об MP3 файлах.

# 2 Основные понятия служб

Службы и их атрибуты могут быть изменены в консоли управления MMC:

Панель управления Windows -> «Administrative Tools» -> «Services».

Управление запуском служб при старте Windows.

Список служб находится в ветке Реестра Windows «HKEY\_LOCAL\_MACHINE \ SYSTEM \ CurrentControlSet \ Services». Значения параметра «Start» имеют тип «REG\_DWORD» и могут принимать значения:

0 - Низкоуровневые драйверы, например драйверы дисков, которые загружаются на самом раннем этапе загрузки – загрузки ядра;

1 - Драйверы, которые загружаются после инициализации ядра ОС;

2 - Службы, которые должны быть загружены Диспетчером Управления Сервисами

(равен параметру - "Авто");

3 - Службы, запускаемые Диспетчером Управления Сервисами,

только в случае получения явной инструкции на загрузку (равен параметру - "Вручную");

4 - Службы, которые не загружаются (равен параметру - "Отключено").

Службы можно легко создавать путем создания приложений, которые устанавливаются в качестве служб. Предположим, что нужно отслеживать данные счетчика производительности и реагировать на пороговые значения. Можно написать приложение службы Windows, которое будет считывать эти показания, развернуть приложение и приступить к сбору и анализу данных.

# 3 Область применения служб

Службы используются в том случае, когда нужно приложение, в котором не требуется участие пользователя или же требуется автоматический запуск программы при включении компьютера, когда пользователь еще не зашел в систему.

# 4 Задача с решением

Для списка файлов, изображенных на Рис. 1 – Список MP3 файлов, сформировать HTML-файл, который будет выводить исполнителя и название композиции.



Рис. 1 – Список MP3 файлов

Решение.

Чтобы узнать исполнителя и название композиции, нужно прочитать тег-данные файла, для этого воспользуемся библиотекой jid3lib.

Узнав тег данные запишем их в HTML-файл, добавляя css-стили для лучшего внешнего вида.

# 5 Написание алгоритма на Java

Класс PhpusrWrapperListener это оболочка для запуска генератора HTML-страницы. Он имеет методы start() и stop(), которые запускаются соответственно при запуске и остановке службы.

Функция start() создает экземпляр класса MP3Tag2HtmlThread и запускает его в отдельном потоке. Метод run() в классе MP3Tag2HtmlThread, каждую секунду сканирует заданную папку пользователем и генерирует HTML-файл, по MP3 трекам в этой папке.

Функция stop() вызывает у экземпляра класса MP3Tag2HtmlThread метод interrupt(), что приводит к остановке выполнения потока и завершению работы службы.

/\*\*

\* Оболочка для сервиса

\*/

public class PhpusrWrapperListener **implements** WrapperListener **{**

private Thread service**;**

private PhpusrWrapperListener**()** **{}**

@Override

public Integer start**(**String**[]** args**)** **{**

System**.**out**.**println**(**">>Service start..."**);**

service **=** **new** MP3Tag2HtmlThread**(**args**[**0**],** args**[**1**]);**

service**.**start**();**

**return** **null;**

**}**

@Override

public int stop**(**int exitCode**)** **{**

System**.**out**.**println**(**">>Service end..."**);**

service**.**interrupt**();**

**return** exitCode**;**

**}**

@Override

public void controlEvent**(**int event**)** **{**

System**.**out**.**println**(**">>Control event: " **+** event**);**

**}**

public static void main**(**String**[]** args**)** **{**

WrapperManager**.**start**(new** PhpusrWrapperListener**(),** args**);**

**}**

**}**

**…**

@Override

public void run**()** **{**

**try** **{**

**while** **(!**isInterrupted**())** **{**

System**.**out**.**println**(**">>Songs:"**);**

generateHtmlFromSongs**(**getSongsForDir**(**songsDir**));**

Thread**.**sleep**(**1000**);**

**}**

**}** **catch** **(**InterruptedException e**)** **{**

System**.**err**.**println**(**e**.**getMessage**());**

**}**

**}**

# 6 Обзор работы программы

1. Устанавливаем службу в ОС запуская скрипт InstallApp-NT.bat в папке bin.

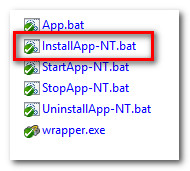


Рис. 2 – Скрипт установки службы

1. Открываем файл wrapper.conf в папке conf, находим строку: «wrapper.app.parameter.1=../music», после равно пишем путь к папке с музыкой для сканирования.

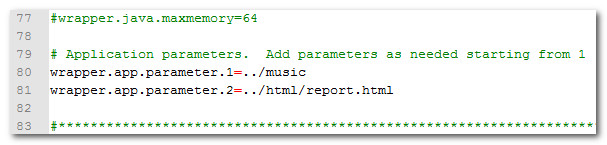


Рис. 3 – Изменение паки для сканирования

1. Запускаем службу скриптом StartApp-NT.bat из папки bin.

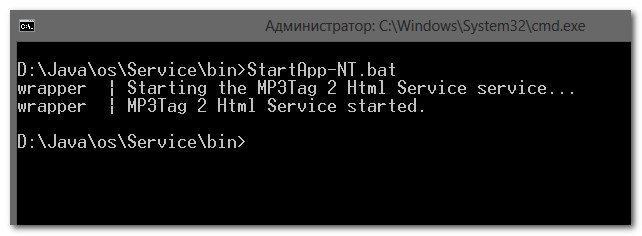


Рис. 4 – Запуск службы

1. В папке html появится файл report.html, который будет содержать информацию об MP3 файлах в заданной папке.

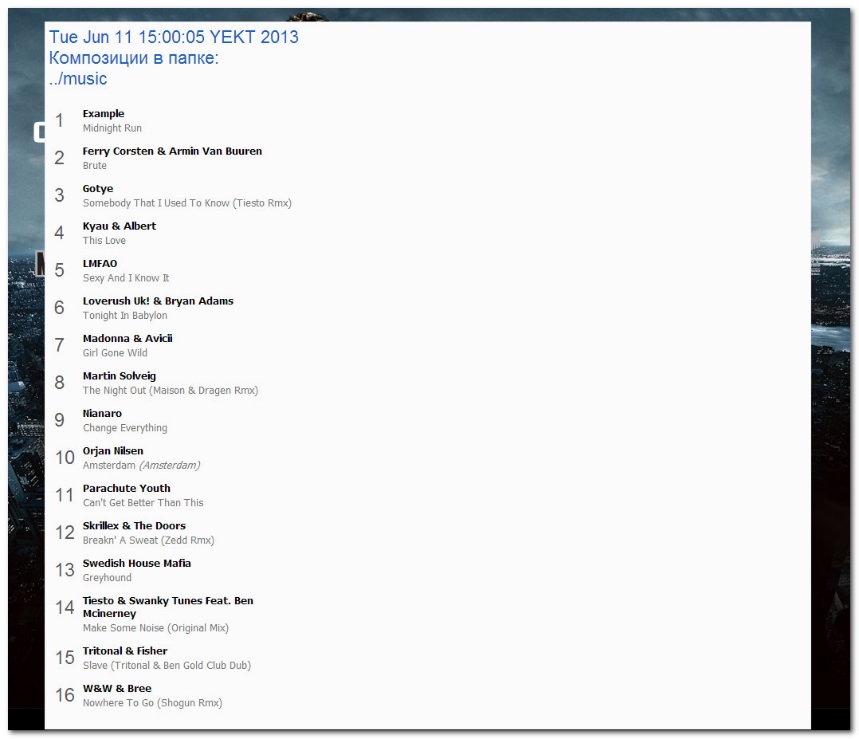


Рис. 5 – Сформированный HTML-файл.

1. Для остановки службы используется скрипт StopApp-NT.bat из папки bin. Для удаления службы – скрипт StopApp-NT.bat.

# 7 Текст программы

Запускающий класс

package com**.**phpusr**.**service**.**wrapper**;**

**import** com**.**phpusr**.**service**.**thread**.**MP3Tag2HtmlThread**;**

**import** org**.**tanukisoftware**.**wrapper**.**WrapperListener**;**

**import** org**.**tanukisoftware**.**wrapper**.**WrapperManager**;**

/\*\*

\* **@author** phpusr

\* Date: 08.06.13

\* Time: 11:35

\*/

/\*\*

\* Оболочка для сервиса

\*/

public class PhpusrWrapperListener **implements** WrapperListener **{**

private Thread service**;**

private PhpusrWrapperListener**()** **{}**

@Override

public Integer start**(**String**[]** args**)** **{**

System**.**out**.**println**(**">>Service start..."**);**

service **=** **new** MP3Tag2HtmlThread**(**args**[**0**],** args**[**1**]);**

service**.**start**();**

**return** **null;**

**}**

@Override

public int stop**(**int exitCode**)** **{**

System**.**out**.**println**(**">>Service end..."**);**

service**.**interrupt**();**

**return** exitCode**;**

**}**

@Override

public void controlEvent**(**int event**)** **{**

System**.**out**.**println**(**">>Control event: " **+** event**);**

**}**

public static void main**(**String**[]** args**)** **{**

WrapperManager**.**start**(new** PhpusrWrapperListener**(),** args**);**

**}**

**}**

Класс генерирующий HTML-файл.

package com**.**phpusr**.**service**.**thread**;**

**import** com**.**phpusr**.**service**.**entity**.**Song**;**

**import** org**.**farng**.**mp3**.**TagException**;**

**import** java**.**io**.\*;**

**import** java**.**util**.**ArrayList**;**

**import** java**.**util**.**Date**;**

**import** java**.**util**.**List**;**

/\*\*

\* **@author** phpusr

\* Date: 09.06.13

\* Time: 18:37

\*/

/\*\*

\* Записывает инф-ю об MP3-файлах в папке, в html-файл

\*/

public class MP3Tag2HtmlThread **extends** Thread **{**

/\*\* Папка с песнями \*/

private String songsDir**;**

/\*\* Имя html-файла \*/

private String htmlFileName**;**

public static void main**(**String**[]** args**)** **throws** InterruptedException**,** IOException**,** TagException **{**

MP3Tag2HtmlThread mp3Tag2HtmlThread **=** **new** MP3Tag2HtmlThread**(**"music"**,** "html/report.html"**);**

mp3Tag2HtmlThread**.**start**();**

Thread**.**sleep**(**5000**);**

mp3Tag2HtmlThread**.**interrupt**();**

**}**

public MP3Tag2HtmlThread**(**String songsDir**,** String htmlFileName**)** **{**

**this.**songsDir **=** songsDir**;**

**this.**htmlFileName **=** htmlFileName**;**

**}**

@Override

public void run**()** **{**

**try** **{**

**while** **(!**isInterrupted**())** **{**

System**.**out**.**println**(**">>Songs:"**);**

generateHtmlFromSongs**(**getSongsForDir**(**songsDir**));**

Thread**.**sleep**(**1000**);**

**}**

**}** **catch** **(**InterruptedException e**)** **{**

System**.**err**.**println**(**e**.**getMessage**());**

**}**

**}**

private static List**<**Song**>** getSongsForDir**(**String dir**)** **{**

File folder **=** **new** File**(**dir**);**

List**<**Song**>** songList **=** **new** ArrayList**<**Song**>();**

**try** **{**

**for** **(**final File file **:** folder**.**listFiles**())** **{**

**if** **(**file**.**isFile**())** **{**

Song song **=** **new** Song**(**file**);**

songList**.**add**(**song**);**

System**.**out**.**println**(**"\t" **+** song**);**

**}**

**}**

**}** **catch** **(**Exception e**)** **{**

System**.**err**.**println**(**e**.**getMessage**());**

**}**

**return** songList**;**

**}**

/\*\* Генерация html-файла из тегов песен \*/

private void generateHtmlFromSongs**(**List**<**Song**>** songList**)** **{**

PrintWriter out **=** **null;**

**try** **{**

out **=** **new** PrintWriter**(new** OutputStreamWriter**(new** FileOutputStream**(**htmlFileName**),** "utf-8"**));**

out**.**println**(**"<html>"**);**

out**.**println**(**"<head><meta charset=\"utf-8\"><link href=\"style/style.css\" type=\"text/css\" rel=\"stylesheet\"/></head>"**);**

out**.**println**(**"<html><body class=\"branding\"><section id=\"main\">"**);**

String**[]** classes **=** **new** String**[]{**"main-holder"**,** "wrap"**,** "white-box home-page"**,** "white-holder"**,** "white-frame"**,** "gray-box"**,** "gray-holder"**,** "column"**};**

**for** **(**String cssClass **:** classes**)** **{**

out**.**print**(**"<div class=\"" **+** cssClass **+** "\">"**);**

**}**

out**.**print**(**"\n<div class=\"topic\"><h2>"**+** **new** Date**()** **+**"<br/>Композиции в папке: <br/>"**);**

out**.**println**(**songsDir **+** "</h2></div>"**);**

out**.**println**(**"<ul class=\"songs-list players-list js-top40-list\">"**);**

int count **=** 0**;**

**for** **(**Song song **:** songList**)** **{**

**if** **(**song**.**getTitle**()** **!=** **null)** **{**

out**.**println**(**"<li class=\"player-in-playlist-holder\">"**);**

out**.**println**(**"\t<div class=\"jp\_container\">"**);**

out**.**println**(**"\t\t<span class=\"number\">"**+(++**count**)+**"</span>"**);**

out**.**println**(**"\t\t<div class=\"jp-title\">"**);**

out**.**println**(**"\t\t\t<strong class=\"title\"><a href=\"#\" title=\"Карточка исполнителя " **+** song**.**getArtist**()** **+** "\">" **+** song**.**getArtist**()** **+** "</a></strong>"**);**

out**.**print**(**"\t\t\t<span>" **+** song**.**getTitle**());**

**if** **(**song**.**getAlbum**()** **!=** **null** **&&** **!**song**.**getAlbum**().**equals**(**""**))** out**.**print**(**" <i>(" **+** song**.**getAlbum**()** **+** ")</i>"**);**

out**.**print**(**"</span>"**);**

out**.**println**(**"\t\t</div>"**);**

out**.**println**(**"\t</div>"**);**

out**.**println**(**"</li>"**);**

**}**

**}**

out**.**println**(**"</ul>"**);**

**for** **(**int i**=**0**;** i**<**classes**.**length**;** i**++)** **{**

out**.**print**(**"</div>"**);**

**}**

out**.**println**(**"\n</section></body></html>"**);**

**}** **catch** **(**IOException e**)** **{**

e**.**printStackTrace**();**

**}** **finally** **{**

**if** **(**out **!=** **null)** **{**

out**.**flush**();**

out**.**close**();**

**}**

**}**

**}**

**}**

Класс работающий с библиотекой тегов MP3-файлов (jid3lib)

package com**.**phpusr**.**service**.**entity**;**

**import** org**.**farng**.**mp3**.**MP3File**;**

**import** org**.**farng**.**mp3**.**TagException**;**

**import** org**.**farng**.**mp3**.**id3**.**AbstractID3**;**

**import** java**.**io**.**File**;**

**import** java**.**io**.**IOException**;**

/\*\*

\* **@author** phpusr

\* Date: 09.06.13

\* Time: 19:53

\*/

/\*\*

\* Музыкальная композиция (Имя файла, Название, Исполнитель, Альбом, Битрейт)

\*/

public class Song **{**

private MP3File mp3File**;**

public Song**(**File file**)** **throws** IOException**,** TagException **{**

mp3File **=** **new** MP3File**(**file**);**

**}**

/\*\* Имя файла \*/

public String getFileName**()** **{**

**return** mp3File**.**getMp3file**().**getName**();**

**}**

/\*\* Название композиции \*/

public String getTitle**()** **{**

**return** getID3Tag**()** **!=** **null** **?** getID3Tag**().**getSongTitle**()** **:** **null;**

**}**

/\*\* Исполнитель \*/

public String getArtist**()** **{**

**return** getID3Tag**()** **!=** **null** **?** getID3Tag**().**getLeadArtist**()** **:** **null;**

**}**

/\*\* Альбом \*/

public String getAlbum**()** **{**

**return** getID3Tag**()** **!=** **null** **?** getID3Tag**().**getAlbumTitle**()** **:** **null;**

**}**

/\*\* Битрейт \*/

public int getBitRate**()** **{**

**return** isNotNull**()** **?** mp3File**.**getBitRate**()** **:** 0**;**

**}**

/\*\* Проверка на null \*/

private boolean isNotNull**()** **{**

**return** mp3File **!=** **null;**

**}**

private AbstractID3 getID3Tag**()** **{**

**if** **(**isNotNull**())** **{**

**return** mp3File**.**getID3v1Tag**()** **!=** **null** **?** mp3File**.**getID3v1Tag**()** **:** mp3File**.**getID3v2Tag**();**

**}** **else** **{**

**return** **null;**

**}**

**}**

@Override

public String toString**()** **{**

**if** **(**isNotNull**())** **{**

**return** getArtist**()** **+** " - " **+** getTitle**()** **+** " (" **+** getAlbum**()** **+** ")"**;**

**}** **else** **{**

**return** getFileName**();**

**}**

**}**

**}**

# Заключение

В соответствии с поставленной задачей в курсовой работе было выполнено следующее:

1. Изучен конкретный раздел операционных систем;
2. разобран алгоритм для службы Windows;
3. разработана программа на Java, реализующая данный алгоритм.

# Список литературы

1. ru.wikipedia.org [Электронный ресурс] // Свободная энциклопедия - http://ru.wikipedia.org/wiki/Службы\_Windows Режим доступа – свободный.
2. www.lvf2004.com [Электронный ресурс] // Статья о службах Windows - http://www.anvir.net/sluzhbyi-windows-servisyi.htm Режим доступа – свободный.
3. habrahabr.ru [Электронный ресурс] // Хабрахабр - http://www.jgraph.com Режим доступа – свободный.