RtorrentManager

Легенда:

|  |  |
| --- | --- |
| Important issues  Current issues  Future issues | New  Assign  ~~Complete~~  **Reject** |

* Global task
  + Task1 [welvet]
  + **Task2 (Исключение при вызове…)**

# Web интерфейс

* Веб сервер (todo посмотреть на jetty)
  + Обработка и логирование всех исключений
  + Авторизация
* Парсеры (парсеры должны создаваться фабрикой по гет запросу и получать в конструкторе все параметры. Строка вида ?do=stop&hash=4S..DW должна запускать парсер stop(String hash). Проще всего сделать это через рефлексию/cglib, тогда не будет проблем при реализации новых парсеров. Выдача результатов должна приводиться в формат ???(xml/json) этой же фабрикой
  + Логика работы с абстрактным парсером
  + Выдача статичного контекста (также необходимо предусмотреть кеширование на стороне клиента)
  + Обработка асинхронных аякс запросов *(todo продумать интерфейс связи jquery и сервера)*
* Дизайн
  + Страница должна быть одна, и она должна быть статической
  + Реализовать css классы для всплывающих окон, кнопок.
  + Поддержка скинов
  + Мультиязычность
* Client side функционал
  + Определить стандарты кодирования jquery запросов, окон. Реализовать базовые асинхронные методы
  + ~~Работа через jquery dataTable [welvet]~~
  + Все настройки сервера должны указываться через веб интерфейс [welvet]

# Rtorrent service

(во время реализации сервиса необходимо смотреть на возможности других торрент клиентов, чтобы интерфейс сервиса можно было безболезненно использовать на других импелементациях *todo при решении задачи «Управление файлами» нужно будет посмотреть как это реализовано в utorrent*)

* Запуск/остановка рторента
  + Класс должен быть имплементацией интерфейса, чтобы в дальнейшем небыло проблем при поддержке utorrenta
* ~~Связь с рторрентом~~
* Управление файлами
* Поддержка utorrent/deluge (необходимо решить, что будет оптимальным для пользователей windows. Utorrent поддерживает связь через )

# Scheduler

* Реализовать/подключить класс шедулера (возможно quartz, хотя он слишком тяжелый)
* Обработка ТоррентСета по расписанию
* Опрос трекереров
* Остановка/запуск рторрента
* Ограничение скорости по расписанию пользователя (запуск/остановка по расписанию)
* Загрузка торрентов в порядке очереди, оперделенной пользователем

# Stand alone client

* Реализовать модуль клиента, работающего на клиентском ПК (клиент должен висеть в трее, в выпадающем меню иметь строки «Запустить/остановить рторрент, добавить торрент, Настройки». Связываться с серверной частью по ??? протоколу с авторизацией. Ассоциировать себя с .торрент файлами в системе)
  + Создание модуля
  + Создание инсталлятора
  + Авторизация на серверной части
  + Автоматическая остановка рторрента при загрузке клиента, и запуск после выключения клиента
* Определить, нужно ли добавить альтернативную возможность управлением рторренМенеджером из стендалоун клиента
* При добавлении торрента клиент должен предоставлять пользователю выбор файлов, в последствии при обновлении торрента системой, закачивать нужно только отмеченые файлы

# Tracker service

* Реализовать поддержку «абстактного» трекера (пользователь /наследующий класс может указать паттерн/границы поиска текста, по которому будет определяться обновлен ли торрент/вышла ли новая версия; пост/гет запросы для авторизации на трекере; и паттерн для поиска ссылки на торрент файл. Вся информация, нужная трекеру от торрент файла, должна храниться в сериализуемом поле торрент, там же должен храниться и идентификатор трекера. *По идее строки для проверки и регексп шаблона должно быть достаточно для своевременного получения торрент файла, однако возможно есть более логичное решение*)
* Релизовать поддержку трекеров (в процессе релизации необходимо решить какой способ добавления абстрактного трекера будет для пользователя оптимальным)
  + Rutracker.org
  + Lostfilm
  + TPB (*возможно это и есть поддержка рсс*, возможно он и не нужен)
* Поддержка рсс

# Others

* Подробно реализовать список задач (необходимо решить какой именно функционал будет в 1й версии)
* Написать ант/мейвен таски для сборки проекта
* Тесты (весь реализуемый функционал (кроме слоя вью) должен иметь функциональные тесты)
  + Реализовать MockTracker
* Логи (Необходимо чтобы пользователь видел, какие ошибки были с момента его последнего визита)
  + Все операции должны сохранять результаты своей работы в логах
  + Написать конфиг логов
  + Логи уровня еррор должны храниться в самом приложении и показываться пользователю после логина. Вероятно нужно добавить отдельный аппендер
* Инициализация приложения, хранение настроек

# Notify service

* Поддержка уведомления через рсс
* Поддержка емейл уведомлений
* Поддержка im
  + Icq
  + jabber