**Servlet API.**

**Задание-7.**

**Почтовый клиент.**

Друзья, добрый день. Я рад представить Вам очередное задание по сервлетам. Это задание мне особенно приятно Вам представлять. Я так сильно хотел его добавить, что оно даже вытеснило предыдущее задание, которое выполняли Ваши предшественники. Ну, не совсем конечно, вытеснило – предыдущая задача только что переехала в тему «Java Server Pages», так что у Вас еще будет шанс встретиться. Но этого задания я ждал уже очень давно, и бесконечно рад, что оно, наконец, появляется.

«Да что-ж, блин, за задание-то скажи уже!» – это Ваш внутренний голос негодует, уже поинмая, что текст задания он увидит только страниц через двадцать.

Ни в коем случае не стану Вас мучать, и сразу же скажу – если коротко – Вам нужно написать почтовый клиент. Но я все таки хотел бы еще пару слов сказать о том, почему же мне так приятно предлагать Вам эту работу.

Дело в том, что где-то в середине нашего курса у нас будет одно или два (а в некоторых группах получалось, что и три (!) – все зависит от скорости усваивания материала, и количества задаваемых вопросов) занятия, которые будет посвящены вопросам архитектуры JEE приложений.

Скажу откровенно – это моя любимая тема. Ни в коем случае не потому, что я считаю себя супер-пупер профессионалом, а потому, что в рамках этих занятий каждый из слушателей находит для себя что-то интересное. Более того – в рамках этих занятий практически нет строгих законов и правил, которые надо зубрить. Почти вся предлагаемая информация – это выраждение собственного мнения и плод личных предпочтений. А собственное мнение имеет один побочный эффект – **с ним почти всегда не согласны другие носители своего собственного мнения**, то есть Вы – дорогие друзья ☺.

Поэтому в рамках этой темы Вы из слушателей и «запоминателей» превращаетесь в людей, которые задают неудобные вопросы, людей, которые не согласны со мной по многим пунктам, людей, которые даже иногда в пылу дискуссии позволяют себе пару фраз чуть покрепче, чем просто – «Извините, а можно вопрос». Серьезно, когда слышишь что-то вроде – «Да что за нах…!» - пусть эта фраза никогда и не договаривается до конца, но все равно думаешь – «О, вот уже ни я один матерюсь в аудитории». Это свидетельствует о том, что занятие приобретает динамику, о том, что группа начинает жить Java-жизнью, и о том – что вот прямо здесь и сейчас рождаются новые профессионалы – будущие тим-лиды и архитекторы. Возможно те архитекторы, которые будут пересказывать текст этого занятия своим коллегам, как пошлый анекдот, а не как принципы архитектуры, ибо не согласны с этой архитектурой ни на грамм. Но навязывание архитектуры и не является главной целью этого занятия, еще раз – это время, когда рождаются новые профессионалы, а у истоков этого рождения – стою я. Вот почему это моя любимая тема ☺ (*Ну почему всегда в пол третьего ночи, если не давать мозгу спать, а заставлять его пистаь что-то, он непременно начинает думать мысли о том, что его хозяин – Бог ☺*).

Но в рамках этой темы будет так же много отрицательных моментов. Я не стану перечислять их все – как никак, у нас для этого два отдельных занятия. Но один момент очень существеннен – эта острая нехватка времени.

И основной нашей задачей будет – победить эту нехватку. И до сих пор она была невыполнима в том числе и потому, что в одном из рассматриваемых примеров **мы вдруг отправляли email**!!!

Все! Все те, кто еще 5 минут назад были начинающими архитекторами, вдруг снова превращались в младецев, которым мама только что распакавала новую погремушку! И пол занятия (а то и все три четверти) мы разговаривали о магии отправки электронного письма на Java! И весь курс вхлам, потому что мы начинаем отставать ☹.

А виноват в этом кто? Правильно – Ваш препод. Потому что за столько лет он не удосужился написать короткий текст о том, как оправить это, так его, письмо. И кажлый раз я скомкано объясняю это прямо на занятии, которое этой теме не посвящено, и стыдливо пряча глаза отправляю Вас в Google, и потом, когда занятие заканчивается, кричу про себя обидные слова, типа – «Ну что, блин? Опять? Ты, тот который рождает архитекторов, где, блин, пример на 10 строчек и текст в два абзата про то, как вызвать метод **send()**?» И ответа на этот вопрос у меня нет…

И вот, наконец, проблема решена! Друзья, я поздравляю Вас – Вас не коснется эта злая участь – затеряться в дебрях Java Mail!

«Ух, ты! Ты создал пример отправки писем? А может быть еще и статью? Нет, наверное, даже – целый курс!» - спросите Вы.

«Нет» – отвечу Вам я, но зато я предлагаю Вам задачу, в рамках которой Вам нужно будет отправлять электронные письма. Эта задача дается в начале курса, а значит, в тот момент, когда мы подойдем к нашим «архитектурным баталиям» для Вас в этом вопросе уже не будет ничего необычного!

Я и сейчас стыдливо отправляю Вас в Google, но делаю это очень заранее ☺.

Вот жеж с… Ну, думаю, введения достаточно.

Итак, друзья, отправкой писем в Java занимается библиотека JavaMail. JavaMail – это библиотека от Oracle, то есть – от владельцев Java. Другими словами, можно говорить, что JavaMail – это часть ядра Java-платформы, пусть и с некоторой натяжкой, так как поставляется этот API все таки в виде отдельной библиотеки, а не вместе с JDK.

Вот официальная страница API:

<http://www.oracle.com/technetwork/java/javamail/index.html>

Здесь Вы сможете скачать последнюю версию библиотеки, почитать документацию, посмотреть примеры.

А после этого, пожалуйста, при помощи технологии сервлетов, реализуйте приложение, которое запрашивает у пользователя:

* Адрес отправителя сообщения.
* Адрес получателя сообщения.
* Тему сообщения.
* Текст сообщения.

Проводит необходимую валидацию данных. И отправляет электронное письмо на указанный адрес.

Ну вот и все. Друзья, успехов Вам в выполнении работы! Пожалуйста, постарайтесь выполнить ее до начала наших «Архитектурных слушаний».