**Servlet API.**

**Задание-3.**

**HTML-Builder для сервлетов.**

Друзья, будет очень неплохо, если текущую задачу Вы будете выполнять после задания №2 – того, в котором от Вас требовалось сверстать HTML-форму при помощи сервлета. В этом случае, Вам будет понятно, почему появилась текущая задача.

Итак, если Вы уже делали задание №2, то, скорее всего, Вы уже возненавидели сервлеты, Java EE, да и вообще, всю Java-платформу, приняли решение никогда больше не программировать на Java, и даже, скорее всего, организовали общество по противодействию вовлечения детей в Java ☺

Я согласен с Вами, так как бесконечность строк типа

**out.println(“<HTML-строка>”)**

Не позволяет Вам видеть ни HTML-а, ни Java-кода.

Так вот, основная цель текущей задачи – разработать класс, который будет:

1. Считывать построчно HTML-шаблон, который Вы верстаете отдельно и располагаете в своем приложении.
2. И записывать каждую строку HTML-шаблона в полученный поток вывода (Writer) при помощи метода **out.println(“<HTML-строка>”).**

Ваш метод может называться **writeContent()** и иметь следующую сигнатуру:

**public void writeContent(Writer out)**

Обратите внимание на то, что поток вывода, в который Вы должны писать HTML, метод получает в качестве параметра.

Кроме потока вывода, Вашему классу потребуется путь к HTML-шаблону, который нужно записать. Этот путь должен быть относительным и рассчитываться от корневого каталога Вашего приложения. Класс должен получить этот путь в качестве параметра конструктора.

Следующий момент, который можно реализовать в этой задаче – передача дополнительных параметров. Например, Ваша страница может выглядеть вот так:

**Имя: Иван**

**Фамилия: Иванов**

Здесь значения **Иван** и **Иванов**, скорее всего, передавались в качестве параметров запроса. Так Вот для того, чтобы реализовать такое динамическое поведение, Ваш HTML-шаблон может выглядеть вот так:

**<p>**

**Имя: ${FIRST\_NAME\_PARAM}**

**</p>**

**<p>**

**Имя: ${FIRST\_NAME\_PARAM}**

**</p>**

Обратите внимание на конструкции **${FIRST\_NAME\_PARAM}**. Ваш класс может искать в HTML-шаблоне выражения типа **${\*}**, где **\*** – это любая последовательность символов, т. е. – любое имя, и заменять эту конструкцию значением, которое Вы передали под этим именем в класс.

Вопрос, как передать множество таких пар «имя-значение» в Ваш класс? Для этого в конструктор класса добавьте еще один параметр – карту параметров, которые потребуются в HTML-шаблоне:

**Map<String, String> params**.

**И заключительное, но очень важное требование в текущей задаче**: класс, который вы разработаете для считывания HTML-шаблона и записи HTML-а в поток вывода respons-а, а так же все вспомогательный классы (если они у вас будут), должны (внимание-внимание):

* Разрабатываться в виде отдельного JSE-проекта.
* Компилироваться в отдельный JAR-файл
* И этот JAR-файл должен использоваться вашим Web-проектом, как сторонняя библиотека, которую вы расположите в папке WEB-INF\lib

После того, как вы выполните это требование, вы начнете потихоньку привыкать к тому, что ваше Web-приложение, это не только тот код, который вы написали для проекта, но еще и множество сторонних библиотек.

Какие-то из этих библиотек будут разрабатываться сторонними компаниями или сообществами (например, Driver-ы для СУБД, фреймворки и так далее).

А какие-то библиотеки будут разрабатываться вами же (или вашей командой), но разрабатываться они будут в виде отдельных проектов, именно для того, чтобы сделать их универсальными и легко переносимыми из одного проекта в другой.

Согласитесь, тот **ContentWriter**, который вы реализуете в рамках этого задания, вы с удовольствием захотите использовать во всех оставшихся сервлетных работах.

Ну и само-собой, на проверку в одно архиве мне нужно будет прислать **два отдельных проекта**. При этом, обратите внимание, что если вы захотите использовать этут библиотеку в дальнейших лабораторных работах, то двух **проектов я отвас не жду** – и вы можете просто вкладывать JAR-ку в каталог **WEB-INF\lib**. Опять же круто – меньше написанного вами кода – это меньше придирок с моей стороны ☺

В заключение стоит сказать, что такой подход способен несколько облегчить для Вас программирование при помощи сервлетов. Но, пожалуйста, никому о нем не рассказывайте! И вовсе не для того, чтобы получить конкурентное преимущество на рынке Java Web разработчиков. Скорее, наоборот – никто не должен знать, что Вы программируете на сервлетах! ☺