**Задание 2. Encoding Filter.**

Реализуйте класс фильтра, который будет выполняться при каждом запросе к Вашему приложению и устанавливать заданную кодировку для объектов **request** и **response**. Значение кодировки фильтр должен получать одним из трех способов:

1. Прежде всего, фильтр должен попытаться прочитать кодировку при помощи параметра фильтра **encoding-value**, которое можно передать в секции **<init-param>** во время конфигурирования фильтра.
2. Если init-параметр из первого пункта не задан, фильтр должен попытаться прочитать значение кодировки из параметра контекста **filter-encoding-value**, которое можно будет сконфигурировать в секции **<context-param>** в файле web.xml.
3. Если же разработчик, во время включения Вашего фильтра в проект, не передал значение кодировки ни первым, ни вторым способом, фильтр должен использовать значение кодировки по умолчанию, которое задано в самом фильтре в виде константы **DEFAULT\_ENCODING\_VALUE**. Значением кодировки по умолчанию должно быть **UTF-8**.

Так же во время работы над заданием, обратите, пожалуйста, внимание на следующие моменты:

1. Вся работа по определению значения кодировки (см. пункты 1-3 выше) должна быть выполнена в **init** методе фильтра.
2. Успешным результатом работы метода **init** должна стать инициализация **private** поля **encodingValue** в фильтре, которое будет хранить кодировку.
3. Помните о том, что фильтры выполняются в многопоточной среде, поэтому поле **encodingValue** должно использоваться классом Вашего фильтра ***только для чтения***. С другой стороны, разрабатываемый фильтр предельно прост, и придумать для поля **encodingValue** некую дополнительную работу будет крайне затруднительно. Однако если у Вас это получится, то работу Вашу я не приму ☺
4. При инициализации поля **encodingValue** фильтр должен выбросить исключение, если разработчик указал значение кодировки ***и в init-параметре фильтра и в context-параметре приложения***. Исключение должно выбрасываться, даже если значения кодировок в этих двух конфигурациях одинаковы.
5. В качестве исключения, которое будет генерироваться в пункте 4, должен выступить Вами же написанный класс. Назовите его, например, **DuplicateEncodingException** и унаследуйте от **RuntimeException**. Класс исключения расположите в том же пакете, что и класс фильтра.

Обратите внимание на то, что Ваше исключение должно выбрасываться из метода **init**, тем самым останавливая работу приложения, в котором используется Ваш фильтр, и распечатывая StackTrace на Web странице. Стоит признать, что это достаточно спорный момент. Возможно, Вы предпочтете сделать приложение более устойчивым к мелким ошибкам разработчика, и в случаях, когда значение кодировки определено дважды в разных местах, просто предпочтете использовать любое установленное значение – например то, которое было считано Вашим фильтром последним. Однако я считаю, что любой код кроме своих прямых обязанностей (в нашем случае – установка кодировки), должен бороться за чистоту и простоту архитектуры проекта всякий раз, когда ему предоставляется такая возможность. В нашем случае такая возможность есть!

Подумайте, как бы Вы объясняли работу своего фильтра коллегам, если бы исключение при двойном определении кодировки не выбрасывалось? Это был бы примерно такой монолог – *«Друзья, мой фильтр установит кодировку для объектов request и response. Значение кодировки Вы можете задать двумя способами – либо при помощи init-параметра фильтра, либо при помощи context-параметра приложения. Помните о том, что у context-параметра более высокий приоритет, поэтому, если вы укажете кодировку и при помощи init-параметра, и при помощи context-параметра, то будет использоваться та кодировка, которая была указана при помощи context-параметра, а не та, которая была указана при помощи init-параметра. Так что если Вам нужно именно значение кодировки, которое вы указали при помощи init-параметра, то не используйте, пожалуйста, конфигурирование при помощи context-параметра. Но, правда, если вы указали одинаковую кодировку и в init-параметре, и в context-параметре, то приоритет уже не играет роли. Потому что хоть приоритет context-параметра и выше, чем приоритет init-параметра, но кодировка в них одинаковая, а значит, кодировка init-параметра равна кодировке context-параметра…».* И все это время, пока Вы рассказываете, Ваши коллеги будут думать про себя, что, скорее всего на них тестируют некую абсолютно новую версию ада, где хоть и не жарко, но невыносимо. (Ну, скорее всего, они будут думать примерно тоже самое, что думаете сейчас Вы, уже пол часа читая задание для фильтра на две строки кода ☺). Но это еще и не самое плохое. Дело в том, что когда Вы закончите свое повествование, все равно у Ваших коллег останется масса вопросов. И вопросы эти будут примерно следующего плана:

* *«Напомни, у какого параметра там выше приоритет?»*
* *«А почему у context-параметра?»*
* *«А можно как-нибудь программно поменять приоритеты и*

*сделать главным init-параметр?»*

* *«Нельзя, да? Жаль…»*
* *«Дружище, а можешь сам переписать и сделать главным*

*init-параметр?»*

* *«Э…. ребята! Мне вот как раз главным нужен context-параметр»*
* *«Дружище, а можешь два таких фильтра забабахать? Ну вот чтоб прям всем хорошо стало?»*
* *«Так они ж одинаковые будут почти!»*
* *«Ну и что, что одинаковые, зато каждый получит свое…»*

Ну и дальше в таком же духе. Подумайте – оно Вам надо? А теперь представьте, как выглядит диалог с коллегами, если исключение существует:

**Вы:** *Друзья, мой фильтр установит кодировку для объектов request и response. Значение кодировки Вы можете задать двумя способами – либо при помощи init-параметра фильтра, либо при помощи context-параметра приложения.*

**Коллеги:** *А у какого параметра Выше приоритет?*

**Снова Вы:** *А нет никаких приоритетов.*

**Снова Коллеги:** *То есть мы можем использовать любой способ?*

**Опять Вы:** *Да, конечно.*

**Коллеги не унимаются:** *И работать будет одинаково?*

**Вы не сдаетесь:** *Базарю!*

**Один, самый настырный коллега:** *А два сразу могу использовать?*

**Вы:** *Да, легко!*

**Снова он:** *Так что же будет, если нет приоритетов?*

**Вы, показывая, что диалог уже не о чем:** *А попробуй – увидишь ;)*

**Он, т. к. последнее слово всегда за ним:** *А я попробую!*

И он попробует! Ведь последнее слово всегда за ним. Эта проба займет у него несколько минут, после чего в окне браузера Он увидит StackTrace Вашего Exception-а, возможно с поясняющей надписью следующего плана – *«You can specify encoding either using init-param or context-param. But you can't use both options!»*, что на русский можно будет перевести как-то так – *«Таки определись уже со своим конфигом, и харэ мазать сопли по web.xml-ю! Хочешь показаться умным – пойди поработай!»*

И вот, наконец, вывод по длинному и нудному пункту №5: друзья, если у Вас есть возможность проконтролировать использование Вашего кода – сделайте это. Если Вы можете избавить Ваших коллег от долгого и мучительного выбора, избавьте их от него. Заметьте, я не сказал – «Лишите пользователей Вашего кода возможности выбирать», я лишь прошу Вас сделать этот выбор как можно более простым. И генерация своего собственного исключения как нельзя лучше подходит для этого случая.

**Примечание:** В жизни автора задания еще ни разу не встречались такие коллеги-нудилы, которые были представлены в данном пункте. И, искренне надеюсь, что не встретятся. Скорее всего, таких нудных разработчиков и не существует вовсе. Так что все персонажи данного пункта выдуманы, исключительно с целью представить ситуацию как можно более явно.

1. Не забывайте, пожалуйста, о том, что метод **setCharacterEncoding** для объекта **request** может выбросить исключение **UnsupportedEncodingException**, в том случае, если разработчик указал неверное значение кодировки, т. е. кодировку, которая не известна Java платформе, или не поддерживается ей.

Не вздумайте его обрабатывать или прятать! Не вздумайте так же устанавливать какое-либо значение кодировки по умолчанию, в случае возникновения такого исключения! Помните, Ваш фильтр – это фундаментальный, узкоспециализированный, няшечно-вкусняшечный инструмент, который делает супер-полезную, но при этом супер-определенную работу – ***он устанавливает кодировку для request-а и response-а***. Не Ваша вина в том, что какой-то разработчик-разгильдяй, получив в руки такой мощнейший инструмент, не умеет им пользоваться.

Поэтому, без тени сомнения, позвольте этому **UnsupportedEncodingException-у** вырваться из Вашего фильтра, и расположить все прелести своего StackTrace-а прям на всю Web-страницу этого бедолаги. Согласитесь, чем быстрее программист увидит этот StackTrace, тем быстрее он исправит проблему. Или, если не сможет ее исправить, задумается о том, что программирование – это не его, уволится, устроится на новую работу по призванию, и станет, наконец, счастлив. В общем, так или иначе, все закончится хорошо, так что не переживайте ☺.

Абсолютно отдавая себе отчет в том, что Вы уже в шаге от прекращения дальнейшего изучения курса, так как сомневаетесь, что текст текущего задания когда-нибудь закончится, и справедливо волнуетесь, что в своей жизни Вы больше ничего не успеете, кроме как дочитать этот текст до конца, все же рискну еще чуть-чуть отвлечь Вас.

Раз уж в последних двух пунктах мы очень подробно останавливались на вопросах генерирования исключительных ситуаций Вашим кодом, и отказе от обработки этих исключений, было бы свинством с моей стороны не предоставить Вам ссылку на какой-нибудь авторитетный источник, подтверждающий мои слова.

Лучшим учебником по этому вопросу, на мой взгляд, является коан о программировании №115 под названием **«Боль»**. Я настолько уважаю авторов коанов о программировании, и те тексты, которые они создали, что не могу позволить себе заниматься плагиатом и перепечатывать их, так что, прежде всего ниже приводится ссылка на этот коан:

<http://thecodelesscode.com/case/115?lang=ru>

С другой стороны, коаны о программировании – это источник истинного знания разработчиков ПО. Источник настолько глубокий и полный, что я не могу себе позволить заставить Вас ждать, пока браузер загрузит Вам страницу по вышеприведенной ссылке. Поэтому еще ниже приводится сам текст коана №115, но не как недостойный плагиат, а лишь в качестве полноправной цитаты. Пока Вы читаете эту цитату, браузер успеет загрузить для Вас сайт с коанами, и Вы, оказавшись на сайте, сможете посвятить себя чтению других коанов, достигая все большего просветления с прочтением каждого текста ☺. Итак, цитата:

*Случай 115. «Боль»*

*Слухи о поврежденных данных достигли ушей Java-мастера Банзена. Изучив код, он позвал к себе одного монаха.*

*«Ты очень стараешься избавиться от NULL», – сказал мастер. – «Инициализируешь String пустой строкой и присваиваешь пустую строку всем аргументам типа String. Кроме того ты хранишь свойства объектов переноса данных в примитивах, отчего NULL колонки типа Integer читаются как нули, а NULL колонки типа Boolean читаются как false. Объяснись».*

*Мальчик ответил: «Я заметил, что наиболее часто встречающая в нашем коде ошибка –* ***NullPointerException****. Я стараюсь максимально снизить количество таких ошибок».*

*Банзен немного подумал, затем бросил в камин медную монетку и засыпал ее толстым слоем раскаленных углей. «Достань монетку голыми руками» – сказал он.*

*Монах трижды пытался выполнить поручение, но каждый раз, обжегшись, с болью отдергивал руку.*

*Банзен схватил мальчишку за плечо, нашел нерв и сильно надавил. Боль прекратилась.*

*«Еще раз», – приказал Банзен, все еще зажимая нерв.*

*Монах попытался еще раз. С радостью он осознал, что может свободно рыться в углях, но радость его превратилась в ужас, когда он почуял и услышал, как горит его плоть. Тут же он отдернул обожженную, покрытую волдырями руку и уронил монетку на пол.*

*Банзен отпустил монаха и покинул комнату. Монаху больше не требовались наставления, но ему пришлось научиться печатать левой рукой.*

Согласитесь, господа, впечатляет! Думаю, после прочтения этого текста, я окончательно убедил Вас. И не только убедил, но, скорее всего, на время сменил программистское настроение на философское. Так что отложите в сторону этот текст на некоторое время, и попейте чайку, размышляя о сущем – я подожду ☺

Ну что, вернулись? Отлично, так как у нас, наконец, финишная прямая. Реализовав фильтр, напишите так же небольшое Web-приложение, которое позволит продемонстрировать его работу.

И вот тут Вы, наверное, думаете про себя – «Имперический котегаритив! Оказывается, чтобы остановить словесный понос, бурным потоком от собеседника идущий, не обязательно орать ему в ответ и грозить расправой. Иногда достаточно лишь оставить его на время, да выпить чаю… Воистину, жизнь наша полна презабавностей… Так а чего же он хотел то от меня в этом задании? Блин, перечитывать или просто забить?..»