**Servlet API.**

**Задание-1.**

**Электронная анкета для оценки качества образования.**

Первая задача по сервлетам преследует три главные цели:

1. Познакомиться основными компонентами HTML-форм, которые можно применять при программировании Web-приложений.
2. Научиться использовать эти компоненты:
   * Передавать параметры с Web-формы на сервер
   * Принимать эти параметры на сервере
   * Извлекать значения пришедших на сервер параметров
   * Проводить валидацию полученных значений
   * Выводить полученные значения на Web-странице и т. д.
3. Ну и, наконец, посмотреть – что же такое сервлет
   * Объявить его
   * Сделать доступным для клиентских запросов
   * И реализовать логику обработки запросов.

Задав основные цели первой работы, можно подробнее описать само задание. В рамках этой задачи Вам предстоит разработать Web-систему для проверки качества образования в некотором виртуальном тренинг центре. К слову сказать, такую анкету Ваш препод (ну замени, замени уже нормальным словом – «преподаватель»!) будет просить Вас заполнить по окончанию нашего курса, но, правда, там и анкета будет настоящая, и ответы не такие радужные, но это совсем другая история, и к текущей задаче она не имеет никакого отношения. Итак, приступим.

Система проверки качества образования представляет собой Web приложение, состоящее из двух страниц:

|  |  |
| --- | --- |
| **Страница ввода данных** | Должна быть реализована при помощи **HTML-страницы**. |
| **Страница отображения результатов** | Должна быть реализована при помощи **сервлета**. |
|  |  |

При первом доступе к приложению управление должно передаваться на страницу ввода данных (форму), внешний вид которой Вы можете видеть на рисисунке ниже:

***Внимание, рисунок в стадии разработки!!!***

Как видно из рисунка, страница для ввода данных отображает достаточно большое количество компонентов различных типов. Более того, к некоторым компонентам могут применяться дополнительные требования по их поведению или валидации. Распишем подробнее требования, которые налагаются к каждому компоненту:

***Примечание:*** *При просмотре требований каждого компонента, обратите, пожалуйста, внимание на то, что* ***«Сообщение об ошибке»*** *– это сообщение, которое должно отобразиться на форме, если пользователь нарушил правила ввода данных в этот компонент (правила валидации).*

1. Фамилия:

|  |  |
| --- | --- |
| **Назначение компонента** | Поле для ввода фамилии анкетируемого. |
| **Тип компонента** | Текстовое поле |
| **Правила валидации** | 1. Поле обязательно к заполнению 2. Строка, введенная в поле:    1. Не должна быть пустой (строкой нулевой дины)    2. Не должна состоять только из пробельных символов |
| **Сообщение об ошибке** | «Фамилия» не может быть пустой |
| **Примечания** | Цель данного пункта – научиться работать с простейшими текстовыми полями, и отметить про себя, что иногда корректность ввода в эти поля должна проверяться. |

2. Имя:

|  |  |
| --- | --- |
| **Назначение компонента** | Поле для ввода имени анкетируемого. |
| **Тип компонента** | Текстовое поле |
| **Правила валидации** | Аналогично полю «Фамилия», см. п. 1 |
| **Сообщение об ошибке** | «Имя» не может быть пустым |
| **Примечания** | Вообще говоря, текущий пункт абсолютно аналогичен пункту №1 и ничего нового Вы из него не узнаете. Но согласитесь, указать фамилию и не указать имя – это очень в стиле – «Выступает товарищ Грачева…». Современные приложения как-то душевнее должны быть, что ли ☺ |

3. Отчество:

|  |  |
| --- | --- |
| **Назначение компонента** | Поле для ввода отчества анкетируемого. |
| **Тип компонента** | Текстовое поле |
| **Правила валидации** | Нет. |
| **Сообщение об ошибке** | Нет |
| **Примечания** | Основная цель данного пункта – заострить Ваше внимание на том, что для отправки формы на сервер пользователь не всегда заполняет *все* поля формы.  Некоторые из них могут быть необязательными и заполняться только лишь при желании пользователя.  Более того, обратите внимание, что требовать от пользователя ввода данных, без которых Ваша система может корректно работать – дурной тон.  Заметьте, как много появилось систем, которые во время регистрации требуют ***только «Login» и «Пароль»***, а все остальные данные, типа «сколько котов было у Вашей прабабушки?» пользователь сможет заполнить позже, например, на странице своего профиля. И то – если сочтет нужным.  Почему так происходит? Потому что у современных людей никогда и ни на что нет времени. И если на регистрацию в Вашей системе пользователь тратит больше 3 минут, то крайне высока вероятность, что спустя три с половиной минуты он будет регистрироваться у Ваших конкурентов. Кто из разработчиков может это допустить? Правильно – никто.  Отметим, однако, что и в лояльные крайности при разработке компьютерных систем впадать не стоит. Как много Вы можете вспомнить случаев в нашем городе, когда банкомат услужливо говорил Вам – *«Чувак, ты забыл pin-код. Но я ведь вижу – тебе срочно нужны деньги. Короче, вот возьми пока наличку, а как pin-код вспомнишь, заскочи на пару минут, введи – все-таки порядок, есть порядок!»* |

4. Секретная Фраза:

|  |  |
| --- | --- |
| **Назначение компонента** | Поле для ввода секретной фразы. |
| **Тип компонента** | Поле для ввода паролей. |
| **Правила валидации** | Аналогично полю «Фамилия», см. п. 1. |
| **Сообщение об ошибке** | «Секретная фраза» не может быть пустой. |
| **Примечания** | На самом деле задача этого поля – просто напомнить Вам, что кроме обычных текстовых полей есть еще и поля, где вводимые символы не видны любопытным варв***а***рам, которые всегда рядом. |

5. Возраст:

|  |  |
| --- | --- |
| **Назначение компонента** | Поле для ввода возраста. |
| **Тип компонента** | Текстовое поле. |
| **Правила валидации** | 1. Поле ***может*** быть пустым. 2. Но если поле не пустое, то оно должно содержать валидное целое число. |
| **Сообщение об ошибке** | «Возраст» измеряется только цифрами. |
| **Примечания** | Обратите внимание на валидацию этого поля. Пользователь может ***не указывать*** возраст. Но, если пользователь решил все-таки заполнить это поле, Ваша система должна следить за тем, чтобы было введено ***валидное целое число***.  Основная цель данного типа валидации заострить Ваше внимание на том, что в машинном мире все точно также, как и у людей – «Или делай хорошо, или не делай вообще». |

6. Пол:

|  |  |
| --- | --- |
| **Назначение компонента** | Указание пола анкетируемого. |
| **Тип компонента** | Радио-кнопки (Radio Buttons). |
| **Возможные значения** | 1. Мужчина. 2. Женщина. |
| **Правила валидации** | 1. Пол должен быть указан. 2. Одновременно можно указать только один пол. |
| **Сообщение об ошибке** | Нет. |
| **Примечания** | Обратите внимание на странную забавность – валидация для этого поля прописана, а сообщение об ошибке – **нет**.  *«Хммм… Наверное очень сложно будет реализовать эти требования а тем более проверить их, если даже сообщение об ошибке ввода решили не использовать»* (сарказм ☺) |

7. Курс:

|  |  |
| --- | --- |
| **Назначение компонента** | Указание курса, который прошел анкетируемый. |
| **Тип компонента** | Выпадающий список (Combobox). |
| **Возможные значения** | 1. Java SE 2. Java EE 3. Тестирование корабельных якорей методом погружения в Северное Море 4. Вышивание крестиком в условиях, близких к невесомости. |
| **Правила валидации** | Должен быть выбран один курс. |
| **Сообщение об ошибке** | Выберите Курс, который Вы проходили. |
| **Примечания** | Вы сами можете составить список курсов, которые Вам интересны. |

8. Преподаватель:

|  |  |
| --- | --- |
| **Назначение компонента** | Указание преподавателя, который вел курс. |
| **Тип компонента** | Список выбора (Listbox). |
| **Возможные значения** | 1. Иванов Иван Иванович 2. Петров Петр Петрович 3. Сидоров Сидор Сидорович |
| **Правила валидации** | Должен быть выбран один преподаватель. |
| **Сообщение об ошибке** | Укажите преподавателя, который вел курс. |
| **Примечания** | Нет. |

9. Оценка курса:

|  |  |
| --- | --- |
| **Назначение компонента** | Выставить оценку, которую заслужил курс и преподаватель. |
| **Тип компонента** | Выпадающий список (Combobox). |
| **Возможные значения** | 1. 10 2. 100500 3. Мильен Мильярдов 4. Еще пока нет такого числа 5. Это было даже лучше, чем 10 000 люстр от Карлсона! |
| **Правила валидации** | Должна быть выбрана одна оценка. |
| **Сообщение об ошибке** | Укажите, как Вы оценили курс. |
| **Примечания** | Нет. |

10. Прочие курсы:

|  |  |
| --- | --- |
| **Назначение компонента** | Указать курсы, которые пользователь хотел бы пройти в будущем. |
| **Тип компонента** | Список выбора с возможностью указания нескольких значений (Multy Listbox). |
| **Возможные значения** | Здесь должны быть перечислены ***все*** курсы, указанные в пункте №7, ***исключая тот***, который выбрал пользователь в пункте №7 |
| **Правила валидации** | Нет. |
| **Сообщение об ошибке** | Нет. |
| **Примечания** | Друзья, этот пункт должен заострить Ваше внимание еще на одном интересном моменте в программировании:  Программе важно не только следить за корректностью ввода пользователя.  Программа ко всему прочему должна следить и за корректностью собственной работы.  Посмотрите еще раз на этот пункт – пользователь не может допустить в нем ошибку, но и программа не должна ошибиться при выводе данных. А именно, проследить за тем, чтобы пользователь ***не мог выбрать тот курс, который он только что прошел***.  Похожие требования – исключения одной или нескольких позиций из общедоступного списка данных – встречается при программировании достаточно часто. |

***Еще одно важное примечание к пункту-10:*** *Так как в этой задаче форма ввода должна быть реализована при помощи HTML-страницы, а не сервлета, то вы лишаетесь своего основного оружия – языка Java. И для того, чтобы реализовать такое требование – «Исключить из списка курс, который пользоватль только что проходил» – Вам потребуется много JavaScript-а. Так как мы не JavaScript разработчики – требовать от Вас тратить три вагона времени на реализацию этого пункта – я не имею права. Тем более, что задач про сервлетам достаточно много. Тем более, что – «Блин, ну это же сервлеты!». Поэтому со временем сложилась такая практика – требования пункта-10 выполняют только те слушатели, которым не сложно работать с JS, или те, кто увидел JS в первый раз, но воспылал к нему безмерно!*

*Если же человек говорит мне, что JS пока дается сложновато, и такая реализация потребует много времени – я совершенно спокойно отношусь к тому, что Вы не реализуете требования пункта-10, и в списке «прочих курсов» будет болтаться и тот, который пользователь только что прошел*.

11. Как Вы о нас узнали:

|  |  |
| --- | --- |
| **Назначение компонента** | Указать источники, при помощи которых пользователь узнал о тренинге. |
| **Тип компонента** | Множество флажков (Checkboxes). |
| **Возможные значения** | 1. Реклама по ТВ. 2. Реклама по радио. 3. Реклама в Интернете. 4. Реклама в метро. 5. Тварь, какая-то мне этими буклетами весь почтовый ящик зас#%ла. 6. Другое. |
| **Правила валидации** | Должен быть выбран хотя бы один вариант. |
| **Сообщение об ошибке** | Укажите, как Вы узнали о нас. |
| **Примечания** | В этом пункте стоит обратить внимание на два момента.  Во-первых, еще одна стратегия валидации, которая может встречаться в реальном программировании – программе не важно, сколько пунктов выберет пользователь, но он должен выбрать ***хотя бы один*** пункт.  Во-вторых, флажок «Другое» должен вести себя следующим образом: если пользователь выбирает его, то многострочное поле ввода под этим флажком должно стать активным (enabled). Подробности смотрите в пункте №12. |

12. Другое:

|  |  |
| --- | --- |
| **Назначение компонента** | Указать подробности, если пользователь выбрал флажок «Другое» в пункте №11. |
| **Тип компонента** | Многострочное поле ввода (Textarea). |
| **Правила валидации** | 1. Если флажок «Другое» в пункте №11 ***не выбран***, поле не доступно для ввода текста. 2. Если флажок «Другое» в пункте №11 ***выбран***, то строка, введенная в поле:    1. Не должна быть пустой (NULL или строка нулевой дины).    2. Не должна состоять только из пробельных символов. |
| **Сообщение об ошибке** | Отвергаешь – предлагай! |
| **Примечания** | И опять-таки, интересный факт о валидации…  Существуют компоненты, которые изначально не требуют проверки корректности ввода.  Но они зависят от других компонентов на форме.  Текущее поле – яркий тому пример – до тех пор, пока флажок «Другое» не выбран, текущее поле никак не валидируется, но стоит отметить этот флажок, и текущее поле позволяет ввести пользователю значение и начинает следить за тем, что именно он ввел.  Дополнительную информацию смотрите в пункте №11. |

13. Прочие Рекомендации:

|  |  |
| --- | --- |
| **Назначение компонента** | Позволить пользователю в свободной форме указать прочие моменты относительно курса, которые не были учтены составителями анкеты. |
| **Тип компонента** | Многострочное поле ввода (Textarea). |
| **Возможные значения** | Нет. |
| **Правила валидации** | Нет. |
| **Сообщение ошибки валидации:** | Нет. |
| **Прочие условия** | Этот пункт был введен по следующим причинам:   1. Позволить клиенту высказаться - это правило хорошего тона во всех компьютерных системах (блин, да это даже в реальной жизни правило хорошего тона). 2. Заострить Ваше внимание на том, что такие моменты, как правило, ***не валидируются***. Опять-таки к вопросам этикета – не дать пользователю высказать свое мнение – это не красиво, но требовать это мнение от пользователя Вам тоже никто не разрешит. 3. Добиться того, чтобы пунктов не получилось 13 ☺ |

14. «Отправить – здесь все правда» и «Отправить – я все равно все наврал»:

|  |  |
| --- | --- |
| **Назначение компонента** | Оба компонента должны отправлять форму на сервер – на страницу отображения результатов. |
| **Тип компонента** | Кнопки (Submit) и скрытое поле (Hidden). |
| **Возможные значения** | Нет. |
| **Правила валидации** | 1. Если Валидацию не прошел хотя бы один из предыдущих компонентов, то валидация кнопок не производится, так как отправка формы на сервер невозможна. 2. Если форма заполнена верно, и пользователь нажал кнопку **«Отправить – здесь все правда»**, то должно отобразиться всплывающее окно (alert) с сообщением **«Браво, честный человек! Смело отправляйся смотреть результат!»** и кнопкой **«Ok»**. При нажатии на кнопку «Ok» форма должна отправиться на сервер. 3. Если форма заполнена верно, и пользователь нажал кнопку **«Отправить – я все равно все наврал»**, то должно отобразиться всплывающее окно (alert) с сообщением «Вообще-то врать не хорошо, даже если не видят… Точно отправляем» и кнопками **«Ok»** и **«Cancel»**.    1. При нажатии на кнопку «Ok» форма должна отправиться на сервер.    2. При нажатии на кнопку «Cancel» пользователь должен остаться на текущей странице. |
| **Сообщение ошибки валидации:** | Смотри выше. |
| **Прочие условия** | Здесь обратите внимание на следующие моменты:   1. Валидация может быть не только «Запретительной», но и «Вопросительной» и «Утвердительной». Т. е. даже когда пользователь заполнил форму верно, наши приложения все равно могут у него что-нибудь спросить или что-нибудь ему посоветовать. Вы можете заметить это просматривая правила валидации текущего пункта для кнопок. 2. В этом пункте **обязательно использование** еще одного компонента формы, который не имеет графического интерфейса – **скрытого поля** (hidden). Собственно пункт придуман в таком виде для того, чтобы Вы научились использовать скрытые поля. 3. Обратите внимание, что значением скрытого поля, отправляющегося на сервер, является информация о том, какую же именно кнопку нажал пользователь. 4. После этого сервлет на основании этого значения сможет распечатать на форме один из вариантов строк:    1. «Он все честно сказал».    2. Или «Не верьте ему – он все наврал» |

Обратите особое внимание на то, что валидация данных в этой работе должна проводиться дважды:

1. На клиентской стороне (на HTML-странице) при помощи JavaScrip-а.
2. И на серверной стороне (на сервлете) при помощи языка Java.

Более того, алгоритмы (или бизнес-логика, если хотите) клиентской и серверной валидации должны быть **абсолютно одинаковыми**. В качестве дополнительного задания, напишите, пожалуйста в письме с задачей свои мысли о том, для чего разработчики дублируют алгоритмы валидации на клиентской и на серверной стороне?

Получается, что HTML-страница не должна отправлять запрос до тех пор, пока пользователь не заполнит все поля правильно. Если JavaScript на HTML-странице обнаруживает ошибки, то необходимо отображать всплывающие окна (Alert-ы) и оставлять пользователя на этой же странице.

При этом, если запрос все-таки ушел на сервер (сервлет) с некорректными данными, то сервлет повторяет валидацию данных и должен на своей странице для всех параметров, которые были переданы неверно, отобразить сообщения об ошибке, вместо их значений.

Для того, чтобы была возможность проверить работу серверной валидации, пожалуйста, добавьте на HTML-форму checkbox **«Отключить JavaScript валидацию»** – это позволит HTML-форме отправлять данные, которые содержат ошибки.

В заключение стоит сказать следующее – если пользователь заполнил форму правильно, и отправил ее на сервер по средствам одной из submit кнопок, то управление должно передаться на сервлет, отвечающий за страницу результатов.

Этот сервлет должен прочитать из запроса все параметры и отобразить их на странице. Дизайн страницы Вы можете продумать самостоятельно. Но, пожалуйста, не делайте его **красивым и сложным**! Очнитесь, Вы же поуши в сервлете! Банальной распечатки в столбик мне будет достаточно.