

Министерство образования и науки Российской Федерации
Федеральное агентство по образованию
Ярославский государственный университет им. П. Г. Демидова
Факультет информатики и вычислительной техники

Заметки по информатике и математике

Сборник научных статей

Выпуск 1

Ярославль 2009

УДК 51(091)
ББК В1я43+3 973.2я43
3 26

*Рекомендовано
Редакционно-издательским советом университета
в качестве научного издания. План 2009 года*

Заметки по информатике и математике : сб. науч.
ст. / отв. ред. А. Н. Морозов ; Яросл. гос. ун-т
им. П. Г. Демидова. – Ярославль : ЯрГУ, 2009. –
3 26 Вып. 1. – 192 с.

В сборник включены материалы научной конференции, представленные аспирантами и студентами факультета ИВТ. Темы докладов касаются важных направлений развития математики, информатики и вычислительной техники.

УДК 51(091)
ББК В1я43+3 973.2я43

Редакционная коллегия:

В. А. Бондаренко

А. Н. Морозов (ответственный редактор)

П. Г. Парфенов

В. А. Соколов

© Ярославский государственный университет им. П. Г. Демидова,
2009

Валидация форм в Spring MVC с использованием AJAX: архитектурно-целостное решение

А. М. Васильев

Основным механизмом интерактивного взаимодействия в рамках веб-приложений являются формы, заполняемые пользователем. В большинстве случаев данные, введенные пользователем, необходимо подвергнуть проверке и сообщить ему о результатах этой проверки. Такой процесс называется валидацией форм.

Важным моментом валидации является наиболее быстрое и эффективное оповещение пользователя об ошибках во введенных данных. В среде веб-приложений одним из подходов, позволяющим решить эту задачу, является технология AJAX. Нарушение архитектуры модели MVC – следствие применения данной технологии.

В данной статье рассматриваются два варианта реализации валидации форм в Spring MVC с использованием технологии AJAX с точки зрения целостности архитектуры модели MVC.

MVC (Model View Controller) – архитектура программного обеспечения, в которой модель данных приложения, пользовательский интерфейс и управляющая логика разделены на три отдельных компонента, так что модификация одного из компонентов оказывает минимальное воздействие на другие компоненты.

Spring Framework – каркас для построения приложений со сложной внутренней структурой [1]. Основное своё применение он находит в приложениях, использующих платформу Java Enterprise Edition (JEE).

Главным компонентом Spring является контейнер объектов IoC (Inversion of Control), основная задача которого – настройка компонентов приложений и управление жизненным циклом объектов Java. Другой компонент, разрабатываемый в рамках проекта Spring, – Spring MVC. Он представляет собой каркас для создания веб-приложений в рамках модели MVC.

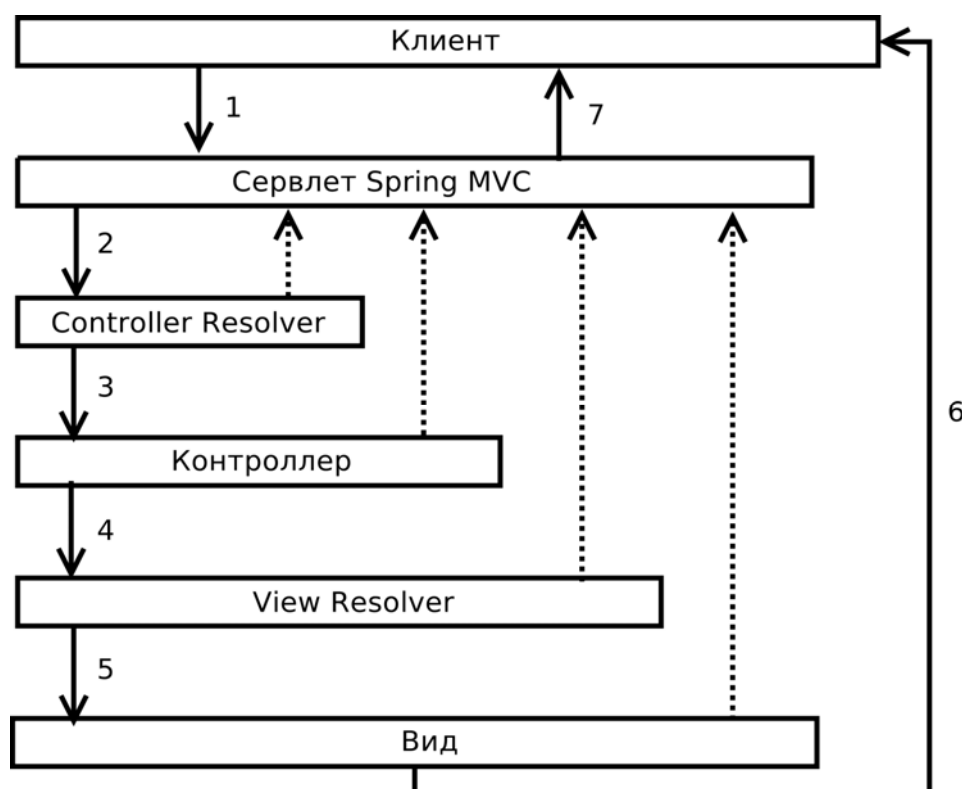


Рис. 1. Схема обработки запросов в Spring MVC

Процесс обработки запросов от клиента в Spring MVC осуществляется следующим образом (рис. 1). Клиент посылает запрос сервлету Spring MVC (1). Далее управление передаётся механизму определения контроллера по параметрам запроса (controller resolver, 2). В зависимости от адреса и типа запроса могут быть выбраны разные контроллеры. Перед передачей управления контроллеру на этапе (3) данные, принятые вместе с запросом, могут быть упакованы в JavaBean-объект.

Результатом работы контроллера служит некоторый набор объектов для отображения и строка, используемая механизмом разрешения объектов-видов (view resolver, 4). После создания или нахождения объекта-вида, удовлетворяющего строке запроса, ему передаётся управление (5). Объект-вид в свою очередь отвечает за формирование результата обработки данных в формате, понятном приложению-клиенту (6). В роли последнего часто выступает веб-браузер.

Если на каком-либо этапе происходит ошибка, то управление передаётся сервлету Spring MVC, который в зависимости от на-

строек может уведомить клиента о ней (7) или обработать ошибку более сложным образом.

На каждом этапе обработки запроса задействуются различные компоненты, область ответственности которых задаётся в ходе настройки Spring, которую можно производить с помощью конфигурационных файлов, механизма аннотирования классов, а также динамически во время функционирования приложения.

Подходы к валидации данных

Проверка введённых данных может производиться контроллером либо после получения их с компонента «вид», либо на стороне клиента перед отправкой данных на сервер.

Проверка данных контроллером требует от пользователя после введения всех полей формы и отправки данных на сервер ждать обновления страницы. Хотя это и одно из лучших решений с технической стороны, оно не слишком удобно для пользователя из-за возникающей задержки при повторной передаче всей страницы, а также из-за эффекта «пропадания» страницы при обновлении.

Существенно более привлекательными для пользователя представляются механизмы, позволяющие выполнить валидацию данных на сервере с последующим частичным обновлением данных на странице без её перезагрузки. Основным средством реализации данного подхода является асинхронный JavaScript (AJAX). Используя AJAX, можно добиться почти моментальной обратной связи, однако ему свойственен ряд недостатков, к числу которых можно отнести трудоёмкость в разработке, подверженность ошибкам и сложность в отладке. Наибольшую проблему при использовании AJAX представляет нарушение архитектуры MVC, связанное с размещением JavaScript-кода, ответственного за взаимодействие с сервером, на стороне клиента. Это усложняет разделение ролей программиста и дизайнера при разработке сложных систем.

В данной работе рассматриваются два подхода, позволяющие решать данную проблему за счёт выработки некоторых архитектурных решений, связанных с использованием на стороне клиента определённых компонентов без необходимости написания клиентского кода в каждом конкретном случае. Реализация этих подходов существенно опирается на возможности Spring