Spring Web Flow
Initial E-mail exchange between Keith and Erwin that led to project kickoff
---

Subject: spring webflow
From: "Keith Donald" <keith.donald at interface21.com>
Date: Fri, 4 Feb 2005 18:30:18 -0500
To: <klr8 at ervacon.com>

Erwin,

Hi! My name is Keith Donald, one of the core developers of the Spring Framework. I am also a consultant with Inteface21 (i21), working with Rod, Juergen, and Colin. I wanted to let you know your spring-webflow system proved to be a great asset on a recent (and quite challenging) i21 consulting project! The project had some tricky requirements that, quite frankly, demanded a higher-level Flow abstraction, and the simple yet flexible model you designed proved a solid base to build on.

I'm writing you first to say 'thank you' for your work! I also wanted to see what you think about us 'officially' integrating such a page flow capability into core Spring going forward. I certainly want to get your take and your blessing, and see if you'd be interested in helping with the effort. I think we can really take Spring MVC to the next level, and help achieve wider adoption of your approach and ideas!

Looking forward to hearing from you soon!

Best,

Keith

Keith Donald
Principal Consultant, Interface21
http://www.springframework.com

--

Subject: Re: spring webflow
From: "Erwin Vervaet" <erwin.vervaet at ervacon.com>
Date: Sat, 5 Feb 2005 09:32:28 +0100
To: "Keith Donald" <keith.donald at interface21.com>

Hello Keith,

Thanks for all this great feedback!

I would certainly welcome an integration of the web flow idea into the core Spring MVC module and I would be more than willing to help with this effort!

How do you see this practically? Would we be developing a new web flow controller from scratch or are you thinking about "moving" (and enhancing) the web flow system as it exists today? For me, both options are open.

(for non workaholics, skip the next part during the weekend :-)

Whatever we do going forward, I would like to discuss some interesting topics with some knowledgable people to get some extra input. I have received quite a bit of feedback on the web flows system lately and I'm looking at integrating some of the requested functionality. This has lead to quite a few questions:

* Redirect after POST: Somebody already submitted code that implements this and I can think of several ways to do it in an elegant way. The only real problem is a view on a final state of a flow that does not have a parent flow. In that case you have the problem that the flow is cleaned up (removed from the session) since it hit a final state. As a result the subsequent request resulting from the redirect can no longer access the flow model...

* Should the request parameter extraction be configurable (via ParameterExtractor)? This seems to complicate things since now only the controller knows how to access the flow (_event, _flowId, ...) parameters in the request. I think it would be better to support both (or all) parameter systems (e.g. _event=bla and _event_bla=) internally, using something similar to WebUtils.hasSubmitParameter().

* Should the names of the flow related properties in the flow model (e.g. "flowId") be configurable? This also complicates things since now only the controller knows these names, which, for instance, makes it hard for an action to extract the flowId from the model without using a magic constant.

* Should we have a conceptual 1-1 link between a controller and a web flow? Currently this is the case and it was inspired by the hard-coded Spring controllers (e.g. AbstractWizardFormController) where each controller has a particular flow. However, technically there is no reason why a single controller can't controll several flows, e.g. receiving the flow bean name as a request parameter. Actually, behind the screens a single controller already runs several flows when you have subflows.

* Handling back-button use should be redesigned (now uses SubFlowBackNavigationExceptionResolver). Now it feels like a bit of a hack. Also, should this be "integrated" with doulbe-submit handling (now uses token management methods in WebFlowUtils)?

Let me know what the plans would be and I would be happy to help!

Erwin Vervaet

--

Subject: RE: spring webflow
From: "Keith Donald" <keith.donald at interface21.com>
Date: Mon, 7 Feb 2005 00:48:56 -0500
To: "'Erwin Vervaet'" <ervacon at telenet.be>

Erwin,

Excellent to hear you're willing to help with this effort! I look forward to working with you closely on this.

Let me give you some more background on where we're at. For the recent project I mentioned, we developed a clean room implementation of your web flow system, using the core state model, terminology, and feature-set you developed as a guiding base to build upon. I've completed checking in that work into the CVS head of the springframework module at sourceforge. It's in the sandbox source tree for now, and the root package is org.springframework.web.flow. I invite you to review it!

The main difference you'll notice between our implementations is the existence of a configuration API for constructs like Flow and State definitions. There are also several new features we added to meet the demands of our last engagement. An important one was further refining of the FlowSessionExecution construct, adding in a well-defined lifecycle with support for FlowSessionExecutionListener lifecycle callbacks. Another was conditional matching for eventId processors (Transitions) enabling, for example, wildcards (e.g. if this event occurs, this transition is executed, but on any other event (*), execute this other transition.) We also added basic support for a MultiActionBean. A final major feature we added is support for strongly typed dependency injection and lookup for a single Flow definition's associated ActionBean, FlowAttributesMapper, and spawnable sub Flow definitions. This enables easy navigation to all executable code for any given root Flow definition from within a developer's IDE.

What we are missing in our implementation that you provide in yours:

- An abandoned web flow detector & cleanup filter. We simply never got around to integrating that! However, I was able to add the capability to track the lastEventTimestamp associated with each FlowSessionExecution, so completing this should be straightforward (all the data is there for the filter to make the decision.)

- A XML configuration format for defining flows. We favored a java-based configuration API decoupled from any one configuration format. I imagine it to be straightforward to adapt your flow xml reader to produce valid object instances using this API.

- Formal integration with Spring MVC, like your FlowController. Yes, can you believe that!? J We were actually required to use Struts on this last project, so we developed Struts front-controller integration (see org.springframework.web.flow.struts), but not yet Spring MVC integration. As I'm sure you can already attest to, doing this should be straightforward.

It's great to hear your users have already provided some good feedback the issues you mentioned in your email! We've been thinking about how to handle many of them as well, and were able to address some of them:

- Redirect after POST: no real progress on addressing this issue, other than some thinking on it from Colin and I. It's great to hear you have some ideas! I certainly hope they can be realized easily with the implementation in the sandbox (which we in general feel is quite strong and extensible.)

- Parameter extraction: yes, this should be configurable, but probably low priority. A simple, singleton configuration object that the flow event dispatchers (front controllers) query should get the job done. Right now, our FlowAction (the struts-based front controller in web.flow.struts) delegates to protected accessors for parameter and request attribute names, very similar to your FlowController. It should be straightforward to use dependency lookup or injection to grab that information from a central configuration service, managed by Spring.

- Flow session execution metadata: We actually tried using magic constants for retrieving information about a flow execution from the action beans and views (e.g lastEventId, currentStateId), but found that a bit error prone. We've since settled on an approach: there is a central FlowSessionExecution object accessible by a single well-known parameter from within the Flow Model (our AttributesAccessor interface.) Once you have that object, you can get to all other data about the currently executing flow session, like the currentState, the lastEvent, the rootFlow, the activeFlow, etc. We provide a FlowUtils.getFlowSessionExecution(model) convenience accessor to make this easy for developers.

- I can think of no compelling reason to force a 1-1 conceptual link between a dispatching front controller and a FlowSessionExecution. We actually favored a N-N approach in the app we developed, as it was easier to manage (we only had to define one FlowAction front controller per root flow, which was fairly small in number, and the views typically posted to

a generic URL, so they were really kept independent of what flow they were participating in.) Of course this now means if you want security, you'll be doing it at the Flow level.

- Browser navigation and duplicate submit handling: yes, we certainly want to discuss and see how we can improve in these areas. We currently support what your implementation does: allow the client to tell the server what the currentState is, but as you know that only works so long as you're not jumping around flows when you do manual browser navigation. We don't yet provide any out-of-the-box support for duplicate submit handing.

Going forward, if you could review the implementation in the sandbox and let me know what you think, that'd be excellent! It would be great to have your feedback, as we certainly tried to take your base work (which was quite good!) and further enhance upon it. I'd recommend checking out FlowTests to see some example Flow definitions (I can also provide additional usage examples). From there, I'd like to see about filling some of the implementation's gaps (e.g no flow cleanup filter, REDIRECT after post, "official" Spring MVC integration, other ideas for improvements you have!)

I look forward to working with you on this Erwin! If you have any questions, don't hesitate to send me an email. I'm also accessible via skype (userid kpdonald).

Best regards,

Keith