Title: Model-View-Controller (MVC 1.0) Specification.

Summary/Description: This JSR is to develop MVC 1.0, a model-view-controller specification for Java EE.

Duration: 2 weeks

Section 1: Identification

Specification Lead: Santiago Pericas-Geertsen

E-Mail Address: santiago.pericasgeertsen@oracle.com

Telephone Number: +1 561 214 4736

Specification Lead: Manfred Riem

E-Mail Address: manfred.riem@oracle.com

Telephone Number: +1 417 365 2548

Initial Group Membership: TBD

Supporting this JSR: TBD

Section 2: Request

2.1 Please describe the proposed Specification:

Model-View-Controller (MVC) is a common pattern in Web frameworks, where it is used predominantly by HTML-based applications. The Model refers to the application data, the View to the application's data presentation and the Controller to the part of the system responsible for managing input and producing representations.

The popularity of the MVC pattern, together with the data collected in the recent Java EE survey:

https://java.net/downloads/javaee-spec/JavaEE8_Community_Survey_Results.pdf

has prompted the need for a standard in this area. Web UI frameworks can be categorized as action-based or component-based. In an action-based framework, HTTP requests are routed to controllers and

turned into actions by application code; in a component-based framework, HTTP requests are grouped and typically handled by each component in an independent manner. The framework defined by this JSR falls into the action-based category and is, therefore, not intended to be a replacement for the component-based JSF, but simply a different approach to building Web applications on the Java EE platform.

It is a goal of this JSR to, whenever possible, leverage existing Java EE technologies. The Model part should leverage CDI and Bean Validation and the View part should leverage existing view technologies like JSPs and Facelets. The expert group will consider whether to leverage existing technologies such as JAX-RS for the Controller part, whether to invent new technology for the Controller part, or whether the Controller can be defined in a way that is independent of the technology used.

It is out of scope for this JSR to define a new view template language; instead existing languages should be evaluated and, possibly, an SPI defined as an extension for additional ones to be integrated.

2.2 What is the target Java platform? (i.e., desktop, server, personal, embedded, card, etc.)

This specification is targeted for Java SE 8 or higher and Java EE 8 or higher platforms.

2.3 The Executive Committees would like to ensure JSR submitters think about how their proposed technology relates to all of the Java platform editions. Please provide details here for which platform editions are being targeted by this JSR, and how this JSR has considered the relationship with the other platform editions.

This JSR is targeted for inclusion in the Java EE 8 platform.

2.4 What need of the Java community will be addressed by the proposed specification?

See 2.1 above.

2.5 Why isn't this need met by existing specifications?

See 2.1 above.

2.6 Please give a short description of the underlying technology or technologies:

See 2.1 above.

2.7 Is there a proposed package name for the API Specification? (i.e., javapi.something, org.something, etc.)

In APIs defined by this JSR will use the root package name javax.mvc.

2.8 Does the proposed specification have any dependencies on specific operating systems, CPUs, or I/O devices that you know of?

No.

2.9 Are there any security issues that cannot be addressed by the current security model?

No. This JSR will leverage existing HTTP security mechanisms provided by the platform.

2.10 Are there any internationalization or localization issues?

This JSR will use the I18N support in Java SE.

2.11 Are there any existing specifications that might be rendered obsolete, deprecated, or in need of revision as a result of this work?

No.

2.12 Please describe the anticipated schedule for the development of this specification.

Q3 2014 Expert Group formed

Q1 2015 Early Draft

Q3 2015 Public Review

Q1 2016 Proposed Final Draft

Q3 2016 Final Release

2.13 Please describe the anticipated working model for the Expert Group working on developing this specification.

The primary means of communication will be e-mail. We will solicit feedback from the community and leverage the open source development model.

2.14 Provide detailed answers to the transparency checklist, making sure to include URLs as appropriate:

MVC 1.0 will use a project site (https://mvc-spec.java.net) for announcements and to track all issues and disseminate information on the progress of the JSR.

- **Is the schedule for the JSR publicly available, current, and updated regularly?**

  Yes, the schedule will be available on the project page for the JSR at http://mvc-spec.java.net.

- **Can the public read and/or write to a wiki for the JSR?**

  No, we'll use a public mailing list for comments instead (users@mvc-spec.java.net).

- **Is there a publicly accessible discussion board for the JSR that you read and respond to regularly?**

  The mailing lists available at https://java.net/projects/mvc-spec/lists.

- **Have you spoken at conferences and events about the JSR recently?**

We plan to publicly announce the MVC 1.0 JSR at JavaOne 2014.

- **Are you using open-source processes for the development of the RI and/or the TCK?**

The Reference Implementation will be developed as an open source project on java.net. The final RI will be available from the download page. The TCK is not open source.

- **What are the Terms of Use required to use the collaboration tools you have prepared to use with the Expert Group, so that prospective EG members can judge whether they are compatible with the JSPA?**

The java.net terms of use are available at https://java.net/terms_of_use.

- **What is the location of your publicly-accessible Issue list? In order to enable EC members to judge whether Issues have been adequately addressed, the list must make a clear distinction between Issues that are still open, Issues that have been deferred, and those that are closed, and must indicate the reason for any change of state.**

All issues related to the specification will be tracked in the publicly accessible MVC specification project issue tracker (https://java.net/jira/browse/MVC_SPEC). The issue tracker link is also accessible from the JSR project page.

- **What is the mechanism for the public to provide feedback on your JSR?**

The user's e-mail alias users@mvc-spec.java.net.

- **Where is the publicly-accessible document archive for your Expert Group?**

Also available at java.net https://java.net/projects/mvc-spec/downloads.

- **Does the Community tab for my JSR have links to and information about all public communication mechanisms and sites for the development of my JSR?**

Yes, see https://mvc-spec.java.net/.

- **Do you have a Twitter account or other social networking feed which people can follow for updates on your JSR?**

Yes, @spericas and @mnriem.

- **Which specific areas of feedback should interested community members (such as the Adopt-a-JSR program) provide to improve the JSR (please also post this to your Community tab)?**

In general, any of the areas listed in Section 2.1. For more information, refer to the Community tab in the JSR page at jcp.org.

2.15 Please describe how the RI and TCK will de delivered, i.e. as part of a profile or platform edition, or stand-alone, or both. Include version information for the profile or platform in your answer.

The reference implementation will be made available standalone and also as part of the reference implementation for the Java EE 8 platform. The TCK will be made available standalone and as part of the Java EE CTS.

2.16 Please state the rationale if previous versions are available stand-alone and you are now proposing in 2.13 to only deliver RI and TCK as part of a profile or platform edition (See sections 1.1.5 and 1.1.6 of the JCP 2 document).

N/A.

2.17 Please provide the full text of the licenses that will apply to your Final Release Specification, Reference Implementation, and Technology Compatibility Kit, or provide links to the same.

Proposed licenses are as follows:

- Specification license
- RI license
    - Commercial use

      The RI will be available for commercial use under the CDDL 1.1 open source license, the GPLv2 with Classpath Exception open source license, or this RI license.

    - Non-Commercial use

  The RI will be available for non-Commercial use under the CDDL 1.1 open source license or the GPLv2 with Classpath Exception open source license.
- TCK license
    - Commercial use

      The TCK will be available for commercial use under this TCK license.

    - Non-Commercial use

As required by the Java Specification Participation Agreement (JSPA), the TCK will be licensed at no charge without support to qualified not-for-profit. The Compatibility Testing Scholarship Program will verify such qualification. Support may also be provided at no charge with approval of the scholarship board. For more information, please refer to:
http://www.oracle.com/technetwork/java/index-137188.html

2.18 Please describe the communications channel you have established for the public to observe Expert Group deliberations, provide feedback, and view archives of all Expert Group communications

The Expert Group will conduct business on a publicly readable alias. The public will have an alias on which to provide feedback and discuss issues related to the JSR. There will also be a publicly accessible JIRA and document archive. (See also 2.19 and 2.20 below.)

2.19 What is the URL of the Issue Tracker that the public can read, and how does the public log issues in the Issue Tracker?

https://java.net/jira/browse/MVC_SPEC

2.20 Please provide the location of the publicly accessible document archive you have created for the Expert Group.

https://java.net/projects/mvc-spec/downloads

Section 3: Contributions

3.1 Please list any existing documents, specifications, or implementations that describe the technology. Please include links to the documents if they are publicly available.

For more information on the MVC pattern, please refer to: http://en.wikipedia.org/wiki/Model-view-controller. There are numerous implementations of this pattern available for the Java platform. An example of such an implementation is the extension available in the JAX-RS Reference Implementation (https://jersey.java.net/documentation/2.4/user-guide.html#mvc).

3.2 Explanation of how these items might be used as a starting point for the work.

The MVC pattern is mature and well understood by the Java community. The EG will consider existing implementations and decide which, if any, to use as a starting point.