YmtdImplementation

Implementation

The first point: Standards

One of the touted advantages of JSP is that it is a "standard" and quite a few people like to hold this in high regard. So much so that they refuse to use any technology that is not "standard." Digging into the reality of this statement reveals that the important correct terminology is that JSP is a "Sun Standard Specification" and not strictly a "standard." This is important because JSP is really no more "standard" than Microsoft ASP or the PHP Group's PHP product. In other words, whatever tool you happen to be using becomes the "standard."

A small group within the Java Community Process (JCP) defines what JSP is. The fact of the matter is that there is a fairly high barrier to joining the JCP because an NDA must be signed, the project leads must approve your entry and in some cases a fee must be paid. One could even stretch as far as to say that the JSP specification is really a proprietary product of Sun!

It is important to note at this point that the primary author of this document (Jon Stevens) is a member of the JSR-053 which defines the Servlet and JSP specification's.

Inside JSR-053, it is clear that not everything is done in the open and decisions are made behind closed doors. Of course the participants could object, but Sun still is the binding force behind the decisions.

The next point: Complexity

JSP is both a specification as well as an implementation. There are various corporations (as well as Open Source) implementations of the specification. The JSP specification is not a particularly easy thing to implement. There are many complex systems involved, some of which even require special hooks into the Servlet Engine in order to obtain optimal performance.

The JSP specification is relatively new (it is still in a 1.x phase). This means that the specification also has several places in it that are not as well defined as others, leaving some of the specific details to be determined by the implementation. Not only does this mean that there is plenty of places to make mistakes, but it also means that JSP template code has the possibility of not behaving the same across implementations. This makes testing JSP based applications across several different containers a nightmare. Especially if there are essoteric bugs (meaning bugs that only show up under extreme conditions) in one and not in the other.

Part of the founding reason for creating the Jakarta Project and having Sun release the source code to Jasper (the JSP reference implementation) is to encourage vendors to adopt a single base for their source code. Unfortunately, this has not happened. There is compatibility testing suites available, however there is no policy in place requiring vendors to pass the tests. Nor is there a place that shows vendors who do not pass the tests in order to publicly humiliate them into submission.

The final point: Velocity

The definition of what Velocity is is set forth by the developers on the project. They have created a comprehensive testing suite that is used for regression testing to ensure that all changes to Velocity will not break existing templates. If you or your company ever wants to join the development of Velocity and help determine its direction that can be done easily by simply joining a mailing list and contributing to the development effort. This means that no one corporation will ever own the Velocity specification and it also means that Velocity will always work with your templates within any servlet engine.

You make the decision.

YmtdEmbeddedUsage <- Previous | Next -> YmtdHosting