Jelly and XSS prevention

Summary

If you are writing plugins for Jenkins 1.339 and later, put <?jelly escape-by-default='true'?> on every file.

Jelly uses the same expression syntax and semantics as JSP, which tends to make view pages vulnerable to cross-site scripting vulnerabilities. That is, the following seemingly harmless use of expression can result in XSS vulnerability (or a rendering problem) if fullName contains characters like '<'. See this document for more complete discussion of this problem.

```
<p>My name is ${it.fullName}</p>
```

To make it easier to avoid this problem, without incurring a backward incompatibility, Jelly scripts in Jenkins 1.339 and later can have the following processing instruction at the top:

```
<?jelly escape-by-default='true'?>
<p>My name is ${it.fullName}</p>
```

This transparently modifies the behavior of ${...} expressions within this XML file such that Jelly will perform HTML escaping automatically. This processing instruction is strongly recommended on all Jelly files, and it should be a part of your new Jelly file template. Note that this only affects the use of ${...} among PCDATA, and not in attribute values (this is so that Jelly tag invocations don't result in a surprising behavior.)

To cancel out the effect of escape-by-default='true' and allow expressions to produce mark up, use <j:out> like this:

```
<p>My name is <j:out value="${it.fullName}"/></p>
```

Also see JENKINS-5135 that tracks the task to have the core use this new feature.

Internationalization and XSS prevention

When using the ${%...} expression to refer to localized resources, localized resources are put into HTML without further escaping. This means your localized messages can contain markups. However, arguments to those messages, if any, are escaped before formatting. While this may sound complex, I believe this is the least intrusive, and it means the view like the following will be XSS-safe by default:

```
<?jelly escape-by-default='true'?>
<p>${%blurb(it.displayName}</p>

blurb=<a href="...">{0}</a>
```

If you have some arguments that legally contains raw HTML, then you can cancel this escaping-by-default behavior for arguments by using h.rawHtml, like this:

```
<?jelly escape-by-default='true'?>
<p>${%blurb(h.rawHtml(it.description)}</p>

blurb=<a href="...">{0}</a>
```