

Orion: Embracing the Web Open Tools Integration

Boris Bokowski, IBM Ottawa Lab

What Is Orion?

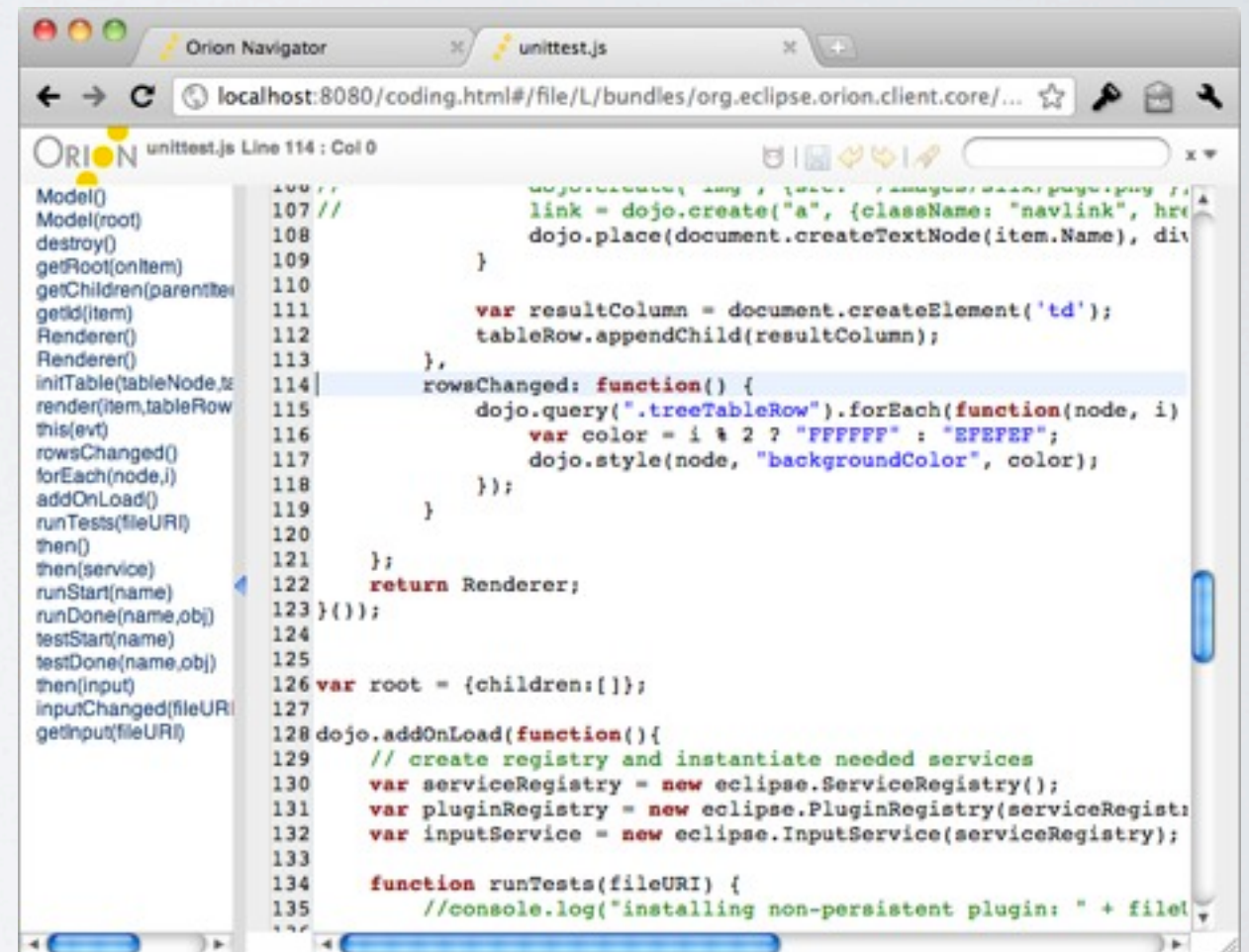
- Software development in a browser:
editing, navigating folders, searching, working with SCM...
- Set of linked web pages, NOT an IDE running in a browser.
 - Makes it easy to integrate other functionality even if hosted on different servers.
- Initial focus on web developers working on client-side JavaScript, HTML, CSS.

In a Browser?

- Bug tracking (Bugzilla, JIRA, Trac, Lighthouse, Rational RTC)
- Builds (e.g. Hudson)
- Code review (Bugzilla, Gerrit, GitHub)
- Documentation, Code Snippets
- Browsing code repositories (GitHub, ViewCVS, Rational RTC)
- Debugger (Firebug, WebKit Inspector)

Code Editor

- Fast
- Scalable
- Works in all desktop browsers
- Faster than desktop Eclipse editor!



The screenshot shows the Orion Navigator web-based code editor. The browser address bar indicates the URL is localhost:8080/coding.html#/file/L/bundles/org.eclipse.orion.client.core/... The editor window title is 'Orion Navigator' and the file name is 'unittest.js'. The code is displayed in a light-colored theme with a dark sidebar on the left containing a file tree. The main editor area shows JavaScript code with line numbers 107 through 135. The code includes comments and function definitions, such as 'rowsChanged' and 'runTests'. The 'rowsChanged' function uses 'dojo.query' and 'dojo.style' to iterate over table rows and change their background color based on their index. The 'runTests' function is partially visible at the bottom of the screenshot.

```
107 //
108
109
110
111
112
113
114 rowsChanged: function() {
115     dojo.query(".treeTableRow").forEach(function(node, i)
116         var color = i % 2 ? "FFFFFF" : "EFEFEF";
117         dojo.style(node, "backgroundColor", color);
118     });
119 }
120
121 };
122 return Renderer;
123 }());
124
125
126 var root = {children:[]};
127
128 dojo.addOnLoad(function(){
129     // create registry and instantiate needed services
130     var serviceRegistry = new eclipse.ServiceRegistry();
131     var pluginRegistry = new eclipse.PluginRegistry(serviceRegistry);
132     var inputService = new eclipse.InputService(serviceRegistry);
133
134     function runTests(fileURI) {
135         //console.log("installing non-persistent plugin: " + fileURI);
```


Orion Design Principles

- Regular Hyperlinks, back button, bookmarking, link sharing.
- Functionality on separate pages. Page = Task+Resource.
- Performant and Lightweight. Speed trumps power.
- Components should have value on their own.
- Low barrier of entry for adopters. Don't require technology buy-in.

Planning Meeting

- March 17/18 at SAP Labs, Palo Alto
- Broad participation: Siemens, SAP, RIM, Nuxeo, Nokia, Nitobi, Mozilla, Motorola, Microsoft, IBM, Google, GitHub, Firebug
- Take aways:
 - no controversy at all about tooling moving to the web
 - concrete next steps identified, for project and adopters

In Closing...

- We (IBM) would like to see a community to form around this forms at Eclipse, and are contributing a seed.
- Orion is in very early development. Looking for adopters and contributors!
- More information: eclipse.org/orion

Dual licensing for Orion (client side) under EPL/EDL

EPL and EDL

- Key difference: EPL requires derivative works be available under EPL (“copyleft”), EDL does not require this.
- EDL is basically the New (three-clause) BSD license.
- Seeking EPL/EDL dual licensing only for client-side code (HTML, CSS, JavaScript), not server-side code (Java).

Why EDL in addition to EPL?

- EPL for consumption at Eclipse, EDL (BSD) for Web community.
 - e.g. Firebug: debugger for Firefox, BSD
 - e.g. WebKit Inspector: debugger for WebKit browsers, BSD
 - Many other opportunities for adoption in the web community.
- We want broadest possible adoption to establish Orion as a platform / as a way to integrate web-based tools

Benefits of EDL

- Both Firebug and WebKit Inspector are potential adopters of the Orion editor. Both would need EDL. Adoption by even one of them would be a big win for Orion (“our skin in the game”)
- Web community has come to expect liberal licenses. e.g. Dojo: BSD(+AFL), jQuery: MIT(+GPL).
- EPL is virtually unknown in the Web community and a barrier for adoption (e.g. no clear “package boundary” as with Java for separating copyleft code from other code).

Potential Concerns

- EDL doesn't require to contribute back. Enabling "leeches"?
 - In practice, it is up to the consumer anyway to contribute back. The EPL only requires that derivative works be published somewhere. Investing real work to merge changes only happens where there is business interest, not because of a license.
 - Orion code is at the level of infrastructure & plumbing, not at the level of value add.

Questions? Comments?