New in GWT 2.5

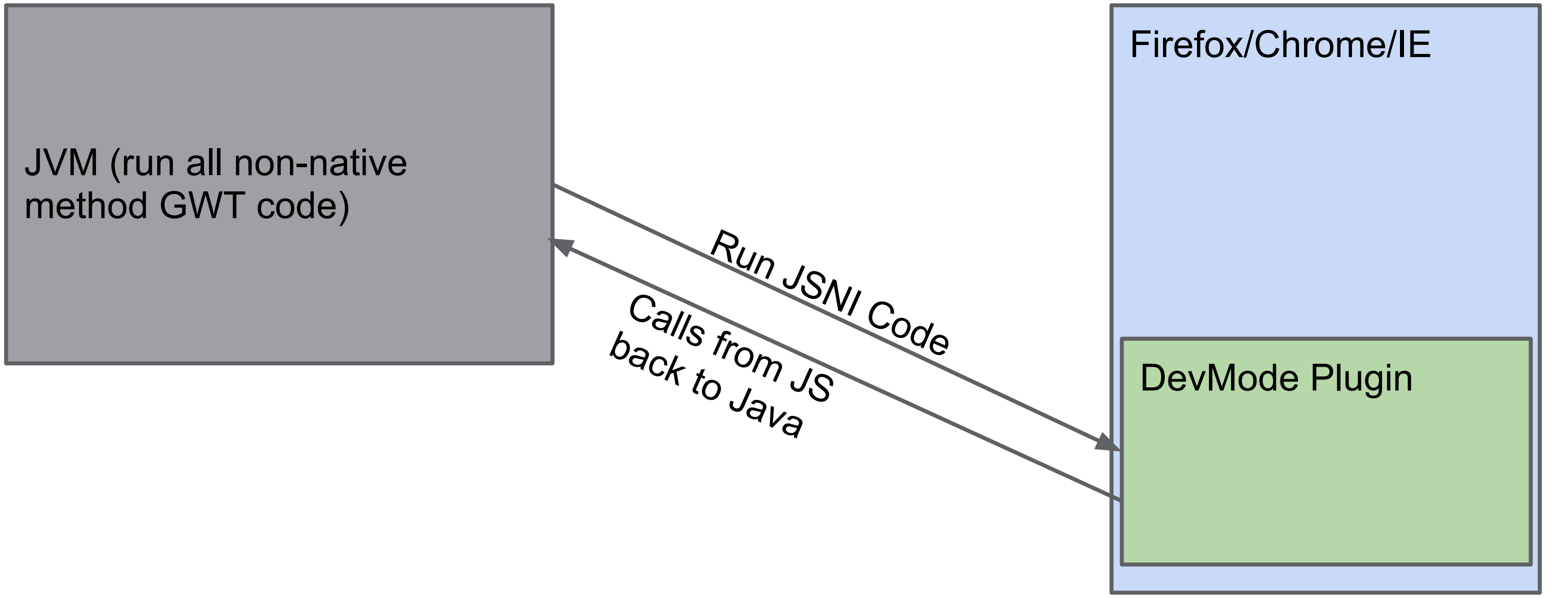
Super Dev Mode and Elemental

Ray Cromwell, Tech Lead, GWT

DevMode Refresher

- Runs Java code as straight bytecode on JVM
- JSNI Javascript code run by browser
- Browser NPAPI C++ plugin for TCP/IP communications
  - between JVM and Browser

DevMode architecture


JVM (run all non-native method GWT code)
Run JSNI Code
Calls from JS back to Java
Firefox/Chrome/IE
DevMode Plugin

DevMode

- Pros:
    - No compilation, Fast JVM refresh
    - JVM code runs very fast (faster than JS)
    - Debug in your favorite Java debugger
- Cons:
    - Lots of JSNI calls are slow (due to TCP/IP roundtrip turnaround)
    - Often must branch JSNI code specially for DevMode e.g. if(GWT.isScript())
    - Must maintain 3 implementations on 3 platforms = 9 plugins

The real problem

- Browsers used to update once a year, now they update every few weeks
- Plugin APIs have been a source of performance and security issues, browser vendors are discouraging their use
- Almost every browser release now breaks our plugins
- Some Browsers, like Safari, removed the APIs altogether, Chrome is on track to do so
- We can no longer keep up, and our Browser API hooks are being removed

Super Dev Mode - Pure Javascript, No Plugins

- Increase speed of compiler
  - Hopefully achieve < 10 second compiles
- Use Source Maps to enable Source Level Java Debugging
  - Debug Java in Web Browser!
  - Lose IDE/JVM debugger
  - Gain JSNI debugging, DOM/Event/XHR breakpoints
- Eventually, eliminate need to special case !GWT.isScript() code
- Debug mobile apps on Android and iOS6 (no source maps yet)
- DEMO

Introducing Elemental

Fast, Lightweight, "to the metal" modern web programming for GWT

- Auto-generated completely from actual WebIDL browser files used by JS engines
  - Don’t wait for hand wrappers, track the actual browser vendor specs/source
- 100% JavaScript overlay types behind Java interfaces which “melt away” after compilation
- Practically zero overhead
- Every HTML5 feature supported, even the bleeding edge
  - WebGL, WebAudio, WebSockets, WebRTC, Web Intents, Shadow DOM, File API, etc
- New Collection classes optimized for size and performance
  - map directly to underlying JS collections with no overhead
- New JSON library, no wrappers, no overhead, high performance
  - Collections and JSON work on server in non-GWT/non-JS environments too
- Excellent ‘thin’ library for direct mobile device development or desktop
- DEMO

Future Directions