Google Web Toolkit

for quick relief of AJAX pain.

Kelly Norton &
Miguel Méndez

Overview

the pleasure of using AJAX apps.

the pain of creating them.

getting some pain relief with GWT.

the tutorial part.

questions.

The pleasure of AJAX

The pleasure of AJAX (for users)

familiar web feeling with greatly improved user experience.

no installation.

safer.

The pleasure of AJAX (for servers)

UI state is maintained on client.

fewer bits go down the wire.

leverages client CPU and RAM.

The pain of AJAX.

The pain of AJAX.

JavaScript is an elegant, expressive disaster.

…with poor IDE support.

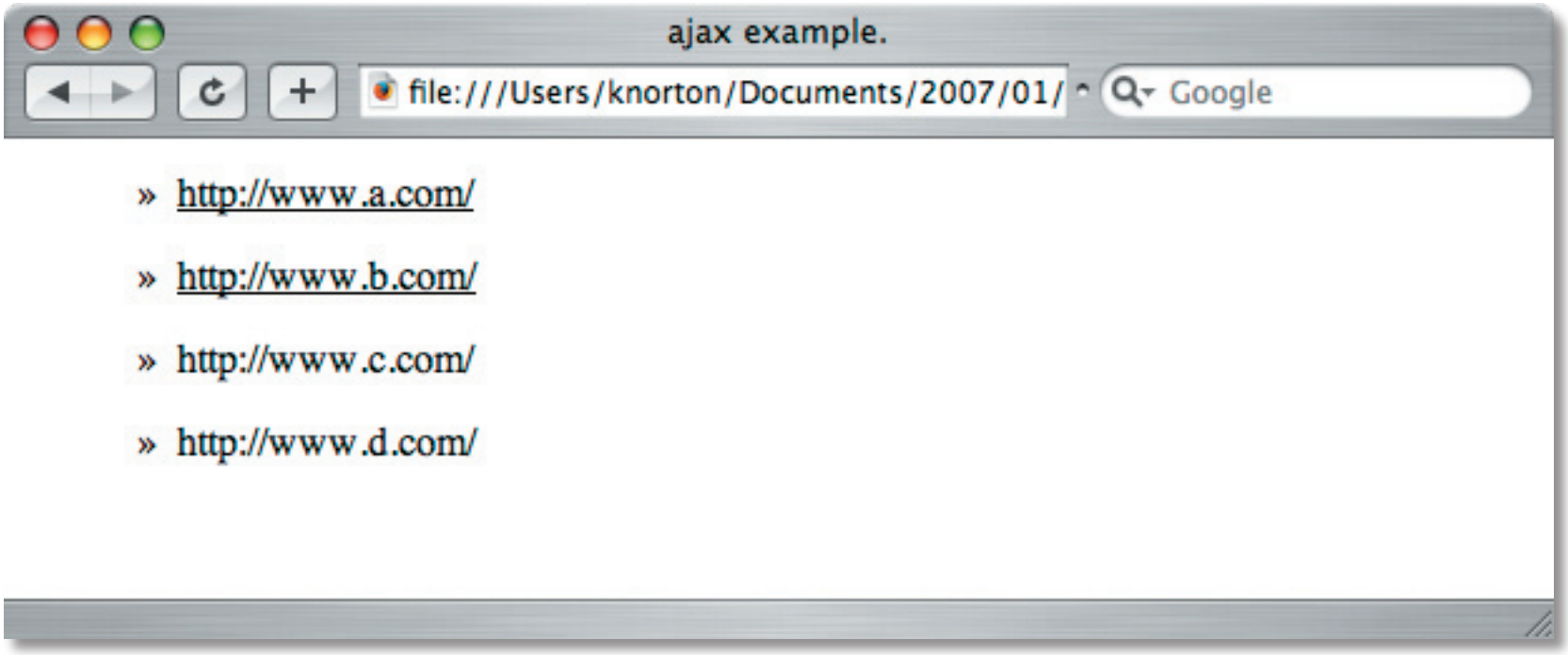
…and inadequate debuggers.

The pain of AJAX.

```
function createLinksForUrls(listOfUrls) {
  for (var i=0;i<listOfUrls.length;++i) {
    var a = listOfUrls[i];
    createLink(a).onclick = function() {
      return confirm("Browse to " + a + "?");
    };
  }
};
```

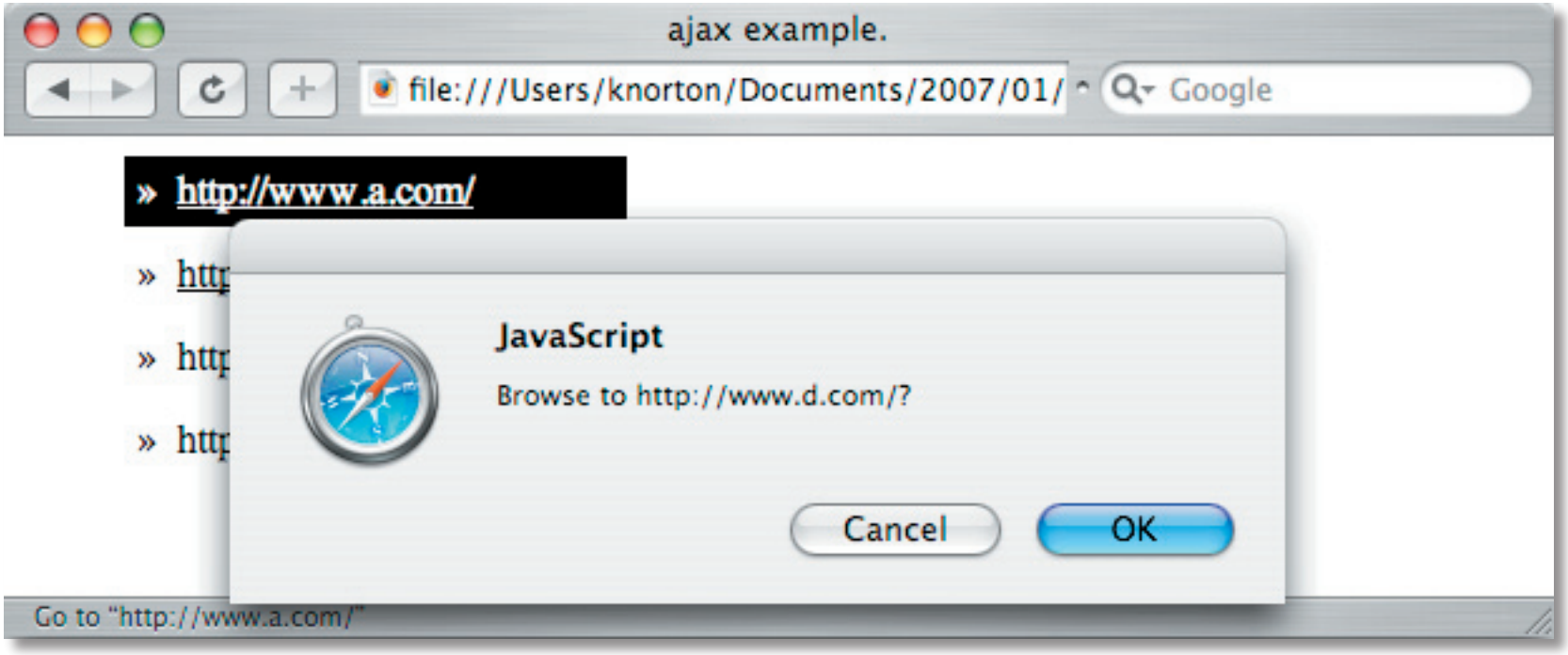

The pain of AJAX.

looks good, let's click on one.



The pain of AJAX.

curse you, binding rules!



The pain of AJAX.

```
function createLinksForUrls(listOfUrls) {
  for (var i=0;i<listOfUrls.length;++i) {
    var a = listOfUrls[i];
    createLink(a).onclick = function() {
      return confirm("Browse to " + a + "?");
    };
  }
};
```


The pain of AJAX.

```
/* elegant? */ ;(function() {
  document.addEventListener('keyup',
    (function() {
      var kh = function(e) { /* key handler */ };
      window.unload = (function(onunload) {
        var ul = function() {
          document.removeEventListener(
            'keyup',kh,true);
        };
        return function() {
          ul.call(window);
          if (onunload)
            onunload.call(window);
        };
      })(window.unlaod);
      return kh; })(), true); })();
```


The pain of AJAX.

```
/* expressive? */ ;(function() {
  document.addEventListener('keyup',
    (function() {
      var kh = function(e) { /* key handler */ };
      window.unload = (function(onunload) {
        var ul = function() {
          document.removeEventListener(
            'keyup',kh,true);
        };
        return function() {
          ul.call(window);
          if (onunload)
            onunload.call(window);
        };
      })(window.unlaod);
      return kh; })(), true); })();
```


The pain of AJAX.

tools and IDE's

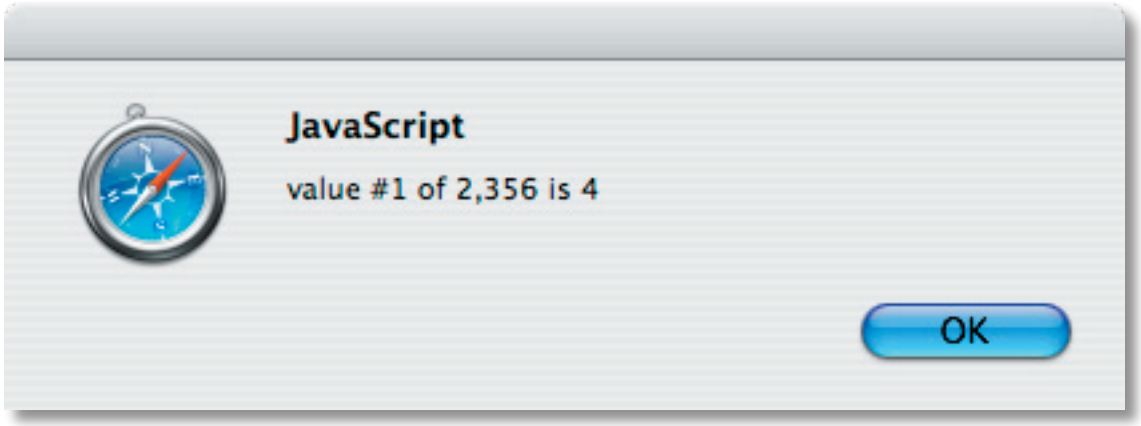
…they're getting better.

…try to depend on common idioms.

…but there's a limit to how good they can get.

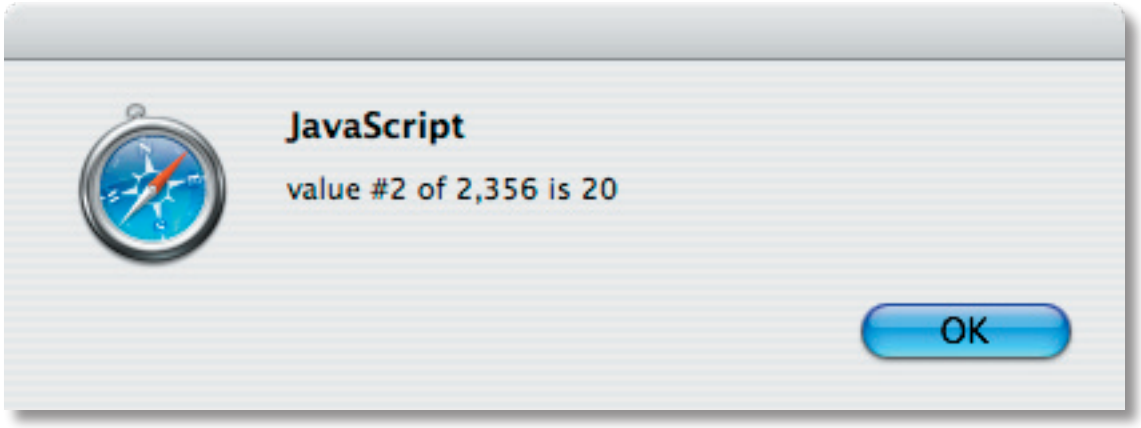
The pain of AJAX.

and let's not get started on debugging.



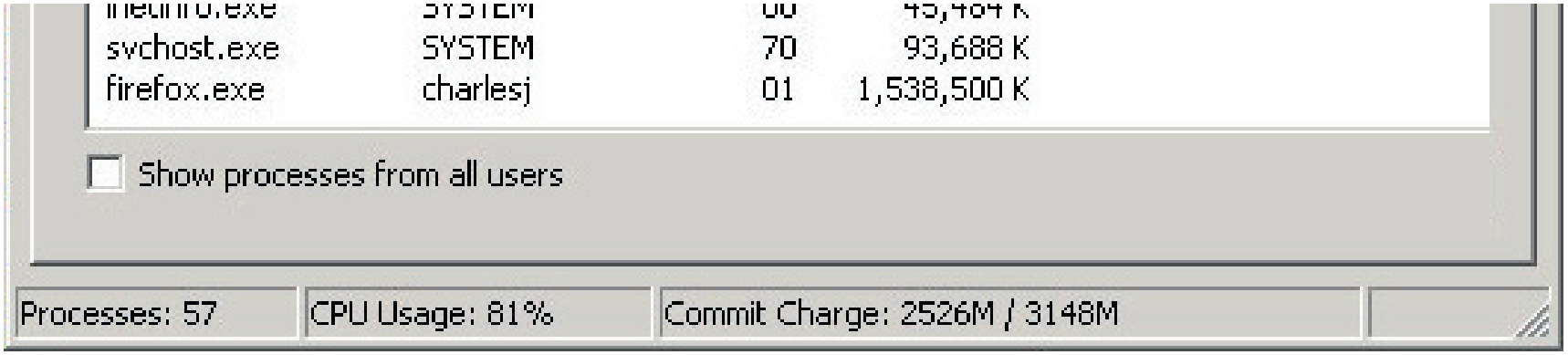
The pain of AJAX.

no seriously, this could take a while.



The pain of AJAX.

then there are memory “leaks”



and contrary to popular belief, they all seem to leak.

The pain of AJAX.

ok, that's javascript. here's a more thorough list of what's involved (we'll use regexp, to keep it short):

HTTPS?, [DX]?HTML (3.2|4.0),
CSS[1-3]
DOM Level[0-3]
(Java|ECMA|J|VB)Script
(X|VR?|Math)ML
SVG, Canvas, Flash
JSONP?, SOAP, XML-RPC

The pain of AJAX.

oh, don't forget to make the back
and forward buttons work properly.

quick relief of AJAX pain

quick relief of AJAX pain

in four easy steps…

① write your apps in java, using a UI lib that looks like a UI lib, and use javascript if you need to.

② use a standard java debugger and run your unit tests in junit.

③ compile and deploy.

④ profit.

quick relief of AJAX pain

STEP #1: writing in java…

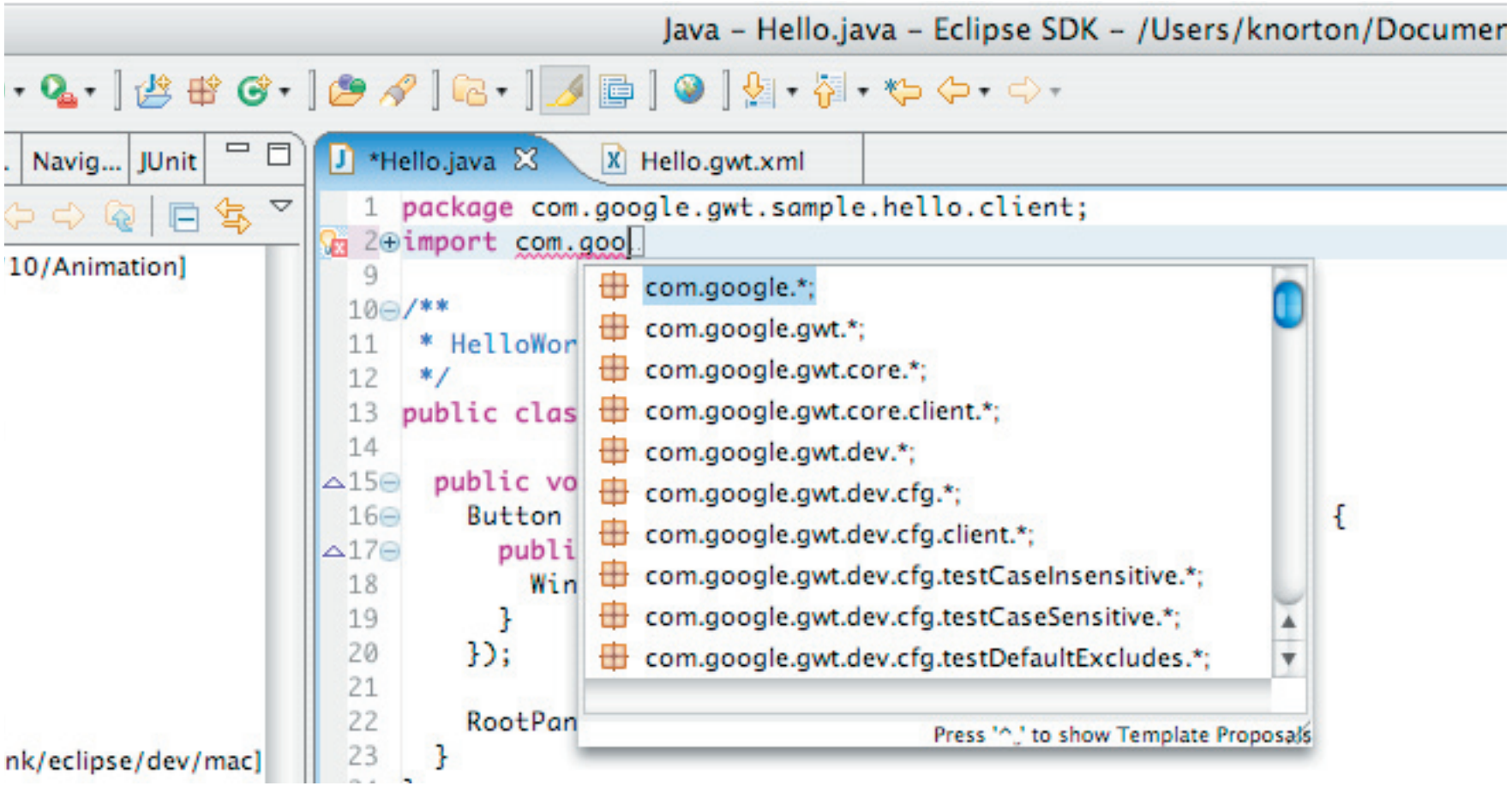
```
public class Hello implements EntryPoint {
  public void onModuleLoad() {

    Button b = new Button("Click me",
        new ClickListener() {
          public void onClick(Widget sender) {
            Window.alert("Hello, AJAX");
          }
        });

    RootPanel.get().add(b);
  }
}
```

quick relief of AJAX pain

really writing in java…



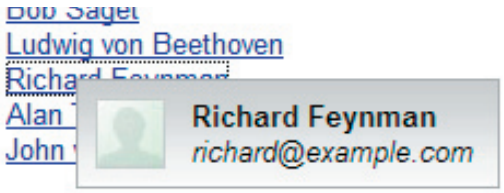
quick relief of AJAX pain

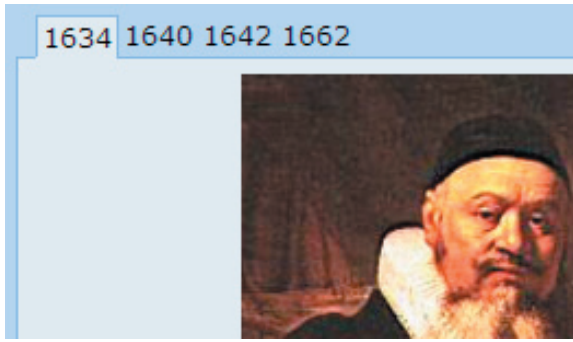
but, if we missed something… JSNI.

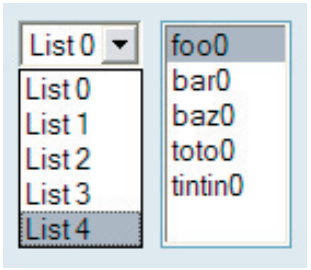
```
public native void appendToFoo(String s) /*-{
  var parent = $doc.getElementById("foo");
  var child = $doc.createTextNode(s);
  parent.appendChild(child);
}-*/;
```

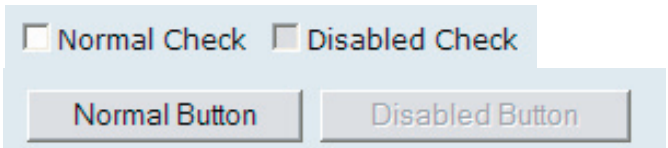

quick relief of AJAX pain

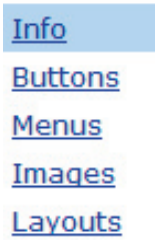
but for most UI, we already have it covered.

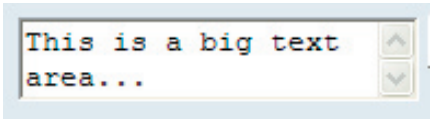


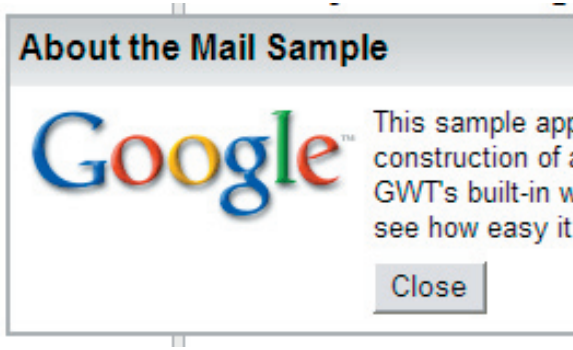


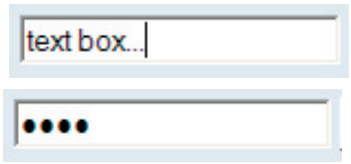


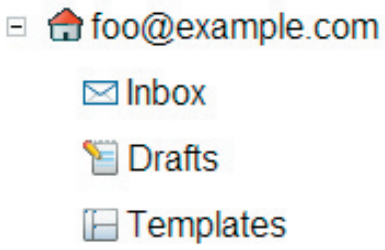


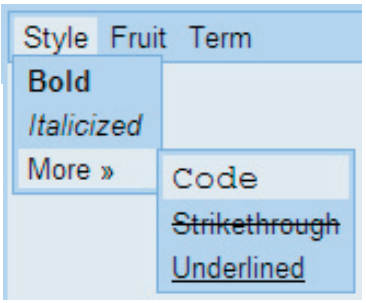


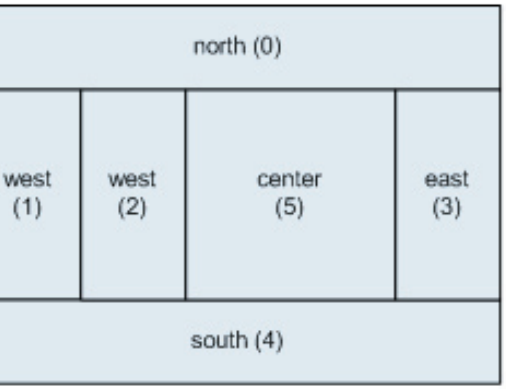


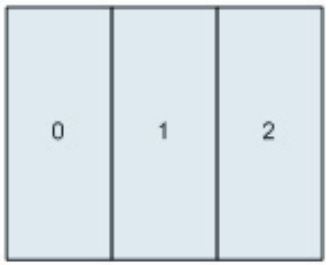


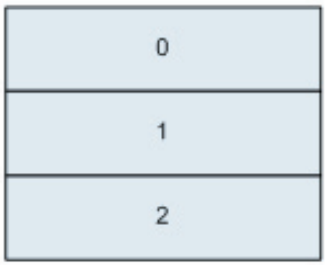


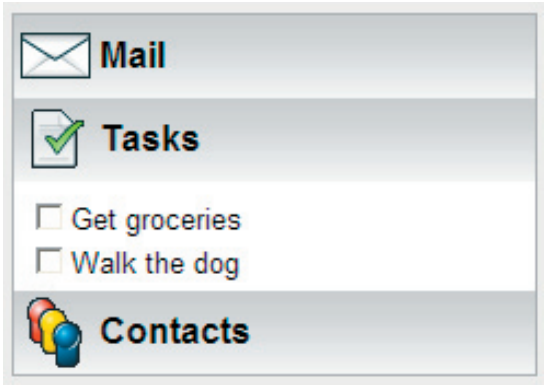












quick relief of AJAX pain

UI framework also properly detaches nodes so that memory leaks are greatly reduced.

this happens as you go, because ignoring long running applications is not really an option. I mean, we are looking at apps that stay in the browser for days, not seconds.

quick relief of AJAX pain

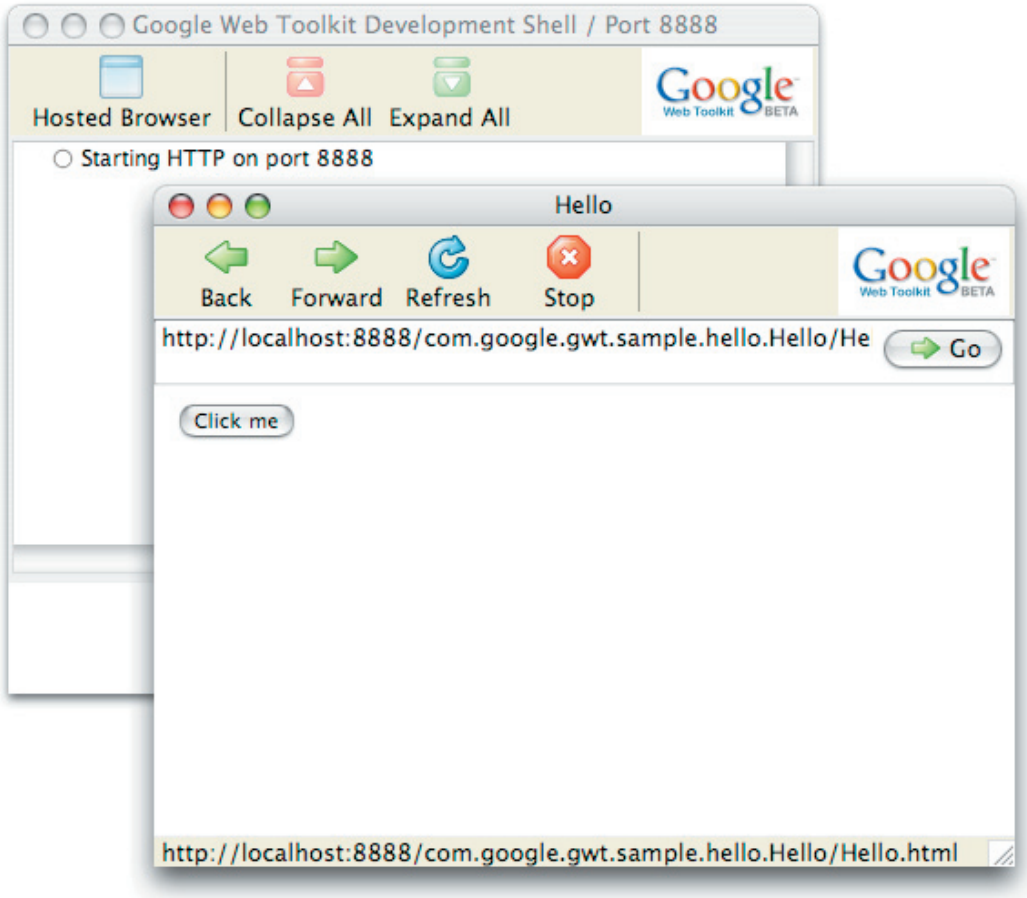
and, of course, they work with css.

quick relief of AJAX pain

and we even have a History class to tend to the pesky back and forward buttons.

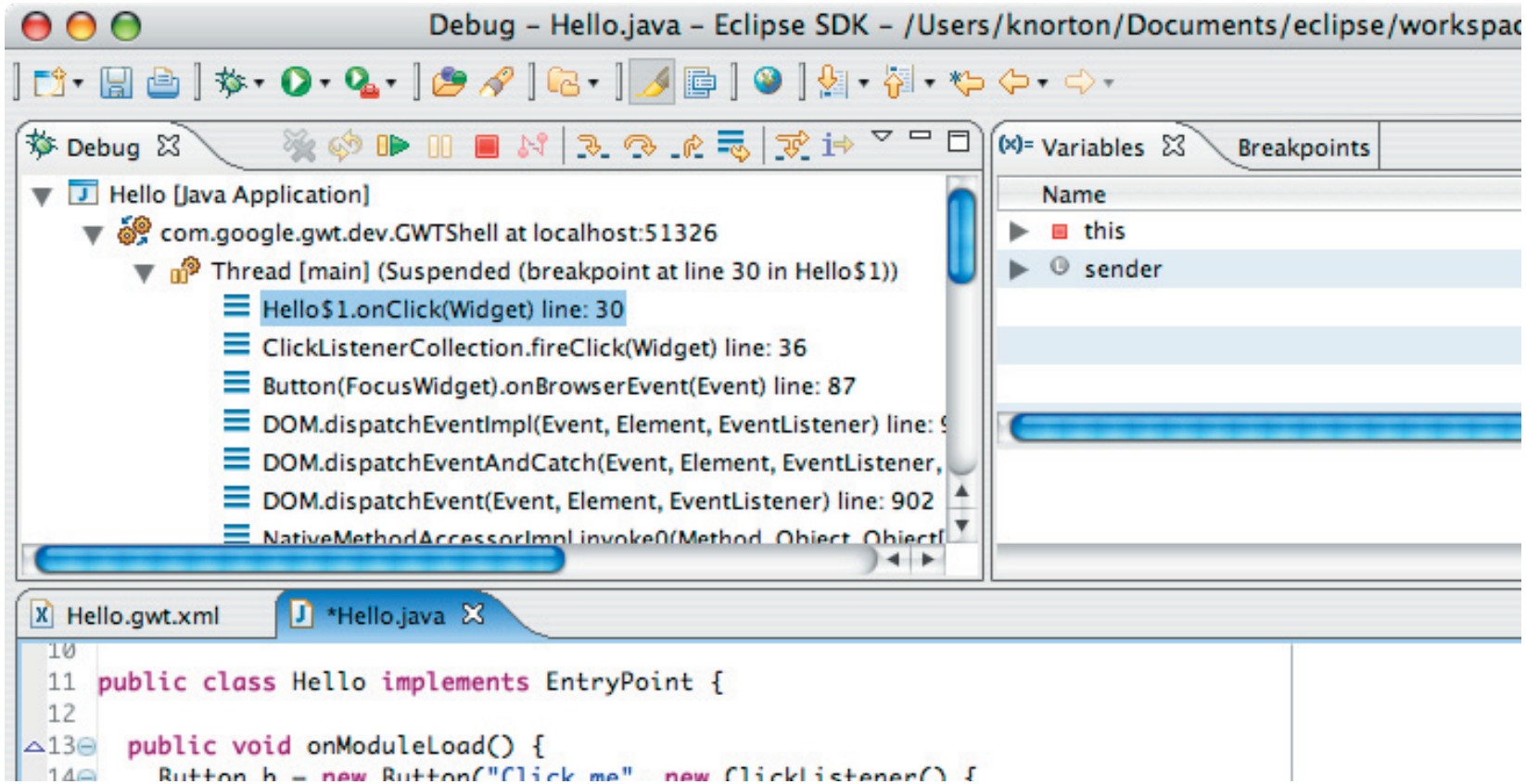
quick relief of AJAX pain

STEP #2: debugging, real browser



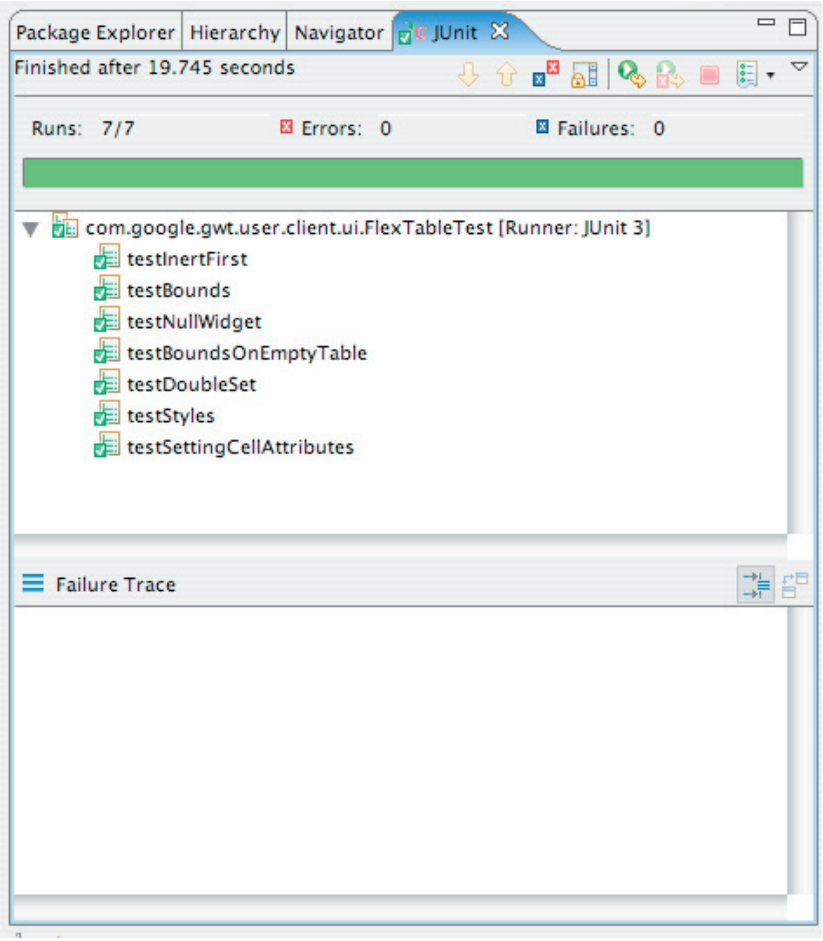
quick relief of AJAX pain

…and real debugger



quick relief of AJAX pain

…even your favorite junit runner



quick relief of AJAX pain

STEP #3: compile.

the GWT compiler works off of java source (uses JDT at the front end).

many of the standard JRE classes are available.

some are not, including reflection and dynamic class loading. why not?…

quick relief of AJAX pain

we're crazy about optimization.

whole program optimization, with inlining, dead code removal, type tightening, etc.

for example:

```
Shape s = new Circle(2);  // radius of 2
double a = s.getArea();
```

becomes:

```
Circle s = new Circle(2); // radius of 2
double a = (s.radius * s.radius * Math.PI);
```


quick relief of AJAX pain

we do a number on the javascript;
you have better source now.

```
function ne(oe){var pe=qe(new
re(),'Click me',se(new
te(),oe));ue(ve(),pe);}
function xe(ye){ze('Hello, AJAX');}
```


quick relief of AJAX pain

plus a lot of *advanced* features:

deferred binding - allows you to precompile for different runtime properties.

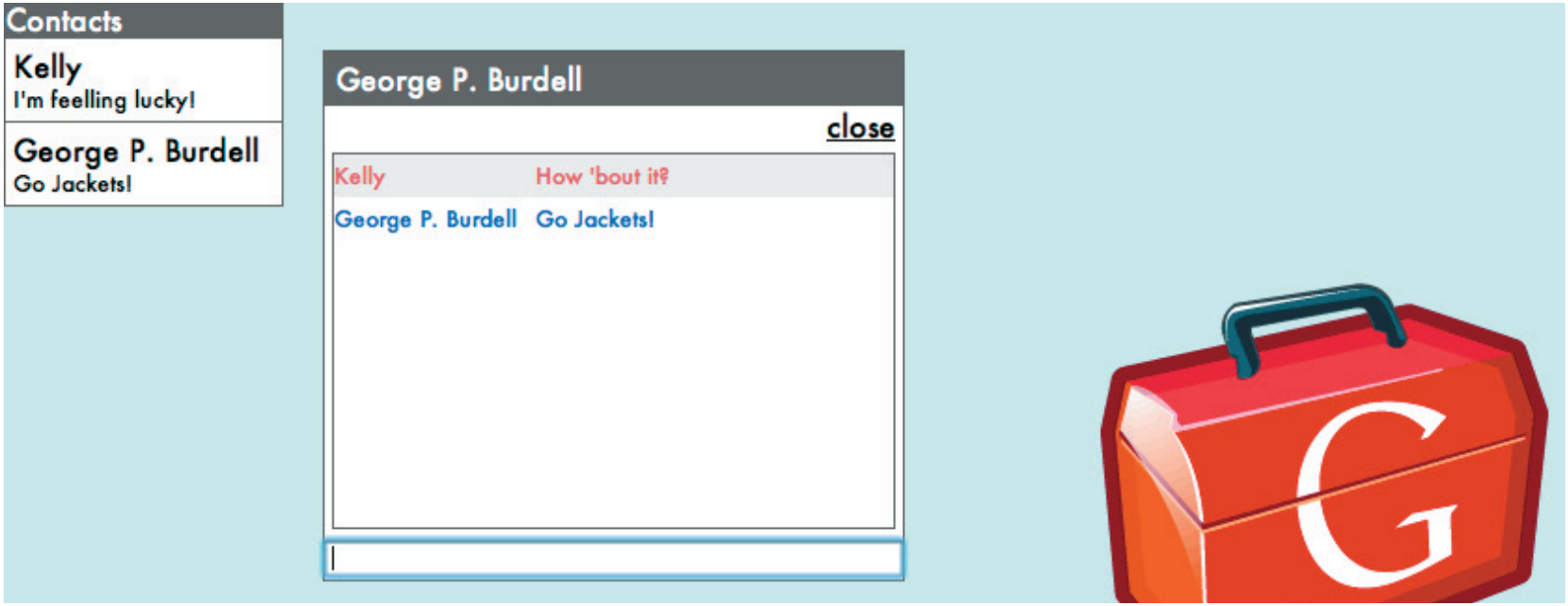
generators - do code generation at compile time (we use this for i18n, RPC, junit).

quick relief of AJAX pain

STEP #4: Profit

this one is left as exercise for the reader.

let's build chattr.



to play along at home:
http://code.google.com/p/gwt-eclipsecon-chat/

(template project is available as a download)

thanks for watching.

We're on google code,
http://code.google.com/webtoolkit/

and google groups,
Google-Web-Toolkit@googlegroups.com

and we're opensource, so come pitch in
http://google-web-toolkit.googlecode.com/svn/