

Testing in WebKit-EFL

From 0% to 99% in 6 months

Leandro Pereira

ProFUSION Embedded Systems

<http://profusion.mobi>

February 04, 2012

WebKit



WebKit



WebKit



- ▶ Heart of Epiphany, Chromium, Safari, Konqueror, and Eve
- ▶ Good standards compliance

WebKit



- ▶ Heart of Epiphany, Chromium, Safari, Konqueror, and Eve
- ▶ Good standards compliance
- ▶ Comprised of WebCore

WebKit



- ▶ Heart of Epiphany, Chromium, Safari, Konqueror, and Eve
- ▶ Good standards compliance
- ▶ Comprised of WebCore, JavaScriptCore

WebKit



- ▶ Heart of Epiphany, Chromium, Safari, Konqueror, and Eve
- ▶ Good standards compliance
- ▶ Comprised of WebCore, JavaScriptCore, one WebKit for each port

WebKit



- ▶ Heart of Epiphany, Chromium, Safari, Konqueror, and Eve
- ▶ Good standards compliance
- ▶ Comprised of WebCore, JavaScriptCore, one WebKit for each port, and WTF.

Testing in WebKit

- ▶ Bug fixed → new test



Testing in WebKit

- ▶ Bug fixed → new test
- ▶ New feature → new test



Testing in WebKit



- ▶ Bug fixed → new test
- ▶ New feature → new test
- ▶ Over 9000 28000 tests
 - ▶ Some imported from other test suites (W3C, Mozilla, etc)
 - ▶ Some written by WebKit contributors

Testing in WebKit



- ▶ Bug fixed → new test
- ▶ New feature → new test
- ▶ Over ~~9000~~ 28000 tests
 - ▶ Some imported from other test suites (W3C, Mozilla, etc)
 - ▶ Some written by WebKit contributors
 - ▶ One of the reasons WebKit repo is so large (1.8GiB just for tests and expected results)
- ▶ Most tests are port-independent

Testing in WebKit



- ▶ Bug fixed → new test
- ▶ New feature → new test
- ▶ Over ~~9000~~ 28000 tests
 - ▶ Some imported from other test suites (W3C, Mozilla, etc)
 - ▶ Some written by WebKit contributors
 - ▶ One of the reasons WebKit repo is so large (1.8GiB just for tests and expected results)
- ▶ Most tests are port-independent
- ▶ Test results are mostly port-dependent

Testing tools

DumpRenderTree (DRT)

- ▶ Works just like an automated web browser; JavaScript:
 - ▶ Controls the output type
 - ▶ Injects keypresses
 - ▶ Tells when the test is done

Testing tools

DumpRenderTree (DRT)

- ▶ Works just like an automated web browser; JavaScript:
 - ▶ Controls the output type
 - ▶ Injects keypresses
 - ▶ Tells when the test is done
- ▶ Historical name
 - ▶ Dumps the render tree
 - ▶ Dumps the textual representation
 - ▶ Dumps screenshots

Testing tools

DumpRenderTree (DRT)

```
layer at (0,0) size 800x600
  RenderView at (0,0) size 800x600
layer at (0,0) size 800x371
  RenderBlock {HTML} at (0,0) size 800x371
    RenderBody {BODY} at (8,3) size 784x352 [bgcolor=#FFFFFF]
      RenderBlock {DIV} at (0,0) size 784x24
        RenderBlock (floating) {DIV} at (0,0) size 377x23
          RenderInline {NOBR} at (0,0) size 377x16
            RenderInline {B} at (0,0) size 29x16
              RenderText {#text} at (0,1) size 29x16
                text run at (0,1) width 29: "Web"
              RenderText {#text} at (35,1) size 4x16
                text run at (35,1) width 4: " "
            RenderInline {A} at (0,0) size 42x16 [color=#0000CC]
              RenderText {#text} at (39,1) size 42x16
                text run at (39,1) width 42: "Images"
              RenderText {#text} at (87,1) size 4x16
                text run at (87,1) width 4: " "
            RenderInline {A} at (0,0) size 40x16 [color=#0000CC]
              RenderText {#text} at (91,1) size 40x16
                text run at (91,1) width 40: "Videos"
              RenderText {#text} at (137,1) size 4x16
```

X

Testing tools

ImageDiff

- ▶ Just like diff, but for images
- ▶ Output is another image
- ▶ Used to compare screenshots

Testing tools

run-webkit-tests

- ▶ Runs DumpRenderTree
- ▶ Compares output with either standard Unix `diff` or `ImageDiff`
- ▶ If it's equal to the baseline, the test passes

Testing tools

run-webkit-tests

- ▶ Runs DumpRenderTree
- ▶ Compares output with either standard Unix `diff` or `ImageDiff`
- ▶ If it's equal to the baseline, the test passes
- ▶ Port-independent

Testing tools

run-webkit-tests

- ▶ Runs DumpRenderTree
- ▶ Compares output with either standard Unix `diff` or `ImageDiff`
- ▶ If it's equal to the baseline, the test passes
- ▶ Port-independent
- ▶ Changed recently from a blob of Perl code to much nicer Python
- ▶ New version is multiprocessing-aware, more resilient to crashing tests, and works better with flaky tests

Other tools

Don't you love scripts?

- ▶ `find-drt-baselines.py`
 - ▶ Compares output of EFL's DRT with other ports expected files
 - ▶ If blocks are the same, and their geometry is within a certain threshold, consider EFL's output correct

Other tools

Don't you love scripts?

- ▶ `find-drt-baselines.py`
 - ▶ Compares output of EFL's DRT with other ports expected files
 - ▶ If blocks are the same, and their geometry is within a certain threshold, consider EFL's output correct
 - ▶ Kind of cheating, but changes in behaviour are noticed
 - ▶ Hacky code, so not upstreamed

75%

- ▶ 75% of tests passes in WebKit-EFL using port-independent baselines

75%

- ▶ 75% of tests passes in WebKit-EFL using port-independent baselines
- ▶ But we don't implement everything...
- ▶ ...so we have about 75% of coverage

75%

- ▶ 75% of tests passes in WebKit-EFL using port-independent baselines
- ▶ But we don't implement everything...
- ▶ ...so we have about 75% of coverage
- ▶ Even then, to get to 99% of these was hard

99%?



WHY NOT 100%?

Debugging WebKit is bad for the environment



- ▶ Linking with debugging symbols takes a good while
- ▶ Machine is pretty much useless while linking
- ▶ gdb crashes

At least there is icecc

The screenshot displays the Picolé web interface. On the left, the 'Hosts' section shows a central node 'the quiet' connected to various other nodes: archer, isa, eturko-laptop, horstader, bartball, bart, annie, nadine, whiterabbit, tycoon, frankenstein, ida, jgatal, voyager, zatopka, vader, and navi. A 'Show as: List · Star' link is visible. Below the diagram, the text 'Recommended build command: make -j37' is shown. On the right, the 'Jobs' section displays a list of build jobs, including source files and compilation commands. The jobs list includes:

- C src/lib/elm_box.c local on satanna
- C src/lib/elm_mapbuf.c local on satanna
- C src/lib/elm_icon.c local on satanna
- C++ WebCore/html/DataGridColumn.cpp by tycoon on vader
- C++ platform/text/LineEnding.cpp by vader on vader
- C++ platform/text/Hyphenation.cpp by vader on frankenstein
- C++ WebCore/html/DataGridColumnList.cpp by tycoon on horstader
- C++ .ccache/tmp/SVGFEdiff.tmp.navi.20754.11 by navi on archer
- C++ .ccache/tmp/SVGFEConv.o.tmp.navi.20749.11 by navi on vader
- C++ platform/text/RegularExpression.cpp by vader on whiterabbit
- C++ WebCore/html/DataComponents.cpp by tycoon on lisa
- C++ platform/text/SegmentedString.cpp by vader on frankenstein
- C++ WebCore/html/FTPDirectoryDocument.cpp by tycoon on ada
- C++ WebCore/html/FormDateList.cpp by tycoon on vader
- C++ WebCore/html/HTMLAllCollection.cpp local on tycoon
- C++ platform/text/String.cpp local on vader
- C++ WebCore/html/HTMLAnchorElement.cpp by tycoon on ada
- C++ .ccache/tmp/SVGFEDisp1.tmp.navi.20763.11 by navi on whiterabbit
- C++ platform/text/StringBuilder.cpp by vader on frankenstein
- C++ WebCore/html/HTMLAppletElement.cpp by tycoon on annie
- C++ WebCore/html/HTMLAreaElement.cpp by tycoon on annie
- C++ platform/text/TextBoundaries.cpp by vader on whiterabbit
- C++ WebCore/html/HTMLBRElement.cpp local on tycoon
- C++ WebCore/html/HTMLBaseElement.cpp local on tycoon
- C++ WebCore/html/HTMLBaseFontElement.cpp by tycoon on whiterabbit
- C++ .ccache/tmp/SVGFEFlood.tmp.navi.20778.11 by navi on vader
- C++ WebCore/html/HTMLBlockquoteElement.cpp by tycoon on whiterabbit
- C++ platform/text/TextCodec.cpp by vader on frankenstein
- C++ WebCore/html/HTMLBodyElement.cpp local on tycoon
- C++ platform/text/TextCodecLatin1.cpp local on vader
- C++ platform/text/TextCodecUTF16.cpp local on vader
- C src/lib/elm_entry.c by satanna on lisa
- C++ WebCore/html/HTMLButtonElement.cpp local on tycoon
- C++ WebCore/html/HTMLCanvasElement.cpp local on tycoon
- C src/lib/elm_list.c by satanna on lisa
- C++ WebCore/html/HTMLCollection.cpp by tycoon on horstader
- C src/lib/elm_toolbar.c by satanna on horstader
- C++ platform/text/TextCodecUserDefined.cpp by vader on frankenstein
- C++ platform/text/TextEncoding.cpp by vader on nadine
- C src/lib/elm_bubble.c by satanna on frankenstein
- C src/lib/elm_slider.c by satanna on whiterabbit

Amount of patches

- ▶ In 6 months, around 100 patches were produced to implement DRT, ImageDiff and fix bugs
- ▶ 60% of these were bug fixes
- ▶ The rest were infrastructure changes and the tools themselves

Milestones

Period	Milestone
1 month	DumpRenderTree running
2 months	75% of tests passing
3 months	80% of tests passing
4 months	87% of tests passing
5 months	92% of tests passing
5.5 months	99% of tests passing
6 months	EFL baselines upstreamed

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.
- ▶ Smart pointers are your friend. Except when they're not.

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.
- ▶ Smart pointers are your friend. Except when they're not.
- ▶ Don't trust your debugger.

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.
- ▶ Smart pointers are your friend. Except when they're not.
- ▶ Don't trust your debugger.
- ▶ But **do** trust Valgrind.

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.
- ▶ Smart pointers are your friend. Except when they're not.
- ▶ Don't trust your debugger.
- ▶ But **do** trust Valgrind.
- ▶ Don't try linking WebKit with debugging symbols on a Pandaboard. Unless you can spare around 20h to link. Set up a cross-compiler early.

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.
- ▶ Smart pointers are your friend. Except when they're not.
- ▶ Don't trust your debugger.
- ▶ But **do** trust Valgrind.
- ▶ Don't try linking WebKit with debugging symbols on a Pandaboard. Unless you can spare around 20h to link. Set up a cross-compiler early.
- ▶ Set up 32- and 64-bit machines early.

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.
- ▶ Smart pointers are your friend. Except when they're not.
- ▶ Don't trust your debugger.
- ▶ But **do** trust Valgrind.
- ▶ Don't try linking WebKit with debugging symbols on a Pandaboard. Unless you can spare around 20h to link. Set up a cross-compiler early.
- ▶ Set up 32- and 64-bit machines early.
- ▶ 4GB of RAM is not enough.

Lessons learned

- ▶ WebKit is a C++ library: don't code in C like you'd do on a PDP – even if others do the same.
- ▶ Smart pointers are your friend. Except when they're not.
- ▶ Don't trust your debugger.
- ▶ But **do** trust Valgrind.
- ▶ Don't try linking WebKit with debugging symbols on a Pandaboard. Unless you can spare around 20h to link. Set up a cross-compiler early.
- ▶ Set up 32- and 64-bit machines early.
- ▶ 4GB of RAM is not enough.

Thank you!

leandro@profusion.mobi
acidx on Freenode

