BlinkOn: Architecture

Eric Seidel and Adam Barth
9/24/2013

Times are a changin’

Compared to 5 years ago

- Browser compatibility is much better
- Better standards (HTML5)
- Mobile

Who are the Joneses?

- Native mobile apps

Who are the Joneses?

- Native mobile apps
- Why is their grass greener?

Who are the Joneses?

- Native mobile apps

- Why is their grass greener?
  - Platform integration

Who are the Joneses?

- Native mobile apps

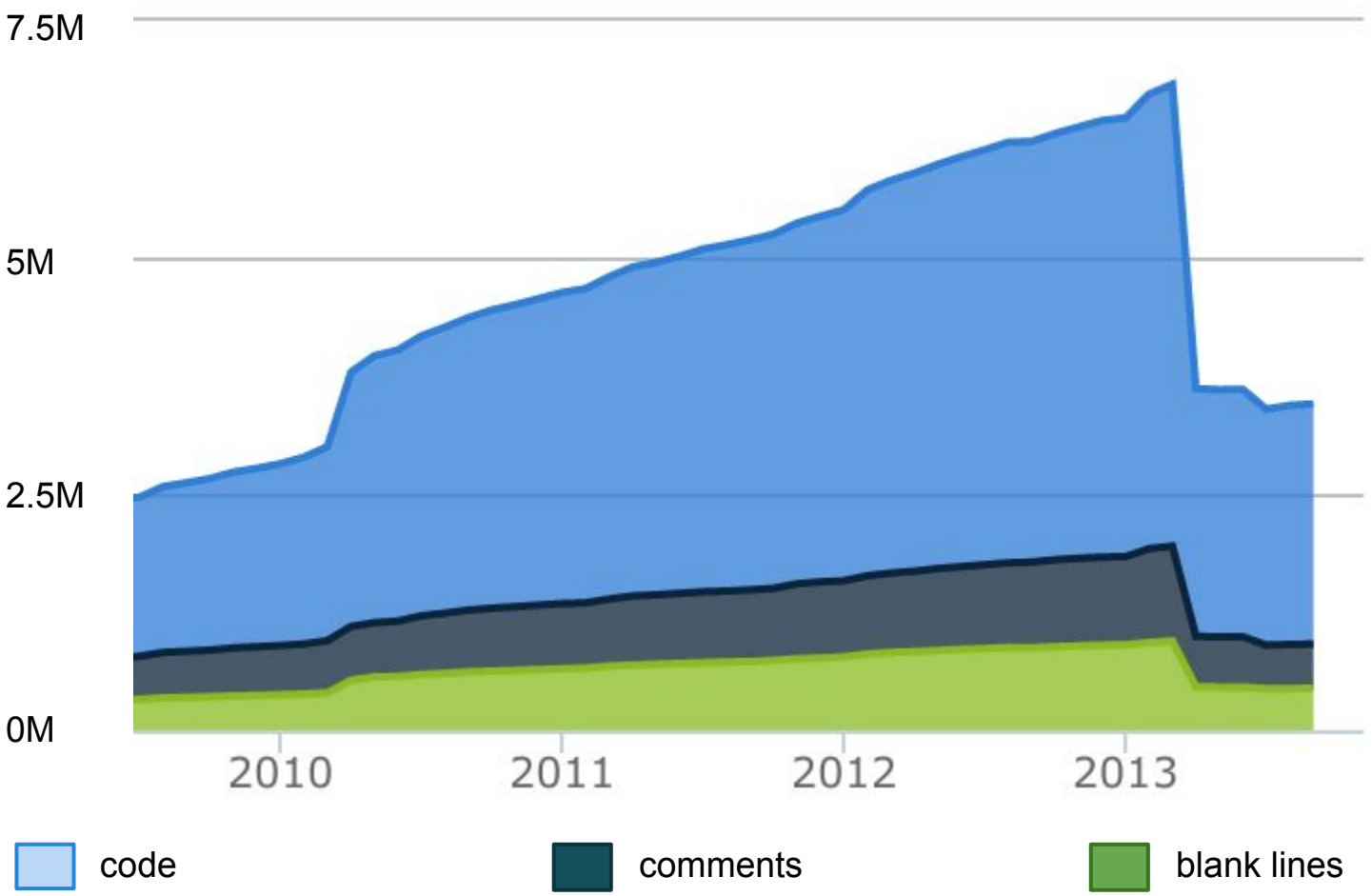
- Why is their grass greener?
  - Platform integration
  - Performance

Focus

- Do what we do faster
  - Parsing, DOM, HTML, SVG, CSS
  - Style resolution, Layout, Rendering

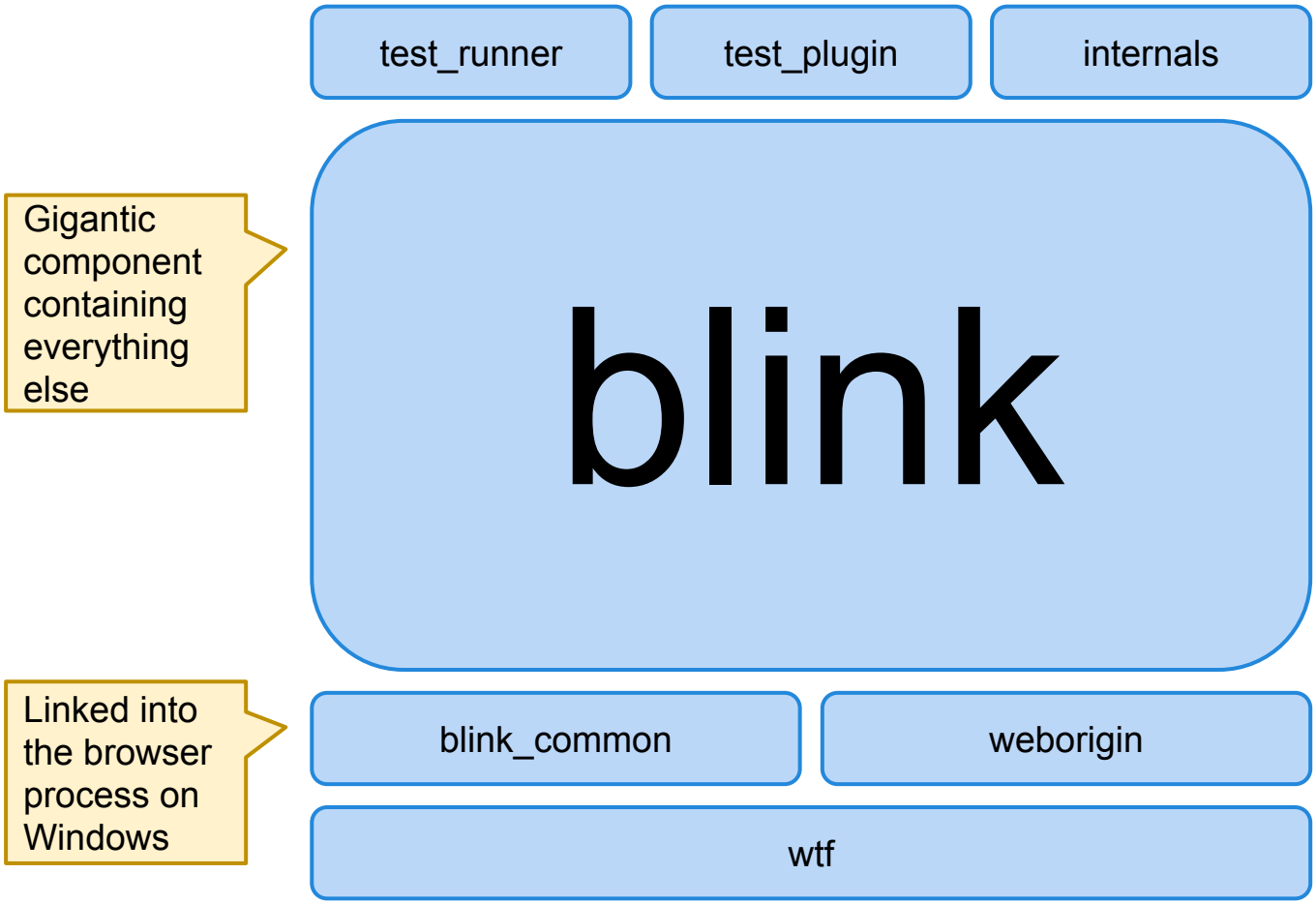
- Move the rest out of Blink
  - OS-specific code (e.g., keycode conversions)
  - Networking (e.g., WebSocket protocol)
  - Rasterization (e.g., impl-side painting)

A Simpler Blink

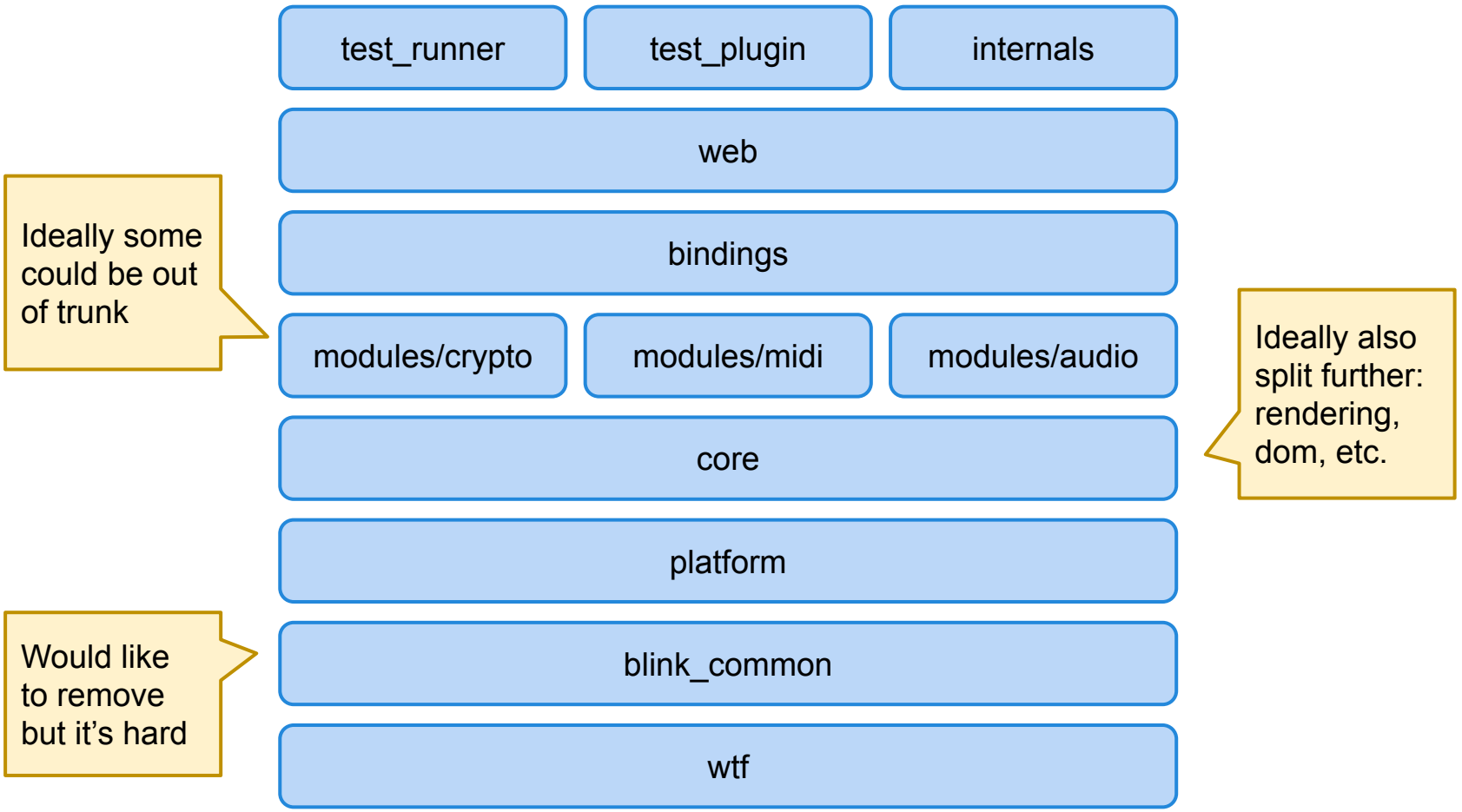


http://www.ohloh.net/p/chromium-blink/analyses/latest/languages_summary

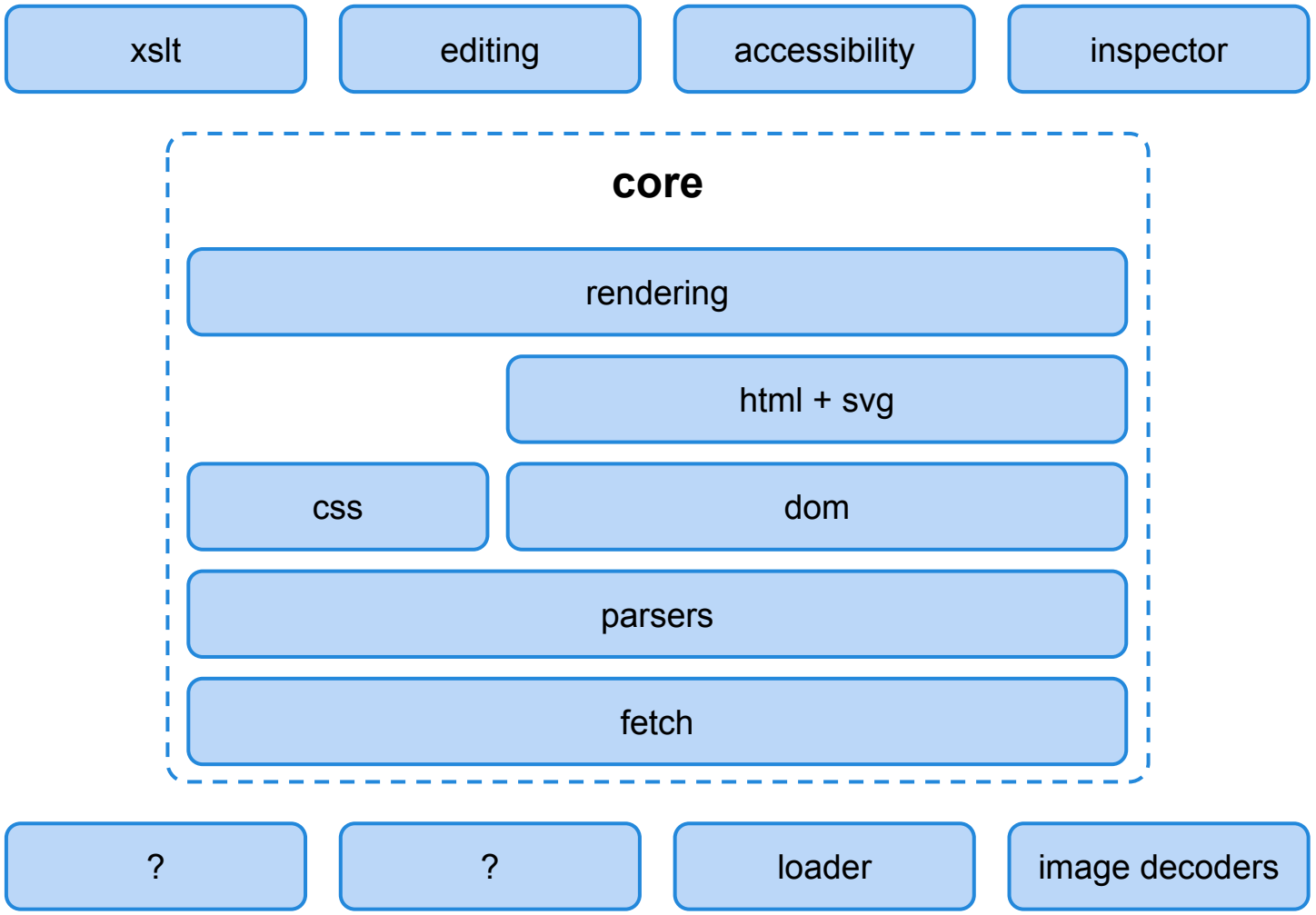
Dependencies (Today)



Dependencies (Soon)



Dependencies (Speculative)



Summary

- Doing fewer things
- Do them faster

Roadmap

Near-term

- Web components performance
  - Optimize for many inline style elements
  - Reduce footprint of Document
- Web animations
- Better compositor integration
- Platform-independent fonts
- Remove vendor prefixes
- API encapsulation (style, rendering, etc.)

Performance

Medium-term

- Layers for SVG
- Remove widget tree
- Out of process iframes
  - Move history to embedder
  - Move CORS to embedder
- Unified C++/JS garbage collector
- Deprecate rarely-used features
  - XSLT? NPAPI? Attr nodes?
- Link modules into a separate dylib

Performance

Speculative

- Moar parallelism
  - Layout, style resolution
- JIT style resolution
- Fast mode
  - Disable slow features (margin collapsing, floats)
  - Optimized code path that goes fast
- Out-of-trunk modules
  - Apps/Extensions bindings in Chromium
- Swappable components
  - Editing and/or XSLT in JavaScript?

Thanks!