web content in GNOME
future visions
using WebKit to enhance the desktop
Are we there yet?

- Applications moving towards hybrid desktop/web technologies
- A lighter and more integrated web toolchain for GTK+ enables
  - Incremental enhancements to existing applications
  - Entirely new approaches to application design
- GNOME 2.24 shaping up as a leader in web/desktop integration
- WebKit gives us the tools to go forward
WebCore content engine

WebKit public API

JavaScriptCore portable C API

applications

C, C++, C#, C++, Vala, Python, Perl...
WebKit language bindings

- **Vala bindings**
  - Maintained by Jürg Billeter

- **Python bindings**
  - Created by the OLPC team
  - Maintained by Jan Alonzo

- **C#/CLR bindings**
  - Also features a JS bridge

- **Perl bindings**
  - Available from CPAN

- gtkmm
Web standards and features

- Full-featured SVG implementation
- HTML5 parser/renderer
- HTML5 canvas
- Fast JS/AJAX
- Strong CSS standards support
- CSS extensions for implicit animations and transformations
- Offline Web application support
Introducing WebView

✍ Write a browser in a dozen lines

```python
import gtk
import webkit

view = webkit.WebView()

sw = gtk.ScrolledWindow()
sw.add(view)

win = gtk.Window(gtk.WINDOW_TOplevel)
win.add(sw)
win.show_all()

view.open("http://planet.gnome.org/")
gtk.main()
```
WebView modes: Scrollable

- Packed in a `GtkScrolledWindow`

  Provides a full-featured browser engine for document display and editing
WebKit modes: Packed

- Packed directly into the UI
  - Acts as an integral part of the surrounding GTK+ UI
  - Place Web content in amongst GTK+ widgets

- Web content / GTK+ size request interop (landing soon)

- Develop and design in parallel
  - Let programmers work on the core
  - Let designers produce UI elements using web skills
When to use a WebView

- When do you use straight GTK+?
- When does Web content enrich the experience?
- Great power; great responsibility

- **WebView** isn't the right tool for every job
  - Continue to use `GtkTextView` for light viewing and editing
  - Use `GtkTreeView` and `GtkIconView` unless you really need a custom look
“The idea was that anybody who used the web would have a space where they could write and so the first browser was an editor, it was a writer as well as a reader.”

Tim Berners-Lee
A writer as well as a reader

Enable WebKit's powerful content editor with one line of code

WebView is designed from the ground up to work as an enhanced GtkTextView
Edit with style

- webkit_web_view_set_editable (WEB_VIEW (view), TRUE);
- Push and retrieve HTML/SVG content with simple accessors or use the upcoming GIO streaming interface
- Perform formatting operations
  - With the basic editing command API
  - Or by manipulating the upcoming GObject DOM directly
- Work on your application's killer features and leave formatting to WebView
Using WebFonts

🔗 SVG/TTF custom fonts are a W3C recommendation

🔗 Apply a distinctive look without compromising usability

🔗 Continue to internationalise with gettext and .po files

🔗 Text selection and editing works as usual

🔗 No installation required

@font-face {
    font-family: 'Bitstream Vera Sans';
    src: url('http://www.freedesktop.org/~alp/tmp/Vera.ttf') format(truetype);
}

h1 {
    font-family: 'Bitstream Vera Sans', sans-serif;
}
Custom fonts in action

SVG fonts
Easy to design with tools like Inkscape

WebFonts
TrueType fonts on the Web
GObject DOM access

- Upcoming GObject DOM features
  - Complete access to the DOM (all levels)
  - Stable API
  - Access to underlying platform objects (eg. GStreamer pipeline, Cairo context)
  - Available in bindings
  - Develop complete web applications in your favourite language, not just JavaScript
COM DOM access on Linux

- DOM access with COM and WebKit/GTK+ on any platform
- Useful stepping-stone for applications migrating from XPCOM/Gecko to WebKit
- GObject DOM is preferred to COM for new applications

Mono 2.0
.NET-compatible
WebControl using WebKit
API: Apply settings with ease

- Use **WebSettings** to group settings for multiple WebViews
- Keep **granular settings** per WebView or enforce global settings when necessary
API: Going asynchronous

- Use decision objects to delegate actions requiring user input or network queries
  - Authentication challenges
  - Navigation requests
  - Script alert and print dialogs

- A dream come true for browser developers
- Allows programmers to **eliminate modality**
- Lets users get on with what they're doing
HTML5 video with GStreamer

Open Source
Web video without proprietary plugins

Versatile
Create stylish DVD/DVB players, video conferencing tools
Perfectly native widget styling
WebKit for browser engineers

- Engine core written in a sensible dialect of C++
- Approachable to C hackers
- Follows a coherent coding style
- Project-wide refactoring and reorganisation is encouraged
- Internal APIs are “informally” abstracted and change frequently while the public API is strictly stable (similar policy to the Linux kernel)
Browse with Epiphany

♀ Epiphany, the GNOME web browser

♀ Originally a GTK+ UI around the Gecko rendering engine

♀ Experimental WebKit support added by Xan Lopez at GUADEC 2007

♀ Boosts performance, standards support and ease of development

♀ Currently stabilising for 2.24
A boost in web standards

Epiphany using WebKit/GTK+
First Open Source browser to pass ACID3 on Linux
Theming notifications the easy way

- notification-daemon
- Writing themes was hard work (C, Cairo) for artists
- WebKit-based themes now possible using HTML and CSS
- Accessible by default
- Compatible w. Growl
Styling IM/chat clients

- Use GTK+ to provide the core UI and shell
- Use **WebView** to emphasise content
- Allow users to customise their applications
- Create a community around your application

Pidgin IM client
WebKit support by Sean Egan
Hosting widgets

GtkWidget *web_view = webkit_web_view_new ();
webkit_web_view_set_transparent (
    WEBKIT_WEB_VIEW (web_view), TRUE);

**HTML5 canvas**
for drawing

**Shell access**
for local operations

**D-Bus IPC**
coming soon

**HTML5 local storage**
for data persistence
Invent a new look and feel

**Composited desktop**

**SVG circles**

**Experiment with fun new UI concepts**
Cross-platform

Supported platforms

- GNU/Linux (X11, DirectFB)
  - All major distributions (Debian and Gentoo packaging teams in particular have helped a lot upstream)
- FreeBSD, DragonFlyBSD (X11)
- OS X (Native, X11, DirectFB)
  - Imendio and Nuanti working to complete the native GTK+ backend
- Windows (Native, X11)
  - Maintained by Nuanti, final patches hitting SVN now
JavaScriptCore C API

- Integrates with the GObject API
  - Stable
  - Fully documented
- Portable
  - Installed with WebKit/GTK+
  - Ships with OS X
- Standalone use
  - Provides a light scripting engine for any application
- Fastest mainstream JavaScript implementation, SquirrelFish

Use it to
- Export functions to JavaScript
- Invoke JavaScript
- Build dynamic language bindings

<table>
<thead>
<tr>
<th>Portable application code</th>
<th>OS X</th>
<th>Linux</th>
<th>Windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebKit</td>
<td>n/a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WebKit/GTK+</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SquirrelFish VM

A new register-based, direct-threaded, high-level bytecode engine, with a sliding register window calling convention. Lazily generates bytecodes from a syntax tree, using a simple one-pass compiler with built-in copy propagation.

[SunSpider Relative Running Times graph]

smaller is better

Independent benchmark result, June 3 2008
http://www.satine.org/archives/2008/06/03/squirreelfish-is-faster-than-tamarin/
Evolution

- Evolution, the GNOME email/collaboration client
- Partial WebKit support beginning 2.24 to add enhanced viewing capabilities
- Advanced WebKit-based editing capabilities targeted for 2.26

LIVE DEMO
WebKit and Yelp

- Yelp, GNOME documentation browser

- Maintainer Don Scorgie says
  - “Blazing fast. Startup goes from 2.8s to 1.9s.”
  - “API rocks. It's like a real gtk+ API. I can understand what's going on in it.”

- WebKit support now in SVN, stabilisation in progress
WebKit and Devhelp

- A browser for Gtk+/GNOME API documentation
- Switch from GtkMozEmbed to WebKit reduced code size by ~1000 LoC
- Faster startup
- Lower memory footprint
- New features
  - Printing support
  - Inline search highlighting
WebKit in GIMP

- Web content used for inline help
- Adds advanced HTML rendering capabilities
- Adds network support for online capabilities
- Documentation as an integral part of the application
Location-aware web apps

- External module to expose location metadata to Web apps
- Implements the locationaware.org spec
- Spec not final
- Uses Gypsy GPS library (alternative location sources possible)
  - http://folks.o-hand.com/iain/gypsy/

```javascript
var geolocator = navigator.getGeolocator();
geolocator.request(function(location) {
  alert(location.latitude+' , '+location.longitude);
});
```
OpenMoko

- OpenMoko WebKit applications
  - Browser
  - Feed reader
- Maintained by Holger Freyther (also a WebKit developer)
- OpenMoko supports the WebKit project by providing a build server for continuous integration
OLPC Sugar activity

Fast
Starts up to 5 seconds faster
renders fast

Light
Small memory footprint
~10M less resident memory use
WebKit e-paper with the Irex iLiad

Midori browser on iLiad
WebKit port by Adam Boeglin

Open questions
How do we reduce repaints?
Can we save memory for grayscale displays?
Poky Linux

Default browser for the Poky mobile Linux distribution
WebKit for Maemo


Cairo is a 2D graphics library with support for multiple output devices. Currently supported output targets include the X Window System, Win32, image buffers, PostScript, PDF, and SVG file output. Experimental backends include OpenGL (through glib), Quartz, and XCB.

Cairo is designed to produce consistent output on all output media while taking advantage of display hardware acceleration when available (e.g. through the X Render Extension).

The cairo API provides operations similar to the drawing operators of PostScript and

http://www.cairographics.org
Media center devices

- Set top boxes often use DirectFB
- Aim to provide a media-oriented user experience
- Integrated multimedia and web capabilities

Extend traditional user interfaces

- Web browser
- Traditional UI

- Integrated UI elements
- Web-based UI

... or switch to a Web-based main view
Browser plugin support

Compatible with proprietary plugins like Flash

Works with open plugins like totem, swfdec, moonlight
Putting the Web in GTK+

- **Enhance** the GTK+ core to meet browser needs
- **Extend** the GTK+ toolchain with Web capabilities

The vision...
Aligning on technologies

- **GIO** stream loader API
- **libsoup** HTTP backend
- **GStreamer** video
- **Cairo** for graphics
- **Pango** for text
- **GLib** primitives and object model
- **GLib** for Unicode (WIP)
- **GTK+** backend/frontend and API
WebKit/GTK+ now features advanced accessibility capabilities, developed by Nuanti and Apple.

Uses ATK to expose core and optional AT-SPI interfaces.

For use by Orca, screen readers, Braille readers, specialised input devices.

Includes support for Accessible Rich Internet Applications (ARIA).
The web in 3D

LIVE DEMO
Future

GTK+ CSS theme engine
Release cycle

- WebKit/GTK+ schedule aligned with GNOME from 2.24 onwards
- Predictable development and stabilisation cycle
- Working directly with application developers
- API follows platform stability guidelines

WebKit 1.0.1

May 2008

 GNOME 2.24 (freeze)

September 2008
“The next big thing is the one that makes the last big thing usable.”

Blake Ross
Are we really nearly there yet?

- Applications moving towards hybrid desktop/web technologies
- A lighter and more integrated web toolchain for GTK+ enables
  - **Incremental enhancements** to existing applications
  - **Entirely new approaches** to application design
- GNOME 2.24 shaping up as a leader in web/desktop integration
- Let's make it happen
Get involved

http://live.gnome.org/WebKit

http://www.webkit.org

IRC: #webkit, #webkit-gtk / FreeNode
IRC: #epiphany / GIMPNet