Proceedings of the 7th USENIX Tcl/Tk Conference

Austin, Texas, USA, February 14–18, 2000

TKGECKO: A FRILL-NECKED LIZARD

Steve Ball

TkGecko: A Frill-Necked Lizard

Steve Ball
Zveno Pty Ltd
Eolas Technologies, Inc
Steve.Ball@zveno.com

Abstract

The Mozilla Open Source project has made a full-featured, fast Web browser, Netscape's Navigator, available in source form to the Internet community. A major component of the project, and an early released package, is the NewLayout (a.k.a. Gecko) HTML/XML rendering engine. One feature of this module is that it is designed to be embeddable in any application. This characteristic is quite compatible with Tcl/Tk.
TkGecko is a project to create a Tk extension that allows the NewLayout rendering engine to be embedded in a Tk application as a Tk widget. Ideally, this new widget will be as completely configurable as all other Tk widgets, with all configuration options able to be queried and changed at run-time. The widget features methods and configuration options that allow the functions of the NewLayout rendering engine to be accessed and changed from a Tcl script.
Currently the TkGecko project is still in its infancy. This paper aims to define the goals of the project and expected milestones.

Keywords

Netscape Navigator, Mozilla, NewLayout, NGLayout, Gecko, Tcl, Tk, TEA.

1. Introduction

In 1997 Netscape Communications Corporation [1] released the source code for their Netscape Navigator Web browser to the Internet community, under the Netscape/Mozilla Public License (NPL or MPL) [2], a license that allows free use of the source, redistribution and modification. Netscape has setup an independent organisation to develop the next release of Netscape Navigator under the Open Source banner called Mozilla.org [3]. Currently, the majority of developers working on Mozilla are employed by Netscape (approximately 120 engineers), but there are some employed by the other organisations (about 25 engineers).

The first major "product" release from Mozilla.org is Gecko, the core HTML/XML page rendering engine for the browser. Gecko is known more formerly as the NewLayout module. This module is written in C++ (in fact, all of Mozilla is written in C++, but there does exist a separate project to reimplement Mozilla in Java), and is designed to be embeddable inside other applications, for example as a HTML-based help viewing system. There are a few examples of Gecko being embedded in applications, such as an ActiveX component [4] (which entirely replaces the Internet Explorer control!) and the DocZilla viewer [5].

Eolas Technologies, Inc. [6] are sponsoring work to create a Tk extension which embeds Gecko as a Tk widget. The working title for this project is "TkGecko". The extension defines a new Tk widget class called "newlayout". The availability of this extension will provide a high-quality viewer of HTML and XML documents to be used by Tk applications, with full support for all of the current Web standards, such as CSS, JavaScript, DOM and XSL. While most of the major Web standards will be supported, it is not clear at this stage whether the extension will be able to handle Java applets or browser plugins. The former will require a Java Runtime Engine (JRE) to be included in the extension.

Although TkGecko will allow the creation of a general-purpose Web browser, that is not the primary motivation for this project since at least two general-purpose Web browsers already exist. Instead, the aim is to support application developers wishing to incorporate a HTML/XML viewer into their products, often for the purpose of displaying help documents. It is certainly now the case that Web browsers are now so ubiquitous that an application developer can assume their availability and simply launch a browser to display a document. However, many developers wish to have a tighter integration of the display of help documents, particularly for context-sensitive help, with their application. An embedded viewer is necessary to satisfy this requirement.

This approach to displaying documents has been already been anticipated by Microsoft, who have made their Internet Explorer browser available not just as a stand-alone browser but also as an ActiveX control [13]. Hence applications running on a Microsoft Windows platform can easily embed the Internet Explorer ActiveX control to realise an embedded HTML/XML document viewer. Alternatively, there is an equivalent ActiveX control for Mozilla [4]. However, this is not a cross-platform solution and therefore the need to have an easy method of embedding the Gecko rendering engine.

1.1. Distribution

In the first phase of the project, the basic embedding of the NewLayout widget as a Tk widget will be achieved. This phase includes writing all of the necessary configuration files to link the numerous required libraries into a dynamically loadable shared library, for the Linux (i386) and Microsoft Windows 95/98/NT/2000 platforms. The new Tk widget class is called `newlayout`.

Subsequent phases of the project will implement configuration options and methods for the newlayout widget, as detailed in the section "Future Directions". It is also an aim of the project to extend the package to the MacOS and LinuxPPC platforms and architectures.

Packaging TkGecko for distribution to interested parties presents some problems. The source code alone for Mozilla is approximately 19MB, but to actually build the browser requires almost 1GB of disk space. However, these requirements may be reduced by customising the build process to compile only what is needed for the Gecko engine. Customising the Mozilla build process has not yet been attempted. Alternatively, the binaries of the Gecko engine are only approximately 1.4MB in size so binary distributions for the platforms specified above are the currently favoured form for making the extension available.

1.2 Terminology

There are numerous names and terms surrounding the Mozilla project and its components. The Gecko HTML/XML rendering engine is more correctly known as the NewLayout or NGLayout module. Netscape Communications Corp. marketing droids originally dubbed it "Gecko", but the developers themselves don't particularly use that term.

2. Mozilla Architecture

In order to design and implement a Tk extension to embed Mozilla, it is vitally important to understand the architecture of the Mozilla browser. While each of the technologies required to construct a Web browser are relatively simple, combining them together into a sophisticated Web browser makes Mozilla a complex piece of software engineering.

Of particular interest in the Mozilla architecture is the NGLayout module. The overall goal of the module is to provide high-quality rendering of HTML and XML documents, while at the same time achieving high-performance in page display. Documents themselves have a complex structure, especially when HTML frames (subdocuments) and tables are taken into account. This module has been designed to allow for the complexities of page display, as well as supporting the JavaScript page scripting language and dynamically loaded content viewers, for rendering different content types such as image formats. It is beyond the scope of this paper to undertake a detailed study of the module, nor to explain how high performance is achieved.

Following is a brief overview of the modules which are of the most importantance to the NewLayout extension. Figure 1 shows their relationship.

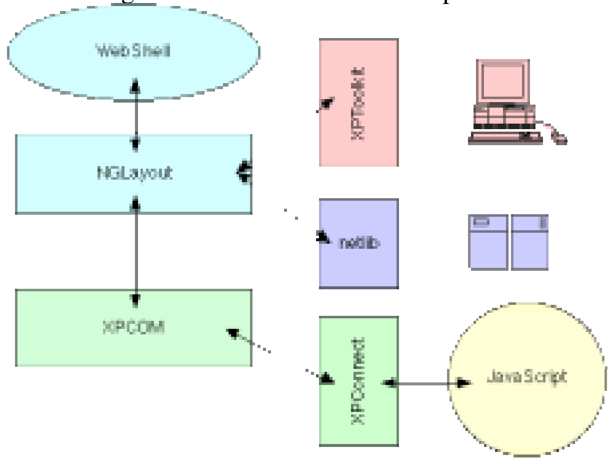


Figure 1: Mozilla Architecture

Underpinning the entire Mozilla architecture is the XPCOM (Cross-Platform Component Object Module) module. XPCOM is the same in concept to Microsoft's COM, but is much simplified and implemented across all operating system platforms. In fact, XPCOM aims to be binary-compatible with COM. One feature of XPCOM is that because it binds component interfaces together at runtime it allows parts of Mozilla to be used separately and to replace an implementation of a component with another seamlessly. This characteristic is very useful for building and shipping the TkGecko extension.

Another module of interest is XPConnect, which provides a bridge between XPCOM and JavaScript. This module allows JavaScript scripts to access and manipulate XPCOM objects, as well as allowing a JavaScript program to provide the implementation for a XPCOM object.

In order to be able to write portable User Interface code, the Mozilla project is developing the XPToolkit (Cross-Platform Toolkit) module. This module uses XUL, the XML User Interface Language, to describe user interfaces. As the name suggests, a XUL document provides the description of the desired widgets and their layouts as a XML document. XPToolkit parses the XML document and then assembles and displays the appropriate native widgets for the interface. The implication of this approach is that building a user interface becomes the simple task of writing a document.

At the lower level of windowing and drawing primitives, the Cross-Platform Front-End (XPFE) module provides a portable interface to the underlying windowing system of the machine platform. On Windows and Macintosh this is obviously the operating system's native windowing facilities, but on Unix/X Window the situation is more complicated due to the plethora of available toolkits. A number of toolkits are supported, but the primary toolkit used by the Mozilla browser is GTk+. However, the embedding interface also supports the direct use of Xlib. An initial approach to embedding the NewLayout in Tk used the GTk+ embedding interface, but since Tk also uses Xlib directly, the TkGecko extension now instructs Mozilla to use the Xlib toolkit. This is discussed in more detail below. It was recently announced that the WebShell interface is to be redesigned, and that the alternative toolkits may be dropped in favour of concentrating on the GTk+ toolkit. USENET newsgroup discussion of the new design also supports retaining the Xlib interface. As a result the TkGecko project will need to reassess its approach to embedding the NewLayout module once the module redesign is underway.

Networking functions and protocol support are provided by the netlib module. There is also a new project to improve netlib called "Necko". These modules handle the transfer of document data, and use threads to deal with latencies. As of Milestone 9 Necko is now used with Mozilla. Integrating the netlib/Necko event model with the Tcl Notifier may be an issue for the TkGecko project, but at this early stage has proven to be simple and straight-forward.

2.1 NewLayout Module

On top of all of these modules (and some others not mentioned for the sake of brevity) is the NewLayout module, the subject of the TkGecko project. The aim of this module is to provide a small, fast rendering engine for Web documents.

The most important layer of the NewLayout module is the WebShell interface. This module also provides the embedding function of the NewLayout module. A WebShell is used to display each subdocument (frame) in a document. WebShells may be nested, and the initial WebShell is known as the root WebShell. Since WebShells may contain other WebShells, the nsIWebShell class is subclassed from the nsIWebShellContainer class. The nsIWebShellContainer class provides additional methods that provide notification of events in the contained WebShell(s), such as the start of loading documents, the end of a document load, and so on.

At times the NewLayout module may need to query the parent window of the root WebShell for a document. This is done using the `nsIBrowserWindow` interface. For example, NewLayout may wish to set the title of the window, or update a progress bar. To support these functions, TkGecko will also need to implement an interface to the `nsIBrowserWindow` class. This is not yet currently done.

The WebShell interface provides a number of methods for manipulating the (sub-)document. A few of these methods include:

`LoadURL`

Load a document, given a URL

`Stop`

Terminate loading the document

`Back`

Load the previous document on the history list

`Forward`

Load the next document on the history list

`GetChildCount`

Return the number of WebShells which are children of the current WebShell

`GetParent`

Return the parent WebShell of the current WebShell

`AddChild`

Add a new WebShell

3. `newlayout` Tk Widget

The newlayout Tk widget is the main feature of the TkGecko extension. Using this widget, a Tk application developer is able to easily embed a Mozilla WebShell widget in a Tk user interface.

3.1 TEA Compliance

A major objective of the TkGecko extension is to make embedding the Mozilla NewLayout module as easy as possible for Tk application developers. The first step to achieving this goal is to make the package TEA (Tcl Extension Architecture) compliant [7]. TEA compliant packages are able to be dynamically loaded into any version of a Tcl interpreter in a forward-compatible fashion without re-linking. Given that compiling Mozilla is a non-trivial undertaking this feature is very useful.
TEA compliance proved to be very easy to do. Modifying the autoconf and automake configuration scripts from the TEA sample extension provided by Scriptics was straight-forward. Mozilla also makes heavy use of autoconf and GNU Make, but in a fashion not entirely compatible with the manner in which TEA does. For example, Mozilla uses Makefiles in a hierarchical configuration, whereas TEA uses automake. Integrating the automated Mozilla configuration and build system with TEA's proved to be a difficult problem, which has not yet been completed to a satisfactory conclusion.

3.2 `newlayout` Widget Architecture

TkGecko has two modules, the main module, written in C, is `newlayout.c` which provides the interface with the Tcl interpreter and Tk. Since Mozilla is written in C++ it is necessary for the TkGecko extension to include another module to provide the C++ interface. This is the `tkmozilla.cpp` module. The TkGecko extension architecture is shown in figure 2.

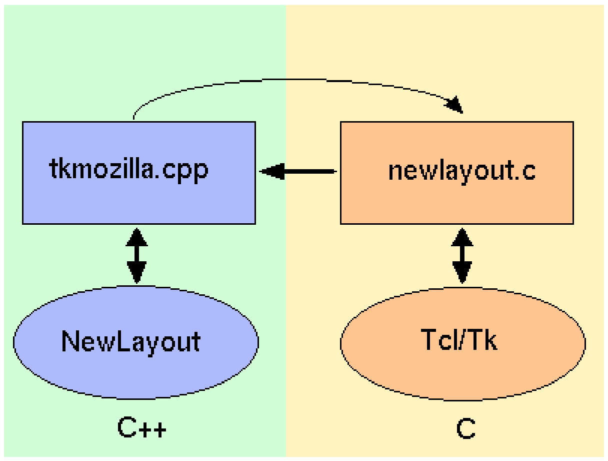


Figure 2: TkGecko Module Structure

The `newlayout.c` module takes care of all of the Tcl/Tk housekeeping. This includes package and stub initialisation, widget class creation command, widget command, widget method implementations, configuration option processing, Tk event handlers and so on. When interaction with the Mozilla NewLayout module is required, a wrapper function in the `tkmozilla.cpp` module is called. The module also includes a number of "call-in" procedures which the `tkmozilla.cpp` may invoke in response to certain events occurring. These procedures check whether a callback has been defined for the event and evaluate the callback if present, otherwise the procedure takes no action.
The `tkmozilla.cpp` module provides an interface to the Mozilla NewLayout module. Each procedure in this module provides an interface that is callable from C code, ie. the `newlayout.c` module. This approach is used to expose the functionality of the Mozilla NewLayout module to the Tcl/Tk interpreter. In addition, this module will "register" procedures to be called to handle certain events. In some cases these are registered explicitly as callbacks and in other cases they become associated with events by subclassing certain NewLayout classes. In either case the events are handled by explicitly invoking a call-in procedure in the `newlayout.c` module.

3.3 Embedding The WebShell

Embedding a WebShell requires only a few steps to be taken [12]:

1. Register the NGLayout libraries
2. Create an event queue
3. Initialize the network library
4. Create a WebShell

The first three functions are performed by the `newlayout.c` module upon loading of the package by

calling `tk_mozilla_init`, a procedure provided by the `tkmozilla.cpp` module. On Unix Necko functions using a `select` style interface. The file descriptor used by Necko is retrieved and added to the Tcl Notifier as an event source, with a callback to the `newlayout.c` module.
The actual creation of a WebShell widget is performed each time a newlayout widget is instantiated. Tk widgets may have several instantiations, so the TkGecko extension must support the creation of multiple WebShell widgets. The WebShell creation interface requires the Xlib window identifier, so the actual creation of the WebShell widget must be delayed until the newlayout Tk widget is first mapped to the screen. This has the unfortunate effect of causing a time lag between the appearance of the widget on screen and in the initialisation of the WebShell widget. This effect seems to be further complicated by the threading involved.

4. Using TkGecko

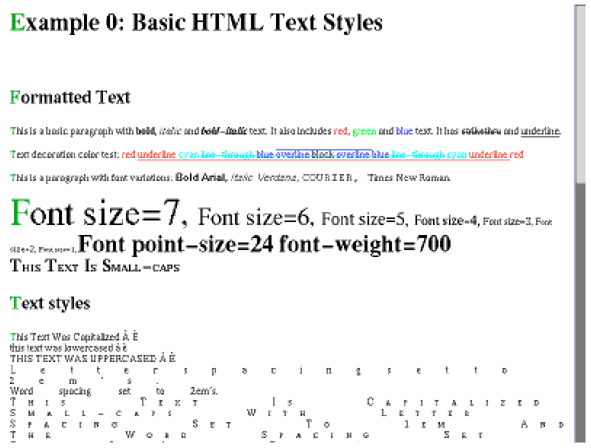
Using TkGecko in a Tk application is now just as simple as any other Tk widget. Consider the following script examples.

```
package require newlayout

newlayout .mozilla
grid .mozilla

.mozilla configure -url
file:///home/steve/samples/test0.html
```

Figure 3 shows the user interface generated by this script. Note carefully the plain scrollbar present on the right-hand side. This scrollbar is added by the NewLayout module and currently TkGecko is not able to disable it or control the scrolling.



Example 0: Basic HTML Text Styles

Formatted Text

This is a basic paragraph with **bold**, *italic* and ***bold–italic*** text. It also includes red, green and blue text. It has ~~strikethru~~ and underline.

Text decoration color test: red underline cyan ~~line–through~~ blue overline black overline blue ~~line–through~~ cyan underline red

This is a paragraph with font variations: **Bold Arial**, *Italic Verdana*, COURIER, Times New Roman.

Font size=7, Font size=6, Font size=5, Font size=4, Font size=3, Font size=2, Font size=1, **Font point–size=24 font–weight=700**

THIS TEXT IS SMALL–CAPS

Text styles

This Text Was Capitalized À È
this text was lowercased à è
THIS TEXT WAS UPPERCASED À È
L e t t e r s p a c i n g s e t t o 2 e m ' s .
Word spacing set to 2em's.
T H I S T E X T I S C A P I T A L I Z E D S M A L L – C A P S W I T H L E T T E R S P A C I N G S E T T O 1 E M A N D T H E W O R D S P A C I N G S E T

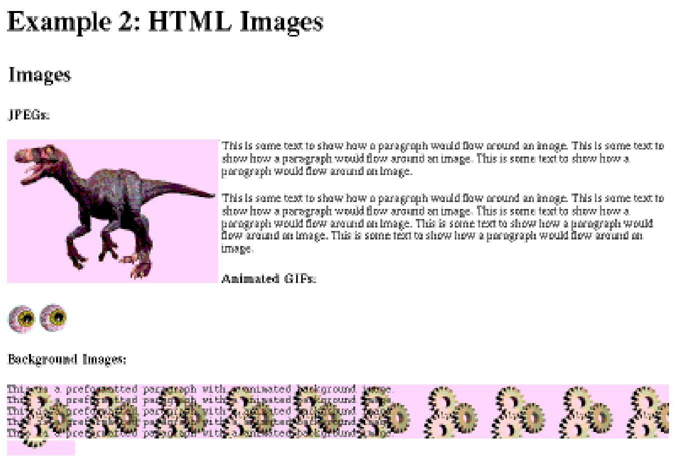
Figure 3: Display Of newlayout Widget

```
package require newlayout

newlayout .mozilla
grid .mozilla

.mozilla configure -url
file:///home/steve/samples/test2.html
```

Figure 4 shows the user interface generated by this script. Due to a malfunction in the NewLayout timer library the animated GIF images do not function correctly (anyone who has seen the bloodshot eyes in action would consider this to be an improvement).



Example 2: HTML Images

Images

JPEGs:

This is some text to show how a paragraph would flow around an image. This is some text to show how a paragraph would flow around an image. This is some text to show how a paragraph would flow around an image.

This is some text to show how a paragraph would flow around an image. This is some text to show how a paragraph would flow around an image. This is some text to show how a paragraph would flow around an image. This is some text to show how a paragraph would flow around an image. This is some text to show how a paragraph would flow around an image.

Animated GIFs:

Background Images:

```
This is a preformatted paragraph with animated background image.
This is a preformatted paragraph with animated background image.
This is a preformatted paragraph with animated background image.
This is a preformatted paragraph with animated background image.
This is a preformatted paragraph with animated background image.
```

Figure 4: Display Of newlayout Widget

4.1 Current Status

At this early stage of the project, loading the extension and displaying a document is the only feature of the TkGecko extension that works. Loading new URLs also works.
The Tk and Mozilla event systems have not been hooked up to the stage where events can flow freely, thus hyperlinks do not work, nor does widget redraw or resize. Instantiating multiple newlayout widgets crashes the application. These bugs and missing features are the highest priority for the project, in order to achieve basic functionality of the widget.

5. Development Plan

Further work is scheduled for TkGecko, with the aim of making the extension generally usable and stable by the end of 1999. The development schedule includes the following milestones:

- Fix crashing bugs and malfunctioning timer libraries.
- Add event processing.
- Fix configuration options.

- Add callbacks for WebShell widget events: BeginLoadURL, EndLoadURL, Progress, and so on.
- Add methods to query and manipulate document content and structure.
- Add an event binding mechanism for document content similar to tags in Tk Canvas and Text widgets.

6. Future Directions

Beyond and apart from the planned development of this project, there are some further areas of research to investigate. Some of these are detailed below. The project will also track developments in Mozilla itself.
As mentioned above, the WebShell interface is to be redesigned and this should lead to more functionality of the widget being exposed through external interfaces. It should be the case that every aspect of the widget's function can be monitored and overridden by the hosting application. In the case of the TkGecko extension, all of these functions would be controlled by Tcl scripts via callbacks.
Firstly, as mentioned above, another interesting project will be to implement a Tcl version of the XPConnect module. The goal of this work would be to allow Tcl to be used within Mozilla in an equivalent way to JavaScript.
Secondly, XUL may have some significant ramifications for Tk. It should be possible to implement XUL tools as Tk applications, such as an interface designer. Another possibly useful tools would be to dump a Tk interface as a XUL document, and be able to recreate the interface from the document in Tk. This would have the advantage to the designer of being able to interactively prototype the interface using Tk first.

6. Related Work

Gecko supports the display of XML documents as well as HTML documents, and so has a number of parsers and modules to support this function. These include James Clark's expat XML parser, a Tcl interface to which has already been created [8].
Scripting XML documents within the browser is achieved using the DOM [9], and Mozilla has an implementation of the DOM in C++ for use by JavaScript. A possible future project may be to create a separate Tcl extension to provide an interface to this DOM implementation. The extension would implement the TclDOM API [10], and so maintain compatibility with existing TclDOM scripts. This would be an alternative to tDOM [11]. The content of a WebShell widget is itself a DOM object, so such an interface may be used to manipulate the content of a widget using a Tcl script.

7. Conclusion

Netscape Communications Corporation have provided an opportunity to create a new Tk widget by releasing the source code of Netscape Navigator under an Open Source license. This Open Source project is commonly known as Mozilla.
The TkGecko project has created a Tk extension which creates a new Tk widget, newlayout, to embed the core rendering engine of the Mozilla browser, NGLayout, in a Tk window.
The newlayout widget has minimal functionality at present, but will eventually expose all of the interfaces of the Mozilla NewLayout module to Tk application scripts. Further, the project may also make Mozilla itself scriptable using Tcl/Tk in addition to JavaScript.

8. References

[1] Netscape Communications Corporation. http://www.netscape.com/

[2] Netscape Public License, Mozilla Public License. http://www.mozilla.org/NPL/

[3] Mozilla.org. http://www.mozilla.org/

[4] Mozilla ActiveX Component. Adam Lock. http://www.iol.ie/~locka/mozilla/mozilla.htm

[5] DocZilla Viewer. CiTEC. http://www.doczilla.com/

[6] Eolas Technologies, Inc. http://www.eolas.com/

[7] Tcl Extension Architecture. http://www.scriptics.com/products/tcltk/tea/

[8] *XML Support For Tcl*. S. Ball. Proceedings of the Sixth Annual Tcl/Tk Conference. September 14-18 1998, San Diego CA USA. http://www.zveno.com/zm.cgi/in-tclxml/

[9] Document Object Model. L. Wood, et al.
World Wide Web Consortium Recommendation.
http://www.w3.org/DOM/

[10] TclDOM. S. Ball.
http://www.zveno.com/zm.cgi/in-tclxml/in-tcldom/

[11] *tDOM - a XML/DOM/XPath implementation for Tcl*. J. Loewer.
http://sdf.lonestar.org/~loewerj/tdom.cgi

[12] *Extending Mozilla Or How To Do The Impossible*. J. Stenback, H. Toivonen.
http://www.doczilla.com/development/extmoz.html

[13] *WebBrowser Control*. Microsoft Corp.
http://msdn.microsoft.com/workshop/browser/webbrowser/wbentry.asp