In Search of the Perfect Mega-widget

Stephen Uhler
Sun Microsystems Laboratories
What is a Mega-widget

- Tk widget written in Tcl
  - Can be entirely in Tcl or a combination of Tcl and C
- Behaves *exactly* like a built-in widget
  - Script writer doesn’t need to know whether a widget is *mega* or *built-in widget*
Sample Uses for Mega-widgets

- Combinations of existing widgets
  - Combo-boxes

- Changing the “look” of existing widgets
  - Example: new `scale` widget

- Adding behaviors to an existing widget
  - `text` ➔ HTML message widget

- Removing functionality from a widget
  - Enforce a standard look & feel
Space-saving Scale Widget

- Stock scale widget

- Space-Saving scale widget

- Sample Space-saving interface
A Mega-widget Solution

- **What it is:**
  - A small change to the “core” of Tk to provide the mechanism for a complete mega-widget capability in Tcl

- **What it isn’t:**
  - A namespace facility
  - A framework for constructing mega-widgets
  - A set of mega-widgets
Properties of Built-in Widgets

- Command interface
- Behavior
- Bindings
Properties: Command Interface

- `widget .foo ?-option -value...?` ➞ `.foo`
- `.foo configure ?-option -value...?`
- `.foo cget -option`
- `.foo sub-command ?args?`
Properties: Behavior

- `winfo command .foo`
  - exists, class, width, height ...

- Option database control of widget parameters
  - `option add` ...

- `focus`
  - `focus ➔ .foo`
Properties: Bindings

```tcl
bind .foo <Event> action
  %-X substitutions in action
  %-W => .foo ...
bindtags .foo {.foo Foo . all}
```
Summary of Changes

- Frame -command option
- Bind event propagation and translation
- Focus management
- Miscellaneous
  - winfo container
  - C API’s to access mega-widgets
Frame -command

- Turns a frame into a container.
- Provides a hook for Tcl code to process the widget command.
- Allows access to actual frame

```tcl
frame .mega -class Mega \
   -command {doMega .mega}

proc doMega {name args} {...}
```
Event Propagation and Translation

- Some events are propagated to the enclosing container.
  - A new event is synthesized
  - %X substitutions are translated as required
  - The event is re-issued to the container

- Propagation and translation is repeated for each enclosing container
Tk Event/Bind Model
Mega-widget Event/Bind Model
Widget Binding Example

```tcl
bind .mega <1> {puts "got: %W %x %y"}

- **Real** widget: got: .mega 157 37
- "Fake" mega-widget got: .mega.inside 15 31
```
Focus Changes

\textbf{Focus}
- returns outermost container that has the focus

\textbf{Focus -container container\_name}
- Returns actual focus widget (or container)
Widget Focus Example

- focus
  - **Real** widget: .mega
  - "Fake" mega-widget .mega.inside
Implementation

- New flags added to Tk window structure
  - IS_CONTAINER, IS_CONTAINED_IN
- Recursive event dispatch
- Event substitution rewriting
- New options/changes added
  - winfo, focus
- C API's to set/get IS_CONTAINER flag
Conclusions

- Most Mega-widget capabilities are already in Tk
- Only small changes to event and focus handling are needed
- Additional features added for convenience
  - Easy access to Tcl code
  - Mega-widget introspection