

eSWT: The New UI Toolkit for Mobile Java

Gorkem Ercan
Nokia

Current Mobile Java UIs

- 2D & 3D Graphics support
- MIDP Limited Capability Device User Interface API (LCDUI)
 - originally designed at the time there were still black and white monochrome displays (Nokia Series 30)
 - No significant improvements during the past years in JCP
 - portability to low-end devices
- Mobile Java lacks the UI functionality of modern smart phone user interface frameworks!

Why do we have eSWT anyway?

- **Open**
 - Open source
 - Licensed under Eclipse Public License (EPL)
 - Part of eRCP project in eclipse.org
- **Capable**
 - Rich user interface component set
 - Flexible and scalable layout system via layout managers
 - Rich user interface events
- **Integrated**
 - Access to native UI functionality on-par with smartphone UI frameworks
 - Java applications look (and behave) like native applications
 - Enables advanced entertainment and enterprise applications
- **Familiar**
 - Subset of SWT
 - Provides familiar java UI concepts
 - Already available on many platforms

Googled eSWT?

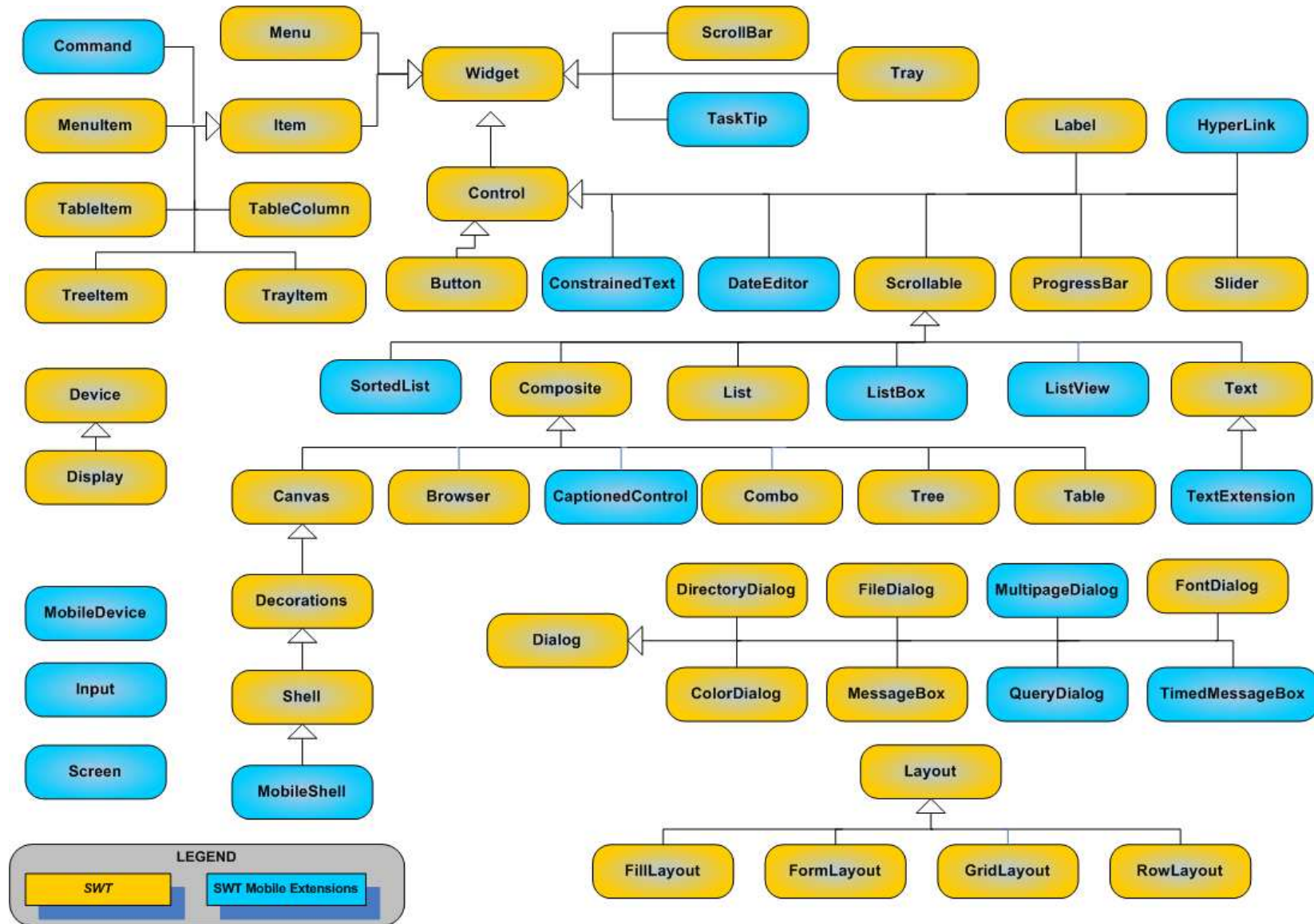
- **Extracorporeal Shock Wave Therapy**

Extracorporeal Shock Wave Therapy (ESWT) devices generate pulses of high-pressure sound that travel through the skin. For reasons that are not fully understood, soft tissue and bone that are subjected to these pulses of high-pressure energy heal back stronger.

Then... What is eSWT ?

- Provides efficient, portable access to the user interface facilities of the operating system
 - Most of the "real" work is done by the optimized, platform-specific code
- Consists of 3 packages
 - Subset of desktop SWT
 - *core package*
 - *expanded package*
 - New mobile specific eSWT components
 - *mobile package*
- Has traditional Java GUI library characteristics
 - UI is constructed by widgets in containers
 - Containers use layout managers to scale the UI.
 - API is non-thread safe and there's single UI thread
 - Applications must explicitly include the event dispatching loop in the *entry*

Class Diagram



eSWT fundamentals

- Widgets are created by the programmer and disposed by the programmer
- Widgets are created using the constructors
 - Operating system resources are allocated during construction
 - No factories
 - No convenience constructors
- A widget can not exist without parent
 - Parent is usually the first parameter in constructor
 - Parent can not be changed (unless it can)
- Style parameter
 - integer bit values used to configure the behavior and appearance of widgets.
 - Can not be changed after creation
- dispose() method must be explicitly called
 - Hides the widget and its children,
 - Releases all associated operating system resources and facilitates garbage collection

```
Text text = new Text (parent,  
SWT.MULTI | SWT.V_SCROLL |  
SWT.H_SCROLL | SWT.BORDER);
```

More fundamentals

- *Bounds* of a Control is the rectangle that describes the location and size within its parent
 - Relative to the parent (except for *Shells*)
 - Can be queried by *Control.getBounds()*, *Control.getLocation()*, *Control.getSize()*
 - Can be moved or resized by changing their bounds
- Controls can be enabled/disabled and set to be visible/hidden
- z-order of a Control can be changed
- Listeners
 - Focus
 - Traverse
 - Move, Resize
 - Paint
 - Key, Mouse
 - Dispose

Display

- Glue between eSWT and the underlying window system
- Must be created and disposed by the user
- A thread is bound to the Display when created. This thread is called UI Thread.
 - only the UI thread can invoke UI operations (apartment threading)
- Provides API for the event loop
 - readAndDispatch()
 - sleep()
- UI thread-specific actions and timers
 - asyncExec(), syncExec()
 - timerExec()
 - update()
- Access to default system resources, such as fonts and colors
 - getSystemColor(), getSystemFont()
- And more utility APIs for UI generalities...

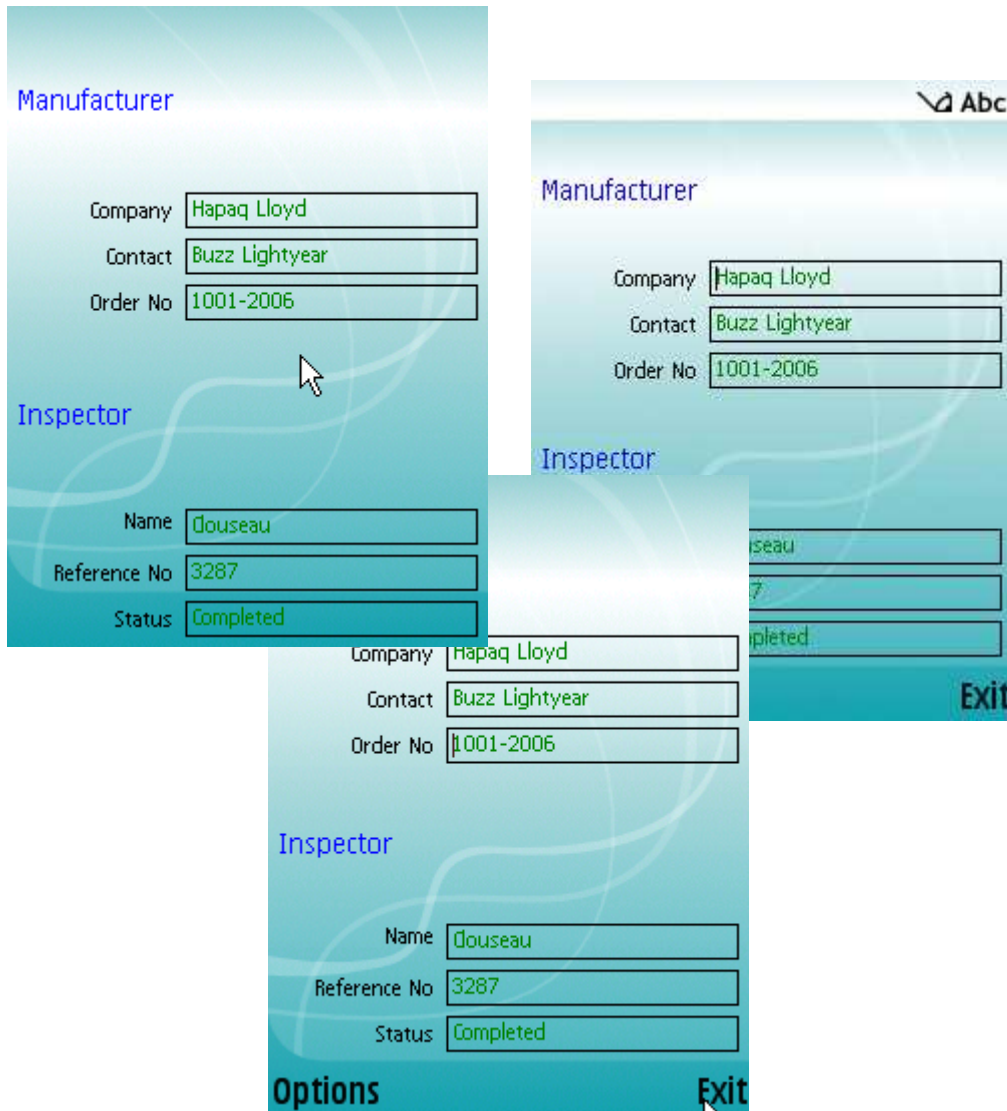
```
while (!shell.isDisposed()) {  
    if (!display.readAndDispatch())  
        display.sleep();  
}
```

Shell & Composite

- Composite is used to create container hierarchies
- Shell represents windows that are managed by the window manager
 - Inherits Composite
- Shells with Display as their parent are called *top-level shells*.
- Shells that have another shell as parent are called *dialog shells*.
- It is possible to create Shells with different trims (SHELL_TRIM, DIALOG_TRIM, NO_TRIM etc.)



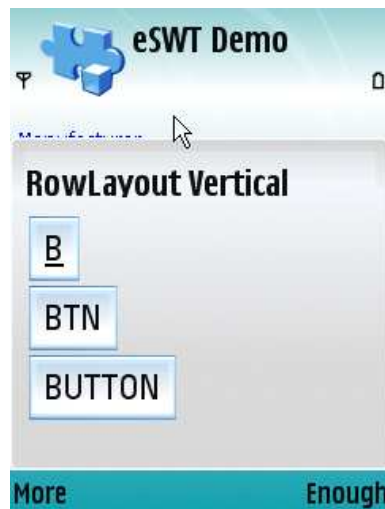
MobileShell



- A device tailored Shell that can change the trimmings dynamically
 - Top-level shell
 - Full screen mode
 - Allows key press polling
- Introduces status pane styles
 - NO_STATUS_PANE
 - SMALL_STATUS_PANE
 - LARGE_STATUS_PANE

Layout Managers

- 4 layout manager objects to control the size and position of components within a container
 - FillLayout: lays controls in a single row or column making them same size
 - RowLayout: lays controls in a rows or columns but it is more robust
 - FormLayout: lays according the the attachment properties (*FormAttachment*) of controls (left, top, bottom, right)
 - GridLayout: lays controls in a grid
- Add custom layout managers



Listeners

- `ControlListener` - control move, resize
- `DisposeListener` – widget disposal
- `FocusListener` – gained, lost
- `KeyListener` – pressed, released with character, keycode and modifier mask
- `MenuListener` – shown, hidden
- `ModifyListener` – text modified in editors
- `MouseListener` – mouse button down, up, double click (x, y)
- `MouseMoveListener` - move (x, y)
- `PaintListener` – a control needs to be (re) Painted (x,y,width,height)
- `SelectionListener` – generic widget selected event; default selection & selection
- `ShellListener` – activated, closed, deactivated, [iconified](#), [deiconified](#)
- `TraverseListener` – keyboard traverse event (next, previous; next group, previous group)
- `VerifyListener` – text verification in editors (end, start, text)
- `TreeListener` – expanded, collapsed

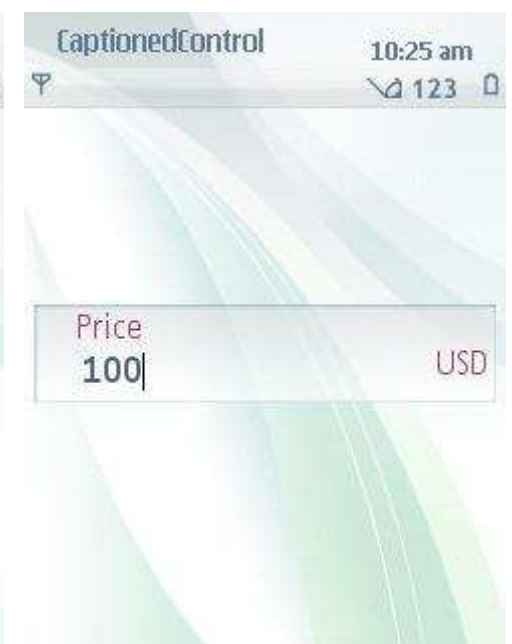
Basic controls

- Label (text or Image)
- Button (Push, Toggle, Check, Radio)
- Text
- List
- Combo (Read only, editable)



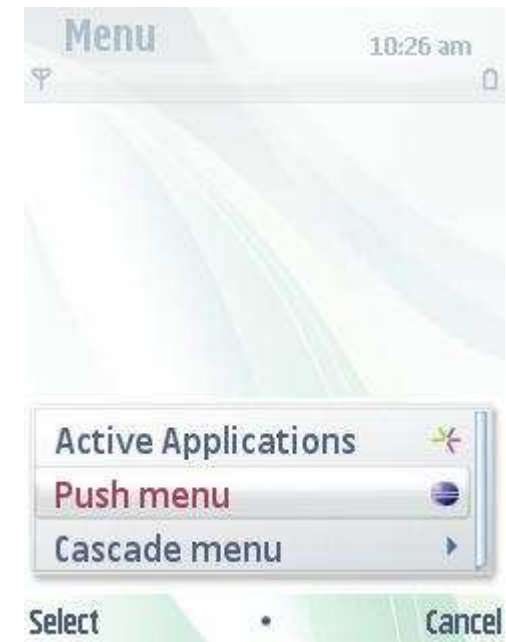
Basic controls - Mobile package

- ConstrainedText
- DateEditor
- HyperLink
- TextExtension
- CaptionedControl
- SortedList



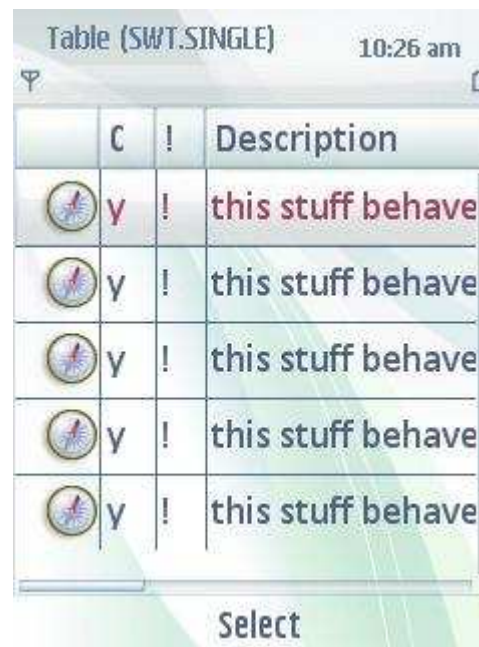
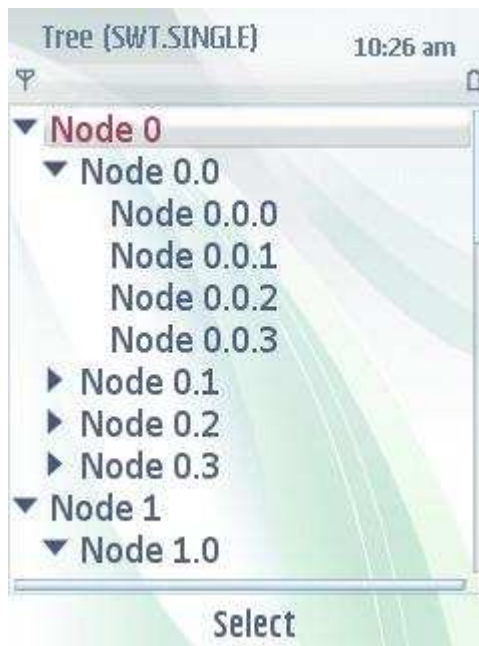
Menus

- Core package
 - Menu: provides several menu styles (BAR, DROP_DOWN, POP_UP, NO_RADIO_GROUP)
 - MenuItem: (PUSH, CHECK, RADIO, SEPARATOR, CASCADE)
 - Acts like buttons, setText(), setImage()
- Mobile package
 - Command: Familiar mobile concept for menus
 - Has logical types that are mapped to Soft keys (GENERAL, SELECT, OK, CANCEL, DELETE, BACK, EXIT, STOP, HELP)
 - Can be bound to focus context



Advanced Controls

- Table: A selectable control that are capable of displaying vertical rows of items
 - Single, Multiple
 - Text and image item
- Tree: A selectable control that is capable of displaying hierarchies of nodes
 - Single, Multiple
- Browser: A wrapper to the native browser that displays HTML content



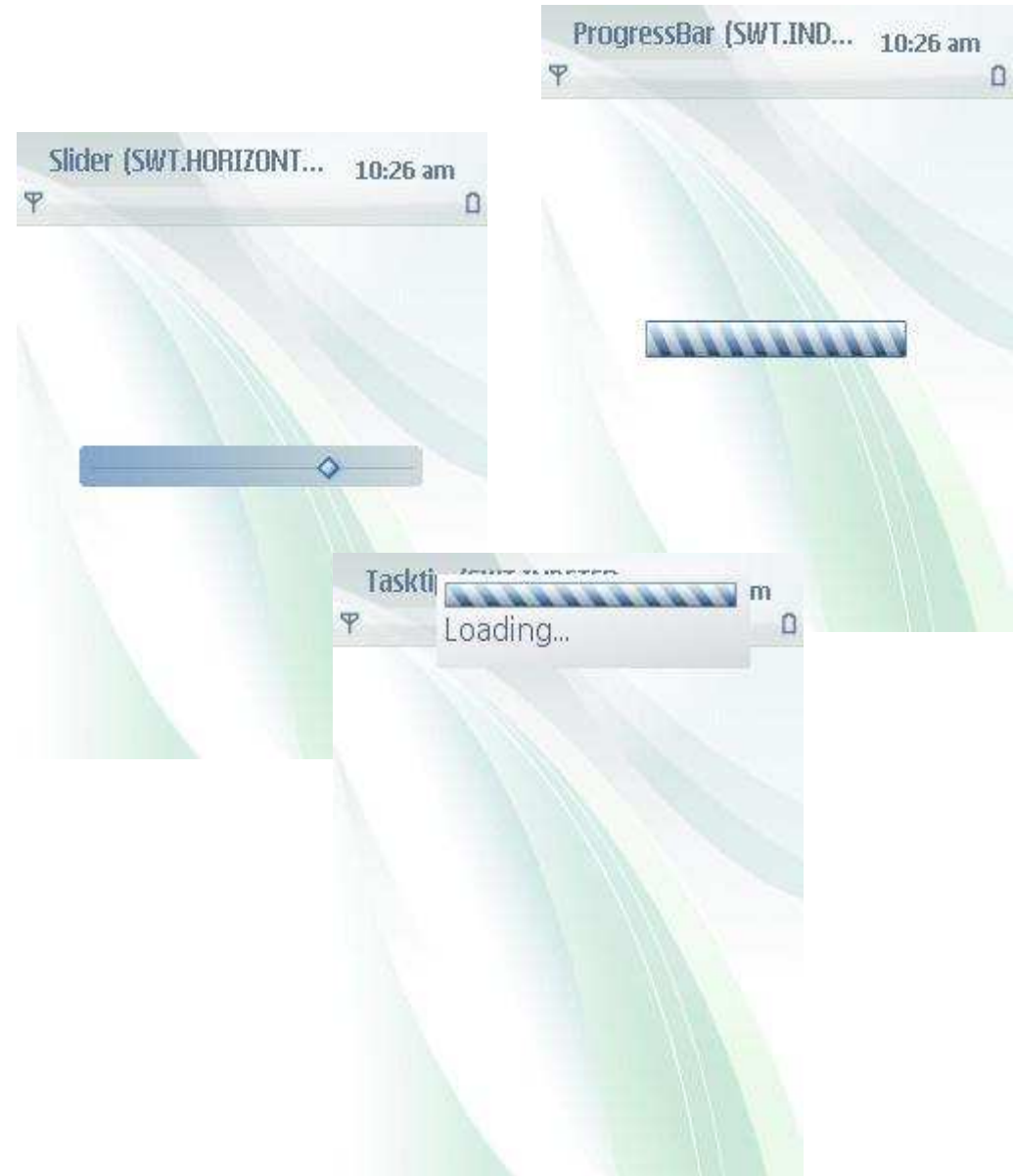
Advanced Controls - Mobile package

- **ListBox** : A list control with enhanced capabilities to display icons
- **ListView** : Selectable control that can display items in a multi-column way
- **MultipageDialog** : A tabbed dialog



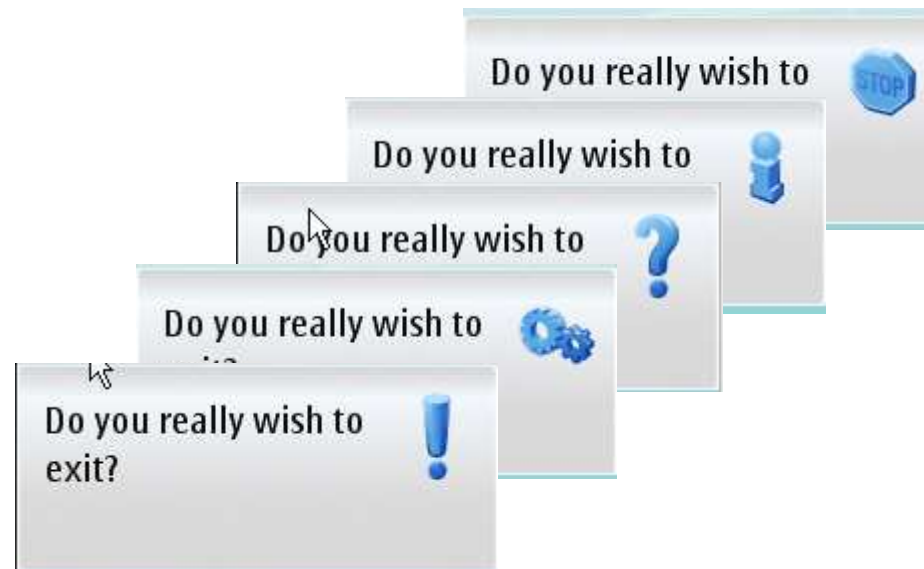
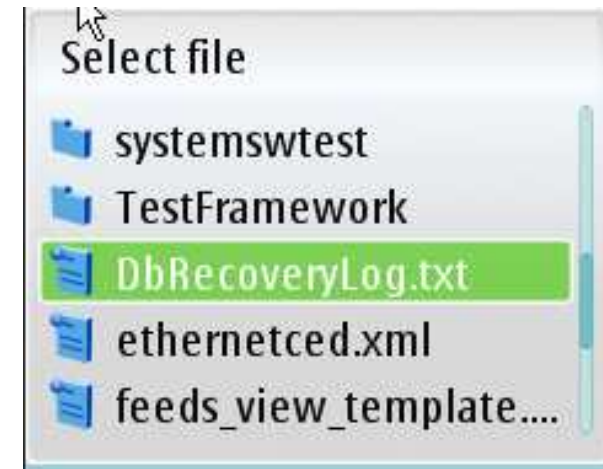
Range based controls

- Core package
 - ProgressBar
 - Vertical, Horizontal
 - Smooth, indeterminate
 - ScrollBar
 - created when their parent is created
 - queried from its parent
 - Slider
 - Vertical, Horizontal
 - Set Thumb and increment
- Mobile package
 - TaskTip
 - Suitable for providing info on long running tasks
 - Text and optional ProgressBar



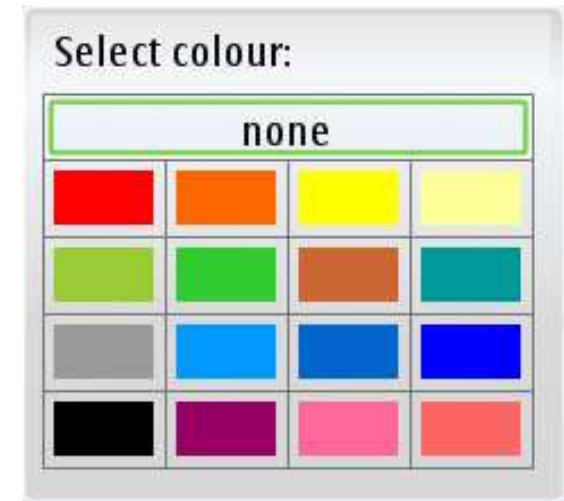
Dialogs

- **MessageBox**
 - 4 Different system Icons (working, information, warning, error)
 - Button combinations yes, no, cancel
retry
- **FileDialog**
 - Set initial filename
 - Set name and path filters

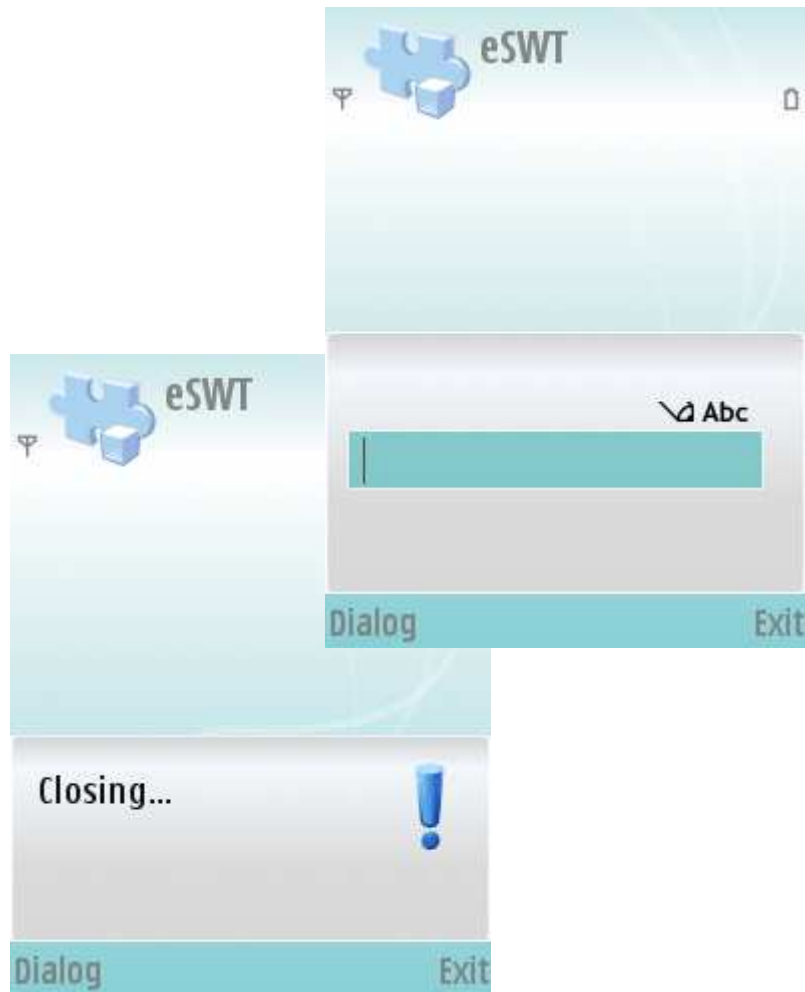


Dialogs

- DirectoryDialog
 - Can set a filter path to limit visible directories
- ColorDialog
 - Select a color from a predefined color palette
 - Set initial selection
- FontDialog
 - Select a font from the available fonts on the system.



Dialogs – Mobile package



- QueryDialog: Several query types
 - STANDARD: alphanumeric input
 - NUMERIC
 - TIME
 - DATE
 - PASSWORD
- TimedMessageBox:
 - 4 Different system Icons (working, information, warning, error)
 - Icons can be replaced

Change the focus order

- It is possible to change the tab order of controls in a Composite
 - `Composite.setTabList(Control[])`
 - An incomplete tab list makes some controls inaccessible
 - It is important to keep the tab list up to date with the control list

Clipboard

- Provides access to system clipboard
 - Within application
 - From one application to another
 - Binary and text data

```
// Copy text to clipboard
Clipboard cb = new Clipboard(display);
String textData = text.getText();
if (textData.length() > 0) {
    TextTransfer textTransfer = TextTransfer.getInstance();
    cb.setContents(new Object[]{textData}, new Transfer[]{textTransfer});
}
```

```
// Copy text from clipboard
Clipboard cb = new Clipboard(display);
TextTransfer transfer = TextTransfer.getInstance();
String data = (String)cb.getContents(transfer);
if (data != null) {
    text.insert(data);
}
```

Graphics

- class GC represents the principal interface to the graphics capabilities of the platform
- ImageLoader to load and save images to a file or stream
- Full access to the native fonts
 - Device.getFontList()

Tips: Building mobile applications using eSWT

- Do not rely on Shell trimmings some platforms do not support trimmings like SWT.CLOSE
- Do not use deep menu hierarchies.
- Use Commands in relation with the focus context to avoid the soft keys getting crowded
- Use MobileDevice, Screen, and Input to adjust the behavior at runtime
 - Active screens, active input methods may change
- Use layout managers
- Check the computed layout size and add scrollbars if greater than screen size
- Tailor your application according to aspect ratios

Right input widget is **VERY** important

	Multiple Lines	Numeric	Decimal	Phone Number	Date/Time	Duration/Offset	E-Mail	URL	Initial Case	Initial Input Mode	Turn off Prediction	Latin Input Only	Password
Text	✓	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘	✓
TextExtension	✓	✓	✓	✓	✓	✓	⊙	⊙	⊙	⊙	⊙	⊙	✓
ConstrainedText	⊘	⊙	⊙	⊙	⊘	⊘	⊘	⊘	⊘	⊘	⊘	⊘	⊘
DateEditor	⊘	⊘	⊘	⊘	⊙	⊙	⊘	⊘	⊘	⊘	⊘	⊘	⊘
QueryDialog	⊘	⊙	✓	✓	⊙	✓	✓	✓	⊘	⊘	⊘	⊘	⊙

Using eSWT with Java ME Midlets



- Required Software
 - Eclipse IDE
 - EclipseME plug-in (www.eclipseme.org)
 - Eclipse MTJ
 - NetBeans
 - Nokia S60 3.2 or later SDK (forum.nokia.com)
- HelloWorld Midlet available in eRCP project's wiki
 - http://wiki.eclipse.org/How_to_use_eSWT_with_Midlets
- Forum Nokia and Forum Nokia has further examples

eSWT availability

- Available from eRCP downloads for 1.1
 - WinCE 5.0 Professional
 - Windows Mobile 2003/5/6
 - Series 80 (eSWT 1.0)
- Installed on Device
 - All S60 3rd edition FP2 or later devices as part of MIDP runtime
 - Updates will be available from eRCP downloads

You can start developing today!

Demo

Future

- Working on eSWT 1.3
 - Link
 - ScrolledComposite
- Following the Eclipse E4 project
 - Animations & Effects
 - Declarative UI
 - Styles

Call for Action

- Start developing eSWT MIDlets today
- Provide feedback
 - Visit our website
 - <http://www.eclipse.org/ercp>
 - Participate on the newsgroup
 - news.eclipse.org/eclipse.dsdp.ercp
 - **Soon to be changed to** news.eclipse.org/eclipse.ercp
 - Join developer list
 - dsdp-ercp-dev@eclipse.org
 - **Soon to be changed to** ercp-dev@eclipse-org