~~Introduction to SWT (The Standard Widget Toolkit)~~

SWT Eye for the Swing Guy!

Steve Northover

SWT Team Lead
IBM Canada
eclipse.org/swt

TS-3853

Goal of This Talk

Write an SWT Application

After this talk you will understand what SWT is and how to write a simple SWT application

Agenda

Background: A Brief History of SWT

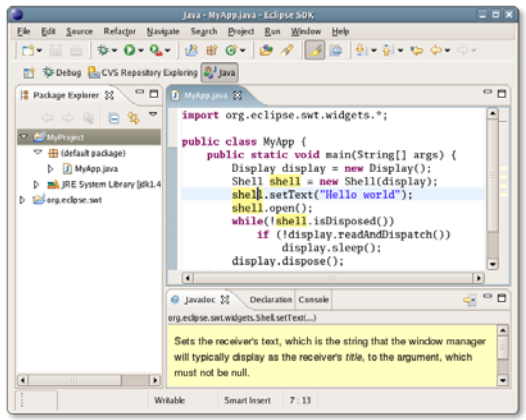
The Basics: Widgets and Graphics

Putting It All Together: An SWT Application

Sneak Peek: Advanced Topics

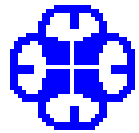
What Is SWT?

- SWT = “Standard Widget Toolkit”
- A native GUI Toolkit for Java™ platform
- Standard UI component for Eclipse
- Roughly equivalent to AWT/Swing



History of SWT

- Object Technology International (OTI)
  - VM's, Class Libraries, Compilers, IDE's
  - Configuration Management (ENVY/Manager)
- Smalltalk (in the early 90's)
  - Major Vendors: ParcPlace, Digitalk, IBM



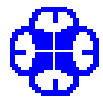
IBM Acquired OTI in 1996

Smalltalk Was… Java™!

- Object Oriented
- Byte Coded (VM, GC, JIT, Hot Replace…)
- Class Libraries (Collections, Streams…)
- Portable (Write Once, Run Anywhere™)

WORA (Write Once, Run Away!)

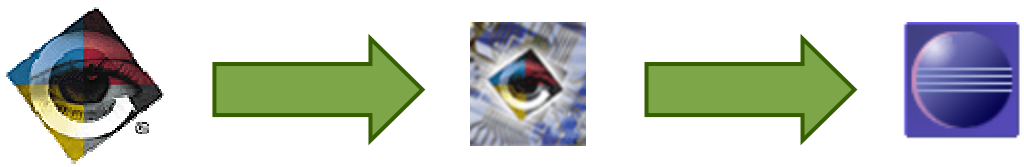
- ParcPlace Smalltalk (emulated widgets)
- Digitalk Smalltalk/V (native + emulated)
  - Windows, OS/2, Mac (emulated)
- IBM Smalltalk (native, written by OTI)
  - Windows, X/Motif, OS/2, Mac, Open Look



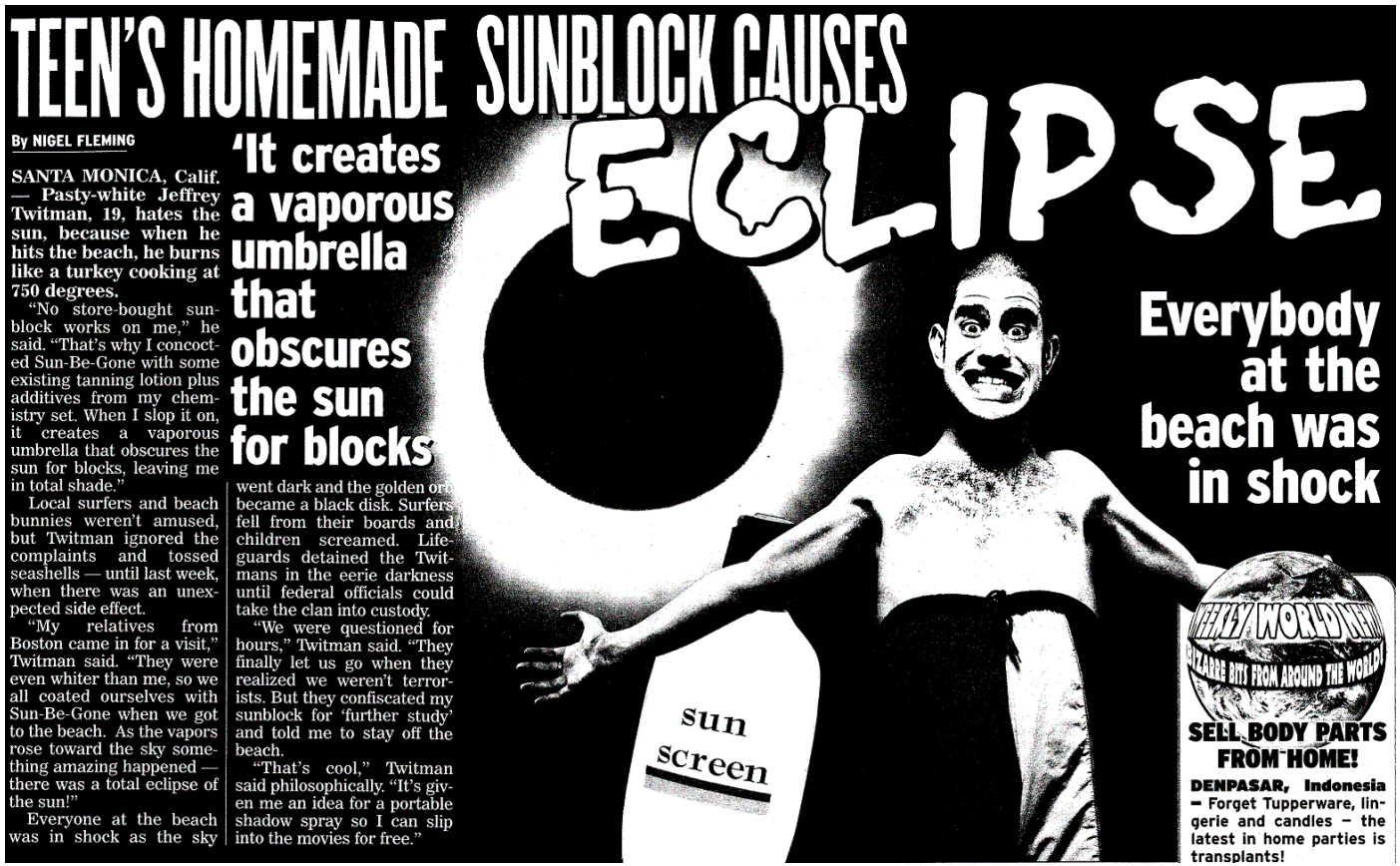
Native vs. Emulated!

The “Leap Frog” Experience

- VisualAge for Java™
  - Written in IBM Smalltalk
- VisualAge Micro Edition
  - Prototype in Java code using AWT/Swing
  - Rewritten in Java code using SWT
- Eclipse



What Is Eclipse?



TEEN'S HOMEMADE SUNBLOCK CAUSES ECLIPSE

By NIGEL FLEMING

SANTA MONICA, Calif. — Pasty-white Jeffrey Twitman, 19, hates the sun, because when he hits the beach, he burns like a turkey cooking at 750 degrees.

"No store-bought sunblock works on me," he said. "That's why I concocted Sun-Be-Gone with some existing tanning lotion plus additives from my chemistry set. When I slop it on, it creates a vaporous umbrella that obscures the sun for blocks, leaving me in total shade."

Local surfers and beach bunnies weren't amused, but Twitman ignored the complaints and tossed seashells — until last week, when there was an unexpected side effect.

"My relatives from Boston came in for a visit," Twitman said. "They were even whiter than me, so we all coated ourselves with Sun-Be-Gone when we got to the beach. As the vapors rose toward the sky something amazing happened — there was a total eclipse of the sun!"

Everyone at the beach was in shock as the sky went dark and the golden orb became a black disk. Surfers fell from their boards and children screamed. Lifeguards detained the Twitmans in the eerie darkness until federal officials could take the clan into custody.

"We were questioned for hours," Twitman said. "They finally let us go when they realized we weren't terrorists. But they confiscated my sunblock for 'further study' and told me to stay off the beach.

"That's cool," Twitman said philosophically. "It's given me an idea for a portable shadow spray so I can slip into the movies for free."

'It creates a vaporous umbrella that obscures the sun for blocks

Everybody at the beach was in shock

sun screen

WEEKLY WORLD NEWS
BIZARRE BITS FROM AROUND THE WORLD!

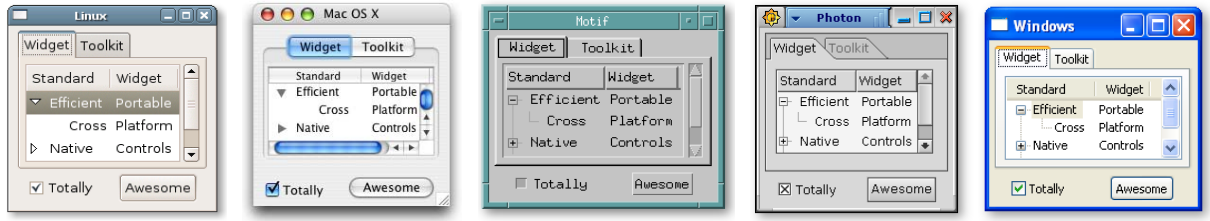
SELL BODY PARTS FROM HOME!

DENPASAR, Indonesia — Forget Tupperware, lingerie and candles — the latest in home parties is transplants!

"The SWT component is designed to provide efficient, portable access to the user-interface facilities of the operating system on which it is implemented"

eclipse.org/swt

SWT Is



- Efficient
  - Thin Java based layer over Java Native Interface calls to the OS
- Portable
  - Windows, GTK, Motif, Macintosh, Photon
  - Java ME, Java SE
- Native
  - Win32, GDI, GDI+, OLE, IE, Carbon, Cocoa, Core Graphics, Quick Draw, Safari, ATSUI, X Windows, X/t, Motif, GTK, GDK, Pango, cairo, ATK, Photon, Mozilla, QNX Voyager, Uniscribe...


A problem has been detected and Windows has been shut down to prevent damage
to your computer.
The problem seems to be caused by the following file: SPCMDCON.SY
PAGE_FAULT_IN_NONPAGED_AREA
If this is the first time you've seen this Stop error sc
restart your computer. If this screen appears again, f
these steps:
Check to make sure any new hardware or soft
If this is a new installation, ask your
for any Windows updates you might need.
If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use Safe Mode to remove or disable components, restart
your computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.
Technical information:
*** STOP: 0x0000
XFD3094C2,0x00000001,0xFBFE7617,0x00000000)
*** SPCMDCON.SYS - Address FBFE7617 base at FBFE5000, DateStamp 3d6dd67c
JUST KIDDING!

Myths

- SWT is better than Swing
- Swing is better than SWT
- SWT is “windows only”
- SWT is proprietary
- SWT applications are not portable
- …and more

Agenda

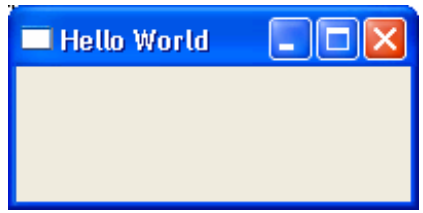
Background: A Brief History of SWT

The Basics: Widgets and Graphics

Putting It All Together: An SWT Application

Sneak Peek: Advanced Topics

Hello World in SWT



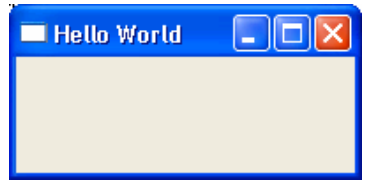

```
import org.eclipse.swt.widgets.*;

public class HelloWorld {
   public static void main(String[] args) {
      Display display = new Display();
      Shell shell = new Shell(display);
      shell.setText("Hello World");
      shell.setSize(200, 100);
      shell.open();
      while (!shell.isDisposed()) {
            if (!display.readAndDispatch())
                  display.sleep ();
      }
      display.dispose ();
   }
}
```

Display

- Connection to the window system
- Represents the “screen”
- Contains a list of Shells
- Normally a singleton

```
public static void main(String[] args) {
   Display display = new Display();
   Shell shell = new Shell(display);
```

Shell



- Represents a “window” on the “screen”
- Root of a tree of Composites and Controls

```
public static void main(String[] args) {
      Display display = new Display();
      Shell shell = new Shell(display);
      shell.setText("Hello World");
      shell.setSize(200, 100);
      shell.open();
```

Composite and Control

- Composite
  - Control that contains other Composites and Controls
- Control
  - A heavyweight operating system object
  - buttons, labels, entry fields, tables, toolbars, and trees are controls (Shells and Composites too)

Shell, Composite, Control are subclasses of Widget

The Event Loop

- Repeatedly reads and dispatches events
- Yields CPU when no events are available
- Applications choose when to exit

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

Events

- JavaBeans™ event model (“typed” events)
- Events come from the event loop (and from widget operations)
- Low level “untyped” events also available

Freeing Resources

- Operating system resources are explicitly released by the programmer

```
display.dispose();
```

Rule #1: “If you created it, you dispose it”

Rule #2: “Disposing a parent disposes the children”

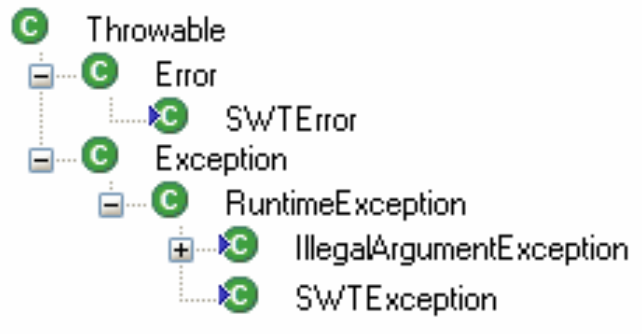
Standard Constructors

- Widgets must be created with a parent
- Style bits used for create-only attributes

```
//Create a push button control
Button button = new Button(shell, SWT.PUSH);

//Create a single line entry field with a border
Text text = new Text(group, SWT.SINGLE|SWT.BORDER);

//Create a shell with dialog trimmings
Shell dialog = new Shell(shell, SWT.DIALOG_TRIM);
```

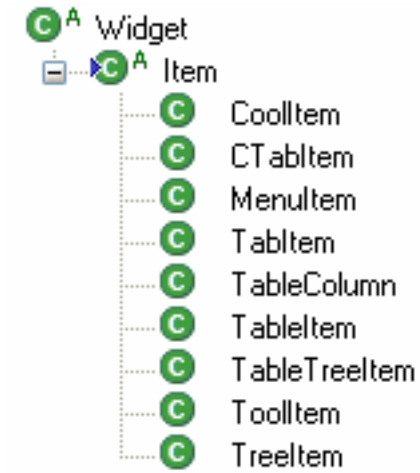
Errors and Exceptions

- SWTError
  - An unrecoverable error
  - The operating system failed
- SWTException
  - A recoverable error occurred
  - Invalid thread access, etc.
- IllegalArgumentException
  - A recoverable error (argument is invalid)
  - Argument cannot be null, etc.



Items

- A lightweight operating system object
- Always occur within a specific widget
  - A Tree contains TreeItems
  - A Table contains TableItems
  - A Menu contains MenuItems
  - …and more



Threads

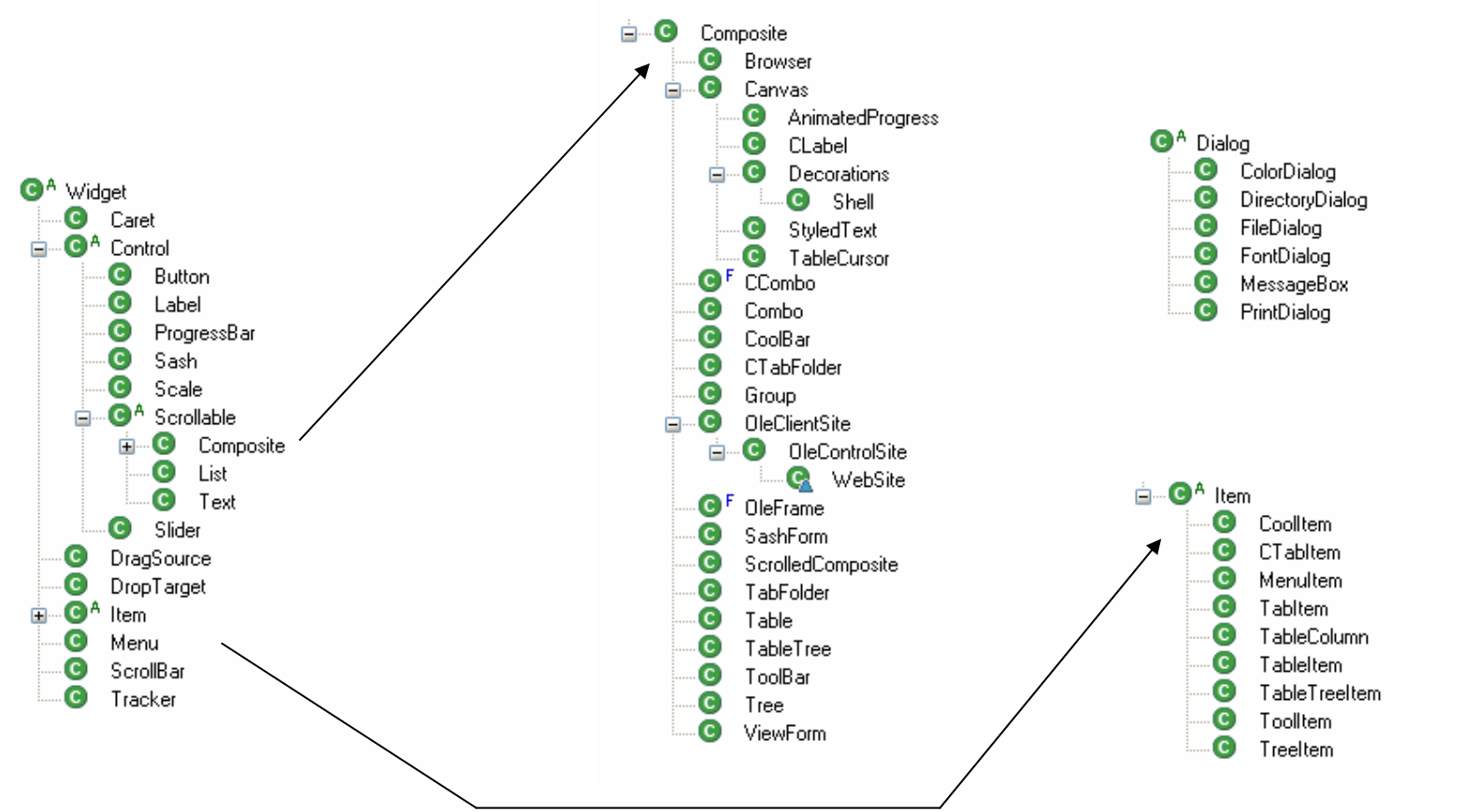
- Single UI-thread (“apartment threaded”)
  - Widget operations must be called from the UI-thread
  - Runnables can be queued to run in the UI-thread
- Background Threads
  - Use Display.syncExec(), asyncExec(), wake()
  - Graphics operations may be called from any thread

Widget Packages

- org.eclipse.swt.widgets
- org.eclipse.swt.dnd
  - Drag and Drop and Clipboard
- org.eclipse.swt.browser
  - HTLM Browser control
- org.eclipse.swt.custom
  - Custom Controls for Eclipse
- org.eclipse.swt.ole.win32
  - OLE (ActiveX) Support

Widget Classes—All of Them!

(Dialogs, D&D, OLE, Browser, Custom Controls)



Graphics Classes—All of Them!

- Resource Based Objects
  - GC (Graphics Context), Color, Image, Font, Path, Region, Transform, Pattern, TextLayout
- Java Based Objects
  - Point, Rectangle, RGB, ImageData, FontData, PaletteData, PathData, TextStyle, FontMetrics, GlyphMetrics, ImageLoader

GC (Graphics Context)

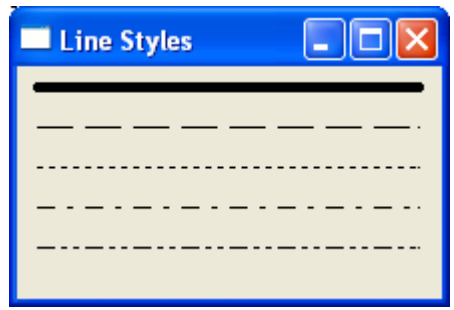
- All line, shape, text, image drawing, clipping, alpha, anti-alias and filling operations
- Created on Control, Image, Display or Printer
  - new GC (Control)
  - new GC (Image)
  - new GC (Display)
  - new GC (Printer)
- Call dispose() when done

GC Line Draw Methods

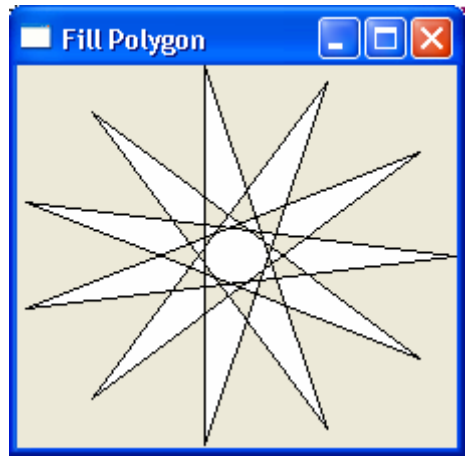
- drawLine(int x1, int y1, int x2, int y2)
- drawPolyline(int[] xyArray)
- setLineWidth(int width)
- setLineStyle(int style)
  - SWT.LINE_SOLID, SWT.LINE_DASH, SWT.LINE_DOT, SWT.LINE_DASHDOT, SWT.LINE_DASHDOTDOT, SWT.LINE_CUSTOM

Draw Lines on a Shell

```
shell.addPaintListener(new PaintListener() {
   public void paintControl(PaintEvent event) {
      GC gc = event.gc;
      gc.setLineWidth(5);
      gc.setLineStyle(SWT.LINE_SOLID);
      gc.drawLine(10, 10, 200, 10);
      gc.setLineWidth(1);
      gc.setLineStyle(SWT.LINE_DASH);
      gc.drawLine(10, 30, 200, 30);
      gc.setLineStyle(SWT.LINE_DOT);
      gc.drawLine(10, 50, 200, 50);
      gc.setLineStyle(SWT.LINE_DASHDOT);
      gc.drawLine(10, 70, 200, 70);
      gc.setLineStyle(SWT.LINE_DASHDOTDOT);
      gc.drawLine(10, 90, 200, 90);
}});
```



Draw and Fill a Polygon on a Shell



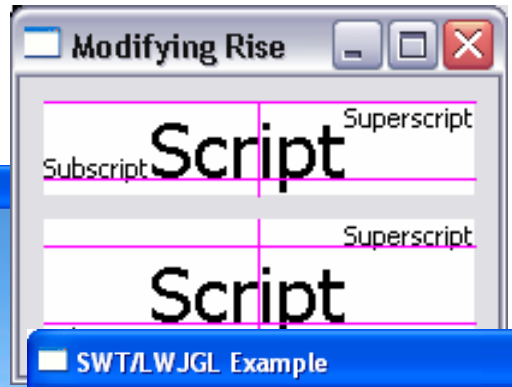

```
public void paintControl(PaintEvent event) {
    Rectangle bounds = shell.getClientArea();
    center.x = bounds.x + bounds.width/2;
    center.y = bounds.y + bounds.height/2;
    int pos = 0;
    for (int i = 0; i < points; ++i) {
        double r = Math.PI*2 * pos/points;
        radial[i*2] = (int)((1+Math.cos(r))*center.x);
        radial[i*2+1] = (int)((1+Math.sin(r))*center.y);
        pos = (pos + points/2) % points;
    }
    event.gc.setBackground(
        display.getSystemColor(SWT.COLOR_WHITE));
    event.gc.fillPolygon(radial);
    event.gc.drawPolygon(radial);
}});
```

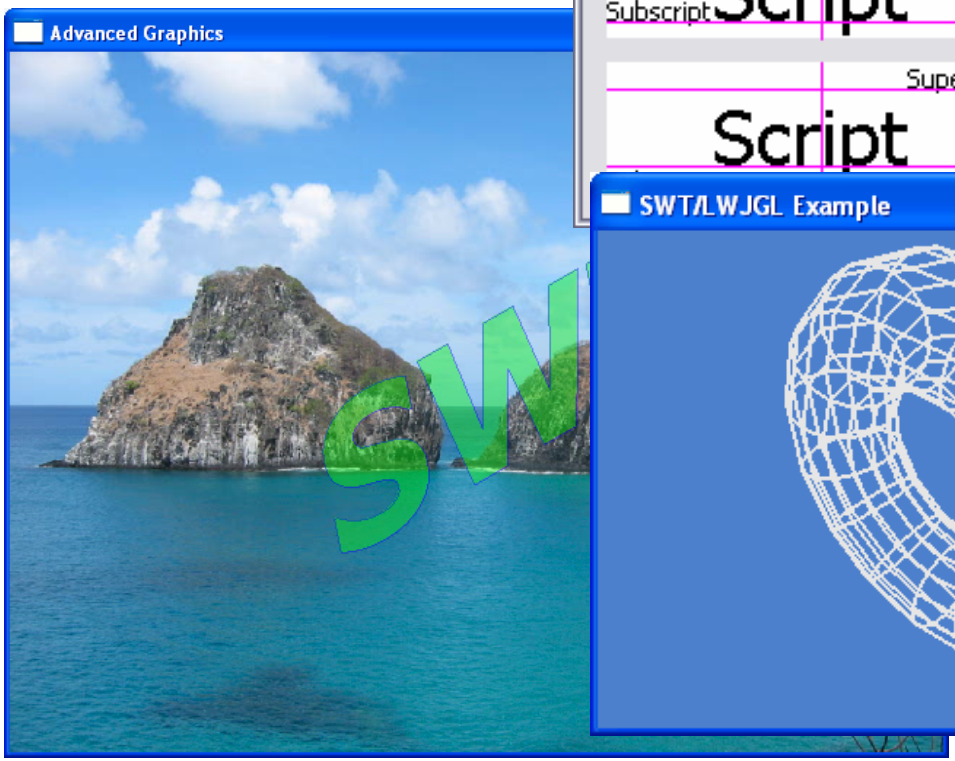
Should This GC Be Disposed?

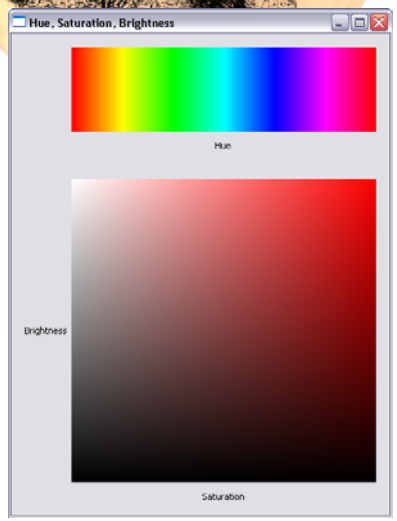
```
shell.addPaintListener(new PaintListener() {
    public void paintControl(PaintEvent event) {
      …
      //do I need to dispose this crazy thing?
      GC gc = event.gc;
      …
}});
```

No. It was not created with “new GC()”

What Else Can SWT Graphics Do?





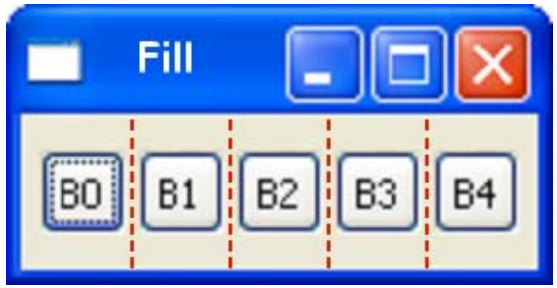


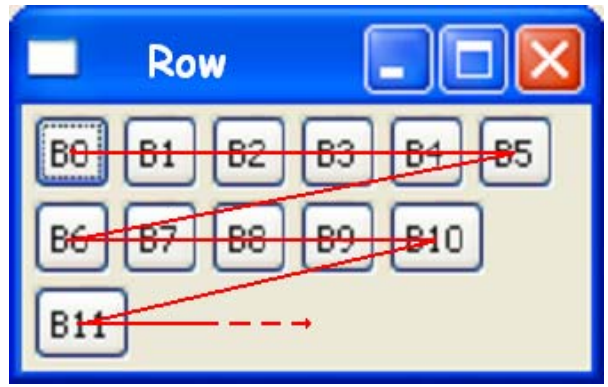
Layout Overview

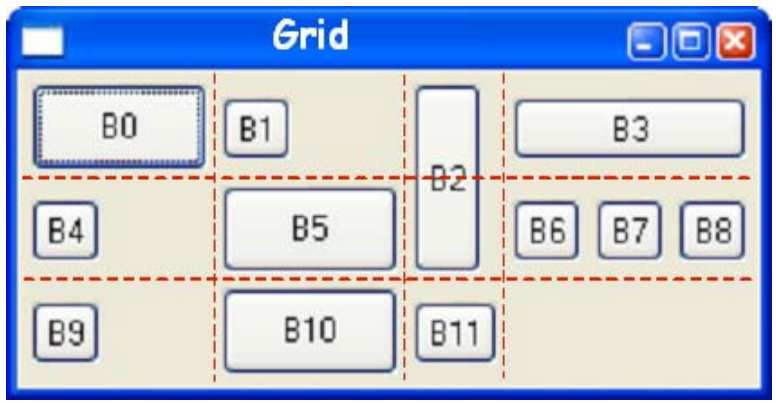
- Layout
  - An algorithm to position and resize controls
  - Use Composite.setLayout()
- Layout Data
  - Algorithm specific data associated with each control
  - Use Control.setLayoutData()

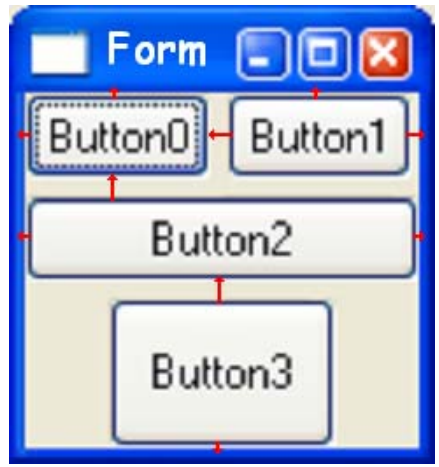
Layout Classes

- FillLayout
- RowLayout
- GridLayout
- FormLayout
- StackLayout









Agenda

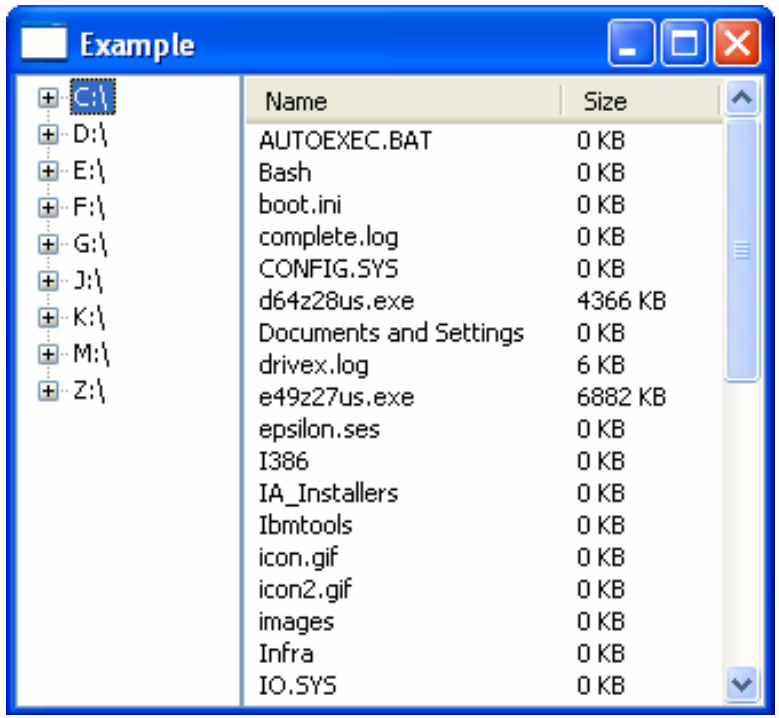
Background: A Brief History of SWT

The Basics: Widgets and Graphics

Putting It All Together: An SWT Application

Sneak Peek: Advanced Topics

Example: FileExplorer



Create the Display

```
public static void main(String[] args) {
    Display display = new Display();
    Shell shell = new Shell(display);
    shell.setText("Example");
    shell.setLayout(new FillLayout());
    Tree tree = createTree(shell);
    Table table = createTable(shell);
    createListeners(tree, table);
    shell.open();
    while (!shell.isDisposed()) {
        if (!display.readAndDispatch()) display.sleep();
    }
    display.dispose();
}
```

1

2

3

Create the Shell

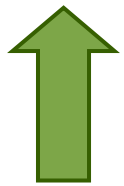
```
public static void main(String[] args) {
    Display display = new Display();
    Shell shell = new Shell(display);
1   shell.setText("Example");
    shell.setLayout(new FillLayout());
    Tree tree = createTree(shell);
    Table table = createTable(shell);
    createListeners(tree, table);
2   shell.open();
    while (!shell.isDisposed()) {
        if (!display.readAndDispatch()) display.sleep();
    }
    display.dispose();
}
```

Create the Tree

```
public static void main(String[] args) {
    Display display = new Display();
    Shell shell = new Shell(display);
    shell.setText("Example");
    shell.setLayout(new FillLayout());
    Tree tree = createTree(shell);
    Table table = createTable(shell);
    createListeners(tree, table);
    shell.open();
    while (!shell.isDisposed()) {
        if (!display.readAndDispatch()) display.sleep();
    }
    display.dispose();
}
```

Create the Tree with TreeItems

```
static Tree createTree (Composite parent) {
    Tree tree = new Tree(parent, SWT.BORDER);
    createTreeItems(tree, null, File.listRoots());
    return tree;
}
```



First Level

Create the TreeItems

```
static void createTreeItems(Tree tree, TreeItem parent,
File [] files) {
    if (files == null) return;
1   for (int i = 0; i < files.length; i++) {
        if (files[i].isDirectory()) {
            TreeItem item;
            if (parent == null) {
                item = new TreeItem(tree, SWT.NONE);
                item.setText(files[i].toString());
            } else {
                item = new TreeItem(parent, SWT.NONE);
                item.setText(files[i].getName());
            }
            item.setData(files[i]);
            new TreeItem(item, SWT.NULL); // force a '+'
        }
    }
}
```

Create the TreeItems

```
static void createTreeItems(Tree tree, TreeItem parent,
File [] files) {
    if (files == null) return;
1   for (int i = 0; i < files.length; i++) {
        if (files[i].isDirectory()) {
            TreeItem item;
            if (parent == null) {
                item = new TreeItem(tree, SWT.NONE);
                item.setText(files[i].toString());
2           } else {
                item = new TreeItem(parent, SWT.NONE);
                item.setText(files[i].getName());
            }
            item.setData(files[i]);
            new TreeItem(item, SWT.NULL); // force a '+'
        }
    }
}
```

Create the TreeItems

```
static void createTreeItems(Tree tree, TreeItem parent,
File [] files) {
    if (files == null) return;
1   for (int i = 0; i < files.length; i++) {
        if (files[i].isDirectory()) {
            TreeItem item;
            if (parent == null) {
                item = new TreeItem(tree, SWT.NONE);
                item.setText(files[i].toString());
2           } else {
                item = new TreeItem(parent, SWT.NONE);
                item.setText(files[i].getName());
3           }
            item.setData(files[i]);
            new TreeItem(item, SWT.NULL); // force a '+'
        }
    }
}
```

Create the TreeItems

```
static void createTreeItems(Tree tree, TreeItem parent,
File [] files) {
    if (files == null) return;
1   for (int i = 0; i < files.length; i++) {
        if (files[i].isDirectory()) {
            TreeItem item;
            if (parent == null) {
                item = new TreeItem(tree, SWT.NONE);
                item.setText(files[i].toString());
2           } else {
                item = new TreeItem(parent, SWT.NONE);
                item.setText(files[i].getName());
3           }
            item.setData(files[i]);
4           new TreeItem(item, SWT.NULL); // force a '+'
          }
    }
}
```

Create the Table

```
public static void main(String[] args) {
    Display display = new Display();
    Shell shell = new Shell(display);
    shell.setText("Example");
    shell.setLayout(new FillLayout());
    Tree tree = createTree(shell);
    Table table = createTable(shell);
    createListeners(tree, table);
    shell.open();
    while (!shell.isDisposed()) {
        if (!display.readAndDispatch()) display.sleep();
    }
    display.dispose();
}
```

Create the Table Without TableItems

```
static Table createTable(Composite parent) {
    Table table = new Table(parent, SWT.BORDER);
    table.setHeaderVisible(true);
    String [] titles = {"Name", "Size"};
    for (int i=0; i<titles.length; i++) {
        TableColumn column =
                new TableColumn(table, SWT.NONE);
        column.setText(titles[i]);
    }
    return table;
}
```

Create the Listeners

```
public static void main(String[] args) {
    Display display = new Display();
    Shell shell = new Shell(display);
    shell.setText("Example");
    shell.setLayout(new FillLayout());
    Tree tree = createTree(shell);
    Table table = createTable(shell);
    createListeners(tree, table);
    shell.open();
    while (!shell.isDisposed()) {
        if (!display.readAndDispatch()) display.sleep();
    }
    display.dispose();
}
```

Create the Listeners

```
static void createListeners(final Tree tree,
final Table table) {
    tree.addSelectionListener(new SelectionAdapter() {
        public void widgetSelected(SelectionEvent event) {
1           TreeItem item = (TreeItem) event.item;
            File file = (File) item.getData();
            createTableItems(table, file.listFiles());
        }});
    tree.addTreeListener(new TreeAdapter() {
        public void treeExpanded(TreeEvent event) {
            TreeItem item = (TreeItem) event.item;
            if (item.getItem(0).getData() != null) return;
            item.removeAll();
            File file = (File) item.getData();
            createTreeItems(null, item, file.listFiles());
        }});
}}
```

Create the Listeners

```
static void createListeners(final Tree tree,
final Table table) {
    tree.addSelectionListener(new SelectionAdapter() {
        public void widgetSelected(SelectionEvent event) {
1           TreeItem item = (TreeItem) event.item;
            File file = (File) item.getData();
2           createTableItems(table, file.listFiles());
        }});
    tree.addTreeListener(new TreeAdapter() {
        public void treeExpanded(TreeEvent event) {
            TreeItem item = (TreeItem) event.item;
            if (item.getItem(0).getData() != null) return;
            item.removeAll();
            File file = (File) item.getData();
            createTreeItems(null, item, file.listFiles());
        }});
}}
```

Create the Listeners

```
static void createListeners(final Tree tree,
final Table table) {
    tree.addSelectionListener(new SelectionAdapter() {
        public void widgetSelected(SelectionEvent event) {
1           TreeItem item = (TreeItem) event.item;
            File file = (File) item.getData();
2           createTableItems(table, file.listFiles());
        }});
    tree.addTreeListener(new TreeAdapter() {
        public void treeExpanded(TreeEvent event) {
            TreeItem item = (TreeItem) event.item;
3           if (item.getItem(0).getData() != null) return;
            item.removeAll();
            File file = (File) item.getData();
            createTreeItems(null, item, file.listFiles());
        }});
}}
```

Create the Listeners

```
static void createListeners(final Tree tree,
final Table table) {
    tree.addSelectionListener(new SelectionAdapter() {
        public void widgetSelected(SelectionEvent event) {
1           TreeItem item = (TreeItem) event.item;
            File file = (File) item.getData();
2           createTableItems(table, file.listFiles());
        }});
    tree.addTreeListener(new TreeAdapter() {
        public void treeExpanded(TreeEvent event) {
            TreeItem item = (TreeItem) event.item;
3           if (item.getItem(0).getData() != null) return;
4           item.removeAll();
            File file = (File) item.getData();
            createTreeItems(null, item, file.listFiles());
        }});
}}
```

Create the Listeners

```
static void createListeners(final Tree tree,
final Table table) {
    tree.addSelectionListener(new SelectionAdapter() {
        public void widgetSelected(SelectionEvent event) {
            TreeItem item = (TreeItem) event.item;
            File file = (File) item.getData();
            createTableItems(table, file.listFiles());
        }});
    tree.addTreeListener(new TreeAdapter() {
        public void treeExpanded(TreeEvent event) {
            TreeItem item = (TreeItem) event.item;
            if (item.getItem(0).getData() != null) return;
            item.removeAll();
            File file = (File) item.getData();
            createTreeItems(null, item, file.listFiles());
        }});
}}
```

1

2

3

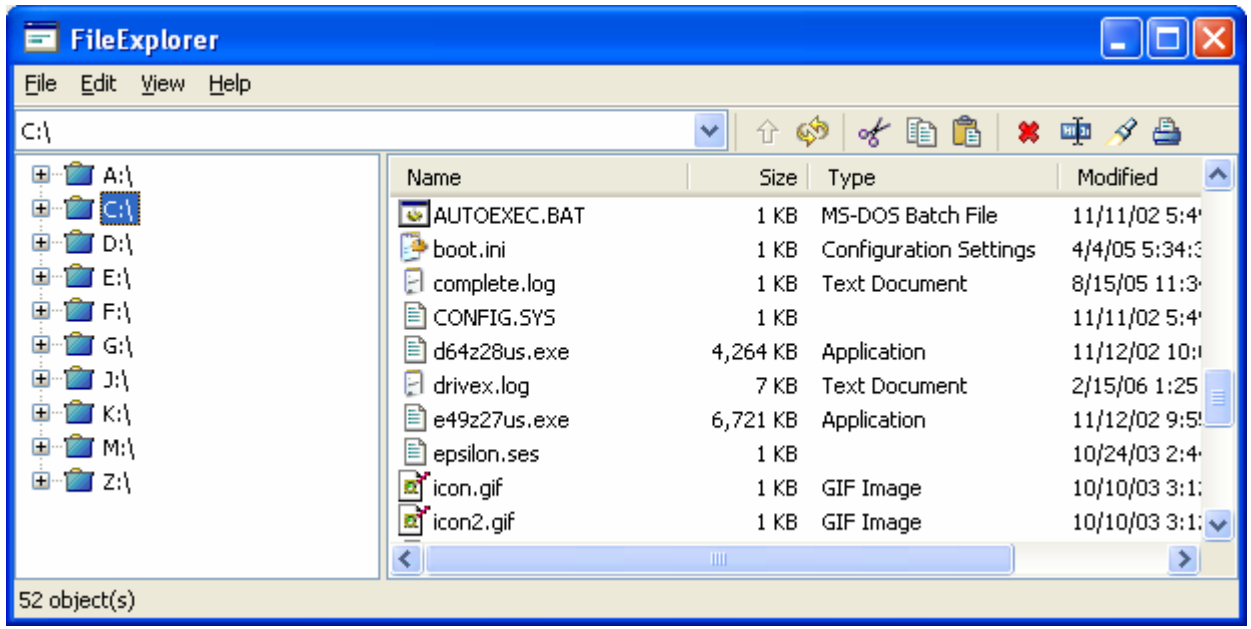
4

Other Levels

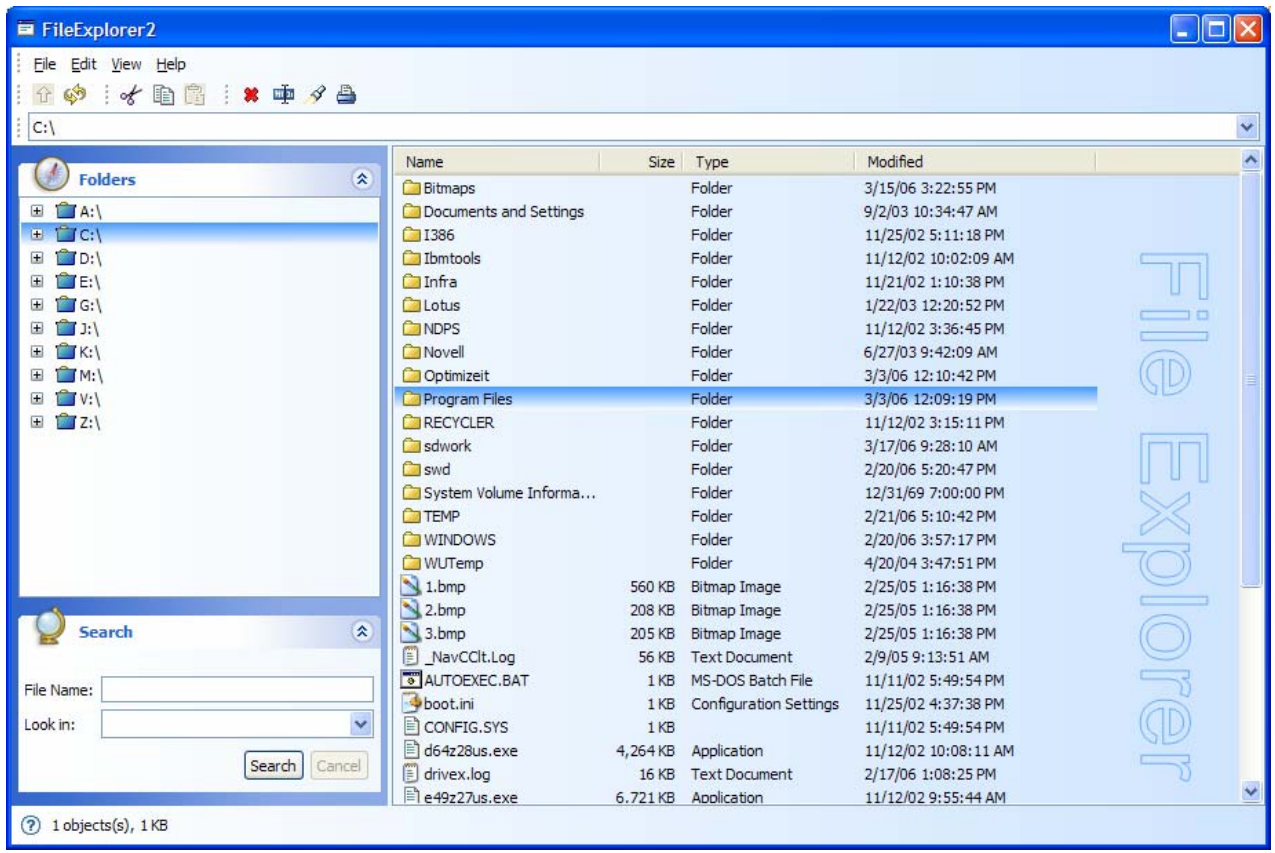
Create the TableItems

```
static void createTableItems(Table table, File [] files) {
    table.removeAll();
    if (files == null) return;
    for (int i = 0; i < files.length; i++) {
        TableItem item = new TableItem(table, SWT.NULL);
        item.setText(new String [] {
                files[i].getName(),
                files[i].length() / 1000 + " KB"
        });
    }
    for (int i=0; i<table.getColumnCount(); i++) {
        table.getColumn (i).pack();
    }
}
```

More Features



Yet More Features



And Finally

“Crown Roast of Frankfurters”

Agenda

Background: A Brief History of SWT

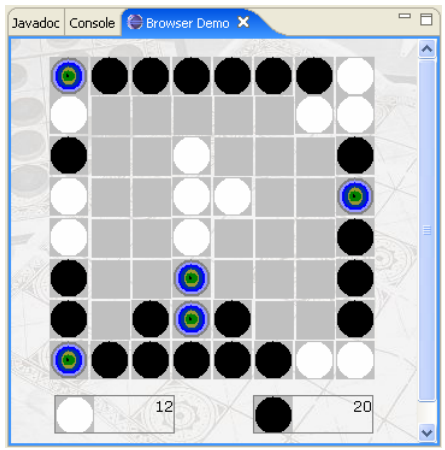
The Basics: Widgets and Graphics

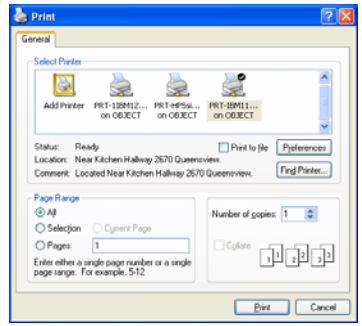
Putting It All Together: An SWT Application

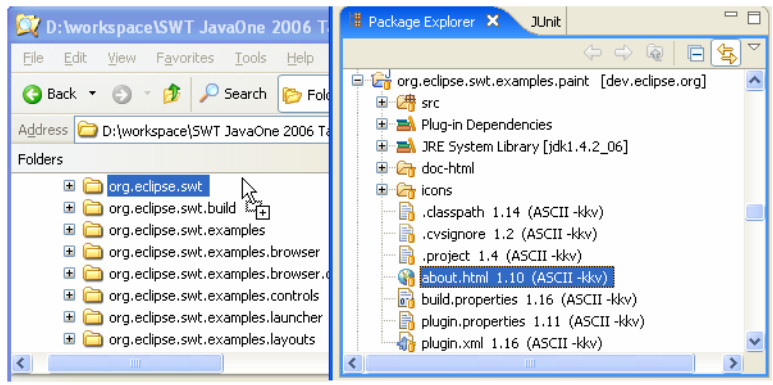
Sneak Peek: Advanced Topics

Sneak Peek: Advanced Topics

- Browser
- Drag and Drop
- Printing
- Program
- Accessibility
- OLE (win32)
- AWT/Swing interop







~~Introduction to SWT (The Standard Widget Toolkit)~~

Steve Northover

SWT Team Lead
IBM Canada
eclipse.org/swt

SWT Eye for the Swing Guy!

TS-3853