

Absolutes Layout

2006-05-31 19:49:24

Des öfteren benötigt man einen `LayoutManager` in Java, mit dem man die Positionen der enthaltenen Komponenten frei wählen kann. Setzt man den `LayoutManager` auf `null`, so kriegt man gar nichts zu sehen. Und wenn man es doch noch schafft, indem man allen Komponenten mit Hilfe von `setBounds` eine Position gibt, machen Komponenten, wie `JScrollPane` schlapp, da sie keinerlei Informationen über `preferred-`, `min-` und `max-Size` erhalten.

Der folgende `LayoutManager` ermöglicht es, Komponenten beliebig zu platzieren, und berechnet dennoch eine plausible `preferred-`, `min-` und `max-Size` für die Elternkomponente, indem es feststellt, welche Breite und Höhe notwendig ist, um die Komponenten an den äußersten Rändern noch anzuzeigen.

```
import java.awt.Component; import java.awt.Container; import java.awt.Dimension; import java.awt.Insets;
import java.awt.LayoutManager; import java.awt.Rectangle; public class
AbsolutePositioningLayoutManager implements LayoutManager { public void
removeLayoutComponent(Component comp) {} public void layoutContainer(Container parent) {} public
void addLayoutComponent(String name, Component comp) {} public Dimension
minimumLayoutSize(Container parent) { return preferredLayoutSize(parent); } public Dimension
preferredLayoutSize(Container parent) { int maxX = 0; int maxY = 0; Rectangle bounds = new
Rectangle(); Insets insets = parent.getInsets(); int componentMaxX; int componentMaxY; for (int i=0;
i<parent.getComponentCount(); i++){ bounds = parent.getComponent(i).getBounds(bounds);
componentMaxX = bounds.x+bounds.width+insets.left*2+insets.right*2; componentMaxY =
bounds.y+bounds.height+insets.bottom*2+insets.top*2; if (componentMaxX>maxX) maxX =
componentMaxX; if (componentMaxY>maxY) maxY = componentMaxY; } return new
Dimension(maxX, maxY); }
```