The Auckland Layout Editor: An Improved GUI Layout Specification Process

Clemens Zeidler, Christof Lutteroth, Gerald Weber
University of Auckland
38 Princes Street
Auckland 1010, New Zealand
czei002@aucklanduni.ac.nz,
{christof, gerald}@cs.auckland.ac.nz

Wolfgang Stürzlinger
York University
4700 Keele St.
Toronto, Canada M3J 1P3
wolfgang@cse.yorku.ca

ABSTRACT

Constraint-based layout managers are more powerful than the common grid, grid-bag, and group layout managers. However, they are also more complex and come with potential problems such as over-constrained specifications and overlap in a GUI. Current GUI builders have little support for layout constraints, and it is not clear how such constraints can be made easily accessible to GUI designers.

We will demonstrate a GUI builder – the Auckland Layout Editor (ALE) – that addresses these challenges, by allowing GUI designers to specify constraint-based layouts using only simple mouse operations. ALE guarantees that all operations lead to sound specifications, making sure that the layout is solvable and non-overlapping. To achieve the latter, we propose an algorithm that automatically generates the missing constraints that are necessary to keep a layout non-overlapping. Today's applications need to run on multiple devices with different screen sizes. For this a layout must have a good appearance at different sizes. To aid the designer in creating a layout with good resizing behavior, we propose a novel automatic layout preview, which displays the layout at its minimal and at an enlarged size chosen to visualize layout problems directly.

Categories and Subject Descriptors

H5.2 [**Information interfaces and presentation**]: User Interfaces. - Graphical user interfaces.

General Terms

Design, Human Factors, Algorithms.

Keywords

GUI builder, layout editing, layout manager, constraint-based layout

CHINZ '12, July 02 - 03 2012, Dunedin, New Zealand Copyright 2012 ACM 978-1-4503-1474-9/12/07 $15.00.

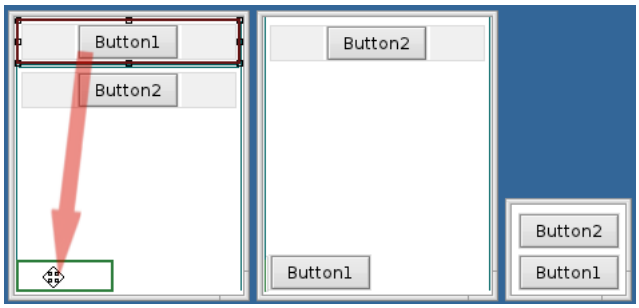

Figure 1: Dropping Button 1 into the left bottom corner of the the large empty area. The result is shown on the middle. On the right a preview window at minimal size is shown.

1. DEMONSTRATION

During the demonstration the user is able to try various aspects of the ALE layout builder. The user gets the opportunity to get familiar with ALE and design an arbitrary constraint-based layout. Because ALE is designed to be easy and intuitive to use, the user only gets a short introduction to the builder interface. Screenshots of some sample layouts will be provided to guide the user in creating non-trivial layouts.

As in most GUI builders, a GUI can be created and edited by simple drag'n drop operations. The set of necessary operations is moving, swapping, resizing, inserting, and removing of layout items. An example of a move drag'n drop operation is shown in Figure 1. After each layout edit operation, ALE calculates a set of constraints to ensure that the widgets do not overlap each other for all layout sizes.

To demonstrate the advantage of this non-overlap algorithm, we disable this feature at one stage during the demo, and show the user layout specifications that cause widgets to overlap after a layout resize. We also demonstrate the layout preview window, which makes potential overlap immediately visible, even though the layout may appear overlap-free in the in the design window. When enabling the non-overlap algorithm, the preview window immediately shows that no overlap will occur (right preview window in Figure 1). Furthermore, we show examples where the preview windows reveals problems in the specifications that are not visible in the design window.