

Introduction to Adobe Flex 3

Guide to Auto-Layout

Trademarks

ChikaraDev and the ChikaraDev logo are trademarks of ChikaraDev. Such trademarks may be registered in the United States or in other jurisdictions, including internationally. This manual may include trademarks, service marks, or trade names of Adobe Systems Incorporated, Inc. and other companies. Such trademarks, service marks, or trade names may be registered in the United States or in other jurisdictions, including internationally.

Third-Party Information

This manual contains information such as links to third-party websites that are not under the control of ChikaraDev, and ChikaraDev is not responsible for the content on any linked site. If you access a third-party website mentioned in this manual, then you do so at your own risk. ChikaraDev has provided these links only as a convenience, and the inclusion of the link does not imply that ChikaraDev endorses or accepts any responsibility for the content on those third-party sites.

Copyright © 2008 ChikaraDev. All rights reserved.

The software described in this manual is provided under an agreement with Adobe Systems Incorporated, and such software can only be used in accordance with the terms of the agreement provided by Adobe Systems. Software code described and provided in this manual is provided under an agreement with ChikaraDev. The software can only be used in accordance with the terms of the agreement.

No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, photocopying, manual, optical, recording, or otherwise, outside the license agreement accompanying these materials, without the prior written permission of ChikaraDev. ChikaraDev claims copyright in this program and documentation as an unpublished work, revisions of which were first licensed on the date indicated in the foregoing notice. Claim of copyright does not imply waiver of other rights of ChikaraDev and its subsidiaries.

Information in this manual may change without notice and does not represent a commitment on the part of ChikaraDev.

NOTICE OF LIABILITY

This information in these training materials is distributed on an “AS IS” basis, without warranty of any kind, either express or implied. While every precaution has been taken in the preparation of these materials, neither ChikaraDev nor its licensors shall have any liability to any person or entity with respect to liability, loss, or damage caused or alleged to be caused directly or indirectly by the instructions contained in these materials or by the computer software and hardware products described herein.

First Edition: July 2008 - ChikaraDev - Cupertino, CA 95014 USA

Overview of Auto-Layout in Adobe Flex

The **Canvas** container always uses **absolute positioning**. The **Application** and **Panel** containers use automatic positioning by default, and absolute positioning if you specify the **layout** property as "**absolute**".

When you use auto-layout in Flex, you can allow components and containers to be sized using their default sizes, you can specify explicit sizes, or they can be sized using percentage-based sizing. Usually you will use a combination of both.

Default sizing in Flex depends on the control or container in question. A Button or Label will be sized to fit its text and the padding around it. The height of a TextInput will be sized to fit its text vertically, and the default width is 160 pixels. A TextArea has a default height of 44 pixels and width of 160 pixels.

The default height of a VBox is large enough to hold all its children at their default or explicit height, plus any vertical gap between the children, plus the top and bottom padding of the container. The width is the default or explicit width of the widest child, plus the left and right padding of the container.

Other containers follow similar default sizing rules with minor variations depending on the container.

Using Auto-Layout

Usually you use a combination of the default and percentage sizing of components and containers to achieve your desired layout.

It is important to remember to set the width and/or height of nested containers to 100% if the parent has a percentage based width or height that is less than 100%, as seen in AutoLayoutExample1.mxml.

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
  <mx:VBox height="50%" backgroundColor="0xFF0000" verticalAlign="middle">
    <mx:HBox height="50%" backgroundColor="0x00FF00" verticalAlign="middle">
      <mx:TextArea height="50%"/>
      <mx:TextArea height="50%"/>
    </mx:HBox>
  </mx:VBox>
</mx:Application>
```

The goal of this code was to make the VBox height 50% of the Application container, and the height of the two TextArea 50% the height of the VBox. To achieve this the HBox height should be set to 100%, otherwise the TextArea height is set to 50% of the HBox height, which itself is set to 50% of the VBox height. So if the layout is not sizing as expected, examine your code to see if some nested containers need to be set to 100% width/height.

AutoLayoutExample2.mxml is a more complex example of using auto-layout with states (see sample app in FB).

Component Positioning with Auto-Layout

When you use auto-layout containers, the containers control the positioning of child components. You cannot specify the x and y positions of components inside auto-layout containers.

In the following application (AutoLayoutPositioning1.mxml), explicit x and y positions are ignored, and the VBox and HBox position children vertically and horizontally, separated only by the containers padding and gaps between components.

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
  <mx:VBox width="100%" height="100%" backgroundColor="0xFF0000">
    <mx:Button label="One" x="500" y="100"/>
    <mx:Button label="Two" x="700" y="200"/>
  </mx:VBox>
  <mx:HBox width="100%" height="100%" backgroundColor="0x0000FF">
    <mx:Button label="Three" x="500" y="900"/>
    <mx:Button label="Four" x="700" y="1000"/>
  </mx:HBox>
</mx:Application>
```

You can use auto-layout and absolute containers together to achieve your desired layout, as seen in AutoAbsoluteLayoutPositioning1.mxml.

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute">
  <mx:HBox top="10" horizontalCenter="0">
    <mx:Label text="Click for more information:"/>
    <mx:Button label="More Info..." click="mx.controls.Alert.show('More information here.');" />
  </mx:HBox>
  <mx>List verticalCenter="-200" left="200" width="100" height="270">
    <mx:String>One</mx:String>
    <mx:String>Two</mx:String>
    <mx:String>Three</mx:String>
    <mx:String>Four</mx:String>
    <mx:String>Five</mx:String>
    <mx:String>Six</mx:String>
    <mx:String>Seven</mx:String>
    <mx:String>Eight</mx:String>
    <mx:String>Nine</mx:String>
    <mx:String>Ten</mx:String>
  </mx>List>
  <mx:HBox bottom="10" horizontalCenter="0">
    <mx:Button label="Save"/>
    <mx:Spacer width="30"/>
    <mx:Button label="Close"/>
  </mx:HBox>
</mx:Application>
```