

```

ZoneLayout layout =
    ZoneLayoutFactory.newZoneLayout();
layout.addRow("bfa~a");
layout.addRow("w+*...");
layout.addRow(".....w");
layout.addRow("s<...s");
layout.addRow("xx2y~y", "myTemplate");
layout.addRow("6.....", "myTemplate");
layout.addRow("v*+..v", "anotherTemplate");
JPanel p = new JPanel(layout);
p.add(new JButton("Back"), "b");
layout.insertTemplate("myTemplate");
p.add(new JLabel("Zone X"), "x");

```

Create a template by giving a row definition the template name. Extend a template by using that name again.

Careful when putting templates back to back! They get the same insertion point and order is not maintained.

Bind a component to a zone by adding the component to the panel and then pass in the zone id as the constraint.

Templates are initially removed from the layout. You need to call `insertTemplate(...)` to stick one in.

Zones can span across templates or be wholly contained within them.

Don't worry about extra space! Zonelayout will optimize it away!

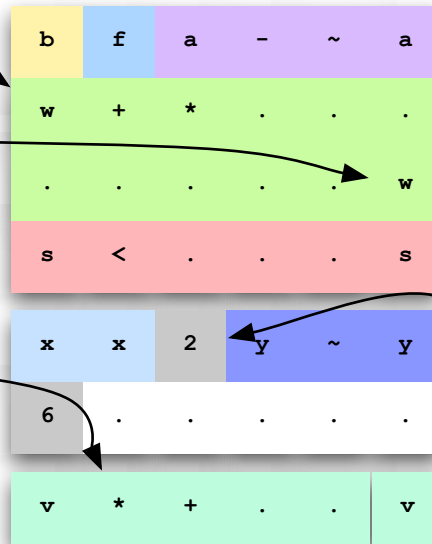
Use Preset spacer components to add space between components.

Create a zone by using a single alphabetical character.

Extend a zone by using the spacer character '.' and then use the zone's alphabetical character to close it.

Always start with the top-left corner of a zone!

Modify a zone by placing a modifier in the boundaries of the zone.



Modifiers

Space

Zone

Component

Default

> Align Right

< Align Left

^ Align Top

_ Align Bottom

- Fill X

| Fill Y

+ Fill

~ Grow X (Take=1)

! Grow Y (Take=1)

* Grow (Take=1)

Resizing

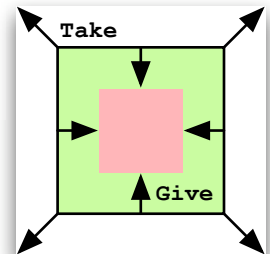
Extra Space is distributed according to Take. Needed space is removed according to Give.

Preferred Size
Minimum Size
Default Give = 1
Default Take = 0

```

layout.getZone("x").setGive(int, int)
layout.getZone("x").setTake(int, int)

```



Presets

Related Items in Toolbar

Related Items

Same Group

