무료 전자 책

배우기

autolayout

Free unaffiliated eBook created from
Stack Overflow contributors.

#autolayout

1
1: 2
2
Examples 2
2
UIImageView 2
2
3
3
6

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: autolayout

It is an unofficial and free autolayout ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official autolayout.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

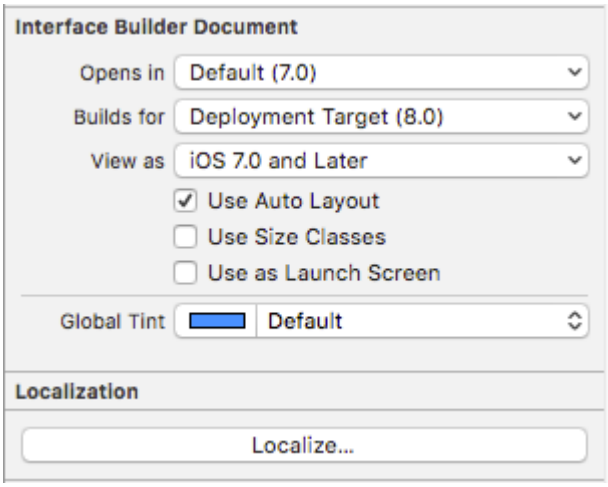
1:

. . .

. . , , .

Examples

. . . . . Xcode .



UIImageView

100 UIImageView UIImageView superview center x constraint center y constraint UIIMageView width, height . . .

```
UIImageView *imageView = [[UIImageView alloc] initWithFrame:CGRectMake(0, 0, 100.0, 100.0);
imageView.self.translatesAutoresizingMaskIntoConstraints = NO;

[imageView addConstraint:[NSLayoutConstraint constraintWithItem:imageView
attribute:NSLayoutAttributeWidth relatedBy:NSLayoutRelationEqual toItem:nil
attribute:NSLayoutAttributeNotAnAttribute multiplier:1.0 constant:100.0]];
[imageView addConstraint:[NSLayoutConstraint constraintWithItem:imageView
attribute:NSLayoutAttributeHeight relatedBy:NSLayoutRelationEqual toItem:nil
attribute:NSLayoutAttributeNotAnAttribute multiplier:1.0 constant:100.0]];
[superview addConstraint:[NSLayoutConstraint constraintWithItem:imageView
attribute:NSLayoutAttributeCenterX relatedBy:NSLayoutRelationEqual toItem:superview
attribute:NSLayoutAttributeCenterX multiplier:1.0 constant:0]];
[superview addConstraint:[NSLayoutConstraint constraintWithItem:imageView
attribute:NSLayoutAttributeCenterX relatedBy:NSLayoutRelationEqual toItem:superview
attribute:NSLayoutAttributeCenterX multiplier:1.0 constant:0]];
```

VFL . . :

- | superview .
- H: V: -
- .
- .
-

.

- @

---

1. `someView` .

```
H:|[someView]|
```

2. `someView` 10 `value` .

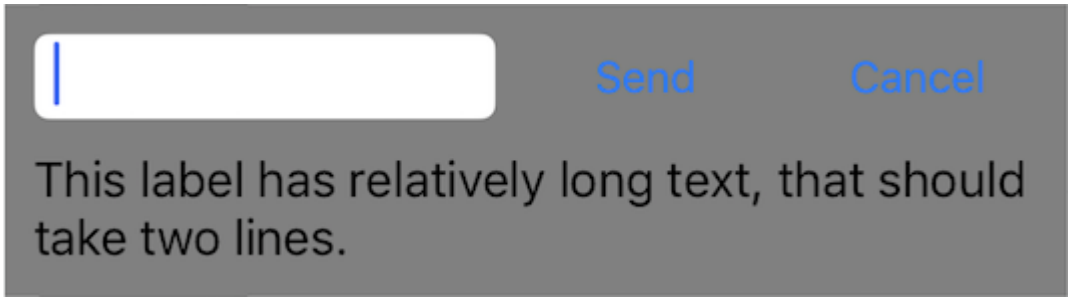
```
V:|-(10)-[someView(value)]
```

3. `someView1` `someView2` - 900 `value1` 800 `value2` :

```
H:[someView1]-(value1@900, value2@800)-[someView2]
```

4. `someView1` `someView2` :

```
V:[someView1(==someView2)]
```



. . .

```
- (void)addView {

    // first lets define a container for our views
    UIView *container = [UIView new];
    // do not forget to disable autoresizing masks for autolayout views
    container.translatesAutoresizingMaskIntoConstraints = NO;
    container.backgroundColor = [UIColor grayColor];

    // now to the subviews. this is mostly boilerplate code:
    UITextField *textField = [UITextField new];
    textField.translatesAutoresizingMaskIntoConstraints = NO;
    textField.borderStyle = UITextBorderStyleRoundedRect;

    UIButton *button1 = [UIButton buttonWithType:UIButtonTypeSystem];
    button1.translatesAutoresizingMaskIntoConstraints = NO;
    [button1 setTitle:@"Send" forState:UIControlStateNormal];

    UIButton *button2 = [UIButton buttonWithType:UIButtonTypeSystem];
    button2.translatesAutoresizingMaskIntoConstraints = NO;
    [button2 setTitle:@"Cancel" forState:UIControlStateNormal];
```

```
    UILabel *label = [UILabel new];
    label.translatesAutoresizingMaskIntoConstraints = NO;
    // this line tells the label to let the text overflow to the next line if needed
    label.numberOfLines = 0;
    label.text = @"This label has relatively long text, that should take two lines.";

    // before adding any constraints the views should be present in the hierarchy
    [container addSubview:textField];
    [container addSubview:button1];
    [container addSubview:button2];
    [container addSubview:label];

    // now lets define two helper dictionaries, one for metrics of our view:
    NSDictionary *metrics = @{@"margin": @10, @"textFieldWidth": @160, @"buttonWidth": @44};
    // and the other for view bindings using a handy macro, which effectively creates a
dictionary with variables of the same name:
    NSDictionary *bindings = NSDictionaryOfVariableBindings(textField, button1, button2,
label);
    // lets define a horizontal format for the first row of views in a variable:
    NSString *horizontalFormat = @"H:|-(margin)-[textField(textFieldWidth)]-(margin)-
[button1(==button2)]-(margin)-[button2]-(margin)-|";
    // this format defines margins of equal size between all views, fixed width for the
textField and sets both buttons to have equal widths
    // lets add these constraints to our container:
    [container addConstraints:[NSLayoutConstraint constraintsWithVisualFormat:horizontalFormat
options:0 metrics:metrics views:bindings]];
    // now lets define horizontal constraints for the second row, where we have the label:
    [container addConstraints:[NSLayoutConstraint constraintsWithVisualFormat:@"H:|-(margin)-
[label]-(margin)-|" options:0 metrics:metrics views:bindings]];
    // another relatively long visual format string:
    NSString *verticalFormat = @"V:|-(margin)-[textField]-(margin)-[label]-(margin)-|";
    // this format string defines vertical constraints for textField and label, and should
also define the height of the container
    // adding these constraints to the container view:
    [container addConstraints:[NSLayoutConstraint constraintsWithVisualFormat:verticalFormat
options:0 metrics:metrics views:bindings]];
    // what we have left are constraints for vertical positions of the buttons
    // lets attach them to the top of the container with a margin:
    [container addConstraints:[NSLayoutConstraint constraintsWithVisualFormat:@"V:|-(margin)-
[button1]" options:0 metrics:metrics views:bindings]];
    [container addConstraints:[NSLayoutConstraint constraintsWithVisualFormat:@"V:|-(margin)-
[button2]" options:0 metrics:metrics views:bindings]];

    // the container is all set up, adding it to the superview:
    [self.view addSubview:container];

    // now lets position our container in its superview
    // you can not use dot notation in the bindings macro, so lets define a temp variable for
the superview:
    UIView *superview = self.view;

    // positioning a view in the center of its superview is not so straightforward
    // we will use a trick from this answer: http://stackoverflow.com/a/14917695/934710
    NSDictionary *containerBindings = NSDictionaryOfVariableBindings(superview, container);
    // width constraint from horizontal format is not part of the trick, but is necessary to
constrain container width
    [self.view addConstraints:[NSLayoutConstraint constraintsWithVisualFormat:@"H:[superview]-
(<=1)-[container(<=superview)]" options:NSLayoutFormatAlignAllCenterY metrics:nil
views:containerBindings]];
    [self.view addConstraints:[NSLayoutConstraint constraintsWithVisualFormat:@"V:[superview]-
```

```
(<=1)-[container]" options:NSLayoutFormatAlignAllCenterX metrics:nil
views:containerBindings]];

}
```

: https://riptutorial.com/ko/autolayout/topic/6858/--

S. No		Contributors
1		Community, Mahesh Agrawal, pckill