EBook Gratuito

APPRENDIMENTO autolayout

Free unaffiliated eBook created from
Stack Overflow contributors.

#autolayout

Sommario

Di

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

Capitolo 1: Iniziare con l'autolayout

Osservazioni

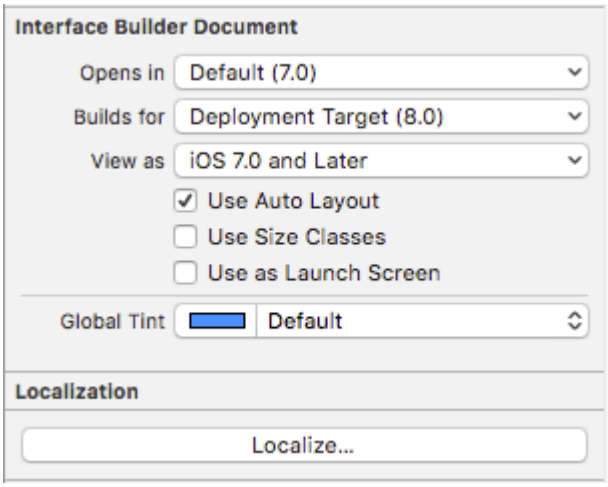
Layout automatico calcola dinamicamente la dimensione e la posizione di tutte le viste nella gerarchia della vista, in base ai vincoli posti su tali viste, da Apple. Il layout automatico può anche essere compreso descrivendolo come una serie di regole che si utilizzano per ciascuna delle viste nella gerarchia della vista, la quale regola ha definito come l'allineamento e il restringimento o l'estensione della vista verranno eseguiti in base alle diverse dimensioni dello schermo dei dispositivi.

Dovremmo utilizzare il layout automatico in tutti i nostri progetti perché questo ci aiuta a gestire il design degli schermi per tutti i dispositivi disponibili che abbiamo come target. Per questo abbiamo solo bisogno di capire cos'è e come funziona e dopo aver capito come funziona, degnare il layout dinamico per più dispositivi sarà divertente.

Examples

Installazione o configurazione

Per utilizzare il layout automatico, è necessario abilitare un flag booleano nello storyboard. È mostrato nell'immagine qui sotto. Dopo questo possiamo usare il layout automatico nel nostro storyboard. C'è un'altra opzione se vogliamo usare le classi di dimensioni o meno. La classe di dimensioni è anche un'opzione utile nel layout automatico che ci aiuta a progettare in modo diverso le diverse dimensioni dello schermo del dispositivo con elementi uguali o diversi. L'immagine per l'installazione va qui. È dal pannello di ispezione dei file dell'editor delle proprietà in Xcode.



Aggiunta di UIImageView al centro dello schermo

Se si desidera aggiungere UIImageView al centro dello schermo con larghezza e altezza di 100 pixel, è necessario impostare il vincolo del centro x e il vincolo del centro y sulla superview da

UIImageView e il vincolo di larghezza e altezza a UIIMageView. Ecco il codice. C'è un modo per farlo nello storyboard ma ho scritto nel codice perché è comprensibile per questa situazione nella documentazione.

```
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]];
```

Utilizzo del linguaggio in formato visivo

C'è un modo per semplificare la definizione del rollload automatico per le viste usando VFL. All'inizio può sembrare difficile, ma in realtà è davvero facile da usare. Prima alcune definizioni:

- `|` rappresenta superview
- `H:` o `V:` rappresenta l'orientamento corrente - orizzontale o verticale
- i nomi delle viste dovrebbero essere racchiusi tra parentesi quadre
- l'altezza e la larghezza della vista devono essere racchiuse tra parentesi
- i margini vengono specificati tra le viste e circondati da trattini
- la priorità per un margine o una dimensione di visualizzazione può essere specificata con @

Esempi di sintassi del formato visivo

1. `someView` è collegato ai bordi sinistro e destro della sua superview senza margini:

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

2. `someView` è attaccato in alto con un margine di 10 punti e ha un'altezza pari al `value` :

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

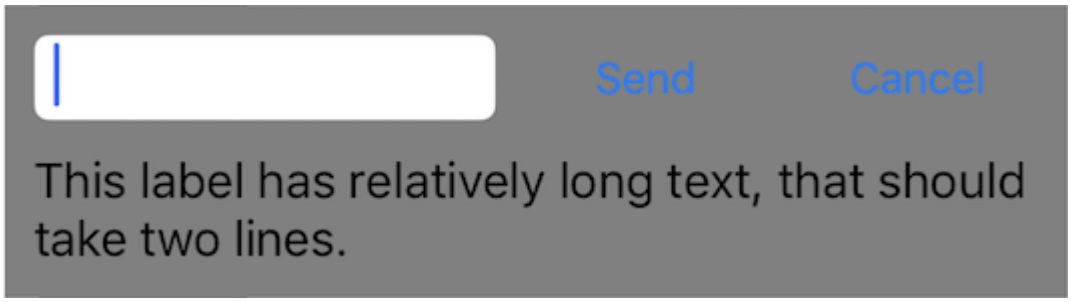
3. `someView1` e `someView2` hanno due margini definiti tra loro - `value1` con priorità di 900 e `value2` con priorità di 800:

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

4. `someView1` height è uguale all'altezza di `someView2` :

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

Esempio di codice



Definiamo una nuova vista, che ha un campo di testo e due pulsanti con altezze uguali con margini tra di loro e un'etichetta sotto. I pulsanti dovrebbero avere larghezza uguale e l'etichetta dovrebbe traboccare alla riga successiva se il contenuto è abbastanza lungo. Questa vista dovrebbe essere dimensionata automaticamente in base al suo contenuto sull'asse orizzontale e verticale e centrata nel mezzo della sua superview.

```
- (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]];

}
```

Leggi Iniziare con l'autolayout online: https://riptutorial.com/it/autolayout/topic/6858/iniziare-con-l-autolayout

Titoli di coda

S. No	Capitoli	Contributors
1	Iniziare con l'autolayout	Community, Mahesh Agrawal, pckill