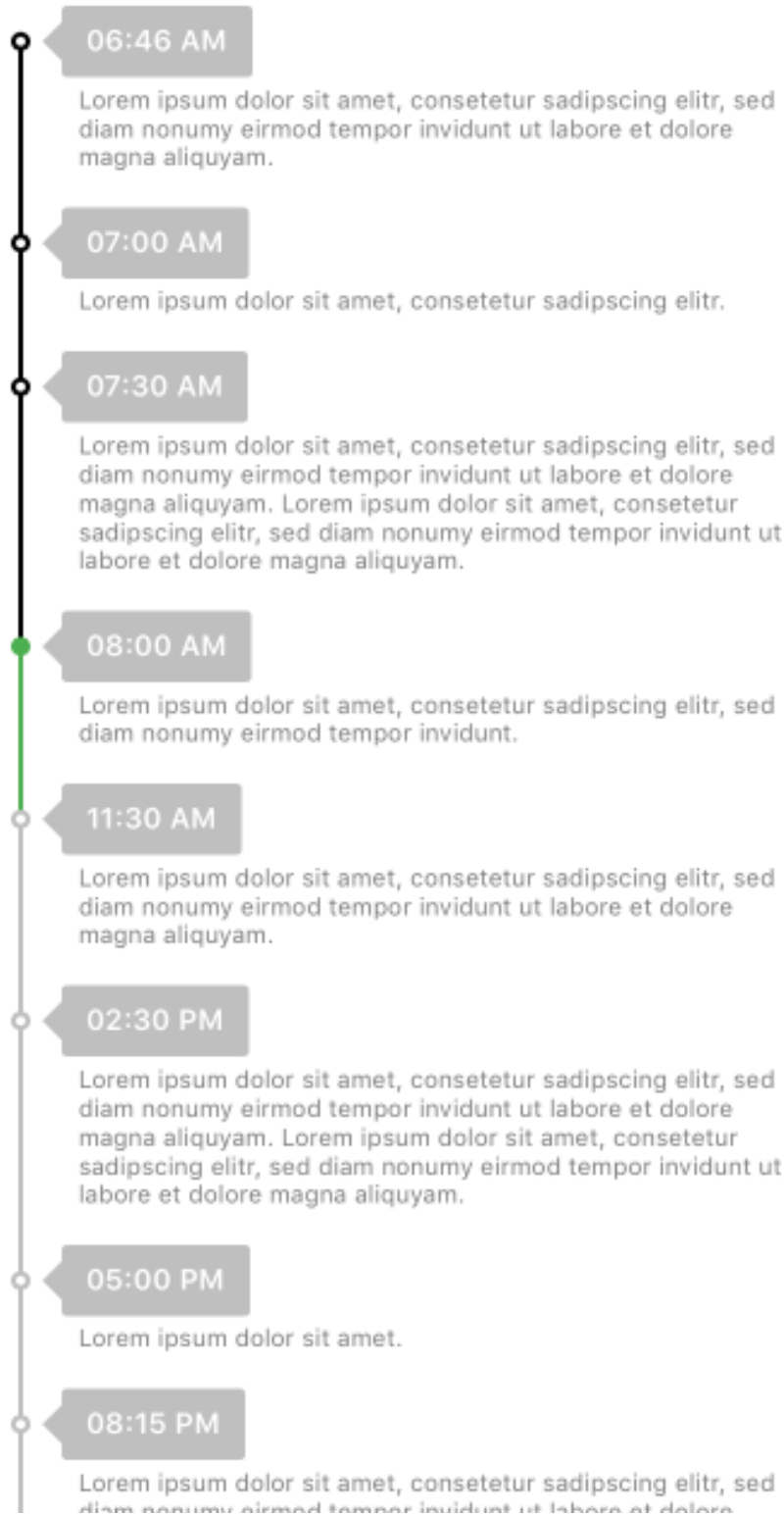

ISTimeline

ISTimeline is a simple timeline view written in Swift 3

license Apache-2.0 license Apache-2.0 Carthage compatible license Apache-2.0



Requirements

- iOS 8.0 or higher
- Swift 3

Installation

CocoaPods

ISTimeline is available through CocoaPods. To install it, simply add the following line to your Podfile:

```
1 pod 'ISTimeline'
```

Carthage

ISTimeline is also available through Carthage. Include the following line into your Cartfile and follow the instructions under getting started:

```
1 github "instant-solutions/ISTimeline"
```

Manually

Just drop the files ISPoint.swift and ISTimeline.swift into your project.

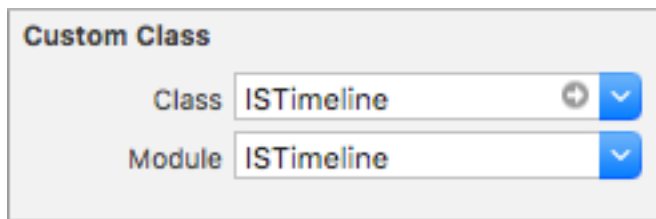
Usage

Import

```
1 import ISTimeline
```

Integration

We recommend to use the timeline view in your storyboard. Just add a plain view and set the custom class and the module property to `ISTimeline`.



Or add the view programmatically:

```
1 let frame = CGRect(x: 0.0, y: 20.0, width: 300.0, height: 400.0)
2
3 let timeline = ISTimeline(frame: frame)
4 timeline.backgroundColor = .white
5
6 self.view.addSubview(timeline)
```

ISPoint

Each bubble is represented by an ISPoint object in the points array. ISPoints has several properties:

`var title:String` shown in the bubble

`var description:String?` shown below the bubble

`var pointColor:UIColor` the color of each point in the line

`var lineColor:UIColor` the color of the line after a point

`var touchUpInside:Optional<(_ point:ISPoint)-> Void>` a callback, which is triggered after a touch inside a bubble

`var fill:Bool` fills the point in the line (default: **false**)

Example point:

```
1 let point = ISPoint(title: "my title")
2 point.description = "my awesome description"
3 point.lineColor = .red
4 point.fill = true
```

Initializers The designated initializer is:

```
1 ISPoint(title:String, description:String, pointColor:UIColor, lineColor:
    UIColor, touchUpInside:Optional<(_ point:ISPoint) -> Void>, fill:
    Bool)
```

You also can use one the convenience initializers:

```
1 ISPoint(title:String, description:String, touchUpInside:Optional<(_
    point:ISPoint) -> Void>)
```

```
1 ISPoint(title:String, touchUpInside:Optional<(_ point:ISPoint) -> Void
  >)
```

Or even this one:

```
1 ISPoint(title:String)
```

Touch events To get touch events you just have to set a callback closure to the property `point.touchUpInside`. It is triggered after a touch up inside a bubble.

```
1 point.touchUpInside =
2   { (point:ISPoint) in
3     // do something
4   }
```

Add points to the timeline

To add points to the timeline you can simple create and set your points array to the property `timeline.points` or you can append each point one after the other.

```
1 let myPoints = [
2   ISPoint(title: "first"),
3   ISPoint(title: "second"),
4   ISPoint(title: "third")
5 ]
6 timeline.points = myPoints
```

Append a single point:

```
1 timeline.points.append(ISPoint(title: "fourth"))
```

Colors

You can customize the following timeline colors:

`var bubbleColor:UIColor` color of every bubble (default `.init(red: 0.75, green: 0.75, blue: 0.75, alpha: 1.0)`)

`var titleColor:UIColor` color of the title in the bubble (default `.white`)

`var descriptionColor:UIColor` color the description text (default `.gray`)

Points can be colored as described above.

Line width and point radius

Some common parameters can be adjusted:

`var pointDiameter:CGFloat` diameter of a point in the line (default 6.0)
`var lineWidth:CGFloat` the thickness of the line (default 2.0)
`var bubbleRadius:CGFloat` the radius of the bubble corners (default 2.0)

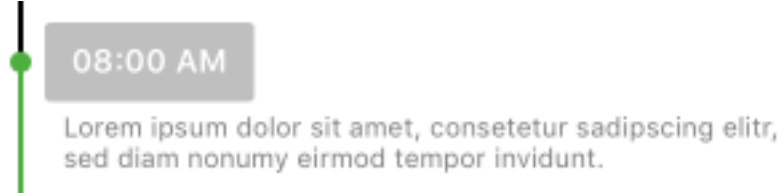
Bubble arrows

With the flag `var bubbleArrows:Bool` it is possible to remove all arrows (default **true**).

With arrows:



And without arrows:



Intents

You can add some padding by setting the content insets. This currently can only be done programmatically.

```
1 timeline.contentInset = UIEdgeInsetsMake(20.0, 20.0, 20.0, 20.0)
```

Clip subviews

Per default, the timeline clips all subviews to its bounds. If you would like to change this behavior just set it to **false**.

```
1 timeline.clipsToBounds = false
```

Working example

```
1 let frame = CGRect(x: 0.0, y: 20.0, width: 300.0, height: 400.0)
2 let timeline = ISTimeline(frame: frame)
3 timeline.backgroundColor = .white
4 timeline.bubbleColor = .init(red: 0.95, green: 0.95, blue: 0.95, alpha:
    1.0)
5 timeline.titleColor = .black
6 timeline.descriptionColor = .darkText
7 timeline.pointDiameter = 7.0
8 timeline.lineWidth = 2.0
9 timeline.bubbleRadius = 0.0
10
11 self.view.addSubview(timeline)
12
13 for i in 0...9 {
14     let point = ISPoint(title: "point \(i)")
15     point.description = "my awesome description"
16     point.lineColor = i % 2 == 0 ? .red : .green
17     point.pointColor = point.lineColor
18     point.touchUpInside =
19         { (point:ISPoint) in
20             print(point.title)
21         }
22
23     timeline.points.append(point)
24 }
```

Demo

ISTimelineDemo is a simple demo app which shows the usage of ISTimeline in a storyboard.

TODOs

- ☐ show images in the timeline
- ☐ animate the appending and removing of an entry

License

ISTimeline is licensed under the terms of the Apache License 2.0. See the LICENSE file for more info.

Contribution

Feel free to fork the project and send us a pull-request! :sunglasses:

Or consider sponsoring us so we can continue to work on this project: GitHub Sponsors :star_struck:

Author

Made with :heart: in Austria by instant:solutions OG