Asynchrony Solutions Blog

- Asolutions.com
- Subscribe

- Newer Blog Post »
- « Older Blog Post

- **Development**

Building Custom Map Annotation Callouts – Part 2

by James Rantanen on September 3rd, 2010 at 8:38 am

Part 1 showed how to build a **custom map** callout that provides more content flexibility than the native callout, but maintains the expected look and behavior. In **part** 2 we will add a very common element of the **map** interface into our **custom** callout – the accessory button. At first glance this seems simple: just add a button to the callout. However, MapKit intercepts touch events and causes undesired callout behavior. The code used to add an accessory button is also applicable to any other button(s) or responders you may want to add to a callout, giving you the flexibility to do what you feel is best for your users.

Add the Button

We will begin by adding the button as we normally would, to see this behavior in action. This will be done by creating a subclass of the **custom** callout from **part** 1. Notice the attempt to call the standard callback for an accessory tap in `calloutAccessoryTapped`.

```
@implementation AccessorizedCalloutMapAnnotationView

@synthesize accessory = _accessory;

- (id) initWithAnnotation:(id <mkannotation>)annotation reuseIdentifier:(NSString *)reuseIdentifier {
        if (self = [super initWithAnnotation:annotation reuseIdentifier:reuseIdentifier]) {
                self.accessory = [UIButton buttonWithType:UIButtonTypeDetailDisclosure];
                self.accessory.exclusiveTouch = YES;
                self.accessory.enabled = YES;
                [self.accessory addTarget: self
                                                   action: @selector(calloutAccessoryTapped)
                                 forControlEvents: UIControlEventTouchUpInside | UIControlEventTouchCancel];
                [self addSubview:self.accessory];
        }
        return self;
}

- (void)prepareContentFrame {
        CGRect contentFrame = CGRectMake(       self.bounds.origin.x + 10,
                                                                                self.bounds.origin.y + 3,
                                                                                self.bounds.size.width - 20,
                                                                                self.contentHeight);
        self.contentView.frame = contentFrame;
}

- (void)prepareAccessoryFrame {
        self.accessory.frame = CGRectMake(self.bounds.size.width - self.accessory.frame.size.width - 15,
                                                                          (self.contentHeight + 3 - self.accessory.frame.size.height) / 2,
                                                                          self.accessory.frame.size.width,
                                                                          self.accessory.frame.size.height);
}

- (void)didMoveToSuperview {
        [super didMoveToSuperview];
        [self prepareAccessoryFrame];
}

- (void) calloutAccessoryTapped {
        if ([self.mapView.delegate respondsToSelector:@selector(mapView:annotationView:calloutAccessoryControlTapped:)]) {
                [self.mapView.delegate mapView:self.mapView
                                                annotationView:self.parentAnnotationView
                 calloutAccessoryControlTapped:self.accessory];
        }
}

@end
```

We will also implement that callback in the **map** view delegate. Normally a new view would be pushed on to the navigation stack at this point, but for this example it will be simpler to just display an alert.

```
- (void)mapView:(MKMapView *)mapView
 annotationView:(MKAnnotationView *)view
calloutAccessoryControlTapped:(UIControl *)control {
        UIAlertView * alert = [[[UIAlertView alloc] initWithTitle:@"Asynchrony Solutions"
                                                                                                          message:@"Callout Accessory Tapped"
                                                                                                         delegate:nil
                                                                                        cancelButtonTitle:@"OK"
                                                                                        otherButtonTitles:nil] autorelease];
        [alert show];
}
```

Prevent Deselection of the Parent Annotation

If the button is tapped now, the alert will be displayed, but the callout is removed because the touch event also caused the parent annotation to be deselected just as if the button were not there. To solve this problem, we will have to disable selection changes on the parent annotation and make a small change to `mapView:didDeselectAnnotationView:` in the mapView delegate.

First off, we need to subclass MKPinAnnotationView (or MKAnnotationView if using a **custom** annotation) to add a `preventSelectionChange` property and override `setSelected:animated:`.

```
@interface BasicMapAnnotationView : MKPinAnnotationView {
        BOOL _preventSelectionChange;
}

@property (nonatomic) BOOL preventSelectionChange;

@end

@implementation BasicMapAnnotationView

@synthesize preventSelectionChange = _preventSelectionChange;

- (void)setSelected:(BOOL)selected animated:(BOOL)animated {
        if (!self.preventSelectionChange) {
                [super setSelected:selected animated: animated];
        }
}

@end
```

When the button is tapped, the callout needs to set the new `preventSelectionChange` property to `YES` and set it back to `NO` a short time later (1 second seems to be a good delay for this call). This needs to be done before the typical touch event callbacks are invoked so we will override `hitTest:withEvent:`. Also, The mapView keeps track of which **annotations** are selected, so when selection changes on the parent are re-enabled, the **map** view needs to be forced to select the annotation again.

```
- (UIView *)hitTest:(CGPoint)point withEvent:(UIEvent *)event {

        UIView *hitView = [super hitTest:point withEvent:event];

        if (hitView == self.accessory) {
                [self preventParentSelectionChange];
                [self performSelector:@selector(allowParentSelectionChange) withObject:nil afterDelay:1.0];
        }

        return hitView;
}

- (void) preventParentSelectionChange {
        BasicMapAnnotationView *parentView = (BasicMapAnnotationView *)self.parentAnnotationView;
        parentView.preventSelectionChange = YES;
}

- (void) allowParentSelectionChange {
        [self.mapView selectAnnotation:self.parentAnnotationView.annotation animated:NO];

        BasicMapAnnotationView *parentView = (BasicMapAnnotationView *)self.parentAnnotationView;
        parentView.preventSelectionChange = NO;
}
```

Even though the selection change is disabled on the parent annotation view, the **map** view will still invoke the delegate method `mapView:didDeselectAnnotationView:`. Add an additional condition to the if-statement to prevent removal when the annotation view is not allowing selection changes.

```
- (void)mapView:(MKMapView *)mapView didDeselectAnnotationView:(MKAnnotationView *)view {
        if (self.calloutAnnotation &&
                view.annotation == self.customAnnotation &&
                !((BasicMapAnnotationView *)view).preventSelectionChange) {
                [self.mapView removeAnnotation: self.calloutAnnotation];
        }
}
```

Prevent Selection of Other Annotations

With the above code, the callout now behaves as expected in most situations; however, if another annotation happens to be under the button, it will be selected. The simplest way to solve this is to disable all the annotation views on the **map** except the **custom** callout and the parent annotation. We can find all of the other annotation views by getting the subviews of the superview of the callout, and checking that they inherit from `MKAnnotationView`. Also, they must be re-enabled a short time later (again, a 1 second delay works well).

```
- (UIView *)hitTest:(CGPoint)point withEvent:(UIEvent *)event {
        UIView *hitView = [super hitTest:point withEvent:event];

        if (hitView == self.accessory) {
                [self preventParentSelectionChange];
                [self performSelector:@selector(allowParentSelectionChange) withObject:nil afterDelay:1.0];
                for (UIView *sibling in self.superview.subviews) {
                        if ([sibling isKindOfClass:[MKAnnotationView class]] && sibling != self.parentAnnotationView) {
                                ((MKAnnotationView *)sibling).enabled = NO;
                                [self performSelector:@selector(enableSibling:) withObject:sibling afterDelay:1.0];
                        }
                }
        }

        return hitView;
}

- (void) enableSibling:(UIView *)sibling {
        ((MKAnnotationView *)sibling).enabled = YES;
}
```

Conclusion

Now the **Custom Map** Callout is complete. Using the code and concepts presented in this post and **Part** 1, you have the tools to build **callouts** that fit your needs. With minor adjustments to this code, you can add multiple buttons, implement a callout with adjustable width, or change the look of the callout to match your application's style.

You may download the full source code to see a working example.

James Rantanen

More Posts - Website

Tags: callout, iOS, iPhone, mapkit, mkannotation, mkannotationview, mobile

16 comments

1.

   To get this to work on pre-4.0 devices, just put a check in the hitTest method: if this is a pre-4.0 device, then after [self preventParentSelectChange];
   invoke
   [self calloutAccessoryTapped];

   Thanks again for this great code.

   Mike on November 8, 2010

2.

Hi

I'm a beginner in Phone dev.
looking at your screenshot, I see that the standard title/subtitle has been replaced by a more enriched object set (an image+button).
what I would like to do in my app is displaying in the callout bubble

- a title
- below the title, the results of a webservice, basically a list such as :
image 1 – name1
image 2 – name2
image 3 – name3
where the callout bubble could fit this content,
+ a button like yours.

is it possible ? how long do you think it would take to be coded ?

thanks for your help that will be very precious to me…
regards
Dan.

Dan on November 25, 2010

3\.

Has anybody managed to deploy this to OS 3.1.3. I'm running XCode 3.2.4 on SDK 4.1, so I've set my Base SDKs to iOS Device 4.1 and the iOS Deployment Target to iOS 3.1.3, but the **custom** callout with the image never shows. The other **callouts** work perfectly. I've tried the comment for the pre-4.0 devices but still doesnt work.

Jean on November 26, 2010

4\.

Jean, to get the callout to show up on pre-4.0 devices you will need to call back to a method your **map** view controller from within setSelected:animated: on the **map** annotation.

James Rantanen on November 29, 2010

5\.

Dan, to add content to the **custom map** callout it is as simple as setting the height required and adding views to the **custom** callout. In this example I just added an image, but it would be simple to add a few UILabels instead.

James Rantanen on November 29, 2010

6\.

Is it possible to have phone number recognition in text in the callout view?
So the user can call that number or do I have to use this solution with a button?

Mikael Bartlett on December 21, 2010

7\.

Hi,

happy new year @all.
Works great now for me with multiple **annotations**. On 3.1 it also works, but what could be the problem that the callout doesn't center on the **map** / the **map** doesn't center the selected annotation??

Greetings

Alex on January 5, 2011

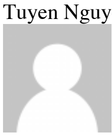
8. [...] **Part** 2 covers adding a button to the **custom** callout, which is not as simple as it sounds. [...]

Building **Custom Map** Annotation **Callouts** – **Part** 1 | Advanced App Development on March 25, 2011

9.

Thank you for your tutorial.
Great work!

Tuyen Nguyen on March 25, 2011

10. 

Hey, great tutorial!
How can i avoid replacing current location (blue bubble) with standard red pin ?

fry1 on April 4, 2011

11.

//found soultion for it ^^ if someone is interested
if (annotation == self.mapView.userLocation){
return nil;
}

fry1 on April 4, 2011

12.

Thanks you very much. This was exactly what I needed. It saved me for a ton of work.

Martin on April 5, 2011

13.

Great work! Thanks for sharing it. Saved me a lot of time and effort!

Matt on July 15, 2011

14.

Hi! Very cool tutorial, thank you for your work. I found what i needed!

azickh on July 31, 2011

15.

Anyone know how to get the **annotations** title to appear inside the callout? I created the UILabel, but cannot get the title information to work.

Adam S on September 9, 2011

16.

Hi,

Thanks for the tutorial. How do I get the accessory button for the text display callout instead of the picture display?

Thanks,
Venkat

Venkat on November 11, 2011

This Post is Open for Discussion

Name

Mail (never published)

Website

Your Comment

Submit Comment

September 2010

M	T	W	T	F	S	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			
« Aug				Oct »		

Categories

- Agile
- Asynchrony
- Change
- Design
- Development
- Visual Thinking

Archives

- April 2012
- February 2012
- November 2011
- September 2011
- August 2011
- July 2011
- June 2011
- May 2011
- April 2011
- March 2011
- February 2011
- January 2011
- December 2010
- November 2010
- October 2010
- September 2010
- August 2010
- July 2010
- June 2010
- May 2010

Tags

acronyms agile android Apple buckminster callout change changeviz CMMI comics CRUD cucumber design dev eclipse Enterprise Architecture interface iOS iPad iPhone Kanban liferay mapkit metrics mkannotation mkannotationview mobile pairing photoshop project management rcp retrospectives risk management Security selenium Test-driven development testing tutorial UCD ui urban design usability ux video vizthink

Interesting Links

- Rogue Project Leader

□ **Asynchrony Twitter Feed**

- asynchrony: Debuting on Twitter: Our new logo! We'll be rolling the new look out to the website soon! Kudos to our designers for their hard work. April 27, 2012
- asynchrony: Tonight is @STLMobileDev! @MarkSawyier on his startup, @BonfyreApp; @markshiz on cross-platform dev with Android NDK. Talks start at 6:30p! March 22, 2012

- asynchrony: RT @railsgrammer: Kanban: It changed my life. A new post at @asynchrony 's blog http://t.co/cHTryvNi February 9, 2012
- asynchrony: RT @StlMobileDev: Thu Jan 26th: Doors: 6p Talks: 6:30p. @jessitron on Android storage; Srdjan Grubor on a factory dashboard for Android. ... January 24, 2012
- asynchrony: RT @SandiStJohn: Asynchrony is looking to hire java/andriod developers for perm opportunities. Call me today for more details 314-678- ... January 11, 2012

Come visit us sometime at:
Our Main Office **1701 Washington Avenue**
Saint Louis, Missouri 63103
314 / 678-2200
Our Scott Air Force Base Office **703 Seibert Road, Suite 3**
Scott AFB, IL 62225
618 / 212-4580

Check out our parent site: **asolutions.com**

Asynchrony Blog Disclaimer