Qt for iOS locks up when app is launched face up. (qiosscreen.mm assertion)

Asked 9 years, 2 months ago Modified 9 years, 2 months ago Viewed 140 times

Part of Mobile Development Collective

0

I'm migrating a Qt for iOS project to Qt 5.5. In iOS 5.1.1 at least, the app fails to start if you launch it with the device face up. An assertion displays an error in qiosscreen.mm at line 344. Here's the Qt source code function that is failing:

```
Qt::ScreenOrientation QIOSScreen::orientation() const
{
    // Auxiliary screens are always the same orientation as their primary
orientation
    if (m_uiScreen != [UIScreen mainScreen])
        return Qt::PrimaryOrientation;

    UIDeviceOrientation deviceOrientation = [UIDevice
currentDevice].orientation;

    // At startup, iOS will report an unknown orientation for the device, even
    // if we've asked it to begin generating device orientation notifications.
    // In this case we fall back to the status bar orientation, which reflects
    // the orientation the application was started up in (which may not match
    // the physical orientation of the device, but typically does unless the
    // application has been locked to a subset of the available orientations).
    if (deviceOrientation == UIDeviceOrientationUnknown)
        deviceOrientation = UIDeviceOrientation([UIApplication
sharedApplication].statusBarOrientation);

    // If the device reports face up or face down orientations, we can't map
    // them to Qt orientations, so we pretend we're in the same orientation
    // as before.
    if (deviceOrientation == UIDeviceOrientationFaceUp || deviceOrientation ==
UIDeviceOrientationFaceDown) {
        Q_ASSERT(screen());
        return screen()->orientation();
    }

    return toQtScreenOrientation(deviceOrientation);
}
```

It displays an assertion at

```
Q_ASSERT(screen());
```

screen() must be returning 0, so screen()->orientation() is attempting to deference a null pointer. That screen() function is defined in the parent class, QPlatformScreen:

```
QScreen *QPlatformScreen::screen() const
{
    Q_D(const QPlatformScreen);
    return d->screen;
}
```

The constructor for that class initializes d->screen to 0:

```
QPlatformScreen::QPlatformScreen()
    : d_ptr(new QPlatformScreenPrivate)
{
    Q_D(QPlatformScreen);
    d->screen = 0;
}
```

From the comments, I infer that d->screen is set at some point when the orientation is portrait or landscape, and then they fall back to that when it becomes face up / down. Since it is starting as face up, there is no prior good value to fall back to.

Has anyone else encountered this, or have a solution? Btw, I am not building from qt source and do not want to, so changing their code is not a solution for me if I can possibly avoid that.

MD c++ ios qt

Share Improve this question Follow

asked Sep 24, 2015 at 14:09
BuvinJ
11k ● 5 ● 94 ● 106

2 This looks like it may be a bug, if so it should be reported to bugreports.qt.io – sjdowling Sep 24, 2015 at 14:21

1 "I am not building from qt source and do not want to" For any sort of a professional deployment, you should be building Qt from source. You probably wish to have control over what goes in to keep the binaries small. There's probably plenty of features you're not using, that can be turned off. – Kuba hasn't forgotten Monica Sep 24, 2015 at 14:27

I wish that if people are going to down vote a question, they would state why... What is possibly the matter with this question? – BuvinJ Sep 24, 2015 at 16:26

I don't build from source because I want Qt to remain a black box which I don't have to worry about the inner workings of it, or maintaining as Qt evolves and changes are required for new versions of the native platforms. That is pretty basic programming engineering theory. I am looking at the source here though because I am showing the bug that I want to work around somehow rather than directly modifying. – BuvinJ Sep 28, 2015 at 16:25

Is there a way I could derive a class from QIOSScreen, overwrite just this function, and then - the part I'm really missing - implement that derived class in place of this? – BuvinJ Sep 28, 2015 at 16:27

1 Answer

Sorted by: Highest score (default)

0

✓

I found that this only seems to be occurring when I launch the app from XCode or QtCreator. If I launch the app on the device the way it normally runs, this bug seems to be averted.

Share Improve this answer Follow

answered Oct 1, 2015 at 15:36

BuvinJ

11k ● 5 ● 94 ● 106