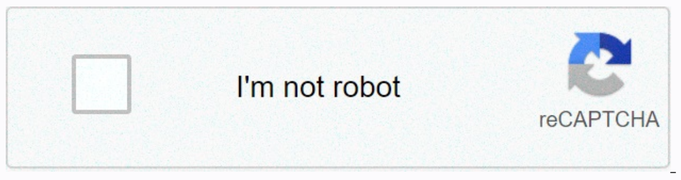

Continue

Cineplex mobile app not compatible

January 7, 2020 3 minutes reader You are reading Indian Entrepreneur, an international franchise of Media Entrepreneurs. For any and every need today, one can find many mobile apps out there. There are individuals and studios coming up with novel ideas to stay ahead in a field that is already super competitive. As a programmer, if mobile app development is what you feel is your call, here's a detailed guide on what you have to do to create a successful career off it. Create a mobile app The first step is to find out the purpose of the app. With so much out there, it's important that your application stands out, even if it solves a problem that already has a solution. For example, you might want to create a new photo editing app. There will be a lot of competition no matter how well done it is. Go back to the drawing board if your idea doesn't add any value to what's already available. After you've figured out what the app will do, the next step is to choose a platform for your app, that is, whether you want it to work on Android or iOS. Other operating systems exist but these are the two most widely used operating systems available today. The programming language for iOS is Objective-C and C, while for Android it's Java.Step By StepBrowse through the respective app stores and a deeper understanding of what you want to solve. Research is an important step. Create a list of all the functions your app will have. Going back to the photo editing example, you might want an option to crop, one to improve brightness, and a few preset filters. Design how each app functions and how end users will interactcoding your appTest app and get a few options to use it. Get feedback and improve until the final product works well. This step is necessary to eliminate errors if any. Upload to the platform it was designed for. FinanceA time-consuming application but if you are already a developer with the know-how of the necessary programming languages, you do not cost much to create a basic language. To get your apps listed on the Google store, you only need to pay a one-time fee of $25 while for the Apple store you pay $100 a year. Part of the reason why Apple charges more is because it is not quite easy to get your app on their store. If you build an app that doesn't need the internet for any of its functions to work, it means you don't really need to rent storage space in the cloud. Otherwise, you may need to pay a little for the cloud, which can be up to as little as $5 a month. One of the most common ways to monetly monetly from an app is in-app ads. Another way to make money is to make in-app purchases. You can keep some functions free and once you have a middle user see if a few exclusive paid features can make them pay. Subscriptions or fully paid apps are also popular, but you need to really offer something different to get people to buy it. Leading Companies Mobile Apps Use this guide to read reviews and customers of Hungary's leading mobile app developers. Preview past companies that work and connect with the best mobile app development companies in Hungary. Recent Posts EnthusiasticAlly Developed, Film Buff, Regular Runner... Email: csl.supernova@gmail.comT I used to work as a web app developer in my previous life and I started working as a mobile app developer about two years ago. I learned swift from an AppCoda book and created my first iOS app in two months. Then I spent four months building a library of reusable modules - or frameworks in iOS conditions - to prepare my second iOS app. At this point, you might ask: Why do I have to spend four months on just one framework - which isn't delivery itself - after I've been able to create my first app in two months? And some of you might ask: If I can go back in time, can I create a framework before my first app? My answer to both questions is: I would probably do the same if I could make my choice again. And I'll tell you why in this article. Starting rocky StartRight from the start, I knew my first app wouldn't be perfect no matter how much time I spent on it. So I just wanted to create something simple - but good enough - that I could build on top of later. I have learned from past experience the difficulties of adding platform support at the end of the software development cycle. So the first request I made when designing the app was to support multi-platform on iOS and Andriod.I started researching what options were available at the time. I met React Native, which seems to be a good choice to provide multi-platform support. But unfortunately, after spending a large amount of time, React Native didn't work out in my case. I'm back to square one. At that time, I spent more than a month just researching, without writing a single line of code for my application. I was disappointed and I wondered: Should I go back to try another option on multi-platform support? Or should I focus on a platform for my first app? I struggled with myself for a few hours, then I made an important decision for my first app - it will only support iOS. About an hour later, I was able to build a HelloWorld app with Swift and run it on my iPhone. I was relieved - apart from words - to finally create a runnable app after all that time devoted to research as well as bootstrapping. And I'm ready to start encrypting my first app. Finding the shortest possible pathHaving decided to focus on iOS, my direction became much clearer. What I need is to create an iOS app and send it to Apple App no need to worry about the Google Play Store for Android apps. After reading online, I understand that submitting the App Store may not always be as simple as expected. The review process can sometimes take a while. So I made another important decision for my first app – it would be the easiest one I could create. This app will serve as my app tester for the App Store before my real app. Because I want to go through the whole submission process from start to finish as soon as possible. The last thing I want to end is, spend a year creating really feature-rich apps and then being rejected from the App Store.Thus I spent about two months building a coffee diary app based on app tutorials from the AppCoda book. Then my first app was made and ready for app store.Stop briefly at DestinationThe App Store process is a time-consuming process for me – or any first timeder. Because it involves a lot of ing development work. Not only do I have to fill in all the necessary app information on the site, but I also have to create app icons and screenshots of different sizes. Learning how to meet metadata requirements before my application can be approved also takes some time. No three kingdoms. It took me almost three weeks to go through the entire application process. I expect my second app to be less because now I know what to expect and what tools I can use in every step of the process. Anyway, after the first app was launched in the App Store, it was time to start my second app. Preparing for The Second TripScing my first app based on the book guide is a good starting point. Because I was able to reuse two essential modules- the UI layout and data storage. But for obvious reasons, the tutorial application did not provide other important modules such as logs and testing. I used simple print reports to debugging and do all the tests manually. It quickly turned all my exciting development tasks into tedious quality assurance tasks. I told myself I wouldn't repeat that in the second app. So before starting on my second application, I knew I had to build a framework of reusable modules. Going from a long wish list, I narrowed it down to four basic modules: Automatic Checking, Logs, UI Layout, and Data Storage. Then I started my mission to build each of the four modules. Gearing Up to Go AgainT I expected to spend about two months building the framework - but it turned out to be four months. However, I did not regret any minute of it. It's one thing to directly implement a feature in an app. It's a completely different learning experience to create a reusable module for the same feature. For example, in handling the layout user interface, the iOS platform provided the MVC architecture. There are others - like MVVM or VIPER- some developers have used because of some shortcomings (they believe) in MVC. After some long research, I believe mvc is good enough for what I need in my application. So I just created some MVC help methods in the layout module my user interface. On the other side, iOS has integrated classes - XCTest and XCUITest - for automatic testing. But there are common frameworks from Google and Facebook that can do more - like Snapshot Testing. So instead of the helper method, I created proxies to wrap around built-in iOS classes in my test module. So I can easily convert iOS integrated classes for other popular frameworks later on. No need to worry about affecting my application built on the test module. However, with the entire framework completed last month, I can finally work on my second app with minimal distraction. Looking back from PresentIn looking back, I would probably choose to go the same route – building a simple first app, a framework, then a second application – if I could go back to doing everything again. But I don't mean to show that this is the best path for everyone to develop their mobile apps. People from different backgrounds will take different routes to suit their own circumstances. What I would suggest is to start with something - not simply but also - easy to build. Then launch the app to the public, as soon as the first release is completed. Because the release of a mobile application is quite unique, compared to other types of software. Going through the whole process sooner rather than later will help to fully understand what it takes to create the next application - the real application. And that's when the journey really began. Start.