As a whole

* + 1. Separation of “global” and “local” commands
       1. global commands don’t get saved in the project data – get saved as global settings / don’t get saved at all (ie. print, debug)
       2. only local commands need to be saved into the project data
    2. imports
    3. permissions
    4. custom functions
    5. Arithmetic

# \*Market Viability

## Analysis of competitors

This technology is not the first of its kind to hit the market. A few notably successful products in the same space currently exist. The most similar competitor in the Android-app-building space is Andromo, a web application best described by its slogan: “Make an Android App. No Coding Required”[[1]](#footnote-1). The product – slightly less than two years old – has seen moderate success, with over 150,000 users registered, a similar number of apps “under development” using the product, and over 12,000 apps published in the Google Play app store[[2]](#footnote-2). Their business model is subscription-based, charging $25/month or $99/year for access to the finished product at the end of the creation process. A premium “professional” subscription is also available at the yearly rate of $499, the main benefit of which is removing the “Powered by Andromo” tag from the published app.

Ignoring the business model, Andromo as a product primarily has one major drawback: Andromo can only make a limited style and type of Android App. In other words, because there is “No Coding Required,” the app can only be customized to a limited degree. In short, Andromo provides a template for users to fill in, and the product is clean and simple, but creating an app outside of that template is impossible. In particular, the navigation flow and style of the app is the same for every Andromo app (“table of contents”-style menu screen that leads to content screens), and the options for content are essentially restricted to displaying web or local content and links to social media pages. Because of this limitation, Andromo is most useful and finds most of its customers with existing businesses who simply want a barebones mobile version of their business, and do not mind if it looks like extremely generic.

Another competitor, AppsGeyser, is similar product advertised as, “Create apps from any web content.” Again, the real core of the product is in its slogan – converting web content to Android apps. According to their website, they have over 100 million installations after launching in January 2011, which is as much an indicator of their tremendous success as the other statistics listed on the website[[3]](#footnote-3). AppsGeyser, however, similar to Andromo, offers very little flexibility – both a strength due to its level of convenience and ease, and a flaw due to its limited nature. Other competitors do not do much more than these two products.

At the very least, the examples of Andromo and AppsGeyser demonstrate that there is clearly a market for an Android-app-building product, and that the product is viable – with great potential for success. With the recent buzz about mobile apps, it is no surprise that these products have met success not long after their launches.

## Key Characteristics

In weighing the pros and cons of the product, it is crucial to keep the context in mind during analysis. That is to say, there are existing, established products out there, operating in a very similar if not the same space, with great success, and backed by some bigger companies. Naturally, EasyAndroid must offer what those other products offer, except better and/or at a cheaper rate. Among those compelling characteristics would be: the ability to turn ideas into reality (Android apps) seamlessly, from start to finish and in a simple, intuitive manner; the scalability of the product – develop once, sell forever; and the strong market demand (as shown by Andromo and AppsGeyser). These are all reasons that make this technology and product a strong investment, but also more or less apply to EasyAndroid’s competitors as well.

There are three key advantages that EasyAndroid could offer over its competitors. The first is simple: aside from the installation and initial setup, EasyAndroid does not require an Internet connection. This means that users can create, develop, and alter their apps without being wired – perhaps most applicable during travel.

Secondly, the product could emphasize learning the Android app creation process. While products like Andromo or AppsGeyser are great at producing the end-product Android app itself, they teach the user nothing about the Android coding. Again, the slogan of “No Code Required” is both a strong point and a weak point for them. EasyAndroid enables users to “try out” Android programming, allowing them to first of all see the actual Java code itself, play with the code, figure out for themselves how different actions affect the code, and see what app features translate to what lines of code.

Lastly, and most significant, flexibility is key. As I described in the previous section, none of the competitors in this market really offer a great amount of flexibility in creating the app. An app created through Andromo will ultimately look and feel like an Andromo app. Similarly, AppsGeyser literally takes a website and converts it to an Android app, with little interaction with the consumer for what happens in between. In contrast, EasyAndroid currently allows users to integrate and use their own custom Java code (if they know Java) into the Android app. Furthermore, the look and feel of the app is up to the user to design and decide – there are no templates for the entire app’s navigation or content, although suggestions are offered and there are defaults (that can be overridden easily) for each item in the app. In short, EasyAndroid offers the user many more options for customization, also in a sense, giving the user more responsibility. While this could be seen as a disadvantage because the app creation process is not quite as seamless, I would argue that the positives greatly outweigh the negatives in this case – especially considering that no product currently offers the user the basic capability of designing their own app’s navigation flow.

Every technology and every product has its flaws, however, and EasyAndroid is no exception. As mentioned before, the fact that EasyAndroid is more flexible than the current competition can also be seen as a negative – that is, because the Android app creation process requires a bit more work on the consumer side, it can be seen as “too complicated,” or “hard to use.” Ideally, there would be a way to simplify the user experience while maintaining the amount of flexibility the product offers.

The lack of an Internet connection can also be seen as a disadvantage. Particularly the setup process is significantly more expensive than the “setup” process for the other products. In contrast the product completely being hosted online, EasyAndroid requires downloading and installing software that becomes a major hindrance to new customers, especially with the current trend of emphasizing speed over all.

But EasyAndroid’s most significant weakness is the fact that it is not yet complete. In particular, the graphical user interface needs to be completed, as well as polishing up the underlying technology, which could take weeks. Moreover, competitors in the same space have been around and established for two years, so even if EasyAndroid as a product were to be completed within the next quarter, there would be a lot of catching up to do. The product would need to be so clearly better than its competitors that everyone switch over to EasyAndroid in order to make it successful.

## The Product

With the context established and analysis of the technology/product complete, hopefully it will be easier to assess the EasyAndroid at Princeton. As previously mentioned, the technology is currently being used to develop a product, through which consumers can transform an app idea into an actual Android app, quite similar to the aforementioned Andromo. To explain the exact mechanics of the product, when the user launches the product, they are shown an editor-like graphical interface, which presents to them options for what exactly they can add to their Android app. If they choose to add a text field for login, for example, the options for determining the properties – height, width, default text, etc. – of that text field show up, asking to be filled (optionally). As they add items to the app, for example, a button or a text block (Andromo adds whole pages at a time), there are shown a preview of what their app looks like with those items, as well as a preview of the hierarchal structure of their app. The user continues this process until they are satisfied with their creation, and then they can build it and install it (or export it) with a single click.

There are several business models that could be used for the commercialization of this product. The most obvious one is the subscription plan used by its competitors at the moment: charge users a monthly or yearly fee for use of the product. The main flaw with this business model is that most users will likely only need to use the product at most once – it defeats the whole purpose of a subscription, unless the plan is to make money off of those who forget to unsubscribe. Another business model is the shrink-wrap model, where revenue would come from one-time purchases of the software (purchase of a valid product key) for use – of course, this can include limited “trial versions,” perhaps online even. I feel that this model means that the average revenue *per* *user* would be much greater, because a one-time purchase means the product can be sold at a higher price – a significantly higher price even, considering that the product essentially replaces an Android developer for small scale Android apps. A final potential business model would be charging for the final step of the app creation (packaging the app for publishing). The main selling point of this business model is that consumers will come back to update or slightly modify their app, and that is where revenue can be made.

Accessing the target customers – businesses or visionaries who want either a simple or prototype Android app before recruiting a software engineering – is perhaps the most difficult aspect of this venture. Aside from standard advertising and marketing means and short of pursuing a partnership with Google, there are no particularly effective methods to reach the target customers. On the other hand, however, words cannot describe how invaluable such a partnership with Google would be. As such, the only method to gain an advantage in the market is to simply be a better product than the competitors, as outlined in the previous sections.

## The Verdict

This technology has a lot of potential. In particular, the market is extremely large for mobile-app-building, shown aptly by the AppsGeyser product. And of course, mobile is *the* market these days. The development and launch of EasyAndroid would mean that anyone – programmer, entrepreneur, or design artist – could potentially create their own Android apps, without needing to hire their own Android specialist to develop their apps. Also, EasyAndroid is quite a small commercial investment. Because it is software, the only real costs would be the development of the product.

But to cut to chase, I would advise *against* investing in EasyAndroid at this moment. Even though the market has a lot of potential right now, strong competitors have already been around for two years – I reference the Beta Golf case on why entering a market with strong incumbent competitors is a poor idea. Furthermore, none of the business models is compelling enough – the product requires a large number of *unique* users, which is difficult to achieve for any product. Maybe if the technology and product were actually complete, and ready to launch, or if Google were backing the product, the venture would succeed. Though it is a relatively small financial investment, the difference between what it is right now – a senior thesis – and a full-fledged product sweeping the market is far too great. So while I support the technology from an academic standpoint and think that it has immense educational value, I cannot confidently say that it would be a profitable investment, and so, again, advise *against* investing in the technology.

# Functionality

1. BRIEF description of Easy Android
2. API
3. Flow of the tool (Make a diagram/flowchart)
   * 1. User has an app idea
     2. User inputs that idea via GUI
     3. Input from GUI is normalized to regular commands
     4. Commands are put into a parser, which converts the commands into a project state
     5. Project state is translated into meaningful Android/Java code
     6. X step: BUILD – Code is compiled and built into apk file via Android tools.
     7. X+1 step: apk installed via ant build script to attached USB device

1. ("Andromo - Make an Android App. No Coding Required") [↑](#footnote-ref-1)
2. ("169,873 Developers Sign Up to Create Apps with Andromo’s Point and Click Online App Maker") [↑](#footnote-ref-2)
3. ("What is AppsGeyser?") [↑](#footnote-ref-3)