

The No Bullshit Bible : Creating Web 2.0 Startups & Programming



**Written by James Gillmore
Edited by Holly Welch**



FaceySpacey

www.faceyspacey.com

Market Research



TABLE OF CONTENTS

Non-Technical

Chapter 8

MARKET RESEARCH	2
1. How to choose your own startup	4
2. How to do market research for your next big idea	11
3. How to pinpoint your niche	15
4. Four product evolution techniques: how to evolve your product for your nich	19
5. Top web/mobile application markets, trends and niches	23
Conclusion & Further Reading	32

8-1 MARKET RESEARCH | HOW TO CHOOSE YOUR OWN STARTUP

So you've finally reached a point where you can work on your own project. You have money saved up, and months to spare to do nothing but work on your new baby instead of client projects. For once!

You're experienced enough to know that you may not get this shot again any time soon. So what you work on has to be what makes it so you never have to do client work again. The project must be able to either make money on its own or warrant being able to raise money for it from serious Silicon Valley venture capitalists. The project must also "float your boat" so to speak. I think the latter is most important.

There was a time when all I cared about was "can it make money?" or "will investors be likely to invest in it?" or "who's likely to acquire this? Google?" etc. I could be selling toilet paper and I wouldn't care--as long as it was very likely to make me rich. This sort of thinking is what spurred the idea for SnackSquare. SnackSquare was a startup that aimed to deliver SMS text messages to people near stores when they were near it. The magic was that we would track people's checkins on Foursquare, Facebook and Twitter to know where they are. And then send SMS text messages on behalf of neighboring businesses to bring them in. To come up with this idea, I followed the trends closely, and predicted that this is where the market was going and that other businesses would eventually figure this out, and people one day would come to expect this location based advertising--i.e. there would be a serious need for such a product, and I'd be far ahead of the curve by making it. The problem was there weren't enough ways to get this information. There still basically isn't (as of August 2011) because people don't check in enough, and because striking deals with carriers to get these locations was extremely difficult, and still didn't get you bonafide opt-ins from users. So anyway, the point is we were ahead of our time, and still are--but did I love location based advertising? No. Not really. And honestly, I think most people don't really love

what their startup is about. I'm sorry, but if you're Zappos.com, shoes can only make you so happy. If you're even Google, optimizing search algorithms can only go so far. Or in Facebook's case, helping everyone else socialize and feel more connected doesn't do that much for you when you just as well rather go use those tools to socialize yourself. And maybe that's the truth of basically any job, it ultimately isn't as good as spending time with friends and family. But then again, maybe it's that your company is an extension of your family.



The point is your project must give you that deeper meaning in life you're looking for, and I'm not sure every software application--no matter what market need it pinpoints or problems of the world it solves--will do that for you. You need to be very discerning about what you decide to commit your life to, and first and foremost you must realize to be successful this must be the one and only project you commit yourself to or you'll be spread too thin and none will succeed. These software startup applications are hard to get right! If Facebook truly did start out as a way for young Zuck to meet girls, that definitely wasn't the case for him--at least not back then. How could that be what he truly wanted to do when he can meet girls a lot more easily in the real world than having to work grueling hours to maintain the world's most popular social application. If he put more thought into it, he might not have done it. Either way, over time it seems

he's rejiggered its purpose to one more in line with what he truly wants, i.e. connecting the world. I just don't buy that he put that self-introspection in at the beginning to know that's what he wanted to do, and I still don't fully buy connecting the world is what he truly wants to do. It sounds more like his marketing pitch for his purpose in life that he just stumbled on. Now he's sucked into this Facebook thing. If he had a year to step back, he may realize it's something totally different he wants to put his time into.

Anyway, so to paint a picture of what does qualify as something you truly want to do from your core--or at least what meets my standards--I'll describe the application I'm currently working on: DreamMakerApp.com. Dream Maker App is a tool for anyone to accomplish their dreams through. To start--in order to follow the FaceySpacey execution formula--it's a simple tool where people can share a small number of goals that make up their dream (5 to be exact), and get advice on them, favors for them, and of course contribute the same to other people's dreams. The end goal however is a learning machine and knowledge generation machine. In the future it will be powered by what I think is a revolutionary way to crowd-source knowledge. Think Wikipedia on steroids. Specifically, it allows people to build tutorials from their own knowledge and the world's knowledge already in the system in a structured way that synthesizes and relates information in a way that is visually the most easy way to learn from. It follows the principles of studying something "General to Specific."

But that's only the first part. The end goal is that this tool will help me find the meaning of life and how we as humans came to be as we are. It will figure out what existence is and its laws. Without going into too much detail about it--since this isn't an infomercial for my own product--I'll share one last thing: basically I've created my own method for studying a subject to learn it as efficiently as possible. I invented this method when I learned to program. I was in a situation where I had to become a coding guru if I ever had any hope of completing some client projects that my life (and my finances & integrity) depended on me getting done. So I plowed through tens of programming

books--just the right books--and got my skills to a professional level in record time. DreamMakerApp's end goal is to help anyone do the same with what they want to learn and therefore make their dreams come true. It crowdsources recipes to learn things and hopefully will crowdsource discovering what existence is at a very deeply level. And here's the most important part: it will become so useful that I too will use it to learn new things I want/need to learn. My hope is it will spawn a new system of education, and change the way schools are run. My hope is that it will crowd-source the discovery of many of the world's problems and things we, as the human race, do not know yet, and otherwise could take centuries to figure out. It will be the genome of the world's knowledge. And I decided that this is what I truly wanted to do because I took inventory of myself and realized that at many stages of my life I was driven by a deep curiosity to figure out what life is and how to best play this game of life. So now I'm building a tool to help me attain that goal.

Therefore, a key sign that you're building what you truly want is if you're building something that will help you accomplish something non-money related that you always wanted to do, most likely unrelated to programming.

Here are more reasons why this is the perfect project for me:

1) I discerned that what's always interested me the most is finding the world's answers. The answers about existence, etc. That's really what gets me going. Not building applications that make money or solve world peace. The world peace bit may truly be your goal. And maybe it's mine too ultimately, but where my humble goal lies is just in figuring out as much as I can about life. And that's what DreamMakerApp is intended to do, and make happen at a rapid pace through its crowd-sourced usage.

2) It represents what I'm truly about. I realized that most my friends and family think I'm all about money. Money Money Money! That was my claim to fame. But that's not good. That's not what life's all about. And that's not what I'm truly about--I just settled for that and gave everyone the wrong impression. So, people are important. There's a time and place for everything. Often it's very

important not to give a shit what people think. Other times it is--i.e. for the people you love and care about. DreamMakerApp represents what I'm truly about, and tells that to the world: again, making dreams come true. That's why I got into software. Software is the closest thing to magic on earth, and if I can help people make their dreams come true, I'm making real life magic happen.

3) Compared to many other applications, I'm best trained to get this project done efficiently.

I'm best as an interface developer, and DreamMakerApp's power comes from its interface which lets people structure information in a way tailored for learning, as well as learn it easily once structured. It's taken me years to master my craft. I only have so many years on this planet. So whatever I do must use the skills I'm best at, especially since I only have so many years on this earth to learn these skills.

4) There still is big money potential, and yes I still want to make money. It's just not top priority, but is still a priority, which is why it's here at #4. The education system is screwed and in the shitter. Education is one of the biggest industries in the world, and one of the most under-served industries because of current incumbents that won't let new technology into schools. I know this point is sounding much more like a standard business plan reason for doing a project, and that's precisely what it is. Again, it's just not top priority...So anyway, lots of money can be made here too. I won't just be broke as a result, which I don't recommend to anyone in the name of doing good. Final Note here: there are lots of acquisition targets that will love to get their hands on it once they see its power--I think it's perfect for Google when they get the Education in bug in a few years.

5) I need to make my mark on the world. Everyone has that goal whether they believe it or not. It's what procreation is about. Making your mark on the world and procreation in general are a manifestation of our innate desire to live forever. Deep inside ourselves, whether we realize it or not, we realize we're more than just one person, but a species (and ultimately everything in the Universe). So to further our true selves--again, that means our species and universe--we must create something as lasting as possible. I've determined that a learning and teaching engine powered by everyone in the world is the way to make the absolute most biggest mark possible! Think of the proliferation of open source and how much code has been built so quickly, i.e. like in Linux. An engine that empowers everyone to work together to discover life's mysteries means our species can work most efficiently to find these answers. That engine has the potential to discover all the startups that should be built in the following year that we'll see on Techcrunch in a fraction of the time it takes isolated entrepreneurs to dream up those ideas on their own. It will commoditize innovation. If I can build that

engine, I've made the biggest mark possible....So that's me having gotten in touch with my true aim in life--simply to make the biggest mark possible while satisfying my own need for answers.

So that's why that project hits the spot for me. The main reason is really #1: my whole life I've been curious about the answers to life, and I intend to make this tool help find them. Period. That's been the driving theme of my life, and what I feel is the reason I've been put on this earth. I used to think that reason was to make money and build an empire, but empires come and go, and I want to make something truly lasting. I want to make my mark. Point blank period. It's cliché, but it's true. And I think it's true for all real entrepreneurs, and really everyone in one way or another. I think Facebook is more akin to an empire. It definitely has figured out a bunch of stuff, and will continue to (same with Apple, and Microsoft), and will influence generations of inventors to come. However, I want to make a discovery engine that will churn out mountains of discoveries, not just a few. What Microsoft has done for computing and software can be explained in one book. DreamMakerApp will be a tool that never stops making discoveries, and lets anyone contribute to them, and the result will be libraries of discoveries. And as a result, I will be able to make the biggest mark possible.

Whatever you do, shoot to make your mark. Do you really find fooling around with gizmos to be fun? Finding out what cool bars and coffee shops are near you (ehem Foursquare)? Tweeting 140 characters back and forth between friends? Like, what I'm saying is "fun" your true end goal. Is that really what makes you tick. Plain old "fun." That's what most of the applications on Techcrunch do--and if they don't, they help extract money from these "fun networks" like all the facebook, search engine and twitter advertising and metric companies. To me, at least, fun is not enough for me to have a complete life. I think people armed with better information about existence, life, and the cosmos will have even more fun and enjoyment than people who rely on going out and drinking on the weekend like most of us (including myself) currently do.



That all said, the real takeaway is that you need to look back on your life and figure out what really has made you tick all along. What theme has been present in all stages of your life. I don't have all the answers--yet ;) . I could be wrong--maybe you don't have to make the biggest mark to find the meaning you need. Without a world-changing communication tool like Twitter or knowledgebase like Wikipedia or awesome tech blog like Techcrunch, I may have never had the opportunity to figure out the above idea I'm developing. So smaller marks are probably just fine to put your mind to. But there must be something that is innately "you" that you should be doing. Are you doing that or just grabbing the first hot idea that embellishes on products you've seen on Techcrunch? Is your product just solving the natural progression in market needs? When people see it and hear about it, will they say to themselves: "*That's so you. Now I truly get you*"? Or will they think: "oh, wow, you made a lot of money selling widgets on the net. great for you."?

You need to truly evaluate your life to figure out what startup you should do. You need to imagine you will do this and only this for the rest of your life, and determine if that will work for you--because these startups take a long time and you may very well be

doing this for a while. Will that be the work contribution you want to make to the world in your life (outside of your family)?

After you've done all of that self-consultation, then run it through the same standard business plan filter you should have ingrained in your head by now from all the Techcrunch and Mashable articles you've read. You still need to make sure it can make money, and you can actually execute it because it's not too pie in the sky. And maybe you'll have to tweak it a bit, or plan monetization features for the future, but overall it's still the same essence you were going for unobstructed by financial goals.

So that's the end of this "tutorial"--although, sure, it's basically my opinions, and lacking actionable to-dos like I hope you'd find in my other tutorials. So all I can say is just look long and hard for what's "you"--something you've always wanted to do long before you were programmer--and you probably will have to spend a lot of time doing projects that don't cut the mustard to figure it out, and not much I can say can aid that. Anyone who has worked a long time for a company or clients, or even their own grueling projects, will eventually come to the conclusion that for whatever I do next I must be able to earnestly say to myself that this is what I want to do for the rest of my life.

8-2 MARKET RESEARCH | HOW TO DO MARKET RESEARCH FOR YOUR NEXT BIG IDEA

6 years ago (i.e. 2005 and before) it was a lot more difficult to find startup ideas. To follow trends of invention required inside knowledge, and any ideas you'd come up with would require too much money for you to realistically do. Now, you can read Techcrunch for a year, and pick from a million startup ideas that would make sense in the market as a natural progression. Tons of other startups are showing you every day how you can embellish on what's already out there to make a successful

company/business. The best way as a budding entrepreneur to figure out what market you want to tackle is to read these tech blogs--daily!

TechCrunch

Whenever I first meet someone that wants to do a web application or mobile application, etc, the first thing I ask them is: "Do you read Techcrunch?" And then I ask: "Do you read it every day?" And then: "Do you read every single article?" If you don't skim through every article every day and devour the ones that are most aligned with your interests, you're a fake-ass software entrepreneur and you shouldn't undertake any startup any time soon. I'm not speaking to super programmers that deal with coding down to the level of the CPU or are immersed in building a new solid state hardware-accelerated database, or hadoop masters that have a genius way to produce answers out of "big data," or whatever. I'm speaking to guys that don't yet code and want to build an application. At the very least, they should be reading techcrunch every day to do their market research. You can't code for godsake--you at least need to be a master of the market, and therefore a master product guy. In addition to [Techcrunch](#), they should also read [gigaom.com](#), [mashable.com](#), and [readwriteweb.com](#). That's the lineup.

That said, there is a very specific type of market research you must be doing. You could have a great idea, and then search techcrunch for competitors and call that market research when you find none. But that's not the sort of market research I'm talking about. Sure you need to be able to do that. It's surprising as hell to me how often I show people how to search for competitors on Techcrunch (by appending

"[site:techcrunch.com](https://www.techcrunch.com)" to my google search for industry keywords), and they're elated with how easy I did the competitive research. It's even more surprising how many people think they have no competitors. But that's another story.

If you're serious about being yet another Web 2.0 wannabe Techcrunch baby, you need to be scanning all the markets, and application entrants they have in order to find the perfect niche and pocket for you to enter with your startup. You need to say to yourself, I don't really care about what my product does. It could be social, based around advertising, a new way to search information, or some new form of communication. It doesn't matter to you. You're looking for the perfect opportunity that you can execute to a point where it's defensible before you have too much competition, and you don't care what the market is. If you're serious about being a tech entrepreneur, you need to have many ideas you're juggling and always in pursuit of a better idea (until you finally start executing it of course).

In my [Market Research 1 - How to Choose Your Own Startup](#) tutorial, I discuss how important is to pick a startup that is dear to your heart and that is what you want to do for the rest of your life. I'm not negating that. In my case, I spent years just scanning these tech blogs looking for market opportunities based on the technologies and applications whose launch they covered daily. I was looking for idea inspiration, and holes these products left in the market. I didn't care what market the product was going to be in, so long as I could make a successful company out of it as efficiently as possible. I think you basically have to go through that if you ever want to have a personal knowledge-base big enough to allow you to pinpoint what the final startup for you will actually be. You need to become adept at crafting market opportunities by comparing your ideas to what's coming out. Once you have that muscle worked out over years, then you can figure out what you truly want to do. You'll also be able more easily figure out how to make what you want to do make money and solve market needs.

So back to the research. My point is that you don't research an opportunity in relation to an idea you already have. You research the markets, and constantly run through new ideas. It's what VCs basically do all day, automated by your help when you bring them your latest and greatest idea (hopefully executed already to some degree if you have any hope of getting any investment). Analyzing all the applications that come out also teaches you about the latest technologies to use and how you can use them, what APIs are available, ideas for fresh user interfaces, etc. You learn how these products progress from their initial launch as an easy product to something more advanced that can make money. You learn from all the struggles these companies have so you don't make the same mistakes. You learn about all the different markets available in the first place, so you can find where you best belong. In the end, you'll most likely mash up ideas from hundreds of different startups to create what ends up being yours, which is the most important reason you need review everything that comes out on Techcrunch. Everything you ever read and learned on Techcrunch and Mashable will find its way into your product. Also, you'll most likely copy many interfaces from other applications, knowingly or unknowingly.

Making a software application isn't like inventing the next style of Jazz. You don't need to be Miles Davis or John Coltrane. Have one truly innovative aspect, but for everything else, copy copy copy. It will save you tons of time, and in the way you combine different influences your product will be unique too. And to do so, you need to read the big 4 daily: Techcrunch, Mashable, Gigaom, and ReadWriteWeb. End of story...And wait at least 1-2 years before you go for it with your own idea. Until then, hone your skills. That's the right way to do market research in the Techcrunch era. God bless 'em!

8-3 MARKET RESEARCH | HOW TO PINPOINT YOUR NICHE

If you haven't done so already, read my [TOP WEB/MOBILE APPLICATION MARKETS, TRENDS & NICHE](#)s article. Once you've read that you'll have in your head 30+ potential markets for you to tap. So let's first figure out which niche is right for you.

First you must take honest inventory of your skills and that of your team. Most likely, if you're like most of my audience members, *Big Data* will be out. So will be *Cloud Computing and Infrastructure*, and all the other highly technical niches. Unless you're a serious computer engineer, you're going to want to stick to interface and user experience driven websites. Let's operate with that assumption.

That leaves the following niches:

-Crowdsourcing

-Gamification

-Advertising

-Content

-Q&A

-Mobile/Tablet Interfaces

-Social

-Messaging

-Education

-NFC

-Deals

-Local

-Video

-possibly Payments

-possibly Games

-possibly Analytics

And I've left out a few like *Support, CRM*, etc, because I don't find them exciting. The point is to find your focus anyway. So we'll do that by using the focus I'd have for myself and I recommend to others similar to me.

Let's further narrow the above list. You're a web developer, and your iOS and Android skills aren't quite up to par yet, or it's just cheaper for you to not go there with your team. But you still want to be in a position where you're not competing with too many companies. And you want to have as many options for ideas available to you as possible. Here's my top 5 picks:

-Crowdsourcing

-Gamification

-Mobile/Tablet Interfaces

-Education

-Local

Now, building slick tablet interfaces in *HTML 5* is definitely on the come up. It's going to be hot for the next 2 years. There's so much that can be done there. Most sites on tablets are just the regular website made for a computer. They all really should be slidey websites that take advantage of gestures. You should be sliding left and right, not just up and down. You should be tapping part of the screen for a mini set of controls to appear through which you can manipulate the page. Etc. There's a lot that can be done here. I highly recommend it for product guys that have done nothing but look at the interfaces of newly launched startups on Techcrunch all day. Just invent a

new interface, and figure out how to build a product around, or sell it as a service to other sites that may want to incorporate it.

Crowdsourcing is also fantastic because--in my opinion--there's so much information out there that has not been crowdsourced yet into neat bundles that are easy to learn from. I really believe in the power of Crowdsourcing and I think we're going to figure out how to put together our heads as the human race to accomplish some great things through tools geared towards lots of distributed people working together. This niche has a lot of potential for you to imagine disruptive ideas.

Education is hot for so many reasons. I'd just like to see tons of textbooks turned into all kinds of learning experiences on tablets. That doesn't even require fighting the current incumbents keeping new technology innovations out.

Local is super solid and will continue to be. There's just so much that can happen there, especially since smart phones just keep getting more and more prevalent. Look for deep integration into other things outside of the phone, i.e. the POS, and other electronics. The phone you carry on you is only going to get more and more deeply connected to everything within close proximity to you. There's still tons of potential for ideas here to make these connections.

Gamification is big because there are so many game dynamics you can invent to motivate people to perform the actions you want them to perform in your application. When I planned ThirstyVIP.com we spent months imagining several complex games before we finally settled on something more communication and transaction oriented. That doesn't mean the gamification was a bad idea. It just would have been better if a company like BadgeVille could plug it into our application. At the time we discovered all sorts of game dynamics, from virtual good stuff, to MMORPG treasure chests of items you collected and earned armor points from and the like to plain badges. There's tons

of stuff you can do. There is basically an endless amount of ways you can reward your users and rules/constraints that govern what users must do.

Ok, so now with the markets we plan to enter in mind, read Techcrunch.com every day until you figure out an idea not done yet that will solve a real problem that some group of users will be willing to pay for...

You think I'm joking? There really is no other way. You need to do this for a while. You need to look out for the articles about all the graduates from programs like Techstars and Y Combinator. I love those articles because it lets me check out 10+ startups quickly all at once. That's how I sharpen my market and product knowledge.

For real though, I wish I could give more advice. I'm trying to think of specifically what happens in my head when I see a newly launched product that gives me an idea for something else related, but new and unique. Typically new ideas will be a mashup of multiple other ideas. Or, it will be a play on an old theme. For example, Scvngr came up with a spin on the group buying daily deals thing where they created basically a game dynamic where the deal gets better or changes each time you come back and redeem it, thereby solving the problem Groupon was having in that it didn't create any repeat customers. Now, I'm not sure how successful it's been, but it still highlights how you can watch market ideas and use a little bit of elbow grease and wit to come up with a new and innovative spin that patches up a hole in what's already out there. This example is exactly what I'm talking about. You need to be able to see what problems current solutions/product do not solve, and like a quarterback sneak through the middle unexpectedly.

8-4 MARKET RESEARCH | FOUR PRODUCT EVOLUTION

TECHNIQUES: HOW TO EVOLVE YOUR PRODUCT FOR YOUR NICHE

So you got the niche pinpointed, and you generally have the problem you're trying to solve. Now you need to craft your product to be exactly right. To demonstrate how this is done, and how you deal with all the competing factors, we're going to run through an example of an actual product I've done: ThirstyVIP.

The goal of ThirstyVIP.com was originally to be a nightlife social network. We however spent a year evolving exactly what it would do in the nightlife industry, and changed the concept several times. Here's the progression of the different ideas:

1) Nightlife social network - this is what my client wanted

2) Nightlife SaaS b2b tool - this was my first alteration on the idea

3) Foursquare specifically for nightlife - we settled on this, but then broke it down further

4) Heavy-duty location based drinking game

5) Even more unnecessary features for Events, etc

6) Reduced concept with super simplified gaming aspect, and generally revolving around communication

7) Integration of a bar tab to keep track of your drink checkins in preparation for a future where actual transactions could be made through the app

So basically what happened is my client, Tyler Beerman, came to me and said he wanted to do something big for Nightlife. He already had a partially built but failed past prototype from another developer in hand. It was an iPhone app that basically served as a directory of what's going on in the nightlife scene around you. He wanted to make it more social.

Keep in mind that this was just before the geo-networks like Foursquare took off. I said to myself: let's make something that can actually make money since that's how I've always rolled. I've always hated building products that have to become immensely popular and take on tons of funding before it can make money. I wanted to make a b2b SaaS solution we could immediately start charging a monthly fee for. So in #2 above I came through with an idea of tracking the performance of Facebook events so nightlife promoters could better optimize their campaigns, as well generally make the application an automated way of reaching out and marketing to your Facebook following. So one of the tools was a facebook event scheduler that would automatically schedule events each week so you didn't have to do it each week. You could then track the aggregate performance of that weekly event over time. There were also tools to automate and simplify inviting people and reaching out to them, etc. We were going to charge a monthly fee for it and come out the gate with something that didn't have the "chicken and the egg problem" and could make money immediately.

Product Evolution Technique #1

If you're not familiar with the "chicken and the egg" scenario, the best example is a real estate site where you need sellers and buyers. Buyers don't want to use your site if you have no sellers, and sellers don't want to use it if you have no buyers. So you're stuck in a tough position where your platform is useless until you get critical mass, and getting critical mass is near impossible. Where I was coming from with the #2 concept was that it didn't need critical mass to be successful. It didn't even need to be filled with tons of content like a video site. The technology just had to work, and it piggy-backed on the data (i.e. friends, fans, etc) that customers would already have living inside their Facebook accounts. This is Product Evolution Technique #1--i.e. eliminating the chicken and the egg problem by making a tool that services a specific group with the data they bring to the table.

Product Evolution Technique #2

After we basically finished planning this product, Tyler, my client, said he rather do something more like a Foursquare for Nightlife. So we scrapped all that and went there. In this stage, we spent a lot of time figuring out how we would be different from Foursquare. We said to ourselves, we have to be totally different if we want to compete. We experimented with tons of game dynamics. What that means is we planned a tool for bar owners/managers to create all sorts of promotions. We ended up with a multi-step and multi-path promotion creator where you could make basically an endless variety of promotions. You could make ones where earn promotions once they reach a certain status based on points. You could make virtual good promotions that you earn and have to redeem immediately. You could make virtual good promotions that the end user will be able to store in their MMORPG style treasure chest and use whenever they want. You could set dates, times, expiration dates, etc, for everything. Really, you could do a lot more. The point is we went overboard. But what we did do was explore basically every possibility of gameplay. And I think that sort of planning and imagining is key to the creation of any startup. Just make sure you don't waste all the time and money to do this in code. We did this in layouts according to my FaceySpacey Speccing techniques. Let's call this buch-wild exploration Product Evolution Technique #2. Basically, every startup has to go through this to earn their stripes.

Product Evolution Technique #3

After that, we went along even farther, and even added tools to incorporate events, as noted in step #5 at the top of this article. And soon enough we got to a point where we realized we were trying to do too much, but nevertheless had a huge repertoire--and ultimately a study--of all the possibilities for us. This led to the execution of Product Evolution Technique #3 where we trimmed the fat. So we removed Events, and made

it so you can create only 1 type of promotion. Well, 2 types, because it had 2 variants. The bottom line is we simplified it by like 95% to just the best part.

At this point in time, we came to a final realization about what ThirstyVIP really is. We realized that this application was going to be a way to make real transactions at bars. Eventually, the app would connect to the POS (the “point of sale” computer where bartenders process your orders) and your bar tab on your phone would be the same as on the POS machine. And of course you could make the transaction through your phone when closing your tab. We realized that this was far off, but what we wanted to do was build the framework to go there. We felt it important for future investors to see our vision now, and for end users to get the picture of what ThirstyVIP was about.

What this all looked like is you could checkin to a bar, and perform drink checkins throughout your night. That was always part of the application. But now in the final stages of speccing this product, we added a fly-up screen that’s always present on every page of the application that lists all the drinks you’ve had, and the # of points they were worth, and of course the totals. The idea was that since we weren’t actually going to process transactions yet, your bar tab sum would be the total points you’ve earned. And you’d earn bonus points for closing tabs with larger totals. The bar tab also let you click back to the posts corresponding to each previous drink checkin you made, as well as one-click drink another similar drink, etc. So it was connected to the social and communication aspects too, and served multiple purposes. In short bar tab fit right in, and made you feel like you were really at a bar while using the application.

Product Evolution Technique #4

So that final addition is Product Evolution Technique #4. It can be summarized as: after you’ve done all your speccing of what you thought you wanted, don’t be afraid to come to a super realization of what your application is truly meant to be; and then backtrack a bit and take the time to rework into the application what’s necessary to

make it happen, possibly trimming the fat from other less necessary things. We found ThirstyVIP's purpose at this point in time and what it was all about. We were able to do so because we explored and examined so many possibly paths the application could take. We did so in a lot less time than it would have taken had we did this all in code by building useable stuff. We were unrelenting, and once we felt we really mastered the terrain, we zeroed in on the core value proposition, and made sure to get that right, while simplifying the rest.

8-5 MARKET RESEARCH | TOP WEB/MOBILE APPLICATION MARKETS, TRENDS & NICHEs

Before you dive into your next startup, it's imperative you know all the opportunities available to you. There are many markets, trends and niches, and some more than others just waiting for you to solve their problems. I think it's very important to be able to imagine products in all these niches so you can compare what's really the best startup for you to tackle next. Doing so will of course greatly improve your product imagination muscle. It will also make it so you can more quickly digest Techcrunch, Mashable, Gigaom, ReadWriteWeb on a daily basis--you'll know exactly what sort of startup you're looking at and what problems it solves with just a few words about it. I can just skim article excerpts, sometimes just the blog article title, and sometimes just the name of a startup to know precisely what it does, and most of the time I'm right. You should be able to do the same if you think you have the magic ability to predict what the market needs next.

So below is a list of all the markets, trends and niches I can think of for 2011. I made the list just by going through the last 20 pages of the aforementioned top tech blogs in order to remind myself of all the niches available. Here's the list with a quick description for each:

1) **Deals** - think Groupon, JetSetter, even Coupons.com, etc. With the economy being messed up, it was the perfect time for the Deals market to explode. Various gimmicks from “flash sales” as provided by Groupon and Living Social to location based deals as seen in Foursquare have proliferated as a result.

2) **Search** - this niche doesn't get as much attention as it has in past years due to Google's dominance, but there have been new players, of which Blekko is the biggest one. There are also related products like Qwiki.com, which offers a generated media experience for tons of topics you can search for, powered by Wikipedia.

3) **Local** - Foursquare, Gowalla and Loopt pioneered this market--i.e. the “checkin” game--in 2009 and 2010. It hasn't exploded as quickly as people might have thought, basically because it's restricted by the pace at which smart phones become popular. However, Foursquare, in particular, has stayed steady, and has made smart moves like partnering with Groupon & Living Social to stuff their app full of deals. It's been said non-stop that checking in and giving out your location will only become mainstream when the quality and quantity of the deals catch up. Foursquare is on their way to making that happen. On a side note, I really feel like Apple and Google will do something major to flank everyone by injecting some serious tools to get deals into their iOS and Android phones, respectively.

4) **Video** - video has long been dominated by Youtube, though with the majority of the content being low quality. Over time a middle tier has emerged of professional content. And lots of niche-specific sites have been created to target their corresponding audiences. One of our client startups, CrimeTV.com, is a perfect example of that in how it features only video of crime movies, shows, etc. Lots of people have video creation tools now. Expect more niche video sites to emerge, and auxiliary services like ZenCoder.com to help you convert your video. Also, HTML 5 has greatly simplified the the tools and skills it takes to put video on your site since you no longer need to know how to code in flash's somewhat obscure ActionScript programming language. Expect more tools like ZenCoder to emerge and to combine with HTML 5 to eliminate all challenges related to getting a professional video site going; and then expect everyone to be doing custom video--and that mainly means custom branded video players.

5) **Big Data** - this market is one of the fastest growing. Gigaom.com does a great job covering this niche. They seem to have made a point of focusing on it. Incumbents are mainly focusing on 2 things: productng intelligence from large data-sets and using Hadoop to do so. I would also lump in

companies like Xeround, NimbusDB, ScaleDB, and ScaleBase, who are working to make SQL scalable in terms of load and data size in a way that allows application developers to connect to their databases the same way they would as if they had one simple MySQL database. Techcrunch doesn't cover "Big Data" much because they focus mainly on consumer-facing startups, but there are some extremely smart guys out there in an arms race to build solutions that can handle the exponentially growing data-sets we're all spitting out day in and day out on facebook, twitter, youtube, etc. Every day, each one of us is contributing more data than we were the year before (according to a report by Facebook on its users' sharing patterns), and applications need to be prepared to handle this. Guys in this market will provide those solutions, and ultimately all the tools approaching artificial intelligence to produce serendipitous results for end users.

6) Gamification - this niche is an important one. Companies like BadgeVille.com provide tools you can easily embed in your site to provide the incentives that go along with gaming. The idea is you can motivate users to perform normal actions such as commenting by giving them a badge, or status, or virtual goods, etc. Expect gamification to gradually grow and become more apart of the sites you frequent in new and unique ways. There are endless ways to shape the dynamics of a game. Expect to see tons of twists in gaming rules and rewards that sit in juxtaposition to the natural things you do on a site to both enhance the experience for you and get you doing the actions the site owner wants you to do.

7) Payments - it's all about mobile payments right now, and NFC payments made by waving your phone. That's what this is all about. Facebook credits are also worth watching, and most likely will become a mobile payment platform as well, but not for a few years.

8) Advertising - Advertising is a huge market with tons of sub-niches. For example, in-photo advertising startups like Luminate.com (formerly Pixazza.com), Gum Gum, and a few others stuff advertisements into the relatively small space provided by photos. And there are of course in-video advertising networks, and many many other twists on the advertising network model. HTML 5 and tablet advertising is on its way up. Location based advertising where the ads know what stores you are near of course is huge too. Wherever an ad can be slapped, an advertising network company will plop up to serve ads there. Whatever information can be used to make ads more targeted will also serve as the basis for a new advertising company. This niche has been big since the dawn of the internet and will continue to be forever.

9) **Games** - I won't say much about this market except that a ton of tools have emerged to create cross-platform games that work on iOS, Android, HTML 5 web browsers, etc. Siblingz is one such company, and here's a few more: Unity, Corona, AppCelerator, etc.

10) **Augment Reality** - Layar is the biggest player here. They and competitors--for now--allow you to aim your phone at the world around you and see popups containing information about what you see. So basically these apps detect what's in the real world. Can't wait for all this stuff to make it into my contact lenses! ps. not too many entrants seem to enter this space--probably because it's a long 10-50 year fight, but don't take your eyes off this niche. We'll probably get a breakthrough soon that will change the game.

11) **Messaging** - Twitter, group messaging platforms like GroupMe, and even Facebook fall into this niche. Messaging and communication is where all innovation has happened first on the web. After all, the power of the web is how it can connect you to others. Startups like Lissn have tried to "pull a twitter" and invent a new twist on communication. Expect many startups to continue to find new ways to shape the communication experience.

12) **Cloud Computing & Infrastructure** - Amazon AWS, OpenStack, MySQL scaling guys like Xeround and ScaleBase, Eucalyptus, etc. To me it's all about Amazon AWS and ultimately nothing else. Scaling big MySQL databases isn't solved yet, but the aforementioned guys will, and I assume Amazon will eventually buy one of those companies and in doing so will continue to provide the complete scaling package.

13) **Mobile/Tablet Interfaces** - to me we got two kings here: FlipBoard in terms of native iOS code on the iPad, and OnSwipe in HTML 5. Both are leading the pack in terms of the interfaces we expect our news content presented within. Expect them and lots of other guys to push the limits from what we expect out of tablet interfaces. A lot can be done with the gestures they can sense that hasn't been done yet. Also, watch out for jQuery Mobile which is obviously a jQuery platform for coding sexy mobile and tablet interfaces. It will bring sexy touch HTML 5 interfaces to lots of sites soon.

14) **Marketplaces** - AirBnB, GetAround, Red Beacon, ZocDoc and tons of other sites offer a place where you can hire service providers in a specific industry, or where you can rent a place or car from someone else. I've always loved marketplace sites. They have easy business models to pinpoint--just charge a cut of a transaction--and people don't mind paying those fees. They expect them. As more

and more people get on the internet, more and more antiquated industries are being made efficient through the internet, while the middle-man charging exorbitant fees are being removed. There's always a new market (or, rather, an "old market" that's under-served) waiting for its two opposing parties to be connected.

15) Content - in this category, I'd put the Huffington Post, blogs like Techcrunch, and all the other top news destinations, information libraries, etc. This niche isn't the sexiest of niches, and isn't really one that you innovate within through genius software concoctions, but nevertheless it needs to be mentioned. You can still go out and make a great content site around some obscure niche and sell ads at a premium to your highly targeted audience.

16) Q&A - here Quora is obviously the golden child of the month, but other sites include: StackOverflow.com for programmers, and ultimately the millions of forums on the net. Q&A has always been core to what the internet is about. It's where you go when you absolutely need to find an answer and can't do so in your regular life. You reach out into the world hope someone anywhere across the globe may have the solution. And they usually do! Startups will always be finding more efficiencies to getting the answers you need.

17) Social - social is basically a part of every niche these days. It's not enough to be just a social startup anymore since Facebook nailed it. Every application basically must have social component to connect its users together. End of story here. You can't just make a "social network" anymore and hope to hit it big. Build a service, and make sure it's very social.

18) Support - products like ZenDesk dominate here. Even bug trackers like Fogbugz solve support by turning emails into tasks. There are a lot of players here. Others include Get Satisfaction, Assistly, etc.

19) Project Management & Productivity - there have always been a ton of project management, to do list apps, and productivity tools. The reason is because software developers need these tools as part of their daily work, and therefore they all seem to think they have some original twist on how to make teams more productive. The biggest players are Yammer, Social Cast, Fogbugz, Pivotal Tracker, Light House, and many many others.

20) Music - the leaders are Spotify, Pandora, Groove Shark, Rd.io, Last.fm. We all love music, and startups continue to fight the record labels here. For a while, it seemed like it was going to be

impossible to make a profit here, but it seems that it's possible now. I wish Groove Shark would figure out how to play by the rules more and get into the iOS app store! They're my favorite music application.

21) Photo Sharing - Instagram, Path, Color, and Facebook itself. With the rise of taking high quality photos via your phone, photo-sharing has become extremely popular. I'm not the biggest fan of this niche. To me, it's boring, and I think there are bigger more important problems to solve, but hey, to each their own.

22) Analytics - I love the analytics niche. Metrics is a great niche to make money out of. The reason is because it produces actionable results that can help customers of these tools make money. These tools innately justify their monthly fees by showing you where you can make more money. These companies also never have the "chicken and the egg" problem where you need 2 parties to make your app useful, and without one you can't get users of the other type, and therefore it's very hard to get any users. Marketplaces like AirBnB, and real estate startups that connect agents and home-owners to those looking for dwellings, generally have these problems. But metrics startups don't require a critical mass of users to become successful. The creators of these startups can just focus on making pretty graphs and crunching tons of numbers. Then they can charge away to each customer they can signup without having to wait until it has users or data/content of its own to become successful. Examples include ChartBeat.com, Viral Heat, BackType, GoodData, Mix Panel, etc. Keep in mind these products each provide different sorts of metrics. For example, Chartbeat tracks visitors on your site in realtime, Viral Heat tracks your social media footprint, and Mix Panel tracks engagement on your site.

23) Education - this is basically the largest under-served niche. It's no secret what's going on here: current incumbents won't let technology innovations into the schools, and lots of legislation is required to change things, etc. My startup DreamMakerApp will revolutionize this industry through its new take on how to structure information and easily learn from it. I also see lots of startups emerging on tablets to replace textbooks. One big player in this space getting a lot of buzz is Khan Academy which offers user generated videos on tons of topics.

24) Ecommerce - like communication, ecommerce has always been at the heart of the web. It's where you communicate through money/transactions. Ecommerce isn't as sexy as it once was since in the first half of the 2000s every store and their mom's store went online to sell xyz widgets. There

are still new flavors of ecommerce being invented though. For example, Style Factory.com has added the twist where people can vote on how and what custom furniture will be built, and Groupon et al in general have added the “group buying” twist to motivate people to buy in mass when a sale “tips” and enough people want it, at which point a deal is unlocked.

25) APIs - I love API companies. One of my particular favorites has always been Twilio--probably since I've used it a lot. Twilio is an API to easily send and receive text messages, and eve phone calls. API companies are highly technical and require thorough technical infrastructures to be snappy, but evade a lot of the “chicken and the egg” critical mass problems because they can make money even if they just have a few customers.

26) Site Builders - there have always been tons of “site builder” companies. Every developer and their mom has the light bulb moment where they realize they can make their work for clients easier if they build a tool to generate a website so they don't have to build it from scratch each time. So there are lots of takes on these sort of tools. OnSwipe for example is pioneering this space for building tablet-optimized sites. For a lot longer, [Mobify.me](#) has been doing the same for mobile phones. For websites, my favorite site builders are [Flavors.me](#), SquareSpace, and Yola.

27) Healthcare - Microsoft Health Vault and Google Health have both tried to make big plays here. Just this summer Google Health shut down though. It shut down because people weren't interested in a health records file cabinet, and what was really needed was a way for people to save money, and for clinicians to be able to make money even if they don't see you in person. In the US, clinicians must see you to be reimbursed by insurance providers. Therefore any digital tools that reduce how often you have to go do doctor appointments don't create any value for clinicians, and therefore such tools won't take off. This niche is in desperate need of legislative reform. To my readers, do not enter this niche. Not yet, at least. I do have a vision for the future when your phone can give you all your vitals, and you can plug into a simple station and get reports from doctors around the world, etc. And several companies have started to deliver on this. Although they're not too advanced, they definitely highlight this future I'm talking about.

28) Crowdsourcing - my favorite startup here is CrowdFlower.com. They provide tools to divvy out tons of small tasks to remote workers (usually in India) to execute them in a high quality way. It's built on Amazon's Mechanical Turk--which is a marketplace to do just this--but it adds various tools to produce higher quality results. For example, it lets you feed in a few intentionally super easy tasks,

and remote workers that get them wrong are rated lower. That allows you to build the best team of remote workers and get the best results. Other crowdsourcing startups I love are 99designs.com and CrowdSpring.com, which let you crowdsource design and web work. To me, crowdsourcing is what the web is all about. My upcoming startup, DreamMakerApp, is all about crowdsourcing knowledge and learning. Crowdsourcing is how we as a human race will become exponentially smarter through working together. We've only touched the tip of the ice berg here. I anticipate tons more tools this century being released that help harness knowledge and contribution to quickly get results through distributed labor.

29) Voice/Visual/Gesture/Etc Recognition - i love startups in this advanced category of interrupting complex input and figure out what it is, whether the input be voice, still images, faces in particular, moving images (i.e. video), gestures, etc. I don't know too much about the actual algorithms that power these applications, and I've always wondered when voice recognition would get perfect and what sort of machine learning it would take to get there, but I do know that these tools do get better with each year. I still think it's going to take some major machine learning break-through unrelated to the recognition niche before these tools really hit the spot. Right now, it's still hit or miss, but when it's accurate, it's awesome. Facial recognition has gotten pretty useful lately, like when you're tagging photos of friends and it can assist you in getting it done quicker. Gesture recognition has also gotten very good thanks to the iPhone and iPad. There's not much else I can say here except I can't wait for recognition technology and the corresponding sensors needed to collect the input to become widespread and part of our daily lives, just automating everything for us. I bet we'll get there by the end of the century.

30) NFC - NFC is about to blow up! Period. Not just for lame-ass swiping of your phone to make payments. I imagine a world where you walk into a store, and as you peruse aisles, your phone is aware of each product you pass and makes a virtual/graphical representation of the same things on the shelves. And when you pick an item up and put it in your physical cart, an application on your phone knows you picked it up, and totals its cost with the other items in your cart. NFC isn't just about phones. What's going on here is they're working to get little passive NFC chips built into products instead of bar codes. There is a consortium for this, and their goal is to bring the price of those NFC bar code chips down to 5 cents, and once that happens this technology will be affordable enough to start putting in all/most products.

31) CRM - Salesforce is king here. Sugar CRM offers an open source solution. Bantam Live, which was acquired about a year ago by Constant Contact, was a social CRM with a Yammer-like interface. There are a bunch of other social CRMs which basically aim to make Twitter a CRM tool when people complain or mention your product. They allow customers to flow in from Twitter into their platform built on top of it. Also, when companies like Foursquare build their little business panels for business owners to see is checking into their stores, that's also considered CRM. The analytics within Facebook's "pages" tools are also very CRMish. Basically any product, especially marketplace apps, that have one party selling a service need CRM tools to manage all the leads coming through the product/marketplace.

32) Finance - from tools that predict stock trends based on data from Twitter to the Zecco's and eTrade's to peer-to-peer lending marketplaces like Prosper.com and LendingClub.com to social investing tools where you can see what others are investing in and invest like them, finance will always have a place on the web. If you don't come from Finance to begin with, don't build a startup here though.

33) Coding Efficiency - this is a niche I invented myself, being that I'm a coder. I've never actually heard this term. But basically there are tons of tools for developers to code faster. From frameworks to IDEs to distributed version control hosting to remote code collaboration tools, there's a lot going on here. Coders love coding things for themselves. So this niche will always be packed full of awesome stuff. One tool to look out for all you Javascript developers is: Cloud 9 IDE. It's a tool to store and collaborate on client-side javascript code in the browser, debug it, and deploy it. It comes with all the standard IDE features like pointing out errors and code completion, and allows you to collaborate with others through tools to add and remove developers, etc. Check it out.

Conclusion & Further Reading

We at FaceySpacey hope you've enjoyed our FaceySpacey Bible, and are coming away many times more ready to succeed at your next software startup. At the very least, you should have a birds-eye-view of what you need to do to get your startup, and have quelled a lot of insecurities you may have had regarding how you should execute it. That said, I will point to you to what you should do next.

As promised, the following is a list of the precise books I read to master web development using HTML/CSS, Javascript, PHP & MySQL. They are presented in the best order to most efficiently learn the subject at hand. It's similar to the order I read them in, but enhanced based on what I learned and the order I wish I read them in. Good luck:

HTML/CSS:

CSS Mastery: Advanced Web Standards Solutions

<http://www.amazon.com/CSS-Mastery-Advanced-Standards-Solutions/dp/1430223979/>

Before you start coding PHP, Javascript, etc, understand how HTML works. This is where you start. HTML is easy. Read this book in combination with studying the HTML & CSS tutorials on w3schools.

PHP & MySQL:

PHP & MySQL For Dummies, 4th Edition

<http://www.amazon.com/PHP-MySQL-Dummies-Janet-Valade/dp/0470527587>

This book--well an older edition--I read a year before I got serious about learning to code. I read it and didn't actually code anything i learned, but what it did was plant seeds in my head with regards to what programming is all about and what databases are all about, and how to connect the two. It assumes very little in what you may

already know, and is an excellent start in your journey to becoming a master programmer.

PHP Object-Oriented Solutions

<http://www.amazon.com/PHP-Object-Oriented-Solutions-David-Powers/dp/1430210117>

This book is where I learned what OOP is. I didn't get the hang of it until reading the following book. Don't worry if you read this and have a hard time with it. This book and the next each have introductory chapters that go over how OOP works. It took me reading basically this book and the next book about the same stuff to get it. This book is a lot less complicated than the following and dives into practical examples & problems, whereas the next is a lot more theoretical.

PHP Objects, Patterns and Practice

<http://www.amazon.com/Objects-Patterns-Practice-Experts-Source/dp/143022925X>

After reading this book, I basically mastered OOP. It's a very hard book to get through if you're new to OOP, and goes into some very advanced stuff, specifically tons and tons of "design patterns." The design patterns are presented in as basic of a form as possible, but they weren't very practical like examples from the previous book in that you probably will never actually need any of the code used in the book. Either way, this is my favorite Programming of all time because it taught me how to think like a coder and how to solve complex problems with concise refactored solutions. It really showed me what is possible with OOP. You don't know PHP unless you've read this book.

Pro PHP: Patterns, Frameworks, Testing and More

<http://www.amazon.com/Pro-PHP-Patterns-Frameworks-Testing/dp/1590598199>

I read this book too just to solidify my experience with PHP, and cover all my bases. Check it out. It's optional.

PHP Functions Essential Reference

<http://www.amazon.com/Functions-Essential-Reference-Torben-Wilson/dp/073570970X>

At some point during my study of PHP I found this book and decided just to learn every PHP function available so that I could better understand the examples in the above books. Start reading this early on, and complete the whole thing. You'll quickly learn patterns in how PHP functions are named, and as a result be able to guess what a function does within the context of the examples in the above books--even if you don't remember precisely what it does.

Agile Web Application Development with Yii 1.1 and PHP5

<http://www.amazon.com/Agile-Web-Application-Development-PHP5/dp/1847199585>

I read this book in combination with reading the Blog Tutorial and Definitive Guide on YiiFramework.com. When you're done studying all these materials, you'll be amazed with how much power you have. This book isn't hard to read either. You'll love it if you reach this stage!

Javascript & jQuery:

Learning jQuery, Third Edition

<http://www.amazon.com/Learning-jQuery-Third-Jonathan-Chaffer/dp/1849516545>

jQuery is a framework built on top of the native browser language of Javascript. Usually one would recommend you learn the base language--Javascript--before learning an abstracted framework on top of it--jQuery. However because of the nature of jQuery and how comprehensive it is and because of how quirky Javascript is coming from PHP, I found it best to jump to jQuery and immediately start accomplishing the DOM manipulation tasks I needed. And ultimately because of syntax similarities between PHP and Javascript I was able to get productive in Javascript without studying a single book just on Javascript. One thing I did different

when studying this book from the PHP books is I did every single tutorial as I read it. The reason is because when I learned PHP, I was learning my first real programming language--so it took me a lot of time to just digest things before I could code a single line, which is why I just read PHP book after PHP book before I got started until it all made sense. However, by the time I got to Javascript & jQuery, I understood how programming in general works and found it helpful for memorization purposes to immediately start doing the tutorials.

jQuery 1.3 with PHP

<http://www.amazon.com/jquery-1-3-PHP-Kae-Verens/dp/1847196985>

With this book I didn't do all the tutorials like I did with *Learning jQuery*, but what reading this book did for me is taught me precisely how Ajax works and what it's all about. After reading it, coding features that required Ajax using Yii and PHP was obvious and a no-brainer.

JavaScript: The Good Parts

<http://www.amazon.com/JavaScript-Good-Parts-Douglas-Crockford/dp/0596517742>

This book gave me a deep understanding of the Javascript language and what it's truly all about. After reading it, many hours of debugging and head-scratching when coding Javascript & jQuery were removed from my schedule--because I finally learned the quirks of the Javascript language I needed to know.

Pro JavaScript Design Patterns

<http://www.amazon.com/JavaScript-Design-Patterns-Recipes-Problem-Solution/dp/159059908X>

Now this book took my Javascript game to the next level, gave me an idea of how jQuery was built, taught me how to do things similar to how you would in a "classical" OOP language like PHP, and completely ended any remaining head-scratching I was having with Javascript, particularly with how "scope" works in Javascript.

Linux:

The Official Ubuntu Server Book, 2nd Edition

<http://www.amazon.com/Official-Ubuntu-Server-Book-2nd/dp/0137081332>

Note: by the time I read this book I had already learned Linux through blogs on the internet. The best thing I can recommend you do is install Linux on your computer from the Ubuntu website, and start navigating around the command line, practicing Linux commands you learn off the web. Just google "linux tutorials" and you'll be off to a running start. That said, by the time I got proficient in Linux and after I read this book, I felt confident that I really knew what I was doing and had practical solutions for the most common problems you'll face at the command line.

Apache Cookbook: Solutions and Examples for Apache Administrators

<http://www.amazon.com/Apache-Cookbook-Solutions-Examples-Administrators/dp/0596529945>

This book I treat like a pocket reference and still refer to it often since it's impossible to remember all the different Apache configurations, given how comprehensive this web server application is. I did read it through when I first got it. I kinda skimmed it though-- just to get an idea of what is possible. Getting an idea of what is possible without mastering a subject matter is so important in programming because you'll know where to look when you face a challenge that the subject matter can solve.

Pro Bash Programming

<http://www.amazon.com/Bash-Programming-Experts-Voice-Linux/dp/1430219971>

This is a little advanced for readers of the *FaceySpacey Bible*, but I'm putting it here because it really took my Linux skills to the next level.

NON-TECHNICAL BOOKS:

Smart and Gets Things Done: Joel Spolsky's Concise Guide to Finding the Best Technical Talent

<http://www.amazon.com/Smart-Gets-Things-Done-Technical/dp/1590598385>

If you plan to grow a small application into a large company, this book is a must. It's short. Read it.

SEO Book.com

<http://www.seobook.com>

This obviously isn't a book, but I read their entire site like a book, and its creator, Aaron Wall, expects you to read it like a book. When I was done reading it, I felt I was completely up to speed regarding what SEO is, how search engines work, and practical techniques to get better rankings in search engines.

For daily Startup Wisdom, checkout FaceySpacey.com/blog daily. And don't forget to download the entire No Bullshit FaceySpacey Bible or more individual chapters here:

<http://www.faceyspacey.com/resources?section=book> .

Á

V@) | • Á* aÁ [{ Áœ^ ^ Û] œ^ ^ Á) áÁ^ Á ~ ^ Á Á & @ & Á ~ of FaceySpacey.com Á - e } Á
{ ! Á ~ ! c @ ! Á } [, | ^ á * ^ Á ^ Á & ! Á Á œ ^ Á [~ ! Á Ů œ c } Á Á @ Á Ů œ • Á

Á