Positioning – what the hell is it? And why is everyone talking about it?

The goal of this template is to help you figure out how to effectively position your product for developers or data teams, and as a consequence be able to write better website copy, pitch candidates better, give more effective demos, improve your documentation, etc. etc.

What positioning is and why it's important

The best definition of positioning I've found is this:

> *"Positioning defines how your product is a leader at delivering something that a well-defined set of customers cares a lot about."*
>
> – April Dunford, positioning...expert or something

This is not just a marketing exercise, because as a founder your job is literally to talk about the product all the time. Think: giving demos, sales calls, pitching candidates, creating your website, writing blog posts – pretty much anything external facing is going to require you to articulate your product's value and place in the market. How confident are you that you're doing it well?

Couched in the above meaning-packed sentence is a lot of stuff you have probably been thinking about:

- Who is the audience for my product?
- What features do they care about?
- How do I differentiate from alternatives?
- ...

Setting aside time to run through these questions, debate with your team, get feedback from experts, and the like will make everything you do in the early stages a lot easier. So how exactly do you go about doing that?

Positioning is usually a game of **choice** – you'll have a few different ways you can tell your story, but eventually you are going to need to commit to one[1]. For an example, consider a

[1] Early on, it's cool to experiment with different positioning strategies. But eventually, you'll want the whole company on the same page with who the audience is and what you provide to them.

product that **gives you visibility into how your Machine Learning models are perfo**rming. What's the right way to position and message that product?

- ML monitoring for engineers
    - For ML engineers who want confidence that your models are performing well in production, our product provides flexible, extensible, and easy-to-use infrastructure for model observability.
- Optimizely for ML
    - For ML leaders who want to build great products powered by ML, our product provides tools to debug, improve, and personalize production model performance.
- Production data platform for ML engineers
    - For busy ML teams tired of spending time wrangling data through fragmented tools, our product provides a data platform for your production models that lets you focus on building models, not managing your data.

These all sound pretty good. And they all define slightly different audiences, angles, contexts, and value propositions. So let's dive into how to define positioning, and ultimately decide on a way of doing it that works.

When do I need to start thinking about positioning?

Pretty much from day 1. How you position and set context will determine which users think you're relevant to them, so even if you don't have any yet, you can't ignore this question. In fact, running through the positioning template below can be a useful exercise for clarifying how you're thinking about your idea, even in the idea stage.

Positioning is also an evolving thing. How you position yourself on day 1 will probably be different than year 2, as you gather more feedback from the market on what's working and what isn't.

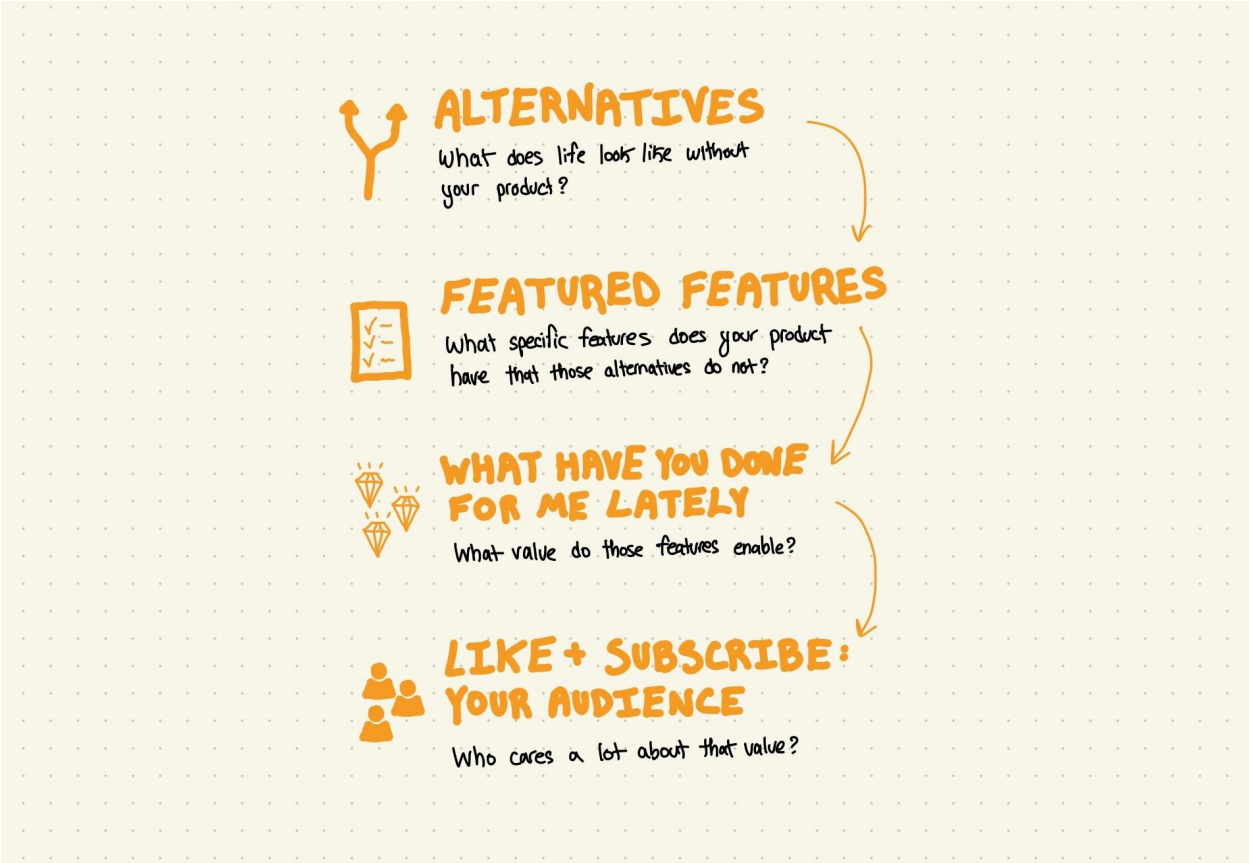
What is the output of positioning?

The output of positioning work is not just a positioning statement or a headline on your website: rather it is a more holistic understanding of your context in the market, and how to talk about what you're doing in a compelling way to your audience. This "understanding" can take form in a document going in depth on your positioning, a set of taglines and answers to questions, or even a recurring presentation to the company explaining how to pitch the company. It is a lot deeper than a good website headline.

In the realm of the sales deck, Andy Raskin's post about positioning in the context of *change* is useful and worth reading.

Defining positioning for technical tools

Here is a (hopefully) useful framework for figuring out your positioning, based on April Dunford's popular one, with a technical bent. It focuses on figuring out your **product context** through getting deep on alternatives, what features are important, what value they enable, and who that value is important to.



Each of these feeds into one another, so it's important to do them in order. We'll use Hex as a walkthrough example.

Alternatives

You probably built your product because you were fed up with whatever life looked like without it. But what exactly did that life look like? Recall the trauma!! If you didn't exist, what would customers use?

- Building things from scratch, plain JavaScript, etc.
- Popular open source frameworks
- Big, clunky ancient products
- Underpowered startup products

E.g. before Hex, data teams built and shared work through an awkward lacework of products. You have Jupyter notebooks for running Python, but they didn't quite play nice with SQL; sharing your work required setting up a server or sharing a literal notebook file. Frameworks like Shiny (if you're using R) would help set up basic frontends, but data teams don't know Flask or how to build UIs. With the advent of products like Mode, there are now more places for data teams to work together; but these tools don't focus on sharing knowledge, which is what effective data teams are all about.

It is *really* worth it to **get into specifics** here. With technical audiences, having a firm grasp on your alternatives can really make them feel like you understand them. And even if *you* have that grasp, does everyone at your company have it? Do your salespeople have it? Can they articulate why it's easier to build internal tools in Retool than it is in React? Can they mention Redux in passing so that the person on the other side of the Zoom feels understood?

Finally, a lot of founders start thinking about positioning because a new **startup competitor** is making them feel insecure about their unique value proposition. This could be a startup that launched after you did or around the same time. A startup competitor is distinct from an open source alternative, a home grown solution, or a big incumbent competitor.

Your startup competitors are part of the universe of alternatives! So you need to position yourself in a way that's differentiated from competitors. Unfortunately, you won't always know what your competitors' products are like or how they position themselves, and it will be constantly changing. So unless you are getting explicit questions about it from customers, in the early stages I would put this on the backburner[2]. Positioning vs. competitive alternatives is always critical; positioning vs. startup competitors is not always critical.

Featured features

[2] This is obviously an oversimplification. Another post for another time.

What are the specific things that you can do in your product that make it much better than those alternatives you just laid out?

Without Hex	**With Hex**
Connecting your data warehouse via an ORM	1-2 clicks connects to any major data warehouse
Awkward SQL in notebook cells	Autocomplete, one click formatting, and an inline schema browser
Python in a notebook, SQL in an IDE	Write Python and SQL in the same environment, and even pass data between them
Saving different notebooks for version control	Git synced version control and release history in a slick UI
…	…

For this section, don't worry about the value (this makes me faster, etc.), just focus on the specific features and design decisions that make using your product better than the alternatives.

Which features you choose to…feature have an important impact on the context you're setting for your audience. Returning to our example ML monitoring solution above, which features we focus on tell and entirely different story:

- Fine grained alerts and thresholds → communicates that this is a monitoring / observability solution
- Powerful charts and funnels → communicates that this is an analytics solution / data platform

Your product probably has a lot of cool things you can do with it, but you need to pick the important ones that best position you against alternatives. There will always be opportunities to market the other ones down the road.

What have you done for me lately? Value

If you had to describe your product's value proposition in 1-2 key pillars, what would they be? A value proposition explains how using your product will improve your users' lives in a particular way. In technical tools, there's a pretty limited world of value props:

- **Speed** – this saves me time
- **Simplicity** – this makes my life easier. This removes mental clutter
- **Power / flexibility** – this gives me a power up, this lets me do things I otherwise couldn't do
- **Impact** – this helps me accomplish something I'm struggling to accomplish, this makes me more impactful at my job

You can get more specific too, but they'll always roll up into making your user faster, better, stronger:

- **Integrations** – this works with my existing tools
- **Security** – this covers all bases and will improve our perimeter

For each one of these pillars, you need to think about which specific features and design decisions make your product actually fulfill this promise. For developers and data teams, just *telling them* that your product will save them time isn't enough; you need to explain and show them how. So take your outputs from the last section and map them to the value you're bringing.

Over the past few years, a common value proposition in technical tools has become "*stop spending time doing _____ so you can focus on what matters*" where "what matters" is something like core feature work. This value proposition is really at its core somewhere between saving time and simplicity, i.e. don't waste your time on something you could just automate.

*E.g. Hex makes it incredibly **fast** and **simple** to create powerful analysis right in your browser. You can connect your data warehouse (we support some wacky ones) in just a few clicks and get straight to writing SQL. Our SQL editor has thoughtful features to make your queries easier to write: autocomplete, one click formatting, and an inline schema browser so you know the column is firstName and not first_name.*

Like + subscribe: your audience

The final and perhaps most important part of technical positioning is **the audience**. If you ask most technical founders who their audience is, they'll say something like:

“We sell to developers”

Sorry, but that is not an audience. It needs to be more specific! Every company in infrastructure is selling things to “developers.” Here are two ways to be more specific:

1. **Demographic specificity**

Getting more specific demographically could mean cutting your audience by a combination of:

- *Role* – full stack, frontend, SRE, DBA, etc.
- *Company size* – startups, enterprises, etc.
- *Industry* – healthcare, fintech, etc.
- *Focus* – ambitious, security conscious, cautious

You will likely need to choose multiple of these for the audience to be specific enough. And even then, it’s hard to know this information in advance of going out to market and figuring out who cares about your product. That’s why I like early audience definition by **problem-context** instead.

2. **Problem-context specificity**

With problem-context specificity, you define your audience by a specific problem they’re experiencing at the current moment.

E.g. Hex is built for data analysts and scientists who are struggling to build and share impactful analysis.

Or:

E.g. Retool is built for developers tasked with building an internal tool for a customer support, marketing, or operations team.

I find this framing useful because it *really* sets context with your audience, and it helps dictate the kinds of content you’d want to create down the road. It turns out that it can be easier to target “developers facing a specific problem” than “developers of a specific kind” sometimes.

Combining the two can work well too.

E.g. Hex is built for data teams at small to mid sized companies with cloud based data warehouses and dbt projects who are struggling to build and share impactful analysis.

How do I get these answers? Viewing positioning as a set of experiments

Ah yes, the hard part. For some of these positioning questions, you'll have a good idea of the answer, usually taken from some combination of:

- Your experience as a former target users of this product
- Talking to potential customers and seeing what sticks
- If you have existing customers / partners, their use cases

For others though, you won't be quite sure: which is why early on, you should **view positioning as a set of experiments**. It's impossible to know for sure what messaging and positioning will resonate most with potential customers, hires, etc. – so you can use every opportunity to crisp up that messaging and see where it falls flat.

If you have a few ideas of what positioning might work, but you aren't sure, try experimenting with using different ones when pitching a new candidate to join or giving a demo to a potential new customer. You probably *already do this*, to a degree. But being intentional about it, and maybe even tracking things methodically, can go a long way.

Your customers will also tell you how to position if you listen.

> *"The thing a lot of founders miss is listening vs. talking. Your users know what their pain points are, and they will tell you all about them if you come at it the right way. Many founders (me included), however, will try to go in their cave and craft the perfect positioning statement in a vacuum. For our messaging stuff now, we insist on always bringing real customer and user quotes into it. If we haven't heard it from a user, we can't use it in our marketing."*
>
> – Barry McCardel, Hex CEO

You don't always want to take what your users say at perfect face value, though. Often customers will ask for a specific solution to their problem which might not actually be the best way to solve it (better horses vs. cars, etc.). With a focus on problem definition, sometimes you need to see past what they're saying to hear the "real" pain. Each customer is going to give you new information, make sure to look at this feedback in aggregate and not make huge shifts based on what one prospect tells you.

Positioning template

	Your Answer	*Hexample*
Competitive Alternatives If you didn't exist, what would customers use?		*Without Hex, doing analysis requires an awkward combination of Python in home grown notebooks and SQL in IDEs. Sharing your work ends up as static screenshots of Excel charts in a presentation.*
Key Unique Attributes What features / capabilities do you have that alternatives don't?		*Hex lets you write SQL and Python in a beautiful cloud-based notebook interface. You can connect your data in only a few clicks, and easily build interactive data apps on top of your work with a drag and drop app builder.*
Value What do the attributes enable for customers?		*Hex brings your analysis and your sharing into the same platform. You'll have a significantly easier time analyzing data, and actually sharing it with your organization so you can have an impact. It turns your work from academic to actual.*
Customers That Care Who cares a lot about that value?		*Data teams at small to mid sized startups, using a cloud data warehouse, working primarily in SQL and Python, focused on making their work impactful throughout the organization.*