



Build lists

Share with Friends

Create games

Rosterizer.com

"Battlescribe's successor? Rosterizer"

—Der Magabotatos

<https://www.magabotato.de/2021/12/248-stammtisch-november-2021>

about us

Rosterizer is a list management app that picks up where other roster builders leave off, providing a rich cross-platform editing experience and robust data authoring that supports any game system.

points	quantity	name		level	defense	attack
12	1x	Brutal Warlord	⊗	☆3	🛡️2	🗡️3
6	1x	Beastmaster	⊗	☆2	🛡️2	🗡️2
3	1x	Runt	⊗	☆1	🛡️1	🗡️1
9	1x	Blood Shaman	⊗	☆3	🛡️2	🗡️2
30 points, 4 figures						

Inception

Rosterizer was born from a desire to make our own tabletop games that would each be accompanied by an app to handle roster building. We made a very basic webapp to track a simple skirmish game, then altered it for a more complex pseudo-RPG. After a few iterations of rules changes, we realized that it would be more fruitful to develop a method of interpreting any arbitrary game rule we wanted to define.

Proliferation

The plan was always to provide a way for other users to create and share their rosters, but once we had a roadmap to be able to support any game rules we could write, we realized it was a small step to be able to interpret ANY game's rules. Small step... right. Fast forward three years later and we're finally seeking crowdfunding capital to take us through the final steps before we publish.

The screenshot shows the Rosterizer app interface. At the top, there's a header with a '1x' icon and a list of figures. Each figure has a 'Points' value, a 'Figure' name, and several attributes: 'Level', 'Attack', 'Defense', 'Advance', and 'Range'. Below the header, there's a 'Description' section for each figure. The first figure is 'Steve' with 66 points, Level 5, Attack 5, Defense 5, Advance 6, and Range 1. The description for Steve says 'Include: Ability'. The second figure is 'Beatrix' with 20 points, Level 3, Attack 1, Defense 2, Advance 4, and Range 1. The description for Beatrix says 'Rank[1]: This figure ignores impassible terrain during its advance and...'. The third figure is 'Flying' with 8 points, Rank 1. The description for Flying says 'Rank[1]: This figure ignores impassible terrain during its advance and...'. The fourth figure is 'Non-lethal' with 2 points. The description for Non-lethal says 'This figure's attacks never use more than one die result for purposes of...'. The fifth figure is 'Launch' with 2 points, Rank 1. The description for Launch says 'Rank[1]: This figure can use its activation to throw friendly figures...'. The interface is dark-themed with orange accents.

the team

Kevin Duda

Principal Architect

Joselin Nute Lybrand

Lead Back-end Dev

James Cormack Robinson

UX Director

sample questions

(paraphrased from conversations about Rosterizer on various platforms)

What are Rosterizer's main advantages over its competitors?

- Dynamic (editable) stats, enabling the kind of tracking common in skirmish-style games or campaign play modes
- A powerful and extensive rule/conditional system that makes it easy to evaluate complex list states and raise errors
- Sync rosters between various device platforms
- Share rosters with friends, or clone theirs to make improvements
- Multiple view and output options (card view, outline, text list, etc...)
- Edit history
- Simple roster format makes updating stats via new manifests trivial
- Rename and describe any asset, as permitted by the manifest

What's the current state of the project?

We've finalized the core functionality and are working on stability and polish ahead of our upcoming open beta test.

You've been working on this for a bit, why publicize now?

We were in stealth mode until about a year ago. Partly because we wanted to make sure we had something that could be viable before we started showing it off everywhere (no point in saying we're working on something and then a year of silence while we tinker). So far, the demos we've given have been really positive and we're pretty sure we have something great on our hands.

Would you also have a web version?

It'll all be a web version (ideally with a serviceWorker-powered offline mode) until we get around to launching something for the app/play stores. And even then it'll probably just be the webapp but wrapped in an app container like electron (haven't looked too deeply into it yet).

If this is a web app, will I be able to use it without an internet connection at my events?

We plan to have an offline mode. The app functions by processing the data on the client's side, which will be provided through our data syncing service and local storage. Being able to store and process everything locally is also important for free users who aren't syncing their data but still need to be able to have all the functionality.

sample questions

How much processing does the app do?

We're pre-processing the manifest, then composing the roster (a bare list) with all of the relevant asset data in the manifest, and then processing all the rules relevant to the included assets. This all takes less than a second for most data, and we're only doing it up to a few hundred times (depending on how much data the user has stored) initially. So we're actually kind of doing a fair bit of processing but we've taken several passes at optimization to ensure we're not over-cranking peoples' hardware.

Can my phone run this?

We've been testing with phones with varying processing power, and so far the app has been running well. Supporting mobile has been a high priority throughout development.

An issue with the current data is time for updates, but if the display for data on web format is available and pulling from data source you guys have, then it would be fantastic.

Given the litigious nature of some gaming brands, we're not planning to host any public data ourselves, that we're not confident we have permission to host. We had some data hosted during closed beta to facilitate testing, but now that it's time for a wider audience, we've wiped anything that's not free to use. Data access will be very much like the BattleScribe method, with a few tweaks to play it safe and also make it easier to find the data you want. For instance, if you load a shared roster that uses data you don't have, you'll see all of the asset names without details, and will be able to assign a data source of your own (if you have it in your account) or search one up.

Is the data just going to be hosted on Github or some other source?

We'll host **your** data, either data that you authored or that you imported, just for you and tied to your account. We'll support importing JSON from clipboard, local upload, or (probably most popular) web URL. We've also implemented a monitoring system to alert people when their URL data sources have updated. For finding data, we plan to implement some kind of guided searching to produce likely results to streamline the experience.

Is someone else going to house that data and then they import it into the system?

Our method will be to permit importing from any URL that provides a JSON in the right format. I imagine github will be the primary solution, but we should also be able to support importing from files hosted any random place, or even from an API if someone wants to set one up.

sample questions

Are you reaching out to the community about building something based on this data format?

As far as the game data goes, part of that is happening during the closed beta. There also exists an eventual potential for third party apps to ingest roster data in order to process it, whether that's for math-hammer style calculations, statistical analysis, alternate outputs, or importing into other apps like Tabletop Simulator.

How will you allow the existing BSData people to switch easily?

We're chatting with folks about how best to handle data translations. We allow CSV importing of bare stats and have also created a BS-to-Ros translator that does its best to extract stats from Battlescribe files. If neither of those fit the bill for a particular game, let's face it; this hobby is rife with nerds who like to put in elbow grease on a project they enjoy. We just have to make sure we've provided a product that they can't ignore.

The 40k tournament community would be a good place to start testing.

We've seen a great deal of engagement from 40k players who are afraid of losing their favorite tool (since the Battlescribe abandonment/delisting scares has everyone shaken), or just want more than Battlescribe can offer; specifically Crusade support but also quality-of-life improvements like changing detachment types without having to re-build.

How will data updates be handled?

We currently track the import source of each of your manifests and offering to update if it detects a newer data file (as well as offering to retain your old file in case things get broken). Regardless, our roster format is decoupled from the data such that you're able to apply ANY game data manifest to any roster without modifying the roster's structure. You could assign any random data to a roster and it would complain about every asset, but would still list them all where you left them. This means that rosters won't break when data is updated unless that data really changes the structure of the game, e.g. if detachments were suddenly handled very differently, and we're also looking into smarter ways to handle things when that kind of breakage DOES occur.

Inquiries

media@rosterizer.com

Web

rosterizer.com

Discord

[@GameKnave](https://discord.com/invite/@GameKnave)

Reddit

reddit.com/r/rosterizer