

ADVERTISEMENT

in: [Updateme](#), [Policy](#), [Design Documents](#)

SIGN IN



REGISTER

WARFRAME

Wiki:Maximization Calculator



EDIT
SOURCE

Design document for improving the [Maximization](#) calculator for [Abilities](#). Currently they are informally sunsetted and will not be frequently updated or have continued maintenance on them.

Readers can discuss this topic on [WARFRAME Wiki talk:Maximization Calculator](#).

Last updated: Fri, 22 Dec 2023 21:10:00 +0000 (UTC) by [User:Cephalon Scientia](#)

☰ Contents

1. Current Implementation
2. Issues
3. Alternative Solutions
 - 3.1. Module Database Model
 - 3.2. Modifying Static Values On Ability Articles
4. Discussions



Current Implementation

Current calculators use [Template:WFmax](#) for adding input form (`<div id="maximization_table_{{{1|}}}">`) and output display (`<div id="maximization_{{{2|}}}">`) HTML and JS script on [MediaWiki:Maximization.js](#) for interactive elements. All of the relevant ability data used for the calculator is stored on [MediaWiki:Maximization.js](#) which means that non-admin editors cannot readily update values without going through an administrator.

Issues

- Ability data is not readily accessible by readers and editors, violating the principle of [open data](#) that we strive for as a publicly editable wiki.
 - Thus it is not scalable to apply to new content or changes to WARFRAME. Often times calculators are outdated because the people who care cannot update these values or unsure where to update them.
 - This issue goes beyond the calculator project and applies wiki-wide. There is a high barrier to entry in wiki editing, especially on this wiki, due to the usage of complex templates, Lua scripts, and data that go way beyond simple text markup. Think there needs to be a community-wide effort to upskill readers to become editors, but that implies a job function rather than a hobby activity. Wiki editing is time consuming and not a traditional medium for entertainment. It takes a special type of individual to be passionate about information sharing, community collaboration, and public service.
- Bloated script. On any given ability article, all of the additional ability data will never be used, using up reader bandwidth.
- No support for [internationalization and localization](#); cannot scale to non-English users.
- Doesn't explore the use of [data-* attributes](https://developer.mozilla.org/en-US/docs/Learn/HTML/Howto/Use_data_attributes) (https://developer.mozilla.org/en-US/docs/Learn/HTML/Howto/Use_data_attributes) to store resultant ability data. Could potentially help with scalability.
- Vanilla MediaWiki does not have a good infrastructure for developing interactive widgets or applets, especially those that interface with databases based on the open data model.
 - Calculator should probably exist as a separate popup instead of being embedded within article content. An overlay window that's dedicated to the



<https://www.mediawiki.org/wiki/Extension:Gadgets> and <https://www.mediawiki.org/wiki/OOUI/Widgets>.

Alternative Solutions

Module Database Model

There was an attempt at porting some of [MediaWiki:Maximization.js](#) data to a module [Module:Maximization/data](#). Though there are few issues with this approach:

- 21:08, 22 December 2023 (UTC) update: We can store JSON native data on pages that have the JSON content model like [NightwaveActs.json](#). There is now native support to read JSON data from pages within Lua modules via `mw.loadData()` which will allow data to be accessed in Lua.
- How would JS interface with data stored in Lua tables? There is currently no infrastructure in place that allows the conversion of Lua tables to JSON.
 - Would have to import a Lua to JSON converter before being able to access the data in JS. This means that the client would have to do lots of overhead work before being able to perform any calculations:
 - First, they have to convert the data to a JS-readable format.
 - Second, they have to interpret strings that store formulas to callable functions or arithmetic operations through some awkward RegEx or string manipulation.
 - A creative solution would involve using [MediaWiki's Action API](#) to fetch the string JSON result of a Lua module function call that converts Lua tables to a JSON string through [Module:JSON](#). Client JS would still have to execute a `JSON.parse()` (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON/parse) to read the JSON string.
 - Unsure if we can cut the middleman and just store data as native JSON like with [NightwaveActs.json](#). Action API doesn't seem to have the option to fetch page content when the content model is in JSON. Would have to fetch JSON using query param `?action=raw`. Still issue of loading more data than necessary when executing script.
 - There is a Lua VM implemented in JS to allow Lua code to be interpreted in client-side JS: <https://github.com/fengari-lua/fengari> and <https://github.com/fengari-lua/fengari-web>. Unsure if this can be used in a



- Is there a universal data schema that can apply to all abilities in the game despite their complex interactions and implementation with ability attributes and augments?
 - Explore [Polish notation](#) when defining formulas? [Lisp-like](#) expressions?
 - Vanilla JS has a [Function\(\) constructor](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function/Function) (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function/Function) that we could use to store calculation logic in data. This way we can have custom editor-defined formulas to calculate stats.

Modifying Static Values On Ability Articles

We can have the calculator modify static values that already exist on all ability articles within [Template:AbilityU10.3](#) arguments. A quick, hacky approach would to use RegEx to extract number values to be manipulated by the calculator. Though this wouldn't work for stats that deviate from the multiplicative calculation ($\text{Final Stat} = \text{Base Stat} * (1 + \text{Ability Modifier Bonus})$). Would have to explicitly define those calculations somewhere, either on the article (within HTML attribute) or in a separate centralized location (like a Lua module).

This approach would store raw ability data on individual ability articles (within templates) instead of having them in a centralized repository. Ideally, we should have both raw ability data and ability stat formulas in one location for editors.

Discussions

- [User blog:TunaInABottle/Interactive Ability Maximization](#), original 2019 blog post on the release of the feature
- [User:FINNER](#) (2021, July 29). *Do you guys even use the interactive ability/augment Maximization stuff?*. Fandom. Archived (<https://web.archive.org/web/https://warframe.fandom.com/f/p/4400000000003490648>) from the original on 2022-09-29.
- https://warframe.fandom.com/wiki/Xata%27s_Whisper?commentId=4400000000003652372&replyId=4400000000015019561 - [User:Cephalon Scientia](#) response to ""Result table not loaded, Javascript not loaded" - So...are we just not allowed to use the maximization portion anymore or what? I doubt this is my browser, it's been like this for months now. I'd love to be able to, well, min/max and all."

WARFRAME Wiki [Policies](#) and Meta Wiki



	Language • Projects • Research • Styling Guide
Community	Assume Good Faith • Banning Policy • Chat Policy • Discord • Onboarding • Protected Pages • Talk Page and Comment Policy
Development	Archiving • Audio Files • Bug Reports • Content Automation • Creating New Pages • Data Migration • Development Guide (JavaScript) • Glossary • Localization Guide (L10n Messages) • Modules (/List) • Porting PublicExport • Public Endpoints • Programming Standards • Templates (/List) • Updating Databases • Updating Infoboxes • Updating Tooltips
Design Documents	Damage Terminology • Data Compression • Expected & Nearly Guaranteed Numbers • Lua Error Handling • Maximization Calculator • Stat Comparison • Stat Display
Business Artifacts	Administrators (organizational/leadership structure) • Career Progression • Comparison of Hosting Services • Contingency Plan • Cost Analysis • Desired Features • Licensing • Procurement • Project Vision and Scope • Solution Stack • Stakeholder Analysis
Sandbox	Special:ApiSandbox • MobileViewTestPage • Sandbox • Special:ExpandTemplates
Metadata	Copyrights • Duplicate image script • ParentPage • Site Maintenance • Special:SpecialPages • Special:Statistics • Special:Version • Special:WantedPages (current redlinks)
Meta	Official Wiki • Surveys • Wiki Forks and Mirrors • Wiki Usage In Media
Fandom-Specific	Customization Policy • DMCA Policy • External Platform Guidelines • Removing Ads • Wiki Forking Policy
Other	Overframe
Activity Feeds	Special:RecentChanges • Special:SocialActivity
Change Logs	2012 • 2013 • 2014 • 2015 • 2016 • 2017 • 2018 • 2019 • 2020 • 2021 • 2022 • 2023 • 2024
	Local Sitemap

Categories

▼

Community content is available under [CC-BY-SA](#) unless otherwise noted.

More Fandoms

Sci-fi

Warframe

