Chore Wheel

Release 0.3

Daniel Kronovet

Apr 24, 2024

OVERVIEW

1 Contents **3**

1.1 Introduction . . . 3
1.2 Getting Started . . . 4
1.3 Chores . . . 5
1.4 Hearts . . . 10
1.5 Things . . . 14
1.6 Conflict Resolution . . . 17
1.7 Monthly Circle . . . 18
1.8 Design Principles . . . 19

Chore Wheel is pioneering software for coliving, built by Zaratan and running on Slack.

Created by **economists** and **game designers**, Chore Wheel is a family of tools helping people share space:

- *Chores* for keeping it clean
- *Hearts* for mutual accountability
- *Things* for bulk purchasing
- and more to come

Chore Wheel is **open-source** and **privacy-preserving**, and contains the latest thinking in **ethical technology**. Chore Wheel draws influences variously from **cognitive science, computer science, electoral theory, economics, cybernetics, and game design**, with four top-level *design principles*:

- No managers or privileged administrative roles
- Simple and intuitive inputs
- Humans for sensing and judgment, machines for bookkeeping
- Continuously available, asynchronous processes

Groups using Chore Wheel find that they spend **less time doing chores**, **less time talking about chores**, and **less time worrying about their house in general**. Check out the *introduction* for a more detailed overview of the project, or our *getting started guide* to get set up for yourself.

Tip: For extra help and support, join the Chore Wheel Community Slack workspace.

CHAPTER

ONE

CONTENTS

1.1 Introduction

Note: While initially designed for large coliving houses, Chore Wheel works just as well helping groups of friends or even couples navigate shared space.

As housing costs continue to climb, the development of quality, affordable housing remains a continual challenge. It is increasingly apparent that **coliving** - housing options where residents have private bedrooms but share common spaces like kitchens, bathrooms, and living rooms - is an important part of the solution.

Perhaps the greatest benefit of coliving is the avoidance of redundant infrastructure (e.g. one large kitchen, rather than three small ones), which lowers costs. However, sharing common resources introduces new coordination challenges (e.g. "who does the dishes"). Historically, such challenges have been overcome through informal norms (e.g. a "dish-zero" rule), deliberative decision-making processes (e.g. house meetings), basic coordination mechanisms (e.g. paper chore schedules), or empowerment of administrators (e.g. house managers). While effective to various degrees, such solutions are too unreliable, burdensome, expensive, and simplistic to consistently meet the needs of a large and varied population.

As Oscar Wilde famously quipped, "the trouble with socialism is that it takes up too many evenings." **We can do better.**

Chore Wheel is *poetic technology*: a family of tools meant to support the healthy functioning of a coliving environment, by **maximizing participation and minimizing dysfunction**. By helping communities handle their most common problems almost "by magic," Chore Wheel lets people channel their time and energy towards more long-term, meaningful projects.

Note: Chore Wheel does not claim to capture all ideation, decision making, and deliberation occuring in a coliving environment. Nor does its use reduce the need for *ongoing investments* in community-building. Rather, it takes its cue from the Pareto principle: a set of simple, general tools which handles the most common 80% of scenarios, leaving the remaining 20% to be handled by people on their own.

1.2 Getting Started

Getting started with Chore Wheel is quick and easy, and you can be up-and-running in twenty minutes.

Tip: It is recommended that everyone be physically present during the set-up process. Perhaps over brunch?

1.2.1 Setting up the workspace

First, designate one person to be the **coordinator** for the process. If the house already has a Slack workspace, they should be one of the admins. If the house doesn't have a Slack workspace, they should create one now. Chore Wheel works equally well on free and paid Slack plans.

Note: One of the design principles of Chore Wheel is "no managers or privileged roles." In practice, allowing a small number of privileged actions makes a few things much easier. These actions are available to **any admin** in the workspace. If your community would prefer that everyone can take these actions, they should make everyone an admin.

Once your house has a Slack workspace, please invite everyone to join, and wait for them to accept their invites.

1.2.2 Setting up the tools

Once everyone has join the workspace, you can begin installing the tools. Once you install each tool, the app home page will give you specific instructions on getting up-and-running.

- Install Chores
- Install Hearts
- Install Things

All tools function as stand-alone apps, but complement each other when used together. There is a rough progression of power/complexity from using *Chores* by itself, to combining it with *Hearts*, to combining both with *Things*.

When using *Chores* as a stand-alone tool, you get all the power of the chore system, but without a long-term accountability structure. By adding *Hearts*, you get long-term accountability both for chores, and the ability to encourage and discourage specific behaviors. By adding *Things*, you get some simple tooling for managing shared funds, useful for larger groups.

The right mix of tools will depend on your specific needs. If you're just exploring, try getting started with *Chores* and *Hearts*.

Tip: If 100% of the workspace votes yes for a proposal, it **passes immediately**. This is useful if folks are gathered in-person and want to get up-and-running quickly.

Warning: **It is normal if initially things are not making sense or working as you expect.** These tools are *dynamic*, so their state evolves over time. People also need time to try out the systems and learn how they work. It usually takes a few days for things to settle into a steady rhythm.

1.3 Chores

Install Chores

A clean and organized house is a shared resource. We all consume that resource as a part of our day-to-day, and we all play a part in ensuring that resource is renewed. We all make messes, and we all help clean them up.

Chores is an organizational tool, used to help structure individual contributions to the preservation of house-as-a-resource. Chores is designed with two goals in mind: first, to ensure an even distribution of domestic labor; second, to give people as much freedom and flexibility as possible in how they meet their obligations.

> **Warning:** It is tempting to shy away from structure, and rely on spontaneous generosity. But spontaneous generosity never lasts, and a few people, usually women, end up doing most of the work.
>
> Alternatives, like physical chore wheels or schedules, have their own drawbacks. Cameras and cleaners have their place, but are expensive and can create a weird vibe.

The core concept of Chores is, unsurprisingly, chores. Every house creates a list of chores that they need done. The key idea is that chores aren't done on a schedule, but rather "whenever necessary", as determined by the residents. Chores are worth points, and the amount of points a chore is worth goes up the longer the chore goes undone.

Everyone in the house needs to earn a certain number of points, about 100 per month. Over the course of the month, all the chores (collectively) gain about 100 points per person. Everyone does the chores they think are worth their time, and the result is a continuous process of domestic renewal.

Note: Here is the key dynamic: everyone would rather do a chore for more points, but if they wait too long, they'll get scooped. So everyone independently make choices about what to do, and when, and how, based on the ever-changing situation on the ground.

Doing chores bit-by-bit over several weeks? *Have fun*.
Waiting until the last day? *Good luck*.
Doing 2 hard chores for 50 points a pop? *Why not*.
Doing 20 easy chores for 5 points each? *Go nuts*.

It all works. There is no right or wrong way to participate, although waiting until the last day is *probably* the worst strategy.

1.3.1 Quickstart

1. Set an app events channel
2. Make a list of 3-5 starter chores/descriptions
3. Enter the chores using `Edit Chores List`, and upvote them in the app channel
4. Wait 1-3 days for the chores to accumulate points
5. Encourage people to check current chore values and to `Claim a Chore` which is well-valued
6. Encourage people to `Set Priorities` if they feel that a chore is being over/under valued

1.3.2 Core Concepts

Chores

The core concept of Chores is, unsurprisingly, chores. A chore is any pre-defined task which people usually won't do on their own. **An ideal chore takes at between 5-30 minutes to do, is largely self-contained, and can be done more-or-less at any time.**

Note: Here is an example of a well-defined chore:

Dishes

- Put away all clean dishes from the drying rack or washing machine
- Put any dirty dishes in the washing machine
- Clean any large pots or pans by hand and put in the drying rack
- Run the washing machine if full

Tip: Chores don't *have* to be limited to literal chores like doing dishes or sweeping floors. Rather, they can include any type of task which folks want to be encouraged to do.

Groups have used Chores to encourage things like food prep and house dinners, as well as movement classes and studio time. It all comes down to what your group wants to collectively prioritize, and the trade-offs it wants to make.

Points

Everyone owes **100 points** per month. When someone does a chore, they earn points. The amount of points a chore is worth is not fixed, but grows over time. When someone claims a chore, the chore's value goes back to zero.

Note: A key idea is that people can't "vote" to give points. Rather, points accumulate slowly and automatically over time. This keeps points roughly correlated to how messy something is, while avoiding meetings and politics.

Priorities

Chores can be assigned different *priorities*, since not all chores are created equal. The higher a chore's priority, the faster it will gain points over time. Priorities are zero-sum: for something to gain priority, something else must lose priority. High-priority chores are usually more intense (bathrooms), or need to be done more often (dishes), or both. Low-priority chores are usually simpler and less critical (sweeping, tidying up common space).

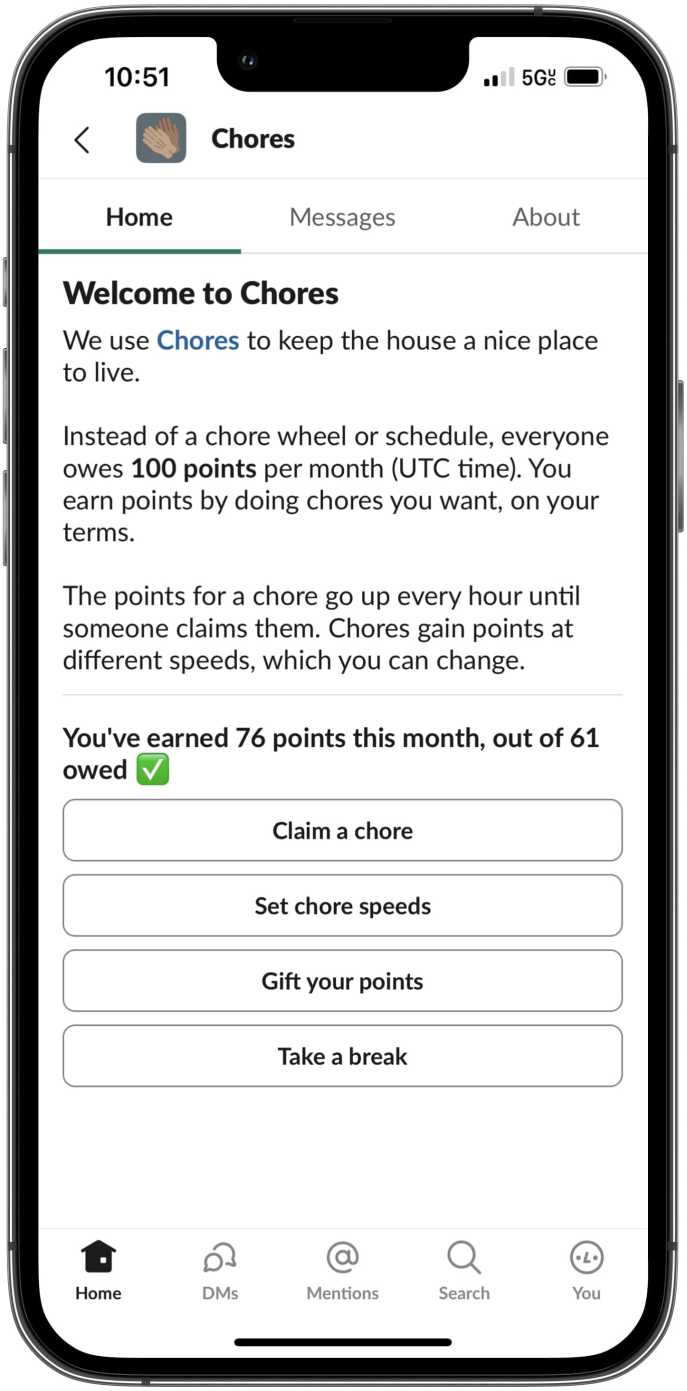
Breaks

If you go out of town, you can take a break from chores. Taking a break means you'll owe less points that month.

Gifts

If someone does something helpful that isn't a "chore", but you still want to recognize it, you can gift them some of the points you've earned. Gifting points also makes it possible to "split" chores.

1.3.3 Basic Functionality



The Chores home page is the chores dashboard. On the home page, folks can see their current and owed points for the month, as well as how many people are around that day (i.e. not exempt and not on break). The app home is also the entryway into the basic functionality, described below:

Claim a chore

When someone does a chore, they "claim" the points that chore is worth at that moment. The claim is then posted publicly, and others can verify that the claim was made honestly. A minimum of **two upvotes** are needed for the large claims (10+ points), equivalent to having someone "sign off" on the chore. It is *not expected* that the entire house will verify every chore. Rather, the person claiming the chore should ensure that at least one other person has verified their work.

After a chore is claimed, that chore's value returns to 0, and begins accumulating points again.

In the unlikely scenario that someone lies about doing a chore (or does an extremely poor job), the rest of the residents may downvote the claim. A failed claim returns the points to the chore, allowing someone else to do the job properly.

Take a break

The point of Chores is to help folks clean up their own messes, with more points (roughly) meaning more mess. When someone is out of town, they aren't making a mess, and so they shouldn't owe as many points. Anyone who goes out of town for at least **3 days** can take a break, and they'll owe less points for that month (also, points will accumulate more slowly on the days that they're gone).

Gift your points

Not every useful piece of work around the house can be expressed as a recurring chore. Things happen randomly, and spontaneously, and it's valuable to be able to recognize those things. As mentioned above, the total amount of points per month is fixed, but there's no reason folks can't give away points that *they themselves have earned*.

After someone has claimed a chore and gotten points, they can gift those points to someone else in recognition of a useful contribution that they've made. It's their choice who to gift and why and how much, since they're the one who earned those points in the first place.

Edit chores list

Before anyone can claim a chore, the chore needs to be defined. Chores can be added, edited, or deleted.

Chore edits start as proposals and go to the house for a vote. If the vote passes, the chore is created and begins accumulating points.

> **Warning:** When defining chores, it is easy to either go **too micro** (e.g. "Wipe off the dinner table") or **too macro** (e.g. "Deep clean the whole kitchen"). If too micro, people will resent having to officially "claim" the chore. If too macro, the chore will never get done, despite being worth a lot of points.
>
> Don't be afraid to experiment and add, remove, or edit chores in the first few weeks.

Set priorities

The **total amount** of points distributed per month is fixed, at 100 points per resident. Those points are distributed continuously over the course of the month. In a 10-person house and a 30-day month, that works out to about **33 points per day** in total. That number can't be changed, as it ensures that chores are done over the entire course of the month. (Imagine everyone getting to 100 points during the first week – the house would be a mess for the rest of the month!). However, those 33 points are divided among the chores in different ways, depending on that chore's "priority".

A high-priority chore gets points faster than a low-priority chore, ostensibly because it needs to be done more often. For example, the kitchen might need to be cleaned daily, while the backyard may need to be cleaned only once a week. So, the kitchen-related chores should be higher-priority than the yard chores, getting perhaps 5 points per day instead of 2. The only rule is that for one chore to gain priority, another one has to lose it – since the total amount of points is fixed, priorities are fundamentally relative.

Chore priorities are determined collectively, but independently, using a novel valuing system. Anyone in the house can, within limits, unilaterally increase the priority of one chore and decrease the priority of another. The idea is that priorities don't need to be set in advance at a meeting, but rather are "discovered" organically as people notice chores being over- or under-valued.

Chore priorities are also interrelated: if you increase a chores priority over many chores, the effect will be bigger than if you increase a chore's priority over only one other chore. If you prioritize a chore over an already high-priority chore, the effect will be bigger than if you prioritize the chore over a low-priority chore. This is a bit analogous to how sports rankings work – beating a top-ranked team has a bigger impact than beating a low-ranked team.

Note: There's more happening under the hood, but it's not important for your day-to-day. If you want to get into the nuts and bolts, go here.

1.3.4 Slash Commands

In addition to the home page, Chores comes with a number of "slash commands" which provide some important management functions. Most people will not need to know about these commands to use Chores.

Note: Commands marked with an asterisk (*) are admin-only

`/chores-channel` *
: The `/chores-channel` command is used by workspace administrators to set the events channel for Chores, which is where app activity is posted and where housemates go to upvote chore claims and proposals. This command takes no arguments, and will set the events channel to the channel in which the command is invoked.

> **Warning:** A channel **must** be set for the app to work.

`/chores-exempt` *
: The `/chores-exempt` command is used to mark certain users as "exempt" from chores, i.e. to indicate that someone in the workspace is not actively present in the house and should not be considered for the purposes of issuing points and voting. In the past this has been used to exempt someone who took a four-month leave of absence, and to exempt an admin account belonging to someone not living in the house.

`/chores-sync`
: The `/chores-sync` command will update the app with the current active users in the workspace, adding any new users and removing any who have been deactivated. Keeping the Chores app synchronized with the workspace is important, as the number of active users determines the total amounts of points issued as well as the minimum number of upvotes needed for proposals to pass.

> **Warning:** Make sure to run `/chores-sync` whenever someone joins or leaves the workspace.

1.3.5 Case Studies

Dish Norms
: A house finds that the *Wash Dishes* chore is under-valued relative to the frequency with which it needs to be done, so they increase the priority of *Wash Dishes*, which routes more points to the chore. This helps, but people also become more comfortable leaving dishes in the sink, thinking someone else will clean them up. At a house circle, the house discusses a norm of "mostly" cleaning dishes – not a hard rule, but an expectation that if time and space allows, people should clean dishes as they go. As a result, there are fewer dishes in the sink, and the dishes that do collect are cleaned quickly by residents who feel fairly compensated. A mix of an increase in points, plus a cultural norm, creates an optimal result.

Handling Weekly Trash
A house adds a *Curb Trash* chore to take the trash to the curb on Monday nights. The trash goes out, but as the chore can only be done once a week, it ends up consistently over-valued, creating conflict as residents compete for the opportunity. The house re-defines the chore as *Trash Takeout*, which consists of either taking the trash to the curb, **or** emptying the kitchen & bathroom trash bins. Now the chore can be done at any time, leading to a better flow of trash throughout the week, while avoiding an over-valuing.

Dealing With Special Situations
The basement floods during a heavy rain. Three housemates work together to help dredge the basement of water, and want recognition for their efforts. There is a chore, *Backyard Tidy*, which has accumulated 60 points, but in the opinion of the house, could easily be skipped. The three housemates claim *Backyard Tidy* and split the points amongst themselves. A temporary suspension of regular rules allows a unique circumstance to be handled smoothly.

Splitting Up Complex Chores
The house finds that a current chore, *Kitchen (heavy)* is prohibitively difficult. As such, it goes undone for long stretches of time, even when worth many points. The house moves to split the chore in two: redefining the initial chore as *Kitchen Floor Clean*, which includes a sweep and mop of the floor, and *Oven & Fridge Clean*, which includes a disposal of old food and a cleaning of the oven and fridge interior. The two chores are now valued and completed on their own terms, at different intervals, and overall more frequently than the larger initial chore.

1.4 Hearts

Install Hearts

Every community begins with the best of intentions. And every community, invariably, experiences conflict. Experiencing conflict is not a choice. Handling it is.

Communities can choose to ignore conflict, letting it build and fester until communal bonds are permanently damaged. Communities can also choose to evade responsibility, giving away power and agency to an authority figure who promises to make the problems go away, but never really does. Or communities could choose to face conflict head-on, engaging in the essential relational processes which lead to fulfilling long-term relationships.

Hearts is an accountability tool, used to help **structure the experience and resolution of conflict**. Hearts is designed with two goals in mind: first, to give members of a community meaningful tools for holding each other accountable; second, to emphasize communication and repair over judgment and punishment.

1.4.1 Quickstart

1. Set an app events channel
2. Encourage people to give karma (`@Kira ++`) if someone exceeds expectations
3. That's it! Everything else is automatic.

1.4.2 Core Concepts

Hearts

The core concept of Hearts is, unsurprisingly, **hearts**. Everyone starts with five hearts, and gains and loses them as the result of various processes.

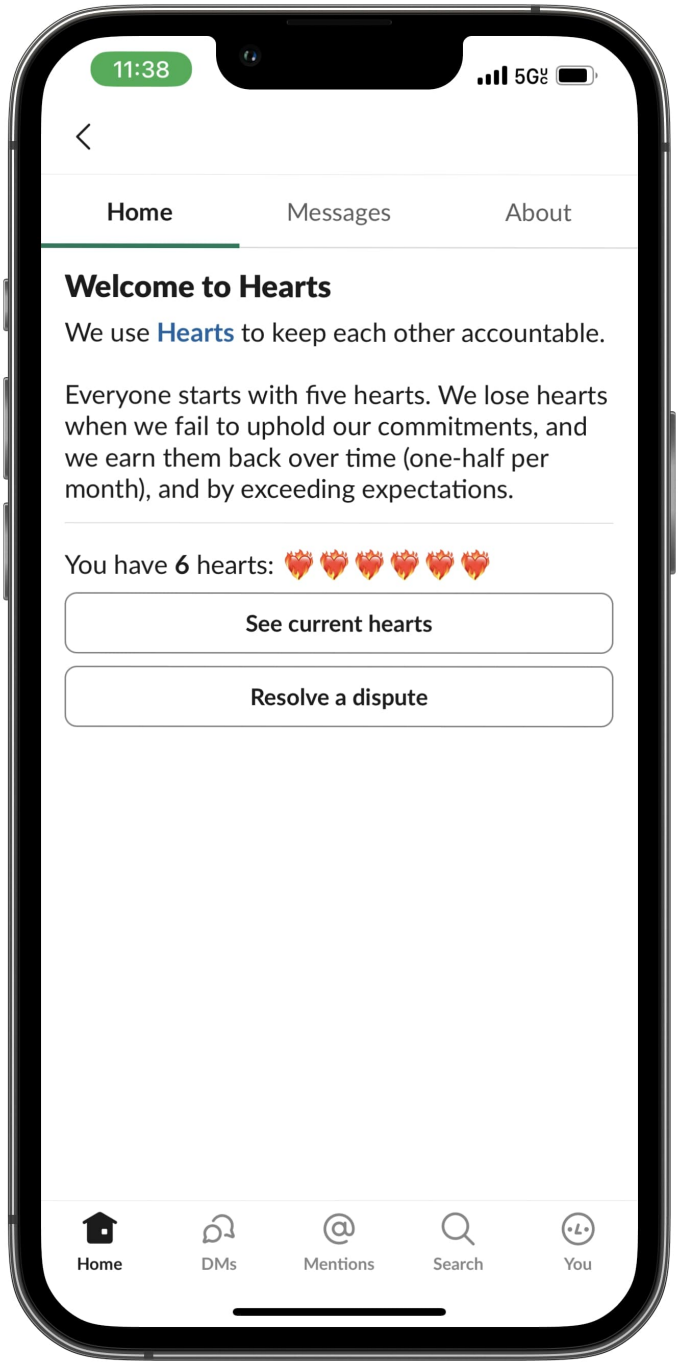
Karma

Much of Hearts is about handling conflict. But what about all the good things that happen, which are in all likelihood the majority of the interactions occurring among the residents? For that we have **karma**. Anyone can give karma to another resident by adding a "plus-plus" (`@Kira ++`) after their name. Every month, all the karma is taken together, and the resident(s) with the most karma get a bonus heart. By earning karma, it is possible to have an epic **six** or even **seven** hearts.

Regeneration & Fading

It is said that time heals all wounds. Whether or not that is true is a question for psychologists and philosophers. What we can say with confidence is that **time restores lost hearts**. Specifically, a half-heart per month, for everybody, always. This is how we center **rehabilitation over punishment.** If you lose a heart for whatever reason, integrate the lesson, and soon enough it will be forgotten. Hearts regeneration will only take you back to the five-heart baseline. If you already have five, you don't get more. If you have bonus hearts (from earning karma), you'll fade back to five at the same rate.

1.4.3 Basic Functionality



The Hearts home page is the hearts dashboard. On the home page, folks can see their current hearts. The app home is also the entryway into the basic functionality, described below:

Resolve a dispute

The most dramatic way to lose hearts is by having another resident call you out for bad behavior, by issuing a

challenge (analogous to "getting a strike"). This, rightly, is an uncomfortable experience, but a necessary one, as the final step in the *conflict resolution* process. If you couldn't be called out, or you couldn't call someone else out, then there would be no accountability: problematic behavior would continue until someone moved out, clearly not an outcome to be proud of. By issuing and receiving challenges, we create a structure for engaging **proactively** with conflict.

Anyone can issue a challenge, stating a number of hearts they think you should lose (up to three), and the reasoning. The issue then goes to a vote. If you lose, you lose hearts. If your challenger loses, they lose hearts. As a way to prevent abuse, a challenger needs a *minimum* of 40% of the house to support them to win, otherwise they lose by default. If losing the challenge will leave the challenged with one heart (or none), then the challenge needs the support of at least 70% of the house. This is how we protect minorities from abuse, while respecting the judgment of the majority.

It is very unlikely someone will be challenged arbitrarily, as the challenger is strongly disincentivized from doing so. Anyone who issues a challenge stands to lose hearts themselves, if other residents feel they are out of line or being abusive. Instead, challenges should (ideally) come as no surprise, and represent a final step in a *respectful process of disagreement*.

See current hearts
Pull up a view showing everyone's hearts, ordered from most to least.

1.4.4 Slash Commands

In addition to the home page, Hearts comes with a number of "slash commands" which provide some important management functions. Most people will not need to know about these commands to use Hearts.

Note: Commands marked with an asterisk (*) are admin-only

`/hearts-channel` *
The `/hearts-channel` command is used by workspace administrators to set the events channel for Hearts, which is where app activity is posted and where housemates go to vote on challenges. This command takes no arguments, and will set the events channel to the channel in which the command is invoked.

> **Warning:** A channel **must** be set for the app to work.

`/hearts-sync`
The `/hearts-sync` command will update the app with the current active users in the workspace, adding any new users and removing any who have been deactivated. The sync command will also add the Hearts app to all public channels, allowing people to give karma in those channels. Keeping the Hearts app synchronized with the workspace is important, as the number of active users determines the minimum number of upvotes needed for proposals to pass.

> **Warning:** Make sure to run `/hearts-sync` whenever someone joins or leaves the workspace.

1.5 Things

Install Things

There's a saying that "organizations are communities with resources". One of the economic benefits of living in community is the opportunity to buy supplies more cheaply, in bulk. Rather than ten tiny bottles of olive oil crowded on the shelf, there can be one large (and cost-effective) bottle (and more shelf space). Examples abound.

Things is a spending tool, letting residents in a house spend out of a shared account, using their own judgment (within limits) on what should be purchased.

Note: Things is a simple tool for managing shared funds directly in Slack. For communities with more advanced needs, consider a more full-featured tool like Open Collective. These tools can also be combined, allowing simple decisions when possible and more advanced decisions when necessary.

1.5.1 Quickstart

1. Set an app events channel
2. Decide on a monthly budget, and use `/things-load` to add it to the account
3. Make a list of 3-5 things that the group buys on a regular basis
4. Enter the things using `Edit Things List`, and upvote them in the app channel
5. Encourage people to `Buy a Thing` whenever they notice something running low

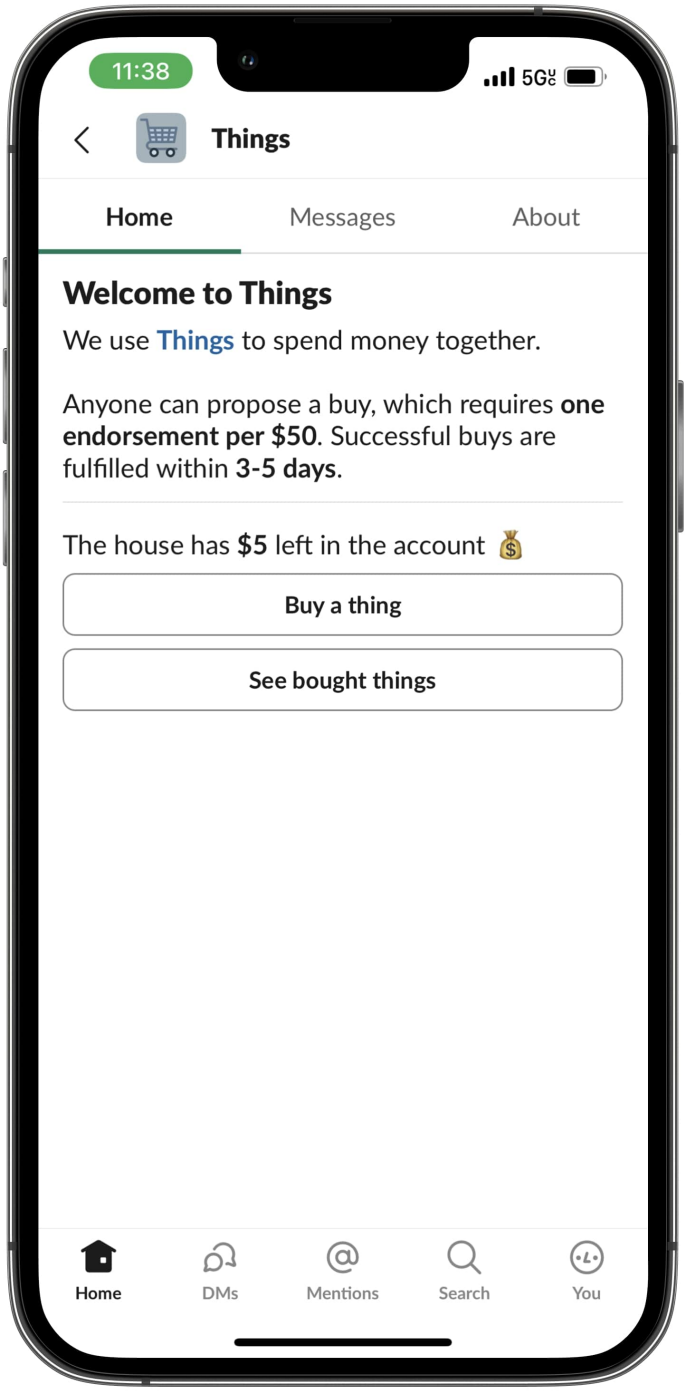
1.5.2 Core Concepts

Things
: A thing is anything that the house intends to buy on an ongoing basis. Things have names and types (beverage, pantry, etc), as well as a price, quantity, and a link to buy.

Buys
: A buy is a **specific** purchase of a thing. Anyone can propose a buy when they notice that something is running low.

Fulfillment
: Fulfillment is the process by which someone actually goes out and physically acquires the thing. There are many ways communities can handle actual fulfillment, from online delivery to pickup-truck farmers market runs. The system is intentionally flexible here to accomodate diverse scenarios.

1.5.3 Basic Functionality



The Things home page is the things dashboard. On the home page, folks can see the current account balance. The app home is also the entryway into the basic functionality, described below:

Buy a thing

Whenever someone notices that a supply is running low, they can propose that the house buy more. The "buy" is then posted publicly, and other residents can upvote or downvote the proposal. A minimum of **one upvote per $50** is needed for the buy to succeed, to prevent one person from spending all the community's funds. For example, to buy $30 worth of olive oil, only one upvote is needed, that of the proposer. To order a $200 house cleaning, on the other hand, a minimum of four upvotes are required.

Once a buy is resolved, it will be fulfilled in some way, shape, or form.

Buy special thing
From time to time, folks want to buy something which isn't "on the list". In that case, they can propose a "special buy" and provide extra details on what they're thinking. Special buys have a longer voting window and require more upvotes to pass, reflecting the extra care required.

Edit things list
Before anyone can buy a thing, the thing needs to be defined. Things can be added, edited, or deleted.

Thing edits start as proposals and go to the house for a vote. If the vote passes, the thing is created and can be bought.

See bought things
Pull up a view showing pending, approved, and historical buys. Historical buys are aggregated by thing, making it easy to see spending trends.

1.5.4 Slash Commands

In addition to the home page, Things comes with a number of "slash commands" which provide some important management functions. Most people will not need to know about these commands to use Things.

Note: Commands marked with an asterisk (*) are admin-only

`/things-channel` *
The `/things-channel` command is used by workspace administrators to set the events channel for Things, which is where app activity is posted and where housemates go to upvote thing buys and proposals. This command takes no arguments, and will set the events channel to the channel in which the command is invoked.

> **Warning:** A channel **must** be set for the app to work.

`/things-load [amount]` *
The `/things-load` command allows admins to add funds to the account.

`/things-fulfill` *
The `/things-channel` command brings up a view for admins to mark buys as "fulfilled".

`/things-update` *
Often, prices and links change. Rather than going through the process of creating an edit proposal, admins can unilaterally update logistical details for existing things. The `/things-update` command brings up a view for admins to update thing details.

`/things-sync`
The `/things-sync` command will update the app with the current active users in the workspace, adding any new users and removing any who have been deactivated. Keeping the Things app synchronized with the workspace is important, as the number of active users determines the minimum number of upvotes needed for proposals to pass.

> **Warning:** Make sure to run `/things-sync` whenever someone joins or leaves the workspace.

1.6 Conflict Resolution

Long excerpt from Frederic Laloux's *Reinventing Organizations*:

> Several other organizations in this research rely on virtually identical conflict resolution mechanisms: **first a one-on-one discussion, then mediation by a trusted peer, and finally mediation by a panel.**
>
> At first, I was struck by what seemed like an extraordinary coincidence. Before engaging in this research, I had never encountered a company with an explicit conflict resolution mechanism, and here I stumbled upon several organizations that had come up with virtually identical processes. In discussions with people at Morning Star, I came to understand that this process is about more than simply managing the occasional workplace conflict. **Conflict resolution is a foundational piece in the puzzle of interlocking self-management practices.** It is the mechanism through which peers hold each other to account for their mutual commitments.
>
> In traditional companies, when one person doesn't deliver, colleagues grumble and complain but leave it to the person's boss to do something about it. In self-managing organizations, people have to step up and confront colleagues who fail to uphold their commitments. Morning Star and other self-managing organizations readily admit that this essential piece can be tricky to put in place and to maintain. **The process is effective to the degree that there is a culture within the workplace where people feel safe and encouraged to hold each other to account, and people have the skills and processes to work through disagreements with maturity and grace.**
>
> Freedom and responsibility are two sides of the same coin — you can't have one without the other (at least not for long). Holding colleagues accountable to their commitment can feel uncomfortable. A clearly outlined conflict resolution process helps people confront each other when needed.

The *Hearts* module exists to formally resolve behavioral disputes. Ideally, however, most conflicts never reach the point of requiring a full house decision. To keep conflicts from escalating, the following process should be used (ideas drawn from this book).

1. First, the two residents having conflict should attempt to resolve the issue amongst themselves.
2. If the two residents are unable to resolve the issue, they should recruit a third resident to serve as an impartial mediator to the dispute.
3. If the three residents are unable to resolve the issue, then a challenge should be issued via *Hearts* to resolve the dispute in favor of one of the parties.

1.6.1 Tips & Tricks

- **Focus on the behavior, not on the person**

  Making it about the behavior will allow everyone to view the situation with some emotional distance, making it much more likely all parties will be able to reach a compromise. When people feel attacked, it will be much harder for them to accommodate your request.

- **Wait a few days**

  If you approach someone immediately after seeing / hearing a problematic behavior, you are much more likely to approach the interaction with a hostile tone (even if this is not your intention). If you see a problematic behavior, unless it is truly urgent, best to wait a few days to consider the best time & place to raise the issue.

1.7 Monthly Circle

> *Ostrom's* seminal work *on shared governance (i.e. self-governance) demonstrated that effective regulatory systems often include educational, co-managerial, rehabilitative, and participatory decision-making components designed to maintain positive, long-term relationships.*
>
> *– Daniel DeCaro*

Note: Chore Wheel provides many useful tools and structures for supporting communal life. However, there are limits to what can be accomplished with formal systems alone. Like two wings of a bird, **structure** must be balanced with **culture** for communities to succeed.

The following is a description of a *possible* community practice, separate from the Chore Wheel tools themselves. Ultimately, it is up to the specific community to develop cultural practices that work for them.

The **monthly circle** is a simple and accessible practice used to pro-actively cultivate shared understanding among the residents of a house, and to better equip the residents to navigate the conflicts and differences of opinion which inevitably arise.

The circle is led by a different resident every month and ideally lasts about 90 minutes. Circles have the following simple structure, which can be easily modified to suit different needs:

1. **Sharing Personal Roses, Thorns, and Buds**
   First, residents go around in a circle sharing personal **roses, thorns, and buds** for the month – things which went well for them, things which did not, and things they are looking forward to in the future.

2. **Sharing House Roses**
   Then, residents go around in a circle sharing **house roses**: positive things which happened in the house and among the residents.

3. **Sharing House Thorns**
   Then, residents go around in a circle sharing **house thorns**: concerns about the house, such as negative patterns or problematic developments. The goal here is less to criticize individuals and more to build a consensus around shared expectations. For groups using Chore Wheel, this is a good opportunity to discuss potential changes, such as editing the chore list or changing chore priorities.

   Tip: During this go-around, there is usually more back-and-forth discussion as people seek clarity around issues discuss possible resolutions. It is recommended to share house thorns in advance, so that others have time to prepare their thoughts and responses.

   ---

4. **Sharing House Buds**
   Finally, residents go around in a circle sharing **house buds**: general hopes and aspirations for the future, such as personal goals, visions for the house, or hopes for the development of current events more broadly.

5. **Closing the Circle**
   At the close of the circle, a different housemate volunteers to lead the next month's circle and sets the date.

1.8 Design Principles

Note: You can read the original project whitepaper here.

Chore Wheel draws influences variously from **cognitive science, computer science, electoral theory, economics, cybernetics, and game design**, with four primary design principles:

- No managers or privileged administrative roles
- Simple and intuitive inputs
- Humans for sensing and judgment, machines for bookkeeping
- Continuously available, asynchronous processes

Taken together, this principles describe a system which is "highly available," both technically and socially. Participants have more opportunities to engage, in a larger variety of ways, while avoiding information overload. Additionally, the minimal role played by the computer allows participants to attach more legitimacy to the outputs.

Although these design principles can be desirable in many settings, we see the coliving setting as having the **greatest potential for benefit**. For example, and by contrast, a government workplace would likely also benefit from simple and intuitive inputs, making it easier for employees to meaningfully participate. But the work done there, being *linear* (i.e. novel, creative) in nature, makes flat organization significantly riskier and more challenging. In a housing environment, on the other hand, much of the work is *cyclical* (i.e. continuous, repetitious, not meaningfully different between iterations), allowing for new approaches which de-emphasize novel ideation and emphasize resource balancing and mutual accountability.

Further, unlike the distributed and anonymous settings of online communities, coliving environments, being real physical spaces, provide many more opportunities for the informal, casual interactions necessary for building strong relationships. As such, we can *assume* the existence of a coherent social sphere, with the software tools merely providing a structure for transparency and accountability.

Note: This project is under active development, and has been supported by the **Open-Source Software** (2x), **Governance Research**, and **Metacrisis** rounds of Gitcoin Grants.

1.8.1 Primary Principles

No Managers or Privileged Administrative Roles

When thinking about governance, most people immediately think about putting someone in charge, and electing leaders is arguably the foundational process in democratic government. Leadership has many benefits: strong leaders can inspire, organize, and guide groups towards successful outcomes. However, leadership also has a cost: selfish leaders can subvert a organization to their own ends, weak leaders can delay and obstruct important processes, and vindictive leaders can use their position to play favorites. Power inevitably corrupts, and when leadership is structurally protected via elected (or appointed) positions, it can be difficult to remove someone who is no longer right for the role.

Chore Wheel avoids this by replacing the "hard power" of specific leaders with the "soft power" of flexible leadership. Anyone who is motivated to provide leadership is welcome to do so, to the extent that they can organize and motivate others. Leadership can look like pioneering a pantry organization, building a vegetable garden, or painting a mural, and people can step into and out of leadership as their circumstances and desires permit.

Chore Wheel conceptualizes leadership as "icing on the cake:" leaders can bring tremendous value, but if no-one is called to leadership, the community still functions.

Simple and Intuitive Inputs

For self-governance to be meaningful, participants must be qualified to engage with the issues and decisions they are presented. Historically, this has been interpreted as a need for public education and the cultivation of an informed community. As society-wide issues grow increasingly complex, however, it becomes increasingly hard for the average person to keep up, making governance a de-facto domain of the privileged.

Chore Wheel approaches this from a different direction: rather than equip the population to to handle complex decisions (worthwhile as that may be), decisions are kept fundamentally simple and accessible. By framing decisions in the simplest possible terms, and ensuring that every decision "contains its own context," participants can **meaningfully** engage in running their communities, without imposing undue burdens and unrealistic expectations. This avoids apathy and disengagement, and helps everyone to feel as though the decisions being made truly reflect the intention of the community.

Humans for Sensing and Judgment, Machines for Bookkeeping

With the rise of machine learning and artificial intelligence, our lives are increasingly been shaped by the outputs of opaque algorithms, trained on questionable data, developed by unaccountable organizations. This has led to a significant loss of public trust in technology and in democratic institutions more broadly.

Having engaged deeply with these critiques, Chore Wheel makes a basic and explicit distinction between the role of the human and the computer. The computer's job is not to make decisions on behalf of a community; rather, its job is to guarantee **process** – to ensure that votes are run fairly and that chores are tallied correctly. All of the *subjective* decisions about good or bad, right or wrong, are made *exclusively* by human beings.

The result is a technical system which people can trust, instead of one which keeps people constantly on guard.

Continuously Available, Asynchronous Processes

As Oscar Wilde famously quipped, "the trouble with socialism is that it takes up too many evenings". Ultimately, the purpose of self-governance is not to sit in meetings, but rather to preserve the right of self-expression in the world. When Robert's Rules of Order (the classic guide to running meetings) was written 200 years ago, people wrote with quills and mail traveled by horse. While the in-person meeting as a format for decision-making has some benefits, the benefits are mostly due to gathering folks in the same place, and can be better achieved by organizing a dinner or games night. As a vehicle for making decisions, apart from the most critical decisions, it is largely obsolete, and an over-reliance on meetings alienates people and concentrates power.

Chore Wheel recognizes that the majority of decisions *do not* require meetings and the imposition on people's time that meetings entail. Instead, all decisions are made *asynchronously*, and people can engage in decision-making at their convenience throughout the day. This significantly reduces the burdens of self-governance, making collective leadership accessible to a wider audience.

Note: While Chore Wheel does not *require* meetings, communities can certainly *have* meetings as needed, as they can be useful for getting alignment on complex issues. In addition, major decisions would *probably* benefit from a meeting, but those choices of when and why are best left to the specific community.

See the section on the *monthly circle* for a possible meeting format.

1.8.2 Secondary Principles

These design principles can be developed further:

Three Institutional Layers

The overall design of Chore Wheel can be understood in terms of three layers, evoking the three layers described in Elinor Ostrom's seminal *Governing the Commons*. The first, or **constitutional layer**, involves the design of the modules themselves. In this first layer, the design of the entire system and its implementation are up for discussion. There are no constraints, as software can be changed in arbitrary ways. The constitutional layer can be understood as governing the system from without by changing rules themselves.

The second layer, the **political layer**, involves participants collaboratively setting explicit parameters that govern the behavior of the system. An example would be choosing the frequency with which a certain chore is to be performed. In the political layer, residents have control over the system's behavior, but only within the constraints set by the constitutional layer. We can think of this as governing the system from within.

Third and finally, the **operational layer** involves residents individually interacting with the system given the constraints created by the constitutional and political layers. In this third layer, residents complete and verify chores, vote on issues, and procure supplies.

This three-layer design is meant to balance flexibility with simplicity - keeping daily interactions clear and straightforward, and providing residents with a structured means for shaping and controlling their environment, while still allowing for unstructured, open-ended changes to be made as needed.

Cheap Information

A guiding motivation for Chore Wheel is the reduction of the cost of information. As observed in *Governing the Commons*, the cost of information is inextricably linked to the design of the system itself. A well-designed system, which makes high-quality information cheaply available, will lead to consistently higher-quality decisions and thus better outcomes. Chore Wheel achieves this by placing an "event stream" at the center of every module. Every action, ultimately an attempt to claim some house resource, creates an event. This can then be interacted with by all residents, most simply in the form of an endorsement or a challenge.

Permissionless by Default

A major design motif for Chore Wheel is "permissionless by default." Whenever possible, synchronous voting should be avoided. In practice, this means that most actions take the form of challenge-response. In such a system, any resident can propose an action (e.g. such as making a purchase out of a shared account). If there is no response to the proposal by other residents, the action will be allowed - and likely occur - after a set period of time. This will be recorded as having passed with a vote of 1-0, representing implicit consent. However, if other residents do not abstain, they may either oppose or support it with their own votes. For major actions, a minimum number or percentage of votes in favor may be required, so as to encourage residents to "do their homework" and establish support prior to initiating the vote.

This approach allows uncontroversial actions to go forward unimpeded (due to a lack of opposition), while allowing for controversial actions to be decided by vote. This "lazy consensus" approach mimics the processes successfully practiced by groups such as the Apache Software Foundation and Wikipedia. To both discourage initiating frivolous voting and encourage participation in out-of-band communication, residents who propose failed actions will receive a small penalty.

Chat-based Interfaces

A second major design motif for Chore Wheel is an orientation around chat-based interfaces. It is currently being developed as a set of Slack applications but is, in principle, portable to Discord, or any extensible chat platform. The vision is for residents to interact with Chore Wheel via a series of chat bots, allowing governance interactions to occur seamlessly alongside other house communication. Each module lives in a dedicated channel and interacts with residents via an events log, which is a series of messages providing information and interactivity. To avoid spam in these channels, they will be read-only for residents. However, residents may add comments and reactions to help keep them engaged with the channels without disrupting their utility. Organizing all interactions as events in a log has positive knock-on effects for auditability and reliability, as any specific state can be reconstructed from the underlying event stream.

Anonymity and Identity

One critical design consideration is the appropriate role and degree of anonymity. What actions must be taken publicly and which can be private? No one should have to respond to anonymous criticism, yet publicly identifying oneself can be intimidating and thus disenfranchising. Ultimately, we choose to require identity for *initial* actions (e.g. completing a chore, issuing a challenge, or making a purchase), but allowing all votes to be anonymous. In this way, at least one person is always linked to any action but the majority of the inputs can be private.

Subjective Inputs

Last but not least, Chore Wheel chooses to use only *subjective* inputs. This means that explicit surveillance is not necessary, and communities using Chore Wheel can sidestep invasive measures practiced elsewhere such

as mounting a camera behind the sink to see who leaves dirty dishes. Such explicit information-gathering approaches create an uncomfortable environment, turn the home into a public sphere, and introduce a new class of measurement error. The constrained physical environment allows for frequent eyeballs to perform the same monitoring function in a more pleasant, less invasive way, while also providing a few degrees of discretion (e.g. "wiggle room").