

# Unscrewer

Leah Liddle

June 30, 2025

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Overview of Magic the Gathering</b>	<b>2</b>
2.1	Gameplay Concepts . . . . .	2
2.2	Land Balancing . . . . .	3
2.3	Overview of Manabase Optimization . . . . .	4
<b>3</b>	<b>Library and Language Choices</b>	<b>5</b>
<b>4</b>	<b>Database Layout</b>	<b>5</b>
4.1	Local vs Online Storage . . . . .	5
4.2	Scryfall Shortcomings . . . . .	6

## 1 Introduction

Magic: The Gathering (MTG) is a deckbuilding Trading Card Game (TCG) produced by Wizards of the Coast (WOTC) in which players use a resource called "mana", generated by "land" cards, to cast "spell" cards. Mana comes in five colours - White, Blue, Black, Red and Green, typically abbreviated to their first letter, with "U" standing in for Blue, and the acronym "WUBRG" referring to all five together - and most spells require one or more specific colours to be cast, meaning that players must chose land cards that maximize the probability that they will be able to generate the colours needed for their spells. While choosing spell cards is a pleasurable creative endeavour, selecting appropriate land cards is simply a matter of picking optimal choices within the limitations of your budget, overall game strategy, and collection. Unscrewer - named after the community term "mana screw", referring to an inability to play your cards due to insufficient mana - is a web app that will automates this process: this software, when presented with a list of spell cards, outputs a list of lands that it has determined will maximize the chance of a player being able to cast their spells.

## 2 Overview of Magic the Gathering

### 2.1 Gameplay Concepts

MTG players begin games drawing a "hand" of cards, and draw one additional card per turn. Each turn, they may play one land card. Mana is extracted from a land card via "tapping" it (turning the physical card 90 degrees), and all lands untap at the start of each turn. A tapped land cannot be used to generate mana until it is next untapped. Players may cast any number of spell cards each turn, as long as they can generate enough mana to cast each one. While the specifics of this nomenclature will be important in this writeup, the overall effect of this is that, absent any other cards which accelerate this process, a player each turn has access to  $N$  mana, where  $N$  is the number of turns so far in the game on which they have managed to play a land. While deckbuilding restrictions and player life totals vary between different "Formats", the aforementioned model remains constant.

Most cards require one or more mana of one or more specific colors along with a quantity of "generic" mana, which may be any color. However, some cards only require generic mana and can be fully paid for with any color, while others only accept specifically colored mana for their entire mana cost. A small subset of cards require specifically colorless mana. A card's mana cost is reflected in its "pips" - a card with three green pips and a generic 3 pip costs three green mana and 3 mana of any color.

The land cards Plains, Island, Swamp, Forest and Mountain are called "Basic Lands" and produce W, U, B, R and G respectively. Decks are normally limited to either one or four copies of each card (depending on the Format), but can have any number of basic land cards. A deck requiring only one colour of mana may fill its manabase entirely with Basics, while a deck requiring all five may use very few.

Some lands, called "Utility lands", produce only colorless mana but provide some other useful in-game effect.

Cards have several "types". "Basic", as described above, is a supertype. "Land" is a type ("Spell", however, is not - Spell cards are split into other types including Instant, Sorcery and Enchantment). Cards may also have several "subtypes" which can be referenced by other cards for synergistic reasons. A card that says "search your library for an X card" is referring to a card whose list of types includes, but is not limited to, X. In addition to being the names of basic lands, Plains, Island, Swamp and Forest are also land subtypes, and any land with that subtype automatically can be tapped for mana of the respective color. The card "Underground Sea", for example, has Island and Swamp as subtypes, and can tap for Blue or Black mana; it can be selected as a target by any card that targets an "Island" or "Swamp" card, but not by a card that targets a "Basic" card. Although all cards with the swamp type tap for B, not every card that taps for B is a swamp.

"Strict Betterness" in magic refers to a card which is in all cases better than another. Underground Sea, for example, is strictly better than an Island or

a Swamp, as it can tap for either type of mana. However, the similar card, "Watery Grave", which is also an Island/Swamp, but enters tapped (and is thus unusable on its first turn) unless a player pays two life is not strictly better, as if a player has only 1 life, 5 lands and needs to cast a spell that requires 6 mana to save their life, an Island will allow them to cast that spell on that turn, and a Watery Grave will not. However, many cards that are not "Strictly better" than others may still be generally regarded as better than others thanks to widespread player consensus: since paying life for a benefit is generally considered a negligible cost, the ability to access multiple colours of mana means that Watery Grave is generally considered to be better than an Island. I will refer to this as "Agreed Betterness."

Many lands are arranged into "cycles", in which each land is mechanically identical save for offering different colours. "Sacred Foundry", for example, belongs to the same cycle, "Shock Lands," as "Watery Grave", differing only in that it is a Mountain Plains land that provides Red or White mana. Cycles differ from types in that they are not an explicit in-game concept.

In player vernacular, which I will use in this writeup, an "X Land" is a land belonging to cycle X, while a "X" - a "Basic", an "Island", etc - refers to any card with those types. Although some lands do produce two mana per tap, this is rare, and a land which "taps for two colours" taps for either of those two colours, not both at once.

## 2.2 Land Balancing

"Balancing" refers to the aspect of card design in which a useful card is given drawbacks to prevent it from being strictly better than many other cards. In spells, this is typically enforced via mana cost: a stronger spell costs more mana.

WOTC generally avoids making lands that are strictly better than basic lands; cards that are, such as Underground Sea, typically reflect an out of fashion design philosophy, and due to age and power are thus rare and expensive. Because of this, most lands that tap for two or more colors of mana are given some sort of drawback. Players will therefore assemble the bulk of their "manabase" (a collective noun for the land cards chosen for their deck) out of a mixture of basic lands, which tap for one colour with no drawbacks, and multicolor lands that feature some drawback, choosing based on personal experience, cards in their collection, and price.

The most common drawback for a multicolor land is to make it enter tapped, and thus cannot be used on the turn they are played. While there are several cycles of two or three color lands that simply enter tapped, many can enter untapped if the player meets a specific criteria, or pays a specific resource, when they are played. Since many cycles enter tapped or untapped depending on what other lands are in play, this gives manabases a self referential character.

There are also supermechanical factors that balance lands: more powerful ones are rarer, making them more expensive, less likely to show up in a player's existing collection, and sometimes bringing a social stigma.

Lands with basic land types are generally considered better than lands that produce the same colors without basic land types. This is due to their synergy with cards that can search your library for basic land types, notably including the "Fetch Lands", a highly-regarded land cycle that, rather than producing any mana, searches your library for a land with one of two basic land types.

Most non-basic non-utility lands - which will be a central focus of this document, and referred to as "nonbasics" going forward - tap for two colors of mana, although some tap for three and some for all five.

## 2.3 Overview of Manabase Optimization

Most strategies to prevent mana screw focus on choosing appropriate spells: a central concept is a deck's "mana curve", referring to the number of spells it has at each mana cost. Choosing appropriate lands is comparatively understudied, as it is a much more numerically complex process - requiring a count of the total number of pips in your deck, and how they are spread across the cards - but also one that is readily addressed by heuristics:

Multicolor lands can broadly be ranked into three categories:

Optimum lands - cycles that are strictly better than Basic Lands (such as Shock Lands or, in some context, Bond Lands), or that have repeatedly proved their efficacy in tournament play (Shock Lands and Fetch Lands).

Strong lands - multicolor lands that enter tapped unless a specific condition is met, or are balanced by another non-negligible factor that keeps them outside the above category. This may also include lands that enter tapped but tap for three or more colours, or which have basic land types and thus may be searched by fetch lands.

Taplands - two-color lands that enter tapped.

When creating a manabase, players will typically start with a quantity of basic lands roughly proportional to the pips used by their library, supplement this with what optimum lands they have access to, and then choose other nonbasic lands from among strong lands and taplands based on personal preference. While some taplands may offer additional benefits beyond mana production that make them appeal to the player (ie, the "Scry Lands", which let the player look at the top card of their library on entering), that sits outside the scope of optimization. From an optimization perspective, the question is - what strong lands are picked? Because of the aforementioned reflexive quality of a manabase, this question in part depends on what other lands the player has access to.

The card "Woodland Chasm" is an instructive example here. It is a tapland with two basic land types, Forest and Swamp. It would be an undesirable situation if the player draws it when having access to exactly one more land in play would allow them to cast a higher cost spell; based on the mana curve, we can assign that probability  $N$ . However, if the deck also contains the six fetch lands capable of finding a Swamp or Forest, that probability would fall significantly, as any prior turn in which the player had access to one of these fetch lands and did not need one new untapped mana would have provided

them with a chance to put Woodland Chasm into play with no detriment to their game.

### 3 Library and Language Choices

My choices of language and libraries were informed by two main priorities. Due to my short turnaround time, it was important that I use libraries and languages with substantial community support for web development. Meanwhile, as a usable app, Unscrewer benefits from high performance so as to maximize the number of simulations it can run, but does not need to offer a complex user interface nor store user data, prompting me to favour high-performance tools over complex and scalable ones.

In places where these requirements are at odds, I prioritized the former: my choice of Python as a backend and Javascript as a frontend was driven largely by the popularity of these languages in web design. However, in other decisions, the two requirements informed each other constructively. I chose Flask as a backend web framework as its simplicity made it both easy to learn and reputedly faster; contenders like Django are made both slower and more complex due to their abundance of features (FastAPI, potentially lighter and faster than Flask, was discarded due to its smaller userbase and thus relative paucity of learning resources). React, which I chose as my frontend framework, similarly sports a wealth of support resources, and features a Virtual DOM that lowers performance overheads when users make small input changes - a relevant concept here, as users will likely order several simulations with small preferential changes on each one.

Lacking access to the popular Django ORM, I interacted with my MTG Card database via SQLAlchemy. To fill that database, I opted to create my own script using the Scrython library that would scrape the MTG Database Scryfall, rather than relying on existing projects that compile data for download, such as mtg.io API or mtg.json, as Scrython, being a widely used player resource, is kept regularly updated, and my script could be easily re-run to account for new sets or price fluctuations.

## 4 Database Layout

### 4.1 Local vs Online Storage

Since new cards are regularly released, and card-price - an important deck-building consideration - constantly fluctuates, a key feature of my database is updatability.

Since my backend has direct access to Scryfall via Scrython, I initially considered trivializing this by foregoing a locally stored database, and simply querying player inputs to Scryfall itself. Testing almost immediately showed this to be untenable: although lands could be downloaded in bulk, referencing a player's

input cards *to determinemanavalue* required an individual search for each card. Even small decklists required several minutes to parse.

Instead, I opted to create a backend object, the DBManager, called not from the server but from a separate script, `Manage.Database.py`, which, on running, would RESTfully scrape Scryfall and update all tables. Although RESTful considerations made this code slow to run - taking around ten minutes - this is fairly manageable on a weekly timeframe, and could be automated in future versions of the software.

## 4.2 Scryfall Shortcomings

Scryfall does exclude some information which is relevant to Unscrewed. Since the end-goal of my work here would be to have a database script that would update automatically on a timescale, I needed my script to parse the following automatically:

1. Land Searching Capabilities - usefully, Scryfall lists the mana that each land produces. However, some lands produce no, or only colorless, mana, and instead sacrifice themselves to search your library for a basic land, which is not listed. Helpfully, most of these lands are sorted into cycles, which I marked in the cycles table as "fetch" lands (a standard term in the community). I then gave each land a cached property "true\_produced" which combined the data from scryfall with, if the land belonged to a fetch cycle, any land types mentioned in the text.
2. Price in GBP - Scryfall lists prices in EUR and USD. Since price fluctuations would prompt most regular updates anyway, I opted to add my own GBP column and calculate it during the scrape using a conversion library.
3. Intermittent price absences - some cards do not have a listed price. I informed my database to list these with a price of minus 1, and WILL FIGURE OUT HOW TO HANDLE THESE.
4. Edge cases - MTG features "Unset" cards, which typically fall comedically outside design norms. "Little Girl", for example, costs half of one white mana *nothercardhasanon - integermanavalue*. These cards are not popular but not inconceivable to use, as players frequently make comedy decks, and although I experimented with database schema that would incorporate these - for example, storing mana value as a float rather than an int to accommodate Little Girl - I decided that warping my model around a single rarelyplayed card was a low-reward development approach. Instead, I equipped my DBManager with an array `edge_cases`, and gave it customized handling instructions for each one (Little Girl's mana value, for example, was rounded to one, as this much more closely fits how she would be played)
5. Cycles - Scryfall does not sort lands into cycles. I addressed this by providing each cycle with a regular expression that matched all lands within

it, and sorting on download.