



The Kartoza Handbook



Your Open Source Geospatial Experts.

Kartoza Pty (Ltd.)

2022



Contents

Home	3
Kartoza Handbook	3
Contributing	5
Polyglot: The Emoji Map	7
Company	9
About Kartoza	9
Kartoza's Strategic Objective	10
Kartoza's Thirty Principles	11
Setting up your PC	13
GIS	21
Cartography Guidelines	21
Technologies	25
GIS Resources	26
Development	27
Development	27
Conventions	28
Technologies	29
Environments	30
DevOps	31
DevOps	31
Security	32
DevOps Procedures	33
Infrastructure	34
Resources	35
Resources	35
Cheatsheets	36
Tutorials	37
Kartoza Media Center	38
Links	39



1 Home

1.1 Kartoza Handbook

This site comprises the organisational and technical documentation for [Kartoza](#).

This is where we highlight the procedures, principles, and processes related to Development, DevOps, and GIS, in line with the organisations best practices.

This is open content, available on [GitHub](#) and freely licensed as public domain content under the terms of [CC0 1.0 Universal](#).



Note:

This content is delivered without any warranty, express or implied. Use at own risk.

1.1.1 Purpose

The purpose and function of this collection of documents is to perform the following:

- Improve consistency in processes and products
- Improve efficiency and innovation
- Increase transparency and accountability
- Improve value for clients
- Provide a space for the dissemination and proliferation of ideas
- Promote a culture of openness and collaboration
- Provide a single source of truth for resources
- Promote personal growth and development
- Add value to the community

1.1.2 Scope

Kartoza is a company that specializes in Open Source Geospatial solutions. As a result, topics covered by this documentation will be limited to categories relevant to the operations of Kartoza. These categories are outlined as follows:

- [Company](#): General practices and procedures for Kartoza staff
- [GIS](#): Resources for Geographic Information Systems and data
- [Development](#): Software development processes, tools, and conventions
- [DevOps](#): Developer operations and system administration
- [Resources](#): Cheatsheets, links, media, and other resources

This repository is limited to these categories, with some slight overlap in domain verticals.

In the majority of instances, where overlaps with adjacent fields of interest, such as Data Science, "Big Data", or to some extent even earth observation and remote sensing, these elements should be primarily remanded to external references in the resources section.

Whilst some resources (such as tutorials on Semi-Automated Classification with QGIS) may be considered valuable additions, the priority of this content is to remain a concise collection of resources directly related to the operations and key competencies of Kartoza staff.



1.1.3 Limitations

This collection of documents does not constitute a replacement for Standard Operating Procedures (SOPs) and company policy. In some cases, our SOPs may point to sections of this handbook, but the SOP itself is canonical as to where the procedure content lies. Due to the rate at which modern technology develops, opinions change, project needs or priorities are adjusted, and the operational requirements of the organisation evolve, this collection is somewhat ephemeral and should be considered a dynamic "living document" which is subject to constant change and iteration.

Processes and documentation from this collection are developed in conjunction with the broader community, independent contractors, temporary staff, juniors, and interns. As such they are not guaranteed to reflect the views of Kartoza, and are not intended to be a substitute for official policy.

As resources and processes mature, they may be incorporated into official SOPs as required.

1.1 Contributing

Although [Kartoza](#) is a privately held Open Source development and consulting company, the organisation deeply values transparency, delivery of value to the broader community, and continuous engagement with all stakeholders.

Community contributions to this documentation site and associated resources are welcome. Contributions are expected to adhere to the QGIS.org [Diversity Statement](#) and [Code of Conduct](#).

 Note:

If you have any queries or feedback, please contact us at info@kartoza.com




1.1.1 Conventions

The following conventions outline expectations for contributions to this documentation project:

- Use grammar checking tools where available, such as grammarly or spell checking extensions for your IDE
- Request a review for internal changes before they are merged into the main repository
- Default to British English spellings rather than American English
- Do not commit sensitive information or links to non-public resources. This includes internal unlisted youtube channels, cloud storage repositories such as nextcloud, or personal details
- Due to the nature of the contents in this repository, when making large edits that do not create new content, communicate with team members to prevent collisions
- When producing assets such as images, ensure they are the minimum viable size and do not commit large resources to git
- Assets and media elements such as images are best left out of the source control where possible. Use an external storage system (e.g. minio/ s3), or paste an image into an image to get a GitHub reference to the media item rather than committing to git. This includes screenshots etc. that are likely to change or be updated over time. Use the assets directory to store assets that are not likely to change such as logos
- When using assets, upload them to a suitable file path according to their primary usage location, e.g. `assets/images/resources/cheatsheets/postgresql/joins.png`
- Due to the depth and breadth of these resources, it is necessary to manually index new pages in various subcategories to ensure access and discoverability
- It makes sense to use a consistent legend of emoji for tagging project and documentation items. Although it may have a steep initial learning curve or implementation strategy, using emoji and unicode symbols to tag elements is a fun and intuitive way to attach metadata to elements which makes visually scanning over documents and commit histories much more effective in multiple languages. Please see the [polyglot](#) document for more information.
- Build and check your changes locally to catch any errors before committing them to the main repository
- TODO: come up with some formatting guideline (e.g. max line length etc)

1.1.2 Tags, Badges, and Shields

[Tagging elements with emoji](#) is useful for visual identification and search of various elements in broader categories, but sometimes more explicit metadata is required to be attached to something to indicate whether it constitutes a general resource, opinion, community standard, or whether something is a known reference item connected to an official SOP. One method of identifying such features may be using [shields.io](#), for example:

-  `! [Best Practice] (https://img.shields.io/badge/kartoza-best--practice-blue)`
-  `! [Community] (https://img.shields.io/badge/community-standard-brightgreen)`
-  `! [Industry] (https://img.shields.io/badge/industry-standard-yellowgreen)`

1.1.3 Translations

Due to the scope and intension for frequent updates to this documentation, additional languages will not be supported at this time.

Translations and i18n are handled by the documentation framework, as outlined in the [mkdocs](#) and [mkdocs-material](#) documentation.

1.1.4 Framework

This documentation uses the [mkdocs-material](#) framework, and site configuration is specified in the `mkdocs.yml` file. Various [extensions](#) are supported to improve usability, such as [pymdown](#), which may be enabled via pull requests. Please note that only extensions which provide relevant value will be considered for integration, and extensions with significant learning curves or duplication should be avoided. Extensions which provide accessibility improvements are welcome.

1.1.5 Building

The online documentation is built using github actions and published to the `gh-pages` branch.

To build the documentation locally, use the docker command `docker run --rm -it -v ${PWD}:/docs squidfunk/mkdocs-material build` to populate the site directory with the static content. To serve the data for testing, a simple solution is to use a python webserver to serve the data at `127.0.0.1:9101` using the command `cd site && python -m http.server --bind 127.0.0.1 9101`.

Note that the generated site data and assets are explicitly excluded from git.

1.1 Polyglot: The Emoji Map

Although it may have a steep initial learning curve or implementation strategy, using emoji and unicode symbols to tag elements is a fun and intuitive way to attach metadata to elements which makes visually scanning over documents and commit histories much more effective in multiple languages.

In order for this to be effective, a consistent method of referencing the emoji meanings is required. This is challenging because the utilisation of emoji are typically context specific, which requires a mapping of emoji meanings for various contexts.

 Note:

Kartoza values inclusion and diversity. Please contribute to ensure that the items represented here remain inclusive and fair wherever possible.

1.1.1 gitmoji

Tagging commit messages in git is a useful tool for visually assessing the issues addressed by a particular commit.

A community standard has already been developed, available at <https://gitmoji.dev/>

1.1.1.1 kitmoji (gitmoji-)

gitmoji is fairly comprehensive collection, and used quite widely in the tech community (relative to similar projects). The downside to this is that it is rather verbose and becomes cumbersome to learn and use.

The simple solution is to select a subset (kit) of the gitmoji icons and use them as broader higher level categories. This keeps things a bit more consistent and allows gradual adoption of the wider collection.

id	Icon	Reference	Function
1	🌟	<code>:sparkles:</code>	New features
2	🐛	<code>:bug:</code>	Bugfix
3	♻️	<code>:recycle:</code>	Refactoring/ Comments
4	📝	<code>:memo:</code>	Documentation
5	💄	<code>:lipstick:</code>	UI
6	⚡	<code>:zap:</code>	Performance
7	🚀	<code>:rocket:</code>	CI/ CD/ Deployment
8	✅	<code>:white_check_mark:</code>	Testing
9	🔒	<code>:lock:</code>	Security
10	🔥	<code>:fire:</code>	Remove data
11	⏮️	<code>:rewind:</code>	Revert changes
12	🧪 / 💩	<code>:alembic:</code> / <code>:poop:</code>	Experiments/ PoC/ bad code

Not every commit has to have a gitmoji, but it's useful for common cases.

1.1.2 GeoMoji

As with gitmoji, we need a consistent way to reference Geographic Information Elements. This could be related to data sources, licenses, tools, or standard metadata categories. Some may [confuse the concept of geomoji with generic symbology and signage](#), but in this instance the reference is to a series of common Emoji characters which can be used as concise, visually effective hashtags that can be used across applications, search tools, git messages, documentation, or social media. It would be great to extend this concept to a collection of map symbols (e.g. using emoji instead of a font library).

A alpha-state concept project is in development at <https://github.com/zacharie/geomoji>.



1 Company

1.1 About Kartoza

In this section we describe the company, our ethos and general expectations of our team members and their use of technology and systems within the company.

All staff members are expected to read and comply with (where applicable) the content laid out in this handbook.

Kartoza is a South Africa-based Free and Open Source GIS (FOSSGIS) service provider. We use GIS software to solve complex location-related problems for individuals, businesses and governments around the world.

Kartoza was formed as a merger between Linfiniti and Afrispatial.

kartoza logo

Learn more at our company website <https://kartoza.com>

- [Strategic Objective](#)
- [Your Kartoza Computer](#)

1.1 Kartoza's Strategic Objective

The Kartoza Strategic Objective is the basis for all corporate and individual decision making.

Statistically we are the largest FOSS geospatial service provider in South Africa and in the top ten globally.

We show up on the first page of google.com and google.co.za searches for these keywords: 'open source GIS'; 'FOSS GIS training'; 'FOSS GIS support'; 'geospatial web development'; 'QGIS development'

We aim to grow Kartoza revenue by 15% year on year.

We aim to grow our staff complement on a sustainable basis till we reach around 25.

We understand that every result is preceded by a 1-2-3-4 step process. It is within these processes that we spend our time, as we relentlessly "work" the systems of the business to perfection.

Our guiding documents are this Strategic Objective, Our [Kartoza Operating Principles](#), and our collection of [Working Procedures](#).

Kartoza's primary offerings are geospatial products and services using Free and Open Source Software. These facilitate spatial decision making and provide tools for economic empowerment.

Through intense commitment to our employees, we will contribute to the success of our clients. The consequence of having loyal, smart, hard-working, long-term, and well-compensated staff is superb quality service to customers.

Our business is complex, with many human, mechanical and computer systems in simultaneous motion. Success depends on refined communication and organisational processes, dedicated staff, documented procedures, first-class office space and equipment, rigorous quality assurance with continuous measurement, assertive innovation, intense planned maintenance and system improvement, aggressive and measured marketing, and relentless attention to detail in every nook and cranny.

Our competitive advantages include an established track record, our ability to solve complex problems with great design, products designed around the unique needs of the customer, thoughtful customer service that is immediate and consistent, the latest high-tech equipment and personal and corporate integrity. We use extraordinarily efficient business systems. We constantly refine and improve all internal systems and mechanisms.

To grow, we follow a two-pronged strategy of:

1. Pursuing substantial consulting, development and implementation projects in our target markets
2. Building products and services that generate passive income, juxtaposed with assertive marketing efforts.

Although we tightly direct Kartoza's operation through guiding documentation, we will modify that documentation immediately if an enhancement can be made: "Our operational framework is rigid, but that framework can be modified instantly."

We segment responsibilities into specialised "expert compartments" with appropriate cross-training among departments. We have backup personnel for all positions.

Kartoza is globally active and locally relevant. Our primary vertical markets include:

- Agriculture
- Disaster preparedness, response and management
- Land information management
- Education
- Humanitarian support
- SDI support
- Monitoring and observation
- Biodiversity and conservation support

(These are fleshed out in [\[Kartoza-Portfolio-Offerings\]](#))

Kartoza aims to model itself on the concept of a [B-Corporation](#), in the sense that through the work we do and the people we employ, we aim to be socially and environmentally responsible.

1.1 Kartoza's Thirty Principles

1. We are innovative. This means we are not set in our ways. We have a diverse toolset and skills and while we retain combinations that work well, we readily change and adapt these and discard those that don't work.
2. Company decisions conform to the [Strategic Objective](#), these Thirty Principles and [Working Procedures](#) documents.
3. We are a lean and agile company and follow [SAFe](#) principles.
4. We promote open and easy access to information.
5. We provide opportunities for community involvement.
6. We provide opportunities for education and training in FOSS tools and GISc.
7. Our internal systems and client-facing solutions are scalable, resilient and reliable.
8. We are the highest-quality spatial IT company in South Africa and equal to the best globally. We do whatever it takes to ensure the quality of service to our clients, employees and suppliers is impeccable.
9. We draw solid lines, thus providing an exact status of where things stand. Documented procedures are the main defence against grey-area problems.
10. "Get the job done." Can the employee do his or her job, or is there always a complication of one kind or another? This ability to "get the job done quickly and accurately without excuses or complications" is the most valuable trait an employee can possess.
11. Employees come first. We employ people who have an innate desire to perform at a hundred percent. We reward them accordingly. The natural outcome is that we serve our clients well.
12. We are not fire killers. We are fire prevention specialists. We don't manage problems; we work on system enhancement and system maintenance to prevent problems from happening in the first place.
13. Problems are gifts that inspire us to action. A problem prompts the act of creating or improving a system or procedure. We don't want setbacks, but when one occurs we think, "thank you for this wake-up call," and take assertive system-improvement action to prevent the setback from happening again.
14. We focus on just [[a few manageable markets | Kartoza Portfolio Offerings]]. Although we watch for new opportunities, in the end we provide "just a few products and services implemented in superb fashion," rather than a complex array of average-quality offerings.
15. We find the simplest solution. Ockham's Law, also called the Law of Economy, states, "Entities are not to be multiplied beyond necessity . . . the simplest solution is invariably the correct solution."
16. The money we save or waste is not Monopoly money! We are careful not to devalue the worth of a Rand (or Dollar or Rupiah or Euro) just because it has to do with the business.
17. We operate the company via documented procedures and systems. "Any recurring problem can be solved with a system." We take the necessary time to create and implement systems and procedures and in the end, it is well worth it. If there is a recurring problem, a written procedure is created to prevent the problem from happening again. On the other hand, we don't bog down the organisation with processes and procedures that target once-in-a-while situations. Sometimes we elect to not create a procedure.
18. "Just don't do it." Eliminate the unnecessary. Many times, elimination of a system, protocol, or potential project is a very good thing. Think simplicity. Automate. Refine to the smallest number of steps or discard altogether. Would a simple "no" save time, energy or money?
19. Our documented systems, procedures, and functions are "off-the-street." This means anyone with appropriate training can perform procedures unassisted. The real-world evidence of this is we can hire an individual "off-the-street" who has good GIS or programming skills and have him or her doing QGIS or development work within a week. For this result, protocols have to be efficient, simple, and thoroughly documented.
20. Do it NOW. All actions build on "point-of-sale" theory. We don't delay an action if it can be done immediately. Just like any major retail outlet, we "update inventories and databases at the exact time the transaction takes place." There is no paperwork floating around the office after a physical transaction. We ask, "How can we perform the task NOW without creating lingering details that we must clean up later?"
21. We glean the Kartoza mindset from Stephen Covey's books, including The 7 Habits of Highly Successful People, First Things First, and The 8th Habit. As well, we consider Good to Great by Jim Collins; The E-Myth Revisited by Michael Gerber; Awaken the Giant Within by Anthony Robbins; Maverick by Ricardo Semler; Work the System by Sam Carpenter.
22. We pattern individual organisation upon Franklin-Covey theory. We use organising mechanisms that are always at hand. We prioritise, schedule, and document. The system is always up-to-date and we use it all the time. For Kartoza,



- these are Github, Google Suite (Mail, Drive, etc.), SyncThing, Sage, OpenProject and others (subject to change as of August 2020)
23. Sequence and priority are critical. We work on the most important tasks first. We spend maximum time on "non-urgent/important" tasks via Stephen Covey's time-matrix philosophy.
 24. We double-check everything before release. If a penchant for double-checking is not an innate personal habit, then it must be cultivated. Double-checking is a conscious step in every task, performed either by the individual managing the task or someone else.
 25. Our environment is spotless: clean and ordered, simple, efficient, functional. No "rat's nests", literally or figuratively.
 26. Employee training is structured, scheduled, and thorough. Assertive client contact is also structured, scheduled, and thorough.
 27. We are deadline-obsessed. If someone in the organisation says they will be finished with a task or project by a certain date and time, then he or she commits to finishing by that deadline (or, if legitimate delays intrude, advise coworkers well in advance that the deadline is impossible to meet).
 28. We maintain equipment and keep it a hundred percent functional at all times. If something is not working as it should, fix it now even if it's not necessary to fix it now. It's a matter of good housekeeping and of maintaining good habits. This is just the way we do things.
 29. Mastery of the English language is critical. We are aware of how we sound and what we write. We do whatever we can to improve. We are patient as a coworker corrects us.
 30. We study to increase our skills. A steady diet of reading and contemplation is vital to personal development. It is a matter of self-discipline.
 31. We avoid multitasking activities. When communicating with someone else, we are a hundred percent present. We give full attention to the person in front of us (or to the task at hand). We focus on listening and understanding. Read the classic 'Treating Type A Behaviour and Your Heart' by Meyer Friedman. "Mindfulness" is paying complete attention to one thing at a time: read 'Full Catastrophe Living' by Jon Kabat-Zinn.
 32. When in the office we work hard on Kartoza business. We keep our heads down; we focus, and in turn the company pays well. That's "the deal". The work week rarely exceeds forty hours.
 33. Complete means "complete." Almost or tomorrow is not "complete." In particular, this is germane to GitHub issues.
 34. We strive for a social climate that is serious and quiet yet pleasant, serene, light, and friendly. Kartoza is a nice place to work.
 35. As opposed to "doing the work," an "operations manager's" job is to create, monitor, and document systems (which consist of people, equipment, procedures, and maintenance schedules).
 36. The Managing Director(s) oversee department heads and overall systems. It is the MD's job to direct, coordinate, and monitor the company.

1.1 Setting up your PC

1.1.1 Hardware

Here are the standard minimum guidelines for hardware for Kartoza Staff:

Laptops are preferred in general. Many of our staff work in areas with unreliable power supply and so you need to be able to work offline for at least four hours.

1.1.1.1 Admin Staff

Admin staff tend to have less demanding activities which is reflected in the hardware:

Feature	Requirements
RAM	8GB
Hard Disk	256GB SSD
Internal Display	1920 x 1080 or better
External Display	1920 x 1080 or better
Operating System	Ubuntu LTR
CPU	Mid range e.g. i5 4 core or Athlon equivalent

1.1.1.2 GIS Staff

GIS Staff need laptops with good storage capacity for accommodating large GIS datasets, and good processing power to perform time-consuming analysis quickly.

Feature	Requirements
RAM	16GB
Hard Disk	1TB SSD
Internal Display	1920 x 1080 or better
External Display	1920 x 1080 or better
Operating System	Ubuntu LTR
CPU	Mid range e.g. i5 4 core or Athlon equivalent

1.1.1.3 Developer Staff & Devops

Developer Staff and Devops need laptops with processing power so they can run multiple containers to emulate the deployment environment for their apps. Developer staff tend to have more technical skills and may install their own preference of Operating System if they prefer.

Feature	Requirements
RAM	16GB
Hard Disk	500GB SSD
Internal Display	1920 x 1080 or better
External Display	1920 x 1080 or better
Operating System	Ubuntu LTR or user preference
CPU	Mid range e.g. i5 4 core or Athlon equivalent

1.1.1.4 Additional Hardware



All staff should in addition be issued with:

- A USB headset. USB headsets include their own DSP (Digital Sound Processor) and will generally have a better sound quality than an analogue headset.
- An external disk for backups. This should again be encrypted. The disk should be 4x the size of the hard disk. Use Déjà Dup Backups to run automatic backups on a nightly basis.
- A Kensington lock. This should be used whenever the laptop is left unattended in a public place (i.e. anywhere other than your home).
- A Yubikey. This will be used to authenticate to Google Apps for Domains (Via Yubikey TOTP), BitWarden, your local PC login (via FIDO2) and other services such as NextCloud. Each staff member should be issued with two of these devices and the second should be stored at home in a safe place in case the first is lost. One of the following models are suggested:



Yubico Security Key NFC - U2F and
FIDO2, USB-A, NFC, Two-Factor
Authentication

★★★★★ ~ 2.408

30,25€

✓prime

Envío GRATIS por Amazon



1.1.2 Base Install Requirements

Every staff computer should have the following as a minimum:

- * Encrypted disk. Under Linux use LUKS when you install to encrypt at a minimum your home partition. Ideally your whole system should be encrypted since if you run docker, postgres and other similar services, you have exposure to data loss if someone steals your PC.
- * Strong password. The password for your account should not be used for any other system.
- * Yubikey PAM Integration. We recommend as an added precaution to set up the YubiKey PAM module which will require to touch your YubiKey after typing in your system password to authenticate. The process for doing this is described [here](#).

 Note:

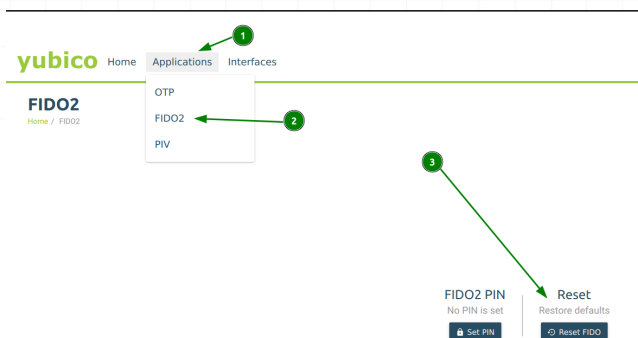
Yubkey locks the FIDO2 Pin by default. You should follow these steps to unlock it first before running through the above tutorial. Note they assume you have installed the PPA in the above tutorial above first.

Install the YubiKey GUI manager, then use the options as shown below.



Code :

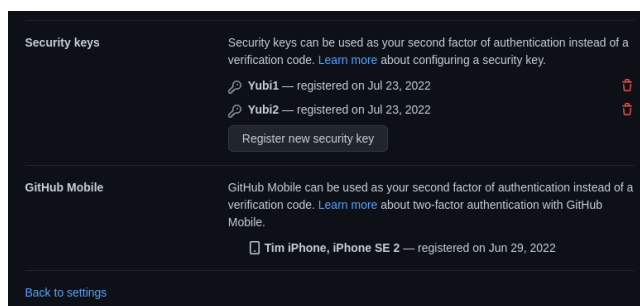
```
sudo apt install yubikey-manager-qt
ykman-gui
```



1.1.3 Online Accounts

You need to have online accounts with the following services:

- [GitHub](#) - then set up your YubiKey as your 2FA [here](#). As a backup 2FA you should use the GitHub mobile app. Note that using SMS for 2FA is not considered secure.



- [Google](#). Set up your YubiKey as your 2FA [here](#). As a backup 2FA you should use the Google mobile app. Note that using SMS for 2FA is not considered secure.

← Security keys


Security keys are a more secure second step. You can add a physical key or use your phone's built-in key. [Learn more](#)

Your security keys













	iPhone (Added: June 17, 2021) Last used: June 15, 2021 iPhone	
	YubiKey2 (Added: Just now) Last used: -	 
	YubiKey1 (Added: Just now) Last used: -	 

[+ Add security key](#)

- [Hetzner](#). If you are a staff member with permission to access Hetzner, set up your YubiKey as your 2FA [here](#). Note that using SMS for 2FA is not considered secure.


ADMINISTRATION
 Profile
 Contact details
 Email addresses
 Data processing agreement
 Newsletter 
 Fault reports
 Invoices
 Overview
 Credit balance
 Payment details
 Settings
 Password
Two-factor authentication
 Delete user account

TWO-FACTOR AUTHENTICATION
 Two-factor authentication status: **ACTIVE** [DISABLE](#)

Status	Description	Method	
	[Redacted]	[Redacted]	
	[Redacted]	[Redacted]	
	[Redacted]	[Redacted]	
	[Redacted]	[Redacted]	
	Tim Home Yubikey	Yubikey	DISABLE 
	Tim Office Yubikey	Yubikey	DISABLE 

[Add new authentication method](#)

- ERNext. Our admin team will provision an account for you.
- NextCloud. Our admin team will provision an account for you. [NextCloud](#). If you are a staff member with permission to access Hetzner, set up your YubiKey as your 2FA [here](#). Note that using SMS for 2FA is not considered secure.


 Personal info
Security
 Activity
 External storages
 Mobile & desktop
 Accessibility
 Sharing
 Flow
 Privacy
 Administration
 Overview
 Basic settings
 Sharing
 Security
 External storages
 Theming

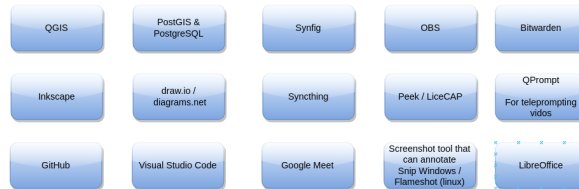
Backup code
 Backup codes have been generated. 0 of 10 codes have been used.
[Regenerate backup codes](#)
 If you regenerate backup codes, you automatically invalidate old codes.
TOTP (Authenticator app)
☒ Enable TOTP
WebAuthn device
 The following devices are configured for WebAuthn two-factor authentication:
 ✓ Yubikey1 ...
 ✓ Yubikey2 ...
[Add WebAuthn device](#)
Passwordless Authentication
 Set up your account for passwordless authentication following the FIDO2 standard.
 The following devices are configured for your account:
 ✓ Yubikey1 ...
 ✓ Yubikey2 ...
[Add WebAuthn device](#)

1.1.4 Software



KARTOZA
OPEN SOURCE GEOSPATIAL SOLUTIONS

Essential Software



1.1.5 Skills

1 GIS

1.1 Cartography Guidelines

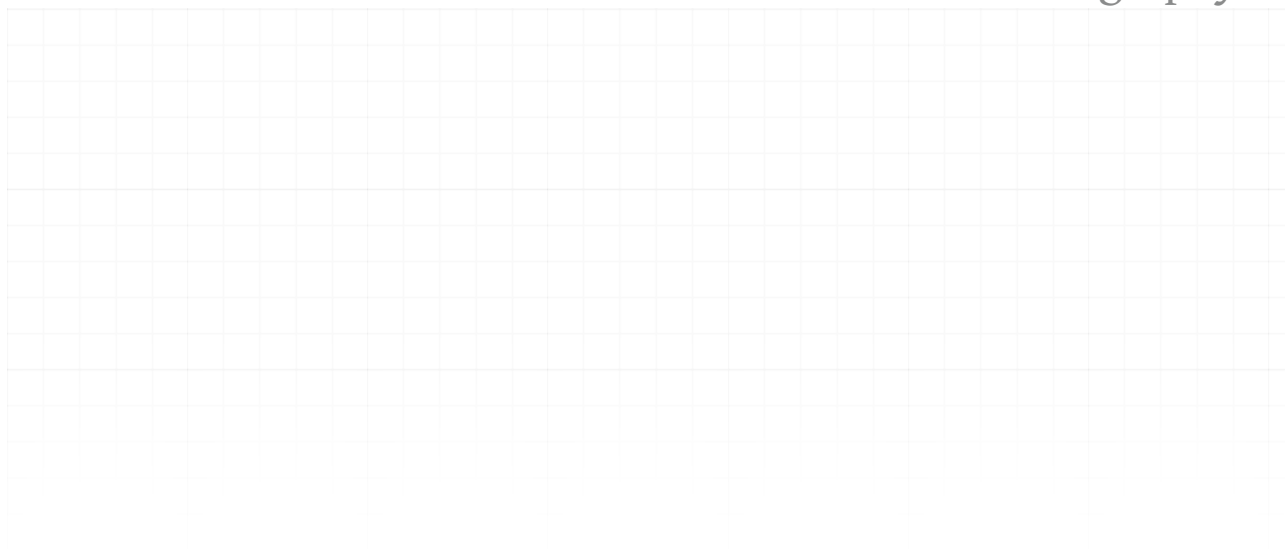
This section is a collection of rules and mantras used by Kartoza to Make beautiful and effective cartography.



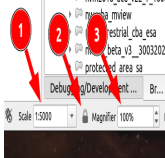

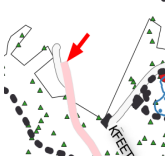
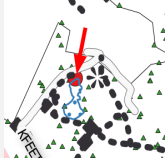

1.1.1 Steps to preparing a map


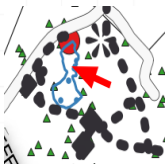
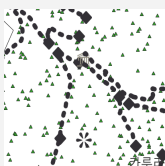
1. Choose your locality. Perhaps you have a client or an assignment which will determine the place. Or perhaps you have the freedom to choose the place yourself.
2. Choose the scale for your map. Again this may be client driven based on a specification or you might have the freedom to choose yourself. But decide up front what scale you will view the map at as it will influence all your decisions as you develop your map. In some cases you may be asked to make a variable scale product e.g. for a web map that can be zoomed in and out. If you are producing a variable scale map, determine which scale ranges and intervals will be used. For example, OpenStreetMap defines [standard scale intervals](#) which are used by many web mapping toolkits. If you are able to choose your own scale, try to choose a humanistic scale e.g. 1:50 000, 1:5000 etc.
3. Assuming you are preparing a fixed scale map, define the extent of your map. This may be determined by the print page size or by client factors. In some cases, the size of the print medium may influence the scale of your map. For example, you may be printing to an A4 map and need to choose a scale that allows you to see the whole town or area of interest.
4. Determine which data layers to feature in your map. Some typical examples will be roads, buildings, rivers, water bodies, points of interest etc. Make a list and then go out and procure your data. For each dataset you find, keep note of the following details in a spreadsheet or notes file:
 1. Date of download
 2. Source (name of person or organisation)
 3. Attribution (citation for the data)
 4. URL
 5. Notes
5. Determine the purpose of your map. Every map has a story to tell. Maybe you want to show a tourist around your city? Perhaps it is a narrative about crime or socio-economics.



1.1.2 Common issues in cartography



No.	Image	Description
1		Don't let labels overlap features.
2		Make sure that major roads cover minor roads.
3		Set your canvas scale to the scale you plan to print at 1 then lock it 2 then when you zoom in and out with your scroll wheel it will zoom into pixels at the fixed scale 3 .
4		Don't include roads on your map that end abruptly unless this indeed reflects reality.
5		Generally you should use a round cap style for your roads to avoid issues of the road cap extending beyond intersecting roads.
6		As a general rule, points should be drawn above polygon and line features.
7		Use a font that matches the language of your map.

No.	Image	Description
8		Use symbol layers to place a solid background behind symbols which are transparent and are getting lost in the background.
9		Do not use transparency inappropriately. Features like dams should be usually rendered with a solid fill.
10		Avoid visual clutter in your map. If there are too many features visible with similar contrast your user will quickly become overwhelmed.



1.1 Technologies

Outline of Key GIS Technologies that all GIS practitioners should be familiar with.

Outline GIS technologies and concepts for developers to get up to speed quickly.



1.1 GIS Resources

Where to find internal resource collections (projects, templates, data, assets etc)

External resource links



1 Development

1.1 Development

 Note:

Note that all users, regardless of role, should understand and review the [security](#) section.

Kartoza is a consulting and development firm. In many instances, projects require developers to leverage existing tools and codebases, and to work with other organisations in a way that is consistent in their own conventions.

This documentation seeks to outline processes and practices so that internal project development and practices within the team remain consistent. Where it is noted that beneficial conventions listed here are not implemented in client projects, recommendations may be made that clients adopt such standards.

For the most part, however, it is less a concern of how a particular outcome is achieved, but rather that the result is consistent with the appropriate conventions.

- [Conventions](#)
- [Technologies](#)
- [Environments](#)



1.1 Conventions

<https://kartoza.com/en/coding-standards/>

- [IDEs](#)
- [Processes](#)
- [Git](#)

1.1.1 Project Conventions

SDLC, agile, scrumboards: i.e. [project processes](#)

1.1.2 Language Conventions

[A foolish consistency...](#)

1.1.2.1 Python

python specific content

1.1.2.2 JavaScript

js specific content

1.1.2.3 C++

c++ specific content



1.1 Technologies

It makes sense to attempt some level of standardisation across development practices, including technologies and frameworks in use, so that the barrier for entry to projects is lowered and it becomes simple to maintain consistent practices across the organisation.

- [Languages](#)
- [Frameworks](#)

1.1.1 Evaluating

It may be helpful to include some "smell tests" for evaluating good projects or technologies



1.1 Environments

Development environments configurations and conventions (conda/ venv/ poetry/ git codespaces/ docker workspaces etc)

[Links and external resources](#)



1 DevOps

1.1 DevOps

Resources, information, and processes related to DevOps and system administration.

 Note:

Note that all users, regardless of role, should understand and review the [security](#) section.

Note that this is not limited to DevOps team members and may include conventions and configuration for the management of workstations, servers, and other systems, which is relevant to multiple roles.

- [Security](#)
- [Procedures](#)
- [Infrastructure](#)



1.1 Security

- Cyber Hygiene Handbook
- Security Guidelines/ Rules
- Development Processes
- [External Resources](#)

1.1.1 Cyber Hygiene Handbook

'twould be handy to collect resources and advice on training users in best practices for cyber-hygiene (applies in broad strokes to staff, trainees, community, and more). Compiled into a nice concise resource with pretty pictures and minimal complexity++

Topics and structure could be outlined as follows:

Users:

- Identifying phishing attacks
- Credential management

Admins:

- Identifying whats on the network
- Account management
- Staying up to date

Developers:

- Authentication management
- Dependency management
- Scanning
- SSH, SSL, SSO (lots of SS), Cryptography, Hashes, Salts



1.1 Infrastructure

- Development Infrastructure
- [Personal Infrastructure](#)
- Containers
- Kubernetes
- Rancher
- Rancher Desktop
- Development
- Staging
- Production
- CI/CD

1.1.1 Rancher Topics

[Using Rancher, K3S, and Longhorn for single node Kubernetes deployment](#)



1 Resources

1.1 Resources

- [Cheatsheets](#)
- [External Resources](#)
- [Kartoza Media Center](#)
- [Tutorials](#)



1.1 Cheatsheets

 Note:

This article is under heavy development and is not considered production ready

Cheatsheets and snippets for important and common operations.

- [PostgreSQL](#)
- [BASH](#)
- [Kubectl](#)
- [OSGeo4W](#)



1.1 Tutorials

Tutorials.

- [General](#): General tutorials
- [QGIS](#): Tutorials related to QGIS
- [Links](#): Links to third party and external tutorials and learning resources



1.1 Kartoza Media Center

Welcome to the Kartoza Media Center.

Here you can find links to media releases, social media feeds, and notices from Kartoza (Pty) Ltd

- [Kartoza.com](https://kartoza.com)
- [GitHub](#)
- [Twitter](#)
- [DockerHub](#)
- [Newsletters](#)



<https://github.com/kartoza/TheKartozaHandbook>