Data Management Plan

Washington Soil Health Initiative: State of the Soils Assessment

Jadey Ryan

Dani Gelardi

2024-01-03

Table of contents

# 1. Overview

The [Washington Soil Health Initiative](https://washingtonsoilhealthinitiative.com/) (WaSHI) is a partnership between the Washington State Department of Agriculture (WSDA), Washington State University (WSU), and the State Conservation Commission. WaSHI establishes a coordinated approach to healthy soil in Washington.

To date, nearly 1,000 soil samples and management surveys across 50 different cropping systems have been collected as a part of the [state of the soils assessment](https://washingtonsoilhealthinitiative.com/state-of-the-soils/) (SoSA). WSDA and WSU lead this project with support from staff, students, conservation districts, and agricultural professionals throughout Washington.



## 1.1 Chapter outline

*This Data Management Plan (DMP) is a living document will be continually reviewed and improved based on lessons learned, new information, and collaborator feedback.*

## 1.2 Roles and responsibilities

All WaSHI personnel who will be interacting with SoSA data must familiarize themselves with the contents of this document. Following chapters with technical details will be referenced when relevant. If all collaborators are not consistently implementing this DMP, then the benefits of effective data management are lost.

The WSDA Data Scientist, supported by the project Principal Investigators (PIs), is responsible for providing guidance to WaSHI staff working with SoSA data and ensuring the implementation of the DMP. The Data Scientist is also responsible for reviewing and updating this document annually, and as needed. Upon updates, the Data Scientist will distribute this document to WaSHI staff and commit the source code to the [GitHub repository](https://github.com/WA-Department-of-Agriculture/washi-dmp).

Current roles as of November 2023

| Role | Affiliation | Name |
| --- | --- | --- |
| CoPI | WSDA | Dani Gelardi |
| CoPI | WSU | Deirdre Griffin LaHue |
| Data Scientist | WSDA | Jadey Ryan |
| Data Stewards | WaSHI personnel |  |

## 1.3 Acknowledgements

This DMP was adapted from the R.J. Cook Agronomy Farm Long-term Agroecological Research Site DMP (Carlson 2021), U.S. Fish and Wildlife Service data management life cycle (U.S. Fish & Wildlife Service 2023), Harvard Medical School Longwood Research Data Management DMP guidelines (Harvard Medical School 2023), and the Data Management in Large-Scale Education Research book (Lewis 2023).

# 2. What is data management?

Effective data management involves properly documenting, storing, and sharing our data and the information we derive from the data. If the data aren’t useable by researchers, policymakers, or growers, the hours spent in the field collecting soil samples and the hours spent in the lab analyzing them may be wasted.

The guidelines detailed in this DMP will help us achieve the above data-driven goals, while also optimizing the value of the data by supporting information sharing and innovation. Our data management policies attempt to follow **FAIR** (**F**indable, **A**ccessible, **I**nteroperable, **R**eusable) principles while also maintaining data privacy (Wilkinson et al. 2016).

## 2.1 Data life cycle

The U.S. Fish and Wildlife Service developed a great graphic to explain the elements of the data life cycle and emphasize the importance of data quality at every step (U.S. Fish & Wildlife Service 2023). Each step within the data life cycle requires careful intention to ensure transparency, quality, and integrity.

|  |
| --- |
|  |

Our adaptation of this data life cycle is outlined below.

|  |  |
| --- | --- |
|  | **Plan**  Each sampling year presents an opportunity to consider what worked and what could be improved from the previous year. Planning involves making decisions about data acquisition, management, and quality control. For example, each year we provide a spreadsheet template with our requested column headers to Soiltest lab to ensure the measurements are reported with correct units and in the format we use. Special projects that deviate from our standard operating procedures require additional planning. |
|  | **Acquire**  We acquire data by collecting and analyzing new samples, deriving new insights from existing data, or accepting datasets from collaborators. | |
|  | **Maintain**  Maintenance involves processing data for aggregation, analyses, and reporting. We create metadata that facilitates interpretation of the data and ensure the data are in a non-proprietary format that is accessible to our collaborators and future selves. |
|  | **Access**  Access refers to data storage, publication, and security. Raw and processed data with accompanying metadata should be stored, backed up, and available for information sharing with our partners. With PI approval, anonymized and aggregated data that does not compromise growers’ personally identifiable information (PII) should be made publicly available in a data repository or data product/decision-support tool. |
|  | **Evaluate**  We evaluate data while processing and analyzing it to maximize accuracy and productivity, while minimizing costs associated with errors or tedious data cleaning labor. Evaluation workflows should be efficient, well-documented, and reproducible. Our evaluated data help us better understand how factors and management decisions impact soil health. |
|  | **Archive**  Properly archiving our results supports the long-term storage and usefulness of our data. While similar to the Access element of the life cycle, archiving focuses on preserving data for long-term/historical retention that aren’t needed for immediate access. For example, we archive each year’s raw data for long-term storage and set those files to Read-Only. |
|  | **Quality Assurance / Quality Control (QA/QC)**  Data quality management prevents data defects that hinder our ability to apply data towards our science-based conservation efforts. Defects include incorrectly entered data, invalid data, and missing or lost data. QA/QC processes should be incorporated in every element of the data life cycle. |

The following chapters describe our internal processes and standards to follow throughout each step in the data life cycle.

# 3. Formats & standards

## 3.1 Data formats

Data generated from or integrated into WaSHI can be non-digital or digital.

### Non-digital data

Non-digital data, such as field forms, management surveys, and chain of custody forms, are manually recorded on paper forms. Paper forms must be transcribed or converted to digital file formats and then stored in the WaSHI filing cabinet in the Natural Resources Building in Olympia.

### Digital data

Digital data include tabular, spatial, and binary data, such as lab results, sample locations, and field photos. Non-conventional data also include code, algorithms, tools, and workflows.

**Tabular data** include comma separated values (csv), tab separated values (tsv), Microsoft Excel open XML spreadsheet (xlsx), and portable document format (pdf).

**Spatial data** include file geodatabases (gdb), vector shapefiles (zipped folder containing multiple file extensions), keyhole markup language (kml or kmz). Tabular data may also contain spatial data as longitude and latitude.

**Binary data** include photos (jpeg, png, gif, tiff), videos (mp4), code (R, py, js), and object-oriented data files (RDS, Rdata, parquet, arrow).

**Proprietary data formats** include Microsoft Excel, Word, and Powerpoint files (xlsx, docx, pptx). RDS and Rdata files are an example of an application-specific data format that can only be opened using the R programming language or RStudio IDE. These types of files should be saved in conjunction with a copy of the data in a non-proprietary and open-standard format, such as csv, to maintain accessibility for those who do not have Microsoft Office or do not use R.

**Written documents and presentations** are in formats including Microsoft Word and Powerpoint (docx and pptx), hypertext markup language (HTML), and pdf.

**Notebooks** combine text with executable code to generate written documents and presentations in docx, pptx, HTML, or pdf formats. These notebooks are stored in formats depending on the programming language: a few examples include R markdown (rmd), Quarto (qmd), and Jupyter notebook (ipynb).

The list below is not exhaustive and will continue to grow as additional useful data sources are discovered.

| **Type** | **Source** | **Formats** |
| --- | --- | --- |
| **Lab results** | Provided by the lab analyzing the soil sample, principal investigator of a study, or grower | csv, xlsx, pdf, xml, json, RDS, RData |
| **Management surveys** | Collected through interviews with grower | csv, xlsx, RDS, RData, paper form (to be digitized) |
| **Field forms** | Completed in the field during/immediately after sampling | pdf, paper form (to be digitized), csv, xlsx |
| **Sample locations** | Identified prior to sampling and may be edited during sampling using ArcGIS Online, Collector, Field Maps or Google Maps | ArcGIS feature layer, shp, kmz, csv, xlsx |
| **Chain of custody forms** | Completed prior to shipping or dropping off samples | pdf, paper form (to be digitized) |
| **Climate data** | [OSU PRISM](https://prism.oregonstate.edu/), [NOAA](https://www.ncdc.noaa.gov/cdo-web/datasets), [Esri Living Atlas](https://livingatlas.arcgis.com/en/home/) | csv, shp, netCDF, tiff, gdb |
| **Soil data** | [NRCS Web Soil Survey](https://websoilsurvey.sc.egov.usda.gov/App/HomePage.htm), [NRCS WA gSSURGO](K:\NRAS\Arc_Data\Soil_Health\NRCS_StatewideSoils) | gdb, accdb |
| **Strata classification** | Provided by Soil Health Institute in 2021 as a [lyr file](K:\NRAS\Arc_Data\Soil_Health\SoilHealthInstitute_Strata) | lyr |
| **Images** | Logos, icons, photos taken in the field | jpeg, png, gif, tiff, svg |
| **Videos** | Recordings of meetings, training videos | mp4 |
| **Documents** | Reports, manuscripts, SOP, QAPP, factsheets, brochures | docx, txt, html, pdf |
| **Presentations** | Powerpoints, slide decks | pptx, html, pdf |
| **Code** | Scripts for wrangling, processing, analyzing data; markdown for producing documents and presentations; style sheets for html outputs | R, py, ipynb, js, yml, rmd, qmd, css, scss |

## 3.2 Data standards

**Date** will be expressed as YYYY-MM-DD according to the [ISO 8601 standard](https://www.iso.org/iso-8601-date-and-time-format.html).

**Date with time** will be expressed as YYYY-MM-DD**T**HH:MM:SS**Z**.

* **T** separates date from time. The **Z** indicates the date-time is using the Universal Time Coordinated (UTC) with no offset.
* Pacific Standard Time (PST) has a UTC-8:00 offset and Pacific Daylight Time (PDT) has a UTC-7:00 offset and would be expressed as YYYY-MM-DD**T**HH:MM:SS-8:00. The **Z** has been replaced with the offset.
* Example: 2023-11-28T14:55:56-08:00.

|  |
| --- |
| "ISO 8601" from Randall Munroe's xkcd |

**Geospatial** data will be accompanied by metadata that abides by the [ISO 19115 standard](https://www.iso.org/standard/53798.html) by following Esri’s [documentation](https://pro.arcgis.com/en/pro-app/latest/help/metadata/create-iso-19115-and-iso-19139-metadata.htm) when working in ArcGIS Pro. Metadata contains information about the identification, the extent, the quality, the spatial and temporal schema, spatial reference, and distribution of digital geographic data.

**Code** will follow the style guide in [Chapter 10](#sec-code-style-guide).

# 4. Naming conventions

When naming folders and files, we want consistent and clear names that are findable and understandabe by both humans and computers. From only a file name, we should immediately know what the file contains and which file is the most recent version.

|  |
| --- |
| “Documents” from Randall Munroe’s xkcd |

## 4.1 Why are conventions important?

* Improves consistency and predictability, making it easier to browse folders and know what the file/folder contains.
* Easily sort and retrieve files by date, conservation district, or other theme.
* Facilitates collaboration so all team members can find the information they need.
* Standardizes file paths and URLs for efficient programming and website hosting.
  + URLs and programming languages are case-sensitive. WaSHI-data.csv and washi-data.csv are completely different files.
  + URLs cannot have spaces in them. They must be escaped with this character entity %20. wasoilhealth.org/producer spotlights would need to be wasoilhealth.org/producer%20spotlights.

For more web-specific naming conventions, see this [Learn the Web webpage](https://learntheweb.courses/topics/naming-conventions/).

## 4.2 Best practices

*Many files and folders in our shared drive do not follow the above best practices or below naming conventions. We are learning and improving as we go.* Going forward, please follow the conventions listed below.

These are guidelines and naming things is hard, so try your best. If you’re not sure how to name a folder or file, talk with Jadey. If you’re adding a bunch of files that came from somewhere else, ask Jadey to help you organize and bulk rename them.

See [Section 4.3](#sec-naming-examples) for a table of examples of folder and file names using these best practices.

### Lowercase only

Different conventions work better for different purposes (folders and file versus programming objects).

* **kebab-case**: all lowercase with hyphens separating words. Use for all folder and file names.
* **snake\_case**: all lowercase with underscores separating words. Only use for functions and variables in R. See [Section 10.2.3.2](#sec-objects-functions) for example R errors when including hyphens in object names.

|  |
| --- |
| Cartoon representations of common cases in coding. A snake screams “SCREAMING\_SNAKE\_CASE” into the face of a camel (wearing ear muffs) with “camelCase” written along its back. Vegetables on a skewer spell out “kebab-case” (words on a skewer). A mellow, happy looking snake has text “snake\_case” along it. Artwork by [@allison\_horst](https://twitter.com/allison_horst). |

### Delimiters convey meaning

Deliberately use underscores and hyphens so we can easily understand the contents and programmatically parse file and folder names.

* Use underscores to delineate metadata elements (i.e. date from name from version date\_name\_version).
* Use hyphens to separate parts of one metadata element (i.e. date YYYY-MM-DD or name wsda-washi-presentation).

### No spaces or special characters

Avoid spaces and special characters (only use underscores and hyphens). Characters like / () ! ? % + " ' have special meaning to computers and can break file paths and URLs.

### Character length matters

Computers are unable to read file paths and file names that surpass a certain character length. Be concise AND descriptive. Windows path limit is 260 characters.

### ‘Back to front’ date

Remember to express date ‘back to front’ of YYYY-MM-DD according to the [ISO 8601 standard](https://www.iso.org/iso-8601-date-and-time-format.html). This maintains the chronological order of records when they are sorted alphanumerically.

| ✅ Do this | ❌ Don’t do this |
| --- | --- |
| 2020-05-28\_agenda.pdf | 2-14-2023\_Agenda.pdf |
| 2023-01-01\_agenda.pdf | 2023-Jan-1\_Agenda.pdf |
| 2023-02-14\_agenda.pdf | Dec052020\_Agenda.pdf |
| 2023-12-05\_agenda.pdf | May\_28\_2020\_Agenda.pdf |

### Group & sort files by name

Consider how folders and files should be grouped and sorted. Put that piece of metadata in the beginning of the file name.

| Sort by district | Sort by date |
| --- | --- |
| cowlitz-coc\_2023-05-01.pdf | 2023-05-01\_cowlitz-coc.pdf |
| cowlitz-coc\_2023-05-23.pdf | 2023-05-01\_cowlitz-tracking.pdf |
| cowlitz-tracking\_2023-05-01.pdf | 2023-05-09\_ferry-cd-tracking.pdf |
| ferry-cd-coc\_2023-05-10.pdf | 2023-05-10\_ferry-cd-coc.pdf |
| ferry-cd-coc\_2023-05-17.pdf | 2023-05-17\_ferry-cd-coc.pdf |
| ferry-cd-coc\_2023-06-06.pdf | 2023-05-23\_cowlitz-coc.pdf |
| ferry-cd-tracking\_2023-05-09.pdf | 2023-06-06\_ferry-cd-coc.pdf |

### Version numbers

If not using the date to version, or to keep multiple drafts from the same date, add version information to the end of the file name. Think about how many possible versions there could be. If there may be more than 10, use leading zeros before single digit numbers so the file name always has the same length. V1 through V15 will not sort the same way as V01 through V15.

| ✅ Do this | ❌ Don’t do this |
| --- | --- |
| sop\_v01.pdf | SOP\_V1.pdf |
| sop\_v02.pdf | SOP\_V10.pdf |
| … [v03 - v09] | SOP\_V11.pdf |
| sop\_v10.pdf | SOP\_V2.pdf |
| sop\_v11.pdf | … [V3 - V9] |

### Collaboration

Add your initials at the end of the file name when “saving as” a file that multiple people are working on (i.e., 2023-11-14\_sop-soil-health-monitoring\_lm\_jr.docx). This ensures a version is kept as a backup. Alternatively, use [Track Changes](https://support.microsoft.com/en-au/office/track-changes-in-word-197ba630-0f5f-4a8e-9a77-3712475e806a) if working in a MS Word document.

### Code and variables

Instead of kebab-case, we use snake\_case for all code objects (R vectors, lists, dataframes, functions, etc.) and variables (column names). Hyphens in object names can cause errors in R and SQL code.

See [Chapter 10](#sec-code-style-guide) for more details in the code style guide.

## 4.3 Naming examples

|  | Naming convention | Examples |
| --- | --- | --- |
| **Folders** | kebab-case | 2024\_sampling  data-management |
| **Files** | kebab-case | 2023-11-15\_survey-perennial.xlsx  records-management\_v01.docx  pend-oreille-cd\_coc\_2023-06-05.pdf  washi-logo-color.png  washi-dmp.Rproj  01\_load-metadata.R  2024\_producer-report.qmd |
| **Code and variables** | snake-case | sample\_id  toc\_percent  pmn\_mg\_per\_kg  crop\_summary  assign\_quality\_codes() |

# 5. Organization

We organize our folders into a hierarchical structure to clearly delineate segments of our projects, improve searchability, and ensure reproducibility across years.

## 5.1 Folder Structure

We strive for a balance between a deep and shallow structure. If too shallow, there are too many files in one folder and they are hard to sort through. If too deep, we have to click too many times to get to a file and specific files can be difficult to find.

Y:\NRAS\Soil\_Health\_Initiative is the parent folder for all WaSHI content.

Within the StateOfTheSoils sub-folder, we use a combination of **date-** (each year has its own sub-folder) and **categorical-** based (dataset and documentation that span across years) folder structures.

Y:\NRAS\Soil\_Health\_Initiative\StateOfTheSoils  
├── \_completeDataset  
├── 2019\_SCBG  
├── 2021\_Sampling  
├── 2022\_PartnershipsInSoilHealth  
├── 2023\_Sampling  
├── 2024\_Sampling  
├── DataManagement  
├── DataSharing  
├── DataSources  
├── Maps  
├── Projects  
├── QAPP  
├── SOPs  
├── TrainingVideos  
├── ArchivedSampleInventory.xlsx  
├── EquipmentInventory.xlsx  
└── SOSImpacts.xlsx

Within the each year sub-folder, we have sub-sub-folders for planning, forms, data, and processes. This structure helps maintain a reproducible workflow year after year. See the 2023\_Sampling for an example:

Y:\NRAS\Soil\_Health\_Initiative\StateOfTheSoils\2023\_Sampling  
├── Applications  
├── CoCs  
├── Equipment  
├── FieldForms  
├── Forms  
├── GIS  
├── LabData  
├── Labels  
├── ManagementSurveys  
├── PublicDocs  
├── Purchases  
├── Reports  
├── SampleIDAssignments  
├── Scripts  
├── 2023\_DataTracking.xlsx  
└── PostSeasonWrapUp\_2023.docx

As mentioned in [Section 6.2](#sec-raw-data), it’s good practice to maintain the raw data. We use additional sub-folders for the LabData folder. Everything in Raw has been set as Read-Only.

Y:\NRAS\Soil\_Health\_Initiative\StateOfTheSoils\2023\_Sampling\LabData  
├── 2023\_DataTemplateSoiltest.xlsx  
├── Clean  
├── QC  
├── Raw  
└── Working

Soil\_Health\_Initative > StateOfTheSoils > 2023\_Sampling > LabData > Clean already has five levels of nesting. We wouldn’t want to add any many more levels or the hierarchy becomes difficult to manage.

## 5.2 Archive folders

When too many drafts or versions begin to clutter a sub-folder, create a new folder with the naming convention of Archive\_FolderDescription. Place the old drafts there. Leave the most current, accurate file in the main folder.

For example, the most recent sample labels for each conservation district are listed in the top level CompletedLabels folder, and previous working drafts were moved to the Archive\_Labels folder.

Y:\NRAS\Soil\_Health\_Initiative\StateOfTheSoils\2023\_Sampling\Labels\CompletedLabels  
├── Archive\_Labels  
│ ├── CowlitzCounty\_Labels.docx  
│ ├── FerryCD\_Labels.docx  
│ ├── LewisCD\_Labels.docx  
│ └── StevensCD\_Labels.docx  
├── CowlitzCounty\_Labels\_V2.docx  
├── FerryCD\_Labels\_V2.docx  
├── KittitasCD\_Labels.docx  
├── LewisCD\_Labels\_V2.docx  
├── ...  
└── WallaWallaCD\_Labels.docx

## 5.3 Code-based project organization

Code-based projects should be organized according to [Section 10.1.1](#sec-project-folder-structure) in the code style guide.

# 6. Storage

Non-digital data, such as paper forms, must be transcribed or converted to digital file formats and then stored in the WaSHI filing cabinet in the Natural Resources Building in Olympia.

All digital data are stored in the WSDA shared drives, and other locations listed below.

**WSDA shared drives:**

* Agency files: <Y:\NRAS\Soil_Health_Initiative>
* GIS: <K:\NRAS\Arc_Data\Soil_Health> (access requires permissions from IT)

**Esri products and services:**

* ArcGIS Online [Soil Health - WSDA Internal Group](https://nras.maps.arcgis.com/home/group.html?id=17a60cfa644c4c60ab622fdd84500f8f#overview)
* WSDA GIS on-premise [ArcGIS REST Services Directory](https://fortress.wa.gov/agr/gis/wsdagis/rest/services/NRAS) (only Jadey, Perry, and Joel can publish to this server; Ed Thompson is the contact for getting access)

**Database for lab results and management data:**

* WISKI, but very likely will migrate to SQL Server or a less water-focused database

**GitHub organizations for code-based projects:**

* [WSDA](https://github.com/WA-Department-of-Agriculture)
* [WaSHI](https://github.com/WASoilHealth)

**Microsoft Teams for data sharing between WSDA and WSU:**

* WSDA and WSU Teams WaSHI channels

**Box.com for external file sharing:**

* WSDA has a box.com account. Only Perry has a license; Jadey and Dani are currently editors.

**Individual devices (laptop, tablet, phone):**

* Must NOT be the only place data are stored!

## 6.1 Backup

Data must be stored in multiple locations. At a bare minimum, data on an individual computer must also be saved on the WSDA shared drive. Backing up data using version control (GitHub) or a cloud service (Microsoft OneDrive or Box.com) is strongly recommended.

## 6.2 Read-only raw data

On our shared drives, raw data such as lab results from Soiltest or exports from ArcGIS Online, should be immediately set to Read-only. Right click the file > click on Properties > check the Read-only attribute box.

|  |
| --- |
|  |

The file should then be copied over to a Working folder for any processing or analyses. The final dataset should be saved in a separate descriptively titled Clean folder. Keeping a readme.txt to document your processing and analysis steps is good practice, as discussed in [Section 7.2.1](#sec-readme).

## 6.3 Version control with Git and GitHub

A version control system records changes to a file or set of files over time. [Git](https://git-scm.com/) is a free and open-source distributed version control system and [GitHub](https://github.com) is the hosting site WSDA and WaSHI use to interface with this system. Git and GitHub are an important foundation of reproducible statistical and data scientific workflows (Bryan 2018).

A major benefit of using version control is ensuring changes are well documented and previous versions are accessible if any changes must be recalled. Additionally, version control makes collaboration across projects much more robust.

Version control is not just for code either! It’s useful for scripts, documents, presentations, and books (like this DMP!). Instead of saving each version of a file with a different name (i.e., Report\_V01.docx and Report\_V02.docx; for a reminder on version naming, see [Section 4.2.7](#sec-version-numbers)), there’s only one file Report.docx which automatically has its history and editors saved with Git and GitHub.

In the screenshot below, you can see who made a commit (which is basically a named version of changes), when that commit was made, and you can click on the commit message to view all of the files that were changed.

|  |
| --- |
|  |

After clicking on the first commit message, we see the documentation.qmd file was changed with additions highlighted in green and deletions highlighted in red.

|  |
| --- |
|  |

### Git and GitHub resources

Read Jenny Bryan’s article [*Excuse Me, Do You Have a Moment to Talk About Version Control*](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\DataManagement\Resources\Bryan_ExcuseMeDoYouHaveMomentTalkVersionControl.pdf) for a great background on Git and GitHub, why we should be using it, and a brief how to get started. For detailed instructions , please follow along with Jenny Bryan’s free online book [*Happy Git and GitHub for the useR*](https://happygitwithr.com/). Another helpful book resource is [*GitHub: A Beginner’s Guide*](https://birdscanada.github.io/BirdsCanada_GitHubGuide/), which was created by Birds Canada (avian conservation NGO) for people without a lot of programming background.

If you prefer to look through slides, check out Byron C. Jaeger’s presentation [*Happier version control with Git and GitHub (and RStudio)*](https://bcjaeger.github.io/seminar-git/)*.*

## 6.4 Staff turnover

When staff leave our group, they take their skills, institutional knowledge, and personal understanding of their file management with them. Proper offboarding is essential to ensure knowledge isn’t lost, time isn’t wasted trying to recreate workflows, and projects keep moving.

Before the employee leaves, the Senior Soil Scientist and Data Scientist ensure that:

* Folders and files are moved from the employee’s personal drive to the shared drive. They are named and organized according to [Chapter 4](#sec-naming).
* Workflows and specific processes the employee was responsible for are well documented.
* Permissions and ownerships are transferred to the appropriate remaining staff.
  + [GitHub WSDA organization](https://github.com/WA-Department-of-Agriculture)
  + ArcGIS data products and online groups
  + Database credentials
  + Box.com folders

More resources and offboarding checklists from [Harvard Research Data Management](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\DataManagement\Resources\StaffTurnover) can be found in our [DataManagement shared drive](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\DataManagement\Resources\StaffTurnover).

# 7. Documentation

Documentation is the process of recording all aspects of project design, sampling, lab analysis, data cleaning, data analyses, data quality control and assurances procedures, and development of decision-support tools. Seem familiar? All of these aspects are also part of our data life cycle. We document our data throughout the life cycle to:

* standardize our procedures
* enable reproducibility
* establish credibility
* ensure others (including our future selves) use and interpret data correctly
* provide searchability

*All documentation (including this document) should be updated (and versioned) as procedures change and lessons are learned.*

Samples collected by WSDA for the SoSA should follow the procedures and standards described in the below documentation.

External data must have at least the documentation outlined in [Section 7.4](#sec-external-data) to be integrated into the SoSA dataset.

## 7.1 Project-level

Project-level documentation includes all descriptive information about the SoSA dataset, as well as planning decisions and process documentation. Documentation includes quality assurance project plans (QAPP), standard operating procedures (SOP), and other high-level documents (i.e., request for proposals, applications, meeting agendas/notes, etc.).

### Quality assurance project plan (QAPP)

The QAPP is the highest level of project documentation and covers everything from the project description, personnel roles and responsibilities, project timelines, data and measurement quality objectives, study design, and overviews of field, laboratory, and quality control.

Ours can be found in <Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\QAPP>, though it needs to be updated.

### Standard operating procedures (SOP)

SOPs provide detailed instructions for field, lab, or data processing procedures and decision making processes.

All sampling related SOPs can be found in <Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\SOPs>.

#### SoSA sampling

The purpose of this SOP is to specify the procedures for a typical site visit in which soil sampling is conducted to measure physical, chemical, and biological soil health indicators. Procedures include equipment preparation prior to sampling, best practices for filling out field forms, the selection of sampling locations, sampling protocols, sample handling and storage, and submitting samples to the lab. This SOP serves to ensure data quality by creating audit trails and enabling verification that data are present, complete, and accurate. Additionally, this SOP will be used to maintain consistent sample collection procedures throughout the state for WSDA employees and partners.

#### Quality control / quality assurance (QA/QC)

This SOP outlines the process for screening sample metadata and lab results for completeness, consistency, and quality. Procedures involve subject matter expertise, investigation, communication with sampling teams and labs, algorithmic quality control, and tagging sample results with quality codes (as shown in the below table). Data are then integrated into the statewide database according to a SOP not yet authored.

#### SOPs we don’t have (yet)

* assigning sample, producer, field IDs
* data cleaning
* data integration into database
* data storage
* external data integration

## 7.2 Dataset-level

Dataset-level documentation applies to lab results, sample locations, grower information, and management data. We use readmes and changelogs to document what each dataset contains, how they are related, potential issues to be aware of, and any alterations made to the data.

### Readme

readme files are plain text documents that contain information about the files in a folder, explanation of versioning, and instructions/metadata for data packages. These files are saved as .txt, not as MS Word documents that take longer to open and can only be opened on computers with Microsoft installed.

#### Describe contents of folder

For example, the \_completeDataset folder contains a [readme.txt](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\_completeDataset\readme.txt) that describes each of the files’ structure, contents, and other pertinent information, such as the data source (i.e. PRISM mean annual precipitation and temperature are 30-year normals from 1991-2020 at 800 m resolution).

#### Explain versions

Another example is the [readme.txt](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\2023_Sampling\LabData\Raw\readme.txt) in the 2023\_Sampling > LabData > Raw folder, which explains why there are two different versions of the lab results and where to find additional information.

#### Provide instructions

An example of a [readme.txt](https://app.box.com/s/jl714sbos0jus6mr9zesqsfpv718j783) that provides instructions on how to use the folder contents can be found in the [ArcGIS soil sample points box.com folder](https://app.box.com/s/4d6xg8q5dr1cjjoxver0g3h35d3s3bao) that is shared with outside partners.

### Changelog

Changelogs are also simple and concise plain text documents saved in a folder alongside data files to document any changes to the dataset.

At the bare minimum, the changelog.txt should contain:

* date of modification
* initials of who made the changes
* description of the changes

## 7.3 Variable-level

Variable-level documentation includes data dictionaries and code books, which are often talked about interchangeably. However, we’ll refer to the *data dictionary* as a tabular collection of names, definitions, and attributes about the variables in a dataset created (ideally) in the planning phase of the project before data are collected. In contrast, *codebooks* provide descriptive, variable-level information and univariate summary statistics to allow users to understand the contents of a dataset without opening it. The codebook is created or updated after data are collected, cleaned, and validated.

### Data dictionary

In a data dictionary, each row is a different variable and each column is a different attribute of that variable. With a data dictionary, any user should be able to properly interpret each variable in our data.

Our [dataDictionary.xlsx](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils_completeDataset\dataDictionary.xlsx) in the [\_completeDataset folder](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils_completeDataset) contains two tabs (labResults and sampleLocations) that describe the attributes of each variable.

### Codebook

Codebooks provide more information (i.e., existing values/ranges and summary statistics) than the data dictionary and can be used to understand a very high-level summary of the processed data. There are many R packages that generate codebooks; however, we have not implemented this type of documentation for our project yet.

Crystal Lewis gave the lightning talk [*A Comparison of Packages to Generate Codebooks*](https://github.com/Cghlewis/codebook-pkg-comparison). I’d like to generate codebooks for our datasets once they live in a database.

## 7.4 External data

External data refers to any data not directly collected by WSDA or trained partners (e.g., WSU or conservation districts) that follow our SOPs. These can include other studies pre-dating WaSHI, special soil health surveys, or publicly available datasets.

The Data Scientist and Senior Soil Scientist will decide whether to integrate an external dataset case by case by considering the below questions:

* How does the study design fit into SoSA goals?
* Who collected the soil samples?
* What field procedures were used and how were they documented?
* Who analyzed the soil samples? With which methods and QA/QC procedures?
* Which pieces of metadata and management data accompany the lab results?
  + Farm, producer and field info[[1]](#footnote-152)
  + Sampling date
  + Sampling depth
  + Latitude and longitude
  + Production system (current crop, crop rotation, etc.)
  + Tillage, livestock grazing, irrigation, soil fertility and amendments, conservation practices, etc.
* Is there a data dictionary or codebook to describe the variables and measurements, units, missing values, etc.?

Generally, the external data should **1)** be well documented, **2)** be collected and analyzed by well-trained scientists and labs; and **3)** have adequate accompanying metadata and management data to facilitate interpretation of the results.

Some publicly available datasets to consider are listed in <Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\DataSources>.

# 8. Data flows

assign unique identifiers

labels

tracking sheet

arcgis (map, domains, field forms, surveys, script to back up)

cross-reference different sources of crop data (field form, survey, initial form)

lab data comes back - qa and cleaning

generate reports (save to box.com)

import to database

save to year folder and \_completeDataset, and WSU Teams channel

archive jars and tubes, update archive spreadsheets

# 9. Data sharing

Our data sharing policies promote FAIR principles such that our data are “*as open as possible, as closed as necessary*” (European Commission 2016). Data should be *open* enough to facilitate efficient re-use and avoid duplicating data collection efforts, enhance scholarly rigor, promote engagement across the research and public communities, and realize many other benefits (Whyte and Pryor 2011). However, data must also be as *closed* as necessary to protect grower privacy and to honor prior agreements with growers or other researchers.

SoSA relies on growers’ willingness to volunteer their fields for sampling and participate in the required management survey. Their willingness depends on their trust in WaSHI to protect their privacy. Only aggregated and anonymized results will be publicly available or shared. The below data privacy statement may be shared with potential participants:

### Data privacy statement

Data will be aggregated and reported in a way which mitigates personal identification of growers. Information will be used to understand broad impacts and characterize trends in soil health and production practices across regions. Results will not be reported in a way that makes individuals identifiable. Information collected in this survey may be subject to release in accordance with RCW 42.56 (Public Records Act).

Procedures for anonymizing data are detailed in [Section 9.2](#sec-maintain-confidentiality).

## 9.1 WaTech data categorization

Under the Washington state policy 141.10 (Securing Information Technology Assets), state agencies must classify data into categories based on the sensitivity of the data. WaTech provides [guidance](https://watech.wa.gov/Categorizing-Data-State-Agency) on the four categories of data.

#### Category 4: “Confidential information requiring special handling”

**WaTech**: Data requires strict handling requirements applied by statues (e.g. HIPAA) or regulations (e.g. rules on employee files).

**SoSA**: We don’t manage any data falling under this category.

#### Category 3: “Confidential information”

**WaTech**: Data includes “personal information” as defined in RCW 42.56.590 (security breaches) and RCW 19.255.010 (personal information disclosure). An individual’s first name or first initial and last name *in combination* with at least one of the following elements: social security number, driver’s license or Washington identification card number, or any account numbers that permit access to their financial account.

**SoSA**: While we do not collect any of the above elements in combination with grower names, we still protect individual and farm names, and latitude and longitude coordinates as confidential information.

#### Category 2: “Sensitive information”

**WaTech**: Data are intended for official use only and withheld unless specifically requested.

**SoSA**: Lab results and management surveys fall under this category. Access to this data requires a [data share agreement](#data-share-agreement).

#### Category 1: “Public information”

**WaTech**: Data is not covered in any of the above categories or is already released to the public.

**SoSA**: De-identified and aggregated data such as the number of soil samples and from which counties and crops they were collected fall under this category. For example, the [SoSA dashboard](https://nras.maps.arcgis.com/apps/dashboards/d4b2d135d47a4ff7a89285fecf67cef2) is publicly available and the map zoom is disabled at the 1:1,600,000 scale (counties level).

## 9.2 Maintain confidentiality

Under no circumstances should the following Category 3 data be released to external collaborators or made publicly available:

* farm name
* grower first and last name
* field names that contain street names or other identifying information
* latitude and longitude coordinates or other geospatial identifiers
* any information that identifies the individual farm or grower

Category 2 data should be anonymized and aggregated to honor our [data privacy statement](#data-privacy-statement) by either removing or replacing Category 3 confidential information with dummy data. The {randomNames} R package can be used to replace real names with fake names. Latitude and longitude should be rounded to a precision that does not identify the farm or fields sampled. See the example R script below, copied from the private [soils-internal GitHub repo](https://github.com/WA-Department-of-Agriculture/soils-internal/blob/main/R/anonymizeData.R).

R code to anonymize data

```{r}  
library(dplyr)  
library(readxl)  
library(janitor)  
library(randomNames)  
  
load("./data/\_completeDataset/2020-2023\_labResults\_wide.Rdata")  
load("./data/\_completeDataset/2020-2023\_labResults\_long.Rdata")  
crops <- read\_excel("./data/reference/crops.xlsx")  
points <- read.csv("./data/\_completeDataset/2020-2023\_sampleLocations.csv") |>  
 select(sampleId, longitude, latitude) |>  
 mutate(across(where(is.numeric), \(x) round(x, 0)))  
  
# Join lab data with gis data  
data <- left\_join(allResults\_wide, points)  
  
# Join data with crop dictionary to get crop groups  
data <- left\_join(data, crops, by = c("crop" = "Crop Type")) |>  
 select(-crop) |>  
 relocate("crop" = "Crop Group", .after = county)  
  
# Anonymize  
anonData <- data |>  
 # Farm name, producer name, sample ID  
 mutate(  
 farmName = paste0("Farm ", sprintf("%03d", cur\_group\_id())),  
 producerName = randomNames(which.names = "first",  
 sample.with.replacement = FALSE),  
 producerId = paste0(  
 paste0(sample(LETTERS, 3, replace = TRUE), collapse = ""),  
 "0",  
 paste0(sample(1:9, 1, replace = TRUE), collapse = "")  
 ),  
 fieldName = paste0("Field ", fieldId),  
 sampleId = paste0(  
 substr(year, 3, 4), "-", producerId, "-", fieldId  
 ),  
 .by = c(county, producerName, producerId)  
 ) |>  
 # county  
 mutate(  
 county = paste0("County ", cur\_group\_id()),  
 .by = c(county)  
 )  
  
# Grab 100 random samples  
anon\_subset <- anonData |>  
 slice\_sample(n = 100)  
  
write.csv(anon\_subset, file = "./data/\_completeDataset/exampleData.csv",  
 na = "",  
 row.names = FALSE)  
```

## 9.3 Data share agreement

SoSA CoPIs created a [Data Sharing and Scope of Work Agreement](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\DataSharing\StateoftheSoils_DataShareAgreement.pdf) that details the type of data to be shared, the scope of work in which the data may be used, and terms for using SoSA data.

Once the agreement has been signed by both CoPIs and the “Partnering Scientists”, the agreement should be saved in its own folder within [Y:/NRAS/Soil\_Health\_Initiative/StateOfTheSoils/DataSharing](Y:\NRAS\Soil_Health_Initiative\StateOfTheSoils\DataSharing). If this agreement is also part of a grant proposal, it should also be saved in its corresponding grant folder [Y:/NRAS/Soil\_Health\_Initiative/ContractsGrants/Grants](Y:\NRAS\Soil_Health_Initiative\ContractsGrants\Grants).

The request correspondence, code to subset the requested data, and the final dataset sent should all be saved in this folder. Internally documenting this information allows us to track publications and attributions resulting from this data sharing. This documentation also helps us track the broader impact of how our collaborators’ use the SoSA dataset.

## 9.4 Public access

Currently the only SoSA data publicly available are the counts of samples across the project, counties, and crop types displayed in the [ArcGIS Online Dashboard](https://nras.maps.arcgis.com/apps/dashboards/d4b2d135d47a4ff7a89285fecf67cef2).

A small, anonymized subset is included as example data in the [{washi}](https://github.com/WA-Department-of-Agriculture/washi) and [{soils}](https://github.com/WA-Department-of-Agriculture/soils) R packages for demonstration purposes.

**TODO**: discussion with team on the below ideas

In the future when the data are more mature and hosted in a proper database, we may publish an anonymized subset in a public repository such as:

* GitHub via an R package or Shiny app
* [Zenodo](https://zenodo.org/): integrates with GitHub and is citable with a DOI
* [Data.WA.gov](https://data.wa.gov/): open data portal for the State of Washington

More inspiration and ideas for enhancing data discoverability and sharing across the agricultural and soil health communities include:

* [USDA LTAR data dashboards](https://ltar.ars.usda.gov/data/data-dashboards)
* [CAF LTAR metadata tool](https://meta.cafltar.org/)

[Chapter 16](https://datamgmtinedresearch.com/share) of *Data Management in Large-Scale Education Research* discusses more considerations for data sharing and choosing public repositories (Lewis 2023).

## 9.5 Acknowledgments

All research and data partially or completely funded by WaSHI must include acknowledgements to the State of Washington. The following text should be included in all publications resulting from this funding:

Data were in part provided by the Washington Soil Initiative, which is supported by the State of Washington and administered by the Washington State Department of Agriculture, Washington State Conservation Commission, and Washington State University.

If WaSHI staff make [substantial scientific contributions](https://www.pnas.org/doi/10.1073/pnas.1715374115) to the manuscript, discuss the possibility of co-authorship credit.

# 10. Code style guide

This style guide is an adaptation of the [*tidyverse style guide*](https://style.tidyverse.org/) (Wickham 2022) and includes best practices from [*R for Data Science (2e)*](https://r4ds.hadley.nz/) (*R4DS*) (Wickham, Çetinkaya-Rundel, and Garrett 2023), [*Data Management in Large-Scale Education Research*](https://datamgmtinedresearch.com/style.html?q=farewell#style) (Lewis 2023), and other resources. **All WaSHI staff who code in R should thoroughly read and consistently implement this style guide.** Using consistent project structures, naming conventions, script structures, and code style will improve code readability, analysis reproducibility, and ease of collaboration.

Good coding style is like correct punctuation: you can manage without it, butitsuremakesthingseasiertoread. This site describes the style used throughout the [tidyverse](http://tidyverse.org/). It was derived from Google’s original R Style Guide - but Google’s [current guide](https://google.github.io/styleguide/Rguide.html) is derived from the tidyverse style guide.

All style guides are fundamentally opinionated. Some decisions genuinely do make code easier to use (especially matching indenting to programming structure), but many decisions are arbitrary. The most important thing about a style guide is that it provides consistency, making code easier to write because you need to make fewer decisions.

- Hadley Wickham in the [tidyverse style guide](https://style.tidyverse.org/)

## 10.1 Projects

All files associated with a given project (input data, R scripts, analytical results, figures, reports) should be kept together in one directory. RStudio has built-in support for this through **projects**. R projects bundle all the work in a portable, self-contained folder that can be moved around on your computer or on to other collaborators’ computers and still work. Learn more about projects in the [Workflow: scripts and projects chapter of *R4DS*](https://r4ds.hadley.nz/workflow-scripts.html#projects) and in Jenny Bryan’s article [*Project-oriented workflow*](https://www.tidyverse.org/blog/2017/12/workflow-vs-script/).

To create a project in RStudio, click File > New Project, then follow the steps in the below figure.

|  |
| --- |
|  |

### Project folder structure

Having a consistent and logical folder structure will make it easier for you (and future you) and collaborators to make sense of the files and work you’ve done. Well documented projects also make it easier to resume a project after some time away with minimal frustration of having to remember where everything is, what you did, and why you did it.

The below structure works most of the time and should be used as a starting point. However, different projects have different needs, so add and remove subfolders as needed.

* **root**: top-level project folder containing the .Rproj file.
* **data**: contains raw and processed data files in subfolders. Raw data should be made *read-only* and not changed in any way.
* **output**: outputs from R scripts such as figures or tables.
* **R**: all R scripts containing data processing or function definitions.
* **reports**: Quarto or RMarkdown files are saved here, as well as the resulting reports.
* **README**: a markdown file (can be generated from Quarto or RMarkdown) to explain the project.

See an example project folder structure

├── project-demo.Rproj  
├── data  
│ ├── processed  
│ │ └── data-clean.csv  
│ └── raw  
│ └── data-raw.xlsx  
├── output  
│ ├── fig-01.png  
│ ├── fig-02.png  
│ ├── tbl-01.png  
│ └── tbl-02.png  
├── R  
│ ├── 01\_import.R  
│ ├── 02\_tidy.R  
│ ├── 03\_transform.R  
│ ├── 04\_visualize.R  
│ └── custom-functions.R  
└── reports  
│ ├── soil-health-report.pdf  
│ └── soil-health-report.qmd  
│ └── images  
│ └── logo.png  
├── README.md  
└── README.qmd

R packages may contain these additional subfolders and files:

* **inst**: any arbitrary additional files to be included in package installation such as CITATION, fonts, and Quarto templates.
* **man**: all .Rd (“R documentation”) files for each function that are generated from [{roxygen2}](https://roxygen2.r-lib.org/).
* **vignettes**: long-form guides that go beyond function documentation and can be used as tutorials to demonstrate a workflow using the package to solve a particular problem.
* **tests**: all tests, usually using [{testthat}](https://testthat.r-lib.org/).
* **pkgdown** and **docs**: if using [{pkgdown}](https://pkgdown.r-lib.org/) to build a website for the package, you may have a pkgdown folder containing the favicon and any additional css and a docs folder containing the website source files.
* **DESCRIPTION**: file containing metadata about the package (authors, current version, dependencies).
* **LICENSE**: file describing the package usage agreement.
* **NAMESPACE**: file generated by [{roxygen2}](https://roxygen2.r-lib.org/) listing functions imported from other packages.
* **NEWS.md**: file documenting user-facing changes to a package.

Learn more about other R package components in [*R Packages (2e)*](https://r-pkgs.org/)(Wickham and Bryan 2023).

### Absolute vs relative paths

Note: directories are synonymous with folders.

❌Absolute paths start with the root directory and provide the full path to a specific file or folder like C:\\Users\\jryan\\Documents\\R\\projects\\project-demo\\data\\processed.[[2]](#footnote-197) You can run getwd() to find out where the current working directory is and setwd() to set a specific folder as your working directory. However, **please don’t do use setwd()** because this absolute file path is going to break your code if you reorganize your folders and is not going to work on any collaborators’ computers as their directory configuration will be different.

|  |
| --- |
| A cartoon of a cracked glass cube looking frustrated with casts on its arm and leg, with bandaids on it, containing "setwd", looks on at a metal riveted cube labeled "R Proj" holding a skateboard looking sympathetic, and a smaller cube with a helmet on labeled "here" doing a trick on a skateboard. Artwork by [@allison\_horst](https://twitter.com/allison_horst). |

✅Instead, always use relative paths in your scripts. Relative paths are *relative* to the working directory (i.e. the project’s home) like data/processed/data-clean.csv. When working in a RStudio project, the default working directory is always the **root** project directory (where the .Rproj file is).

For example, say 01-import.R contains the code read.csv("data/processed/data-clean.csv"). This will read the file from C:\\Users\\jryan\\Documents\\R\\projects\\project-demo\\data\\processed\\data-clean.csv". The magic of relative paths means that if Dani were have this project on her desktop and run this code, it would read the file from C:\\Users\\dgelardi\\Desktop\\project-demo\\data\\processed\\data-clean.csv". This is why relative paths are so important – they work no matter where the project folder is!

### {here} package

In combination with R projects, use the {here} package to build relative file paths. This is especially important in Quarto files because when the .qmd file renders, its default current working directory is wherever the .qmd file lives. If we are using the above example project structure and wanted to read in our clean data to our soil-health-report.qmd file, we would get an error running read.csv("data/processed/data-clean.csv") because it would be looking for a data subfolder in the reports folder. Instead, we can use the {here} package to build a relative path from our root with read.csv(here::here("data", "processed", "data-clean.csv")). {here} takes care of the backslashes or forward slashes so the relative path will work no matter the operating system.

|  |
| --- |
| A cartoon showing two paths side-by-side. On the left is a scary spooky forest, with spiderwebs and gnarled trees, with file paths written on the branches like "~/mmm/nope.csv" and "setwd("/haha/good/luck/"), with a scared looking cute fuzzy monster running out of it. On the right is a bright, colorful path with flowers, rainbow and sunshine, with signs saying "here!" and "it's all right here!" A monster facing away from us in a backpack and walking stick is looking toward the right path. Stylized text reads "here: find your path." Learn more about [here](https://here.r-lib.org/). Artwork by [@allison\_horst](https://twitter.com/allison_horst). |

## 10.2 Naming conventions

"There are only two hard things in Computer Science: cache invalidation and naming things."

— Phil Karlton

Based on this quote, Indrajeet Patil developed [slides](https://indrajeetpatil.github.io/second-hardest-cs-thing/) with a lot of detailed advice on naming things in computer science (completely language-agnostic).

R code specific naming conventions are listed below. They mostly follow the best practices from [Section 4.2](#sec-naming-best-practices) in the naming conventions chapter still apply, with one exception as seen in [Section 10.2.3.1](#sec-variables-in-a-dataset).

TODO: add Python naming conventions (they differ from R as static variables aka constants are supposed to be SCREAMING\_SNAKE\_CASE).

### Project folder and GitHub repository

The project folder and GitHub repository name should be the same. Be concise and descriptive. Use kebab-case.

**Example**: washi-dmp, soils.

### Files

Be concise and descriptive. Avoid using special characters. Use kebab-case with underscores to separate different metadata groups (date\_good-name).

**Examples**: 2024\_producer-report.qmd, tables.R, create-soils.R.

If files should be run in a particular order, prefix them with numbers. Left pad with zero if there may be more than 10 files.

**Example**:

01\_import.R  
02\_tidy.R  
03\_transform.R  
04\_visualize.R

### Variables, objects, and functions

Let’s first define these terms so we’re on the same page. In this style guide, **variables** are essentially column names, **objects** are all data structures in R (vectors, lists, data frames), and **functions** are self-contained modules of code that accomplish a specific task.

#### Variables in a dataset

This is where we differ from the [*tidyverse style guide*](https://style.tidyverse.org/). Hadley recommends using only lowercase letters, numbers, and \_ for all variables (column names and R objects). However, we are defining variables as only column names and objects as only data structures in R. For column names, we use lowerCamelCase for most variables, except for measurement names, we use \_ to separate the measurement name from the unit and . as a substitute for / (fraction bar).

**Example of general variable**: sampleId

**Example of soil health indicator variable**: 96-hr-min-c\_mg-c-per-kg-per-day = 96 hour mineralizable carbon (mg C/kg/day).

#### Objects and functions

Objects names should be nouns and function names should be verbs (Wickham 2022). Again, use lowercase letters, numbers, and underscores. Do not put a number as the first character of the name. Do not use hyphens. Do not use names of common functions or variables.

**Object examples**:

# Good  
primary\_color  
data\_2023  
  
# Bad  
  
# Less human readable, inconsistent with style guide  
primarycolor  
  
# Using a hyphen in an object name causes error  
data-2023 <- read.csv("2023\_data-clean.csv")  
Error in data - 2023 <- read.csv("2023\_data-clean.csv") : could not find function "-<-"  
   
# Starting an object name with a number also causes error  
2023\_data <- read.csv("2023\_data-clean.csv")  
Error: unexpected input in "2023\_"  
  
# Overwrites R shortcut for TRUE  
T <- FALSE  
  
# Overwrites R function  
c <- 10

**Function examples**:

# Good  
add\_row()  
assign\_quality\_codes()  
  
# Bad  
  
# Uses noun instead of verb  
row\_adder()   
  
# Inconsistent with style guide  
assignQualityCodes()  
  
# Overwrites common base R function  
mean()

## 10.3 R Scripts

### Header template

Including a header in every R script helps standardize the metadata elements provided at the beginning of your code and documents its purpose. Dr. Timothy S Farewell wrote a great [blog post](https://timfarewell.co.uk/my-r-script-header-template/) for creating a template for the header of every R script. The following template and instructions are adapted from his post (Farewell 2018).

1. **Script name**: meaningful and concise.
2. **Purpose**: brief description of what the script aims to accomplish.
3. **Author(s) and email**: it’s good to know where the script originated from if there are any questions, comments, or improvements.
4. **Date created**: this is automatically filled in from the template.
5. **Notes**: free-text space for any thoughts or todos.

## ---------------------------  
## Script name: check-crops.R  
##  
## Purpose: Cross reference sample requests, Field Maps forms, and management   
## surveys to get the correct crop planted at the time of sampling.  
##  
## Author: Jadey Ryan   
##  
## Email: jryan@agr.wa.gov  
##  
## Date created: 2024-01-02  
##  
## Notes:  
##   
## ---------------------------  
   
# Attach packages ==============================================================  
  
library(readxl)  
library(writexl)  
library(janitor)  
library(dplyr)  
library(tidyr)  
  
# Load data ====================================================================

Add this template to RStudio using [snippets](https://docs.posit.co/ide/user/ide/guide/productivity/snippets.html):

1. Modify the below code with your name and preferred packages.
2. In RStudio, go to Tools > Edit Code Snippets.
3. Scroll to the bottom of the R code snippets, and paste your modified code (the indent and tabs are important!).
4. Click Save and close the window.
5. Try it out by opening a new blank .R script, typing header, and then Shift + Tab.

snippet header  
 ## Header ======================================================================  
 ##  
 ## Script name:   
 ##  
 ## Purpose:   
 ##  
 ## Author: Jadey Ryan   
 ##  
 ## Email: jryan@agr.wa.gov  
 ##  
 ## Date created: `r paste(Sys.Date())`  
 ##  
 ## Notes:  
 ##   
   
 # Attach packages ==============================================================  
  
 library(readxl)  
 library(writexl)  
 library(janitor)  
 library(dplyr)  
 library(tidyr)  
   
 # Load data ====================================================================

### Section template

The above header template also uses section breaks (commented lines with = that break up the script into easily readable chunks). Section breaks are fantastic tools in RStudio because they allow you to easily show or hide blocks of code, see an outline of your script, and navigate through the source file. Read more about code folding and sections in this [Posit article](https://support.posit.co/hc/en-us/articles/200484568-Code-Folding-and-Sections-in-the-RStudio-IDE).

The snippet to create this section template that fills in the rest of the line with = was adapted from this [stack overflow answer](https://stackoverflow.com/questions/32633920/dynamic-rstudio-code-snippet).

snippet end  
 `r strrep("=", 84 - rstudioapi::primary\_selection(rstudioapi::getActiveDocumentContext())$range$start[2])`

To create a new section with the heading “Tidy data”, type in your script:

# Tidy data end<Shift+Tab> which will result in:

# Tidy data ====================================================================

|  |
| --- |
|  |

## 10.4 Code styling

Review the Syntax chapter of the [*tidyverse style guide*](https://style.tidyverse.org/syntax.html) for a lengthy section that covers spacing, function calls, long lines, semicolons, assignments, comments, etc. Also skim through [Chapter 4 Workflow: code style chapter of *R4DS*](https://r4ds.hadley.nz/workflow-style.html), which highlights the opinionated “most important parts of the tidyverse style guide”. Instead of rewriting all of these details and conventions into this style guide and making us all memorize the content, we should all just use the [{styler}](https://styler.r-lib.org/) (as advised in R4DS Chapter 4).

{styler} is a package and RStudio Addin that will format your code for you, so we can keep our coding style consistent across projects and better facilitate collaboration. We’ll deviate slightly from the tidyverse style and instead use [{grkstyle}](https://github.com/gadenbuie/grkstyle). {grkstyle} is an extension package for {styler} that Garrick Aden-Buie developed based on the tidyverse style guide. I prefer {grkstyle} over the tidyverse style that {styler} defaults to mainly because of how it handles line breaks in function calls.

The below example and installation instructions are pretty much copied directly from Garrick’s {grkstyle} README.

#### Examples

##### grkstyle

do\_something\_very\_complicated(  
 something = "that",  
 requires = many,  
 arguments = "some of which may be long"  
)

##### styler::tidyverse\_style

do\_something\_very\_complicated(  
 something = "that", requires = many,  
 arguments = "some of which may be long"  
)

#### Installation

Install {styler} and {grkstyle} with:

install.packages("styler")  
  
options(repos = c(  
 gadenbuie = "https://gadenbuie.r-universe.dev",  
 getOption("repos")  
))  
  
# Download and install grkstyle in R  
install.packages("grkstyle")

Set grkstyle as the default in {styler} functions and addins by running

# Set default code style for {styler} functions  
grkstyle::use\_grk\_style()

or adding the following to your ~/.Rprofile

options(styler.addins\_style\_transformer = "grkstyle::grk\_style\_transformer()")

To edit your .Rprofile, you can use usethis::edit\_r\_profile() to open the file in your RStudio.

#### Usage

Once {styler} and {grkstyle} are installed, you can apply the style to your .R, .qmd, and .Rmd files using the command palette, keyboard shortcut, or addins menu.

##### Command palette

Use RStudio’s **command palette** to quickly and easily access any RStudio command and see keyboard shortcuts. Open the command palette with Cmd/Ctrl + Shift + P, then type “styler” to see the shortcuts offered by {styler}.

##### Keyboard shortcuts

I often use Cmd/Ctrl + Shift + A to style the entire active file every time I finish a code block or section. To style just a selection, use Cmd/Ctrl + Alt + Shift + A.

##### Addins menu

You can also use the Addins menu in RStudio to style your files by clicking on a button to run the command.

|  |
| --- |
|  |

# References

Bryan, Jennifer. 2018. “Excuse Me, Do You Have a Moment to Talk about Version Control?” *The American Statistician* 72 (1): 20–27. <https://doi.org/10.1080/00031305.2017.1399928>.

Carlson, Bryan. 2021. “Data Management Plan for the R.J. Cook Agronomy Farm Long-Term Agroecological Research Site.”

European Commission. 2016. “H2020 Programme Guidelines on FAIR Data Management in Horizon 2020.” <https://ec.europa.eu/research/participants/data/ref/h2020/grants_manual/hi/oa_pilot/h2020-hi-oa-data-mgt_en.pdf>.

Farewell, Dr Timothy S. 2018. “My Easy r Script Header Template Tim Farewell.” <https://timfarewell.co.uk/my-r-script-header-template/>.

Harvard Medical School. 2023. “Data Management Plans.” <https://datamanagement.hms.harvard.edu/plan-design/data-management-plans>.

Lewis, Crystal. 2023. *Data Management in Large-Scale Education Research [in Preparation]*. <https://datamgmtinedresearch.com/>.

U.S. Fish & Wildlife Service. 2023. “Data Management Life Cycle.” <https://www.fws.gov/data/life-cycle>.

Whyte, Angus, and Graham Pryor. 2011. “Open Science in Practice: Researcher Perspectives and Participation.” *International Journal of Digital Curation* 6 (March): 199–213. <https://doi.org/10.2218/ijdc.v6i1.182>.

Wickham, Hadley. 2022. *The Tidyverse Style Guide*. <https://style.tidyverse.org/index.html>.

Wickham, Hadley, and Jennifer Bryan. 2023. *R Packages (2e)*. 2nd ed. <https://r-pkgs.org/>.

Wickham, Hadley, Mine Çetinkaya-Rundel, and Grolemund Garrett. 2023. *R for Data Science (2e)*. 2nd ed. <https://r4ds.hadley.nz/>.

Wilkinson, Mark D., Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, et al. 2016. “The FAIR Guiding Principles for Scientific Data Management and Stewardship.” *Scientific Data* 3 (1): 160018. <https://doi.org/10.1038/sdata.2016.18>.

1. Enough farm, producer and field info to distinguish unique farmers and fields and assign IDs to. This info doesn’t need to be personally identifiable information. [↑](#footnote-ref-152)
2. Note the two backslashes. Windows paths use backslashes, which mean something specific in R. to get a single backslash in the path, we need to type two backslashes (or use forward slashes). [↑](#footnote-ref-197)