pxmake — The Cheatsheet

Make, read, edit, validate and write PX-files in R

STATISTICS GREENLAND

Install & load

```
install.packages("pxmake")
library(pxmake)
```

Quick start

```
x <- px(input = NULL, data = df)
```

No metadata?

`px()` add minimal valid defaults. Then set keywords

• Minimal keywords are created when missing; set more via helpers (title, units, codes, etc.)

px() (constructor)

```
px(input = NULL, data = NULL, validate = TRUE)
```

• `input`: path/URL to PX/Excel, or `NULL`

• `data`: tidy data.frame (one column per variable + one value/measure column)

px_save()

Writes based on file extension: `.px`, `.xlsx`, `.R`.

```
px_save(x, "table.xlsx", save_data = TRUE, data_path = NULL)
```

• `.R`: reproducible script that reconstructs the px object.

• Use `data_path` to reference external data files (e.g. `.rds`/`.parquet`).

Read / write

• Read PX → `x <- px("file.px")`

• Read Excel → `x <- px("meta.xlsx")`

• Save PX → `px_save(x, "table.px")`

• Save Excel → `px_save(x, "table.xlsx")`

• Save R → `px_save(x, "table.R")`

Alternative input (pxweb saved query)

You can load a PX object directly from a saved query in a pxweb database (e.g. bank.stat.gl).

Append `.px` to the saved query URL:

```
x <- px("https://bank.stat.gl/sq/5058908a-cc8a-44aa-bee7-7e2827948459.px")
```

Note: the `.px` at the end is required, if the saved query is not already defined as PX.

Full PX-file

A saved query is typically just a subset of a PX-file.

Fetch the entire file by adjusting the API URL:

`https://bank.stat.gl/api/v1/en/Greenland/BE/BE01/BE0120/BEXSTA.px` →
`https://bank.stat.gl/Resources/PX/Databases/Greenland/BE/BE01/BE0120/BEXSTA.px`

Note: Server must serve `.px` with MIME `text/x-pcaxis` (e.g. IIS).

Metadata helpers

`px_title()`, `px_subject()`, `px_stub()`/`px_heading()`, `px_codes()`, `px_values()`, `px_units()`, `px_matrix()`, `px_contents()`, `px_language()`, `px_notes()`...

Data helpers

`px_add_totals()`, `px_elimination()`, `px_map()`, `px_domain()`, `px_pivot()`, `px_micro()`.

Validation

`px_validate(x)` — run checks. Many setters accept `validate=`.

Multilingual

```
x |> px_languages(c("en","da","kl")) |>
px_language("en")
```

Large datasets (Parquet)

df <- arrow::read_parquet("data.parquet")

Keep PX metadata in x and export with `px_save()` (use `data_path`).

From tidy microdata → tidy table data

Summarise micro rows to tidy table counts with `dplyr`:

```
library(dplyr)
df_tbl <- micro_df |>
count(across(everything()), name =
"value")
```

Then build the PX object:

```
x <- px(data = df_tbl) |>
px_stub("var1", "var2") |>
px_heading("time")
```

Reproducibility

`px_save(x, "table.R")` writes an R script that reconstructs *x*.

End-to-end

```
df <- tibble::tibble(sex=c("F","M"),
time=c("2024","2025"), value=c(42,43))
x <- px(data = df) |> px_title("Population
by sex and year") |> px_stub("sex") |>
px_heading("time")
px_validate(x)
px_save(x, "population.px")
px_save(x, "population.xlsx")
px_save(x, "population.R")
```

Tips

• Column names become variable names.

• Use `px_value_order()` for level order.

• For large tables, skip validation in setters.

Further reading (Community Book)

• Building PX-files from scratch • Handling time • Codes & labels • Multilingual • Validation • Large PX-files

See: `https://thranholm.quarto.pub/px-and-r`

Help

`?px` — constructor overview

`?px_save` — export & options

`?px_add_totals`, `?px_elimination`, ... — function help

`help(package = "pxmake")` — package index

Docs

CRAN: `?px`, `?px_save` • GitHub: StatisticsGreenland/pxmake

CRAN (pxmake)

GitHub repo

Community Book

Version 1.0 — Oct 2025