# FiberQ 1.4.0 — Validating a design, and fixing the lengths

A user guide to the validation engine introduced in **FiberQ 1.4.0**. For rule-by-rule detail, see the validation rules reference. For the field-by-field data model, see the schema reference.

FiberQ 1.4.0 lets you **audit a fibre design before it leaves your desk**. One command runs fourteen checks over the whole project — topology, referential integrity, feature identity, required attributes, value domains, length coherence, coordinate systems and geometry health — and lists what it finds in a dockable panel you can filter, click through, and export as a report.

It also fixes how FiberQ measures. Routes and cables used to store their length in **map units**; they now store **true ground metres**, and a repair command rewrites the old values.

---

## 1. Running a validation

**Plugins → FiberQ → Validate project**, or the √-on-a-page button on the FiberQ toolbar.

Nothing is modified. Validation only reads your project, so it is safe to run at any point — mid-design, before a review, or as the last thing you do before handover.

The **FiberQ validation** panel opens with the results, and the QGIS message bar gives you the headline:

```
FiberQ: 4 error(s), 9 warning(s), 1 info
```

A clean project says so, and says what it looked at:

```
FiberQ: Validation found no issues (14 rules, 15 layers).
```

That second half matters. "No issues" from fourteen rules across fifteen layers and "no issues" because there was nothing to check read identically otherwise — and only one of them is good news.

> **If a rule crashes on your data**, the run continues and the panel says how many rules gave up. Which ones, and why, are in **View → Panels → Log Messages**, on the FiberQ tab, and in the reports: under "Rules that failed to run" in the HTML, and as `run.rule_errors` in the JSON. A run where every rule failed is never reported as a clean project. Please open an issue if you see one.

## 2. Reading the panel

Each row is one finding: **Severity · Rule · Layer · Feature · Message**.

| Severity | Meaning | What it should stop |
|---|---|---|
| **Error** | The project is internally inconsistent — something references what is not there, or a geometry is unusable. | Handover, export, anything downstream. |
| **Warning** | Probably wrong, or incomplete. A human needs to look. | A final sign-off, but not your work in progress. |
| **Info** | Worth knowing. Often a near-miss that may be deliberate. | Nothing on its own. |

Errors sort to the top. A design still being drawn is expected to produce warnings; **the number that matters at handover is the error count**.

What you can do from the panel:

- **Double-click any row** (or select it and press Enter — Cmd+O on macOS) to centre the map on that finding. Layer-level findings — a missing CRS, a layer with no identity field — have no coordinate to jump to, so nothing moves.
- **Filter** by severity, layer or rule. The Layer and Rule combos only offer values that are actually present; Severity always lists all three.
- **Sort** any column. Severity sorts by rank rather than by its translated label, and Feature sorts numerically, so 9 comes before 10.
- **Hover a message** to see the fix hint where a rule has one.
- **Re-run** after you fix something, without leaving the panel.
- **Export report…** to write the run to disk (section 5).

Rule ids (`A1`, `D3`, …) are **stable and untranslated** everywhere — panel, exports, this document — so you can filter on them, quote them in a ticket, and have them mean the same thing in every language.

## 3. The fourteen rules

Full detail, including causes and what to do about each finding, is in the validation rules reference. At a glance:

| | Rule | Severity | Checks |
|---|---|---|---|
| **A** | A1 | Warning | Cable endpoints reach an element or another cable |
| Topology | A2 | Info | Endpoints that miss by just outside tolerance |
| | A3 | Warning | Elements are attached to the network at all |
| **B** | B1 | Error | Optical slack references an existing **cable** |
| Integrity | B2 | Error | Fibre break references an existing **cable** |
| | B3 | Warning | Those references are spatially coherent, not stale |
| | B4 | Error | `fiberq_uuid` present and unique on every feature |
| **C** | C1 | Warning | Required attributes are filled in |
| Complete-ness | C2 | Warning | The project contains FiberQ layers at all |
| **D** | D1 | Warning | Values come from the allowed domain |
| Attributes | D2 | Warning | Numbers are physically plausible |
| | D3 | Warning | Stored lengths agree with the drawn geometry |
| **E** | E1 | Warning | The project has a CRS, and layers share it |
| Geometry | E2 | Error / Warn-ing | Geometry present and non-empty (Error); lines and polygons well formed (Warning) |

The A-rules use a snap tolerance — "close enough to be the same point" — of **5 map units**, a setting the project keeps; the rules reference's Configuring the rules shows how to change it, from the Python console, since this release has no dialog for it. In a national grid those are true metres. In Web Mercator they are not, which is one of several reasons to work in your national grid where you can; E1 will tell you when the CRS is actively breaking the other rules.

## 4. Lengths: what changed, and how to repair a project

**This is the fix most likely to affect a project you have already delivered.**

Before 1.4.0, routes and cables stored the length QGIS measures **in map units**. In a projected national grid that is correct. In **Web Mercator (EPSG:3857)** — what a design traced over a web basemap really uses — distances are inflated by `1 / cos(lat-itude)`:

| Where | Latitude | Stored length was too long by |
|---|---|---|
| Serbia | ≈ 45° | **≈ 41 %** |
| Netherlands | ≈ 52° | **≈ 62 %** |
| Southern Finland | ≈ 60° | **≈ 100 %** |

Pipes were already measured correctly. So in the same project, a trench and the duct inside it could disagree by that margin — and a bill of materials built on the cable figure over-ordered by it.

From 1.4.0, every route, cable and pipe length is measured **on the project ellipsoid**, exactly the way the QGIS measure tool does, and new features are correct as drawn. (Two tools lagged behind when the project had no ellipsoid set, both fixed in 1.6.0: **Cut infrastructure** wrote map units for the two parts it creates, and the **Service Area** tools stored `perim_m` and `area_m2` in map units. A Service Area made before 1.6.0 keeps the figures it was stored with: neither D3 nor **Recalculate lengths** looks at them, so draw it again to replace them.) From 1.6.0 lengths also stay correct as you edit: move a vertex of a saved line and the stored length follows in the same undo step, and other edits are re-measured when you save. Rule D3 lists the edits FiberQ cannot follow.

## Repairing existing features

**Plugins → FiberQ → Recalculate lengths…**

It is menu-only on purpose: this one rewrites your attributes, so it should be a deliberate trip to the menu rather than a click away on the toolbar.

It shows you **exactly what will change before writing anything** — how many features, broken down per layer, and the single largest change. On the demo project (section 6) it shows:

```
Recalculate stored lengths?

  1 feature(s) will have their stored length rewritten from the
  drawn geometry.

    Route: 1

  Largest change: duzina 210.00 -> 144.61 on Route
```

Then, and only if you say yes, it rewrites `duzina`, `duzina_m`, `duzina_km` and `total_len_m` from the geometry.

- **Slack values are read but never changed.** Slack is a deliberate surplus of cable, not a measurement of the line on screen, so recomputing it from geometry would be wrong.
- **Layers open for editing are skipped** and named, even with nothing unsaved, so nothing collides with an edit session you have open. Save or discard there, leave edit mode, and run it again.
- If the panel is already open, validation re-runs afterwards so you can see the D3 findings disappear.

> **Compare it yourself.** Take any route, measure it with the QGIS measure tool (with ellipsoidal measurement on, which is the default), and compare with the attribute table. Before recalculation the attribute is the larger number; after, the two agree.

### Which number is right?

The ellipsoidal one — the one you get after recalculating. It is the distance along the ground, which is what you order cable by and what a surveyor would find on site. The old figure was the length of the line as the map projection draws it, which at 45° north is a 41 % exaggeration nobody asked for.

---

## 5. Exporting a report

With results in the panel, **Export report…** writes the run in whichever format the file-name asks for, so a hand-typed name and the selected filter can never disagree:

| Format | Use it for |
|---|---|
| **HTML** | Handover and audit files. Self-contained — no internet needed, prints cleanly, opens in any browser years from now. |
| **JSON** | Tooling. Stable keys, machine-readable, and the basis of the WP3 interchange format. |
| **CSV** | Spreadsheets and issue trackers. One row per issue. |

The report carries the project name, CRS, plugin version and a timestamp, so a file in an audit folder still says what it was a run of.

Rule ids, severities and categories stay in English in every format so downstream tools can key on them; only the prose is translated. The JSON carries a `format_version` — check it before parsing if you are building on top of it.

---

## 6. Try it on the demo project

The repository ships a small, deliberately imperfect design in `docs/samples/` — twelve of the fourteen rules fire, each from exactly one planted fault, alongside the report it produces.

1. Open `docs/samples/demo_project.qgz`.
2. **Plugins → FiberQ → Validate project**.
3. You should see **4 errors, 9 warnings and 1 info**, matching the committed `demo-validation.html`.
4. Double-click a row to jump to it.
5. **Plugins → FiberQ → Recalculate lengths…**, then validate again: the D3 finding goes away and the warning count drops to 8.

The data is synthetic — invented coordinates in an empty stretch of the Adriatic, at a European latitude so the Web Mercator scale factor is realistic. No real network, no customer data. See `docs/samples/README.md` for what each planted fault is and why.

## 7. Also in 1.4.0

- **Placing an extension or a fibre break no longer creates an empty "Poles" layer** as a side effect.
- **The fibre-break tool now targets cable layers only.** It used to accept a click near any line, so a break could be recorded against a route or a duct and stored as though it were on a cable. B1 and B2 report the bad references already sitting in your projects.
- **Shipped layer styles no longer name a Windows-only font.** They asked for "MS Shell Dlg 2", so every Linux and macOS user got substituted label fonts and a warning per layer on load. They now use the portable "Sans Serif".
- **Opening a QGIS 4 project in QGIS 3 no longer re-runs the schema migration.** QGIS 4 stores project properties in a form QGIS 3 cannot read, so the schema marker now falls back to the copy in the GeoPackage. The same QGIS limitation drops the project CRS on the way down, which E1 now reports. From 1.6.0 FiberQ also warns on open that QGIS 3 cannot read the project's other FiberQ settings and that saving it there removes them permanently: reopen such a project in QGIS 4 rather than resetting the CRS and saving it in QGIS 3.
- **The plugin is now translatable.** The English source catalogue and the tooling are in place; see `docs/TRANSLATING.md`, which assumes no programming, Git or terminal experience. Partial translations are welcome and ship as-is.

## 8. Quality assurance and compatibility

- **Automated tests on both Qt generations.** The pytest suite — including the validation engine, every rule, the panel, the report writers and the sample report itself — runs green against **QGIS 3 / Qt5** and **QGIS 4 / Qt6** in CI.
- **The published sample cannot go stale.** `tests/test_sample_report.py` fails the build if a fresh run over the demo project no longer matches the committed report, so a rule that starts reporting something new is caught rather than quietly changing what is published.
- **Clean repository scan.** Zero findings on the plugins.qgis.org Security & Quality scan — Bandit at all severities, secrets detection, flake8, file permissions and suspicious files — and the **Qt6 compatibility check** passes.
- **Manual acceptance testing across QGIS versions.** 1.4.0 was verified end to end on **QGIS 3.40 (Qt5)** and **QGIS 4.2 (Qt6)**, across four projects: a from-scratch design, a project created in QGIS 4.2 and opened in 3.40, a pre-1.0 project migrated from an earlier FiberQ, and the demo project — in both a projected national grid (EPSG:3909) and Web Mercator, with drawn geometry checked against the QGIS measure tool in each. Nine defects found that way are fixed in this release, each with a regression test.
- **Scale.** Validation of a 25,500-feature project completes in about two seconds, and the panel renders a 20,000-issue result in about one.
- **Backward compatibility.** FiberQ 1.4.0 targets QGIS 3.22 LTR through QGIS 4 / Qt6, and opens projects from earlier FiberQ versions via the automatic migration introduced in 1.3.0 (see the project versioning guide).

---

FiberQ is free software under the GPL-3.0-or-later licence. Source, issues and releases: https://github.com/vukovicvl/fiberq.

The validation engine and its reporting were developed with support from the NLnet NGI Zero Commons Fund. The length-measurement fix, the stray-layer fix, the i18n groundwork and the support links were not funded by it.