Development Guidelines

Last updated by | Nils Johannsen | Jul 5, 2024 at 8:41 AM GMT+2

Development Guidelines

When creating examples, the aim should be that they are written in such a way that they can be officially made available to customers as examples or can be included in products.

C++

When developing **Code in C++** try to be confident with C++ Core Guidlines
https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md ↗

An optional convention is:

- TwinCAT C++ Naming Guideline

PLC

For developing **PLC code** try to confident with the PLC Coding Guideline
https://infosys.beckhoff.com/english.php?
content=../content/1033/tc3_plc_intro/12049233675.html&id=6398798947359024199 ↗.

Also have a look on:

- InfoSys-Samples_HowTo.pdf

An alternative convention is:

- https://docs.tcopengroup.org/articles/Conventions/Conventions.html ↗

HMI

To develop the HMI use our recommendations: https://azdevops01.beckhoff.com/ApplicationCommon/tf2x-hmi/_wiki/wikis/tf2x-hmi.wiki/41/Guidelines

It is a collection of guides for all areas of our HMI: CSS, HTML, Javascript etc.

Wording

Follow the **Corporate Wording**, especially for company and product names:

- https://intranet.beckhoff.com/myworkplace/tools_transl_proofr_terminology/Pages/Proofreading.aspx ↗

Concepts

- DRY: Don't repeat yourself ↗
- KISS: Keep it simple, stupid ↗
- SOC: Separation of concerns ↗
- SSOT: Single source of thruth ↗

- Code should be easy to change
- when making changes to other people's code, adapt the code's style