FeatStd 3.7.0

- FeatStd Stream interface
- FeatStd TCP/IP reimplementation
- State machine
- Generic Handle
- Thread Priority
- Thread Nice Value
- Updated AsyncRequestDispatcherWorkerThread interface

FeatStd Stream interface

An interface for streams has been added. This shall be an abstraction layer for read and write actions. A stream implementation has been added for TCP/IP: See also FeatStd TCP/IP reimplementation Classes for buffered streaming and classes for primitive datatype reading and writing from and to streams have been added.

FeatStd TCP/IP reimplementation

The concept of TCP/IP has been reworked. There are two new classes available: TcpServer and TcpClient. The server can open a port to listen to. A connection to such a server will return a stream which is of type TcpClient. The client can then be used for communication. This concept allows multiple connections at the same time with the convenient stream interface. The old TcpIp interface is still available and extended by an own socket concept. Using the old functions work completely equal to the old concept. Therefore, only one connection to the server is allowed. After the connection is lost, it cannot be reused. A shutdown is required.

> The old TcpIp will be deprecated in future versione with an upgrade of Monitor to the new concept.

State machine

The state machine has been extended by a monitor feature. This feature can be enabled by the CMake flag: FEATSTD_STATEMACHINE_MONITOR_ENABLED. This feature adds events on different locations to track changes within the state machine. Depending on the type of change a MonitorEvent or a TransitionMonitorEvent will be sent. To receive this events, an event listener has to be attached to Candera::Internal::StateMachineBehaviorData::State::GetMonitorEventSource().

Generic Handle

Relevant functions were inlined, and a reverse lookup was added.

Thread Priority

Class Thread was enhanced with the posibility to set/get the thread priority. This feature is implemented for Posix, Windows and Integrity platform. For all other platforms setting the priority has no effect. See FeatStd::Thread class for more details.

Thread Nice Value

Class Thread was enhanced with the posibility to set/get the thread nice value. This feature is supported only for Linux Posix platforms. On all other platforms it has no effect. See FeatStd::Thread::SetNiceLevel for more details.

Updated AsyncRequestDispatcherWorkerThread interface

Class AsyncRequestDispatcherWorkerThread was updated with the option to set the worker thread priority on thread start.