



# Data-Types

④ With FESA3 we distinguish between default types and custom types (user defined types)

④ default

- scalar
- array
- array2D

④ custom

- custom-type-scalar
- custom-type-array
- custom-type-array2D

# Scalar data types



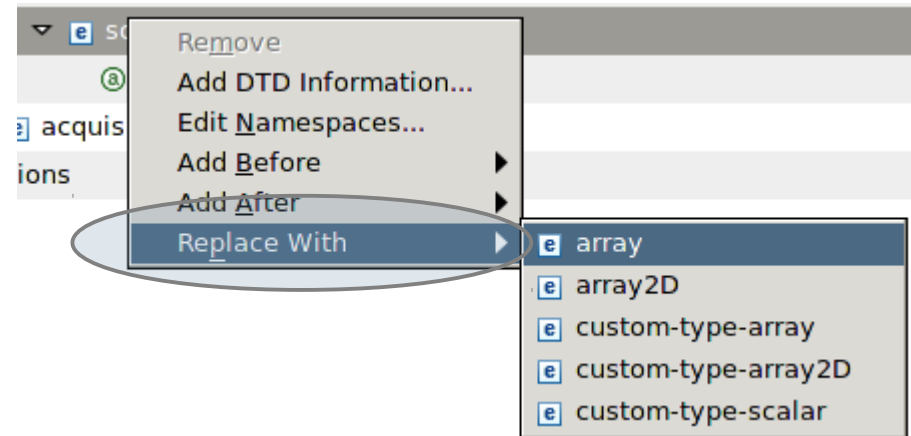
```
bool
int8_t
int16_t
int32_t
int64_t
uint8_t
uint16_t
uint32_t
uint64_t
float
double
```

① + custom types

① type “char” currently only for arrays

# Array(2D) data types

- ① Use the context menu (rightclick on item)
- ① arrays can have different dimensions
  - ① one-dimensional (i.e. “string”)
  - ① two-dimensional (i.e. “string-array”)
  - ① a dimension can be constant or device-specific (variable)
- ① arrays can contain custom types





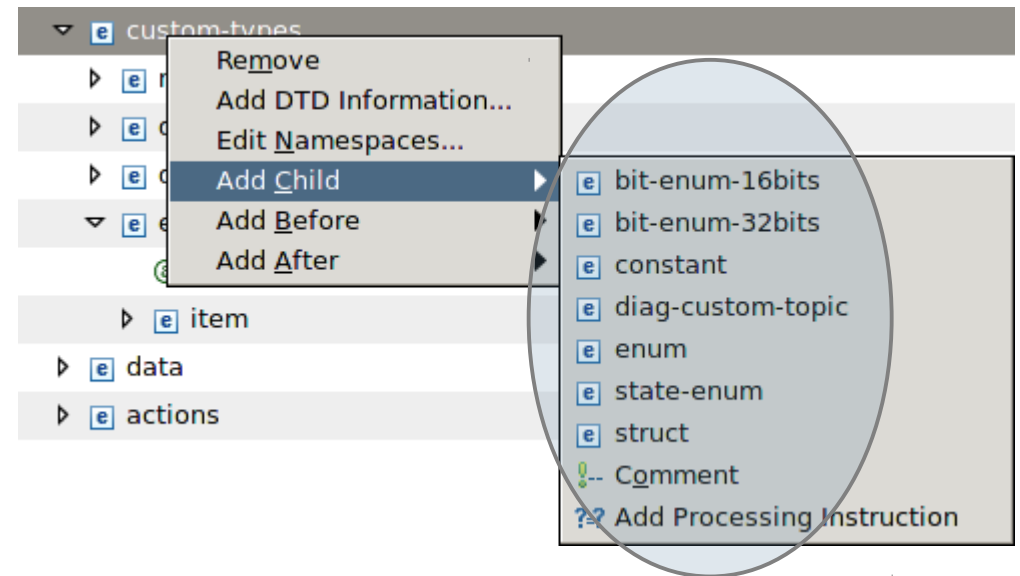
# Custom data types

- ① Custom-types can be used in the design and in the C++ code
- ① Like in plain code, they make things more readable, prevent duplication and stop the usage of “magic-numbers”
- ① Use them whenever possible!



# Custom data types

## **enum**

-  name to use in design and C++ code
-  items
  - Unique value
  - symbol to use in code
  - access (RW, RO, WO)



## **state-enum**

-  More or less the same than enum
-  Some values predefined

## **constant**

## **struct**

## **bit-enum 16bits**

## **bit-enum 32bits**

# Custom data types

- enum
- state-enum
- constant
- struct**
  - struct-item type can be any fesa-data-type
  - currently not possible to send structs directly via the middleware
- bit-enum 16bits
- bit-enum 32bits

▼ <b>e</b> struct	(description*,
<b>a</b> name	GSI_ERROR
▼ <b>e</b> struct-item	(description*,
<b>a</b> name	error_string
▷ <b>e</b> array	((dim   custor
▼ <b>e</b> struct-item	(description*,
<b>a</b> name	error_code
▷ <b>e</b> scalar	
▷ <b>e</b> struct-item	(description*,
▷ <b>e</b> struct-item	(description*,

# Custom data types

- enum
- state-enum
- constant
- struct
- bit-enum 16bits**

▼ <b>e</b> bit-enum-32bits	(description*, b0?, b1?)
Ⓐ name	DETAILED_STATUS
▼ <b>e</b> b0	
Ⓐ name	CoolingWaterIsEmpty
▼ <b>e</b> b1	
Ⓐ name	CoolingWaterToHot
▼ <b>e</b> b2	
Ⓐ name	FanIsOn

- Can be used as bitmask to address single-bit in the C++ code
- only for scalar-types
- often used for hardware-components
- will get further updates in coming fesa-versions
- bit-enum 32bits**
- same as bit-enum 16bits, just more bits

# Mission



- ① **Create a class which periodically measures an array of values(use random values) in the RT-Action.**
- ① **The measured array has to be offered by an acquisition-property.**
- ① **Use a custom-type for the array-size.**
- ① **Use custom-types to define the minimum and maximum rand-values.**
- ① **The client should be able to switch the measurement ON and OFF. Use an enum-type for this.**

**(Manual Notification!)**



Thank you for your listening !!