Innovative Teaching Approaches in development of Software Designed Instrumentation and its application in real-time systems

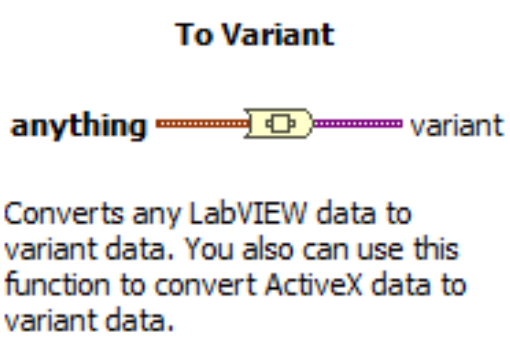
The Advanced Applications of LabVIEW

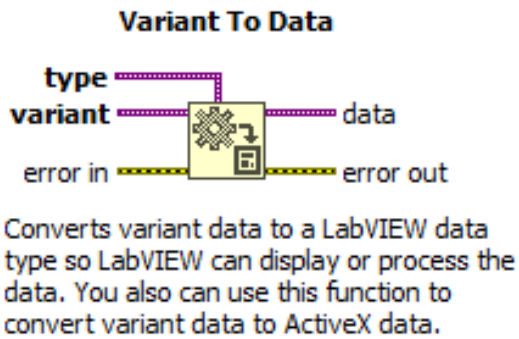
Lecture 5: Queued Message Handler

Co-funded by the
Erasmus+ Programme
of the European Union

Variants

- Variant is a generic data type, which allows you to store and transfer any data types in LabVIEW.
- Variants are useful when you would like to:
  - create subVI, which can operate using different data types depending on your needs,
  - create application based on producer/consumer design pattern and you need a flexibility of data types you would like to transfer between loops.



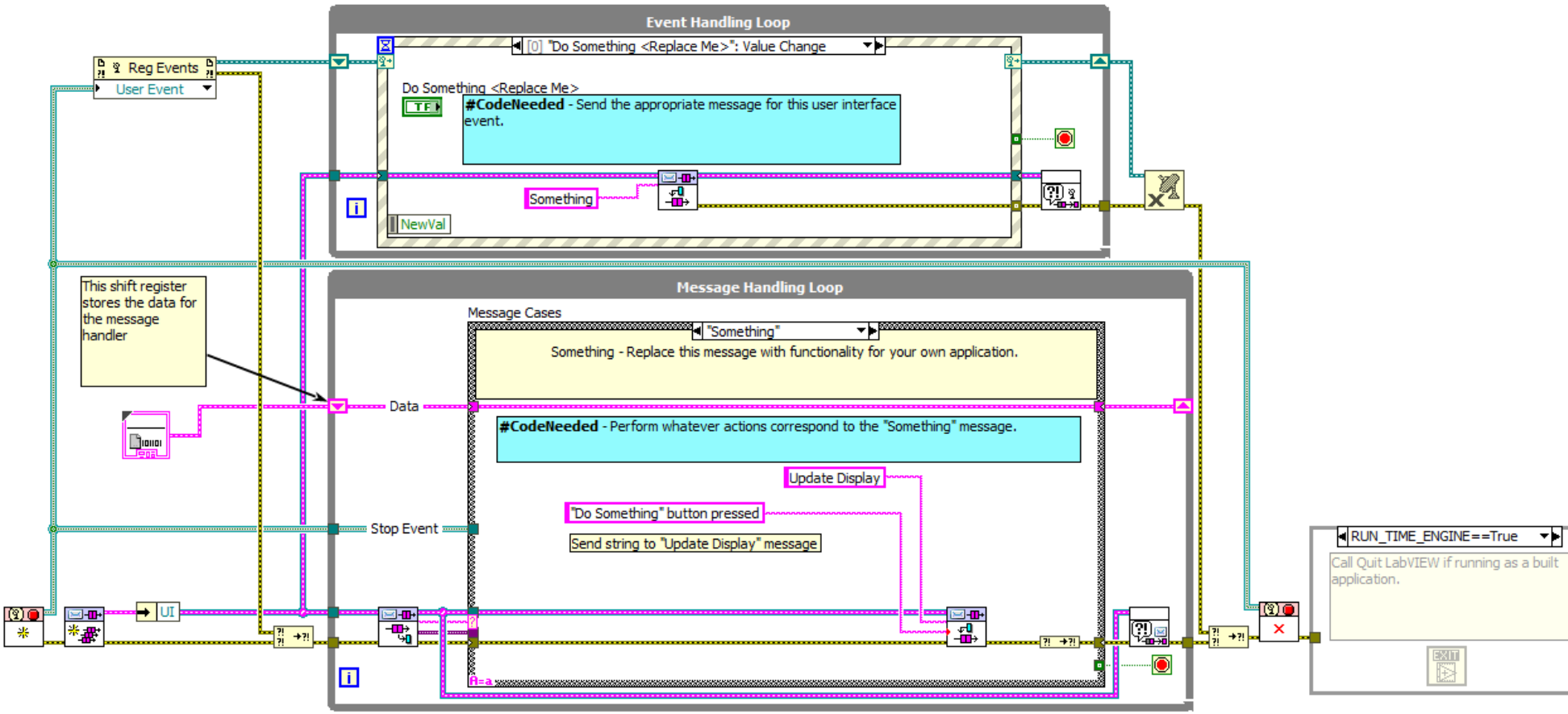


Characteristic of QMH

- The Queued Message Handler (QMH) design pattern is a modification of producer/consumer design pattern.
- In this case, the producer loop sends messages to the consumer loop. Consumer loop analyses messages and also can add new messages to the queue (sending to itself).
- In practice, it means that new elements are added to queue in producer and consumer loops.
- The QMH transfer a message and message data (optional) between loops
- THE QMH is created by choosing: ***File*** $\rightarrow$ ***Create project*** $\rightarrow$ ***Queued Message Handler***.

Component	Function	Data types
Message	to send an order to slave/consumer/message loop	String or enum
Message data	to send necessary data to slave/consumer/message loop	string or variant

Template


Event Handling Loop
[0] "Do Something <Replace Me>": Value Change
Reg Events
User Event
Do Something <Replace Me>
TF
#CodeNeeded - Send the appropriate message for this user interface event.
Something
NewVal
This shift register stores the data for the message handler
Message Handling Loop
Message Cases
"Something"
Something - Replace this message with functionality for your own application.
Data
#CodeNeeded - Perform whatever actions correspond to the "Something" message.
Update Display
"Do Something" button pressed
Stop Event
Send string to "Update Display" message
UI
RUN_TIME_ENGINE==True
Call Quit LabVIEW if running as a built application.
EXIT

Thank you for attention!

Lecture was prepared based on materials from: "LabVIEW Core 3 Course Manual".

This project has been funded with support from the European Commission. This communication reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.