FABRIC ENGINE®

It's Time For Real–Time

Splice API

Splice API for C/C++

- Abstracts the Fabric Core
- Simplifies the use of KL within C/C++
- Base layer for all DCC integrations
- Allows integration into custom graphs

Scons

- Cross platform build system
- Of course you can use others too (like CMake)
- Similar workflow in Visual Studio

Static vs. Shared

- Splice API is provided as both static + dynamic
- For static linking you can choose to use a shared or also static *FabricCore*.
- You need to speficy the library kind used, both for the Core and for Splice, using the *FEC_STATIC*, *FEC_SHARED*, *FECS_STATIC* and *FECS_SHARED* defines.

Hello World

File reference: 02_spliceapi/01_helloWorld/main.cpp

- Create a *FabricSplice::DGGraph*
- Create a single *FabricCore::DGNode*
- Create a single *KL Operator* to report "*Hello World*"

Logging

File reference: 02_spliceapi/02_logging/main.cpp

- Custom log callbacks
- Custom error callbacks
- Custom KL report callbacks
- Custom KL status callbacks
- Custom compiler error callbacks

KL Types and Extensions

File reference: 02_spliceapi/03_kltypes/main.cpp

- The FABRIC_RT_PATH is used for KL types
- The FABRIC_EXTS_PATH is used for extensions
- FabricSplice::addRTFolder
- FabricSplice::addExtFolder
- Callbacks for runtime provided types

Ports – Data IO

File reference: 02_spliceapi/04_ports/main.cpp

- Used to push data in and pull data out
- Map to a member on the DGNode
- Provide information about the data

RTVals

File reference: 02_spliceapi/05_rtvals/main.cpp

- RTVals provide a way to access KL types in C/C++
- You can iterate the data structure
- You can invoke methods on them

Subgraphs

File reference: 02_spliceapi/06_subgraphs/main.cpp

- A single DGGraph can contain several DGNodes
- DGNodes can depend on other DGNodes
- KL Operators can access ports on dependencies
- Independent branches perform in parallel (branch based multi threading)

Persistence

File reference: 02_spliceapi/07_persistence/main.cpp

- A DGGraph can be saved to a Splice file, or to a Variant dictionary.
- These files represent portable tools.
- DGPorts can store arbitrary options, which are understood as hints for each integration.
- An empty DGGraph can load a Splice file, which reconstructs all elements.