13 WorkflowDescriptors

Dependent workflows

What	Required	Description	schema location
WorkflowDescriptors.xml	yes	Definition of workflows	https://fewsdocs.deltares.nl/schemas/version1.0/workflowDescriptors.xsd

Each root workflow file configured in DELFT-FEWS must be registered in the WorkflowDescriptors configuration. This is required to identify the root workflows to DELFT-FEWS. The configuration also sets some properties of the workflow. Do not add workflow descriptors for workflow files that are only used as a sub workflow file.

The default name of the WorkflowDescriptors is: WorkflowDescriptors.xml. This file is located in the RegionConfigFiles folder.

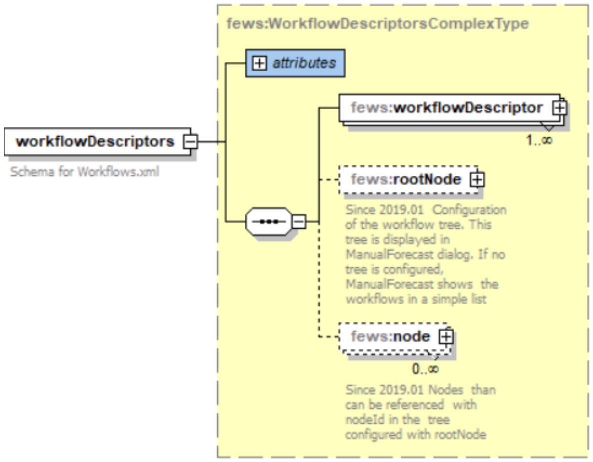
fews:WorkflowDescriptorsComplexType
attributes
workflowDescriptors
Schema for Workflows.xml
fews:workflowDescriptor
1..∞
fews:rootNode
Since 2019.01 Configuration of the workflow tree. This tree is displayed in ManualForecast dialog. If no tree is configured, ManualForecast shows the workflows in a simple list
fews:node
0..∞
Since 2019.01 Nodes than can be referenced with nodeId in the tree configured with rootNode

Figure: Elements of the workflowDescriptor configuration.

workflowDescriptor

Root element of the WorkflowDescriptor. A new element is required for each workflow. Multiple instances may be defined.

Attributes:

- **Id**: Id of the workflow. This Id must be unique. Normally a string is used that gives some understanding of the role of the module (*e.g. ImportExternal*).
- **name**: Optional name for the module. Used for reference purposes only.
- **visible**: Boolean toggle to indicate if workflow is visible for selection in the manual forecast display. Default is true.
- **Forecast**: Boolean flag to indicated if workflow is identified as a forecast. This should be the case for workflows with simulated time series as a results. Only Forecast workflows can be approved and made current. Import workflows of external data are not forecasts. Default is true.
- **allowApprove**: Boolean flag to indicate if workflow may be approved a-priori through manual forecast display (stand-alone only). Default is true.
- **allowSelection**: Boolean flag to indicate if workflow can be executed for specific selected locations. Default is false
- **autoApprove**: Boolean flag to indicate workflow should automatically be approved *a-priori* (stand-alone only). Default is false.
- **autoSetSystemTime**: Boolean flag to indicate workflow should automatically adjust the system time. When the workflow is completed and is fully or partly successful, the system time wil be set to the start time of the period written by this workflow.
  If the start time is not a valid time in accordance with the cardinal timestep, the next valid time wil be used.
  Default flag value is false. Applicable only on stand-alone.
- **maxEnsembleParts**: Positive integer. Determines in how many parts (i.e. forecasting shell instances available) at most the workflow can be split up to run in parallel for ensembles. Ensembles can/will only be split up if this element is specified and the number specified is larger than 1.
- **timeOut**: optional timeout that a workflow is allowed to run before the Forecasting Shell terminates it. Defaults to 4 hours.
- **showWarningInLogCompletion**: Boolean to report whether warning messages were logged during the execution of the workflow, default false.
- **waterCoachDelay**: Prevents simulated forecasts becoming available too early in water coach mode. The forecasts are finished in real life at time zero plus delay. The delay includes the run time and the time0 shift. When a forecast is scheduled at 7:00 am with a time0 of 0:00 am and the runs takes approximately an hour you should specify an delay of 8 hours. For external forecasts the delay is configured in the moduleInstanceDescriptors.xml
- **runExpirytime**: Used for runs of simulated timeseries to override expiry time from global properties DEFAULT_EXPIRY_DAYS. When timeseriesets expirytime is also configured, the minimum of runExpiryTime and the timeSeriesSets expirytime is used. Obsolete for import data since 2017.01.
- **cardinalTimeStep**: this settings allows for specification of the timesteps the workflow can be ran (eg specific hours per day, or every day at 00:00). This can be useful for imports or model runs that should take place on certain moments. In the Manual Forecast Display the effect can be seen when changing the T0. The cardinaltimestep is also respected by the MasterController: if the workflow is scheduled at a different moment than the cardinalTimeStep, the T0 will be adjusted (comparable to a T0 timeshift).
- **workflowFileName**: Defines the root workflow config file that should be started. By default the config file name equals the workflow id. Multiple descriptors can share the same config file by using moduleConfigProperties. The config file is used as template and the $key$ are replaced by the property values, also in all references child workflow files and module config files

- **properties**: These properties can be referenced in workflow and module config files with key, similar to global properties. key is resolved by first searching in the properties defined in the workflow files defined and than in the workflow descriptor of the running task and after that in the moduleConfigFileProperties defined in the the region config and at last in the global properties in the region home

```
<workflowDescriptor id="Run_Model_Catchment_A">
        <workflowFileName>Run_Model_Template</workflowFileName>
        <properties>
                <string key="CATCHMENT" value="A"/>
        </properties>
</workflowDescriptor>
```

Node (since 2019.01)

The workflows can be organised in a folder structure in the Manual Forecast Display with the node element. The workflow desctiptor ID's need to be added to the nodes. If this functionality is included in the workflowDescriptors, workflows that are not included in the node tree will not be visible anymore in the Manual Forecast Display.

```
<rootNode>
                <node name="Import">
                        <workflowId>Import_Hydro_Observations</workflowId>
                        <workflowId>Import_Forecasts</workflowId>
                        <workflowId>Import_EarthObservation</workflowId>
                        <workflowId>Import_Earth2Observe</workflowId>
                        <workflowId>Import_HYCOM_All</workflowId>
                        <workflowId>Import_WaveWatch3</workflowId>
                </node>
                <node name="Process">
                        <workflowId>Create_GFS_Ensembles</workflowId>
                        <workflowId>Copy_GFS_Grid</workflowId>
                        <workflowId>Modify_GFS_Grid</workflowId>
                </node>
                <node name="Rest">
                        <workflowId>Process_Operational_Grids</workflowId>
                        <workflowId>WFlow_historic</workflowId>
                        <workflowId>WFlow_forecast</workflowId>
                        <workflowId>WFlow_E2O_historic</workflowId>
                        <workflowId>Fractions_forecast</workflowId>
                </node>
        </rootNode>
```

ApprovalEventCode (since 2020.01)

ApprovalEventCode is user configured event code that is logged upon approving a forecast, instead of default event code “DataStore.NewCurrentRun” . ApprovalEventCode make it possible to log different approval event codes per workflow, so that different actions can be triggered by Master-Controller after approving particular forecast(s) . ApprovalEventCode can be any string that meets requirements of the event code pattern .

Use ‘ approvalEventCode’ to take more control over the triggering of the actions by Master_controller after the forecast approval .
Both workflows below do exactly the same forecast, but log different approval event code. Only the code FluvialForecast.Approved is used in Event Action Mapping (more info Event and Actions)
Master-Controller triggers any follow up action only when FluvialForecast becomes approved. FluvialForecastIntermediate can be run without any follow up actions.

approvalEventCode

```
<workflowDescriptor id="FluvialForecastIntermediate" forecast="true" visible="true" autoApprove="true">
             <approvalEventCode>FluvialForecastIntermediate.Approved</approvalEventCode>
</workflowDescriptor>
<workflowDescriptor id="FluvialForecast" forecast="true" visible="true" autoApprove="true">
             <approvalEventCode>FluvialForecast.Approved</approvalEventCode>
</workflowDescriptor>
```