

FEWS PI SOAP Web Service - EOL 2021.02



The Delft-FEWS PI SOAP Web Services has been removed since Delft-FEWS 2022.01. See: [Delft-FEWS End of Life Modules and Displays#FEWSEndofLifeModulesandDisplays-FEWSWebServicesAPIsthatwillberemoved](#)

[DAC installation guide \(final\).doc](#) (DAC installation is obsolete since FEWS 2017.02)

- [Introduction](#)
- [PI SOAP Web Service API](#)
 - [Getter methods](#)
 - [getTimeZoneld](#)
 - [getFilters](#)
 - [getLocations](#)
 - [getLocationsAsStream](#)
 - [getParameters](#)
 - [getParametersAsStream](#)
 - [getTimeSeriesHeadersForFilter](#)
 - [getTimeSeriesHeadersForFilter2](#)
 - [getTimeSeriesForFilter](#)
 - [getTimeSeriesForFilter2](#)
 - [getTimeSeries](#)
 - [getTimeSeriesAsStream](#)
 - [getWorkflows](#)
 - [getSamplesAsStream](#)
 - [getModifiers](#)
 - [getTimeSeriesModifiers](#)
 - [Setter methods](#)
 - [putTimeSeriesForFilters](#)
 - [putModifiers](#)
 - [Run methods](#)
 - [runTask](#)
 - [runTaskWithRunParameters](#)
 - [getTaskRunStatus](#)
- [Installing a Tomcat Fews PI Service](#)
- [Configuration](#)
 - [FEWS Client Config File \(clientConfigFileId\)](#)
- [Example SOAP calls](#)
- [Introduction](#)
- [PI SOAP Web Service API](#)
 - [Getter methods](#)
 - [getTimeZoneld](#)
 - [getFilters](#)
 - [getLocations](#)
 - [getLocationsAsStream](#)
 - [getParameters](#)
 - [getParametersAsStream](#)
 - [getTimeSeriesHeadersForFilter](#)
 - [getTimeSeriesHeadersForFilter2](#)
 - [getTimeSeriesForFilter](#)
 - [getTimeSeriesForFilter2](#)
 - [getTimeSeries](#)
 - [getTimeSeriesAsStream](#)
 - [getWorkflows](#)
 - [getSamplesAsStream](#)
 - [getModifiers](#)
 - [getTimeSeriesModifiers](#)
 - [Setter methods](#)
 - [putTimeSeriesForFilters](#)
 - [putModifiers](#)
 - [Run methods](#)
 - [runTask](#)
 - [runTaskWithRunParameters](#)
 - [getTaskRunStatus](#)
- [Installing a Tomcat Fews PI Service](#)
- [Configuration](#)
 - [FEWS Client Config File \(clientConfigFileId\)](#)
- [Example SOAP calls](#)

Introduction

The Tomcat Fews PI service is hosted in a Tomcat service container. This service allows SOAP clients to interact with a FewsPiService that is connected to a FEWS system through the FEWS [DataAccessComponent](#). With this API the SOAP client can retrieve data from the FEWS system. Before a client application can access the FEWS system there is some configuration work that needs to be done.

For some examples on how to connect to the service, see: [FEWS PI-XML SOAP Examples](#). For some sample SOAP request please see [FEWS PI SOAP Web Service Request examples](#).

PI SOAP Web Service API

Description of the methods provided by the PI SOAP Service API.

Getter methods

getTimeZoneId

```
String getTimeZoneId(String piVersion);
```

Get ID of Configured timezone for the webservice.

- clientId: *<id not used>*
- returns: String representation of time zone.

getFilters

```
String getFilters(String filterId, String piVersion);
```

Retrieve Pi Filters configuration from the Fews instance hosting the current FewsPiService.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher)

- filterId: Subset filter id. (optional).
- piVersion: File format version (optional).
- returns: PiFilters xml file content.

getLocations

```
String getLocations(String clientId, String filterId, String piVersion);
```

Retrieve Pi Locations file containing all locations that are available for the passed 'filterId' argument.

If argument 'null' is passed then all locations configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId *<id not used>*.
- filterId Filter id (optional).
- piVersion File format version (optional)
- returns PiLocations XML file content.

getLocationsAsStream

```
DataHandler getLocationsAsStream(String filterId, String piVersion);
```

Retrieve Pi Locations file containing all locations that are available for the passed 'filterId' argument.

If argument 'null' is passed then all locations configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- filterId Filter id (optional).
- piVersion File format version (optional)
- returns javax.activation.DataHandler that allows client to stream response. The response is returned following the MTOM protocol.

getParameters

```
String getParameters(String clientId, String filterId, String piVersion);
```

Retrieve Pi Parameters file containing all parameters that are available for the passed 'filterId' argument.

If argument 'null' is passed then all parameters configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- filterId Filter id (optional).
- piVersion File format version (optional)
- returns PiTimeSeriesParameters XML file content.

getParametersAsStream

```
DataHandler getParametersAsStream(String filterId, String piVersion);
```

Retrieve Pi Parameters file containing all parameters that are available for the passed 'filterId' argument.

If argument 'null' is passed then all parameters configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- filterId Filter id (optional).
- piVersion File format version (optional)
- returns javax.activation.DataHandler that allows client to stream response.

getTimeSeriesHeadersForFilter

```
String getTimeSeriesHeadersForFilter(String clientId, Date startTime, Date timeZero, Date endTime, String filterId, String[] locationIds, String[] parameterIds, boolean useDisplayUnits, String piVersion);
```

Read the timeseries header information from the webservice. Returns a pi timeseries xml file containing the timeseries headers information belonging to the filter defined by the 'filterId'.

Retrieve the timeseries data using the method *getTimeSeriesForFilter*

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- useDisplayUnits Export values using display units (optional).
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeriesHeadersForFilter2

```
String getTimeSeriesHeadersForFilter2(String clientId, Date startTime, Date timeZero, Date endTime, String filterId, String[] locationIds, String[] parameterIds, boolean useDisplayUnits, String ensembleId, String piVersion);
```

Read the timeseries header information from the webservice. Returns a pi timeseries xml file containing the timeseries headers information belonging to the filter defined by the 'filterId'.

Retrieve the timeseries data using the method *getTimeSeriesForFilter*

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- useDisplayUnits Export values using display units (optional).
- ensembleId Ensemble id (optional)
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeriesForFilter

```
String getTimeSeriesForFilter(String clientId, Date startTime, Date timeZero, Date endTime, String filterId,
String[] locationIds, String[] parameterIds, boolean convertDatum, boolean useDisplayUnits, String piVersion);
```

Returns a pi timeseries xml file containing the timeseries data belonging to the filter defined by the 'filterId'.

The 'convertDatum' argument is to allow timeseries that support a global datum to have their values converted from a value relative to the location height to an absolute value.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- convertDatum Convert values from relative location height to absolute height values (optional).
- useDisplayUnits Export values using display units (optional).
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeriesForFilter2

```
String getTimeSeriesForFilter2(String clientId, Date startTime, Date timeZero, Date endTime, String filterId,
String[] locationIds, String[] parameterIds, boolean convertDatum, boolean useDisplayUnits, String ensembleId,
String piVersion);
```

Returns a pi timeseries xml file containing the timeseries data belonging to the filter defined by the 'filterId'.

The 'convertDatum' argument is to allow timeseries that support a global datum to have their values converted from a value relative to the location height to an absolute value.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- convertDatum Convert values from relative location height to absolute height values (optional).
- useDisplayUnits Export values using display units (optional).
- ensembleId Ensemble id (optional)
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeries

```
String getTimeSeries(QueryParams queryParameters);
```

QueryParameters is defined by following schema definition:

```

<xs:complexType name="queryParameters">
  <xs:sequence>
    <xs:element name="convertDatum" type="xs:boolean"/></xs:element>
    <xs:element name="endCreationTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="endForecastTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="endTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="ensembleId" type="xs:string"/></xs:element>
    <xs:element name="externalForecastTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="filterId" type="xs:string" minOccurs="0"/></xs:element>
    <xs:element name="forecastSearchCount" type="xs:string" minOccurs="0"/></xs:element>
    <xs:element name="importFromExternalDataSource" type="xs:boolean"/></xs:element>
    <xs:element name="locationIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/></xs:element>
    <xs:element name="omitMissing" type="xs:boolean"/></xs:element>
    <xs:element name="onlyHeaders" type="xs:boolean"/></xs:element>
    <xs:element name="parameterIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/></xs:element>
    <xs:element name="piVersion" type="xs:string" minOccurs="0"/></xs:element>
    <xs:element name="showStatistics" type="xs:boolean"/></xs:element>
    <xs:element name="showThresholds" type="xs:boolean"/></xs:element>
    <xs:element name="startCreationTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="startForecastTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="startTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="useDisplayUnits" type="xs:boolean"/></xs:element>
  </xs:sequence>
</xs:complexType>

```

Returns a pi timeseries xml file containing the timeseries data belonging to the query parameters defined in the query parameters object.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- **convertDatum** Convert values from relative location height to absolute height values.
- **endCreationTime** End time of search period that looks for creation time of timeseries. Note: creation time of timeseries is actually the creation time of the task that produced/imported these timeseries.
- **endForecastTime** End time of search period that looks for timeseries produced by forecasts that have their forecast time within this period.
- **endTime** End time of search period that looks for timeseries values that lie within this period.
- **ensembleId** Ensemble identifier of for timeseries
- **externalForecastTime** Time value of for external forecast time.
- **filterId** Filter id.
- **forecastSearchCount** Number of forecast runs to return when using start- and end- forecast time. Default is 1.
- **importFromExternalDataSource** Option to look data up in a linked external data source (eg. archive) or not.
- **locationIds** Subset of locations for which to retrieve timeseries.
- **omitMissing** Toggle omitting or returning of missing values in response
- **onlyHeaders** Toggle to return only header information or also data
- **parameterIds** Subset of parameters for which to retrieve timeseries.
- **piVersion** File format version
- **showStatistics** Toggle to return statistics information about timeseries. Use in combination with 'onlyHeaders=true'. Returns additional information about data availability of timeseries (available from 2015.01)
- **showThresholds** Option to toggle the returning of threshold information in the headers
- **startCreationTime** Start time of search period that looks for creation time of timeseries. Note: creation time of timeseries is actually the creation time of the task that produced/imported these timeseries.
- **startForecastTime** Start time of search period that looks for timeseries produced by forecasts that have their forecast time within this period.
- **startTime** Start time of search period that looks for timeseries values that lie within this period..
- **useDisplayUnits** Export values using display units.
- **return PiTimeseries** xml file content.

getTimeSeriesAsStream

```
DataHandler getTimeSeriesAsStream(QueryParams queryParameters);
```

Same as getTimeSeries method except this method returns a DataHandler using the MTOM protocol.

getWorkflows

```
String getWorkflows(String piVersion);
```

Retrieve Pi Workflow file containing all workflows that can be run using the method **runTask**.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- piVersion File format version (optional)
- return PiWorkflows xml file content.

getSamplesAsStream

```
DataHandler getSamplesAsStream(SampleQueryParams sampleQueryParams);
```

Returns a pi samples xml file containing the sample data matching the query parameters. Since FEWS2016.01.

SampleQueryParameters is defined by following schema definition:

```
<xs:complexType name="sampleQueryParameters">
  <xs:sequence>
    <xs:element name="endCreationTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="endTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="filterId" type="xs:string" minOccurs="0"/>
    <xs:element name="locationIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="omitMissing" type="xs:boolean"/>
    <xs:element name="onlyHeaders" type="xs:boolean"/>
    <xs:element name="parameterIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="qualifierIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="sampleIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="startCreationTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="startTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="version" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

Returns a pi samples xml file as DataHandler using the MTOM protocol containing the sample data belonging to the query parameters defined in the query parameters object.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.21 or higher).

- endCreationTime End time of search period that looks for creation time of samples. Note: creation time of samples is actually the creation time of the task that produced/imported these samples.
- endTime End time of search period that looks for sample values that lie within this period.
- filterId Filter id.
- locationIds Subset of locations for which to retrieve samples.
- omitMissing Toggle omitting or returning of missing values in response
- onlyHeaders Toggle to return only header information or also data
- parameterIds Subset of parameters for which to retrieve samples.
- sampleIds Subset of sample id's for which to retrieve samples
- startCreationTime Start time of search period that looks for creation time of samples. Note: creation time of samples is actually the creation time of the task that produced/imported these samples.
- startTime Start time of search period that looks for sample values that lie within this period..
- version Pi file format version
- return Samples xml file content.

Since all parameters are optional, some defaults are chosen for the search period in case no search period or creation period was specified. The following rules apply:

- If no startCreationTime and endCreationTime have been set, the startTime and endTime are used to determine the search period.
- If no startTime and endTime have been specified, the search period will be set to the current system time minus one day and one hour until the current system time plus one day and one hour.

getModifiers

```
String getModifiers(Date startTime, Date endTime, String modifierTypeId);
```

- *startTime*: start time of modifiers search period
- *endTime*: end time of modifiers search period.
- *modifierTypeId*: filter on modifiers by modifierTypeId as specified in the ModifierTypes.xml configuration. If modifier type cannot be found, not filtering on type is done.

If no startTime and endTime are given, the search period is any time.

getTimeSeriesModifiers

```
String getTimeSeriesModifiers(Date startTime, Date endTime, String userId, boolean active, String[] locationIds, String[] moduleInstanceIds, String[] userDefinedDescriptions, String modifierTypeId);
```

Get a list of all timeSeries modifiers

- startTime Start of the search period
- endTime end of the search period
- userId If this option is used only the modifiers of this user will be returned
- active If this option is set to true only the enabled modifiers will be returned, otherwise all modifiers will be returned
- locationIds Only return modifiers which are made for one of the specified locations if this parameter is set
- moduleInstanceIds Only return modifiers which are made for one of the specified module instances if this parameter is set
- userDefinedDescriptions Only return modifiers which have a user defined description which is equal to one the descriptions in this list if this parameter is set
- modifierTypeId Only return modifiers which are of the defined type if this parameter is set

Setter methods

putTimeSeriesForFilters

```
void putTimeSeriesForFilters(String clientId, String filterId, String piTimeSeriesXmlContent, byte[] piTimeSeriesBinaryContent, boolean convertDatum);
```

Write timeseries data to the FEWS system using the timeseries sets defined by the filters.

For performance reasons it is possible to split the timeseries header information from the timeseries data. The header information is stored in the *piTimeSeriesXmlContent* and the timeseries data is stored in the *piTimeSeriesBinaryContent*. If both header information and data is stored in the XML content the the 'piTimeSeriesBinaryContent' argument can be null.

The 'convertDatum' argument is to allow timeseries that support a datum to have their values converted to a value relative to the location height. If values are already relative to location then enter FALSE or omit.

- clientId *<id not used>*.
- filterId Filter id for when the input timeseries maps to multiple internal timeseries. Within the scope of the filter the input timeseries should only map to one internal timeseries.
- piTimeSeriesXmlContent PiTimeseries xml file content
- piTimeSeriesBinaryContent FastInfoSet PiTimeseries file content(optional)
- convertDatum Convert values from relative location height to absolute height values (optional).

The endpoint accepts a time format with or without milliseconds, and automatically recognises the correct format.

Example:

```
<event date="2018-02-12" time="09:15:00.000" value="0.186" flag="2"/>
<event date="2018-02-12" time="09:16:00.250" value="0.186" flag="2"/>
```

```
<event date="2018-02-12" time="09:15:00" value="0.186" flag="2"/>
<event date="2018-02-12" time="09:16:00" value="0.186" flag="2"/>
```

putModifiers

```
void putModifiers(String piModifiersXmlContent, boolean commitModifiers, boolean deleteAllModifiers);
```

The method can be used to upload new modifiers to a FEWS system by using a xml file. The option commitModifiers indicates if the modifiers are committed are not after the upload.

- piModifiersXml. A String which contains modifiers defined in xml-format,
- commitModifiers. Indicates if the modifiers are committed after the upload.
- deleteAllModifiers. Indicates if the modifiers should be deleted prior to inserting the new modifiers.

Run methods

runTask

```
String runTask(String clientId, String workflowId, Date startTime, Date timeZero, Date endTime, String coldStateId, String scenarioId, String piParametersXml, String userId, String description);
```

Runs a workflow task for a given **workflowId**. Returns a handle to the task in the form of a **taskId**. This taskId can be used to track the status of the workflow using method call **getTaskRunStatus**. Additionally there are a few arguments that can be passed to control the run.

- clientId *<id not used>*.
- workflowId Identifier of the task to run.
- startTime Start of run period. Used for state selection period.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime End of run period. Used to define forecast length.
- coldStateId Id of a coldstate. Can be used to force state selection (optional).
- scenarioId Id of a predefined WhatIf scenario. Can be used to alter run parameters (optional).
- piParametersXml Contents of a [Pi ModelParameters XML](#) file. PI ModelParameters can be exported by the General Adapter to provide information to external models being run by FEWS (optional).
- userId User id (optional).
- description Descriptive text to identify run (optional)
- return taskId of run.

runTaskWithRunParameters

```
String runTaskWithRunParameters(RunParameters runParameters);
```

RunParameters is defined by following schema definition:

```
<xs:complexType name="runParameters">
  <xs:sequence>
    <xs:element name="coldStateId" type="xs:string" minOccurs="0"/>
    <xs:element name="description" type="xs:string" minOccurs="0"/>
    <xs:element name="endTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="piModuleParametersXml" type="xs:string" minOccurs="0"/>
    <xs:element name="startTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="timeZero" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="useColdState" type="xs:boolean"/>
    <xs:element name="workflowId" type="xs:string" minOccurs="0"/>
    <xs:element name="whatIfId" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

Runs a workflow task for a given **workflowId**. Returns a handle to the task in the form of a **taskId**. This taskId can be used to track the status of the workflow using method call **getTaskRunStatus**. Additionally there are a few arguments that can be passed to control the run.

- workflowId Identifier of the task to run.
- startTime Start of run period. Used for state selection period.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime End of run period. Used to define forecast length.
- coldStateId Id of a coldstate. Can be used to force state selection (optional).
- scenarioId Id of a predefined WhatIf scenario. Can be used to alter run parameters (optional).
- piParametersXml Contents of a [Pi ModelParameters XML](#) file. PI ModelParameters can be exported by the General Adapter to provide information to external models being run by FEWS (optional).
- useColdState Option to force the use of a cold state even when warm state is available.
- description Descriptive text to identify run (optional)
- return taskId of run.

getTaskRunStatus

```
String getTaskRunStatus(String taskId, long maxWaitMillis)
```

Runs a workflow task for a given **workflowId**. Returns a handle to the task in the form of a **taskId**. This taskId can be used to track the status of the workflow using method call **getTaskRunStatus**. Additionally there are a few arguments that can be passed to control the run.

- taskId Id returned by runTask.
- maxWaitMillis time in milliseconds to wait for response
- return task run status which can be one of the following values:


```
Valid values are
I = Invalid,
P = Pending,
T = Terminated,
R = running,
F = Failed,
C = Completed fully successful,
D = Completed partly successful,
A = Approved,
B = Approved partly successful
null = No status available (produces when method call times-out)
```

Installing a Tomcat Fews PI Service

N.B.: The DAC installation has become obsolete since FEWS 2017.02.

The file [DAC installation guide \(final\).doc](#) describes how to install a Tomcat instance containing a DataAccessComponent instance. Once these steps are completed the FewsPiService.WAR can be deployed using a context xml file such as [FewsPiService.xml](#).

For the online installation manual check [here](#).

Configuration

N.B.: The DAC configuration has become obsolete since FEWS 2017.02.

There are two levels of configuration. The first is the *DataAccessComponent* which is a FEWS client hosted as a GlobalResource in Tomcat. The *DataAccessComponent* uses the regular FEWS configuration which falls outside the scope of this documentation. The second level of configuration is what is used by the FewsPiService webservice which has been deployed in Tomcat. To deploy this webservice a context XML file is required which looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
Licensed to the Apache Software Foundation (ASF) under one or more
contributor license agreements. See the NOTICE file distributed with
this work for additional information regarding copyright ownership.
The ASF licenses this file to You under the Apache License, Version 2.0
(the "License"); you may not use this file except in compliance with
the License. You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
-->

<!-- Location of specific AI war file. (Can be a symlink) -->
<Context docBase="${catalina.home}/fews/FewsPiService.war" antiJARLocking="true" crossContext="true" >
    <!-- 'clientConfigFileId' points to a FEWS configuration file in the FEWS configuration directory
    'PiClientConfigFiles'. -->
    <Parameter name="clientConfigFileId" value="FewsPiService.properties" override="false"/>
    <!-- resourceId must match the 'name' of 'ResourceLink' below -->
    <Parameter name="resourceId" value="DAC" override="false"/>
    <Parameter name="serviceName" value="FewsPiServiceImpl" override="false"/>
    <Parameter name="namespaceUri" value="http://fewpiservice.wldelft.nl" override="false"/>
    <Parameter name="portName" value="FewsPiServiceImplPort" override="false"/>
    <!-- 'fewspi_config_naam' must match the file name of this configuration file without the XML extension. --
    >
    <Parameter name="wsdl" value="http://localhost:8080/FewsPiService/fewspiservice?wsdl" override="false"/>
    <!-- Global 'fews_dac' must be the same as the DAC resource name in the file 'server.xml'-->
    <ResourceLink name="DAC"
        global="fews_dac"
        type="nl.wldelft.fews.system.data.dac.DataAccessComponent"/>
    <Loader loaderClass="nl.wldelft.fews.system.data.dac.DacClassLoader" delegate="true"/>
</Context>
```

In the above configuration example you see a number of 'Parameter' values which will be explained below:

- *clientConfigFileId* = This is a reference to a FEWS configuration file. With this file it is possible to configure the webservice instance. We will discuss this file in more detail below. The field is optional. If omitted then the **defaultFilterId** must be configured in the **Filters.xml** FEWS configuration file.
- *resourceId* = Tells the webservice what Tomcat resource link to use.

The next parameters have been added for the FewsPiService test page. This is the page that appears when browsing to the root URI of the FewsPiService. Technically these parameters would not be required, but because it is a 'test' page, the configuration has not been optimized yet.

- *serviceName* = Name of the service
- *namespaceUri* = Namespace of the service
- *portName* = Name of the service port
- *wsdl* = URI of the service WSDL file

FEWS Client Config File (clientConfigFileId)

This is a simple text configuration file that is located in the FEWS configuration in the directory:

```
%REGION_HOME%/Config/PiClientConfigFiles/FewsPiService.properties
```

In the above example the **clientConfigFileId** would be **FewsPiService.properties**.

N.B.: Since FEWS 20170.2 the clientConfigFileId can no longer be configured and should always be called FewsPiService.properties.

For more information about the possible properties, see [FEWS Web Services Configuration FewsPiService.properties \(deprecated since 2022.02\)](#).

Example SOAP calls

See [soap-example-calls.zip](#) for examples SOAP client calls.

[DAC installation guide \(final\).doc](#) (DAC installation is obsolete since FEWS 2017.02)

- Introduction
- PI SOAP Web Service API
 - Getter methods
 - [getTimeZoneld](#)
 - [getFilters](#)
 - [getLocations](#)
 - [getLocationsAsStream](#)
 - [getParameters](#)
 - [getParametersAsStream](#)
 - [getTimeSeriesHeadersForFilter](#)
 - [getTimeSeriesHeadersForFilter2](#)
 - [getTimeSeriesForFilter](#)
 - [getTimeSeriesForFilter2](#)
 - [getTimeSeries](#)
 - [getTimeSeriesAsStream](#)
 - [getWorkflows](#)
 - [getSamplesAsStream](#)
 - [getModifiers](#)
 - [getTimeSeriesModifiers](#)
 - Setter methods
 - [putTimeSeriesForFilters](#)
 - [putModifiers](#)
 - Run methods
 - [runTask](#)
 - [runTaskWithRunParameters](#)
 - [getTaskRunStatus](#)
- Installing a Tomcat Fews PI Service
- Configuration
 - [FEWS Client Config File \(clientConfigFileId\)](#)
- Example SOAP calls
- Introduction
- PI SOAP Web Service API
 - Getter methods
 - [getTimeZoneld](#)
 - [getFilters](#)
 - [getLocations](#)
 - [getLocationsAsStream](#)
 - [getParameters](#)
 - [getParametersAsStream](#)
 - [getTimeSeriesHeadersForFilter](#)
 - [getTimeSeriesHeadersForFilter2](#)
 - [getTimeSeriesForFilter](#)
 - [getTimeSeriesForFilter2](#)
 - [getTimeSeries](#)
 - [getTimeSeriesAsStream](#)
 - [getWorkflows](#)
 - [getSamplesAsStream](#)
 - [getModifiers](#)
 - [getTimeSeriesModifiers](#)
 - Setter methods
 - [putTimeSeriesForFilters](#)
 - [putModifiers](#)
 - Run methods
 - [runTask](#)
 - [runTaskWithRunParameters](#)
 - [getTaskRunStatus](#)
- Installing a Tomcat Fews PI Service
- Configuration
 - [FEWS Client Config File \(clientConfigFileId\)](#)
- Example SOAP calls



The Delft-FEWS PI SOAP Web Services has been deprecated and will be removed. See: [Delft-FEWS End of Life Modules and Displays#FEWSEndofLifeModulesandDisplays-FEWSWebServicesAPIsthatwillberemoved](#)

Introduction

The Tomcat Fews PI service is hosted in a Tomcat service container. This service allows SOAP clients to interact with a FewsPiService that is connected to a FEWS system through the FEWS [DataAccessComponent](#). With this API the SOAP client can retrieve data from the FEWS system. Before a client application can access the FEWS system there is some configuration work that needs to be done.

For some examples on how to connect to the service, see: [FEWS PI-XML SOAP Examples](#). For some sample SOAP request please see [FEWS PI SOAP Web Service Request examples](#).

PI SOAP Web Service API

Description of the methods provided by the PI SOAP Service API.

Getter methods

getTimeZoneId

```
String getTimeZoneId(String piVersion);
```

Get ID of Configured timezone for the webservice.

- clientId: *<id not used>*
- returns: String representation of time zone.

getFilters

```
String getFilters(String filterId, String piVersion);
```

Retrieve Pi Filters configuration from the Fews instance hosting the current FewsPiService.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher)

- filterId: Subset filter id. (optional).
- piVersion: File format version (optional).
- returns: PiFilters xml file content.

getLocations

```
String getLocations(String clientId, String filterId, String piVersion);
```

Retrieve Pi Locations file containing all locations that are available for the passed 'filterId' argument.

If argument 'null' is passed then all locations configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId *<id not used>*.
- filterId Filter id (optional).
- piVersion File format version (optional)
- returns PiLocations XML file content.

getLocationsAsStream

```
DataHandler getLocationsAsStream(String filterId, String piVersion);
```

Retrieve Pi Locations file containing all locations that are available for the passed 'filterId' argument.

If argument 'null' is passed then all locations configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- filterId Filter id (optional).
- piVersion File format version (optional)
- returns javax.activation.DataHandler that allows client to stream response. The response is returned following the MTOM protocol.

getParameters

```
String getParameters(String clientId, String filterId, String piVersion);
```

Retrieve Pi Parameters file containing all parameters that are available for the passed 'filterId' argument.

If argument 'null' is passed then all parameters configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- filterId Filter id (optional).
- piVersion File format version (optional)
- returns PiTimeSeriesParameters XML file content.

getParametersAsStream

```
DataHandler getParametersAsStream(String filterId, String piVersion);
```

Retrieve Pi Parameters file containing all parameters that are available for the passed 'filterId' argument.

If argument 'null' is passed then all parameters configured in the pre-defined filter will be returned.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- filterId Filter id (optional).
- piVersion File format version (optional)
- returns javax.activation.DataHandler that allows client to stream response.

getTimeSeriesHeadersForFilter

```
String getTimeSeriesHeadersForFilter(String clientId, Date startTime, Date timeZero, Date endTime, String filterId, String[] locationIds, String[] parameterIds, boolean useDisplayUnits, String piVersion);
```

Read the timeseries header information from the webservice. Returns a pi timeseries xml file containing the timeseries headers information belonging to the filter defined by the 'filterId'.

Retrieve the timeseries data using the method *getTimeSeriesForFilter*

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- useDisplayUnits Export values using display units (optional).
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeriesHeadersForFilter2

```
String getTimeSeriesHeadersForFilter2(String clientId, Date startTime, Date timeZero, Date endTime, String filterId, String[] locationIds, String[] parameterIds, boolean useDisplayUnits, String ensembleId, String piVersion);
```

Read the timeseries header information from the webservice. Returns a pi timeseries xml file containing the timeseries headers information belonging to the filter defined by the 'filterId'.

Retrieve the timeseries data using the method *getTimeSeriesForFilter*

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- useDisplayUnits Export values using display units (optional).
- ensembleId Ensemble id (optional)
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeriesForFilter

```
String getTimeSeriesForFilter(String clientId, Date startTime, Date timeZero, Date endTime, String filterId,
String[] locationIds, String[] parameterIds, boolean convertDatum, boolean useDisplayUnits, String piVersion);
```

Returns a pi timeseries xml file containing the timeseries data belonging to the filter defined by the 'filterId'.

The 'convertDatum' argument is to allow timeseries that support a global datum to have their values converted from a value relative to the location height to an absolute value.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- convertDatum Convert values from relative location height to absolute height values (optional).
- useDisplayUnits Export values using display units (optional).
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeriesForFilter2

```
String getTimeSeriesForFilter2(String clientId, Date startTime, Date timeZero, Date endTime, String filterId,
String[] locationIds, String[] parameterIds, boolean convertDatum, boolean useDisplayUnits, String ensembleId,
String piVersion);
```

Returns a pi timeseries xml file containing the timeseries data belonging to the filter defined by the 'filterId'.

The 'convertDatum' argument is to allow timeseries that support a global datum to have their values converted from a value relative to the location height to an absolute value.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- clientId <id not used>.
- startTime start date/time of run.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime end date/time of run.
- filterId Filter id (optional).
- locationIds Subset of locations for which to retrieve timeseries (optional).
- parameterIds Subset of parameters for which to retrieve timeseries (optional).
- convertDatum Convert values from relative location height to absolute height values (optional).
- useDisplayUnits Export values using display units (optional).
- ensembleId Ensemble id (optional)
- piVersion File format version (optional)
- return PiTimeseries xml file content.

getTimeSeries

```
String getTimeSeries(QueryParams queryParameters);
```

QueryParameters is defined by following schema definition:

```

<xs:complexType name="queryParameters">
  <xs:sequence>
    <xs:element name="convertDatum" type="xs:boolean"/></xs:element>
    <xs:element name="endCreationTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="endForecastTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="endTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="ensembleId" type="xs:string"/></xs:element>
    <xs:element name="externalForecastTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="filterId" type="xs:string" minOccurs="0"/></xs:element>
    <xs:element name="forecastSearchCount" type="xs:string" minOccurs="0"/></xs:element>
    <xs:element name="importFromExternalDataSource" type="xs:boolean"/></xs:element>
    <xs:element name="locationIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/></xs:element>
    <xs:element name="omitMissing" type="xs:boolean"/></xs:element>
    <xs:element name="onlyHeaders" type="xs:boolean"/></xs:element>
    <xs:element name="parameterIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/></xs:element>
    <xs:element name="piVersion" type="xs:string" minOccurs="0"/></xs:element>
    <xs:element name="showStatistics" type="xs:boolean"/></xs:element>
    <xs:element name="showThresholds" type="xs:boolean"/></xs:element>
    <xs:element name="startCreationTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="startForecastTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="startTime" type="xs:dateTime" minOccurs="0"/></xs:element>
    <xs:element name="useDisplayUnits" type="xs:boolean"/></xs:element>
  </xs:sequence>
</xs:complexType>

```

Returns a pi timeseries xml file containing the timeseries data belonging to the query parameters defined in the query parameters object.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- **convertDatum** Convert values from relative location height to absolute height values.
- **endCreationTime** End time of search period that looks for creation time of timeseries. Note: creation time of timeseries is actually the creation time of the task that produced/imported these timeseries.
- **endForecastTime** End time of search period that looks for timeseries produced by forecasts that have their forecast time within this period.
- **endTime** End time of search period that looks for timeseries values that lie within this period.
- **ensembleId** Ensemble identifier of for timeseries
- **externalForecastTime** Time value of for external forecast time.
- **filterId** Filter id.
- **forecastSearchCount** Number of forecast runs to return when using start- and end- forecast time. Default is 1.
- **importFromExternalDataSource** Option to look data up in a linked external data source (eg. archive) or not.
- **locationIds** Subset of locations for which to retrieve timeseries.
- **omitMissing** Toggle omitting or returning of missing values in response
- **onlyHeaders** Toggle to return only header information or also data
- **parameterIds** Subset of parameters for which to retrieve timeseries.
- **piVersion** File format version
- **showStatistics** Toggle to return statistics information about timeseries. Use in combination with 'onlyHeaders=true'. Returns additional information about data availability of timeseries (available from 2015.01)
- **showThresholds** Option to toggle the returning of threshold information in the headers
- **startCreationTime** Start time of search period that looks for creation time of timeseries. Note: creation time of timeseries is actually the creation time of the task that produced/imported these timeseries.
- **startForecastTime** Start time of search period that looks for timeseries produced by forecasts that have their forecast time within this period.
- **startTime** Start time of search period that looks for timeseries values that lie within this period..
- **useDisplayUnits** Export values using display units.
- **return PiTimeseries** xml file content.

getTimeSeriesAsStream

```
DataHandler getTimeSeriesAsStream(QueryParams queryParameters);
```

Same as getTimeSeries method except this method returns a DataHandler using the MTOM protocol.

getWorkflows

```
String getWorkflows(String piVersion);
```

Retrieve Pi Workflow file containing all workflows that can be run using the method **runTask**.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.9 or higher).

- piVersion File format version (optional)
- return PiWorkflows xml file content.

getSamplesAsStream

```
DataHandler getSamplesAsStream(SampleQueryParams sampleQueryParams);
```

Returns a pi samples xml file containing the sample data matching the query parameters. Since FEWS2016.01.

SampleQueryParameters is defined by following schema definition:

```
<xs:complexType name="sampleQueryParameters">
  <xs:sequence>
    <xs:element name="endCreationTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="endTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="filterId" type="xs:string" minOccurs="0"/>
    <xs:element name="locationIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="omitMissing" type="xs:boolean"/>
    <xs:element name="onlyHeaders" type="xs:boolean"/>
    <xs:element name="parameterIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="qualifierIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="sampleIds" type="xs:string" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="startCreationTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="startTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="version" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

Returns a pi samples xml file as DataHandler using the MTOM protocol containing the sample data belonging to the query parameters defined in the query parameters object.

The PiVersion defines the format of the return file content. If omitted the latest version of the Pi file format will be used. (Expected 1.21 or higher).

- endCreationTime End time of search period that looks for creation time of samples. Note: creation time of samples is actually the creation time of the task that produced/imported these samples.
- endTime End time of search period that looks for sample values that lie within this period.
- filterId Filter id.
- locationIds Subset of locations for which to retrieve samples.
- omitMissing Toggle omitting or returning of missing values in response
- onlyHeaders Toggle to return only header information or also data
- parameterIds Subset of parameters for which to retrieve samples.
- sampleIds Subset of sample id's for which to retrieve samples
- startCreationTime Start time of search period that looks for creation time of samples. Note: creation time of samples is actually the creation time of the task that produced/imported these samples.
- startTime Start time of search period that looks for sample values that lie within this period..
- version Pi file format version
- return Samples xml file content.

Since all parameters are optional, some defaults are chosen for the search period in case no search period or creation period was specified. The following rules apply:

- If no startCreationTime and endCreationTime have been set, the startTime and endTime are used to determine the search period.
- If no startTime and endTime have been specified, the search period will be set to the current system time minus one day and one hour until the current system time plus one day and one hour.

getModifiers

```
String getModifiers(Date startTime, Date endTime, String modifierTypeId);
```

- *startTime*: start time of modifiers search period
- *endTime*: end time of modifiers search period.
- *modifierTypeId*: filter on modifiers by modifierTypeId as specified in the ModifierTypes.xml configuration. If modifier type cannot be found, not filtering on type is done.

If no startTime and endTime are given, the search period is any time.

getTimeSeriesModifiers


```
String getTimeSeriesModifiers(Date startTime, Date endTime, String userId, boolean active, String[] locationIds, String[] moduleInstanceIds, String[] userDefinedDescriptions, String modifierTypeId);
```

Get a list of all timeSeries modifiers

- startTime Start of the search period
- endTime end of the search period
- userId If this option is used only the modifiers of this user will be returned
- active If this option is set to true only the enabled modifiers will be returned, otherwise all modifiers will be returned
- locationIds Only return modifiers which are made for one of the specified locations if this parameter is set
- moduleInstanceIds Only return modifiers which are made for one of the specified module instances if this parameter is set
- userDefinedDescriptions Only return modifiers which have a user defined description which is equal to one the descriptions in this list if this parameter is set
- modifierTypeId Only return modifiers which are of the defined type if this parameter is set

Setter methods

putTimeSeriesForFilters

```
void putTimeSeriesForFilters(String clientId, String filterId, String piTimeSeriesXmlContent, byte[] piTimeSeriesBinaryContent, boolean convertDatum);
```

Write timeseries data to the FEWS system using the timeseries sets defined by the filters.

For performance reasons it is possible to split the timeseries header information from the timeseries data. The header information is stored in the *piTimeSeriesXmlContent* and the timeseries data is stored in the *piTimeSeriesBinaryContent*. If both header information and data is stored in the XML content the the 'piTimeSeriesBinaryContent' argument can be null.

The 'convertDatum' argument is to allow timeseries that support a datum to have their values converted to a value relative to the location height. If values are already relative to location then enter FALSE or omit.

- clientId *<id not used>*.
- filterId Filter id for when the input timeseries maps to multiple internal timeseries. Within the scope of the filter the input timeseries should only map to one internal timeseries.
- piTimeSeriesXmlContent PiTimeseries xml file content
- piTimeSeriesBinaryContent FastInfoSet PiTimeseries file content(optional)
- convertDatum Convert values from relative location height to absolute height values (optional).

The endpoint accepts a time format with or without milliseconds, and automatically recognises the correct format.

Example:

```
<event date="2018-02-12" time="09:15:00.000" value="0.186" flag="2"/>
<event date="2018-02-12" time="09:16:00.250" value="0.186" flag="2"/>
```

```
<event date="2018-02-12" time="09:15:00" value="0.186" flag="2"/>
<event date="2018-02-12" time="09:16:00" value="0.186" flag="2"/>
```

putModifiers

```
void putModifiers(String piModifiersXmlContent, boolean commitModifiers, boolean deleteAllModifiers);
```

The method can be used to upload new modifiers to a FEWS system by using a xml file. The option commitModifiers indicates if the modifiers are committed are not after the upload.

- piModifiersXml. A String which contains modifiers defined in xml-format,
- commitModifiers. Indicates if the modifiers are committed after the upload.
- deleteAllModifiers. Indicates if the modifiers should be deleted prior to inserting the new modifiers.

Run methods

runTask

```
String runTask(String clientId, String workflowId, Date startTime, Date timeZero, Date endTime, String coldStateId, String scenarioId, String piParametersXml, String userId, String description);
```

Runs a workflow task for a given **workflowId**. Returns a handle to the task in the form of a **taskId**. This taskId can be used to track the status of the workflow using method call **getTaskRunStatus**. Additionally there are a few arguments that can be passed to control the run.

- clientId *<id not used>*.
- workflowId Identifier of the task to run.
- startTime Start of run period. Used for state selection period.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime End of run period. Used to define forecast length.
- coldStateId Id of a coldstate. Can be used to force state selection (optional).
- scenarioId Id of a predefined WhatIf scenario. Can be used to alter run parameters (optional).
- piParametersXml Contents of a [Pi ModelParameters XML](#) file. PI ModelParameters can be exported by the General Adapter to provide information to external models being run by FEWS (optional).
- userId User id (optional).
- description Descriptive text to identify run (optional)
- return taskId of run.

runTaskWithRunParameters

```
String runTaskWithRunParameters(RunParameters runParameters);
```

RunParameters is defined by following schema definition:

```
<xs:complexType name="runParameters">
  <xs:sequence>
    <xs:element name="coldStateId" type="xs:string" minOccurs="0"/>
    <xs:element name="description" type="xs:string" minOccurs="0"/>
    <xs:element name="endTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="piModuleParametersXml" type="xs:string" minOccurs="0"/>
    <xs:element name="startTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="timeZero" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="useColdState" type="xs:boolean"/>
    <xs:element name="workflowId" type="xs:string" minOccurs="0"/>
    <xs:element name="whatIfId" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

Runs a workflow task for a given **workflowId**. Returns a handle to the task in the form of a **taskId**. This taskId can be used to track the status of the workflow using method call **getTaskRunStatus**. Additionally there are a few arguments that can be passed to control the run.

- workflowId Identifier of the task to run.
- startTime Start of run period. Used for state selection period.
- timeZero Forecast time zero. If missing System time is used (optional)
- endTime End of run period. Used to define forecast length.
- coldStateId Id of a coldstate. Can be used to force state selection (optional).
- scenarioId Id of a predefined WhatIf scenario. Can be used to alter run parameters (optional).
- piParametersXml Contents of a [Pi ModelParameters XML](#) file. PI ModelParameters can be exported by the General Adapter to provide information to external models being run by FEWS (optional).
- useColdState Option to force the use of a cold state even when warm state is available.
- description Descriptive text to identify run (optional)
- return taskId of run.

getTaskRunStatus

```
String getTaskRunStatus(String taskId, long maxWaitMillis)
```

Runs a workflow task for a given **workflowId**. Returns a handle to the task in the form of a **taskId**. This taskId can be used to track the status of the workflow using method call **getTaskRunStatus**. Additionally there are a few arguments that can be passed to control the run.

- taskId Id returned by runTask.
- maxWaitMillis time in milliseconds to wait for response
- return task run status which can be one of the following values:

```
Valid values are
I = Invalid,
P = Pending,
T = Terminated,
R = running,
F = Failed,
C = Completed fully successful,
D = Completed partly successful,
A = Approved,
B = Approved partly successful
null = No status available (produces when method call times-out)
```

Installing a Tomcat Fews PI Service

N.B.: The DAC installation has become obsolete since FEWS 2017.02.

The file [DAC installation guide \(final\).doc](#) describes how to install a Tomcat instance containing a *DataAccessComponent* instance. Once these steps are completed the *FewsPiService.WAR* can be deployed using a context xml file such as [FewsPiService.xml](#).

Configuration

N.B.: The DAC configuration has become obsolete since FEWS 2017.02.

There are two levels of configuration. The first is the *DataAccessComponent* which is a FEWS client hosted as a *GlobalResource* in Tomcat. The *DataAccessComponent* uses the regular FEWS configuration which falls outside the scope of this documentation. The second level of configuration is what is used by the *FewsPiService* webservice which has been deployed in Tomcat. To deploy this webservice a context XML file is required which looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
Licensed to the Apache Software Foundation (ASF) under one or more
contributor license agreements. See the NOTICE file distributed with
this work for additional information regarding copyright ownership.
The ASF licenses this file to You under the Apache License, Version 2.0
(the "License"); you may not use this file except in compliance with
the License. You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
-->

<!-- Location of specific AI war file. (Can be a symlink) -->
<Context docBase="${catalina.home}/fews/FewsPiService.war" antiJARLocking="true" crossContext="true" >
    <!-- 'clientConfigFileId' points to a FEWS configuration file in the FEWS configuration directory
    'PiClientConfigFiles'. -->
    <Parameter name="clientConfigFileId" value="FewsPiService.properties" override="false"/>
    <!-- resourceId must match the 'name' of 'ResourceLink' below -->
    <Parameter name="resourceId" value="DAC" override="false"/>
    <Parameter name="serviceName" value="FewsPiServiceImpl" override="false"/>
    <Parameter name="namespaceUri" value="http://fewpiservice.wldelft.nl" override="false"/>
    <Parameter name="portName" value="FewsPiServiceImplPort" override="false"/>
    <!-- 'fewspi_config_naam' must match the file name of this configuration file without the XML extension. -->
    <Parameter name="wsdl" value="http://localhost:8080/FewsPiService/fewspiservice?wsdl" override="false"/>
    <!-- Global 'fews_dac' must be the same as the DAC resource name in the file 'server.xml' -->
    <ResourceLink name="DAC"
        global="fews_dac"
        type="nl.wldelft.fews.system.data.dac.DataAccessComponent"/>
    <Loader loaderClass="nl.wldelft.fews.system.data.dac.DacClassLoader" delegate="true"/>
</Context>
```

In the above configuration example you see a number of 'Parameter' values which will be explained below:

- *clientConfigFileId* = This is a reference to a FEWS configuration file. With this file it is possible to configure the webservice instance. We will discuss this file in more detail below. The field is optional. If omitted then the **defaultFilterId** must be configured in the **Filters.xml** FEWS configuration file.
- *resourceId* = Tells the webservice what Tomcat resource link to use.

The next parameters have been added for the FewsPiService test page. This is the page that appears when browsing to the root URI of the FewsPiService. Technically these parameters would not be required, but because it is a 'test' page, the configuration has not been optimized yet.

- *serviceName* = Name of the service
- *namespaceUri* = Namespace of the service
- *portName* = Name of the service port
- *wsdl* = URI of the service WSDL file

FEWS Client Config File (clientConfigFileId)

This is a simple text configuration file that is located in the FEWS configuration in the directory:

```
%REGION_HOME%/Config/PiClientConfigFiles/FewsPiService.properties
```

In the above example the **clientConfigFileId** would be **FewsPiService.properties**.

N.B.: Since FEWS 20170.2 the clientConfigFileId can no longer be configured and should always be called FewsPiService.properties.

For more information about the possible properties, see [FEWS Web Services Configuration FewsPiService.properties \(deprecated since 2022.02\)](#).

Example SOAP calls

See [soap-example-calls.zip](#) for examples SOAP client calls.