Reference manual

FICO® Xpress Optimization

FICO Application Studio/Optimization Executor Integration Package

Reference Manual

Release 8.3

Last update 14 April, 2016

www.fico.com | Make every decision count^{TM}

FICO® Xpress Optimization

Deliverable Version: A

Last Revised: 14 April, 2016

Version 8.3

Contents

CHAPTER 1
Introduction

The helper package *fssappstudio* provides a set of routines to transform input and result data of a Mosel model into the XML record format of FICO Application Studio. This package is imported by adding the following line to the top of your model:

```
uses "fssappstudio"
```

1.1 XML record format

The XML record format used by FICO Application Studio has the following structure: The top-level element is `InputRecordList` (for inputs) or `ResultRecordList` (for results), which always contains a single `InputRecord` or `ResultRecord`. Data comes in the form of scalars or arrays.

- Scalars are represented as values enclosed in a named element, as shown in the example below.
- Arrays are represented as a 'table' comprised of a named element which contains multiple 'rec' (`_rec`) elements that represent rows of the table. Each table row contains multiple elements representing individual cells. The following example has a table named 'Channels' with three columns: 'Channel', 'Cost', and 'Capacity'.

```
<?xml version="1.0"?>
<InputRecordList xmlns="http://www.fico.com/input"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <InputRecord>
    <MyScalarValue>7.5</MyScalarValue>
    <AnotherScalar>a string value</AnotherScalar>
    <Channels>
      <Channels_rec>
        <Channel>1</Channel>
        <Cost>0.5</Cost>
        <Capacity>100</Capacity>
      </Channels_rec>
      <Channels_rec>
        <Channel>2</Channel>
        <Cost>1.2</Cost>
        <Capacity>50</Capacity>
      </Channels_rec>
      <Channels_rec>
        <Channel>3</Channel>
        <Cost>3</Cost>
        <Capacity>40</Capacity>
      </Channels_rec>
    </Channels>
  </InputRecord>
</InputRecordList>
```

In a Mosel model, data input from such XML files needs to be preceeded by a call to `appsxmlimport` to register the inputs:

```
declarations
  myScalar: real
  ChannelCost,ChannelCapacity: array(CHANNELS: range) of real
end-declarations

appsxmlimport
initialisations from appsxmlin("MyScalarValue")
  myScalar as "[](MyScalarValue)"
end-initialisations
initialisations from appsxmlin("Channels")
  [ChannelCost,ChannelCapacity] as "[](Channel,Cost,Capacity)
end-initialisations
```

Inversely, data output from a Mosel model needs to be terminated with a call to `appsxmlexport` to generate the XML format data and the corresponding XSD schema definition from the output data:

```
declarations
  ResultCost,ResultCapacity: array(CHANNELS: range) of real
end-declarations

initialisations appsxmlout("ResultChannels")
  [ChannelCost,ChannelCapacity] as "[](ResultChannel,ResultCost,ResultCapacity)"
end-initialisations
appsxmlexport
```

Please note that array data needs to be specified in *sparse format*, that is, all indices are specified along with the value columns.

CHAPTER 2
Package functionality

2.1 Constants

The following publicly declared objects can be employed by Mosel models that load the package *fssappstudio*:

APPSXML_FMT_TYPED_RECORDLIST = 1
Normal XML format, use `InputRecordList/InputRecord` or `ResultRecordList/ResultRecord` elements

APPSXML_FMT_UNTYPED_RECORDLIST = 2
Untyped XML format, use `RecordList/Record` elements

2.2 Subroutines

`appsxmlexport`	Build XML from CVS files.	p. 4
`appsxmlfnin`	Generate a file name for an input file.	p. 5
`appsxmlfnout`	Generate a file name for an output file.	p. 6
`appsxmlimport`	Extract all data files included in provided XML.	p. 7
`appsxmlin`	Generate the extended filename for 'initialisations from' from a label.	p. 8
`appsxmlout`	Generate the extended filename for 'initialisations to' from a label.	p. 9

appsxmlexport

Purpose

Build XML from CVS files.

Synopsis

```
procedure appsxmlexport(fname:string, type:integer)
procedure appsxmlexport(fname:string)
procedure appsxmlexport
```

Arguments

`fname`	name of the XML output file, defaults to "result"
`type`	entity label; value

`APPSXML_FMT_TYPED_RECORDLIST`	use `ResultRecordList`/ `ResultRecord` elements (default)
`APPSXML_FMT_TYPED_RECORDLIST`	
`APPSXML_FMT_UNTYPED_RECORDLIST`	use `RecordList`/ `Record` elements

Example

See function `appsxmlout`.

Further information

1. This function transforms result data that has been generated by a Mosel model using `initializations to` with `appsxmlout` to the XML result data format of *fssappstudio*.
2. `appsxmlexport` also generates the XSD schema that corresponds to the resulting XML file.
3. FICO Application studio expects the fixed filename `result` for XML output data, using a single file per model.
4. The reverse operation, transforming XML format input data to CSV files is performed by `appsxmlimport`.

Related topics

`appsxmlout`, `appsxmlimport`

appsxmlfnin

Purpose

Generate a file name for an input file.

Synopsis

```
function appsxmlfnin(label:string):string
```

Argument

`label`	entity label

Return value

Filename or `null:` if the file has not been generated.

Example

The following line of Mosel code outputs the full path to a file called `myScalar` in a temporary directory created by this package.

```
writeln(appsxmlfnin("myScalar"))
```

Further information

This function returns the name of a temporary input file as is used by `appsxmlin`. It needs to be preceeded by a call to `appsxmlimport`.

Related topics

`appsxmlin`, `appsxmlimport`

appsxmlfnout

Purpose

Generate a file name for an output file.

Synopsis

```
function appsxmlfnout(label:string):string
```

Argument

`label`	entity label

Return value

Filename.

Example

The following line of Mosel code outputs the full path to a file called `myScalar` in a temporary directory created by this package.

```
writeln(appsxmlfnout("myScalar"))
```

Further information

This function generates and records the name of a temporary output file as is used by `appsxmlout`. If the file already exists it is deleted.

Related topics

`appsxmlout`

appsxmlimport

Purpose

Extract all data files included in provided XML.

Synopsis

```
procedure appsxmlimport(fname:string, type:integer)
procedure appsxmlimport(fname:string)
procedure appsxmlimport
```

Arguments

`fname`	name of an XML input file, defaults to "input"
`type`	data structure type; value

`APPSXML_FMT_TYPED_RECORDLIST`	use `InputRecordList`/ `InputRecord` elements (default)
`APPSXML_FMT_TYPED_RECORDLIST`	
`APPSXML_FMT_UNTYPED_RECORDLIST`	use `RecordList`/ `Record` elements

Example

See function `appsxmlin`.

Further information

1. This function prepares input data provided in *fssappstudio* XML format for reading them via `initializations from` using `appsxmlin` by transforming the XML data into a set of temporary CSV format files (one per entity).
2. `appsxmlimport` also generates the XSD schema that corresponds to the provided XML file.
3. FICO Application studio expects the fixed filename `input` for XML input data, using a single file per model.
4. The reverse operation, transforming CSV format data into the XML result data format expected by *fssappstudio* is performed by `appsxmlexport`.

Related topics

`appsxmlin`, `appsxmlexport`

appsxmlin

Purpose

Generate the extended filename for 'initialisations from' from a label.

Synopsis

```
function appsxmlin(label:string):string
```

Argument

`label`	entity label

Return value

An extended CSV filename

Example

The following lines of Mosel code

```
declarations
  myStr: string
end-declarations

appsxmlimport("input")
initialisations from appsxmlin("AnotherScalar")
  myStr as "[](AnotherScalar)"
end-initialisations
```

read the value of the scalar from an input file of this form:

```
<InputRecordList>
  <InputRecord>
    <MyScalarValue>7.5</MyScalarValue>
    <AnotherScalar>a string value</AnotherScalar>
  </InputRecord>
</InputRecordList>
```

Further information

1. This function generates the extended file name to be used by `initializations from` for the specified label.
2. Data input from the resulting file needs to be preceeded by a call to `appsxmlimport`.

Related topics

`appsxmlimport`

appsxmlout

Purpose

Generate the extended filename for 'initialisations to' from a label.

Synopsis

```
function appsxmlout(label:string):string
```

Argument

`label`	entity label

Return value

An extended CSV filename

Example

The following lines of Mosel code

```
declarations
  myArr: array(range) of real
end-declarations

myArr::[1..3][1.2, 3.5, 6.7]
initialisations to appsxmlout("AnArray")
  myArr as "[](ArIndex,ArValue)"
end-initialisations
appsxmlexport("result")
```

output the array `myArr` to a result file of this form:

```
<ResultRecordList>
  <ResultRecord>
    <AnArray>
      <AnArray_rec>
        <ArIndex>1</ArIndex>
        <ArValue>1.2</ArValue>
      </AnArray_rec>
      <AnArray_rec>
        <ArIndex>2</ArIndex>
        <ArValue>3.5</ArValue>
      </AnArray_rec>
      <AnArray_rec>
        <ArIndex>3</ArIndex>
        <ArValue>6.7</ArValue>
      </AnArray_rec>
    </AnArray>
  </ResultRecord>
</ResultRecordList>
```

Further information

1. This function generates the extended file name to be used by `initializations to` for the specified label.
2. Data output to the resulting file needs to be terminated by a call to `appsxmlexport` in order to generate the XML file.

Related topics

`appsxmlexport`

APPENDIX A

Contacting FICO

FICO provides clients with support and services for all our products. Refer to the following sections for more information.

Product support

FICO offers technical support and services ranging from self-help tools to direct assistance with a FICO technical support engineer. Support is available to all clients who have purchased a FICO product and have an active support or maintenance contract. You can find support contact information on the Product Support home page (www.fico.com/support).

On the Product Support home page, you can also register for credentials to log on to FICO Online Support, our web-based support tool to access Product Support 24x7 from anywhere in the world. Using FICO Online Support, you can enter cases online, track them through resolution, find articles in the FICO Knowledge Base, and query known issues.

Please include *'Xpress'* in the subject line of your support queries.

Product education

FICO Product Education is the principal provider of product training for our clients and partners. Product Education offers instructor-led classroom courses, web-based training, seminars, and training tools for both new user enablement and ongoing performance support. For additional information, visit the Product Education homepage at www.fico.com/en/product-training or email producteducation@fico.com.

Product documentation

FICO continually looks for new ways to improve and enhance the value of the products and services we provide. If you have comments or suggestions regarding how we can improve this documentation, let us know by sending your suggestions to techpubs@fico.com.

Sales and maintenance

USA, CANADA AND ALL AMERICAS

Email: XpressSalesUS@fico.com

WORLDWIDE

Email: XpressSalesUK@fico.com

Tel: +44 207 940 8718
Fax: +44 870 420 3601

Xpress Optimization, FICO
FICO House
International Square
Starley Way
Birmingham B37 7GN
UK

Related services

Strategy Consulting: Included in your contract with FICO may be a specified amount of consulting time to assist you in using FICO Optimization Modeler to meet your business needs. Additional consulting time can be arranged by contract.

Conferences and Seminars: FICO offers conferences and seminars on our products and services. For announcements concerning these events, go to www.fico.com or contact your FICO account representative.

About FICO

FICO (NYSE:FICO) delivers superior predictive analytics solutions that drive smarter decisions. The company's groundbreaking use of mathematics to predict consumer behavior has transformed entire industries and revolutionized the way risk is managed and products are marketed. FICO's innovative solutions include the FICO® Score—the standard measure of consumer credit risk in the United States—along with industry-leading solutions for managing credit accounts, identifying and minimizing the impact of fraud, and customizing consumer offers with pinpoint accuracy. Most of the world's top banks, as well as leading insurers, retailers, pharmaceutical companies, and government agencies, rely on FICO solutions to accelerate growth, control risk, boost profits, and meet regulatory and competitive demands. FICO also helps millions of individuals manage their personal credit health through www.myfico.com. Learn more at www.fico.com. FICO: Make every decision count^{TM}.

Index