

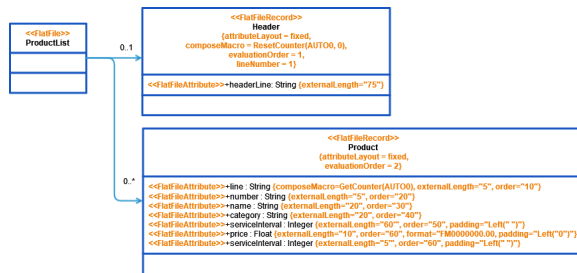
Defining a Flat File Data Structure

A flat file in the Designer is defined via the following structure:

1. **A flat file definition**
This is a class having the stereotype **FlatFile**. It represents the file itself. The properties of this class represent the data records.
2. **One or more flat file record definitions**
These are classes having the stereotype **FlatFileRecord**. They represent the data records of the flat file and are used as a type for the properties of the **FlatFile** class.
3. **One or more flat file record properties**
These are properties of the **FlatFileRecord** having the stereotype **FlatFileAttribute**. They represent the data fields of the flat file record.

Each flat file stereotype comes with a list of attributes to define certain functionality like length, format, order and more.

The following class diagram shows the flat file structure of the related example:



On this Page:

- [Defining the Flat File](#)
- [Defining the Flat File Record](#)
- [Defining Flat File Record Properties](#)
 - [Attribute Settings for Separated Flat File Record Layout](#)
 - [Attribute Settings for Fixed Flat File Record Layout](#)
- [The Flat File Example](#)

FlatFileAdapter_ProductExport_Example



Click the icon to download a simple example model that shows the usage of the Flat File adapter in **Scheer PAS Designer**.

Defining the Flat File

Define a flat file by creating a root class with stereotype **FlatFile**. The properties of this class must be complex and specified as to be [flat file records](#). A flat file can have multiple flat file record definitions representing different record types. In the example above these are **Header** and **Product**.



Assign a meaningful name because the output object flow of the Flat File adapter action uses this class as a type.

A flat file can have the following stereotype attributes specified:


Attribute	Description	Allowed Values				Example
recordSeparator	Separator of the different records, normally line feed and carriage return. For serialized files, any other character can be defined.	any character or one of		C Syntax	Character (Dec.)	<newline>
		<esc>		\x 1B	27	
		<newline>	Parses correctly on Unix and Windows platforms.	\n	10	
		<space>			32	
		<tab> <tabulator>		\t	9	
		<unixnewline>	Composes newline.	\n	10	
		<windowsnewline>	Composes newline and carriage return.	\r \n	13, 10	
escapeCharacter	Defines the character used for escaping when a reserved character is used within a field value.	any character				/
quoteCharacter	The quoteCharacter will be ignored by reading field value.	any character				/
fillCharacter	Defines a dummy character to fill non-existent values (results in NULL). Used for fixed property layout only.	any character				0
reservedCharacters	Defines a list of characters to be escaped automatically when the file is composed.	any character				{ " / " , " % " , " & " , " { " , " } " }
composeEmptyAttributes	Controls whether empty trailing attributes of data segments will be written during composition.	true/false				

Defining the Flat File Record

Define a flat file record by applying the stereotype **FlatFileRecord** to a class. Such classes can be used as types for properties of [flat file definitions](#) (**ProductList** in the example above).

Flat file records have [properties](#) that specify the structure and layout of the record. You can specify e.g. the sorting order of the properties within the record, the format of number properties, the record layout (separated, fixed, pattern) and more.

A flat file record can have the following stereotype attributes specified:

Attribute	Description	Allowed Values		Example
pattern	A pattern to identify the record. The pattern is checked before the fields are separated. If no pattern is defined, all records will be parsed.	a valid regular expression		^Pattern.*
attributeLayout	Defines the property layout of the flat file record.	fixed	Fixed property layout. For property values that are shorter than the maximum length of the field, the field is filled with a fill character. As per default, this is blank space, but you can change the fill character in attribute fillCharacter of the FlatFile class (see the Attributes of the FlatFile class above).	
		separated	Separated property layout. Specify the separator using attributeSeparator (see below).	
		pattern	Property layout is specified by a RegEx pattern in attributePattern (see below).	
attributePattern	A RegEx pattern to parse the record content into the properties using capture groups.	a valid regular expression		^(.?) (:/) ([A-Za-z0-9.]) ([0-9]) (/.)\$
attributeSeparator	Defines the property/field separator.	any character	Use the specified character as property separator.	
		comma (,)	Use the comma (,) as property separator.	
		<Tab>	Use tabulator as property separator.	
ignoreEmptyRecords	Boolean value for ignoring empty records. If set to true, no item will be generated, if none of the defined properties or sub records have any content. <div><div> Note, that a record containing only empty Strings is not empty – in opposition to a record composed from NULLS. See ignoreEmptyStrings below to skip processing of records containing only empty Strings.</div></div>	true	Ignore empty records.	
		false	Process empty records.	
suppressEscaping	Boolean value to suppress escaping. If suppressEscaping on a FlatFileRecord is true, FlatFileComplexAttribute that are part of this record will inherit this setting.	true	Property values of this record will not be un-escaped (parser) or escaped (composer) (default).	
		false	Escaping/un-escaping is not suppressed.	
composeMacro	A macro that is executed while parsing /composing a file or complex field.	any valid macro expression (see Using Macro Expressions on Parsing or Composing a Flat File)		GetCounter(AUTO0)

parseMacro	<p>This macro can contain multiple commands separated by commas or spaces. Macros on classes are executed before the processing of its properties or associations. The ID represents a counter.</p> <p>The following counters are available:</p> <ul style="list-style-type: none">• eight automatic counters with ID <code>AUTO0 .. AUTO7</code>• two automatic line counters with ID <code>LINE0</code> and <code>LINE1</code> (parsing only)• unlimited custom counters with ID <code>CUSTOM0 .. CUSTOMx</code> <p>Automatic counters are increased by 1 for each processed record. Custom counters have to be increased manually using the <code>increase</code> macro. All counters have the initial value of 0 when they process the first record.</p> <p>For more details on macro commands see Using Macro Expressions on Parsing or Composing a Flat File.</p>						
lineNumber	Specifies the number of a record in the file. The first record is <code>lineNumber=1</code> , the second <code>lineNumber=2</code> , etc.	any integer					
evaluationOrder	Defines the order in which the association of the classes starting on same parent class must be processed.	any integer					
ignoreEmptyStrings	Boolean value for ignoring empty string properties. If set to true , empty string values will be processed to <code>NULL</code> . Use this attribute in combination with ignoreEmptyRecords to skip processing of records containing only empty Strings .	<table><tr><td>true</td><td>Ignore empty string values.</td></tr><tr><td>false</td><td>Preserve empty string values.</td></tr></table>	true	Ignore empty string values.	false	Preserve empty string values.	
true	Ignore empty string values.						
false	Preserve empty string values.						

Defining Flat File Record Properties

Properties of a flat file record specify the structure and layout of the record. All properties need to have stereotype **FlatFileAttribute** applied. Depending on the layout type of the flat file (separated, fixed or pattern), you can specify different attributes.

Attribute Settings for Separated Flat File Record Layout

Whether a flat file has a separated layout is specified on the flat file record class in attribute **attributeLayout**. For records with a separated layout, you must specify a separator in attribute **attributeSeparator**. The default separator is `,` (comma).

The order of properties is defined on the property by either the **order** attribute or by specifying an **offset** within the record.

A flat file record property that is part of a can have the following stereotype attributes specified:

Attribute	Description	Allowed Values	Example
suppressEscaping	Boolean value to suppress escaping.		
parseMacro composeMacro	<p>A macro that is executed while parsing/composing a file or complex field.</p> <p>This macro can contain multiple commands separated by commas or spaces. Macros on classes are executed before the processing of its properties or associations. The ID represents a counter.</p> <p>The following counters are available:</p> <ul style="list-style-type: none"> • eight automatic counters with ID <code>AUTO0 .. AUTO7</code> • two automatic line counters with ID <code>LINE0</code> and <code>LINE1</code> (parsing only) • unlimited custom counters with ID <code>CUSTOM0 .. CUSTOMx</code> <p>Automatic counters are increased by 1 for each processed record. Custom counters have to be increased manually using the <code>increase</code> macro. All counters have the initial value of 0 when they process the first record.</p> <p>For more details on macro commands see Using Macro Expressions on Parsing or Composing a Flat File.</p>	any valid macro expression (see Using Macro Expressions on Parsing or Composing a Flat File)	<code>GetCounter(0)</code>

format	Pattern for formatting numeric and date & time values. For details see Number Formatting respectively Date and Time Formatting Patterns .		
order	The evaluation order of the properties. If offset is not used, order reflects the field number within the record.	any integer	
offset	The relative position of the field in respect of the other fields in the record, e.g. field number 3 has offset = 2.	any integer	

Attribute Settings for Fixed Flat File Record Layout


Whether a flat file has a fixed layout is specified on the flat file record class in attribute **attributeLayout**. With fixed layouts, each record property has a fixed length given by the **externalLength** attribute. Optionally, you can specify a padding to fill the attribute from the left or right side with a given character. The order of properties is defined on the property by either the **order** attribute or by specifying an **offset** within the record.

A flat file record property can have the following stereotype attributes specified:

Attribute	Description	Allowed Values		Example
suppressEscaping	Boolean value to suppress escaping.	true	Property values of this property will not be un-escaped (parser) or escaped (composer) (default).	
		false	Escaping/un-escaping is not suppressed.	
parseMacro	<p>A macro that is executed while parsing/composing a file or complex field.</p> <p>This macro can contain multiple commands separated by commas or spaces. Macros on classes are executed before the processing of its properties or associations. The ID represents a counter.</p> <p>The following counters are available:</p> <ul style="list-style-type: none"> • eight automatic counters with ID <code>AUTO0 .. AUTO7</code> • two automatic line counters with ID <code>LINE0</code> and <code>LINE1</code> (parsing only) • unlimited custom counters with ID <code>CUSTOM0 .. CUSTOMx</code> <p>Automatic counters are increased by 1 for each processed record. Custom counters have to be increased manually using the <code>increase</code> macro. All counters have the initial value of 0 when they process the first record.</p> <p>For more details on macro commands see Using Macro Expressions on Parsing or Composing a Flat File.</p>	any valid macro expression (see Using Macro Expressions on Parsing or Composing a Flat File)		GetCounter(0)
composeMacro				
padding	Defines the padding rule for the field from the left or right side.	<code>left ("<any character>")</code> <code>right ("<any character>")</code>	<ul style="list-style-type: none"> • Parsing: Ignore the specified character from the left/right side to the first different character. • Composing: Fill the property from the left /right side to the first different character using the specified character. 	<code>left("0")</code> <code>right(" ")</code>
format	Pattern for formatting numeric and date & time values. For details see Number Formatting respectively Date and Time Formatting Patterns .	any valid number or dateTime pattern		<code>S9G999G990D00</code> <code>%Y.%m.%d-%H:%M:%S</code>
order	The evaluation order of the properties. If offset is not used, order reflects the field number within the record.	any integer		
offset	The character position of this field within the record.	any integer		
externalLength	Number of characters of the field (only for fixed length records relevant).	any integer		

The Flat File Example

The flat file example service defines a flat file with a fixed layout. The following table explains the flat file definitions for this example:

Element		Stereotype	Stereotype Attribute		Description
Flat File	ProductList	FlatFile	-		Defines the flat file structure ProductList .
Flat File Record	Header	FlatFileRecord	attributeLayout	fixed	Specifies the header record as to be structured by fixed lengths.
			composeMacro	ResetCounter (AUTO 0)	Reset the automatic counter AUTO0 when this record is composed.
			evaluationOrder	1	Evaluate this record first, before evaluating the product records.
			lineNumber	1	This header record is the first record in the file.
	Product	FlatFileRecord	attributeLayout	fixed	Specifies the product record as to be structured by fixed lengths.
			evaluationOrder	2	Evaluate this record second, after evaluating the header record.
Flat File Property	line	FlatFileAttribute	composeMacro	GetCounter (AUTO 0)	This property is set from the automatic counter AUTO0.
			externalLength	5	This property has a length of 5 characters in the record.
			order	10	This property is the first property in the record (as all other properties have a higher order). <div> Use two-digit numbers for order. This way, the record layout can easily be updated, and you can easily fit an additional property somewhere in between.</div>
	number	FlatFileAttribute	externalLength	5	This property has a length of 5 characters in the record.
			order	20	This property is the second property in the record.
	name	FlatFileAttribute	externalLength	20	This property has a length of 20 characters in the record.
			order	30	This property is the third property in the record.
	category	FlatFileAttribute	externalLength	20	This property has a length of 20 characters in the record.
			order	40	This property is the fourth property in the record.
	type	FlatFileAttribute	externalLength	10	This property has a length of 10 characters in the record.
			order	50	This property is the fourth property in the record.
	price	FlatFileAttribute	externalLength	10	This property has a length of 10 characters in the record.
			order	60	This property is the sixth property in the record.

			format	FM0000000.00	The price should be formatted like specified, e.g. 0000065.50 .
			padding	Left("0")	The price should be filled with zeros from the left side, e.g. 0000065.50 .
	serviceInterval	FlatFileAttribute	externalLength	5	This property has a length of 5 characters in the record.
			order	70	This property is the seventh property in the record.
			padding	Left(" ")	The service interval should be filled with blanks from the left side, e.g. " 52 ".