webMethods EntireX

Internationalization with EntireX

Version 9.5 SP1

November 2013

This document applies to webMethods EntireX Version 9.5 SP1.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: EXX-INTERNAT-95SP1-20140628

Table of Contents

Preface v
1 Introduction to Internationalization 1
Overview 2
ICU Conversion 3
ICU Resources 5
Translation 7
Translation User Exit 9
Translation User Exit Replacement with ICU Conversion 10
SAGTRPC User Exit 11
Arabic Shaping 12
2 What is the Best Internationalization Approach to use? 13
Conversion Overview 14
Conversion Details 16
Codepage Requirements for RPC Data Stream Conversions 19
3 Locale String Mapping 21
Broker's Locale String Processing 22
Broker's Built-in Locale String Mapping 23
Broker's Locale String Defaults 26
Configuring Broker's Locale String Defaults 26
Bypassing Broker's Built-in Locale String Mapping 28
Using the Abstract Codepage Name LOCAL 29
4 Locale String Mapping Tables 31
Mapping Table Usage by Internationalization Approach 32
Mapping 2-character Language Codes 32
Mapping UNIX Codepage Names 34
Mapping Java Codepage Names 36
Mapping Codepage Numbers 37
5 Preparing EntireX Components for Internationalization 41
Rules and Defaults 42
Table of Components and Locale String Handling 43
Troubleshooting 46

Preface

Introduction	Software internationalization is the process of designing products and services so that they can be adapted easily to a variety of different local languages and cultures. Internationalization within EntireX means internationalization of messages: the incoming and outgoing messages are converted to the desired codepage of the platform in use.
Best Approach	Provides information to assist you in deciding which internationalization approach is the most appropriate.
Locale String Mapping	A locale provides a means of identifying a specific region for the purposes of internationalization and localization. EntireX sends properties of the operating system locale to the Broker, using the locale string.
Locale String Mapping Tables	Provides tables used during the process of mapping the locale string to codepages within the broker for the internationalization approaches ICU conversion and SAGTRPC user exit.
Preparing EntireX Components for Internationalization	How to prepare or configure EntireX components to force a locale string to be sent.

1 Introduction to Internationalization

- Overview 2
- ICU Conversion 3
- ICU Resources 5
- Translation 7
- Translation User Exit 9
- Translation User Exit Replacement with ICU Conversion 10
- SAGTRPC User Exit 11
- Arabic Shaping 12

This chapter provides an introduction to the topic of internationalization with EntireX and describes the various approaches offered.

Overview

The translation and conversion of codepages is a symmetric process. Everything that is valid for the request (client to server) relates also to the reply (server to client), with opposite roles. Therefore the terms sender and receiver are used instead of client and server in this section.

Internationalization with EntireX provides the following:

- Codepage conversion is available for senders and receivers, so any participant is able to work with the desired codepage. A participant tells the broker the codepage they use to send and receive messages. This means the broker is able to perform a conversion from/to the desired characters (code points) within the codepages.
- Codepage conversion deals with the user's payload data in broker's send and receive buffers. The broker ACI control block is handled differently and does not require special attention concerning internationalization. See *Broker ACI Fields* in the ACI Programming documentation.
- For the simpler approaches Translation and Translation User Exits, participants give the codepage to the broker implicitly - nothing needs to be configured for EntireX components (senders and receivers).
- For the more accurate approaches of ICU Conversion and SAGTRPC User Exit, the codepage that an EntireX component (sender and receiver) uses is described by so-called *locale strings* (alias name of a codepage) sent along with the request to the broker. The locale string always requires some attention. Depending on the platform your EntireX component is running on, the locale string is sent automatically by default or must be provided.
  - As long as you use your platform's default codepage and the locale string is provided automatically, nothing else needs to be considered.
  - If the locale string is not provided automatically, providing one can be a programming issue or a configuration issue, depending on the EntireX component used.
  - For information on how to provide locale strings, or whether locale strings are sent automatically, see table ***Preparing EntireX Components for Internationalization***.
- Codepage conversion is available for all of the communication models the broker offers, for example:
  - *ACI-based Programming* in its various language bindings (Java, C, Assembler, Natural, etc.).
  - *RPC-based Components* and *Reliable RPC* such as DCOM Wrapper, Java Wrapper, XML/SOAP Wrapper, Web Services Wrapper, COBOL Wrapper, PL/I Wrapper, .NET Wrapper, etc.
  - Publish and Subscribe.

The following sections discuss all of the internationalization approaches offered by EntireX.

ICU Conversion

- Introduction
- Using ICU Conversion
- Requirements for ICU Conversion
- ICU's Conversion Technique

Introduction

ICU conversion is based on IBM's project International Components for Unicode. It is a mature, widely used set of C/C++ and Java libraries for Unicode support, software internationalization and globalization.

ICU comes with a set of ICU converters (codepages) based on codepages from ISO and software vendors such as Microsoft and IBM. It is a standardized approach, and it is possible to extend the set with ICU custom converters.

Using ICU Conversion

You can use ICU conversion in the following situations:

- if you need multiple codepages per environment, for example more than one unique ASCII, IBM mainframe or Fujitsu mainframe codepage
- if you need single-byte, double-byte or multibyte conversions
- if you need standardized codepages and the ICU converters provided meet your requirements
- if you need *Arabic Shaping*

If you require special codepages that are not delivered, you can install user-written ***ICU Custom Converters***.

Requirements for ICU Conversion

For ICU conversion to function correctly, the following requirements must be met:

- The broker must be configured for the platform it is running on. See *Configuring ICU Conversion* under *Configuring Broker for Internationalization* in the platform-specific administration documentation. The service-specific or topic-specific broker attribute `CONVERSION` in the attribute file must be set:
  - for *ACI-based Programming*, use SAGTCHA for any type of codepage that has single-byte, double-byte and multibyte encoding schemes. See ***Conversion Details***.
  - for *RPC-based Components* and *Reliable RPC*, use SAGTRPC for any type of codepage that has single-byte, double-byte and multibyte encoding schemes. See ***Conversion Details***.

We recommend always using SAGTRPC for RPC data streams. ***Conversion with Multibyte, Double-Byte and other Complex Codepages*** will always be correct, and ***Conversion with Single-byte Codepages*** is also efficient because SAGTRPC detects single-byte codepages automatically. See ***Conversion Details***.

- EntireX components (sender and receiver) must send a locale string to the broker. Depending on the platform your EntireX component is running on, this is done automatically by default - nothing else needs to be configured as long as you use your platform's default codepage. If the locale string is not provided automatically, it can be set as a programming issue or a configuration issue, depending on the EntireX component used. See ***Preparing EntireX Components for Internationalization***.
- The locale string sent by EntireX components (sender and receiver), the encoding of the ACI payload or RPC stream, and the ICU converter (codepage) must always match, otherwise unpredictable results occur. Checking for the availability of the correct ICU converters (codepages) is mandatory.

ICU's Conversion Technique

ICU uses algorithmic conversion, non-algorithmic conversion and combinations of both. With non-algorithmic conversion, tables are provided that contain a mapping of codepage characters to Unicode as a definition of a codepage. This format is also called ***UCM Format***.

ICU conversion is a two-step process:

1. The conversion table designated by the sender is used to convert from characters of the source codepage to Unicode.
2. The conversion table designated by the receiver in the reverse direction is used to convert from Unicode to characters of the target codepage.

ICU uses line-oriented text files to define non-algorithmic converters. For complex codepages, partially or fully algorithmic converters may be used, which cannot be defined as simple text files.

ICU Resources

Please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". This document is part of the product documentation, located at **http://documentation.software-ag.com/legal/**.

This section covers the following topics:

- ICU Homepage
- ICU Converter Explorer
- ICU Converter Resources
- ICU Custom Converters
- UCM Format

ICU Homepage

The ICU home page (**http://www.icu-project.org/**) is the main point of entry for information on International Components for Unicode (ICU).

ICU Converter Explorer

The ICU Converter Explorer available at ***http://demo.icu-project.org/icu-bin/convexp*** shows aliases and more information on ICU converters. An ICU converter is the codepage definition used by ICU. The ICU converter is defined by a so-called UCM format. If the location has changed since this documentation was published, perform an internet search for the ICU home page and follow the links to the ICU Converter Explorer.

The mapping of aliases to ICU converters is also provided as a text source within an EntireX installation. The location depends on the operating system:

- UNIX: */opt/softwareag/EntireX/etc/convrtrs.txt*
- Windows: *<drive>:\SoftwareAG\EntireX\etc\convrtrs.txt*

ICU Converter Resources

EntireX includes a standard set of the most commonly used ICU converters (codepages) in binary format packed into shared libraries.

ICU Custom Converters

If the provided standard ICU converters (codepages) do not match your requirements, the ICU codepages can be extended by user-written ICU custom converters. This is done with the ICU tool `makeconv` delivered with EntireX. With `makeconv`, ICU converter files in ***UCM Format*** are compiled into a binary format with extension `cnv`. The binary format `cnv` depends on the endianness (big/little endian) and charset family (ASCII/EBCDIC) where `makeconv` is executed. See *Building and Installing ICU Custom Converters* in the platform-specific administration documentation.

UCM Format

The codepage definition text files for ICU are described in UCM format (extension ".ucm"). You can edit them with any text editor. The most important section is the mapping table between the CHARMAP and END CHARMAP lines. Each line contains a Unicode code point and the related codepage character byte sequence followed by an optional precision indicator. Four kinds of definitions are supported by the precision indicator:

- 0 - normal roundtrip mapping from a Unicode code point and back.
- 1 - fallback mappings are used during conversion from Unicode to the codepage, but not back again. This definition may be present if a character exists in Unicode but not in the codepage. This feature is useful for human-readable output where the missing character is mapped to a similar looking one.
- 2 - substitution mappings resulting in assignment of the alternative substitution sequence (subchar1 in UCM format) when a non-convertible character occurs, instead of assigning the default substitution sequence (subchar in UCM format).
- 3 - reverse fallback mappings are used during conversion from the codepage to Unicode, but not back again. This definition results in assigning the same Unicode code point for different codepage character byte sequences.

This brief explanation does not intend to describe the UCM file format fully. For further explanation of the UCM file format, see the ICU home page under ***ICU Resources*** above.

Translation

Introduction

Translation is the quick-start approach with little configuration required, only service-specific or topic-specific broker attribute `TRANSLATION` in the broker attribute file has to be set to the value SAGTCHA. Nothing needs to be configured or considered for the EntireX component (sender or receiver). Translation does not need locale strings. If translation is specified and an EntireX component sends a locale string, the locale string will be ignored.

Translation has limitations on the number of environments supported and the number of different codepages for the environment in which your EntireX components (sender or receiver) are running:

- all ASCII environments (UNIX, Windows, etc.) must use the same ASCII codepage
- all IBM mainframes must use the same EBCDIC codepage
- all Fujitsu mainframes must use the same EBCDIC codepage

Translation Codepages

Translation has further limitations on the code points used within the codepages provided. The translation routine SAGTCHA is loosely based on the following platform-dependent codepages:

Environment	Indicator sent from EntireX Component to Broker	Based on Codepage	Description
All ASCII environments (UNIX, Windows etc.)	x'80'	Microsoft Windows codepage 1252	Translation of characters for ASCII environments is loosely based on Windows codepage 1252. Not all of the characters of Windows codepage 1252 are supported by translation. All of the characters supported have the same code point in codepage ISO 8859-1, thus this is also suitable for UNIX.
IBM mainframe	x'22'	IBM codepage 273	Translation of characters for the IBM mainframe platform is loosely based on IBM codepage 273. Not all of the characters of the IBM codepage 273 are supported by translation.
Fujitsu mainframe	x'42'	EDF 03 national version for Germany	Translation of characters is loosely based on the EDF03 codepage for Germany.

Characters (code points) supported by SAGTCHA are the same as in the Translation user exit example. See *Writing Translation User Exits* under *Configuring Broker for Internationalization* in the platform-specific administration documentation.

Using Translation

You can use translation in the following situations:

- if you have a mixed environment consisting of ASCII, IBM mainframes and/or Fujitsu mainframe platforms but
  - all of your ASCII Environments use the same codepage
  - all of your IBM mainframes use the same codepage
  - all of your Fujitsu mainframes use the same codepage
- if single-byte codepages meet your requirements.
- if the code points within the delivered codepages meet your requirements. Please note that not all code points implemented by SAGTCHA are roundtrip-compatible even if in your environment the Microsoft Windows codepage 1252, IBM codepage 273 and EDF 03 national version for Germany are used. Roundtrip incompatibility means that if you transfer a character from an ASCII platform to IBM EBCDIC or Fujitsu EBCDIC and back again you will get a different character. Important code points (characters) such as uppercase letters A-Z, lowercase letters a-z, digits 0-9 and the characters listed in ***Codepage Requirements for RPC Data Stream Conversions*** and also others are roundtrip-compatible.
- for *RPC-based Components*, *Reliable RPC* as well as *ACI-based Programming* and other communication models.

See *Configuring Translation* in the platform-specific administration documentation.

Translation User Exit

Introduction

With translation user exits, the code points of the codepage used are under your control. You can adapt them to meet your requirements. This requires programming a user-specific translation routine. See *Writing Translation User Exits* under *Configuring Broker for Internationalization* in the platform-specific administration documentation. The delivered model for the translation user exit supports single-byte codepages only, but in principle any type of codepage can be implemented.

With translation user exits, you can make any structure of the data (mixture of text and binary data) within your payload known to the user exit by means of the ACI field `ENVIRONMENT`, which can be shared between your application and the translation user exit. For more information, see *Using the `ENVIRONMENT` Field with the Translation User Exit* under *Writing Applications: Client and Server | Publish and Subscribe* in the ACI Programming documentation.

Configuration effort is easy, only service-specific or topic-specific broker attribute `TRANSLATION` in the broker attribute file has to be set to the name of your user exit. Nothing needs to be configured or considered for the EntireX component (sender or receiver). Translation does not need locale strings. If a translation user exit is specified and an EntireX component sends a locale string, the locale string will be ignored.

The limitations on the number of environments and different codepages per environment remain the same as for translation.

Using Translation User Exit

You can use a translation user exit in the following situations:

- if you want to invent your own conversion package
- if you have to consider any payload data structure for translation
- if it is necessary to share data between the application (client/server or publisher/subscriber) and the translation routine by using the broker ACI field `ENVIRONMENT`
- if you have a mixed environment consisting of ASCII, IBM mainframes and/or Fujitsu mainframe platforms, but
  - all of your ASCII environments use the same codepage
  - all of your IBM mainframes use the same codepage
  - all of your Fujitsu mainframes use the same codepage
- for *ACI-based Programming*, if single-byte codepages meet your requirements. Otherwise you will have to invent a model for other types of codepages such as multibyte, double-byte and EBCDIC stateful - this can become very complicated and involve considerable effort.

- for *RPC-based Components*, if single-byte codepages meet your requirements *only*. The codepage you implement must meet the ***Codepage Requirements for RPC Data Stream Conversions***.

Translation User Exit Replacement with ICU Conversion

If a ***Translation User Exit*** is used to adapt code points only, that is, to implement a standard ASCII/EBCDIC codepage, the same functionality can be achieved with ICU conversion, simply by using ***Broker's Locale String Defaults***, well configured, and service-specific or topic-specific broker attribute `CONVERSION OPTION=SUBSTITUTE` set for the same error behavior as translation. See *`OPTION` Values for Conversion*.

Example

For an environment running in Spain using clients with the Windows 1252 codepage and servers on IBM mainframe with codepage 1145, set the following *Codepage-specific Attributes*:

```
DEFAULTS=CODEPAGE
            /* Broker Locale String defaults */
            DEFAULT_ASCII=windows-1252
            DEFAULT_EBCDIC_IBM=ibm-1145
```

For *ACI-based Programming*, set the service-specific or topic-specific broker attribute `CONVERSION`:

```
DEFAULTS=SERVICE
            . . .
            CONVERSION=(SAGTCHA,OPTION=SUBSTITUTE)
            . . .
```

For *RPC-based Components* and *Reliable RPC*, set the service-specific or topic-specific broker attribute `CONVERSION`

```
DEFAULTS=SERVICE
            . . .
            CONVERSION=(SAGTRPC,OPTION=SUBSTITUTE)
            . . .
```

For more examples see ***Configuring Broker's Locale String Defaults***.

SAGTRPC User Exit

Introduction

With the SAGTRPC user exit you can invent your own conversion package/method for *RPC-based Components* and *Reliable RPC* if for any reason a codepage is not supported by ***ICU Conversion*** and SAGTRPC conversion. SAGTRPC user exit cannot be used for *ACI-based Programming*.

SAGTRPC user exit allows you to adapt codepages and their characters (code points) to meet your requirements. This requires some effort in programming a SAGTRPC user exit. See *Writing SAGTRPC User Exits* in the platform-specific administration documentation. The delivered model for the SAGTRPC user exit supports single-byte codepages only, but in principle any type of codepage can be implemented.

Using SAGTRPC User Exit

You can use SAGTRPC user exit in the following situations:

- if you want to invent your own conversion package for RPC-based data stream conversion
- if you require different types of conversions depending on individual IDL field types
- for *RPC-based Components* and *Reliable RPC only*; it cannot be used for ACI-based programming.
- if the codepages you implement meet the ***Codepage Requirements for RPC Data Stream Conversions***

Requirements for SAGTRPC User Exit

For SAGTRPC user exit to function correctly, the following requirements must be met:

- The broker must be configured for the platform it is running on. See *Configuring SAGTRPC User Exits* under *Configuring Broker for Internationalization* in the platform-specific administration documentation. The `CONVERSION` in the broker attribute file must be set to the name of your routine.
- Locale strings may be provided. It depends on your implementation of the SAGTRPC user exit whether the components (sender and receiver) have to send a locale string to the broker or not. See ***Preparing EntireX Components for Internationalization***.
- The handling of the different IDL type fields depends on the implementation of the SAGTRPC user exit, which is the customer's responsibility. See *Writing SAGTRPC User Exits* in the platform-specific administration documentation.

Arabic Shaping

Arabic shaping is part of ***ICU Conversion*** and is available between UTF-8, the Arabic ASCII codepage windows-1256 and the Arabic EBCDIC codepage IBM-420 for all of the communication models EntireX Broker offers, for example:

- *ACI-based Programming* in its various language bindings (Java, C, Assembler, Natural, etc.)
- *RPC-based Components* and *Reliable RPC*, such as DCOM Wrapper, Java Wrapper, XML/SOAP Wrapper, Web Services Wrapper, COBOL Wrapper, PL/I Wrapper, .NET Wrapper etc.
- Publish and Subscribe.

Shaping is performed only on the codepages listed above. See also ***Conversion with Multibyte, Double-Byte and other Complex Codepages***.

2 What is the Best Internationalization Approach to use?

- Conversion Overview 14
- Conversion Details 16
- Codepage Requirements for RPC Data Stream Conversions 19

It is assumed that you have read the chapter *Introduction to Internationalization* and are familiar with the various internationalization approaches described there.

This chapter provides information to help you decide which internationalization approach is the most appropriate.

Conversion Overview

This table gives an overview of the internationalization approaches that can be used. The approach you choose depends on

- ACI or RPC payload
- the type of codepage used by participants (client and server): single-byte or complex codepage configuration [(1)], for example multibyte, double-byte, EBCDIC stateful codepages, Arabic shaping etc.

Internationalization Approach	Using Locale Strings	All components use single-byte codepages		One component uses a complex codepage configuration [1]		Usage Hint
		ACI	RPC [2]	ACI	RPC [2]	
ICU Conversion	yes [3,5]	yes	yes	yes	yes	ICU conversion is recommended. In the Broker attribute file, set the service-specific or topic-specific broker attribute `CONVERSION`: ■ for *ACI-based Programming* to `CONVERSION=SAGTCHA` ■ for *RPC-based Components* and *Reliable RPC* to `CONVERSION=SAGTRPC` We recommend always using SAGTRPC for RPC data streams. ***Conversion with Multibyte, Double-Byte and other Complex Codepages*** will always be correct, and ***Conversion with Single-byte Codepages*** is also efficient because SAGTRPC detects single-byte codepages automatically. See ***Conversion Details***. See also *Configuring ICU Conversion* under *Configuring Broker for Internationalization* in the platform-specific administration documentation under *Configuring Broker for Internationalization* in the platform-specific administration documentation.
Translation	no	yes	yes	no	no	Translation is not recommended for the following reasons: ■ limited support of code points for ASCII, IBM EBCDIC and Fujitsu EBCDIC ■ code points are not 100% compatible with standardized ASCII or EBCDIC codepages, which means that some code points are not roundtrip-compatible Consider instead using ICU conversion, see first row in this table.
Translation User Exit	no	yes	yes	yes	no	Translation User Exit is not recommended. If you only wish to adapt code points, it is too much effort. We recommend you use ICU conversion instead. See ***Translation User Exit Replacement with ICU Conversion***.
SAGTRPC User Exit	optional [4,5]	no	yes	no	yes	Requires considerable effort for implementation. See ***Conversion Details***. Consider instead using ICU conversion. See first row in this table.

Notes:

1. A complex codepage configuration is in effect where one participant (client or server) of a communication uses a codepage listed in ***Conversion with Multibyte, Double-Byte and other Complex Codepages***.
2. All codepages used for *RPC-based Components* and *Reliable RPC* must meet the ***Codepage Requirements for RPC Data Stream Conversions***.
3. The locale string (codepage)
   - must follow the rules described under ***Locale String Mapping***
   - must be a codepage supported by the broker
   - must be the codepage used in your environment, otherwise unpredictable results may occur.
4. It depends on the implementation of the ***SAGTRPC User Exit*** whether locale strings (codepages) are used. See *Character Set and Codepage* under *Configuring Broker for Internationalization* in the platform-specific administration documentation in section *Configuring Broker for Internationalization*. If they are used, they must follow the rules described under ***Locale String Mapping***.
5. If the participant (client or server) does not send a codepage (locale string) you can optionally
   - set the *Codepage-specific Attributes* (`DEFAULTS=CODEPAGE`) under *Broker Attributes* in the administration documentation to meet your requirements, or
   - configure the participant (client or server). See ***Preparing EntireX Components for Internationalization***.

Conversion Details

- Conversion with Single-byte Codepages
- Conversion with Multibyte, Double-Byte and other Complex Codepages

Conversion with Single-byte Codepages

This table gives an overview of the conversion effort if two participants (client and server) of a communication use single-byte codepages only. It is valid for ICU conversion. For RPC, SAGTRPC detects single-byte codepages automatically and converts them efficiently in one step (a single ICU call) from source to target encoding. This is the same as SAGTCHA for ACI. The same applies if you have invented your own internationalization approach with ***Translation User Exit***.

The effort does not depend on ACI or RPC payload - there is no difference. If one participant (client or server) uses a complex codepage configuration, the information given here does not apply; see ***Conversion with Multibyte, Double-Byte and other Complex Codepages*** instead.

To find out if a codepage is single-byte, see ***ICU Resources***.

Codepage Configuration	ACI [1]	RPC [2][3]
Single-byte codepages	Conversion is fast and efficient in one step.	Conversion is fast and efficient in one step.

Notes:

1. *ACI-based Programming*: in the Broker attribute file, the service-specific or topic-specific broker attribute `CONVERSION` is set to `CONVERSION=SAGTCHA`.
2. *RPC-based Components* and *Reliable RPC*: in the Broker attribute file, the service-specific broker attribute `CONVERSION` is set to `CONVERSION=SAGTRPC`.
3. All codepages used for *RPC-based Components* and *Reliable RPC* must meet the ***Codepage Requirements for RPC Data Stream Conversions***.

Conversion with Multibyte, Double-Byte and other Complex Codepages

This table gives an overview on the conversion effort if one participant (client or server) of a communication use a multi-byte, doublebyte or other complex codepage configuration (see the table), including Arabic shaping. It applies to ICU conversion. For RPC, SAGTRPC detects complex codepage configurations automatically and converts them as described (see column RPC) from source to target encoding. If you have invented your own internationalization approach with

- ***Translation User Exit*** for ACI, consider the rules in column ACI
- ***SAGTRPC User Exit*** for RPC, consider the rules in column RPC

depending on codepage type.

If two participants (client and server) of a communication use single-byte codepages only, see ***Conversion with Single-byte Codepages***. With a complex codepage configuration, the effort depends on:

- ACI or RPC payload
- the type of codepage used: multi-byte, doublebyte or EBCDIC stateful, etc.
- whether Arabic shaping is required

To find out if a codepage is multibyte, double-byte or EBCDIC stateful, see ***ICU Resources***.

Codepage Configuration	ACI [1]	RPC [2][3]
Multibyte or double-byte codepages	There is no additional effort compared to ***Conversion with Single-byte Codepages***. Conversion is performed in one step, the same as with single-byte codepages. Please note the payload may	If at least one participant (client or server) uses a multibyte or double-byte codepage with RPC, each IDL parameter (see `simple-parameter-definition` under *Software AG IDL Grammar* in the *IDL Editor* documentation) must be converted separately. The data in IDL type A, AV, K and KV and RPC metadata may increase or decrease after

Codepage Configuration	ACI [1]	RPC [2][3]
	change its length in bytes during conversion.	conversion from the sender's source codepage to the receiver's target codepage. The following must be honored: ■ *increasing* or *decreasing* data within IDL type AV and KV (without maximum) and RPC metadata (such as user ID, IDL library and IDL program). ■ *increasing* or *decreasing* data within IDL type A and K and AV, KV (with maximum) in its IDL defined field length boundaries. Data must be truncated if the field boundaries are crossed for increase - otherwise the RPC data stream is destroyed and unpredictable errors occur. If the data decreases, fields are padded at the end with blanks. All other IDL data types are converted as with single-byte code pages.
EBCDIC stateful codepages, encoded with escape technique (SI/SO bytes)	There is no additional effort compared to ***Conversion with Single-byte Codepages***. Conversion is performed in one step, the same as with single-byte codepages. Please note the payload may change its length in bytes during conversion. There is no special handling for SI/SO bytes as with RPC.	If at least one participant (client or server) uses an EBCDIC stateful codepage with RPC, each IDL parameter (see `simple-parameter-definition` under *Software AG IDL Grammar* in the *IDL Editor* documentation) must be converted separately. Also, the IDL types K and KV allow you to transfer double-byte data without SO and SI escape characters. This feature is designed for use in Asian countries. The disadvantage is that IDL fields must be converted field-by-field. To convert the fields correctly, RPC programmers have to consider the following rules, otherwise unpredictable results may occur: ■ SO and SI escape characters may not be contained in IDL type K and KV ■ double-byte characters are allowed in IDL type K and KV only ■ single-byte characters cannot be transferred in IDL type K and KV All other IDL data types are converted as with single-byte code pages.
Hebrew CP803 [4]	There is no additional effort compared to ***Conversion with Single-byte Codepages***. Conversion is performed in one step, the same as with single-byte codepages. Latin lowercase characters cannot be used and lead to conversion errors. See *`OPTION` Values for*	If at least one participant (client or server) uses the Hebrew codepage CP803, each IDL parameter (see `simple-parameter-definition` under *Software AG IDL Grammar* in the *IDL Editor* documentation) must be converted separately, because CP803 does not include Latin lowercase characters [3]. Please note the following: ■ All IDL types can be used.

Codepage Configuration	ACI [1]	RPC [2][3]
	Conversion to tune error behavior to meet your requirements.	■ Latin lowercase characters cannot be used within IDL type A and AV. ■ IDL program and IDL library cannot contain Latin lowercase characters, but Hebrew characters are OK. ■ RPC error text, PING replies etc. are converted to uppercase before conversion to CP803. This makes such texts readable at both ends (clients and server).
Arabic shaping [5]	The additional effort compared to ***Conversion with Single-byte Codepages***. The conversion itself is performed in one step, the same as with single-byte codepages. Shaping is performed on the complete ACI payload.	If Arabic shaping is required, each IDL parameter (see `simple-parameter-definition` under *Software AG IDL Grammar* in the *IDL Editor* documentation) must be converted separately. Shaping is performed on IDL data types A, AV, K and KV. All other IDL data types are converted as with single-byte code pages.

Notes:

1. *ACI-based Programming*: in the Broker attribute file, the service-specific or topic-specific broker attribute `CONVERSION` is set to `CONVERSION=SAGTCHA`.
2. *RPC-based Components* and *Reliable RPC*: in the Broker attribute file, the service-specific broker attribute `CONVERSION` is set to `CONVERSION=SAGTRPC`.
3. All codepages used for *RPC-based Components* and *Reliable RPC* must meet the ***Codepage Requirements for RPC Data Stream Conversions***.
4. The Hebrew CP 803 does not contain Latin lowercase characters and does not meet the ***Codepage Requirements for RPC Data Stream Conversions***. Despite this non-compliance, it can still be used for RPC.
5. Arabic shaping is in effect if all participants (client and server) use one of the following codepages: UTF-8, windows-1256 or ibm-420 codepage.

Codepage Requirements for RPC Data Stream Conversions

Codepages used to convert RPC data streams must meet several requirements:

1. Codepages used to convert RPC data streams must have the following code points (characters) defined:

Character	also known as	Rendered	Unicode Code Point
uppercase letters A-Z without special characters		A - Z	0x0041 to 0x005A
lowercase letters a-z without special characters		a - z	0x0061 to 0x007A
digits		0-9	0x0030 to 0x0039
SPACE		" "	0x0020
LEFT PARENTHESIS	OPENING PARENTHESIS	"("	0x0028
RIGHT PARENTHESIS	CLOSING PARENTHESIS	")"	0x0029
PLUS SIGN		"+"	0x002B
HYPHEN	MINUS	"-"	0x002D
SOLIDUS	SLASH	"/"	0x002F
COLON		":"	0x003A
COMMA		","	0x002C
FULL STOP	PERIOD	"."	0x002E
EQUALS SIGN		"="	0x003D

2. All code points (characters) listed in the table above must have a unique mapping (without any fallbacks and reverse fallbacks) to/from Unicode, that is, they must be roundtrip-compatible.
3. If the codepage used is a multibyte or double-byte codepage, the code points (characters) listed in the table above must have a length of 1 byte within the codepage. Therefore UTF-16 encoding cannot be used, but UTF-8 encoding is possible.

Codepages that do not obey the rules above cannot be used for RPC-based components, because those code points (characters) are used to code for example the IDL library and IDL program, descriptive metadata and IDL type fields in numeric, integer and binary form.

3 Locale String Mapping

- Broker's Locale String Processing 22
- Broker's Built-in Locale String Mapping 23
- Broker's Locale String Defaults 26
- Configuring Broker's Locale String Defaults 26
- Bypassing Broker's Built-in Locale String Mapping 28
- Using the Abstract Codepage Name LOCAL 29

It is assumed that you have read the chapter *Introduction to Internationalization* and are familiar with the various internationalization approaches described there.

A locale provides a means of identifying a specific region for the purposes of internationalization and localization. EntireX sends properties of the operating system locale to the Broker, using the locale string.

This chapter describes the mapping of the locale string to codepages within the broker for the internationalization approaches ICU conversion and SAGTRPC user exit. It does not apply to other approaches.

Broker's Locale String Processing

Depending on the internationalization approach in effect for a service or topic, the result of the locale string processing within the broker is either the ICU converter alias or the codepage number given to SAGTRPC user exit.

There are always two EntireX components involved - client or server, sender or receiver - so for genuine conversion this process is run through for the sender to determine its codepage as well as for the receiver to get the codepage. It is important to know both codepages in order to predict conversion behavior accurately.

1. If no locale string is provided by an EntireX component (sender or receiver), the *Broker's Locale String Defaults* will apply. These can be customized in the broker attribute file; see *Configuring Broker's Locale String Defaults*.
2. If a locale string is provided by an EntireX component (sender or receiver), the broker first refers to the codepage section of the attribute file searching for a keyword entry identical to the locale string sent. You can also bypass these entries to fit your needs; see *Bypassing Broker's Built-in Locale String Mapping*.
3. If an entry is found in step 2, this codepage is used directly and no further mapping occurs.
4. If no entry is found in step 2, the *Broker's Built-in Locale String Mapping* is entered.

A *prerequisite* for the broker to use internationalization correctly is a suitably configured environment. This includes the broker's platform and may include the platforms that other EntireX components (sender and receiver) run on.

- The data that EntireX components send to the broker *must* be in the encoding described by the locale string.
- The data that EntireX components send to the broker *must* be in the encoding of the codepage the broker uses for conversion or translation.
- After broker's locale string processing (steps 1-4 above), the resulting ICU converter or the code points implemented with the Translation user exit or with the SAGTRPC user exit *must* match

the defined code points of the original codepage of your application's environment. (Matching code points is not a trivial matter. Almost every hardware and software vendor provides its own codepage, based on standards organizations such as ISO etc. It is even more difficult to find matching codepages because there is no unique scheme of identifiers for codepages across all organizations.)

If one of the prerequisites above is not met, results will be unpredictable.

Broker's Built-in Locale String Mapping

The table in this section describes the built-in mechanism the broker uses to map a locale string to a codepage.

Whenever possible, we recommend using the abstract codepage name "LOCAL". Without adapting your EntireX component, this setting allows you to

- reconfigure your system codepage
- install your EntireX application in systems configured for different countries and regions using different codepages

See *Using the Abstract Codepage Name LOCAL*.

The form ECS `<ecs-number>` is deprecated and should no longer be used.

The last 2 forms in the table below CP `<number>` and `<codepage-name>` are the forms administrators and programmers use to configure or provide a codepage manually. This is necessary if

- a locale string is not sent by default to the broker, and
- the abstract codepage name LOCAL cannot be used.

Depending on the format of the locale string sent by the EntireX component, various rules for mapping apply:

Locale String Sent to Broker	Environment	Locale String Format Sent	ICU Converter Alias	SAGTRPC User Exit Codepage Number
<*language*>_<*country*>.<*number*> or <*language*>-<*country*>.<*number*> where any abbreviation is accepted for <*language*> and <*country*> and <*number*> is the decimal number of a Windows codepage.	Windows	■ *by default* if communicating with broker version 7.2.x and above ■ if communicating with broker version 7.1.x and below and the ***Using the Abstract Codepage Name LOCAL*** is given. ■ if manually configured or provided by programmer.	The part <*number*> is extracted from the locale string and prefixed by the term windows. The result is used as the ICU converter alias. The parts <*language*> and <*country*> are not used. Example: `german_Germany.1252` results in the alias `windows-1252` for the ICU converter alias.	The part <*number*> is extracted and used together with the table ***Mapping Codepage Numbers*** to evaluate the number given to the SAGTRPC user exit. The parts <*language*> and <*country*> are not used.
<*ll*>_<*cc*> where <*ll*> is the 2-letter language abbreviation and <*cc*> the 2 or 3-letter language code according to ISO 639 standard.	UNIX	■ if the***Using the Abstract Codepage Name LOCAL*** is given.	The part <*ll*> is extracted and used together with the table ***Mapping 2-character Language Codes*** to evaluate the ECS codepage number. The part <*cc*> is not used.	The part <*ll*> is extracted and used together with the table ***Mapping 2-character Language Codes*** to evaluate the number given to the SAGTRPC user exit. The part <*cc*> is not used.
<*ll*>_<*cc*>.<*UNIX-codepage-name*> where <*ll*> is the 2-letter language abbreviation, <*cc*> the 2 or 3-letter language code according to ISO 639 standard and <*UNIX-codepage-name*> is any alias name for a codepage.	UNIX	■ if the ***Using the Abstract Codepage Name LOCAL*** is given.	The part <*UNIX-codepage-name*> is extracted from the locale string and used for the ICU converter alias. The parts <*ll*> and <*cc*> are not used.	The part <*UNIX-codepage-name*> is extracted and used together with the table ***Mapping UNIX Codepage Names*** to evaluate the number given to the SAGTRPC user exit. The parts <*ll*> and <*cc*> are not used.
CP <*number*> where <*number*> is the decimal number of a codepage.	All	■ *by default* in Java environments if	The locale string CP <*number*> is used as is for the ICU Converter alias.	The part <*number*> is extracted and used together with the table ***Mapping Codepage***

Locale String Sent to Broker	Environment	Locale String Format Sent	ICU Converter Alias	SAGTRPC User Exit Codepage Number
		communicating with broker version 7.2.x and above ■ if manually configured or provided by a programmer		*Numbers* to evaluate the number given to the SAGTRPC user exit.
<codepage-name> where <codepage-name> is any alias name for a codepage.	All	■ *by default* in Java environments if communicating with broker version 7.2.x and above ■ if manually configured or provided by a programmer	The locale string <codepage-name> is used as is for the ICU Converter alias.	The locale string <codepage-name> is used together with the table *Mapping Java Codepage Names* to evaluate the number given to the SAGTRPC user exit.

With ICU Conversion

- Once you have determined the ICU converter alias, use the ICU Converter Explorer under *ICU Resources* to determine the real ICU converter's canonical name and get more information on the ICU converter.

With SAGTRPC User Exit

- Once you have determined the number given to SAGTRPC user exit, the implementation of the codepage is your responsibility. No more information is provided.

Broker's Locale String Defaults

If a locale string is not sent by an EntireX component (sender or receiver), the broker itself makes a rough assumption to assign an *ICU Resources* or a numeric codepage number with SAGTRPC user exit.

The broker can distinguish between ASCII environments, IBM mainframe and Fujitsu mainframe operating systems if the EntireX component does not indicate anything by the locale string. See the following table:

Platform EntireX Component is running on	ICU Converter	SAGTRPC User Exit
UNIX, Windows	ibm-5349_P100-1998	1252
IBM Mainframe	ibm-37_P100-1995	0037
Fujitsu Mainframe	bs2000-edf04drv	3587

Note that the broker's built-in defaults above may match for the following countries:

- Many western countries using Windows configured with the ICU converter ibm-5349_P100-1998 (an ICU-supported alias name is, for example, CP1252) including Java environments.
- The United States using OS/390 or z/OS IBM mainframe configured with the ICU converter ibm-37_P100-1995 (ICU-supported alias names are CP37 and ibm-37).

The defaults do not match in the following scenarios, and possibly other scenarios also:

- if codepages other than the ones listed above are used for an operating system
- for UNIX operating systems, including Java environments

You can customize the defaults to your own requirements. See the respective attribute in *Codepage-specific Attributes* (`DEFAULTS=CODEPAGE`) under *Broker Attributes* in the administration documentation for how to customize the broker's locale string defaults. For examples of how to configure the broker's locale strings, see next section, *Configuring Broker's Locale String Defaults*.

Configuring Broker's Locale String Defaults

The broker's built-in defaults for locale strings can be overridden by assigning the ICU converter name directly or an alias of the ICU converter. See *Codepage-specific Attributes* (`DEFAULTS=CODEPAGE`) under *Broker Attributes* in the administration documentation.

This procedure is useful

- if the built-in default does not meet your requirements

- if EntireX components (sender or receiver) do not send locale strings.

Example 1

For this example it is assumed that the internationalization approach is ICU conversion.

An environment running in Spanish-speaking countries using clients with Windows 1252 codepages and servers on IBM mainframe with codepage 1145. Because 1252 is the broker's default for ASCII environments, the default for IBM mainframe is changed to codepage 1145 only, using the ICU converter alias ibm-1145. The previously used codepage 284 is also possible, but does not contain the euro sign.

```
DEFAULTS=CODEPAGE
            /* Broker Locale String defaults */
            DEFAULT_EBCDIC_IBM=ibm-1145
```

As a result, the related ICU converter used as the default for IBM mainframe is ibm-1145_P100-1997. See ICU Converter under ***ICU Resources***.

Example 2

For this example it is assumed that the internationalization approach is ICU conversion.

An environment running in German-speaking countries using Windows 1252 codepages and servers on IBM mainframe with codepage 1141. Because 1252 is the broker's default for ASCII environments, the default for IBM mainframe is changed to codepage 1141 only, using the ICU converter alias ibm-1141. The previously used codepage 273 is still possible but does not contain the euro sign.

```
DEFAULTS=CODEPAGE
            /* Broker Locale String defaults */
            DEFAULT_EBCDIC_IBM=ibm-1141
```

As a result, the related ICU converter used as the default for IBM mainframe is ibm-1141_P100-1997, see ***ICU Converter Explorer*** under ***ICU Resources***.

Example 3

For this example it is assumed that the internationalization approach is ICU conversion.

An environment running in Hong Kong using clients with the Windows 950 (big5) codepage and servers on IBM mainframe with codepage 937. For suitable default values, the broker's default for ASCII environments as well as for IBM mainframes is adapted by assigning ICU converters' alias names.

```
DEFAULTS=CODEPAGE
            /* Broker Locale String defaults */
            DEFAULT_ASCII=windows-950
            DEFAULT_EBCDIC_IBM=ibm-937
```

As a result, the related ICU converter used as the default, (see ***ICU Converter Explorer*** under ***ICU Resources***) is

- for ASCII environments: ibm-1373_P100-2002
- for IBM mainframe: ibm-937_P110-1999

Example 4

For this example it is assumed that the internationalization approach is ICU conversion.

An environment running in Turkey using clients with the Windows 1254 codepage and servers on IBM mainframe with codepage 1026. For suitable default values, the broker's default for ASCII environments as well as for IBM mainframes is adapted by assigning ICU converters' alias names.

```
DEFAULTS=CODEPAGE
            /* Broker Locale String defaults */
            DEFAULT_ASCII=windows-1254
            DEFAULT_EBCDIC_IBM=ibm-1026
```

As a result, the related ICU converter used as the default, (see ***ICU Converter Explorer*** under ***ICU Resources***) is

- for ASCII environments: ibm-1254_P100-1995
- for IBM mainframe: ibm-1026_P100-1995

Bypassing Broker's Built-in Locale String Mapping

The broker's built-in mechanism of mapping locale strings to codepages can be bypassed by assigning the ICU converter name directly or an alias of the ICU converter. See *Codepage-specific Attributes* (`DEFAULTS=CODEPAGE`) under *Broker Attributes* in the administration documentation. Locale string matching is case-insensitive when bypassing the broker's built-in mechanism, that is, when the broker examines the `DEFAULTS=CODEPAGE` section in the attribute file.

- If an EntireX component (sender and receiver) sends a locale string where the broker's built-in mechanism fails and no codepage is found, you can assign a codepage.
- If an EntireX component sends a locale string where the broker's built-in mechanism selects the wrong codepage, you can assign the correct codepage.
- If you cannot adapt the locale string sent by your EntireX component when an incorrect locale string is sent.

Example

For this example it is assumed that the internationalization approach is ICU conversion.

An EntireX Java component sends ASCII as the locale string. This is mapped by ICU to US-ASCII; instead, ISO 8859_1 should be used. This is done with the following configuration:

```
DEFAULTS=CODEPAGE
            /* Broker Locale String Codepage Assignments */
            ASCII=ISO8859_1
```

Using the Abstract Codepage Name LOCAL

In *Windows environments*, the default Windows ANSI codepage configured for your system can also be determined automatically and sent to the broker if the abstract codepage name LOCAL as the locale string is given by the EntireX component. This requires configuration of the EntireX component for an administrator or coding it for a developer and is necessary if communicating with broker version 7.1.x and below. See ***Preparing EntireX Components for Internationalization***. If communicating with broker version 7.2.x and above, the default Windows ANSI codepage is sent to the broker by default.

In *UNIX environments*, no locale string is sent to the broker by default. EntireX components must always be set up for internationalization. See ***Preparing EntireX Components for Internationalization***. Using the abstract codepage name LOCAL requires a well-configured UNIX system for internationalization. See the environment variables `LC_TYPE` and `LC_LANG` and see your UNIX documentation.

In *Java environments* (and with most EntireX components), the default encoding configured for your Java virtual machine can also be determined automatically and sent to the broker if the abstract codepage name LOCAL as the locale string is given by the EntireX component. This requires configuration of the EntireX component for an administrator or coding it for a developer and is necessary if communicating with broker version 7.1.x and below. See ***Preparing EntireX Components for Internationalization***. If communicating with broker version 7.2.x and above, the default encoding configured for your Java virtual machine is sent to the broker by default.

In *all other* operating systems and environments, the abstract codepage name LOCAL is *not* supported. See ***Preparing EntireX Components for Internationalization***.

4 Locale String Mapping Tables

- Mapping Table Usage by Internationalization Approach 32
- Mapping 2-character Language Codes 32
- Mapping UNIX Codepage Names 34
- Mapping Java Codepage Names 36
- Mapping Codepage Numbers 37

It is assumed that you have read the chapter *Introduction to Internationalization* and are familiar with the various internationalization approaches described there.

A locale provides a means of identifying a specific region for the purposes of internationalization and localization. EntireX sends properties of the operating system locale to the Broker, using the locale string.

This chapter provides tables used during the process of mapping the locale string to codepages within the broker for the internationalization approaches *ICU conversion* and *SAGTRPC User Exit*. It does not apply to the other approaches. The locale string is case-insensitive, also dashes '-' and underscores '_' are ignored (dashes and underscore improve human readability) when the broker examines the mapping tables described here.

Mapping Table Usage by Internationalization Approach

The following table provides an overview of what table to use for each internationalization approach.

Mapping Table	Used for Internationalization Approach	
	ICU Conversion	SAGTRPC User Exit
Mapping 2-character Language Codes	yes	yes
Mapping UNIX Codepage Names	no	yes
Mapping Java Codepage Names	no	yes
Mapping Codepage Numbers	no	yes

Mapping 2-character Language Codes

This table is sorted by the ISO 639-2 language codes.

- **If ICU conversion is in effect for a service or topic:**
  Once you have determined the ICU converter alias, use the ICU Converter Explorer under *ICU Resources* to determine the real ICU converter's canonical name.
- **If SAGTRPC user exit is in effect for a service or topic:**
  See column **Mapping to SAGTRPC User Exit Codepage Number** to determine the codepage number given to SAGTRPC user exit.

Two-character Language Code	Language Name	Mapping to SAGTRPC User Exit Codepage Number		Mapping to ICU Converter Alias
		Decimal	Hex	
ar	Arabic	0009	009	ibm-1089
bg	Bulgarian	0915	393	ibm-915
cs	Czech	0912	390	ibm-912
Da	Danish	0850	352	ibm-850
da	Danish	0819	333	ibm-819
De	German	0850	352	ibm-850
de	German	0819	333	ibm-819
el	Greek	0813	32d	ibm-813
En	English	0850	352	ibm-850
en	English	0819	333	ibm-819
Es	Spanish Castilian	0850	352	ibm-850
es	Spanish Castilian	0819	333	ibm-819
Fi	Finnish	0850	352	ibm-850
fi	Finnish	0819	333	ibm-819
Fr	French	0850	352	ibm-850
fr	French	0819	333	ibm-819
he	Hebrew	0916	394	ibm-916
hr	Croatian	0912	390	ibm-912
hu	Hungarian	0912	390	ibm-912
Is	Icelandic	0850	352	ibm-850
is	Icelandic	0819	333	ibm-819
It	Italian	0850	352	ibm-850
it	Italian	0819	333	ibm-819
Ja	Japanese	0932	3a4	ibm-932
ja	Japanese	0018	012	ibm-33722
mk	Marshallese	0915	393	ibm-915
Nl	Dutch	0850	352	ibm-850
nl	Dutch	0819	333	ibm-819
No	Norwegian	0850	352	ibm-850
no	Norwegian	0819	333	ibm-819
pl	Polish	0912	390	ibm-912
Pt	Portuguese	0850	352	ibm-850
pt	Portuguese	0819	333	ibm-819
ro	Romanian	0912	390	ibm-912

Two-character Language Code	Language Name	Mapping to SAGTRPC User Exit Codepage Number		Mapping to ICU Converter Alias
		Decimal	Hex	
ru	Russian	0915	393	ibm-915
sh	Serbo-Croatian	0912	390	ibm-912
sl	Slovenian	0912	390	ibm-912
sk	Slovak	0912	390	ibm-912
sr	Serbian	0915	393	ibm-915
Sv	Swedish	0850	352	ibm-850
sv	Swedish	0819	333	ibm-819
tr	Turkish	0920	398	ibm-920

Mapping UNIX Codepage Names

This table is sorted by UNIX codepage names. It is used if SAGTRPC user exit is in effect for a service or topic. It is not used for other internationalization approaches. You can determine the number given to the SAGTRPC user exit as codepage number.

UNIX Codepage Name	Mapping to SAGTRPC User Exit Codepage Number	
	Decimal	Hex
ascii	0003	003
big5	0950	3b6
eucJP	0018	012
IBM-037	0037	025
IBM-1006	1006	3ee
IBM-1026	1026	402
IBM-1027	1027	403
IBM-1027-DOS	1027	403
IBM-1043	1043	413
IBM-273	0273	111
IBM-277	0277	115
IBM-278	0278	116
IBM-280	0280	118
IBM-284	0284	11c
IBM-285	0285	11d
IBM-290	0290	122
IBM-290-DOS	0290	122

UNIX Codepage Name	Mapping to SAGTRPC User Exit Codepage Number	
	Decimal	Hex
IBM-297	0297	129
IBM-420	0420	1a4
IBM-424	0424	1a8
IBM-500	0500	1f4
IBM-850	0850	352
IBM-850-GL	0850	352
IBM-850-GR	0850	352
IBM-871	0871	367
IBM-918	0918	396
IBM-921	0921	399
IBM-922	0922	39a
IBM-927	0927	39f
IBM-932	0932	3a4
IBM-948	0948	3b4
IBM-eucJP	0018	012
iso88591	0819	333
ISO8859-1	0819	333
iso885910	0919	397
ISO8859-10	0919	397
iso88592	0912	390
ISO8859-2	0912	390
ISO8859-3	0006	006
iso88593	0006	006
iso88594	0914	392
ISO8859-4	0914	392
iso88595	0915	393
ISO8859-5	0915	393
ISO8859-6	0009	009
iso88596	0009	009
iso88597	0813	32d
ISO8859-7	0813	32d
iso88598	0916	394
ISO8859-8	0916	394
iso88599	0920	398
ISO8859-9	0920	398

UNIX Codepage Name	Mapping to SAGTRPC User Exit Codepage Number	
	Decimal	Hex
SJIS	0932	3a4
utf8	4091	ffb

Mapping Java Codepage Names

This table is sorted by the Java codepage names. It is only used if SAGTRPC user exit is in effect for a service or topic. It is not used for other internationalization approaches. You can determine the number given to the SAGTRPC user exit as codepage number.

Java Codepage Name	Mapping to SAGTRPC User Exit Codepage Number	
	Decimal	Hex
ASCII	0003	003
Big5	0950	3b6
EUC_JP	0018	012
ISO8859_1	0819	333
ISO8859_10	0919	333
ISO8859_2	0912	390
ISO8859_3	0006	006
ISO8859_4	0914	392
ISO8859_5	0915	393
ISO8859_6	0009	009
ISO8859_7	0813	32d
ISO8859_8	0916	394
ISO8859_9	0920	398
KOI8_R	2084	824
SJIS	0932	3a4
UFF8	4091	ffb
UTF16	4095	fff

Mapping Codepage Numbers

This table is sorted by the codepage number of the environment of origin. It is only used if SAGTRPC user exit is in effect for a service or topic. It is not used for other internationalization approaches. You can determine the number given to the SAGTRPC user exit as codepage number.

Codepage number	Mapping to SAGTRPC User Exit Codepage Number	
	Decimal	Hex
0037	0037	025
0273	0273	111
0277	0277	115
0278	0278	116
0280	0280	118
0284	0284	11c
0285	0285	11d
0290	0290	122
0297	0297	129
0300	0300	12c
0301	0301	12d
0420	0420	1a4
0424	0424	1a8
0437	0437	1b5
0500	0500	1f4
0775	2087	827
0813	0813	32d
0819	0819	333
0835	0835	343
0836	0836	344
0837	0837	345
0850	0850	352
0858	0858	35a
0870	0870	366
0871	0871	367
0875	0875	36b
0912	0912	390
0913	0006	006
0914	0914	392

Codepage number	Mapping to SAGTRPC User Exit Codepage Number	
	Decimal	Hex
0915	0915	393
0916	0916	394
0918	0918	396
0919	0919	397
0920	0920	398
0921	0921	399
0922	0922	39a
0927	0927	39f
0932	0932	3a4
0935	0935	3a7
0936	0936	3a8
0937	0937	3a9
0942	0942	3ae
0948	0948	3b4
0949	0949	3b5
0950	0950	3b6
0951	0951	3b7
1006	1006	3ee
1025	1025	401
1026	1026	402
1027	1027	403
1041	1041	411
1043	1043	413
1047	1047	417
1088	1088	440
1089	0009	009
1112	1112	458
1115	1115	45b
1122	1122	462
1200	4095	fff
1250	1250	4e2
1251	1251	4e3
1252	1252	4e4
1253	1253	4e5
1254	1254	4e6

Codepage number	Mapping to SAGTRPC User Exit Codepage Number	
	Decimal	Hex
1255	1255	4e7
1256	1256	4e8
1257	1257	4e9
1258	2258	8d2
1380	1380	564
1381	1381	565
3585	3585	e01
3586	3586	e02
3587	3587	e03
3588	3588	e04
3589	3589	e05
5026	3026	bd2
5035	3035	bdb
10001	3001	bb9
20127	0003	003
20866	2084	824
28709	3709	e7d
33722	0018	012
50005	0819	333
50006	0950	3b6
50010	0932	3a4
50012	4091	ffb
51932	0018	012
65001	4091	ffb

5 Preparing EntireX Components for Internationalization

- Rules and Defaults 42
- Table of Components and Locale String Handling 43
- Troubleshooting 46

It is assumed that you have read the chapter ***Introduction to Internationalization*** and are familiar with the various internationalization approaches described there.

This chapter summarizes how to prepare or configure EntireX components to force a locale string to be sent. Sending locale strings by EntireX components is strongly recommended for the internationalization approaches ICU conversion and SAGTRPC user exit. It is not necessary for other internationalization approaches.

Rules and Defaults

Preparing EntireX components for internationalization basically means configuring or providing a locale string, i.e. telling the broker the codepage to be used by the EntireX component. This can be a programming or an administration issue depending on the component used.

It may not be necessary to prepare or configure locale strings

- if the broker's locale string defaults already match (see ***Broker's Locale String Defaults***)
- if the locale string is sent by default to the broker. See column “Locale string sent by default” in the table below

The codepage determined by the broker's locale string processing must be one that is supported by the broker. See ***Locale String Mapping***. This depends on the internationalization approach:

- For ICU conversion it must be an ICU converter.
- For SAGTRPC user exit, the code points implemented must match the code points defined by your application environment's original codepage. See *Writing SAGTRPC User Exits* in the platform-specific administration documentation.

If these rules are not observed, results will be unpredictable.

Table of Components and Locale String Handling

EntireX Component	Locale String sent by Default			Responsibility	Additional information
	Windows	UNIX	z/OS		
Broker ActiveX Control	yes, default Windows "ANSI Codepage" if communicating with all broker version	./.	./.	Programmer	See *Using Internationalization with Broker ActiveX Control*.
EntireX Broker ACI	yes, default Windows "ANSI Codepage"	no	no	Programmer	See *Using Internationalization* under *Writing Applications: Client and Server* in the EntireX Broker ACI Programming documentation.
EntireX Broker Agent	same as on incoming locale string	same as on incoming locale string	./.	not applicable	There is no need to consider anything regarding internationalization. The broker agent does not convert payload data, nor does it suppress, change or add a locale string. The ACI request is rerouted "as is" to the target.
EntireX Broker	not applicable	not applicable	not applicable	Administrator	For information on how to prepare the broker for internationalization, see *Configuring Broker for Internationalization* in the platform-specific administration documentation.
EntireX DCOM Wrapper	yes, default Windows "ANSI Codepage"	./.	./.	Programmer and Administrator	See *Using Internationalization with the DCOM Wrapper* under *Writing Applications with the DCOM Wrapper*.
EntireX Java ACI	yes, default file encoding of JVM	yes, default file encoding of JVM	./.	Programmer	See *Using Internationalization with Java ACI* under *Writing Advanced Applications - EntireX Java ACI*.
EntireX Java Wrapper	yes, default file encoding of JVM	yes, default file encoding of JVM	./.	Programmer	See *Using Internationalization with Java ACI* under *Writing Advanced Applications - EntireX Java ACI*.

EntireX Component	Locale String sent by Default			Responsibility	Additional information
	Windows	UNIX	z/OS		
EntireX Wrapper for Enterprise JavaBeans	yes, default file encoding of JVM	yes, default file encoding of JVM	./.	Programmer and Administrator	See *Using Internationalization with Wrapper for EJB* under *Controlling Applications - EntireX Wrapper for Enterprise JavaBeans*.
EntireX Java RPC Server	yes, default file encoding of JVM	yes, default file encoding of JVM	./.	Administrator	See *Using Internationalization with Java RPC Server* in *Administering the EntireX Java RPC Server* in the UNIX and Windows administration documentation.
EntireX IDL Tester	yes, default file encoding of JVM	yes, default file encoding of JVM	./.		It is not possible to send a locale string to a broker version 7.1.x and below.
EntireX .NET Wrapper	yes, default Windows "ANSI Codepage"	./.	./.	Programmer and Administrator	See *Using Internationalization with the .NET Wrapper* under *Writing Applications with the .NET Wrapper*.
EntireX .NET RPC Server	yes, default Windows "ANSI Codepage"	./.	./.	Administrator	See *Using Internationalization with the .NET Wrapper* under *Writing Applications with the .NET Wrapper*.
C Wrapper	yes, default Windows "ANSI Codepage"	no	./.	Programmer and Administrator	See *Using Internationalization with the C Wrapper* under *Writing Advanced Applications with the C Wrapper*
EntireX RPC Server	yes, default Windows "ANSI Codepage"	no	no	Administrator	See *Using Internationalization with the RPC Server* in the platform-specific administration documentation
Broker HTTP(S) Agent (formerly Tunnel Servlet)	same as on incoming locale string	same as on incoming locale string	./.	not applicable	There is no need to consider anything regarding internationalization. The Broker HTTP(S) Agent does not convert payload data, nor does it suppress, change or add a locale string. The ACI request is rerouted "as is" to the target.
EntireX XML/SOAP RPC Server	yes, default file encoding of JVM	yes, default file encoding of JVM	./.	Administrator	See *Command-line Parameters* in *Administering the EntireX XML/SOAP RPC Server* in the UNIX and Windows administration documentation

EntireX Component	Locale String sent by Default			Responsibility	Additional information
	Windows	UNIX	z/OS		
					and *Using Internationalization with EntireX XML Components* under *Writing Advanced Applications with the XML/SOAP Wrapper*.
EntireX XML Tester	yes, default file encoding of JVM	yes, default file encoding of JVM	./.		It is not possible to send a locale string to a broker version 7.1.x and below.
EntireX XML/SOAP Listener			./.	Administrator	See *Using Internationalization with EntireX XML Components* under *Writing Advanced Applications with the XML/SOAP Wrapper*.
EntireX XML/SOAP Wrapper	yes, default file encoding of JVM	yes, default file encoding of JVM	./.	Programmer	See *Using Internationalization with EntireX XML Components* under *Writing Advanced Applications with the XML/SOAP Wrapper*.
EntireX RPC-ACI Bridge	yes, default file encoding of JVM	yes, default file encoding of JVM	./.	Administrator	See *Configuring the RPC Server Side* under *Administering EntireX RPC-ACI Bridge*.

Legend

- **EntireX component**
  This is the EntireX component for which the information is provided.
- **Locale string sent by default**
  Whether a locale string is sent to the broker depends on the operating system the component is running on and whether the broker being communicated with is version 7.2.x and later. Locale strings are never sent by default to earlier brokers; they must be enabled manually.
- **Responsibility**
  This column specifies who is responsible for allowing locale strings to be sent. The fact that an administrator can change the default codepage of an environment, i.e. the Windows "ANSI codepage" or the file encoding property of a JVM is not documented - this can always be done.
  - **Programmer**
    means that the programmer has control of and responsibility for the locale strings. The programmer can force a fixed codepage to be used and define the default codepage of the environment that is to be used. In the first case no administration tasks remain and in the latter case the administrator can change the codepage but cannot prevent use of locale strings.

- **Administrator**
  means that an administrator can enable or disable whether locale strings are sent or not after an application has been built.
- **Programmer and administrator**
  means that the administrator can enable or disable locale strings to be sent after an application has been built, if the programmer has left the item open.

See the component documentation for details.

- **Additional Information**
  This column indicates where you can find additional information.

Troubleshooting

It is important to know the locale strings of both components (sender and receiver) to predict conversion behavior accurately. See *Locale String Mapping*. The quickest and easiest place to check this and the place where both locale strings are captured and are visible is the broker kernel trace.

We recommend using a broker kernel trace with level 1 to find out and observe whether both sides of an application (sender and receiver) send locale strings.