webMethods EntireX

Internationalization with EntireX

Version 10.9

April 2023

WEBMETHODS

This document applies to webMethods EntireX Version 10.9 and all subsequent releases.

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

Document ID: EXX-INTERNAT-109-20230403

Table of Contents

Preface

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.

Introduction	Introduction to ICU Conversion; provides links to configuration and ICU resources; provides information to Arabic shaping, EBCDIC Stateful, multi-byte and doublebyte codepages; broker's mechanism to select the character conversion approach, etc.
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.

1 About this Documentation

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

In addition, you can also access the cloud product documentation via **https://www.softwareag.cloud**. Navigate to the desired product and then, depending on your solution, go to "Developer Center", "User Center" or "Documentation".

Product Training

You can find helpful product training material on our Learning Portal at **https://knowledge.softwareag.com**.

Tech Community

You can collaborate with Software AG experts on our Tech Community website at **https://tech-community.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://hub.docker.com/publishers/softwareag** and discover additional Software AG resources.

Product Support

Support for Software AG products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Introduction to Internationalization

Overview

Character conversion 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. Character conversion with EntireX provides the following:

- Character 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.
- Character conversion deals with the user's payload data in broker's send and receive buffers.
- For ICU Conversion, 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. Depending on the platform your EntireX component is running on, the locale string is sent automatically by default or must be provided. A huge set of codepages is available and supported; see ***ICU Converter Explorer***.
- Inside the EntireX Broker an automatic mechanism tries to find the best character conversion approach for your scenario. See *Broker's Mechanism for Choosing the Character Conversion Approach*.

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

ICU Conversion

This section covers the following topics:

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***.

Configuration and Usage

To use ICU conversion, 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.

By default it is assumed that the payload sent to/received from the broker matches the platform's default code page. EntireX components running under the Windows operating system and Java-based EntireX components send this platform default code page identifier automatically to the broker, so in most cases nothing needs to be configured or considered by a programmer here. Configuration or programmer attention is required in the following cases:

- The EntireX component is running under the operating systems z/OS, Linux, z/VSE or BS2000. No code page identifier is sent automatically on these platforms.
- You require a code page other than the platform default.

Configuration for RPC Servers and Listeners

Refer to the respective sections of the documentation for how to enable RPC servers and listeners to send a codepage identifier to the broker or send a different identifier than the default codepage for the platform.

- *Configuring the RPC Server* under C | .NET | XML/SOAP | Java | z/OS (CICS, Batch, IMS) | BS2000
- *Configuring the IBM MQ Side* (RPC Server for IBM MQ | Listener for IBM MQ)
- *Configuring the IMS Connect Side*
- *Configuring the CICS IPIC Side*
- *Configuring the CICS ECI Side*
- *Configuring the IBM AS/400 Side*

Configuration for RPC Clients

Enabling RPC clients to send a codepage identifier to the broker or send a different identifier than the default codepage for the platform is a task for a programmer. See the following sections of the documentation:

- *Using Internationalization* with the C Wrapper | DCOM Wrapper | .NET Wrapper | Java Wrapper | PL/I Wrapper

Configuration for ACI-based Programming

For ACI-based programming, see:

- *Using Internationalization with Java ACI*
- For Broker ActiveX Control, see `localeString` under *Reference* > *Properties*
- For *EntireX Broker ACI* for Assembler | C | COBOL | Natural | PL/I | RPG, see `LOCALE-STRING` under *Broker ACI Fields*.

ICU Resources

This section covers the following topics:

ICU Homepage

The ICU home page (***http://site.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:

- Linux: */<Install_Dir>/EntireX/etc/convrtrs.txt*
- Windows: *<drive>:\SoftwareAG\EntireX\etc\convrtrs.txt*

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.

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 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.

Arabic Shaping

Introduction

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:

- 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.

Shaping is performed only on the codepages listed above. Arabic text data must be in *logical* order; *visual* order is not supported.

During character conversion, data length may increase or decrease for this type of codepage. The ***Rules for Data Length Changes due to Character Conversion*** apply.

Configuration

Configuration is the same as with ICU Conversion, see ***Configuration and Usage*** in section *ICU Conversion*.

Hebrew Codepage 803

Introduction

CP803 does not include Latin lowercase characters. For RPC-based components and Reliable RPC, error messages, ping replies etc. are converted to uppercase before conversion to CP803, so Hebrew Codepage 803 can be used with RPC. Application Latin lowercase characters cannot be used within application data IDL type A, IDL AV, IDL program and IDL library.

For ACI-based programming there is no special behavior. Latin lowercase characters cannot be used.

Configuration

Configuration is the same as with ICU Conversion, see ***Configuration and Usage*** in section *ICU Conversion*.

EBCDIC Stateful Codepages

Introduction

These codepages are designed for use in Asian countries. They are encoded using escape technique (SI/SO bytes). An example is CP930 (Japan).

For RPC-based components and Reliable RPC, we recommend RPC programmers observe the following, otherwise unpredictable results may occur:

- **IDL data types K, KV**
  - SO and SI escape characters may not be contained
  - only double-byte characters allowed
  - single-byte characters cannot be transferred
- **IDL data types A, AV**
  - single-byte and double-byte characters can be transferred using SO and SI escape characters

For ACI-based programming, single-byte and double-byte characters can be transferred using SO and SI escape characters.

During character conversion, data length may inrease or decrease for this type of codepage. The ***Rules for Data Length Changes due to Character Conversion*** apply.

Configuration

Configuration is the same as with ICU Conversion, see ***Configuration and Usage*** in section *ICU Conversion*.

Multibyte or Double-byte Codepages

Examples are UTF8, CP950, Big5 (traditional Chinese).

Introduction

During character conversion, data length may inrease or decrease for this type of codepage. The ***Rules for Data Length Changes due to Character Conversion*** apply.

Configuration

Configuration is the same as with ICU Conversion, see ***Configuration and Usage*** in section *ICU Conversion*.

Rules for Data Length Changes due to Character Conversion

Character conversion may force a decrease or increase in data lengths. Such data length changes occur for ***Multibyte or Double-byte Codepages*** (for example UTF8), ***EBCDIC Stateful Codepages***, and if ***Arabic Shaping*** is in effect. Data length changes do not occur for single-byte codepages.

For RPC-based components and Reliable RPC, behavior depends on the IDL data type (see *IDL Data Types* in the IDL Editor documentation). RPC programmers should be aware of the following:

- **IDL data types AV, KV**
  - IDL field length may increase or decrease. The resulting length is set accordingly.
- **IDL data types A, K**
  - IDL field length cannot change.
  - If the data length decreases, padding with SPACE occurs.
  - If the data length increases characters are cut off at character boundaries, which may force again padding with SPACE.

- **IDL data types AVn, KVn**
  - IDL field length has a maximum.
  - If the data length decreases, the field length is adjusted accordingly.
  - If the data length increases over the maximum length characters are cut off at character boundaries. The resulting length is set accordingly.

For ACI-based programming, the complete payload may change its length in bytes during conversion.

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.

Broker's Mechanism for Choosing the Character Conversion Approach

The automatic mechanism for choosing the character conversion approach applies to the following versions:

- EntireX Broker 10.1 and above (z/OS, Linux, Windows)
- EntireX Broker 10.3 and above (BS2000)

For example, RPC components indicate to the broker that the data stream is RPC; the broker uses this information to choose the character conversion approach. In this way, incorrect configurations are detected and corrected.

Broker Attribute File Definition	RPC Data Stream Detected [1]	ACI Data Stream Detected [2]	ACI or RPC Data Stream [3]
CONVERSION=*user exit*	*user exit*	*user exit*	*user exit*
TRANSLATION=*user exit*	*user exit*	*user exit*	*user exit*
CONVERSION=SAGTCHA	**CONVERSION=SAGTRPC**	CONVERSION=SAGTCHA	CONVERSION=SAGTCHA
CONVERSION=SAGTRPC	CONVERSION=SAGTRPC	**CONVERSION=SAGTCHA**	CONVERSION=SAGTRPC
CONVERSION=NO or TRANSLATION=NO	**CONVERSION=SAGTRPC**	no character conversion	no character conversion
TRANSLATION=SAGTCHA [4]	**CONVERSION=SAGTRPC**	**CONVERSION=SAGTCHA**	**CONVERSION=SAGTCHA**

Key

BOLD Character conversion is determined by the broker automatically; the definition in the broker attribute file is ignored and a warning message (one of 00200781, 00200782, 00200786, 00200787, 00200788, 00200789, 00200790) is written to the broker log file. Adapt your broker attribute file to avoid the message.

Notes

1. RPC data stream is detected automatically by the broker if the version of RPC server component is the following (or above):
   - EntireX RPC Server (BS2000) 10.3
   - EntireX RPC Server (z/OS CICS, z/OS Batch, z/OS IMS, C, .NET) 9.10
   - EntireX RPC Server (Java, CICS ECI, IMS Connect, XML/SOAP, RPC-ACI, IBM MQ) 9.9

- EntireX Adapter 9.9
- Natural RPC Server (Mainframe 8.2.7, Open Systems 8.4.1)

2. ACI data stream is detected automatically by the broker from EntireX Java ACI 9.12 or later.
3. If ACI communication is used from non-Java environments, or the EntireX RPC server or Natural RPC server is from an earlier version than listed under Note 1, the data stream can be ACI or RPC.
4. `TRANSLATION=SAGTCHA` is ignored. The broker uses `CONVERSION`.

Order of Precedence

Character conversion is chosen by the broker in the following order of precedence:

1. You can always write your own ***User Exits*** if this matches your requirements. This is the first choice if defined.
2. If the broker detects an RPC data stream (see Note 1 above), ICU conversion with broker attribute `CONVERSION=SAGTRPC` is used.
3. If neither broker attribute `CONVERSION` nor broker attribute `TRANSLATION` is defined (the attribute is omitted or set to `NO`) no character conversion occurs.
4. If the broker detects an ACI data stream (see Note 2 above), ICU conversion with broker attribute `CONVERSION=SAGTCHA` is used.
5. If the broker attribute `CONVERSION=SAGTRPC` is defined, ICU conversion approach SAGTRPC is used.
6. In all other cases, ICU conversion approach SAGTCHA is used.

User Exits

Translation User Exit

Introduction

With translation user exits, the code points of the codepage used are under your control. You can distinguish between ASCII, IBM EBCDIC and BS2000 EBCDIC environments (where the caller or participant is running). Code points can be adapted 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.

For RPC-based components and Reliable RPC, the codepage you implement must meet the ***Codepage Requirements for RPC Data Stream Conversions***.

For ACI-based programming, 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.

Configuration

Configuration effort is easy, only `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).

We do not recommend using a translation user exit. If you only need to adapt codepoints, consider migration to ***ICU Conversion***.

Migration to ICU Conversion

Introduction

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 `CONVERSION-OPTION-SUBSTITUTE` set for the same error behavior as translation. See *`OPTION` Values for Conversion*.

Configuration

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 attribute `CONVERSION`:

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

For RPC-based components and Reliable RPC, set the service-specific 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, and `CONVERSION=SAGTRPC` is configured in the broker attribute file. 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. See *Writing SAGTRPC User Exits* under *Configuring Broker for Internationalization* in the platform-specific Administration documentation. The codepage you implement must meet the ***Codepage Requirements for RPC Data Stream Conversions***.

Configuration

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.

3 Locale String Mapping

It is assumed that you have read the *Introduction to Internationalization*.

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 character conversion with ICU conversion and SAGTRPC user exit. It does not apply to other approaches.

Broker's Locale String Processing

Depending on the character conversion approach in effect for a service, 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 suit 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 for correct character conversion 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.

The last two 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.

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

Locale String Sent to Broker	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.	■ *by default* from EntireX components running in Windows ■ 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.	■ if manually configured or provided by programmer.	The part <ll> is extracted and used together with the table ***Mapping Two-character Language Codes*** to map to an ICU converter alias. The part <cc> is not used.	The part <ll> is extracted and used together with the table ***Mapping Two-character Language Codes*** to evaluate the number given to the SAGTRPC user exit. The part <cc> is not used.
<ll>_<cc>.<Linux-codepage-name> where <ll> is the 2-letter language abbreviation, <cc> the 2 or 3-letter language code according to ISO 639 standard and <Linux-codepage-name> is any alias name for a codepage.	■ if manually configured or provided by programmer.	The part <Linux-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 <Linux-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.	■ *by default* from EntireX components running in Java environments ■ if manually configured or provided by a programmer	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 Numbers*** to evaluate the number given to the SAGTRPC user exit.

Locale String Sent to Broker	Locale String Format Sent	ICU Converter Alias	SAGTRPC User Exit Codepage Number
<codepage-name> where <codepage-name> is any alias name for a codepage.	■ *by default* from EntireX components running in Java environments ■ if manually configured or provided by a programmer	The locale string <codepage-nam>e 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.

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
Linux, 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).

You can customize the defaults to your own requirements. See the respective attribute in *Codepage-specific Attributes* 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*.

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 character conversion 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 character conversion 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 character conversion 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 is

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

See *ICU Converter Explorer* under *ICU Resources*.

Example 4

For this example it is assumed that the character conversion 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 is

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

See *ICU Converter Explorer* under *ICU Resources*.

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*. 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 character conversion 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
```

4 Locale String Mapping Tables

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 underscores 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 Two-character Language Codes	yes	yes
Mapping UNIX Codepage Names	no	yes
Mapping Java Codepage Names	no	yes
Mapping Codepage Numbers	no	yes

Mapping Two-character Language Codes

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

- **If ICU conversion is in effect for a service:**
  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:**
  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. 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. 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. 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