

# Type Conversion Library

## Table of contents

1	1. Type Conversion Library (####)	2
1.1	1.1 typeConversion_lang.xsl	2
1.2	1.2 typeConversion_chdr.xsl	2
2	2. Type Conversion Library (####)	3
2.1	2.1 W3C_typeConversion_lang.xsl	3
2.2	2.2 W3C_typeConversion_chdr.xsl	3
3	3. Type Conversion Library (Library Function)	4
3.1	3.1 W3C_typeConversionLibrary.{h,c}	4
3.2	3.2 Type Conversion Library (Library Function)	5

## 1. 1. Type Conversion Library (#####)

UMS ###RELAX NG #####

- encode##datatype
- datatype#####

#####  
Library ##### umsCodeGenerator ##### XSLT #####

##### umsCodeGenerator ### XSLT ## ##### Type Conversion Library #####  
umsCodeGenerator ### XSL ##datatypeLibrary ##### 'library'  
#####template ##### template #'library' ##### # Type Conversion Library  
## XSLT ##### # Type Conversion Library #####

### Type Conversion Library ##### XSLT #####

##### [#####](#) (tableTools-2005062851.txt) ([#####](#) (meta.tar.gz) )

### 1.1. 1.1 typeConversion\_lang.xsl

```
<stylesheet>
  <include href="file://localhost/W3C_typeConversion_lang.xsl" />

  <template name="initLibrary" />

  <template name="encode2valueTypeConversion"
    param="library, datatype, encode, cont10, cont20,
constant"/>
  <template name="encode2datatypeConversion"
    param="library, datatype, encode, cont10, cont20"/>
  <template name="datatype2langTypeConversion"
    param="library, datatype, langtype, varname"/>
  <template name="langtype2datatypeConversion"
    param="library", datatype, langtype, varname"/>
  <template name="datatype2encodeConversion"
    param="library, datatype, encode, cont10, cont20"/>
  <template name="valueType2encodeConversion"
    param="library, datatype, encode, cont10, cont20,
constant"/>
</stylesheet>
```

### 1.2. 1.2 typeConversion\_chdr.xsl

```
<stylesheet>
  <include href="file://localhost/W3C_typeConversion_chdr.xsl" />
```

## Type Conversion Library

```
<template name="useLibrary" />
</stylesheet>
```

### 2.2. Type Conversion Library (####)

```
##### Type Conversion Library ##### umsCodeGenerator
##### W3C #####

#### template ##### datatype ##### data #### value ### type
##### umsCodeGenerator ### XSL ##### # type conversion library
##### #### datatype #####

## XSL #####

<value /> #####
##### type ##### XSL ##### JAVA #####
```

#### 2.1. 2.1 W3C\_typeConversion\_lang.xml

```
<stylesheet>

  <template name="W3C_initLibrary"/>

  <template name="W3C_dataTypeDefinition"
    param="datatype"/>
  <template name="W3C_encode2valuetypeConversion"
    param="datatype, encode, cont10, cont20, constant"/>
  <template name="W3C_encode2datatypeConversion">
    param="datatype, encode, cont10, cont20"/>
  <template name="W3C_datatype2langtypeConversion">
    param="datatype, langtype, varname"/>
  <template name="W3C_langtype2datatypeConversion">
    param="datatype, langtype, varname"/>
  <template name="W3C_datatype2encodeConversion">
    param="datatype, encode, cont10, cont20"/>
  <template name="W3C_valuetype2encodeConversion">
    param="datatype, encode, cont10, cont20, constant"/>

</stylesheet>
```

#### 2.2. 2.2 W3C\_typeConversion\_chdr.xml

```
<stylesheet>
  <template name="W3C_useLibrary"/>
</stylesheet>
```

### 3.3. Type Conversion Library (Library Function)

```
#####
#####
#####umsCodeGenerator#####
#####typeConversionLibrary # #####
```

#### 3.1.3.1 W3C\_typeConversionLibrary.{h,c}

```
void ums__W3C_typeConversionLibrary_init(void);
void ums__W3C_typeConversionLibrary_end(void);

char *ums__W3C_txtEncode_stringType
(ums__dat_t *dat, int epos, ums__exception_t *ex);
char *ums__W3C_txtEncode_tokenType
(ums__dat_t *dat, int epos, ums__exception_t *ex);
int32_t ums__W3C_txtEncode_intType
(ums__dat_t *dat, int epos, ums__exception_t *ex);
uint32_t ums__W3C_txtEncode_unsignedIntType
(ums__dat_t *dat, int epos, ums__exception_t *ex);
double ums__W3C_txtEncode_doubleType
(ums__dat_t *dat, int epos, ums__exception_t *ex);
int32_t ums__W3C_signedEncode_intType
(ums__dat_t *dat, int epos, ums__exception_t *ex);
uint32_t ums__W3C_unsignedEncode_unsignedIntType
(ums__dat_t *dat, int epos, ums__exception_t *ex);
double ums__W3C_ieee754doubleEncode_doubleType
(ums__dat_t *dat, int epos, ums__exception_t *ex);

void ums__W3C_stringType_txtEncode
(char * str, ums__dat_t *dat, int epos, ums__exception_t *ex);
void ums__W3C_tokenType_txtEncode
(char * str, ums__dat_t *dat, int epos, ums__exception_t *ex);
void ums__W3C_intType_txtEncode
(int32_t num, ums__dat_t *dat, int epos, ums__exception_t *ex);
void ums__W3C_unsignedIntType_txtEncode
(uint32_t num, ums__dat_t *dat, int epos, ums__exception_t *ex);
void ums__W3C_doubleType_txtEncode
(double num, ums__dat_t *dat, int epos, ums__exception_t *ex);
void ums__W3C_intType_signedEncode
(int32_t num, ums__dat_t *dat, int epos, ums__exception_t *ex);
void ums__W3C_unsignedIntType_unsignedEncode
(uint32_t num, ums__dat_t *dat, int epos, ums__exception_t *ex);
void ums__W3C_doubleType_ieee754doubleEncode
(double num, ums__dat_t *dat, int epos, ums__exception_t *ex);

void ums__W3C_txtEncode_stringValue
(ums__dat_t *dat, int epos, char *literal, ums__exception_t *ex);
void ums__W3C_txtEncode_tokenValue
(ums__dat_t *dat, int epos, char *literal, ums__exception_t *ex);
```

## Type Conversion Library

```
void ums__W3C_txtEncode_intValue
    (ums__dat_t *dat, int epos, int32_t literal, ums__exception_t *ex);
void ums__W3C_txtEncode_unsignedIntValue
    (ums__dat_t *dat, int epos, uint32_t literal, ums__exception_t *ex);
void ums__W3C_txtEncode_doubleValue
    (ums__dat_t *dat, int epos, double literal, ums__exception_t *ex);
void ums__W3C_signedEncode_intValue
    (ums__dat_t *dat, int epos, int32_t literal, ums__exception_t *ex);
void ums__W3C_unsignedEncode_unsignedIntValue
    (ums__dat_t *dat, int epos, uint32_t literal, ums__exception_t *ex);
void ums__W3C_ieee754doubleEncode_doubleValue
    (ums__dat_t *dat, int epos, double literal, ums__exception_t *ex);
```

### 3.2. 3.2 Type Conversion Library (Library Function)

```
public class W3C_TypeConversionLibrary {

    public static String txtRead(UMSDat dat, int epos)
        throws UMSEException;
    public static String txtEncode_stringType(UMSDat dat, int epos)
        throws UMSEException;
    public static String txtEncode_tokenType(UMSDat umsDat, int epos)
        throws UMSEException;
    public static int txtEncode_intType(UMSDat umsDat, int epos)
        throws UMSEException;
    public static double txtEncode_doubleType(UMSDat umsDat, int epos)
        throws UMSEException;
    public static int signedEncode_intType(UMSDat umsDat, int epos)
        throws UMSEException;
    public static double ieee754doubleEncode_doubleType(UMSDat umsDat, int
    epos)
        throws UMSEException;
    public static void bitWrite(long num, UMSDat umsDat, int epos)
        throws UMSEException;
    public static void stringType_txtEncode(String str, UMSDat umsDat, int
    epos)
        throws UMSEException;
    public static void tokenType_txtEncode(String str, UMSDat umsDat, int
    epos)
        throws UMSEException;
    public static void intType_txtEncode(int num, UMSDat umsDat, int epos)
        throws UMSEException;
    public static void doubleType_txtEncode(double num, UMSDat umsDat, int
    epos)
        throws UMSEException;
    public static void intType_signedEncode(int num, UMSDat umsDat, int epos)
        throws UMSEException;
    public static void doubleType_ieee754doubleEncode(double num, UMSDat
    umsDat, int epos)
        throws UMSEException;
    public static void txtEncode_stringValue(UMSDat dat, int epos, String
    literal)
```

```
        throws UMSEException;
    public static void txtEncode_intValue(UMSDat dat, int epos, int literal)
        throws UMSEException;
    public static void txtEncode_doubleValue(UMSDat dat, int epos, double
literal)
        throws UMSEException;
    public static void signedEncode_intValue(UMSDat dat, int epos, int
literal)
        throws UMSEException;
    public static void ieee754doubleEncode_doubleValue(UMSDat dat, int epos,
double literal)
        throws UMSEException;
}
```