Design Documentation for the byte datatype in Moto

Overview

We need to add a byte datatype to moto to pave the way both for better handling of binary data objects and allowing the char datatype to move to 16 bits.

Implications

For consistency we should probably add a Byte datatype as well.

mx_string.c
* add byte_toString

stringbuffer.c
* add buf_puty

motoutil.c
* add yv(MotoVal*) function

type.h
* add BYTE_TYPE
* add byteVal

env.c
* modify addBuiltInType to include “byte”, BYTE_TYPE
* modify initVal to initialize the BYTE_TYPE to 0
* modify setVarVal
* modify getVarVal
* modify moto_castVal to cast to and from the other elementary types
* modify moto_typeToCType to map BYTE_TYPE to “unsigned char”
* modify moto_defaultValForCType to give the BYTE_TYPE a default value of 0

motoc.c
* modify arr_gen_get to call the appropriate array sub function (ysub)
* modify motoc_array_new
* modify motoc_bitop
* modify motoc_print (should see how java handles byte output)
* modify motoc_incDec
* modify motoc_doMath

motoi.c
* modify motoi_array_rval
* modify motoi_bitop
* modify motoi_bitnot
* modify motoi_shift
* modify motoi_fillargs
* modify motoi_incdec
* modify motoi_fn
* modify motoi_value
* modify motoi_dorel
* modify buf_putmv
* modify do_math

motov.c
* modify motov_bitop
* modify motov_shiftop
* modify motov_bitnot
* modify motov_incdec
* modify motov_domath
* modify motov_checkVarAssign
* modify motov_checkValAssign
* modify motov_checkCast

* mxarr.c

* mxarr.h
* add a ysub function for byte arrays

New moto methods that should take or recieve byte arrays
- String String::String(byte[])
- byte[] String::getBytes()
- byte[] StringBuffer::append(byte[])
- byte[] MySQLResultSet::getBytes()
- byte[] PGResultSet::getBytes()
- byte[] MIMEMessage::getBytes()

- *byte[] File::getBytes() ?*

Design Considerations

We need to be able to output byte arrays!

Offerring StringBuffer::append(byte[]) makes just as much sense as being able to create Strings from byte arrays. The nice side benefit of this for the 0.14.0 release is that since output is bufferred in StringBuffers getting byte[]s into the output stream should be pretty easy

If StringBuffers go unicode then the append(byte[]) method will do locale specific byte conversion and we will *potentially* stop buffering output in StringBuffers since stdout **is really a byte stream**

Do we need ByteBuffers ?