**Java Primitive Types:**

|  |  |
| --- | --- |
| **Data Type** | **Information** |
| Byte | **How to Read Bytes:** An eight bit signed two’s complement number. For example, if you have the byte 01111100 since it starts with a zero, you know it is a positive number. So you just convert it to integer (base 10) and you get  (0)(27) +(1)(26) +(1)(25) +(1)(24) +(1)(23) +(1)(22) +(0)(21) +(0)(20) = 124.  Now Supose you had the byte 10000100. In this case, since it starts with a 1, you know it is negative. To find the magnitude of the negative number, invert all the bits and add 1. This yields  10000100 –-inverted--> 01111011 ---adding one--> 01111100  As we seen above, that number is 124, so the value is 10000100 is -124.  **Maximum and Minimum Values:** The maximum value is 01111111 which yields 127. The minimum value would be 10000000 which when you invert, becomes 0111111 + 1 = 10000000 = 128. This means the maximum value is 127 and the minimum value is -128 for a single byte. In general the range the {max, min} range a data type with n-bits can store/represent is {-2n-1, 2n-1 – 1}.  **Initializing a single byte in Java:** You would have to type cast from integer. You also have to watch out for overflows. The maximum number you could store in a single byte without overflow is 127. So if you try and type cast 129 to a byte, it would overflow and -127 would be stored in the byte. If you try to store (129 + 127 = =256) into the byte, it would overflow by a total of 128 which would give a byte value of zero (-128 + 128 = 0). |
| Integer | An integer is stored in 32 bits or 4 bytes. The range that an integer can hold without overflow is {-231, 231 – 1}. |
| Long | A long is like an integer but with 64 bits. It has a much wider range of numbers it can represent without overflow. |
| String |  |
| String Builder |  |
| Arrays |  |