Forge Notification System Protocol Specification Version 1.0

1) Connection is achieved via a simple TCP Connection as implemented by the Java `java.net.Socket` / `java.net.ServerSocket` classes.
2) The Byte Stream of the Connection is split into Packets (not to confuse with TCP packets!). Multiple packets can be send without establishing a new Connection for each Packet.
3) There is no defined Packet order, each packet can be send at any time
4) The template for each Packet is as follows:
   - A short value, representing the total length of the packet in bytes, excluding this short value
   - 128 bytes, representing an RSA-1024-bit encrypted SHA1 hash of the packet contents (excluding the hash itself)
   - A String containing the protocol version (More on Data types below). Currently "`1.0`"
     Packets must not be parsed and the Connection should be closed if the received Protocol version does not match the Protocol version that the receiving Program implements
   - A String defining the Type of the packet. The type is case insensitive.
     Currently defined types:
     - `REWARD` represents a reward that should be given to a player
       Contents are:
       A String representing the username of the player the reward should be given to
       A long value representing the UNIX timestamp of when the reward was earned in UTC
   - The actual Packet contents, depending on the packet type (see above)
   - A single byte. A value of 0 represents the connection should be closed after this packet. Any value different from 0 represents that the connection should be kept open to receive further packets. The value gives no indication on how many packets will arrive still.
5) Short transfer: A short value is transferred as 2 bytes (`a + b`), such that the short value can be reconstructed with `(a << 8) | (b & 0xff)`
   The short value is, unless otherwise noted, unsigned so that values from 0 through 65535 can be transferred
6) Long transfer: A long value is transferred as 8 bytes (`a – h`) such that the long value can be reconstructed with
```
((a & 0xff) << 56) |
((b & 0xff) << 48) |
((c & 0xff) << 40) |
((d & 0xff) << 32) |
((e & 0xff) << 24) |
((f & 0xff) << 16) |
((g & 0xff) <<  8) |
((h & 0xff)))
```
7) String transfer: A String is transferred as follows:
   A short value that describes the total length of the UTF-8 representation of the String in bytes
   The UTF-8 representation of the String as a number of bytes