# Verso Library F.A.Q.

## General questions

### What is Verso Library?

It's a networking library. It's used to create servers, clients or any kind of network application.

### What does Verso Library require?

It requires the .Net / Mono framework.

### Why should I use Verso Library?

Verso Library provides a high level API for network application creation. It encapsulates the complexities of networking, thread management, security, message composing and protocols, providing and easy to use, event oriented interface.

Verso Library is lightweight, current build size is about 30 KB. It's efficient, fast, with almost negligible memory footprint.

Verso Library features out-of-the-box support for several text, binary and mixed protocols, currently supporting plain text, sized binary messages, Websockets, AMF3, and more to come. Implementation of new protocols is straightforward extending the base ones.

Verso Library is fully compatible with Unity for all target platforms, and it's also a perfect solution to build back-ends for multiuser Flash applications.

### Does Verso Library support secure connections?

It supports secure connections, SSL or TLS. It supports server authentication or mutual authentication.

## Platform specific

### How can I benefit from Verso Library as a Unity developer?

Verso Library is a pure .Net library specially built to be fully compatible with Unity built-in Mono framework. It works on every Unity target platform: PC/Mac standalone, iOS, Android, and all others.

Most classes in UnityEngine assembly are not thread friendly. Verso Library features a TaskManager class that allows your application to run all Verso Library events in your application's main thread.

### How can I benefit from Verso Library as a Flash developer?

Using the built-in AMF3 protocol, you can send and receive full objects from AS3 client to C# server and viceversa. No client library is required, native methods *writeObject* and

*readObject* from the Socket class are fully compatible. Objects, arrays and data types are translated using the following table:

| Actionscript | .Net |
|---|---|
| int | int |
| Number | double |
| String | string |
| Date | DateTime |
| Array | List<object> |
| Object | Dictionary<string, object> |
| Boolean | bool |
| Null | null |

# Server implementation

### Can I build a really performant server using Verso Library?

Verso Library relies on the robust, well-proven .Net framework thread pool. All network operations are asynchronous and receiving threads are independent from worker threads.

Verso Library includes a Task Manager that allows the addition and removal of worker threads at runtime. Recommended number of workers for optimal performance is one for each processing core in your system.

# Thread safety

### Is Verso Library multi-threaded?

Verso Library is multi-threaded in nature since network data transfers are performed in dedicated threads. From the application's point of view, Verso Library can behave as multi-threaded or single-threaded depending of your needs.

For multi-threaded operation, you must create worker threads by using the Verso Library Task Manager. Worker threads will check for pending tasks and will dispatch the corresponding events.

For single-threaded operation, you must call the Task Manager provided method from your application's main loop to execute all pending events.

# Protocols

## Does Verso Library support simple text transfer?

Verso Library provides a base text protocol out-of-the-box. This is a configurable protocol capable of sending and receiving text with customized encoding and line endings.

We can use this protocol to transfer simple text lines, or serialized objects, or we can use it to implement higher level application protocols.

## Does Verso Library support JSON objects?

JSON is a text based object format. Since Verso Library provides a base protocol for text transfers, JSON objects can be transferred out-of-the-box. Verso Library does not provide encoding or decoding for JSON objects. You can use the native JSON serialization in the .Net framework or some of the freely available serialization libraries like MiniJSON.

## Does Verso Library support XML / SOAP?

Verso Library does not provide SOAP serialization by itself, but it integrates perfectly with the .Net native SoapFormatter class.

## Does Verso Library support raw and/or binary data transfer?

Although not recommended, your application can receive raw packets as they come from the socket, just by listening to the events in the base Connection class. Normally, application level messages don't match the raw packets. Messages are usually broken into several packets, or a packet contains several messages, or a combination of both. It's recommended to rely on higher level protocols where an event is dispatched when a full message is received, leaving the complexities of message building hidden to your application.

If your application needs to send and/or receive binary data blocks of variable size, such as binary serialized objects, the included BinaryProtocol and BinaryConnection classes will perfectly fit the purpose.

## Does Verso Library provide binary serialization for simple or complex types?

Verso Library doesn't provide serialization capabilities by itself, because it integrates perfectly with .Net serializers. For simple data types use System.IO.BinaryReader and System.IO.BinaryWriter classes, for complex objects use BinaryFormatter class.