### Versions

Adapter baselibrary .net35, .net4, .net45, .pcl

Kludges

Lazy

Nameable

### Bundles

#### Bundle01Web

Platforms

MvcTurbine

SPG2010

Web

Web.Mvc2

Web.Mvc3

Web.Mvc4

Web.Mvc5

### EventSource

IEventBus

|  |  |
| --- | --- |
| Method | Description |
| TMessage CreateMessage<TMessage>(Action<TMessage> messageBuilder)  where TMessage : class | Creates the message. |
| IServiceBusCallback Send(IServiceBusEndpoint destination, params object[] messages) | Sends the specified destination. |
| void Reply(params object[] messages) | Replies the specified messages. |

IEventSource

|  |  |
| --- | --- |
| Method | Description |
| IAggregateRootRepository MakeRepository<T>(T arg, ITypeSerializer serializer) | Makes the repository. |

IEventHandler<TEvent> where TEvent : IEvent

|  |  |
| --- | --- |
| Method | Description |
| void Handle(TEvent @event) | Handles the specified @event. |

|  |  |
| --- | --- |
| IMPLEMENTATIONS |  |
| EventStore - ESEventSource   |  |  | | --- | --- | | Method | Description | | ESEventSource(IStoreEvents store) | Initializes a new instance of the ESEventSource class. | | MongoDB - MongoDBEventSource   |  |  | | --- | --- | | Method | Description | | MongoDBEventSource(string connectionString) | Initializes a new instance of the MongoDBEventSource class. | |
| MSSql - MSSqlEventSource   |  |  | | --- | --- | | Method | Description | | MSSqlEventSource(string connectionString) | Initializes a new instance of the MSSqlEventSource class. | | NEventStore - NESEventSource   |  |  | | --- | --- | | Method | Description | | NESEventSource(IStoreEvents store) | Initializes a new instance of the NESEventSource class. | |
| SQLite  |  |  | | --- | --- | | Method | Description | | Not Implemented |  | |  |

### Service Buses

IPublishingServiceBus : IServiceBus

|  |  |
| --- | --- |
| Method | Description |
| void Publish(params object[] messages) | Publishes the specified messages. |
| void Subscribe(Type messageType, Predicate<object> condition) | Subscribes the specified message type. |
| void Unsubscribe(Type messageType) | Unsubscribes the specified message type. |
| void Publish<TMessage>(Action<TMessage> messageBuilder)  where TMessage : class | \*Publishes the specified service bus. |
| void Subscribe<TMessage>()  where TMessage : class | \*Subscribes the specified service bus. |
| void Subscribe<TMessage>(Predicate<TMessage> condition)  where TMessage : class | \*Subscribes the specified service bus. |
| void Subscribe(Type messageType) | \*Subscribes the specified service bus. |
| void Unsubscribe<TMessage>()  where TMessage : class | \*Unsubscribes the specified service bus. |

IServiceBus : IServiceProvider

|  |  |
| --- | --- |
| Method | Description |
| TMessage CreateMessage<TMessage>(Action<TMessage> messageBuilder)  where TMessage : class | Creates the message. |
| IServiceBusCallback Send(IServiceBusEndpoint destination, params object[] messages) | Sends the specified destination. |
| void Reply(params object[] messages) | Replies the specified messages. |
| IServiceBusCallback Send<TMessage>( Action<TMessage> messageBuilder)  where TMessage : class | \*Sends the specified service bus. |
| IServiceBusCallback Send<TMessage>( string destination, Action<TMessage> messageBuilder)  where TMessage : class | \*Sends the specified service bus. |
| IServiceBusCallback Send<TMessage>(IServiceBusEndpoint destination, Action<TMessage> messageBuilder)  where TMessage : class | \*Sends the specified service bus. |
| IServiceBusCallback Send(params object[] messages) | \*Sends the specified service bus. |
| IServiceBusCallback Send(string destination, params object[] messages) | \*Sends the specified service bus. |
| void Reply<TMessage>(Action<TMessage> messageBuilder)  where TMessage : class | \*Replies the specified service bus. |
| T BehaveAs<T>()  where T : class, IServiceBus | ^\*Behaves as. |

IServiceBusCallback

|  |  |
| --- | --- |
| Method | Description |
| void Register<T>(Action<T> callback) | Registers the specified callback. |
| IAsyncResult Register(AsyncCallback callback, object state) | Registers the specified callback. |

IServiceBusEndpoint

|  |  |
| --- | --- |
| Method | Description |
| None |  |

IServiceBusHostRuntime

|  |  |
| --- | --- |
| Method | Description |
| void Initialize() | Initializes this instance. |
| void Open(IServiceBus bus) | Opens the specified bus. |
| void Close() | Closes this instance. |

IServiceMessageHandler<TMessage> where TMessage : class

|  |  |
| --- | --- |
| Method | Description |
| void Handle(TMessage message) | Handles the specified message. |

LiteralServiceBusEndpoint : IServiceBusEndpoint

|  |  |
| --- | --- |
| Method | Description |
| LiteralServiceBusEndpoint(string literal) | Initializes a new instance of the LiteralServiceBusEndpoint class. |
| string Value { get; } | Gets the value. |

ServiceBusExtensions

|  |  |
| --- | --- |
| Method | Description |
| void AddMessageHandlersByScan<TMessageHandler>(IServiceBus bus)  where TMessageHandler : class | Adds the message handlers by scan. |
| void AddMessageHandlersByScan(IServiceBus bus, Type handlerType) | Adds the message handlers by scan. |
| void AddMessageHandlersByScan(IServiceBus bus, Predicate<Type> predicate, params Assembly[] assemblies) | Adds the message handlers by scan. |
| void AddMessageHandler(IServiceBus bus, Type handlerType) | Adds the message handler. |

|  |  |
| --- | --- |
| IMPLEMENTATIONS |  |
| MTServiceBus | NServiceBus |
| RabbitMq | RhinoServiceBus |
| ZeroMQ | \*AppServiceBus |

### Service Caches

delegates

|  |  |
| --- | --- |
| Method | Description |
| void CacheEntryRemovedCallback(string name, object value) | Defines a delegate type for use by <see cref="M:Instinct.CacheCommand.OnRemoveCallback">CacheCommand.OnRemoveCallback</see>.  Used to provide notification of removal of an item from an instance of a <see cref="T:Instinct.Cache">Cache</see> object. |
| void CacheEntryUpdateCallback(string name, object value) | Defines a delegate type for use by <see cref="M:Instinct.CacheCommand.OnCreatedCallback">CacheCommand.OnCreatedCallback</see>.  Used to provide notification of removal of an item from an instance of a <see cref="T:Instinct.Cache">Cache</see> object. |
| object CacheItemBuilder(object tag, object[] values) | CacheItemBuilder |
| object CacheItemDependency(IServiceCache cache, IServiceCacheRegistration registration, object tag, object[] values) | CacheItemDependency |

CacheItemHeader

|  |  |
| --- | --- |
| Method | Description |
| string Item { get; set; } | Gets or sets the item. |
| object[] Values { get; set; } | Values |

CacheItemPolicy

|  |  |
| --- | --- |
| Method | Description |
| readonly CacheItemPolicy Default | Default |
| readonly CacheItemPolicy Infinite | Infinite |
| CacheItemPolicy() | Initializes a new instance of the CacheItemPolicy class. |
| CacheItemPolicy(int floatingAbsoluteMinuteTimeout) | Initializes a new instance of the CacheItemPolicy class. |
| CacheItemDependency Dependency { get; set; } | Gets or sets the object instance that contains dependency information that dictates when an item added to cache should be considered invalid. |
| DateTime AbsoluteExpiration { get; set; } | Gets or sets the DateTime instance that represent the absolute expiration of the item being added to cache. |
| TimeSpan FloatingAbsoluteExpiration { get; set; } | Gets or sets the DateTime instance that represent the absolute expiration of the item being added to cache. |
| TimeSpan SlidingExpiration { get; set; } | Gets or sets the TimeSpan instance that indicate the length of time to use for a sliding or dynamic expiration time. A sliding expiration is one that is constantly updated, based on the time value provided, whenever the underlying item having been cached is retrieved from cache. |
| CacheItemPriority Priority { get; set; } | Gets or sets the value of the CacheItemPriority enumeration associated with this CacheCommand instance. |
| CacheEntryUpdateCallback UpdateCallback { get; set; } | Gets or sets the on CacheItemCreatedCallback instance that is invoked whenever the cached item associated with this instance of CacheCommand has been created for cache. |
| CacheEntryRemovedCallback RemovedCallback { get; set; } | Gets or sets the on CacheItemRemovedCallback instance that is invoked whenever the cached item associated with this instance of CacheCommand has been removed from cache. |

CacheItemPriority

|  |  |
| --- | --- |
| Enum | Description |
| AboveNormal | Cached item has an above normal relative priority value. |
| BelowNormal | Cached item has a below normal relative priority value. |
| Default | Cached item has a default priority value. |
| High | Cached item has a high priority value. |
| Low | Cached item has a low priority value. |
| Normal | Cached item has a normal priority value. |
| NotRemovable | Cached item has a priority value indicating it cannot be removed from cache. |

DefaultServiceCacheRegistrationDispatcher

|  |  |
| --- | --- |
| Method | Description |
| T Get<T>(IServiceCache cache, IServiceCacheRegistration registration, object tag, object[] values) | Gets the specified cache. |
| void Send(IServiceCache cache, IServiceCacheRegistration registration, object tag, params object[] messages) | Sends the specified cache. |
| IEnumerable<T> Query<T>(IServiceCache cache, IServiceCacheRegistration registration, object tag, params object[] messages) | Queries the specified cache. |
| void Remove(IServiceCache cache, IServiceCacheRegistration registration) | Removes the specified cache. |

IDistributedServiceCache : IServiceCache

|  |  |
| --- | --- |
| Method | Description |
| None |  |

IServiceCache

|  |  |
| --- | --- |
| Method | Description |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

ITouchableCacheItem

|  |  |
| --- | --- |
| Method | Description |
| void Touch(object tag, string[] names) | Touches the specified tag. |
| object MakeDependency(object tag, string[] names) | Makes the dependency. |

ServiceCache

|  |  |
| --- | --- |
| Method | Description |
|  |  |
|  |  |

ServiceCacheByDispatcher

|  |  |
| --- | --- |
| Method | Description |
| ServiceCacheByDispatcher Empty | Empty |
| ServiceCacheByDispatcher(IServiceCacheRegistration registration, object[] values, CacheItemHeader header) | Initializes a new instance of the ServiceCacheByDispatcher struct. |
| IServiceCacheRegistration Registration | Registration |
| object[] Values | Values |
| CacheItemHeader Header | Header |

ServiceCacheForeignRegistration : IServiceCacheRegistration

|  |  |
| --- | --- |
| Method | Description |
| ServiceCacheForeignRegistration(string name) | Initializes a new instance of the ServiceCacheForeignRegistration class. |
| ServiceCacheForeignRegistration(string name, Type foreignType, string foreignName) | Initializes a new instance of the ServiceCacheForeignRegistration class. |
| string AbsoluteName { get; } | Gets the name of the absolute. |
| bool UseHeaders { get; } | Gets the use headers. |
| ServiceCacheRegistrar Registrar { get; } | Gets the registrar. |
| string Name { get; set; } | Gets or sets the name. |
| Type ForeignType { get; set; } | Gets or sets the foreign type. |
| string ForeignName { get; set; } | Gets or sets the name of the foreign. |
| void AttachRegistrar(ServiceCacheRegistrar registrar, string absoluteName) | Attaches the registrar. |

ServiceCacheOptions

|  |  |
| --- | --- |
| Enum | Description |
| ReturnsCachedValueOnRemove | ReturnsCachedValueOnRemove |
| UseDBNullWithRegistrations | UseDBNullWithRegistrations |

ServiceCacheRegistrar

|  |  |
| --- | --- |
| Method | Description |
| ServiceCacheRegistrar Get<TAnchor>() | Gets this instance. |
| ServiceCacheRegistrar Get(Type anchorType) | Gets the specified anchor type. |
| void RegisterAllBelow<T>() | Registers all below. |
| void RegisterAllBelow(Type type) | Registers all below. |
| Type AnchorType { get; } | Gets the type of the anchor. |
| void Register(string name, CacheItemBuilder builder, params string[] cacheTags) | Registers the specified registration. |
| void Register(string name, int minuteTimeout, CacheItemBuilder builder, params string[] cacheTags) | Registers the specified registration. |
| void Register(string name, CacheItemPolicy itemPolicy, CacheItemBuilder builder, params string[] cacheTags) | Registers the specified registration. |
| void Register(IServiceCacheRegistration registration) | Registers the specified registration. |
| void Clear() | Clears all registrations. |
| bool Contains(IServiceCacheRegistration registration) | Determines whether the specified key contains key. |
| bool Contains(string name) | Determines whether the specified name has been registered. |
| bool Remove(IServiceCacheRegistration registration) | Removes the specified registration. |
| bool Remove(string name) | Removes the specified registration. |
| bool TryGet(Type anchorType, out ServiceCacheRegistrar registrar, bool createIfRequired) | Tries the get. |
| bool TryGetValue(IServiceCacheRegistration registration, ref int recurses, out IServiceCacheRegistration foundRegistration) | Tries the get value. |
| bool TryGetValue(Type anchorType, string registrationName, ref int recurses, out IServiceCacheRegistration foundRegistration) | Tries the get value. |

IServiceCacheRegistration

|  |  |
| --- | --- |
| Method | Description |
| string Name { get; } | Gets the name. |
| string AbsoluteName { get; } | Gets the name of the absolute. |
| bool UseHeaders { get; } | Gets the use headers. |
| ServiceCacheRegistrar Registrar { get; } | Gets the registrar. |
| void AttachRegistrar(ServiceCacheRegistrar registrar, string absoluteName) | Attaches the registrar. |

ServiceCacheRegistration

|  |  |
| --- | --- |
| Method | Description |
|  |  |
|  |  |

ServiceCacheSettings

|  |  |
| --- | --- |
| Method | Description |
| ServiceCacheSettings() | Initializes a new instance of the ServiceCacheSettings class. |
| ServiceCacheSettings(ITouchableCacheItem touchable) | Initializes a new instance of the ServiceCacheSettings class. |
| string RegionMarker { get; set; } | Gets or sets the RegionMarker. |
| ServiceCacheOptions Options { get; set; } | Gets or sets the options. |
| ServiceCacheRegistration.IDispatcher RegistrationDispatcher { get; set; } | Gets or sets the registration dispatcher. |
| ITouchableCacheItem Touchable { get; set; } | Gets or sets the touchable. |
| bool TryGetRegion(ref string name, out string regionName) | Tries to get the region. |

|  |  |
| --- | --- |
| IMPLEMENTATIONS |  |
| Memcached | ServerAppFabric |
| \*StaticServiceCache | \*SystemServiceCache |

### Service Locators

Dd

Ddd

|  |  |
| --- | --- |
| IMPLEMENTATIONS |  |
| Autofac | CastleWindsor |
| Hiro | Munq |
| Ninject | Spring |
| StructureMap | Unity |
| \*MicroServiceLocator |  |

### Service Logs

IServiceLog

|  |  |
| --- | --- |
| Method | Description |
| string Name { get; } | Gets the name. |
| IServiceLog Get(string name) | Gets the specified name. |
| IServiceLog Get(Type type) | Gets the specified name. |
| void Write(ServiceLogLevel level, Exception ex, string s) | Writes the specified level. |
| IServiceLog Get<T>() | \*Gets the specified service. |
| void Fatal(string s) | \*Fatals the specified service. |
| void Fatal(Exception ex) | \*Fatals the specified service. |
| void Fatal(Exception ex, string s) | \*Fatals the specified service. |
| void FatalFormat(string format, params object[] args) | \*Fatals the specified service. |
| void FatalFormat(Exception ex, string format, params object[] args) | \*Fatals the specified service. |
| void Error(string s) | \*Errors the specified service. |
| void Error(Exception ex) | \*Errors the specified service. |
| void Error(Exception ex, string s) | \*Errors the specified service. |
| void ErrorFormat(string format, params object[] args) | \*Errors the specified service. |
| void ErrorFormat(Exception ex, string format, params object[] args) | \*Errors the specified service. |
| void Warning(string s) | \*Warnings the specified service. |
| void Warning(Exception ex) | \*Warnings the specified service. |
| void Warning(Exception ex, string s) | \*Warnings the specified service. |
| void WarningFormat(string format, params object[] args) | \*Warnings the specified service. |
| void WarningFormat(Exception ex, string format, params object[] args) | \*Warnings the specified service. |
| void Information(string s) | \*Informations the specified service. |
| void Information(Exception ex) | \*Informations the specified service. |
| void Information(Exception ex, string s) | \*Informations the specified service. |
| void InformationFormat(string format, params object[] args) | \*Informations the specified service. |
| void InformationFormat(Exception ex, string format, params object[] args) | \*Informations the specified service. |
| void Debug(string s) | \*Debugs the specified service. |
| void Debug(Exception ex) | \*Debugs the specified service. |
| void Debug(Exception ex, string s) | \*Debugs the specified service. |
| void DebugFormat(string format, params object[] args) | \*Debugs the specified service. |
| void DebugFormat(Exception ex, string format, params object[] args) | \*Debugs the specified service. |
| T BehaveAs<T>()  where T : class, IServiceLog | \*Behaves as. |

ServiceLog

|  |  |
| --- | --- |
| Method | Description |
| None |  |

ServiceLogLevel

|  |  |
| --- | --- |
| Enum | Description |
| None | None |
| Fatal | Fatal |
| Error | Error |
| Warning | Warning |
| Information | Information |
| Debug | Debug |

|  |  |
| --- | --- |
| IMPLEMENTATIONS |  |
| CommonLogging | Log4Net |
| NLog | \*ConsoleServiceLog |
| \*EventLogServiceLog | \*StreamServiceLog |
| \*TraceSourceServiceLog |  |

### Service Maps

IServiceMap

|  |  |
| --- | --- |
| Method | Description |
| None |  |

ServiceMap

|  |  |
| --- | --- |
| Method | Description |
| None |  |

|  |  |
| --- | --- |
| IMPLEMENTATIONS |  |
| AutoMapper |  |