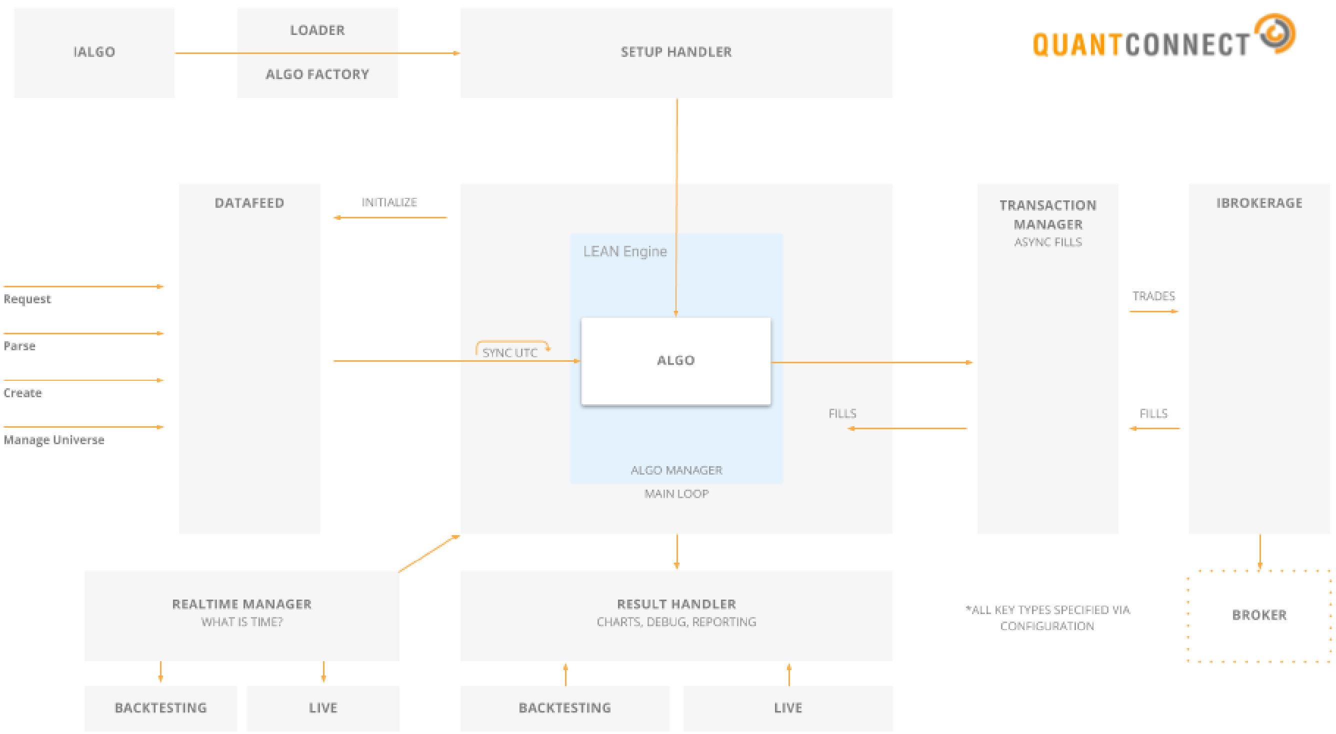


`config.json`

`IResultHandler`

`IDataFeed`

`ITransactionHandler`

`IRealtimeHandler`

`ISetupHandler`

`config.json` `Launcher`

-
-

```
$ git clone https://github.com/QuantConnect/Lean.git
$ cd Lean
```

`QuantConnect.Lean.sln`

`Project > Restore NuGet Packages`

`Run > Start Debugging`

`dll`

`Build > Build All`

```
$ cd Lean/Launcher/bin/Debug
$ dotnet QuantConnect.Lean.Launcher.dll
```

```
$ dotnet build QuantConnect.Lean.sln
```

```
$ cd Launcher/bin/Debug
$ dotnet QuantConnect.Lean.Launcher.dll
```

`ib-tws-dir` `ib-controller-dir`

`config.json`

`ib-port` `config.json`

`QuantConnect.Lean.sln`

`Build Menu -> Build Solution`

`F5`

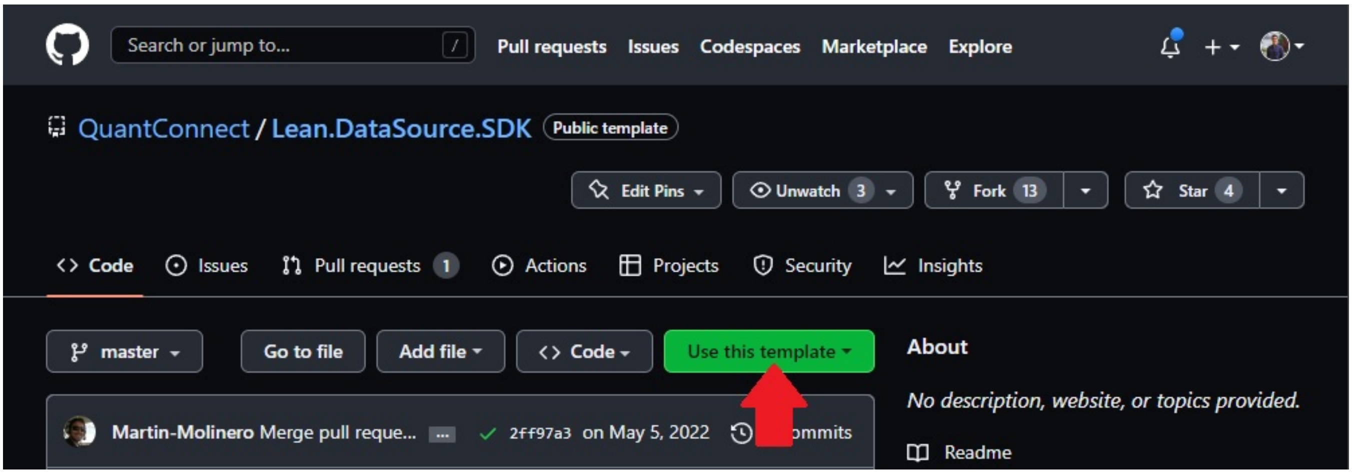
`GetSource`

`transportMedium` `SubscriptionTransportMedium.LocalFile` `format` `FileFormat.Csv`

`DataTimeZone`

`DataTimeZone`

RequiresMapping




```
$ git clone https://github.com/username/Lean.DataSource.<vendorNameDatasetName>.git
```

```
$ chmod +x ./renameDataset
```

```
$ renameDataset.sh
```

```
1997-01-01,905.2,941.4,905.2,939.55,38948210,978.21
1997-01-02,941.95,944,925.05,927.05,49118380,1150.42
1997-01-03,924.3,932.6,919.55,931.65,35263845,866.74
...
2014-07-24,7796.25,7835.65,7771.65,7830.6,117608370,6271.45
2014-07-25,7828.2,7840.95,7748.6,7790.45,153936037,7827.61
2014-07-28,7792.9,7799.9,7722.65,7748.7,116534670,6107.78
```

`SomeCustomProperty`

`Destination`

`ProtoMember`

`ProtoMember`

```py
self.AddData(Fred, Fred.LIBOR.OneWeekBasedOnUSD)
# Instead of
# self.AddData(Fred, "USD1WKD156N")
```

`config.Symbol.Value`

Reader

Symbol = config.Symbol EndTime

BaseData Value Time Value

Time

EndTime Time EndTime

```
public class VendorNameDatasetName : BaseData
{
    public override DateTimeZone DataTimeZone()
    {
        return DateTimeZone.Utc;
    }
}
```

using QuantConnect TimeZones DateTimeZone

TimeZones.Utc TimeZones.NewYork

SupportedResolutions

```
public class VendorNameDatasetName : BaseData
{
    public override List<Resolution> SupportedResolutions()
    {
        return DailyResolution;
    }
}
```

Resolution

DefaultResolution

DefaultResolution

```
public class VendorNameDatasetName : BaseData
{
    public override Resolution DefaultResolution()
    {
        return Resolution.Daily;
    }
}
```

`IsSparseData`

```
public class VendorNameDatasetName : BaseData
{
    public override bool IsSparseData()
    {
        return true;
    }
}
```

```
public class VendorNameDatasetName : BaseData
{
    public override bool RequiresMapping()
    {
        return true;
    }
}
```

`Clone`

```
public class VendorNameDatasetName : BaseData
{
    public override BaseData Clone()
    {
        return new VendorNameDatasetName
        {
            Symbol = Symbol,
            Time = Time,
            EndTime = EndTime,
            SomeCustomProperty = SomeCustomProperty,
        };
    }
}
```

`ToString`

```
public class VendorNameDatasetName : BaseData
{
    public override string ToString()
    {
        return $"{Symbol} - {SomeCustomProperty}";
    }
}
```

```
A R735QTJ8XC9X,A,17.19,109700,1885743,False,0.9904858,1
AA R735QTJ8XC9X,AA,71.25,513400,36579750,False,0.3992678,0.750075
AAB R735QTJ8XC9X,AAB,16.38,5000,81900,False,0.9902758,1
...
ZSEV R735QTJ8XC9X,ZSEV,10.5,800,8400,False,0.8981684,1
ZTR R735QTJ8XC9X,ZTR,9.56,102300,977988,False,0.0803037,3.97015016
ZVX R735QTJ8XC9X,ZVX,10,15600,156000,False,1,0.666667
```

`SomeCustomProperty` `SomeNumericProperty`

`Destination` `FlightPassengerCount`

`date`

`Reader`

`new Symbol(SecurityIdentifier.Parse(csv[0]), csv[1])`

`Symbol`

`Time` `date - Period`

```
public class VendorNameDatasetNameUniverse : BaseData
{
    public override DateTimeZone DataTimeZone()
    {
        return DateTimeZone.Utc;
    }
}
```

using QuantConnect TimeZones DateTimeZone

TimeZones.Utc TimeZones.NewYork

SupportedResolutions

```
public class VendorNameDatasetNameUniverse : BaseData
{
    public override List<Resolution> SupportedResolutions()
    {
        return DailyResolution;
    }
}
```

Resolution

DefaultResolution

DefaultResolution

```
public class VendorNameDatasetNameUniverse : BaseData
{
    public override Resolution DefaultResolution()
    {
        return Resolution.Daily;
    }
}
```

IsSparseData

```
public class VendorNameDatasetNameUniverse : BaseData
{
    public override bool IsSparseData()
    {
        return true;
    }
}
```

```
public class VendorNameDatasetNameUniverse : BaseData
{
    public override bool RequiresMapping()
    {
        return true;
    }
}
```

`Clone`

```
public class VendorNameDatasetNameUniverse : BaseData
{
    public override BaseData Clone()
    {
        return new VendorNameDatasetName
        {
            Symbol = Symbol,
            Time = Time,
            EndTime = EndTime,
            SomeCustomProperty = SomeCustomProperty,
        };
    }
}
```

`ToString`

```
public class VendorNameDatasetNameUniverse : BaseData
{
    public override string ToString()
    {
        return $"{Symbol} - {SomeCustomProperty}";
    }
}
```

`GetSource`

-

-
-

  `Main`
-

```
$ dotnet build .\DataProcessing\DataProcessing.csproj
```

`CLRImports`

-

`CLRImports`

```py
from CLRImports import *
```

-

```py
map_file_provider = LocalZipMapFileProvider()
map_file_provider.Initialize(DefaultDataProvider())
```

-

```py
sid = SecurityIdentifier.GenerateEquity(point_in_time_ticker,
    Market.USA, True, map_file_provider, csv_date)
```

```
$ python process.sample.py
```

-
-

`GetSource`


```
$ dotnet build .\DataProcessing\DataProcessing.csproj
```

GetSource

-
-

  Main

-

```
$ dotnet build .\DataProcessing\DataProcessing.csproj
```

CLRImports

-

CLRImports

```py
from CLRImports import *
```

-

```py
map_file_provider = LocalZipMapFileProvider()
map_file_provider.Initialize(DefaultDataProvider())
```

-

```py
sid = SecurityIdentifier.GenerateEquity(point_in_time_ticker,
    Market.USA, True, map_file_provider, csv_date)
```

-
-
-

```
$ git clone https://github.com/<username>/Lean.git
```

```
"algorithm-type-name": "<vendorNameDatasetName>Algorithm”,
"algorithm-language": "CSharp",
"algorithm-location": "QuantConnect.Algorithm.CSharp.dll",
```

```
"algorithm-type-name": "<vendorNameDatasetName>Algorithm”,
"algorithm-language": "Python",
"algorithm-location": "../../../Algorithm.Python/<vendorNameDatasetName>Algorithm.py",
```

`CreateNewInstance` `DataSource`

```
$ dotnet build tests/Tests.csproj
$ dotnet test tests/bin/Debug/net6.0/Tests.dll
```


`IBrokerageFactory`

`IBrokerage`

`ISymbolMapper`

`IBrokerageModel`

`IDataQueueHandler`

`IHistoryProvider`

`IDataDownloader`

`IFeeModel`

ISecurityTransactionModel

```
$ git clone https://github.com/username/Lean.Brokerages.<brokerageName>.git
```

```
$ chmod +x ./renameBrokerage
```

```
$ renameBrokerage.sh
```

BrokerageFactory

live-mode-brokerage

oanda-access-token oanda-account-id

<string,string>

BrokerageData Config

Config.Get("oanda-access-token") "oanda-access-token"

BitfinexBrokerageFactory

IBrokerageFactory

Composer.Instance.AddPart<IDataQueueHandler>(dataQueueHandler)

DataQueueHandler

`BrokerageName`

`BrokerageNameBrokerageModel` `XYZBrokerageModel`

`GetBrokerageModel`

`Brokerage`

`BaseWebsocketsBrokerage`

`CreateBrokerage`

`CreateBrokerage`

`job` `BrokerageData`

`BrokerageData`

live-<brokerageName>

live-<brokerageName>

```
// defines the 'live-brokerage-name' environment
"live-brokerage-name": {
  "live-mode": true,

  "live-mode-brokerage": "BrokerageName",

  "setup-handler": "QuantConnect.Lean.Engine.Setup.BrokerageSetupHandler",
  "result-handler": "QuantConnect.Lean.Engine.Results.LiveTradingResultHandler",
  "data-feed-handler": "QuantConnect.Lean.Engine.DataFeeds.LiveTradingDataFeed",
  "data-queue-handler": [ "QuantConnect.Lean.Engine.DataFeeds.Queues.LiveDataQueue" ],
  "real-time-handler": "QuantConnect.Lean.Engine.RealTime.LiveTradingRealTimeHandler",
  "transaction-handler":
"QuantConnect.Lean.Engine.TransactionHandlers.BacktestingTransactionHandler"
},
```

`brokerage-name` `"BrokerageName"`

environment

"live-brokerage-name"

`IBrokerage` `partial`

`IBrokerage`

`Brokerage`

`abstract`

`BaseWebsocketsBrokerage`

`partial`

`CreateBrokerage`

	BrokerageFactory BaseWebsocketBrokerage

`string Name`

`Name`

`void Connect()`

`Connect`

`BrokerageFactory`

`Connect`

`BrokerageMessageEvent`

`void Disconnect()`

`Disconnect`

`bool IsConnected`

`IsConnected`

`bool PlaceOrder(Order order)`

`PlaceOrder`

`PlaceOrder` `Order`

`BrokerOrder ConvertOrder(Order order)`

`BrokerageSymbolMapper`

`IsConnected`

`PlaceOrder`

`bool UpdateOrder(Order order)`

`UpdateOrder`

`UpdateOrder`

`bool CancelOrder(Order order)`

`bool UpdateOrder(Order order)`

`List<Order> GetOpenOrders()`

`List<Holding> GetAccountHoldings()`

```
List<Cash> GetCashBalance()

bool AccountInstantlyUpdated

IEnumerable<BaseData> GetHistory(HistoryRequest request)

bool AccountInstantlyUpdated
```

AvgDollarVolume

	$k = \begin{cases} \frac{100,000}{AvgDollarVolume}, & \text{if } AvgDollarVolume \neq 0 \\ 10, & \text{otherwise} \end{cases}$ min

`_fastTradingVolumeDiscountFactor`