Quant Research On-Premise

Securely deploy quantitative strategies on-premise with proprietary datasets.

Table of Content

Key Concepts

Key Concepts > Getting Started

Key Concepts

Getting Started

QUANTCONNECT LOCAL PLATFORM

Guide through creating a project, running your first backtest, and live algo trading in QuantConnect Local Platform.
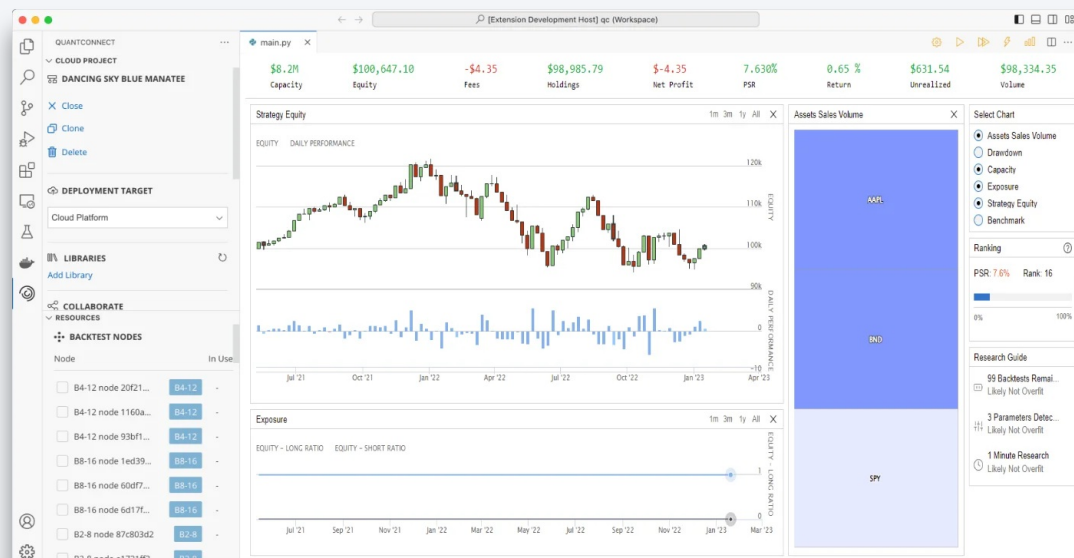
The Local Platform enables you to seamlessly develop quant strategies on-premise and in QuantConnect Cloud, getting the best of both environments. With Local Platform, you can harness your local version control, autocomplete, and coding tools with the full power of a scalable cloud at your finger tips. We intend to keep complete feature parity with our cloud environment, allowing you to harness cloud or local datasets to power on-premise quantitative research.

We encourage a hybrid "cloud + local" workflow, so you can use right tool for each stage of your development process. With the Local Platform, you can create, debug, and run projects on premise while using your own on-site tools. With the Cloud Platform you can deploy backtests at scale and harness our massive data library at low cost.

Follow these steps to create a new trading algorithm and backtest it in QuantConnect Cloud:

1. Install Local Platform .

2. Open ◁ Visual Studio Code.

3. In the Initialization Checklist panel, click **Login to QuantConnect** .
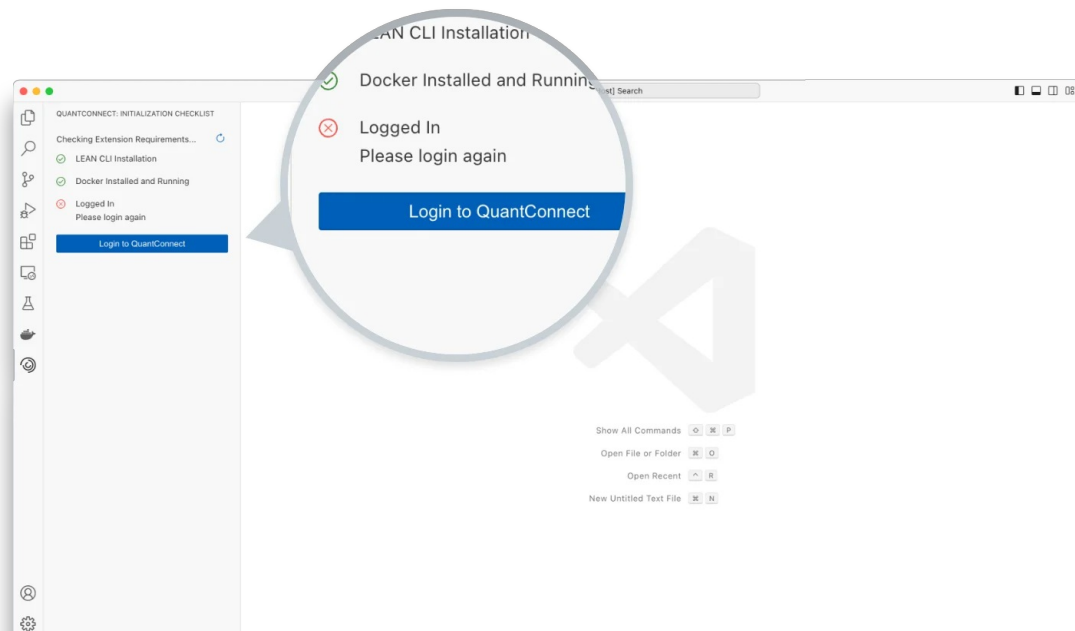
4. In the Visual Studio Code window, click **Open** .
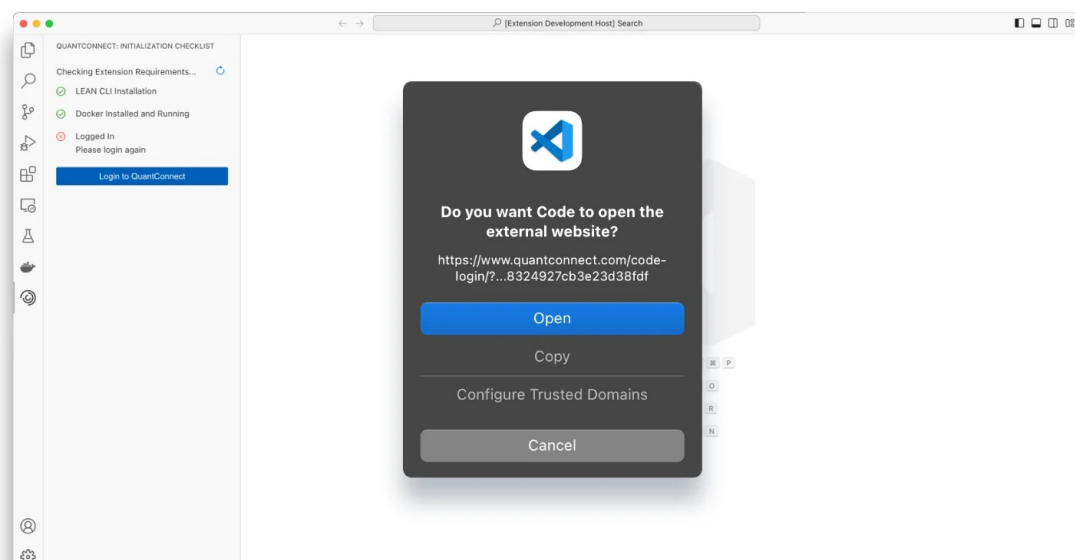
5. On the Code Extension Login page, click **Grant Access** .

6. In VS Code, in the Select Workspace panel, click **Pull Organization Workspace** .
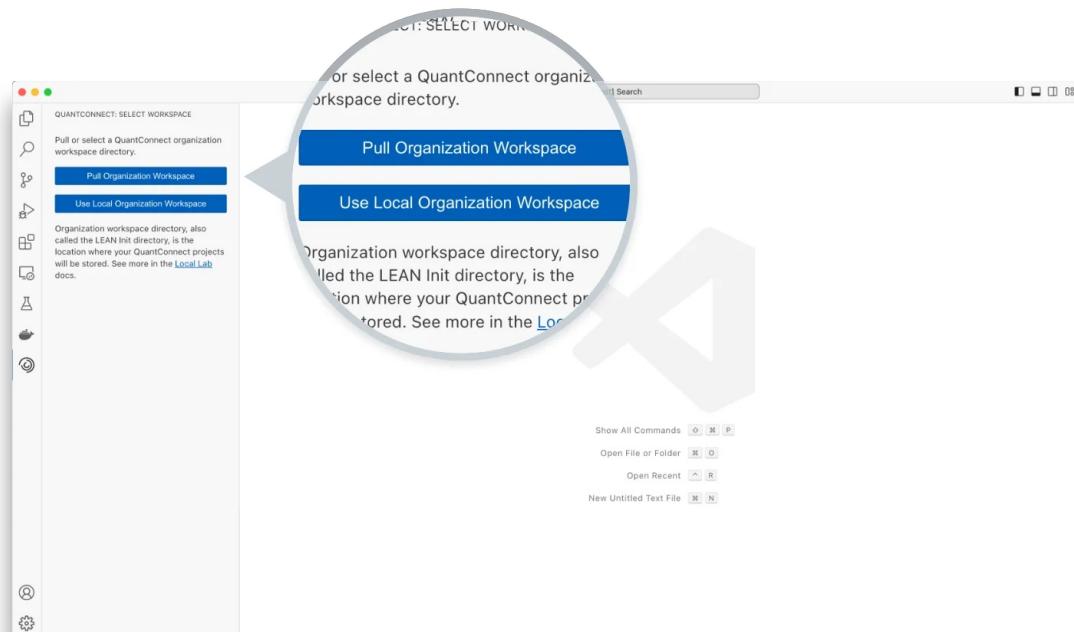
7. In the Pull QuantConnect Organization Workspace window, click the cloud workspace (organization) that you want to pull.
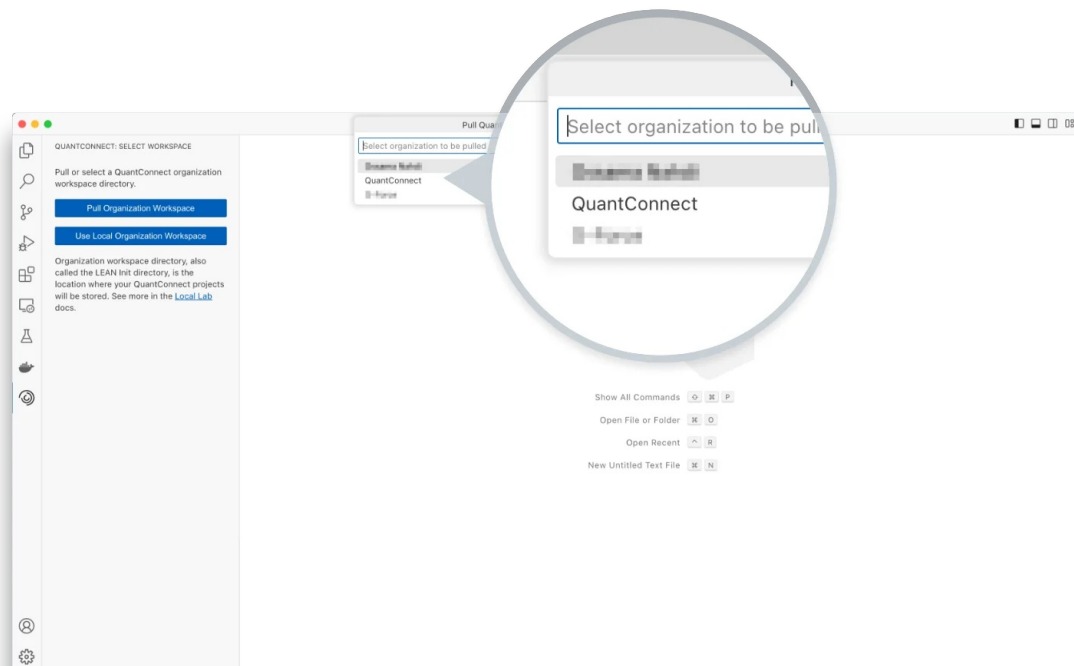
8. In the Pull QuantConnect Organization Workspace window, create a directory to serve as the organization workspace and then click **Select** .

If you are running Docker on Windows using the legacy Hyper-V backend instead of the new WSL 2 backend,

you need to enable file sharing for your temporary directories and for your organization workspace. To do so, open your Docker settings, go to **Resources > File Sharing** and add **C: / Users / / AppData / Local / Temp** and your organization workspace path to the list. Click **Apply & Restart** after making the required changes.

9. In the Open Project panel, click **Create Project** .

10. Enter the project name and then press **Enter** .
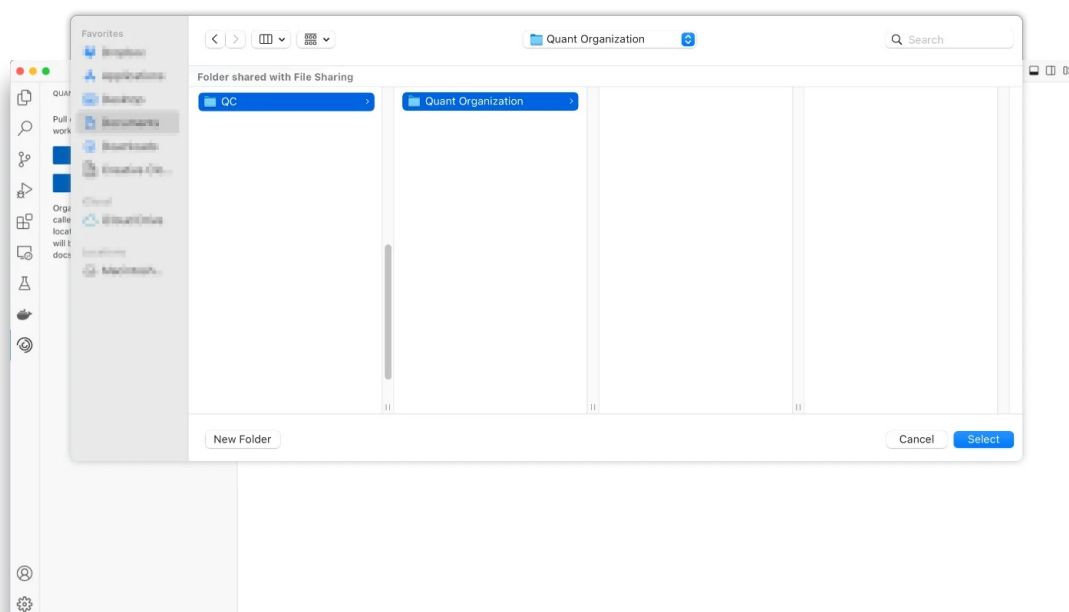
Congratulations! You just created your first local project.
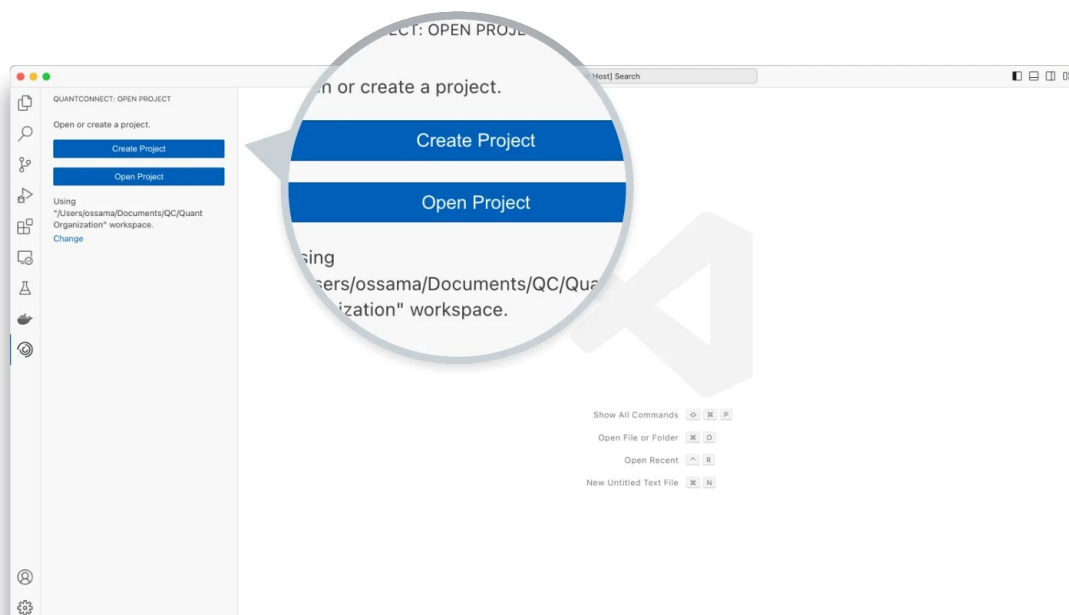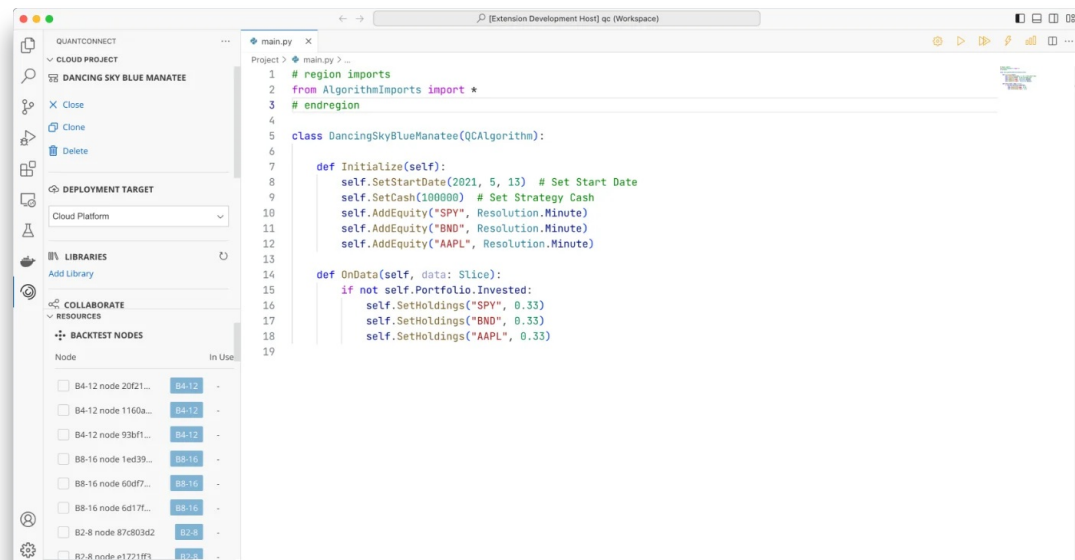
11. In the top-right corner of VS Code, click ⚙ **Build** and then click ▷ **Backtest** .

The backtest results page displays your algorithm's performance over the backtest period.
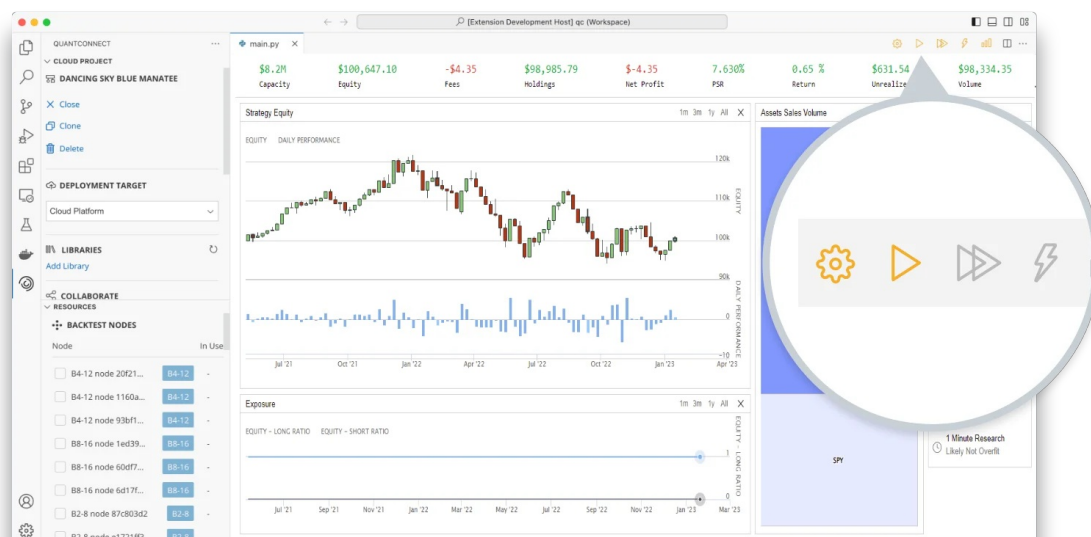
Key Concepts

Features

Introduction

There are 5 tiers of organizations and each tier has its own set of features on Local Platform. To accommodate the growth of your trading skills and business, you can adjust the tier of your organization at any time.

Hybrid Workflow

The Local Platform lets you run backtests, deploy research notebooks, and deploy live algorithms on your local machine and in QuantConnect Cloud. This gives you the best of both works where you can utilize your local hardware or our scalable cloud compute systems.

Version Control

The Local Platform syncs your local and cloud project files. If you pull your cloud projects to your local machine, you can use your own version control systems to track project changes.

Self-Sovereign Security

The Local Platform offers you the ability to take ownership of your project security. On the Institution tier, you can create local projects without pushing them to QuantConnect Cloud.

Custom LEAN Images

The latest master branch on the LEAN GitHub repository is the default engine branch that runs backtests, research notebooks, and live trading algorithms. The latest version of LEAN is generally the safest as it includes all bug fixes.

On-Premise Compute

The Local Platform enables you to run backtests, deploy research notebooks, and deploy live algorithms on your local hardware.

Coding

The following table shows the coding features of each platform:

Feature	Platform	
	Local Platform	**Cloud Platform**
Development Environment The tool you can use to edit project files	Any IDE	Cloud-hosted VS Code
Version Control Track file changes over time and easily revert mistakes	Your own git systems	Access historical project files through your backtest results
Anonymous Projects Create and edit local projects without syncing to QuantConnect Cloud	Self-sovereign security	Managed by QuantConnect
Custom LEAN Versions Build and run custom versions of LEAN		✓
Autocomplete Easy-to-use tool to speed up your development	✓	✓

Backtesting & Optimization

The following table shows the backtesting and optimization features of each platform:

Feature	Platform	
	Local Platform	**Cloud Platform**
Data Source Where does the data come from?	Licensed data	Provided by QuantConnect
Data Maintenance Who ensures the data is clean and ready?	Self-maintaned	QuantConnect Team
Compute Where is the hardware that runs backtests?	Your compute	QuantConnect Cloud compute
Proprietary Data Data that's not in the Dataset Market	Never leaves premise	Upload to cloud
Debugging Easy-to-use tool for solving coding errors	✓	✓

Live Trading

The following table shows the parameter optimization features of each platform:

Feature	Platform	
	Local Platform	Cloud Platform
Data Source Where does the data come from?	Licensed data	Provided by QuantConnect
Stability How stable is your live trading environment?	Your local setup	Stable co-located environment
Notifications SMS, email, Telegram, and webhooks		✓

Key Concepts

Deployment Targets

Introduction

The deployment target setting allows you to switch modes from local to cloud platforms, choosing where you run your algorithm. Local Platform targets are denoted with blue icons and Cloud Platform targets are denoted with gold icons.

Local

The Local Platform deployment target is your local machine. Follow these steps to set the deployment target of a project to Local Platform:

1. Create a project or open an exisiting one .
2. In the Project panel, click the **Deployment Target** field and then click **Local Platform** from the drop-down menu.

After you set the deployment target to Local Platform, the following icons are blue:

Icon	Name
⚙	**Build**
▷	**Backtest**
🐞▷	**Debug**
▷▷	**Optimize**
⚡	**Live Trading**
☰	**Backtest Results**

Cloud

The Cloud Platform deployment target is a collection of servers that the QuantConnect team manages. It's the same deployment target you use if you create projects, spin up research nodes, and deploy algorithms on the QuantConnect website. For more information about QuantConnect Cloud, including our infrastructure and usage quotas, see Cloud Platform .

Follow these steps to set the deployment target Cloud Platform:

1. Create a project or open an exisiting one .
2. In the Project panel, click the **Deployment Target** field and then click **Cloud** from the drop-down menu.

After you set the deployment target to Cloud Platform, the following icons are gold:

Icon	Name
⚙	**Build**
▷	**Backtest**
▷▷	**Optimize**
⚡	**Live Trading**
☰	**Backtest Results**

Comparison

Note the following differences between the Local Platform and Cloud Platform deployment targets.

Data

The Local Platform target uses your on-premise data . The Cloud Platform target has access to the data in the Dataset Market . Both targets enable you to import custom datasets .

Compute

The Local Platform target uses your on-premise hardware. The Cloud Platform target uses the QuantConnect Cloud compute. For more information about the backtesting, research, and live trading nodes in QuantConnect Cloud, see Resources .

Management

The Local Platform target is under the management of your on-premise team. The Cloud Platform target is under the management of the QuantConnect team.

Hardware Procurement

The Local Platform target uses your on-premise hardware, so it requires you to procure and management your own hardware. The Cloud Platform target uses the hardware in QuantConnect Cloud, so you don't need to procure or manage any of the hardware.

Installation

It takes 10 minutes to install Local Platform and about 1 hour to download the latest LEAN image. The Local Platform requires Docker. When you launch Local Platform, we scan for Docker and prompt you to install it to continue. We run all algorithms in a Docker container to avoid installing any dependencies on your computer.

Install on Windows

Install on macOS

Install on Linux

See Also

[LEAN CLI](#)

Installation

Install on Windows

Introduction

It takes 10 minutes to install Local Platform and about 1 hour to download the latest LEAN image. The Local Platform requires Docker. When you launch Local Platform, we scan for Docker and prompt you to install it to continue. We run all algorithms in a Docker container to avoid installing any dependencies on your computer.

Requirements

Windows systems must meet the following minimum requirements to run Local Platform:

- A 64-bit processor
- 4 GB RAM or more
- Windows 10, version 1903 or higher (released May 2019)
- Hardware virtualization enabled in the BIOS
- 20 GB hard drive or more

You need an internet connection for things like downloading updates, collaborating with team members, and syncing your projects with QuantConnect Cloud. Trading Firm and Institution organizations can run local backtests and research notebooks without an internet connection for up to 24 hours.

Install Docker

If you run the LEAN engine locally with QuantConnect Local Platform, LEAN executes in a Docker container. These Docker containers contain a minimal Linux-based operating system, the LEAN engine, and all the packages available to you on QuantConnect.com. It is therefore required to install Docker if you plan on using QuantConnect Local Platform to run the LEAN engine locally.

Follow these steps to install Docker:

1. Follow the Install Docker Desktop on Windows tutorial in the Docker documentation.

   As you install docker, enable WSL 2 features.

2. Restart your computer.

3. If Docker prompts you that the WSL 2 installation is incomplete, follow the instructions in the dialog shown by Docker to finish the WSL 2 installation.

4. Open PowerShell with adminstrator privledges and run:

```
$ wsl --update
```

By default, Docker doesn't automatically start when your computer starts. So, when you run the LEAN engine with QuantConnect Local Platform for the first time after starting your computer, you must manually start Docker. To automatically start Docker, open the Docker Desktop application, click **Settings > General** , and then enable the **Start Docker Desktop when you log in** check box.

Install Local Platform

Follow these steps to install Local Platform:

1. Install Docker .
2. Open a terminal and download the latest LEAN image.

```
$ docker pull quantconnect/lean
```

It takes about an hour to download the image. While it's downloading, continue to the next step. When you use Local Platform, it automatically pulls the latest LEAN image if your current version is more than a week old.

3. Install Visual Studio Code .
4. Install Local Platform .

If you open Visual Studio Code and it asks you to log in to QuantConnect, you successfully installed Local Platform.

Next Steps

After you install Local Platform, follow these steps:

1. Log in to your account .
2. Set up your first organization workspace .

3. Install the Python stubs for autocomplete.

Troubleshooting

The following sections explain how to solve some issues you may encounter while installing Local Platform.

Docker with WSL 2 Features

When you download Docker Desktop, you need to select the **Enable WSL 2 Features** check box. After you install Docker and restart your computer, if Docker prompts you that the WSL 2 installation is incomplete, follow the instructions in the dialog shown by Docker to finish the WSL 2 installation.

Docker Not Found

If you have Docker installed but the Local Platform can't detect it, update your **Executable Path: Docker** setting to be the path to your Docker executable.

LEAN CLI Account Syncronization

Local Platform and the LEAN CLI share your login credentials. If you log in to your account on Local Platform or the LEAN CLI, you log into that account for both Local Platform and the LEAN CLI.

Further Support

For further support with installing Local Platform, contact us .

Installation

Install on macOS

Introduction

It takes 10 minutes to install Local Platform and about 1 hour to download the latest LEAN image. The Local Platform requires Docker. When you launch Local Platform, we scan for Docker and prompt you to install it to continue. We run all algorithms in a Docker container to avoid installing any dependencies on your computer.

Requirements

Mac systems must meet the following minimum requirements to run Local Platform:

- Mac hardware from 2010 or newer with an Intel processor
- macOS 10.14 or newer (Mojave, Catalina, or Big Sur)
- 4 GB RAM or more
- 20 GB hard drive or more

You need an internet connection for things like downloading updates, collaborating with team members, and syncing your projects with QuantConnect Cloud. Trading Firm and Institution organizations can run local backtests and research notebooks without an internet connection for up to 24 hours.

Install Docker

If you run the LEAN engine locally with QuantConnect Local Platform, LEAN executes in a Docker container. These Docker containers contain a minimal Linux-based operating system, the LEAN engine, and all the packages available to you on QuantConnect.com. It is therefore required to install Docker if you plan on using QuantConnect Local Platform to run the LEAN engine locally.

To install Docker, see Install Docker Desktop on Mac in the Docker documentation.

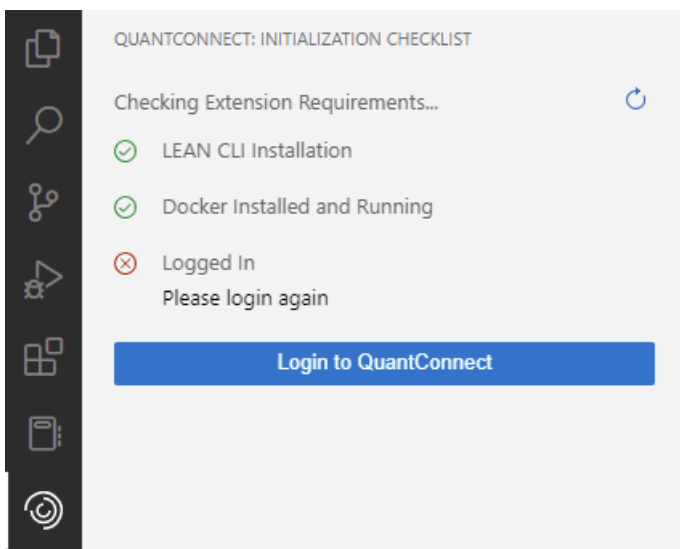### Install Local Platform

Follow these steps to install Local Platform:

1. Install Docker .
2. Open a terminal and download the latest LEAN image.

```
$ docker pull quantconnect/lean
```

It takes about an hour to download the image. While it's downloading, continue to the next step. When you use Local Platform, it automatically pulls the latest LEAN image if your current version is more than a week old.

3. Install Visual Studio Code .
4. Install Local Platform .

If you open Visual Studio Code and it asks you to log in to QuantConnect, you successfully installed Local Platform.

Next Steps

After you install Local Platform, follow these steps:

1. Log in to your account .
2. Set up your first organization workspace .
3. Install the Python stubs for autocomplete.

Troubleshooting

The following sections explain how to solve some issues you may encounter while installing Local Platform.

Docker Not Found

If you have Docker installed but the Local Platform can't detect it, update your **Executable Path: Docker** setting to be the path to your Docker executable.

LEAN CLI Account Syncronization

Local Platform and the LEAN CLI share your login credentials. If you log in to your account on Local Platform or the LEAN CLI, you log into that account for both Local Platform and the LEAN CLI.

Further Support

For further support with installing Local Platform, contact us .

Installation

Install on Linux

Introduction

It takes 10 minutes to install Local Platform and about 1 hour to download the latest LEAN image. The Local Platform requires Docker. When you launch Local Platform, we scan for Docker and prompt you to install it to continue. We run all algorithms in a Docker container to avoid installing any dependencies on your computer.

Requirements

Linux systems must meet the following minimum requirements to run Local Platform:

- 4 GB RAM or more
- 20 GB hard drive or more

You need an internet connection for things like downloading updates, collaborating with team members, and syncing your projects with QuantConnect Cloud. Trading Firm and Institution organizations can run local backtests and research notebooks without an internet connection for up to 24 hours.

Install Docker

If you run the LEAN engine locally with QuantConnect Local Platform, LEAN executes in a Docker container. These Docker containers contain a minimal Linux-based operating system, the LEAN engine, and all the packages available to you on QuantConnect.com. It is therefore required to install Docker if you plan on using QuantConnect Local Platform to run the LEAN engine locally.

To install, see Install Docker Desktop on Linux in the Docker documentation.

Install Local Platform
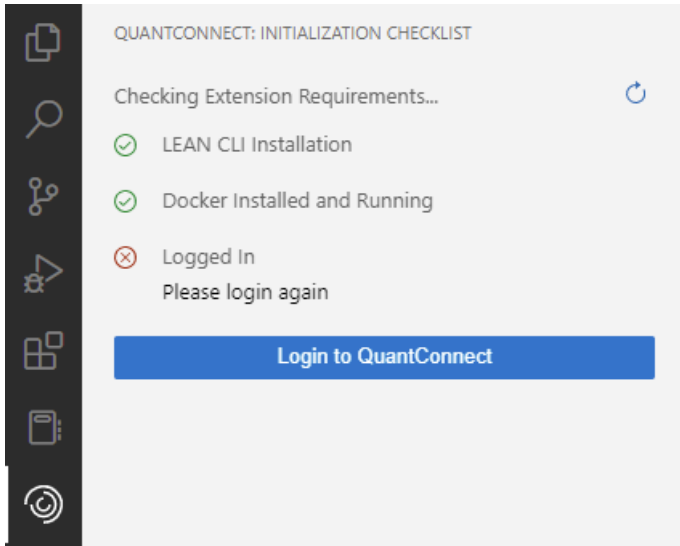
Follow these steps to install Local Platform:

1. Install Docker .
2. Open a terminal and download the latest LEAN image.

   ```
   $ docker pull quantconnect/lean
   ```

   It takes about an hour to download the image. While it's downloading, continue to the next step. When you use Local Platform, it automatically pulls the latest LEAN image if your current version is more than a week

old.

3. Install Visual Studio Code .
4. Install Local Platform .

If you open Visual Studio Code and it asks you to log in to QuantConnect, you successfully installed Local Platform.

Next Steps

After you install Local Platform, follow these steps:

1. Log in to your account .
2. Set up your first organization workspace .
3. Install the Python stubs for autocomplete.

Troubleshooting

The following sections explain how to solve some issues you may encounter while installing Local Platform.

Docker Not Found

If you have Docker installed but the Local Platform can't detect it, update your **Executable Path: Docker** setting to be the path to your Docker executable.

LEAN CLI Account Syncronization

Local Platform and the LEAN CLI share your login credentials. If you log in to your account on Local Platform or the LEAN CLI, you log into that account for both Local Platform and the LEAN CLI.

Further Support

For further support with installing Local Platform, contact us .

Development Environment

Development Environment

Authentication

Introduction

To use Local Platform, you need to grant it access to your QuantConnect account.

Log In

Follow these steps to log in to Local Platform:

1. Log in to the Algorithm Lab.
2. Start Docker Desktop.
3. Open Visual Studio Code.
4. In the left navigation menu, click the **QuantConnect** icon.
5. The Project panel checks the following requirements on your local machine. If any of the checks fail, see the related documentation.

- LEAN CLI is installed .
- Docker is installed and running .
- You are logged in to QuantConnect.

- In the Initialization Checklist panel, click **Login to QuantConnect** .
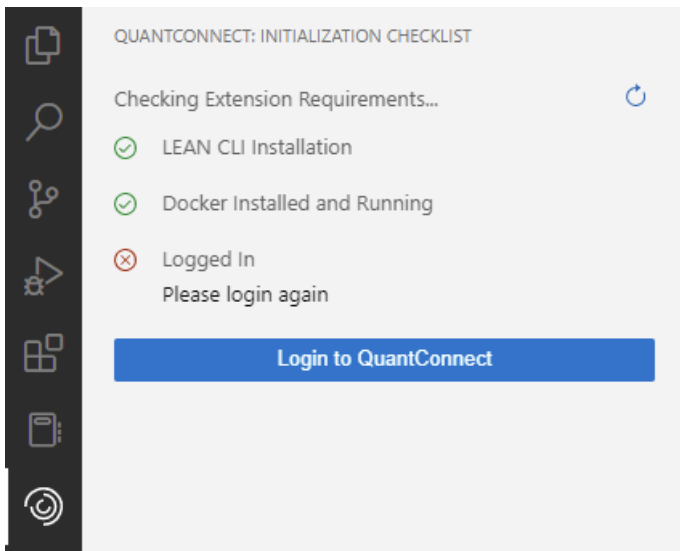
- In the Visual Studio Code window, click **Open** .
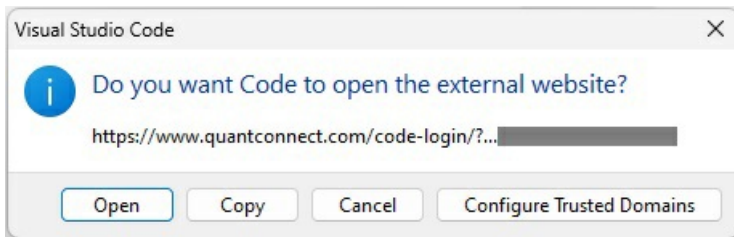
- On the Code Extension Login page, click **Grant Access** .

Log Out

Follow these steps to log out of Local Platform:

1. Open Visual Studio Code.

2. Press **F1** .

3. Enter **QuantConnect: Logout of QuantConnect** and then press **Enter** .
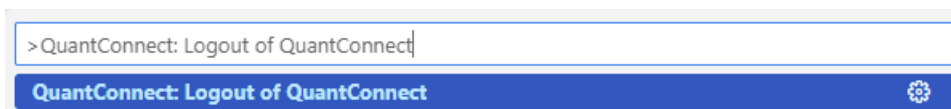
Troubleshooting

Local Platform and the LEAN CLI share your login credentials. If you log in to your account on Local Platform or the LEAN CLI, you log into that account for both Local Platform and the LEAN CLI.

Development Environment

Organization Workspaces

Introduction

An organization workspace is a directory that contains a **data** directory, a Lean configuration file, and all your project files from one of your organizations. You can have a separate organization workspace directory for each organization you're a member of on QuantConnect. These directories need a **data** directory and a Lean configuration file in order to run the LEAN engine on your local machine.

Pull Cloud Organization Workspaces

Follow these steps to pull one of your cloud organization workspaces and set it as your local organization workspace:

1. Log in to Local Platform .
2. In the left navigation menu, click the ⊚ **QuantConnect** icon.
3. In the Select Workspace panel, click **Pull Organization Workspace** .
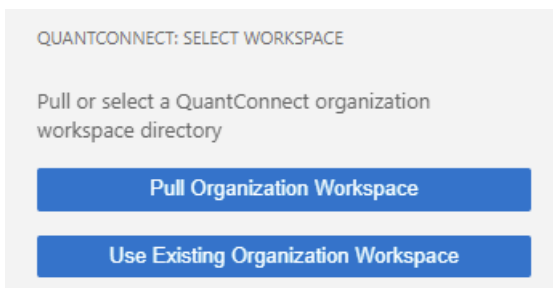
   QUANTCONNECT: SELECT WORKSPACE

   Pull or select a QuantConnect organization
   workspace directory

   Pull Organization Workspace

   Use Existing Organization Workspace

4. In the Pull QuantConnect Organization Workspace window, click the cloud workspace (organization) that you want to pull.

   Pull QuantConnect Organization Workspace

   |Select organization to be pulled|
   Quant Organization
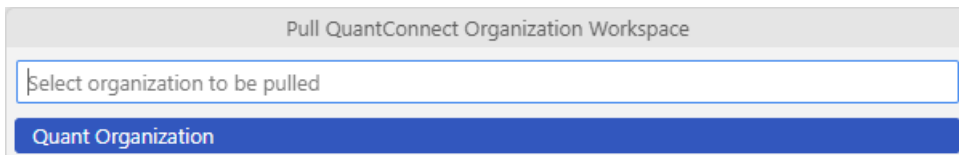
5. In the Pull QuantConnect Organization Workspace window, create a directory to serve as the organization workspace and then click **Select** .
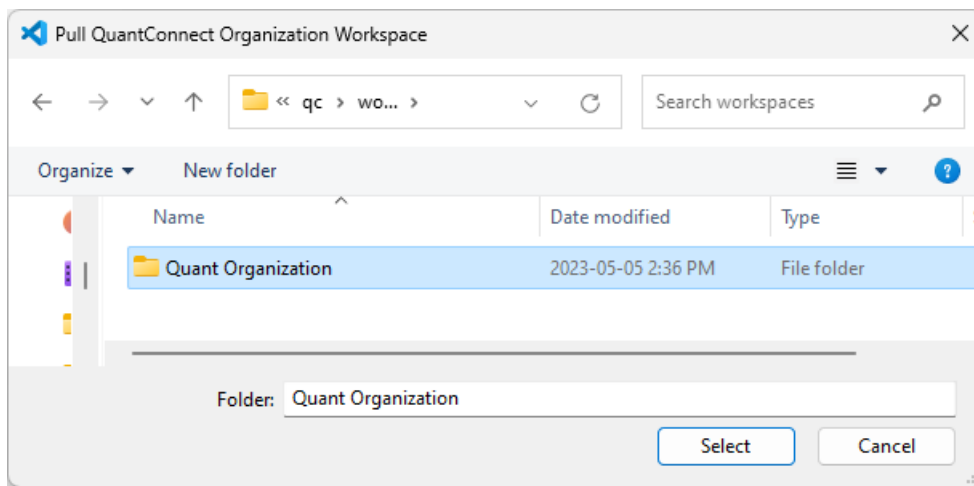
It takes a few minutes to create a new organization workspace directory and populate it with the the initial file structure . After the organization workspace is populated with the initial file structure, it pulls your cloud project files .

If you are running Docker on Windows using the legacy Hyper-V backend instead of the new WSL 2 backend, you need to enable file sharing for your temporary directories and for your organization workspace. To do so, open your Docker settings, go to **Resources > File Sharing** and add **C: / Users / <username> / AppData / Local / Temp** and your organization workspace path to the list. Click **Apply & Restart** after making the required changes.

Change Organization Workspaces

Follow these steps to change organization workspaces:

1. Log in to Local Platform .
2. In the left navigation menu, click the **QuantConnect** icon.
3. If a project is already open, close it .
4. In the Open Project panel, click **Change** .
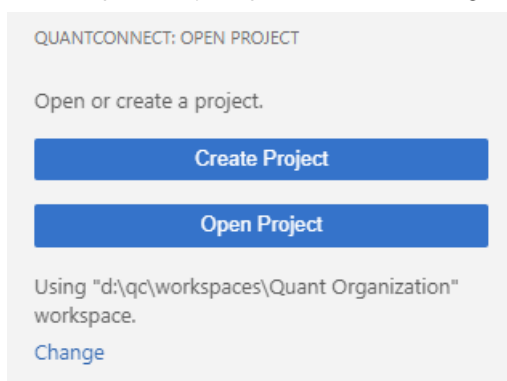
5. Pull a cloud workspace .

Directory Structure

The organization workspace directory initially has following structure:

```
.
├── data/
│   ├── alternative/
│   ├── crypto/
│   ├── equity/
│   ├── ...
│   ├── market-hours/
│   ├── option/
│   ├── symbol-properties/
│   └── readme.md
└── lean.json
```

These files contain the following content:

File/Directory	Description
data /	This directory contains the local data that LEAN uses to run locally. This directory is comes with sample data from the QuantConnect/Lean repository . As you download additional data from the dataset market, it's stored in this directory. Each organization workspace has its own **data** directory because each organization has its own data licenses.
lean.json	This file contains the Lean configuration that is used when running the LEAN engine locally. The configuration is stored as JSON with support for both single-line and multiline comments. The Lean configuration file is based on the Launcher/config.json file from the Lean repository. When you create a new organization workspace, the latest version of this file is downloaded and stored on your local drive.

As you add projects , the project files are added to your organization workspace directory. If you create and use shared libraries in your projects, the library files are added to a **Library** directory in your organization workspace.

Development Environment Configuration

Introduction

The Local Platform is configured by extension settings in VS Code and by the LEAN Engine settings. Change these settings at any time to suit your needs.

Extension Settings

Follow these steps to view the settings of the Local Platform extension:

1. Open VS Code.
2. In the top navigation bar, click **File > Preferences > Settings** .
3. On the Settings page, in the left navigation menu, click **Extensions > QuantConnect** .

The following table describes each setting:

Name	Description
Executable Path: Docker	A path to the Docker installation you want to use.
Executable Path: Lean	A path to the LEAN CLI executable you want to use.
Lean: Init	A path to the current organization workspace.
Sync: Local And Cloud Projects	**Yes** to synchronize cloud and local projects. Otherwise, **No** . **No** is only available for Institution organizations.
User: Preferred Language	The programming language to use when creating new projects. **Py** for Python or **C#** for C#.

LEAN Settings

The Lean configuration contains settings for locally running the LEAN engine. This configuration is created in the **lean.json** file when you pull or create an organization workspace . The configuration is stored as JSON, with support for both single-line and multiline comments.

The Lean configuration file is based on the **Launcher / config.json** file from the Lean GitHub repository. When you pull or create an organization workspace, the latest version of this file is downloaded and stored in your organization workspace. Before the file is stored, some properties are automatically removed because the Local

Platform automatically sets them.

The Local Platform can update most of the values of the **lean.json** file. The following table shows the configuration settings that you need to manually adjust in the **lean.json** file if you want to change their values:

Name	Description	Default
show-missing-data-logs	Log missing data files. This is useful for debugging.	true
maximum-warmup-history-days-look-back	The maximum number of days of data the history provider will provide during warm-up in live trading. The history provider expects older data to be on disk.	5
maximum-data-points-per-chart-series	The maximum number of data points you can add to a chart series in backtests.	4000

Development Environment

Autocomplete

Introduction

Intellisense is a GUI tool in your code files that shows auto-completion options and presents the members that are accessible from the current object. The tool works by searching for the statement that you're typing, given the context. You can use Intellisense to auto-complete method names and object attributes. When you use it, a pop-up displays in the IDE with the following information:

- Member type
- Member description
- The parameters that the method accepts (if the member is a method)

Use Intellisense to speed up your algorithm development. It works with all of the default class members in Lean, but it doesn't currently support class names or user-defined objects.

Install Python Stubs

Before you use autocomplete, you may need to run the following command in a terminal to get the latest Python stubs:

```
$ pip install --upgrade quantconnect-stubs
```
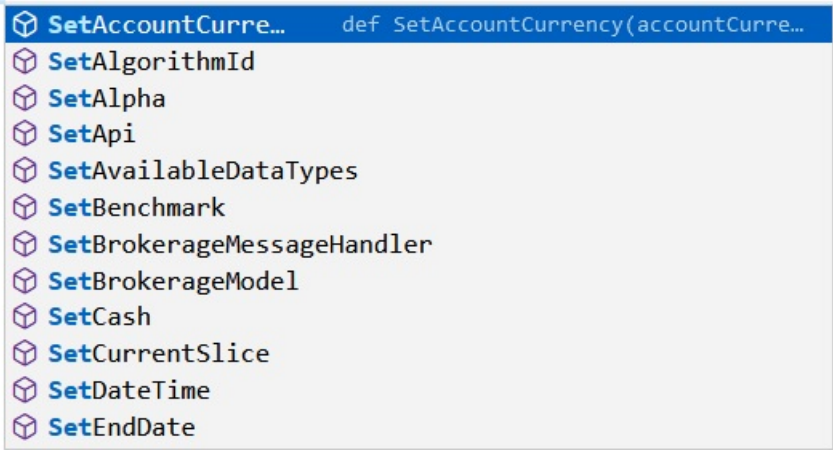
Use Autocomplete

Follow these steps to use autocomplete:

1. Open a project .
2. Type the first few characters of a variable, function, class, or class member that you want to autocomplete (for example, `self.Set` or `SimpleMovingAverage.Upda`).
3. Press **CTRL+Space** .

   If there are class members that match the characters you provided, a list of class members displays.

```
10    💡   │    self.Set
11                ⬡ SetAccountCurre...      def SetAccountCurrency(accountCurre...
12                ⬡ SetAlgorithmId
13                ⬡ SetAlpha
14                ⬡ SetApi
15                ⬡ SetAvailableDataTypes
16                ⬡ SetBenchmark
17                ⬡ SetBrokerageMessageHandler
18                ⬡ SetBrokerageModel
19                ⬡ SetCash
20                ⬡ SetCurrentSlice
21                ⬡ SetDateTime
22                ⬡ SetEndDate
```

4. Select the class member that you want to autocomplete.

The rest of the class member name is automatically written in the code file.

Development Environment

Collaboration

Introduction

Project collaboration is a real-time coding experience with other members of your team. Collaborating can speed up your development time. By working with other members in an organization, members within the organization can specialize in different parts of the project. On Local Platform, you can collaborate with your remote team members.

Video Demo

When there are multiple people working on the same project, the cursor of each member is visible in the IDE and all file changes occur in real-time for everyone. The following video demonstrates the collaboration feature:
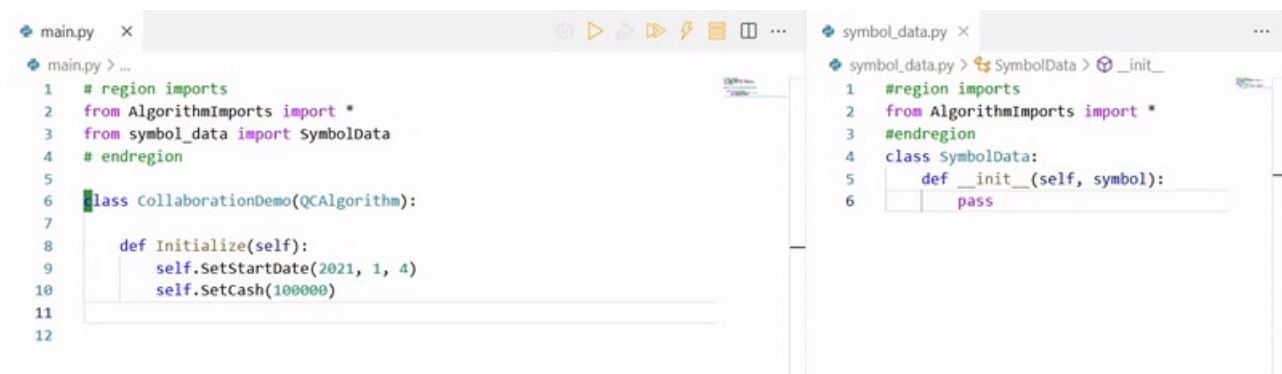
Add Team Members

You need to own the project to add team members to it.

Follow these steps to add team members to a project:

1. Open the project .
2. In the left navigation menu, click the ⦿ **QuantConnect** icon.
3. In the Collaborate section of the Project panel, click **Add Collaborator** .
4. Click the **Select User...** field and then click a member from the drop-down menu.
5. If you want to give the member live control of the project, select the **Live Control** check box.
6. Click **Add User** .

   The member you add receives an email with a link to the project.

If the project has a shared library , the collaborator can access the project, but not the library. To grant them access to the library, add them as a collaborator to the library project.

Collaborator Quotas

The number of members you can add to a project depends on your organization's tier . The following table shows the number of collaborators each tier can have per project:

Tier	Collaborators per Project
Free	Unsupported
Quant Researcher	Unsupported
Team	10
Trading Firm	Unlimited
Institution	Unlimited

Toggle Live Control

You need to have added a member to the project to toggle their live control of the project.

Follow these steps to enable and disable live control for a team member:

1. Open the project .
2. In the left navigation menu, click the ◎ **QuantConnect** icon.
3. In the Collaborate section of the Project panel, click the profile image of the team member.
4. Click the **Live Control** check box.
5. Click **Save Changes** .

Remove Team Members

Follow these steps to remove a team member from a project you own:

1. Open the project .
2. In the left navigation menu, click the ◎ **QuantConnect** icon.
3. In the Collaborate section of the Project panel, click the profile image of the team member.
4. Click **Remove User** .

To remove yourself as a collaborator from a project you don't own, delete the project .

Development Environment

LEAN Engine Versions

Introduction

The latest master branch on the LEAN GitHub repository is the default engine branch that runs backtests, research notebooks, and live trading algorithms. The latest version of LEAN is generally the safest as it includes all bug fixes.

Change Branches

Follow these steps to change the LEAN engine branch that runs your backtests and live trading algorithms:

1. Open a project .
2. In the left navigation menu, click the 🎯 **QuantConnect** icon.
3. In the Project panel, click the **LEAN Engine** field and then click a branch from the drop-down menu.
4. (Optional) Click **About Version** to display the branch description.
5. If you want to always use the master branch, select the **Always use Master Branch** check box.
6. Click **Select** .

Changing the Lean engine branch only affects the current project. If you create a new project , the new project will use the master branch by default.

Custom Branches

To create and use custom versions of LEAN, see Custom Docker Images .

Development Environment Synchronization

Introduction

Unless you are working on an anonymous project, Local Platform automatically syncs your local project files with QuantConnect Cloud. Every time you save a file, Local Platform saves the changes in your local project and in the cloud version of the project.

Anonymous Projects

Anonymous projects are projects that are on your local machine and not synced with QuantConnect Cloud. These types of projects are only available for members in Institution organizations. Anonymous projects provide organizations the opportunity to take ownership of their projects security.

Supported File Types
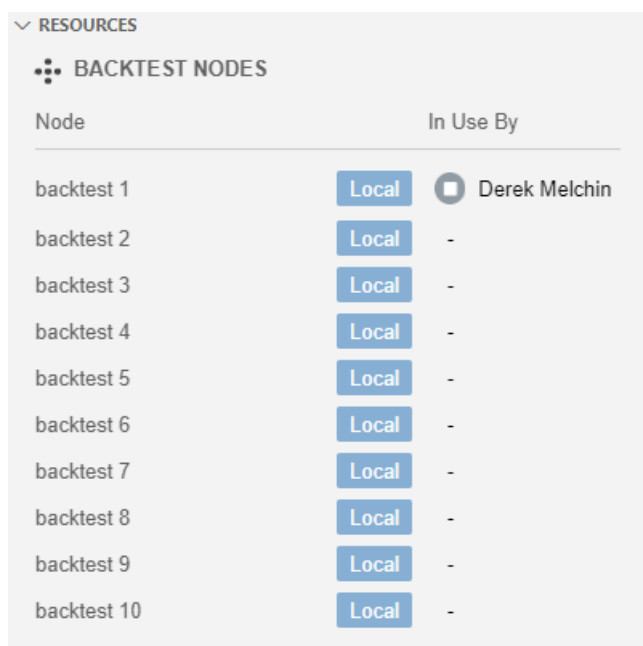
When you save your local projects and push them to QuantConnect Cloud, it only pushes the Python, C#, and notebook files in your project. Projects can contain many other file types like **json** , **csv** , and **html** , but Local Platform only pushes your **py** , **cs** , and **ipynb** files.

Development Environment

Resource Management

Introduction

The Resources panel shows all of the backtest, research, and live trading nodes that Local Platform can use or is already using.

The **In Use By** column displays the owner and name of the project using the node.

View Resources

To view the Resources panel, open a project and then, in the left navigation menu, click the ◎ **QuantConnect** icon. The Resources panel is at the bottom of the Project panel.

Stop Nodes

To stop a node, open the Resources panel and then click the **stop** button next to the node.

Development Environment

Packages and Libraries

Introduction

Libraries (or packages) are third-party software that you can use in your projects. You can use many of the available open-source libraries to complement the classes and methods that you create. Libraries reduce your development time because it's faster to use a pre-built, open-source library than to write the functionality. Libraries can be used in backtesting, research, and live trading. The environments support various libraries for machine learning, plotting, and data processing. As members often request new libraries, we frequently add new libraries to the underlying docker image that runs the Lean engine.

This feature is primarily for Python algorithms as not all Python libraries are compatible with each other. We've bundled together different sets of libraries into distinct environments. To use the libraries of an environment, set the environment in your project and add the relevant `import` statement of a library at the top of your file.

Set Environment

Follow these steps to set the library environment:

1. Open a project .
2. In the left navigation menu, click the ⊚ **QuantConnect** icon.
3. In the Project panel, click the **Python Foundation** field and then select an environment from the drop-down menu.

Default Environment

The default environment supports the following libraries:

```
                                                               PY
# Name                     Version
absl-py                    1.4.0
adagio                     0.2.4
aesara                     2.8.12
aiodns                     3.0.0
aiohttp                    3.8.4
aiohttp-cors               0.7.0
aiosignal                  1.3.1
alabaster                  0.7.13
ale-py                     0.7.4
alembic                    1.10.3
alpaca-trade-api           0.26
alphalens-reloaded         0.4.3
altair                     4.2.2
ansi2html                  1.8.0
antlr4-python3-runtime     4.11.1
anyio                      3.6.2
```

```
appdirs                        1.4.4
arch                           5.3.1
argon2-cffi                    21.3.0
argon2-cffi-bindings           21.2.0
arviz                          0.15.1
astropy                        5.2.1
asttokens                      2.2.1
astunparse                     1.6.3
async-timeout                  4.0.2
asyncio-nats-client            0.11.5
attrs                          23.1.0
autograd                       1.5
autokeras                      1.1.0
autoray                        0.6.3
AutoROM                        0.4.2
AutoROM.accept-rom-license     0.6.1
ax-platform                    0.3.0
Babel                          2.12.1
backcall                       0.2.0
bayesian-optimization          1.4.2
beautifulsoup4                 4.11.2
bleach                         6.0.0
blessed                        1.20.0
blis                           0.7.9
blosc2                         2.0.0
bokeh                          2.4.3
boltons                        23.0.0
botorch                        0.8.2
Bottleneck                     1.3.7
brotlipy                       0.7.0
cachetools                     5.3.0
captum                         0.6.0
catalogue                      2.0.8
catboost                       1.1.1
category-encoders              2.6.0
ccxt                           3.0.72
certifi                        2022.12.7
cffi                           1.15.1
chardet                        5.1.0
charset-normalizer             2.0.4
check-shapes                   1.0.0
click                          7.1.2
clikit                         0.6.2
cloudpickle                    2.2.1
cmaes                          0.9.1
cmdstanpy                      1.1.0
colorama                       0.4.6
colorcet                       3.0.1
colorful                       0.5.5
colorlog                       6.7.0
colorlover                     0.3.0
colour                         0.1.5
comm                           0.1.3
commonmark                     0.9.1
conda                          23.3.1
conda-content-trust            0.1.3
conda-package-handling         2.0.2
conda_package_streaming        0.7.0
confection                     0.0.4
cons                           0.4.5
contourpy                      1.0.7
convertdate                    2.4.0
copulae                        0.7.7
copulas                        0.8.0
cramjam                        2.6.2
crashtest                      0.3.1
creme                          0.6.1
cryptography                   38.0.4
cufflinks                      0.17.3
cvxopt                         1.3.0
cvxpy                          1.3.0
cycler                         0.11.0
cymem                          2.0.7
Cython                         0.29.33
darts                          0.23.1
dash                           2.9.3
dash-core-components           2.0.0
```

```
dash-cytoscape              0.3.0
dash-html-components        2.0.0
dash-table                  5.0.0
dask                        2023.4.0
deap                        1.3.3
debugpy                     1.5.1
decorator                   5.1.1
defusedxml                  0.7.1
Deprecated                  1.2.13
dgl                         1.0.1
dill                        0.3.6
dimod                       0.12.3
diskcache                   5.6.1
distlib                     0.3.6
distributed                 2021.4.1
dm-tree                     0.1.8
docutils                    0.19
DoubleML                    0.5.2
dtreeviz                    2.2.1
dtw-python                  1.3.0
dwave-cloud-client          0.10.3
dwave-drivers               0.4.4
dwave-greedy                0.3.0
dwave-hybrid                0.6.9
dwave-inspector             0.3.0
dwave-inspectorapp          0.3.0
dwave-neal                  0.6.0
dwave-networkx              0.8.12
dwave-ocean-sdk             6.1.1
dwave-preprocessing         0.5.3
dwave-samplers              1.0.0
dwave-system                1.18.0
dwave-tabu                  0.5.0
dwavebinarycsp              0.2.0
dx                          0.1.22
ecos                        2.0.12
EMD-signal                  1.4.0
empyrical                   0.5.5
empyrical-reloaded          0.5.9
en-core-web-md              3.5.0
en-core-web-sm              3.5.0
entrypoints                 0.4
ephem                       4.1.4
et-xmlfile                  1.1.0
etuples                     0.3.8
exceptiongroup              1.1.1
exchange-calendars          4.2.6
executing                   1.2.0
ezyrb                       1.3.0.post2304
fastai                      2.7.11
fastai2                     0.0.30
fastcore                    1.5.29
fastdownload                0.0.7
fasteners                   0.18
fastjsonschema              2.16.3
fastparquet                 2023.2.0
fastprogress                1.0.3
fasttext                    0.9.2
featuretools                0.23.1
filelock                    3.12.0
findiff                     0.9.2
finrl                       0.3.1
FixedEffectModel            0.0.5
Flask                       2.0.3
flatbuffers                 23.3.3
fonttools                   4.39.3
fracdiff                    0.9.0
frozendict                  2.3.7
frozenlist                  1.3.3
fs                          2.4.16
fsspec                      2023.4.0
fst-pso                     1.8.1
fugue                       0.8.3
fugue-sql-antlr             0.1.6
future                      0.18.3
fuzzy-c-means               1.6.3
FuzzyTM                     2.0.5
```

```
gast                        0.4.0
gensim                      4.3.0
gevent                      22.10.2
gluonts                     0.12.3
google-api-core             2.11.0
google-auth                 2.17.3
google-auth-oauthlib        0.4.6
google-pasta                0.2.0
googleapis-common-protos    1.59.0
gpflow                      2.7.0
gplearn                     0.4.2
gpustat                     1.1
GPUtil                      1.4.0
gpytorch                    1.9.1
graphviz                    0.20.1
greenlet                    2.0.2
grpcio                      1.54.0
gym                         0.21.0
Gymnasium                   0.26.3
gymnasium-notices           0.0.1
h2o                         3.40.0.1
h5netcdf                    1.1.0
h5py                        3.8.0
hijri-converter             2.2.4
hmmlearn                    0.2.8
holidays                    0.23
holoviews                   1.15.4
homebase                    1.0.1
hopcroftkarp                1.2.5
html5lib                    1.1
httpstan                    4.9.1
hurst                       0.0.5
hvplot                      0.8.2
hyperopt                    0.2.7
idna                        3.4
iisignature                 0.24
ijson                       3.2.0.post0
imageio                     2.27.0
imagesize                   1.4.1
imbalanced-learn            0.10.1
importlib-metadata          4.13.0
importlib-resources         5.12.0
iniconfig                   2.0.0
injector                    0.20.1
interpret                   0.3.0
interpret-core              0.3.2
ipykernel                   6.22.0
ipython                     8.12.0
ipython-genutils            0.2.0
ipywidgets                  8.0.4
itsdangerous                2.1.2
jax                         0.4.4
jaxlib                      0.4.4
jedi                        0.18.2
Jinja2                      3.1.2
joblib                      1.2.0
jqdatasdk                   1.8.11
json5                       0.9.11
jsonpatch                   1.32
jsonpointer                 2.0
jsonschema                  4.17.3
jupyter                     1.0.0
jupyter-bokeh               3.0.5
jupyter-console             6.6.3
jupyter-dash                0.4.2
jupyter-resource-usage      0.7.2
jupyter-server              1.24.0
jupyter_client              8.2.0
jupyter_core                5.3.0
jupyterlab                  3.4.4
jupyterlab-pygments         0.2.2
jupyterlab-widgets          3.0.7
jupyterlab_server           2.22.1
keract                      4.5.1
keras                       2.11.0
keras-nlp                   0.4.1
keras-rl                    0.4.2
```

```
keras-tuner                   1.3.5
kiwisolver                    1.4.4
kmapper                       2.0.1
korean-lunar-calendar         0.3.1
kt-legacy                     1.0.5
langcodes                     3.3.0
lark                          1.1.5
lazy_loader                   0.2
lazypredict                   0.2.12
libclang                      16.0.0
lightgbm                      3.3.5
lime                          0.2.0.1
line-profiler                 4.0.2
linear-operator               0.3.0
littleutils                   0.2.2
livelossplot                  0.5.5
llvmlite                      0.39.1
locket                        1.0.0
logical-unification           0.4.5
LunarCalendar                 0.0.9
lxml                          4.9.2
lz4                           4.3.2
Mako                          1.2.4
Markdown                      3.4.3
MarkupSafe                    2.1.2
marshmallow                   3.19.0
matplotlib                    3.7.0
matplotlib-inline             0.1.6
miniful                       0.0.6
miniKanren                    1.0.3
minorminer                    0.2.9
mistune                       2.0.5
mljar-supervised              0.11.5
mlxtend                       0.21.0
mmh3                          2.5.1
modin                         0.18.1
mpi4py                        3.1.4
mplfinance                    0.12.9b7
mpmath                        1.2.1
msgpack                       1.0.4
mthree                        2.4.0
multidict                     6.0.4
multipledispatch              0.6.0
multiprocess                  0.70.14
multitasking                  0.0.11
murmurhash                    1.0.9
nbclassic                     0.5.5
nbclient                      0.7.3
nbconvert                     7.3.1
nbeats-keras                  1.8.0
nbeats-pytorch                1.8.0
nbformat                      5.8.0
nest-asyncio                  1.5.6
networkx                      2.8.8
neural-tangents               0.6.2
neuralprophet                 0.5.0
nfoursid                      1.0.1
nltk                          3.8.1
nose                          1.3.7
notebook                      6.5.4
notebook_shim                 0.2.2
ntlm-auth                     1.5.0
numba                         0.56.4
numerapi                      2.13.2
numexpr                       2.8.4
numpy                         1.23.5
numpydoc                      1.5.0
nvidia-cublas-cu11            11.10.3.66
nvidia-cuda-nvrtc-cu11        11.7.99
nvidia-cuda-runtime-cu11      11.7.99
nvidia-cudnn-cu11             8.5.0.96
nvidia-ml-py                  11.525.112
oauthlib                      3.2.2
openai                        0.26.5
opencensus                    0.11.2
opencensus-context            0.1.3
opencv-python                 4.7.0.72
```

```
openpyxl                    3.1.1
opt-einsum                  3.3.0
optuna                      3.1.0
orjson                      3.8.10
ortools                     9.4.1874
osqp                        0.6.2.post9
outdated                    0.2.2
packaging                   23.1
pandas                      1.5.3
pandas-datareader           0.10.0
pandas-flavor               0.5.0
pandas-market-calendars     4.1.4
pandas-ta                   0.3.14b0
pandocfilters               1.5.0
panel                       0.14.3
param                       1.13.0
parso                       0.8.3
partd                       1.4.0
pastel                      0.2.1
pathos                      0.3.0
pathy                       0.10.1
patsy                       0.5.3
pbr                         5.11.1
penaltymodel                1.0.2
PennyLane                   0.29.0
PennyLane-Lightning         0.29.0
PennyLane-qiskit            0.29.0
persim                      0.3.1
pexpect                     4.8.0
pickleshare                 0.7.5
Pillow                      9.5.0
pingouin                    0.5.3
pip                         22.3.1
pkgutil_resolve_name        1.3.10
platformdirs                3.2.0
plotly                      5.13.1
plotly-resampler            0.8.3.2
plucky                      0.4.3
pluggy                      1.0.0
ply                         3.11
pmdarima                    2.0.2
polars                      0.16.9
pox                         0.3.2
ppft                        1.7.6.6
pprofile                    2.1.0
ppscore                     1.2.0
preshed                     3.0.8
prometheus-client           0.16.0
prompt-toolkit              3.0.38
property-cached             1.6.4
prophet                     1.1.2
protobuf                    3.19.6
psutil                      5.9.5
psycopg2-binary             2.9.6
ptvsd                       4.3.2
ptyprocess                  0.7.0
PuLP                        2.7.0
pure-eval                   0.2.2
py-cpuinfo                  9.0.0
py-heat                     0.0.6
py-heat-magic               0.0.2
py-lets-be-rational         1.0.1
py-spy                      0.3.14
py-vollib                   1.0.1
py4j                        0.10.9.7
pyaml                       21.10.1
pyarrow                     11.0.0
pyasn1                      0.4.8
pyasn1-modules              0.2.8
pybind11                    2.10.4
pycares                     4.3.0
pycosat                     0.6.4
pycparser                   2.21
pyct                        0.5.0
pydantic                    1.10.7
pyDeprecate                 0.3.2
pydevd-pycharm              201.8538.36
```

```
pydmd                       0.4.0.post2301
pyerfa                      2.0.0.3
pyfolio                     0.9.2
pyfolio-reloaded            0.9.5
pyFUME                      0.2.25
Pygments                    2.15.1
pykalman                    0.9.5
pylev                       1.4.0
pyluach                     2.2.0
pymannkendall               1.4.3
pymc                        5.0.2
pymdptoolbox                4.0b3
PyMeeus                     0.5.12
PyMySQL                     1.0.3
pynndescent                 0.5.9
pyod                        1.0.9
Pyomo                       6.5.0
pyOpenSSL                   22.0.0
pyparsing                   3.0.9
pyportfolioopt              1.5.4
pyqubo                      1.3.1
pyrb                        1.0.1
pyro-api                    0.1.2
pyro-ppl                    1.8.4
pyrsistent                  0.19.3
pysimdjson                  5.0.2
PySocks                     1.7.1
pystan                      3.6.0
pytensor                    2.9.1
pytest                      7.3.1
python-dateutil             2.8.2
python-statemachine         1.0.3
pytorch-ignite              0.4.11
pytorch-lightning           1.7.4
pytz                        2023.3
pyvinecopulib               0.6.2
pyviz-comms                 2.2.1
PyWavelets                  1.4.1
PyYAML                      6.0
pyzmq                       25.0.2
qdldl                       0.1.7
qiskit                      0.41.1
qiskit-aer                  0.11.2
qiskit-ibmq-provider        0.20.1
qiskit-terra                0.23.2
qpd                         0.4.1
qtconsole                   5.4.2
QtPy                        2.3.1
quadprog                    0.1.11
quantecon                   0.6.0
QuantLib                    1.29
QuantStats                  0.0.59
rauth                       0.7.3
ray                         2.3.0
rectangle-packer            2.0.1
regex                       2023.3.23
requests                    2.28.1
requests-ntlm               1.1.0
requests-oauthlib           1.3.1
retrying                    1.3.4
retworkx                    0.12.1
rich                        12.4.4
ripser                      0.6.4
Riskfolio-Lib               4.0.3
riskparityportfolio         0.4
river                       0.14.0
rsa                         4.9
ruamel.yaml                 0.17.21
ruamel.yaml.clib            0.2.6
ruptures                    1.1.7
rustworkx                   0.12.1
SALib                       1.4.7
scikeras                    0.10.0
scikit-image                0.19.3
scikit-learn                1.2.1
scikit-learn-extra          0.2.0
scikit-multiflow            0.5.3
```

```
scikit-optimize              0.9.0
scikit-plot                  0.3.7
scikit-tda                   1.0.0
scipy                        1.10.1
scs                          3.2.3
sdeint                       0.3.0
seaborn                      0.12.2
semantic-version             2.10.0
Send2Trash                   1.8.0
setuptools                   64.0.2
setuptools-scm               7.1.0
shap                         0.41.0
simpful                      2.10.0
simplejson                   3.19.1
simpy                        4.0.1
six                          1.16.0
sklearn-json                 0.1.0
skope-rules                  1.0.1
sktime                       0.16.1
slicer                       0.0.7
smart-open                   6.3.0
sniffio                      1.3.0
snowballstemmer              2.2.0
sortedcontainers             2.4.0
soupsieve                    2.4.1
spacy                        3.5.0
spacy-legacy                 3.0.12
spacy-loggers                1.0.4
Sphinx                       6.1.3
sphinxcontrib-applehelp      1.0.4
sphinxcontrib-devhelp        1.0.2
sphinxcontrib-htmlhelp       2.0.1
sphinxcontrib-jsmath         1.0.1
sphinxcontrib-qthelp         1.0.3
sphinxcontrib-serializinghtml 1.1.5
SQLAlchemy                   1.4.47
sqlglot                      11.5.5
srsly                        2.4.6
ssm                          0.0.1
stable-baselines3            1.7.0
stack-data                   0.6.2
statsforecast                1.5.0
statsmodels                  0.13.5
stellargraph                 1.2.1
stevedore                    5.0.0
stochastic                   0.7.0
stockstats                   0.5.2
stumpy                       1.11.1
symengine                    0.10.0
sympy                        1.11.1
ta                           0.10.2
TA-Lib                       0.4.18
tables                       3.8.0
tabulate                     0.8.10
tadasets                     0.0.4
tbats                        1.1.3
tblib                        1.7.0
tenacity                     8.2.2
tensorboard                  2.11.2
tensorboard-data-server      0.6.1
tensorboard-plugin-wit       1.8.1
tensorboardX                 2.6
tensorflow                   2.11.0
tensorflow-addons            0.19.0
tensorflow-decision-forests  1.2.0
tensorflow-estimator         2.11.0
tensorflow-hub               0.13.0
tensorflow-io-gcs-filesystem 0.32.0
tensorflow-probability       0.19.0
tensorflow-text              2.11.0
tensorly                     0.8.0
tensortrade                  1.0.3
termcolor                    2.2.0
terminado                    0.17.1
tf2jax                       0.3.3
thinc                        8.1.9
threadpoolctl                3.1.0
```

```
        thriftpy2              0.4.16
        thundergbm             0.3.17
        tick                   0.7.0.1
        tifffile               2023.4.12
        tinycss2               1.2.1
        toml                   0.10.2
        tomli                  2.0.1
        toolz                  0.12.0
        torch                  1.13.1
        torch-cluster          1.6.0
        torch-geometric        2.2.0
        torch-scatter          2.1.0
        torch-sparse           0.6.16
        torch-spline-conv      1.2.1
        torchmetrics           0.9.3
        torchvision            0.14.1
        tornado                6.3
        tqdm                   4.64.1
        trace-updater          0.0.9.1
        trading-calendars      2.1.1
        traitlets              5.9.0
        treeinterpreter        0.2.3
        triad                  0.8.6
        tsfresh                0.20.0
        tslearn                0.5.3.2
        tweepy                 4.10.0
        typeguard              3.0.2
        typer                  0.3.2
        typing_extensions      4.5.0
        umap-learn             0.5.3
        urllib3                1.26.14
        virtualenv             20.21.0
        wasabi                 1.1.1
        wcwidth                0.2.6
        webargs                8.2.0
        webencodings           0.5.1
        websocket-client       1.5.1
        websockets             10.4
        Werkzeug               2.2.3
        wheel                  0.37.1
        widgetsnbextension     4.0.7
        wordcloud              1.8.2.2
        wrapt                  1.14.1
        wrds                   3.1.6
        wurlitzer              3.0.3
        xarray                 2023.1.0
        xarray-einstats        0.5.1
        xgboost                1.7.4
        xlrd                   2.0.1
        XlsxWriter             3.1.0
        yarl                   1.8.2
        yellowbrick            1.5
        yfinance               0.2.18
        zict                   3.0.0
        zipp                   3.15.0
        zope.event             4.6
        zope.interface         6.0
        zstandard              0.18.0
```

Pomegranate Environment

The Pomegranate environment supports the following libraries:

```
                                                          PY
        # Name                 Version
        absl-py                1.4.0
        adagio                 0.2.4
        aesara                 2.8.12
        aiodns                 3.0.0
        aiohttp                3.8.4
        aiohttp-cors           0.7.0
        aiosignal              1.3.1
        alabaster              0.7.13
```

```
alabaster                   0.7.13
ale-py                      0.7.4
alembic                     1.10.3
alpaca-trade-api            0.26
alphalens-reloaded          0.4.3
altair                      4.2.2
ansi2html                   1.8.0
antlr4-python3-runtime      4.11.1
anyio                       3.6.2
appdirs                     1.4.4
arch                        5.3.1
argon2-cffi                 21.3.0
argon2-cffi-bindings        21.2.0
arviz                       0.15.1
astropy                     5.2.1
asttokens                   2.2.1
astunparse                  1.6.3
async-timeout               4.0.2
asyncio-nats-client         0.11.5
attrs                       23.1.0
autograd                    1.5
autokeras                   1.1.0
autoray                     0.6.3
AutoROM                     0.4.2
AutoROM.accept-rom-license  0.6.1
ax-platform                 0.3.0
Babel                       2.12.1
backcall                    0.2.0
bayesian-optimization       1.4.2
beautifulsoup4              4.11.2
bleach                      6.0.0
blessed                     1.20.0
blis                        0.7.9
blosc2                      2.0.0
bokeh                       2.4.3
boltons                     23.0.0
botorch                     0.8.2
Bottleneck                  1.3.7
brotlipy                    0.7.0
cachetools                  5.3.0
captum                      0.6.0
catalogue                   2.0.8
catboost                    1.1.1
category-encoders           2.6.0
ccxt                        3.0.72
certifi                     2022.12.7
cffi                        1.15.1
chardet                     5.1.0
charset-normalizer          2.0.4
check-shapes                1.0.0
click                       7.1.2
clikit                      0.6.2
cloudpickle                 2.2.1
cmaes                       0.9.1
cmdstanpy                   1.1.0
colorama                    0.4.6
colorcet                    3.0.1
colorful                    0.5.5
colorlog                    6.7.0
colorlover                  0.3.0
colour                      0.1.5
comm                        0.1.3
commonmark                  0.9.1
conda                       23.3.1
conda-content-trust         0.1.3
conda-package-handling      2.0.2
conda_package_streaming     0.7.0
confection                  0.0.4
cons                        0.4.5
contourpy                   1.0.7
convertdate                 2.4.0
copulae                     0.7.7
copulas                     0.8.0
cramjam                     2.6.2
crashtest                   0.3.1
creme                       0.6.1
cryptography                38.0.4
cufflinks                   0.17.3
```

```
colrlinks                      0.17.3
cvxopt                         1.3.0
cvxpy                          1.3.0
cycler                         0.11.0
cymem                          2.0.7
Cython                         0.29.33
darts                          0.23.1
dash                           2.9.3
dash-core-components           2.0.0
dash-cytoscape                 0.3.0
dash-html-components           2.0.0
dash-table                     5.0.0
dask                           2023.4.0
deap                           1.3.3
debugpy                        1.5.1
decorator                      5.1.1
defusedxml                     0.7.1
Deprecated                     1.2.13
dgl                            1.0.1
dill                           0.3.6
dimod                          0.12.3
diskcache                      5.6.1
distlib                        0.3.6
distributed                    2021.4.1
dm-tree                        0.1.8
docutils                       0.19
DoubleML                       0.5.2
dtreeviz                       2.2.1
dtw-python                     1.3.0
dwave-cloud-client             0.10.3
dwave-drivers                  0.4.4
dwave-greedy                   0.3.0
dwave-hybrid                   0.6.9
dwave-inspector                0.3.0
dwave-inspectorapp             0.3.0
dwave-neal                     0.6.0
dwave-networkx                 0.8.12
dwave-ocean-sdk                6.1.1
dwave-preprocessing            0.5.3
dwave-samplers                 1.0.0
dwave-system                   1.18.0
dwave-tabu                     0.5.0
dwavebinarycsp                 0.2.0
dx                             0.1.22
ecos                           2.0.12
EMD-signal                     1.4.0
empyrical                      0.5.5
empyrical-reloaded             0.5.9
en-core-web-md                 3.5.0
en-core-web-sm                 3.5.0
entrypoints                    0.4
ephem                          4.1.4
et-xmlfile                     1.1.0
etuples                        0.3.8
exceptiongroup                 1.1.1
exchange-calendars             4.2.6
executing                      1.2.0
ezyrb                          1.3.0.post2304
fastai                         2.7.11
fastai2                        0.0.30
fastcore                       1.5.29
fastdownload                   0.0.7
fasteners                      0.18
fastjsonschema                 2.16.3
fastparquet                    2023.2.0
fastprogress                   1.0.3
fasttext                       0.9.2
featuretools                   0.23.1
filelock                       3.12.0
findiff                        0.9.2
finrl                          0.3.1
FixedEffectModel               0.0.5
Flask                          2.0.3
flatbuffers                    23.3.3
fonttools                      4.39.3
fracdiff                       0.9.0
frozendict                     2.3.7
frozenlist                     1.3.3
```

```
frozenlist                  1.3.3
fs                          2.4.16
fsspec                      2023.4.0
fst-pso                     1.8.1
fugue                       0.8.3
fugue-sql-antlr             0.1.6
future                      0.18.3
fuzzy-c-means               1.6.3
FuzzyTM                     2.0.5
gast                        0.4.0
gensim                      4.3.0
gevent                      22.10.2
gluonts                     0.12.3
google-api-core             2.11.0
google-auth                 2.17.3
google-auth-oauthlib        0.4.6
google-pasta                0.2.0
googleapis-common-protos    1.59.0
gpflow                      2.7.0
gplearn                     0.4.2
gpustat                     1.1
GPUtil                      1.4.0
gpytorch                    1.9.1
graphviz                    0.8.4
greenlet                    2.0.2
grpcio                      1.54.0
gym                         0.21.0
Gymnasium                   0.26.3
gymnasium-notices           0.0.1
h2o                         3.40.0.1
h5netcdf                    1.1.0
h5py                        3.8.0
hijri-converter             2.2.4
hmmlearn                    0.2.8
holidays                    0.23
holoviews                   1.15.4
homebase                    1.0.1
hopcroftkarp                1.2.5
html5lib                    1.1
httpstan                    4.9.1
hurst                       0.0.5
hvplot                      0.8.2
hyperopt                    0.2.7
idna                        3.4
iisignature                 0.24
ijson                       3.2.0.post0
imageio                     2.27.0
imagesize                   1.4.1
imbalanced-learn            0.10.1
importlib-metadata          4.13.0
importlib-resources         5.12.0
iniconfig                   2.0.0
injector                    0.20.1
interpret                   0.3.0
interpret-core              0.3.2
ipykernel                   6.22.0
ipython                     8.12.0
ipython-genutils            0.2.0
ipywidgets                  8.0.4
itsdangerous                2.1.2
jax                         0.4.4
jaxlib                      0.4.4
jedi                        0.18.2
Jinja2                      3.1.2
joblib                      1.2.0
jqdatasdk                   1.8.11
json5                       0.9.11
jsonpatch                   1.32
jsonpointer                 2.0
jsonschema                  4.17.3
jupyter                     1.0.0
jupyter-bokeh               3.0.5
jupyter-console             6.6.3
jupyter-dash                0.4.2
jupyter-resource-usage      0.7.2
jupyter-server              1.24.0
jupyter_client              8.2.0
jupyter-core                5.3.0
```

```
jupyter_core               5.3.0
jupyterlab                 3.4.4
jupyterlab-pygments        0.2.2
jupyterlab-widgets         3.0.7
jupyterlab_server          2.22.1
keract                     4.5.1
keras                      2.11.0
keras-nlp                  0.4.1
keras-rl                   0.4.2
keras-tuner                1.3.5
kiwisolver                 1.4.4
kmapper                    2.0.1
korean-lunar-calendar      0.3.1
kt-legacy                  1.0.5
langcodes                  3.3.0
lark                       1.1.5
lazy_loader                0.2
lazypredict                0.2.12
libclang                   16.0.0
lightgbm                   3.3.5
lime                       0.2.0.1
line-profiler              4.0.2
linear-operator            0.3.0
littleutils                0.2.2
livelossplot               0.5.5
llvmlite                   0.38.1
locket                     1.0.0
logical-unification        0.4.5
LunarCalendar              0.0.9
lxml                       4.9.2
lz4                        4.3.2
Mako                       1.2.4
Markdown                   3.4.3
MarkupSafe                 2.1.2
marshmallow                3.19.0
matplotlib                 3.7.0
matplotlib-inline          0.1.6
miniful                    0.0.6
miniKanren                 1.0.3
minorminer                 0.2.9
mistune                    2.0.5
mljar-supervised           0.11.5
mlxtend                    0.21.0
mmh3                       2.5.1
modin                      0.18.1
mpi4py                     3.1.4
mplfinance                 0.12.9b7
mpmath                     1.2.1
msgpack                    1.0.4
mthree                     2.4.0
multidict                  6.0.4
multipledispatch           0.6.0
multiprocess               0.70.14
multitasking               0.0.11
murmurhash                 1.0.9
mxnet                      1.9.1
nbclassic                  0.5.5
nbclient                   0.7.3
nbconvert                  7.3.1
nbeats-keras               1.8.0
nbeats-pytorch             1.8.0
nbformat                   5.8.0
nest-asyncio               1.5.6
networkx                   2.8.8
neural-tangents            0.6.2
neuralprophet              0.5.0
nfoursid                   1.0.1
nltk                       3.8.1
nose                       1.3.7
notebook                   6.5.4
notebook_shim              0.2.2
ntlm-auth                  1.5.0
numba                      0.55.1
numerapi                   2.13.2
numexpr                    2.8.4
numpy                      1.21.5
numpydoc                   1.5.0
```

```
nvidia-cublas-cu11            11.10.3.66
nvidia-cuda-nvrtc-cu11        11.7.99
nvidia-cuda-runtime-cu11      11.7.99
nvidia-cudnn-cu11             8.5.0.96
nvidia-ml-py                  11.525.112
oauthlib                      3.2.2
openai                        0.26.5
opencensus                    0.11.2
opencensus-context            0.1.3
opencv-python                 4.7.0.72
openpyxl                      3.1.1
opt-einsum                    3.3.0
optuna                        3.1.0
orjson                        3.8.10
ortools                       9.4.1874
osqp                          0.6.2.post9
outdated                      0.2.2
packaging                     23.1
pandas                        1.5.3
pandas-datareader             0.10.0
pandas-flavor                 0.5.0
pandas-market-calendars       4.1.4
pandas-ta                     0.3.14b0
pandocfilters                 1.5.0
panel                         0.14.3
param                         1.13.0
parso                         0.8.3
partd                         1.4.0
pastel                        0.2.1
pathos                        0.3.0
pathy                         0.10.1
patsy                         0.5.3
pbr                           5.11.1
penaltymodel                  1.0.2
PennyLane                     0.29.0
PennyLane-Lightning           0.29.0
PennyLane-qiskit              0.29.0
persim                        0.3.1
pexpect                       4.8.0
pickleshare                   0.7.5
Pillow                        9.5.0
pingouin                      0.5.3
pip                           22.0.4
pkgutil_resolve_name          1.3.10
platformdirs                  3.2.0
plotly                        5.13.1
plotly-resampler              0.8.3.2
plucky                        0.4.3
pluggy                        1.0.0
ply                           3.11
pmdarima                      2.0.2
polars                        0.16.9
pomegranate                   0.14.8
pox                           0.3.2
ppft                          1.7.6.6
pprofile                      2.1.0
ppscore                       1.2.0
preshed                       3.0.8
prometheus-client             0.16.0
prompt-toolkit                3.0.38
property-cached               1.6.4
prophet                       1.1.2
protobuf                      3.19.6
psutil                        5.9.5
psycopg2-binary               2.9.6
ptvsd                         4.3.2
ptyprocess                    0.7.0
PuLP                          2.7.0
pure-eval                     0.2.2
py-cpuinfo                    9.0.0
py-heat                       0.0.6
py-heat-magic                 0.0.2
py-lets-be-rational           1.0.1
py-spy                        0.3.14
py-vollib                     1.0.1
py4j                          0.10.9.7
pyaml                         21.10.1
```

```
pyarrow                  11.0.0
pyasn1                   0.4.8
pyasn1-modules           0.2.8
pybind11                 2.10.4
pycares                  4.3.0
pycosat                  0.6.4
pycparser                2.21
pyct                     0.5.0
pydantic                 1.10.7
pyDeprecate              0.3.2
pydevd-pycharm           201.8538.36
pydmd                    0.4.0.post2301
pyerfa                   2.0.0.3
pyfolio                  0.9.2
pyfolio-reloaded         0.9.5
pyFUME                   0.2.25
Pygments                 2.15.1
pykalman                 0.9.5
pylev                    1.4.0
pyluach                  2.2.0
pymannkendall            1.4.3
pymc                     5.0.2
pymdptoolbox             4.0b3
PyMeeus                  0.5.12
PyMySQL                  1.0.3
pynndescent              0.5.9
pyod                     1.0.9
Pyomo                    6.5.0
pyOpenSSL                22.0.0
pyparsing                3.0.9
pyportfolioopt           1.5.4
pyqubo                   1.3.1
pyrb                     1.0.1
pyro-api                 0.1.2
pyro-ppl                 1.8.4
pyrsistent               0.19.3
pysimdjson               5.0.2
PySocks                  1.7.1
pystan                   3.6.0
pytensor                 2.9.1
pytest                   7.3.1
python-dateutil          2.8.2
python-statemachine      1.0.3
pytorch-ignite           0.4.11
pytorch-lightning        1.7.4
pytz                     2023.3
pyvinecopulib            0.6.2
pyviz-comms              2.2.1
PyWavelets               1.4.1
PyYAML                   6.0
pyzmq                    25.0.2
qdldl                    0.1.7
qiskit                   0.41.1
qiskit-aer               0.11.2
qiskit-ibmq-provider     0.20.1
qiskit-terra             0.23.2
qpd                      0.4.1
qtconsole                5.4.2
QtPy                     2.3.1
quadprog                 0.1.11
quantecon                0.6.0
QuantLib                 1.29
QuantStats               0.0.59
rauth                    0.7.3
ray                      2.3.0
rectangle-packer         2.0.1
regex                    2023.3.23
requests                 2.28.1
requests-ntlm            1.1.0
requests-oauthlib        1.3.1
retrying                 1.3.4
retworkx                 0.12.1
rich                     12.4.4
ripser                   0.6.4
Riskfolio-Lib            4.0.3
riskparityportfolio      0.4
river                    0.14.0
```

```
rsa                              4.9
ruamel.yaml                      0.17.21
ruamel.yaml.clib                 0.2.6
ruptures                         1.1.7
rustworkx                        0.12.1
SALib                            1.4.7
scikeras                         0.10.0
scikit-image                     0.19.3
scikit-learn                     1.2.1
scikit-learn-extra               0.2.0
scikit-multiflow                 0.5.3
scikit-optimize                  0.9.0
scikit-plot                      0.3.7
scikit-tda                       1.0.0
scipy                            1.8.0
scs                              3.2.3
sdeint                           0.3.0
seaborn                          0.12.2
semantic-version                 2.10.0
Send2Trash                       1.8.0
setuptools                       56.0.0
setuptools-scm                   7.1.0
shap                             0.41.0
simpful                          2.10.0
simplejson                       3.19.1
simpy                            4.0.1
six                              1.16.0
sklearn-json                     0.1.0
skope-rules                      1.0.1
sktime                           0.16.1
slicer                           0.0.7
smart-open                       6.3.0
sniffio                          1.3.0
snowballstemmer                  2.2.0
sortedcontainers                 2.4.0
soupsieve                        2.4.1
spacy                            3.5.0
spacy-legacy                     3.0.12
spacy-loggers                    1.0.4
Sphinx                           6.1.3
sphinxcontrib-applehelp          1.0.4
sphinxcontrib-devhelp            1.0.2
sphinxcontrib-htmlhelp           2.0.1
sphinxcontrib-jsmath             1.0.1
sphinxcontrib-qthelp             1.0.3
sphinxcontrib-serializinghtml    1.1.5
SQLAlchemy                       1.4.47
sqlglot                          11.5.5
srsly                            2.4.6
ssm                              0.0.1
stable-baselines3                1.7.0
stack-data                       0.6.2
statsforecast                    1.5.0
statsmodels                      0.13.5
stellargraph                     1.2.1
stevedore                        5.0.0
stochastic                       0.7.0
stockstats                       0.5.2
stumpy                           1.11.1
symengine                        0.10.0
sympy                            1.11.1
ta                               0.10.2
TA-Lib                           0.4.18
tables                           3.8.0
tabulate                         0.8.10
tadasets                         0.0.4
tbats                            1.1.3
tblib                            1.7.0
tenacity                         8.2.2
tensorboard                      2.11.2
tensorboard-data-server          0.6.1
tensorboard-plugin-wit           1.8.1
tensorboardX                     2.6
tensorflow                       2.11.0
tensorflow-addons                0.19.0
tensorflow-decision-forests      1.2.0
tensorflow-estimator             2.11.0
```

```
tensorflow-hub                 0.13.0
tensorflow-io-gcs-filesystem   0.32.0
tensorflow-probability         0.19.0
tensorflow-text                2.11.0
tensorly                       0.8.0
tensortrade                    1.0.3
termcolor                      2.2.0
terminado                      0.17.1
tf2jax                         0.3.3
thinc                          8.1.9
threadpoolctl                  3.1.0
thriftpy2                      0.4.16
thundergbm                     0.3.17
tick                           0.7.0.1
tifffile                       2023.4.12
tigramite                      5.1.0.3
tinycss2                       1.2.1
toml                           0.10.2
tomli                          2.0.1
toolz                          0.12.0
torch                          1.13.1
torch-cluster                  1.6.0
torch-geometric                2.2.0
torch-scatter                  2.1.0
torch-sparse                   0.6.16
torch-spline-conv              1.2.1
torchmetrics                   0.9.3
torchvision                    0.14.1
tornado                        6.3
tqdm                           4.64.1
trace-updater                  0.0.9.1
trading-calendars              2.1.1
traitlets                      5.9.0
treeinterpreter                0.2.3
triad                          0.8.6
tsfresh                        0.20.0
tslearn                        0.5.3.2
tweepy                         4.10.0
typeguard                      3.0.2
typer                          0.3.2
typing_extensions              4.5.0
umap-learn                     0.5.3
urllib3                        1.26.14
virtualenv                     20.21.0
wasabi                         1.1.1
wcwidth                        0.2.6
webargs                        8.2.0
webencodings                   0.5.1
websocket-client               1.5.1
websockets                     10.4
Werkzeug                       2.2.3
wheel                          0.37.1
widgetsnbextension             4.0.7
wordcloud                      1.8.2.2
wrapt                          1.14.1
wrds                           3.1.6
wurlitzer                      3.0.3
xarray                         2023.1.0
xarray-einstats                0.5.1
xgboost                        1.7.4
xlrd                           2.0.1
XlsxWriter                     3.1.0
yarl                           1.8.2
yellowbrick                    1.5
yfinance                       0.2.18
zict                           3.0.0
zipp                           3.15.0
zope.event                     4.6
zope.interface                 6.0
zstandard                      0.18.0
```

Request New Libraries

To request a new library, contact us . We will add the library to the queue for review and deployment. Since the

libraries run on our servers, we need to ensure they are secure and won't cause harm. The process of adding new libraries takes 2-4 weeks to complete. View the list of libraries currently under review on the Issues list of the Lean GitHub repository .

Development Environment

Working With VS Code

Introduction

The VS Code Integrated Development Environment (IDE) lets you work on research notebooks and develop algorithms for backtesting and live trading. When you open a project , the IDE automatically displays. You can access your trading algorithms from anywhere in the world with just an internet connection and a browser.

Supported Languages

The Lean engine supports C# and Python. Python is less verbose, has more third-party libraries, and is more popular among the QuantConnect community than C#. C# is faster than Python and it's easier to contribute to Lean if you have features written in C# modules. Python is also the native language for the research notebooks, so it's easier to use in the Research Environment.
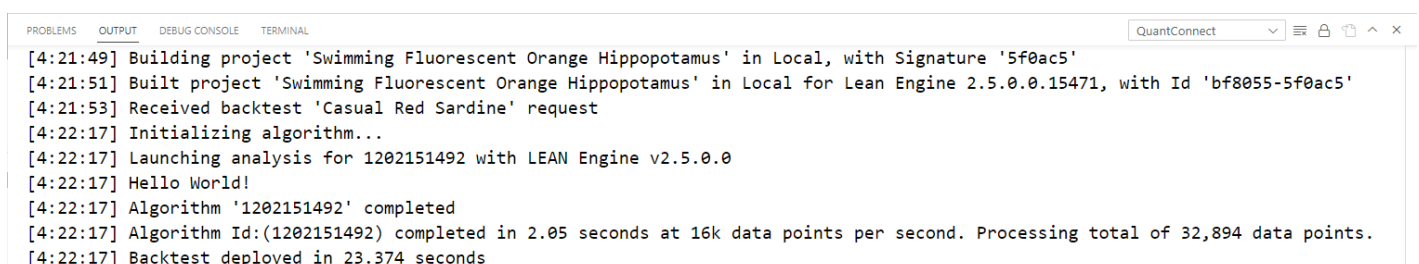
The programming language that you have set on your account determines how autocomplete and IntelliSense are verified and determines the types of files that are included in your new projects. If you have Python set as your programming language, new projects will have .py files. If you have C# set as your programming language, new projects will have .cs files.

Change Languages

To change the default programming language for your new projects, adjust the **User: Preferred Language** extension setting .
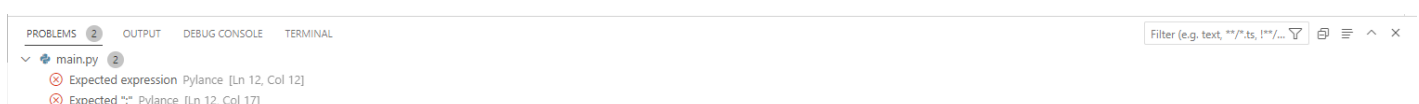
Terminal

The terminal panel at the bottom of the IDE shows API messages, errors, and the logs from your algorithms.

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                                                      QuantConnect
[4:21:49] Building project 'Swimming Fluorescent Orange Hippopotamus' in Local, with Signature '5f0ac5'
[4:21:51] Built project 'Swimming Fluorescent Orange Hippopotamus' in Local for Lean Engine 2.5.0.0.15471, with Id 'bf8055-5f0ac5'
[4:21:53] Received backtest 'Casual Red Sardine' request
[4:22:17] Initializing algorithm...
[4:22:17] Launching analysis for 1202151492 with LEAN Engine v2.5.0.0
[4:22:17] Hello World!
[4:22:17] Algorithm '1202151492' completed
[4:22:17] Algorithm Id:(1202151492) completed in 2.05 seconds at 16k data points per second. Processing total of 32,894 data points.
[4:22:17] Backtest deployed in 23.374 seconds
```

The **Problems** tab of the panel highlights the coding errors in your algorithms.

```
PROBLEMS  2   OUTPUT   DEBUG CONSOLE   TERMINAL                                               Filter (e.g. text, **/*.ts, !**/...
⌄  ⌄ ● main.py  2
      ⊗ Expected expression  Pylance  [Ln 12, Col 12]
      ⊗ Expected ":"  Pylance  [Ln 12, Col 17]
```

Navigate the File Outline

The **Outline** section in the Explorer panel is an easy way to navigate your files. The section shows the name of classes, members, and functions defined throughout the file. Click one of the names to jump your cursor to the respective definition in the file. To view the **Outline** , open a project and then, in the left navigation menu, click the 🗐 **Explorer** icon.
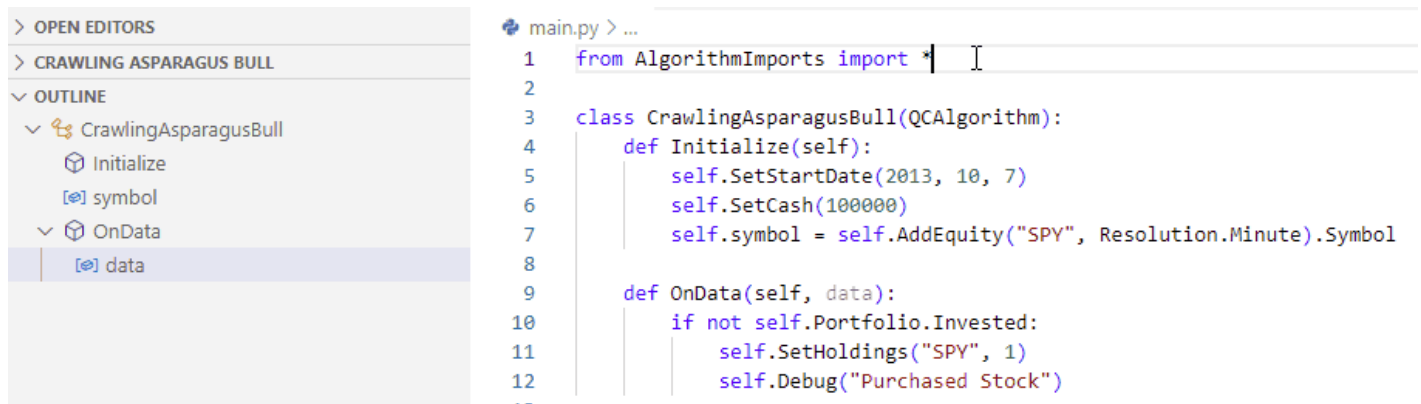
Split the Editor

The editor can split horizontally and vertically to display multiple files at once. Follow these steps to split the editor:

1. Open a project .
2. In the left navigation bar, click the 🗐 **Explorer** icon.
3. In the **QC (Workspace)** section, drag and drop the files you want to open.
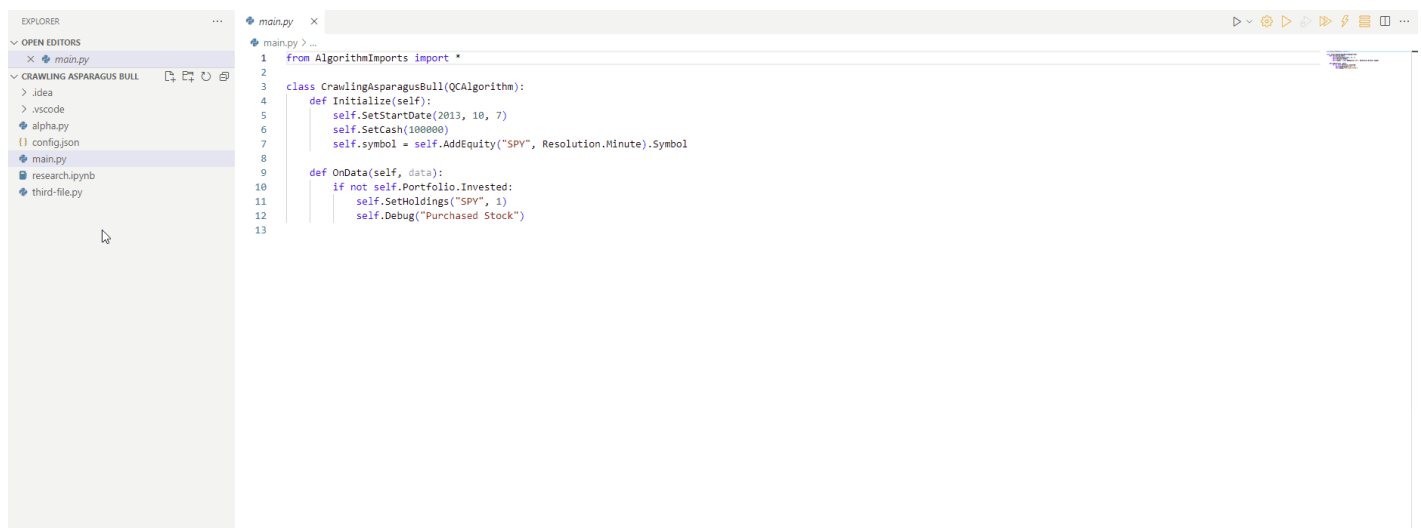
Show and Hide Code Blocks

The editor can hide and show code blocks to make navigating files easier. To hide and show code blocks, open a project and then click the **arrow** icon next to a line number.

```
1 ∨ class MyAlgorithm(QCAlgorithm):
2        def Initialize(self):
3            self.SetStartDate(2021, 1, 1)
4            self.SetCash(100000)
5            self.AddEquity("SPY")
6
```

Keyboard Shortcuts

Keyboard shortcuts are combinations of keys that you can issue to manipulate the IDE. They can speed up your workflow because they remove the need for you to reach for your mouse.

Follow these steps to view the keyboard shortcuts of your account:

1. Open a project .
2. Press **F1** .
3. Enter "Preferences: Open Keyboard Shortcuts".
4. Click **Preferences: Open Keyboard Shortcuts** .

To set a key binding for a command, click the **pencil** icon in the left column of the keyboard shortcuts table, enter the key combination, and then press **Enter** .

Projects

Projects

Getting Started

Introduction

Projects contain files to run backtests, launch research notebooks, perform parameter optimizations, and deploy live trading strategies. You need to create projects in order to create strategies and share your work with other members. Projects enable you to generate institutional-grade reports on the performance of your backtests. You can create your projects from scratch or you can utilize pre-built libraries and third-party packages to expedite your development process.

View All Projects

To view all your projects, open the organization workspace directory on your local machine.

Create Projects

Follow these steps to create a project on Local Platform:

1. Log in to Local Platform .
2. In the left navigation menu, click the ◉ **QuantConnect** icon.
3. If a project is already open, close it .
4. In the Open Project panel, click **Create Project** .

   QUANTCONNECT: OPEN PROJECT

   Open or create a project.

   Create Project

   Open Project

5. Enter the project name and then press **Enter** .

   Measured Yellow Barracuda

   Using base path 'd:\qc\workspaces\Quant Organization'. Please enter project name (Press 'Enter' to confirm or 'Escape' to cancel)
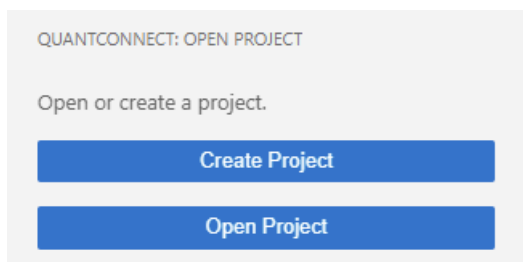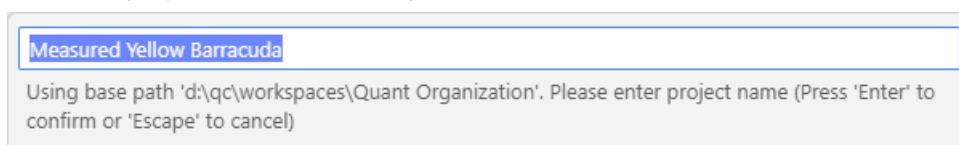
The new project directory is added to your organization workspace directory and the project opens.

Open Projects

Follow these steps to open a project on Local Platform:

1. Log in to Local Platform .

2. In the left navigation menu, click the ◎ **QuantConnect** icon.

3. If a project is already open, close it .

4. In the Project panel, click **Open Project** .
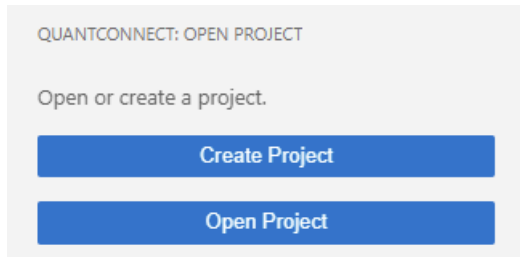
5. In the Open QuantConnect Project window, click a project in your organization workspace and then click **Select** .
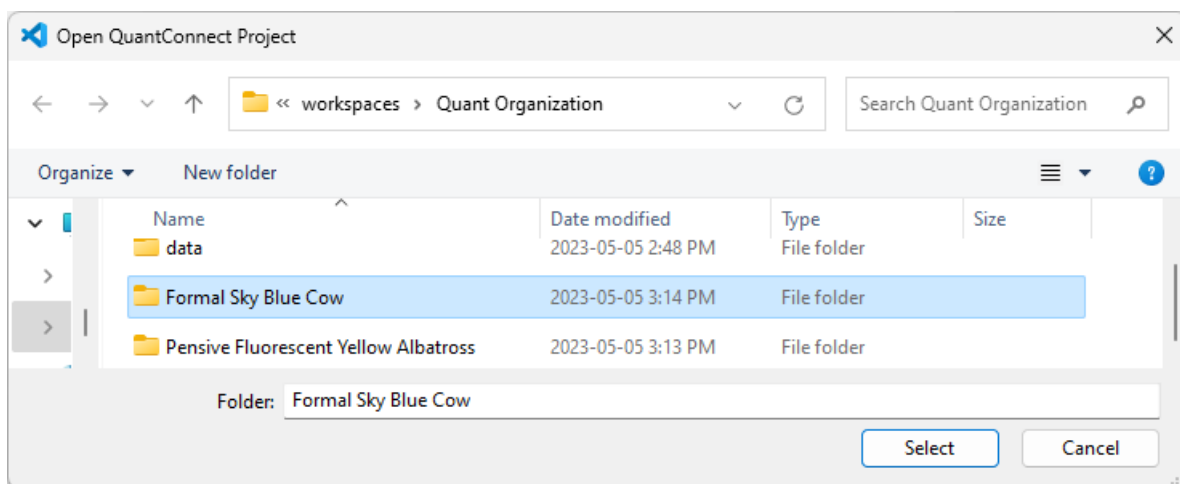
6. If an **I Trust the Authors** button appears, click it.

Close Projects

Follow these steps to close a project:

1. In the left navigation menu, click the ◎ **QuantConnect** icon.

2. In the Project panel, click **Close** .

Clone Projects

Clone a project to create a new copy of the project and save it within the same organization. When you clone a

project, the project files are duplicated but the backtest results are not retained. Cloning enables you to test small changes in your projects before merging the changes back into the original project.

Follow these steps to clone a project:

1. Open the project .
2. In the left navigation menu, click the ◎ **QuantConnect** icon.
3. In the Project panel, click **Clone** .

The cloned version of the project opens in a new VS Code window.

Rename Projects

Follow these steps to rename a project:

1. Open the project .
2. In the left navigation menu, click the ◎ **QuantConnect** icon.
3. In the Project panel, hover over the project name and then click the **pencil** icon that appears.
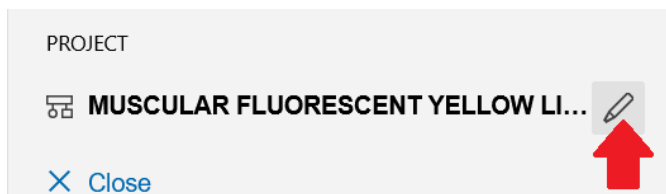
4. In the **Name** field, enter the new project name and then click **Save Changes** .

The project name must only contain - , _ , letters, numbers, and spaces. The project name can't start with a space or be any of the following reserved names: CON, PRN, AUX, NUL, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, or LPT9.

Create Project Directories

Set the name of a project to **directoryName / projectName** to create a project directory.

Set Descriptions

You can give a project a description to provide a high-level overview of the project and its functionality. Descriptions make it easier to return to old projects and understand what is going on at a high level without having to look at the code. The project description is also displayed at the top of backtest reports, which you can create after your backtest completes.

Follow these steps to set the project description:

1. Open the project .

2. In the Project panel, hover over the project name and then click the **pencil** icon that appears.
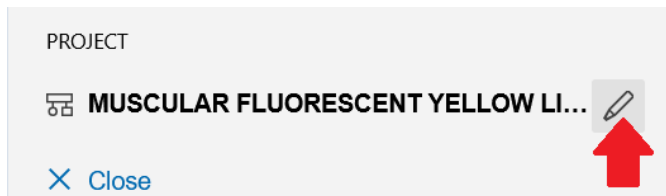
3. In the **Description** field, enter the new project description and then click **Save Changes** .

Edit Parameters

Algorithm parameters are hard-coded values for variables in your project that are set outside of the code files. Add parameters to your projects to remove hard-coded values from your code files and to perform parameter optimizations. You can add parameters, set default parameter values, and remove parameters from your projects.

Add Parameters

Follow these steps to add an algorithm parameter to a project:

1. Open the project .
2. In the left navigation menu, click the ◎ **QuantConnect** icon.
3. In the Project panel, click **Add New Parameter** .
4. Enter the parameter name.

   The parameter name must be unique in the project.

5. Enter the default value.
6. Click **Create Parameter** .

To get the parameter values into your algorithm, see Get Parameters .

Set Default Parameter Values

Follow these steps to set the default value of an algorithm parameter in a project:

1. Open the project .
2. In the left navigation menu, click the ◎ **QuantConnect** icon.
3. In the Project panel, hover over the algorithm parameter and then click the **pencil** icon that appears.
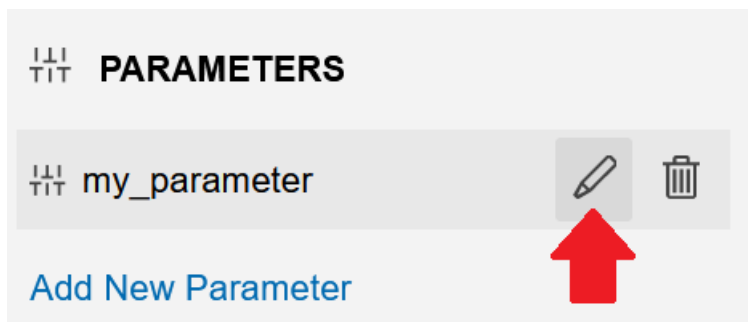
4. Enter a default value for the parameter and then click **Save** .

   The Project panel displays the default parameter value next to the parameter name.

Delete Parameters

Follow these steps to delete an algorithm parameter in a project:

1. Open the project .

2. In the left navigation menu, click the ⊚ **QuantConnect** icon.

3. In the Project panel, hover over the algorithm parameter and then click the **trash can** icon that appears.
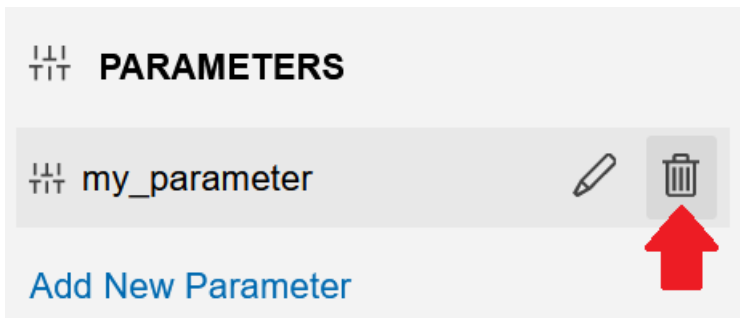
4. Remove the `GetParameter` calls that were associated with the parameter from your code files.

Delete Projects

Follow these steps to delete a project:

1. Open the project .

2. In the left navigation menu, click the ⊚ **QuantConnect** icon.

3. In the Project panel, click **Delete** .

Projects

Files

Introduction

The files in your projects enable you to implement trading algorithms, perform research, and store important information. Python projects start with a **main.py** and a **research.ipynb** file. C# projects start with a **Main.cs** and a **Research.ipynb** file. Use the **main.py** or **Main.cs** file to implement trading algorithms and use the **ipynb** file to access the Research Environment.

Supported File Types

Local Platform supports **.py** , **.cs** , and **.ipynb** files in your projects.

Code Files

The **.py** / **.cs** files are code files. These are the files where you implement your trading algorithm. When you backtest the project or deploy the project to live trading, the LEAN engine executes the algorithm you define in these code files.

Notebook Files

The **.ipynb** files are notebook files. These are the files you open when you want to access the Research Environment to perform quantitative research.

Configuration Files

Projects also contain configuration files, which are **.json** files, but they aren't displayed in the Explorer panel. These files contain information like the project description, parameters, and shared libraries. For more information about project configuration files, see Configuration .

Result Files

When you run a backtest, optimize some parameters, or deploy a strategy to live trading on your local machine, the results are saved as phyical files in the project directory. Local Platform doesn't push these result files to QuantConnect Cloud.

View Files

To view the files in a project, open the project and then, in the left navigation bar, click the ⬚ **Explorer** icon.

The **QC (Workspace)** section of the Explorer panel shows the files in the project.
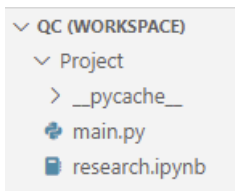
Add Files

Follow these steps to add a file to a project:

1. Open the project .

2. In the left navigation menu, click the ⬚ **Explorer** icon.

3. In the Explorer panel, expand the **QC (Workspace)** section.

4. Click the ⬚ **New File** icon.

5. Enter a file name and extension.

6. Press **Enter** .
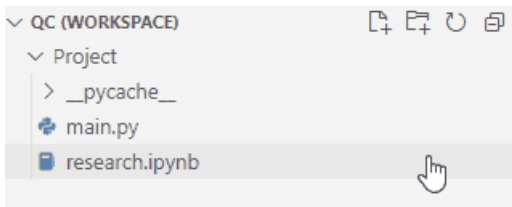
Add Directories

You can organize the code and notebook files in your project into directories to make navigating them easier. For example, if you have multiple Alpha models in your strategy, you can create an **alphas** directory in your project to hold a file for each Alpha model.

Follow these steps to add a directory to a project:

1. Open the project .

2. In the left navigation menu, click the ⬚ **Explorer** icon.

3. In the Explorer panel, expand the **QC (Workspace)** section.

4. Click the ⬚ **New Directory** icon.

5. Enter a directory name and then press **Enter** .
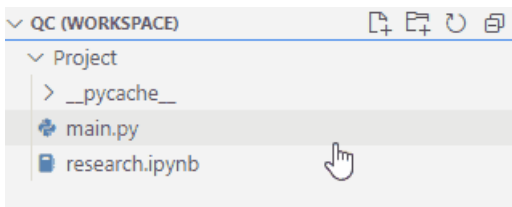
Open Files

Follow these steps to open a file in a project:

1. Open the project .

2. In the left navigation menu, click the ⬚ **Explorer** icon.

3. In the Explorer panel, click the file you want to open.

Close Files

To close a file, at the top of VS Code, click the **x** button on the file tab you want to close.

To close all of the files in a project, at the top of VS Code, right-click one of the file tabs and then click **Close All** .

Rename Files and Directories

Follow these steps to rename a file or directory in a project:

1. Open the project .

2. In the left navigation menu, click the ⧉ **Explorer** icon.

3. In the Explorer panel, right-click the file or directory you want to rename and then click **Rename** .

4. Enter the new name and then press **Enter** .
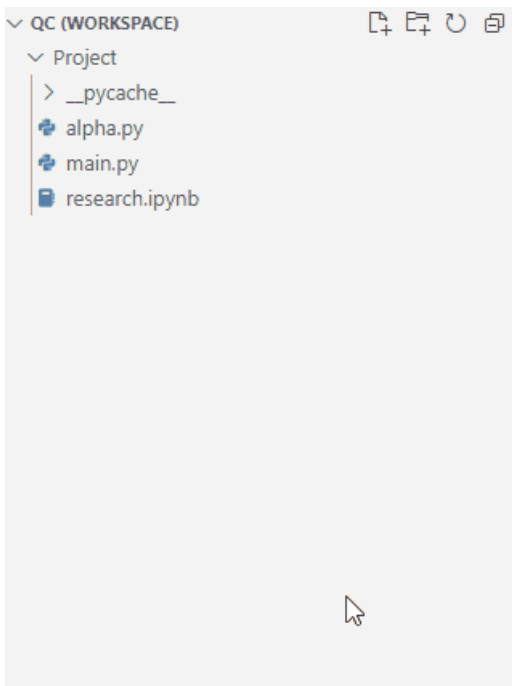
Delete Files and Directories

Follow these steps to delete a file or directory in a project:

1. Open the project .

2. In the left navigation menu, click the ⧉ **Explorer** icon.

3. In the Explorer panel, right-click the file or directory you want to delete and then click **Delete Permanently** .

4. Click **Delete** .

Projects

Shared Libraries

Introduction

Project libraries are QuantConnect projects you can merge into your project to avoid duplicating code files. If you have tools that you use across several projects, create a library.

Create Libraries

Follow these steps to create a library:

1. Open a project .
2. In the left navigation menu, click the ⊚ **QuantConnect** icon.
3. In the Project panel, click **Add Library** .
4. Click **Create New** .
5. In the **Input Library Name** field, enter a name for the library.
6. Click **Create Library** .

   The template library files are added to a new project in the **Library** directory in your organization workspace .

7. In the left navigation menu, click the ⧉ **Explorer** icon.
8. In Explorer panel, open the **Library.py** file and implement your library.

Add Libraries

Follow these steps to add a library to your project:

1. Open the project .
2. In the left navigation menu, click the ⊚ **QuantConnect** icon.
3. In the Project panel, click **Add Library** .
4. Click the **Choose a library...** field and then click a library from the drop-down menu.
5. Click **Add Library** (e.g. **Calculators**).

   The library files are added to your project. To view the files, in the right navigation menu, click the ⧉ **Explorer** icon.

6. Import the library into your project to use the library.

```py
from Calculators.TaxesCalculator import TaxesCalculator
class AddLibraryAlgorithm(QCAlgorithm):
    taxes_calculator = TaxesCalculator()
```

Rename Libraries

To rename a library, open the library project file and then rename the project .

Remove Libraries

Follow these steps to remove a library from your project:

1. Open the project that contains the library you want to remove.

2. In the left navigation menu, click the ◎ **QuantConnect** icon.

3. In the Project panel, hover over the library name and then click the **trash can** icon that appears.
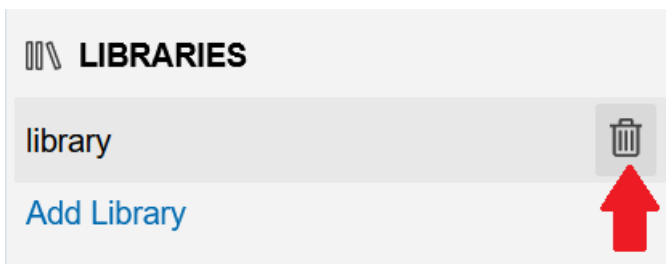
The library files are removed from your project.

Delete Libraries

To delete a library, delete the library project file .

Projects

Version Control

Introduction

Version control is the practice of tracking and managing changes to code files. By using version control, you can save an extra back up of your project files in the cloud, keep a history of all code changes, and easily revert changes to your projects.

Add Projects to Version Control

Follow these steps to add projects to your version control systems:

1. In your version control system, create a new repository for the project.
2. Open a terminal in your organization workspace and then clone the new repository to a temporary directory.

   ```
   $ git clone <repoUrl> temp
   ```

3. Move the **.git** directory from the temporary directory to the project directory.

   ```
   $ mv temp/.git <projectDirectory>/.git
   ```

4. Delete the temporary directory.

   ```
   $ rm -r temp
   ```

Data Management

Data Management > Getting Started

Data Management

Getting Started

Introduction

You need local data to run algorithms and perform research on Local Platform.

Download Formats

You can download individual data files or bulk download entire universes of assets. For more information about each of these data formats, see Downloading Data .

Physical Location

When you download data from the Dataset Market, it's stored in the **data** directory in your organization workspace . This is the same directory that LEAN reads data from when you run an algorithm or spin up a research notebook. To change the directory from which LEAN reads the data, open the configuration file and adjust the value under the `data-folder` key. The data folder should be fast and spacious. Follow these steps to check the approximate size of the datasets in the Dataset Market:

1. Open the Download in Bulk page.
2. Click a dataset.
3. Scroll down to the **Size and Format** section.

Data Formats

LEAN strives to use an open, human-readable format, so all data is stored in flat files (formatted as CSV or JSON). The data is compressed on disk using zip. For more information about general data formats, see the LEAN Data Formats README in the LEAN GitHub repository. The following README files explain the data formats of specific asset classes:

- Equity
- Equity Options

- [Crypto](#)

- [Forex](#)

- [Futures](#)

- [CFD](#)

Live Trading

You need live data feeds to inject data into your algorithm so that you can make real-time trading decisions and so that the values of the securities in your portfolio are updated. The data feeds you have available to you depend on where you deploy the algorithm. When you [request historical data](#) in local algorithms, the historical data comes from your data feed vendor. Familiarize yourself with the quotas and limits of your data feed to avoid errors.

Local Deployments

When you deploy local algorithms, you can use any of the following data feeds:

- [A brokerage data feed](#)

  These are streams of live security prices that come directly from your brokerage.

- IQFeed

  To view the asset classes that our IQFeed integration supports, see [Supported Assets](#) .

- Polygon

  To view the asset classes that our Polygon integration supports, see [Supported Assets](#) .

Cloud Deployments

When you deploy algorithms to QuantConnect Cloud, you can use any of [the data feeds we support in the cloud](#) , including SIP, CTA, CME, and select brokerages. Your live algorithms run on our co-located servers that have 10 GB transfer speeds and low latency.

Data Management

Downloading Data

To locally run the LEAN engine, you need local data. If you have a Download license, you can store datasets on your local machine. This download is for the licensed organization's internal LEAN use only and cannot be redistributed or converted in any format. If you study the data and produce some charts, you may share images of the charts online if the original data can't be to reconstructed from the image. The cost of the license depends on the dataset and it's calculated on a per-file or per-day basis. If you bulk download datasets, you can download historical data packages or daily updates. In most cases, you need both.

Download By Ticker

Low cost option for individual tickers

Download in Bulk

All tickers to avoid selection bias

See Also

Datasets

Backtesting

Backtesting

Getting Started

Introduction

Backtesting is the process of simulating a trading algorithm on historical data. By running a backtest, you can measure how the algorithm would have performed in the past. Although past performance doesn't guarantee future results, an algorithm that has a proven track record can provide investors with more confidence when deploying to live trading than an algorithm that hasn't performed favorably in the past. If you run local backtests, you can leverage your local data and hardware.

Run Backtests

Local Platform provides multiple deployment targets to enable you to run backtests on-premise and in the cloud. To run a backtest, open a project and then click the ▷ / ▷ **Backtest** icon. If the project successfully builds, "Received backtest backtestName request" displays. If the backtest successfully launches, the IDE displays the backtest results page in a new tab. If the backtest fails to launch due to coding errors, the new tab displays the error. As the backtest executes, you can close Local Platform and Docker Desktop without interfering with the backtest. Just don't quit Docker Desktop.

View All Backtests

Follow these steps to view all of the backtests of a project:

1. Open the project that contains the backtests you want to view.
2. In the top-right corner of the IDE, click the ▤ / ▤ **Backtest Results** icon.

   A table containing all of the backtest results for the project is displayed. If there is a **play** icon to the left of the name, it's a backtest result . If there is a **fast-forward** icon next to the name, it's an optimization result . The Platform column displays the deployment target of the backtest.

Results					Show	All ˅
Name		Platform	PSR ↓	Sharp…	Trades	Requested
▷ Calculated Brown Bear	✓ Completed	Cloud	11.568	0.292	3	2023-04-24 21:41:47
▷ Logical Brown Bee	✓ Completed	Local	0.078	0.2212	0	2023-04-24 20:37:15
▷ Creative Orange Seahorse	✓ Completed	Local	0.078	0.2212	0	2023-04-24 20:39:42
▷ Logical Orange Monkey	✓ Completed	Local	0.078	0.2212	0	2023-04-24 20:39:50
▷ Alert Fluorescent Orange Chinchilla	✓ Completed	Local	0.078	0.2212	0	2023-04-24 21:20:58

1 to 5 of 5 |< < Page 1 of 1 > >| ☐ Hide Runtime Errors

3. (Optional) In the top-right corner, select the **Show** field and then select one of the options from the drop-down menu to filter the table by backtest or optimization results.

4. (Optional) In the bottom-right corner, click the **Hide Error** check box to remove backtest and optimization results from the table that had a runtime error.

5. (Optional) Use the pagination tools at the bottom to change the page.

6. (Optional) Click a column name to sort the table by that column.

7. Click a row in the table to open the results page of that backtest or optimization.

Rename Backtests

We give an arbitrary name (for example, "Smooth Apricot Chicken") to your backtest result files, but you can follow these steps to rename them:

1. Open the backtest history of the project.

2. Hover over the backtest you want to rename and then click the **pencil** icon that appears.

▷ Swimming Light Brown Salmon	✎ ⧉ 🗑	21.489	0.581	1	2022-03-08 22:00:20

3. Enter the new backtest name and then click **OK** .

Results

The backtest results page and the **backtests** directory in your project show your algorithm's performance. Review the results to see how your algorithm has performed during the backtest and to investigate how you might improve your algorithm before live trading. For more information about backtest results, see Results .

Backtesting

Deployment

Introduction

Deploy a backtest to simulate the historical performance of your trading algorithm. Since the same Lean engine is used to run backtests and live trading algorithms, it's easy to transition from backtesting to live trading once you are satisfied with the historical performance of your algorithm. If you find any issues with Lean or our historical data, we'll resolve the issue.

Nodes

A **node** is a term to describe the compute hardware when your algorithm runs. We create "virtual nodes", which enable you to spin up multiple backtests with your on-premise hardware. When you run multiple backtests, each one runs in a separate container on the same host machine. To view all your virtual nodes, see the Resources panel.
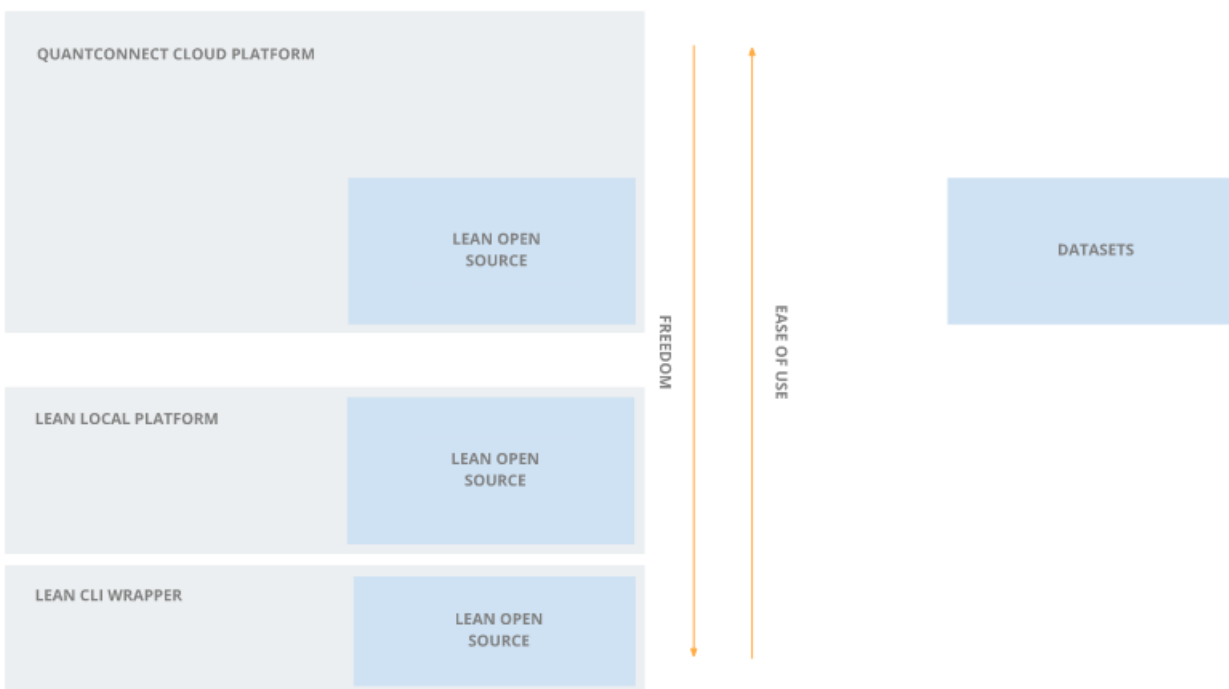
Concurrent Backtesting

Concurrent backtesting is the process of running multiple backtests at the same time. Concurrent backtesting speeds up your strategy development because you don't have to wait while a single backtest finishes executing.

You can run as many concurrent backtests as your CPU and RAM will handle.

Build Projects

If the compiler finds errors during the build process, the IDE highlights the lines of code that caused the errors in red. Your projects will automatically build after each keystroke. To manually build a project, open the project and then click the ⚙ / ⚙ **Build** icon.

Run Backtests

Local Platform provides multiple deployment targets to enable you to run backtests on-premise and in the cloud. To run a backtest, open a project and then click the ▷ / ▷ **Backtest** icon. If the project successfully builds, "Received backtest backtestName request" displays. If the backtest successfully launches, the IDE displays the backtest results page in a new tab. If the backtest fails to launch due to coding errors, the new tab displays the error. As the backtest executes, you can close Local Platform and Docker Desktop without interfering with the backtest. Just don't quit Docker Desktop.

Stop Backtests

To stop a running backtest, stop the backtesting node .

Backtesting

Results

Introduction

The backtest results page and the **backtests** directory in your project show your algorithm's performance. Review the results to see how your algorithm has performed during the backtest and to investigate how you might improve your algorithm before live trading.

View Backtest Results

The backtest results page automatically displays when you [deploy a backtest](#) . The backtest results page presents the equity curve, trades, logs, performance statistics, and much more information.
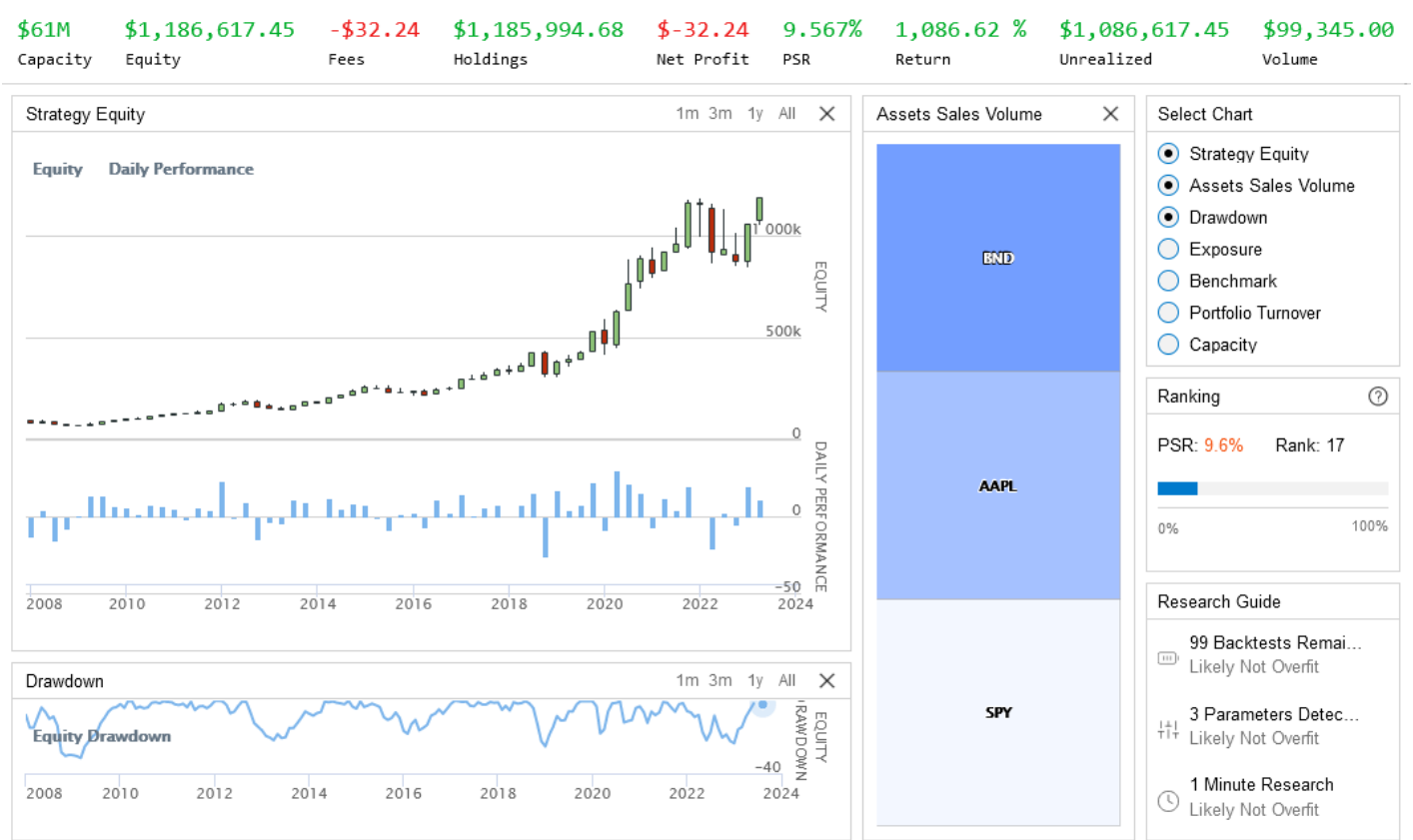
The content in the backtest results page updates as your backtest executes. You can close Local Platform without interrupting the backtest as long as you keep Docker running. If you close the page, to open it again, [view all of the project's backtests](#) . Only you can view the results of local backtests. If you run the backtest in QuantConnect Cloud, only you can view its results unless you explicitly make the backtest public. If you delete a backtest result or you are inactive for 12 months, we archive your backtest results.

The information on the backtest results page is also available in its raw form. To access it, see View Result Files .

Runtime Statistics

The banner at the top of the backtest results page displays the runtime statistics of your backtest.

$61M	$1,186,617.45	-$32.24	$1,185,994.68	$-32.24	9.567%	1,086.62 %	$1,086,617.45	$99,345.00
Capacity	Equity	Fees	Holdings	Net Profit	PSR	Return	Unrealized	Volume

The following table describes the default runtime statistics:

Statistic	Description
Capacity	The maximum amount of money an algorithm can trade before its performance degrades from market impact.
Equity	The total portfolio value if all of the holdings were sold at current market rates.
Fees	The total quantity of fees paid for all the transactions.
Holdings	The absolute sum of the items in the portfolio.
Net Profit	The dollar-value return across the entire trading period.
PSR	The probability that the estimated Sharpe ratio of an algorithm is greater than a benchmark (1).
Return	The rate of return across the entire trading period.
Unrealized	The amount of profit a portfolio would capture if it liquidated all open positions and paid the fees for transacting and crossing the spread.
Volume	The total value of assets traded for all of an algorithm's transactions.

To view the runtime statistics data in JSON format, open the **<organizationWorkspace> / <projectName> / backtests / <unixTimestamp> / <algorithmId>.json** file and search for the `RuntimeStatistics` key.

To add a custom runtime statistic, see Runtime Statistics .

Built-in Charts

The backtest results page displays a set of built-in charts to help you analyze the performance of your algorithm. The following table describes the charts displayed on the page:

Chart	Description
Strategy Equity	Time series of equity and daily performance
Capacity	Time series of strategy capacity snapshots
Drawdown	Time series of equity peak-to-trough value
Benchmark	Time series of the benchmark closing price (SPY, by default)
Exposure	Time series of long and short exposure ratios
Assets Sales Volume	Chart showing the proportion of total volume for each traded security
Portfolio Turnover	Time series of the portfolio turnover rate

To view the chart data in JSON format, open the **\<organizationWorkspace\> / \<projectName\> / backtests / \<unixTimestamp\> / \<algorithmId\>.json** file and search for the `Charts` key.

Custom Charts

The results page shows the custom charts that you create.

Supported Chart Types

We support the following types of charts:

If you use `SeriesType.Candle` and plot enough values, the plot displays candlesticks. However, the `Plot` method only accepts one numerical value per time step, so you can't plot candles that represent the open, high, low, and close values of each bar in your algorithm. The charting software automatically groups the data points you provide to create the candlesticks, so you can't control the period of time that each candlestick represents.

To create other types of charts, save the plot data in the ObjectStore and then load it into the Research Environment. In the Research Environment, you can create other types of charts with third-party charting packages .

Supported Markers

When you create scatter plots, you can set a marker symbol. We support the following marker symbols:

Chart Quotas

You can chart up to 4,000 data points. If you exceed the quota, the terminal displays the following message:

Exceeded maximum points per chart, data skipped

To adjust the data point quota, open your LEAN configuration file and adjust the value of the `maximum-data-points-per-chart-series` key.

You can create up to 10 custom chart series per algorithm. If you exceed the quota, your algorithm stops executing

and the Cloud Terminal displays the following message:

Exceeded maximum series count: Each backtest can have up to 10 series in total.

Demonstration

For more information about creating custom charts, see Charting .

Adjust Charts

You can manipulate the charts displayed on the backtest results page.

Toggle Charts

To display and hide a chart on the backtest results page, in the **Select Chart** section, click the name of a chart.

Toggle Chart Series

To display and hide a series on a chart on the backtest results page, click the name of a series at the top of a chart.

Adjust the Display Period

To zoom in and out of a time series chart on the backtest results page, perform either of the following actions:

- Click the **1m** , **3m** , **1y** , or **All** period in the top-right corner of the chart.
- Click a point on the chart and drag your mouse horizontally to highlight a specific period of time in the chart.

If you adjust the zoom on a chart, it affects all of the charts.

After you zoom in on a chart, slide the horizontal bar at the bottom of the chart to adjust the time frame that displays.
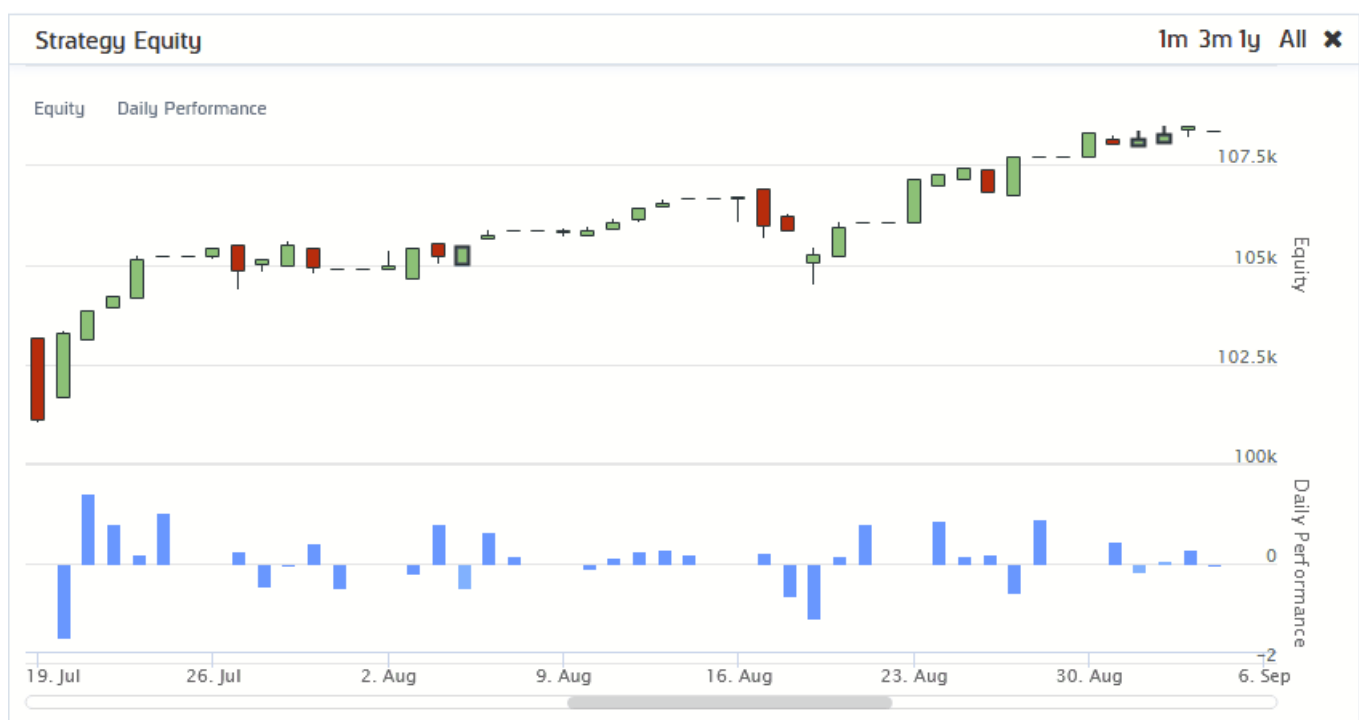
Resize Charts

To resize a chart on the backtest results page, hover over the bottom-right corner of the chart. When the resize cursor appears, hold the left mouse button and then drag to the desired size.

Move Charts

To move a chart on the backtest results page, click, hold, and drag the chart title.

Refresh Charts

Refreshing the charts on the backtest results page resets the zoom level on all the charts. If you refresh the charts while your algorithm is executing, only the data that was seen by the Lean engine after you refreshed the charts is displayed. To refresh the charts, in the **Select Chart** section, click the **reset** icon.

Key Statistics

The backtest results page displays many key statistics to help you analyze the performance of your algorithm.

Overall Statistics

The **Overview** tab on the backtest results page displays tables for Overall Statistics and Rolling Statistics. The Overall Statistics table displays the following statistics:

- Probabilistic Sharpe Ratio (PSR)
- Total Trades
- Average Loss
- Drawdown
- Net Profit
- Loss Rate
- Profit-Loss Ratio
- Beta
- Annual Variance
- Tracking Error
- Total Fees
- Lowest Capacity Asset
- Sharpe Ratio
- Average Win
- Compounding Annual Return
- Expectancy
- Win Rate
- Alpha
- Annual Standard Deviation
- Information Ratio
- Treynor Ratio
- Estimated Strategy Capacity

Some of the preceding statistics are sampled throughout the backtest to produce a time series of rolling statistics. The time series are displayed in the Rolling Statistics table.

To view the data from the Overall Statistics and Rolling Statistics tables in JSON format, open the **<organizationWorkspace>** / **<projectName>** / **backtests** / **<unixTimestamp>** / **<algorithmId>.json** file.

Ranking

If you run a cloud backtest, the backtest results page displays a Ranking section that shows the PSR and rank (percentile) of your algorithm relative to other algorithms in QuantConnect Cloud. For more information about this section, see Key Statistics .

Research Guide

For information about the Research Guide, see Research Guide .

Reports

The backtest report provides a summary of your algorithm's performance. Follow these steps to generate one:

1. Open the backtest results page for which you want to generate a report.
2. Click the **Report** tab.
3. If the project doesn't have a description, enter one and then click **Save** .
4. Click **Download Report** .

   The report may take a minute to generate.

5. If the IDE says that the report is being generated, repeat step 4.

If you create a report for a local backtest, the report is stored in the **<organizationWorkspace>** **/** **<projectName>** **/** **backtests** **/** **<unixTimestamp>** directory as **report.html** and **report.pdf** .

Orders

The backtest results page displays the orders of your algorithm and you can view them on your local machine.

View in the GUI

To see the orders that your algorithm created, open the backtest results page and then click the **Orders** tab. If there are more than 10 orders, use the pagination tools at the bottom of the Orders Summary table to see all of the orders. Click on an individual order in the Orders Summary table to reveal all of the order events , which include:

- Submissions
- Fills
- Partial fills
- Updates
- Cancellations
- Option contract exercises and expiration

The timestamps in the Order Summary table are based in Eastern Time (ET).

Access the Order Summary CSV

To view the orders data in CSV format, open the backtest results page, click the **Orders** tab, and then click **Download Orders** . The content of the CSV file is the content displayed in the Orders Summary table when the table rows are collapsed. The timestamps in the CSV file are based in Coordinated Universal Time (UTC). If you download the order summary CSV for a local backtest, the file is stored in **<organizationWorkspace>** **/**

<projectName> / backtests / <unixTimestamp> / orders.csv .

Access the Orders JSON

To view all of the content in the Orders Summary table, open the **<organizationWorkspace> / <projectName> / backtests / <unixTimestamp> / <algorithmId>.json** file and search for the `Orders` key.

Access the Order Events JSON

To view all of the order events for a local backtest, open the **<organizationWorkspace> / <projectName> / backtests / <unixTimestamp> / <algorithmId>-order-events.json** file.

Insights

The backtest results page displays the insights of your algorithm and you can view the raw insight data on your local machine.

View in the GUI

To see the insights your algorithm emit, open the backtest result page and then click the **Insights** tab. If there are more than 10 insights, use the pagination tools at the bottom of the Insights Summary table to see all of the insights. The timestamps in the Insights Summary table are based in Eastern Time (ET).

Open Raw JSON

To view the insights in JSON format, open the backtest result page, click the **Insights** tab, and then click **Download Insights** . The timestamps in the CSV file are based in Coordinated Universal Time (UTC).

If you run a local backtest, the JSON file is also available in the **<organizationWorkspace> / <projectName> / backtests / <unixTimestamp> / <algorithmId>-alpha-insights.json** file.

Logs

The backtest results page displays the logs of your backtest and you can view them on your local machine. The timestamps of the statements in the log file are based in your algorithm time zone .

View in the GUI

To see the log file that was created throughout a backtest, open the backtest result page and then click the **Logs** tab.

Download Log Files

To download the log file that was created throughout a backtest, follow these steps:

1. Open the backtest result page.
2. Click the **Logs** tab.
3. Click **Download Logs** .

If you ran a local backtest, the log file is automatically saved on your local machine when the backtest completes.

Access Local Log Files

To view the log file of a local backtest, open the **\<organizationWorkspace\> / \<projectName\> / backtests / \<unixTimestamp\> / \<algorithmId\>-log.txt** file.

Project Files

The backtest results page displays the project files used to run the backtest. To view the files, click the **Code** tab. By default, the **main.py** or **Main.cs** file displays. To view other files in the project, click the file name and then select a different file from the drop-down menu.
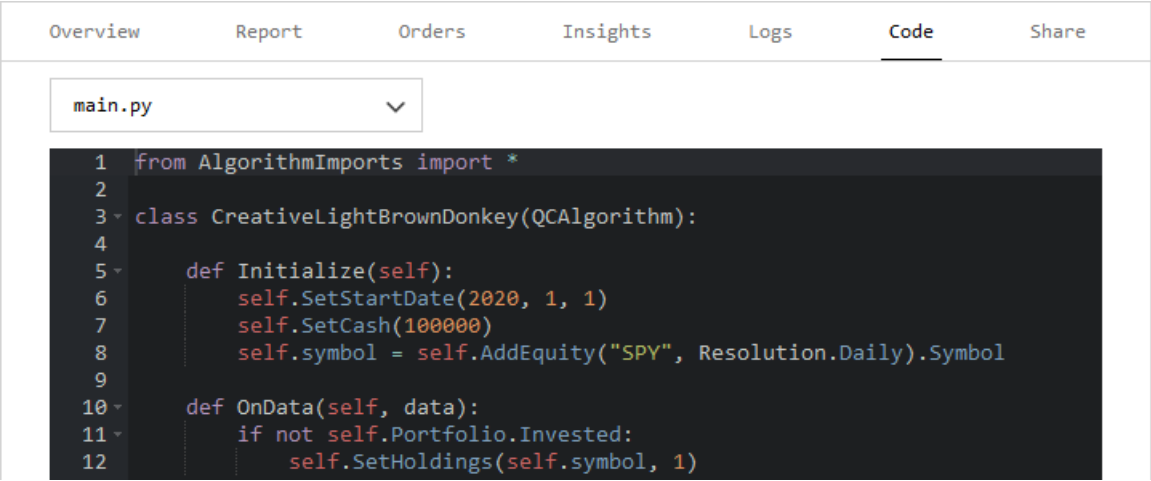
If you ran a local backtest, the project files are also available in the **\<organizationWorkspace\> / \<projectName\> / backtests / \<unixTimestamp\> / code** directory.

View Result Files

To view the results files of local backtests, run a local backtest and then open the **\<organizationWorkspace\> / \<projectName\> / backtests / \<unixTimestamp\>** directory. The following table describes the initial contents of the backtest result directories:

File/Directory	Description
code /	A directory containing a copy of the files that were in the project when you ran the backtest.
<algorithmId>-alpha-results.json Ex: **1967791529-alpha-results.json**	A file containing all of the backtest insights . This file only exists if you emit insights during the backtest.
<algorithmId>-log.txt Ex: **1967791529-log.txt**	A file containing all of the backtest logs .
<algorithmId>-order-events.json Ex: **1967791529-order-events.json**	A file containing all of the backtest order events .
<algorithmId>.json Ex: **1967791529.json**	A file containing the following data: - Runtime statistics - Charts - The data in the **Overview** tab - The data in the **Orders** tab - The algorithm configuration settings
config	A file containing some configuration settings, including the backtest Id, Docker container name, and backtest name.
data-monitor-report-<backtestDate><unixTimestamp>.json Ex: **data-monitor-report-20230614155459950.json**	A file containing statistics on the algorithm's data requests.
failed-data-requests-<backtestDate><unixTimestamp>.txt Ex: **failed-data-requests-20230614155451004.txt**	A file containing all the local data paths that LEAN failed to load during the backtest.
log.txt	A file containing the syslog.
succeeded-data-requests-<backtestDate><unixTimestamp>.txt Ex: **succeeded-data-requests-20230614155451004.txt**	A file containing all the local data paths that LEAN successfully loaded during the backtest.

The backtest result directories can contain the following additional files if you request them:

File	Description	Request Procedure
orders.csv	A file containing all of the data from the Orders table when the table rows are collapsed.	See Orders
report.html and **report.pdf**	A file containing the backtest report	See Reports

View All Backtests

Follow these steps to view all of the backtests of a project:

1. Open the project that contains the backtests you want to view.

2. In the top-right corner of the IDE, click the ▤ / ▤ **Backtest Results** icon.

   A table containing all of the backtest results for the project is displayed. If there is a **play** icon to the left of the name, it's a backtest result . If there is a **fast-forward** icon next to the name, it's an optimization result . The Platform column displays the deployment target of the backtest.

3. (Optional) In the top-right corner, select the **Show** field and then select one of the options from the drop-down menu to filter the table by backtest or optimization results.

4. (Optional) In the bottom-right corner, click the **Hide Error** check box to remove backtest and optimization results from the table that had a runtime error.

5. (Optional) Use the pagination tools at the bottom to change the page.

6. (Optional) Click a column name to sort the table by that column.

7. Click a row in the table to open the results page of that backtest or optimization.

Rename Backtests

We give an arbitrary name (for example, "Smooth Apricot Chicken") to your backtest result files, but you can follow these steps to rename them:

1. Hover over the backtest you want to rename and then click the **pencil** icon that appears.

2. Enter the new backtest name and then click **OK** .

Clone Backtests

Hover over the backtest you want to clone, and then click the **clone** icon that appears to clone the backtest.

| ▷ | Swimming Light Brown Salmon | ✏ | ⧉ | 🗑 | | 21.489 | 0.581 | 1 | 2022-03-08 22:00:20 |

A new project is created with the backtest code files.

Delete Backtests

Hover over the backtest you want to delete, and then click the **trash can** icon that appears to delete the backtest.

| ▷ | Swimming Light Brown Salmon | ✏ | ⧉ | 🗑 | | 21.489 | 0.581 | 1 | 2022-03-08 22:00:20 |

Backtesting

Debugging

Introduction

The debugger is a built-in tool to help you debug coding errors while backtesting. The debugger enables you to slow down the code execution, step through the program line-by-line, and inspect the variables to understand the internal state of the program.

Requirements

You need to install v2023.4.0 of Microsoft's Python VS Code extension to run the debugger.

Breakpoints

Breakpoints are lines in your algorithm where execution pauses. You need at least one breakpoint in your code files to start the debugger. Open a project to start adjusting its breakpoints.

Add Breakpoints

Click to the left of a line number to add a breakpoint on that line.

```
5    class MyAlgorithm(QCAlgorithm):
6        def Initialize(self):
7            self.SetStartDate(2022, 1, 1)
8            self.SetCash(100000)
9            self.AddEquity("SPY", Resolution.Minute)
```

Edit Breakpoint Conditions

Follow these steps to customize what happens when a breakpoint is hit:

1. Right-click the breakpoint and then click **Edit Breakpoint...** .
2. Click one of the options in the following table:

Option	Additional Steps	Description
Expression	Enter an expression and then press **Enter** .	The breakpoint only pauses the algorithm when the expression is true.
Hit Count	Enter an integer and then press **Enter** .	The breakpoint doesn't pause the algorithm until its hit the number of times you specify.

Enable and Disable Breakpoints

To enable a breakpoint, right-click it and then click **Enable Breakpoint** .

To disable a breakpoint, right-click it and then click **Disable Breakpoint** .

Follow these steps to enable and disable all breakpoints:

1. In the left navigation menu, click the **Run and Debug** icon.
2. In the Run and Debug panel, hover over the **Breakpoints** section and then click the **Toggle Active Breakpoints** icon.

Remove Breakpoints

To remove a breakpoint, right-click it and then click **Remove Breakpoint** .

Follow these steps to remove all breakpoints:

1. In the left navigation menu, click the **Run and Debug** icon.
2. In the Run and Debug panel, hover over the **Breakpoints** section and then click the **Remove All Breakpoints** icon.

Launch Debugger

Follow these steps to launch the debugger:

1. Open the project you want to debug.
2. In your project's code files, add at least one breakpoint.
3. Click the **Debug** icon.

If the Run and Debug panel is not open, it opens when the first breakpoint is hit.

Control Debugger

After you launch the debugger, you can use the following buttons to control it:

Button	Name	Default Keyboard Shortcut	Description
▷	Continue		Continue execution until the next breakpoint
↱	Step Over	Alt+F10	Step to the next line of code in the current or parent scope
↓	Step Into	Alt+F11	Step into the definition of the function call on the current line
↺	Restart	Shift+F11	Restart the debugger
⚡	Disconnect	Shift+F5	Exit the debugger

Inspect Variables

After you launch the debugger, you can inspect the state of your algorithm as it executes each line of code. You can inspect local variables or custom expressions. The values of variables in your algorithm are formatted in the IDE to improve readability. For example, if you inspect a variable that references a DataFrame, the debugger represents the variable value as the following:

```
                 close      high  ...     open       volume
symbol        time                    ...
SPY R735QTJ8XC9X 2021-05-11 415.246836 420.005970 ... 419.827130  74068713.0
                 2021-05-12 411.540872 412.594042 ... 410.457896 108355811.0
                 2021-05-13 402.797579 409.921376 ... 408.580075 121823945.0
                 2021-05-14 407.636197 409.692758 ... 404.397204 100076664.0
                 2021-05-15 413.895600 414.799736 ... 410.567187  72148360.0
                 2021-05-18 412.842431 413.696889 ... 412.713268  60431782.0
                 2021-05-19 409.285500 413.378951 ... 413.110691  54278598.0
                 2021-05-20 408.212459 409.285500 ... 404.307784  98038857.0
                 2021-05-21 412.603977 413.940310 ... 409.116595  72172531.0
                 2021-05-22 412.266168 415.505161 ... 414.143989  69119554.0

[10 rows x 5 columns]
```

Local Variables

The **Variables** section of the Run and Debug panel shows the local variables at the current breakpoint. If a variable in the panel is an object, click it to see its members. The panel updates as the algorithm runs.
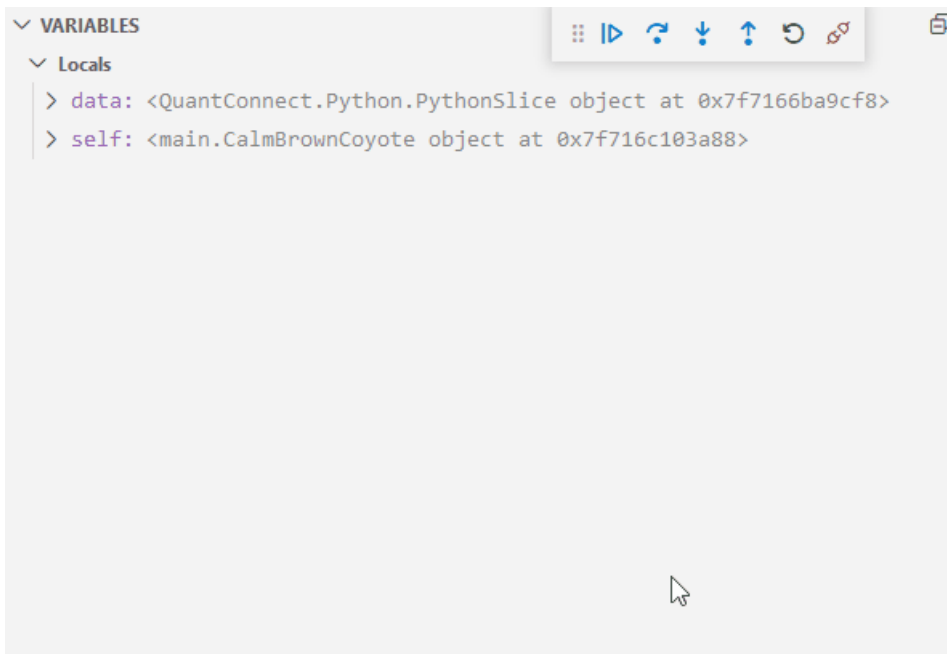
Follow these steps to update the value of a variable:

1. In the Run and Debug panel, right-click a variable and then click **Set Value** .

2. Enter the new value and then press **Enter** .

Custom Expressions

The **Watch** section of the Run and Debug panel shows any custom expressions you add. For example, you can add an expression to show the current date in the backtest.

Follow these steps to add a custom expression:

1. Hover over the **Watch** section and then click the **plus** icon that appears.

2. Enter an expression and then press **Enter** .

Backtesting

Troubleshooting

Introduction

This page explains some common troubleshooting topics that arise for backtests.

Speed Issues

If you notice that backtests run faster in QuantConnect Cloud than on-premise, it's likely because we host special hardware that's optimized for LEAN. For more information about our hardware, see Backtesting Nodes .

Local and Cloud Result Differences
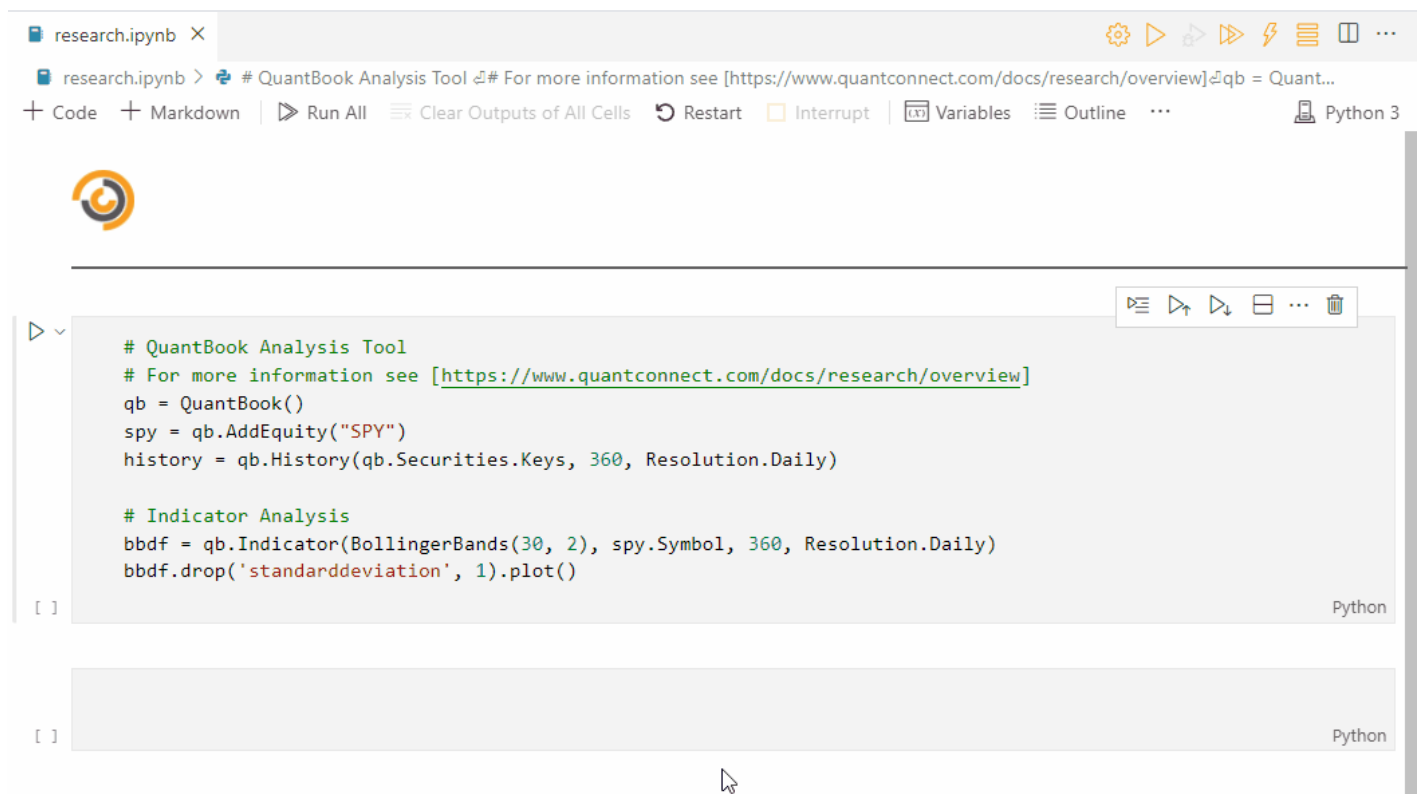
If your algorithm produces different results when you backtest it in QuantConnect Cloud versus on-premise, it's usually because of differences in data. For example, if you don't have the latest version of the US Equity Security Master , you will likely be missing some splits and dividends, which impact the historical prices of adjusted data. In this case, to avoid differences in the backtest results, update your local copy of the US Equity Security Master every day. For more information about downloading data from the Dataset Market so you have the same data on-premise as in QuantConnect Cloud, see Downloading Data .

Research

Research

Getting Started

Introduction

The Research Environment is a [Jupyter notebook](#) -based, interactive commandline environment where you can access your local data or our cloud data through the `QuantBook` class. The environment supports both Python and C#. If you use Python, you can import code from the code files in your project into the Research Environment to

aid development.

Before you run backtests, we recommend testing your hypothesis in the Research Environment. It's easier to perform data analysis and produce plots in the Research Environment than in a backtest.

Before backtesting or live trading with machine learning models, you may find it beneficial to train them in the Research Environment, save them in the ObjectStore, and then load them from the ObjectStore into the backtesting and live trading environment

In the Research Environment, you can also use the QuantConnect API to import your backtest results for further analysis.

Note: This chapter is an introduction to the Research Environment for Local Platform. For more comprehensive information on using research notebooks, see our dedicated Research Environment documentation.

Example

The following snippet demonstrates how to use the Research Environment to plot the price and Bollinger Bands of the S&P 500 index ETF, SPY:

```PY
# Create a QuantBook
qb = QuantBook()

# Create a security subscription
spy = qb.AddEquity("SPY")

# Request some historical data
history = qb.History(qb.Securities.Keys, 360, Resolution.Daily)

# Calculate the Bollinger Bands
bbdf = qb.Indicator(BollingerBands(30, 2), spy.Symbol, 360, Resolution.Daily)

# Plot the data
bbdf.drop('standarddeviation', 1).plot()
```

Out[2]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8af2445e10>

Open Notebooks

Local Platform provides multiple deployment targets to enable you to open notebooks on-premise and in

QuantConnect Cloud. Each new project you create contains a notebook file by default. Follow these steps to open it:

1. Open the project .
2. In the left navigation menu, click the 🗗 **Explorer** icon.
3. In the Explorer panel, expand the **QC (Workspace)** section.
4. Click the **research.ipynb** file.

Run Notebook Cells

Notebooks are a collection of cells where you can write code snippets or MarkDown. To execute a cell, press **Shift+Enter** .

```
3+7
```
[1]   Python

··· 10

```
print("Hello world!")
```
[2]   Python

··· Hello world!

The following describes some helpful keyboard shortcuts to speed up your research:

Keyboard Shortcut	Description
Shift+Enter	Run the selected cell.
a	Insert a cell above the selected cell.
b	Insert a cell below the selected cell.
x	Cut the selected cell.
v	Paste the copied or cut cell.
z	Undo cell actions.

Stop Nodes

Follow these steps to stop a research node:

1. If you opened the research notebook on Local Platform, close the notebook file .
2. In the Resources panel , click the **stop** button next to the research node you want to stop.
3. In the Visual Studio Code window, click **Yes** .

Add Notebooks

Follow these steps to add notebook files to a project:

1. Open the project .
2. In the right navigation menu, click the ⬚ **Explorer** icon.
3. In the Explorer panel, expand the **QC (Workspace)** section.
4. Click the ⬚ **New File** icon.
5. Enter **fileName .ipynb** .
6. Press **Enter** .
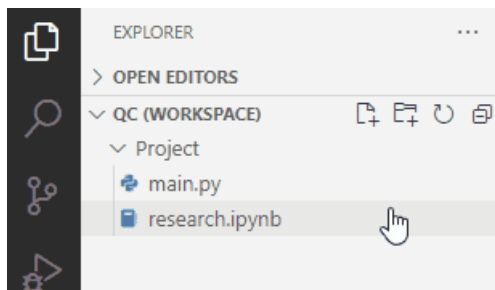
Rename Notebooks

Follow these steps to rename a notebook in a project:

1. Open the project .
2. In the left navigation menu, click the ⬚ **Explorer** icon.
3. In the Explorer panel, right-click the notebook you want to rename and then click **Rename** .
4. Enter the new name and then press **Enter** .

Delete Notebooks

Follow these steps to delete a notebook in a project:

1. Open the project .
2. In the left navigation menu, click the ⬚ **Explorer** icon.
3. In the Explorer panel, right-click the notebook you want to delete and then click **Delete Permanently** .
4. Click **Delete** .

Learn Jupyter

The following table lists some helpful resources to learn Jupyter:

Type	Name	Producer
Text	Jupyter Tutorial	tutorialspoint
Text	Jupyter Notebook Tutorial: The Definitive Guide	DataCamp
Text	An Introduction to DataFrame	Microsoft Developer Blogs

Research

Deployment

Introduction

This page is an introduction to the Research Environment for the Local Platform Lab. For more comprehensive information on using research notebooks, see the Research Environment documentation product.

Nodes

A **node** is a term to describe the compute hardware when your notebook runs. We create "virtual nodes", which enable you to spin up multiple research notebook with your on-premise hardware. When you run multiple notebooks, each one runs in a separate container on the same host machine. To view all your virtual nodes, see the Resources panel .
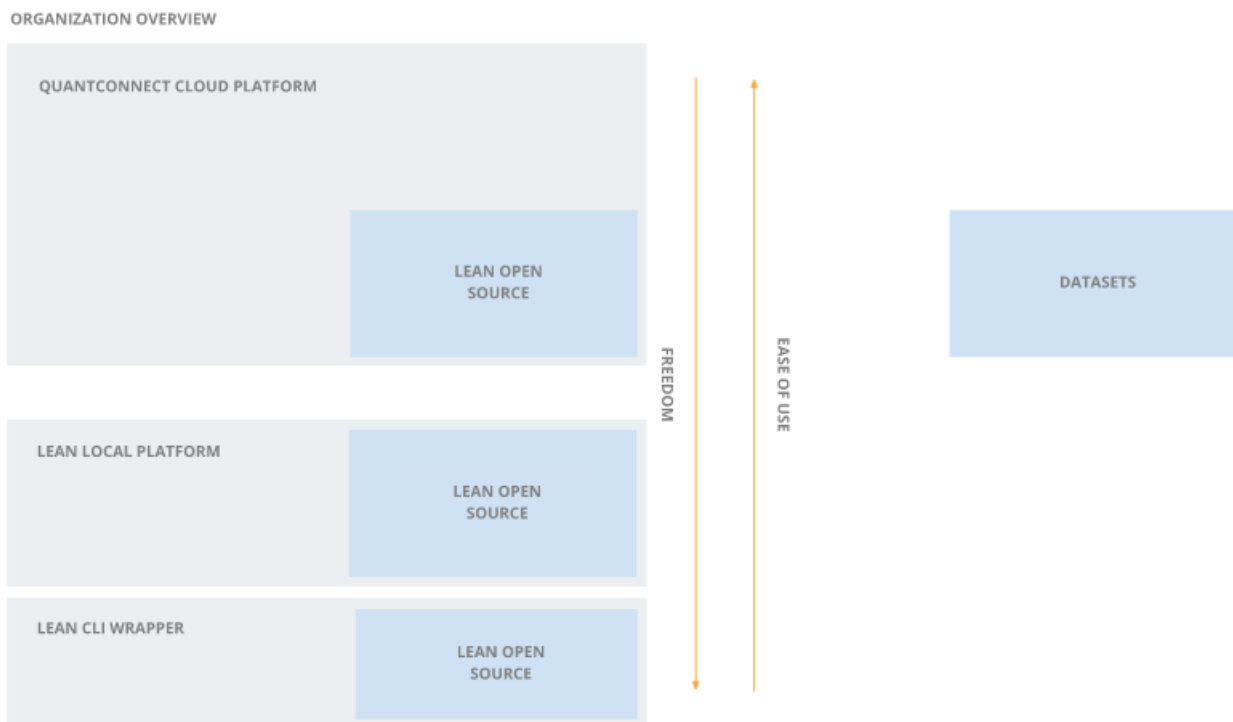
Concurrent Research

Concurrent research is the process of running multiple notebooks at the same time. Concurrent research speeds up your research process because you don't have to wait while a cell from a notebook finishes executing. You can run as many concurrent local notebooks as your CPU and RAM can handle.

Deployment Targets

Local Platform provides multiple deployment targets to enable you to open notebooks on-premise and in QuantConnect Cloud. When you open a notebook, it uses the hardware and data that's available on the deployment target machine.

Select Kernel

When you open a notebook , it automatically tries to connect to the correct Jupyter server and select the correct kernel. If it doesn't correctly select the kernel, the top-right corner of the notebook displays a **Select Kernel** button and the notebook won't let you run any of the cells. If this occurs, follow these steps to fix the issue:

1. In the top-right corner of the notebook, click **Select Kernel** .
2. In the Select Kernel window, click **QCRemoteJupyterServer** .
3. In the Select a Jupyter Server window, click **Local QuantConnect Research** or **QuantConnect Cloud** .
4. In the Select a Kernel window, click **Foundation-Py-Default** .

Save Notebooks

To save notebooks, press **CTRL+S** .

When you save a notebook, it saves the content of the cells. If you stop the Research Environment node or even just close the notebook , when you open the notebook again, you'll see the cell output. However, if you stop the Research Environment node and close the project , you'll need to run the cells again to generate the output.

Optimization

Optimization > Getting Started

Optimization

Getting Started

Introduction

Parameter optimization is the process of finding the optimal algorithm parameters to maximize or minimize an objective function. For instance, you can optimize your indicator parameters to maximize the Sharpe ratio that your algorithm achieves over a backtest. Optimization can help you adjust your strategy to achieve better backtesting performance, but be wary of overfitting. If you select parameter values that model the past too closely, your algorithm may not be robust enough to perform well using out-of-sample data.

Launch Optimization Jobs

Local Platform provides multiple deployment targets to enable you to run backtests on-premise and in QuantConnect Cloud.

You need the following to optimize parameters:

- At least one algorithm parameter in your project .
- The GetParameter method in your project.
- A successful backtest of the project.

Follow these steps to optimize parameters:

1. Open the project that contains the parameters you want to optimize.
2. In the top-right corner of the IDE, click the ▷▷ / ▷▷ **Optimize** icon.
3. On the Optimization page, in the **Parameter & Constraints** section, enter the name of the parameter to optimize.

   The parameter name must match a parameter name in the Project panel.

4. Enter the minimum and maximum parameter values.
5. Click the **gear** icon next to the parameter and then enter a step size.

6. If you want to add a second parameter to optimize, click **Add Parameter** .

   You can optimize a maximum of two parameters. To optimize more parameters, run local optimizations with the CLI .

7. If you want to add optimization constraints , follow these steps:
   1. Click **Add Constraint** .
   2. Click the **target** field and then select a target from the drop-down menu.
   3. Click the **operation** field and then an operation from the drop-down menu.
   4. Enter a constraint value.

8. If you are deploying to QuantConnect Cloud, in the **Estimated Number and Cost of Backtests** section, click an optimization node and then select a maximum number of nodes to use.

9. In the **Strategy & Target** section, click the **Choose Optimization Strategy** field and then select a strategy from the drop-down menu.

10. Click the **Select Target** field and then select a target from the drop-down menu.

    The target (also known as objective) is the performance metric the optimizer uses to compare the backtest performance of different parameter values.

11. Click **Maximize** or **Minimize** to maximize or minimize the optimization target, respectively.

12. Click **Launch Optimization** .

    The optimization results page displays. If you deploy a local optimization job, you can close Local Platform and Docker Desktop as the optimization job runs without interfering with the backtests. Just don't quit Docker Desktop. If you deploy the optimization job to QuantConnect Cloud, you can close Local Platform and Docker Desktop without interrupting with the backtests because the nodes are processing on our servers.

    To abort a running optimization job, in the Status panel, click **Abort** and then click **Yes** .

View Individual Backtest Results

The optimization results page displays a Backtests table that includes all of the backtests that ran during the optimization job. The table lists the parameter values of the backtests in the optimization job and their resulting values for the objectives.

Open the Backtest Results Page

To open the backtest result page of one of the backtests in the optimization job, click a backtest in the table.

Download the Table

To download the table, right-click one of the rows, and then click **Export > CSV Export** .

Filter the Table

Follow these steps to apply filters to the Backtests table:

1. On the right edge of the Backtests table, click **Filters** .
2. Click the name of the column to which you want the filter to be applied.
3. If the column you selected is numerical, click the **operation** field and then select one of the operations from the drop-down menu.
4. Fill the fields below the operation you selected.

Toggle Table Columns

Follow these steps to hide and show columns in the Backtests table:

1. On the right edge of the Backtests table, click **Columns** .

2. Select the columns you want to include in the Backtests table and deselect the columns you want to exclude.

Sort the Table Columns

In the Backtests table, click one of the column names to sort the table by that column.

View All Optimizations

Follow these steps to view all of the optimization results of a project:

1. Open the project that contains the optimization results you want to view.
2. At the top of the IDE, click the ▤ / ▤ **Results** icon.

   A table containing all of the backtest and optimization results for the project is displayed. If there is a **play** icon to the left of the name, it's a backtest result . If there is a **fast-forward** icon next to the name, it's an optimization results.

Results				Platform	PSR ↓	Sharp...	Trades	Requested
Name								
▷	Calculated Brown Bear	✓	Completed	Cloud	11.568	0.292	3	2023-04-24 21:41:47
▷	Logical Brown Bee	✓	Completed	Local	0.078	0.2212	0	2023-04-24 20:37:15
▷	Creative Orange Seahorse	✓	Completed	Local	0.078	0.2212	0	2023-04-24 20:39:42
▷	Logical Orange Monkey	✓	Completed	Local	0.078	0.2212	0	2023-04-24 20:39:50
▷	Alert Fluorescent Orange Chinchilla	✓	Completed	Local	0.078	0.2212	0	2023-04-24 21:20:58

1 to 5 of 5 |< < Page 1 of 1 > >| ☐ Hide Runtime Errors

3. (Optional) In the top-right corner, select the **Show** field and then select one of the options from the drop-down menu to filter the table by backtest or optimization results.
4. (Optional) In the bottom-right corner, click the **Hide Error** check box to remove backtest and optimization results from the table that had a runtime error.
5. (Optional) Use the pagination tools at the bottom to change the page.
6. (Optional) Click a column name to sort the table by that column.
7. Click a row in the table to open the results page of that backtest or optimization.

Rename Optimizations

We give an arbitrary name (for example, "Smooth Apricot Chicken") to your optimization result files, but you can follow these steps to rename them:

1. Hover over the optimization you want to rename and then click the **pencil** icon that appears.

▷▷	Emotional Fluorescent Pink Chicken	✏	🗑		63.049	1.324	13	2022-03-10 23:27:27

2. Enter the new name and then press **Enter** .

Delete Optimizations

Hover over the optimization you want to delete and then click the **trash can** icon that appears to delete the

optimization result.

Result Files

To view the results files of a local optimization job, open the **<organizationWorkspace> / <projectName> /
optimizations / <optimizationName>** directory. The following table describes the initial contents of the
optimization result directories:

File/Directory	Description
code /	A directory containing a copy of the files that were in the project when you ran the optimization.
<backtestId> / Ex: **1c5b8eff-89c0-432f-98c7-60b73265b188 /**	A directory containing the backtest result files for one of the backtest in the optimization job. There is separate directory for each backtest in the optimization job.
log.txt	A file containing the syslog.
config	A file containing some configuration settings, including the optimization Id, Docker container name, and optimization name.
optimizer-config.json	A file containing some configuration settings, including the parameters, strategy, and constrains.
optimization-result-<optimizationId>.json Ex: **optimization-result-2455523408.json**	A file containing additional results, including runtime statistics.

Live Trading

Live Trading

Getting Started

Introduction

A live algorithm is an algorithm that trades in real-time with real market data. Local Platform provides multiple deployment targets to enable you to run live algorithms on-premise and in QuantConnect Cloud. If you run live algorithms on-premise, you are prone to downtime from power outages, computer crashes, and natural disasters. If you don't want to be at risk to these, run your algorithms on QuantConnect Cloud

Deploy Live Algorithms

Follow these steps to deploy a live trading algorithm with the Interactive Brokers (IB) brokerage:

1. Open the project that you want to deploy.
2. Click the ⚡ / ⚡ **Deploy Live** icon.
3. On the Deploy Live page, click the **Brokerage** field and then click **Interactive Brokers** from the drop-down menu.
4. Enter your IB user name, ID, and password.

   If you use IB data feeds and trade with a paper trading account, you need to share the data feed with your paper trading account. For instructions on sharing data feeds, see Account Types .

   Your account details are not saved on QuantConnect.

5. In the **Weekly Restart UTC** field, enter the Coordinated Universal Time (UTC) time of when you want to receive notifications on Sundays to re-authenticate your account connection.

   For example, 4 PM UTC is equivalent to 11 AM Eastern Standard Time, 12 PM Eastern Daylight Time, 8 AM Pacific Standard Time, and 9 AM Pacific Daylight Time. To convert from UTC to a different time zone, see the UTC Time Zone Converter on the UTC Time website.

   If your IB account has 2FA enabled, you receive a notification on your IB Key device every Sunday to re-

authenticate the connection between IB and your live algorithm. If you don't re-authenticate before the timeout period, your algorithm quits executing.

6. Click the **Node** field and then click the live trading node that you want to use from the drop-down menu.

7. (Optional) If you are deploying to QuantConnect Cloud, set up notifications .

8. Configure the **Automatically restart algorithm** setting.

   By enabling automatic restarts, the algorithm will use best efforts to restart the algorithm if it fails due to a runtime error. This can help improve the algorithm's resilience to temporary outages such as a brokerage API disconnection.

9. Click **Deploy** .

10. If your IB account has 2FA enabled, tap the notification on your IB Key device and then enter your pin.

The deployment process can take up to 5 minutes. When the algorithm deploys, the live results page displays. If you know your brokerage positions before you deployed, you can verify they have been loaded properly by checking your equity value in the runtime statistics, your cashbook holdings, and your position holdings.

To deploy a live algorithm with a different brokerage, see the **Deploy Live Algorithms** section of the brokerage integration documentation .

Stop Live Algorithms

The live trading results page has a **Stop** button to immediately stop your algorithm from executing. When you stop a live algorithm, your portfolio holdings are retained. Stop your algorithm if you want to perform any of the following actions:

- Update your project's code files
- Update the settings you entered into the deployment command
- Place manual orders through your brokerage account

Furthermore, if you receive new securities in your portfolio because of a reverse merger, you also need to stop and redeploy the algorithm.

LEAN actively terminates live algorithms when it detects interference outside of the algorithm's control to avoid conflicting race conditions between the owner of the account and the algorithm, so avoid manipulating your brokerage account and placing manual orders on your brokerage account while your algorithm is running. If you need to adjust your brokerage account holdings, stop the algorithm, manually place your trades, and then redeploy the algorithm.

Follow these steps to stop your algorithm:

1. Open your algorithm's live results page.

2. Click **Stop** .

3. Click **Stop** again.

Liquidate Live Algorithms

The live results page has a **Liquidate** button that acts as a "kill switch" to sell all of your portfolio holdings. If your algorithm has a bug in it that caused it to purchase a lot of securities that you didn't want, this button let's you easily liquidate your portfolio instead of placing many manual trades. When you click the **Liquidate** button, if the market is open for an asset you hold, the algorithm liquidates it with market orders. If the market is not open, the algorithm places market on open orders. After the algorithm submits the liquidation orders, it stops executing.

Follow these steps to liquidate your positions:

1. Open your algorithm's live results page.
2. Click **Liquidate** .
3. Click **Liquidate** again.

Data Feeds

Local Platform currently supports several brokerage data feeds . To use other data feeds, contact us .

Result Files

When you deploy a live algorithm, the live results page automatically displays. To view the results in their raw form, open the **<organizationWorkspace>** **/ <projectName> / live / <timeStamp>** directory. The following table describes the initial contents of the live result directories:

File/Directory	Description
code /	A directory containing a copy of the files that were in the project when you deployed the algorithm.
L-<deploymentId>.json Ex: **L-3712451018.json**	A file containing the following data: • Holdings and cash • Account currency • Charts • Orders • Runtime statistics • Server statistics
L-<deploymentId>-<date>_minute.json Ex: **L-3712451018-2023-06-22_minute.json**	A file similiar to the **L-<deploymentId>.json** file, but the values of the chart data are only sampled every 10 minutes.
L-<deploymentId>-<date>_10minute.json Ex: **L-3712451018-2023-06-22_10minute.json**	A file similiar to the **L-<deploymentId>.json** file, but the values of the chart data are only sampled every 10 minutes.
L-<deploymentId>-<date>_second_Strategy%20Equity.json Ex: **L-3712451018-2023-06-22-19_second_Strategy%20Equity.json**	A file containing the algorithm holdings, chart, and orders. The values of the chart data are sampled every few seconds.
L-<deploymentId>-log.txt Ex: **L-3712451018**	A file containing all of the live trading logs.
log.txt	A file containing the syslog.
config	A file containing some configuration settings, including the deployment Id, brokerage name, and Docker container name.

Data Storage

Introduction

The ObjectStore is an project-specific key-value storage location to save and retrieve data. Similar to a dictionary or hash table, a key-value store is a storage system that saves and retrieves objects by using keys. A key is a unique string that is associated with a single record in the key-value store and a value is an object being stored. Some common use cases of the ObjectStore include the following:

- Transporting data between the backtesting environment and the research environment.
- Training machine learning models in the research environment before deploying them to live trading.

Supported Types

The ObjectStore has helper methods to store strings and bytes.

```PY
self.ObjectStore.Save(string_key, string_value)
self.ObjectStore.SaveBytes(bytes_key, bytes_value)
```

To store an object that is in a different format, you need to encode it to one of the supported data types. For instance, if you train a machine learning model and it is in binary format, encode it into base 64 before saving it.

The ObjectStore also has helper methods to retrieve the stored objects.

```PY
string_value = self.ObjectStore.Read(string_key)
bytes_value = self.ObjectStore.ReadBytes(bytes_key)
```

For complete examples of using the ObjectStore, see Object Store .

Storage Location

The Object Store is project-specific. When you save data in the Object Store , it creates a new file in the **<organizationWorkspace>** **/ <projectName> / storage** directory and names the file with the key you provide. To access the storage data of project A from project B, duplicate the files in the **storage** directory of project A to the **storage** directory of project B.

Research to Live Considerations

When you deploy a live algorithm, you can access the data immediately after modifying the ObjectStore. Ensure your algorithm is able to handle a changing dataset.

Delete Storage

To free up storage space, delete the key-value pairs in the ObjectStore by calling the `Delete` method with a key.

```py
    self.ObjectStore.Delete(key)
```

Alternatively, delete the files in **<organizationWorkspace> / <projectName> / storage** directory.