

Overview

Tradovate's Custom Indicators is implemented as a directly accessible function which can be launched directly from any indicator configuration modal or by adding the "Code Explorer" module which can be found in the Workspace Manager (under Third-party Modules).

This document will provide a high-level overview of how to activate and use of the Custom Indicators and will touch on the following area:

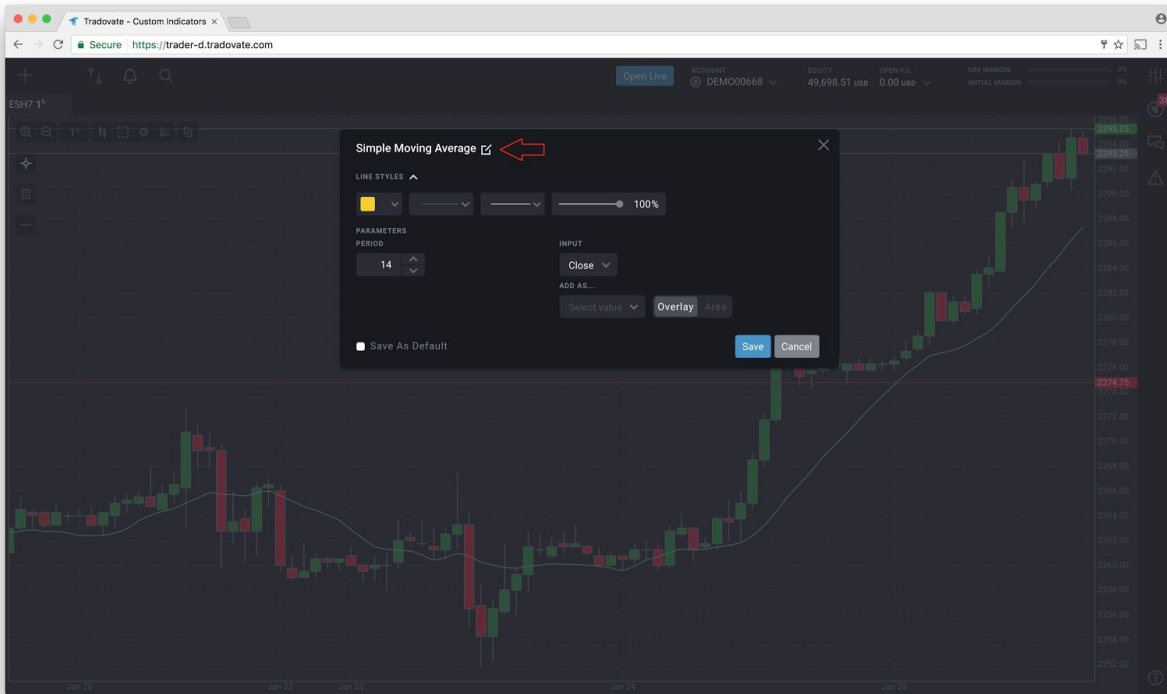
1. Accessing the Code Editor
2. Custom Indicators Editor Overview
3. View Tradovate's Indicator Code
4. Working with Indicators
 - a. New Indicators
 - b. Editing Indicators
 - c. Deleting Indicators
5. Managing Menu Structure
6. Importing and Exporting Indicators
7. Sharing Indicators / Community

Tradovate's indicators are written in JavaScript and all current indicators have been open-sourced to allow users to see how they have been constructed and reuse any code that is currently available in developing their own indicators.

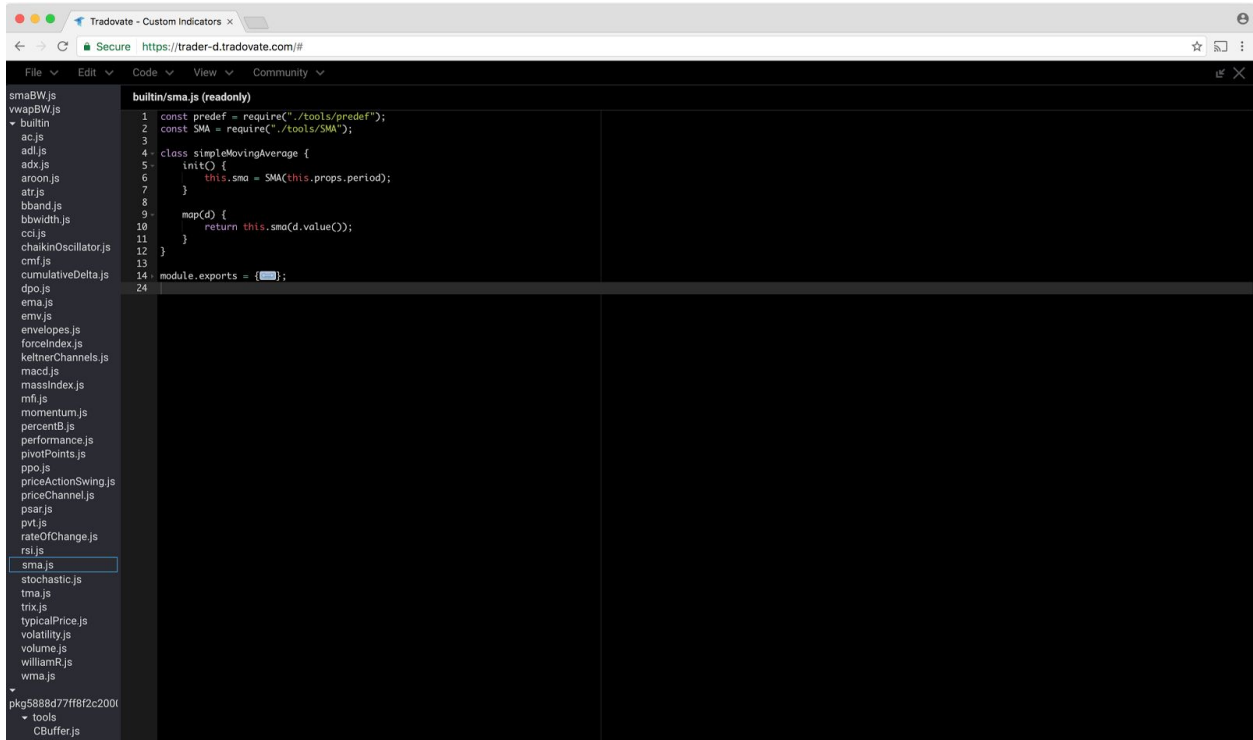
Accessing the Code Editor

The indicator code editor can be accessed in 2 different ways:

- From inside any Indicator Configuration Modal, there is an “Edit” icon next to the indicator name



Clicking on this icon will launch the editor, and show the code for the selected indicator

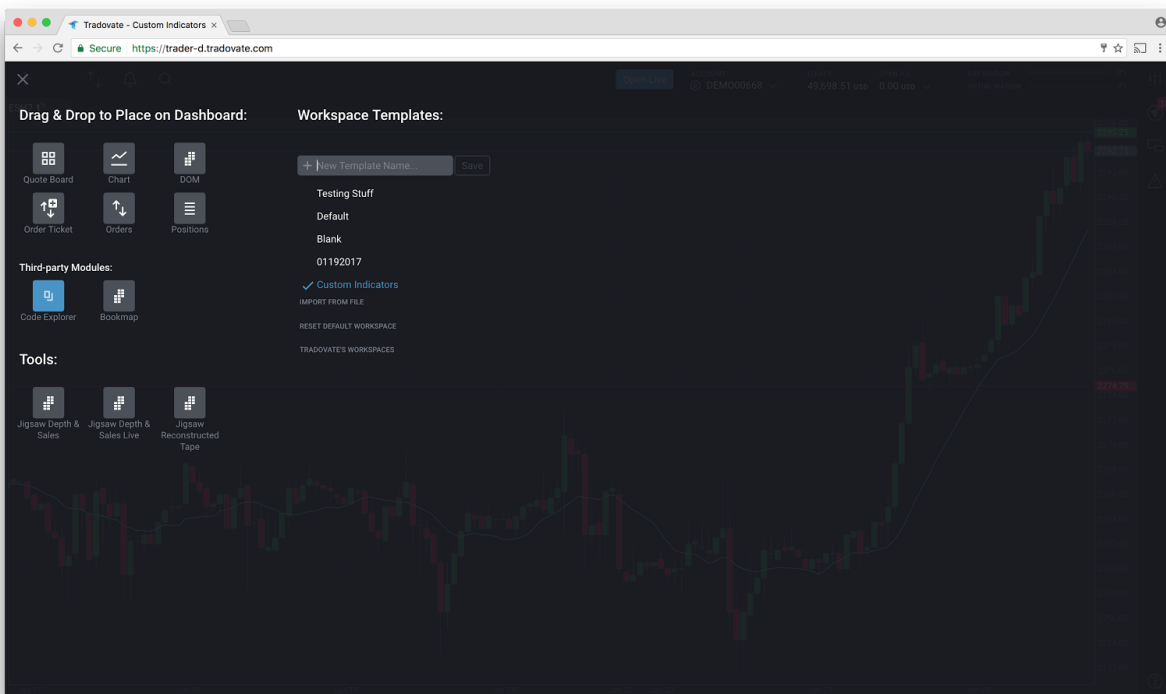


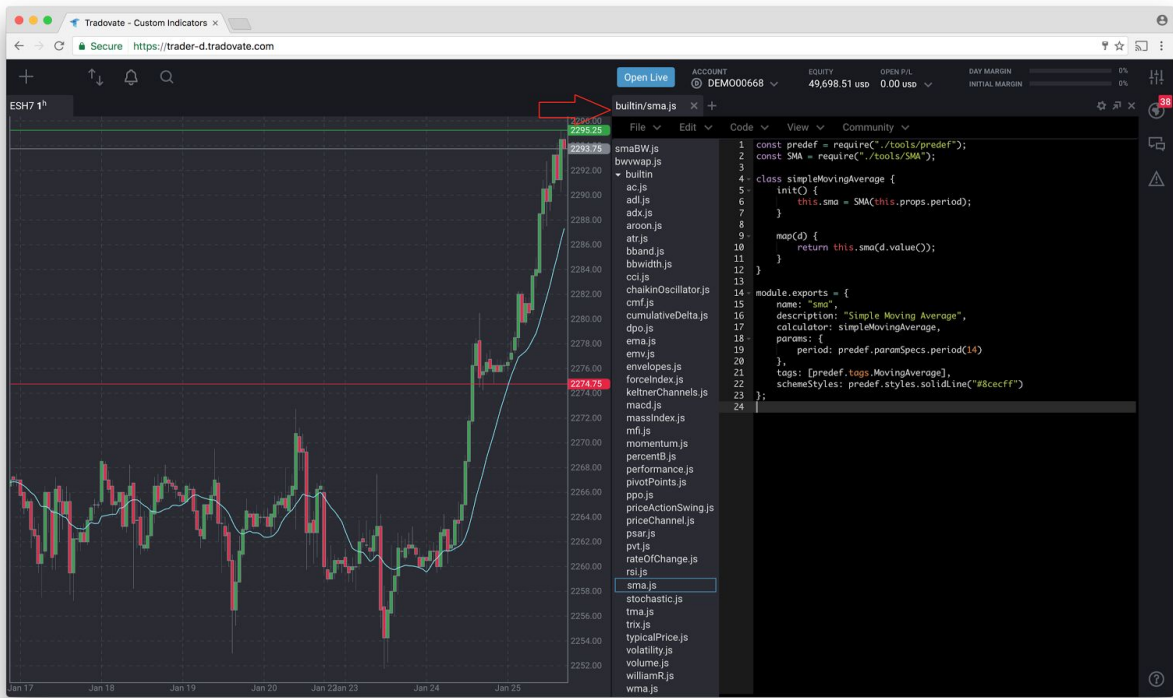
```

1 const predef = require("../tools/predef");
2 const SMA = require("../tools/SMA");
3
4 class simpleMovingAverage {
5   init() {
6     this.sma = SMA(this.props.period);
7   }
8
9   map(d) {
10    return this.sma(d.value());
11  }
12 }
13
14 module.exports = { };
24

```

- Additionally, the Code Editor can be added to the Layout as a module that can be positioned anywhere in the layout. This module can be found in the Workspace manager under the Third-party Modules





The screenshot displays the Tradovate Custom Indicators interface. On the left, a candlestick chart for ESH7 1h is shown with a blue Simple Moving Average (SMA) line. A red arrow points to the SMA line on the chart. The right side of the interface shows a code editor for the 'sma.js' file, which contains the following code:

```

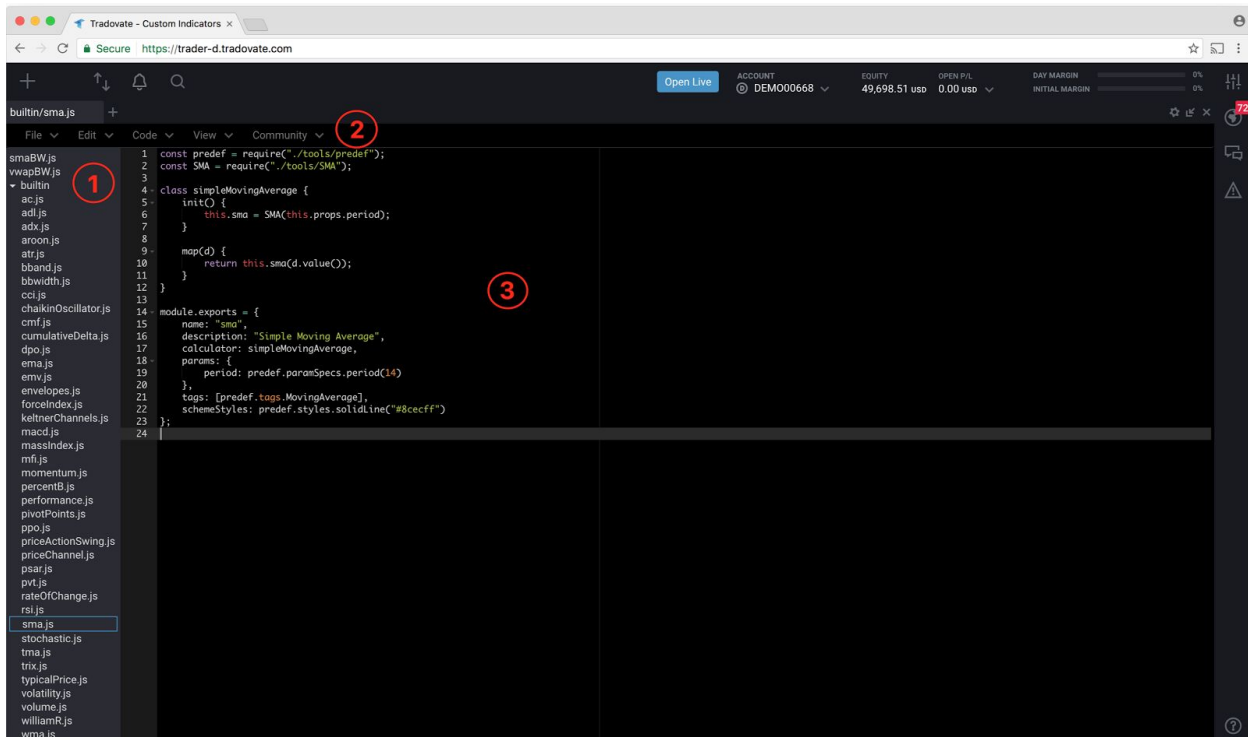
1 const predef = require("../tools/predef");
2 const SMA = require("../tools/SMA");
3
4 class simpleMovingAverage {
5   init() {
6     this.sma = SMA(this.props.period);
7   }
8   map(d) {
9     return this.sma(d.value());
10  }
11 }
12
13
14 module.exports = {
15   name: "sma",
16   description: "Simple Moving Average",
17   calculator: simpleMovingAverage,
18   params: {
19     period: predef.paramSpecs.period(14)
20   },
21   tags: [predef.tags.MovingAverage],
22   schemeStyles: predef.styles.solidLine("#8cecff")
23 };
24

```

Indicator Editor Overview

The editor is composed of 3 different areas:

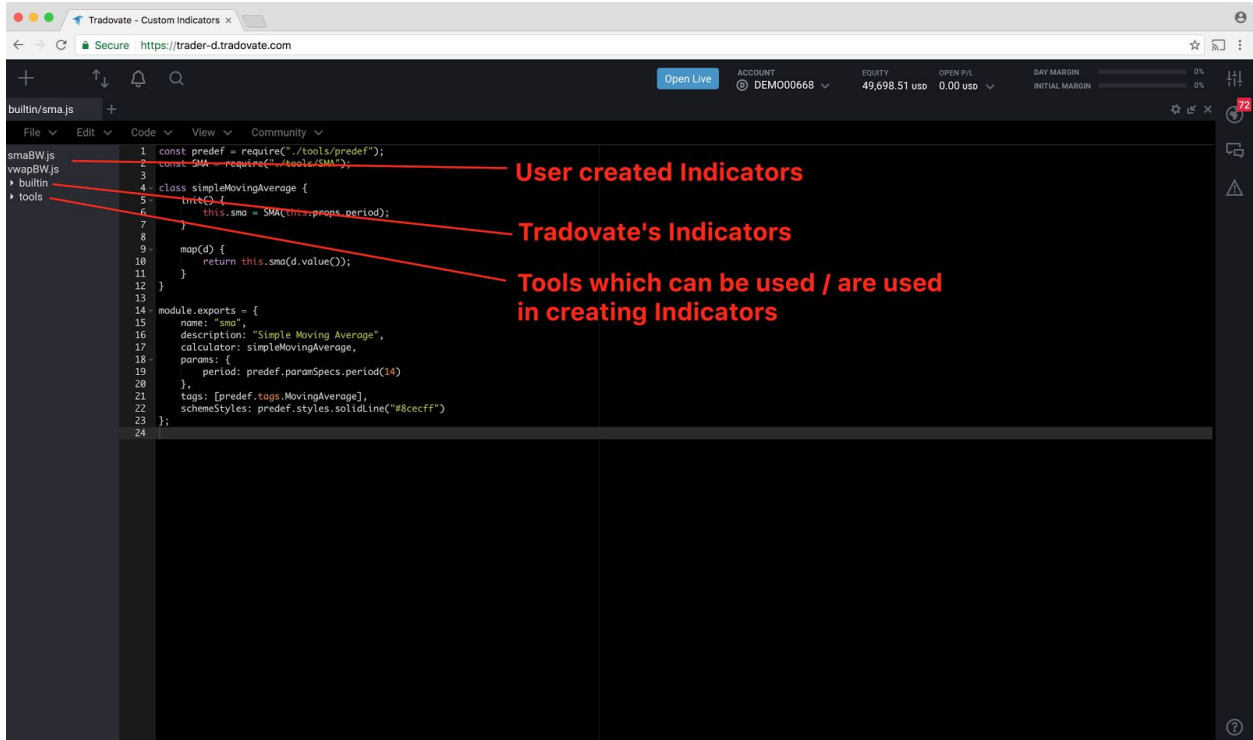
1. File List - Listing of prebuilt “Tradovate” Indicators
2. Menu - Common controls for file management, code editing as well as access to the Indicator Sharing Community
3. Code Editor



File List

The file list area is separated into 3 different areas:

1. User created indicators - Files that have been created by the user
2. Tradovate’s Indicators - Tradovates current list of indicators. Code is exposed for users to view / reuse
3. Tools which can be used / are used in creating indicators - A set of tools which are being used in Tradovate’s Indicators and can be used by individual users to create their own indicators



Additionally, if you install an indicator that another user posted to the community, those installs will show up in a separate section. More on that in the “Sharing Indicators” section

Menu

The Editor Menu contains common functions associated with code editing

- File - Common File management functions

Field	Description
New	Create New File
Save	Save changes to currently selected File
Save As	Save the File / changes as a New File
Save All	Save All changes made in all Files
Export	Export the current file to an Indicator JS file
Import	Import an Indicator JS File
Delete	Delete the selected file

- Edit - Common editing functions

Field	Description
Undo	Undo last change
Redo	Redo last deleted change
Select All	Select all text in the Editor
Find	Find term
Find Next	Find next term
Find Previous	Find previous term
Replace	Replace selected value with entered term
To Lower Case	Convert selected text to lower case
To Upper Case	Convert selected text to upper case

- Code - Common Code Editing / Formatting functions

Field	Description
Go to Next Error	Move to next error in code
Go to Previous Error	Move to previous error in code
Block Indent	Indent selected text
Block Outdent	Outdent selected text
Fold	Selectively Hide Foldable Code
Fold All	Hide All Foldable Code
Unfold	Selectively Show Hidden Code
Unfold All	Show All Hidden Code
Toggle Block Comment	Multiline Comment

- View - Functions allowing you to Show/Hide the file tree, Show/Hide the console, Show/Hide syntax errors

Field	Description
-------	-------------

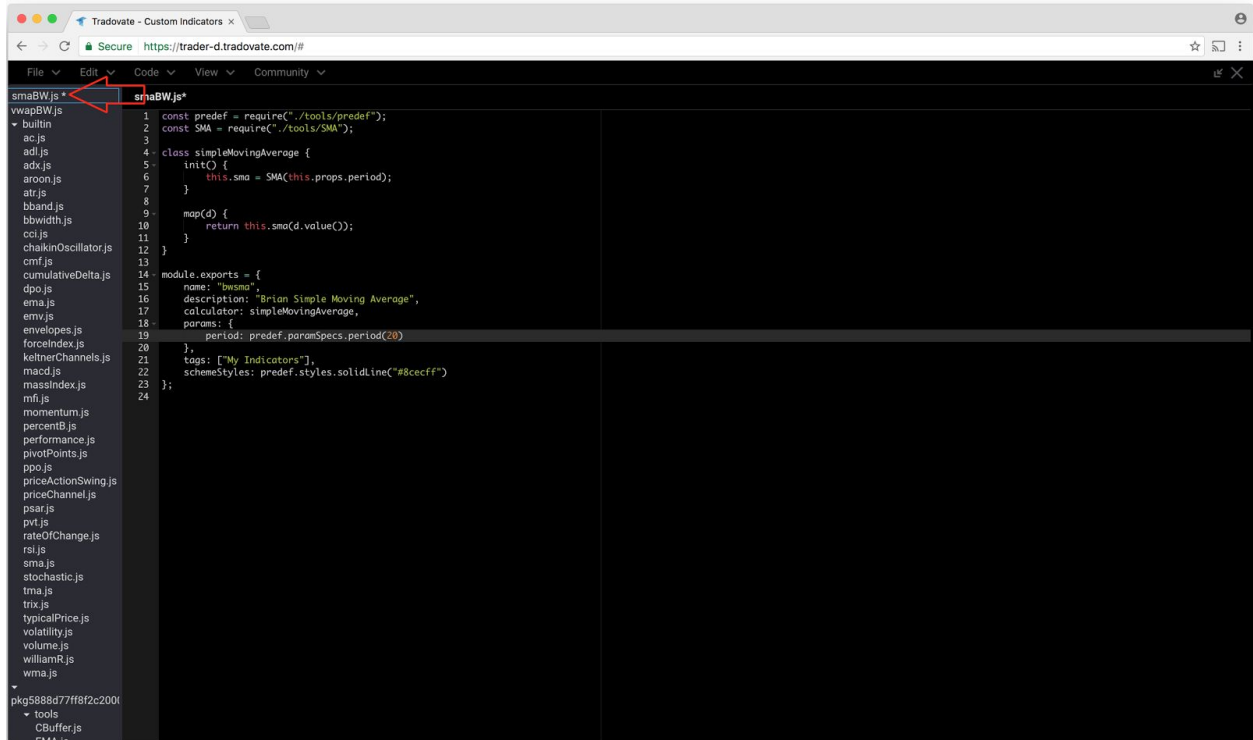
Console Output	Show/Hide the Console Output view
Syntax Errors	Show/Hide the Syntax Error view
File Tree	Show/Hide the File Tree
Clear Console	Clear Console messages

- Community - Provides access to an Indicator Sharing community allowing you to view indicators which have been shared and share your own indicators. More on these functions in the “Sharing Indicators” section

Field	Description
Explore	Show all indicators that have been shared by Tradovate Users
Installed	Show indicators that have been installed from Tradovate Community Members
Share	Share an Indicator that you have developed
My Entries	Show all indicators you have shared with the Tradovate Community

Code Editor

The code editor will display the JavaScript code of the selected indicator allowing the user to edit the code. If changes have been made to the indicator, the file will be flagged with a star after the file name in the file tree indicating changes were made and have not yet been saved.



```

1 const predef = require("../tools/predef");
2 const SMA = require("../tools/SMA");
3
4 class simpleMovingAverage {
5   init() {
6     this.sma = SMA(this.props.period);
7   }
8
9   map(d) {
10    return this.sma(d.value());
11  }
12 }
13
14 module.exports = {
15   name: "sma",
16   description: "Brien Simple Moving Average",
17   calculator: simpleMovingAverage,
18   params: {
19     period: predef.paramSpecs.period(20)
20   },
21   tags: ["My Indicators"],
22   schemeStyles: predef.styles.solidLine("#8cecf")
23 };
24

```

Once changes are made and saved the indicator will be updated real-time and changes can be viewed directly on the chart.

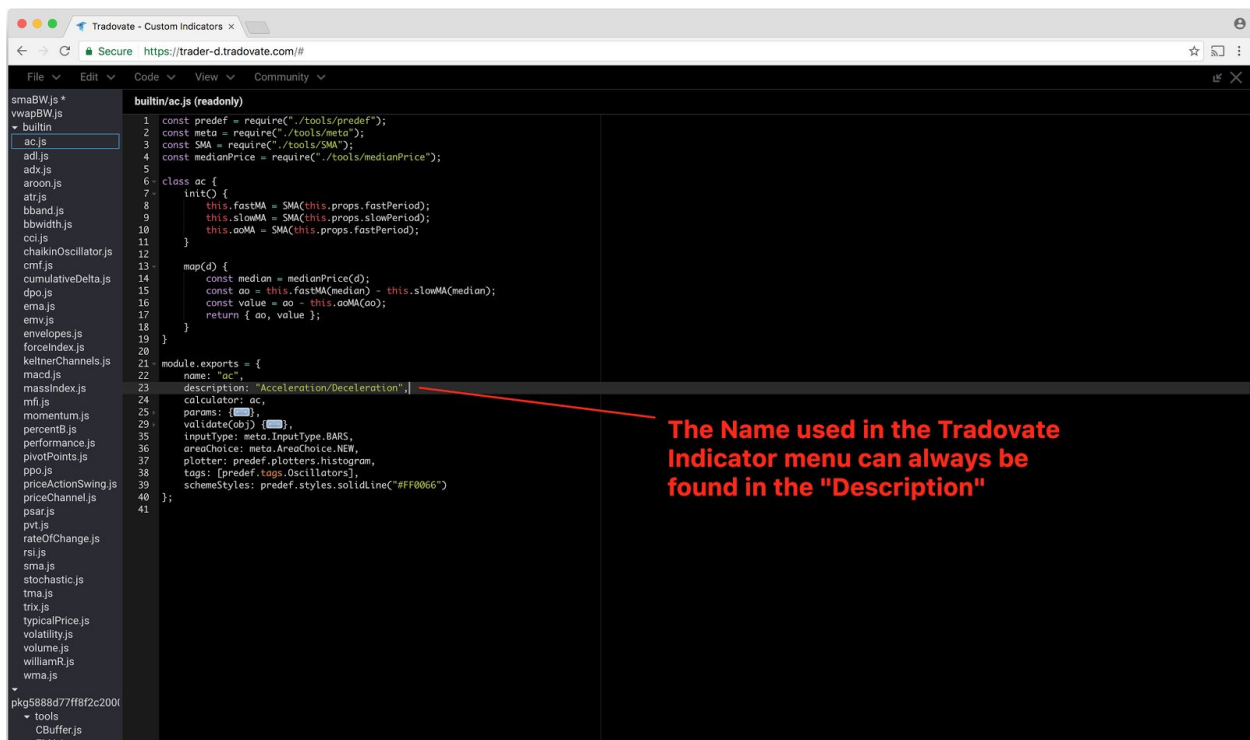
View Tradovate's Indicator Code

The code for all of the current indicators (as well as Tools used for indicator construction) available in Tradovate Charts (as well as any new indicators that are added in the future) can be viewed / copied / referenced / reused.

Tradovate's Indicators

The code for the indicators can be accessed by:

1. Make sure View>File Tree is enabled
2. Select "Builtin" to expand this section of the file tree if it is not already
3. Select the indicator you would like to view
4. The underlying code for that indicator will be displayed in the Code Editor



```

1  const predef = require("../tools/predef");
2  const meta = require("../tools/meta");
3  const SMA = require("../tools/SMA");
4  const medianPrice = require("../tools/medianPrice");
5
6  class ac {
7    init() {
8      this.fastMA = SMA(this.props.fastPeriod);
9      this.slowMA = SMA(this.props.slowPeriod);
10     this.aoMA = SMA(this.props.fastPeriod);
11   }
12
13   map(d) {
14     const median = medianPrice(d);
15     const ao = this.fastMA(median) - this.slowMA(median);
16     const value = ao - this.aoMA(ao);
17     return { ao, value };
18   }
19 }
20
21 module.exports = {
22   name: "ac",
23   description: "Acceleration/Deceleration",
24   calculator: ac,
25   params: {
26     fastPeriod: { type: "number", value: 14 },
27     slowPeriod: { type: "number", value: 34 },
28     fastPeriod2: { type: "number", value: 14 },
29   },
30   inputTypes: meta.InputType.BARS,
31   areaChoice: meta.AreaChoice.NEW,
32   plotter: predef.plotters.histogram,
33   tags: [predef.tags.Oscillators],
34   schemeStyles: predef.styles.solidLine("#FF0066")
35 };

```

The Name used in the Tradovate Indicator menu can always be found in the "Description"

Tools

The tools are a prebuilt toolkit that can be used to build any indicators. These are referenced in the Tradovate Indicators and can be referenced by users in their own Custom Indicators.

If additional inputs are required, users can reference any files they choose to create and use.

Managing Menu Structure

Indicator placement inside the Tradovate Indicator menu menu structure is determined by “tags” under module.exports

The screenshot displays the Tradovate Custom Indicators interface. On the left, a menu structure is visible with a red arrow pointing to the 'Volume-based' category. The main area shows a candlestick chart for ESH7 1m. On the right, a code editor displays the JavaScript code for the 'vwap.js' indicator. The code includes a 'module.exports' object with a 'tags' property set to '["predef.tags.Volumes"]', which determines its placement in the menu. A red arrow points to this 'tags' property in the code.

```

module.exports = {
  name: "vwap",
  description: "VWAP",
  calculator: vwap,
  inputType: meta.InputType.BARS,
  tags: ["predef.tags.Volumes"],
  schemeStyles: predef.styles.solidLine("#8cecff")
}

```

Importing and Exporting Indicators

User can export files created in Tradovate Custom Indicators. To export an indicator file, select the file from the file tree and select File>Export and the download process will begin.

To import an indicator, select File>Import, select the .js file to import

