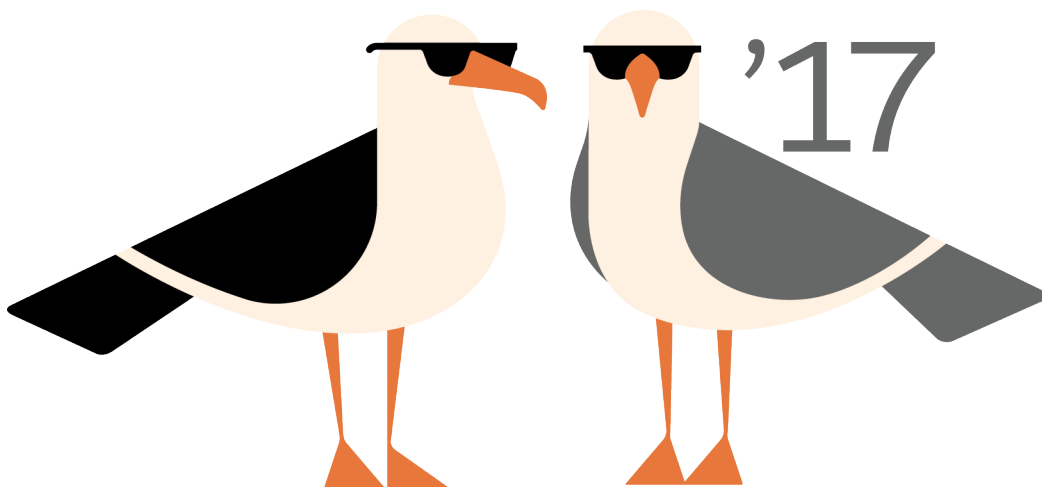


---

# Force.com IDE 2 Developer Guide (Beta)

Version 40.0, Summer '17





# CONTENTS

<b>Chapter 1: Get Started with Force.com IDE 2 (Beta)</b>	<b>1</b>
Force.com IDE 2 System Requirements	2
Force.com IDE 2 Setup Prerequisites	2
Install Force.com IDE 2	2
Add a Force.com IDE 2 Update Site for Eclipse Oxygen	3
Specify an External Browser	3
Update Force.com IDE 2	4
Set Up a Project in Force.com IDE 2	4
Create an Empty Salesforce DX Project in Force.com IDE 2	5
Create a Force.com IDE 2 Project from Source on Disk	5
Clone a Repository and Create an Eclipse Project with Force.com IDE 2	6
<b>Chapter 2: Build Your App in Force.com IDE 2</b>	<b>7</b>
Write Code in Force.com IDE 2	8
Work with Scratch Orgs in Force.com IDE 2	8
Create Source Files in Force.com IDE 2	10
Open Lightning App Builder in Force.com IDE 2	10
Write Better Code in Force.com IDE 2	11
Lightning Code Analyzer	11
Apex Code Analysis in Force.com IDE 2	13
Run Apex Tests in Force.com IDE 2	14
<b>Chapter 3: Force.com IDE 2 Troubleshooting</b>	<b>18</b>
Obtain Error Logs and Version Information in Force.com IDE 2	19
Find Missing Lightning Framework Features in Force.com IDE 2	19
Non-Executing Commands in Force.com IDE 2	19
Unable to Work After Failed Org Authorization	19
<b>Chapter 4: Open-Source Acknowledgments for Force.com IDE 2</b>	<b>21</b>
<b>Chapter 5: Resources for Force.com IDE 2 Users</b>	<b>22</b>
<b>Chapter 6: Considerations for Force.com IDE 2 (Beta)</b>	<b>23</b>



# CHAPTER 1    Get Started with Force.com IDE 2 (Beta)

## In this chapter ...

- [Force.com IDE 2 System Requirements](#)
- [Force.com IDE 2 Setup Prerequisites](#)
- [Install Force.com IDE 2](#)
- [Add a Force.com IDE 2 Update Site for Eclipse Oxygen](#)
- [Specify an External Browser](#)
- [Update Force.com IDE 2](#)
- [Set Up a Project in Force.com IDE 2](#)

The original Force.com IDE was designed for traditional Salesforce development and deployment. Force.com IDE 2 is tailor-made for Salesforce DX. It uses the Salesforce CLI plug-in to interact with your scratch orgs and works with version control systems.



**Note:** This release contains a beta version of Force.com IDE 2, which means it's a high-quality feature with known limitations. Force.com IDE 2 isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for Force.com IDE 2 in the [Salesforce DX Beta](#) group in the Success Community.

# Force.com IDE 2 System Requirements

---

Before you install Force.com IDE 2, make sure that your system meets these requirements.

## Supported Operating Systems

### Windows

Microsoft Windows 7 (64-bit) or later

To use the embedded browser, Windows 10

### macOS

macOS 10.11 or later

### Linux

Ubuntu 14.0.4 or later

To use the embedded browser, major versions of Linux, such as Ubuntu 16.04 or Centos 7

## Force.com IDE 2 Setup Prerequisites

---

Before you install Force.com IDE 2, make sure that you've installed the Salesforce CLI and the items on which the CLI and IDE depend.

### Salesforce CLI

Each Force.com IDE 2 version depends on a matched version of the Salesforce CLI. Install the Salesforce CLI and its prerequisites before installing Force.com IDE 2.

For Salesforce CLI installation instructions, see "[System Requirements](#)" and "[Install the Salesforce Command Line Interface \(CLI\)](#)" in the *Salesforce DX Setup Guide (Beta)*.

### Java 8 Platform, Standard Edition Development Kit

Force.com IDE 2 is built on the Eclipse platform. Eclipse requires the Java 8 Platform, Standard Edition Development Kit (JDK).

If you don't already have the JDK installed, install the latest version of the Java 8 JDK from [Java SE Development Kit 8 Downloads](#).

SEE ALSO:

[Salesforce DX Setup Guide \(Beta\)](#)

## Install Force.com IDE 2

---

You can use our installer to install Force.com IDE 2.

Before you install Force.com IDE 2, make sure that your system is set up with the items listed in [Force.com IDE 2 System Requirements](#) and [Force.com IDE 2 Setup Prerequisites](#).

1. For macOS only, ensure that you're allowing apps downloaded from the Mac App Store and identified developers. Select **System Settings > Security & Privacy**.
2. Download the IDE.

### Linux

[https://developer.salesforce.com/media/force-ide/sfdx/beta/force-ide-Beta-linux.gtk.x86\\_64.tar.gz](https://developer.salesforce.com/media/force-ide/sfdx/beta/force-ide-Beta-linux.gtk.x86_64.tar.gz)

**macOS**

[https://developer.salesforce.com/media/force-ide/sfdx/beta/force-ide-Beta-macosx.cocoa.x86\\_64.tar.gz](https://developer.salesforce.com/media/force-ide/sfdx/beta/force-ide-Beta-macosx.cocoa.x86_64.tar.gz)

**Windows**

[https://developer.salesforce.com/media/force-ide/sfdx/beta/force-ide-Beta-win32.win32.x86\\_64.zip](https://developer.salesforce.com/media/force-ide/sfdx/beta/force-ide-Beta-win32.win32.x86_64.zip)

3. Unzip and install the IDE.
4. If you're using macOS 10.12 Sierra, move `Force IDE.app` to the `Applications` directory after you install it. Otherwise, you can't update to new IDE versions.
5. Launch the IDE.
6. If you're upgrading from an earlier version of Force.com IDE 2, to see the latest features, select **Window > Perspective > Reset Perspective**.

## Add a Force.com IDE 2 Update Site for Eclipse Oxygen

---

If you're already using Eclipse Oxygen, you can install or upgrade our plug-in without losing your customizations. Don't install Force.com IDE 2 in an Eclipse instance that has the classic version of Force.com IDE installed. Instead, use a new instance of Eclipse.

Before you install Force.com IDE 2, make sure that your system has the items listed in [Force.com IDE 2 System Requirements](#) and [Force.com IDE 2 Setup Prerequisites](#).

1. Launch Eclipse, and select **Help > Install New Software**.
2. In the Add Repository dialog, set the name to *Force.com IDE 2* and the location to:

`https://developer.salesforce.com/media/force-ide/sfdx`

3. Click **OK**.  
Eclipse downloads the plug-ins and displays them in the Available Software dialog.
4. Select **Group items by category**.
5. Select **Salesforce**, and then click **Next**.
6. In the Install Details dialog, click **Next** or **Finish**.
7. If you see a Review Licenses dialog, accept the terms and click **Finish**.
8. Eclipse displays a warning dialog about installing software that contains unsigned content. We are bundling third-party plug-ins to support Lightning components. Salesforce doesn't own these third-party plug-ins, so we don't sign them. Click **OK** to proceed.
9. When the installation is complete, you are prompted to restart. Click **Yes**.
10. When Eclipse restarts, select **Window > Perspective > Open Perspective > Other**. Double-click **Salesforce DX**.
11. If you're upgrading from an earlier version of Force.com IDE 2, to see all the latest features, select **Window > Perspective > Reset Perspective**.

## Specify an External Browser

---

If you're using an operating system for which Force.com IDE 2 isn't fully supported, such as Windows 7 or 8, specify an external web browser. Force.com IDE 2 launches this browser when you invoke Lightning App Builder from within the IDE.

1. Select **Force.com IDE > Preferences > General > Web Browser**.

If you installed Force.com IDE 2 into an existing instance of Eclipse rather than using our installer, select **Eclipse > Preferences > General > Web Browser**.

2. Select **Use external web browser**.
3. To use your system's default web browser, select **Default system web browser**. To use a different web browser, select **New > Browse**.

SEE ALSO:

[Force.com IDE 2 System Requirements](#)

## Update Force.com IDE 2

---

Update your Force.com IDE 2 installation to the latest IDE version.

- You can enable, disable, or change the frequency of automatic updating. Select **Force.com IDE > Preferences > Install/Update > Automatic Updates**.  
If you installed Force.com IDE 2 into an existing instance of Eclipse rather than using our installer, select **Eclipse > Preferences > Install/Update > Automatic Updates**.
- You can also manually check whether an update is available. Select **Help > Check for Updates**.

## Set Up a Project in Force.com IDE 2

---

Whether you're starting from a source control repository, from Salesforce DX–formatted source on disk, or from scratch, Force.com IDE 2 has tools to help you set up a Salesforce DX project.

[Create an Empty Salesforce DX Project in Force.com IDE 2](#)

You can create a Salesforce DX project from scratch.

[Create a Force.com IDE 2 Project from Source on Disk](#)

If you're not using Git, if you're using a version control system (VCS) other than Git, or if you don't want to use the tools in the Git Repositories panel, set up a project from source that's stored on your computer. Before setting up a project, your source must be in Salesforce DX format.

[Clone a Repository and Create an Eclipse Project with Force.com IDE 2](#)

Force.com IDE 2 has built-in tools for working with Git repositories. You can also use a different version control system (VCS), but Git is the VCS we've provided tooling for. Use Force.com IDE 2 to work with source from a Git repository that's in the Salesforce DX format. To work with org metadata that you haven't converted to the Salesforce DX format, use the Salesforce CLI to retrieve your metadata and set up a Salesforce DX project. For information on retrieving and converting your metadata, see the Salesforce DX Developer Guide (Beta).

SEE ALSO:

[Salesforce DX Developer Guide \(Beta\): Create a Salesforce DX Project from Existing Source](#)

[Salesforce DX Developer Guide \(Beta\): Retrieve Source from an Existing Managed Package](#)

[Salesforce DX Developer Guide \(Beta\): Retrieve Unpackaged Source Defined in a package.xml File](#)

[Salesforce DX Developer Guide \(Beta\): Retrieve Unpackaged Source by Creating a Temporary Unmanaged Package](#)

[Salesforce DX Developer Guide \(Beta\): Convert the Metadata API Source](#)



## Create an Empty Salesforce DX Project in Force.com IDE 2

You can create a Salesforce DX project from scratch.

1. Select **File > New > SFDX Project**.
2. To select a location on your local file system to store your new project, click **Browse**.
3. Enter a name for your project.
4. (Optional) If you want to change the name of the default package directory, enter the name in Default Project.
5. (Optional) Enter a value for Associated Namespace.



**Important:** If you're creating an exploratory project, choose a disposable namespace. Don't choose a namespace that you want to use in the future for a production org. If you're registering a namespace for a production org, choose it carefully. After you associate a namespace with an org, you can't change it or reuse it.

6. (Optional) If you use a login instance other than `https://login.salesforce.com`—for example, `https://test.salesforce.com`—enter it in Login URL.



**Tip:** You can change your login URL later, by editing your `sfdx-project.json` file.

7. Click **Finish**.
8. (Optional) To use Git for version control, complete the steps in [Basic Tutorial: Adding a project to version control](#) in the Eclipse documentation.

If you added an associated namespace, you must [register the namespace with Salesforce](#).

## Create a Force.com IDE 2 Project from Source on Disk

If you're not using Git, if you're using a version control system (VCS) other than Git, or if you don't want to use the tools in the Git Repositories panel, set up a project from source that's stored on your computer. Before setting up a project, your source must be in Salesforce DX format.

If you're using source that you retrieved using `mdapi:retrieve`, you must convert it. See [Convert the Metadata API Source](#) in the *Salesforce DX Developer Guide (Beta)*.

1. Select **File > New > Project > General > Project > Next**.
2. Enter a project name.
3. Deselect **Use default location**.
4. To set the location to the directory where your source is stored, click **Browse** to navigate to the directory and then click **Open**.
5. In the location field, append the name of your project to the path. For example, to place the myProject project in `/Users/me/development/`, change the location to `/Users/me/development/myProject`.
6. Click **Finish**.
7. In the Project Explorer, right-click your project and select **Configure > Enable Lightning Support**.
8. (Optional) To use Git for version control, complete the steps in [Basic Tutorial: Adding a project to version control](#) in the Eclipse documentation.

## Clone a Repository and Create an Eclipse Project with Force.com IDE 2

Force.com IDE 2 has built-in tools for working with Git repositories. You can also use a different version control system (VCS), but Git is the VCS we've provided tooling for. Use Force.com IDE 2 to work with source from a Git repository that's in the Salesforce DX format. To work with org metadata that you haven't converted to the Salesforce DX format, use the Salesforce CLI to retrieve your metadata and set up a Salesforce DX project. For information on retrieving and converting your metadata, see the Salesforce DX Developer Guide (Beta).

### [Clone a Git Repository with Force.com IDE 2](#)

Cloning a Git repo using Force.com IDE 2's built-in Git Repositories view removes much of the guesswork involved in setting up a project.

### [Create a Force.com IDE 2 Project from a Git Repository](#)

If you didn't create a project when you set up a Git repository, create one from the cloned repository. If you created a Force.com IDE 2 project when you cloned a Git repository, you don't need to create one now.

## Clone a Git Repository with Force.com IDE 2

Cloning a Git repo using Force.com IDE 2's built-in Git Repositories view removes much of the guesswork involved in setting up a project.

1. In the Git Repositories panel, click **Clone a Git repository**.
2. Enter the URI for the repo that contains your Salesforce DX source.
3. Click **Next**, choose the branches to clone, and click **Next** again.
4. Set the destination directory to the location where you want to store your source.
5. If your repo contains a `.project` file, to set up an Eclipse project using the cloned source, select **Import all existing Eclipse projects after clone finishes**.
6. Click **Finish**.
7. If you imported an existing project, when your new project is ready, in the Project Explorer, right-click your project and then select **Configure > Enable Lightning Support**.

## Create a Force.com IDE 2 Project from a Git Repository

If you didn't create a project when you set up a Git repository, create one from the cloned repository. If you created a Force.com IDE 2 project when you cloned a Git repository, you don't need to create one now.

1. In the Git Repositories panel, right-click your repository.
2. Select **Import Projects**.
3. Deselect **Search for nested projects**.
4. Deselect **Detect and configure project natures**.  
By default, when your project contains Lightning components, Eclipse's EGit functionality creates a JavaScript-natured project. Deselecting this option enables Force.com IDE 2's Lightning Components features to work properly.
5. Click **Directory** and choose the directory that contains your repository.
6. In the Folder column, select your repository.
7. Click **Finish**.
8. In the Project Explorer, right-click your project and select **Configure > Enable Lightning Support**.

## CHAPTER 2 Build Your App in Force.com IDE 2

### In this chapter ...

- [Write Code in Force.com IDE 2](#)
- [Write Better Code in Force.com IDE 2](#)
- [Run Apex Tests in Force.com IDE 2](#)

Force.com IDE 2 provides features to help you write and analyze your code.

## Write Code in Force.com IDE 2

---

Force.com IDE 2 includes tools to help you interact with Salesforce DX scratch orgs, create and edit source files, and edit Lightning Pages.

### [Work with Scratch Orgs in Force.com IDE 2](#)

A scratch org is a temporary org used for building and testing app functionality as part of the Salesforce DX workflow. Scratch orgs are managed from a Dev Hub org. Force.com IDE 2 helps you manage which Dev Hub and scratch org are associated with a project. You also use the IDE to push and pull source to and from your scratch org.

### [Create Source Files in Force.com IDE 2](#)

Use Force.com IDE 2 to add Apex classes, Visualforce pages and components, and Lightning bundles to your projects.

### [Open Lightning App Builder in Force.com IDE 2](#)

Sometimes a browser-based editor is the best tool for the job. For example, Lightning App Builder is the best tool for editing Lightning Pages (also known as FlexiPages). Force.com IDE 2 includes an embedded browser for editing Lightning Pages.

SEE ALSO:

[Write Better Code in Force.com IDE 2](#)

## Work with Scratch Orgs in Force.com IDE 2

A scratch org is a temporary org used for building and testing app functionality as part of the Salesforce DX workflow. Scratch orgs are managed from a Dev Hub org. Force.com IDE 2 helps you manage which Dev Hub and scratch org are associated with a project. You also use the IDE to push and pull source to and from your scratch org.

### [Authorize a Dev Hub in Force.com IDE 2](#)

Associate a Dev Hub org with each Force.com IDE 2 project that you create. The IDE uses the Dev Hub to create and manage the scratch orgs that you use with the project.

### [Create a Scratch Org in Force.com IDE 2](#)

Create a scratch org for each Force.com IDE 2 project. Use a scratch org to test your changes and use browser-based editors, such as Lightning App Builder.

### [Authorize a Scratch Org in Force.com IDE 2](#)

You can change your project's default scratch org. For example, switch to a previous scratch org when you're working on a branch of code that was associated with that org. To use a different scratch org, authorize it for the project.

### [Move Source to and from Scratch Orgs in Force.com IDE 2](#)

After you've authorized a scratch org for your project, so that you can view and test your application, push your source to your scratch org. As you make local changes, push them to the org so that you can test the changes. After you make changes using a browser-based editor, pull the changes from your scratch org to your project.

### [Delete a Scratch Org in Force.com IDE 2](#)

When you finish a development task for your project, you're ready to move on from your scratch org. Pull all your changes from the scratch org to your project. Optionally, commit the changes to your version control system. Then you can delete the org. Scratch orgs are periodically deleted automatically, so it's best to delete them on your terms.

SEE ALSO:

[Salesforce DX Developer Guide \(Beta\): Scratch Orgs](#)

## Authorize a Dev Hub in Force.com IDE 2

Associate a Dev Hub org with each Force.com IDE 2 project that you create. The IDE uses the Dev Hub to create and manage the scratch orgs that you use with the project.

To set up a Dev Hub org, see [Enable the Dev Hub in Your Org](#) in the *Salesforce DX Setup Guide (Beta)*.

1. In the Project Explorer, right-click your Salesforce DX project, then select **Salesforce DX > Authorize a Dev Hub**.
2. In your browser, complete the login flow using your Dev Hub org's credentials.

## Create a Scratch Org in Force.com IDE 2

Create a scratch org for each Force.com IDE 2 project. Use a scratch org to test your changes and use browser-based editors, such as Lightning App Builder.

Before associating a scratch org with your project, [Authorize a Dev Hub in Force.com IDE 2](#).

1. In the Project Explorer, right-click your Salesforce DX project, then select **Salesforce DX > Create Scratch Org**.
2. Click **Browse** and double-click a scratch org definition file. Double-click **project-scratch-def.json** to create a scratch org set up for basic development.
3. Click **Finish**.  
Force.com IDE 2 uses the Salesforce CLI to create a scratch org for the Dev Hub that's associated with your project. The new org is the default scratch org for your project.

## Authorize a Scratch Org in Force.com IDE 2

You can change your project's default scratch org. For example, switch to a previous scratch org when you're working on a branch of code that was associated with that org. To use a different scratch org, authorize it for the project.

Before you associate a scratch org with your project, [Authorize a Dev Hub in Force.com IDE 2](#).

1. To set a password for your scratch org, from a terminal, run:

```
sfdx force:user:password:generate -u your.scratch.org@example.com
```

2. To see the password of your scratch org, from a terminal, run:

```
sfdx force:org:display -u your.scratch.org@example.com
```

3. In Force.com IDE 2, in the Project Explorer, right-click your Salesforce DX project and select **Salesforce DX > Authorize a Scratch Org**.
4. In your browser, complete the login flow using your scratch org's credentials.

## Move Source to and from Scratch Orgs in Force.com IDE 2

After you've authorized a scratch org for your project, so that you can view and test your application, push your source to your scratch org. As you make local changes, push them to the org so that you can test the changes. After you make changes using a browser-based editor, pull the changes from your scratch org to your project.

1. To push your project metadata to your scratch org, in the Project Explorer, right-click your project and select **Salesforce DX > Push Source**.
2. To push changes to your scratch org, save the changes, and then right-click your project and select **Salesforce DX > Push Source**.
3. To see how the pushed changes work, right-click your project and select **Salesforce DX > Open Org**.

The scratch org opens in your default browser.

4. To pull changes that you made in your browser into your project, right-click your project and select **Salesforce DX > Pull Source**.

## Delete a Scratch Org in Force.com IDE 2

When you finish a development task for your project, you're ready to move on from your scratch org. Pull all your changes from the scratch org to your project. Optionally, commit the changes to your version control system. Then you can delete the org. Scratch orgs are periodically deleted automatically, so it's best to delete them on your terms.

1. In the Project Explorer, right-click your Salesforce DX project.
2. To pull the changes from the scratch org into your project, select **Salesforce DX > Pull Source**.
3. To delete the project's default scratch org, select **Salesforce DX > Delete Scratch Org**.

## Create Source Files in Force.com IDE 2

Use Force.com IDE 2 to add Apex classes, Visualforce pages and components, and Lightning bundles to your projects.

1. In the Project Explorer, find the directory that you want to add to. For example, to create an Apex class in the DreamHouse app's `classes` directory, expand **force-app > main > default**, and then find **classes**. If the directory you want to add to doesn't exist yet, right-click the directory under which you want the new directory and select **New > Folder**.



**Tip:** Be sure to store your Lightning bundles in the `aura` directory.


2. Right-click the directory that you want to add to, select **New**, and then select the type of resource that you want to create. For example, to create an Apex class, select **New > Apex Class**.
3. Name the new resource, select a template, and then click **Finish**.  
The IDE uses the Salesforce CLI to create the resource and the associated metadata files.
4. When you're ready to push your new source to your default scratch org, right-click your project and select **Salesforce DX > Push Source**.

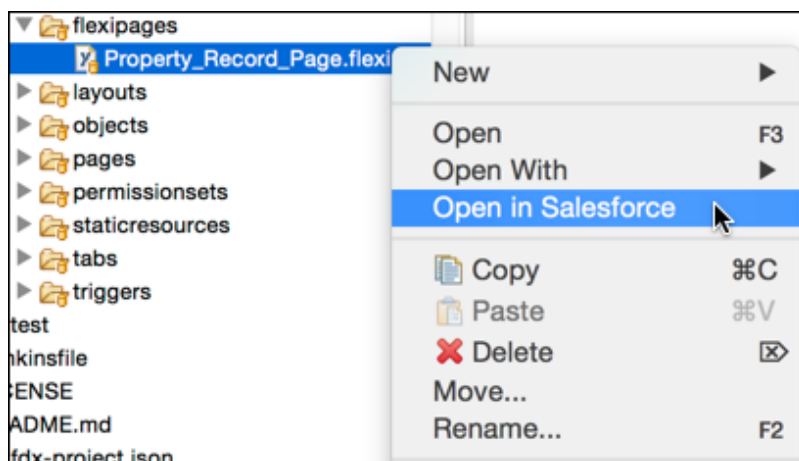
## Open Lightning App Builder in Force.com IDE 2

Sometimes a browser-based editor is the best tool for the job. For example, Lightning App Builder is the best tool for editing Lightning Pages (also known as FlexiPages). Force.com IDE 2 includes an embedded browser for editing Lightning Pages.

The embedded browser is supported only for some operating systems. For details, see [Force.com IDE 2 System Requirements](#). If the embedded browser isn't supported for your operating system, opening a Lightning Page in Salesforce opens your default browser.

1. In the Project Explorer, right-click your `.flexipage-meta.xml` file, then select **Open in Salesforce**.

 **Note:** The Open in Salesforce command is grouped with the other Open commands, not in the Salesforce DX submenu.



2. Edit your Lightning Page in your default scratch org using Lightning App Builder.
3. To pull the changes that you made into your project, right-click the project and select **Salesforce DX > Pull Source**.

## Write Better Code in Force.com IDE 2

---

Force.com IDE 2 includes features to help you improve the quality of your code.

### [Lightning Code Analyzer](#)

Lightning Code Analyzer uses the power of ESLint and the Salesforce Lightning CLI to identify problems in the JavaScript files that are part of your Lightning bundles.

### [Apex Code Analysis in Force.com IDE 2](#)

Force.com IDE 2 includes Apex syntax error detection and highlighting and code completion in Apex source files. The outline view shows the structure of your Apex classes and triggers. These features are available in any Salesforce DX project that includes a valid `sfdx-project.json` file that you import into the IDE.

## Lightning Code Analyzer

Lightning Code Analyzer uses the power of ESLint and the Salesforce Lightning CLI to identify problems in the JavaScript files that are part of your Lightning bundles.

### [Enable and Configure Lightning Code Analyzer](#)

Configuring Lightning Code Analyzer involves defining the analysis rules. You can choose from the predefined rule sets or create your own.

### [Analyze Your Lightning Components in Force.com IDE 2](#)

Lightning Code Analyzer analyzes the .js files from your Lightning bundles that are open in Force.com IDE 2. Identified problems in the files that you have open in the editor are shown inline in your source files and in the Lightning Code Analyzer view. Lightning Code Analyzer provides automatic fixes for some identified problems.

## Enable and Configure Lightning Code Analyzer

Configuring Lightning Code Analyzer involves defining the analysis rules. You can choose from the predefined rule sets or create your own.

1. To open the Lightning Code Analyzer preferences pane, select **Force.com IDE > Preferences > Lightning > Code Analyzer**. (If you added an update site to Eclipse instead of using our installer, select **Eclipse > Preferences > Lightning > Code Analyzer**.)
2. If Lightning Code Analyzer isn't already enabled, select **Analyze open Lightning components**.
3. Configure your analysis rules.
  - To use a predefined set of rules, select a rule set from the ESLint configuration dropdown menu.
  - To define your own rules, click **New**.

SEE ALSO:

[ESLint User Guide: Configuring ESLint](#)

[ESLint User Guide: Rules](#)

[Lightning Components Developer Guide: Salesforce Lightning CLI Rules](#)

## Analyze Your Lightning Components in Force.com IDE 2

Lightning Code Analyzer analyzes the .js files from your Lightning bundles that are open in Force.com IDE 2. Identified problems in the files that you have open in the editor are shown inline in your source files and in the Lightning Code Analyzer view. Lightning Code Analyzer provides automatic fixes for some identified problems.

- View errors and warnings for all your open files.
  - Select the **Lightning Code Analyzer** view in the bottom pane of the IDE.
  - Click a problem in the list to get more information.
  - To go to the problem's location in the code editor, double-click the problem.
- Find errors and warnings in a file at a glance.
  - Look for icons in the left margin of the editing pane—red for errors, yellow for warnings.
  - Hover over the icon to see a description of the problem.
  - To get more information, locate the dotted lines that indicate the problematic text and hover over the text.
- Identify problems in a file.
  - Hover over text that's underlined in red (for errors) or yellow (for warnings).
  - Click the links in the popup to see details about the problem. To prevent the popup from closing when you move your cursor, press F2.
- Implement Quick Fix solutions for some problems.
  - In the popup for an underlined problem, select **Quick Fix**.
  - To prevent the popup from closing when you move your cursor, press F2.

SEE ALSO:

[Enable and Configure Lightning Code Analyzer](#)



## Apex Code Analysis in Force.com IDE 2

Force.com IDE 2 includes Apex syntax error detection and highlighting and code completion in Apex source files. The outline view shows the structure of your Apex classes and triggers. These features are available in any Salesforce DX project that includes a valid `sfdx-project.json` file that you import into the IDE.



**Note:** Apex code analysis is powered by the Apex Language Server. Because the Apex Language Server uses a socket connection, it requires network access. The first time that you open an Apex class or trigger in Force.com IDE 2, some operating systems display a prompt asking whether you want to allow network connections. To enable Apex code analysis to function, allow this access.

### [Apex Code Analysis Features in Force.com IDE 2](#)

View outlines of Apex classes and triggers, see code-completion suggestions, and find syntactic errors in your code.

### [Apex Language Server \(Beta\)](#)

The Apex Language Server is an IDE-agnostic way for tools to access code-editing capabilities, such as code completion, go to definition, find all usage, and refactoring. It provides a powerful way for Force.com IDE 2 to implement an Apex analyzer that's also accessible to other IDEs.

## Apex Code Analysis Features in Force.com IDE 2

View outlines of Apex classes and triggers, see code-completion suggestions, and find syntactic errors in your code.

### View an Outline of Your Apex Class or Trigger

The Apex outline view shows the structure of the Apex class or trigger that's open in the editor. Two versions of the Apex outline view are available in Force.com IDE 2: a dedicated outline view and a quick outline view. The dedicated outline view persists in the bottom pane of the IDE. The quick outline view appears temporarily in the editor.

- To enable the dedicated outline view, select **Window > Show View > Outline**.
- To open the quick outline view, press `Cmd+Shift+O` (macOS) or `Ctrl+Shift+O` (Windows or Linux).

### View Code-Completion Suggestions

To see code-completion suggestions, press `Ctrl+Space` when you're working in a `.cls` or `.trigger` file. To navigate between the suggestions, use the arrow keys. To auto-complete a suggestion from the list, press `Enter`.


### See Syntax Errors in Your Code

If you forget to type a `;`, `}`, or `)`, the syntax error is marked with a red squiggly line in the editor. The Problems view in the bottom pane of the IDE also lists your syntax errors. Double-click the problem to go to the source file.

Eclipse also identifies spelling mistakes in your files, which it marks with a red dotted line. If you don't want to spell-check your code, select **Force.com IDE > Preferences > General > Editors > Text Editors > Spelling**, and deselect **Enable spell checking**. (If you added an update site to Eclipse instead of using our installer, select **Eclipse > Preferences > General > Editors > Text Editors > Spelling**, and deselect **Enable spell checking**.)

## Apex Language Server (Beta)

The Apex Language Server is an IDE-agnostic way for tools to access code-editing capabilities, such as code completion, go to definition, find all usage, and refactoring. It provides a powerful way for Force.com IDE 2 to implement an Apex analyzer that's also accessible to other IDEs.

 **Note:** This release contains a beta version of the Apex Language Server, which means it's a high-quality feature with known limitations. The Apex Language Server isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the Apex Language Server in the [Salesforce DX Beta](#) group in the Success Community.

Apex code analysis in Force.com IDE 2 is powered by the Apex Language Server. The Apex Language Server is an implementation of the Language Server Protocol 3.0 specification. The Language Server Protocol allows a tool (in this case, Force.com IDE 2) to communicate with a language smartness provider (the server). We built the Apex Language Server using this common specification to enable our tooling partners to use it to improve the smartness of their tools.

If you're a tooling provider and want to discuss working with the Apex Language Server, contact Josh Kaplan, senior director of product management, on Twitter at [@JoshSfdc](#).

SEE ALSO:

[Langserver.org: A community-driven source of knowledge for Language Server Protocol implementations](#)

[GitHub: Language Server Protocol](#)

[Eclipse Newsletter, May 2017: Language Server Protocol](#)

## Run Apex Tests in Force.com IDE 2

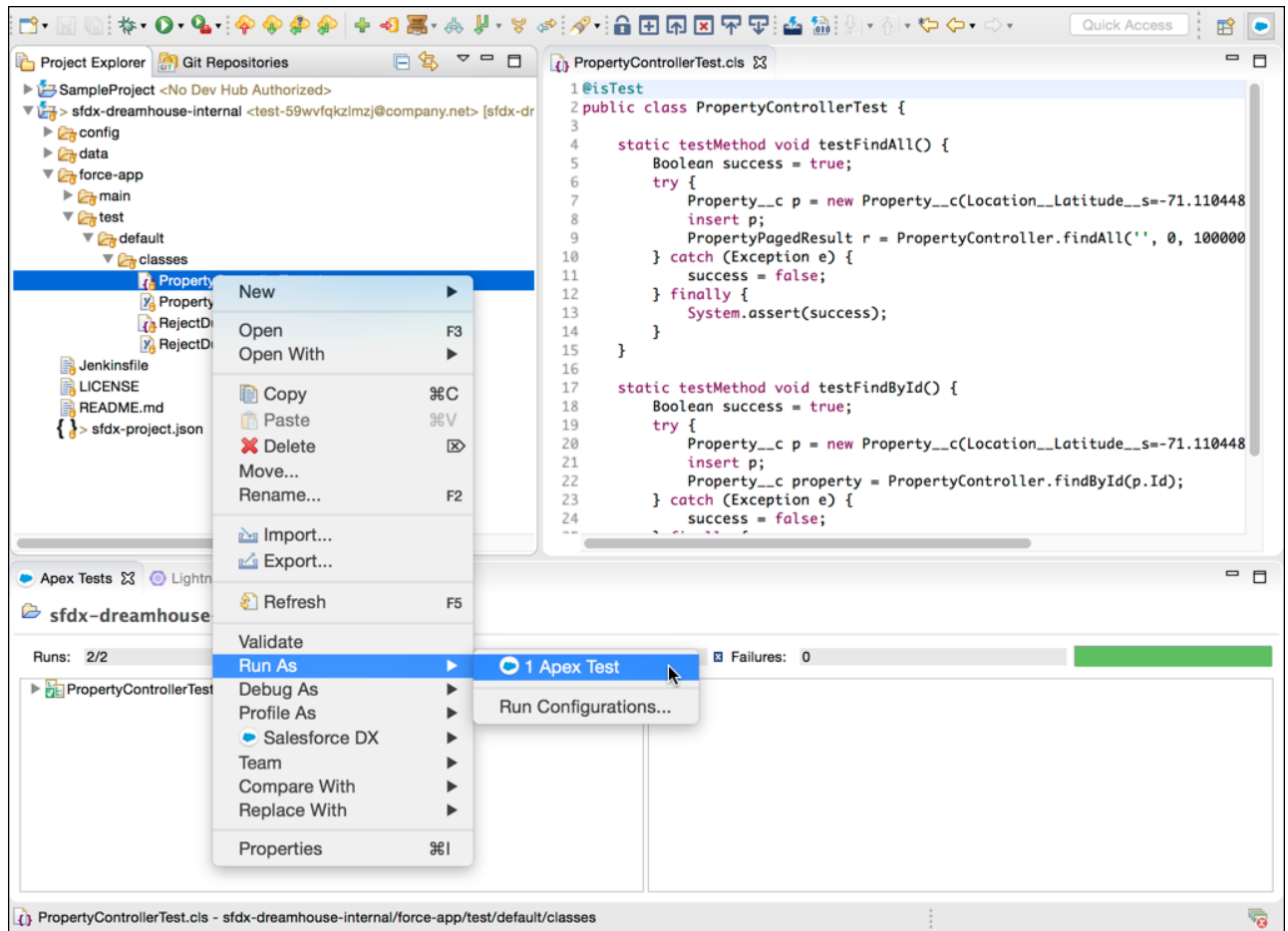
---

You can run all Apex tests in a class from your project's contextual menu or from the test class's source file. Or, you can run the methods in a combination of classes or suites. When complete, you can inspect your test results in the Apex Tests view.

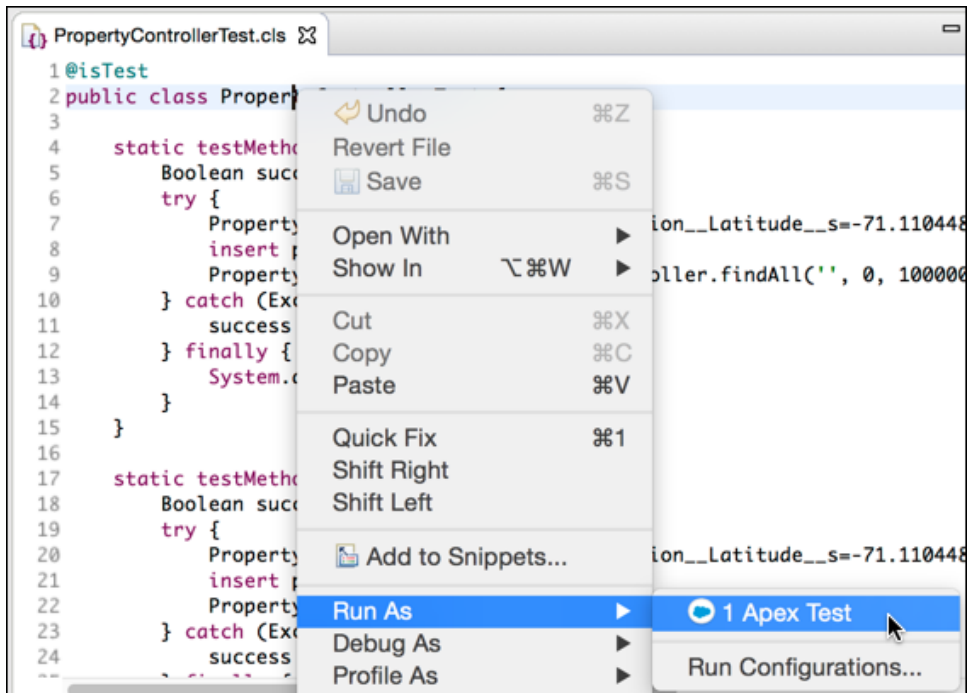
1. To push changes to your default scratch org, including changes to your Apex tests, right-click your project and select **Salesforce DX > Push Source**.
2. Run the Apex tests from the Project Explorer or from a test class, or run a combination of test classes or suites.

### From the Project Explorer:

- a. In the Project Explorer, navigate to your project's test classes directory. For example, in the DreamHouse project, expand **force-app > test > default > classes**.
- b. Right-click the test class `.cls` file.
- c. Select **Run As > Apex Test**.

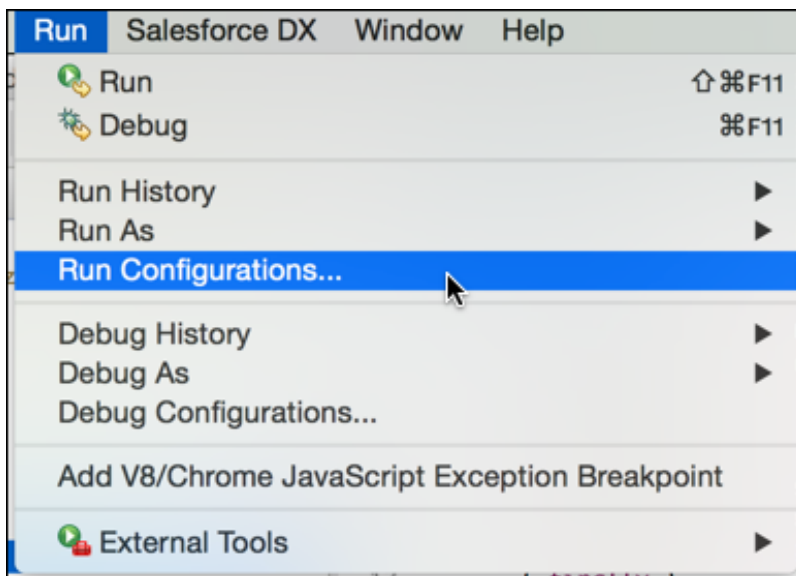
**Within a test class file:**


- a. Open your test class.
- b. Right-click anywhere in your code, and select **Run As > Apex Test**.



#### Combination of test classes or suites:

- a. Select **Run** > **Run Configurations**.



- b. In the left column, select **Force.com**, then click the New launch configuration icon (  ).
- c. Name your run configuration.
- d. If the name of your project isn't shown in the Project field, click **Browse** and select your project.
- e. From the Run dropdown menu, select either **Test Classes Selected Below**, **Test Suites Selected Below**, or **All My Tests and Managed Tests**. The All My Tests and Managed Tests option runs all your tests and the tests in your installed managed packages.

- f. If you selected **Test Classes Selected Below** or **Test Suites Selected Below**, select the classes or suites to include in the test run.
  - g. Click **Apply**.
  - h. To see which command the IDE has constructed to run your tests using the Salesforce CLI, click the **Summary** tab.
  - i. To run your tests, click **Run**.
3. Inspect your test results.
  - a. At the bottom of the IDE window, select the **Apex Tests** view.
  - b. To view the test results for individual test methods, expand the results for a test class.
  - c. To see details about an error or failure, click the name of a method with an X icon next to it.
  - d. If you had an active trace flag during your test run, you can view the debug log for the test run. To open the debug log, right-click a test class or method in the test results and select **View Apex Log**.
  - e. To open the test class that was part of the test run, double-click the name of a class or method in the results list.

## CHAPTER 3 Force.com IDE 2 Troubleshooting

### In this chapter ...

- Obtain Error Logs and Version Information in Force.com IDE 2
- Find Missing Lightning Framework Features in Force.com IDE 2
- Non-Executing Commands in Force.com IDE 2
- Unable to Work After Failed Org Authorization

Refer to these tips to help you use Force.com IDE 2.

## Obtain Error Logs and Version Information in Force.com IDE 2

---

When you contact Salesforce or request help from the community, share your CLI and IDE version information and the contents of your error log.

- To obtain Force.com IDE 2 error logs, select **Window > Show View > Error Log**.
- To check which Force.com IDE 2 version you're running:
  - On macOS, select **Force.com IDE > About Force.com IDE**.
  - On Windows or Linux, select **Help > About Force.com IDE**.
  - If you added an update site for Eclipse instead of running the Force.com IDE 2 installer:
    - On macOS, select **Eclipse > About Eclipse > Installation Details**.
    - On Windows or Linux, select **Help > About Eclipse > Installation Details**.
- To check which Salesforce DX CLI version you're using, run this command from a terminal:

```
sfdx plugins
```

- To report issues with Apex code analysis, provide a copy of your `apex.log` file. This file is stored in the `tools` subdirectory within your project's hidden `.sfdx` directory:

```
YourProject/.sfdx/tools/apex.log
```

## Find Missing Lightning Framework Features in Force.com IDE 2

---

If you don't see the Force.com IDE 2 features for working with Lightning Components, enable Lightning support for your project using the Project Explorer's contextual menu.

- In the Project Explorer, right-click your project and select **Configure > Enable Lightning Support**.

## Non-Executing Commands in Force.com IDE 2

---

If the actions you initiate in Force.com IDE 2 don't complete, check whether the IDE is looking for the Salesforce CLI binary in the correct location.

1. Select **Force.com IDE > Preferences > Salesforce DX**.
2. Check whether the path to `sfdx` is a valid location.
3. If the path isn't valid, select **Other**, click **Browse**, and select the location where your Salesforce CLI binary is stored.

## Unable to Work After Failed Org Authorization

---

Sometimes you try to authorize a Dev Hub or a scratch org using the Salesforce CLI or Force.com IDE 2, but you don't successfully log in to the org. The port remains open for the stray authorization process, and you can't use the CLI or IDE. To proceed, end the process manually.

## macOS or Linux

To recover from a failed org authorization on macOS or Linux, use a terminal to kill the process running on port 1717.

1. From a terminal, run:

```
lsof -i tcp:1717
```

2. In the results, find the ID for the process that's using the port.
3. Run:

```
kill -9 <the process ID>
```

## Windows

To recover from a failed org authorization on Windows, use the Task Manager to end the Node process.

1. Press Ctrl+Alt+Delete, then click **Task Manager**.
2. Select the **Process** tab.
3. Find the process named Node.



**Note:** If you're a Node.js developer, you might have several running processes with this name.

4. Select the process that you want to end, and then click **End Process**.



## CHAPTER 4 Open-Source Acknowledgments for Force.com IDE 2

Force.com IDE 2 relies on various open-source technologies.

### **Eclipse Oxygen**

Force.com IDE 2 is an Eclipse plug-in. The beta version uses Eclipse Oxygen.

### **EGit**

EGit is an Eclipse Team provider for the [Git version control system](#). It provides the Force.com IDE 2 Git integration tools.

### **Language Server Protocol**

The Apex Language Server Protocol follows the Language Server Protocol specification. Force.com IDE 2 uses this protocol to provide features such as autocompletion and error analysis.

### **Eclipse LSP4E**

The Apex Language Server Protocol is supported through the LSP4J and LSP4E projects from the Eclipse Foundation.


## CHAPTER 5 Resources for Force.com IDE 2 Users

Here are some resources to help you succeed with Force.com IDE 2.

- [Eclipse documentation](#)
- [Salesforce DX Setup Guide \(Beta\)](#)
- [Salesforce DX Developer Guide \(Beta\)](#)
- [Salesforce CLI Command Reference \(Beta\)](#)

## CHAPTER 6 Considerations for Force.com IDE 2 (Beta)

Here are some known issues you might run into while using Force.com IDE 2.

 **Note:** This release contains a beta version of Force.com IDE 2, which means it's a high-quality feature with known limitations. Force.com IDE 2 isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for Force.com IDE 2 in the [Salesforce DX Beta](#) group in the Success Community.

### General Known Issues in Force.com IDE 2

---

#### Creating Your First Project or Source File Can Fail or Take a Long Time

**Description:** If you use Force.com IDE 2 before installing the Salesforce DX CLI plug-ins, the IDE installs the plug-ins for you when you create a project or source file. This process can slow down source file creation or cause it to fail. If you try to create a project before installing the plug-ins, it fails and you get an error that a class couldn't be instantiated.

**Workaround:** Try again to create the source file or project. Because the IDE has installed the CLI plug-ins for you, the creation request succeeds.

#### Org Status Label for Project Doesn't Refresh Automatically

**Description:** When you authorize a Dev Hub, or create or authorize a scratch org, the label that shows the status of your connected org in the Project Explorer doesn't refresh.

**Workaround:** To refresh the label, click the name of your project in the Project Explorer.

### Known Issues in Apex Code Analysis

---

#### Completion Unavailable for Some Types

**Description:** The Force.com IDE 2 Apex code analysis is powered by a beta version of the Apex Language Server, which conforms to the [Language Server Protocol](#). This beta version offers code completion only for the following types:

- Standard Apex library
- User-defined types (classes, inner classes, interfaces, inner interfaces, enums, and inner enums) in your project

Code completion for standard objects, custom objects, and the code in managed packages isn't available.

**Workaround:** None

### Fixed Syntax Errors Sometimes Don't Disappear

**Description:** After you fix a syntax error, the corresponding error indication sometimes still shows up in Problems view and your source file.

**Workaround:** Restart Force.com IDE 2 by selecting **File > Restart**.

### Syntax Highlighting Sometimes Fails on Newly Opened Files

**Description:** When you open an Apex class or trigger, syntax highlighting sometimes doesn't work.

**Workaround:** Close and reopen the file.

### Numerous Errors Are Sometimes Logged

**Description:** When you're editing a file, errors are occasionally logged on each keystroke.

**Workaround:** Restart Force.com IDE 2 by selecting **File > Restart**. If the issue persists, in the Error Log view, disable **Activate on new events**. To access this option, click the Error Log view's arrow button.

## Known Issues in Apex Testing

---

### Running Test Classes with No Test Methods Shows a Confusing Error

**Description:** If your test run includes an Apex class that's annotated with `@isTest` but doesn't contain test methods, the test run fails. The message `ERROR running force:apex:test:run: Cannot read property 'message' of undefined.` is returned in the error log.

**Workaround:** If your test run fails with a generic error, make sure that all test classes in the run contain at least one method with the `@isTest` annotation or the `testMethod` keyword.

### Test Suites Must Be Hand-Coded or Retrieved via Metadata API

**Description:** You can't pull Apex test suites from a scratch org.

**Workaround:** Retrieve your test suites using `sfdx force:mdapi:retrieve` and then convert your project to the Salesforce DX format using `sfdx force:mdapi:convert`. Or, create your test suites by hand.

1. Create a `testSuites` directory. From the Project Explorer, right-click the directory within `tests` where you want to store your test suites and select **New > Folder**.
2. Right-click your `testSuites` directory and select **New > File**.
3. Give your test suite a name that ends in `.testSuite`. For example, `MySuite.testSuite`.
4. Customize your test suite using the XML format specified in [ApexTestSuite](#) in the *Metadata API Developer Guide*.
5. To push the test suite to your default scratch org, right-click your project in the Project Explorer and select **Salesforce DX > Push Source**.
6. Set up a test run configuration that includes the suite, as described in [Run Apex Tests in Force.com IDE 2](#).