Using Composer Dependency Manager

The Composer Dependency Manager is very helpful when you need to install PHP-related frameworks, for example PHPUnit.

You can use *Composer* from IntelliJ IDEA in three modes:

- Using the dedicated IntelliJ IDEA interface for most *basic* operations such as initializing a project with a `composer.json` stub created and adding dependencies.
- To have all *Composer* commands at disposal, you need to configure *Composer* as a command line tool and use it in the command line mode from the dedicated Command Line Tools Console Tool Window or pop-up window. For more details, see Running Command Line Tool Commands.
- You can have IntelliJ IDEA generate a project stub from an existing package using *Composer*. In this case, *Composer* will be initialized and one package will be installed during the project stub creation. After that you can use *Composer* from the command line or through the user interface.

On this page:

Preparing to run Composer in the command line mode

You can use the full range of *Composer* commands from IntelliJ IDEA only in the command line mode. To use *Composer* in this mode, you need to configure it as a command line tool. For more details, see Using Command Line Tools.

1. Download the Composer.phar definition file.
2. Make sure the *Command Line Tool Support* plugin is downloaded and enabled.

   The plugin is not bundled with IntelliJ IDEA, but it is available from the JetBrains plugin repository. Once enabled, the plugin is available at the IDE level, that is, you can use it in all your IntelliJ IDEA projects.
3. Open the project settings and click **Command Line Tool Support**.
4. On the Command Line Tool Support page, click the **Add** button. In the **Choose Tool to Add** dialog box that opens, choose **Composer**.
5. In the **Composer** dialog box that opens, specify the location of the `composer.phar` file in the **Path to composer.phar** text box. Type the path manually or click the **Browse** button ... and choose the desired location in the dialog box that opens. IntelliJ IDEA parses the contents of the specified file for Composer commands. When the file analyses is completed, IntelliJ IDEA displays the specified file in the list of command line tools available in IntelliJ IDEA.

   You can have several instances of Composer configured and switch between them from project to another by specifying the relevant one during Composer initialization.
6. In the **Alias** text box, specify the alias to use in calls of tool commands when running in the command line mode. Accept the default alias `c` or edit it, if necessary.
7. To activate the detected command set, select the **Enable** check box.
8. Customize the command set, if necessary.
9. In the **Show console in** area, specify where you want the **Input** pane for typing commands opened:
   - To have the **Input** pane opened in a pop-up window, choose the **Pop-up** option.
   - To have the **Input** pane opened as a text box at the bottom of the Command Line Tools Console tool window, choose the **Tool window** option.

For information about *running* the tool in the command line mode, see Running Command Line Tool Commands.

Using Composer via the dedicated user interface

IntelliJ IDEA provides a dedicated interface for such basic *Composer* operations as initializing a project and adding dependencies. In the UI mode, *Composer* is available at the project level. For information on using the tool from the command line, see Running Command Line Tool Commands.

Setting up Composer in an existing project

The dependencies added to a project through *Composer* are listed in the project-specific `composer.json` file. Therefore setting up the tool in a project starts with creating a project-specific `composer.json` file. This operation is also referred to as *Composer initialization*.

1. Open the project to use Composer in.
2. On the main menu, choose **Tools | Composer | Init Composer**.
3. In the Composer Settings dialog that opens, specify the location of the `composer.phar` file to use. Type the path manually or click the **Browse** button … and choose the desired location in the dialog box that opens.

   > If you have not downloaded the `composer.phar` file yet, click the **Download** link to have IntelliJ IDEA download it now and specify the folder to store the file in. You can afterwards configure the downloaded `composer.phar` as a command line tool and use it in the command line mode.

4. When you click **OK**, IntelliJ IDEA creates a stub of the `composer.json` file.
5. Open the `composer.json` file in the editor and complete the code or accept the default values. For more details, see Composer.json: Project Setup.

Adding Composer dependencies

When you add a dependency to the project using Composer, a new subfolder is created under the *Vendor* folder, the added package is stored in this subfolder, and the added package is added to the list in the `require` section of `Composer.json`. For information on adding dependencies manually in the command line mode, see Running Command Line Tool Commands. Below are instructions for adding dependencies using the interface provided by IntelliJ IDEA.

1. Open the project where Composer is set up.
2. On the main menu, choose **Tools | Composer | Add Dependency**.
3. In the Add Composer Dependency dialog that opens, select the package to add from the **Available Packages** list, possibly using the search field. The list shows all the available packages, however, the packages that have been already installed are marked with a tick. Choose the relevant version from the **Version to install** drop-down list, and then click **Install**.

When IntelliJ IDEA completes the installation, the package is stored in a new folder under the **Vendor** node and is added to the list in the `require` section of `Composer.json`.

Generating a project stub using Composer

1. On the main menu, choose **File | New Project**.

   > Alternatively, click the **New Project** button on the **Welcome screen**.

2. In the **Create New Project** dialog box that opens, specify the project name and location.
3. From the **Project type** drop-down list, choose **Composer Project**. The Composer Project Dialog opens.
4. Appoint the *Composer* instance to use.
   - To use commands from a previously downloaded `composer.phar`, choose **Use existing composer.phar:** and specify it location in the text box.
   - To have a new instance of *Composer* downloaded, choose **Download composer.phar from getcomposer.org:**. The `composer.phar` file will be saved under the project root folder specified in the Create New Project dialog.
5. In the **Package** area, specify the package to install during the project creation. Select the package to add from the **Available Packages** list, possibly using the search field, and choose the relevant version from the **Version to install** drop-down list.

When IntelliJ IDEA completes the project stub generation, the package is stored in a new folder under the **Vendor** node and is added to the list in the `require` section of `Composer.json`.

See Also

Procedures:

- Enabling a Command Line Tool
- Using Command Line Tools
- PHP-Specific Guidelines
- Enabling PHPUnit Support

Reference:

- Command Line Tool Support
- Command Line Tools Console Tool Window
- Composer Project Dialog
- Composer Settings Dialog
- Add Composer Dependency Dialog

Web Resources:

- Developer Community