Creating and Editing File Templates

IntelliJ IDEA supports several ways of creating file templates:

- Creating a file template from scratch
- Creating a file template from an existing one
- Creating a file template from a file
- Creating and referencing include templates

Creating a file template from scratch

1. Open the IDE Settings and select File and Code Templates page.
2. Switch to the **Templates** tab.
3. Click the **Add** button + on the toolbar and specify the template name, the file extension, and the body of the template, which can contain:
    1. Plain text.
    2. `#parse` directives to work with includes.
    3. Predefined variables to be expanded into corresponding values in the format `${<variable_name>}`.

       The available predefined file template variables are:

       - `${PACKAGE_NAME}` - the name of the target package where the new class or interface will be created.
       - `${PROJECT_NAME}` - the name of the current project.
       - `${FILE_NAME}` - the name of the PHP file that will be created.
       - `${NAME}` - the name of the new file which you specify in the **New File** dialog box during the file creation.
       - `${USER}` - the login name of the current user.
       - `${DATE}` - the current system date.
       - `${TIME}` - the current system time.
       - `${YEAR}` - the current year.
       - `${MONTH}` - the current month.
       - `${DAY}` - the current day of the month.
       - `${HOUR}` - the current hour.
       - `${MINUTE}` - the current minute.
       - `${PRODUCT_NAME}` - the name of the IDE in which the file will be created.
       - `${MONTH_NAME_SHORT}` - the first 3 letters of the month name. Example: Jan, Feb, etc.
       - `${MONTH_NAME_FULL}` - full name of a month. Example: January, February, etc.

       IntelliJ IDEA provides a set of additional variables for PHP include templates, that is, templates of reusable fragments that can be included in other *PHP file template*. The built-in *PHP include templates* are intended for generating file headers and PHPDoc documentation comments. The following variables are available in *PHP include templates*:

       - `${NAME}` - the name of the class, field, or function (method) for which the PHPDoc comment will be generated.
       - `${NAMESPACE}` - the fully qualified name (without a leading slash) of the class or field namespace.
       - `${CLASS_NAME}` - the name of the class where the field to generate the PHPDoc comment for is defined.

- `${STATIC}` - gets the value `static` if the function (method) or field to generate the comment for is `static`. Otherwise evaluates to an `empty string`.
- `${TYPE_HINT}` - a prompt for the `return` value of the function (method) to generate the comment for. If the return type cannot be detected through the static analysis of the function (method), evaluates to `void`.
- `${PARAM_DOC}` - - a documentation comment for parameters. Evaluates to a set of lines `@param type name`. If the function to generate comments for does not contain any parameters, the variable evaluates to empty content.
- `${THROWS_DOC}` - a documentation comment for exceptions. Evaluates to a set of lines `@throws type`. If the function to generate comments for does not throw any exceptions, the variable evaluates to empty content.
- `${DS}` - a dollar character (`$`). The variable evaluates to a plain dollar character (`$`) and is used when you need to escape this symbol so it is not treated as a prefix of a variable.
- `${CARET}` - indicated the position of the caret after generating and adding the comment.

  > This `${CARET}` variable is applied only when a PHPDoc comment is generated and inserted during file creation. When a PHPDoc comment is created through **Code | Generate | PHPDoc block**, multiple selection of functions or methods is available so documentation comments can be created to several classes, functions, methods, or fields. As a result, IntelliJ IDEA cannot "choose" the block to apply the `${CARET}` variable in, therefore in this case the `${CARET}` variable is ignored.

4. Custom variables. Their names can be defined right in the template through the `#set` directive or will be defined during the file creation.

4. To have the dollar character (`$`) in a variable rendered "as is", use the `${DS}` variable instead. This variable evaluates to a plain dollar character (`$`).
5. Apply the changes and close the dialog box.

Creating a file template from an existing one

1. Open the File Templates settings page and switch to the **Templates** tab.
2. Click **Copy** on the toolbar and change the template name, extension, and source code as required.
3. Apply the changes and close the dialog box.

Creating a file template from a file

1. Open the desired file in the editor.
2. On the main menu, choose **Tools | Save File as Template**.
3. In the File and Code Templates dialog box that opens specify the new template name and edit the source code, if necessary.
4. Apply the changes and close the dialog box.

Creating and referencing include templates

Include templates are used to define reusable pieces of code to be inserted in file templates through `#parse` directives.

1. In the File and Code Templates settings page, switch to the **Includes** tab.
2. Click **Add** + on the toolbar and specify the template name, extension, and the source code. Do one of the following:
   - Use the predefined file template variables.
   - Create custom template variables and define their values right in the include template using the `#set` VTL directive.

     For example, if you want to your full name inserted in the file header instead of your login name defined through the `${USER}`, write the following construct:

     ```
     #set( $MyName = "John Smith" )
     ```

     If, when applying a template, the values of certain template variable are not known, IntelliJ IDEA will ask you to specify them.

   You can prevent treating dollar characters (`$`) in template variables as prefixes. If you need a dollar character (`$` inserted as is, use the `${DS}` file template variable instead. When the template is applied, this variable evaluates to a plain dollar character (`$`).
3. To use the include template, switch to the **Templates** tab, select the desired template and click **Edit**.
4. To include a template, insert the #parse directive in the source code.

See Also

Concepts:

- File and Code Templates

Procedures:

- Creating Template-based Files

Reference:

- File and Code Templates

Web Resources:

- Developer Community