## Advanced

File | Settings | Project Settings | Language Injection - Advanced

Use this tab to specify different names for the base-annotations to be used. This helps avoid any dependencies on foreign code where it is not desired or possible. The custom annotations should provide the same properties as the original ones, i.e. `value` for all of them and an optional (default = "") `prefix` and `suffix` for the `@Language` replacement.

Also configure the runtime checks to be generated for the `@Pattern` validation.

| Item | Description |
|---|---|
| Annotation Classes | In this area, specify the classes that implement annotations of the following types:<br>- **Language annotations**<br>- **Pattern annotations**<br>- **Substitution annotations**<br>Type the class names, possibly using code completion. If necessary, use the **Browse** button … to open the **Select Class** dialog box, where you can you can locate the desired class in the project tree view. Alternatively, switch to the **Search by Name** tab and start typing the class name. As you type, the list of available classes narrows down to match your entry. |
| Runtime Pattern Validation | In this area, configure the runtime checks to be generated for the `@Pattern` validation. The available options are:<br>- **No runtime instrumentation** - if this option is selected, no checks will be inserted and any compiled class files will be affected.<br>- **Instrument with assertions** - if this option is selected, pattern-validation is controlled with the `-ea` JVM switch and throws `AssertionError`.<br>Selecting this option is recommended due to the potentially negative impact on the performance for methods that are invoked very often.<br>- **Instrument with IllegalArgumentException** - select this option to have the same result as when the `@NotNull` instrumentation of IntelliJ IDEA is used. |

| Item | Description |
|---|---|
| Performance | Click one of the radio buttons in this area to select the level of analysis and performance of the language injections resolution procedure.<br>- **Do not analyze (fast) -** if this option is selected, IntelliJ IDEA does not analyze injections.<br>Selecting the **Do not analyze** option significantly improves performance.<br>- **Analyze references -** if this option is selected, IntelliJ IDEA attempts to recognize injections introduced through variables.<br>- **Look for variable assignments -** if this option is selected, IntelliJ IDEA does not perform the dataflow analysis to detect the substitution strings, but looks for variable assignments only.<br>- **Use dataflow analysis (slow) -** if this option box is selected, IntelliJ IDEA applies dataflow analysis to language injections. |
| Include undefined operands to text in concatenation | If this check box is selected, IntelliJ IDEA inserts an injected operand as a literal, if its type is not recognized. |

### See Also

Concepts:

- Using Language Injections

Reference:

- Language Injections

External Links:

- IntelliLang
- IntelliLang Configuration

Web Resources:

- Developer Community