Editor's Override Indicators

(PDT)

Writer	Date	Comment	Approved
Shalom Gibly	11/5/2008	1. Dev2QA	
		2. Target Version – PDT 1.1	

TOC

1. Introduction 2
1.1 Requirement Rationale 2
1.2 New Terminology 2
2. Detailed Description 2
2.1 The Override Indicators UI 2
2.2 Tool-tips 3
2.3 Preferences Addition 4
3. Overriding rule 4

1. Introduction

1.1 Requirement Rationale

The 'Override Indicators' markers is a feature that displays special decoration icons to indicate methods that override or implement other methods from a super-types class / interface.

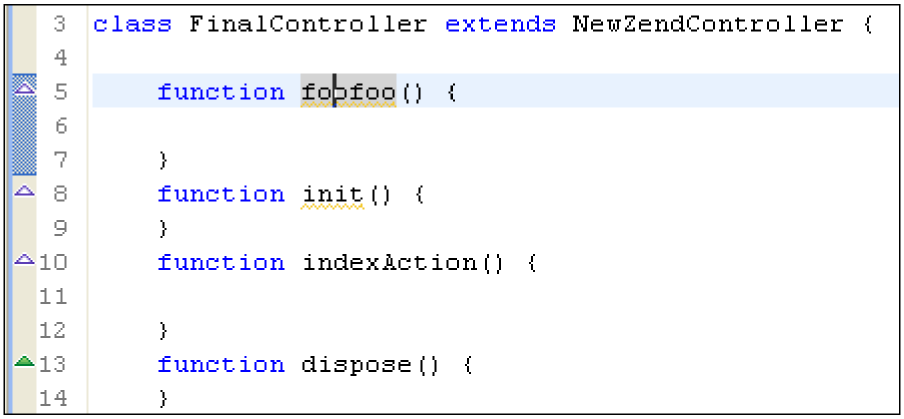
1.2 New Terminology

- *Overrides* – A class-method that re-define a *non-abstract* method that was defined in one of its super-classes.
  In PHP, a method 'overrides' another method if it has a matching name.
  Notes:
  - The names are case *insensitive*.
  - We do not take into consideration the method's arguments or return type.
- *Implements* – A class-method that provides a concrete implementation for a super-type abstract method that was defined in a class or an interface.
  Or, an interface method that was predefined in a super-interface.

2. Detailed Description

2.1 The Override Indicators UI

The override indicators are visible in the editor's 'Vertical Ruler' (editor's left side).
In case that a method overrides or implements another method, a small triangle appears on the ruler next to the method's definition.



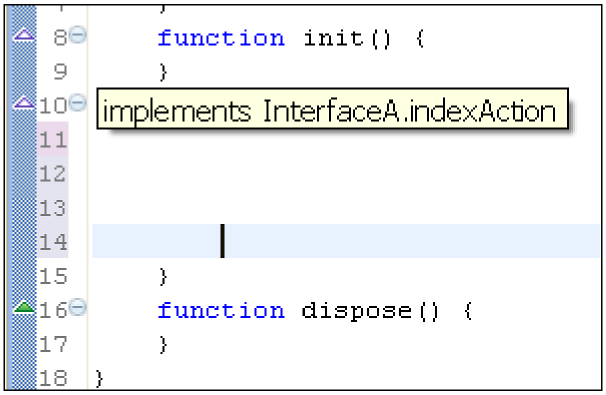
```
class FinalController extends NewZendController {

    function fofoo() {

    }
    function init() {
    }
    function indexAction() {

    }
    function dispose() {
    }
```

Override indicators on the 'vertical ruler'

Green triangle marks an overridden method, while a **white** triangle marks an implemented method.

2.2 Tool-tips

Hovering over a triangle displays a tool-tip that reveals the overridden or the implemented method super-type.



```
    function init() {
    }
implements InterfaceA.indexAction




    }
    function dispose() {
    }
}
```

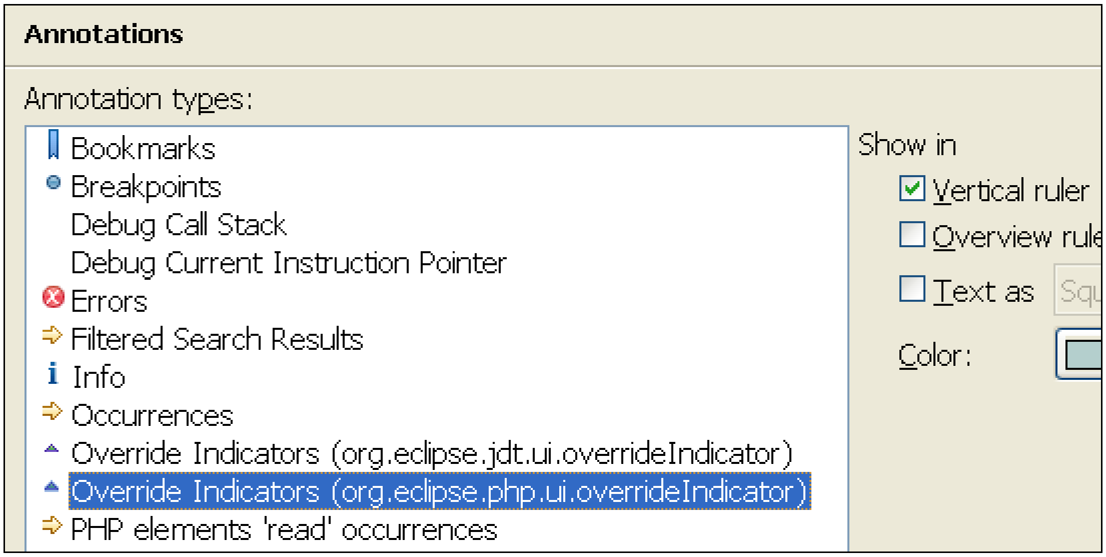
Override marker tool-tip

The tool-tip presents this syntax:

"implements / overrides SuperType.SuperTypeMethod"

2.3 Preferences Addition

Since the override indicators are basically ‘annotations’, the way that they are displayed is controlled through the ‘Annotations’ preferences page.



Annotations

Annotation types:

- Bookmarks
- Breakpoints
- Debug Call Stack
- Debug Current Instruction Pointer
- Errors
- Filtered Search Results
- Info
- Occurrences
- Override Indicators (org.eclipse.jdt.ui.overrideIndicator)
- Override Indicators (org.eclipse.php.ui.overrideIndicator)
- PHP elements 'read' occurrences

Show in

☑ Vertical ruler

□ Overview rule

□ Text as Squ

Color:

Override indicators in the Annotations preferences

3. Overriding rule

It’s pretty much obvious that markers will not appear on methods that does not override or implement any other method. But note that our implementation also does not display the markers when they are not complying with the overriding rules.

PHP overriding rules are as follows:

- You cannot override a non-static method with a static method.
- The overriding method must not be more restrictive than the overridden method:

Overridden Method	Overriding must be
`public`	`public`
`protected`	`public`, `protected`

(In PHP, a method without a visibility modifier is considered as ‘public’)

- ‘`private`’ methods are not inherited and cannot be overridden.

Shalom Gibly