

What do we expect from an editor

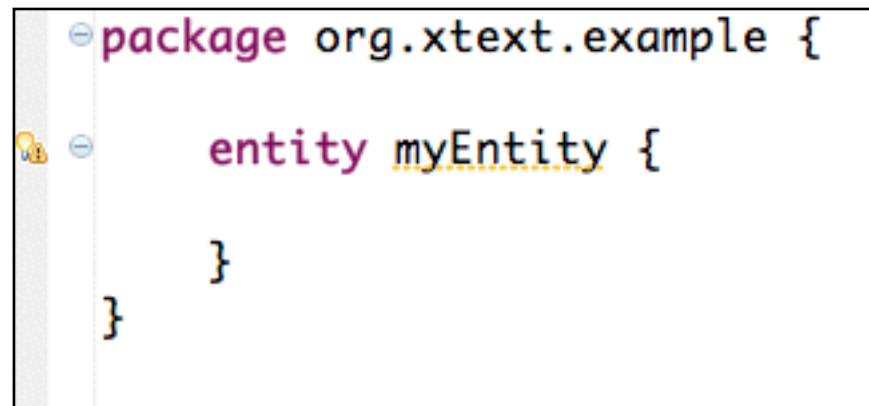
Syntaxhighlighting

```
package org.xtext.example {  
    entity MyEntity {  
    }  
}
```

What do we expect from an editor

Syntaxhighlighting

```
package org.xtext.example {  
  
    entity MyEntity {  
  
    }  
  
}
```



A screenshot of an IDE window showing the same Xtext code as the previous block. The code is: `package org.xtext.example {

 entity myEntity {

 }

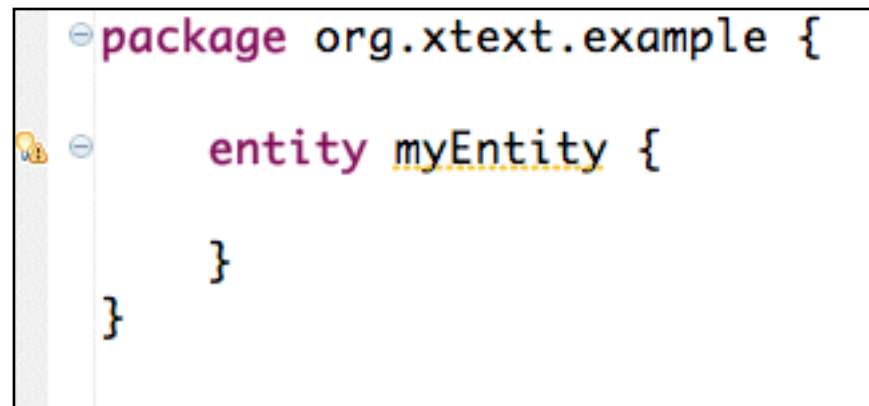
}`. The word `myEntity` is underlined with a yellow dashed line. A yellow warning icon (a lightbulb with an exclamation mark) is visible in the left margin next to the line containing `entity myEntity {`. The IDE has a light gray background and a vertical scrollbar on the right.

Validation / Quickfix / Annotations

What do we expect from an editor

Syntaxhighlighting

```
package org.xtext.example {  
  
    entity MyEntity {  
  
    }  
  
}
```

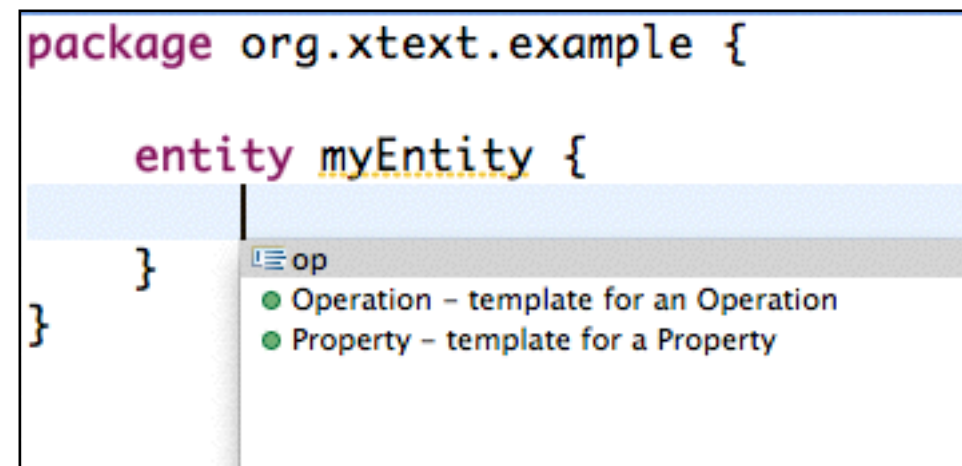


A screenshot of an IDE window showing the same Xtext code as the previous block. The code is syntax-highlighted. In the left margin (gutter), there are two icons: a blue minus sign and a yellow lightbulb icon, indicating features like folding and code suggestions.

```
package org.xtext.example {  
  
    entity myEntity {  
  
    }  
  
}
```

Validation / Quickfix / Annotations

Contentassist



A screenshot of an IDE window showing the same Xtext code. The text 'myEntity' is selected and highlighted with a yellow dashed border. A content assist popup menu is visible, showing two suggestions: 'op' (Operation) and 'Property' (Property), each preceded by a green dot. The popup also includes a small icon of a list and the text 'op'.

```
package org.xtext.example {  
  
    entity myEntity {  
  
    }  
  
}
```

- op
- Operation – template for an Operation
- Property – template for a Property

and much more ...

What's the challenge?

Create an editor on a Composite

What's the challenge?

Create an editor on a Composite

Operate on a real or synthetic resource

What's the challenge?

Create an editor on a Composite

Operate on a real or synthetic resource

Edit only a part of the model

What's the challenge?

Create an editor on a Composite

Operate on a real or synthetic resource

Edit only a part of the model

Get the text for a specific object

What's the challenge?

Create an editor on a Composite

Operate on a real or synthetic resource


Edit only a part of the model

Get the text for a specific object

Synchronize back the changes

What's there in Xtext?

EmbeddedEditorFactory

 `org.eclipse.xtext.ui.editor.embedded.EmbeddedEditorFactory`

Factory to create embedded editors for arbitrary [resources](#). A typical usage pattern looks like this:

```
EmbeddedEditorFactory editorFactory;  
  
EmbeddedEditor editor = editorFactory  
    .newEditor(resourceProvider)  
    .showErrorAndWarningAnnotations()  
    .withParent(parentComposite);  
EmbeddedEditorModelAccess model = editor.createPartialEditor();  
// work with the model
```

```
EmbeddedEditor editor = editorFactory  
    .newEditor(resourceProvider)  
    .showErrorAndWarningAnnotations()  
    .withParent(parentComposite);  
EmbeddedEditorModelAccess model = editor.createPartialEditor();
```

Builder API

What about the resource?

```
public Builder newEditor(IEditedResourceProvider resourceProvider)
```

What about the resource?

```
public Builder newEditor(IEditedResourceProvider resourceProvider)
```

```
    public interface IEditedResourceProvider {  
        XtextResource createResource();  
    }
```

What about the resource?

```
public Builder newEditor(IEditedResourceProvider resourceProvider)
```

```
    public interface IEditedResourceProvider {  
        XtextResource createResource();  
    }
```

Resource could be empty or pre populated

What about the resource?

```
public Builder newEditor(IEditedResourceProvider resourceProvider)
```

```
    public interface IEditedResourceProvider {  
        XtextResource createResource();  
    }
```

Resource could be empty or pre populated

ResourceSet must be fully configured

What about the resource?

```
@Inject  
org.eclipse.xtext.ui.resource.IResourceSetProvider rsProvider
```

```
new IEditedResourceProvider() {
```

```
@Override
```

```
public XtextResource createResource() {
```

ResourceSet

```
XtextResourceSet resourceSet = (XtextResourceSet) resource.getResourceSet();  
final String SYNTHETIC_SCHEME = "synthetic";  
ResourceSet rs = rsProvider.get(((IJavaProject) resourceSet.getClasspathURIContext()).getProject());
```

```
URI uri = URI.createURI(SYNTHETIC_SCHEME + "://embeddedResource"+ "fileextension");  
XtextResource resource = (XtextResource) rs.createResource(uri);
```

```
return resource;
```

```
}
```

```
}
```

What about the resource?

```
@Inject
org.eclipse.xtext.ui.resource.IResourceSetProvider rsProvider

new IEditedResourceProvider() {

    @Override
    public XtextResource createResource() {

        XtextResourceSet resourceSet = (XtextResourceSet) resource.getResourceSet();
        final String SYNTHETIC_SCHEME = "synthetic";
        ResourceSet rs = rsProvider.get(((IJavaProject) resourceSet.getClasspathURIContext()).getProject());

        URI uri = URI.createURI(SYNTHETIC_SCHEME + "://embeddedResource"+ "fileextension");
        XtextResource resource = (XtextResource) rs.createResource(uri);

        return resource;
    }
}
```

Resource

What about the resource?

```
@Inject
org.eclipse.xtext.ui.resource.IResourceSetProvider rsProvider

new IEditedResourceProvider() {

    @Override
    public XtextResource createResource() {

        XtextResourceSet resourceSet = (XtextResourceSet) resource.getResourceSet();
        final String SYNTHETIC_SCHEME = "synthetic";
        ResourceSet rs = rsProvider.get(((IJavaProject) resourceSet.getClasspathURIContext()).getProject());

        URI uri = URI.createURI(SYNTHETIC_SCHEME + ":/embeddedResource"+ "fileextension");
        XtextResource resource = (XtextResource) rs.createResource(uri);

        return resource;
    }
}
```

Resource

ResourceFactory identified by the fileextension

```
org.eclipse.xtext.resource.FileExtensionProvider

public String getPrimaryFileExtension()
```

Builder API

```
public Builder showErrorAndWarningAnnotations()
```

Show warnings and errors

Builder API

```
public Builder showErrorAndWarningAnnotations()
```

Show warnings and errors

```
public Builder showAnnotations(String... annotationTypes)
```

Builder API

```
public Builder showErrorAndWarningAnnotations()
```

Show warnings and errors

```
public Builder showAnnotations(String... annotationTypes)
```

```
public Builder processIssuesBy(IValidationIssueProcessor issueProcessor)
```

Create annotations (optional)

Builder API

```
public Builder showErrorAndWarningAnnotations()
```

Show warnings and errors

```
public Builder showAnnotations(String... annotationTypes)
```

```
public Builder processIssuesBy(IValidationIssueProcessor issueProcessor)
```

Create annotations (optional)

```
public Builder withResourceValidator(IResourceValidator resourceValidator)
```

Custom validator (optional)

Builder API

```
public Builder showErrorAndWarningAnnotations()
```

Show warnings and errors

```
public Builder showAnnotations(String... annotationTypes)
```

```
public Builder processIssuesBy(IValidationIssueProcessor issueProcessor)
```

Create annotations (optional)

```
public Builder withResourceValidator(IResourceValidator resourceValidator)
```

Custom validator (optional)

```
public Builder readOnly()
```

Builder API

```
public Builder showErrorAndWarningAnnotations()
```

Show warnings and errors

```
public Builder showAnnotations(String... annotationTypes)
```

```
public Builder processIssuesBy(IValidationIssueProcessor issueProcessor)
```

Create annotations (optional)

```
public Builder withResourceValidator(IResourceValidator resourceValidator)
```

Custom validator (optional)

```
public Builder readOnly()
```

```
public EmbeddedEditor withParent(final Composite parent)
```

Edit only a part of the model

```
public EmbeddedEditorModelAccess createPartialEditor(  
    String prefix, String editablePart, String suffix,  
    boolean insertLineBreaks)
```

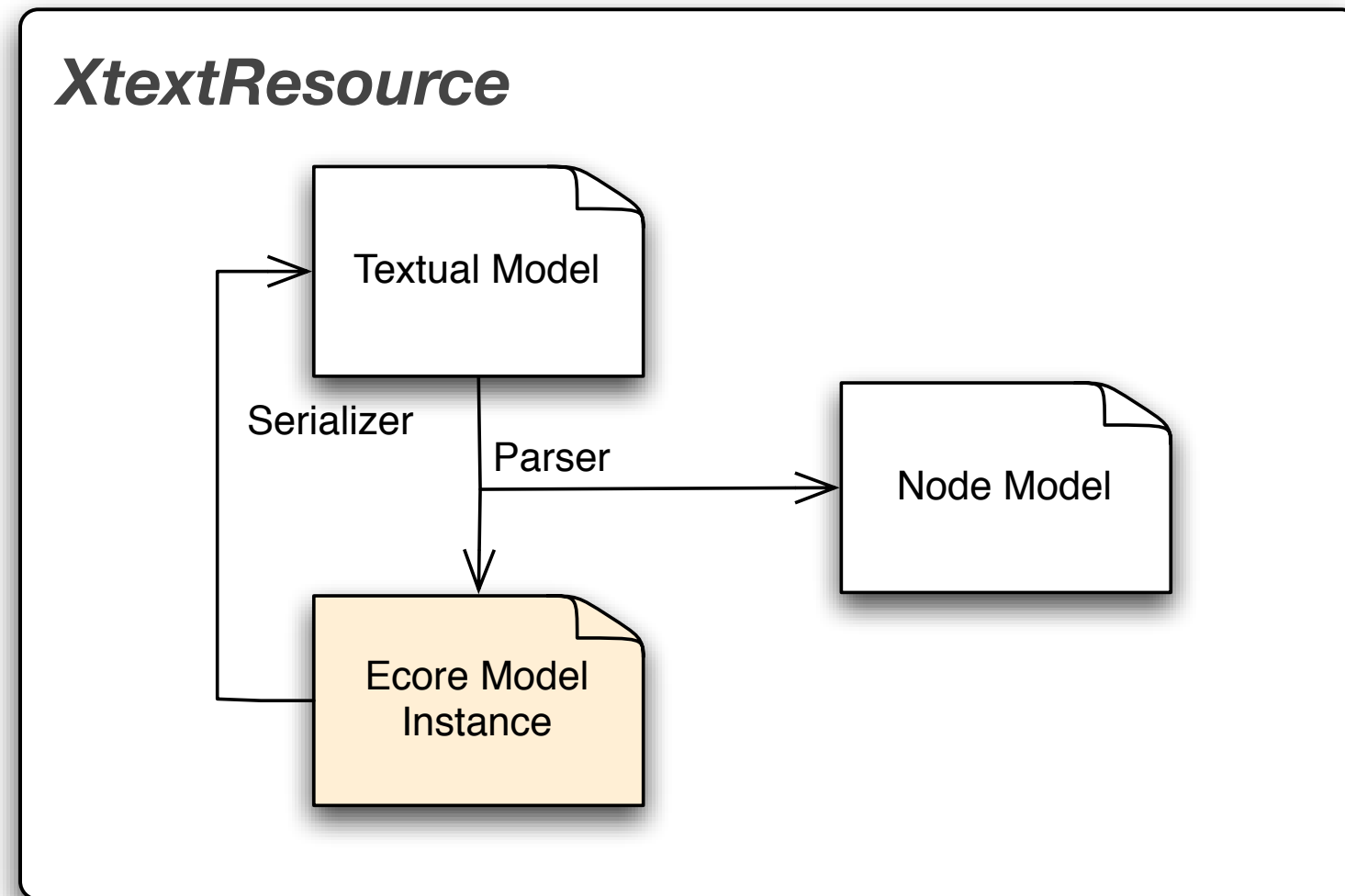
Prefix - non visible part

editablePart - visible part

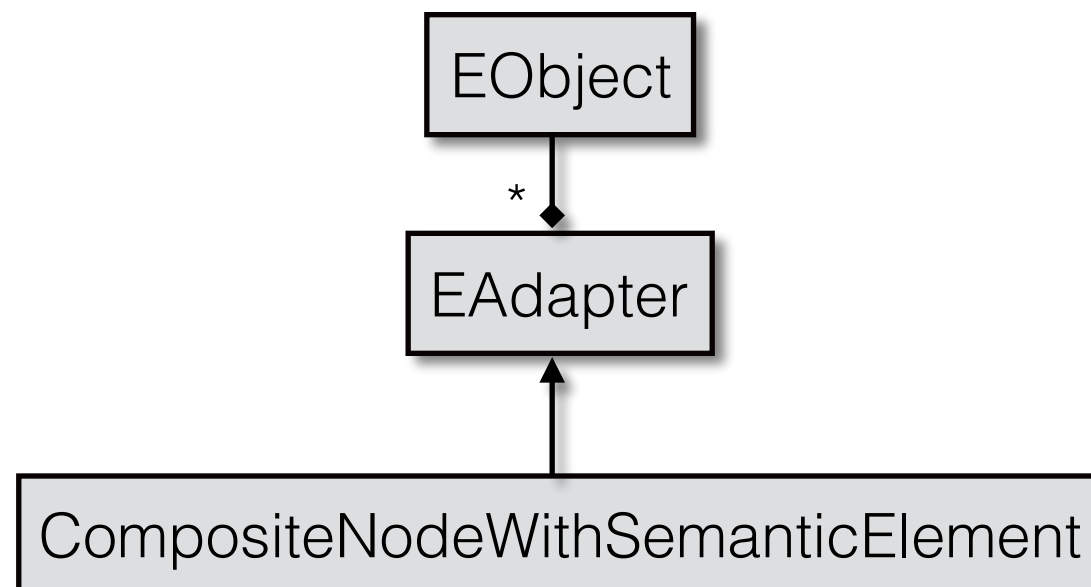
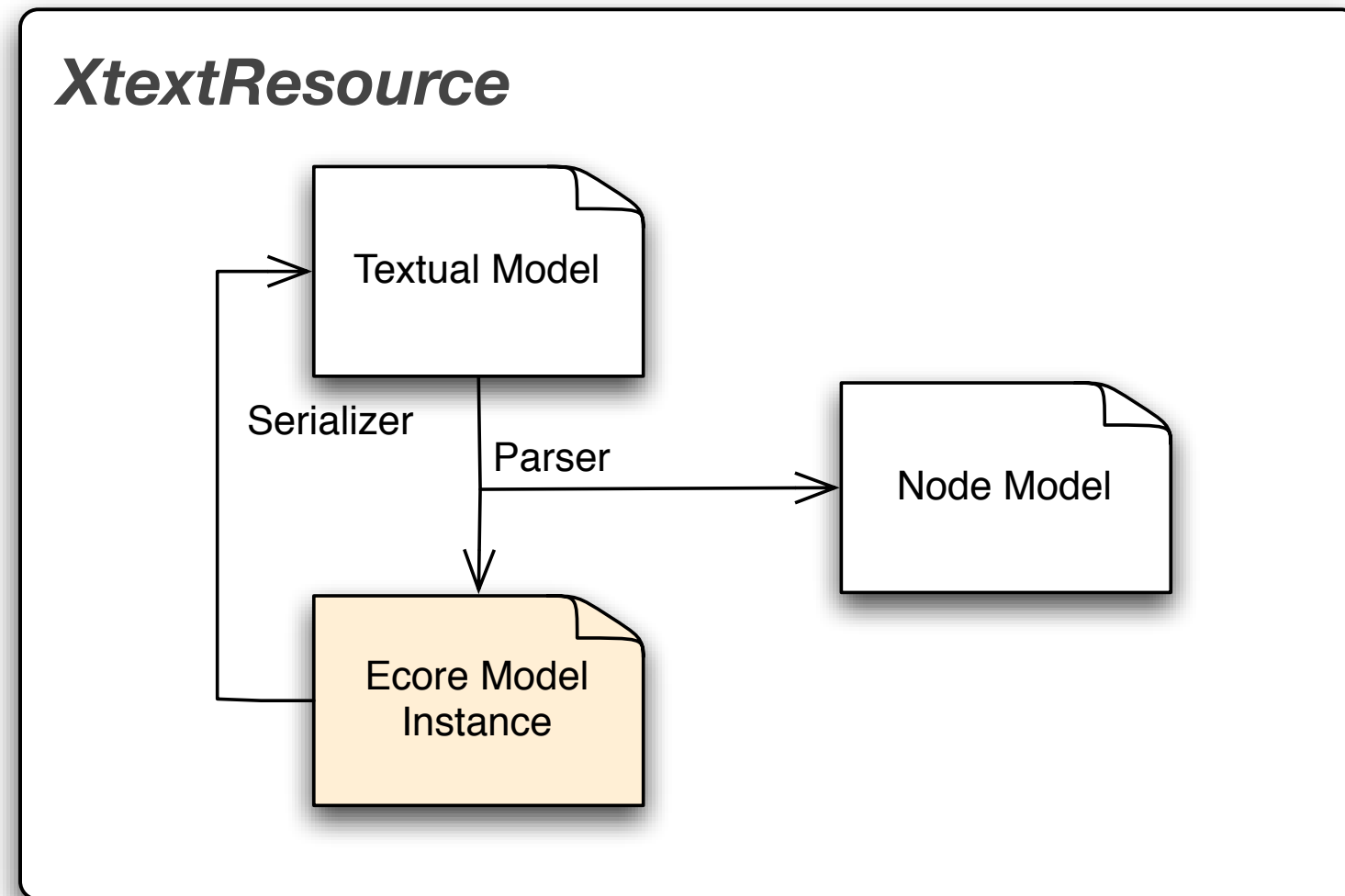
suffix - non visible part

insertLineBreaks - linebreaks between the parts

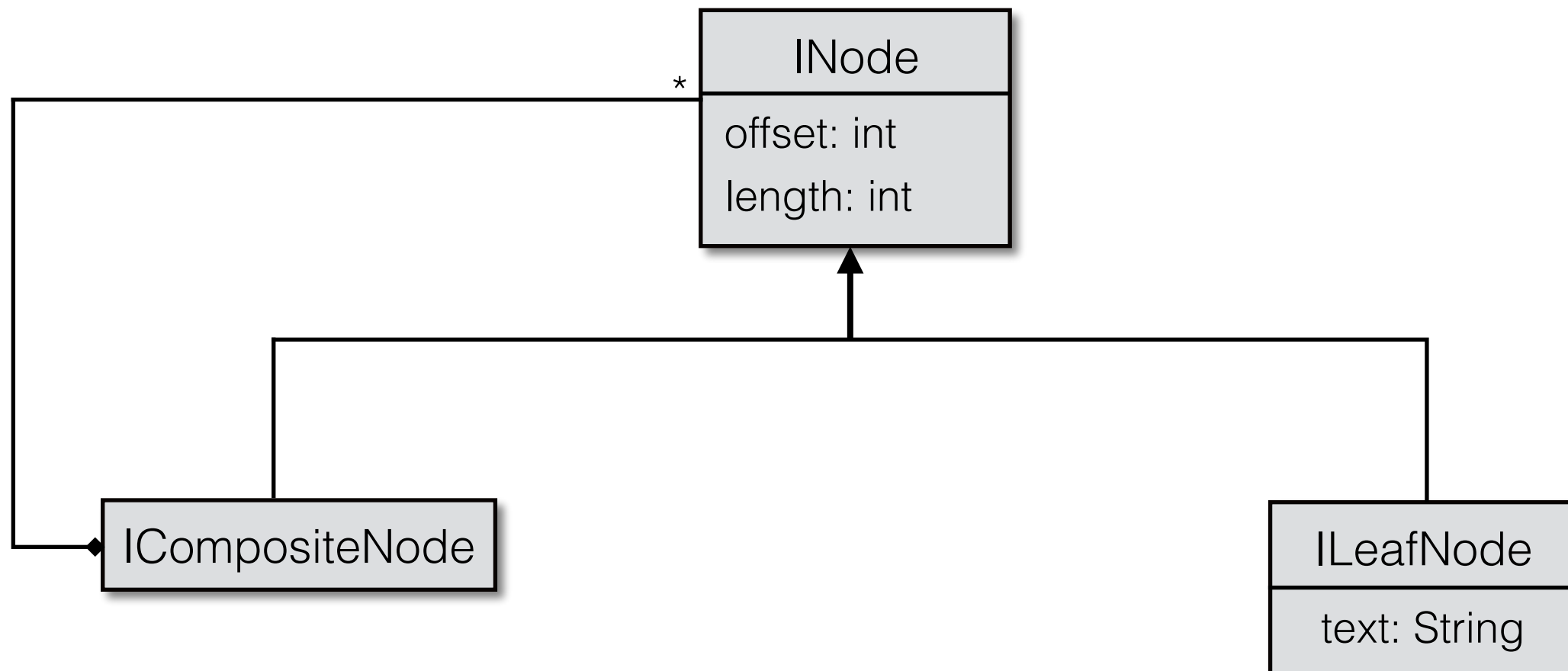
How to get the editablePart?



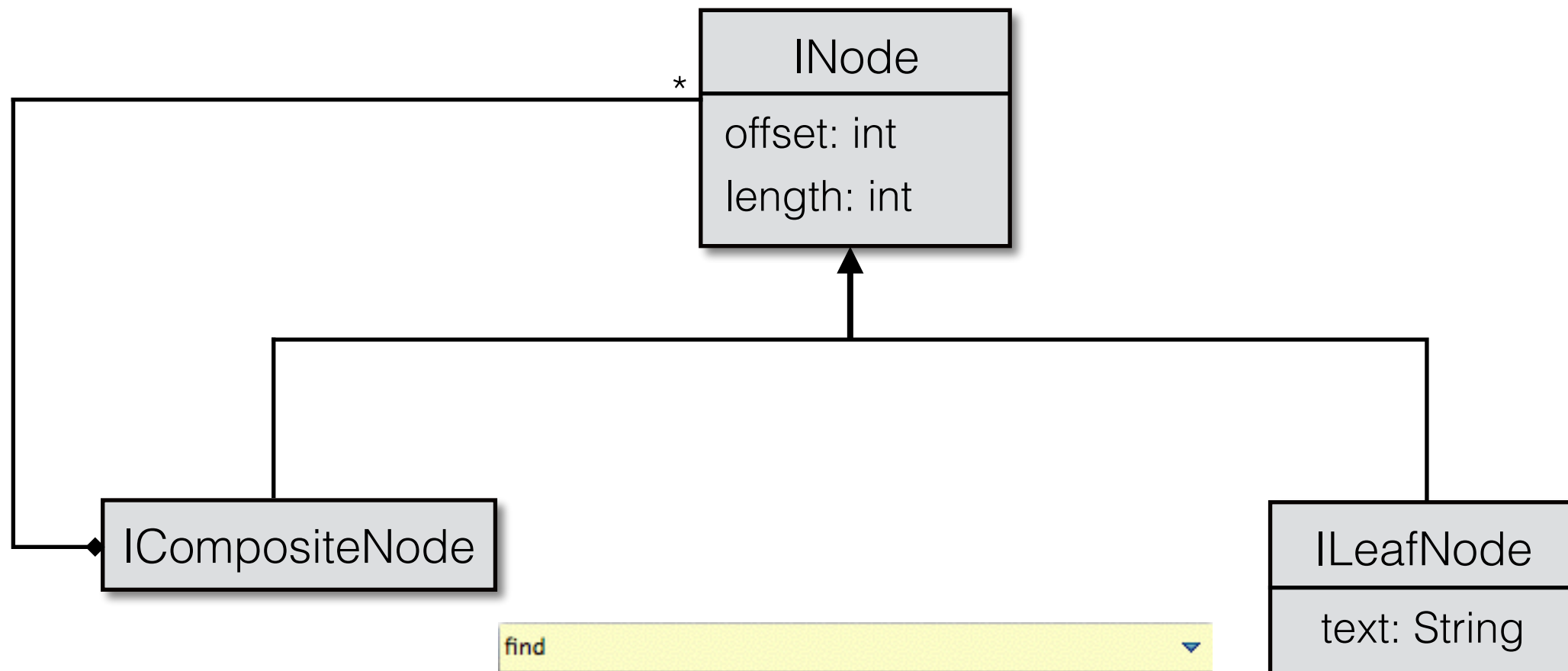
How to get the editablePart?



How to get the editablePart?



How to get the editablePart?



```
find
NodeModelUtils
  findLeafNodeAtOffset(INode, int) : ILeafNode
  findNodesForFeature(EObject, EStructuralFeature) : List
  findActualNodeFor(EObject) : ICompositeNode
  findActualSemanticObjectFor(INode) : EObject
```

Press '%O' to show inherited members

Synchronize back the changes

Depends on the use case

Synchronize back the changes

Depends on the use case

Simple String edited with tooling and written back to the model as a String

Synchronize back the changes

Depends on the use case

Simple String edited with tooling and written back to the model as a String

Complete model edited and editable part should be updated in resource

Synchronize back the changes

```
org.eclipse.xtext.resource.XtextResource.update(int, int, String)
```

```
public void update(int offset, int replacedTextLength, String newText)
```


Synchronize back the changes

```
org.eclipse.xtext.resource.XtextResource.update(int, int, String)
```

```
public void update(int offset, int replacedTextLength, String newText)
```

We already know offset and length from the Node

Update the embedded editor

```
public EmbeddedEditorModelAccess createPartialEditor(  
    String prefix, String editablePart, String suffix,  
    boolean insertLineBreaks)
```

org.eclipse.xtext.ui.editor.embedded.EmbeddedEditorModelAccess

```
public void updateModel(String prefix, String editablePart, String suffix)
```

DEMO

Thank you!

