CS 635 Advanced Object-Oriented Design & Programming
Spring Semester, 2010
Doc 24 Dynamic Factory, Extension Object, Value Object
4 May 2010

References

The Dynamic Factory Pattern, Welicki, Yoder, Wirfs-Brock, http://www.hillside.net/plop/2008/papers/ACMVersions/welicki.pdf

The Extension Objects Pattern, Erich Gamma, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.48.196&amp;rep=rep1&amp;type=pdf

Value Object, Riehle, http://www.hillside.net/plop/2006/Papers/ACM_Version/Value_Object.pdf

Contributing to Eclipse: Princiles, Patterns, and Plug-Ins, Gamma & Beck, Addison-Wesley, 2004

Eclipse on-line Documentation, http://help.eclipse.org/galileo/index.jsp

OpenDoc, Wikipedia, The Free Encyclopedia,http://en.wikipedia.org/w/index.php?title=OpenDoc&oldid=324657114, 4 May 2010 21:12 UTC

OpenDoc

MS Office

Word

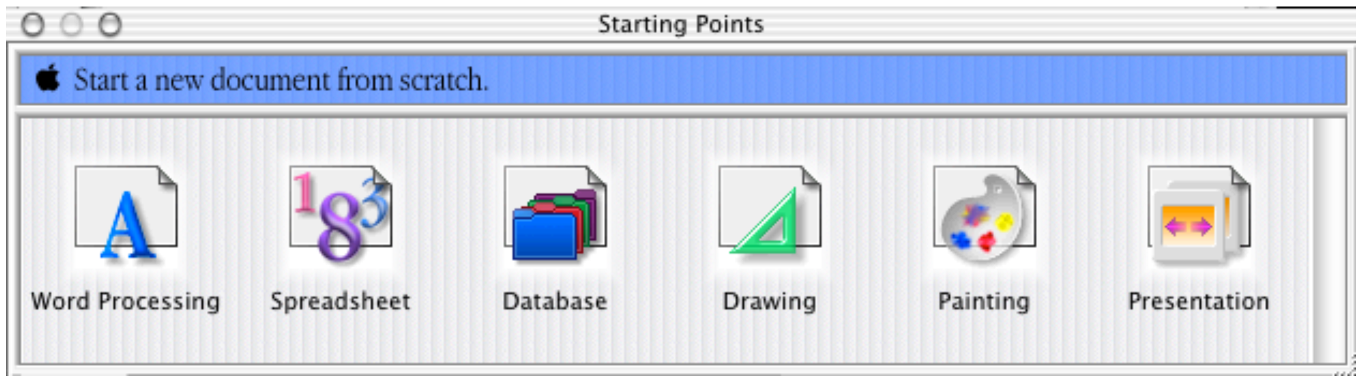
Excel

PowerPoint

Entourage

Embed documents

All-in-One Application



OpenDoc

1992-1997

Framework for components to work together

Small reusable components added by user to application

Components would create part of document

Main interface was the document

OpenDoc

How to structure the application

People can add components & functionality

How to Structure the Document

Composite pattern

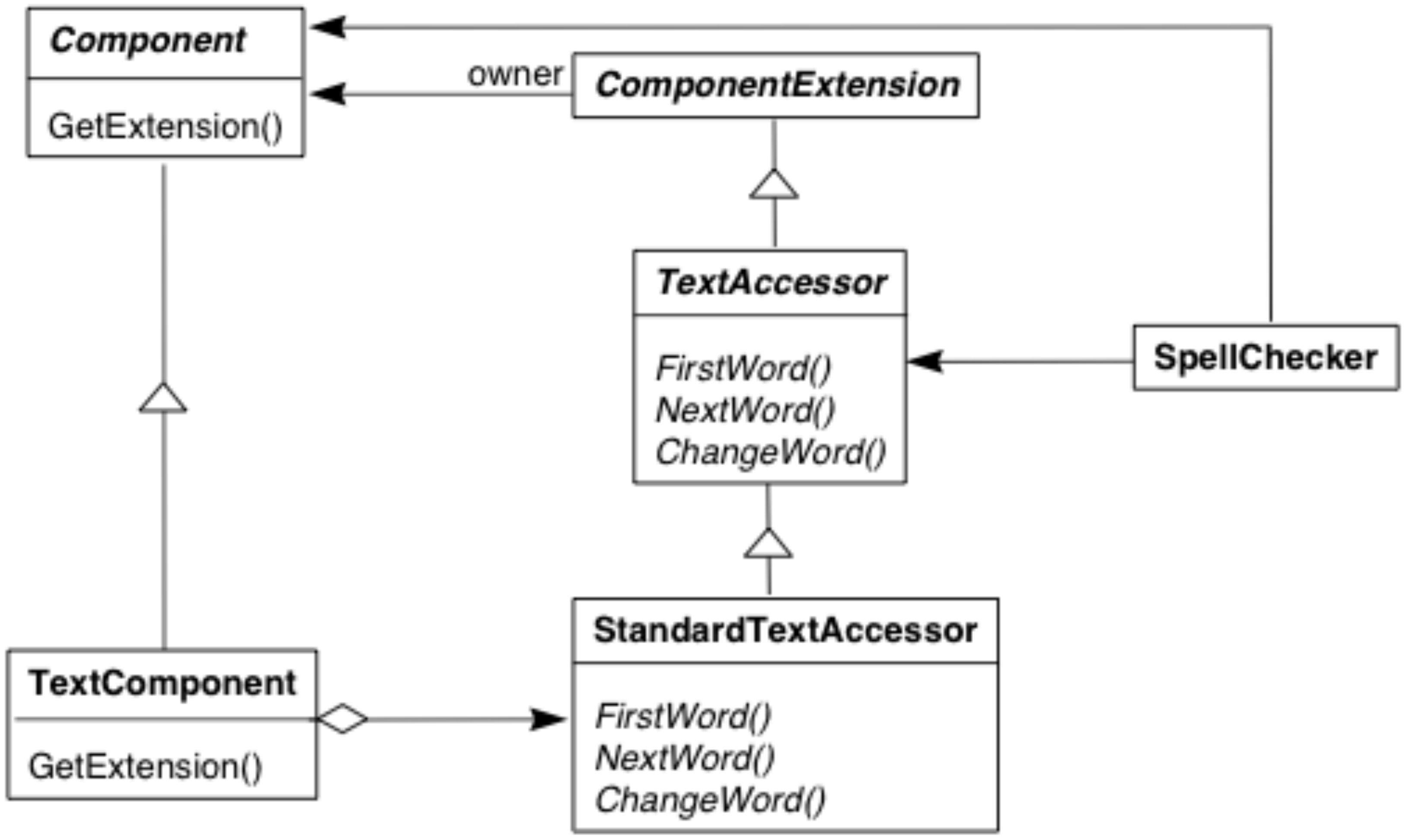
How do we add new functionality?

Add translation component?
Add spell checker

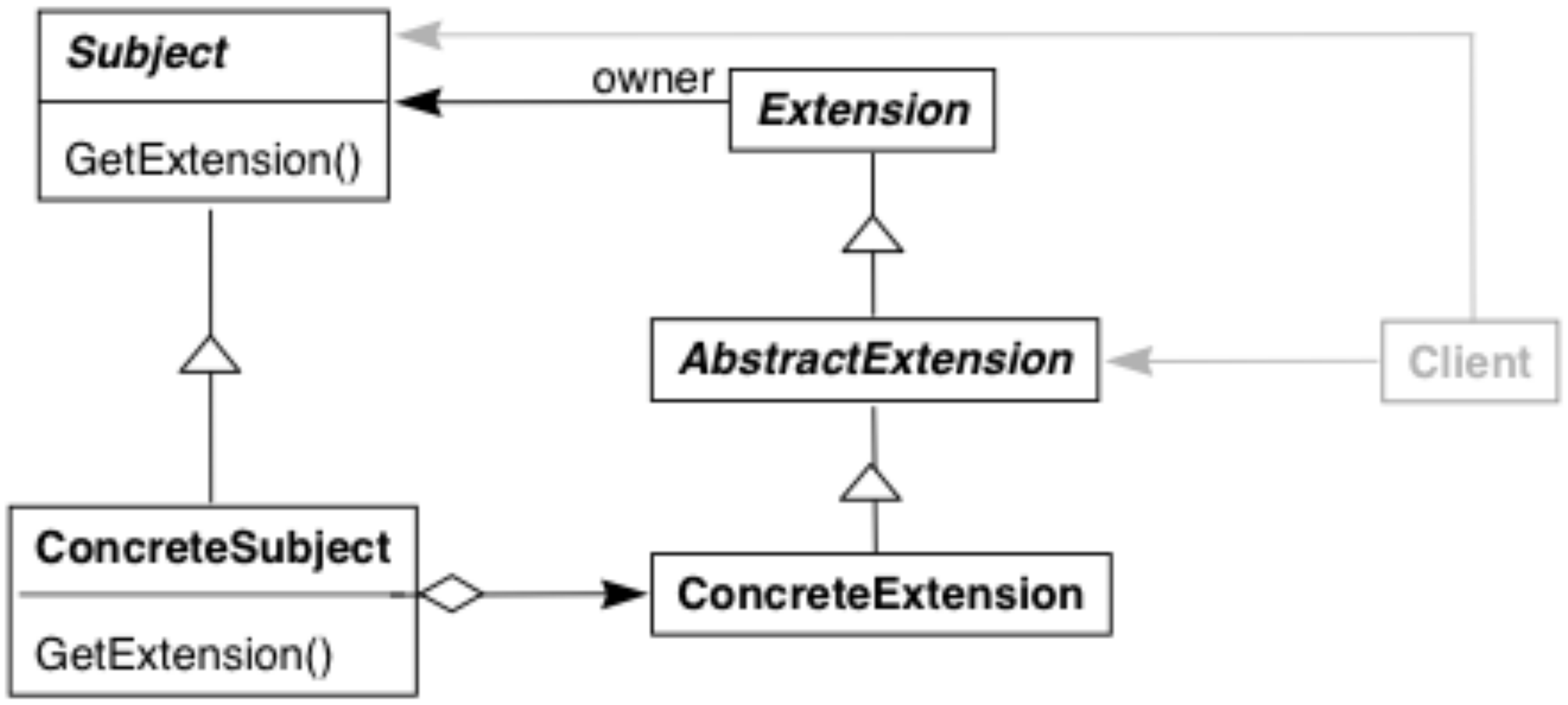
How does new component get required data from document?

We don't know what the component will need when designing application?

Extension Object



Structure


Subject
GetExtension()
owner
Extension
AbstractExtension
Client
ConcreteSubject
GetExtension()
ConcreteExtension

Applicability

a class should be extensible with new behavior without subclassing from it

Need to support new or unforeseen interfaces to existing classes
you don't want to impact clients that don't need this new interface

a class representing a key abstraction plays different roles for different clients. The number of roles the class can play should be open-ended. There is a need to preserve the key abstraction itself.

Some Problems

Clients become more complex

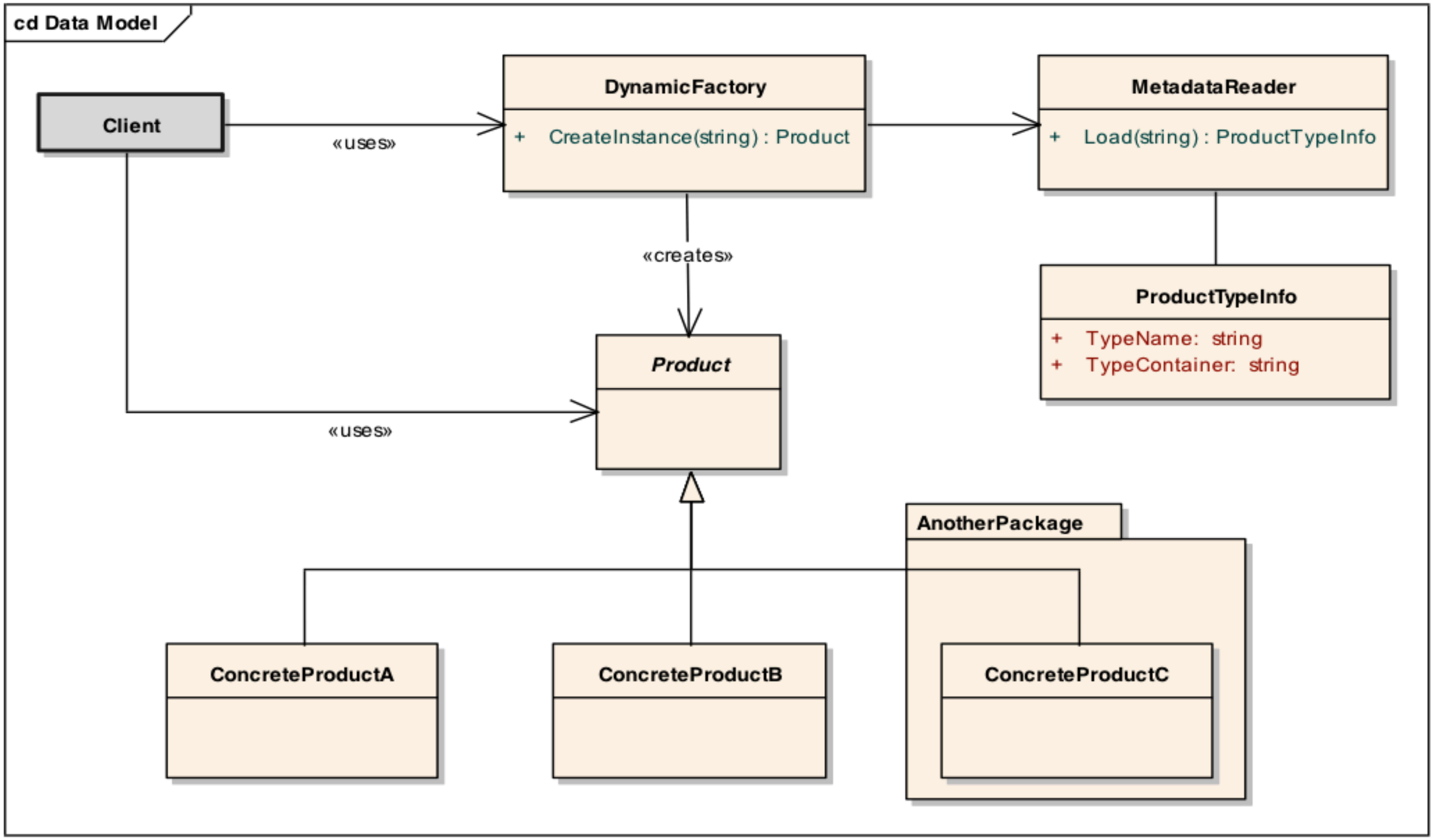
Tension to abuse extensions for concepts that should be explicitly modeled

Eclipse Plugin

How do we design Eclipse to allow plugins?

Eclipse has to run code it knows nothing about

Dynamic Factory



Eclipse Plugin

Plugin contains:

- Doc folder
- help folder
- icon folder
- lib folder
- MANIFEST.MF
- plugin.xml

com.teaminabox.eclipse.wiki Manifest

```
Manifest-Version: 1.0
Bundle-Name: Eclipse Wiki
Bundle-ClassPath: wiki.jar,lib/java2html_4.1.jar,lib/commons-lang-2.2.jar
Bundle-Activator: com.teaminabox.eclipse.wiki.WikiPlugin
Bundle-ManifestVersion: 2
Bundle-RequiredExecutionEnvironment: J2SE-1.5
Bundle-Vendor: Channing Walton
Bundle-SymbolicName: com.teaminabox.eclipse.wiki;singleton:=true
Require-Bundle: org.eclipse.ui,org.eclipse.jface.text,org.eclipse.core
 .resources,org.eclipse.ui.editors,org.eclipse.ui.ide,org.eclipse.ui.w
 orkbench.texteditor,org.eclipse.ui,org.eclipse.core.runtime.compatibi
 lity,org.eclipse.jdt.core,org.eclipse.ui.forms,org.eclipse.ui.views,o
 rg.eclipse.jdt.ui,org.eclipse.jdt.launching,org.junit4
Bundle-Version: 2.7.1.200906042321
Eclipse-LazyStart: true
```

Plugin.xml

Metadata for the plugin

Extension points

- Places were Eclipse can be extended

Extensions

- Code that extends Eclipse
- plugin may have many extensions

part of Plugin.xml

```
<extension
      point="org.eclipse.ui.editors">
    <editor
        name="Wiki Editor"
        default="true"
        icon="icons/wiki.png"
        extensions="wiki"
        contributorClass="com.teaminabox.eclipse.wiki.editors.WikiBrowserEditorContributor"
        class="com.teaminabox.eclipse.wiki.editors.WikiBrowserEditor"
        symbolicFontName="com.teaminabox.eclipse.wiki.fontDefinition"
        id="com.teaminabox.eclipse.wiki.editors.WikiEditor">
    </editor>
  </extension>
```

Indicates that when Eclipse edits a file ending in ".wiki" it should run the class com.teaminabox.eclipse.wiki.editors.WikiBrowserEditor to edit the file.

IEditorPart

Eclipse Editors must implement IEditorPart interface

IEditorInput getEditorInput()

  Returns the input for this editor

  If this value changes the part must fire a property listener event with PROP_INPUT

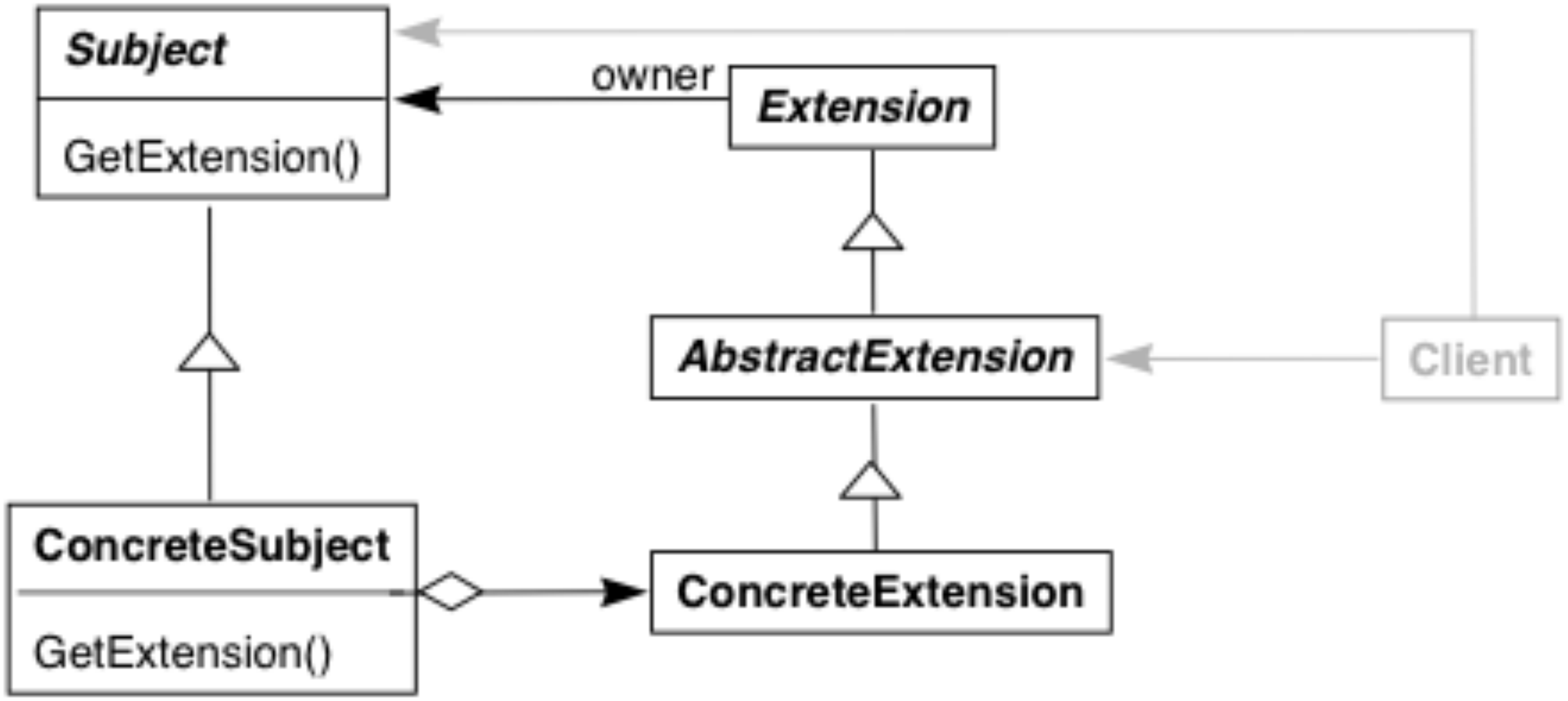
IEditorSite getEditorSite()

  Returns the site for this editor

void init(IEditorSite site, IEditorInput input)

  Initializes this editor with the given editor site and input.

Extension pattern Structure



Dynamic Factory Consequences

Benefits

Extensibility
Flexibility
Configurability
Agility

Liabilities

Run-time errors
Complexity
Over-engineering
Performance
Security
Debugging

Value Object

An Object that represents a value

- Money amounts
- protocol names
- percents
- fractions
- integrals
- meters, liters, etc

Value Object Pattern

Make value object immutable if

- Domain concept represents a value type
- Resulting class does not become so heavyweight it slows down performance

Benefits

Better domain modeling & understanding

Make value objects act like "normal numbers"

```
amount = amount + deposit;

amount.add(deposit);
```

Safer programs

Immutable object have not side-effects

Potentially better performance

Only copies when changed

More Benefits

Concurrency
- No need for serialization

Persistence
- No need for object id

Serialization
- No need to worry about references to object

Memory consumption & garbage collection
- Can use sharing (Flyweight)

Liabilities

More complex code

Changes in coding style

```
amount = amount + deposit;
  instead of
amount.add(deposit);
```

Potentially lower performance

Implementation

Identity, equality & hash codes

Sharing

- only makes sense on finite types
- Flyweight & factory require synchronization