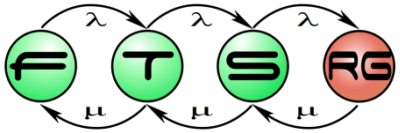
IDE Plug-in Development



Motto

- Any problem in computer science can be solved with another level of indirection.

David Wheeler

Table of Contents

- Connecting Views and Editors
  - Workbench Selection Service
  - Using Adapters (with Properties view)
- Resource Handling
  - File System
  - Project Management
- Background Jobs
- Case Study: JDT

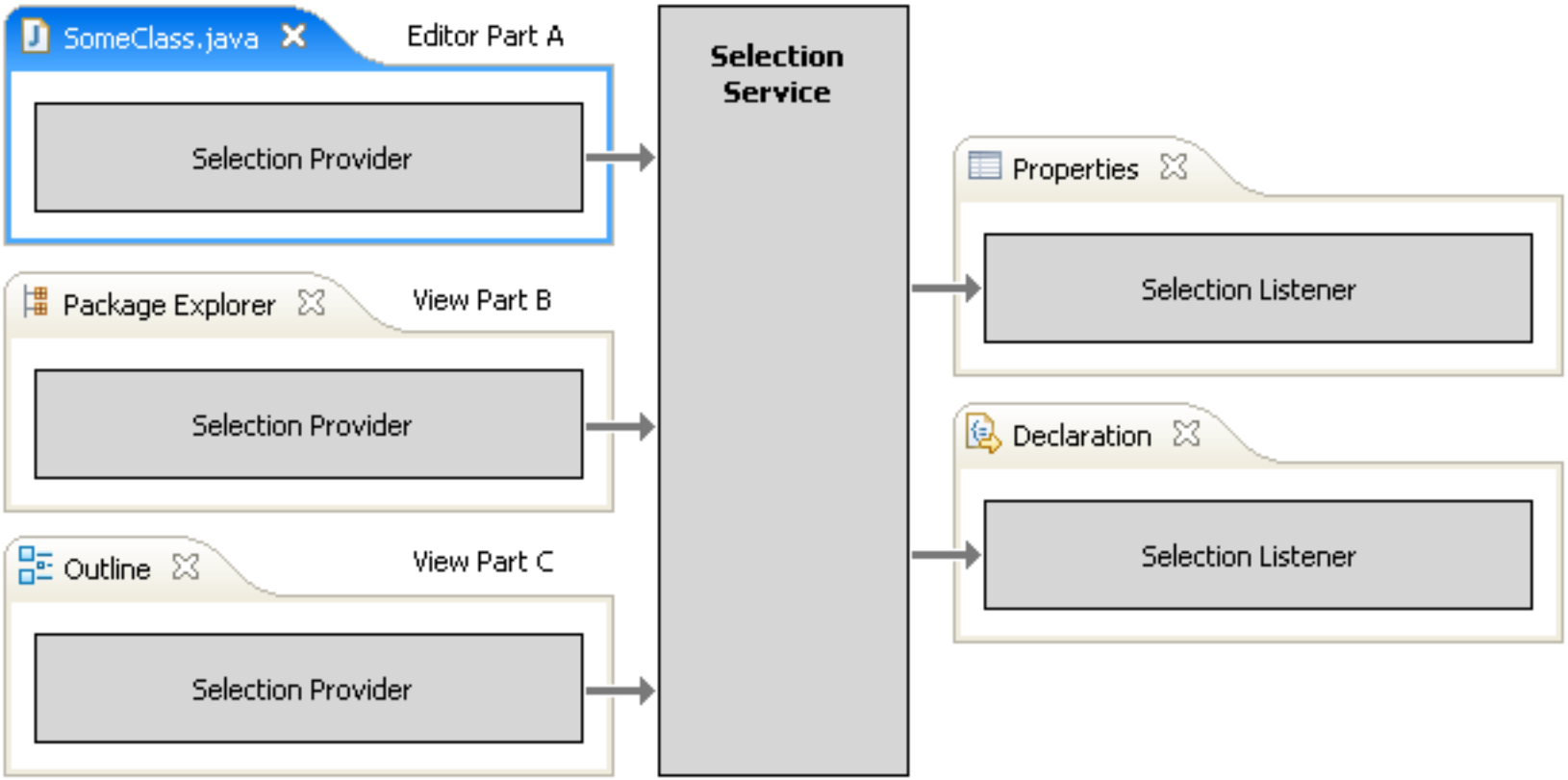
Workbench Selection Service

Handling Selection

- Common UI task: master-detail sections
- Inside a Part (editor/view)
  - SelectionChangedListener
  - Master-detail data binding
- Between Parts
  - Instance of other part is not known!
    - Most often not even its class
  - Usually part ID is known
    - Dependency!
    - Not tracked by PDE

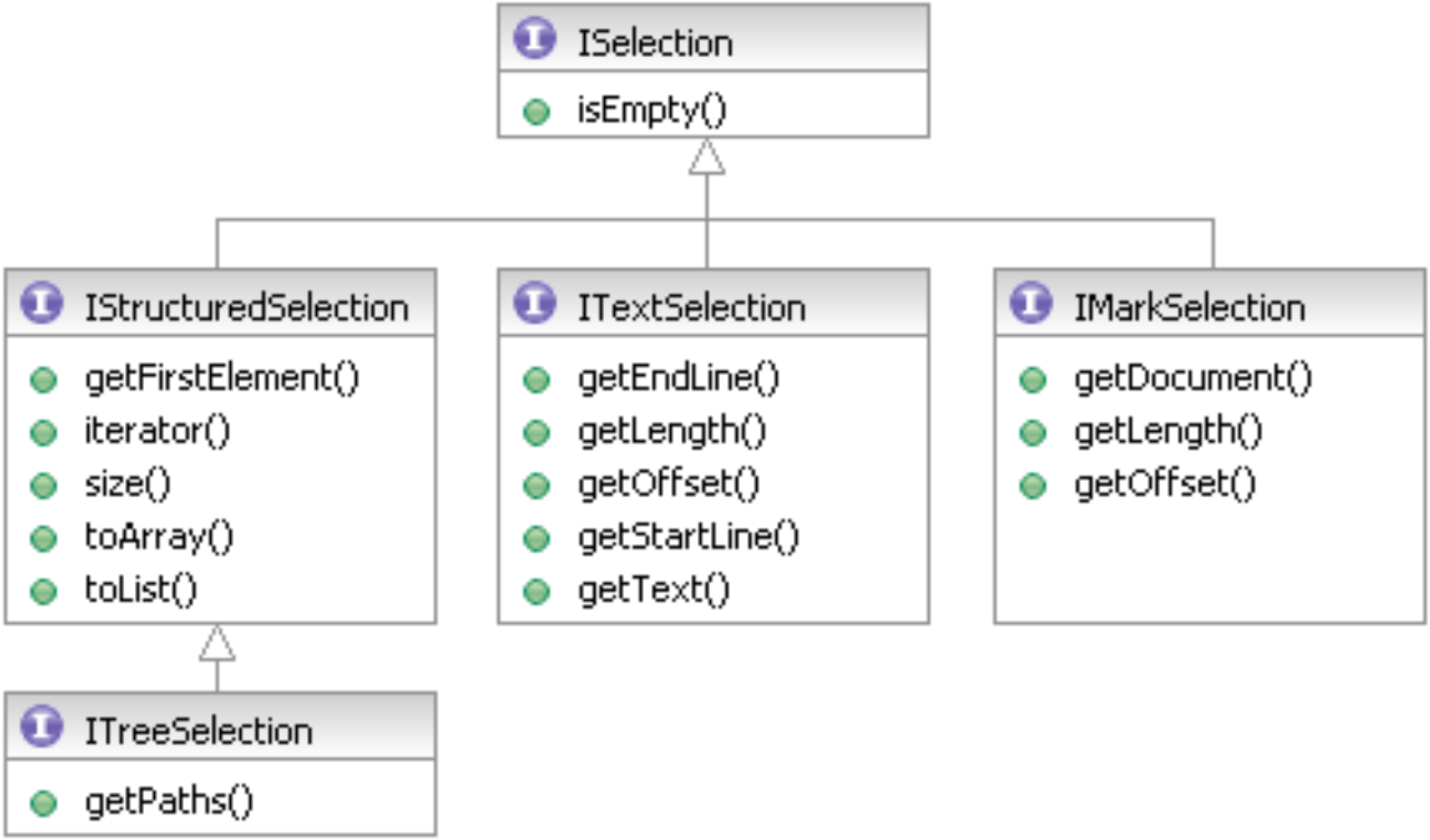
Handling Selection

- SelectionService - indirection



Selection

- ISelection hierarchy



ISelectionProvider interface

- Reports the change of selection
- Implementation
  - JFace viewers always implement this
  - Custom provider can be created
- Registration

```
getSite().setSelectionProvider(viewer);
```

JFace implementation

Viewer	Selection type
ComboViewer	IStructuredSelection
ListViewer	IStructuredSelection
TreeViewer	IStructuredSelection, ITreeSelection
+-- CheckboxTreeViewer	IStructuredSelection, ITreeSelection
TableViewer	IStructuredSelection
+-- CheckboxTableViewer	IStructuredSelection
TextViewer	ITextSelection, IMarkSelection
+-- SourceViewer	ITextSelection, IMarkSelection

Finding to Selection

- ISelectionService lookup
  - `getSite().getWorkbenchWindow().getSelectionService()`
- Getting actual selection
  - `ISelection getSelection()`
    - Return current selection of active part
  - `ISelection getSelection(String partId)`
    - Return current selection of selected part

Selection Changes

- ISelectionListener interface
- Single method:
  - `public void selectionChanged(IWorkbenchPart sourcepart, ISelection selection)`

Listener Registration

- Listening to Global Selection
  - `void addSelectionListener(ISelectionListener listener)`
  - `void removeSelectionListener(ISelectionListener listener)`
- Part-dependent Selection Listeners
  - `void addSelectionListener(String partId, ISelectionListener listener)`
  - `void removeSelectionListener(String partId, ISelectionListener listener)`

Cleanup

- Never forget to remove listeners
  - Garbage collection is not enough
  - Reuse dispose methods of parts

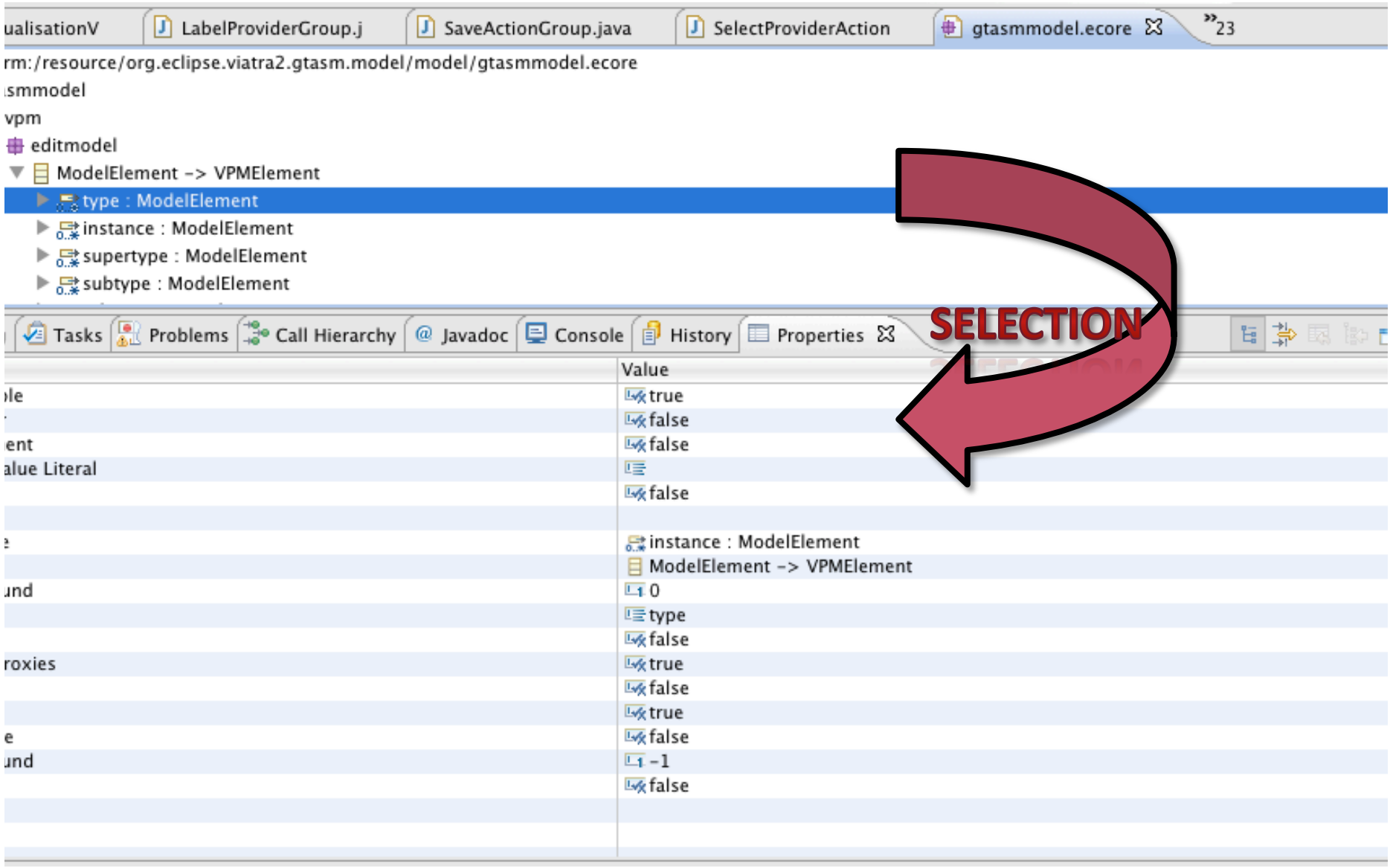
```
public void dispose() {
  ISelectionService s = getSite().
    getWorkbenchWindow().
    getSelectionService();
  s.removeSelectionListener(mylistener);
  super.dispose();
}
```

More Details

- Marc R. Hoffmann: Eclipse Workbench: Using the Selection Service
  - http://eclipse.org/articles/Article-WorkbenchSelections/article.html

Using Adapters

Example: Properties view



Example: Properties View

- Selection Handling
  - IPropertySource inferface needed (+ IPropertySource2)

```
public interface IPropertySource {
        public Object getEditableValue();
        public IPropertyDescriptor[] getPropertyDescriptors();
        public Object getPropertyValue(Object id);
        public boolean isPropertySet(Object id);
        public void resetPropertyValue(Object id);
        public void setPropertyValue(Object id, Object value);
}

public interface IPropertySource2 extends IPropertySource {
     boolean isPropertyResettable(Object id);
     public boolean isPropertySet(Object id);
}
```

Implementing IPropertySource

Implementing IPropertySource

- Direct implementation
  - `class FileProperties implements IPropertySource2`
- Problems
  - What about multiple interfaces for different views?
    - Unmaintainable
  - May require API changes
  - Beware dependencies!
    - IPropertySource is UI dependent
    - Selected object is often UI-independent

Implementing IPropertySource

- Direct implementation
  - `class FileProperties implements IPropertySource2`
- Problems
  - What about multiple interfaces for different views?
    - Unmaintainable
  - May require API changes
  - Beware dependencies!
    - IPropertySource is UI dependent
    - Selected object is often UI-independent

Implementing IPropertySource

- Direct implementation
  - `class FileProperties implements IPropertySource2`
- Problems
  - What about multiple interfaces for different views?
    - Unmaintainable
  - May require API changes
  - Beware dependencies!
    - IPropertySource is UI dependent
    - Selected object is often UI-independent

Implementing IPropertySource

- Direct implementation
  - `class FileProperties implements IPropertySource2`
- Problems
  - What about multiple interfaces for different views?
    - Unmaintainable
  - May require API changes
  - Beware dependencies!
    - IPropertySource is UI dependent
    - Selected object is often UI-independent

Implementing IPropertySource

- Direct implementation
  - `class FileProperties implements IPropertySource2`
- Problems
  - What about multiple interfaces for different views?
    - Unmaintainable
  - May require API changes
  - Beware dependencies!
    - IPropertySource is UI dependent
    - Selected object is often UI-independent

Implementing IPropertySource

- Direct implementation
  - `class FileProperties implements IPropertySource2`
- Problems
  - What about multiple interfaces for different views?
    - Unmaintainable
  - May require API changes
  - Beware dependencies!
    - IPropertySource is UI dependent
    - Selected object is often UI-independent

Unwanted UI-dependency!

Implementing IPropertySource

- Create Adapter class

```
public class FigureProperties implements IPropertySource2{

        IFigure figure;

        public FigureProperties(IFigure figure){this.figure = figure;}
        public IFigure toFigure() { return this.figure; }

        @Override
        public Object getEditableValue() {…}
        @Override
        public Object getPropertyValue(Object id) {…}
        @Override
        public boolean isPropertySet(Object id) {…}
        @Override
        public void resetPropertyValue(Object id) {…}
        @Override
        public void setPropertyValue(Object id, Object value) {…}


}
```

Implementing IPropertySource

- Create Adapter class

```
public class FigureProperties implements IPropertySource2{

        IFigure figure;

        public FigureProperties(IFigure figure){this.figure = figure;}
        public IFigure toFigure() { return this.figure; }

        @Override
        public Object getEditableValue() {…}
        @Override
        public Object getPropertyValue(Object id) {…}
        @Override
        public boolean isPropertySet(Object id) {…}
        @Override
        public void resetPropertyValue(Object id) {…}
        @Override
        public void setPropertyValue(Object id, Object value) {…}


}
```

Connecting to Model

Implementing IPropertySource

- Create Adapter class

```
public class FigureProperties implements IPropertySource2{

        IFigure figure;

        public FigureProperties(IFigure figure){this.figure = figure;}
        public IFigure toFigure() { return this.figure; }

        @Override
        public Object getEditableValue() {…}
        @Override
        public Object getPropertyValue(Object id) {…}
        @Override
        public boolean isPropertySet(Object id) {…}
        @Override
        public void resetPropertyValue(Object id) {…}
        @Override
        public void setPropertyValue(Object id, Object value) {…}


}
```

Connecting to Model

Interface Realization

Creating Adapters - Directly

```
public class AdaptableShape implements IFigure,
IAdaptable {

  @Override
  public Object getAdapter(Class adapter) {
    if (IPropertySource2.class.equals(adapter)) {
      return new FigureProperties(this);
    }
    return null;
  }
…
}
```

Creating Adapters - Directly

IAdaptable interface

```
public class Ad                mplements IFigure,
IAdaptable {

  @Override
  public Object getAdapter(Class adapter) {
    if (IPropertySource2.class.equals(adapter)) {
      return new FigureProperties(this);
    }
    return null;
  }
…
}
```

Creating Adapters - Directly

IAdaptable interface

Get Adapter by Class

```
public class Ad                mplements IFigure,
IAdaptable {

  @Override
  public Object getAdapter(Class a
    if (IPropertySource2.class.equals(adapter)) {
      return new FigureProperties(this);
    }
    return null;
  }
…
}
```

Creating Adapters - Directly

```
public class Ad                mplements IFigure,
IAdaptable {

  @Override
  public Object getAdapter(Class a
    if (IPropertySource2.class.equals(adapter)) {
      return new FigureProperties(this);
    }
    return null;
  }
…
}
```

IAdaptable interface

Get Adapter by Class

Not much better

Creating Adapters - Platform

```
public class AdaptableShape implements IFigure,
IAdaptable {

  @Override
  public Object getAdapter(Class adapter) {
    if (IPropertySource2.class.equals(adapter)) {
     return new FigureProperties(this);
    }
     return Platform.getAdapterManager().
      getAdapter(this, adapter);
  }
…
}
```

Creating Adapters - Platform

```
public class AdaptableShape implements IFigure,
IAdaptable {

  @Override
  public Object getAdapter(Class adapter) {
    if (IPropertySource2.class.equals(adapter)) {
     return new FigureProperties(this);
    }
    return Platform.getAdapterManager().
     getAdapter(this, adapter);
  }
…
}
```

Built-in adapters

Creating Adapters - Platform

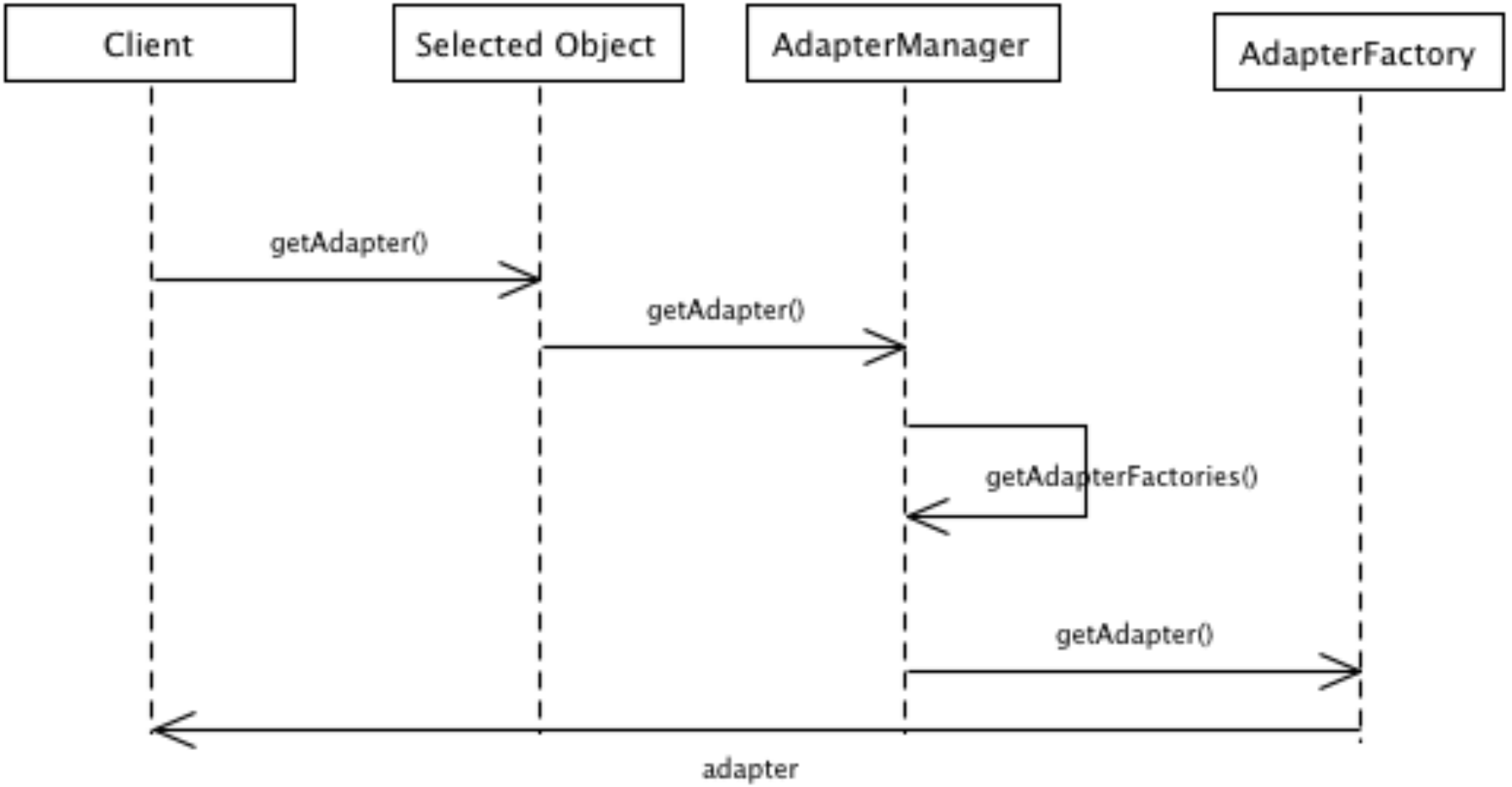
```
public class AdaptableShape implements IFigure,
IAdaptable {

  @Override
  public Object getAdapter(Class adapter) {
    if (IPropertySource2.class.equals(adapter)) {
     return new FigureProperties(this);
    }
    return Platform.getAdapterManager().
     getAdapter(this, adapter);
  }
…
}
```

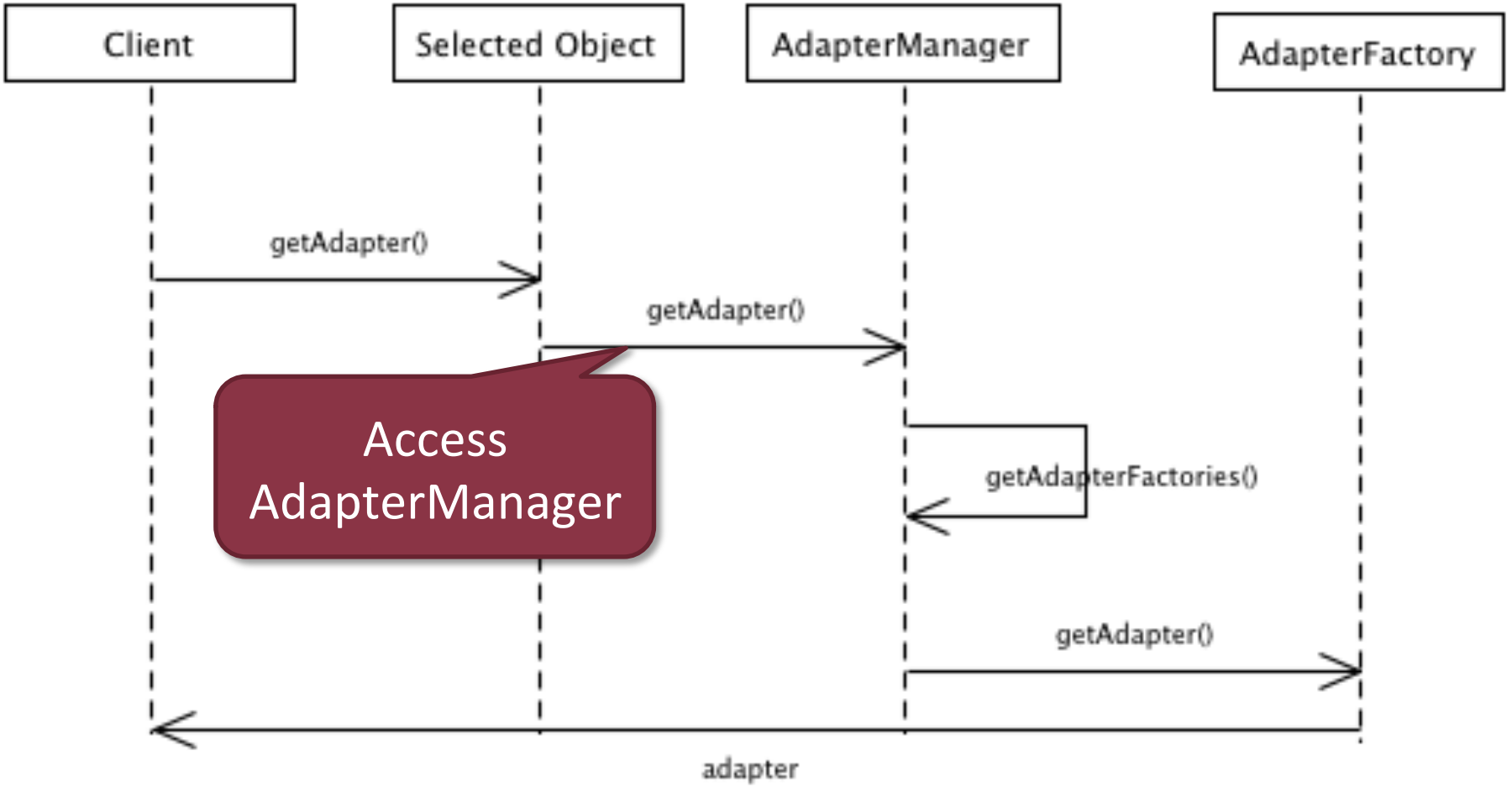
Built-in adapters

Registered adapters

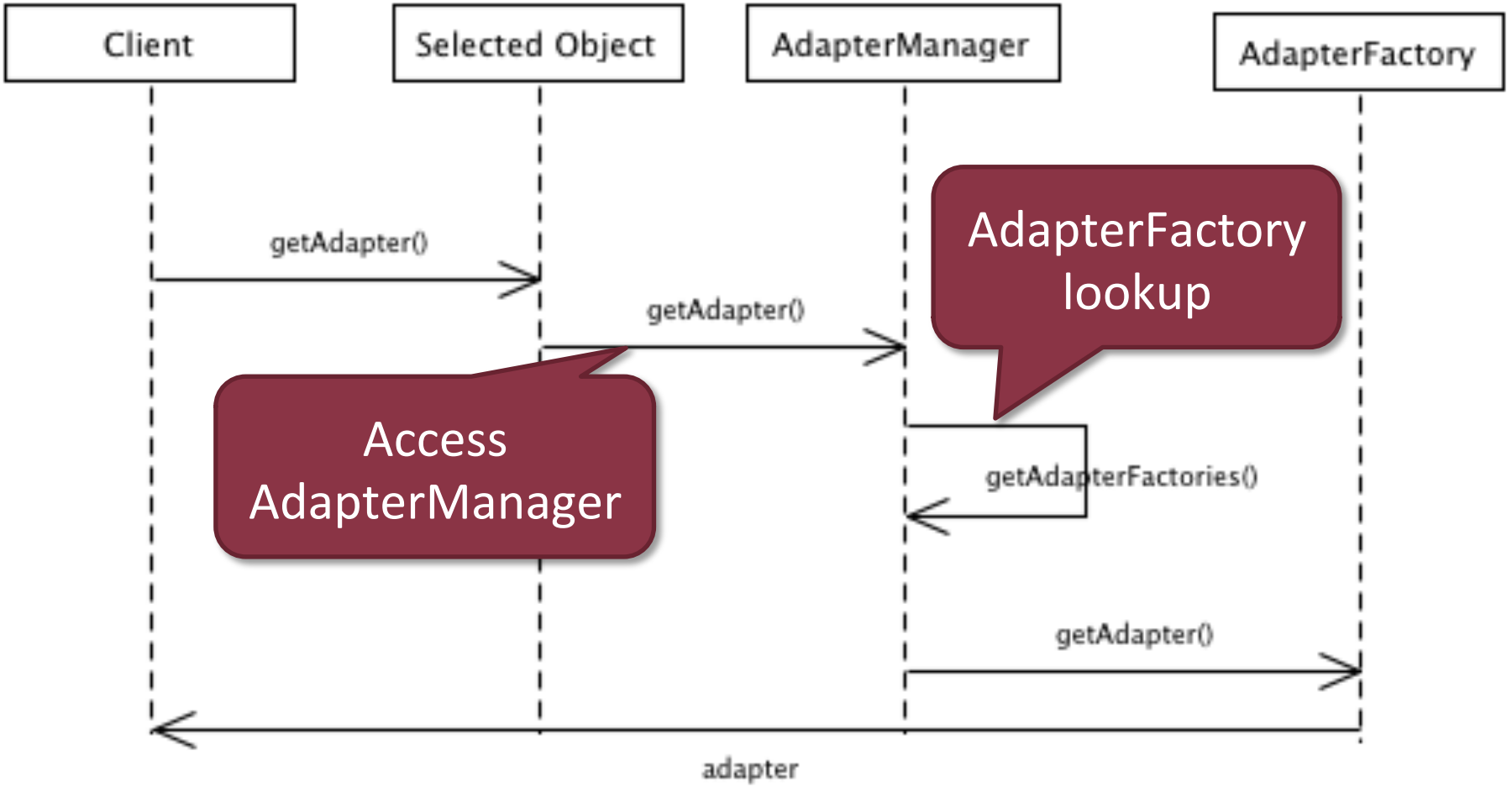
Accessing Adapters



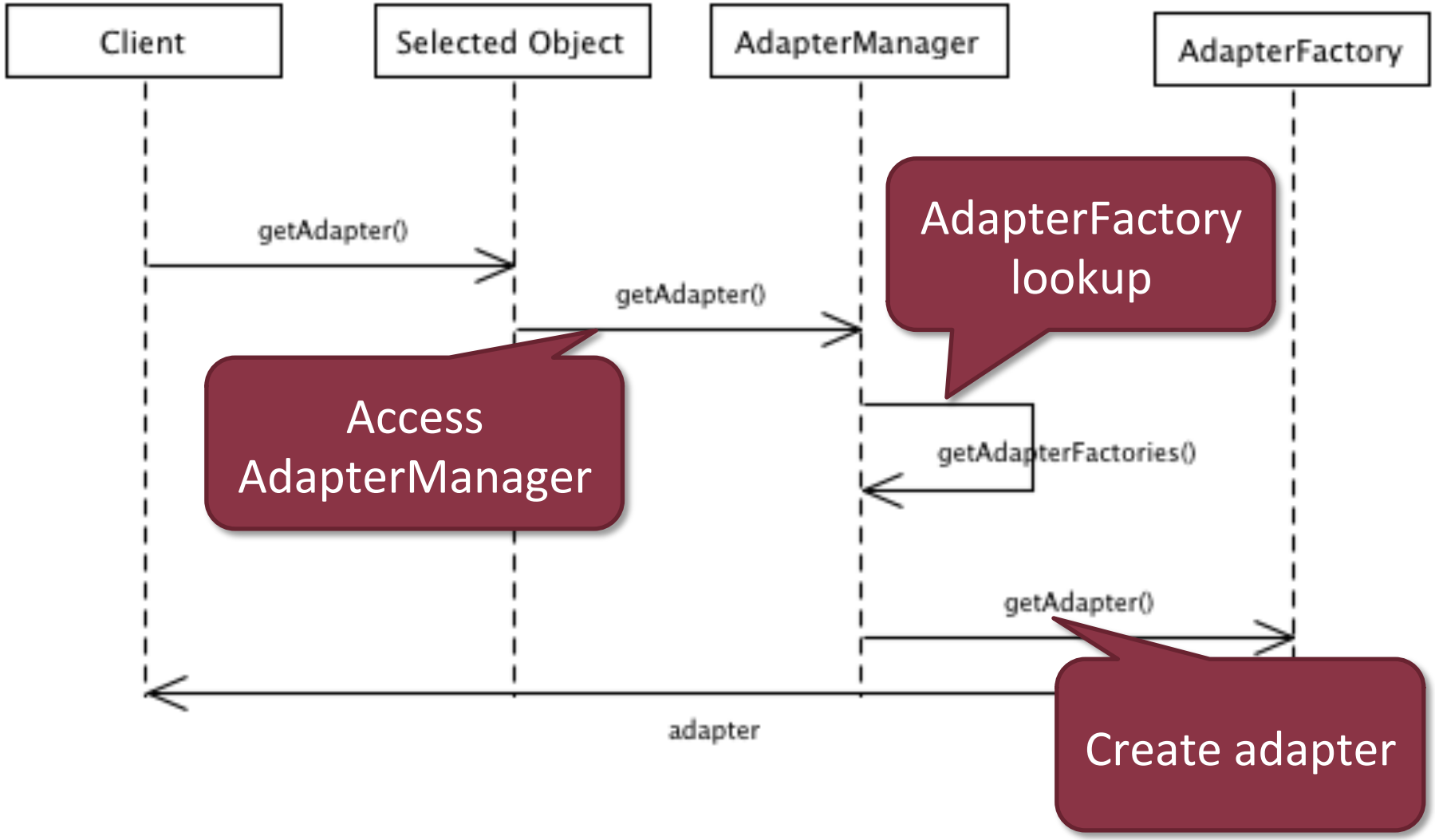
Accessing Adapters



Accessing Adapters



Accessing Adapters



Registering adapter

- AdapterFactory
  - Class that creates an adapter for a model element
  - Important!
    - Do not provide multiple adapters for the same object
- Registration
  - Extension point *org.eclipse.core.runtime.adapters*
  - Information
    - Source type
    - Target type
    - Factory class

More Details

- Dicky Johan: Take control of your properties
  - http://eclipse.org/articles/Article-Properties-View/properties-view.htm
- Jeffrey Ricker: Eclipse Adapters: A Hands-on Hand Holding Explanation
  - http://eclipse.org/resources/resource.php?id=407
- Wayne Beaton: Adapters
  - http://www.eclipse.org/articles/article.php?file=Article-Adapters/index.html

Resource Handling

Resources

- In Eclipse everything is stored in the file system
  - No repository in-between
  - File modifications
    - In Eclipse
    - Outside Eclipse (direct modification)
- Resources
  - Created/modified/deleted
  - State should not be stored multiple times
    - We need stateless references (handles)

Handles

- Combination of Proxy and Bridge design patterns
  - Proxy
    - Replacement for a selected object
    - Manages access control
    - Helps avoiding inconsistent states
  - Bridge
    - Separates interface and implementation
      - Defining higher-level instruction set
    - Independent changes possible
    - Strong separation

Resources

- Handles for file system
  - A key to a file
  - Information object
    - Stores all file information
  - Interfaces: IFile, IFolder, IProject, IWorkspaceRoot
    - Never implement them directly!

Resource handles

- Value types
  - Equality based on referred resource
  - Hashable
  - Multiple handles for the same resource possible
- Defines behaviour
- Does not store state!

Resource Handles – 2.

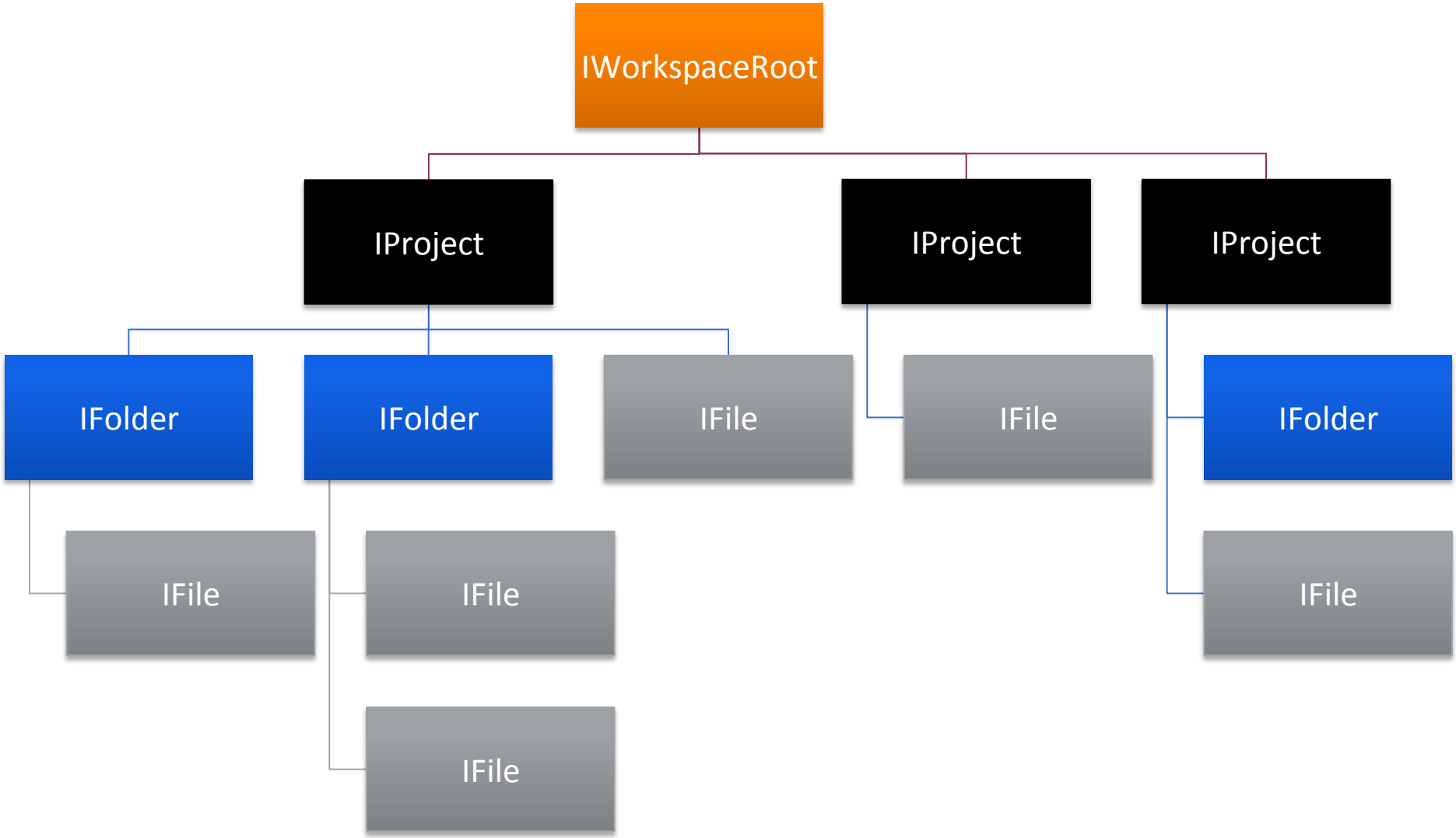
- Handles for non-existent resources possible
  - An operation work if only metadata is required
  - CoreException if resource is required
  - For existence testing: exists()

Resource Handles – 3.

- Creating resources
  - Creating handle from parent handle
    - Root is created by platform
    - See ResourcesPlugin#getWorkspaceRoot
  - Handle has create method

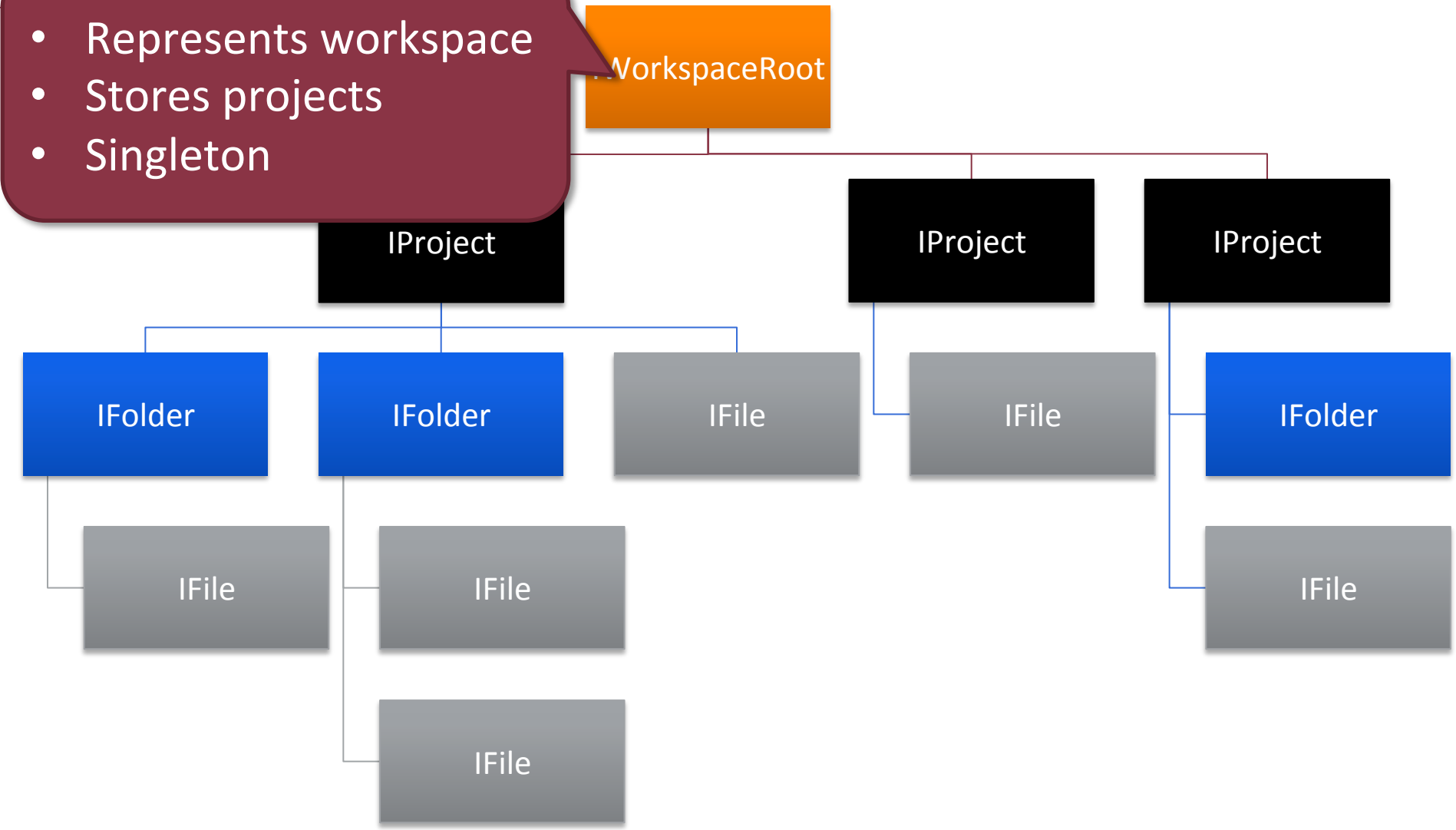
```
IProject project;
IFolder folder =
  project.getFolder(“someFolder”);
folder.create(...);
```

Workspace Hierarchy



Workspace Hierarchy

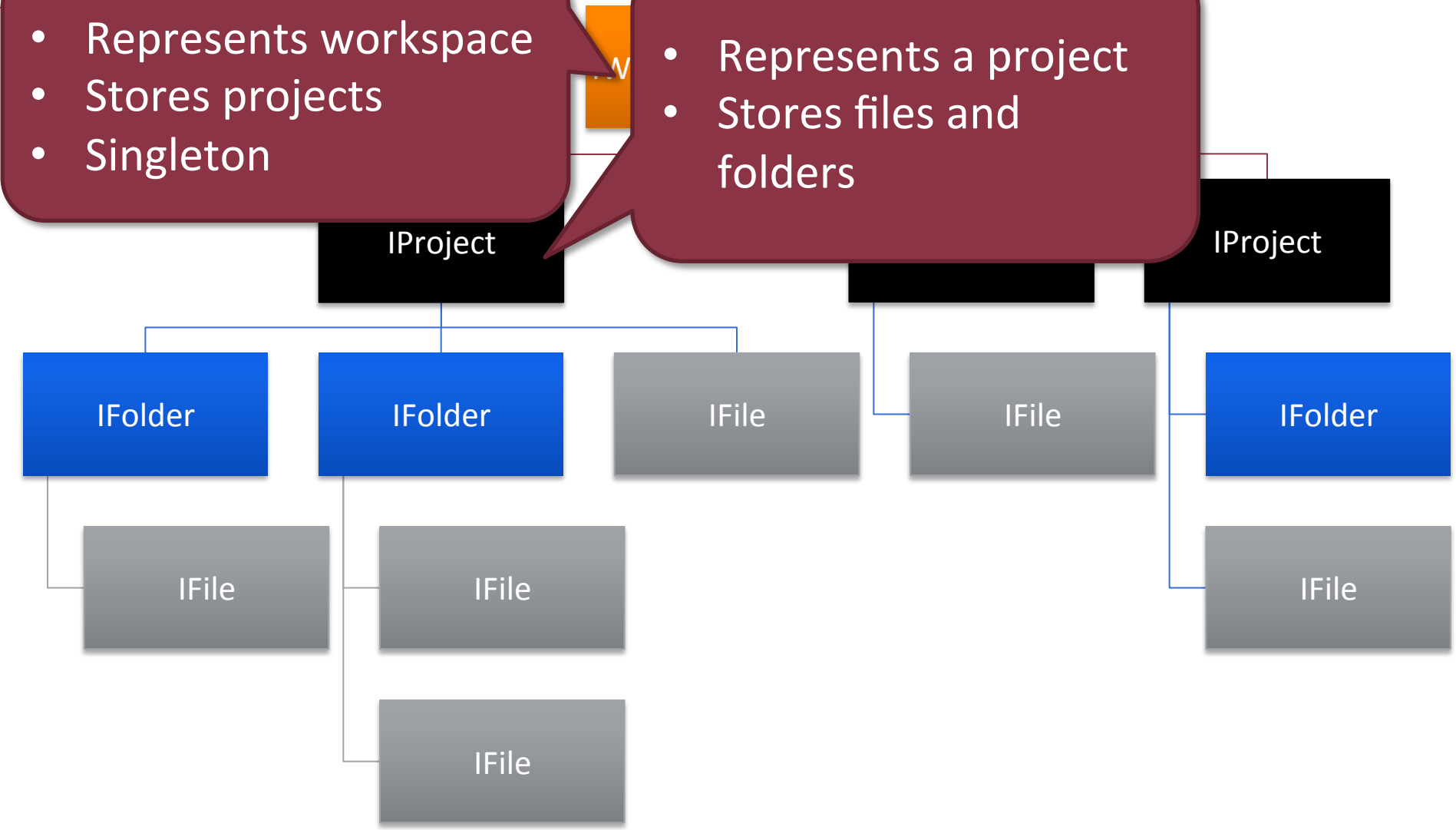
- Represents workspace
- Stores projects
- Singleton



Workspace Hierarchy

- Represents workspace
- Stores projects
- Singleton

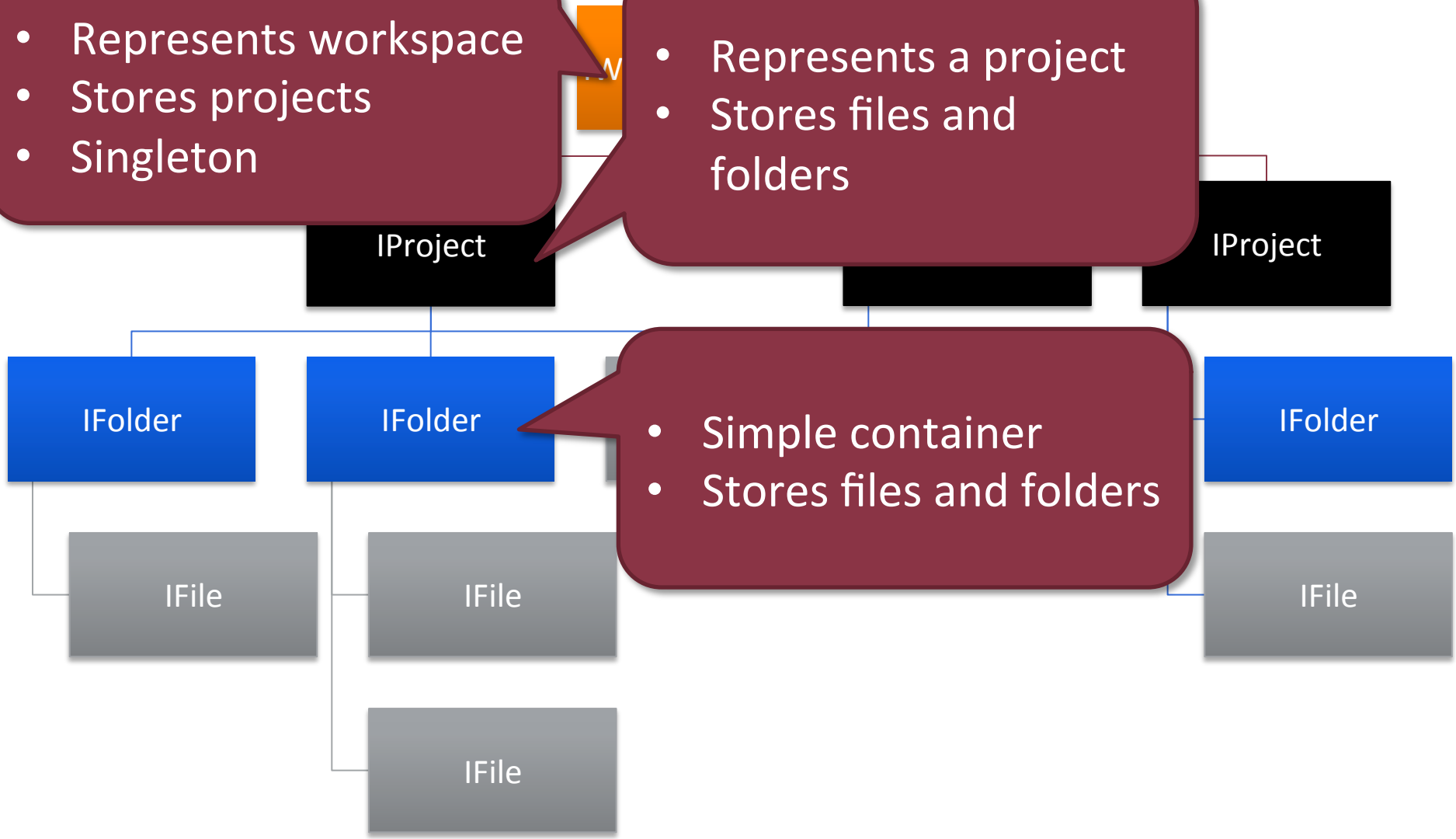
- Represents a project
- Stores files and folders



Workspace Hierarchy

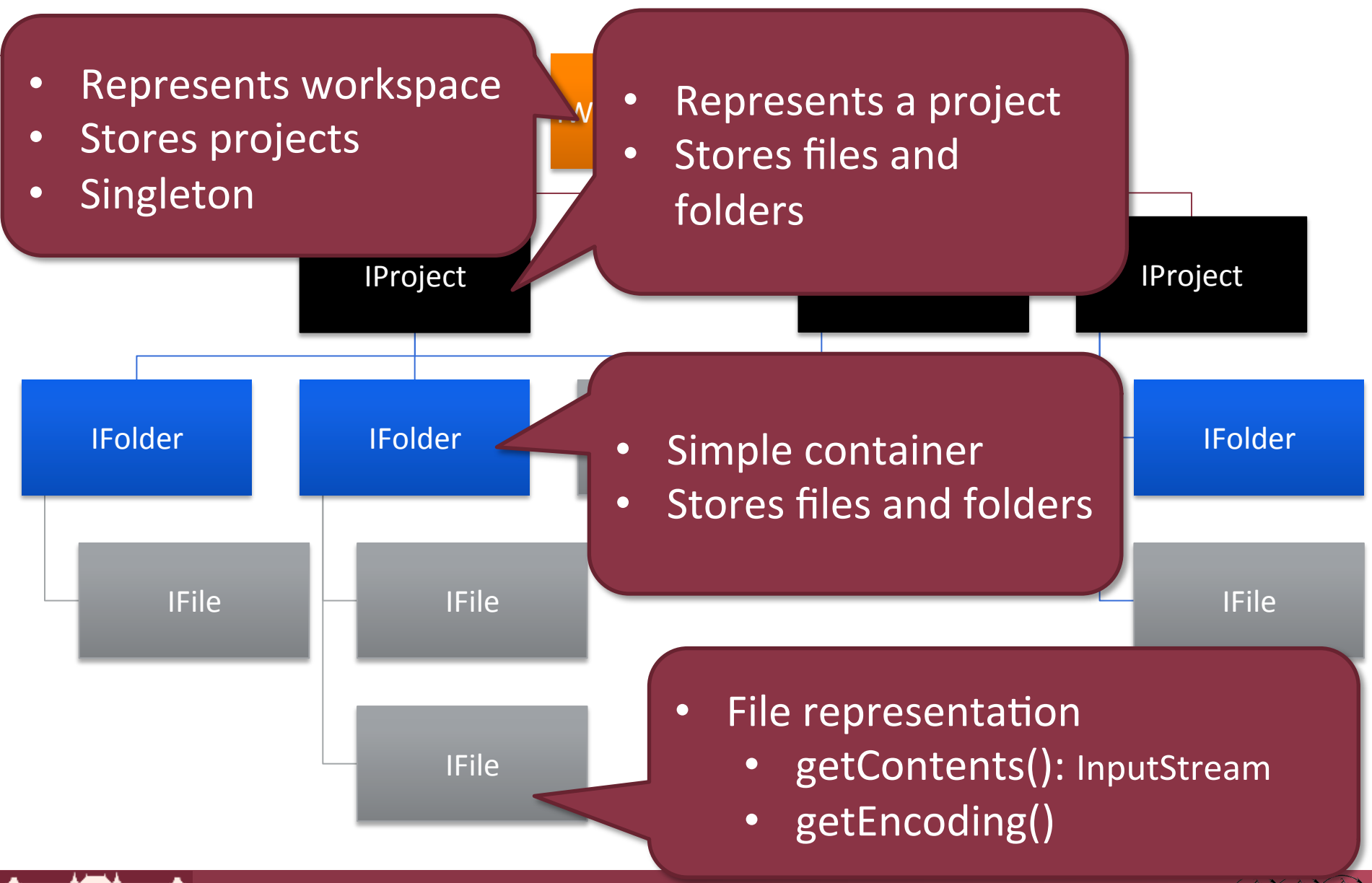
- Represents workspace
- Stores projects
- Singleton

- Represents a project
- Stores files and folders

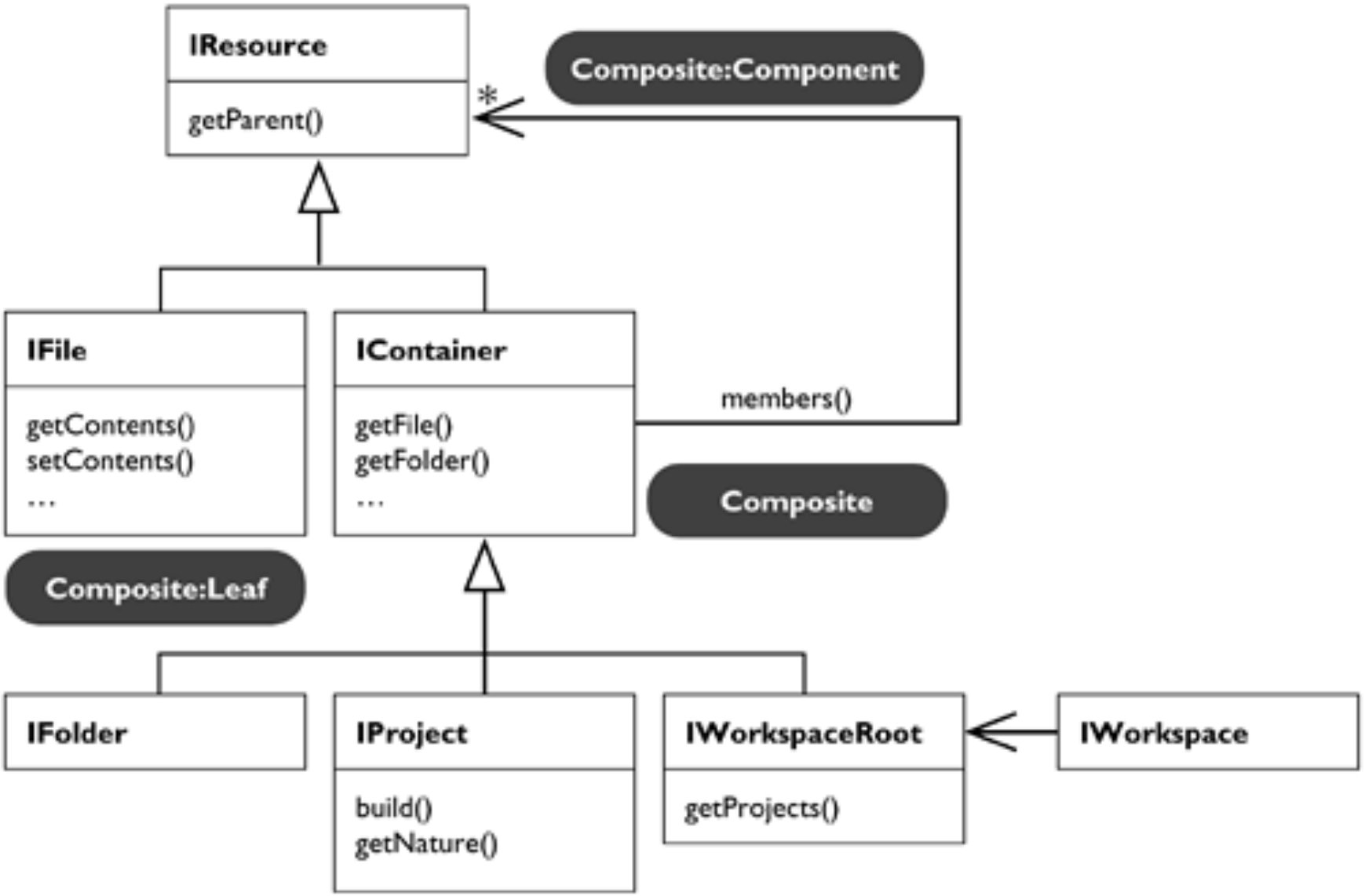


- Simple container
- Stores files and folders

Workspace Hierarchy



Workspace hierarchy – Class diagram



Resource Selection

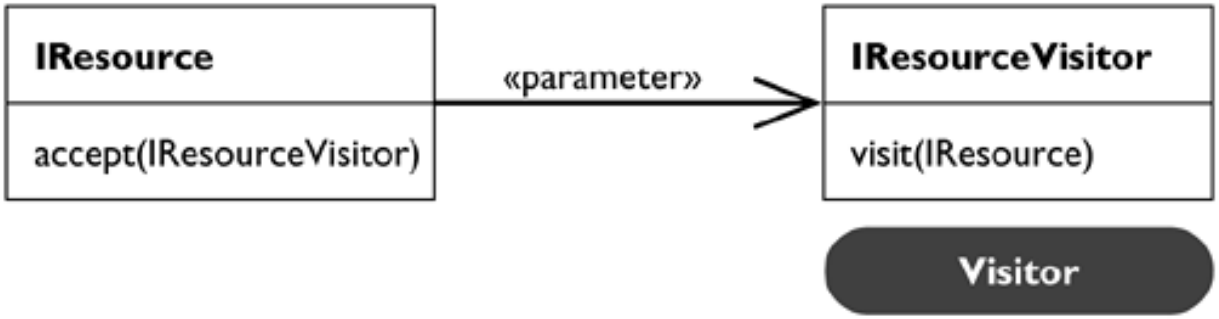
- IPath implementation
  - Describes file path
  - Path operations (e.g. concat) supported
  - **Exception**: Path class can be instantiated
- Works over
  - Real file system
    - Still platform independent
  - Over Eclipse file system
    - Mostly workspace relative
    - But can refer to different external folders (e.g. see Git)

Traversing the Resource Tree – Visitor Pattern

- Visitor pattern
  - Define a single operation
  - Execute it over a model tree
- Traversal is separated for operations
  - Single traversal implementation is enough

Resource Traversal - Visitor

- Structure



- IResource#accept()
  - traverses the subtree
  - Executes the visitor over each Resource
- IResourceVisitor#visit()
  - Executes operation
  - Return value true, if children are to be traversed

Listening to Resource Changes

- Resources can be changed
  - In-Eclipse modifications
  - File system synchronization
- State changes can be observed
  - Efficient state update
  - Workspace registraton

Describing Changes: Resource Delta

Describing Changes: Resource Delta

- Unified storage for changes
  - Either single or multiple changes
- Changes are stored in a tree structure
  - Subset of Resource tree
    - Leaves: changed files
    - Root: Common ancestor of leaves
  - For every node change metadata is stored
    - Created/removed/modified flag
    - For removed elements only the handle is available
  - Processable with a DeltaVisitor

Describing Changes: Resource Delta

- Unified storage for changes
  - Either single or multiple changes
- Changes are stored in a tree structure
  - Subset of Resource tree
    - Leaves: changed files
    - Root: Common ancestor of leaves
  - For every node change metadata is stored
    - Created/removed/modified flag
    - For removed elements only the handle is available
  - Processable with a DeltaVisitor

Describing Changes: Resource Delta

- Unified storage for changes
  - Either single or multiple changes
- Changes are stored in a tree structure
  - Subset of Resource tree
    - Leaves: changed files
    - Root: Common ancestor of leaves
  - For every node change metadata is stored
    - Created/removed/modified flag
    - For removed elements only the handle is available
  - Processable with a DeltaVisitor

Describing Changes: Resource Delta

- Unified storage for changes
  - Either single or multiple changes
- Changes are stored in a tree structure
  - Subset of Resource tree
    - Leaves: changed files
    - Root: Common ancestor of leaves
  - For every node change metadata is stored
    - Created/removed/modified flag
    - For removed elements only the handle is available
  - Processable with a DeltaVisitor

Describing Changes: Resource Delta

- Unified storage for changes
  - Either single or multiple changes
- Changes are stored in a tree structure
  - Subset of Resource tree
    - Leaves: changed files
    - Root: Common ancestor of leaves
  - For every node change metadata is stored
    - Created/removed/modified flag
    - For removed elements only the handle is available
  - Processable with a DeltaVisitor

Batch Changes

- Avoid change event storms!
- Composite actions
  - IWorkspaceRunnable
  - Reduced number of notifications
    - In theory, only at the end of the operation
    - Sometimes more often

Batch Changes - Example

```
public static void createMarker(final IResource
  resource, final Map attributes, final String
  markerType) throws CoreException {
  IWorkspaceRunnable r = new IWorkspaceRunnable()
  {
    public void run(IProgressMonitor monitor)
  throws CoreException {
      IMarker marker =
  resource.createMarker(markerType);
      marker.setAttributes(attributes);
    }
  };
  resource.getWorkspace().run(r, null);
}
```

Markers

- Assigning custom data for IResource instances
  - E.g. error or bookmark markers
  - State is saved
    - Depending on Marker type
    - Saving is automatic
- Use IMarker interface
  - Never implement it directly
  - Data is stored via key-value pairs

More Details

- John Arthorne: How You've Changed!
  - http://eclipse.org/articles/Article-Resource-deltas/resource-deltas.html
- Dejan Glozic: Mark My Words!
  - http://eclipse.org/articles/Article-Mark%20My%20Words/mark-my-words.html

Project Handling

Project Natures

- Example
  - Java project
  - Plug-in project (also a Java project!)
  - …
- Tasks
  - Identifier -> code/command can be executed
  - Configuration -> dependent on project types
  - Compiling -> registering builders

Project Natures

- Nature extension point
  - org.eclipse.core.resources.natures
  - Abstract definition
  - Used for attaching additional services

- Testing whether a project has a nature
  - IProject#hasNature(natureID)

Creating a Project

- IProject creating
  - DO NOT implement
  - Use the following pattern:

```
try {
    IProject p = ResourcesPlugin.getWorkspace().
            getRoot().getProject();
    p.create(new NullProgressMonitor());
} catch (CoreException e1) {
    //Exception handling
}
```

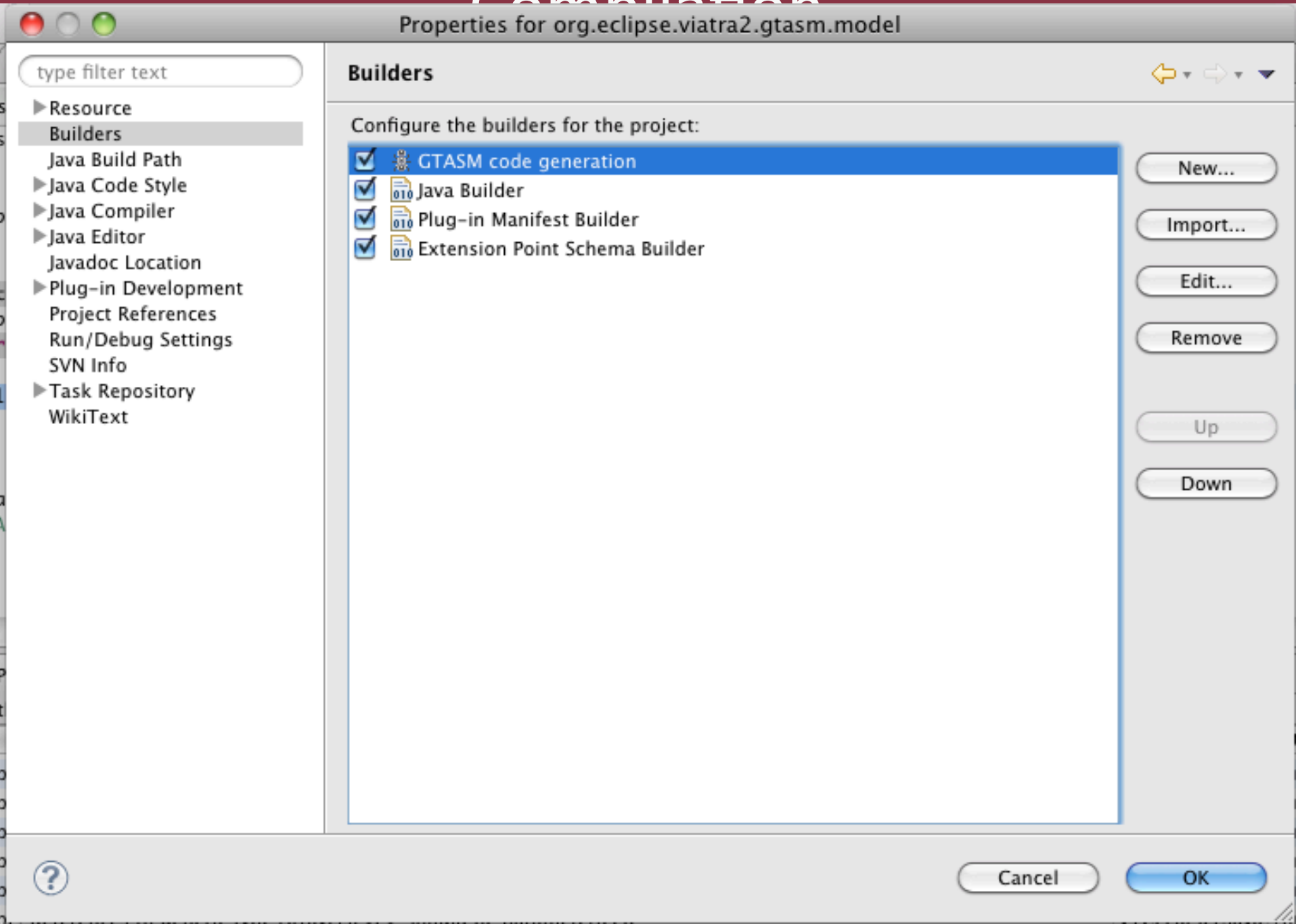
Compilation

- Incremental Builder
- Build types
  - INCREMENTAL
    - Build Project/Build All
  - AUTO
    - Automatically triggered build
    - The same as incremental
  - FULL
    - Do not try to reuse previous build states
  - CLEAN
    - Remove previous build state

Compilation

- External builders
  - Ant build script
  - Batch file/Shell script
  - Other external builder (e.g. parser generator)
  - …

Compilation



Creating Builder

- Integrated builders
  - Extension point: org.eclipse.core.resources.builder
  - IncrementalProjectBuilder base class
- Registration
  - Commonly via nature
  - Needs to be done in code

.project file – 1.

```
<?xml version="1.0" encoding="UTF-8"?>
<projectDescription>
        <name>org.eclipse.viatra2.gtasm.interpreter</name>
        <comment></comment>
        <projects>
        <project>viatra_gtasm_emf_model</project>
        <project>viatra_pattern_matcher</project>
        </projects>
        <buildSpec>
        <buildCommand>
        <name>org.eclipse.jdt.core.javabuilder</name>
        <arguments>
        </arguments>
        </buildCommand>
        <buildCommand>
```

.project file – 2.

```
        <name>org.eclipse.pde.ManifestBuilder</name>
        <arguments>
        </arguments>
        </buildCommand>
        <buildCommand>
        <name>org.eclipse.pde.SchemaBuilder</name>
        <arguments>
        </arguments>
        </buildCommand>
        </buildSpec>
        <natures>
        <nature>org.eclipse.pde.PluginNature</nature>
        <nature>org.eclipse.jdt.core.javanature</nature>
        </natures>
</projectDescription>
```

More Details

- John Artorne: Project Builders and Natures
  - http://eclipse.org/articles/Article-Builders/builders.html

Background Jobs

Job API

Background tasks

- Execute long-running tasks in background
  - Avoid user interface lagging
- Eclipse support
  - Job: a single background job
  - Scheduling
  - UI feedback

Job

- Custom Job descendant class
- Most important methods
  - run(IProgressMonitor monitor)
    - Executable code – always redefine it
  - setName(String name)
    - Set task name
  - schedule(long delay) / schedule()
    - Start task

Hello, world! Job

```
Job job = new Job("My First Job") {
  protected IStatus
  run(IProgressMonitor monitor) {
      System.out.println("Hello
  World (from a background job)");
      return Status.OK_STATUS;
  } };
job.setPriority(Job.SHORT);
job.schedule(); // start as soon as
  possible
```

Job State

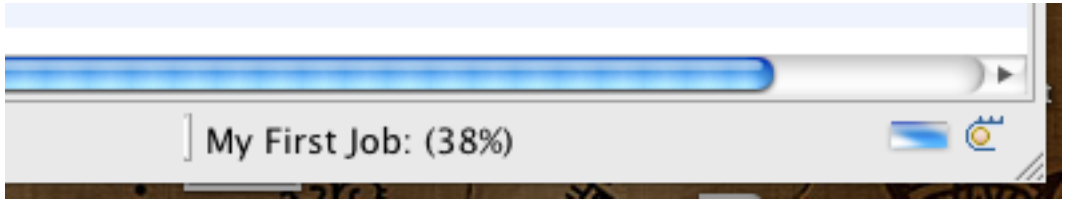
- IProgressMonitor
  - Interface for feedback
  - Name and state stored
- State changes
  - Executed task updates it
  - Stores a cancel request

Feedback via IProgressMonitor

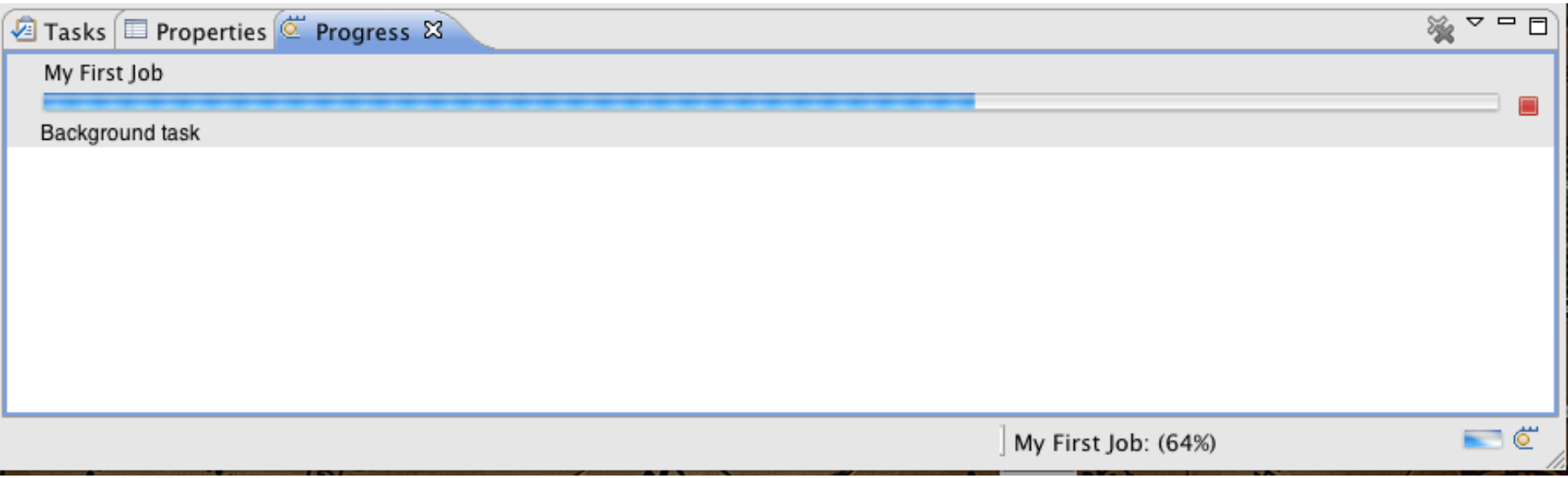
```
Job job = new Job("My First Job") {
  protected IStatus run(IProgressMonitor
  monitor) {
    monitor.beginTask("Background task",
  1000);
    for (int i=0; i<1000; i++){
      calculationStep();
      monitor.worked(1);
    }
    monitor.done();
    return Status.OK_STATUS;
  }
};
```

Feedback – 1.

- Status bar



- Progress view



Feedback – 2.

- In case of user tasks pop-up dialog
  - Setting
    - job.setUser(true);
  - Only for tasks directly initialized by users
    - Model pop-up
    - Closable (for ever)

Cancelling jobs

- Cancel can be initialized
  - User interface
    - Red stop button in feedback forms
  - In code
    - Call Job#`cancel()`
- This is just a request
  - Does not interrupt execution
  - Executing code needs to check for this

Stop task

- Stopping task
  - Poll IProgressMonitor#isCanceled()
  - Exit
    - `return Status.CANCEL_STATUS;`
    - Throw `OperationCanceledException`
      - Even deep in the call hierarchy

Cancelling job

```
Job job = new Job("My First Job") {
  protected IStatus run(IProgressMonitor monitor) {
    monitor.beginTask("Background task", 1000);
    for (int i=0; i<1000; i++){
      calculationStep();
      monitor.worked(1);
      if (monitor.isCancelled())
        return STATUS.CANCEL_STATUS;
    }
    monitor.done();
    return Status.OK_STATUS;
  }
};
```

Call display thread

- UI can only be updated on the display thread
  - Display#syncExec(Runnable r)
    - Background thread waits
  - Display#asyncExec(Runnable r)
    - Background thread does not wait

More Details

- Michael Valenta: On the Job: The Eclipse Jobs API
  - http://eclipse.org/articles/Article-Concurrency/jobs-api.html

Case Study: JDT

Resources and Java Elements

- Java-centric user interface
  - Extends Resource via Adapter pattern
    - Java Object Model
      - Program structure, classes, etc.
    - Abstract Syntax Tree
      - Java class with all expressions

Java model	Resources
Packages	Resources
CompilationUnit	Files

Java Object Model

- Classes
  - IJavaModel – Contains all Java projects (root)
  - IJavaProject – Single Java project, with settings
  - IPackageFragmentRoot – Java folder (jar, zip)
  - IPackageFragment – A package (or part of)

Java Object Model

- Further classes
  - ICompilationUnit – A single source file
  - IPackageDeclaration – Package declaration ina file
  - IImportContainer – All imports of a file
  - IImportDeclaration – A single import of a file
  - IType – A type definition (e.g. class, interface, enum)
  - IField – an attribute of a class
  - IMethod – a method of a class
  - IInitializer – Static initializer block
  - IClassFile – Represents a binary (class) file

Java Object Model

- WorkingCopy – local copy to work on
  - Primary: shared between users
    - commitWorkingCopy() – finish change transaction
    - discardWorkingCopy() – rollback transaction
  - Own copy
    - getWorkingCopy(owner)
    - commitWorkingCopy()
    - E.g. used by refactoring to preview changes

Java Object Model

- Events
  - IJavaElementDelta – element changes
    - ADDED – added
    - REMOVED – removed
    - CHANGED - changes
      - F_CHILDREN – children changed
      - F_CONTENT – content changed
      - …
- Event handling possible

Abstract Syntax Tree

Abstract Syntax Tree

- Abstract Syntax Tree (AST)
  - Statement-level model
  - Goal: source code analysis and manipulation
  - Why two models?
    - Reduced resource usage
    - Speed

Summary

Table of Contents

- Connecting Views and Editors
  - Workbench Selection Service
  - Using Adapters (with Properties view)
- Resource Handling
  - File System
  - Project Management
- Background Jobsda
- Case Study: JDT

Motto

- Any problem in computer science can be solved with another level of indirection.
- … But that usually will create another problem.

David Wheeler

void ihate()

eclipse

don't be shy, just say it!

tell us why?

i want to provide my details

SEND TO HATE LOG

[180]

Dear Eclipse! If there is any file modifications outside of your weird interface, don't blow and tell me that I have to press F5. Just refresh it or gimme an option for it. Arghhh!

anger level **(844)**

Isa Goksu thinks like this, what about you? I agree | I disagree | add a comment

Workspace default settings: spell checking on, showing editor line numbers off. that totally makes sense.

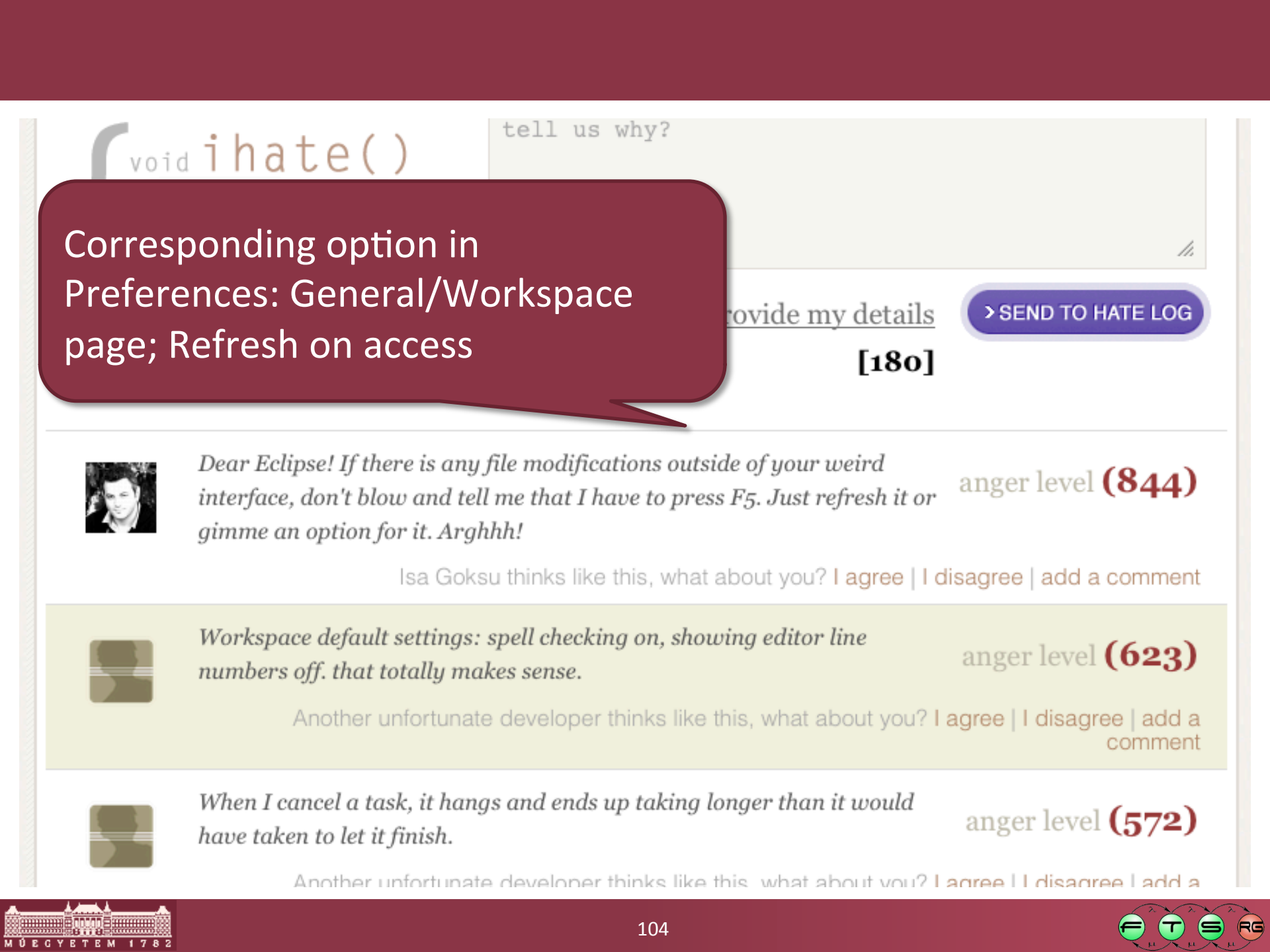
anger level **(623)**

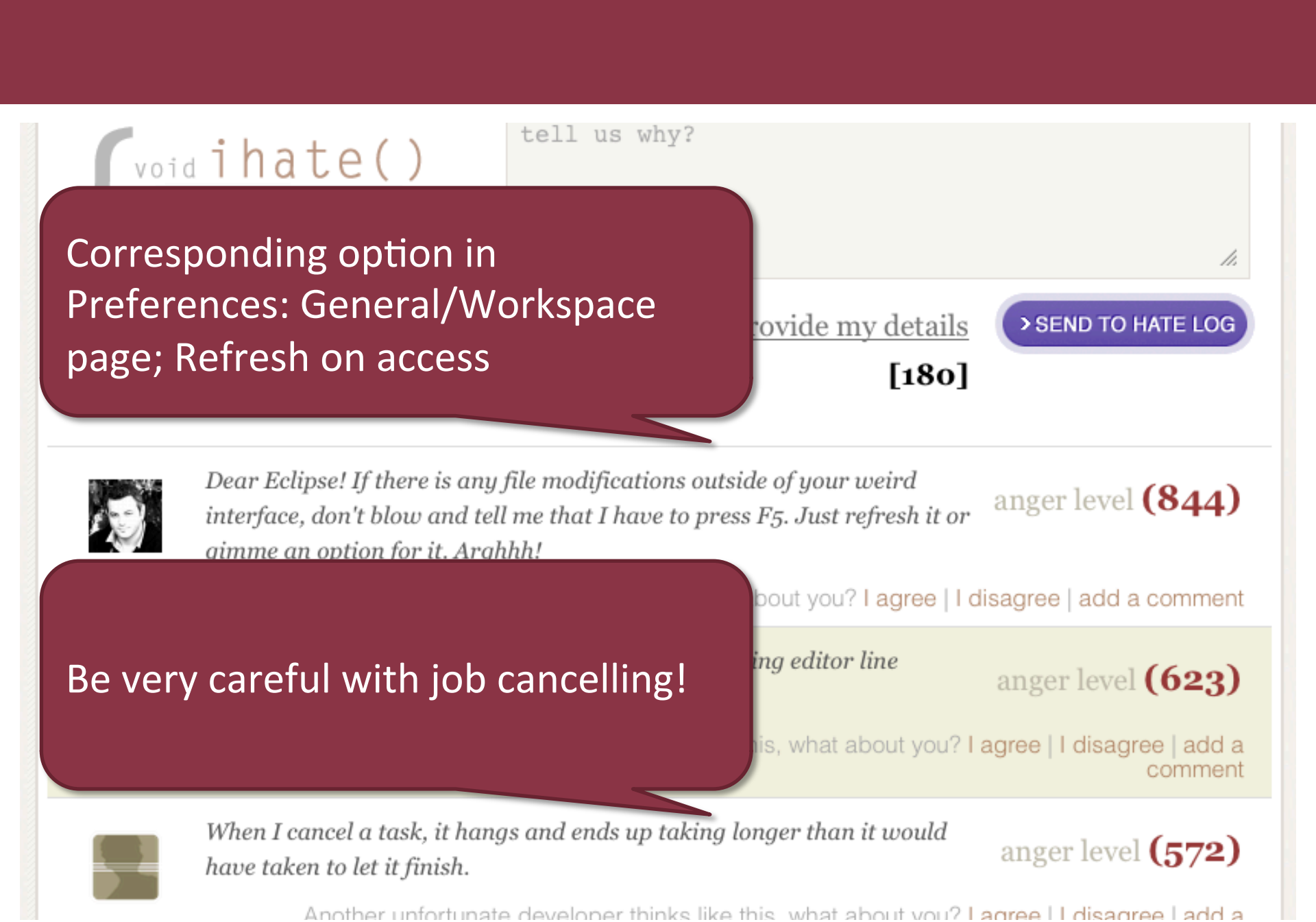
Another unfortunate developer thinks like this, what about you? I agree | I disagree | add a comment

When I cancel a task, it hangs and ends up taking longer than it would have taken to let it finish.

anger level **(572)**

Another unfortunate developer thinks like this, what about you? I agree | I disagree | add a


void ihate()
tell us why?
Corresponding option in Preferences: General/Workspace page; Refresh on access
rovide my details
SEND TO HATE LOG
[180]
Dear Eclipse! If there is any file modifications outside of your weird interface, don't blow and tell me that I have to press F5. Just refresh it or gimme an option for it. Arghhh!
anger level (844)
Isa Goksu thinks like this, what about you? I agree | I disagree | add a comment
Workspace default settings: spell checking on, showing editor line numbers off. that totally makes sense.
anger level (623)
Another unfortunate developer thinks like this, what about you? I agree | I disagree | add a comment
When I cancel a task, it hangs and ends up taking longer than it would have taken to let it finish.
anger level (572)


void ihate()
tell us why?
rovide my details
SEND TO HATE LOG
[180]
Dear Eclipse! If there is any file modifications outside of your weird interface, don't blow and tell me that I have to press F5. Just refresh it or gimme an option for it. Arghhh!
anger level (844)
bout you? I agree | I disagree | add a comment
ing editor line
anger level (623)
is, what about you? I agree | I disagree | add a comment
When I cancel a task, it hangs and ends up taking longer than it would have taken to let it finish.
anger level (572)
Another unfortunate developer thinks like this, what about you? I agree | I disagree | add a
Corresponding option in Preferences: General/Workspace page; Refresh on access
Be very careful with job cancelling!