ODCSSS 2007

Jakub Dostál, UCD

Project 13 – Programmer and Prover Memory
Project Supervisor: Dr. Joe Kiniry
Mentor: Radu Grigore

Facts:

Formal methods are not widely used,

Question 1:
Have you ever used formal methods?

especially for larger projects.

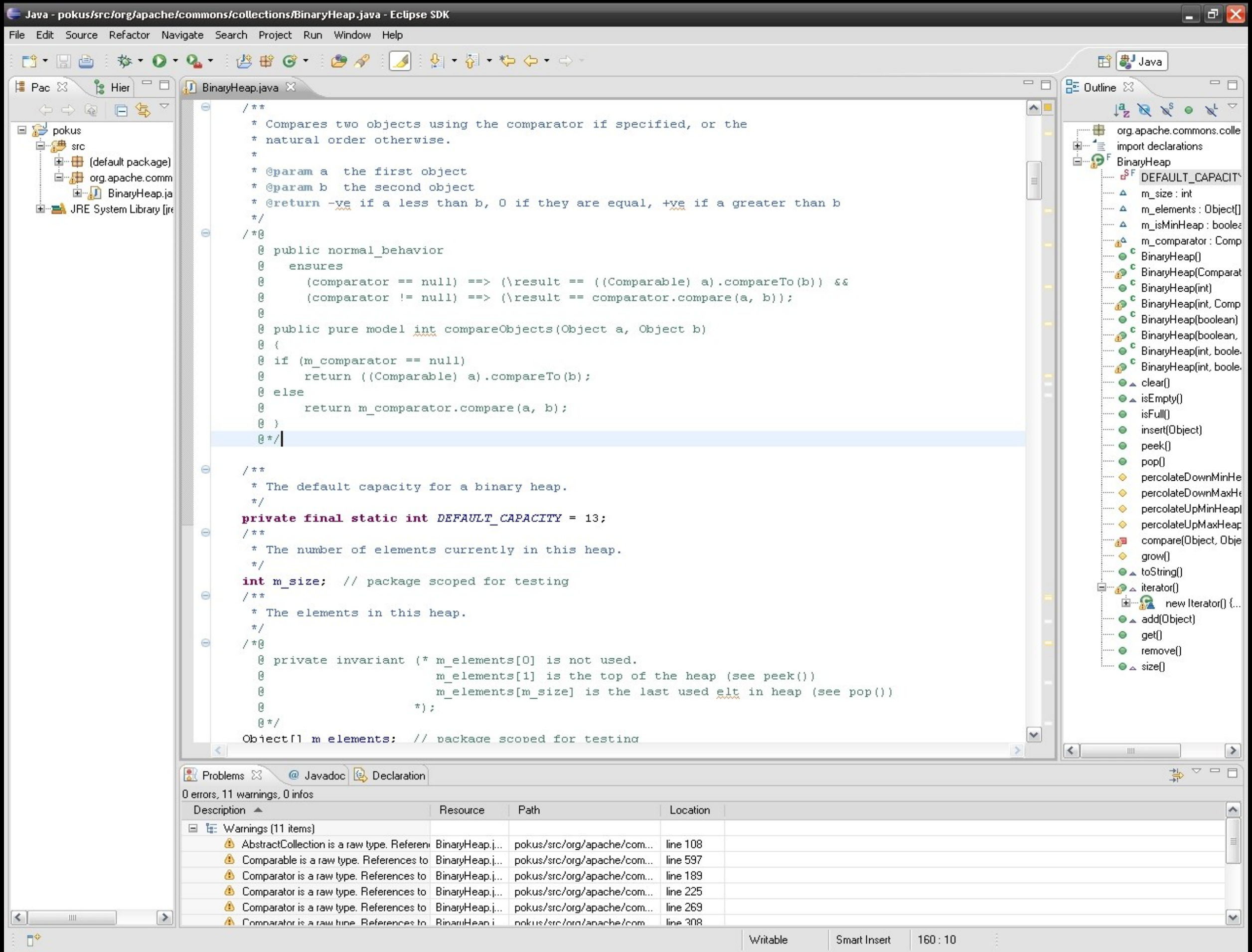
Question 2:
Have you ever worked with a piece
of code that someone else wrote?
Did you get lost in it?

But where there is will...

A lot of developers use...

IDEs like **Eclipse**

so...

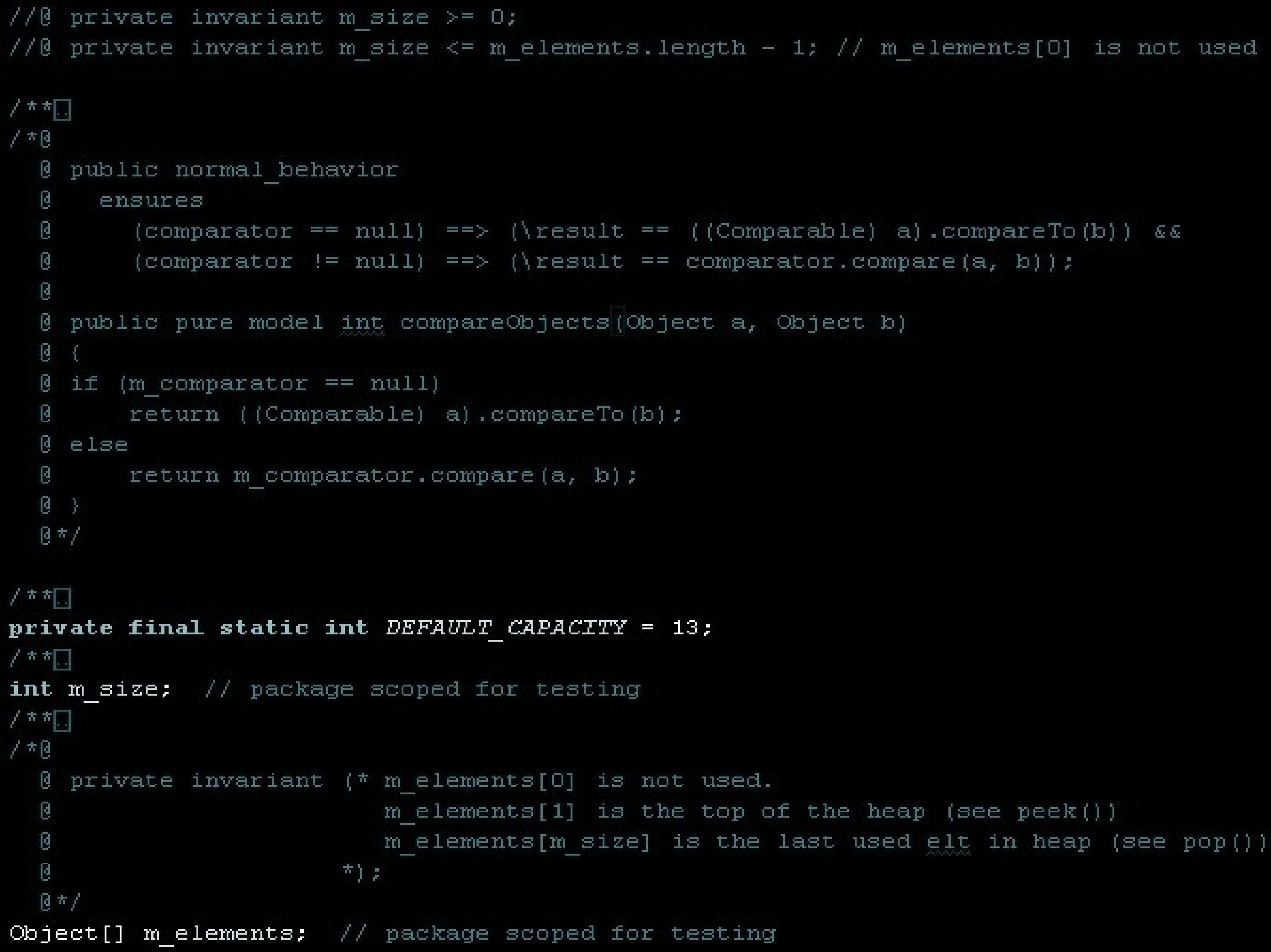


```
    /**
     * Compares two objects using the comparator if specified, or the
     * natural order otherwise.
     *
     * @param a  the first object
     * @param b  the second object
     * @return -ve if a less than b, 0 if they are equal, +ve if a greater than b
     */
    /*@
      @ public normal_behavior
      @   ensures
      @     (comparator == null) ==> (\result == ((Comparable) a).compareTo(b)) &&
      @     (comparator != null) ==> (\result == comparator.compare(a, b));
      @
      @ public pure model int compareObjects(Object a, Object b)
      @ {
      @ if (m_comparator == null)
      @     return ((Comparable) a).compareTo(b);
      @ else
      @     return m_comparator.compare(a, b);
      @ }
      @*/

    /**
     * The default capacity for a binary heap.
     */
    private final static int DEFAULT_CAPACITY = 13;
    /**
     * The number of elements currently in this heap.
     */
    int m_size;  // package scoped for testing
    /**
     * The elements in this heap.
     */
    /*@
      @ private invariant (* m_elements[0] is not used.
      @                      m_elements[1] is the top of the heap (see peek())
      @                      m_elements[m_size] is the last used elt in heap (see pop())
      @                    *);
      @*/
    Object[] m_elements;  // package scoped for testing
```

0 errors, 11 warnings, 0 infos

Description	Resource	Path	Location
Warnings (11 items)			
AbstractCollection is a raw type. Referen	BinaryHeap.j...	pokus/src/org/apache/com...	line 108
Comparable is a raw type. References to	BinaryHeap.j...	pokus/src/org/apache/com...	line 597
Comparator is a raw type. References to	BinaryHeap.j...	pokus/src/org/apache/com...	line 189
Comparator is a raw type. References to	BinaryHeap.j...	pokus/src/org/apache/com...	line 225
Comparator is a raw type. References to	BinaryHeap.j...	pokus/src/org/apache/com...	line 269
Comparator is a raw type. References to	BinaryHeap.j...	pokus/src/org/apache/com...	line 308

...one way to motivate them to use formal methods ...
(e.g. specification languages like **JML**)



```
//@ private invariant m_size >= 0;
//@ private invariant m_size <= m_elements.length - 1; // m_elements[0] is not used

/**
/*@
  @ public normal_behavior
  @   ensures
  @     (comparator == null) ==> (\result == ((Comparable) a).compareTo(b)) &&
  @     (comparator != null) ==> (\result == comparator.compare(a, b));
  @
  @ public pure model int compareObjects(Object a, Object b)
  @ {
  @ if (m_comparator == null)
  @     return ((Comparable) a).compareTo(b);
  @ else
  @     return m_comparator.compare(a, b);
  @ }
  @*/

/**
private final static int DEFAULT_CAPACITY = 13;
/**
int m_size;  // package scoped for testing
/**
/*@
  @ private invariant (* m_elements[0] is not used.
  @                      m_elements[1] is the top of the heap (see peek())
  @                      m_elements[m_size] is the last used elt in heap (see pop())
  @                   *);
  @*/
Object[] m_elements;  // package scoped for testing
```

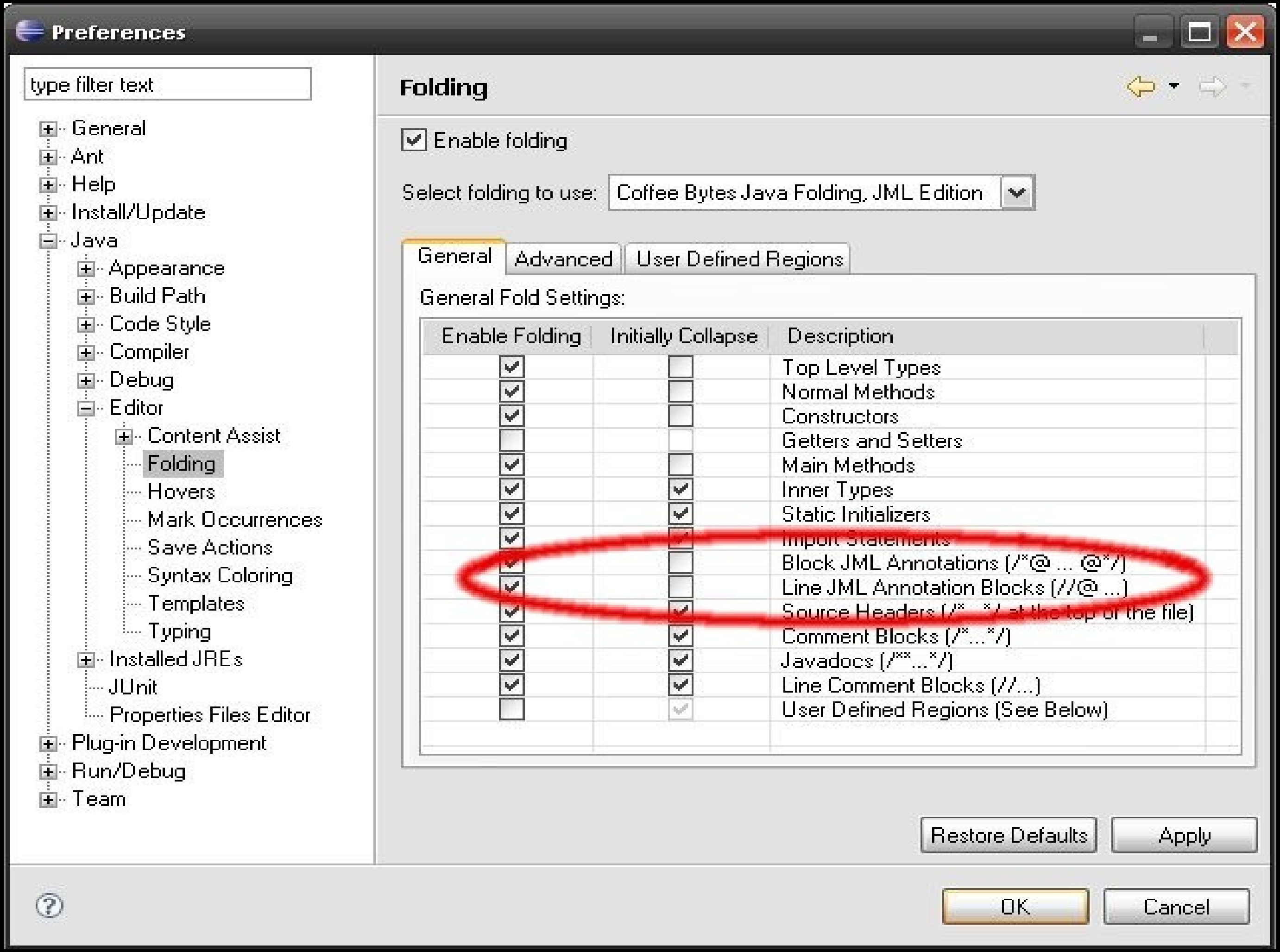
... is to provide them with tools that
will make it easy for them to use them, ...

... which is where I come in.

(and obviously my supervisors group and a whole lot of other people :o))

So what have I done so far?

Extended folding facilities of Eclipse
to support JML annotations
(i.e. support for formal methods...)



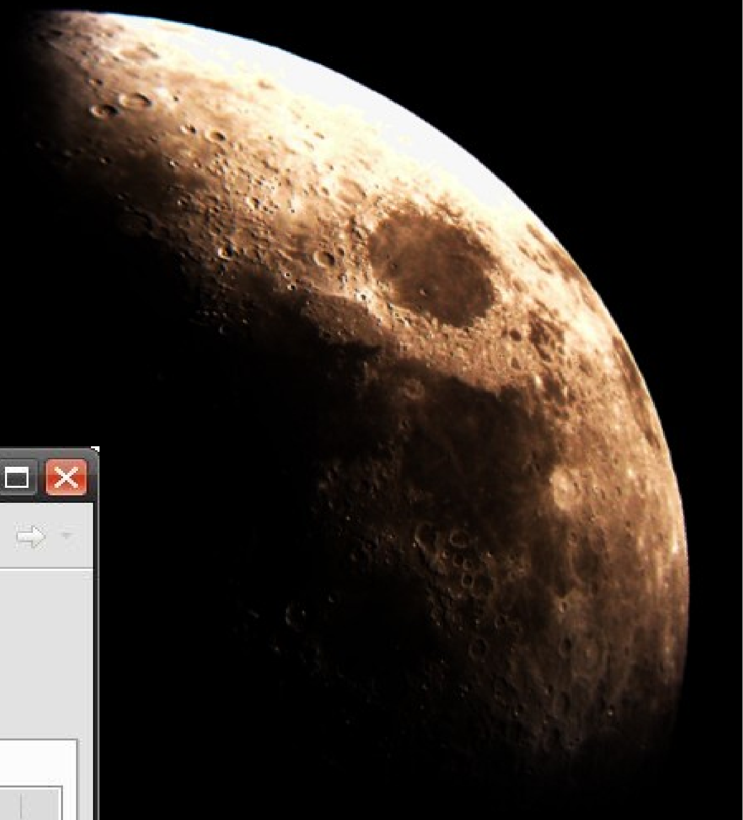
Folding

☑ Enable folding

Select folding to use: Coffee Bytes Java Folding, JML Edition

General Fold Settings:

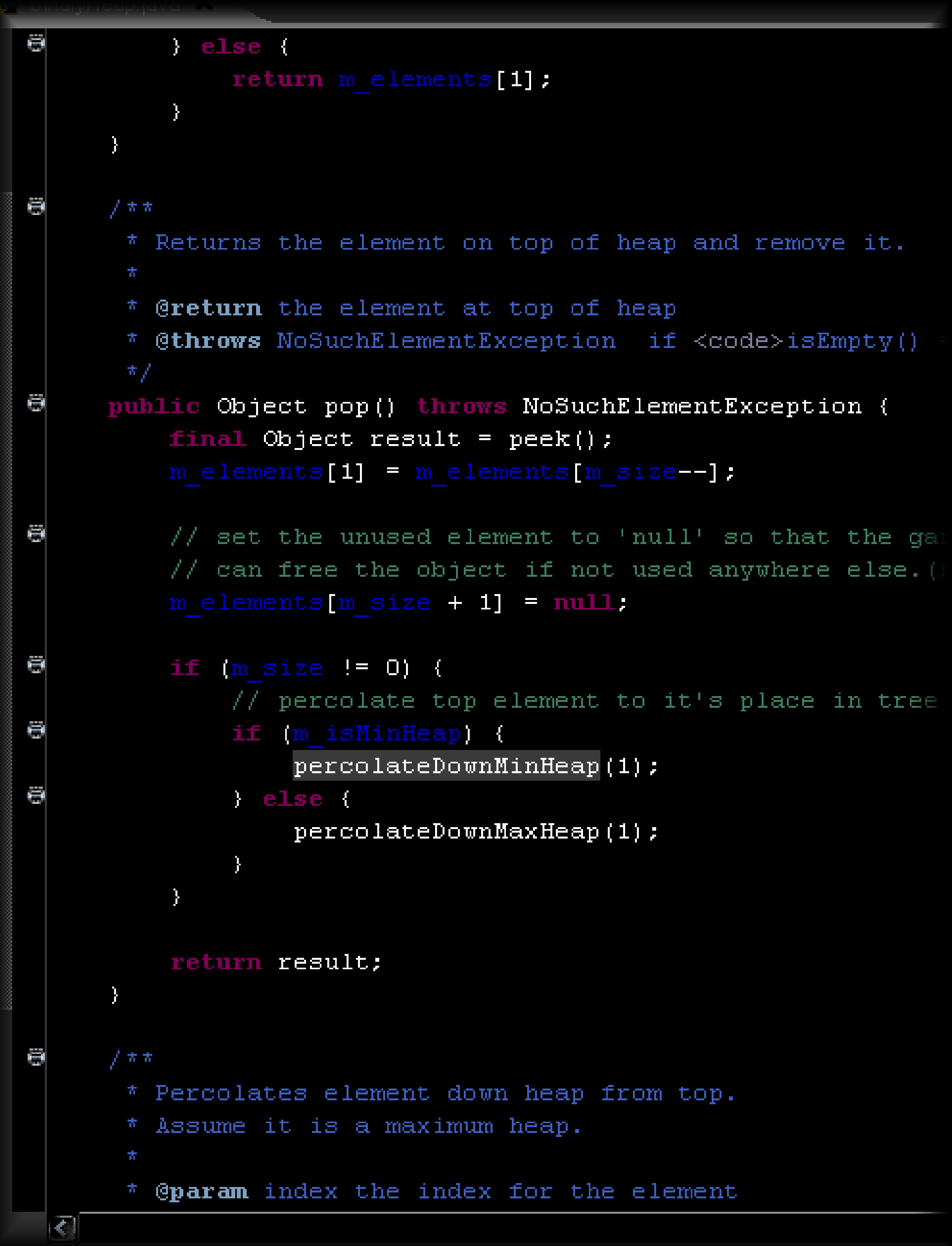
Enable Folding	Initially Collapse	Description
☑	□	Top Level Types
☑	□	Normal Methods
☑	□	Constructors
□	□	Getters and Setters
☑	□	Main Methods
☑	☑	Inner Types
☑	☑	Static Initializers
☑	☑	Import Statements
☑	□	Block JML Annotations (/*@ ... @*/)
☑	□	Line JML Annotation Blocks (//@ ...)
☑	☑	Source Headers (/*...*/ at the top of the file)
☑	☑	Comment Blocks (/*...*/)
☑	☑	Javadocs (/**...*/)
☑	☑	Line Comment Blocks (//...)
□	☑	User Defined Regions (See Below)



And what am I
working on now?

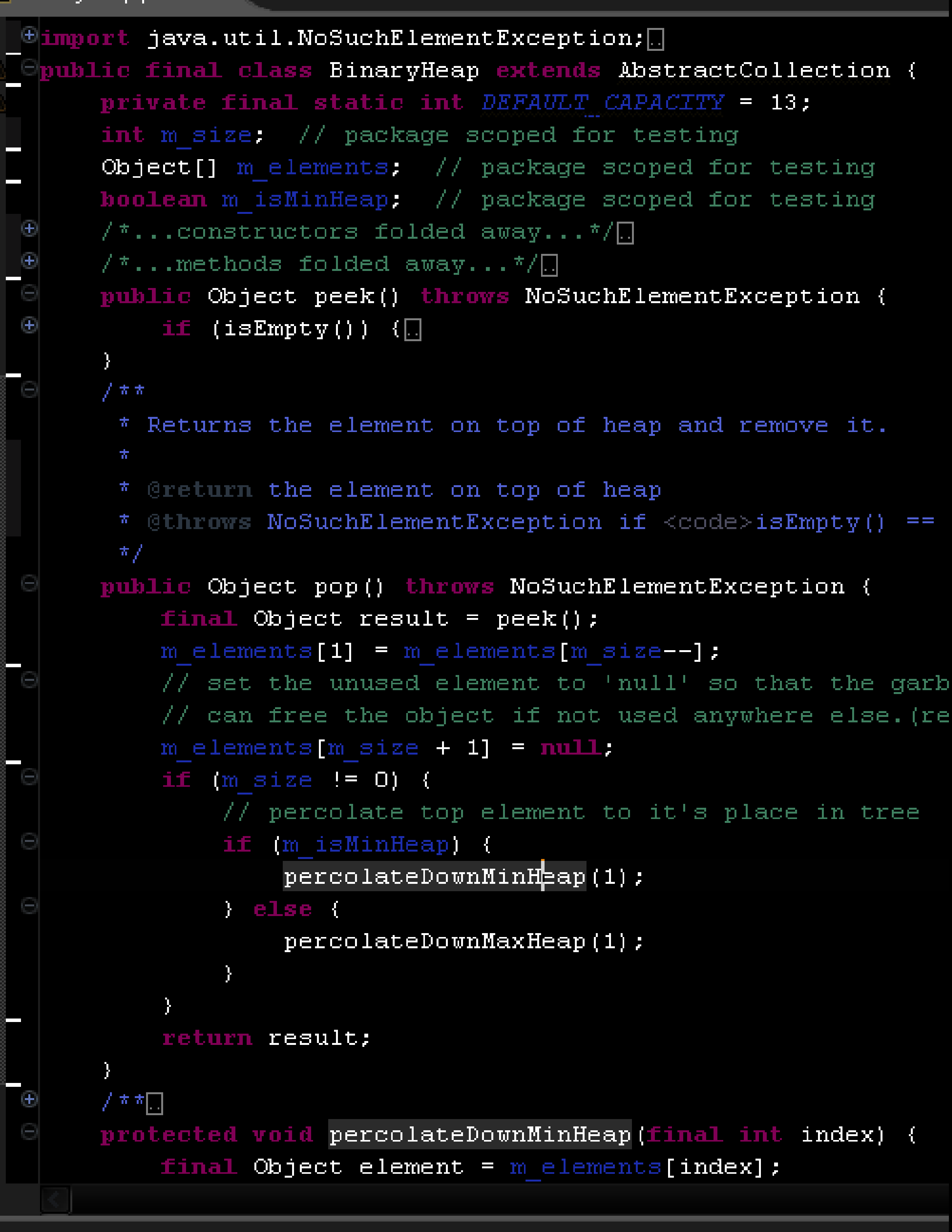
A **Fisheye view**
of code in Eclipse

... to provide better orientation in code for everyone.



```
        } else {
            return m_elements[1];
        }
    }

    /**
     * Returns the element on top of heap and remove it.
     *
     * @return the element at top of heap
     * @throws NoSuchElementException  if <code>isEmpty()
     */
    public Object pop() throws NoSuchElementException {
        final Object result = peek();
        m_elements[1] = m_elements[m_size--];

        // set the unused element to 'null' so that the ga
        // can free the object if not used anywhere else.(
        m_elements[m_size + 1] = null;

        if (m_size != 0) {
            // percolate top element to it's place in tree
            if (m_isMinHeap) {
                percolateDownMinHeap(1);
            } else {
                percolateDownMaxHeap(1);
            }
        }

        return result;
    }

    /**
     * Percolates element down heap from top.
     * Assume it is a maximum heap.
     *
     * @param index the index for the element
```

<< normal view



```
import java.util.NoSuchElementException;□
public final class BinaryHeap extends AbstractCollection {
    private final static int DEFAULT_CAPACITY = 13;
    int m_size;  // package scoped for testing
    Object[] m_elements;  // package scoped for testing
    boolean m_isMinHeap;  // package scoped for testing
    /*...constructors folded away...*/□
    /*...methods folded away...*/□
    public Object peek() throws NoSuchElementException {
        if (isEmpty()) {□
    }
    /**
     * Returns the element on top of heap and remove it.
     *
     * @return the element on top of heap
     * @throws NoSuchElementException if <code>isEmpty() ==
     */
    public Object pop() throws NoSuchElementException {
        final Object result = peek();
        m_elements[1] = m_elements[m_size--];
        // set the unused element to 'null' so that the garb
        // can free the object if not used anywhere else.(re
        m_elements[m_size + 1] = null;
        if (m_size != 0) {
            // percolate top element to it's place in tree
            if (m_isMinHeap) {
                percolateDownMinHeap(1);
            } else {
                percolateDownMaxHeap(1);
            }
        }
        return result;
    }
    /**□
    protected void percolateDownMinHeap(final int index) {
        final Object element = m_elements[index];
```

<< and Fisheye view

Thank you!

Contacts:

ODCSSS	http:// www.odcsss.ie
Jakub Dostál	dostal.j@gmail.com

Project download site

http://sort.ucd.ie/projects/grok/

Joe Kiniry	kiniry@acm.com
Radu Grigore	radugrigore@gmail.com
SRG	http://secure.ucd.ie
Mobius tool	http://mobius.inria.fr/

sfi
science foundation ireland
fondúireacht eolaíochta éireann

DCU

UCD
DUBLIN

AIC adaptive information cluster