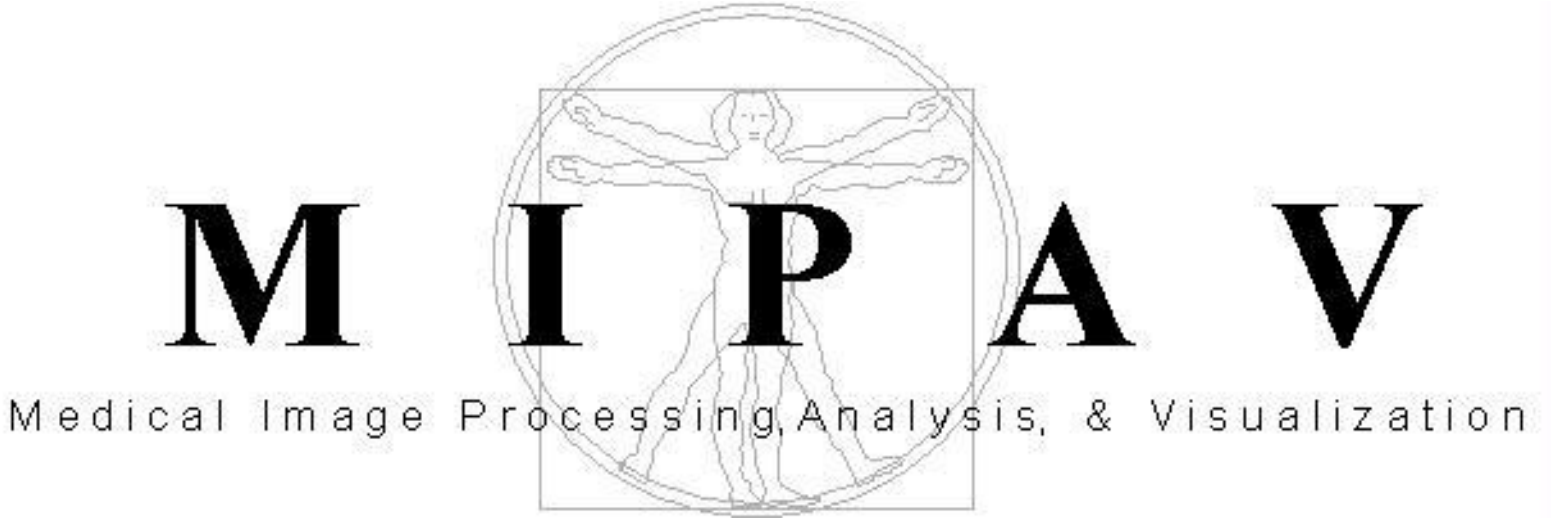
Java Fundamentals, Part 1



http://mipav.cit.nih.gov

MIPAV Team

Employees

Ruida Cheng

William Gandler

Matthew McAuliffe

Evan McCreedy

Justin Senseney

Fellows

Sara Shen

Contractors

Alexandra Bokinsky, Geometric Tools Inc. (Visualization)

Olga Vovk, SRA International Inc. (Technical Writing)

Alumni

Paul Hemler, Agatha Munzon, Nishith Pandya,

David Parsons, Beth Tyriee, Hailong Wang

Medical Image Processing, Analysis & Visualization & JAVA

Justin Senseney

SenseneyJ@mail.nih.gov

Biomedical Imaging Research Services Section (BIRSS)

Imaging Sciences Laboratory
Division of Computational Bioscience

Center for Information Technology

(301) 594-5887

http://mipav.cit.nih.gov

http://dcb.cit.nih.gov/~senseneyj

Warm-up

```
private int myMethod(int initial) {
    if(initial <= 0) {
        return 0;
    } else {
        return initial+myMethod(initial-1);
    }
}

private void init() {

    System.out.println(myMethod(1));

    System.out.println(myMethod(2));

    System.out.println(myMethod(10));

    System.out.println(myMethod(-1));
```

Review

- Creating methods
- If/Else if
- While
- For
- Switch/case

Arrays

- Store data, refer to location

```
int[] ar = new int[3];

 ar[0] = 4;
 ar[1] = 6;
 ar[2] = 7;

 System.out.println(ar.length);

 System.out.println(ar[0]);
```

Arrays

- Access by index

```
int[] ar = new int[3];

 for(int i=0; i<ar.length; i++) {
     System.out.println("The value at "+i+" location is: "+ar[i]);
 }
```

Arrays

- Can now use iterations

```
int[] ar = new int[3];

for(int i=0; i<ar.length; i++) {
    ar[i] = myMethod(i);
    System.out.println("The result is: "+ar[i]);
}
```

String methods

- Methods can return objects

```
String str = "test ";

char c = str.charAt(0);

String str2 = str.substring(0, 2);

String str3 = str.toUpperCase().trim();

System.out.println(str+c+str2+str3);
```

New Objects

```
ArrayList ar = new ArrayList();

 String str = ar.toString();

 System.out.println(ar+str);
```

New Object Methods

- Eclipse code completion shows possibilities

```
ArrayList ar = new ArrayList();

ArrayList ar2 = new ArrayList();

System.out.println(ar == ar2);

System.out.println(ar.equals(ar2));
```

Java API

Overview (Java Platform SE

docs.oracle.com/javase/6/docs/api/

Java™ Platform Standard Ed. 6

All Classes

Packages

- java.applet
- java.awt
- java.awt.color
- java.awt.datatransfer
- java.awt.dnd
- java.awt.event
- java.awt.font
- java.awt.geom

All Classes

AbstractAction
AbstractAnnotationValueVisitor6
AbstractBorder
AbstractButton
AbstractCellEditor
AbstractCollection
AbstractColorChooserPanel
AbstractDocument
AbstractDocument.AttributeContext
AbstractDocument.Content
AbstractDocument.ElementEdit
AbstractElementVisitor6
AbstractExecutorService
AbstractInterruptibleChannel
AbstractLayoutCache
AbstractLayoutCache.NodeDimens
AbstractList
AbstractListModel
AbstractMap
AbstractMap.SimpleEntry
AbstractMap.SimpleImmutableEntry
AbstractMarshallerImpl
AbstractMethodError
AbstractOwnableSynchronizer
AbstractPreferences
AbstractProcessor
AbstractQueue
AbstractQueuedLongSynchronizer
AbstractQueuedSynchronizer
AbstractScriptEngine
AbstractSelectableChannel
AbstractSelectionKey
AbstractSelector
AbstractSequentialList
AbstractSet
AbstractSpinnerModel
AbstractTableModel
AbstractTypeVisitor6
AbstractUndoableEdit

Overview Package Class Use **Tree** **Deprecated** **Index** **Help**

PREV NEXT FRAMES NO FRAMES

Java™ Platform Standard Ed. 6

Java™ Platform, Standard Edition 6
API Specification

This document is the API specification for version 6 of the Java™ Platform, Standard Edition.

See:
Description

Packages	
java.applet	Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
java.awt	Contains all of the classes for creating user interfaces and for painting graphics and images.
java.awt.color	Provides classes for color spaces.
java.awt.datatransfer	Provides interfaces and classes for transferring data between and within applications.
java.awt.dnd	Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.
java.awt.event	Provides interfaces and classes for dealing with different types of events fired by AWT components.
java.awt.font	Provides classes and interface relating to fonts.
java.awt.geom	Provides the Java 2D classes for defining and performing operations on objects related to two-dimensional geometry.
java.awt.im	Provides classes and interfaces for the input method framework.
java.awt.im.spi	Provides interfaces that enable the development of input methods that can be used with any Java runtime environment.
java.awt.image	Provides classes for creating and modifying images.
java.awt.image.renderable	Provides classes and interfaces for producing rendering-independent images.
java.awt.print	Provides classes and interfaces for a general printing API.
java.beans	Contains classes related to developing *beans* -- components based on the JavaBeans™ architecture.
java.beans.beancontext	Provides classes and interfaces relating to bean context.
java.io	Provides for system input and output through data streams, serialization and the file system.
java.lang	Provides classes that are fundamental to the design of the Java programming language.
java.lang.annotation	Provides library support for the Java programming language annotation facility.
java.lang.instrument	Provides services that allow Java programming language agents to instrument programs running on the JVM.
java.lang.management	Provides the management interface for monitoring and management of the Java virtual machine as well

Objects

Object (Java Platform SE 6)

docs.oracle.com/javase/6/docs/api/

Java™ Platform Standard Ed. 6

All Classes

Packages

java.applet
java.awt
java.awt.color
java.awt.datatransfer
java.awt.dnd
java.awt.event
java.awt.font
java.awt.geom

CMMException
Codec
CodecFactory
CodecFactoryHelper
CodecFactoryOperations
CodecOperations
CoderMalfunctionError
CoderResult
CODESET_INCOMPATIBLE
CodeSets
CodeSigner
CodeSource
CodingErrorAction
CollapsedStringAdapter
CollationElementIterator
CollationKey
Collator
CollatorProvider
Collection
CollectionCertStoreParameters
Collections
Color
ColorChooserComponentFactory
ColorChooserUI
ColorConvertOp
ColorModel
ColorSelectionModel
ColorSpace
ColorSupported
ColorType

Overview Package Class Use Tree Deprecated Index Help

Java™ Platform Standard Ed. 6

PREV CLASS NEXT CLASS FRAMES NO FRAMES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

java.lang

Class Object

`java.lang.Object`

`public class Object`

Class `Object` is the root of the class hierarchy. Every class has `Object` as a superclass. All objects, including arrays, implement the methods of this class.

Since:

JDK1.0

See Also:

`Class`

Constructor Summary

`Object()`

Method Summary

protected Object	`clone()` Creates and returns a copy of this object.
boolean	`equals(Object obj)` Indicates whether some other object is "equal to" this one.
protected void	`finalize()` Called by the garbage collector on an object when garbage collection determines that there are no more references to the object.
Class<?>	`getClass()` Returns the runtime class of this `Object`.
int	`hashCode()` Returns a hash code value for the object.

Collections

- Interface – defines method signature

Collection (Java Platform S ×

docs.oracle.com/javase/6/docs/api/

Java™ Platform Standard Ed. 6

All Classes

Packages

java.applet
java.awt
java.awt.color
java.awt.datatransfer
java.awt.dnd
java.awt.event
java.awt.font
java.awt.geom

CMMException
Codec
CodecFactory
CodecFactoryHelper
CodecFactoryOperations
CodecOperations
CoderMalfunctionError
CoderResult
CODESET_INCOMPATIBLE
CodeSets
CodeSigner
CodeSource
CodingErrorAction
CollapsedStringAdapter
CollationElementIterator
CollationKey
Collator
CollatorProvider
Collection
CollectionCertStoreParameters
Collections

Overview Package Class Use Tree Deprecated Index Help

Java™ Platform Standard Ed. 6

PREV CLASS NEXT CLASS FRAMES NO FRAMES
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

java.util

Interface Collection<E>

All Superinterfaces:

Iterable<E>

All Known Subinterfaces:

BeanContext, BeanContextServices, BlockingDeque<E>, BlockingQueue<E>, Deque<E>, List<E>, NavigableSet<E>, Queue<E>, Set<E>, SortedSet<E>

All Known Implementing Classes:

AbstractCollection, AbstractList, AbstractQueue, AbstractSequentialList, AbstractSet, ArrayBlockingQueue, ArrayDeque, ArrayList, AttributeList, BeanContextServicesSupport, BeanContextSupport, ConcurrentLinkedQueue, ConcurrentSkipListSet, CopyOnWriteArrayList, CopyOnWriteArraySet, DelayQueue, EnumSet, HashSet, JobStateReasons, LinkedBlockingDeque, LinkedBlockingQueue, LinkedHashSet, LinkedList, PriorityBlockingQueue, PriorityQueue, RoleList, RoleUnresolvedList, Stack, SynchronousQueue, TreeSet, Vector

```
public interface Collection<E>
extends Iterable<E>
```

The root interface in the *collection hierarchy*. A collection represents a group of objects, known as its *elements*. Some collections allow duplicate elements and others do not. Some are ordered and others unordered. The JDK does not provide any *direct* implementations of this interface: it provides implementations of more specific subinterfaces like `Set` and `List`. This interface is typically used to pass collections around and manipulate them where maximum generality is desired.

Bags or *multisets* (unordered collections that may contain duplicate elements) should implement this interface directly.

Writers/Readers

```
try {
    BufferedWriter write = new BufferedWriter(new FileWriter(“tR.txt"));

    write.write("myText");
    write.newLine();
    write.flush();
    write.close();

    BufferedReader read = new BufferedReader(new FileReader("tW.txt"));

    read.readLine();
    read.close();

} catch (IOException e1) {
    e1.printStackTrace();
}
```

JComponents

- JTextField
- JRadioButton
- JPanel
- JLabel
- All are contained within a JFrame

Listeners

- Java handling of events

```
JCheckBox box1 = new JCheckBox("Process?");

box1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        //action based on checking box
    }
});
```

Quiz 1 - Evaluate

```
int[] ar = new int[10];

 for(int i=0; i<ar.length; i++) {
     ar[i] = i/2;
 }
```

Quiz 2 - Describe

```
final long time = System.currentTimeMillis();

 final ArrayList ar = new ArrayList();

 JCheckBox box1 = new JCheckBox("Process?");

 box1.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent arg0) {
         ar.add(System.currentTimeMillis());
         System.out.println("Since time: "+
               (Long.valueOf(ar.get(ar.size()-1).toString()) - time));
     }
 });
```

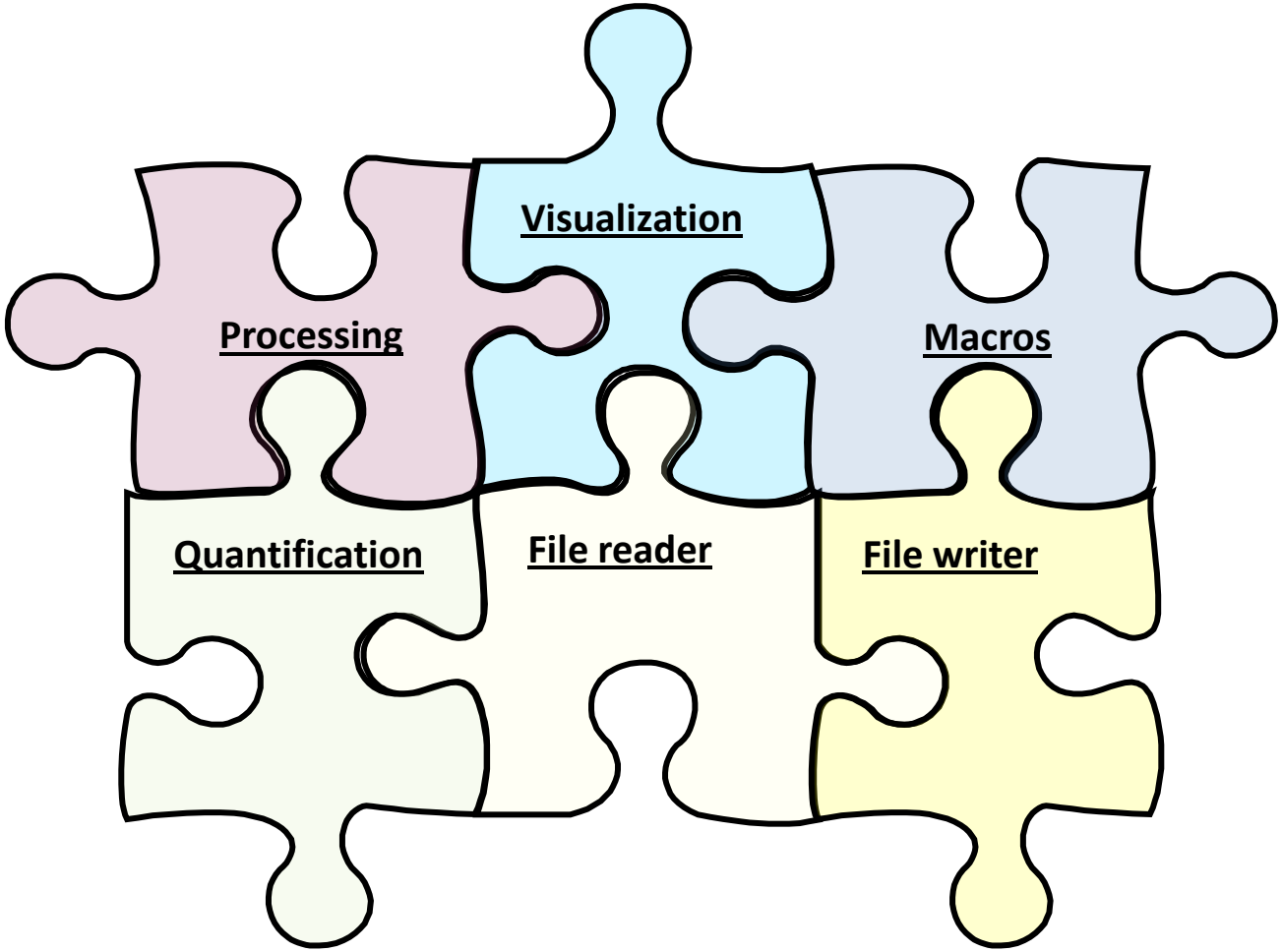
Task - Write

- Method to write the contents of an array to a file
- Method to convert an array to an ArrayList

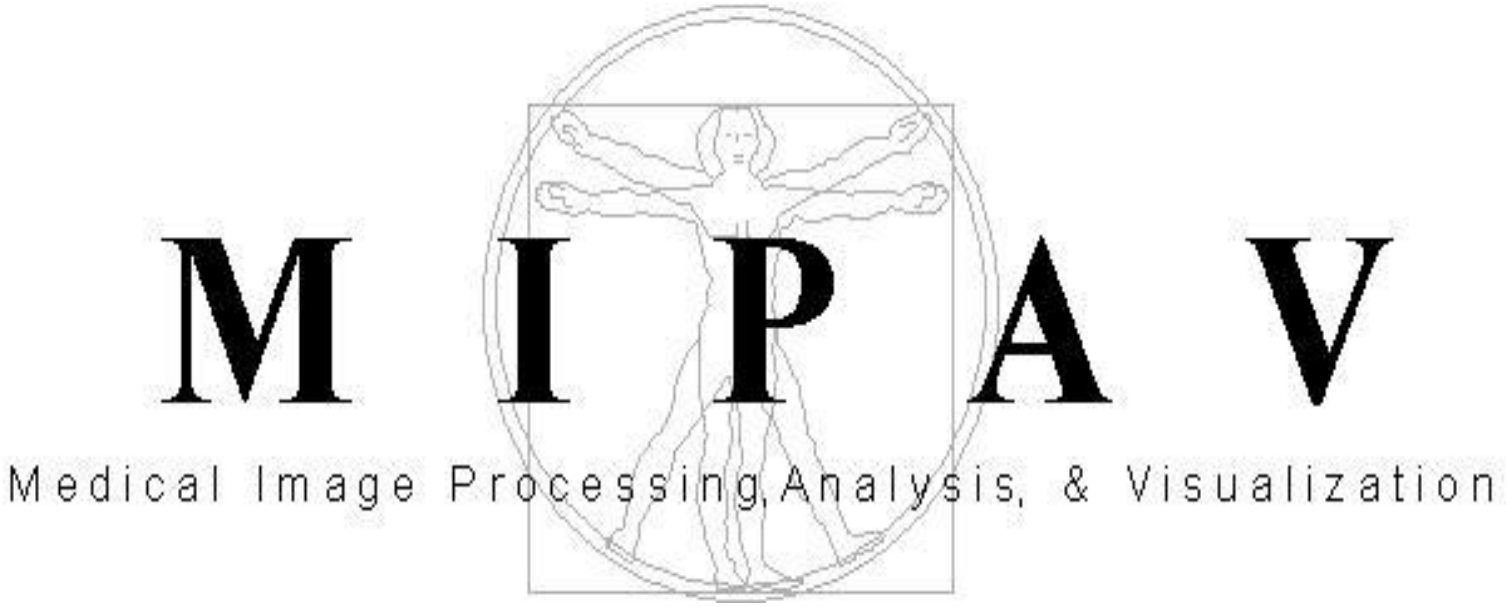
Summary

- Arrays
- Methods
- Objects
- API

MIPAV


Visualization
Processing
Macros
Quantification
File reader
File writer

http://mipav.cit.nih.gov



SenseneyJ@mail.nih.gov