

> Lesson Summary and Outcomes

Students will be able to use `ArrayLists` for adding and removing values. In this lesson, students will learn to cast all elements from the generic `Object` type to a more specific class type. The `ArrayList` class—a much improved version of arrays—takes care of all of these problems. Students will receive a handout that includes a table of commonly used `ArrayList` methods that describes their function and gives an example of their usage.

> Student Assessment

How well do students perform each of the following tasks:

- Add the `showBooksOut ()` method to list books checked out and use it in the `main ()`
- Add an option and a method to check books in
- Add an option and a method to add new books to the `onShelf` `arraylist`
- Add a loop so that the user can continue checking books in and out
- Comment everything
- Save, compile, and test often

> Prerequisite Knowledge and Skills

BlueJ:

- Creating projects
- Adding new classes
- Instantiating objects
- Testing methods from instantiated objects

Java:

- Class structure, including methods
- `ArrayList` class and its methods
- External packages

> This Lesson Targets the Following Subject Areas(s):

Pre-Java	Java Programming
<input type="checkbox"/> Hardware Basics	<input type="checkbox"/> Applet Programming
<input type="checkbox"/> Software Basics	<input type="checkbox"/> Subroutine Programming
<input type="checkbox"/> Networks and Servers	<input checked="" type="checkbox"/> Full Scale Programming
<input type="checkbox"/> HTML	
<input type="checkbox"/> Action Scripting	
<input type="checkbox"/> Java Scripting	

> Curriculum-Framing Questions

Essential Question	<ul style="list-style-type: none"> • What is an <code>ArrayList</code>?
Lesson Questions	<ul style="list-style-type: none"> • How does an <code>ArrayList</code> differ from an array? • What other methods are available? (Hint: Check the online Java API [application program interface].) • What data members does an <code>ArrayList</code> have?
Sample Content Questions	<ul style="list-style-type: none"> • How do you find items in a list? • Can you add items to the <code>ArrayList</code> once it has been defined?

> Targeted Content Standards, Benchmarks, or State Frameworks

Denver Public Schools (DPS) Standards:

Secondary DPS Information Literacy and Technology Proficiencies:

- **Student as an Efficient Information and Technology User:**
 - 2 – Location skills library and Internet

Introduction to organization of High School

 - 4 – Library catalog (LION)/Dewey Decimal

Create and export lists and bibliographies from LION

International Society of Technology Education (ISTE):

- **Technology Foundation Standards for Students:**
 - **Standard 3** – Students use technology tools to enhance learning, increase productivity, and promote creativity. Students use productivity tools to collaborate in constructing technology-enhanced models, prepare publications, and produce other creative works.
 - **Standard 5** – Students use technology to locate, evaluate, and collect information from a variety of sources. Students

use technology tools to process data and report results. Students evaluate and select new information resources and technological innovations based on the appropriateness for specific tasks.

> Materials and Resources Required for Lesson

Equipment	<ul style="list-style-type: none"> • Computer for each student with BlueJ and Java SDK installed
Textbooks/Lesson Guides	<ul style="list-style-type: none"> • Handouts of lessons
Technology	<ul style="list-style-type: none"> • Windows-based machines running Windows 98 SE or higher
Internet Resources	<ul style="list-style-type: none"> • http://java.sun.com/j2se/1.5.0/docs/api/

> Lesson Plan Outline

Overview: In this lesson, students will create a simple library. To keep track of the books, they will employ `ArrayLists` for storing books checked in and out. Students will flesh out the specified structure, perhaps by using some of the other `ArrayList` methods.

To begin, create a new project in BlueJ named Library, and within that project create a class also called Library.

First, enter the code as illustrated in the handout. This code provides you with a basic starting point. You will need to make additions and modifications to it. A slightly more robust version of the code is provided in the addendum.

Next, add the external package `chn.util` using the instructions as follows:

1. Create a directory specifically for packages and classes. For example, you might design a directory hierarchy that looks like `\java\utilities`.
2. Extend that hierarchy to reflect your new package. For instance, if your package is named `files.readwrite`, then your hierarchy should be extended to read `\java\utilities\files\readwrite`.
3. Store your new package in that directory, making sure your package is named `files.readwrite`.
4. Set your CLASSPATH environmental variable to include the path `\java\utilities`.
5. Make sure that all of the classes in your package begin with the designation `package files.readwrite`. Nothing should appear in the file above that line.
6. To include the new package, add to your program the import directive `import files.readwrite.*;`

To use packages in BlueJ, take the following steps:

1. Follow Steps 1–6 above
2. From the BlueJ menu, select Tools→Preferences
3. Select the Libraries tab
4. Click on the Add button on the right-hand side of the dialog box
5. Navigate to the `.jar` or `.zip` file containing your package and select that file
6. Click OK to confirm that addition and return to BlueJ
7. Save your project and exit BlueJ so that the new `library / package` can load

8. Return to `BlueJ`
9. Select `Tools`→`Preferences` followed by the `Libraries` tab to verify that your `package / library` has been loaded

Next, make additions and modifications to the code.

Tips, Hints and Tricks: Get creative and make the program better. For example, you could display a count of books on each list.

> Pacing/Timeline

Code and compile Library class: 1–2 days

- Add extra functionality: 3–4 days
- Test thoroughly: 1 day

> Teacher Reflection (For example, what worked well in this lesson? What would you change if you were to teach it again?)

- Did students understand `ArrayLists` or did they need more help?
- Were students able to add external packages?
- What roadblocks did students encounter in implementing this project?
- Did students employ additional `ArrayList` methods?

> Sample Code

```

/**
 * Library.java demonstrates use of ArrayLists
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.util.*;
import chn.util.*;
public class Library
{
    // instance variables: arraylists for books checked in
    // and out
    private ArrayList onShelf;
    private ArrayList onLoan;
    /**
     * Constructor for objects of class Library
     */
    public Library()
    {
        onLoan = new ArrayList(); // initialize onLoan arraylist
        onShelf = new ArrayList(); // initialize onShelf
                                   // arraylist
        onShelf.add("Green Eggs and Spam"); // add some books
        onShelf.add("Spam Wonderful Spam");
    }
    /**
     * isCheckedOut() returns true if book is onLoan, false
     * otherwise
     */
    public boolean isCheckedOut(String title)
    {
        if (onLoan.contains(title)) // check the arraylist for
                                   // title
            return true; // if it exists, the test is true
        else
            return false; // if not, test is false
    } // end of isCheckedOut()
    /**
     * Move book to onLoan list
     */
    public void checkOut(String title)
    {
        if (isCheckedOut(title)) // see if already checked out
            System.out.println("We're sorry, that book is not
                               available.\n");
        else // not checked out
        {

```

```

        int index = onShelf.indexOf(title); // get its
            // index
        String temp = (String)onShelf.get(index); // save
            // the name
        onShelf.remove(index); // remove it from the
            // onShelf list
        onLoan.add(temp); // add it to the onLoan list
    }
} // end of checkout()
/**
 * Move book to onShelf list
 */
public void checkIn(String title)
{
    if (isCheckedOut(title)) // see if already checked out
    {
        int index = onLoan.indexOf(title); // get its
            // index
        String temp = (String)onLoan.get(index); // save
            // the name
        onLoan.remove(index); // remove it from the
            // onShelf list
        onShelf.add(temp); // add it to the onLoan list
    }
} // end of checkIn()
/**
 * Add new Book to onShelf list
 */
public void addBook(String title)
{
    onShelf.add(title); // add it to the onLoan list
} // end of addBook()
/**
 * Lists all books checked in
 */
public void showBooksIn()
{
    String temp;
    System.out.println("Spamville Library\nBooks checked
        in\n-----\n");
    Iterator iter = onShelf.iterator(); // create an
        // iterator object
    while (iter.hasNext()) // and keep going so long as
        // there are more elements in the list
    {
        temp = (String)iter.next(); // get & convert
            // current element
        System.out.println(temp); // print it!
    }
} // end of showBooksIn()

```

```

/**
 * Lists all books checked out
 */
public void showBooksOut()

{
    String temp;
    System.out.println("Spamville Library\nBooks checked
        out\n-----\n");
    Iterator iter = onLoan.iterator(); // create an
        // iterator object
    while (iter.hasNext()) // and keep going so long as
        // there are more elements in the list
    {
        temp = (String)iter.next(); // get & convert
            // current element
        System.out.println(temp); // print it!
    } // end while loop
} // end method ShowBooksOut
/**
 * Main
 */
public static void main(String[] args)
{
    Library lib = new Library(); // create the library
        // object
    chn.util.ConsoleIO console = new chn.util.ConsoleIO(); //
    enable
    // keyboard input
    String input;
    lib.showBooksIn(); // list books checked in
    do
    {
        System.out.println("\n\nEnter 'out' to check out,
            'in' to check in, 'add' to add new book or
            'q' to quit:");
        input = console.readLine(); // get user's choice
        if (input.equals("out")) // user wants to check
            // one out
        {
            lib.showBooksIn(); // list books checked in
            System.out.print("Enter the title to check
                out: ");
            input = console.readLine(); // get the
                // title
            lib.checkOut(input); // check it out using
                // the checkout method
        }
        else if (input.equals("in")) // user wants to
            // check in

```

```

    {
        lib.showBooksOut(); // list books checked
        // out
        System.out.print("Enter the title to check
            out: ");
        input = console.readLine(); // get the
        // title
        lib.checkIn(input); // check it out using
        // the checkIn method
    }
    else if (input.equals("add"))
    {
        System.out.print("Enter the title to add:
            ");
        input = console.readLine(); // get the
        // title
        lib.addBook(input); // check it out using
        // the addBook method
    }
    }
    while (!(input.equals("q")));
} // end of main()
} // end of Library class

```

> Student Handout: Java Programming ArrayLists—A Library Example

Arrays are very handy, but they do have limitations such as their fixed size. You may also not be able to add or remove values, or you may have to force all elements to be the same type. The `ArrayList` class has resolved all of these problems. Here are most commonly used methods of this class.

This <code>ArrayList</code> method	Fills this need	And looks like this
<code>int size();</code>	Returns number of elements	<code>int len = myList.size();</code>
<code>boolean add(obj);</code> <code>void add(int index, obj);</code>	Adds <code>obj</code> to the end of the <code>ArrayList</code> or at the specified index	<code>myList.add("some object");</code> <code>myList.add(4, "another object");</code> <code>// both add a String object</code>
<code>Object remove(int index)</code>	Returns and removes the object at the given index	<code>myList.remove(3);</code>
<code>Object get(int index)</code>	Returns the object at the given index	<code>Object obj = myList.get(4);</code>

<code>int indexOf(obj);</code>	Searches for the first obj in the list and returns its index	<code>int index = myList.indexOf("Hubert");</code>
<code>boolean contains(obj)</code>	Returns true if the list contains obj; otherwise returns false	<code>boolean found = myList.contains("Boo"); // looks for a String object "Boo"</code>

Let's model a simple library where users can check books in and out, and where new books can be added. Create a new project in BlueJ named Library and create a class also called Library. Enter the following code:

```
/**
 * Library.java demonstrates use of ArrayLists
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.util.*;
import chn.util.*;
public class Library
{
    // instance variables: arraylists for books checked in and out
    private ArrayList onShelf;
    private ArrayList onLoan;
    /**
     * Constructor for objects of class Library
     */
    public Library()
    {
        onLoan = new ArrayList(); // initialize onLoan
                                   // arraylist
        onShelf = new ArrayList(); // initialize onShelf
                                   // arraylist
        onShelf.add("Green Eggs and Spam"); // add some books
        onShelf.add("Spam Wonderful Spam");
    }
    /**
     * isCheckedOut() returns true if book is onLoan, false
     * otherwise
     */
    public boolean isCheckedOut(String title)
    {
        if (onLoan.contains(title)) // check the arraylist for
                                   // title
            return true; // if it exists, the test is true
        else
            return false; // if not, test is false
    } // end of isCheckedOut()
    /**
```

```

* Move book to onLoan list
*/
public void checkOut(String title)
{
    if (isCheckedOut(title)) // see if already checked out
        System.out.println("We're sorry, that book is not
            available.\n");
    else // not checked out
    {
        int index = onShelf.indexOf(title); // get its
            // index
        String temp = (String)onShelf.get(index); // save
            // the name
        onShelf.remove(index); // remove it from the
            // onShelf list
        onLoan.add(temp); // add it to the onLoan list
    }
} // end of checkout()
/**
* Move book to onShelf list
*/
public void checkIn(String title)
{
    if (isCheckedOut(title)) // see if already checked out
    {
        int index = onLoan.indexOf(title); // get its
            // index
        String temp = (String)onLoan.get(index); // save
            // the name
        onLoan.remove(index); // remove it from the
            // onShelf list
        onShelf.add(temp); // add it to the onLoan list
    }
} // end of checkIn()
/**
* Add new Book to onShelf list
*/
public void addBook(String title)
{
    onShelf.add(title); // add it to the onLoan list
} // end of addBook()
/**
* Lists all books checked in
*/
public void showBooksIn()
{
    String temp;
    System.out.println("Spamville Library\nBooks checked
in\n-----
-----\n");
}

```

```

        Iterator iter = onShelf.iterator(); // create an
                                   // iterator object
        while (iter.hasNext()) // and keep going so long as
                                   //there are more elements in the list
        {
            temp = (String)iter.next(); // get & convert
                                   // current element
            System.out.println(temp); // print it!
        }
    } // end of showBooksIn()
    /**
    * Lists all books checked out
    */
    public void showBooksOut()
    {
        String temp;
        System.out.println("Spamville Library\nBooks checked
            out\n-----\n");
        Iterator iter = onLoan.iterator(); // create an
                                   // iterator object
        while (iter.hasNext()) // and keep going so long as
                                   // there are more elements in the list
        {
            temp = (String)iter.next(); // get & convert
                                   // current element
            System.out.println(temp); // print it!
        } // end while loop
    } // end method ShowBooksOut
    /**
    * Main
    */
    public static void main(String[] args)
    {
        Library lib = new Library(); // create the library
                                   // object
        chn.util.ConsoleIO console = new chn.util.ConsoleIO(); //
        enable keyboard input
        String input;
        lib.showBooksIn(); // list books checked in
        do
        {
            System.out.println("\n\nEnter 'out' to check out,
                'in' to check in, 'add' to add new book or
                'q' to quit:");
            input = console.readLine(); // get user's choice
            if (input.equals("out")) // user wants to check
                                   // one out
            {
                lib.showBooksIn(); // list books checked in
                System.out.print("Enter the title to check

```

```
        out: ");
        input = console.readLine(); // get the
            // title
        lib.checkOut(input); // check it out using
            // the checkout method
    }
    else if (input.equals("in")) // user wants to
        // check in

    {
        lib.showBooksOut(); // list books checked
            // out
        System.out.print("Enter the title to check
            out: ");
        input = console.readLine(); // get the
            // title
        lib.checkIn(input); // check it out using
            // the checkIn method
    }
    else if (input.equals("add"))
    {
        System.out.print("Enter the title to add:
            ");
        input = console.readLine(); // get the
            // title
        lib.addBook(input); // check it out using
            // the addBook method
    }
    }
    while (!(input.equals("q")));
} // end of main()
} // end of Library class
```

> YOUR JOB:

- Add the `showBooksOut()` method to list books checked out and use it in `main()`
- Add an option and a method to check books in
- Add an option and a method to add new books to the `onShelf ArrayList`
- Add a loop so that the user can continue checking books in and out
- Comment everything
- Save, compile, and test OFTEN!

> EXTRAS YOU CAN ADD:

- Get creative and make the program better. Display a count of the number of books on each list.
- Add a method to automatically check in all books currently checked out.