The Big Picture

- partially full and dynamic arrays come up a lot
- the lazy programmer might want a DynamicArray class to avoid having to write add, remove, grow, etc code over and over…

Java provides `ArrayList` for this purpose.

When to Use What?

- arrays – built-in language feature
  - regular – fixed capacity, always full
  - partially full – fixed capacity, not necessarily full
  - dynamic – grow/shrink, not necessarily full
- `ArrayList` – Java class implementation of partially full dynamic arrays

---

- typical usage
  - arrays for fixed capacity always full situations
  - `ArrayList` for everything else
- so why study partially full and dynamic arrays?
  - many languages provide arrays, but not all have something like `ArrayList`
  - you may need to figure out your own array manipulations
  - understanding of how `ArrayList` works is important for making efficient data structure choices

Generics

- the only difference between a dynamic array of `int`s and a dynamic array of `double`s (or `boolean`s or `char`s or `String`s or `Dice` or …) is what you write for the base type
- Java supports *parameterized classes* (or *generics*) where a type can be provided as a parameter
  - how to recognize that a class is a generic
    - you see something like <E> in the class name in the API
  - how to use a generic class
    - replace the type parameter with the name of the actual type

Module java.base
Package java.util

Class ArrayList<E>

java.lang.Object
  java.util.AbstractCollection<E>
    java.util.AbstractList<E>
      java.util.ArrayList<E>

Type Parameters:

E - the type of elements in this list

Wrapper Classes and Autoboxing

- only class types can be used for a type parameter
- Java provides *wrapper classes* `Integer`, `Double`, `Boolean`, `Character`, etc for the primitive types
- the system will automatically convert between an object of a wrapper class type and a value of the corresponding primitive type (*autoboxing*)

Using `ArrayList`

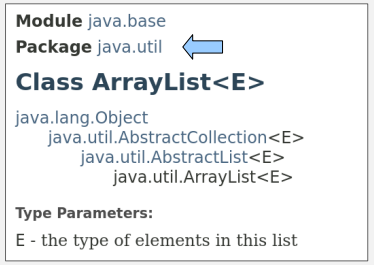
```
// declare variable – ArrayList of integers
                     numbers;

// create a new ArrayList object
numbers = new                      ;

// add 42 to the list
numbers.        ;

// retrieve the first number in the list,
//   storing it in a variable
int x =                  ;
```


Using `ArrayList`

```
// declare variable – ArrayList of integers
ArrayList<Integer> numbers;

// create a new ArrayList object
numbers = new ArrayList<Integer>();

// add 42 to the list
numbers.add(42);

// retrieve the first number in the list,
//   storing it in a variable
int x = numbers.get(0);
```


Packages and Imports

- *packages* divide up the Java namespace
  - the full name of a class in package `pkg` is `pkg.ClassName`
- *import* a class to be able to write just `ClassName` instead of `pkg.ClassName`
  - `import pkg.ClassName;` at the beginning of the file (before `public class`)
  - can `import pkg.*;` to import all class names in `pkg`
- package information can be found in the API documentation


Module java.base
Package java.util
Class ArrayList<E>
java.lang.Object
java.util.AbstractCollection<E>
java.util.AbstractList<E>
java.util.ArrayList<E>
Type Parameters:
E - the type of elements in this list