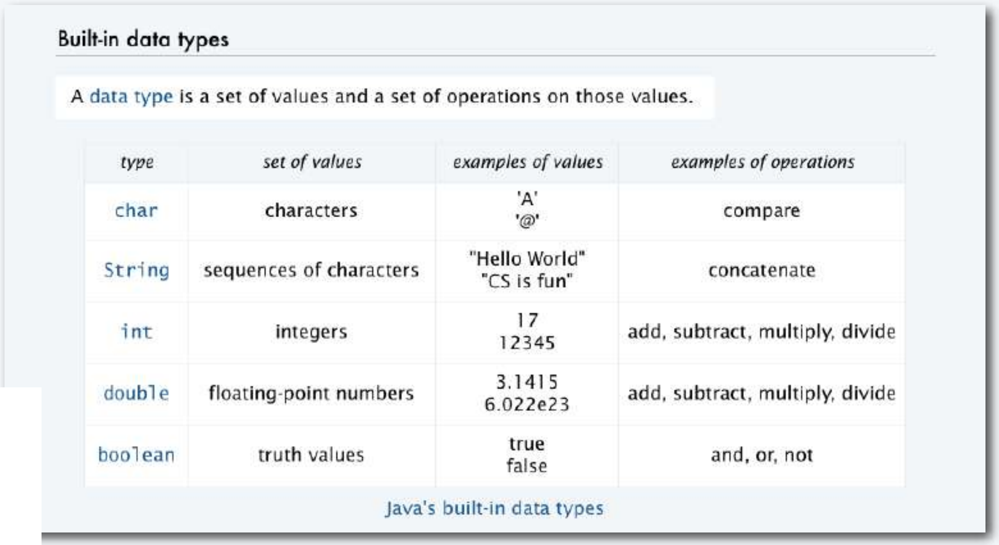
Abstract data types

A data type is a set of values and a set of operations on those values.

Primitive types

- *values* immediately map to machine representations
- *operations* immediately map to machine instructions.



Built-in data types

A data type is a set of values and a set of operations on those values.

type	*set of values*	*examples of values*	*examples of operations*
`char`	characters	'A' '@'	compare
`String`	sequences of characters	"Hello World" "CS is fun"	concatenate
`int`	integers	17 12345	add, subtract, multiply, divide
`double`	floating-point numbers	3.1415 6.022e23	add, subtract, multiply, divide
`boolean`	truth values	true false	and, or, not

Java's built-in data types

We want to write programs that process other types of data.

- Colors, pictures, strings,
- Complex numbers, vectors, matrices,
- ...

An abstract data type is a data type whose representation is hidden from the client.

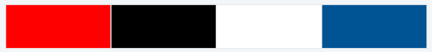
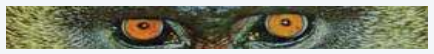
Object-oriented programming (OOP)

Object-oriented programming (OOP).

- Create your own data types.
- Use them in your programs (manipulate *objects*).

An object holds a data type value.
Variable names refer to objects.

Examples (stay tuned for details)

data type	*set of values*	*examples of operations*	
`Color`	three 8-bit integers	get red component, brighten	
`Picture`	2D array of colors	get/set color of pixel (i, j)	
`String`	sequence of characters	length, substring, compare	C A T A G C G C

Best practice: Use *abstract* data types (representation is *hidden from the client*).

Impact: Clients can use ADTs without knowing implementation details.

- This lecture: how to write client programs for several useful ADTs
- Next lecture: how to implement your own ADTs

Strings

We have *already* been using ADTs!

A `String` is a sequence of Unicode characters. ← defined in terms of its ADT values (typical)

Java's String ADT allows us to write Java programs that manipulate strings.
The exact representation is hidden (it could change and our programs would still work).

stay tuned for more complete API later in this lecture

Operations (API)

`public class String`	
`String(String s)`	*create a string with the same value*
`int length()`	*string length*
`char charAt(int i)`	*ith character*
`String substring(int i, int j)`	*ith through (j-1)st characters*
`boolean contains(String sub)`	*does string contain* `sub`*?*

Using a data type: constructors and methods

To use a data type, you need to know:

- Its name (capitalized, in Java).
- How to *construct* new objects.
- How to *apply operations* to a given object.

To construct a new object

- Use the keyword new to invoke a *constructor*.
- Use data type name to specify type of object.

To apply an operation (invoke a method)

- Use object name to specify which object.
- Use the dot operator to indicate that an operation is to be applied.
- Use a method name to specify which operation.

`new Building()`

```
String s;
s = new String ("Hello, World");
StdOut.println( s.substring(0, 5) );
```

Pop quiz on ADTs

Q. What is a data type?

A. A set of values and a set of operations on those values.

Q. What is an abstract data type?

A. A data type whose representation is hidden from the client.