Using Objects

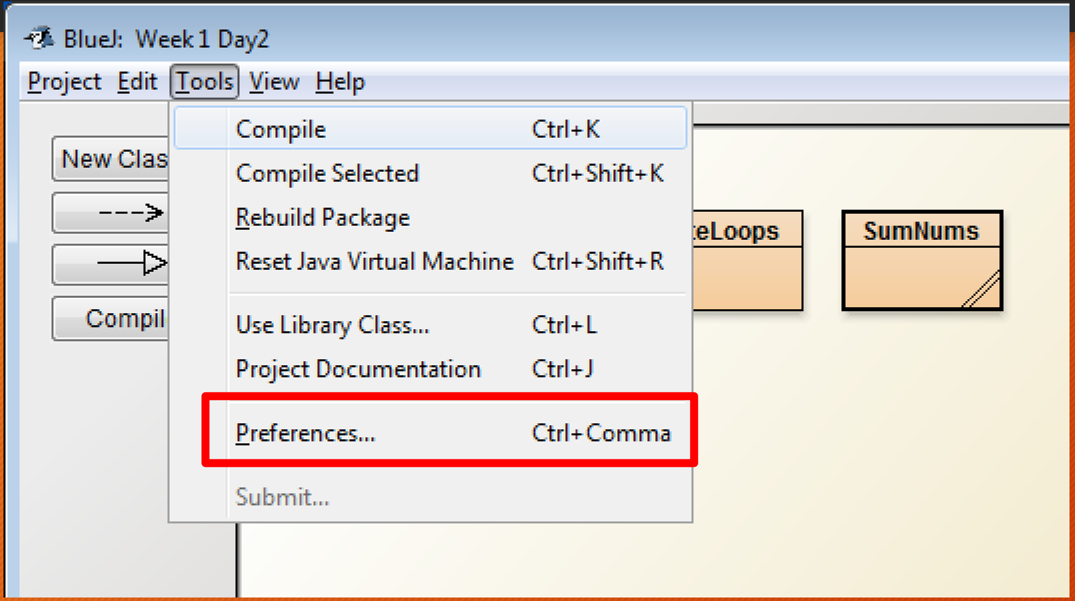
CSC142, Week 2a

Textbook: Chapter 3.3–3.4

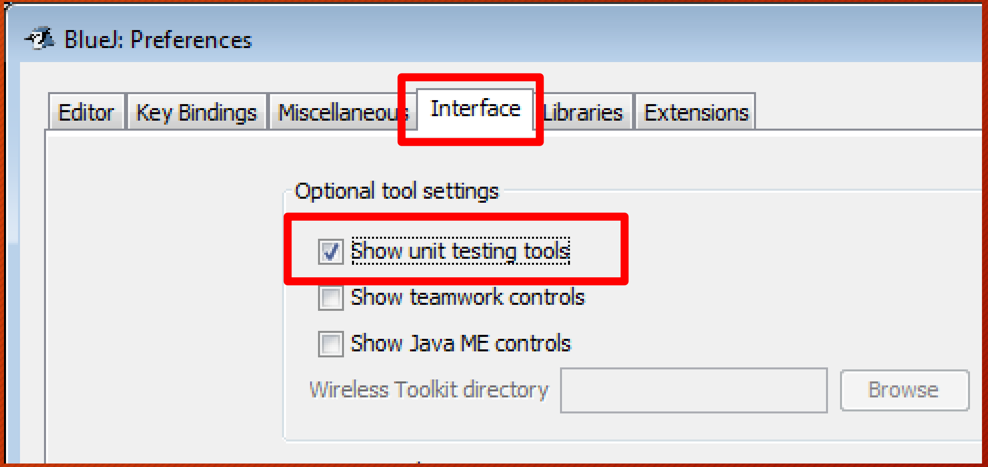
Agenda

- Topics
  - Introduce the JUnit unit test framework
  - Present a quick overview of classes and objects, methods and constants
  - Discuss how to use the String data type and key string concepts
  - Discuss object constructors, reference types, and object diagrams
  - Present how to create and use Scanner objects for user input
- Activities:
  - Create a test class for SumNums, then write test cases in it
  - Use object methods and constants
  - Gather use input using a Scanner object
  - If time: write an interactive program using string manipulation

BlueJ: Turn on JUnit UI



Compile Ctrl+K
Compile Selected Ctrl+Shift+K
Rebuild Package
Reset Java Virtual Machine Ctrl+Shift+R
Use Library Class... Ctrl+L
Project Documentation Ctrl+J
Preferences... Ctrl+Comma
Submit...

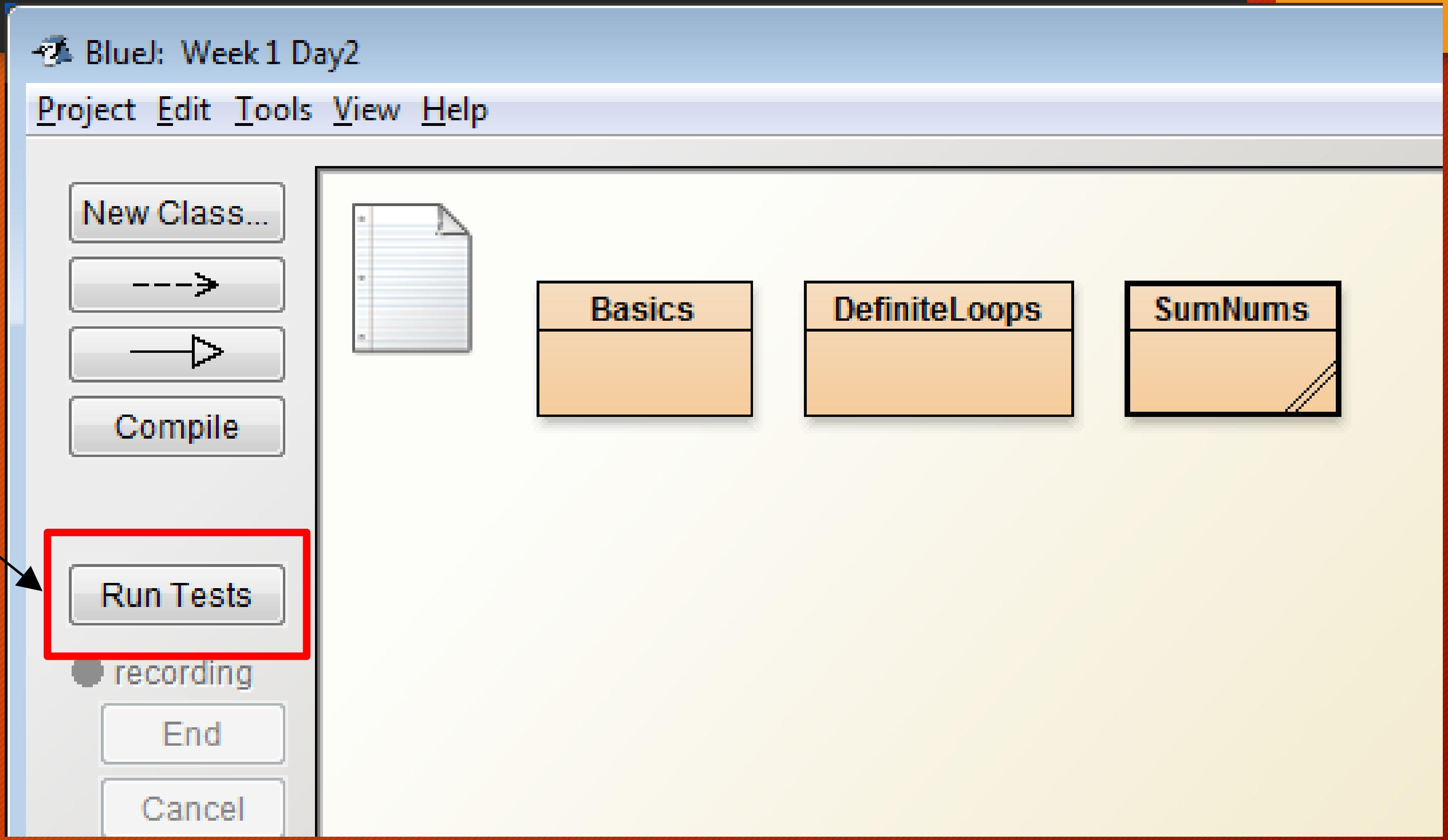


Optional tool settings

- [✓] Show unit testing tools
- [ ] Show teamwork controls
- [ ] Show Java ME controls

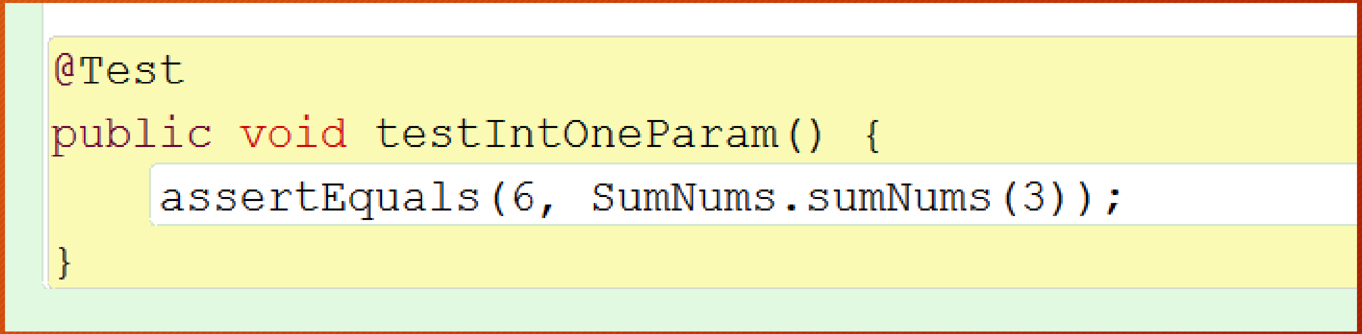
BlueJ: Test UI

Runs all compiled tests (all test classes and methods)



Anatomy of a JUnit Test Class

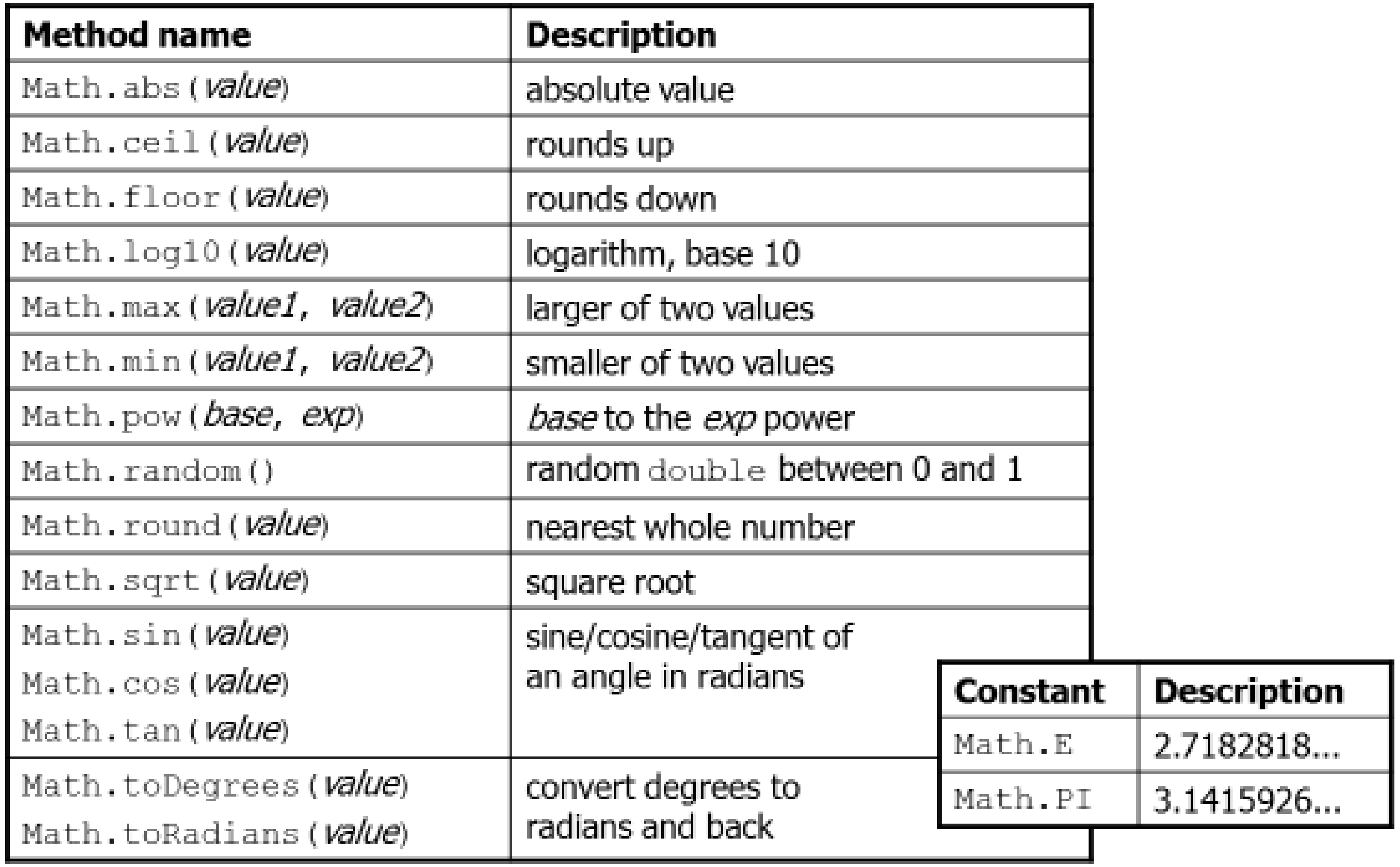
- At the top, imports for necessary classes
- A section for code to run **@Before** each test
- A section for code to run **@After** each test
- A series of **@Test** methods, code that represents one or a series of related tests
  - Each test should run a verification, usually by calling assertEquals:



```
@Test
public void testIntOneParam() {
    assertEquals(6, SumNums.sumNums(3));
}
```

Java's Math Class

We'll use both the **methods** and the **constants** shown here



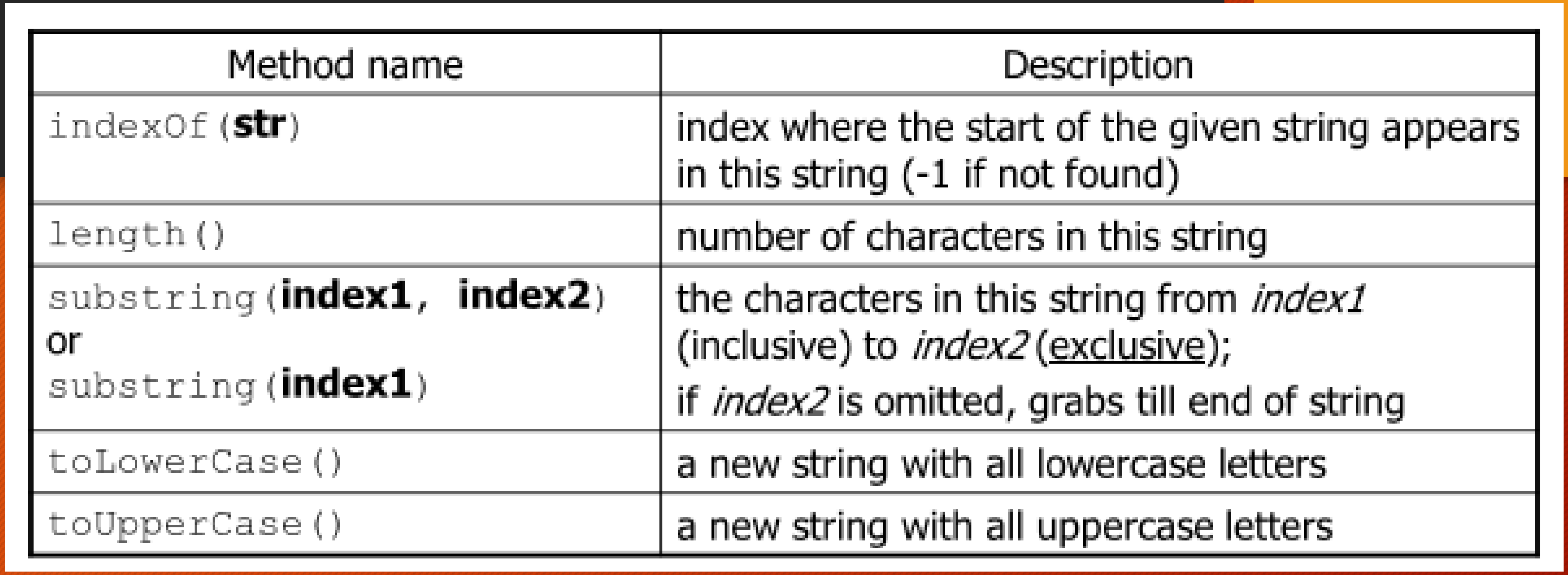
Method name	Description
`Math.abs(`*value*`)`	absolute value
`Math.ceil(`*value*`)`	rounds up
`Math.floor(`*value*`)`	rounds down
`Math.log10(`*value*`)`	logarithm, base 10
`Math.max(`*value1*`,` *value2*`)`	larger of two values
`Math.min(`*value1*`,` *value2*`)`	smaller of two values
`Math.pow(`*base*`,` *exp*`)`	*base* to the *exp* power
`Math.random()`	random `double` between 0 and 1
`Math.round(`*value*`)`	nearest whole number
`Math.sqrt(`*value*`)`	square root

`Math.sin(`*value*`)`
`Math.cos(`*value*`)`
`Math.tan(`*value*`)`
	sine/cosine/tangent of an angle in radians

`Math.toDegrees(`*value*`)`
`Math.toRadians(`*value*`)`
	convert degrees to radians and back

Constant	Description
`Math.E`	2.7182818...
`Math.PI`	3.1415926...

String Methods



Method name	Description
`indexOf(`**str**`)`	index where the start of the given string appears in this string (-1 if not found)
`length()`	number of characters in this string
`substring(`**index1**`, `**index2**`)` or `substring(`**index1**`)`	the characters in this string from *index1* (inclusive) to *index2* (exclusive); if *index2* is omitted, grabs till end of string
`toLowerCase()`	a new string with all lowercase letters
`toUpperCase()`	a new string with all uppercase letters

String methods are called using dot notation, for example:

```
String name = "Bill";
System.out.println(name.length());    // 4
```

THE END