Services

- remember that a class contains methods
- we can think of these methods as the **services** provided by the class
  - they provide capabilities our program can use
  - also called **service methods**
- EX: all of the classes we’ve written so far provide a single service: a **main** method that performs some action

API

- the services that a class provides are documented in the class **Application Programming Interface (API)**
- the API contains, for each service:
  - the method header
  - a brief description of what the method does

Some API’s

- the API for three classes is provided
- **Output & Input** class API’s
- **Math** class API
  - the **Math** class provides math-related services
  - comes with Java
  - automatically available to all Java programs (no need to have **Math.class** in same directory)
Method Header

- the method header contains all of the information required to invoke the method

**syntax:**

\[\text{accessType} \quad \text{static} \quad \text{returnType} \quad \text{name} \quad (\text{argument-list})\]

- it’s a good idea to memorize the method header syntax

Access Type

**syntax:**

\[\text{accessType} \quad \text{static} \quad \text{returnType} \quad \text{name} \quad (\text{argument-list})\]

- indicates who can invoke this method
- all service methods have an access type of **public**, meaning any class can invoke them

static

**syntax:**

\[\text{accessType} \quad \text{static} \quad \text{returnType} \quad \text{name} \quad (\text{argument-list})\]

- will explain next week
- for now, all services will have **static** in their method header

Return Type

**syntax:**

\[\text{accessType} \quad \text{static} \quad \text{returnType} \quad \text{name} \quad (\text{argument-list})\]

- the data type of the value returned by the method
- a return type of **void** indicates the method does **not** return a value
Method Name

**syntax:**

```plaintext
accessType static returnType name (argument-list)
```

- the name of the method

Argument List

**syntax:**

```plaintext
accessType static returnType name (argument-list)
```

- consists of:
  ```plaintext
dataType1 argName1, dataType2 argName2, etc.
```

- empty parentheses () indicate the method takes no arguments

Examples

**the Output class provides a service for displaying an integer value with the following method header:**

```java
public static void showValue (String msg, int i)
```

Examples

**the Math class provides a service for calculating the sine of an angle with the following method header:**

```java
public static double sin (double ang)
```
Method Invocation

✴ with an API and our method invocation syntax we can invoke any method
✴ EX: invoke the `showValue` method of the `Output` class to output the integer `num`
✴ method invocation syntax:

```
ClassName.methodName(arguments);
Output.showValue("num is: ", num);
```

Arguments

✴ the arguments in the invocation must match the arguments in the API argument list
✴ must be same number of arguments
✴ must be in the same order
✴ must have the correct types
✴ EX:

```
public static void showValue (String msg, int i)
```

```
Output.showValue("num is: ", num)
```

Methods with Returns

✴ if a method has a return type other than `void` it returns a value that we can use

```
public static void showValue (String msg, int i)
```

```
Output.showValue("num is: ", num)
```

Return Example

✴ EX: invoke the `sin` method of the `Math` class to calculate the sine of 1.5 radians, and output the result
✴ method header for `sin`:

```
public static double sin (double ang)
```

✴ means method invocation is:

```
Math.sin(1.5)
```

✴ note that `sin` has a return type of `double`
Return Example (cont.)

✴ we can assign the returned value to a variable and output it:
  ```java
  double s;
  s = Math.sin(1.5);
  Output.showValue("sine is ", s);
  ```
✴ Note: the variable s isn’t really necessary (unless we want to do more with the value)
✴ since the method returns a double, it can be used in any expression where a double value is needed

Return Example (cont.)

✴ so we could accomplish the same thing as the above code with the single line:
  ```java
  Output.showValue("sine is ", Math.sin(1.5));
  ```
✴ when executed, the Math.sin invocation gets replaced with the returned value, which is then output

Module 16 Vocabulary

services service methods
Application Programming Interface (API)
access type public

return type void
argument list

Questions?

Email Kevin at sahrk@sou.edu