Useless Fact of the Day

- MySpace is currently the most visited domain by U.S. internet users. Two years ago, only 0.1% of Internet visits were to MySpace sites, while today 4.46% of visits are. For comparison, 3.89% of visits are to Google.

Anatomy of a Method

```java
public boolean isAwesomeMovie(String script, int explosions)
{
if (script.contains("pirate") || script.contains("ninja") ||
    script.contains("alligator wrestling") || explosions > 30)
{
    return true;
}
else
{
    return false;
}
}
```
Usefulness

• Package a unit of code
• Name and package a well-defined unit of work
• Aids in higher-level design, where you don’t have to know the internal workings of the method; instead, the name becomes a proxy for the work
• For example, FANG Engine methods
• Make your code cleaner and more organized (and easier to figure out what’s going on at a high level)

Usefulness

• These two blocks of code are exactly the same:

Example: Pong Collisions

• Avoid repetitive code
• If a section of code is repeated, or almost repeated, many times...
• ...you can use often replace the section of code with a method
• Write once, use many times
• Methods can be flexible, to even handle non-identical (but very similar) repeated code
Communication

• Methods need to communicate/share info with the rest of the program
• Three ways:
  • **Instance variables** - get info both in and out
    • These variables are “known” throughout a class, so any method in the class can see and modify them
  • **Parameters** - get info in (and sometimes out)
  • **Return statements** - get info out

Parameters

• Mainly get info **into** a method
  • in-only for Primitive-typed arguments (int, double, float, boolean, char)
  • Objects passed in as parameters sometimes allow info to get **out**
  • This happens if the method modifies the passed-in object
  • You won’t see this extremely often (sometimes modifying an object is not even possible — you cannot modify a String object, for example)

Return Statement

• Copies info **out of** a method
  • “Returns” some variable to the **invoking** statement or expression (the bit of code that called the method)
  • Often, the result is assigned to a variable, using =
  • The returned variable basically replaces the method call when the code executes
  • Every method must specify a **return type**
    • If it doesn’t return anything, the return type is **void**

Return: Example

```java
public double multiplyByThree(double x)
{
    return x * 3;
}
```
public Point2D.Double findMidpoint(Sprite s1, Sprite s2)
{
    Point2D.Double midpoint;
    Point2D.Double location1, location2;
    midpoint = new Point2D.Double(0, 0);
    location1 = s1.getLocation();
    location2 = s2.getLocation();
    midpoint.x = 0.5 * (location1.x + location2.x);
    midpoint.y = 0.5 * (location1.y + location2.y);
    return midpoint;
}