# Exercise 2: Get/Set Methods and Method Overriding

In Exercise 1, you extended the class **LineSwimmer** to create a BoringFish that accepted and passed along a name, and always moved 10 pixels at a time in both the X and Y directions.

## Your Assignment

Given the provided creatures, write a class called **FrightenedFish**, a fish that is scared of the surface of the water! After it touches the surface of the water, the FrightenedFish gets scared and swims down to the bottom faster than it was previously moving. When it touches the bottom of the aquarium, it calms down, and starts swimming upwards slower than its descent. It never changes its horizontal speed.

When it is created, this fish should:

* Initially move throughout the aquarium 10 pixels at a time in both the X and Y directions (a capability ChangingSwimmer provides);
* Accept and pass along a name its owner would like to use to identify it

When the fish hits the surface, it should:

* Reverse direction on the vertical axis (move down)
* Move twice as fast as its previous speed

When the fish hits the floor, it should:

* Reverse direction on the vertical axis (move up)
* Move half as fast as its previous speed

The FrightenedFish is very similar to the BoringFish when it is first created, so its constructor should be very similar to that of the BoringFish.

The FrightenedFish has a new behavior that the LineSwimmer and BoringFish did not: its speed in the vertical direction (Y axis) **changes**! To help you with this, we have given you a new class: the ChangingSwimmer. Make sure that you copy the ChangingSwimmer.java file from the network location to your local computer’s aquarium\creatures folder.

The ChangingSwimmer is very similar to the LineSwimmer, except that it provides methods to **get** and **set** its private variables deltaX and deltaY. Those private variables determine how quickly the creature is moving in the horizontal and vertical directions. Since they are private, you cannot change them directly in any class that extends the ChangingSwimmer. This is why you have the methods: *getSpeedX* and *getSpeedY* return the value of the deltaX and deltaY variables, and *setSpeedX* and *setSpeedY* can be used to change the value of the deltaX and deltaY variables. The setSpeedX function gives an example of why you might want to use a “set” method: it prevents the horizontal speed from being set to 0, which would ‘kill’ the fish! In general, “set” methods let you **control** how a private variable is modified.

How do you determine if the fish has hit the surface or the floor? You don’t have to! This logic is handled by the Aquarium simulation. What you do have to focus on is determining what the fish will do when it does hit the surface or floor. Open up the ChangingSwimmer class, and look at the methods hitFloor, hitSurface, hitLeftWall, and hitRightWall. Notice how they change the X or Y speed of the fish to be the opposite of its previous value. This makes sure that the fish stays in the aquarium, and doesn’t keep swimming off the screen! You will have to **override** the hitFloor and hitSurface methods to make the FrightenedFish do something different. To do this, you simply need to define new versions of the hitFloor and hitSurface methods in the FrightenedFish class that have the exact same method signature as the versions from ChangingFish. That means that the access level, return type, method name, and parameters all need to be the same. However, the contents of the method (the code in the squiggly brackets) will be different! You’ll probably need to use both the getSpeedY and setSpeedY methods in your implementation.

After implementing FrightenedFish, add a line to aquarium\Aquarium.java's “fillWithCreatures” method that adds a new FrightenedFish called “Felix” to the aquarium. If you need help figuring out how to do this, check out the example from exercise 1!