Introduction to GreenFoot: The Little Crab

With thanks to Bruce Chittenden and Michael Kolling

The Little Crab!

Inheritance
The Arrows Denote Hierarchy
Crab is an Animal
Animal is an Actor
Therefore, It Follows That
The Crab is Also an Actor
Click Compile all
Make a Crab

Right-click on Crab and select "new Crab()"

Find a spot on the sand to place the crab.

Run Your Program

Click Run and Nothing Happens
Let’s Make Some Changes

Right Click the Crab Class
Click On Open Editor

Welcome to Java!

When Run is Clicked
the Crab does nothing

This is because there is no Source Code in
the act Method for the Crab.
import greenfoot.*;  // (World, Actor, GreenfootImage, and Greenfoot)

/*
 * This class defines a crab. Crabs live on the beach.
 */

class Crab extends Animal
{
    public void act()
    {
        move();
    }
}
Almost Ready To Go...

Add the call to move();
Click Compile
Make sure it says “no syntax errors” before proceeding. If it does check to make sure that what you typed matches the above exactly.

“Act” moves one step at a time
“Run” moves continuously

Right Click on the Crab Class
Drag a Crab into the World
Click the Act and Run Buttons
Whee...

Crab Moves to Edge of the World

Crab Panic!

Place Multiple Crabs into the World
Click the Run Buttons
Click the Rest Button to clear the old crabs and start over
Turning

public void act ()
{
    turn (5);
}

Set turn to 5 and recompile
Whoa, I’m Gonna Barf...

Crab Spins to the Right

Now What Happens?

Set turn to -5 and recompile
What Does This Do?

```java
import greenfoot.*; // (World, Actor, GreenfootImage, and Greenfoot)

/*
 * This class defines a crab. Crabs live on the beach.
 */
public class Crab extends Animal {
    public void act() {
        move();
        turn(5);
    }
}
```

Give it a try!

Dealing with Screen Edges

Right Click on Animal Class and Click on Open editor
This is What an Animal Can Do
Method Signatures

- void turn (int angle)
- boolean atWorldEdge()
- void move()
A Crab is an Animal

Right Click on the Crab and Select inherited from Animal

Then Click on atWorldEdge()

Boolean: true or false
atWorldEdge now?

Move the Crab to the Edge of the World and Repeat the Experiment

Yup!
React to World Edge Now

import greenfoot.*; // (World, Actor, GreenfootImage, and Greenfoot)

/*
 * This class defines a crab. Crabs live on the beach.
 */
public class Crab extends Animal
{
    public void act()
    {
        if (atWorldEdge())
        {
            turn(17);
        }
        move();
    }
}

Give it a try!

Crab 2.0!

Crab Runs Around the World Turning when it Encounters a Wall
Crab 2.1

Set the Turn to 180 and the Crab Should Reverse Direction

Exercise 2.14

Place the move () Inside the if Statement
Summary

If the Crab is Not Near the Edge Nothing Happens.
If the Crab is at the Edge, Then It Turns Around.

What Have We Learned?

In this chapter, we have seen how to call methods such as move(), with and without parameters. This will form the basis for all further Java Programming.

We have encountered a glimpse of inheritance. Classes inherit the methods from their superclasses.

And, very important we have seen how to make decisions. We have used an if-statement for conditional execution.
Let’s try Crab 3.0!

Adding Random Behaviour

• The Greenfoot Environment has a Method to Give a Random Number.

• Greenfoot.getRandomNumber (20);
  • Parameter Limit of 20
  • Return a Random Number Range 0 to 19
Randomness

if ( Greenfoot.getRandomNumber (100) < 10) {
    // Code Here Executes 10% of the Time
}

Randomness

if ( Greenfoot.getRandomNumber (100) < 7) {
    // Code Here Executes 7% of the Time
}
Java Comparison Operators

- `<`  less than
- `<=`  less than or equal
- `==`  equal
- `!=`  not equal
- `>=`  greater than or equal
- `>`  greater than

Adding Randomness

```java
import greenfoot.*; // (World, Actor, GreenfootImage, and Greenfoot)

/*
 * This class defines a crab. Crabs live on the beach..
 */

public class Crab extends Animal
{
    public void act()
    {
        if ( atWorldEdge() )
        {
            turn(17);
        }

        if ( Greenfoot.getRandomNumber(100) < 10 )
        {
            turn(5);
        }

        move();
    }
}

Give it a try!
```
Drunken Crab

Crab Randomly 10% of the time Turns
Crab Turns Only to the Right
Crab Turns 5 Degrees When it Turns

import greenfoot.*; // (World, Actor, GreenfootImage, and Greenfoot)
/*
 * This class defines a crab. Crabs live on the beach.
 */
public class Crab extends Animal
{
    public void act()
    {
        if ( atWorldEdge() )
        {
            turn(17);
        }

        if ( Greenfoot.getRandomNumber(100) < 10 )
        {
            turn( Greenfoot.getRandomNumber(45) );
        }

        move();
    }
}

Adding Randomness

Give it a try!
Drunken Crab

Drunken Crab

Adding Randomness

import greenfoot.*; // (World, Actor, GreenfootImage, and Greenfoot)
/*
 * This class defines a crab. Crabs live on the beach.
 */
public class Crab extends Animal {
    public void act() {
        if (atWorldEdge()) {
            turn(17);
        }

        if (Greenfoot.getRandomNumber(100) < 10) {
            turn(Greenfoot.getRandomNumber(91) - 45);
        }

        move();
    }
}

Give it a try!
Drunken Crab

- Crab Randomly 10% of the time Turns
- Crab Turns Both Left and Right
- Crab Randomly Turns Between -45 and 45 Degrees When it Turns

Let’s Add Worms

Right Click on the Animal Class and then Click on New subclass
Let’s Add Worms

Don’t forget to click Compile all.
Crab Meet Worm

Crabs Move Randomly
Worms Do Nothing
Crabs Ignore the Worms

Eating Worms

boolean canSee (java.lang.Class cls)
void eat (java.lang.Class cls)
public void act()
{
    if ( atWorldEdge() )
    {
        turn(17);
    }
    if ( Greenfoot.getRandomNumber(100) < 10 )
    {
        turn( Greenfoot.getRandomNumber(91)-45 );
    }
    move();
    if ( canSee (Worm.class) )
    {
        eat (Worm.class);
    }
}
Crabs Eat the Worms

Java Library Documentation

Click on Java Library Documentation
Look for java.lang.class
Java 2 Platform Packages

Package java.lang

Description

Java 2 Platform Packages

java.lang

Contains all of the classes for creating user interfaces, including graphics and images.

java.awt

Contains classes necessary to create an applet and the classes that an applet uses to communicate with the applet container.

drag and drop

Drag and drop is a drag-and-drop interface for Swing. It provides a mechanism to transfer information between two entities that are logically associated with presentation elements in the GUI.

java.beans

Provides mechanisms and classes for dealing with different types of events fired by JFC components.

java.applet

Provides classes and interfaces relating to applets.

java.awt.datatransfer

Provides interfaces and classes for transferring data between and within applications.

java.swing

Drag and drop is a drag-and-drop interface for Swing. It provides a mechanism to transfer information between two entities that are logically associated with presentation elements in the GUI.

java.lang

Provides classes and interfaces relating to basic concepts.

java.nio

Provides classes and interfaces for basic file system and stream access.

java.io

Provides classes and interfaces for basic file system and stream access.

java.util

Provides classes and interfaces for basic file system and stream access.

java.text

Provides classes and interfaces for basic file system and stream access.

java.lang

Provides classes that are fundamental to the design of the Java programming language.
Creating New Behaviours Through Methods

/*
 * Check whether we have stumbled upon a worm.
 * If we have, eat it. If not, do nothing.
 */
public void lookForWorm()
{
    if ( canSee(Worm.class) )
    {
        eat(Worm.class);
    }
}

Add it to Our Crab Behaviour

public void act()
{
    if ( atWorldEdge() )
    {
        turn(17);
    }

    if ( Greenfoot.getRandomNumber(100) < 10 )
    {
        turn(5);
    }

    move();
    lookForWorm();
}

We have to define what this does by adding in the code we just saw...
We Need To Add This To Crab

/*
 * Check whether we have stumbled upon a worm.
 * If we have, eat it. If not, do nothing.
 */
public void lookForWorm()
{
    if ( canSee(Worm.class) )
    {
        eat(Worm.class);
    }
}

Worm Buffet
Eventually the Crabs Eat All the Worms