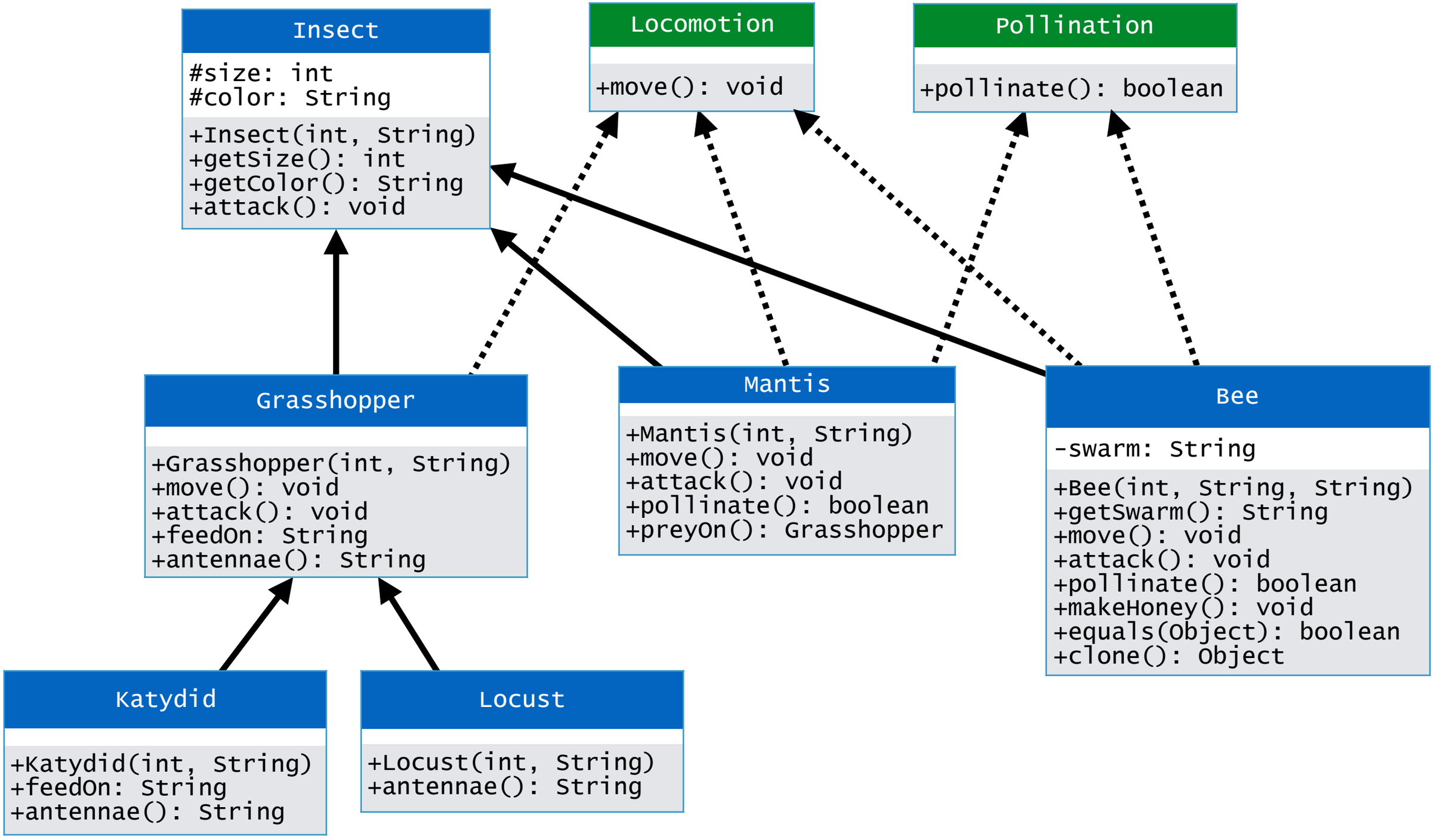
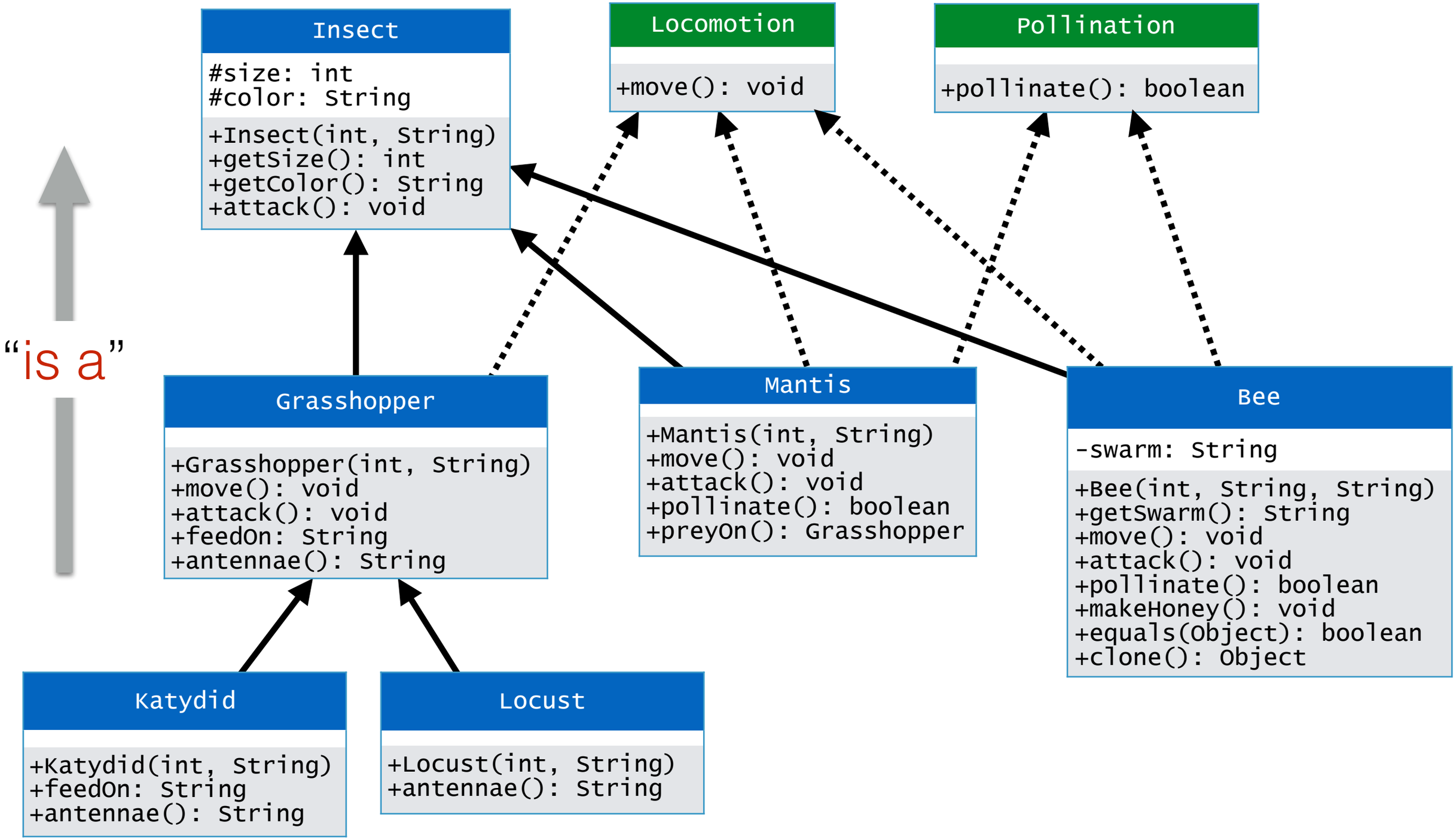
The `Insect` Class

Source for all images: Wikipedia


Insect
#size: int
#color: String
+Insect(int, String)
+getSize(): int
+getColor(): String
+attack(): void
Locomotion
+move(): void
Pollination
+pollinate(): boolean
Grasshopper
+Grasshopper(int, String)
+move(): void
+attack(): void
+feedOn: String
+antennae(): String
Mantis
+Mantis(int, String)
+move(): void
+attack(): void
+pollinate(): boolean
+preyOn(): Grasshopper
Bee
-swarm: String
+Bee(int, String, String)
+getSwarm(): String
+move(): void
+attack(): void
+pollinate(): boolean
+makeHoney(): void
+equals(Object): boolean
+clone(): Object
Katydid
+Katydid(int, String)
+feedOn: String
+antennae(): String
Locust
+Locust(int, String)
+antennae(): String


Insect
#size: int
#color: String
+Insect(int, String)
+getSize(): int
+getColor(): String
+attack(): void
Locomotion
+move(): void
Pollination
+pollinate(): boolean
"is a"
Grasshopper
+Grasshopper(int, String)
+move(): void
+attack(): void
+feedOn: String
+antennae(): String
Mantis
+Mantis(int, String)
+move(): void
+attack(): void
+pollinate(): boolean
+preyOn(): Grasshopper
Bee
-swarm: String
+Bee(int, String, String)
+getSwarm(): String
+move(): void
+attack(): void
+pollinate(): boolean
+makeHoney(): void
+equals(Object): boolean
+clone(): Object
Katydid
+Katydid(int, String)
+feedOn: String
+antennae(): String
Locust
+Locust(int, String)
+antennae(): String

```
package insect;

public abstract class Insect
{
    protected int size;        // inches
    protected String color;

    public Insect(int size, String color)
    {
        this.size = size;
        this.color = color;
    }

    public int getSize()
    {
        return size;
    }

    public String getColor()
    {
        return color;
    }

    public abstract void attack();
}
```

Interfaces

```
package insect;

public interface Locomotion
{
   void move();
}
```

```
package insect;

public interface Pollination
{
   boolean pollinate();
}
```

Grasshopper

Gaudy grasshopper, *Atractomorpha lata*, evades predators with camouflage.

Lubber grasshopper, *Titanacris albipes*, has deimatically coloured wings, used to startle predators.

Leaf grasshopper, *Phyllochoreia ramakrishnai*, mimics a green leaf.

Painted grasshopper, *Dactylotum bicolor*, deters predators with warning coloration.

Spotted grasshopper, *Aularches miliaris*, defends itself with toxic foam and warning colours.[39]

```
package insect;

public abstract class Grasshopper extends Insect
    implements Locomotion
{
    public Grasshopper(int size, String color)
    {
        super(size, color);
    }

    public void move()
    {
        System.out.println("hop");
    }

    @Override
    public void attack()
    {
        System.out.println("bite");
    }


    public String feedOn()
    {
        return "grass";
    }

    public abstract String antennae();
}
```

Katydid

```
package insect;

public class Katydid extends Grasshopper
{
    public Katydid(int size, String color)
    {
        super(size, color);
    }

    @Override
    public String feedOn()
    {
        return "variety";
    }

    @Override
    public String antennae()
    {
        return "Long";
    }
}
```

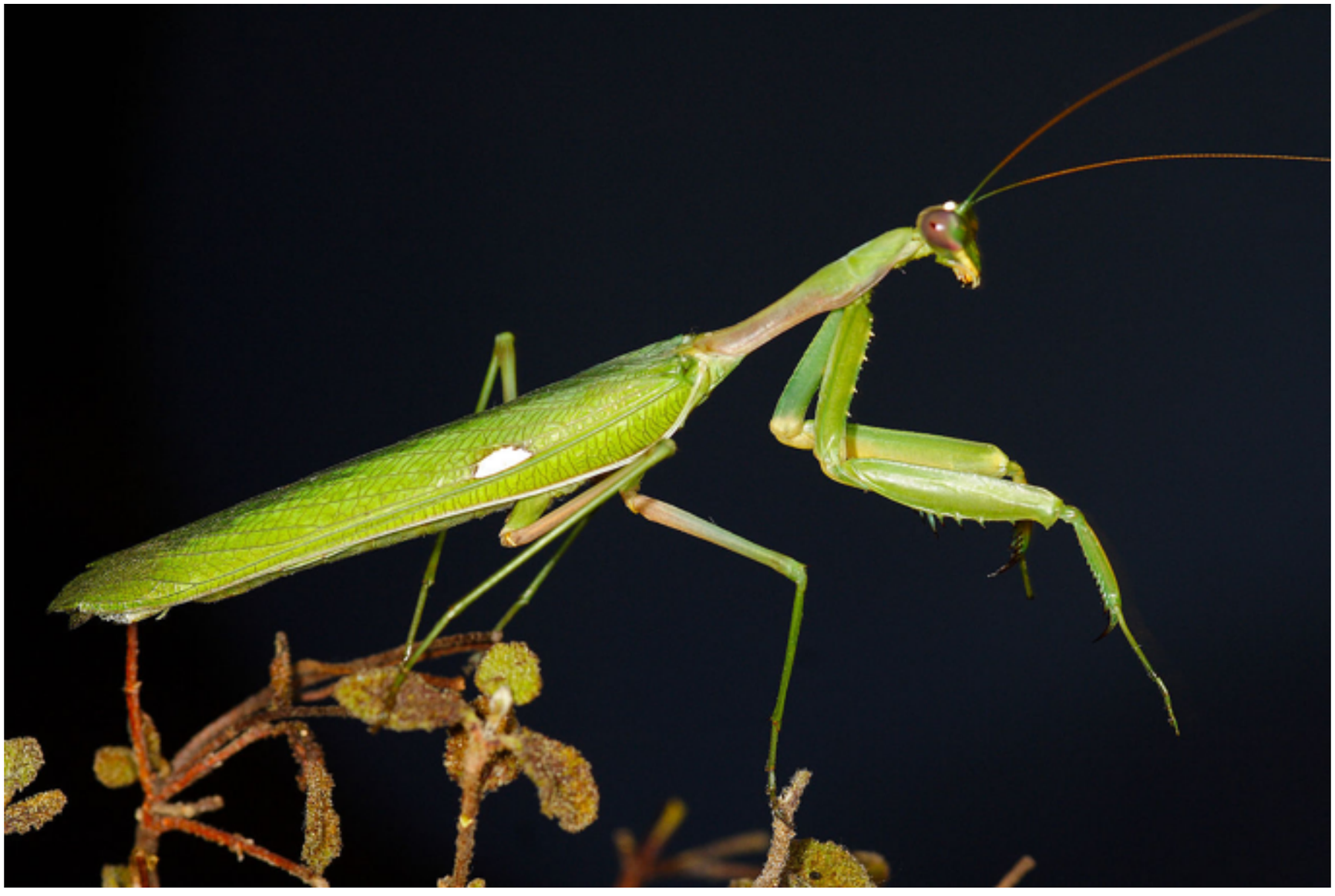
Locust

```
package insect;

public class Locust extends Grasshopper
{
    public Locust(int size, String color)
    {
        super(size, color);
    }

    @Override
    public String antennae()
    {
        return "Short";
    }
}
```

Mantis



```java
package insect;

public class Mantis extends Insect
    implements Locomotion, Pollination
{
    public Mantis(int size, String color)
    {
        super(size, color);
    }

    @Override
    public void move()
    {
        System.out.println("crawl");
    }

    @Override
    public boolean pollinate()
    {
        return false;
    }
```

```
    @Override
    public void attack()
    {
        System.out.println("strike");
    }

    public Grasshopper preyOn()
    {
        return new Locust(3, "Brown");
    }
}
```

Bee

```java
package insect;

public class Bee extends Insect implements Locomotion, Pollination
{
    private String swarm;

    public Bee(int size, String color, String swarm)
    {
        super(size, color);
        this.swarm = swarm
    }

    public String getSwarm()
    {
        return swarm;
    }

    @Override
    public void move()
    {
        System.out.println("fly");
    }
```

```java
@Override
public void attack()
{
    System.out.println("sting");
}

@Override
public boolean pollinate()
{
    return true;
}

public void makeHoney()
{
    System.out.println("Orange Blossom");
}
```

```java
@Override
public boolean equals(Object o)
{
    if (o == null || o.getClass() != getClass())
    {
        return false;
    }

    // typecast o to Bee so that we can compare data members
    Bee b = (Bee) o;

    // Compare the data members and return accordingly
    return b.size == size
            && (b.color == color || b.color != null &&
                b.color.equals(color))
            && (b.swarm == swarm || b.swarm != null &&
                b.swarm.equals(swarm));
}
```

```
    @Override
    public Object clone()
    {
/*      try
        {
            Bee b = (Bee) super.clone();
            b.swarm = this.swarm;
            return b;
        }
        catch (CloneNotSupportedException e)
        {
            return null;
        } */

        Bee b = new Bee(size, color, swarm);
        return b;
    }


}
```

InsectComparator

```
package insect;

import java.util.Comparator;

public class InsectComparator implements Comparator<Insect>
{
    @Override
    public int compare(Insect i1, Insect i2)
    {
        return i1.getSize() - i2.getSize();
    }
}
```

InsectComparator

```
package insect;

import java.util.Comparator;

public class InsectComparator implements Comparator<Insect>
{
    @Override
    public int compare(Insect i1, Insect i2)
    {
        return i1.getSize() - i2.getSize();
    }
}
```