Introduction to Computational Thinking

Inheritance and object construction;
Method Overriding; Object Hierarchy;
Event-Driven Programming

Qiao Xiang, Qingyu Song
https://sngroup.org.cn/courses/ct-xmuf25/index.shtml

12/20/2025

This deck of slides are heavily based on cs112 at Yale University and cs101 at UCAS, respectively, by courtesy of Dr. Y. Richard Yang and Dr. Zhiwei Xu.

Coordination

Coordinating Critter Exercise: Hipster (潮人)

- A group of hipster critters want to hangout together
- Each hipster can suddenly become inspired and choose a random board location called an edgy (前卫) bar
- A hipster go north until reaches edgy bar's horizontal, then east until reaching the bar

Solution 1 (See Hipster.java)

```
public class Hipster extends Critter {
    private Random rand;
    private int edgyBarX, edgyBarY;
    private int nextT, t;
    private final int FLASH_INTERVAL = 200;
    public Hipster() {
        rand = new Random();
        t = 0; nextT = rand.nextInt(FLASH_INTERVAL);
    }
    public Direction getMove(String[][] grid) {
        t ++;
        if (t == nextT) {
            edgyBarX = rand.nextInt( getWidth() );
            edgyBarY = rand.nextInt( getHeight() );
            t = 0; nextT = rand.nextInt(FLASH_INTERVAL);
        }

        if (getY() != edgyBarY) {
            return Direction.NORTH;
        } else if (getX() != edgyBarX) {
            return Direction.EAST;
        } else {
            return Direction.CENTER;
        }
    }
    public String toString() {
        return "H(" + edgyBarX + "," +edgyBarY + ")”; }
```

Solution 1 (See Hipster.java)

```
public class Hipster extends Critter {
  private Random rand;
  private int edgyBarX, edgyBarY;
  private int nextT, t;
  private final int FLASH_INTERVAL = 200;
  public Hipster() {
      rand = new Random();
      t = 0; nextT = rand.nextInt(FLASH_INTERVAL);
  }
  public Direction getMove(String[][] grid) {
      t ++;
      if (t == nextT) {
          edgyBarX = rand.nextInt( getWidth() );
          edgyBarY = rand.nextInt( getHeight() );
          t = 0; nextT = rand.nextInt(FLASH_INTERVAL);
      }

      if (getY() != edgyBarY) {
          return Direction.NORTH;
      } else if (getX() != edgyBarX) {
          return Direction.EAST;
      } else {
          return Direction.CENTER;
      }
  }
  public String toString() {
      return "H(" + edgyBarX + "," +edgyBarY + ")"; }
```

Problem: Each hipster goes to a different bar. We want all hipsters to **share the same bar location.**

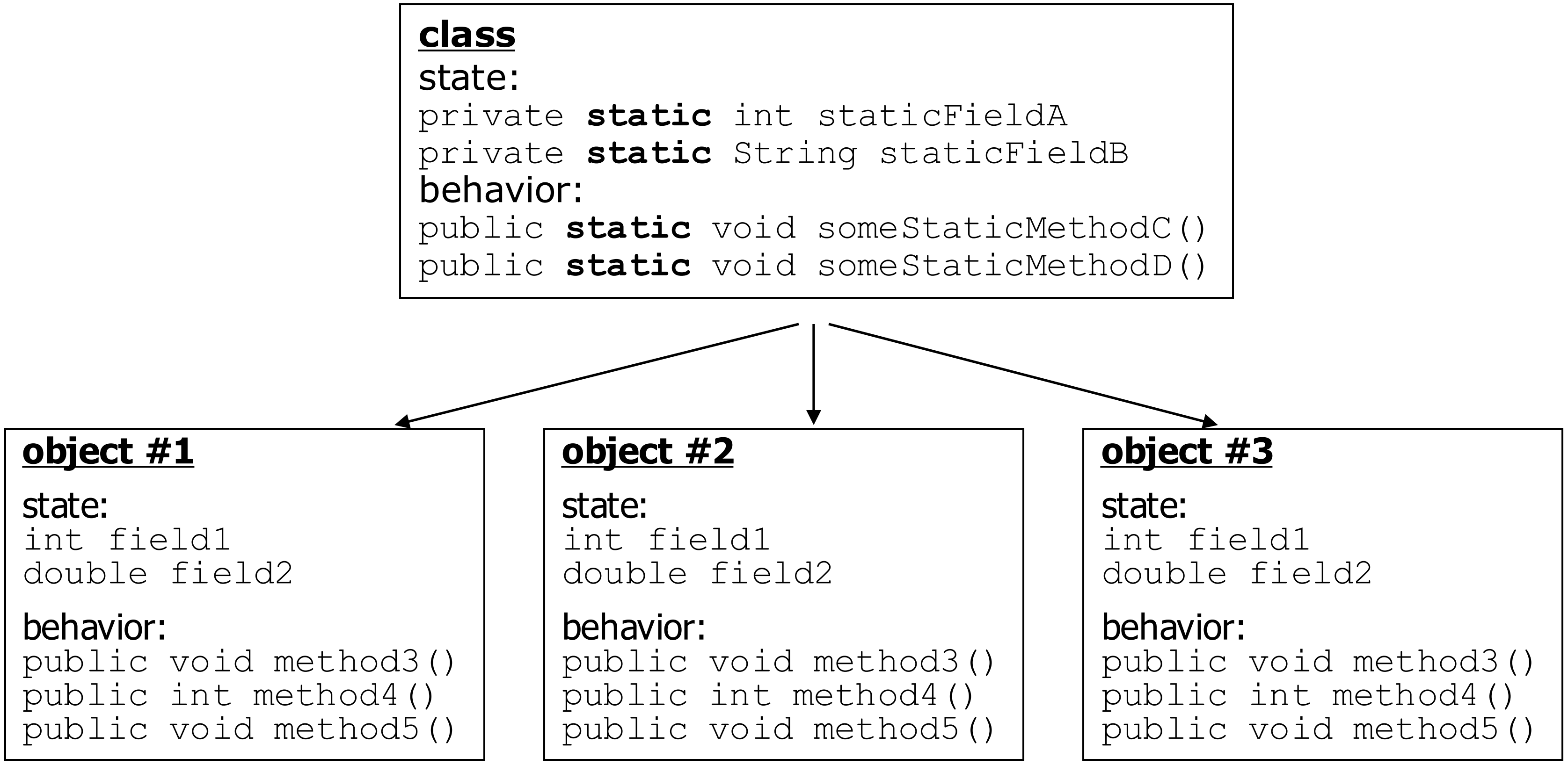
Solution 1 (See Hipster.java)

```
public class Hipster extends Critter {
  private Random rand;
  private static int edgyBarX, edgyBarY;
  private int nextT, t;
  private final int FLASH_INTERVAL = 200;
  public Hipster() {
      rand = new Random();
      t = 0; nextT = rand.nextInt(FLASH_INTERVAL);
  }
  public Direction getMove(String[][] grid) {
      t ++;
      if (t == nextT) {
         edgyBarX = rand.nextInt( getWidth() );
         edgyBarY = rand.nextInt( getHeight() );
         t = 0; nextT = rand.nextInt(FLASH_INTERVAL);
      }

      if (getY() != edgyBarY) {
         return Direction.NORTH;
      } else if (getX() != edgyBarX) {
         return Direction.EAST;
      } else {
         return Direction.CENTER;
      }
  }
  public String toString() {
      return "H(" + edgyBarX + "," +edgyBarY + ")”; }
```

Recall: Static members

- **static**: Part of a class, rather than part of an object.
  - Object classes can have static methods *and fields*.
  - Not copied into each object; shared by all objects of that class.



Accessing static fields

- From inside the class where the field was declared:

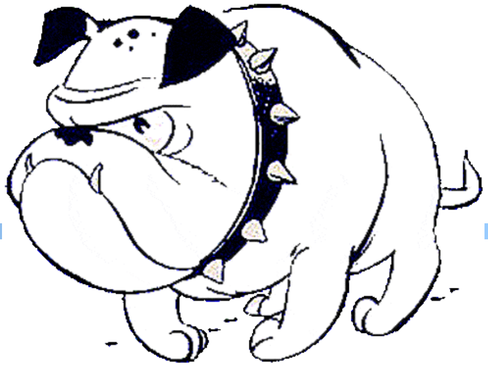
```
fieldName                      // get the value
fieldName = value;             // set the value
```

- From another class (if the field is `public`):

```
ClassName.fieldName            // get the value
ClassName.fieldName = value;   // set the value
```

  - *generally static fields are not public unless they are* `final`

Designing Bulldog



- Be open minded
- Think about strategies, e.g.,
  - How much state do your bulldogs keep and probe state?
  - When do your bulldogs eat/mate?
  - Is there an “optimal” fight strategy for a specific type of opponent?
  - Do your bulldogs play disguise(伪装)?
  - Does your strategy change with time?
  - Do your bulldogs coordinate their behaviors to form some kind of patterns?