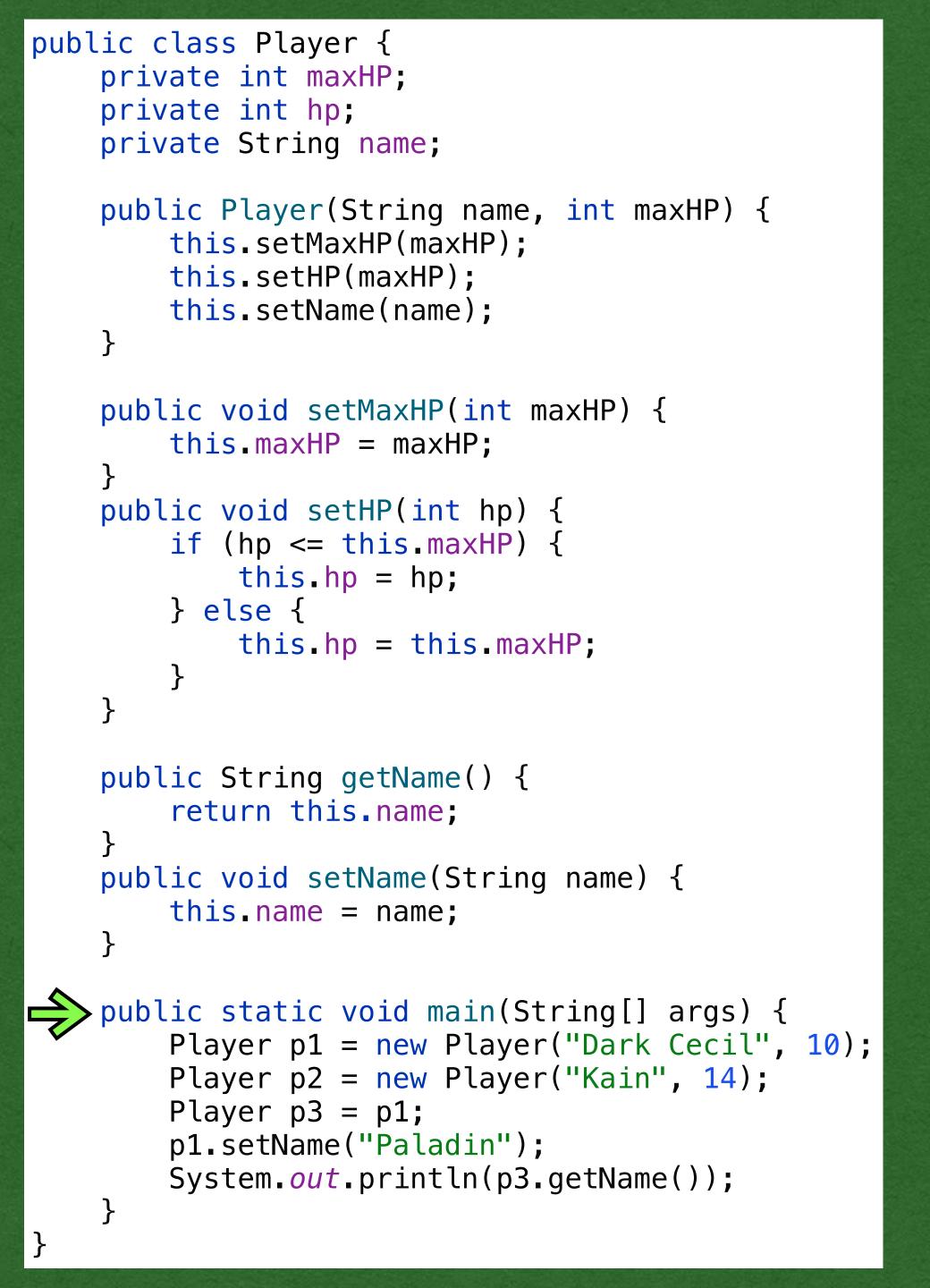
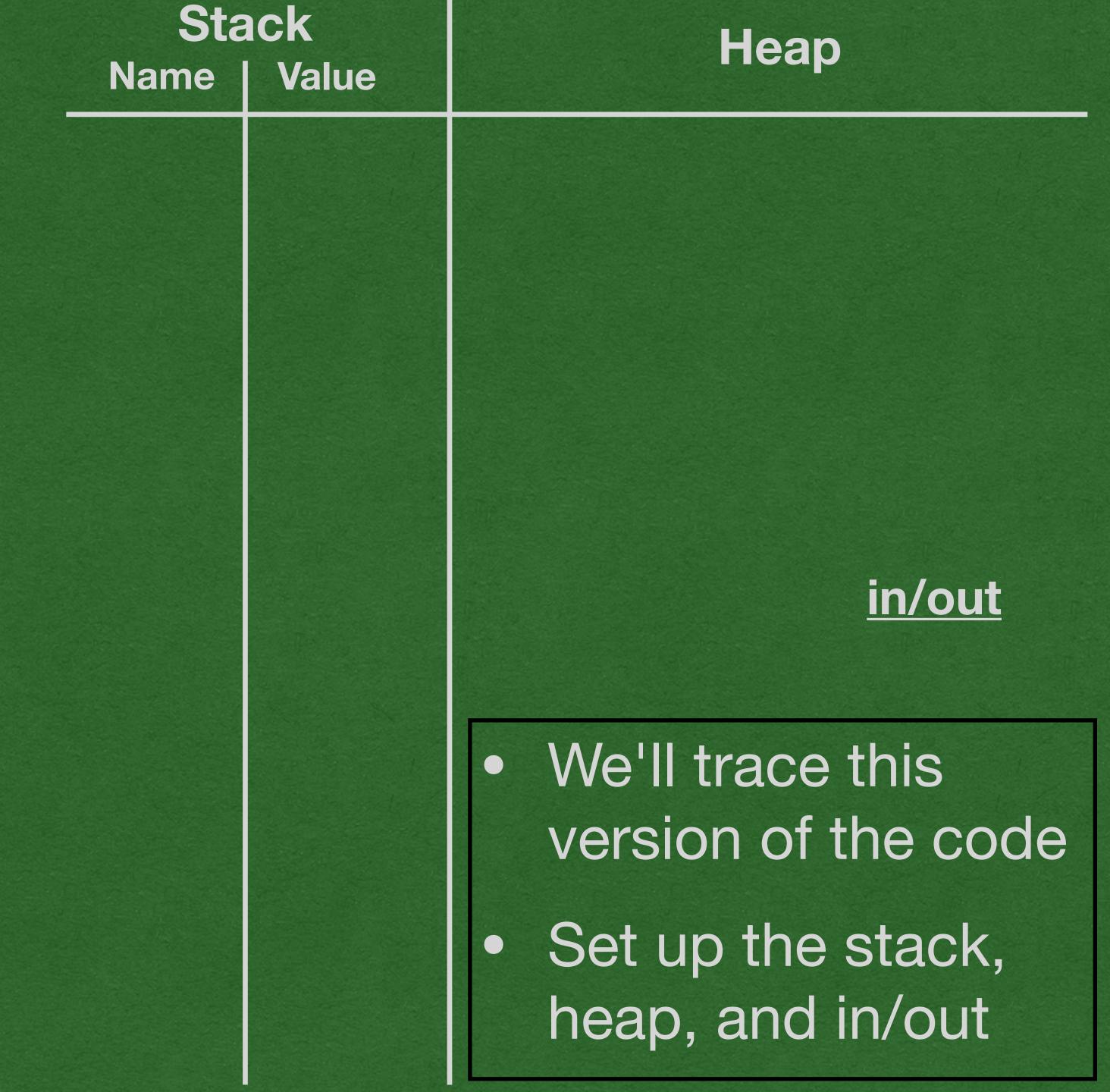
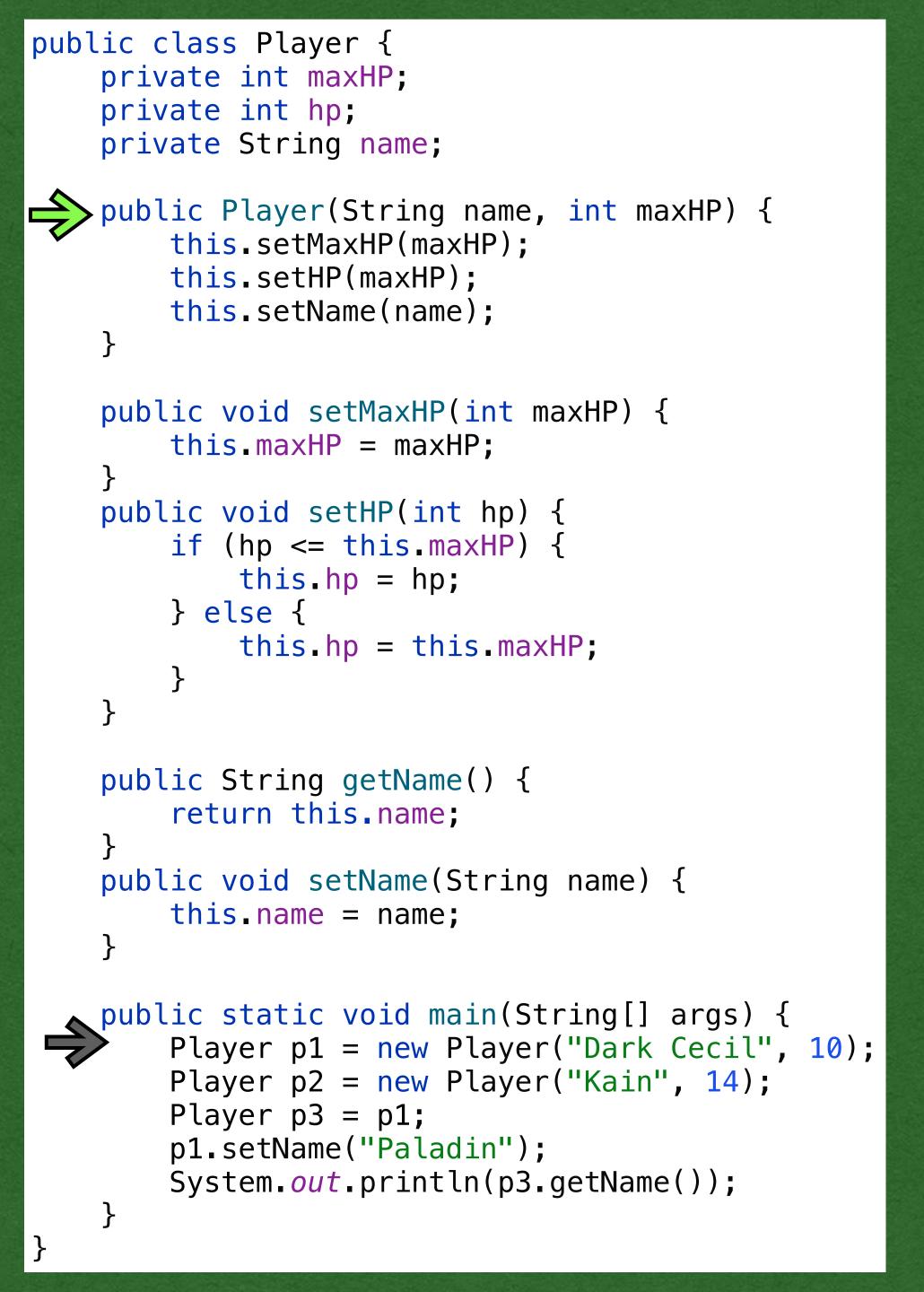
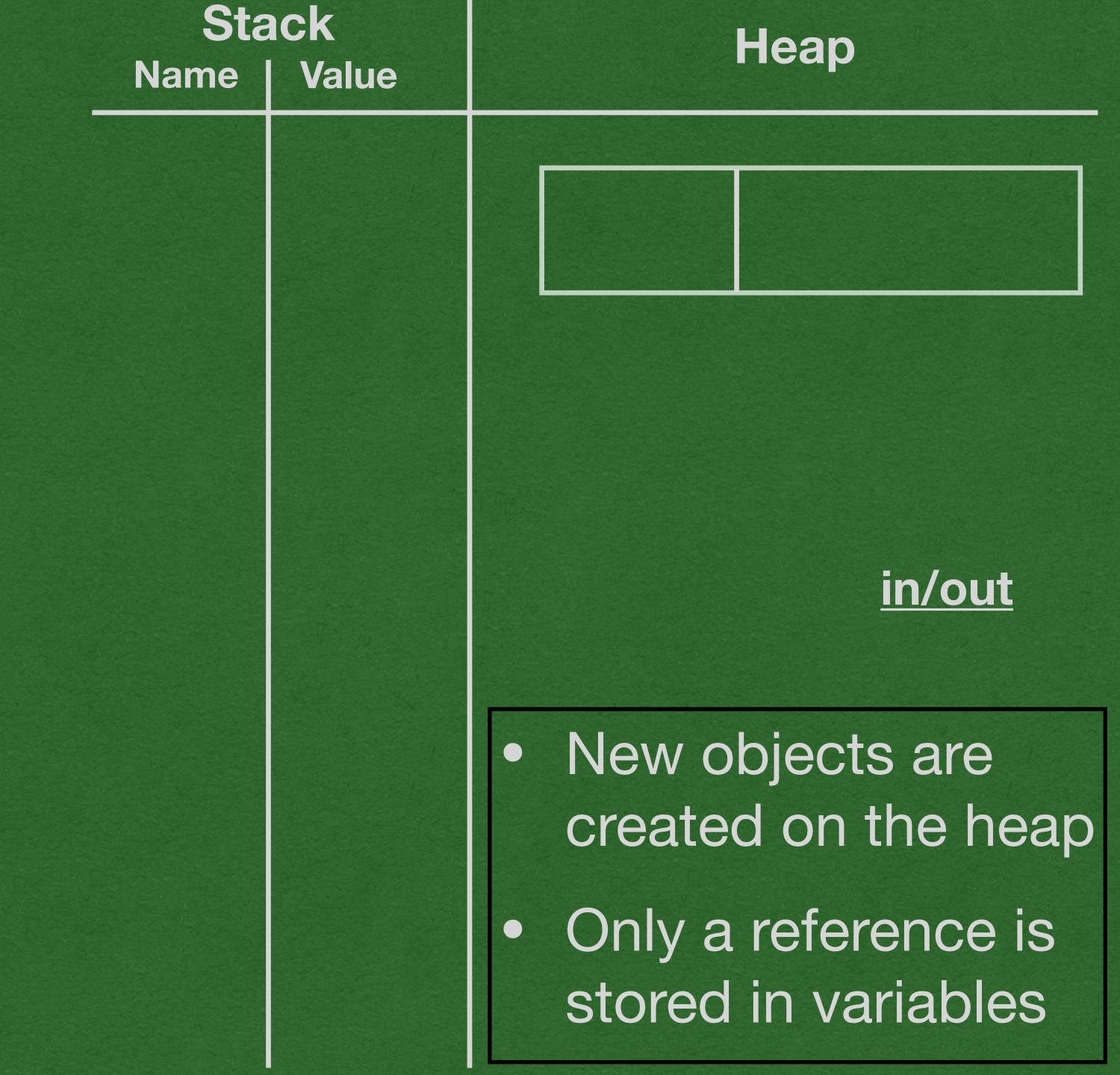
Classes: Part 2

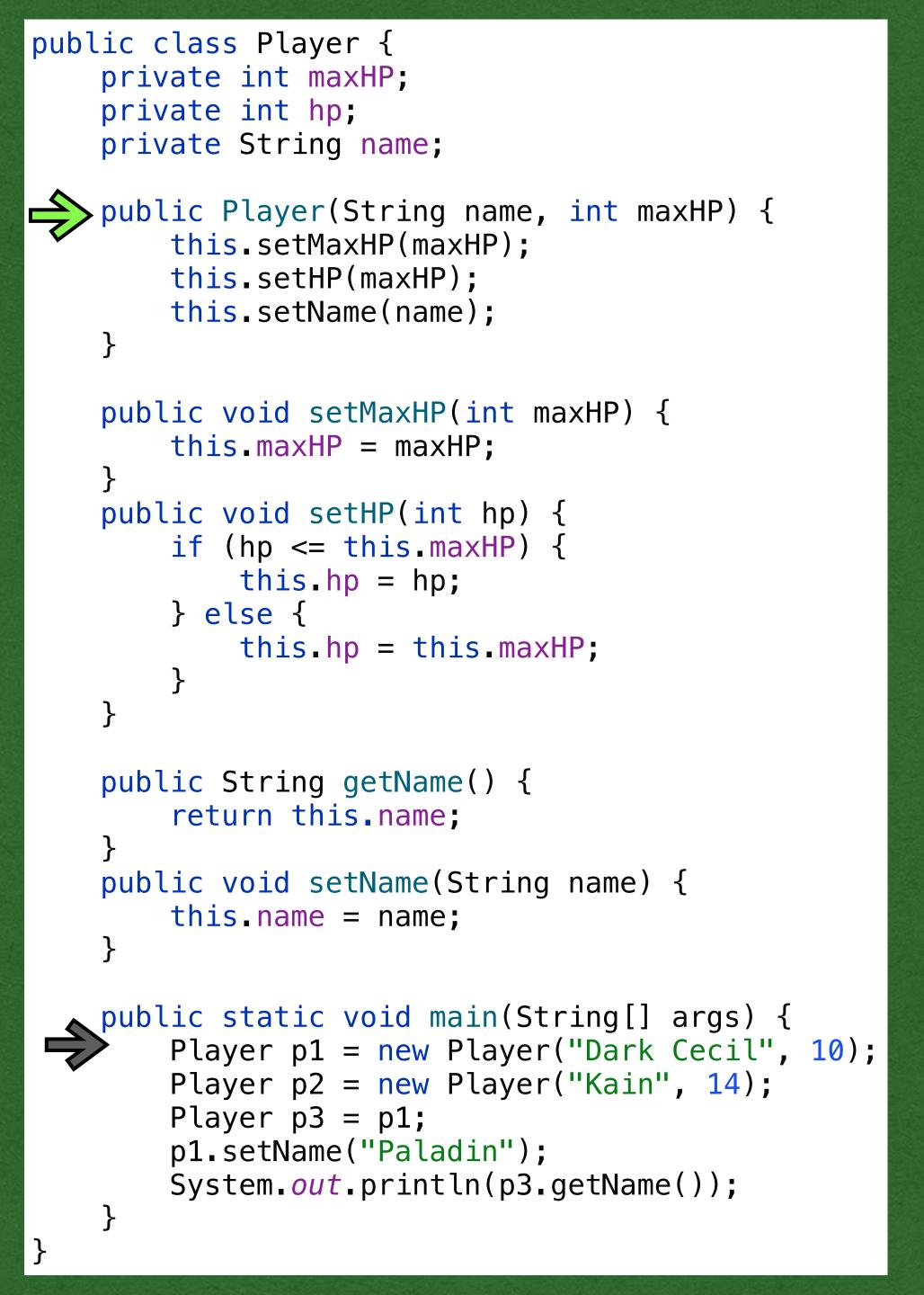
Diagram

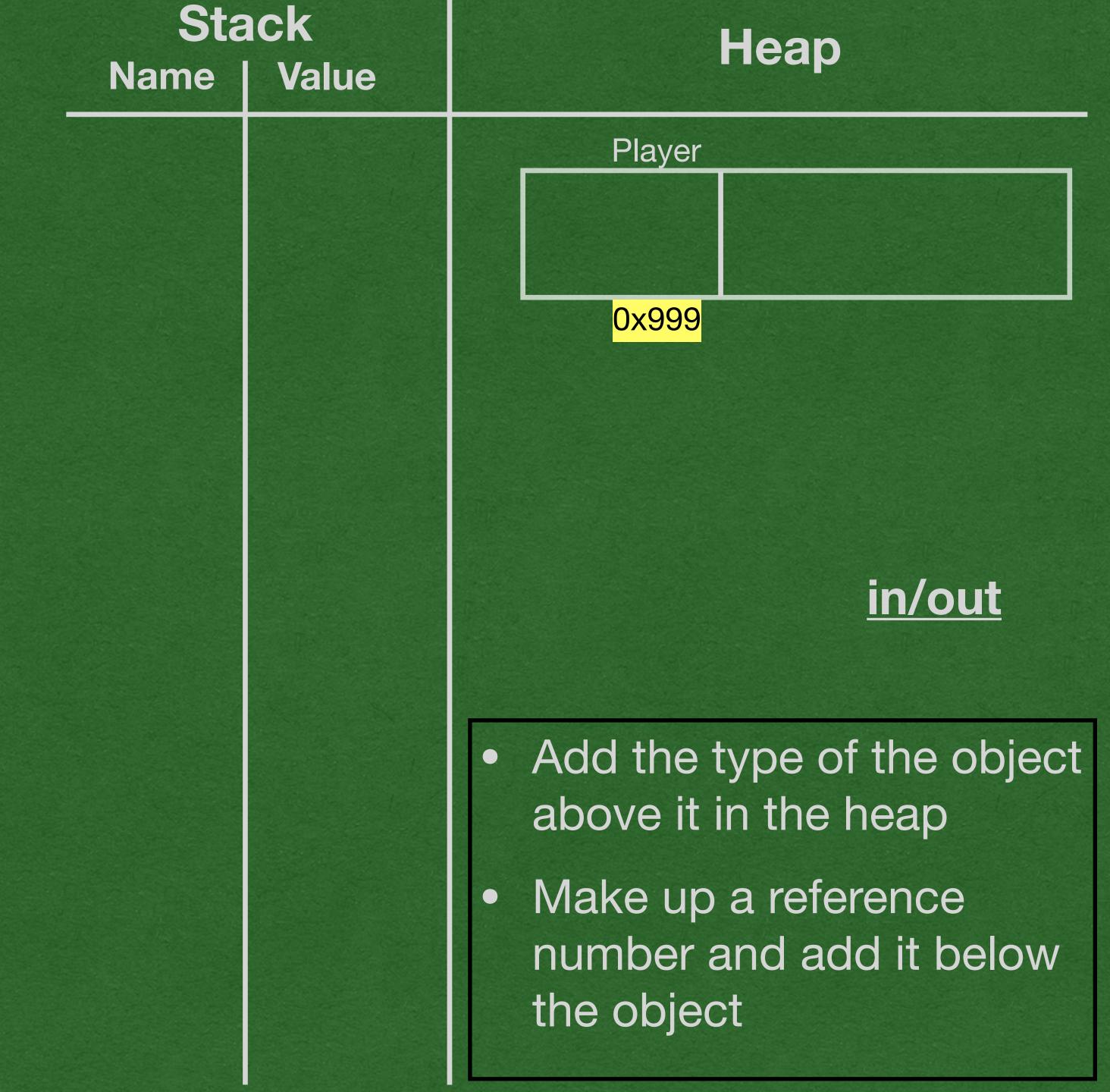


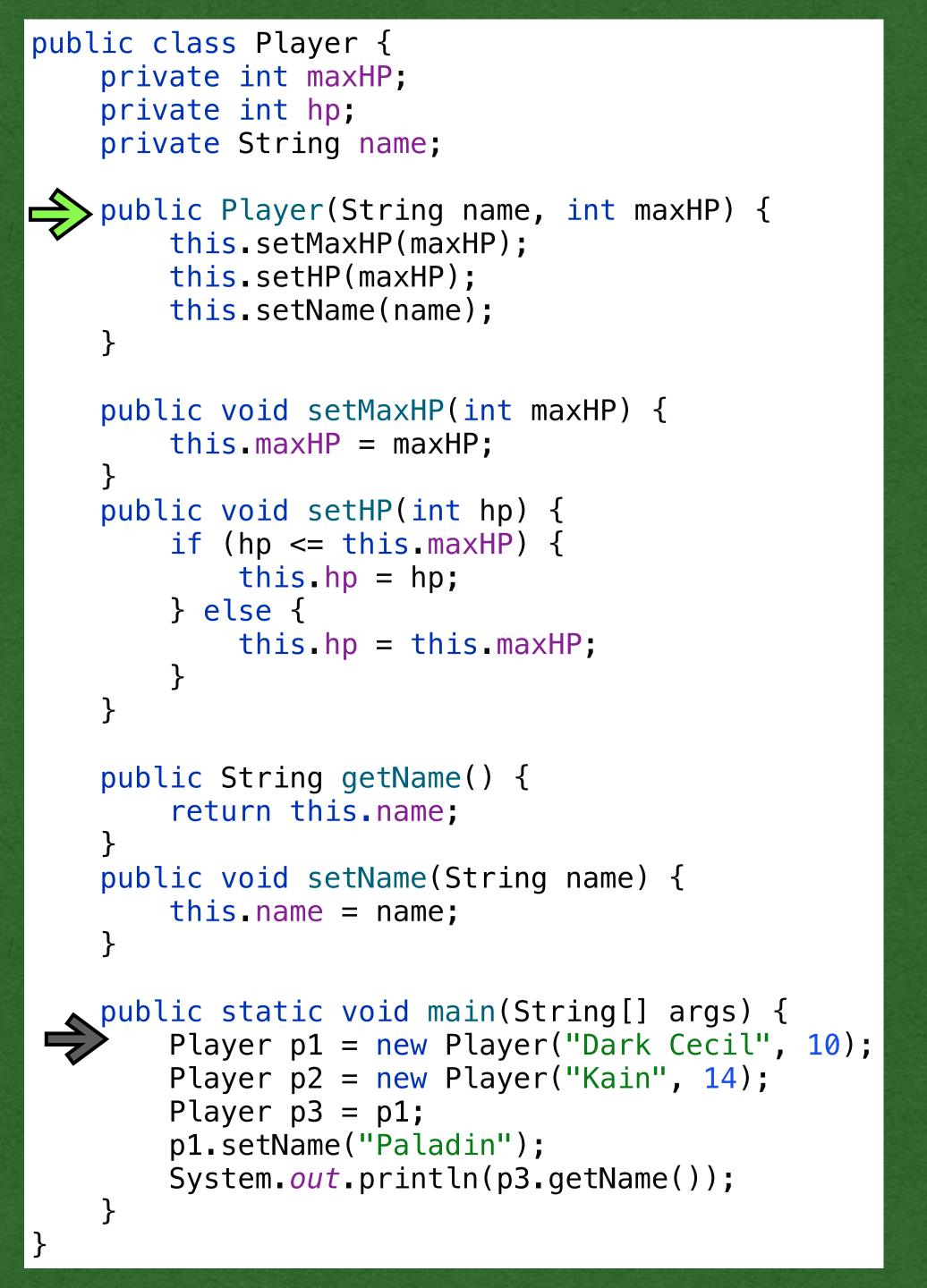














Player

maxHP

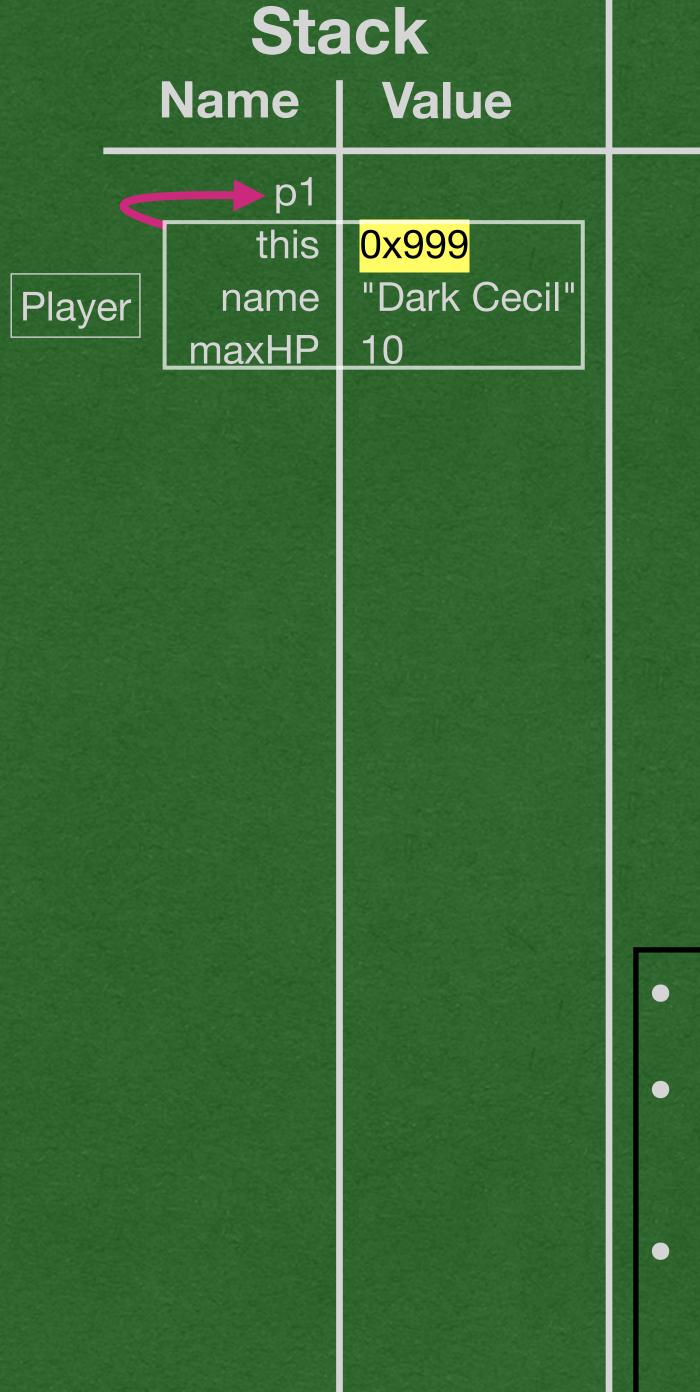
hp

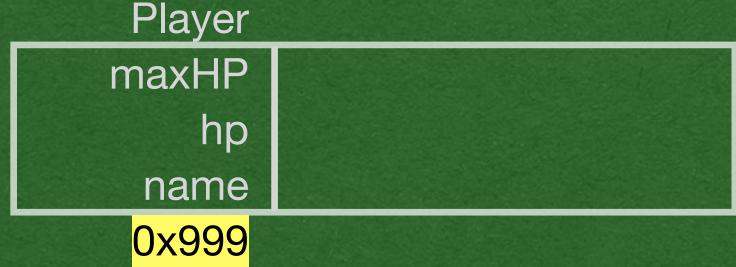
name

0x999

- Create all the instance variables and add them to the object
- Each new object will have its own copies of each instance variable

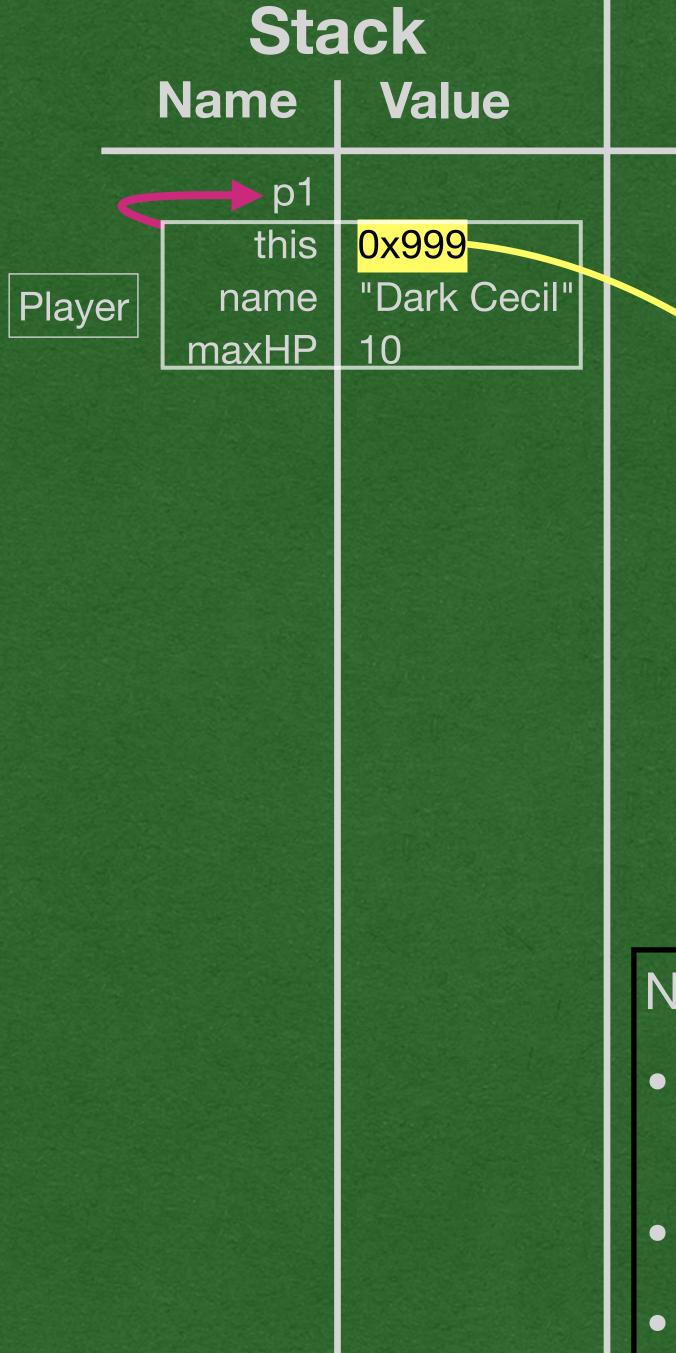
```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
   public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
   public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```

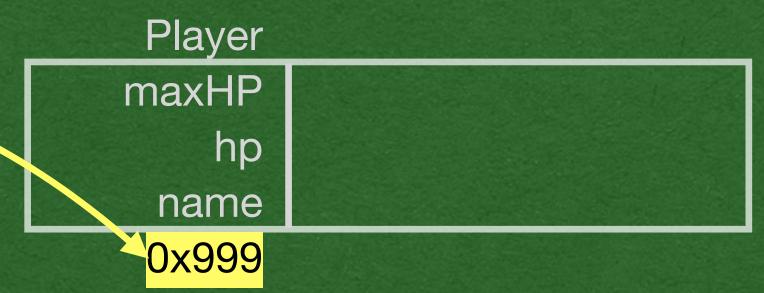




- The constructor is a method
- Add it to the stack along with all of its parameters
- Constructors return a reference to the object that was created

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
   public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
   public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



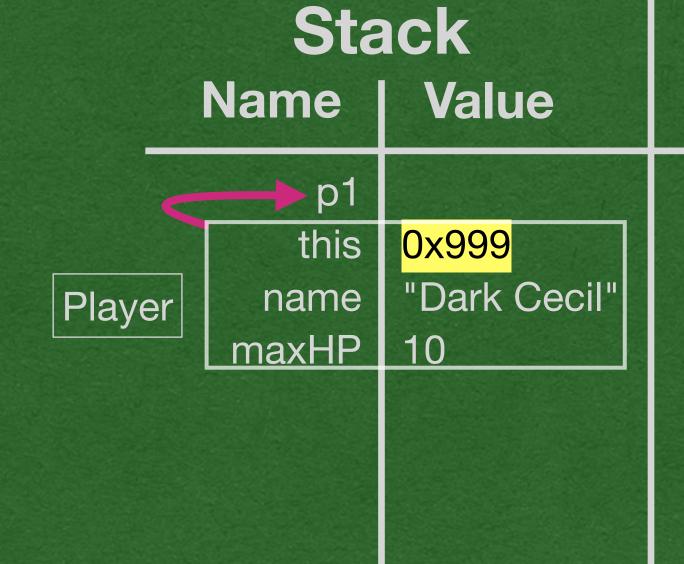


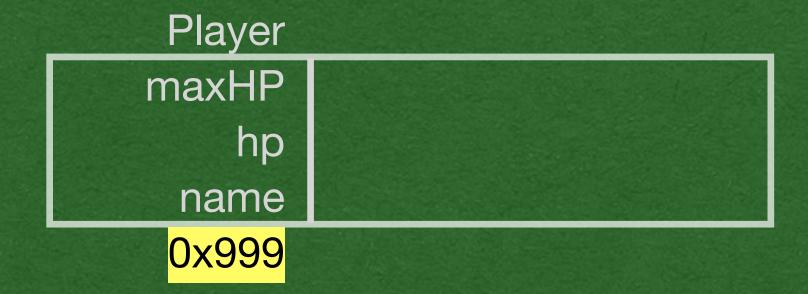
in/out

Notation Note:

- I'll stop drawing reference arrows in slides to reduce clutter
- References will be color coded
- It's recommended that you use arrows on quizzes

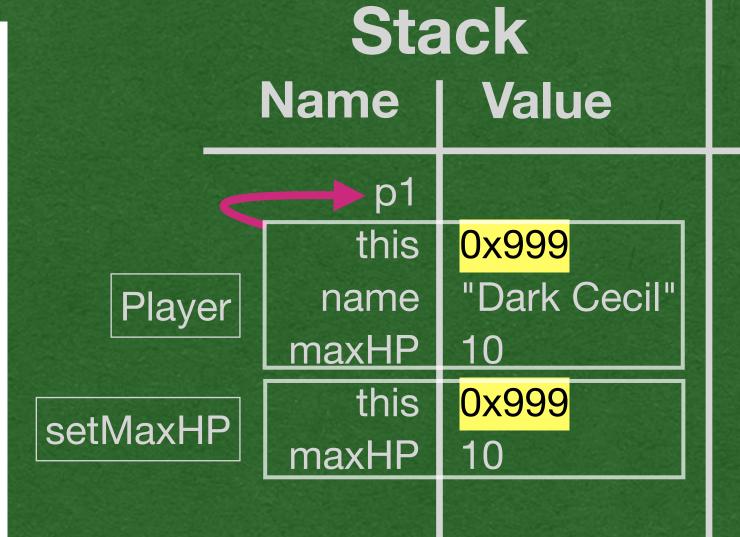
```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
   public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
   public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```

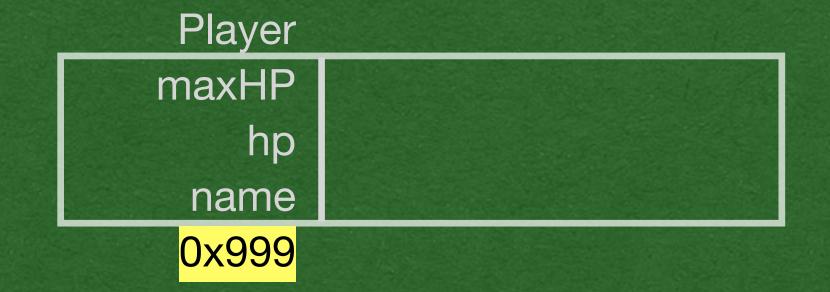




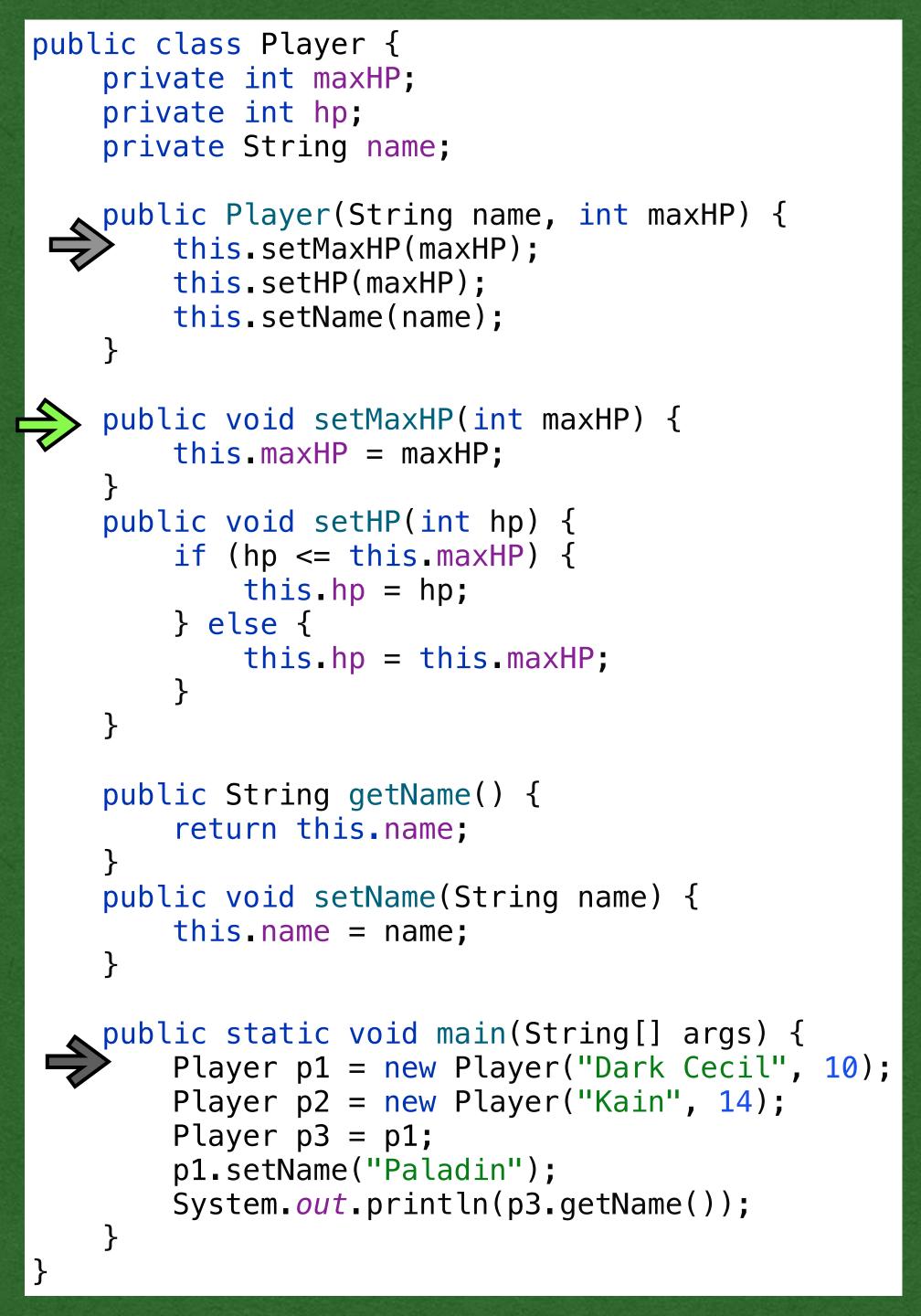
- All [non-static] methods have a "hidden" parameter of this
- For constructors, **this** stores a reference to the object that is being created

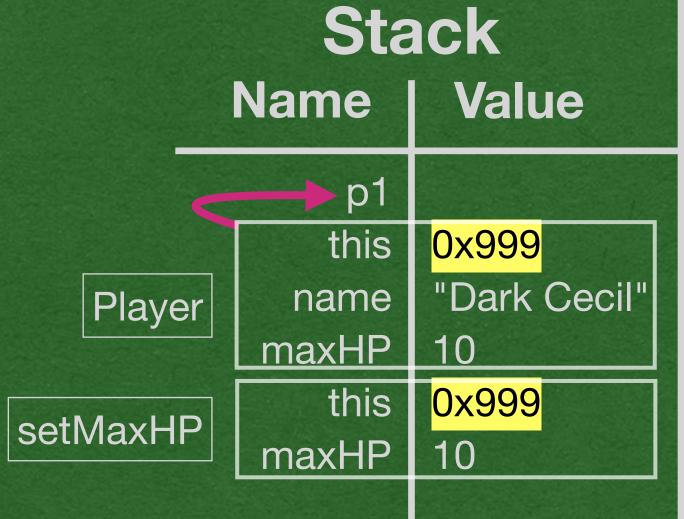
```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
   public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
   public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```

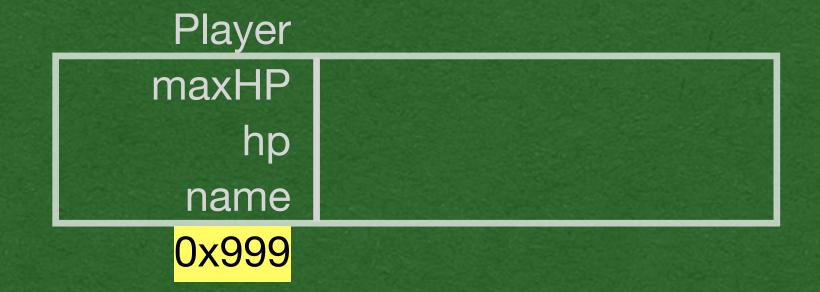




- Our constructor calls a setter method
- Methods contain a reference a reference to the calling object in this

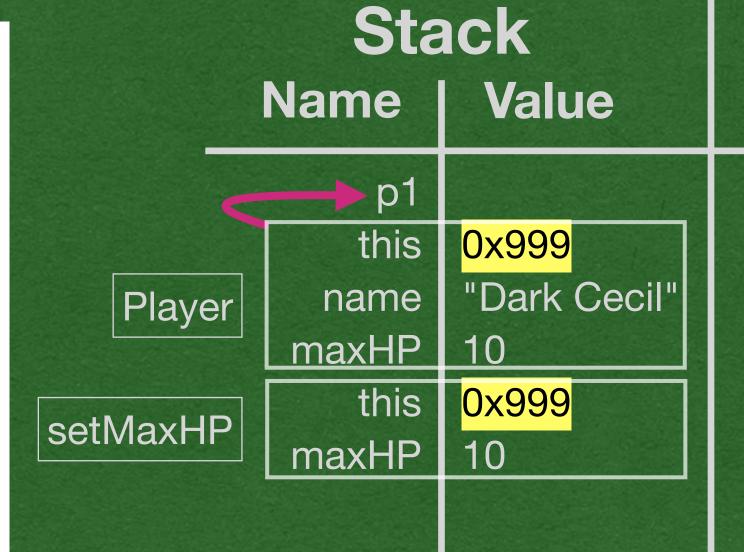


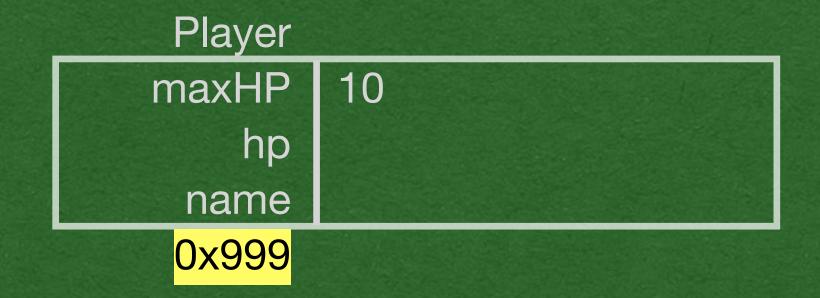




- This method was called by this in the Player constructor stack frame which stores the reference 0x999
- 0x999 is the object that called setMaxHP so that stack frame's this stores 0x999

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
   public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
   public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```

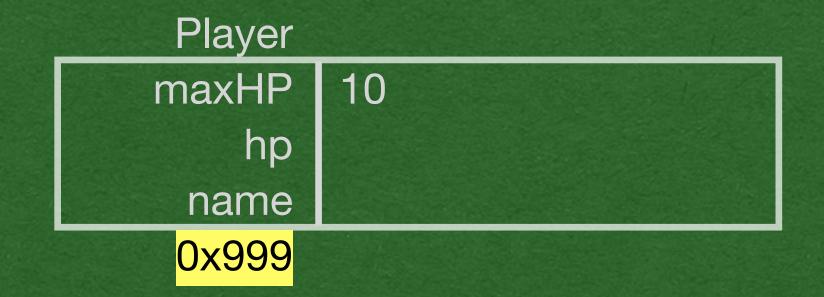




- The setter method changes the value of a variable stored in the heap
- Follow the reference stored in this and set it's maxHP variable

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
   public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
   public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



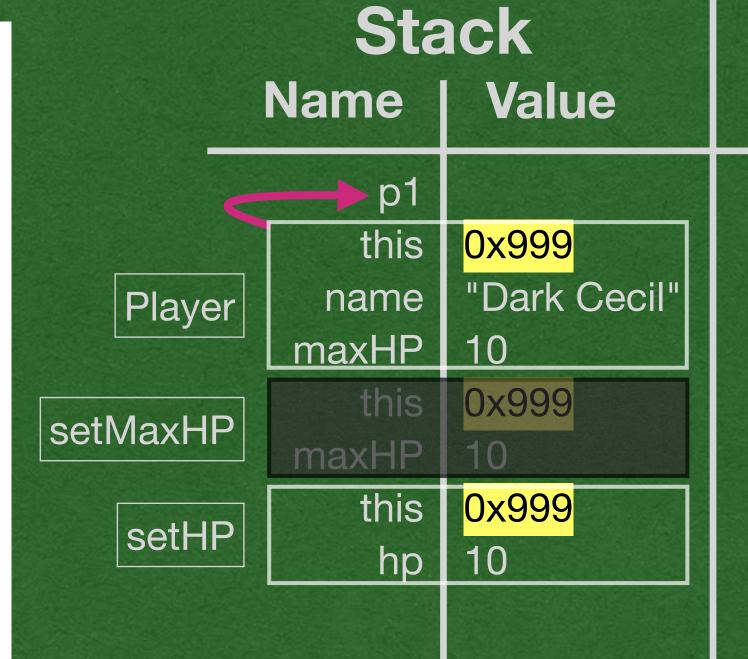


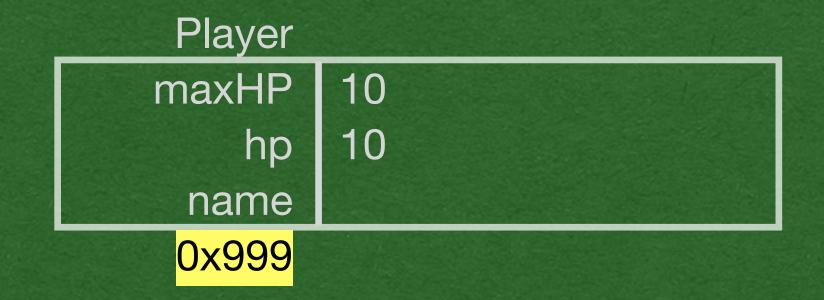
in/out

Notation Note:

- I'll gray out a stack frame that is removed from the stack
- This will have the same meaning as crossing it out
- Makes the variables readable

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
          else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
  public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



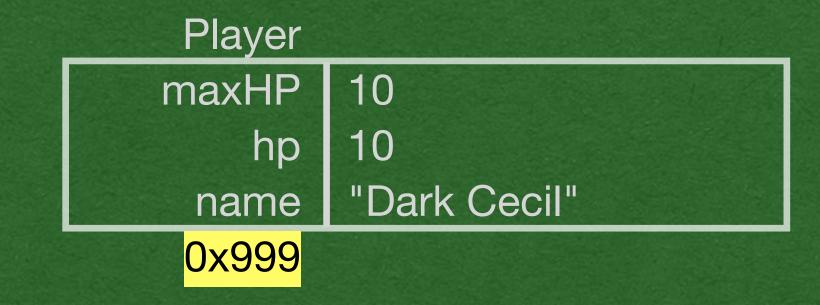


in/out

 Calling setHP will set the hp variable on the stack for this object

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
  public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```

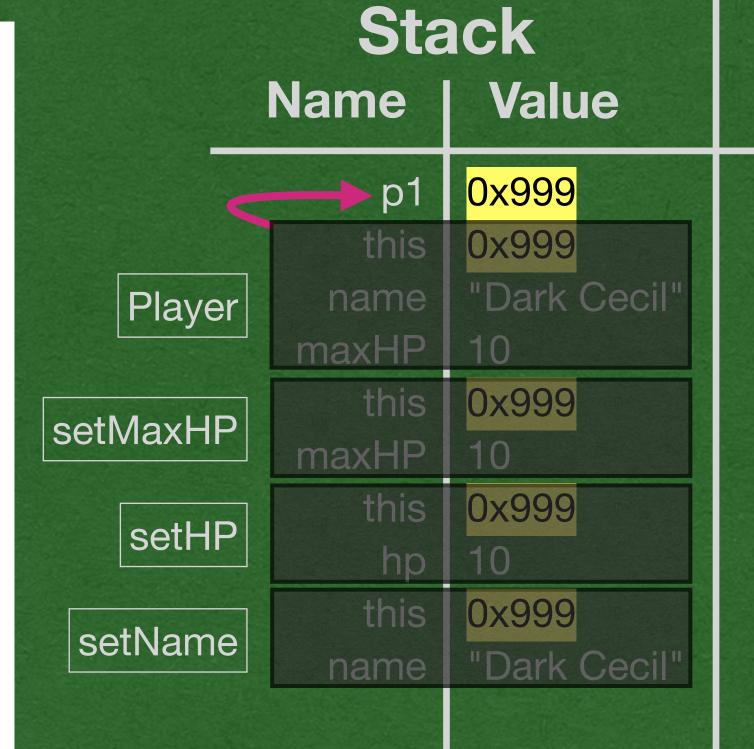




in/out

 Repeat the process for name

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this.hp = this.maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
  public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



```
Player

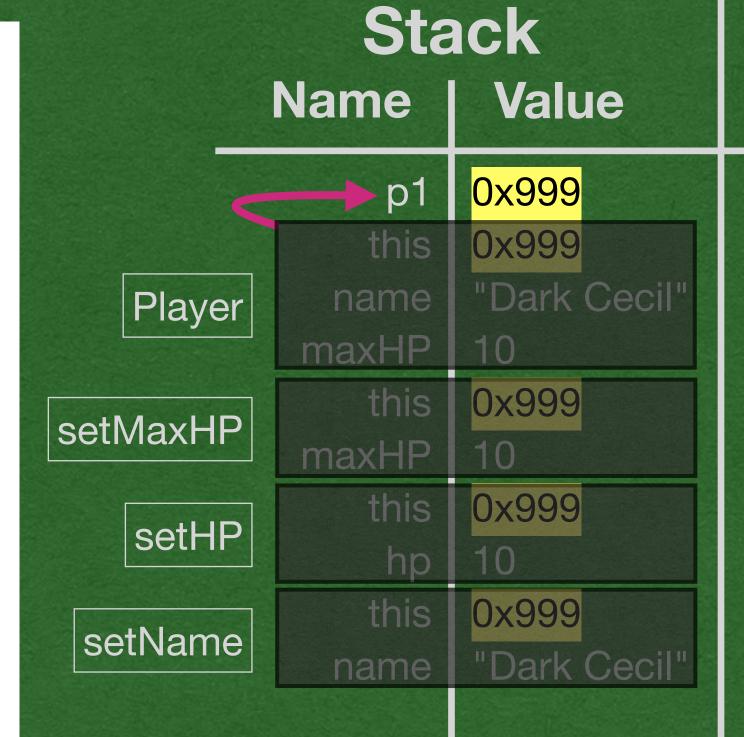
maxHP 10
hp 10
name "Dark Cecil"

0x999
```

in/out

 Constructor method calls return a reference to the object that was created

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



```
Player

maxHP 10

hp 10

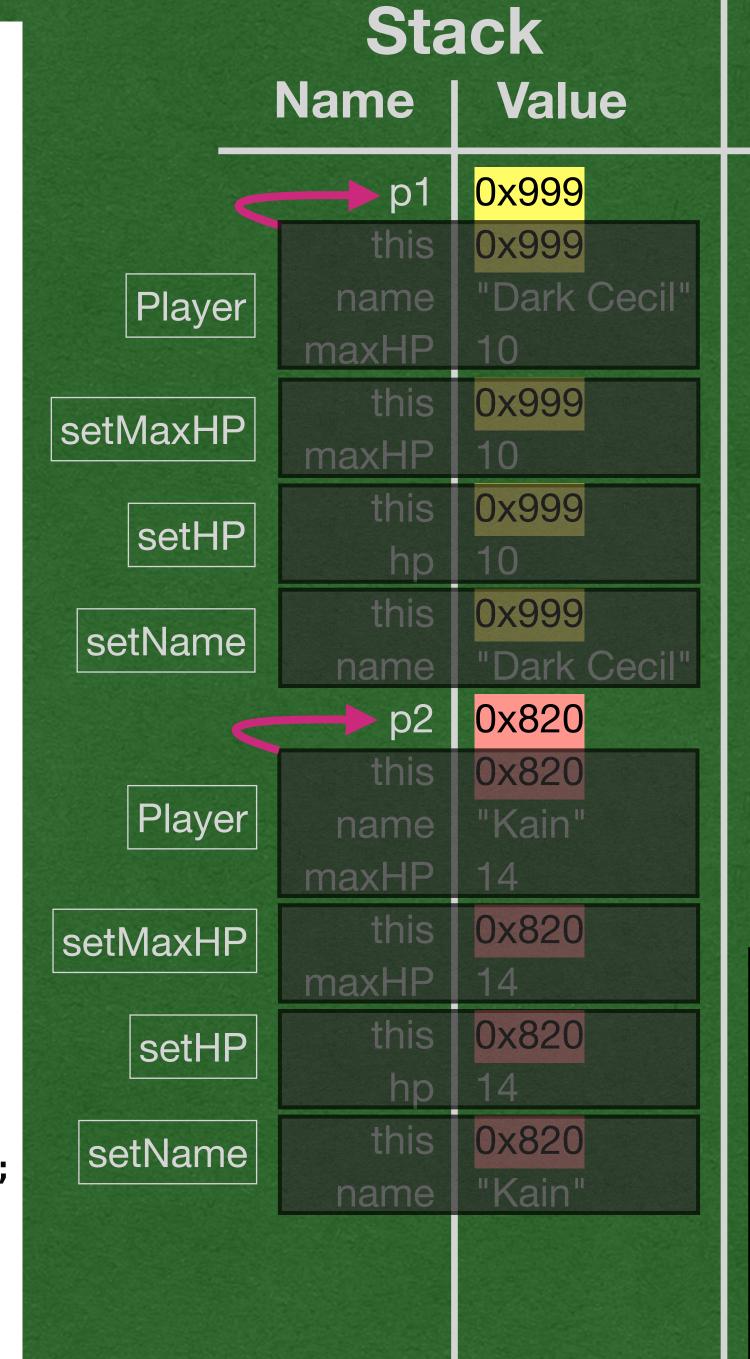
name "Dark Cecil"

0x999
```

in/out

 What happens on the line that initializes p2?

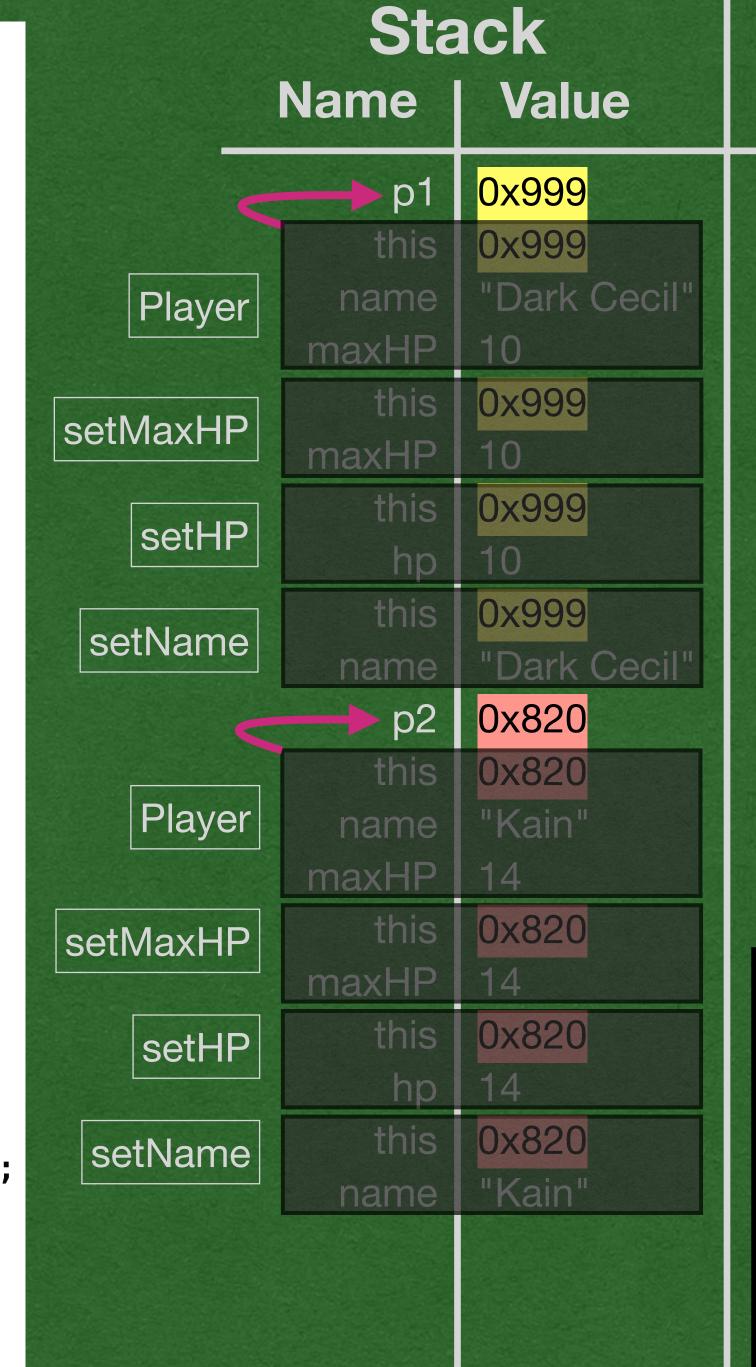
```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```

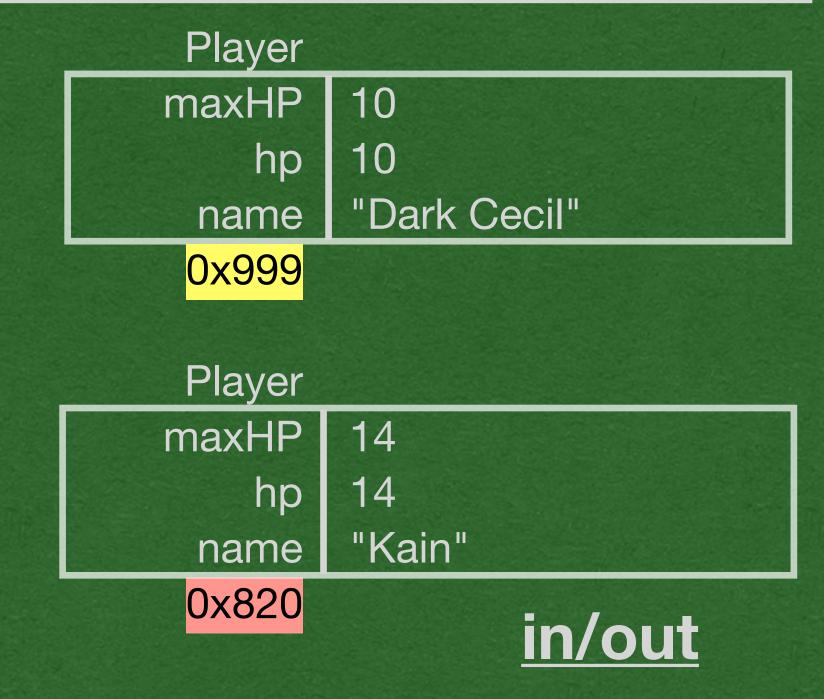




- Whenever you see *new*, a new object is created on the heap
- We have 2 objects of type Player
 - Each object has its own copies of each instance variable

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```





 What happens on the line that initializes p3?

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



```
Player
maxHP
        10
    hp
        "Dark Cecil"
 name
 0x999
 Player
maxHP
         14
         14
    hp
        "Kain"
 name
 0x820
                in/out
```

- If you don't see new, no object is created
- Assign p3 the same reference stored in p1
- Still only 2 objects on the heap

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



```
Player
maxHP
         10
    hp
         "Dark Cecil" "Paladin"
 name
 0x999
 Player
maxHP
         14
         14
    hp
         "Kain"
 name
 0x820
                 in/out
```

- setName is called from p1 which stores 0x999
- this is assigned 0x999

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
   public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```



```
Player
maxHP
        10
    hp
         "Dark Cecil" "Paladin"
 name
 0x999
 Player
maxHP
         14
         14
    hp
         "Kain"
 name
 0x820
                 in/out
```

- getName is called from p3 which stores 0x999
- this is assigned 0x999

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```





- p1 and p3 *refer* to the same object
- Any change made using one variable, affects both variables!

```
public class Player {
    private int maxHP;
    private int hp;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public void setMaxHP(int maxHP) {
        this.maxHP = maxHP;
    public void setHP(int hp) {
        if (hp <= this.maxHP) {</pre>
            this.hp = hp;
        } else {
            this hp = this maxHP;
    public String getName() {
        return this.name;
    public void setName(String name) {
        this.name = name;
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        Player p2 = new Player("Kain", 14);
        Player p3 = p1;
        p1.setName("Paladin");
        System.out.println(p3.getName());
```





End Program

Stack Memory

- Only primitive types are stored directly on the stack
 - double
 - int
 - char
 - boolean
 - String*
 - Double/Integer/Character/Boolean*
- Everything else is stored on the heap with only their references stored on the stack**
 - This includes every object created from a class that you wrote

*Strings and boxed types are actually more complex, but we will treat them as though they are on the stack in this course because they behave exactly as a value on the stack

**Stack and heap allocations vary by compiler and JVM implementations. With modern optimizations, we can never be sure where our values will be stored We'll use this simplified view so we can move on and learn Computer Science

```
package week3;
public class Player {
    private int maxHP;
    private int hp;
    private int attackPower = 4;
    private String name;
    public Player(String name, int maxHP) {
       this.setMaxHP(maxHP);
       this.setHP(maxHP);
        this.setName(name);
 /** Getters and Setters removed for slide **/
    public static void main(String[] args) {
       Player p1 = new Player("Dark Cecil", 10);
        System.out.println(p1)
```

week3.Player@279f2327

toString

- When you print one of your objects to the screen
 - It prints garbage
- Fully qualified name
- @
- "random" hex value
- Almost always not what you want

```
package week3;
public class Player {
    private int maxHP;
    private int hp;
    private int attackPower = 4;
    private String name;
    public Player(String name, int maxHP) {
        this.setMaxHP(maxHP);
        this.setHP(maxHP);
        this.setName(name);
    public String toString() {
        String out = "health: " + this.hp + "/";
        out += this.maxHP;
        return out;
  /** Getters and Setters removed for slide **/
    public static void main(String[] args) {
        Player p1 = new Player("Dark Cecil", 10);
        System.out.println(p1);
```

health: 10/10

toString

- If we write a special method named "toString" that returns a String
 - This method will be called when we print an object of this type

```
public class Player {
   private int maxHP;
   private int hp;
   private int attackPower = 4;
   private String name;
    public Player(String name, int maxHP) {
       this.setMaxHP(maxHP);
       this.setHP(maxHP);
       this.setName(name);
    public String toString() {
       String out = "health: " + this.hp + "/";
       out += this.maxHP;
       return out;
    public void takeDamage(int damage) {
       this.hp -= damage;
   public void attack(Player otherPlayer) {
       otherPlayer.takeDamage(this.attackPower);
 /** Getters and Setters removed for slide **/
   public static void main(String[] args) {
       Player p1 = new Player("Dark Cecil", 10);
       Player p2 = new Player("Kain", 14);
       Player p3 = p2;
        p1.attack(p2);
        p1.attack(p2);
```

Types

- Classes define types
- Now that we have a Player type, we can use it wherever we need a type
- Here, we use Player as the type of a method parameter

Party Example

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
public class Party {
    private ArrayList<Character> members;
    private int battlesWon = 0;
    public Party() {
       this.members = new ArrayList<>();
    public void addCharacter(Character member) {
        this.members.add(member);
    public void winBattle(int xp) {
        this.battlesWon++;
        for (int x=0; x < this.members.size(); x++) {</pre>
            this.members.get(x).winBattle(xp);
    public static void main(String[] args) {
        Character hero = new Character();
        hero = new Character();
        Character fighter = new Character();
        hero.winBattle(10);
        Party party = new Party();
        party.addCharacter(hero);
        party.addCharacter(fighter);
        party.winBattle(20);
```

Party

- Let's look at another example
- We'll highlight 2 new concepts with this code:
 - Composition
 - Garbage Collection

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
public class Party {
   private ArrayList<Character> members;
    private int battlesWon = 0;
    public Party() {
       this.members = new ArrayList<>();
    public void addCharacter(Character member) {
        this.members.add(member);
    public void winBattle(int xp) {
        this.battlesWon++;
        for (int x=0; x < this.members.size(); x++) {</pre>
            this.members.get(x).winBattle(xp);
    public static void main(String[] args) {
        Character hero = new Character();
        hero = new Character();
        Character fighter = new Character();
        hero.winBattle(10);
        Party party = new Party();
        party.addCharacter(hero);
        party.addCharacter(fighter);
        party.winBattle(20);
```

Composition

- Instance variables of objects can store references to the objects
- The Party class is composed of an ArrayList of Characters

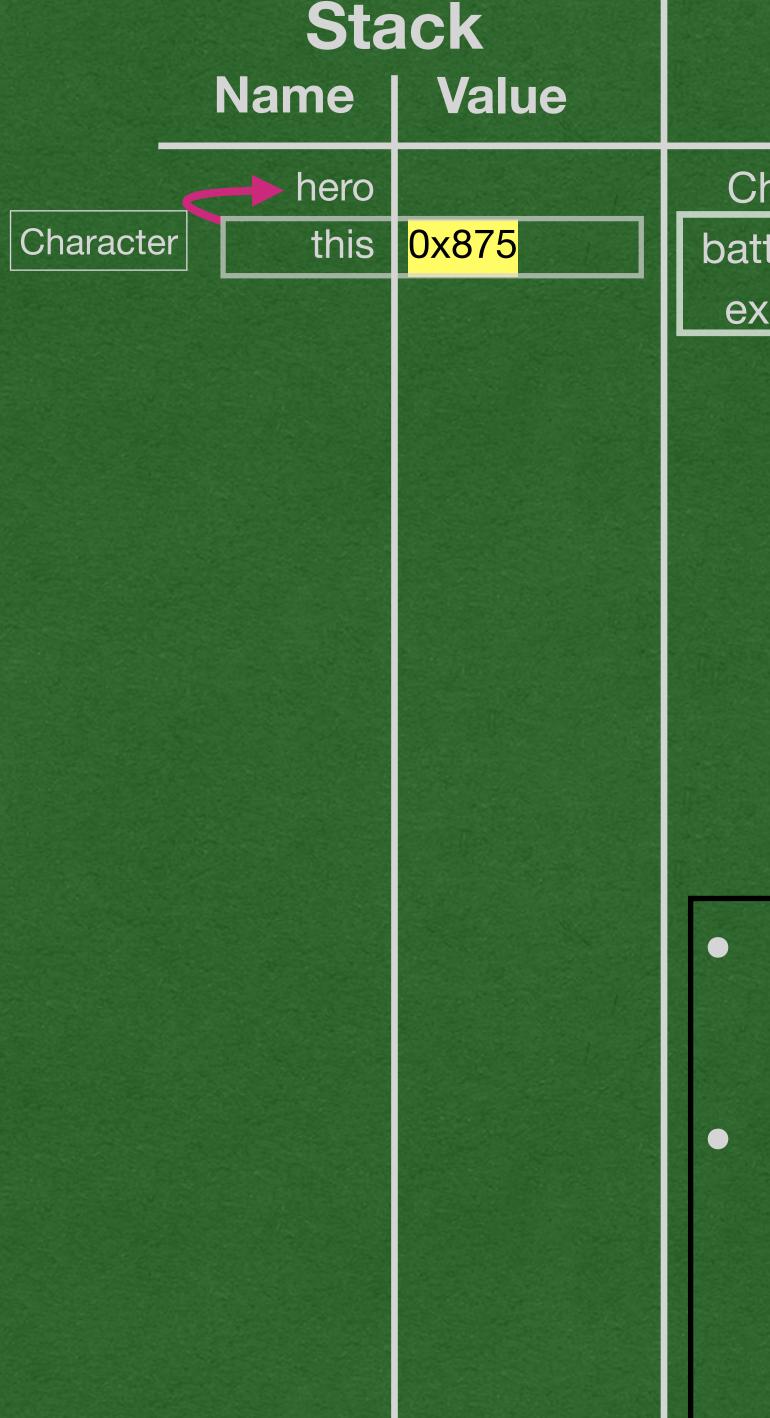
```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
public class Party {
    private ArrayList<Character> members;
    private int battlesWon = 0;
    public Party() {
       this.members = new ArrayList<>();
    public void addCharacter(Character member) {
        this.members.add(member);
    public void winBattle(int xp) {
        this.battlesWon++;
        for (int x=0; x < this.members.size(); x++) {</pre>
            this.members.get(x).winBattle(xp);
    public static void main(String[] args) {
        Character hero = new Character();
       hero = new Character();
        Character fighter = new Character();
        hero.winBattle(10);
        Party party = new Party();
        party.addCharacter(hero);
        party.addCharacter(fighter);
        party.winBattle(20);
```

Garbage Collection

- hero is assigned a reference to an instance/object of type Character
- hero is then reassigned to a new reference
- We no longer have a reference to the first Character object in memory
 - Since we cannot access this object, it will be removed from memory

Memory Diagram

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
  public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
    this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
    hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





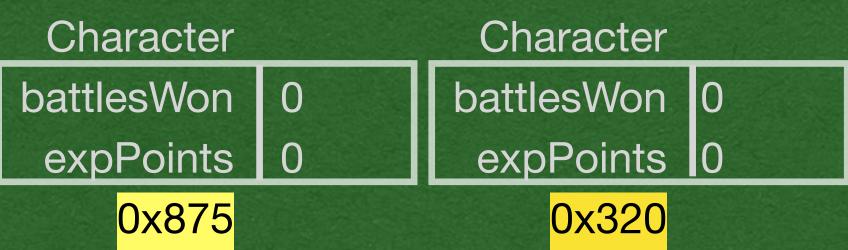
- Start by creating a new
 Character object
- The constructor is empty, but we still create the instance variables

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;

    public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
    }
}
```

```
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
  public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
    this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
  public static void main(String[] args) {
    Character hero = new Character();
   hero = new Character();
    Character fighter = new Character();
    hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





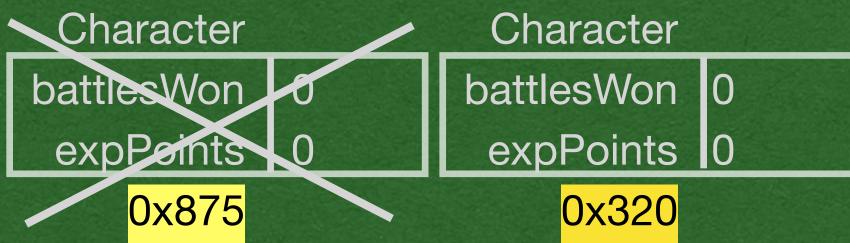
- We immediately replace the reference stored in hero with a reference to a new object
- *You wouldn't actually do this. This is only for the example

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;

    public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
    }
}
```

```
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
  public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
    this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
  public static void main(String[] args) {
    Character hero = new Character();
   hero = new Character();
    Character fighter = new Character();
    hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





in/out

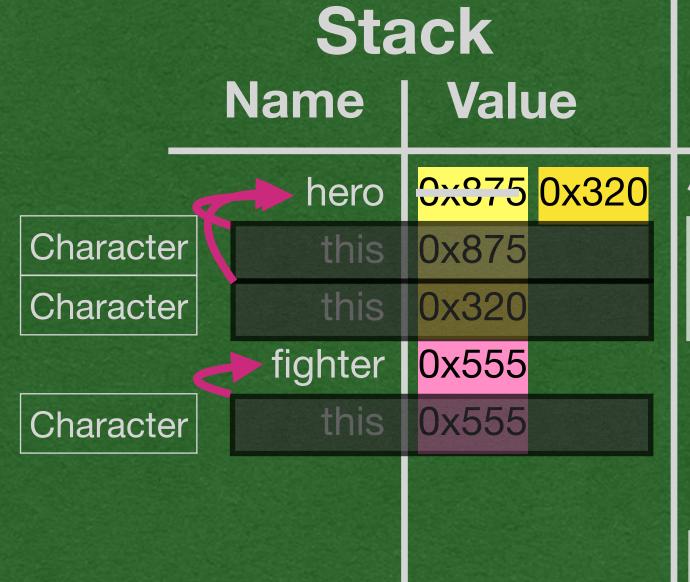
Garbage Collection

- When we no longer have access to a reference to an object on the heap, it is deleted from memory
- Java does this in the background as we reassign variables

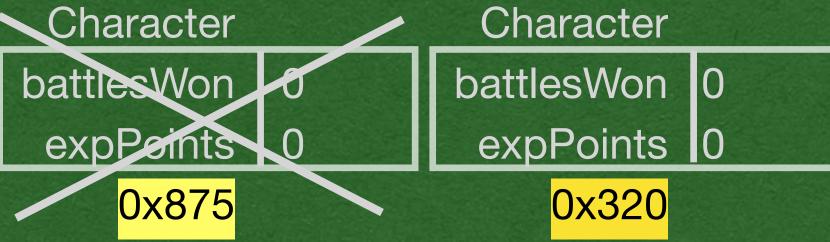
```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;

public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
    }
}
```

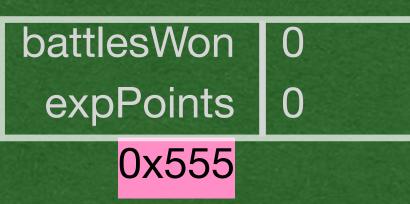
```
public class Party {
 private ArrayList<Character> members;
  private int battlesWon = 0;
 public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
   this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
 public static void main(String[] args) {
    Character hero = new Character();
   hero = new Character();
   Character fighter = new Character();
   hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
   party.winBattle(20);
```







Character

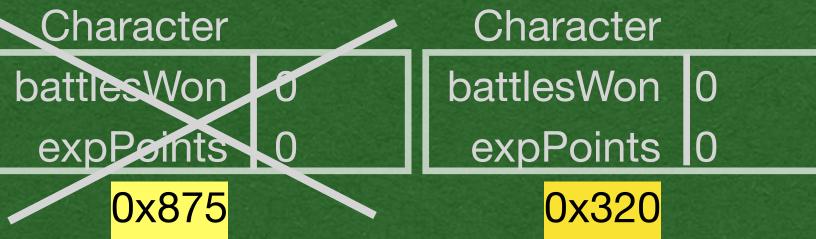


in/out

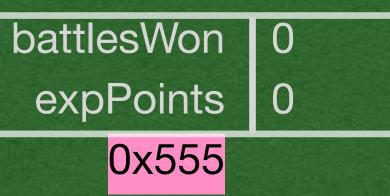
 Create another instance of the Character class (Object of type character)

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
  public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
    this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
   hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





Character

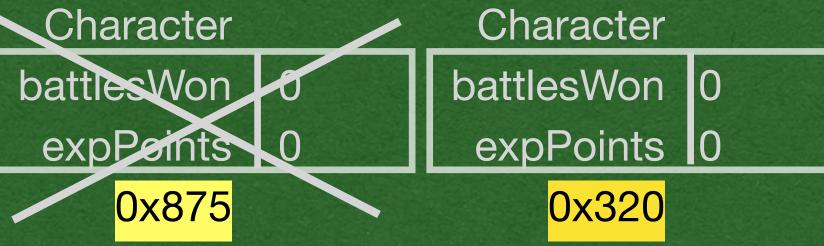


in/out

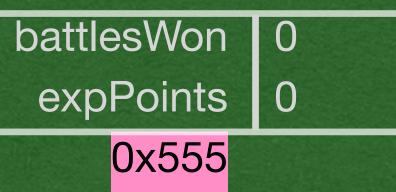
- When we call a method:
 - this is added to the stack frame
- this stores reference to the calling object

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
  public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
    this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
   hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





Character



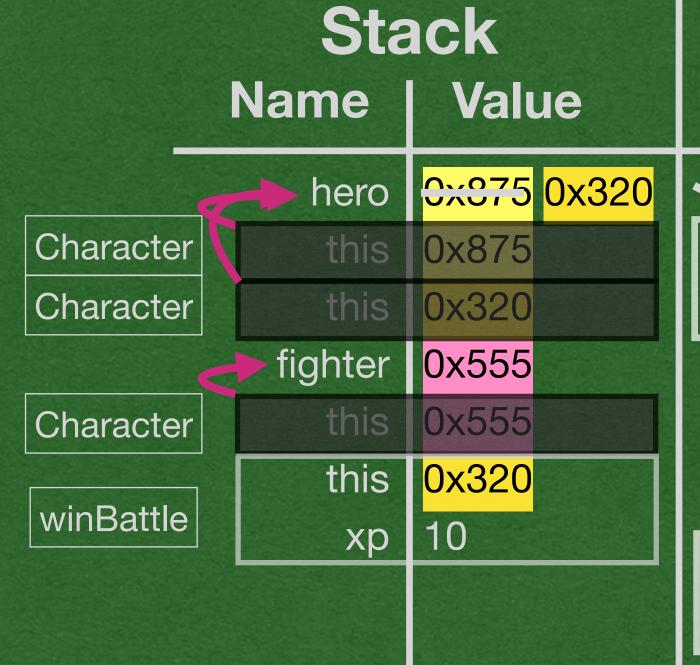
in/out

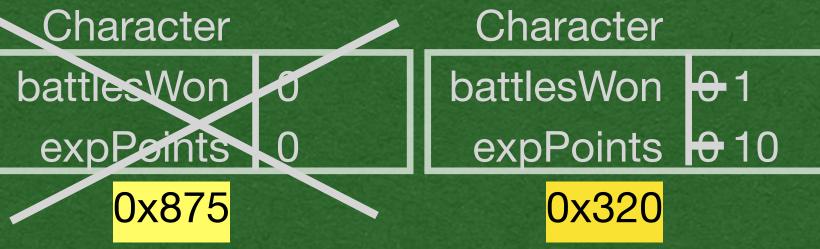
- winBattle was called by hero
- hero stores the reference 0x320
- this in the stack frame is 0x320

```
public class Character {
   private int battlesWon = 0;
   private int expPoints = 0;

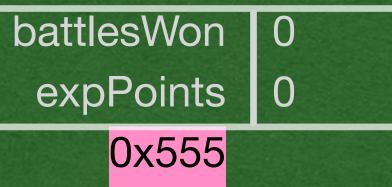
public Character() {}
   public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
}
```

```
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
  public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
    this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





Character

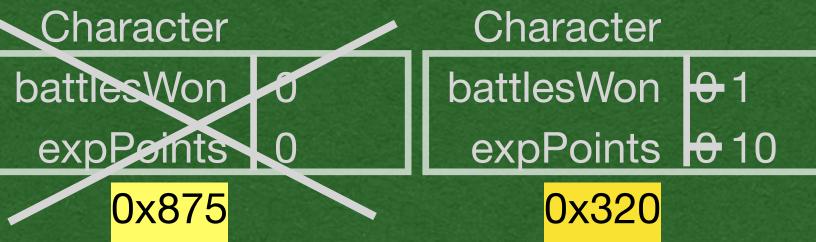


in/out

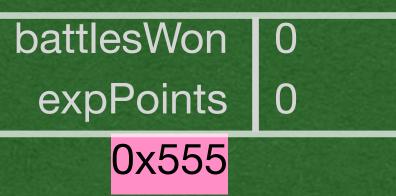
 Follow the reference stored in this to update the instance variables

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
    public void winBattle(int xp) {
        this.battlesWon++;
        this.expPoints += xp;
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
  public Party() {
   this.members = new ArrayList<>();
  public void addCharacter(Character member) {
    this.members.add(member);
  public void winBattle(int xp) {
    this battlesWon++;
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
hero.winBattle(10);
 Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





Character



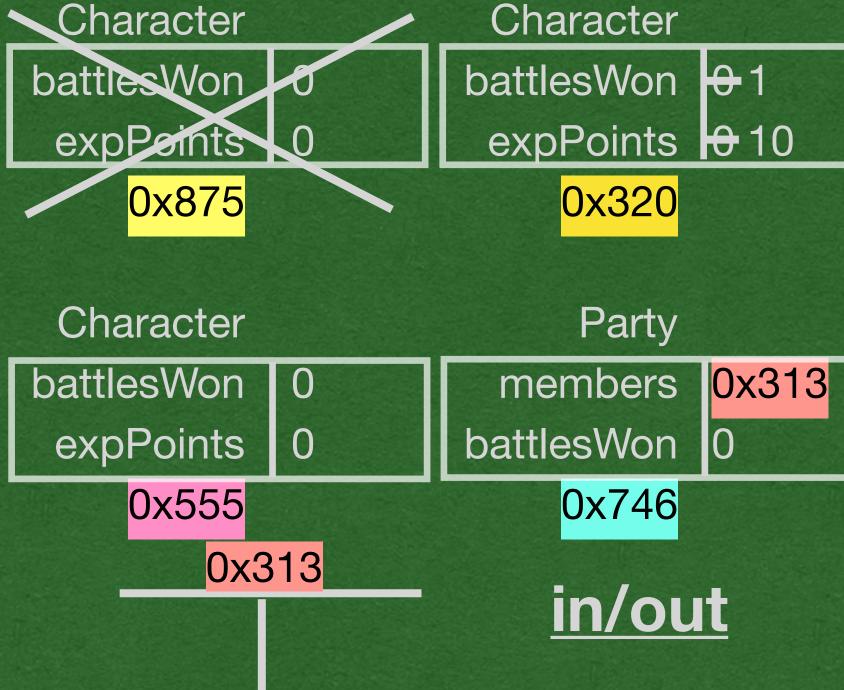
in/out

- This method's got sideeffects!
- winBattle returns void
- winBattle was called to change the state of an object on the heap

```
public class Character {
     private int battlesWon = 0;
     private int expPoints = 0;
     public Character() {}
     public void winBattle(int xp) {
         this.battlesWon++;
         this.expPoints += xp;
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
   public Party() {
this members = new ArrayList<>();
  public void addCharacter(Character member) {
     this.members.add(member);
   public void winBattle(int xp) {
    this battlesWon++;
     for (int x=0; x < this.members.size(); x++) {</pre>
       this.members.get(x).winBattle(xp);
   public static void main(String[] args) {
     Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
    hero.winBattle(10);
Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```

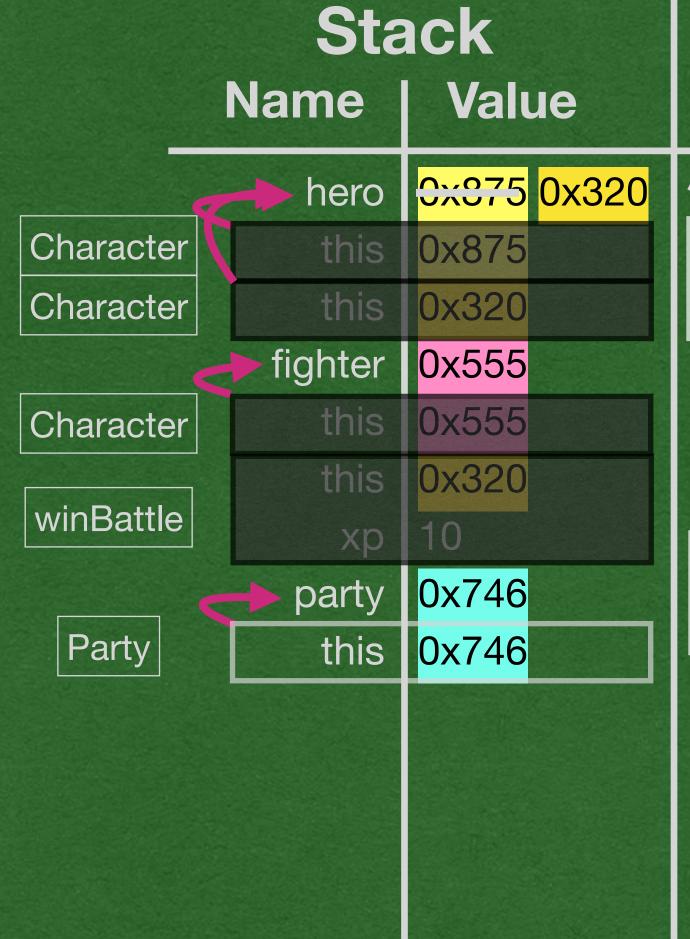


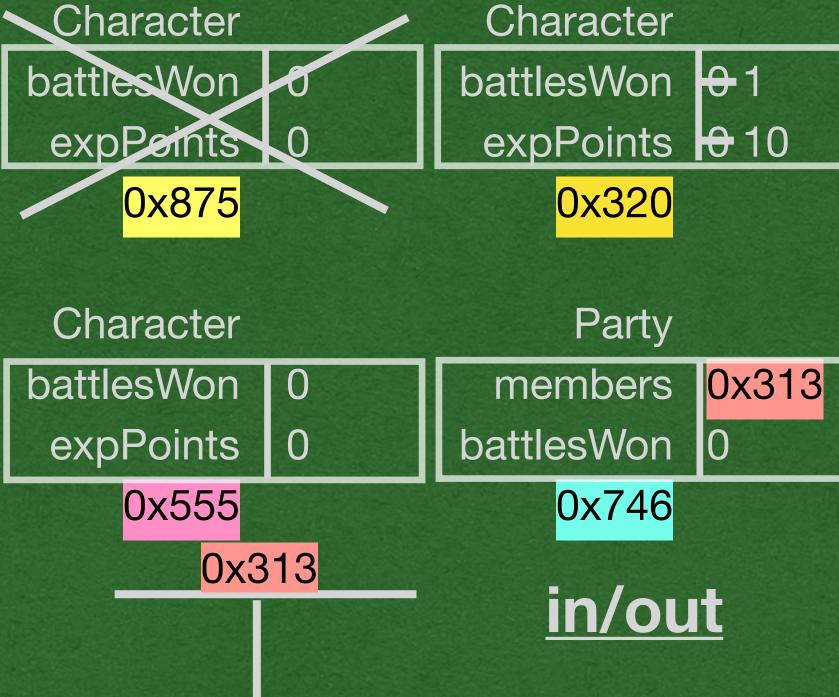




- When we create a new Party, we also create a new ArrayList via composition
- Both objects are created on the heap

```
public class Character {
     private int battlesWon = 0;
     private int expPoints = 0;
     public Character() {}
     public void winBattle(int xp) {
         this.battlesWon++;
         this.expPoints += xp;
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
   public Party() {
this members = new ArrayList<>();
  public void addCharacter(Character member) {
     this.members.add(member);
   public void winBattle(int xp) {
     this.battlesWon++;
     for (int x=0; x < this.members.size(); x++) {</pre>
       this.members.get(x).winBattle(xp);
   public static void main(String[] args) {
     Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
    hero.winBattle(10);
Party party = new Party();
    party.addCharacter(hero);
     party.addCharacter(fighter);
    party.winBattle(20);
```





- The is a stack frame created for the ArrayList constructor
- When the code is part of Java, and not our code, we don't add it to the memory diagram

```
Stack
public class Character {
                                                                                                    Heap
    private int battlesWon = 0;
                                                                       Value
                                                            Name |
   private int expPoints = 0;
    public Character() {}
                                                                hero 0x375 0x320
                                                                                     Character
                                                                                                           Character
    public void winBattle(int xp) {
                                                  Character
                                                                 this | 0x875
                                                                                    battles Won 0
                                                                                                          battlesWon 0-1
       this.battlesWon++;
       this.expPoints += xp;
                                                  Character
                                                                 this 0x320
                                                                                                           expPoints 0 10
                                                                                     expPeints 0
                                                              fighter 0x555
                                                                                         0x875
                                                                                                               0x320
                                                                 this 0x555
                                                  Character
public class Party {
 private ArrayList<Character> members;
                                                                 this 0x320
                                                                                      Character
                                                                                                                Party
 private int battlesWon = 0;
                                                  winBattle
                                                                                    battlesWon 0
                                                                                                            members 0x313
 public Party() {
                                                               party 0x746
  this.members = new ArrayList<>();
                                                                                                          battlesWon 0
                                                                                      expPoints
                                                    Party
                                                                 this | 0x746
                                                                                         0x555
                                                                                                               0x746
 public void addCharacter(Character member) {
                                                                 this 0x746
   this.members.add(member);
                                                addCharacter
                                                                                             0x313
                                                             member 0x320
                                                                                                              in/out
                                                                                                 0x320
 public void winBattle(int xp) {
   this battlesWon++;
   for (int x=0; x < this.members.size(); x++) {</pre>
     this.members.get(x).winBattle(xp);

    addCharacter takes a reference

                                                                                       to a Character as a parameter
 public static void main(String[] args) {
   Character hero = new Character();
                                                                                      dot means "follow the
   hero = new Character();
   Character fighter = new Character();
                                                                                       reference"
   hero.winBattle(10);
   Party party = new Party();
party.addCharacter(hero);

    We follow 2 references in one

    party.addCharacter(fighter);
                                                                                       line to get to the ArrayList in
   party.winBattle(20);
                                                                                       memory
```

```
Stack
public class Character {
                                                                                                    Heap
    private int battlesWon = 0;
                                                            Name I
                                                                       Value
   private int expPoints = 0;
   public Character() {}
                                                                hero 0x375 0x320
                                                                                     Character
                                                                                                            Character
    public void winBattle(int xp) {
                                                  Character
                                                                 this | 0x875
                                                                                     battlesWon 0
                                                                                                          battlesWon 0-1
       this.battlesWon++;
       this.expPoints += xp;
                                                                 this | 0x320
                                                  Character
                                                                                                            expPoints 0 10
                                                                                      expPeints 0
                                                               fighter 0x555
                                                                                          0x875
                                                                                                               0x320
                                                                 this 0x555
                                                  Character
public class Party {
 private ArrayList<Character> members;
                                                                 this 0x320
                                                                                      Character
                                                                                                                Party
  private int battlesWon = 0;
                                                  winBattle
                                                                                     battlesWon 0
                                                                                                            members 0x313
 public Party() {
                                                               party 0x746
  this.members = new ArrayList<>();
                                                                                                          battlesWon <del>0</del> 1
                                                                                      expPoints
                                                    Party
                                                                 this | 0x746
                                                                                         0x555
                                                                                                               0x746
  public void addCharacter(Character member) {
                                                                 this | 0x746
   this.members.add(member);
                                                addCharacter
                                                                                             0x313
                                                             member
                                                                      0x320
                                                                                                               in/out
                                                                                                 0x320
 public void winBattle(int xp) {
                                                                 this | 0x746
   this battlesWon++;
                                                addCharacter
                                                                                                 0x555
   for (int x=0; x < this.members.size(); x++) {</pre>
                                                                      0x555
                                                             member
     this.members.get(x).winBattle(xp);
                                                                 this 0x746

    Check out this line!

                                                  winBattle
                                                                  xp 20
  public static void main(String[] args) {
                                                                                        dot means follow the
   Character hero = new Character();
   hero = new Character();
                                                                                        reference
   Character fighter = new Character();
   hero.winBattle(10);
   Party party = new Party();
                                                                                       We'll jump around in
   party.addCharacter(hero);
   party.addCharacter(fighter);
                                                                                        memory 3 times to reach
    party.winBattle(20);
                                                                                        the Player at 0x320
```

```
Stack
public class Character {
    private int battlesWon = 0;
                                                                           Value
                                                               Name |
    private int expPoints = 0;
    public Character() {}
                                                                    hero 0x375 0x320
    public void winBattle(int xp) {
                                                     Character
                                                                     this | 0x875
        this battlesWon++;
        this.expPoints += xp;
                                                     Character
                                                                     this | 0x320
                                                                  fighter 0x555
                                                                     this 0x555
                                                     Character
public class Party {
  private ArrayList<Character> members;
                                                                    this 0x320
  private int battlesWon = 0;
                                                     winBattle
  public Party() {
                                                                   party 0x746
   this.members = new ArrayList<>();
                                                       Party
                                                                     this | 0x746
  public void addCharacter(Character member) {
                                                                    this | 0x746
    this.members.add(member);
                                                   addCharacter
                                                                member
                                                                         0x320
  public void winBattle(int xp) {
                                                                    this | 0x746
    this battlesWon++;
                                                   addCharacter
    for (int x=0; x < this.members.size(); x++) {</pre>
                                                                         0x555
                                                                 member
     this.members.get(x).winBattle(xp);
                                                                    this 0x746
                                                     winBattle
                                                                     xp 20
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
                                                                    this 0x320
                                                     winBattle
    Character fighter = new Character();
                                                                     xp 20
    hero.winBattle(10);
    Party party = new Party();
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```



- Since we were at 0x320 in memory when this method is called
 - 0x320 is the calling object
 - this stores 0x320

```
Stack
public class Character {
    private int battlesWon = 0;
                                                                Name |
    private int expPoints = 0;
    public Character() {}
    public void winBattle(int xp) {
                                                     Character
        this battlesWon++;
        this.expPoints += xp;
                                                     Character
                                                     Character
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
                                                     winBattle
  public Party() {
   this.members = new ArrayList<>();
                                                       Party
  public void addCharacter(Character member) {
    this.members.add(member);
                                                   addCharacter
                                                                 member
  public void winBattle(int xp) {
    this battlesWon++;
                                                   addCharacter
    for (int x=0; x < this.members.size(); x++) {</pre>
                                                                 member
     this.members.get(x).winBattle(xp);
                                                     winBattle
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
    Character fighter = new Character();
                                                     winBattle
    hero.winBattle(10);
```

Party party = new Party();

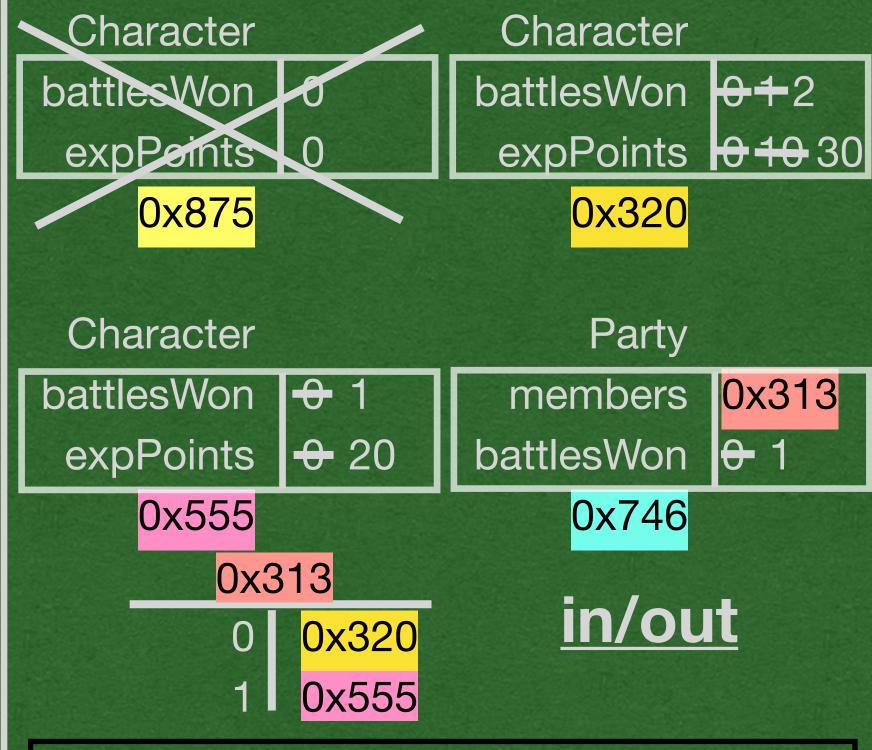
party.addCharacter(fighter);

party.addCharacter(hero);

party.winBattle(20);







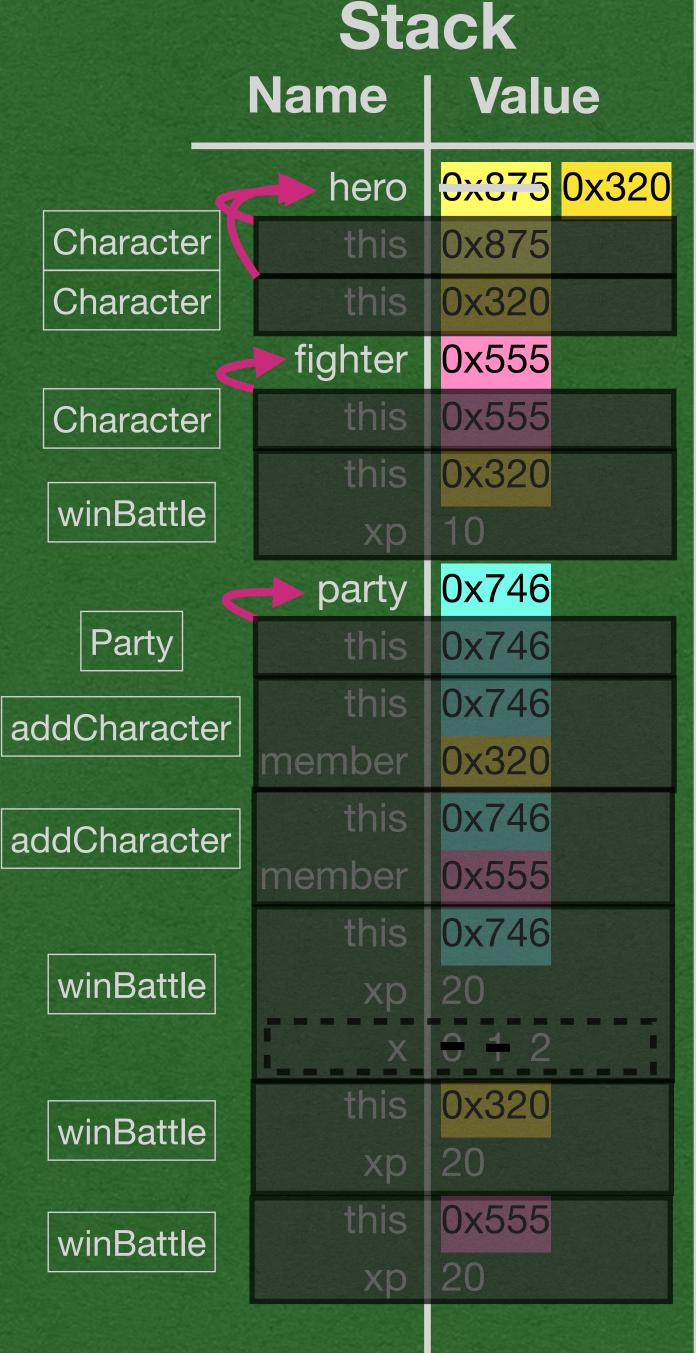
Repeat the process with 0x555

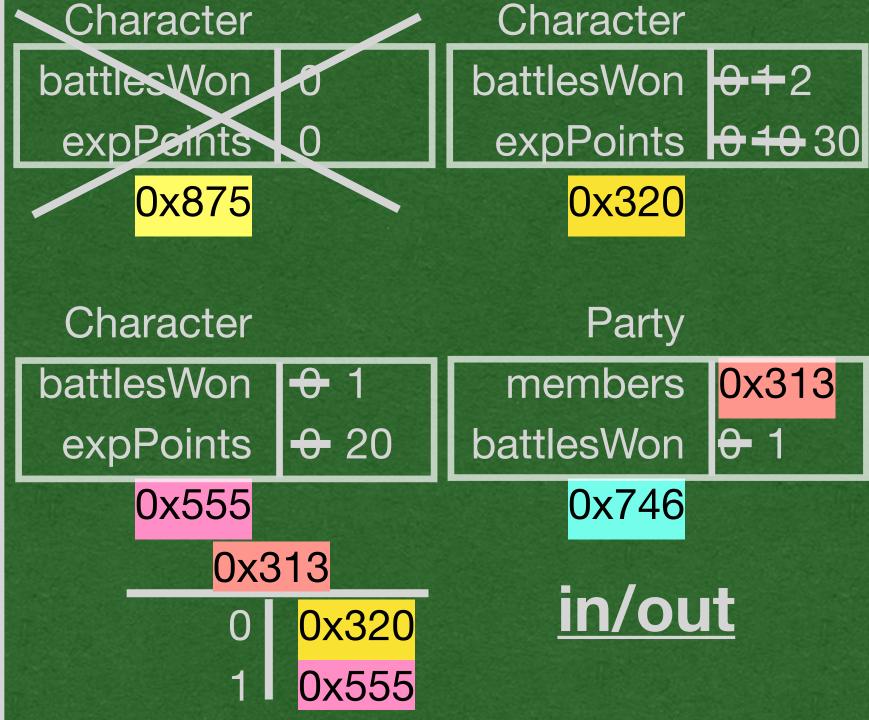
```
Stack
public class Character {
                                                                                                    Heap
    private int battlesWon = 0;
                                                                       Value
                                                            Name |
   private int expPoints = 0;
    public Character() {}
                                                                hero 0x375 0x320
                                                                                     Character
                                                                                                           Character
    public void winBattle(int xp) {
                                                  Character
                                                                 this | 0x875
                                                                                    battles Won 0
                                                                                                          battlesWon <del>0 1</del>2
       this.battlesWon++;
       this.expPoints += xp;
                                                  Character
                                                                 this
                                                                     0x320
                                                                                                           expPoints 0 10 30
                                                                                     expPeints 0
                                                              fighter 0x555
                                                                                         0x875
                                                                                                               0x320
                                                                 this 0x555
                                                  Character
public class Party {
 private ArrayList<Character> members;
                                                                 this 0x320
                                                                                      Character
                                                                                                               Party
 private int battlesWon = 0;
                                                  winBattle
                                                                                    battlesWon
                                                                                                +
                                                                                                            members 0x313
 public Party() {
                                                               party 0x746
  this.members = new ArrayList<>();
                                                                                     expPoints + 20
                                                                                                          battlesWon 0 1
                                                    Party
                                                                 this | 0x746
                                                                                         0x555
                                                                                                               0x746
 public void addCharacter(Character member) {
                                                                 this | 0x746
   this.members.add(member);
                                                addCharacter
                                                                                             0x313
                                                             member
                                                                     0x320
                                                                                                              in/out
                                                                                                0x320
 public void winBattle(int xp) {
                                                                 this | 0x746
   this battlesWon++;
                                                addCharacter
                                                                                                 0x555
   for (int x=0; x < this.members.size(); x++) {</pre>
                                                                     0x555
                                                             member
     this.members.get(x).winBattle(xp);
                                                                 this 0x746

    The loop ends

                                                  winBattle
                                                                 xp 20
 public static void main(String[] args) {
                                                                                         when x reaches 2
   Character hero = new Character();
   hero = new Character();
                                                                 this | 0x320
                                                  winBattle
   Character fighter = new Character();
   hero.winBattle(10);
                                                                                        x is removed from
   Party party = new Party();
                                                                 this 0x555
                                                  winBattle
   party.addCharacter(hero);
                                                                  xp 20
   party.addCharacter(fighter);
                                                                                         memory
   party.winBattle(20);
```

```
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;
    public Character() {}
    public void winBattle(int xp) {
                                                      Character
        this.battlesWon++;
        this.expPoints += xp;
                                                      Character
                                                      Character
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;
                                                      winBattle
  public Party() {
   this.members = new ArrayList<>();
                                                        Party
  public void addCharacter(Character member) {
    this.members.add(member);
                                                    addCharacter
  public void winBattle(int xp) {
    this battlesWon++;
                                                    addCharacter
    for (int x=0; x < this.members.size(); x++) {</pre>
      this.members.get(x).winBattle(xp);
                                                      winBattle
  public static void main(String[] args) {
    Character hero = new Character();
    hero = new Character();
                                                      winBattle
    Character fighter = new Character();
    hero.winBattle(10);
    Party party = new Party();
                                                      winBattle
    party.addCharacter(hero);
    party.addCharacter(fighter);
    party.winBattle(20);
```





- The method calls end
- The program ends