Party Example

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

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

```java
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++) {
            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 using objects
- We'll highlight 2 new concepts with this code:
  - Composition
  - Garbage Collection

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

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

```java
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++) {
            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 other objects
- The Party class is composed of an ArrayList of Characters

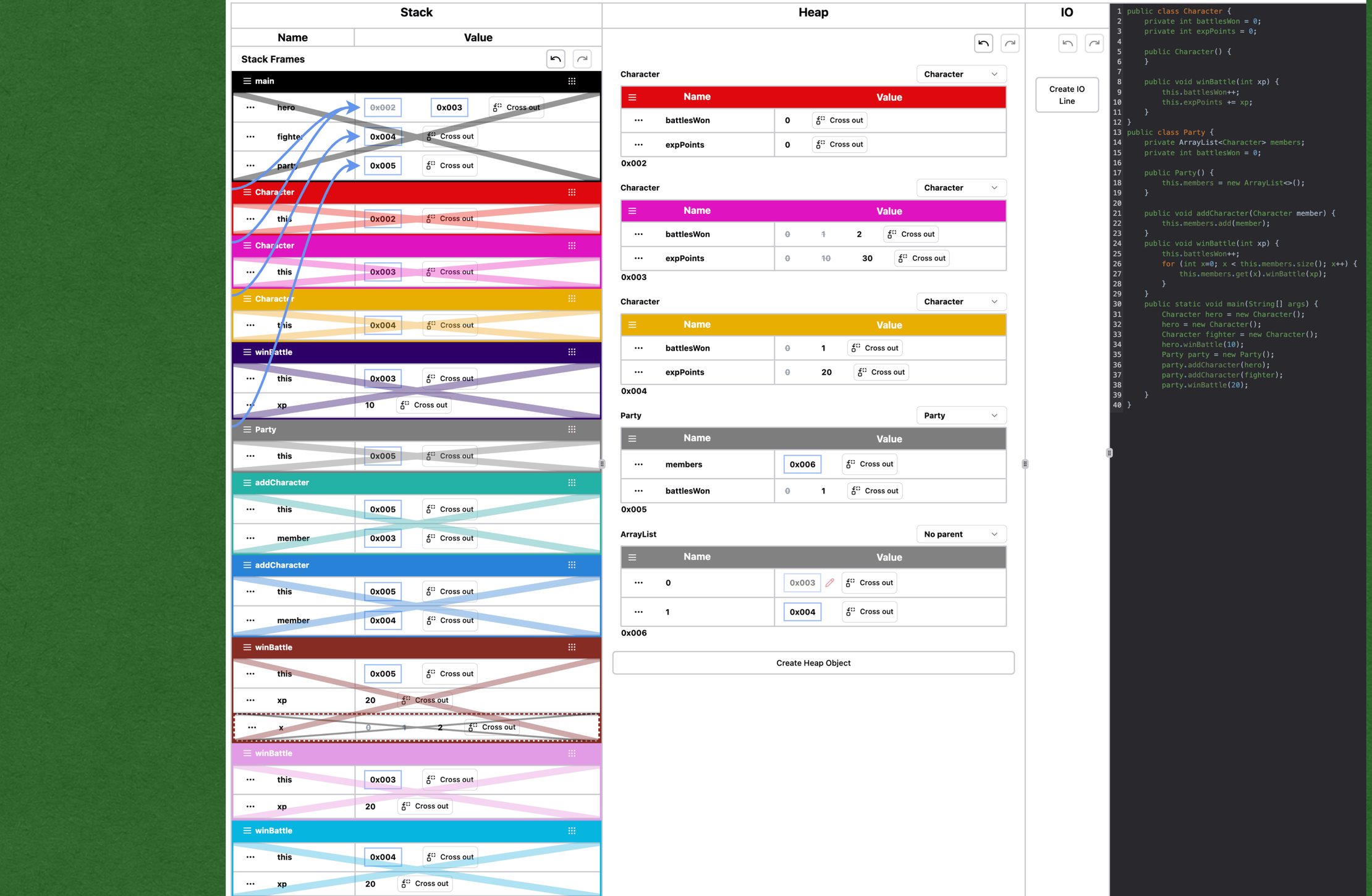
```java
public class Character {
    private int battlesWon = 0;
    private int expPoints = 0;

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

```java
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++) {
            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

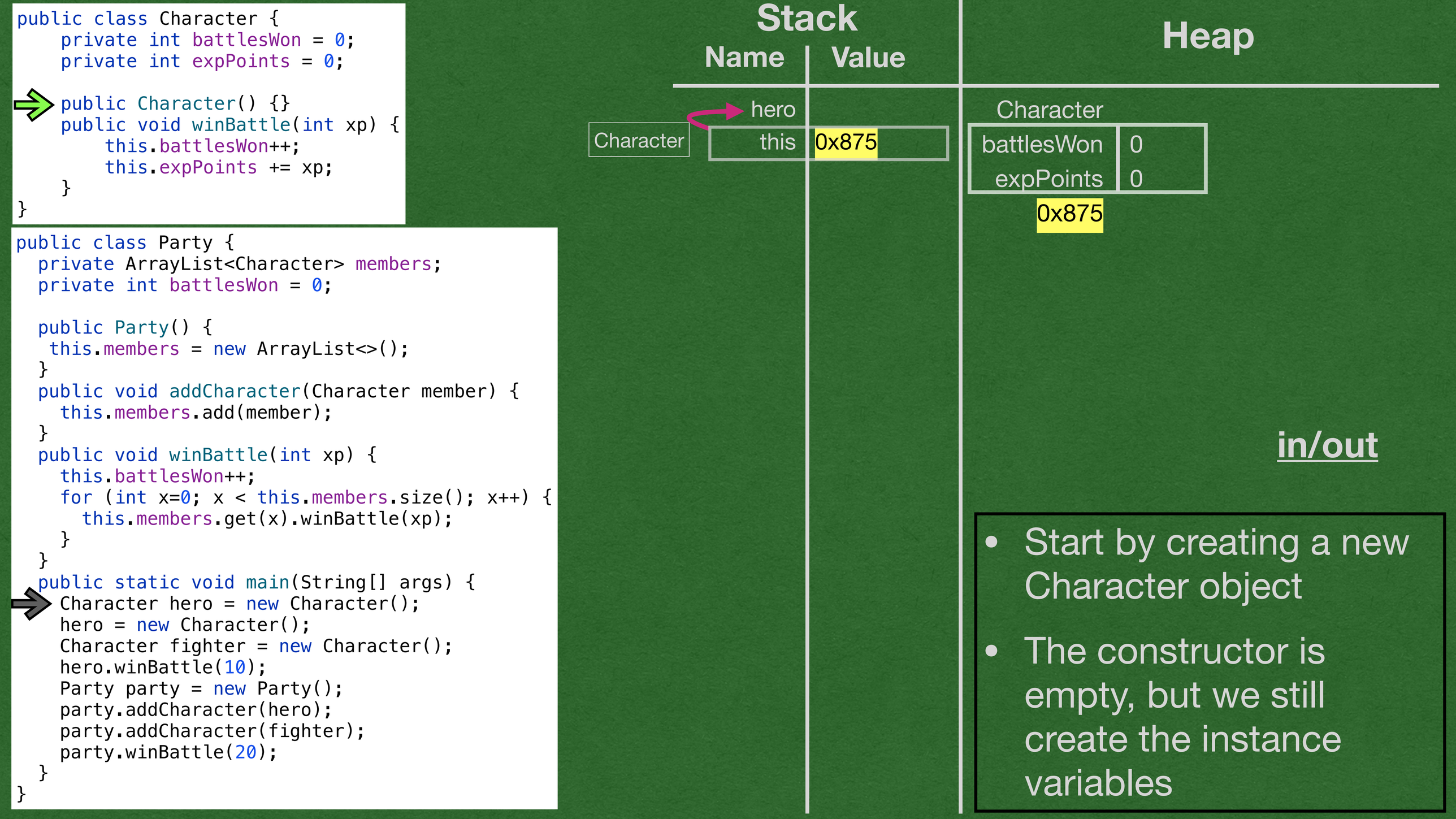
- hero is assigned a reference to an instance/object of type Character
- hero is then reassigned to a new reference
- We no longer have a reference to the first Character object in memory
  - Since we cannot access this object, it will be removed from memory

Memory Diagram


Stack
Name
Value
Stack Frames
main
hero
0x002
0x003
Cross out
fighter
0x004
party
0x005
Character
this
0x002
Character
this
0x003
Character
this
0x004
winBattle
this
0x003
xp
10
Party
this
0x005
addCharacter
this
0x005
member
0x003
addCharacter
this
0x005
member
0x004
winBattle
this
0x005
xp
20
x
0
1
2
winBattle
this
0x003
xp
20
winBattle
this
0x004
xp
20
Heap
Character
Character
Name
Value
battlesWon
0
expPoints
0
0x002
Character
Character
battlesWon
0
1
2
expPoints
0
10
30
0x003
Character
Character
battlesWon
0
1
expPoints
0
20
0x004
Party
Party
members
0x006
battlesWon
0
1
0x005
ArrayList
No parent
0
0x003
1
0x004
0x006
Create Heap Object
IO
Create IO Line
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++) {
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);
}
}

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

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

```java
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++) {
      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

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

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

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

```java
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++) {
      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);
  }
}
```

Stack

Name	Value
hero	~~0x875~~ 0x320
Character this	0x875
Character this	0x320

Heap

Character	
battlesWon	0
expPoints	0

0x875

Character	
battlesWon	0
expPoints	0

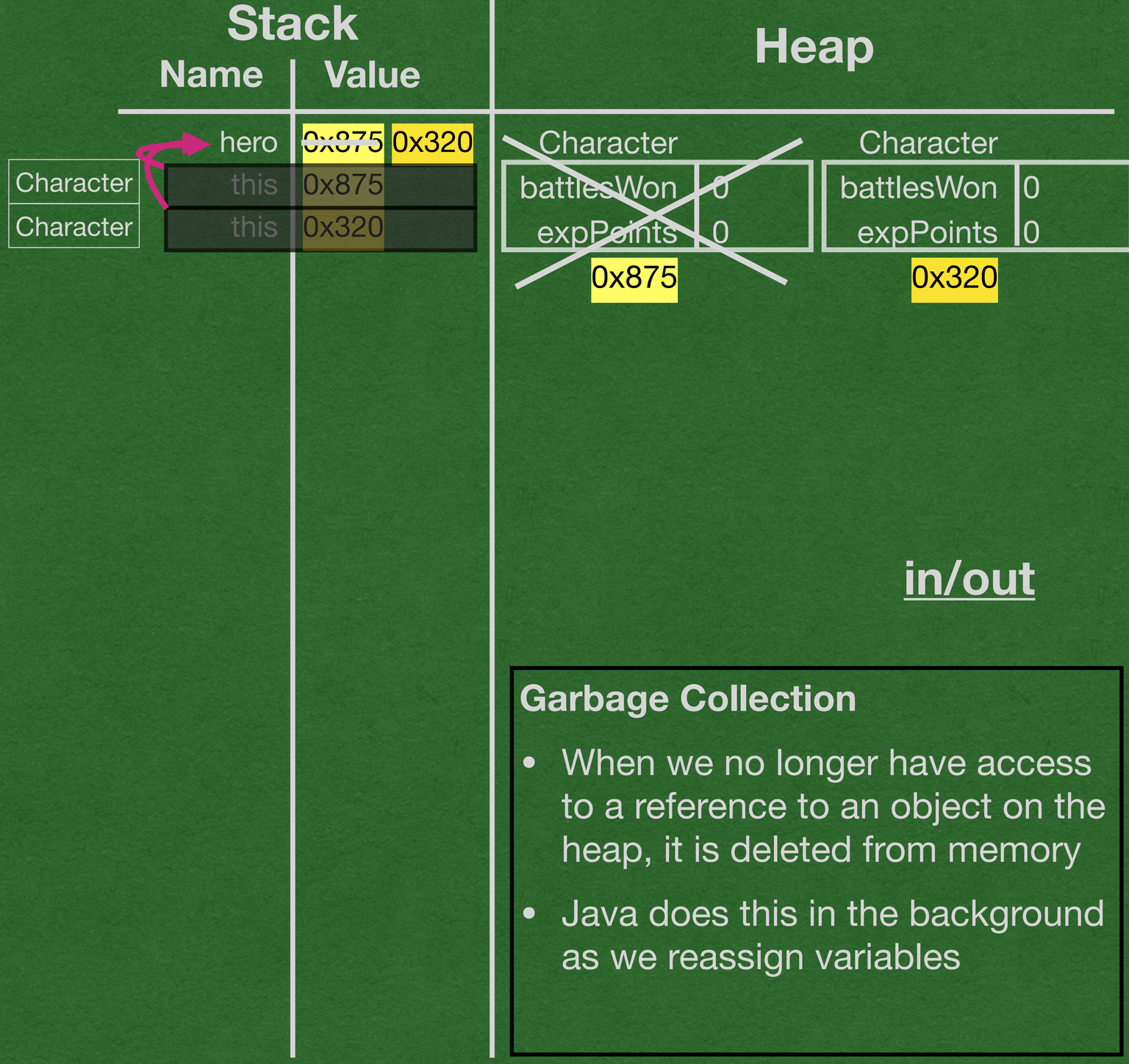
0x320

in/out

- 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

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

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

```java
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++) {
      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);
  }
}
```



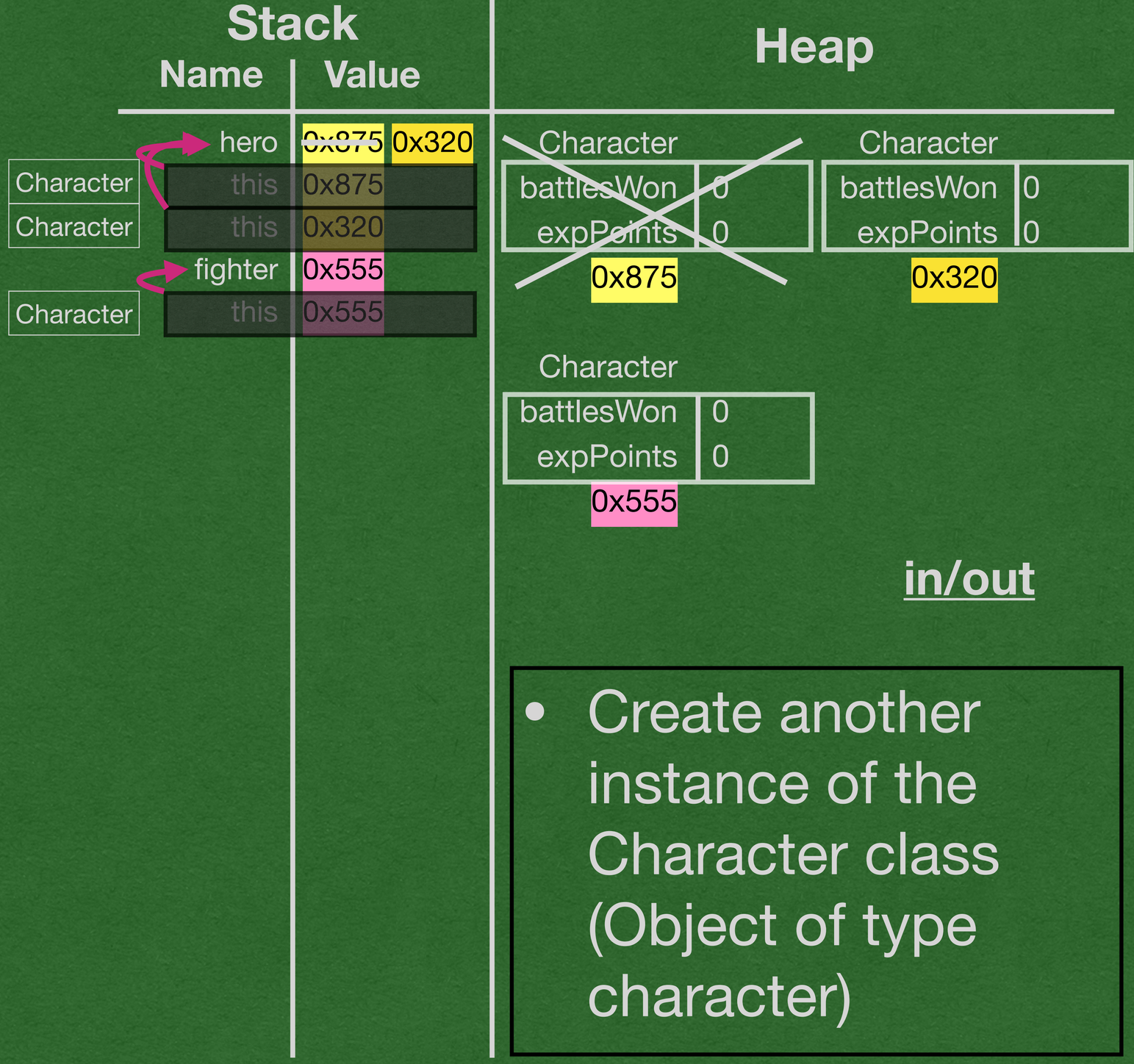
in/out

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

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

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

```java
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++) {
      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);
  }
}
```

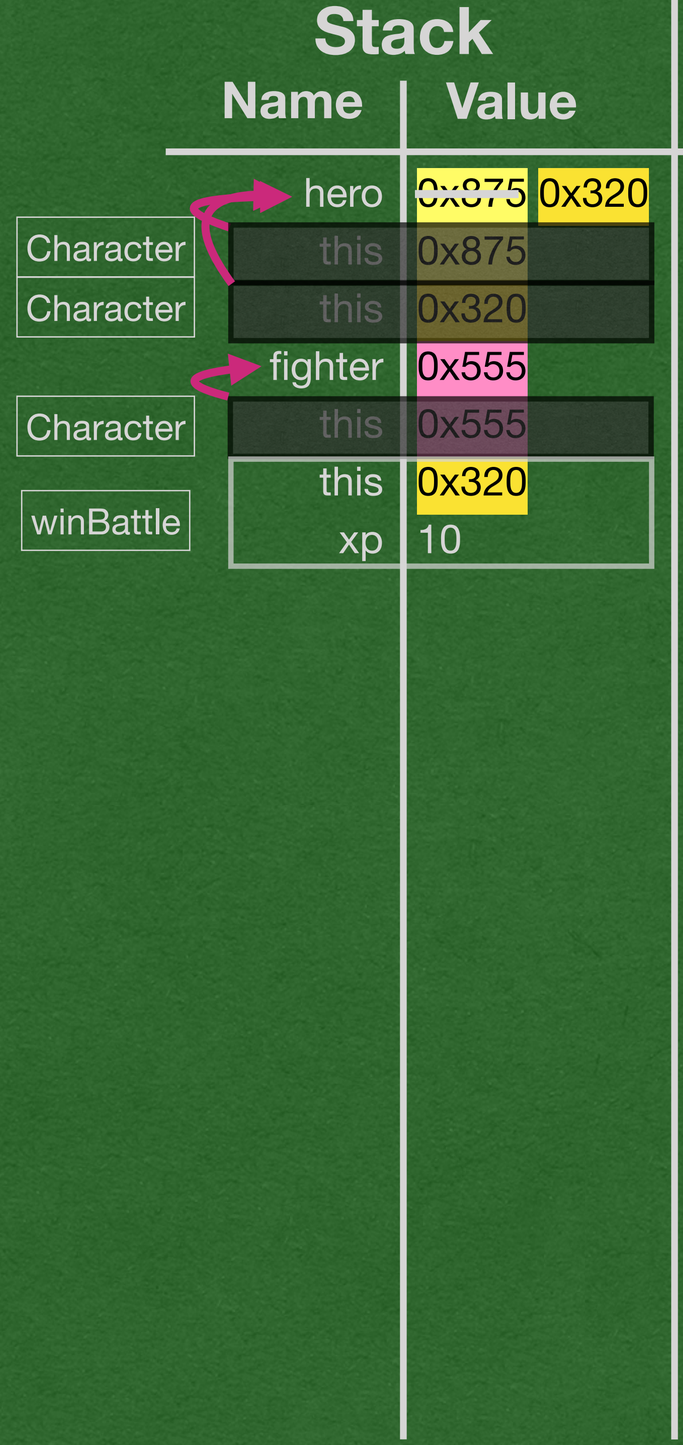


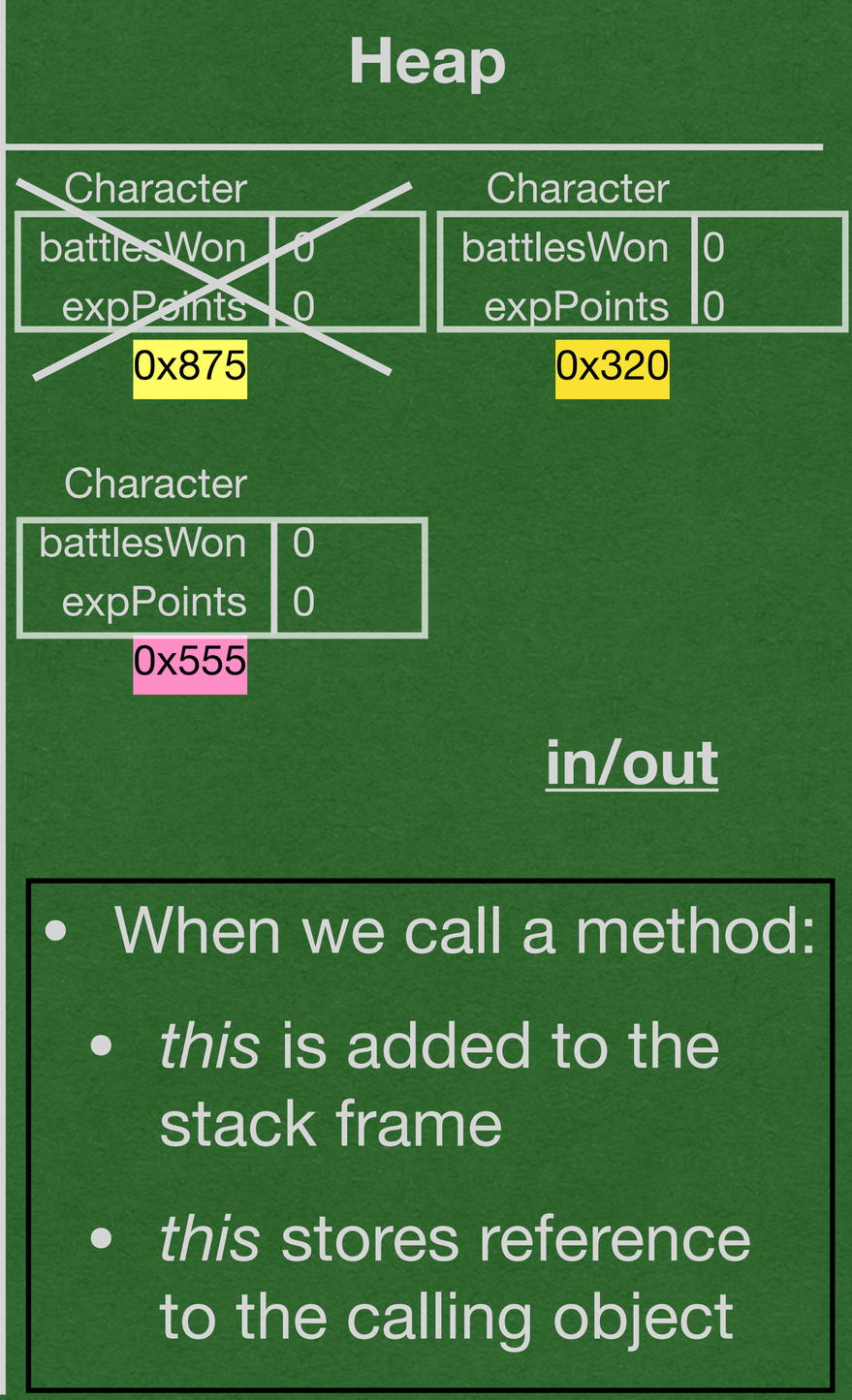
in/out

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

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

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

```java
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++) {
      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

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

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

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

```java
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++) {
      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);
  }
}
```

Stack

	Name	Value
	hero	~~0x875~~ 0x320
Character	this	0x875
Character	this	0x320
	fighter	0x555
Character	this	0x555
winBattle	this	0x320
	xp	10

Heap

Character
battlesWon 0
expPoints 0
0x875

Character
battlesWon 0
expPoints 0
0x320

Character
battlesWon 0
expPoints 0
0x555

in/out

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

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

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

```java
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++) {
      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);
  }
}
```

Stack

Name	Value
hero	~~0x875~~ 0x320
Character this	0x875
Character this	0x320
fighter	0x555
Character this	0x555
winBattle this	0x320
winBattle xp	10

Heap

Character — 0x875

battlesWon	0
expPoints	0

Character — 0x320

battlesWon	~~0~~ 1
expPoints	~~0~~ 10

Character — 0x555

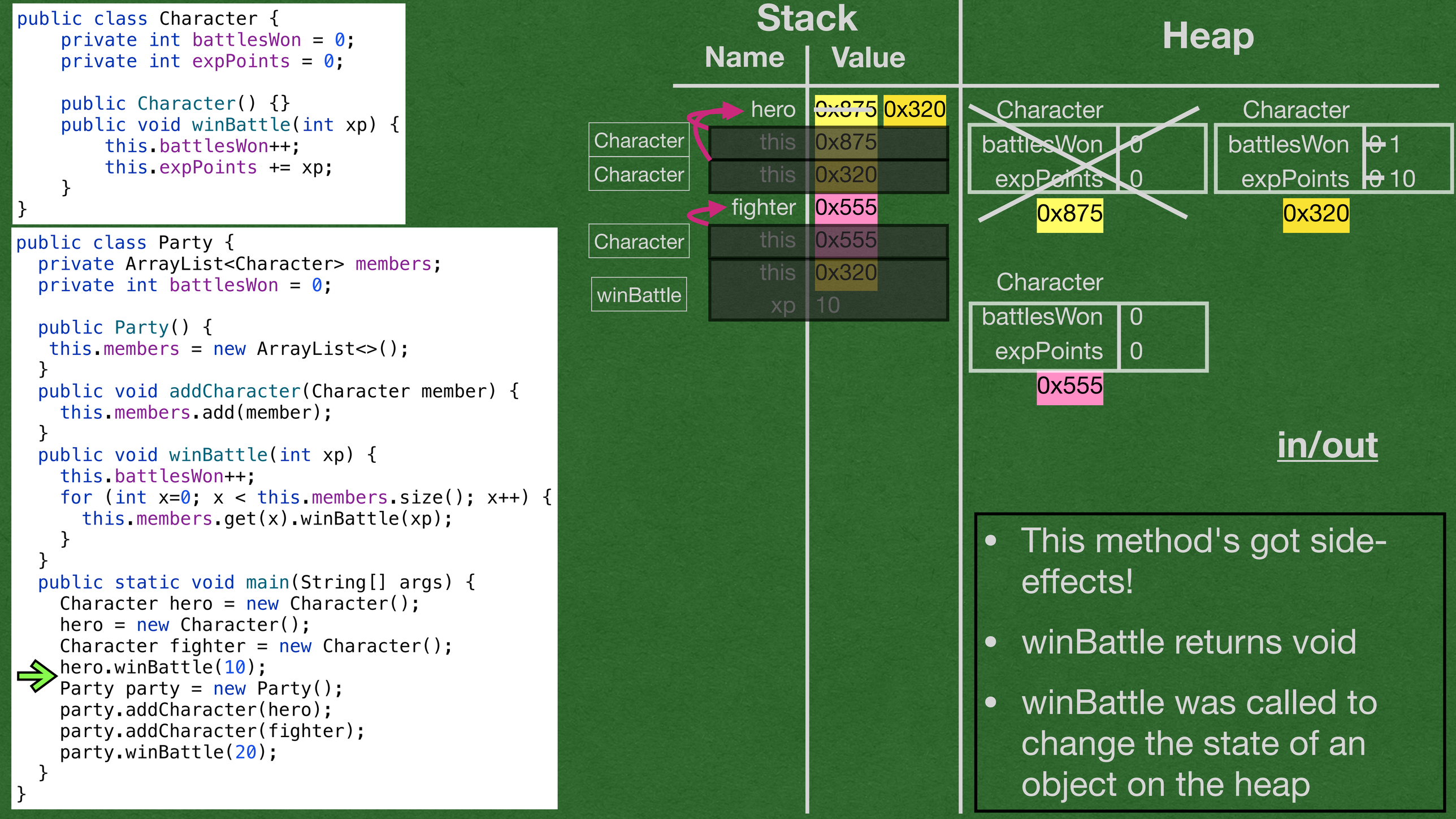
battlesWon	0
expPoints	0

in/out

- Follow the reference stored in *this* to update the instance variables

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

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

```java
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++) {
      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);
  }
}
```

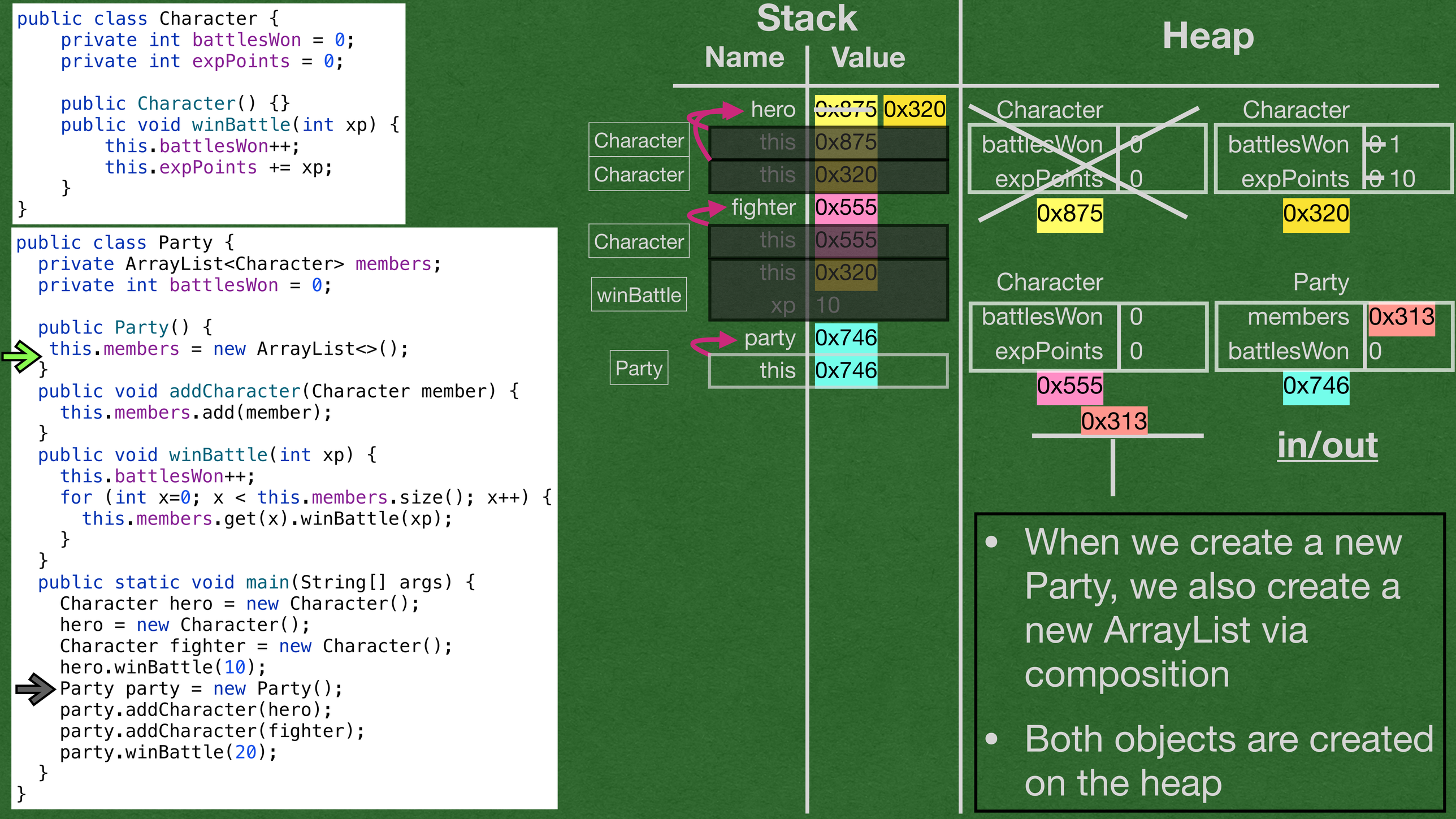


in/out

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

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

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

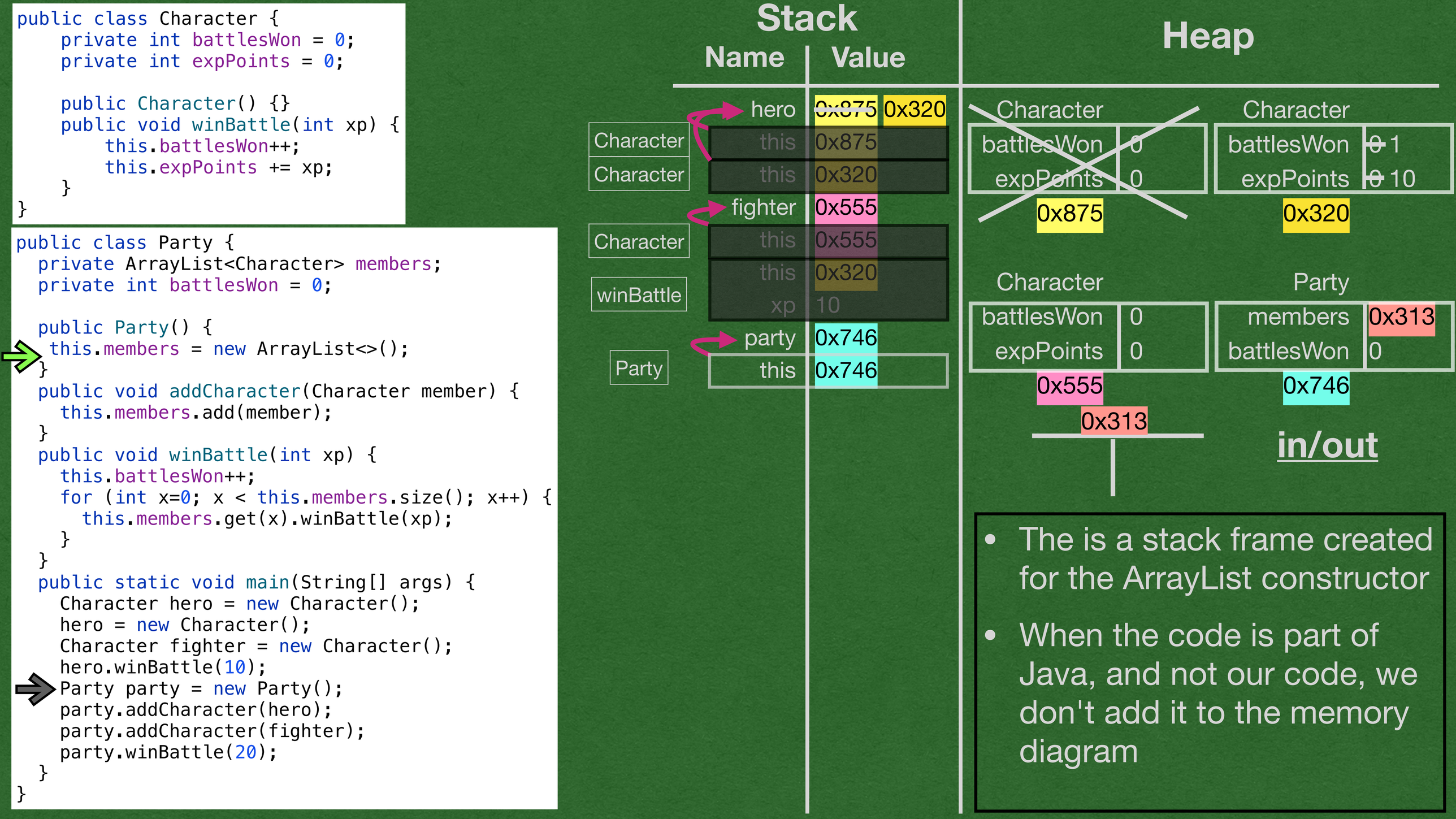
```java
public class Party {
  private ArrayList<Character> members;
  private int battlesWon = 0;

  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++) {
      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

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

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

```java
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++) {
      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

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

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

```java
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++) {
      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);
  }
}
```

Stack

	Name	Value
	hero	~~0x875~~ 0x320
Character	this	0x875
Character	this	0x320
	fighter	0x555
Character	this	0x555
	this	0x320
winBattle	xp	10
	party	0x746
Party	this	0x746
addCharacter	this	0x746
	member	0x320

Heap

Character (0x875) — ~~battlesWon 0~~, ~~expPoints 0~~

Character (0x320) — battlesWon ~~0~~ 1, expPoints ~~0~~ 10

Character (0x555) — battlesWon 0, expPoints 0

Party (0x746) — members 0x313, battlesWon 0

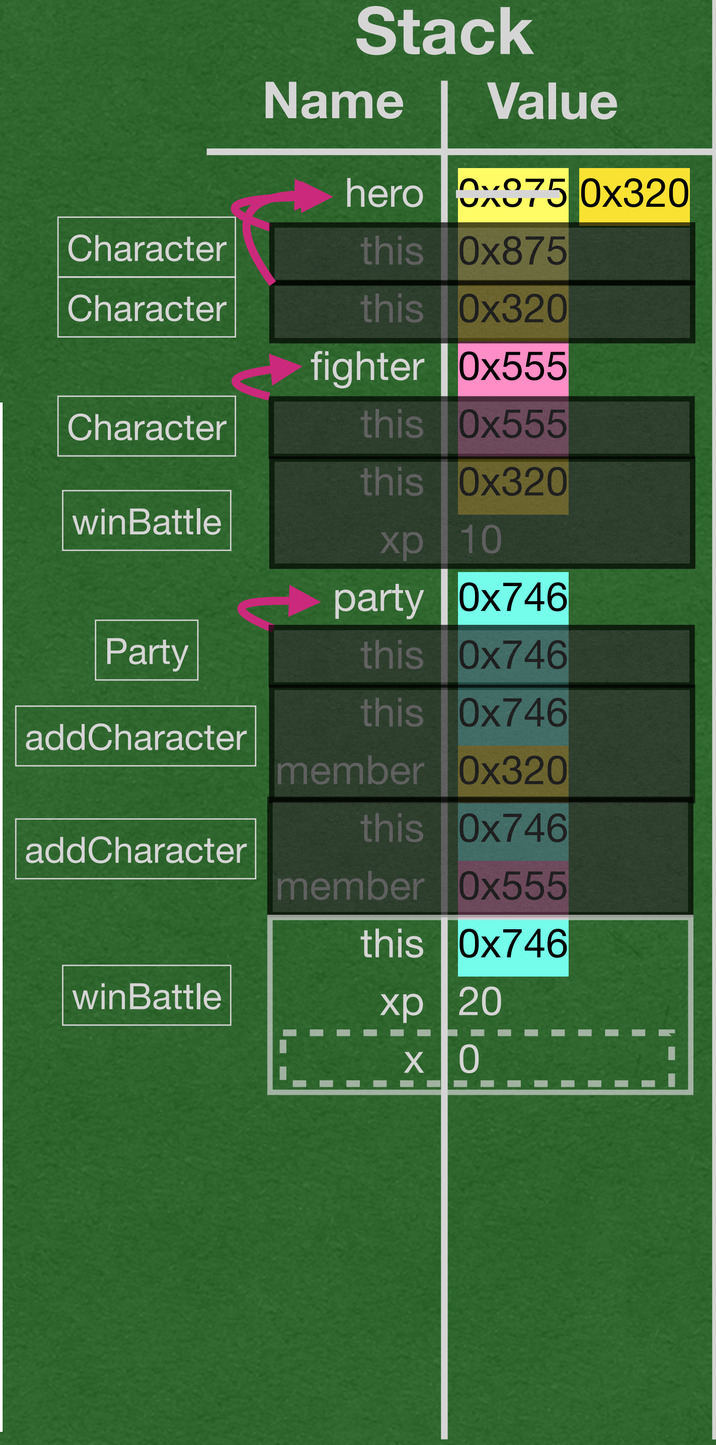
0x313 — 0 | 0x320

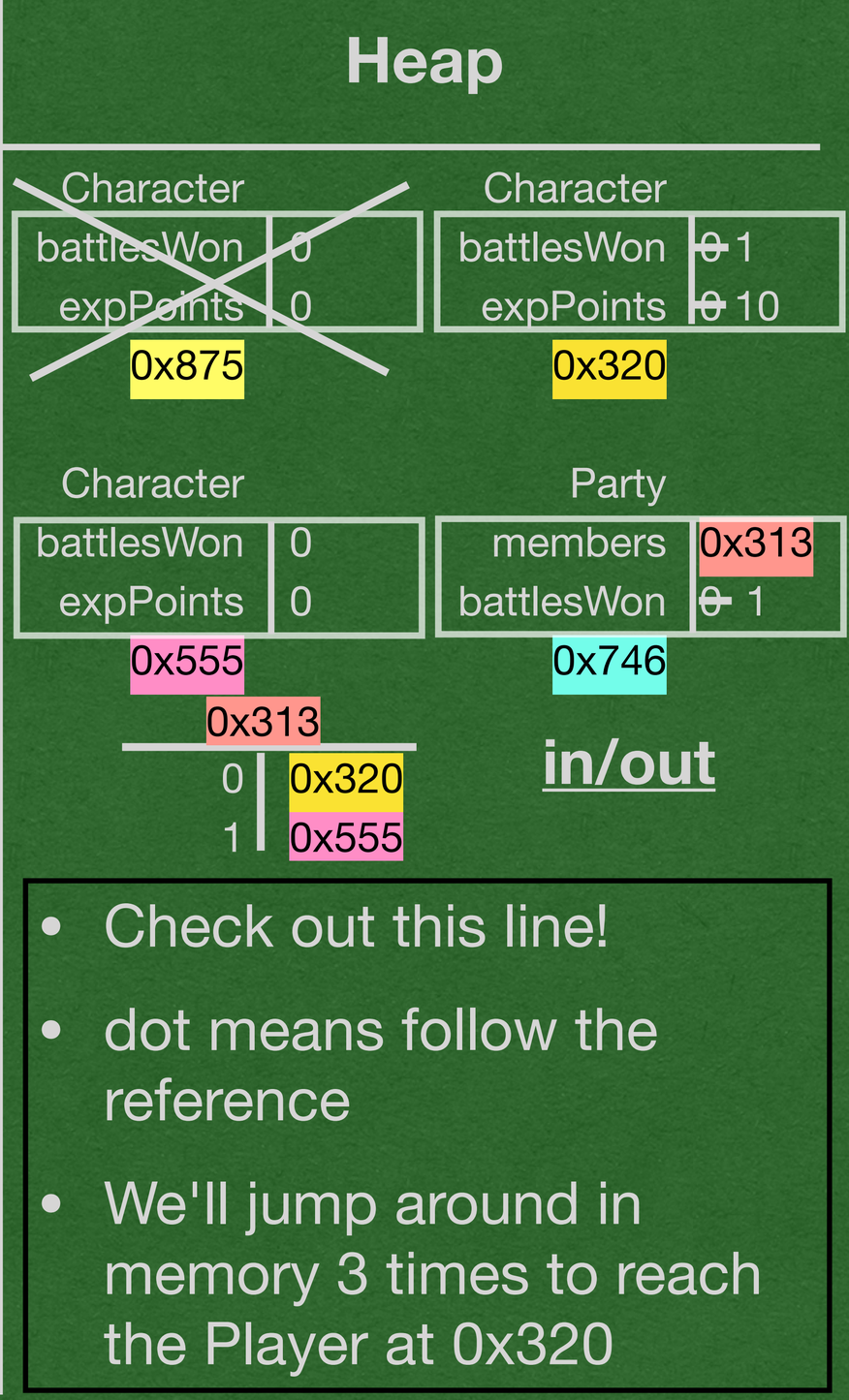
in/out

- addCharacter takes a reference to a Character as a parameter
- dot means "follow the reference"
- We follow 2 references in one line to get to the ArrayList in memory

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

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

```java
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++) {
      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);
  }
}
```





- Check out this line!
- dot means follow the reference
- We'll jump around in memory 3 times to reach the Player at 0x320

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

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

```java
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++) {
      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);
  }
}
```

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

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

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

```java
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++) {
      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);
  }
}
```

Stack

	Name	Value
	hero	~~0x875~~ 0x320
Character	this	0x875
Character	this	0x320
	fighter	0x555
Character	this	0x555
winBattle	this	0x320
	xp	10
	party	0x746
Party	this	0x746
addCharacter	this	0x746
	member	0x320
addCharacter	this	0x746
	member	0x555
winBattle	this	0x746
	xp	20
	x	~~0~~ 1
winBattle	this	0x320
	xp	20
winBattle	this	0x555
	xp	20

Heap

~~Character~~ (~~0x875~~)

battlesWon	0
expPoints	0

Character (0x320)

battlesWon	~~0~~ 1 ~~2~~
expPoints	~~0~~ ~~10~~ 30

Character (0x555)

battlesWon	~~0~~ 1
expPoints	~~0~~ 20

Party (0x746)

members	0x313
battlesWon	~~0~~ 1

0x313

0	0x320
1	0x555

in/out

- Repeat the process with 0x555

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

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

```java
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++) {
      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);
  }
}
```

Stack

Name	Value
hero	~~0x875~~ 0x320
Character: this	0x875
Character: this	0x320
fighter	0x555
Character: this	0x555
winBattle: this	0x320
winBattle: xp	10
party	0x746
Party: this	0x746
addCharacter: this	0x746
addCharacter: member	0x320
addCharacter: this	0x746
addCharacter: member	0x555
winBattle: this	0x746
winBattle: xp	20
winBattle: x	~~0 1~~ 2
winBattle: this	0x320
winBattle: xp	20
winBattle: this	0x555
winBattle: xp	20

Heap

~~Character~~ ~~battlesWon 0~~ ~~expPoints 0~~ ~~0x875~~

Character: battlesWon ~~0 1~~ 2; expPoints ~~0 10~~ 30; 0x320

Character: battlesWon ~~0~~ 1; expPoints ~~0~~ 20; 0x555

Party: members 0x313; battlesWon ~~0~~ 1; 0x746

0x313

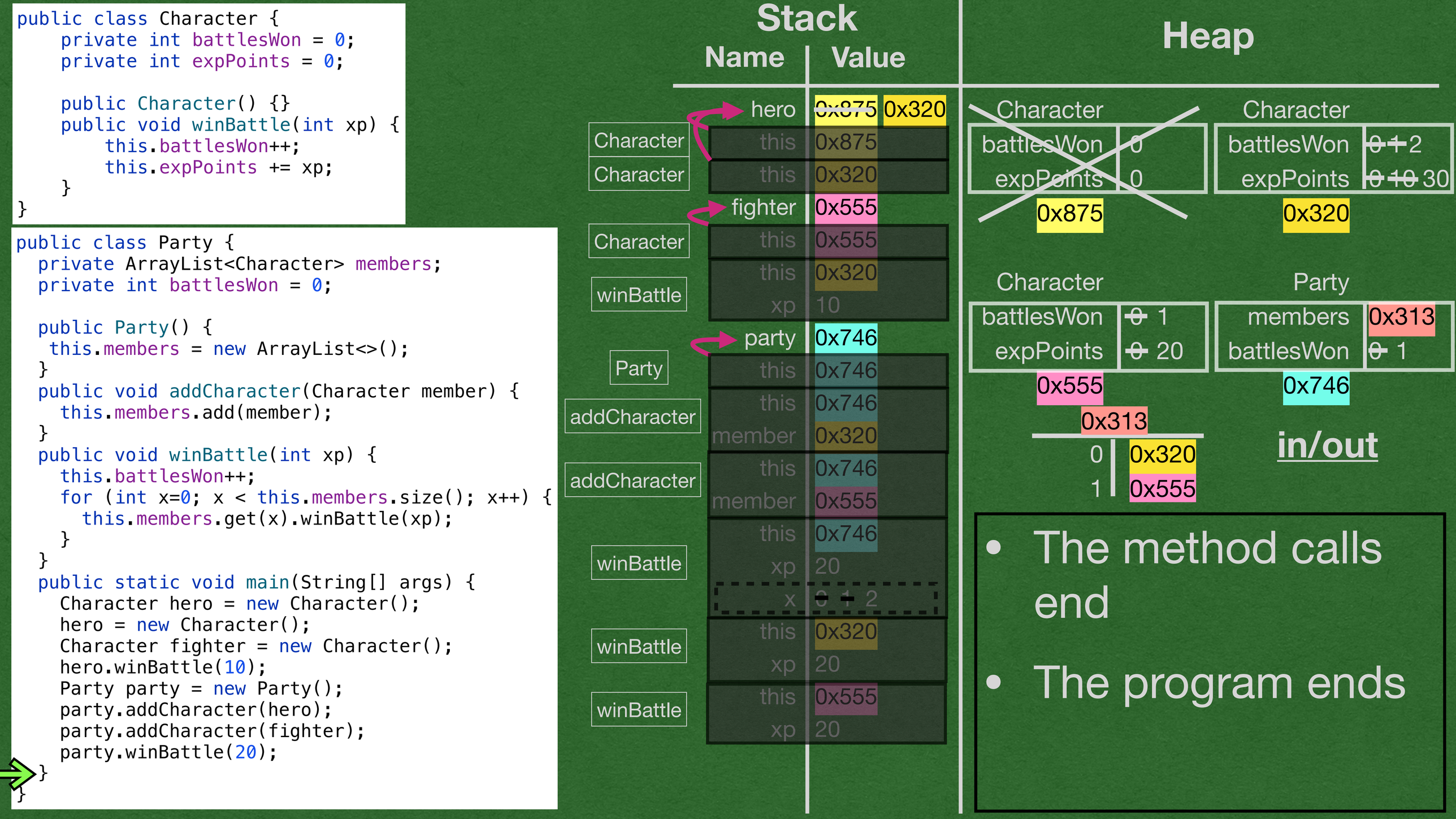
0	0x320
1	0x555

in/out

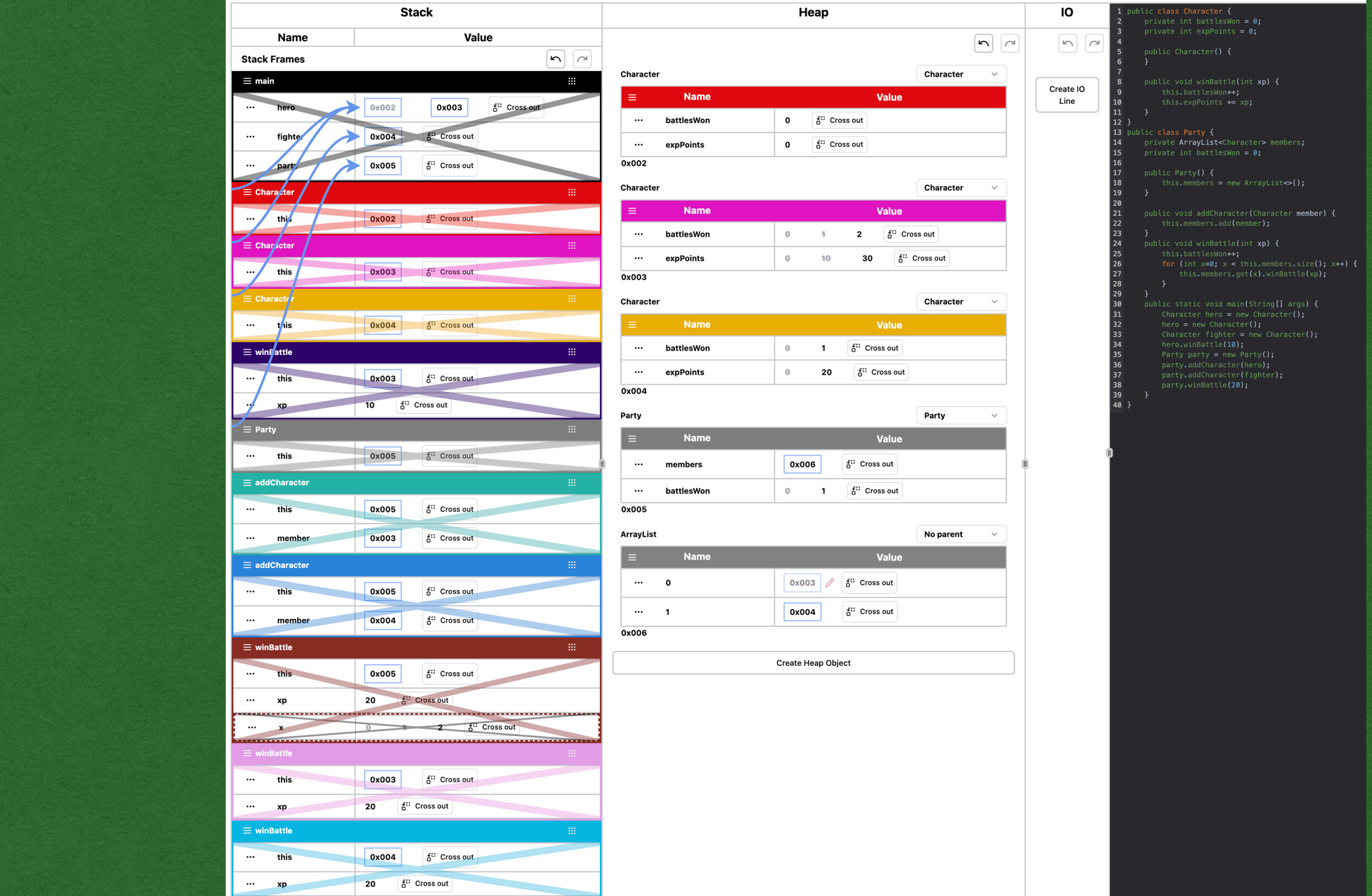
- The loop ends when x reaches 2
- x is removed from memory

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

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

```java
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++) {
      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 method calls end
- The program ends


Stack
Name
Value
Stack Frames
main
hero
0x002
0x003
Cross out
fighter
0x004
party
0x005
Character
this
0x002
Character
this
0x003
Character
this
0x004
winBattle
this
0x003
xp
10
Party
this
0x005
addCharacter
this
0x005
member
0x003
addCharacter
this
0x005
member
0x004
winBattle
this
0x005
xp
20
x
0
1
2
winBattle
this
0x003
xp
20
winBattle
this
0x004
xp
20
Heap
Character
Character
Name
Value
battlesWon
0
expPoints
0
0x002
Character
Character
battlesWon
0
1
2
expPoints
0
10
30
0x003
Character
Character
battlesWon
0
1
expPoints
0
20
0x004
Party
Party
members
0x006
battlesWon
0
1
0x005
ArrayList
No parent
0
0x003
1
0x004
0x006
Create Heap Object
IO
Create IO Line
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++) {
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);
}
}