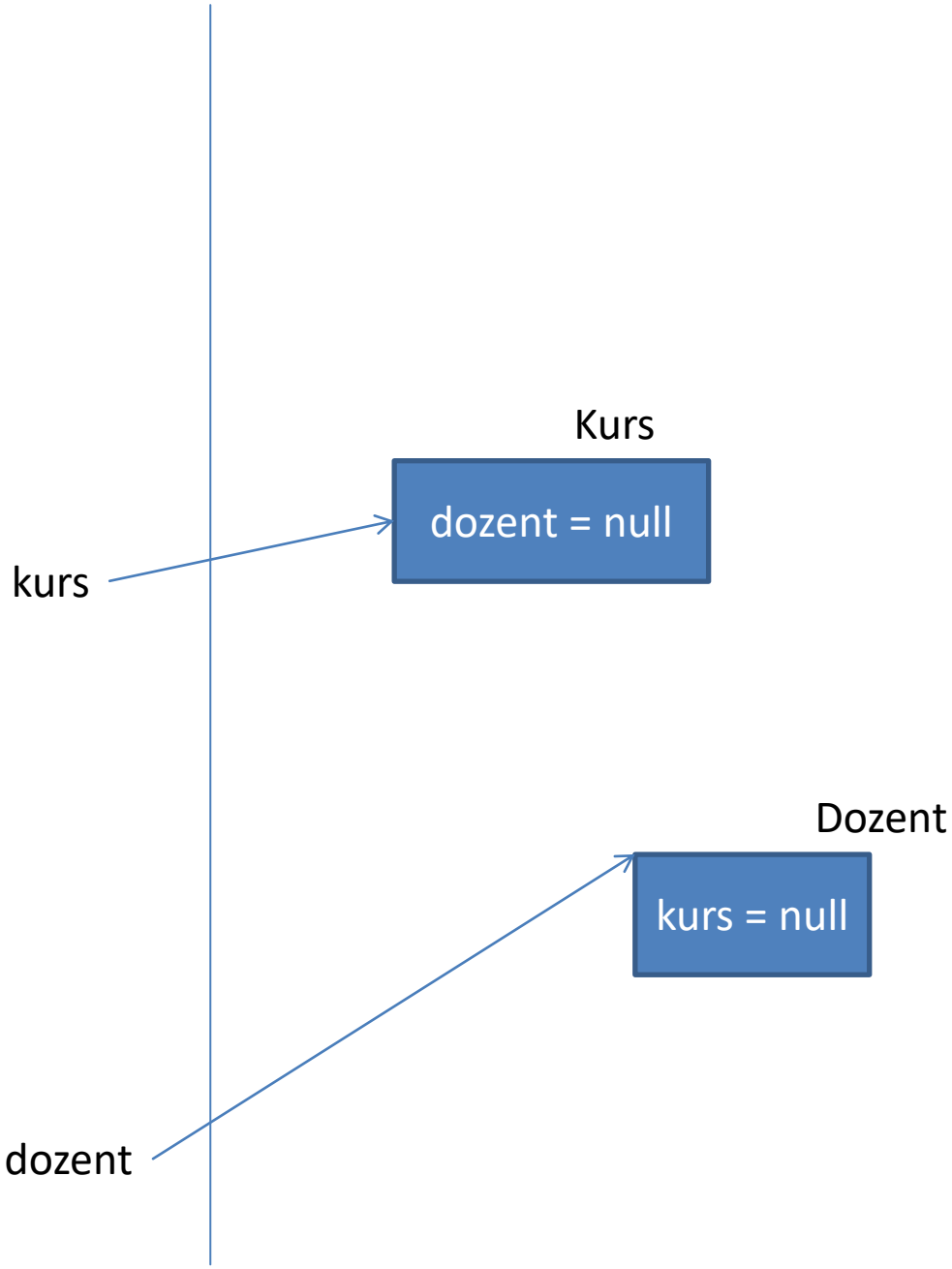
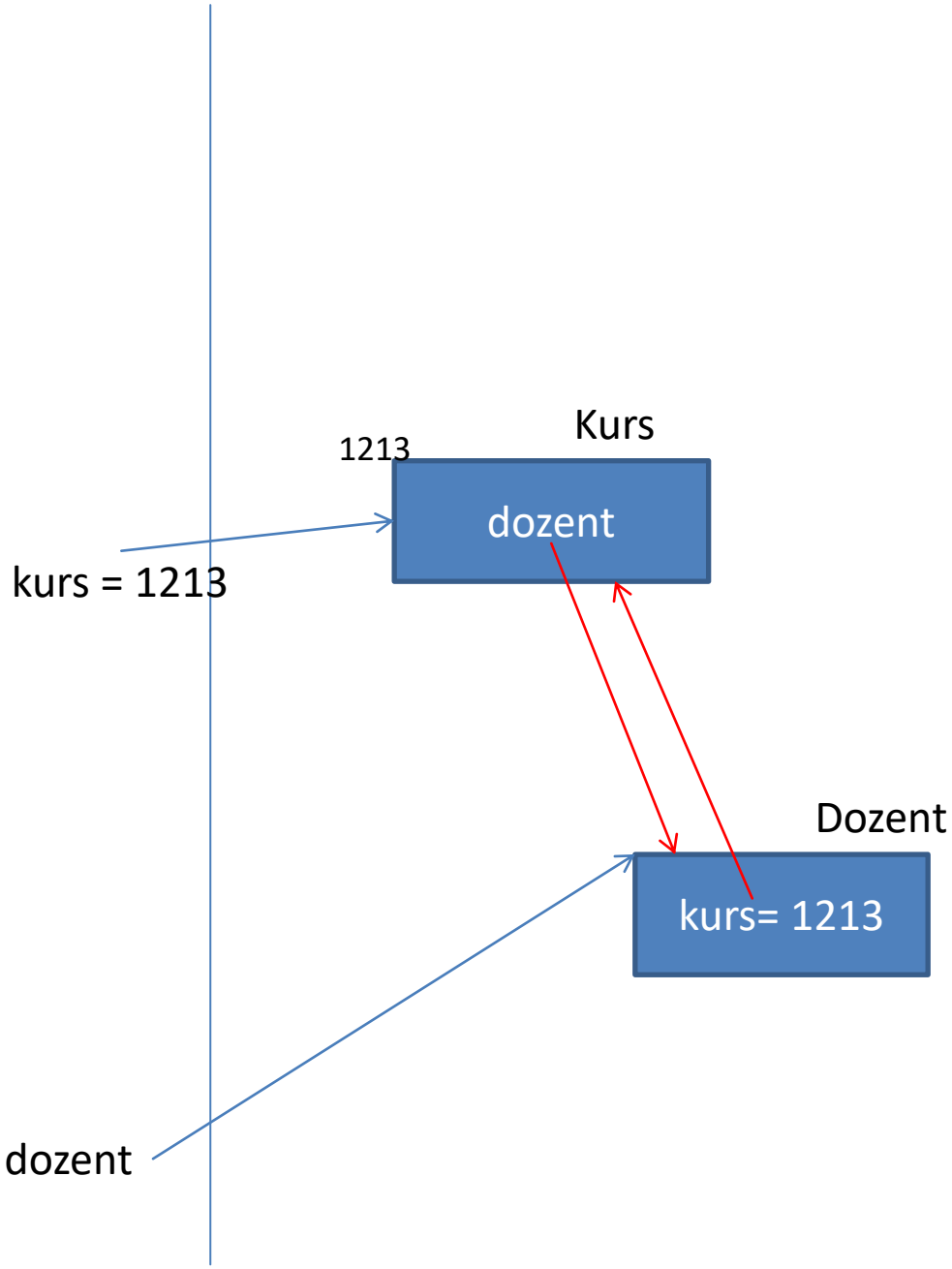
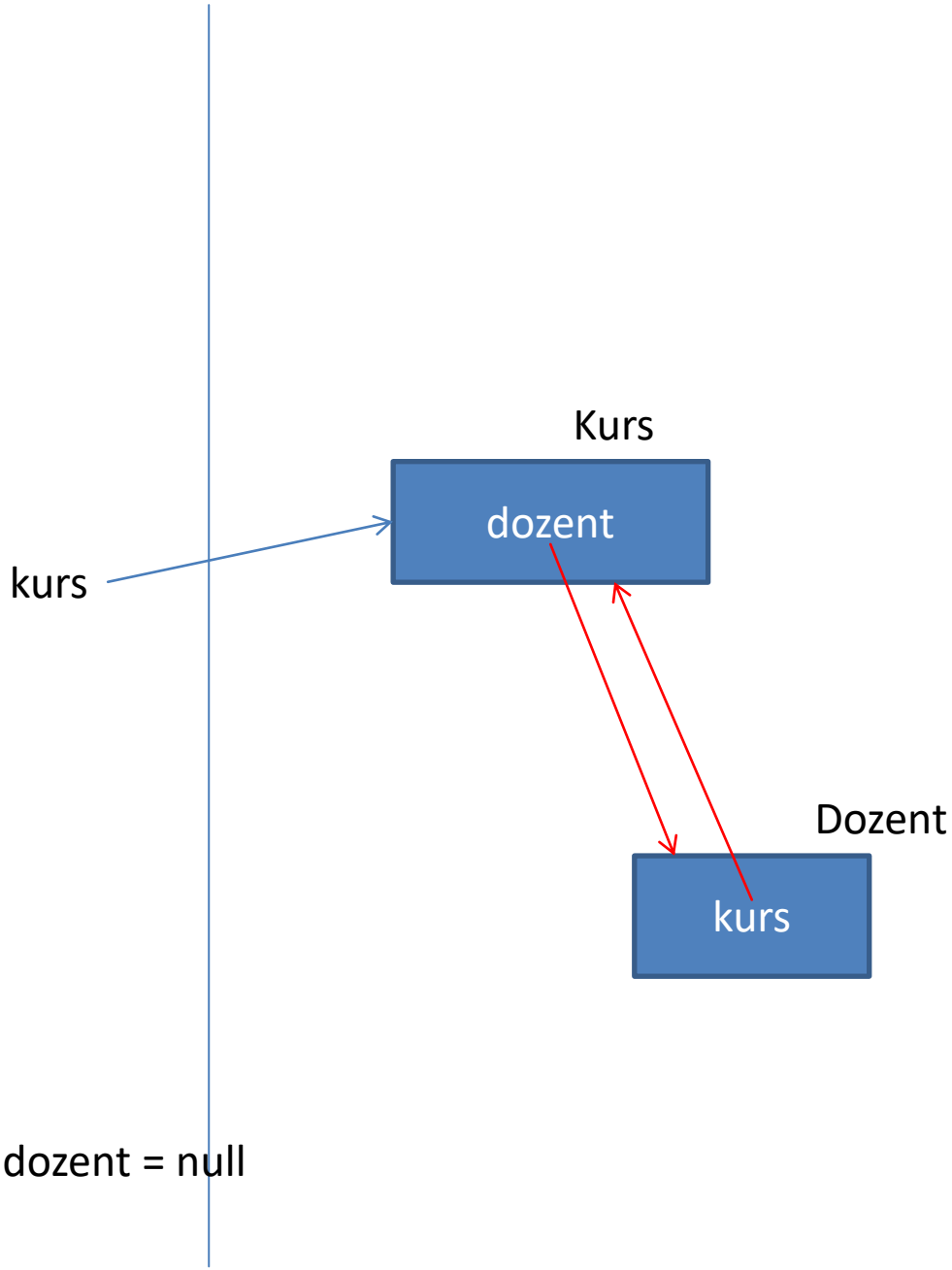
Inselgruppe




```
class Dozent {
  Kurs kurs;
}

class Kurs {
   Dozent dozent;
}

… main …

Dozent dozent = new Dozent();
Kurs kurs = new Kurs();
dozent.kurs = kurs;
kurs.dozent = dozent;
// Zeile A: 0 Objekte für GC
dozent = null;
// Zeile B: 0 Objekte für GC
kurs = null;
// Zeile C: 2 Objekte für GC
```

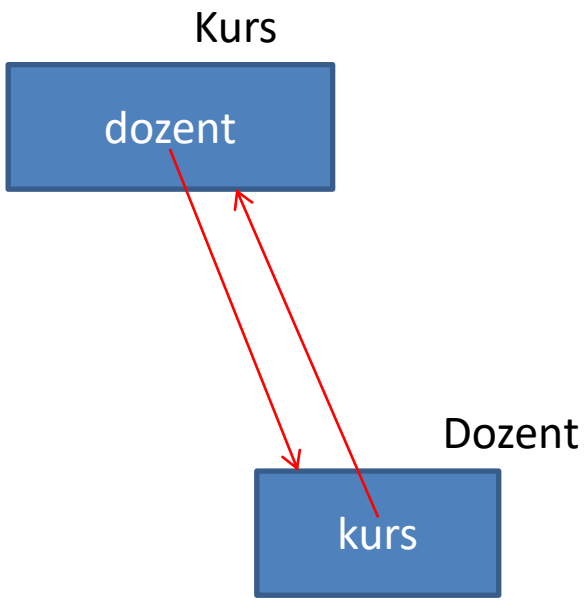



```
class Dozent {
   Kurs kurs;
}

class Kurs {
    Dozent dozent;
}

… main …

Dozent dozent = new Dozent();
Kurs kurs = new Kurs();
dozent.kurs = kurs;
kurs.dozent = dozent;
// Zeile A: 0 Objekte für GC
dozent = null;
// Zeile B: 0 Objekte für GC
kurs = null;
// Zeile C: 2 Objekte für GC
```




```
class Dozent {
   Kurs kurs;
}

class Kurs {
    Dozent dozent;
}

… main …

Dozent dozent = new Dozent();
Kurs kurs = new Kurs();
dozent.kurs = kurs;
kurs.dozent = dozent;
// Zeile A: 0 Objekte für GC
dozent = null;
// Zeile B: 0 Objekte für GC
kurs = null;
// Zeile C: 2 Objekte für GC
```

kurs = null



dozent = null

```
class Dozent {
   Kurs kurs;
}

class Kurs {
    Dozent dozent;
}

… main …

Dozent dozent = new Dozent();
Kurs kurs = new Kurs();
dozent.kurs = kurs;
kurs.dozent = dozent;
// Zeile A: 0 Objekte für GC
dozent = null;
// Zeile B: 0 Objekte für GC
kurs = null;
// Zeile C: 2 Objekte für GC
```

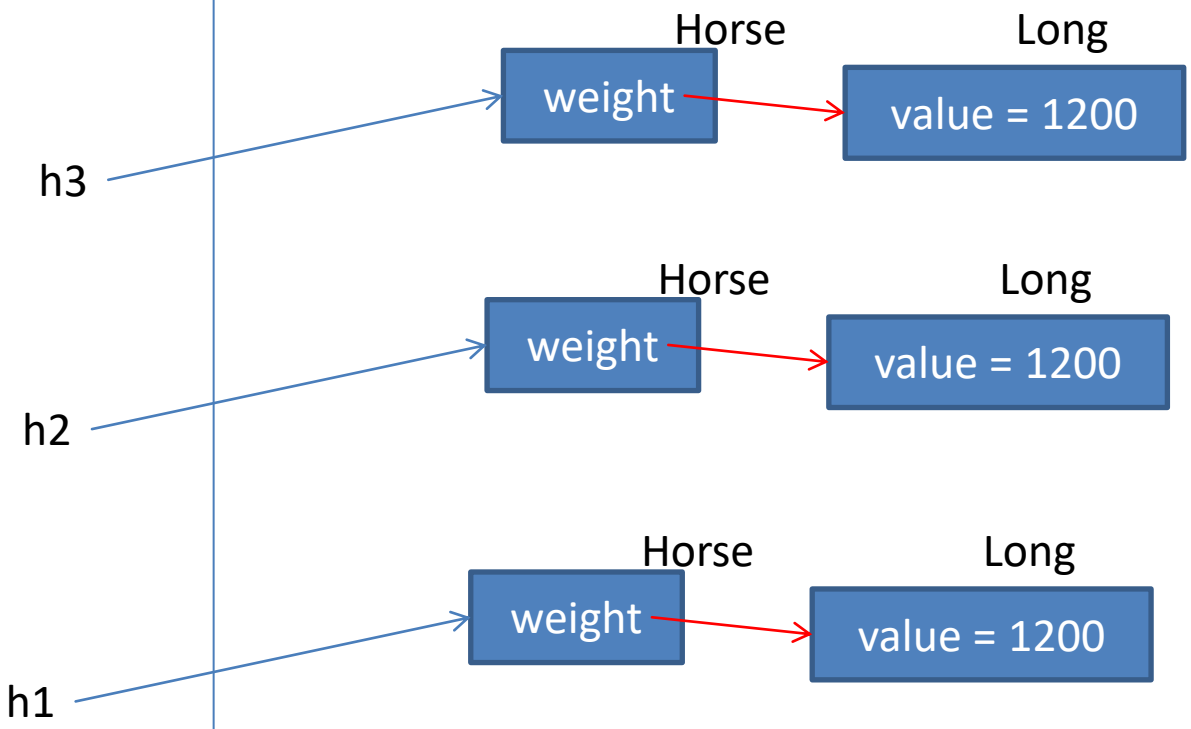
STACK

Autoboxing:
Long weight = Long.valueOf(1200L);

```
class Horse {
  Long weight = 1200L;
}
```

…main…

```
Horse h1 = new Horse();
Horse h2 = new Horse();
Horse h3 = new Horse();

h3 = h1;
h1 = h2;
h2 = null;
h3 = h1;

// Zeile A. ? Objekte für GC
```

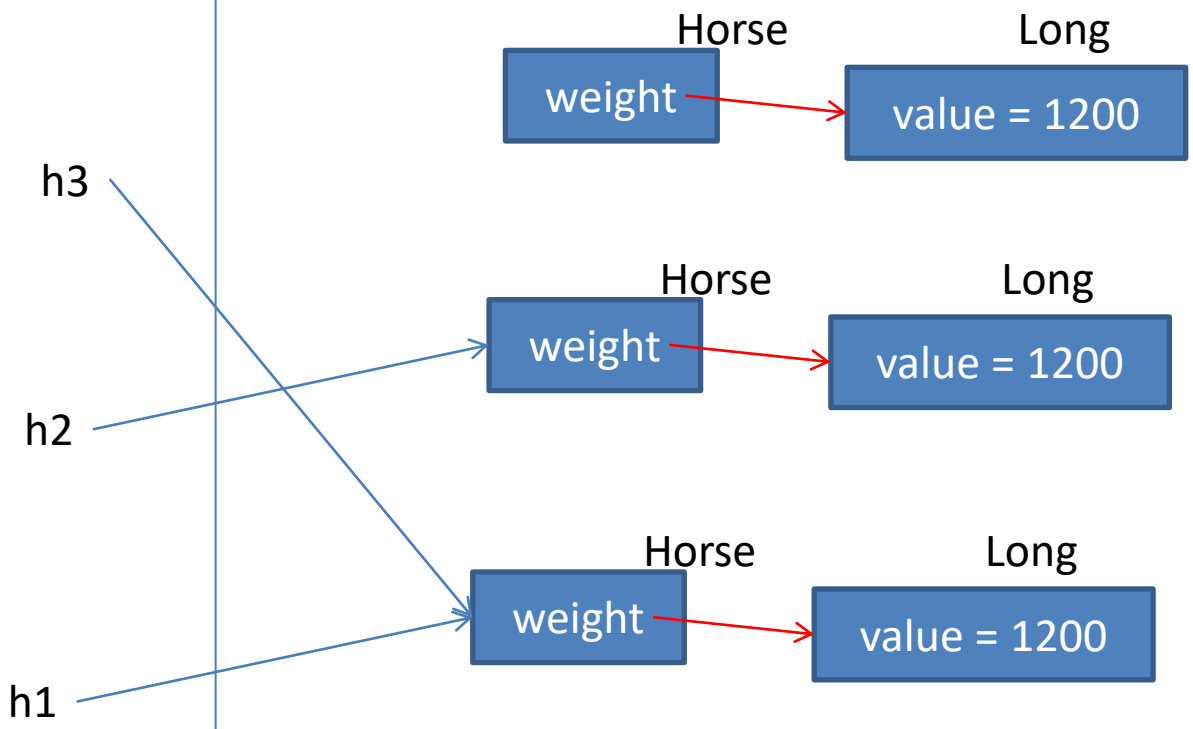


STACK

Autoboxing:
Long weight = Long.valueOf(1200L);

```
class Horse {
  Long weight = 1200L;
}
```

```
…main…

Horse h1 = new Horse();
Horse h2 = new Horse();
Horse h3 = new Horse();

h3 = h1;
h1 = h2;
h2 = null;
h3 = h1;

// Zeile A. ? Objekte für GC
```

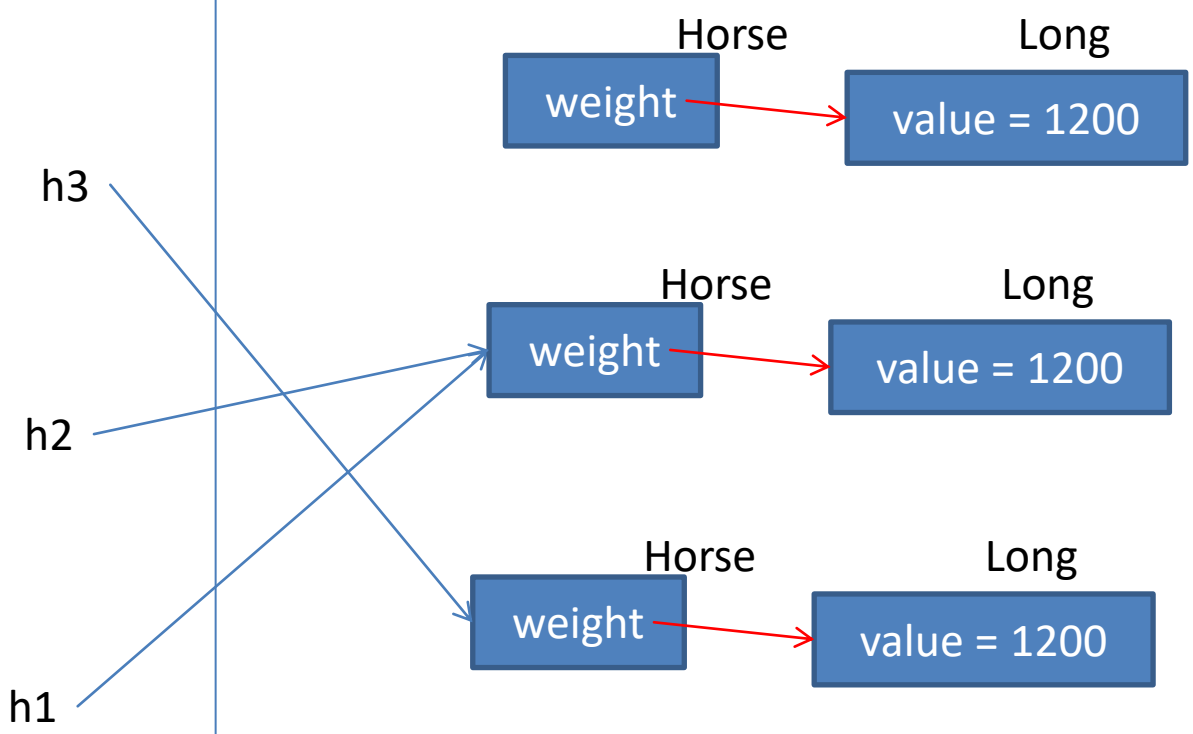


STACK

Autoboxing:
Long weight = Long.valueOf(1200L);

```
class Horse {
  Long weight = 1200L;
}
```

```
…main…

Horse h1 = new Horse();
Horse h2 = new Horse();
Horse h3 = new Horse();

h3 = h1;
h1 = h2;
h2 = null;
h3 = h1;

// Zeile A. ? Objekte für GC
```

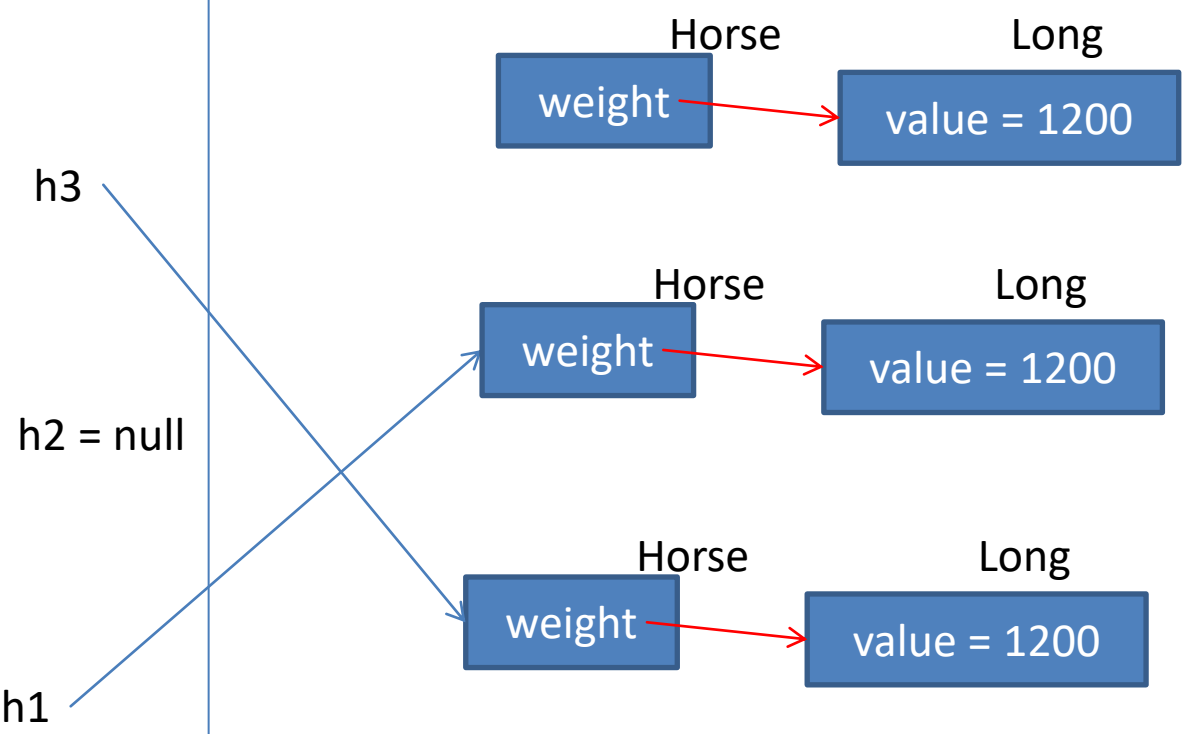


STACK

Autoboxing:
Long weight = Long.valueOf(1200L);

```
class Horse {
  Long weight = 1200L;
}
```

```
…main…

Horse h1 = new Horse();
Horse h2 = new Horse();
Horse h3 = new Horse();

h3 = h1;
h1 = h2;
h2 = null;
h3 = h1;

// Zeile A. ? Objekte für GC
```

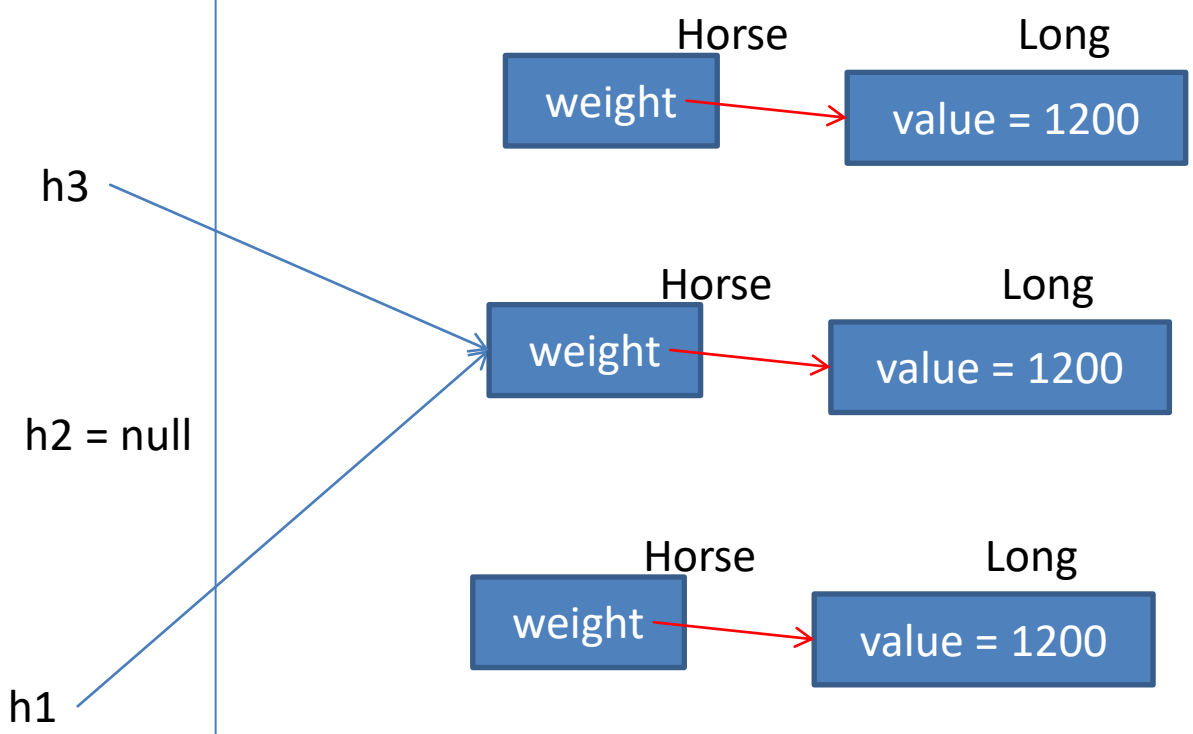


STACK

Autoboxing:
Long weight = Long.valueOf(1200L);

```
class Horse {
  Long weight = 1200L;
}
```

```
…main…

Horse h1 = new Horse();
Horse h2 = new Horse();
Horse h3 = new Horse();

h3 = h1;
h1 = h2;
h2 = null;
h3 = h1;

// Zeile A. ? Objekte für GC
```



Fragen:

1. Wie viele String-Objekte werden beim Aufruf der Methode 'getString' erzeugt ?

2. Wie viele String-Objekte stehen in der Zeile A dem Garbage Collector zur Verfügung?

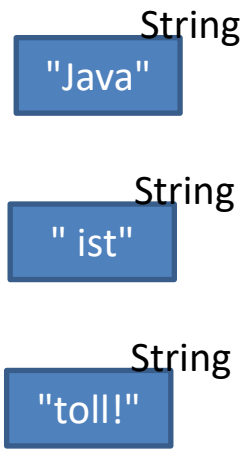
```
static String getString() {
   String s = "Java";

    s += " ist";
    s.concat("toll!");

    return s.toString();
}

public static void main(String[] args) {
     String s = getString();
    // Zeile A
}
```

Im weiteren werden String-Objekte vereinfacht dargestellt (ohne das dazugehörige char[] zu zeichnen)

Bevor die main startet, wird ihre Klasse geladen und String-Pool gefüllt.

```
static String getString() {
   String s = "Java";

    s += " ist";
    s.concat("toll!");

    return s.toString();
}

public static void main(String[] args) {
     String s = getString();
    // Zeile A
}
```

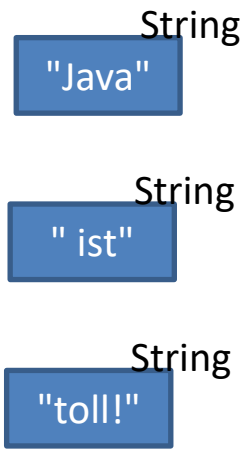
String-Pool:



Die main startet

```
static String getString() {
   String s = "Java";

   s += " ist";
   s.concat("toll!");

   return s.toString();
}

public static void main(String[] args) {
    String s = getString();
   // Zeile A
}
```

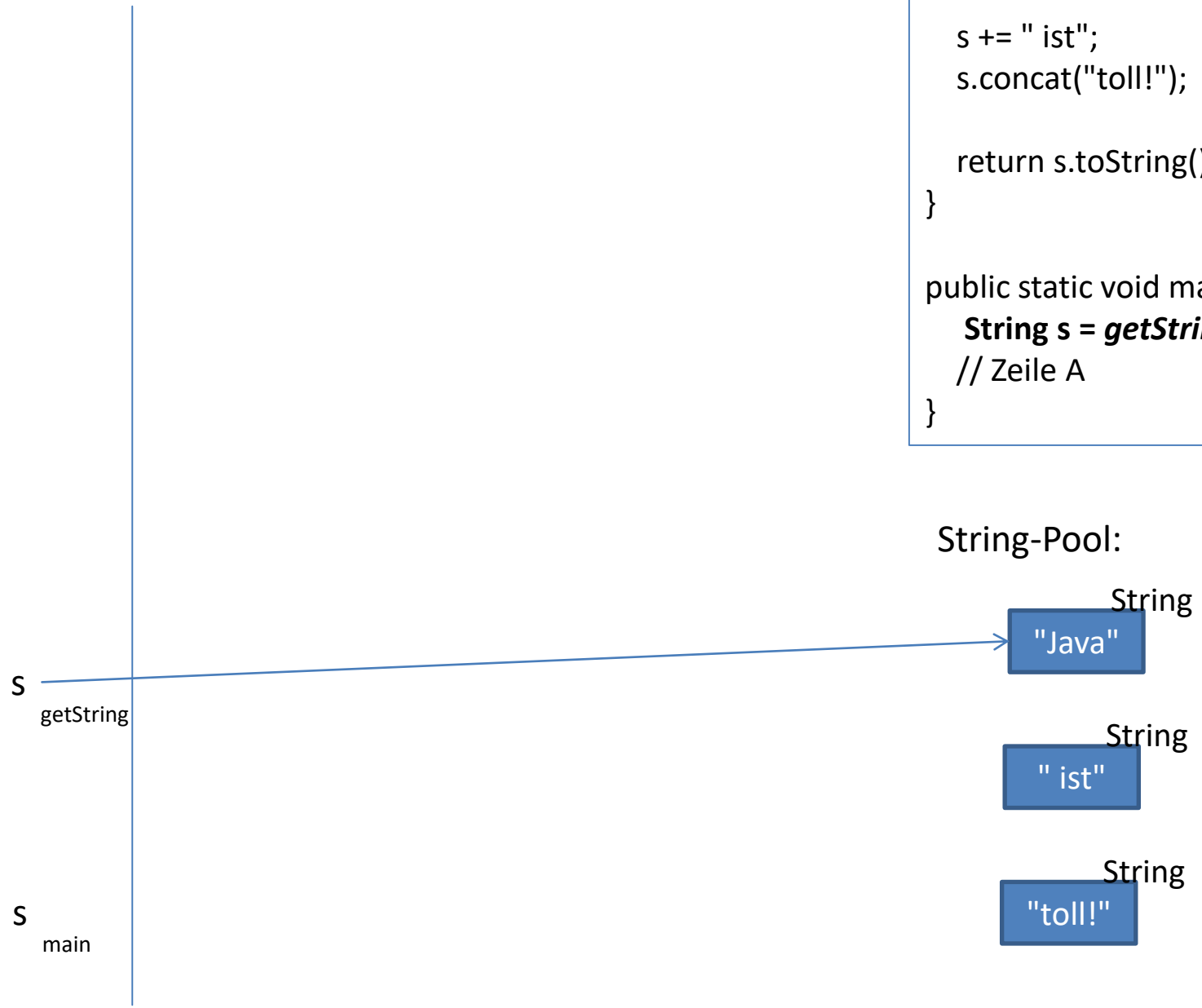
String-Pool:



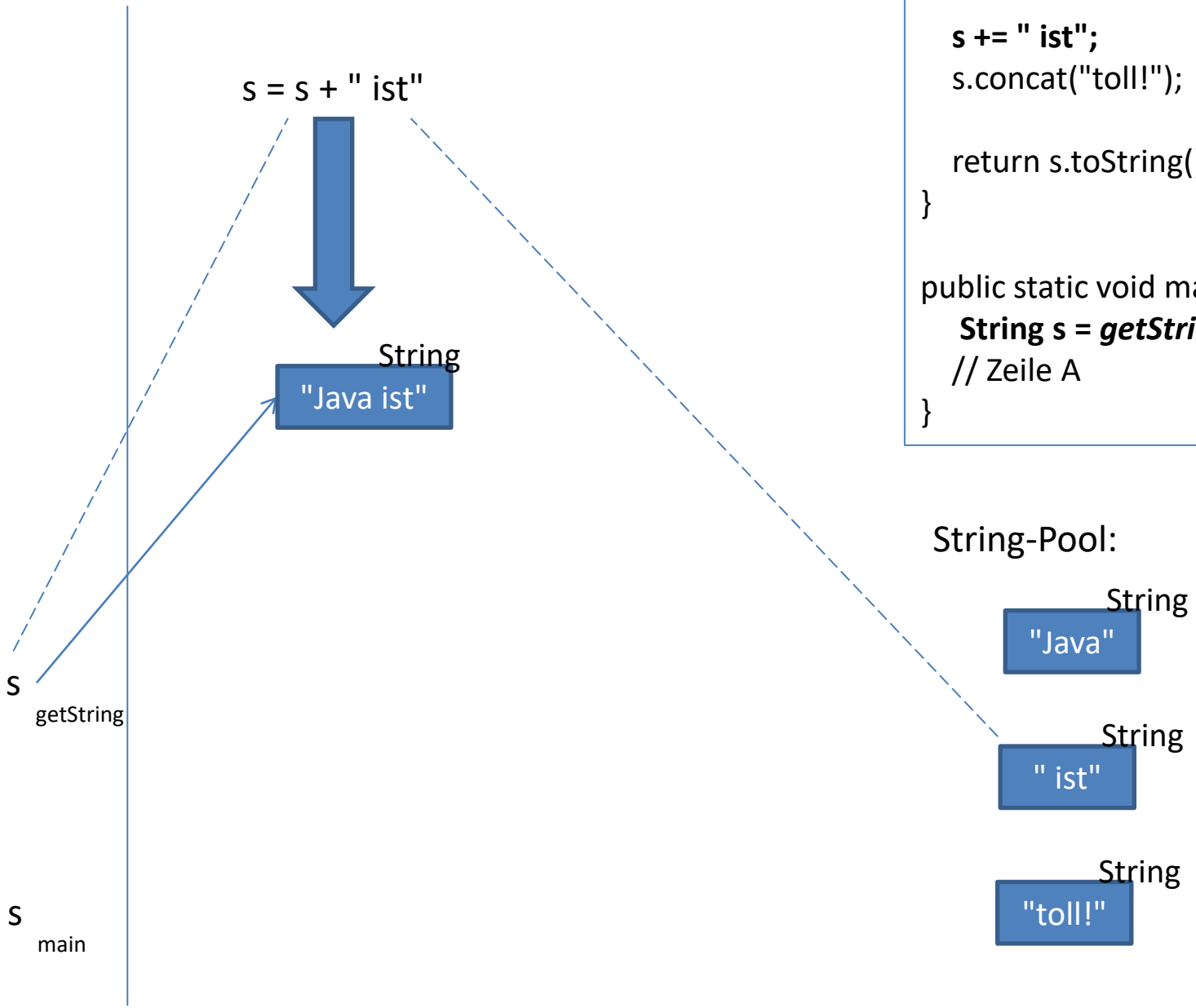
s

Die main startet

```
static String getString() {
  String s = "Java";

   s += " ist";
   s.concat("toll!");

   return s.toString();
}

public static void main(String[] args) {
    String s = getString();
   // Zeile A
}
```



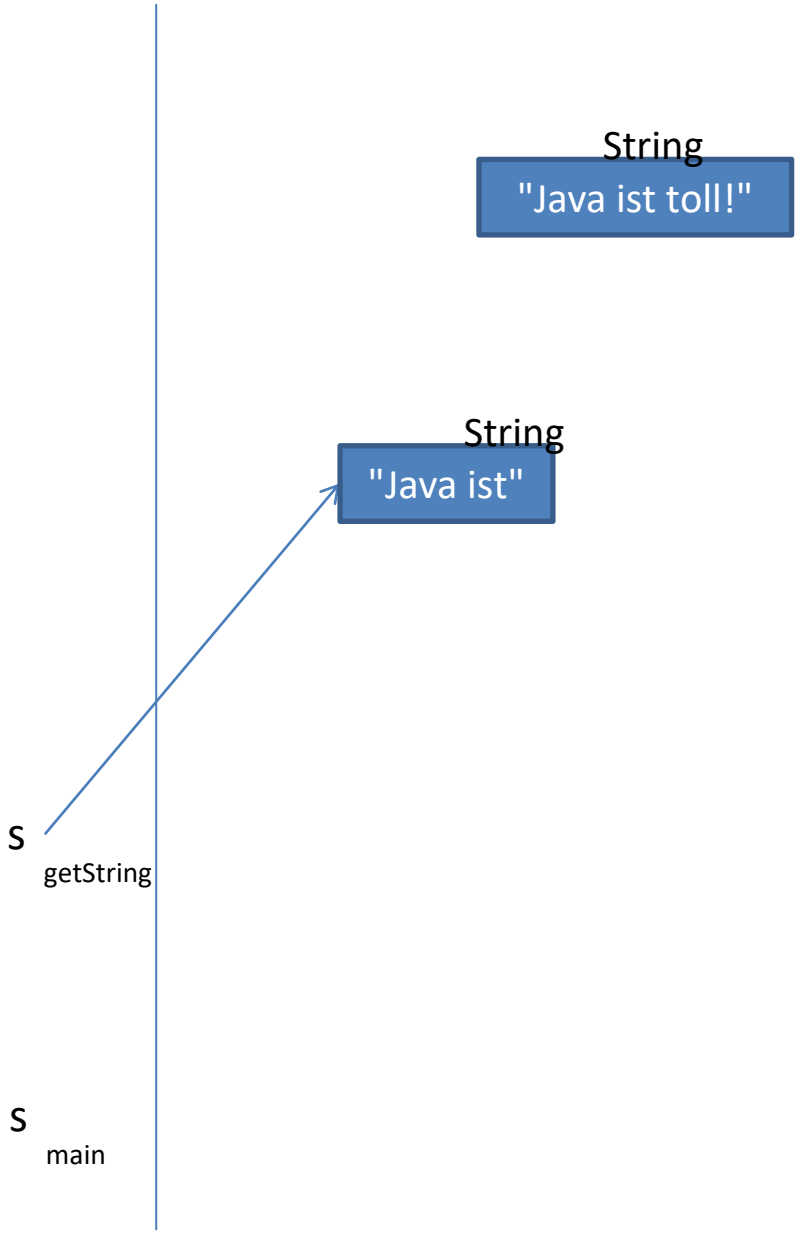
Die main startet




```
static String getString() {
   String s = "Java";

   s += " ist";
   s.concat("toll!");

   return s.toString();
}

public static void main(String[] args) {
    String s = getString();
   // Zeile A
}
```

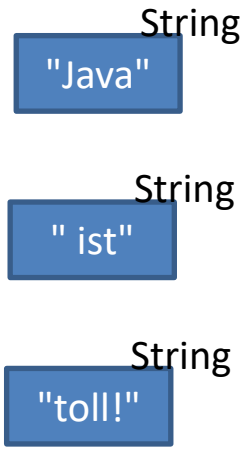
String-Pool:

Die main startet




```
static String getString() {
   String s = "Java";

   s += " ist";
   s.concat("toll!");

   return s.toString();
}

public static void main(String[] args) {
    String s = getString();
   // Zeile A
}
```

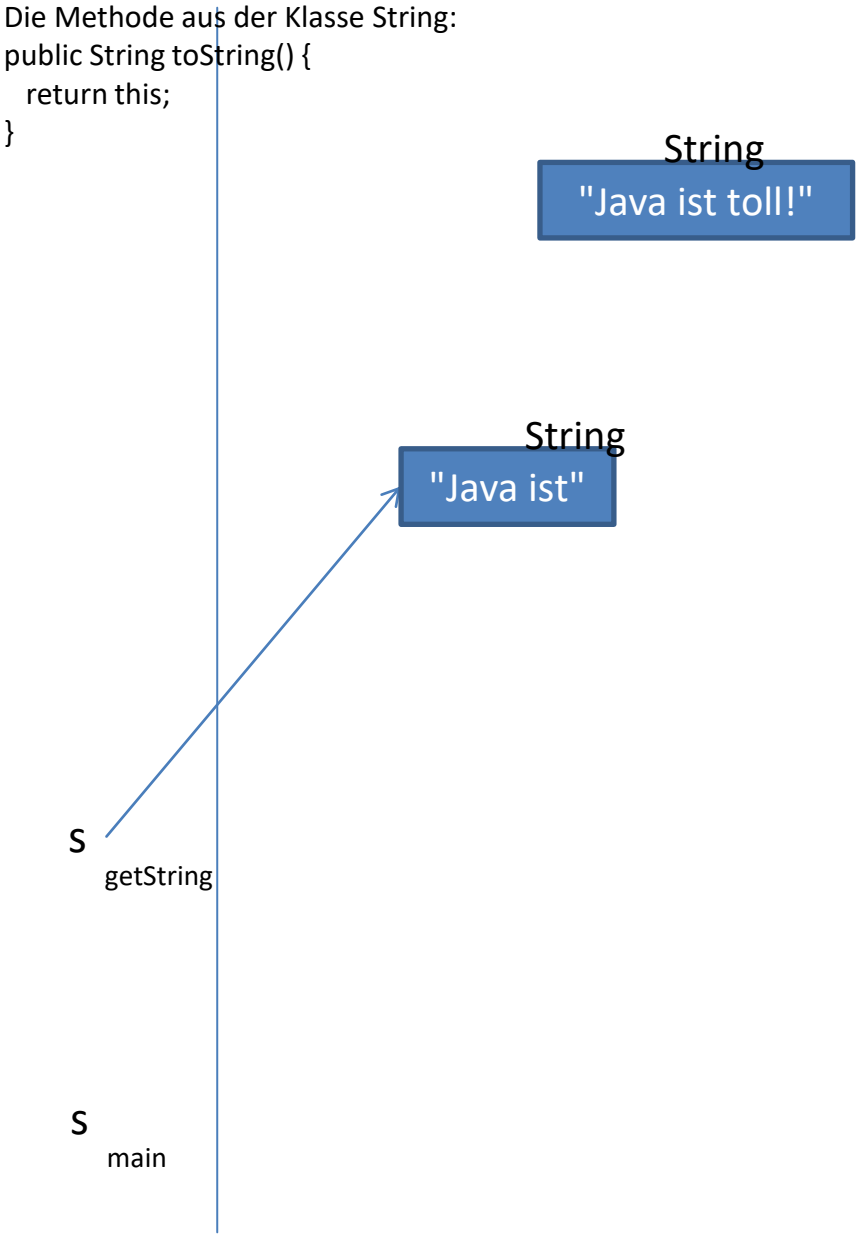
String-Pool:



Die main startet

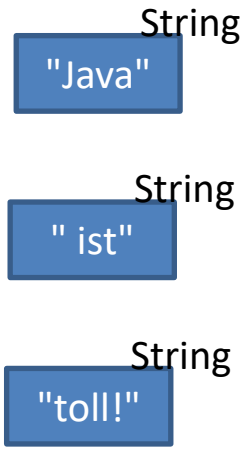
Die Methode aus der Klasse String:

```
public String toString() {
   return this;
}
```

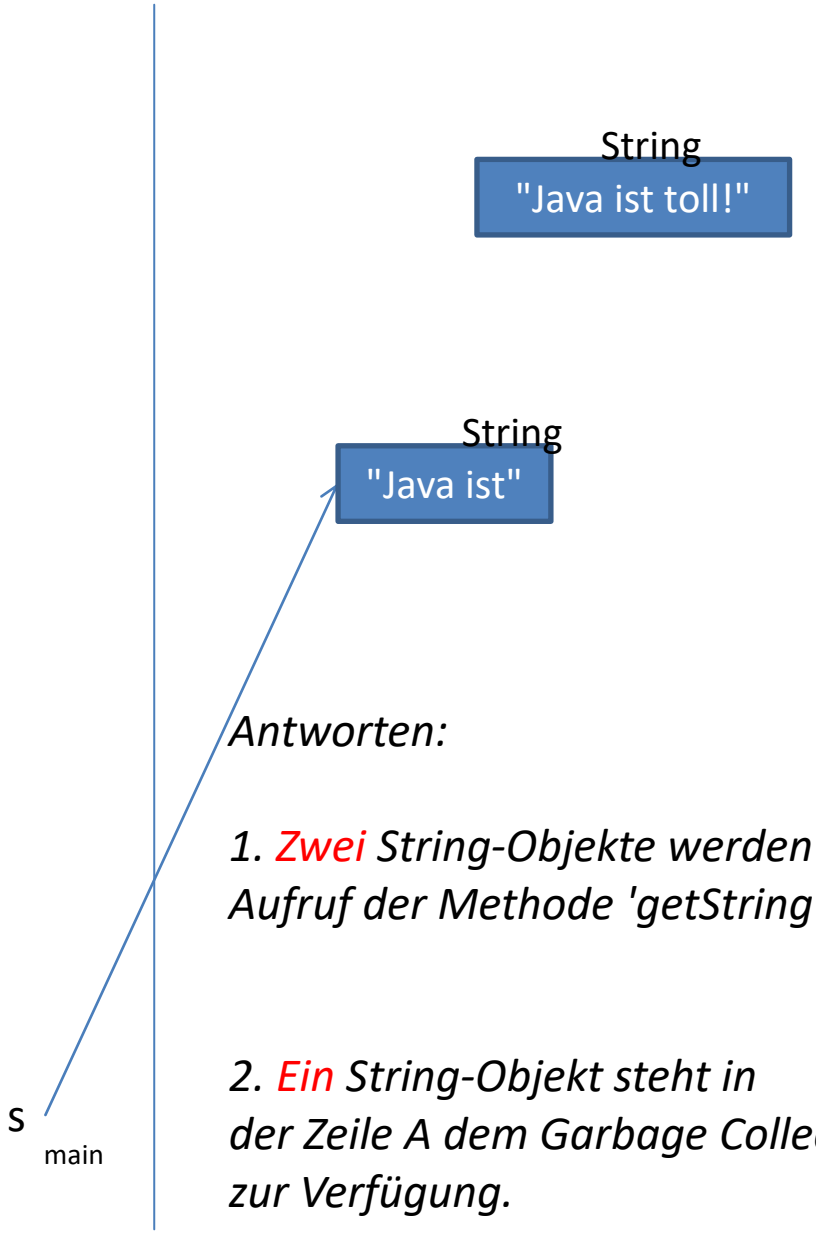



```
static String getString() {
   String s = "Java";

   s += " ist";
   s.concat("toll!");

   return s.toString();
}

public static void main(String[] args) {
    String s = getString();
   // Zeile A
}
```

String-Pool:

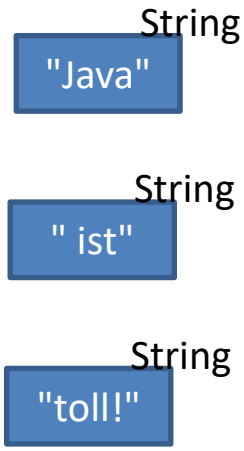


Die main startet




```
static String getString() {
   String s = "Java";

    s += " ist";
    s.concat("toll!");

    return s.toString();
}

public static void main(String[] args) {
     String s = getString();
    // Zeile A
}
```

String-Pool:



Antworten:

1. Zwei String-Objekte werden beim Aufruf der Methode 'getString' erzeugt.

2. Ein String-Objekt steht in der Zeile A dem Garbage Collector zur Verfügung.