

# Memory Diagrams: C# Generics Deep Dive

## Table of Contents

1. [Generic Method Memory Layout]
  2. [Generic Stack - Memory Structure]
  3. [Generic Class Inheritance]
  4. [Generic Interface Implementation]
  5. [Type Parameters at Runtime]
  6. [Stack vs Heap with Generics]
  7. [Complete Execution Flow]
- 

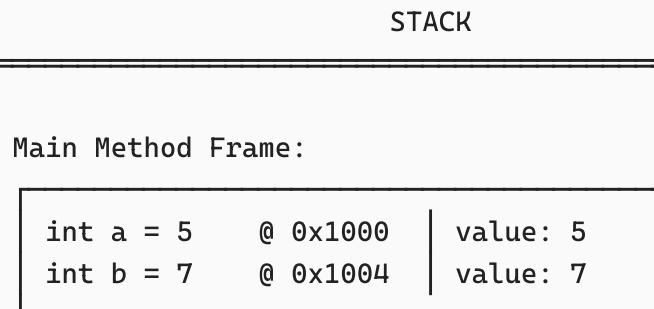
## Generic Method Memory Layout

### Code Example

```
public void Swap<T>(ref T x, ref T y)
{
    T temp = x;
    x = y;
    y = temp;
}
```

## Memory Diagram: Swap with Integers

BEFORE Swap<int>(ref a, ref b):



↓                    ↓  
|                    |  
Swap<int> Method Frame:

|                              |             |
|------------------------------|-------------|
| ref T x → points to @ 0x1000 | (reference) |
| ref T y → points to @ 0x1004 | (reference) |
| T temp @ 0x1008   value: ?   | (local var) |

Step 1: T temp = x;

|                          |
|--------------------------|
| temp copies value from x |
| temp = 5                 |

Stack After Step 1:

|                    |          |                 |
|--------------------|----------|-----------------|
| int a = 5 @ 0x1000 | value: 5 | ← x points here |
| int b = 7 @ 0x1004 | value: 7 | ← y points here |
| temp = 5 @ 0x1008  | value: 5 | ← copied value  |

Step 2: x = y;

|  |
|--|
| Value at address pointed by y (7)<br>is copied to address pointed by x |
|--|

Stack After Step 2:

|                    |          |            |
|--------------------|----------|------------|
| int a = 7 @ 0x1000 | value: 7 | ← CHANGED! |
| int b = 7 @ 0x1004 | value: 7 |            |
| temp = 5 @ 0x1008  | value: 5 |            |

Step 3: y = temp;

|  |
|--|
| Value of temp (5)<br>is copied to address pointed by y |
|--|

Stack After Step 3:

|                    |          |                 |
|--------------------|----------|-----------------|
| int a = 7 @ 0x1000 | value: 7 | ← x points here |
| int b = 5 @ 0x1004 | value: 5 | ← CHANGED!      |
| temp = 5 @ 0x1008  | value: 5 |                 |

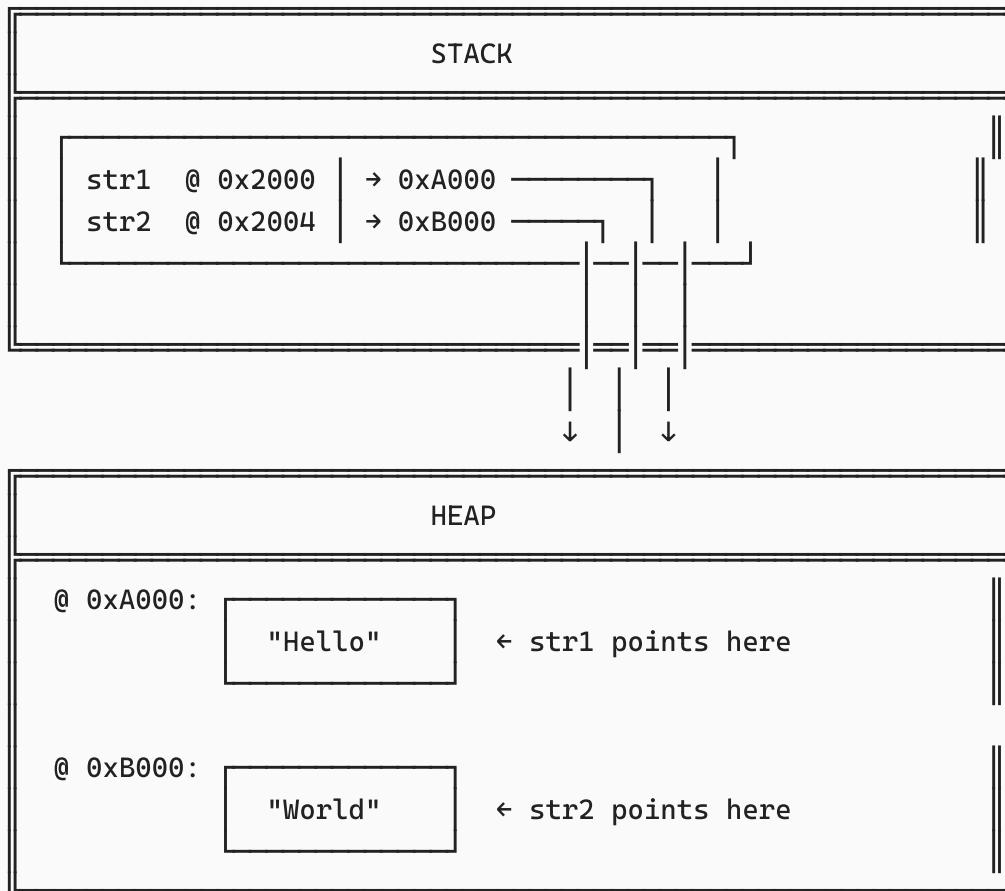
AFTER Swap – Method Frame Cleaned:

|           |          |          |            |
|-----------|----------|----------|------------|
| int a = 7 | @ 0x1000 | value: 7 | ✓ Swapped! |
| int b = 5 | @ 0x1004 | value: 5 | ✓ Swapped! |

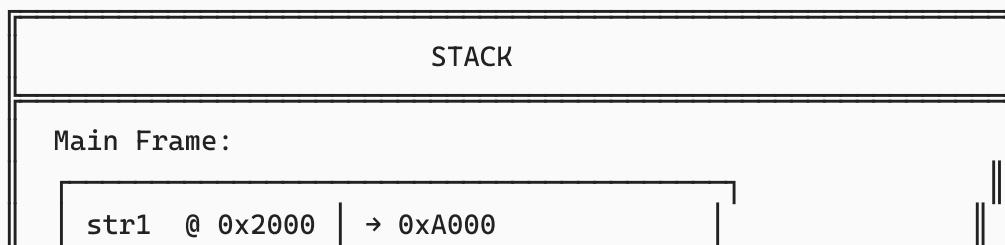
## Memory Diagram: Swap with Reference Types

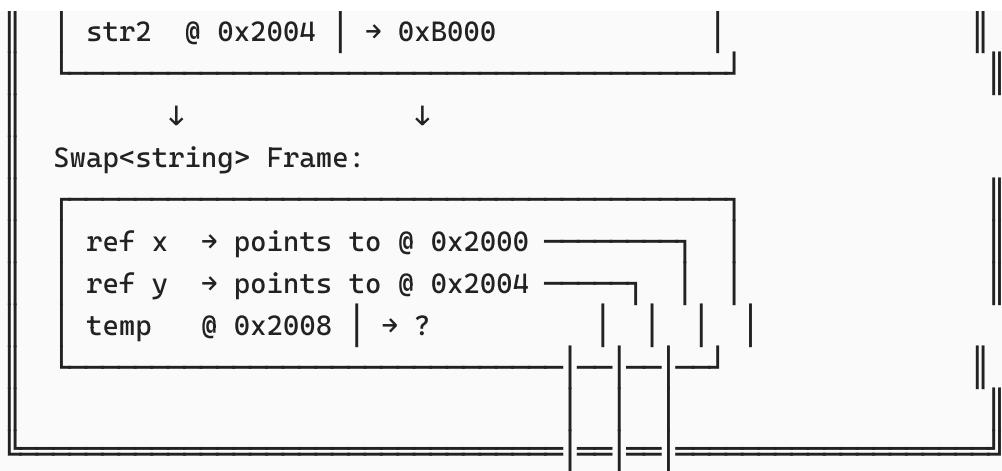
```
string str1 = "Hello";
string str2 = "World";
Swap<string>(ref str1, ref str2);
```

BEFORE Swap<string>(ref str1, ref str2):



Swap Method:





After Swap (Step 1: temp = x):

temp now holds 0xA000  
(copy of str1's reference)

Stack:

|      |          |          |
|------|----------|----------|
| str1 | @ 0x2000 | → 0xA000 |
| str2 | @ 0x2004 | → 0xB000 |
| temp | @ 0x2008 | → 0xA000 |

(copied)

After Step 2: x = y (str1 = str2):

|      |          |          |
|------|----------|----------|
| str1 | @ 0x2000 | → 0xB000 |
| str2 | @ 0x2004 | → 0xB000 |
| temp | @ 0x2008 | → 0xA000 |

← CHANGED!

After Step 3: y = temp (str2 = temp):

|      |          |          |
|------|----------|----------|
| str1 | @ 0x2000 | → 0xB000 |
| str2 | @ 0x2004 | → 0xA000 |
| temp | @ 0x2008 | → 0xA000 |

- ✓ Now points to "World"
- ✓ Now points to "Hello"

HEAP (unchanged - just references swapped):

|                   |                        |
|-------------------|------------------------|
| @ 0xA000: "Hello" | ← now str2 points here |
| @ 0xB000: "World" | ← now str1 points here |

# Generic Stack - Memory Structure

## Code

```
Stack<int> s = new Stack<int>(5);
s.Push(33);
s.Push(25);
s.Push(40);
```