From Java Code to Machine Code

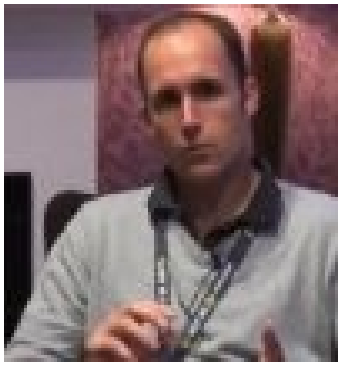
Understanding how to write optimizeable code

Chris Bailey
IBM Runtime Technologies

Chris Bailey

IBM Runtime Monitoring and Diagnostics Architect

- 14 years working with Java and JVM technologies
- 14 months working with Node.js and V8



Current Role(s):

- IBM Java monitoring and diagnostics
- Node Application Metrics project lead

Node Foundation Member
Benchmarking and Post-mortem WGs

JavaOne RockStar
Author on performance and memory analysis

baileyc@uk.ibm.com
chrisbaileyibm
cnbailey
@seabaylea
@Chris__Bailey

2

Anatomy of a Class File

Anatomy of a Java Class File

```
struct Class_File_Format {
    u4 magic_number;
    u2 minor_version;
    u2 major_version;

    u2 constant_pool_count;
    cp_info constant_pool[constant_pool_count - 1];

    u2 access_flags;
    u2 this_class;
    u2 super_class;

    u2 interfaces_count;
    u2 interfaces[interfaces_count];

    u2 fields_count;
    field_info fields[fields_count];

    u2 methods_count;
    method_info methods[methods_count];

    u2 attributes_count;
    attribute_info attributes[attributes_count];
}
```

Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
      0      11     0  this   LmyClass;
```

```
flags: ACC_PUBLIC, ACC_SUPER
Constant pool:
   #1 = Class              #2
   #2 = Utf8                 MyClass
   #3 = Class                #4
   #4 = Utf8                 java/lang/Object
   #5 = Utf8                 myField
   #6 = Utf8                 I
   #7 = Utf8                 <init>
   #8 = Utf8                 ()V
   #9 = Utf8                 Code
  #10 = Methodref            #4.#11
  #11 = NameAndType          #7:#8
  #12 = Fieldref             #1.#13
  #13 = NameAndType          #5:#6
  #14 = Utf8                 LineNumberTable
  #15 = Utf8                 LocalVariableTable
  #16 = Utf8                 this
  #17 = Utf8                 LmyClass;
  #18 = Utf8                 SourceFile
  #19 = Utf8                 MyClass.java
```

Constant Pool Resolution

Anatomy of a Java Class File: Constant Pool

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
      0      11     0  this    LmyClass;
```

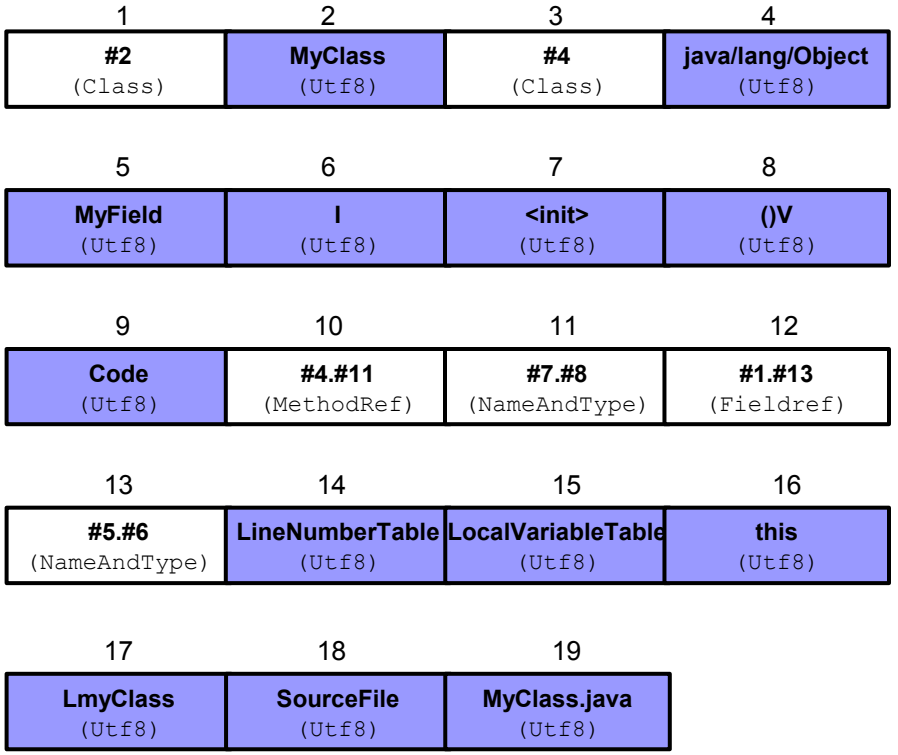
1	2	3	4
#2 (Class)	**MyClass** (Utf8)	**#4** (Class)	**java/lang/Object** (Utf8)
5	6	7	8
MyField (Utf8)	**I** (Utf8)	**<init>** (Utf8)	**()V** (Utf8)
9	10	11	12
Code (Utf8)	**#4.#11** (MethodRef)	**#7.#8** (NameAndType)	**#1.#13** (Fieldref)
13	14	15	16
#5.#6 (NameAndType)	**LineNumberTable** (Utf8)	**LocalVariableTable** (Utf8)	**this** (Utf8)
17	18	19	
LmyClass (Utf8)	**SourceFile** (Utf8)	**MyClass.java** (Utf8)	

Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11     0  this   LmyClass;
```

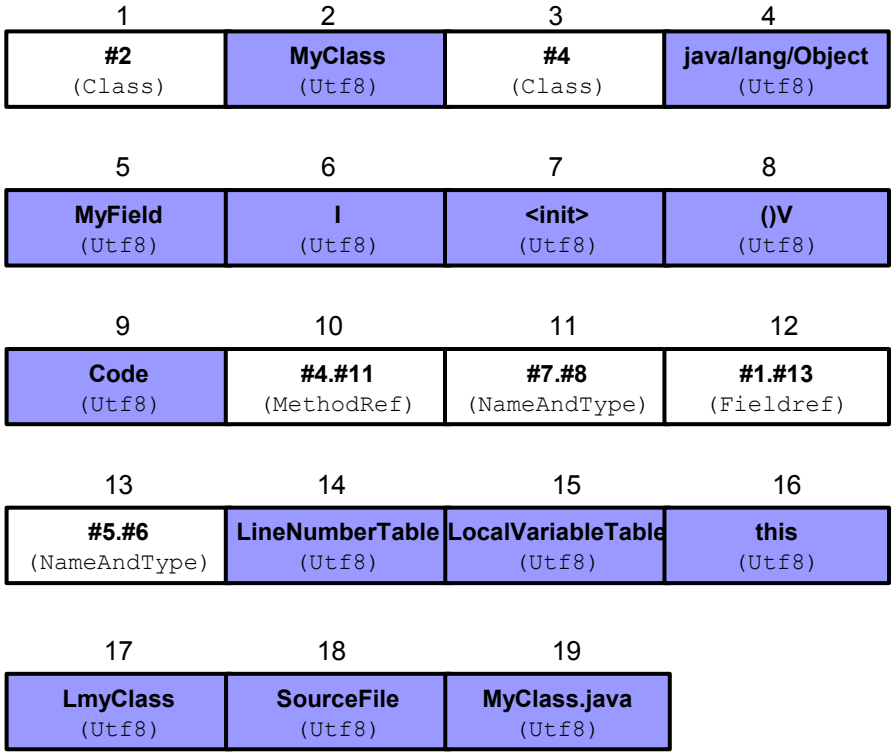


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11     0  this   LmyClass;
```

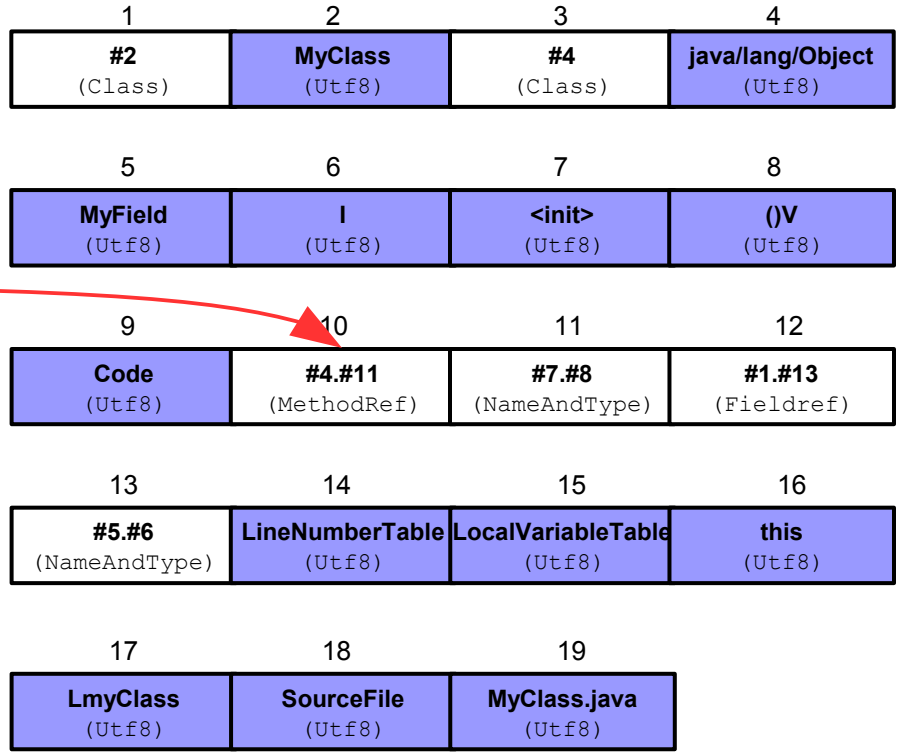


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

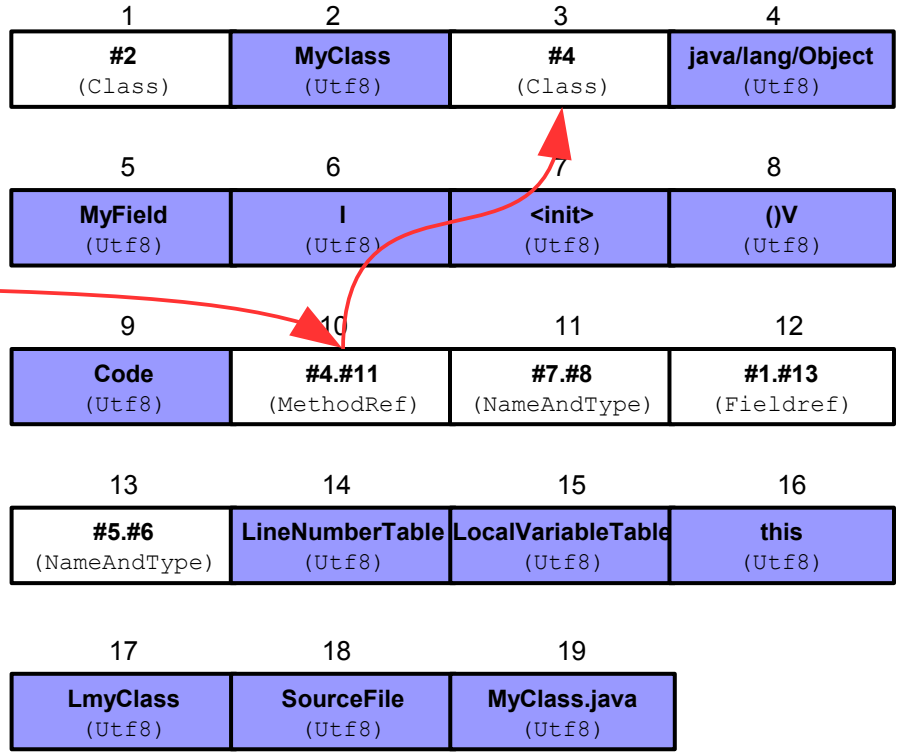


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

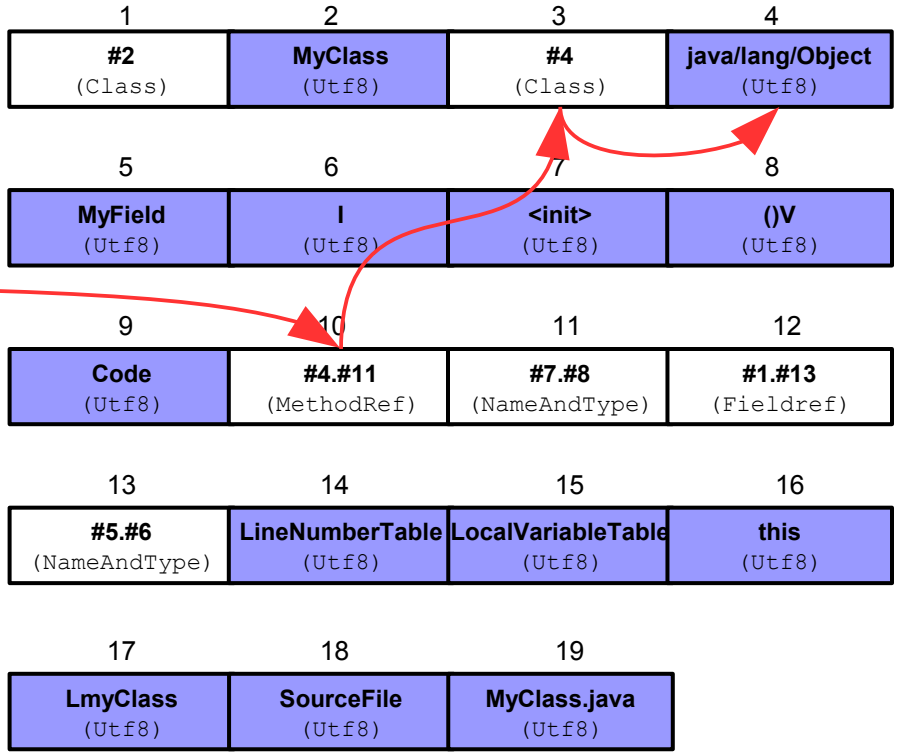


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

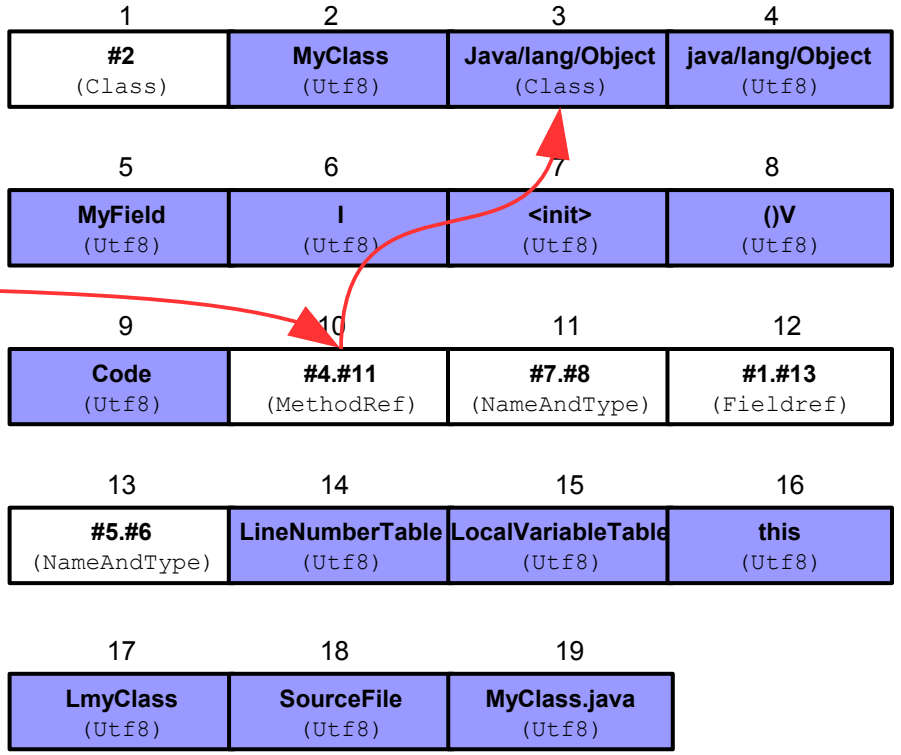


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

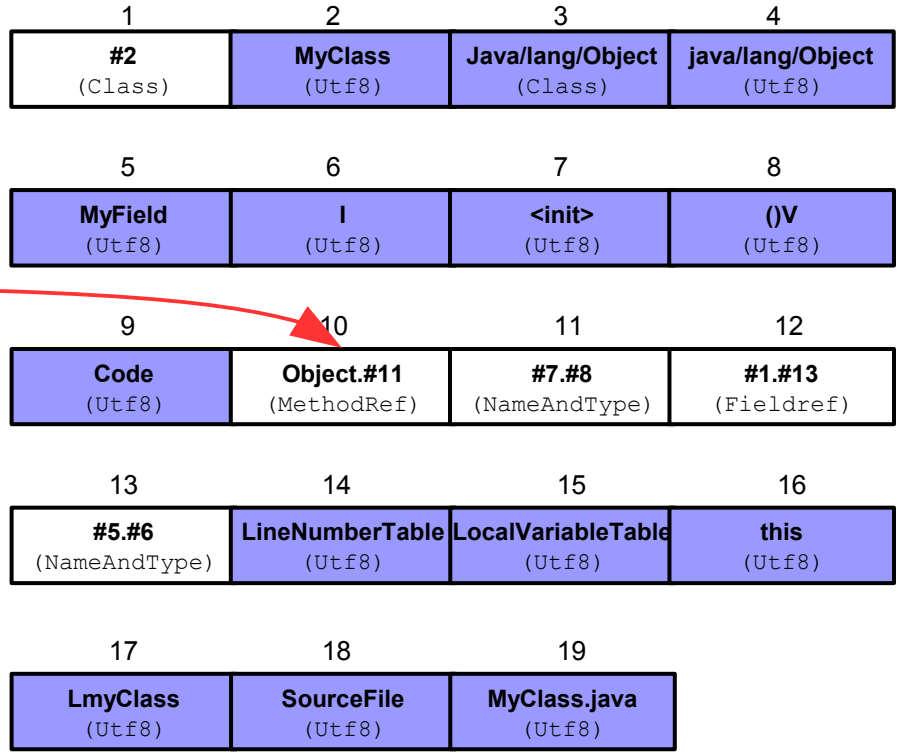


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

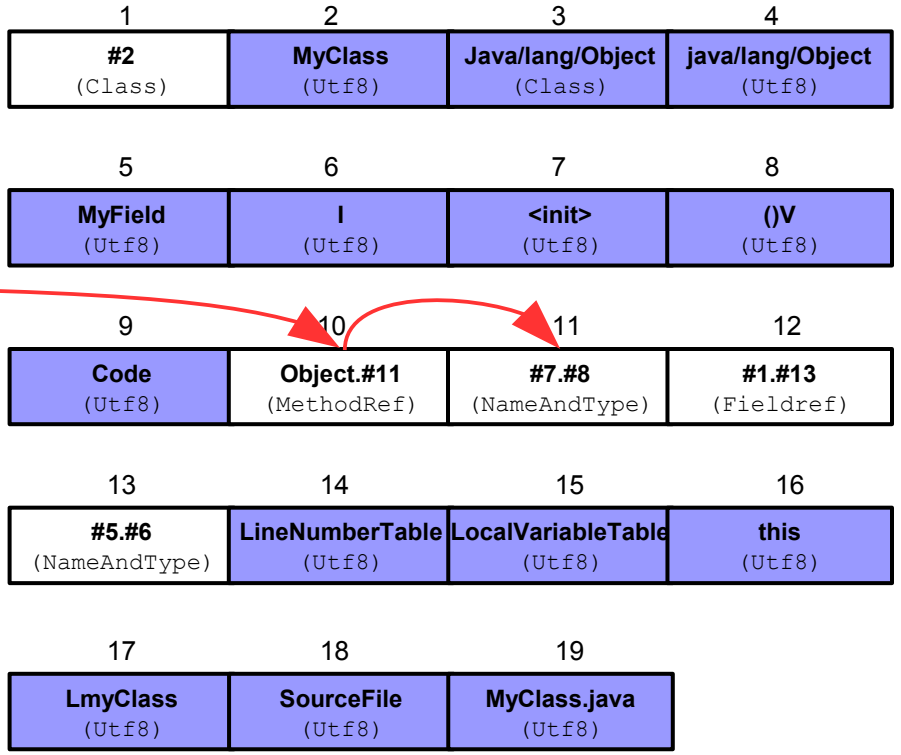


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

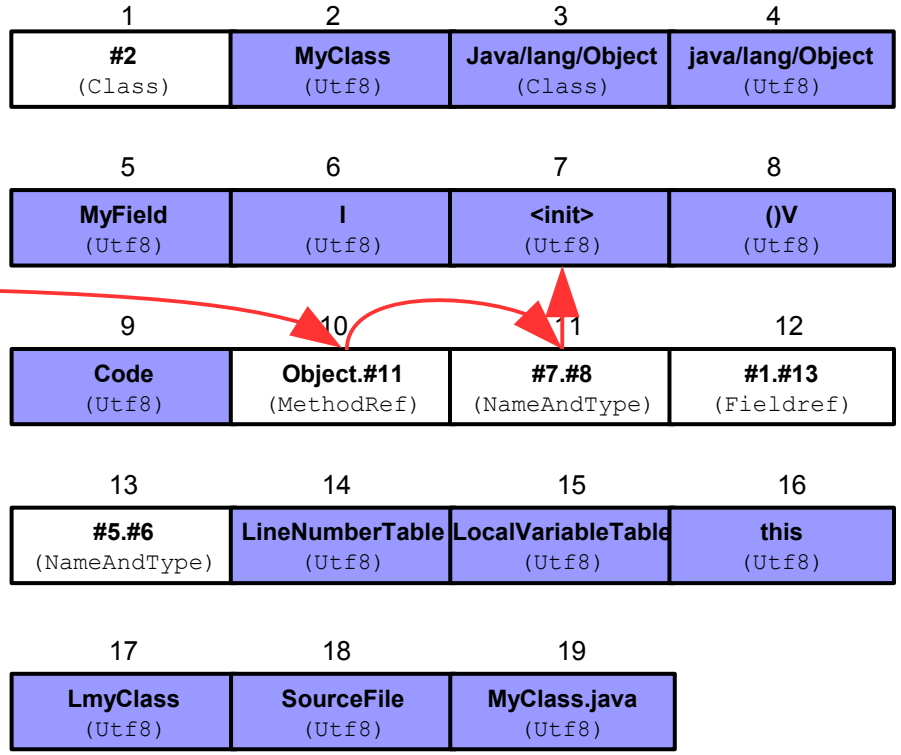


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

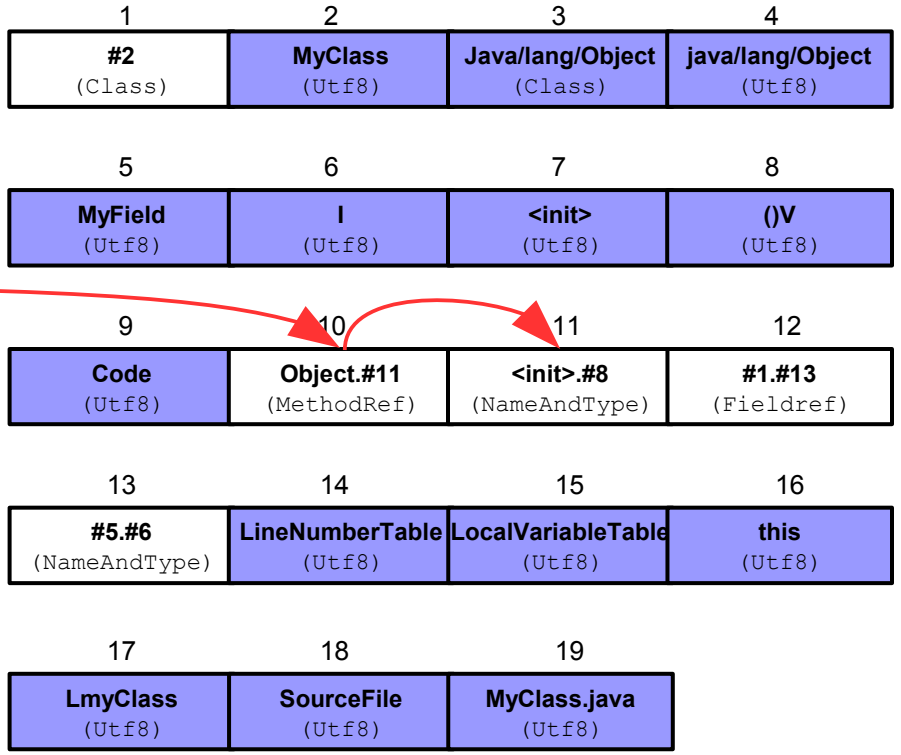


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

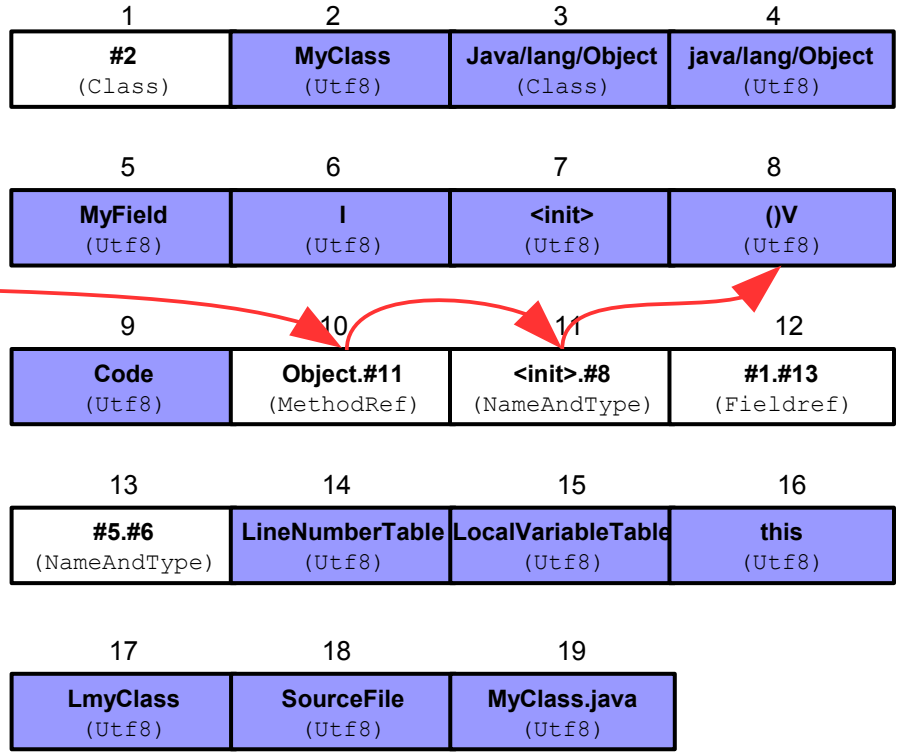


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

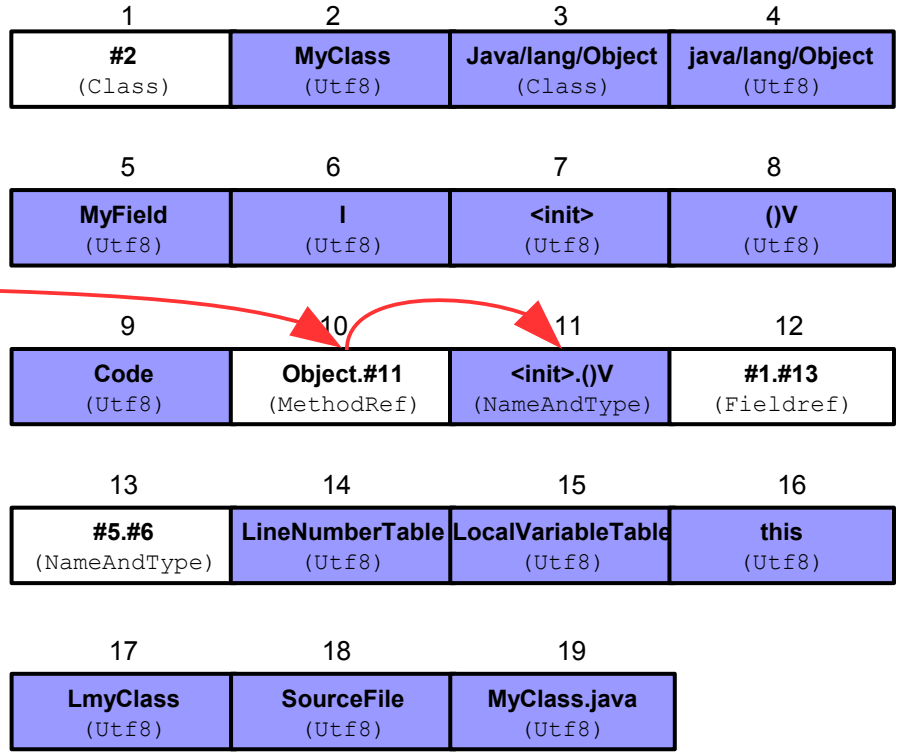


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

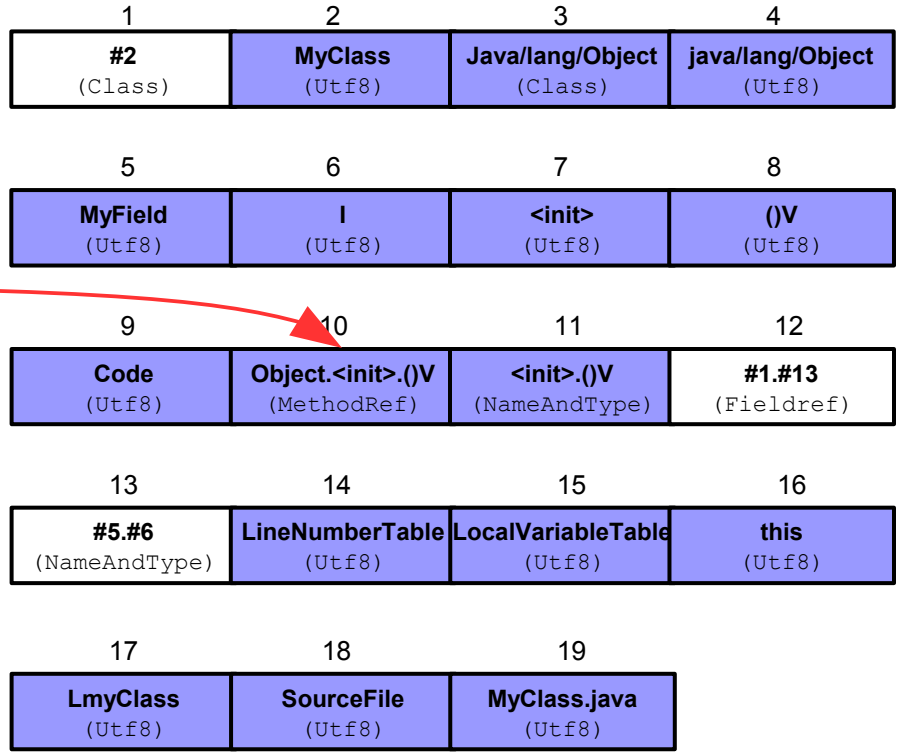


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial #10
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

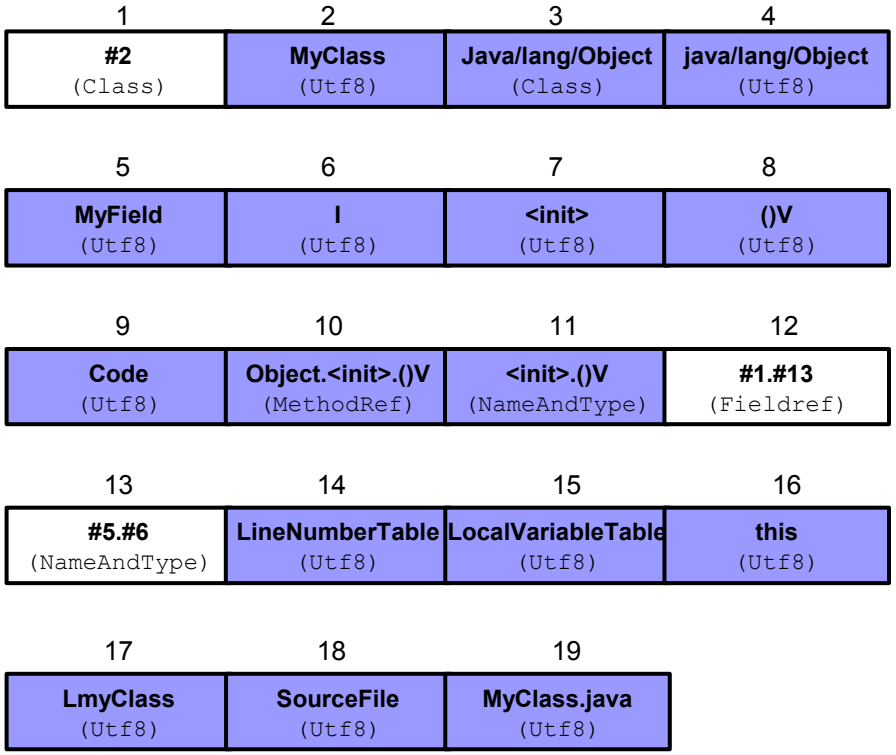


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11     0  this   LmyClass;
```

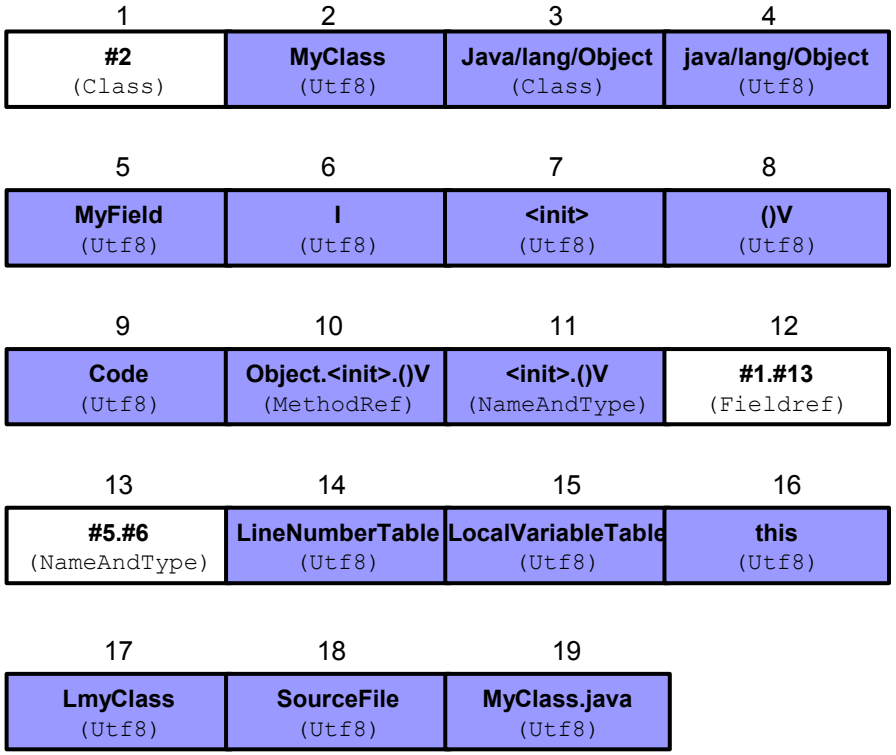


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11     0  this    LmyClass;
```

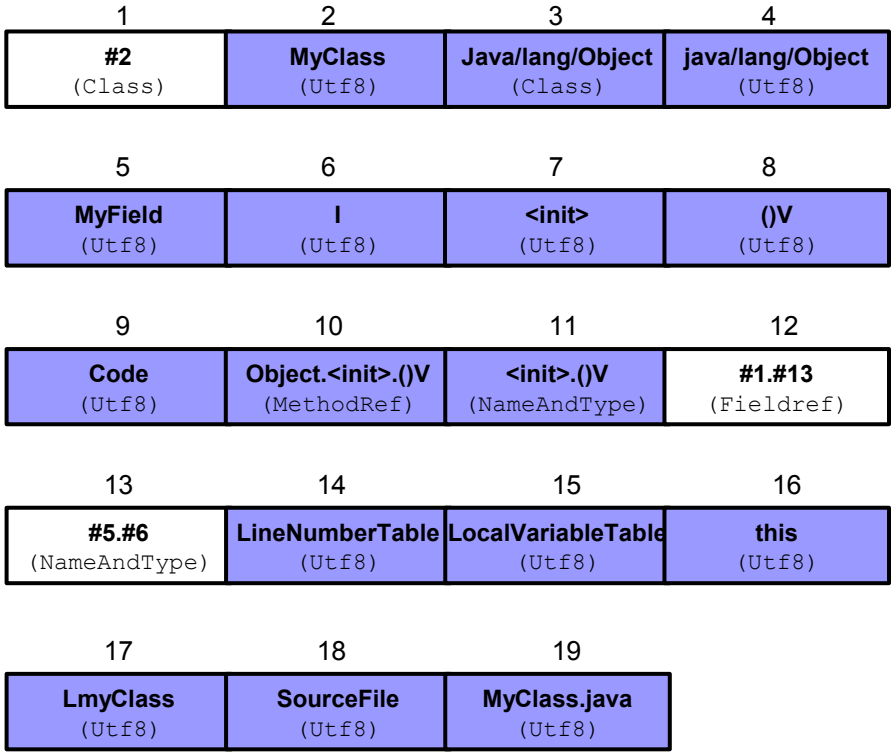


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```

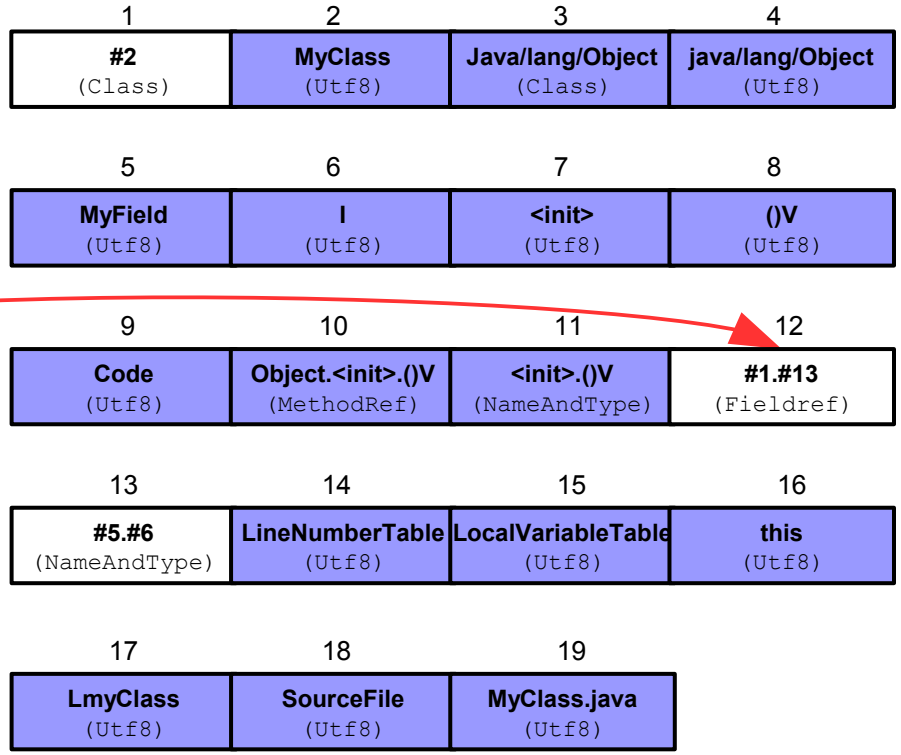


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

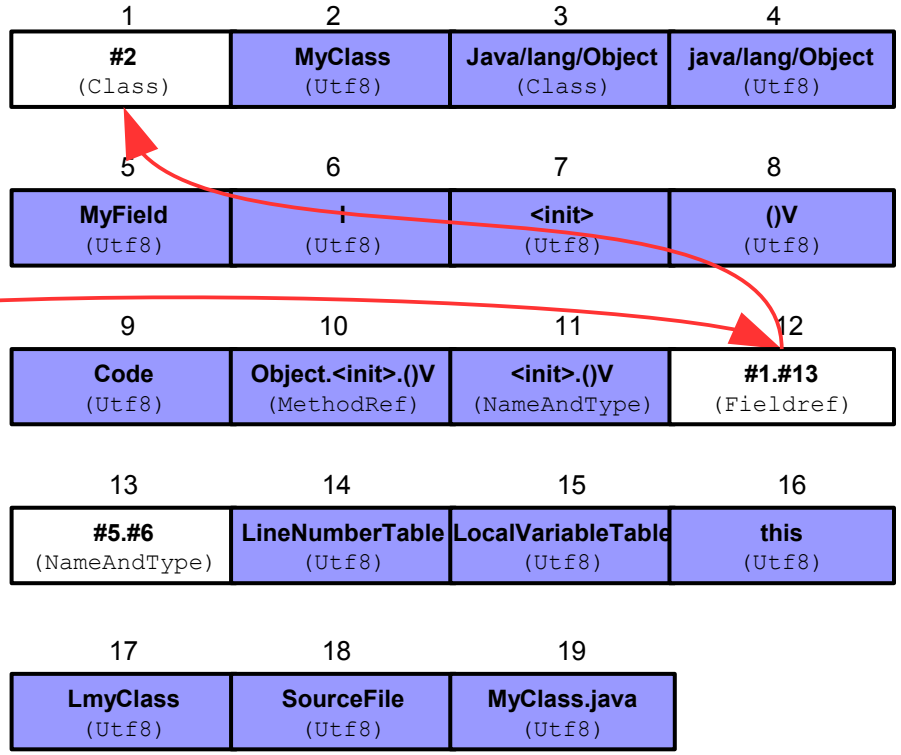


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

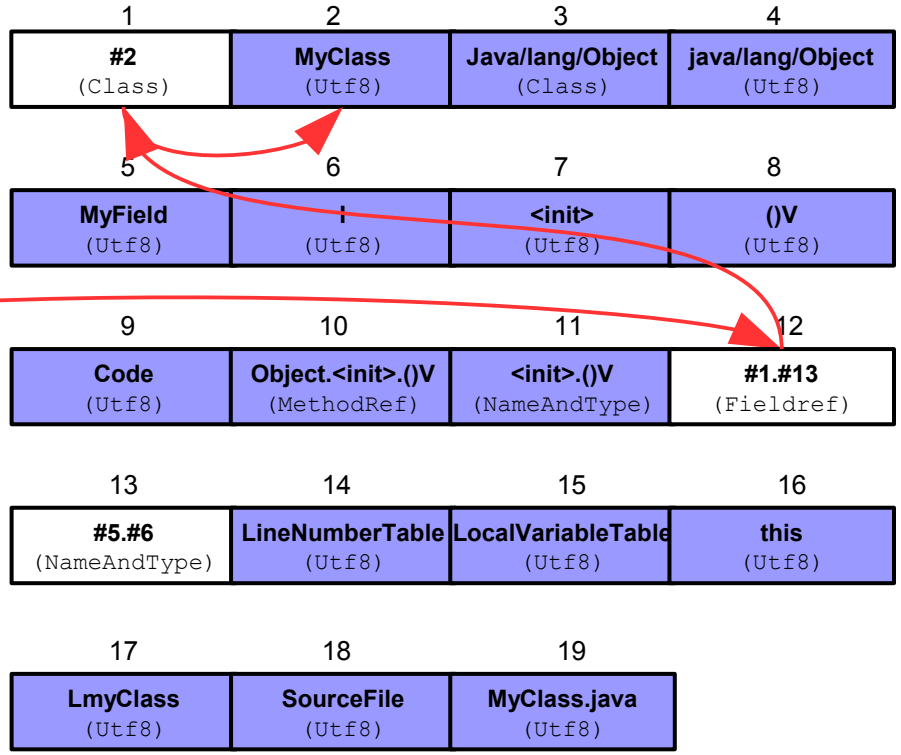


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```

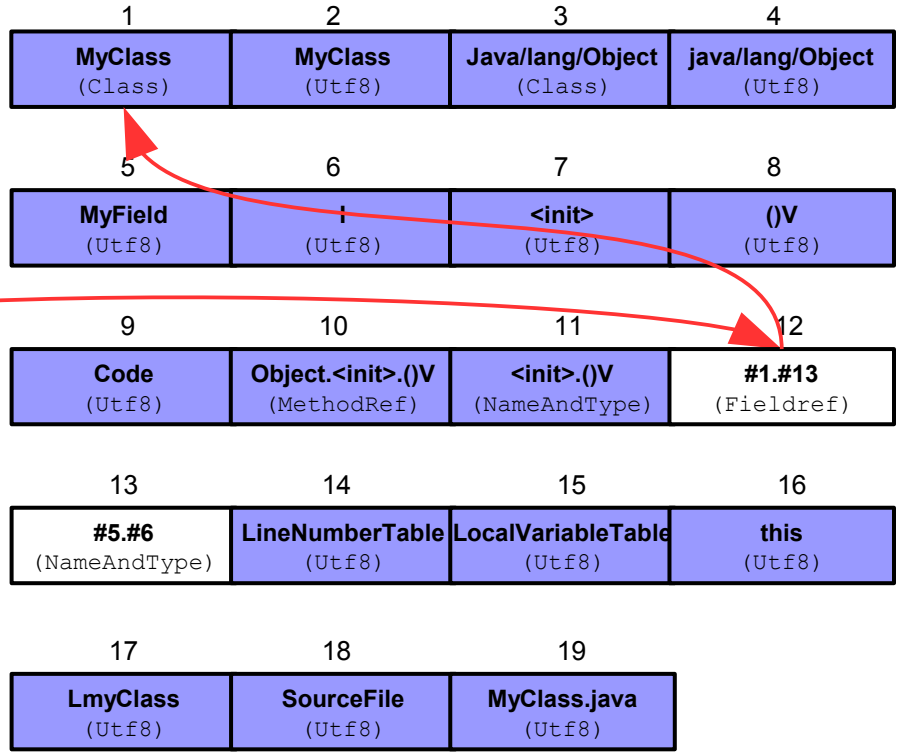


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```

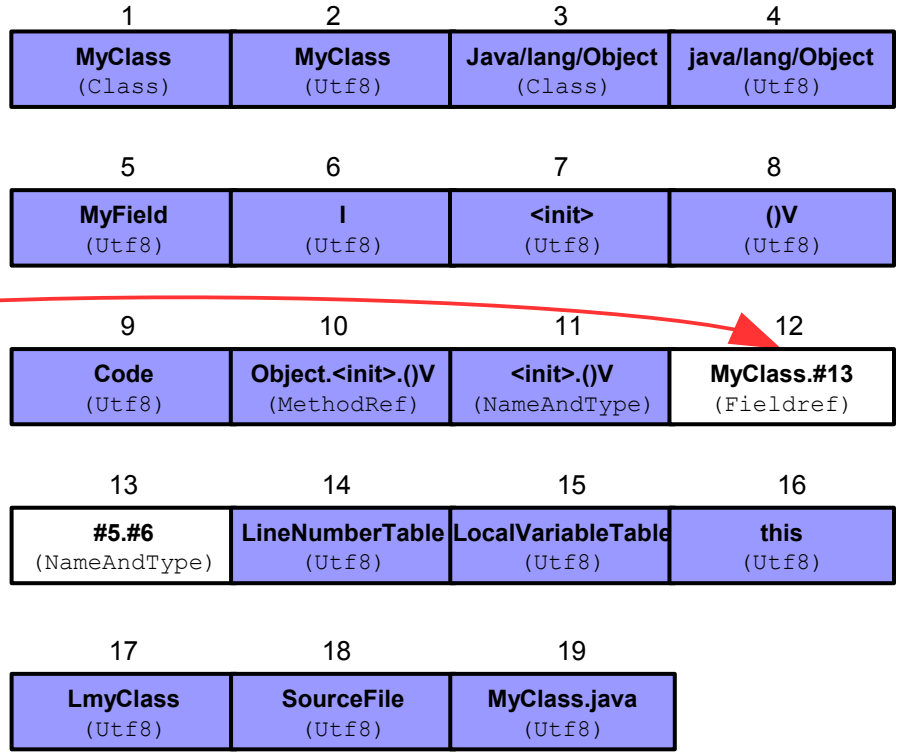


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

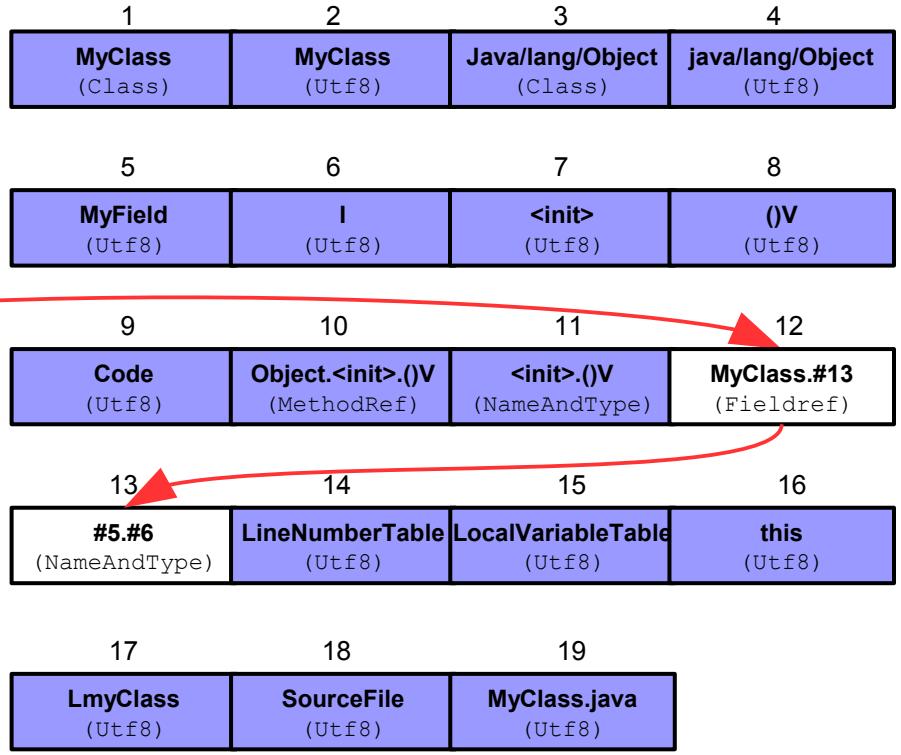


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

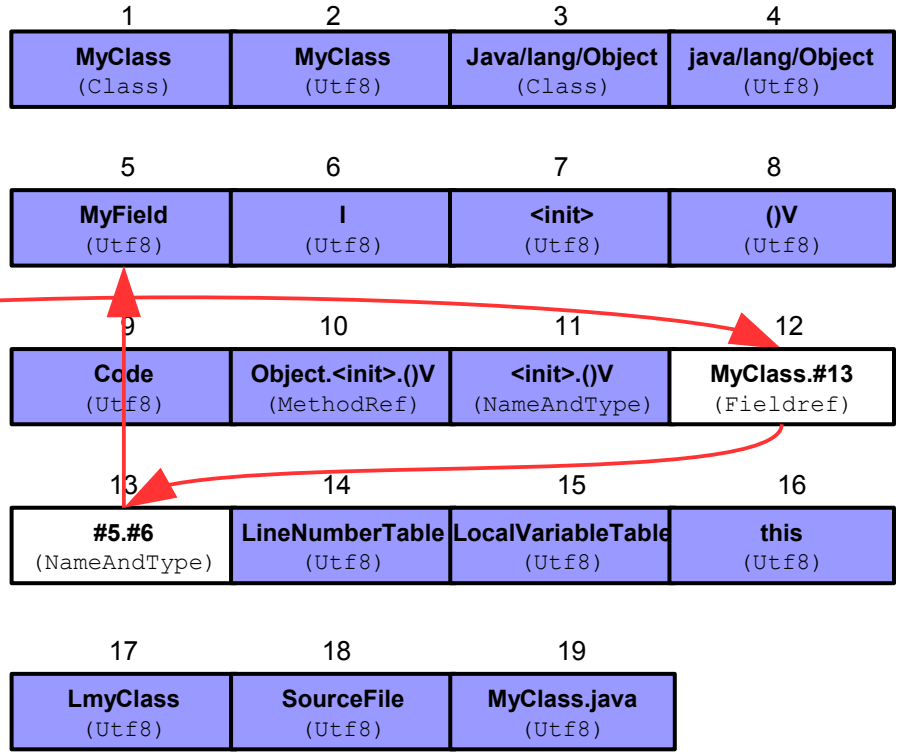


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

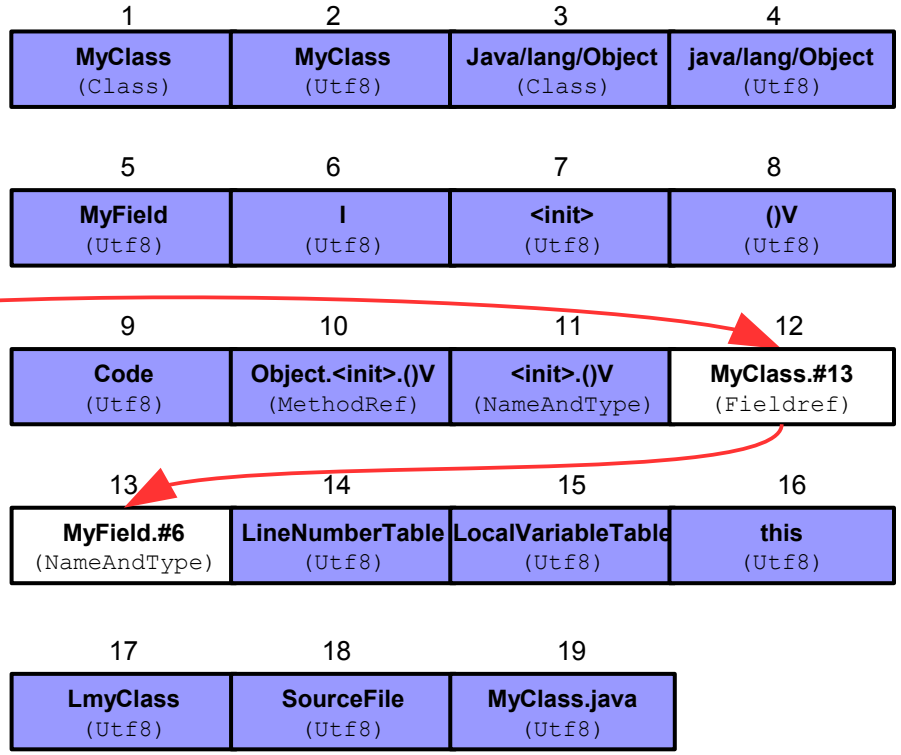


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```

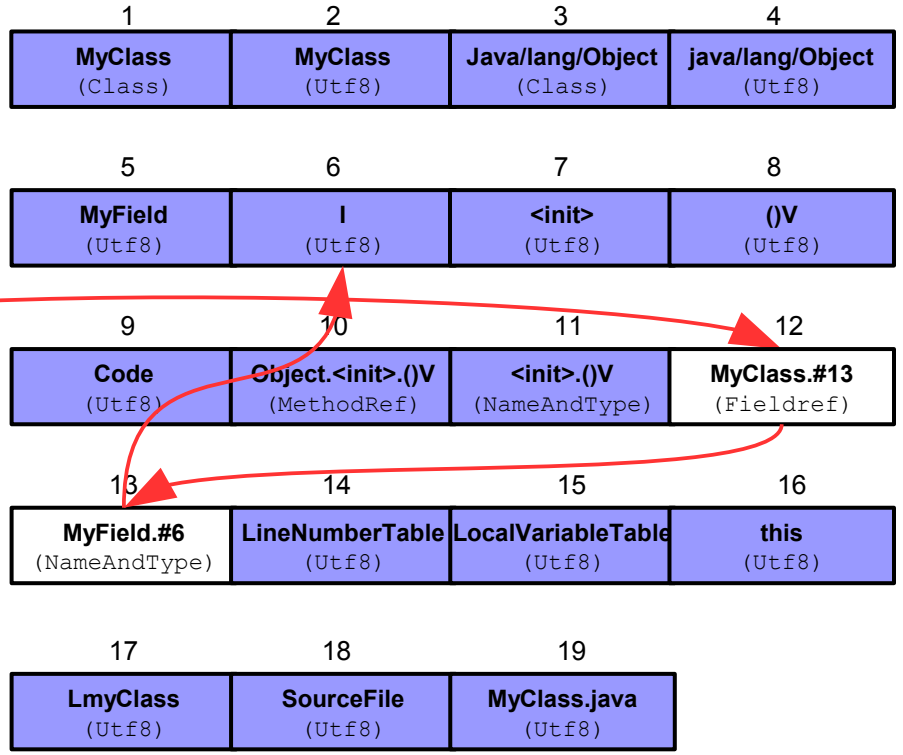


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
 Code:
 stack=2, locals=1, args_size=1
   0: aload_0
   1: invokespecial Object.<init>.()V
   4: aload_0
   5: bipush        48
   7: putfield      #12
  10: return
 LineNumberTable:
   line 2: 0
   line 3: 4
   line 2: 10
 LocalVariableTable:
   0       11      0  this    LmyClass;
```

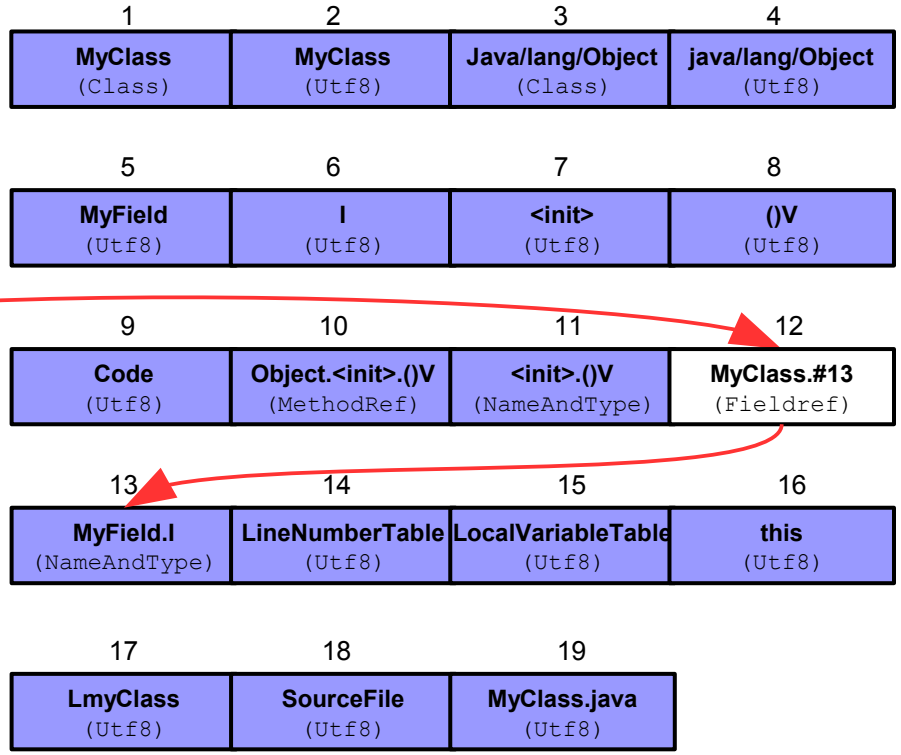


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

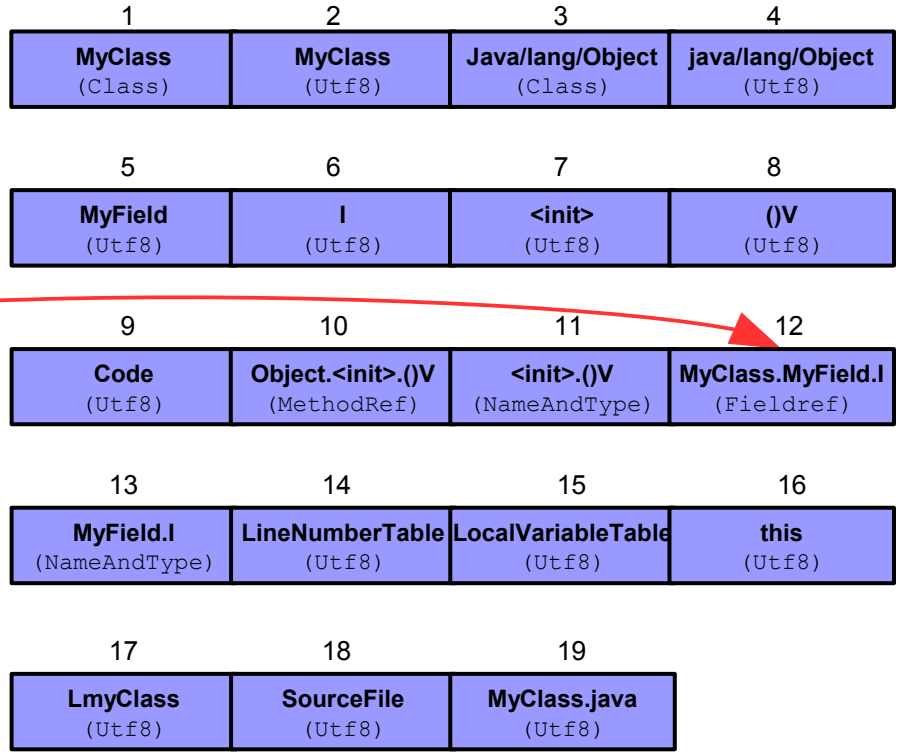


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      #12
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```

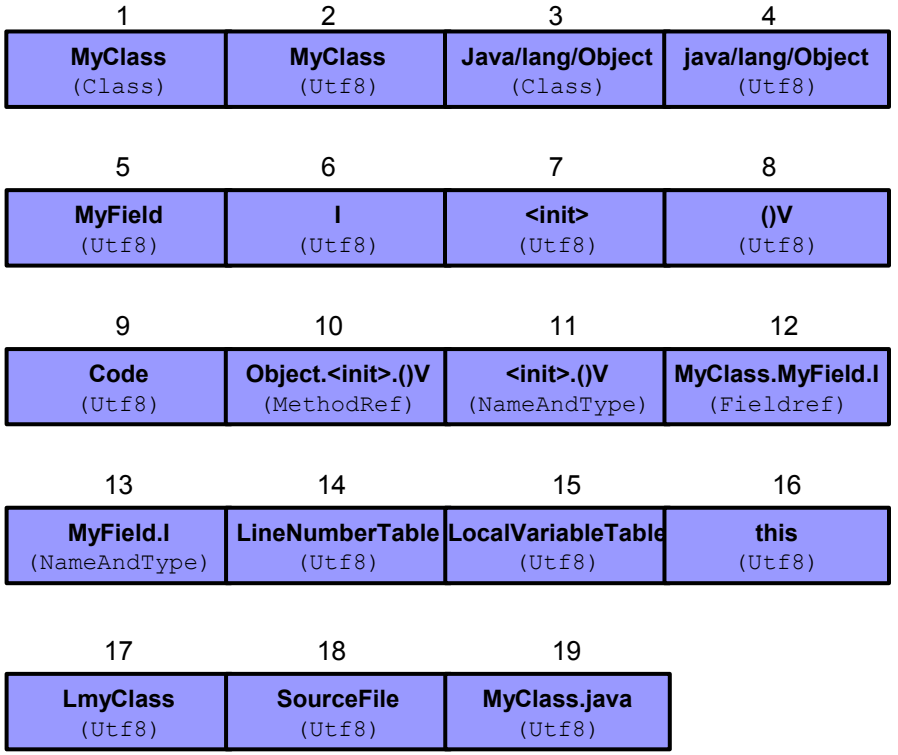


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```

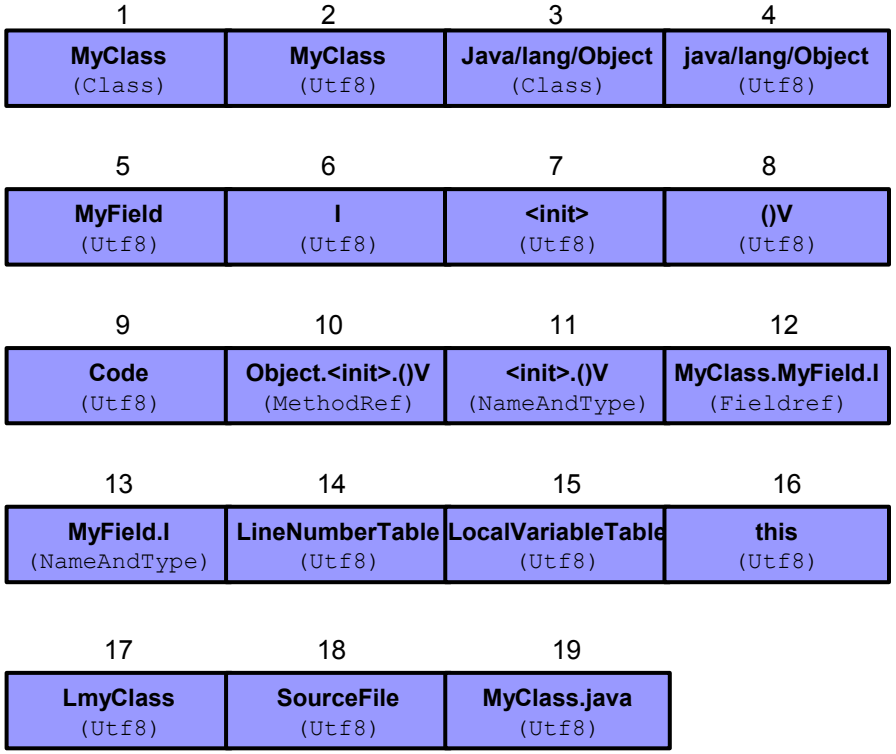


Anatomy of a Java Class File

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
      0      11     0  this   LmyClass;
```



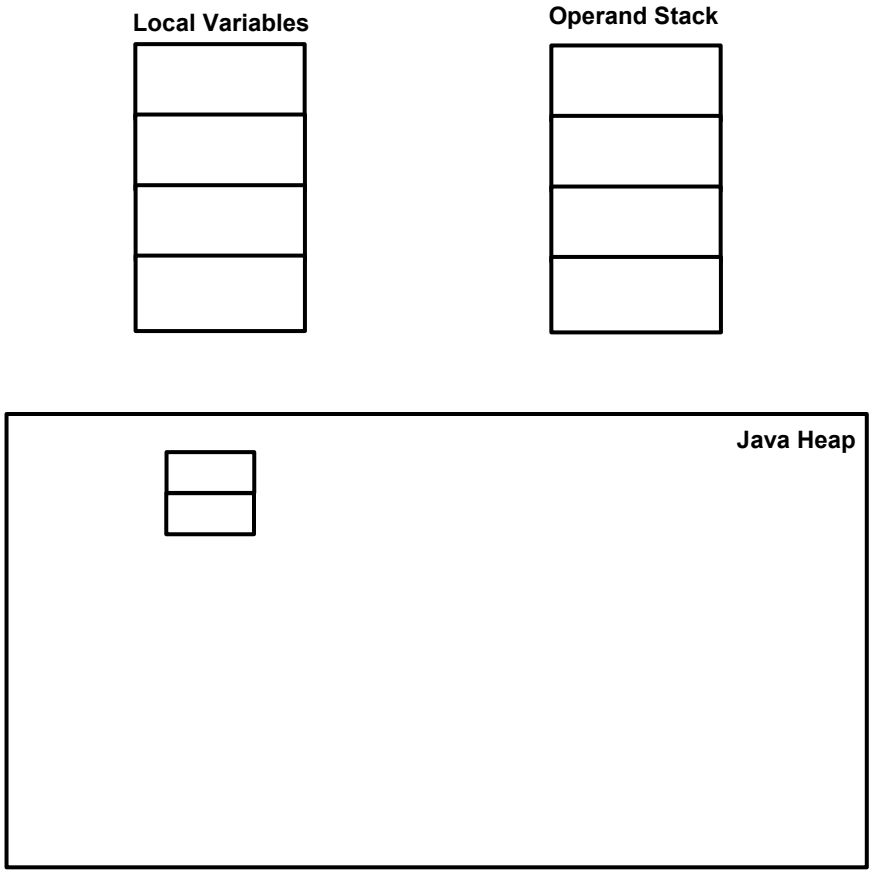
Java ByteCode Execution

Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11     0  this   LmyClass;
```

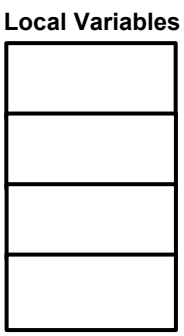


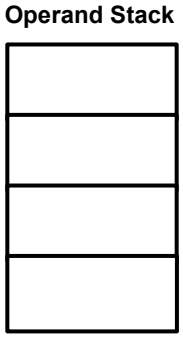
Executing Java Code

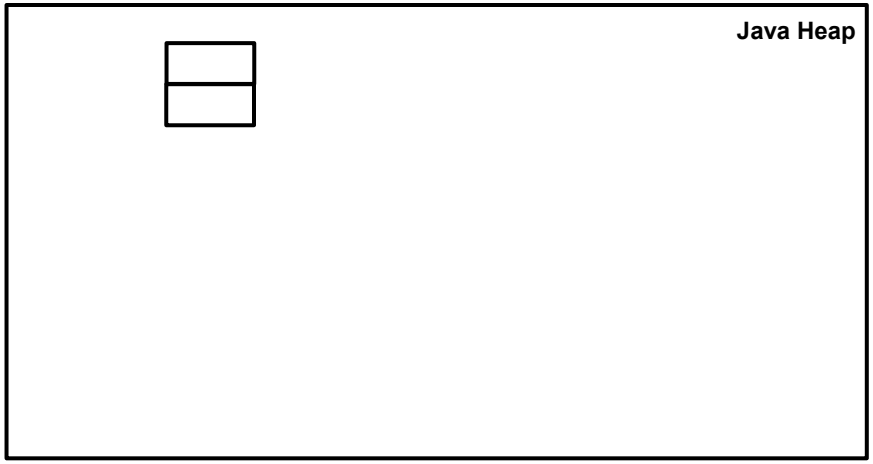
```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11     0  this   LmyClass;
```





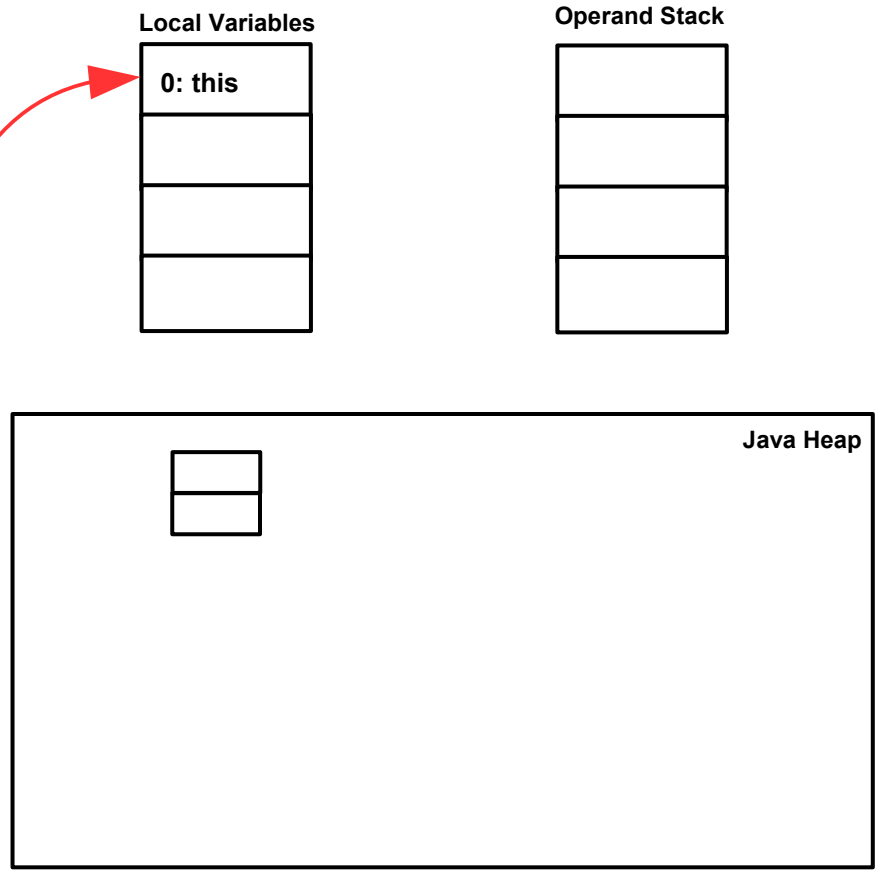


Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11      0  this    LmyClass;
```

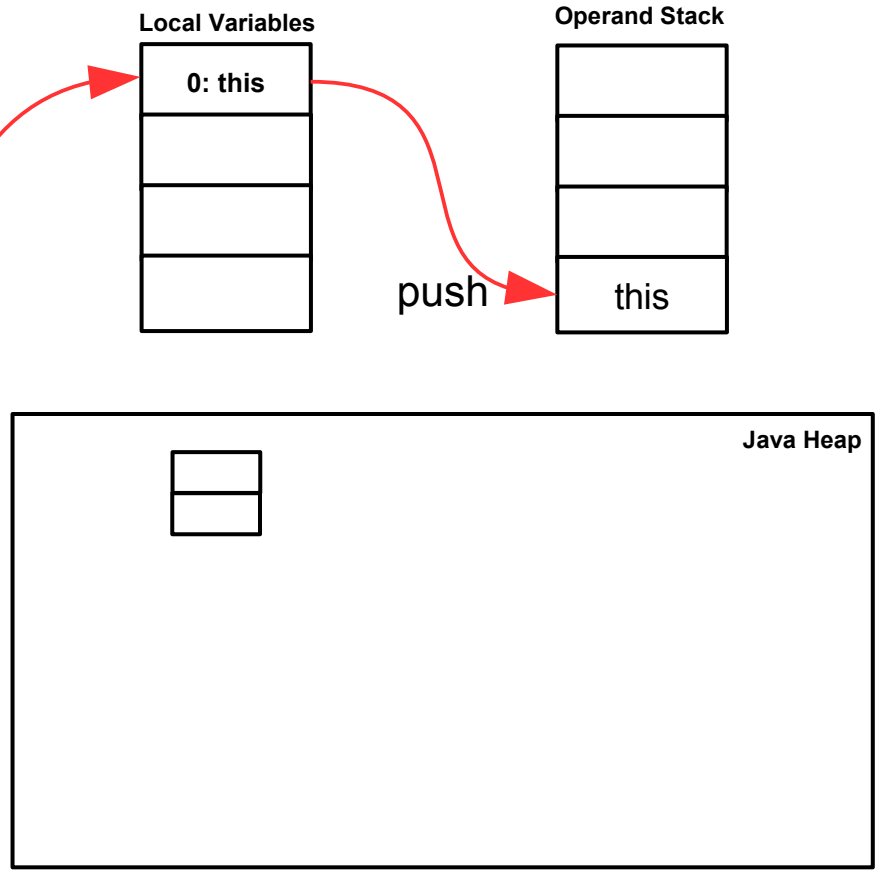


Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11      0  this    LmyClass;
```

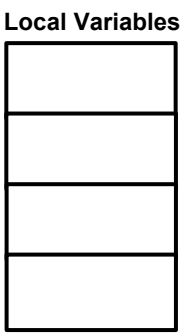


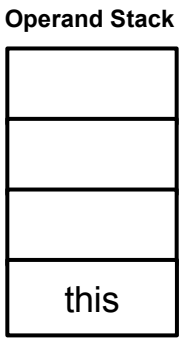
Executing Java Code

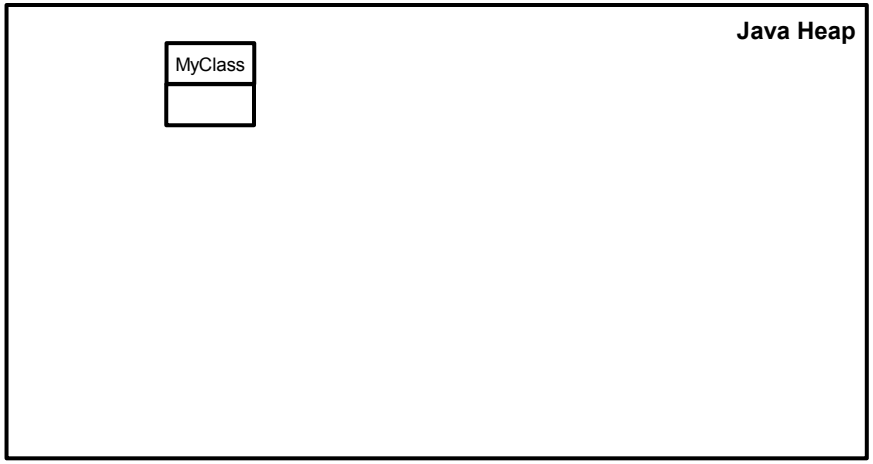
```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0      11     0  this   LmyClass;
```





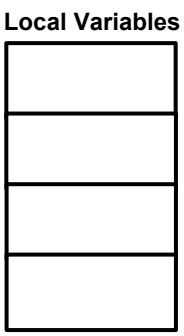


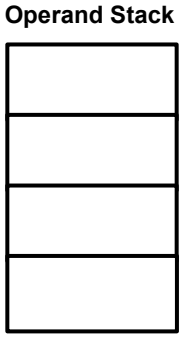
Executing Java Code

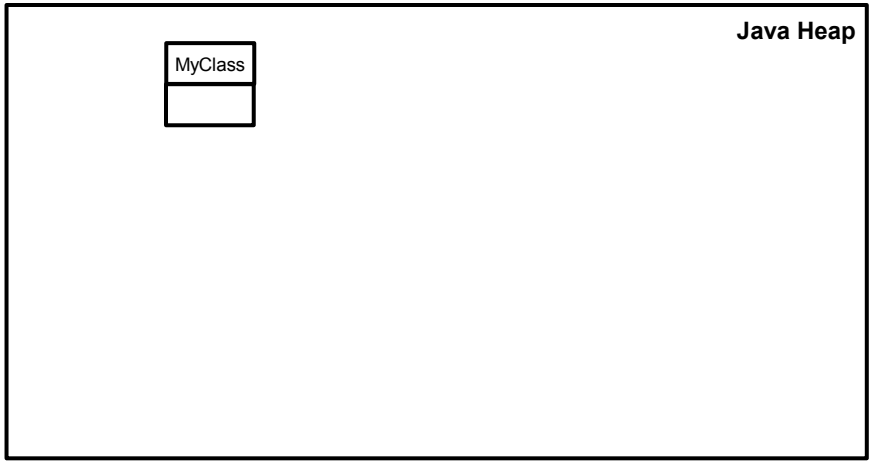
```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this   LmyClass;
```





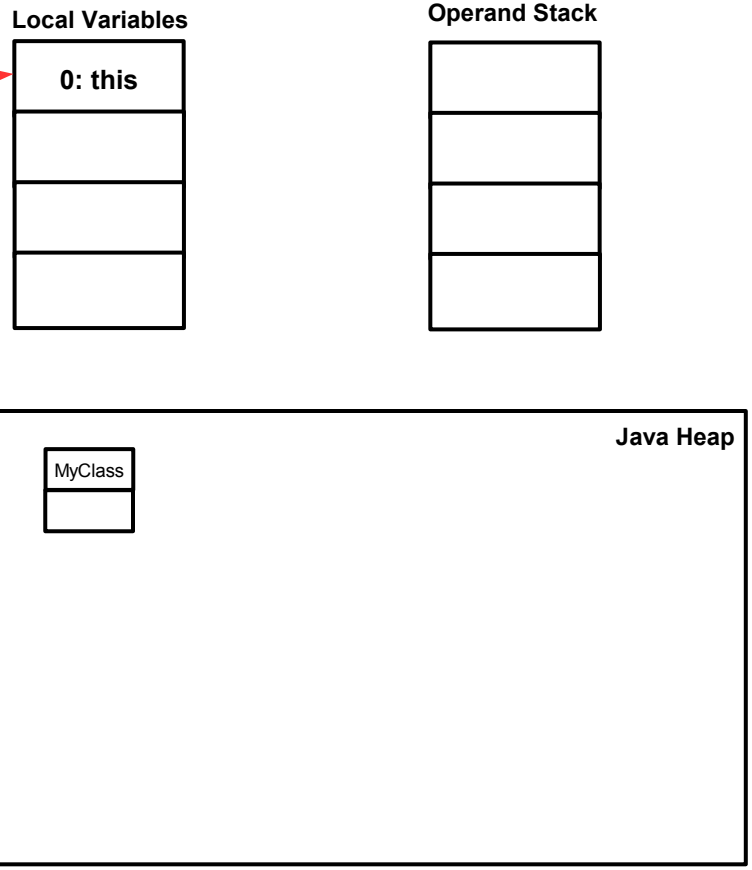


Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11     0  this    LmyClass;
```

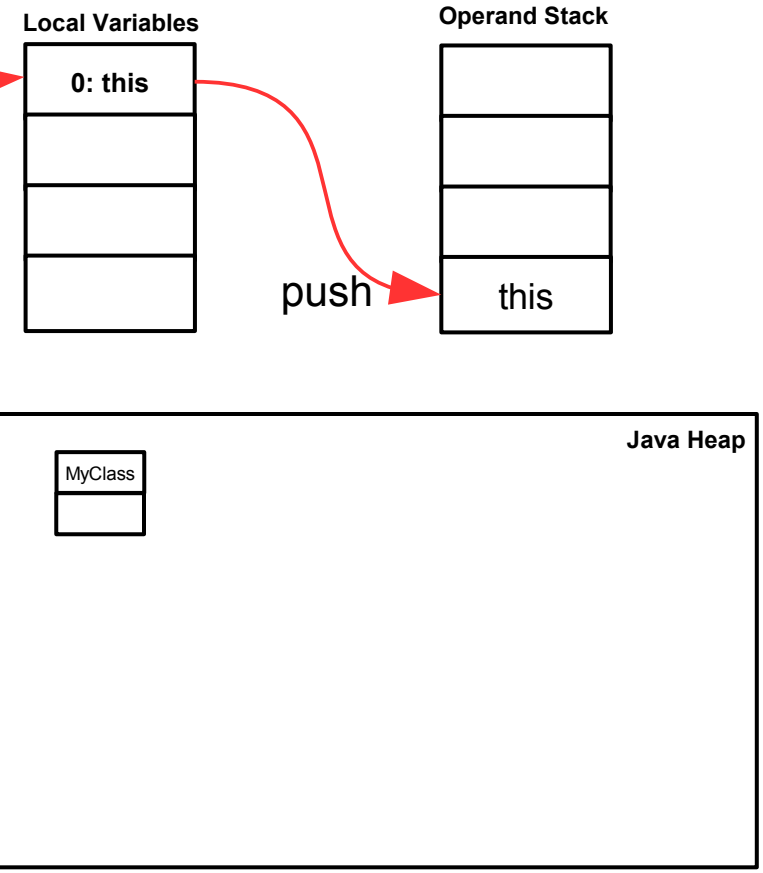


Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11      0  this    LmyClass;
```

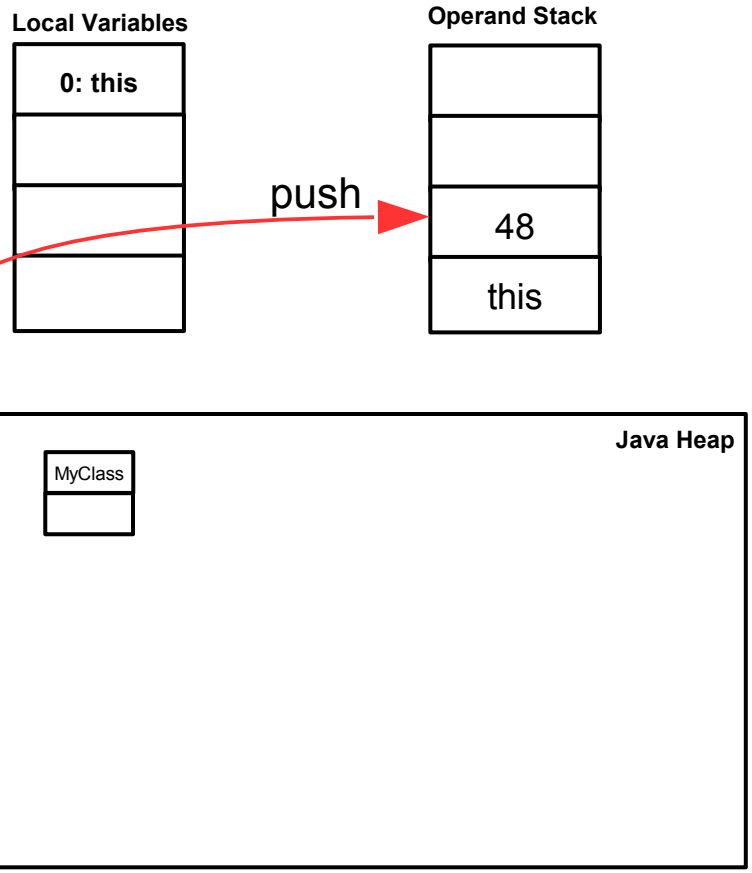


Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
 Code:
 stack=2, locals=1, args_size=1
   0: aload_0
   1: invokespecial Object.<init>.()V
   4: aload_0
   5: bipush        48
   7: putfield      MyClass.MyField.I
  10: return
 LineNumberTable:
   line 2: 0
   line 3: 4
   line 2: 10
 LocalVariableTable:
   0       11      0  this   LmyClass;
```

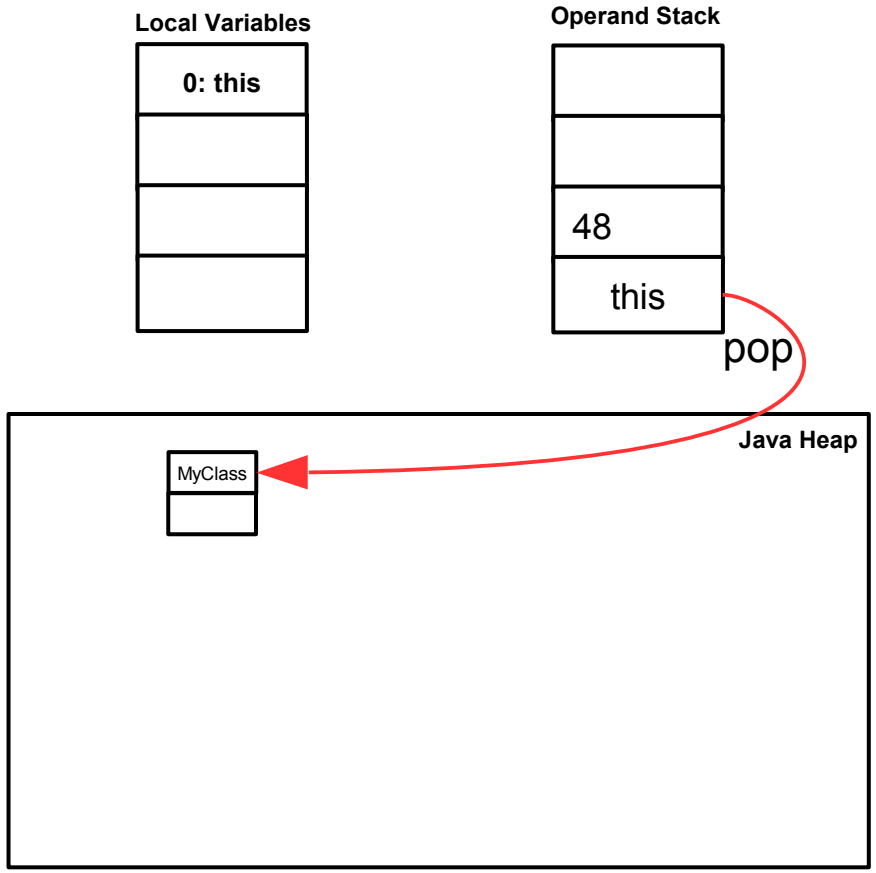


Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11     0  this   LmyClass;
```

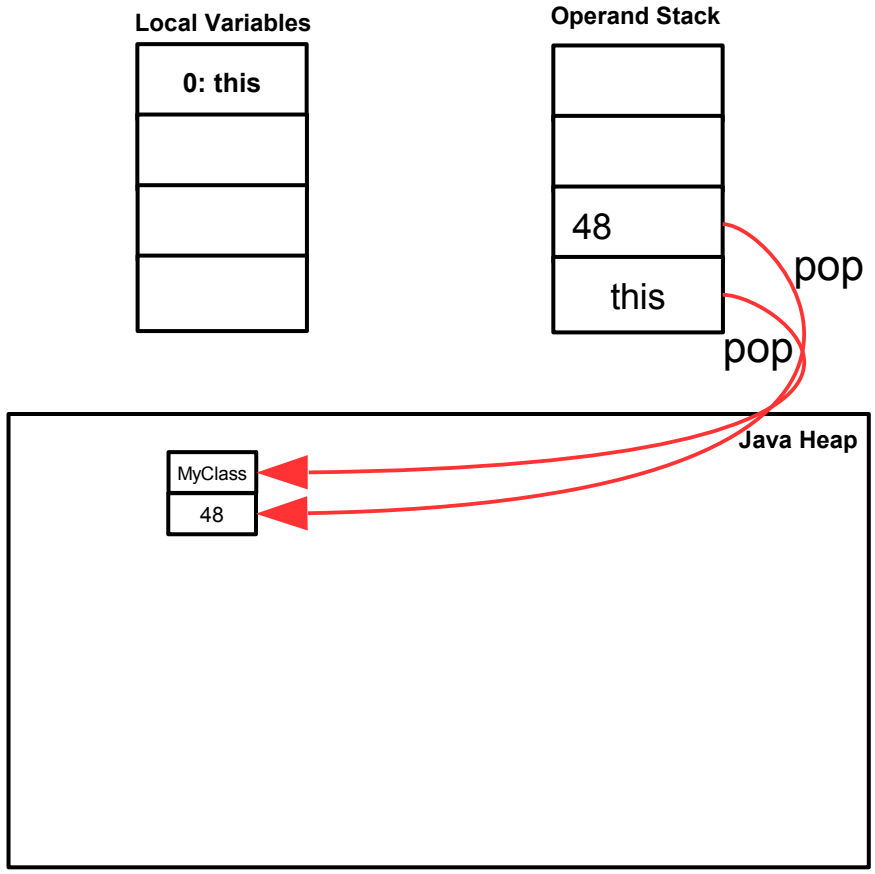


Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11     0  this   LmyClass;
```

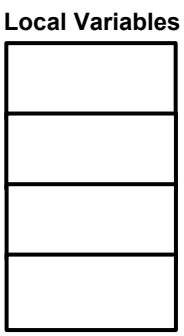


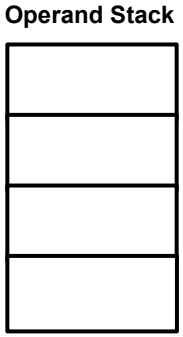
Executing Java Code

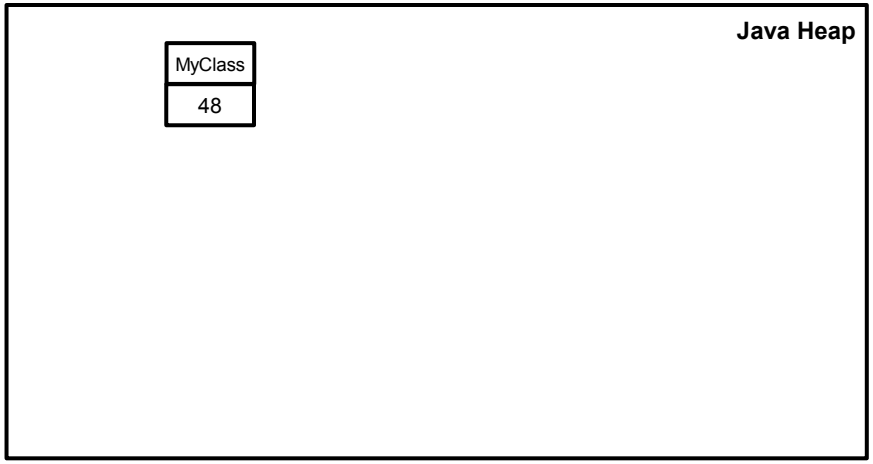
```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
   flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
     0      11     0  this   LmyClass;
```







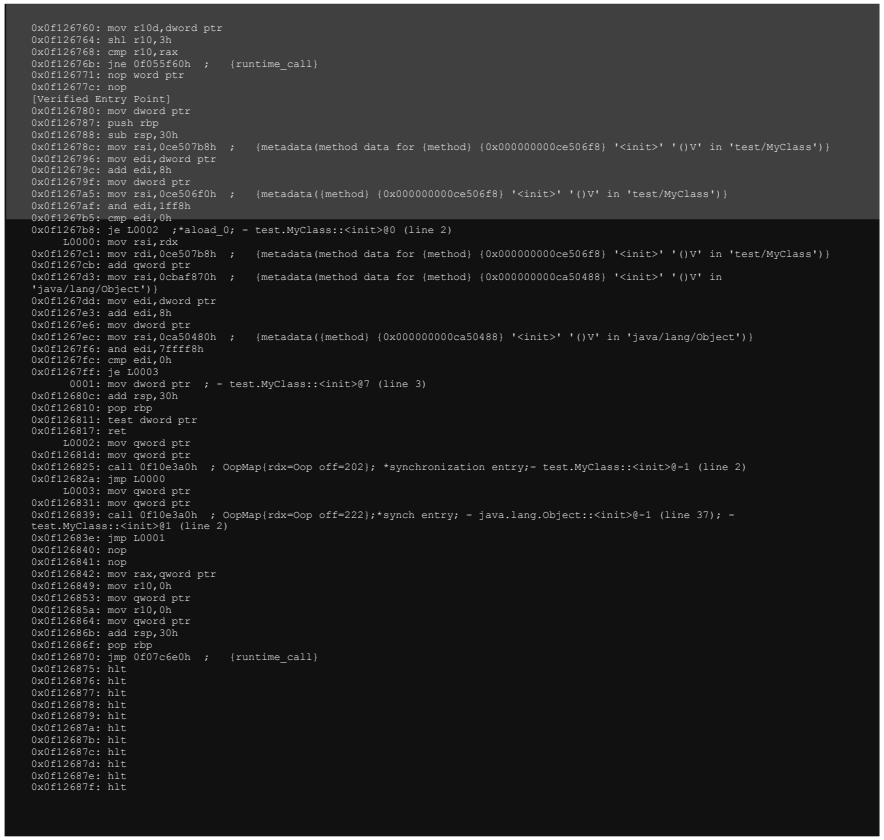
Machine Code

Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
 Code:
 stack=2, locals=1, args_size=1
   0: aload_0
   1: invokespecial Object.<init>.()V
   4: aload_0
   5: bipush        48
   7: putfield      MyClass.MyField.I
  10: return
 LineNumberTable:
   line 2: 0
   line 3: 4
   line 2: 10
 LocalVariableTable:
   0       11      0  this    LmyClass;
```



Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
 Code:
 stack=2, locals=1, args_size=1
   0: aload_0
   1: invokespecial Object.<init>.()V
   4: aload_0
   5: bipush        48
   7: putfield      MyClass.MyField.I
  10: return
 LineNumberTable:
   line 2: 0
   line 3: 4
   line 2: 10
 LocalVariableTable:
   0       11      0  this    LmyClass;
```

```
0x0f126760: mov r10d,dword ptr
0x0f126764: shl r10,3h
0x0f126768: cmp r10,rax
0x0f12676b: jne 0f055f60h  ;   {runtime_call}
0x0f126771: nop word ptr
0x0f12677c: nop
[Verified Entry Point]
0x0f126780: mov dword ptr
0x0f126787: push rbp
0x0f126788: sub rsp,30h
0x0f12678c: mov rsi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f126796: mov edi,dword ptr
0x0f12679c: add edi,8h
0x0f12679f: mov dword ptr
0x0f1267a5: mov rsi,0ce506f0h  ;   {metadata({method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267af: and edi,1ff8h
0x0f1267b5: cmp edi,0h
0x0f1267b8: je L0002  ;*aload_0; - test.MyClass::<init>@0 (line 2)
      L0000: mov rsi,rdx
0x0f1267c1: mov rdi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267cb: add qword ptr
0x0f1267d3: mov rsi,0cbaf870h  ;   {metadata(method data for {method} {0x000000000ca50488} '<init>' '()V' in
'java/lang/Object')}
0x0f1267dd: mov edi,dword ptr
0x0f1267e3: add edi,8h
0x0f1267e6: mov dword ptr
0x0f1267ec: mov rsi,0ca50480h  ;   {metadata({method} {0x000000000ca50488} '<init>' '()V' in 'java/lang/Object')}
0x0f1267f6: and edi,7ffff8h
0x0f1267fc: cmp edi,0h
0x0f1267ff: je L0003
      0001: mov dword ptr  ; - test.MyClass::<init>@7 (line 3)
0x0f12680c: add rsp,30h
0x0f126810: pop rbp
0x0f126811: test dword ptr
0x0f126817: ret
      L0002: mov qword ptr
0x0f12681d: mov qword ptr
0x0f126825: call 0f10e3a0h  ; OopMap{rdx=Oop off=202}; *synchronization entry;- test.MyClass::<init>@-1 (line 2)
0x0f12682a: jmp L0000
      L0003: mov qword ptr
0x0f126831: mov qword ptr
0x0f126839: call 0f10e3a0h  ; OopMap{rdx=Oop off=222};*synch entry; - java.lang.Object::<init>@-1 (line 37); -
test.MyClass::<init>@1 (line 2)
0x0f12683e: jmp L0001
0x0f126840: nop
0x0f126841: nop
0x0f126842: mov rax,qword ptr
0x0f126849: mov r10,0h
0x0f126853: mov qword ptr
0x0f12685a: mov r10,0h
0x0f126864: mov qword ptr
0x0f12686b: add rsp,30h
0x0f12686f: pop rbp
0x0f126870: jmp 0f07c6e0h  ;   {runtime_call}
0x0f126875: hlt
0x0f126876: hlt
0x0f126877: hlt
0x0f126878: hlt
0x0f126879: hlt
0x0f12687a: hlt
0x0f12687b: hlt
0x0f12687c: hlt
0x0f12687d: hlt
0x0f12687e: hlt
0x0f12687f: hlt
```

Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
 Code:
 stack=2, locals=1, args_size=1
   0: aload_0
   1: invokespecial Object.<init>.()V
   4: aload_0
   5: bipush        48
   7: putfield      MyClass.MyField.I
  10: return
 LineNumberTable:
   line 2: 0
   line 3: 4
   line 2: 10
 LocalVariableTable:
   0       11      0  this    LmyClass;
```

```
0x0f126760: mov r10d,dword ptr
0x0f126764: shl r10,3h
0x0f126768: cmp r10,rax
0x0f12676b: jne 0f055f60h  ;   {runtime_call}
0x0f126771: nop word ptr
0x0f12677c: nop
[Verified Entry Point]
0x0f126780: mov dword ptr
0x0f126787: push rbp
0x0f126788: sub rsp,30h
0x0f12678c: mov rsi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f126796: mov edi,dword ptr
0x0f12679c: add edi,8h
0x0f12679f: mov dword ptr
0x0f1267a5: mov rsi,0ce506f0h  ;   {metadata({method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267af: and edi,1ff8h
0x0f1267b5: cmp edi,0h
0x0f1267b8: je L0002  ;*aload_0; - test.MyClass::<init>@0 (line 2)
      L0000: mov rsi,rdx
0x0f1267c1: mov rdi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267cb: add qword ptr
0x0f1267d3: mov rsi,0cbaf870h  ;   {metadata(method data for {method} {0x000000000ca50488} '<init>' '()V' in
'java/lang/Object')}
0x0f1267dd: mov edi,dword ptr
0x0f1267e3: add edi,8h
0x0f1267e6: mov dword ptr
0x0f1267ec: mov rsi,0ca50480h  ;   {metadata({method} {0x000000000ca50488} '<init>' '()V' in 'java/lang/Object')}
0x0f1267f6: and edi,7ffff8h
0x0f1267fc: cmp edi,0h
0x0f1267ff: je L0003
      0001: mov dword ptr  ; - test.MyClass::<init>@7 (line 3)
0x0f12680c: add rsp,30h
0x0f126810: pop rbp
0x0f126811: test dword ptr
0x0f126817: ret
      L0002: mov qword ptr
0x0f12681d: mov qword ptr
0x0f126825: call 0f10e3a0h  ; OopMap{rdx=Oop off=202}; *synchronization entry;- test.MyClass::<init>@-1 (line 2)
0x0f12682a: jmp L0000
      L0003: mov qword ptr
0x0f126831: mov qword ptr
0x0f126839: call 0f10e3a0h  ; OopMap{rdx=Oop off=222};*synch entry; - java.lang.Object::<init>@-1 (line 37); -
test.MyClass::<init>@1 (line 2)
0x0f12683e: jmp L0001
0x0f126840: nop
0x0f126841: nop
0x0f126842: mov rax,qword ptr
0x0f126849: mov r10,0h
0x0f126853: mov qword ptr
0x0f12685a: mov r10,0h
0x0f126864: mov qword ptr
0x0f12686b: add rsp,30h
0x0f12686f: pop rbp
0x0f126870: jmp 0f07c6e0h  ;   {runtime_call}
0x0f126875: hlt
0x0f126876: hlt
0x0f126877: hlt
0x0f126878: hlt
0x0f126879: hlt
0x0f12687a: hlt
0x0f12687b: hlt
0x0f12687c: hlt
0x0f12687d: hlt
0x0f12687e: hlt
0x0f12687f: hlt
```

Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
 Code:
 stack=2, locals=1, args_size=1
   0: aload_0
   1: invokespecial Object.<init>.()V
   4: aload_0
   5: bipush        48
   7: putfield      MyClass.MyField.I
  10: return
 LineNumberTable:
   line 2: 0
   line 3: 4
   line 2: 10
 LocalVariableTable:
   0       11      0  this    LmyClass;
```

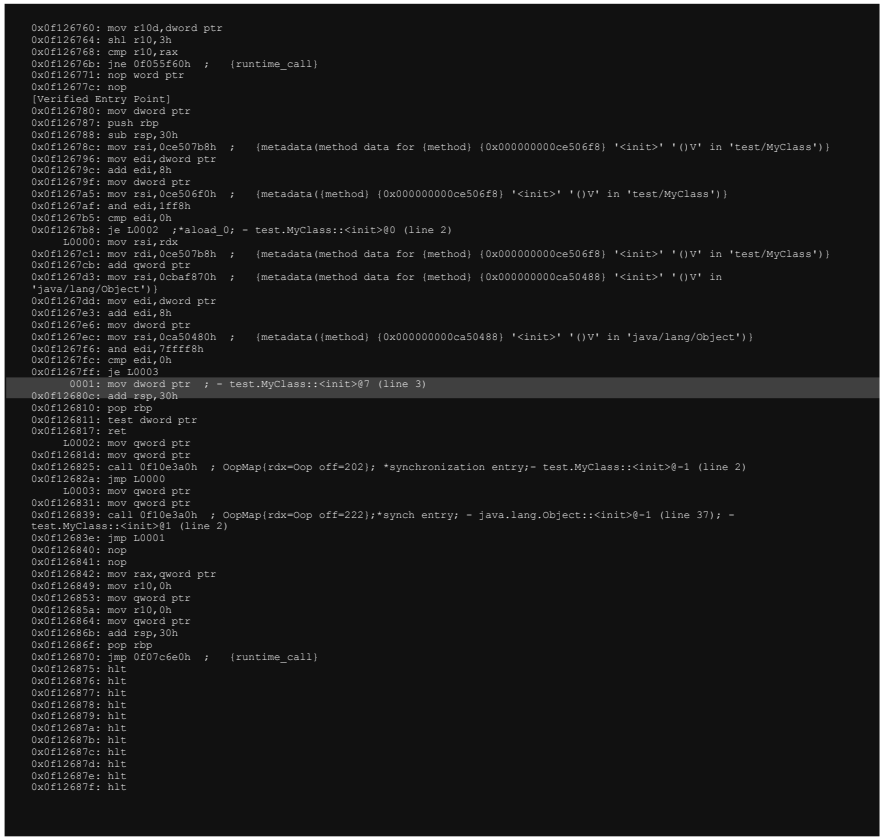
```
0x0f126760: mov r10d,dword ptr
0x0f126764: shl r10,3h
0x0f126768: cmp r10,rax
0x0f12676b: jne 0f055f60h  ;   {runtime_call}
0x0f126771: nop word ptr
0x0f12677c: nop
[Verified Entry Point]
0x0f126780: mov dword ptr
0x0f126787: push rbp
0x0f126788: sub rsp,30h
0x0f12678c: mov rsi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f126796: mov edi,dword ptr
0x0f12679c: add edi,8h
0x0f12679f: mov dword ptr
0x0f1267a5: mov rsi,0ce506f0h  ;   {metadata({method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267af: and edi,1ff8h
0x0f1267b5: cmp edi,0h
0x0f1267b8: je L0002  ;*aload_0; - test.MyClass::<init>@0 (line 2)
      L0000: mov rsi,rdx
0x0f1267c1: mov rdi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267cb: add qword ptr
0x0f1267d3: mov rsi,0cbaf870h  ;   {metadata(method data for {method} {0x000000000ca50488} '<init>' '()V' in
'java/lang/Object')}
0x0f1267dd: mov edi,dword ptr
0x0f1267e3: add edi,8h
0x0f1267e6: mov dword ptr
0x0f1267ec: mov rsi,0ca50480h  ;   {metadata({method} {0x000000000ca50488} '<init>' '()V' in 'java/lang/Object')}
0x0f1267f6: and edi,7ffff8h
0x0f1267fc: cmp edi,0h
0x0f1267ff: je L0003
      0001: mov dword ptr  ; - test.MyClass::<init>@7 (line 3)
0x0f12680c: add rsp,30h
0x0f126810: pop rbp
0x0f126811: test dword ptr
0x0f126817: ret
      L0002: mov qword ptr
0x0f12681d: mov qword ptr
0x0f126825: call 0f10e3a0h  ; OopMap{rdx=Oop off=202}; *synchronization entry;- test.MyClass::<init>@-1 (line 2)
0x0f12682a: jmp L0000
      L0003: mov qword ptr
0x0f126831: mov qword ptr
0x0f126839: call 0f10e3a0h  ; OopMap{rdx=Oop off=222};*synch entry; - java.lang.Object::<init>@-1 (line 37); -
test.MyClass::<init>@1 (line 2)
0x0f12683e: jmp L0001
0x0f126840: nop
0x0f126841: nop
0x0f126842: mov rax,qword ptr
0x0f126849: mov r10,0h
0x0f126853: mov qword ptr
0x0f12685a: mov r10,0h
0x0f126864: mov qword ptr
0x0f12686b: add rsp,30h
0x0f12686f: pop rbp
0x0f126870: jmp 0f07c6e0h  ;   {runtime_call}
0x0f126875: hlt
0x0f126876: hlt
0x0f126877: hlt
0x0f126878: hlt
0x0f126879: hlt
0x0f12687a: hlt
0x0f12687b: hlt
0x0f12687c: hlt
0x0f12687d: hlt
0x0f12687e: hlt
0x0f12687f: hlt
```

Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```



Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```

```
0x0f126760: mov r10d,dword ptr
0x0f126764: shl r10,3h
0x0f126768: cmp r10,rax
0x0f12676b: jne 0f055f60h  ;   {runtime_call}
0x0f126771: nop word ptr
0x0f12677c: nop
[Verified Entry Point]
0x0f126780: mov dword ptr
0x0f126787: push rbp
0x0f126788: sub rsp,30h
0x0f12678c: mov rsi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f126796: mov edi,dword ptr
0x0f12679c: add edi,8h
0x0f12679f: mov dword ptr
0x0f1267a5: mov rsi,0ce506f0h  ;   {metadata({method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267af: and edi,1ff8h
0x0f1267b5: cmp edi,0h
0x0f1267b8: je L0002  ;*aload_0; - test.MyClass::<init>@0 (line 2)
      L0000: mov rsi,rdx
0x0f1267c1: mov rdi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267cb: add qword ptr
0x0f1267d3: mov rsi,0cbaf870h  ;   {metadata(method data for {method} {0x000000000ca50488} '<init>' '()V' in
'java/lang/Object')}
0x0f1267dd: mov edi,dword ptr
0x0f1267e3: add edi,8h
0x0f1267e6: mov dword ptr
0x0f1267ec: mov rsi,0ca50480h  ;   {metadata({method} {0x000000000ca50488} '<init>' '()V' in 'java/lang/Object')}
0x0f1267f6: and edi,7ffff8h
0x0f1267fc: cmp edi,0h
0x0f1267ff: je L0003
      0001: mov dword ptr  ; - test.MyClass::<init>@7 (line 3)
0x0f12680c: add rsp,30h
0x0f126810: pop rbp
0x0f126811: test dword ptr
0x0f126817: ret
      L0002: mov qword ptr
0x0f12681d: mov qword ptr
0x0f126825: call 0f10e3a0h  ; OopMap{rdx=Oop off=202}; *synchronization entry;- test.MyClass::<init>@-1 (line 2)
0x0f12682a: jmp L0000
      L0003: mov qword ptr
0x0f126831: mov qword ptr
0x0f126839: call 0f10e3a0h  ; OopMap{rdx=Oop off=222};*synch entry; - java.lang.Object::<init>@-1 (line 37); -
test.MyClass::<init>@1 (line 2)
0x0f12683e: jmp L0001
0x0f126840: nop
0x0f126841: nop
0x0f126842: mov rax,qword ptr
0x0f126849: mov r10,0h
0x0f126853: mov qword ptr
0x0f12685a: mov r10,0h
0x0f126864: mov qword ptr
0x0f12686b: add rsp,30h
0x0f12686f: pop rbp
0x0f126870: jmp 0f07c6e0h  ;   {runtime_call}
0x0f126875: hlt
0x0f126876: hlt
0x0f126877: hlt
0x0f126878: hlt
0x0f126879: hlt
0x0f12687a: hlt
0x0f12687b: hlt
0x0f12687c: hlt
0x0f12687d: hlt
0x0f12687e: hlt
0x0f12687f: hlt
```

Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
  Code:
  stack=2, locals=1, args_size=1
    0: aload_0
    1: invokespecial Object.<init>.()V
    4: aload_0
    5: bipush        48
    7: putfield      MyClass.MyField.I
   10: return
  LineNumberTable:
    line 2: 0
    line 3: 4
    line 2: 10
  LocalVariableTable:
    0       11      0  this    LmyClass;
```

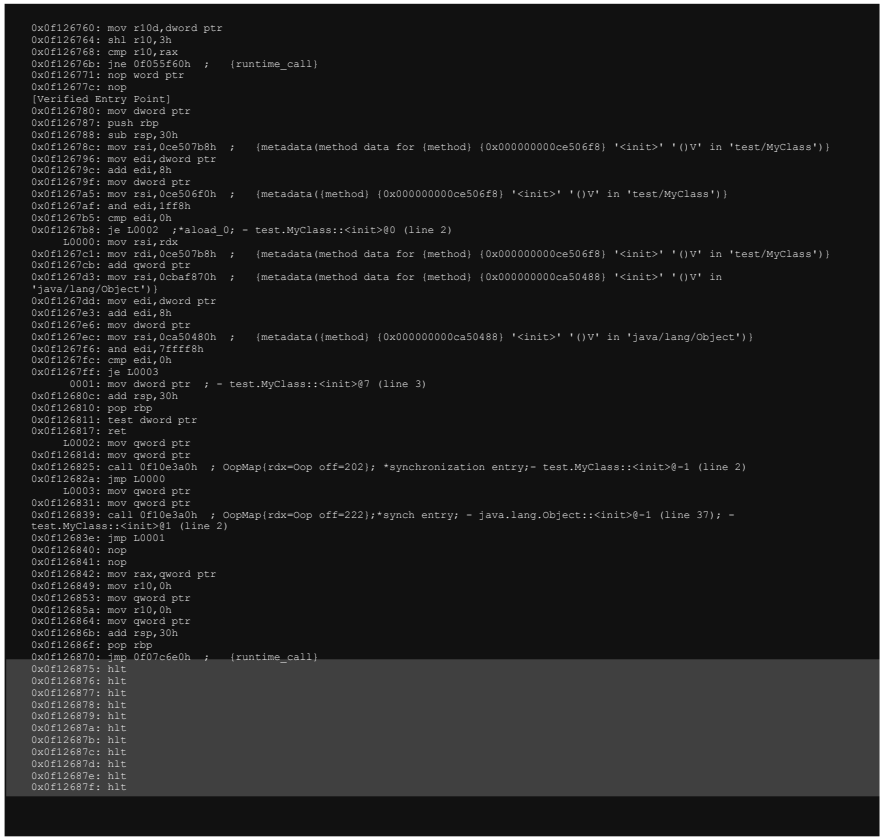
```
0x0f126760: mov r10d,dword ptr
0x0f126764: shl r10,3h
0x0f126768: cmp r10,rax
0x0f12676b: jne 0f055f60h  ;   {runtime_call}
0x0f126771: nop word ptr
0x0f12677c: nop
[Verified Entry Point]
0x0f126780: mov dword ptr
0x0f126787: push rbp
0x0f126788: sub rsp,30h
0x0f12678c: mov rsi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f126796: mov edi,dword ptr
0x0f12679c: add edi,8h
0x0f12679f: mov dword ptr
0x0f1267a5: mov rsi,0ce506f0h  ;   {metadata({method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267af: and edi,1ff8h
0x0f1267b5: cmp edi,0h
0x0f1267b8: je L0002  ;*aload_0; - test.MyClass::<init>@0 (line 2)
      L0000: mov rsi,rdx
0x0f1267c1: mov rdi,0ce507b8h  ;   {metadata(method data for {method} {0x000000000ce506f8} '<init>' '()V' in 'test/MyClass')}
0x0f1267cb: add qword ptr
0x0f1267d3: mov rsi,0cbaf870h  ;   {metadata(method data for {method} {0x000000000ca50488} '<init>' '()V' in
'java/lang/Object')}
0x0f1267dd: mov edi,dword ptr
0x0f1267e3: add edi,8h
0x0f1267e6: mov dword ptr
0x0f1267ec: mov rsi,0ca50480h  ;   {metadata({method} {0x000000000ca50488} '<init>' '()V' in 'java/lang/Object')}
0x0f1267f6: and edi,7ffff8h
0x0f1267fc: cmp edi,0h
0x0f1267ff: je L0003
      0001: mov dword ptr  ; - test.MyClass::<init>@7 (line 3)
0x0f12680c: add rsp,30h
0x0f126810: pop rbp
0x0f126811: test dword ptr
0x0f126817: ret
      L0002: mov qword ptr
0x0f12681d: mov qword ptr
0x0f126825: call 0f10e3a0h  ; OopMap{rdx=Oop off=202}; *synchronization entry;- test.MyClass::<init>@-1 (line 2)
0x0f12682a: jmp L0000
      L0003: mov qword ptr
0x0f126831: mov qword ptr
0x0f126839: call 0f10e3a0h  ; OopMap{rdx=Oop off=222};*synch entry; - java.lang.Object::<init>@-1 (line 37); -
test.MyClass::<init>@1 (line 2)
0x0f12683e: jmp L0001
0x0f126840: nop
0x0f126841: nop
0x0f126842: mov rax,qword ptr
0x0f126849: mov r10,0h
0x0f126853: mov qword ptr
0x0f12685a: mov r10,0h
0x0f126864: mov qword ptr
0x0f12686b: add rsp,30h
0x0f12686f: pop rbp
0x0f126870: jmp 0f07c6e0h  ;   {runtime_call}
0x0f126875: hlt
0x0f126876: hlt
0x0f126877: hlt
0x0f126878: hlt
0x0f126879: hlt
0x0f12687a: hlt
0x0f12687b: hlt
0x0f12687c: hlt
0x0f12687d: hlt
0x0f12687e: hlt
0x0f12687f: hlt
```

Executing Java Code

```
public class MyClass {String[] args) {
    public int myField = 48;
}
```

```
public int myField;
  flags: ACC_PUBLIC

public MyClass();
  flags: ACC_PUBLIC
 Code:
 stack=2, locals=1, args_size=1
   0: aload_0
   1: invokespecial Object.<init>.()V
   4: aload_0
   5: bipush        48
   7: putfield      MyClass.MyField.I
  10: return
 LineNumberTable:
   line 2: 0
   line 3: 4
   line 2: 10
 LocalVariableTable:
   0       11      0  this    LmyClass;
```



Just In Time Compilation

Bytecode optimization

```
public int[] intArr = {0, 1, 2};
```

Approach #1:

```
public synchronized int sync1(){
    return intArr[0];
}
```

5 bytecodes

```
0: aload_0
1: getfield      #12
4: iconst_0
5: iaload
6: ireturn
```

Approach #2:

```
public int sync2(){
   synchronized (this){
       return intArr[0];
   }
}
```

14 bytecodes

```
 0: aload_0
 1: dup
 2: astore_1
 3: monitorenter
 4: aload_0
 5: getfield      #12
 8: iconst_0
 9: iaload
10: aload_1
11: monitorexit
12: ireturn
13: aload_1
14: monitorexit
15: athrow
```

Which is faster?

- A: approach #1
- B: approach #2

Bytecode optimization

```
public int[] intArr = {0, 1, 2};
```

Approach #1:

```
public synchronized int sync1(){
    return intArr[0];
}
```

5 bytecodes

```
0: aload_0
1: getfield      #12
4: iconst_0
5: iaload
6: ireturn
```

Approach #2:

```
public int sync2(){
   synchronized (this){
       return intArr[0];
   }
}
```

14 bytecodes

```
 0: aload_0
 1: dup
 2: astore_1
 3: monitorenter
 4: aload_0
 5: getfield      #12
 8: iconst_0
 9: iaload
10: aload_1
11: monitorexit
12: ireturn
13: aload_1
14: monitorexit
15: athrow
```

Which is faster?

- A: approach #1
- B: approach #2

Answer: C – they're both the same

Inlining

Inlining

```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial = multiply(factorial, i);
    }
    return factorial;
}


public static int multiply(int a, int b){
    return a * b;
}
```

Inlining

```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial = multiply(factorial, i);
    }
    return factorial;
}
```

67 instructions

```
public static int multiply(int a, int b){
    return a * b;
}
```

Inlining

```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial = multiply(factorial, i);
    }
    return factorial;
}
```

67 instructions

```
public static int multiply(int a, int b){
    return a * b;
}
```

9 instructions

Inlining

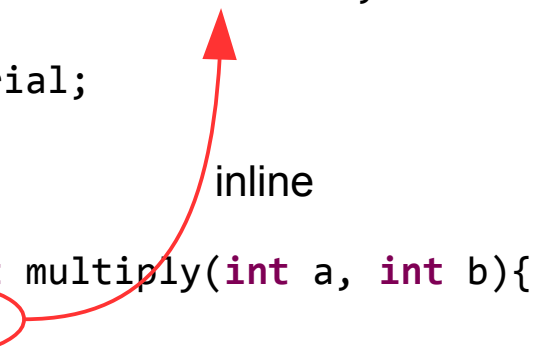
```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial =
    }
    return factorial;
}


public static int multiply(int a, int b){
    return a * b;
}
```

Inlining

```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial = factorial * i;
    }
    return factorial;
}

public static int multiply(int a, int b){
    return a * b;
}
```



Inlining

```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial = factorial * i;
    }
    return factorial;
}
```

71 instructions

```
public static int multiply(int a, int b){
    return a * b;
}
```

Inlining

```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial = factorial * i;
    }
    return factorial;
}
```

71 instructions

```
public static int multiply(int a, int b){
    return a * b;
}
```

9 instructions

Inlining

```
public static int factorial(int num) {
    int factorial = 1;
    for (int i = 1; i <= num; i++) {
        factorial = factorial * i;
    }
    return factorial;
}
```

71 instructions

```
public static int multiply(int a, int b){
    return a * b;
}
```

9 instructions

5 instructions saved per iteration

Loop Unrolling

```
private static final String[] numbers = {"0", "1", "2", "3", "4"};

public static int total() {
    int result = 0;

    for (String num : numbers) {
        result += parse(num);
    }
    return result;
}
```

28 instructions
14 on loop

```
private static int parse(String num) {
    return Integer.parseInt(num);
}
```

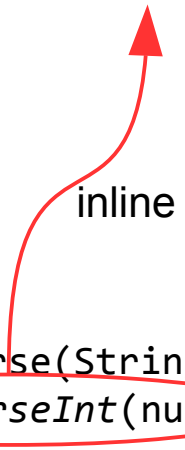
13 instructions

Loop Unrolling

```
private static final String[] numbers = {"0", "1", "2", "3", "4"};

public static int total() {
    int result = 0;

    for (String num : numbers) {
        result += Integer.parseInt(num);
    }
    return result;
}
```




```
private static int parse(String num) {
    return Integer.parseInt(num);
}
```

- Inlining removes the cost of calling *parse*

Loop Unrolling

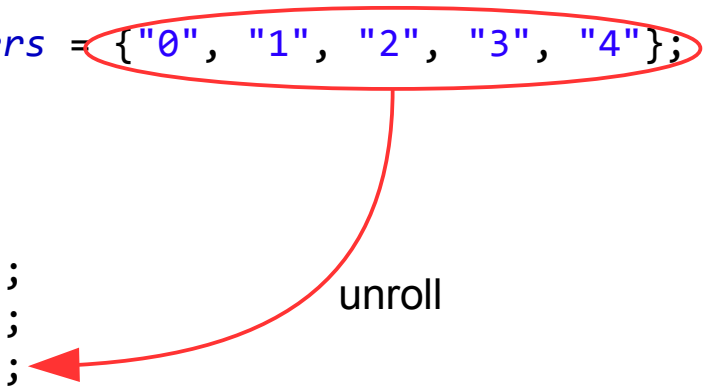
```
private static final String[] numbers = {"0", "1", "2", "3", "4"};

public static int total() {
    int result = 0;

    result += Integer.parseInt("0");
    result += Integer.parseInt("1");
    result += Integer.parseInt("2");
    result += Integer.parseInt("3");
    result += Integer.parseInt("4");

    return result;
}

private static int parse(String num) {
    return Integer.parseInt(num);
}
```



- Inlining removes the cost of calling *parse*
- Unrolling removes the cost of loop iteration

Loop Unrolling

```
private static final String[] numbers = {"0", "1", "2", "3", "4"};

public static int total() {
    int result = 0;

    result += Integer.parseInt("0");
    result += Integer.parseInt("1");
    result += Integer.parseInt("2");
    result += Integer.parseInt("3");
    result += Integer.parseInt("4");

    return result;
}

private static int parse(String num) {
    return Integer.parseInt(num);
}
```

Unrolling possible because collection is **final**

- Inlining removes the cost of calling *parse*
- Unrolling removes the cost of loop iteration

Synchronization

Synchronization approaches

```
public int[] intArr = {0, 1, 2};
```

```
public synchronized int sync1(){
    return intArr[0];
}
```

```
public int sync2(){
   synchronized (this){
       return intArr[0];
   }
}
```

- Unnecessary bytecodes removed

Lock Coarsening

```
public int[] intArr = {0, 1, 2};

public int arrayCount() {
    int result = 0;

    for (int i = 0; i < intArr.length; i++) {
        result += getEntry(i);
    }

    return result;
}

public synchronized int getEntry(int entry){
    return intArr[entry];
}
```

Lock Coarsening

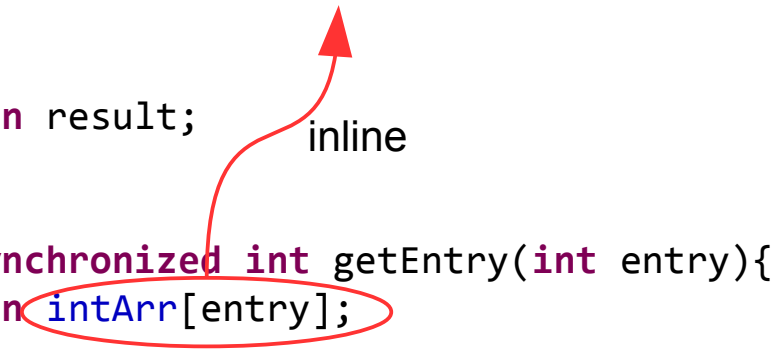
```
public int[] intArr = {0, 1, 2};

public int arrayCount() {
    int result = 0;

    for (int i = 0; i < intArr.length; i++) {
        result += intArr[entry];
    }

    return result;
}

public synchronized int getEntry(int entry){
    return intArr[entry];
}
```

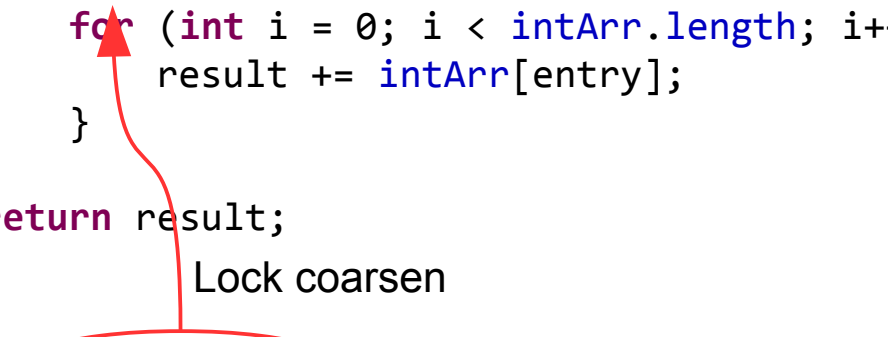


Lock Coarsening

```
public int[] intArr = {0, 1, 2};

public int arrayCount() {
    int result = 0;
    synchronized {
        for (int i = 0; i < intArr.length; i++) {
            result += intArr[entry];
        }
    }
    return result;
}

public synchronized int getEntry(int entry){
    return intArr[entry];
}
```

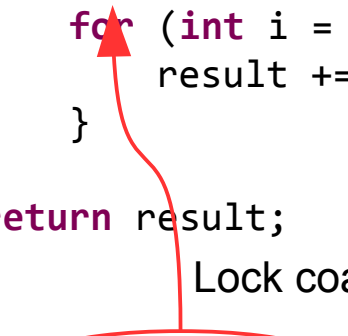


Lock Coarsening

```
public int[] intArr = {0, 1, 2};

public int arrayCount() {
    int result = 0;
    synchronized {
        for (int i = 0; i < intArr.length; i++) {
            result += intArr[entry];
        }
    }
    return result;
}

public synchronized int getEntry(int entry){
    return intArr[entry];
}
```



- Repeated locking removed

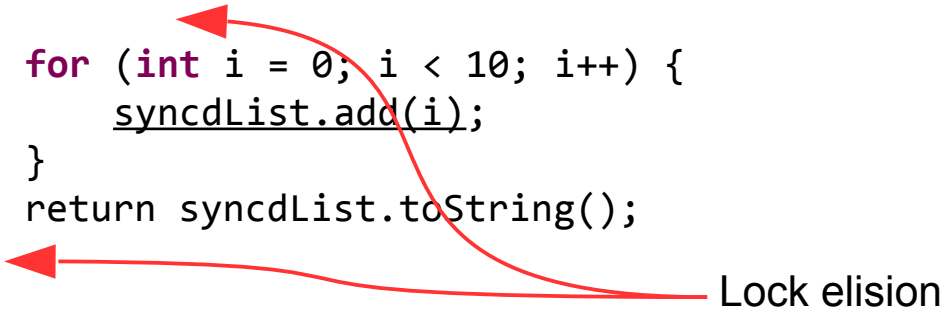
Lock Elision

```
public String walkLockedList() {
    List syncdList = new ArrayList();
    synchronized (syncdList) {
        for (int i = 0; i < 10; i++) {
            syncdList.add(i);
        }
        return syncdList.toString();
    }
}
```

Lock Elision

```
public String walkLockedList() {
    List syncdList = new ArrayList();

        for (int i = 0; i < 10; i++) {
            syncdList.add(i);
        }
        return syncdList.toString();
}
```



- Lock removed as un-necessary

Field Access

Accessing Fields: Static Fields

```
public static int test = 10;
public static int getTest() {
    return test;
}
```

30 instructions

```
private static int test = 10;
public static int getTest() {
    return test;
}
```

30 instructions

```
public final static int test = 10;
public static int getTest() {
    return test;
}
```

17 instructions

Accessing Fields: Instance Fields

```
public int test = 10;
public int getTest() {
    return test;
}
```

23 instructions

```
private int test = 10;
public int getTest() {
    return test;
}
```

23 instructions

```
public final int test = 10;
public int getTest() {
    return test;
}
```

17 instructions

Enabling Optimizations

Certainty

- Locals are faster than globals
  - Fields and statics slow; parameters and locals fast
- Constants are faster than variables
  - `final` is your friend
- `private` can be faster than `public`
  - `protected` and `package` are always just as slow as `public`
- Small methods (≤100 bytecodes) are good
- Simple is faster than complex

Analyzing the JIT

Digging into the generated assembler

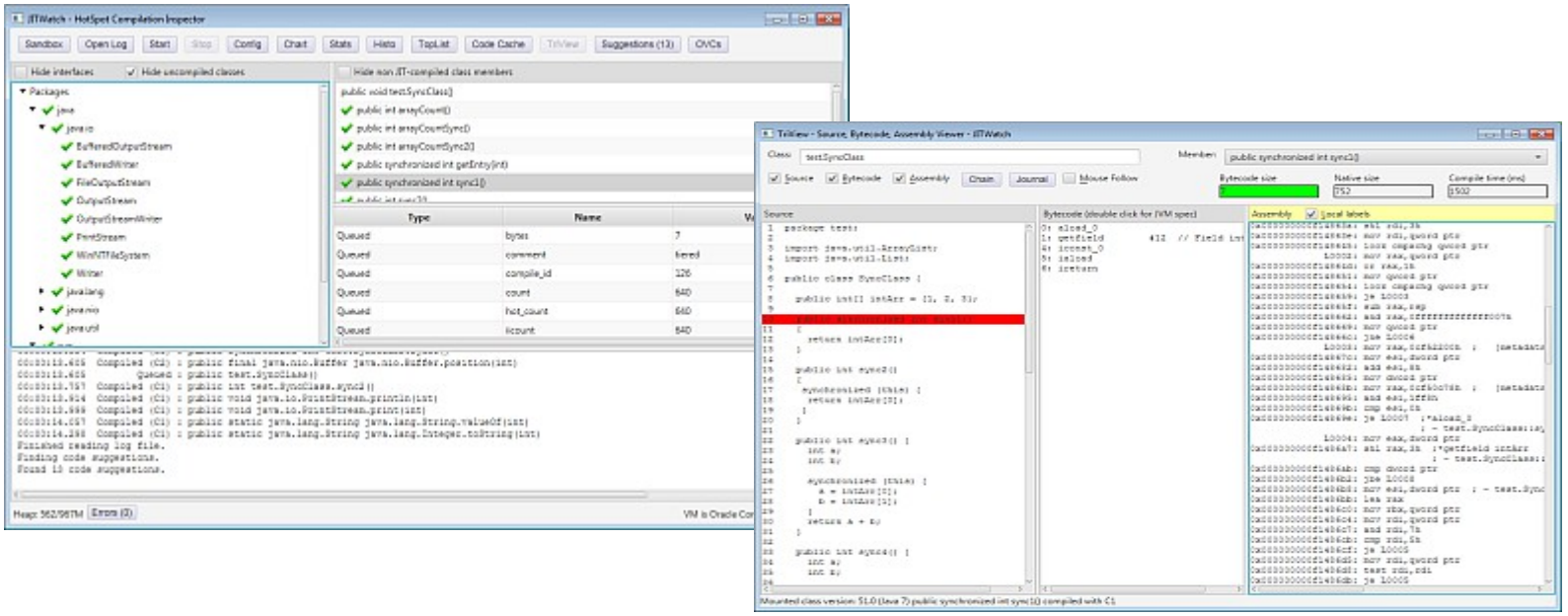
- HotSpot JIT provides options to look at JIT behaviour:
  - `-XX:+UnlockDiagnosticVMOptions`
  - `-XX:+TraceClassLoading`
  - `-XX:+LogCompilation`
  - `-XX:+PrintAssembly`

- -XX:+PrintAssembly requires additional `hdis` library
  - Installed into `jre/bin/server`

- Generates output to console and `hotspot_pid<pid>.log` file

Visualizing with JITWatch

- Open source JITWatch tool provides visualization of data:



- Also attempts to make optimization suggestions

Questions?

Copyright and Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., and registered in many jurisdictions worldwide.

Other product and service names might be trademarks of IBM or other companies.

A current list of IBM trademarks is available on the Web – see the IBM “Copyright and trademark information” page at URL: www.ibm.com/legal/copytrade.shtml