INVOKEVERVIRUAL STEP 1

activity:

 presently executing method "foo". It wants to call method "bar".

LV (local variable) ptr  

SP  

foo's stack

IV

activity:
• method foo pushes OBJREF, which is a ptr to object from which it will call the bar method.

BIPUSH <OBJREF>

INVOKEVIRTUAL STEP 2

code:
method foo pushes three arguments to bar.

activity:

code:

INVOKESPECIAL

<table>
<thead>
<tr>
<th>Obj stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>arg 3 to bar</td>
</tr>
<tr>
<td>arg 2 to bar</td>
</tr>
<tr>
<td>arg 1 to bar</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LV (local variable) ptr</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIPUSH ARG1</td>
</tr>
<tr>
<td>BIPUSH ARG2</td>
</tr>
<tr>
<td>BIPUSH ARG3</td>
</tr>
<tr>
<td>INVOKEVIRTUAL STEP 3</td>
</tr>
</tbody>
</table>
INVOKEVIRTUAL

activity:
- INVOKEVIRTUAL called with offset into constant pool
- this offset dereferenced into ptr to "bar" method itself
- look into bar to determine how many args it accepts
- subtract this from SP to get ptr to base of arg stack for "bar"

arg 1 to bar
arg 2 to bar
arg 3 to bar

constant pool

method area

ptr to "bar" in method area

invokes LV

foo's stack

INVOKEMETHOD

INVOKEVIRTUAL 71

ptr to "bar" in method area

loc 71

Method

#args  #locals

foo's LV

foo's stack

OBJREF

arg 1 to bar
arg 2 to bar
arg 3 to bar

INVOKEMETHOD STEP 4
activity:

• have ptr to "bar"

look into bar to determine how many args it accepts

- subject this from SP to get ptr to base of arg stack for "bar"

INVOKEVIRTUAL 71

code:

INVOKEVIRTUAL bar

ptr to "bar" in method area

#locals #args

ptr to "bar"

SP – #args

INVOKEVIRTUAL STEP 5

INVOKEVIRTUAL STEP 5

foo's stack

foo's LV

SP

OBJREF

constant pool

method area

ptr to "bar" in method area

constent pool
INVOKEVIRTUAL Step 6

InvokEvirtual 71

Activity:
- Save foo's PC where link ptr is pointing
- Where OBJREF lived, install link ptr
- Have ptr (old SP - #args)
- Add bar's #args + #locals to reserve space for bar
- Have ptr (old SP - #args)

Code:

- INVOKEVIRTUAL 71
  - ptr to "bar" in method area
  - #args           #locals

Method area

Constant pool

foo's stack

foo's LV

new SP

link ptr

old SP - #args

foo's PC

ptr to "bar" in method area

arg 1 to bar

arg 2 to bar

arg 3 to bar

bars locals

space for bar's locals
INVOKENVIRONMENTAL STEP 7

Activity:
- Begin executing bar's code
- Set new LV (for bar's context) at link ptr location
- Push foo's LV on stack

Code:
`INVOKEVIRTUAL 71 ptr to "bar" in method area``

Method area

Constant pool

Foo's stack

Foo's LV

Foo's PC

SP

Bar's scratch stack

Foo's LV

Method area

`ptr to "bar" in method area`

`ptr to "bar"`

#args

#locals

LV for bar

Link ptr

Arg 1 to bar

Arg 2 to bar

Arg 3 to bar

Space for bar's locals

Arg 1 to bar

Arg 2 to bar

Arg 3 to bar

LV for bar
RETURN

Activity:
- bar executes IRETURN
- bar has cleared its scratch stack and pushed its return value

I JVM code:
IRETURN

RETURN STEP 1

foo's stack
SP
foo's LV
LV for bar
foo's PC

bar's locals

space for bar's locals

link ptr
arg 3 to bar
arg 2 to bar
arg 1 to bar
arg

foo's stack

arg 1 to bar
arg 2 to bar
arg 3 to bar

return value
**RETURN COMPLETE**

From foo into bar:
- Write return value onto new TOS, execute opcode
- Restore foo’s LV from 2 steps back
- From i to step SP points at loc of link ptr
- From foo’s PC from rd 2 steps back, read in foo’s LV, fetch
- Inc MAR to point at foo’s LV
- Temporarily set LV to get at foo’s PC
- Bar’s return value is already in TOS register
- Dereference bar’s LV to get link ptr, that’s new SP

**Activity:**
- Dereference bar’s LV to get bar’s link ptr
- Dereference bar’s link ptr
- Dereference bar’s link ptr
- Dereference bar’s LV

**Microcode code:**
1. MAR = SP = LV; read
2. LV = MAR = MDR; rd
3. MAR = LV + 1
4. PC = MDR; rd; fetch
5. MAR = SP
6. LV = MDR
7. MDR = TOS; wr; goto Main1

**Space for bar’s locals**
- When done

**Stacks**
- foo’s stack
- bar’s stack
- foo’s return value
- bar’s return value

**Process:**
- arg 1 to bar
- arg 2 to bar
- arg 3 to bar

**Variables:**
- foo’s PC
- foo’s LV
- bar’s return value
- TOS