Where did all the time go?

First make it right, then make it fast.
How long does it take?

• several methods on blocks let you find out how long the block takes to execute

  ▶ [data detectMax: [ :each | each ] ] timeToRun ➞ 7

  ▶ [data detectMax: [ :each | each ] ] durationToRun ➞ 0:00:00:00.007

  ▶ [data detectMax: [ :each | each ] ] bench ➞ '153.78 per second.'
Why did it take so long?

- TimeProfileBrowser is your friend
  - probabilistic sampling of pc (roughly once per ms)
- Really easy to use:
  - TimeProfileBrowser onBlock: [ <your code here> ]
- Let's use it on some examples:
  - TimeProfileBrowser onBlock:
    [String streamContents:
      [ :str | data do: [ :each | str print: each; space ]]]
20 tallies: barely enough samples to be valid
• Let’s try:
  
  ▸ TimeProfileBrowser onBlock:
    ```plaintext
    [String streamContents:
      [ :str l data do: [ : each l str print: each; space ]]]
    bench
    ```
  
  ▸ `<code>` bench does 5 seconds’-worth of computation (or 1 iteration)
**Tree**
87.1% {4364ms} WriteStream (Stream)>>print:
| 84.6% {4238ms} SmallInteger (Number)>>printOn:
| 82.9% {4153ms} SmallInteger (Integer)>>printOn:base:
| 78.7% {3943ms} SmallInteger (Integer)>>printStringBase:
| 17.0% {852ms} WriteStream>>nextPut:
| | 8.8% {441ms} WriteStream>>pastEndPut:
| | | 2.9% {145ms} ByteString class (String class)>>new:
| | | 2.1% {105ms} primitives
| | | 4.5% {225ms} Character>>isOctetCharacter
| | | 3.7% {185ms} primitives
| | | 16.0% {802ms} WriteStream class (PositionableStream class)>>on:
| | | 11.9% {596ms} WriteStream>>on:

md 2/24/2006 19:50 • private • 5 implementors • in no change set •

```
| pastEndPut: anObject

"Grow the collection by creating a new bigger collection and then
  copy over the contents from the old one. We grow by doubling the size
  but the growth is kept between 20 and 1000000.
  Finally we put <anObject> at the current write position."

| oldSize grownCollection |
oldSize := collection size.
collection := grownCollection replaceFrom: 1 to: oldSize with: collection startingAt: 1.
writeLimit := collection size.
collection at: (position := position + 1) put: anObject.
"return the argument - added by kwl"
↑ anObject
```