Ibland kommer frågan upp vad "`length`" i en array är för något. Det är ett tal som innehåller längden på en array, det vet vi. Men var kommer den från och vad är det?

Så här är det:

"The array's length is available as a final instance variable length." [0]

Man kan också sluta sig till att det är så genom att konstatera att det inte är några parenteser i uttryck så som:

```
 i = args.length;
```

Ett bra sätt att undersöka hur Java funkar är att fråga kompilatorn ;-)
Vi kan observera vad vår vän javac säger om man försöker kompilera ett uttryck så som:

```
i = arr.length();
"Arr.java:3: error: cannot find symbol
        int i = args.length();
                     ^
  symbol:   method length()
  location: variable args of type String[]"
```

javac klagar på "`symbol:   method length()`" det finns ju ingen sådan **metod**.

eller

```
i = args.lengthhhh;

"Arr.java:4: error: cannot find symbol
        int i = args.lengthhhhh;
                     ^
  symbol:   variable lengthhhhh
  location: variable args of type String[]"
```

javac klagar på "`symbol:   variable lengthhhhh`" det finns ju ingen sådan **variabel**.

Vad händer om vi försöker ändra length?

```
args.length=19;
```

javac blir gnällig igen:

```
"Arr.java:5: error: cannot assign a value to final variable length
        args.length=19;"
```

(final betyder att en variabel är "konstant" och inte får ändras när den väl fått ett värde)

Sammantaget får vi här ledtrådar till att length är en final variable. Men vilken typ?

Vi testar:

```
byte b = args.length;
```

javac (som vid det här laget börjar ge upp hoppet om mig) gnäller vidare:

```
"Arr.java:6: error: possible loss of precision
        byte b = args.length;
                      ^
  required: byte
  found:    int"
```

"found: int" betyder att args.length har typen int.

Nu vet vi att det som står i språkspecifikationen måste stämma (för åtminstone min implementation
av Java). Nu kan vi slå fast att length är deklarerad som en instansvariabel på följande vis:

public final int length;

(och den lär få sitt värde i konstruktorn, som anropas av antingen new `<typ>[<size>]` eller av initializersyntaxen `{<elem0>,<elem1>...<elemSize-1>};`)

En array deklareras ju så här:
```
  int[] intArray = new int[5];
```
Alternativt om man vill ge initiala värden direkt såhär:
```
  int[] intArray = {3,5,7,9,10};
```

I bägge fallen sätts length till 5 för array-objektet som refereras till av referensvariabeln intArray.

Namngivningskonventioner, då?

Konstanter, kan man läsa sig till, brukar namnges med VERSALER. Men sådana konstanter brukar vara publika klass-konstanter (som t ex Math.PI vilket är en konstant för pi (3.141592...)).

Men length kan ju knappast vara en klass-konstant (alla arrayer har ju knappast samma längd), utan snarare en "instans-konstant" (varje array-objekt håller reda på sin egen längd och har sin egen length-variabel för detta).

Jag antar att vi kan utgå från att det är OK med lower case för namn på en instans-konstant, då man
brukar säga att static final-variabler ska ha `UPPER_CASE_NAMES` men faktiskt inte säger något om final instansvariabler.

Jag har inte sett någon konvention kring hur man ska namnge final instansvariabler när jag letat kring. Och length hos arrayer kommer ju från själva språket Java, så jag antar att det är OK då att köra length med bara små bokstäver.

Antar att det är för att static final brukar användas för konstanter att användas i uttryck (typ Math.PI) medan instansvariabler som är final brukar snarare handla om att de inte får ändras när de väl har satts (typ array.length) och inte om några "globala konstanter" som ska användas just i stället för ett tal som vi känner till (PI) men inte bör skriva in själva, eftersom det är tydligare i ett program om det står:

```
  area = Math.PI * radius * radius;
```

än om vi skriver in siffor:

```
  area = 3.141592 * 5 * 5; // om radien är t ex 5 för en cirkel
```

Poängerna med detta dokument är

- Man kan använda kompilatorn för att undersöka en språklig konstuktion i Java - man får faktiskt ledtrådar från javac
- Man kan läsa i Java Language Specification (om man är väldigt nyfiken och inte allergisk mot tekniska specifikationer)
- `final` är en modifierarare som för variabler betyder att de är konstanta, inte får ändras värde på
- Det finns namngivningskonventioner för det mesta men inte riktigt allt. Om SUN/Oracle använder en konvention i sjävla språket, så är väl den konventionen OK
- Bonus-diskussion: Det är tydligare att använda konstanter än att skriva in siffror i uttryck - `intArray.length` är ju tydligt vad det betyder, liksom `Math.PI`

//Rikard

0. JLS http://docs.oracle.com/javase/specs/jls/se8/html/jls-10.html#jls-10.3