Variabler och konstanter

Variabler

- ✓ Program kommer ihåg värden med hjälp av variabler.
- ✓ Genom att binda ett namn till ett värde (tilldela variabeln ett värde) kan du referera till värdet i kod du skriver.
- ✓ En variabel deklareras med hjälp av nyckelordet `let` (innan ES6 användes `var`) följt av namnet på variabeln.

```
let count                    // undefined
let result = (4 + 8) * 3 + 8  // 44
```

`let` infördes eftersom `var` har flera förvirrande beteenden — variabler läcker ut ur block, kan omdeklareras utan varning, och går att använda innan de deklareras. `let` (och `const`) håller sig istället strikt inom sitt block och ger tydliga fel när något går fel, vilket fångar buggar tidigare.

- ✓ Efter att en variabel deklareras kan den användas i uttryck.

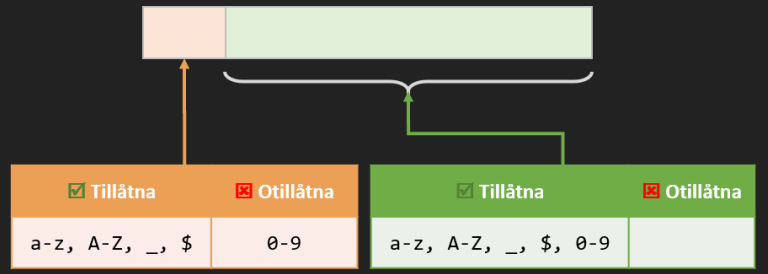
```
let result = (4 + 8) * 3 + 8       // 44
let theMeaningOfLife = result - 2  // 42
```

- ✓ En variabels värde kan ändras.

```
let result = (4 + 8) * 3 + 8  // 44
result =  13                  // 13
result = result + 29          // 42
```

Variablers namn

- ✓ Beskrivande namn är centralt. Håll dig till ett språk, gärna engelska.
- ✓ Kan vara vilket ord som helst bara det inte är ett nyckelord eller ett reserverat ord.
- ✓ Rekommenderade tecknen `a-z`, `A-Z`, `0-9`, `_` och `$`.
  - Får inte börja med en siffra.
  - Fler tecken är tillåtna, som t.ex. åäö, men använd aldrig dessa.



- ✓ Kan bestå av flera ord. Tillämpa i så fall så kallad kamelnotation (*"camelCasing"*).
  - Ett sätt att skriva samman ord utan bindestreck eller mellanslag.
  - Varje ord, utom det första, inleds med versal.

```
let bestBookEver = 'The Hitchhiker\'s Guide to the Galaxy'
let theMeaningOfLife = 42
```

Nyckelord och reserverade ord

- ✓ Problem vid definition av en variabel? Du kanske har använt något av orden nedan.

```
await       delete      import      this
break       do          in          throw
case        else        instanceof  typeof
catch       export      new         try
class       extends     let         var
const       finally     return      void
continue    for         static      while
debugger    function    super       with
default     if          switch      yield
false       null        true

enum        interface   private     public
implements  package     protected
```

Konstanter

- ✓ Konstanter fungerar precis som variabler med skillnaderna:
  - En konstant måste initieras till ett värde då den deklareras.
  - En konstants värde kan inte ändras.
- ✓ En konstant deklareras med hjälp av nyckelordet `const` följt av namnet på konstanten och dess värde.

```
const myFavoriteNumber = 42  // camelCase (vanligast)
const MY_FAVORITE_NUMBER = 42 // SCREAMING_SNAKE_CASE
```

Det finns olika uppfattningar om hur konstanters namn ska skrivas! Kursens tumregel: SCREAMING_SNAKE_CASE när värdet redan är känt när koden skrivs, annars camelCase.

- ✓ Det blir fel, kastas ett undantag, om du försöker ändra på en konstants värde.

```
const MY_FAVORITE_NUMBER = 42
MY_FAVORITE_NUMBER = 21
// MY_FAVORITE_NUMBER = 21
//                    ^
//
// TypeError: Assignment to constant variable.
```

Variabler och konstanters räckvidd (*”scope”*)

- ✓ En variabels räckvidd *(”scope”)* är det område där variabeln kan användas.
- ✓ Variablers, och konstanters, räckvidd är *”block scope”*, vilket innebär att de bara kan användas inom det område, block, där de har deklarerats.
  - Ett block med kod är, lite förenklat, kod som står inom ett par klammerparenteser.

```
{
    let bestBookEver = 'The Hitchhiker\'s Guide to the Galaxy'
    console.log(bestBookEver)
}
console.log(bestBookEver) // Utanför blocket kan inte variabeln användas och ett fel inträffar.

// OUTPUT:
// The Hitchhiker's Guide to the Galaxy
// Uncaught ReferenceError: bestBookEver is not defined
```