Funzioni in Javascript

Dichiarazione di funzione

```
function nomeFunzione ( )
{
    comandi Javascript…
}
```

Chiamata di funzione

Sintassi:

```
nomeFunzione();
```

Esempio:

```
onClick="calcola();"
```

definizione vs. chiamata

- la definizione di una funzione arricchisce l'insieme delle funzioni disponibili in Javascript;
- la sola definizione
  non causa l'esecuzione di azioni

- la chiamata di una funzione
  causa l'esecuzione delle istruzioni
  previste dalla funzione

Esempio di dichiarazione

```
function calcola()
{
  var prezzo = parseFloat(document.modulo.prezzoUnitario.value);
  var quantita = parseInt(document.modulo.quantita.value);
  var totale = prezzo*quantita;
  var iva = totale*20/100;
  var localeTot = totale + iva;
  document.modulo.totale.value = totale;
  document.modulo.iva.value = iva;
  document.modulo.totaleConIva.value = localeToT;
  return true;
}
```

Dichiarazioni nell'intestazione

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript">
…. Dichiarazione di funzione…
</SCRIPT>
</HEAD>
```

Chiamata nel corpo del documento

<BODY>

… evento =“nomeFunzione();”….

…

</BODY>

</HTML>

utilità delle funzioni

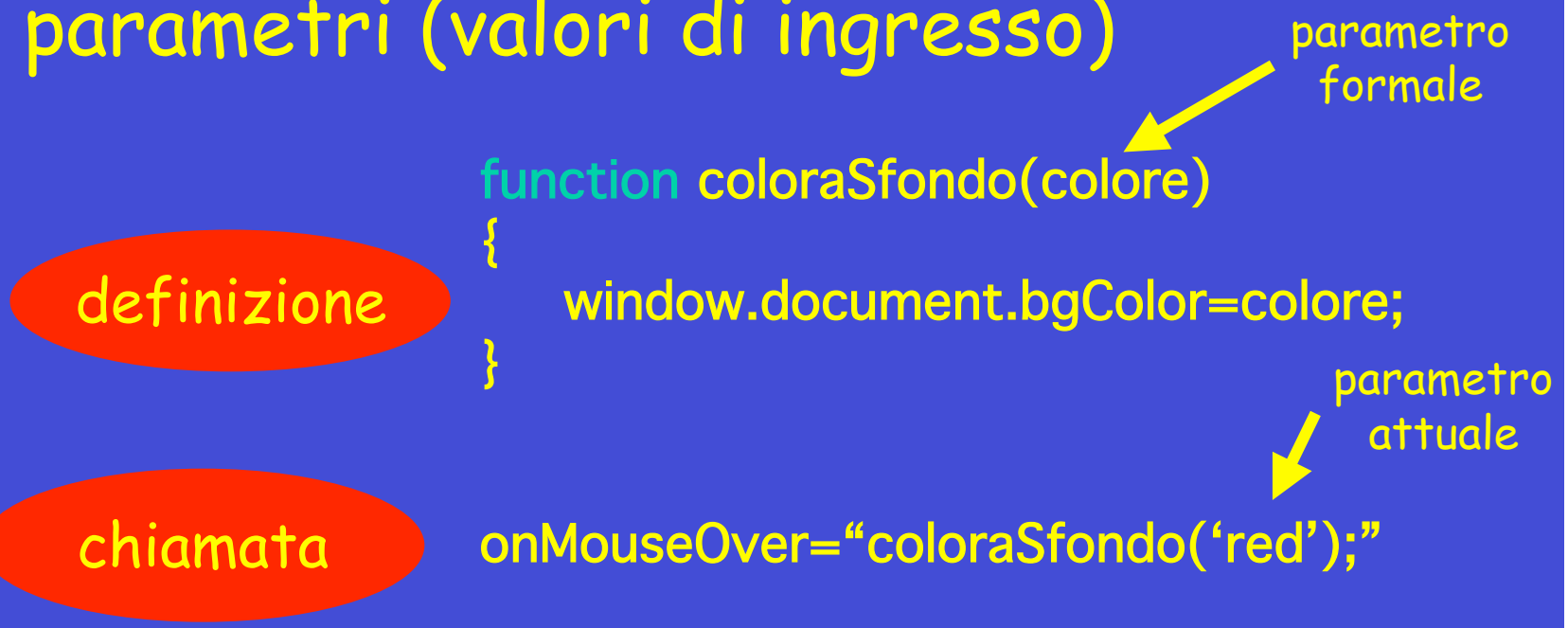
- una funzione viene definita una sola volta;
- nelle pagine html possono essere presenti più chiamate alla stessa funzione

codice più snello

riuso del codice
(nella stessa pagina
o altre pagine)

funzioni con parametri

- per aumentare la riutilizzabilità delle funzioni possono essere definite funzioni con parametri (valori di ingresso)



scopo delle funzioni

- si definiscono funzioni per raggiungere diversi obiettivi:
  - eseguire azioni (vedi funzione `calcola()`)
  - calcolare risultati

funzioni che restituiscono risultati

- una funzione può restituire un risultato attraverso l'istruzione

  return espressione;

- return termina l'esecuzione della funzione
- la chiamata della funzione assume il valore della espressione calcolata nel return

esempio di funzione con risultato

```
function approssimaAICentesimo(importo)
{
    var importoInCentesimi =
   Math.round(importo*100);
    return importoInCentesimi / 100;
}
```

definizione

chiamata

```
x = approssimaAICentesimo(lordo.value)
```

Istruzione condizionale (if … else)

```
if (  condizione )
{
    comandi Javascript…
}
```

Significato dell'istruzione if

Se la condizione è verificata, allora si eseguono i comandi racchiusi tra parentesi graffe {...}

Altrimenti, si passa direttamente al prossimo comando dopo la }

Esempio di utilizzo

```
function calcola()
{
    var quantita = parseFloat(document.ordine.quantita.value);
    var prezzo = parseFloat(document.ordine.prezzoUnitario.value);
    var aliquota = document.ordine.aliquota.value;
    var imp = prezzo * quantita;
    var iva = imp * aliquota;
    var totale = imp + iva;
    if (document.ordine.trasporto.checked==true) {
          totale = totale + 35;
    }
    document.ordine.imponibile.value = imp;
    document.ordine.iva.value = iva;
    document.ordine.totale.value = totale;
    return true;
}
```

Operatori di confronto

la condizione in una istruzione if è spesso il risultato di un confronto

if(quantita > 38) …

==	confronto di uguaglianza
!=	confronto per diversità
>	maggiore
<	minore
>=	maggiore o uguale
<=	minore o uguale

Istruzione condizionale (IF-ELSE)

Sintassi:

```
if ( condizione ) {
        … comandi …
}
else {
        … comandi …
}
```

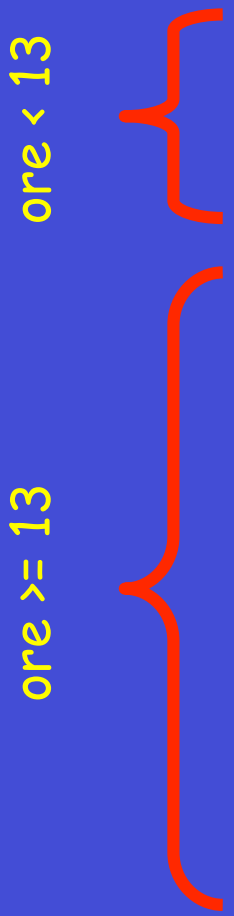
ESEGUITI SOLO SE LA CONDIZIONE è VERA

ESEGUITI SOLO SE LA CONDIZIONE è FALSA

if nidificati

- tra le istruzioni presenti all'interno di una istruzione if possono comparire altre istruzioni if
- un corretto incolonnamento del codice è indispensabile per permettere al programmatore di comprendere il funzionamento del codice

esempio




```
if (ore < 13)
{
    document.writeln("Buongiorno!");
}
else
{
    if (ore < 22)
    {
        document.writeln("Buonasera!");
    }
    else
    {
        document.writeln("Buonanotte...");
    }
}
```

condizioni composte

una condizione può essere formata dalla combinazione di più condizioni elementari, attraverso operatori logici

cond && cond	operatore AND
cond || cond	operatore OR
! cond	operatore NOT

esempio di condizioni composte

```
if ((valorePA >= 115)  &&  (valorePA <= 150))
{
   document.writ    n("Pressione normale");
}
```

le parentesi aggiunte
(in questo caso non necessarie)
facilitano la lettura