C Programming
for JavaScript Programmers

Networks and Embedded Systems

Second Grade Level

Wolfgang Neff

C Programming (1)

- JavaScript, C and C++ are almost the same

JavaScript

```
var n = 0;
if (n != 0) {
   n = 0;
}
for (var i = 0; i < 5; i++) {
   n = n + i;
}
```

C/C++

```
int n = 0;
if (n != 0) {
   n = 0;
}
for (int i = 0; i < 5; i++) {
   n = n + i;
}
```

C Programming (2)

- Do not forget the semicolon in C and C++

JavaScript

```
var n = 0
if (n != 0) {
   n = 0
}
for (var i = 0; i < 5; i++) {
   n = n + i
}
```

C/C++

```
int n = 0;
if (n != 0) {
   n = 0;
}
for (int i = 0; i < 5; i++) {
   n = n + i;
}
```

Semicolons are obligatory in C.

C Programming (3)

- In C and C++ everything has a datatype
  - The most important datatypes are:
    - **int**: integer numbers, e. g. 1, 2, 3, …
    - **double**: floating point numbers, e. g. 1.5, 2.0, …
    - **char**: characters, e. g. 'a', 'b', 'c', …
    - **char***: strings, e. g. "abc", “hello", …
    - **void**: function has no return value
  - We use **int** and **void**, only.

C Programming (4)

- Use datatype instead of *var* in C and C++

JavaScript

```
var x = 1;
var y = 2;
var z = x + y;
var pi = 3.14;
```

C/C++

```
int x = 1;
int y = 2;
int z = x + y;
double pi = 3.14;
```

C Programming (5)

- Use datatype instead of *function* in C and C++

JavaScript

```
function add(a,b)
{
   return a + b
}
function doit()
{
   // Do something
}
```

C/C++

```
int add(int a, int b)
{
   return a + b;
}
void doit()
{
   // Do something
}
```

C Programming (6)

- Do not forget the datatype of the parameters

Best Practice

```
int add(int a, int b)
{
   return a + b;
}
```

Worst Practice

```
int add(a, b)
{
   return a + b;
}
```

Everything must have a datatype

C Programming (7)

- Functions can be defined and called
  - Definition

```
int add(int a, int b)
{
    return a + b;
}
```

  - Call

```
int n = add(1,2);
```

C Programming (8)

- No function definition without datatype

Best Practice

```
int add(int a, int b)
{
   return a + b;
}
```

Worst Practice

```
add(a, b)
{
   return a + b;
}
```

C Programming (9)

- No function call with datatype

Best Practice

```
int n = add(1,2);
```

Worst Practice

```
int n = int add(int 1,2);
```

C Programming (10)

- There are global and local variables

```
// A global variable
int n = 1;
int f1(void)
{
    // A local variable
    int n = 2;
}
int f1(void)
{
    // Another local variable
    int n = 3;
}
```

C Programming (11)

- In C or C++ there is no global code

JavaScript

```
var a = 1;
var b = 2;
alert (a + b);
```

C/C++

```
void doit(void)
{
    int a = 1;
    int b = 2;
    printf("%d", a + b);
}
// Somewhere else
// ...
    doit();
// ...
```

C Programming (12)

- In C or C++ there are no local functions

Best Practice

```
int sqr(int n)
{
   return n * n;
}
double hyp(int a, int b)
{
   int a2 = sqr(a);
   int b2 = sqr(b);
   return sqrt(a2+b2);
}
```

Worst Practice

```
double hyp(int a, int b)
{
   // A local function
   int sqr(int n)
   {
      return n * n;
   }
   int a2 = sqr(a);
   int b2 = sqr(b);
   return sqrt(a2+b2);
}
```