



FUNCTIONS & SCOPE

FUNCTION DECLARATION

syntax

```
function name(parameters) {  
  // do something  
}
```

example

```
function speak(name) {  
  console.log("Hello, " + name);  
}
```

FUNCTION EXPRESSION

syntax

```
let name = function(parameters) {  
  // do something  
};
```

example

```
let speak = function(name) {  
  console.log("Hello, " + name);  
}
```

ARROW FUNCTION

syntax

```
let name = (parameters) => {  
  // do something  
}
```

example

```
let speak = (name) => {  
  console.log("Hello, " + name);  
}
```

CALLING A FUNCTION

syntax

```
name(arguments);
```

example

```
speak("Michelle");
```

VAR, LET, AND CONST

keyword	local scope	mutable	browsers
var	function only	yes	all
let	any block	yes	modern
const	any block	no	modern

FUNCTIONS AND HOISTING

type	name hoisted	content hoisted
declaration	yes	yes
let expression	no	no
var expression	yes	no

GLOBAL, LOCAL, AND BLOCK SCOPE

a variable declared outside of a function is in the **global scope**

```
let temp = 75;
function predict() {
  let forecast = 'Sun';
  console.log(temp + " and " + forecast);
  // 75 and Sun
}
console.log(temp + " and " + forecast);
// 'forecast' is undefined
```

a variable declared within a function is in the **local scope** of that function

```
let temp = 75;
if (temp > 70) {
  let forecast = 'It's gonna be warm!';
  console.log(temp + "! " + forecast);
  // 75! It's gonna be warm!
}
console.log(temp + "! " + forecast);
// 'forecast' is undefined
```

a variable declared with let and within a block, such as an if statement, is in the **block scope** of that statement