Functions

Part 1

What is a Function?

A function is a block of organized, reusable code that is used to perform a single, related action.

Functions we know

`console.log`

`prompt`

`Math.floor`

`Array.push`

Why Functions?

- to reduce repetition
- to associate names with subprograms
- to isolate these subprograms from each other
- to structure larger programs

Defining Functions

```
function functionName(functionArguments) {
  // function body
}
```

Declaration

```
var myFunction = function(...) {
        ...
}
```

```
function myFunction(...) {
        ...
}
```

EXAMPLE

Invocation

```
function functionName(functionArguments) {
  // function body
}

functionName(argumentValues);
```

Default values

If a parameter is not provided, then its value becomes undefined.

But you can declare default values for such cases.

EXAMPLE

Default values

Keep in mind!
Default values work only for undefined values. If value of the passed argument is null, it will remain null in the whole funcion.

EXAMPLE

Arguments And Parameters

- no declared types
- no type checking
- fewer arguments are ok. they set to undefined
- even sometimes desirable to have optionals, give them reasonable defaults

EXAMPLE

Return

```
return expression;
```

- may appear only within the body of a function
- it is a Syntax Error to appear anywhere else

Return

```
return;
return true;
return false;
return x;
return x + y / 3;
```

Function and Return

When the return statement is executed, the function that contains it returns the value of expression to its caller.

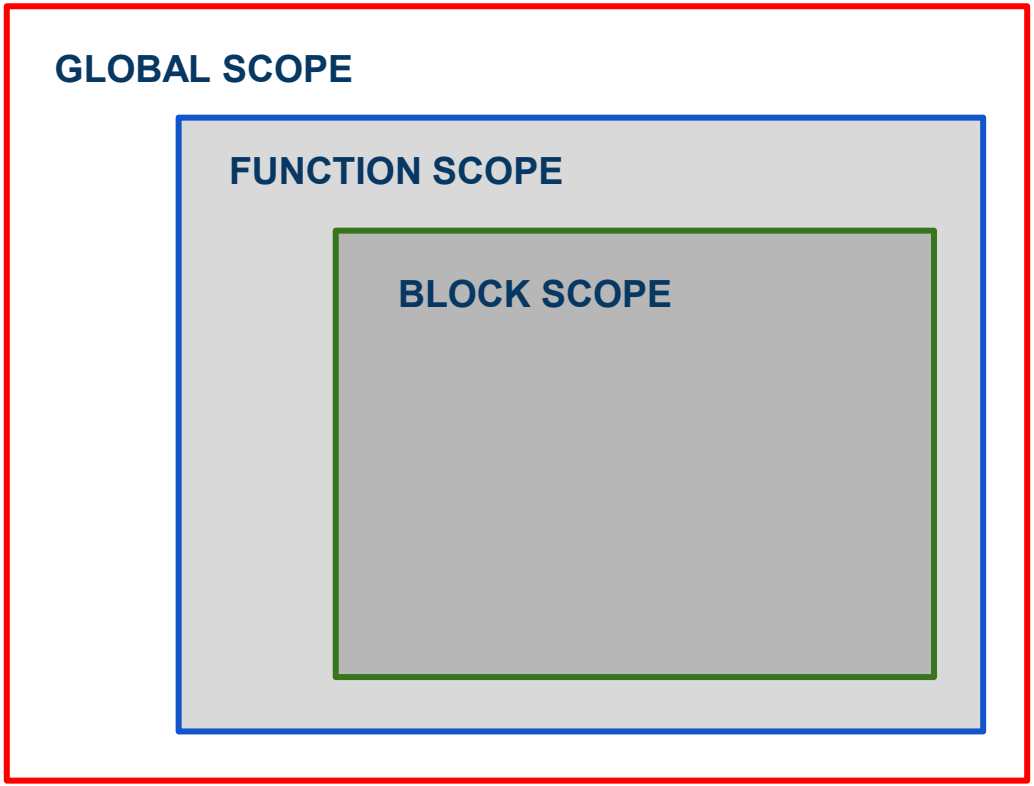
With no return statement, a function invocation simply executes each of the statements in the function body in turn until it reaches the end of the function and returns undefined.

EXAMPLE

Arrow functions

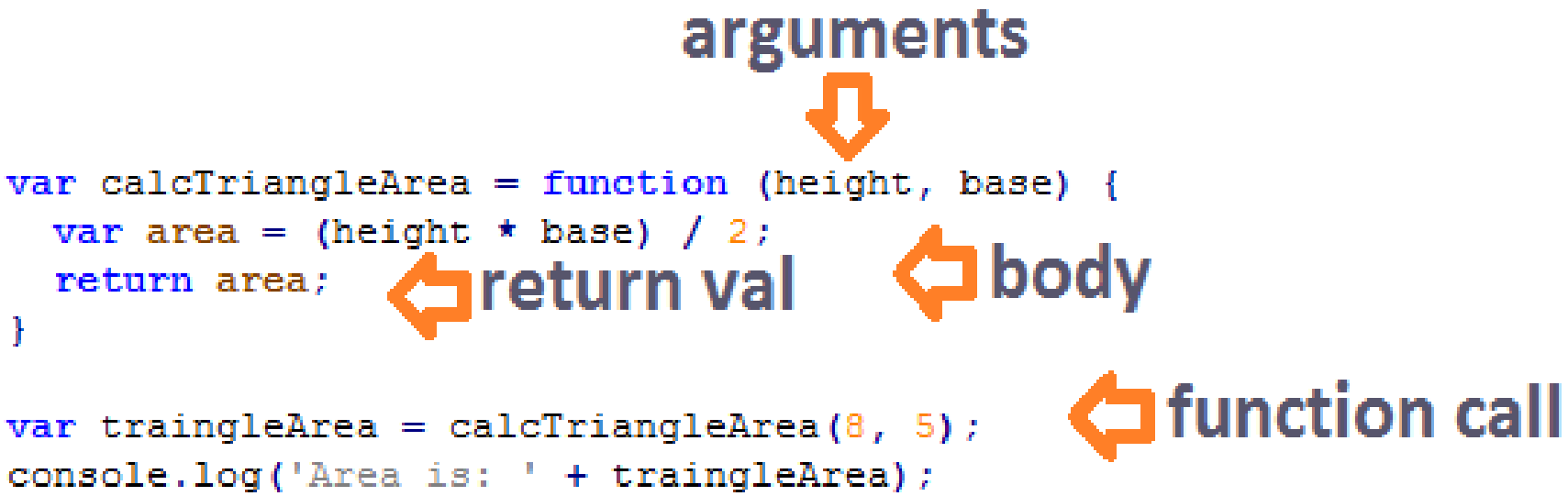
```
let func = (arg1, arg2, ...argN) => expression
```

SCOPE



EXAMPLE

Declaration, Arguments, Return, Call



Thank You!