

Conditionals }

Client Side Web

Conditionals

- { Conditionals
- { Conditional Operators
- { Returning Values

Conditionals



Conditionals

The comparison of at least two values resulting in either true or false.



Conditionals are everywhere!

If the gas pedal is
pressed go faster.

If the gas pedal is

1

```
if(peadal=="pressed"){  
    goFaster()  
}
```

Types of Conditionals

- { if
- { else
- { if else
- { switch

If Statement

```
if(condition) {  
    doSomething  
}
```

If Statement

```
if (condition) {  
    doSomething  
}
```

**The condition is
evaluated as
true or false.**

If Statement

```
if(condition) {  
    doSomething  
}
```

**If condition is met,
the code block
will execute.**



Comparison Operators

Comparison Operators

Used to compare the value of at least two values.

Comparison Operators

`{ ==`

`{ !=`

`{ >`

`{ >=`

`{ <`

`{ <=`

`{ !`

Comparison Operators

{ ==

{ !=

{ >

{ >=

{ <

{ <=

**Learn more about
each operator via
the link below.**

http://www.w3schools.com/JS/js_comparisons.asp

Comparison Operators in Action

```
var name="josh"
```

```
if (name=="josh") {
```

```
    alert("hello Josh!");
```

```
}
```

Comparison Operators in Action

```
var name="josh"
```

```
if(name=="josh"){
```

```
    alert("hello Josh!");
```

```
}
```

== compares the two values. It's asking if the values are equal.

Comparison Operators in Action

```
var name="josh"
```

```
if (name=="josh") {
```

```
    alert("hello Josh!");
```

```
}
```

is name...

Comparison Operators in Action

```
var name="josh"
```

```
if (name=="josh") {
```

```
    alert("hello Josh!");
```

```
}
```

...the same as "josh"?

Comparison Operators in Action

```
var name="josh"
```

```
if(name=="josh"){
```

```
    alert("hello Josh!");
```

```
}
```

Yes!

Comparison Operators in Action

```
var name="josh"
```

```
if (name=="josh") {
```

```
    alert("hello Josh!");
```

```
}
```

**Because
name="josh".**

Returning Values



Returning Values

A way of sending a “receipt” of what it has executed inside of a function.

Typical Function

```
function addMe(){  
  var result  
  result = 2 + 3;  
}
```

Typical Function

```
function addMe() {  
  var result  
  result = 2 + 3;  
}
```

**We know by looking
at the code the
answer is 5.**

Returning Values

```
function addMe(){  
  var result  
  result = 2 + 3;  
  return result;  
}
```

Returning Values

```
function addMe(){  
  var result  
  result = 2 + 3;  
  return result;  
}
```

**Now we can send
the answer when
the function
has executed.**

Returning Values

```
function addMe(){  
  var result  
  result = 2 + 3;  
  return result;  
}
```

```
alert(addMe());  
// alerts 5
```

Returning Values

```
function addMe(){  
  var result  
  result = 2 + 3;  
  return result;  
}
```

```
alert(addMe());  
// alerts 5
```

This would alert 5

Neat... but why?

It's a cleaner way to get information from a calculation, and it's often used for comparison with conditionals.

script.js

```
function checkUser(user) {  
    if(user=='josh') {  
        return true;  
    }  
    return false;  
}  
  
function allowAccess() {  
    if(checkUser('josh')) {  
        alert('you are allowed here');  
    }  
}
```

script.js

```
function checkUser(user) {  
    if(user=='josh') {  
        return true;  
    }  
    return false;  
}  
  
function allowAc  
    if(checkUser(''  
        alert('you a  
    }  
}
```

**Note the values
being returned.**

script.js

```
function checkUser() {
  if (user === 'josh') {
    return true;
  }
  return false;
}

function allowAccess() {
  if (checkUser('josh')) {
    alert('you are allowed here');
  }
}
```

**It's being used
for comparison in a
conditional. Sneaky.**

index.html

```
<form>
```

```
  <input type='button' value='check'  
onclick='allowAccess()' />
```

```
</form>
```

index.html

```
<form>
```

```
  <input type='button' value='check'  
onclick='allowAccess()' />
```

```
</form>
```

What happens?



You are allowed in.

Conditionals: Summary

- { Conditionals
- { Conditional Operators
- { Returning Values