

JavaScript Functions

Adding Capabilities to Web Pages

JavaScript

Function Declaration

Functions enable the developer to expand capabilities

- Declared using the 'function' keyword and curly braces to contain the code
- Each function has a name
- Parenthesis designate the parameters, even if none exist

```
function getUsername() {  
    var userName = prompt("What's your name?", "");  
}
```

JavaScript

Variable Scope

- Global
 - Declared outside of any function or object
 - Can be accessed from anywhere

```
<script>
var greeting = "Hello World"; // Global Scope.

function getUsername ( ) {
    // Local Scope - userName only works here.
    var userName = prompt("What's your name?", "");
}

function sayHello ( ) {
    alert (greeting); // This works.
    alert ("Hello " + userName); // This doesn't.
}
</script>
```

JavaScript

Variable Scope

- Global

- Declared outside of any function or object
- Can be accessed from anywhere

```
<script>
var greeting = "Hello World"; // Global Scope.

function getUser_name( ) {
    // Local Scope - userName only works here.
    var userName = prompt("What's your name?", "");
}
```

- Local

- Declared inside code blocks
- Only accessible from within the code block

```
function sayHello ( ) {
    alert (greeting); // This works.
    alert ("Hello " + userName); // This doesn't.
}
</script>
```

JavaScript

Function Parameters

- Parameters pass data into a function
 - Parameters are local in scope
 - Helps to customize the function

```
<script>
var greeting = "Hello World"; // Global Scope.

function getUsername( ) {
    // Local Scope - userName only works here.
    var userName = prompt("What's your name?", "");
}

function sayHello (nameToUse) {
    if (nameToUse == "") {
        alert (greeting);
    } else {
        alert ("Hello " + nameToUse);
    }
}
</script>
```

JavaScript

Returning Values from a Function

- Functions can send data back to the code that called it
 - Keyword 'return' is used to send data
 - 'return' immediately stops execution of the function

```
<script>
var greeting = "Hello World"; // Global Scope.

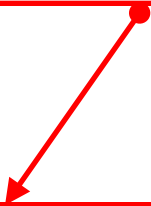
function getUsername( ) {
    // Local Scope - userName only works here.
    var userName = prompt("What's your name?", "");
    return userName;
}

function sayHello (nameToUse) {
    if (nameToUse == "") {
        alert (greeting);
    } else {
        alert ("Hello " + nameToUse);
    }
}
</script>
```

JavaScript

Returning Values from a Function

```
var name = getUserName();  
sayHello(name);
```



```
sayHello(getUserName());
```



```
<input type="button" value="Say Hello"  
onclick="sayHello(getUserName())">
```

```
<script>  
var greeting = "Hello World"; // Global Scope.  
  
function getUserName( ) {  
    // Local Scope - userName only works here.  
    var userName = prompt("What's your name?", "");  
    return userName;  
}  
  
function sayHello (nameToUse) {  
    if (nameToUse == "") {  
        alert (greeting);  
    } else {  
        alert ("Hello " + nameToUse);  
    }  
}  
</script>
```