Netscape Navigator 4 Object Road Map

Learn how to create "Web Pages That Think"™

How to Use This Map

The hierarchy diagram shows the relationships among various HTML-generated objects in Netscape Navigator 4. Follow the dark lines to create references to objects:

```
window.document.formName.textName.value
```

The three columns in the individual object listings show the Properties, Methods, and Event Handlers for the object.

©1998 Danny Goodman. All Rights Reserved.
**Browser Compatibility Guide**

This map is a Netscape-centric view of JavaScript. No items unique to Microsoft Internet Explorer (MSIE) are listed, but Netscape-only items are.

Items that have no superscript notation are available to all JavaScript-enabled browsers, including Navigator 2 and MSIE 3. Superscript notations are keyed as follows:

- **N3** -- New in Netscape 3
- **N4** -- New in Netscape Navigator 4
- **M3** -- New in Netscape 3 and MSIE 3
- **M4** -- New in MSIE 4
- **J2** -- New in MSIE 3, JScript.dll Version 2
- (S) -- Requires Netscape Navigator 4 Signed Scripts

If an item shows N3 or N4 but not M3 or M4, it means that the feature is Netscape-only, and is not available in MSIE.

**JavaScript Levels**

JavaScript has gone through three versions. The following browsers support features from the corresponding JavaScript level:

<table>
<thead>
<tr>
<th>Browser</th>
<th>JavaScript Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Netscape Navigator 2.0x</td>
<td>1.0</td>
</tr>
<tr>
<td>MS Internet Explorer 3.0x</td>
<td>1.0</td>
</tr>
<tr>
<td>Netscape Navigator 3.0x</td>
<td>1.1</td>
</tr>
<tr>
<td>Netscape Navigator 4.0x</td>
<td>1.2</td>
</tr>
<tr>
<td>MS Internet Explorer 4.0x</td>
<td>1.2</td>
</tr>
</tbody>
</table>

No other mainstream browsers support JavaScript.

©1998 Danny Goodman. All Rights Reserved.
### Operators

<table>
<thead>
<tr>
<th>Comparison</th>
<th>Arithmetic</th>
<th>Assignment</th>
<th>Boolean</th>
<th>Bitwise</th>
<th>Miscellaneous</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code> Equals</td>
<td><code>+</code> Plus (and string concat.)</td>
<td><code>=</code> Equals</td>
<td><code>&amp;&amp;</code> AND</td>
<td><code>&amp;</code> Bitwise AND</td>
<td><code>delete</code> Property destroyer</td>
</tr>
<tr>
<td><code>!=</code> Does not equal</td>
<td><code>-</code> Minus</td>
<td><code>+=</code> Add by value</td>
<td>`</td>
<td></td>
<td>` OR</td>
</tr>
<tr>
<td><code>&gt;</code> Is greater than</td>
<td><code>*</code> Multiply</td>
<td><code>-=</code> Subtract by value</td>
<td><code>!</code> NOT</td>
<td><code>^</code> Bitwise XOR</td>
<td><code>this</code> Object self-reference</td>
</tr>
<tr>
<td><code>&gt;=</code> Is greater than or equal to</td>
<td><code>/</code> Divide</td>
<td><code>*=</code> Multiply by value</td>
<td><code>~</code> Bitwise NOT</td>
<td><code>&lt;&lt;</code> Left shift</td>
<td><code>typeof</code> Value type</td>
</tr>
<tr>
<td><code>&lt;</code> Is less than</td>
<td><code>%</code> Modulo by value</td>
<td><code>/=</code> Divide by value</td>
<td><code>&gt;&gt;</code> Right shift</td>
<td><code>&gt;&gt;=</code> Zero fill right shift</td>
<td><code>void</code> Return no value</td>
</tr>
<tr>
<td><code>&lt;=</code> Is less than or equal to</td>
<td><code>++</code> Increment</td>
<td><code>+=</code> Add by value</td>
<td><code>&gt;&gt;&gt;</code> Zero fill right shift</td>
<td><code>&gt;&gt;&gt;&gt;</code></td>
<td><code>watch(id, handler)</code></td>
</tr>
<tr>
<td></td>
<td><code>--</code> Decrement</td>
<td><code>-=</code> Subtract by value</td>
<td></td>
<td></td>
<td><code>unwatch(id)</code></td>
</tr>
<tr>
<td></td>
<td><code>-val</code> Negation</td>
<td><code>*=</code> Multiply by value</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Control Statements

```plaintext
if (condition) {
  statements
}

if (condition) {
  statements
} else {
  statements
}
```

```plaintext
result = condition ? expression1 : expression2
```

```plaintext
for ([init expr]; [condition]; [update expr]) {
  statements
}
```

```plaintext
for (var in object) {
  statements
}
```

```plaintext
while (condition) {
  statements
}
```

```plaintext
with (object) {
  statements
}
```

```plaintext
do {
  statements
} while (condition)
```

```plaintext
switch (expression) {
  case label1 :
    statements
    [break]
  case label2 :
    statements
    [break]
  ...
  [default :
    statements]
}
```

```plaintext
function continues [label] while (condition)
```

### Global Functions

```plaintext
watch(id, handler)
```

```plaintext
unwatch(id)
```

```plaintext
toString()
```

```plaintext
escape("str", [1])
```

```plaintext
unescape("str")
```

```plaintext
isNaN(expression)
```

```plaintext
Number("str")
```

```plaintext
parseFloat("str")
```

```plaintext
parseInt("str")
```

```plaintext
parseFloat("str")
```

```plaintext
parseInt("str")
```

```plaintext
isNaN(expression)
```

```plaintext
Number("str")
```

```plaintext
parseFloat("str")
```

```plaintext
parseInt("str")
```

```plaintext
isNaN(expression)
```

```plaintext
Number("str")
```

```plaintext
parseFloat("str")
```

```plaintext
parseInt("str")
```

```plaintext
isNaN(expression)
```

```plaintext
Number("str")
```

```plaintext
parseFloat("str")
```

```plaintext
parseInt("str")
```

```plaintext
isNaN(expression)
```

```plaintext
Number("str")
```

```plaintext
parseFloat("str")
```

```plaintext
parseInt("str")
```