The **`Node.childNodes`** read-only property returns a live `NodeList` of child `nodes` of the given element where the first child node is assigned index 0.

Syntax

```
let nodeList = elementNodeReference.childNodes;
```

Examples

Simple usage

```
// parg is an object reference to a <p> element

// First check that the element has child nodes
if (parg.hasChildNodes()) {
  let children = parg.childNodes;

  for (let i = 0; i < children.length; i++) {
    // do something with each child as children[i]
    // NOTE: List is live! Adding or removing children will change the
  }
}
```

Remove all children from a node

```
// This is one way to remove all children from a node
// box is an object reference to an element
```

```
while (box.firstChild) {
    //The list is LIVE so it will re-index each call
    box.removeChild(box.firstChild);
}
```

Notes

The items in the collection of nodes are objects, not strings. To get data from node objects, use their properties. (For example, to get the name of the first childNode: `elementNodeReference.childNodes[1].nodeName`.)

The `document` object itself has 2 children: the Doctype declaration and the root element, typically referred to as `documentElement`. (In (X)HTML documents this is the `HTML` element.)

`childNodes` includes *all* child nodes—including non-element nodes like text and comment nodes. To get a collection of only elements, use `ParentNode.children` instead.