Home > puppeteer > ExecutionContext > evaluate

ExecutionContext.evaluate() method

Signature:

```
evaluate<ReturnType extends any>(pageFunction: Function | string, ...args:
unknown[]): Promise<ReturnType>;
```

Parameters

Parameter	Type	Description
pageFunction	Function | string	a function to be evaluated in the `executionContext`
args	unknown[]	argument to pass to the page function

Returns:

Promise<ReturnType>

A promise that resolves to the return value of the given function.

Remarks

If the function passed to the `executionContext.evaluate` returns a Promise, then `executionContext.evaluate` would wait for the promise to resolve and return its value. If the function passed to the `executionContext.evaluate` returns a non-serializable value, then `executionContext.evaluate` resolves to `undefined`. DevTools Protocol also supports transferring some additional values that are not serializable by `JSON`: `-0`, `NaN`, `Infinity`, `-Infinity`, and bigint literals.

Example 1

```
const executionContext = await page.mainFrame().executionContext();
const result = await executionContext.evaluate(() => Promise.resolve(8 * 7))* ;
console.log(result); // prints "56"
```

Example 2

A string can also be passed in instead of a function.

```
console.log(await executionContext.evaluate('1 + 2')); // prints "3"
```

Example 3

JSHandle instances can be passed as arguments to the `executionContext.* evaluate`:

```
const oneHandle = await executionContext.evaluateHandle(() => 1);
const twoHandle = await executionContext.evaluateHandle(() => 2);
const result = await executionContext.evaluate(
   (a, b) => a + b, oneHandle, * twoHandle
);
await oneHandle.dispose();
await twoHandle.dispose();
console.log(result); // prints '3'.
```