Steve
Software Engineer (2006–present)Author has **907** answers and **1M** answer views4y

If I'm understanding your question correctly, then given the following code, you are wanting to retrieve the function `internal()` from the variable `wrapped`.

```
function internal() {
  console.log(this, arguments);
}

var wrapped = internal.bind({});
```

You can't do this. The bind function works like this:

```
Function.prototype.bind = function(target, ...arg) {
  var fn = this;
  return function(...uments) {
    return fn.apply(target, [...arg, ...uments]);
  };
};
```

You are receiving a JavaScript closure, and the reference to the function exists within that closure. Although a child scope can access its parent scope, the reverse is not true. If you want to be able to access the original function, you should probably attach it as a property:

```
var wrapped = internal.bind({});
wrapped.bound = internal;
```

That'll ensure it's available when you need it later.

For a slightly deeper explanation of JavaScript's scope and how it works, you can read my post about it at JavaScript:
Scope (https://www.stevethedev.com/blog/programming/javascript-scope-primer).