On Contracts, Sandboxes, and Proxies for JavaScript

Matthias Keil, Peter Thiemann
University of Freiburg, Germany

Pörtschach am Wörthersee, Österreich

TreatJS

- Language embedded contract system for JavaScript
- Enforced by run-time monitoring
- Standard abstractions for higher-order-contracts (base, function, and dependent contracts) [Findler,Felleisen'02]
- Systematic blame calculation
- Contract constructors generalize dependent contracts
- Internal noninterference

Base Contract [Findler,Felleisen'02]

- Base Contracts are built from predicates
- Specified by a plain JavaScript function

Base Contract

```javascript
function isNumber (arg) {
  return (typeof arg) === 'number';
};
var Number = Contract.Base(isNumber);
assert(1, Number);
assert('a', Number);
Enter the blame the subject
```
Function Contract [Findler,Felleisen’02]

```
// Number × Number → Number
function plus(x, y) {
  return x + y;
}
```

Function Contract

```
var Plus = Contract Function([Number, .Number], .Number);
var plusNumber = assert(plus, Plus);
plusNumber(1, 1); ✓
plusNumber(2, 2); # blame the context
```

JavaScript Proxies

```
handler.get(target, 'x', proxy);
handler.set(target, 'y', 1, proxy);
...
target['x'];
target['y']=1;
```

Noninterference

The contract system should not interfere with the execution of application code.

- **External Noninterference** arises from the interaction of the contract system with the host program
  - Exceptions
  - Object equality
- **Internal Noninterference** arises from executing unrestricted code in predicates

Notizen

Notizen

Notizen
Internal Noninterference

- No syntactic restrictions on predicates
- Predicates may try to write to data that is visible to the application
- Solution: Predicate evaluation takes place in a sandbox

Faulty Predicate

```javascript
function isNumber(arg) {
    type = typeof arg; // access forbidden
    return type === 'number';
}
```

Solution: Predicate evaluation takes place in a sandbox

TreatJS Sandbox

- All contracts guarantee noninterference
- Read-only access is safe

Predicate with Read-Access

```javascript
function isArray(arg) {
    return (arg instanceof Array);
}
```

Predicate execution may violate contracts

Solution: Sandbox redefines the responsibility

Predicate with Read-Access

```javascript
function addOne(arg) {
    return plusNumber(arg, '1'); // blame the contract
}
```
**Sandbox Encapsulation**

**Faulty Predicate**

```javascript
function isNumber(arg) {
    type = (typeof arg);
    return type === 'number';
}
```

- Place a write protection on objects (e.g. `this`, `arg`)
- Remove external bindings of functions (e.g. `type`)

---

**Identity Preserving Membrane**

```
ProxyA
ProxyB
ProxyC
```

```
TargetA
TargetB
TargetC
```

---

**Shadow Objects**

```
handler.get(target, 'x', proxy);
handler.set(target, 'y', 1, proxy);
```

```
proxy.x;
proxy.y=1;
target['x'];
target['y']=1;
```

---

Notizen

Notizen

Notizen
Shadow Objects

```
handler.get(target, 'x', proxy);
handler.set(target, 'y', 1, proxy);
handler.get(target, 'y', proxy);
```

```
target['x'];
scaffold['y'] = 1;
scaffold['y'];
```

Meta-Level
Base-Level

Matthias Keil, Peter Thiemann
On Contracts, Sandboxes, and Proxies
October 5, 2015 12 / 18

Function Recompilation

```
var x = 1;

function f (){
    function g (y) {
        var z = 1;
        return x+y+z;
    }
}
```

Matthias Keil, Peter Thiemann
On Contracts, Sandboxes, and Proxies
October 5, 2015 13 / 18

Function Recompilation

```
var x = 1;

function f (){
    "function g (y) {
        var z = 1;
        return x+y+z;
    }"
}
```

Matthias Keil, Peter Thiemann
On Contracts, Sandboxes, and Proxies
October 5, 2015 13 / 18

Notizen
Function Recompilation

```javascript
var x = 1;
with(sbxglobal){
    eval("function g (y) {
        var z = 1;
        return x+y+z;
    }" );
}
```

Matthias Keil, Peter Thiemann
On Contracts, Sandboxes, and Proxies
October 5, 2015 13 / 18

Conclusion

- TreatJS:
  - Language embedded, dynamic, higher-order contract system for full JavaScript
  - Guarantees noninterference

- Sandbox:
  - Language embedded sandbox for full JavaScript
  - Runs JavaScript code in isolation