Weak Pointers and Ephemeron Tables for EcmaScript Harmony

Mark S. Miller

Classic Soft Fields

```
private decr;
function Purse(balance) {
  const purse = Object.freeze({
    deposit: function(amount, src) {
      src[decr](-amount);
      balance += amount;
    }
  });
  purse[decr] = function(delta) {
    // ... throw if delta isn't right  ...
    balance -= delta;
  };
  return purse;
}
```

```
const decr = EphemeronTable();
function Purse(balance) {
  const purse = Object.freeze({
    deposit: function(amount, src) {
      decr.get(src)(-amount);
      balance += amount;
    }
  });
  decr.set(purse, function(delta) {
    // ... throw if delta isn't right  ...
    balance -= delta;
  });
  return purse;
}
```

Soft Fields with Inheritance

Google

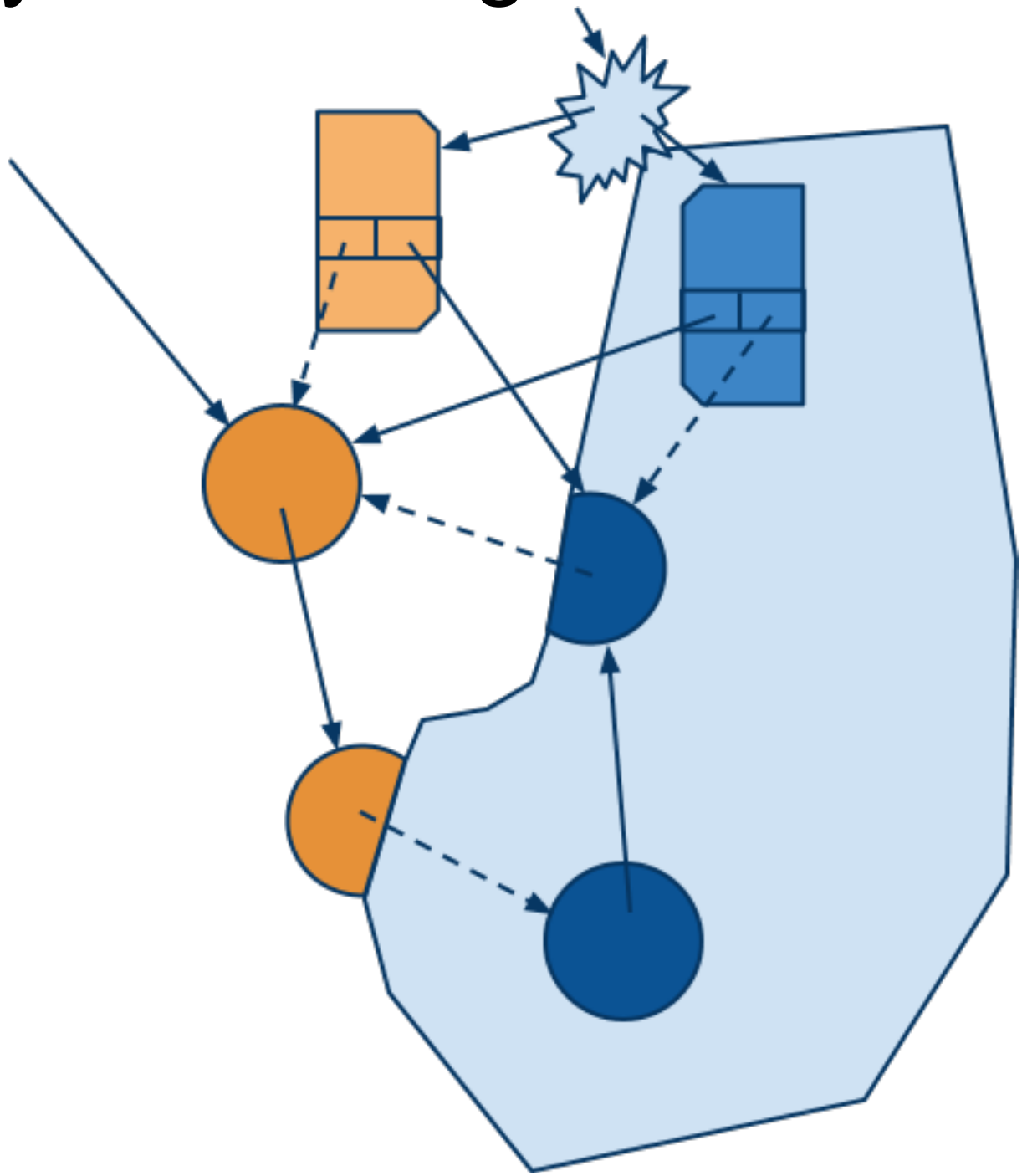
```
function SoftField() {
  const et = EphemeronTable();
  return Object.freeze({
    get: function(key) {
      while (key !== null) {
        const result = et.get(key);
        if (result !== undefined) { return result; }
        key = Object.getPrototypeOf(key);
      }
    },
    set: et.set;
  });
}
```

Unique Labeller

Google

```
function Labeler() {
  const et = EphemeronTable();
  let count = 0;
  return Object.freeze({
    label: function(obj) {
      const result = et.get(obj);
      if (result) { return result; }
      et.put(obj, ++count);
      return count;
    }
  });
}
```

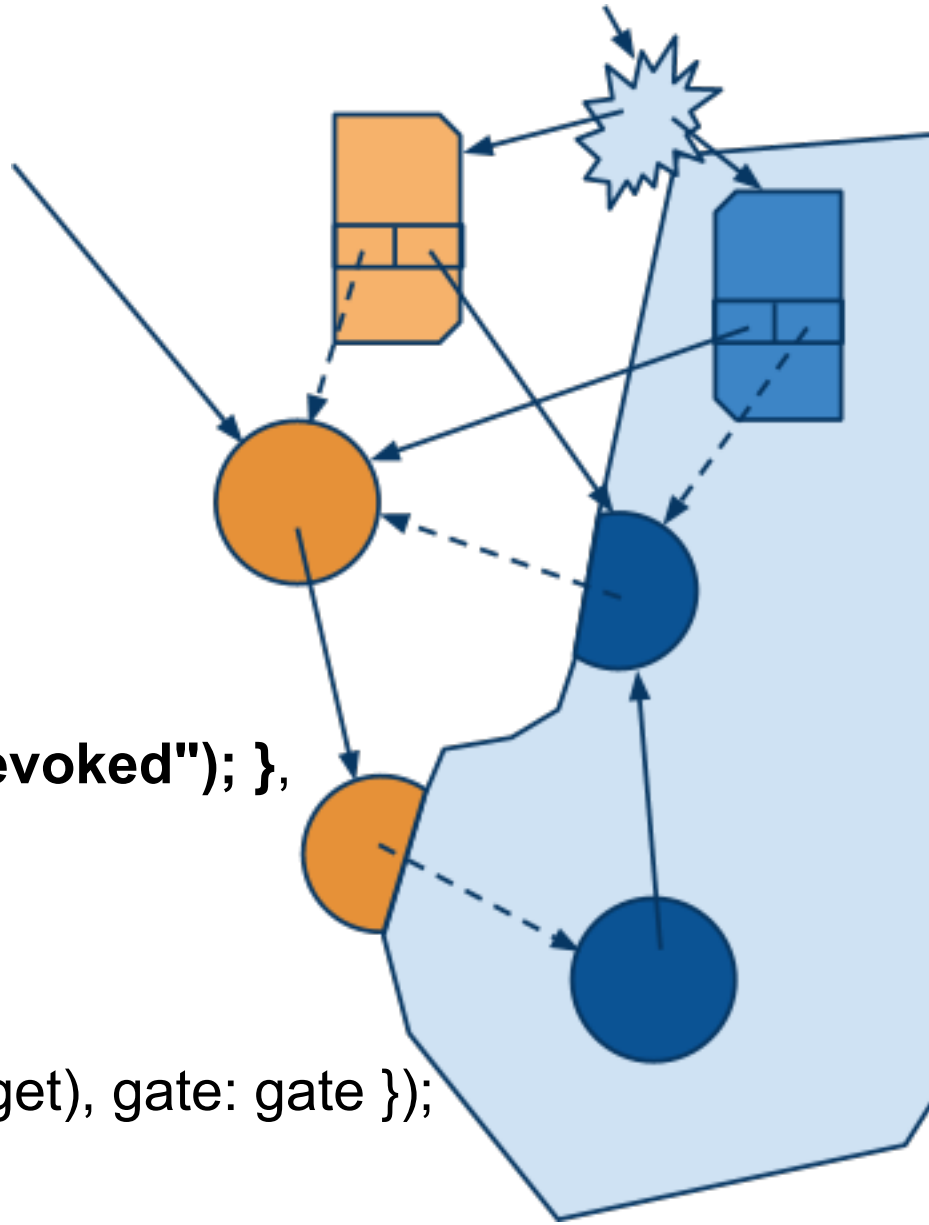
Identity Preserving Membranes



Better Membrane (Overview)

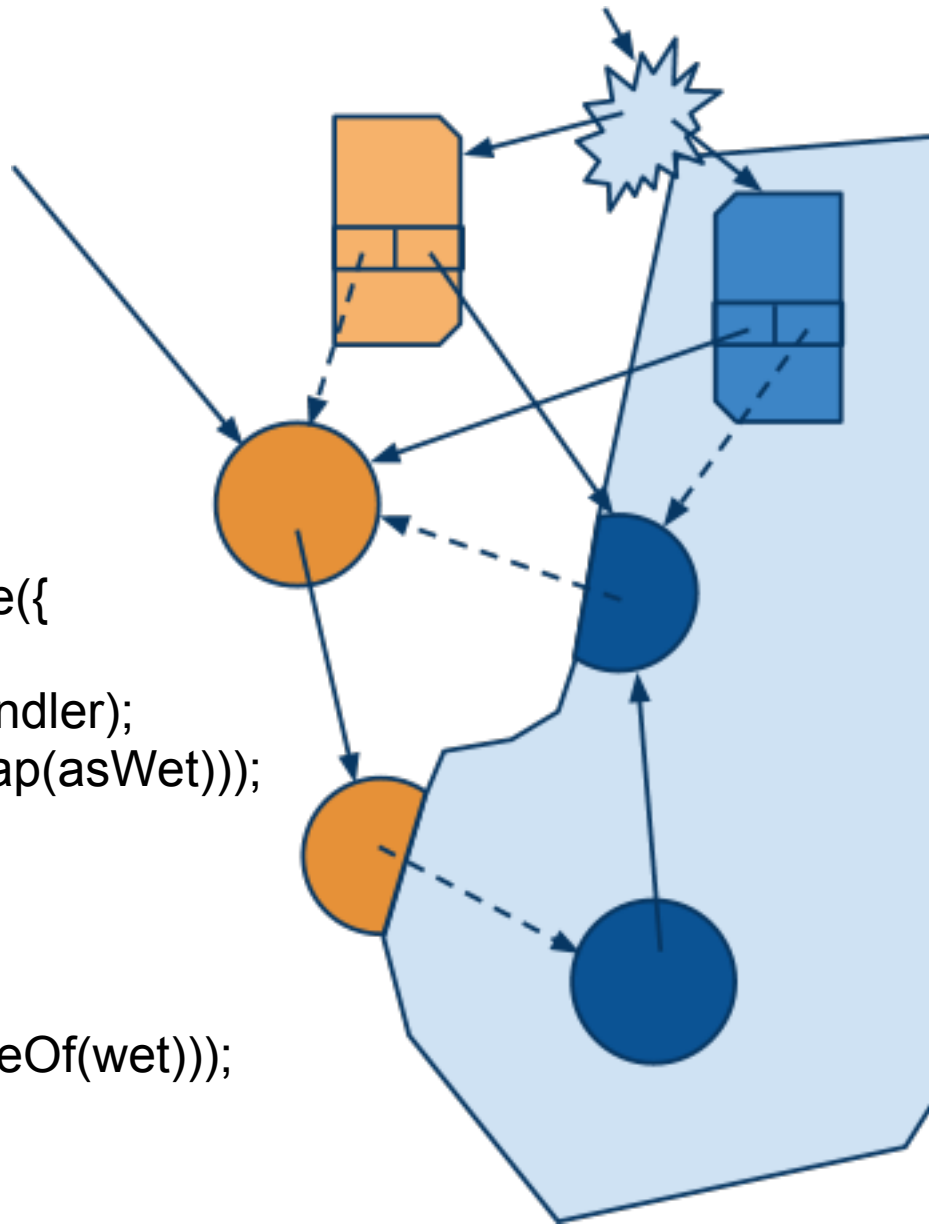
Google

```
function makeMembrane(wetTarget) {
  var wet2dry = EphemeronTable(true);
  var dry2wet = EphemeronTable(true);

  function asDry(wet) {/*...*/}
  function asWet(dry) {/*...*/}

  var gate = Object.freeze({
    revoke: function() {
      dry2wet = wet2dry = Object.freeze({
        get: function(key) { throw new Error("revoked"); },
        set: function(key, val) {}
      });
    }
  });
  return Object.freeze({ wrapper: asDry(wetTarget), gate: gate });
}
```



Better Membrane (Detail)

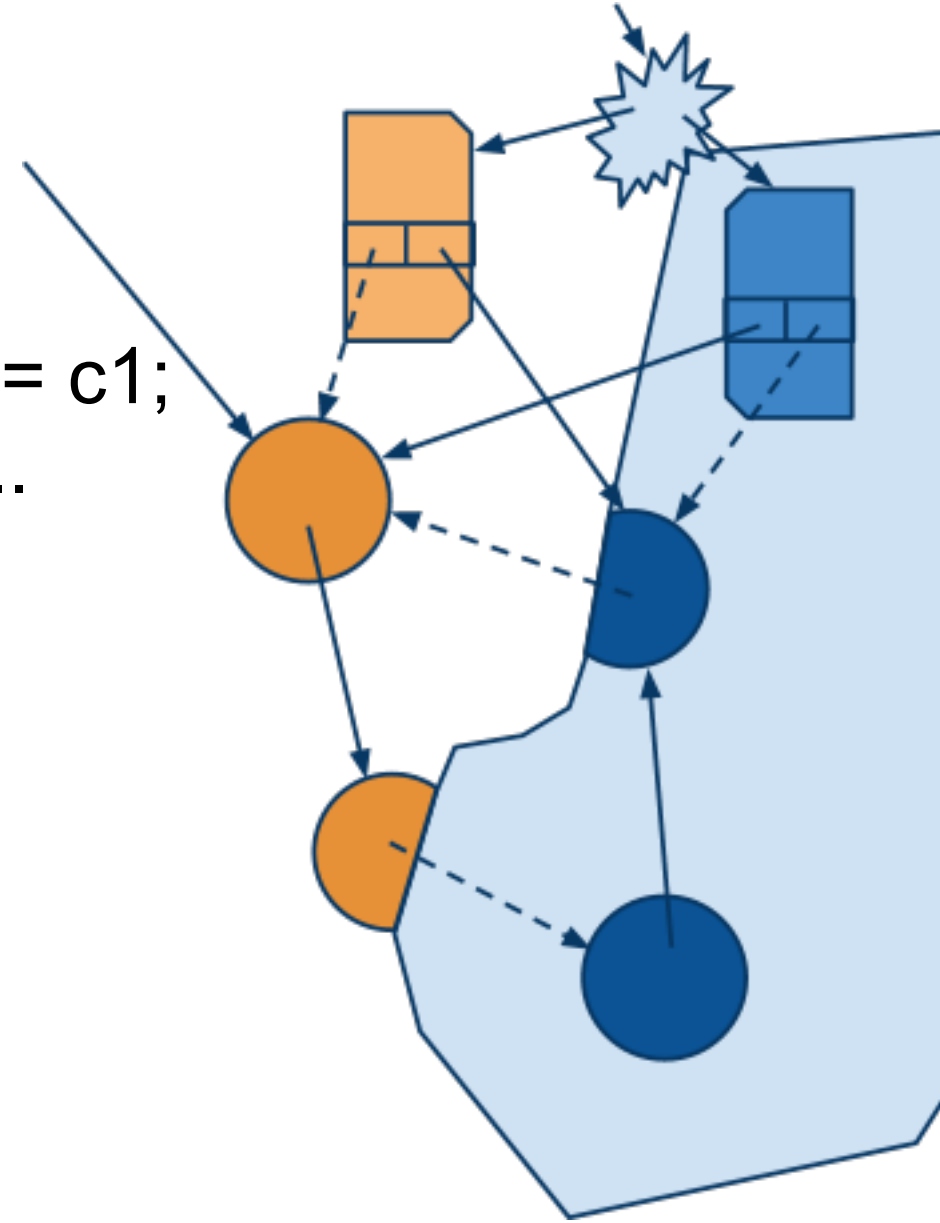
```
function makeMembrane(wetTarget) {
  var wet2dry = EphemeronTable(true);
  var dry2wet = EphemeronTable(true);

  function asDry(wet) {
    if (wet !== Object(wet)) { return wet; }
    var dryResult = wet2dry.get(wet);
    if (dryResult) { return dryResult; }

    var wetHandler = makeHandler(wet);
    var dryRevokeHandler = Proxy.create(Object.freeze({
      invoke: function(rcvr, name, dryArgs) {
        var optWetHandler = dry2wet.get(dryRevokeHandler);
        return asDry(optWetHandler[name](...dryArgs.map(asWet)));
      }
    }));
    dry2wet.set(dryRevokeHandler, wetHandler);
    dryResult = Proxy.create(dryRevokeHandler,
                             asDry(Object.getPrototypeOf(wet)));
    wet2dry.set(wet, dryResult);
    dry2wet.set(dryResult, wet);
    return dryResult;
  }
....
```



Compartments

Google

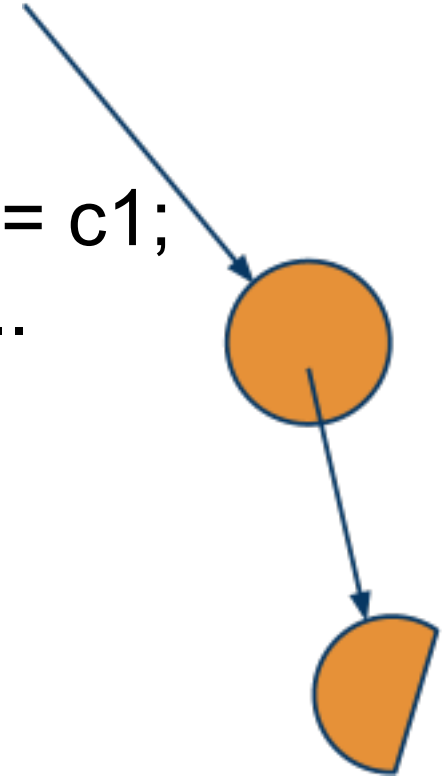
```
var c1 = makeMembrane(eval);
var {wrapper: eval1, gate: gate1} = c1;
var badCode = ... get bad code  ...
var result = eval1(badCode);
... use result ...
gate1.revoke();
... compartment gone  ...
```



Compartments

Google

```
var c1 = makeMembrane(eval);
var {wrapper: eval1, gate: gate1} = c1;
var badCode = ... get bad code  ...
var result = eval1(badCode);
... use result ...
gate1.revoke();
... compartment gone  ...
```



Backup slides