

Module Primordials

Man of Straw

Kris Kowal

For ECMA TC-39, January 27-28 2010

* grammar

* primordials

* why

Grammar


```
import id [from pkg]opt [with scope]opt
```



```
(import id)
```

```
require(id)
```



```
(import id from pkg)
```

```
require(id, pkg)
```



```
(import id with scope)
```

```
load(id) (scope)
```

DESUGAR: CALL A MAKER FROM THE CURRENT PACKAGE


```
(import id from pkg with scope)
```

```
load(id, pkg)(scope)
```


Primordials


```
new Context ( )
```

CREATE A NEW EXECUTION CONTEXT; THIS IS THE INTERFACE


```
let RhinoContext =  
    Packages.org.mozilla.javascript.Context;  
let rhinoContext = new RhinoContext();  
rhinoContext.initStandardObjects();
```



```
new Module(  
    text,  
    fileNameopt,  
    lineNoopt  
)
```



```
new Function(  
    ...args,  
    text,  
    fileNameopt,  
    lineNoopt  
)
```

```
new Module(  
    ...args,  
    text,  
    fileNameopt,  
    lineNoopt  
)
```



```
new Module(  
  text,  
  fileNameopt,  
  lineNoopt  
)
```



```
module(  
  scopeopt,  
  loadopt,  
  requireopt,  
  contextopt  
)
```



```
let f = new Function("x", "y", "return x + y");  
let z = f(10, 20);
```

```
let m = new Module("return x + y", "m", 1);  
let z = f({"x": 10, "y": 20});
```



```
let m = new Module("import 'm'");  
m.dependencies[0][0] == "m";
```


array [[
array {	[id, pkg],
string id;	[id, pkg],
any pkg;	[id, pkg]
};	...
] dependencies;]

Roadmap

- * require, exports, module
- * Migration: `<script>`able modules
- * Context, Module, import, export
- * Migration: require, exports, module

Why

- ✱ frozen globals vs.
mutable globals vs.
isolated worlds
- ✱ singleton modules vs.
e-makers
- ✱ require, exports, module vs.
import, export
- ✱ return promises vs.
alter exports object

