

Callable Class Constructor proposal – status update

Allen Wirfs-Brock

Callable Class constructors

- Yehuda present informal proposal at Sept. 24 meeting.
 - <https://github.com/tc39/tc39-notes/blob/master/es7/2015-09/sept-24.md#56-proposal-call-constructor>
 - Approved for stage 1
- In October Allen posted a formal proposal with spec. language
 - <https://github.com/tc39/ecma262/blob/master/workingdocs/callconstructor.md>
- At the time we thought it was ready for stage 2 and ES2016

Refresher

Instead of this:

```
// these functions are defined in the appendix
import { initializeDate, ToDateString } from './date-implementation';

export function Date(...args) {
  if (new.target) {
    // [[Construct]] branch
    initializeDate(this, ...args);
  } else {
    // [[Call]] branch
    return ToDateString(clockGetTime());
  }
}
```

Refresher

We want to be able to say:

```
import { initializeDate, ToDateString } from './date-implementation';
```

```
class Date {  
  constructor(...args) {      //activated by: new Date()  
    initializeDate(super(), ...args);  
  }  
  
  call constructor() {  
    return ToDateString(clockGetTime()); //activated by: Date()  
  }  
}
```

Refresher

- The “call constructor” is not a *property* of the constructor function.
- It just provides an alternative *FunctionBody* that is used when the constructor is invoked via `[[Call]]` MOP operation.
- One function – two bodies

Interesting feed back via twitter and other channels after publicizing proposal

- Some people strongly want the default call action to throw (as it current does)
- Some people strongly want `[[Call]]` and `[[Construct]]` to do the same thing.
 - Or to only allow `[[Call]]`
 - Inability to `[[Call]]` a class constructor causes some people to not use class declarations

More feedback

- Many people found the one-function with two bodies approach very confusing
 - They assume that that the call function is a value in a visible property of the constructor
 - They assume that the call function is inherited by subclass constructors.

What to do?

- Need a way to enable calling class constructors
- Call does an implicit new is the most common request, so it should be easy.
- Perhaps two bodies is too confusing.

An alternative approach

- The 99% case is using a constructor as a factory function:

```
//Most common case:  
class RegExp{  
  
    factory constructor(pattern, flags)  
    ...  
}  
}
```

Use single body and new.target for Date-like case

```
import { initializeDate, ToDateString } from './date-implementation';
```

```
Export class Date {
```

```
  factory constructor(...args) {           //activated by: new Date() and Date();
    if (new.target) {
      // [[Construct]] branch
      initializeDate(this, ...args);
    } else {
      // [[Call]] branch
      //ignore this value
      return ToDateString(clockGetTime());
    }
  }
}
```

Advantages

- `new.target` test only required when `[[Construct]]` and `[[Call]]` have differing behavior
- Eliminates one function/two bodies confusion.
- “factory” syntax reflects the “99%” use case